UNIT 10 CLASSIFICATION

Structure

10.1 INTRODUCTION

What exactly does learning entail, anyway? What exactly is meant by "machine learning"? These are philosophical problems, but we won't be focusing too much on philosophy in this lesson; the whole focus will be on gaining a solid understanding of how things work in practise. In the subject of data mining, many of the ideas, such as classification and clustering, are being addressed, and so here in this Unit, we are going to once again investigate those concepts. Therefore, in order to achieve a better knowledge, the first step is to differentiate between the two fields of study known as data mining and machine learning.

It's possible that, at their core, data mining and machine learning are both about learning from data and improving one's decision-making. On the other hand, they approach things in a different manner. To get things started, let's start with the most important question, What exactly is the difference between Data Mining and Machine Learning?

What is data mining? Data mining is a subset of business analytics that involves exploring an existing huge dataset in order to discover previously unknown patterns, correlations, and anomalies that are present in the data. This process is referred to as "data exploration." It enables us to come up with wholly original ideas and perspectives.

What exactly is meant by "machine learning"? The field of artificial intelligence (AI) includes the subfield of machine learning . Machine learning involves computers performing analyses on large data sets, after which the computers "learn" patterns that will assist them in making predictions regarding additional data sets. It is not necessary for a person to interact with the computer for it to learn from the data; the initial programming and possibly some fine-tuning are all that are required.

It has come to our attention that there are a number of parallels between the two ideas, namely Data Mining and Machine Learning. These parallels include the following:

- Both are considered to be analytical processes;
- Both are effective at recognising patterns;
- Both focus on gaining knowledge from data in order to enhance decision-making capabilities;
- Both need a substantial quantity of information in order to be precise

Due to the mentioned similarities between the two, generally the people confuses the two concepts. So, to clearly demarcate the two concepts one should understand that What are the key differences between the two i.e. Data Mining and Machine Learning,?

The following are some of the most important distinctions between the two:

- Machine learning goes beyond what has happened in the past to make predictions about future events based on the pre-existing data. Data mining, on the other hand, consists of just looking for patterns that already exist in the data.

- At the beginning of the process of data mining, the 'rules' or patterns that will be used are unknown. In contrast, when it comes to machine learning, the computer is typically provided with some rules or variables to follow in order to comprehend the data and learn from it.

- The mining of data is a more manual process that is dependent on the involvement and choice-making of humans. With machine learning, on the other hand, once the foundational principles have been established, the process of information extraction, as well as "learning" and refining, is fully automated and does not require the participation of a human. To put it another way, the machine is able to improve its own level of intelligence.

- Finding patterns in an existing dataset (like a data warehouse) can be accomplished through the process of data mining. On the other hand, machine learning is trained on a data set referred to as a "training" data set, which teaches the computer how to make sense of data and then how to make predictions about fresh data sets.

The approaches to data mining problems are based on the type of information/ knowledge to be mined. We will emphasis on three different approaches: Classification, Clustering, and Association Rules.

The classification task puts data into groups or classes that have already been set up. The value of a user-specified goal attribute shows what type of thing a tuple is. Tuples are made up of one or more predicating attributes and one or more goal attributes. The task is to find some kind of relationship between the predicating attributes and the goal attribute, so that the information or knowledge found can be used to predict the class of new tuple (s).

The purpose of the clustering process is to create distinct classes from groups of tuples that share characteristic values. Clustering is the process of defining a mapping, using as input a database containing tuples and an integer value k, in such a way that the tuples are mapped to various clusters.

The idea entails increasing the degree of similarity within a class while decreasing the degree of similarity between classes. There is not an objective attribute in the clustering process. Clustering, on the other hand, is an example of an unsupervised classification, in contrast to classification, which is supervised by the aim attribute.

The goal of association rule mining is to find interesting connections between elements in a data set. Its initial use was for "market basket data." The rule is written as X➔Y, where X and Y are two sets of objects that do not intersect. Support and confidence are the two metrics for any rule. The aim is to identify, using rules with support and confidence above, minimum support and minimum confidence given the user-specified minimum support and minimum confidence.

The distance measure determines the distance between items or their dissimilarity. The following are the measures used in this unit:

$$\text{Euclidean distance: } dis(ti,tj) = \sqrt{\sum_{h=1}^{k} (t_{ih} - t_{jh})^2}$$

$$\text{Manhattan distance: } dis(ti,tj) = \sum_{h=1}^{k} |(t_{ih} - t_{jh})|$$

where ti and tj are tuples and h are the different attributes which can take values from 1 to k

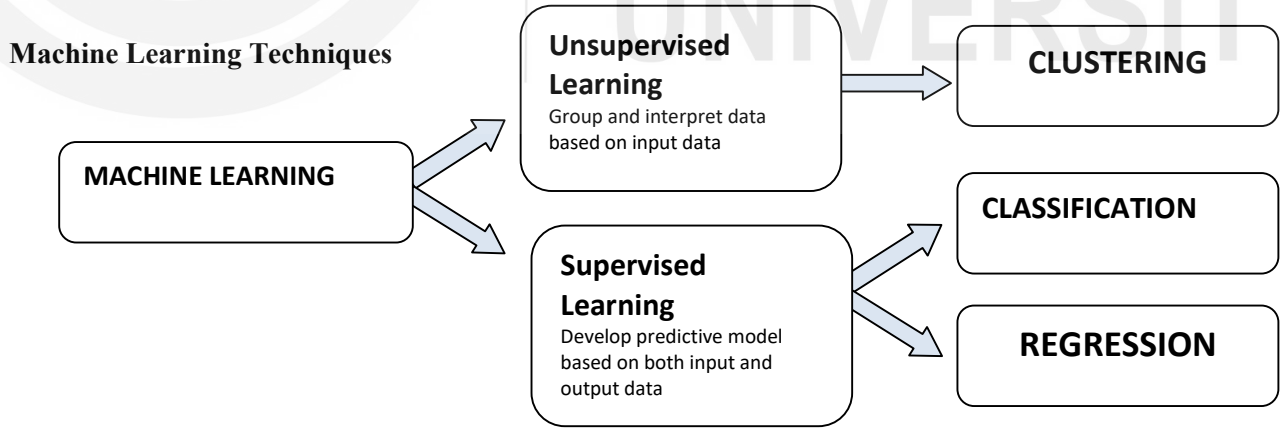
There are some clear differences between the two, though. But as businesses try to get better at predicting the future, machine learning and data mining may merge more in the future. For example, more businesses may want to use machine learning algorithms to improve their data mining analytics.

Machine learning algorithms use computational methods to "learn" information directly from data, without using an equation as a model. As more examples are available for learning, the algorithms get better and better at what they do.

Machine learning algorithms look for patterns in data that occur naturally. This gives you more information and helps you make better decisions and forecasts. They are used every day to make important decisions in diagnosing medical conditions, trading stocks, predicting energy load, and more. Machine learning is used by media sites to sort through millions of options and suggest songs or movies. It helps retailers figure out what their customers buy and how they buy it. With the rise of "big data," machine learning has become very important for solving problems in areas like:

• Computational finance, for applications such as credit scoring and algorithmic trading
• Face identification, motion detection, and object detection can all be accomplished through image processing and computer vision.
• Tumor detection, drug development, and DNA sequencing can all be accomplished through computational biology.
• Production of energy, for the sake of pricing and load forecasting • Automotive, aerospace, and manufacturing, for the purpose of predictive maintenance • Processing of natural languages

In general, Classical Machine Learning Algorithms can be put into two groups: Supervised Learning Algorithms, which use data that has been labelled, and Un-Supervised Learning Algorithms, which use data that has not been labelled and are used for Clustering. We will talk more about Clustering in Unit 15, which is part of Block 4 of this course.



In this unit we will be discussing about the Supervised Learning Algorithms, which are mainly used for the classification purpose.

10.2 OBJECTIVES

After completing this unit you should be able to :

- Understand Supervised Learning
- Understand Un-Supervised Learning
- Understanding various Classification Algorithms

10.3 UNDERSTANDING OF SUPERVISED LEARNING

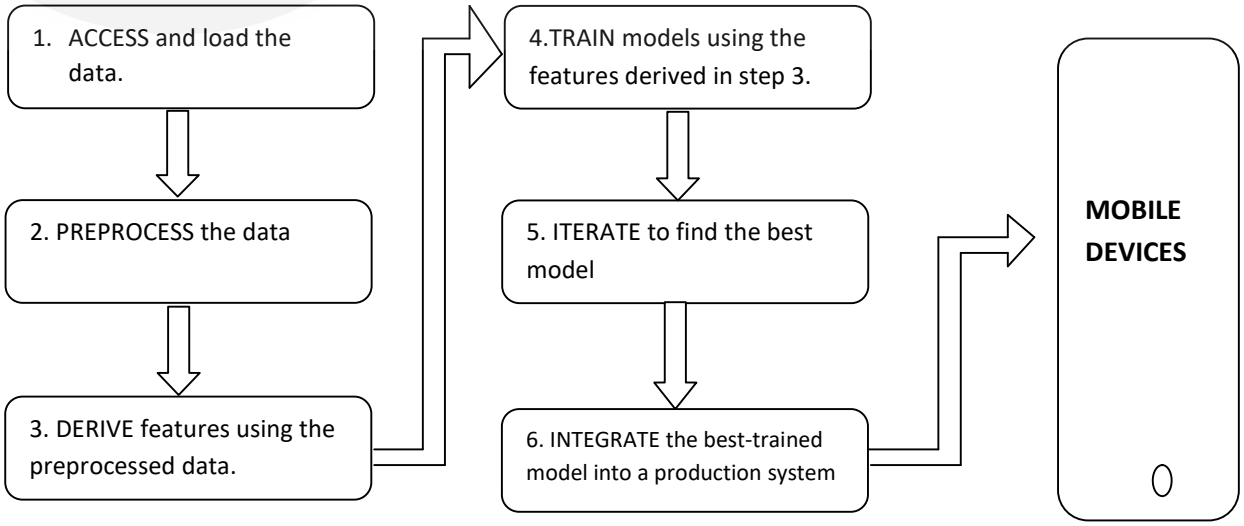
To use machine learning techniques effectively, you need to know how they work. You can't just use them without knowing how they work and expect to get good results. Different techniques work for different kinds of problems, but it's not always clear which techniques will work in a given situation. You need to know something about the different kinds of solutions.

Every workflow for machine learning starts with the following three questions:

- What kind of data do you have available to work with?
- What kinds of realisations are you hoping to arrive at as a result of it?
- In what ways and contexts will those realisations be utilised?

Your responses to these questions will assist you in determining whether supervised or unsupervised learning is best for you.

Workflow at a Glance



In an interesting way, supervised machine learning is like how humans and animals learn "concepts" or "categories." This is defined as "the search for and listing of attributes that can be used to tell exemplars from non-exemplars of different categories."

