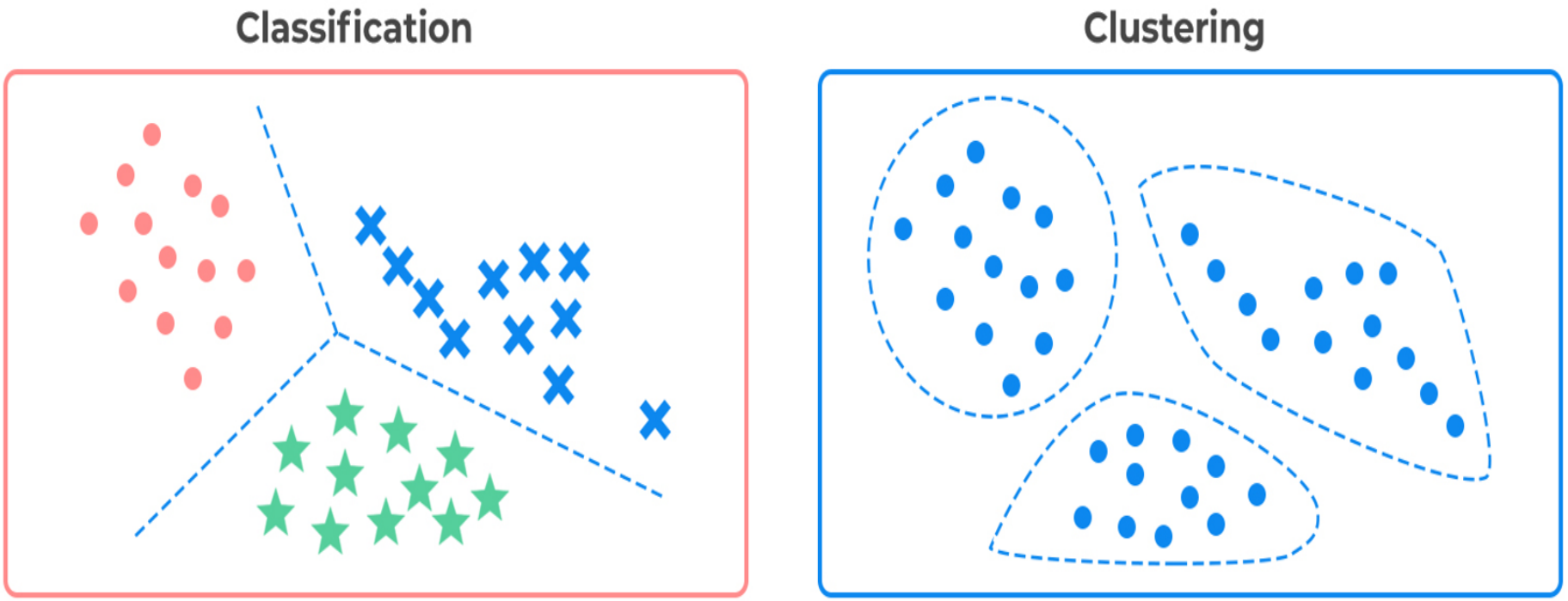
Unit-4

Clustering

What is Clustering



What is Clustering

- Clustering is the process of making a group of abstract objects into classes of similar objects.
  - A cluster of data objects can be treated as one group.
  - While doing cluster analysis, we first partition the set of data into groups based on data similarity and then assign the labels to the groups.
  - The main advantage of clustering over classification is that, it is adaptable to changes and helps single out useful features that distinguish different groups.

https://www.tutorialspoint.com/data_mining/dm_cluster_analysis.htm

1. Search Engines : Image Search

You may be familiar with the concept of image search which Google provides. So what this system does is that first, it applies the clustering algorithm on all the images available in the database available. After which similar images would fall under the same cluster. So when a particular user provides an image for reference what it will be doing is applying the trained clustering model on the image to identify its cluster once this is done it simply returns all the images from this cluster.

https://www.analyticsvidhya.com/blog/2021/11/quick-tutorial-clustering-data-science/

2. Customer Segmentation

We can also cluster our customers based on their purchase history and their activity on our website. This is really important and useful to understand who our customers are and what they require so that our system can adapt to their requirements and suggest products to each respective segment accordingly.

https://www.analyticsvidhya.com/blog/2021/11/quick-tutorial-clustering-data-science/

3. Semi-supervised Learning.

When you are working on semi-supervised learning in which you are only provided with a few labels, there you could perform clustering algorithms and generate labels for all instances falling under the same cluster. This technique is really good for increasing the number of labels after which a supervised learning algorithm can be used and its performance gets better.

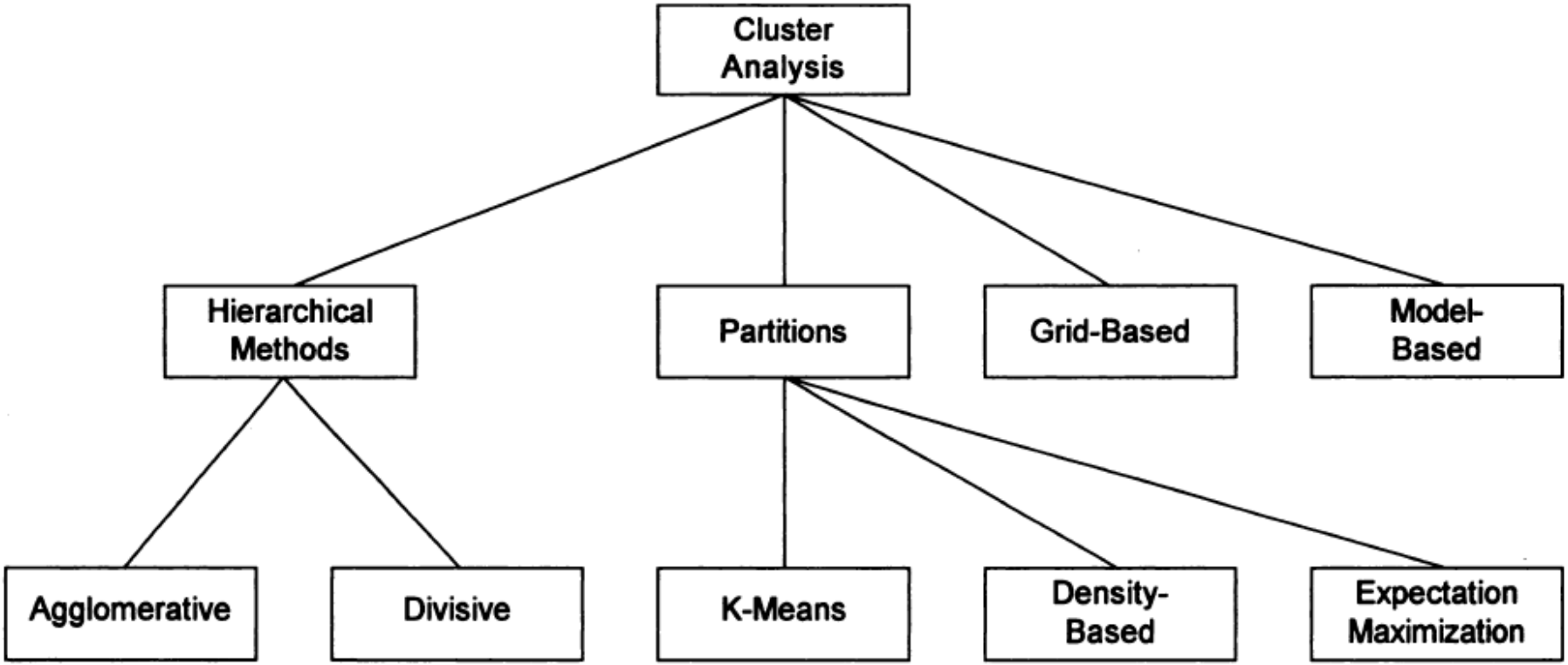
https://www.analyticsvidhya.com/blog/2021/11/quick-tutorial-clustering-data-science/

Applications of Cluster Analysis

- Clustering analysis is broadly used in many applications such as market research, pattern recognition, data analysis, and image processing.
- Clustering can also help marketers discover distinct groups in their customer base. And they can characterize their customer groups based on the purchasing patterns.
- In the field of biology, it can be used to derive plant and animal taxonomies, categorize genes with similar functionalities and gain insight into structures inherent to populations.
- Clustering also helps in identification of areas of similar land use in an earth observation database. It also helps in the identification of groups of houses in a city according to house type, value, and geographic location.
- Clustering is also used in outlier detection applications such as detection of credit card fraud.

Clustering Methods

- Clustering methods can be classified into the following categories –
  - Partitioning Method
  - Hierarchical Method
  - Density-based Method
  - Grid-Based Method
  - Model-Based Method
  - Constraint-based Method


Cluster Analysis
Hierarchical Methods
Partitions
Grid-Based
Model-Based
Agglomerative
Divisive
K-Means
Density-Based
Expectation Maximization

Partitioning Method

- Suppose we are given a database of ‘n’ objects and the partitioning method constructs ‘k’ partition of data. Each partition will represent a cluster and k ≤ n. It means that it will classify the data into k groups, which satisfy the following requirements −
- Each group contains at least one object.
- Each object must belong to exactly one group.
- **Points to remember –**
  - For a given number of partitions (say k), the partitioning method will create an initial partitioning.
  - Then it uses the iterative relocation technique to improve the partitioning by moving objects from one group to other.

K-Means Clustering

K-means is the simplest and most popular classical clustering method that is easy to implement. The classical method can only be used if the data about all the objects is located in the main memory. The method is called K-means since each of the K clusters is represented by the mean of the objects (called the centroid) within it. It is also called the *centroid method* since at each step the centroid point of each cluster is assumed to be known and each of the remaining points are allocated to the cluster whose centroid is closest to it. Once this allocation is completed, the centroids of the clusters are recomputed using simple means and the process of allocating points to each cluster is repeated until there is no change in the clusters (or some other stopping criterion, e.g. no significant reduction in the squared error, is met). The method may also be looked at as a search problem where the aim is essentially to find the optimum clusters given the number of clusters and seeds specified by the user. Obviously, we cannot use a brute-force or exhaustive search method to find the optimum, so we consider solutions that may not be optimal but may be computed efficiently.

The K-means method may be described as follows:

1. Select the number of clusters. Let this number be k.
2. Pick k seeds as centroids of the k clusters. The seeds may be picked randomly unless the user has some insight into the data.
3. Compute the Euclidean distance of each object in the dataset from each of the centroids.
4. Allocate each object to the cluster it is nearest to based on the distances computed in the previous step.
5. Compute the centroids of the clusters by computing the means of the attribute values of the objects in each cluster.
6. Check if the stopping criterion has been met (e.g. the cluster membership is unchanged). If yes, go to Step 7. If not, go to Step 3.
7. [Optional] One may decide to stop at this stage or to split a cluster or combine two clusters

Apply K-Means Algorithm

Student	*Age*	*Mark1*	*Mark2*	*Mark3*
S_1	18	73	75	57
S_2	18	79	85	75
S_3	23	70	70	52
S_4	20	55	55	55
S_5	22	85	86	87
S_6	19	91	90	89
S_7	20	70	65	60
S_8	21	53	56	59
S_9	19	82	82	60
S_{10}	47	75	76	77

Steps 1 and 2: Let the three seeds be the first three students as shown in Table

Student	*Age*	*Mark1*	*Mark2*	*Mark3*
S_1	18	73	75	57
S_2	18	79	85	75
S_3	23	70	70	52

Steps 3 and 4: Now compute the distances using the four attributes and using the sum of absolute differences for simplicity (i.e. using the K-median method).
Based on these distances, each student is allocated to the nearest cluster.

