M.Tech Program

Advanced Industry Integrated Programs

Jointly offered by University and LTIMindTree

Python for Data Science

Knowledge partner

LTIMindtree

Implementation partner

L&T EduTech

Modules to cover

1. Python - Data Structures, OOPS & Modules
2. Python - Numpy, Pandas & DS Libraries
3. Visualization
4. Regression and Classification
5. **Unsupervised and Advanced Machine Learning**

Unsupervised and Advanced Machine Learning

Unsupervised and Advanced Machine Learning

Introduction to KNN Algorithm

- K-Nearest Neighbors (KNN) is a non-parametric, lazy learning algorithm
- KNN is used for both classification and regression tasks.
- KNN is based on Similarity-based learning

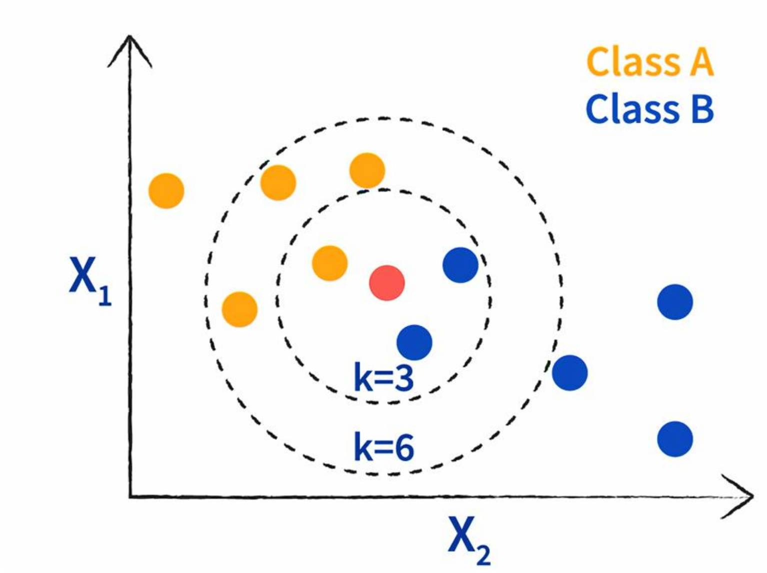

Fig 5.1: Visual Representation of K-Nearest Neighbors (KNN) Classification

Unsupervised and Advanced Machine Learning

Introduction to KNN Algorithm

Common Distance Measures

- **Euclidean** - distance between two vectors
- **Hamming** - difference between two binary vectors
- **Manhattan** - difference between vectors or real values
- **Minkowski** - generalization of Euclidean and Manhattan

Unsupervised and Advanced Machine Learning

KNN Algorithm - Classification

How KNN Algorithm for Classification Works

1. Select the Number of Neighbors (K)
2. Calculate the Distance
3. Sort the Distances
4. Select the K Nearest Neighbors
5. Assign the Class

Unsupervised and Advanced Machine Learning

KNN Coding with Python

1. Load the dataset
2. Data preprocessing and feature engineering
3. Choose the value of k

```
k = 5
```

4. Initialize the model

```
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors=k)
```

Unsupervised and Advanced Machine Learning

KNN Coding with Python

5. Fit the model

   knn.fit(X_train, y_train)

6. Predict cluster labels

   y_pred = knn.predict(X_test)

