UNIT-I

Introduction

Definition of machine learning

Arthur Samuel, an early American leader in the field of computer gaming and artificial intelligence, coined the term "Machine Learning" in 1959 while at IBM. He defined machine learning as "the field of study that gives computers the ability to learn without being explicitly programmed." However, there is no universally accepted definition for machine learning. Different authors define the term differently. We give below two more definitions.

1. Machine learning is programming computers to optimize a performance criterion using example data or past experience. We have a model defined up to some parameters, and learning is the execution of a computer program to optimize the parameters of the model using the training data or past experience. The model may be predictive to make predictions in the future, or descriptive to gain knowledge from data, or both.

2. The field of study known as machine learning is concerned with the question of how to construct computer programs that automatically improve with experience.

Definition of learning

A computer program is said to *learn* from experience E with respect to some class of tasks T and performance measure P , if its performance at tasks T , as measured by P , improves with experience E.

Examples

i) Handwriting recognition learning problem

- Task T : Recognizing and classifying handwritten words within images
- Performance P : Percent of words correctly classified
- Training experience E: A dataset of handwritten words with given

classifications ii)A robot driving learning problem

- Task T : Driving on highways using vision sensors
- Performance measure P : Average distance traveled before an error
- training experience: A sequence of images and steering commands recorded while observing a human driver

iii) A chess learning problem

- Task T : Playing chess
- Performance measure P : Percent of games won against opponents
- Training experience E: Playing practice games against itself

Definition

A computer program which learns from experience is called a *machine learning program* or simply a *learning program*. Such a program is sometimes also referred to as a *learner*.

How machines learn

Basic components of learning process

The learning process, whether by a human or a machine, can be divided into four components, namely, data storage, abstraction, generalization and evaluation. Figure 1.1 illustrates the various components and the steps involved in the learning process.

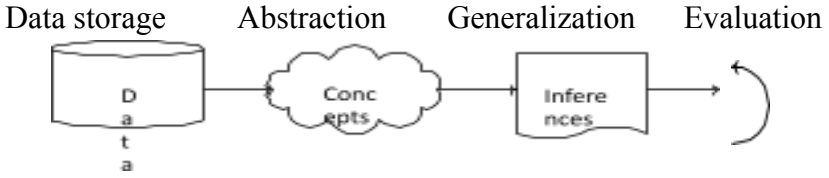


Figure 1.1: Components of learning process

1. ***Data storage***

   Facilities for storing and retrieving huge amounts of data are an important component of the learning process. Humans and computers alike utilize data storage as a foundation for advanced reasoning.

   - In a human being, the data is stored in the brain and data is retrieved using electrochem- ical signals.
   - Computers use hard disk drives, flash memory, random access memory and similar devices to store data and use cables and other technology to retrieve data.

2. ***Abstraction***

   The second component of the learning process is known as *abstraction*.

   Abstraction is the process of extracting knowledge about stored data. This involves creating general concepts about the data as a whole. The creation of knowledge involves application of known models and creation of new models.

   The process of fitting a model to a dataset is known as *training*. When the model has been trained, the data is transformed into an abstract form that summarizes the original information.

3. ***Generalization***

   The third component of the learning process is known as *generalisation.*

   The term generalization describes the process of turning the knowledge about stored data into a form that can be utilized for future action. These actions are to be carried out on tasks that are similar, but not identical, to those what have been seen before. In generalization, the goal is to discover those properties of the data that will be most

relevant to future tasks.

4. ***Evaluation***

   Evaluation is the last component of the learning process.

   It is the process of giving feedback to the user to measure the utility of the learned knowledge. This feedback is then utilised to effect improvements in the whole learning process.

Applications of Machine Learning

- Email spam detection
- Face detection and matching (e.g., iPhone X)
- Web search (e.g., DuckDuckGo, Bing, Google)
- Sports predictions
- Post office (e.g., sorting letters by zip codes)
- ATMs (e.g., reading checks)
- Credit card fraud
- Stock predictions
- Smart assistants (Apple Siri, Amazon Alexa, . . .)
- Product recommendations (e.g., Netflix, Amazon)
- Self-driving cars (e.g., Uber, Tesla)
- Language translation (Google translate)
- Sentiment analysis
- Drug design
- Medical diagnoses

Different forms of data

1. ***Numeric data***

   If a feature represents a characteristic measured in numbers, it is called a numeric feature.

2. ***Categorical or nominal***

   A categorical feature is an attribute that can take on one of a limited, and usually fixed, number of possible values on the basis of some qualitative property. A categorical feature is also called a nominal feature.

3. ***Ordinal data***

   This denotes a nominal variable with categories falling in an ordered list. Examples include clothing sizes such as small, medium, and large, or a measurement of customer satisfaction on a scale from “not at all happy” to “very happy.”

features

year	model	price	mileage	color	transmission
2011	SEL	21992	7413	Yellow	AUTO
2011	SEL	20995	10926	Gray	AUTO
2011	SEL	19995	7351	Silver	AUTO
2011	SEL	17809	11613	Gray	AUTO
2012	SE	17500	8367	White	MANUAL
2010	SEL	17495	25125	Silver	AUTO
2011	SEL	17000	27393	Blue	AUTO
2010	SEL	16995	21026	Silver	AUTO
2011	SES	16995	32655	Silver	AUTO

examples

Figure 1.2: Example for "examples" and "features" collected in a matrix format (data relates to automobiles and their features)

In the data given in Fig.1.2, the features "year", "price" and "mileage" are numeric and the features "model", "color" and "transmission" are categorical.

Different types of learning

In general, machine learning algorithms can be classified into three types.

Supervised learning

Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs.

In supervised learning, each example in the training set is a pair consisting of an input object (typically a vector) and an output value. A supervised learning algorithm analyzes the training data and produces a function, which can be used for mapping new examples. In the optimal case, the function will correctly determine the class labels for unseen instances. Both classification and regression problems are supervised learning problems.

A wide range of supervised learning algorithms are available, each with its strengths and weak- nesses. There is no single learning algorithm that works best on all supervised learning problems.

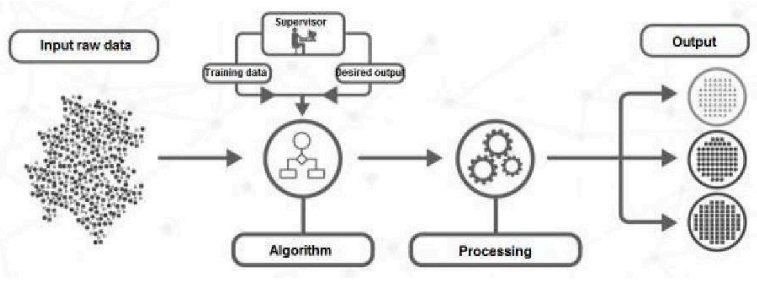


Figure 1.4: Supervised learning

Remarks

A "supervised learning" is so called because the process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process. We know the correct answers (that is, the correct outputs), the algorithm iteratively makes predictions on the training data and is corrected by the teacher. Learning stops when the algorithm achieves an acceptable level of performance.