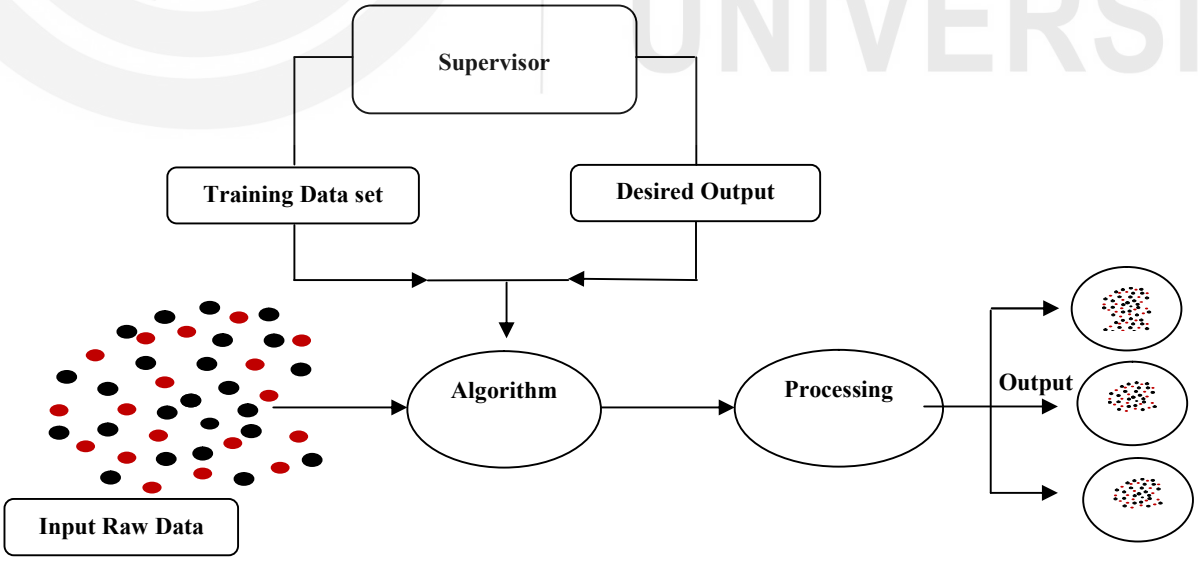
Technically, supervised learning means learning a function that gives an output for a given input based on a set of input-output pairs that have already been defined. It does this with the help of something called "training data," which is a set of examples for training.

In supervised learning, the data used for training is labelled. For example, every shoe is labelled as a shoe, and the same goes for every pair of socks. This way, the system knows the labels, and if it sees a new type of shoes, it will recognise them as "shoes" even if it wasn't told to do so.

In the example above, the picture of shoes and the word "shoes" are both inputs, and the word "shoes" is the output. After learning from hundreds or thousands of different pictures of shoes and the word "shoes" as well as the word "socks," our system will know what to do when given only a new picture of shoes (name: shoes).

supervised ML is often represented by the function y = f(x), where x is the input data and y is the output variable, which is a function of x that needs to be predicted. In training data, the example pair is usually made up of an input, which is usually a vector, and an output (a collection of features determining a sample). The output value we want, which we call a "supervisory signal" and whose meaning is clear from the name.

In fact, the goal of supervised machine learning is to build a model that can make predictions based on evidence even when there is uncertainty. A supervised learning algorithm uses a known set of input data and known responses to the data (output) to train a model to make reasonable predictions about the response to new data.



Example: Predicting heart attacks with the help of supervised learning: Let's say doctors want to know if someone will have a heart attack in the next year. They have information about the age,

weight, height, and blood pressure of past patients. They know if the patients who were there before had heart attacks within a year. So the problem is making a model out of the existing data that can tell if a new person will have a heart attack in the next year.

Following Steps are Involved in Supervised Learning, and they are self explanatory:

1. Determine the Type of Training Examples
2. Prepare/Gather the Training Data
3. Determine Relation Between Input Feature & Representing Learned Function
4. Select a Learning Algorithm
5. Run the Selected Algorithm on Training Data
6. Evaluate the Accuracy of the Learned Function Using Values from Test Set

There are some common issues which are generally faced when one applies the Supervised Learning, and they are listed below:

(i) Training and classifying require a lot of computer time, especially when big data is involved.

(ii) Overfitting: The model may learn so much from the noise in the data that instead of seeing it as a mistake, it can be seen as a learning concept.

(iii) A key difference between supervised and unsupervised learning is that if an input doesn't fit into any class, the model will add it to one of the existing ones instead of making a new one.

Lets discuss some of the Practical Applications of Supervised Machine Learning. For beginners at least, probably knowing ‘what does supervised learning achieve’ becomes equally or more important than simply knowing ‘what is supervised learning’.

A very large number of practical applications of the method can be outlined, but the following are some of the common ones

a) Detection of spam
b) Detection of fraudulent banking or other activities
c) Medical Diagnosis
d) Image recognition
e) Predictive maintenance

With increasing applications each day in all the fields, machine learning knowledge is an essential skill.

☞ Check Your Progress 1

1. Compare between Supervised and Un-Supervised Learning.

   ……

   ……

2. List the Steps Involved in Supervised Learning

   ……

   ……

3. What are the Common Issues Faced While Using Supervised Learning

   ……

   ……

10.4 INTRODUCTION TO CLASSIFICATION

Every supervised learning approach can be classified as either a regression or a classification method, depending on the nature of the data being analysed. The creation of predictive models is possible through supervised learning by utilising classification and regression methods. This can be performed by using these methods.

• **Classification techniques** : Classification methods make predictions about specific outcomes, such as whether an email is legitimate or spam or whether a tumour is malignant or benign. Classification models classify incoming data into categories. Applications such as medical imaging, speech recognition, and credit scoring are typical examples.

• **Regression techniques** : The techniques of regression can accurately forecast continuous reactions, such as shifts in temperature or variations in the amount of power required. Examples of typical applications are as follows: A few examples of applications are the predicting of stock prices, the recognition of handwriting, the forecasting of power load, and acoustic signal processing.

Note: *It's important to know whether a problem is a classification problem or a regression problem.*

> *• Can your information be tagged or put into groups? Use classification algorithms if your data can be put into clear groups or classes.*
>
> *• Are you working with a set of data? Use regression techniques if your answer is a real number, like TEMP. or the time until a piece of equipment fails to work.*

Before moving ahead lets understand some of the key terms, which will be frequently occurring in this course, they are listed below :

- **Classification** The process of organizing data into a predetermined number of categories is referred to as classification. Finding out which category or class a new collection of data falls under is the primary objective of a classification problem, which can be stated as follows, Data that is structured as well as data that is not structured can be utilized for classification:

  Structured data (data that is in a fixed field in a file or record is called "structured data"). A relational database (RDBMS) is where most structured data is kept.

  Unstructured data (unstructured data may have a natural structure, but it isn't set up in a way that can be predicted). There is no data model, and the data is stored in the format in which it was created. Rich media, text, social media activity, surveillance images, and so on, are all types of unstructured data.Following are some of the terminologies frequently encountered in machine learning – classification:

- **Classifier:** A classifier is an algorithm that puts the data you give it into a certain category. A classifier is an algorithm that does classification on a dataset.
- **Classification model:** A classification model looks at the output values to try to figure out what the values used for training mean. It will guess the class labels or categories of the new data.
- **Feature:** property of any object (real or virtual) that can be measured on its own is called a feature.
- **Classification predictive modeling** involves assigning a class label to input examples.

This section covers the following types of classification:

- **Binary classification** is a task for which there are only two possible outcomes. It means predicting which of the two classes will be correct. For example, dividing people into male and female.

Some popular algorithms that can be used to divide things into two groups are:

- Logistic Regression.
- k-Nearest Neighbors.
- Decision Trees.
- Support Vector Machine.
- Naive Bayes.

Major application areas of Binary Classification:

- Detection of Email spam.
- Prediction of Churn.
- Prediction of Purchase or Conversion(buy or not).

- **Multi-class classification.:**Multi-class classification means putting things into more than two groups. In multiclass classification, there is only one target label for each sample. This is done by predicting which of more than two classes the sample belongs to. An

animal, for instance, can either be a cat or a dog, but not both. Face classification, plant species classification, and optical character recognition are some of the examples.
Popular algorithms that can be used for multi-class classification include:

- k-Nearest Neighbors.
- Decision Trees.
- Naive Bayes.
- Random Forest.
- Gradient Boosting.

Binary classification algorithms can be changed to work for problems with more than two classes. This is done by fitting multiple binary classification models for each class vs. all other classes (called "one-vs-rest") or one model for each pair of classes (called one-vs-one).

- **One versus the Rest**: Fit one binary classification model for each class versus all of the other classes in the dataset.
- **One-versus-one**: Fit one binary classification model for each pair of classes using the one-on-one comparison method.

Binary classification techniques such as logistic regression and support vector machine are two examples of those that are capable of using these strategies for multi-class classification.

- **Multi-label classification:** Multi-label classification, also known as more than one class classification, is a classification task in which each sample is mapped to a collection of target labels. This classification task involves making predictions about one or more classes; say for example, a news story can be about Games, People, and a Location all together at the same time.

  Note : Classification algorithms used for binary or multi-class classification cannot be used directly for multi-label classification.

Specialized versions of standard classification algorithms can be used, so-called multi-label versions of the algorithms, including:

- Multi-label Decision Trees
- Multi-label Random Forests
- Multi-label Gradient Boosting

Another approach is to use a separate classification algorithm to predict the labels for each class.

• **An imbalanced classification** is a task in which the number of examples in each class is not the same. Examples includes Fraud detection, Outlier detection, Medical diagnostic tests.

These problems are modelled as two-way classification tasks, but you may need to use specialised methods to solve them.

The different types of classifications discussed above, have to deal with different type of learners, and **Learners in Classification Problems are categorized into following two types :**

1. **Lazy Learners:** Lazy Learner will first store the training dataset, and then it will wait until it is given the test dataset. In the case of the Lazy learner, classification is done based on the information in the training dataset that is most relevant to the question at hand. Less time is needed for training, but more time is needed for making predictions. K-NN algorithm and Case-based reasoning are two examples.
2. **Eager Learners:** Eager Learners use a training dataset to make a classification model before they get a test dataset. Eager learners, on the other hand, spend less time on training and more time on making predictions. Decision Trees, Naive Bayes, and ANN are some examples.

Examples of eager learners include the classification techniques of Bayesian classification, decision tree induction, rule-based classification, classification by back-propagation, support vector machines, and classification based on association rule mining. Eager learners, when presented with a collection of training pairs, will construct a generalisation model (also known as a classification model) before being presented with new tuples, also known as test tuples, to classify. One way to think about the learnt model is as one that is prepared and eager to categorise tuples that have not been seen before.

Imagine, on the other hand, in the lazy learner approach the learner is required to wait until the final moment, to develop a model for classification of the given test tuple, i.e. in the lazy approach the learner just stores the training tuple, given for classification. It does not do generalization until it is given a test tuple, after receiving the test tuple, it classifies the tuple based on how similar it is to the training tuples that it has previously stored. Lazy learning methods, on the other hand, do less work when a training pair is shown but more work when classifying or making a prediction. Because lazy learners keep the training tuples, which are also called "instances." So, they are also called "instance-based learners," even though this is how most people learn.

When classifying or making a prediction, lazy learners can take a lot of processing power. They need efficient ways to store information and can be done well on parallel hardware. They don't explain or show much about how the data is put together. Lazy learners, on the other hand, tend to be in favour of incremental learning. They can make models of complex decision spaces with hyper-polygonal shapes that other learning algorithms may not be able to do as well (such as hyper-rectangular shapes modeled by decision trees). The k-nearest neighbour classifier and the case-based reasoning classifier are both types of lazy learners.

☞ **Check Your Progress 2**

4. Compare between Multi Class and Multi Label Classification

……

……

5. Compare between structured and unstructured data

……

……

6. Compare between Lazy learners and Eager Learners algorithms for machine learning.

……

……

10.5 CLASSIFICATION ALGORITHMS

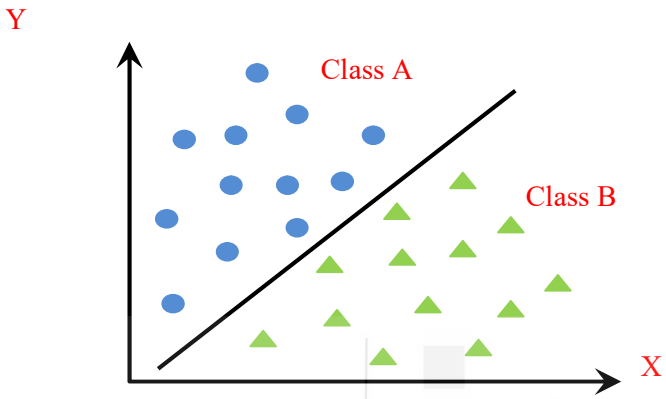
The Classification algorithm is a type of Supervised Learning that uses the training data to figure out the category of new observations. This method is used to figure out what kind of thing a new observation is. Classification is the process by which a computer programme learns from a set of data or observations and then sorts new observations into different classes or groups. "Yes" or "No," "0" or "1," "Spam" or "Not Spam," "Cat or Dog," and so on are all good examples. Classes are the same thing that have different names, like categories, objectives, and labels.

