Unit-IV Introduction to Pattern Recognition and Machine Learning: Patterns, features, pattern representation, the curse of dimensionality, dimensionality reduction. Classification—linear and non-linear. Bayesian, Perceptron, Nearest neighbor classifier, Logistic regression, Naïve-Bayes, decision trees and random forests; boosting and bagging.Clustering---partitional and hierarchical; k-means clustering. Regression. Cost functions, training and testing a classifier. Cross-validation, Class-imbalance – ways of handling, Confusion matrix, evaluation metrics.

https://www.javatpoint.com/supervised-machine-learning

What is Pattern Recognition?

Pattern recognition is the process of recognizing patterns by using machine learning algorithm. Or Pattern Recognition is defined as the process of identifying the trends in the given pattern. A **pattern** can be defined as anything that follows a trend and exhibits some kind of regularity. The recognition of patterns can be done physically, mathematically or by the use of **algorithms**. When we talk about pattern recognition in **machine learning**, it indicates the use of powerful algorithms for identifying the regularities in the given data. **Pattern recognition** is widely used in the new age technical domains like computer vision, speech recognition, face recognition, etc.

Ex:
Pattern recognition is widely used in the new age technical domains like computer vision, speech recognition, face recognition, etc.

Patterns:

Patterns are everywhere. It belongs to every aspect of our daily lives. Starting from the design and colour of our clothes to using intelligent voice assistants, When we were in school, we were often given the task of identifying the missing alphabets or to predict which number would come in a sequence next or to join the dots for completing the figure. The prediction of the missing number or alphabet involved analyzing the trend followed by the given numbers or alphabets. This is what pattern recognition in Machine Learning exactly means.

Facebook posts and photos, your eyes can stop on a familiar face (despite tons of other information next to it). We are so used to pattern recognition thanks to our brains that we don't even stop to think about what powers the technological side of it.

our brain when we match some information that we encounter with data stored in our memory. For example, when a mom teaches her kid to count, she says, "One, two, three." After multiple repetitions, when mom says, "One, two…", the child can respond with "Three." As we can see, the child recognized the pattern.

What is pattern recognition in computer science? In computer science and machine learning, pattern recognition is a technology that matches the information stored in the database with the incoming data.

What is a pattern?

A pattern represents a physical object or an abstract notion. For ex- ample, the pattern may represent physical objects like balls, animals or furniture. Abstract notions could be like whether a person will play tennis or not (depending on features like weather etc.).

Pattern is everything around in this digital world. A pattern can either be seen physically or it can be observed mathematically by applying algorithms.
Example: The colors on the clothes, speech pattern etc. In computer science, a pattern is represented using vector feature values.

Machine learning:

ML is a subset of AI. It acts like a human. ML has an ability to learn from the data and makes predictions, improve from the experience. It was introduced by Arthur lee Samuel. In 1959 it was implemented.

Types of ML:

1.supervised- input variables and output variables in dataset.

2.unsupervised

3.Reinforcement learning

Terminology:

Data set consists of rows and columns.

Predict esal:

Emno, empexp, esal.

1	1	1000
2	2	2000
3	3	3000
4	4	?

Emno, empsal we pass to model as input and esal as output label. We will train the model using the data.

Based on empex let us find out esal. Let us predict esal. Esal are called as class labels or depended variables or target variables or output variables.
Emno and empexp are input variables or features or independent variables.

Def: Supervised learning is the types of machine learning in which machines are trained using well "labelled" training data, and on basis of that data, machines predict the output. The labelled data means some input data is already tagged with the correct output.

It applies the same concept as a student learns in the supervision of the teacher. Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to **find a mapping function to map the input variable(x) with the output variable(y)**.

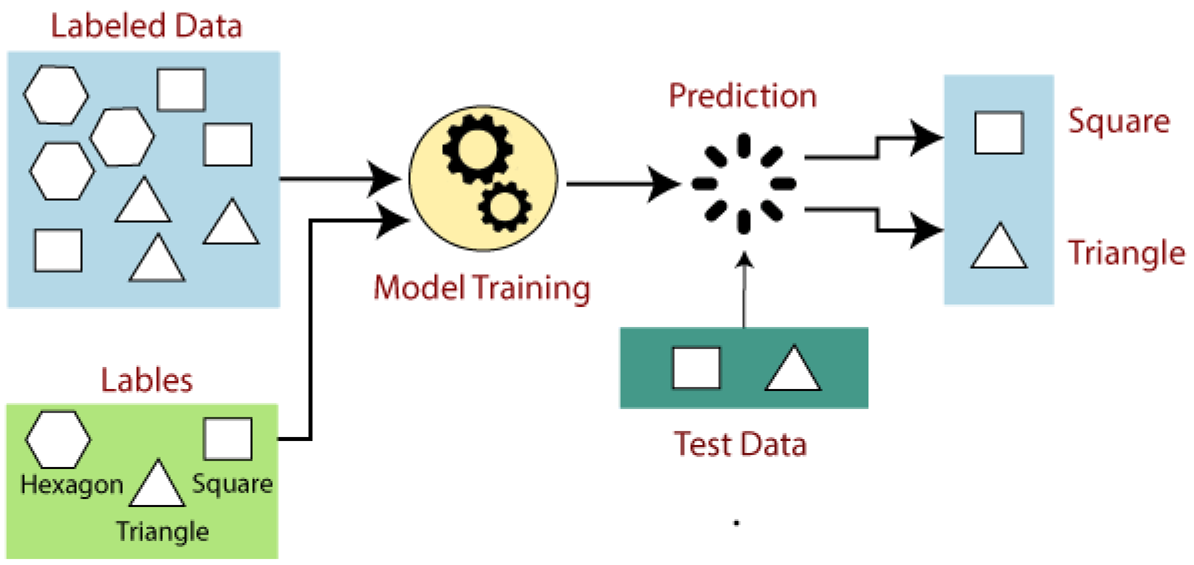
supervised learning can be used for **Risk Assessment, Image classification, Fraud Detection, spam filtering**, etc.

ML will work small dataset. DL will work on Large dataset.

How Supervised Learning Works?

In supervised learning, models are trained using labelled dataset, where the model learns about each type of data. Once the training process is completed, the model is tested on the basis of test data (a subset of the training set), and then it predicts the output.

The working of Supervised learning can be easily understood by the below example and diagram:



Suppose we have a dataset of different types of shapes which includes square, rectangle, triangle, and Polygon. Now the first step is that we need to train the model for each shape.

- If the given shape has four sides, and all the sides are equal, then it will be labelled as a **Square**.
- If the given shape has three sides, then it will be labelled as a **triangle**.
- If the given shape has six equal sides then it will be labelled as **hexagon**.

Now, after training, we test our model using the test set, and the task of the model is to identify the shape.

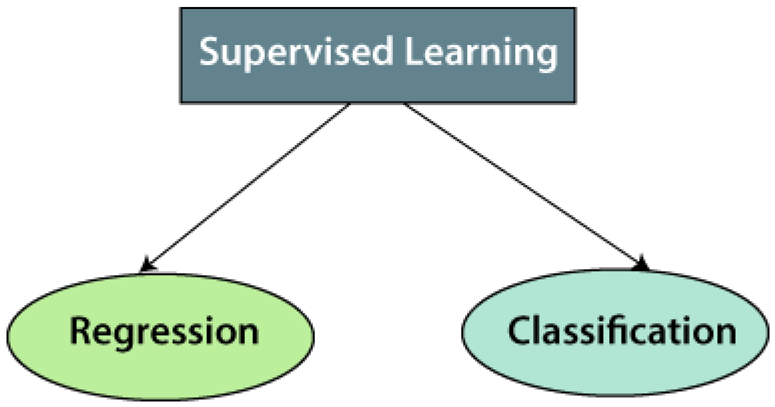
The machine is already trained on all types of shapes, and when it finds a new shape, it classifies the shape on the bases of a number of sides, and predicts the output.

Steps Involved in Supervised Learning:

- First Determine the type of training dataset
- Collect/Gather the labelled training data.
- Split the training dataset into training **dataset, test dataset, and validation dataset**.
- Determine the input features of the training dataset, which should have enough knowledge so that the model can accurately predict the output.
- Determine the suitable algorithm for the model, such as support vector machine, decision tree, etc.
- Execute the algorithm on the training dataset. Sometimes we need validation sets as the control parameters, which are the subset of training datasets.
- Evaluate the accuracy of the model by providing the test set. If the model predicts the correct output, which means our model is accurate.

Types of supervised Machine learning Algorithms:

Supervised learning can be further divided into two types of problems:



1. Regression

Regression algorithms are used if there is a relationship between the input variable and the output variable. It is used for the prediction of continuous variables, such as Weather forecasting, Market Trends, etc. Below are some popular Regression algorithms which come under supervised learning:

- Linear Regression
- Regression Trees
- Non-Linear Regression
- Bayesian Linear Regression
- Polynomial Regression

House price prediction based on features

Area	no of rooms	cost of house
150.6	2	15
200	3	20.5

2. Classification

Classification algorithms are used when the output variable/dependent is categorical/numerical, which means there are two classes such as Yes-No, Male-Female, True-false,0 or 1 etc.

Smarks	grade
98	p

75	p
32	f

Spam Filtering, credit card detection

Classification are two types: binary classification and multi classification

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

Advantages of Supervised learning:

- With the help of supervised learning, the model can predict the output on the basis of prior experiences.
- In supervised learning, we can have an exact idea about the classes of objects.
- Supervised learning model helps us to solve various real-world problems such as **fraud detection, spam filtering**, etc.

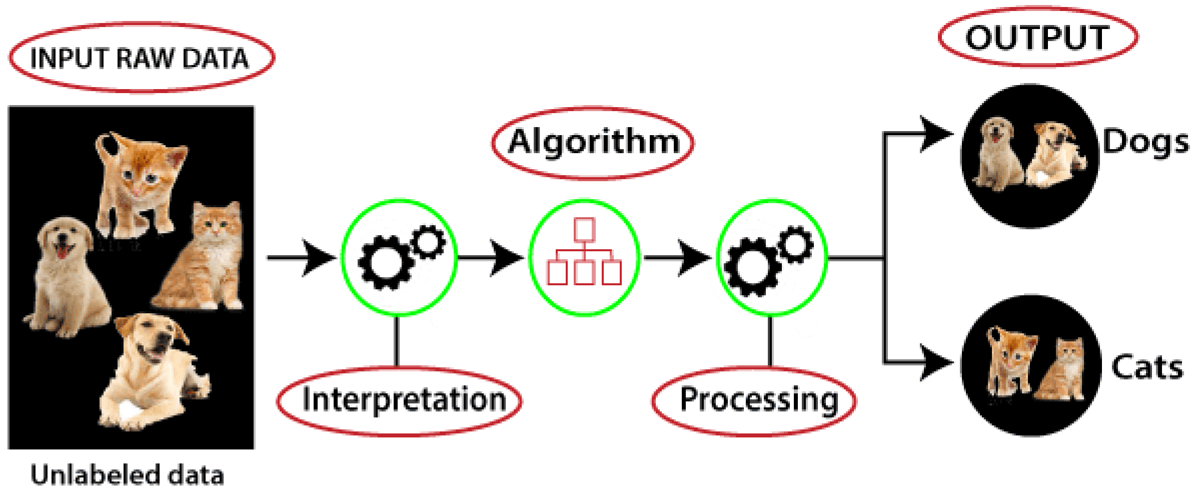
Disadvantages of supervised learning:

- Supervised learning models are not suitable for handling the complex tasks.
- Supervised learning cannot predict the correct output if the test data is different from the training dataset.
- Training required lots of computation times.
- In supervised learning, we need enough knowledge about the classes of object.

