UNIT I

Introduction: What does it mean to learn, Some canonical Learning Problems, The Decision Tree Model of Learning, Formalizing the Learning Problem [Reference 1], ID3 Algorithm [[Reference 2] Limits of Learning: Data Generating Distributions, Inductive Bias, Not Everything is learnable, Underfitting and Overfitting, Separation of training and test Data, Models, parameters and Hyperparameters, Real World Applications of Machine Learning [Reference 1] Geometry and Nearest Neighbors: From Data to Feature Vectors, k-Nearest Neighbors, Decision Boundaries, k-means Clustering, High Dimensions [Reference 1].

1.1 What Does it Mean to Learn

Q1. What do we mean by learning? Explain about machine learning.

Ans :

Learning is "a process that leads to change, which occurs as a result of experience and increases the potential for improved performance and future learning". The change in the learner may happen at the level of knowledge, attitude or behavior.

Machine Learning

Machine learning is the concept that a computer program can learn and adapt to new data without human intervention. Machine learning is a field of artificial intelligence (AI) that keeps a computer's built-in algorithms current regardless of changes in the worldwide economy.

Key Take a ways

- Machine learning is an area of artificial intelligence (AI) with a concept that a computer program can learn and adapt to new data without human intervention.
- A complex algorithm or source code is built into a computer that allows for the machine to identify data and build predictions around the data that it identifies.
- Machine learning is useful in parsing the immense amount of information that is consistently and readily available in the world to assist in decision making.
- Machine learning can be applied in a variety of areas, such as in investing, advertising, lending, organizing news, fraud detection, and more.

Understanding Machine Learning

Various sectors of the economy are dealing with huge amounts of data available in different formats from disparate sources. The enormous amount of data, known as big data, is becoming easily available and accessible due to the progressive use of technology, specifically advanced computing capabilities and cloud storage. Companies and governments realize the huge insights that can be gained from tapping into big data but lack the resources and time required to comb through its wealth of information. As such, artificial intelligence measures are being employed by different industries to gather, process, communicate, and share useful information from data sets. One method of AI that is increasingly utilized for big data processing is machine learning.

The various data applications of machine learning are formed through a complex algorithm or source code built into the machine or computer. This programming code creates a model that identifies the data and builds predictions around the data it identifies. The model uses parameters built in the algorithm to form patterns for its decision-making process. When new or additional data becomes available, the algorithm automatically adjusts the parameters to check for a pattern change, if any. However, the model shouldn't change.

Uses of Machine Learning

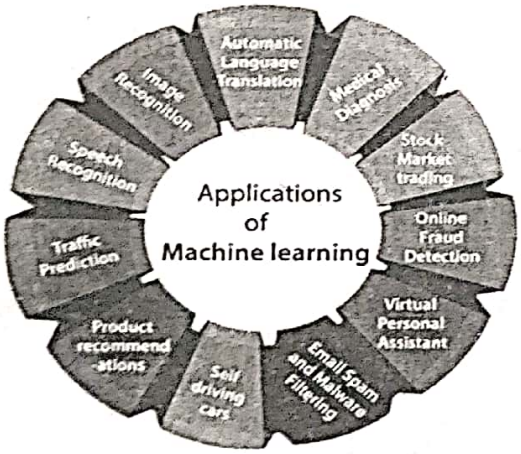
Machine learning is used in different sectors for various reasons. Trading systems can be calibrated to identify new investment opportunities.

Marketing and e-commerce platforms can be tuned to provide accurate and personalized recommendations to their users based on the users' internet search history or previous transactions. Lending institutions can incorporate machine learning to predict bad loans and build a credit risk model. Information hubs can use machine learning to cover huge amounts of news stories from all corners of the world. Banks can create fraud detection tools from machine learning techniques. The incorporation of machine learning in the digital-savvy era is endless as businesses and governments become more aware of the opportunities that big data presents.

Q2. What are the applications of Machine learning.

Ans :

Machine learning is a buzzword for today's technology, and it is growing very rapidly day by day. We are using machine learning in our daily life even without knowing it such as Google Maps, Google assistant, Alexa, etc. Below are some most trending real-world applications of Machine Learning:



1. Image Recognition

Image recognition is one of the most common applications of machine learning. It is used to identify objects, persons, places, digital images, etc. The popular use case of image recognition and face detection is, Automatic friend tagging suggestion.

Facebook provides us a feature of auto friend tagging suggestion. Whenever we upload a photo with our Facebook friends, then we automatically get a tagging suggestion with name, and the technology behind this is machine learning's face detection and recognition algorithm.

It is based on the Facebook project named "Deep Face," which is responsible for face recognition and person identification in the picture.

2. Speech Recognition

While using Google, we get an option of "Search by voice," it comes under speech recognition, and it's a popular application of machine learning.

Speech recognition is a process of converting voice instructions into text, and it is also known as "Speech to text", or "Computer speech recognition." At present, machine learning algorithms are widely used by various applications of speech recognition. Google assistant, Siri, Cortana, and Alexa are using speech recognition technology to follow the voice instructions.

3. Traffic prediction

If we want to visit a new place, we take help of Google Maps, which shows us the correct path with the shortest route and predicts the traffic conditions.

It predicts the traffic conditions such as whether traffic is cleared, slow-moving, or heavily congested with the help of two ways:

- Real Time location of the vehicle form Google Map app and sensors
- Average time has taken on past days at the same time.

Everyone who is using Google Map is helping this app to make it better. It takes information from the user and sends back to its database to improve the performance.

4. Product recommendations

Machine learning is widely used by various e-commerce and entertainment companies such as Amazon, Netflix, etc., for product recommendation to the user. Whenever we search for some product on Amazon, then

we started getting an advertisement for the same product while internet surfing on the same browser and this is because of machine learning.

Google understands the user interest using various machine learning algorithms and suggests the product as per customer interest.

As similar, when we use Netflix, we find some recommendations for entertainment series, movies, etc., and this is also done with the help of machine learning.

5. Self-driving cars

One of the most exciting applications of machine learning is self-driving cars. Machine learning plays a significant role in self-driving cars. Tesla, the most popular car manufacturing company is working on self-driving car. It is using unsupervised learning method to train the car models to detect people and objects while driving.

6. Email Spam and Malware Filtering

Whenever we receive a new email, it is filtered automatically as important, normal, and spam. We always receive an important mail in our inbox with the important symbol and spam emails in our spam box, and the technology behind this is Machine learning. Below are some spam filters used by Gmail:

- Content Filter
- Header filter
- General blacklists filter
- Rules-based filters
- Permission filters

Some machine learning algorithms such as Multi-Layer Perceptron, Decision tree, and Naïve Bayes classifier are used for email spam filtering and malware detection.

7. Virtual Personal Assistant

We have various virtual personal assistants such as Google assistant, Alexa, Cortana, Siri. As the name suggests, they help us in finding the information using our voice instruction. These assistants can help us in various ways just by our voice instructions such as Play music, call someone, Open an email, Scheduling an appointment, etc.

These virtual assistants use machine learning algorithms as an important part.

These assistant record our voice instructions, send it over the server on a cloud, and decode it using ML algorithms and act accordingly.

8. Online Fraud Detection

Machine learning is making our online transaction safe and secure by detecting fraud transaction. Whenever we perform some online transaction, there may be various ways that a fraudulent transaction can take place such as fake accounts, fake ids, and steal money in the middle of a transaction. So to detect this, Feed Forward Neural network helps us by checking whether it is a genuine transaction or a fraud transaction.

For each genuine transaction, the output is converted into some hash values, and these values become the input for the next round. For each genuine transaction, there is a specific pattern which gets change for the fraud transaction hence, it detects it and makes our online transactions more secure.

9. Stock Market trading

Machine learning is widely used in stock market trading. In the stock market, there is always a risk of up and downs in shares, so for this machine learning's long short term memory neural network is used for the prediction of stock market trends.

10. Medical Diagnosis

In medical science, machine learning is used for diseases diagnoses. With this, medical technology is growing very fast and able to build 3D models that can predict the exact position of lesions in the brain.

It helps in finding brain tumors and other brain-related diseases easily.

11. Automatic Language Translation

Nowadays, if we visit a new place and we are not aware of the language then it is not a

problem at all, as for this also machine learning helps us by converting the text into our known languages. Google's GNMT (Google Neural Machine Translation) provide this feature, which is a Neural Machine Learning that translates the text into our familiar language, and it called as automatic translation.

The technology behind the automatic translation is a sequence to sequence learning algorithm, which is used with image recognition and translates the text from one language to another language.

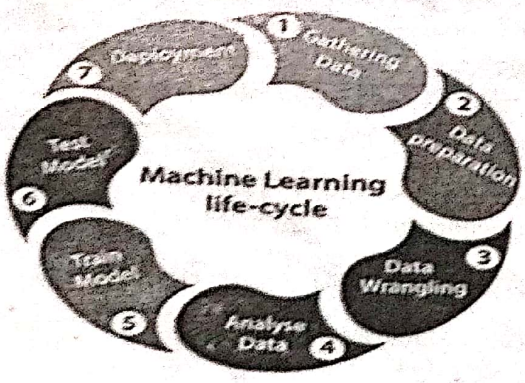
Q3. Explain about Machine learning Life cycle.

Ans : **(Imp.)**

Machine learning has given the computer systems the abilities to automatically learn without being explicitly programmed. But how does a machine learning system work? So, it can be described using the life cycle of machine learning. Machine learning life cycle is a cyclic process to build an efficient machine learning project. The main purpose of the life cycle is to find a solution to the problem or project.

Machine learning life cycle involves seven major steps, which are given below:

- Gathering Data
- Data preparation
- Data Wrangling
- Analyse Data
- Train the model
- Test the model
- Deployment



The most important thing in the complete process is to understand the problem and to know the purpose of the problem. Therefore, before starting the life cycle, we need to understand the problem because the good result depends on the better understanding of the problem.

In the complete life cycle process, to solve a problem, we create a machine learning system called "model", and this model is created by providing "training". But to train a model, we need data, hence, life cycle starts by collecting data.

1. Gathering Data

Data Gathering is the first step of the machine learning life cycle. The goal of this step is to identify and obtain all data-related problems.

In this step, we need to identify the different data sources, as data can be collected from various sources such as files, database, internet, or mobile devices. It is one of the most important steps of the life cycle. The quantity and quality of the collected data will determine the efficiency of the output. The more will be the data, the more accurate will be the prediction.

This step includes the below tasks:

- Identify various data sources
- Collect data
- Integrate the data obtained from different sources

By performing the above task, we get a coherent set of data, also called as a data set. It will be used in further steps.

2. Data preparation

After collecting the data, we need to prepare it for further steps. Data preparation is a step where we put our data into a suitable place and prepare it to use in our machine learning training.

In this step, first, we put all data together, and then randomize the ordering of data.

This step can be further divided into two processes:

- **Data exploration:** It is used to understand the nature of data that we have to work with. We need to understand the characteristics, format, and quality of data.

  A better understanding of data leads to an effective outcome. In this, we find Correlations, general trends, and outliers.

- **Data pre-processing:** Now the next step is preprocessing of data for its analysis.

3. Data Wrangling

Data wrangling is the process of cleaning and converting raw data into a useable format. It is the process of cleaning the data, selecting the variable to use, and transforming the data in a proper format to make it more suitable for analysis in the next step. It is one of the most important steps of the complete process. Cleaning of data is required to address the quality issues.

It is not necessary that data we have collected is always of our use as some of the data may not be useful. In real-world applications, collected data may have various issues, including:

- Missing Values
- Duplicate data
- Invalid data
- Noise

So, we use various filtering techniques to clean the data.

It is mandatory to detect and remove the above issues because it can negatively affect the quality of the outcome.

4. Data Analysis

Now the cleaned and prepared data is passed on to the analysis step. This step involves:

- Selection of analytical techniques
- Building models
- Review the result

The aim of this step is to build a machine learning model to analyze the data using various analytical techniques and review the outcome. It starts with the determination of the type of the problems, where we select the machine learning techniques such as Classification, Regression, Cluster analysis, Association, etc. then build the model using prepared data, and evaluate the model.

5. Train Model

Now the next step is to train the model, in this step we train our model to improve its performance for better outcome of the problem.

We use data sets to train the model using various machine learning algorithms. Training a model is required so that it can understand the various patterns, rules, and, features.

6. Test Model

Once our machine learning model has been trained on a given data set, then we test the model. In this step, we check for the accuracy of our model by providing a test data set to it.

Testing the model determines the percentage accuracy of the model as per the requirement of project or problem.

7. Deployment

The last step of machine learning life cycle is deployment, where we deploy the model in the real-world system.

If the above-prepared model is producing an accurate result as per our requirement with acceptable speed, then we deploy the model in the real system. But before deploying the project, we will check whether it is improving its performance using available data or not. The deployment phase is similar to making the final report for a project.

