

B.Tech-CSE(AI&ML)

B.Tech II Year - II Semester	MACHINE LEARNING (23HPC3301)	L	T	P	C
		3	0	0	3

Course Objectives: The objectives of the course are

- Understand the basic concepts of machine Learning
- Apply different machine learning algorithms
- Implement clustering techniques

,

Course Outcomes:

- Identify machine learning techniques suitable for a given problem. (L3)
- Solve the problems using various machine learning algorithms(L3)
- Apply data processing techniques (L3)
- Apply the design of intelligent machines (L3)
- Evaluate different clustering techniques(L5)

UNIT-I: Introduction to Machine Learning:

Evolution of Machine Learning, Paradigms for ML, Learning by Rote, Learning by Induction, Reinforcement Learning, Types of Data, Matching, Stages in Machine Learning, Data Acquisition, Feature Engineering, Data Representation, Model Selection, Model Learning, Model Evaluation, Model Prediction, Search and Learning, Data Sets.

UNIT-II: Nearest Neighbor-Based Models:

Introduction to Proximity Measures, Distance Measures, Non-Metric Similarity Functions, Proximity Between Binary Patterns, Different Classification Algorithms Based on the Distance Measures ,K-Nearest Neighbor Classifier, Radius Distance Nearest Neighbor Algorithm, KNN Regression, Performance of Classifiers, Performance of Regression Algorithms.

UNIT-III: Models Based on Decision Trees:

Decision Trees for Classification, Impurity Measures, Properties, Regression Based on Decision Trees, Bias–Variance Trade-off, Random Forests for Classification and Regression.

The Bayes Classifier: Introduction to the Bayes Classifier, Bayes' Rule and Inference, The Bayes Classifier and its Optimality, Multi-Class Classification | Class Conditional Independence and Naive Bayes Classifier (NBC)

UNIT-IV: Linear Discriminants for Machine Learning:

Introduction to Linear Discriminants, Linear Discriminants for Classification, Perceptron Classifier, Perceptron Learning Algorithm, Support Vector Machines, Linearly Non-Separable Case, Non-linear SVM, Kernel Trick, Logistic Regression, Linear Regression, Multi-Layer Perceptrons (MLPs), Back propagation for Training an MLP.

UNIT-V: Clustering :

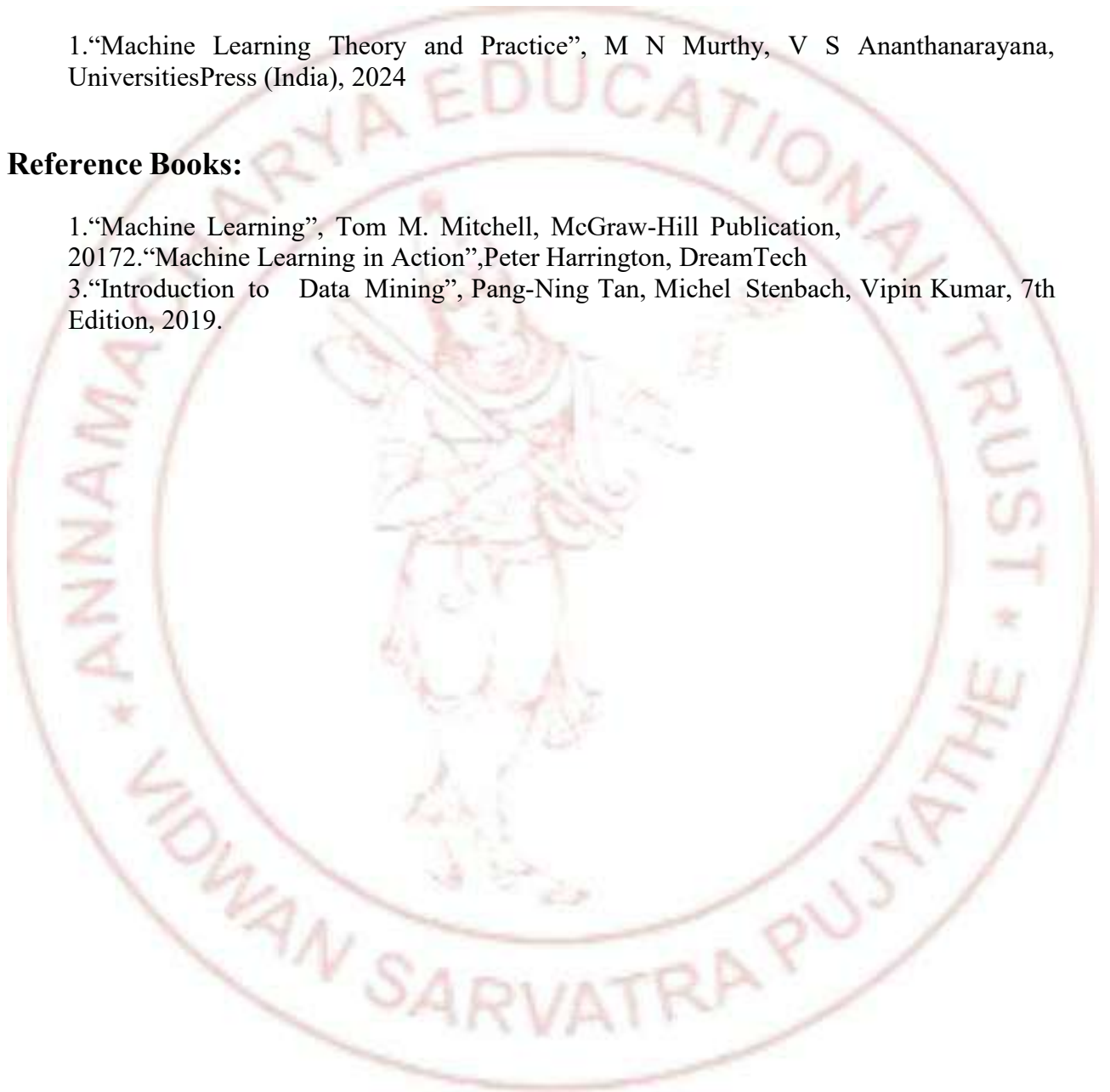
Introduction to Clustering, Partitioning of Data, Matrix Factorization | Clustering of Patterns, Divisive Clustering, Agglomerative Clustering, Partitional Clustering, K-Means Clustering, Soft Partitioning, Soft Clustering, Fuzzy C-Means Clustering, Rough Clustering, Rough K-Means Clustering Algorithm, Expectation Maximization-Based Clustering, Spectral Clustering.

Text Books:

1.“Machine Learning Theory and Practice”, M N Murthy, V S Ananthanarayana, Universities Press (India), 2024

Reference Books:

1.“Machine Learning”, Tom M. Mitchell, McGraw-Hill Publication, 2012.
2.“Machine Learning in Action”, Peter Harrington, DreamTech
3.“Introduction to Data Mining”, Pang-Ning Tan, Michel Stenbach, Vipin Kumar, 7th Edition, 2019.



Introduction to Machine Learning:

Machine learning is the practice of programming computers to learn from data. The program will easily be able to determine if given are important or spam.

ML is defined as a discipline of AI (Artificial Intelligence) that provides machines the ability to automatically learn from data and past experiences to identify patterns and make predictions with minimal human intervention.

It uses computer algorithms and classifiers that improve their efficiency automatically through experience.

It is an application of AI that enables systems to learn from vast volumes of data and solve specific problems.

What is learning?

It is a process by which a system improves performance (P) from experience (E) at some specific task (T).

Example: A well-defined learning task is given by $\langle P, T, E \rangle$.

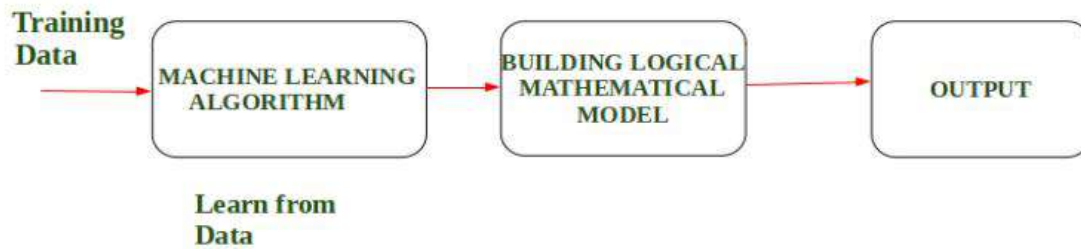
How does ML work?

A machine learning system learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it.

Flow chart:

Past Data → Process & Analyze Data → Create Models

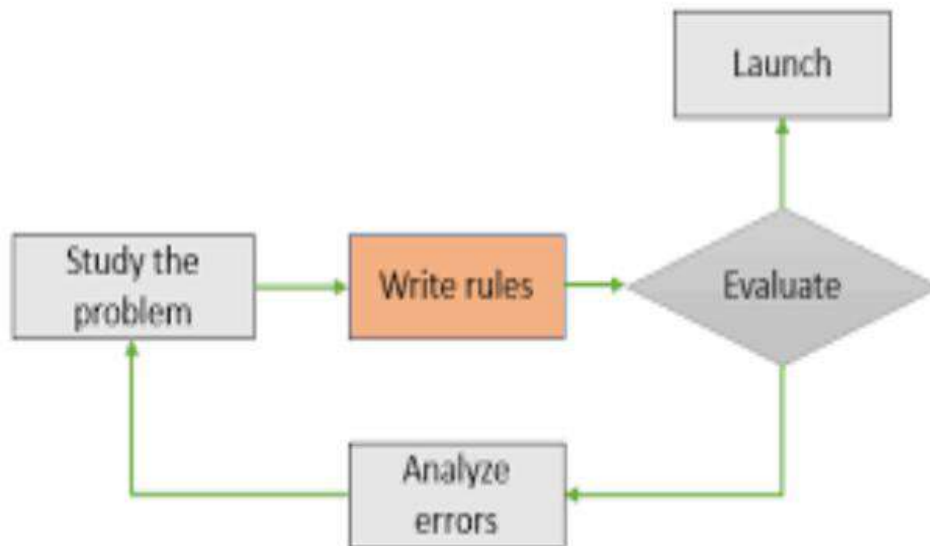
The below block diagram explains the working of ML algorithm:



Why ML (or Need for Machine Learning):

The need for ML is increasing day by day. The reason behind the need for ML is that it is capable of doing tasks that are too complex for a person to implement directly.

As a human, we have some limitations as we cannot access the huge amount of data manually. So for this, we need some computer systems and here comes the ML to make things easy for us.



Evolution of Machine Learning:

1.The birth of ML(1950):

Before 1950, there was a lot of research and theoretical studies for machine learning, but the year 1950 is marked as the real birth of machine learning.

“Alan Turing” the famous mathematician, researcher, Computer genius submitted a paper called “imitation game” and made the world astonished.

2)1951 - First Neural Network:

The very first neural network was made by "Marvin Minsky with Dean Edmonds in 1951. Neural networks connect the thinking process of machines and computers.

3)1974 - Naming of Machine Learning:

It was 1974, the year coined the name for ML from the proposed words like Informatics, Computational Intelligence, and Artificial Intelligence.

4)1996 - Game Changer:IBM's Deep Blue computer beat the world-famous champion Garry Kasparov in chess. This proved that machines can also think like humans.

5)ML at Present:

It is now responsible for some of the most significant advancements in technology. It is being used for the new industry of "self-driving vehicles".

Some of the most trending real-world applications of ML at present are:

- Image Recognition

- Speech Recognition

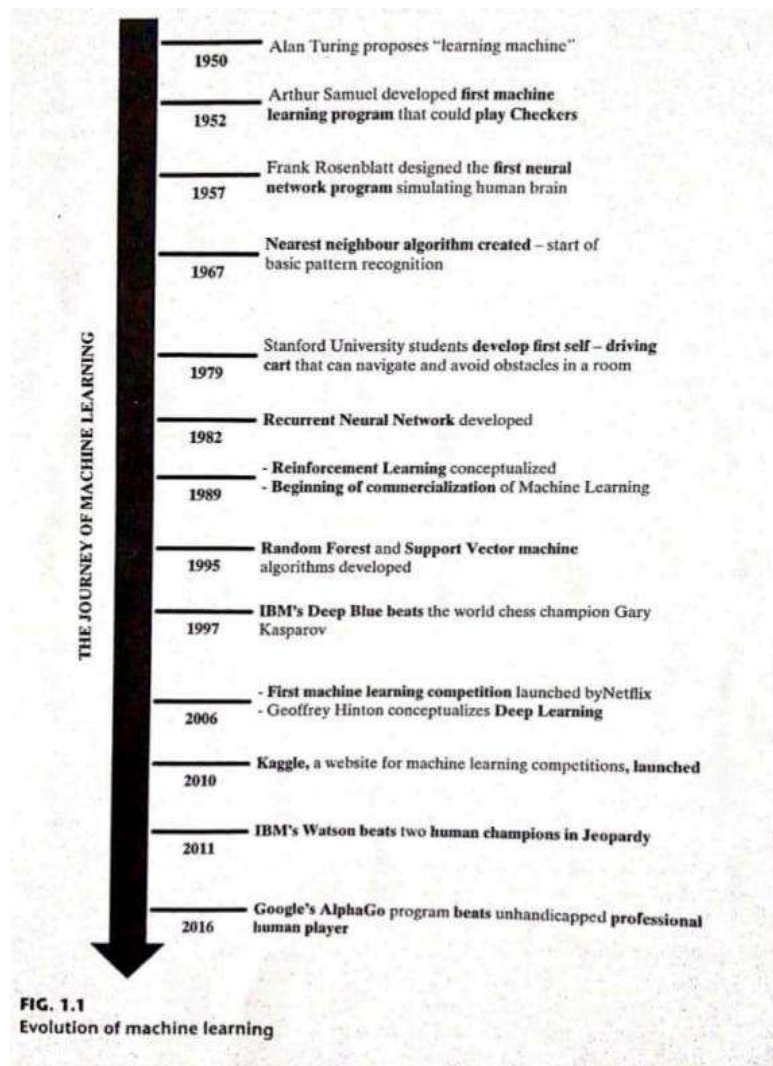
- Traffic Prediction

- Product Recommendations

- Email Spam & Malware Filtering

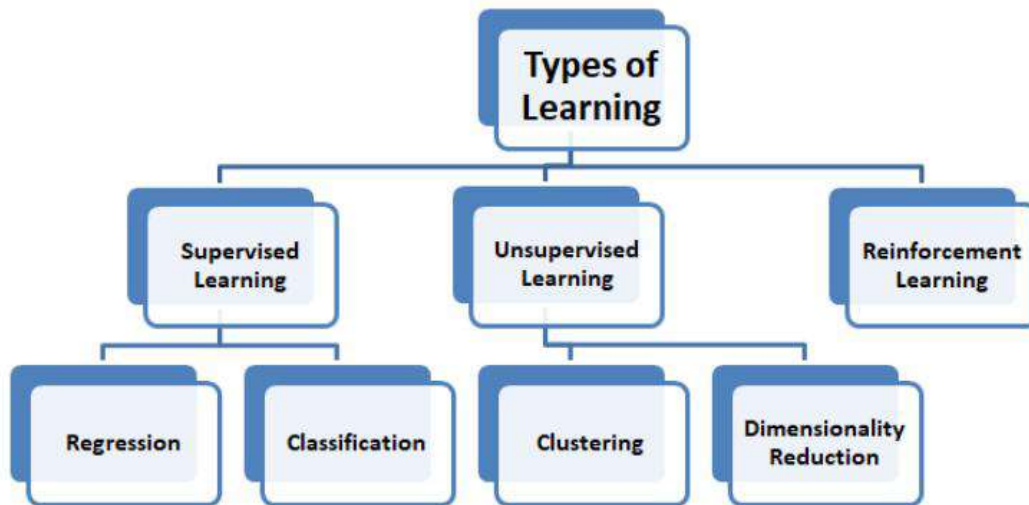
- Self driving cars

- Online fraud detection etc.



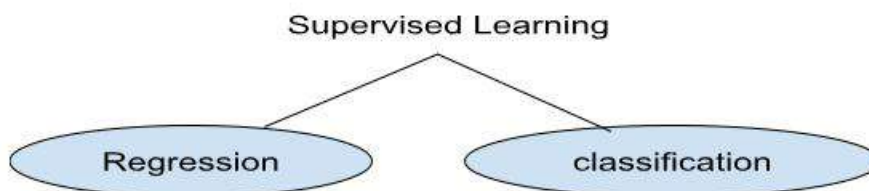
Paradigms (or) Types of ML Analysis (or) Taxonomy of ML

1. ML is divided into three primary learning model approaches:
 - Supervised
 - Unsupervised
 - Reinforcement
2. Each model differs in training. Each has its strengths and faces different tasks or problems.
3. When choosing an ML model to deploy, an organization needs to understand the available data and the problem to be solved.



1) Supervised Machine Learning:

- 1) Supervised ML algorithms are the most commonly used for predictive analytics.
- 2) It requires human interaction to label data ready for accurate supervised learning.
- 3) The model is taught by example using input and output datasets processed by human experts, usually data scientists.
- 4) It is commonly used for solving regression and classification problems.



Regression:

1. It involves estimating the mathematical relationship between a continuous variable and one or more other variables.

2.It is used for the prediction of continuous variables such as **weather forecasting, market trends**, etc.

3.Below are some popular regression algorithms:

- ✧ Linear Regression
- ✧ Non-linear Regression
- ✧ Regression Trees
- ✧ Polynomial Regression
- ✧ Bayesian Linear Regression

Classification:

1. Classification algorithms are used when the output variable is categorical, which means there are two classes such as Yes-No, Male-Female, True-False, Pass-Fail, etc.

2.Below are some popular classification algorithms. They are:

- ✧ Random Forest
- ✧ Decision Trees
- ✧ Logistic Regression
- ✧ Support Vector Machines

•

Advantages of supervised learning:

1.It helps us to solve various real-world problems such as "fraud detection," "spam filtering," etc.

2.It can predict the output on the basis of prior experience.

Disadvantages of supervised learning:

1.It is not suitable for handling complex tasks.

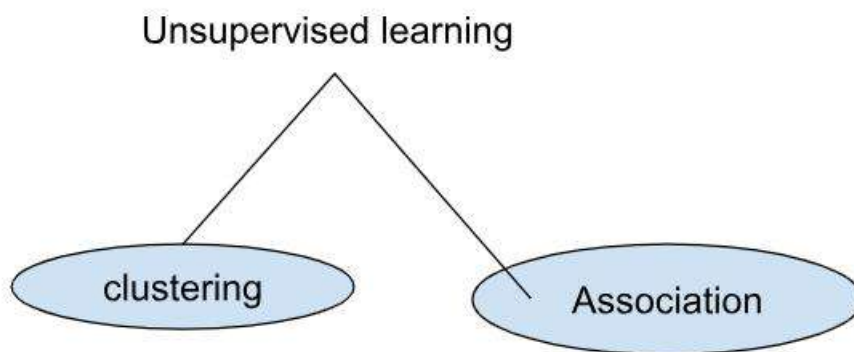
2.It requires lots of computation time.

3.It cannot predict the correct output if the test data is different from the training dataset.

2) Unsupervised learning:

1.It is a type of ML in which models are trained using unlabeled datasets and are allowed to act on that data without any supervision.

- 2.It is to find the underlying structure of datasets, group that data according to similarities and represent that dataset in a compressed format.
- 3.It works on unlabeled and uncategorized data, which makes unsupervised learning more important.
- 4.It is commonly used for solving clustering methods and association methods.



Clustering:

- 1.It is the grouping of data that have similar characteristics.
- 2.It helps segment data into groups and analyze each to find patterns.
Example: Clustering algorithms identify groups of users based on their online purchasing history and then send each member targeted ads.

Association:

- 1.It consists of discovering groups of items frequently observed together.
- 2.Online retailers use associations to suggest additional purchases to a user based on the content of their shopping cart.

Unsupervised Learning Algorithms:

Below is the list of some popular unsupervised learning algorithms:

- ✧ K-means Clustering
- ✧ KNN (K-Nearest Neighbors)

- ✧ Neural networks
- ✧ Independent component analysis etc.

Advantages of Unsupervised Learning:

- 1.It is easy to get unlabeled data in comparison to labeled data.
- 2.It is used for more complex tasks as compared to supervised learning because, in unsupervised learning, we don't have labeled input data.

Disadvantages of Unsupervised Learning:

- 1.It is intrinsically more difficult than supervised learning, as it does not have corresponding output.
- 2.The result of the unsupervised learning algorithm may be less accurate as input data is not labeled.

The main differences between Supervised and Unsupervised Learning are given below:

Supervised Learning	Unsupervised Learning
1) Supervised learning algorithms are trained using labeled data.	1) Unsupervised learning algorithms are trained using unlabeled data.
2) It takes direct feedback to check if it is predicting the correct output or not.	2) It does not take feedback.
3) Supervised learning models predict the output.	3) Unsupervised learning models find the hidden patterns in data.
4) It can be categorized in Classification and regression problems	4) It can be classified in clustering and Association problems

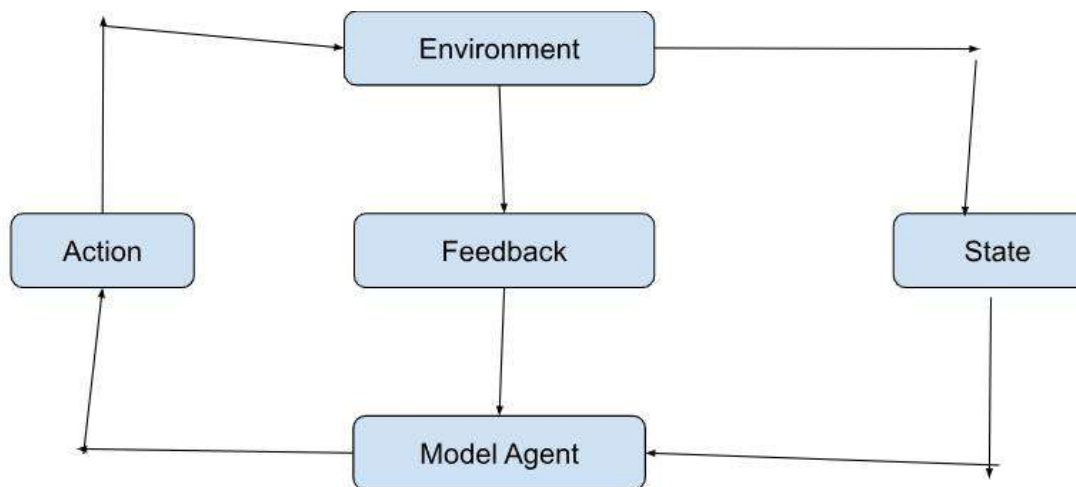
5) It produces an accurate result
result as

5) It may give less accurate

Compared to supervised learning

3) Reinforcement Learning:

- ✧ It works on a feedback-based process, in which an agent (AI) automatically explores its surroundings by hit & trial, taking action, learning from experiences, and improving its performances.
- ✧ Agent get rewarded for each good action and get punished for each bad action, hence the goal of these learning agents is to maximize the rewards.
- ✧ Its learning process is similar to a human being.
For example: A child learns various things by experiences in his day-to-day life.
- ✧ Reinforcement learning problems can be formalized using the "Markov Decision Process (MDP)". In MDP, the agent constantly interacts with the environment & performs actions. At each action, the environment responds & generates a new state.



Categories of RL:

It is mainly categorized into two types of methods:

1. Positive RL
2. Negative RL

1)Positive RL:

It enhances the strength of the behavior of the agent and positively impacts it.

2)Negative RL:

It works exactly opposite to the positive RL.

Real-world use cases of Reinforcement Learning:

- 1.Video games
- 2.Resource management
- 3.Robotics
- 4.Text mining

Advantages of RL:

- 1.Versatility:** Wide range of applications like robotics, autonomous vehicles, healthcare, and game playing, etc.
- 2.Scalability:** It can be scaled to handle large and complex problems.

Disadvantages of RL:

- 1.High computational cost
- 2.Overfitting to specific environments

Note:

To overcome the drawbacks of supervised learning & unsupervised learning algorithms, the concept of supervised learning is introduced.

Learning by Rote:

1)Learning by rote in machine learning refers to the phenomenon where a model memorizes the training data instead of learning the underlying patterns or generalizations.

2)It is compared to how students might memorize answers without understanding concepts.

3)In ML, this behavior often manifests as "overfitting."

4) It is a memorization technique based on repetition. This involves memorization in an effective manner. It is a form of learning that is popular in elementary schools where the alphabet and numbers are memorized.

5) Memorizing simple addition and multiplication tables are also examples of rote learning. In the case of data caching, we store computed values so that we do not have to recompute them later.

6) Caching is implemented by search engines and it may be viewed as another popular scheme of rote learning. When computation is more expensive than recall, this strategy can save a significant amount of time.

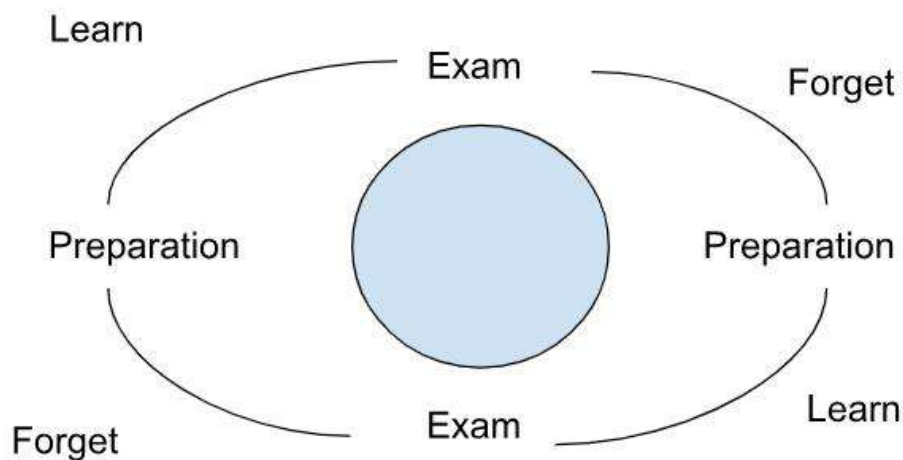


Fig: The Rote Loop

Rote Learning vs. Meaningful Learning:

Rote learning:

- ✧ Requires the learner to repeat facts and procedures until they are thoroughly memorized.

Meaningful learning:

- ✧ Focuses on understanding rather than just memorizing.
- ✧ Requires connecting new information to prior knowledge.

How It Works:

- ✧ A machine is programmed to store new information and compare new input to its history of inputs and outputs.
- ✧ If the machine has previously encountered the input, it can retrieve the stored output.

Benefits of Rote Learning:

It can save time compared to recomputing values. It can also be used to establish a foundational understanding of basic concepts.

Learning by Deduction:

Deductive learning deals with the exploitation of deductions made earlier. This type of learning is based on reasoning that is truth-preserving. Given AAA, and if AAA then BBB ($A \rightarrow BA \rightarrow B$), we can deduce BBB. We can use BBB along with $B \rightarrow CB \rightarrow C$ ($B \rightarrow CB \rightarrow C$) to deduce CCC. Note that whenever AAA and $A \rightarrow BA \rightarrow B$ are **True**, then BBB is **True**, ratifying the truth-preserving nature of learning by deduction.

Consider the following statements:

1. It is raining.
2. If it rains, the roads get wet.
3. If a road is wet, it is slippery.

Learning by Abduction:

Here, we infer A from B and ($A \rightarrow B$). Notice that this is not truth-preserving like in deduction, as both B and ($A \rightarrow B$) can be **True** and A can be **False**. Consider the following inference:

- 1) An aeroplane is a flying object ($\text{aeroplane} \rightarrow \text{flying object}$).
- 2) A is a flying object.

From (1) and (2), we infer using abduction that A is an aeroplane. This kind of reasoning can lead to incorrect conclusions. For example, A could be a bird or a kite.

Learning by Induction:

This is the most popular and effective form of ML. Here, learning is achieved with the help of examples or observations. It may be categorized as follows:

- ✧ **Learning from Examples:** Here, it is assumed that a collection of labeled examples is provided, and the ML system uses these examples to make a prediction on a new data pattern. In supervised classification or learning from examples, we deal with two ML problems: **classification** and **regression**.

1.Classification: The problem is to learn an ML model using such data to classify a new data pattern. This is also called **supervised learning** as the model is trained with the help of such exemplar data. It may be provided by an expert in several practical situations. For example, a medical doctor may provide examples of normal patients and patients infected by COVID-19 based on some test results.

2.Regression: Contrary to classification, there are several prediction applications where labels come from a possibly infinite set. For example, the share value of a stock could be a positive real number. The stock may have different values at a particular time, and each of these values is a real number. This is a typical regression or curve-fitting problem.

- ✧ **Learning from Observations:** Observations are also instances like examples but are different because observations need not be labeled. In this case, we cluster or group observations into a smaller number of groups. Such grouping is performed with the help of a clustering algorithm that assigns similar patterns to the same group/cluster.

Challenges in Inductive Learning:

- 1.Noise in Data:** Can lead to incorrect generalizations.
- 2.Over-generalization:** Creating overly broad rules that don't fit well with specific data points.
- 3.Computational Complexity:** Searching through large hypothesis spaces can be resource-intensive.

Applications of Learning by Induction:

- ✧ **Image Classification** (e.g., recognizing handwritten digits).
- ✧ **Natural Language Processing** (e.g., spam detection).
- ✧ **Predictive Analytics** (e.g., forecasting sales).
- ✧ **Robotics** (e.g., learning motion patterns).

Introduction of ML

Introduction of ML:

- * Machine learning is the practice of programming computers (how to learn from data).
- * The program will be easily able to determine if we given important or spam.
- * ML is defined as a discipline of "AI" that provides machine ability to automatically learn from data and past experiences, to identify pattern and make, minimal human interaction. Prediction
- * It uses computer algorithm and classifiers that improve their efficiency automatically through experience.
- * It is an application of AI that enables systems to learn from vast volumes of data and solve specific problems.

What is Learning?

It is a process by which a system improves performance

(P) Performance, (E) Experiences, (T) at specific task

Eg: A well defined learning task is given by

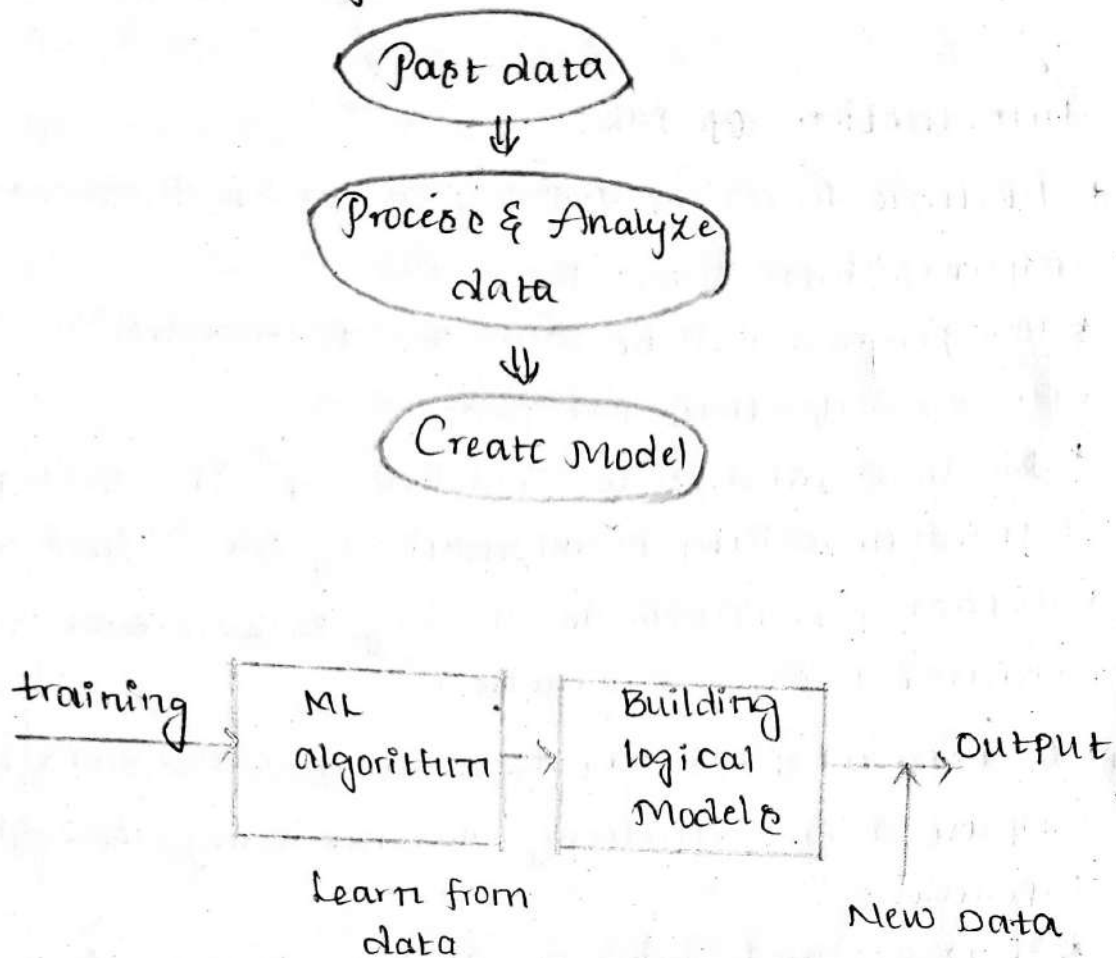
$\langle P, T, E \rangle$

How does ML work?

A ML System learns from historical data, builds the prediction models and when ever it needs a new data, Predicts the o/p for it.

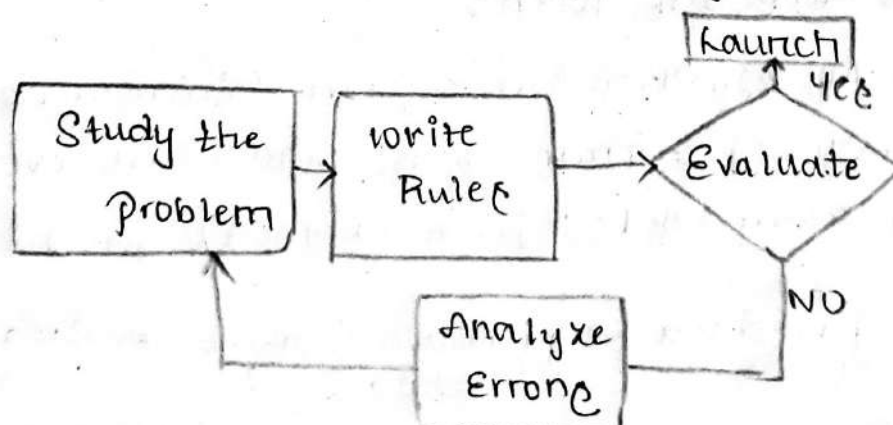
Past data \rightarrow Process & Analyze data \rightarrow Create models

The below block diagram Explains the work of ML algorithm.



Need for ML:

The need for ML is increasing day by day the reason it is capable of doing tasks that are 2 Complex for a Person to implement directly
* As a movement we have some limitation as cannot access the huge amount of data manually, so for these we need from computer systems and here comes the ML to make things easy for us.



