Machine Learning

R20

(20A05602T)

Unit-I

Chapter 1 Introduction to Machine Learning

INTRODUCTION

The Rapid development in the area of machine learning has triggered a question in everyone's mind – can machines learn better than human? To find its answer, the first step would be to understand what learning is from a human perspective. Then, more light can be shed on what machine learning is. In the end, we need to know whether machine learning has already surpassed or has the potential to surpass human learning in every facet of life.
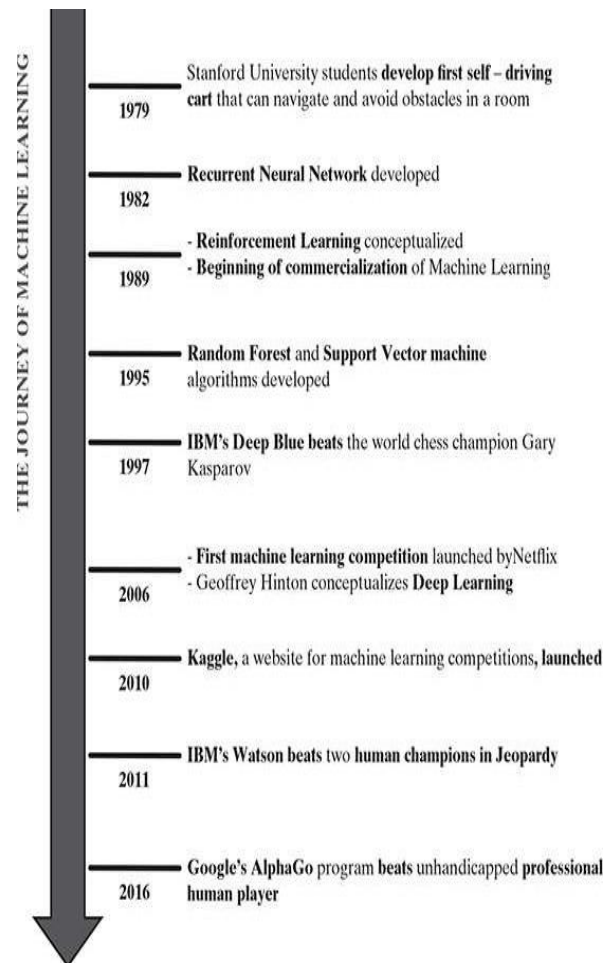
FIG. 1.1 Evolution of machine learning

WHAT IS HUMAN LEARNING?

In cognitive science, learning is typically referred to as the process of gaining information through observation. And why do we need to learn? In our daily life, we need to carry out multiple activities. It may be a task as simple as walking down the street or doing the homework. Or it may be some complex task like deciding the angle in which a rocket should be launched so that it can have a particular trajectory. To do a task in a proper way, we need to have prior information on one or more things related to the task. Also, as we keep learning more or in other words acquiring more information, the efficiency in doing the tasks keep improving. For example, with more knowledge, the ability to do homework with less number of mistakes increases. In the same way, information from past rocket launches helps in taking the right precautions and makes more successful rocket launch. Thus, with more learning, tasks can be performed more efficiently.

TYPES OF HUMAN LEARNING

Thinking intuitively, human learning happens in one of the three ways – (1) either somebody who is an expert in the subject directly teaches us, (2) we build our own notion indirectly based on what we have learnt from the expert in the past, or (3) we do it ourselves, may be after multiple attempts, some being unsuccessful. The first type of learning, we may call, falls under the category of learning directly under expert guidance, the second type falls under learning guided by knowledge gained from experts and the third type is learning by self or self- learning. Let's look at each of these types deeply using real-life examples and try to understand what they mean.

Learning under expert guidance

An infant may inculcate certain traits and characteristics, learning straight from its guardians. He calls his hand, a 'hand', because that is the information he gets from his parents. The sky is 'blue' to him because that is what his parents have taught him. We say that the baby 'learns' things from his parents.

The next phase of life is when the baby starts going to school. In school, he starts with basic familiarization of alphabets and digits. Then the baby learns how to form words from the alphabets and numbers from the digits. Slowly more complex learning happens in the form of sentences, paragraphs, complex mathematics, science, etc. The baby is able to learn all these things from his teacher who already has knowledge on these areas.

Then starts higher studies where the person learns about more complex, application-oriented skills. Engineering students get skilled in one of the disciplines like civil, computer science, electrical, mechanical, etc. medical students learn about anatomy, physiology, pharmacology, etc. There are some experts, in general the teachers, in the respective field who have in-depth subject matter knowledge, who help the students in learning these skills.

Then the person starts working as a professional in some field. Though he might have gone through enough theoretical learning in the respective field, he still needs to learn more about the hands-on application of the knowledge that he has acquired. The professional mentors, by virtue

of the knowledge that they have gained through years of hands-on experience, help all new comers in the field to learn on-job.

In all phases of life of a human being, there is an element of guided learning. This learning is imparted by someone, purely because of the fact that he/she has already gathered the knowledge by virtue of his/her experience in that field. So guided learning is the process of gaining information from a person having sufficient knowledge due to the past experience.

Learning guided by knowledge gained from experts

An essential part of learning also happens with the knowledge which has been impacted by teacher or mentor at some point of time in some other form/context. For example, a baby can group together all objects of same color even if his parents have not specifically taught him to do so. He is able to do so because at some point of time or other his parents have told him which color is blue, which is red, which is green, etc. A grown-up kid can select one odd word from a set of words because it is a verb and other words being all nouns. He could do this because of his ability to label the words as verbs or nouns, taught by his English teacher long back. In a professional role, a person is able to make out to which customers he should market a campaign from the knowledge about preference that was given by his boss long back.

In all these situations, there is no direct learning. It is some past information shared on some different context, which is used as a learning to make decisions.

Learning by self

In many situations, humans are left to learn on their own. A classic example is a baby learning to walk through obstacles. He bumps on to obstacles and falls down multiple times till he learns that whenever there is an obstacle, he needs to cross over it. He faces the same challenge while learning to ride a cycle as a kid or drive a car as an adult. Not all things are taught by others. A lot of things need to be learnt only from mistakes made in the past. We tend to form a check list on things that we should do, and things that we should not do, based on our experiences.

<u>**WHAT IS MACHINE LEARNING?**</u>

Before answering the question 'What is machine learning?' more fundamental questions that peep into one's mind are Do machines really learn? If so, how do they learn? Which problem can we consider as a well-posed learning problem? What are the important features that are required to well-define a learning problem?

At the onset, it is important to formalize the definition of machine learning. This will itself address the first question, i.e. if machines really learn. There are multiple ways to define machine learning. But the one which is perhaps most relevant, concise and accepted universally is the one stated by Tom M. Mitchell, Professor of Machine Learning Department, School of Computer Science, Carnegie Mellon University. Tom M. Mitchell has defined machine learning as

'A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.'

What this essentially means is that a machine can be considered to learn if it is able to gather experience by doing a certain task and improve its performance in doing the similar tasks in the future. When we talk about past experience, it means past data related to the task. This data is an input to the machine from some source.

In the context of the learning to play checkers, E represents the experience of playing the game, T represents the task of playing checkers and P is the performance measure indicated by the percentage of games won by the player. The same mapping can be applied for any other machine learning problem, for example, image classification problem. In context of image classification, E represents the past data with images having labels or assigned classes (for example whether the image is of a class cat or a class dog or a class elephant etc.), T is the task of assigning class to new, unlabelled images and P is the performance measure indicated by the percentage of images correctly classified.

The first step in any project is defining your problem. Even if the most powerful algorithm is used, the results will be meaningless if the wrong problem is solved.

How do machines learn?

The basic machine learning process can be divided into three parts.

1. Data Input: Past data or information is utilized as a basis for future decision-making
2. Abstraction: The input data is represented in a broader way through the underlying algorithm
3. Generalization: The abstracted representation is generalized to form a framework for making decisions.

Figure 1.2 is a schematic representation of the machine learning process.

FIG. 1.2 Process of machine learning

Let's put the things in perspective of the human learning process and try to understand the machine learning process more clearly. Reason is, in some sense, machine learning process tries to emulate the process in which humans learn to a large extent.

Let's consider the situation of typical process of learning from classroom and books and preparing for the examination. It is a tendency of many students to try and memorize (we often call it 'learn by heart') as many things as possible. This may work well when the scope of learning is not so vast. Also, the kinds of questions which are asked in the examination are pretty much simple and straightforward. The questions can be answered by simply writing the same things which have been memorized. However, as the scope gets broader and the questions asked in the examination gets more complex, the strategy of memorizing doesn't work well. The number of topics may get too vast for a student to memorize. Also, the capability of memorizing varies from student to student. Together with that, since the questions get more complex, a direct reproduction of the things memorized may not help. The situation continues to get worse as the student graduates to higher classes.

So, what we see in the case of human learning is that just by great memorizing and perfect recall, i.e. just based on knowledge input, students can do well in the examinations only till a certain stage. Beyond that, a better learning strategy needs to be adopted:

1. To be able to deal with the vastness of the subject matter and the related issues in memorizing it

2. To be able to answer questions where a direct answer has not been learnt

A good option is to figure out the key points or ideas amongst a vast pool of knowledge. This helps in creating an outline of topics and a conceptual mapping of those outlined topics with the entire knowledge pool. For example, a broad pool of knowledge may consist of all living animals and their characteristics such as whether they live in land or water, whether they lay eggs, whether they have scales or fur or none, etc. It is a difficult task for any student to memorize the characteristics of all living animals – no matter how much photographic memory he/she may possess. It is better to draw a notion about the basic groups that all living animals belong to and the characteristics which define each of the basic groups. The basic groups of animals are invertebrates and vertebrates. Vertebrates are further grouped as mammals, reptiles, amphibians, fishes, and birds. Here, we have mapped animal groups and their salient characteristics.

1.   Invertebrate: Do not have backbones and skeletons

2.   Vertebrate

- Fishes: Always live in water and lay eggs
- Amphibians: Semi-aquatic i.e. may live in water or land; smooth skin; lay eggs
- Reptiles: Semi-aquatic like amphibians; scaly skin; lay eggs; cold-blooded
- Birds: Can fly; lay eggs; warm-blooded
- Mammals: Have hair or fur; have milk to feed their young; warm-blooded

This makes it easier to memorize as the scope now reduces to know the animal groups that the animals belong to. Rest of the answers about the characteristics of the animals may be derived from the concept of mapping animal groups and their characteristics.

Moving to the machine learning paradigm, the vast pool of knowledge is available from the data input.

However, rather than using it in entirety, a concept map, much in line with the animal group to characteristic mapping explained above, is drawn from the input data. This is nothing but knowledge abstraction as performed by the machine. In the end, the abstracted mapping from the input data can be applied to make critical conclusions. For example, if the group of an animal is given, understanding of the characteristics can be automatically made. Reversely, if the characteristic of an unknown animal is given, a definite conclusion can be made about the animal group it belongs to. This is generalization in context of machine learning.

Abstraction

During the machine learning process, knowledge is fed in the form of input data. However, the data cannot be used in the original shape and form. As we saw in the example above, abstraction helps in deriving a conceptual map based on the input data. This map, or a model as it is known in the machine learning paradigm, is summarized knowledge representation of the raw data. The model may be in any one of the following forms Computational blocks like if/else rules Mathematical equations Specific data structures like trees or graphs Logical groupings of similar observations. The choice of the model used to solve a specific learning problem is a human task. The decision related to the choice of model is taken based on multiple aspects, some of which are listed below:

The type of problem to be solved: Whether the problem is related to forecast or prediction, analysis of trend, understanding the different segments or groups of objects, etc.

Nature of the input data: How exhaustive the input data is, whether the data has no values for many fields, the data types, etc.

Domain of the problem: If the problem is in a business critical domain with a high rate of data input and need for immediate inference, e.g. fraud detection problem in banking domain.

Once the model is chosen, the next task is to fit the model based on the input data. Let's understand this with an example. In a case where the model is represented by a mathematical equation, say '$y = c1 + c2x$' (the model is known as simple linear regression which we will study in a later chapter), based on the input data, we have to find out the values of c1 and c2. Otherwise, the equation (or the model) is of no use. So, fitting the model, in this case, means finding the values of the unknown coefficients or constants of the equation or the model. This process of fitting the model based on the input data is known as training. Also, the input data based on which the model is being finalized is known as training data.

Generalization:

The first part of machine learning process is abstraction

i.e. abstract the knowledge which comes as input data in the form of a model. However, this abstraction process, or more popularly training the model, is just one part of machine learning. The other key part is to tune up the abstracted knowledge to a form which can be used to take future decisions. This is achieved as a part of generalization. This part is quite difficult to achieve. This is because the model is trained based on a finite set of data, which may possess a limited set of characteristics. But when we want to apply the model to take decision on a set of unknown data, usually termed as test data, we may encounter two problems:

1.   The trained model is aligned with the training data too much, hence may not portray the actual trend.

2.   The test data possess certain characteristics apparently unknown to the training data.

Hence, a precise approach of decision-making will not work. An approximate or heuristic approach, much like gut-feeling-based decision-making in human beings, has to be adopted. This approach has the risk of not making a correct decision – quite obviously because certain assumptions that are made may not be true in reality.

But just like machines, same mistakes can be made by humans too when a decision is made based on intuition or gut-feeling – in a situation where exact reason-based decision-making is not possible.

Well-posed learning problem:

For defining a new problem, which can be solved using machine learning, a simple framework, highlighted below, can be used. This framework also helps in deciding whether the problem is a right candidate to be solved using machine learning. The framework involves answering three questions:

1.   What is the problem?

2.   Why does the problem need to be solved?

3.   How to solve the problem?

Step 1: What is the Problem?

A number of information should be collected to know what is the problem.

Informal description of the problem, e.g. I need a program that will prompt the next word as and when I type a word.

Formalism: Use Tom Mitchell's machine learning formalism stated above to define the T, P, and E for the problem.

 For example:

Task (T): Prompt the next word when I type a word. Experience (E): A corpus of commonly used English words and phrases.

Performance (P): The number of correct words prompted considered as a percentage (which in machine learning paradigm is known as learning accuracy).

Assumptions - Create a list of assumptions about the problem.

Similar problems:

What other problems have you seen or can you think of that are similar to the problem that you are trying to solve?

Step 2: Why does the problem need to be solved?

Motivation:

What is the motivation for solving the problem? What requirement will it fulfil?

For example, does this problem solve any long- standing business issue like finding out potentially fraudulent transactions? Or the purpose is more trivial like trying to suggest some movies for upcoming weekend.

Solution benefits:

Consider the benefits of solving the problem. What capabilities does it enable?

It is important to clearly understand the benefits of solving the problem. These benefits can be articulated to sell the project.

Solution use

How will the solution to the problem be used and the life time of the solution is expected to have?

Step 3: How would I solve the problem?

Try to explore how to solve the problem manually.

Detail out step-by-step data collection, data preparation, and program design to solve the problem. Collect all these details and update the previous sections of the problem definition, especially the assumptions.

Summary

Step 1: What is the problem? Describe the problem informally and formally and list assumptions and similar problems.

Step 2: Why does the problem need to be solved? List the motivation for solving the problem, the benefits that the solution will provide and how the solution will be used.

Step 3: How would I solve the problem? Describe how the problem would be solved manually to flush domain knowledge.
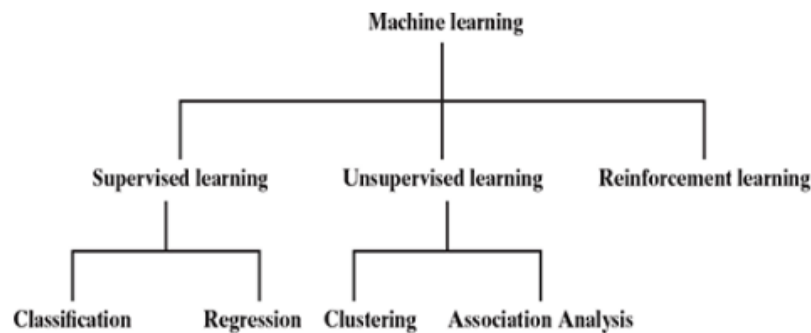
TYPES OF MACHINE LEARNING

FIG. 1.3 Types of machine learning

As highlighted in Figure 1.3, Machine learning can be classified into three broad categories:

1.   Supervised learning – Also called predictive learning. A machine predicts the class of unknown objects based on prior class- related information of similar objects.

2.   Unsupervised learning – Also called descriptive learning. A machine finds patterns in unknown objects by grouping similar objects together.

3.   Reinforcement learning – A machine learns to act on its own to achieve the given goals.

Supervised learning:

The major motivation of supervised learning is to learn from past information. So what kind of past information does the machine need for supervised learning? It is the information about the task which the machine has to execute. In context of the definition of machine learning, this past information is the experience. Let's try to understand it with an example.

Say a machine is getting images of different objects as input and the task is to segregate the images by either shape or color of the object. If it is by shape, the images which are of round-shaped objects need to be separated from images of triangular-shaped objects, etc. If the segregation needs to happen based on color, images of blue objects need to be separated from images of green objects. But how can the machine know what is round shape, or triangular shape? Same way, how can the machine distinguish image of an object based on whether it is blue or green in color? A machine is very much like a little child whose parents or adults need to guide him with the basic information on shape and color before he can start doing the task. A

machine needs the basic information to be provided to it. This basic input, or the experience in the paradigm of machine learning, is given in the form of training data. Training data is the past information on a specific task. In context of the image segregation problem, training data will have past data on different aspects or features on a number of images, along with a tag on whether the image is round or triangular, or blue or green in color. The tag is called 'label' and we say that the training data is labeled in case of supervised learning.
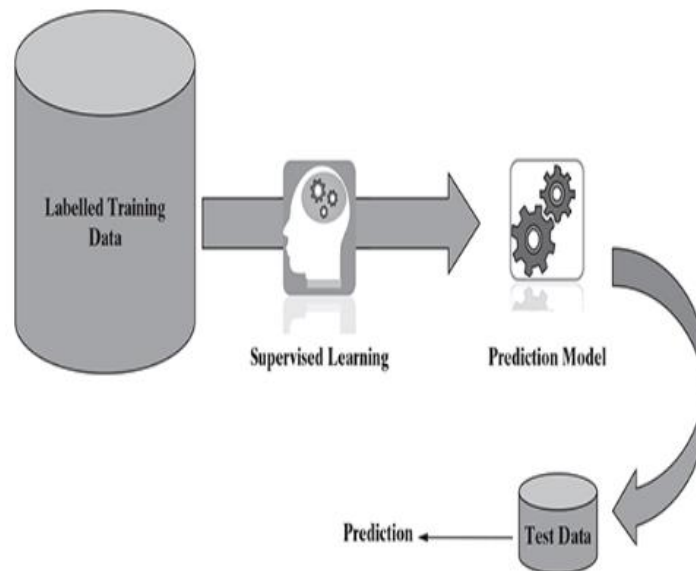

FIG. 1.4 Supervised learning

Figure 1.4 is a simple depiction of the supervised learning process. Labeled training data containing past information comes as an input. Based on the training data, the machine builds a predictive model that can be used on test data to assign a label for each record in the test data.

Some examples of supervised learning are

- Predicting the results of a game
- Predicting whether a tumor is malignant or benign
- Predicting the price of domains like real estate, stocks, etc.
- Classifying texts such as classifying a set of emails as spam or non- spam

Now, let's consider two of the above examples, say 'predicting whether a tumour is malignant or benign' and 'predicting price of domains such as real estate'. Are these two problems same in nature? The answer is 'no'.

Though both of them are prediction problems, in one case we are trying to predict which category or class an unknown data belongs to whereas in the other case we are trying to predict an absolute value and not a class. When we are trying to predict a categorical or nominal variable, the problem is known as a classification problem. Whereas when we are trying to predict a real- valued variable, the problem falls under the category of regression.

Let's try to understand these two areas of supervised learning, i.e. classification and regression in more details.

a) Classification

Let's discuss how to segregate the images of objects based on the shape . If the image is of a round object, it is put under one category, while if the image is of a triangular object, it is put under another category. In which category the machine should put an image of unknown category, also called a test data in machine learning parlance, depends on the information it gets from the past data, which we have called as training data. Since the training data has a label or category defined for each and every image, the machine has to map a new image or test data to a set of images to which it is similar to and assign the same label or category to the test data.

 So we observe that the whole problem revolves around assigning a label or category or class to a test data based on the label or category or class information that is imparted by the training data. Since the target objective is to assign a class label, this type of problem as classification problem. Figure 1.5 depicts the typical process of classification.
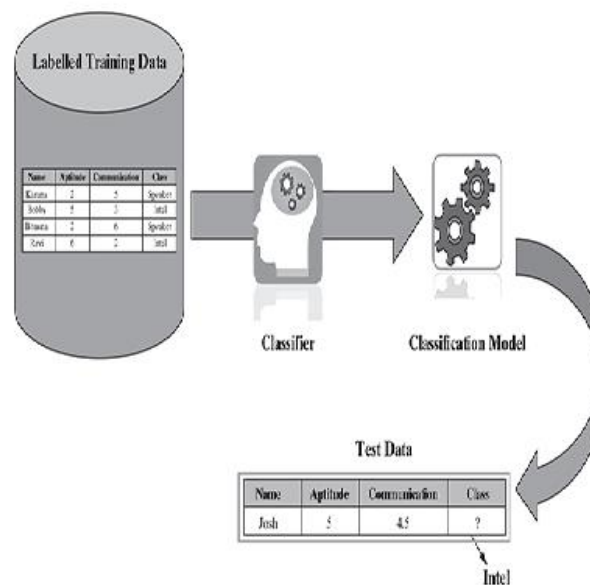

FIG. 1.5 Classification

There are number of popular machine learning algorithms which help in solving classification problems. To name a few, Naïve Bayes, Decision tree, and k- Nearest Neighbour algorithms are adopted by many machine learning practitioners.

A critical classification problem in context of banking domain is identifying potential fraudulent transactions. Since there are millions of transactions which have to be scrutinized and assured whether it might be a fraud transaction, it is not possible for any human being to carry out this task. Machine learning is effectively leveraged to do this task and this is a classic case of

classification. Based on the past transaction data, specifically the ones labeled as fraudulent, all new incoming transactions are marked or labeled as normal or suspicious. The suspicious transactions are subsequently segregated for a closer review.

In summary, classification is a type of supervised learning where a target feature, which is of type categorical, is predicted for test data based on the information imparted by training data. The target categorical feature is known as class.

Some typical classification problems include:

- Image classification Prediction of disease Win–loss prediction of games
- Prediction of natural calamity like earthquake, flood, etc. Recognition of handwriting

b) Regression

In linear regression, the objective is to predict numerical features like real estate or stock price, temperature, marks in an examination, sales revenue, etc. The underlying predictor variable and the target variable are continuous in nature. In case of linear regression, a straight line relationship is 'fitted' between the predictor variables and the target variables, using the statistical concept of least squares method. As in the case of least squares method, the sum of square of error between actual and predicted values of the target variable is tried to be minimized. In case of simple linear regression, there is only one predictor variable whereas in case of multiple linear regression, multiple predictor variables can be included in the model.

Let's take the example of yearly budgeting exercise of the sales managers. They have to give sales prediction for the next year based on sales figure of previous years vis-à-vis investment being put in. Obviously, the data related to past as well as the data to be predicted are continuous in nature. In a basic approach, a simple linear regression model can be applied with investment as predictor variable and sales revenue as the target variable.

Figure 1.6 shows a typical simple regression model, where regression line is fitted based on values of target variable with respect to different values of predictor variable. A typical linear regression model can be represented in the form –where 'x' is the predictor variable and 'y' is the target variable.

The input data come from a famous multivariate data set named Iris introduced by the British statistician and biologist Ronald Fisher. The data set consists of 50 samples from each of three species of Iris – Iris setosa, Iris virginica, and Iris versicolor. Four features were measured for each sample – sepal length, sepal width, petal length, and petal width. These features can uniquely discriminate the different species of the flower.

The Iris data set is typically used as a training data for solving the classification problem of predicting the flower species based on feature values. However, we can also demonstrate regression using this data set, by predicting the value of one feature using another feature as predictor. In Figure 1.6, petal length is a predictor variable which, when fitted in the simple linear regression model, helps in predicting the value of the target variable sepal length.
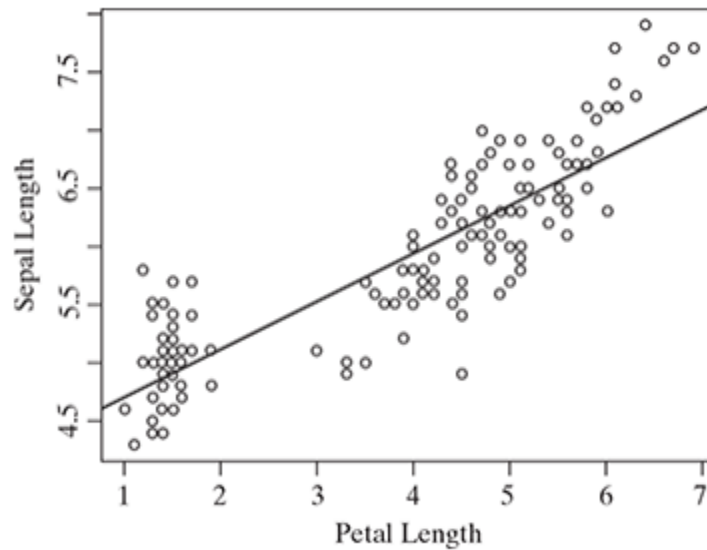
FIG. 1.6 Regression

Typical applications of regression can be seen in

- Demand forecasting in retails
- Sales prediction for managers
- Price prediction in real estate
- Weather forecast
- Skill demand forecast in job market

Unsupervised learning:

Unlike supervised learning, in unsupervised learning, there is no labelled training data to learn from and no prediction to be made. In unsupervised learning, the objective is to take a dataset as input and try to find natural groupings or patterns within the data elements or records. Therefore, unsupervised learning is often termed as descriptive model and the process of unsupervised learning is referred as pattern discovery or knowledge discovery. One critical application of unsupervised learning is customer segmentation.

Clustering is the main type of unsupervised learning. It intends to group or organize similar objects together. For that reason, objects belonging to the same cluster are quite similar to each other while objects belonging to different clusters are quite dissimilar. Hence, the objective of clustering to discover the intrinsic grouping of unlabelled data and form clusters, as depicted in Figure 1.7. Different measures of similarity can be

applied for clustering. One of the most commonly adopted similarity measure is distance. Two data items are considered as a part of the same cluster if the distance between them is less. In the same way, if the distance between the data items is high, the items do not generally belong to the same cluster. This is also known as distance-based clustering. Figure 1.8 depicts the process of clustering at a high level.
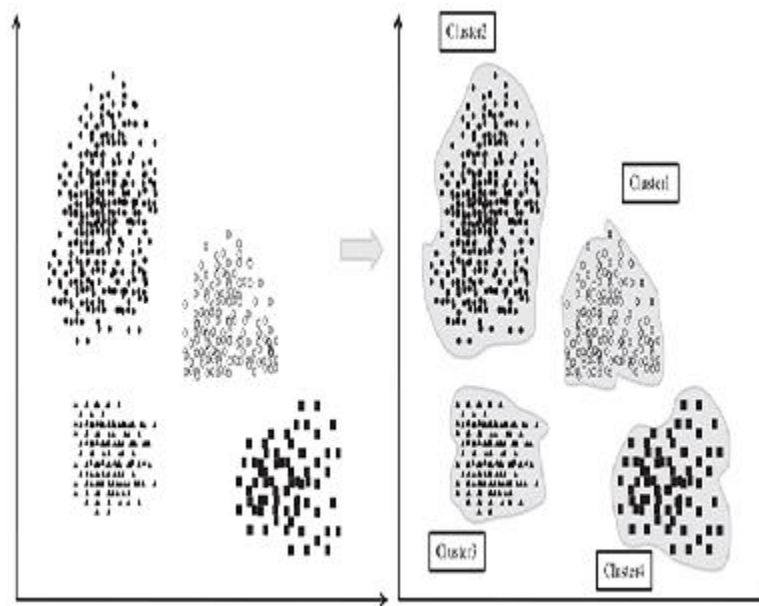

FIG. 1.7 Distance-based clustering

Other than clustering of data and getting a summarized view from it, one more variant of unsupervised learning is association analysis. As a part of association analysis, the association between data elements is identified. Let's try to understand the approach of association analysis in context of one of the most common examples, i.e. market basket analysis as shown in Figure 1.9. From past transaction data in a grocery store, it may be observed that most of the customers who have bought item A, have also bought item B and item C or at least one of them. This means that there is a strong association of the event 'purchase of item A' with the event 'purchase of item B', or 'purchase of item C'. Identifying these sorts of associations is the goal of association analysis. This helps in boosting up sales pipeline, hence a critical input for the sales group.

Critical applications of association analysis include market basket analysis and recommender systems.
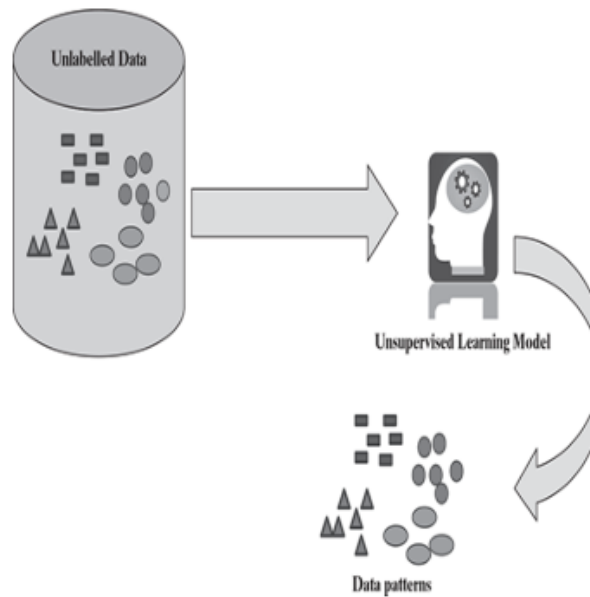
FIG. 1.8 Unsupervised learning

TransID	Items Bought
1	{Butter, Bread}
2	{Diaper, Bread, Milk, Beer}
3	{Milk, Chicken, Beer, Diaper}
4	{Bread, Diaper, Chicken, Beer}
5	{Diaper, Beer, Cookies, Ice cream}
...	...

Market Basket transactions

Frequent itemsets → (Diaper, Beer)

Possible association: Diaper → Beer

FIG. 1.9 Market basket analysis

Reinforcement learning:

We have seen babies learn to walk without any prior knowledge of how to do it. Often we wonder how they really do it. They do it in a relatively simple way.

First they notice somebody else walking around, for example parents or anyone living around. They understand that legs have to be used, one at a time, to take a step. While walking, sometimes they fall down hitting an obstacle, whereas other times they are able to walk smoothly avoiding bumpy obstacles. When they are able to walk overcoming the obstacle, their parents are

elated and appreciate the baby with loud claps / or may be a chocolates. When they fall down while circumventing an obstacle, obviously their parents do not give claps or chocolates. Slowly a time comes when the babies learn from mistakes and are able to walk with much ease.

In the same way, machines often learn to do tasks autonomously. Let's try to understand in context of the example of the child learning to walk. The action tried to be achieved is walking, the child is the agent and the place with hurdles on which the child is trying to walk resembles the environment. It tries to improve its performance of doing the task. When a sub-task is accomplished successfully, a reward is given. When a sub-task is not executed correctly, obviously no reward is given. This continues till the machine is able to complete execution of the whole task. This process of learning is known as reinforcement learning. Figure 1.10 captures the high-level process of reinforcement learning.
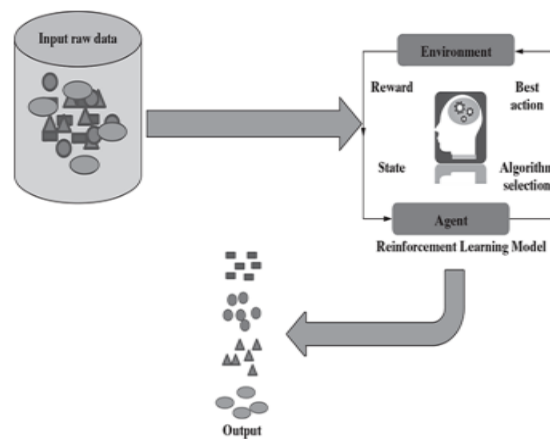

FIG. 1.10 Reinforcement learning

One contemporary example of reinforcement learning is self-driving cars. The critical information which it needs to take care of our speed and speed limit in different road segments, traffic conditions, road conditions, weather conditions, etc. The tasks that have to be taken care of are start/stop, accelerate/decelerate, turn to left / right, etc.

SUPERVISED	UNSUPERVISED	REINFORCEMENT
This type of learning is used when you know how to classify a given data, or in other words classes or labels are available.	This type of learning is used when there is no idea about the class or label of a particular data. The model has to find pattern in the data.	This type of learning is used when there is no idea about the class or label of a particular data. The model has to do the classification – it will get rewarded if the classification is correct, else get punished.
Labelled training data is needed. Model is built based on training data.	Any unknown and unlabelled data set is given to the model as input and records are grouped.	The model learns and updates itself through reward/punishment.
The model performance can be evaluated based on how many misclassifications have been done based on a comparison between predicted and actual values.	Difficult to measure whether the model did something useful or interesting. Homogeneity of records grouped together is the only measure.	Model is evaluated by means of the reward function after it had some time to learn.
There are two types of supervised learning problems – classification and regression.	There are two types of unsupervised learning problems – clustering and association.	No such types.
Simplest one to understand.	More difficult to understand and implement than supervised learning.	Most complex to understand and apply.
Standard algorithms include • Naïve Bayes • k-nearest neighbour (kNN) • Decision tree • Linear regression • Logistic regression • Support Vector Machine SVM), etc.	Standard algorithms are • k-means • Principal Component Analysis (PCA) • Self-organizing map (SOM) • Apriori algorithm • DBSCAN etc.	Standard algorithms are • Q-learning • Sarsa
Practical applications include • Handwriting recognition • Stock market prediction • Disease prediction • Fraud detection, etc.	Practical applications include • Market basket analysis • Recommender systems • Customer segmentation, etc.	Practical applications include • Self-driving cars • Intelligent robots • AlphaGo Zero (the latest version of DeepMind's AI system playing Go)

PROBLES NOT TO BE SOLVED USING MACHINE LEARNING

Machine learning should not be applied to tasks in which humans are very effective or frequent human intervention is needed. For example, air traffic control is a very complex task needing intense human involvement. At the same time, for very simple tasks which can be implemented using traditional programming paradigms, there is no sense of using machine learning. For example, simple rule-driven or formula-based applications like price calculator engine, dispute tracking application, etc. do not need machine learning techniques.

Machine learning should be used only when the business process has some lapses. If the task is already optimized, incorporating machine learning will not serve to justify the return on investment.

For situations where training data is not sufficient, machine learning cannot be used effectively. This is because, with small training data sets, the impact of bad data is exponentially worse. For the quality of prediction or recommendation to be good, the training data should be sizeable.

APPLICATIONS OF MACHINE LEARNING

Wherever there is a substantial amount of past data, machine learning can be used to generate actionable insight from the data. Though machine learning is adopted in multiple forms in every business domain, we have covered below three major domains just to give some idea about what type of actions can be done using machine learning.

Banking and finance

In the banking industry, fraudulent transactions, especially the ones related to credit cards, are extremely prevalent. Since the volumes as well as velocity of the transactions are extremely high, high performance machine learning solutions are implemented by almost all leading banks across the globe. The models work on a real-time basis, i.e. the fraudulent transactions are spotted and prevented right at the time of occurrence.

This helps in avoiding a lot of operational hassles in settling the disputes that customers will otherwise raise against those fraudulent transactions.

Customers of a bank are often offered lucrative proposals by other competitor banks. Proposals like higher bank interest, lower processing charge of loans, zero balance savings accounts, no overdraft penalty, etc. are offered to customers, with the intent that the customer switches over to the competitor bank. Also, sometimes customers get demotivated by the poor quality of services of the banks and shift to competitor banks. Machine learning helps in preventing or at least reducing the customer churn. Both descriptive and predictive learning can be applied for reducing customer churn. Using descriptive learning, the specific pockets of problem, i.e. a specific bank or a specific zone or a specific type of offering like car loan, may be spotted where

maximum churn is happening. Quite obviously, these are troubled areas where further investigation needs to be done to find and fix the root cause. Using predictive learning, the set of vulnerable customers who may leave the bank very soon, can be identified. Proper action can be taken to make sure that the customers stay back.

Insurance

Insurance industry is extremely data intensive. For that reason, machine learning is extensively used in the insurance industry. Two major areas in the insurance industry where machine learning is used are risk prediction during new customer on boarding and claims management. During customer on boarding, based on the past information the risk profile of a new customer needs to be predicted. Based on the quantum of risk predicted, the quote is generated for the prospective customer.

When a customer claim comes for settlement, past information related to historic claims along with the adjustor notes are considered to predict whether there is any possibility of the claim to be fraudulent. Other than the past information related to the specific customer, information related to similar customers, i.e. customer belonging to the same geographical location, age group, ethnic group, etc., are also considered to formulate the model.

Healthcare

 Wearable device data form a rich source for applying machine learning and predict the health conditions of the person real time. In case there is some health issue which is predicted by the learning model, immediately the person is alerted to take preventive action. In case of some extreme problem, doctors or healthcare providers in the vicinity of the person can be alerted. Suppose an elderly person goes for a morning walk in a park close to his house. Suddenly, while walking, his blood pressure shoots up beyond a certain limit, which is tracked by the wearable. The wearable data is sent to a remote server and a machine learning algorithm is constantly analyzing the streaming data. It also has the history of the elderly person and persons of similar age group. The model predicts some fatality unless immediate action is taken. Alert can be sent to the person to immediately stop walking and take rest. Also, doctors and healthcare providers can be alerted to be on standby. Machine learning along with computer vision also plays a crucial role in disease diagnosis from medical imaging.

STATE-OF-THE-ART LANGUAGES/TOOLS IN MACHINE LEARNING

The algorithms related to different machine learning tasks are known to all and can be implemented using any language/platform. It can be implemented using a Java platform or C / C++ language or in .NET. However, there are certain languages and tools which have been

developed with a focus for implementing machine learning. Few of them, which are most widely used, are covered below.

Python

Python is one of the most popular, open source programming languages widely adopted by machine learning community. It was designed by Guido van Rossum and was first released in 1991. The reference implementation of Python, i.e. CPython, is managed by Python Software Foundation, which is a non-profit organization.Python has very strong libraries for advanced mathematical functionalities (NumPy), algorithms and mathematical tools (SciPy) and numerical plotting (matplotlib). Built on these libraries, there is a machine learning library named scikit-learn, which has various classification, regression, and clustering algorithms embedded in it.

R

R is a language for statistical computing and data analysis. It is an open source language, extremely popular in the academic community – especially among statisticians and data miners. R is considered as a variant of S, a GNU project which was developed at Bell Laboratories. Currently, it is supported by the R Foundation for statistical computing.

R is a very simple programming language with a huge set of libraries available for different stages of machine learning. Some of the libraries standing out in terms of popularity are plyr/dplyr (for data transformation), caret ('Classification and Regression Training' for classification), RJava (to facilitate integration with Java), tm (for text mining), ggplot2 (for data visualization).Other than the libraries, certain packages like Shiny and R Markdown have been developed around R to develop interactive web applications, documents and dashboards on R without much effort.

Matlab

MATLAB (matrix laboratory) is licensed commercial software with a robust support for a wide range of numerical computing. MATLAB has a huge user base across industry and academia. MATLAB is developed by Math Works, a company founded in 1984. Being proprietary software, MATLAB is developed much more professionally, tested rigorously, and has comprehensive documentation.

MATLAB also provides extensive support of statistical functions and has a huge number of machine learning algorithms in-built. It also has the ability to scale up for large datasets by parallel processing on clusters and cloud.

SAS

SAS (earlier known as 'Statistical Analysis System') is another licenced commercial software which provides strong support for machine learning functionalities. Developed in C by SAS Institute, SAS had its first release in the year 1976.

SAS is a software suite comprising different components. The basic data management functionalities are embedded in the Base SAS component whereas the other components like SAS/INSIGHT, Enterprise Miner, SAS/STAT, etc. help in specialized functions related to data mining and statistical analysis.

Other languages/tools

There are a host of other languages and tools that also support machine learning functionalities. Owned by IBM, SPSS (originally named as Statistical Package for the Social Sciences) is a popular package supporting specialized data mining and statistical analysis.

Originally popular for statistical analysis in social science (as the name reflects), SPSS is now popular in other fields as well. Released in 2012, Julia is an open source, liberal licence programming language for numerical analysis and computational science. It has baked in all good things of MATLAB, Python, R, and other programming languages used for machine learning for which it is gaining steady attention from machine learning development community. Another big point in favor of Julia is its ability to implement high-performance machine learning algorithms.

ISSUES IN MACHINE LEARNING

Machine learning is a field which is relatively new and still evolving. Also, the level of research and kind of use of machine learning tools and technologies varies drastically from country to country. The laws and regulations, cultural background, emotional maturity of people differ drastically in different countries. All these factors make the use of machine learning and the issues originating out of machine learning usage are quite different.

The biggest fear and issue arising out of machine learning is related to privacy and the breach of it. The primary focus of learning is on analyzing data, both past and current, and coming up with insight from the data. This insight may be related to people and the facts revealed might be private enough to be kept confidential. Also, different people have a different preference when it comes to sharing of information. While some people may be open to sharing some level of information publicly, some other people may not want to share it even to all friends and keep it restricted just to family members.

Classic examples are a birth date (not the day, but the date as a whole), photographs of a dinner date with family, educational background, etc. Some people share them with all in the social platforms like Face book while others do not, or if they do, they may restrict it to friends only. When machine learning algorithms are implemented using those information, inadvertently people may get upset. For example, if there is a learning algorithm to do preference-based customer segmentation and the output of the analysis is used for sending targeted marketing campaigns, it will hurt the emotion of people and actually do more harm than good. In certain countries, such events may result in legal actions to be taken by the people affected.

Even if there is no breach of privacy, there may be situations where actions were taken based on machine learning may create an adverse reaction. Let's take the example of knowledge

discovery exercise done before starting an election campaign. If a specific area reveals an ethnic majority or skewness of a certain demographic factor, and the campaign pitch carries a message keeping that in mind, it might actually upset the voters and cause an adverse result.

So a very critical consideration before applying machine learning is that proper human judgment should be exercised before using any outcome from machine learning. Only then the decision taken will be beneficial and also not result in any adverse impact.

CHAPTER 2

PREPARING TO MODEL

INTRODUCTION

In the last chapter, we got introduced to machine learning. In the beginning, we got a glimpse of the journey of machine learning as an evolving technology. It all started as a proposition from the renowned computer scientist Alan Turing – machines can 'learn' and become artificially intelligent. Gradually, through the next few decades path-breaking innovations came in from Arthur Samuel, Frank Rosenblatt, John Hopfield, Christopher Watkins, Geoffrey Hinton and many other computer scientists. They shaped up concepts of Neural Networks, Recurrent Neural Network, Reinforcement Learning, Deep Learning, etc. which took machine learning to new heights. In parallel, interesting applications of machine learning kept on happening, with organizations like IBM and Google taking a lead. What started with IBM's Deep Blue beating the world chess champion Gary Kasparov, continued with IBM's Watson beating two human champions in a Jeopardy competition.Google also started with a series of innovations applying machine learning. The Google Brain, Sibyl, Waymo, AlphaGo programs – are all extremely advanced applications of machine learning which have taken the technology a few notches up. Now we can see an all-pervasive presence of machine learning technology in all walks of life.

We have also seen the types of human learning and how that, in some ways, can be related to the types of machine learning – supervised, unsupervised, and reinforcement. Supervised learning, as we saw, implies learning from past data, also called training data, which has got known values or classes. Machines can 'learn' or get 'trained' from the past data and assign classes or values to unknown data, termed as test data. This helps in solving problems related to prediction. This is much like human learning through expert guidance as happens for infants from parents or students through teachers.

So, supervised learning in case of machines can be perceived as guided learning from human inputs. Unsupervised machine learning doesn't have labeled data to learn from. It tries to find patterns in unlabelled data. This is much like human beings trying to group together objects of similar shape. This learning is not guided by labeled inputs but uses the knowledge gained from the labels themselves. Last but not the least is reinforcement learning in which machine tries to learn by itself through penalty/ reward mechanism – again pretty much in the same way as human self-learning happens.

Lastly, we saw some of the applications of machine learning in different domains such as banking and finance, insurance, and healthcare. Fraud detection is a critical business case which is implemented in almost all banks across the world and uses machine learning predominantly. Risk prediction for new

customers is a similar critical case in the insurance industry which finds the application of machine learning. In the healthcare sector, disease prediction makes wide use of machine learning, especially in the developed countries.

While development in machine learning technology has been extensive and its implementation has become widespread, to start as a practitioner, we need to gain some basic understanding. We need to understand how to apply the array of tools and technologies available in the machine learning to solve a problem. In fact, that is going to be very specific to the kind of problem that we are trying to solve. If it is a prediction problem, the kind of activities that will be involved is going to be completely different vis-à-vis if it is a problem where we are trying to unfold a pattern in a data without any past knowledge about the data. So how a machine learning project looks like or what are the salient activities that form the core of a machine learning project will depend on whether it is in the area of supervised or unsupervised or reinforcement learning area. However, irrespective of the variation, some foundational knowledge needs to be built before we start with the core machine learning concepts and key algorithms. In this section, we will have a quick look at a few typical machine learning activities and focus on some of the foundational concepts that all practitioners need to gain as pre-requisites before starting their journey in the area of machine learning.

MACHINE LEARNING ACTIVITIES

The first step in machine learning activity starts with data. In case of supervised learning, it is the labelled training data set followed by test data which is not labelled. In case of unsupervised learning, there is no question of labelled data but the task is to find patterns in the input data. A thorough review and exploration of the data is needed to understand the type of the data, the quality of the data and relationship between the different data elements. Based on that, multiple pre-processing activities may need to be done on the input data before we can go ahead with core machine learning activities.

 Following are the typical preparation activities done once the input data comes into the machine learning system:

Understand the type of data in the given input data set. Explore the data to understand the nature and quality.

Explore the relationships amongst the data elements, e.g. inter- feature relationship.

Find potential issues in data.

Do the necessary remediation, e.g. impute missing data values, etc., if needed.

Apply pre-processing steps, as necessary.

Once the data is prepared for modelling, then the learning tasks start off. As a part of it, do the following activities:

The input data is first divided into parts – the training data and the test data (called holdout). This step is applicable for supervised learning only.

Consider different models or learning algorithms for selection.

Train the model based on the training data for supervised learning problem and apply to unknown data. Directly apply the chosen unsupervised model on the input data for unsupervised learning problem.

After the model is selected, trained (for supervised learning), and applied on input data, the performance of the model is evaluated. Based on options available, specific actions can be taken to improve the performance of the model, if possible.
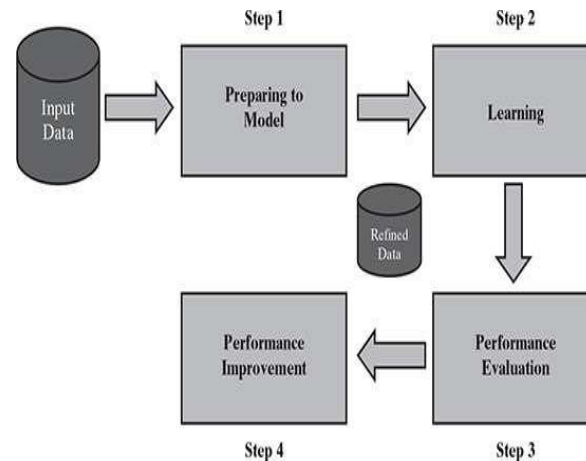

Table 2.1 contains a summary of steps and activities involved:

Step #	Step Name	Activities Involved

Step 1	Preparing to Model	• Understand the type of data in the given input data set
		• Explore the data to understand data quality
		• Explore the relationships amongst the data elements, e.g. inter-feature relationship
		• Find potential issues in data
		• Remediate data, if needed
		• Apply following pre-processing steps, as necessary:
		  ✓ Dimensionality reduction
		  ✓ Feature subset selection

Step 2	Learning	• Data partitioning/holdout • Model selection • Cross-validation
Step 3	Performance evaluation	• Examine the model performance, e.g. confusion matrix in case of classification • Visualize performance trade-offs using ROC curves
Step 4	Performance improvement	• Tuning the model • Ensembling • Bagging • Boosting

In this chapter, we will cover the first part, i.e. preparing to model.

BASIC TYPES OF DATA IN MACHINE LEARNING

Before starting with types of data, let's first understand what a data set is and what are the elements of a data set. A data set is a collection of related information or records. The information may be on some entity or some subject area. For example (Fig. 2.2), we may have a data set on students in which each record consists of information about a specific student. Again, we can have a data set on student performance which has records providing performance, i.e. marks on the individual subjects.

Each row of a data set is called a record. Each data set also has multiple attributes, each of which gives information on a specific characteristic. For example, in the data set on students, there are four attributes namely Roll Number, Name, Gender, and Age, each of which understandably is a specific characteristic about the student entity. Attributes can also be termed as feature, variable, dimension or field. Both the data sets, Student and Student Performance, are having four features or dimensions; hence they are told to have four- dimensional data space. A row or record represents a point in the four-dimensional data space as each row has specific values for each of the four attributes or features. Value of an attribute, quite understandably, may vary from record to record. For example, if we refer to the first two records in the Student data set, the value of attributes Name, Gender, and Age are different (Fig.2.3).

Student data set:

Roll Number	Name	Gender	Age
129/011	Mihir Karmarkar	M	14
129/012	Geeta Iyer	F	15
129/013	Chanda Bose	F	14
129/014	Sreenu Subramanian	M	14
129/015	Pallav Gupta	M	16
129/016	Gajanan Sharma	M	15

Student performance data set:

Roll Number	Maths	Science	Percentage
129/011	89	45	89.33%
129/012	89	47	90.67%
129/013	68	29	64.67%
129/014	83	38	80.67%
129/015	57	23	53.33%
129/016	78	35	75.33%

FIG. 2.2 Examples of data set

Roll Number	Name	Gender	Age
129/011	Mihir Karmarkar	M	14
129/012	Geeta Iyer	F	15

FIG. 2.3 Data set records and attributes

Now that a context of data sets is given, let's try to understand the different types of data that we generally come across in machine learning problems. Data can broadly be divided into following two types:

1.   Qualitative data

2.   Quantitative data

Qualitative data provides information about the quality of an object or information which cannot be measured. For example, if we consider the quality of performance of students in terms of 'Good', 'Average', and 'Poor', it falls under the category of qualitative data. Also, name or roll number of students are information that cannot be measured using some scale of measurement. So they would fall under qualitative data. Qualitative data is also called categorical data.

Qualitative data can be further subdivided into two types as follows:

1.   Nominal data

2.   Ordinal data

Nominal data is one which has no numeric value, but a named value. It is used for assigning named values to attributes. Nominal values cannot be quantified.

Examples of nominal data are

1.   Blood group: A, B, O, AB, etc.

2.   Nationality: Indian, American, British, etc.

3.   Gender: Male, Female, Other

Note:

A special case of nominal data is when only two labels are possible, e.g. pass/fail as a result of an examination. This sub-type of nominal data is called 'dichotomous'.

It is obvious, mathematical operations such as addition, subtraction, multiplication, etc. cannot be performed on nominal data. For that reason, statistical functions such as mean, variance, etc. can also not be applied on nominal data. However, a basic count is possible. So mode, i.e. most frequently occurring value, can be identified for nominal data.

Ordinal data, in addition to possessing the properties of nominal data, can also be naturally ordered. This means ordinal data also assigns named values to attributes but unlike nominal data, they can be arranged in a sequence of increasing or decreasing value so that we can say whether a value is better than or greater than another value. Examples of ordinal data are

1.   Customer satisfaction: 'Very Happy', 'Happy', 'Unhappy', etc.

2.   Grades: A, B, C, etc.

3.   Hardness of Metal: 'Very Hard', 'Hard', 'Soft', etc.

Like nominal data, basic counting is possible for ordinal data. Hence, the mode can be identified. Since ordering is possible in case of ordinal data, median, and quartiles can be identified in addition. Mean can still not be calculated.

Quantitative data relates to information about the quantity of an object – hence it can be measured. For example, if we consider the attribute 'marks', it can be measured using a scale of measurement. Quantitative data is also termed as numeric data. There are two types of quantitative data:

1.   Interval data

2.   Ratio data

Interval data is numeric data for which not only the order is known, but the exact difference between values is also known. An ideal example of interval data is Celsius temperature. The difference between each value remains the same in Celsius temperature. For example, the difference between 12°C and 18°C degrees is measurable and is 6°C as in the case of difference between 15.5°C and 21.5°C. Other examples include date, time, etc.

For interval data, mathematical operations such as addition and subtraction are possible. For that reason, for interval data, the central tendency can be measured by mean, median, or mode. Standard deviation can also be calculated.

However, interval data do not have something called a 'true zero' value. For example, there is nothing called '0 temperature' or 'no temperature'. Hence, only addition and subtraction applies for interval data. The ratio cannot be applied. This means, we can say a temperature of 40°C is equal to the temperature of 20°C + temperature of 20°C. However, we cannot say the temperature of 40°C means it is twice as hot as in temperature of 20°C.

Ratio data represents numeric data for which exact value can be measured. Absolute zero is available for ratio data. Also, these variables can be added, subtracted, multiplied, or divided. The central tendency can be measured by mean, median, or mode and methods of dispersion such as standard deviation. Examples of ratio data include height, weight, age, salary, etc.

Figure 2.4 gives a summarized view of different types of data that we may find in a typical machine learning problem.
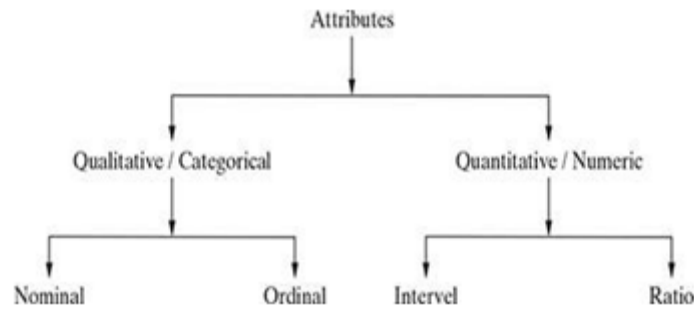
FIG. 2.4 Types of data

Apart from the approach detailed above, attributes can also be categorized into types based on a number of values that can be assigned. The attributes can be either discrete or continuous based on this factor.Discrete attributes can assume a finite or countably infinite number of values. Nominal attributes such as roll number, street number, pin code, etc. can have a finite number of values whereas numeric attributes such as count, rank of students, etc. can have countably infinite values. A special type of discrete attribute which can assume two values only is called binary attribute.

Examples of binary attribute include male/ female, positive/negative, yes/no, etc.

Continuous attributes can assume any possible value which is a real number. Examples of continuous attribute include length, height, weight, price, etc.

Note:

In general, nominal and ordinal attributes are discrete. On the other hand, interval and ratio attributes are continuous, barring a few exceptions, e.g. 'count' attribute.

EXPLORING STRUCTURE OF DATA

By now, we understand that in machine learning, we come across two basic data types – numeric and categorical. With this context in mind, we can delve deeper into understanding a data set. We need to understand that in a data set, which of the attributes are numeric and which are categorical in nature. This is because the approach of exploring numeric data is different than the approach of exploring categorical data. In case of a standard data set, we may have the data dictionary available for reference. Data dictionary is a metadata repository, i.e. the repository of all information related to the structure of each data element contained in the data set. The data dictionary gives detailed information on each of the attributes – the description as well as the data type and other relevant details. In case the data dictionary is not available, we need to use standard library function of the machine learning tool that we are using and get the details. For the time being, let us move ahead with a standard data set from UCI machine learning repository.

The data set that we take as a reference is the Auto MPG data set available in the UCI repository. Figure 2.5 is a snapshot of the first few rows of the data set.

mpg	cylinder	displace-ment	horse-power	weight	accel-eration	model year	origin	car name
18	8	307	130	3504	12	70	1	Chevrolet chev-elle malibu
15	8	350	165	3693	11.5	70	1	Buick skylark 320
18	8	318	150	3436	11	70	1	Plymouth satellite
16	8	304	150	3433	12	70	1	Amc rebel sst
17	8	302	140	3449	10.5	70	1	Ford torino
15	8	429	198	4341	10	70	1	Ford galaxie 500
14	8	454	220	4354	9	70	1	Chevrolet impala
14	8	440	215	4312	8.5	70	1	Plymouth fury iii
14	8	455	225	4425	10	70	1	Pontiac catalina
15	8	390	190	3850	8.5	70	1	Amc achassador dpl
15	8	383	170	3563	10	70	1	Dodge challenger se
14	8	340	160	3609	8	70	1	Plymouth 'cuda 340
15	8	400	150	3761	9.5	70	1	Chevrolet monte carlo
14	8	455	225	3086	10	70	1	Buick estate wagon (sw)
24	4	113	95	2372	15	70	3	Toyota corona mark ii
22	6	198	95	2933	15.5	70	1	Plymouth duster
18	6	199	97	2774	15.5	70	1	Amc hornet

FIG. 2.5 Auto MPG data set

As is quite evident from the data, the attributes such as 'mpg', 'cylinders', 'displacement', 'horsepower', 'weight', 'acceleration', 'model year', and 'origin' are all numeric. Out of these attributes, 'cylinders', 'model year', and 'origin' are discrete in nature as the only finite number of values can be assumed by these attributes. The remaining of the numeric attributes, i.e. 'mpg', 'displacement', 'horsepower', 'weight', and 'acceleration' can assume any real value.

Exploring numerical data

There are two most effective mathematical plots to explore numerical data – box plot and histogram. We will explore all these plots one by one, starting with the most critical one, which is the box plot.

Understanding central tendency

To understand the nature of numeric variables, we can apply the measures of central tendency of data, i.e. mean and median. In statistics, measures of central tendency help us understand the central point of a set of data.