1.2 Some Canonical Learning Problems

Q4. Describe the types of Machine Learning Problems.

Ans :

There are common classes of problem in Machine Learning. The problem classes below are the types for most of the problems we refer to when we are doing Machine Learning.

- **Classification:** Data is labelled meaning it is assigned a class, for example spam/non-spam or fraud/non-fraud. The decision being modelled is to assign labels to new unlabelled pieces of data. This can be thought of as a discrimination problem, modelling the differences or similarities between groups.
- **Regression:** Data is labelled with a real value (think floating point) rather then a label. Examples that are easy to understand are time series data like the price of a stock over time, The decision being modelled is what value to predict for new unpredicted data.
- **Clustering:** Data is not labelled, but can be divided into groups based on similarity and other measures of natural structure in the data. An example from the above list would be organizing pictures by faces without names, where the human user has to assign names to groups, like iPhoto on the Mac.
- **Rule Extraction:** Data is used as the basis for the extraction of propositional rules (antecedent/consequent aka if-then). Such rules may, but are typically not directed, meaning that the methods discover statistically supportable relationships between attributes in the data, not necessarily involving something that is being predicted. An example is the discovery of the relationship between the purchase of beer and diapers (this is data mining folk-law, true or not, it's illustrative of the desire and opportunity).

When you think a problem is a machine learning problem (a decision problem that needs to be modelled from data), think next of what type of problem you could phrase it as easily or what type of outcome the client or requirement is asking for and work backwards.

Decision Tree Analysis is a general, predictive modelling tool that has applications spanning a number of different areas. In general, decision trees are constructed via an algorithmic approach that identifies ways to split a data set based on different conditions. It is one of the most widely used and practical methods for supervised learning. Decision Trees are a non-parametric supervised learning method used for both classification and regression tasks. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

The decision rules are generally in form of if-then-else statements. The deeper the tree, the more complex the rules and fitter the model.

Before we dive deep, let's get familiar with some of the terminologies:

- **Instances:** Refer to the vector of features or attributes that define the input space
- **Attribute:** A quantity describing an instance
- **Concept:** The function that maps input to output
- **Target Concept:** The function that we are trying to find, i.e., the actual answer
- **Hypothesis Class:** Set of all the possible functions
- **Sample:** A set of inputs paired with a label, which is the correct output (also known as the Training Set)
- **Candidate Concept:** A concept which we think is the target concept
- **Testing Set:** Similar to the training set and is used to test the candidate concept and determine its performance

1.3 The Decision Tree Model of Learning

Q5. Explain about decision tree model of learning.

Ans : **(Imp.)**

A decision tree is a tree-like graph with nodes representing the place where we pick an attribute and ask a question; edges represent the answers to the question; and the leaves represent the actual output or class label. They are used in non-linear decision making with simple linear decision surface.

Decision trees classify the examples by sorting them down the tree from the root to some leaf node, with the leaf node providing the classification to the example. Each node in the tree acts as a test case for some attribute, and each edge descending from that node corresponds to one of the possible answers to the test case. This process is recursive in nature and is repeated for every subtree rooted at the new nodes.

Let's illustrate this with help of an example. Let's assume we want to play badminton on a particular day — say Saturday — how will you decide whether to play or not. Let's say you go out and check if it's hot or cold, check the speed of the wind and humidity, how the weather is, i.e. is it sunny, cloudy, or rainy. You take all these factors into account to decide if you want to play or not.

So, you calculate all these factors for the last ten days and form a lookup table like the one below.

Day	Weather	Temperature	Humidity	Wind	Play?
1	Sunny	Hot	High	Weak	No
2	Cloudy	Hot	High	Weak	Yes
3	Sunny	Mild	Normal	Strong	Yes
4	Cloudy	Mild	High	Strong	Yes
5	Rainy	Mild	High	Strong	No
6	Rainy	Cool	Normal	Strong	No
7	Rainy	Mild	High	Weak	Yes
8	Sunny	Hot	High	Strong	No
9	Cloudy	Hot	Normal	Weak	Yes
10	Rainy	Mild	High	Strong	No

Table 1. Obeservations of the last ten days.

Now, you may use this table to decide whether to play or not. But, what if the weather pattern on Saturday does not match with any of rows in the table? This may be a problem. A decision tree would be a great way to represent data like this because it takes into account all the possible paths that can lead to the final decision by following a tree-like structure.

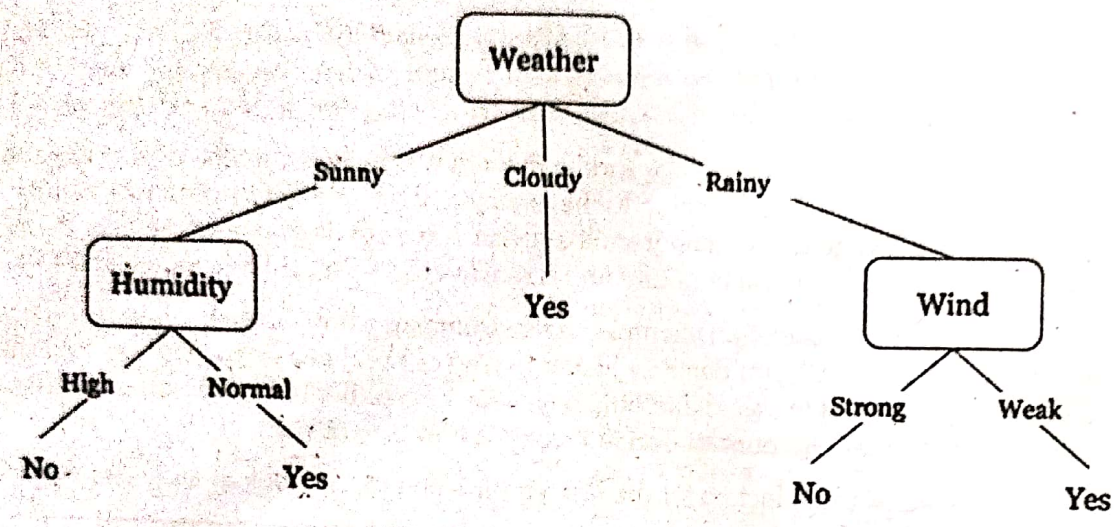


Fig.: A decision tree for the concept Play Badminton

Fig .illustrates a learned decision tree. We can see that each node represents an attribute or feature and the branch from each node represents the outcome of that node. Finally, its the leaves of the tree where the final decision is made. If features are continuous, internal nodes can test the value of a feature against a threshold (see Fig.).

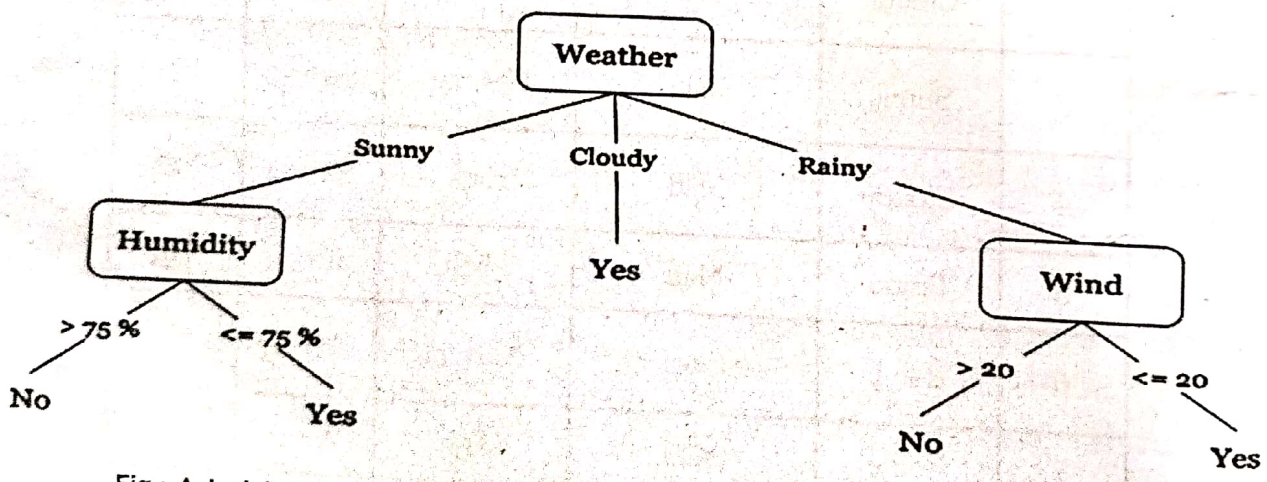


Fig.: A decision tree for the concept Play Badminton (when attributes are continuous)

A general algorithm for a decision tree can be described as follows:

1. Pick the best attribute/feature. The best attribute is one which best splits or separates the data.
2. Ask the relevant question.
3. Follow the answer path.
4. Go to step 1 until you arrive to the answer.

The best split is one which separates two different labels into two sets.

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

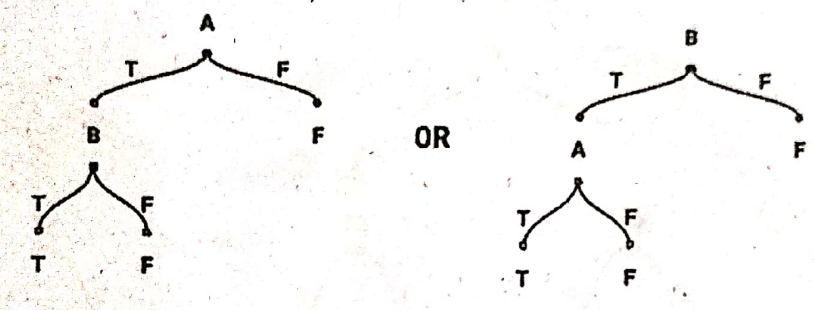


Fig.: Decision tree for an AND operation.

In Fig ., 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

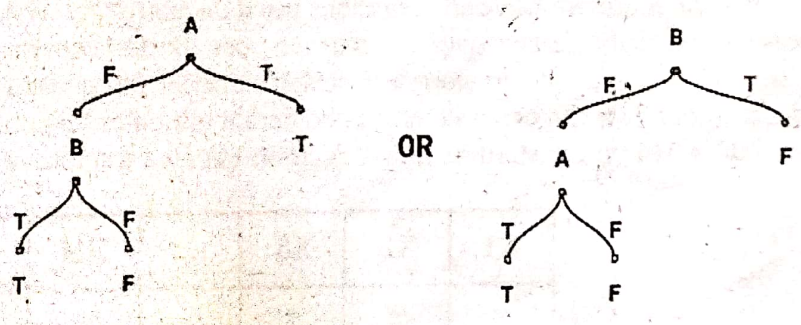


Fig.: Decision tree for an OR operation

Boolean Function: XOR

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

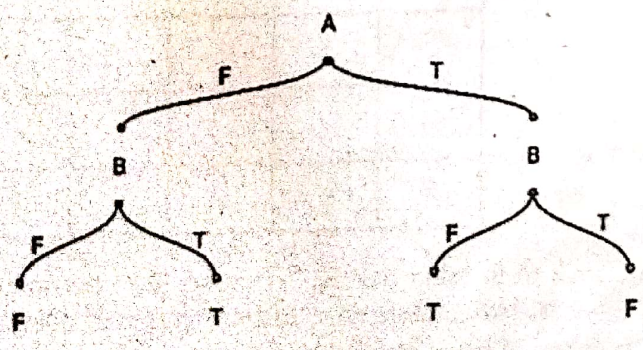


Fig.: Decision tree for an XOR operation.

Let's produce a decision tree performing XOR functionality using 3 attributes:

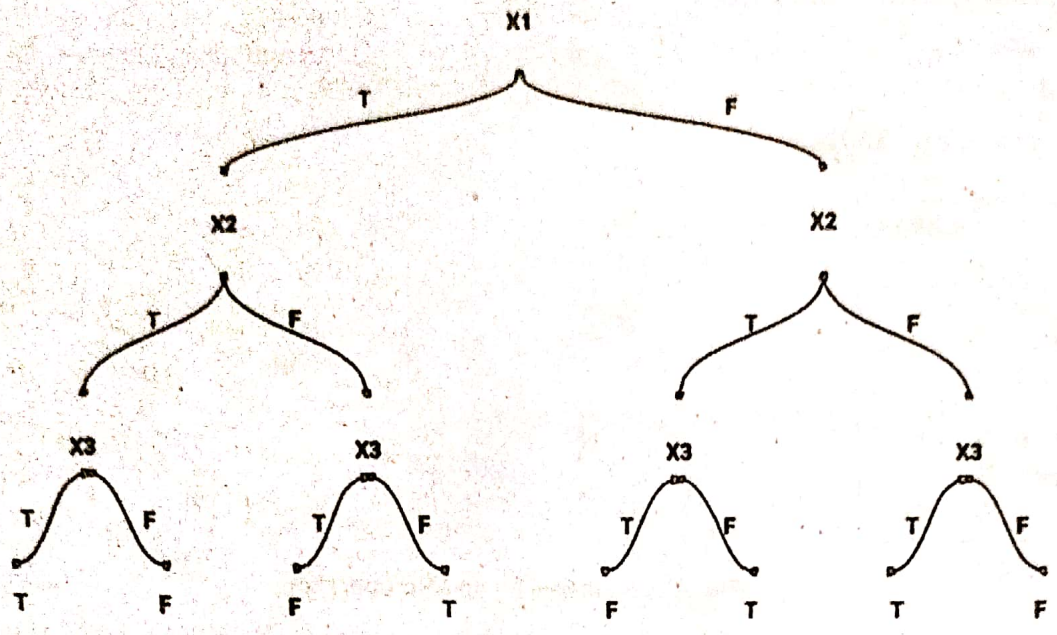


Fig.: Decision tree for an XOR operation involving three operands

In the decision tree, shown above (Fig.), for three attributes there are 7 nodes in the tree, i.e., for $n = 3$, number of nodes = 2^3-1. Similarly, if we have n attributes, there are 2^n nodes (approx.) in the decision tree. So, the tree requires exponential number of nodes in the worst case.

