Machine Learning

Unsupervised Algorithms

Learning Objectives

By the end of this lesson, you will be able to:

- Study different types of unsupervised learning
- Analyze when to use unsupervised algorithms and discuss the different clustering types
- Calculate Singular Value Decomposition and its applications
- Examine clustering methods such as BIRCH

Business Scenario

A popular clothing company wants to analyze the sales patterns of its products to improve its marketing strategies. It has a large database of sales data but is not sure what insights can be gained from it.

The company decides to use unsupervised learning algorithms to extract previously unknown patterns from the data set and apply clustering algorithms to group similar products and identify patterns in sales trends. It also decides to use the associative technique to find relationships between different variables, such as customer demographics and product preferences.

Moreover, by detecting outliers and anomalies, it can identify which products are not selling well and need to be removed from the inventory. Principal component analysis and singular value decomposition are used to analyze the sales patterns and make predictions for future sales trends. The insights gained from this analysis can help the clothing company improve its marketing strategies and increase sales.

Unsupervised Algorithms

Unsupervised Algorithms

Unsupervised machine learning is a process used to extract patterns from the data.

It deals with unlabeled data sets.

It allows algorithms to work on their own to discover hidden information from the data without any guidance.

It classifies unsorted information according to patterns, differences, or similarities.

Unsupervised Algorithms: Example

A data scientist feeds information about elderly people admitted to the hospital. The algorithm has no input to influence categorization.

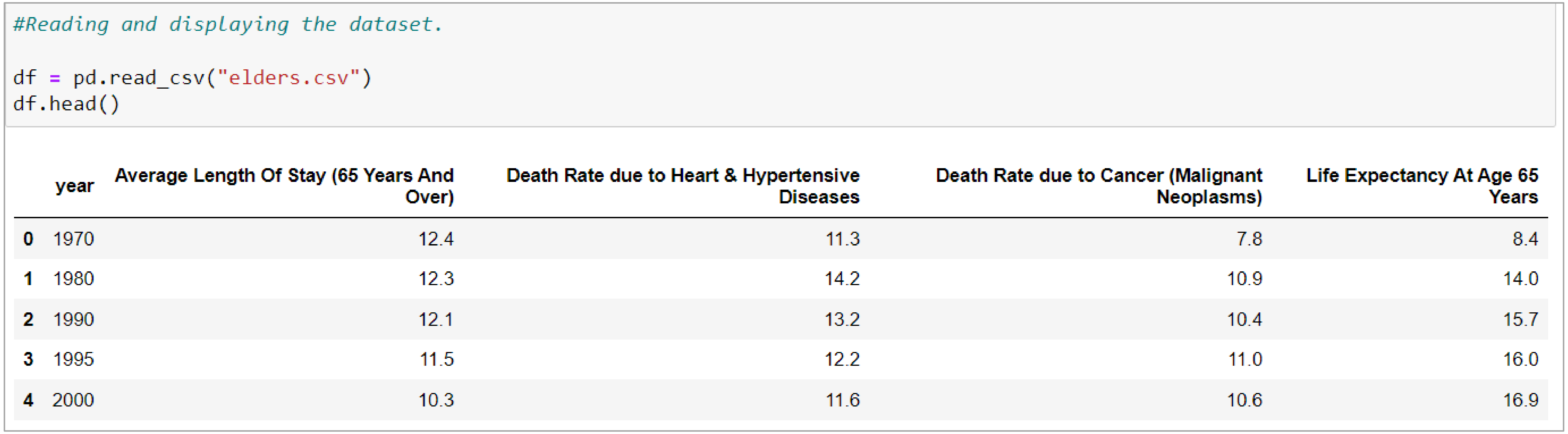
Unsupervised learning algorithms are used to detect the following in the data:

- Patterns
- Differences
- Similarities

The categorization of the given data can be done based on age, average time spent in hospital and the types of diseases they're suffering from.

Unsupervised Algorithms: Example

The final categorization of a data set helps to derive some conclusions based on different patterns generated from the data.



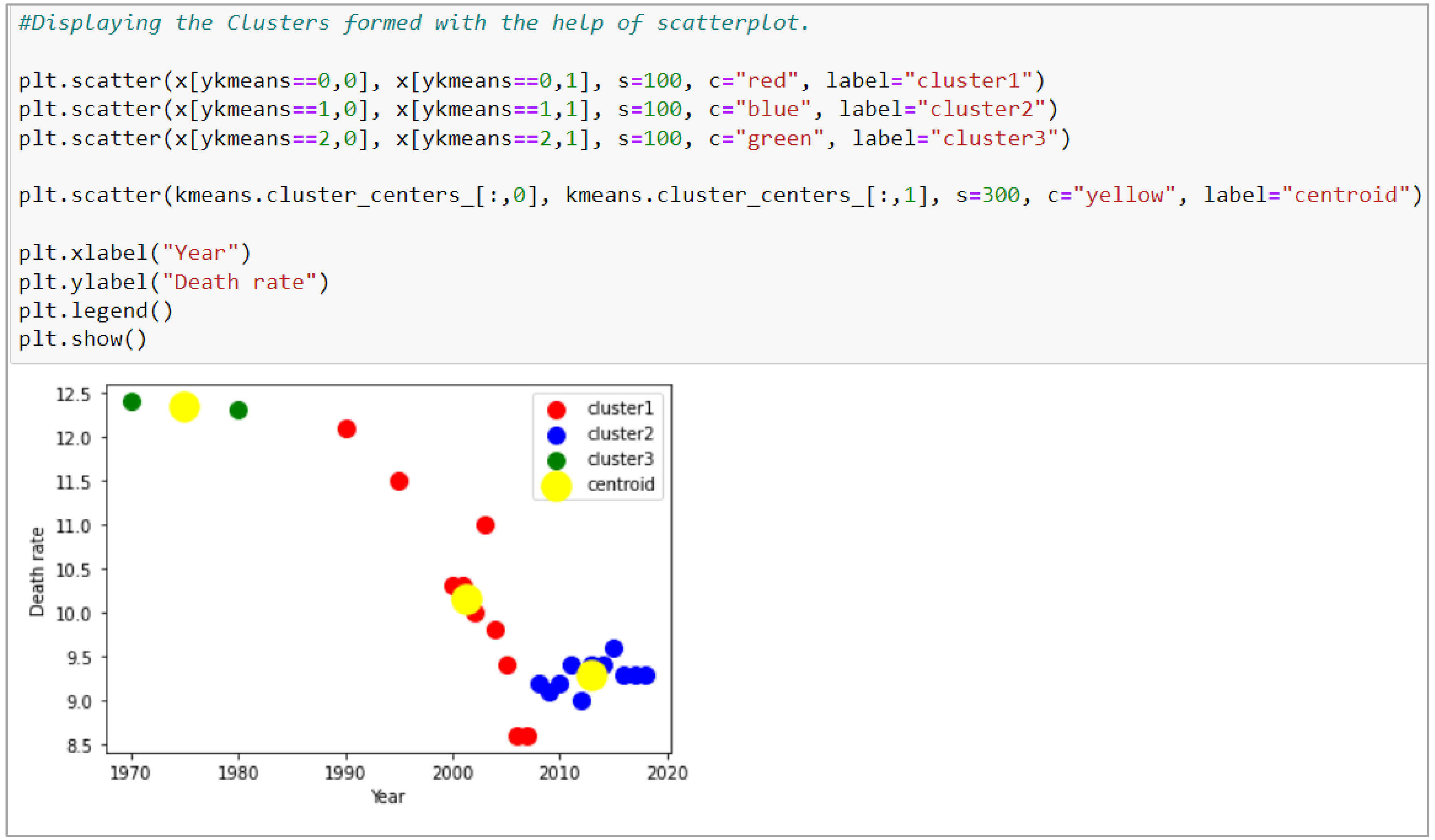
```python
#Reading and displaying the dataset.

df = pd.read_csv("elders.csv")
df.head()
```

	year	Average Length Of Stay (65 Years And Over)	Death Rate due to Heart & Hypertensive Diseases	Death Rate due to Cancer (Malignant Neoplasms)	Life Expectancy At Age 65 Years
0	1970	12.4	11.3	7.8	8.4
1	1980	12.3	14.2	10.9	14.0
2	1990	12.1	13.2	10.4	15.7
3	1995	11.5	12.2	11.0	16.0
4	2000	10.3	11.6	10.6	16.9

Unsupervised Algorithms: Example

The clusters formed according to the data set provide the following inferences:



```
#Displaying the Clusters formed with the help of scatterplot.

plt.scatter(x[ykmeans==0,0], x[ykmeans==0,1], s=100, c="red", label="cluster1")
plt.scatter(x[ykmeans==1,0], x[ykmeans==1,1], s=100, c="blue", label="cluster2")
plt.scatter(x[ykmeans==2,0], x[ykmeans==2,1], s=100, c="green", label="cluster3")

plt.scatter(kmeans.cluster_centers_[:,0], kmeans.cluster_centers_[:,1], s=300, c="yellow", label="centroid")

plt.xlabel("Year")
plt.ylabel("Death rate")
plt.legend()
plt.show()
```

Three clusters are formed based on the average time spent in the hospital.

The death rate of patients suffering from cancer and heart disease has reduced after 1970.

The average time spent in the hospital has decreased.

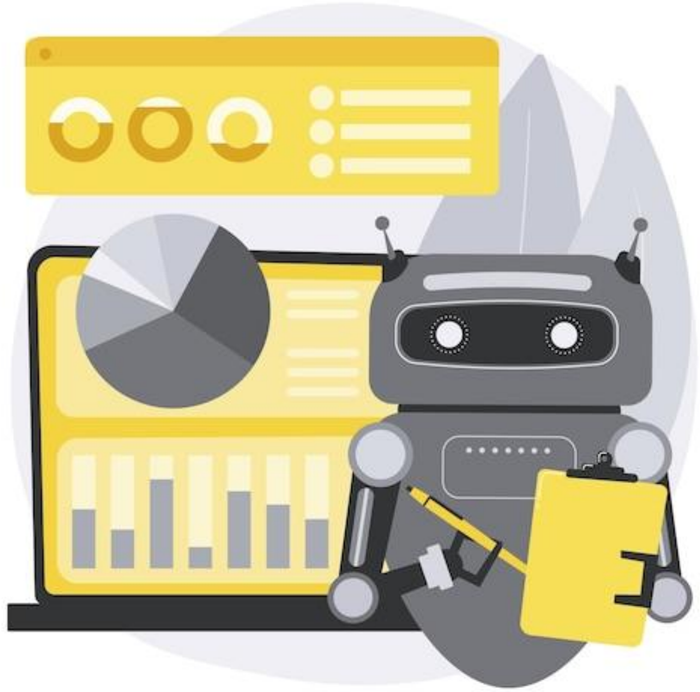
The life expectancy of a 65-year-old is higher today as compared to the earlier years.

The decreasing death rate indicates progress in medical science.

The goal of unsupervised machine learning is to learn from the data.

Unsupervised Algorithms

In unsupervised learning, the model derives insights from the data without being taught anything.



It extracts several previously unknown patterns in the dataset.

It helps find features that may be useful for categorization.

Types of Unsupervised Algorithms

Unsupervised Algorithms

There are several main types of unsupervised learning algorithms that extract patterns from unlabeled or uncategorized data.

They use clustering and association techniques to find the structure of the data set and group them based on similarity.

Clustering

Clustering techniques divide a set of data points into multiple clusters such that the data points within a cluster are similar to each other.

It aims to segregate data points with similar traits.

Association

An association rule involves finding the relationship between variables in a large data set.

It determines the dependencies of one data item over another.

It is utilized in market basket analysis.

Types of Unsupervised Algorithms

The most common clustering algorithms used in unsupervised learning algorithms are:

K-means clustering	It is an iterative clustering algorithm that groups the data set into K predefined nonoverlapping clusters.
K-medoids	It is similar to K-means but with the requirement that the cluster centers coincide with points in the data.
Density Based Spatial Clustering of Applications With Noise (DBSCAN)	It finds core samples of high density and expands clusters from them. It is suitable for a data set containing clusters of similar density.
Agglomerative clustering	It forms a hierarchy of clusters by treating each data point as a separate cluster and then progressively merging similar clusters.

Types of Unsupervised Algorithms

Some of the commonly used unsupervised learning algorithms are:

Principal component analysis (PCA)

It identifies patterns in a data set and simplifies the data set by reducing dimensionality.

Agglomerative clustering

It forms a hierarchy of clusters by treating each data point as a separate cluster and then progressively merging similar clusters.

Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH)

It is a memory-efficient online learning algorithm. It constructs a tree data structure with the cluster centroid being read off the leaf.

Apriori

It is a well-known association algorithm for locating common itemsets. It finds itemsets that fulfill minimal support criteria by continuously scanning the dataset.

When to Use Unsupervised Algorithms?

When to Use Unsupervised Algorithms?

Unsupervised algorithms are useful when the goal is to extract data from unlabeled data and the output is not known.

Parameters	Purpose
Goal	To extract patterns from unlabeled data
Type of data	Unlabeled
Supervision	Not required
Type of problems	Clustering, association, and dimensionality reduction
Output	Unknown by the user: The expected format depends on the algorithm. Example: clusters or anomalies

When to Use Unsupervised Algorithms?

Unsupervised learning models are used when:

Anomalies in the data set should be detected

Data points must be grouped based on similar traits

Data inputs in a large data set with many features must be reduced to a manageable size

The relationship between different features of a large data set should be determined

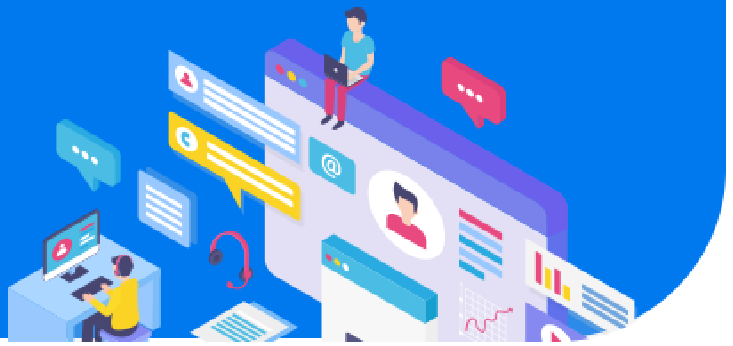
Applications of Unsupervised Learning Algorithms

Unsupervised learning algorithms find many critical real-world applications, including:

- Market basket analysis
- Fraud detection
- Malware detection
- Customer segmentation
- Targeted marketing campaigns

The ability of unsupervised algorithms to group unsorted data based on patterns, similarities and differences make them an ideal choice for a myriad of applications.

Assisted Practices



Let’s understand the topic below using Jupyter Notebook.

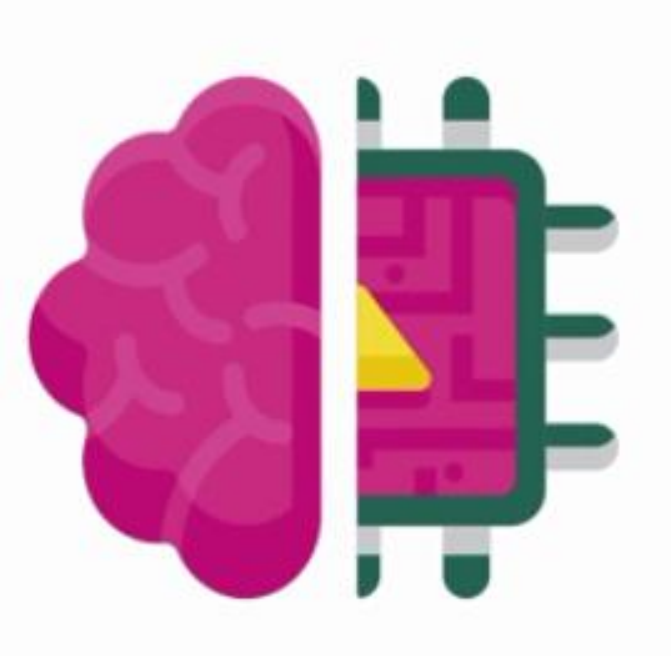
- 6.5_Visualizing Outputs

Note: Please download the pdf files for each topic mentioned above from the Reference Material section.

Performance Parameters

Performance Parameters

Machine learning is the science of creating predictive models based on data.

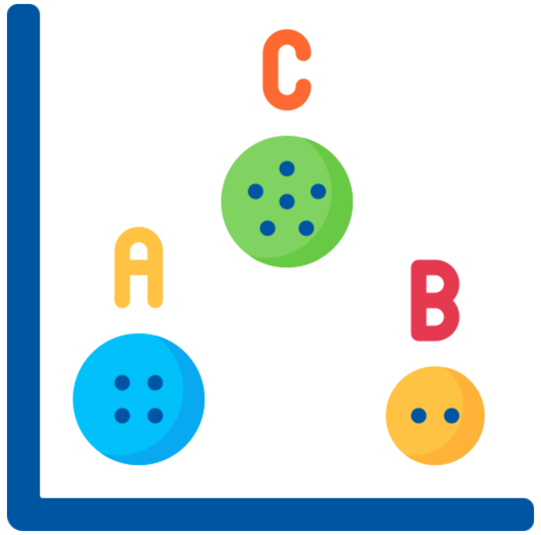


The output here is probabilistic, and hence, evaluating the performance of the model becomes important.

Various metrics are used to evaluate the performance of a model.

Performance Parameters

The parameters must be carefully chosen based on the type of problem and the type of unsupervised learning algorithm.