Mean, by definition, is a sum of all data values divided by the count of data elements. For example, mean of a set of observations – 21, 89, 34, 67, and 96 is calculated as below.

$$\text{Mean} = \frac{21 + 89 + 34 + 67 + 96}{5} = 61.4.$$

If the above set of numbers represents marks of 5 students in a class, the mean marks, or the falling in the middle of the range is 61.4.

Median, on contrary, is the value of the element appearing in the middle of an ordered list of data elements. If we consider the above 5 data elements, the ordered list would be – 21, 34, 67, 89, and 96. Since there are 5 data elements, the 3rd element in the ordered list is considered as the median. Hence, the median value of this set of data is 67.

There might be a natural curiosity to understand why two measures of central tendency are reviewed. The reason is mean and median are impacted differently by data values appearing at the beginning or at the end of the range. Mean being calculated from the cumulative sum of data values, is impacted if too many data elements are having values closer to the far end of the range, i.e. close to the maximum or minimum values. It is especially sensitive to outliers, i.e. the values which are unusually high or low, compared to the other values.

Mean is likely to get shifted drastically even due to the presence of a small number of outliers. If we observe that for certain attributes the deviation between values of mean and median are quite high, we should investigate those attributes further and try to find out the root cause along with the need for remediation.

So, in the context of the Auto MPG data set, let's try to find out for each of the numeric attributes the values of mean and median. We can also find out if the deviation between these values is large. In Figure 2.6, the comparison between mean and median for all the attributes has been shown. We can see that for the attributes such as 'mpg', 'weight', 'acceleration', and 'model.year' the deviation between mean and median is not significant which means the chance of these attributes having too many outlier values is less.

However, the deviation is significant for the attributes 'cylinders', 'displacement' and 'origin'. So, we need to further drill down and look at some more statistics for these attributes. Also, there is some problem in the values of the attribute 'horsepower' because of which the mean and median calculation is not possible.

	mpg	cylin-ders	dis-place-ment	horse-power	weight	accel-eration	model year	origin
Median	23	4	148.5	?	2804	15.5	76	1
Mean	23.51	5.455	193.4	?	2970	15.57	76.01	1.573
Deviation	2.17	26.67%	23.22%		5.59%	0.45%	0.01%	36.43%
	Low	High	High		Low	Low	Low	High

FIG. 2.6 Mean vs. Median for Auto MPG

With a bit of investigation, we can find out that the problem is occurring because of the 6 data elements, as shown in Figure 2.7, do not have value for the attribute 'horsepower'.

mpg	cylinders	displace-ment	horse-power	weight	accel-eration	model year	origin	car name
25	4	98	?	2046	19	71	1	Ford pinto
21	6	200	?	2875	17	74	1	Ford maverick
40.9	4	85	?	1835	17.3	80	2	Renault lecar deluxe
23.6	4	140	?	2905	14.3	80	1	Ford mustang cobra
34.5	4	100	?	2320	15.8	81	2	Renault 18i
23	4	151	?	3035	20.5	82	1	Amc concord di

FIG. 2.7 Missing values of attribute 'horsepower' in Auto MPG

For that reason, the attribute 'horsepower' is not treated as a numeric. That's why the operations applicable on numeric variables, like mean or median, are failing. So we have to first remediate the missing values of the attribute 'horsepower' before being able to do any kind of exploration. However, we will cover the approach of remediation of missing values a little later.

Understanding data spread

Now that we have explored the central tendency of the different numeric attributes, we have a clear idea of which attributes have a large deviation between mean and median. Let's look closely at those attributes. To drill down more, we need to look at the entire range of values of the attributes, though not at the level of data elements as that may be too vast to review manually. So we will take a granular view of the data spread in the form of

1. Dispersion of data
2. Position of the different data values
3. Measuring data dispersion

Consider the data values of two attributes

1.   Attribute 1 values : 44, 46, 48, 45, and 47

2.   Attribute 2 values : 34, 46, 59, 39, and 52

Both the set of values have a mean and median of 46.

However, the first set of values that is of attribute 1 is more concentrated or clustered around the mean/median value whereas the second set of values of attribute 2 is quite spread out or dispersed. To measure the extent of dispersion of a data, or to find out how much the different values of a data are spread out, the variance of the data is measured. The variance of a data is measured using the formula given below:

$$\text{Variance } (x) = \frac{\sum_{i=1}^{n} x_i^2}{n} - \left(\frac{\sum_{i=1}^{n} x_i}{n}\right)^2, \text{ where } x \text{ is the}$$

variable or attribute whose variance is to be measured and n is the number of observations or values of variable x.

Standard deviation of a data is measured as follows:

$$\text{Standard deviation } (x) = \sqrt{\text{Variance } (x)}$$

Larger value of variance or standard deviation indicates more dispersion in the data and vice versa. In the above example, let's calculate the variance of attribute 1 and that of attribute 2. For attribute 1, So it is quite clear from the measure that attribute 1 values are quite concentrated around the mean while attribute 2 values are extremely spread out. Since this data was small, a visual inspection and understanding were possible and that matches with the measured value.

$$\text{Variance} = \frac{\sum_{i=1}^{n} x_i^2}{n} - \left(\frac{\sum_{i=1}^{n} x_i}{n}\right)^2$$

$$= \frac{44^2 + 46^2 + 48^2 + 45^2 + 47^2}{5} - \left(\frac{44 + 46 + 48 + 45 + 47}{5}\right)^2$$

$$= \frac{1936 + 2116 + 2304 + 2025 + 2209}{5} - \left(\frac{230}{5}\right)^2 = \frac{10590}{5} - (46)^2 = 2$$

For attribute 2,

$$\text{Variance} = \frac{\sum_{i=1}^{n} x_i^2}{n} - \left(\frac{\sum_{i=1}^{n} x_i}{n}\right)^2$$

$$= \frac{34^2 + 46^2 + 59^2 + 39^2 + 52^2}{5} - \left(\frac{34 + 46 + 59 + 39 + 52}{5}\right)^2$$

$$= \frac{1156 + 2116 + 3481 + 1521 + 2704}{5} - \left(\frac{230}{5}\right)^2 = \frac{10978}{5} - (46)^2 = 79.6$$

Measuring data value position

When the data values of an attribute are arranged in an increasing order, we have seen earlier that median gives the central data value, which divides the entire data set into two halves. Similarly, if the first half of the data is divided into two halves so that each half consists of one- quarter of the data set, then that median of the first half is known as first quartile or Q1. In the same way, if the second half of the data is divided into two halves, then that median of the second half is known as third quartile or Q3. The overall median is also known as second quartile or Q2. So, any data set has five values -minimum, first quartile (Q1), median (Q2), third quartile (Q3), and maximum.

Let's review these values for the attributes 'cylinders', 'displacement', and 'origin'. Figure 2.8 captures a summary of the range of statistics for the attributes. If we take the example of the attribute 'displacement', we can see that the difference between minimum value and Q1 is 36.2 and the difference between Q1 and median is

On the contrary, the difference between median and Q3 is 113.5 and Q3 and the maximum value is 193. In other words, the larger values are more spread out than the smaller ones. This helps in understanding why the value of mean is much higher than that of the median for the attribute 'displacement'. Similarly, in case of attribute 'cylinders', we can observe that the difference between minimum value and median is 1 whereas the difference between median and the maximum value is 4. For the attribute 'origin', the difference between minimum value and median is 0 whereas the difference between median and the maximum value is 2.

	cylinders	displacement	origin
Minimum	3	68	1
Q1	4	104.2	1
Median	4	148.5	1
Q3	8	262	2
Maximum	8	455	3

FIG. 2.8 Attribute value drill-down for Auto MPG

However, we still cannot ascertain whether there is any outlier present in the data. For that, we can better adopt some means to visualize the data. Box plot is an excellent visualization medium for numeric data.

DATA PRE-PROCESSING

Dimensionality reduction

Till the end of the 1990s, very few domains were explored which included data sets with a high number of attributes or features. In general, the data sets used in machine learning used to be in few 10s. However, in the last two decades, there has been a rapid advent of computational biology like genome projects. These projects have produced extremely high-dimensional data sets with 20,000 or more features being very common.

Also, there has been a wide-spread adoption of social networking leading to a need for text classification for customer behavior analysis.

High-dimensional data sets need a high amount of computational space and time. At the same time, not all features are useful – they degrade the performance of machine learning algorithms. Most of the machine learning algorithms performs better if the dimensionality of data set, i.e. the number of features in the data set, is reduced. Dimensionality reduction helps in reducing irrelevance and redundancy in features. Also, it is easier to understand a model if the number of features involved in the learning activity is less.

Dimensionality reduction refers to the techniques of reducing the dimensionality of a data set by creating new attributes by combining the original attributes. The most common approach for dimensionality reduction is known as Principal Component Analysis (PCA). PCA is a statistical technique to convert a set of correlated variables into a set of transformed, uncorrelated variables called principal components. The principal components are a linear combination of the original variables. They are orthogonal to each other. Since principal components are uncorrelated, they capture the maximum amount of variability in the data. However, the only challenge is that the original attributes are lost due to the transformation. Another commonly used technique which is used for dimensionality reduction is Singular Value Decomposition (SVD).

Feature subset selection

Feature subset selection or simply called feature selection, both for supervised as well as unsupervised learning; try to find out the optimal subset of the entire feature set which significantly reduces computational cost without any major impact on the learning accuracy. It may seem that a feature subset may lead to loss of useful information as certain features are going to be excluded from the final set of features used for learning. However, for elimination only features which are not relevant or redundant are selected.

A feature is considered as irrelevant if it plays an insignificant role (or contributes almost no information) in classifying or grouping together a set of data instances. All irrelevant features are eliminated while selecting the final feature subset. A feature is potentially redundant when the information contributed by the feature is more or less same as one or more other features. Among a group of potentially redundant features, a small number of features can be selected as a part of the final feature subset without causing any negative impact to learn model accuracy.

UNIT-2

Modelling and Evaluation

OBJECTIVE OF THE CHAPTER :

The previous chapter gives a comprehensive understanding of the basic data types in the context of machine learning. It also enables a beginner in the field of machine learning to acquire an understanding about the nature and quality of the data by effective exploration of the data set. In this chapter, the objective is to introduce the basic concepts of learning. In this regard, the information shared concerns the aspects of model selection and application. It also imparts knowledge regarding how to judge the effectiveness of the model in doinga specific learning task, supervised or unsupervised, and how to boost the model performance using different tuning parameters.

INTRODUCTION

The learning process of machines may seem quite magical to somebody who is new to machine learning. The thought that a machine is able to think and take intelligent action may be mesmerizing – much like a science fiction or a fantasy story. However, delving a bit deeper helps them realize that it is not as magical as it may seem to be. In fact, it tries to emulate human learning by applying mathematical and statistical formulations. In that sense, both human and machine learning strives to build formulations or mapping basedon a limited number of observations. As introduced in <u>Chapter 1</u>, the basic learning process, irrespective of the fact that the learner is a human or a machine, can bedivided into three parts:

1. Data Input
2. Abstraction
3. Generalization

Though in <u>Chapter 1</u> we have understood these aspectsin details, let's quickly refresh our memory with an example. It's a fictitious situation. The detective department of New City Police has got a tip that in a campaign gathering for the upcoming election, a criminal is going to launch an attack on the main candidate. However, it is not known who the person is and quite obviously the person might use some disguise. The only thing that is for sure is the person is a history- sheeter or a criminal having a long record of serious crime. From the criminal database, a list of such criminals along with their photographs has been collected. Also, the photos taken by security cameras positioned at different places near the gathering are available with the detective department. They have to match the photos from the criminal database with the faces in the gathering to spot the potential attacker. So the main problem here is to spot the face of the criminal based on the match with the photos in the criminal database.

This can be done using human learning where a person from the detective department can scan through each shortlisted photo and try to match that photo with the faces in the gathering. A person having a strong memory can take a glance at the photos of all criminals in one shot and then try to find a face in the gathering which closely resembles one of the

criminal photos that she has viewed. Easy, isn't it? But that is not possible in reality. The number of criminals in the database and hence the count of photos runs in hundreds, if not thousands. So taking a look at all the photos and memorizing them is not possible. Also, an exact match isout of the question as the criminal, in most probability, will come in disguise. The strategy to be taken here is to match the photos in smaller counts and also based on certain salient physical features like the shape of the jaw,the slope of the forehead, the size of the eyes, the structure of the ear, etc. So, the photos from the criminal database form the input data. Based on it, key features can be abstracted. Since human matching for each and every photo may soon lead to a visual as well as mental fatigue, a generalization of abstracted feature-based datais a good way to detect potential criminal faces in the gathering. For example, from the abstracted feature- based data, say it is observed that most of the criminals have a shorter distance between the inner corners of the eyes, a smaller angle between the nose and the corners ofthe mouth, a higher curvature to the upper lip, etc.

Hence, a face in the gathering may be classified as 'potentially criminal' based on whether they match withthese generalized observations. Thus, using the input data, feature-based abstraction could be built and by applying generalization of the abstracted data, human learning could classify the faces as potentially criminal ultimately leading to spotting of the criminal.

The same thing can be done using machine learning too. Unlike human detection, a machine has no subjective baggage, no emotion, no bias due to past experience, and above all no mental fatigue. The machine can also use the same input data, i.e. criminal database photos, apply computational techniques to abstract feature-based concept map from the input data and generalize the same in the form of a classification algorithm to decide whether a face in the gathering is potentially criminal or not.

When we talk about the learning process, abstraction is a significant step as it represents raw input data in a summarized and structured format, such that a meaningful insight is obtained from the data. This structured representation of raw input data to the meaningful pattern is called a **model**. The model mighthave different forms. It might be a mathematical equation, it might be a graph or tree structure, it might be a computational block, etc. The decision regarding which model is to be selected for a specific data set is taken by the learning task, based on the problem to be solved and the type of data. For example, when the problem is related to prediction and the target field is numeric and continuous, the regression model is assigned. The process of assigning a model, and fitting aspecific model to a data set is called model **training.**

Once the model is trained, the raw input data issummarized into an abstracted form.

However, with abstraction, the learner is able to only summarize the knowledge. This knowledge might be stillvery broad-based – consisting of a huge number of feature-based data and inter-relations. To generate actionable insight from such broad-based knowledge is very difficult. This is where generalization comes into play. Generalization searches through the huge set of abstracted knowledge to come up with a small and manageable set of key findings. It is not possible to do anexhaustive search by reviewing each of the abstracted findings one-by-one. A heuristic search is employed, an approach which is also used for human learning (often termed as 'gut-feel'). It is quite obvious that the heuristics sometimes

result in erroneous result. If the outcome is systematically incorrect, the learning is said to have a **bias**.

<table>
<tr><td>

Points to Ponder:

- A machine learning algorithm creates its cognitive capability by building a mathematical formulation or function, known as target function, based on the features in the input data set.
- Just like a child learning things for the first time needs her parents guidance to decide whether she is right or wrong, in machine learning someone has to provide some non-learnable parameters, also called hyper-parameters. Without these human inputs, machine learning algorithms cannot be successful.

</td></tr>
</table>

SELECTING A MODEL

Now that you are familiar with the basic learning process and have understood model abstraction and generalization in that context, let's try to formalize it in context of a motivating example. Continuing the thread of the potential attack during the election campaign, New City Police department has succeeded in foiling the bid to attack the electoral candidate. However, this was a wake-up call for them and they want to take a proactive action to eliminate all criminal activities in the region.

They want to find the pattern of criminal activities in the recent past, i.e. they want to see whether the number of criminal incidents per month has any relation with an average income of the local population, weapon sales, the inflow of immigrants, and other such factors. Therefore, an association between potential causes of disturbance and criminal incidents has to be determined. In other words, the goal or target is to develop a model to infer how the criminal incidents change based on the potential influencing factors mentioned above.

In machine learning paradigm, the potential causes of disturbance, e.g. average income of the local population, weapon sales, the inflow of immigrants, etc. are input variables. They are also called predictors, attributes, features, independent variables, or simply variables. The number of criminal incidents is an output variable (also called response or dependent variable). Input variables can be denoted by X, while individual input variables are represented as $X_1, X_2, X_3, \ldots, X_n$ and output variable by symbol Y. The relationship between X and Y is represented in the general form: $Y = f(X) + e$, where 'f' is the **target function** and 'e' is a random error term.

Note:

Just like a target function with respect to a machine learning model, some other functions which are frequently tracked are

- A **cost function** (also called **error function**) helps to measure the extent to which the model is going wrong in estimating the relationship between *X* and *Y*. In that sense, cost function can tell how bad the model is performing. For example, R-squared (to be discussed later in this chapter) is a cost function of regression model. **Loss function** is almost synonymous to cost function – only difference being loss function is usually a function defined on a data point, while cost function is for the entire training data set.

Machine learning is an optimization problem. We try to define a model and tune the parameters to find the most suitable solution to a problem. However, we need to have

- a way to evaluate the quality or optimality of a solution. This is done using **objective function**. Objective means goal.
- Objective function takes in data and model (along with parameters) as input and returns a value. Target is to find values of model parameter to maximize or minimize the return value. When the objective is to minimize the value, it becomes synonymous to cost function. Examples: maximize the reward function in reinforcement learning, maximize the posterior probability in Naive Bayes, minimize squared error in regression

But the problem that we just talked about is one specific type of problem in machine learning. We have seen in <u>Chapter 1</u> that there are three broad categories of machine learning approaches used for resolving different types of problems. Quickly recapitulating, they are

1. Supervised
   1. Classification
   2. Regression
2. Unsupervised
   1. Clustering
   2. Association analysis
3. Reinforcement

For each of the cases, the model that has to be created/trained is different. Multiple factors play a role when we try to select the model for solving a machine learning problem. The most important factors are (i) the kind of problem we want to solve using machine learning and (ii) the nature of the underlying data. The problem may be related to the prediction of a class value like whether a tumour is malignant or benign, whether the next day will be snowy or rainy, etc. It may be related to prediction – but of some numerical value like what the price of a house should be in the next quarter, what is the expected growth of a certain IT stock in the next 7 days, etc. Certain problems are related to grouping of data like finding customer segments that are using a certain product, movie genres which have got more box office success in the last one year, etc. So, it is very difficult to give a generic guidance related to which machine learning has to be selected. In other words, there is no one model that works best for every machine learning problem. This is what '**No Free Lunch'** theorem also states.

Any learning model tries to simulate some real-worldaspect. However, it is simplified to a large extent removing all intricate details. These simplifications are based on certain assumptions – which are quite dependent on situations. Based on the exact situation,i.e. the problem in hand and the data characteristics, assumptions may or may not hold. So the same model may yield remarkable results in a certain situation while it may completely fail in a different situation. That's why,while doing the data exploration, which we covered inthe previous chapter, we need to understand the data characteristics, combine this understanding with the problem we are trying to solve and then decide which model to be selected for solving the problem.

Let's try to understand the philosophy of model selection in a structured way. Machine learning algorithms are broadly of two types: models for supervised learning, which primarily focus on solvingpredictive problems and models for unsupervised learning, which solve descriptive problems.

Predictive models

Models for supervised learning or predictive models, as is understandable from the name itself, try to predict certain value using the values in an input data set. The learning model attempts to establish a relation between the target feature, i.e. the feature being predicted, andthe predictor features. The predictive models have a clearfocus on what they want to learn and how they want to learn.

Predictive models, in turn, may need to predict thevalue of a category or class to which a data instance belongs to. Below are some examples:

1. Predicting win/loss in a cricket match
2. Predicting whether a transaction is fraud
3. Predicting whether a customer may move to another product

The models which are used for prediction of target features of categorical value are known as classification models. The target feature is known as a class and the categories to which classes are divided into are called levels. Some of the popular classification models include *k*-Nearest Neighbor (kNN), Naïve Bayes, and Decision Tree.

Predictive models may also be used to predict numerical values of the target feature based on thepredictor features. Below are some examples:

1. Prediction of revenue growth in the succeeding year
2. Prediction of rainfall amount in the coming monsoon
3. Prediction of potential flu patients and demand for flu shots nextwinter

The models which are used for prediction of the numerical value of the target feature of a data instance are known as regression models. Linear Regression andLogistic Regression models are popular regression models.

Points to Ponder:

- Categorical values can be converted to numerical values and vice versa. For example, for stock price growth prediction, any growth percentage lying between certain ranges may be represented by a categorical value, e.g. 0%–5% as 'low', 5%– 10% as 'moderate', 10%–20% as 'high' and > 20% as 'booming'. In a similar way, a categorical value can be converted to numerical value, e.g. in the tumor malignancy detection problem, replace 'benign' as 0 and 'malignant' as 1. This way, the models can be used interchangeably, though it may not work always.
- There are multiple factors to be considered while selecting a model. For example, while selecting the model for prediction, the training data size is an important factor to be considered. If the training data set is small, low variance models like Naïve Bayes are supposed to perform better because model overfitting needs to be avoided in this situation. Similarly, when the training data is large, low bias models like logistic regression should be preferred because they can represent complex relationships in a more effective way.

Few models like Support Vector Machines and Neural Network can be used for both classifications as well as for regression.

Descriptive models

Models for unsupervised learning or descriptive models are used to describe a data set or gain insight from a dataset. There is no target feature or single feature of interest in case of unsupervised learning. Based on the value of all features, interesting patterns or insights are derived about the data set.

Descriptive models which group together similar data instances, i.e. data instances having a similar value of the different features are called clustering models. Examples of clustering include

1. Customer grouping or segmentation based on social, demographic, ethnic, etc. factors

2. Grouping of music based on different aspects like genre, language, time-period, etc.

3. Grouping of commodities in an inventory

The most popular model for clustering is *k*-Means.

Descriptive models related to pattern discovery is used for market basket analysis of transactional data. In market basket analysis, based on the purchase pattern available in the transactional data, the possibility of purchasing one product based on the purchase of another product is determined. For example, transactional data may reveal a pattern that generally a customer who purchases milk also purchases biscuit at the same time. This can be useful for targeted promotions or in-store set up. Promotions related to biscuits can be sent to customers of milk products or vice versa. Also, in the store products related to milk can be placed close to biscuits.

TRAINING A MODEL (FOR SUPERVISED LEARNING)

Holdout method

In case of supervised learning, a model is trained using the labelled input data. However, how can we understand the performance of the model? The test data may not be available immediately. Also, the label value of the test data is not known. That is the reason why a part of the input data is held back (that is how the name holdout originates) for evaluation of the model. This subset of the input data is used as the test data for evaluating the performance of a trained model. In general 70%–80% of the input data (which is obviously labelled) is used for model training. The remaining 20%–30% is used as test data for validation of the performance of the model. However, a different proportion of dividing the input data into training and test data is also acceptable. To make sure that the data in both the buckets are similar in nature, the division is done randomly. Random numbers are used to assign data items to the partitions. This method of partitioning the input data into two parts – training and test data (depicted in Figure 3.1), which is by holding back a part of the input data for validating the trained model is known as holdout method
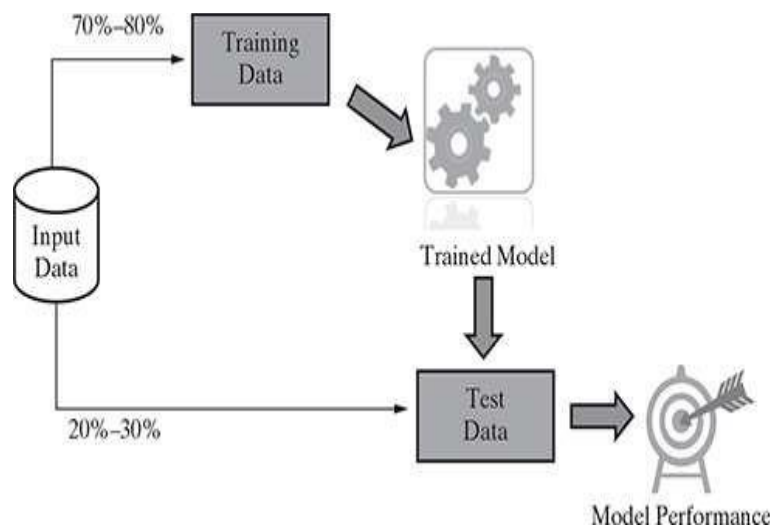
FIG. 3.1 Holdout method

Once the model is trained using the training data, the labels of the test data are predicted using the model's target function. Then the predicted value is compared with the actual value of the label. This is possible because the test data is a part of the input data with known labels. The performance of the model is in general measured by the accuracy of prediction of the label value.

In certain cases, the input data is partitioned into three portions – a training and a test data, and a third validation data. The validation data is used in place of test data, for measuring the model performance. It is used in iterations and to refine the model in each iteration. The test data is used only for once, after the model is refined and finalized, to measure and report the final performance of the model as a reference for future learning efforts.

An obvious problem in this method is that the division of data of different classes into the training and test data may not be proportionate. This situation is worse if the overall percentage of data related to certain classes is much less compared to other classes. This may happen despite the fact that random sampling is employed for test data selection. This problem can be addressed to some extent by applying stratified random sampling in place of sampling. In case of stratified random sampling, the whole data is broken into several homogenous groups or strata and a random sample is selected from each such stratum. This ensures that the generated random partitions have equal proportions of each class.

K-fold Cross-validation method

Holdout method employing stratified random sampling approach still heads into issues in certain specific situations. Especially, the smaller data sets may have the challenge to divide the data of some of the classes proportionally amongst training and test data sets. A special variant of holdout method, called repeated holdout, is sometimes employed to ensure the randomness of the composed data sets. In repeated holdout, several random holdouts are used to measure the model performance. In the end, the average of all performances is taken. As multiple holdouts have been drawn, the training and test data (and also validation data, in case it is drawn) are more likely to contain representative data from all classes and resemble the original input data closely. This process of repeated holdout is the basis of *k*-fold cross-validation technique. In *k*-fold cross-validation, the data set is divided into *k*-completely distinct or non-overlapping random partitions called folds. Figure 3.2 depicts an overall approach for *k*-fold cross-validation.

The value of '*k*' in *k*-fold cross-validation can be set to any number. However, there are two approaches which are extremely popular:

1. 10-fold cross-validation (10-fold CV)
2. Leave-one-out cross-validation (LOOCV)

10-fold cross-validation is by far the most popular approach. In this approach, for each of the 10-folds, each comprising of approximately 10% of the data, one of the folds is used as the test data for validating model performance trained based on the remaining 9 folds (or 90% of the data). This is repeated 10 times, once for each of the 10 folds being used as the test data and the remaining folds as the training data. The average performance across all folds is being reported. Figure 3.3 depicts the detailed approach of selecting the '*k*' folds in *k*-fold cross-validation. As can be observed in the figure, each of the circles resembles a record in the input data set whereas the different colors indicate the different classes that the records belong to. The entire data set is broken into '*k*' folds – out of which one fold is selected in each iteration as the test data set. The fold selected as test data set in each of the '*k*' iterations is different. Also, note that though in figure 3.3 the circles resemble the records in the input data set, the contiguous circles represented as folds do not mean that they are subsequent records in the data set. This is more a virtual representation and not a physical representation. As already mentioned, the records in a fold are drawn by using random sampling technique.
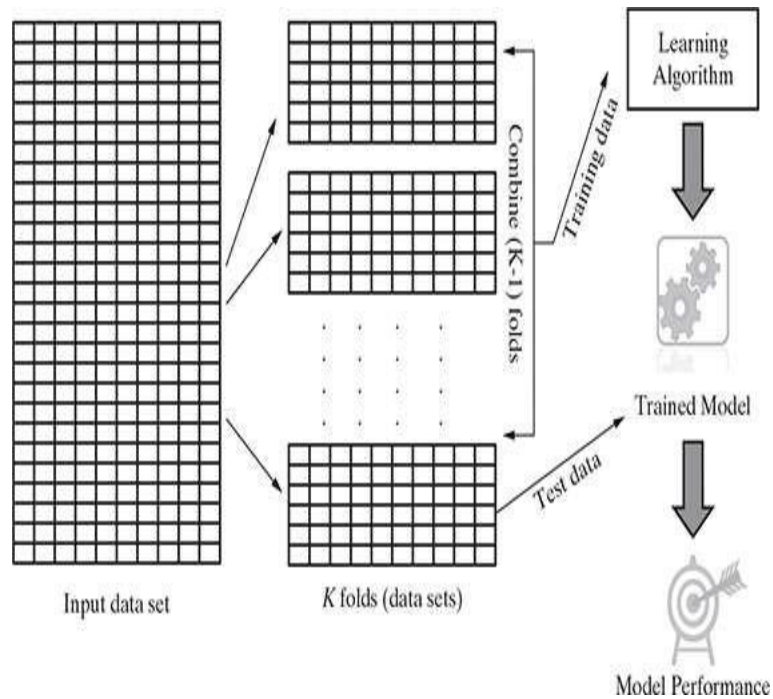
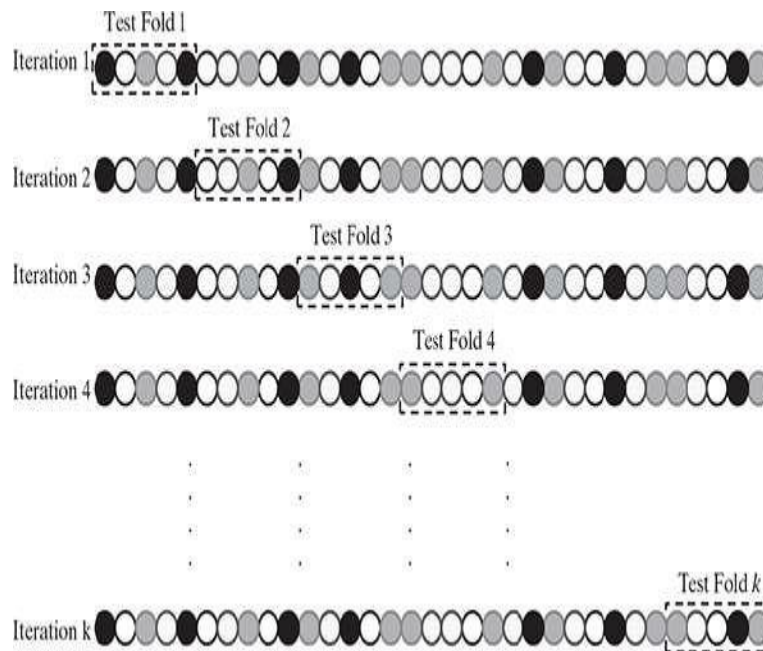**FIG. 3.2** Overall approach for *K*-fold cross-validation

FIG. 3.3 Detailed approach for fold selection

Leave-one-out cross-validation (LOOCV) is an extremecase of *k*-fold cross-validation using one record or data instance at a time as a test data. This is done to maximizethe count of

data used to train the model. It is obvious that the number of iterations for which it has to be run is equal to the total number of data in the input data set.

Hence, obviously, it is computationally very expensiveand not used much in practice.

Bootstrap sampling

Bootstrap sampling or simply bootstrapping is a popular way to identify training and test data sets from the input data set. It uses the technique of Simple Random Sampling with Replacement (SRSWR), which is a well- known technique in sampling theory for drawing randomsamples. We have seen earlier that k-fold cross- validation divides the data into separate partitions – say 10 partitions in case of 10-fold cross-validation. Then it uses data instances from partition as test data and the remaining partitions as training data. Unlike this approach adopted in case of k-fold cross- validation, bootstrapping randomly picks data instances from the input data set, with the possibility of the same data instance to be picked multiple times. This essentially means that from the input data set having 'n' data instances, bootstrapping can create one or more training data sets having 'n' data instances, some of the data instances being repeated multiple times. Figure 3.4 briefly presents the approach followed in bootstrap sampling.

This technique is particularly useful in case of input data sets of small size, i.e. having very less number of data instances.
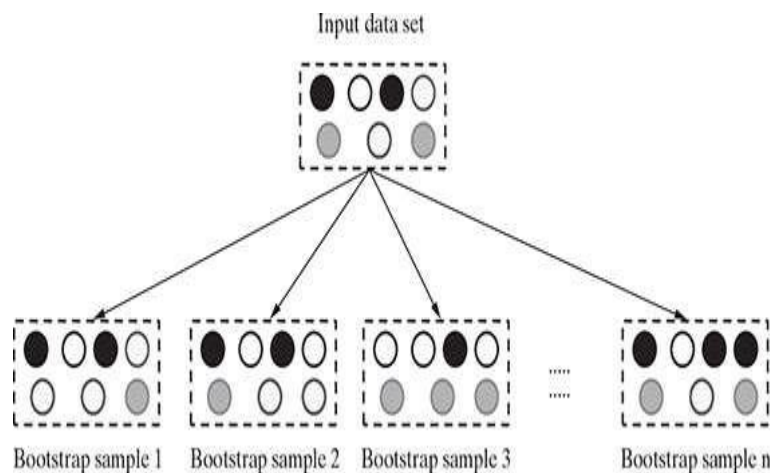

FIG. 3.4 Bootstrap sampling

CROSS-VALIDATION	BOOTSTRAPPING
It is a special variant of holdout method, called repeated holdout. Hence uses stratified random sampling approach (without replacement). Data set is divided into 'k' random partitions, with each partition containing approximately $\frac{n}{k}$ number of unique data elements, where 'n' is the total number of data elements and 'k' is the total number of folds.	It uses the technique of Simple Random Sampling with Replacement (SRSWR). So the same data instance may be picked up multiple times in a sample.
The number of possible training/test data samples that can be drawn using this technique is finite.	In this technique, since elements can be repeated in the sample, possible number of training/test data samples is unlimited.

Lazy vs. Eager learner

Eager learning follows the general principles of machine learning – it tries to construct a generalized, input- independent target function during the model training phase. It follows the typical steps of machine learning, i.e. abstraction and generalization and comes up with a trained model at the end of the learning phase. Hence, when the test data comes in for classification, the eager learner is ready with the model and doesn't need to referback to the training data. Eager learners take more time in the learning phase than the lazy learners. Some of thealgorithms which adopt eager learning approach include

Some of thealgorithms which adopt eager learning approach include Decision Tree, Support Vector Machine, Neural Network,etc.

Lazy learning, on the other hand, completely skips the abstraction and generalization processes, as explained incontext of a typical machine learning process. In that respect, strictly speaking, lazy learner doesn't 'learn' anything. It uses the training data in exact, and uses the knowledge to classify the unlabelled test data. Since lazy learning uses training data as-is, it is also known as rote learning (i.e. memorization technique based on repetition). Due to its heavy dependency on the given training data instance, it is also known as instance learning. They are also called non-parametric learning. Lazy learners take very little time in training because notmuch of training actually happens. However, it takes quite some time in classification as for each tuple of test data, a comparison-based assignment of label happens. One of the most popular algorithm for lazy learning is k- nearest neighbor.

MODEL REPRESENTATION AND INTERPRETABILITY

We have already seen that the goal of supervised machine learning is to learn or derive a target function which can best determine the target variable from the set of input variables. A key consideration in learning the target function from the training data is the extent of generalization. This is because the input data is just a limited, specific view and the new, unknown data in the test data set may be differing quite a bit from the trainingdata.

Fitness of a target function approximated by a learning algorithm determines how correctly it is able to classify a set of data it has never seen.

Underfitting

If the target function is kept too simple, it may not be able to capture the essential nuances and represent theunderlying data well. A typical case of underfitting mayoccur when trying to represent a non-linear data with alinear model as demonstrated by both cases of underfitting shown in figure 3.5. Many times underfitting happens due to unavailability of sufficient training data. Underfitting results in both poor performance with training data as well as poor generalization to test data. Underfitting can be avoidedby

1. using more training data
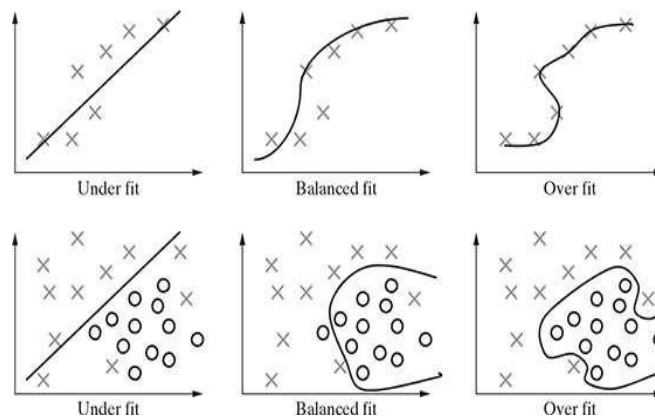2. reducing features by effective feature selection

FIG. 3.5 Underfitting and Overfitting of models

Overfitting

Overfitting refers to a situation where the model has been designed in such a way that it emulates the trainingdata too closely. In such a case, any specific deviation in the training data, like noise or outliers, gets embedded inthe model. It adversely impacts the performance of the model on the test data. Overfitting, in many cases, occur as a result of trying to fit an excessively complex model to closely match the training data. This is represented with a sample data set in <u>figure 3.5</u>. The target function, in these cases, tries to make sure all training data points are correctly partitioned by the decision boundary.

However, more often than not, this exact nature is not replicated in the unknown test data set. Hence, the targetfunction results in wrong classification in the test data set. Overfitting results in good performance with trainingdata set, but poor generalization and hence poor performance with test data set. Overfitting can be avoided by

1. using re-sampling techniques like k-fold cross validation
2. hold back of a validation data set
3. remove the nodes which have little or no predictive power forthe given machine learning problem.

Both underfitting and overfitting result in poorclassification quality which is reflected by low classification accuracy.

Bias – variance trade-off

In supervised learning, the class value assigned by the learning model built based on the training data may differ from the actual class value. This error in learning can be of two types – errors due to 'bias' and error due to'variance'. Let's try to understand each of them in details.

Errors due to 'Bias'

Errors due to bias arise from simplifying assumptions made by the model to make the target function less complex or easier to learn. In short, it is due to underfitting of the model. Parametric models generally have high bias making them easier to understand/interpret and faster to learn. These algorithms have a poor performance on data sets, which are complex in nature and do not align with the simplifying assumptions made by the algorithm. Underfitting results in high bias.

Errors due to 'Variance'

Errors due to variance occur from difference in training data sets used to train the model. Different training data sets (randomly sampled from the input data set) are usedto train the model. Ideally the difference in the data sets should not be significant and the model trained using different training data sets should not be too different.

However, in case of overfitting, since the model closelymatches the training data, even a small difference in training data gets magnified in the model.
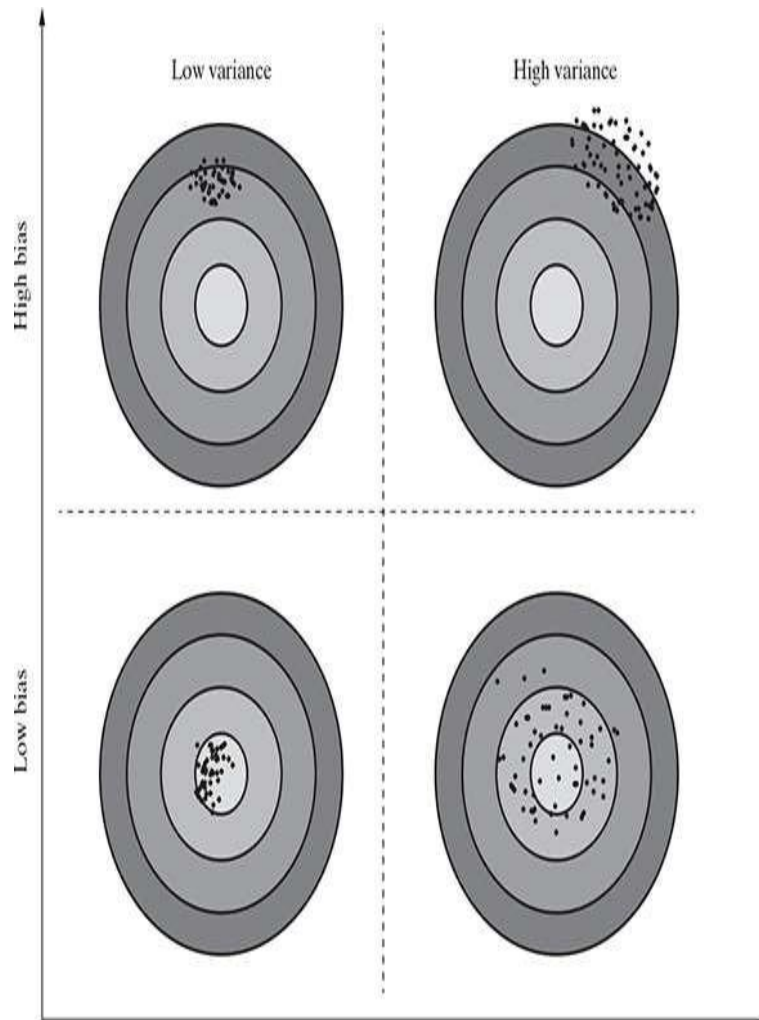
FIG. 3.6 Bias-variance trade-off

So, the problems in training a model can either happenbecause either (a) the model is too simple and hence failsto interpret the data grossly or (b) the model is extremelycomplex and magnifies even small differences in the training data.

As is quite understandable:

- Increasing the bias will decrease the
- variance, and Increasing the variance will decrease the bias

On one hand, parametric algorithms are generally seento demonstrate high bias but low variance. On the other hand, non-parametric algorithms demonstrate low bias and high variance.

As can be observed in Figure 3.6 , the best solution is to have a model with low bias as well as low variance. However, that may not be possible in reality. Hence, the goal of supervised machine learning is to achieve a balance between bias and variance. The learning algorithm chosen and the user parameters which can be configured helps in striking a trade-off between bias and variance. For example, in a popular supervised algorithm k-Nearest Neighbors or kNN, the user configurable parameter 'k' can be used to do a trade-off between bias and variance. In one hand, when the value of 'k' is decreased, the model becomes simpler to fit

and bias increases. On the other hand, when the value of 'k' is increased, the variance increases.

EVALUATING PERFORMANCE OF A MODEL

Supervised learning – classification

In supervised learning, one major task is classification. The responsibility of the classification model is to assign class label to the target feature based on the value of the predictor features. For example, in the problem of predicting the win/loss in a cricket match, the classifier will assign a class value win/loss to target feature based on the values of other features like whether the team wonthe toss, number of spinners in the team, number of wins the team had in the tournament, etc. To evaluate the performance of the model, the number of correct classifications or predictions made by the model has to be recorded. A classification is said to be correct if, say for example in the given problem, it has been predictedby the model that the team will win and it has actually won.

Based on the number of correct and incorrect classifications or predictions made by a model, the accuracy of the model is calculated. If 99 out of 100 timesthe model has classified correctly, e.g. if in 99 out of 100 games what the model has predicted is same as what the outcome has been, then the model accuracy is said to be 99%. However, it is quite relative to say whether a model has performed well just by looking at the accuracy value. For example, 99% accuracy in case of a sports win predictor model may be reasonably good but the same number may not be acceptable as a good threshold when the learning problem deals with predicting a critical illness. In this case, even the 1% incorrect prediction may lead to loss of many lives. So the model performance needs to be evaluated in light of the learning problem in question. Also, in certain cases, erring on the side of caution may be preferred at the cost of overall accuracy. For that reason, we need to look more closely at the model accuracy and also at the same time look at other measures of performance of a model like sensitivity, specificity, precision, etc. So, let's start with looking at model accuracy more closely. And let's try to understand it with an example.

There are four possibilities with regards to the cricketmatch win/loss prediction:

1. the model predicted win and the team won
2. the model predicted win and the team lost
3. the model predicted loss and the team won
4. the model predicted loss and the team lost

In this problem, the obvious class of interest is 'win'.

The first case, i.e. the model predicted win and the team won is a case where the model has correctly classified data instances as the class of interest. Thesecases are referred as True Positive (TP) cases.

The second case, i.e. the model predicted win and the team lost is a case where the model incorrectly classified data instances as the class of interest. These cases arereferred as False Positive (FP) cases.

The third case, i.e. the model predicted loss and theteam won is a case where the model has incorrectly classified as not the class of interest. These cases are referred as False Negative (FN) cases.
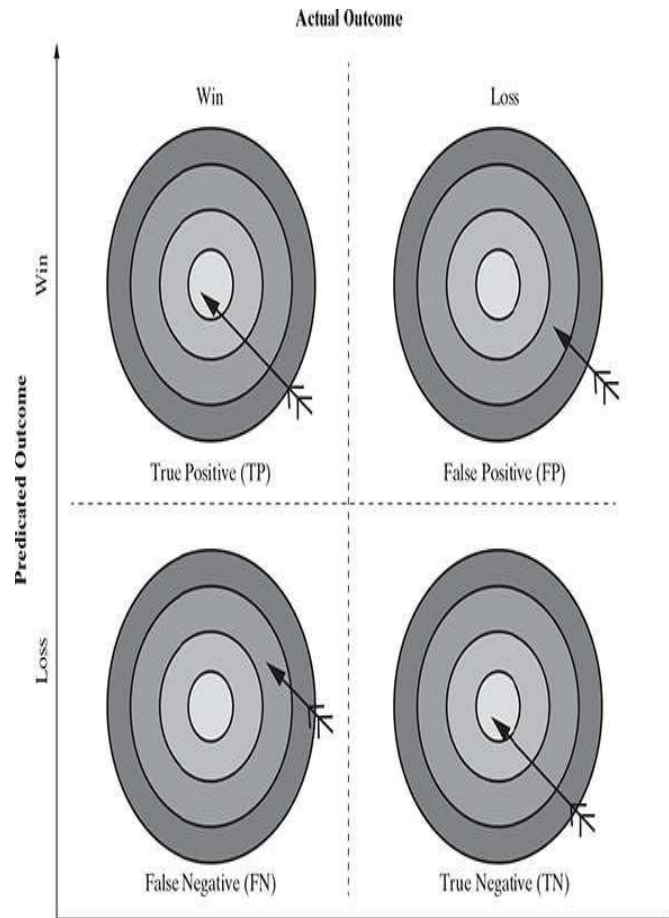

FIG. 3.7 Details of model classification

The fourth case, i.e. the model predicted loss and the team lost is a case where the model has correctly classified as not the class of interest. These cases are referred as True Negative (TN) cases. All these four casesare depicted in Figure 3.7 .

For any classification model, **model accuracy** is given by total number of correct classifications (either asthe class of interest, i.e. True Positive or as not the class of interest, i.e. True Negative) divided by total number of classifications done.

$$\text{Model accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$

A matrix containing correct and incorrect predictions in the form of TPs, FPs, FNs and TNs is known as **confusion matrix**. The win/loss prediction of cricket match has two classes of interest – win and loss. For thatreason it will generate a 2 × 2 confusion matrix. For a classification problem involving three classes, the confusion matrix would be 3 × 3, etc.

Let's assume the confusion matrix of the win/lossprediction of cricket match problem to be as below:

	ACTUAL WIN	ACTUAL LOSS
Predicted Win	85	4
Predicted Loss	2	9

In context of the above confusion matrix, total count of TPs = 85, count of FPs = 4, count of FNs = 2 and count of TNs = 9.

$$\therefore \text{Model accuracy} = \frac{TP + TN}{TP + FP + FN + TN} = \frac{85 + 9}{85 + 4 + 2 + 9} = \frac{94}{100} = 94\%$$

The percentage of misclassifications is indicated using **error rate** which is measured as

$$\text{Error rate} = \frac{FP + FN}{TP + FP + FN + TN}$$

In context of the above confusion matrix,

$$\text{Error rate} = \frac{FP + FN}{TP + FP + FN + TN} = \frac{4 + 2}{85 + 4 + 2 + 9} = \frac{6}{100} = 6\%$$

$$= 1 - \text{Model accuracy}$$

Sometimes, correct prediction, both TPs as well as TNs, may happen by mere coincidence. Since these occurrences boost model accuracy, ideally it should not happen. **Kappa** value of a model indicates the adjustedthe model accuracy. It is calculated using the formula below:

$$\text{Kappa value (k)} = \frac{P(a) - P(p_r)}{1 - P(p_r)}$$

P(a) = Proportion of observed agreement between actual and predicted in overall data set

$$= \frac{TP + TN}{TP + FP + FN + TN}$$

P(p$_r$) = Proportion of expected agreement between actual and predicted data both in case of class of interest as well as the other classes

$$= \frac{TP + FP}{TP + FP + FN + TN} \times \frac{TP + FN}{TP + FP + FN + TN} + \frac{FN + TN}{TP + FP + FN + TN}$$
$$\times \frac{FP + TN}{TP + FP + FN + TN}$$

In context of the above confusion matrix, total count of TPs = 85, count of FPs = 4, count of FNs = 2 and count of TNs = 9.

$$\therefore P(a) = \frac{TP + TN}{TP + FP + FN + TN} = \frac{85 + 9}{85 + 4 + 2 + 9} = \frac{94}{100} = 0.94$$

$$P(p_r) = \frac{85 + 4}{85 + 4 + 2 + 9} \times \frac{85 + 2}{85 + 4 + 2 + 9} + \frac{2 + 9}{85 + 4 + 2 + 9} \times \frac{4 + 9}{85 + 4 + 2 + 9}$$

$$= \frac{89}{100} \times \frac{87}{100} + \frac{11}{100} \times \frac{13}{100} = 0.89 \times 0.87 + 0.11 \times 0.13 = 0.7886$$

$$\therefore k = \frac{0.94 - 0.7886}{1 - 0.7886} = 0.7162$$

As discussed earlier, in certain learning problems it is critical to have extremely low number of FN cases, if needed, at the cost of a conservative classification model.Though it is a clear case of misclassification and will impact model accuracy adversely, it is still required as missing each class of interest may have serious consequence. This happens more in problems from medical domains like disease prediction problem. For example, if a tumor is malignant but wrongly classified as benign by the classifier, then the repercussion of such misclassification is fatal. It does not matter if higher number of tumours which are benign are wrongly classified as malignant. In these problems there are somemeasures of model performance which are more important than accuracy. Two such critical measurements are sensitivity and specificity of the model.

The **sensitivity** of a model measures the proportionof TP examples or positive cases which were correctly classified. It is measured as

$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

In the context of the above confusion matrix for thecricket match win prediction problem,

$$\text{Sensitivity} = \frac{TP}{TP + FN} = \frac{85}{85 + 2} = \frac{85}{87} = 97.7\%$$

So, again taking the example of the malignancy prediction of tumours, class of interest is 'malignant'. Sensitivity measure gives the proportion of tumours which are actually malignant and have been predicted asmalignant. It is quite obvious that for such problems themost critical measure of the performance of a good model is sensitivity. A high value of sensitivity is more desirable than a high value of accuracy.

Specificity is also another good measure to indicate agood balance of a model being excessively conservative or excessively aggressive. Specificity of a model measuresthe proportion of negative examples which have been correctly classified. In the context, of malignancy prediction of tumours, specificity gives the proportion of benign tumours which have been correctly classified. In the context of the above confusion matrix for the cricket match win prediction problem,

$$\text{Specificity} = \frac{TN}{TN + FP} = \frac{9}{9 + 4} = \frac{9}{13} = 69.2\%$$

A higher value of specificity will indicate a better model performance. However, it is quite understandable that a conservative approach to reduce False Negatives might actually push up the number of FPs. Reason for this is that the model, in order to reduce FNs, is going to classify more tumours as malignant. So the chance that benign tumours will be classified as malignant or FPs will increase.

There are two other performance measures of a supervised learning model which are similar to sensitivity and specificity. These are **precision** and **recall**. While precision gives the proportion of positive predictions which are truly positive, recall gives the proportion of TP cases over all actually positive cases.

$$\text{Precision} = \frac{TP}{TP + FP}$$

Precision indicates the reliability of a model in predicting a class of interest. When the model is related to win / loss prediction of cricket, precision indicates how often it predicts the win correctly. In context of the above confusion matrix for the cricket match win prediction problem,

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{85}{85 + 4} = \frac{85}{89} = 95.5\%$$

It is quite understandable that a model with higher precision is perceived to be more reliable.

Recall indicates the proportion of correct prediction of positives to the total number of positives. In case of win/loss prediction of cricket, recall resembles what proportion of the total wins were predicted correctly.

$$\text{Recall} = \frac{TP}{TP + FN}$$

In the context of the above confusion matrix for the cricket match win prediction problem,

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{85}{85 + 2} = \frac{85}{87} = 97.7\%$$

F-measure

F-measure is another measure of model performance which combines the precision and recall. It takes the harmonic mean of precision and recall as calculated as

$$\text{F-measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

In context of the above confusion matrix for the cricket match win prediction problem,

$$\text{F-measure} = \frac{2 \times 0.955 \times 0.977}{0.955 + 0.977} = \frac{1.866}{1.932} = 96.6\%$$

As a combination of multiple measures into one, *F*- score gives the right measure using which performance of different models can be compared. However, one assumption the calculation is based on is that precision and recall have equal weight, which may not always be true in reality. In certain problems, the disease prediction problems, e.g., precision may be given far more weightage. In that case, different weightages may be assigned to precision and recall. However, there may be a serious dilemma regarding what value to be adopted for each and what is the basis for the specific value adopted.

Receiver operating characteristic (ROC) curves

As we have seen till now, though accuracy is the most popular measure, there are quite a number of other measures to evaluate the performance of a supervised learning model. However, visualization is an easier and more effective way to understand the model performance. It also helps in comparing the efficiency oftwo models.

Receiver Operating Characteristic (ROC) curve helps in visualizing the performance of a classification model. It shows the efficiency of a model in the detection of true positives while avoiding the occurrence of false positives. To refresh our memory, true positives are those cases where the model has correctly classified data instances asthe class of interest. For example, the model has correctly classified the tumours as malignant, in case of a tumour malignancy prediction problem. On the other hand, FPs are those cases where the model incorrectly classified data instances as the class of interest. Using thesame example, in this case, the model has incorrectly classified the tumours as malignant, i.e. tumours whichare actually benign have been classified as malignant.

$$\text{True Positive Rate TPR} = \frac{TP}{TP + FN}$$

$$\text{False Positive Rate FPR} = \frac{FP}{FP + TN}$$

In the ROC curve, the FP rate is plotted (in the horizontal axis) against true positive rate (in the vertical axis) at different classification thresholds. If we assume a lower value of classification threshold, the model classifies more items as positive. Hence, the values of both False Positives and True Positives increase. The area under curve (AUC) value, as shown in figure 3.8a , isthe area of the two-dimensional space under the curve extending from (0, 0) to (1, 1), where each point on the curve gives a set of true and false positive values at a specific classification threshold. This curve gives an indication of the predictive quality of a model. AUC valueranges from 0 to 1, with an AUC of less than 0.5 indicating that the classifier has no predictive ability.

Figure 3.8b shows the curves of two classifiers – classifier 1 and classifier 2. Quite obviously, the AUC ofclassifier 1 is more than the AUC of classifier 2. So, we can draw the inference that classifier 1 is better than classifier 2.
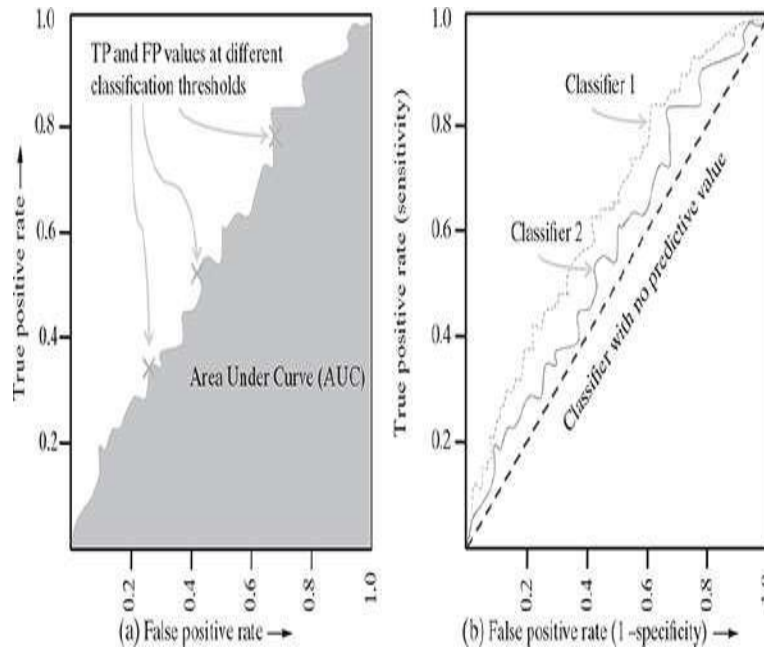
FIG. 3.8 ROC curve

A quick indicative interpretation of the predictive values from 0.5 to 1.0 is given below:

- – 0.6 ➜ Almost no predictive ability
- – 0.7 ➜ Weak predictive ability
- – 0.8 ➜ Fair predictive ability

- – 0.9 ➜ Good predictive ability
- – 1.0 ➜ Excellent predictive ability

Supervised learning – regression

A well-fitted regression model churns out predicted values close to actual values. Hence, a regression model which ensures that the difference between predicted and actual values is low can be considered as a good model. Figure 3.9 represents a very simple problem of real estate value prediction solved using linear regression model. If 'area' is the predictor variable (say x) and 'value' is the target variable (say y), the linear regression model can be represented in the form:
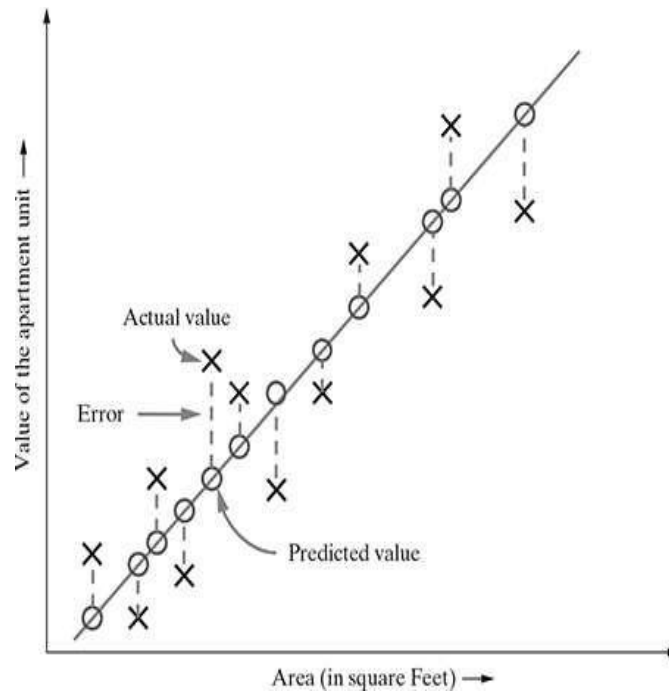
$$y = \alpha + \beta x$$

FIG. 3.9 Error – Predicted vs. actual value

For a certain value of x, say $\hat{x}$, the value of y is predicted as $\hat{y}$ whereas the actual value of y is Y (say). The distance between the actual value and the fitted or predicted value, i.e. $\hat{y}$ is known as **residual**. The regression model can be considered to be fitted well if the difference between actual and predicted value, i.e. the residual value is less.