Unsupervised learning:

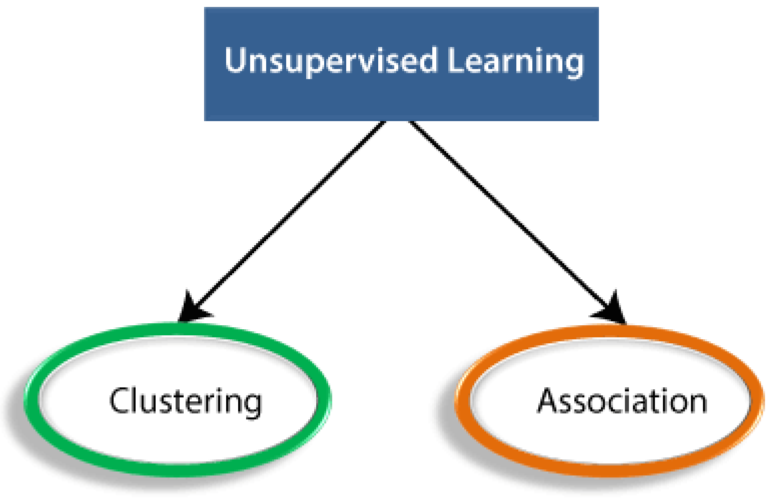
Unsupervised learning is a type of machine learning in which models are trained using unlabeled dataset . **find the underlying structure of dataset, group that data according to similarities.** Suppose the unsupervised learning algorithm is given an input dataset containing images of different types of cats and dogs. Unsupervised learning algorithm will perform this task by clustering the image dataset into the groups according to similarities between images.

Working of unsupervised learning can be understood by the below diagram:



Here, we have taken an unlabeled input data, which means it is not categorized and corresponding outputs are also not given. Now, this unlabeled input data is fed to the machine learning model in order to train it.

The unsupervised learning algorithm can be further categorized into two types of problems:



- **Clustering**: Clustering is a method of grouping the objects into clusters such that objects with most similarities remains into a group and has less or no similarities with the objects of another group.

- **Association**: An association rule is an unsupervised learning method which is used for finding the relationships between variables in the large database. It determines the set of items that occurs together in the dataset. Association rule makes marketing strategy more effective. Such as people who buy X item (suppose a bread) are also tend to purchase Y (Butter/Jam) item. A typical example of Association rule is Market Basket Analysis.

Unsupervised Learning algorithms:

Below is the list of some popular unsupervised learning algorithms:

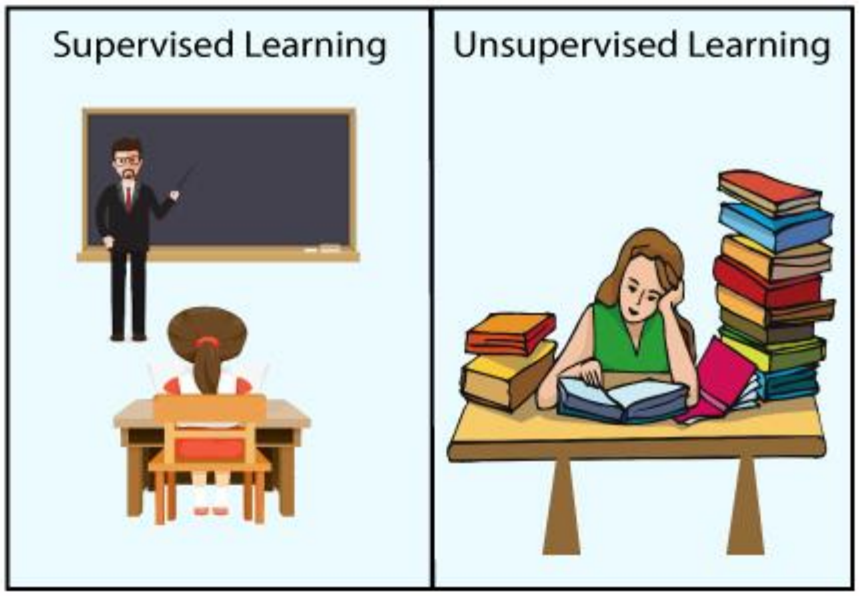
- K-means clustering
- KNN (k-nearest neighbors)
- Hierarchal clustering
- Anomaly detection
- Neural Networks
- Principle Component Analysis
- Independent Component Analysis
- Apriori algorithm
- Singular value decomposition

Advantages of Unsupervised Learning

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

Disadvantages of Unsupervised Learning

- Unsupervised learning is intrinsically more difficult than supervised learning as it does not have corresponding output.
- The result of the unsupervised learning algorithm might be less accurate as input data is not labeled, and algorithms do not know the exact output in advance.



https://drive.google.com/drive/folders/1DnDnzOUYsaCSFsp1Rbi1vBsIJvbjP98Y

Training a Pattern Recognition system

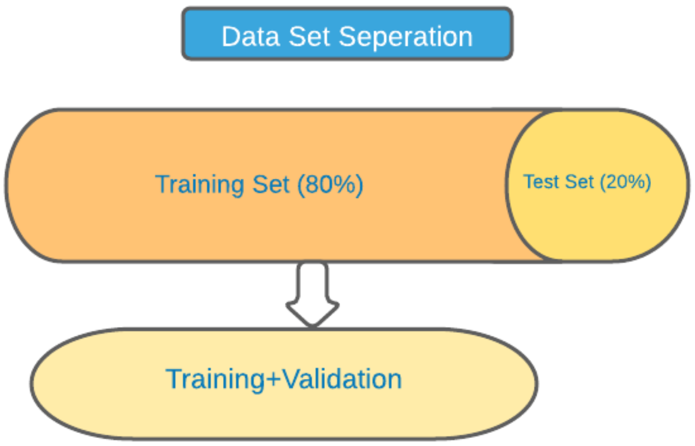
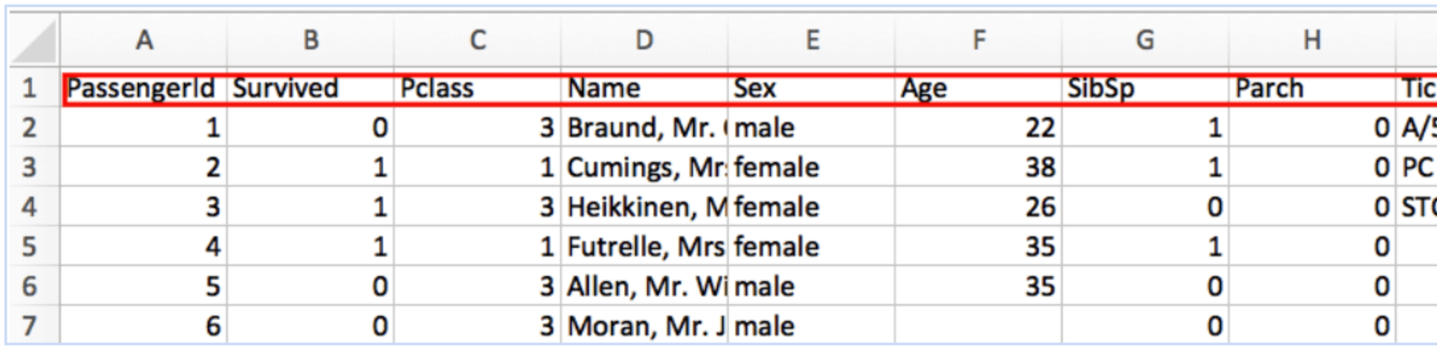


Fig (1): Dataset

Features may be represented as continuous, discrete or discrete binary variables. A feature is a function of one or more measurements, computed so that it quantifies some significant characteristics of the object.
Example: consider our face then eyes, ears, nose etc are features of the face.
A set of features that are taken together, forms the **features vector**.

A feature is a measurable property of the object you're trying to analyze. In datasets, features appear as columns:



	A	B	C	D	E	F	G	H	
1	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Tic
2	1	0	3	Braund, Mr. (	male	22	1	0	A/
3	2	1	1	Cumings, Mr	female	38	1	0	PC
4	3	1	3	Heikkinen, M	female	26	0	0	ST
5	4	1	1	Futrelle, Mrs	female	35	1	0	
6	5	0	3	Allen, Mr. Wi	male	35	0	0	
7	6	0	3	Moran, Mr. J	male		0	0	

Each feature, or column, represents a measurable piece of data that can be used for analysis: Name, Age, Sex, Fare, and so on. Features are also sometimes referred to as "variables" or "attributes." Depending on what you're trying to analyze, the features you include in your dataset can vary widely.

Why are Feature Variables Important?

Features are the basic building blocks of datasets. The quality of the features in your dataset has a major impact on the quality of the insights you will gain when you use that dataset for machine learning. different business problems within the same industry do not necessarily require the same features. You can improve the quality of your dataset's features with processes like feature selection and feature engineering.. If these techniques are done well, the resulting optimal dataset will contain all of the essential features that might have leading to the best possible model outcomes and the most beneficial insights.

Example: In the above example of face, if all the features (eyes, ears, nose etc) taken together then the sequence is feature vector([eyes, ears, nose]). Feature vector is the sequence of a features represented as a d-dimensional column vector. Sequence of first 13 features forms a feature vector.

Pattern representation:

What are classes?

- The patterns belong to two or more classes.
- The task of pattern recognition pertains to finding the class to whicha pattern belongs.

What is classification?

- Given a pattern, the task of identifying the class to which the pattern belongs is called classification.
- Generally, a set of patterns is given where the class label of each pattern is known. This is known as the training data.
- The information in the training data should be used to identify the class of the test pattern.

Representation of patterns

- Patterns can be represented in a number of ways.

Representing patterns as vectors

- The most popular method of representing patterns is as vectors.
- Here, the training dataset may be represented as a matrix of size (nxd), where each row corresponds to a pattern and each column representsa feature.
- Each attribute/feature/variable is associated with a domain. A domainis a set of numbers, each number pertains to a value of an attribute forthat particular pattern.
- The class label is a dependent attribute which depends on the ‘d’ in- dependent attributes.

Example

The dataset could be as follows :

	f_1	f_2	f_3	f_4	f_5	f_6	Class label
Pattern 1:	1	4	3	6	4	7	1
Pattern 2:	4	7	5	7	4	2	2
Pattern 3:	6	9	7	5	3	1	3
Pattern 4:	7	4	6	2	8	6	1
Pattern 5:	4	7	5	8	2	6	2
Pattern 6:	5	3	7	9	5	3	3
Pattern 7:	8	1	9	4	2	8	3

In this case, n=7 and d=6. As can be seen,each pattern has six attributes(or features). Each attribute in this case is a number between 1 and 9. The last number in each line gives the class of the pattern. In this case, the class of the patterns is either 1, 2 or 3.

Representing patterns as strings

- Here each pattern is a string of characters from an alphabet.
- This is generally used to represent gene expressions.
- For example, DNA can be represented as

  GTGCATCTGACTCCT...

  RNA is expressed as
  GUGCAUCUGACUCCU....

  This can be translated into protein which would be of the form
  VHLTPEEK ....
- Each string of characters represents a pattern. Operations like pattern matching or finding the similarity between strings are carried out with these patterns.