First iteration—allocating each object to the nearest cluster

					Distances from clusters			
C_1	*18.0*	*73.0*	*75.0*	*57.0*	*From*	*From*	*From*	*Allocation*
C_2	*18.0*	*79.0*	*85.0*	*75.0*	C_1	C_2	C_3	*to the nearest*
C_3	*23.0*	*70.0*	*70.0*	*52.0*				*cluster*
S_1	18.0	73.0	75.0	57.0	0.0	34.0	18.0	C_1
S_2	18.0	79.0	85.0	75.0	34.0	0.0	52.0	C_2
S_3	23.0	70.0	70.0	52.0	18.0	52.0	0.0	C_3
S_4	20.0	55.0	55.0	55.0	42.0	76.0	36.0	C_3
S_5	22.0	85.0	86.0	87.0	57.0	23.0	67.0	C_2
S_6	19.0	91.0	90.0	89.0	66.0	32.0	82.0	C_2
S_7	20.0	70.0	65.0	60.0	18.0	46.0	16.0	C_3
S_8	21.0	53.0	56.0	59.0	44.0	74.0	40.0	C_3
S_9	19.0	82.0	82.0	60.0	20.0	22.0	36.0	C_1
S_{10}	47.0	75.0	76.0	77.0	52.0	44.0	60.0	C_2

Step 5: compares the cluster means of clusters with the original seeds.

Comparing new centroids and the seeds

	Age	*Mark1*	*Mark2*	*Mark3*
C_1	18.5	77.5	78.5	58.5
C_2	26.5	82.5	84.3	82.0
C_3	21	61.5	61.5	56.5
Seed1	18	73	75	57
Seed2	18	79	85	75
Seed3	23	70	70	52

Second iteration—allocating each object to the nearest cluster

					Distances from clusters			
C_1	*18.5*	*77.5*	*78.5*	*58.5*	*From*	*From*	*From*	*Allocation*
C_2	*26.5*	*82.5*	*84.3*	*82.0*	C_1	C_2	C_3	*to the nearest*
C_3	*21.0*	*62.0*	*61.5*	*56.5*				*cluster*
S_1	18.0	73.0	75.0	57.0	10.0	52.3	28.0	C_1
S_2	18.0	79.0	85.0	75.0	25.0	19.8	62.0	C_2
S_3	23.0	70.0	70.0	52.0	27.0	60.3	23.0	C_3
S_4	20.0	55.0	55.0	55.0	51.0	90.3	16.0	C_3
S_5	22.0	85.0	86.0	87.0	47.0	13.8	79.0	C_2
S_6	19.0	91.0	90.0	89.0	56.0	28.8	92.0	C_2
S_7	20.0	70.0	65.0	60.0	24.0	60.3	16.0	C_3
S_8	21.0	53.0	56.0	59.0	50.0	86.3	17.0	C_3
S_9	19.0	82.0	82.0	60.0	10.0	32.3	46.0	C_1
S_{10}	47.0	75.0	76.0	77.0	52.0	41.3	74.0	C_2

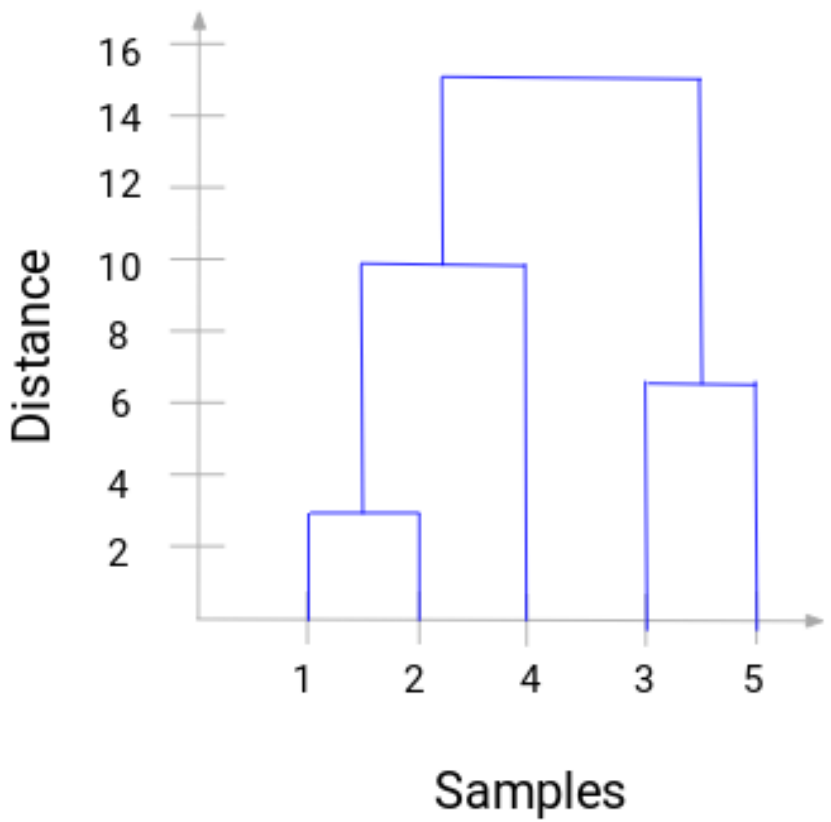
Hierarchical Clustering

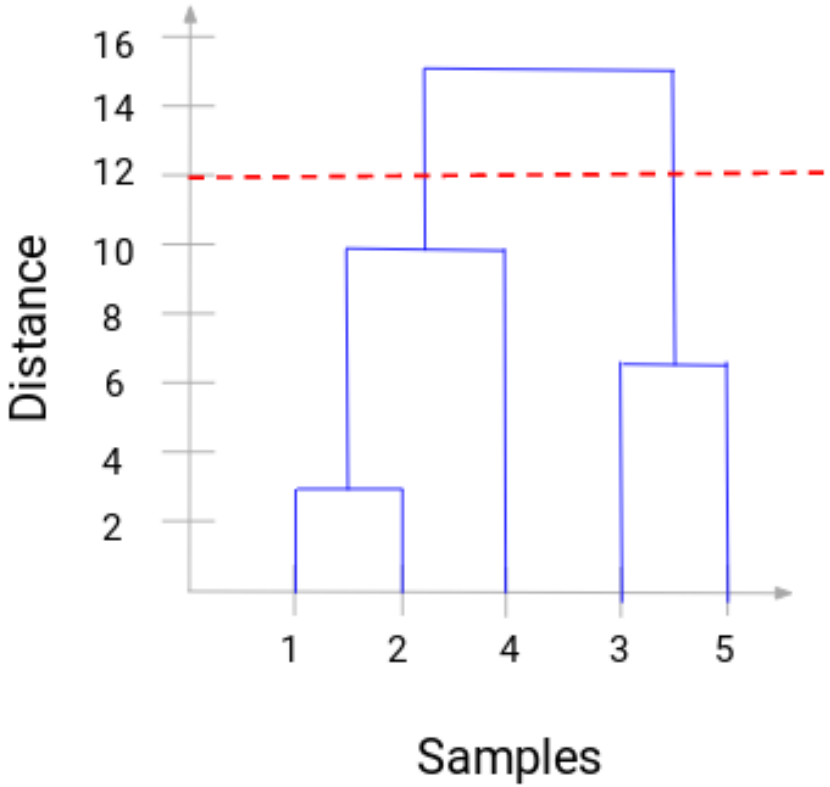
Hierarchical methods produce a nested series of clusters as opposed to the partitional methods which produce only a flat set of clusters. Essentially the hierarchical methods attempt to capture the structure of the data by constructing a tree of clusters. This approach allows clusters to be found at different levels of granularity.

Hierarchical Clustering Algorithm Types

- **Agglomerative hierarchical algorithms** – In agglomerative hierarchical algorithms, each data point is treated as a single cluster and then successively merge or agglomerate (bottom-up approach) the pairs of clusters. The hierarchy of the clusters is represented as a dendrogram or tree structure.
- **Divisive hierarchical algorithms** – On the other hand, in divisive hierarchical algorithms, all the data points are treated as one big cluster and the process of clustering involves dividing (Top-down approach) the one big cluster into various small clusters.