7. Visualize the results

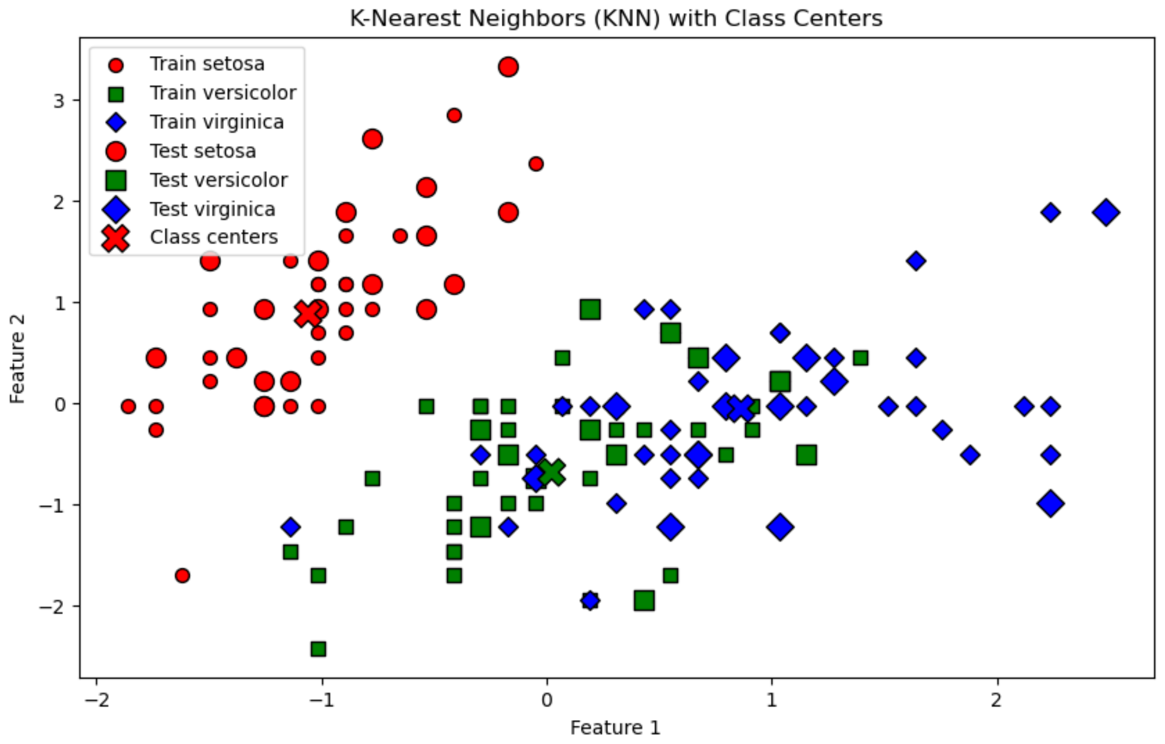

Fig 5.2: K-Nearest Neighbors (KNN) with Class Centers

Unsupervised and Advanced Machine Learning

KNN Classification Project Overview

Problem Statement:

The given Titanic dataset contains information about the passengers who were on board the Titanic. Build a predictive model using K-Nearest Neighbors to predict the survival of passengers on the Titanic based on a set of features.

Objective:

Develop a K-Nearest Neighbors model to predict the survival status of passengers in the test set using the given features. The goal is to classify each passenger in the test set as either 0 (did not survive) or 1 (survived).

Unsupervised and Advanced Machine Learning

KNN Classification Project Overview

Steps:

1. **Data Collection and Exploration**

   Load the dataset and identify the key features and target variable.

2. **Data Preprocessing**

   Clean and prepare the data for modeling.

   - Handle missing values
   - Encode categorical variables
   - Normalize numerical features

Unsupervised and Advanced Machine Learning

KNN Classification Project Overview

Steps:

3. **Feature Engineering**

   Create additional features to improve the predictive power of the model.

4. **Splitting Data**

   Split the data into training and testing sets

Unsupervised and Advanced Machine Learning

KNN Classification Project Overview

Steps:

5. **Implementing KNN Model**

Develop and train the KNN model to predict survival of the passenger.

```
from sklearn.neighbors import KNeighborsRegressor
knn = KNeighborsClassifier()
knn.fit(x_train,y_train)
```

6. **Model Evaluation**

Assess the performance of the KNN model using metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared (R^2).

Unsupervised and Advanced Machine Learning

Introduction to Support Vector Machines (SVM)

- SVM is a supervised learning algorithm used for classification and regression.
- Finds the optimal hyperplane that best separates different classes in the feature space.