Representing patterns by using logical operators

- Here each pattern is represented by a sentence(well formed formula) ina logic.
- An example would be
  if (beak(x) = red) and (colour(x) = green) then parrot(x)
  This is a rule where the antecedent is a conjunction of primitives andthe consequent is the class label.
- Another example would be
  if (has-trunk(x)) and (colour(x) = black) and (size(x) = large) then elephant(x)

Representing patterns using fuzzy

- The features in a fuzzy pattern may consist of linguistic values, fuzzy numbers and intervals.
- For example, linguistic values can be like tall, medium, short for heightwhich is very subjective and can be modelled by fuzzy membership values.
- A feature in the pattern maybe represented by an interval instead of asingle number. This would give a range in which that feature falls. An example of this would be the pattern

  (3, *small*, 6.5, [1, 10])

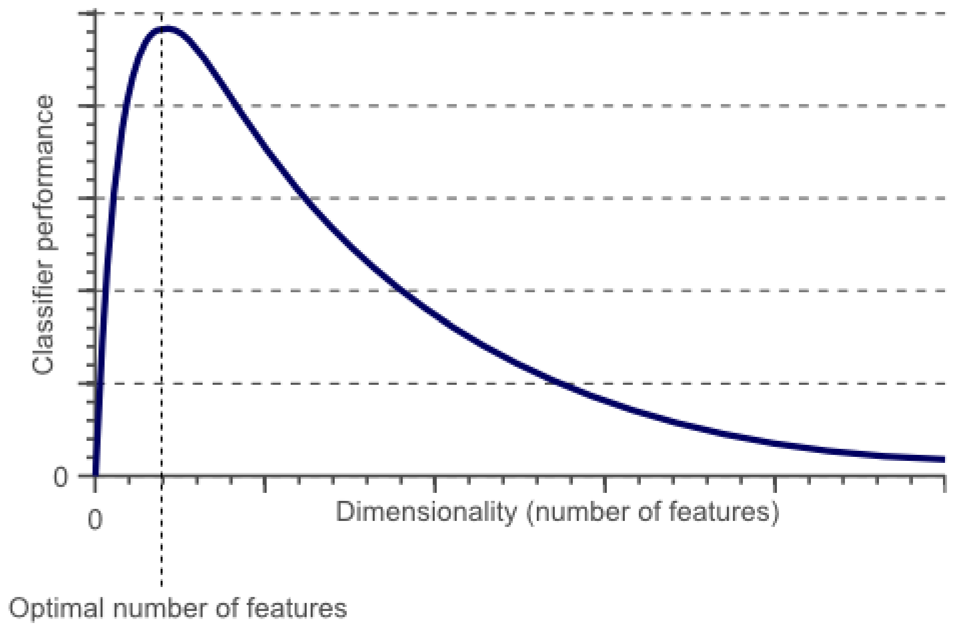
The above example gives a pattern with 4 features. The 4th featureis in the form of an interval. In this case the feature falls within the range 1 to 10. This is also used when there are missing values. When a particular feature of a pattern is missing, looking at other patterns, we can find a range of values which this feature can take. This can be represented as an interval.

The curse of dimensionality: (image pixels: picture element .

If we're analyzing grayscale images sized 50x50, we're working in a space with 2,500 dimensions. If the images are RGB-colored, the dimensionality increases to 7,500 dimensions (one dimension for each color channel in each pixel in the image).

data has too many features.

As the number of features increases, the classifier's performance increases as well until we reach the optimal number of features. Adding more features based on the same size as the training set will then degrade the classifier's performance.



The dimension of a dataset corresponds to the number of attributes/features that exist in a dataset. A dataset with a large number of attributes, generally of the order of a hundred or more, is referred to as high dimensional data.

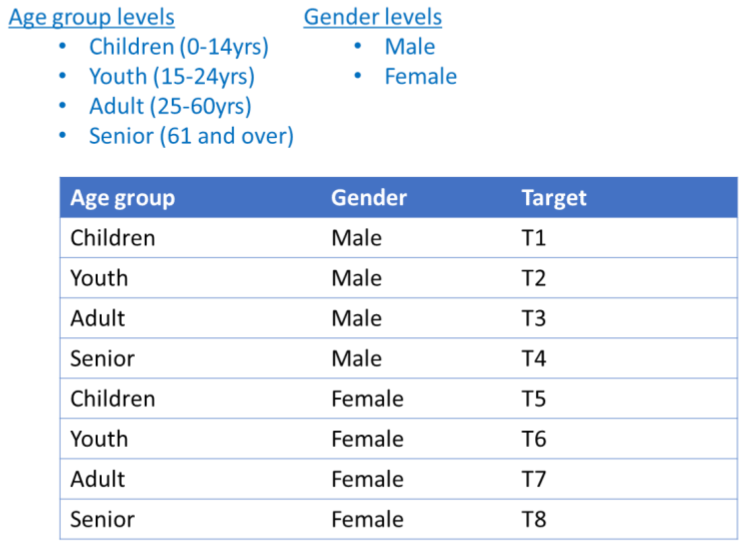
https://www.mygreatlearning.com/blog/understanding-curse-of-dimensionality/

the difficulties that come with high dimensional data during analyzing or visualizing the data to identify patterns. The difficulties related to training machine learning models due to high dimensional data is referred to as 'Curse of Dimensionality'. The problems with curse of dimensionality; 'data sparsity' and 'distance concentration'.

Data Sparsity

While training a model, the available data is used such that part of the data is used for training the model, and a part of the data is used for testing. The testing step helps us establish whether the model is generalized or not.

For instance, if we are trying to predict a target, that is dependent on two attributes: gender and age group, we should ideally capture the targets for all possible combinations of values for the two attributes as shown in figure 1. As long as the future unseen data comes from this distribution (a combination of values), the model w ould predict the target accurately.



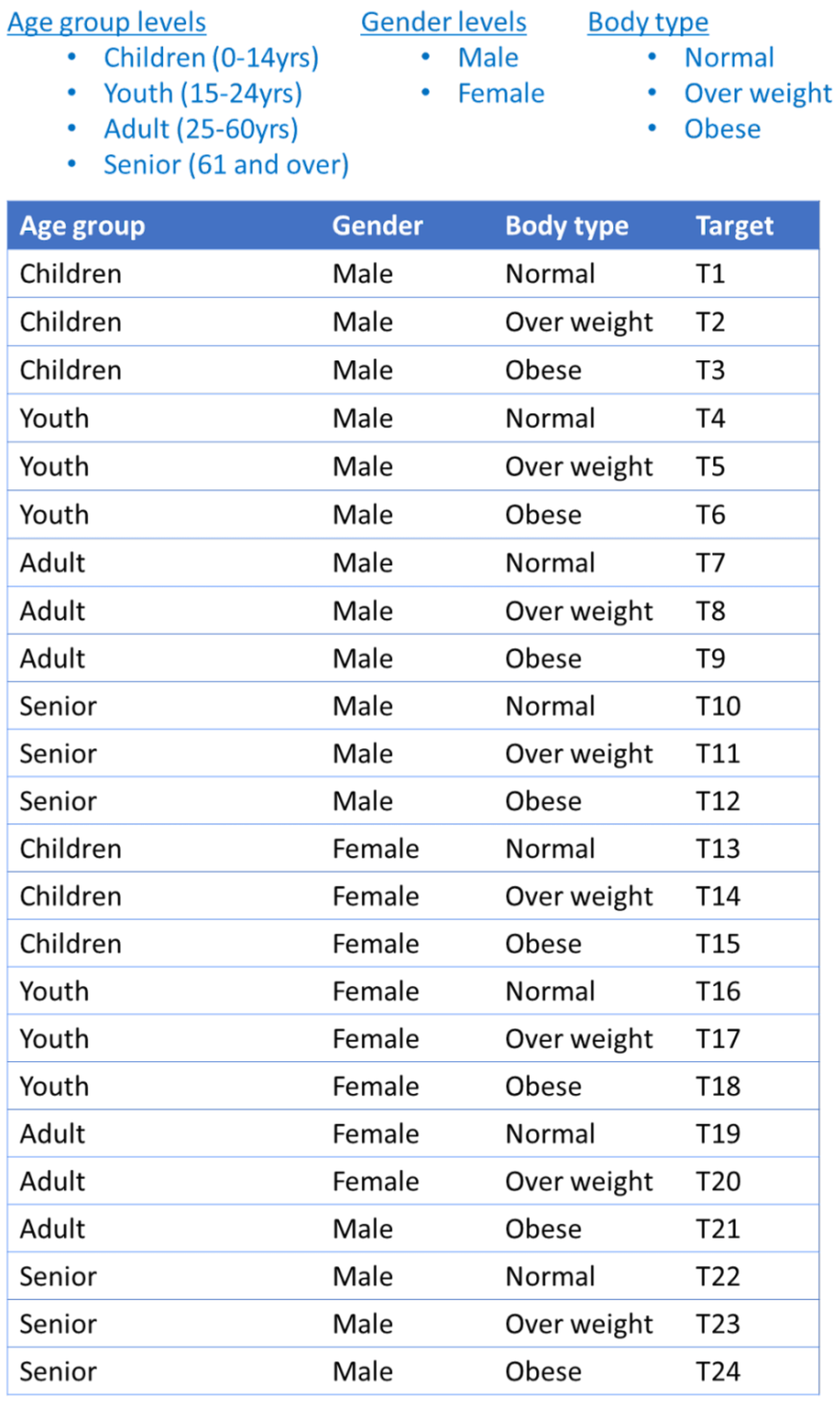
Age group levels
- Children (0-14yrs)
- Youth (15-24yrs)
- Adult (25-60yrs)
- Senior (61 and over)

Gender levels
- Male
- Female

Age group	Gender	Target
Children	Male	T1
Youth	Male	T2
Adult	Male	T3
Senior	Male	T4
Children	Female	T5
Youth	Female	T6
Adult	Female	T7
Senior	Female	T8

Figure 1. Combination of values of 2 attributes for generalizing a model

In the above example, we assume that the target value depends on gender and age group only. If the target depends on a third attribute, let's say body type, the number of training samples required to cover all the combinations increases phenomenally. The combinations are shown in figure 2. For two variables, we needed eight training samples. For three variables, we need 24 samples.



Age group levels
- Children (0-14yrs)
- Youth (15-24yrs)
- Adult (25-60yrs)
- Senior (61 and over)

Gender levels
- Male
- Female

Body type
- Normal
- Over weight
- Obese

Age group	Gender	Body type	Target
Children	Male	Normal	T1
Children	Male	Over weight	T2
Children	Male	Obese	T3
Youth	Male	Normal	T4
Youth	Male	Over weight	T5
Youth	Male	Obese	T6
Adult	Male	Normal	T7
Adult	Male	Over weight	T8
Adult	Male	Obese	T9
Senior	Male	Normal	T10
Senior	Male	Over weight	T11
Senior	Male	Obese	T12
Children	Female	Normal	T13
Children	Female	Over weight	T14
Children	Female	Obese	T15
Youth	Female	Normal	T16
Youth	Female	Over weight	T17
Youth	Female	Obese	T18
Adult	Female	Normal	T19
Adult	Female	Over weight	T20
Adult	Male	Obese	T21
Senior	Male	Normal	T22
Senior	Male	Over weight	T23
Senior	Male	Obese	T24

Figure 2.

Combination of values of 3 attributes for generalizing a model

The above examples show that, as the number of attributes or the dimensions increases, the number of training samples required to generalize a model also increase phenomenally.

The available training samples may not have observed targets for all combinations of the attributes. This is because some combination occurs more often than others. Due to this, the training samples available for building the model may not capture all possible combinations. This aspect, where the training samples do not capture all combinations, is referred to as ‘**Data sparsity**’ or simply ‘**sparsity’** in high dimensional data. Data sparsity is one of the facets of the curse of dimensionality. Training a model with sparse data could lead to high-variance or overfitting condition. This is because while training the model, the model has learnt from the frequently occurring combinations of the attributes and can predict the outcome accurately. In real-time when less frequently occurring combinations are fed to the model, it may not predict the outcome accurately.