We can represent boolean operations using decision trees. But, what other kind of functions can we represent and if we search over the various possible decision trees to find the right one, how many decision trees do we have to worry about. Let's answer this question by finding out the possible number of decision trees we can generate given N different attributes (assuming the attributes are boolean). Since a truth table can be transformed into a decision tree, we will form a truth table of N attributes as input.

X1	X2	X3	....	XN	OUTPUT
T	T	T	...	T	T
T	T	T	...	F	F
...	...	...	...	...	....
...	...	...	...	...	.....
...	...	...	...	...	.....
F	F	F	...	F	F

The above truth table has 2^n rows (i.e. the number of nodes in the decision tree), which represents the possible combinations of the input attributes, and since each node can a hold a binary value, the number of ways to fill the values in the decision tree is ${2^{2^n}}$. Thus, the space of decision trees, i.e, the hypothesis space of the decision tree is very expressive because there are a lot of different functions it can represent. But, it also means one needs to have a clever way to search the best tree among them.

Decision tree boundary

Decision trees divide the feature space into axis-parallel rectangles or hyperplanes. Let's demonstrate this with help of an example. Let's consider a simple AND operation on two variables (see Fig 3.). Assume X and Y to be the coordinates on the x and y axes, respectively, and plot the possible values of X and Y (as seen the table below). Fig 7. represents the formation of the decision boundary as each decision is taken. We can see that as each decision is made, the feature space gets divided into smaller rectangles and more data points get correctly classified.

X	Y	X AND Y
0	0	0
0	1	0
1	0	0
1	1	1

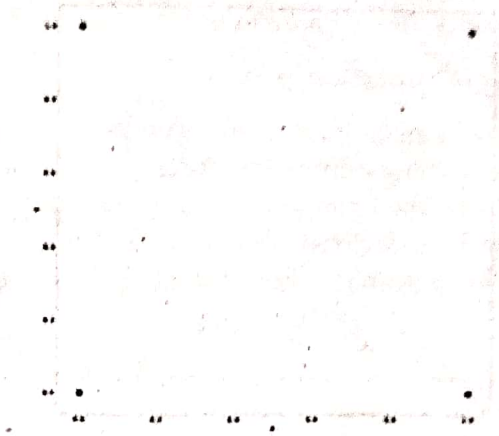


Fig.: Animation showing the formation of the decision tree boundary for AND operation

1.4 FORMALIZING THE LEARNING PROBLEM, ID3 ALGORITHM

Q6. Explain in detail about decision tree learning algorithm.

Ans :

The decision tree learning algorithm

The basic algorithm used in decision trees is known as the ID3 (by Quinlan) algorithm. The ID3 algorithm builds decision trees using a top-down, greedy approach. Briefly, the steps to the algorithm are: - Select the best attribute 'A' - Assign A as the decision attribute (test case) for the NODE. - For each value of A, create a new descendant of the NODE. - Sort the training examples to the appropriate descendant node leaf. - If examples are perfectly classified, then STOP else iterate over the new leaf nodes.

For ID3, we think of the best attribute has the most information gain, a measure that expresses how well an attribute splits that data into groups based on classification.

Pseudocode: ID3 is a greedy algorithm that grows the tree top-down, at each node selecting the attribute that best classifies the local training examples. This process continues until the tree perfectly classifies the training examples or until all attributes have been used.

The pseudocode assumes that the attributes are discrete and that the classification is binary. Examples are the training example. Target_attribute is the attribute whose value is to be predicted by the tree. Attributes is a list of other attributes that may be tested by the learned decision tree. Finally, it returns a decision tree that correctly classifies the given Examples.

ID3 (Examples, Target_attribute, Attributes): Create a root node for the tree. - If all Examples are positive, return the single-node tree root, with positive labels. - If all Examples are negative, return the single-node tree root, with negative labels. - If Attributes is empty, return the single-node tree root, with the most common labels of the Target_attribute in Examples. - Otherwise, begin - A ! the attribute from Attributes that best* classifies Examples - The decision attribute for root ! A - For each possible value v_i, of A, - Add a new tree branch below root, corresponding to the test A = v_i - Let Examples_vi be the subset of Examples that have value v_i for A. - If Examples_vi is empty - Then, below this new branch add a leaf node with the labels having the most common value of Target_attribute in Examples. - Else, below this new branch add the subtree (or call the function) - ID3(Examples_vi, Target_attribute, Attributes-{A}) - End - Return root

Calculating information gain

As stated earlier, information gain is a statistical property that measures how well a given attribute separates the training examples according to their target classification. In the figure below, we can see that an attribute with low information gain (right) splits the data relatively evenly and as a result doesn't bring us any closer to a decision. Whereas, an attribute with high information gain (left) splits the data into groups with an uneven number of positives and negatives and as a result helps in separating the two from each other.

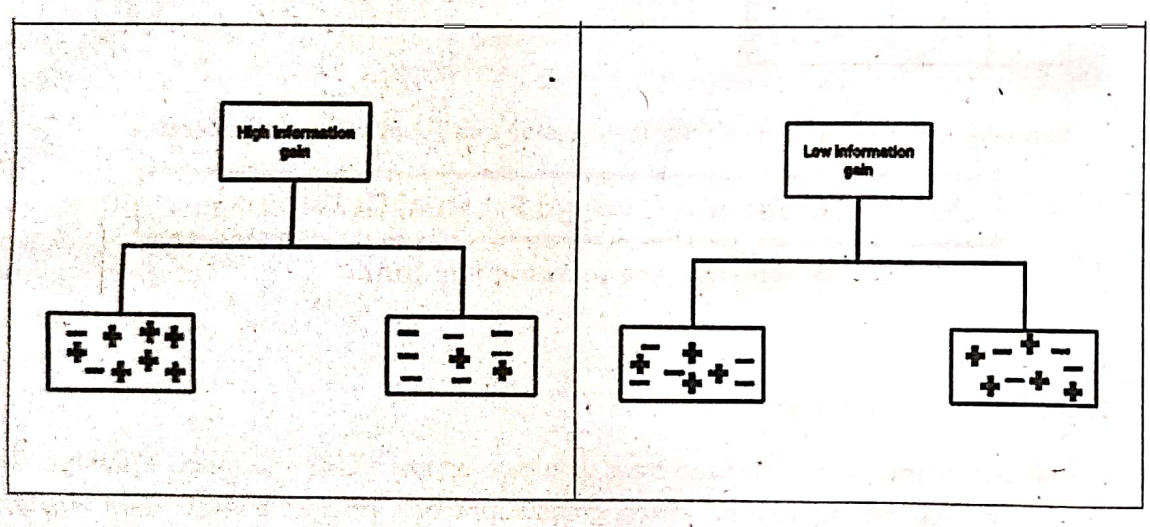


Fig.: Distribution of points in case of high and low information gain.

To define information gain precisely, we need to define a measure commonly used in information theory called entropy that measures the level of impurity in a group of examples. Mathematically, it is defined as:

Entropy: Σi=1"p"log2(pi)

pi=Probability of classification.

Since, the basic version of the ID3 algorithm deal with the case where classification are either positive or negative, we can define entropy as :

Entropy(S)=- p+log2p+"p"log2p-

where,

S is a sample of training examples

p_+ is the proportion of positive examples in S

p_- is the proportion of negative examples in S

To illustrate, suppose S is a sample containing 14 boolean examples, with 9 positive and 5 negative examples. Then, the entropy of S relative to this boolean classification is:

Entropy ([9+, 5-]) = -(9/14)$\cdot log_2$(9/14) - (5/14)$\cdot log_2$(5/14) = 0.940

Note that entropy is 0 if all the members of S belong to the same class. For example, if all members are positive (p_+=1), then p_- is 0, and Entropy(S) = -1$\cdot log_2$(1) -0$\cdot log_2$(0) = 0. Entropy is 1 when the sample contains an equal number of positive and negative examples. If the sample contains unequal number of positive and negative examples, entropy is between 0 and 1. The following figure shows the form of the entropy function relative to a boolean classification as p_+ varies between 0 and 1.

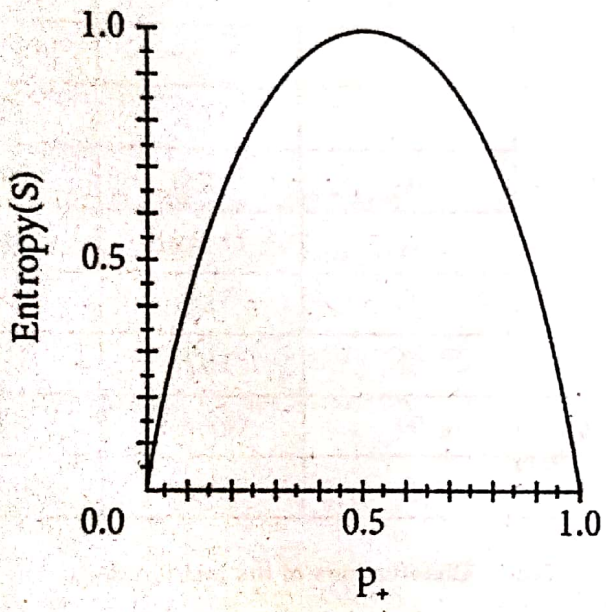


Fig.: Entropy function to a boolean classification, as the proportion p_+, of positive examples varies between 0 & 1.

Issues in decision trees

Avoiding overfitting

Since the ID3 algorithm continues splitting on attributes until either it classifies all the data points or there are no more attributes to splits on. As a result, it is prone to creating decision trees that overfit by performing really well on the training data at the expense of accuracy with respect to the entire distribution of data.

There are, in general, two approaches to avoid this in decision trees: Allow the tree to grow until it overfits and then prune it. Prevent the tree from growing too deep by stopping it before it perfectly classifies the training data.

A decision tree's growth is specified in terms of the number of layers, or depth, it's allowed to have. The data available to train the decision tree is split into training and testing data and then trees of various sizes are created with the help of the training data and tested on the test data. Cross-validation can also be used as part of this approach. Pruning the tree, on the other hand, involves testing the original tree against pruned versions of it. Leaf nodes are removed from the tree as long as the pruned tree performs better on the test data than the larger tree.

Incorporating continuous valued attributes

Our initial definition of ID3 is restricted to attributes that take on a discrete set of values. One way to make the ID3 algorithm more useful with continuous variables is to turn them, in a way, into discrete variables. Let's say in our example of Play Badminton the temperature is continuous (see the following table), we could test the information gain of certain partitions of the temperature values, such as temperature > 42.5. Typically, whenever the classification changes from no to yes or yes to no, the average of the two temperatures is taken as a potential partition boundary.

Day	Weather	Temperature	Humidity	Wind	Play?
1	Sunny	80	High	Weak	No
2	Cloudy	66	High	Weak	Yes
3	Sunny	43	Normal	Strong	Yes
4	Cloudy	82	High	Strong	Yes
5	Rainy	65	High	Strong	No
6	Rainy	42	Normal	Strong	No
7	Rainy	70	High	Weak	Yes
8	Sunny	81	High	Strong	No
9	Cloudy	69	Normal	Weak	Yes
10	Rainy	67	High	Strong	No

Table.: Observations of the last ten days.

Because 42 corresponds to No and 43 corresponds to Yes, 42.5 becomes a candidate. If any of the partitions end up exhibiting the greatest information gain, then it is used as an attribute and temperature is removed from the set of potential attributes to split on.

Alternative measures for selecting attributes

The information gain formula used by ID3 algorithm treats all of the variables the same regardless of their distribution and their importance. This is a problem when it comes to continuous variables or discrete variables with many possible values because training examples may be few and far between for each possible value, which leads to low entropy and high information gain by virtue of splitting the data into small subsets but results in a decision tree that might not generalize well.