R-squared is a good measure to evaluate the model fitness. It is also known as the coefficient of determination, or for multiple regression, the coefficient of multiple determination. The R-squared value lies between 0 to 1 (0%–100%) with a larger value representing a better fit. It is calculated as:

$$R^2 = \frac{SST - SSE}{SST}$$

Sum of Squares Total (SST) = squared differences of each observation from the overall mean=

$$\sum_{i=1}^{n}(y_i - \bar{y})^2$$

where $\bar{y}$ is the mean.

Sum of Squared Errors (SSE) (of prediction) = sum of the squared residuals=

$$\sum_{i=1}^{n}(Y_i - \hat{y})^2$$

Where ^yi is the predicted value of y_i and Y_i is the actual value of y_i.

Unsupervised learning – clustering

Clustering algorithms try to reveal natural groupings amongst the data sets. However, it is quite tricky to evaluate the performance of a clustering algorithm. Clustering, by nature, is very subjective and whether the cluster is good or bad is open for interpretations. It was noted, 'clustering is in the eye of the beholder'. This stems from the two inherent challenges which lie in the process of clustering:

1.It is generally not known how many clusters can be formulated from a particular data set. It is completely open-ended in most cases and provided as a user input to a clustering algorithm.

2.Even if the number of clusters is given, the same number of clusters can be formed with different groups of data instances.

In a more objective way, it can be said that a clustering algorithm is successful if the clusters identified using the algorithm is able to achieve the right results in the overall problem domain. For example, if clustering is applied for identifying customer segments for a marketing campaign of a new product launch, the clustering can be considered successful only if the marketing campaign ends with a success, i.e. it is able to create the right brand recognition resulting in steady revenue from new product sales. However, there are couple of popular approaches which are adopted for cluster quality evaluation.

1.Internal evaluation

In this approach, the cluster is assessed based on the underlying data that was clustered. The internal evaluation methods generally measure cluster quality based on homogeneity of data belonging to the same cluster and heterogeneity of data belonging to different clusters. The homogeneity/heterogeneity is decided by some similarity measure. For example, **silhouette coefficient**, which is one of the most popular internal evaluation methods, uses distance (Euclidean or Manhattan distances most commonly used) between data elements as a similarity measure. The value of silhouette width ranges between -1 and $+1$, with a high value indicating high intra- cluster homogeneity and inter-cluster heterogeneity.

For a data set clustered into 'k' clusters, silhouette width is calculated as:

$$\text{Silhouette width} = \frac{b(i) - a(i)}{\max\{a(i),\ b(i)\}}$$

$a(i)$ is the average distance between the i th data instance and all other data instances belonging to the same cluster and $b(i)$ is the lowest average distance between the i-the data instance and data instances of all other clusters.

Let's try to understand this in context of the example depicted in <u>figure 3.10</u>. There are four clusters namely cluster 1, 2, 3, and 4. Let's consider an arbitrary data element 'i' in cluster 1, resembled by the asterisk. $a(i)$ is the average of the distances $a_{i1}, a_{i2}, \ldots, a_{in1}$ of the different data elements from the i th data element in cluster 1, assuming there are n_1 data elements in cluster 1. Mathematically,

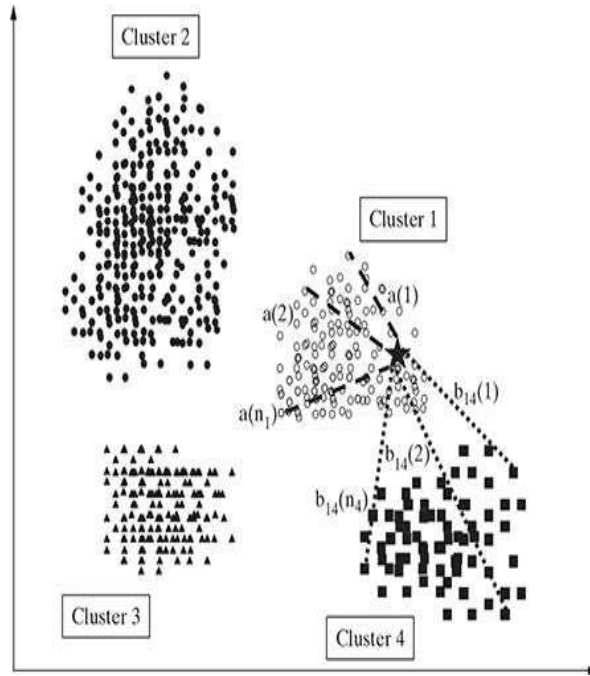$$a(i) = \frac{a_{i1} + a_{i2} + \ldots + a_{in_1}}{n_1}$$

FIG. 3.10 Silhouette width calculation

In the same way, let's calculate the distance of an arbitrary data element 'i' in cluster 1 with the different data elements from another cluster, say cluster 4 and take an average of all those distances. Hence,

$$b_{14}(\text{average}) = \frac{b_{14}(1) + b_{14}(2) + \ldots + b_{14}(n_4)}{(n_4)}$$

where n_4 is the total number of elements in cluster 4. In the same way, we can calculate the values of $b12$ (average) and $b13$ (average). b(i) is the minimum of all these values. Hence, we can say that,

b(i) = minimum [$b12$(average), $b13$(average), $b14$(average)]

2.External evaluation

In this approach, class label is known for the data set subjected to clustering. However, quite obviously, the known class labels are not a part of the data used in clustering. The cluster algorithm is assessed based on how close the results are compared to those known class labels. For example, **purity** is one of the most popular measures of cluster algorithms – evaluates the extent to which clusters contain a single class.

For a data set having 'n' data instances and 'c' known class labels which generates 'k' clusters, purity is measured as:

$$\text{Purity} = \frac{1}{n} \sum_{k} \max(k \cap c)$$

IMPROVING PERFORMANCE OF A MODEL

Now we have almost reached the end of the journey of building learning models. We have got some idea about what modelling is, how to approach about it to solve a learning problem and how to measure the success of our model. Now comes a million dollar question. Can we

improve the performance of our model? If so, then what are the levers for improving the performance? In fact, even before that comes the question of model selection – which model should be selected for which machine learning task? We have already discussed earlier that the model selection is done one several aspects:

1. Type of learning the task in hand, i.e. supervised or unsupervised

2. Type of the data, i.e. categorical or numeric

3. Sometimes on the problem domain

4. Above all, experience in working with different models to solve problems of diverse domains

So, assuming that the model selection is done, what are the different avenues to improve the performance of models?

One effective way to improve model performance is by tuning model parameter. **Model parameter tuning** is the process of adjusting the model fitting options. For example, in the popular classification model k-Nearest Neighbour (kNN), using different values of 'k' or the number of nearest neighbours to be considered, the model can be tuned. In the same way, a number of hidden layers can be adjusted to tune the performance in neural networks model. Most machine learning models have at least one parameter which can be tuned.

As an alternate approach of increasing the performance of one model, several models may be combined together. The models in such combination are complimentary to each other, i.e. one model may learn one type data sets well while struggle with another type of data set. Another model may perform well with the data set which the first one struggled with. This approach of combining different models with diverse strengths is known as **ensemble** (depicted in Figure 3.11).

Ensemble helps in averaging out biases of the different underlying models and also reducing the variance.

Ensemble methods combine weaker learners to create stronger ones. A performance boost can be expected even if models are built as usual and then ensembled.

Following are the typical steps in ensemble process:

- Build a number of models based on the training data
- For diversifying the models generated, the training data subset can be varied using the allocation function. Sampling techniques like bootstrapping may be used to generate unique training data sets.
- Alternatively, the same training data may be used but the models combined are quite varying, e.g, SVM, neural network, kNN, etc. The outputs from the different models are combined using a combination function.
- A very simple strategy of combining, say in case of a prediction task using ensemble, can be majority voting of the different models combined. For example, 3 out of 5 classes predict 'win' and 2 predict 'loss' – then the final outcome of the ensemble using majority vote would be a 'win'.
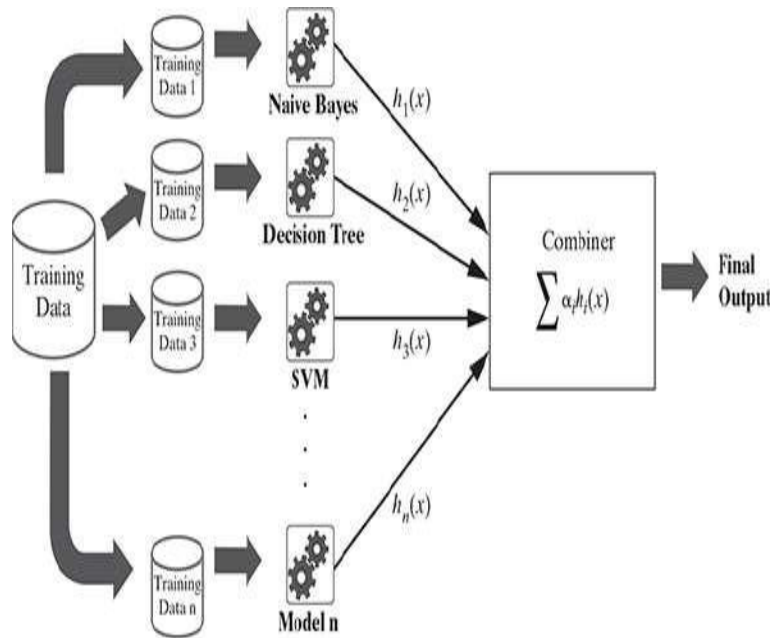
FIG. 3.11 Ensemble

One of the earliest and most popular ensemble modelsis **bootstrap aggregating** or **bagging**. Bagging uses bootstrap sampling method (refer section 3.3.3) to generate multiple training data sets. These training data sets are used to generate (or train) a set of models using the same learning algorithm. Then the outcomes of the models are combined by majority voting (classification) or by average (regression). Bagging is a very simple ensemble technique which can perform really well for unstable learners like a decision tree, in which a slight change in data can impact the outcome of a model significantly.

Just like bagging, **boosting** is another key ensemble- based technique. In this type of ensemble, weaker learning models are trained on resampled data and the outcomes are combined using a weighted voting approach based on the performance of different models. **Adaptive boosting** or **AdaBoost** is a special variant ofboosting algorithm. It is based on the idea of generating weak learners and slowly learning.

Random forest is another ensemble-based technique. It is an ensemble of decision trees – hence the name random forest to indicate a forest of decision trees.It has been discussed in more details in chapter 7.

In this chapter, you have been introduced to the crux of machine learning, i.e. modelling. Thorough understanding of the technical aspects elaborated in this chapter is extremely crucial for the success of any machine learning project. For example, the first dilemma comes about which model to select. Again, in case of supervised learning, how can we deal with the unavailability of sufficient training data. In the same way, once the model is trained in case of supervised learning or the grouping is done in case of clustering, how we can understand whether the model training (for supervised) or grouping done (for unsupervised) is good or bad. All these and more have been addressed as a part of this chapter.

SAMPLE QUESTIONS

MULTIPLE-CHOICE QUESTIONS (1 MARK QUESTIONS):

1. Structured representation of raw input data to meaningful _____ is called a model.

   1. pattern

   2. data

   3. object

   4. none of the above

2. For supervised learning we have ___ model.

   1. interactive

   2. predictive

   3. descriptive

   4. prescriptive

3. For unsupervised learning we have_model.

   1. interactive

   2. predictive

   3. descriptive

   4. prescriptive

4. Which of the following measure is not used for a classification model?

   1. Accuracy

   2. Recall

   3. Purity

   4. Error rate

5. Which of the following is a performance measure for regression?

   1. Accuracy

   2. Recall

   3. RMSE

   4. Error rate

6.Which of the following is the measure of cluster quality?

  1.Purity

  2.Distance

  3.Accuracy

  4.all of the above

7.Out of 200 emails, a classification model correctly predicted 150 spam emails and 30 ham emails. What is the accuracy of the model?

  1. 10%

  2. 90%

  3. 80%

  4. none of the above

8.Out of 200 emails, a classification model correctly predicted 150 spam emails and 30 ham emails. What is the error rate of the model?

  1. 10%

  2. 90%

  3. 80%

  4. none of the above

9.There is no one model that works best for every machine learning problem. This is stated as

  1.Fit gap model theorem

  2.One model theorem

  3.Free lunch theorem

  4.No free lunch theorem

10.LOOCV in machine learning stands for

  1.Love one-out cross validation

  2.Leave-one-out cross-validation

  3.Leave-object oriented cross-validation

  4.Leave-one-out class-validation

SHORT-ANSWER TYPE QUESTIONS (5 MARKS QUESTIONS):

1. What is a model in context of machine learning? How can you train a model?

2. Explain "No Free Lunch" theorem in context of machine learning.

3. Explain, in details, the process of K-fold cross-validation.

4. Explain the bootstrap sampling. Why is it needed?

5. Why do we need to calculate Kappa value for a classification model? Show, with a sample set of data, how to calculate Kappa value of a classification model.

6. Explain the process of ensemble of models. What role does in play in machine learning?

7. What is the main purpose of a descriptive model? State some real-world problems solved using descriptive models.

8. Explain the process of evaluating a linear regression model.

9. Differentiate (any two):

   1. Predictive vs. descriptive models

   2. Model underfitting vs. overfitting

   3. Cross-validation vs. bootstrapping

10. Write short notes on any two:

   1. LOOCV

   2. F-measure

   3. Silhouette width

   4. ROC curve

LONG-ANSWER TYPE QUESTIONS (10 MARKS QUESTIONS):

1. What is a target function? Express target function in context of a real-life example. How is the fitness of a target function measured?

2. What are predictive models? What are descriptive models? Give examples of both types of models. Explain the difference between these types of models.

3. Explain, in details, the process of evaluating the performance of a classification model. Explain the different parameters of measurement.

4. 1. What is underfitting in context of machine learning models? What is the major cause of underfitting?

   2. What is overfitting? When does it happen?

   3. Explain bias-variance trade-off in context of model fitting.

5. Can the performance of a learning model be improved? If yes, explain how.

6.How would you evaluate the success of an unsupervised learning model? What are the most popular measures of performance for an unsupervised learning model?

7.Is there a way to use a classification model for a numerical data or a regression model on a categorical data? Explain your answer.

8.Describe the process of predictive modelling for numerical values. How is it different from predictive modelling for categorical values?

9.While predicting malignancy of tumour of a set of patients using a classification model, following are the data recorded:

   1.Correct predictions – 15 malignant, 75 benign

   2.Incorrect predictions – 3 malignant, 7 benign

10.Calculate the error rate, Kappa value, sensitivity, precision, and F-measure of the model.

   1. Write short notes on any <u>two</u>:

      1.Holdout method

      2.10-fold cross-validation

      3.Parameter tuning

   2.Write the difference between (any <u>two</u>):

      1.Purity vs. Silhouette width

      2.Bagging vs. Boosting

      3.Lazy vs. Eager learner

CHAPTER-II

Basics of Feature Engineering

OBJECTIVE OF THE CHAPTER

In the last three chapters, you have been introduced to the basic concepts of machine learning. Also, the process to start modelling a problem has also been discussed in details. With this context in mind, in this chapter, we will introduce you to another very important aspect of machine learning, that is featureengineering. Though not a core part of the machine learning processes, feature engineering is a critical allied task that we need to perform to make learningmore effective. It has three key components – feature construction, feature selection, and feature transformation, each of which will be covered in details in this chapter.

INTRODUCTION

In the last three chapters, we had a jumpstart to the machine learning process. We first started with what human learning is and how the different types of machine learning emulate the aspects of human learning. We had a detailed view of the different types of problem that can be solved using machine learning techniques.

Before applying machine learning to solve the problems, there are certain preparatory steps. These preparatory steps have been covered in details. After that, we have done a step-by-step navigation of the different activities of modelling a problem using machine learning.

Modelling alone doesn't help us to realize the effectiveness of machine learning as a problem- solving tool. So we also learnt how to measure the effectiveness of machine learning models in solving problems. In case a specific model is not effective, we can use different levers to boost the effectiveness. Those levers of boostingthe model performance were also covered.

Now that we are ready (well almost ready!) to start solving problems using machine learning, we need to touch upon another key aspect which plays a critical role in solving any machine learning problem – feature engineering. Though feature engineering is a part of the preparatory activities which have already been coveredin Chapter 2, the criticality and vastness of the area call for treating it separately. This area deals with features of the data set, which form an important input of any machine learning problem – be supervised or unsupervised learning. Feature engineering is a critical preparatory process in machine learning. It is responsible for taking raw input data and converting thatto well-aligned features which are ready to be used by themachine learning models. But before we start discussing feature engineering,let's try to understand more clearly what feature is.

What is a feature?

A feature is an attribute of a data set that is used in a machine learning process. There is a view amongst certain machine learning practitioners that only those attributes which are meaningful to a machine learning problem are to be called as features, but this view has to be taken with a pinch of salt. In fact, selection of the subset of features which are meaningful for machine learning is a sub-area of feature engineering which draws a lot of research interest. The features in a data set are also called its dimensions. So a data set having 'n' features is called an n-dimensional data set.

Let's take the example of a famous machine learning data set, Iris, introduced by the British statistician and biologist Ronald Fisher, partly shown in Figure 4.1. It has five attributes or features namely Sepal.Length, Sepal.Width, Petal.Length, Petal. Width and Species. Out of these, the feature 'Species' represent the class variable and the remaining features are the predictor variables. It is a five-dimensional data set.

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
6.7	3.3	5.7	2.5	Virginica
4.9	3	1.4	0.2	Setosa
5.5	2.6	4.4	1.2	Versicolor
6.8	3.2	5.9	2.3	Virginica
5.5	2.5	4	1.3	Versicolor
5.1	3.5	1.4	0.2	Setosa
6.1	3	4.6	1.4	versicolor

FIG. 4.1 Data set features

What is feature engineering?

Feature engineering refers to the process of translating a data set into features such that these features are able to represent the data set more effectively and result in a better learning performance.

As we know already, feature engineering is an important pre-processing step for machine learning. It has two major elements:

1. feature transformation
2. feature subset selection

Feature transformation transforms the data – structured or unstructured, into a new set of features which can represent the underlying problem which machine learning is trying to solve. There are two variants of feature transformation:

1. feature construction
2. feature extraction

Both are sometimes known as feature discovery.

Feature construction process discovers missing information about the relationships between features and augments the feature space by creating additional features. Hence, if there are 'n' features or dimensions ina data set, after feature construction 'm' more features ordimensions may get added. So at the end, the data set will become '$n + m$' dimensional.

Feature extraction is the process of extracting or creating a new set of features from the original set offeatures using some functional mapping.

Unlike feature transformation, in case of **feature subset selection** (or simply **feature selection**) no new feature is generated. The objective of feature selection is to derive a subset of features from the full feature set which is most meaningful in the context of a specific machine learning problem. So, essentially the jobof feature selection is to derive a subset F_j (F_1, F_2, …, F_m) of F_i (F_1, F_2, …, F_n), where $m < n$, such that F_j is most meaningful and gets the best result for a machine learning problem. We will discuss these concepts in detail in the next section.

Points to Ponder

Data scientists and machine learning practitioners spend significant amount of time in different featureengineering activities. Selecting the right features has a critical role to play in the success of a machine learning model.

It is quite evident that feature construction expands the feature space, while feature extraction and feature selection reduces the feature space.

FEATURE TRANSFORMATION

Engineering a good feature space is a crucial prerequisitefor the success of any machine learning model. However,often it is not clear which feature is more important. For that reason, all available attributes of the data set are used as features and the problem of identifying the important features is left to the learning model. This is definitely not a feasible approach, particularly for certaindomains e.g. medical image classification, text categorization, etc. In case a model has to be trained to classify a document as spam or non-spam, we can represent a document as a bag of words. Then the featurespace will contain all unique words occurring across all documents. This will easily be a feature space of a few hundred thousand features. If we start including bigramsor trigrams along with words, the count of features will run in millions. To deal with this problem, feature transformation comes into play. Feature transformation is used as an effective tool for dimensionality reduction and hence for boosting learning model performance.

Broadly, there are two distinct goals of featuretransformation:

- Achieving best reconstruction of the original features in the data set
- Achieving highest efficiency in the learning task

Did you know?

In the field of natural language processing, 'n-gram' is a contiguous set of n items for example words in a text block or document. Using numerical prefixes, n-gram of size 1 is called unigram (i.e. a single word), size 2 is called bigram (i.e. a two-word phrase), size 3 is called trigram(i.e, a three-word phase)etc.

Feature construction

Feature construction involves transforming a given set of input features to generate a new set of more powerful features. To understand more clearly, let's take the example of a real estate data set having details of all apartments sold in a specific region.

The data set has three features – apartment length, apartment breadth, and price of the apartment. If it is used as an input to a regression problem, such data can be training data for the regression model. So given the training data, the model should be able to predict the price of an apartment whose price is not known or which has just come up for sale. However, instead of using length and breadth of the apartment as a predictor, it is much convenient and makes more sense to use the area of the apartment, which is not an existing feature of the data set. So such a feature, namely apartment area, can be added to the data set. In other words, we transform the three-dimensional data set to a four-dimensional data set, with the newly 'discovered' feature apartment area being added to the original data set. This is depicted in Figure 4.2.

apartment_length	apartment_breadth	apartment_price		apartment_length	apartment_breadth	apartment_area	apartment_price
80	59	23,60,000		80	59	4,720	23,60,000
54	45	12,15,000		54	45	2,430	12,15,000
78	56	21,84,000		78	56	4,368	21,84,000
63	63	19,84,000		63	63	3,969	19,84,500
83	74	30,71,000		83	74	6,142	30,71,000
92	86	39,56,000		92	86	7912	39,56,000

FIG. 4.2 Feature construction (example 1)

Note: Though for the sake of simplicity the features apartment length and apartment breadth have been retained in Figure 4.2, in reality, it makes more sense to exclude these features when building the model.

There are certain situations where feature construction is an essential activity before we can start with the machine learning task. These situations are

- when features have categorical value and machine learning needs numeric value inputs
- when features having numeric (continuous) values and need to be converted to ordinal values
- when text-specific feature construction needs to be done

Encoding categorical (nominal) variables

Let's take the example of another data set on athletes, as presented in Figure 4.3a. Say the data set has features age, city of origin, parents athlete (i.e. indicate whether any one of the parents was an athlete) and Chance of Win. The feature chance of a win is a class variable while the others are predictor variables. We know that any machine learning algorithm, whether it's a classification algorithm (like kNN) or a regression algorithm, requires numerical figures to learn from. So there are three features – City of origin, Parents athlete,

and Chance of win, which are categorical in nature and cannot be used by any machine learning task.

In this case, feature construction can be used to create new dummy features which are usable by machine learning algorithms. Since the feature 'City of origin' has three unique values namely City A, City B, and City C, three dummy features namely origin_ city_A, origin_city_B, and origin_city_C is created. In the same way, dummy features parents_athlete_Y and parents_athlete_N are created for feature 'Parents athlete' and win_chance_Y and win_chance_N are created for feature 'Chance of win'. The dummy features have value 0 or 1 based on the categorical value for the original feature in that row. For example, the second rowhad a categorical value 'City B' for the feature 'City of origin'. So, the newly created features in place of 'City of origin', i.e. origin_city_A, origin_city_B and origin_city_C will have values 0, 1 and 0, respectively. In the same way, parents_athlete_Y and parents_athlete_N will have values 0 and 1, respectively in row 2 as the original feature 'Parents athlete' had a categorical value 'No' in row 2. The entire set of transformation for athletes' data set is shown in Figure 4.3b.

Age (Years)	City of origin	Parents athlete	Chance of win
18	City A	Yes	Y
20	City B	No	Y
23	City B	Yes	Y
19	City A	No	N
18	City C	Yes	N
22	City B	Yes	Y

(a)

Age (Years)	origin_city_A	origin_city_B	origin_city_C	parents_athlete_Y	parents_athlete_N	win_chance_Y	win_chance_N
18	1	0	0	1	0	1	0
20	0	1	0	0	1	1	0
23	0	1	0	1	0	1	0
19	1	0	0	0	1	0	1
18	0	0	1	1	0	0	1
22	0	1	0	1	0	1	0

(b)

Age (Years)	origin_city_A	origin_city_B	origin_city_C	parents_athlete_Y	win_chance_Y
18	1	0	0	1	1
20	0	1	0	0	1
23	0	1	0	1	1
19	1	0	0	0	0
18	0	0	1	1	0
22	0	1	0	1	1

(c)

FIG. 4.3 Feature construction (encoding nominal variables)

However, examining closely, we see that the features 'Parents athlete' and 'Chance of win' in the original data set can have two values only. So creating two features from them is a kind of duplication, since the value of onefeature can be decided from the value of the other. To avoid this duplication, we can just leave one feature and eliminate the other, as shown in Figure 4.3c

Encoding categorical (ordinal) variables

Let's take an example of a student data set. Let's assumethat there are three variable – science marks, maths marks and grade as shown in Figure 4.4a. As we can see,the grade is an ordinal variable with values A, B, C, andD. To transform this variable to a numeric variable, we can create a feature num_grade mapping a numeric value against each ordinal value. In the context of the current example, grades A, B, C, and D in Figure 4.4a ismapped to values 1, 2, 3, and 4 in the transformed variable shown in Figure 4.4b.

marks_science	marks_maths	Grade
78	75	B
56	62	C
87	90	A
91	95	A
45	42	D
62	57	B

(a)

marks_science	marks_maths	num_grade
78	75	2
56	62	3
87	90	1
91	95	1
45	42	4
62	57	2

(b)

FIG. 4.4 Feature construction (encoding ordinal variables)

Transforming numeric (continuous) features tocategorical features

Sometimes there is a need of transforming a continuous numerical variable into a categorical variable. For example, we may want to treat the real estate price prediction problem, which is a regression problem, as a real estate price category prediction, which is a classification problem. In that case, we can 'bin' the numerical data into multiple categories based on the data range. In the context of the real estate price prediction example, the original data set has a numericalfeature apartment_price as shown in Figure 4.5a. It can be transformed to a categorical variable price-grade either as shown in Figure 4.5b or as shown in Figure 4.5c.

Text-specific feature construction

In the current world, text is arguably the most predominant medium of communication. Whether we think about social networks like Facebook or micro- blogging channels like Twitter or emails or short messaging services such as Whatsapp, text plays a major role in the flow of information. Hence, text mining is animportant area of research – not only for technology practitioners but also for industry practitioners.

However, making sense of text data, due to the inherentunstructured nature of the data, is not so straightforward. In the first place, the text data chunks that we can think about do not have readily available features, like structured data sets, on which machine learning tasks can be executed. All machine learning models need numerical data as input. So the text data inthe data sets need to be transformed into numerical features.

apartment_ area	apartment_ price
4,720	23,60,000
2,430	12,15,000
4,368	21,84,000
3,969	19,84,500
6,142	30,71,000
7,912	39,56,000

(a)

apartment_ area	apartment_ grade
4,720	Medium
2,430	Low
4,368	Medium
3,969	Low
6,142	High
7,912	High

(b)

apartment_ area	apartment_ grade
4,720	2
2,430	1
4,368	2
3,969	1
6,142	3
7,912	3

(c)

FIG. 4.5 Feature construction (numeric to categorical)

Text data, or corpus which is the more popular keyword, is converted to a numerical representation following a process is known as vectorization. In this process, word occurrences in all documents belonging tothe corpus are consolidated in the form of bag-of-words.There are three major steps that are followed:

1. tokenize
2. count
3. normalize

In order to tokenize a corpus, the blank spaces and punctuations are used as delimiters to separate out the words, or tokens. Then the number of occurrences of each token is counted, for each document. Lastly, tokens are weighted with reducing importance when they occur in the majority of the documents. A matrix is then formed with each token representing a column and a specific document of the corpus representing each row. Each cell contains the count of occurrence of the token ina specific document. This matrix is known as a document-term matrix (also known as a term-document matrix). Figure 4.6 represents a typical document-term matrix which forms an input to a machine learning model.

This	House	Build	Feeling	Well	Theatre	Movie	Good	Lonely	...
2	1	1	0	0	1	1	1	0	
0	0	0	1	1	0	0	0	0	
1	0	0	2	1	1	0	0	1	
0	0	0	0	1	0	1	1	0	
.	.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	.	

FIG. 4.6 Feature construction (text-specific)

Feature extraction

In feature extraction, new features are created from a combination of original features. Some of the commonlyused operators for combining the original features include

1.For Boolean features: Conjunctions, Disjunctions, Negation, etc.

2.For nominal features: Cartesian product, M of N, etc.

3.For numerical features: Min, Max, Addition, Subtraction, Multiplication, Division, Average, Equivalence, Inequality, etc.

Let's take an example and try to understand. Say, we have a data set with a feature set F_i $(F_1, F_2, …, F_n)$. After feature extraction using a mapping function f $(F_1, F_2, …, F_n)$ say, we will have a set of features

$\acute{F_i}(\acute{F_1}, \acute{F_2}, …, \acute{F_m})$ such that $\acute{F_i} = f(F_i)$ and $m < n$. For example, . This is depicted in Figure4.7. $\acute{F_1} = k_1 F_1 + k_2 F_2$

Feat$_A$	Feat$_B$	Feat$_C$	Feat$_D$		Feat$_1$	Feat$_2$
34	34.5	23	233		41.25	185.80
44	45.56	11	3.44		54.20	53.12
78	22.59	21	4.5	$\rightarrow$	43.73	35.79
22	65.22	11	322.3		65.30	264.10
22	33.8	355	45.2		37.02	238.42
11	122.32	63	23.2		113.39	167.74

$$Feat_1 = 0.3 \times Feat_A + 0.9 \times Feat_A$$
$$Feat_2 = Feat_A + 0.5\,Feat_B + 0.6 \times Feat_C$$

FIG. 4.7 Feature extraction

Let's discuss the most popular feature extraction algorithms used in machine learning:

Principal Component Analysis

Every data set, as we have seen, has multiple attributes or dimensions – many of which might have similarity with each other. For example, the height and weight of a person, in general, are quite related. If the height is more, generally weight is more and vice versa. So if a data set has height and weight as two of the attributes, obviously they are expected to be having quite a bit of similarity. In general, any machine learning algorithm performs better as the number of related attributes or features reduced. In other words, a key to the success of machine learning lies in the fact that the features are less in number as well as the similarity between each other is very less. This is the main guiding philosophy of principal component analysis (PCA) technique of feature extraction.

In PCA, a new set of features are extracted from the original features which are quite dissimilar in nature. So an n-dimensional feature space gets transformed to an m-dimensional feature space, where the dimensions are orthogonal to each other, i.e. completely independent of each other. To understand the concept of orthogonality, we have to step back and do a bit of dip dive into vector space concept in linear algebra.

We all know that a vector is a quantity having both magnitude and direction and hence can determine the position of a point relative to another point in the Euclidean space (i.e. a two or three or 'n' dimensional space). A vector space is a set of vectors. Vector spaces have a property that they can be represented as a linear combination of a smaller set of vectors, called basis vectors. So, any vector 'v' in a vector space can be represented as

$$v = \sum_{i=1}^{n} a_i u_i$$

where, a_i represents 'n' scalars and u_i represents the basis vectors. Basis vectors are orthogonal to each other. Orthogonality of vectors in n-dimensional vector space can be thought of an extension of the vectors being perpendicular in a two-dimensional vector space. Two orthogonal vectors are completely unrelated or independent of each other. So the transformation of a set of vectors to the corresponding set of basis vectors such that each vector in the original set can be expressed as a linear combination of basis vectors helps in decomposing the vectors to a number of independent components.

Now, let's extend this notion to the feature space of a data set. The feature vector can be transformed to a vector space of the basis vectors which are termed as principal components. These principal components, just like the basis vectors, are orthogonal to each other. So a set of feature vectors which may have similarity with each other is transformed to a set of principal components which are completely unrelated. However, the principal components capture the variability of the original feature space. Also, the number of principal component derived, much like the basis vectors, is much smaller than the original set of features.

The objective of PCA is to make the transformation in such a way that

1. The new features are distinct, i.e. the covariance between the new features, i.e. the principal components is 0.

2. The principal components are generated in order of the variability in the data that it captures. Hence, the first principal component should capture the maximum variability, the second principal component should capture the next highest variability etc.

3. The sum of variance of the new features or the principal components should be equal to the sum of variance of the original features.

PCA works based on a process called eigenvalue decomposition of a covariance matrix of a data set. Below are the steps to be followed:

1. First, calculate the covariance matrix of a data set.

2. Then, calculate the eigenvalues of the covariance matrix.

3. The eigenvector having highest eigenvalue represents the direction in which there is the highest variance. So this will help in identifying the first principal component.

4. The eigenvector having the next highest eigenvalue represents the direction in which data has the highest remaining variance and also orthogonal to the first direction. So this helps in identifying the second principal component.

5. Like this, identify the top 'k' eigenvectors having top 'k' eigenvalues so as to get the 'k' principal components.

Singular value decomposition

Singular value decomposition (SVD) is a matrix factorization technique commonly used in linear algebra. SVD of a matrix A ($m \times n$) is a factorization of the form:

$$A = U \sum V$$

where, U and V are orthonormal matrices, U is an $m \times m$ unitary matrix, V is an $n \times n$ unitary matrix and $\sum$ is an $m \times n$ rectangular diagonal matrix. The diagonal entries of $\sum$ are known as singular values of matrix A. The columns of U and V are called the left-singular and right-singular vectors of matrix A, respectively.

SVD is generally used in PCA, once the mean of each variable has been removed. Since it is not always advisable to remove the mean of a data attribute, especially when the data set is sparse (as in case of text data), SVD is a good choice for dimensionality reduction in those situations.

SVD of a data matrix is expected to have the properties highlighted below:

1. Patterns in the attributes are captured by the right-singular vectors, i.e. the columns of V.

2. Patterns among the instances are captured by the left-singular, i.e. the columns of U.

3. Larger a singular value, larger is the part of the matrix A that it accounts for and its associated vectors.

4. New data matrix with 'k' attributes is obtained using the equation

'

$$D = D \times [v_1, v_2, \ldots, v_k]$$

Thus, the dimensionality gets reduced to k ,SVD is often used in the context of text data.

Linear Discriminant Analysis

Linear discriminant analysis (LDA) is another commonlyused feature extraction technique like PCA or SVD. The objective of LDA is similar to the sense that it intends to transform a data set into a lower dimensional feature space. However, unlike PCA, the focus of LDA is not to capture the data set variability. Instead, LDA focuses on class separability, i.e. separating the features based on class separability so as to avoid over-fitting of the machine learning model.

Unlike PCA that calculates eigenvalues of the covariance matrix of the data set, LDA calculates eigenvalues and eigenvectors within a class and inter- class scatter matrices. Below are the steps to be followed:

1.Calculate the mean vectors for the individual classes.

2.Calculate intra-class and inter-class scatter matrices.

3.Calculate eigenvalues and eigenvectors for S_W and S_B, where S_W is the intra-class scatter matrix and S_B is the inter-classscatter matrix

$$S_W = \sum_{i=1}^{c} S_i;$$

$$S_i = \sum_{x \in D_i}^{n} (x - m_i)(x - m_i)^T$$

where, m_i is the mean vector of the i-th class

$$S_B = \sum_{i=1}^{c} N_i (m_i - m) (m_i - m)^T$$

where, mi is the sample mean for each class, m is the overallmean of the data set, N_i is the sample size of each class

4.Identify the top 'k' eigenvectors having top 'k' eigenvalues

FEATURE SUBSET SELECTION

Feature selection is arguably the most critical pre- processing activity in any machine learning project. It intends to select a subset of system attributes or features which makes a most meaningful contribution in a machine learning activity. Let's quickly discuss a practical example to understand the philosophy behind feature selection. Say we are trying to predict the weight of students based on past information about similar students, which is captured in a 'student weight' data set.The student weight data set has features such as Roll Number, Age, Height, and Weight. We can well understand that roll number can have no bearing, whatsoever, in predicting student weight. So we can eliminate the feature roll number and build a feature subset to be considered in this machine learning problem. The subset of features is expected to give betterresults than the full set. The same has been **depicted i**n <u>Figure 4.8</u>.

Roll Number	Age	Height	Weight		Age	Height	Weight
12	12	1.1	23		12	1.1	23
14	11	1.05	21.6		11	1.05	21.6
19	13	1.2	24.7		13	1.2	24.7
32	11	1.07	21.3		11	1.07	21.3
38	14	1.24	25.2		14	1.24	25.2
45	12	1.12	23.4		12	1.12	23.4

FIG. 4.8 Feature selection

But before we go forward with more detailed discussion on feature selection, let's try to understand the issues which have made feature selection such a relevant problem to be solved.

Issues in high-dimensional data

With the rapid innovations in the digital space, the volume of data generated has increased to an unbelievable extent. At the same time, breakthroughs in the storage technology area have made storage of large quantity of data quite cheap. This has further motivated the storage and mining of very large and high- dimensionality data sets.

Points to Ponder

'High-dimensional' refers to the high number of variables or attributes or features present in certain data sets, more so in the domains like DNA analysis, geographic information systems (GIS), social networking, etc. The high-dimensional spaces often have hundreds or thousands of dimensions or attributes, e.g. DNA microarray data can have up to 450,000 variables (gene probes).

Alongside, two new application domains have seen drastic development. One is that of biomedical research, which includes gene selection from microarray data. The other one is text categorization which deals with huge volumes of text data from social networking sites, emails, etc. The first domain, i.e. biomedical research generates data sets having a number of features in the range of a few tens of thousands. The text data generated from different sources also have extremely high dimensions.

In a large document corpus having few thousand documents embedded, the number of unique word tokens which represent the feature of the text data set, can also be in the range of a few tens of thousands. To get insight from such high-dimensional data may be a big challenge for any machine learning algorithm. On one hand, very high quantity of computational resources and high amount of time will be required. On the other hand the performance of the model – both for supervised and unsupervised machine learning task, also degrades sharply due to unnecessary noise in the data. Also, a model built on an extremely high number of features may be very difficult to understand. For this reason, it is necessary to take a subset of the features instead of the full set.

The objective of feature selection is three-fold:

- Having faster and more cost-effective (i.e. less need for computational resources) learning model
- Improving the efficiency of the learning model
- Having a better understanding of the underlying model that generated the data

Key drivers of feature selection – feature relevance and redundancy

Feature relevance

In supervised learning, the input data set which is the training data set, has a class label attached. A model is inducted based on the training data set – so that the inducted model can assign class labels to new, unlabelled data. Each of the predictor variables, is expected to contribute information to decide the value of the class label. In case a variable is not contributing any information, it is said to be irrelevant. In case the information contribution for prediction is very little, the variable is said to be weakly relevant. Remaining variables, which make a significant contribution to the prediction task are said to be strongly relevant variables.

In unsupervised learning, there is no training data set or labelled data. Grouping of similar data instances are done and similarity of data instances are evaluated based on the value of different variables. Certain variables do not contribute any useful information for deciding the similarity of dissimilarity of data instances. Hence, those variables make no significant information contribution in the grouping process. These variables are marked as irrelevant variables in the context of the unsupervised machine learning task.

To get a perspective, we can think of the simple example of the student data set that we discussed at the beginning of this section. Roll number of a student doesn't contribute any significant information in predicting what the Weight of a student would be.

Similarly, if we are trying to group together students with similar academic capabilities, Roll number can really not contribute any information whatsoever. So, in context of the supervised task of predicting student Weight or the unsupervised task of grouping students with similar academic merit, the variable Roll number is quite irrelevant.

Any feature which is irrelevant in the context of a machine learning task is a candidate for rejection when we are selecting a subset of features. We can consider whether the weakly relevant features are to be rejected or not on a case-to-case basis.

Feature redundancy

A feature may contribute information which is similar to the information contributed by one or more other features. For example, in the weight prediction problem referred earlier in the section, both the features Age and Height contribute similar information. This is because with an increase in Age, Weight is expected to increase. Similarly, with the increase of Height also Weight is expected to increase. Also, Age and Height increase with each other. So, in context of the Weight prediction problem, Age and Height contribute similar information. In other words, irrespective of whether the feature height is present as a part of the feature subset, the learning model will give almost same results. In the same way, without age being part of the predictor variables, the outcome of the learning model will be more or

less same.In this kind of a situation when one feature is similar to another feature, the feature is said to be potentially redundant in the context of the learning problem.

All features having potential redundancy are candidates for rejection in the final feature subset. Only asmall number of representative features out of a set of potentially redundant features are considered for being apart of the final feature subset.

So, in a nutshell, the main objective of feature selection is to remove all features which are irrelevant and take a representative subset of the features which arepotentially redundant. This leads to a meaningful featuresubset in context of a specific learning task.

Now, the question is how to find out which of the features are irrelevant or which features have potential redundancy. For that multiple measures are being used, some of which have been covered in the next sub-section.

Measures of feature relevance and redundancy

Measures of feature relevance

As mentioned earlier, feature relevance is to be gauged by the amount of information contributed by a feature. For supervised learning, mutual information is considered as a good measure of information contribution of a feature to decide the value of the class label. That's why it is a good indicator of the relevance of a feature with respect to the class variable. Higher the value of mutual information of a feature, more relevant isthat feature. Mutual information can be calculated as follows:

$$MI(C, f) = H(C) + H(f) - H(C, f)$$

where, marginal entropy of the class, $H(C) = -\sum_{i=1}^{k} p(C_i) \log_2 p(C_i)$
marginal entropy of the feature 'x', $H(f) = -\sum_c p(f = x) \log_2 p(f = x)$

and K = number of classes, C = class variable, f =feature set that take discrete values.

In case of unsupervised learning, there is no class variable. Hence, feature-to-class mutual information cannot be used to measure the information contributionof the features. In case of unsupervised learning, the entropy of the set of features without one feature at a time is calculated for all the features. Then, the features are ranked in a descending order of information gain from a feature and top 'β' percentage (value of 'β' is a design parameter of the algorithm) of features are selected as relevant features. The entropy of a feature f is calculated using Shannon's formula below:

$$H(f) = -\sum_x p(f = x) \log_2 p(f = x)$$

$\sum_x$ is used only for features that take discrete values.

For continuous features, it should be replaced by discretization performed first to estimate probabilities$p(f = x)$.

Measures of Feature redundancy

Feature redundancy, as we have already discussed, is based on similar information contribution by multiplefeatures. There are multiple measures of similarity of information contribution, salient ones being

1.Correlation-based measures

2.Distance-based measures, and

3.Other coefficient-based measure

1.Correlation-based similarity measure

Correlation is a measure of linear dependency betweentwo random variables. Pearson's product moment correlation coefficient is one of the most popular and accepted measures of correlation between two randomvariables. For two random feature variables F_1 and F_2, Pearson correlation coefficient is defined as:

$$\alpha = \frac{cov(F_1, F_2)}{\sqrt{var(F_1).var(F_2)}}$$

$$cov(F_1, F_2) = \sum (F_{1_i} - \overline{F_1}).(F_{2_i} - \overline{F_2})$$

$$var(F_1) = \sum (F_{1_i} - \overline{F_1})^2, \text{where } \overline{F_1} = \frac{1}{n}.\sum F_{1_i}$$

$$var(F_2) = \sum (F_{2_i} - \overline{F_2})^2, \text{where } \overline{F_2} = \frac{1}{n}.\sum F_{2_i}$$

Correlation values range between +1 and −1. A correlation of 1 (+ / −) indicates perfect correlation, i.e. the two features having a perfect linear relationship. In case the correlation is 0, then the features seem to have no linear relationship. Generally, for all feature selection problems, a threshold value is adopted to decide whether two features have adequate similarity or not.

2.Distance-based similarity measure

The most common distance measure is the **Euclidean distance**, which, between two features F_1 and F_2 are calculated as:

$$d(F_1, F_2) = \sqrt{\sum_{i=1}^{n} (F_{1_i} - F_{2_i})^2}$$

where F_1 and F_2 are features of an n-dimensional data set. Refer to the Figure 4.9. The data set has two features, aptitude (F_1) and communication (F_2) under consideration. The Euclidean distance between the features has been calculated using the formula provided above.

Aptitude (F_1)	Communication (F_2)	($F_1 - F_2$)	($F_1 - F_2$)^2
2	6	−4	16
3	5.5	−2.5	6.25
6	4	2	4
7	2.5	4.5	20.25
8	3	5	25
6	5.5	0.5	0.25
6	7	−1	1
7	6	1	1
8	6	2	4
9	7	2	4
			81.75

FIG. 4.9 Distance calculation between features

A more generalized form of the Euclidean distance isthe **Minkowski distance**, measured as

$$d(F_1, F_2) = \sqrt{\sum_{i=1}^{n} (F_{1_i} - F_{2_i})^r}$$

Minkowski distance takes the form of Euclidean distance (also called **L₂ norm**) when $r=2$. At $r = 1$, it takes the form of **Manhattan distance** (also called **L₁ norm**), as shown below:

$$d(F_1, F_2) = \sum_{i=1}^{n} |F_{1_i} - F_{2_i}|$$

A specific example of Manhattan distance, used morefrequently to calculate the distance between binary vectors is the **Hamming distance**. For example, the Hamming distance between two vectors 01101011 and 11001001 is 3, as illustrated in Figure 4.10a.

3.Other similarity measures

Jaccard index/coefficient is used as a measure of similarity between two features. The **Jaccard distance**, a measure of dissimilarity between two features, iscomplementary of Jaccard index.

(a) Hamming distance measurement

(b) Jaccard coefficient measurement

(c) SMC measurement
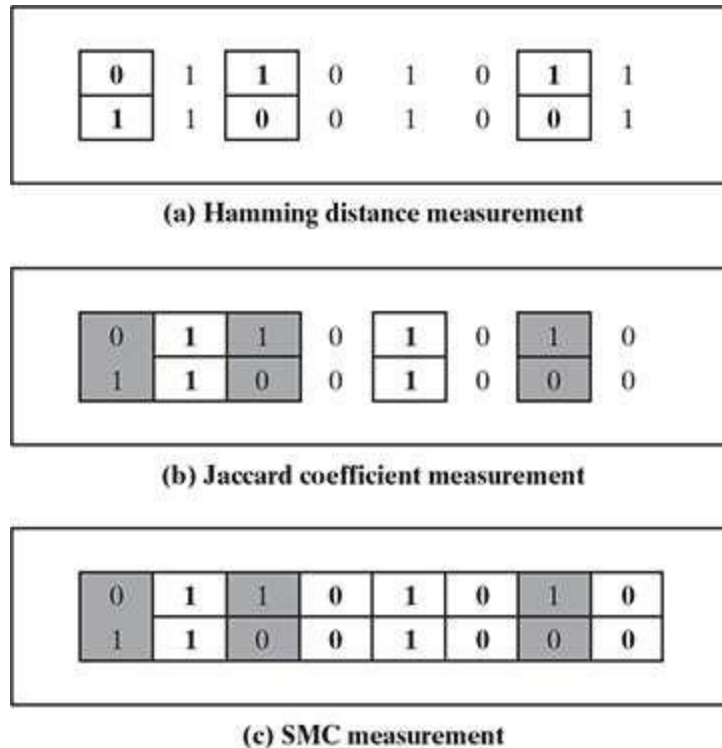
FIG. 4.10 Distance measures between features

For two features having binary values, Jaccard index is measured as

$$J = \frac{n_{11}}{n_{01} + n_{10} + n_{11}}$$

where, n_{11} = number of cases where both the features have value 1

n_{01} = number of cases where the feature 1 has value 0 and feature 2 has value 1

n_{10} = number of cases where the feature 1 has value 1 and feature 2 has value 0

Jaccard distance, $dJ = 1 - J$

Let's consider two features F_1 and F_2 having values (0,1, 1, 0, 1, 0, 1, 0) and (1, 1, 0, 0, 1, 0, 0, 0). Figure 4.10b shows the identification of the values of n_{11}, n_{01} and n_{10}. As shown, the cases where both the values are 0 have been left out without border – as an indication of the fact that they will be excluded in the calculation of Jaccard coefficient.

Jaccard coefficient of F_1 and F_2, $J =$

$$\frac{n_{11}}{n_{01} + n_{10} + n_{11}} = \frac{2}{1 + 2 + 2} = \frac{2}{5} \text{ or } 0.4.$$

∴ Jaccard distance between F_1 and F_2, $dJ = 1 - J = \frac{1}{2}$ or 0.6.

Simple matching coefficient (SMC) is almost same as Jaccard coefficient except the fact that it includes a number of cases where both the features have a value of 0.

$$SMC = \frac{n_{11} + n_{00}}{n_{00} + n_{01} + n_{10} + n_{11}}$$

where, n_{11} = number of cases where both the featureshave value 1

n_{01} = number of cases where the feature 1 has value 0and feature 2 has value 1

n_{10} = number of cases where the feature 1 has value 1 andfeature 2 has value 0

n_{00} = number of cases where both the features havevalue 0

Quite understandably, the total count of rows, $n = n_{00} + n_{01} + n_{10} + n_{11}$. As shown in Figure 4.10c, all valueshave been included in the calculation of SMC.

$$\therefore SMC \text{ of } F_1 \text{ and } F_2 = \frac{n_{11} + n_{00}}{n_{00} + n_{01} + n_{10} + n_{11}} = \frac{2 + 3}{3 + 1 + 2 + 2} = \frac{1}{2} \text{ or } 0.5.$$

One more measure of similarity using similarity coefficient calculation is **Cosine Similarity**. Let's take the example of a typical text classification problem. The text corpus needs to be first transformed into features with a word token being a feature and the number of times the word occurs in a document comes as a value in each row. There are thousands of features in such a text data set. However, the data set is sparse in nature as onlya few words do appear in a document, and hence in a rowof the data set. So each row has very few non-zero values.However, the non-zero values can be anything integer value as the same word may occur any number of times. Also, considering the sparsity of the data set, the 0-0 matches (which obviously is going to be pretty high) need to be ignored. Cosine similarity which is one of the most popular measures in text classification is calculated as:

$$\cos(x, y) = \frac{x.y}{\|x\| \cdot \|y\|}$$

where, $x.y$ = vector dot product of x and y = $\sum_{i=1}^{n} x_i y_i$

$$\|x\| = \sqrt{\sum_{i=1}^{n} x_i^2} \text{ and } \|y\| = \sqrt{\sum_{i=1}^{n} y_i^2}$$

Let's calculate the cosine similarity of x and y, where $x = (2, 4, 0, 0, 2, 1, 3, 0, 0)$ and $y = (2, 1, 0, 0, 3, 2, 1, 0, 1)$.

In this case, $x.y = 2*2 + 4*1 + 0*0 + 0*0 + 2*3 + 1*2 + 3*1 + 0*0 + 0*1 = 19$

$$\|x\| = \sqrt{2^2 + 4^2 + 0^2 + 0^2 + 2^2 + 1^2 + 3^2 + 0^2 + 0^2} = \sqrt{34} = 5.83$$

$$\|y\| = \sqrt{2^2 + 1^2 + 0^2 + 0^2 + 3^2 + 2^2 + 1^2 + 0^2 + 1^2} = \sqrt{20} = 4.47$$

$$\therefore \cos(x, y) = \frac{19}{5.83*4.47} = 0.729$$

Cosine similarity actually measures the angle (refer to Fig. 4.11) between x and y vectors. Hence, if cosine similarity has a value 1, the angle between x and y is $0°$ which means x and y are same except for the magnitude.If cosine similarity is 0, the angle between x and y is

90°.Hence, they do not share any similarity (in case of text data, no term/word is common). In the above example, the angle comes to be 43.2°.
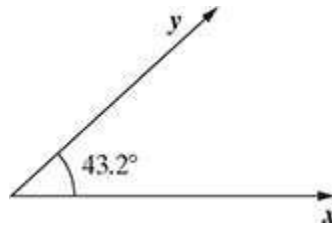

FIG. 4.11 Cosine similarity

Overall feature selection process

Feature selection is the process of selecting a subset of features in a data set. As depicted in Figure 4.12, a typicalfeature selection process consists of four steps:

1. generation of possible subsets
2. subset evaluation
3. stop searching based on some stopping criterion
4. validation of the result
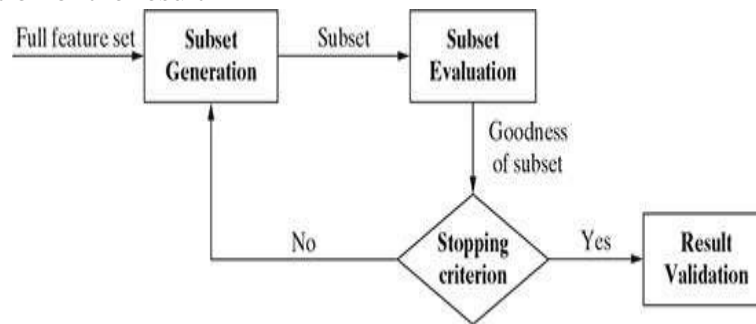
FIG. 4.12 Feature selection process

Subset generation, which is the first step of any feature selection algorithm, is a search procedure whichideally should produce all possible candidate subsets.

However, for an n-dimensional data set, 2^n subsets can be generated. So, as the value of 'n' becomes high, finding an optimal subset from all the 2^n candidate subsets becomes intractable. For that reason, different approximate search strategies are employed to find candidate subsets for evaluation. On one hand, the search may start with an empty set and keep adding features. This search strategy is termed as a sequential forward selection. On the other hand, a search may start with a full set and successively remove features. This strategy is termed as sequential backward elimination. Incertain cases, search start with both ends and add and remove features simultaneously. This strategy is termed as a bi-directional selection.

Each candidate subset is then evaluated and comparedwith the previous best performing subset based on certain **evaluation criterion**. If the new subset performs better, it replaces the previous one.

This cycle of subset generation and evaluation continues till a pre-defined **stopping criterion** is fulfilled. Some commonly used stopping criteria are

1.the search completes

2.some given bound (e.g. a specified number of iterations) is reached

3.subsequent addition (or deletion) of the feature is not producing a better subset

4.a sufficiently good subset (e.g. a subset having better classification accuracy than the existing benchmark) is selected

Then the selected best subset is **validated** either against prior benchmarks or by experiments using real-life or synthetic but authentic data sets. In case of supervised learning, the accuracy of the learning model may be the performance parameter considered for validation. The accuracy of the model using the subset derived is compared against the model accuracy of the subset derived using some other benchmark algorithm. In case of unsupervised, the cluster quality may be the parameter for validation.

Feature selection approaches

There are four types of approach for feature selection:

1. Filter approach
2. Wrapper approach
3. Hybrid approach
4. Embedded approach

In the **filter approach** (as depicted in Fig. 4.13), the feature subset is selected based on statistical measures done to assess the merits of the features from the data perspective. No learning algorithm is employed to evaluate the goodness of the feature selected. Some of the common statistical tests conducted on features as a part of filter approach are – Pearson's correlation, information gain, Fisher score, analysis of variance (ANOVA), Chi-Square, etc.
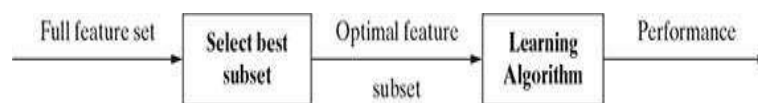
FIG. 4.13 Filter approach

In the **wrapper approach** (as depicted in Fig. 4.14), identification of best feature subset is done using the induction algorithm as a black box. The feature selection algorithm searches for a good feature subset using the induction algorithm itself as a part of the evaluation function. Since for every candidate subset, the learning model is trained and the result is evaluated by running the learning algorithm, wrapper approach is computationally very expensive. However, the performance is generally superior compared to filter approach.
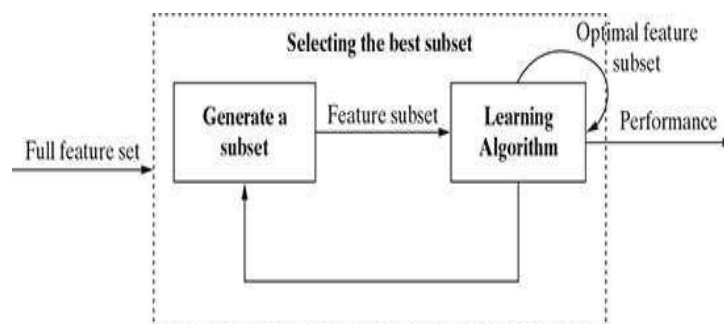

FIG. 4.14 Wrapper approach

Hybrid approach takes the advantage of both filter and wrapper approaches. A typical hybrid algorithm makes use of both the statistical tests as used in filter approach to decide the best subsets for a given cardinality and a learning algorithm to select the final best subset among the best subsets across different cardinalities.

Embedded approach (as depicted in Fig. 4.15) is quite similar to wrapper approach as it also uses and inductive algorithm to evaluate the generated feature subsets. However, the difference is it performs feature selection and classification simultaneously.
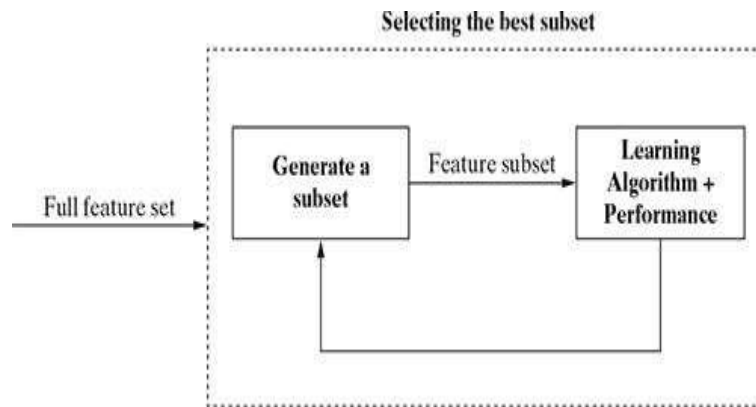
FIG. 4.15 Embedded approach

SAMPLE QUESTIONS

MULTIPLE-CHOICE QUESTIONS (1 MARK QUESTIONS)

1. Engineering a good feature space is a crucial_____for the success of any machine learning model.

   1. Pre-requisite

   2. Process

   3. Objective

   4. None of the above

2. n-gram of size 1 is called

   1. Bigram

   2. Unigram

   3. Trigram

   4. None of the above

3. Feature_____involves transforming a given set of input features to generate a new set of more powerful features.

   1. Selection

   2. Engineering

3.Transformation

4.Re-engineering

4.Conversion of a text corpus to a numerical representation is done using__process.