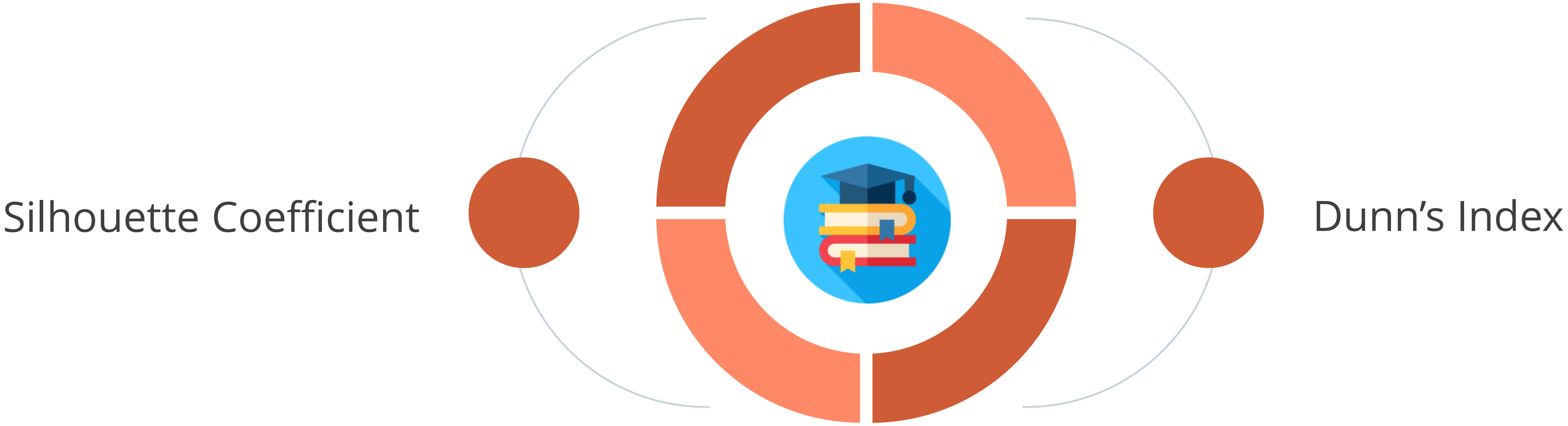


The end goal of an unsupervised learning algorithm is to create clusters of similar objects.

A model is deemed to perform well if members of a given cluster exhibit similar traits.

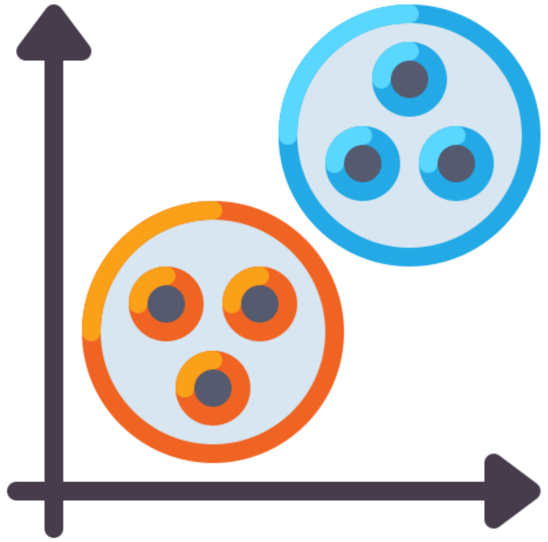
Performance Parameters

The two commonly used metrics for performance parameters are:



Silhouette Coefficient

It is a measure of the similarity of a sample to its cluster when compared to other clusters.



The value of the coefficient ranges from -1 to +1.

A lower value indicates that the sample is less cohesive to its own cluster.

A higher value indicates high cohesion.

Silhouette Coefficient

The coefficient is the average distance between a sample and all remaining points in the same cluster, divided by the average distance between the sample and all the points in the next closest cluster.

$$S = \frac{b - a}{\max(a, b)}$$

Where, a represents cohesion and b represents the separation

- a is the average distance between the sample and all the remaining points in the same cluster.
- b is the average distance between the sample and the points in the next closest cluster.

Silhouette Coefficient

The Silhouette Coefficient for a group of samples is the average of the Silhouette Coefficient for each sample.

The score lies between -1 and +1.

A score of +1 indicates a highly dense cluster, and -1 indicates an incorrect cluster.

If the score is around zero, we can infer that the clusters overlap.

The higher the score, the denser and more well-separated the cluster will be.

Dunn’s Index

Dunn’s index (DI) is used to determine sets of clusters that are cohesive and well separated.

The score lies between -1 and +1.

A higher DI value indicates better clustering.

It is obtained by dividing the minimum inter-cluster distance by the maximum intra-cluster distance.

$$\text{Dunn’s index} = \frac{\text{min (inter-cluster distance)}}{\text{max (intra-cluster distance)}}$$

Both the Silhouette Coefficient and Dunn's index are internal performance evaluation schemes where the result is based on the cluster data itself.

Clustering Types

Clustering

Clustering is an unsupervised machine learning technique that involves grouping or clustering data points.

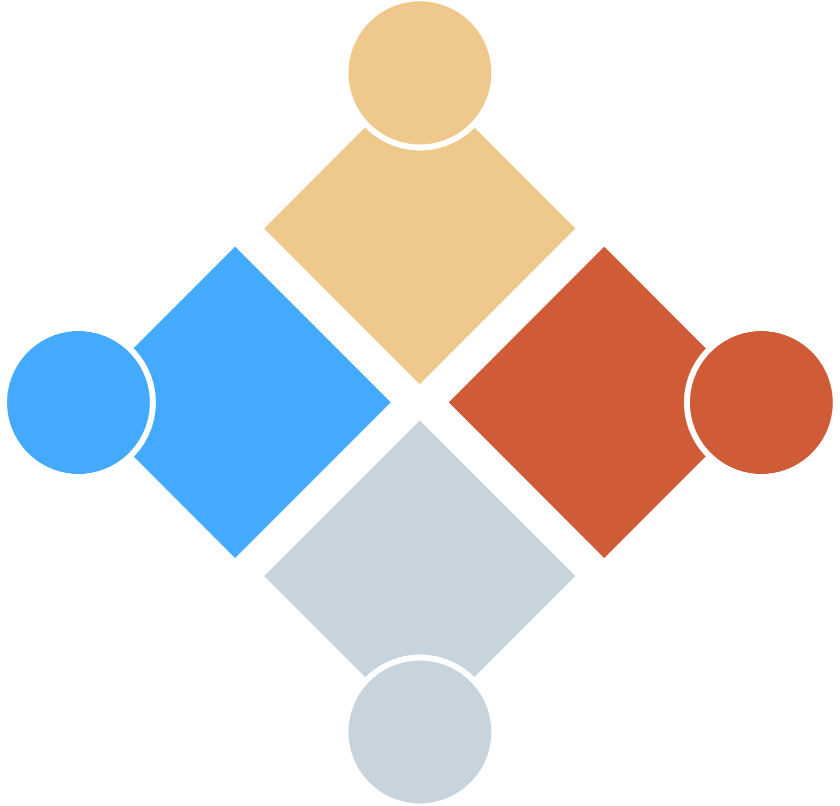
The output generated by running the clustering algorithm over a data set should be understandable and solve the business problem at hand correctly.

The goal of a clustering exercise is to classify data into distinct groups so that each group offers similar observations.

Clustering Types

There are various types of clustering methods to choose from depending on the type of problem to be solved. The most common ones are:

Density clustering



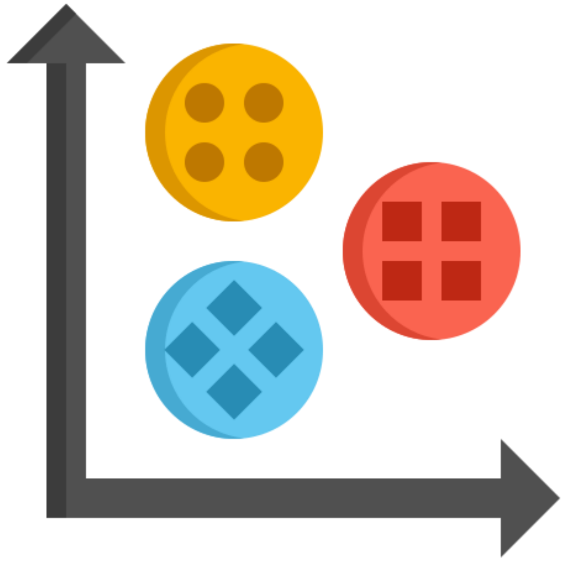
Centroid-based clustering

Distribution clustering

Hierarchical clustering

Centroid-Based Clustering

It is one of the iterative clustering algorithms in which clusters are formed based on the proximity of data points to their centroid.



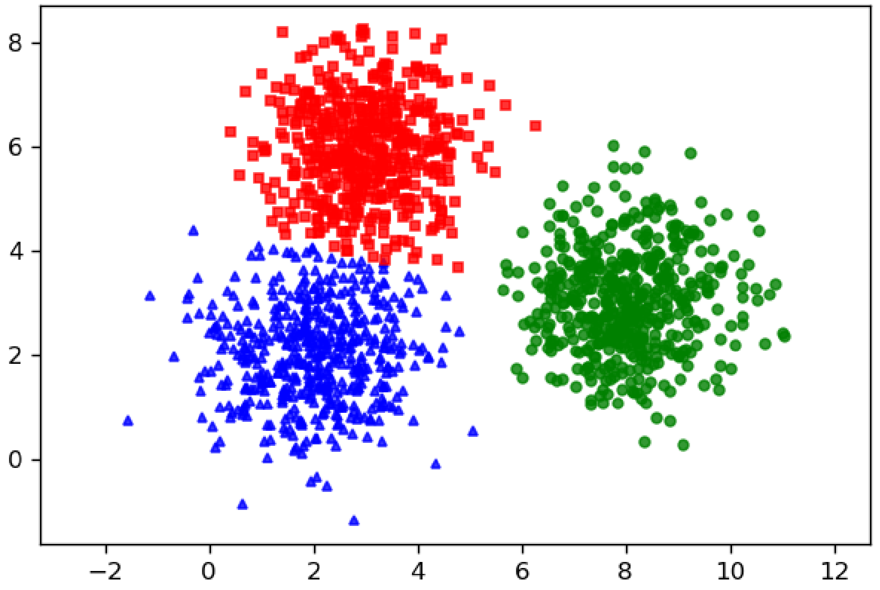
Example

K-means algorithm is one of the popular example of this clustering.

The cluster center, i.e. centroid, is positioned so that the data points are as close to the center as possible.

Density-Based Clustering

It identifies distinct clusters in the data, based on the idea that clusters in a data set are contiguous areas of high density, separated by sparse areas.



Example

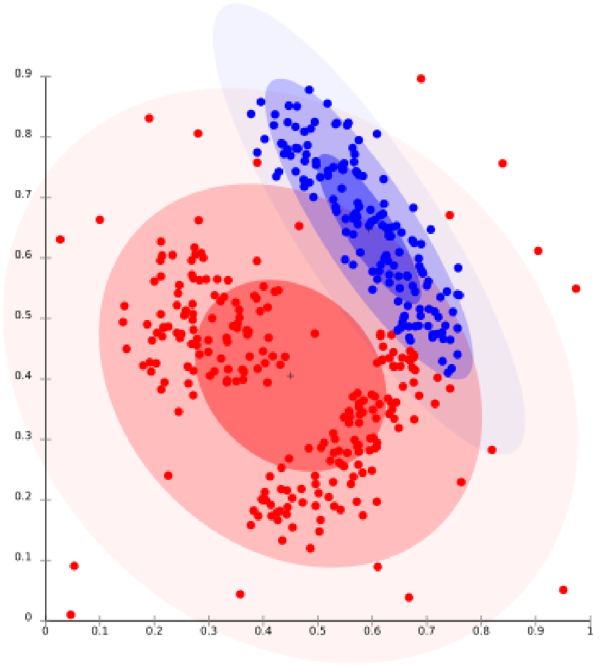
Data scientists use this clustering to identify malfunctioning servers, group genes with similar expression patterns and detect anomalies in biomedical images.

Typically, data points are sparse and the different regions are considered noise or outliers.

Distribution Clustering

This is a method of grouping data on the probability that they belong to the same distribution and grouping can be either normal or gaussian.

This results in grouping as shown in the figure:



It generates clusters that assume concisely defined mathematical models for underlying the data, which is a strong assumption for some data distributions.

Hierarchical Clustering

Hierarchical cluster analysis (HCA) groups data points in a hierarchical fashion. At each level of the hierarchy, the algorithm merges the two most similar data points.

It adopts either of the following approaches for grouping data:

Bottom-to-top approach

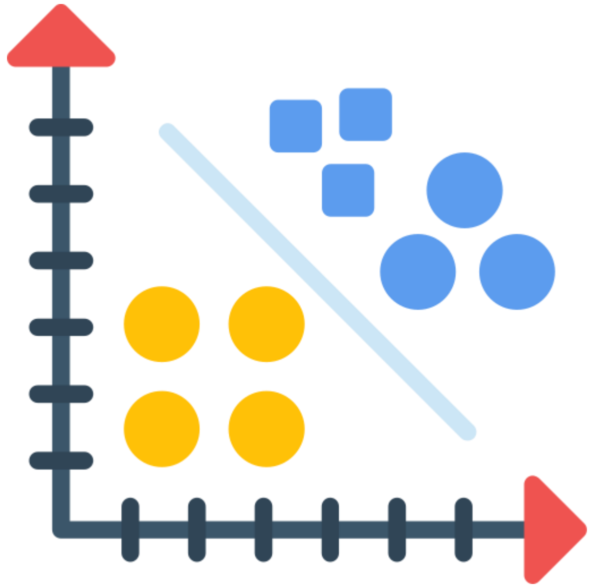
Agglomerative Hierarchical Cluster Analysis

Top-to-bottom approach

Divisive Hierarchical Cluster Analysis

Hierarchical Clustering

Every data point in a data set is deemed to be a cluster first.



As similarities are observed between the closest pairs of data points, objects are added to a specific cluster.

The endpoint of the algorithm when each cluster in a set is distinct from every other cluster in the data set and the objects within each cluster are similar to each other.

Hierarchical Clustering

Consider a data set of n different types of animals.

01 Assume that each animal is a distinct cluster by itself, that is, n clusters.

02 Take the two closest data points and make them into a cluster. Now, there are n-1 clusters.

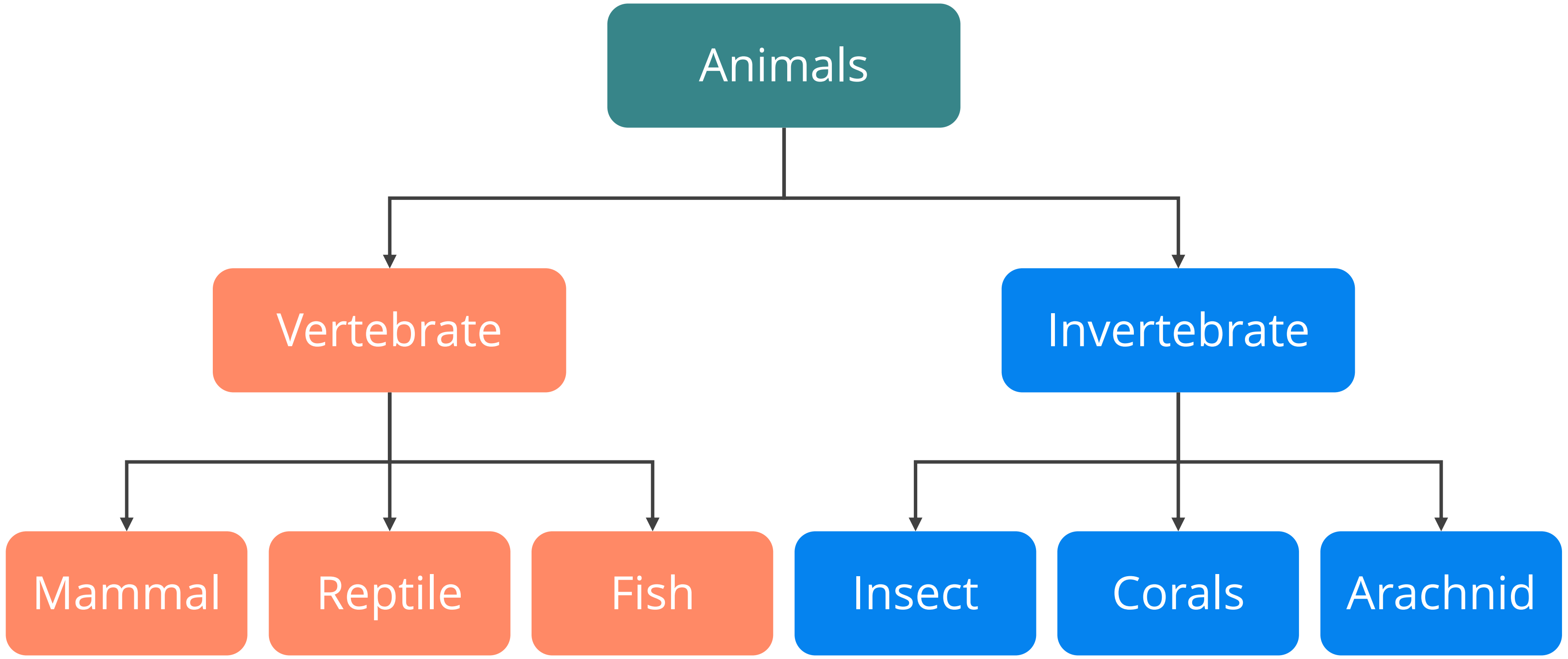
03 Repeat the process as mammals are grouped into one cluster, reptiles into another, fish into a third cluster and so on.

04 Group mammals, reptiles and fish into the vertebrate cluster and insects, corals and arachnids into the invertebrate cluster.

Finally, there is one large cluster of animals.

Hierarchical Clustering

Hierarchical clustering results in the creation of a tree-shaped structure known as a dendrogram.