One way to avoid this is to use some other measure to find the best attribute instead of information gain. An alternative measure to information gain is gain ratio (Quinlan 1986). Gain ratio tries to the correct the information gain's bias towards attributes with many possible values by adding a denominator to information gain called split information. Split Information tries to measure how broadly and uniformly the attribute splits the data:

$Split Information (S, A) = - \sum_{i=1}^{c}rac{|S_i|}{|S|} \cdot log_2 rac{|S_i|}{|S|}$

The Gain Ratio is defined in terms of Gain and Split Information as,

$Gain Ratio(S, A) \equivrac{Gain(S, A)}{Split Information(S, A)}$

One practical issue that arises in using gain ratio in place of information gain is that the denominator can be zero or very small when $|S_i|pprox|S|$ for one of the S_i. This either makes the Gain ratio undefined or very large for attributes that happen to have the same value for nearly all members of S.For example, if there's just one possible value for the attribute, then the formula equals $log_2$1 = 0.Luckily, we tend not to include attributes with 1 possible value in our training data because it is impossible to carry out the ID3 algorithm by splitting on an attribute with only 1 value, so Gain ratio doesn't have to handle the possibility of a denominator of 0. On the other hand, our continuous temperature example has 10 possible values in our training data, each of which occur once, which leads to -(1/10)$\cdot log_2$(1/10) = $log_2$10 . In general, the Split Information of an attribute with n equally distributed values is log_2n. These relatively large denominators significantly affect an attribute's chances of being the best attribute after an iteration of the ID3 algorithm and help in avoiding choices that perform particularly well on the training data but not so well outside of it.

Advantages and Disadvantages

Following are the advantages of decision trees:

- Easy to use and understand
- Can handle both categorical and numerical data
- Resistant to outliers, hence require little data preprocessing
- New features can be easily added
- Can be used to build larger classifiers by using ensemble methods.

Following are the disadvantages of decision trees:

- Prone to overfitting
- Require some kind of measurement as to how well they are doing
- Need to be careful with parameter tuning
- Can create biased learned trees if some classes dominate.

1.5 LIMITS OF LEARNING

1.5.1 Data Generating Distributions

Q7. Explain about limits of learning.

Ans :

Machine learning is a very general and useful framework, but it is not "magic" and will not always work. In order to better understand when it will and when it will not work, it is useful to formalize the learning problem more. This will also help us develop debugging strategies for learning algorithms.

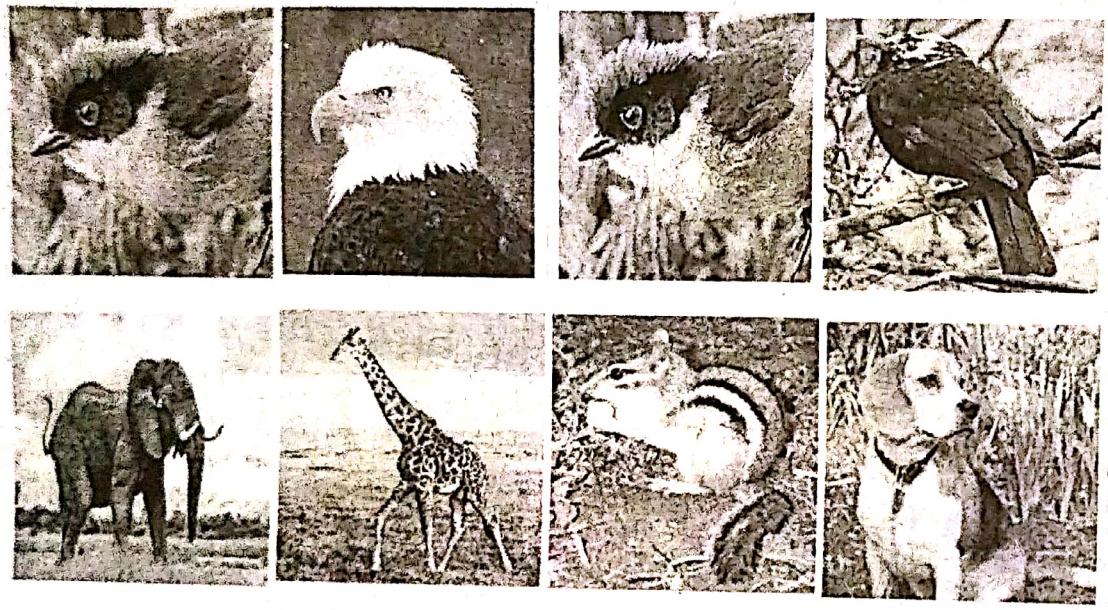
Data Generating Distributions Our underlying assumption for the majority of this book is that learning problems are characterized by some unknown probability distribution D over input/output pairs (x, y) " X×Y. Suppose that someone told you what D was. In particular, they gave you a Python function compute D that took two inputs, x and y, and returned the probability of that x, y pair under D. If you had access to such a function, classification becomes simple. We can define the Bayes optimal classifier as the classifier that, for any test input x^, simply returns the y^ that maximizes compute D(x^, y^), or, more formally.

1.6 Inductive Bias: What We Know Before the Data Arrives

Q8. Explain about inductive bias.

Ans :

In machine learning the term inductive bias refers to a set of assupmtion made by a learning in order to perform induction, that is, to generalize a finite set of observation (reading data) into a general made of the domain. In the below fig(a) we will binding data for a binary classification problem The two labels are 'A' and 'B' and you can see four examples for each label. Below, in Figure., you will see some test data. These images are left unlabeled. Go through quickly and, based on the training data, label these images.



Class 'A'

Class 'B'

Fig.: (a) Training data for a binary classification problem.

Most likely you produced one of two labelings: either ABBA or AABB. Which of these solutions is right? The answer is that you cannot tell based on the training data. If you give this same exam ple to 100 people, 60"70 of them come up with the ABBA prediction and 30 "40 come up with the AABB prediction. Why? Presumably be cause the first group believes that the relevant distinction is between"bird"and"non-bird"while these cond group believes that there levant distinction is between "fly" and" no-fly." This preference for one distinction (bird/non-bird)over an other (fly/no-fly) is abias that different human learners have.

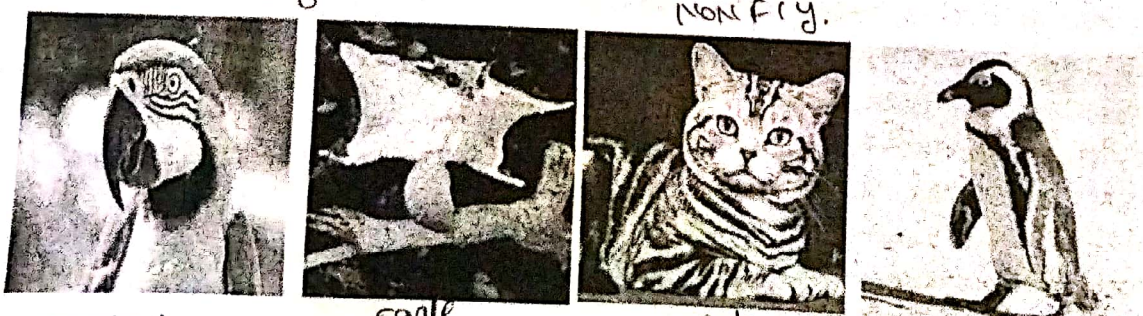


Fig.: (b) Test data for the same classification problem.

In the context of ML, it is called inductive bias in the absence of data that narrow down the relevant concept, what type of solutions are we more likely to prefer? Two thirds of people seem to have an inductive bias in favor of bird/non-bird, and one third seems to have an inductive bias in favor of fly/no-fly.

Through out this book you willl earn about several approaches to machine learning.The decision tree modelist he first such approach. These approaches differ primarily in thes or to finductive bias that they exhibit.

Consider a variant of the decision tree learning algorithm. In this variant, we will not allow the trees to grow beyond some predefined maximum depth, d. That is, once we have queried on many features, we can not query on any more and must just make the best guess we can at that point. This variant is called a shallow decision tree.

The key question is: What is the inductive bias of shallow decision trees is Roughly, their bias is that decisions can be made by only looking at a small number of features. For instance, as hallow decision tree would be very good at learning a function like students only like AI courses. It would be very bad at learning a function like if this student has liked an odd number of their past courses, they will like the next one; otherwise they will not. This latter is the parity function, which requires you to inspecte very feature to make a prediction. The inductive bias of a decision tree is that the sorts of things we want to learn to predictare more like the first example and less like these kind example.

1.7 Not Everything is Learnable

Q9. Write about not everything learnable.

Ans :

Although machine learning works well perhaps astonishingly well—in many cases, it is important to keep in mind that it is not magical.There are many reasons why a machine learning algorithm might fail on some learning task.

There could be noise in the training data. Noise can occur both at the feature level and at the la bel level. Some features might correspond to measurements taken by sensors. For instance, a robot might use a laser range finder to compute its distance to a wall. However, this sens or might fail and return an in correct value. In a sentiment classification problem, some one might have a type their review of a course. These would lead to noise at the feature level. Theremig also be noise at the label level. A student might write a scathingly negative review of a course, but then accidentally click the wrong but on for the course rating.

The features available for learning might simply be insufficient. For example, in a medical context, you might wish to diagnose whether a patient has cancer or not.You may be able to collect a large amount of data about this patient, such as gene expressions, X-rays, family histories, etc. But, even knowing all of this information exactly, it might still be impossible to judge for sure whether this patient has cancer or not. As a more contrived example, you might try to classify course reviews as positive or negative.

Some examples may not have a single correct answer. You might be building a system for safe web search, which removes of fensive web pages from search results. To build this system, you would collect a set of web pages and ask people to classify the mas of fensive or not. However, what one person considers offensive might be completely reasonable for another person. It is common to consider this as a form of label noise. Never the less, since you, as the designer of the learning system, have some control over this problem, it is some times help 2 ful to isolate it as as our difficulty. Finally, learning might fail because the inductive bias of the learning algorithm is to of ar away from the concept that is being learned.

In the bird/non-bird data, you might think that if you had got ten a few more training examples, you might have been able to tell whether this was in tended to be abird/non-bird classification or afly/no-fly classification. However, no one I've talked to has ever come up with the background is infocus classification. Even with many more training points, this is such an unusual distinction that it may be hard for any one to figure out it. In this case, the inductive bias of the learner is simply too mis aligned with the target classification toleam.

Note that the inductive bias source of error is fundamentally different than the other three sources of error. In the inductive bias case, it is the particular learning algorithm that you are using that can not cope with the dat a. May be if you switched to a different learning algorithm, you would be able to learn well. For instance, Neptunians might have evolved to care greatly about whether backgrounds are in focus, and for them this would be an easy classification to learn. For the other three sources of error, it is not an issue to do with the particular learning algorithm. The errorisa fundamental part of the learning problem.

1.8 UNDERFITTING AND OVERFITTING

Q10. Explain about Underfitting and Overfitting.

Ans :

Let us consider that we are designing a machine learning model. A model is said to be a good machine learning model if it generalizes any new input data from the problem domain in a proper way. This helps us to make predictions in the future data that the data model has never seen. Now, suppose we want to check how well our machine learning model learns and generalizes to the new data. For that, we have overfitting and underfitting, which are majorly responsible for the poor performances of the machine learning algorithms.

Before diving further let's understands two important terms:

- **Bias:** Assumptions made by a model to make a function easier to learn.
- **Variance:** If you train your data on training data and obtain a very low error, upon changing the data and then training the same previous model you experience a high error, this is variance.

Underfitting:

A statistical model or a machine learning algorithm is said to have underfitting when it cannot capture the underlying trend of the data. (It's just like trying to fit undersized pants!) Underfitting destroys the accuracy of our machine learning model. Its occurrence simply means that our model or the algorithm does not fit the data well enough. It usually happens when we have fewer data to build an accurate model and also when we try to build a linear model with fewer non-linear data. In such cases, the rules of the machine learning model are too easy and flexible to be applied on such minimal data and therefore the model will probably make a lot of wrong predictions. Under fitting can be avoided by using more data and also reducing the features by feature selection.