Example

Consider the following data regarding patients entering a clinic. The data consists of the gender and age of the patients and each patient is labeled as "healthy" or "sick".

gender	age	label
M	48	sick
M	67	sick
F	53	healthy
M	49	healthy
F	34	sick
M	21	healthy

Based on this data, when a new patient enters the clinic, how can one predict whether he/she is healthy or sick?

Unsupervised learning

Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labeled responses.

In unsupervised learning algorithms, a classification or categorization is not included in the observations. There are no output values and so there is no estimation of functions. Since the examples given to the learner are unlabeled, the accuracy of the structure that is output by the algorithm cannot be evaluated.

The most common unsupervised learning method is cluster analysis, which is used for exploratory data analysis to find hidden patterns or grouping in data.

Example

Consider the following data regarding patients entering a clinic. The data consists of the gender and age of the patients.

Classification

1. Definition

In machine learning, *classification* is the problem of identifying to which of a set of categories a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known.

2. Example

Consider the following data:

Score1	29	22	10	31	17	33	32	20
Score2	43	29	47	55	18	54	40	41
Result	Pass	Fail	Fail	Pass	Fail	Pass	Pass	Pass

Table 1.1: Example data for a classification problem

Data in Table 1.1 is the training set of data. There are two attributes “Score1” and “Score2”. The class label is called “Result”. The class label has two possible values “Pass” and “Fail”. The data can be divided into two categories or classes: The set of data for which the class label is “Pass” and the set of data for which the class label is“Fail”.

Let us assume that we have no knowledge about the data other than what is given in the table. Now, the problem can be posed as follows: If we have some new data, say “Score1 = 25” and “Score2 = 36”, what value should be assigned to “Result” corresponding to the new data; in other words, to which of the two categories or classes the new observation should be assigned? See Figure
1.3 for a graphical representation of the problem.

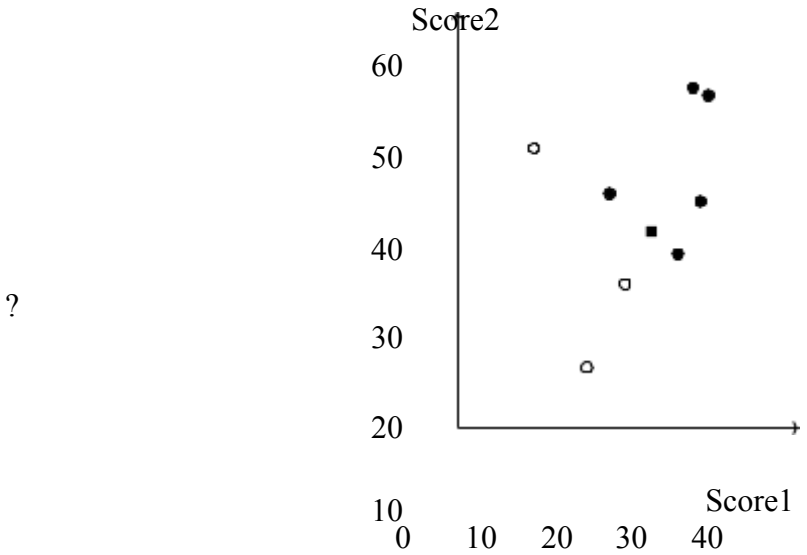


Figure 1.3: Graphical representation of data in Table 1.1. Solid dots represent data in “Pass” class and hollow dots data in “Fail” class. The class label of the square dot is to be determined.

To answer this question, using the given data alone we need to find the rule, or the formula,

or the method that has been used in assigning the values to the class label "Result". The problem of finding this rule or formula or the method is the classification problem. In general, even the general form of the rule or function or method will not be known. So several different rules, etc. may have to be tested to obtain the correct rule or function or method.

3. Real life examples

i) Optical character recognition

Optical character recognition problem, which is the problem of recognizing character codes from their images, is an example of classification problem. This is an example where there are multiple classes, as many as there are characters we would like to recognize. Especially interesting is the case when the characters are handwritten. People have different handwrit- ing styles; characters may be written small or large, slanted, with a pen or pencil, and there are many possible images corresponding to the same character.

ii) Face recognition

In the case of *face recognition*, the input is an image, the classes are people to be recognized, and the learning program should learn to associate the face images to identities. This prob- lem is more difficult than optical character recognition because there are more classes, input image is larger, and a face is three-dimensional and differences in pose and lighting cause significant changes in the image.

iii) Speech recognition

In *speech recognition*, the input is acoustic and the classes are words that can be uttered.

iv) Medical diagnosis

In *medical diagnosis*, the inputs are the relevant information we have about the patient and the classes are the illnesses. The inputs contain the patient's age, gender, past medical history, and current symptoms. Some tests may not have been applied to the patient, and thus these inputs would be missing.

v) Knowledge extraction

Classification rules can also be used for knowledge extraction. The rule is a simple model that explains the data, and looking at this model we have an explanation about the process underlying the data.

vi) Compression

Classification rules can be used for *compression*. By fitting a rule to the data, we get an explanation that is simpler than the data, requiring less memory to store and less computation to process.

vii) More examples

Here are some further examples of classification problems.

(a) An emergency room in a hospital measures 17 variables like blood pressure, age, etc. of newly admitted patients. A decision has to be made whether to put the patient in an ICU. Due to the high cost of ICU, only patients who may survive a month or more

are given higher priority. Such patients are labeled as “low-risk patients” and others are labeled “high-risk patients”. The problem is to device a rule to classify a patient as a “low-risk patient” or a “high-risk patient”.

(b) A credit card company receives hundreds of thousands of applications for new cards. The applications contain information regarding several attributes like annual salary, age, etc. The problem is to devise a rule to classify the applicants to those who are credit-worthy, who are not credit-worthy or to those who require further analysis.

(c) Astronomers have been cataloguing distant objects in the sky using digital images cre- ated using special devices. The objects are to be labeled as star, galaxy, nebula, etc. The data is highly noisy and are very faint. The problem is to device a rule using which a distant object can be correctly labeled.

Regression

1. Definition

In machine learning, a *regression problem* is the problem of predicting the value of a numeric vari- able based on observed values of the variable. The value of the output variable may be a number, such as an integer or a floating point value. These are often quantities, such as amounts and sizes. The input variables may be discrete or real-valued.

2. Example

Consider the data on car prices given in Table 1.2.

Price (US$)	Age (years)	Distance (KM)	Weight (pounds)
13500	23	46986	1165
13750	23	72937	1165
13950	24	41711	1165
14950	26	48000	1165
13750	30	38500	1170
12950	32	61000	1170
16900	27	94612	1245
18600	30	75889	1245
21500	27	19700	1185
1295	23	71138	1105

Table 1.2: Prices of used cars: example data for regression

Suppose we are required to estimate the price of a car aged 25 years with distance 53240 KM and weight 1200 pounds. This is an example of a regression problem beause we have to predict the value of the numeric variable “Price”.

3. General approach

Let x denote the set of input variables and y the output variable. In machine learning, the general approach to regression is to assume a model, that is, some mathematical relation between x and y, involving some parameters say, θ, in the following form:

$$y = f(x, \theta)$$

The function $f(x, \theta)$ is called the *regression function*. The machine learning algorithm optimizes the parameters in the set θ such that the approximation error is minimized; that is, the estimates of the values of the dependent variable y are as close as possible to the correct values given in the training set.