Dimensionality reduction:

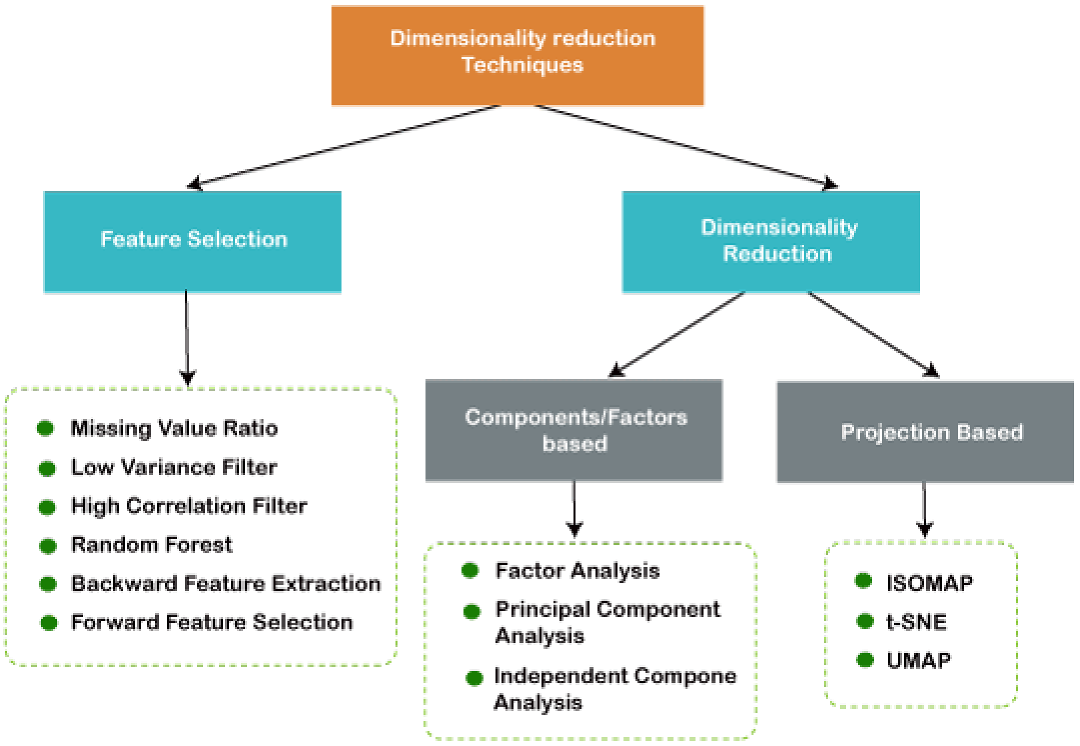
The number of input variables or features for a dataset is referred to as its dimensionality. Dimensionality reduction refers to techniques that reduce the number of input variables in a dataset. More input features often make a predictive modeling task more challenging to model, more generally referred to as the curse of dimensionality. Large numbers of input features can cause poor performance.

The performance of machine learning algorithms can degrade with too many input variables. If your data is represented using rows and columns, such as in a spreadsheet, then the input variables are the columns that are fed as input to a model to predict the target variable. Input variables are also called features.

Dimensionality reduction refers to techniques for reducing the number of input variables in training data. Dimensionality reduction is a data preparation technique performed on data prior to modeling.

The various methods used for dimensionality reduction include:

- Principal Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)
- Generalized Discriminant Analysis (GDA)



Feature Selection

Feature selection is the process of selecting the subset of the relevant features and leaving out the irrelevant features present in a dataset to build a model of high accuracy.

Approaches of Dimension Reduction

There are two ways to apply the dimension reduction technique, which are given below:

Feature Selection

Feature selection is the process of selecting the subset of the relevant features and leaving out the irrelevant features present in a dataset to build a model of high accuracy. In other words, it is a way of selecting the optimal features from the input dataset.

Three methods are used for the feature selection:

1. Filters Methods

2. Wrappers Methods

3. Embedded Methods

Feature Extraction:

Feature extraction is the process of transforming the space containing many dimensions into space with fewer dimensions. This approach is useful when we want to keep the whole information but use fewer resources while processing the information.

feature extraction techniques are:

a. Principal Component Analysis
b. Linear Discriminant Analysis
c. Kernel PCA
d. Quadratic Discriminant Analysis

Classification—linear and non-linear:

Supervised Machine Learning algorithm can be broadly classified into Regression and Classification Algorithms. In Regression algorithms, we have predicted the output for continuous values, but to predict the categorical values, we need Classification algorithms.

The Classification algorithm is a Supervised Learning technique that is used to identify the category of new observations on the basis of training data. Such as, **Yes or No, 0 or 1, Spam or Not Spam, cat or dog,** etc. Classes can be called as targets/labels or categories.

The algorithm which implements the classification on a dataset is known as a classifier. There are two types of Classifications:

- **Binary Classifier:** If the classification problem has only two possible outcomes, then it is called as Binary Classifier.
  Examples: YES or NO, MALE or FEMALE, SPAM or NOT SPAM, CAT or DOG, etc.
- **Multi-class Classifier:** If a classification problem has more than two outcomes, then it is called as Multi-class Classifier.
  Example: Classifications of types of crops, Classification of types of music.

Classification Algorithms can be further divided into the Mainly two category:

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

Confusion Matrix:

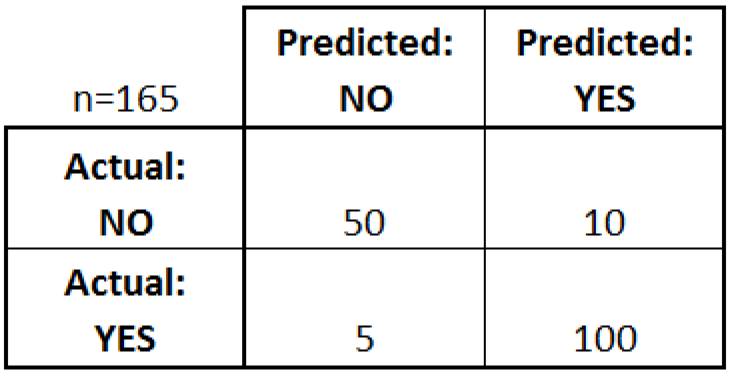
- The confusion matrix provides us a matrix/table as output and describes the performance of the model.
- It is also known as the error matrix.
- The matrix consists of predictions result in a summarized form, which has a total number of correct predictions and incorrect predictions. The matrix looks like as below table:

	Actual Positive	**Actual Negative**
Predicted Positive	True Positive-TP	False Positive-FP
Predicted Negative	False Negative-FN	True Negative-TN

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{Total Population}}$$

confusion matrix for a binary classifier

:

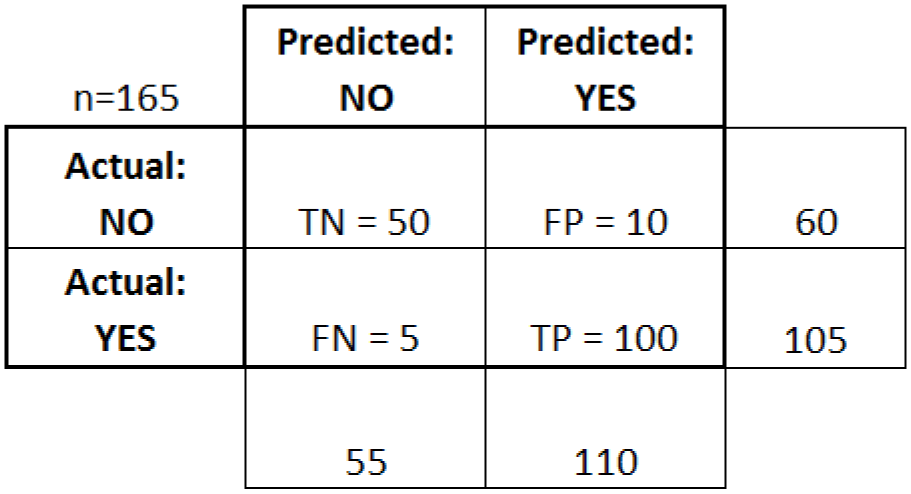


n=165	Predicted: NO	Predicted: YES
Actual: NO	50	10
Actual: YES	5	100

- There are two possible predicted classes: "yes" and "no". If we were predicting the presence of a disease, for example, "yes" would mean they have the disease, and "no" would mean they don't have the disease.
- The classifier made a total of 165 predictions (e.g., 165 patients were being tested for the presence of that disease).
- Out of those 165 cases, the classifier predicted "yes" 110 times, and "no" 55 times.
- In reality, 105 patients in the sample have the disease, and 60 patients do not.

- **true positives (TP):** These are cases in which we predicted yes (they have the disease), and they do have the disease.
- **true negatives (TN):** We predicted no, and they don't have the disease.
- **false positives (FP):** We predicted yes, but they don't actually have the disease. (Also known as a "Type I error.")
- **false negatives (FN):** We predicted no, but they actually do have the disease. (Also known as a "Type II error.")

added the row and column totals:



n=165	Predicted: NO	Predicted: YES	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

This is a list of rates that are often computed from a confusion matrix for a binary classifier:

- **Accuracy:** Overall, how often is the classifier correct?
    - (TP+TN)/total = (100+50)/165 = 0.91
- **Misclassification Rate:** Overall, how often is it wrong?
    - (FP+FN)/total = (10+5)/165 = 0.09
    - equivalent to 1 minus Accuracy
    - also known as "Error Rate"
- **True Positive Rate:** When it's actually yes, how often does it predict yes?
    - TP/actual yes = 100/105 = 0.95
    - also known as "Sensitivity" or "Recall"
- **False Positive Rate:** When it's actually no, how often does it predict yes?
    - FP/actual no = 10/60 = 0.17
- **True Negative Rate:** When it's actually no, how often does it predict no?
    - TN/actual no = 50/60 = 0.83
    - equivalent to 1 minus False Positive Rate
    - also known as "Specificity"
- **Precision:** When it predicts yes, how often is it correct?
    - TP/predicted yes = 100/110 = 0.91
- **Prevalence:** How often does the yes condition actually occur in our sample?
    - actual yes/total = 105/165 = 0.64

Cross Validation:

Cross-Validation is **a statistical method of evaluating and comparing learning algorithms by dividing data into two segments**: one used to learn or train a model and the other used to validate the model.

Whenever we change the random_state parameter present in train_test_split(), We get different accuracy for different random_state and hence we can't exactly point out the accuracy for our model.

Steps:

1. Reserve some portion of sample data-set.
2. Using the rest data-set train the model.
3. Test the model using the reserve portion of the data-set.

Ex: we perform training on the 50% of the given data-set and rest 50% is used for the testing purpose. The major drawback of this method is that we perform training on the 50% of the dataset, it may possible that the remaining 50% of the data contains some important information which we are leaving while training our model i.e higher bias.

LOOCV (Leave One Out Cross Validation)
In this method, we perform training on the whole data-set but leaves only one data-point of the

100 SAMPLES

1^{-}SAMPLE –TESTING, 2-100-TRAINING

1,3-100-TRAING, 2-TESTING

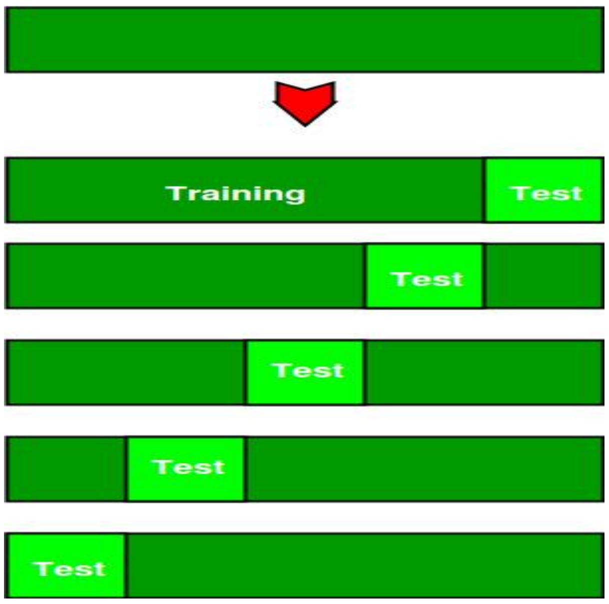
1-2,4-100, 3-testing

1-3,5-100, 4-testinhg

available data-set and then iterates for each data-point. It has some advantages as well as disadvantages also.
An advantage of using this method is that we make use of all data points and hence it is low bias. The major drawback of this method is that it leads to higher variation in the testing model as we are testing against one data point. If the data point is an outlier it can lead to higher variation. Another drawback is it takes a lot of execution time as it iterates over 'the number of data points' times.