THE JOURNEY OF MACHINE LEARNING

1950

Alan Turing proposes "learning machine"

1952

Arthur Samuel developed **first machine learning program** that could **play Checkers**

1957

Frank Rosenblatt designed the **first neural network program** simulating human brain

1967

Nearest neighbour algorithm created – start of basic pattern recognition

1979

Stanford University students **develop first self – driving cart** that can navigate and avoid obstacles in a room

1982

Recurrent Neural Network developed

1989

- **Reinforcement Learning** conceptualized
- **Beginning of commercialization** of Machine Learning

1995

Random Forest and Support Vector machine algorithms developed

1997

IBM's Deep Blue beats the world chess champion Gary Kasparov

2006

- **First machine learning competition** launched by Netflix
- Geoffrey Hinton conceptualizes **Deep Learning**

2010

Kaggle, a website for machine learning competitions, **launched**

2011

IBM's Watson beats two human champions in Jeopardy

2016

Google's AlphaGo program beats unhandicapped professional human player

Theorem proving

Logic

Inverted index

Data structures

Approximation, randomization

Algorithms

Group, relation function, graph

Discrete structures

Clustering

Linear algebra

Bayes, max likelihood

Probability statistics

Max margin, backpropagation

Optimization

Entropy

Information theory

Artificial intelligence

Machine learning (ML) is the process of learning a model that can be used in prediction based on data. Prediction involves assigning a data item to one of the classes or associating the data item with a number.

Deep learning (DL) is an offshoot of ML. In fact, Perceptron was the earliest popular ML tool and it forms the basic building block of various DL architectures.

(In the early days of AI system. some of).

In the early days of AI, it was opined that mathematical logic was the ideal vehicle for building AI systems. Some of the initial contributions in this area like the general Problem solver (GPS), Automatic Theorem Proving (ATP), rule-based systems. and programming languages like Prolog and lisp (lambda calculus-based) were all outcome of this view.

* Evolution of ML :-

1) 1950 - The birth of ML :-

1) Before 1950, there was a lot of research and theoretical studies for ML, but the year 1950 is marked as the real birth of machine learning.

2) "Alan Turing" the famous mathematician,

researcher, computer genius, and thinker, submitted a paper called "imitation game" and made the world astonished.

II) 1951 - First Neural Network :-

The very first neural network was made by "Marvin Minsky" with Dean Edmonds in 1951. Neural Networks connect the thinking process of machines and computers.

III) 1974 - Naming of Machine Learning :-

It was 1974, the year coined the name for ML from the proposed words like Informatics, Computational intelligence, and artificial intelligence.

IV) 1996 - Game Changer :-

IBM's Deep Blue Computer beat the world-famous champion, "Garry Kasparov" in chess. This proved that machines can also think like humans.

V) ML At Present :-

1) It is now responsible for some of the most significant advancements in technology. It is being used for the new industry of "self-driving vehicles".

2) Some of the most trending real-world applications of ML at present is

1) Image Recognition

2) Speech Recognition

3) Traffic prediction

4) Product recommendations

5) Email spam & malware filtering

1.2.1 Learning by Rote

This involves memorization in an effective manner. It is a form of learning that is popular in elementary schools where the alphabet and numbers are memorized. Memorizing simple addition and multiplication tables are also examples of rote learning. In the case of data caching, we store computed values so that we do not have to recompute them later. Caching is implemented by search engines and it may be viewed as another popular scheme of rote learning. When computation is more expensive than recall, this strategy can save a significant amount of time. Chess masters spend a lot of time memorizing the great games of the past. It is this rote learning that teaches them how to 'think' in chess. Various board positions and their potential to reach the winning configuration are exploited in games like chess and checkers.

Learning by rote:

Learning by rote in machine learning refers to the phenomena of the model memorises the training data

Instead of learning the underlying patterns for generalization

* It is compared to how student might memorize answers without conceptual understanding.

* In ML is behaviour manifested of overfitting

* It is a memorization technique based on repetition.

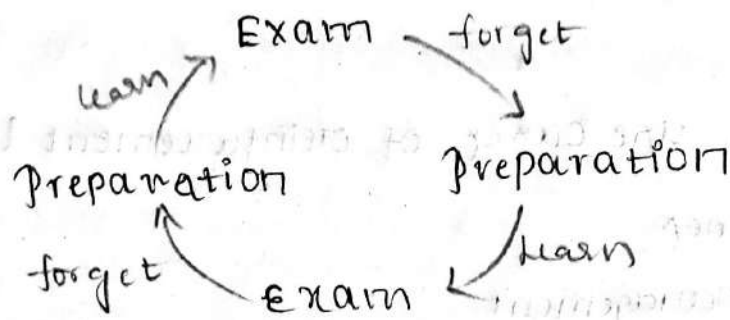


fig: learning by rote loop.

~~Group~~ rote learning vs Meaningful learning:

It requires the learner to repeat facts and procedures until they are curiously memorize.

* It requires to a focus on understanding rather than just memorizing.

* It requires connecting new information to prior knowledge.

How it works:

The Machine is a program to store new information and compare new ip to its history of new ip and op.

* If the machine has previous encounters with the ip then it can be retrieve the op very fastly.

Benefits of rote learning:

* It can save time compared to decomposing values.

* It can also be used for foundational understanding basic concepts.

1.2.4 Learning by Induction

This is the most popular and effective form of ML. Here, learning is achieved with the help of examples or observations. It may be categorized as follows:

- **Learning from Examples:** Here, it is assumed that a collection of labelled examples are provided and the ML system uses these examples to make a prediction on a new data pattern. In supervised classification or learning from examples, we deal with two ML problems: classification and regression.
 1. **Classification:** Consider the handwritten digits shown in Fig. 1.3. Here, each row has 15 examples of each of the digits. The problem is to learn an ML model using such data to classify a new data pattern. This is also called **supervised learning** as the model is learnt with the help of such exemplar data. It may be provided by an expert in several practical situations. For example, a medical doctor may provide examples of normal patients and patients infected by COVID-19 based on some test results. In the case of handwritten digits, we have 10 class labels, one class label corresponding to each of the digits from 0 to 9. In classification, we would like to assign an appropriate class label from these labels to a new pattern.
 2. **Regression:** Contrary to classification, there are several prediction applications where the labels come from a possibly infinite set. For example, the share value of a stock could be a positive real number. The stock may have different values at a particular time and each of these values is a real number. This is a typical regression or curve fitting problem. The practical need here is to predict the share value of a stock at a future time instance based on past data in the form of examples.
- **Learning from Observations:** Observations are also instances like examples but they are different because observations need not be labelled. In this case, we cluster or group the observations into a smaller number of groups. Such grouping is performed with the help of a clustering algorithm that assigns similar patterns to the same group/cluster. Each cluster



FIG. 1.3 Examples of handwritten digits labelled 0 to 9

could be represented by its centroid or mean. Let x_1, x_2, \dots, x_p be p elements of a cluster. Then the centroid of the cluster is defined by

$$\frac{1}{p} \sum_{i=1}^p x_i$$

Let us consider the handwritten digit data set of 3 classes: 0, 1 and 3. By using the class labels and clustering patterns of each class separately, we obtain 3 clusters that give us the 9 centroids shown in Fig. 1.4.



FIG. 1.4 Cluster centroids using the class labels in clustering

However, when we cluster the entire data of digits 0, 1 and 3 into 9 clusters, we obtain the centroids shown in Fig. 1.5. So, the clusters and their representatives could differ based on how we exploit the class labels.

331 311 001

FIG. 1.5 Cluster centroids without using the class labels in clustering

Key Concepts in Learning by Induction:

① Inductive Reasoning:

- Starts with specific examples and generalizes to broader rules or hypothesis

Ex: Observing several examples of labeled data (This, is a cat, that is not a cat) and deriving a model that distinguishes cats from non-cats.

② Training and Generalization:

Training: The process of fitting a model to a training dataset

Generalization: The ability of the model to perform well on unseen data, not just the training data

③ Hypothesis Space :-

- Refers to the set of all possible models or functions the learning algorithm can choose from.
- The goal is to find the best hypothesis that fits the training data and generalizes to unseen data.

④ Bias and Variance :-

Bias :- Assumptions made by the model to simplify learning (eg. linear relationships)

Variance :- ~~stability~~ Sensitivity of the model to variations in the training data

→ A balance b/w bias & variance is crucial for effective induction.

⑤ overfitting & underfitting :-

overfitting :- When the model learns the training data too well, including noise, leading to poor generalization.

underfitting :- When the model is too simplistic to capture the underlying patterns in the data.

⑥ Algorithms that use Induction :-

Decision trees :- Learn decision rules from data

SVM's (Support Vector machines) :- Find the hyperplane that best separates data classes.

Neural Networks :- Learn representations through multiple layers of abstraction.

Linear regression :- infers a linear relationship b/w

inputs & outputs.

challenges in Inductive learning:

- 1) Noise in Data: Can lead to incorrect generalizations
- 2) over-generalizations: Creating overly broad rules that don't fit well with specific data points.
- 3) Computational Complexity: Searching through large hypothesis spaces can be resource intensive.

Applications of Learning by Induction:

- 1) Image Classification (eg. recognizing handwritten digits)
- 2) Natural language processing (eg. spam detection)
- 3) Predictive Analytics (eg. forecasting sales)
- 4) Robotics (eg. learning motion patterns)

Topic 5:

* Basic Types of Data in Machine Learning:

1) The first step in ML activity, starts with Data.

2) ML Activities starts with below steps

(i) ~~pre~~paring to model

(ii) Learning

(iii) Performance Evaluation

(iv) Performance improvement

3) Before starting with type of data, let's first understand what a dataset is and what are the Elements of a data set.

- 4) A dataset is a collection of "related Information" or records. Each row of a dataset is called a "record".
- 5) Each dataset also has multiple attributes, each of which gives information on a specific characteristic.

*Student data set:

Roll no	Name	Gender	Age

*Student performance dataset:

Roll no	Maths	Physics	Percentage

In the student dataset there are 4 attributes namely roll no, name, gender, age.

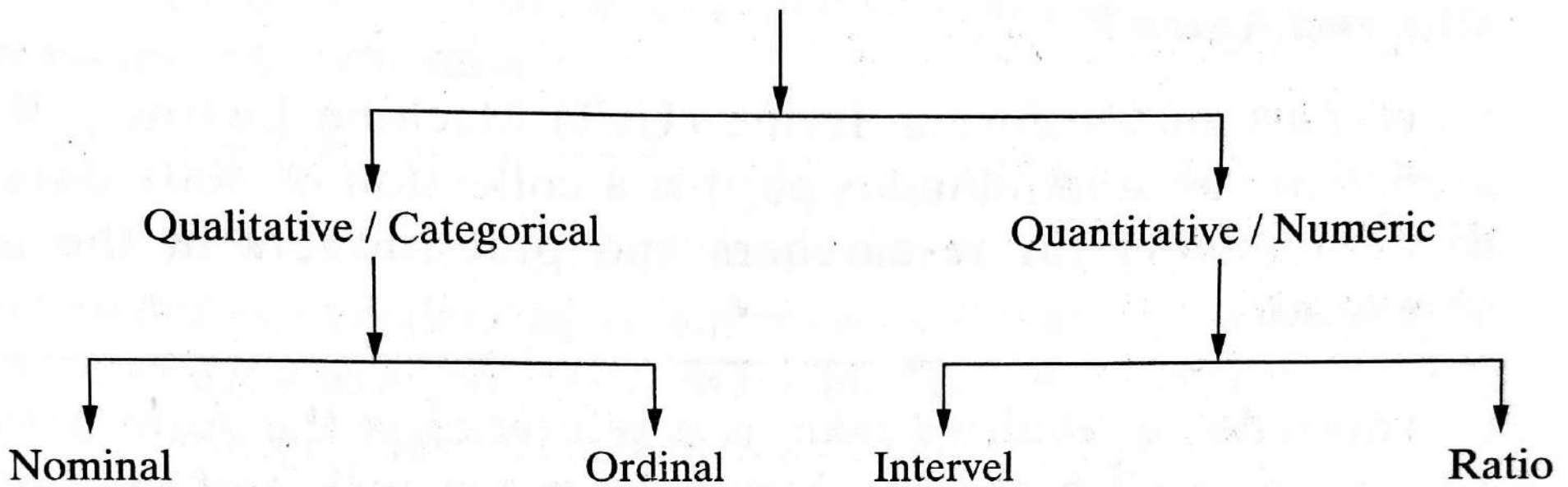
Attributes can also be termed as feature, variable, dimension or field.

Both the data sets student & student performance are having 4 dimensions or 4 features. Hence they are told to have 4 dimensional data set.

*Data set records & attributes:

Let's try to understand the diff. types of data that we generally come across in ML problems. Data can

Attributes



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.

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

Chapter 2 Preparing to Model

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.

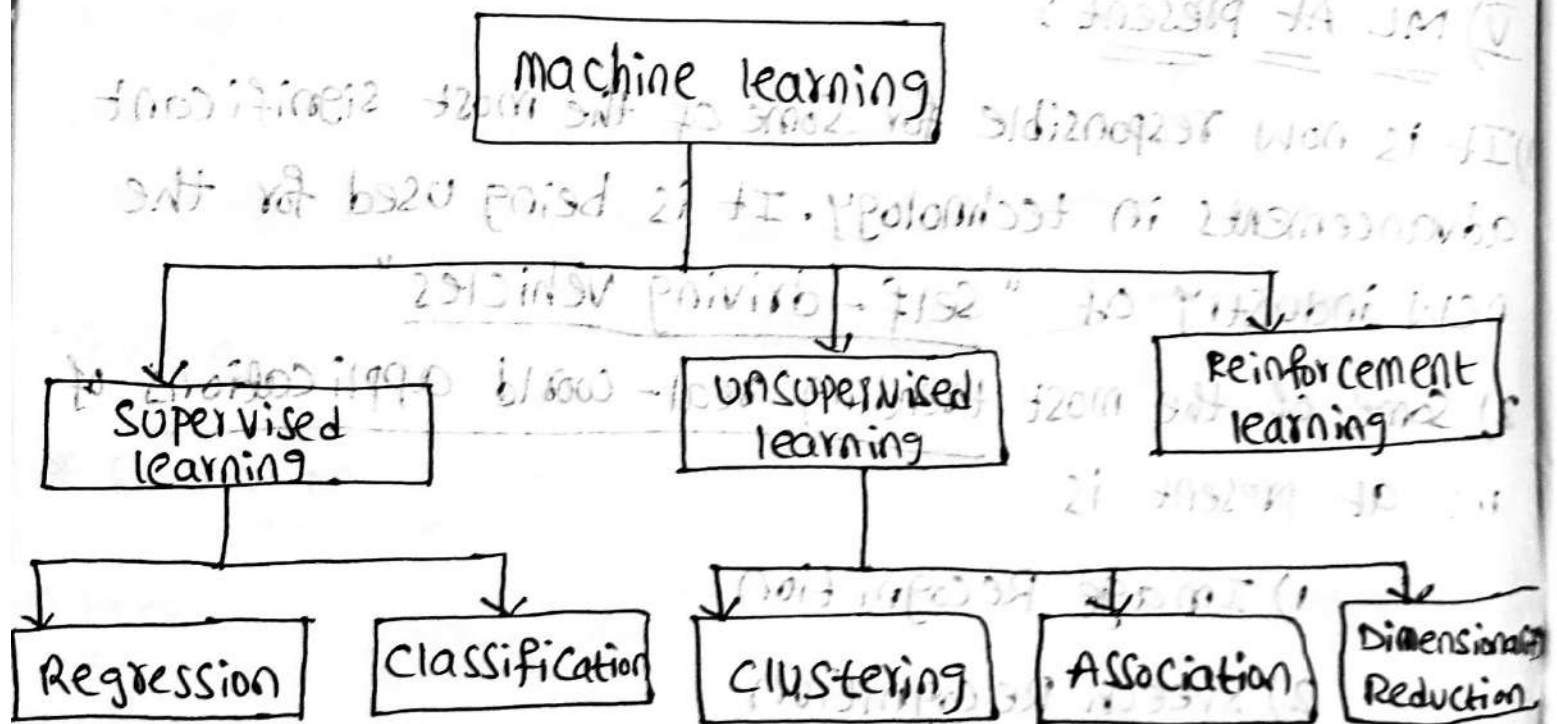
* Paradigms (or) types of ML Analysis (or) taxonomy of ML

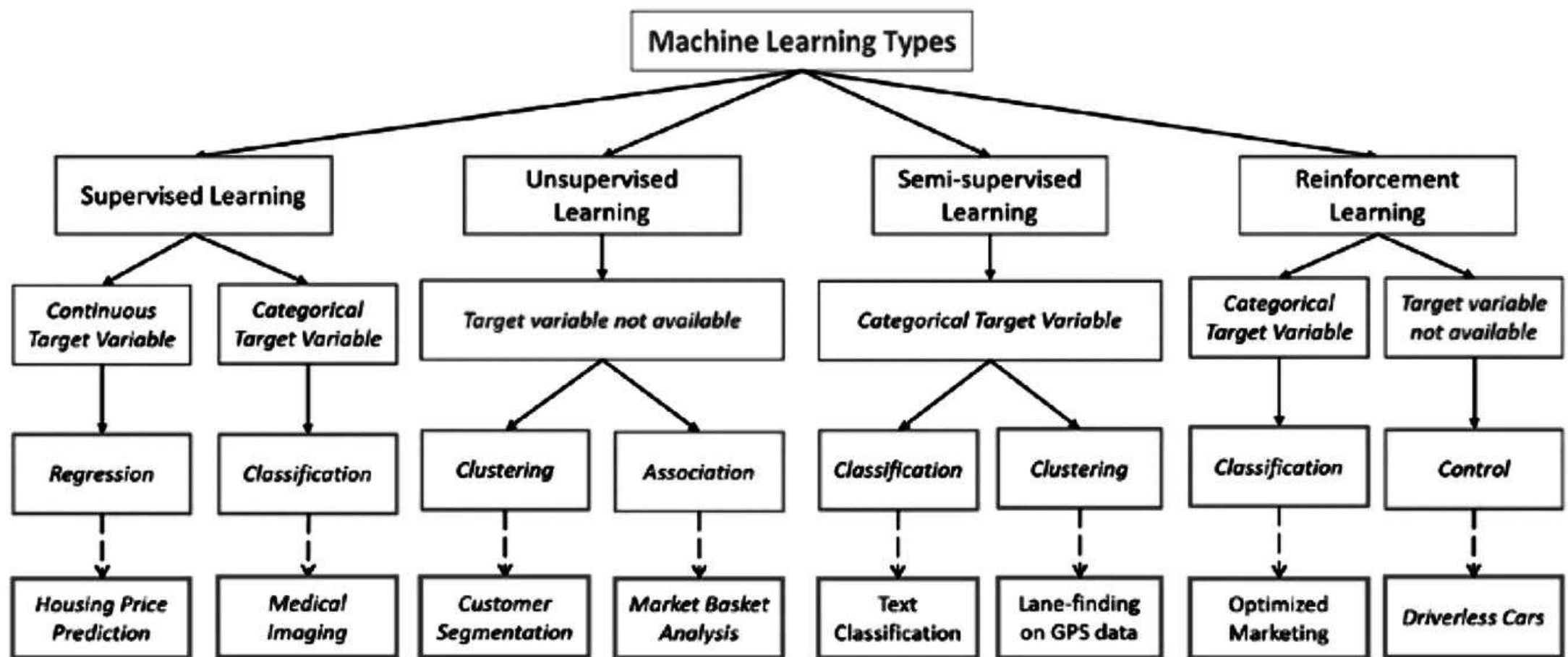
1) ML is divided into three primary learning model approaches

- 1) Supervised
- 2) Unsupervised
- 3) Reinforcement

2) Each model differs in training, Each has its strength and faces different tasks or problems.

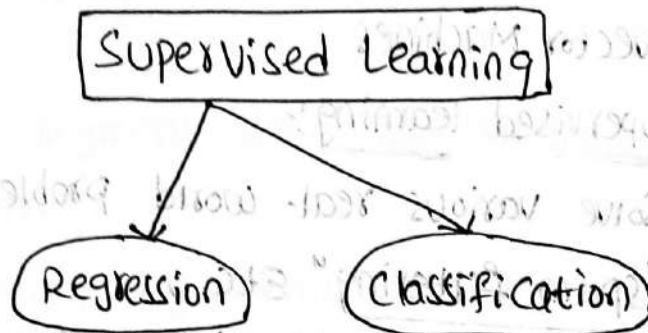
3) When choosing a ML model to deploy, an organization needs to understand the available data and the problem to be solved.





1) Supervised ML :-

- 1) Supervised ML algorithms are the most commonly used for predictive Analytics.
- 2) It requires human interaction to label data read for accurate supervised learning.
- 3) The model is taught by example using input and output datasets processed by human experts, usually data scientists.
- 4) It is commonly used for solving regression and classification problems.



(i) Regression :- It involves estimating the mathematical relationship between a continuous variable and one or more other variables.

- 2) It is used for the prediction of continuous variables such as Weather forecasting, Market Trends etc.,
- 3) Below are some popular regression algorithms

- Linear regression
- Non-linear regression
- Regression trees
- Polynomial Regression
- Bayesian linear regression

(ii) Classification:

1) Classification algorithms are used when the output variable is categorical, which means there are two classes such as Yes-No, Male-Female, True-False, Pass-fail etc.,

2) Below are some popular classification algorithms. They are

- Random Forest
- Decision Trees
- Logistic Regression
- Support Vector Machines

Advantages of supervised learning:

- 1) It helps us to solve various real-world problems such as "fraud detection", "spam filtering" etc.
- 2) It can predict the output on the basis of prior experience.

Disadvantages of supervised learning:

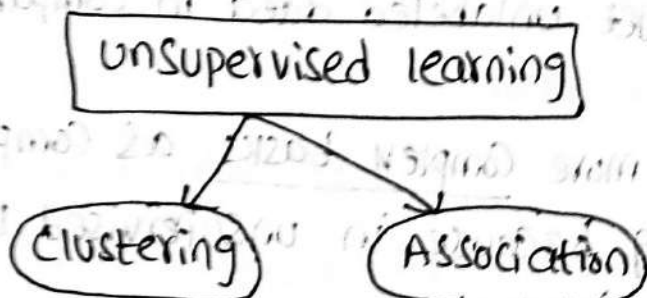
- 1) It is not suitable for handling the complex tasks.
- 2) It required lots of computation times.
- 3) It cannot predict the correct output if the test data is different from the training dataset.

② unsupervised learning:

- 1) It is a type of ML in which models are trained using unlabeled dataset and are allowed to act on that data without any supervision.
- 2) It is to find the underlying structure of dataset, group that data according to similarities and

represent that dataset in a compressed format.

- 3) It works on unlabeled and uncategorized data which make unsupervised learning more important.
- 4) It is commonly used for solving clustering methods and Association methods.



Clustering:

1) It is the grouping of data that have similar characteristics.

2) It helps segment data into groups and analyze each to find patterns.

Ex: clustering algorithms identify groups of users based on their online purchasing history and then send each member targeted ads.

Association:

1) It consists of discovering groups of items frequently observed together.

2) Online retailers use associations to suggest additional purchases to a user based on the content of their shopping cart.

unsupervised learning algorithms:

Below is the list of some popular unsupervised learning algorithms:

- K-means clustering

- KNN (K-Nearest Neighbors)

- Neural Networks

- Independent Component Analysis etc.,

Advantages of unsupervised learning:

- 1) It is easy to get unlabeled data in comparison to labeled data.
- 2) It is used for more complex tasks as compared to supervised learning because, in unsupervised learning, we don't have labeled input data.

Disadvantages of unsupervised learning:

- 1) It is intrinsically more difficult than supervised learning as it does not have corresponding output.
- 2) The result of the unsupervised learning algorithm might be less accurate as input data is not labeled.

The main difference b/w supervised and unsupervised learning are given below:

Supervised learning	Unsupervised learning
1) Supervised learning algorithms are trained using <u>labeled data</u> .	1) Unsupervised learning algorithms are trained using <u>unlabeled data</u> .
2) It takes <u>direct feedback</u> to check if it is predicting correct output or not.	2) It does <u>not take any feedback</u> .
3) Supervised learning model <u>predicts the output</u> .	3) Unsupervised learning model <u>finds the hidden patterns in data</u> .

4) It can be categorized in Classification and Regression problems

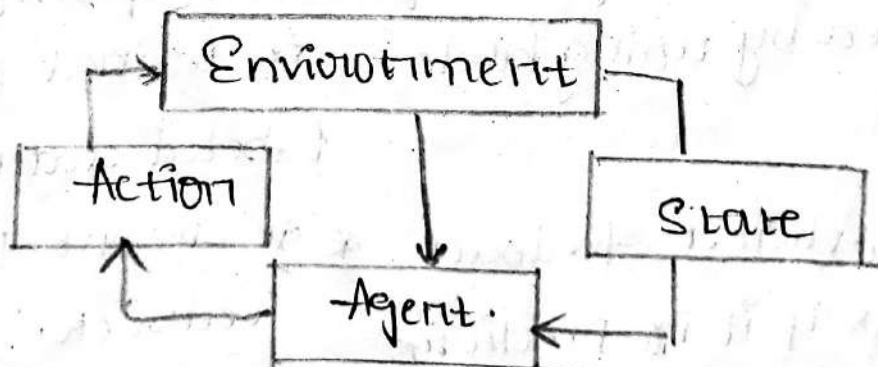
5) It produces an accurate result

4) It can be classified in Clustering & Association problems.

5) It may give less accurate result as compared to supervised learning.

Reinforcement learning:

- * It works on feedback based process in which an agent (AI) automatically explore its surrounding by hitting and trial learning from experience and improve from its experiences.
- * Agent gets rewarded for each good action and get Punish for each bad action, hence the goal of these learning agent is to maximize the rewards.
- * Its learning process is similar to a human being. Eg: a child learns various things by experience in his day-to-day life.
- * Reinforcement learning problem can be formulated using "Markov Decision Process" (MDP)
- * In MDP the agent constantly interacts with the environment and performed actions. at each action the environment responds and generates a new state.



Categories Of Reinforcement learning:

They are 2 types of Reinforcement is Categorized

→ Penalty/punishment

→ Reward

Real world use cases of reinforcement learning:

→ video games

→ resource management

→ robotics

→ text mining

Advantages of Reinforcement learning:

→ versatility:

wide range of applications like robotics, autonomous vehicles, healthcare and game playing etc.

* Scalability:

It can be scale to handle large and complex problems.

Disadvantages of Reinforcement learning:

* High computational cost

* over fitting to specific learning

NOTE: To overcome the drawbacks of supervised learning & unsupervised learning the concept of semi-supervised learning is used.

Reinforcement Learning?

In Supervised learning the ML model is learnt in such a way as to maximize a Performance measure like Prediction accuracy. In the case of reinforcement learning, an agent learns a optimal Policy to optimize some reward function. The learnt Policy helps the agent in taking an action based on the current configuration or state of the problem. Robot Path Planning is a typical applications of Reinforcement learning.

1.4 MATCHING

Matching is an important activity in ML. It is used in both supervised learning and in learning from observations. Matching is carried out by using a **proximity measure** which can be a distance/dissimilarity measure or a similarity measure. Two data items, u and v , represented as l -dimensional vectors, match better when the distance between them is smaller or when the similarity between them is larger.

A popular distance measure is the **Euclidean distance** and a popular similarity measure is the **cosine of the angle between vectors**. The Euclidean distance is given by

$$d(u, v) = \sqrt{\sum_{i=1}^l (u(i) - v(i))^2}$$

The cosine similarity is given by

$$\cos(u, v) = \frac{u^t v}{||u|| ||v||},$$

where $u^t v$ is the dot product between vectors u and v and $||u||$ is the Euclidean distance between u and the origin; it is also called the Euclidean norm.

Some of the important applications of matching in ML are in:

- **Finding the Nearest Neighbor of a Pattern:** Let x be an l -dimensional pattern vector. Let $\mathcal{X} = \{x_1, x_2, \dots, x_n\}$ be a collection of n data vectors. The nearest neighbor of x from \mathcal{X} , denoted by $NN_x(\mathcal{X})$, is x_j if

$$d(x, x_j) \leq d(x, x_i), \forall x_i \in \mathcal{X}$$

This is an approximate search where a pattern that best matches x is obtained. If there is a tie, that is, when both $x_p \in \mathcal{X}$ and $x_q \in \mathcal{X}$ are the nearest neighbors of x , we can break the tie arbitrarily or choose either of the two to be the nearest neighbor of x . This step is useful in classification and will be discussed in the next chapter.

- **Assigning to a Set with the Nearest Representative:** Let C_1, C_2, \dots, C_K be K sets with x^1, x^2, \dots, x^K as their respective representatives. A pattern x is assigned to C_i if

$$d(x, x^i) \leq d(x, x^j), \text{ for } j \in \{1, 2, \dots, K\}$$

* Matching :- It is an important activity in ML. It is used in both supervised learning and in learning from observations.

→ It is carried out by using a "proximity measure"
→ which can be a distance / dissimilarity measure or a similarity measure.

→ Two data items 'u' and 'v' represented as d-dimensional vectors, match better when the distance b/w them is smaller or when the similarity b/w them is larger.

→ A Popular distance measure is the "Euclidean distance" and a popular similarity measure is the "cosine of the angle between vectors". The Euclidean distance is given by

$$d(u, v) = \sqrt{\sum_{i=1}^d (u(i) - v(i))^2}$$

The cosine similarity is given by

$$\cos(u, v) = \frac{u^t v}{\|u\| \|v\|}$$

where $u^t v$ is the dot product b/w vectors u and v and $\|u\|$ is the Euclidean distance b/w u and the origin. It is also called "Euclidean Norm".

Some of the important applications of matching in ML are:-

1) Finding the Nearest Neighbor of a Pattern:-

Let x be an d -dimensional pattern vector.

Let $\mathcal{X} = \{x_1, x_2, x_3, \dots, x_n\}$ be a collection of n data vectors.

→ The nearest neighbor of x from \mathcal{X} , denoted by $NN_x(\mathcal{X})$ is x_j if

$$d(x, x_j) \leq d(x, x_i), \quad \forall x_i \in \mathcal{X}$$

↑ less than or equal to ↑ for all x_i belongs to the set \mathcal{X}

Importance of steps in ML:

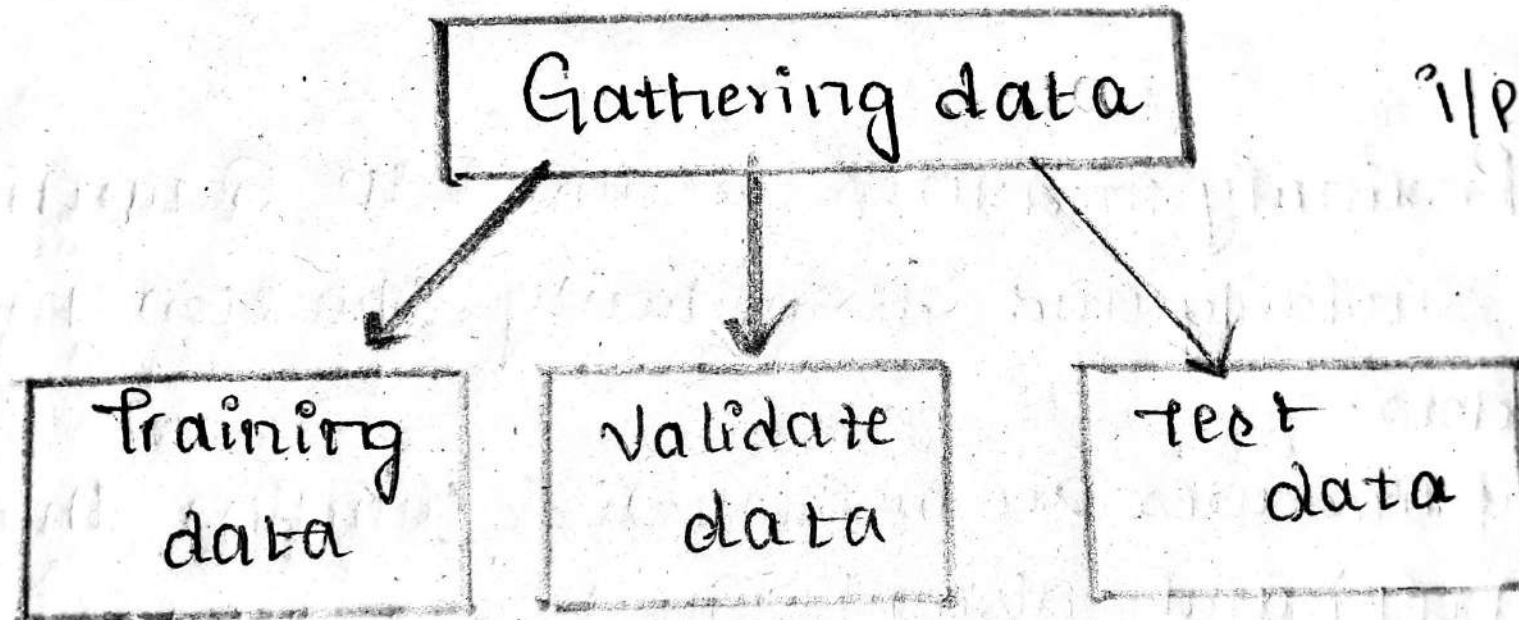
(or)

Stages in ML:

- 1) Application Domain \rightarrow Data acquisition
Feature Engineering = Data preprocessing + data representation
- 2) Model Selection \rightarrow choose a model \leftarrow Domain knowledge
- 3) Model learning \rightarrow train the model \leftarrow training data
- 4) Model Evaluation \rightarrow Evaluate the data \leftarrow validation data
- 5) Model Prediction \rightarrow learn the model \leftarrow test data
- 6) Model Explanation \rightarrow Explain the model \leftarrow Expert Feed back

Important steps in ML system:

Typically the available data is splitted into 3 Types



ifp → Meaningful data

"data" }

1.5.4 Model Selection

Selection of the model to be used to train an ML system depends upon the nature of the data and knowledge of the application domain. For some applications, only a subset of the ML models can be used. For example, if some features are numerical and others are categorical, then classifiers based on perceptrons and support vector machine (SVM) are not suitable as they compute the dot product between vectors and dot products do not make sense when some values in the corresponding vectors are non-numerical. On the other hand, Bayesian models and decision tree-based models are ideally suited to deal with such data as they depend upon the frequency of occurrence of values.

1.5.5 Model Learning

This step depends on the size and type of the training data. In practice, a subset of the labelled data is used as training data for learning the model and another subset is used for model validation or model evaluation. Some of the ML models are highly transparent while others are opaque or black box models. For example, decision tree-based models are ideally suited to provide transparency; this is because in a decision tree, at each internal or decision node, branching is carried out based on the value assumed by a feature. For example, if the height of an object is larger than 5 feet, it is likely to be an adult and not a child; such easy-to-understand rules are abstracted by decision trees. Neural networks are typically opaque as the outputs of intermediate/hidden layer neurons may not offer transparency.

1.5.6 Model Evaluation

This step is also called **model validation**. This step requires specifically earmarked data called **validation data**. It is possible that the ML model works well on the training data; then we say

that the model is well trained. However, it may not work well on the validation data. In such a case, we say that the ML model overfits the training data. In order to overcome overfitting, we typically use the validation data to tune the ML model so that it works well on both the training and validation data sets.

1.5.7 Model Prediction

This step deals with testing the model that is learnt and validated. It is used for prediction because both classification and regression tasks are predictive tasks. This step employs the test data set earmarked for the purpose. In the real world, the model is used for prediction as new patterns keep coming in. Imagine an ML model built for medical diagnosis. It is like a doctor who predicts and makes a diagnosis when a new patient comes in.

1.5.8 Model Explanation

This step is important to explain to an expert or a manager why a decision was taken by the ML model. This will help in explicit or implicit feedback from the user to further improve the model. Explanation had an important role earlier in expert systems and other AI systems. However, explanation has become very important in the era of DL. This is because DL systems typically employ neural networks that are relatively opaque. So, their functioning cannot be easily explained at a level of detail that can be appreciated by the domain expert/user. Such opaque behaviour has created the need for explainable AI.

1.6 SEARCH AND LEARNING

Search is a very basic and fundamental operation in both ML and AI. Search had a special role in conventional AI where it was successfully used in problem solving, theorem proving, planning and knowledge-based systems.