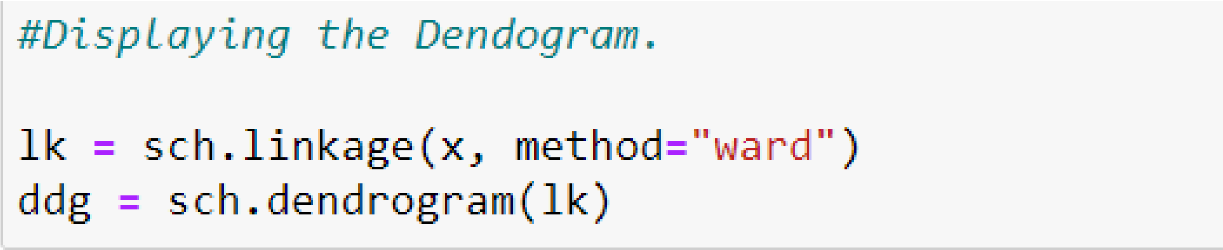


Dendrogram is a visual interpretation of hierarchical connection of items.

The goal is to find the best approach to assign items to a cluster.

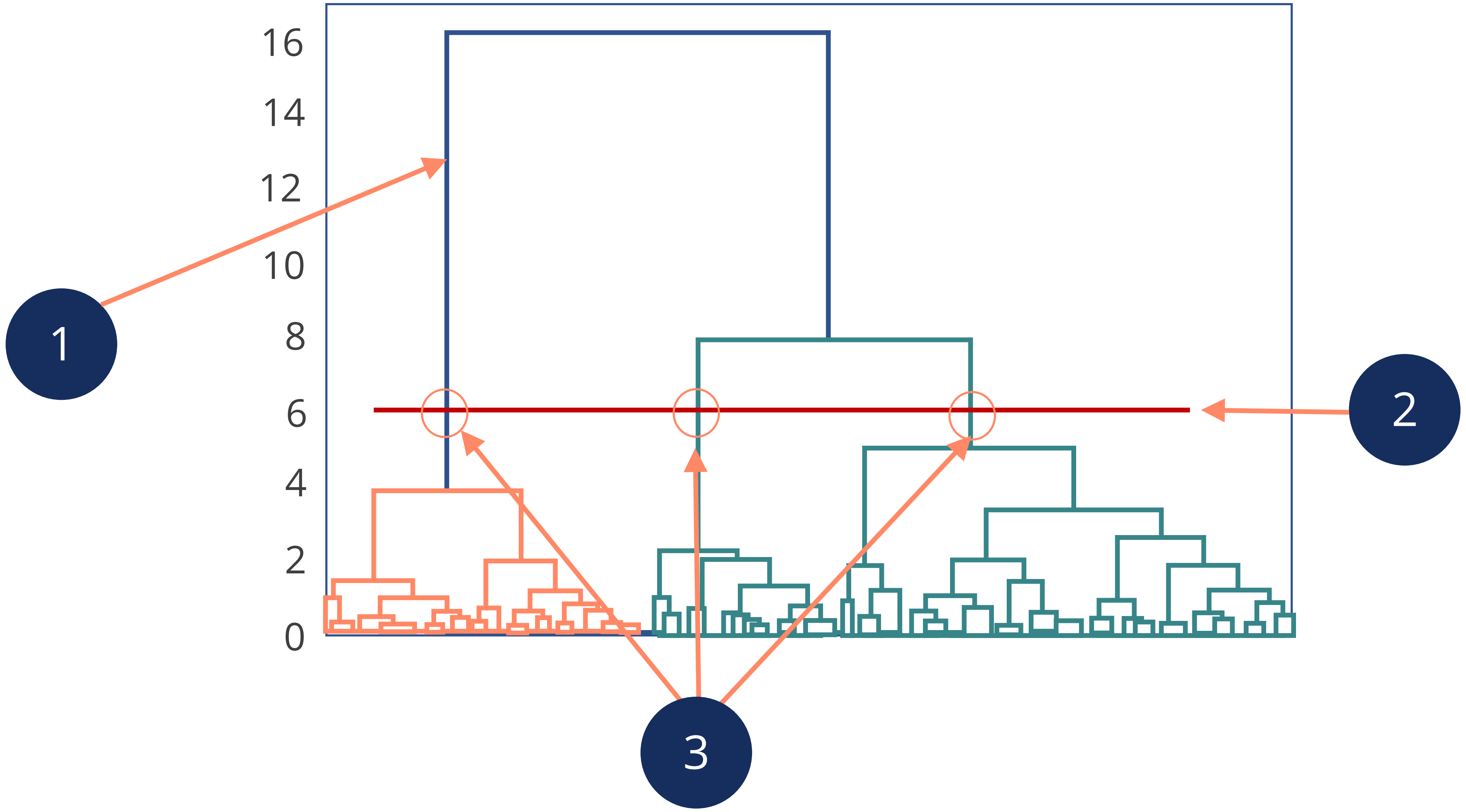
Choosing the Number of Clusters

Consider the following diagram:



```
#Displaying the Dendogram.

lk = sch.linkage(x, method="ward")
ddg = sch.dendrogram(lk)
```



Choosing the Number of Clusters

To choose the number of clusters to be created:

1. Identify the longest line that traverses the maximum vertical distance without intersecting any of the merging points in the dendrogram.
2. Draw a horizontal line where the line can traverse the maximum vertical distance without intersecting the merging point.
3. The number of vertical lines it intersects is the optimal number of clusters.

The optimal number of clusters is 3.

Applications of Hierarchical Cluster Analysis

Hierarchical cluster analysis can be used to track the spread of viruses such as COVID-19.

If data such as location of viral detection and number of infected people is extracted, hierarchical cluster analysis helps to identify the origin of the spread of the virus.

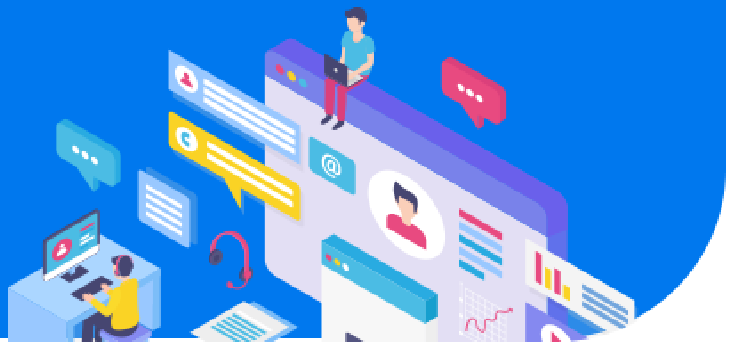
Applications of Hierarchical Cluster Analysis

Hierarchical cluster analysis can help with customer segmentation activities.

Without any input, hierarchical cluster analysis can help a company or industry segment their customers to understand their behaviors better.

This is a crucial activity in product design and marketing.

Assisted Practices



Let’s understand the topic below using Jupyter Notebook.

- 6.9_Applying Hierarchical Clustering

Note: Please download the pdf files for each topic mentioned above from the Reference Material section.

ASSISTED PRACTICE

K-Means Clustering

K-Means Clustering

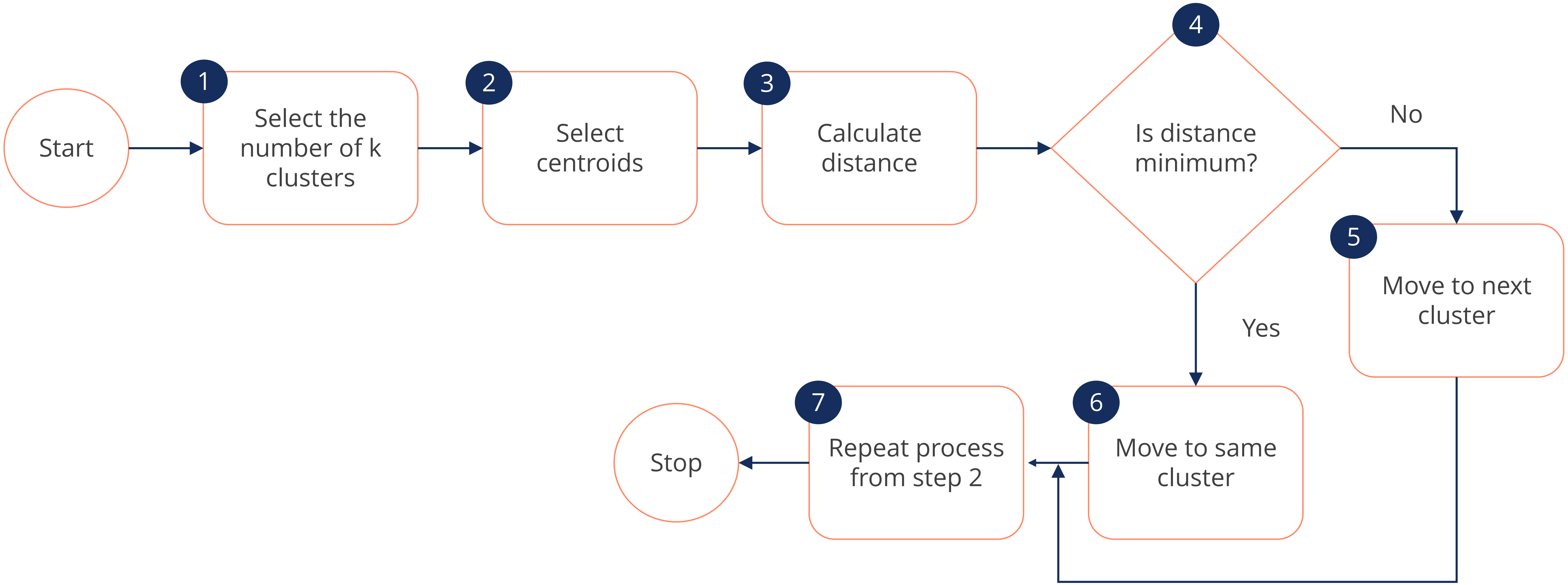
The k-means clustering algorithm is a popular unsupervised machine learning algorithm.

It groups unlabeled data into clusters by identifying the K number of centroids.

It assigns every data point to the closest cluster by calculating and using the pairwise Euclidean distance between points.

K-Means Clustering

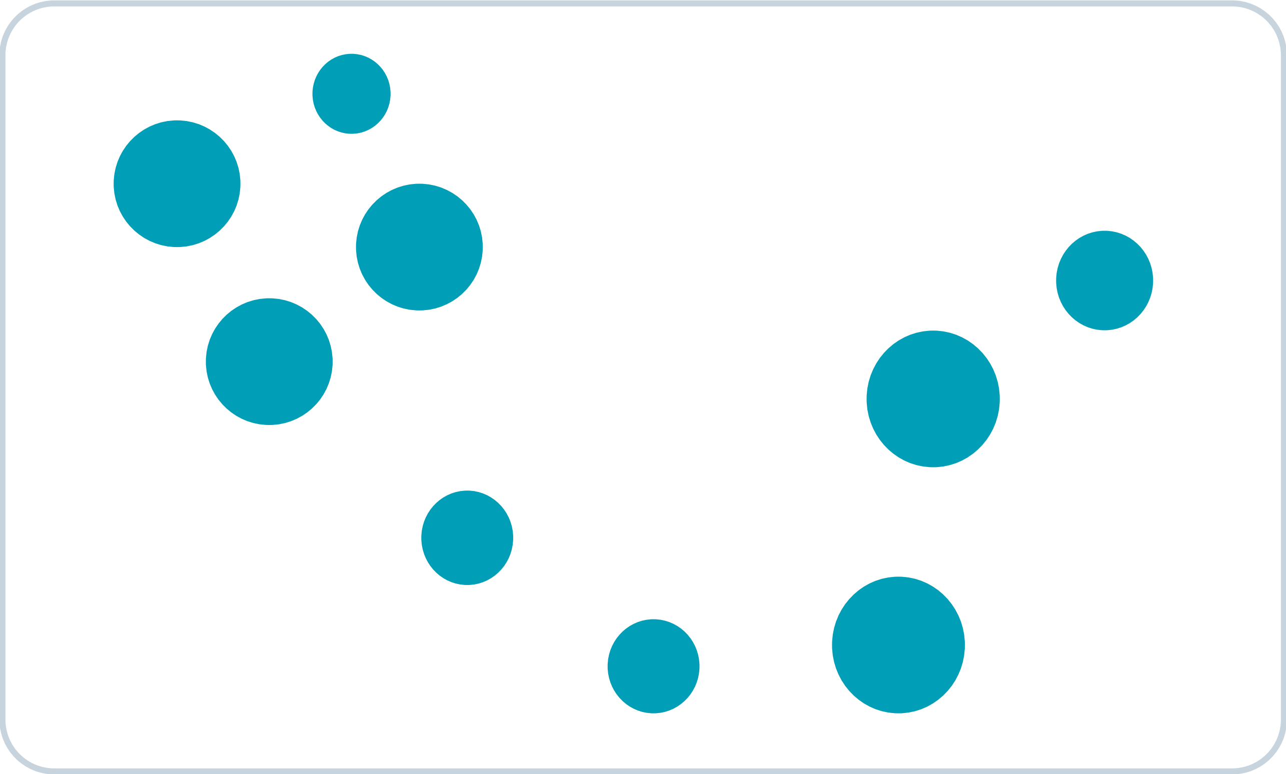
A flowchart for k-means clustering can be seen here:



Example: The K-means algorithm is a well-known example of this type of clustering.

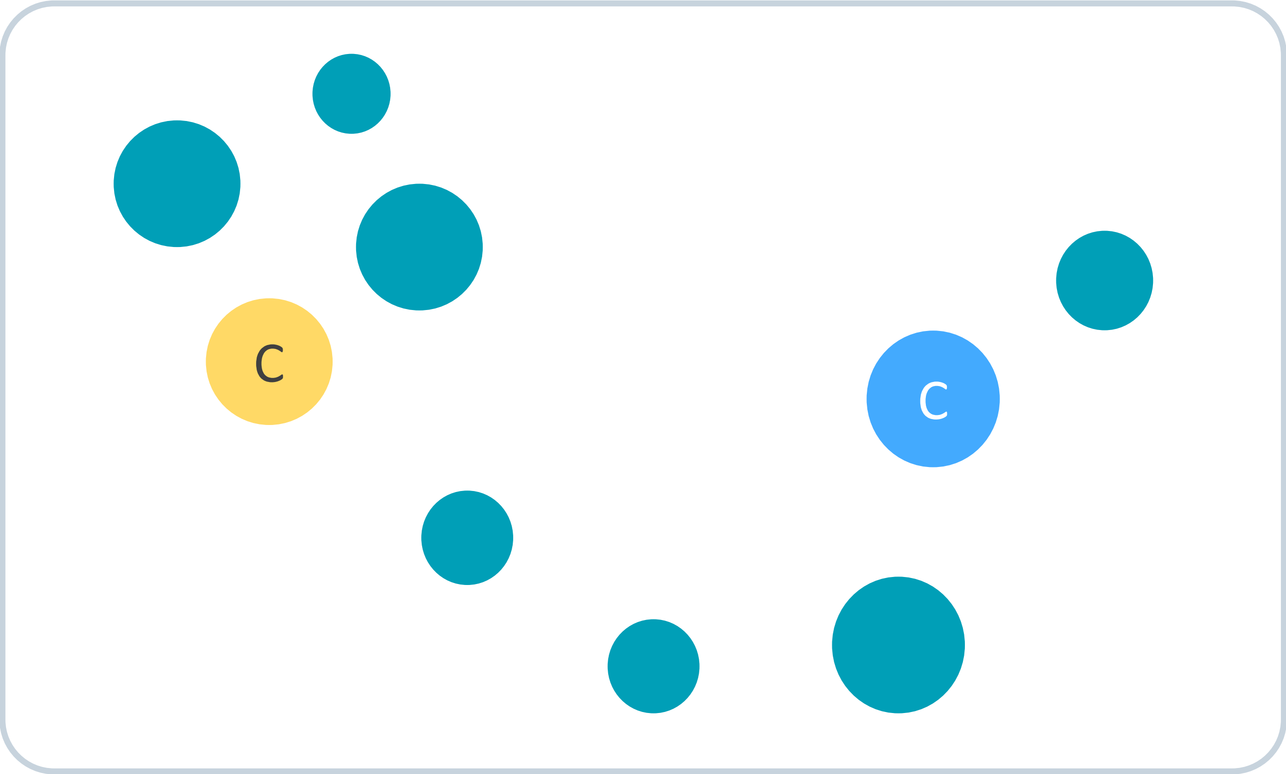
K-Means Clustering

Step 1: Select the number of k clusters. Take k = 2.



K-Means Clustering

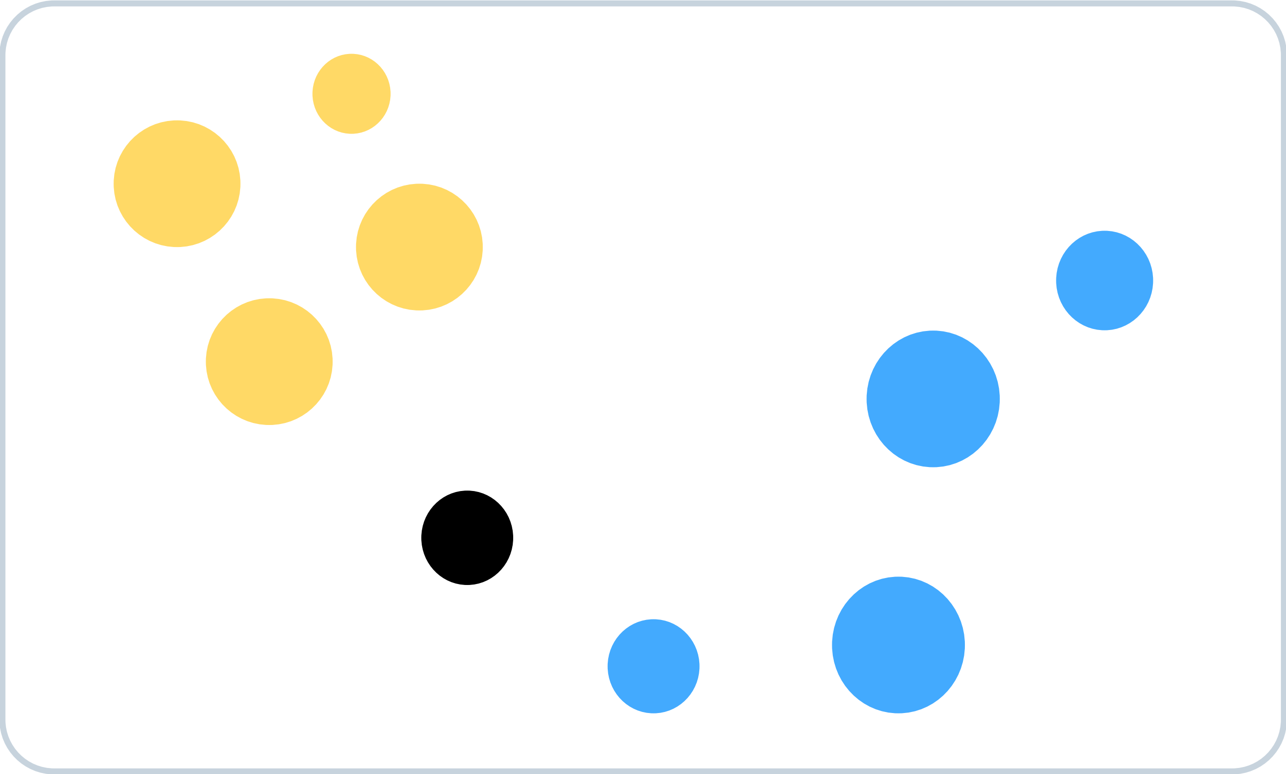
Step 2: From the data set, select k number of random points as centroids.