In classification, the output variable is not a value but a category, such as "Green or Blue," "Fruit or Animal," etc. This is different from regression, where the output variable is a value. Since the classification method is a supervised learning method, it needs data that has been labelled in order to work. This means that the implementation of the algorithm includes both the input and the output that go with it.

As the name suggests, classification algorithms do the job of predicting a label or putting a variable into a category (categorization). For example, classifying something as "socks" or "shoes" from our last example. Classification Predictive Algorithm is used every day in the spam detector in emails. It looks for features that help it decide if an email is spam or not spam.

The primary objective of the Classification algorithm is to determine the category of the dataset that is being provided, and these algorithms are primarily utilised to forecast the output for the data that is categorical in nature. A discrete output function, denoted by y, is mapped to an input variable, denoted by x, in a classification process. Therefore, y = function (x), where y denotes the categorical output. The best example of an ML classification algorithm is **Email Spam Detector**.

The diagram that follows can be used to have a better understanding of classification methods. There are two classes, Class A and Class B, depicted in the graphic that may be found below. These classes share characteristics that distinguish them from other classes but also distinguish them from one another.



The question arises as a result of the existence of a variety of algorithms under both supervised and unsupervised learning. How Should One Choose Which Algorithm to Employ? The task of selecting the appropriate machine learning algorithm can appear to be insurmountable because there are dozens of supervised and unsupervised machine learning algorithms, and each takes a unique approach to the learning process. There is no single approach that is superior or universally applicable. Finding the appropriate algorithm requires some amount of trial and error; even highly experienced data scientists are unable to determine whether or not an algorithm will work without first putting it to the test themselves. However, the choice of algorithm also depends on the quantity and nature of the data being worked with, as well as the insights that are desired from the data and the applications to which those insights will be put.

- **Choose supervised learning** if you need to train a model to make a prediction, for instance, the future value of a continuous variable, such as TEMP. or a stock price; use regression techniques and use classification techniques in situations such as identifying makes of cars from webcam video footage or identifying spams from emails; etc.

- **Choose unsupervised learning** if you need to investigate your data and want to train a model to find a decent internal representation, such as by dividing the data into clusters. This type of learning allows for more freedom in exploring and representing the data.

Note : The algorithm for Supervised Machine Learning can be broken down into two basic categories: regression algorithms and classification algorithms. We have been able to forecast the output for continuous values using the Regression methods; but, in order to predict the categorical values, we will need to use the Classification algorithms.

Let's take a closer look at the most commonly used algorithms for supervised machine learning.

Classification algorithms can be further divided into the mainly two categories, *Linear Models and Non Linear Models*, which includes various algorithms under them, the same are listed below :

- **Linear Models** : Involves Logistic Regression, Support Vector Machines
- **Non-linear Models :** Involves K-Nearest Neighbours, Kernel SVM, Naïve Bayes, Decision Tree Classification, Random Forest Classification

In order to build a classification model following steps are to be followed :

1. Start the classifier that will be utilised from scratch.
2. Train the classifier: In order to fit the model (training), all classifiers in scikit-learn use a function called fit(X, y) to do so. This method takes as input the train data X and the train label y.
3. Predict the target: Given an observation X that is not labelled, the predict(X) function returns the label that was predicted for the observation.
4. Conduct an analysis of the classification model.

EVALUATING A CLASSIFICATION MODEL: Now that we have a classification model, let's learn how to evaluate it. After we have finished developing our model, we need to assess how well it works, regardless of whether it is a regression or classification model. The following are some of the methods that can be used to evaluate a classification model:

1. Log Loss or Cross-Entropy Loss:

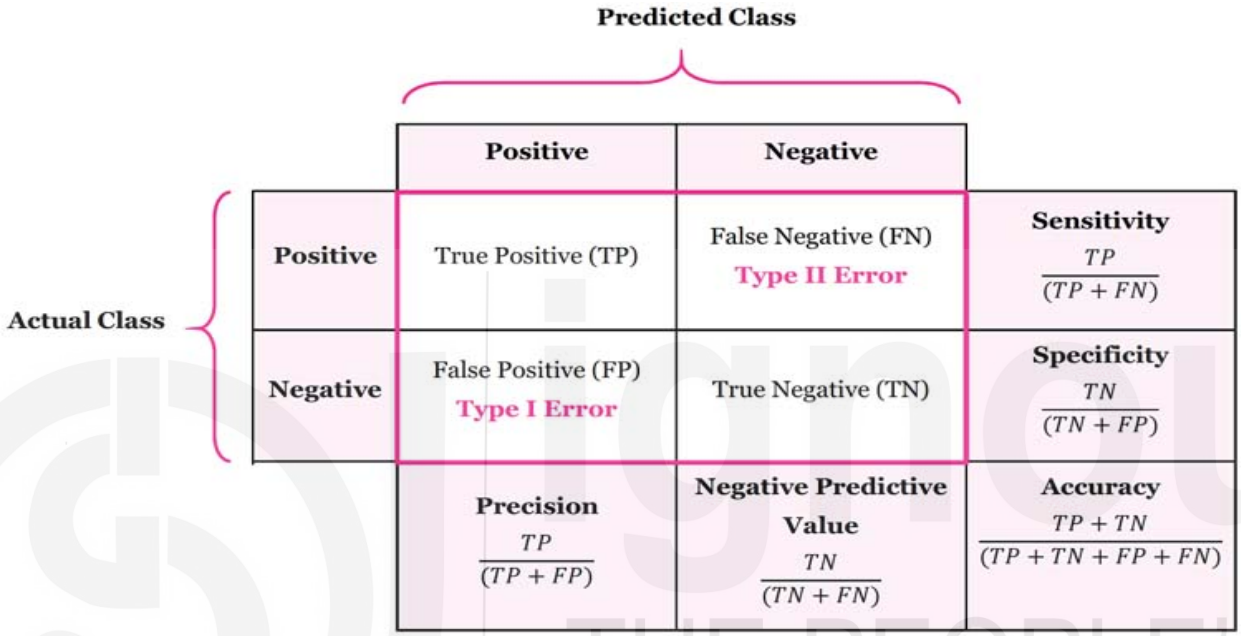
- It's used to measure how well a classifier works, which gives a probability value between 0 and 1.
- The value of log loss should be close to 0 for a good binary classification model.
- If the predicted value is different from the actual value, the value of log loss goes up.
- The model is more accurate when the log loss is lower.
- For binary classification, the cross-entropy is calculated by taking the actual output (y) and the expected output (p). The formula for cross-entropy is shown below.
  $-(y\log(p) + (1-y)\log(1-p))$

2. Confusion Matrix:

- The confusion matrix tells us how well the model works and gives us a matrix or table as Output.

- Sometimes, this kind of structure is called the error matrix.
- The matrix is a summary of the results of the predictions. It shows how many predictions were right and how many were wrong.

The matrix looks like as below table:



3. AUC-ROC curve:

- The letters AUC and ROC stand for "area under the curve" and "receiver operating characteristics curve," respectively.
- This graph shows how well the classification model works at several different thresholds.
- We use the AUC-ROC Curve to see how well the multi-class classification model is doing.
- The (TPR)TruePositiveRate and the (FPR)FalsePositiveRate are used to plot the ROC curve, with TPR on the Y-axis and FPR on the X-axis.

Classification algorithms have several applications, Following are some popular applications or use cases of Classification Algorithms:

- Detecting Email Spam
- Recognizing Speech
- Detection of Cancer tumor cells.
- Classifying Drugs
- Biometric Identification, etc.

□ Check Your Progress 3

4. List the classification algorithms under the categories of Linear and Non-Linear Models. Also Discuss the various methods used for evaluating a classification model

……

……

10.5.1 NAÏVE BAYES

This is an example of a statistical classification, which estimates the likelihood that a particular sample belongs to a particular group given the sample in question. The Bayes theorem provides the foundation for it. When used to big databases, the Bayesian classification demonstrates both improved accuracy and increased speed. In this section, we will talk about the most basic kind of Bayesian categorization.

"The effect of a given attribute value on a certain class is unaffected by the values of other attributes, i.e. both are independent," is one of the fundamental underlying assumptions that underpin the native Bayesian classification, which is the simplest form of Bayesian classification. Class conditional independence is another name for this basic assumption.

Let's go into greater depth about the naïve Bayesian classification, shall we? But before we get into it, let's take a moment to define the fundamental theorem that underpins this classification i.e. the Bayes Theorem.

Bayes Theorem: In order to understand this theorem firstly lets understand the meaning of the following symbols or assumptions, they are as follows :

- X is an example of a data set whose class needs to be determined.
- H refers to the hypothesis which states that the data sample X falls into the class C.
- P(H | X): The probability that the data sample X belongs to the class C is expressed by the formula P(H | X), where H is the hypothesis and X is the data sample. This formula represents the likelihood that the data sample X belongs to the class C. The likelihood that the condition H is true for the sample X is often referred to as the posterior probability.
- The prior probability of the H condition is denoted by the notation P(H), which is based on the training data.
- The posterior probability of the X sample is denoted by the symbol P(X | H), which assumes that H is correct.
- The prior probability on the sample X is denoted by the letter P(X).

Note: From the data sample X and the data used for training, we may get the parameters P(X), P(X | H), and P(H). Whereas, P(H | X) is the only variable that, by itself, may define the likelihood that X belongs to a class C; this probability, however, cannot be calculated. This purpose is served by Bayes' theorem in particular.

The Bayer's theorem states:

$$P(H \mid X) = \frac{P(X \mid H)\, P(H)}{P(X)}$$

Now after defining the Bayes theorem, let us explain the Bayesian classification with the help of an example.

i) Consider the sample having an n-dimensional feature vector. For our example, it is a 3-dimensional (Department, Age, Salary) vector with training data as given in the Figure 3.

ii) Assume that there are m classes C_1 to C_m. And an unknown sample X. The problem is to data mine which class X belongs to. As per Bayesian classification, the sample is assigned to the class, if the following holds:

$$P(C_i|X) > P(C_j|X) \text{ where j is from 1 to m but } j \neq i$$

In other words the class for the data sample X will be the class, which has the maximum probability for the unknown sample. **Please note:** The $P(C_i|X)$ will be found using:

$$P(C_i|X) = \frac{P(X|C_i)\,P(C_i)}{P(X)} \qquad (3)$$

In our example, we are trying to classify the following data:
X = (Department = "Personal", Age = "31 – 40" and Salary = "Medium_Range)

into two classes (based on position) C_1=_BOSS_ OR C_2=_ASSISTANT_.

iii) The value of P(X) is constant for all the classes, therefore, only $P(X|C_i)\,P(C_i)$ needs to be found to be maximum. Also, if the classes are equally, then,
$P(C_1)=P(C_2)=\ldots..P(C_n)$, then we only need to maximise $P(X|C_i)$.

How is $P(C_i)$ calculated?

$$P(C_i) = \frac{\text{Number of training samples for Class } C_i}{\text{Total Number of Training Samples}}$$

In our example,

$$P(C_1) = \frac{5}{11}$$

and

$$P(C_2) = \frac{6}{11}$$

So $P(C_1) \neq P(C_2)$

iv) $P(X|C_i)$ calculation may be computationally expensive if, there are large numbers of attributes. To simplify the evaluation, in the naïve Bayesian classification, we use the condition of class conditional independence, that is the values of attributes are independent of each other. In such a situation:

$$P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i) \qquad \ldots.(4)$$

where x_k represent the single dimension or attribute.