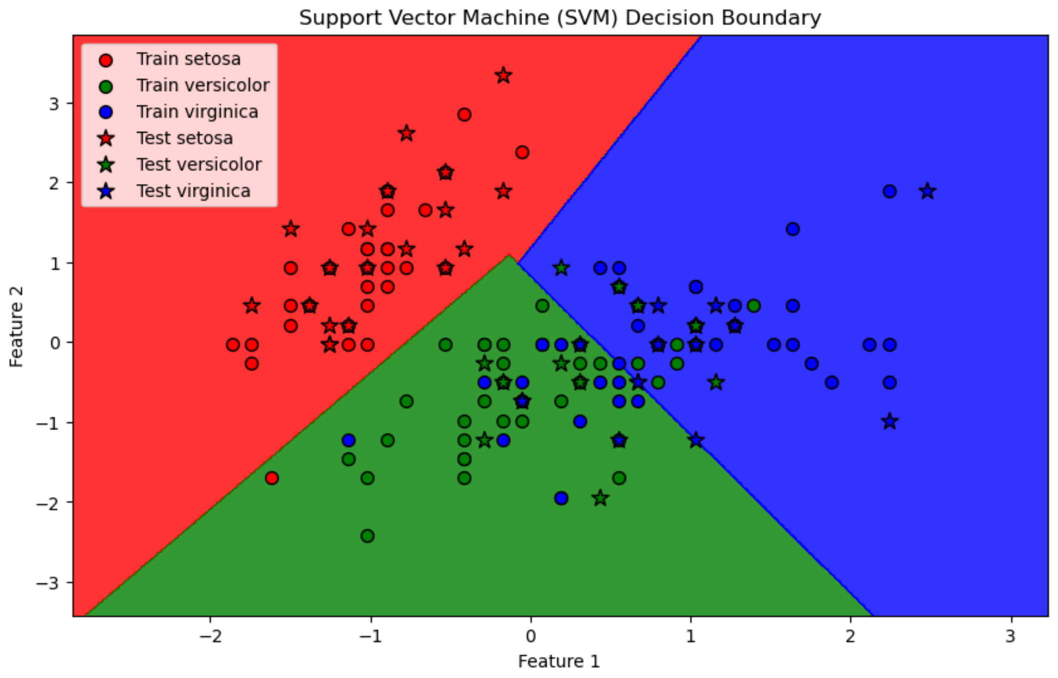

Fig 5.3: Support Vector Machine (SVM) Decision Boundary

Unsupervised and Advanced Machine Learning

History of Support Vector Machines

- **Origin**: Introduced by Vladimir Vapnik and Alexey Chervonenkis in 1963.
- **Development**: Gained popularity in the 1990s with the introduction of the kernel trick.
- **Evolution**: Extended to handle non-linear classification with kernel functions.
- **Milestones**: Key contributions and research that advanced SVM.
- **Impact**: Significant influence on machine learning and pattern recognition research.

Unsupervised and Advanced Machine Learning

SVM - Theory and Intuition - Hyperplanes and Margins

- **Hyperplanes**: A decision boundary that separates different classes.
- **Margins**: The distance between the hyperplane and the nearest data points from each class.
- **Support Vectors**: Data points that lie closest to the hyperplane and influence its position.
- **Maximizing Margin**: SVM aims to maximize the margin to achieve better generalization.
- **Mathematical Formulation**: The optimization problem underlying SVM.

Unsupervised and Advanced Machine Learning

Kernel Intuition

- **Purpose**: Allows SVM to perform in a higher-dimensional space without explicitly mapping data.
- **Types**: Common kernels include linear, polynomial, radial basis function (RBF), and sigmoid.
- **Non-linearity**: Enables SVM to handle non-linearly separable data.
- **Inner Product**: Kernels compute inner products in higher-dimensional spaces efficiently.

Unsupervised and Advanced Machine Learning

Kernel Trick and Mathematics

- **Kernel Trick**: Technique to transform data into higher dimensions to make it linearly separable
- **Mathematical Foundation**: Utilizing kernel functions to compute the dot product in transformed space.
- **Common Kernels**: Formulas and usage scenarios for linear, polynomial, RBF, and sigmoid kernels.
- **Implementation**: Implementing kernels within the SVM algorithm involves integrating kernel functions into the training and prediction processes
- **Advantages**: Reduces computational cost and complexity of working in high-dimensional spaces.

Unsupervised and Advanced Machine Learning

Kernel Trick and Mathematics

Knowledge Check

What is a characteristic feature of the "Kernel Trick" in machine learning?

A. It simplifies the computation of distance metrics between data points.

B. It allows for the nonlinear transformation of data into a higher-dimensional space.

C. It improves the convergence rate of iterative optimization algorithms.

D. It enhances the interpretability of feature importance in models.

Unsupervised and Advanced Machine Learning

Kernel Trick and Mathematics

Knowledge Check

What is a characteristic feature of the "Kernel Trick" in machine learning?

A. It simplifies the computation of distance metrics between data points.

B. It allows for the nonlinear transformation of data into a higher-dimensional space.

C. It improves the convergence rate of iterative optimization algorithms.

D. It enhances the interpretability of feature importance in models.

Unsupervised and Advanced Machine Learning

SVM with Scikit-Learn and Python - Classification

- **Libraries Required**: numpy, pandas, scikit-learn, and matplotlib.
- **Data Preparation**: Import dataset, handle missing values, and normalize features.
- **Model Implementation**: Use SVC (Support Vector Classifier) from scikit-learn for classification tasks.
- **Model Training**: Fit the SVM model using training data.
- **Model Evaluation**: Use accuracy, confusion matrix, and cross-validation for performance evaluation.

Unsupervised and Advanced Machine Learning

Regression Tasks

- **Definition**: Regression tasks involve predicting a continuous output variable.
- **Types**: Linear regression, polynomial regression, and support vector regression (SVR).
- **Applications**: Predicting prices, forecasting, and risk assessment.
- **Evaluation Metrics**: Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared.
- **Tools**: scikit-learn for implementing various regression algorithms in Python.

Unsupervised and Advanced Machine Learning

Introduction to Tree Based Methods

- Tree-based methods involve decision trees and ensemble techniques like random forests and gradient boosting.
- Used for both classification and regression tasks.
- Advantages: easy to interpret, handle non-linear relationships, and work well with mixed data types.
- Disadvantages: prone to overfitting if not properly pruned.
- Applications: used in finance, healthcare, and marketing for decision-making processes.