4. Different regression models

There are various types of regression techniques available to make predictions. These techniques mostly differ in three aspects, namely, the number and type of independent variables, the type of dependent variables and the shape of regression line. Some of these are listed below.

- *Simple linear regression*: There is only one continuous independent variable x and the assumed relation between the independent variable and the dependent variable y is

$$y = a + bx.$$

- *Multivariate linear regression*: There are more than one independent variable, say $x_1, \ldots, x_n$, and the assumed relation between the independent variables and the dependent variable is

$$y = a_0 + a_1x_1 + \cdots + a_nx_n.$$

- *Polynomial regression*: There is only one continuous independent variable x and the assumed model is

$$y = a_0 + a_1x + \cdots + a_nx^n.$$

- *Logistic regression*: The dependent variable is binary, that is, a variable which takes only the values 0 and 1. The assumed model involves certain probability distributions.

Distance Based Method

Nearest Neighbour Method

Nearest neighbor algorithms are among the "simplest" supervised machine learning algorithms and have been well studied in the field of pattern recognition over the last century. While nearest neighbor algorithms are not as popular as they once were, they are still widely used in practice, and I highly recommend that you are at least considering the k-Nearest Neighbor algorithm in classification projects as a predictive performance benchmark when you are trying to develop more sophisticated models.

NN is just a special case of k NN, where k = 1. To avoid making this text unnecessarily convoluted, we will only use the abbreviation NN if we talk about concepts that do not apply to k NN in general. Otherwise, we will use k NN to refer to nearest neighbor algorithms in general, regardless of the value of k.

KNN

k NN is an algorithm for supervised learning that simply stores the labeled training examples,during the training phase. For this reason, k NN is also called a lazy learning algorithm.

What it means to be a lazy learning algorithm is that the processing of the training examples is postponed until making predictions again, the training consists literally of just storing the training data.

Then, to make a prediction (class label or continuous target), the kNN algorithms find the k nearest neighbors of a query point and compute the class label (classification) or continuous target (regression) based on the k nearest (most "similar") points. The exact mechanics will be explained in the next sections. However, the overall idea is that instead of approximating the target function f (x) = y globally, during each prediction, k NN approximates the target function locally. In practice, it is easier to learn to approximate a function locally than globally.

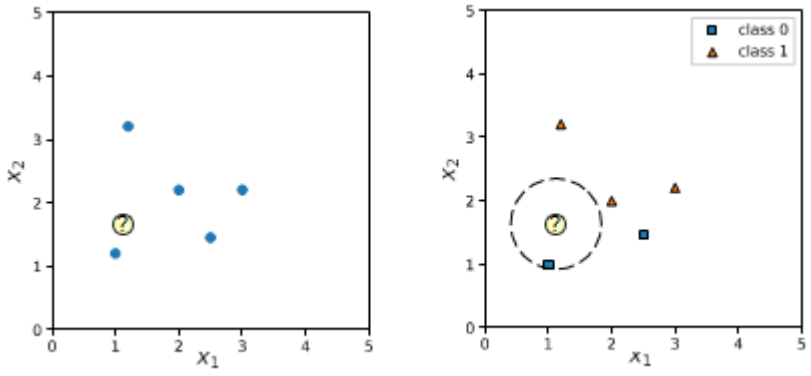


Figure 1: Illustration of the nearest neighbor classification algorithm in two dimensions (features x 1 and x 2). In the left subpanel, the training examples are shown as blue dots, and a query point that we want to classify is shown as a question mark. In the right subpanel, the class labels are, and the dashed line indicates the nearest neighbor of the query point, assuming a Euclidean distance metric. The predicted class label is the class label of the closest data point in the training set (here: class 0).

Further, instead of devising one global model or approximation of the target function, for each different data point, there is a different local approximation, which depends on the data point itself as well as the training data points. Since the prediction is based on a comparison of a query point with data points in the training set (rather than a global model), *k* NN is also categorized as **instance-based** (or "memory-based") method.

Common Use Cases of k NN

While neural networks are gaining popularity in the computer vision and pattern recognition field, one area where *k* -nearest neighbors models are still commonly and successfully being used is in the intersection between computer vision, pattern classification, and biometrics

k -Nearest Neighbor Classification and Regression

Previously, we described the NN algorithm, which makes a prediction by assigning the class label or continuous target value of the most similar training example to the query point (where similarity is typically measured using the Euclidean distance metric for continuous features).

Instead of basing the prediction of the single, most similar training example, k NN considers the k nearest neighbors when predicting a class label (in classification) or a continuous target value (in regression).

Classification

In the classification setting, the simplest incarnation of the k NN model is to predict the target class label as the class label that is most often represented among the k most similar training examples for a given query point. In other words, the class label can be considered as the "mode" of the k training labels or the outcome of a "plurality voting." Note that in literature, k NN classification is often described as a "majority voting." While the authors usually mean the right thing, the term "majority voting" is a bit unfortunate as it typically refers to a reference value of >50% for making a decision. In the case of binary predictions (classification problems with two classes), there is always a majority or a tie. Hence, a majority vote is also automatically a plurality vote. However, in multi-class settings, we do not require a majority to make a prediction via k NN. For example, in a three-class setting a frequency > 1 (approx 33.3%) could already enough to assign a class label.

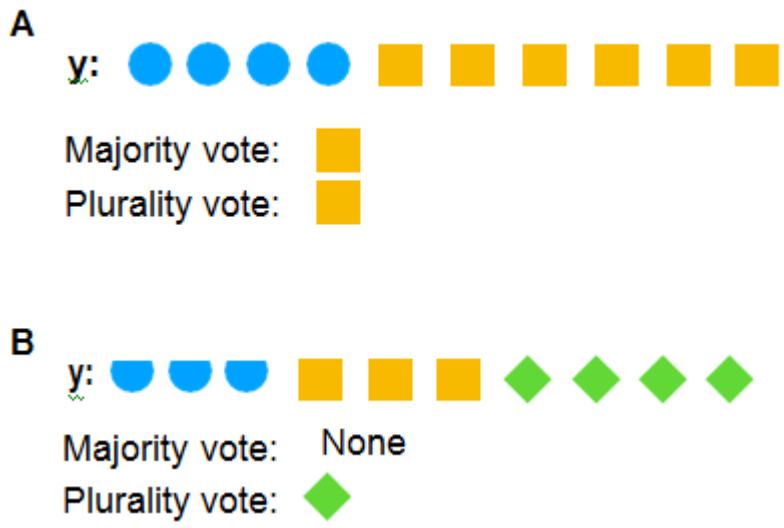


Figure 4: Illustration of plurality and majority voting.

Example Problem based on KNN

We have a survey for loan defaulters. As per the Age loan amount will be sanctioned.

Age	Loan	Default
25	$40,000	N
35	$60,000	N
45	$80,000	N
20	$20,000	N
35	$120,000	N
52	$18,000	N
23	$95,000	Y
40	$62,000	Y
60	$100,000	Y
48	$220,000	Y
33	$150,000	Y

If a new person would like to have loan with age 48 for an amount 1,42,000. Implement the KNN check the class of the person.