Further, search plays an important role in several computer science applications. Some of them are as follows:

- Exact search is popular in databases for answering queries, in operating systems for operations like *grep*, and in looking for entries in symbol tables.
- In ML, search is important in learning a classification model, a proximity measure for clustering and classification, and the appropriate model for regression.
- Inference is search in logic and probability. In linear algebra, matrix factorization is search. In optimization, we use a regularizer to simplify the search in finding a solution. In information theory, we search for purity (low entropy).

So, several activities including optimization, inference and matrix factorization that are essential for ML are all based on search. Learning itself is search. We will examine how search aids learning of each ML model in the respective chapters.

1.7 EXPLANATION OFFERED BY THE MODEL

Conventional AI systems were logic-based or rule-based systems. So, the corresponding reasoning systems naturally exhibited transparency and, as a consequence, explainability. Both forward and

backward reasoning was possible. In fact, the same knowledge base, based on experts' input, was used in both diagnosis and in teaching because of this flexibility. Specifically, the knowledge base used by the MYCIN expert system was used in tutoring medical students using another expert system called GUIDON.

However, there were some problems associated with conventional AI systems:

- There was no general framework for building AI systems. Acquiring knowledge, using additional heuristics and dealing with exceptions led to adhocism; experience in building one AI system did not simplify the building of another AI system.
- Acquiring knowledge was a great challenge. Different experts typically differed in their conclusions, leading to inconsistencies. Conventional logic-based systems found it difficult to deal with such inconsistent knowledge.

There has been a gradual shift from using knowledge to using data in building AI systems. Current-day AI systems, which are mostly based on DL, are by and large data dependent. They can learn representations automatically. They employ variants of multi-layer neural networks and backpropagation algorithms in training models.

Some difficulties associated with DL systems are:

- They are data dependent. Their performance improves as the size of the data set increases. So, they need larger data sets. Fortunately, it is not difficult to provide large data sets in most of the current applications.
- Learning in DL systems involves a simple change of weights in the neural network to optimize the objective function. This is done with the help of backpropagation, which is a gradient-descent algorithm and which can get stuck with a locally optimal solution. Combining this with large data sets may possibly lead to overfitting. This is typically avoided by using a variety of simplifications in the form of regularizers and other heuristics.
- A major difficulty is that DL systems are black box systems and lack explanation capability. This problem is currently attracting the attention of AI researchers.

We will be discussing how each of the ML models is equipped with explanation capability in the respective chapters.

1.8 DATA SETS USED

In this book, we make use of two data sets to conduct experiments and present results in various chapters. These are:

• Data Sets for Classification

1. *MNIST Handwritten Digits Data Set:*

- There are 10 classes (corresponding to digits 0, 1, ..., 9) and each digit is viewed as an image of size 28×28 ($= 784$) pixels; each pixel having values in the range 0 to 255.
- There are around 6000 digits as training patterns and around 1000 test patterns in each class and the class label is also provided for each of the digits.
- For more details, visit <http://yann.lecun.com/exdb/mnist/>

2. *Fashion MNIST Data Set:*

- It is a data set of Zalando's article images, consisting of a training set of 60,000 examples and a test set of 10,000 examples.

- Each example is a 28×28 greyscale image, associated with a label from 10 classes.
- It is intended to serve as a possible replacement for the original MNIST data set for benchmarking ML models.
- It has the same image size and structure of training and testing splits as the MNIST data.
- For more details, visit <https://www.kaggle.com/datasets/zalando-research/fashionmnist>

3. Olivetti Face Data Set:

- It consists of 10 different images each of 40 distinct subjects. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses).
- All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement).
- Each image is of size $64 \times 64 = 4096$.
- It is available on the scikit-learn platform.
- For more details, visit https://ai.stanford.edu/~marinka/nimfa/nimfa.examples.orl_images.html

4. Wisconsin Breast Cancer Data Set:

- It consists of 569 patterns and each is a 30-dimensional vector.
- There are two classes *Benign* and *Malignant*. The number of patterns from *Benign* is 357 and the number of *Malignant* class patterns is 212.
- It is available on the scikit-learn platform.
- For more details, visit https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_breast_cancer.html

• Data Sets for Regression

1. Boston Housing Data Set:

- It has 506 patterns.
- Each pattern is a 13-dimensional vector.
- It is available on the scikit-learn platform.
- For more details, visit https://scikit-learn.org/0.15/modules/generated/sklearn.datasets.load_boston.html

2. Airline Passengers Data Set:

- This data set provides monthly totals of US airline passengers from 1949 to 1960.
- This data set is taken from an inbuilt data set of Kaggle called AirPassengers.
- For more details, visit <https://www.kaggle.com/datasets/chirag19/air-passengers>

3. Australian Weather Data Set:

- It provides various weather record details for cities in Australia.
- The features include location, min and max temperature, etc.
- For more details, visit <https://www.kaggle.com/datasets/arunavakrchakraborty/australia-weather-data>

• Unit-2

• MODELLING AND EVALUATION & BASICS OF FEATURE ENGINEERING

• Part-1

INTRODUCTION

The basic learning process, irrespective of the fact that the learner is a human or a machine, can be divided into three parts:

- Data Input
- Abstraction
- Generalization

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.

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.

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.

The same thing can be done using machine learning too.

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**.

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 an exhaustive 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**.

• SELECTING A MODEL

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 $X_1, X_2, X_3, \dots, X_n$ 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.

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.

There are three broad categories of machine learning approaches used for resolving different types of problems.

They are

- 1. **Supervised**
 - Classification
 - Regression
- 2. **Unsupervised**
 - Clustering
 - 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. 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.

Machine learning algorithms are broadly of two types: models for supervised learning, which primarily focus on solving predictive problems and models for unsupervised learning, which solve descriptive problems.

- **1.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, and the predictor features.

The predictive models have a clear focus on what they want to learn and how they want to learn.

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

- Predicting win/loss in a cricket match
- Predicting whether a transaction is fraud
- 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 the predictor features.

Below are some examples:

- Prediction of revenue growth in the succeeding year
- Prediction of rainfall amount in the coming monsoon
- Prediction of potential flu patients and demand for flu shots next winter

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 and Logistic Regression models are popular regression models.

• **2.Descriptive models**

Models for unsupervised learning or descriptive models are used to describe a data set or gain insight from a data set.

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

- Customer grouping or segmentation based on social, demographic, ethnic, etc. factors
- Grouping of music based on different aspects like genre, language, time-period, etc.
- 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.

3. Holdout method

In case of supervised learning, a model is trained using the labelled input data. 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.

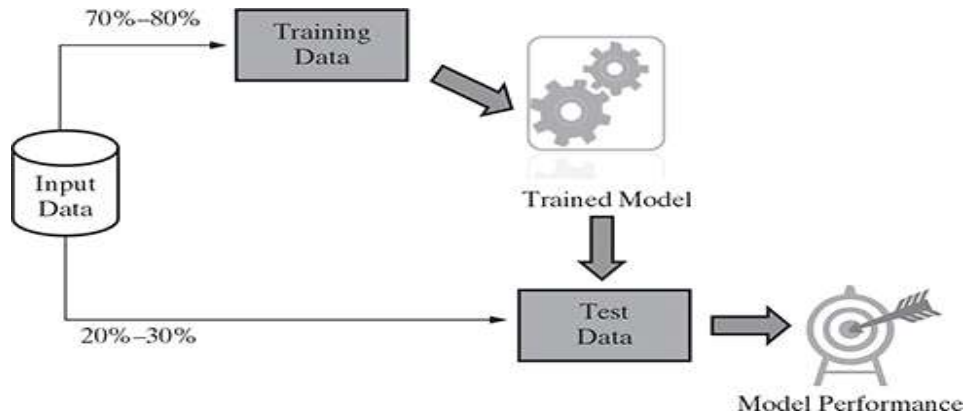


FIG. 3.1 Holdout method

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.

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.

- **4.K-fold Cross-validation method**

Holdout method employing stratified random sampling approach still heads into issues in certain specific situations..

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:

- 10-fold cross-validation (10-fold CV)
- 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., the records in a fold are drawn by using random sampling technique.

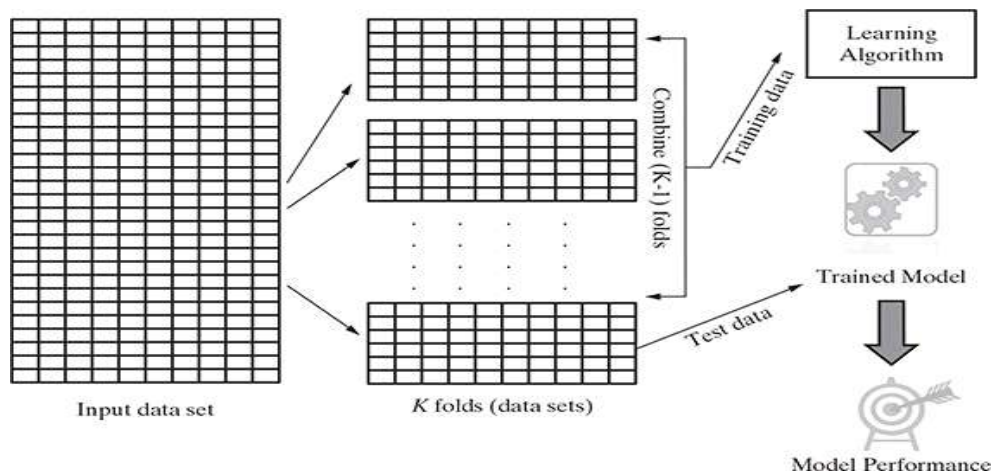


FIG. 3.2 Overall approach for K -fold cross-validation

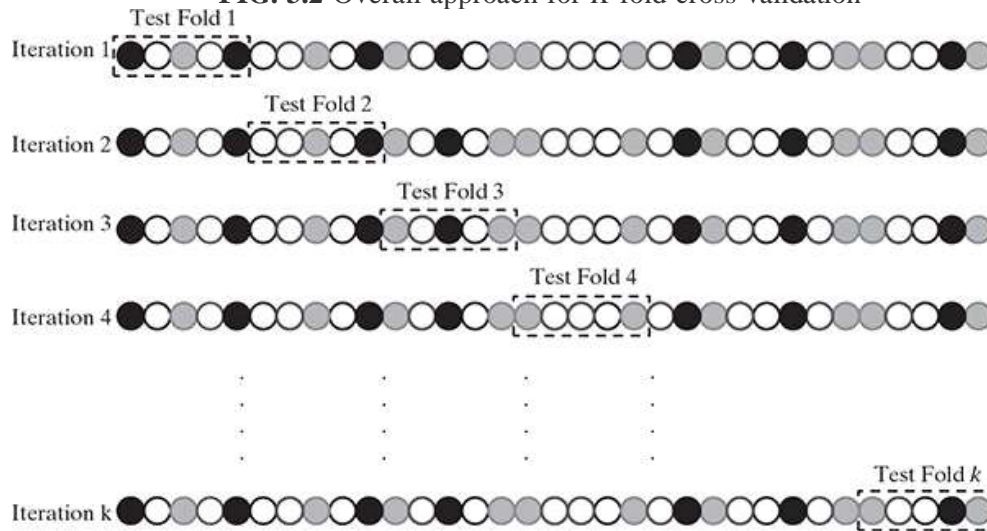


FIG. 3.3 Detailed approach for fold selection

Leave-one-out cross-validation (LOOCV) is an extreme case of k -fold cross-validation using one record or data instance at a time as a test data. This is done to maximize the count of data used to train the model.

- **5.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 random samples.

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.

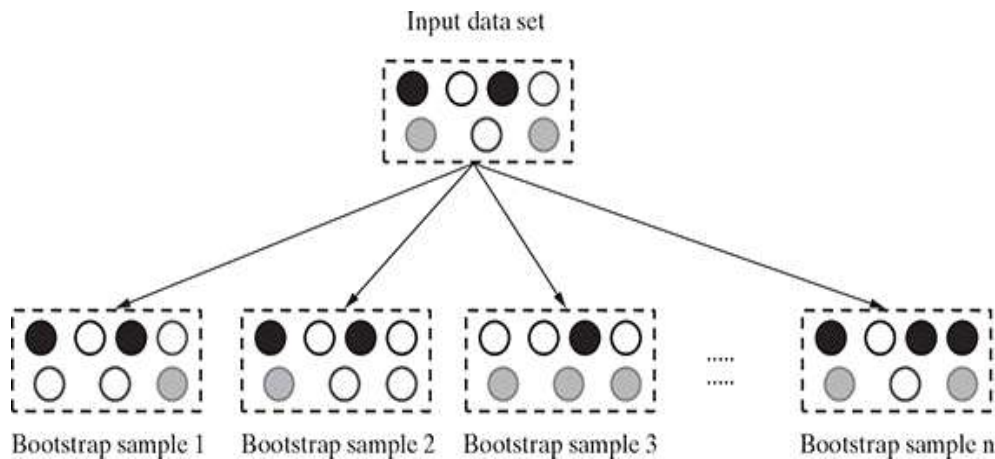


FIG. 3.4 Bootstrap sampling

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 refer back to the training data.

Eager learners take more time in the learning phase than the lazy learners.

Some of the algorithms 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, 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 not much of training actually happens.

One of the most popular algorithm for lazy learning is k -nearest neighbor.

MODEL REPRESENTATION AND INTERPRETABILITY

- **1.Underfitting**

If the target function is kept too simple, it may not be able to capture the essential nuances and represent the underlying data well.

A typical case of underfitting may occur when trying to represent a non-linear data with a linear 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 avoided by

- using more training data
- reducing features by effective feature selection

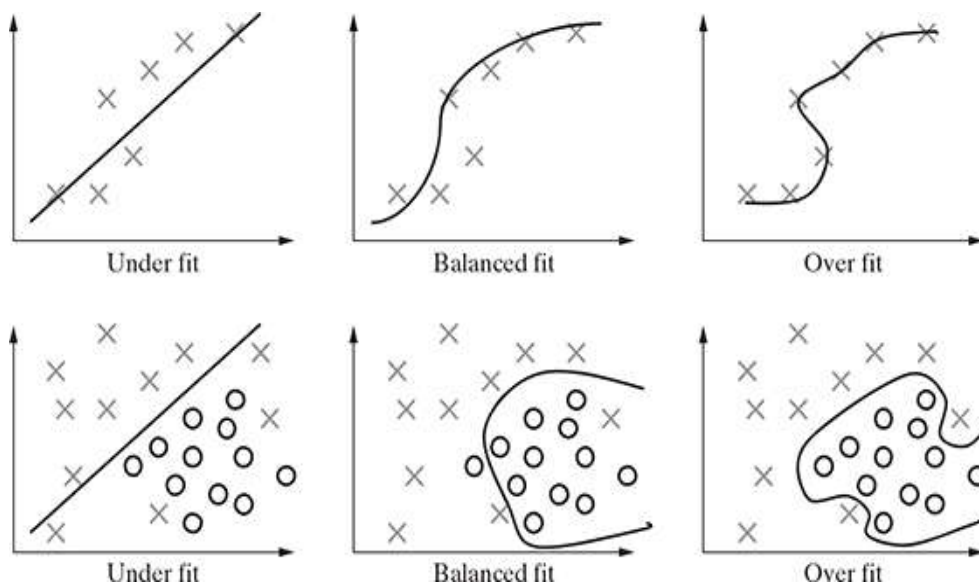


FIG. 3.5 Underfitting and Overfitting of models

• 2.Overfitting

Overfitting refers to a situation where the model has been designed in such a way that it emulates the training data too closely.

In such a case, any specific deviation in the training data, like noise or outliers, gets embedded in the 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 figure 3.5 .

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 target function results in wrong classification in the test data set.

Overfitting results in good performance with training data set,

but poor generalization and hence poor performance with test data set. Overfitting can be avoided by

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

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

- **3.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.

- **1.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.

- **2.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 used to 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 closely matches the training data, even a small difference in training data gets magnified in the model.

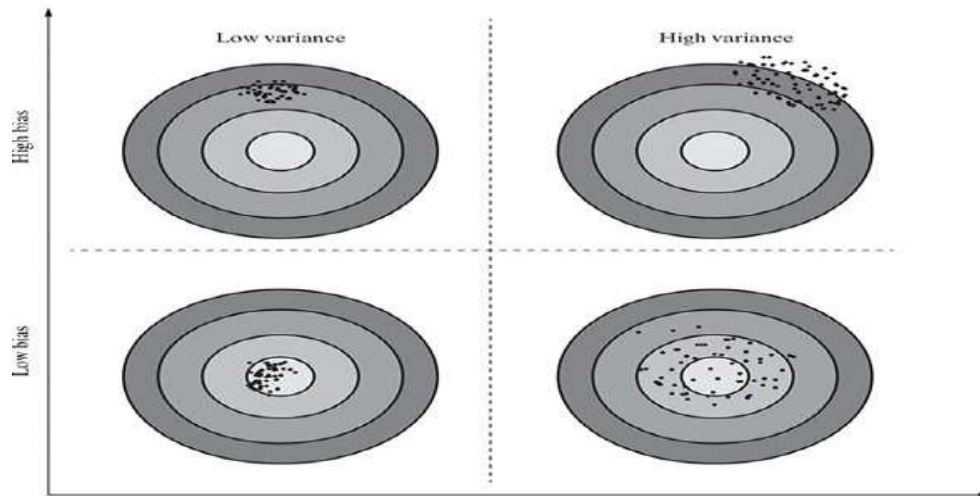


FIG. 3.6 Bias-variance trade-off

So, the problems in training a model can either happen because either (a) the model is too simple and hence fails to interpret the data grossly or

(b) the model is extremely complex and magnifies even small differences in the training data.i.e.,

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

On one hand, parametric algorithms are generally seen to 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.

For example, in a popular supervised algorithm k -Nearest Neighbors or k NN, 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**

- **1.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.

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 predicted by 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 times the model has classified correctly, then the model accuracy is said to be 99%.

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 cricket match win/loss prediction:

- the model predicted win and the team won
- the model predicted win and the team lost
- the model predicted loss and the team won
- 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.

These cases 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 are referred as False Positive (FP) cases.

The third case, i.e. the model predicted loss and the team 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.

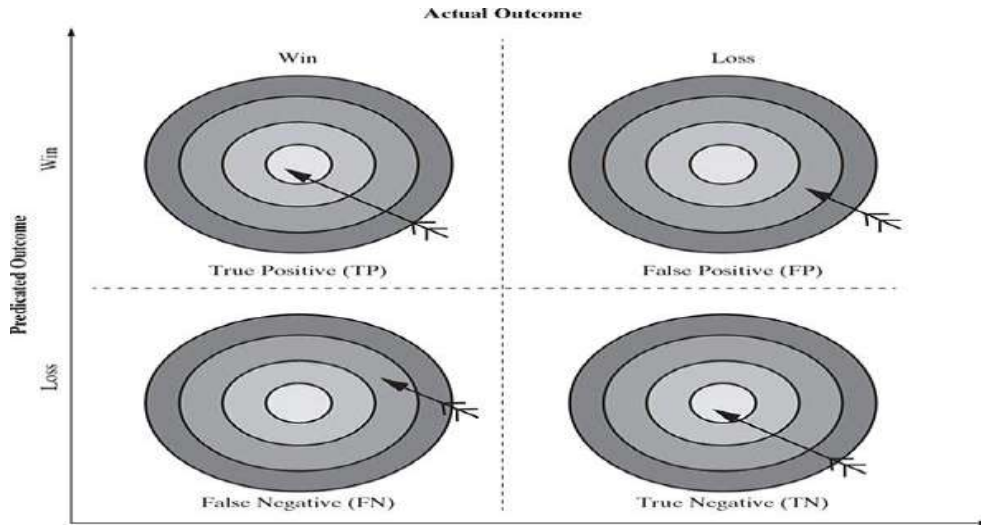


FIG. 3.7 Details of model classification

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 cases are depicted in Figure 3.7 .

For any classification model, **model accuracy** is given by total number of correct classifications (either as the 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.

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

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 that reason 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/loss prediction 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.

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

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

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

In context of the above confusion matrix,

$$\begin{aligned} \text{Error rate} &= \frac{\text{FP} + \text{FN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} = \frac{4 + 2}{85 + 4 + 2 + 9} = \frac{6}{100} = 6\% \\ &= 1 - \text{Model accuracy} \end{aligned}$$

Sometimes, correct prediction, both TPs as well as TNs, may happen by mere coincidence.

Kappa value of a model indicates the adjusted the 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.

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

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

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

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

The **sensitivity** of a model measures the proportion of 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 the cricket match win prediction problem,

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

Sensitivity measure gives the proportion of tumours which are actually malignant and have been predicted as malignant. A high value of sensitivity is more desirable than a high value of accuracy.

Specificity is also another good measure to indicate a good balance of a model being excessively conservative or excessively aggressive.

Specificity of a model measures the proportion of negative examples which have been correctly classified.

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

A higher value of specificity will indicate a better model performance.

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.

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

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,

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

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.

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

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

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

- ***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

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

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

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

As a combination of multiple measures into one, *F*-score gives the right measure using which performance of different models can be compared.

- ***Receiver operating characteristic (ROC) curves***

Receiver Operating Characteristic (ROC) curve helps in visualizing the performance of a classification model.

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

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

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.

The area under curve (AUC) value, as shown in figure 3.8a , is the area of the two-dimensional space under the curve extending from (0, 0) to (1, 1). AUC value ranges 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 of classifier 1 is more than the AUC of classifier 2. So, we can draw the inference that classifier 1 is better than classifier 2.

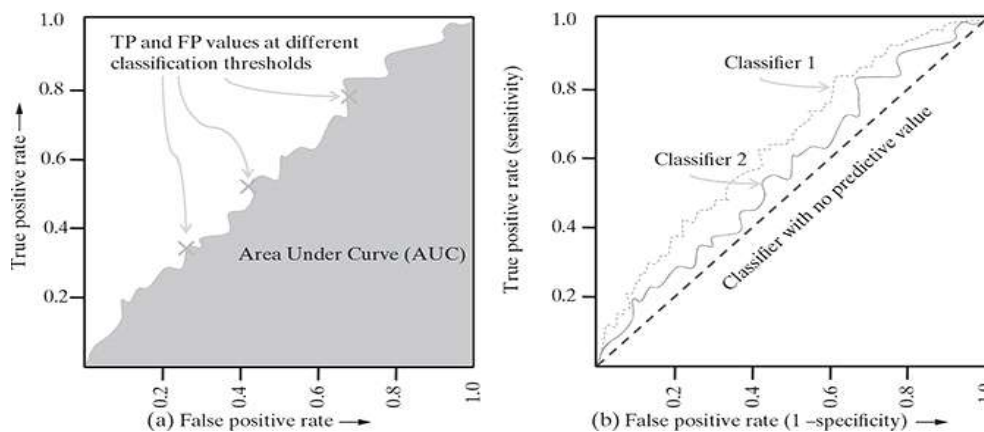


FIG. 3.8 ROC curve

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

- 0.5 – 0.6 → Almost no predictive ability
- 0.6 – 0.7 → Weak predictive ability
- 0.7 – 0.8 → Fair predictive ability
- 0.8 – 0.9 → Good predictive ability
- 0.9 – 1.0 → Excellent predictive ability

- **Supervised learning – regression**

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:

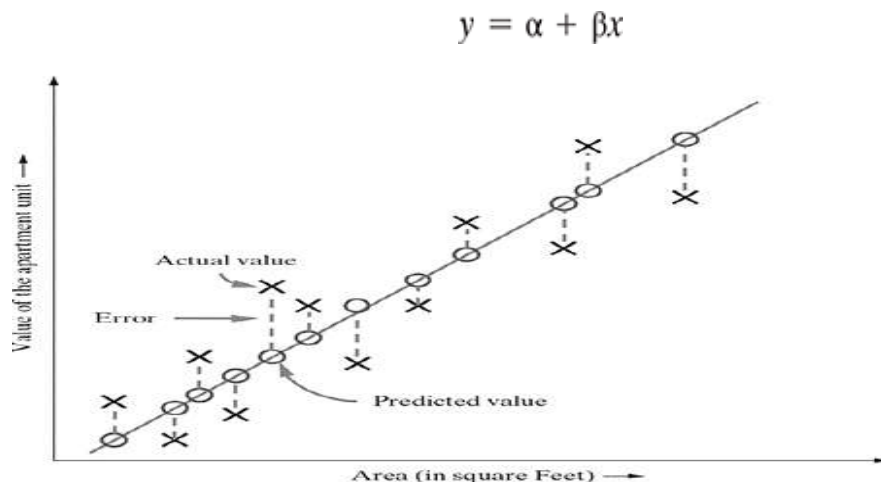


FIG. 3.9 Error – Predicted vs. actual value

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 \hat{y} 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.

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

A clustering algorithm is successful if the clusters identified using the algorithm is able to achieve the right results in the overall problem domain.

There are couple of popular approaches which are adopted for cluster quality evaluation.

- *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 -th data instance and data instances of all other clusters.

Let's try to understand this in context of the example depicted in figure 3.10. 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} , ..., a_{in_1} 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,

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

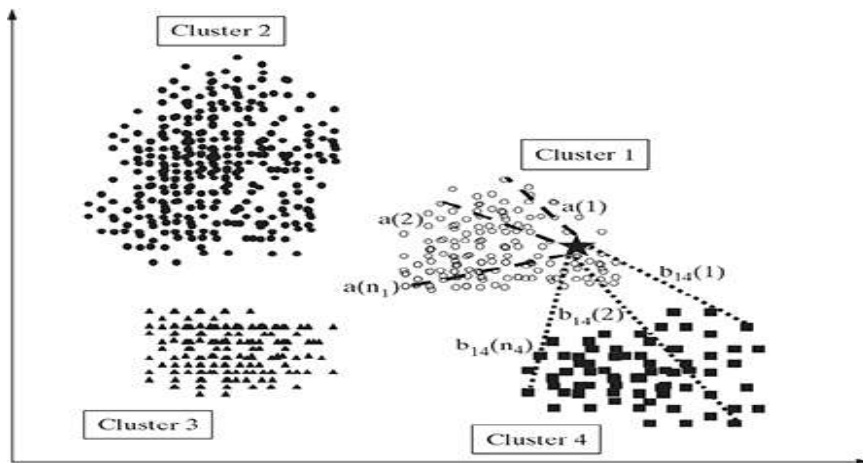


FIG. 3.10 Silhouette width calculation

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,

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

where n_4 is the total number of elements in cluster 4. In the same way, we can calculate the values of $b_{12}(\text{average})$ and $b_{13}(\text{average})$. $b(i)$ is the minimum of all these values. Hence, we can say that,

$$b(i) = \text{minimum} [b_{12}(\text{average}), b_{13}(\text{average}), b_{14}(\text{average})]$$

- *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:

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

• IMPROVING PERFORMANCE OF A MODEL

Model selection is done on several aspects:

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

- Type of the data, i.e. categorical or numeric
- Sometimes on the problem domain
- 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 (k NN), 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.

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, k NN, etc.

The outputs from the different models are combined using a combination function.

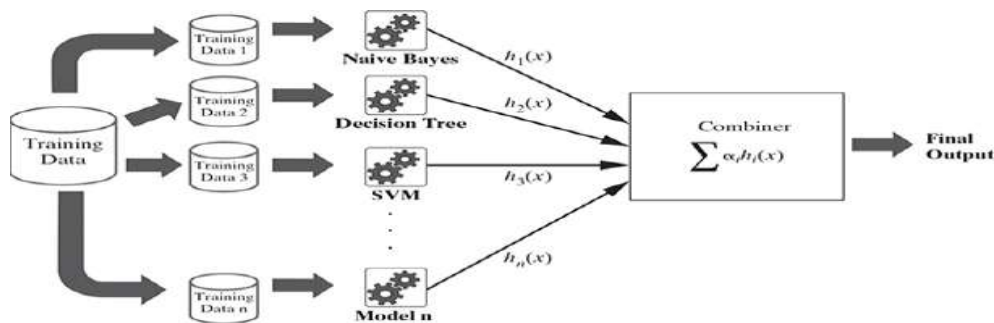


FIG. 3.11 Ensemble

One of the earliest and most popular ensemble models is **bootstrap aggregating** or **bagging**.

Bagging uses bootstrap sampling method 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

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 of boosting 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.

•

Part-2

Basics of Feature Engineering

• INTRODUCTION

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.

We need to touch upon another key aspect which plays a critical role in solving any machine learning problem – feature engineering.

Feature engineering is a critical preparatory process in machine learning.

It is responsible for taking raw input data and converting that to well-aligned features which are ready to be used by the machine learning models.

- What is a feature?

A feature is an attribute of a data set that is used in a machine learning process.

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.

Feature engineering is an important pre-processing step for machine learning. It has two major elements:

- feature transformation
- 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:

- feature construction
- 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 in a data set, after feature construction ‘ m ’ more features or dimensions 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 of features 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 job of feature selection is to derive a subset F_j (F_1, F_2, \dots, F_m) of F_i (F_1, F_2, \dots, F_n), where $m < n$, such that F_j is most meaningful and gets the best result for a machine learning problem.

- **FEATURE TRANSFORMATION**

Engineering a good feature space is a crucial prerequisite for 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 certain domains e.g. medical image classification, text categorization, etc.

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 feature transformation:

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

- **1.Feature construction**

Feature construction involves transforming a given set of input features to generate a new set of more powerful features.

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
80	59	23,60,000
54	45	12,15,000
78	56	21,84,000
63	63	19,84,000
83	74	30,71,000
92	86	39,56,000

apartment_ length	apartment_ breadth	apartment_ area	apartment_ price
80	59	4,720	23,60,000
54	45	2,430	12,15,000
78	56	4,368	21,84,000
63	63	3,969	19,84,500
83	74	6,142	30,71,000
92	86	7,912	39,56,000

FIG. 4.2 Feature construction (example 1)

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

- *2.Encoding categorical (nominal) variables*

Let's take the example of another data set on athletes, as presented in [Figure 4.3a](#).

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.

Any machine learning algorithm, whether it's a classification algorithm (like k NN) 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 row had 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)

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 one feature 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.

- **3.Encoding categorical (ordinal) variables**

Let’s take an example of a student data set. Let’s assume that 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, and D.

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 is mapped 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)

- ***4.Transforming numeric (continuous) features to categorical 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. The numerical feature 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](#).

- ***5.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 an important area of research – not only for technology practitioners but also for industry practitioners.

All machine learning models need numerical data as input. So the text data in the data sets need to be transformed into numerical features.

apartment_area	apartment_price	apartment_area	apartment_grade
4,720	23,60,000	4,720	Medium
2,430	12,15,000	2,430	Low
4,368	21,84,000	4,368	Medium
3,969	19,84,500	3,969	Low
6,142	30,71,000	6,142	High
7,912	39,56,000	7,912	High

(a) (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 to the corpus are consolidated in the form of bag-of-words. There are three major steps that are followed:

- tokenize
- count
- 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 in a 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 commonly used operators for combining the original features include

- For Boolean features: Conjunctions, Disjunctions, Negation, etc.
- For nominal features: Cartesian product, M of N, etc.
- 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 $F_i (F_1, F_2, \dots, F_n)$. After feature extraction using a mapping function $f (F_1, F_2, \dots, F_n)$ say, we

$\dot{F}_i(\dot{F}_1, \dot{F}_2, \dots, \dot{F}_m)\dot{F}_i = f(F_i)$ will have a set of features such that and $m < n$. For example, $\dot{F}_1 = k_1 F_1 + k_2 F_2$. This is depicted in Figure 4.7.

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		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 × Feat _A + 0.9 × Feat _A			
Feat ₂ = Feat _A + 0.5 Feat _B + 0.6 × Feat _C			

FIG. 4.7 Feature extraction

The most popular feature extraction algorithms used in machine learning are:

- **1. Principal Component Analysis**

Every data set, 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.

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.

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

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

- First, calculate the covariance matrix of a data set.
- Then, calculate the eigenvalues of the covariance matrix.
- 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.

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

• **2.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:

$$A = U\Sigma V$$

where, U and V are orthonormal matrices, U is an $m \times m$ unitary matrix, V is an $n \times n$ unitary matrix and Σ is an $m \times n$ rectangular diagonal matrix. The diagonal entries of Σ 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:

- Patterns in the attributes are captured by the right-singular vectors, i.e. the columns of V .
- Patterns among the instances are captured by the left-singular, i.e. the columns of U .
- Larger a singular value, larger is the part of the matrix A that it accounts for and its associated vectors.
- New data matrix with ' k ' attributes is obtained using the equation

$$D' = D \times [v_1, v_2, \dots, v_k]$$

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

• **3.Linear Discriminant Analysis**

Linear discriminant analysis (LDA) is another commonly used 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:

- Calculate the mean vectors for the individual classes.
- Calculate intra-class and inter-class scatter matrices.
- Calculate eigenvalues and eigenvectors for S_W^{-1} and S_B , where S_W is the intra-class scatter matrix and S_B is the inter-class scatter 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, m_i is the sample mean for each class, m is the overall mean of the data set, N_i is the sample size of each class

- 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.

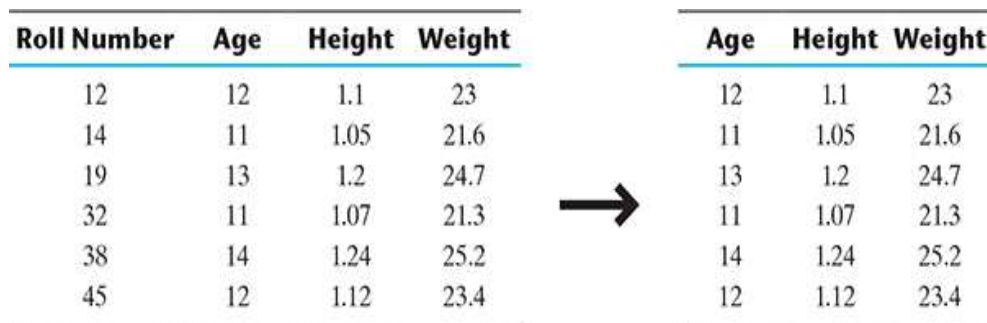
Let's quickly discuss a practical example to understand the philosophy behind feature selection.

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.

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 better results than the full set. The same has been **depicted in Figure 4.8.**



The diagram illustrates feature selection. On the left is a table with 4 columns: Roll Number, Age, Height, and Weight. An arrow points to the right, where a second table is shown with 3 columns: Age, Height, and Weight. The 'Roll Number' column has been removed from the second table, as it is deemed irrelevant for predicting weight.

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

Age	Height	Weight
12	1.1	23
11	1.05	21.6
13	1.2	24.7
11	1.07	21.3
14	1.24	25.2
12	1.12	23.4

FIG. 4.8 Feature selection

The issues which have made feature selection such a relevant problem to be solved are

- 1. 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.

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**

- ***1.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 or 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.

Let take a simple example of the student data set. 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.

- ***2.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, 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.

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 a small number of representative features out of a set of potentially redundant features are considered for being a part of the final feature subset.

The main objective of feature selection is to remove all features which are irrelevant and take a representative subset of the features which are potentially redundant.

This leads to a meaningful feature subset in context of a specific learning task.

- **Measures of feature relevance and redundancy**

- ***1.Measures of feature relevance***

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 is that 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.

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 $p(f = x)$.

- **2.Measures of Feature redundancy**

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

- Correlation-based measures
- Distance-based measures, and
- Other coefficient-based measure

- **a.Correlation-based similarity measure**

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

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

$$cov(F_1, F_2) = \sum (F_{1_i} - \bar{F}_1).(F_{2_i} - \bar{F}_2)$$

$$var(F_1) = \sum (F_{1_i} - \bar{F}_1)^2, \text{ where } \bar{F}_1 = \frac{1}{n} \cdot \sum F_{1_i}$$

$$var(F_2) = \sum (F_{2_i} - \bar{F}_2)^2, \text{ where } \bar{F}_2 = \frac{1}{n} \cdot \sum F_{2_i}$$

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.