In a nutshell, Underfitting – High bias and low variance

Techniques to reduce underfitting:

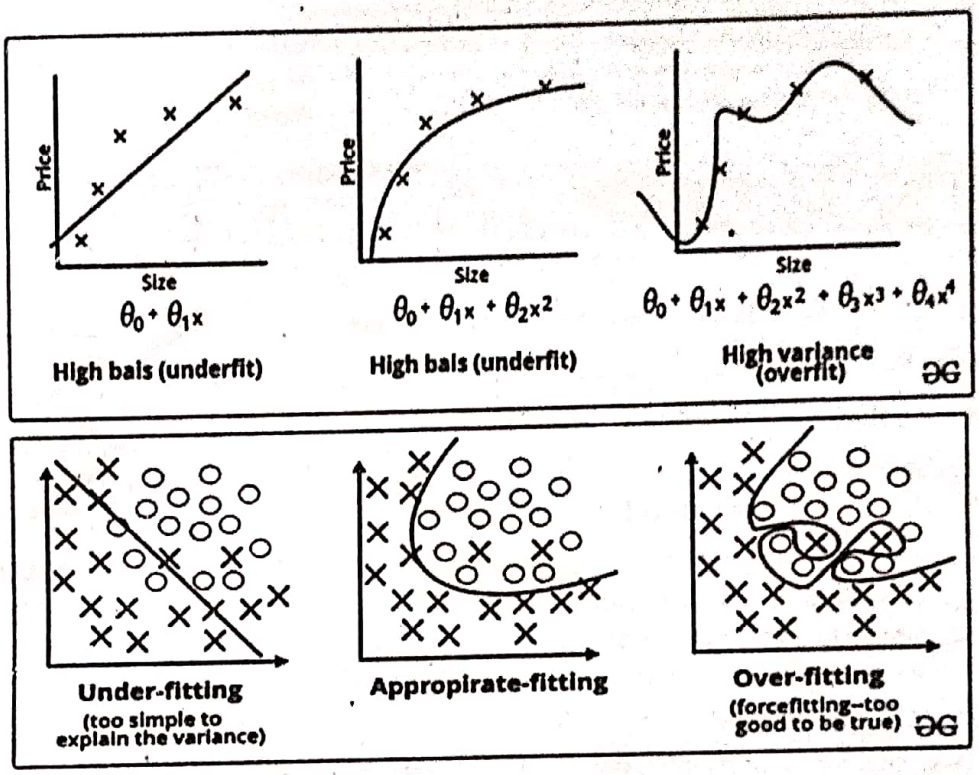
1. Increase model complexity
2. Increase the number of features, performing feature engineering
3. Remove noise from the data.
4. Increase the number of epochs or increase the duration of training to get better results.

Overfitting

A statistical model is said to be over fitted when we train it with a lot of data (just like fitting ourselves in oversized pants!). When a model gets trained with so much data, it starts learning from the noise and inaccurate data entries in our data set. Then the model does not categorize the data correctly, because of too many details and noise. The causes of overfitting are the non-parametric and non-linear methods because these types of machine learning algorithms have more freedom in building the model based on the data set and therefore they can really build unrealistic models. A solution to avoid overfitting is using a linear algorithm if we have linear data or using the parameters like the maximal depth if we are using decision trees.

In a nutshell, **Overfitting – High variance and low bias**

Examples:



Techniques to reduce overfitting:

1. Increase training data.
2. Reduce model complexity.
3. Early stopping during the training phase (have an eye over the loss over the training period as soon as loss begins to increase stop training).
4. Ridge Regularization and Lasso Regularization
5. Use dropout for neural networks to tackle overfitting.

Good Fit in a Statistical Model:

Ideally, the case when the model makes the predictions with 0 errors, is said to have a good fit on the data. This situation is achievable at a spot between overfitting and under fitting. In order to understand it, we will have to look at the performance of our model with the passage of time, while it is learning from training data set.

With the passage of time, our model will keep on learning and thus the error for the model on the training and testing data will keep on decreasing. If it will learn for too long, the model will become more prone to overfitting due to the presence of noise and less useful details. Hence the performance of our model will decrease. In order to get a good fit, we will stop at a point just before where the error starts increasing. At this point, the model is said to have good skills on training data sets as well as our unseen testing data set.

1.9 Separation of Training and Test Data

Q11. Write about separation of training and test data.

Ans :

- training set—a subset to train a model.
- test set—a subset to test the trained model.

You could imagine slicing the single data set as follows:

Fig.: Slicing a single data set into a training set and test set.

Make sure that your test set meets the following two conditions:

- Is large enough to yield statistically meaningful results.
- Is representative of the data set as a whole. In other words, don't pick a test set with different characteristics than the training set.

Assuming that your test set meets the preceding two conditions, your goal is to create a model that generalizes well to new data. Our test set serves as a proxy for new data. For example, consider the following figure. Notice that the model learned for the training data is very simple. This model doesn't do a perfect job—a few predictions are wrong. However, this model does about as well on the test data as it does on the training data. In other words, this simple model does not overfit the training data.

T wo models: one run on training data and the other on test data. The model is very simple, just a line dividing the orange dots from the blue dots. The loss on the training data is similar to the loss on the test data.

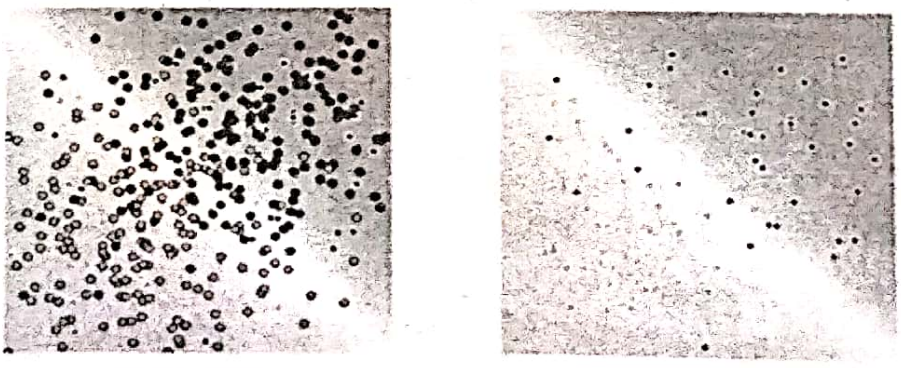


Fig.: Validating the trained model against test data.

Never train on test data. If you are seeing surprisingly good results on your evaluation metrics, it might be a sign that you are accidentally training on the test set. For example, high accuracy might indicate that test data has leaked into the training set.

For example, consider a model that predicts whether an email is spam, using the subject line, email body, and sender's email address as features. We apportion the data into training and test sets, with an 80-20 split. After training, the model achieves 99% precision on both the training set and the test set. We'd expect a lower precision on the test set, so we take another look at the data and discover that many of the examples in the test set are duplicates of examples in the training set (we neglected to scrub duplicate entries for the same spam email from our input database before splitting the data). We've inadvertently trained on some of our test data, and as a result, we're no longer accurately measuring how well our model generalizes to new data.

1.10 Models, Parameters and Hyper Parameters

Q12. What is a Model Parameter?

Ans :

A model parameter is a configuration variable that is internal to the model and whose value can be estimated from data.

- They are required by the model when making predictions.
- They values define the skill of the model on your problem.
- They are estimated or learned from data.
- They are often not set manually by the practitioner.
- They are often saved as part of the learned model.

Parameters are key to machine learning algorithms. They are the part of the model that is learned from historical training data.

In classical machine learning literature, we may think of the model as the hypothesis and the parameters as the tailoring of the hypothesis to a specific set of data.

Often model parameters are estimated using an optimization algorithm, which is a type of efficient search through possible parameter values.

- **Statistics:** In statistics, you may assume a distribution for a variable, such as a Gaussian distribution. Two parameters of the Gaussian distribution are the mean (mu) and the standard deviation (sigma). This holds in machine learning, where these parameters may be estimated from data and used as part of a predictive model.
- **Programming:** In programming, you may pass a parameter to a function. In this case, a parameter is a function argument that could have one of a range of values. In machine learning, the specific model you are using is the function and requires parameters in order to make a prediction on new data.

Whether a model has a fixed or variable number of parameters determines whether it may be referred to as "parametric" or "nonparametric".

Some examples of model parameters include:

- The weights in an artificial neural network.
- The support vectors in a support vector machine.
- The coefficients in a linear regression or logistic regression.

Q13. What is a Model Hyper parameter?

Ans :

A model hyperparameter is a configuration that is external to the model and whose value cannot be estimated from data.

- They are often used in processes to help estimate model parameters.
- They are often specified by the practitioner.
- They can often be set using heuristics.
- They are often tuned for a given predictive modeling problem.

We cannot know the best value for a model hyperparameter on a given problem. We may use rules of thumb, copy values used on other problems, or search for the best value by trial and error.

When a machine learning algorithm is tuned for a specific problem, such as when you are using a grid search or a random search, then you are tuning the hyperparameters of the model or order to discover the parameters of the model that result in the most skillful predictions.

However, our 'test data' was not magic. We simply took our 1000 examples, called 800 of them training data and called the other 200 "test" data. So instead, let's do the following. Let's take our original 1000 data points, and select 700 of them as training data. From the remainder, take 100 as development data and the remaining 200 as test data.

The job of the development data is to allow us to tune hyper parameters. The general approachis as follows:

1) Split your data into 70% training data, 10% development data and 20% test data.
2) For each possible setting of your hyper parameters:
   a) Train a model using that setting of hyper parameters on the training data.
   b) Compute this model's error rate on the development data.
3) From the above collection of models, choose the one that achieved the lowest error rate on developmen tdata.
4) Evaluate that model on the test data to estimate future test performance.

1.11 Real World Applications of Machine Learning

Q14. Explain the Applications of Machine Learning.

Ans : **(Imp.)**

For answer refer to Unit-I, Q.No. 3.

1.12 Geometry and Nearst Neighbors

1.12.1 From data to feature vectors

Q15. Explain in detail about Data to feature vector ? And K-nearest neighbors.

Ans : **(Imp.)**

In ML, feature vectors are used to represent numeric (or) symbolic characteristics called "Features" an object in mathematical different areas of ML and pattern processing ML algorithms tripically require a numerical representation of objects in order for the algorithms to do processing and statistical analysis feature vectors are the equivalent of vectors of explantory variables that are used in statistical procedures such as "Linear Regrention".

K-nearest neighbors

Prediction tasks as mapping inputs (course reviews) to outputs (course ratings). decomposing an input into a collection of features (e.g., words that occur in there view) forms a useful abstraction for learning. Therefore, inputs are nothing more than lists of feature values. This suggests a geometric view of data, where we have one dimension for every feature. In this view, examples are points in a high-dimensional space

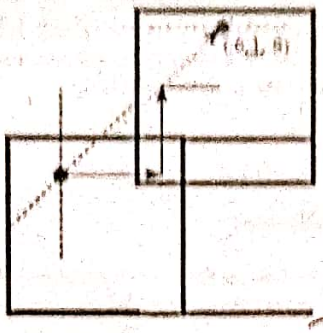
Once wet hink of a data set as a collection of points in high dimensional space, we can start performing geometric operations on this data. For instance, suppose you need to predict whether Alice willlike Algorithms. Perhaps we can try to find another student who is most similar to Alice, in terms of favorite courses. Say this student is Jeremy. If Jeremy liked Algorithms, then we might guess that Alice will as well. This is an example of a nearest neighbor model of learning. By inspecting this model, we'll see a completely different set of answers to the key learning questions.

Q16. Write about K-Nearest Neighbors with an example?

Ans :

The biggest advantage to thinking of examples as vectors in a high dimensional space is that it allows us to apply geometric concepts to machine learning. For instance, one of them most basic things that one can do in a vector space is compute distances. In two-dimensional space, the distance between 2,3 and 6,1 is given by $\sqrt{(2-6)^2+(3-1)^2} = 18 \approx 4.24$ .In general, in D-dimensional space, the Euclide an distance between vectors a and b is given by (see Figure for geometric intuition in three dimensions):

Now that you have access to distances between examples, you can start thinking about what it means to learn again. Consider Figure 3.3. We have a collection of training data consisting of positive examples and negative examples. There is a test point marked by a question mark. Your job is to guess the correct label for that point. Most likely, you decided that the label of this test point is positive.



One reason why you might have thought that is that you believe that the label for an example should be similar to the label of near by points. This is an example of a new form of inductive bias.

The nearest neighbor classifier is build upon this in sight. In comparis on to decision trees, the algorithm is ridiculously simple. Attraining time, we simply store the entire training set. At test time, we getatest examplex. To predictits label, we find the training examplex that is mosts imilartox. In particular, we find the training examplex that minimizesd(x, $\hat{x}$). Since x is a training example, it has a corresponding label, y. We predict that thelabel of $\hat{x}$ is a l soy.

Despite its simplicity, this near estneighbor classifier is incredibly effective. (Some might say frustratingly effective.) However, it is particularly prone to overfitting label noise. Consider the data in Figure You would probably want to label the test point positive. Unfortunately, it's near est neighbor happens to be negative. Since the nearest neighbor algorithm only looks at the single nearest neighbor, it cannot consider the" preponderance of evidence" that this points hould probably actually be a positive example. It will make an un-necessary error.

A solution to this problem is to consider more than just the single near est neighbor when making a classification decision. We can consider the k-nearest neighbors and let them vote on the correct class for this test point. If you consider the 3-nearest neighbors of the test point in Figure , you will see that two of them are positive and one is negative. Through voting, positive would win.