Working of KNN Algorithm

K-nearest neighbors (KNN) algorithm uses 'feature similarity' to predict the values of new datapoints which further means that the new data point will be assigned a value based on how closely it matches the points in the training set. We can understand its working with the help of following steps −

Step 1 − For implementing any algorithm, we need dataset. So during the first step of KNN, we must load the training as well as test data.

Step 2 − Next, we need to choose the value of K i.e. the nearest data points. K can be any integer.

Step 3 − For each point in the test data do the following −

- **3.1** − Calculate the distance between test data and each row of training data with the help of any of the method namely: Euclidean, Manhattan or Hamming distance. The most commonly used method to calculate distance is Euclidean.
- **3.2** − Now, based on the distance value, sort them in ascending order.
- **3.3** − Next, it will choose the top K rows from the sorted array.
- **3.4** − Now, it will assign a class to the test point based on most frequent class of these rows.

Step 4 – End

In this problem we are using Euclidean distance to find the distance from the tuples.

Age	Loan	Default	Distance
25	$40,000	N	102000
35	$60,000	N	82000
45	$80,000	N	62000
20	$20,000	N	122000
35	$120,000	N	22000
52	$18,000	N	124000
23	$95,000	Y	47000
40	$62,000	Y	80000
60	$100,000	Y	42000
48	$220,000	Y	78000
33	$150,000	Y	8000
48	$142,000	?	

$$D_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

sqrt [(48-25)²+(142000-40000)²]=102000.0025

Here we are using 3-NN , as per the majority from the table the class of the person is Yes.

Decision Trees:

Decision tree algorithms can be considered as an iterative, top-down construction of the hypothesis; picture a hierarchy of decision, forking the dataset into subspaces.
_ Can represent any Boolean (binary) function, and the hypothesis space being searched is the entire space of Boolean functions 1; however, we need to keep in mind that a critical challenge in machine learning is whether an algorithm can learn/_nd the \right" function or a good approximation.
_ Considering only binary (or Boolean) features, at each node, there 2m potential splits to be evaluated given m features.
_ Hypothesis space is searched greedily (i.e., decision tree algorithms perform a greedy search2 over all possible trees); an exhaustive search is not feasible because of its exponential nature of the problem (2m potential splits to evaluate given m features).

Terminology
_ Root node: no incoming edge, zero or more outgoing edges.
_ Internal node: one incoming edge, two (or more) outgoing edges.
_ Leaf node: each leaf node is an assigned a class label (if nodes are pure; otherwise majority vote).

_ Parent and child nodes: If a node is split, we refer to that given node as the parent node, and the resulting nodes are called child nodes, respectively.

Expressiveness of decision trees

Decision trees can represent any boolean function of the input attributes. Let's use decision trees to perform the function of three boolean gates AND, OR and XOR.

Boolean Function: AND

A	B	A AND B
F	F	F
F	T	F
T	F	F
T	T	T

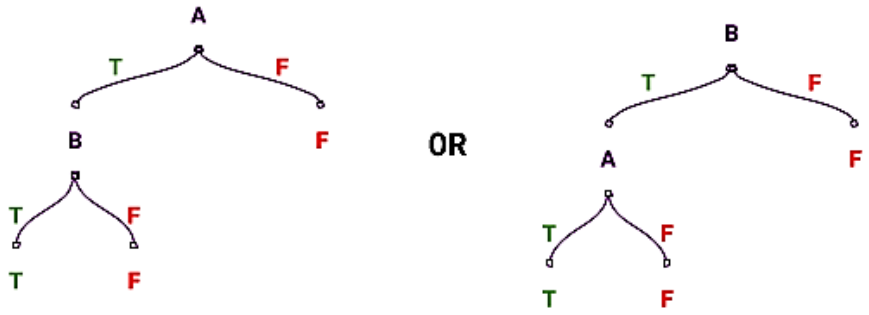


Fig 3. Decision tree for an AND operation.

In Fig 3., we can see that there are two candidate concepts for producing the decision tree that performs the AND operation. Similarly, we can also produce a decision tree that performs the boolean OR operation.

Boolean Function: OR

A	B	A OR B
F	F	F
F	T	T
T	F	T
T	T	T

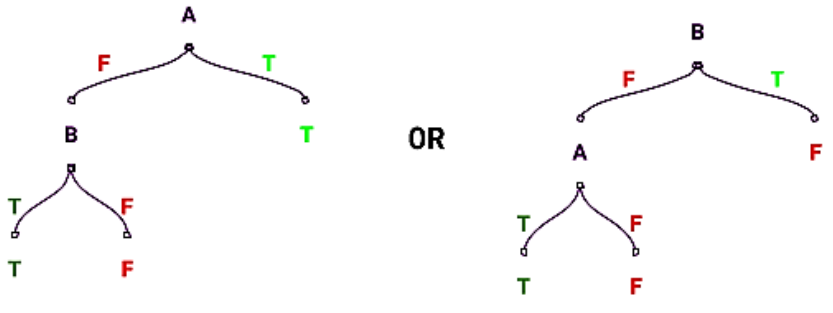


Fig 4. Decision tree for an OR operation

Boolean Function: XOR

A	B	A XOR B
F	F	F
F	T	T
T	F	T
T	T	F

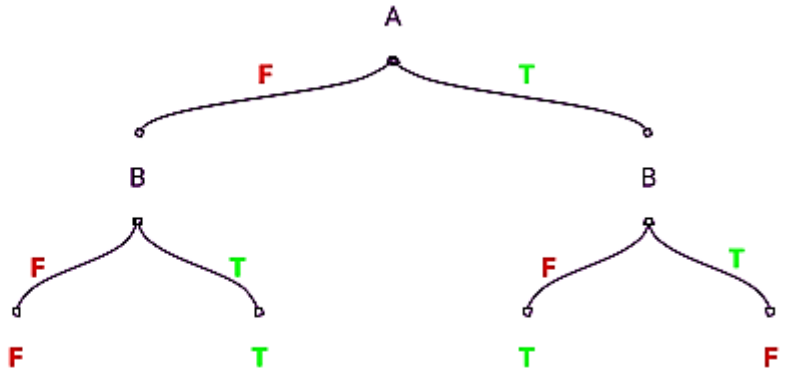


Fig 5. Decision tree for an XOR operation.