The $P(x_k|C_i)$ can be calculated using mathematical function if it is continuous, otherwise, if it is categorical then, this probability can be calculated as:

$P(x_k|C_i)$= Number of training samples of class C_i having the value x_k for the attribute A_k
Number of training samples belonging to C_i

For our example, we have x_1 as Department= "_PERSONNEL_"

x_2 as Age="31 – 40" and

x_3 as Salary= "Medium_Range"

P (Department= "_PERSONNEL_" | Position = "_BOSS_") = 1/5

P (Department= "_PERSONNEL_" | Position = "_ASSISTANT_") = 2/6

P (Age="31 – 40" | Position = "_BOSS_") = 3/5

P (Age="31 – 40" | Position = "_ASSISTANT_") = 2/6

P (Salary= "Medium_Range"| Position = "_BOSS_") = 3/5

P (Salary= "Medium_Range" | Position = "_ASSISTANT_") = 3/6

Using the equation (4) we obtain:

P (X | Position = "_BOSS_") = 1/5 * 3/5 * 3/5

P (X | Position = "_ASSISTANT_") = 2/6 * 2/6 * 3/6

Thus, the probabilities:

P (X | Position = "_BOSS_") P(Position = "_BOSS_")

= (1/5 * 3/5 * 3/5) * 5/11

= 0.032727

P (X | Position = "_ASSISTANT_") P(Position = "_ASSISTANT_")

= (2/6 * 2/6 * 3/6) * 6/11

= 0.030303

Since, the first probability of the above two is higher, the sample data may be classified into the _BOSS_ position. Kindly check to see that you obtain the same result from the decision tree .

Naiive Bayes : Steps to perform naiive bayes algorithm

Step 1: Handling Data : Data is loaded from the CSV File and spread into training and tested assets.

Step 2: Summarizing the Data : Summarise the properties in the training data set to calculate the probabilities and make predictions.

Step 3: Making a Prediction : A particular prediction is made using a summarise of the data set to make a single prediction.

Step 4: Making all the Predictions : Generate prediction given a test data set and a summarise data set.

Step 5: Evaluate Accuracy : Accuracy of the prediction model for the test data set as a percentage correct out of them all the predictions made.

Step 6: Tying all Together : Finally, we tie to all steps together and form our own model of Naive Bayes Classifier.

With the help of the following example, you can see how Naive Bayes' Classifier works:

Example: Let's say we have a list of _WEATHER_ conditions and a target variable called "Play" that goes with it. So, using this set of data, we need to decide whether or not to play on a given day based on the _WEATHER_.
If it's _SUNNY_, should the Player play?
So, here are the steps we need to take to solve this problem:

1. Make frequency tables out of the given dataset.

2. Make a Likelihood table by figuring out how likely each feature is.

Use Bayes's theorem to figure out the posterior probability.

To figure this out, first look at the dataset given below:

	OUTLOOK	_PLAY_
0	_RAINY_	YES
1	_SUNNY_	YES
2	_OVERCAST_	YES
3	_OVERCAST_	YES
4	_SUNNY_	NO
5	_RAINY_	YES
6	_SUNNY_	YES
7	_OVERCAST_	YES
8	_RAINY_	NO
9	_SUNNY_	NO
10	_SUNNY_	YES
11	_RAINY_	NO
12	_OVERCAST_	YES
13	_OVERCAST_	YES

Frequency table for the _WEATHER_ Conditions:

WEATHER	NO	YES
OVERCAST	0	5
RAINY	2	2
SUNNY	2	3
TOTAL	4	10

Likelihood table _WEATHER_ condition:

WEATHER	NO	YES	
OVERCAST	0	5	5/14=0.35
RAINY	2	2	4/14=0.29
SUNNY	2	3	5/14=0.35
ALL	4/14 = 0.29	10/14 = 0.71	

Applying Bayes'theorem:

P(Yes|_SUNNY_)= P(_SUNNY_|Yes)*P(Yes)/P(_SUNNY_)

P(_SUNNY_|Yes)= 3/10= 0.3

P(_SUNNY_)= 0.35

P(Yes)=0.71

So P(Yes|_SUNNY_) = 0.3*0.71/0.35= **0.60**

P(No|_SUNNY_)= P(_SUNNY_|No)*P(No)/P(_SUNNY_)

P(_SUNNY_|NO)= 2/4=0.5

P(No)= 0.29

P(_SUNNY_)= 0.35

So P(No|_SUNNY_)= 0.5*0.29/0.35 = **0.41**

So as we can see from the above calculation that **P(Yes|_SUNNY_)>P(No|_SUNNY_)**

Hence on a _SUNNY_ day, Player can play the game.

Check Your Progress 4

8. Predicting a class label using naïve Bayesian classification. We wish to predict the class label of a tuple using naïve Bayesian classification, given the training data as shown in Table-1 Below. The data tuples are described by the attributes age, income, student, and credit rating.

RID	age	income	student	credit_rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

The class label attribute known as "buys computer" can take on one of two distinct values—specifically, "yes" or "no." Let's say that C1 represents the class buying a computer and C2 represents the class deciding not to buy a computer. We are interested in classifying X as having the following characteristics: (age = youth, income = medium, student status = yes, credit rating = fair).

10.5.2 K-NEAREST NEIGHBOURS (K-NN)

This approach, places items in the class to which they are “closest” to their neighbour.It must determine distance between an item and a class. Classes are represented by centroid (Central value) and the individual points.One of the algorithms that is used is K-Nearest Neighbors.

We know that The classification task maps data into predefined groups or classes. Given database/dataset $D=\{t_1,t_2,\ldots,t_n\}$ and a set of classes $C=\{C_1,\ldots,C_m\}$, the classification Problem is to define a mapping $f:D\rightarrow C$ where each t_i is assigned to one class, that is, it divides database/dataset D into classes specified in the Set C.
A few very simple examples to elucidate classification could be:

- Teachers classify students’ marks data into a set of grades as A, B, C, D, or F.
- Classification of the height of a set of persons into the classes tall, medium or short.

The basic approaches to classification are:

- To create specific models by, evaluating training data, which is basically the old data, that has already been classified by using the domain of the experts' knowledge.
- Now applying the model developed to the new data.

Please note that in classification, the classes are predefined.

Some of the most common techniques used for classification may include the use of Decision Trees, K-NN etc. Most of these techniques are based on finding the distances or uses statistical methods.

The distance measure finds, the distance or dissimilarity between objects the measures that are used in this unit are as follows:

- Euclidean distance: $dis(t_i,t_j)= \sqrt{\sum_{h=1}^{k} (t_{ih} - t_{jh})^2}$
- Manhattan distance: $dis(t_i,t_j)= \sum_{h=1}^{k} |(t_{ih} - t_{jh})|$

where t_i and t_j are tuples and h are the different attributes which can take values from 1 to k

In this section, we look at the distance based classifier i.e. the k-nearest neighbor classifiers.

A test tuple is compared to training tuples that are used in the classification process that are similar to it. This is how nearest-neighbor classifiers work. There are n different characteristics that can be used to define the training tuples. Each tuple can be thought of as a point located in a space that has n dimensions. In this method, each and every one of the training tuples is preserved within an n-dimensional pattern space. A K-nearest-neighbor classifier searches the pattern space in order to find the k training tuples that are the most comparable to an unknown tuple. These k training tuples are referred to as the "k nearest neighbours" of the unknown tuple.

A distance metric, like Euclidean distance, is used to define "closeness."The Euclidean distance between two points or tuples, say, X1 = (x11, x12,..., x1n) and X2 = (x21, x22,..., x2n), is

$$dist(\boldsymbol{X_1}, \boldsymbol{X_2}) = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2}. \quad \textbf{(eq. 1)}$$

In other words, for each numeric attribute, we take the difference in value between the corresponding values of that property in tuple X1 and the values of that attribute in tuple X2 and then square this difference. Finally, we add up all of these differences. The final tally of all the accumulated distances is used to calculate the square root. In most cases, prior to making use of Equation, we first normalize the values of every property (eq. 1). This helps avoid attributes with initially high ranges (like income, for example) from outweighing attributes with originally lower ranges, which helps prevent unfairness (such as binary attributes). Min-max normalization, for example, can be used to change the value of a numeric attribute A from v to v' in the range [0, 1].

$$v' = \frac{v - min_A}{max_A - min_A},$$ **(eq. 2)**

Where, min_A and max_A are the minimum and maximum values of attribute A

For the purpose of k-nearest-neighbor classification, the unknown tuple is assigned to the class that has the highest frequency among its k closest neighbours. When k equals 1, the class of the training tuple that is assigned to the unknown tuple is the one that is most similar to the unknown tuple in pattern space. It is also possible to utilise nearest neighbour classifiers for prediction, which means that they can be used to deliver a real-valued forecast for a given unknown tuple. The result that the classifier produces in this scenario is the weighted average of the real-valued labels that are associated with the unknown tuple's k nearest neighbours.

Classification Using Distance (K-Nearest Neighbours) - Some of the basic points to be noted about this algorithm are:

- The training set includes *classes* along with other attributes. (Please refer to the training data given in the *Table* given below).
- The value of the K defines the number of *near items* (items that have less distance to the attributes of concern) that should be used from the given set of training data (just to remind you again, training data is already classified data). This is explained in point (2) of the following example.
- A new item is placed in the class in which the most number of close items are placed. (Please refer to point (3) in the following example).
- The value of K should be <= $\sqrt{Number\_of\_training\_items}$ However, in our example for limiting the size of the sample data, we have not followed this formula.

Example: Consider the following data, which tells us the person's class depending upon gender and height

Name	Gender	Height	Class
Sunita	F	1.6m	Short
Ram	M	2m	Tall
Namita	F	1.9m	Medium
Radha	F	1.88m	Medium
Jully	F	1.7m	Short
Arun	M	1.85m	Medium
Shelly	F	1.6m	Short
Avinash	M	1.7m	Short
Sachin	M	2.2m	Tall
Manoj	M	2.1m	Tall
Sangeeta	F	1.8m	Medium
Anirban	M	1.95m	Medium
Krishna	F	1.9m	Medium
Kavita	F	1.8m	Medium
Pooja	F	1.75m	Medium

1) You have to classify the tuple <Ram, M, 1.6> from the training data that is given to you.

2) Let us take only the **height** attribute for distance calculation and suppose K=5 then the following are the near five tuples to the data that is to be classified (using Manhattan distance as a measure on the height attribute).

Name	Gender	Height	Class
Sunita	F	1.6m	Short
Jully	F	1.7m	Short
Shelly	F	1.6m	Short
Avinash	M	1.7m	Short
Pooja	F	1.75m	Medium

3) On examination of the tuples above, we classify the tuple <Ram, M, 1.6> to *Short* class since most of the tuples above belongs to *Short* class.

Example- To classify whether a special paper tissue is Fine or not, we used data from a questionnaire survey (to get people's opinions) and objective testing with two properties (acid durability and strength). Here are four examples of training.

X1 = Acid_Durability_(seconds)	X2 = Strength(gram/Cm2)	Y = Classification
7	7	Poor
7	4	Poor
3	4	Fine
1	4	Fine

Now, the firm is producing a new kind of paper tissue that is successful in the laboratory and has the values X1 = 3 and X2 = 7 respectively. Can we make an educated judgement about the classification of this novel tissue without doing yet another expensive survey?

1. Find the value of the parameter K as the number of the nearest neighbours Suppose use K = 3
2. Determine the distance that separates the query-instance from each of the samples used for training

The coordinates of the query instance are (3, 7), and rather than computing the distance, we compute the square distance, which is a more efficient calculation (without square root)

X1 = Acid_Durability_(seconds)	X2 = Strength(gram/Cm2)	**Square Distance to query instance (3, 7)**
7	7	$(7-3)^2+(7-7)^2=16$
7	4	$(7-3)^2+(4-7)^2=25$
3	4	$(3-3)^2+(4-7)^2=9$
1	4	$(1-3)^2+(4-7)^2=13$

2. *Sort the distance and determine nearest neighbors based on the K-th minimum distance*

X1 = Acid_Durability_(seconds)	X2 = Strength (gram/Cm2)	Square Distance to query instance (3, 7)	**Rank minimum distance**	**Is it included in 3-Nearest neighbors?**
7	7	$(7-3)^2+(7-7)^2=16$	**3**	**Yes**
7	4	$(7-3)^2+(4-7)^2=25$	**4**	**No**
3	4	$(3-3)^2+(4-7)^2=9$	**1**	**Yes**
1	4	$(1-3)^2+(4-7)^2=13$	**2**	**Yes**

3. *Gather the category (Y) of the nearest neighbors.* Notice in the second row last scolumn that the category of nearest neighbor (Y) is not included because the rank of this data is more than 3 (=K).