K-Fold Cross Validation
In this method, we split the data-set into k number of subsets(known as folds) then we perform training on the all the subsets but leave one(k-1) subset for the evaluation of the trained model. In this method, we iterate k times with a different subset reserved for testing purpose each time.
Note:
It is always suggested that the value of k should be 10 as the lower value
of k is takes towards validation and higher value of k leads to LOOCV

The diagram below shows an example of the training subsets and evaluation subsets generated in k-fold cross-validation. Here, we have total 25 instances. In first iteration we use the first 20 percent of data for evaluation, and the remaining 80 percent for training([1-5] testing and [5-25] training) while in the second iteration we use the second subset of 20 percent for evaluation, and the remaining three subsets of the data for training([5-10] testing and [1-5 and 10-25] training), and so on.



```
Total instances: 25

Value of k      : 5


No. Iteration                 Training set observations
Testing set observations

 1       [ 5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24]
[0 1 2 3 4]

 2       [ 0  1  2  3  4 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24]
[5 6 7 8 9]

 3       [ 0  1  2  3  4  5  6  7  8  9 15 16 17 18 19 20 21 22 23 24]
[10 11 12 13 14]

 4       [ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 20 21 22 23 24]
[15 16 17 18 19]

 5       [ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19]
[20 21 22 23 24]
```

Advantages of train/test split:

1. This runs K times faster than Leave One Out cross-validation because K-fold cross-validation repeats the train/test split K-times.

Stratified K-Fold Cross Validation:

Suppose you want to take a survey and decided to call 1000 people from a particular state, If you pick either 1000 male completely or 1000 female completely or 900 female and 100 male (randomly) to ask their opinion on a particular product. Then based on these 1000 opinion you can't decide the opinion of that entire state on your product.
in Stratified Sampling, Let the population for that state be 51.3% male and 48.7% female, Then for choosing 1000 people from that state if you pick 531 male (51.3% of 1000) and 487 female (48.7% for 1000) i.e 531 male + 487 female (Total=1000 people) to ask their opinion. Then these groups of people represent the entire state. This is called as Stratified Sampling.

Class-imbalance – ways of handling

Imbalance means that the number of data points available for different the classes is different: If there are two classes, then balanced data would mean 50% points for each of the class. For most machine learning techniques, little imbalance is not a problem. So, if there are 60% points for one class and 40% for the other class, it should not cause any significant performance degradation. Only when the class imbalance is high, e.g. 90% points for one class and 10% for the other, standard optimization criteria or performance measures may not be as effective and would need modification.
example of imbalanced data is encountered in e-mail classification problem where emails are classified into ham or spam. The number of spam emails is usually lower than the number of relevant (ham) emails. So, using the original distribution of two classes leads to imbalanced dataset.

Fraud detection

Total Observations = 1000

Fraudulent Observations = 20

Non Fraudulent Observations = 980

Event Rate= 2 %=20/1000*100

Fraud Indicator = 0 for Non-Fraud Instances

Fraud Indicator = 1 for Fraud

Techniques for handling imbalanced datasets

Re-Sampling Technique

In this strategy we focus on balancing the classes in the training data (data preprocessing) before providing the data as input to the machine learning algorithm.

Random Under-Sampling

Random Undersampling aims to balance class distribution by randomly eliminating majority class examples. This is done until the majority and minority class instances are balanced out.

Total Observations = 1000

Fraudulent Observations =20

Non Fraudulent Observations = 980

Event Rate= 2 %

In this case we are taking 10 % samples without replacement from Non Fraud instances. And combining them with Fraud instances.

Non Fraudulent Observations after random under sampling = 10 % of 980 =98

Total Observations after combining them with Fraudulent observations = 20+98=118

Event Rate for the new dataset after under sampling = 20/118 = 17%

Advantages

- It can help improve run time and storage problems by reducing the number of training data samples when the training data set is huge.

- **Disadvantages**
  - It can discard potentially useful information which could be important for building rule classifiers.
  - The sample chosen by random under sampling may be a biased sample. And it will not be an accurate representative of the population. Thereby, resulting in inaccurate results with the actual test data set.

Random Over-Sampling

Over-Sampling increases the number of instances in the minority class by randomly replicating them in order to present a higher representation of the minority class in the sample.

Total Observations = 1000

Fraudulent   Observations =20

Non Fraudulent Observations = 980

Event Rate= 2 %

In this case we are replicating 20 fraud observations   20 times.

Non Fraudulent Observations =980

Fraudulent Observations after replicating the minority class observations= 400

Total Observations in the new data set after oversampling=1380

Error rate=minority/total

=400/1000+380

Event Rate for the new data set after under sampling= 400/1380 = 29 %

- **Advantages**
  - Unlike under sampling this method leads to no information loss.
  - Outperforms under sampling

- **Disadvantages**
  - It increases the likelihood of overfitting since it replicates the minority class events.

Cluster-Based Over Sampling

In this case, the K-means clustering algorithm is independently applied to minority and majority class instances. This is to identify clusters in the dataset. Subsequently, each cluster is oversampled such that all clusters of the same class have an equal number of instances and all classes have the same size.

Total Observations = 1000

Fraudulent   Observations =20

Non Fraudulent Observations = 980

Event Rate= 2 %

- **Majority Class Clusters**
  1. Cluster 1: 150 Observations
  2. Cluster 2: 120 Observations

3. Cluster 3: 230 observations
4. Cluster 4: 200 observations
5. Cluster 5: 150 observations
6. Cluster 6: 130 observations

- **Minority Class Clusters**
  1. Cluster 1: 8 Observations
  2. Cluster 2: 12 Observations

After oversampling of each cluster, all clusters of the same class contain the same number of observations.

- **Majority Class Clusters**
  1. Cluster 1: 170 Observations
  2. Cluster 2: 170 Observations
  3. Cluster 3: 170 observations
  4. Cluster 4: 170   observations
  5. Cluster 5: 170   observations
  6. Cluster 6: 170   observations

- **Minority   Class Clusters**
  1. Cluster 1: 250 Observations
  2. Cluster 2: 250 Observations

Event Rate post cluster based oversampling sampling = 500/ (1020+500) = 33 %

- **Advantages**
  - This clustering technique helps overcome the challenge between class imbalance. Where the number of examples representing positive class differs from the number of examples representing a negative class.
  - Also, overcome challenges within class imbalance, where a class is composed of different sub clusters. And each sub cluster does not contain the same number of examples.

- **Disadvantages**
  - The main drawback of this algorithm, like most oversampling techniques is the possibility of over-fitting the training data.

2.1.4 Informed Over Sampling: Synthetic Minority Over-sampling Technique for imbalanced data

This technique is followed to avoid overfitting which occurs when exact replicas of minority instances are added to the main dataset. A subset of data is taken from the minority class as an example and then new synthetic similar instances are created. These synthetic instances are then added to the original dataset. The new dataset is used as a sample to train the classification models.

Total Observations = 1000

Fraudulent Observations = 20

Non Fraudulent Observations = 980

Event Rate = 2 %

A sample of 15 instances is taken from the minority class and similar synthetic instances are generated 20 times

Post generation of synthetic instances, the following data set is created

Minority Class (Fraudulent Observations) = 300

Majority Class (Non-Fraudulent Observations) = 980

Event rate= 300/1280 = 23.4 %

Advantages

- Mitigates the problem of overfitting caused by random oversampling as synthetic examples are generated rather than replication of instances
- No loss of useful information

- **Disadvantages**
- While generating synthetic examples SMOTE does not take into consideration neighboring examples from other classes. This can result in increase in overlapping of classes and can introduce additional noise
- SMOTE is not very effective for high dimensional data

Association between variables:
Linear regression:y=a+bx
Y is dependent variable
A is y-intercept
M is slope
X is idependent variable
M=y2-y1/x2-x1
Using least square method we find the best fit. We can draw the infinite many lines separating two labels.
A= sigma(xi-xbar)(y1-ybar)/x(x-xbar)square
Correlation:r= [-1,1]

R=0 no linear correlation

Regression:

- **Linear Regression**
- **Logistic Regression**
- **Polynomial Regression**
- **Support Vector Regression**
- **Decision Tree Regression**
- **Random Forest Regression**
- **Ridge Regression**
- **Lasso Regression:**

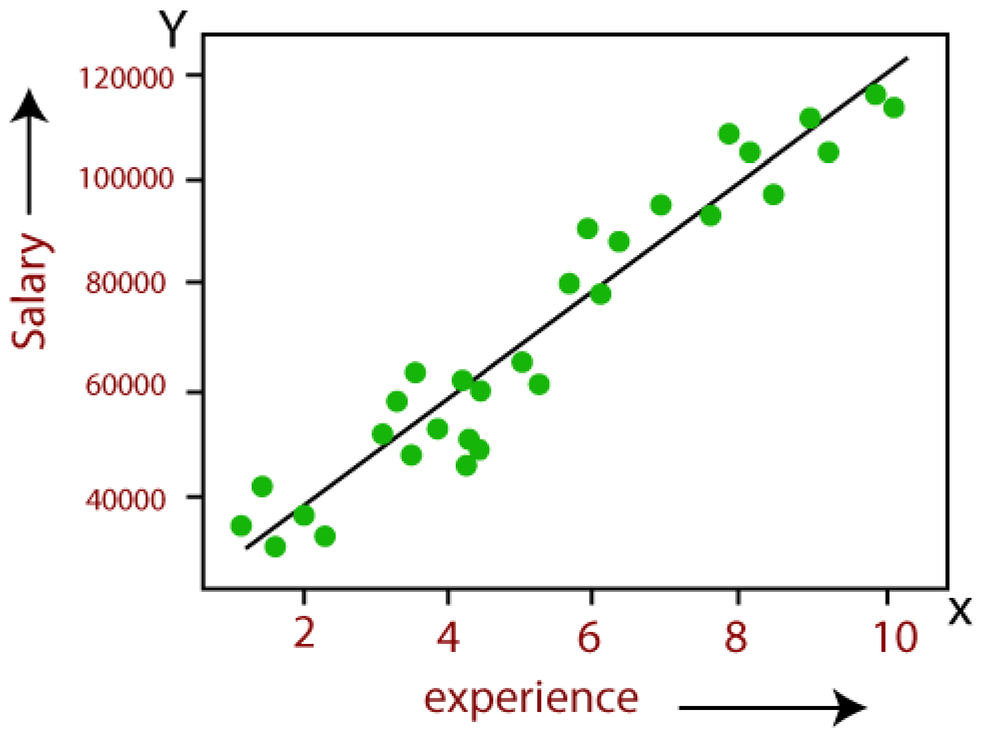
Simple linear regression:
Objective:
1.Relationship between two variables. (positive, negative or no relationship)
Ex:income and wage –positive
Student height and exam score- no relation
2.forecast new observation
What will be the sales over next quarter?
Variable roles:
Dependent variable it is denoted as y. it is the value to be predicted.
Independent variable: values are independent. Denoted by x
Y=mx+b

- If there is only one input variable (x), then such linear regression is called **simple linear regression**. And if there is more than one input variable, then such linear regression is called **multiple linear regression**.

Linear Regression:

- Linear regression is a statistical regression method which is used for predictive analysis.
- It is one of the very simple and easy algorithms which works on regression and shows the relationship between the continuous variables.
- It is used for solving the regression problem in machine learning.

- Linear regression shows the linear relationship between the independent variable (X-axis) and the dependent variable (Y-axis), hence called linear regression.
- If there is only one input variable (x), then such linear regression is called **simple linear regression**. And if there is more than one input variable, then such linear regression is called **multiple linear regression**.
- The relationship between variables in the linear regression model can be explained using the below image. Here we are predicting the salary of an employee on the basis of **the year of experience**.