Relationship Between Decision Trees and Rule-based Learning

Intuitively, we can also think of decision tree as nested \if-else" rules. And a rule is simply a conjunction of conditions. For example,

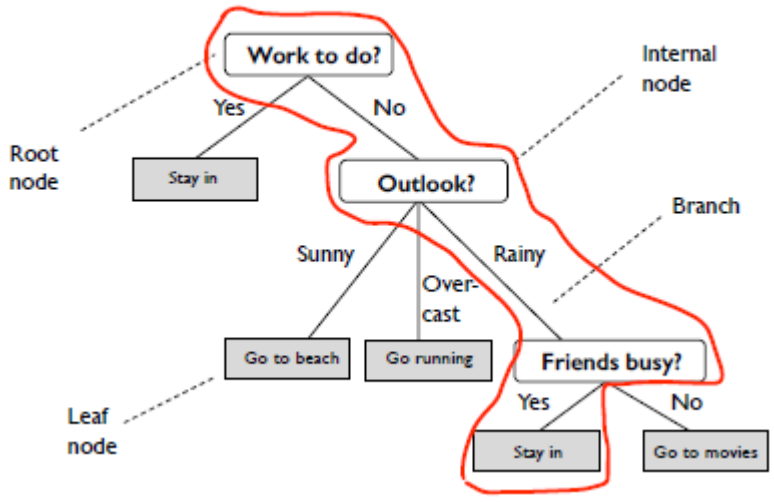
Rule 1 = (if x = 1) and (if y = 2) and…. (1)

Multiple rules can then be joined into a set of rules, which can be applied to predict the target value of a training example or test instance. For example,

Class 1 = if (Rule 1=True) or (Rule 2=True) or……. (2)

Each leaf node in a decision tree represents such a set of rules as illustrated in the following _gure, which depicts the rule,

(Work to do? = False) and(Outlook? = Rainy)and (Friends busy? = Yes)…. (3)



A rule for a given leaf node (circled): (work to do? = False) and (outlook? = Rainy) and (friends busy? = Yes)

Decision Tree Construction steps

1) Pick the feature that when parent node is split, it results in the largest information gain
2) Stop if child nodes are pure or no improvement in class purity can be made

3) Go back to step 1 for each of the two child nodes

Tree Based models

There are various simple, but widely used, models that work by partitioning the input space into cuboid regions, whose edges are aligned with the axes, and then assigning a simple model (for example, a constant) to each region. They can be viewed as a model combination method in which only one model is responsible for making predictions at any given point in input space. The process of selecting a specific model, given a new input **x**, can be described by a sequential decision making process corresponding to the traversal of a binary tree (one that splits into two branches at each node). Here we focus on a particular tree-based framework called *classification and regression trees*, or *CART* (Breiman *et al.*, 1984), although there are many other variants going by such names as ID3 and C4.5 (Quinlan, 1986; Quinlan, 1993).

Figure 14.5 shows an illustration of a recursive binary partitioning of the input space, along with the corresponding tree structure. In this example, the first step divides the whole of the input space into two regions according to whether $x1 \_ \theta1$ or $x1 > \theta1$ where $\theta1$ is a parameter of the model. This creates two subregions, each of which can then be subdivided independently. For instance, the region $x1 \_ \theta1$ is further subdivided according to whether $x2 \_ \theta2$ or $x2 > \theta2$, giving rise to the regions denoted A and B. The recursive subdivision can be described by the traversal of the binary tree shown in Figure 14.6. For any new input **x**, we determine which region it falls into by starting at the top of the tree at the root node and following a path down to a specific leaf node according to the decision criteria at each node.

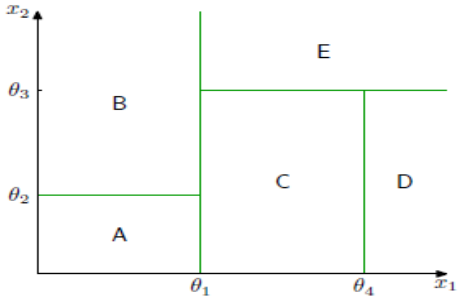


Figure 14.5 Illustration of a two-dimensional input space that has been partitioned into five regions using axis-aligned boundaries.

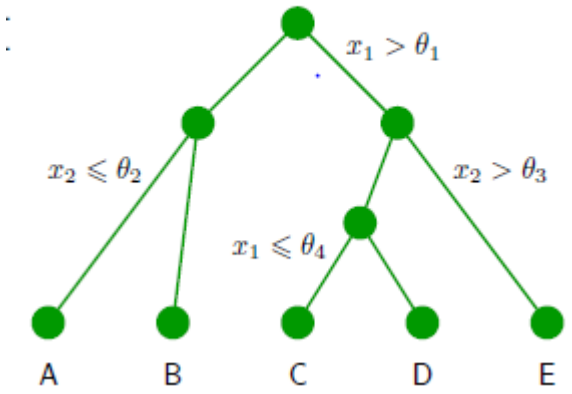


Figure 14.6 Binary tree corresponding to the partitioning of input space shown in Figure 14.5.

Within each region, there is a separate model to predict the target variable. For instance, in regression we might simply predict a constant over each region, or in classification we might assign each region to a specific class. A key property of tree based models, which makes them popular in fields such as medical diagnosis, for example, is that they are readily interpretable by humans because they correspond to a sequence of binary decisions applied to the individual input variables. For instance, to predict a patient's disease, we might first ask "is their temperature greater than some threshold?". If the answer is yes, then we might next ask "is their blood pressure less than some threshold?". Each leaf of the tree is then associated with a specific diagnosis. In order to learn such a model from a training set, we have to determine the structure of the tree, including which input variable is chosen at each node to form the split criterion as well as the value of the threshold parameter θi for the split. We also have to determine the values of the predictive variable within each region.

1. Information Gain

When we use a node in a decision tree to partition the training instances into smaller subsets the entropy changes. Information gain is a measure of this change in entropy.

Definition: Suppose S is a set of instances, A is an attribute, S_v is the subset of S with A = v, and Values (A) is the set of all possible values of A, then

$$Gain(S, A) = Entropy(S) - \sum_{vcValues(A)} \frac{|S_v|}{|S|}.Entropy(S_v)$$

Entropy

Entropy is the measure of uncertainty of a random variable, it characterizes the impurity of an arbitrary collection of examples. The higher the entropy more the information content.

Definition: Suppose S is a set of instances, A is an attribute, S_v is the subset of S with A = v, and Values (A) is the set of all possible values of A, then

$$Gain(S, A) = Entropy(S) - \sum_{vcValues(A)} \frac{|S_v|}{|S|}.Entropy(S_v)$$

Example:

For the set X = {a,a,a,b,b,b,b,b}

Total intances: 8

Instances of b: 5

Instances of a: 3

$$Entropy H(X) = -\left[\left(\frac{3}{8}\right) log_2 \frac{3}{8} + \left(\frac{5}{8}\right) log_2 \frac{5}{8}\right]$$

= -[0.375 * (-1.415) + 0.625 * (-0.678)]

=-(-0.53-0.424)

= 0.954

Pros and Cons of Decision Trees

Listed below are some of the pros and cons of using decision trees as a predictive model.

_ (+) Easy to interpret and communicate
_ (+) Independent of feature scaling
_ (-) Easy to overfit
_ (-) Elaborate pruning required
_ (-) Expensive to just fit a “diagonal line"
_ (-) Output range is bounded (dep. on training examples) in regression trees

Naïve Bayes Classification

The Bayesian classifier is an algorithm for classifying multiclass datasets. This is based on the Bayes’ theorem in probability theory. Bayes in whose name the theorem is known was an English statistician who was known for having formulated a specific case of a theorem that bears his name. The classifier is also known as “naive Bayes Algorithm” where the word “naive” is an English word
with the following meanings: simple, unsophisticated, or primitive.