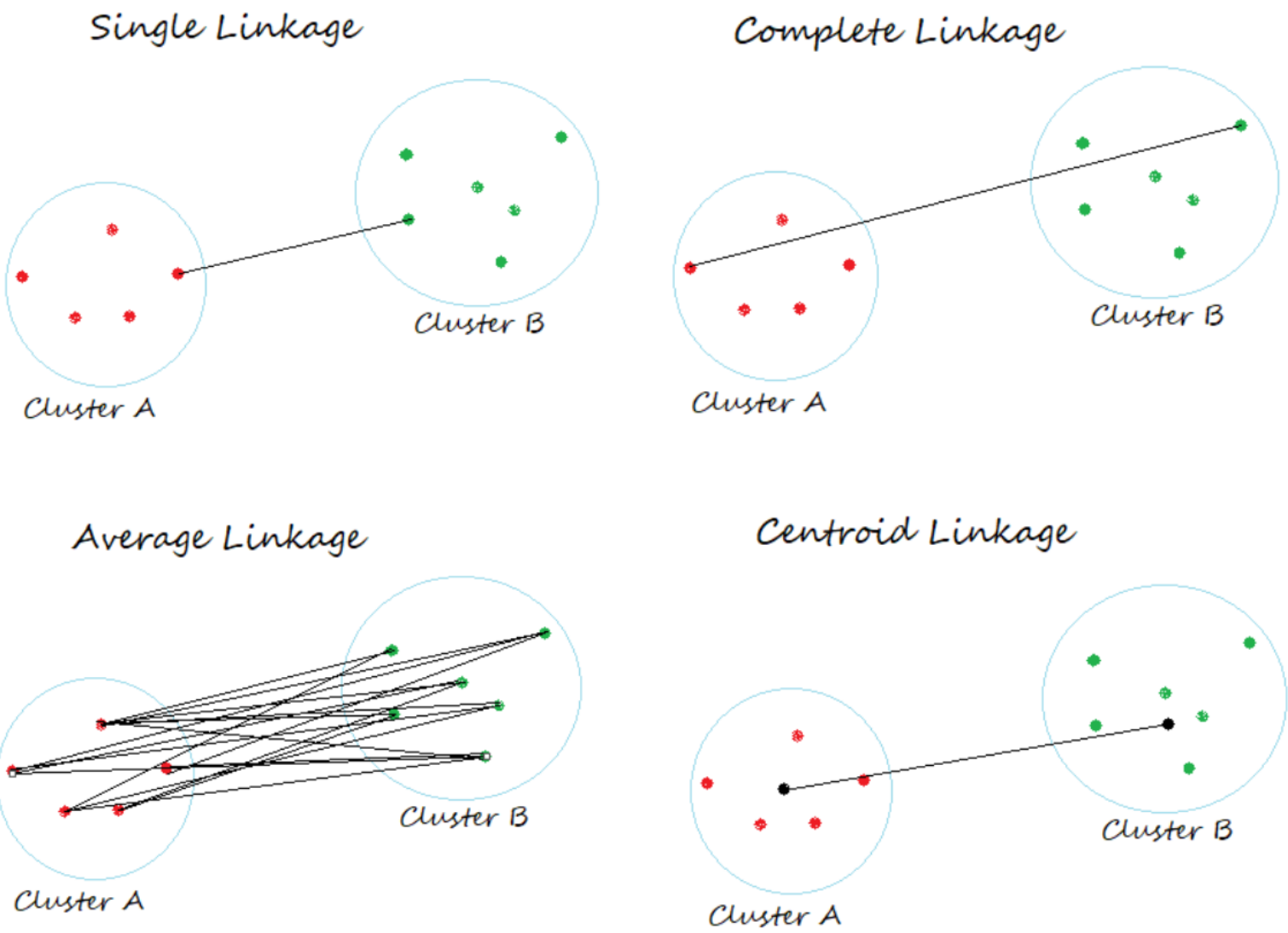
Dendrogram


16
14
12
10
8
6
4
2
Distance
1
2
4
3
5
Samples


16
14
12
10
8
6
4
2
Distance
1
2
4
3
5
Samples

Linkage Criteria

- In **Single Linkage,** the distance between two clusters is the minimum distance between members of the two clusters
- In **Complete Linkage,** the distance between two clusters is the maximum distance between members of the two clusters
- In **Average Linkage,** the distance between two clusters is the average of all distances between members of the two clusters
- In **Centroid Linkage,** the distance between two clusters is is the distance between their centroids


Single Linkage
Cluster B
Cluster A
Complete Linkage
Cluster B
Cluster A
Average Linkage
Cluster B
Cluster A
Centroid Linkage
Cluster B
Cluster A

Advantages of the hierarchical approach

1. The hierarchical approach can provide more insight into the data by showing a hierarchy of clusters than a flat cluster structure created by a partitioning method like the K-means method.
2. Hierarchical methods are conceptually simpler and can be implemented easily.
3. In some applications only proximity data is available and then the hierarchical approach may be better.
4. Hierarchical methods can provide clusters at different levels of granularity.

Disadvantages of the hierarchical approach

1. The hierarchical methods do not include a mechanism by which objects that have been incorrectly put in a cluster may be reassigned to another cluster.
2. The time complexity of hierarchical methods can be shown to be $O(n^3)$.

Density-Based Method

Density-based methods

In this class of methods, typically for each data point in a cluster, at least a minimum number of points must exist within a given radius. Density-based methods can deal with arbitrary shape clusters since the major requirement of such methods is that each cluster be a dense region of points surrounded by regions of low density.

DBSCAN

Density-based spatial clustering of applications with noise

DBSCAN (density based spatial clustering of applications with noise) is one example of a density-based method for clustering. The method was designed for spatial databases but can be used in other applications. It requires two input parameters: the size of the neighbourhood (R) and the minimum points in the neighbourhood (N). Essentially these two parameters determine the density within the clusters the user is willing to accept since they specify how many points must be in a region. The number of points not only determines the density of acceptable clusters but it also determines which objects will be labelled outliers or noise. Objects are declared to be outliers if there are few other objects in their neighbourhood. The size parameter R determines the size of the clusters found. If R is big enough, there would be one big cluster and no outliers. If R is small, there will be small dense clusters and there might be many outliers.

We now define a number of concepts that are required in the DBSCAN method:

1. ***Neighbourhood:*** The neighbourhood of an object y is defined as all the objects that are within the radius R from y.
2. ***Core object:*** An object y is called a core object if there are N objects within its neighbourhood.
3. ***Proximity:*** Two objects are defined to be in proximity to each other if they belong to the same cluster. Object x_1 is in proximity to object x_2 if two conditions are satisfied:
   (a) The objects are close enough to each other, i.e. within a distance of R.
   (b) x_2 is a core object as defined above.

Classification

What is Classification

- Classification may be defined as the process of predicting class or category from observed values or given data points. The categorized output can have the form such as “Black” or “White” or “spam” or “no spam”.
- Mathematically, classification is the task of approximating a mapping function (f) from input variables (X) to output variables (Y). It is basically belongs to the supervised machine learning in which targets are also provided along with the input data set.

Types of Classifications

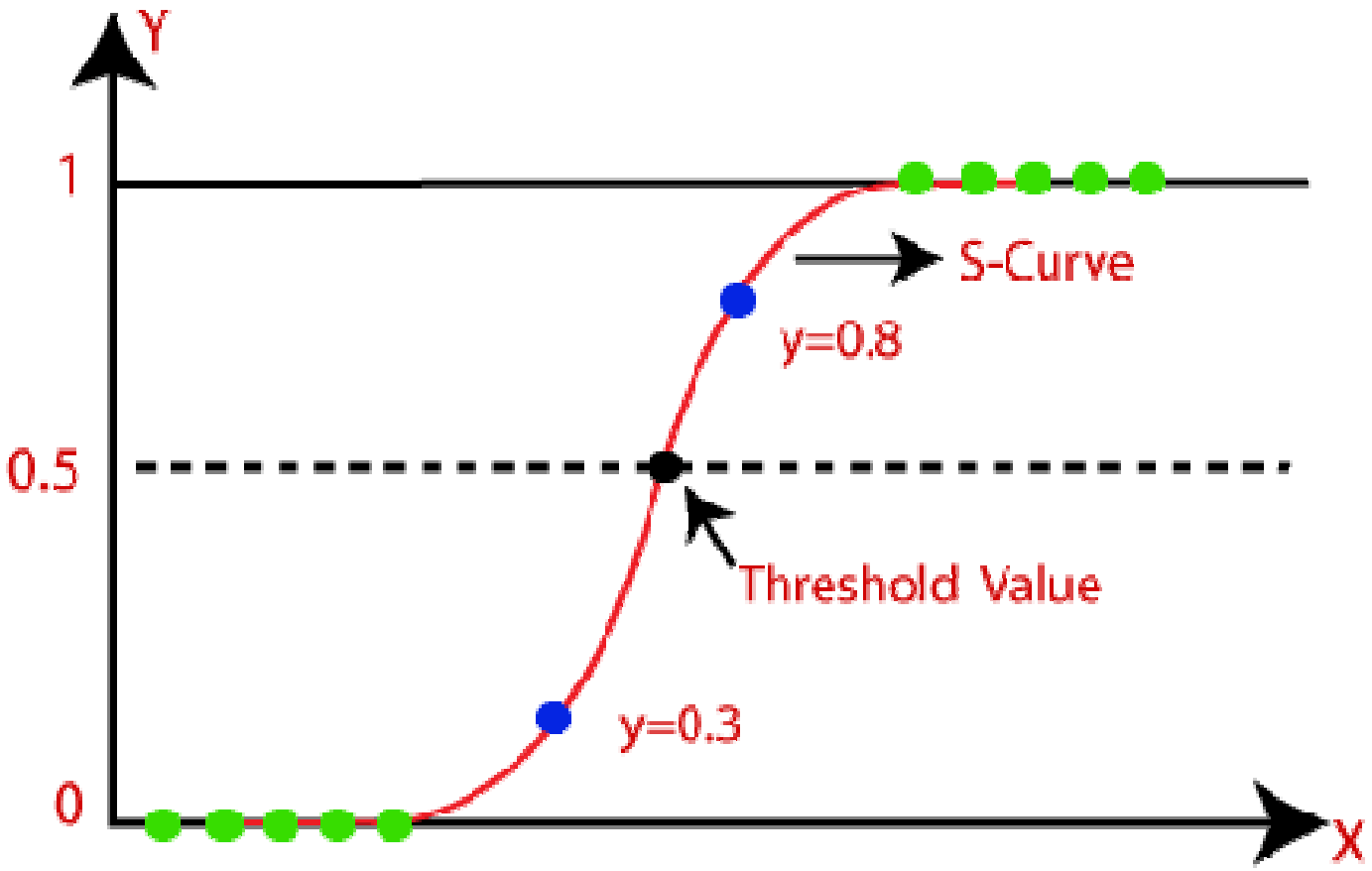
- **Binary Classifier:** If the classification problem has only two possible outcomes, then it is called as Binary Classifier.
  Examples: YES or NO, MALE or FEMALE, SPAM or NOT SPAM, CAT or DOG, etc.
- **Multi-class Classifier:** If a classification problem has more than two outcomes, then it is called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

Logistic Regression

- 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.
- 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**.