X1 = Acid_Durability_(seconds)	X2 = Strength (gram/Cm2)	Square Distance to query instance (3, 7)	Rank minimum distance	Is it included in 3-Nearest neighbors?	**Y = Category of nearest Neighbor**
7	7	$(7-3)^2+(7-7)^2=16$	3	Yes	**Poor**
7	4	$(7-3)^2+(4-7)^2=25$	4	No	-
3	4	$(3-3)^2+(4-7)^2=9$	1	Yes	**Fine**
1	4	$(1-3)^2+(4-7)^2=13$	2	Yes	**Fine**

4. *Use simple majority of the category of nearest neighbors as the prediction value of the query instance*

Use the simple majority of the nearest neighbours as the query instance's prediction value.

Since 2<1 and we have 2 Fine and 1 Poor, So we can say that a new paper tissue that passed the lab test with X1 = 3 and X2 = 7 is in the **Fine** category.

"However, distance cannot be determined using qualities that are categorical, as opposed to quantitative, such as color." The preceding description operates under the presumption that all of the attributes that are used to describe the tuples are numeric. When dealing with categorical attributes, a straightforward way is to contrast the value of the attribute that corresponds to tuple X1 with the value that corresponds to tuple X2. If there is no difference between the two (for example, If tuple X1 and X2 both contain the blue color, then the difference between the two is

regarded as being equal to zero. If the two are distinct from one another (for example, if tuple X1 carries blue and tuple X2 carries red), then the comparison between them is counted as 1. It's possible that other ways will incorporate more complex systems for differentiating grades (such as in a scenario in which a higher difference score is provided, say, for blue and white than for blue and black).

"What about the missing values?" If the value of a certain attribute A is absent from either the tuple X1 or the tuple X2, we will, as a rule, assume the greatest feasible disparity between the two. Imagine for a moment that each of the traits has been mapped to the interval [0, 1]. When it comes to categorical attributes, the difference value is set to 1 if either one of the related values of A or both of them are absent. If A is a number and it is absent from both the tuple X1 and the tuple X2, then the difference is also assumed to be 1. If there is just one value that is absent and the other value (which we will refer to as v 0) is present and has been normalised, Consequently, we can either take the difference to be |1 – v' | or |0 – v' | (i.e., 1–v' or v'), depending on which of the two is larger.

Nearest-neighbor classifiers use comparisons based on distance to give each attribute an equal amount of weight. So, they can be less accurate if their attributes are noisy or don't make sense. But the method has been changed to include the weighting of attributes and the removal of noisy data tuples. How you measure distance can be very important. You could also use the city block distance or another way to measure distance.

Nearest neighbour classifiers can be very slow when classifying test tuples into groups. If D is a training database that contains |D| tuples and k is equal to one, then in order to classify a given test tuple, it must be compared to |D| training tuples. It is possible to reduce the total number of comparisons to O(log(|D|) by first putting the stored tuples in search trees and then performing the comparisons. The running time can be reduced to O(1) if parallel implementation is used, which is a constant that doesn't change no matter how big D is. You can also use partial distance calculations and change the stored tuples to cut down on the time it takes to classify. In the partial distance method, we use only some of the n attributes to figure out how far apart two things are. If this distance exceeds a specified threshold, the procedure aborts the execution of the current stored tuple and continues on to the next. Training tuples that aren't required are removed using the editing procedure. This strategy is also known as pruning or condensing. because it minimises the number of stored tuples.

Check Your Progress 5

9. Apply KNN classification algorithm to the following data and predict value for (10,7) for K = 3

Feature 1	Feature 2	Class
1	1	A
2	3	A
2	4	A
5	3	A
8	6	B
8	8	B
9	6	B
11	7	B

……

……

10.5.3 DECISION TREES

Given a data set D = $\{t_1,t_2 \ldots, t_n\}$ where $t_i=<t_{i1}, \ldots, t_{ih}>$, that is, each tuple is represented by h attributes, assume that, the database schema contains attributes as $\{A_1, A_2, \ldots, A_h\}$. Also, let us suppose that the classes are $C=\{C_1, \ldots., C_m\}$, then:

Decision or Classification Tree is a tree associated with D such that

- Each internal node is labeled with attribute, A_i
- Each arc is labeled with the predicate which can be applied to the attribute at the parent node.
- Each leaf node is labeled with a class, C_j

Basics steps in the Decision Tree are as follows:

- Building the tree by using the training set dataset/database.
- Applying the tree to the new dataset/database.

Decision Tree Induction is the process of learning about the classification using the inductive approach. During this process, we create a decision tree from the training data. This decision tree can, then be used, for making classifications. To define this we need to define the following.

Let us assume that we are given probabilities $p_1, p_2, .., p_s$ whose sum is 1. Let us also define the term Entropy, which is the measure of the amount of randomness or surprise or uncertainty. Thus our basic

goal in the classification process is that, the entropy for a classification should be zero, that, if no surprise then, entropy is equal to zero. Entropy is defined as:

$$H(p_1,p_2,\ldots p_s)= \sum_{i=1}^{s} (p_i * \log(1/p_i)) \qquad \ldots\ldots. \quad (1)$$

ID3 Algorithm for Classification

This algorithm creates a tree using the algorithm given below and tries to reduce the expected number of comparisons.

Algorithm: ID3 algorithm for creating decision tree from the given training data.
Input: The *training data* and the *attribute-list*.

Output: A decision tree.

Process: Step 1: Create a node N

Step 2: If all of the sample data belong to the same class, C, which means the probability is 1, then return N as a leaf node with the class C label.

Step 3: Return N as a leaf node if attribute-list is empty, and label it with the most common class with in training data; // majority voting

Step 4: Select *split-attribute*, which is the attribute in the *attribute-list* with the highest information gain;

Step 5: label node N with *split-attribute*;

Step 6: for each known value Ai, of *split-attribute* // partition the samples

Create a branch from node N for the condition: *split-attribute* = A_i;

// Now consider a partition and recursively create the decision tree:

Let xi be the set of data from training data that satisfies the condition:

split-attribute = A_i

if the set xi is empty then

attach a leaf labeled with the most common class in the prior set of training data;

else

attach the node returned after recursive call to the program with training data as xi and

new attribute list = present attribute-list – split-attribute;

End of Algorithm.

Please note: The algorithm given above, chooses the split attribute with the highest information gain, that is, calculated as follows:

$$\text{Gain (D,S)} = H(D) - \sum_{i=1}^{s} (P(D_i) * H(D_i)) \quad \text{.....(2)}$$

where S is new states $=\{D_1, D_2, D_3 \ldots D_S\}$ and H(D) finds the amount of order in that state
Consider the following data in which *Position* attribute acts as class

Department	Age	Salary	Position
PERSONNEL	31-40	Medium_Range	_BOSS_
PERSONNEL	21-30	Low_Range	_ASSISTANT_
PERSONNEL	31-40	Low_Range	_ASSISTANT_
MIS	21-30	Medium_Range	_ASSISTANT_
MIS	31-40	High_Range	_BOSS_
MIS	21-30	Medium_Range	_ASSISTANT_
MIS	41-50	High_Range	_BOSS_
ADMIN	31-40	Medium_Range	_BOSS_
ADMIN	31-40	Medium_Range	_ASSISTANT_
SECURITY	41-50	Medium_Range	_BOSS_
SECURITY	21-30	Low_Range	_ASSISTANT_

Figure 3: Sample data for classification

We are applying ID3 algorithm, on the above dataset as follows:
The initial entropy of the dataset using formula at (1) is

$$H(\text{initial}) = (6/11)\log(11/6) + (5/11)\log(11/5) = 0.29923$$
(_ASSISTANT_) (_BOSS_)

Now let us calculate gain for the departments using the formula at (2)

Gain(Department) = H(initial) – [P(_PERSONNEL_) * H(_MIS_) + P(_MIS_) * H(_PERSONNEL_) +
P(_ADMIN_) * H(_ADMIN_) + P(_SECURITY_) * H(_SECURITY_)]

= *0.29923*- { (3/11)[(1/3)log3 +(2/3)log(3/2)] + (4/11)[(2/4)log 2 + (2/4)log 2] +
(2/11)[(1/2)log 2 + (1/2)log 2] + (2/11)[(1/2)log 2 + (1/2)log 2] }
= 0.29923- 0.2943
= 0.0049

Similarly:

Gain(Age) = 0.29923- { (4/11)[(4/4)log(4/4)] + (5/11)[(3/5)log(5/3) + (2/5)log (5/2)] +
(2/11)[(2/2) log (2/2)] }
= 0.29923 - 0.1328
= 0.1664

Gain(Salary) = 0.29923 – { (3/11)[(3/3)log 3] + (6/11)[(3/6) log2 + (3/6)log 2] +
(2/11) [(2/2 log(2/2)) }
= 0.29923 – 0.164
= 0.1350

Since age has the maximum gain, so, this attribute is selected as the first splitting attribute. In age range 31-40, class is not defined while for other ranges it is defined.

So, we have to again calculate the spitting attribute for this age range (31-40). Now, the tuples that belong to this range are as follows:

Department	Salary	Position
PERSONNEL	Medium_Range	_BOSS_
PERSONNEL	Low_Range	_ASSISTANT_
MIS	High_Range	_BOSS_
ADMIN	Medium_Range	_BOSS_
ADMIN	Medium_Range	_ASSISTANT_

Again the initial entropy = (2/5)log(5/2) + (3/5)log(5/3) = 0.29922
(_ASSISTANT_) (_BOSS_)

Gain(Department)= 0.29922- { (2/5)[(1/2)log 2 + (1/2)log 2] +1/5[(1/1)log1] +
(2/5)[(1/2)log 2 + (1/2)log 2] }
=0.29922- 0.240
=0.05922
Gain (Salary) = 0.29922- { (1/5) [(1/1)log1] +(3/5)[(1/3)log 3 + (2/3)log(3/2)] +
(1/5) [(1/1)log1] }
= 0.29922- 0.1658
=0.13335
The Gain is maximum for salary attribute, so we take salary as the next splitting attribute. In middle range salary, class is not defined while for other ranges it is defined. So, we have to again calculate the spitting attribute for this middle range. Since only department is left, so, department will be the next splitting attribute. Now, the tuples that belong to this salary range are as follows:

Department	Position
PERSONNEL	_BOSS_
ADMIN	_BOSS_
ADMIN	_ASSISTANT_

Again in the _PERSONNEL_ department, all persons are _BOSS_, while, in the _ADMIN_ there is a tie between the classes. So, the person can be either _BOSS_ or _ASSISTANT_ in the _ADMIN_ department.