• ***b.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:

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

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 is the **Minkowski distance**, measured as

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

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 more frequently 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.

• *Other similarity measures*

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

0	1	1	0	1	0	1	1
1	1	0	0	1	0	0	1

(a) Hamming distance measurement

0	1	1	0	1	0	1	0
1	1	0	0	1	0	0	0

(b) Jaccard coefficient measurement

0	1	1	0	1	0	1	0
1	1	0	0	1	0	0	0

(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, $d_J = 1 - J$

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.

$$\text{Jaccard coefficient of } F_1 \text{ and } F_2, J = \frac{n_{11}}{n_{01} + n_{10} + n_{11}} = \frac{2}{1 + 2 + 2} = \frac{2}{5} \text{ or } 0.4.$$

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

2.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 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

n_{00} = number of cases where both the features have value 0 Quite

understandably, the total count of rows, $n = n_{00} + n_{01}$

+ $n_{10} + n_{11}$. As shown in Figure 4.10c, all values have been included in the calculation of SMC.

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

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.

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:

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

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

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

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, $x \cdot y = 2*2 + 4*1 + 0*0 + 0*0 + 2*3 + 1*2 + 3*1 + 0*0 + 0*1 = 19$

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

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

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

Cosine similarity actually measures the angle (refer to Fig.

- 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° .

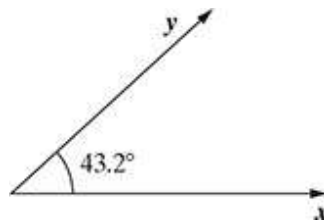


FIG. 4.11 Cosine similarity

• 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 typical feature selection process consists of four steps:

- generation of possible subsets
- subset evaluation
- stop searching based on some stopping criterion
- validation of the result

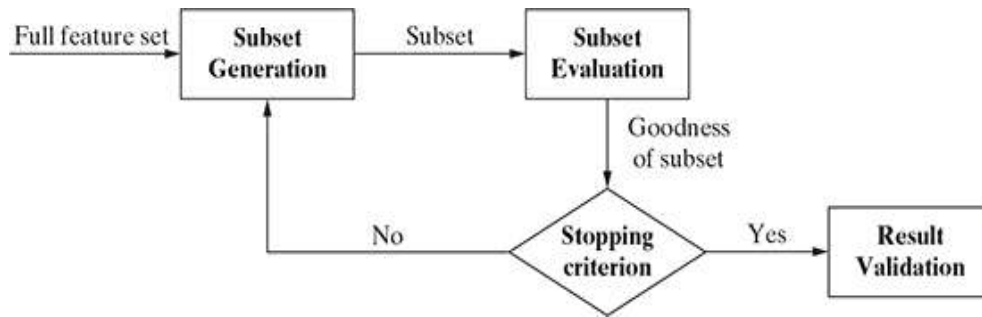


FIG. 4.12 Feature selection process

Subset generation, which is the first step of any feature selection algorithm, is a search procedure which ideally 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. In certain 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 compared with 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

- the search completes
- some given bound (e.g. a specified number of iterations) is reached
- subsequent addition (or deletion) of the feature is not producing a better subset
- 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:

- Filter approach
- Wrapper approach
- Hybrid approach
- 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.

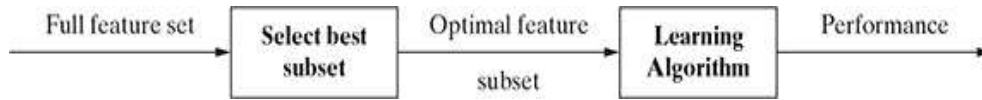


FIG. 4.13 Filter approach

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.

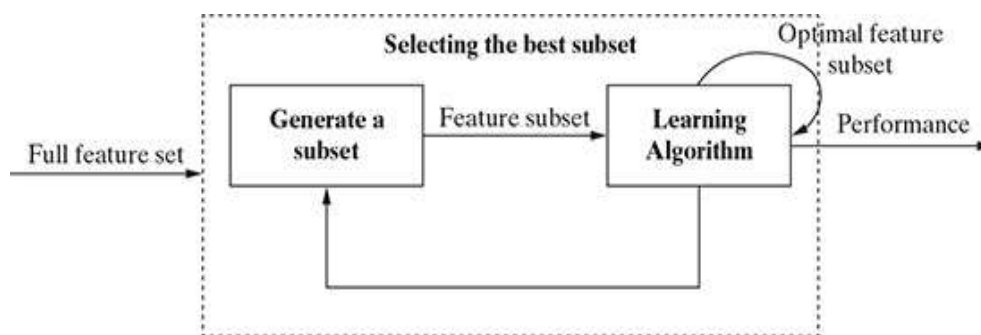


FIG. 4.14 Wrapper approach

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 an inductive algorithm to evaluate the generated feature subsets. However, the difference is it performs feature selection and classification simultaneously.

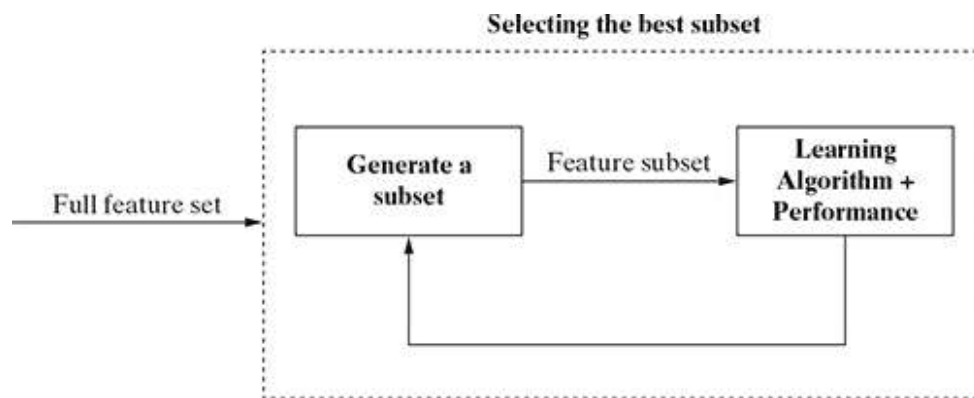


FIG. 4.15 Embedded approach

-
-
-
- **Important Points to Remember**

A feature is an attribute of a data set that is used in a machine learning process. Feature engineering is an important pre-processing step for machine learning, having two major elements:

- feature transformation
- feature subset selection

Feature transformation transforms data into a new set of features which can represent the underlying machine learning problem

There are two variants of feature transformation:

- feature construction
- feature extraction

Feature construction process discovers missing information about the relationships between features and augments the feature space by creating additional features. Feature extraction is the process of extracting or creating a new set of features from the original set of features using some functional mapping. Some popular feature extraction algorithms used in machine learning:

- Principal Component Analysis (PCA)
- Singular Value Decomposition (SVD)
- Linear Discriminant Analysis (LDA)

Feature subset selection is intended to derive a subset of features from the full feature set. No new feature is generated.

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

Feature selection intends to remove all features which are irrelevant and take a representative subset of the features which are potentially redundant. This leads to a meaningful feature subset in context of a specific learning task.

Feature relevance is indicated by the information gain from a feature measured in terms of relative entropy.

Feature redundancy is based on similar information contributed by multiple features measured by feature-to-feature:

- Correlation
- Distance (Minkowski distances, e.g. Manhattan, Euclidean, etc. used as most popular measures)
- Other coefficient-based (Jaccard, SMC, Cosine similarity, etc.)

Main approaches for feature selection are

- Filter
- Wrapper
- Hybrid
- Embedded

- Jaccard coefficient vs. SMC

Bayesian Concept Learning.

Bayesian theorem provides the basis for Machine learning.

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.

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 hypothesis 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

The MAP hypothesis output is used 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 $\{(x_1, t_1), (x_2, t_2), \dots, (x_m, t_m)\}$, where x_i is the instance of x and t_i is the target concept of x_i defined as $t_i = c(x_i)$.

We can assume that the Sequence of instances of x $\{x_1, \dots, x_m\}$ is held fixed and then, the Sequence of target values becomes $T = \{t_1, \dots, t_m\}$.

For Calculating the highest posterior probability, we can use Bayes's 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 hmap with the highest posterior probability

$$h_{map} = \underset{h \in H}{\operatorname{argmax}} P(h/T)$$

Let us try to Connect the Concept learning problem with the Problem of identifying the hmap

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 & independent of each

On the 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

$$P(h) = \frac{1}{|H|} \text{ for all } h \text{ within } H \quad \dots \dots (3)$$

$P(T|h)$ is the probability of observing the target values t_i in the fixed set of instances $\{x_1, \dots, x_m\}$ in the Space where h holds true and describes the Concept C correctly. Using assumptions 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:

$$P(T|h) = \begin{cases} 1 & \text{if } t_i = h(x_i) \text{ for all } t_i \text{ within } T \\ 0 & \text{Otherwise} \end{cases} \quad \dots \dots (4)$$

Using Bayes's theorem to identify the posterior probability

$$P(h|T) = \frac{P(T|h)P(h)}{P(T)} \quad \dots \dots (5)$$

For the Cases when h is inconsistent with the training data T , using (5) we get

$$P(h|T) = \frac{0 \times P(h)}{P(T)} = 0, \text{ when } h \text{ is inconsistent with}$$

' T ' And when h is consistent with T

$$P(h|T) = \frac{1 \times \frac{1}{|H|}}{P(T)} = \frac{1}{|H|P(T)} \quad \dots \dots (6)$$

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

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

This makes as (6)

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

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

$$P(h|T) = \begin{cases} \frac{1}{|H_D|} & \text{if } h \text{ is Consistent with } T \\ 0 & \text{Otherwise} \end{cases} \dots (7)$$

where H_D is the number of hypothesis 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 hypothesis become zero. And the total probability that sums up to 1 is distributed equally among the consistent hypothesis in the set. So, under this condition, each consistent hypothesis is a MAP hypothesis with posterior probability $\frac{1}{|H_0|}$.

• Concept of Consistent learners

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.

Bayes's theorem can characterize the behaviour of learning algorithms even when the algorithm does not explicitly manipulate the probability.

• Bayes Optimal Classifier

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 0.4, 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 weighted 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

$$\text{Bayes optimal Classifier} = \left[\arg \max_{c_i \in C} \sum_{h_i \in H} P(c_i|h_i) p(h_i|T) \right]$$

the basis of the combined predictions of all alternative hypotheses, weighted by their posterior probabilities.

So, extending the above Example,

$$\begin{matrix} h_1 = \text{True} \\ h_2 \& h_3 = \text{False} \end{matrix}$$

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

$$\begin{aligned} p(h_1|T) &= 0.4, p(\text{False}|h_1) = 0, p(\text{True}|h_1) = 1, p(h_2|T) = 0.3, \\ p(\text{False}|h_2) &= 1, p(\text{True}|h_2) = 0, p(h_3|T) = 0.3, p(\text{False}|h_3) \\ &= 1, p(\text{True}|h_3) = 0 \end{aligned}$$

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

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

$$\begin{aligned} &\text{combine 2} \\ &h_2 \& h_3 = 0.6 \end{aligned}$$

(4)

arg max.

$$c_i \in \{\text{True}, \text{false}\} \sum_{h_i \in H} p(c_i | h_i) p(h_i | T) = \text{false}$$

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 .

Posterior probability = (Prior probability \times Conditional probability)

Evidence.

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

Parameter Estimation for Naive Bayes models uses the method of ML.

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

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

Let us see the basis of deriving the principle of Naive 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 example for target function, and the set of attributes $\{a_1, a_2, \dots, a_n\}$ for the new instance are known to us.

Our task is to predict the classification of the new instance.

According to the approach in Baye's 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.

$\{a_1, a_2, \dots, a_n\}$. So, $C_{MAP} = \underset{c_i \in C}{\operatorname{argmax}} \sum_{h_i \in H} p(c_i | a_1, a_2, \dots, a_n)$

which can be rewritten using Baye's theorem as

$$C_{MAP} = \underset{c_i \in C}{\operatorname{argmax}} \sum_{h_i \in H} \frac{p(a_1, a_2, \dots, a_n | c_i) p(c_i)}{p(a_1, \dots, a_n)}$$

(5)

$$C_{MAP} = \operatorname{argmax}_{c_i \in C} \sum_{h_i \in H} \frac{p(a_1, a_2, \dots, a_n | c_i) p(c_i)}{p(a_1, \dots, a_n)}$$

As Combined Probability of the attributes defining the new instance fully is always 1

$$C_{MAP} = \operatorname{argmax}_{c_i \in C} \sum_{h_i \in H} p(a_1, a_2, \dots, a_n | c_i) p(c_i) \dots (8)$$

So, to get the most probable classifier, we have to evaluate the two terms $p(a_1, a_2, \dots, a_n | c_i)$ and $p(c_i)$.

In a practical scenario, it is possible to calculate $p(c_i)$ by calculating the frequency of each target value c_i in the training dataset. But the $p(a_1, a_2, \dots, a_n | c_i)$ 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.

So, applying this simplification, we can now say that for a target value of an instance, the probability of observing the combination a_1, a_2, \dots, a_n is the product of probabilities of individual attributes $p(a_i | c_j)$.

$$p(a_1, a_2, \dots, a_n | c_j) = \prod p(a_i | c_j)$$

Then, from Equation (7), we get the approach for the Naive Bayes Classifier as:

$$C_{NB} = \operatorname{argmax}_{c_i \in C} \sum_{h_i \in H} p(c_i) \prod_i p(a_i | c_j) \dots (9)$$

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 the Naive Bayes Classifier approach is that it is not required to Search the whole hypothesis Space for this Algorithm.

A Naive 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) to gauge the parameters (mean and differences of the variables) Essential for the Classification (arrangement). In the Naive Bayes Classifier, independent variable for each class should be determined and not the whole Covariance matrix. Because of the rather naive assumption that all features of the dataset are Equally important and independent, this is called Naive Bayes.

Strengths	Weakness
Simple and fast in calculation but yet effective in result.	The basic 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 no. 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 no. 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.	

Table - 1 Strengths and weakness of Bayes classifier.

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. 3) for actual match outcome, while four parameters are considered - weather Condition (Rainy, Overcast, or Sunny), how many matches, or three matches, Humidity Condition (High or Normal), and whether they won the toss (True or false). Using Naive 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, then won two of the last three matches, Humidity

weather Condition	wins/lost 3 Matches	Humidity	wintoss	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: 3 Training data for the Naive Bayesian Method.

Naive Bayes Classifier Steps

Step 1: First Construct a frequency table. A frequency table is drawn for each attribute against the target outcomes.

For Example, in figure(3) the various attributes are (1) weather Condition, (2) How many matches, won by

(7)

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.

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

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

$P(\text{yes})$ will give the overall probability of favourable Condition in the given Scenario.

• Solving the above problem with Naive 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, won match is 'yes', is, $3/(3+4+2)$
 $= 3/9$

Figure (4) Shows the frequency table thus Constructed.

Step2: To predict whether the team will win for given weather
1. Condition (a_1) = Rainy, wins in last three matches (a_2)
= 2 wins, Humidity (a_3) = Normal and win loss (a_4) = True
we need to Choose 'yes' from the above table for the
given Conditions.

From Bayes' theorem, we get

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

This Equation becomes much easier to resolve if we recall that Naive 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 Equ. Significantly as

$$\begin{aligned} P(\text{win match} | a_1, a_2, a_3, a_4) &= \\ \frac{P(a_1 | \text{win match}) P(a_2 | \text{win match}) P(a_3 | \text{win match}) P(a_4 | \text{win match}) P(\text{win match})}{P(a_1) P(a_2) P(a_3) P(a_4)} \\ &= \frac{2}{9} * \frac{4}{9} * \frac{6}{9} * \frac{9}{14} = 0.016109347. \end{aligned}$$

won match			won match		
weather condition	yes	No	Humidity	yes	No
Sunny	3	2	High	3	4
overcast	4	0	Normal	6	1
Rainy	2	3			
Total	9	5	Total	9	5

won match			won match		
wins in last 3 matches	yes	No	wint-loss	yes	No
3 wins	2	2	FALSE	6	2
1 win 2 wins	4	2	TRUE	3	3
2 win 1 win	3	1			
Total	9	5	Total	9	5

fig 4 Construct frequency table

Step 3: by normalizing the above two probabilities, we can

Ensure that the Sum of these two probabilities is 1.

$$\begin{aligned}
 P(\text{win match}) &= \frac{P(\text{win match})}{P(\text{win match}) + P(!\text{win match})} \\
 &= \frac{0.014109347}{0.014109347 + 0.010285714} \\
 &= 0.578368999
 \end{aligned}$$

$$\begin{aligned}
 P(\text{! win match}) &= \frac{P(\text{! win match})}{P(\text{win match}) + P(\text{! win match})} \\
 &= \frac{0.010285714}{0.014109347 + 0.010285714} \\
 &= 0.421631001
 \end{aligned}$$

Conclusion: This shows that there is 58% probability that the team will win if the above conditions become true for that particular day. Thus, Naive Bayes classifier provides a simple yet powerful way to consider the influence of multiple attributes.

- Applications of Naive Bayes Classifier.

Text classification: Naive 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 has various applications in document categorization, language detection, and sentiment detection, which are very useful for traditional retailers, e-retailers.

Spam filtering: Spam filtering is the best known use of Naive Bayesian text classification. Presently, almost all the

the email providers have this as a built-in functionality, which makes use of a Naive Bayes Classifier to identify Spam Email on the basis of certain conditions and also the probability of classifying an email as 'spam'.

Naive Bayesian Spam Sifting has turned into a mainstream mechanism to recognize illegitimate a spam email from an honest to goodness email (Sometimes called 'ham').

Hybrid Recommender System: it uses Naive Bayes Classifier and Collaborative filtering. Recommender Systems (used by e-retailers like eBay, Alibaba, Target, Flipkart, etc..) apply ML 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 person. One of the Algorithms is combining a Naive Bayes classification approach with collaborative filtering, and experimental results show that this Algorithm provides better performance regarding accuracy and coverage than other algorithms.

Online Sentiment Analysis: - The online Applications use Supervised ML (Naive Bayes) and useful Computing. In the case of Sentiment Analysis, let us assume there are three sentiments such as nice, nasty, or neutral, and Naive Bayes

Classifier is used to distinguish b/w them.

Simple Emotion modelling Combines a statistically based classifier with a dynamical model.

The Naïve Bayes Classifier Employs 'Single words' and 'word Pairs' like features and determines the Sentiments of the users. It allocates user utterance into nice, nasty, and neutral classes, labelled as +1, -1, and 0, respectively.

- Handling Continuous Numeric Features in Naïve Bayes Classifier.

In the above Example, we saw that the Naïve Bayes classifier model uses a frequency table of the training data for its Calculation.

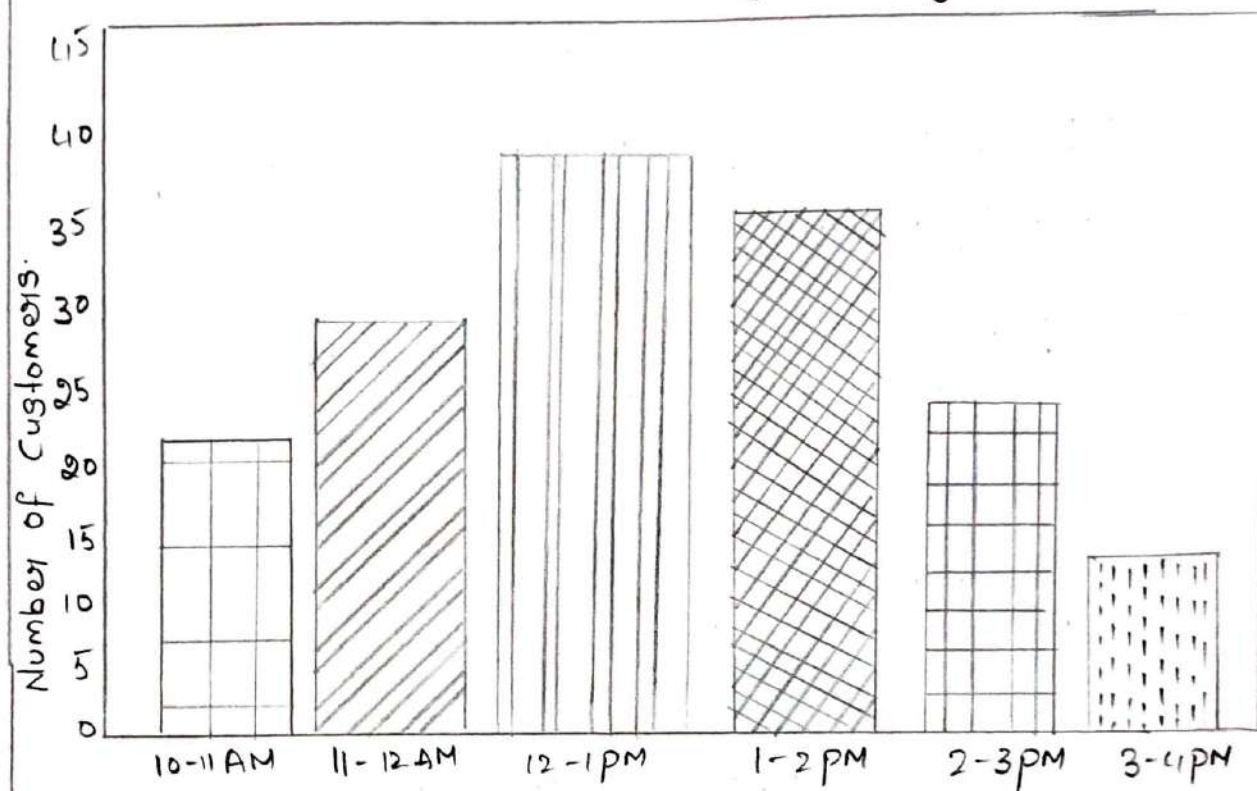
Thus, Each attribute data should be categorical in nature so, that the Combination of Classes and feature values can be Created. But this is not possible in the case of Continuous numeric data as it does not have the Categories of data.

If we plot the number of Customers visiting the bank during the 8 hrs 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 new Credit Card.

The bins created by categorizing the customers by their (name) time of Entry looks like fig(5).

This creates eight natural bins for us (or we may change the number of bins by changing our categorizing criteria), which can now be used for Bayes analysis.



Banking hour

Fig: 5 The distribution of bins based on the time of Entry of Customers in the bank.

- Bayesian Belief Network.

In Naive Bayes Classifier the attribute values a_1, a_2, \dots, a_n are Conditionally independent for a target value.

The Naive Bayes Classifier generates optimal output when this condition is met. Bayesian Belief network, which assumes that within the set of attributes, the probability distribution can have conditional probability relationship

as well as Conditional independence assumptions.

This is different from the Naive Bayes assumption of Conditional independence of all the attributes.

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:

$$P(A|k) = \frac{P(A, k)}{P(k)} \quad \dots (60)$$

Re arranging (9), we get the product rule.

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

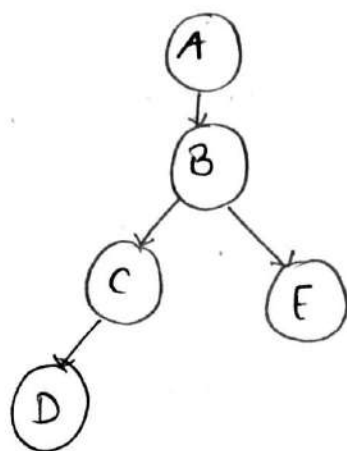
This can be extended for three variables or attributes as

$$P(A, k, c) = P(A|k, c) P(k, c) = P(A|k, c) P(k|c) P(c)$$

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

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

This generalized version of the product rule is called the Chain Rule.



Fig(6) Chain rule.

Let us understand the Chain rule by using the diagram in Fig(6). From the joint probability formula (10), we can 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 Fig(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 Eqn as

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

• 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 said to 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 said to 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, \dots, A_n is Conditionally independent of the Set of variables B_1, B_2, \dots, B_m given the set of variables C_1, C_2, \dots, C_l if

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

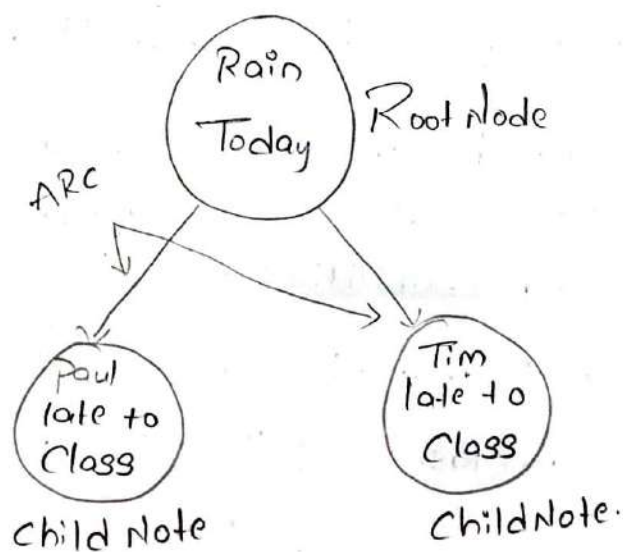
if we Compare this definition with our assumption in the Naive Bayes Classifier, we see that the Naive Bayes Classifier assumes that the instance attribute A_1 is Conditionally independent of the instance attribute A_2 , given the target value v , which can be written using the general product rule and 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 joint Space. In fig(7), a Bayesian Belief network is presented. 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 N/w
 if two variables A and B are connected 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 fig(7)



Rain Today (Root Node) probability	
True	0.1
False	0.9

Node	Rain Today	
	True	False
Paul late to class	True 0.6	0.5
	False 0.4	0.5

Node	Rain Today	
	True	False
Tim late to class	True 0.8	0.1
	False 0.2	0.9

fig(7) Bayesian belief N/w.

• The uncod.

1. The unconditional probability that Tim is late to class -

$$\begin{aligned} - P(\text{Tim is late to class}) &= P(\text{Tim late} | \text{Rain Today})P(\text{Rain Today}) \\ &+ P(\text{Tim late} | \text{No Rain Today}) * P(\text{No Rain Today}) \\ &= (0.8 \times 0.1) + (0.1 \times 0.9) = 0.17 \end{aligned}$$

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.

2. The revised probability that there was rain today.

$$\begin{aligned} P(\text{Rain Today} | \text{Tim late to class}) &= \frac{P(\text{Tim late} | \text{Rain today}) * P(\text{Rain today})}{P(\text{Tim late})} \\ &= \frac{(0.8 \times 0.1)}{0.17} = 0.47 \end{aligned}$$

= $P(\text{Rain Today})$ as Tim late to class is already known

3. The revised probability that Paul will be late to class today $P(\text{Paul late to class today})$

$$= P(\text{Paul late} | \text{Rain today}) * P(\text{Rain today}) + P(\text{Paul late} | \text{No rain today}) * P(\text{No rain today})$$

$$= (0.6 \times 0.47) + (0.5 \times (1 - 0.47))$$

$$= 0.55$$

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 have 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 n/w Can represent much more complex Scenarios with dependence and independence Concepts. There are three types of Connections possible in a Bayesian Belief n/w.

Diverging Connection: In this type of Connection, the Evidence can be transmitted b/w two child nodes of the same parent provided that the parent is not instantiated. In fig 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: (8) for illustration)

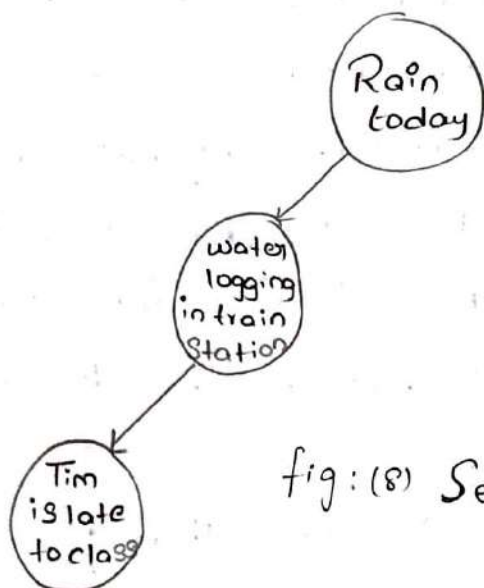
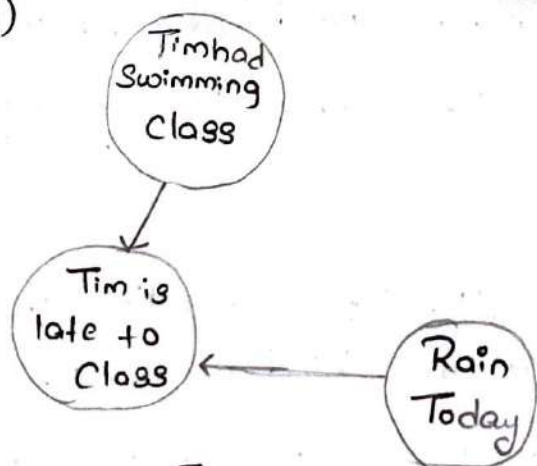


fig: (8) Serial Connection.

Converging Connection: In this type of Connection, the Evidence can only be transmitted b/w two parents when the child (converging) node has received some evidence and that evidence can be soft or hard (See fig: (9) illustration)



fig(9): Convergent Connection.

• Use of the Bayesian Belief n/w in ML.

Bayesian n/w 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 n/w 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 detection.

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 quite effective.

In other words, the Bayesian n/w is a mechanism for automatically (biology) applying Bayes' theorem to complex problems.

Bayesian n/w's are used for modelling beliefs in domains like computational biology & bioinformatics such as protein structure and gene regulatory n/w's, medicines, forensics, document classification, information

Retrieval, Image Processing, Decision Support Systems,
Sports betting and gaming, Property market analysis &
various other fields.

Example of Supervised learning

In Supervised learning, the labelled training data provide the basis for learning.

Training data is the past information with known value of class field or 'label'. Hence, we say that the training data is labelled. In the case of Supervised learning 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.

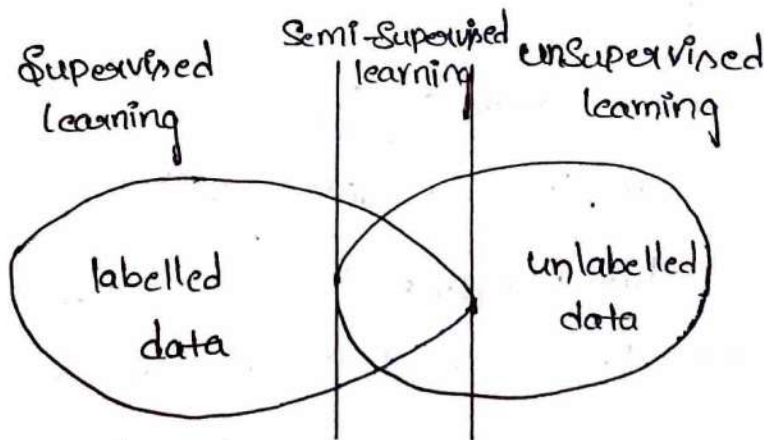


fig.7.1

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.

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 unsupervised 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.

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 a prediction. However, for tumour prediction, we are trying to predict with category or class, i.e. 'malignant' or 'benign', an unknown input data related 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 nominal variable, the problem is known as a classification problem. A classification problem is one where the output variable is a category such as

'red' & 'blue' & 'malignant tumour' & 'benign ^{Part-2} tumour', etc. ②
 whereas when we are trying to predict a numerical variable
 such as 'price', 'weight', etc. the problem falls under the category
 of regression.

Supervised Learning : Classification

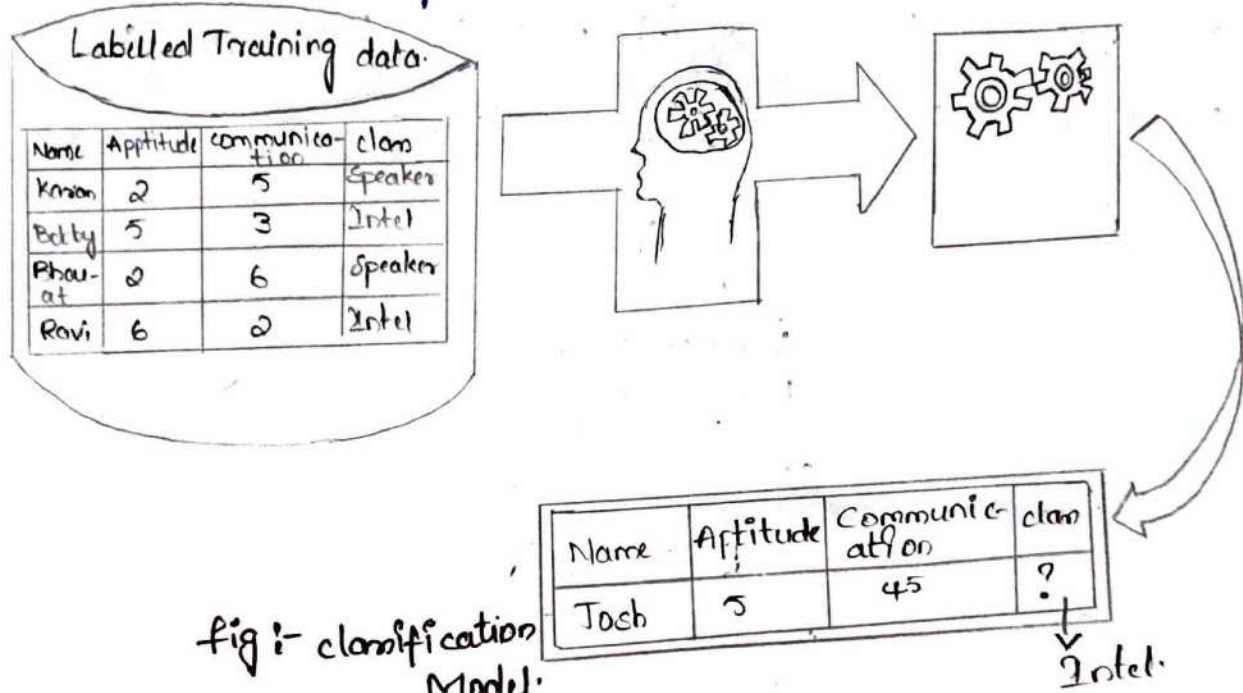


fig:- classification Model.

Classification model

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 classified categorical feature is known as class.

Some typical classification problems include the following:

- Image classification
- Disease prediction
- coin-loss prediction of games
- prediction of natural calamity such as Earthquake, flood, etc.
- Hand writing recognition.