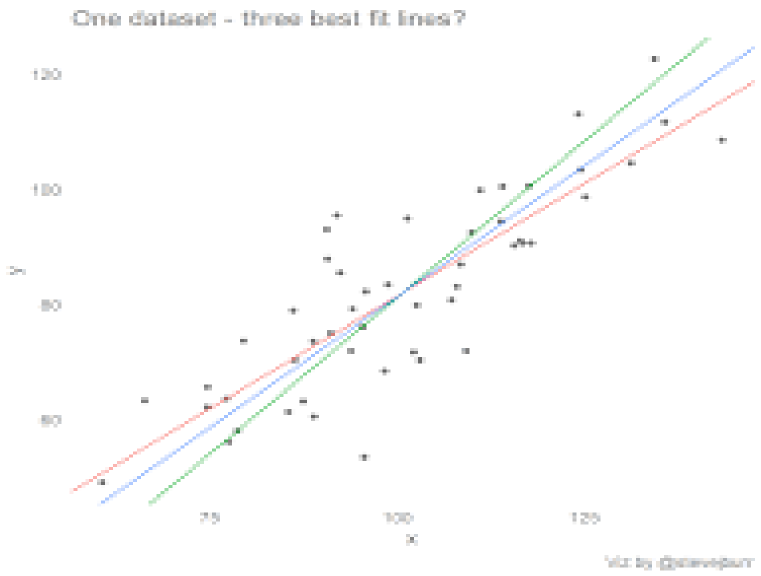
- Below is the mathematical equation for Linear regression:

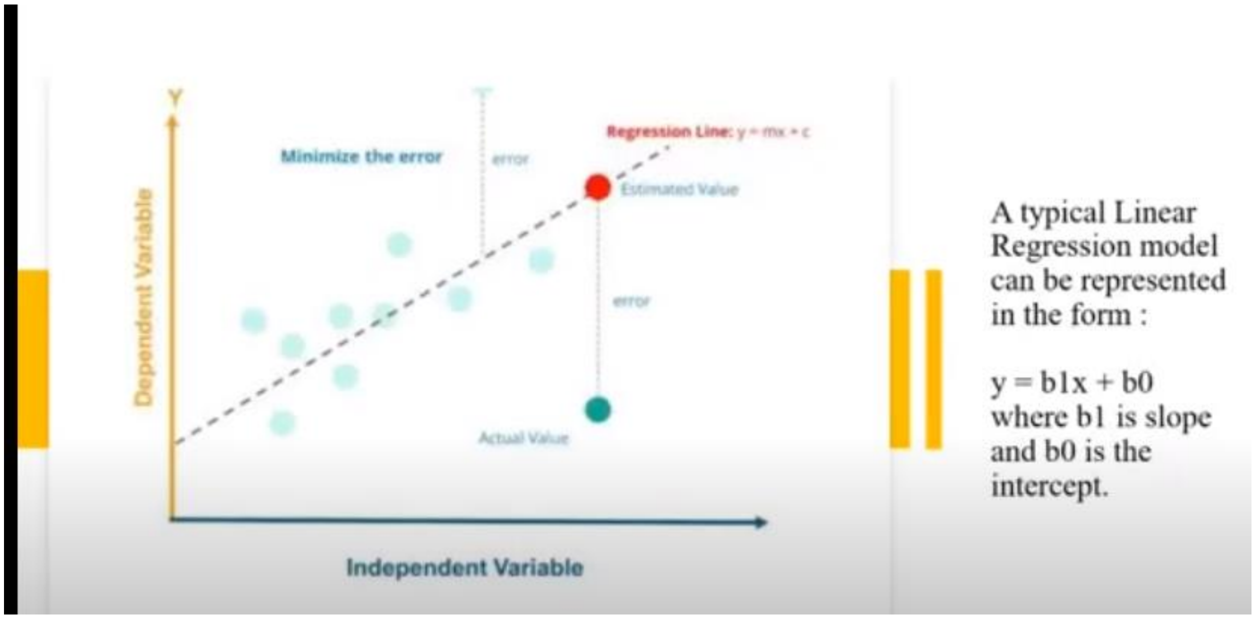
1. Y= aX+b
2. Calculate:a,b
3. X is input
4. Cal y
5.

Here, Y = dependent variables (target variables), X= Independent variables (predictor variables), a and b are the linear coefficients

Some popular applications of linear regression are:

- **Analyzing trends and sales estimates**
- **Salary forecasting**
- **Real estate prediction**
- **Arriving at ETAs in traffic.**





A typical Linear Regression model can be represented in the form :

y = b1x + b0 where b1 is slope and b0 is the intercept.

Simple Linear Regression is a type of Regression algorithms that models the relationship between a dependent variable and a single independent variable. The relationship shown by a Simple Linear Regression model is linear or a sloped straight line, hence it is called Simple Linear Regression.

The key point in Simple Linear Regression is that the ***dependent variable must be a continuous/real value***. However, the independent variable can be measured on continuous or categorical values.

Simple Linear regression algorithm has mainly two objectives:

- **Model the relationship between the two variables.** Such as the relationship between Income and expenditure, experience and Salary, etc.
- **Forecasting new observations.** Such as Weather forecasting according to temperature, Revenue of a company according to the investments in a year, etc.
- $y= a_0+a_1x+ \varepsilon$
- Where,
- **a0= It is the intercept of the Regression line (can be obtained putting x=0)**
  a1= It is the slope of the regression line, which tells whether the line is increasing or decreasing.
  ε = The error term. (For a good model it will be negligible)

Problem Statement example for Simple Linear Regression:

- **We want to find out if there is any correlation between these two variables**

- **We will find the best fit line for the dataset.**
- **How the dependent variable is changing by changing the independent variable.**

Steps:1. preprocessing

- The first step for creating the Simple Linear Regression model is data pre-processing.
- import the three important libraries, which will help us for loading the dataset, plotting the graphs, and creating the Simple Linear Regression model.
- Next, we will split both variables into the test set and training set. We have 30 observations, so we will take 20 observations for the training set and 10 observations for the test set. We are splitting our dataset so that we can train our model using a training dataset and then test the model using a test dataset. The code for this is given below:

2.

WHAT IS R-SQUARED?

R-squared is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression.

The definition of R-squared is fairly straight-forward; it is the percentage of the response variable variation that is explained by a linear model. Or:

R-squared = Explained variation / Total variation

R-squared is always between 0 and 100%:

- 0% indicates that the model explains none of the variability of the response data around its mean.
- 100% indicates that the model explains all the variability of the response data around its mean.

In general, the higher the R-squared, the better the model fits your data.

Multiple Linear Regression

Multiple Linear Regression is one of the important regression algorithms which models the linear relationship between a single dependent continuous variable and more than one independent variable.

- For MLR, the dependent or target variable(Y) must be the continuous/real, but the predictor or independent variable may be of continuous or categorical form.

$$\hat{Y} = b_0 + b_1 X_1 + b_2 X_2 + \ldots + b_p X_p$$

Y= Output/Response variable

b_0, b_1, b_2, b_3 , b_n....= Coefficients of the model.

x_1, x_2, x_3, x_4,...= Various Independent/feature variable

Assumptions for Multiple Linear Regression:

- A **linear relationship** should exist between the Target and predictor variables.
- The regression residuals must be **normally distributed**.
- MLR assumes little or **no multicollinearity** (correlation between the independent variable) in data.

We have a dataset of **50 start-up companies**. This dataset contains five main information: **R&D Spend, Administration Spend, Marketing Spend, State, and Profit for a financial year**. Our goal is to create a model that can easily determine which company has a maximum profit, and which is the most affecting factor for the profit of a company.

Since we need to find the Profit, so it is the dependent variable, and the other four variables are independent variables. Below are the main steps of deploying the MLR model:

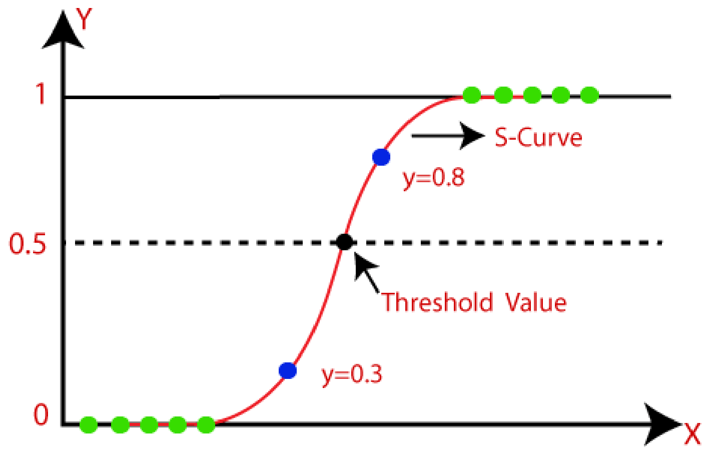
1. **Data Pre-processing Steps**
2. **Fitting the MLR model to the training set**
3. **Predicting the result of the test set**

Logistic Regression in Machine Learning

- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
-

- Binary or Binomial
- In such a kind of classification, a dependent variable will have only two possible types either 1 and 0. For example, these variables may represent success or failure, yes or no, win or loss etc.
- Multinomial
- In such a kind of classification, dependent variable can have 3 or more possible ***unordered*** types or the types having no quantitative significance. For example, these variables may represent "Type A" or "Type B" or "Type C".

- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1**.
- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas **Logistic regression is used for solving the classification problems**.
- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- $$\sigma(\mathrm{x}) = \frac{1}{1 + e^{-\mathrm{x}}}$$
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
- The below image is showing the logistic function:



Logistic Function (Sigmoid Function):

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1.
- The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.

Cost function

It's a function that determines how well a Machine Learning model performs for a given set of data. The Cost Function calculates the difference between anticipated and expected values and shows it as a single real number. The term ‘loss' in machine learning refers to the difference between the anticipated and actual value.

The purpose of Cost Function is to be either:

Minimum- When a value is reduced to its simplest form, it is referred to as a cost, loss, or mistake. The aim is to identify the model parameter settings for which the Cost Function gives the smallest possible number.

Maximum- When something is maximised, the value it produces is referred to as a reward. The aim is to discover model parameter values with as large a returned number as feasible.

It's quite easy to minimise and maximise a function:

(a) Calculate the difference between the function and the parameter and equal to 0,

and (b) Differentiate the function w.r.t the parameter and equate to 0.

- For minimization – the function value of the double differential should be greater than 0.
- For maximization - the function value of the double differential should be less than 0

<u>Types of Cost function</u>

There are many cost functions in machine learning.

1. **<u>Regression cost Function</u>**

Regression models are used to forecast a continuous variable, such as an employee's pay, the cost of a car, the likelihood of obtaining a loan, and so on. The "Regression Cost Function" is a cost function utilised in the regression issue. They are determined as follows depending on the distance-based error:

Error = y-y'

Where, Y – Actual Input and Y' – Predicted output

2. **Binary Classification cost Functions**

The cost functions used in classification problems are not the same as the cost functions used in regression problems. Cross-entropy loss is a popular classification loss function.

For predicting class 1, cross-entropy will compute a score that represents the average difference between the actual and predicted probability distributions. A perfect cross-entropy value is 0 when the score is minimised.

3. **Multi-class Classification cost Functions**

Predictive modelling issues involving multi-class categorization are ones in which instances are allocated to one of more than two classes.

The issue is frequently presented as predicting an integer value, with each class given a different integer value ranging from 0 to (num classes – 1). Predicting the likelihood of an example belonging to each known class is a common way to solve the problem.

K-Means Clustering:

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 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 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 dataset).