Unsupervised and Advanced Machine Learning

Decision Tree - History

- Decision Tree was introduced in the 1960s, gained prominence with the development of algorithms like ID3 (Iterative Dichotomiser 3) and CART (Classification and Regression Trees) in the 1980s.
- Enhanced with techniques like pruning, boosting, and random forests.
- Key developments and algorithms shaped decision tree learning.
- Widely used due to their interpretability and effectiveness.
- Ongoing research in optimizing decision trees and integrating them with other models.

Unsupervised and Advanced Machine Learning

Decision Tree

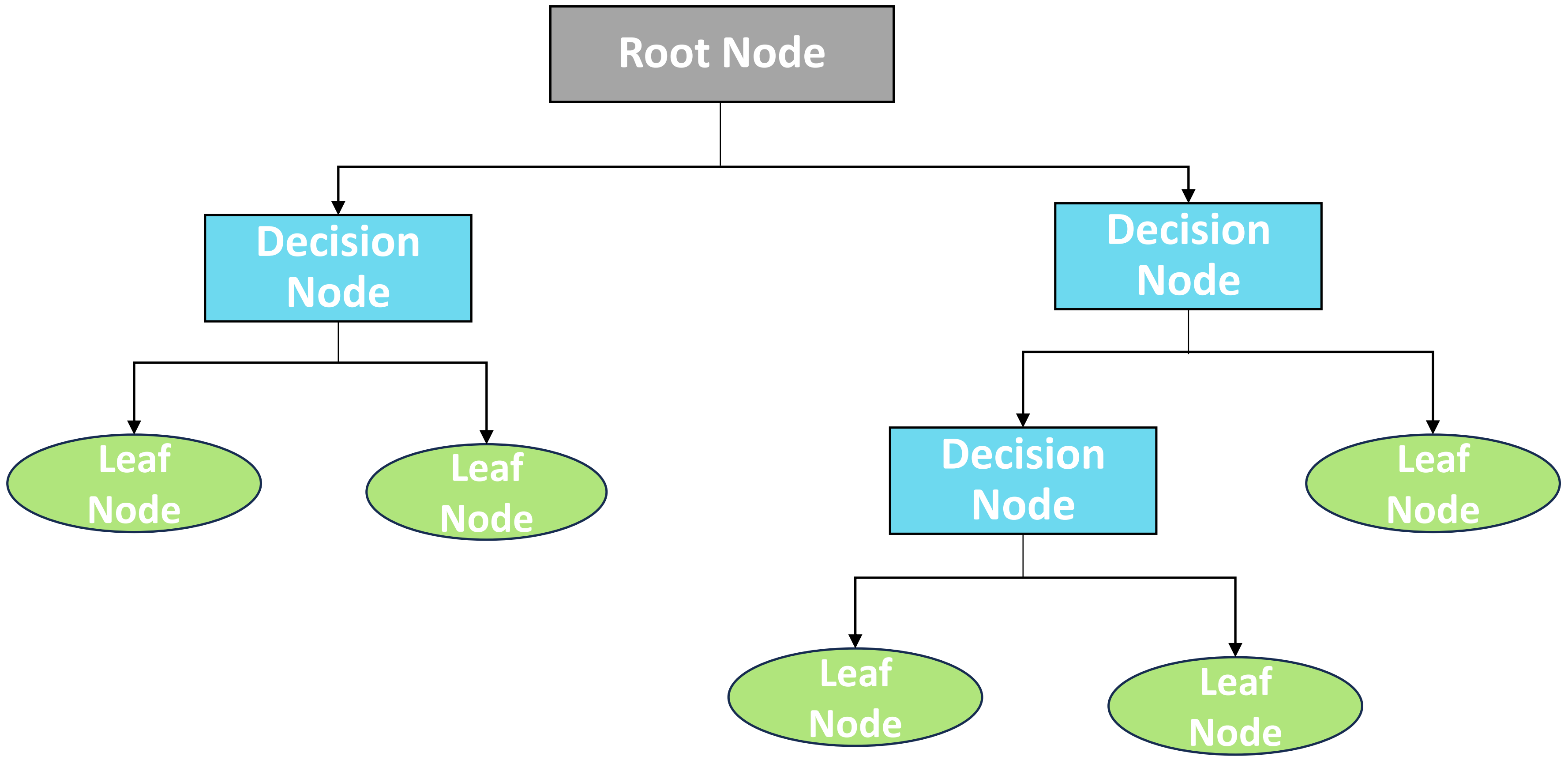

Fig 5.4: A Decision Tree

Unsupervised and Advanced Machine Learning

Terminology

- **Root node:** the topmost node representing the entire dataset.
- **Splitting:** dividing a node into sub-nodes based on feature values.
- **Leaf/terminal node:** a node that represents a class label or continuous value.
- **Pruning:** removing sub-nodes to prevent overfitting.
- **Depth:** the length of the longest path from the root to a leaf.