1.Tokenization

2.Normalization

3.Vectorization

4.None of the above

5._____approach uses induction algorithm for subset validation.

1.Filter

2.Hybrid

3.Wrapper

4.Embedded

6.In feature extraction, some of the commonly used_are used for combining the original features.

1.Operators

2.Delimiters

3.Words

4.All of the above

7.Hamming distance between binary vectors 1001 and 0101 is

1. 1

2. 2

3. 3

4. 4

8.PCA is a technique for

1.Feature extraction

2.Feature construction

3.Feature selection

4.None of the above

9.The new features created in PCA are known as

1.Principal components

2.Eigenvectors

3.Secondary components

4.None of the above

10.In LDA, intra-class and inter-classmatrices are calculated.

1.Scatter

2.Adjacency

3.Similarity

4.None of the above

11.Cosine similarity is most popularly used in

1.Text classification

2.Image classification

3.Feature selection

4.None of the above

12.This approach is quite similar to wrapper approach as it also uses and inductive algorithm to evaluate the generated feature subsets.

1.Embedded approach

2.Filter approach

3.Pro Wrapper approach

4.Hybrid approach

13.In__approach, identification of best feature subset is done using the induction algorithm as a black box.

1.Embedded

2.Filter

3.Wrapper

4.Hybrid

SHORT-ANSWER TYPE QUESTIONS (5 MARKS QUESTIONS)

1.What is a feature? Explain with an example.

2.What are the different situations which necessitate feature construction?

3.Explain the process of encoding nominal variables.

4.Explain the process of transforming numeric features to categorical features.

5.Explain the wrapper approach of feature selection. What are the merits and de-merits of this approach?

6.When can a feature be termed as irrelevant? How can it be measured?

7.When can a feature be termed as redundant? What are the measures to determine the potentially redundant features?

8.What are the different distance measures that can be used to determine similarity of features?

9.Compare Euclidean distance with Manhattan distance?

10.Differentiate feature transformation with feature selection

11.Write short notes on any <u>two</u>:

  1.SVD

  2.Hybrid method of feature selection

  3.Silhouette width

  4.ROC curve

LONG-ANSWER TYPE QUESTIONS (10 MARKS QUESTIONS)

1.What is feature engineering? Explain, in details, the different aspects of feature engineering?

2.What is feature selection? Why is it needed? What are the different approaches of feature selection?

3.Explain the filter and wrapper approaches of feature selection. What are the merits and demerits of these approaches?

4.1. Explain the overall process of feature selection

   2.Explain, with an example, the main underlying concept of feature extraction. What are the most popular algorithms for feature extraction?

5.Explain the process of feature engineering in context of a text categorization problem.

6.Why is cosine similarity a suitable measure in context of text categorization? Two rows in a document-term matrix have values - (2, 3, 2, 0, 2, 3, 3, 0, 1) and (2, 1, 0, 0, 3, 2, 1, 3, 1). Find the cosine similarity.

7.1. How can we calculate Hamming distance? Find the Hamming distance between 10001011 and 11001111.

  2.Compare the Jaccard index and similarity matching coefficient of two features having values (1, 1, 0, 0, 1, 0, 1,1) and (1, 0, 0, 1, 1, 0, 0, 1).

8.What do you understand by a high-dimensional data set? Give a few practical examples? What is the challenge while applying machine learning technique on a high-dimensional data set? How can that be addressed?

9.1. Write short notes on any <u>two</u>:

  1.PCA

  2.Vectorization

  3.Embedded method

  2.Write the difference between (any <u>two</u>):

  1.Sequential forward selection vs. sequential backward elimination

  2.Filter vs. wrapper method of feature selection

  3.Jaccard coefficient vs. SMC

UNIT-3

CHAPTER-1

BAYESIAN CONCEPT LEARNING

OBJECTIVE OF THE CHAPTER:

Principles of probability for classification are an important area of machine learning algorithms. In our practical life, our decisions are affected by our prior knowledge or belief about an event. Thus, an event that is otherwise very unlikely to occur may beconsidered by us seriously to occur in certain situations if we know that in the past, the event had certainly occurred when other events were observed.The same concept is applied in machine learning using Bayes' theorem and the related algorithms discussed in this chapter. The concepts of probabilities discussed in the previous chapters are used extensively in this chapter.

INTRODUCTION

In the last chapter, we discussed the rules of probability and possible uses of probability, distribution functions, and hypothesis testing principles in the machine learningdomain. In this chapter, we will discuss the details of the Bayesian theorem and how it provides the basis for machine learning concepts. The technique was derived from the work of the 18th century mathematician Thomas Bayes. He developed the foundational mathematical principles, known as Bayesian methods, which describe the probability of events, and more importantly, how probabilities should be revised when there is additional information available.

WHY BAYESIAN METHODS ARE IMPORTANT?

Bayesian learning algorithms, like the naive Bayes classifier, are highly practical approaches to certain typesof learning problems as they can calculate explicit probabilities for hypotheses. In many cases, they are equally competitive or even outperform the other learning algorithms, including decision tree and neural network algorithms.

Bayesian classifiers use a simple idea that the trainingdata are utilized to calculate an observed probability of each class based on feature values. When the same classifier is used later for unclassified data, it uses the observed probabilities to predict the most likely class forthe new features. The application of the observations from the training data can also be thought of as applyingour prior knowledge or prior belief to the probability of an outcome, so that it has higher probability of meeting the actual or real-life outcome. This simple concept is used in Bayes' rule and applied for training a machine in machine learning terms. Some of the real-life uses of Bayesian classifiers are as follows:

- Text-based classification such as spam or junk mail filtering, authoridentification, or topic categorization
- Medical diagnosis such as given the presence of a set of observed symptoms during a disease, identifying the probability of new patients having the disease
- Network security such as detecting illegal intrusion or anomaly in computer networks

One of the strengths of Bayesian classifiers is that they utilize all available parameters to subtly change the predictions, while many other algorithms tend to ignore the features that

have weak effects. Bayesian classifiers assume that even if few individual parameters have small effect on the outcome, the collective effect of those parameters could be quite large. For such learning tasks, the naive Bayes classifier is most effective.

Some of the features of Bayesian learning methods that have made them popular are as follows:

- Prior knowledge of the candidate hypothesis is combined with the observed data for arriving at the final probability of a hypothesis. So, two important components are the prior probability of each candidate hypothesis and the probability distribution over the observed data set for each possible hypothesis
- The Bayesian approach to learning is more flexible than the other approaches because each observed training pattern can influence the outcome of the hypothesis by increasing or decreasing the estimated probability about the hypothesis, whereas most of the other algorithms tend to eliminate a hypothesis if that is inconsistent with the single training pattern
- Bayesian methods can perform better than the other methods while validating the hypotheses that make probabilistic predictions. For example, when starting a new software project, on the basis of the demographics of the project, we can predict the probability of encountering challenges during execution of the project
- Through the easy approach of Bayesian methods, it is possible to classify new instances by combining the predictions of multiple hypotheses, weighted by their respective probabilities
- In some cases, when Bayesian methods cannot compute the outcome deterministically, they can be used to create a standard for the optimal decision against which the performance of other methods can be measured

As we discussed above, the success of the Bayesian method largely depends on the availability of initial knowledge about the probabilities of the hypothesis set. So, if these probabilities are not known to us in advance, we have to use some background knowledge, previous data or assumptions about the data set, and the related probability distribution functions to apply this method. Moreover, it normally involves high computational cost to arrive at the optimal Bayes hypothesis.

BAYES' THEOREM

Before we discuss Bayes' theorem and its application in concept learning, we should be clear about what is **concept learning**. Let us take an example of how a child starts to learn meaning of new words, e.g. 'ball'.

The child is provided with positive examples of 'objects' which are 'ball'. At first, the child may be confused with many different colours, shapes and sizes of the balls and may also get confused with some objects which look similar to ball, like a balloon or a globe. The child's parent continuously feeds her positive examples like 'that is a ball', 'this is a green ball', 'bring me that small ball', etc. Seldom there are negative examples used for such concept teaching, like 'this is a non-ball', but the parent may clear the confusion of the child when it points to a balloon and says it is a ball by saying 'that is not a ball'. But it is observed that the learning is most influenced through positive examples rather than through negative examples, and the expectation is that the child will be able to identify the object 'ball' from a wide variety of objects and different types of balls kept together once the concept of a ball is clear

to her. We can extend this example to explain how we can expect machines to learn through the feeding of positive examples, which forms the basis for concept learning.

To relate the above-mentioned learning concept with the mathematical model of Bayes, we can correlate the learning process of 'meaning of a word' as equivalent to learning, a concept using binary classification. Let us define a concept set C and a corresponding function $f(k)$. We also define $f(k) = 1$, when k is within the set C and $f(k) = 0$ otherwise. Our aim is to learn the indicator function f that defines which elements are within the set

C. So, by using the function f, we will be able to classify the element either inside or outside our concept set. In Bayes' theorem, we will learn how to use standard probability calculus to determine the uncertainty about the function f, and we can validate the classification by feeding positive examples.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

There are few notations that will be introduced before going into the details of Bayes' theorem. In <u>Chapter 5</u>, we already discussed Bayes' probability rule as given below:

where A and B are conditionally related events and $p(A|B)$ denotes the probability of event A occurring when event B has already occurred.

Let us assume that we have a training data set D where we have noted some observed data. Our task is to determine the best hypothesis in space H by using the knowledge of D.

Prior

The prior knowledge or belief about the probabilities of various hypotheses in H is called Prior in context of Bayes' theorem. For example, if we have to determine whether a particular type of tumour is malignant for a patient, the prior knowledge of such tumours becoming malignant can be used to validate our current hypothesis and is a prior probability or simply called Prior.

Let us introduce few notations to explain the concepts.

We will assume that $P(h)$ is the initial probability of a hypothesis 'h' that the patient has a malignant tumour based only on the malignancy test, without considering the prior knowledge of the correctness of the test processor the so-called training data. Similarly, $P(T)$ is the prior probability that the training data will be observed or, in this case, the probability of positive malignancy test results. We will denote $P(T|h)$ as the probability of observing data T in a space where 'h' holds true, which means the probability of the test results showing a positive value when the tumour is actually malignant.

Posterior

The probability that a particular hypothesis holds for a data set based on the Prior is called the posterior probability or simply Posterior. In the above example, the probability of the hypothesis that the patient has a malignant tumour considering the Prior of correctness of the malignancy test is a posterior probability. In our notation, we will say that we are interested in finding out $P(h|T)$, which means whether the hypothesis holds true given the observed

training data T. This is called the posterior probability or simply Posterior in machine learning language. So, the prior probability $P(h)$, which represents the probability of the hypothesis independentof the training data (Prior), now gets refined with the introduction of influence of the training data as $P(h|T)$.

According to Bayes' theorem

$$P(h|T) = \frac{P(T|h)P(h)}{P(T)}$$

combines the prior and posterior probabilitiestogether.

From the above equation, we can deduce that $P(h|T)$increases as $P(h)$ and $P(T|h)$ increases and also as $P(T)$decreases. The simple explanation is that when there ismore probability that T can occur independently of h then it is less probable that h can get support from T inits occurrence.

It is a common question in machine learning problemsto find out the maximum probable hypothesis h from a set of hypotheses H ($h{\in}H$) given the observed training data T. This maximally probable hypothesis is called the **maximum a posteriori (MAP)** hypothesis. By using Bayes' theorem, we can identify the MAP hypothesis from the posterior probability of each candidate hypothesis:

$$h_{\text{MAP}} = \text{argmax}_{h \in H} P(h|T)$$
$$= \text{argmax}_{h \in H} \frac{P(T|h)P(h)}{P(T)}$$

and as $P(T)$ is a constant independent of h, in this case,we can write

Likelihood

In certain machine learning problems, we can further simplify equation 6.1 if every hypothesis in H has equal probable priori as $P(h_i) = P(h_j)$, and then, we can determine $P(h|T)$ from the

$$= \text{argmax}_{h \in H} P(T|h)P(h) \qquad\qquad (6.1)$$

probability $P(T|h)$ only. Thus,$P(T|h)$ is called the likelihood of data T given h, and any hypothesis that maximizes $P(T|h)$ is called the maximumlikelihood (ML) hypothesis, h_{ML}. See figure 6.1 and 6.2 for the conceptual and mathematical representation of Bayes theorem and the relationship of Prior, Posterior and Likelihood.

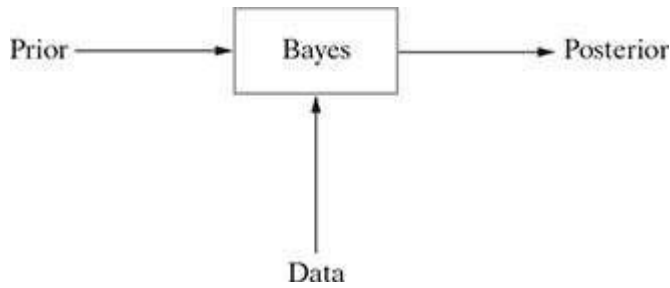$$h_{\text{ML}} = \text{argmax}_{h \in H} P(T|h) \qquad (6.2)$$

FIG. 6.1

Bayes' theorem

$$P(h|T) = \frac{P(T|h)\, P(h)}{P(T)}$$

likelihood — prior probability — posterior probability — marginal likelihood

FIG. 6.2

Concept of prior, posterior, and likelihood

> **Points to Ponder:**
>
> Arriving at the refined probability of an event in thelight of probability of a related event is a powerful concept and relates very closely with our day-to-dayhandling of events and using our knowledge to influence the decisions.

Example. Let us take the example of malignancy identification in a particular patient's tumour as an application for Bayes rule. We will calculate how the prior knowledge of the percentage of cancer cases in a sample population and probability of the test result being correct influence the probability outcome of the correct diagnosis. We have two alternative hypotheses:

a particular tumour is of malignant type and (2) aparticular tumour is non-malignant type. The priori available are—1. only 0.5% of the population has this kind of tumour which is malignant, 2. the laboratory report has some amount of incorrectness as it could detect the malignancy was present only with 98% accuracy whereas could show the malignancy was not present correctly only in 97% of cases. This means the test predicted malignancy was present which actually was a false alarm in 2% of the cases, and also missed detecting the real malignant tumour in 3% of the cases.

Solution: Let us denote Malignant Tumour = MT,

Positive Lab Test = PT, Negative Lab Test = NT

h_1 = the particular tumour is of malignant type = MT in our example

h_2 = the particular tumour is not malignant type = !MT in our example

$P(MT) = 0.005$ $P(!MT) = 0.995$

$P(PT|MT) = 0.98$ $P(PT|!MT) = 0.02$

$P(NT|!MT) = 0.97$ $P(NT|MT) = 0.03$

So, for the new patient, if the laboratory test report shows positive result, let us see if we should declare this as the malignancy case or not:

$$
\begin{aligned}
P(h_1|PT) &= \frac{P(PT|h_1).P(h_1)}{P(PT)} \\
&= P(PT|MT)P(MT) \\
&= 0.98 \times 0.005 \\
&= 0.0049 \\
&= 0.49\%
\end{aligned}
$$

$$
\begin{aligned}
P(h_2|PT) &= \frac{P(PT|h_2).P(h_2)}{P(PT)} \\
&= P(PT|!MT)P(!MT) \\
&= 0.02 \times 0.995 \\
&= 0.0199 \\
&= 1.99\%
\end{aligned}
$$

As $P(h_2|PT)$ is higher than $P(h_1|PT)$, it is clear that the hypothesis h_2 has more probability of being true. So, hMAP = $h2$ = !MT.

This indicates that even if the posterior probability of malignancy is significantly higher than that of non- malignancy, the probability of this patient not having malignancy is still higher on the basis of the prior knowledge. Also, it should be noted that through Bayes' theorem, we identified the probability of one hypothesis being higher than the other hypothesis, and we did not completely accept or reject the hypothesis by this theorem. Furthermore, there is very high dependency on the availability of the prior data for successful application of Bayes' theorem.

BAYES' THEOREM AND CONCEPT LEARNING

One simplistic view of concept learning can be that if we feed the machine with the training data, then it can calculate the posterior probability of the hypotheses and outputs the most probable hypothesis. This is also called brute-force Bayesian learning algorithm, and it is also observed that consistency in providing the right probable hypothesis by this algorithm is very comparable to the other algorithms.

Brute-force Bayesian algorithm

We will now discuss how to use the MAP hypothesis output to design a simple learning algorithm called brute-force map learning algorithm. Let us assume that the learner considers a finite hypothesis space H in which the learner will try to learn some target concept $c:X \rightarrow \{0,1\}$ where X is the instance space corresponding to H. The sequence of training examples is $\{(x_1, t_1), (x_2, t_2),\ldots, (x_m, t_m)\}$, where x_i is the instance of X and t_i is the target concept of x_i defined as $t_i = c(x_i)$. Without impacting the efficiency of the algorithm, we can assume that the sequence of instances of x $\{x_1,\ldots, x_m\}$ is held fixed, and then, the sequence of target values becomes T

$= \{t_1,\ldots, t_m\}$.

For calculating the highest posterior probability, we can use Bayes' theorem as discussed earlier in this chapter:

Calculate the posterior probability of each hypothesis h in H:

$$P(h|T) = \frac{P(T|h)P(h)}{P(T)}$$

Identify the h_{MAP} with the highest posterior probability

$h_{MAP} = \text{argmax}_{h \in H} P(h|T)$

Please note that calculating the posterior probability for each hypothesis requires a very high volume of computation, and for a large volume of hypothesis space, this may be difficult to achieve.

Let us try to connect the concept learning problem with the problem of identifying the h_{MAP}. On the basis of the probability distribution of $P(h)$ and $P(T|h)$, we can derive the prior knowledge of the learning task. There are few important assumptions to be made as follows:

The training data or target sequence T is noise free, which means that it is a direct function of X only (i.e. $t_i = c(x_i)$)

The concept c lies within the hypothesis space H

Each hypothesis is equally probable and independent of each other

On the basis of assumption 3, we can say that each hypothesis h within the space H has equal prior probability, and also because of assumption 2, we can say that these prior probabilities sum up to 1. So, we can write

$$P(h) = \frac{1}{|H|} \text{ for all } h \text{ within } H \qquad\qquad (6.3)$$

$P(T|h)$ is the probability of observing the target values t_i in the fixed set of instances $\{x_i,\ldots, x_m)$ in the space where h holds true and describes the concept c correctly.

Using assumption 1 mentioned above, we can say that if T is consistent with h, then the probability of data T given the hypothesis h is 1 and is 0 otherwise:

$$P(T|h) = \begin{cases} 1 & \text{if } t_i = h(x_i) \text{ for all } t_i \text{ within } T \\ 0 & \text{otherwise} \end{cases} \qquad (6.4)$$

Using Bayes' theorem to identify the posterior probability

$$P(h|T) = \frac{P(T|h)P(h)}{P(T)} \qquad (6.5)$$

For the cases when h is inconsistent with the training data T, using 6.5 we get

$$P(h|T) = \frac{0 \times P(h)}{P(T)} = 0$$

when h is inconsistent with T,

and when h is consistent with T

$$P(h|T) = \frac{1 \times \dfrac{1}{|H|}}{P(T)} = \frac{1}{|H|P(T)} \qquad (6.6)$$

Now, if we define a subset of the hypothesis H which is consistent with T as H_D, then by using the total probability equation, we get

$$\begin{aligned} P(T) &= \sum_{h_i \in H_D} P(T|h_i)P(h_i) \\ &= \sum_{h_i \in H_D} 1.\frac{1}{|H|} + \sum_{h_i \notin H_D} 0.\frac{1}{|H|} \\ &= \sum_{h_i \in H_D} 1.\frac{1}{|H|} \\ &= \frac{|H_D|}{|H|} \end{aligned}$$

This makes 6.5 as

$$\begin{aligned} P(h|T) &= \frac{1}{|H|.\dfrac{|H_D|}{|H|}} \\ &= \frac{1}{|H_D|} \end{aligned}$$

So, with our set of assumptions about $P(h)$ and $P(T|h)$, we get the posterior probability $P(h|T)$ as

$$P(h|T) = \begin{cases} \dfrac{1}{|H_D|} & \text{if } h \text{ is consistent with } T \\ 0 & \text{otherwise} \end{cases} \qquad (6.7)$$

where HD is the number of hypotheses from the space H which are consistent with target data set T. The interpretation of this evaluation is that initially, each hypothesis has equal probability and, as we introduce the training data, the posterior probability of inconsistent hypotheses becomes zero and the total probability that sums up to 1 is distributed equally among the consistent hypotheses in the set. So, under this condition, each consistent hypothesis is a MAP hypothesis with posterior

probability $\dfrac{1}{|H_D|}$.

Concept of consistent learners

From the above discussion, we understand the behaviour of the general class of learner whom we call as consistent learners. So, the group of learners who commit zero error over the training data and output the hypothesis are called *consistent learners*. If the training data is noise free and deterministic (i.e. $P(D|h) = 1$ if D and h are consistent and 0 otherwise) and if there is uniform prior probability distribution over H (so, $P(h_m) = P(h_n)$ for all m, n), then every consistent learner outputs the MAP hypothesis. An important application of this conclusion is that Bayes' theorem can characterize the behaviour of learning algorithms even when the algorithm does not explicitly manipulate the probability. As it can help to identify the optimal distributions of $P(h)$ and $P(T|h)$ under which the algorithm outputs the MAP hypothesis, the knowledge can be used to characterize the assumptions under which the algorithms behave optimally.

Though we discussed in this section a special case of Bayesian output which corresponds to the noise-free training data and deterministic predictions of hypotheses where $P(T|h)$ takes on value of either 1 or 0, the theorem can be used with the same effectiveness for noisy training data and additional assumptions about the probability distribution governing the noise.

Bayes optimal classifier

In this section, we will discuss the use of the MAP hypothesis to answer the question what is the most probable classification of the new instance given the training data. To illustrate the concept, let us assume three hypotheses h_1, h_2, and h_3 in the hypothesis space

H. Let the posterior probability of these hypotheses be, 0.3, and 0.3, respectively. There is a new instance x, which is classified as true by h_1, but false by h_2 and h_3.

Then the most probable classification of the new instance (x) can be obtained by combining the predictions of all hypotheses weighed by their corresponding posterior probabilities. By denoting the possible classification of the new instance as c_i from the set C, the probability $P(c_i|T)$ that the correct classification for the new instance is c_i is

$$P(c_i|T) = \sum_{h_i \in H} P(c_i|h_i)P(h_i|T)$$

The optimal classification is for which $P(c_i|T)$ is maximum is

$$\text{Bayes optimal classifier} = \underset{c_i \in C}{\text{argmax}} \sum_{h_i \in H} P(c_i|h_i)P(h_i|T)$$

Points to Ponder:

The approach in the Bayes optimal classifier is to calculate the most probable classification of each new instance on the basis of the combined predictions of all alternative hypotheses, weighted by their posterior probabilities.

So, extending the above example,

The set of possible outcomes for the new instance x is within the set $C = \{\text{True, False}\}$ and

$P(h1 \mid T) = 0.4$, $P(\text{False} \mid h1) = 0$, $P(\text{True} \mid h1) = 1$ $P(h2 \mid T) = 0.3$, $P(\text{False} \mid h2) = 1$, $P(\text{True} \mid h2) = 0$ $P(h3 \mid T) = 0.3$, $P(\text{False} \mid h3) = 1$, $P(\text{True} \mid h3) = 0$

Then,

$$\sum_{h_i \in H} P(\text{True}|h_i)P(h_i|T) = 0.4$$

$$\sum_{h_i \in H} P(\text{False}|h_i)P(h_i|T) = 0.6$$

and

$$\underset{c_i \in \{\text{True, False}\}}{\text{argmax}} \sum_{h_i \in H} P(c_i|h_i)P(h_i|T) = \text{False}$$

This method maximizes the probability that the new instance is classified correctly when the available training data, hypothesis space and the prior probabilities of the hypotheses are known. This is thus also called Bayes optimal classifier.

Naïve Bayes classifier

Naïve Bayes is a simple technique for building classifiers: models that assign class labels to problem instances. The basic idea of Bayes rule is that the outcome of a hypothesis can be predicted on the basis of some evidence (E) that can be observed.

From Bayes rule, we observed that

A prior probability of hypothesis h or $P(h)$: This is the probability of an event or hypothesis before the evidence is observed.

A posterior probability of *h* or $P(h|D)$: This is the probability of an event after the evidence is observed within the population *D*.

$$\text{Posterior probability} = \frac{(\text{Prior probability} \times \text{Conditional Probability})}{\text{Evidence}}$$

Posterior Probability is of the format 'What is the probability that a particular object belongs to class i given its observed feature values?'

For example, a person has height and weight of 182 cm and 68 kg, respectively. What is the probability that this person belongs to the class 'basketball player'? This can be predicted using the Naïve Bayes classifier. This is known as probabilistic classifications.

In machine learning, a probabilistic classifier is a classifier that can be foreseen, given a perception or information (input), a likelihood calculation over a set of classes, instead of just yielding (outputting) the most likely class that the perception (observation) should belong to. Parameter estimation for Naïve Bayes models uses the method of ML.

Bayes' theorem is used when new information can be used to revise previously determined probabilities.

Depending on the particular nature of the probability model, Naïve Bayes classifiers can be trained very professionally in a supervised learning setting.

Let us see the basis of deriving the principles of Naïve Bayes classifiers. We take a learning task where each instance *x* has some attributes and the target function ($f(x)$) can take any value from the finite set of classification values *C*. We also have a set of training examples for target function, and the set of attributes $\{a_1, a_2, \ldots, a_n\}$ for the new instance are known to us. Our task is to predict the classification of the new instance.

According to the approach in Bayes' theorem, the classification of the new instance is performed by assigning the most probable target classification C_{MAP} on the basis of the attribute values of the new instance $\{a_1, a_2, \ldots, a_n\}$. So,

$$C_{MAP} = \underset{c_i \in C}{\text{argmax}} \sum_{h_i \in H} P(c_i | a_1, a_2, \ldots, a_n)$$

which can be rewritten using Bayes' theorem as

$$C_{MAP} = \underset{c_i \in C}{\text{argmax}} \sum_{h_i \in H} \frac{P(a_1, a_2, \ldots, a_n | c_i) P(c_i)}{P(a_1, \ldots a_n)}$$

As combined probability of the attributes defining the new instance fully is always 1

$$C_{MAP} = \overset{\text{argmax}}{\underset{c_i \in C}{}} \sum_{h_i \in H} P(a_1, a_2, \ldots, a_n | c_i) P(c_i) \tag{6.8}$$

So, to get the most probable classifier, we have to evaluate the two terms $P(a_1, a_2, c, a_n | c_i)$ and $P(ci)$. In apractical scenario, it is possible to calculate $P(ci)$ by calculating the frequency of each target value c_i in the training data set. But the $P(a_1, a_2, c, a_n | c_i)$ cannot be estimated easily and needs a very high effort of calculation. The reason is that the number of these terms is equal to the product of number of possible instances and the number of possible target values, and thus, each instance in the instance space needs to be visited many times to arrive at the estimate of the occurrence. Thus, the Naïve Bayes classifier makes a simple assumption that the attribute values are conditionally independent ofeach other for the target value. So, applying this simplification, we can now say that for a target value of an instance, the probability of observing the combination $a1, a2, \ldots, a_n$ is the product of probabilities of individual attributes $P(a_i | c_j)$.

$$P(a_1, a_2, \ldots, a_n | c_j) = \overset{\Pi}{_i} P(a_i | c_j)$$

Then, from equation 6.7, we get the approach for theNaïve Bayes classifier as

$$C_{NB} = \overset{\text{argmax}}{\underset{c_i \in C}{}} \sum_{h_i \in H} P(c_i) \overset{\Pi}{_i} P(a_i | c_j) \tag{6.9}$$

Here, we will be able to compute $P(a_i | c_j)$ as we have to calculate this only for the number of distinct attributes values (a_i) times the number of distinct target values (c_j), which is much smaller set than the product of both the sets. The most important reason for the popularity of theNaïve Bayes classifier approach is that it is not required to search the whole hypothesis space for this algorithm, but rather we can arrive at the target classifier by simply counting the frequencies of various data combinations within the training example.

To summarize, a Naïve Bayes classifier is a primary probabilistic classifier based on a view of applying Bayes' theorem (from Bayesian inference with strong naive) independence assumptions. The prior probabilities in Bayes' theorem that are changed with the help of newly available information are classified as posterior probabilities.

A key benefit of the naive Bayes classifier is that it requires only a little bit of training information (data) togauge the parameters (mean and differences of the variables) essential for the classification (arrangement). In the Naïve Bayes classifier, independent variables are always assumed, and only the changes (variances) of thefactors/variables for each class should be determined and not the whole covariance matrix. Because of the rather naïve assumption that all features of the dataset are equally important and independent, this is called Naïve Bayes classifier.

Naïve Bayes classifiers are direct linear classifiers that are known for being the straightforward, yet extremely proficient result. The modified version of Naïve Bayes

classifier originates from the assumption that information collection (data set) is commonly autonomous (mutually independent). In most of the practical scenarios, the 'independence' assumption is regularly violated. However, Naïve Bayes classifiers still tend to perform exceptionally well.

Some of the key strengths and weaknesses of NaïveBayes classifiers are described in Table 6.1.

Table 6.1 Strengths and Weaknesses of Bayes Classifiers

Strengths	Weakness
Simple and fast in calculation but yet effective in result	The basis assumption of equal importance and independence often does not hold true
In situations where there are noisy and missing data, it performs well	If the target dataset contains large numbers of numeric features, then the reliability of the outcome becomes limited
Works equally well when smaller number of data is present for training as well as very large number of training data is available	Though the predicted classes have a high reliability, estimated probabilities have relatively lower reliability
Easy and straightforward way to obtain the estimated probability of a prediction	

Example. Let us assume that we want to predict the outcome of a football world cup match on the basis of the past performance data of the playing teams. We have training data available (refer Fig. 6.3) for actual match outcome, while four parameters are considered – Weather Condition (Rainy, Overcast, or Sunny), how many matches won were by this team out of the last three matches (one match, two matches, or three matches), Humidity Condition (High or Normal), and whether they won the toss (True or False). Using Naïve Bayesian, you need to classify the conditions when this team wins and then predict the probability of this team winning a particular match when Weather Conditions = Rainy, they won two of the last three matches, Humidity = Normal and they won the toss in the particular match.

Weather Condition	Wins in last 3 matches	Humidity	Win toss	Won match?
Rainy	3 wins	High	FALSE	No
Rainy	3 wins	High	TRUE	No
OverCast	3 wins	High	FALSE	Yes
Sunny	2 wins	High	FALSE	Yes
Sunny	1 win	Normal	FALSE	Yes
Sunny	1 win	Normal	TRUE	No
OverCast	1 win	Normal	TRUE	Yes
Rainy	2 wins	High	FALSE	No
Rainy	1 win	Normal	FALSE	Yes
Sunny	2 wins	Normal	FALSE	Yes
Rainy	2 wins	Normal	TRUE	Yes
OverCast	2 wins	High	TRUE	Yes
OverCast	3 wins	Normal	FALSE	Yes
Sunny	2 wins	High	TRUE	No

FIG. 6.3 Training data for the Naïve Bayesian method

6.4.4.1 Naïve Bayes classifier steps

Step 1: First construct a frequency table. A frequency table is drawn for each attribute against the target outcome. For example, in Figure 6.3, the various attributes are (1) Weather Condition, (2) How many matches won by this team in last three matches, (3) Humidity Condition, and (4) whether they won the toss and the target outcome is will they win the match or not?

Step 2: Identify the cumulative probability for 'Won match = Yes' and the probability for 'Won match = No' on the basis of all the attributes. Otherwise, simply multiply probabilities of all favourable conditions to derive 'YES' condition. Multiply probabilities of all non-favourable conditions to derive 'No' condition.

Step 3: Calculate probability through normalization by applying the below formula

$$P(\text{Yes}) = \frac{P(\text{Yes})}{P(\text{Yes}) + P(\text{No})}$$

$$P(\text{No}) = \frac{P(\text{No})}{P(\text{Yes}) + P(\text{No})}$$

P(Yes) will give the overall probability of favourable condition in the given scenario.

P(No) will give the overall probability of non-favourable condition in the given scenario.

Solving the above problem with Naïve Bayes

Step 1: Construct a frequency table. The posterior probability can be easily derived by constructing a frequency table for each attribute against the target. For example, frequency of Weather Condition variable with values 'Sunny' when the target value Won match is 'Yes', is, $3/(3+4+2) = 3/9$.

Figure 6.4 shows the frequency table thus constructed.

Step 2:

To predict whether the team will win for given weather conditions (a_1)

= Rainy, Wins in last three matches (a_2) = 2 wins, Humidity (a_3) = Normal and Win toss (a_4) = True, we need to choose 'Yes' from the above table for the given conditions.

From Bayes' theorem, we get

$$P\left(\text{Win match}|a_1 \cap a_2 \cap a_3 \cap a_4\right) = \frac{P(a_1 \cap a_2 \cap a_3 \cap a_4|\text{Win match})P(\text{Win match})}{P(a_1 \cap a_2 \cap a_3 \cap a_4)}$$

This equation becomes much easier to resolve if we recall that Naïve Bayes classifier assumes independence among events. This is specifically true for class-conditional independence,

which means that the events are independent so long as they are conditioned on the same class value. Also, we know that if the events are independent, then the probability rule says, $P(A \cap B) = P(A) P(B)$, which helps in simplifying the above equation significantly as

$P(\text{Win match}|a_1 \cap a_2 \cap a_3 \cap a_4)$

$$= \frac{P(a_1|\text{Win match})P(a_2|\text{Win match})P(a_3|\text{Win match})P(a_4|\text{Win match})P(\text{Win match})}{P(a_1)P(a_2)P(a_3)P(a_4)}$$

$= 2/9*4/9*6/9*9/14$

$= 0.014109347$

This should be compared with

$P(!\text{Win match}|a_1 \cap a_2 \cap a_3 \cap a_4)$

$$= \frac{P(a_1|!\text{Win match})P(a_2|!\text{Win match})P(a_3|!\text{Win match})P(a_4|!\text{Win match})P(!\text{Win match})}{P(a_1)P(a_2)P(a_3)P(a_4)}$$

Weather condition	Won Match			Humidity	Won Match	
	Yes	No			Yes	No
Sunny	3	2		High	3	4
OverCast	4	0		Normal	6	1
Rainy	2	3				
Total	9	5		Total	9	5

Wins in last 3 matches	Won Match			Win toss	Won Match	
	Yes	No			Yes	No
3 wins	2	2		FALSE	6	2
1 win	4	2		TRUE	3	3
2 wins	3	1				
Total	9	5		Total	9	5

FIG. 6.4 Construct frequency table

Step 3: by normalizing the above two probabilities, we can ensure that the sum of these two probabilities is 1.

$$P(\text{Win match}) = \frac{P(\text{Win match})}{P(\text{Win match}) + P(!\text{Win match})}$$
$$= \frac{0.014109347}{0.014109347 + 0.010285714}$$
$$= 0.578368999$$

$$P(!\text{Win match}) = \frac{P(!\text{Win match})}{P(\text{Win match}) + P(!\text{Win match})}$$
$$= \frac{0.010285714}{0.014109347 + 0.010285714}$$
$$= 0.421631001$$

Conclusion: This shows that there is 58% probability that the team will win if the above conditions become true for that particular day. Thus, Naïve Bayes classifier provides a simple yet powerful way to consider the influence of multiple attributes on the target outcome and refine the uncertainty of the event on the basis of the prior knowledge because it is able to simplify the calculation through independence assumption.

Applications of Naïve Bayes classifier

Text classification: Naïve Bayes classifier is among the most successful known algorithms for learning to classify text documents. It classifies the document where the probability of classifying the text is more. It uses the above algorithm to check the permutation and combination of the probability of classifying a document under a particular 'Title'. It has various applications in document categorization, language detection, and sentiment detection, which are very useful for traditional retailers, e-retailors, and other businesses on judging the sentiments of their clients on the basis of keywords in feedback forms, social media comments, etc.

Spam filtering: Spam filtering is the best known use of Naïve Bayesian text classification. Presently, almost all the email providers have this as a built-in functionality, which makes use of a Naïve Bayes classifier to identify spam email on the basis of certain conditions and also the probability of classifying an email as 'Spam'. Naïve Bayesian spam sifting has turned into a mainstream mechanism to recognize illegitimate a spam email from an honest-to-goodness email (sometimes called 'ham'). Users can also install separate email filtering programmes. Server-side email filters such as DSPAM, Spam Assassin, Spam Bayes, and ASSP make use of Bayesian spam filtering techniques, and the functionality is sometimes embedded within the mail server software itself.

Hybrid Recommender System: It uses Naïve Bayes classifier and collaborative filtering. Recommender systems (used by e-retailers like eBay, Alibaba, Target, Flipkart, etc.) apply machine learning and data mining techniques for filtering unseen information and can predict whether a user would like a given resource. For example, when we log in to these retailer websites, on the basis of the usage of texts used by the login and the historical data of purchase, it automatically recommends the product for the particular login persona. One of the algorithms is combining a Naïve Bayes classification approach with collaborative filtering,

and experimental results show that this algorithm provides better performance regarding accuracy and coverage than otheralgorithms.

Online Sentiment Analysis: The online applicationsuse supervised machine learning (Naïve Bayes) and useful computing. In the case of sentiment analysis, let us assume there are three sentiments such as nice, nasty,or neutral, and Naïve Bayes classifier is used to distinguish between them. Simple emotion modelling combines a statistically based classifier with a dynamicalmodel. The Naïve Bayes classifier employs 'single words' and 'word pairs' like features and determines the sentiments of the users. It allocates user utterances into nice, nasty, and neutral classes, labelled as +1, −1, and 0,respectively. This binary output drives a simple first- order dynamical system, whose emotional state represents the simulated emotional state of the experiment's personification.

Handling Continuous Numeric Features in Naïve BayesClassifier

In the above example, we saw that the Naïve Bayes classifier model uses a frequency table of the training data for its calculation. Thus, each attribute data shouldbe categorical in nature so that the combination of class and feature values can be created. But this is not possiblein the case of continuous numeric data as it does not have the categories of data.

The workaround that is applied in these cases is discretizing the continuous data on the basis of some data range. This is also called binning as the individual categories are termed as bins. For example, let us assumewe want to market a certain credit card to all the customers who are visiting a particular bank. We have to classify the persons who are visiting a bank as either interested candidate for taking a new card or non- interested candidate for a new card, and on the basis of this classification, the representative will approach the customer for sale. In this case, the customers visit the bank continuously during banking hours and have different values for the attributes we want to evaluate before classifying them into the interested/non-interested categories.

If we plot the number of customers visiting the bank during the 8 hours of banking time, the distribution graph will be a continuous graph. But if we introduce a logic to categorize the customers according to their time of entering the bank, then we will be able to put the customers in 'bins' or buckets for our analysis. We can then try to assess what time range is best suited for targeting the customers who will have interest in the newcredit card. The bins created by categorizing the customers by their time of entry looks like Figure 6.5.

This creates eight natural bins for us (or we may change the number of bins by changing our categorizingcriteria), which can now be used for Bayes analysis.
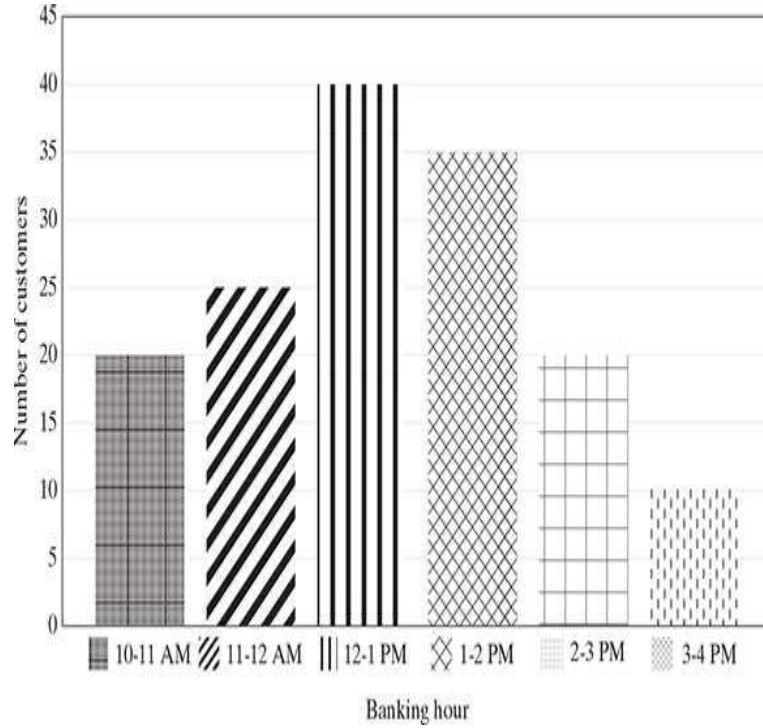
FIG. 6.5 The distribution of bins based on the time of entry of customers in the bank

BAYESIAN BELIEF NETWORK

We must have noted that a significant assumption in the Naïve Bayes classifier was that the attribute values $a_1, a_2,…, a_n$ are conditionally independent for a target value. The Naïve Bayes classifier generates optimal output when this condition is met. Though this assumption

significantly reduces the complexity of computation, in many practical scenarios, this requirement of conditional independence becomes a difficult constraint for the application of this algorithm. So, in this section, we will discuss the approach of Bayesian Belief network, which assumes that within the set of attributes, the probability distribution can have conditional probability relationships as well as conditional independence assumptions. This is different from the Naïve Bayes assumption of conditional independence of all the attributes as the belief network provides the flexibility of declaring a subset of the attributes as conditionally dependent while leaving rest of the attributes to hold the assumptions of conditional independence. The prior knowledge or belief about the influence of one attribute over the other is handled through joint probabilities as discussed later in this section.

Let us refresh our mind on the concept of conditional probability. If an uncertain event A is conditional on a knowledge or belief K, then the degree of belief in A with the assumption that K is known is expressed as $P(A|K)$. Traditionally, conditional probability is expressed by joint probability as follows:

$$P(A|K) = \frac{P(A, K)}{P(K)} \qquad (6.10)$$

Rearranging (6.9), we get the product rule

$P(A, K) = P(A|K)P(K)$

This can be extended for three variables or attributes as

$P(A, K, C) = P(A|K, C)P(K, C) = P(A|K, C)P(K|C)P(C)$

For a set of n attributes, the generalized form of theproduct rule becomes

$P(A_1, A_2, \ldots, A_n) = P(A_1|A_2, \ldots, A_n)P(A_2|A_3, \ldots, A_n)P(A_{n-1}|A_n)P(A_n)$ (6.10)

This generalized version of the product rule is calledthe Chain Rule.
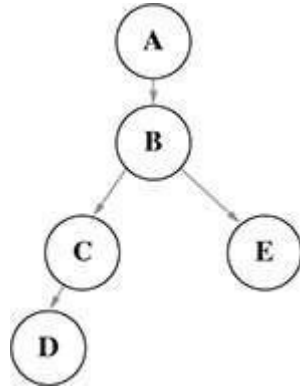
FIG. 6.6 Chain rule

Let us understand the chain rule by using the diagram in Figure 6.6. From the joint probability formula 6.10, wecan write

$P(A, B, C, D, E) = P(A|B, C, D, E)P(B|C, D, E)P(C|D, E)P(D|E)P(E)$

But from Figure 6.6, it is evident that E is not related to C and D, which means that the probabilities of variables C and D are not influenced by E and vice versa.Similarly, A is directly influenced only by B. By applying this knowledge of independence, we can simplify the above equation as

$P(A, B, C, D, E) = P(A|B)P(B|C, D)P(C|D)P(D)P(E)$

Let us discuss this concept of independence and conditional independence in detail in the next section.

Independence and conditional independence

We represent the conditional probability of A with knowledge of K as $P(A|K)$. The variables A and K are saidto be independent if $P(A|K) = P(A)$, which means that there is no influence of K on the uncertainty of A. Similarly, the joint probability can be written as $P(A,K) = P(A)P(K)$.

Extending this concept, the variables A and K are saidto be conditionally independent given C if $P(A|C) = P(A|K, C)$.

This concept of conditional independence can also be extended to a set of attributes. We can say that the set of variables $A_1, A_2,\ldots, A_n$ is conditionally independent of theset of variables $B_1, B_2,\ldots, B_m$ given the set of variables $C_1, C_2,\ldots, C_l$ if

$$P(A_1, A_2,\ldots, A_n|B_1, B_2,\ldots, B_m, C_1, C_2,\ldots, C_l) = P(A_1, A_2,\ldots, A_n|C_1, C_2,\ldots, C_l)$$

If we compare this definition with our assumption inthe Naïve Bayes classifier, we see that the Naïve Bayes classifier assumes that the instance attribute A_1 is conditionally independent of the instance attribute A_2, given the target value V, which can be written using thegeneral product rule and application of conditional independence formula as

$$P(A_1, A_2|V) = P(A_1|A_2, V)P(A_2, V) = P(A_1, V)P(A_2, V)$$

A Bayesian Belief network describes the joint probability distribution of a set of attributes in their jointspace. In Figure 6.7, a Bayesian Belief network is presented. The diagram consists of nodes and arcs. The nodes represent the discrete or continuous variables for which we are interested to calculate the conditional probabilities. The arc represents the causal relationship of the variables.

The two important information points we get from this network graph are used for the determining the joint probability of the variables. First, the arcs assert that the node variables are conditionally independent of its non- descendants in the network given its immediate predecessors in the network. If two variables A and B areconnected through a directed path, then B is called the descendent of A. Second, the conditional probability table for each variable provides the probability distribution of that variable given the values of its immediate predecessors. We can use Bayesian probability to calculate different behaviours of the variables in Figure 6.7.

Rain Today (Root Node) Probability	
True	0.1
False	0.9

Node Conditional probability	Rain Today	
	True	False
Paul Late to Class	True 0.6	0.5
	False 0.4	0.5

Node Conditional probability	Rain Today	
	True	False
Tim Late to Class	True 0.8	0.1
	False 0.2	0.9
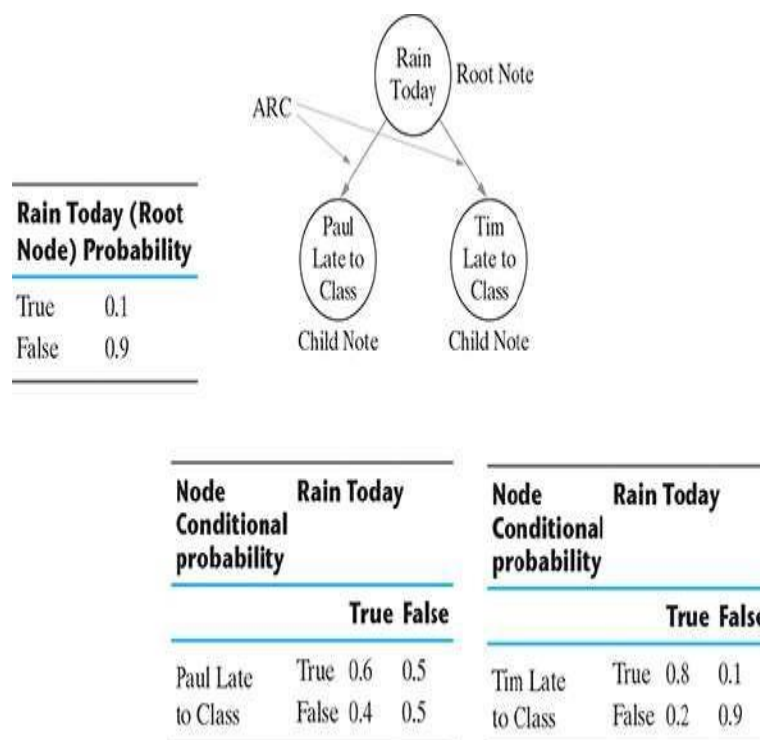
FIG. 6.7 Bayesian belief network

The unconditional probability that Tim is late to class –

P(Tim is late to class)

= *P*(Tim late|Rain Today*)P*(Rain Today) + *P*(Tim late|No Rain Today)**P*(No Rain Today)

= $(0.8 \times 0.1) \times (0.1 \times 0.9)$

= 0.17

From this unconditional probability, the most important use of the Bayesian Belief network is to find out the revised probability on the basis of the prior knowledge. If we assume that there was rain today, then the probability table can quickly provide us the information about the probability of Paul being late to class or the probability of Tim being late to class from the probability distribution table itself. But if we do not know whether there was rain today or not, but we only know that Tim is late to class today, then we can arrive at the following probabilities –

The revised probability that there was rain today –

$$P(\text{Rain Today|Tim late to class}) = \frac{P(\text{Tim late|Rain today}) \, P(\text{Rain today})}{P(\text{Tim late})}$$

$$= \frac{(0.8 \times 0.1)}{0.17}$$

$$= 0.47$$

$$= P(\text{Rain Today}) \text{ as Tim late to class is already known}$$

The revised probability that Paul will be late to class today –

P(Paul late to class today)

= *P*(Paul late|Rain today)*P*(Rain today) + *P*(Paul late|No rain today)*P*(No rain today)

= $(0.6 \times 0.47) + (0.5 \times (1\text{-}0.47))$

= 0.55

Here, we used the concept of hard evidence and soft evidence. Hard evidence (instantiation) of a node is evidence that the state of the variable is definitely as a particular value. In our above example, we had hard evidence that 'Tim is late to class'. If a particular node is instantiated, then it will block propagation of evidence further down to its child nodes. Soft evidence for a node is the evidence that provides the prior probability values for the node. The node 'Paul is late to class' is soft evidenced with the prior knowledge that 'Tim is late to class'.

The Bayesian Belief network can represent much more complex scenarios with dependence and independence concepts. There are three types of connections possible in a Bayesian Belief network.

Diverging Connection: In this type of connection, the evidence can be transmitted between two child nodes of the same parent provided that the parent is not instantiated. In Figure 6.7, we already saw the behaviour of diverging connection.

Serial Connection: In this type of connection, any evidence entered at the beginning of the connection can be transmitted through the directed path provided that no intermediate node on the path is instantiated (see Fig. 6.8 for illustration).
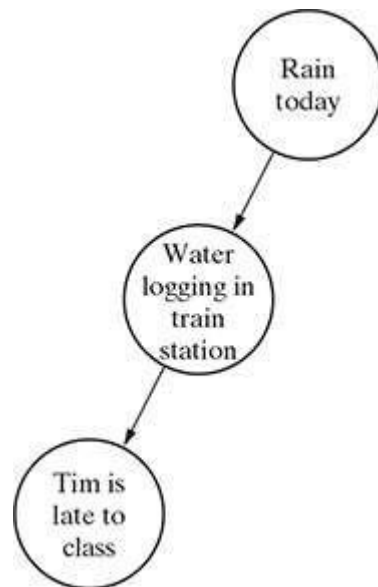
FIG. 6.8 Serial connection

Converging Connection: In this type of connection, the evidence can only be transmitted between two parents when the child (converging) node has received some evidence and that evidence can be soft or hard (see Fig. 6.9 for illustration).
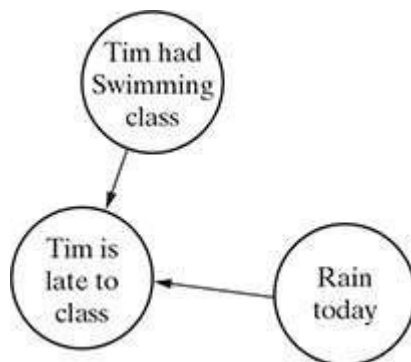
FIG. 6.9 Convergent connection

As discussed above, by using the Bayesian network, we would like to infer the value of a target variable on the basis of the observed values of some other variables.

Please note that it will not be possible to infer a single value in the case of random variables we are dealing with, but our intention is to infer the probability distribution of the target variable given the observed values of other variables. In general, the Bayesian network can be used to compute the probability distribution of any subset of node variables given the values or distribution of the remaining variables.

<u>Use of the Bayesian Belief network in machine learning</u>

We have seen that the Bayesian network creates a complete model for the variables and their relationships and thus can be used to answer probabilistic queries about them. A common use of the network is to find out the updated knowledge about the state of a subset of variables, while the state of the other subset (known as the evidence variables) is observed. This concept,

often known as probabilistic inference process of computing the posterior distribution of variables, given some evidences, provides a universal sufficient statistic for applications related to detections. Thus if one wants to choose the values for a subset of variables in order to minimize some expected loss functions or decision errors, then this method is quiet effective. In other words, the Bayesian network is a mechanism for automatically applying Bayes' theorem to complex problems. Bayesian networks are used for modelling beliefs in domains like computational biology and bioinformatics such as protein structure and gene regulatory networks, medicines, forensics, document classification, information retrieval, image processing, decision support systems, sports betting and gaming, property market analysis and various other fields.

SAMPLE QUESTIONS

MULTIPLE CHOICE QUESTIONS (1 MARK EACH)

1.Three companies X, Y, and Z supply 40%, 45%, and 15% of the uniforms to a school. Past experience shows that 2%, 3%, and 4% of the uniforms supplied by these companies are defective. If a uniform was found to be defective, what is the probability that the uniform was supplied by Company X?

1. $^{34}/_{69}$

2. $^{25}/_{55}$

3. 16/55

4. $^{16}/_{12}$

2.A box of apples contains 10 apples, of which 6 are defective. If 3 of the apples are removed from the box in succession without replacement, what is the probability that all the 3 apples are defective?

1. (6*5*4)/ (10*10*10)

2. (6*5*4)/ (10*9*8)

3. (3*3*3)/ (10*10*10)

4. (3*2*1)/ (10*9*8)

3.Two boxes containing chocolates are placed on a table. The boxes are labelled B_1 and B_2. Box B_1 contains 6 dark chocolates and 5 white chocolates. Box B_2 contains 3 dark chocolates and 8 orange chocolates. The boxes are arranged so that the probability of selecting box B_1 is 1 /3.and the probability of selecting box B_2 is 2/3. Sneha is blindfolded and asked to select a chocolate. She will win Rs. 10,000 if she selects a dark chocolate. What is the probability that Sneha will win Rs.10,000 (that is, she will select a dark chocolate)?

1. $^{7}/_{33}$

2. $^{6}/_{33}$

3. $^{11}/_{33}$

4. $^{10}/_{33}$

4. Two boxes containing chocolates are placed on a table. The boxes are labelled B1 and B2. Box B1 contains 6 Cadbury chocolates and 5 Amul chocolates. Box B_2 contains 3 Cadbury chocolates and 8 Nestle chocolates. The boxes are arranged so that the probability of selecting box B1 is 1/3 and the probability of selecting box B_2 is 2/3. Sneha is blindfolded and asked to select a chocolate. She will win Rs. 10,000 if she selects a Cadbury chocolate. If she win Rs 10,000, what is the probability that she selected a Cadbury chocolate from the first box?

1. $^{7}/_{12}$

2. $^{6}/_{12}$

3. $^{11}/_{12}$

4. $^{10}/_{33}$

5. In a certain basketball club, there are 4% of male players who are over 6 feet tall and 1% of female players who are over 6 feet tall. The ratio of male to female players in the total player population is male : female = 2:3. A player is selected at random from among all those who are over 6 feet tall. What is the probability that the player is a female?

1. 3/11

2. 2/5

3. 2/11

4. 1/11

6. The probability that a particular hypothesis holds for a data set based on the Prior is called

1. Independent probabilities

2. Posterior probabilities

3. Interior probabilities

4. Dependent probabilities

7. One main disadvantage of Bayesian classifiers is that they utilize all available parameters to subtly change the predictions.

1. True

2. False

8. In a bolt factory, machines A1, A2, and A3 manufacture respectively 25%, 35%, and 40% of the total output. Of these 5%, 4%, and 2% are defective bolts. A bolt is drawn at random from the product and is found to be defective. What is the probability that it was manufactured by machine A2?

1. 0.0952

2. 0.452

3. 0.952

4. 0.125

9.Bayesian methods can perform better than the other methods while validating the hypotheses that make probabilistic predictions.

1.True

2.False

10.Naïve Bayes classifier makes the naïve assumption that the attribute values are conditionally dependent given the classification of the instance.

1.True

2.False

SHORT ANSWER-TYPE QUESTIONS (5 MARKS EACH)

1. What is prior probability? Give an example.
2. What is posterior probability? Give an example.
3. What is likelihood probability? Give an example.
4. What is Naïve Bayes classifier? Why is it named so?
5. What is optimal Bayes classifier?
6. Write any two features of Bayesian learning methods.
7. Define the concept of consistent learners.
8. Write any two strengths of Bayes classifier.
9. Write any two weaknesses of Bayes classifier.
10. Explain how Naïve Bayes classifier is used for
   a. Text classification

   b. Spam filtering

   c. Market sentiment analysis

LONG ANSWER-TYPE QUESTIONS (10 MARKS EACH)

1. Explain the concept of Prior, Posterior, and Likelihood with an example.
2. How Bayes' theorem supports the concept learning principle?
3. Explain Naïve Bayes classifier with an example of its use in practical life.
4. Is it possible to use Naïve Bayes classifier for continuous numeric data? If so, how?
5. What are Bayesian Belief networks? Where are they used? Can they solve all types of problems?
6. In an airport security checking system, the passengers are checked to find out any intruder. Let I with $i \in \{0, 1\}$ be the random variable which indicates whether somebody is an intruder ($i = 1$) or not ($i = 0$) and A with $a \in \{0, 1\}$ be the variable indicating alarm. An alarm will be raised if an intruder is identified with probability $P(A = 1|I = 1) = 0.98$ and a non- intruder with probability $P(A = 1|I = 0) = 0.001$, which implies the error factor. In the population of passengers, the probability of someone is intruder is $P(I = 1) = 0.00001$. What is the probability that an alarm is raised when a person actually is an intruder?
7. An antibiotic resistance test (random variable T) has 1% false positives (i.e. 1% of those not resistance to an antibiotic show positive result in the test) and 5% false negatives (i.e. 5% of those actually resistant to an antibiotic test negative). Let us assume that 2% of those tested are resistant to antibiotics. Determine the probability that somebody who tests positive is actually resistant (random variable D).
8. For preparation of the exam, a student knows that one question is to be solved in the exam

which is either of types A, B, or C. The probabilities of A, B, or C appearing in the exam are 30%, 20%, and 50% respectively. During the preparation, the student solved 9 of 10 problems of type A, 2 of 10 problems of type B, and 6 of 10 problems of type C.

  a. What is the probability that the student will solve the problem of the exam?
  b. Given that the student solved the problem, what is the probability that it was of type A?