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)

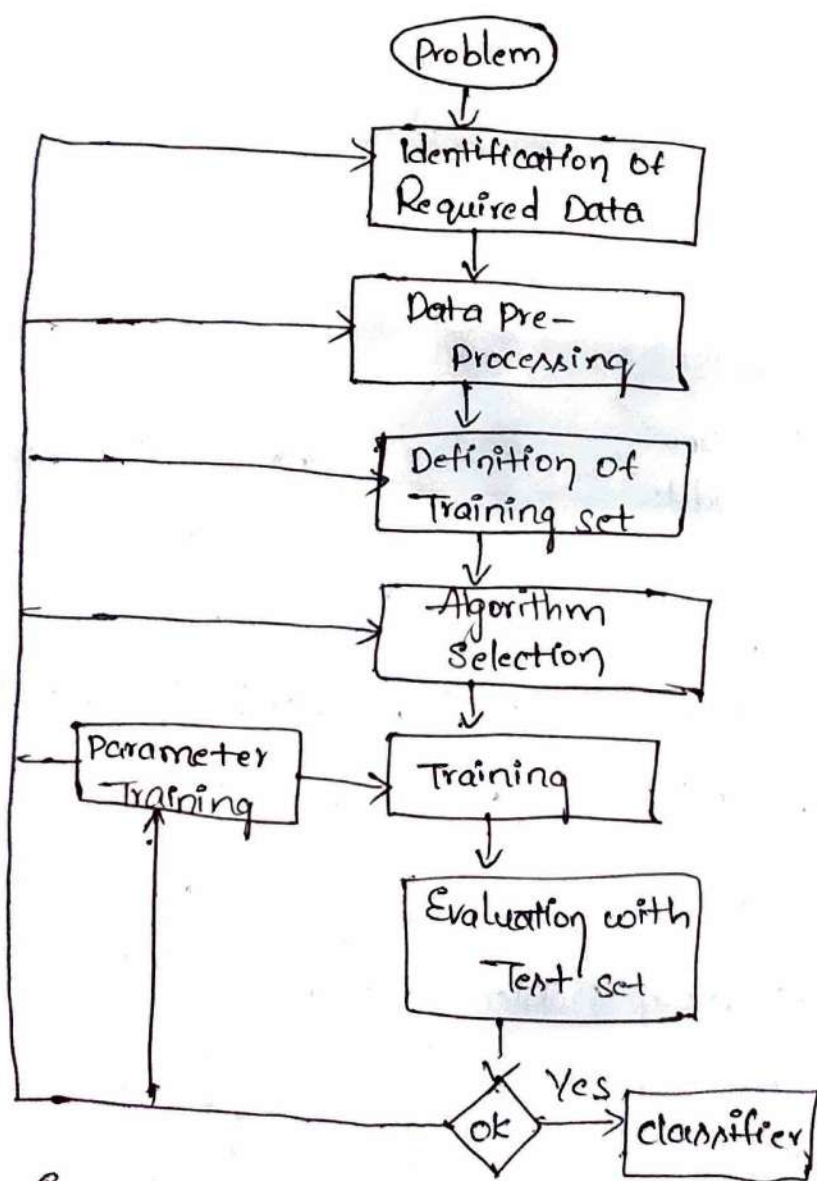


fig. 7.3

classification model steps

Problem Identification: Identifying the problem is the first step^③ in the Supervised learning model. The problem needs to be well-formed problem, i.e. a problem with well-defined goals and benefit, which has a long-term impact.

Identification of Required Data: on the basis of the problem identified above, the required dataset 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 not~~

Data pre-processing: This is related to the cleaning / transforming the dataset. This step ensures that all the unnecessary / irrelevant data elements are removed, before feeding the same into the algorithm. This step ensures that the data is ready to be fed into the machine 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 hand written word (i.e. a group of the alphabets) or an entire line of handwriting.

Algorithm Selection: This involves determining the structure of the learning function and the corresponding learning algorithm. 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 algorithms require the user to determine specific control parameters (which are given as inputs to the algorithm).

Evaluation with the Text Data Set: Training data is run on the algorithm, and its performance is measured here.

Common Classification Algorithms

Following are the most common classification algorithms.

1. K-Nearest Neighbour (KNN)
2. Decision tree
3. Random forest
4. Support vector Machine (SVM)
5. Naive Bayes Classifier

K-Nearest Neighbour (KNN)

The KNN algorithm is a simple but extremely powerful classification algorithm. The name of the algorithm originates from the underlying philosophy of KNN - 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.

How KNN works

(4)

Let us consider a very simple student dataset as depicted in fig. 7.4. It consists of 15 students studying in a class. Each of the students has been assigned a score on a scale of 100 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 good level 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 'Speaker', '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
Suxant	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.

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.

But there are two challenges:

1. What is the basis of this Similarity & when can we say that two data elements are similar?
2. 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 KNN to measure Similarity between two data elements is.

	Name	Aptitude	Communication	Class
Training Data	Karuna	2	5	Speaker
	Bhuvna	2	6	Speaker
	Gaurav	7	6	Leader
	Paul	7	2.5	Intel
	Dinesh	8	6	Leader
	Jani	4	7	Speaker
	Bobby	5	3	Intel
	Parimal	3	5.5	Speaker
	Goviind	8	3	Intel
	Susant	6	5.5	Leader
	Gouri	6	4	Intel
	Bharat	6	7	Leader
	Ravi	6	2	Intel
	Pradheep	9	7	Leader
Test Data →	Josh.	5	4.5	Intel

Fig 7.5 Segregated Student data set.

Euclidean distance. Considering a very simple data set having two features (say f_1 and f_2), Euclidean distance b/w two data elements d_1 and d_2 can be measured by

(5)

$$\text{Euclidean distance} = \sqrt{(f_{11} - f_{12})^2 + (f_{21} - f_{22})^2}$$

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

So, as depicted in figure T-6, the training data points of the student data set considering only the features 'Aptitude' and 'Communication' can be represented as dots.

Name	Aptitude	Communication	Class
Karuna	2	5	Speaker
Bhurna	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
Gauri	6	4	Intel
Bharat	6	7	leader
Ravi	6	2	Intel
Pradeep	9	7	leader
☆ Josh	5	4.5	???

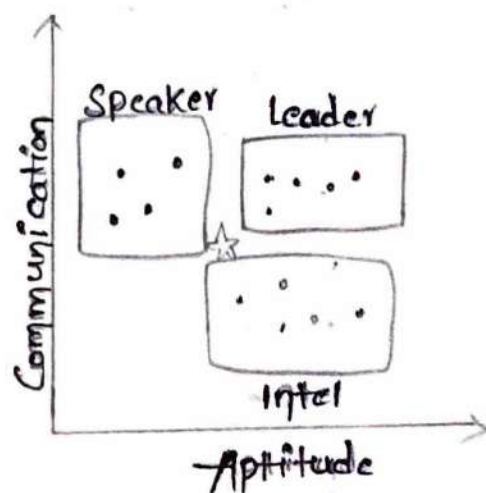


Fig. T-6

2-D representation of the student data set.

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 the three 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.

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			
Gowli	6	4	Intel	1.118	1.118	1.118	1.118
Parul	7	2.5	Intel	2.828			
Govind	8	3	Intel	3.354			
Suvarit	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 b/w test and training points.

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 no. of records in the training data), the class label of the majority class of the training dataset will be assigned to the test data regardless of the class labels of the neighbours nearest

-to the test data.

⑥

- If the value of k is very small (in the extreme case Equal to 1), the class value of a query data or outlier in the training dataset which is the nearest neighbour to the test data will be assigned to the test data.

The best k value is somewhere b/w 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 no. 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 $k \geq 1$

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?

That Eager learners follow the general steps of machine learning, i.e. Performing abstraction of the information obtained from the input data and then follow it through by a generalization step.

Strengths of the KNN algorithm.

- Extremely Simple algorithm - Easy to understand.
- Very Effective in certain situations, e.g. for recommender system design.
- Very fast & almost no time required for the training phase.

Weakness 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.

- 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.

(7)

One of the most popular areas in machine learning where the knn algorithm is widely adopted is 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.

A decision tree is used for multi-dimensional analysis with multiple classes. It is characterized by fast execution time 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.

A decision tree is usually represented in the format depicted in Figure 7.8.

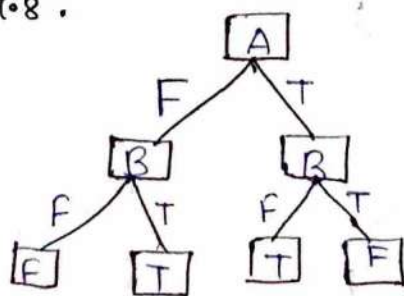


Fig. 7.8 Decision tree structure.

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 Root Node (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

Fig 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.

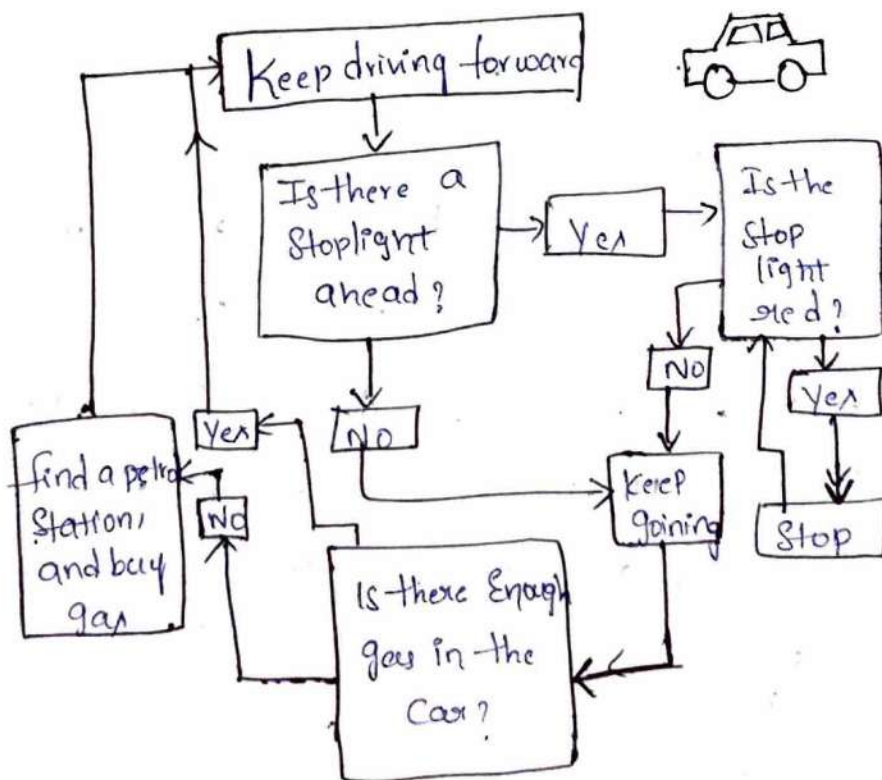


Fig. 7.9
Decision tree Example.

Building a decision tree.

(8)

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 dataset. It first selects the feature which predicts the target class in the strongest way. 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.

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 whether Chandra 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 decision tree.

For Aptitude = High, job offer condition is TRUE for all 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 diagram for the table given in Fig 7.10.

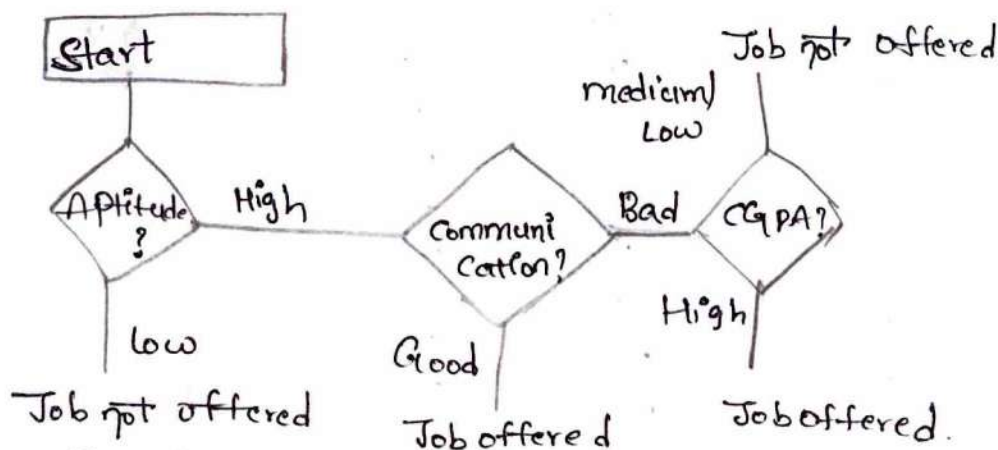


Fig. 7.11

Decision tree based on the training data

Searching a decision tree

⑨

By using the above decision tree depicted in fig 7.11, we need to predict whether chandra might get a Job offer for 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.

Exhaustive Search

1. place the item in the first group (class). Recursively examine solutions with the item in the first group (class).
2. place the item in the second group (class). Recursively examine solution with the item in the second group (class).
3. Repeat the above steps until the solution is reached.

Branch and bound Search

Branch and bound uses an existing best solution to substep 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.

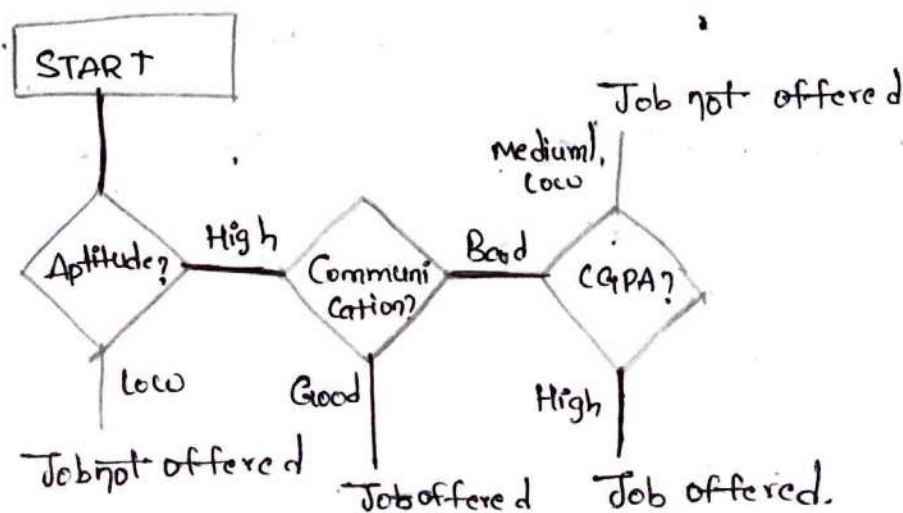


Fig. 7.12

Decision tree based on the training data (depicting a simple 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 can be made. as chandra will get the job offer.

There are many implementations of decision tree, the most prominent ones being CS.O, 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.

Entropy of a decision tree. Split class partitions - pure
Entropy is a measure of impurity
Let us say S is the sample set of training examples. Then.

Entropy (S) measuring the impurity of S is defined as

$$\text{Entropy}(S) = \sum_{i=1}^c -p_i \log_2 p_i$$

of an attribute or feature
adopted by many algorithms
such as ID3 & CS.O

where c is the number of different class labels and p_i refers 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 'Job offered?' - 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

$$\text{Entropy}(S) = -0.44 \log_2(0.44) - 0.56 \log_2(0.56) = 0.99$$

Information gain of a decision tree.

(10)

The information gain is created on the basis of the decrease in Entropy (S) after a dataset 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 homogenous branches). If the information gain is 0, it means that there is no reduction in Entropy due to split of the dataset according to that particular feature. On the other hand, the maximum amount of information gain which may happen is the Entropy of the dataset before the split.

Information gain for a particular feature A is calculated by the difference in Entropy before a split (S_{bs}) with the Entropy after the split (S_{as})

$$\text{Information Gain } (S, A) = \text{Entropy } (S_{bs}) - \text{Entropy } (S_{as})$$

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 T-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 -

1. when the feature 'CGPA' is used for the split
2. when the feature 'Communication' is used for the split.
3. when the feature 'Aptitude' is used for the split
4. when the feature 'programming skills' is used for the split

Figure T-8a 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
P_i	0.44	0.56	
$-P_i \cdot \log(P_i)$	0.52	0.47	0.99
<u>Total Entropy = 0.99</u>			

(b) Splitting data set based on the feature 'CGPA':

(11)

CGPA = High

CGPA = Medium

	Yes	No	Total
Count	4	2	6
P_i	0.67	0.33	
$-P_i \log(P_i)$	0.39	0.53	0.92

Total Entropy = 0.69

	Yes	No	Total
Count	4	3	7
P_i	0.57	0.43	
$-P_i \log(P_i)$	0.46	0.52	0.99

Information Gain = 0.30.

(c) Splitting data set based on the feature 'Communication':

Communication = Good

Communication = Bad

	Yes	No	Total
Count	7	2	9
P_i	0.78	0.22	
$-P_i \log(P_i)$	0.28	0.48	0.76

Total Entropy = 0.63

	Yes	No	Total
Count	1	8	9
P_i	0.11	0.89	
$-P_i \log(P_i)$	0.35	0.15	0.50

Information Gain = 0.36.

(d) Splitting data set based on the feature 'Communication':

Aptitude = High

Aptitude = Low

	Yes	No	Total
Count	8	3	11
P_i	0.73	0.27	
$-P_i \log(P_i)$	0.33	0.51	0.85

Total Entropy = 0.52.

	Yes	No	Total
Count	0	7	7
P_i	0.00	1.00	
$-P_i \log(P_i)$	0.00	0.00	0.00

Information Gain = 0.47

(e) Splitting data set based on the feature 'Programming Skill':

Programming Skill = Good

Programming Skill = Bad

	Yes	No	Total
Count	5	4	9
P_i	0.56	0.44	
$-P_i \log(P_i)$	0.47	0.52	0.99

Total Entropy = 0.95

	Yes	No	Total
Count	3	6	9
P_i	0.33	0.67	
$-P_i \log(P_i)$	0.53	0.39	0.92

Information Gain = 0.04

Fig: 7.13 - Entropy and Information gain calculation (level 1)

As a part

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 11.36 presents calculation 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.
- 0.80 when the feature 'programming skill' is used for Split.

Hence, the information gain after Split with the features $CGPA^{(2)}$ Communication and programming skill is 0.52, 0.55 and 0.05, respectively. 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

Fig. 7.13B (continued)

(a) level 2 Starting Set:

	Yes	No	Total
Count	8	3	11
P_i	0.73	0.27	
$-P_i \cdot \log(P_i)$	0.33	0.51	0.85

Total Entropy = 0.85

(b) Splitting data set based on the feature 'CGPA':

CGPA = High.

	Yes	No	Total
Count	4	0	4
P_i	1.00	0.00	
$-P_i \log(P_i)$	0.00	0.00	0.00

Total Entropy = 0.33.

CGPA = Low

	Yes	No	Total
Count	0	2	2
P_i	0.00	1.00	
$-P_i \log(P_i)$	0.00	0.00	0.00

CGPA = Medium

	Yes	No	Total
Count	4	1	5
P_i	0.80	0.20	
$-P_i \log(P_i)$	0.26	0.46	0.72

Information Gain = 0.52

(c) Splitting dataset based on the feature 'Communication':

Communication = Good.

	Yes	No	Total
Count	7	0	7
P_i	1.00	0.00	
$-P_i \log(P_i)$	0.00	0.00	0.00

Total Entropy = 0.30

Communication = Bad

	Yes	No	Total
Count	1	3	4
P_i	0.25	0.75	
$-P_i \log(P_i)$	0.50	0.31	0.81

Information Gain = 0.55

(d) Splitting data set based on the feature 'Programming Skill':

Programming Skill = Good

	Yes	No	Total
Count	5	1	6
P_i	0.83	0.17	
$-P_i \log(P_i)$	0.22	0.43	0.65

Total Entropy = 0.80.

Programming Skill = Bad

	Yes	No	Total
Count	3	2	5
P_i	0.60	0.40	
$-P_i \log(P_i)$	0.44	0.53	0.97

Information Gain = 0.65

Fig 7.13B. Entropy and information gain calculation (level 2)

As a part of level 3, we will thus have only one branch ^(B)
 to navigate in this case: - the one for Communication = Bad.

Figure 7.13c presents calculations for level 3. As can be seen from the figure, the Entropy Value is as follows:

- 0.81 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 0 Starting Set:

	Yes	No	Total
Count	1	3	4
p_i	0.25	0.75	
$-p_i \log(p_i)$	0.50	0.31	0.81

Total Entropy = 0.81.

b) Splitting data set based on the feature ('CGPA'):

CGPA = High

	Yes	No	Total
Count	1	0	1
p_i	1.00	0.00	
$-p_i \log(p_i)$	0.00	0.00	0.00

Total Entropy = 0.00

CGPA = medium

	Yes	No	Total
Count	0	1	1
p_i	0.00	1.00	
$-p_i \log(p_i)$	0.00	0.00	0.00

Information Gain = 0.81

As a part of level 3, we will thus have only one branch⁽³⁾

to navigate in this case, - the one for Communication = Bad.

Figure 7.13c presents calculations for level 3. As can be seen from

the figure, the Entropy Value is as follows:

- 0.81 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 & Starting Set:

	Yes	No	Total
Count	1	3	4
p_i	0.25	0.75	
$-p_i \log(p_i)$	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
p_i	1.00	0.00	
$-p_i \log(p_i)$	0.00	0.00	0.00

Total Entropy = 0.00

CGPA = medium

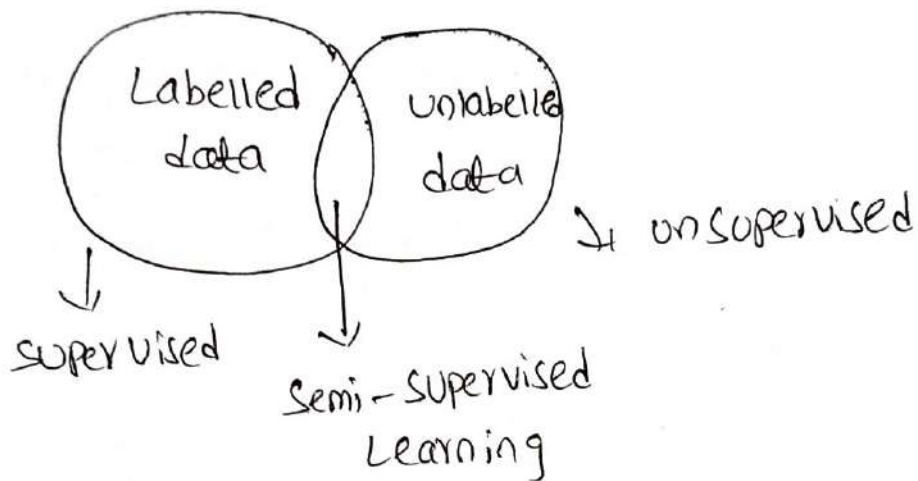
	Yes	No	Total
Count	0	1	1
p_i	0.00	1.00	
$-p_i \log(p_i)$	0.00	0.00	0.00

Information Gain = 0.81

Unit - III

Models Based on Decision Trees

Training data is the past information with known value of class field or label. Hence we say that the "training data is labelled"



→ To start solving a classification problem by using some "standard classifiers Algorithms". They are

- 1) KNN
- 2) Decision tree
- 3) Random Forest
- 4) Support vector machine
- 5) Naive Bayes Classifier (Baye's Theorem)

- Rule Engines
- Expert systems
- Knowledge graphs
- Symbolic AI

1) Decision tree is based on "series of logical decisions" which resembles a tree with branches.

2) It is one of the most widely adopted algorithms for classification.

It builds a model in the form of a tree structure. It groups the subsets with different strategies and it is exceptionally productive

The main goal of build the decision tree is, "create a model", based on the past data

D-T Consists of 3 types of nodes.

a) Root Node (or) Parent Node

b) Branch Node (or) Internal Node (or) Child Node.

c) Leaf Node (or) Terminal Node.

1) Root Node :- first node is called "Root Node"
It has no incoming edges and zero or more outgoing edges.

2) Branch Node (or) Internal Node (or) child :-
Branch Nodes are associated with root node
Each of which has exactly one incoming edge
and two or more outgoing edges.

3) Leaf Node :- It is also known as terminal nodes, Each of which has exactly one incoming edge and no outgoing edges.

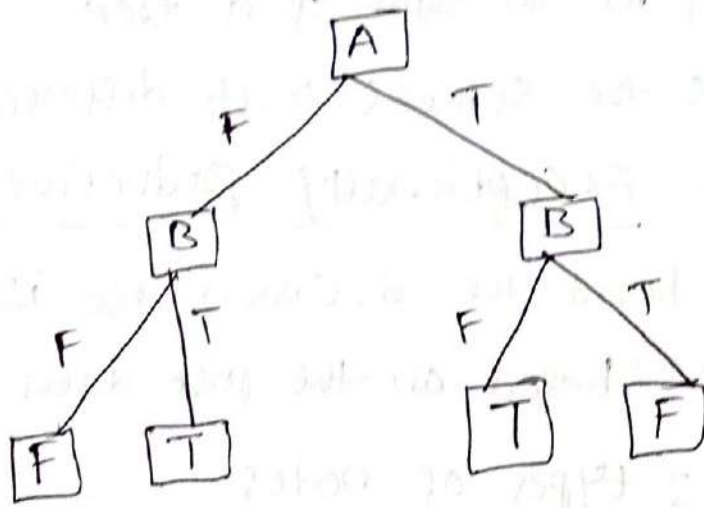


Fig:- Decision tree structure.

*

Building a decision tree:-

Decision trees are built corresponding to the training data following an approach called "recursive partitioning".

→ The algorithm continues splitting the nodes on the basis of the feature which helps in the best partition.

→ This continues till "stopping criterion is reached".

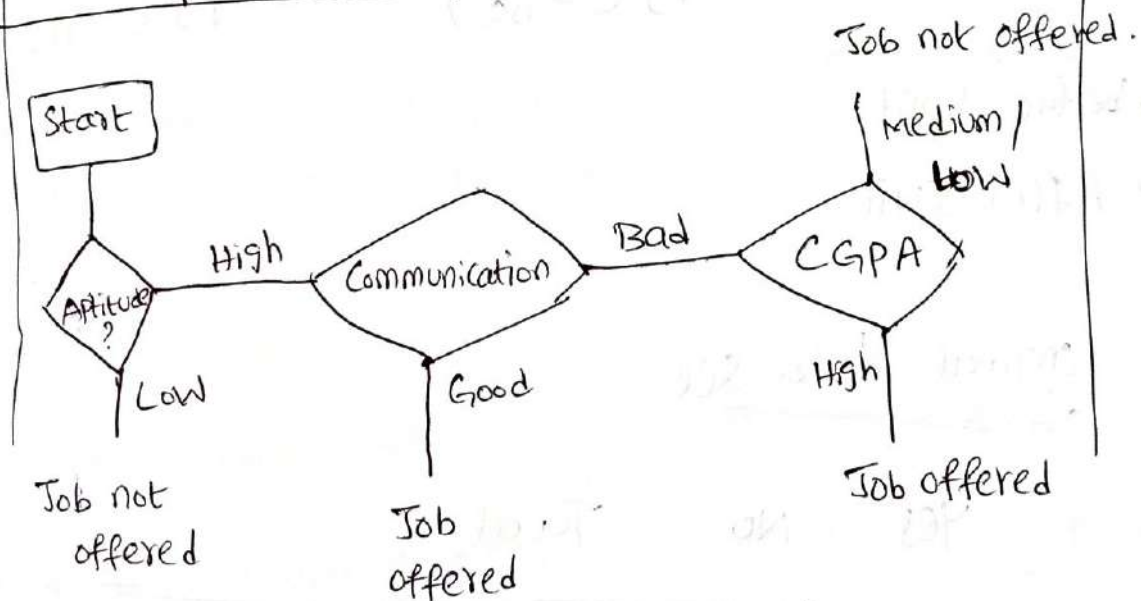
1) All or most of the examples at a particular node have the same class.

2) All features have been used up in the partitioning.

3) The tree has grown to a pre-defined threshold limits

1:

<u>SUMMARY</u>				
1) ICU 2) Campus drive				
CGPA	Communication	Aptitude	Programming Skill	Job offered?
High	Good	High	Good	Yes
Medium	Bad	Low	Bad	No
Low				



Searching D.T

• Exhaustive Search

• Branch & Bound

• Entropy of a D.T

→ Shannon's Entropy

$$\text{Entropy}(S) = \sum_{i=1}^n -P_i \log_2 P_i$$

where 'C' is no. of different class labels.
 'P' refers to the proportion of values falling to the i th class label.

$$\text{Entropy}(S) = \sum_{i=1}^n W_i \text{Entropy}(P_i)$$

Information Gain:

• decrease of Entropy (\downarrow)

• if the information gain is '0'

• $I.G = \text{Entropy}(S_{bs}) - \text{Entropy}(S_{As})$

• before split

• After split

original data set

yes	No	Total
-----	----	-------

Count

P_i

$-P_i \times \log(P_i)$

- top-down recursive partitioning
- data fragmentation problem
- divide-and-conquer partitioning strategy.

The border b/w two neighboring regions of diff classes is known as a "decision boundary"

* Random Forest Model :-

- 1) How does Random Forest Work?
- 2) OOB error in Random Forest?
- 3) Strength of R.F
- 4) Weakness of R.F
- 5) Application of R.F

① OOB Error in R.F :-

* The total Error rate of predictions for such samples is termed as OOB Error rate.

② Strength of R.F :-

- It runs Efficiently on large & Expansive D.S
- It has a robust method

missing values,
maintain precision

"When a large portion of data is absent"

- It has powerful techniques for balancing errors

③ Weakness of R.F :-

- It is not easy to understand a decision tree model.
- Computational much more Expansive

Applications of R.F.:-

- It Combines the versatility of many decision tree models into a single model.
- Wide range of classification problems.

* Properties of D.T.:-

- 1) It is Greedy
- 2) It can be Expensive
- 3) It Can overfit
- 4) Random Forest
- 5) AdaBoost
- 6) Gradient Boost

Strength	Weakness
Simple understandable rules	→ overfitting & underfitting
Computational knowledge	are common.
Numerical & Categorical	→ computationally Expensive
Small & large training	large trees are complex to understand.

slip test Questions:

* Decision tree:

- 1) Decision tree
- 2) Building a D.T
- 3) searching a D.T { Exhaustive Search
Branch & Bound
Search
- 4) Entropy of a D.T
- 5) Information Gain of a D.T
- 6) Avoiding overfitting in D.T — pruning
 - pre pruning
 - post pruning
- 7) Strength of D.T
- 8) Weakness of D.T
- 9) Application of D.T
- 10) properties of D.T

Note: Entropy is a measure of impurity of an attribute or feature

- CART (Classification & Regression tree)
- ID3 (Iterative Dichotomiser 3)
- Chi-Square Algorithm.

Weighted summation of the Entropy:

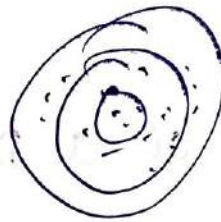
$$\text{Entropy}(S_{as}) = \sum_{i=1}^C w_i \text{Entropy}(P_i)$$

Bias - Variance trade off:

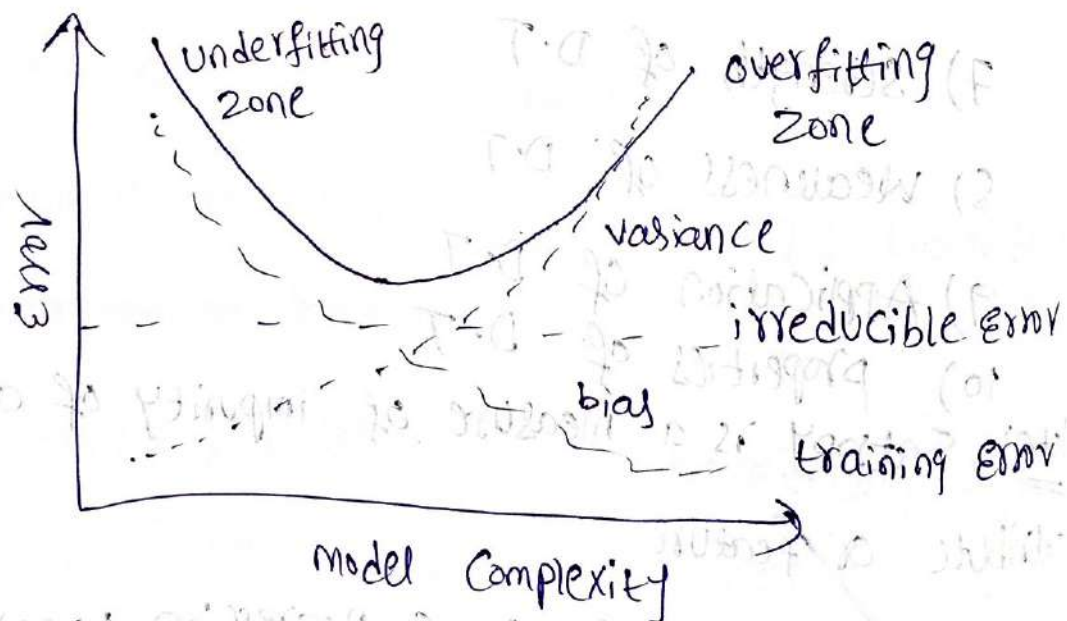
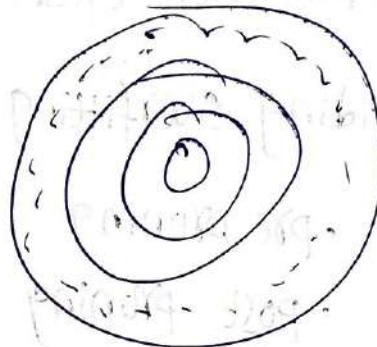
Low variance

high variance

Low
Bias



high
Bias



Baye's Theorem:-

- 1) It describes the "probability of occurrence of an event" related to any condition. It is also considered for the case of "conditional probability"
- 2) Bayes Theorem is also known as the formula for the "probability of causes"

Eg:- If we have to calculate the probability of taking a blue ball from the second bag out of three different bags of balls, where each bag contains three different colour balls i.e red, blue, black. in this case the probability of occurrence of an event is calculated depending on other condition is known as "conditional probability".

Conditional probability Baye's Theorem:-

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

where $P(B) \neq 0$

$P(A/B)$ is the probability of condition when event A is occurring while event B has already occurred.

Bayes Theorem Statement:-

Let $E_1, E_2, E_3, \dots, E_n$ be a set of events associated with a sample space 's' where all the events E_1, E_2, \dots, E_n have non zero probability of occurrence and they form a partition of 's'. Let 'A' be any event associated with 's' then according

to Bayes's Theorem.

$$P(E_i/A) = \frac{P(E_i)P(A/E_i)}{\sum_{k=1}^n P(E_k)P(A/E_k)}$$

for any $k = 1, 2, 3, \dots, n$

Bayes Theorem proof :-

According to the Conditional probability formula,

$$P(E_i/A) = \frac{P(E_i \cap A)}{P(A)} \rightarrow \textcircled{1}$$

Using the multiplication rule of probability

$$P(E_i \cap A) = P(E_i)P(A/E_i) \rightarrow \textcircled{2}$$

Using total probability Theorem,

$$P(A) = \sum_{k=1}^n P(E_k)P(A/E_k) \rightarrow \textcircled{3}$$

Putting the values from eq $\textcircled{2}$ & eq $\textcircled{3}$ in eq $\textcircled{1}$,
we get

$$P(E_i/A) = \frac{P(E_i)P(A/E_i)}{\sum_{k=1}^n P(E_k)P(A/E_k)}$$

Note: The following terminologies are also used when the Bayes Theorem is applied.

① Hypothesis: The event e_1, e_2, \dots, e_n is called the hypothesis.

② priori probability: The probability $P(E_i)$ is considered as the priori probability of hypothesis E_i .

③ posteriori probability: The probability $P(E_i/A)$ is considered as the posteriori probability of hypothesis E_i .
From the definition of conditional probability, Baye's theorem can be derived for events as given below.

$$P(A/B) = \frac{P(A \cap B)}{P(B)} \quad \text{where } P(B) \neq 0$$

$$P(B/A) = \frac{P(B \cap A)}{P(A)} \quad \text{where } P(A) \neq 0$$

Here, the Joint probability $P(A \cap B)$ of both events A and B being true such that,

$$P(B \cap A) = P(A \cap B)$$

$$P(A \cap B) = P(A/B) P(B) = P(B/A) P(A)$$

$$P(A/B) = \frac{P(B/A) P(A)}{P(B)} \quad \text{where } P(B) \neq 0$$

Example: A bag 1 Contains 4 white and 6 black balls while another bag 2 Contains 4 white and 3 black balls. One ball is drawn at random from one of the bags and it is found to be black.

Find the probability that it was drawn from Bag 1?