Since the value of k was 2, two random centroids, yellow and blue, are considered.

K-Means Clustering

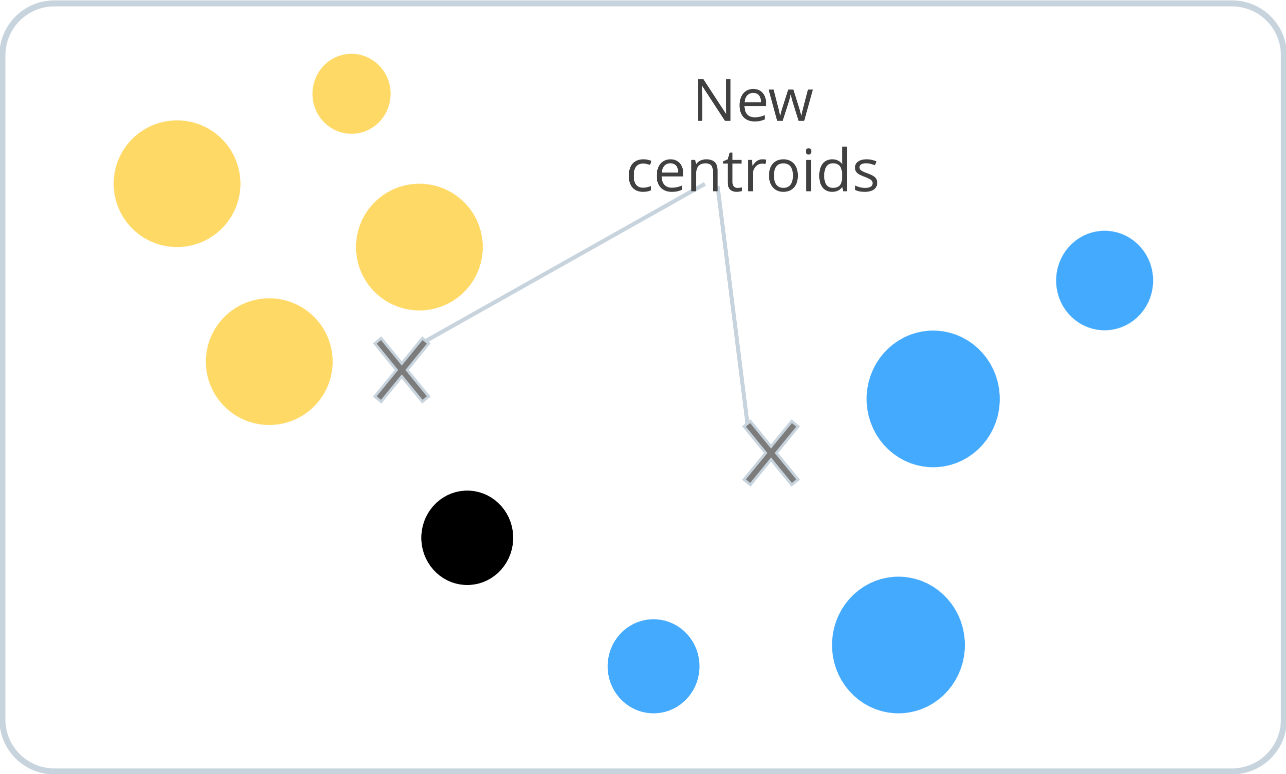
Step 3: Allot all data points to the nearest available cluster centroid.



All data points are assigned to the nearest centroid, either blue or yellow.

K-Means Clustering

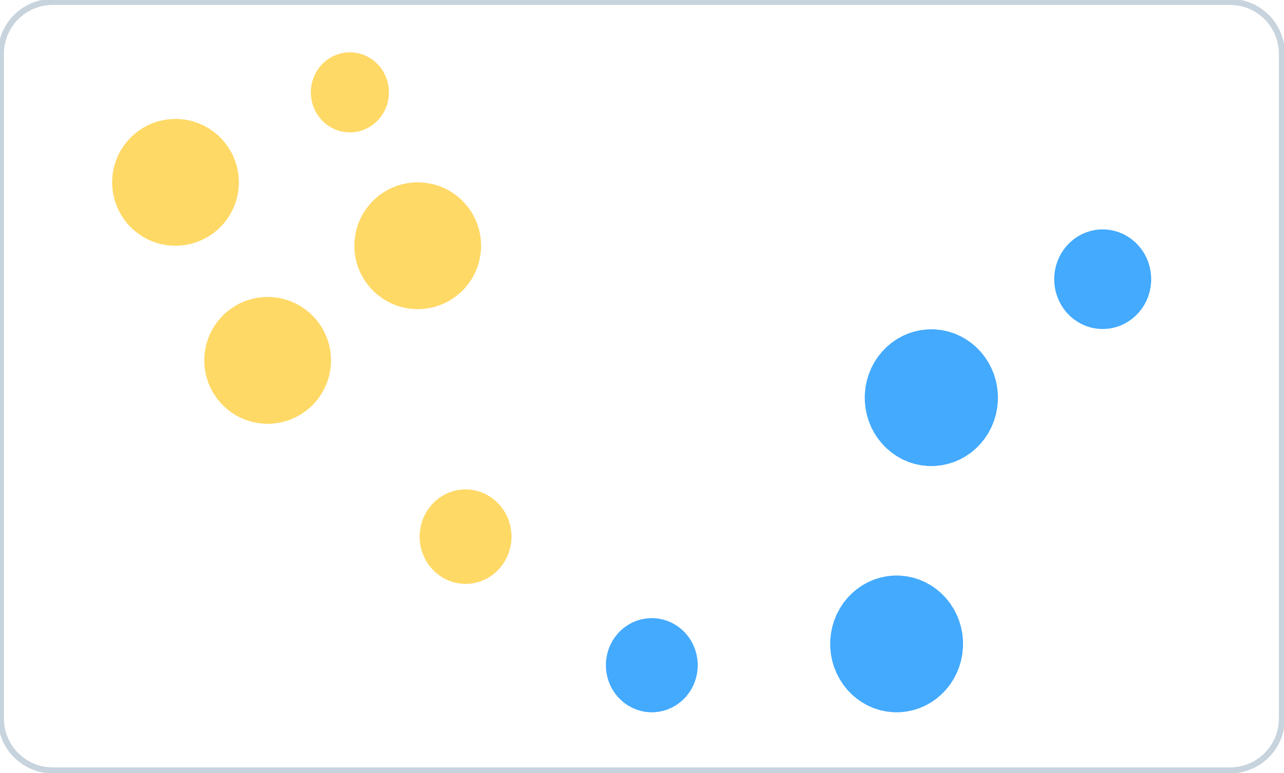
Step 4: Calculate and place the newly generated centroid of each cluster.



A centroid is generated considering the newly formed clusters.

K-Means Clustering

Step 5: Reassign each data point to the new closest centroid. If no reassignment happens, then stop, else go to step 4.



The black-colored datapoint got assigned to the yellow cluster.

K-means rapidly forms clusters of data points from which one can derive inferences.

K-Means Clustering

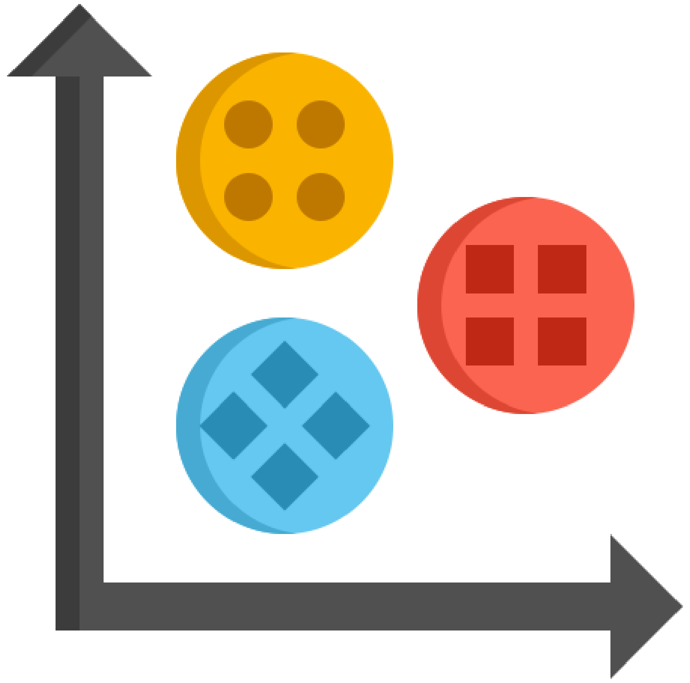
This algorithm can be applied for:

Insurance fraud detection

Crime locality detection

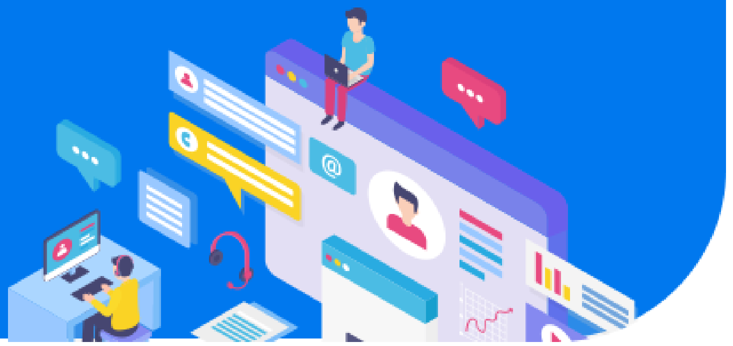
K-Means Clustering Algorithm

The k-means clustering algorithm is a centroid-based algorithm that groups unlabeled data into a K number of clusters, each with one centroid.



It requires the number of clusters to be predefined.

Assisted Practices



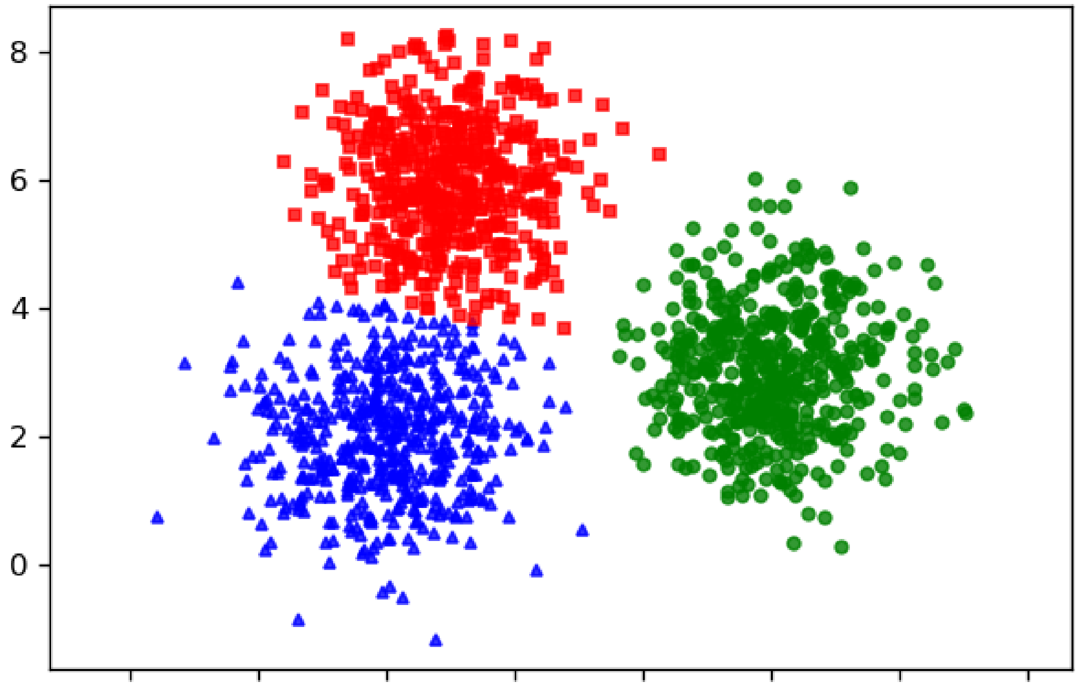
Let’s understand the topic below using Jupyter Notebook.

- 6.12_Applying K-Means Clustering

Note: Please download the pdf files for each topic mentioned above from the Reference Material section.

K-Medoids Algorithm

K-Medoids Algorithm



1. The k-medoids algorithm is a classical partitioning technique where a data set is grouped into K number of clusters.

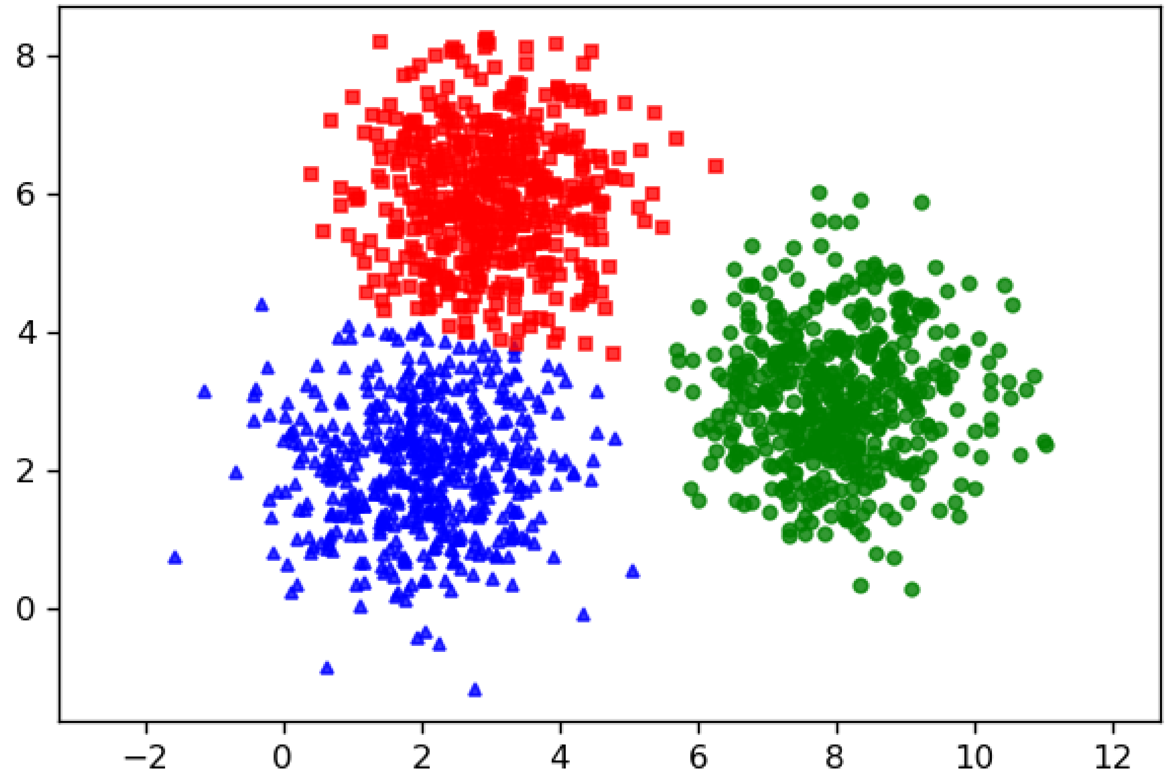
2. k is predefined before executing the algorithm. The value of k can be validated by silhouette methods.

3. Medoids are representative objects of a data set or a cluster within a data set whose sum of dissimilarities to all the objects in the cluster is minimal.

4. This algorithm is similar to the k-means clustering algorithm and uses the concept of dissimilarity to form clustering groups.

K-Medoids Algorithm

It is used when the centroid or the mean cannot be derived from similar data points.



The algorithm is more robust against noise and outliers as it minimizes general pairwise dissimilarities, instead of a sum of squared Euclidean distances.

K-Means vs. K-Medoids

Both the algorithms are divisional, that is, the algorithms break the data set into groups.

K-means attempts to decrease the total squared error whereas k-medoids decreases the sum of dissimilarities between points designated to be in a cluster.

Unlike the k-means algorithm, k-medoids chooses data points as centers.