9. A CCTV is installed in a bank to monitor the incoming customers and take a photograph. Though there are continuous flows of customers, we create bins of timeframe of 5 min each. In each time frame of 5 min, there may be a customer moving into the bank with 5% probability or there is no customer (again, for simplicity, we assume that either there is 1 customer or none, not the case of multiple customers). If there is a customer, it will be detected by the CCTV with a probability of 99%. If there is no customer, the camera will take a false photograph by detecting other thing's movement with a probability of 10%.

  a. How many customers enter the bank on average per day (10 hours)?

  b. How many false photographs (there is a photograph taken even though there is no customer) and how many missed photographs (there is no photograph even though there is a customer) are there on average per day?

  c. If there is a photograph, what is the probability that there is indeed a customer?

10. Draw the Bayesian Belief network to represent the conditional independence assumptions of the Naïve Bayes classifier for the match winning prediction problem of Section 6.4.4. Construct the conditional probability table associated with the node Won Toss.

CHAPTER-2

SUPERVISED LEARNING: CLASSIFICATION

INTRODUCTION

OBJECTIVE OF THE CHAPTER:

In the last chapter on Bayesian Concept Learning, you were introduced to an important supervised learning algorithm – the Naïve Bayes algorithm. Aswe have seen, it is a very simple but powerful classifier based on Bayes' theorem of conditional probability. However, other than the Naïve Bayes classifier, there are more algorithms for classification. This chapter will focus on other classification algorithms.

The first algorithm we will delve into in this chapter is k-Nearest Neighbour (kNN), which tries to classify unlabelled data instances based on the similarity with the labelled instances in the trainingdata.

Then, another critical classifier, named as decision tree, will be explained in detail. Decision tree, as the name suggests, is a classifier based on aseries of logical decisions, which resembles a tree with branches.

Next, we will explore the random forest classifier,which, in very simplistic terms, can be thought as a collection of many decision trees.

Finally, a very powerful and popular classifiernamed Support Vector Machine (SVM) will be explored.

So, by the end of this chapter, you will gain enough knowledge to start solving a classification problem by using some standard classifiers.

EXAMPLE OF SUPERVISED LEARNING

In supervised learning, the labelled training data providethe basis for learning. According to the definition of machine learning, this labelled training data is the experience or prior knowledge or belief. It is called supervised learning because the process of learning from the training data by a machine can be related to a teachersupervising the learning process of a student who is new to the subject. Here, the teacher is the training data.

Training data is the past information with known valueof class field or '**label**'. Hence, we say that the '**training data is labelled**' in the case of supervised learning (refer Fig. 7.1). Contrary to this, there is no labelled training data for unsupervised learning. Semi-supervised learning, as depicted in Figure 7.1, uses a small amount of labelled data along with unlabelled data for training.
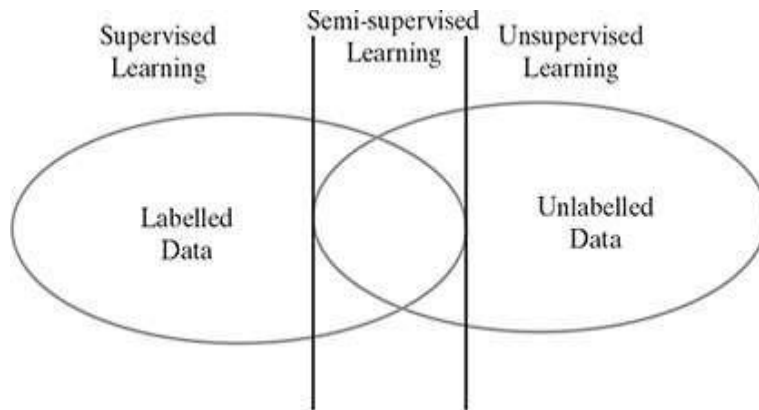
FIG. 7.1 Supervised learning vs. unsupervised learning

In a hospital, many patients are treated in the general wards. In comparison, the number of beds in the Intensive Care Unit (ICU) is much less. So, it is always a cause of worry for the hospital management that if the health condition of a number of patients in the general ward suddenly aggravates and they would have to be moved to ICU. Without previous planning and preparations, such a spike in demand becomes difficult for the hospital to manage. This problem can be addressed in a much better way if it is possible to predict which of the patients in the normal wards have a possibility of their health condition deteriorating and thus need to be moved to ICU.

This kind of prediction problem comes under the purview of supervised learning or, more specifically, under classification. The hospital already has all past patient records. The records of the patients whose health condition aggravated in the past and had to be moved to ICU can form the training data for this prediction problem. Test results of newly admitted patients are used to classify them as high-risk or low-risk patients.

Some more examples of supervised learning are as follows:

Prediction of results of a game based on the past analysis of results Predicting whether a tumour is malignant or benign on the basis of the analysis of data

Price prediction in domains such as real estate, stocks, etc.

Did you know?

'IBM Watson Health' developed by IBM software provides evidence-backed cancer care to each patient by understanding millions of data points. 'Watson for Oncology' helps physicians quickly identify vital information in a patient's medical record, surface relevant evidence, and explore various treatment options for patients

(Source: https://www.ibm.com/watson/health/oncology-and-genomics/oncology).

CLASSIFICATION MODEL

Let us consider two examples, say 'predicting whether a tumour is malignant or benign' and 'price prediction in the domain of real estate'. Are these two problems same in nature?

The answer is 'no'. It is true that both of them are problems related to prediction. However, for tumour prediction, we are trying to predict which category or class, i.e. 'malignant' or 'benign', an unknown input datarelated to tumour belongs to. In the other case, that is, for price prediction, we are trying to predict an absolute value and not a class.

When we are trying to predict a categorical or nominalvariable, the problem is known as a classification problem. A classification problem is one where the output variable is a category such as 'red' or 'blue' or 'malignant tumour' or 'benign tumour', etc.

Whereas when we are trying to predict a numerical variable such as 'price', 'weight', etc. the problem falls under the category of regression.

Note that:
Supervised machine learning is as good as the data used to train it. If the training data is poor in quality,the prediction will also be far from being precise.

We can observe that in classification, the whole problem centres around assigning a label or category or class to a test data on the basis of the label or category orclass information that is imparted by the training data. Because the target objective is to assign a class label, we call this type of problem as a classification problem.

Figure 7.2 depicts the typical process of classification where a classification model is obtained from the labelled training data by a classifier algorithm. On the basis of the model, a class label (e.g. 'Intel' as in the case of the test data referred in Fig. 7.2) is assigned to the testdata.

A critical classification problem in the context of the banking domain is identifying potentially fraudulent transactions. Because there are millions of transactionswhich have to be scrutinized to identify whether a particular transaction might be a fraud transaction, it is not possible for any human being to carry out this task. Machine learning is leveraged efficiently to do this task, and this is a classic case of classification. On the basis of the past transaction data, especially the ones labelled as fraudulent, all new incoming transactions are marked orlabelled as usual or suspicious. The suspicious transactions are subsequently segregated for a closer review.
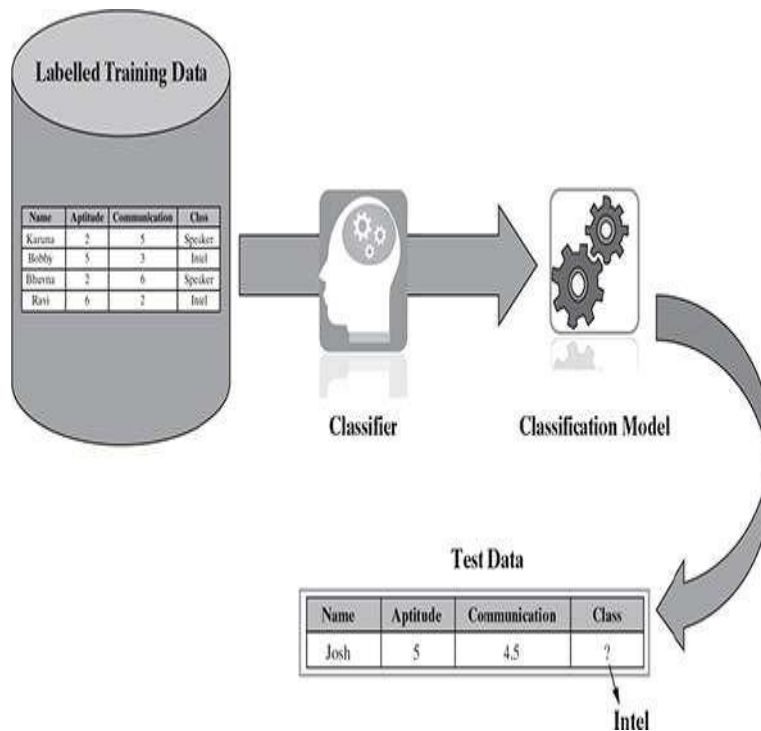
FIG. 7.2 Classification model

In summary, classification is a type of supervised learning where a target feature, which is of categorical type, is predicted for test data on the basis of the information imparted by the training data. The target categorical feature is known as **class** .

Some typical classification problems include the following:

- Image classification
- Disease prediction
- Win–loss prediction of games
- Prediction of natural calamity such as earthquake, flood, etc.
- Handwriting recognition

Did you know?
Machine learning saves lives – it can spot 52% of breast cancer cells at least a year before patients are diagnosed
US Postal Service uses machine learning for handwriting recognition
Facebook's news feed uses machine learning to personalize each member's feed

CLASSIFICATION LEARNING STEPS

First, there is a problem which is to be solved, and then, the required data (related to the problem, which is already stored in the system) is evaluated and pre- processed based on the algorithm. Algorithm selection is a critical point in supervised learning. The result after iterative training rounds is a classifier for the problem in hand (refer Fig. 7.3).
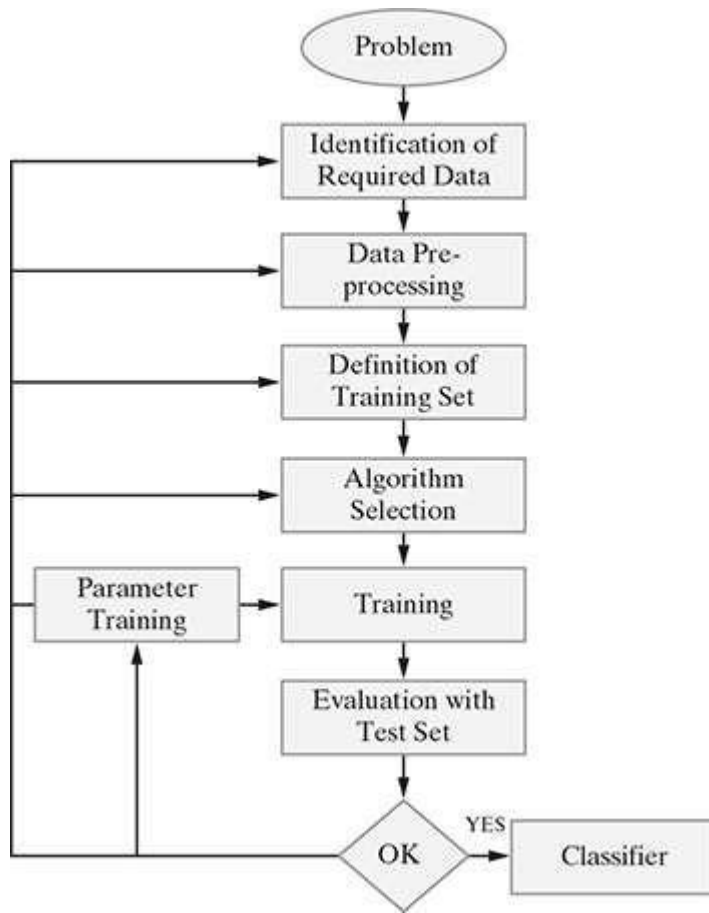
FIG. 7.3 Classification model steps

Problem Identification: Identifying the problem is the first step in the supervised learning model. The problem needs to be a well-formed problem,i.e. a problem with well-defined goals and benefit, which has along-term impact.

Identification of Required Data: On the basis of theproblem identified above, the required data set that precisely represents the identified problem needs to be identified/evaluated. For example: If the problem is to predict whether a tumour is malignant or benign, then the corresponding patient data sets related to malignanttumour and benign tumours are to be identified.

Data Pre-processing: This is related to the cleaning/transforming the data set. This step ensures that all the unnecessary/irrelevant data elements are removed. Data pre-processing refers to the transformations applied to the identified data before feeding the same into the algorithm. Because the data isgathered from different sources, it is usually collected ina raw format and is not ready for immediate analysis.

This step ensures that the data is ready to be fed into themachine learning algorithm.

Definition of Training Data Set: Before starting the analysis, the user should decide what kind of data set is to be used as a training set. In the case of signature analysis, for example, the training data set might be a single handwritten alphabet, an entire handwritten word(i.e. a group of the alphabets) or an entire line of handwriting (i.e. sentences or a group of words).

Thus, a set of 'input meta-objects' and corresponding 'output meta-objects' are also gathered. The training set needs tobe actively representative of the real-world use of the given scenario. Thus, a set of data input (*X*) and corresponding outputs (*Y*) is gathered either from human experts or experiments.

Algorithm Selection: This involves determining the structure of the learning function and the correspondinglearning algorithm. This is the most critical step of supervised learning model. On the basis of various parameters, the best algorithm for a given problem is chosen.

Training: The learning algorithm identified in the previous step is run on the gathered training set for further fine tuning. Some supervised learning algorithmsrequire the user to determine specific control parameters(which are given as inputs to the algorithm). These parameters (inputs given to algorithm) may also be adjusted by optimizing performance on a subset (called as validation set) of the training set.

Evaluation with the Test Data Set: Training data isrun on the algorithm, and its performance is measured here. If a suitable result is not obtained, further trainingof parameters may be required.

COMMON CLASSIFICATION ALGORITHMS

Let us now delve into some common classification algorithms. Following are the most common classification algorithms, out of which we have already learnt about the Naïve Bayes classifier in Chapter 6. We will cover details of the other algorithms in this chapter.

1. *k*-Nearest Neighbour (*k*NN)
2. Decision tree
3. Random forest
4. Support Vector Machine (SVM)
5. Naïve Bayes classifier

k -Nearest Neighbour (*k*NN)

The *k*NN algorithm is a simple but extremely powerful classification algorithm. The name of the algorithm originates from the underlying philosophy of *k*NN – i.e. people having similar background or mindset tend to stay close to each other. In other words, neighbours in a locality have a similar background. In the same way, as apart of the *k*NN algorithm, the unknown and unlabelled data which comes for a prediction problem is judged on the basis of the training data set elements which are similar to the unknown element. So, the class label of theunknown element is assigned on the basis of the class labels of the similar training data set elements (metaphorically can be considered as neighbours of the unknown element). Let us try to understand the algorithm with a simple data set.

How kNN works

Let us consider a very simple Student data set as depicted in Figure 7.4. It consists of 15 students studyingin a class. Each of the students has been assigned a scoreon a scale of 10 on two performance parameters – 'Aptitude' and 'Communication'. Also, a class value is assigned to each student based on the following criteria:

1. Students having good communication skills as well as a goodlevel of aptitude have been classified as 'Leader'

2. Students having good communication skills but not so good level of aptitude have been classified as 'Speaker'
3. Students having not so good communication skill but a good level of aptitude have been classified as 'Intel'

Name	Aptitude	Communication	Class
Karuna	2	5	Speaker
Bhuvna	2	6	Speaker
Gaurav	7	6	Leader
Parul	7	2.5	Intel
Dinesh	8	6	Leader
Jani	4	7	Speaker
Bobby	5	3	Intel
Parimal	3	5.5	Speaker
Govind	8	3	Intel
Susant	6	5.5	Leader
Gouri	6	4	Intel
Bharat	6	7	Leader
Ravi	6	2	Intel
Pradeep	9	7	Leader
Josh	5	4.5	Intel

FIG. 7.4 Student data set

As we have already seen in <u>Chapter 3</u>, while building a classification model, a part of the labelled input data is retained as test data. The remaining portion of the input data is used to train the model – hence known as training data. The motivation to retain a part of the data as test data is to evaluate the performance of the model. As we have seen, the performance of the classification model is measured by the number of correct classifications made by the model when applied to an unknown data set. However, it is not possible during model testing to know the actual label value of an unknown data. Therefore, the test data, which is a part of the labelled input data, is used for this purpose. If the class value predicted for most of the test data elements matches with the actual class value that they have, then we say that the classification model possesses a good accuracy. In context of the Student data set, to keep the things simple, we assume one data element of the input data set as the test data. As depicted in <u>Figure 7.5</u>, the record of the student named Josh is assumed to be the test data. Now that we have the training data and test data identified, we can start with the modelling.

As we have already discussed, in the *k*NN algorithm, the class label of the test data elements is decided by the class label of the training data elements which are neighbouring, i.e. similar in nature. But there are two challenges:

What is the basis of this similarity or when can we say that two data elements are similar?

How many similar elements should be considered for deciding the class label of each test data element?

To answer the first question, though there are many measures of similarity, the most common approach adopted by *k*NN to measure similarity between two data elements is Euclidean
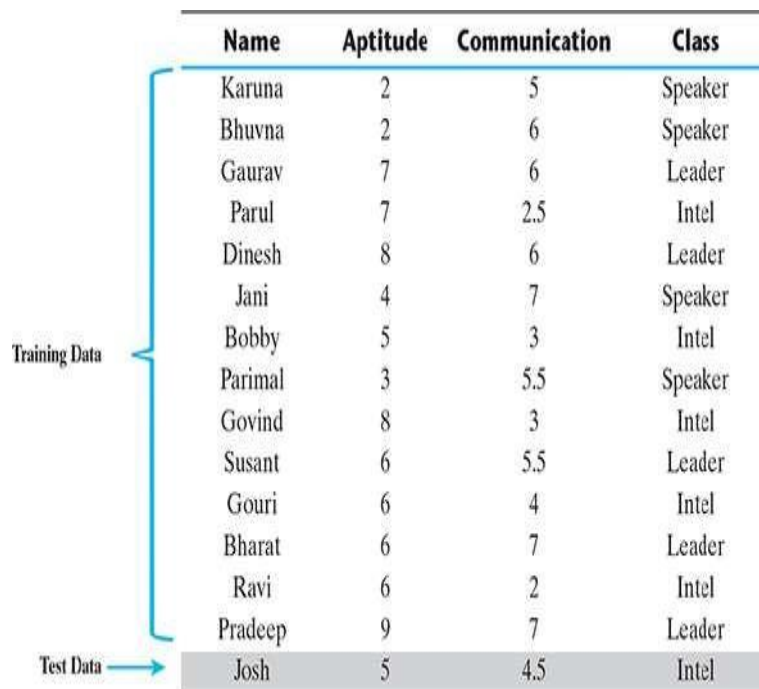
distance. Considering a very simple data set having two features (say f_1 and f_2), Euclidean distance between two data elements d_1 and d_2 can be measured by

$$\text{Euclidean distance} = \sqrt{(f_{11} - f_{12})^2 + (f_{21} - f_{22})^2}$$

where f_{11} = value of feature f_1 for data element d_1 f_{12} = value of feature f_1 for data element d_2

f_{21} = value of feature f_2 for data element d_1

f_{22} = value of feature f_2 for data element d_2

	Name	Aptitude	Communication	Class
	Karuna	2	5	Speaker
	Bhuvna	2	6	Speaker
	Gaurav	7	6	Leader
	Parul	7	2.5	Intel
	Dinesh	8	6	Leader
	Jani	4	7	Speaker
Training Data	Bobby	5	3	Intel
	Parimal	3	5.5	Speaker
	Govind	8	3	Intel
	Susant	6	5.5	Leader
	Gouri	6	4	Intel
	Bharat	6	7	Leader
	Ravi	6	2	Intel
	Pradeep	9	7	Leader
Test Data →	Josh	5	4.5	Intel

FIG. 7.5 Segregated student data set

So, as depicted in Figure 7.6, the training data points of the Student data set considering only the features 'Aptitude' and 'Communication' can be represented as dots in a two-dimensional feature space. As shown in thefigure, the training data points having the same class value are coming close to each other. The reason for considering two-dimensional data space is that we are considering just the two features of the Student data set,

i.e. 'Aptitude' and 'Communication', for doing the classification. The feature 'Name' is ignored because, as we can understand, it has no role to play in deciding the class value. The test data point for student Josh is represented as an asterisk in the same space. To find outthe closest or nearest neighbours of the test data point, Euclidean distance of the different dots need to be calculated from the asterisk. Then, the class value of the closest neighbours helps in assigning the class value of the test data element.
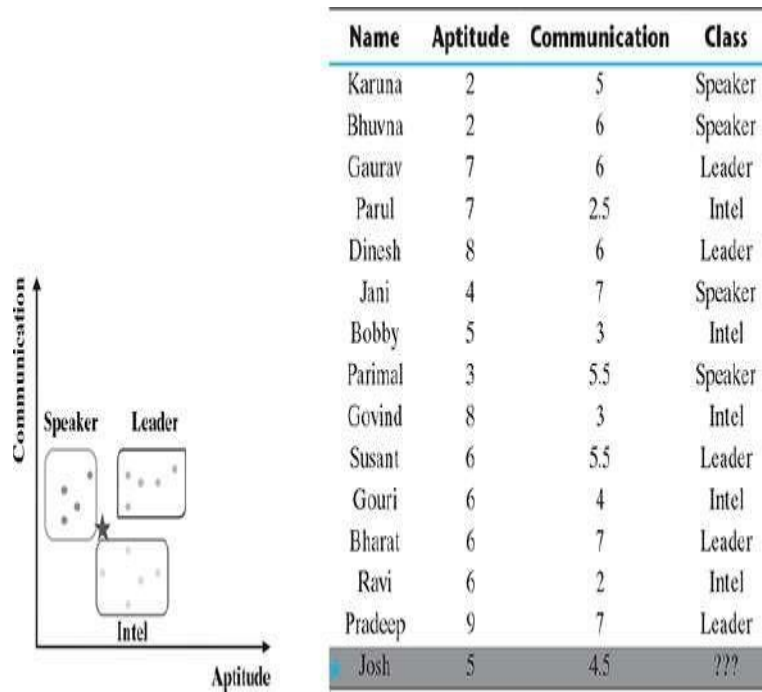
Name	Aptitude	Communication	Class
Karuna	2	5	Speaker
Bhuvna	2	6	Speaker
Gaurav	7	6	Leader
Parul	7	2.5	Intel
Dinesh	8	6	Leader
Jani	4	7	Speaker
Bobby	5	3	Intel
Parimal	3	5.5	Speaker
Govind	8	3	Intel
Susant	6	5.5	Leader
Gouri	6	4	Intel
Bharat	6	7	Leader
Ravi	6	2	Intel
Pradeep	9	7	Leader
Josh	5	4.5	???

FIG. 7.6 2-D representation of the student data set

Now, let us try to find the answer to the second question, i.e. how many similar elements should be considered. The answer lies in the value of 'k' which is a user-defined parameter given as an input to the algorithm. In the kNN algorithm, the value of 'k' indicates the number of neighbours that need to be considered. For example, if the value of k is 3, only three nearest neighbours or three training data elements closest to the test data element are considered. Out of thethree data elements, the class which is predominant is considered as the class label to be assigned to the test data. In case the value of k is 1, only the closest training data element is considered. The class label of that data element is directly assigned to the test data element. Thisis depicted in Figure 7.7

Name	Aptitude	Communication	Class	Distance	$k = 1$	$k = 2$	$k = 3$
Karuna	2	5	Speaker	3.041			
Bhuvna	2	6	Speaker	3.354			
Parimal	3	5.5	Speaker	2.236			
Jani	4	7	Speaker	2.693			
Bobby	5	3	Intel	1.500			1.500
Ravi	6	2	Intel	2.693			
Gouri	6	4	Intel	1.118	1.118	1.118	1.118
Parul	7	2.5	Intel	2.828			
Govind	8	3	Intel	3.354			
Susant	6	5.5	Leader	1.414			
Bharat	6	7	Leader	2.693			
Gaurav	7	6	Leader	2.500			
Dinesh	8	6	Leader	3.354			
Pradeep	9	7	Leader	4.717			
Josh	5	4.5	???				

FIG. 7.7 Distance calculation between test and training points

Let us now try to find out the outcome of the algorithm for the Student data set we have. In other words, we want to see what class value *k*NN will assign for the test data for student Josh. Again, let us refer back to Figure 7.7. As is evident, when the value of *k* is taken as 1, only one training data point needs to be considered. The training record for student Gouri comes as the closest one to test record of Josh, with a distance value of 1.118. Gouri has class value 'Intel'. So, the test data point is also assigned a class label value 'Intel'. When the value of *k* is assumed as 3, the closest neighbours of Josh in the training data set are Gouri, Susant, and Bobby with distances being 1.118, 1.414, and 1.5, respectively. Gouri and Bobby have class value 'Intel', while Susant has class value 'Leader'.

In this case, the class value of Josh is decided by majority voting. Because the class value of 'Intel' is formed by the majority of the neighbours, the class value of Josh is assigned as 'Intel'. This same process can be extended for any value of *k*.

But it is often a tricky decision to decide the value of *k*.

The reasons are as follows:

- If the value of *k* is very large (in the extreme case equal to the total number of records in the training data), the class label of the majority class of the training data set will be assigned to the test data regardless of the class labels of the neighbours nearest to the test data
- If the value of *k* is very small (in the extreme case equal to 1), the class value of a noisy data or outlier in the training data set which is the nearest neighbour to the test data will be assigned to the test data

The best *k* value is somewhere between these two extremes.

Few strategies, highlighted below, are adopted by machine learning practitioners to arrive at a value for *k*.

- One common practice is to set *k* equal to the square root of the number of training records
- An alternative approach is to test several *k* values on a variety of test data sets and choose the one that delivers the best performance
- Another interesting approach is to choose a larger value of *k*, but apply a weighted voting process in which the vote of close neighbours is considered more influential than the vote of distant neighbours

kNN algorithm

Input: Training data set, test data set (or data points), value of 'k' (i.e. number of nearest neighbours to be considered)

Steps:

Do for all test data points

Calculate the distance (usually Euclidean distance) of the test data point from the different training data points.

Find the closest 'k' training data points, i.e. training data points whose distances are least from the test data point.

If k = 1

Then assign class label of the training data point to the test data point

Else

Whichever class label is predominantly present in the training data points, assign that class label to the test data point

End do

Why the kNN algorithm is called a lazy learner?

We have already discussed in Chapter 3 that eager learners follow the general steps of machine learning, i.e. perform an abstraction of the information obtained from the input data and then follow it through by a generalization step. However, as we have seen in the case of the kNN algorithm, these steps are completely skipped. It stores the training data and directly applies the philosophy of nearest neighbourhood finding to arrive at the classification. So, for kNN, there is no learning happening in the real sense. Therefore, kNN falls under the category of lazy learner.

Strengths of the kNN algorithm

- Extremely simple algorithm – easy to understand
- Very effective in certain situations, e.g. for recommender system design
- Very fast or almost no time required for the training phase

Weaknesses of the kNN algorithm

- Does not learn anything in the real sense. Classification is done completely on the basis of the training data. So, it has a heavy reliance on the training data. If the training data does not represent the problem domain comprehensively, the algorithm fails to make an effective classification.
- Because there is no model trained in real sense and the classification is done completely on the basis of the training data, the classification process is very slow.
- Also, a large amount of computational space is required to load the training data for classification.

Application of the kNN algorithm

One of the most popular areas in machine learning where the kNN algorithm is widely adopted is recommender systems. As we know, recommender systems recommend users different items which are similar to a particular item that the user seems to like. The liking pattern may be revealed from past purchases or browsing history and the similar items are identified using the kNN algorithm.

Another area where there is widespread adoption of kNN is searching documents/ contents similar to a given document/content. This is a core area under information retrieval and is known as concept search.

Decision tree

Decision tree learning is one of the most widely adopted algorithms for classification. As the name indicates, it builds a model in the form of a tree structure. Its grouping exactness is focused with different strategies, and it is exceptionally productive.

A decision tree is used for multi-dimensional analysis with multiple classes. It is characterized by fast executiontime and ease in the interpretation of the rules. The goal of decision tree learning is to create a model (based on the past data called past vector) that predicts the value ofthe output variable based on the input variables in the feature vector.

Each node (or decision node) of a decision tree corresponds to one of the feature vector. From every node, there are edges to children, wherein there is an edge for each of the possible values (or range of values) of the feature associated with the node. The tree terminates at different leaf nodes (or terminal nodes) where each leaf node represents a possible value for theoutput variable. The output variable is determined by following a path that starts at the root and is guided by the values of the input variables.

A decision tree is usually represented in the formatdepicted in Figure 7.8.
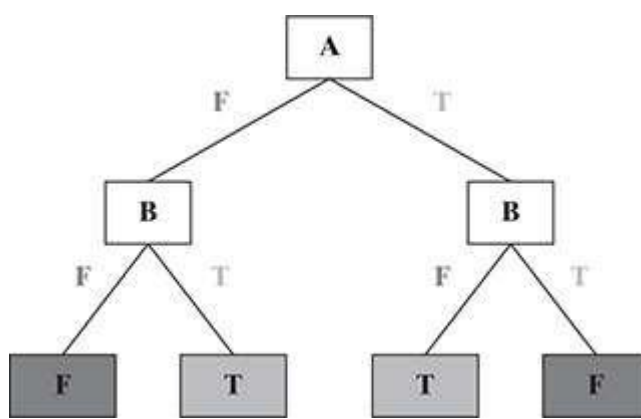
FIG. 7.8 Decision tree structure

Each internal node (represented by boxes) tests an attribute (represented as 'A'/'B' within the boxes). Each branch corresponds to an attribute value (T/F) in the above case. Each leaf node assigns a classification. The first node is called as 'Root' Node. Branches from the root node are called as 'Leaf' Nodes where 'A' is the RootNode (first node). 'B' is the Branch Node. 'T' & 'F' are Leaf Nodes.

Thus, a decision tree consists of three types of nodes:

- Root Node
- Branch Node
- Leaf Node

Figure 7.9 shows an example decision tree for a car driving – the decision to be taken is whether to 'Keep Going' or to 'Stop', which depends on various situations as depicted in the figure. If the signal is RED in colour, then the car should be stopped. If there is not enough gas(petrol) in the car, the car should be stopped at the next available gas station.
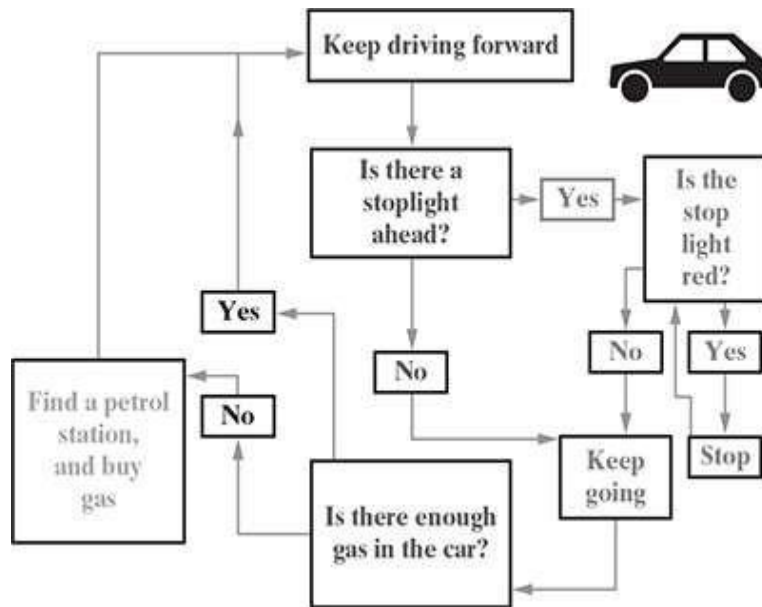
FIG. 7.9 Decision tree example

<u>Building a decision tree</u>

Decision trees are built corresponding to the training data following an approach called recursive partitioning. The approach splits the data into multiple subsets on the basis of the feature values. It starts from the root node, which is nothing but the entire data set. It first selects the feature which predicts the target class in the strongest way. The decision tree splits the data set into multiple partitions, with data in each partition having a distinct value for the feature based on which the partitioning has happened. This is the first set of branches. Likewise, the algorithm continues splitting the nodes on the basis of the feature which helps in the best partition. This continues till a stopping criterion is reached. The usual stopping criteria are –

1. All or most of the examples at a particular node have the same class
2. All features have been used up in the partitioning
3. The tree has grown to a pre-defined threshold limit

Let us try to understand this in the context of an example. Global Technology Solutions (GTS), a leading provider of IT solutions, is coming to College of Engineering and Management (CEM) for hiring B.Tech. students. Last year during campus recruitment, they had shortlisted 18 students for the final interview. Being a company of international repute, they follow a stringent interview process to select only the best of the students. The information related to the interview evaluation results of shortlisted students (hiding the names) on the basis of different evaluation parameters is available for reference in Figure 7.10. Chandra, a student of CEM, wants to find out if he may be offered a job in GTS. His CGPA is quite high. His self-evaluation on the other parameters is as follows:

Communication – Bad; Aptitude – High; Programming skills – Bad

CGPA	Communication	Aptitude	Programming Skill	Job offered?
High	Good	High	Good	Yes
Medium	Good	High	Good	Yes
Low	Bad	Low	Good	No
Low	Good	Low	Bad	No
High	Good	High	Bad	Yes
High	Good	High	Good	Yes
Medium	Bad	Low	Bad	No
Medium	Bad	Low	Good	No
High	Bad	High	Good	Yes
Medium	Good	High	Good	Yes
Low	Bad	High	Bad	No
Low	Bad	High	Bad	No
Medium	Good	High	Bad	Yes
Low	Good	Low	Good	No
High	Bad	Low	Bad	No
Medium	Bad	High	Good	No
High	Bad	Low	Bad	No
Medium	Good	High	Bad	Yes

FIG. 7.10 Training data for GTS recruitment

Let us try to solve this problem, i.e. predicting whetherChandra will get a job offer, by using the decision tree model. First, we need to draw the decision tree corresponding to the training data given in Figure 7.10. According to the table, job offer condition (i.e. the outcome) is FALSE for all the cases where **Aptitude = Low**, irrespective of other conditions. So, the feature Aptitude can be taken up as the first node of the decisiontree.

For **Aptitude = High**, job offer condition is TRUE forall the cases where **Communication = Good**. For cases where **Communication = Bad**, job offer condition is TRUE for all the cases where **CGPA = High**.

Figure 7.11 depicts the complete decision tree diagramfor the table given in Figure 7.10.
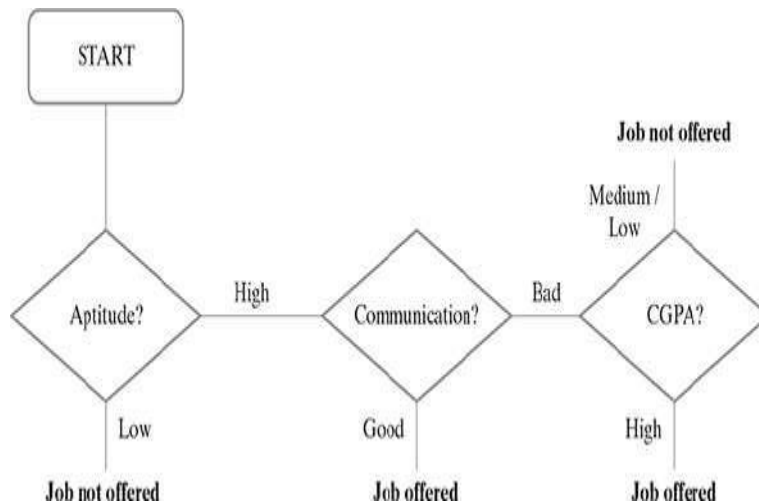
FIG. 7.11 Decision tree based on the training data

<u>Searching a decision tree</u>

By using the above decision tree depicted in Figure 7.11, we need to predict whether Chandra might get a job offerfor the given parameter values: CGPA = **High**, Communication = **Bad**, Aptitude = **High**, Programming skills = **Bad**. There are multiple ways to search through the trained decision tree for a solution to the given prediction problem.

<u>Exhaustive search</u>

Place the item in the first group (class). Recursively examine solutions with the item in the first group (class).

Place the item in the second group (class). Recursively examine solutions with the item in the second group (class).

Repeat the above steps until the solution is reached.

Exhaustive search travels through the decision treeexhaustively, but it will take much time when the decision tree is big with multiple leaves and multiple attribute values.

<u>Branch and bound search</u>

Branch and bound uses an existing best solution to sidestep searching of the entire decision tree in full.When the algorithm starts, the best solution is well defined to have the worst possible value; thus, any solution it finds out is an improvement. This makes the algorithm initially run down to the left-most branch of the tree, even though that is unlikely to produce a realistic result. In the partitioning problem, that solution corresponds to putting every item in one group, and it is an unacceptable solution. A programme can speed up theprocess by using a fast heuristic to find an initial solution. This can be used as an input for branch and bound. If the heuristic is right, the savings can be substantial.
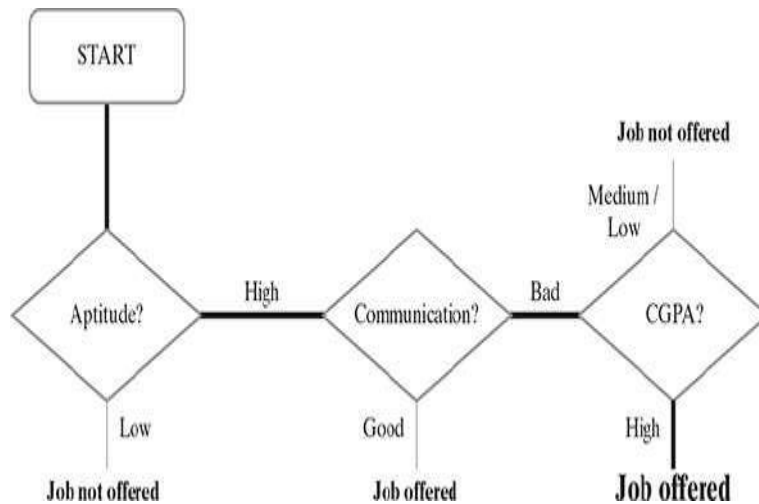
FIG. 7.12 Decision tree based on the training data (depicting a sample path)

Figure 7.12 depicts a sample path (thick line) for the conditions CGPA = **High**, Communication = Bad, Aptitude = High and Programming skills = Bad.

According to the above decision tree, the prediction canbe made as Chandra **will get the job offer**.

There are many implementations of decision tree, the most prominent ones being C5.0, CART (Classification and Regression Tree), CHAID (Chi-square Automatic Interaction Detector) and ID3 (Iterative Dichotomiser 3) algorithms. The biggest challenge of a decision tree algorithm is to find out which feature to split upon. The main driver for identifying the feature is that the data should be split in such a way that the partitions created by the split should contain examples belonging to a single class. If that happens, the partitions are considered to be **pure**. Entropy is a measure of impurity of an attribute or feature adopted by many algorithms such as ID3 and C5.0. The information gain is calculated on the basis of the decrease in entropy (S) after a data setis split according to a particular attribute (A).

Constructing a decision tree is all about finding an attribute that returns the highest information gain (i.e.the most homogeneous branches).

Note:

Like information gain, there are other measures likeGini index or chi-square for individual nodes to decide the feature on the basis of which the split hasto be applied. The CART algorithm uses Gini index, while the CHAID algorithm uses chi-square for deciding the feature for applying split.

Entropy of a decision tree

Let us say S is the sample set of training examples. Then,Entropy (S) measuring the impurity of S is defined as

$$\text{Entropy}(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

where c is the number of different class labels and prefers to the proportion of values falling into the i-th class label.

For example, with respect to the training data in Figure 7.10, we have two values for the target class 'JobOffered?' – Yes and No. The value of p_i for class value 'Yes' is 0.44 (i.e. 8/18) and that for class value 'No' is

0.56 (i.e. 10/18). So, we can calculate the entropy as

Entropy(S) = -0.44 log $_2$(0.44) - 0.56 log $_2$(0.56) = 0.99.

<u>Information gain of a decision tree</u>

The information gain is created on the basis of the decrease in entropy (*S*) after a data set is split according to a particular attribute (*A*). Constructing a decision tree is all about finding an attribute that returns the highest information gain (i.e. the most homogeneous branches). If the information gain is 0, it means that there is no reduction in entropy due to split of the data set according to that particular feature. On the other hand, the maximum amount of information gain which may happen is the entropy of the data set before the split.

Information gain for a particular feature A is calculated by the difference in entropy before a split (or S_{bs}) with the entropy after the split (S_{as}).

Information Gain (S, A) = Entropy (S$_{bs}$) − Entropy (S$_{as}$)

For calculating the entropy after split, entropy for all partitions needs to be considered. Then, the weighted summation of the entropy for each partition can be taken as the total entropy after split. For performing weighted summation, the proportion of examples falling into each partition is used as weight.

$$\text{Entropy}(S_{as}) = \sum_{i=1}^{n} w_i \, \text{Entropy}(p_i)$$

Let us examine the value of information gain for the training data set shown in Figure 7.10. We will find the value of entropy at the beginning before any split happens and then again after the split happens. We will compare the values for all the cases –

when the feature 'CGPA' is used for the split

when the feature 'Communication' is used for the split

when the feature 'Aptitude' is used for the split

when the feature 'Programming Skills' is used for the split

Figure 7.13a gives the entropy values for the first level split for each of the cases mentioned above.

As calculated, entropy of the data set before split (i.e.Entropy (S_{bs})) = 0.99, and entropy of the data set after split (i.e. Entropy (S_{as})) is

- . 0.69 when the feature 'CGPA' is used for split
- · 0.63 when the feature 'Communication' is used for split
- · 0.52 when the feature 'Aptitude' is used for split
- · 0.95 when the feature 'Programming skill' is used for split

(a) Original data set:

	Yes	No	Total
Count	8	10	18
pi	0.44	0.56	
-pi*log(pi)	0.52	0.47	0.99

Total Entropy = 0.99

(b) Splitted data set (based on the feature 'CGPA'):

CGPA = High

	Yes	No	Total
Count	4	2	6
pi	0.67	0.33	
-pi*log(pi)	0.39	0.53	0.92

Total Entropy = 0.69

CGPA = Medium

	Yes	No	Total
Count	4	3	7
pi	0.57	0.43	
-pi*log(pi)	0.46	0.52	0.99

Information Gain = 0.30

CGPA = Low

	Yes	No	Total
Count	0	5	5
pi	0.00	1.00	
-pi*log(pi)	0.00	0.00	0.00

(c) Splitted data set (based on the feature 'Communication'):

Communication = Good

	Yes	No	Total
Count	7	2	9
pi	0.78	0.22	
-pi*log(pi)	0.28	0.48	0.76

Total Entropy = 0.63

Communication = Bad

	Yes	No	Total
Count	1	8	9
pi	0.11	0.89	
-pi*log(pi)	0.35	0.15	0.50

Information Gain = 0.36

(d) Splitted data set (based on the feature 'Aptitude'):

Aptitude = High

	Yes	No	Total
Count	8	3	11
pi	0.73	0.27	
-pi*log(pi)	0.33	0.51	0.85

Total Entropy = 0.52

Aptitude = Low

	Yes	No	Total
Count	0	7	7
pi	0.00	1.00	
-pi*log(pi)	0.00	0.00	0.00

Information Gain = 0.47

(e) Splitted data set (based on the feature 'Programming Skill'):

Programming Skill = Good   Programming Skill = Bad

	Yes	No	Total
Count	5	4	9
pi	0.56	0.44	
-pi*log(pi)	0.47	0.52	0.99

Total Entropy = 0.95

	Yes	No	Total
Count	3	6	9
pi	0.33	0.67	
-pi*log(pi)	0.53	0.39	092

Information Gain = 0.04

FIG. 7.13A Entropy and information gain calculation (Level 1)

Therefore, the information gain from the feature 'CGPA' = 0.99 − 0.69 = 0.3, whereas the information gain from the feature 'Communication' = 0.99 − 0.63 =

0.36. Likewise, the information gain for 'Aptitude' and 'Programming skills' is 0.47 and 0.04, respectively.

Hence, it is quite evident that among all the features, 'Aptitude' results in the best information gain when adopted for the split. So, at the first level, a split will be applied according to the value of 'Aptitude' or in other words, 'Aptitude' will be the first node of the decision tree formed. One important point to be noted here is that for **Aptitude = Low**, entropy is 0, which indicates that always the result will be the same irrespective of the values of the other features. Hence, the branch towards Aptitude = Low will not continue any further.

As a part of level 2, we will thus have only one branch to navigate in this case – the one for **Aptitude = High**. Figure 7.13b presents calculations for level 2. As can be seen from the figure, the entropy value is as follows:

- . 0.85 before the split
- ˙ 0.33 when the feature 'CGPA' is used for split
- ˙ 0.30 when the feature 'Communication' is used for split
- ˙ when the feature 'Programming skill' is used for split

Hence, the information gain after split with the features CGPA, Communication and Programming Skill is 0.52, 0.55 and 0.05, respectively. Hence, the feature Communication should be used for this split as it results in the highest information gain. So, at the second level, a split will be applied on the basis of the value of 'Communication'. Again, the point to be noted here is that for **Communication = Good**, entropy is 0, which indicates that always the result will be the same irrespective of the values of the other features. Hence, the branch towards Communication = Good will not continue any further.

Aptitude = High

CGPA	Communication	Programming Skill	Job offered?
High	Good	Good	Yes
Medium	Good	Good	Yes
High	Good	Bad	Yes
High	Good	Good	Yes
High	Bad	Good	Yes
Medium	Good	Good	Yes
Low	Bad	Bad	No
Low	Bad	Bad	No
Medium	Good	Bad	Yes
Medium	Bad	Good	No
Medium	Good	Bad	Yes

(a) Level 2 starting set:

	Yes	No	Total
Count	8	3	11
pi	0.73	0.27	
-pi*log(pi)	0.33	0.51	0.85

Total Entropy = 0.85

(b) Splitted data set (based on the feature 'CGPA'):

CGPA = High

	Yes	No	Total
Count	4	0	4
pi	1.00	0.00	
-pi*log(pi)	0.00	0.00	0.00

CGPA = Medium

	Yes	No	Total
Count	4	1	5
pi	0.80	0.20	
-pi*log(pi)	0.26	0.46	0.72

CGPA = Low

	Yes	No	Total
Count	0	2	2
pi	0.00	1.00	
-pi*log(pi)	0.00	0.00	0.00

Total Entropy = 0.33   Information Gain = 0.52

(c) Splitted data set (based on the feature 'Communication'):

Communication = Good   Communication = Bad

	Yes	No	Total
Count	7	0	7
pi	1.00	0.00	
-pi*log(pi)	0.00	0.00	0.00

	Yes	No	Total
Count	1	3	4
pi	0.25	0.75	
-pi*log(pi)	0.50	0.31	0.81

Total Entropy = 0.30

Information Gain = 0.55

(d) Spitted data set (based on the feature 'Programming Skill'):

Programming Skill = Good

Programming Skill = Bad

	Yes	No	Total
Count	5	1	6
pi	0.83	0.17	
-pi*log(pi)	0.22	0.43	0.65

	Yes	No	Total
Count	3	2	5
pi	0.60	0.40	
-pi*log(pi)	0.44	0.53	0.97

Total Entropy = 0.80

Information Gain = 0.05

FIG. 7.13B Entropy and information gain calculation (Level 2)

As a part of level 3, we will thus have only one branchto navigate in this case – the one for **Communication= Bad**. Figure 7.13c presents calculations for level 3. Ascan be seen from the figure, the entropy value is as follows:

- . before the split
- · 0 when the feature 'CGPA' is used for split
- · 0.50 when the feature 'Programming Skill' is used for split
·

Aptitude = High & Communication = Bad

CGPA	Programming Skill	Job offered?
High	Good	Yes
Low	Bad	No
Low	Bad	No
Medium	Good	No

(a) Level 2 starting set:

	Yes	No	Total
Count	1	3	4
pi	0.25	0.75	
-pi*log(pi)	0.50	0.31	0.81

Total Entropy = 0.81

(b) Splitted data set (based on the feature 'CGPA'):

CGPA = High

	Yes	No	Total
Count	1	0	1
pi	1.00	0.00	
-pi*log(pi)	0.00	0.00	0.00

CGPA = Medium

	Yes	No	Total
Count	0	1	1
pi	0.00	1.00	
-pi*log(pi)	0.00	0.00	0.00

CGPA = Low

	Yes	No	Total
Count	0	2	2
pi	0.00	1.00	
-pi*log(pi)	0.00	0.00	0.00

Total Entropy = 0.00

Information Gain = 0.81

(c) Splitted data set (based on the feature 'Programming Skill'):

Programming Skill = Good

	Yes	No	Total
Count	1	1	2
pi	0.50	0.50	
-pi*log(pi)	0.50	0.50	1.00

Programming Skill = Bad

	Yes	No	Total
Count	0	2	2
pi	0.00	1.00	
-pi*log(pi)	0.00	0.00	0.00

Total Entropy = 0.50

Information Gain = 0.31

FIG. 7.13C Entropy and information gain calculation (Level 3)

Hence, the information gain after split with the featureCGPA is 0.81, which is the maximum possible information gain (as the entropy before split was 0.81). Hence, as obvious, a split will be applied on the basis of the value of 'CGPA'. Because the maximum informationgain is already achieved, the tree will not continue any further.

<u>Algorithm for decision tree</u>

Input: Training data set, test data set (or data points)

Steps:

Do for all attributes

Calculate the entropy E_i of the attribute F_i

if $E_i < E_{min}$

then $E_{min} = E_i$ and $F_{min} = F_i$

end ifEnd do

Split the data set into subsets using the attribute F_{min}

Draw a decision tree node containing the attribute F_{min} and split the data set into subsets

Repeat the above steps until the full tree is drawn covering all the attributes of the original table.

Avoiding overfitting in decision tree – pruning

The decision tree algorithm, unless a stopping criterion is applied, may keep growing indefinitely – splitting for every feature and dividing into smaller partitions till the point that the data is perfectly classified. This, as is quite evident, results in overfitting problem. To prevent a decision tree getting overfitted to the training data, pruning of the decision tree is essential. Pruning a decision tree reduces the size of the tree such that the model is more generalized and can classify unknown and unlabelled data in a better way.

There are two approaches of pruning:

- . Pre-pruning: Stop growing the tree before it reaches perfection.
- Post-pruning: Allow the tree to grow entirely and then post-prune some of the branches from it.

In the case of pre-pruning, the tree is stopped from further growing once it reaches a certain number of decision nodes or decisions. Hence, in this strategy, the algorithm avoids overfitting as well as optimizes computational cost. However, it also stands a chance to ignore important information contributed by a feature which was skipped, thereby resulting in miss out of certain patterns in the data.

On the other hand, in the case of post-pruning, the tree is allowed to grow to the full extent. Then, by using certain pruning criterion, e.g. error rates at the nodes, the size of the tree is reduced. This is a more effective approach in terms of classification accuracy as it considers all minute information available from the training data. However, the computational cost is obviously more than that of pre-pruning.

Strengths of decision tree

- . It produces very simple understandable rules. For smaller trees, not much mathematical and computational knowledge is required to understand this model.
- Works well for most of the problems.
- It can handle both numerical and categorical variables.
- Can work well both with small and large training data sets.
- Decision trees provide a definite clue of which features are more useful for classification.

Weaknesses of decision tree

- Decision tree models are often biased towards features having more number of possible values, i.e. levels.
- This model gets overfitted or underfitted quite easily.
- Decision trees are prone to errors in classification problems with many classes and relatively small number of training examples.
- A decision tree can be computationally expensive to train.
- Large trees are complex to understand.

Application of decision tree

Decision tree can be applied in a data set in which there is a finite list of attributes and each data instance stores a value for that attribute (e.g. 'High' for the attribute CGPA). When each attribute has a small number of distinct values (e.g. 'High', 'Medium', 'Low'), it is easier/quicker for the decision tree to suggest (or choose) an effective solution. This algorithm can be extended to handle real-value attributes (e.g. a floating point temperature).

The most straightforward case exists when there are only two possible values for an attribute (Boolean classification). Example: Communication has only two values as 'Good' or 'Bad'. It is also easy to extend the decision tree to create a target function with more than two possible output values. Example: CGPA can take one of the values from 'High', 'Medium', and 'Low'. Irrespective of whether it is a binary value/ multiple values, it is discrete in nature. For example, Aptitude can take the value of either 'High' or 'Low'. It is not possible to assign the value of both 'High' and 'Low' to the attribute Aptitude to draw a decision tree.

There should be no infinite loops on taking a decision. As we move from the root node to the next level node, it should move step-by-step towards the decision node.

Otherwise, the algorithm may not give the final result for a given data. If a set of code goes in a loop, it would repeat itself forever, unless the system crashes.

A decision tree can be used even for some instances with missing attributes and instances with errors in the classification of examples or in the attribute values describing those examples; such instances are handled well by decision trees, thereby making them a robust learning method.

Points to Ponder

Balancing overfitting and underfitting in decision tree is a very tricky topic, involving more of an art than science. The way to master this art is experience in working with more number of data sets with a lot of diversity.

Random forest model

Random forest is an ensemble classifier, i.e. a combining classifier that uses and combines many decision tree classifiers. Ensembling is usually done using the concept of bagging with different feature sets. The reason for using large number of trees in random forest is to train the trees enough such that contribution from each feature comes in a number of models. After the random forest is generated by combining the trees, majority vote is applied to combine the output of the different trees. A simplified random forest model is depicted in Figure 7.14. The result from the ensemble model is usually better than that from the individual decision tree models.
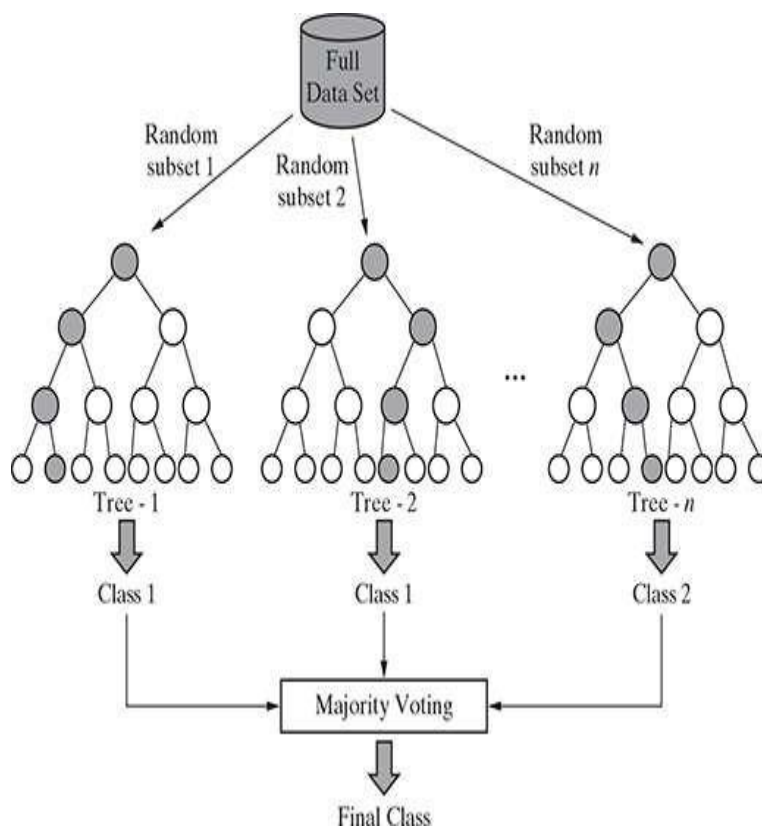
FIG. 7.14 Random forest model

How does random forest work?

The random forest algorithm works as follows:

If there are N variables or features in the input data set, select a subset of 'm' ($m < N$) features at random out of the N features. Also, the observations or data instances should be picked randomly.

Use the best split principle on these 'm' features to calculate the number of nodes 'd'.

Keep splitting the nodes to child nodes till the tree is grown to the maximum possible extent.

Select a different subset of the training data 'with replacement' to train another decision tree following steps (1) to (3). Repeat this to build and train 'n' decision trees.

Final class assignment is done on the basis of the majority votes from the '*n*' trees.

Points to Ponder

In the random forest classifier, if the number of trees is assumed to be excessively large, the model may get overfitted. In an extreme case of overfitting, the model may mimic the training data, and training error might be almost 0. However, when the model is run on an unseen sample, it may result in a very high validation error.

Out-of-bag (OOB) error in random forest

In random forests, we have seen, that each tree is constructed using a different bootstrap sample from the original data. The samples left out of the bootstrap and not used in the construction of the i-th tree can be used to measure the performance of the model. At the end of the run, predictions for each such sample evaluated each time are tallied, and the final prediction for that sample is obtained by taking a vote. The total error rate of predictions for such samples is termed as out-of-bag (OOB) error rate.

The error rate shown in the confusion matrix reflects the OOB error rate. Because of this reason, the error rate displayed is often surprisingly high.

Strengths of random forest

- It runs efficiently on large and expansive data sets.
- It has a robust method for estimating missing data and maintains precision when a large proportion of the data is absent.
- It has powerful techniques for balancing errors in a class population of unbalanced data sets.
- It gives estimates (or assessments) about which features are the most important ones in the overall classification.
- It generates an internal unbiased estimate (gauge) of the generalization error as the forest generation progresses.
- Generated forests can be saved for future use on other data.
- Lastly, the random forest algorithm can be used to solve both classification and regression problems.