Sol: Let E_1 be the Event of Choosing bag 1

E_2 be the Event of Choosing bag 2

A be the Event of drawing a black ball

Then

$$P(E_1) = P(E_2) = \frac{1}{2} = 0.5$$

$$\text{Also } P(A/E_1) = \frac{6}{10} = \frac{3}{5}$$

$$P(A/E_2) = \frac{3}{7}$$

By using Baye's Theorem the probability of drawing a black ball from bag 1 out of two bags

$$\begin{aligned} P(E_1/A) &= \frac{P(E_1) \cdot P(A/E_1)}{P(E_1) \cdot P(A/E_1) + P(E_2) \cdot P(A/E_2)} \\ &= \frac{\frac{1}{2} \times \frac{3}{5}}{\frac{1}{2} \times \frac{3}{5} + \frac{3}{7} \times \frac{1}{2}} = \frac{7}{12} \end{aligned}$$

* Baye's Classifier and its optimality:-

Baye's optimal classifier is a probabilistic model that makes the most probable prediction for a new Example.

$$P(A/B) = \frac{P(B/A) \cdot P(A)}{P(B)}$$

For a dataset

→ Yes/No

$$x = \{x_1, x_2, x_3, \dots, x_n\} \in \{I\}$$

substitution given data in baye's Theorem

$$P(y/x_1, x_2, \dots, x_n) = \frac{[P(x_1/y) \cdot P(x_2/y) \cdot \dots \cdot P(x_n/y)] \times P(y)}{P(x_1) \cdot P(x_2) \cdot P(x_3) \cdot \dots \cdot P(x_n)}$$

$$P(y) \prod_{i=1}^n P(x_i/y) \text{ where } i=1, 2, 3, \dots$$

$$\Rightarrow \frac{P(y) \prod_{i=1}^n P(x_i/y)}{P(x_1) \cdot P(x_2) \cdot \dots \cdot P(x_n)}$$

$$\Rightarrow P(y) \prod_{i=1}^n P(x_i/y)$$

Example: outlook

outlook	yes	No	$P(y)$	$P(N)$
Sunny	2	3	2/9	3/5
outcast	4	0	4/9	0/5
rain	3	2	3/9	2/5
Total	9	5	100%	100%

Temperature:

Temp	Yes	No	P(Y)	P(N)
Hot	2	2	2/9	2/5
mild	4	2	4/9	2/5
Cold	3	1	3/9	1/5
Total	9	5	100%	100%

Play:

Yes	9	9/14
No	5	5/14
Total	14	100%

Now we need to calculate the probability of sunny, hot

$$P(\text{Yes/sunny, hot}) = \frac{P(\text{sunny/yes}) \times P(\text{hot/yes}) \times P(\text{Yes})}{P(\text{sunny}) \times P(\text{hot})}$$

We will eliminate denominator

$$= \frac{2}{9} \times \frac{2}{9} \times \frac{9}{14}$$

$$P(\text{Yes/sunny, hot}) = 0.031$$

$$P(\text{No}/\text{sunny, hot}) = \frac{P(\text{sunny}/\text{No}) \times P(\text{hot}/\text{No}) \times P(\text{No})}{P(\text{sunny}) \times P(\text{hot})}$$

we will eliminate Denominator

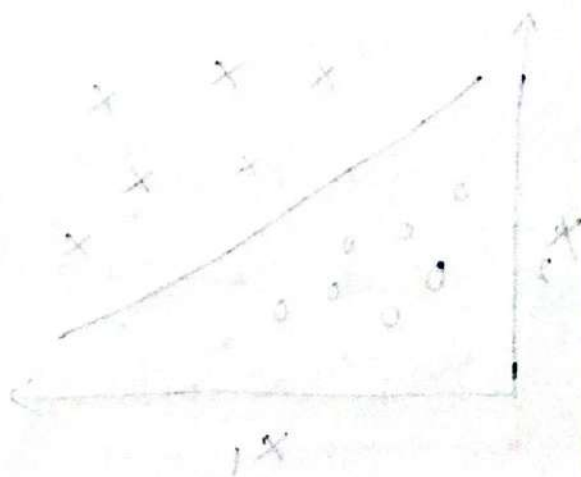
$$= \frac{3}{5} \times \frac{2}{5} \times \frac{5}{14} = 0.08571$$

$$\begin{aligned} \text{Total} &= 0.031 + 0.08571 \\ &= 0.27 \end{aligned} \quad P(\text{yes}) + P(\text{No})$$

$$P(\text{yes}) = \frac{0.031}{0.27} = 0.114$$

$$P(\text{No}) = \frac{0.08571}{0.27} = 0.317$$

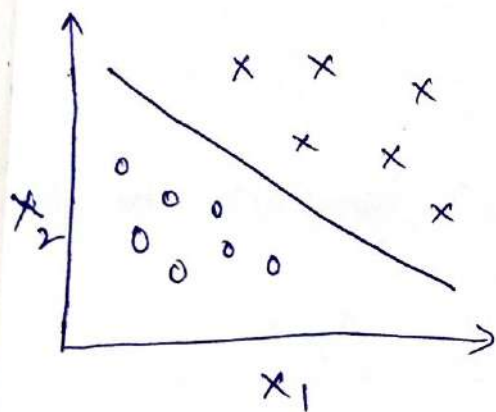
In our problem the probability of No is more so the play will not enjoy the sport.



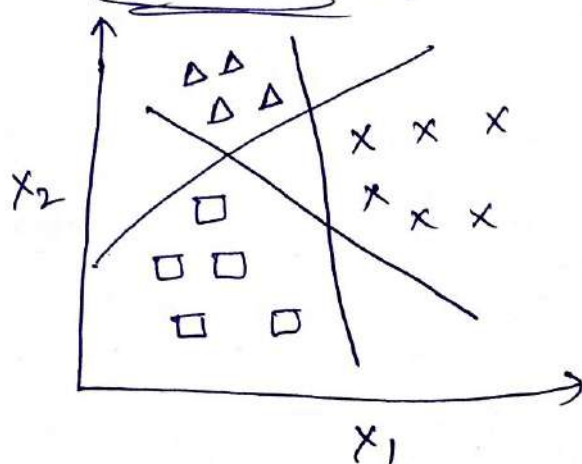
* Multi Class Classification :-

- 1) When we solve a classification problem having only two class labels, then it becomes easy for us to filter the data, apply any classification algorithm, train the model with filtered data and predict the outcomes.
- 2) But when we have "more than two class instances" in input train data, then it might get Complex to analyze the data, train the model and predict relatively accurate results.
- 3) To handle these multiple class instances, we use multi-class classification.
- 4) Multi-Class Classification is the classification technique that allows us to categorize the test data into multiple class labels present in trained data as a model prediction.

Binary vs Multi-Class Classification :-



Binary
Classification



Multi-Class
Classification

Binary Classification:-

- 1) only two class instances are present in the dataset
- 2) it requires only one classifier model.
- 3) Confusion Matrix is easy to derive and understand

Example: Check Email is spam or not, predicting gender based on height & weight.

Multi class Classification:-

- 1) Multiple class labels are present in the dataset.
- 2) The no. of classifier models depends on the classification technique we ^{are} applying to.
- 3) one vs All :- N -class instances then ' N ' binary classifier models.

- 4) one vs one :- N -class instances then $N * (N-1) / 2$ binary classifier models.

Example: Check whether the fruit is apple, banana, or orange.

One vs All (One-vs-Rest)

In one-vs-All classification, for the N -class instances dataset, we must generate the N -binary classifier models.

→ The no. of class labels present in the dataset and the no. of generated binary classifiers

must be same.

→ Consider we have three classes. for example :-
Green, Blue and Red.

Now, we create three classifiers: here for those
respective classes.

Classifier 1 : [Green] VS [Red, Blue]

Classifier 2 : [Blue] VS [Green, Red]

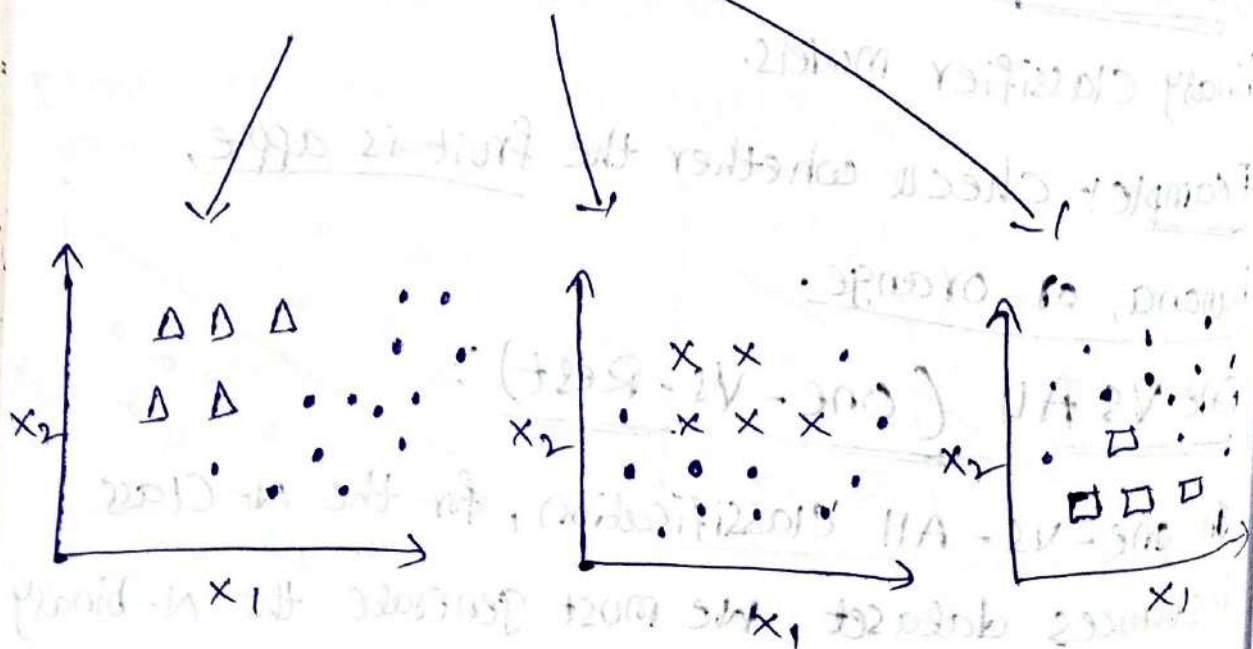
Classifier 3 : [Red] VS [Green, Blue]



Class 1 : Green

Class 2 : Blue

Class 3 : Red



Classifier 1

Classifier 2

Classifier 3

→ You can see that there are three class labels Green, Blue and Red present in the dataset. Now we must create a training dataset for each class.

Features			Classes
x_1	x_2	x_3	G
x_4	x_5	x_6	B
x_7	x_8	x_9	R
x_{10}	x_{11}	x_{12}	G
x_{13}	x_{14}	x_{15}	B
x_{16}	x_{17}	x_{18}	R

original
main dataset

Training Dataset 1: Class Green

Features			Classes
x_1	x_2	x_3	+1
x_4	x_5	x_6	-1
x_7	x_8	x_9	-1
x_{10}	x_{11}	x_{12}	+1
x_{13}	x_{14}	x_{15}	-1
x_{16}	x_{17}	x_{18}	-1

Where Every Green put
+1 and Everywhere -1

Similarly Class Blue and Red

Training Dataset 2: Class Blue

Training Dataset 3: Class Red

Let's understand with one Example by taking three test features values as y_1, y_2, y_3 respectively.

→ We passed test data to the Classifier models.

→ Let's say we got the outcomes as

Green class Classifier → positive with a Probability Score of (0.9)

Max in Positive

Blue class Classifier → positive with a P.S of (0.4)

Red class Classifier → Negative with a P.S of (0.5)

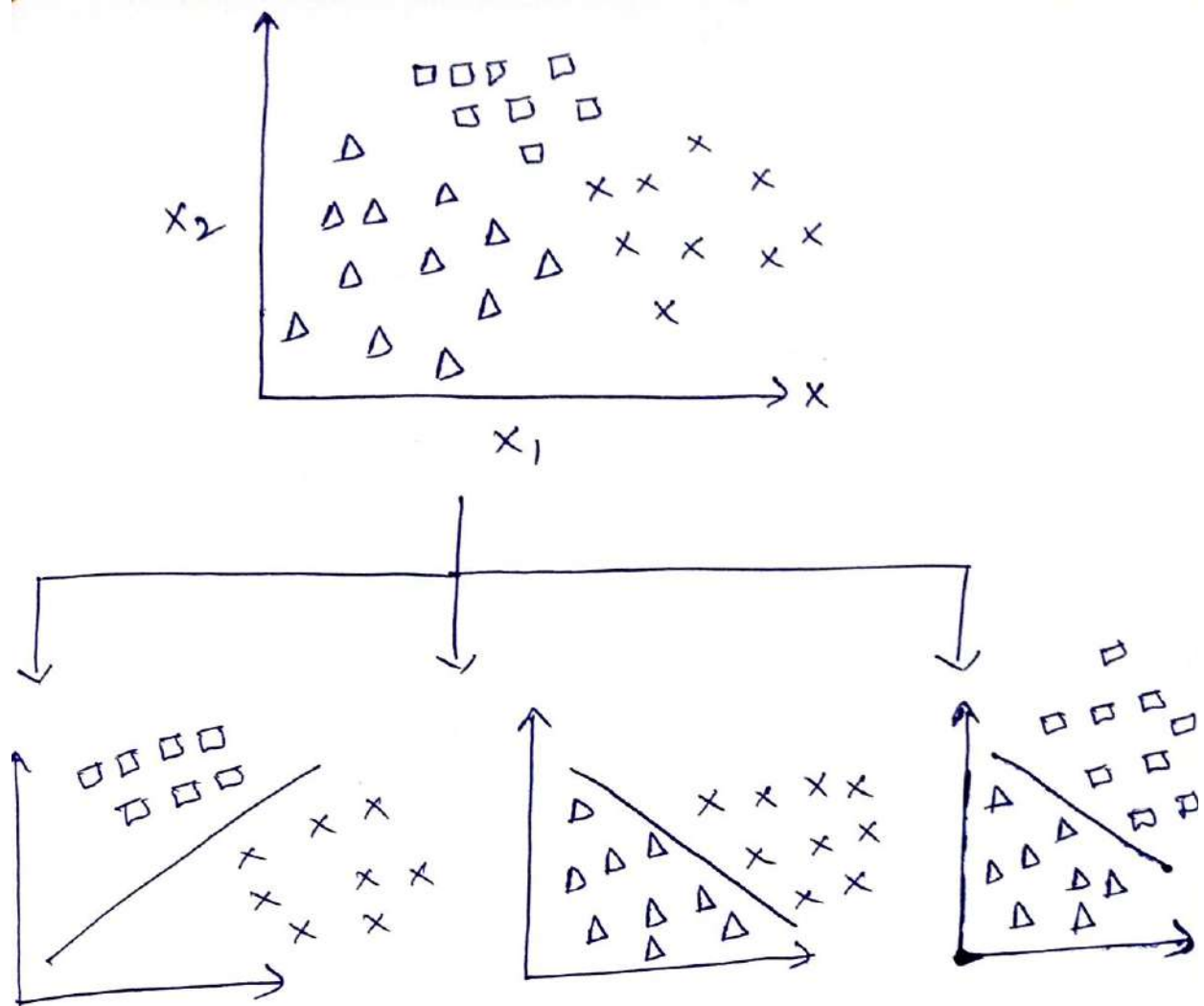
Hence, based on the positive responses and decisive probability score, we say that our test input belongs to the Green class

One vs one :

→ In one-vs-one classification for the N -class instances dataset, we must generate the $\frac{N*(N-1)}{2}$ binary Classifier models.

→ Using this Classification approach, we split the primary dataset into one dataset for each class opposite to every other class.

→ Taking the above Example, we have a Classification problem Solving three types Green, Blue and Red. ($N=3$)



→ We divide this problem into $N * (N-1) / 2 = 3$ binary classifier problems :

Classifier 1 : Green VS Blue

Classifier 2 : Green VS Red

Classifier 3 : Blue VS Red

→ Each binary classifier predicts one class label.

→ When we input the test data to the Classifier then the model with the majority Counts is concluded as a result.

Unit - V

* Introducing to clustering :-

→ It refers to the "process of arranging or organizing objects according to specific criteria".

→ It involves dividing or grouping the larger data sets into smaller datasets based on similarities or dissimilarities.

→ Depending on the requirements this grouping can "lead to various outcomes" such as

Data partitioning.

Data Re-organization

Data Compression

Data Summarization.

→ clustering of data is crucial aspect of efficient data access in database-based applications.

* Data partitioning :-

It will divide the larger dataset into smaller dataset in order to make the subsets based on similarities and dissimilarities.

Data Re-organization:

pattern	f_1	f_2	f_3	f_4	f_5	f_6
1	1	0	1	0	1	0
2	0	1	0	1	0	1
3	1	0	1	0	1	0
4	0	1	0	1	0	1

The table contains binary patterns with matrix display.

pattern	f_1	f_2	f_3	f_4	f_5	f_6
1	1	0	1	0	1	0
3	1	0	1	0	1	0
2	0	1	0	1	0	1
4	0	1	0	1	0	1

Tab:- Data Re-organization using row as the criteria.

pattern	f_1	f_3	f_5	f_2	f_4	f_6
1	1	1	1	0	0	0
3	1	1	1	0	0	0
2	0	0	0	1	1	1
4	0	0	0	1	1	1

$$f_1 \wedge f_3 \wedge f_5 \Rightarrow \text{cluster 1}$$

$$f_2 \wedge f_4 \wedge f_6 \Rightarrow \text{cluster 2}$$

Data Compression:

Pattern	f_1	f_2	f_3	f_4	f_5	f_6	Count
1, 3	1	0	1	0	1	0	2
2, 4	0	1	0	1	0	1	2

Data Summarization:

It is to Extract a representative subset of samples from large dataset.

→ The purpose is to Simplify Analysis on a smaller dataset.

Topic 2:

Matrix factorization in Clustering:

It is a mathematical technique used in clustering, where a given data matrix X is approximated as the product of two smaller matrices 'B' and 'C'.

$$X \approx BC$$

→ ' X ' is the original data matrix (size $n \times d$) where ' n ' represent data points and ' d ' represent d -dimensional space, where each row represents a data point.

→ 'B' is the cluster Assignment matrix (size $n \times K$) which indicates, which cluster each data point belongs to.

→ 'C' is the representative matrix (size $K \times d$) which contains the representative values such as centroids of each cluster.

Example: Consider the data matrix

$$X = \begin{bmatrix} 6 & 6 & 6 \\ 6 & 6 & 8 \\ 2 & 4 & 2 \\ 2 & 2 & 2 \end{bmatrix}$$

Here, we have four patterns (rows) in a three-dimensional space (columns)

→ using the Leader Clustering Algorithm with a threshold of 3 units:

cluster 1: $\{ (6, 6, 6), (6, 6, 8) \}$ with $(6, 6, 6)$ as the leader.

cluster 2: $\{ (2, 4, 2), (2, 2, 2) \}$ with $(2, 4, 2)$ as the leader.

Thus, the 'B' matrix (cluster Assignment) is

$$B = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

→ Each row in 'B' has a 1 in the Column corresponding to its cluster.

→ The C Matrix (cluster representative values) is

$$C = \begin{bmatrix} 6 & 6 & 6 \\ 2 & 4 & 2 \end{bmatrix}$$

Thus, the approximation of 'X' using Matrix factorization is

$$X \approx BC = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 6 & 6 & 6 \\ 2 & 4 & 2 \end{bmatrix}$$

Expanding:

$$X \approx \begin{bmatrix} 6 & 6 & 6 \\ 6 & 6 & 6 \\ 2 & 4 & 2 \\ 2 & 4 & 2 \end{bmatrix}$$

Topic 3:-

Clustering is an unsupervised learning technique used to group similar data points (patterns)

together while keeping dissimilar ones separate.

→ It is widely used in pattern recognition,

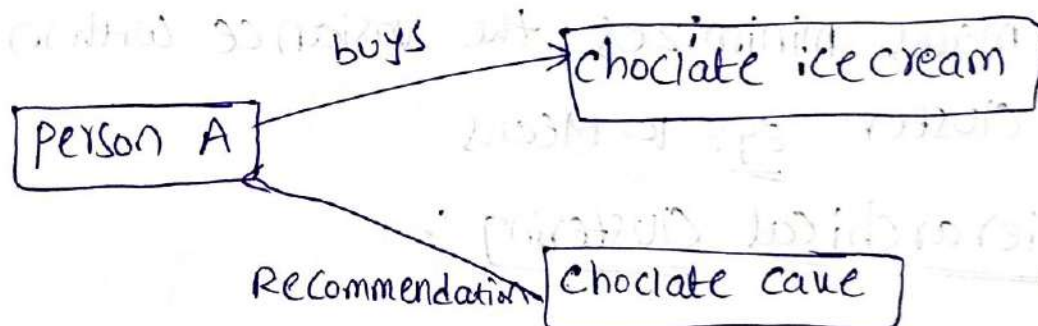
image segmentation, customer segmentation

& anomaly detection.

What is Clustering of Patterns?

It involves organizing a dataset into meaningful groups (clustering) based on similarity. A pattern is a data point represented as a feature vector & clustering aims to discover natural groupings within the dataset.

For example, if we have a dataset of customer transactions, clustering can group customers with similar purchasing behaviours.



Send by database based on frequency count of chocolate icecream purchased by person A.

Types of Clustering

There are two types of clustering. They are

- 1) Hard Clustering
- 2) Soft Clustering

① H.C: Each data point belongs "strictly" to one cluster"

Eg:- K-means Clustering.

② Soft Clustering: A datapoint can belong to multiple clusters with different probabilities
→ It is also known as Fuzzy Clustering
Eg:- Fuzzy C-means Clustering.

Types of Clustering Algorithms:

① Partition-based Clustering:-

→ Divides the dataset into k clusters based on distance metrics.

→ k -means minimizes the variance within each cluster. Eg:- k -Means

② Hierarchical Clustering:-

Builds a hierarchy of clusters using bottom-up (merging) or top-down (splitting) approaches

Eg:- Agglomerative, divisive

③ Density-based Clustering:-

→ Forms clusters based on dense regions in the data space.

→ Can detect outliers and irregular cluster shapes.

Eg:- DBSCAN, OPTICS

↓ Density Based Spatial Clustering

Applications
With Noise

④ Grid based Clustering :-

Divides the data space into grids and performs clustering within each grid.

→ Efficient for large datasets.

Eg: STING, CLIQUE

⑤ Model-based clustering :-

Assumes data is generated from a mixture of probability distributions.

Eg: Gaussian Mixture Models.

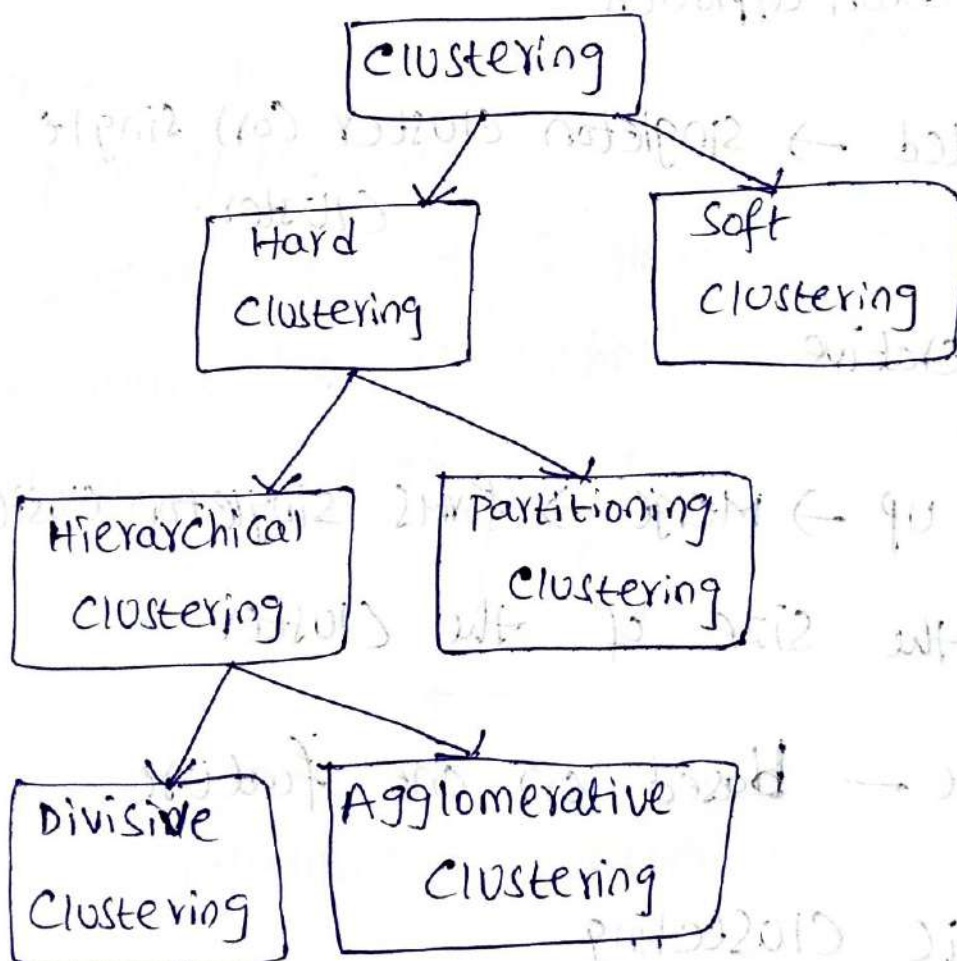


fig: Classification of Clustering Algorithms.

divisive clustering:

The process of generating a sequence of data partitions using hierarchical algorithms can be represented using a tree structure called "dendrogram"

1) divisive

2) Agglomerative

① divisive :-

top-down approach



splitting → singleton cluster (or) single cluster.

Agglomerative



bottom up → Merge → forms singleton cluster

Reduces the size of the cluster.

① divisive — based on one feature

monothetic clustering

poly-thetic clustering

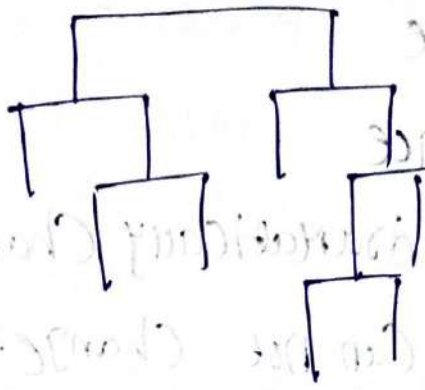


Fig:- Polythetic clustering

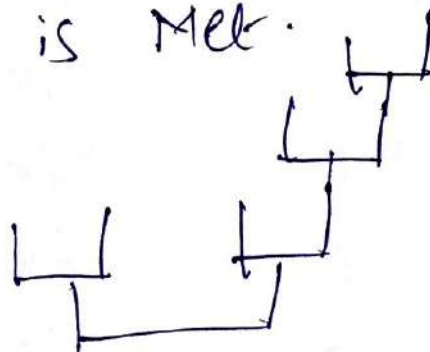
Time Complexity: $O(n \log n)$

where 'n' is no. of data points

2) Agglomerative clustering:-

Merges based on similar clusters; closest pair of clusters, based on distance measures or proximity measures.

→ In order to form singleton cluster then terminate the algorithm. (or) Stopping criteria is Met.



Agglomerative clustering

Time complexity $O(n^2)$

→ Inter Cluster distance

→ Intra Cluster distance

→ Centroid → shift dramatically Changes.

→ Medoid → stable Can not change.

↓

It is the point that minimizes the sum of distances to all other points in the cluster.

$$\text{Centroid} = \frac{1}{n_c} \times \sum (x_i \in C)$$

where $x = \{x_1, x_2, \dots, x_n\}$ n data points which are mapped to C

$n_c =$ no. of patterns in the cluster C



* Partitional Clustering :-

one of the most popular algorithms under this category is the k-Means Clustering.

k-Means Clustering :-

1) Form the given 'n' data points. Randomly select 'k' and initialize the 'u' value. The remaining (n-k) data points are then assigned to one of the k clusters based on their proximity to the closest cluster.

2) This is done by calculating the Mean or Centroid of all the data points belonging to each cluster.

3) By Iteratively updating the cluster centres & reassigning data points.

Example: Consider data set containing Eight points with 2 features

Data points	f_1	f_2
x_1	1	1
x_2	1	3
x_3	3	3
x_4	7	2
x_5	8	2
x_6	9	3
x_7	7	9
x_8	8	9

Let us assume that no. of clusters, $k=3$. If we select the initial clusters as x_1, x_4, x_7

$$x_1 \Rightarrow \text{Cluster } C_1 = (1, 1)$$

$$x_4 \Rightarrow \text{Cluster } C_2 = (7, 2)$$

$$x_7 \Rightarrow \text{Cluster } C_3 = (7, 9)$$

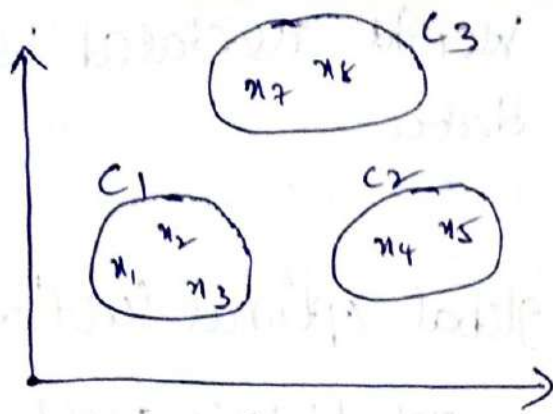
Table: Euclidean distance with Cluster Assignment

Data points	Euclidean dist from C_1	Euclidean dist from C_2	Euclidean dist from C_3	Cluster Assignment
x_1	0	$\sqrt{37}$	10	C_1
x_2	2	$\sqrt{37}$	$\sqrt{72}$	C_1
x_3	$\sqrt{8}$	$\sqrt{17}$	$\sqrt{52}$	C_1
x_4	$\sqrt{37}$	0	7	C_2
x_5	$\sqrt{50}$	1	$\sqrt{50}$	C_2
x_6	$\sqrt{68}$	$\sqrt{5}$	$\sqrt{40}$	C_2
x_7	10	7	0	C_3
x_8	$\sqrt{113}$	$\sqrt{50}$	1	C_3

$$\text{Cluster } C_1 = x_1, x_2, x_3$$

$$\text{Cluster } C_2 = x_4, x_5$$

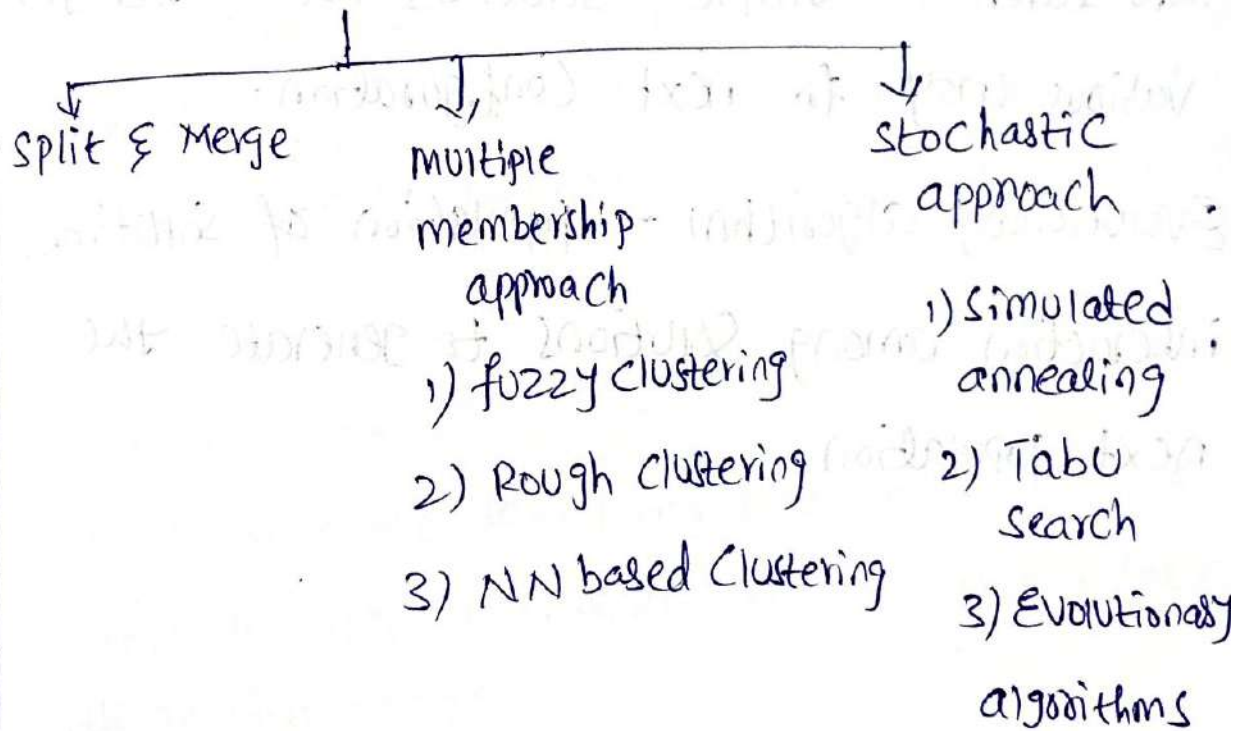
$$\text{Cluster } C_3 = x_7, x_8$$



Note: Alternatively the Elbow method is a popular approach to determine the value of k .

* Soft partitioning:-

- locally optimal solution
- various solutions were proposed.
- initial seeds.



fuzzy clustering — assigned to multiple clusters.

Rough clustering — non-lapping part overlapping part

NN based clustering - weight associated with data.

Stochastic approach - global optimal solution for cluster formation. probabilistic methods

simulated annealing - current solution is randomly updated

→ resulting solution

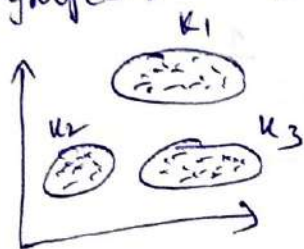
→ probability range from 0 to 1

Tabu Search - multiple solutions are stored for various way for next configuration.

Evolutionary algorithm - population of solution interaction among solutions to generate the next population

*Soft clustering:-

clustering is a distance based unsupervised ML Algorithm where data points that are close to each other are grouped in a given no. of clusters/groups.

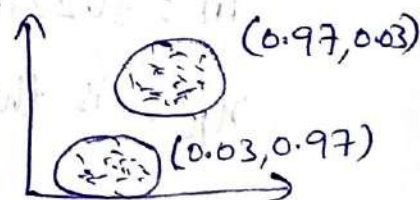


① Hard clustering:- In hard clustering each data point is assigned only a single cluster.