Now the decision tree will be as follows:

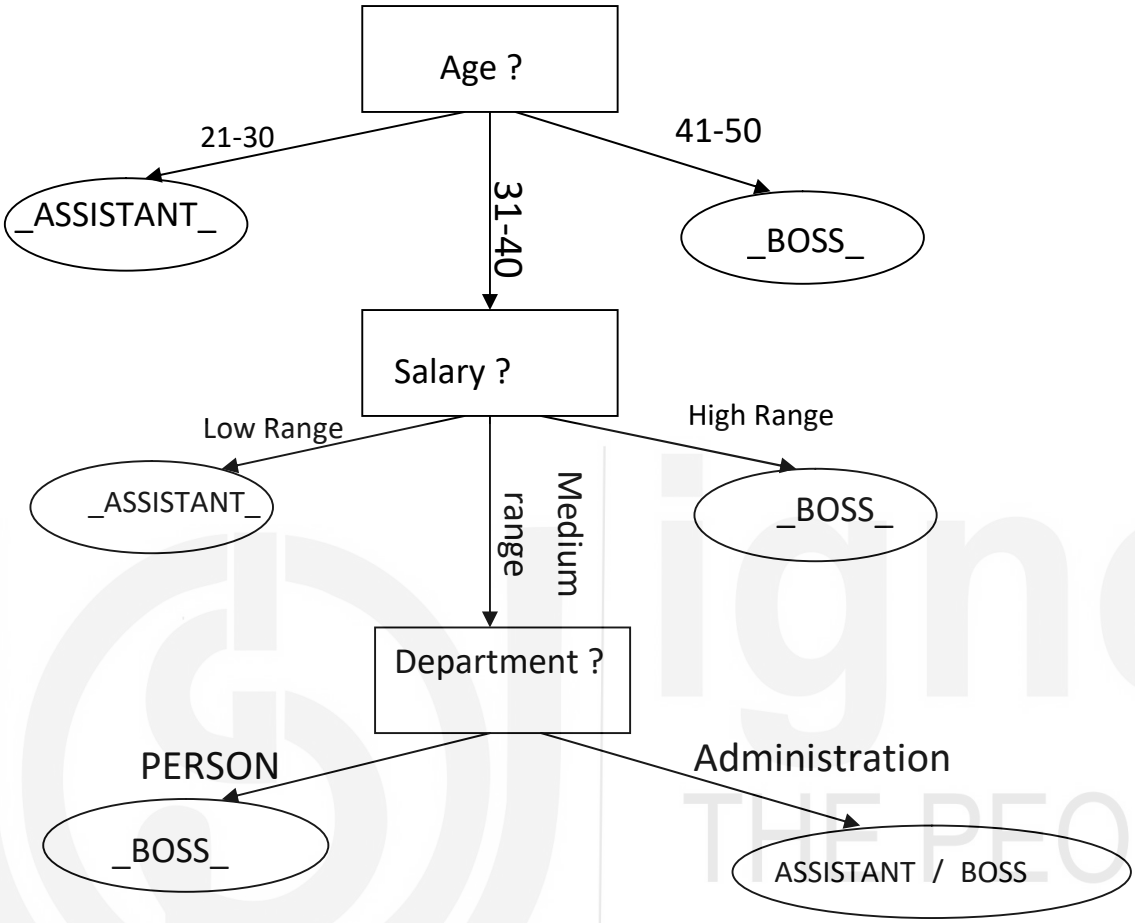


Figure 4: The decision tree using ID3 algorithm for the sample data

Now, we will take a new dataset and we will classify the class of each tuple by applying the decision tree that we have built above.

Steps of algorithm of decision tree

1. Data Pre-processing step
2. Fitting a Decision-Tree algorithm to the Training set
3. Predicting the test result
4. Test accuracy of the result(Creation of Confusion matrix)
5. Visualizing the test set result.

Example : Problem on Decision Tree - Consider whether a dataset based on which we will determine whether to play football or not.

OUTLOOK	TEMP.	HUMIDITY	WIND	PLAY FOOTBALL(YES/NO)
SUNNY	_HOT_	HIGH	WEAK	NO
SUNNY	_HOT_	HIGH	STRONG	NO
OVERCAST	_HOT_	HIGH	WEAK	YES
RAINY	_MILD_	HIGH	WEAK	YES
RAINY	_COOL_	NORMAL	WEAK	YES
RAINY	_COOL_	NORMAL	STRONG	NO
OVERCAST	_COOL_	NORMAL	STRONG	YES
SUNNY	_MILD_	HIGH	WEAK	NO
SUNNY	_COOL_	NORMAL	WEAK	YES
RAINY	_MILD_	NORMAL	WEAK	YES
SUNNY	_MILD_	NORMAL	STRONG	YES
OVERCAST	M	HIGH	STRONG	YES
OVERCAST	_HOT_	NORMAL	WEAK	YES
RAINY	_MILD_	HIGH	STRONG	NO

Here There are for independent variables to determine the dependent variable. The independent variables are _OUTLOOK_, TEMP., Humidity, and Wind. The dependent variable is whether to play football or not.

As the first step, we have to find the parent node for our decision tree. For that follow the steps:

Find the entropy of the class variable. E(S) = -[(9/14)log(9/14) + (5/14)log(5/14)] = 0.94

note: Here typically we will take log to base 2.Here total there are 14 yes/no. Out of which 9 yes and 5 no.Based on it we calculated probability above.

From the above data for _OUTLOOK_ we can arrive at the following table easily

		PLAY		
		YES	NO	TOTAL
	SUNNY	3	2	5
OUTLOOK	_OVERCAST_	4	0	4
	RAINY	2	3	5
				14

Now we have to calculate average weighted entropy. ie, we have found the total of weights of each feature multiplied by probabilities.

E(S, _OUTLOOK_) = (5/14)*E(3,2) + (4/14)*E(4,0) + (5/14)*E(2,3) = (5/14)(-(3/5)log(3/5)-(2/5)log(2/5))+ (4/14)(0) + (5/14)((2/5)log(2/5)-(3/5)log(3/5)) = 0.693

The next step is to find the information gain. It is the difference between parent entropy and average weighted entropy we found above.

IG(S, _OUTLOOK_) = 0.94 - 0.693 = 0.247

Similarly find Information gain for TEMP., Humidity, and Windy.

IG(S, TEMP.) = 0.940 - 0.911 = 0.029

IG(S, Humidity) = 0.940 - 0.788 = 0.152

IG(S, Windy) = 0.940 - 0.8932 = 0.048

Now select the feature having the largest entropy gain. Here it is _OUTLOOK_. So it forms the first node(root node) of our decision tree.

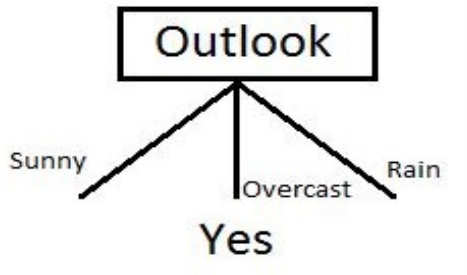
Now our data look as follows

OUTLOOK	TEMP.	HUMIDITY	WIND	PLAYED(YES/NO)
SUNNY	_HOT_	HIGH	WEAK	NO
SUNNY	_HOT_	HIGH	STRONG	NO
SUNNY	_MILD_	HIGH	WEAK	NO
SUNNY	_COOL_	NORMAL	WEAK	YES
SUNNY	_MILD_	NORMAL	STRONG	YES

OUTLOOK	TEMP.	HUMIDITY	WIND	PLAYED(YES/NO)
OVERCAST	_HOT_	HIGH	WEAK	YES
OVERCAST	_COOL_	NORMAL	STRONG	YES
OVERCAST	_MILD_	HIGH	STRONG	YES
OVERCAST	_HOT_	NORMAL	WEAK	YES

OUTLOOK	TEMP.	HUMIDITY	WIND	PLAYED(YES/NO)
RAIN	_MILD_	HIGH	WEAK	YES
RAIN	_COOL_	NORMAL	WEAK	YES
RAIN	_COOL_	NORMAL	STRONG	NO
RAIN	_MILD_	NORMAL	WEAK	YES
RAIN	_MILD_	HIGH	STRONG	NO

Since _OVERCAST_ contains only examples of class ‘Yes’ we can set it as yes. That means If _OUTLOOK_ is _OVERCAST_ football will be played. Now our decision tree looks as follows.



The next step is to find the next node in our decision tree. Now we will find one under _SUNNY_. We have to determine which of the following TEMP., Humidity or Wind has higher information gain.

OUTLOOK	TEMP.	HUMIDITY	WIND	PLAYED(YES/NO)
SUNNY	_HOT_	HIGH	WEAK	NO
SUNNY	_HOT_	HIGH	STRONG	NO
SUNNY	_MILD_	HIGH	WEAK	NO
SUNNY	_COOL_	NORMAL	WEAK	YES
SUNNY	_MILD_	NORMAL	STRONG	YES

Calculate parent entropy E(_SUNNY_)

E(_SUNNY_) = (-(3/5)log(3/5)-(2/5)log(2/5)) = 0.971.

Now Calculate the information gain of TEMP.. IG(_SUNNY_, TEMP.)

		PLAY		
		YES	NO	TOTAL
	HOT	0	2	2
TEMP.	_COOL_	1	1	2
	MILD	1	0	1
				5

E(_SUNNY_, TEMP.) = (2/5)*E(0,2) + (2/5)*E(1,1) + (1/5)*E(1,0)=2/5=0.4

Now calculate information gain.

IG(_SUNNY_, TEMP.) = 0.971–0.4 =0.571

Similarly we get

IG(_SUNNY_, Humidity) = 0.971

IG(_SUNNY_, Windy) = 0.020

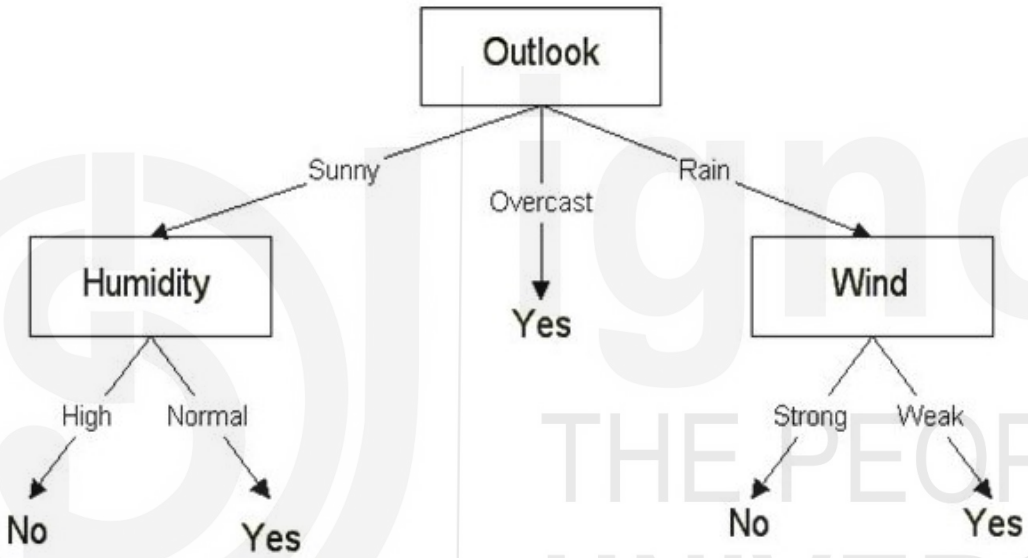
Here IG(_SUNNY_, Humidity) is the largest value. So Humidity is the node that comes under _SUNNY_.

		PLAY		
		YES	NO	TOTAL
	HIGH	0	3	3
HUMIDITY	NORMAL	2	0	2
				5

For humidity from the above table, we can say that play will occur if humidity is normal and will not occur if it is high. Similarly, find the nodes under _RAINY_.

Note: A branch with entropy more than 0 needs further splitting.