Unsupervised and Advanced Machine Learning

Understanding Gini Impurity

- Gini impurity measures how often a randomly chosen element would be incorrectly classified.
- Calculated as $G = 1 - \sum_{i=1}^{n} P_i^2$

  where P_i is the probability of an element being classified to a particular class.
- Used to determine the best feature to split the data at each node.
- Lower Gini impurity indicates a better split.

Unsupervised and Advanced Machine Learning

Understanding Gini Impurity

Example of calculating Gini Index

1. Calculate the total number of samples (N) = 10

2. Calculate the frequency of each class,

   Apple: 4 instances

   Orange: 4 instances

   Banana: 2 instances

Sample	Fruit Type
1	Apple
2	Orange
3	Apple
4	Apple
5	Orange
6	Orange
7	Banana
8	Apple
9	Banana
10	Orange

Table 5.1: A Sample Dataset

Unsupervised and Advanced Machine Learning

Understanding Gini Impurity

Example of calculating Gini Index

3. Calculate Gini index:

   Probability of Apple (p_{Apple}) = 4/10 = 0.4

   Probability of Orange (p_{Orange}) = 4/10=0.4

   Probability of Banana (p_{Banana}) = 2/10=0.2

   Gini index $= 1-(0.4)^2-(0.4)^2-(0.2)^2$

   $=1-0.16-0.16-0.04$

   Gini Index = 0.64

Sample	Fruit Type
1	Apple
2	Orange
3	Apple
4	Apple
5	Orange
6	Orange
7	Banana
8	Apple
9	Banana
10	Orange

Table 5.1: A Sample Dataset

Unsupervised and Advanced Machine Learning

Constructing Decision Trees with Gini Impurity

1. **Start with the Root Node:**Calculate the Gini impurity $Gini(D)$ for the entire dataset D.
2. **Splitting Criteria**: At each node, split the data based on the feature that results in the lowest Gini impurity for the child nodes.
3. **Gini Impurity Calculation**: For each potential split, calculate the Gini impurity, which measures the probability of a random sample being misclassified.
4. **Choosing the Best Split**: Select the feature and split point that minimize the weighted average Gini impurity of the child nodes.
5. **Recursion**: Recursively apply this process to each child node until a stopping criterion (e.g., maximum depth or minimum samples per leaf) is met.

Unsupervised and Advanced Machine Learning

Creating a model using Decision tree

- **Import necessary libraries**: Import the libraries required for building and evaluating the Decision Tree model.
- **Split the dataset**: Divide the data into training and testing sets to evaluate the model's performance.
- **Initialize the model**: Create an instance of the Decision Tree classifier.
- **Train the model**: Fit the classifier to the training data.
- **Evaluate the model**: Assess the model's accuracy on the test data.

Unsupervised and Advanced Machine Learning

Introduction to Random Forests Algorithm

- Random Forest algorithm is an ensemble learning method used for both classification and regression tasks that constructs a multitude of decision trees during training time and outputs the class that is the mode of the classes or mean prediction of the individual trees.
- Combines the predictions from multiple trees to improve accuracy and control overfitting.
- Used in various fields such as finance, healthcare, and image recognition for tasks like predicting credit default, diagnosing diseases, and object detection.