Weaknesses of random forest

- This model, because it combines a number of decision tree models, is not as easy to understand as a decision tree model.
- It is computationally much more expensive than a simple model like decision tree.

Application of random forest

Random forest is a very powerful classifier which combines the versatility of many decision tree models into a single model. Because of the superior results, this ensemble model is gaining wide adoption and popularity amongst the machine learning practitioners to solve a wide range of classification problems.

Support vector machines

SVM is a model, which can do linear classification as well as regression. SVM is based on the concept of a surface, called a hyperplane, which draws a boundary between data instances plotted in the multi-dimensional feature space. The output prediction of an SVM is one of two

conceivable classes which are already defined in the training data. In summary, the SVM algorithm builds an N-dimensional hyperplane model that assigns future instances into one of the two possible output classes.

Classification using hyperplanes

In SVM, a model is built to discriminate the data instances belonging to different classes. Let us assume for the sake of simplicity that the data instances are linearly separable. In this case, when mapped in a two- dimensional space, the data instances belonging to different classes fall in different sides of a straight line drawn in the two-dimensional space as depicted in Figure 7.15a. If the same concept is extended to a multi- dimensional feature space, the straight line dividing datainstances belonging to different classes transforms to a hyperplane as depicted in Figure 7.15b.

Thus, an SVM model is a representation of the input instances as points in the feature space, which are mapped so that an apparent gap between them divides the instances of the separate classes. In other words, the goal of the SVM analysis is to find a plane, or rather a hyperplane, which separates the instances on the basis oftheir classes. New examples (i.e. new instances) are then mapped into that same space and predicted to belong to a class on the basis of which side of the gap the new instance will fall on. In summary, in the overall training process, the SVM algorithm analyses input data and identifies a surface in the multi-dimensional feature space called the hyperplane. There may be many possible hyperplanes, and one of the challenges with the SVMmodel is to find the optimal hyperplane.

(a) Two-dimensional feature space   (b) Multi-dimensional feature space
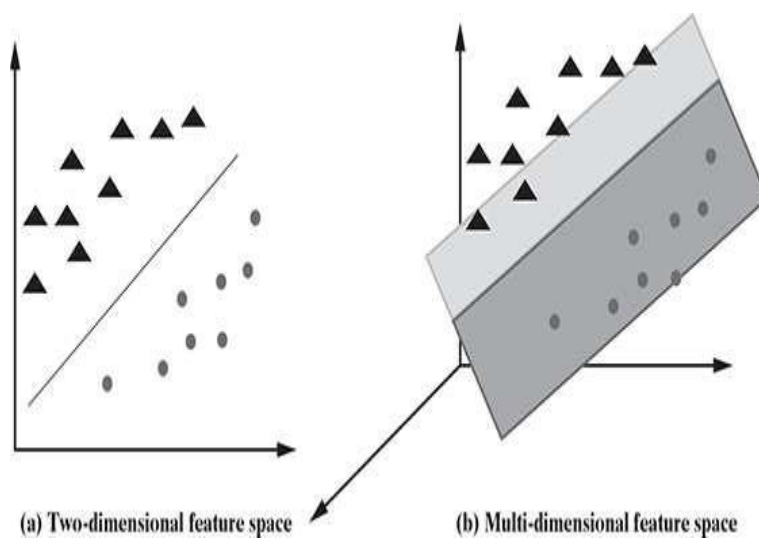
FIG. 7.15 Linearly separable data instances

Training data sets which have a substantial grouping periphery will function well with SVM. Generalization error in terms of SVM is the measure of how accuratelyand precisely this SVM model can predict values for previously unseen data (new data). A hard margin in terms of SVM means that an SVM model is inflexible in classification and tries to work exceptionally fit in the training set, thereby causing overfitting.

Support Vectors: Support vectors are the data points (representing classes), the critical component in a data set, which are near the identified set of lines (hyperplane). If support vectors are removed, they will alter the position of the dividing hyperplane.

Hyperplane and Margin: For an *N*-dimensional feature space, hyperplane is a flat subspace of dimension $(N-1)$ that separates and classifies a set of data. For example, if we consider a two-dimensional feature space (which is nothing but a data set having two features and a class variable), a hyperplane will be a one-dimensional subspace or a straight line. In the same way, for a three- dimensional feature space (data set having three features and a class variable), hyperplane is a two-dimensional subspace or a simple plane. However, quite understandably, it is difficult to visualize a feature space greater than three dimensions, much like for a subspace or hyperplane having more than three dimensions.

Mathematically, in a two-dimensional space, hyperplane can be defined by the equation:

$c_0 + c_1 X_1 + c_2 X_2 = 0$, which is nothing but an equation of a straight line.

Extending this concept to an *N*-dimensional space, hyperplane can be defined by the equation:

$c_0 + c_1 X_1 + c_2 X_2 + \ldots + c_N X_N = 0$ which, in short, can be represented as follows:

$$\vec{c}.\vec{X} + c_0 = 0$$

Spontaneously, the further (or more distance) from the hyperplane the data points lie, the more confident we can be about correct categorization. So, when a new testing data point/data set is added, the side of the hyperplane it lands on will decide the class that we assign to it. The distance between hyperplane and data points is known as **margin**.

Identifying the correct hyperplane in SVM

As we have already discussed, there may be multiple options for hyperplanes dividing the data instances belonging to the different classes. We need to identify which one will result in the best classification. Let us examine a few scenarios before arriving to that conclusion. For the sake of simplicity of visualization, the hyperplanes have been shown as straight lines in most of the diagrams.

Scenario 1

As depicted in Figure 7.16, in this scenario, we have three hyperplanes: A, B, and C. Now, we need to identify the correct hyperplane which better segregates the two classes represented by the triangles and circles. As we can see, hyperplane 'A' has performed this task quite well.
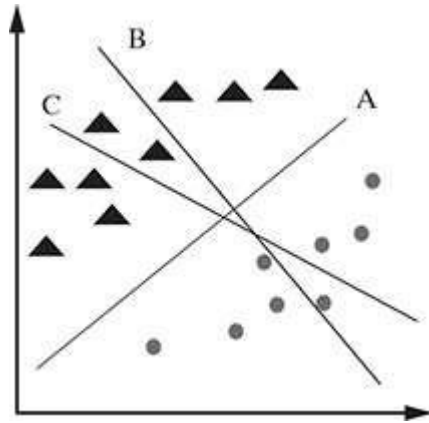
FIG. 7.16 Support vector machine: Scenario 1

As depicted in Figure 7.17, we have three hyperplanes: A,B, and C. We have to identify the correct hyperplane which classifies the triangles and circles in the best possible way. Here, maximizing the distances between the nearest data points of both the classes and hyperplane will help us decide the correct hyperplane.

This distance is called as **margin**.

In Figure 7.17b, you can see that the margin for hyperplane A is high as compared to those for both B and C. Hence, hyperplane A is the correct hyperplane. Another quick reason for selecting the hyperplane with higher margin (distance) is robustness. If we select a hyperplane having a lower margin (distance), then there is a high probabili ty of misclassification.
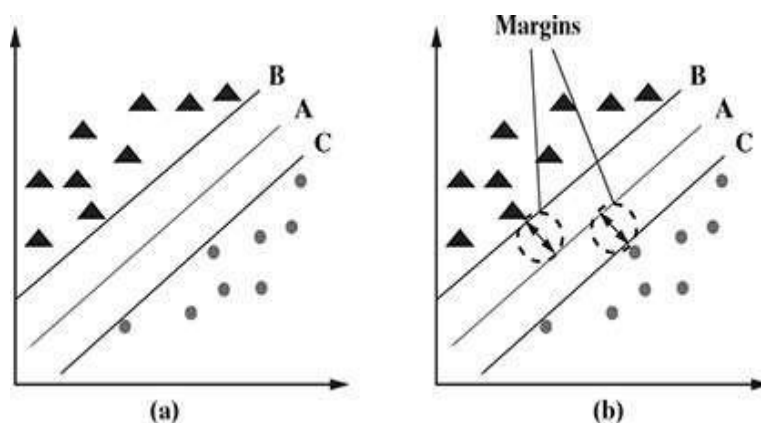
FIG. 7.17 Support vector machine: Scenario 2

Scenario 3

Use the rules as discussed in the previous section to identify the correct hyperplane in the scenario shown in Figure 7.18. Some of you might have selected hyperplane B as it has a higher margin (distance from the class) than

A. But, here is the catch; SVM selects the hyperplane which classifies the classes accurately before maximizing the margin. Here, hyperplane B has a classification error, and A has classified all data instances correctly. Therefore, A is the correct hyperplane.
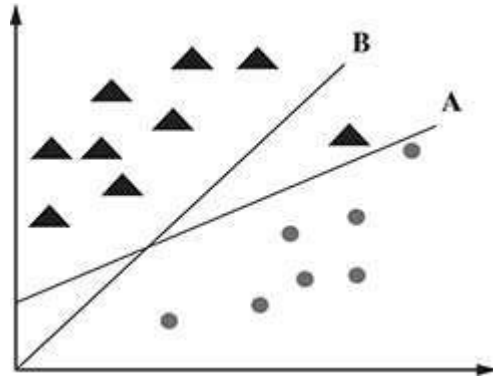
FIG. 7.18 Support vector machine: Scenario 3

Scenario 4

In this scenario, as shown in Figure 7.19a, it is not possible to distinctly segregate the two classes by using astraight line, as one data instance belonging to one of the classes (triangle) lies in the territory of theother class (circle) as an outlier.

One triangle at the other end is like an outlier for thetriangle class. SVM has a feature to ignore outliers and find the hyperplane that has the maximum margin (hyperplane A, as shown in Fig. 7.19b). Hence, we can say that SVM is robust to outliers.
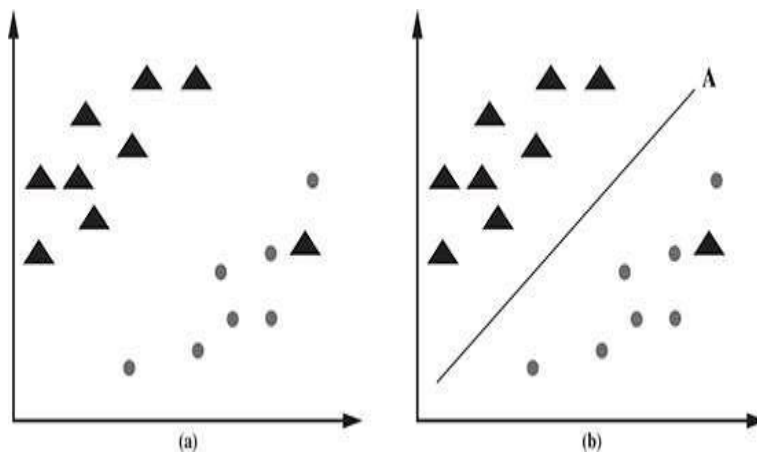
FIG. 7.19 Support vector machine: Scenario 4

So, by summarizing the observations from thedifferent scenarios, we can say that

The hyperplane should segregate the data instances belonging tothe two classes in the best possible way.

It should maximize the distances between the nearest datapoints of both the classes, i.e. maximize the margin.

If there is a need to prioritize between higher margin and lessermisclassification, the hyperplane should try to reduce misclassifications.

Our next focus is to find out a way to identify ahyperplane which maximizes the margin.

Maximum margin hyperplane

Finding the Maximum Margin Hyperplane (MMH) is nothing but identifying the hyperplane which has the largest separation with the data instances of the two classes. Though any set of three hyperplanes can do thecorrect classification, why do we need to search for the set of hyperplanes causing the largest separation? The answer is that doing so helps us in achieving more generalization and hence less number of issues in theclassification of unknown data.
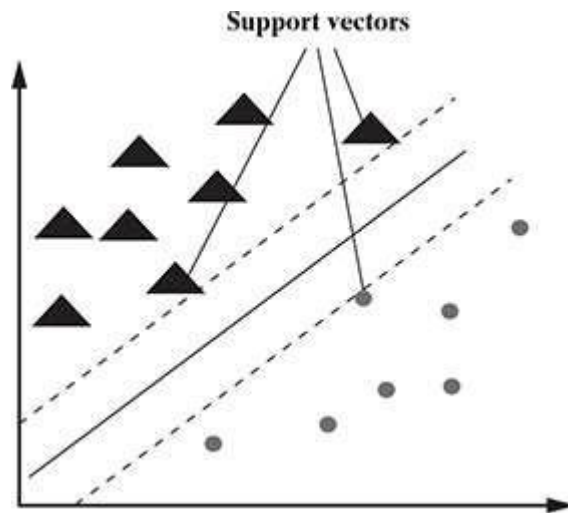
FIG. 7.20 Support vectors

Support vectors, as can be observed in Figure 7.20, are data instances from the two classes which are closestto the MMH. Quite understandably, there should be at least one support vector from each class. The identification of support vectors requires intense mathematical formulation, which is out of scope of this book. However, it is fairly intuitive to understand that modelling a problem using SVM is nothing but identifying the support vectors and MMH correspondingto the problem space.

Identifying the MMH for linearly separable data

Finding out the MMH is relatively straightforward for the data that is linearly separable. In this case, an outerboundary needs to be drawn for the data instances belonging to the different classes. These outer boundaries are known as convex hull, as depicted in Figure 7.21. The MMH can be drawn as the perpendicular bisector of the shortest line (i.e. the connecting line having the shortest length) between theconvex hulls.
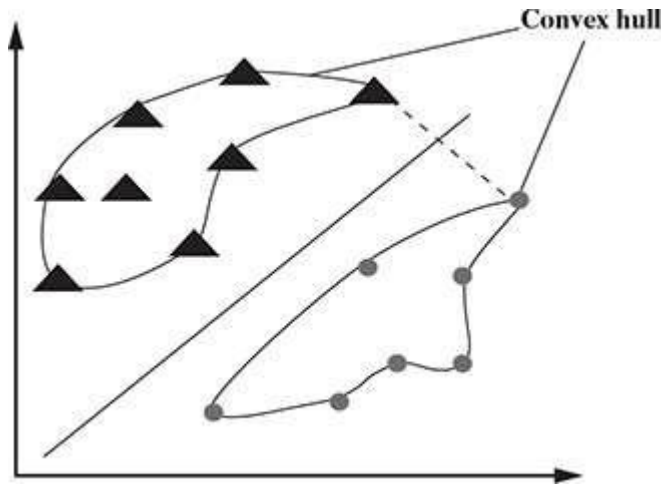
FIG. 7.21 Drawing the MMH for linearly separable data

We have already seen earlier that a hyperplane in the N-dimensional feature space can be represented by the

equation: $\vec{c}.\vec{X} + c_0 = 0$

Using this equation, the objective is to find a set of values for the vector $\vec{c}$ such that two hyperplanes,

represented by the equations below, can be specified.

$$\vec{c}.\vec{X} + c_0 \geq +1$$
$$\vec{c}.\vec{X} + c_0 \leq -1$$

This is to ensure that all the data instances that belong to one class falls above one hyperplane and all the data instances belonging to the other class falls below another hyperplane. According to vector geometry, the distance

of these planes should be $\dfrac{2}{\vec{c}}$. It is quite obvious that in

order to maximize the distance between hyperplanes, the value of $\vec{c}$ should be minimized. So, in summary, the task of SVM is to solve the optimization problem:

$$\min \left(\frac{1}{2} \vec{c} \right)$$

Identifying the MMH for non-linearly separable data

Now that we have a clear understanding of how to identify the MMH for a linearly separable data set, let us do some study about how non-linearly separable data needs to be handled by SVM. For this, we have to use a slack variable ξ, which provides some soft margin for data instances in one class that fall on the wrong side of the hyperplane. As depicted in Figure 7.22, a data instance belonging to the circle class falls on the side of the hyperplane designated for the data instances belonging to the triangle class. The same issue also happens for a data instance belonging to the triangle class.
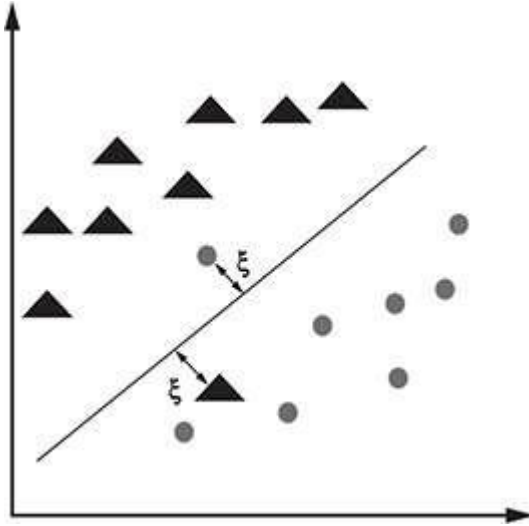
FIG. 7.22 Drawing the MMH for non-linearly separable data

A cost value 'C' is imposed on all such data instances that fall on the wrong side of the hyperplane. The task ofSVM is now to minimize the total cost due to such data instances in order to solve the revised optimization problem:

$$\min \left(\frac{1}{2}\vec{c}^2\right) + C\sum_{i=1}^{N} \xi_i$$

Kernel trick

As we have seen in the last section, one way to deal with non-linearly separable data is by using a slack variable and an optimization function to minimize the cost value.However, this is not the only way to use SVM to solve machine learning problems involving non-linearly separable data sets. SVM has a technique called the **kernel trick** to deal with non-linearly separable data. As shown in <u>Figure 7.23</u>, these are functions which can transform lower dimensional input space to a higher dimensional space. In the process, it converts linearly non-separable data to a linearly separable data. These functions are called **kernels**.

(a) Lower dimensional feature space   (b) Higher dimensional feature space
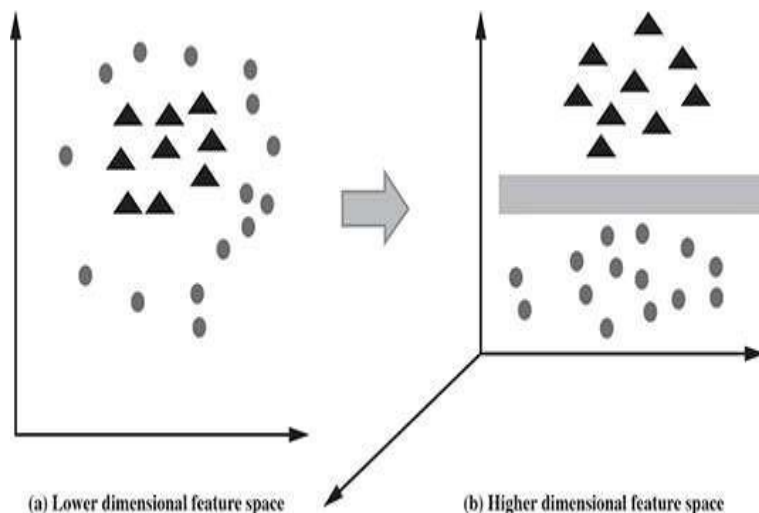
FIG. 7.23 Kernel trick in SVM

Some of the common kernel functions for transforming from a lower dimension 'i' to a higher dimension 'j' used by different SVM implementations areas follows:

- Linear kernel: It is in the $K(\vec{x_i}, \vec{x_j}) = \vec{x_i}.\vec{x_j}$
- Polynomial kernel: It is in the form

$$K(\vec{x_i}, \vec{x_j}) = (\vec{x_i}.\vec{x_j} + 1)^d$$

- Sigmoid kernel: It is in the form

$$K(\vec{x_i}, \vec{x_j}) = \tan h(k\vec{x_i}.\vec{x_j} - \partial)$$

- Gaussian RBF kernel: It is in the form

$$K(\vec{x_i}, \vec{x_j}) = e^{\dfrac{-\vec{x_i} - \vec{x_j}^2}{2\sigma^2}}$$

When data instances of the classes are closer to each other, this method can be used. The effectiveness of SVMdepends both on the

- selection of the kernel function
- adoption of values for the kernel parameters

Strengths of SVM

- SVM can be used for both classification and regression.
- It is robust, i.e. not much impacted by data with noise or outliers.
- The prediction results using this model are very promising.

Weaknesses of SVM

- SVM is applicable only for binary classification, i.e. when there areonly two classes in the problem domain.
- The SVM model is very complex – almost like a black box when it deals with a high-dimensional data set. Hence, it is very difficult andclose to impossible to understand the model in such cases.
- It is slow for a large dataset, i.e. a data set with either a large numberof features or a large number of instances.
- It is quite memory-intensive.

Application of SVM

SVM is most effective when it is used for binary classification, i.e. for solving a machine learning problemwith two classes. One common problem on which SVM can be applied is in the field of bioinformatics – more specifically, in detecting cancer and other genetic disorders. It can also be used in detecting the image of a face by binary classification of images into face and non- face components. More such applications can be described.

MULTIPLE CHOICE QUESTIONS

1.Predicting whether a tumour is malignant or benign is anexample of?

a. Unsupervised Learning

b. Supervised Regression Problem

c. Supervised Classification Problem

d. Categorical Attribute

2. Price prediction in the domain of real estate is an example of?

a. Unsupervised Learning

b. Supervised Regression Problem

c. Supervised Classification Problem

d. Categorical Attribute

3.Let us consider two examples, say 'predicting whether a tumour is malignant or benign' and 'price prediction in the domain of real estate'. These two problems are same in nature.

a. TRUE

b. FALSE

4.Supervised machine learning is as good as the data used to train it.

a. TRUE

b. FALSE

5.Which is a type of machine learning where a target feature, which is of categorical type, is predicted for the test data on the basis of the information imparted by the training data?

a. Unsupervised Learning

b. Supervised Regression

c. Supervised Classification

d. Categorical Attribute

6.Classification is a type of supervised learning where a target feature, which is of categorical type, is predicted for the test data on the basis of the information imparted by the training data. The target categorical feature is known as?

a. Object

b. Variable

c. Method

d. Class

7.This is the first step in the supervised learning model.

a. Problem Identification

b. Identification of Required Data

c. Data Pre-processing

d. Definition of Training Data Set

8.This is the cleaning/transforming the data set in the supervised learning model.

    a. Problem Identification

    b. Identification of Required Data

    c. Data Pre-processing

    d. Definition of Training Data Set

9.This refers to the transformations applied to the identified data before feeding the same into the algorithm.

    a. Problem Identification

    b. Identification of Required Data

    c. Data Pre-processing

    d. Definition of Training Data Set

10.This step of supervised learning determines 'the type of training data set'.

    a. Problem Identification

    b. Identification of Required Data

    c. Data Pre-processing

    d. Definition of Training Data Set

11.Entire design of the programme is done over here in supervised learning.

    a. Problem Identification

    b. Training

    c. Data Pre-processing

    d. Definition of Training Data Set

12.Training data run on the algorithm is called as?

    a. Program

    b. Training

    c. Training Information

    d. Learned Function

13.SVM is an example of?

    a. Linear Classifier and Maximum Margin Classifier

    b. Non-linear Classifier and Maximum Margin Classifier

    c. Linear Classifier and Minimum Margin Classifier

d. Non-linear Classifier and Minimum Margin Classifier

14. in terms of SVM means that an SVM is inflexible in classification

    a. Hard Margin

    b. Soft Margin

    c. Linear Margin

    d. Non-linear Classifier

15. are the data points (representing classes), the important component in a data set, which are near the identified set of lines (hyperplane).

    a. Hard Margin

    b. Soft Margin

    c. Linear Margin

    d. Support Vectors

16. is a line that linearly separates and classifies a set of data.

    a. Hyperplane

    b. Soft Margin

    c. Linear Margin

    d. Support Vectors

17. The distance between hyperplane and data points is called as:

    a. Hyper Plan

    b. Margins

    c. Error

    d. Support Vectors

18. Which of the following is true about SVM?

    a. It is useful only in high-dimensional spaces

    b. It always gives an approximate value

    c. It is accurate

    d. Understanding SVM is difficult

19. Which of the following is true about SVM?

    a. It is useful only in high-dimensional spaces

    b. It requires less memory

    c. SVM does not perform well when we have a large data set

d. SVM performs well when we have a large data set

20. What is the meaning of hard margin in SVM?

    a. SVM allows very low error in classification

    b. SVM allows high amount of error in classification

    c. Underfitting

    d. SVM is highly flexible

21. What sizes of training data sets are not best suited for SVM?

    a. Large data sets

    b. Very small training data sets

    c. Medium size training data sets

    d. Training data set size does not matter

22. Support Vectors are near the hyperplane.

    a. True

    b. False

23. In SVM, these functions take a lower dimensional input space and transform it to a higher dimensional space.

    a. Kernels

    b. Vector

    c. Support Vector

    d. Hyperplane

24. Which of the following options is true about the kNN algorithm?

    a. It can be used only for classification

    b. It can be used only for regression

    c. It can be used for both classification and regression

    d. It is not possible to use for both classification and regression

25. Which of the following will be Euclidean distance between the two data points A(4,3) and B(2,3)?

    a. 1

    b. 2

    c. 4

    d. 8

26. Which of the following will be Manhattan distance between the two data points A(8,3) and B(4,3)?

    a. 1

    b. 2

    c. 4

    d. 8

27. When you find many noises in data, which of the following options would you consider in kNN?

    a. Increase the value of k

    b. Decrease the value of k

    c. Noise does not depend on k

    d. $K = 0$

28. What would be the relationship between the training time taken by 1-NN, 2-NN, and 3-NN?

    a. 1-NN > 2-NN > 3-NN

    b. 1-NN < 2-NN < 3-NN

    c. 1-NN ~ 2-NN ~ 3-NN

    d. None of these

29. Which of the following algorithms is an example of the ensemble learning algorithm?

    a. Random Forest

    b. Decision Tree

    c. kNN

    d. SVM

30. Which of the following is not an inductive bias in a decision tree?

    a. It prefers longer tree over shorter tree

    b. Trees that place nodes near the root with high information gain are preferred

    c. Overfitting is a natural phenomenon in a decision tree

    d. Prefer the shortest hypothesis that fits the data

SHORT ANSWER-TYPE QUESTIONS (5 MARKS EACH)

1. What is supervised learning? Why it is called so?
2. Give an example of supervised learning in a hospital industry.
3. Give any three examples of supervised learning.
4. What is classification and regression in a supervised learning?
5. Give some examples of common classification algorithms.

6. Explain, in brief, the SVM model.
7. What is cost of misclassification in SVM?
8. Define Support Vectors in the SVM model.
9. Define kernel in the SVM model.
10. What are the factors determining the effectiveness of SVM?
11. What are the advantages of the SVM model?
12. What are the disadvantages of the SVM model?
13. Write notes on
    a. validation error in the *k*NN algorithm
    b. choosing *k* value in the *k*NN algorithm
    c. inductive bias in a decision tree
14. What are the advantages of the *k*NN algorithm?
15. What are the disadvantages of the *k*NN algorithm?
16. Explain, in brief, the decision tree algorithm.
17. What is node and leaf in decision tree?
18. What is entropy of a decision tree?
19. Define information gain in a decision tree.
20. Write any three strengths of the decision tree method.
21. Write any three weaknesses of the decision tree method.
22. Explain, in brief, the random forest model?

LONG ANSWER-TYPE QUESTIONS (10 MARKS EACH)

1. Distinguish between supervised learning, semi-supervised learning, and unsupervised learning.
2. Explain any five examples of classification problems in detail.
3. Explain classification steps in detail.
4. Discuss the SVM model in detail with different scenarios.
5. What are the advantages and disadvantages associated with SVM?
6. Discuss the *k*NN model in detail.
7. Discuss the error rate and validation error in the *k*NN algorithm.
8. Discuss how to calculate the distance between the test data and the training data for *k*NN.
9. Write the algorithm for *k*NN.
10. What is decision tree? What are the different types of nodes? Explain in detail
11. Explain various options of searching a decision tree.
12. Discuss the decision tree algorithm in detail.
13. What is inductive bias in a decision tree? How to avoid overfitting?
14. What are the strengths and weaknesses of the decision tree method?
15. Discuss appropriate problems for decision tree learning in detail.
16. Discuss the random forest model in detail. What are the features of random forest?
17. Discuss OOB error and variable importance in random forest.

Machine Learning

R20

(20A05602T)

Unit-IV

Chapter 4

Supervised Learning: Regression

INTRODUCTION

OBJECTIVE OF THE CHAPTER :

In the last two chapters, you have got quite a good conceptual overview of supervised learning algorithm for categorical data prediction. You got a detailed understanding of all the popular models of classification that are used by machine learning practitioners to solve a wide array of prediction problems where the target variable is a categorical variable.In this chapter, we will build concepts on prediction of numerical variables – which is another key area of supervised learning. This area, known as regression, focuses on solving problems such as predicting value of real estate, demand forecast in retail, weather forecast, etc.First, you will be introduced to the most popular and simplest algorithm, namely simple linear regression. This model roots from the statistical concept of fitting a straight line and the least squares method. We will explore this algorithm in detail. In this same context, we will also explore the concept of multiple linear regression.We will then briefly touch upon the other important algorithms in regression, namely multivariate adaptive regression splines, logistic regression, and maximum likelihood estimation.By the end of this chapter, you will gain sufficient knowledge in all the aspects of supervised learning and become ready to start solving problems on your own.

EXAMPLE OF REGRESSION

We have mentioned many times that real estate price prediction is a problem that can be solved by supervised learning or, more specifically, by regression. So, what this problem really is? Let us delve a little deeper into the problem.

New City is the primary hub of the commercial activities in the country. In the last couple of decades, with increasing globalization, commercial activities have intensified in New City. Together with that, a large number of people have come and settled in the city with a dream to achieve professional growth in their lives. As an obvious fall-out, a large number of housing projects have started in every nook and corner of the city. But the demand for apartments has still outgrown the supply. To get benefit from this boom in real estate business, Karen has started a digital market agency for buying and selling real estates (including apartments, independent houses, town houses, etc.). Initially, when the business was small, she used to interact with buyers and sellers personally and help them arrive at a price quote – either for selling a property (for a seller) or for buying a property (for a buyer). Her long experience in real estate business helped her develop an intuition on what the correct price quote of a property could be – given the value of certain standard parameters such as area (sq. m.) of the property, location, floor, number of years since purchase, amenities available, etc. However, with the huge surge in the business, she is facing a big challenge. She is not able to manage personal interactions as well as setting the correct price quote for the properties all alone. She hired an assistant for managing customer interactions. But the assistant,

being new in the real estate business, is struggling with price quotations. How can Karen solve this problem?

Fortunately, Karen has a friend, Frank, who is a data scientist with in-depth knowledge in machine learning models. Frank comes up with a solution to Karen's problem. He builds a model which can predict the correct value of a real estate if it has certain standard inputs such as area (sq. m.) of the property, location, floor, number of years since purchase, amenities available, etc. Wow, that sounds to be like Karen herself doing the job! Curious to know what model Frank has used? Yes, you guessed it right. He used a regression model to solve Karen's real estate price prediction problem.

So, we just discussed about one problem which can be solved using regression. In the same way, a bunch of other problems related to prediction of numerical value can be solved using the regression model. In the context of regression, dependent variable (Y) is the one whose value is to be predicted, e.g. the price quote of the real estate in the context of Karen's problem. This variable is presumed to be functionally related to one (say, X) or more independent variables called predictors. In the context of Karen's problem, Frank used area of the property, location, floor, etc. as predictors of the model that he built. In other words, the dependent variable depends on independent variable(s) or predictor(s). Regression is essentially finding a relationship (or) association between the dependent variable (Y) and the independent variable(s) (X), i.e. to find the function 'f ' for the association Y = f (X).

COMMON REGRESSION ALGORITHMS

The most common regression algorithms are:

- Simple linear regression
- Multiple linear regression
- Polynomial regression
- Multivariate adaptive regression splines
- Logistic regression
- Maximum likelihood estimation (least squares)

Simple Linear Regression

As the name indicates, simple linear regression is the simplest regression model which involves only one predictor. This model assumes a linear relationship between the dependent variable and the predictor variable as shown in Figure 8.1.
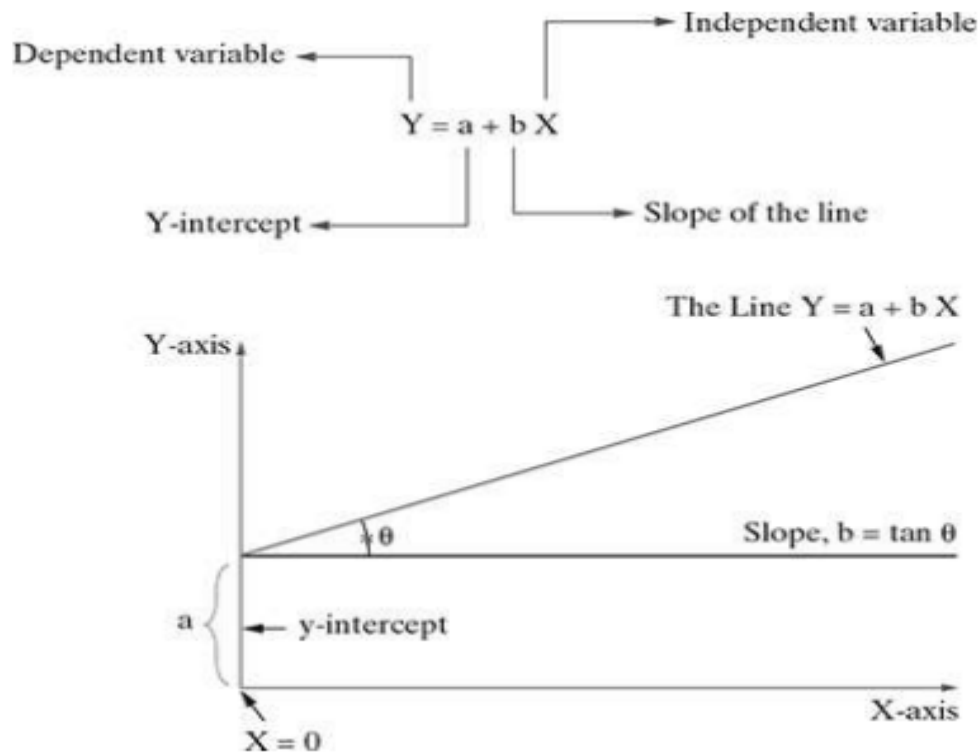
FIG. 8.1 Simple linear regression

In the context of Karen's problem, if we take Price of a Property as the ependent variable and the Area of the Property (in sq. m.) as the predictor variable, we can build a model using simple linear regression.

$$Price_{Property} = f(Area_{Property})$$

Assuming a linear association, we can reformulate the model as

$$Price_{Property} = a + b. Area_{Property}$$

where 'a' and 'b' are intercept and slope of the straight line, respectively.

Just to recall, straight lines can be defined in a slope– intercept form Y = (a + bX), where a = intercept and b = slope of the straight line. The value of intercept indicates the value of Y when X = 0. It is known as 'the intercept or Y intercept' because it specifies where the straight line crosses the vertical or Y-axis (refer to Fig. 8.1).

Slope of the simple linear regression model

Slope of a straight line represents how much the line in a graph changes in the vertical direction (Y-axis) over a change in the horizontal direction (X-axis) as shown in Figure 8.2.

Slope = Change in Y/Change in X

Rise is the change in Y-axis (Y − Y) and Run is the change in X-axis (X − X). So, slope is represented as given below:
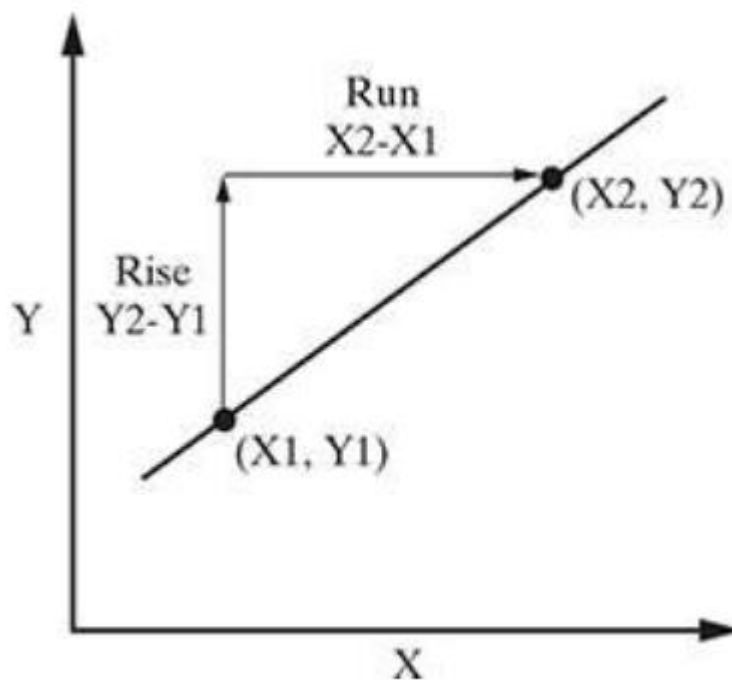
FIG. 8.2 Rise and run representation

$$\text{Slope} = \frac{\text{Rise}}{\text{Run}} = \frac{Y2 - Y1}{X2 - X1}$$

Example of slope

Let us find the slope of the graph where the lower point on the line is represented as (−3, −2) and the higher point on the line is represented as (2, 2).

(X1, Y1) = (−3, −2) and (X2 , Y2) = (2, 2)

Rise = (Y 2− Y 1) = (2 − (−2)) = 2 + 2 = 4

Run = (X 2− X1) = (2 − (−3)) = 2 + 3 = 5

Slope = Rise/Run = 4/5 = 0.8

There can be two types of slopes in a linear regression model: positive slope and negative slope. Different types of regression lines based on the type of slope include

- Linear positive slope
- Curve linear positive slope
- Linear negative slope
- Curve linear negative slope

Linear positive slope

A positive slope always moves upward on a graph from left to right (refer to Fig. 8.3).
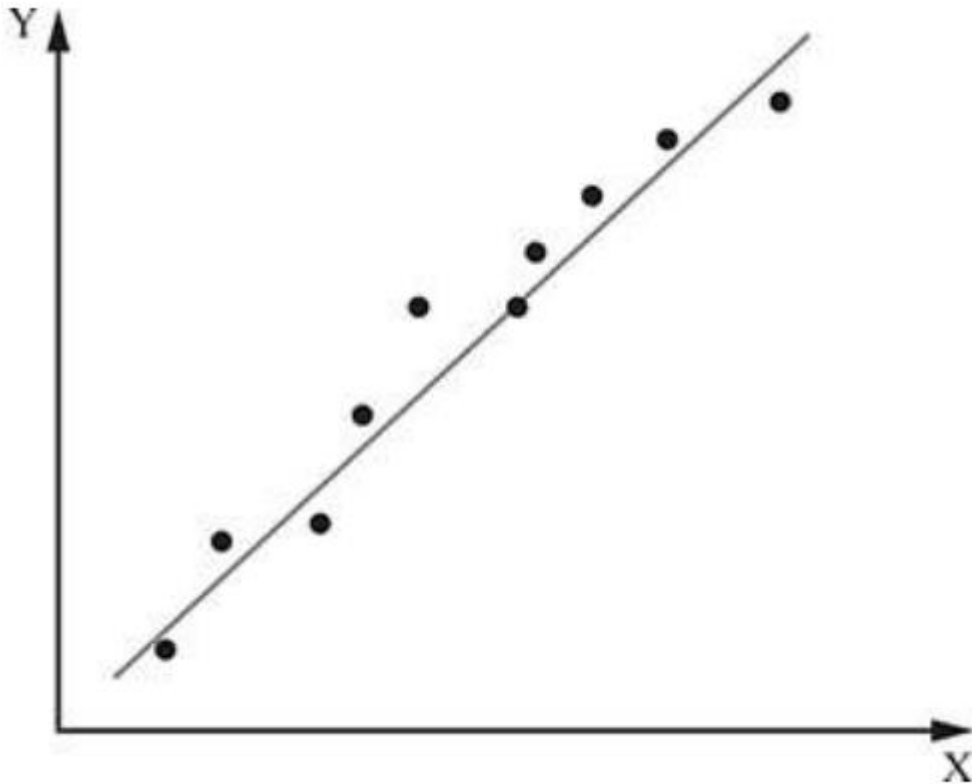
FIG. 8.3 Linear positive slope

Slope = Rise/Run = (Y − Y) / (X − X) = Delta (Y) / Delta(X)

- Scenario 1 for positive slope: Delta (Y) is positive and Delta (X) is positive

- Scenario 2 for positive slope: Delta (Y) is negative and Delta (X) is negative

Curve linear positive slope
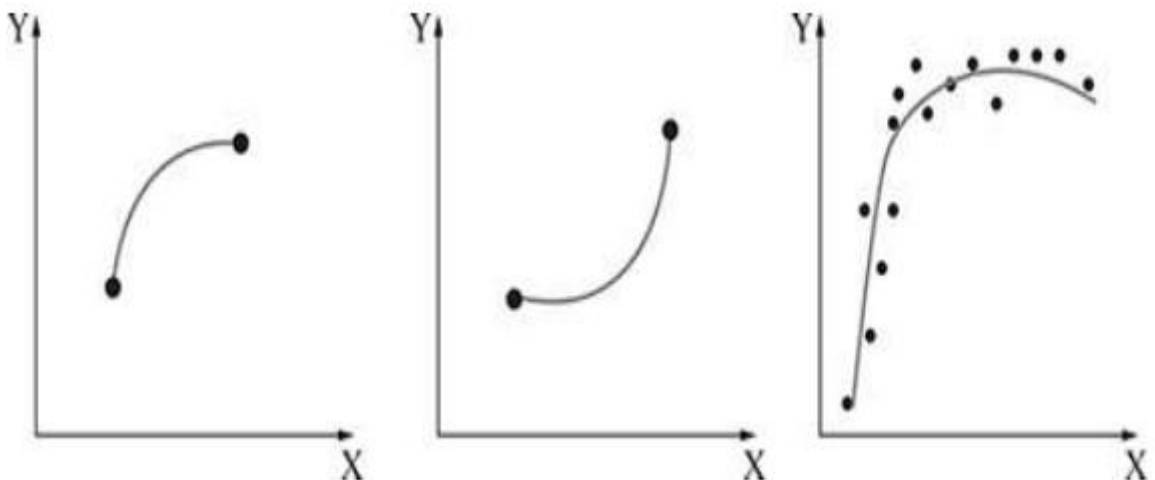
Curves in the above graphs are slope downward from left to right.

Slope = (Y2− Y1) / (X2− X1) = Delta (Y) / Delta(X)

Slope for a variable (X) may vary between two graphs,

but it will always be positive; hence, the above graphs are

called as graphs with curve linear positive slope.

Linear negative slope

A negative slope always moves downward on a graph

from left to right. As X value (on X-axis) increases, Y

value decreases (refer to Fig. 8.5).
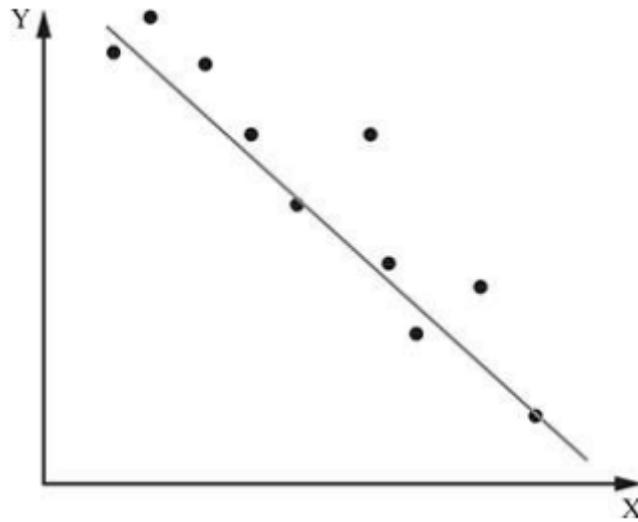
FIG. 8.5 Linear negative slope

$$\text{Slope} = \text{Rise/Run} = (Y - Y) / (X - X) = \text{Delta (Y) / Delta(X)}$$

Scenario 1 for negative slope: Delta (Y) is positive and Delta (X) is

negative

Scenario 2 for negative slope: Delta (Y) is negative and Delta (X) is

positive

**Curve linear
negative slope**
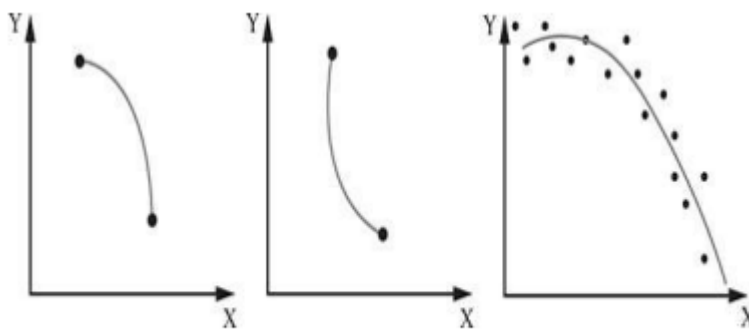
FIG. 8.6 Curve linear negative slope

Curves in these graphs (refer to Fig. 8.6) slope downward

from left to right.

Slope = (Y − Y) / (X − X) = Delta (Y) / Delta(X)

Slope for a variable (X) may vary between two graphs, but it will always be negative; hence, the above graphs are called as graphs with curve linear negative slope.

No relationship graph

Scatter graph shown in Figure 8.7 indicates 'no relationship' curve as it is very difficult to conclude whether the relationship between X and Y is positive or negative.
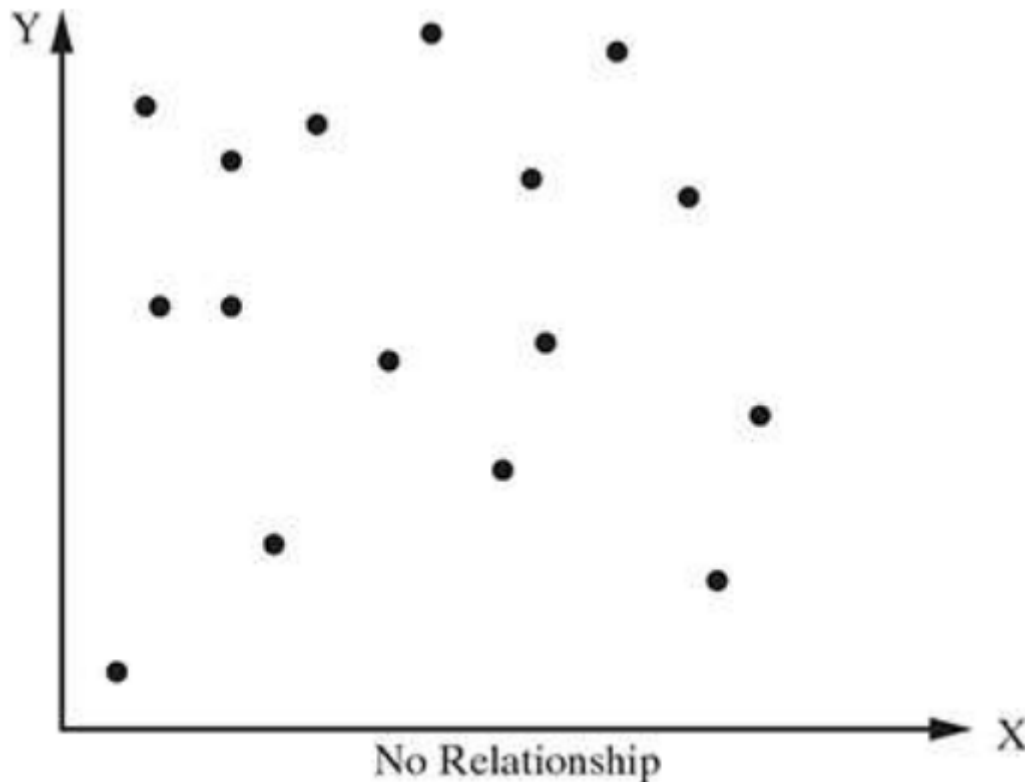
FIG. 8.7 No relationship graph

Error in simple regression

The regression equation model in machine learning uses the above slope–intercept format in algorithms. X and Y values are provided to the machine, and it identifies the values of a (intercept) and b (slope) by relating the values of X and Y. However, identifying the exact match of values for a and b is not always possible. There will be some error value (ε) associated with it. This error is called marginal or residual error.

$$Y = (a + bX) + \varepsilon$$

Now that we have some context of the simple regression model, let us try to explore an example to understand clearly how to decide the parameters of the model (i.e. values of a and b) for a given problem.

Example of simple regression

A college professor believes that if the grade for internal examination is high in a class, the grade for external examination will also be high. A random sample of 15 students in that class was selected, and the data is given below:

Internal Exam	15	23	18	23	24	22	22	19	19	16	24	11	24	16	23
External Exam	49	63	58	60	58	61	60	63	60	52	62	30	59	49	68

A scatter plot was drawn to explore the relationship between the independent variable (internal marks) mapped to X-axis and dependent variable (external marks) mapped to Y-axis as depicted in Figure 8.8.
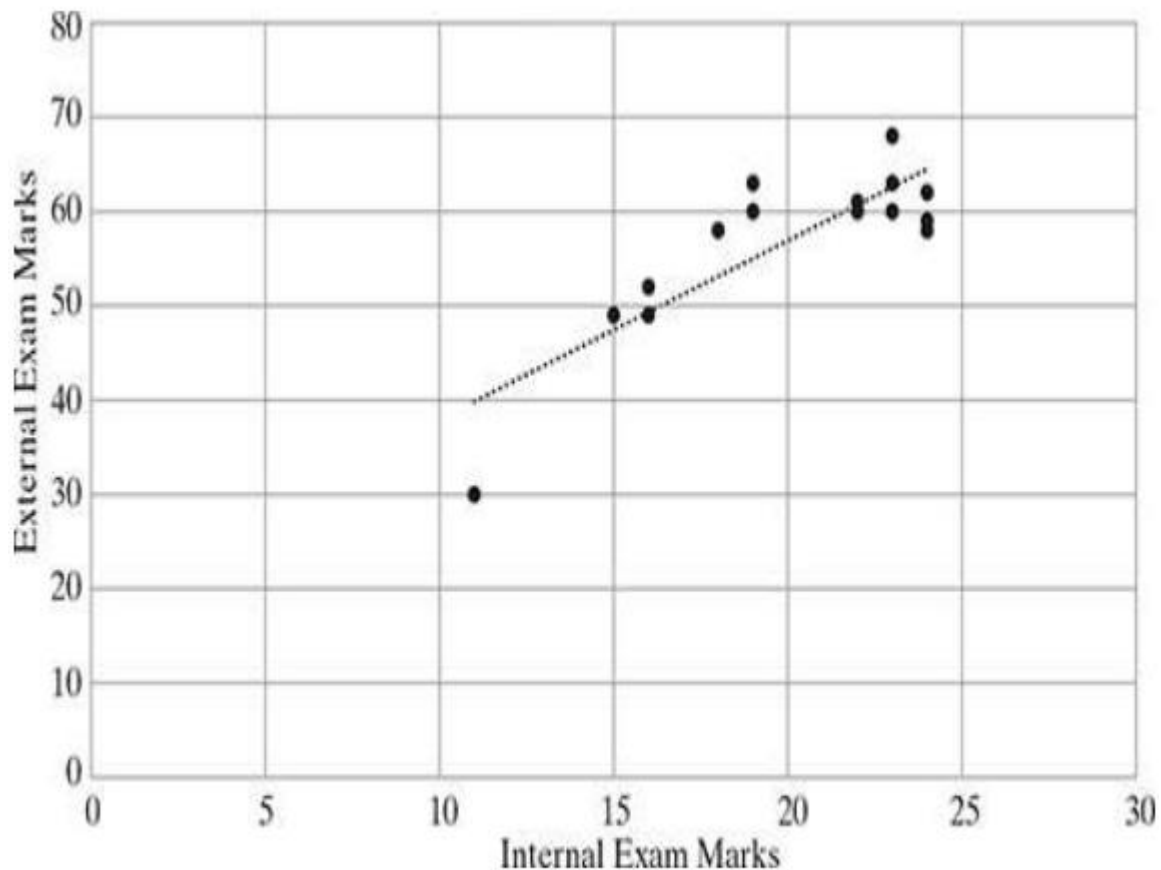
FIG. 8.8 Scatter plot and regression line

As you can observe from the above graph, the line (i.e. the regression line) does not predict the data exactly (refer to Fig. 8.8). Instead, it just cuts through the data. Some predictions are lower than expected, while some others are higher than expected.

Residual is the distance between the predicted point (on the regression line) and the actual point as depicted in Figure 8.9.
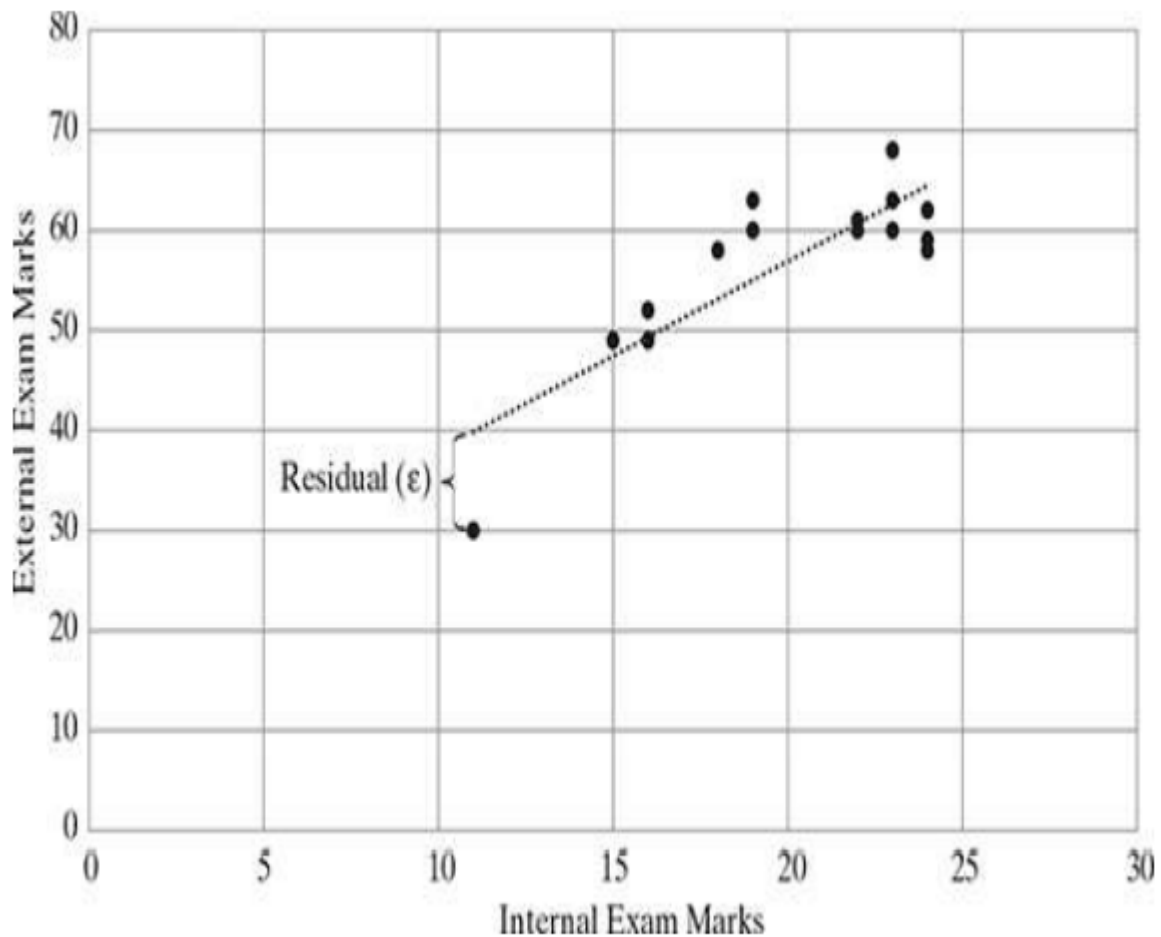
FIG. 8.9 Residual error

As we know, in simple linear regression, the line is drawn using the regression formula.

$$Y = (a + bX) + \varepsilon$$

If we know the values of 'a' and 'b', then it is easy to predict the value of Y for any given X by using the above formula. But the question is how to calculate the values of 'a' and 'b' for a given set of X and Y values?

A straight line is drawn as close as possible over the points on the scatter plot. Ordinary Least Squares (OLS) is the technique used to estimate a line that will minimize the error (ε), which is the difference between the predicted and the actual values of Y. This means summing the errors of each prediction or, more appropriately, the Sum of the Squares of the Errors

$$(SSE) \left(\text{i.e.} \sum_i \varepsilon_i^2 \right).$$

It is observed that the SSE is least when b takes the value

$$b = \frac{\sum_i (X_i - \overline{X})(Y_i - \overline{Y})}{\sum_i (X_i - \overline{X})^2} = \frac{\text{cov}(X, Y)}{\text{Var}(X)}$$

The corresponding value of 'a' calculated using the above value of 'b' is

$$a = \overline{Y} - b\overline{X}$$

So, let us calculate the value of a and b for the given example. For detailed calculation, refer to Figure 8.10.