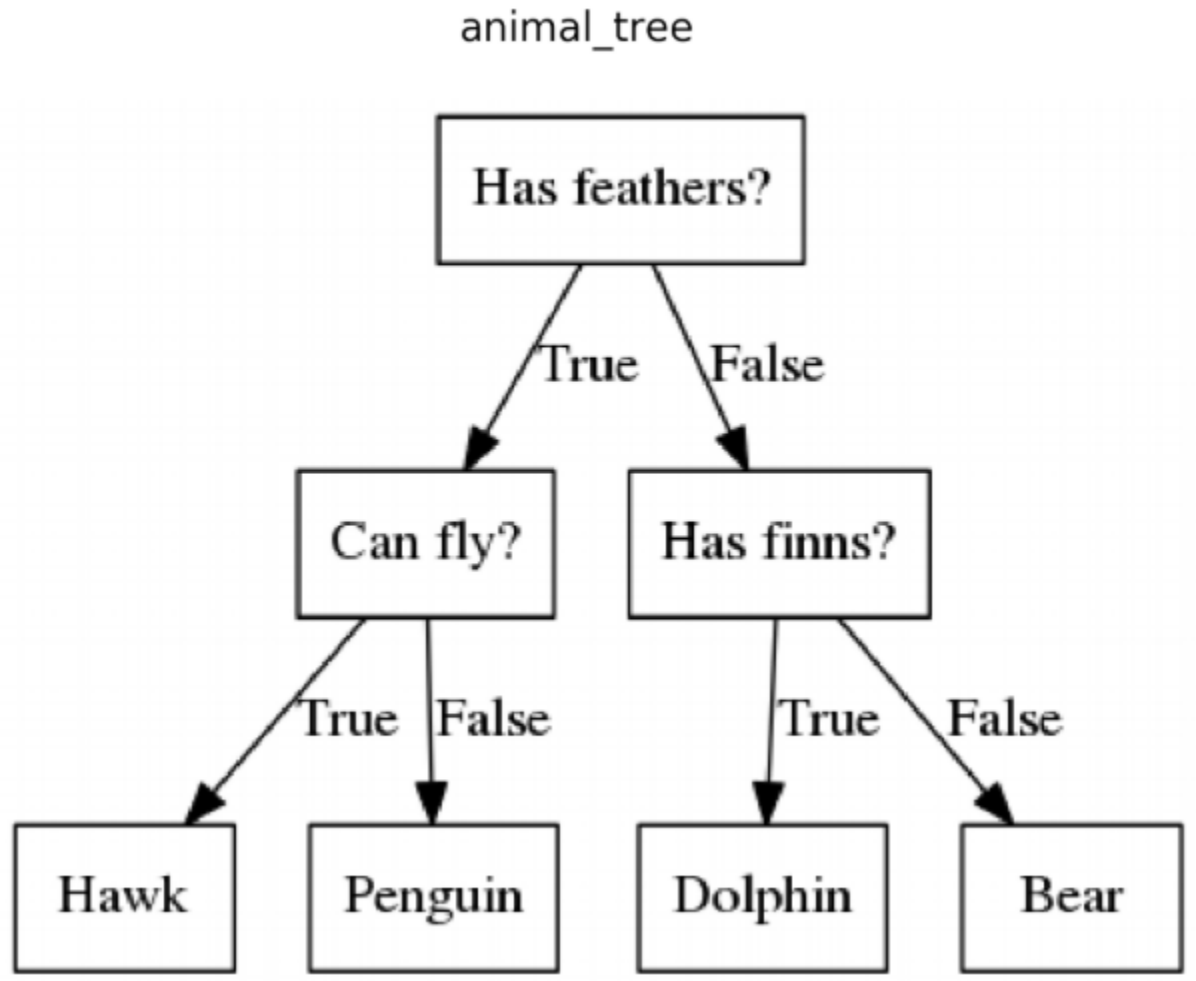
Logistic Regression

- 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).



https://www.javatpoint.com/logistic-regression-in-machine-learning

Decision Tree

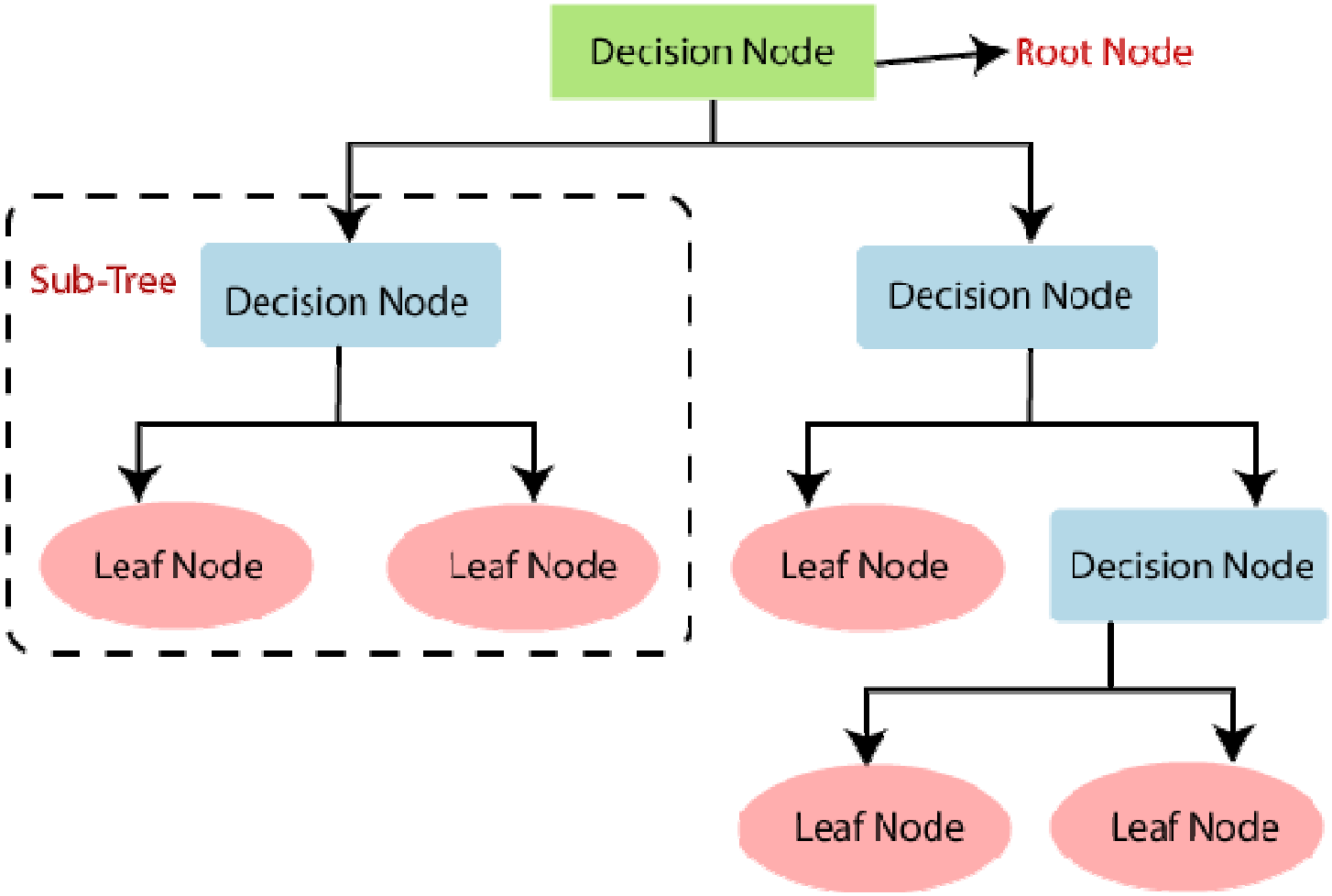


Decision Tree

- 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.**
- 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.

Decision Tree

- ***It 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.



https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm

While building a Decision tree, the main thing is to select the best attribute from the total features list of the dataset for the root node as well as for sub-nodes. The selection of best attributes is being achieved with the help of a technique known as the **Attribute selection measure (ASM)**.

There are two techniques for ASM:

a) Information Gain

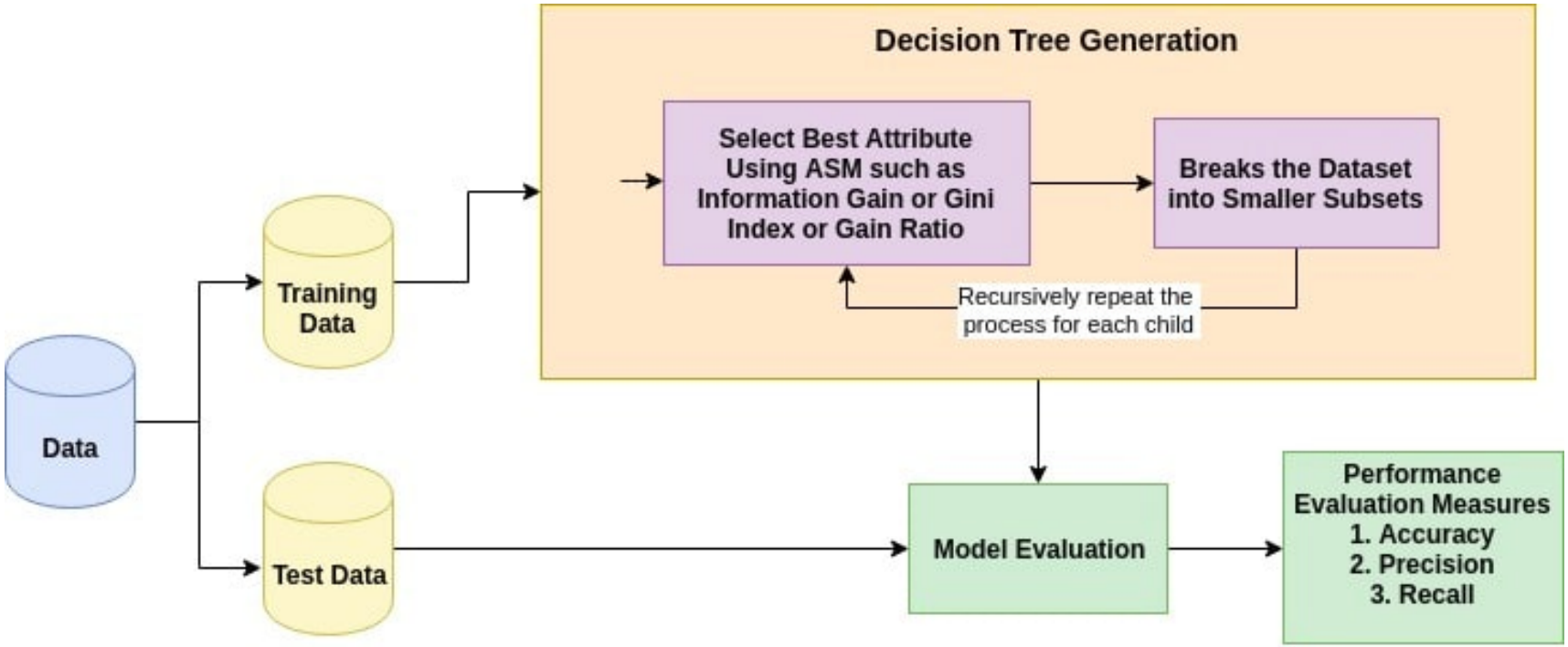
b) Gini Index

How Does the Decision Tree Algorithm works?