Unsupervised and Advanced Machine Learning

Introduction to Random Forests Algorithm

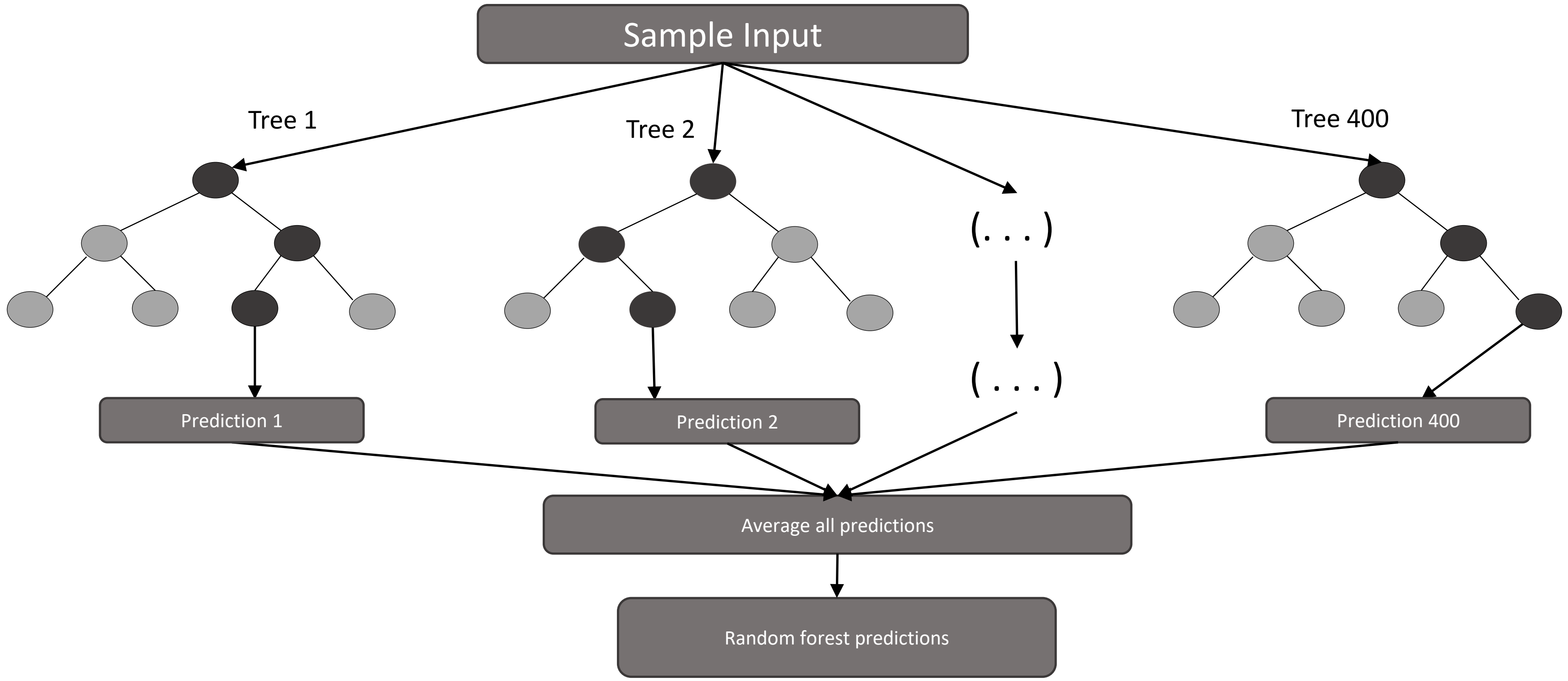

Fig 5.5: A Random Forest Tree

Unsupervised and Advanced Machine Learning

Random Forests Algorithm - History and Motivation

History

- Random forest algorithm was introduced by Leo Breiman and Adele Cutler.
- "Random Forests" by Leo Breiman (2001) provided a comprehensive explanation of Random Forests algorithm, describing their structure and benefits, Adele Cutler co-developed the methodology and contributed to Random forest algorithm’s practical applications.

Unsupervised and Advanced Machine Learning

Random Forests - History and Motivation

Motivation

1. Addressing Overfitting
2. Improving Accuracy
3. Handling High Dimensionality
4. Reducing Variance
5. Feature Importance

Unsupervised and Advanced Machine Learning

Key Hyperparameters of Random Forest

1. Number of Estimators (**n_estimators**)
2. Maximum Depth (**max_depth**)
3. Minimum Samples Split (**min_samples_split**)
4. Minimum Samples Leaf (**min_samples_leaf**)
5. Maximum Features (**max_features**)
6. Bootstrap (**bootstrap**)
7. Criterion (**criterion**)
8. Maximum Samples (**max_samples**)
9. Random State (**random_state**)
10. Class Weight (**class_weight**)

Unsupervised and Advanced Machine Learning

Number of Estimators and Features in Subsets

Number of estimators (n_estimators)

- Each tree in the Random Forest makes an independent prediction, and the final prediction is typically the average or the majority vote of these individual tree predictions.
- More trees generally lead to a more robust model, as they reduce the variance and improve generalization to new data.