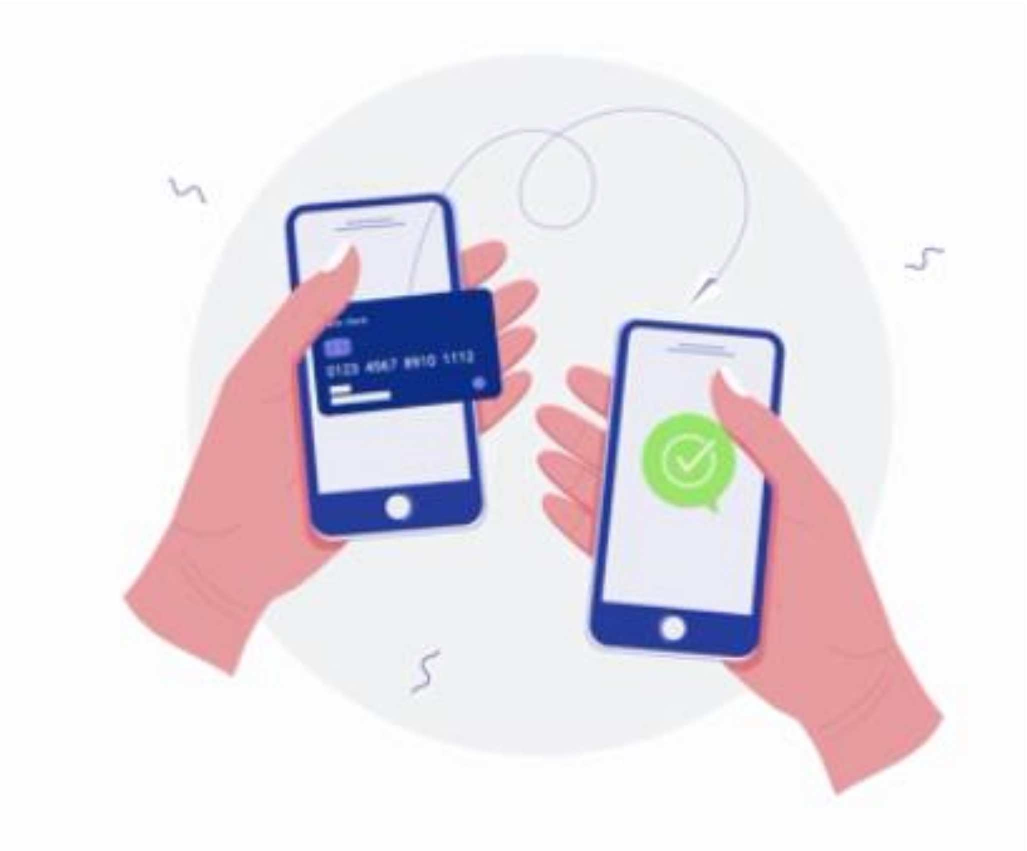
K-medoids is very flexible and can be used with any similarity measure, whereas k-means fails to converge.

K-means must only be used with distances that are consistent with the mean.

Outliers

Outliers

An outlier is any data point that differs greatly from the other observations in a data set.



Example

If a customer's usual transactions were for less than $1,000, however, there was only one transaction of $10,000, it is an outlier.

Outliers can exist for reasons such as data entry errors or intentional manipulation of information.

Outliers

Outliers are broadly divided into two categories:

Univariate

Data point containing the maximum value in one variable

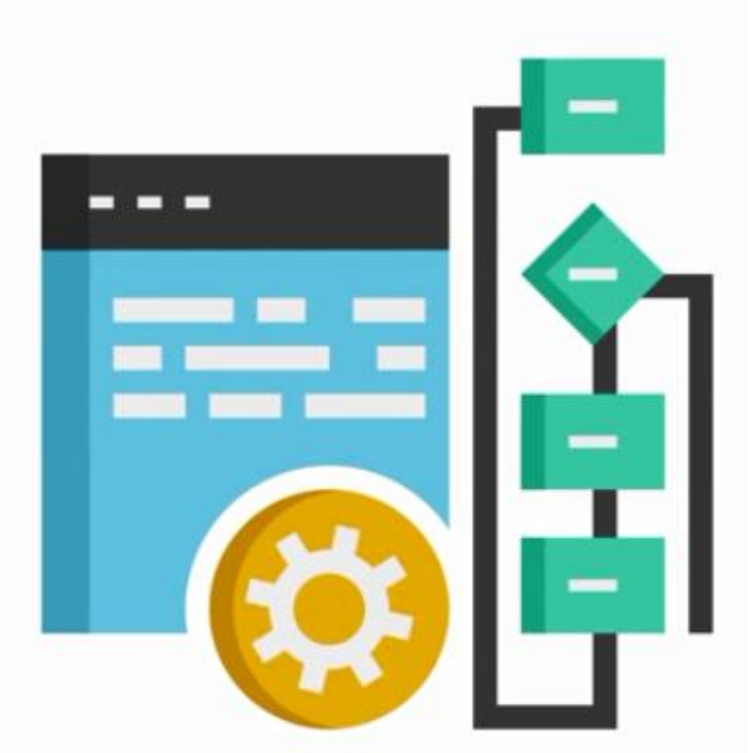
Multivariate

Combined value of at least two variables

While graphs like box plots, histograms and scatter-plots are used to detect outliers, detector algorithms recognize outliers in large data sets.

Python Outlier Detection (PyOD)

PyOD is a library that is scalable, useful for detecting outliers, and works with 20 different algorithms.



PyOD has many advantages:

It is open-source.

It supports advanced models.

Its performance is optimized.

It is compatible with Python 2 and 3.

Outlier Detection

Outlier Detection Algorithms in PyOD

Some outlier detection algorithms in PyOD are:

- Angle-based outlier detection (ABOD)
- K-nearest neighbors detector
- Isolation forest
- Histogram-based outlier detection
- Local correlation integral (LOCI)
- Feature bagging
- Clustering-based local outlier factor (LOF)

Angle-Based Outlier Detection (ABOD)

It considers the relationship between every data point and its nearest neighbor and performs well on multi-dimensional data.

There are two versions of ABOD:

Uses k-nearest neighbors to approximate

Fast ABOD

Original ABOD

Considers all training points with quick complexity

K-Nearest Neighbors Detector

For any data point, the outlier score is the distance to the kth nearest neighbor.

The three KNN detectors are:

Largest

It takes the distance of the kth neighbor as the outlier score.

Mean

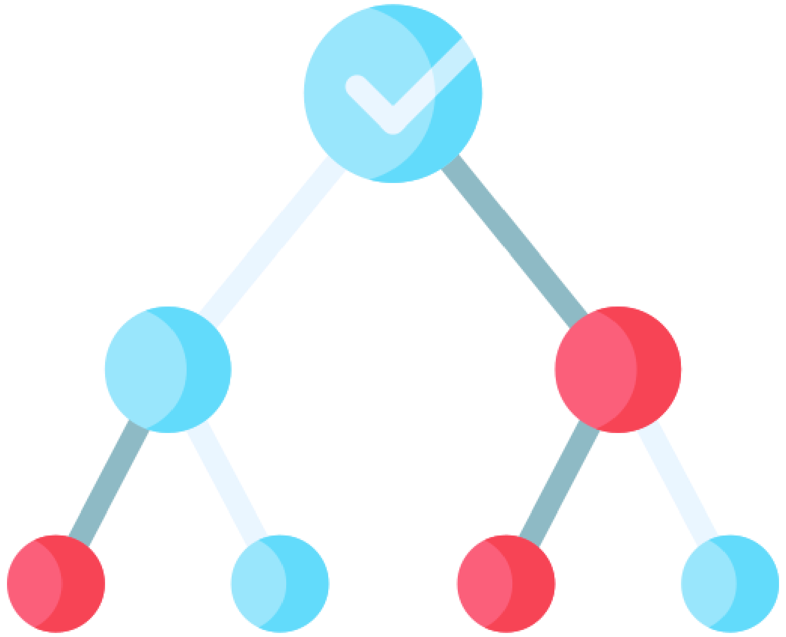
It takes the average of all K neighbors as the outlier score.

Median

It takes a median of the distance to k neighbors as the outlier score.

Isolation Forest

This uses the scikit-learn library.

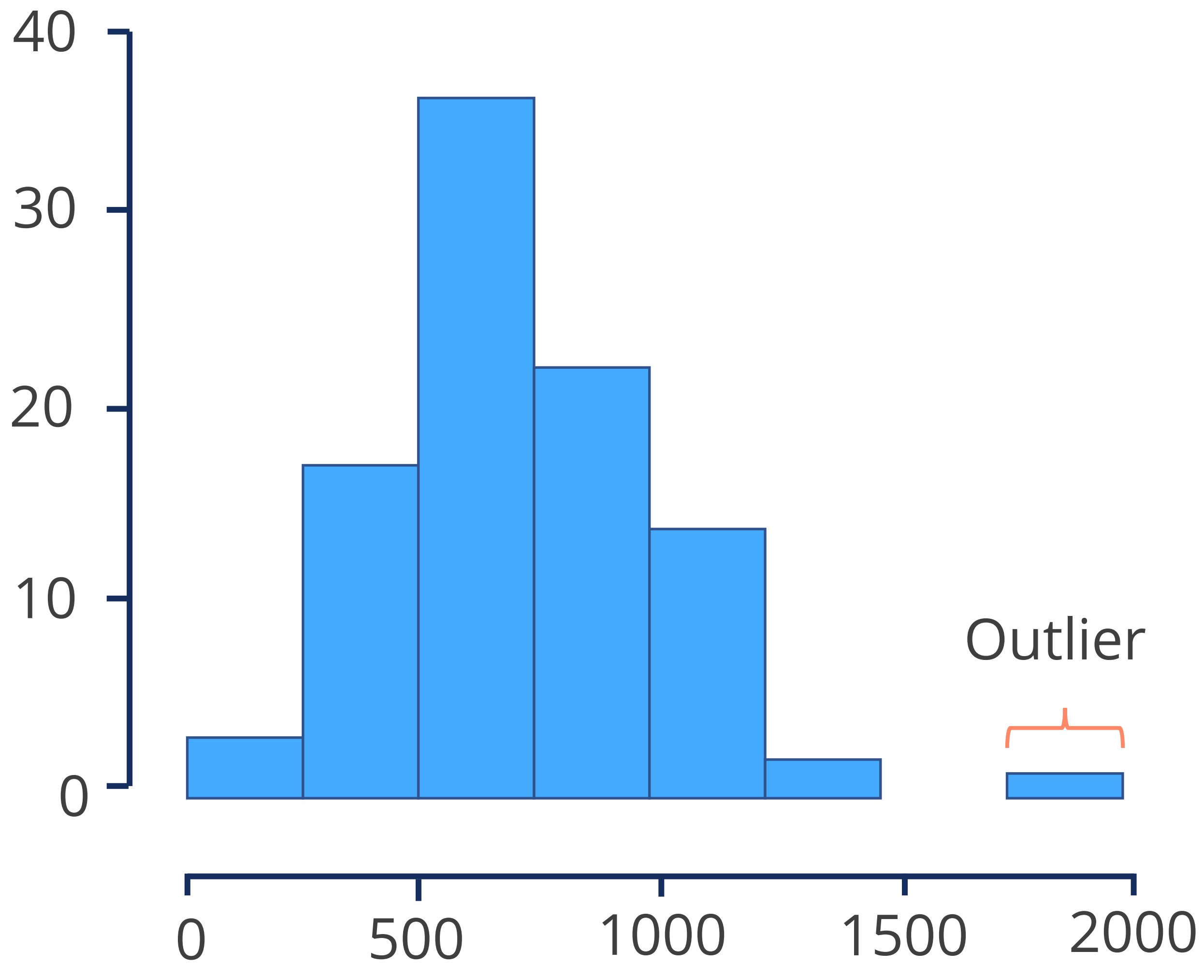


It uses the data partitioning method with a set of trees.

It gives an anomaly score, which indicates how isolated the points are to further identify outliers.

Histogram-Based Outlier Detection

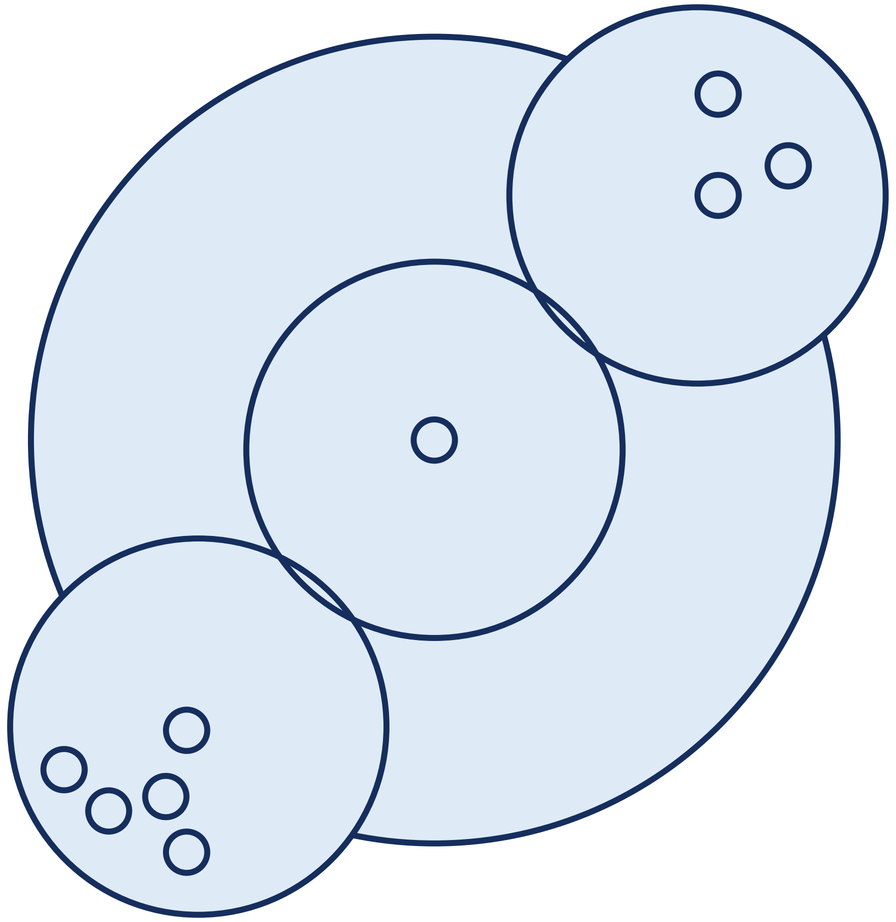
It is the most efficient unsupervised method to calculate outlier scores.



It is faster than multivariate approaches but less precise.

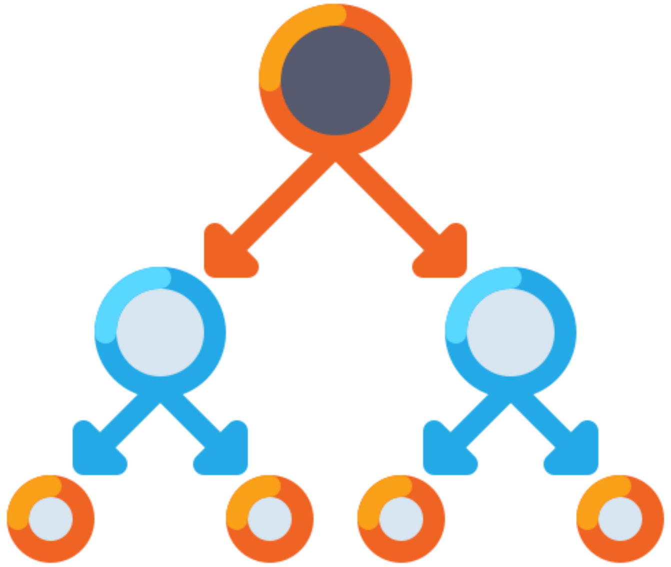
Local Correlation Integral (LOCI)

It is very effective to detect outliers and provides a LOCI plot for each point with an exhaustive summary information.



Feature Bagging

It fits a number of base detectors on various subsamples of the data set.



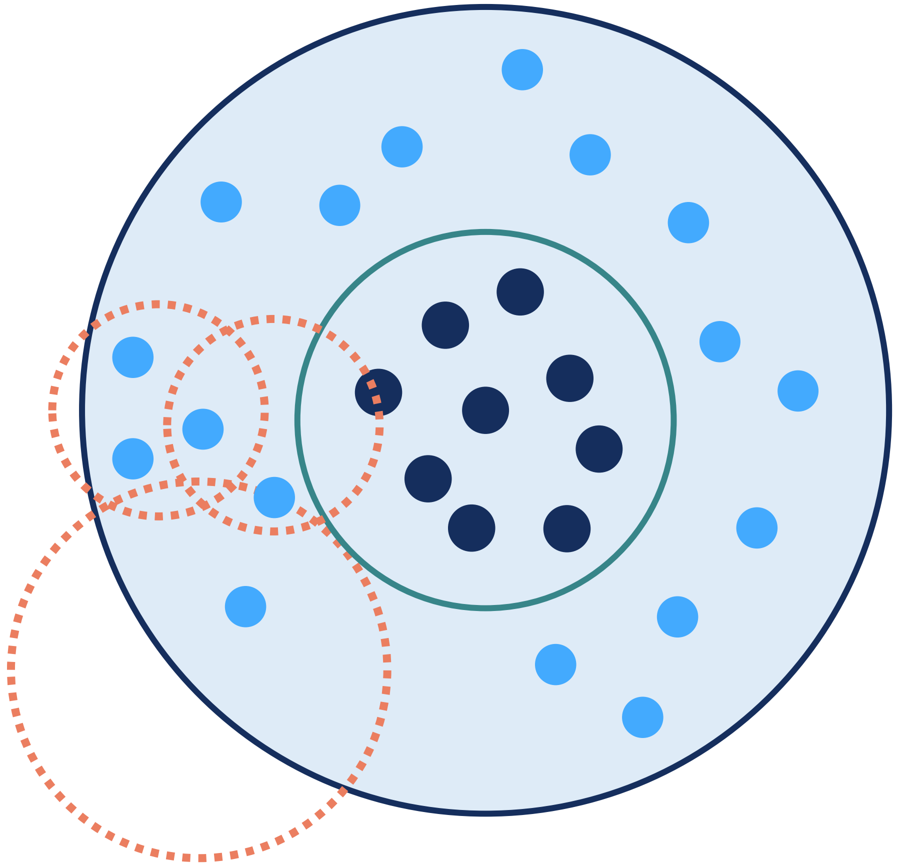
It uses aggregating or other combination methods for improved prediction accuracy.