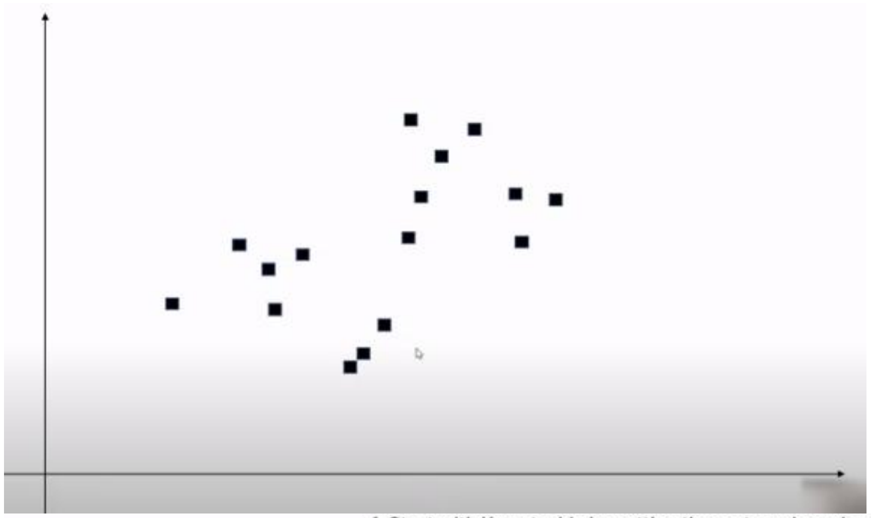
Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.

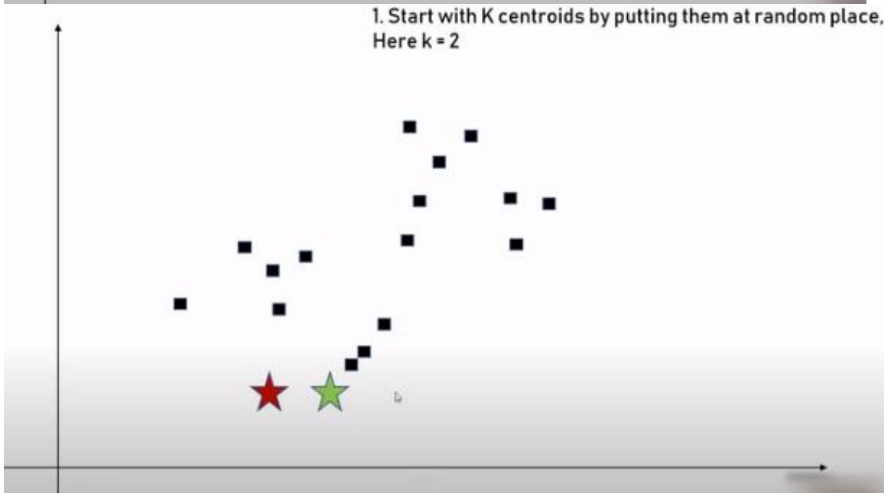
Step-4: Calculate the variance and place a new centroid of each cluster.

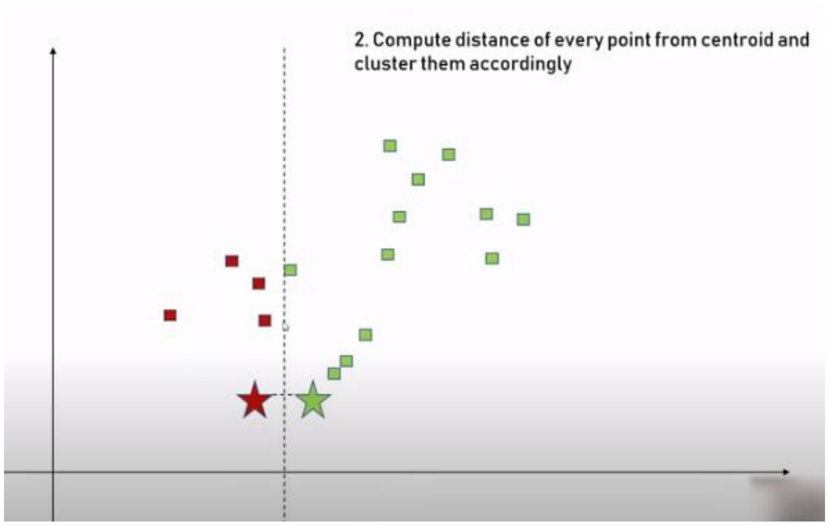
Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

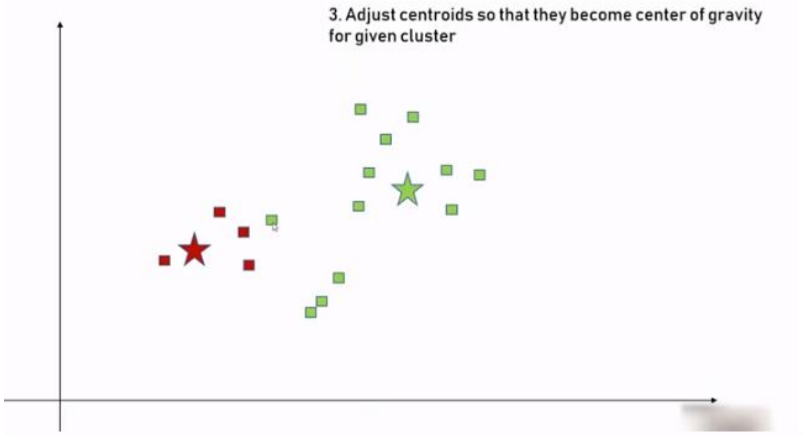
Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.

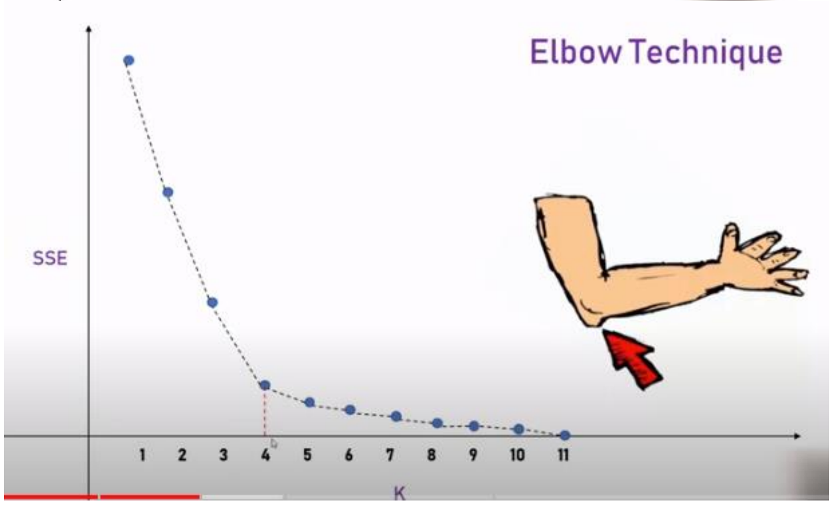
Step-7: The model is ready.






2. Compute distance of every point from centroid and cluster them accordingly


3. Adjust centroids so that they become center of gravity for given cluster

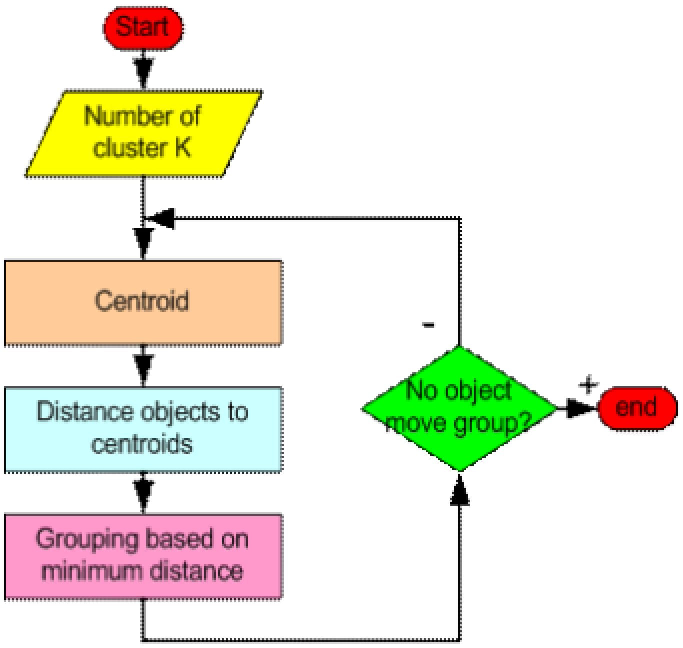

Elbow Technique

K Means Algorithm

Kmeans algorithm is an iterative algorithm that tries to partition the dataset into Kpre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group. It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster's centroid (arithmetic mean of all the data points that belong to that cluster) is at the minimum. The less variation we have within clusters, the more homogeneous (similar) the data points are within the same cluster.

Iterate until *stable* (= no object move group):

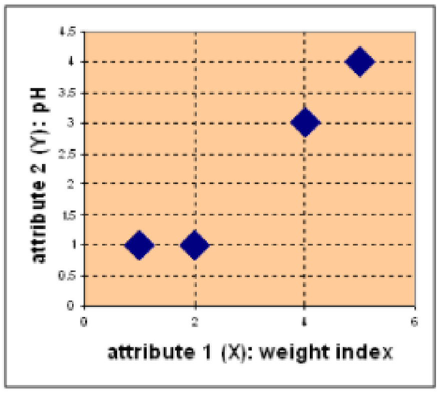
1. Determine the centroid coordinate
2. Determine the distance of each object to the centroids
3. Group the object based on minimum distance


Start
Number of cluster K
Centroid
Distance objects to centroids
Grouping based on minimum distance
No object move group?
-
+
end

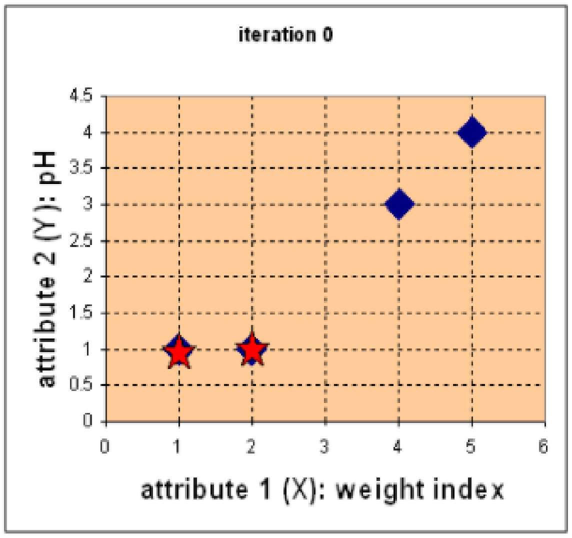
Suppose we have several objects (4 types of medicines) and each object have two attributes or features as shown in table below. Our goal is to group these objects into K=2 group of medicine based on the two features (pH and weight index).

Object	attribute 1 (X): weight index	attribute 2 (Y): pH
Medicine A	1	1
Medicine B	2	1
Medicine C	4	3
Medicine D	5	4

Each medicine represents one point with two attributes (X, Y) that we can represent it as coordinate in an attribute space as shown in the figure below.



1. *Initial value of centroids* : Suppose we use medicine A and medicine B as the first centroids. Let c_1 and c_2 denote the coordinate of the centroids, then $c_1 = (1,1)$ and $c_2 = (2,1)$



2. *Objects-Centroids distance* : we calculate the distance between cluster centroid to each object. Let us use Euclidean distance , then we have distance matrix at iteration 0 is

$$\mathbf{D}^0 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 1 & 0 & 2.83 & 4.24 \end{bmatrix} \begin{array}{ll} \mathbf{c}_1 = (1,1) & group-1 \\ \mathbf{c}_2 = (2,1) & group-2 \end{array}$$

$$\begin{array}{cccc|c} A & B & C & D & \\ 1 & 2 & 4 & 5 & X \\ 1 & 1 & 3 & 4 & Y \end{array}$$

3. *Objects clustering* : We assign each object based on the minimum distance. Thus, medicine A is assigned to group 1, medicine B to group 2, medicine C to group 2 and medicine D to group 2. The element of Group matrix below is 1 if and only if the object is assigned to that group.

$$\mathbf{G}^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{array}{l} group-1 \\ group-2 \end{array}$$

$$\begin{array}{cccc} A & B & C & D \end{array}$$

4. *Iteration-1, determine centroids* : Knowing the members of each group, now we compute the new centroid of each group based on these new memberships. Group 1 only has one member thus the centroid remains in $\mathbf{c}_1 = (1,1)$. Group 2 now has three members, thus the centroid is the average coordinate among the three members: $\mathbf{c}_2 = (\frac{2+4+5}{3}, \frac{1+3+4}{3}) = (\frac{11}{3}, \frac{8}{3})$.