The full a lgorithm for K-near est neighbor classification is given in Algorithm. Note that there actually is no "training" phase for K-nearest neighbors. In this algorithm we have introduced five new conventions:

1. The training data is denoted by D.
2. We assume that there are N-many training examples.
3. These examples are pairs(x_1, y_1), (x_2, y_2), . . . , (x_N, y_N). (Warning: do not confusex_n, the n^{th} training example, with x_d, the d^{th} feature for examplex.)
4. Weuse [] to denote an empty list and $\oplus$ .to append·to that list.
5. Our predictionon $\hat{x}$ is called $\hat{y}$.

The first step in this algorithm is to compute distances from the test point to all training points (lines 2-4). The data points are then sorted according to distance. We then apply a clever trick of summing the class labels for each of the K nearest neighbors (lines 6-10) and using the sign of this sum as our prediction.

The big question, of course, i show to choose K. As we've seen, with K=1, we run the risk of over fitting.On the other hand, if K is large (for instance, K=N), then KNN-Predict will always predict the majority class. Clearly that is under fitting. So, K is a hyperparameter of the KNN algorithm that allows ustotrade-off between over fitting(small value of K) and underfitting (large value of K).

Algorithm 3 KNN- Predict (D,K, $\hat{x}$)

1: S←[]
2: fom=I to Ndo
3: S←S⊕d(xn,x²) //store distance to training examplen
4: end for
5: S←sort(S) //put lowest-distance objects first
6: $\hat{y}$ ←0
7: fork=I to Kdo
8: (dist, n)←Sk //n this is the kth closest data point
9: $\hat{y}$ ←+$\hat{y}$+ yn //vote according to the label for then thtraining point
10: end for
11: return sign($\hat{y}$) //return+1if $\hat{y}$ >0 and – 1if $\hat{y}$ <0

1.13 Decision Boundaries

Q17. What Is Decision Boundary? Explain in detail.

Ans :

A Decisin Boundary is a line, where all samples of one class are on one side of that line, and all samples of the other can are an opposite side of the line. The line separated one class from the other.

While training a classifier on a data set, using a specific classification algorithm, it is required to define a set of hyper-planes, called Decision Boundary that separates the data points into specific classes, where the algorithm switches from one class to another. On one side a decision boundary, a datapoints is more likely to be called as class A on the other side of the boundary, it's more likely to be called as class B.

Let's take an example of a Logistic Regression.

The goal of logistic regression, is to figure out some way to split the datapoints to have an accurate prediction of a given observation's class using the information present in the features.

Let's suppose we define a line that describes the decision boundary. So, all of the points on one side of the boundary shall have all the data points belong to class A and all of the points on one side of the boundary shall have all the data points belong to class B.

S(z)=1/(1+e^-z)

- S(z) = Output between 0 and 1 (probability estimate)
- z = Input to the function (z= mx + b)
- e = Base of natural log

Our current prediction function returns a probability score between 0 and 1. In order to map this to a discrete class (A/B), we select a threshold value or tipping point above which we will classify values into class A and below which we classify values into class B.

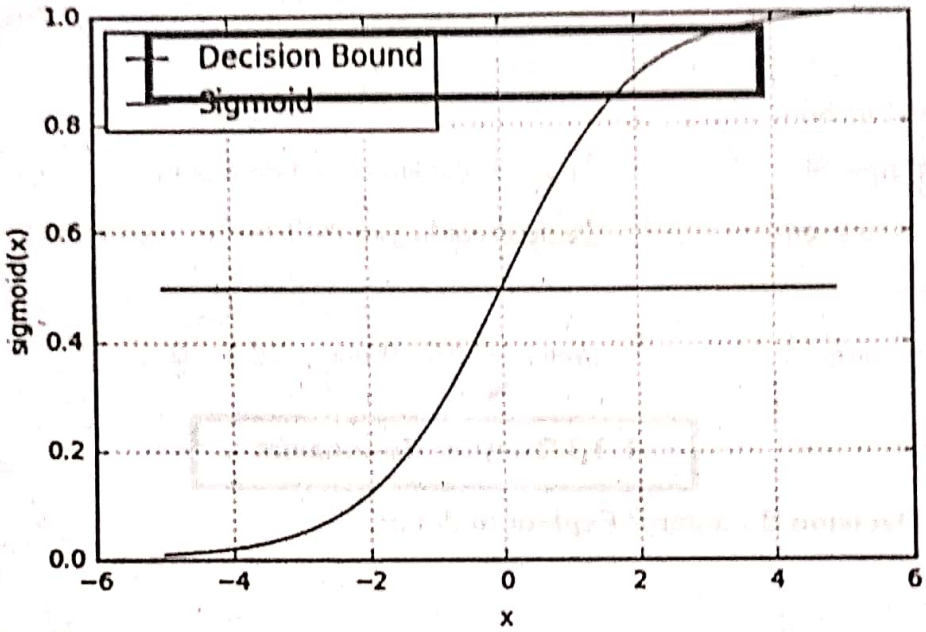
p>=0.5, class=A

p<=0.5, class=B

If our threshold was .5 and our prediction function returned .7, we would classify this observation belongs to class A. If our prediction was .2 we would classify the observation belongs to class B.

So, line with 0.5 is called the decision boundary.

In order to map predicted values to probabilities, we use the Sigmoid function.



IMPORTANCE OF DECISION BOUNDARY

A decision boundary, is a surface that separates data points belonging to different class lables. Decision Boundaries are not only confined to just the data points that we have provided, but also they span through the entire feature space we trained on. The model can predict a value for any possible combination of inputs in our feature space. If the data we train on is not 'diverse', the overall topology of the model will generalize poorly to new instances. So, it is important to analyse all the models which can be best suitable for 'diverse' data set, before using the model into production.

Examining decision boundaries is a great way to learn how the training data we select affects performance and the ability for our model to generalize. Visualization of decision boundaries can illustrate how sensitive models are to each data set, which is a great way to under stand how specific algorithms work, and their limitations for specific data sets.

1.14 K-MEANS CLUSTERING, HIGH DIMENSIONS

Q18. Explain in detail about k-mean clustering.

Ans :

K-Means Clustering Algorithm

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science. In this topic, we will learn what is K-means clustering algorithm, how the algorithm works, along with the Python implementation of k-means clustering.

K-Means Algorithm

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled data set into different clusters.

Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.

"It is an iterative algorithm that divides the unlabeled data set into 'k' different clusters in such a way that each data set belongs only one group that has similar properties."

It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled data set on its own without the need for any training.

It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

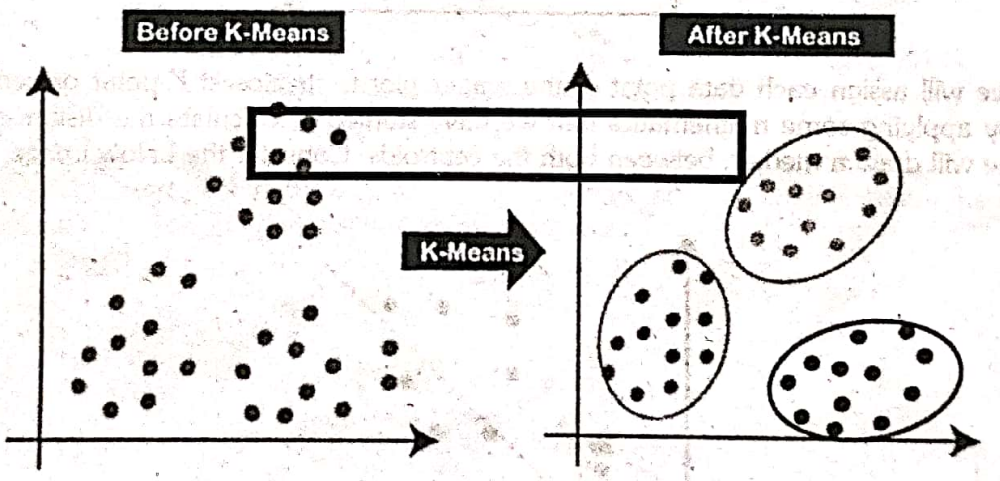
The algorithm takes the unlabeled data set as input, divides the data set into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be pre determined in this algorithm.

The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

Hence each cluster has data points with some commonalities, and it is away from other clusters.

The below diagram explains the working of the K-means Clustering Algorithm:



The working of K-Means Algorithm

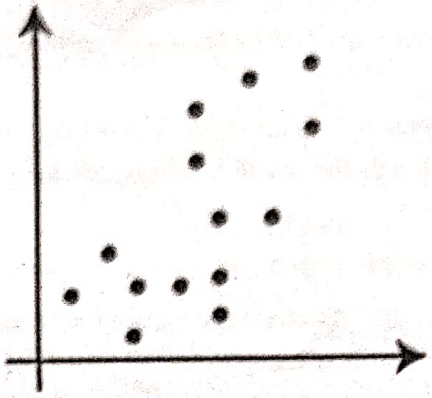
The working of the K-Means algorithm is explained in the below steps:

Step-1: Select the number K to decide the number of clusters.

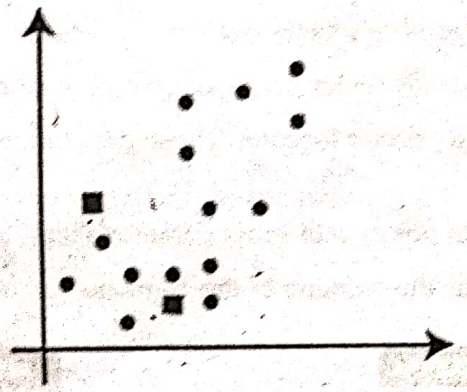
Step-2: Select random K points or centroids. (It can be other from the input data set).

Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.

Suppose we have two variables M1 and M2. The x-y axis scatter plot of these two variables is given below:

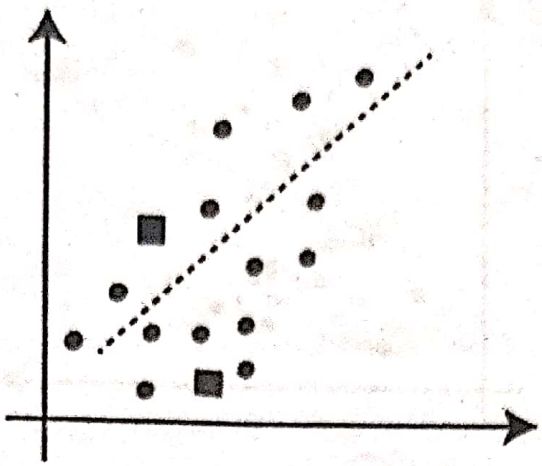


- Let's take number k of clusters, i.e., K=2, to identify the data set and to put them into different clusters. It means here we will try to group these data sets into two different clusters.
- We need to choose some random k points or centroid to form the cluster. These points can be either the points from the data set or any other point. So, here we are selecting the below two points as k points, which are not the part of our data set. Consider the below image:

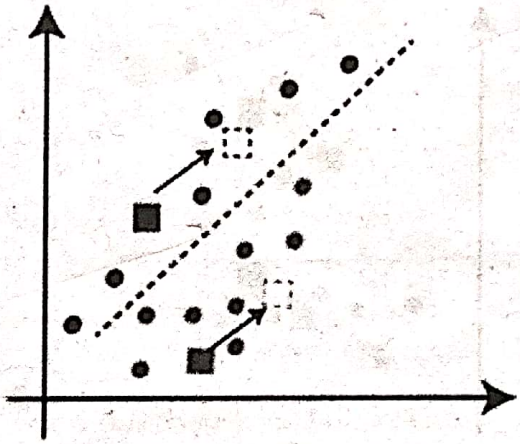


Now we will assign each data point of the scatter plot to its closest K-point or centroid. We will compute it by applying some mathematics that we have studied to calculate the distance between two points. So, we will draw a median between both the centroids. Consider the below image:

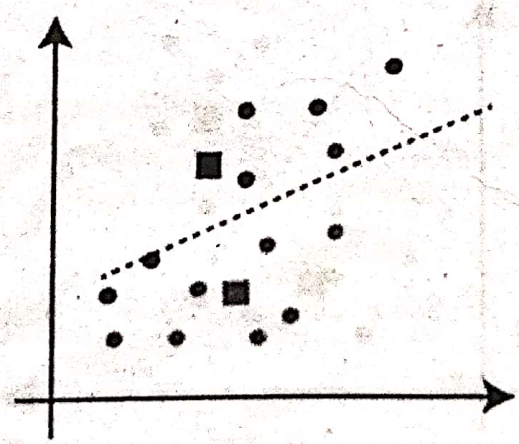
From the above image, it is clear that points left side of the line is near to the K1 or blue centroid, and points to the right of the line are close to the yellow centroid. Let's color them as blue and yellow for clear visualization.