Conditional probability

The probability of the occurrence of an event A given that an event B has already occurred is called the conditional probability of A given B and is denoted by P(ASB). We have

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \quad \text{if} \quad P(B) \neq 0.$$

Bayes’ theorem

Let A and B any two events in a random experiment. If P(A) != 0, then

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

Naive Bayes algorithm

Basic idea

Suppose we have a training data set consisting of N examples having n features. Let the features be named as (F1; : : : ;Fn). A feature vector is of the form (f1; f2; : : : ; fn). Associated with each example, there is a certain class label. Let the set of class labels be {c1; c2; : : : ; cp}.
Suppose we are given a test instance having the feature vector
X = (x1; x2; : : : ; xn):
We are required to determine the most appropriate class label that should be assigned to the test instance. For this purpose we compute the following conditional probabilities
P(c1/X);P(c2/X); : : : ;P(cp/X):
and choose the maximum among them. Let the maximum probability be P(ciSX). Then, we choose ci as the most appropriate class label for the training instance having X as the feature vector.

Algorithm: Naive Bayes

Let there be a training data set having n features $F_1, \ldots, F_n$. Let f_1 denote an arbitrary value of F_1, f_2 of F_2, and so on. Let the set of class labels be $\{c_1, c_2, \ldots, c_p\}$. Let there be given a test instance having the feature vector

$$X = (x_1, x_2, \ldots, x_n).$$

We are required to determine the most appropriate class label that should be assigned to the test instance.

Step 1. Compute the probabilities $P(c_k)$ for $k = 1, \ldots, p$.

Step 2. Form a table showing the conditional probabilities

$$P(f_1|c_k), \quad P(f_2|c_k), \quad \ldots \quad , P(f_n|c_k)$$

for all values of $f_1, f_2, \ldots, f_n$ and for $k = 1, \ldots, p$.

Step 3. Compute the products

$$q_k = P(x_1|c_k)P(x_2|c_k)\cdots P(x_n|c_k)P(c_k)$$

for $k = 1, \ldots, p$.

Step 4. Find j such $q_j = \max\{q_1, q_2, \ldots, q_p\}$.

Step 5. Assign the class label c_j to the test instance X.

Example Problem
Consider a training data set consisting of the fauna of the world. Each unit has three features named "Swim", "Fly" and "Crawl". Let the possible values of these features be as follows:

Swim Fast, Slow, No
Fly Long, Short, Rarely, No
Crawl Yes, No
For simplicity, each unit is classified as "Animal", "Bird" or "Fish". Let the training data set be as in Table Use naive Bayes algorithm to classify a particular species if its features are (Slow, Rarely, No)?

Sl. No.	Swim	Fly	Crawl	Class
1	Fast	No	No	Fish
2	Fast	No	Yes	Animal
3	Slow	No	No	Animal
4	Fast	No	No	Animal
5	No	Short	No	Bird
6	No	Short	No	Bird
7	No	Rarely	No	Animal
8	Slow	No	Yes	Animal
9	Slow	No	No	Fish
10	Slow	No	Yes	Fish
11	No	Long	No	Bird
12	Fast	No	No	Bird

Solution
In this example, the features are

F1 = “Swim”; F2 = “Fly”; F3 = “Crawl”:

The class labels are

c1 = “Animal”; c2 = “ Bird”; c3 = “Fish”:

The test instance is (Slow, Rarely, No) and so we have:
x1 = “Slow”; x2 = “Rarely”; x3 = “No”:
We construct the frequency table shown in Table 6.2 which summarises the data. (It may be noted that the construction of the frequency table is not part of the algorithm.)

Class	Features									Total
	Swim (F_1)			Fly (F_2)				Crawl (F_3)		
	Fast	Slow	No	Long	Short	Rarely	No	Yes	No	
Animal (c_1)	2	2	1	0	0	1	4	2	3	5
Bird (c_2)	1	0	3	1	2	0	1	1	3	4
Fish (c_3)	1	2	0	0	0	0	3	0	3	3
Total	4	4	4	1	2	1	8	4	8	12

Step 1. We compute following probabilities.

$$P(c_1) = \frac{\text{No. of records with class label "Animal"}}{\text{Total number of examples}}$$
$$= 5/12$$
$$P(c_2) = \frac{\text{No. of records with class label "Bird"}}{\text{Total number of examples}}$$
$$= 4/12$$
$$P(c_3) = \frac{\text{No of records with class label "Fish"}}{\text{Total number of examples}}$$
$$= 3/12$$

Step 2. We construct the following table of conditional probabilities:

Class	Features								
	Swim (F_1) f_1			Fly (F_2) f_2				Crawl (F_3) f_3	
	Fast	Slow	No	Long	Short	Rarely	No	Yes	No
Animal (c_1)	2/5	2/5	1/5	0/5	0/5	1/5	4/5	2/5	3/5
Bird (c_2)	1/4	0/4	3/4	1/4	2/4	0/4	1/4	0/4	4/4
Fish (c_3)	1 3	2/3	0/3	0/3	0/3	0/3	3/3	0/3	3/3

Note: The conditional probabilities are calculated as follows:

$$P((F_1 = \text{Slow})|c_1) = \frac{\text{No. of records with } F_1 = \text{Slow and class label } c_1}{\text{No. of records with class label } c_1}$$
$$= 2/5.$$

Step 3. We now calculate the following numbers:

$$
\begin{aligned}
q_1 &= P(x_1|c_1)P(x_2|c_1)P(x_3|c_1)P(c_1) \\
&= (2/5) \times (1/5) \times (3/5) \times (5/12) \\
&= 0.02 \\
q_2 &= P(x_1|c_2)P(x_2|c_2)P(x_3|c_2)P(c_2) \\
&= (0/4) \times (0/4) \times (3/4) \times (4/12) \\
&= 0 \\
q_3 &= P(x_1|c_3)P(x_2|c_3)P(x_3|c_3)P(c_3) \\
&= (2/3) \times (0/3) \times (3/3) \times (3/12) \\
&= 0
\end{aligned}
$$

Step 4. Now max{q1; q2; q3} = 0:02
Step 5. The maximum is q1 an it corresponds to the class label c1 = " Animal"

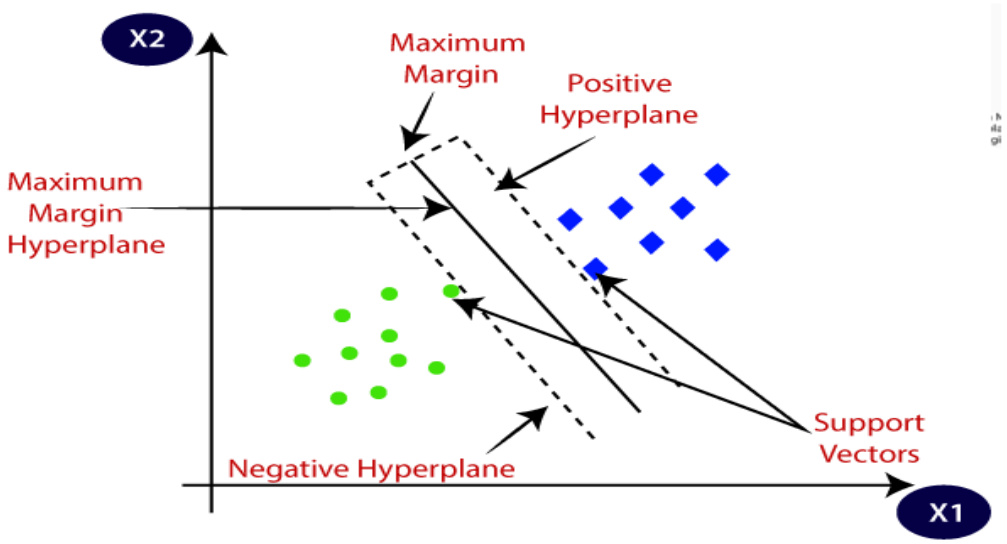
So we assign the class label "Animal" to the test instance "(Slow, Rarely, No)".