1. Select the best Feature using Attribute Selection Measures(ASM) to split the records.
2. Make that attribute/feature a decision node and break the dataset into smaller subsets.
3 Start the tree-building process by repeating this process recursively for each child until one of the following condition is being achieved :
   a) All tuples belonging to the same attribute value.
   b) There are no more of the attributes remaining.
   c) There are no more instances remaining.

https://www.analyticsvidhya.com/blog/2021/04/beginners-guide-to-decision-tree-classification-using-python/

How Does the Decision Tree Algorithm works?



https://www.analyticsvidhya.com/blog/2021/04/beginners-guide-to-decision-tree-classification-using-python/

Information Gain (Entropy)

- Information gain is the measurement of changes in entropy value after the splitting/segmentation of the dataset based on an attribute.
- It tells how much information a feature/attribute provides us.
- Following the value of the information gain, splitting of the node and decision tree building is being done.
- Decision tree always tries to maximize the value of the information gain, and a node/attribute having the highest value of the information gain is being split first

Day	Outlook	Temp	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Calculation Formulas

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

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

Attribute: Outlook

$$Values\,(Outlook) = Sunny, Overcast, Rain$$

$$S = [9+, 5-] \qquad Entropy(S) = -\frac{9}{14} log_2 \frac{9}{14} - \frac{5}{14} log_2 \frac{5}{14} = 0.94$$

$$S_{Sunny} \leftarrow [2+, 3-] \qquad Entropy(S_{Sunny}) = -\frac{2}{5} log_2 \frac{2}{5} - \frac{3}{5} log_2 \frac{3}{5} = 0.971$$

$$S_{Overcast} \leftarrow [4+, 0-] \qquad Entropy(S_{Overcast}) = -\frac{4}{4} log_2 \frac{4}{4} - \frac{0}{4} log_2 \frac{0}{4} = 0$$

$$S_{Rain} \leftarrow [3+, 2-] \qquad Entropy(S_{Rain}) = -\frac{3}{5} log_2 \frac{3}{5} - \frac{2}{5} log_2 \frac{2}{5} = 0.971$$

$$Gain\,(S, Outlook) = Entropy(S) - \sum_{v \in \{Sunny, Overcast, Rain\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, Outlook)$$

$$= Entropy(S) - \frac{5}{14} Entropy(S_{Sunny}) - \frac{4}{14} Entropy(S_{Overcast})$$

$$- \frac{5}{14} Entropy(S_{Rain})$$

$$Gain(S, Outlook) = 0.94 - \frac{5}{14} 0.971 - \frac{4}{14}\ 0 - \frac{5}{14} 0.971 = 0.2464$$

Attribute: Temp

$$Values\ (Temp) = Hot, Mild, Cool$$

$$S = [9+, 5-] \qquad Entropy(S) = -\frac{9}{14} log_2 \frac{9}{14} - \frac{5}{14} log_2 \frac{5}{14} = 0.94$$

$$S_{Hot} \leftarrow [2+, 2-] \qquad Entropy(S_{Hot}) = -\frac{2}{4} log_2 \frac{2}{4} - \frac{2}{4} log_2 \frac{2}{4} = 1.0$$

$$S_{Mild} \leftarrow [4+, 2-] \qquad Entropy(S_{Mild}) = -\frac{4}{6} log_2 \frac{4}{6} - \frac{2}{6} log_2 \frac{2}{6} = 0.9183$$

$$S_{Cool} \leftarrow [3+, 1-] \qquad Entropy(S_{Cool}) = -\frac{3}{4} log_2 \frac{3}{4} - \frac{1}{4} log_2 \frac{1}{4} = 0.8113$$

$$Gain\ (S, Temp) = Entropy(S) - \sum_{v \in \{Hot, Mild, Cool\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, Temp)$$

$$= Entropy(S) - \frac{4}{14} Entropy(S_{Hot}) - \frac{6}{14} Entropy(S_{Mild})$$

$$- \frac{4}{14} Entropy(S_{Cool})$$

$$Gain(S, Temp) = 0.94 - \frac{4}{14} 1.0 - \frac{6}{14}\ 0.9183 - \frac{4}{14} 0.8113 = 0.0289$$

Attribute: Humidity

$$Values\,(Humidity) = High, Normal$$

$$S = [9+, 5-] \qquad Entropy(S) = -\frac{9}{14} log_2 \frac{9}{14} - \frac{5}{14} log_2 \frac{5}{14} = 0.94$$

$$S_{High} \leftarrow [3+, 4-] \qquad Entropy(S_{High}) = -\frac{3}{7} log_2 \frac{3}{7} - \frac{4}{7} log_2 \frac{4}{7} = 0.9852$$

$$S_{Normal} \leftarrow [6+, 1-] \qquad Entropy(S_{Normal}) = -\frac{6}{7} log_2 \frac{6}{7} - \frac{1}{7} log_2 \frac{1}{7} = 0.5916$$

$$Gain\,(S, Humidity) = Entropy(S) - \sum_{v \in \{High, Normal\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, Humidity)$$

$$= Entropy(S) - \frac{7}{14} Entropy(S_{High}) - \frac{7}{14} Entropy(S_{Normal})$$

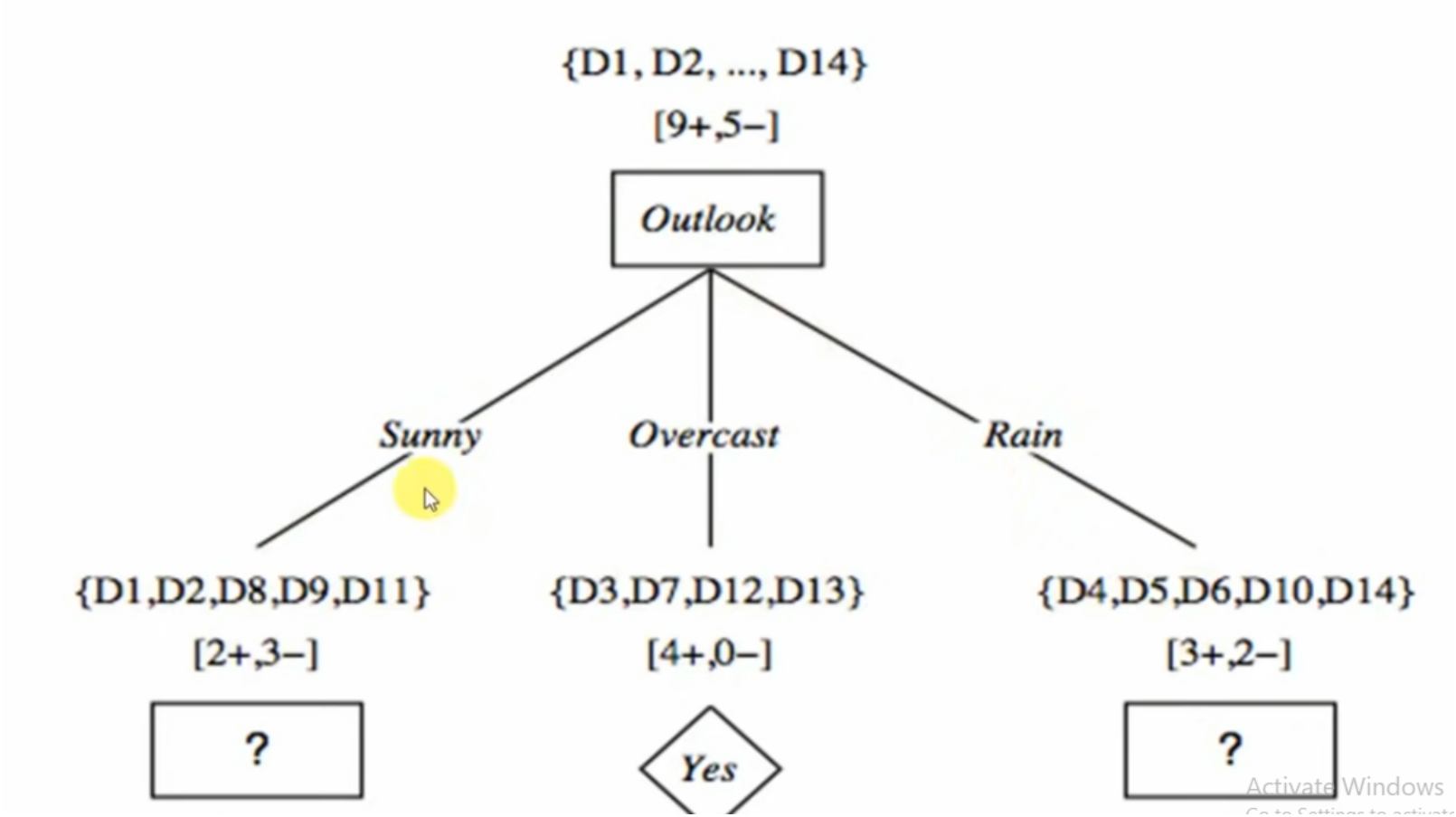
$$Gain(S, Humidity) = 0.94 - \frac{7}{14} 0.9852 - \frac{7}{14}\ 0.5916 = 0.1516$$

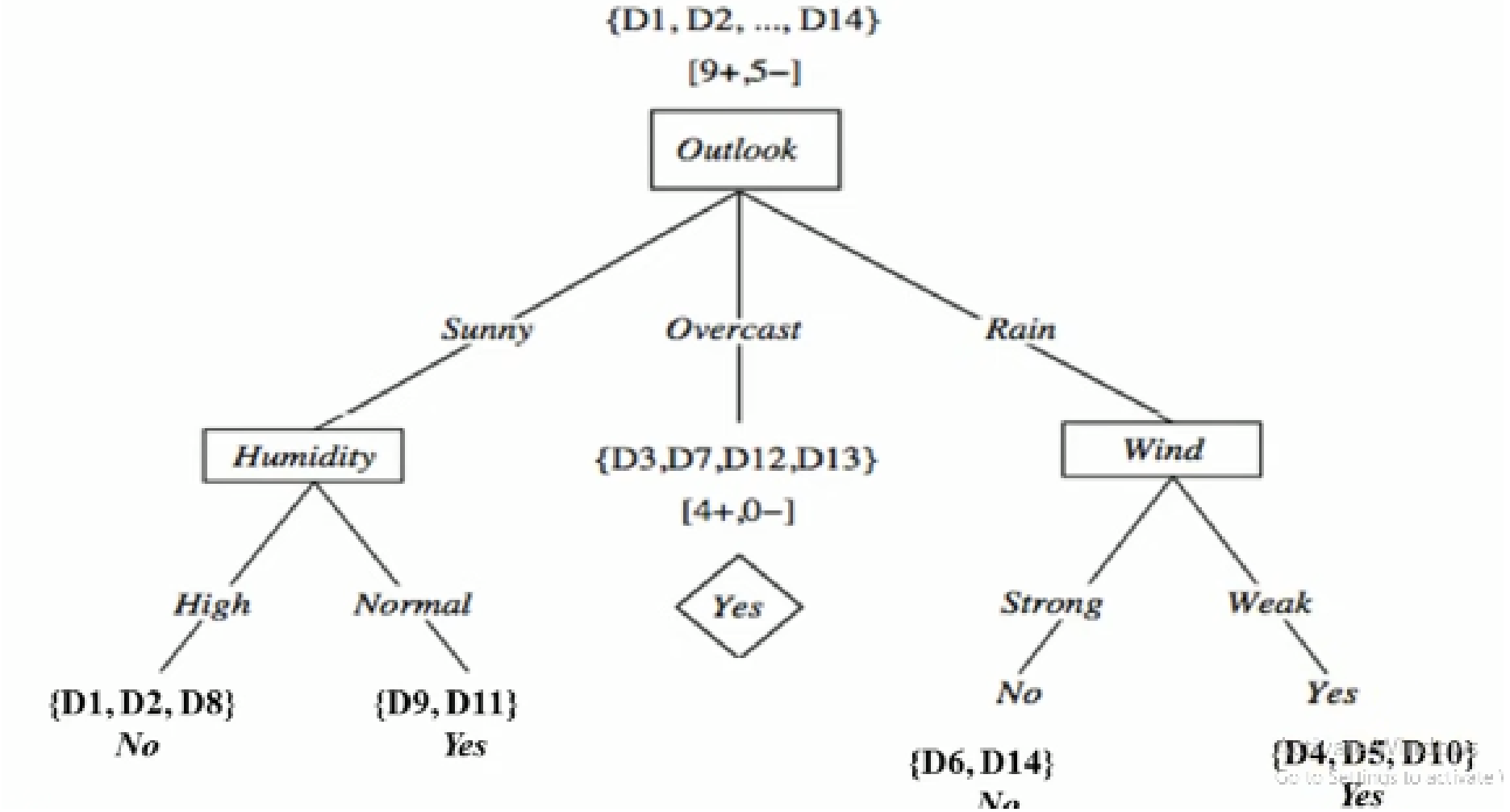
$$Gain(S, Outlook) = 0.2464$$

$$Gain(S, Temp) = 0.0289$$

$$Gain(S, Humidity) = 0.1516$$

$$Gain(S, Wind) = 0.0478$$




{D1, D2, ..., D14}
[9+,5−]
Outlook
Sunny
Overcast
Rain
Humidity
{D3,D7,D12,D13}
[4+,0−]
Wind
High
Normal
Yes
Strong
Weak
No
Yes
{D1, D2, D8}
No
{D9, D11}
Yes
{D6, D14}
No
{D4, D5, D10}
Yes