<div align="center">Calculation summary</div>

Sum of X = 299

Sum of Y = 852

Mean X, M = 19.93

Mean Y, M = 56.8

Sum of squares (SS) = 226.9333

Sum of products (SP) = 429.8

Regression equation = ŷ = bX + a

$$b = \frac{SP}{SS_X} = \frac{429.8}{226.93} = 1.89395$$

a = M − bM = 56.8 − (1.89 × 19.93) = 19.0473

ŷ = 1.89395X + 19.0473

Hence, for the above example, the estimated regression equation is constructed on the basis of the estimated values of a and b:

<div align="center">ŷ = 1.89395X + 19.0473</div>

So, in the context of the given problem, we can say

Marks in external exam = 19.04 + 1.89 × (Marks in internal exam)

or, M = 19.04 + 1.89 × M Int

X	Y	X- mean (X)	Y- Mean (Y)	$(Xi-\bar{X})(Yi-\bar{Y})$	$(Xi-\bar{X})^2$
15	49	-4.93	-7.8	38.454	24.3049
23	63	3.07	6.2	19.034	9.4249
18	58	-1.93	1.2	-2.316	3.7249
23	60	3.07	3.2	9.824	9.4249
24	58	4.07	1.2	4.884	16.5649
22	61	2.07	4.2	8.694	4.2849
22	60	2.07	3.2	6.624	4.2849
19	63	-0.93	6.2	-5.766	0.8649
19	60	-0.93	3.2	-2.976	0.8649
16	52	-3.93	-4.8	18.864	15.4449
24	62	4.07	5.2	21.164	16.5649
11	30	-8.93	-26.8	239.324	79.7449
24	59	4.07	2.2	8.954	16.5649
16	49	-3.93	-7.8	30.654	15.4449
23	68	3.07	11.2	34.384	9.4249
19.9	56.8		$\Sigma(Xi-\bar{X})(Yi-\bar{Y})$	429.8	226.9335

Step 1 — Step 2 — Step 3 — Step 4 — Step 5 — Step 6

Step 7: Divide (step4 / step6)

$b = 429.28 / 226.93 = 1.89$

Step 8: Calculate a using the value of b

$a = \bar{Y} - b\bar{X}$

$a = 56.8 - 1.89 \times 19.9$

$a = 19.05$

FIG. 8.10 Detailed calculation of regression parameters

The model built above can be represented graphically as

an extended version (refer to Fig. 8.11)

a zoom-in version (refer to Fig. 8.12)

Interpretation of the intercept

As we have already seen, the simple linear regression model built on the data in the example is

$$M = 19.04 + 1.89 \times M\ int$$

The value of the intercept from the above equation is 19.05. However, none of the internal mark is 0. So, intercept = 19.05 indicates that 19.05 is the portion of the external examination marks not explained by the internal examination marks.
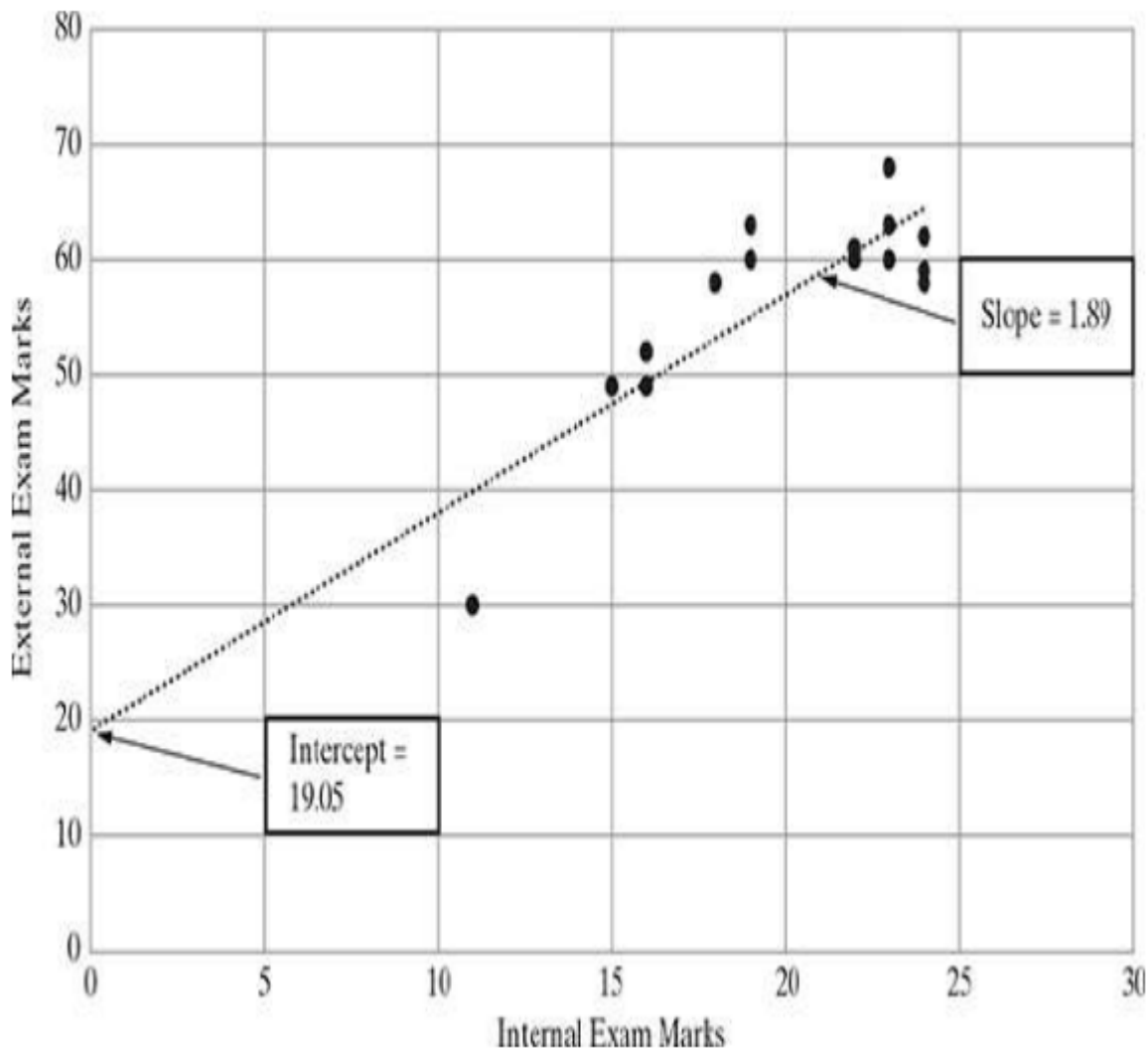
FIG. 8.11 Extended version of the regression graph.
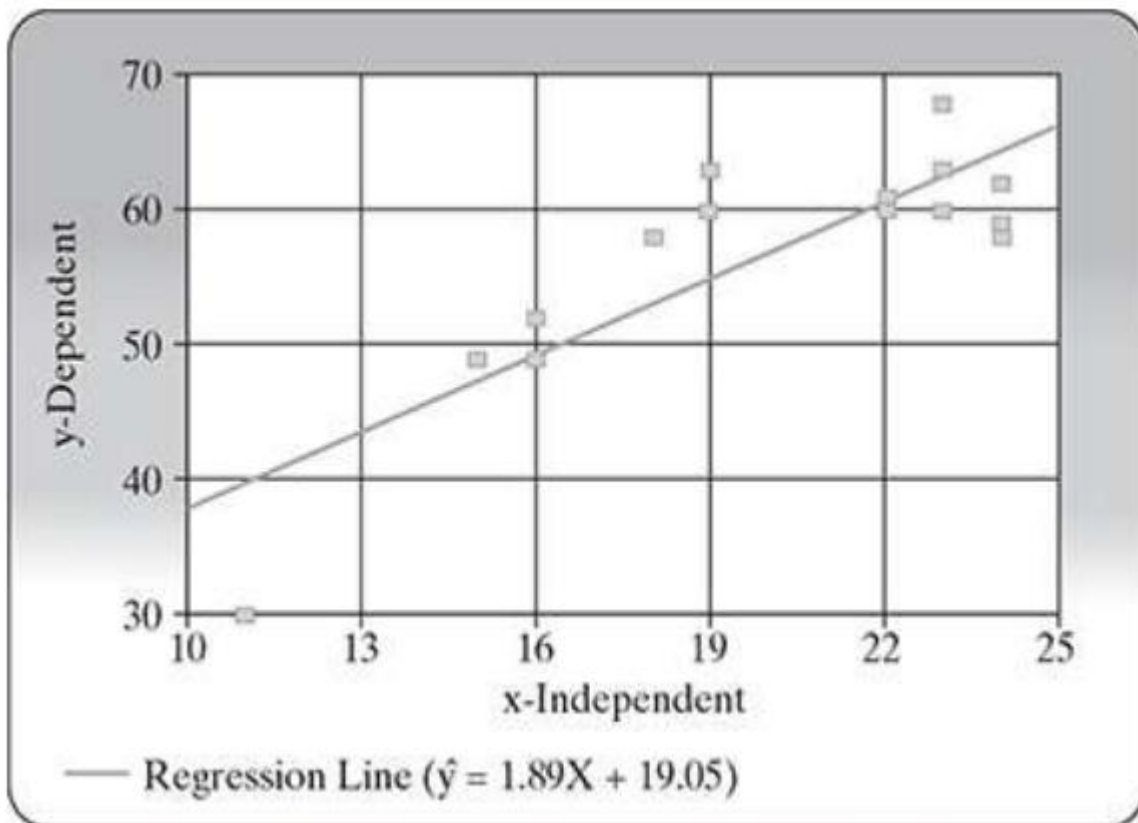
FIG. 8.12 Zoom-in regression line

Slope measures the estimated change in the average value of Y as a result of a one-unit change in X. Here, slope = 1.89 tells us that the average value of the external examination marks increases by 1.89 for each additional 1 mark in the internal examination.

Now that we have a complete understanding of how to build a simple linear regression model for a given problem, it is time to summarize the algorithm.

OLS algorithm

- Step 1: Calculate the mean of X and Y

- Step 2: Calculate the errors of X and Y

- Step 3: Get the product

- Step 4: Get the summation of the products

- Step 5: Square the difference of X

- Step 6: Get the sum of the squared difference

- Step 7: Divide output of step 4 by output of step 6 to calculate 'b'

- Step 8: Calculate 'a' using the value of 'b'

Maximum and minimum point of curves

Maximum (shown in Fig. 8.13) and minimum points (shown in Fig. 8.14) on a graph are found at points where the slope of the curve is zero. It becomes zero either from positive or negative value. The maximum point is the point on the curve of the graph with the highest ycoordinate and a slope

of zero. The minimum point is the point on the curve of the graph with the lowest ycoordinate and a slope of zero.
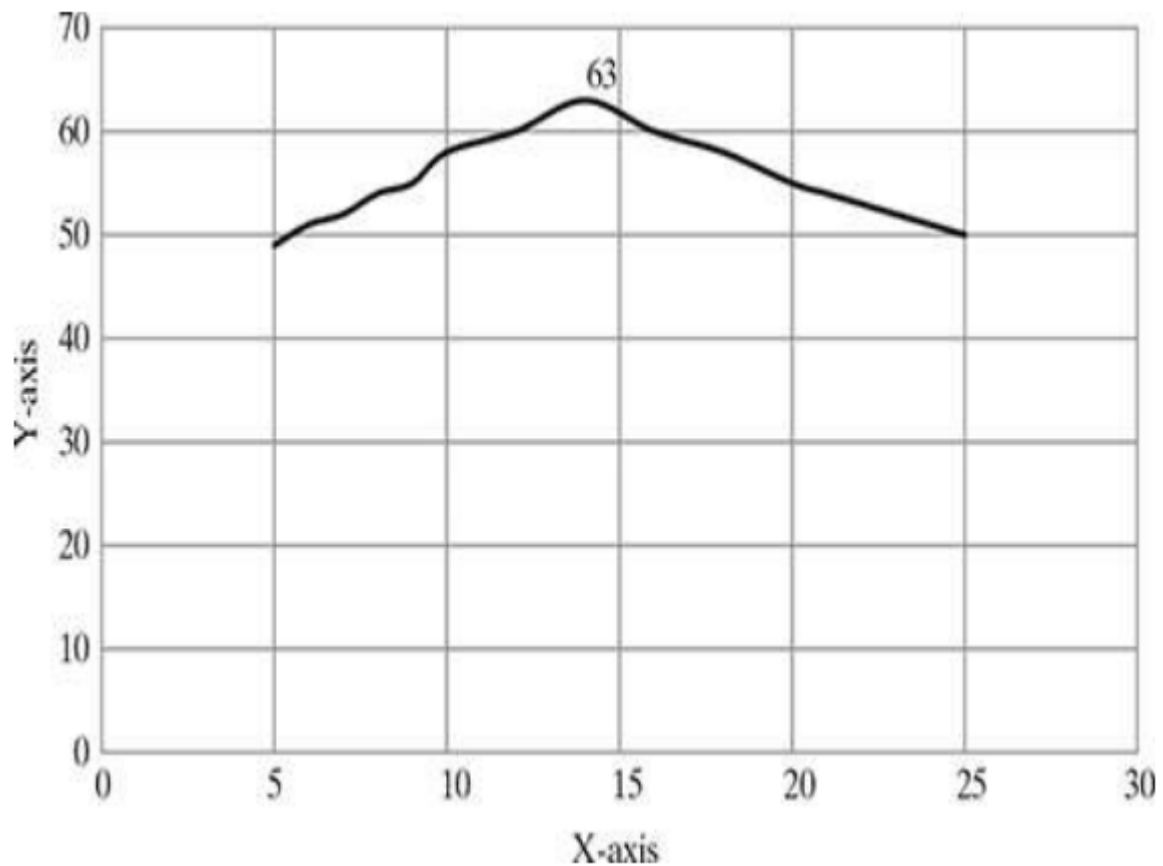
FIG. 8.13 Maximum point of curve

Point 63 is at the maximum point for this curve (refer to Fig. 8.13). Point 63 is at the highest point on this curve. It has a greater y-coordinate value than any other point on the curve and has a slope of zero.

Point 40 (marked with an arrow in Fig. 8.14) is the minimum point for this curve. Point 40 is at the lowest point on this curve. It has a lesser y-coordinate value than any other point on the curve and has a slope of zero.
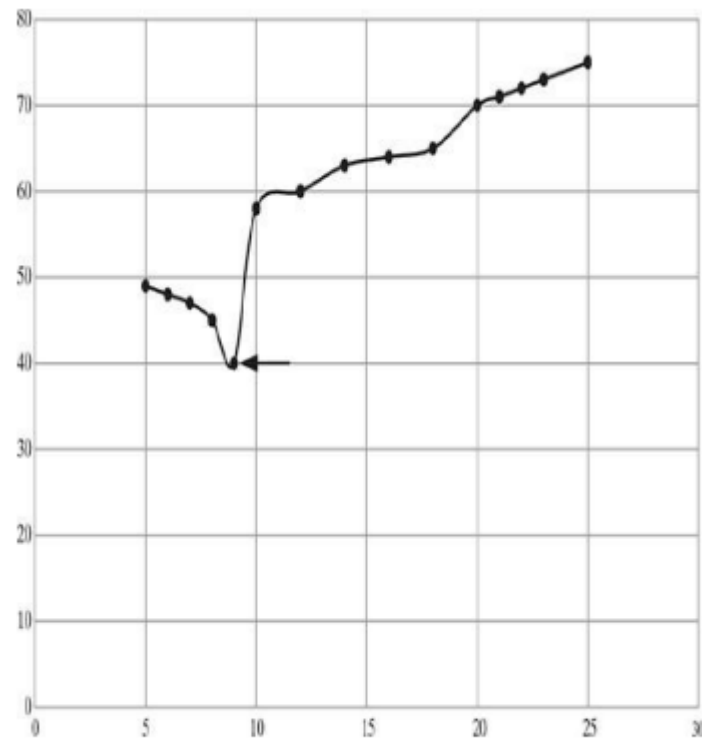
FIG. 8.14 Minimum point of curve

Multiple Linear Regression

In a multiple regression model, two or more independent variables, i.e. predictors are involved in the model. If we think in the context of Karen's problem, in the last section, we came up with a simple linear regression by considering Price of a Property as the dependent variable and the Area of the Property (in sq. m.) as the predictor variable. However, location, floor, number of years since purchase, amenities available, etc. are also important predictors which should not be ignored. Thus, if we consider Price of a Property (in $) as the dependent variable and Area of the Property (in sq. m.), location, floor, number of years since purchase and amenities available as the independent variables, we can form a multiple regression equation as shown below:

$$\text{Price}_{\text{Property}} = f(\text{Area}_{\text{Property}}, \text{ location, floor, Ageing, Amenities})$$

The simple linear regression model and the multiple regression model assume that the dependent variable is continuous.

The following expression describes the equation involving the relationship with two predictor variables, namely X and X .

$$\hat{Y} = a + b\,X + b\,X$$

The model describes a plane in the three-dimensional space of $\hat{Y}$, X , and X . Parameter 'a' is the intercept of this plane. Parameters 'b ' and 'b ' are referred to as partial regression coefficients. Parameter b represents the change in the mean response corresponding to a unit change in X when X is held constant. Parameter b represents the change in the mean response corresponding to a unit change in X when X is held constant.

Consider the following example of a multiple linear regression model with two predictor variables, namely X and X (refer to Fig. 8.15).

$$\hat{Y} = 22 + 0.3X + 1.2X$$
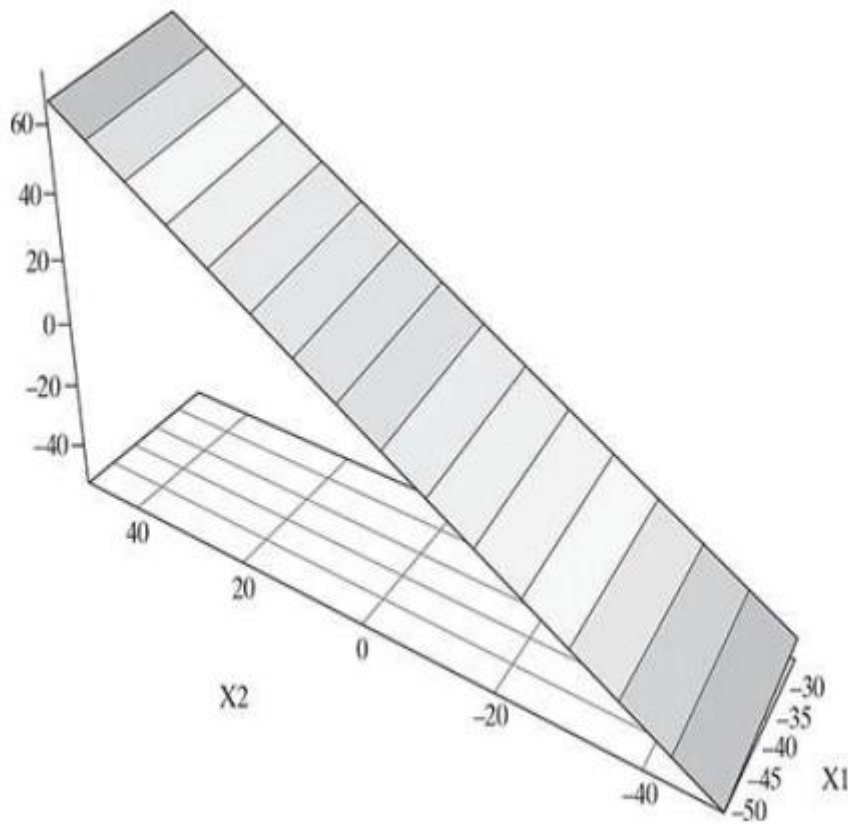
FIG. 8.15 Multiple regression plane

Multiple regression for estimating equation when there are 'n' predictor variables is as follows:

$$\hat{Y} = a + b1X1 + b2\ X2 + b3\ X3 + \ldots + bn\ Xn$$

While finding the best fit line, we can fit either a polynomial or curvilinear regression. These are known as polynomial or curvilinear regression, respectively.

Assumptions in Regression Analysis

1. The dependent variable (Y) can be calculated / predicated as a linear function of a specific set of independent variables (X's) plus an error term (ε).

2. The number of observations (n) is greater than the number of parameters (k) to be estimated, i.e. $n > k$.

3. Relationships determined by regression are only relationships of association based on the data set and not necessarily of cause and effect of the defined class.

4. Regression line can be valid only over a limited range of data. If the line is extended (outside the range of extrapolation), it may only lead to wrong predictions.

5. If the business conditions change and the business assumptions underlying the regression model are no longer valid, then the past data set will no longer be able to predict future trends.

6. Variance is the same for all values of X (homoskedasticity).

7. The error term (ε) is normally distributed. This also means that the mean of the error (ε) has an expected value of 0.

8. The values of the error (ε) are independent and are not related to any values of X. This means that there are no relationships between a particular X, Y that are related to another specific value of X, Y.

Given the above assumptions, the OLS estimator is the Best Linear Unbiased Estimator (BLUE), and this is called as Gauss-Markov Theorem.

Main Problems in Regression Analysis

In multiple regressions, there are two primary problems: multicollinearity and heteroskedasticity.

Multicollinearity

Two variables are perfectly collinear if there is an exact linear relationship between them. Multicollinearity is the situation in which the degree of correlation is not only between the dependent variable and the independent variable, but there is also a strong correlation within (among) the independent variables themselves. A multiple regression equation can make good predictions when there is multicollinearity, but it is difficult for us to determine how the dependent variable will change if each independent variable is changed one at a time. When multicollinearity is present, it increases the standard errors of the coefficients. By overinflating the standard errors, multicollinearity tries to make some variables statistically insignificant when they actually should be significant (with lower standard errors). One way to gauge multicollinearity is to calculate the Variance Inflation Factor (VIF), which assesses how much the variance of an estimated regression coefficient increases if the predictors are correlated. If no factors are correlated, the VIFs will be equal to 1.

The assumption of no perfect collinearity states that there is no exact linear relationship among the independent variables. This assumption implies two aspects of the data on the independent variables. First, none of the independent variables, other than the variable associated with the intercept term, can be a constant. Second, variation in the X's is necessary. In general, the more variation in the independent variables, the better will be the OLS estimates in terms of identifying the impacts of the different independent variables on the dependent variable.

Heteroskedasticity

Heteroskedasticity refers to the changing variance of the error term. If the variance of the error term is not constant across data sets, there will be erroneous predictions. In general, for a regression equation to make accurate predictions, the error term should be independent, identically (normally) distributed (iid).

Mathematically, this assumption is written as

$$\text{var}(u_i|X) = \sigma^2 \quad \text{and}$$
$$\text{cov}(u_i u_j|X) = 0 \quad \text{for } i \neq j$$

where 'var' represents the variance, 'cov' represents the covariance, 'u' represents the error terms, and 'X' represents the independent variables.

This assumption is more commonly written as

$$\text{var}(u_i) = \sigma^2 \quad \text{and}$$
$$\text{cov}(u_i u_j) = 0 \quad \text{for } i \neq j.$$

Improving Accuracy of the Linear Regression Model

Let us understand bias and variance in the regression model before exploring how to improve the same. The concept of bias and variance is similar to accuracy and prediction. Accuracy refers to how close the estimation is near the actual value, whereas prediction refers to continuous estimation of the value.

High bias = low accuracy (not close to real value)

High variance = low prediction (values are scattered)

Low bias = high accuracy (close to real value)

Low variance = high prediction (values are close to each other)

Let us say we have a regression model which is highly accurate and highly predictive; therefore, the overall error of our model will be low, implying a low bias (high accuracy) and low variance (high prediction). This is highly preferable. Similarly, we can say that if the variance increases (low prediction), the spread of our data points increases, which results in less accurate prediction. As the bias increases (low accuracy), the error between our predicted value and the observed values increases. Therefore, balancing out bias and accuracy is essential in a regression model.

In the linear regression model, it is assumed that the number of observations (n) is greater than the number of parameters (k) to be estimated, i.e. n > k, and in that case, the least squares estimates tend to have low variance and hence will perform well on test observations.

However, if observations (n) is not much larger than parameters (k), then there can be high variability in the least squares fit, resulting in overfitting and leading to poor predictions.

If k > n, then linear regression is not usable. This also indicates infinite variance, and so, the method cannot be used at all.

Accuracy of linear regression can be improved using the following three methods:

1. Shrinkage Approach

2. Subset Selection

3. Dimensionality (Variable) Reduction

Shrinkage (Regularization) approach

By limiting (shrinking) the estimated coefficients, we can try to reduce the variance at the cost of a negligible increase in bias. This can in turn lead to substantial improvements in the accuracy of the model.

Few variables used in the multiple regression model are in fact not associated with the overall response and are called as irrelevant variables; this may lead to unnecessary complexity in the regression model.

This approach involves fitting a model involving all predictors. However, the estimated coefficients are shrunken towards zero relative to the least squares estimates. This shrinkage (also known as regularization) has the effect of reducing the overall variance. Some of the coefficients may also be estimated to be exactly zero, thereby indirectly performing variable selection. The two best-known techniques for shrinking the regression coefficients towards zero are

1. ridge regression

2. lasso (Least Absolute Shrinkage Selector Operator)

Ridge regression performs L2 regularization, i.e. it adds penalty equivalent to square of the magnitude of coefficients

Minimization objective of ridge = LS Obj + α × (sum of square of coefficients)

Ridge regression (include all k predictors in the final model) is very similar to least squares, except that the coefficients are estimated by minimizing a slightly different quantity. If k > n, then the least squares estimates do not even have a unique solution, whereas ridge regression can still perform well by trading off a small increase in bias for a large decrease in variance. Thus, ridge regression works best in situations where the least squares estimates have high variance. One disadvantage with ridge regression is that it will include all k predictors in the final model. This may not be a problem for prediction accuracy, but it can create a challenge in model interpretation in settings in which the number of variables k is quite large. Ridge regression will perform better when the response is a function of many predictors, all with coefficients of roughly equal size.

Lasso regression performs L1 regularization, i.e. it adds penalty equivalent to the absolute value of the magnitude of coefficients.

Minimization objective of ridge = LS Obj + α × (absolute value of the magnitude of coefficients)

The lasso overcomes this disadvantage by forcing some of the coefficients to zero value. We can say that the lasso yields sparse models (involving only subset) that are simpler as well as more interpretable. The lasso can be expected to perform better in a setting where a relatively small number of predictors have substantial coefficients, and the remaining predictors have coefficients that are very small or equal to zero.

Subset selection

Identify a subset of the predictors that is assumed to be related to the response and then fit a model using OLS on the selected reduced subset of variables. There are two methods in which subset of the regression can be selected:

1. Best subset selection (considers all the possible (2))

2. Stepwise subset selection

    1. Forward stepwise selection (0 to k)

    2. Backward stepwise selection (k to 0)

In best subset selection, we fit a separate least squares regression for each possible subset of the k predictors. For computational reasons, best subset selection cannot be applied with very large value of predictors (k). The best subset selection procedure considers all the possible (2) models containing subsets of the p predictors.

The stepwise subset selection method can be applied to choose the best subset. There are two stepwise subset selection:

1. Forward stepwise selection (0 to k)

2. Backward stepwise selection (k to 0)

Forward stepwise selection is a computationally efficient alternative to best subset selection. Forward stepwise considers a much smaller set of models, that too step by step, compared to best set selection. Forward stepwise selection begins with a model containing no predictors, and then, predictors are added one by one to the model, until all the k predictors are included in the model. In particular, at each step, the variable (X) that k k gives the highest additional improvement to the fit is added.

Backward stepwise selection begins with the least squares model which contains all k predictors and then iteratively removes the least useful predictor one by one.

Dimensionality reduction (Variable reduction)

The earlier methods, namely subset selection and shrinkage, control variance either by using a subset of the original variables or by shrinking their coefficients towards zero. In dimensionality reduction, predictors (X) are transformed, and the model is set up using the transformed variables after dimensionality reduction. The number of variables is reduced using the dimensionality reduction method. Principal component analysis is one of the most important dimensionality (variable) reduction techniques.

Polynomial Regression Model

Polynomial regression model is the extension of the simple linear model by adding extra predictors obtained by raising (squaring) each of the original predictors to a power. For example, if there are three variables, X, X , and X are used as predictors. This approach provides a simple way to yield a non-linear fit to data.

$$f(x) = c0 + c\ 1.X + c\ 2.X + c3\ .X$$

In the above equation, c 0, c 1, c 2, and c 3are the coefficients.

Example: Let us use the below data set of (X, Y) for degree 3 polynomial.

Internal Exam (X)	15	23	18	23	24	22	22	19	19	16	24	11	24	16	23
External Exam (Y)	49	63	58	60	58	61	60	63	60	52	62	30	59	49	68

As you can observe, the regression line (refer to Fig. 8.16) is slightly curved for polynomial degree 3 with the above 15 data points. The regression line will curve further if we increase the polynomial degree (refer to Fig. 8.17). At the extreme value as shown below, the regression line will be overfitting into all the original values of X.
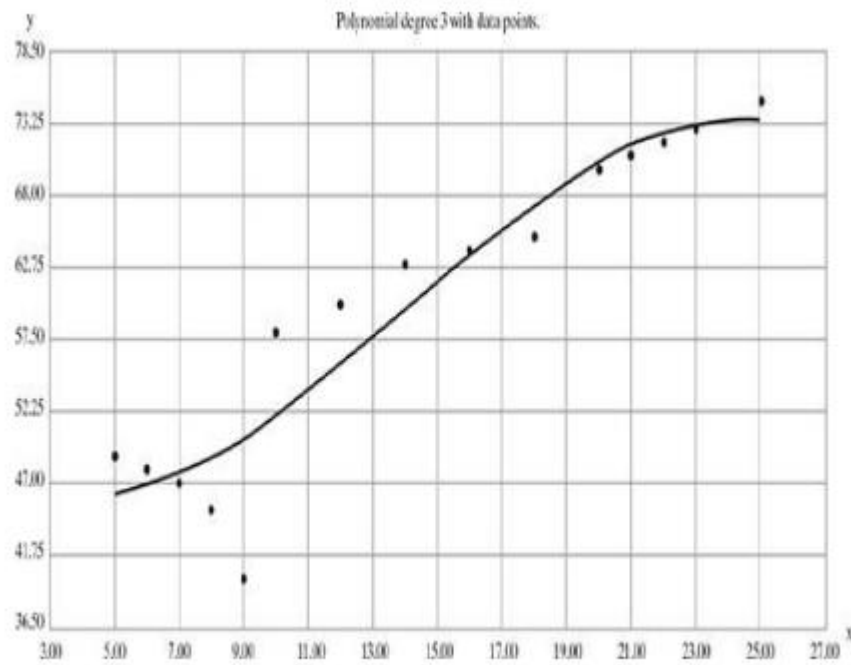
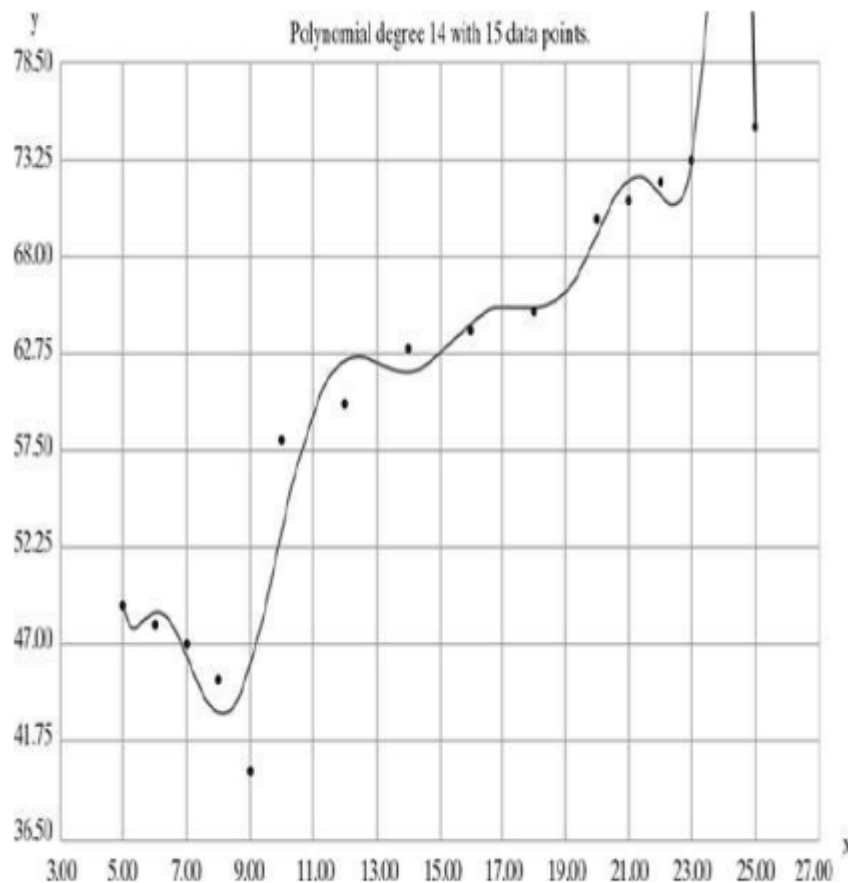**FIG. 8.16** Polynomial regression degree 3

FIG. 8.17 Polynomial regression degree 14

Logistic Regression

Logistic regression is both classification and regression technique depending on the scenario used. Logistic regression (logit regression) is a type of regression analysis used for predicting the outcome of a categorical dependent variable similar to OLS regression. In logistic regression, dependent variable (Y) is binary (0,1) and independent variables (X) are continuous in nature. The probabilities describing the possible outcomes (probability that Y = 1) of a single trial are modelled as a logistic function of the predictor variables. In the logistic regression model, there is no R to gauge the fit of the overall model; however, a chi-square test is used to gauge how well the logistic regression model fits the data. The goal of logistic regression is to predict the likelihood that Y is equal to 1 (probability that Y = 1 rather than 0) given certain values of X. That is, if X and Y have a strong positive linear relationship, the probability that a person will have a score of Y = 1 will increase as values of X increase. So, we are predicting probabilities rather than the scores of the dependent variable.

For example, we might try to predict whether or not a small project will succeed or fail on the basis of the number of years of experience of the project manager handling the project. We presume that those project managers who have been managing projects for many years will be more likely to succeed. This means that as X (the number of years of experience of project manager) increases, the probability that Y will be equal to 1 (success of the new project) will tend to increase. If we take a

hypothetical example in which 60 already executed projects were studied and the years of experience of project managers ranges from 0 to 20 years, we could represent this tendency to increase the probability that Y = 1 with a graph.

To illustrate this, it is convenient to segregate years of experience into categories (i.e. 0–8, 9–16, 17–24, 25–32, 33–40). If we compute the mean score on Y (averaging the 0s and 1s) for each category of years of experience, we will get something like

X	Y
0–8	0.27
9–16	0.5
17–24	0.6
25–32	0.66
33–40	0.93

When the graph is drawn for the above values of X and Y, it appears like the graph in Figure 8.18. As X increases, the probability that Y = 1 increases. In other words, when the project manager has more years of experience, a larger percentage of projects succeed. A perfect relationship represents a perfectly curved S rather than a straight line, as was the case in OLS regression. So, to model this relationship, we need some fancy algebra / mathematics that accounts for the bends in the curve.

An explanation of logistic regression begins with an explanation of the logistic function, which always takes values between zero and one. The logistic formulae are stated in terms of the probability that Y = 1, which is referred to as P. The probability that Y is 0 is 1 − P.

$$\ln\left(\frac{P}{1-P}\right) = a + bX$$

$$\ln(p/1-p) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \varepsilon$$
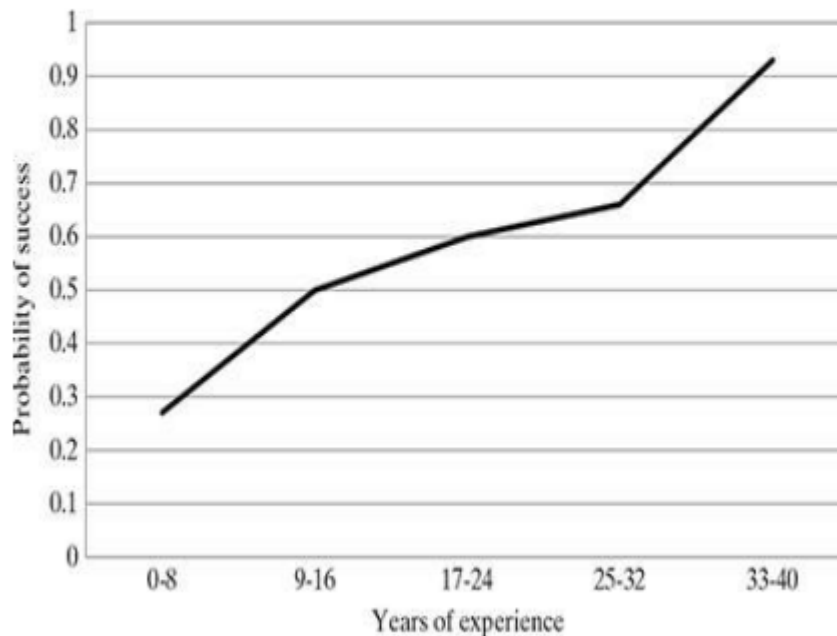
FIG. 8.18 Logistic regression

The 'ln' symbol refers to a natural logarithm and a + bX is the regression line equation. Probability (P) can also be computed from the regression equation. So, if we know the regression equation, we could, theoretically, calculate the expected probability that Y = 1 for a given value of X.

$$P = \frac{\exp(a + bX)}{1 + \exp(a + bx)} = \frac{e^{a+bx}}{1 + e^{a+bx}}$$

'exp' is the exponent function, which is sometimes also written as e.

Let us say we have a model that can predict whether a person is male or female on the basis of their height. Given a height of 150 cm, we need to predict whether the person is male or female.

We know that the coefficients of a = −100 and b = 0.6. Using the above equation, we can calculate the probability of male given a height of 150 cm or more formally P(male|height = 150).

$$y = e^{(a + b \times X)}/(1 + e^{(a + b \times X)})$$

$$y = \exp(-100 + 0.6 \times 150)/(1 + EXP(-100 + 0.6 \times X) \quad y = 0.000046$$

or a probability of near zero that the person is a male.

Assumptions in logistic regression

The following assumptions must hold when building a logistic regression model:

- There exists a linear relationship between logit function and independent variables

- The dependent variable Y must be categorical (1/0) and take binary value, e.g. if pass then Y = 1; else Y = 0

- The data meets the 'iid' criterion, i.e. the error terms, ε, are independent from one another and identically distributed

- The error term follows a binomial distribution [n, p]

  n = # of records in the data

  p = probability of success (pass, responder)

Maximum Likelihood Estimation

The coefficients in a logistic regression are estimated using a process called Maximum Likelihood Estimation (MLE). First, let us understand what is likelihood function before moving to MLE. A fair coin outcome flips equally heads and tails of the same number of times. If we toss the coin 10 times, it is expected that we get five times Head and five times Tail.

Let us now discuss about the probability of getting only Head as an outcome; it is 5/10 = 0.5 in the above case. Whenever this number (P) is greater than 0.5, it is said to be in favour of Head. Whenever P is lesser than 0.5, it is said to be against the outcome of getting Head.

Let us represent 'n' flips of coin as X1, X2, X3,…, Xn. Now Xi can take the value of 1 or 0.

Xi = 1 if Head is the outcome

Xi = 0 if Tail is the outcome

When we use the Bernoulli distribution represents each flip of the coin:

$$f(x_i|\theta) = \theta^{x_i}(1 - \theta)^{1-x_i}$$

Each observation X is independent and also identically distributed (iid), and the joint distribution simplifies to a product of distributions.

$$f(x_1, ..., x_n|\theta)\prod_{i=1}^{n}f(x_i|\theta) = \theta^{x_1}(1 - \theta)^{1-x_1}...\theta^{x_n}(1 - \theta)^{1-x_n} = \theta^{\#H}(1 - \theta)^{n-\#H},$$

where #H is the number of flips that resulted in the expected outcome (heads in this case).

The likelihood equation is

$$L(\theta|x) = \prod_{i=1}^{n}f(x_i|\theta)$$

But the likelihood function is not a probability. The likelihood for some coins may be 0.25 or 0 or 1. MLE is about predicting the value for the parameters that maximizes the likelihood function.

$$\log L(\theta|x) = \sum_{i=1}^{n} \log f(x_i|\theta)$$

SUMMARY

- In supervised learning, when we are trying to predict a real-value variable such as 'Price', 'Weight', etc., the problem falls under the category of regression. A regression problem tries to forecast results as a continuous output.

- Dependent Variable (Y) is the value to be predicted. This variable is presumed to be functionally related to the independent variable (X). In other words, dependent variable(s) depends on independent variable(s).

- Independent Variable (X) is called as predictor. The independent variable (X) is used in a regression model to estimate the value of the dependent variable (Y).

- Regression is essentially finding a relationship (or) association between the dependent variable (Y) and the independent variables (X).

- If the regression involves only one independent variable, it is called simple regression. Thus, if we take 'Price of a used car' as the i dependent variable and the 'Year of manufacturing of the car' as the independent variable, we can build a simple regression.

- Slope represents how much the line in a graph changes in the vertical direction (Y-axis) over a change in the horizontal direction (X-axis). Slope is also referred as the rate of change in a graph.

- Maximum and minimum points on a graph are found at points where the slope of the curve is zero. It becomes zero either from positive or from negative value.

- If two or more independent variables are involved, it is called multiple regression. Thus, if we take 'Price of a used car' as the dependent variable and year of manufacturing (Year), brand of the car (Brand), and mileage run (Miles run) as the independent variables, we can form a multiple regression equation as given below:

  Price of a used car ($) = function (Year, Brand, Miles run)

- Multicollinearity is the situation in which the degree of correlation is not only between the dependent variable and the independent variable, but there also exists a strong correlation within (among) the independent variables itself.

- Heteroskedasticity refers to the changing variance of the error term. If the variance of the error term is not constant across data sets, there will be erroneous predictions. In general, for a regression equation to make accurate predictions, the error term should be independent, normally (identically) distributed (iid). The error terms should not be related to each other.

- Accuracy of linear regression can be improved using the following three methods:

  1. Shrinkage Approach

  2. Subset Selection

  3. Dimensionality Reduction

- Polynomial regression model is the extension of the simple linear model by adding extra predictors, obtained by raising (squaring) each of the original predictors to a power. For example, if there are three variables, X, X , and X are used as predictors.

- In logistic regression, the dependent variable (Y) is binary (0,1) and independent variables (X) are continuous in nature. The probabilities describing the possible outcomes (probability that Y = 1) of a single trial are modelled as a function of the explanatory (predictor) variables by using a logistic function.

SAMPLE QUESTIONS

MULTIPLE-CHOICE QUESTIONS

1. When we are trying to predict a real-value variable such as '$', 'Weight', the problem falls under the category of

      1. Unsupervised learning

      2. Supervised regression problem

      3. Supervised classification problem

      4. Categorical attribute

2. Price prediction of crude oil is an example of

      1. Unsupervised learning

      2. Supervised regression problem

      3. Supervised classification problem

      4. Categorical attribute

3. Value to be predicted in machine learning is called as

      1. Slope

      2. Regression

      3. Independent variable

      4. Dependent variable

4. This is called as predictor.

      1. Slope

      2. Regression

      3. Independent variable

      4. Dependent variable

5. This is essentially finding a relationship (or) association between the dependent variable (Y) and the independent variables (X).

      1. Slope

      2. Regression

      3. Classification

      4. Categorization

6. If the regression involves only one independent variable, it is called as

    1. Multiple regression

    2. One regression

    3. Simple regression

    4. Independent regression

7. Which equation represents simple imperfect relationship?

    1. $Y = (a + bx) + \varepsilon$

    2. $Y = (a + bx)$

    3. DY = Change in Y / Change in X

    4. $Y = a + b1X1 + b2X2$

8. Which equation represents simple perfect relationship?

    1. $Y = (a + bx) + \varepsilon$

    2. $Y = (a + bx)$

    3. DY = Change in Y / Change in X

    4. $Y = a + b1X1 + b2X2$

9. What is the formula for slope in a simple linear equation?

    1. $Y = (a + bx) + \varepsilon$

    2. $Y = (a + bx)$

    3. DY = Change in Y / Change in X

    4. $Y = a + b1X1 + b2X2$

10. This slope always moves upward on a graph from left to right.

    1. Multilinear slope

    2. No relationship slope

    3. Negative slope

    4. Positive slope

11. This slope always moves downwards on a graph from left to right.

    1. Multilinear slope

    2. No relationship slope

    3. Negative slope

    4. Positive slope

12. Maximum and minimum points on a graph are found at points where the slope of the curve is

    1. Zero

2. One

3. 0.5

4. Random number

13. In the OLS algorithm, the first step is

    1. Calculate the mean of Y and X

    2. Calculate the errors of X and Y

    3. Get the product (multiply)

    4. Sum the products

14. In the OLS algorithm, the last step is

    1. Calculate 'a' using the value of 'b'

    2. Calculate 'b' using the value of 'a'

    3. Get the product (multiply)

    4. Sum the products

15. Which equation below is called as Unexplained Variation?

    1. SSR (Sum of Squares due to Regression)

    2. SSE (Sum of Squares due to Error)

    3. SST (Sum of Squares Total):

    4. R-square (R2)

16. Which equation below is called as Explained Variation?

    1. SSR (Sum of Squares due to Regression)

    2. SSE (Sum of Squares due to Error)

    3. SST (Sum of Squares Total):

    4. R-square (R2)

17. When new predictors (X) are added to the multiple linear regression model, how does R behave?

    1. Decreasing

    2. Increasing or decreasing

    3. Increasing and decreasing

    4. Increasing or remains constant

18. Predicting stochastic events precisely is not possible.

    1. True

    2. False

SHORT ANSWER-TYPE QUESTIONS

(5 MARKS EACH)

1. What is a dependent variable and an independent variable in a linear equation?

2. What is simple linear regression? Give one example.

3. Define slope in a linear regression.

4. Find the slope of the graph where the lower point on the line is represented as $(-3, -2)$ and the higher point on the line is represented as $(2, 2)$.

5. What are the conditions of a positive slope in linear regression?

6. What are the conditions of a negative slope in linear regression?

7. What is multiple linear regression?

8. Define sum of squares due to error in multiple linear regression.

9. Define sum of squares due to regression in multiple linear regression.

10. What is multicollinearity in regression equation?

11. What is heteroskedasticity?

12. Explain ridge regression.

13. Explain lasso regression.

14. What is polynomial regression?

15. Explain basis function.

16. Explain logistic regression.

LONG ANSWER-TYPE QUESTIONS

(10 MARKS EACH)

1. Define simple linear regression using a graph explaining slope and intercept.

2. Explain rise, run, and slope in a graph.

3. Explain slope, linear positive slope, and linear negative slope in a graph along with various conditions leading to the slope.

4. Explain curve linear negative slope and curve linear positive slope in a graph.

5. Explain maximum and minimum point of curves through a graph.

6. Explain ordinary least square with formula for a and b.

7. Explain the OLS algorithm with steps. 2

8. What is standard error of the regression? Draw a graph to represent the same.

9. Explain multiple linear regression with an example.

10. Explain the assumptions in regression analysis and BLUE concept.

11. Explain two main problems in regression analysis.

12. How to improve accuracy of the linear regression model?

13. Explain polynomial regression model in detail with an example.

14. Explain logistic regression in detail.

15. What are the assumptions in logistic regression?

16. Discuss maximum likelihood estimation in detail.

UNIT V

UNSUPERVISED LEARNING

Introduction, Unsupervised vs Supervised Learning, Application of Unsupervised Learning, Clustering – Clustering as a machine learning task, Different types of clustering techniques, Partitioning methods, K-Medoids: a representative object-based technique, Hierarchical clustering, Density-based methods-DBSCAN Finding Pattern using Association Rule Definition of common terms, Association rule, Theapriori algorithm for association rule learning, Build the apriori principle rules.

OBJECTIVE OF THE CHAPTER

We have discussed how to train our machines with past data on the basis of which they can learn, gain intelligence, and apply that intelligence on a new set of data. There are, however, situations when we do not have any prior knowledge of the data set we are working with, but we still want to discover interesting relationships among the attributes of the data or group the data in logical segments for easy analysis. The task of the machine is then to identify this knowledge without any prior training and that isthe space of unsupervised learning. In this chapter, we will discuss how Clustering algorithms help in grouping data sets into logical segments and the association analysis which enables to identify a pattern or relationship of attributes within the data set. An interesting application of the association analysis is the Market Basket Analysis, which is usedwidely by retailers and advertisers across the globe.

INTRODUCTION

Unsupervised learning is a machine learning concept where the unlabelled and unclassified information is analysed to discover hidden knowledge. The algorithms work on the data without any prior training, but they are constructed in such a way that they can identify patterns, groupings, sorting order, and numerous other interesting knowledge from the set of data.

UNSUPERVISED VS SUPERVISED LEARNING

Till now, we have discussed about supervised learning where the aim was to predict the outcome variable Y on the basis of the feature set $X : X : \dots X$, and we discussed methods such as regression and classification for the same. We will now introduce the concept of unsupervised learning where the objective is to observe only the features $X : X : \dots X$; we are not going to predict any outcome variable, but rather our intention is to find out the association between the features or their grouping to understand the nature of the data. This analysis may

reveal an interesting correlation between the features or a common behaviour within the subgroup of the data, which provides better understanding of the data.

In terms of statistics, a supervised learning algorithm will try to learn the probability of outcome Y for a particular input X, which is called the posterior probability. Unsupervised learning is closely related to density estimation in statistics. Here, every input and the corresponding targets are concatenated to create a new set of input such as {(X , Y), (X , Y),…, (X , Y)}, which leads to a better understanding of the correlation of X and Y; this probability notation is called the joint probability.

Let us take an example of how unsupervised learning helps in pushing movie promotions to the correct group of people. In earlier days, movie promotions were blind push of the same data to all demography, such that everyone used to watch the same posters or trailers irrespective of their choice or preference. So, in most of the cases, the person watching the promotion or trailer would end up ignoring it, which leads to waste of effort and money on the promotion. But with the advent of smart devices and apps, there is now a huge database available to understand what type of movie is liked by what segment of the demography. Machine learning helps to find out the pattern or the repeated behaviour of the smaller groups/clusters within this database to provide the intelligence about liking or disliking of certain types of movies by different groups within the demography. So, by using this intelligence, the smart apps can push only the relevant movie promotions or trailers to the selected groups, which will significantly increase the chance of targeting the right interested person for the movie.

We will discuss two methods in this chapter for explaining the principle underlying unsupervised learning – Clustering and Association Analysis. **Clustering** is a broad class of methods used for discovering unknown subgroups in data, which is the most important concept in unsupervised learning. Another technique is **Association Analysis** which identifies a low-dimensional representation of the observations that can explain the variance and identify the association rule for the explanation.

APPLICATION OF UNSUPERVISED LEARNING

Because of its flexibility that it can work on uncategorized and unlabelled data, there are many domains where unsupervised learning finds its application. Few examples of such applications are as follows:

- Segmentation of target consumer populations by an advertisement consulting agency on the basis of few dimensions such as demography, financial data, purchasing habits, etc. so that the advertisers can reach their target consumers efficiently
- Anomaly or fraud detection in the banking sector by identifying the pattern of loan defaulters
- Image processing and image segmentation such as face recognition, expression identification, etc.
- Grouping of important characteristics in genes to identify important influencers in new areas of genetics

- Utilization by data scientists to reduce the dimensionalities in sample data to simplify modelling
- Document clustering and identifying potential labelling options

Today, unsupervised learning is used in many areas involving Artificial Intelligence (AI) and Machine Learning (ML). Chat bots, self-driven cars, and many more recent innovations are results of the combination of unsupervised and supervised learning.

So, in this chapter, we will cover two major aspects of unsupervised learning, namely Clustering which helps in segmentation of the set of objects into groups of similar objects and Association Analysis which is related to the identification of relationships among objects in a data set.

CLUSTERING

Clustering refers to a broad set of techniques for finding subgroups, or clusters, in a data set on the basis of the characteristics of the objects within that data set in such a manner that the objects within the group are similar (or related to each other) but are different from (or unrelated to) the objects from the other groups. The effectiveness of clustering depends on how similar or related the objects within a group are or how different or unrelated the objects in different groups are from each other. It is often domain specific to define what is meant by two objects to be similar or dissimilar and thus is an important aspect of the unsupervised machine learning task.

As an example, suppose we want to run some advertisements of a new movie for a countrywide promotional activity. We have data for the age, location, financial condition, and political stability of the people in different parts of the country. We may want to run a different type of campaign for the different parts grouped according to the data we have. Any logical grouping obtained by analysing the characteristics of the people will help us in driving the campaigns in a more targeted way. Clustering analysis can help in this activity by analysing different ways to group the set of people and arriving at different types of clusters.

There are many different fields where cluster analysis is used effectively, such as

- Text data mining: this includes tasks such as text categorization, text clustering, document summarization, concept extraction, sentiment analysis, and entity relation modelling
- Customer segmentation: creating clusters of customers on the basis of parameters such as demographics, financial conditions, buying habits, etc., which can be used by retailers and advertisers to promote their products in the correct segment
- Anomaly checking: checking of anomalous behaviours such as fraudulent bank transaction, unauthorized computer intrusion, suspicious movements on a radar scanner, etc.
- Data mining: simplify the data mining task by grouping a large number of features from an extremely large data set to make the analysis manageable

In this section, we will discuss the methods related to the machine learning task of clustering, which involves finding natural groupings of data. The focus will be on

- how clustering tasks differ from classification tasks and how clustering defines groups
- a classic and easy-to-understand clustering algorithm, namely k means, which is used for clustering along with the k-medoids algorithm
- application of clustering in real-life scenarios

Clustering as a machine learning task

The primary driver of clustering knowledge is discovery rather than prediction, because we may not even know what we are looking for before starting the clustering analysis. So, clustering is defined as an unsupervised machine learning task that automatically divides the data into clusters or groups of similar items. The analysis achieves this without prior knowledge of the types of groups required and thus can provide an insight into the natural groupings within the data set. The primary guideline of clustering task is that the data inside a cluster should be very similar to each other but very different from those outside the cluster. We can assume that the definition of similarity might vary across applications, but the basic idea is always the same, that is, to create the group such that related elements are placed together. Using this principle, whenever a large set of diverse and varied data is presented for analysis, clustering enables to represent the data in a smaller number of groups. It helps to reduce the complexity and provides insight into patterns of relationships to generate meaningful and actionable structures within the data. The effectiveness of clustering is measured by the homogeneity within a group as well as the difference between distinct groups. See Figure 5.1 for reference.

From the above discussion, it may seem that through clustering, we are trying to label the objects with class labels. But clustering is somewhat different from the classification and numeric prediction discussed in supervised learning chapters. In each of these cases, the goal was to create a model that relates features to an outcome or to other features and the model identifies patterns within the data. In contrast, clustering creates new data. Unlabelled objects are given a cluster label which is inferred entirely from the relationship of attributes within the data.

Let us take an example. You were invited to take a session on Machine Learning in a reputed university for induction of their professors on the subject. Before you create the material for the session, you want to know the level of acquaintance of the professors on the subject so that the session is successful. But you do not want to ask the inviting university, but rather do some analysis on your own on the basis of the data available freely. As Machine Learning is the intersection of Statistics and Computer Science, you focused on identifying the professors from these two areas also. So, you searched the list of research publications of these professors from the internet, and by using the machine learning algorithm, you now want to group the papers and thus infer the expertise of the professors into three buckets – Statistics, Computer Science, and Machine Learning.

After plotting the number of publications of these professors in the two core areas, namely Statistics and Computer Science, you obtain a scatter plot as shown in Figure 5.2.
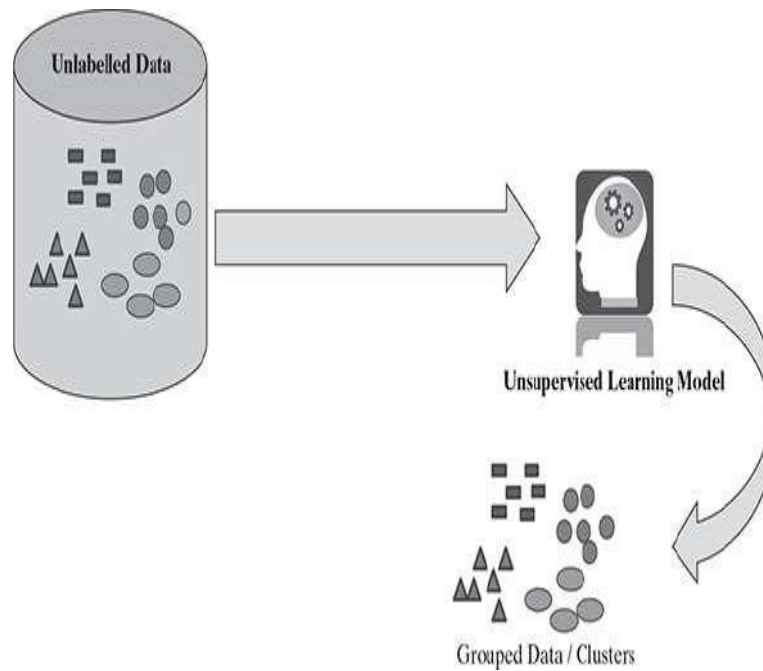
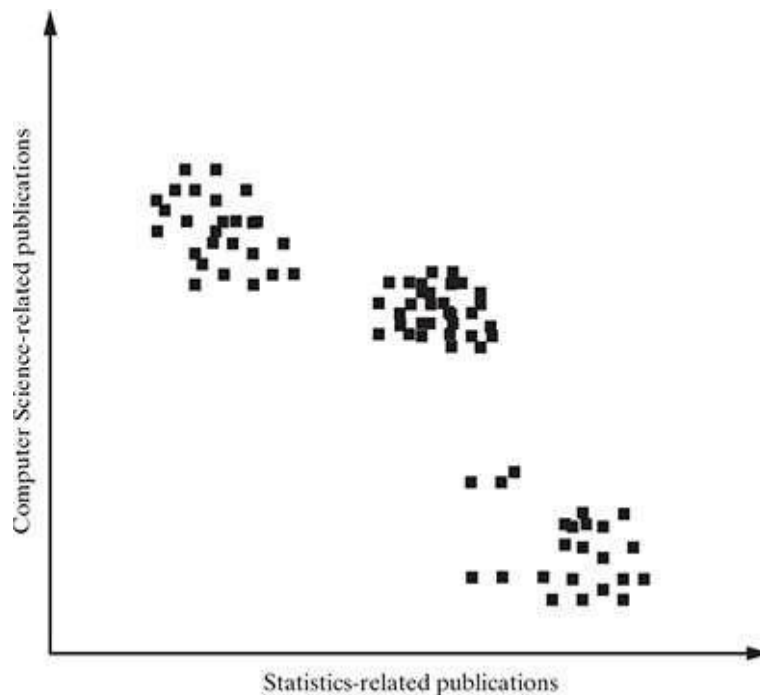**FIG. 5.1** Unsupervised learning – clustering

FIG. 5.2 Data set for the conference attendees

Some inferences that can be derived from the pattern analysis of the data is that there seems to be three groups or clusters emerging from the data. The pure statisticians have very less Computer Science-related papers, whereas the pure Computer Science professors have less number of statistics-related papers than Computer Science related papers. There is a third cluster of professors who have published papers on both these areas and thus can be assumed to be the persons knowledgeable in machine learning concepts, as shown in Figure 5.3.