→ The k -means, k -medoid clustering algorithms are hard clustering algorithms

② Soft clustering:- on the other hand in soft clustering each datapoint belongs to a cluster with a certain probability also "known as membership value"

→ FCM (Fuzzy C-means clustering) Algorithm is an example of soft clustering



Fuzzy C-Means clustering steps:-

Step 1:- Given the data points based on the no. of clusters required initialize the membership table with random values

→ Suppose the given data points are

$$\{(1, 3), (2, 5), (6, 8), (7, 9)\}$$

Cluster	(1, 3)	(2, 5)	(4, 8)	(7, 9)
1	0.8	0.7	0.2	0.1
2	0.2	0.3	0.8	0.9
	$\sum = 1$	1	1	1

Step 2 :- Find out the Centroid

The formula for finding out the Centroid (V) is

$$V_{ij} = \frac{\sum_{k=1}^n Y_{ik}^m \times x_k}{\sum_{k=1}^n Y_{ik}^m}$$

where Y : fuzzy membership value

m : fuzziness parameter generally taken as 2

x_k : is the data point

$$V_{11} = \frac{(0.8^2 * 1 + 0.7^2 * 2 + 0.2^2 * 4 + 0.1^2 * 7)}{0.8^2 + 0.7^2 + 0.2^2 + 0.1^2}$$

$$= 1.568$$

$$V_{12} = \frac{(0.8^2 * 3 + 0.7^2 * 5 + 0.2^2 * 8 + 0.1^2 * 9)}{(0.8^2 + 0.7^2 + 0.2^2 + 0.1^2)}$$

$$= 4.051$$

$$V_{21} = \frac{(0.2^2 * 1 + 0.3^2 * 2 + 0.8^2 * 4 + 0.9^2 * 7)}{(0.2^2 + 0.3^2 + 0.8^2 + 0.9^2)}$$

$$= 5.35$$

$$V_{22} = \frac{(0.2^2 * 3 + 0.3^2 * 5 + 0.8^2 * 8 + 0.9^2 * 9)}{(0.2^2 + 0.3^2 + 0.8^2 + 0.9^2)}$$

$$= 8.215$$

Centroids are (1.568, 4.051) and

(5.35, 8.215)

Steps 3: Find out the distance of Each points from the Centroid

Cluster	(1,3)	(2,5)	(4,8)	(7,9)
1	0.8	0.7	0.2	0.1
2	0.2	0.3	0.8	0.9

$$D_{11} = \sqrt{(1 - 1.568)^2 + (3 - 4.051)^2} = 1.2$$

$$D_{12} = \sqrt{(1 - 5.35)^2 + (3 - 8.215)^2} = 6.79$$

$$D_{21} = \sqrt{(2 - 1.568)^2 + (5 - 4.051)^2} = 1.04$$

$$D_{22} = \sqrt{(2 - 5.35)^2 + (5 - 8.215)^2} = 4.64$$

$$D_{31} = \sqrt{(4 - 1.568)^2 + (8 - 4.051)^2} = 4.63$$

$$D_{32} = \sqrt{(4 - 1.568)^2 + (8 - 8.215)^2} = 1.36$$

$$D_{31} = \sqrt{(7-1.568)^2 + (9-4.051)^2} = 7.34$$

$$D_{32} = \sqrt{(7-5.35)^2 + (9-8.215)^2} = 1.82$$

Cluster	(1,3)	(2,5)	(4,8)	(7,9)	Cluster Assignment
1	0.8	0.7	0.2	0.1	
2	0.2	0.3	0.8	0.9	

Step 4 :- Updating membership values :-

$$Y_{ki} = \left(\sum_{j=1}^n \left\{ \frac{d_{kj}^2}{d_{ki}^2} \right\} \left(\frac{1}{(m-1)} \right) \right)^{-1} \quad \begin{matrix} i = \text{Constant} \\ k = \text{data pt} \end{matrix}$$

for point 1 new membership values are

$$Y_{11} = \left(\left\{ \frac{(1.2)^2}{(1.2)^2} + \frac{(1.2)^2}{(6.79)^2} \right\} \left(\frac{1}{(2-1)} \right) \right)^{-1} = 0.97$$

$$Y_{12} = \left(\left\{ \frac{(6.79)^2}{(1.2)^2} + \frac{(6.79)^2}{(6.79)^2} \right\} \left(\frac{1}{(2-1)} \right) \right)^{-1} = 0.03$$

for point 2 new membership values are

$$Y_{21} = \left(\left\{ \frac{(1.04)^2}{(1.04)^2} + \frac{(1.04)^2}{(4.64)^2} \right\} \left(\frac{1}{(2-1)} \right) \right)^{-1} = 0.95$$

$$Y_{12} = \left(\left\{ \frac{(1.36)^2}{(7.34)^2} + \text{etc.} \right\} \right)$$

$$Y_{22} = \left(\left\{ \frac{(4.64)^2}{(1.04)^2} + \frac{(4.64)^2}{(4.64)^2} \right\} \left(\frac{1}{(2-1)} \right) \right)^{-1} = 0.05$$

for point 3 new membership values are.

$$Y_{31} = \left(\left\{ \frac{(4.63)^2}{(4.63)^2} + \frac{(4.63)^2}{(1.36)^2} \right\} \left(\frac{1}{(2-1)} \right) \right)^{-1} = 0.08$$

$$Y_{32} = \left(\left\{ \frac{(1.36)^2}{(4.63)^2} + \frac{(1.36)^2}{(1.36)^2} \right\} \left(\frac{1}{(2-1)} \right) \right)^{-1} = 0.92$$

for point 4 new membership values are

$$Y_{41} = \left(\left\{ \frac{(7.34)^2}{(7.34)^2} + \frac{(7.34)^2}{(1.82)^2} \right\} \left(\frac{1}{(2-1)} \right) \right)^{-1} = 0.06$$

$$Y_{42} = \left(\left\{ \frac{(1.82)^2}{(7.34)^2} + \frac{(1.82)^2}{(1.82)^2} \right\} \left(\frac{1}{(2-1)} \right) \right)^{-1} = 0.94$$

cluster	(1,3)	(2,5)	(4,8)	(7,9)
c_1	0.97	0.95	0.08	0.06
c_2	0.03	0.05	0.92	0.94

Step 5: Repeat the steps (2-4) until the constant values are obtained for the membership values or the difference is less than the tolerance value.

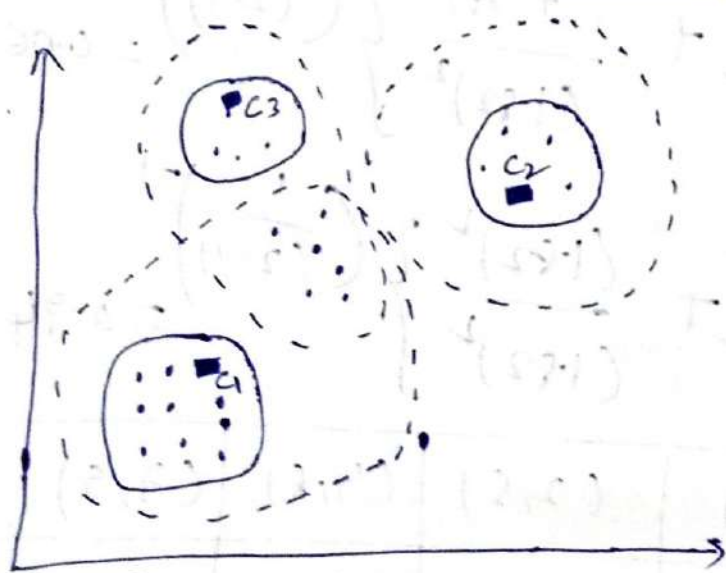
* Rough Clustering:

→ Rough clustering divides a set of data points into several rough clusters. A rough cluster consists of the "lower and upper" approximation.

→ Objects in the lower approximation of a cluster belong to this cluster only, while objects in the upper approximation will also be members of upper approximation of other clusters.

→ Like K-means clustering, an iterative process is required for rough K-means to generate a set of cluster representation.

→ uncertainty zone where objects belong to multiple clusters is known as "Boundary Region".



• = Data points

dotted boundaries (---)

= upper approximation

lined boundary (—)

Lower approximation

* Rough K-Means Clustering Algorithm:-

→ K means algorithm is an "unsupervised learning algorithm".

→ Given a dataset of items, with certain features, and values for these features, the algorithm will categorize the items into 'K' groups or clusters of similarity.

→ To calculate the similarity, we can use the Euclidean distance, Manhattan distance, Hamming distance as measurement.

Implementation steps:-

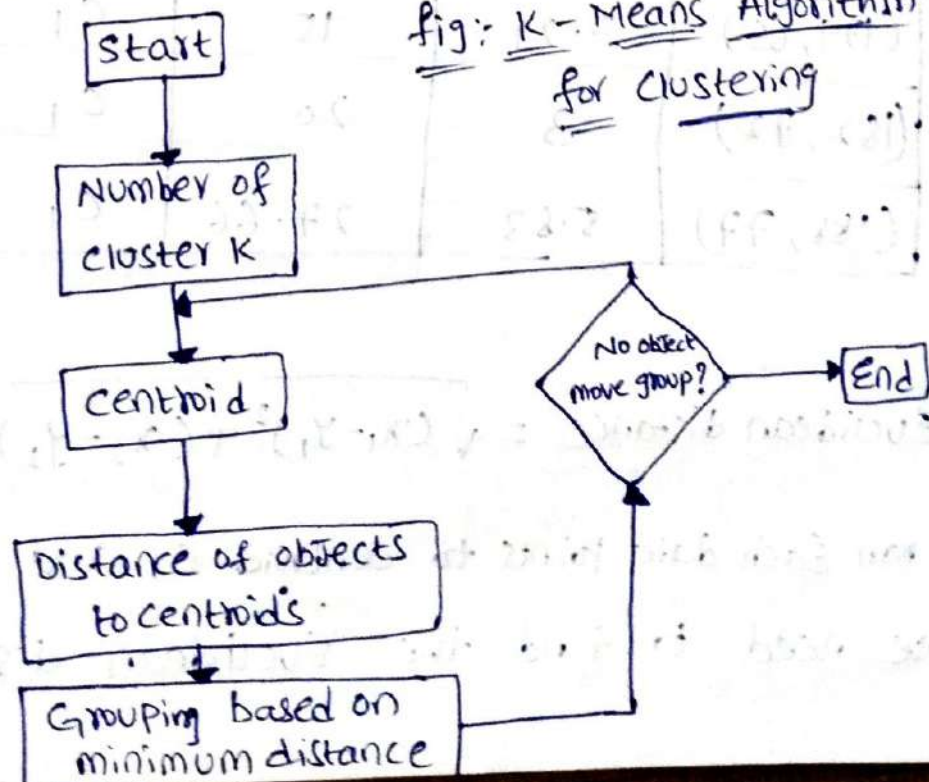
1) Choose 'K' number of random data points as initial centroids (cluster centers).

2) Repeat till cluster centers stabilize.

a) Allocate each point into the nearest of K^{th} centroids.

b) Compute Centroid for the cluster using all points in the cluster.

fig: K-Means Algorithm for clustering



Example: Given data points

$(185, 72), (170, 56), (168, 60), (179, 68), (182, 72),$
 $(188, 77)$

→ The distance function used is "Euclidean distance"

→ First two data points as initial centroids.

initial centroids:

$C_1 : (185, 72)$

$C_2 : (170, 56)$

Data points	Distance to		Cluster	New Cluster
	C_1 $(185, 72)$	C_2 $(170, 56)$		
$(185, 72)$	0	21.93	C_1	
$(170, 56)$	21.93	0	C_2	
$(168, 60)$	20.81	4.47	C_2	
$(179, 68)$	7.21	15	C_1	
$(182, 72)$	3	20	C_1	
$(188, 77)$	5.83	27.66	C_1	

$$\text{Euclidean distance} = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$

From Each data points to centroid C_1 and centroid C_2
we need to find the Euclidean distance.

From data point $(185, 72)$ to centroid $C_1 (185, 72)$

$$d(d_1, C_1) = \sqrt{(185 - 185)^2 + (72 - 72)^2} = 0$$

$$d(d_2, C_1) = \sqrt{(170 - 185)^2 + (56 - 72)^2} = 21.93$$

$$d(d_3, C_1) = \sqrt{(168 - 185)^2 + (60 - 72)^2} = 20.81$$

$$d(d_4, C_1) = \sqrt{(179 - 185)^2 + (68 - 72)^2} = 7.21$$

$$d(d_5, C_1) = \sqrt{(182 - 185)^2 + (72 - 72)^2} = 3$$

$$d(d_6, C_1) = \sqrt{(188 - 185)^2 + (77 - 72)^2} = 5.83$$

From data points to centroid $C_2 (170, 56)$ we need to find the distance.

$$d(d_1, C_2) = \sqrt{(185 - 170)^2 + (72 - 56)^2} = 21.93$$

$$d(d_2, C_2) = \sqrt{(170 - 170)^2 + (56 - 56)^2} = 0$$

$$d(d_3, C_2) = \sqrt{(168 - 170)^2 + (60 - 56)^2} = 4.47$$

$$d(d_4, C_2) = \sqrt{(179 - 170)^2 + (68 - 56)^2} = 15$$

$$d(d_5, C_2) = \sqrt{(182 - 170)^2 + (72 - 56)^2} = 20$$

$$d(d_6, C_2) = \sqrt{(188 - 170)^2 + (77 - 56)^2} = 27.66$$

Now $(185, 72)$
 $(179, 68)$
 $(182, 72)$
 $(188, 77)$

Belongs to cluster 1

$(170, 56)$
 $(168, 60)$

Belongs to cluster 2

Now we need to find new Centroid

$$\text{Centroid } d_1 = \frac{185 + 179 + 182 + 188}{4} = 183.5$$

for

$$\text{Centroid } d_2 = \frac{72 + 68 + 72 + 77}{4} = 72.25$$

for

from cluster C_1 new Centroid values $C_1 (183.5, 72.25)$

Now we need to perform for cluster 2

$$\text{Centroid } d_1 = \frac{170 + 168}{2} = 169$$

for

$$\text{Centroid } d_2 = \frac{56 + 60}{2} = 58$$

for

from Cluster 2 new Centroid values $C_2 (169, 58)$

New centroids :

$$C_1 : (183.5, 72.25)$$

$$C_2 : (169, 58)$$

Now we need to find distance from data points to new Centroid values C_1

$$d(d_1, C_1) = \sqrt{(185 - 183.5)^2 + (72 - 72.25)^2} = 1.52$$

$$d(d_2, C_1) = \sqrt{(170 - 183.5)^2 + (56 - 72.25)^2} = 21.13$$

$$d(d_3, C_1) = \sqrt{(168 - 183.5)^2 + (60 - 72.25)^2} = 19.46$$

$$d(d_4, c_1) = \sqrt{(179 - 183.5)^2 + (68 - 72.65)^2} = 6.19$$

$$d(d_5, c_1) = \sqrt{(182 - 183.5)^2 + (72 - 72.65)^2} = 1.52$$

$$d(d_6, c_1) = \sqrt{(188 - 183.5)^2 + (77 - 72.65)^2} = 6.54$$

Now we need to find distance from data points to new centroid values c_2

$$d(d_1, c_2) = \sqrt{(185 - 169)^2 + (72 - 86)^2} = 22.02$$

$$d(d_2, c_2) = \sqrt{(170 - 169)^2 + (56 - 86)^2} = 2.83$$

$$d(d_3, c_2) = \sqrt{(168 - 169)^2 + (60 - 86)^2} = 2.00$$

$$d(d_4, c_2) = \sqrt{(179 - 169)^2 + (68 - 86)^2} = 14.87$$

$$d(d_5, c_2) = \sqrt{(182 - 169)^2 + (72 - 86)^2} = 19.80$$

$$d(d_6, c_2) = \sqrt{(188 - 169)^2 + (77 - 86)^2} = 27.59$$

Data points	Distance to		cluster	New cluster
	$c_1(183.5, 72.25)$	$c_2(169, 86)$		
$(185, 72)$	1.52	22.02	c_1	c_1
$(170, 56)$	21.13	2.83	c_2	c_2
$(168, 60)$	19.76	2.00	c_2	c_2
$(179, 68)$	6.19	14.87	c_1	c_1
$(182, 72)$	1.52	19.80	c_1	c_1
$(188, 77)$	6.54	27.59	c_1	c_1

* Expectation maximization based clustering

→ The Expectation-maximization algorithm can be used for the latent variables (variables that are not directly observable and are actually inferred from the values of the other observed variables)

→ This algorithm is actually the base for many unsupervised clustering algorithms in the field of Machine learning.

→ In the real-world applications of ML, it is very common that there are many relevant features available for learning but only a small subset of them are observable.

Let us understand the EM algorithm in detail :-

1) Initially, a set of initial values of the parameters are considered. A set of incomplete observed data is given to the system with the assumption that the observed data comes from a specific model.

2) The next step is known as "Expectation" step (or) E-step. In this step, we use the observed data in order to estimate or guess the values of the missing or incomplete data. It is basically used to update the variables.

3) The next step is known as "maximization" step (or) M-step. In this step we use the complete data generated in the preceding Expectation step in order to update the values of the parameters. It is basically used to update the hypothesis.

4) Now, in the fourth step, it is checked, whether the values are converging or not, if yes then "stop" otherwise repeat step 2 and step 3 i.e. E-step and M-step until the convergence occurs.

EM Algorithm:

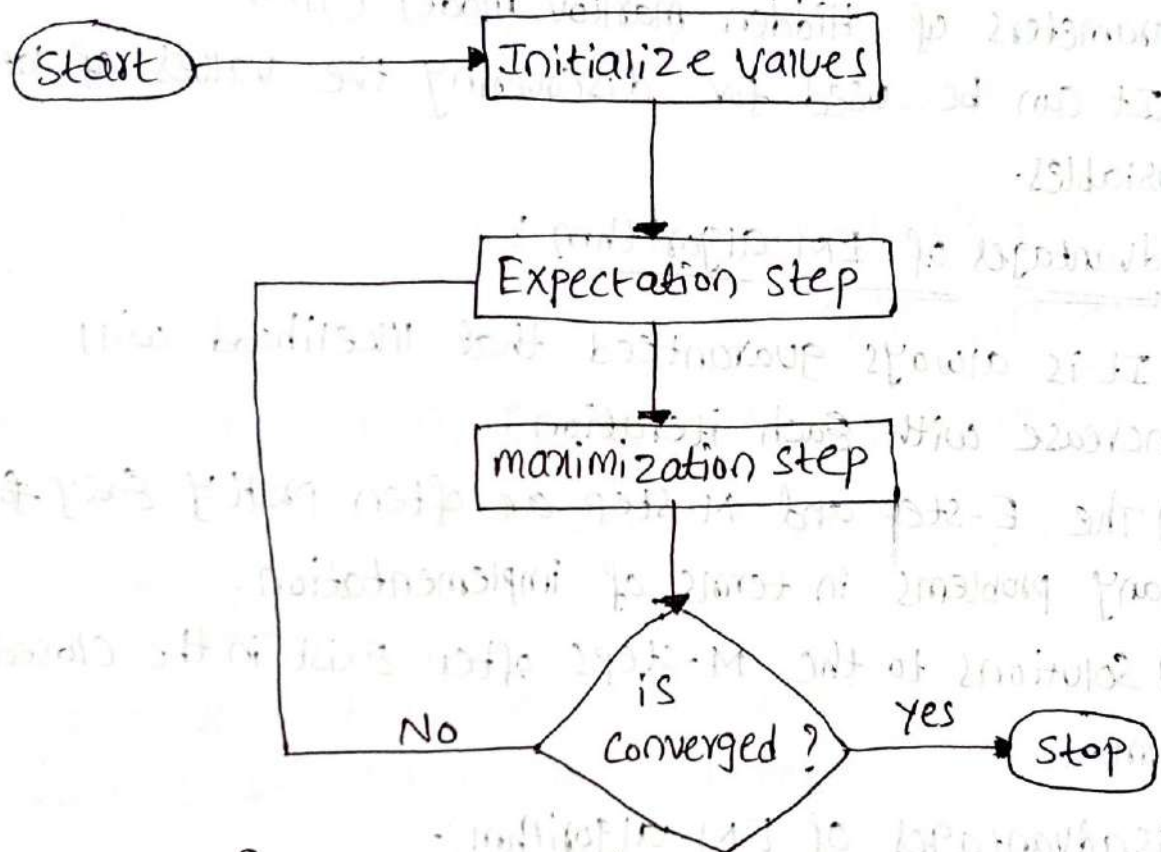


Fig: Expectation - maximization

- 1) Given a set of incomplete data, Consider a set of starting parameters:
- 2) Expectation step (E-step): using the observed available data of the dataset, estimate (guess) the values of the missing data.
- 3) Maximization step (M-step): Complete data generated after the Expectation (E) step is used in order to update the parameters.
- 4) Repeat step 2 & step 3 until Converged.

Usage of EM algorithm:-

- 1) It can be used to fill the missing data in a sample.
- 2) It can be used as the basis of unsupervised learning of clusters.
- 3) It can be used for the purpose of estimating the parameters of Hidden Markov Model (HMM)
- 4) It can be used for discovering the values of latent variables.

Advantages of EM algorithm:-

- 1) It is always guaranteed that likelihood will increase with each iteration.
- 2) The E-step and M-step are often pretty easy for many problems in terms of implementation.
- 3) Solutions to the M-steps often exist in the closed form.

Disadvantages of EM algorithm:-

- 1) It has slow convergence
- 2) It makes convergence to the local optimal only.
- 3) It requires both the probabilities, forward and backward (numerical optimization requires only forward probability)

Key points:-

- 1) EM - a very popular technique for estimating parameters of probabilistic models.
- 2) many popular algorithms like Hidden Markov Model (HMM), Gaussian mixtures, etc., uses

EM techniques:

3) It is beneficial when working with data that is incomplete, has missing data points or has unobserved latent variables.

Example: Assume that we have two coins, C_1 and C_2

→ Assume the bias of C_1 is θ_1 (probability of getting heads with C_1).

→ Assume the bias of C_2 is θ_2 (probability of getting heads with C_2).

→ We want to find θ_1, θ_2 by performing a number of trials (coin tosses).

First Experiment

- we choose 5 times one of the coins
- we toss the chosen coin 10 times

③ H T T T H H T H T H

① H H H H T H H H H H

① H T H H H H H T H H

③ H T H T T T H H T T

① T H H H T H H H T H

Coin A	Coin B
	5 H, 5 T
9 H, 1 T	
8 H, 2 T	
	4 H, 6 T
7 H, 3 T	

total = 24 H, 6 T

9 H, 11 T

$$\theta_1 = \frac{\text{Number of heads using } C_1}{\text{total no. of flips using } C_1} = \frac{24}{24+6} = 0.8$$

$$\theta_2 = \frac{\text{no. of heads using } C_2}{\text{total no. of flips using } C_2} = \frac{9}{9+11} = 0.45$$

* EM: It is an iterative method used in unsupervised ML to estimate unknown parameters in statistical models.

E-step: Estimates missing or hidden values using current parameter estimates.

M-step: updates model parameters to maximize the likelihood based on the estimated value.

* Spectral Clustering:

However, we can also use connectivity between the datapoints as a feature to cluster our data points.

→ Using connectivity we can cluster two data points into the same clusters even if the distance b/w the two data points is larger.

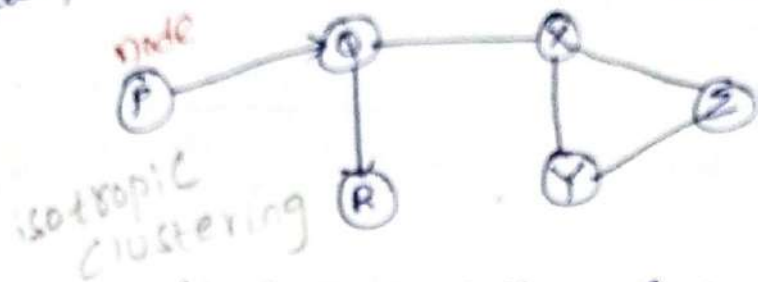
S.C: It is a variant of the clustering algorithm that uses the connectivity between the data points to form the clustering.

- 1) Eigen values
 - 2) Eigen vectors
- } data Matrix to forecast the data into lower dimensions

Graph representation of data.

where the data points are represented as nodes & similarity b/w

data points are represented by edges



Graphical representation of a dataset with six vertices.

Eigen value — It is a scalar value that is associated with a linear transformation of vector space

$$AV = \lambda V$$

where 'A' is the Square Matrix

'V' is the Eigen vector

' λ ' is the Eigen value.

Eigen vectors — are the directions that remain unchanged during a transformation, even if they get longer or shorter.

Types of Eigen vector

Right Eigen vector

$$AV_R = \lambda V_R$$

left Eigen vect

$$AV_L = \lambda V_L$$

* Support Vector Machines (SVM):

→ SVM is one of the most popular supervised learning algorithms, which is used for classification as well as Regression problems.

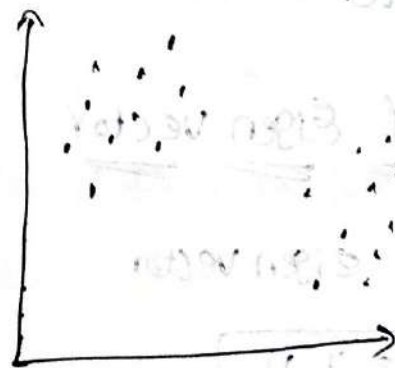
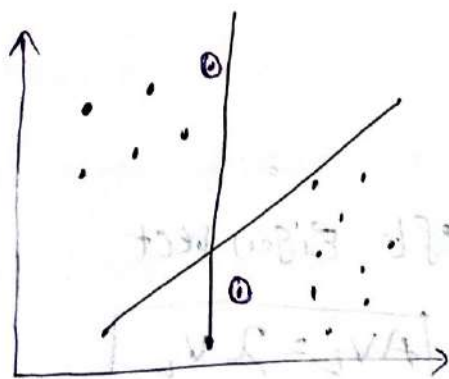
→ However, primarily, it is used for classification problems in ML.

→ The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n -dimensional space into classes so that we can easily put the new datapoint in the correct category in the future.

→ This best decision boundary is called a "hyperplane".

→ There can be multiple lines / decision boundaries to segregate the classes in n -dimensional space, but we need to find out the best decision boundary that helps to classify the data points.

→ This best boundary is known as the hyperplane of SVM.



→ The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2-features, then hyperplane will be a straight line.

→ And if there are 3 features, then hyperplane will

be a 2-dimension plane.

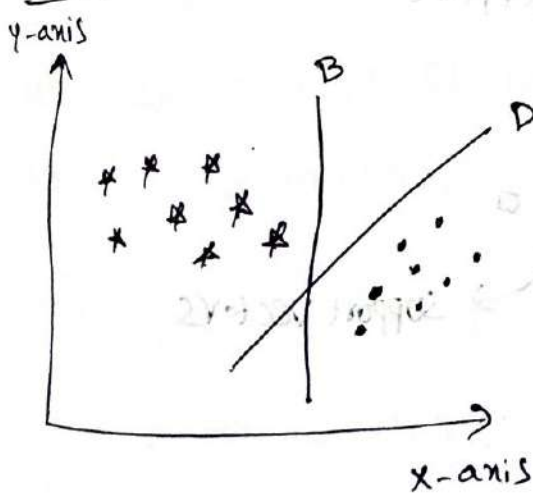
→ we always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

SVM can be of two types:

1) Linear SVM: It is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data and classifier is used as linear SVM classifier.

2) Non Linear SVM: It is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as non-linear SVM classifier.

Example:- Linear SVM.



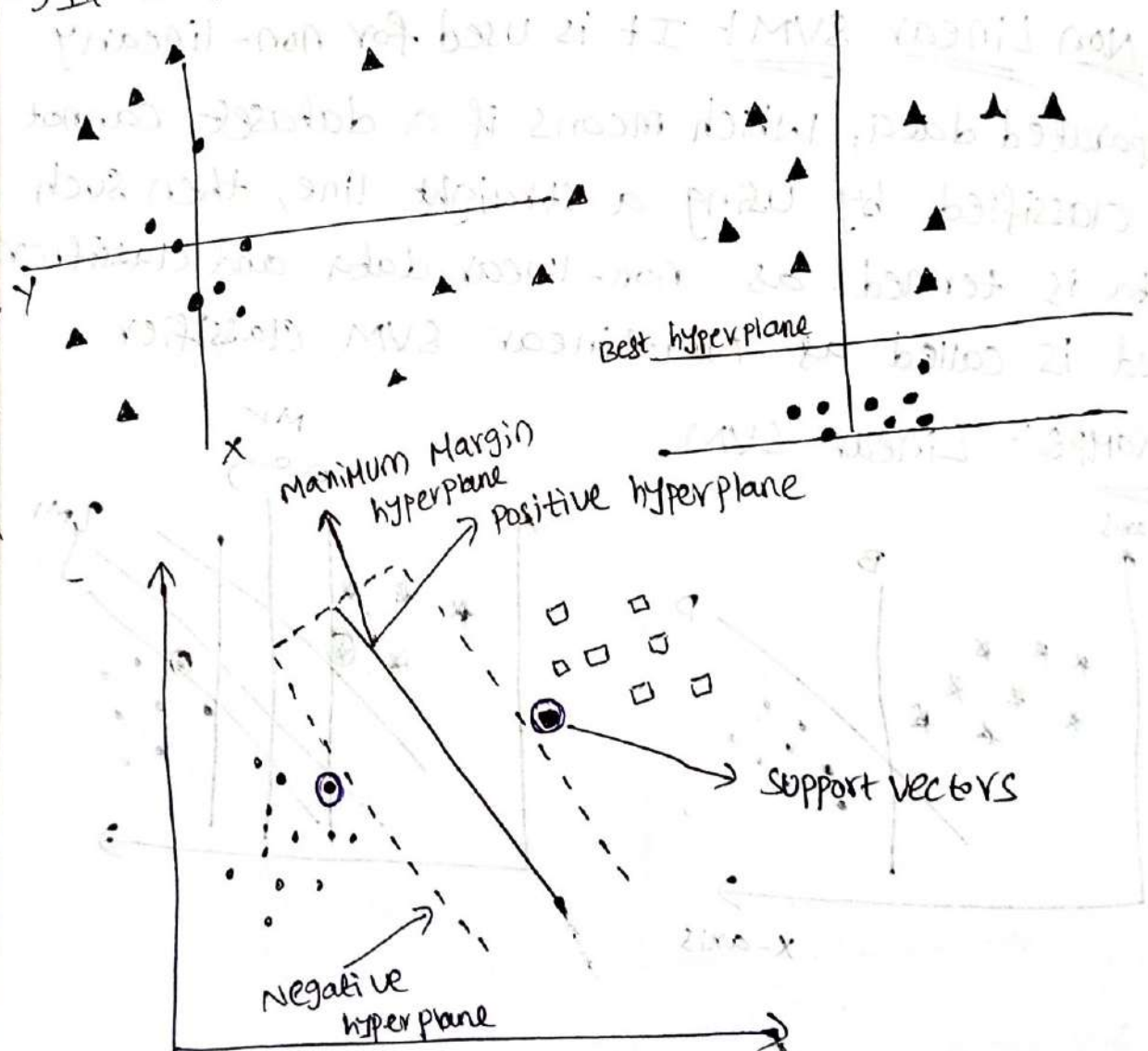
② Non-Linear SVM:

→ if data is linearly arranged; then we can separate it by using a straight line, but for non-linear data, we cannot draw a straight line.

→ So to separate these data points, we need to add one more dimension.

→ For linear data, we have used two dimensions x and y , so for non-linear data, we will add a third dimension z .

→ It can be calculated as $z = \sqrt{x^2 + y^2}$



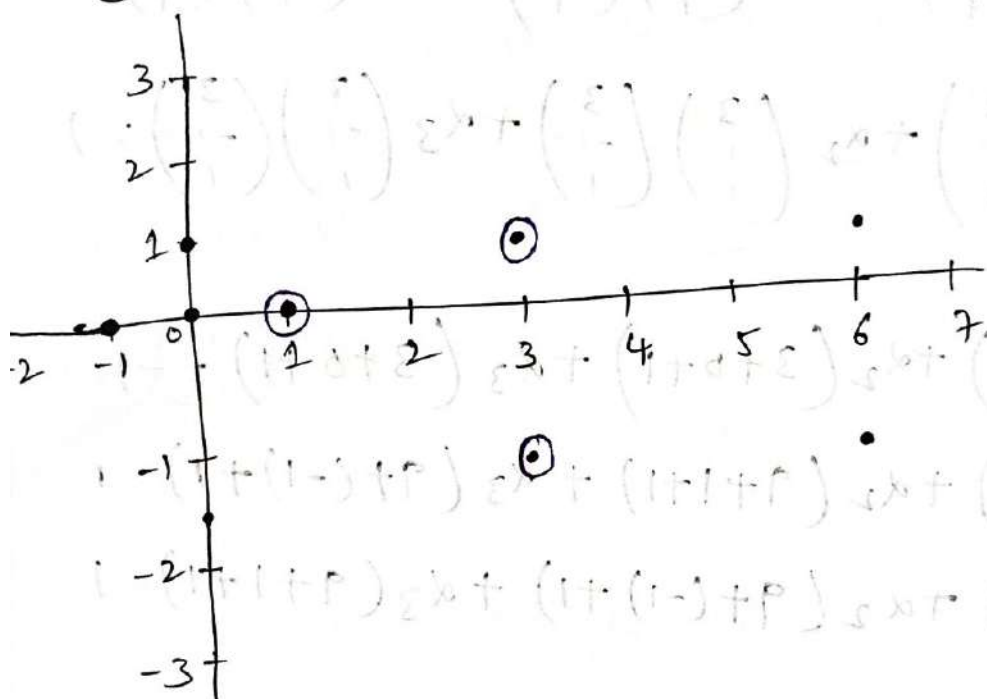
Linear SVM?

suppose we are given the following positively labeled data points.

$$\left\{ \begin{pmatrix} 3 \\ 1 \end{pmatrix}, \begin{pmatrix} 3 \\ -1 \end{pmatrix}, \begin{pmatrix} 6 \\ 1 \end{pmatrix}, \begin{pmatrix} 6 \\ -1 \end{pmatrix} \right\}$$

and the following negatively labeled data points.

$$\left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \end{pmatrix} \right\}$$



By inspection, it should be obvious that there are three support vectors,

$$s_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad s_2 = \begin{pmatrix} 3 \\ 1 \end{pmatrix} \quad s_3 = \begin{pmatrix} 3 \\ -1 \end{pmatrix}$$

Each vector is augmented with a 1 as a bias input so

$$s_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ then } \bar{s}_1 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \text{ similarly}$$

$$s_2 = \begin{pmatrix} 3 \\ 1 \end{pmatrix} \text{ then } \bar{s}_2 = \begin{pmatrix} 3 \\ 1 \\ 1 \end{pmatrix} \quad s_3 = \begin{pmatrix} 3 \\ -1 \end{pmatrix} \quad \bar{s}_3 = \begin{pmatrix} 3 \\ -1 \\ 1 \end{pmatrix}$$

$$\alpha_1 \bar{S}_1 \cdot \bar{S}_1 + \alpha_2 \bar{S}_2 \cdot \bar{S}_1 + \alpha_3 \bar{S}_3 \cdot \bar{S}_1 = -1$$

$$\alpha_1 \bar{S}_1 \cdot \bar{S}_2 + \alpha_2 \bar{S}_2 \cdot \bar{S}_2 + \alpha_3 \bar{S}_3 \cdot \bar{S}_2 = +1$$

$$\alpha_1 \bar{S}_1 \cdot \bar{S}_3 + \alpha_2 \bar{S}_2 \cdot \bar{S}_3 + \alpha_3 \bar{S}_3 \cdot \bar{S}_3 = +1$$

$$\alpha_1 \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + \alpha_2 \begin{pmatrix} 3 \\ 1 \\ 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + \alpha_3 \begin{pmatrix} 3 \\ -1 \\ 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = -1$$

$$\alpha_1 \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \begin{pmatrix} 3 \\ 1 \\ 1 \end{pmatrix} + \alpha_2 \begin{pmatrix} 3 \\ 1 \\ 1 \end{pmatrix} \begin{pmatrix} 3 \\ 1 \\ 1 \end{pmatrix} + \alpha_3 \begin{pmatrix} 3 \\ -1 \\ 1 \end{pmatrix} \begin{pmatrix} 3 \\ 1 \\ 1 \end{pmatrix} = 1$$

$$\alpha_1 \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \begin{pmatrix} 3 \\ -1 \\ 1 \end{pmatrix} + \alpha_2 \begin{pmatrix} 3 \\ 1 \\ 1 \end{pmatrix} \begin{pmatrix} 3 \\ -1 \\ 1 \end{pmatrix} + \alpha_3 \begin{pmatrix} 3 \\ -1 \\ 1 \end{pmatrix} \begin{pmatrix} 3 \\ -1 \\ 1 \end{pmatrix} = 1$$

$$\alpha_1 (1+0+1) + \alpha_2 (3+0+1) + \alpha_3 (3+0+1) = -1$$

$$\alpha_1 (3+0+1) + \alpha_2 (9+1+1) + \alpha_3 (9+(-1)+1) = 1$$

$$\alpha_1 (3+0+1) + \alpha_2 (9+(-1)+1) + \alpha_3 (9+1+1) = 1$$

$$2\alpha_1 + 4\alpha_2 + 4\alpha_3 = -1$$

$$4\alpha_1 + 11\alpha_2 + 9\alpha_3 = 1$$

$$4\alpha_1 + 9\alpha_2 + 11\alpha_3 = 1$$

$$\alpha_1 = -3.5$$

$$\alpha_2 = 0.75$$

$$\alpha_3 = 0.75$$

weight vector

$$\bar{W} = \sum_i \alpha_i \bar{S}_i$$

$i = \text{Neighbors data points}$
(3)

$$= -3.5 \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + 0.75 \begin{pmatrix} 3 \\ 1 \\ 1 \end{pmatrix} + 0.75 \begin{pmatrix} 3 \\ -1 \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 \\ 0 \\ -2 \end{pmatrix}$$

Finally, remembering that our vectors are augmented with a bias.