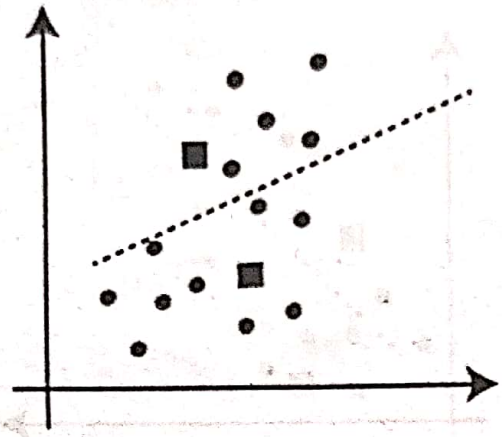
As we need to find the closest cluster, so we will repeat the process by choosing a new centroid. To choose the new centroids, we will compute the center of gravity of these centroids, and will find new centroids as below:



Next, we will reassign each data point to the new centroid. For this, we will repeat the same process of finding a median line. The median will be like below image:

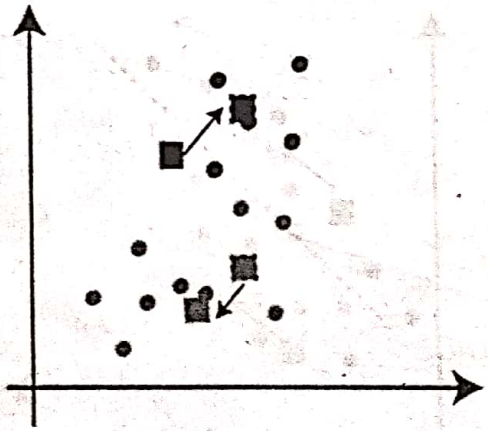


From the above image, we can see, one yellow point is on the left side of the line, and two blue points are right to the line. So, these three points will be assigned to new centroids.

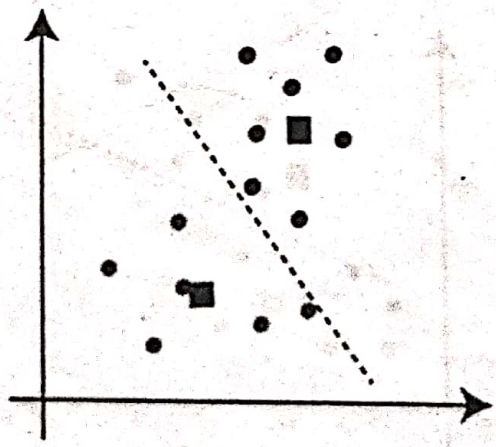


As reassignment has taken place, so we will again go to the step-4, which is finding new centroids or K-points.

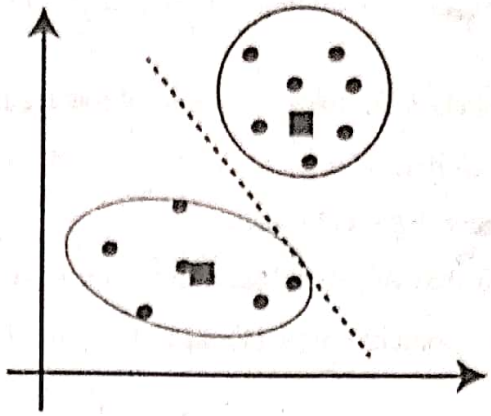
- We will repeat the process by finding the center of gravity of centroids, so the new centroids will be as shown in the below image:



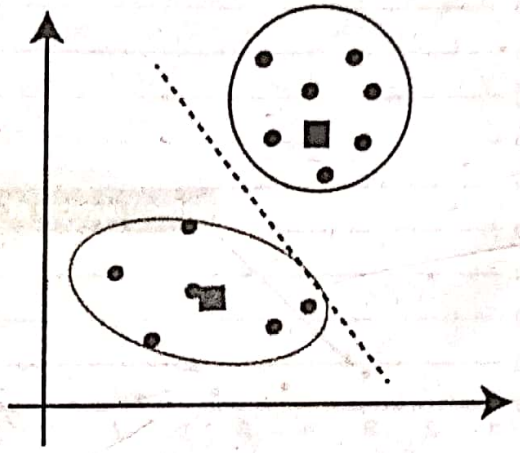
As we got the new centroids so again will draw the median line and reassign the data points. So, the image will be:



We can see in the above image; there are no dissimilar data points on either side of the line, which means our model is formed. Consider the below image:



As our model is ready, so we can now remove the assumed centroids, and the two final clusters will be as shown in the below image:



How to choose the value of "K number of clusters" in K-means Clustering?

The performance of the K-means clustering algorithm depends upon highly efficient clusters that it forms. But choosing the optimal number of clusters is a big task. There are some different ways to find the optimal number of clusters, but here we are discussing the most appropriate method to find the number of clusters or value of K. The method is given below:

Elbow Method

The Elbow method is one of the most popular ways to find the optimal number of clusters. This method uses the concept of WCSS value. WCSS stands for Within Cluster Sum of Squares, which defines the total variations within a cluster. The formula to calculate the value of WCSS (for 3 clusters) is given below:

$$WCSS = \Sigma_{Pi \text{ in Cluster1}} \text{ distance}(P_i\ C_1)^2 + \Sigma_{Pi \text{ in Cluster2}} \text{distance}\ (P_i\ C_2)^2 + \Sigma_{Pi \text{ in CLuster3}} \text{distance}\ (P_i\ C_3)^2$$

In the above formula of WCSS,

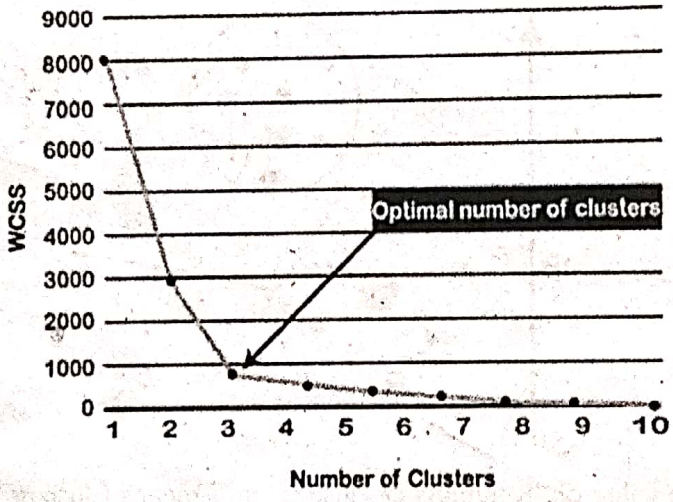
$\Sigma_{Pi \text{ in Cluster1}}$ distance $(P_i\ C_1)^2$: It is the sum of the square of the distances between each data point and its centroid within a cluster1 and the same for the other two terms.

To measure the distance between data points and centroid, we can use any method such as Euclidean distance or Manhattan distance.

To find the optimal value of clusters, the elbow method follows the below steps:

- It executes the K-means clustering on a given data set for different K values (ranges from 1-10).
- For each value of K, calculates the WCSS value.
- Plots a curve between calculated WCSS values and the number of clusters K.
- The sharp point of bend or a point of the plot looks like an arm, then that point is considered as the best value of K.

Since the graph shows the sharp bend, which looks like an elbow, hence it is known as the elbow method. The graph for the elbow method looks like the below image:



Short Question and Answers

1. What Is Machine Learning?

Ans :

Machine learning is the concept that a computer program can learn and adapt to new data without human intervention. Machine learning is a field of artificial intelligence (AI) that keeps a computer's built-in algorithms current regardless of changes in the worldwide economy.

- Machine learning is an area of artificial intelligence (AI) with a concept that a computer program can learn and adapt to new data without human intervention.
- A complex algorithm or source code is built into a computer that allows for the machine to identify data and build predictions around the data that it identifies.
- Machine learning is useful in parsing the immense amount of information that is consistently and readily available in the world to assist in decision making.
- Machine learning can be applied in a variety of areas, such as in investing, advertising, lending, organizing news, fraud detection, and more.

2. What are the applications of ML?

Ans :

Machine learning is a buzzword for today's technology, and it is growing very rapidly day by day. We are using machine learning in our daily life even without knowing it such as Google Maps, Google assistant, Alexa, etc. Below are some most trending real-world applications of Machine Learning:

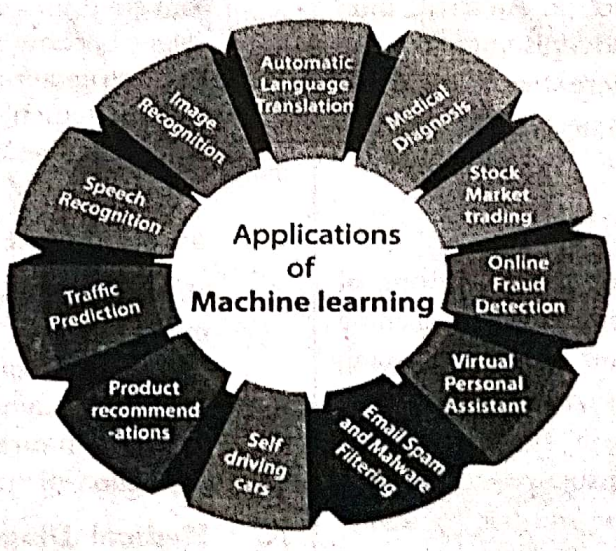


Image Recognition

Image recognition is one of the most common applications of machine learning. It is used to identify objects, persons, places, digital images, etc. The popular use case of image recognition and face detection is, Automatic friend tagging suggestion.

Speech Recognition

While using Google, we get an option of "Search by voice," it comes under speech recognition, and it's a popular application of machine learning.

3. Traffic prediction

If we want to visit a new place, we take help of Google Maps, which shows us the correct path with the shortest route and predicts the traffic conditions.

It predicts the traffic conditions such as whether traffic is cleared, slow-moving, or heavily congested with the help of two ways:

- Real Time location of the vehicle form Google Map app and sensors
- Average time has taken on past days at the same time.

Everyone who is using Google Map is helping this app to make it better. It takes information from the user and sends back to its database to improve the performance.

4. Product recommendations

Machine learning is widely used by various e-commerce and entertainment companies such as Amazon, Netflix, etc., for product recommendation to the user. Whenever we search for some product on Amazon, then we started getting an advertisement for the same product while internet surfing on the same browser and this is because of machine learning.

5. Self-driving cars

One of the most exciting applications of machine learning is self-driving cars. Machine learning plays a significant role in self-driving cars. Tesla, the most popular car manufacturing company is working on self-driving car. It is using unsupervised learning method to train the car models to detect people and objects while driving.

6. Email Spam and Malware Filtering

Whenever we receive a new email, it is filtered automatically as important, normal, and spam. We always receive an important mail in our inbox with the important symbol and spam emails in our spam box, and the technology behind this is Machine learning. Below are some spam filters used by Gmail:

7. Virtual Personal Assistant

We have various virtual personal assistants such as Google assistant, Alexa, Cortana, Siri. As the name suggests, they help us in finding the information using our voice instruction. These assistants can help us in various ways just by our voice instructions such as Play music, call someone, Open an email, Scheduling an appointment, etc.

8. Online Fraud Detection

Machine learning is making our online transaction safe and secure by detecting fraud transaction. Whenever we perform some online transaction, there may be various ways that a fraudulent transaction can take place such as fake accounts, fake ids, and steal money in the middle of a transaction. So to detect this, Feed Forward Neural network helps us by checking whether it is a genuine transaction or a fraud transaction.

For each genuine transaction, the output is converted into some hash values, and these values become the input for the next round. For each genuine transaction, there is a specific pattern which gets change for the fraud transaction hence, it detects it and makes our online transactions more secure.

9. Stock Market trading

Machine learning is widely used in stock market trading. In the stock market, there is always a risk of up and downs in shares, so for this machine learning's long short term memory neural network is used for the prediction of stock market trends.

10. Medical Diagnosis

In medical science, machine learning is used for diseases diagnoses. With this, medical technology is growing very fast and able to build 3D models that can predict the exact position of lesions in the brain.

It helps in finding brain tumors and other brain-related diseases easily.

11. Automatic Language Translation

Nowadays, if we visit a new place and we are not aware of the language then it is not a problem at all, as for this also machine learning helps us by converting the text into our known languages. Google's GNMT (Google Neural Machine Translation) provide this feature, which is a Neural Machine Learning that translates the text into our familiar language, and it called as automatic translation.

The technology behind the automatic translation is a sequence to sequence learning algorithm, which is used with image recognition and translates the text from one language to another language.

3. Explain about Machine learning Life cycle.

Ans :

Machine learning has given the computer systems the abilities to automatically learn without being explicitly programmed. But how does a machine learning system work? So, it can be described using the life cycle of machine learning. Machine learning life cycle is a cyclic process to build an efficient machine learning project. The main purpose of the life cycle is to find a solution to the problem or project.