Video Tutorial

https://www.youtube.com/watch?v=coOTEc-0OGw

Video Tutorial Link for Decision Tree

Example Decision Tree

https://www.youtube.com/watch?v=JO2wiZif2OM

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- $\sum_j P_j^2$**

Advantages & Disadvantages of Decision Tree

- Advantages of the Decision Tree
  - It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
  - It can be very useful for solving decision-related problems.
  - It helps to think about all the possible outcomes for a problem.
  - There is less requirement of data cleaning compared to other algorithms.
- Disadvantages of the Decision Tree
  - The decision tree contains lots of layers, which makes it complex.
  - It may have an overfitting issue, which can be resolved using the **Random Forest algorithm.**
  - For more class labels, the computational complexity of the decision tree may increase.

Random Forest Classification

Random Forest

- Random forest is a ***Supervised Machine Learning Algorithm*** that is ***used widely in Classification and Regression problems***. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.
- One of the most important features of the Random Forest Algorithm is that it can handle the data set containing ***continuous variables*** as in the case of regression and ***categorical variables*** as in the case of classification. It performs better results for classification problems.

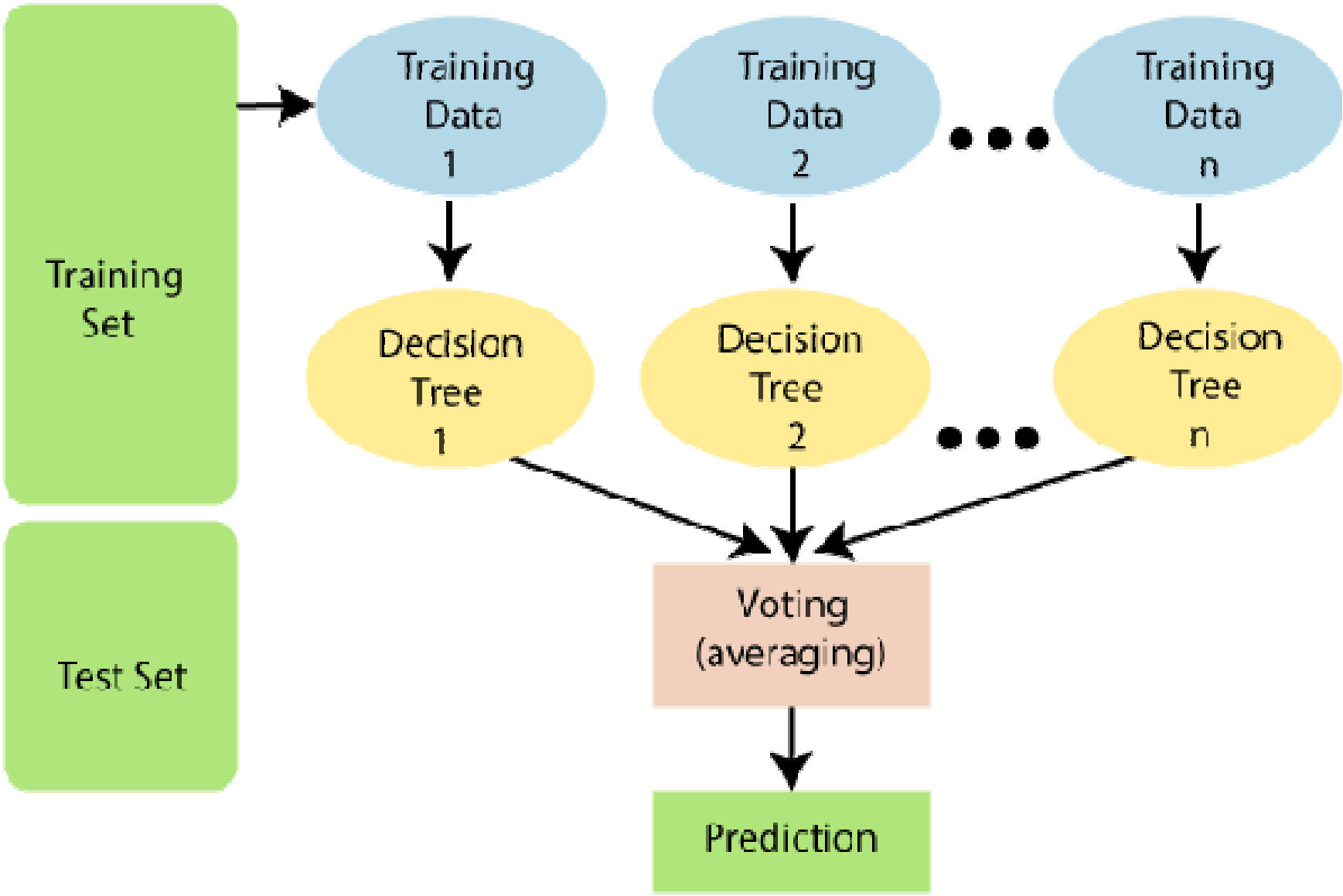
https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/

Random Forest

- It is based on the concept of **ensemble learning,** which is a process of *combining multiple classifiers to solve a complex problem and to improve the performance of the model.*
- As the name suggests, ***"Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."*** Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.
- **The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.**

Random Forest Algorithm

- **Step 1**: The algorithm select random samples from the dataset provided.
- **Step 2:** The algorithm will create a decision tree for each sample selected. Then it will get a prediction result from each decision tree created.
- **Step 3: V**oting will then be performed for every predicted result. For a classification problem, it will use **mode**, and for a regression problem, it will use **mean**.
- **Step 4**: And finally, the algorithm will select the most voted prediction result as the final prediction.


Training Set
Test Set
Training Data 1
Training Data 2
Training Data n
Decision Tree 1
Decision Tree 2
Decision Tree n
Voting (averaging)
Prediction

Assumptions for Random Forest

- Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random forest classifier:
- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

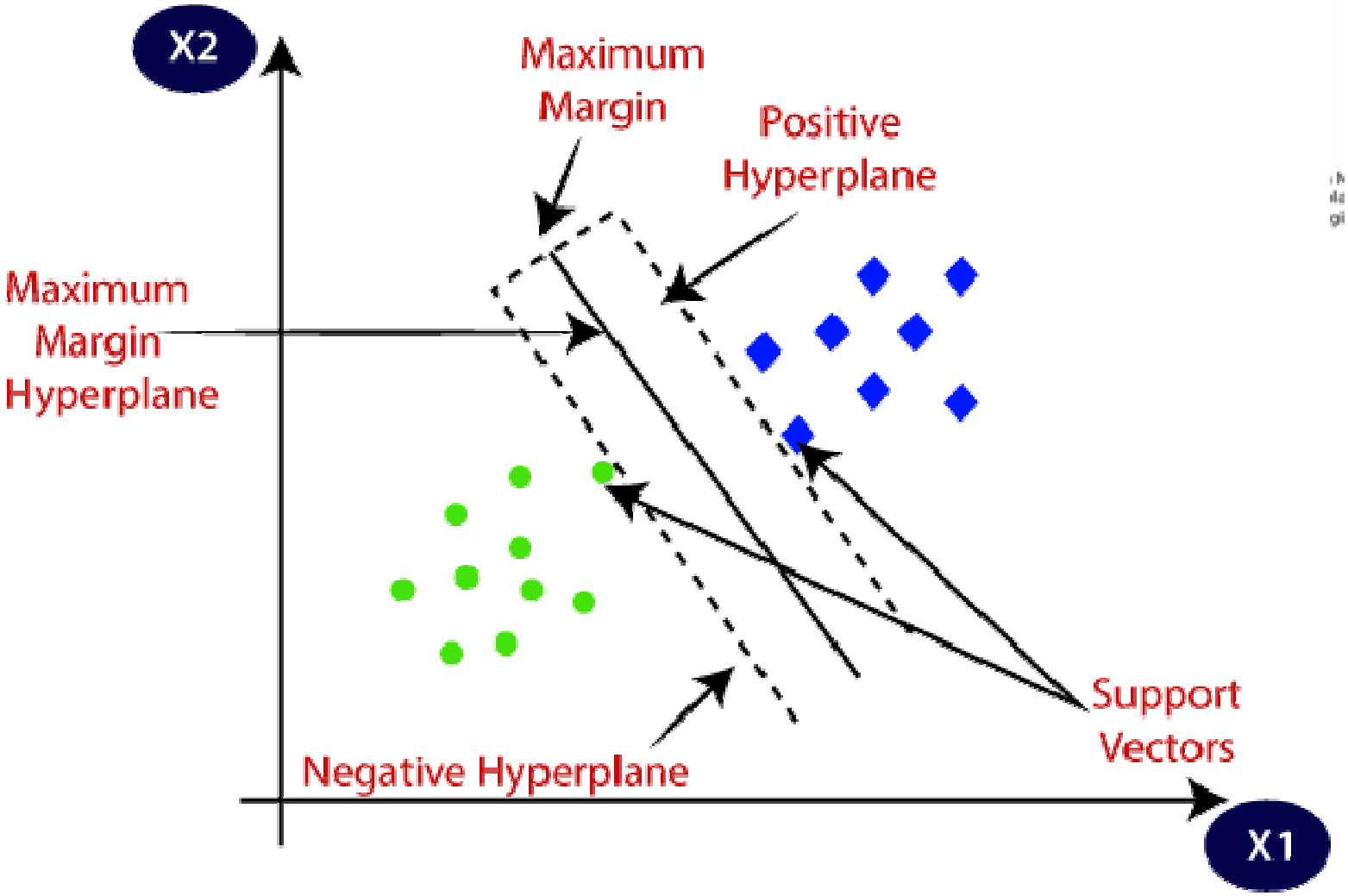
Decision trees	Random Forest
1. Decision trees normally suffer from the problem of overfitting if it's allowed to grow without any control.	1. Random forests are created from subsets of data and the final output is based on average or majority ranking and hence the problem of overfitting is taken care of.
2. A single decision tree is faster in computation.	2. It is comparatively slower.
3. When a data set with features is taken as input by a decision tree it will formulate some set of rules to do prediction.	3. Random forest randomly selects observations, builds a decision tree and the average result is taken. It doesn't use any set of formulas.