It constructs n subsamples by randomly selecting the subset of features.

By default, LOF is used as the base estimator, though KNN or ABOD may also be used.

Clustering-Based Local Outlier Factor

It classifies data into small and large clusters.



It calculates the anomaly score based on the size of the cluster and the distance of the nearest largest cluster.

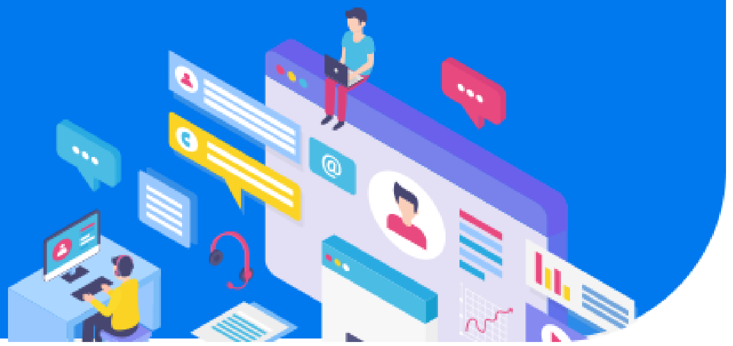
Outlier Detection

Outlier detection is widely used in:

Finance fraud detection: Unique or uncommon transactions that are dissimilar from the others make it easy to find frauds.

Candidate selection: PyOD finds a candidate application different from others to choose the best candidate.

Assisted Practices



Let’s understand the topic below using Jupyter Notebook.

- 6.16_Demo KNN for Anomaly Detection

Note: Please download the pdf files for each topic mentioned above from the Reference Material section.

Principal Component Analysis

Principal Component Analysis

Problems in the real-world typically deal with data sets that have a huge number of features.

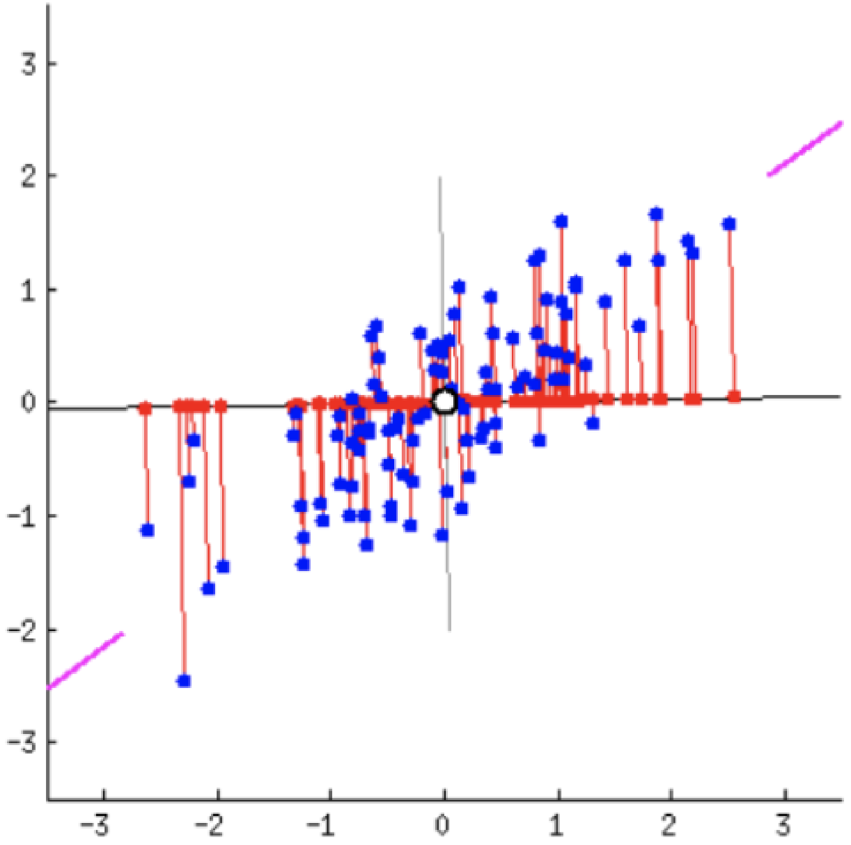
Example

High resolution images that need classification or power allocation exercises across multiple communication channels that have high dimensionality.

Dealing with such data sets demands increased computational power and more complex algorithms which may lead to overfitting.

Principal Component Analysis

The principal component analysis (PCA) is an unsupervised learning technique used to preprocess the data sets and reduce their dimensionality while preserving the original data set.



Machine learning models can still learn from the original data.

Common Terms in PCA

Dimensionality	It is the number of features present in the data.
Correlation	It tells how strongly the features are related to each other.
	The correlation value ranges between -1 and +1.
	The value is -1 if the variables are inversely proportional to each other and +1 if the variables are directly proportional to each other.

Common Terms in PCA

Orthogonal	It is used when the variables are not correlated to each other.
Eigenvector	If a square matrix (m) and a nonzero vector (v) are given, v is the eigenvector and Av is the scalar multiple of v.
Covariance matrix	It is a matrix that contains the covariance between the variables.

Steps of PCA

PCA is performed using the following five steps:

1. Standardization
2. Covariance and matrix computation
3. Identifying the principal components
4. Feature vector
5. Recasting data along the principal component axes

Standardization

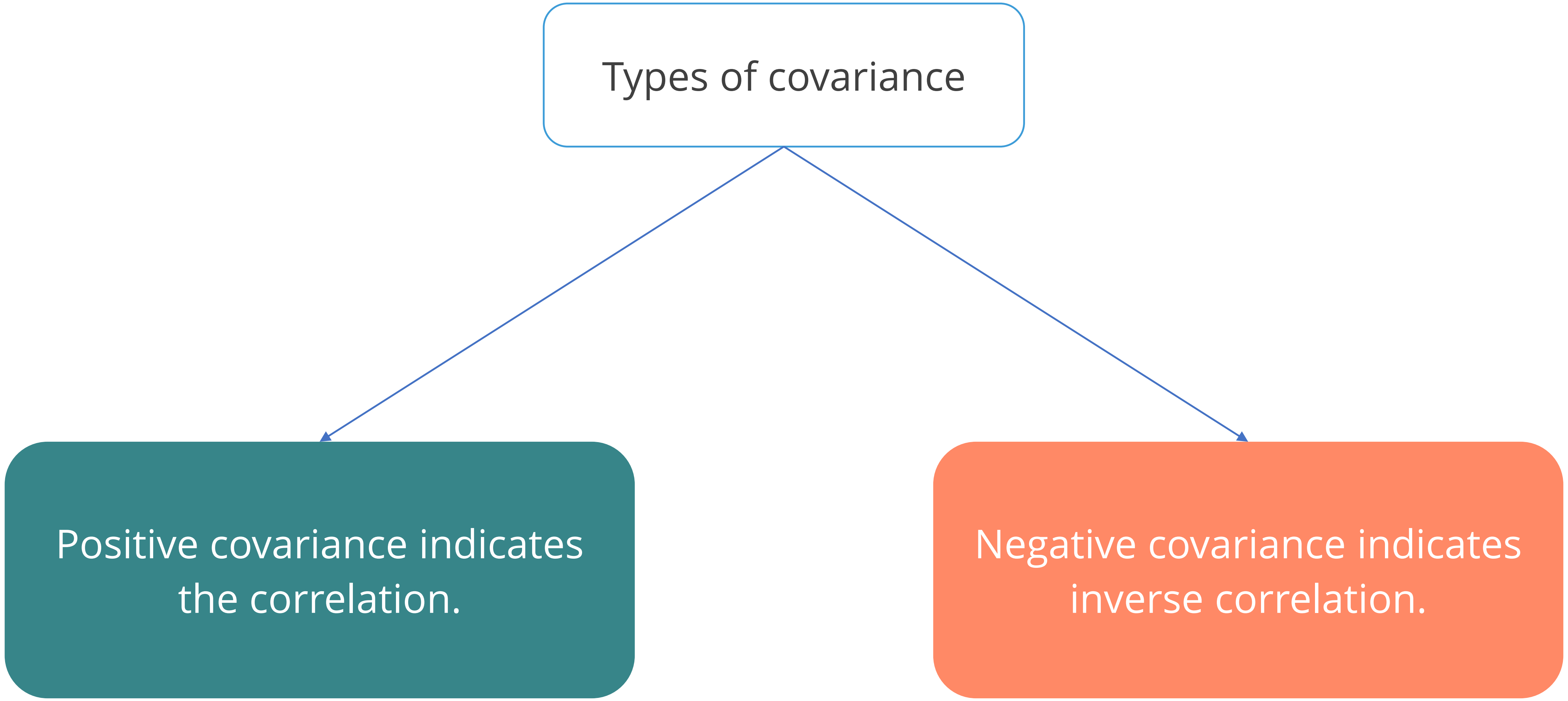
The range of variables is standardized so that the contribution of each is equal.

$$Z = \frac{\text{Value - mean}}{\text{Standard deviation}}$$

This helps to normalize the dominance that variables with larger ranges would normally exert over those with smaller values.

Covariance Matrix Computation

It helps check the correlation between features in a dataset.



Types of covariance

Positive covariance indicates the correlation.

Negative covariance indicates inverse correlation.

Covariance Matrix Computation

The covariance matrix is a tabular representation that provides a summary of the relationships (correlations) between variables.

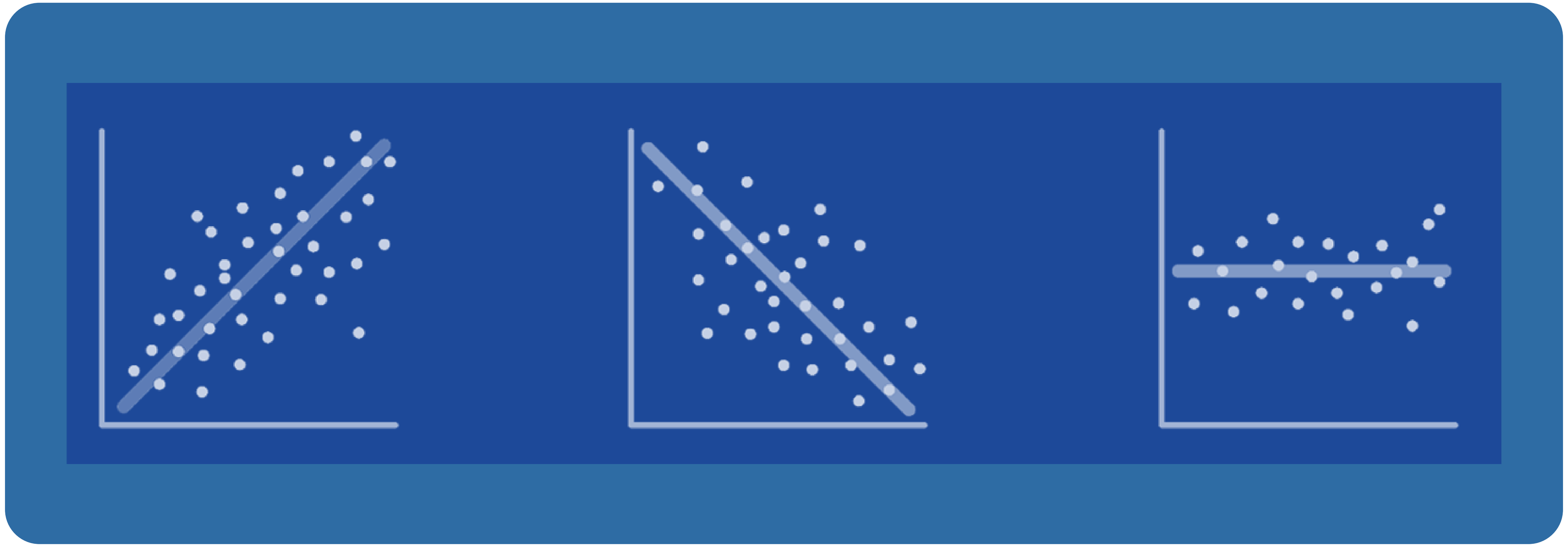
$$\begin{bmatrix} \mathrm{Cov}(x,x) & \mathrm{Cov}(x,y) \\ \mathrm{Cov}(y,x) & \mathrm{Cov}(y,y) \end{bmatrix}$$

Example

In a 2D dataset containing two variables, x and y, the covariance matrix is a 2 x 2 matrix.

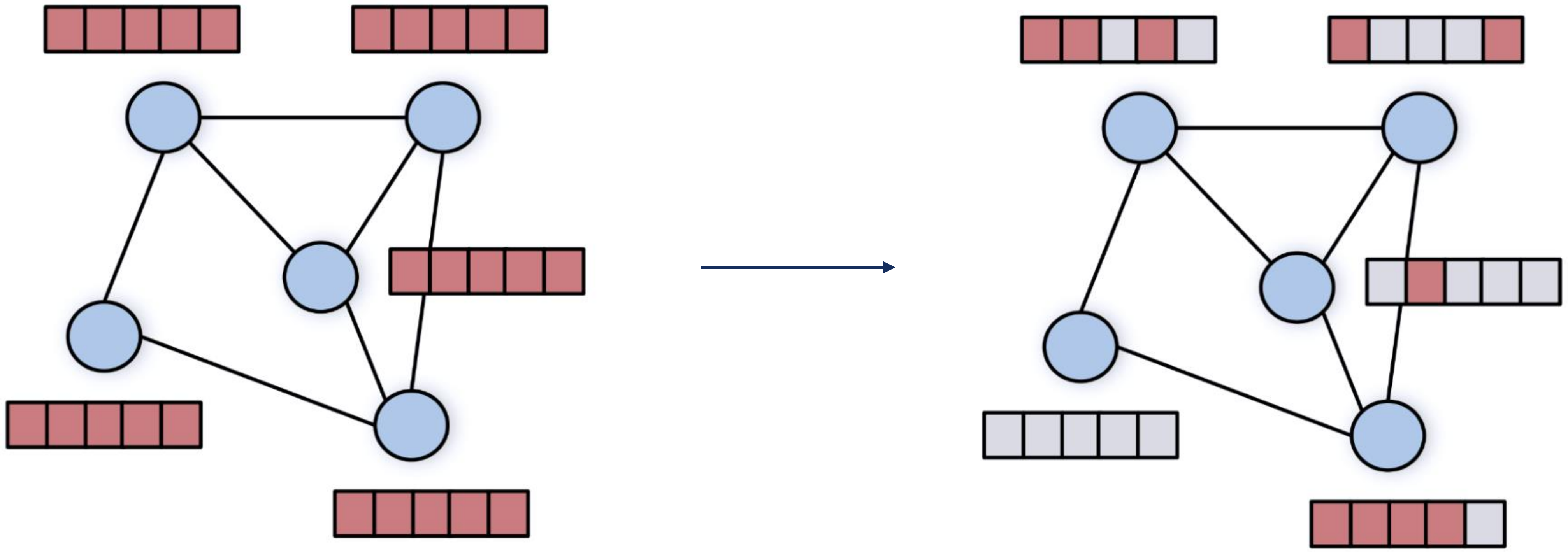
Identifying Principal Components

In this step, the new noncorrelated variables that are being constructed by squeezing or compressing features are identified.



Feature Vector

This is also the first dimensionality reduction step.



One can use this step to determine if the PCA features previously identified need to be retained or discarded.

Recasting Data Along Principal Component Axes

The data is oriented from its original axes to new axes represented by the principal components that have been identified.

$$\text{FinalDataSet} = \text{FeatureVector}^{T} * \text{OriginalDataSet}^{T}$$

This is done by multiplying the transpose of the feature vector by the transpose of the original data set.

Why Use PCA?

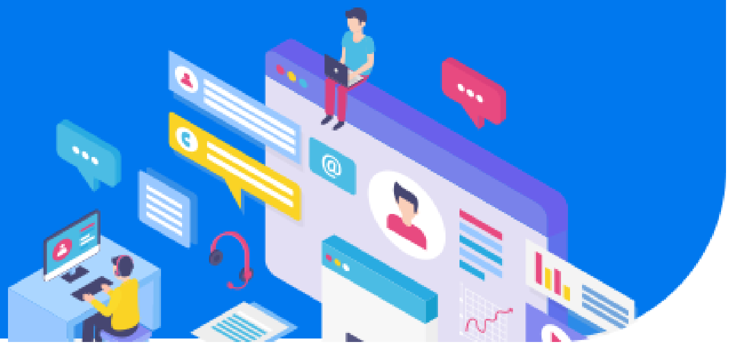
PCA is used to compress information into a smaller set with new dimensions.

It is used in neuroscience to identify the action potential of neurons by their shape.

It is used in quantitative finance to reduce the complexity of stock analysis.

It helps discover the most important features in a large data set.

Assisted Practices



Let’s understand the topic below using Jupyter Notebook.

- 6.18_Applying Principal Component Analysis (PCA)

Note: Please download the pdf files for each topic mentioned above from the Reference Material section.