Machine learning life cycle involves seven major steps, which are given below:

- Gathering Data
- Data preparation
- Data Wrangling
- Analyse Data
- Train the model
- Test the model
- Deployment

4. What are the types of ML Problems?

Ans :

There are common classes of problem in Machine Learning. The problem classes below are archetypes for most of the problems we refer to when we are doing Machine Learning.

- **Classification**

Data is labelled meaning it is assigned a class, for example spam/non-spam or fraud/non-fraud. The decision being modelled is to assign labels to new unlabelled pieces of data. This can be thought of as a discrimination problem, modelling the differences or similarities between groups.

- **Regression**

Data is labelled with a real value (think floating point) rather then a label. Examples that are easy to understand are time series data like the price of a stock over time, The decision being modelled is what value to predict for new unpredicted data.

- **Clustering**

Data is not labelled, but can be divided into groups based on similarity and other measures of natural structure in the data. An example from the above list would be organising pictures by faces without names, where the human user has to assign names to groups, like iPhoto on the Mac.

- **Rule Extraction**

Data is used as the basis for the extraction of propositional rules (antecedent/consequent aka if-then). Such rules may, but are typically not directed, meaning that the methods discover statistically supportable relationships between attributes in the data, not necessarily involving something that is being predicted. An example is the discovery of the relationship between the purchase of beer and diapers (this is data mining folk-law, true or not, it's illustrative of the desire and opportunity).

5. Explain about decision tree model of learning.

Ans :

A decision tree is a tree-like graph with nodes representing the place where we pick an attribute and ask a question; edges represent the answers to

the question; and the leaves represent the actual output or class label. They are used in non-linear decision making with simple linear decision surface.

Decision trees classify the examples by sorting them down the tree from the root to some leaf node, with the leaf node providing the classification to the example. Each node in the tree acts as a test case for some attribute, and each edge descending from that node corresponds to one of the possible answers to the test case. This process is recursive in nature and is repeated for every subtree rooted at the new nodes.

6. Write a Short notes on decision tree learning algorithm.

Ans :

The decision tree learning algorithm

The basic algorithm used in decision trees is known as the ID3 (by Quinlan) algorithm. The ID3 algorithm builds decision trees using a top-down, greedy approach. Briefly, the steps to the algorithm are: - Select the best attribute '! A - Assign A as the decision attribute (test case) for the NODE. - For each value of A, create a new descendant of the NODE. - Sort the training examples to the appropriate descendant node leaf. - If examples are perfectly classified, then STOP else iterate over the new leaf nodes.

Now, the next big question is how to choose the best attribute. For ID3, we think of the best attribute in terms of which attribute has the most information gain, a measure that expresses how well an attribute splits that data into groups based on classification.

Pseudocode: ID3 is a greedy algorithm that grows the tree top-down, at each node selecting the attribute that best classifies the local training examples. This process continues until the tree perfectly classifies the training examples or until all attributes have been used.

The pseudocode assumes that the attributes are discrete and that the classification is binary. Examples are the training example. Target_ attribute is the attribute whose value is to be predicted by the tree. Attributes is a list of other attributes that may be tested by the learned decision tree.

7. Explain about limits of learning.

Ans :

Data Generating Distributions

Machine learning is a very general and useful framework, but it is not "magic" and will not always work. In order to better understand when it will and when it will not work, it is useful to formalize the learning problem more. This will also help us develop debugging strategies for learning algorithms.

Data Generating Distributions Our underlying assumption for the majority of this book is that learning problems are characterized by some unknown probability distribution D over input/ output pairs (x, y) " X×Y. Suppose that someone told you what D was. In particular, they gave you a Python function computeD that took two inputs, x and y, and returned the probability of that x, y pair under D. If you had access to such a function, classification becomes simple. We can define the Bayes optimal classifier as the classifier that, for any test input x^, simply returns the y^ that maximizes computeD(x^, y^), or, more formally.

8. What is a Model Parameter?

Ans :

A model parameter is a configuration variable that is internal to the model and whose value can be estimated from data.

- They are required by the model when making predictions.
- They values define the skill of the model on your problem.
- They are estimated or learned from data.
- They are often not set manually by the practitioner.
- They are often saved as part of the learned model.

Parameters are key to machine learning algorithms. They are the part of the model that is learned from historical training data.

In classical machine learning literature, we may think of the model as the hypothesis and the parameters as the tailoring of the hypothesis to a specific set of data.

Often model parameters are estimated using an optimization algorithm, which is a type of efficient search through possible parameter values.

- **Statistics**

  In statistics, you may assume a distribution for a variable, such as a Gaussian distribution. Two parameters of the Gaussian distribution are the mean (mu) and the standard deviation (sigma). This holds in machine learning, where these parameters may be estimated from data and used as part of a predictive model.

- **Programming**

  In programming, you may pass a parameter to a function. In this case, a parameter is a function argument that could have one of a range of values. In machine learning, the specific model you are using is the function and requires parameters in order to make a prediction on new data.

Whether a model has a fixed or variable number of parameters determines whether it may be referred to as "parametric" or "nonparametric".

Some examples of model parameters include:

- The weights in an artificial neural network.
- The support vectors in a support vector machine.
- The coefficients in a linear regression or logistic regression.

9. What is a Model Hyperparameter?

Ans :

A model hyperparameter is a configuration that is external to the model and whose value cannot be estimated from data.

- They are often used in processes to help estimate model parameters.
- They are often specified by the practitioner.
- They can often be set using heuristics.
- They are often tuned for a given predictive modeling problem.

We cannot know the best value for a model hyperparameter on a given problem. We may use rules of thumb, copy values used on other problems, or search for the best value by trial and error.

When a machine learning algorithm is tuned for a specific problem, such as when you are using a grid search or a random search, then you are tuning the hyperparameters of the model or order to discover the parameters of the model that result in the most skillful predictions.

However, our 'test data' was not magic. We simply took our 1000 examples, called 800 of them training data and called the other 200 "test" data. So instead, let's do the following. Let's take our original 1000data points, and select 700 of them as training data. From theremainder, take 100 as development data and the remaining 200 as test data.

The job of the development data is to allowus to tune hyper parameters. The general approachis as follows:

1) Split your data into 70% training data, 10% development data and 20% test data.
2) For each possible setting of your hyper parameters:
   a) Train a model using that setting of hyper parameters on the training data.
   b) Compute this model's error rate on the development data.
3) From the above collection of models, choose the one that achieved the lowest error rate on developmen tdata.
4) Evaluate that model on the test data to estimate future test performance.

10. Explain in detail about k-nearest neighbors.

Ans :

Prediction tasks as mapping inputs to outputs decomposing an input into a collection of features (e.g., words that occurin there view) forms a useful abstraction for learning. Therefore, inputs are nothing more thanlists of feature values.This suggests a geometric viewof data, where we have one dimension for every feature.In this view, examples are points in a high-dimensionalspace

Once wet hink of a data setas acollection of points in high dimensional space, we can start performing geometric operations on this data. For instance, suppose you need to predict whether Alice willlike Algorithms. Perhaps we can try to find another student who is most similar to Alice, interms of favorite courses. Say this student is Jeremy. If Jeremy liked Algorithms, then we might guess that Alice will as well. This is an example of a nearest neighbor model of learning. By inspecting this model, we'll see a completely different set of answers to the key learning questions.

11. What Is Decision Boundary?

Ans :

While training a classifier on a dataset, using a specific classification algorithm, it is required to define a set of hyper-planes, called Decision Boundary that separates the data points into specific classes, where the algorithm switches from one class to another. On one side a decision boundary, a datapoints is more likely to be called as class A on the other side of the boundary, it's more likely to be called as class B.

Let's take an example of a Logistic Regression.

The goal of logistic regression, is to figure out some way to split the datapoints to have an accurate prediction of a given observation's class using the information present in the features.

Let's suppose we define a line that describes the decision boundary. So, all of the points on one side of the boundary shall have all the datapoints belong to class A and all of the points on one side of the boundary shall have all the datapoints belong to class B.

S(z)=1/(1+e^-z)

- S(z) = Output between 0 and 1 (probability estimate)
- z = Input to the function (z= mx + b)
- e = Base of natural log

Our current prediction function returns a probability score between 0 and 1. In order to map this to a discrete class (A/B), we select a threshold value or tipping point above which we will classify values into class A and below which we classify values into class B.

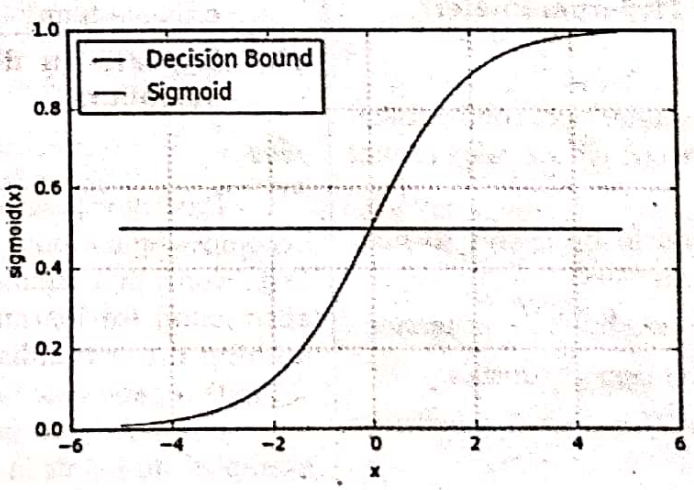
p>=0.5,class=A

p<=0.5,class=B

If our threshold was .5 and our prediction function returned .7, we would classify this observation belongs to class A. If our prediction was .2 we would classify the observation belongs to class B.

So, line with 0.5 is called the decision boundary.

In order to map predicted values to probabilities, we use the Sigmoid function.



12. Write about K-Means Clustering Algorithm.

Ans :

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science. In this topic, we will learn what is K-means clustering algorithm, how the algorithm works, along with the Python implementation of k-means clustering.

What is K-Means Algorithm?

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters.

Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.

"It is an iterative algorithm that divides the unlabeled dataset into 'k' different clusters in such a way that each dataset belongs only one group that has similar properties."

It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.

It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

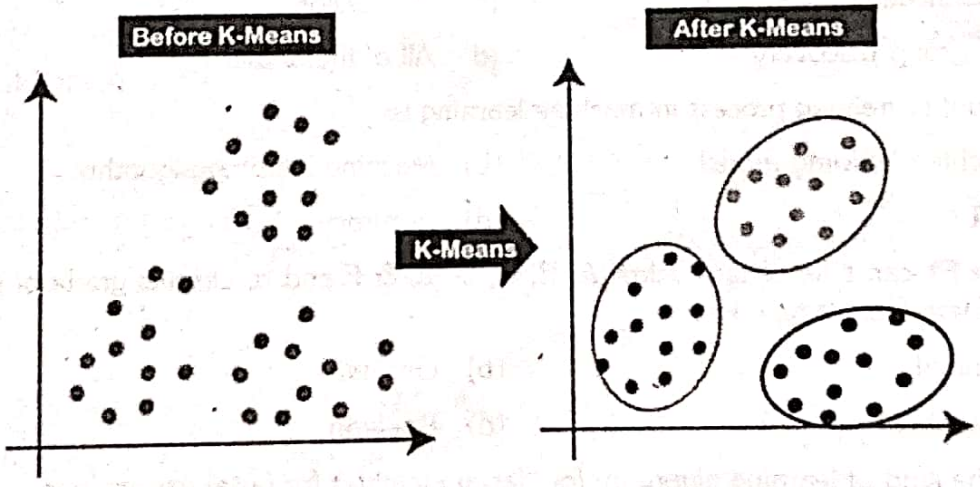
The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

Hence each cluster has datapoints with some commonalities, and it is away from other clusters.

The below diagram explains the working of the K-means Clustering Algorithm:



One Mark Answers

1. What is Decision Boundary?

Ans :

While training a classifier on a dataset, using a specific classification algorithm, it is required to define a set of hyper-planes, called Decision Boundary.

2. What do we mean by learning?

Ans :

Learning is "a process that leads to change, which occurs as a result of experience and increases the potential for improved performance and future learning". The change in the learner may happen at the level of knowledge, attitude or behavior..

3. What Is Machine Learning?

Ans :

Machine learning is the concept that a computer program can learn and adapt to new data without human intervention. Machine learning is a field of artificial intelligence (AI) that keeps a computer's built-in algorithms current regardless of changes in the worldwide economy.

4. What is K-Means Algorithm?

Ans :

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters.

5. Define Decision tree?

Ans :

A decision tree is a tree-like graph with nodes representing the place where we pick an attribute and ask a question; edges represent the answers to the question; and the leaves represent the actual output or class label. They are used in non-linear decision making with simple linear decision surface.