Naive Bayes Classifiers

- Naive Bayes classifiers are a collection of classification algorithms based on **Bayes' Theorem**. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.
- Video Tutorial
- https://www.youtube.com/watch?v=XzSlEA4ck2I

Support Vector Machine Algorithm

- The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.
- SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.



https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm

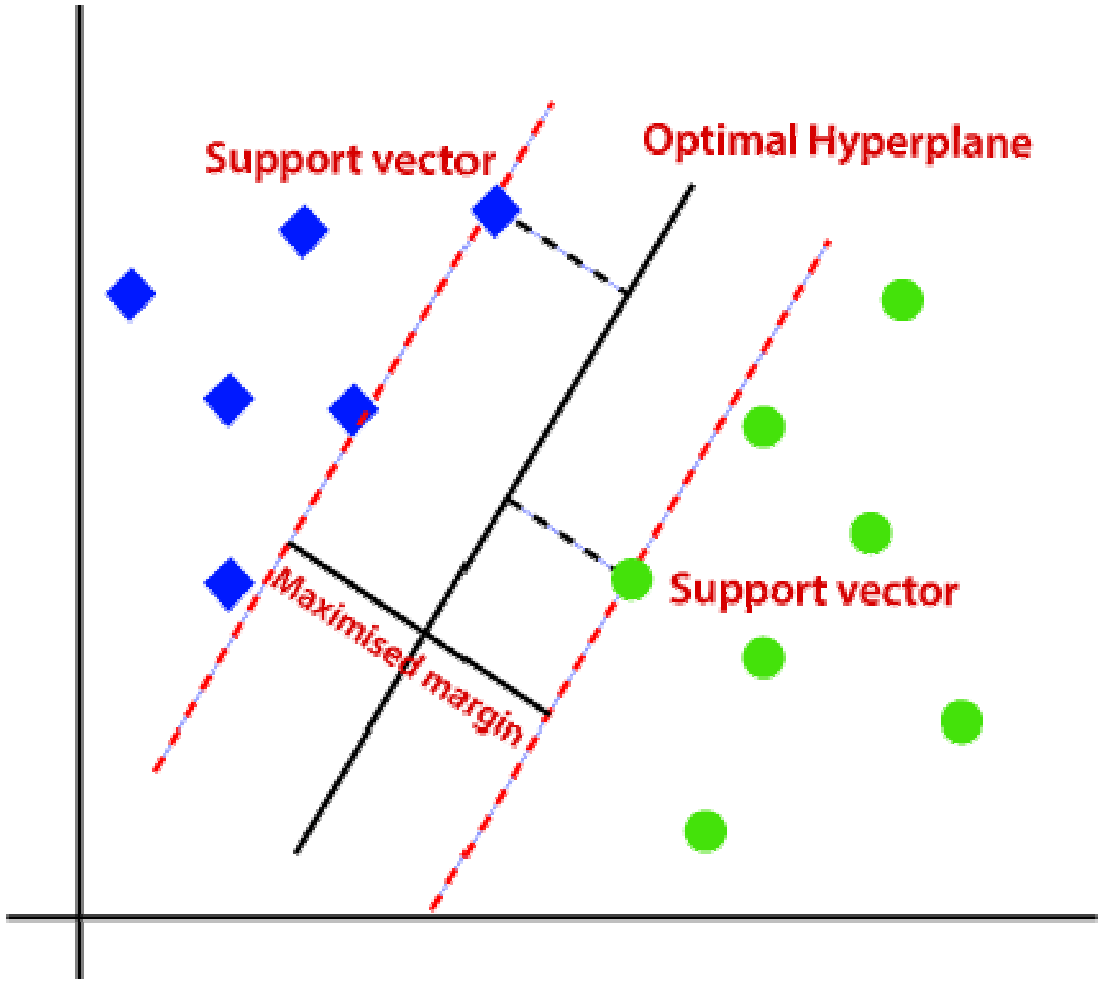
Support Vectors

- The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

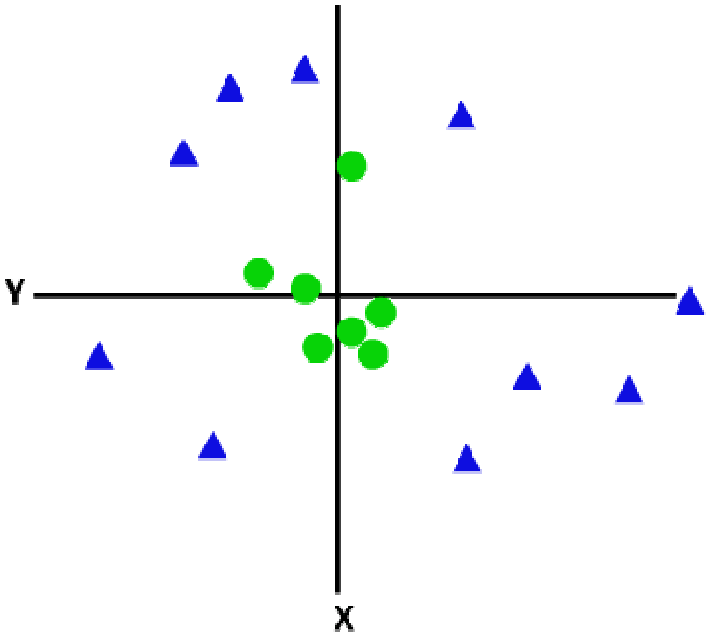
SVM Types

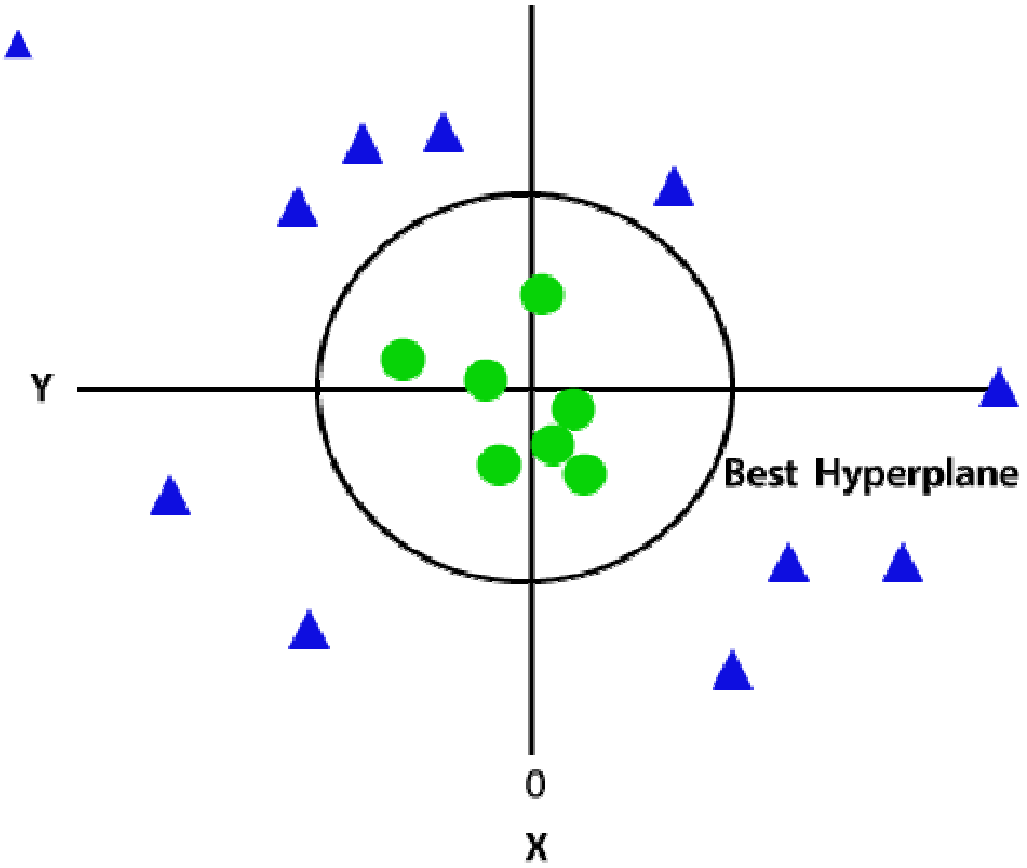
- **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
- **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

Linear SVM


Support vector
Optimal Hyperplane
Maximised margin
Support vector

Non-linear SVM





K-Nearest Neighbors (KNN)

- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.

https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

K-Nearest Neighbours (KNN)

- K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data.
- It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

KNN Algorithm

Step-1: Select the number K of the neighbors

Step-2: Calculate the Euclidean distance of **K number of neighbors**

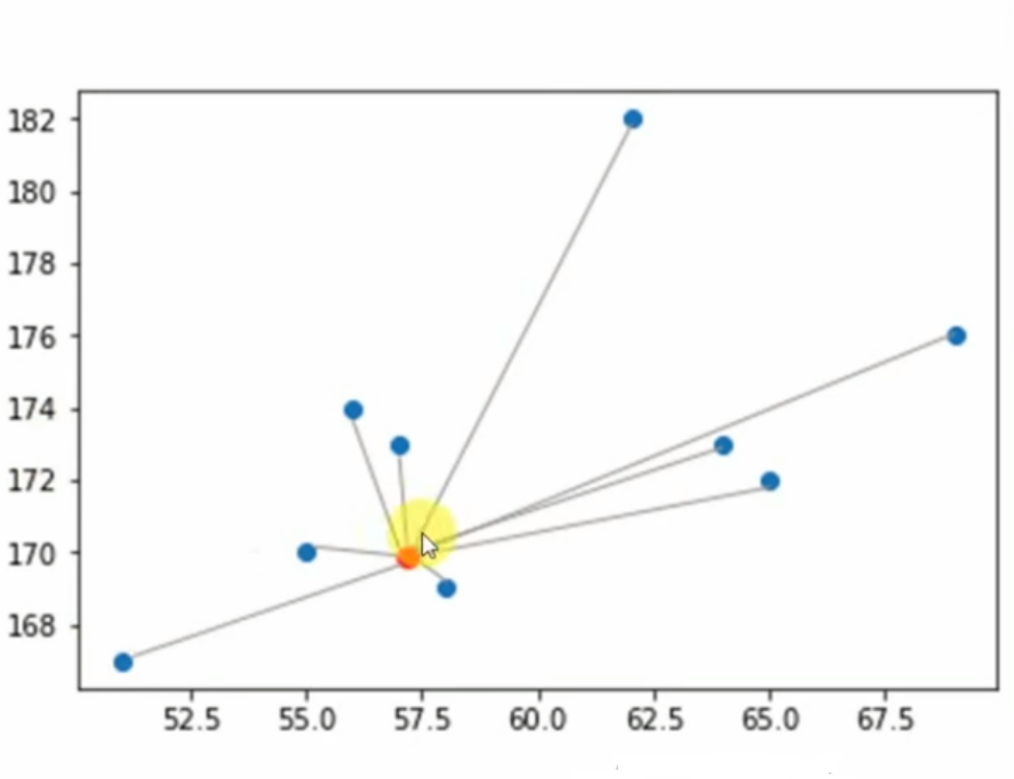
Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.