Correspondance Analysis and Multiple Correspondance Analysis (MCA)

Correspondence Analysis

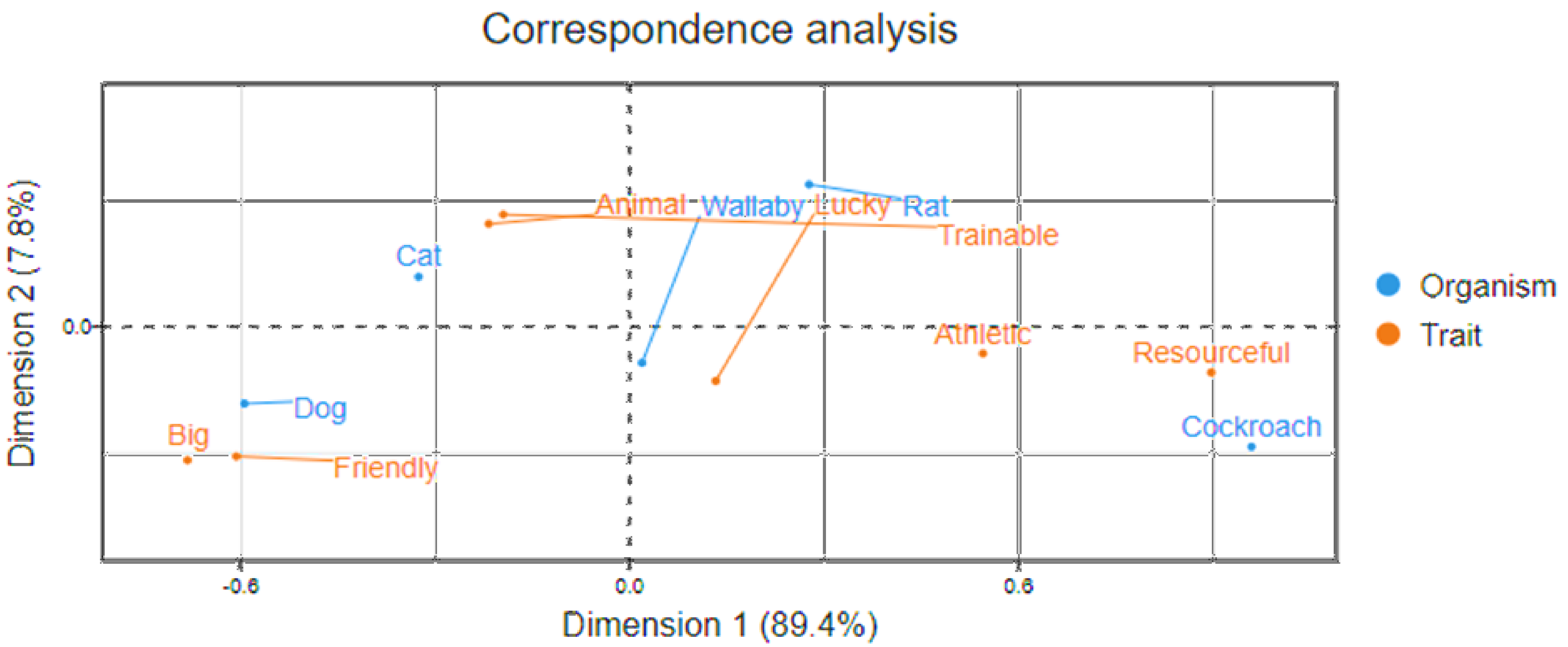
It is a multivariate statistical tool used to study the associations between two variables.

- It starts with a big table that is difficult to read and ends with a relatively simple visualization.
- It is based on the idea of representing the data as points in a low-dimensional space.
- The points are then used to visualize the relationships between the different categories of data.

The primary goal is to represent as much inertia as possible on the first principal axis, the maximum residual inertia on the second principal axis and so forth.

Correspondence Analysis

The following image shows correspondence analysis done for the traits of certain animals:



The animals are plotted in rows, and the traits are plotted in columns.

Correspondence Analysis

The correspondence analysis indicates that cat and dog are most similar, whereas dog and cockroach are least similar.

To compare a row label to a column label, observe the length of the line connecting the row label to the origin.

The longer the line, the stronger the association between the row label and the column label.

Interpreting the correspondence analysis charts helps one better understand the data in the tables.

Multiple Correspondence Analysis

Multiple correspondence analysis (MCA) is an extension of correspondence analysis that explores the relationships among multiple variables.

It helps one conduct correspondence analysis if there is a table with:

At least two rows and two columns

No missing data

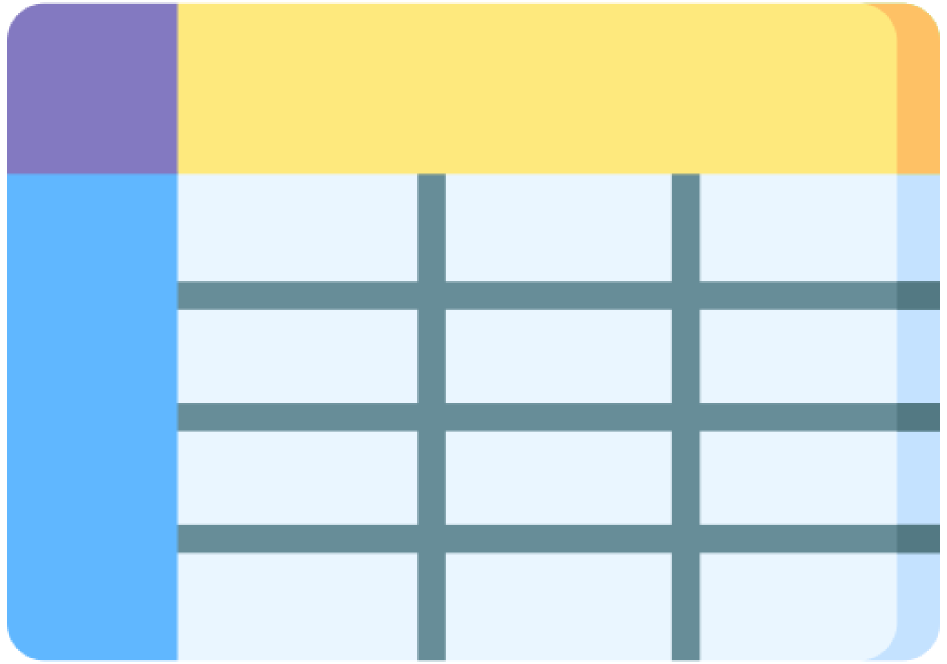
No negative values

All the data with the same scale

It’s usually used to analyze contingency tables through row and column profiles.

Contingency Tables

It is one in which the row and column categories are mutually exclusive.



It is useful in the study of the correlation between two or more categorical variables.

Contingency Tables

The following contingency table shows people's favorite games and snacks during family game night.

	Pizza rolls	Chips and dip	Cookies	Total
Poker	10	3	12	**25**
Trivial pursuit	8	14	7	**29**
Monopoly	14	17	7	**38**
Wii bowling	12	7	4	**23**
Total	**44**	**41**	**30**	**115**

There are 12 people who picked Poker as their favorite game and cookies as their favorite snack.

Correspondence analysis has found applications in several fields, ranging from epidemiology to market research to social sciences.

Singular Value Decomposition

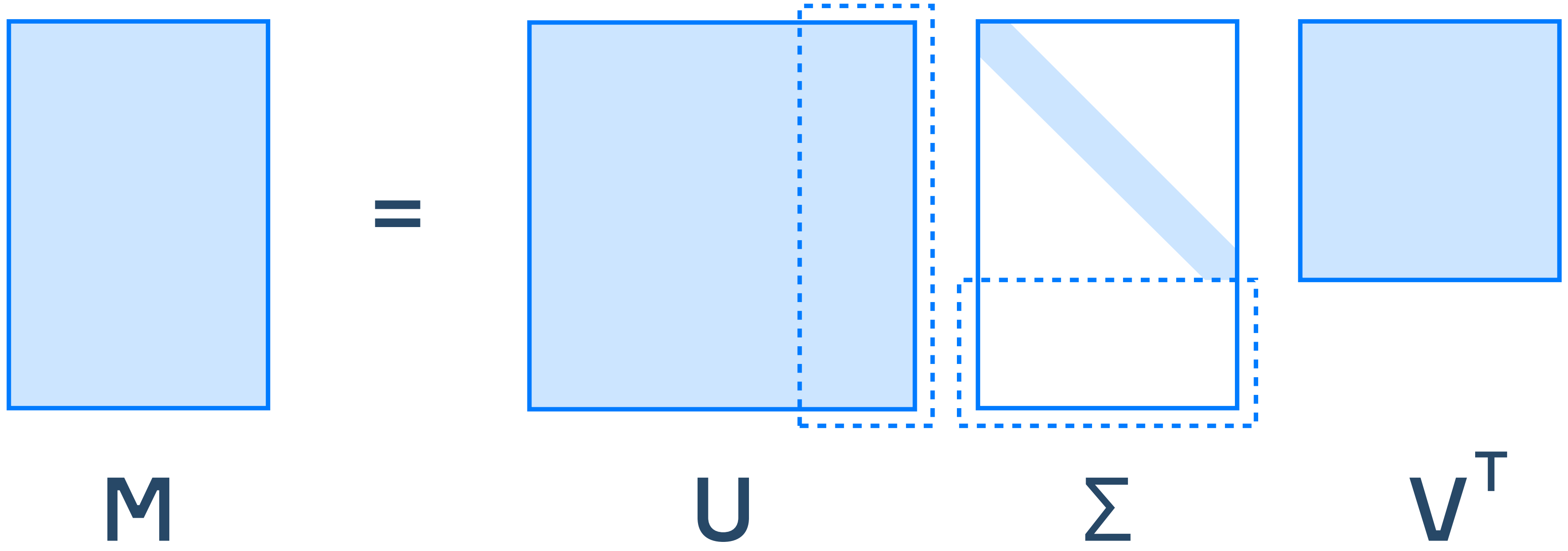
Singular Value Decomposition

Singular value decomposition (SVD), a powerful tool in linear algebra, is a data reduction method used in machine learning.

It enables the extraction and untangling of information from high-dimensional raw data.

Singular Value Decomposition

It is a matrix decomposition technique that decomposes a matrix into three generic and familiar matrices.



It makes certain subsequent matrix calculations simpler with matrices that are easy to manipulate and analyze.

Singular Value Decomposition

It states that any matrix A can be factorized into two unitary matrices U and V that are:

Orthogonal in nature

Rectangle diagonal of singular value sigma

Singular Value Decomposition

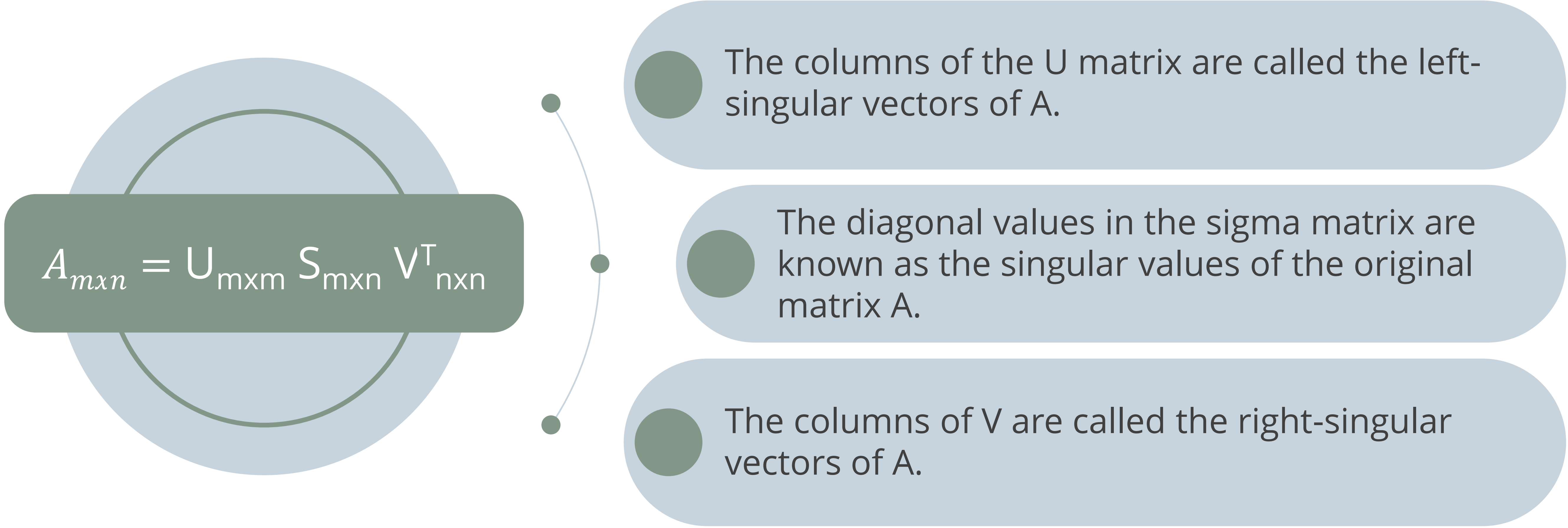
The formula is as follows:

$$A = U \ \Sigma V^T$$

Where:

- A: It is the real m x n matrix that we wish to decompose.
- U: It is an m x m matrix.
- Σ : It is an m x n diagonal matrix.
- V^T: It is the transpose of an n x n matrix.

Singular Value Decomposition



$A_{mxn} = \mathrm{U}_{\mathrm{mxm}}\ \mathrm{S}_{\mathrm{mxn}}\ \mathrm{V}^{\mathrm{T}}{}_{\mathrm{nxn}}$

- The columns of the U matrix are called the left-singular vectors of A.
- The diagonal values in the sigma matrix are known as the singular values of the original matrix A.
- The columns of V are called the right-singular vectors of A.

SVD breaks a matrix down into singular vectors.

Singular Value Decomposition

It has uses in machine learning and is the foundation of the recommendation algorithms that power well-known brands like:

Amazon

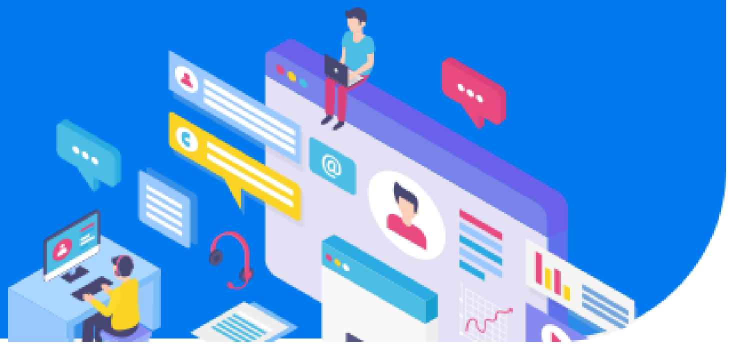
YouTube

Google

Facebook

It can also be used for image compression, which facilitates memory-efficient image storage.

Assisted Practices



Let’s understand the topic below using Jupyter Notebook.

- 6.21_Applying Singular Value Decomposition

Note: Please download the pdf files for each topic mentioned above from the Reference Material section.

Independent Component Analysis

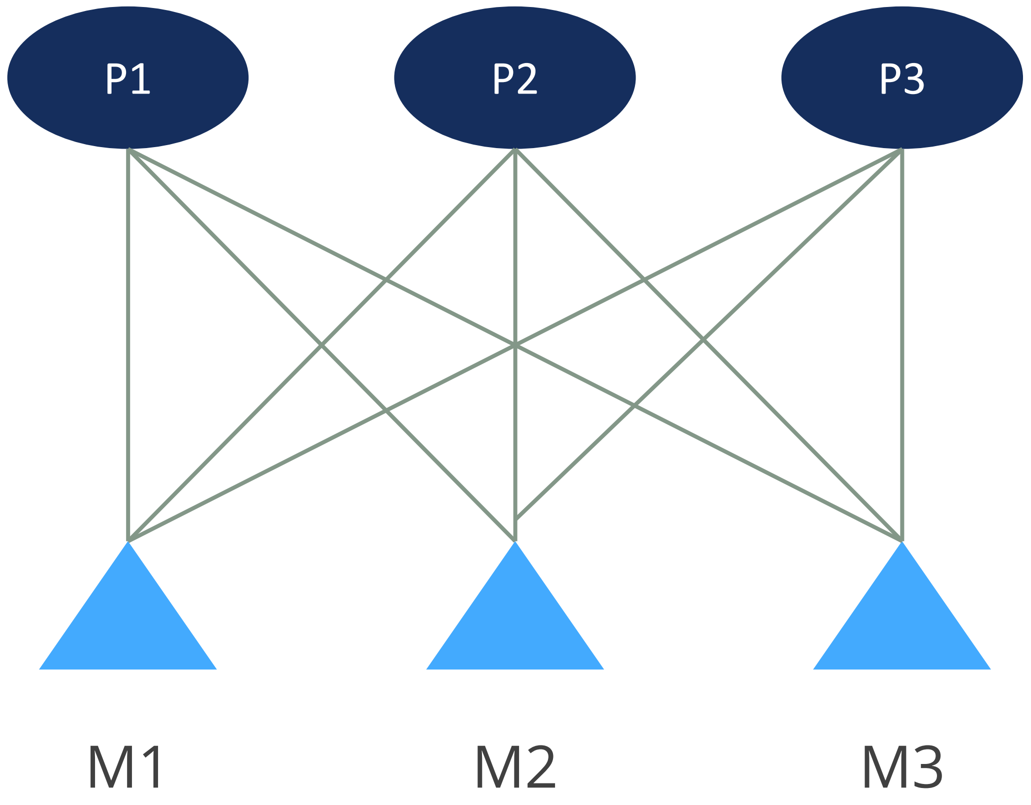
Independent Component Analysis

Independent component analysis (ICA) is a machine learning technique used to distinguish independent sources from a mixed signal.

While principal component analysis focuses on maximizing the variance of the data points, independent component analysis focuses on the independence of components.

Independent Component Analysis

Example: microphone placement on the stage of a reality show



P1, P2 and P3 are three people present in the show.

M1, M2 and M3 are the microphones installed at different distances from each of them, recording the voice signals passed by them.

Number of speakers = Number of microphones

The goal is to separate the three speakers’ voices from the microphone recordings.

Independent Component Analysis

It helps to separate the mixed signals of each microphone recording into independent speech signals.

[P1, P2, P3] => [X1, X2,Y3]

Where:

- P1, P2 and P3 are the original signals present that are mixed signals.
- X1, X2 and X3 are the new features that are independent of each other.