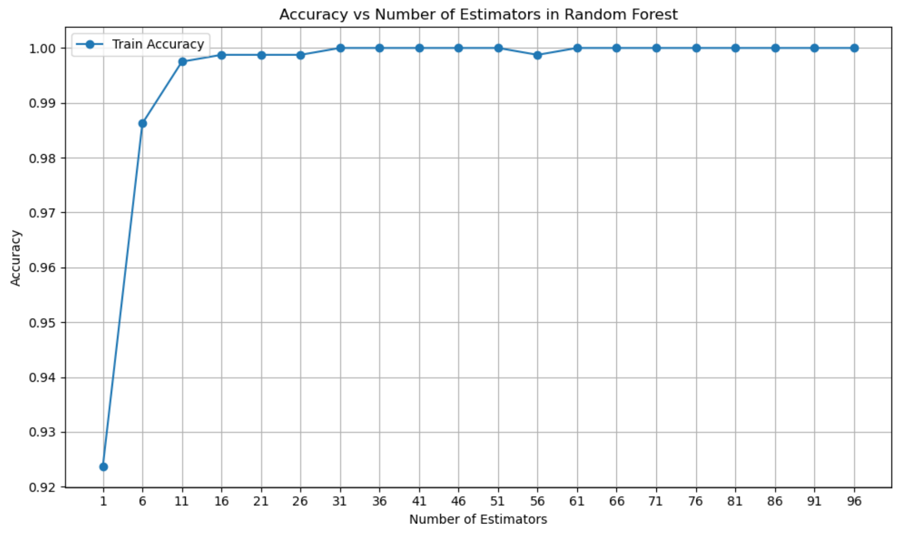

Fig 5.6: Plot between Accuracy and Number of estimators in Random Forest Algorithm

Unsupervised and Advanced Machine Learning

Number of Estimators and Features in Subsets

Maximum Features (max_features)

- **max_features** : the number of features to consider when looking for the best split at each node.
- The “**max_features**” hyperparameter controls the number of features randomly chosen at each split point, adding randomness and diversity to the trees.
- By limiting the number of features considered, Random Forests reduce the correlation between individual trees, improving overall model performance.
- Various options considered are **sqrt, log2, None**

Unsupervised and Advanced Machine Learning

Number of Estimators and Features in Subsets

Interaction Between n_estimators and max_features

- Using a higher number of trees with a smaller subset of features can effectively balance performance and computational efficiency.
- Increasing n_estimators generally decreases the model variance, reducing overfitting.
- Adjusting max_features can help control the trade-off between bias and variance, affecting both overfitting and underfitting.

Unsupervised and Advanced Machine Learning

Bootstrapping and Out-of-Bag Error

Bootstrapping in Random Forests Algorithm

- Bootstrapping involves sampling with replacement from the original dataset to create multiple subsets of data for training individual decision trees.
- Each decision tree in a Random Forest is trained on a different bootstrap sample, which introduces diversity and reduces overfitting compared to using the entire dataset for each tree.

Unsupervised and Advanced Machine Learning

Bootstrapping and Out-of-Bag Error

Out-of-Bag (OOB) Error Estimation

- When training each tree, about one-third of the original data points are not included in the bootstrap sample. These are referred to as out-of-bag (OOB) samples.
- **Calculation of OOB error**
  - For each data point, its predictions across all trees for which it was OOB are averaged to obtain an OOB prediction.
  - OOB error is computed as the average error of predictions on OOB samples across all trees. $\frac{1}{N}\sum_{i=1}^{N} I(\hat{Y}_{OOB_i} \neq y_i)$