Finally, our decision tree will look as below:



10.5.4 LOGISTIC REGRESSION

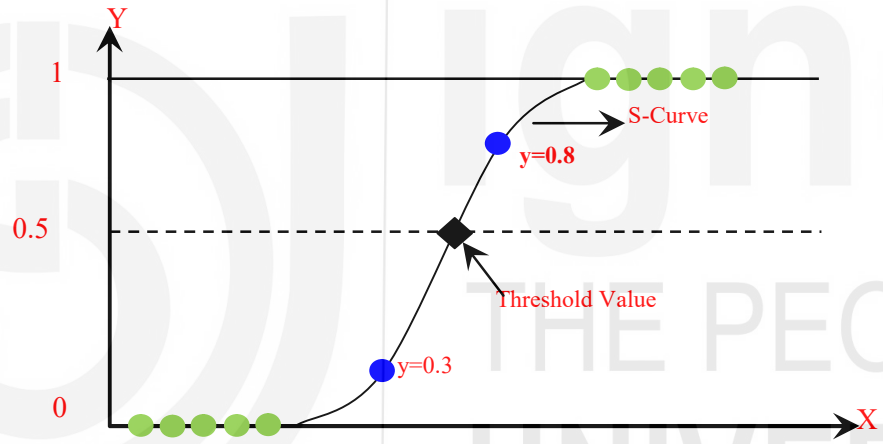
Logistic Regression in Machine Learning

- Logistic regression, which is part of the Supervised Learning method, is one of the most popular Machine Learning algorithms. It is used to predict the categorical dependent variable based on a set of independent variables.
- Logistic regression predicts the outcome of a dependent variable that has a "yes" or "no" answer. Because of this, the result must be a discrete or categorical value. It can be Yes or No, 0 or 1, true or false, etc., but instead of giving the exact value as 0 or 1, it gives the probabilistic values that lie between 0 and 1.

- Logistic Regression is a lot like Linear Regression, but the way they are used is different. Linear regression is used to solve regression problems, while logistic regression is used to solve classification problems.
- In logistic regression, we fit a "S"-shaped logistic function, which predicts two maximum values, instead of a regression line (0 or 1).
- The curve from the logistic function shows how likely something is, like whether the cells are cancerous or not, whether a mouse is overweight or not based on its weight, etc.

Logistic Regression is an important machine learning algorithm because it can use both continuous and discrete datasets to give probabilities and classify new data.

Logistic regression can be used to classify observations based on different types of data, and it is easy to figure out which variables are the most useful for classifying. The logistic function is shown in the picture below:



Note: Logistic regression is based on the idea of predictive modeling as regression, so that's why it's called "logistic regression." However, it's used to classify samples, so it's a part of the classification algorithm.

Logistic Function (Sigmoid Function):

- The math "function" called the "sigmoid function" turns predicted values into probabilities.
- The sigmoid function is a special case of the logistic function. It is usually written as (x) or sig (x). The formula for it is: $\sigma(x) = 1/(1+\exp(-x))$
- It turns any real number into a different number between 0 and 1.
- The value of the logistic regression must be between 0 and 1, and it can't be higher than that. Because of this, it forms a curve that looks like the letter "S." The Sigmoid function or the logistic function is the name for the curve in the shape of a S.

- The threshold value tells us how likely it is that either 0 or 1 will happen in logistic regression. For example, most values above the threshold are 1, and most values below it are 0.

Assumptions for Logistic Regression:

- The dependent variable must be a categorical one.
- The independent variable shouldn't be related to more than one other variable.

Types of Logistic Regression: Based on the categories, Logistic Regression can be divided into three types:

• **Binomial**: In binomial logistic regression, the dependent variables can only be either 0 or 1, Pass or Fail, etc.

• **Multinomial**: In multinomial Logistic regression, the dependent variable can be one of three or more types that are not in order, such as "cats," "dogs," or "sheep."

• **Ordinal:** In ordinal Logistic regression, the dependent variables can be ranked, such as "low," "medium," or "high."

Logistic Regression Equation From the Linear Regression equation, you can figure out what the Logistic Regression equation is. Here are the steps you need to take in math to get Logistic Regression equations:

- We know the equation of the straight line can be written as:

  $$y = b_0+b_1X_1+b_2X_2+\ldots\ldots+b_nX_n$$

- In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y): y/(1-y); 0 for y=0 and infinity for y=1
- But we need range between -[infinity] to +[infinity], then take logarithm of the equation it will become: $\mathbf{Log[y/(1-y)] = b_0+b_1X_1+b_2X_2+\ldots\ldots+b_nX_n}$

The above equation is the final equation for Logistic Regression.

The relationship between a numerical response and a numerical or categorical predictor is the subject of the statistical technique known as simple linear regression. While multiple regression looks at the relationship between a single numerical response and a number of different numerical and/or categorical predictors, single regression looks at the relationship between a single numerical response and a single. What should be done, however, when the predictors are odd (nonlinear, intricate dependence structure, and so on), or when the response is unusual (categorical, count data, and so on)? When this occurs, we deal with odds, which are another method of measuring the likelihood of an event and are frequently applied in the context of gambling (and logistic regression).

Odds For some event E is expressed as,

$$odds(E) = P(E)/P(Ec\ ) = P(E)/(1 - P(E))$$

Similarly, if we are told the odds of E are x to y then

$$odds(E) = x/\ y = \{x/(x + y)\}/\{y/(x + y)\}$$

which implies P(E) = x/(x + y), P(E c) = y/(x + y)

Logistic regression is a statistical approach for modelling a binary categorical variable using numerical and categorical predictors, and this idea of Odds is commonly employed in it. We suppose the outcome variable was generated by a binomial distribution, and we wish to build a model with p as the probability of success for a given collection of predictors. There are other alternatives, but the logit function is the most popular.

$$\text{Logit function: } \text{logit}(p) = \log\{p/(1-p)\}, \text{ for } 0 \le p \le 1$$

Example-1: In a survey of 250 customers of an auto dealership, the service department was asked if they would tell a friend about it. The number of people who said "yes" was 210, where "p" is the percentage of customers in the group from which the sample was taken who would answer "yes" to the question. Find the sample odds and sample proportion.

Solution: The number of customers who would respond Yes in an simple random sample (SRS) of size n has the binomial distribution with parameters n and p. The sample size of customers is n = 250, and the number who responded Yes is the count X = 210. Therefore, the sample proportion is p'=210/250 = 0.84

Since, Logistic regressions work with odds rather than proportions. We need to calculate the Odds, the odds are simply the ratio of the proportions for the two possible outcomes. If p' is the proportion for one outcome, then 1 – p' is the proportion for the second out

$$\text{odds} = p' / (1 - p')$$

A similar formula for the population odds is obtained by substituting p for p' in this expression

Odds of responding Yes. For the customer service data, the proportion of customers who would recommend the service in the sample of customers is p' = 0.84, so the proportion of customers who would not recommend the service department will be 1 – p' i.e. 1 – p' = 1 - 0.84 = 0.16

Therefore, the odds of recommending the service department are

$$\text{odds} = p'/1 - p' = 0.84/0.16 = 5.25$$

When people speak about odds, they often round to integers or fractions. If we round 5.25 to 5 = 5/1, we would say that the odds are approximately 5 to 1 that a customer would recommend the service to a friend. In a similar way, we could describe the odds that a customer would not recommend the service as 1 to 5.

Check Your Progress 6

Q1 Odds of drawing a heart. If you deal one card from a standard deck, the probability that the card is a heart is 13/52 = 1/4.

(a) Find the odds of drawing a heart.
(b) Find the odds of drawing a card that is not a heart.

10.5.5 SUPPORT VECTOR MACHINES

Support Vector Machine, also called Support Vector Classification, is a supervised and linear Machine Learning technique that is most often used to solve classification problems. In this section, we will take a look at Support Vector Machines, a new approach for categorising data that has a lot of potential and can be used for both linear and nonlinear datasets. A support vector machine, often known as an SVM, is a type of algorithm that transforms the primary training data into a new format that has a higher dimension by making use of a nonlinear mapping. It searches for the ideal linear separating hyperplane in this additional dimension. This hyperplane is referred to as a "decision boundary" since it separates the tuples of one class from those of another. A hyperplane can always be used to split data from two classes if the appropriate nonlinear mapping to a high enough dimension is used. This hyperplane is located by the SVM through the use of support vectors, also known as important training tuples, and margins (defined by the support vectors). We'll go into further detail about these fresh concepts in the next paragraphs.

While studying machine learning, one of the classifiers that we come across is called a Support Vector Machine, or SVM for short. One of the most common approaches for categorising data in the field of machine learning, which performs admirably on both small and large datasets. SVMs, which stands for support vector machines, can be utilised for both classification and regression jobs; however, their performance is superior when applied to classification scenarios. When they were first introduced in the 1990s, they quickly became quite popular, and even now, with only minor adjustments, they are the solution of choice when a high-performing algorithm is required.

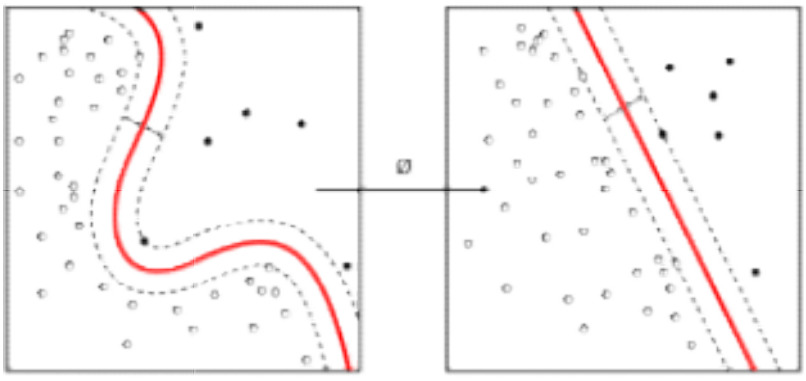


Fig1. Showing data into two groups

There are two main categories that can be applied to SVM:

• **Linear Support Vector Machine**: You can only use Linear SVM if the data can be completely separated into linear categories. The ability to separate a set of data points into two classes using just one straight line is what is meant when we talk about something being "completely linearly separable" (if 2D).

• **Non-Linear Support Vector Machine**: When the data isn't linearly separable, we can use Non-Linear SVM, which means that we apply advanced techniques like kernel tricks to categorise the data points that can't be divided into two classes using a straight line. This allows us to use Non-Linear SVM to classify the data points that aren't linearly separable (in 2D). We don't discover datapoints that are linearly separable in the majority of real-world applications, so we use the kernel approach to solve them instead.

Let's take a look of some SVM terminology.

- **Support Vectors:** The points on the hyperplane that are closest to the object in question are referred to as support vectors. The boundary between the two groups will be determined with the help of these data points.Infact these are the spots on the hyperplane that are closest to it. These data points will be used to define a separation line.
- **Margin:** The margin is the distance between the hyperplane and the nearest observations to the hyperplane (support vectors). A margin that is high is considered to be a favourable margin by SVM.In Short, It's the distance between the hyperplane and the hyperplane's nearest observations (support vectors). SVM considers a high margin to be a favourable margin.

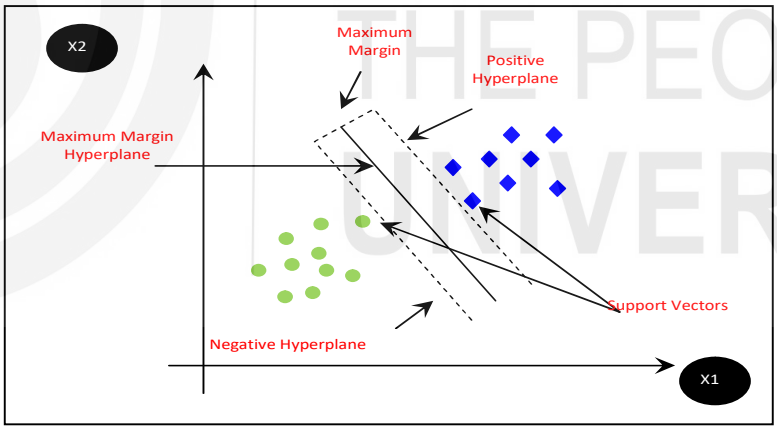


Fig2. Diagram of Support Vector Machine

Working of SVM

SVM is defined solely in terms of support vectors; we do not need to be concerned with any other observations because the margin is calculated based on the points that are support vectors that are closest to the hyperplane. This is in contrast to logistic regression, in which the classifier is defined over all of the points. Because of this, SVM is able to take use of some natural speedups.

To further understand how SVM operates, let's look at an example. Suppose we have a dataset that has two different classes (green and blue). It is necessary for us to decide whether the new data point should be classified as blue or green.