5. *Iteration-1, Objects-Centroids distances* : The next step is to compute the distance of all objects to the new centroids. Similar to step 2, we have distance matrix at iteration 1 is

$$\mathbf{D}^1 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \begin{array}{ll} \mathbf{c}_1 = (1,1) & group-1 \\ \mathbf{c}_2 = (\frac{11}{3}, \frac{8}{3}) & group-2 \end{array}$$

$$\begin{array}{cccc|c} A & B & C & D & \\ 1 & 2 & 4 & 5 & X \\ 1 & 1 & 3 & 4 & Y \end{array}$$

6. *Iteration-1, Objects clustering:* Similar to step 3, we assign each object based on the minimum distance. Based on the new distance matrix, we move the medicine B to Group 1 while all the other objects remain. The Group matrix is shown below

$$\mathbf{G}^1 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{array}{l} group-1 \\ group-2 \end{array}$$

$$\begin{array}{cccc} A & B & C & D \end{array}$$

7. *Iteration 2, determine centroids:* Now we repeat step 4 to calculate the new centroids coordinate based on the clustering of previous iteration. Group1 and group 2 both has two members, thus the new centroids are $\mathbf{c}_1 = (\frac{1+2}{2}, \frac{1+1}{2}) = (1\frac{1}{2}, 1)$ and $\mathbf{c}_2 = (\frac{4+5}{2}, \frac{3+4}{2}) = (4\frac{1}{2}, 3\frac{1}{2})$

8. *Iteration-2, Objects-Centroids distances* : Repeat step 2 again, we have new distance matrix at iteration 2 as

$$\mathbf{D}^2 = \begin{array}{c} \begin{bmatrix} 0.5 & 0.5 & 3.20 & 4.61 \\ 4.30 & 3.54 & 0.71 & 0.71 \end{bmatrix} \\ \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \end{array} \begin{array}{l} \mathbf{c}_1 = (1\tfrac{1}{2}, 1) \quad group-1 \\ \mathbf{c}_2 = (4\tfrac{1}{2}, 3\tfrac{1}{2}) \quad group-2 \\ \\ X \\ Y \end{array}$$

9. *Iteration-2, Objects clustering:* Again, we assign each object based on the minimum distance.

$$\mathbf{G}^2 = \begin{array}{c} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \\ \begin{array}{cccc} A & B & C & D \end{array} \end{array} \begin{array}{l} group-1 \\ group-2 \\ \\ \end{array}$$

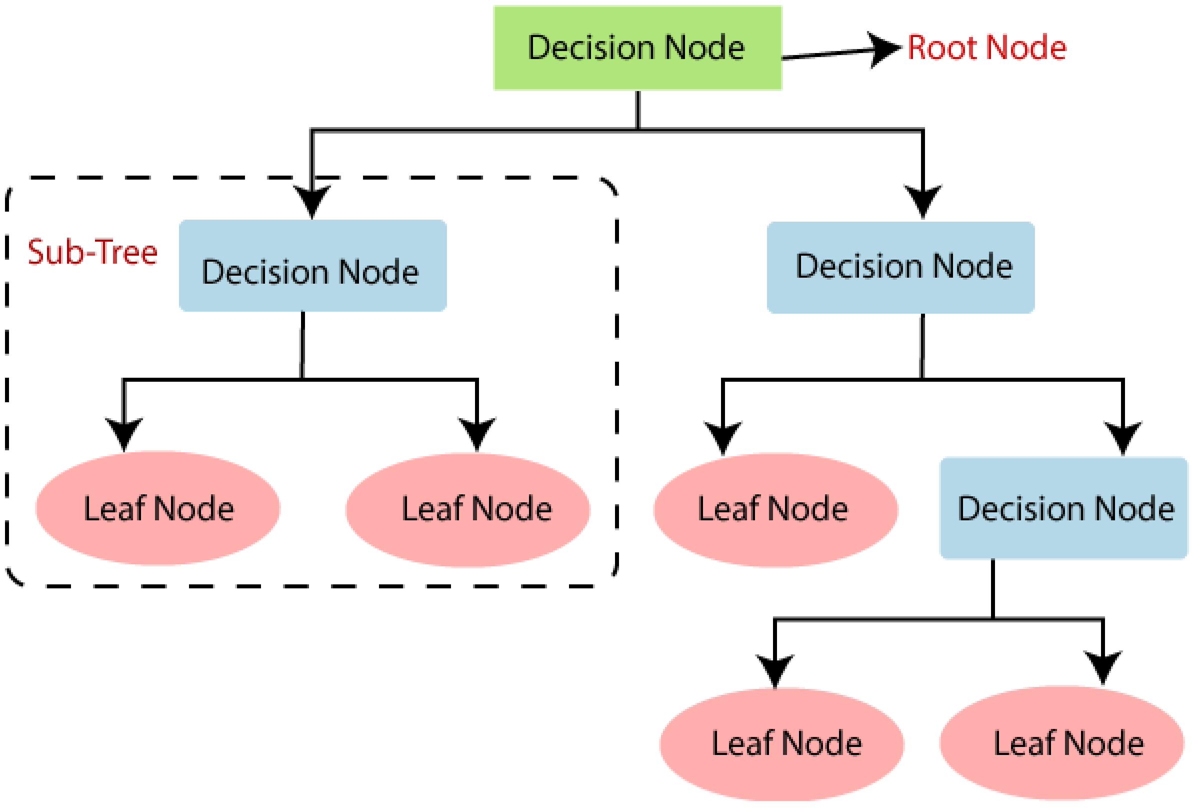
We obtain result that $\mathbf{G}^2 = \mathbf{G}^1$. Comparing the grouping of last iteration and this iteration reveals that the objects does not move group anymore. Thus, the computation of the k-mean clustering has reached its stability and no more iteration is needed. We get the final grouping as the results

Object	Feature 1 (X): weight index	Feature 2 (Y): pH	Group (result)
Medicine A	1	1	1
Medicine B	2	1	1
Medicine C	4	3	2
Medicine D	5	4	2

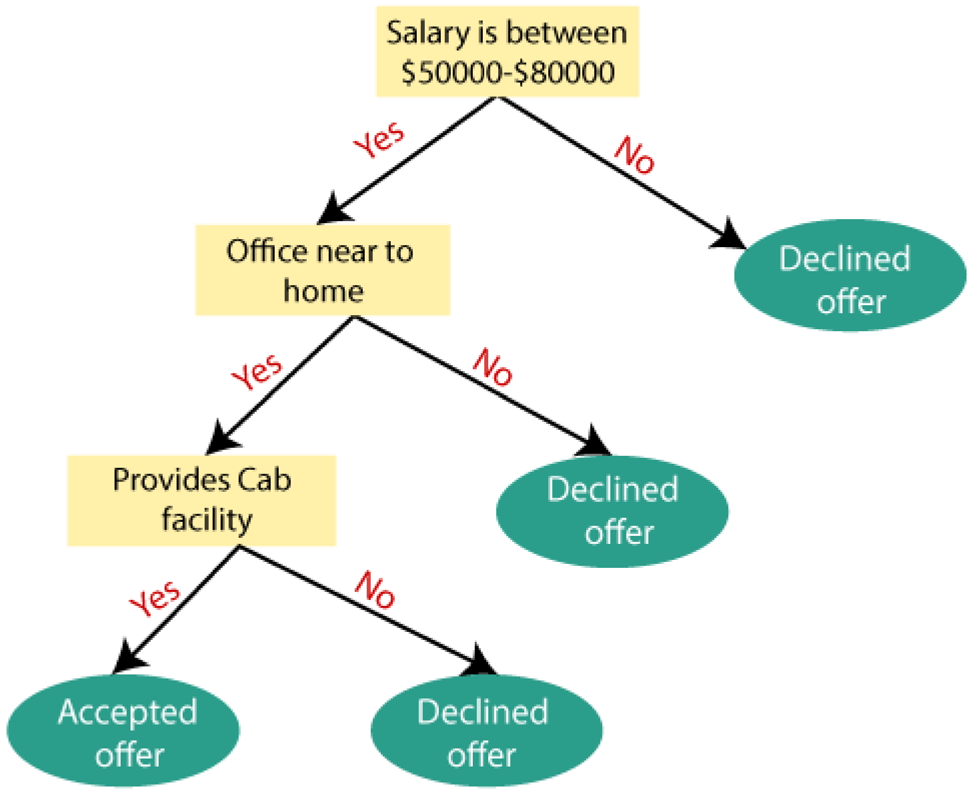
Decision Tree Algorithm:

- Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**
- ***t is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.***
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
- In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
- Below diagram explains the general structure of a decision tree:

Note: A decision tree can contain categorical data (YES/NO) as well as numeric data.



uppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or Not. So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM). The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels. The next decision node further gets split into one decision node (Cab facility) and one leaf node. Finally, the decision node splits into two leaf nodes (Accepted offers and Declined offer). Consider the below diagram:



Attribute Selection Measures

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as **Attribute selection measure or ASM.** By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

- **Information Gain**
- **Gini Index**

1. Information Gain:

- Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
- It calculates how much information a feature provides us about a class.
- According to the value of information gain, we split the node and build the decision tree.

- A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

1. Information Gain= Entropy(S)- [(Weighted Avg) *Entropy(each feature)

Entropy: Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

```
Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)
```

Where,

- **S= Total number of samples**
- **P(yes)= probability of yes**
- **P(no)= probability of no**

2. Gini Index:

- Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.
- An attribute with the low Gini index should be preferred as compared to the high Gini index.
- It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.
- Gini index can be calculated using the below formula:

Gini Index= $1- \sum_j P_j^2$

training and testing a classifier:

The most important thing you can do to properly evaluate your model is to not train the model on the entire dataset. machine learning algorithms is their tendency to “memorize” the data they have been trained on. In data science terms this is called over-fitting the data and can make your model look great on the training data but in actuality it has no ability to generalize on new data that is given.

To gain perspective into how the model is actually doing we use a train/test split.

A typical train/test split would be to use 70% of the data for training and 30% of the data for testing.

The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model.

scikit-learn machine learning library to perform the train-test split procedure.

Train-Test Split Evaluation

The train-test split is a technique for evaluating the performance of a machine learning algorithm.

It can be used for classification or regression problems and can be used for any supervised learning algorithm.

The procedure involves taking a dataset and dividing it into two subsets. The first subset is used to fit the model and is referred to as the training dataset. The second subset is not used to train the model; instead, the input element of the dataset is provided to the model, then predictions are made and compared to the expected values. This second dataset is referred to as the test dataset.

- **Train Dataset**: Used to fit the machine learning model.
- **Test Dataset**: Used to evaluate the fit machine learning model.

common split percentages include:

- Train: 80%, Test: 20%
- Train: 67%, Test: 33%
- Train: 50%, Test: 50%
-

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=1)
```

```
>>> import numpy as np
>>> from sklearn.model_selection import train_test_split
>>> X, y = np.arange(10).reshape((5, 2)), range(5)
>>> X
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7],
       [8, 9]])
```

```
>>> list(y)
[0, 1, 2, 3, 4]

>>> X_train, X_test, y_train, y_test = train_test_split(
...     X, y, test_size=0.33, random_state=42)
...
>>> X_train
array([[4, 5],
       [0, 1],
       [6, 7]])
>>> y_train
[2, 0, 3]
>>> X_test
array([[2, 3],
       [8, 9]])
>>> y_test
[1, 4]
```