Thus, in the above problem, we used visual indication of logical grouping of data to identify a pattern or cluster and labelled the data in three different clusters. The main driver for our clustering was the closeness of the points to each other to form a group. The clustering algorithm uses a very similar approach to measure how closely the data points are related and decides whether they can be labelled as a homogeneous group. In the next section, we will discuss few important algorithms for clustering.
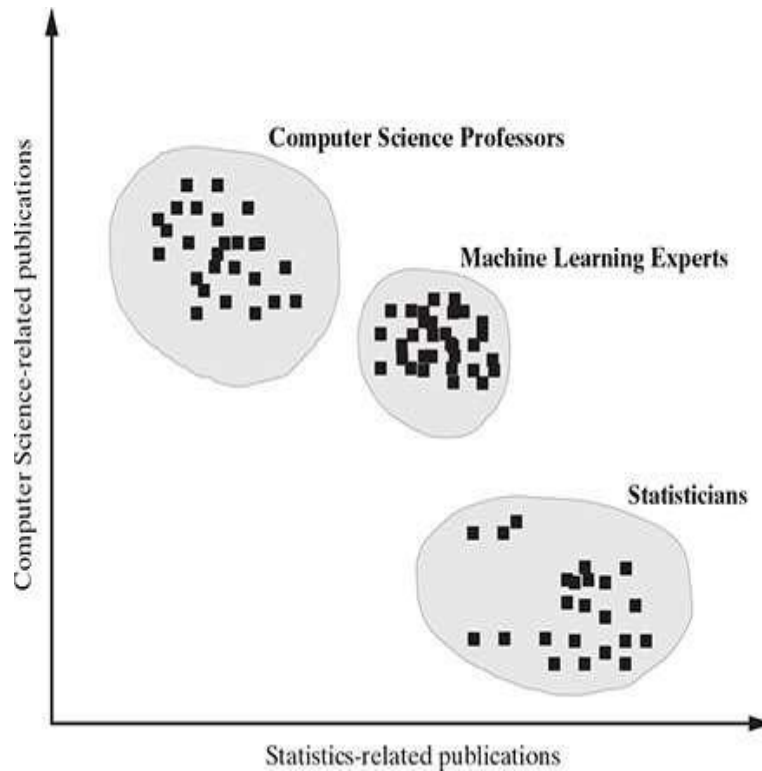
FIG. 5.3 Clusters for the conference attendees

DIFFERENT TYPES OF CLUSTERING TECHNIQUES

The major clustering techniques are

- Partitioning methods,
- Hierarchical methods, and
- Density-based methods.

Their approach towards creating the clusters, way to measure the quality of the clusters, and applicability are different. Table 5.1 summarizes the main characteristics of each method for each reference.

Method	Characteristics
Partitioning methods	• Uses mean or medoid (etc.) to represent cluster centre • Adopts distance-based approach to refine clusters • Finds mutually exclusive clusters of spherical or nearly spherical shape • Effective for data sets of small to medium size
Hierarchical methods	• Creates hierarchical or tree-like structure through decomposition or merger • Uses distance between the nearest or furthest points in neighbouring clusters as a guideline for refinement • Erroneous merges or splits cannot be corrected at subsequent levels
Density-based methods	• Useful for identifying arbitrarily shaped clusters • Guiding principle of cluster creation is the identification of dense regions of objects in space which are separated by low-density regions • May filter out outliers

Table 5.1 Different Clustering Methods

We will discuss each of these methods and their related techniques in details in the following sections.

PARTITIONING METHODS

Two of the most important algorithms for partitioning-based clustering are k-means and k-medoid. In the k means algorithm, the centroid of the prototype is identified for clustering, which is normally the mean of a group of points. Similarly, the k-medoid algorithm identifies the medoid which is the most representative point for a group of points. We can also infer that in most cases, the centroid does not correspond to an actual data point, whereas medoid is always an actual data point. Let us discuss both these algorithms in detail.

K-means - A centroid-based technique

This is one of the oldest and most popularly used algorithm for clustering. The basic principles used by this algorithm also serves as the basis for other more sophisticated and complex algorithms. Table 5.2 provides the strengths and weaknesses of this algorithm.

The principle of the k-means algorithm is to assign each of the 'n' data points to one of the K clusters where 'K' is a user-defined parameter as the number of clusters desired. The objective is to maximize the homogeneity within the clusters and also to maximize the differences between the clusters. The homogeneity and differences are measured in terms of the distance between the objects or points in the data set.

Step 1: Select *K* points in the data space and mark them as initial centroids

loop

Step 2: Assign each point in the data space to the nearest centroid to form K clusters

Step 3: Measure the distance of each point in the cluster from the centroid

Step 4: Calculate the Sum of Squared Error (SSE) to measure the quality of the clusters *(described later in this chapter)*

Step 5: Identify the new centroid of each cluster on the basis of distance between points

Step 6: Repeat Steps 2 to 5 to refine until centroids do not change

end loop

Algorithm 5.1 shows the simple algorithm of *K*-means

Strengths	Weaknesses
• The principle used for identifying the clusters is very simple and involves very less complexity of statistical terms • The algorithm is very flexible and thus can be adjusted for most scenarios and complexities • The performance and efficiency are very high and comparable to those of any sophisticated algorithm in term of dividing the data into useful clusters	• The algorithm involves an element of random chance and thus may not find the optimal set of cluster in some cases • The starting point of guessing the number natural clusters within the data requires some experience of the user, so that the final outcome is efficient

Table 5.2 Strengths and Weaknesses of K-means

Let us understand this algorithm with an example. In Figure 5.3, we have certain set of data points, and we will apply the k-means algorithm to find out the clusters generated from this data set. Let us fix K = 4, implying that we want to create four clusters out of this data set. As the first step, we assign four random points from the data set as the centroids, as represented by the * signs, and we assign the data points to the nearest centroid to create four clusters. In the second step, on the basis of the distance of the points from the corresponding centroids, the centroids are updated and points are reassigned to the updated centroids. After three iterations, we found that the centroids are not moving as there is no scope for refinement, and thus, the k-means algorithm will terminate. This provides us the most logical four groupings or cluster of the data sets where the homogeneity within the groups is highest and difference between the groups is maximum.
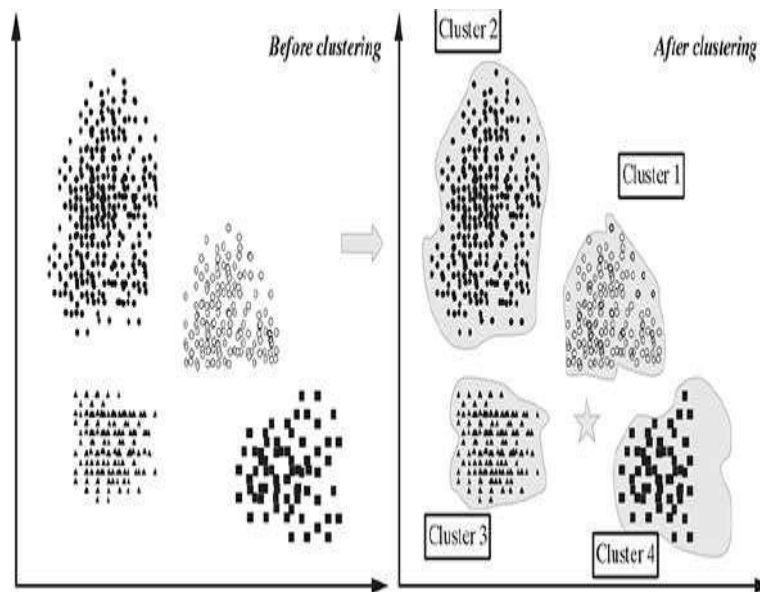
FIG. 5.3 Clustering concept – before and after clustering

Choosing appropriate number of clusters

One of the most important success factors in arriving at correct clustering is to start with the correct number of cluster assumptions. Different numbers of starting cluster lead to completely different types of data split. It will always help if we have some prior knowledge about the number of clusters and we start our k-means algorithm with that prior knowledge. For example, if we are clustering the data of the students of a university, it is always better to start with the number of departments in that university. Sometimes, the business needs or resource limitations drive the number of required clusters. For example, if a movie maker wants to cluster the movies on the basis of combination of two parameters – budget of the movie: high or low, and casting of the movie: star or non-star, then there are 4 possible combinations, and thus, there can be four clusters to split the data.

For a small data set, sometimes a rule of thumb that is followed is

$$K = \sqrt{\frac{n}{2}}$$

which means that K is set as the square root of n/2 for a data set of n examples. But unfortunately, this thumb rule does not work well for large data sets. There are several statistical methods to arrive at the suitable number of clusters.

Elbow method

This method tries to measure the homogeneity or heterogeneity within the cluster and for various values of 'K' and helps in arriving at the optimal 'K'. From Figure 5.4, we can see the homogeneity will increase or heterogeneity will decrease with increasing 'K' as the number of data points inside each cluster reduces with this increase. But these iterations take significant computation effort, and after a certain point, the increase in homogeneity benefit is no longer in accordance with the investment required to achieve it, as is evident from the figure. This point is known as the elbow point, and the 'K' value at this point produces the optimal

clustering performance. There are a large number of algorithms to calculate the homogeneity and heterogeneity of the clusters, which are not discussed in this book.

Choosing the initial centroids

Another key step for the k-means algorithm is to choose the initial centroids properly. One common practice is to choose random points in the data space on the basis of the number of cluster requirement and refine the points as we move into the iterations. But this often leads to higher squared error in the final clustering, thus resulting in sub-optimal clustering solution. The assumption for selecting random centroids is that multiple subsequent runs will minimize the SSE and identify the optimal clusters. But this is often not true on the basis of the spread of the data set and the number of clusters sought. So, one effective approach is to employ the hierarchical clustering technique on sample points from the data set and then arrive at sample K clusters. The centroids of these initial K clusters are used as the initial centroids. This approach is practical when the data set has small number of points and K is relatively small compared to the data points. There are procedures such as bisecting k-means and use of post-processing to fix initial clustering issues; these procedures can produce better quality initial centroids and thus better SSE for the final clusters.
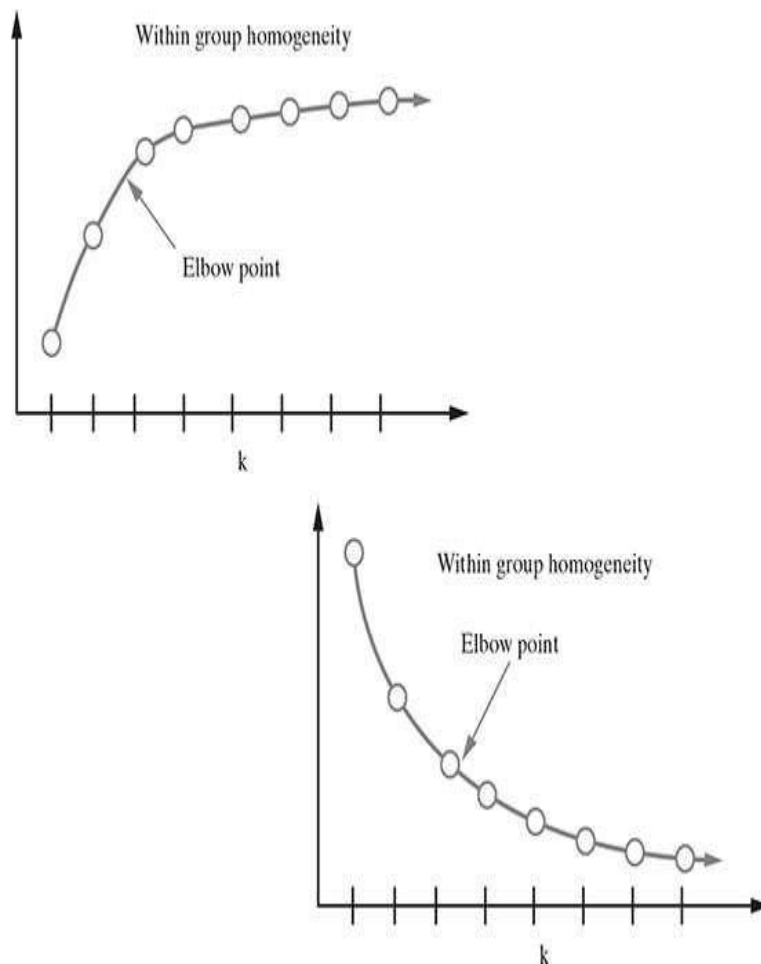
FIG. 5.4 Elbow point to determine the appropriate number of clusters

Recomputing cluster centroids

We discussed in the k-means algorithm that the iterative step is to recalculate the centroids of the data set after each iteration. The proximities of the data points from each other within a cluster is measured to minimize the distances. The distance of the data point from its nearest centroid can also be calculated to minimize the distances to arrive at the refined centroid. The Euclidean distance between two data points is measured as follows:

$$\text{dist}(x, y) = \sqrt{\sum_{1}^{n} (x_i - y_i)^2} \qquad (9.1)$$

Using this function, the distance between the example data and its nearest centroid and the objective is calculated to minimize this distance. The measure of quality of clustering uses the SSE technique. The formula used is as follows:

$$\text{SSE} = \sum_{i=1}^{k} \sum_{x \in C_i} \text{dist}(c_i, x)^2 \qquad (9.2)$$

where dist() calculates the Euclidean distance between the centroid c of the cluster C and the data points x in the cluster. The summation of such distances over all the 'K' clusters gives the total sum of squared error. As you can understand, the lower the SSE for a clustering solution, the better is the representative position of the centroid. Thus, in our clustering algorithm in Algorithm 9.1, the re-computation of the centroid involves calculating the SSE of each new centroid and arriving at the optimal centroid identification. After the centroids are repositioned, the data points nearest to the centroids are assigned to form the refined clusters. It is observed that the centroid that minimizes the SSE of the cluster is its mean. One limitation of the squared error method is that in the case of presence of outliers in the data set, the squared error can distort the mean value of the clusters.

Let us use this understanding to identify the cluster step for the data set in Figure 5.5. Assume that the number of cluster requirement, K = 4. We will randomly select four cluster centroids as indicated by four different colours in Figure 5.6.

Now, on the basis of the proximity of the data points in this data set to the centroids, we partition the data set into four segments as represented by dashed lines in Figure 5.7. This diagram is called **Voronoi diagram** which creates the boundaries of the clusters. We got the initial four clusters, namely C 1 , C 2 , C 3 , and C 4 , created by the dashed lines from the vertex of the clusters, which is the point with the maximal distance from the centre of the clusters. It is now easy to understand the areas covered by each cluster and the data points within each cluster through this representation.
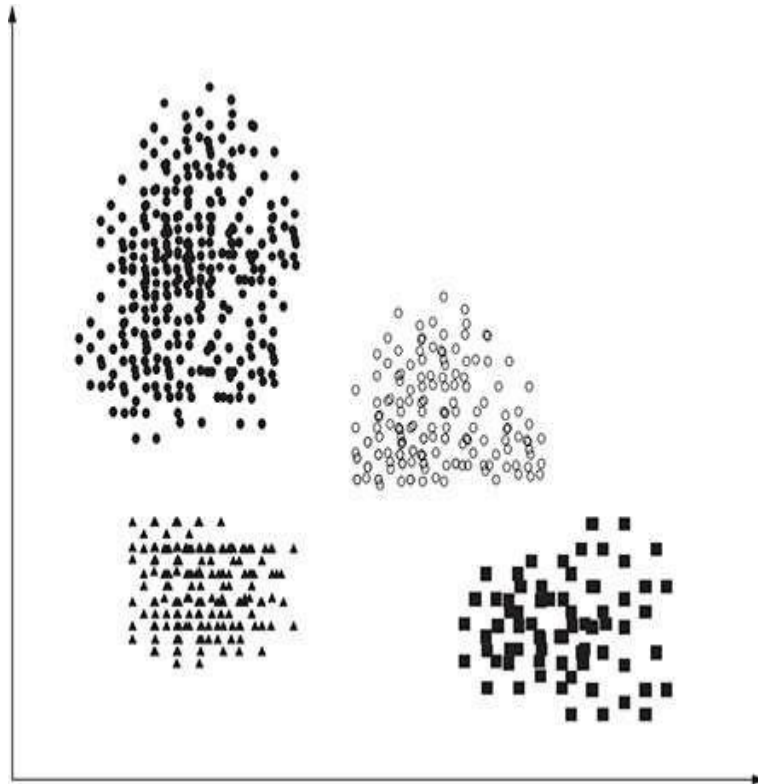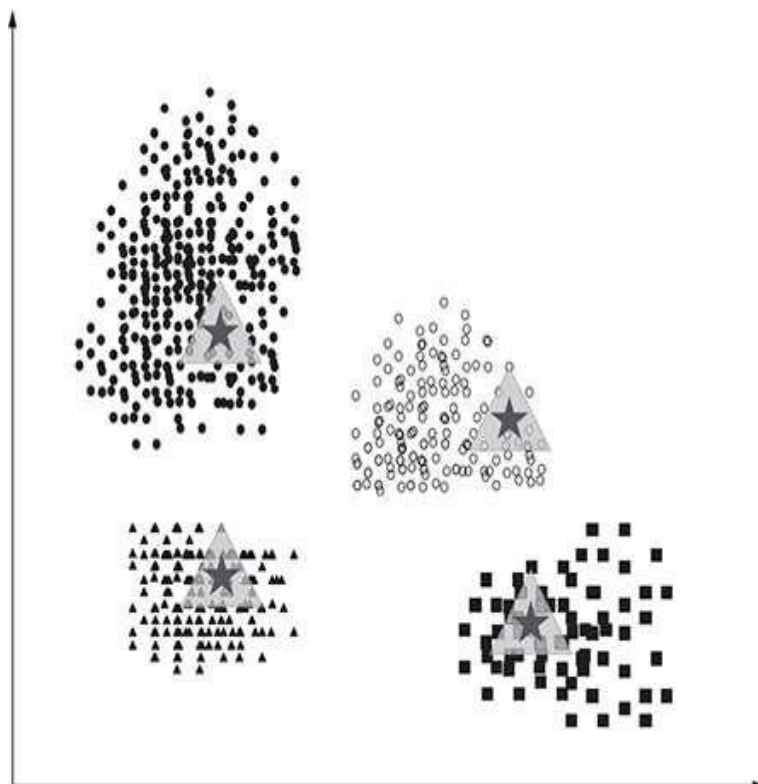
FIG. 5.5 Clustering of data set

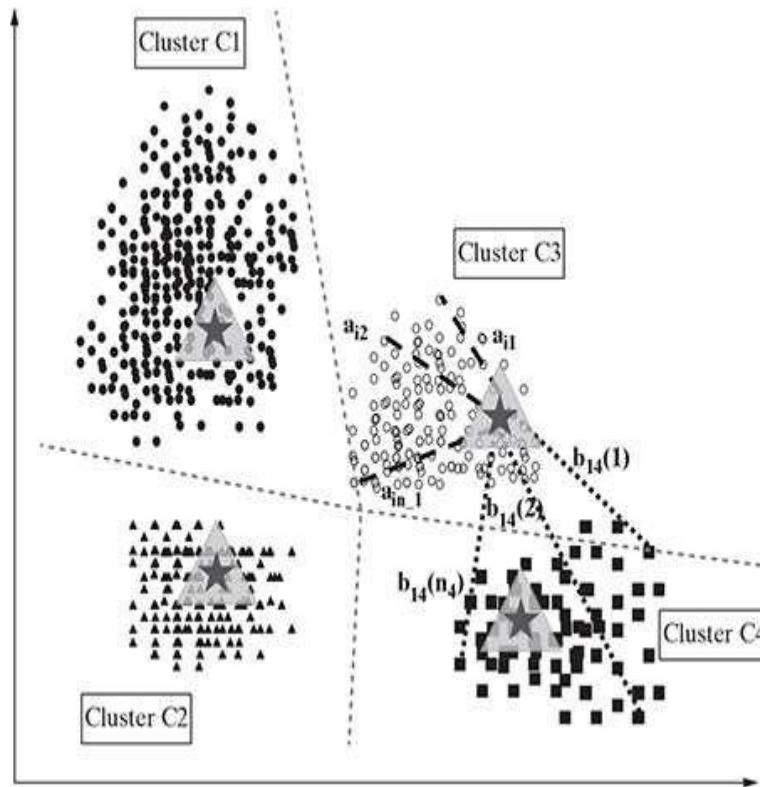**FIG. 5.6** Clustering with initial centroids

FIG. 5.7 Iteration 1: Four clusters and distance of points from the centroids

The next step is to calculate the SSE of this clustering and update the position of the centroids. We can also proceed by our understanding that the new centroid should be the mean of the data points in the respective clusters. The distances of the data points currently marked as Cluster C from the centroid of cluster C are marked as a , a ,…, a in the figure and those determine the homogeneity within cluster C . On the other hand, the distances of data points of cluster C from the centroid of cluster C determine the heterogeneity among these two different clusters. Our aim is to minimize the homogeneity within the clusters and maximize the heterogeneity among the different clusters. So, the revised centroids are as shown in Figure 5.8.

We can also find out that the cluster boundaries are refined on the basis of the new centroids and the identification of the nearest centroids for the data points and reassigning them to the new centroids. The new points reclaimed by each cluster are shown in the diagram.

The k-means algorithm continues with the update of the centroid according to the new cluster and reassignment of the points, until no more data points are changed due to the centroid shift. At this point, the algorithm stops. Figure 5.9 shows the final clustering of the data set we used. The complexity of the k-means algorithm is O (nKt), where ' n ' is the total number of data points or objects in the data set, K is the number of clusters, and ' t ' is the number of iterations. Normally, ' K ' and ' t ' are kept much smaller than ' n ', and thus, the k-means method is relatively scalable and efficient in processing large data sets.
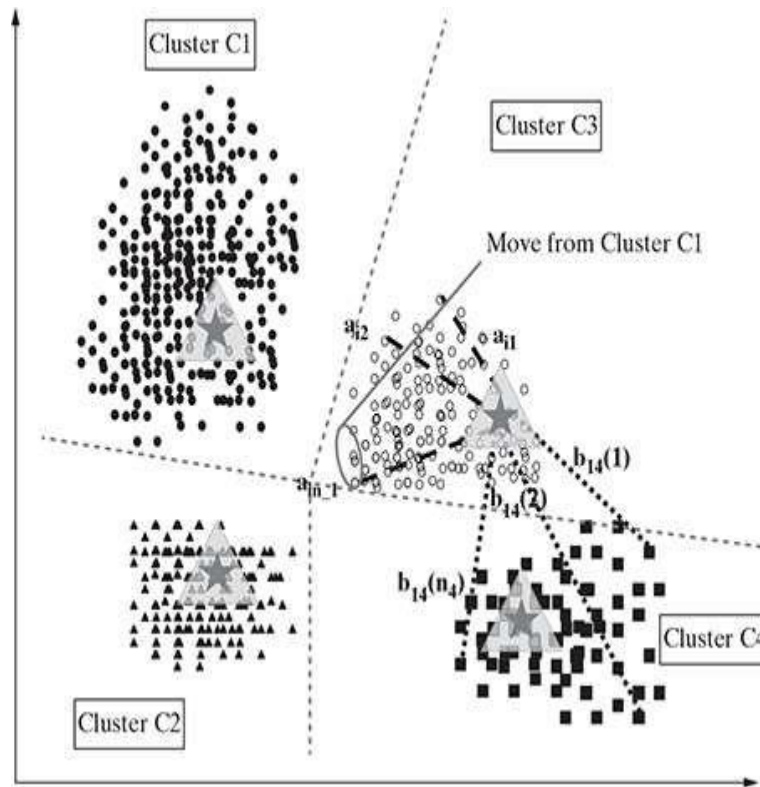
FIG. 5.8 Iteration 2: Centroids recomputed and points redistributed among the clusters according to the nearest centroid
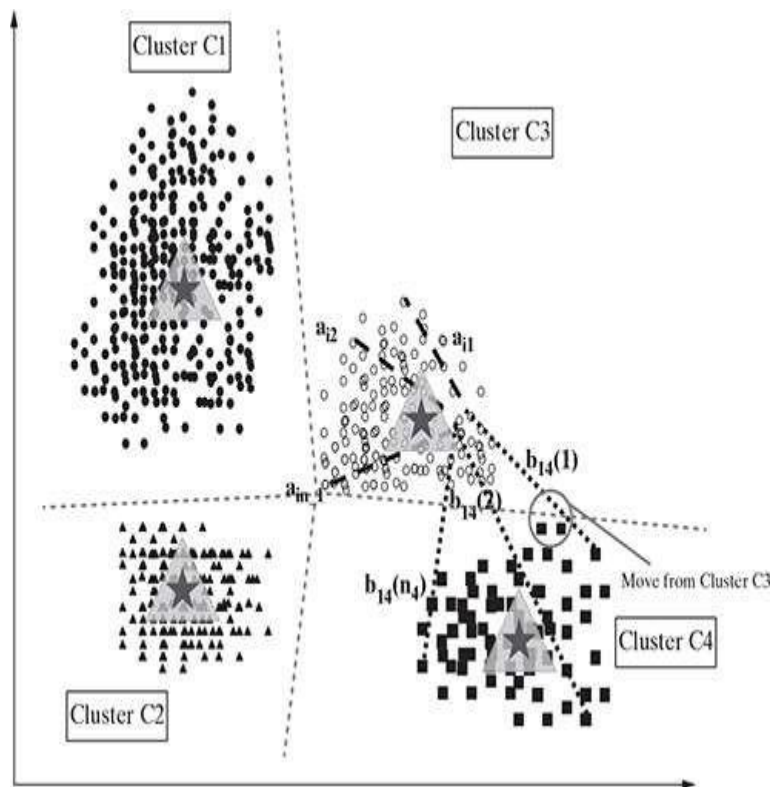

FIG. 5.9 Iteration 3: Final cluster arrangement: Centroids recomputed and points redistributed among the clusters according to the nearest centroid

k-means often produce local optimum and not global optimum. Also, the result depends on the initial selection of random cluster centroids. It is a common practice to run the k-means algorithm multiple times with different cluster centres to identify the optimal clusters. The necessity to set the initial 'K' values is also perceived as a disadvantage of the k-means algorithm. There are methods to overcome this problem, such as defining a range for 'K' and then comparing the results of clustering with those different 'K' values to arrive at the best possible cluster. The ways to improve cluster performance is an area of further study, and many different techniques are employed to achieve that. This is out of scope of this book but can be pursued for advanced machine learning studies.

NOTE

Clustering is often used as the first step of identifying the subgroups within a unlabeled set of data which then is used for classifying the new observed data. At the beginning we are not clear about the pattern or classes that exist within the unlabeled data set. By using the clustering algorithm at that stage we find out the groups of similar objects within the data set and form sub-groups and classes. Later when a new object is observed, then using the classification algorithms we try to place that into one of the sub-groups identified in the earlier stage. Let's take an example. We are running a software testing activity and we identified a set of defects in the software. For easy allocation of these defects to different developer groups, the team is trying to identify similar groups of defects. Often text analytics is used as the guiding principle for identifying this similarity. Suppose there are 4 sub groups of defects identified, namely, GUI related defects, Business logic related defects, Missing requirement defects and Database related defects. Based on this grouping, the team identified the developers to whom the defects should be sent for fixing. As the testing continues, there are new defects getting created. We have the categories of defects identified now and thus the team can use classification algorithms to assign the new defect to one of the 4 identified groups or classes which will make it easy to identify the developer who should be fixing it.

K-MEDOIDS: A REPRESENTATIVE OBJECT-BASED TECHNIQUE

As discussed earlier, the k-means algorithm is sensitive to outliers in the data set and inadvertently produces skewed clusters when the means of the data points are used as centroids. Let us take an example of eight data points, and for simplicity, we can consider them to be 1-D data with values 1, 2, 3, 5, 9, 10, 11, and 25. Point 25 is the outlier, and it affects the cluster formation negatively when the mean of the points is considered as centroids.

With K = 2, the initial clusters we arrived at are {1, 2, 3, 6} and {9, 10, 11, 25}.

The mean of the cluster $\{1, 2, 3, 6\} = \dfrac{12}{4} = 3,$

and the mean of the cluster

$\{9, 10, 12, 25\} = \dfrac{56}{4} = 14.$

So, the SSE within the clusters is

$$(1 - 3)^2 + (2 - 3)^2 + (3 - 3)^2 + (6 - 3)^2 + (9 - 14)^2$$
$$+ (10 - 14)^2 + (12 - 14)^2 + (25 - 14)^2 = 179$$

If we compare this with the cluster {1, 2, 3, 6, 9} and {10, 11, 25},

the mean of the cluster $\{1, 2, 3, 6, 9\} = \dfrac{21}{5} = 4.2,$

and the mean of the cluster

$\{10, 12, 25\} = \dfrac{47}{3} = 15.67.$

So, the SSE within the clusters is

$$(1 - 4.2)^2 + (2 - 4.2)^2 + (3 - 4.2)^2 + (6 - 4.2)^2 + (9 - 4.2)^2$$
$$+ (10 - 15.67)^2 + (12 - 15.67)^2 + (25 - 15.67)^2 = 113.84$$

Because the SSE of the second clustering is lower, k   means tend to put point 9 in the same cluster with 1, 2, 3, and 6 though the point is logically nearer to points 10 and 11. This skewedness is introduced due to the outlier point 25, which shifts the mean away from the centre of the cluster.

k-medoids provides a solution to this problem. Instead of considering the mean of the data points in the cluster, k-medoids considers k representative data points from the existing points in the data set as the centre of the clusters. It then assigns the data points according to their distance from these centres to form k clusters. Note that the medoids in this case are actual data points or objects from the data set and not an imaginary point as in the case when the mean of the data sets within cluster is used as the centroid in the k-means technique. The SSE is calculated as

$$SSE = \sum_{i=1}^{k} \sum_{x \in C_i} dist\,(o_i, x)^2 \qquad (9.3)$$

where o is the representative point or object of cluster C .

Thus, the k-medoids method groups n objects in k clusters by minimizing the SSE. Because of the use of medoids from the actual representative data points, k medoids is less influenced by the outliers in the data. One of the practical implementation of the k-medoids principle is the Partitioning Around Medoids (PAM) algorithm. Refer to Algorithm 2 table:

Algorithm 2: PAM

Step 1: Randomly choose k points in the data set as the initial representative points

loop

Step 2: Assign each of the remaining points to the cluster which has the nearest representative point

Step 3: Randomly select a non-representative point o_r in each cluster

Step 4: Swap the representative point o_j with o_r and compute the new SSE after swapping

Step 5: If SSE new < SSE old, then swap o_j with o_r to form the new set of k representative objects;

Step 6: Refine the k clusters on the basis of the nearest representative point. Logic continues until there is no change

end loop

In this algorithm, we replaced the current representative object with a non-representative object and checked if it improves the quality of clustering. In the iterative process, all possible replacements are attempted until the quality of clusters no longer improves. i i r j r j r If o ,…, o are the current set of representative objects or medoids and there is a non-representative object o , then to determine whether o is a good replacement of o ($1 \leq j \leq k$), the distance of each object x is calculated from its nearest medoid from the set {o , o ,…, o , o , o ,…, o } and the SSE is calculated. If the SSE after replacing o with o decreases, it means that o represents

the cluster better than o , and the data points in the set are reassigned according to the nearest medoids now.
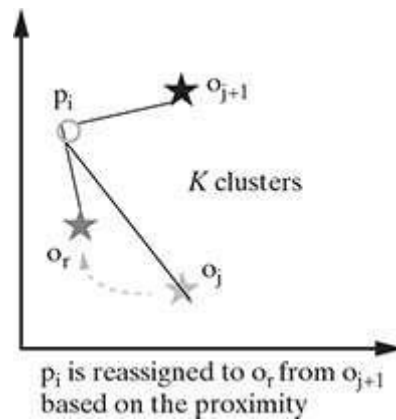

FIG. 5.10 PAM algorithm: Reassignment of points to different clusters

As shown in Figure 5.10, point p was belonging to the cluster with medoid o in the first iteration, but after o was replaced by o , it was found that p is nearest to the new random medoid and thus gets assigned to it. In this way, the clusters get refined after each medoid is replaced with a new non-representative medoid. Each time a reassignment is done, the SSE based on the new medoid is calculated. The difference between the SSE before and after the swap indicates whether or not the replacement is improving the quality of the clustering by bringing the most similar points together.

Points to Ponder:

k-medoids methods like PAM works well for small set of data, but they are not scalable for large set of data because of computational overhead. A sample based technique is used in the case of large data set where the sample should be a good representative of the whole data set.

Though the k-medoids algorithm provides an effective way to eliminate the noise or outliers in the data set, which was the problem in the k-means algorithm, it is expensive in terms of calculations. The complexity of each iteration in the k-medoids algorithm is O(k(n - k)). For large value of 'n' and 'k', this calculation becomes much costlier than that of the k-means algorithm.

HIERARCHICAL CLUSTERING

Till now, we have discussed the various methods for partitioning the data into different clusters. But there are situations when the data needs to be partitioned into groups at different levels such as in a hierarchy. The hierarchical clustering methods are used to group the data into hierarchy or tree-like structure. For example, in a machine learning problem of organizing employees of a university in different departments, first the employees are grouped under the different departments in the university, and then within each department, the employees can be grouped according to their roles such as professors, assistant professors, supervisors, lab assistants, etc. This creates a hierarchical structure of the employee data and eases visualization and analysis. Similarly, there may be a data set which has an underlying hierarchy structure that we want to discover and we can use the hierarchical clustering methods to achieve that.

There are two main hierarchical clustering methods: agglomerative clustering and divisive clustering. Agglomerative clustering is a bottom-up technique which starts with individual objects as clusters and then iteratively merges them to form larger clusters. On the other hand, the divisive method starts with one cluster with all given objects and then splits it iteratively to form smaller clusters. See Figure 5.11.

The agglomerative hierarchical clustering method uses the bottom-up strategy. It starts with each object forming its own cluster and then iteratively merges the clusters 2 according to their similarity to form larger clusters. It terminates either when a certain clustering condition imposed by the user is achieved or all the clusters merge into a single cluster.

The divisive hierarchical clustering method uses a top-down strategy. The starting point is the largest cluster with all the objects in it, and then, it is split recursively to form smaller and smaller clusters, thus forming the hierarchy. The end of iterations is achieved when the objects in the final clusters are sufficiently homogeneous to each other or the final clusters contain only one object or the user-defined clustering condition is achieved.

In both these cases, it is important to select the split and merger points carefully, because the subsequent splits or mergers will use the result of the previous ones and there is no option to perform any object swapping between the clusters or rectify the decisions made in previous steps, which may result in poor clustering quality at the end.

A dendrogram is a commonly used tree structure representation of step-by-step creation of hierarchical clustering. It shows how the clusters are **merged** iteratively (in the case of agglomerative clustering) or **split** iteratively (in the case of divisive clustering) to arrive at the optimal clustering solution. Figure 5.12 shows a dendro-gram with four levels and how the objects are merged or split at each level to arrive at the hierarchical clustering.
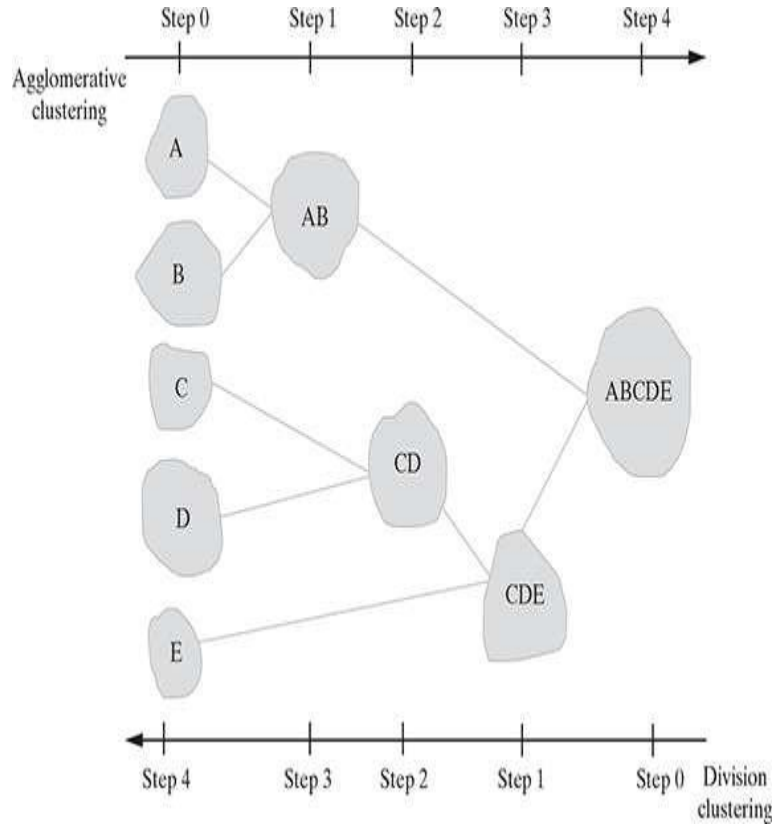
FIG. 5.11 Agglomerative and divisive hierarchical clustering

One of the core measures of proximities between clusters is the distance between them. There are four standard methods to measure the distance between clusters: Let C and C be the two clusters with n and n respectively. p and p represents the points in clusters C and C respectively. We will denote the mean of cluster C as m i.

Minimum distance $D_{\min}(C_i, C_j) = \min\limits_{p_i \in C_i, \, p_j \in C_j} \{|p_i - p_j|\}$ (9.4)

Maximum distance $D_{\max}(C_i, C_j) = \max\limits_{p_i \in C_i, \, p_j \in C_j} \{|p_i - p_j|\}$ (9.5)

Mean distance $D_{\mathrm{mean}}(C_i, C_j) = \{|m_i - m_j|\}$ (9.6)

Average distance $D_{\mathrm{avg}}(C_i, C_j) = \dfrac{1}{n_i n_j} \sum\limits_{p_i \in C_i, \, p_j \in C_j} |pi - pj|$ (9.7)

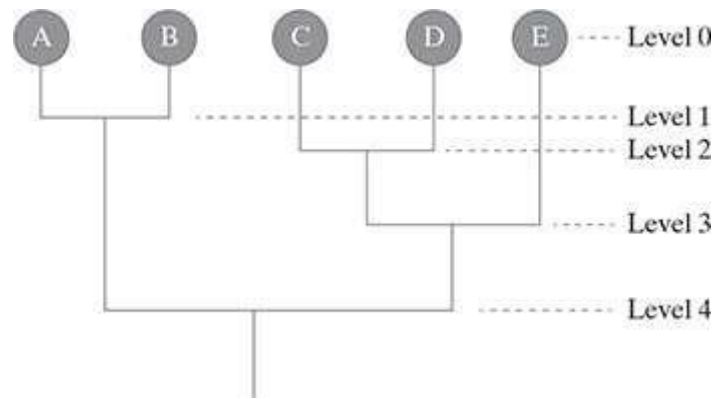Refer to Figure 5.13 for understanding the concept of these distances

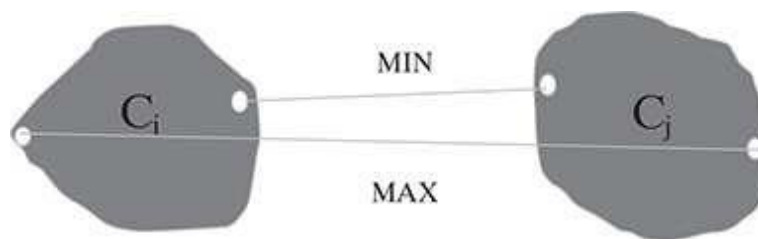FIG. 5.12 Dendrogram representation of hierarchical clustering

FIG. 5.13 Distance measure in algorithmic methods

Often the distance measure is used to decide when to terminate the clustering algorithm. For example, in an agglomerative clustering, the merging iterations may be stopped once the MIN distance between two neighbouring clusters becomes less than the user-defined threshold. So, when an algorithm uses the minimum distance D to measure the distance between the clusters, then it is referred to as nearest neighbour min clustering algorithm, and if the decision to stop the algorithm is based on a user-defined limit on D , then it is called single linkage algorithm.

On the other hand, when an algorithm uses the maximum distance D to measure the distance between the clusters, then it is referred to as furthest neighbour clustering algorithm, and if the decision to stop the algorithm is based on a user-defined limit on D then it is called complete linkage algorithm.

As minimum and maximum measures provide two extreme options to measure distance between the clusters, they are prone to the outliers and noisy data. Instead, the use of mean and average distance helps in avoiding such problem and provides more consistent results.

DENSITY-BASED METHODS – DBSCAN

You might have noticed that when we used the partitioning and hierarchical clustering methods, the resulting clusters are spherical or nearly spherical in nature. In the case of the other shaped clusters such as S shaped or uneven shaped clusters, the above two types of method do not provide accurate results. The density based clustering approach provides a solution to identify clusters of arbitrary shapes. The principle is based on identifying the dense area and sparse area within the data set and then run the clustering algorithm. DBSCAN is one

of the popular density-based algorithm which creates clusters by using connected regions with high density.

FINDING PATTERN USING ASSOCIATION RULE

Association rule presents a methodology that is useful for identifying interesting relationships hidden in large data sets. It is also known as **association analysis**, and the discovered relationships can be represented in the form of association rules comprising a set of frequent items. A common application of this analysis is the **Market Basket Analysis** that retailers use for cross-selling of their products. For example, every large grocery store accumulates a large volume of data about the buying pattern of the customers. On the basis of the items min max max purchased together, the retailers can push some cross-selling either by placing the items bought together in adjacent areas or creating some combo offer with those different product types. The below association rule signifies that people who have bought bread and milk have often bought egg also; so, for the retailer, it makes sense that these items are placed together for new opportunities for cross-selling.

$$\{Bread, Milk\} \rightarrow \{Egg\}$$

The application of association analysis is also widespread in other domains such as bioinformatics, medical diagnosis, scientific data analysis, and web data mining. For example, by discovering the interesting relationship between food habit and patients developing breast cancer, a new cancer prevention mechanism can be found which will benefit thousands of people in the world. In this book, we will mainly illustrate the analysis techniques by using the market basket example, but it can be used more widely across domains to identify association among items in transactional data. The huge pool of data generated everyday through tracked transactions such as barcode scanner, online purchase, and inventory tracking systems has enabled for machine learning systems to learn from this wealth of data. We will discuss the methods for finding useful associations in large databases by using simple statistical performance measures while managing the peculiarities of working with such transactional data. One significant challenge in working with the large volume of data is that it may be computationally very expensive to discover patterns from such data. Moreover, there may be cases when some of the associations occurred by chance, which can lead to potentially false knowledge. While discussing the association analysis, we will discuss both these points.

We will use the transaction data in the table below for our examples of association analysis. This simplified version of the market basket data will show how the association rules can be effectively used for the market basket analysis.

DEFINITION OF COMMON TERMS

Let us understand few common terminologies used in association analysis.

Itemset

One or more items are grouped together and are surrounded by brackets to indicate that they form a set, or more specifically, an **itemset** that appears in the data with some regularity. For example, in Table 5.3, {Bread, Milk, Egg} can be grouped together to form an itemset as those are frequently bought together. To generalize this concept, if $I = \{i, i, \ldots, i\}$ are the items in a

market basket data and T = {t , t ,…, t } are the set of all the transactions, then each transaction t contains a subset of items from I. A collection of zero or more items is called an itemset. A null itemset is the one which does not contain any item. In the association analysis, an itemset is called k-itemset if it contains k number of items. Thus, the itemset {Bread, Milk, Egg} is a three-itemset.

Transaction Number	Purchased Items
1	{Bread, Milk, Egg, Butter, Salt, Apple}
2	{Bread, Milk, Egg, Apple}
3	{Bread, Milk, Butter, Apple}
4	{Milk, Egg, Butter, Apple}
5	{Bread, Egg, Salt}
6	{Bread, Milk, Egg, Apple}

Table 5.3 Market Basket Transaction Data

Support count

Support count denotes the number of transactions in which a particular itemset is present. This is a very important property of an itemset as it denotes the frequency of occurrence for the itemset. This is expressed as

$$\sigma(X) = |\{t_i | X \subseteq t_i, t_i \varepsilon T\}|$$

where |{ }| denotes the number of elements in a set

In Table 5.3, the itemset {Bread, Milk, Egg} occurs together three times and thus have a support count of 3.

ASSOCIATION RULE

The result of the market basket analysis is expressed as a set of **association rules** that specify patterns of relationships among items. A typical rule might be expressed as{Bread, Milk}→{Egg}, which denotes that if Bread and Milk are purchased, then Egg is also likely to be purchased. Thus, association rules are learned from subsets of item sets. For example, the preceding rule was identified from the set of {Bread, Milk, Egg}.

It should be noted that an association rule is an expression of X → Y where X and Y are disjoint item sets, i.e. X ∩ Y = 0.

Support and confidence are the two concepts that are used for measuring the strength of an association rule. Support denotes how often a rule is applicable to a given data set. Confidence indicates how often the items in Y appear in transactions that contain X in a total transaction of N. Confidence denotes the predictive power or accuracy of the rule. So, the mathematical expressions are

$$\text{Support, } s(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{N} \qquad (9.8)$$

$$\text{Confidence, } c(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)} \qquad (9.9)$$

In our data set 5.3, if we consider the association rule {Bread, Milk} → {Egg}, then from the above formula

$$\text{Confidence } c(\{Bread, Milk\} \rightarrow \{Egg\}) = \frac{\text{support count of } \{Bread, Milk, Egg\}}{\text{support count of } \{Bread, Milk\}}$$

$$= \frac{3}{4}$$

$$= 0.75$$

$$\text{Confidence } c(\{Bread, Milk\} \rightarrow \{Egg\}) = \frac{\text{support count of } \{Bread, Milk, Egg\}}{\text{support count of } \{Bread, Milk\}}$$

$$= \frac{3}{4}$$

$$= 0.75$$

It is important to understand the role of support and confidence in the association analysis. A low support may indicate that the rule has occurred by chance. Also, from its application perspective, this rule may not be a very attractive business investment as the items are seldom bought together by the customers. Thus, support can provide the intelligence of identifying the most interesting rules for analysis.

Similarly, confidence provides the measurement for reliability of the inference of a rule. Higher confidence of a rule X → Y denotes more likelihood of to be present in transactions that contain X as it is the estimate of the conditional probability of Y given X.

Also, understand that the confidence of X leading to Y is not the same as the confidence of Y leading to X. In our example, confidence of {Bread, Milk} → {Egg} = 0.75 but confidence of

$$\{Egg\} \rightarrow \{Bread, Milk\} = \frac{3}{5} = 0.6.$$

Here, the rule {Bread, Milk} → {Egg} is the strong rule.

Association rules were developed in the context of Big Data and data science and are not used for prediction. They are used for unsupervised knowledge discovery in large databases, unlike the classification and numeric prediction algorithms. Still we will find that association rule learners are closely related to and share many features of the classification rule learners. As association rule learners are unsupervised, there is no need for the algorithm to be trained; this means that no prior labelling of the data is required. The programme is simply run

on a data set in the hope that interesting associations are found.

Obviously, the downside is that there is not an easy way to objectively measure the performance of a rule learner, aside from evaluating them for qualitative usefulness. Also, note that the association rule analysis is used to search for interesting connections among a very large number of variables. Though human beings are capable of such insight quite intuitively, sometimes it requires expert-level knowledge or a great deal of experience to achieve the performance of a rule-learning algorithm. Additionally, some data may be too large or complex for humans to decipher and analyse so easily.

THE APRIORI ALGORITHM FOR ASSOCIATION RULE LEARNING

As discussed earlier, the main challenge of discovering an association rule and learning from it is the large volume of transactional data and the related complexity. Because of the variation of features in transactional data, the number of feature sets within a data set usually becomes very large. This leads to the problem of handling a very large number of item sets, which grows exponentially with the number of features. If there are k items which may or may not be part of an itemset, then there is 2 ways of creating item sets with those items. For example, if a seller is dealing with 100 different items, then the learner need to evaluate $2 = 1 \times e$ item sets for arriving at the rule, which is computationally impossible. So, it is important to filter out the most important (and thus manageable in size) item sets and use the resources on those to arrive at the reasonably efficient association rules.

The first step for us is to decide the minimum support and minimum confidence of the association rules. From a set of transaction T, let us assume that we will find out all the rules that have support ≥ min S and confidence ≥ min C, where min S and min C are the support and confidence thresholds, respectively, for the rules to be k 100 30 considered acceptable. Now, even if we put the min S = 20% and min C = 50%, it is seen that more than 80% of the rules are discarded; this means that a large portion of the computational efforts could have been avoided if the item sets for consideration were first pruned and the item sets which cannot generate association rules with reasonable support and confidence were removed. The approach to achieve this goal is discussed below.

Step 1: decouple the support and confidence requirements. According to formula 9.8, the support of the rule X → Y is dependent only on the support of its corresponding item sets. For example, all the below rules have the same support as their item sets are the same {Bread, Milk, Egg}:

- {Bread, Milk} → {Egg}
- {Bread, Egg} → {Milk}
- {Egg, Milk} → {Bread}
- {Bread} → {Egg, Milk}
- {Milk} → {Bread, Egg}
- {Egg} → {Bread, Milk}

So, the same treatment can be applied to this association rule on the basis of the frequency of the itemset. In this case, if the itemset {Bread, Milk, Egg} is rare in the basket transactions, then all these six rules can be discarded without computing their individual support and confidence values. This identifies some important strategies for arriving at the association rules:

1. **Generate Frequent Itemset**: Once the minS is set for a particular assignment, identify all the itemsets that satisfy minS. These itemsets are called frequent itemsets.
2. **Generate Rules**: From the frequent itemsets found in the previous step, discover all the high confidence rules. These are called strong rules.

Please note that the computation requirement for identifying frequent itemsets is more intense than the rule generation. So, different techniques have been evolved to optimize the performance for frequent itemset generation as well as rule discovery as discussed in the next section.

BUILD THE APRIORI PRINCIPLE RULES

One of the most widely used algorithm to reduce the number of itemsets to search for the association rule is known as Apriori. It has proven to be successful in simplifying the association rule learning to a great extent. The principle got its name from the fact that the algorithm utilizes a simple prior belief (i.e. a priori) about the properties of frequent itemsets:

If an itemset is frequent, then all of its subsets must also be frequent.

This principle significantly restricts the number of itemsets to be searched for rule generation. For example, if in a market basket analysis, it is found that an item like 'Salt' is not so frequently bought along with the breakfast items, then it is fine to remove all the itemsets containing salt for rule generation as their contribution to the support and confidence of the rule will be insignificant.

The converse also holds true:

If an itemset is frequent, then all the supersets must be frequent too

These are very powerful principles which help in pruning the exponential search space based on the support measure and is known as support-based pruning. The key property of the support measure used here is that the support for an itemset never exceeds the support for its subsets. This is also known as the anti-monotone property of the support measure.

Let us use the transaction data in Table 5.3 to illustrate the Apriori principle and its use. From the full itemset of six items {Bread, Milk, Egg, Butter, Salt, Apple}, there are 2 ways to create baskets or itemsets (including the null itemset) as shown in Figure 5.14:
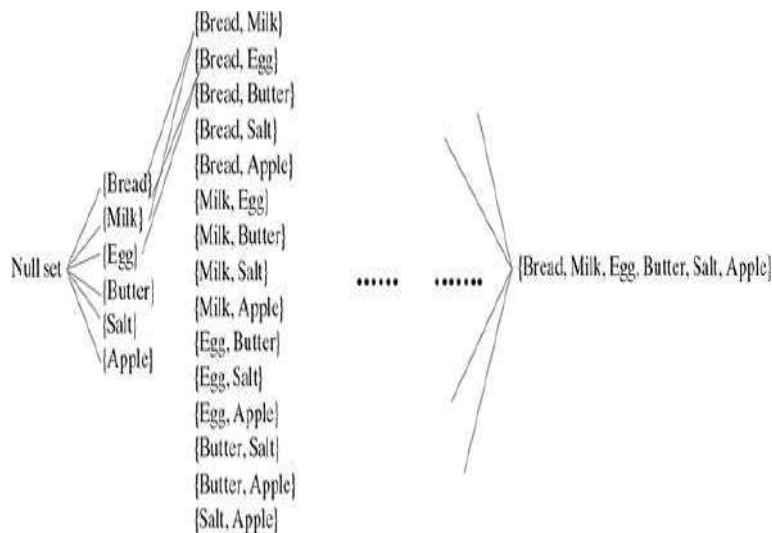
FIG. 5.14 Sixty-four ways to create item sets from 6 items

Without applying any filtering logic, the brute-force approach would involve calculating the support count for each itemset in Figure 5.15. Thus, by comparing each item in the generated itemset with the actual transactions mentioned in Table 5.3, we can determine the support count of the itemset. For example, if {Bread, Milk} is present in any transactions in Table 5.3, then its support count will be incremented by 1. As we can understand, this is a very computation heavy activity, and as discussed earlier, many of the computations may get wasted at a later point of time because some item sets will be found to be infrequent in the transactions. To get an idea of the total computations to be done, the number of comparisons to be done is $T \times N \times L$, where T is the number of transactions (6 in our case), N is the number of candidate item sets (64 in our case), and L is the maximum transaction width (6 in our case).
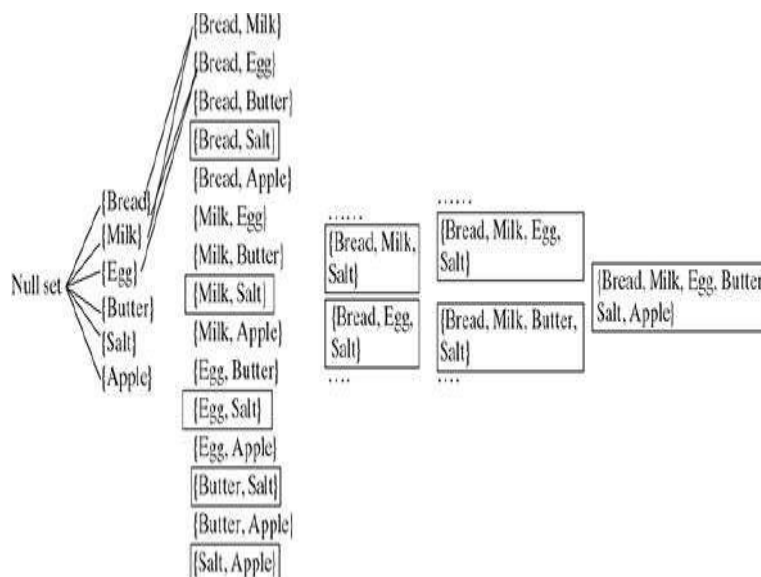

FIG. 5.15 Discarding the item sets consisting of Salt

Let us apply the Apriori principle on this data set to reduce the number of candidate item sets (N). We could identify from the transaction Table 5.3 that Salt is an infrequent item. So, by applying the Apriori principle, we can say that all the item sets which are superset of Salt will be infrequent and thus can be discarded from comparison to discover the association rule as shown in **Figure 5.15.**

This approach reduces the computation effort for a good number of item sets and will make our search process more efficient. Thus, in each such iteration, we can determine the support count of each itemset, and on the basis of the min support value fixed for our analysis, any itemset in the hierarchy that does not meet the min support criteria can be discarded to make the rule generation faster and easier.

To generalize the example in order to build a set of rules with the Apriori principle, we will use the Apriori principle that states that all subsets of a frequent itemset must also be frequent. In other words, if {X, Y} is frequent, then both {X} and {Y} must be frequent. Also by definition, the support metric indicates how frequently an itemset appears in the data. Thus, if we know that {X} does not meet a desired support threshold, there is no reason to consider {X, Y} or any itemset containing {X}; it cannot possibly be frequent.

This logic of the Apriori algorithm excludes potential association rules prior to actually evaluating them. The actual process of creating rules involves two phases:

- Identifying all item sets that meet a minimum support threshold set for the analysis
- Creating rules from these item sets that meet a minimum confidence threshold which identifies the strong rules

The first phase involves multiple iterations where each successive iteration requires evaluating the support of storing a set of increasingly large item sets. For instance, iteration 1 evaluates the set of one-item item sets (one-item sets), iteration 2 involves evaluating the two-item sets, etc.. The result of each iteration N is a set of all N-item sets that meet the minimum support threshold. Normally, all the item sets from iteration N are combined in order to generate candidate item sets for evaluation in iteration N + 1, but by applying the Apriori principle, we can eliminate some of them even before the next iteration starts. If {X}, {Y}, and {Z} are frequent in iteration 1 while {W} is not frequent, then iteration 2 will consider only {X, Y}, {X, Z}, and {Y, Z}. We can see that the algorithm needs to evaluate only three item sets rather than the six that would have been evaluated if sets containing W had not been eliminated by apriori.

By continuing with the iterations, let us assume that during iteration 2, it is discovered that {X, Y} and {Y, Z} are frequent, but {X, Z} is not. Although iteration 3 would normally begin by evaluating the support for {X, Y, Z}, this step need not occur at all. The Apriori principle states that {X, Y, Z} cannot be frequent, because the subset {X, Z} is not. Therefore, in iteration 3, the algorithm may stop as no new itemset can be generated.

Once we identify the qualifying item sets for analysis, the second phase of the Apriori algorithm begins. For the given set of frequent item sets, association rules are generated from all possible subsets. For example, {X, Y} would result in candidate rules for {X} → {Y} and {Y} → {X}. These rules are evaluated against a minimum confidence threshold, and any rule that does not meet the desired confidence level is eliminated, thus finally yielding the set of strong rules.

Though the Apriori principle is widely used in the market basket analysis and other applications of association rule help in the discovery of new relationship among objects, there are certain strengths and weaknesses we need to keep in mind before employing it over the target data set:

Strengths	Weaknesses
• Provides reasonable accuracy while working with very large amounts of transactional data • Discovers rules that are easy to understand • Provides valuable insight into the unexpected knowledge in data sets, which is a key aspect of learning	• Not very accurate in the case the data set is small as the smaller occurrences of itemsets may not be due to chance • Some effort is involved to separate the insight from the common sense • In the case of widespread presence of random patterns, the principle can draw spurious conclusions