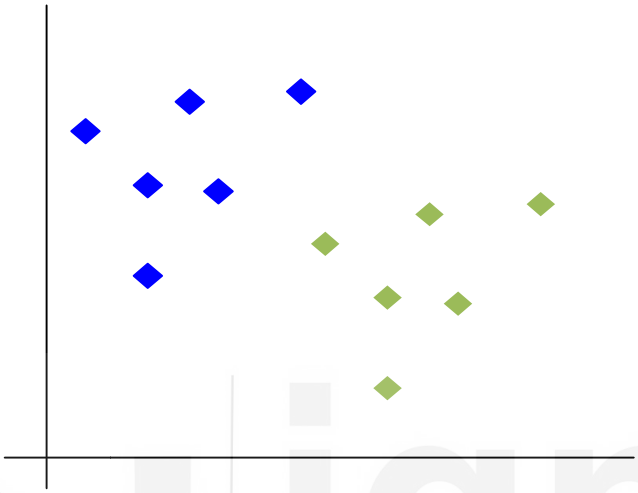


Fig 3. Dataset with two classes.

There are many ways to put these points into groups, but the question is which is the best and how do we find it?

NOTE: We call this decision boundary a "straight line" because we are plotting data points on a two-dimensional graph. If there are more dimensions, we call it a "hyperplane."

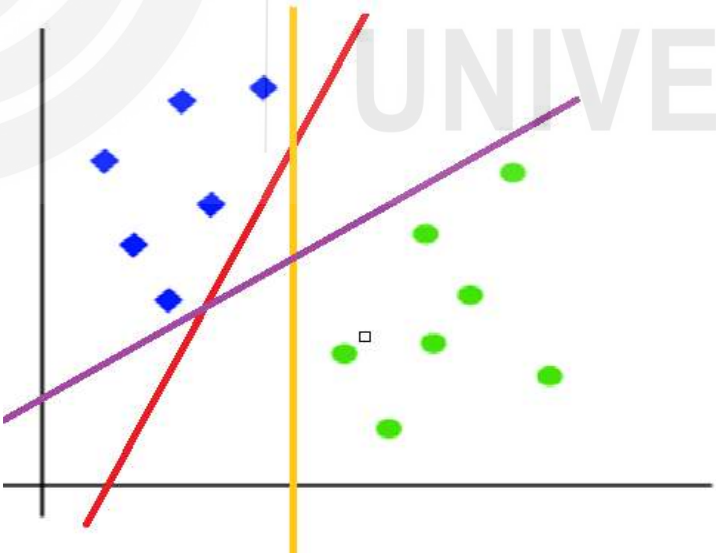


Fig4. Hyperplane

SVM is all about finding the hyperplane with the most space between the two classes, such hyperplane is the best hyperplane. This is done by finding many hyperplanes that best fit the labels and then picking the one that is farthest from the data points or has the biggest margin.i.e. The best hyperplane is the one with the greatest distance between the two classes, and this is what SVM is all about.

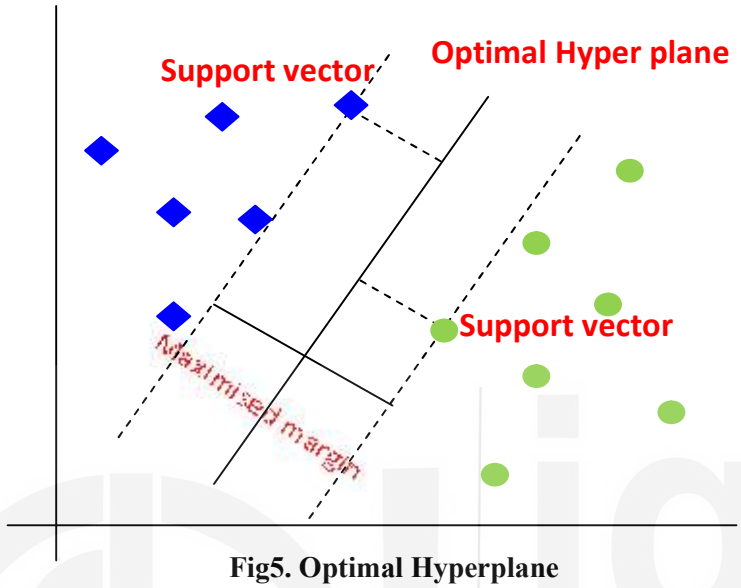


Fig5. Optimal Hyperplane

In geometry, a hyperplane is the name given to a subspace that is one dimension smaller than the ambient space. Despite the fact that this definition is accurate, it is not very clear. Instead of making use of it, we will concentrate on acquiring knowledge about lines in order to better understand what a hyperplane is. If you can recall the mathematics that you studied in high school, you presumably know that a line has an equation of the form, that the constant is called the slope, and that the y-axis is crossed by. If you can't remember those things, you should look them up. It is important to note, however, that the linear equation y = a x + b involves two variables. These variables are denoted by the letters y and x, but we are free to give them any name we like. This formula is valid for a wide range of possibilities for the value of, and we refer to the collection of those possibilities as a line.

Another notation for the equation of a line may be obtained if we define the two-dimensional vectors x=(x1,x2) and w=(a,-1) as follows: where w.x is the dot product of w and x.

$$\mathbf{w.x+b=0}$$

Now we need to locate a hyperplane : locating a hyperplane with the largest margin (a margin is a buffer zone around the hyperplane equation), and working toward having the largest margin while having the fewest points possible (known as support vectors).

"The goal is to maximise the minimum distance," to put it another way. for the sake of distance. If the point from the positive group is substituted in the hyperplane equation while generating predictions on the training data that was binary classified as positive and negative groups, we will get a value larger than 0. (zero), Mathematically, $w^T(\mathbf{\Phi(x)}) + \mathbf{b} > \mathbf{0}$ And predictions from the negative group in the hyperplane equation would give negative value as $w^T(\mathbf{\Phi(x)}) + \mathbf{b} < \mathbf{0}$. The indicators, on the other hand, were about

training data, which is how we're training our model. Give a positive sign for a positive class and a negative sign for a negative class.

However, if we properly predict a positive class (positive sign or greater than zero sign) as positive while testing this model on test data, then two positives equals positive and hence a greater than zero result. The same is true if we correctly forecast the negative group, because two negatives equal a positive.

However, if the model incorrectly identifies the positive group as a negative group, one plus and one minus equals a minus, resulting in a result that is less than zero. Thus summarising this we can say that The product of a predicted and actual label would be greater than 0 (zero) on correct prediction, otherwise less than zero.

$$y_n\left[w^T\phi(x) + b\right] = \begin{cases} \geq 0 \; if \; correct \\ < 0 \; if \; incorrect \end{cases}$$

CHECK YOUR PROGRESS-7

11. Suppose you are using a Linear SVM classifier with 2 class classification problem. Now you have been given the data in which some points are circled red that are representing support vectors. If you remove any one red points from the data. Does the decision boundary will change?

A) Yes
B) No

12. The effectiveness of an SVM depends upon:

A) Selection of Kernel
B) Kernel Parameters
C) Soft Margin Parameter C
D) All of the above

13. The SVM's are less effective when:

A) The data is linearly separable
B) The data is clean and ready to use
C) The data is noisy and contains overlapping points

10.11 SOLUTIONS/ANSWERS

□ Check Your Progress 1

1. Compare between Supervised and Un-Supervised Learning.

Solution : Refer to section 10.3

2. List the Steps Involved in Supervised Learning

Solution : Refer to section 10.3

3. What are the Common Issues Faced While Using Supervised Learning

Solution : Refer to section 10.3

□ Check Your Progress 2

4. Compare between Multi Class and Multi Label Classification

   Solution : Refer to section 10.4

5. Compare between structured and unstructured data

   Solution : Refer to section 10.4

6. Compare between Lazy learners and Eager Learners algorithms for machine learning.

   Solution : Refer to section 10.4

□ Check Your Progress 3

7. List the classification algorithms under the categories of Linear and Non-Linear Models. Also Discuss the various methods used for evaluating a classification model

   Solution : Refer to section 10.5

□ Check Your Progress 4

8. Using naive Bayesian classification, predict a class label. Given the training data in Table-1 below, we want to use naive Bayesian classification to predict the class label of a tuple. The characteristics age, income, student, and credit rating characterise the data tuples.

RID	age	income	student	credit_rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Solution : Also Refer to section 10.5.1

The buys computer class label attribute has two unique values (yes and no). Let C1 represent the buys computer = yes class and C2 represent the buys computer = no class. The tuple we want to categorise is

X = (youthful age, medium income, student status, fair credit rating)

For i = 1, 2, we must maximise P(X|Ci)P(Ci). The prior probability for each class, P(Ci), can be calculated using the training tuples:

P(buys computer = yes) = 9/14 = 0.643
P(buys computer = no) = 5/14 = 0.357
To compute PX|Ci), for i = 1, 2, we compute the following conditional probabilities:
P(age = youth | buys computer = yes) = 2/9 = 0.222
P(age = youth | buys computer = no) = 3/5 = 0.600
P(income = medium | buys computer = yes) = 4/9 = 0.444
P(income = medium | buys computer = no) = 2/5 = 0.400
P(student = yes | buys computer = yes) = 6/9 = 0.667
P(student = yes | buys computer = no) = 1/5 = 0.200
P(credit rating = fair | buys computer = yes) = 6/9 = 0.667
P(credit rating = fair | buys computer = no) = 2/5 = 0.400
Using the above probabilities, we obtain

P(X|buys computer = yes) = P(age = youth | buys computer = yes) ×
P(income = medium | buys computer = yes) ×
P(student = yes | buys computer = yes) ×
P(credit rating = fair | buys computer = yes)
= 0.222×0.444×0.667×0.667 = 0.044.

Similarly, P(X|buys computer = no) = 0.600×0.400×0.200×0.400 = 0.019.

To find the class, Ci , that maximizes P(X|Ci)P(Ci), we compute
P(X|buys computer = yes)P(buys computer = yes) = 0.044×0.643 = 0.028
P(X|buys computer = no)P(buys computer = no) = 0.019×0.357 = 0.007

Therefore, the naïve Bayesian classifier predicts buys computer = yes for tuple X.

Check Your Progress 5

9. Apply KNN classification algorithm to the following data and predict value for (10,7) for K = 3

Feature 1	Feature 2	Class
1	1	A
2	3	A

2	4	A
5	3	A
8	6	B
8	8	B
9	6	B
11	7	B

Solution : Refer to section 10.5.2

□ Check Your Progress 6

10. Odds of drawing a heart. If you deal one card from a standard deck, the probability that the card is a heart **is 13/52 = 1**/4.

(a) Find the odds of drawing a heart.
(b) Find the odds of drawing a card that is not a heart.

Solution : Refer to section 10.5.4

□ Check Your Progress 7

11. Suppose you are using a Linear SVM classifier with 2 class classification problem. Now you have been given the data in which some points are circled red that are representing support vectors. If you remove any one red points from the data. Does the decision boundary will change?

A) Yes
B) No

Solution: A

12. The effectiveness of an SVM depends upon:

A) Selection of Kernel
B) Kernel Parameters
C) Soft Margin Parameter C
D) All of the above

Solution: D

The SVM effectiveness depends upon how you choose the basic 3 requirements mentioned above in such a way that it maximises your efficiency, reduces error and overfitting.

13. The SVM's are less effective when:

A) The data is linearly separable
B) The data is clean and ready to use
C) The data is noisy and contains overlapping points

Solution: C

When the data has noise and overlapping points, there is a problem in drawing a clear hyperplane without misclassifying.

10.12 FURTHER READINGS

1. Machine learning an algorithm perspective, Stephen Marsland, 2nd Edition, CRC Press,, 2015.
2. Machine Learning, Tom Mitchell, 1st Edition, McGraw- Hill, 1997.
3. Machine Learning: The Art and Science of Algorithms that Make Senseof Data, Peter Flach, 1st Edition, Cambridge University Press, 2012.