Features of ICA

The features of ICA are:

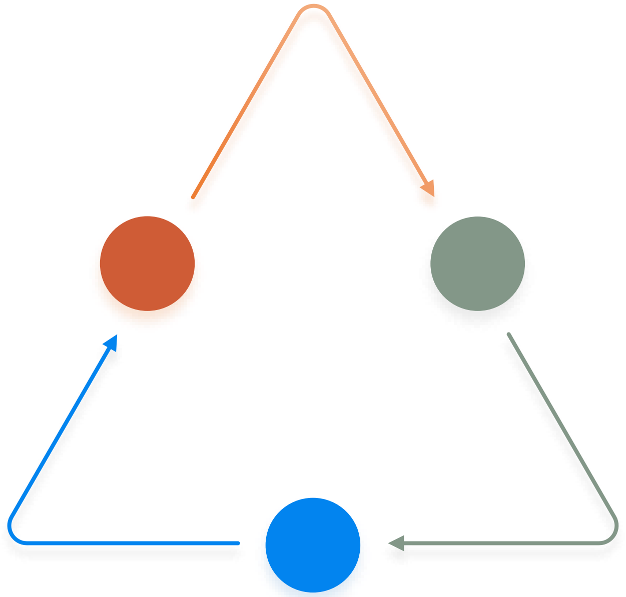
It segregates the mixed signal into its independent source signal.

It deals with the independent components.

It gives priority to the mutual independence of the components.

Restrictions of ICA

ICA has the following restrictions:



Each independent component created by the ICA is statistically independent of others.

The components generated by the ICA have a non-Gaussian distribution.

The independent components generated by the ICA are equal to the number of observed mixtures.

Assisted Practices

Let’s understand the topic below using Jupyter Notebook.

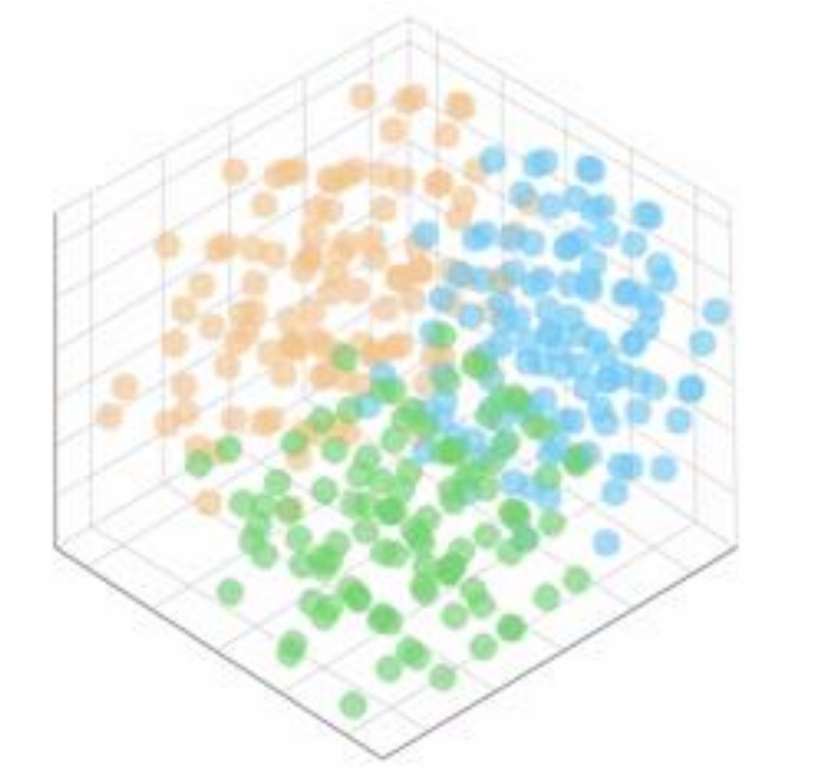
- 6.23_Applying Independent Component Analysis

Note: Please download the pdf files for each topic mentioned above from the Reference Material section.

Balanced Iterative Reducing and Clustering Using Hierarchies (BIRCH)

BIRCH

It is an unsupervised clustering algorithm that was developed to process massive datasets.

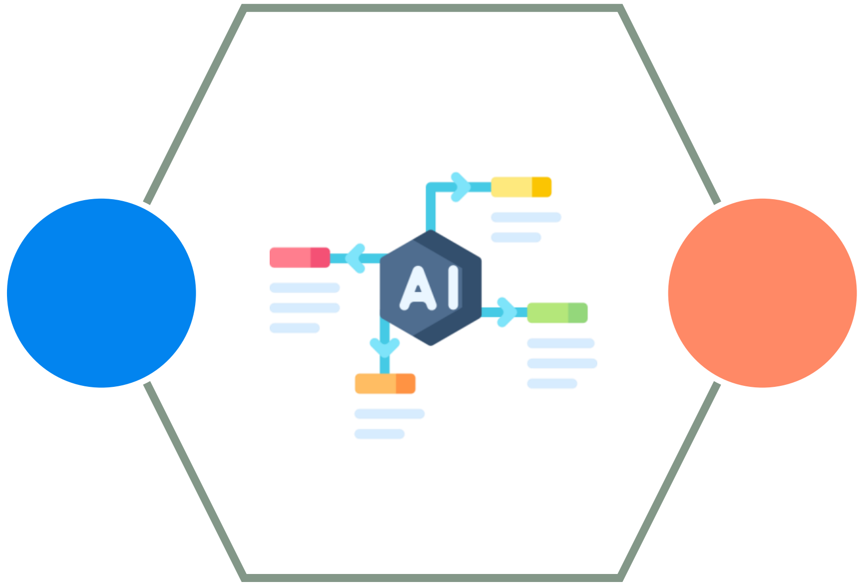


It clusters large data sets by first making a small and compact summary of the large data set.

It then clusters this compact summary instead of the large data set.

BIRCH

The algorithm is based on:

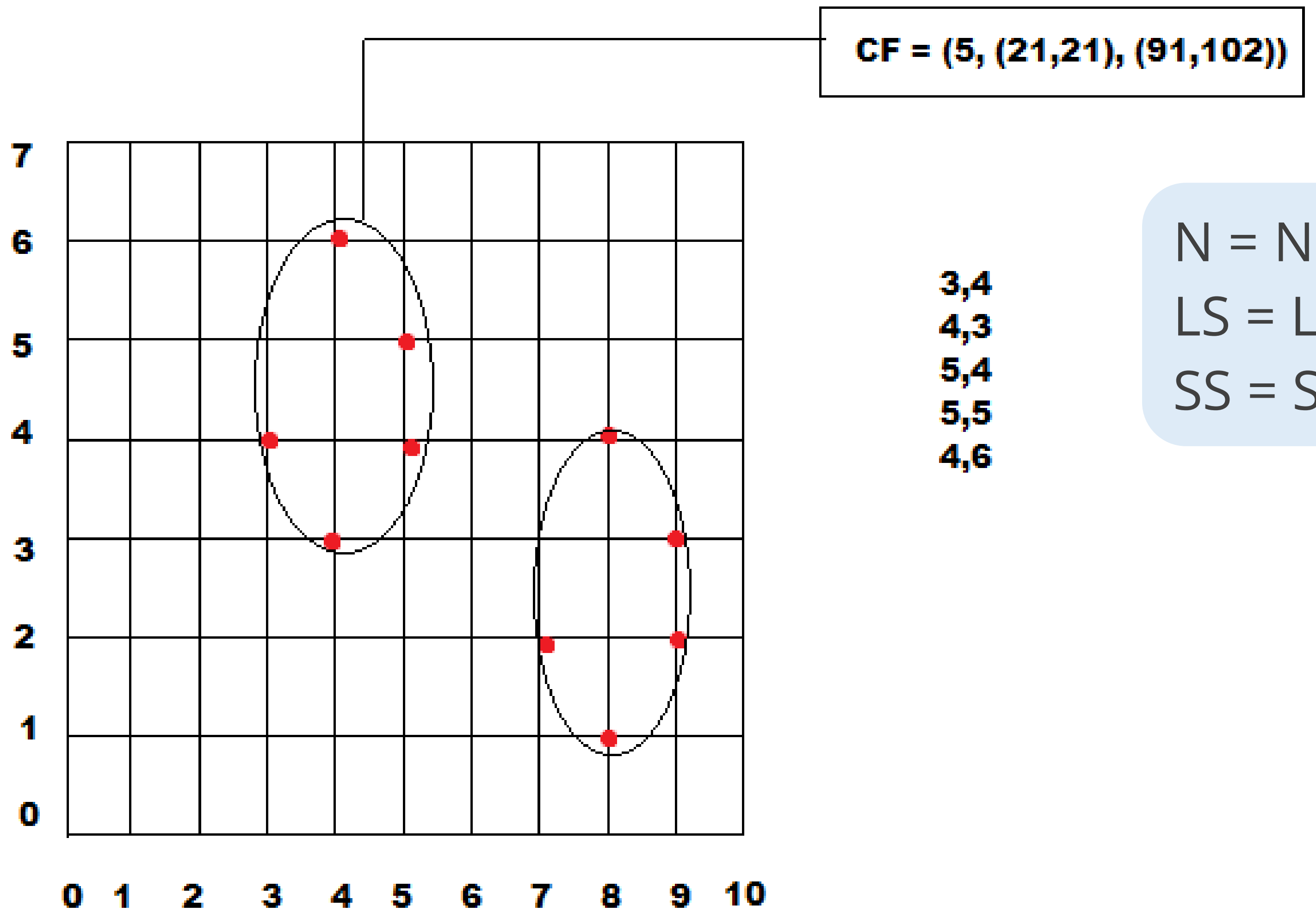


Clustering Feature

Clustering Feature (CF) Tree

Clustering Feature

The clustering feature entry is a triplet of N, LS and SS.



CF = (5, (21,21), (91,102))

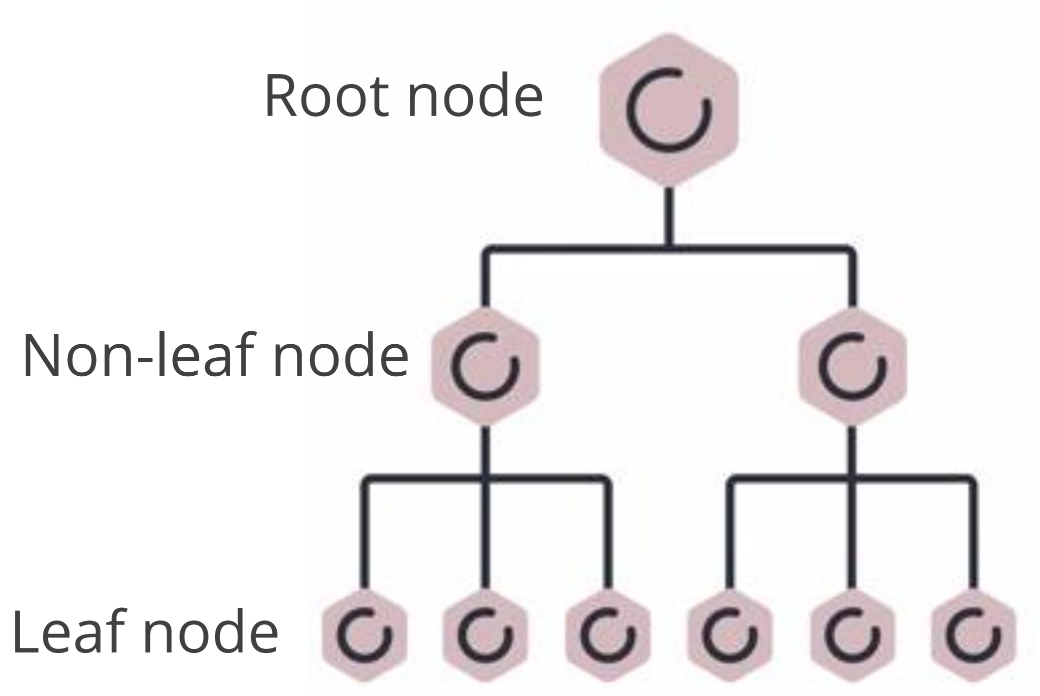
3,4
4,3
5,4
5,5
4,6

N = Number of data points in a cluster
LS = Linear sum of the data points
SS = Squared sum of the data points in the cluster

In this example:
CF = (N, LS, SS) = (5, (21,21), (91,102))

Clustering Feature Tree

A CF tree is a tree with leaf and non-leaf nodes where each leaf node contains a subcluster.



Each entry of a CF tree has a pointer to the child node.

The CF tree entry is made by the sum of CF entries in the child node.

There is a limit to the number of entries for each leaf node called threshold.

BIRCH Algorithm

The BIRCH algorithm has four phases:

Scanning the data

Data is loaded in the model, post-which the algorithm scans all the data and fits it into CF trees.

Condensing data

The size of the data is adjusted by the algorithm for better fitting into the CF tree.

Global clustering

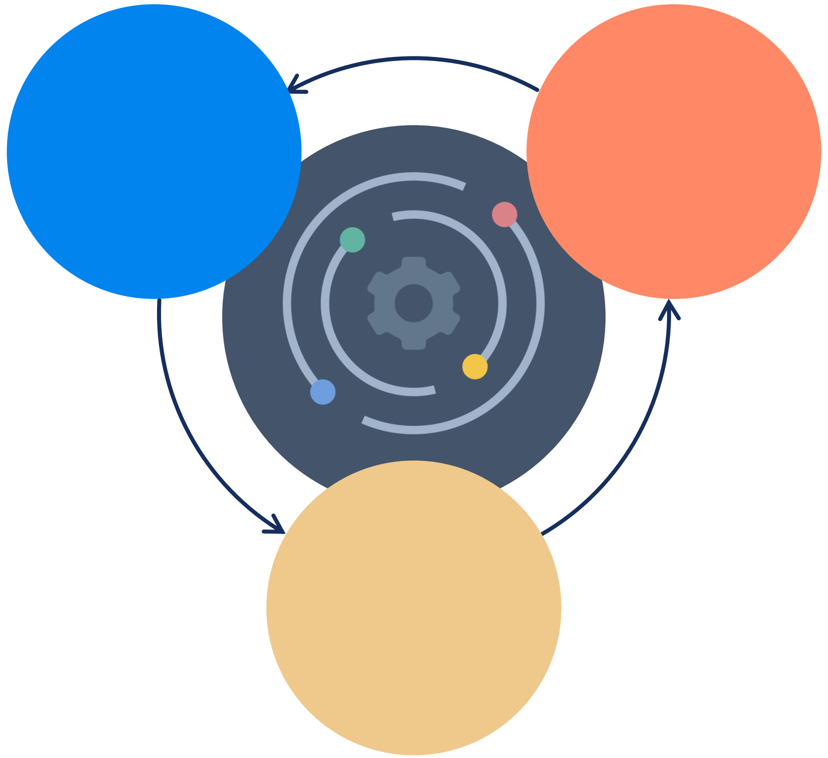
With the help of an existing algorithm, clustering is performed on CF trees.

Refining clustering

All incorrect assignments of observations are fixed.

BIRCH Algorithm

The BIRCH algorithm has three parameters.



Threshold

Indicates the maximum number of observations that a subcluster can have

Branching factor

Represents the number of CF subclusters in a node

N clusters

Indicates the number of clusters

Assisted Practices

Let’s understand the topic below using Jupyter Notebook.

- 6.25_Applying BIRCH

Note: Please download the pdf files for each topic mentioned above from the Reference Material section.

Key Takeaways

- Unsupervised learning extracts previously unknown patterns from a data set.
- There are two main types of unsupervised machine learning algorithms: clustering and associative algorithms.
- Clustering is an unsupervised technique that involves grouping or clustering of data points.
- The association technique involves finding the relationships between variables in a large data set.

Key Takeaways

- PyOD is a library that is very useful for detecting outliers and works with 20 different algorithms.
- Principal component analysis is an unsupervised learning technique used to preprocess the data sets and reduce their dimensionality while preserving the original data set.
- Independent component analysis is a machine learning technique used to distinguish independent sources from a mixed signal.
- BIRCH clusters large data sets by making a small and compact summary of the large data set and then clustering it instead of the large data set.

Knowledge Check

Knowledge Check

1

What is the goal of hierarchical clustering?

A. To classify data into distinct groups

B. To identify anomalies in data sets

C. To reduce the dimensionality of input data

D. To create a tree-shaped structure known as a dendrogram

Knowledge Check

1

What is the goal of hierarchical clustering?

A. To classify data into distinct groups

B. To identify anomalies in data sets

C. To reduce the dimensionality of input data

D. To create a tree-shaped structure known as a dendrogram

The correct answer is **D**

The goal of hierarchical clustering is to create a tree-shaped structure known as a dendrogram that shows the hierarchical connection of items and finds the best approach to assign items to a cluster based on similarities and dissimilarities.

Knowledge Check

2

What is the first step to choose the optimal number of clusters in hierarchical clustering?

A. Choosing a random number

B. Identifying the longest line that traverses the maximum vertical distance without intercepting any of the merging points in the dendrogram

C. Counting the number of horizontal lines in the dendrogram

D. Looking at the colors in the dendrogram

Knowledge Check

2

What is the first step to choose the optimal number of clusters in hierarchical clustering?

A. Choosing a random number

B. Identifying the longest line that traverses the maximum vertical distance without intercepting any of the merging points in the dendrogram

C. Counting the number of horizontal lines in the dendrogram

D. Looking at the colors in the dendrogram

The correct answer is **B**

The first step to choose the optimal number of clusters in hierarchical clustering is by identifying the longest line that traverses the maximum vertical distance without intercepting any of the merging points in the dendrogram.

Knowledge Check

3

How does ICA differ from PCA?

A. ICA and PCA are the same techniques with different names.

B. ICA is used for supervised learning, while PCA is used for unsupervised learning.

C. ICA focuses on maximizing the independence of components, while PCA focuses on maximizing the variance of data points.

D. ICA and PCA are both used for data reduction in machine learning.

Knowledge Check

3

How does ICA differ from PCA?

A. ICA and PCA are the same techniques with different names.

B. ICA is used for supervised learning, while PCA is used for unsupervised learning.

C. ICA focuses on maximizing the independence of components, while PCA focuses on maximizing the variance of data points.

D. ICA and PCA are both used for data reduction in machine learning.

The correct answer is **C**

ICA focuses on maximizing the independence of components, while PCA focuses on maximizing the variance of data points.

Thank You!