Step-4: Among these k neighbors, count the number of the data points in each category.

Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.

Step-6: Our model is ready.

Height (CM)	Weight (KG)	Class
167	51	Underweight
182	62	Normal
176	69	Normal
173	64	Normal
172	65	Normal
174	56	Underweight
169	58	Normal
173	57	Normal
170	55	Normal
170	57	?



THE DISTANCE FORMULA

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

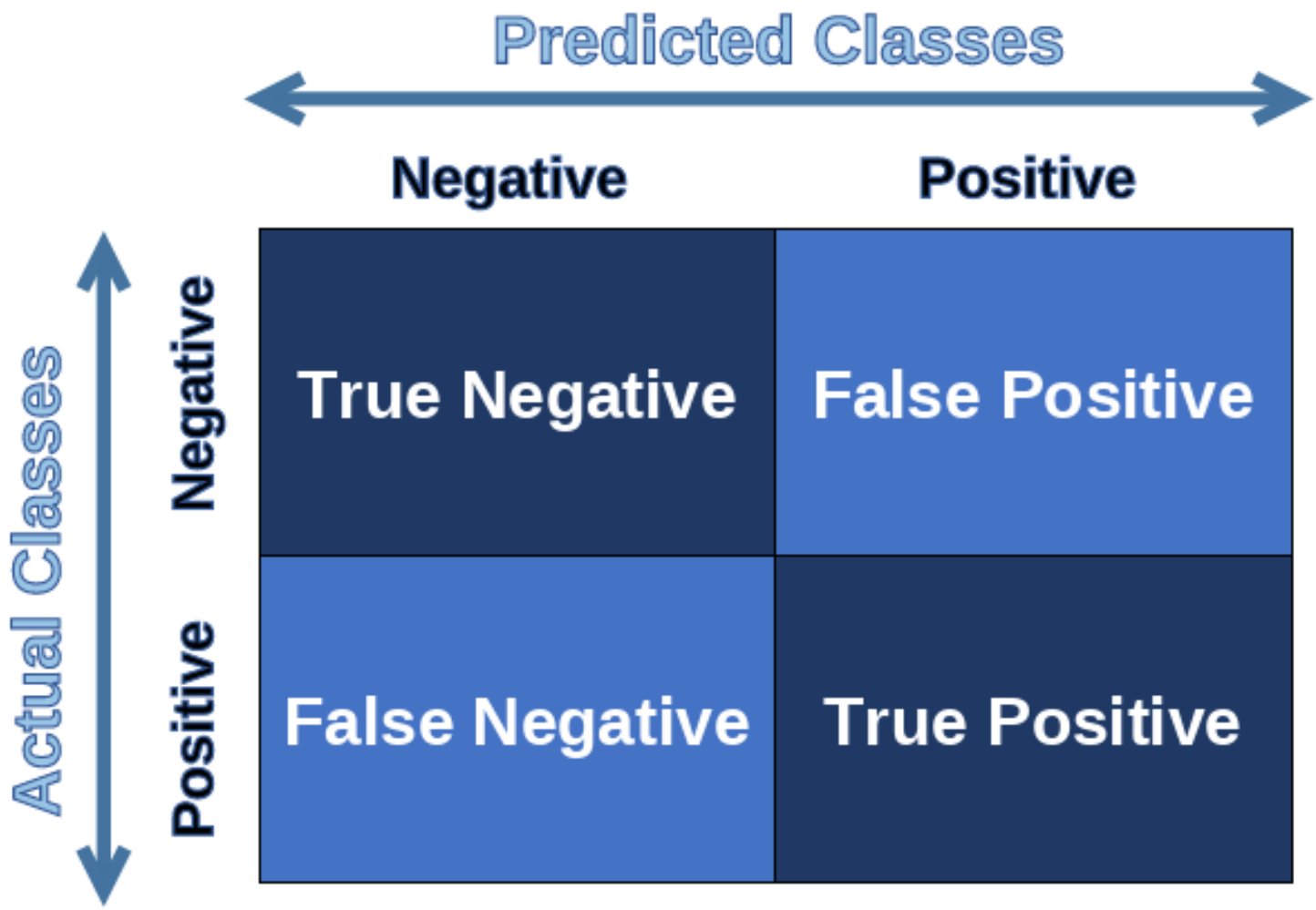
https://www.youtube.com/watch?v=HZT0lxD5h6k

Height (CM)	Weight (KG)	Class	Distance	Rank
169	58	Normal	1.4	1
170	55	Normal	2	2
173	57	Normal	3	3
174	56	Underweight	4.1	4
167	51	Underweight	6.7	5
173	64	Normal	7.6	6
172	65	Normal	8.2	7
182	62	Normal	13	8
176	69	Normal	13.4	9
170	57	?		

Evaluating Classification Models Performance

Test Case No.	Actual Answer	Model Predicted	This case is called as . . .
1	A	A	**True Positive**
2	A	Not A	**False Negative**
3	Not A	A	**False Positive**
4	Not A	Not A	**True Negative**

Confusion Matrix

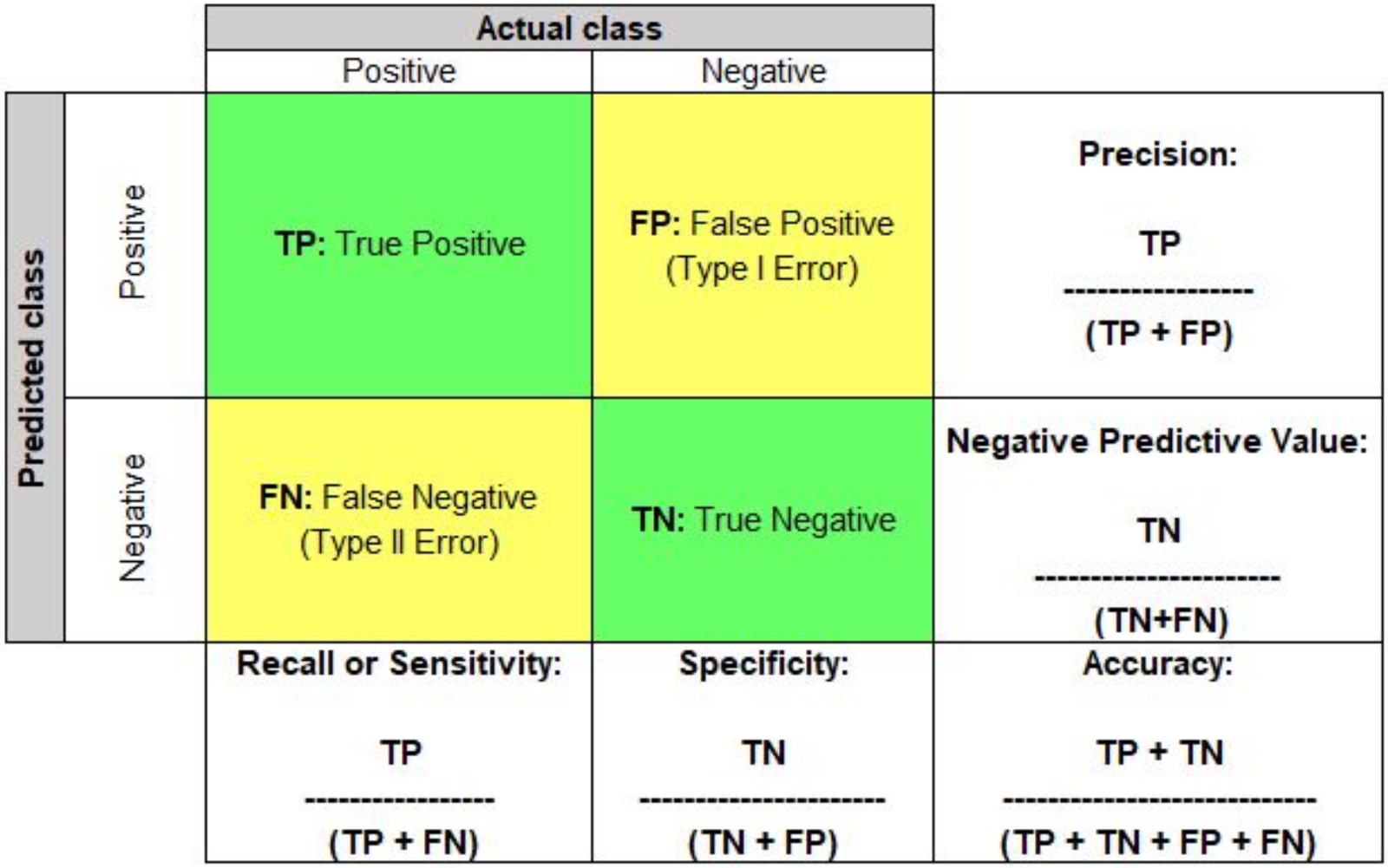


Confusion Matrix

- **True Positive (TP):** Let's say the patient was actually suffering from Covid and on doing the required assessment, the doctor classified him as a Covid patient. This is called TP or True Positive. This is because the case is positive in real and at the same time the case was classified correctly.
- **False Positive (FP):**
- Let's say the patient was not suffering from Covid and he was only showing symptoms of seasonal flu but the doctor diagnosed him with Covid. This is called FP or False Positive. This is because the case was actually negative but was falsely classified as positive.

Confusion Matrix

- **True Negative (TN):** Let's say the patient was not suffering from Covid and the doctor also gave him a clean chit. This is called TN or True Negative. This is because the case was actually negative and was also classified as negative which is the right thing to do.
- **False Negative (FN):**
- Let's say the patient was suffering from Covid and the doctor did not diagnose him with Covid. This is called FN or False Negative as the case was actually positive but was falsely classified as negative.



https://www.analyticsvidhya.com/blog/2020/12/decluttering-the-performance-measures-of-classification-models/

n = 100	Actual: No	Actual: Yes	
Predicted: No	TN: 65	FP: 3	68
Predicted: Yes	FN: 8	TP: 24	32
	73	27	

Precision & Sensitivity

- **Precision** can be defined as the number of correct outputs provided by the model or out of all positive classes that have predicted correctly by the model, how many of them were actually true.
- **Sensitivity (Recall)** is defined as the out of total positive classes, how our model predicted correctly. The recall must be as high as possible.

Specificity

- Specificity is the ratio of true negatives to all negative outcomes. This metric is of interest if you are concerned about the accuracy of your negative rate and there is a high cost to a positive outcome.

Accuracy

- It is one of the important parameters to determine the accuracy of the classification problems. It defines how often the model predicts the correct output. It can be calculated as the ratio of the number of correct predictions made by the classifier to all number of predictions made by the classifiers.