Support Vector Machine(SVM)

Support Vector Machine or 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 Machine Learning.

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 data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:



Types of SVM

SVM can be of two types:

- **Linear SVM:** Linear SVM 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 called as Linear SVM classifier.
- **Non-linear SVM:** Non-Linear SVM 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.

SVM Terminology:

The followings are important concepts in SVM −

- **Support Vectors** − Datapoints that are closest to the hyperplane is called support vectors. Separating line will be defined with the help of these data points.
- **Hyperplane** − As we can see in the above diagram, it is a decision plane or space which is divided between a set of objects having different classes.
- **Margin** − It may be defined as the gap between two lines on the closet data points of different classes. It can be calculated as the perpendicular distance from the line to the support vectors. Large margin is considered as a good margin and small margin is considered as a bad margin.

SVM Kernels

SVM algorithm is implemented with kernel that transforms an input data space into the required form. SVM uses a technique called the kernel trick in which kernel takes a low dimensional input space and transforms it into a higher dimensional space. In simple words, kernel converts non-separable problems into separable problems by adding more dimensions to it. It makes SVM

more powerful, flexible and accurate. The following are some of the types of kernels used by SVM.

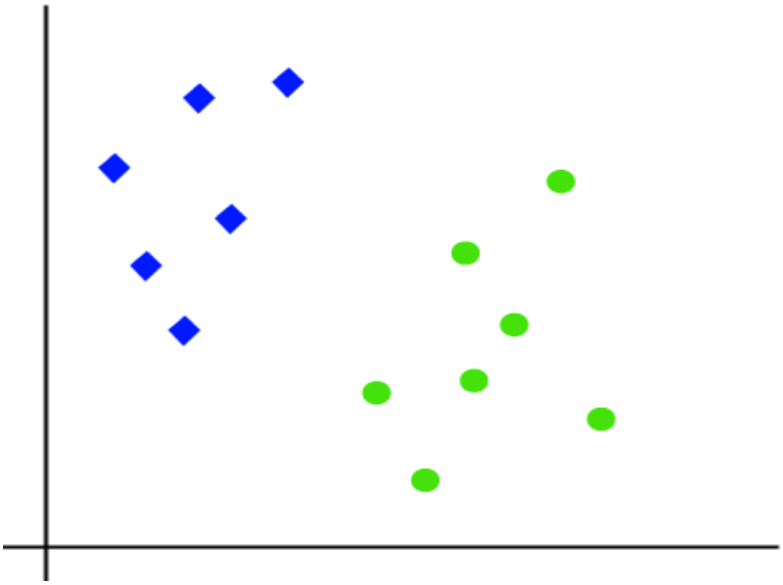
Linear Kernel

It can be used as a dot product between any two observations. The formula of linear kernel is as below −

K(x,xi)=sum(x∗xi)

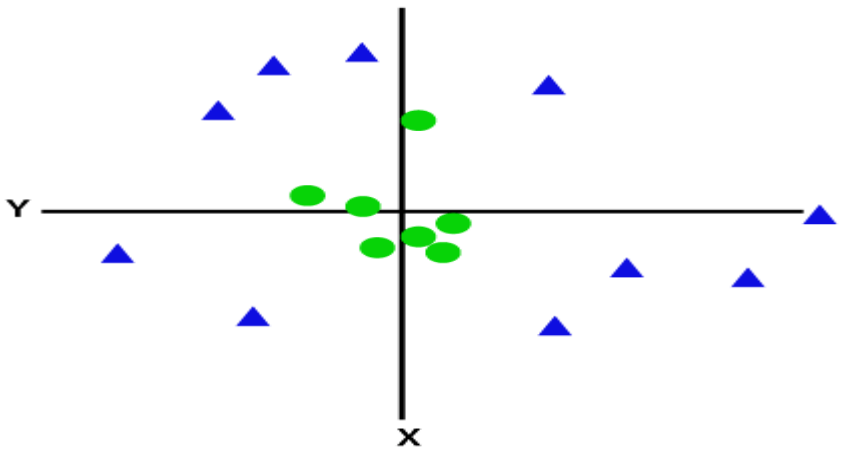
From the above formula, we can see that the product between two vectors say x & xi is the sum of the multiplication of each pair of input values.

- The working of the SVM algorithm can be understood by using an example. Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x1 and x2. We want a classifier that can classify the pair(x1, x2) of coordinates in either green or blue. Consider the below image:



So as it is 2-d space so by just using a straight line, we can easily separate these two classes.

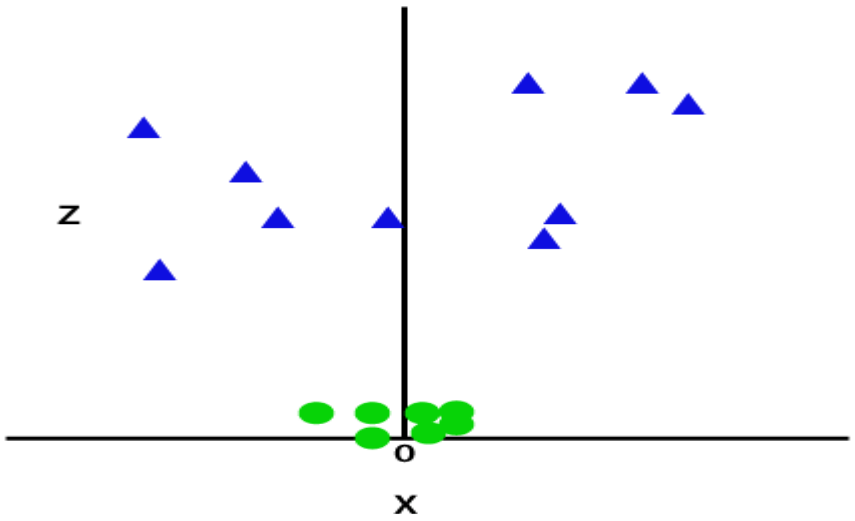
Non-Linear SVM: If data is linearly arranged, then we can separate it by using a straight line, but for non-linear data, we cannot draw a single straight line. Consider the below image:



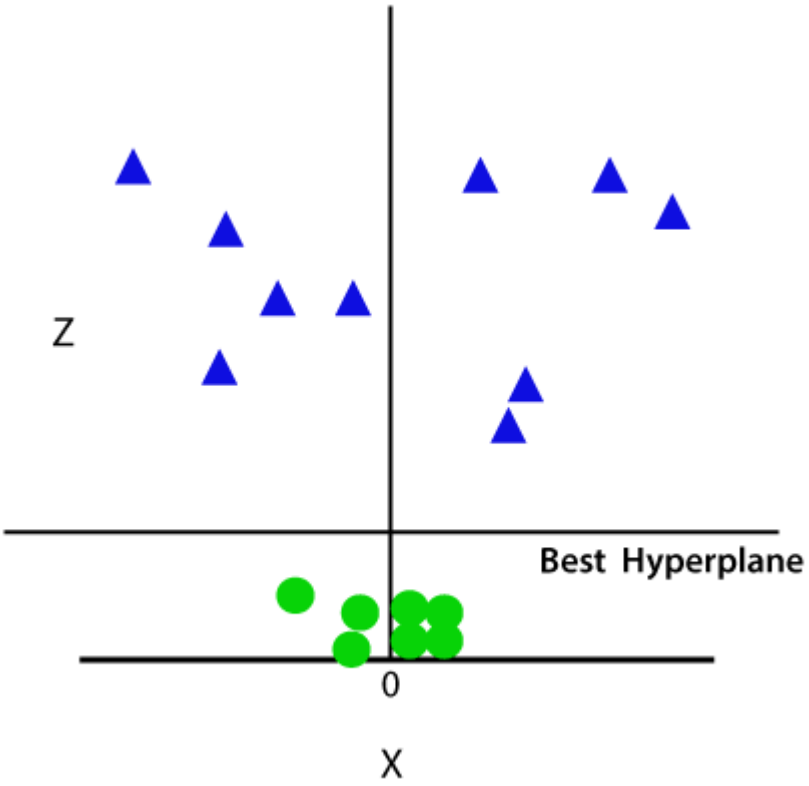
So to separate these data points, we need to add one more dimension. For linear data, we have used two dimensions x and y, so for non-linear data, we will add a third dimension z. It can be calculated as:

$z=x^2 +y^2$

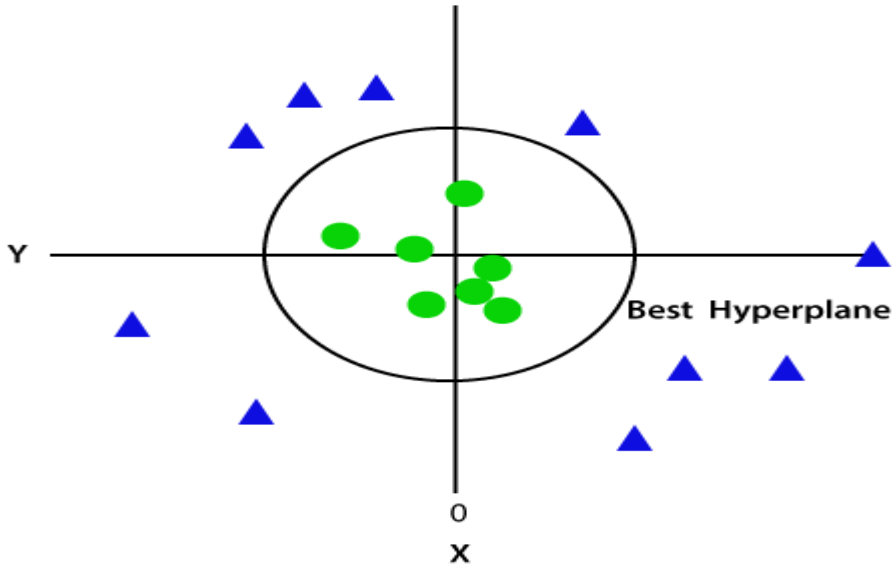
By adding the third dimension, the sample space will become as below image:



So now, SVM will divide the datasets into classes in the following way. Consider the below image:



Since we are in 3-d Space, hence it is looking like a plane parallel to the x-axis. If we convert it in 2d space with z=1, then it will become as:



Hence we get a circumference of radius 1 in case of non-linear data.

We will use other kernel models as follows

Polynomial Kernel

It is more generalized form of linear kernel and distinguish curved or nonlinear input space. Following is the formula for polynomial kernel −

$$k(X,Xi)=1+sum(X*Xi)^dk(X,Xi)=1+sum(X*Xi)^d$$

Here d is the degree of polynomial, which we need to specify manually in the learning algorithm.

Radial Basis Function (RBF) Kernel

RBF kernel, mostly used in SVM classification, maps input space in indefinite dimensional space. Following formula explains it mathematically −

$$K(x,xi)=exp(−gamma*sum(x−xi^2))K(x,xi)=exp(−gamma*sum(x−xi^2))$$

Here, *gamma* ranges from 0 to 1. We need to manually specify it in the learning algorithm. A good default value of *gamma* is 0.1.

Binary/Multi class Classification

In machine learning, the multiclass classification is the problem of classifying instances into one of three or more classes. Classifying instances into one of the two classes is called binary classification.

- A problem with two classes is often called a two-class or binary classification problem.

In a two-class data set, the set of values of the target variable may be {“yes”, “no”}, or {“TRUE”, ”FALSE”}, or {0, 1}, or {−1,+1} or any such similar set.

- A problem with more than two classes is often called a multi-class classification problem.

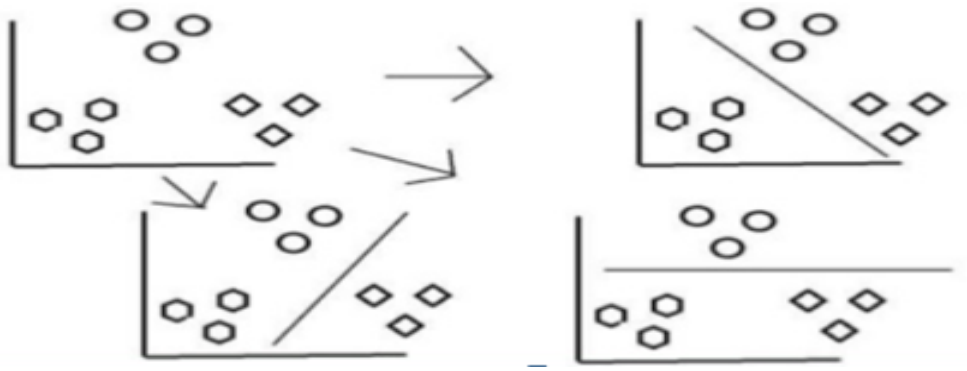
In machine learning, the multiclass classification is the problem of classifying instances into one of three or more classes.

Two methods are used to handle multi class classification

1) ”One-against-all“ or “one-vs-all”
2) “one-against-one” or “ one-vs-one”

One-against-all Method

- In this strategy we would divide the data sets such that a hypothesis separates one class label from all of the rest



one-against-one Method

In one vs one strategy we would divide the data sets into pairs and then do classification.

Note: the one hypothesis separates only two classes irrespective of other classes. If there are p different class labels, a total of $p(p-1)/2$ classifiers are constructed.

Ex: If a,b,c classes then the pairs would be ab,ac,bc 3*(3-1)/=3