→ We can equate the last entry in \bar{W} as the hyperplane offset by write the separating.

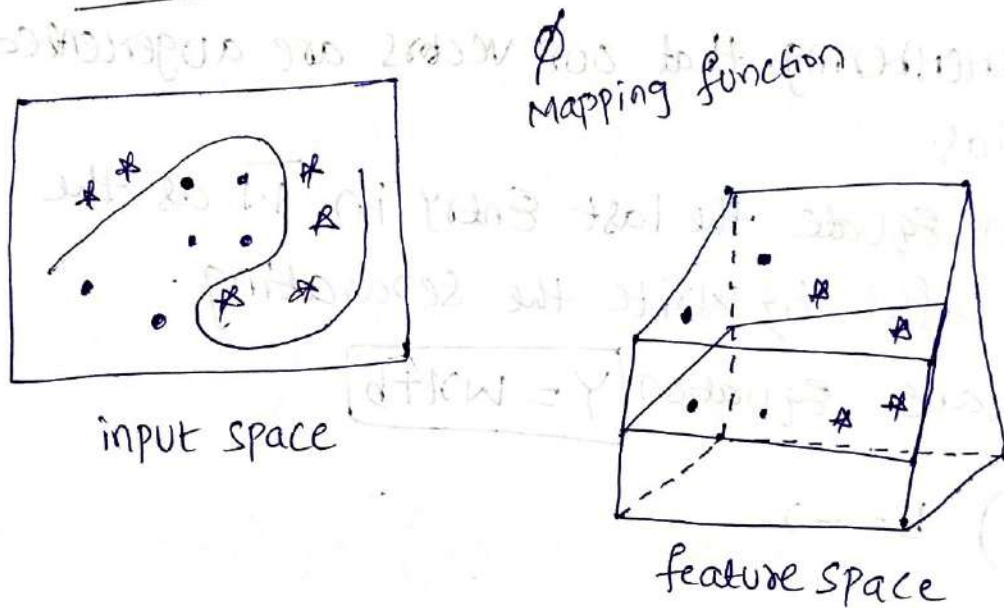
→ Hyperplane Equation $Y = WX + b$

$$W = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad b = -2$$

* kernel trick :

In ML applications, the data can be any format it may be text, image, or video. So there is a need to Extract features from these data prior to classification.

→ Hence, in the real world, many Classification models are complex and mostly require non-linear hyperplanes



For example, one mapping function $\phi: \mathbb{R}^2 \rightarrow \mathbb{R}^3$ used to transform a 2D data into 3D data is given as follows.

$$\phi(x, y) = (x^2, \sqrt{2}xy, y^2)$$

→ Consider a point $(2, 3)$ in 2D space, if you apply above mapping function we can convert it into 3D space and it looks like this, Here $x=2$ & $y=3$

→ Hence Datapoints in 3D space is

$$\phi(2, 3) = (2^2, \sqrt{2} * 2 * 3, 3^2) = (4, 6\sqrt{2}, 9)$$

→ By doing these the non-linear data should be converted (or) transformed to linear data.

→ while mapping functions play an important role, there are many disadvantages, as mapping involves more Computations and learning Costs.

→ Also, disadvantages of transformations are that there is no generalized thumb rule available describing what transformations should be applied and if the data is large, mapping function process take huge amount of time.

→ In real applications, there might be many features in the data and applying transformations that involves many polynomial combinations of these features will lead to Extremely high & impractical Computational Cost.

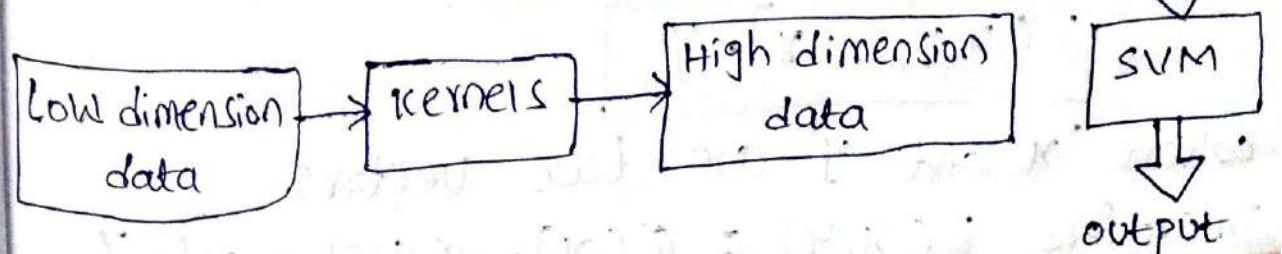
→ In this context, only kernels are useful

→ kernels are used to compute the value without transforming the data.

What is a kernel? :-

→ kernels are a set of functions used to transform data from lower dimensions to higher dimensions and to manipulate data using dot product at higher dimensions.

→ The use of kernels is to apply transformation to data & perform classification at the higher dimension as shown in figure.



kernel trick for 2^{nd} degree polynomial mapping

$$\begin{aligned}\phi(a)^T \cdot \phi(b) &= \begin{bmatrix} a_1^2 \\ \sqrt{2}a_1a_2 \\ a_2^2 \end{bmatrix}^T \cdot \begin{bmatrix} b_1^2 \\ \sqrt{2}b_1b_2 \\ b_2^2 \end{bmatrix} \\ &= a_1^2 b_1^2 + 2a_1b_1a_2b_2 + a_2^2 b_2^2 \\ &= (a_1b_1 + a_2b_2)^2 = \left[\begin{bmatrix} a_1 \\ a_2 \end{bmatrix}^T \cdot \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \right]^2 \\ &= (a^T \cdot b)^2\end{aligned}$$

Here a, b are data points. 'a' contains two features a_1, a_2 and 'b' contains two features b_1, b_2 by using mapping function $\phi(x, y) = (x^2, \sqrt{2}xy, y^2)$

Types of kernels:-

- 1) linear kernel
- 2) Polynomial kernel
- 3) Exponential kernel
- 4) Homogeneous kernel
- 5) Gaussian kernel etc.,

① Linear kernel: Linear kernels are of the type

$$K(x, y) = x^T \cdot y$$

where 'x' and 'y' are two vectors

therefore $K(x, y) = \phi(x) \cdot \phi(y) = x^T \cdot y$

② polynomial kernel :- polynomial kernels are of the type:

$$K(x, y) = (x^T y)^q$$

→ This is called homogeneous kernel

Here 'q' is the degree of the polynomial

if $q=2$ then it is called "quadratic kernel"

→ For inhomogeneous kernels, this is given as

$$K(x, y) = (C + x^T y)^q$$

→ Here 'C' is a constant and 'q' is the degree of the polynomial.

→ if 'C' is zero and degree is one, the polynomial kernel is reduced to a linear kernel.

→ The value of degree 'q' should be optimal as more degree may lead to overfitting.

Example:- Consider two datapoints $x = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$ and

$y = (2, 3)$. Apply linear homogeneous, and inhomogeneous kernels.

→ if $q=2$; $C=1$ it is called inhomogeneous kernel.

$$\begin{aligned} K(x, y) &= (C + x^T y)^q = \left(1 + \begin{pmatrix} 1 \\ 2 \end{pmatrix}^T (2, 3) \right)^2 \\ &= (1 + 8)^2 = 9^2 = 81 \end{aligned}$$

* Logistic Regression !

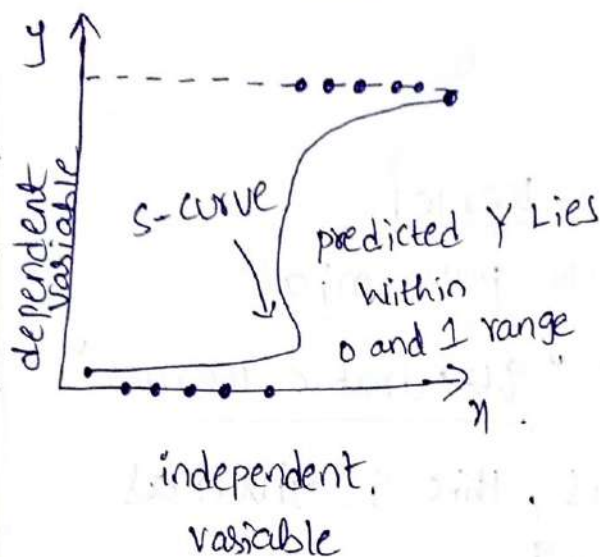


Fig: Logistic Regression

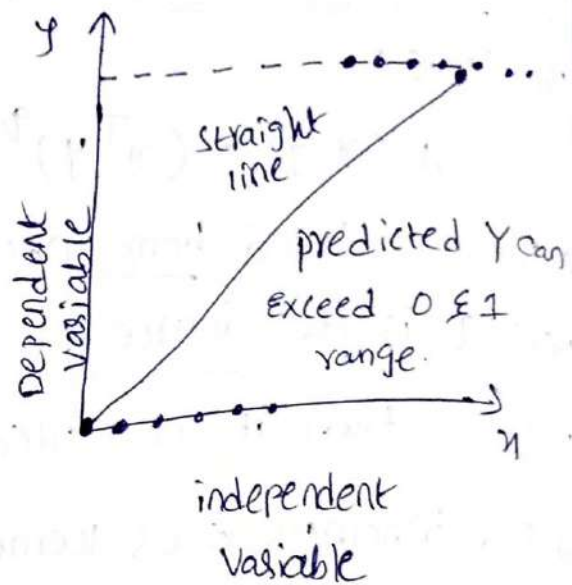


Fig: Linear Regression

Linear regression predicts the numerical response but is not suitable for predicting the categorical variables.

→ When categorical variables are involved, it is called classification problem.

→ Logistic regression is suitable for binary.

Classification problem:

For Example, The following scenarios are instances of predicting categorical variables:

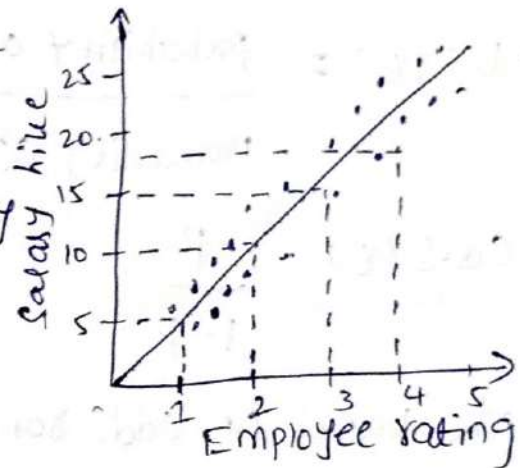
- 1) Is the mail spam or not spam? The answer is yes or no. Thus, categorical dependent variables is a binary response of yes or no.
- 2) If the student should be admitted or not is based on Entrance Examination marks. Here categorical variable response is admitted or not.
- 3) The student being pass or fail is based on

marks secured.

How does the logistic regression algorithm work:-

→ consider the following Example.

an. ① An organization wants to determine an Employee's Salary increase based on their performance.

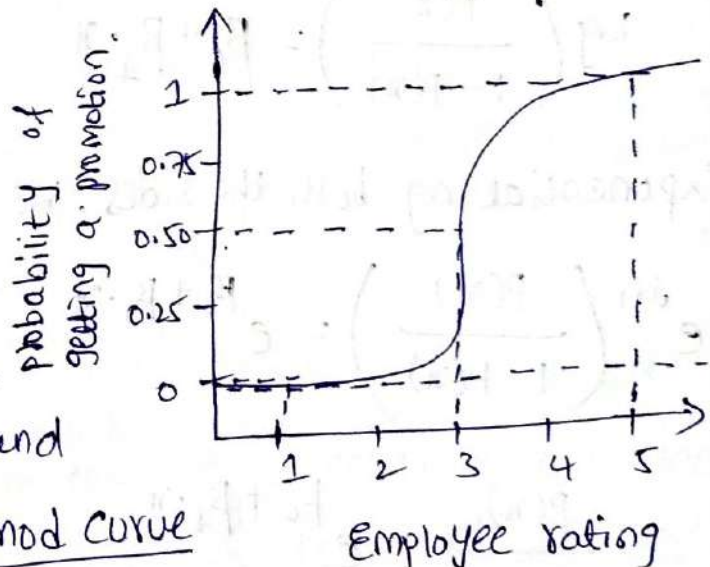


② For this purpose, a linear regression algorithm will help them decide.

not ③ plotting a regression line by considering the Employee's performance as the independent variable, and the salary increase as the dependent variable will make their task easier.

ed. ④ Now, what if the organization wants to know whether an Employee would get promotion or not based on their performance.

⑤ The above linear graph won't be suitable in this case.



⑥ As such, we clip the line at zero and one, and convert it into a sigmoid curve (S curve).

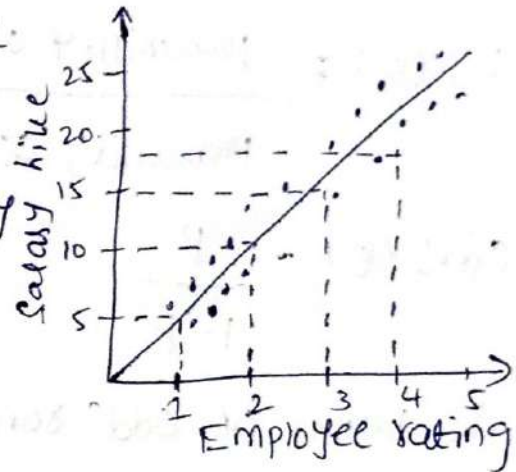
used ⑦ Based on the threshold values, the organization can decide whether the Employee will get a salary increase or not.

marks secured.

How does the logistic regression algorithm work?

→ consider the following Example.

- an
- ① An organization wants to determine an Employee's Salary increase based on their performance.

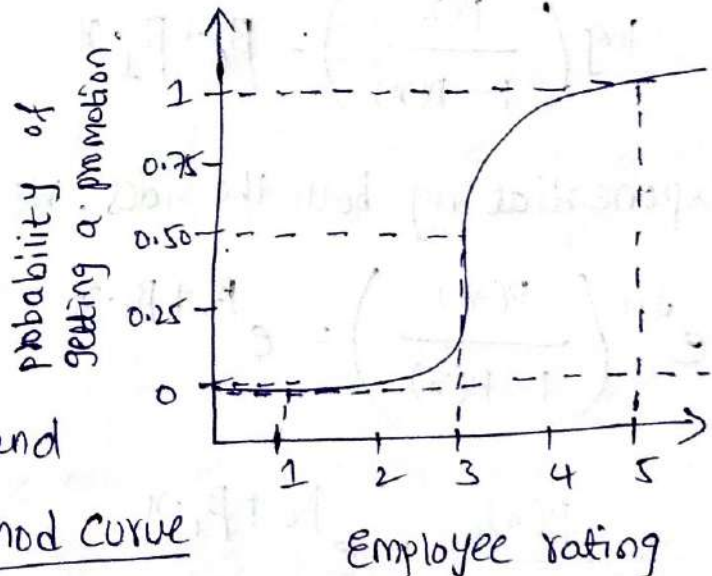


- ② For this purpose, a linear regression algorithm will help them decide.

- ③ plotting a regression line by considering the Employee's performance as the independent variable, and the salary increase as the dependent variable will make their task easier.

- ↓
- ④ Now, what if the organization wants to know whether an Employee would get promotion or not based on their performance.

- ⑤ The above linear graph won't be suitable in this case.



- ⑥ As such, we clip the line at zero and one, and convert it into a sigmoid curve (S curve).

- ↓
- ⑦ Based on the threshold values, the organization can decide whether the Employee will get a salary increase or not.

To understand logistic regression by using Mathematical Equation, Let's go over the odds of success

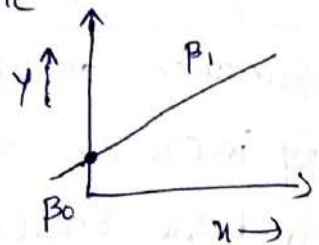
$$\text{odds}(\theta) = \frac{\text{probability of an event happening}}{\text{probability of an event not happening}}$$

$$\text{odds}(\theta) = \frac{P}{1-P}$$

→ The values of odds range from zero to ∞ and the values of probability lies b/w zero and one.

→ Consider the Equation of a straight line

$$Y = \beta_0 + \beta_1 * x$$



Now to predict the odds of success, we take log on odds formula:

$$\log\left(\frac{P(x)}{1-P(x)}\right) = \beta_0 + \beta_1 \cdot x$$

Exponentiating both the sides, we have

$$e^{\ln\left(\frac{P(x)}{1-P(x)}\right)} = e^{\beta_0 + \beta_1 \cdot x}$$

$$e^{\ln(x)} = x$$

$$\frac{P(x)}{1-P(x)} = e^{\beta_0 + \beta_1 \cdot x}$$

$$\text{Let } Y = e^{\beta_0 + \beta_1 \cdot x}$$

$$\text{Then } \frac{P(x)}{1-P(x)} = Y$$

$$P(x) = Y(1 - P(x))$$

$$P(x) = Y - Y(P(x))$$

$$P(x) + Y(P(x)) = Y$$

$$P(x)(1 + Y) = Y$$

$$P(x) = \frac{Y}{1 + Y}$$

$$P(x) = \frac{e^{\beta_0 + \beta_1 x}}{1 + e^{\beta_0 + \beta_1 x}}$$

We will simplify the Equation by dividing the Equation with numerator

$$P(x) = \frac{e^{\beta_0 + \beta_1 x}}{e^{\beta_0 + \beta_1 x}}$$

$$\frac{1}{e^{\beta_0 + \beta_1 x}} + \frac{e^{\beta_0 + \beta_1 x}}{e^{\beta_0 + \beta_1 x}} = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

$$\therefore P(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

Example: The student dataset has Entrance mark based on the historic data of those who are selected or not selected

→ Based on the logistic regression, the values of the learnt parameters are $\beta_0 = 1$ and $\beta_1 = 8$

→ Assuming Marks of $x = 60$, compute the resultant Class. $\beta_0 + \beta_1 x = 481$

$$P(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}} = \frac{1}{1 + e^{-481}} = 0.44$$

If we assume the threshold value as 0.5, then it is observed that $0.44 < 0.5$ therefore the candidate with max G_0 is not selected.

* Linear regression :-

* Linear Regression

Assume that there is only one independent variable 'X'. if the relationship between 'X' and 'Y' (dependent variable output variable) is modeled by the relation.

$Y = a + bx$

↓ ↓

dependent independent
variable variable

→ Then the regression model is called "linear regression model".

steps to find 'a' and 'b' :-

1) First find the mean and Covariance

Means of 'x' and 'y' are given by

$$\bar{x} = \frac{1}{n} \sum x_i \quad \bar{y} = \frac{1}{n} \sum y_i$$

where n = total no. of data points.

① Variance of 'x' is given by

$$\text{var}(x) = \frac{1}{n-1} \sum (x_i - \bar{x}_i)^2$$

The Co-variance of 'x' and 'y' denoted by $\text{Cov}(x, y)$ is defined as:

$$\text{Cov}(x, y) = \frac{1}{n-1} \cdot \sum (x_i - \bar{x})(y_i - \bar{y})$$

Now the values of 'a' and 'b' can be computed using the following formulae :-

$$a = \bar{y} - b\bar{x}$$

$$b = \frac{\text{Cov}(x, y)}{\text{Var}(x)}$$

Example:

x	y
1	1
2	2
3	1.3
4	3.75
5	2.25

$$n = 5$$

$$\bar{x} = \frac{1}{5} (1 + 2 + 3 + 4 + 5) = 3$$

$$\bar{y} = \frac{1}{5} (1 + 2 + 1.3 + 3.75 + 2.25) = 2.06$$

$$\begin{aligned} \text{Cov}(x, y) &= \frac{1}{4} [(1-3)(1-2.06) + \dots \\ &\quad (5-3)(2.25-2.06)] \\ &= 1.0625 \end{aligned}$$

$$\text{Var}(x) = \frac{1}{4} [(1-3)^2 + (2-3)^2 + \dots + (5-3)^2]$$

$$= 2.5$$

$$b = \frac{\text{Cov}(x, y)}{\text{Var}(x)} = \frac{1.0625}{2.5} = 0.425$$

$$a = \bar{y} - b\bar{x} = 2.06 - 0.425 \times 3.0 = 0.785$$

Therefore, the linear regression model for the data is

$$Y = a + bx$$

$$Y = 0.785 + 0.425 \cdot x$$

* Multi-layer perception :- MLP

→ It is an artificial Neural Network (ANN) widely used for solving classification & regression tasks.

→ MLP consists of fully connected dense layers that transform input data from one dimension to another.

→ It is called "multi-layer" because it contains an input layer, one or more hidden layers and an output layer.

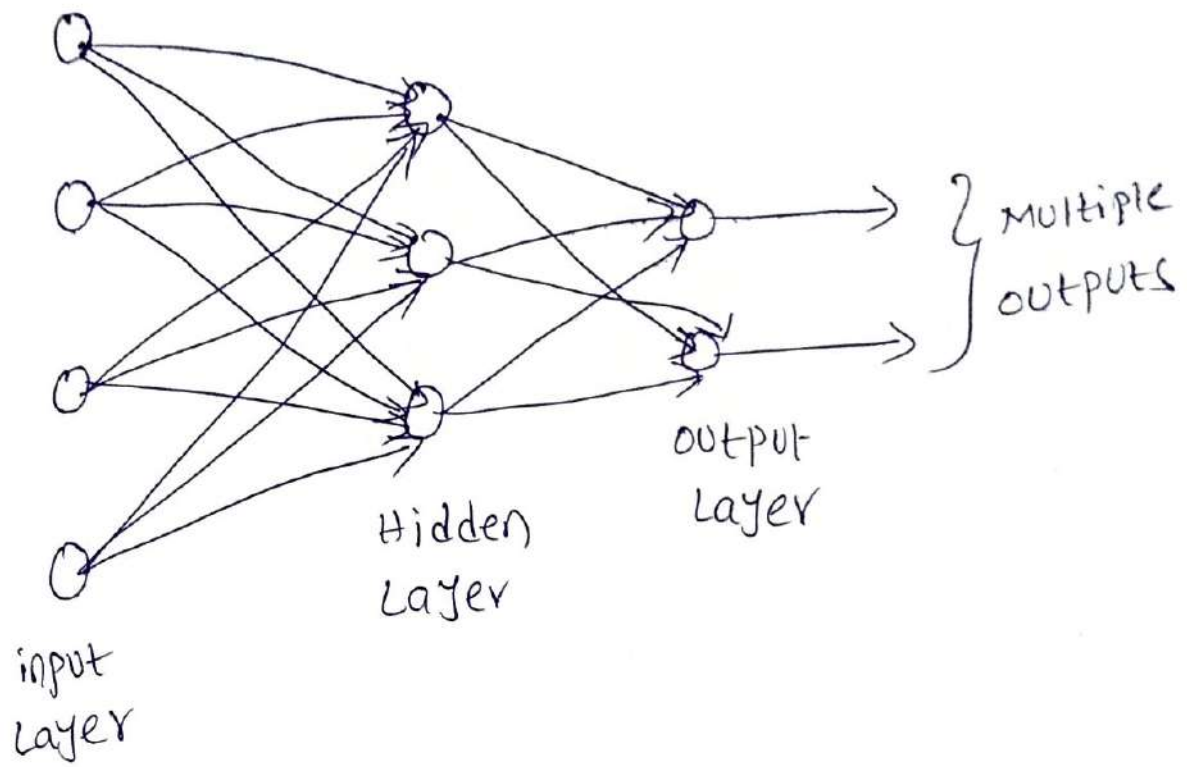
→ The purpose of an MLP is to model complex relationship b/w inputs and outputs making it a powerful tool for various machine learning tasks.

(a) Input Layer :- Each neuron (or node) in this layer corresponds to an input feature. For instance, if you have three i/p features, the i/p layer will have three neurons.

(b) Hidden Layers :- An MLP can have any no. of hidden layers, with each layer containing any number of nodes.

→ These layers process the information received from the i/p layer.

(c) Output Layer :- It generates the final prediction or result.



→ Every Connection in the diagram is a representation of the fully Connected nature of an MLP.

Radius Distance NN Algorithm:

$$B_R(T) = \{x_i \in X \text{ s.t. } \|T - x_i\| \leq r\}$$

$B_R(T)$ is Empty, o/p the majority Class of the Entire data set.

$B_R(T)$ is not Empty, o/p the majority Class of the data points within $B_R(T)$

Tree-Based NN Algorithm:

→ Transaction database

→ Collected from various sources such as business, Scientific Experiments.

→ Also "known as market Basket data"

→ differ size

→ "Association rule mining"

→ Frequent Patterns (FP)

Transaction Database

Digit

0

4

3

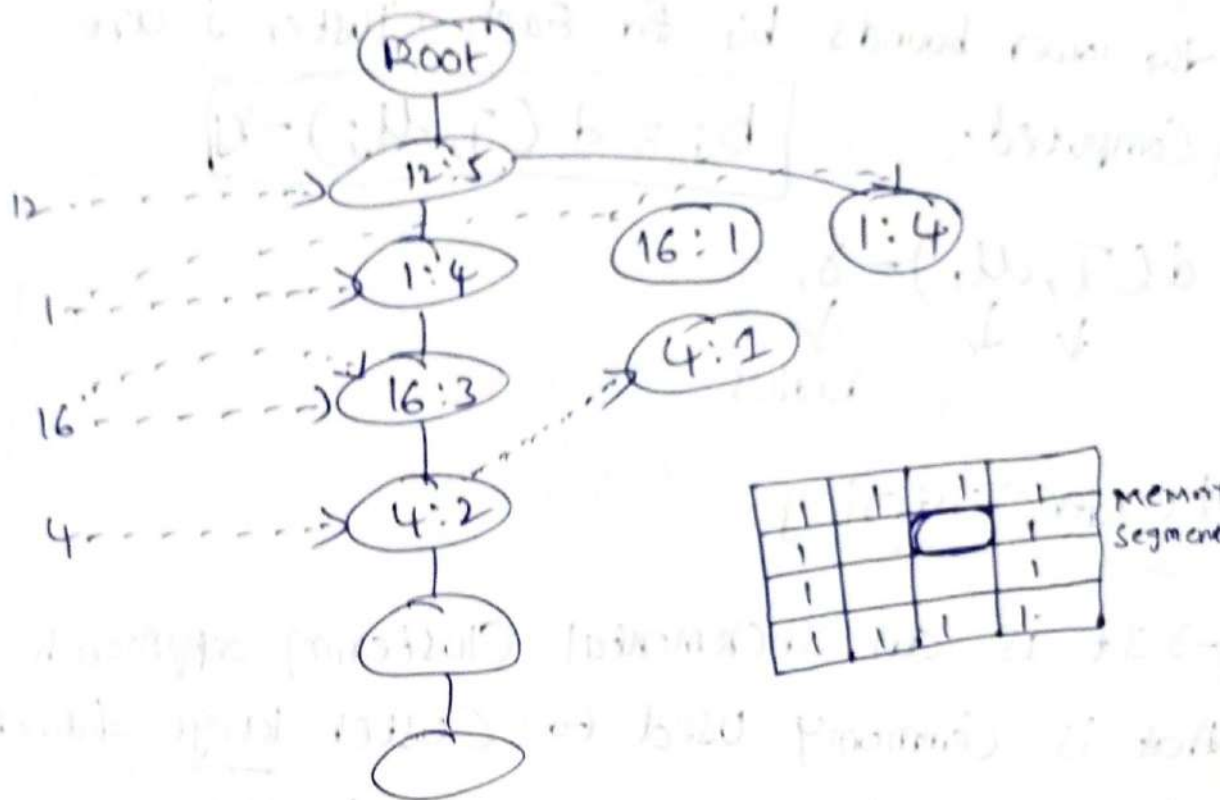
2

Transaction (positional information of a digit)

1, 2, 3, 12,

4, 8, 12, 16

→ ordered databases



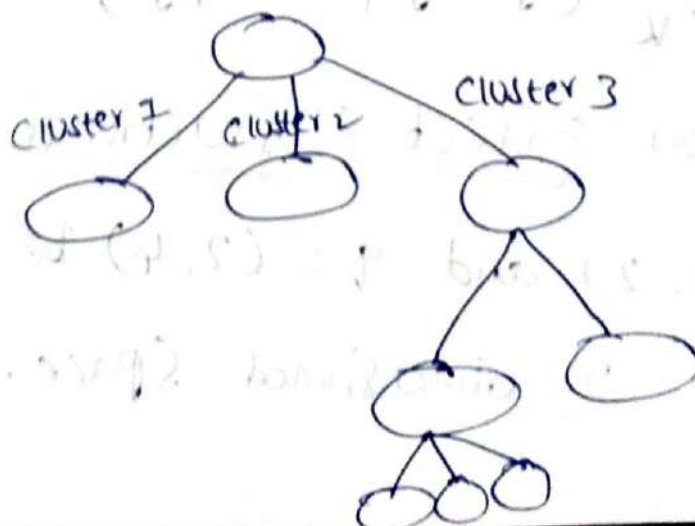
Tree based transaction database.

Branch & Bound Method:

→ ordered d.s. such as a tree-like structure

→ By clustering the data into representative groups.

→ data clustering using hierarchically into subsets until there are clusters.



Ex: To find NN of a new point $T = (21, 0.7)$
the lower bounds b_j for each cluster j are
computed.

$$b_j = d(T, \mu_j) - r_j$$

$$d(T, \mu_j) - r_j =$$

$\downarrow \quad \downarrow \quad \downarrow$
radius

Leader Clustering:

- It is an Incremental Clustering approach that is commonly used to cluster large datasets that cannot be accommodated in the main memory of the machine processing the data.
- data stored in secondary memory.

KNN Regression:

$$X = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$$

n is a data vector

$$\hat{y} = \frac{1}{n} (y^1 + y^2 + \dots + y^n)$$

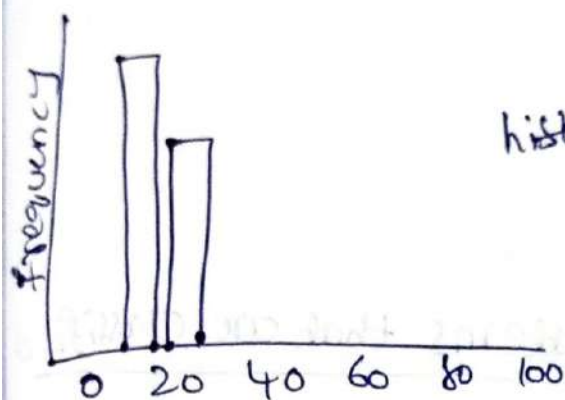
Concentration Effect & Fractional Norms:

Let $p = (4, 2)$ and $q = (2, 4)$ be two points in 2-dimensional space.

L_∞ norm (or) max norm = $\max(|14-2|, |12-4|) = 2$

L_2 norm or Euclidean = $2\sqrt{2}$

L_1 norm or City Block distance : $|14-2| + |12-4| = 4$



histogram ^{"data driven decisions"}

histograms (visual representation of distribution of Quantitative data)

Dis b/w a pair L_4 norm

Ex: $P = (4, 2)$ and $Q = (2, 4)$ The fractional norms give the distance as

$L_{0.5}$ norm

$L_{0.25}$ norm

$L_{0.1}$ norm

* KNN Regression :

Given a set X of ' n ' labelled dataset, where

$$X = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$$

Here $x_i, i=1, 2, \dots, n$ is a data vector

$y_i, i=1, 2, \dots, n$ is a scalar

→ To find the value of ' y ' for a new vector ' x '.

Steps:

1) Find the KNN of ' x ' from n data vectors, let them be

$$x^1, x^2, \dots, x^k$$

2) Consider the y values associated with these x^i 's. Let them be y^1, y^2, \dots, y^k

3) Take the average of these y^i 's and declare this average value to be the predicted value of ' y ' associated with ' x '.

So the predicted value of ' y ' call it \hat{y} is

$$\hat{y} = \frac{1}{k} (y_1 + y_2 + y_3 + \dots + y_k)$$

Example data for KNN regression:

Number (i)	Pattern (x_i)	Target (y_i)
1	(0.2, 0.4)	8
2	(0.4, 0.2)	8
3	(0.6, 0.4)	12
4	(0.8, 0.6)	16

Let the new pattern be $x = (0.3, 0.4)$

→ Let us see how to predict the "target value" for x using KNN regression.

→ Let $k=3$; 3 NNS of x from the patterns in the table are $(0.2, 0.4)$, $(0.4, 0.2)$, $(0.5, 0.5)$

→ Corresponding target values is 6, 8, 12

$$\frac{6+8+12}{3} = 9.33$$

The predicted target value for $x = 9.33$

Real world dataset

Boston Housing dataset or Wisconsin Breast Cancer dataset.

→ We consider 250 pattern vector

200 for Training Pattern
50 for Test pattern

Concentration effect & fractional norms

The distance values b/w various pairs of points, may not show much dynamic range

Ex: Let $p = (4, 2)$ and $q = (2, 4)$ be two points in a 2-dimensional space. Values of the distance using some popular distance norms are

$$1) \text{ } L_{\infty} \text{ norm (or) Max norm} = \max(|4-2|, |2-4|) = 2$$

2) L_2 norm (∞) Euclidean norm $\vdash 2\sqrt{2}$

Calculation

$$P(4,2) \quad Q(2,4) \vdash \sqrt{(4-2)^2 + (2-4)^2}$$

$$= \sqrt{(2)^2 + (-2)^2}$$

$$= \sqrt{4+4} = \sqrt{8}$$

$$= \sqrt{4 \times 2} = 2\sqrt{2}$$

Worst performance norm

3) L_1 norm (∞) City-Block distance $\vdash |4-2| + |2-4| = 4$

Note \vdash Observe that as the ' r ' value in the Minkowski norm keeps decreasing, the distance b/w the pair

(P, Q) keeps increasing.

$$d^r(P, Q) = \sum_{k=1}^d (|P_k - Q_k|^r)^{1/r}$$

$r \downarrow$ distance b/w $P, Q \uparrow$

$\rightarrow r$ is a fractional norm \uparrow the dynamic range and

$P, Q (\downarrow)$ Concentration effects

$$4) L_{0.5} \text{ norm} = \left(\sqrt{|4-2|} + \sqrt{|2-4|} \right)^{1/0.5}$$

$$= 8$$

$$5) L_{0.25} \text{ norm} = 32$$

$$6) L_{0.1} \text{ norm} = 2048$$

Best \uparrow performance norm