Unsupervised and Advanced Machine Learning

Bootstrapping and Out-of-Bag Error

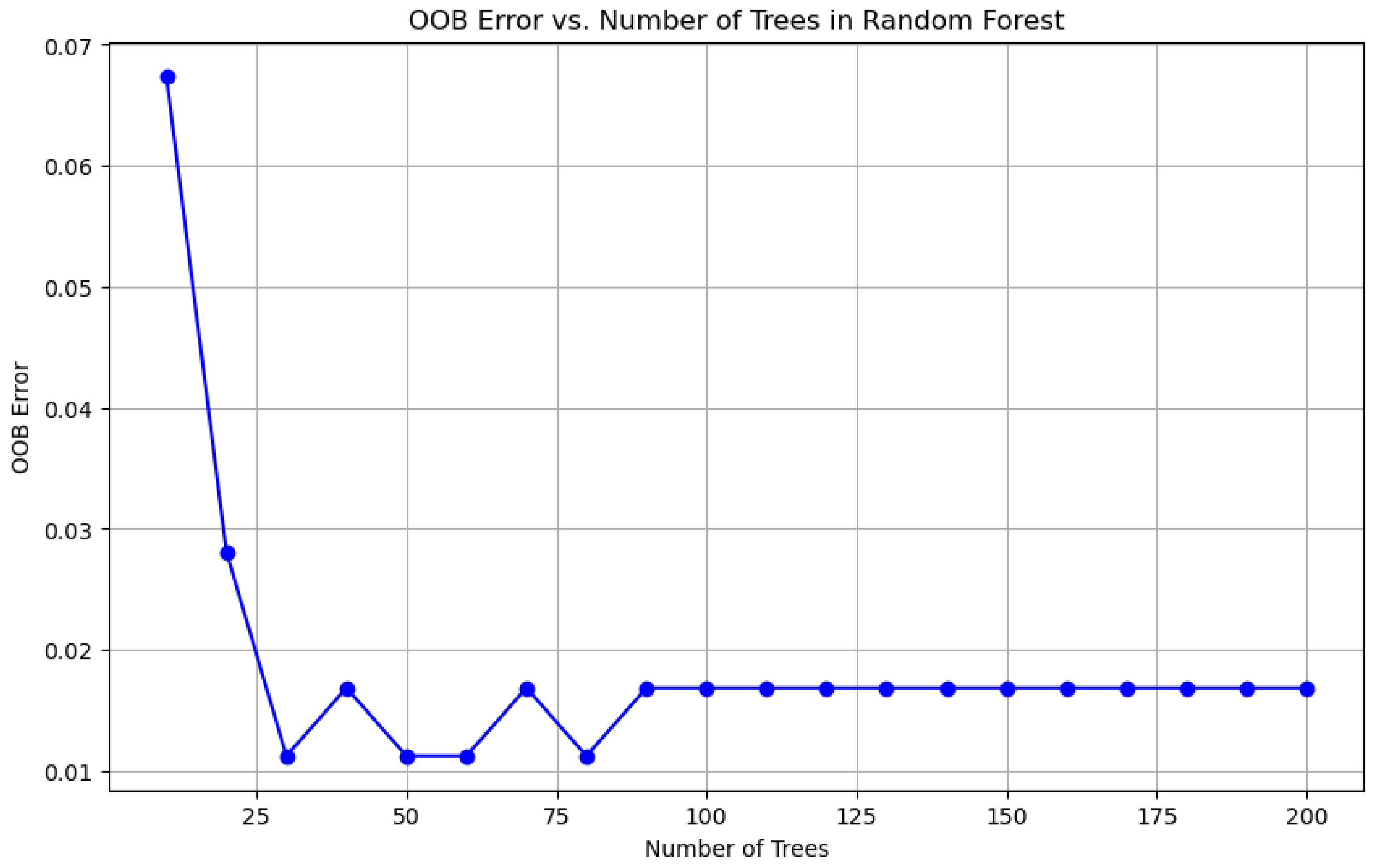

Fig 5.7: Plot between OOB Error vs the Number of Trees in the Random Forest

Unsupervised and Advanced Machine Learning

Coding - Classification with Random Forest Classifier

Steps:

1. **Data Collection and Exploration**

   Load the dataset and identify the key features and target variable.

2. **Data Preprocessing**

   Clean and prepare the data for modeling.

   - Handle missing values
   - Encode categorical variables
   - Normalize numerical features

3. **Feature Engineering**

   Create additional features to improve the predictive power of the model.

Unsupervised and Advanced Machine Learning

Coding - Classification with Random Forest Classifier

Steps:

4. **Splitting Data**

   Split the data into training and testing sets

5. **Implementing Random forest classification Model**

   Train a random forest classifier using the training dataset to predict the survival of the passenger.

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV
rf = RandomForestClassifier(random_state=42)
```

Unsupervised and Advanced Machine Learning

Coding - Classification with Random Forest Classifier

Steps:

```
param_grid = { 'n_estimators': [100, 200, 300], 'max_depth': [None, 10, 20, 30],
'min_samples_split': [2, 10, 20], 'min_samples_leaf': [1, 5, 10] }
grid = GridSearchCV(rf, param_grid, refit=True, verbose=2, cv=5)
grid.fit(X_train, y_train)
```

6. **Model Evaluation**

   Assess the performance of the Random forest classifier model using metrics such as accuracy, precision, recall, and F1-score on the test dataset.

Unsupervised and Advanced Machine Learning

Coding - Regression with Random Forest Regressor -Data

Steps:

1. **Data Collection and Exploration**

   Load the dataset and identify the key features and target variable.

2. **Data Preprocessing**

   Clean and prepare the data for modeling.

   - ➢ Handle missing values
   - ➢ Encode categorical variables
   - ➢ Normalize numerical features

3. **Feature Engineering**

   Create additional features to improve the predictive power of the model.

Unsupervised and Advanced Machine Learning

Coding - Regression with Random Forest Regressor -Data

Steps:

4. **Splitting Data**

   Split the data into training and testing sets

5. **Implementing Random forest regressor Model**

   Train a random forest regressor using the training dataset to predict the survival of the passenger.

```
from sklearn.ensemble import RandomForestRegressor
rf_regressor = RandomForestRegressor(n_estimators=100, random_state=42)
rf_regressor.fit(X_train, y_train)
```

Unsupervised and Advanced Machine Learning

Coding - Regression with Random Forest Regressor -Data

Steps:

6. **Model Evaluation**

   Assess the performance of the Random forest regressor model using metrics such as accuracy, precision, recall, and F1-score on the test dataset.

Unsupervised and Advanced Machine Learning

Introduction to K-Means Clustering Algorithm

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

Main tasks

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

Unsupervised and Advanced Machine Learning

Clustering - General Overview

- Clustering is an unsupervised learning technique that groups data points into clusters based on similarity.
- It aims to discover hidden patterns or structure within data without prior knowledge of labels.

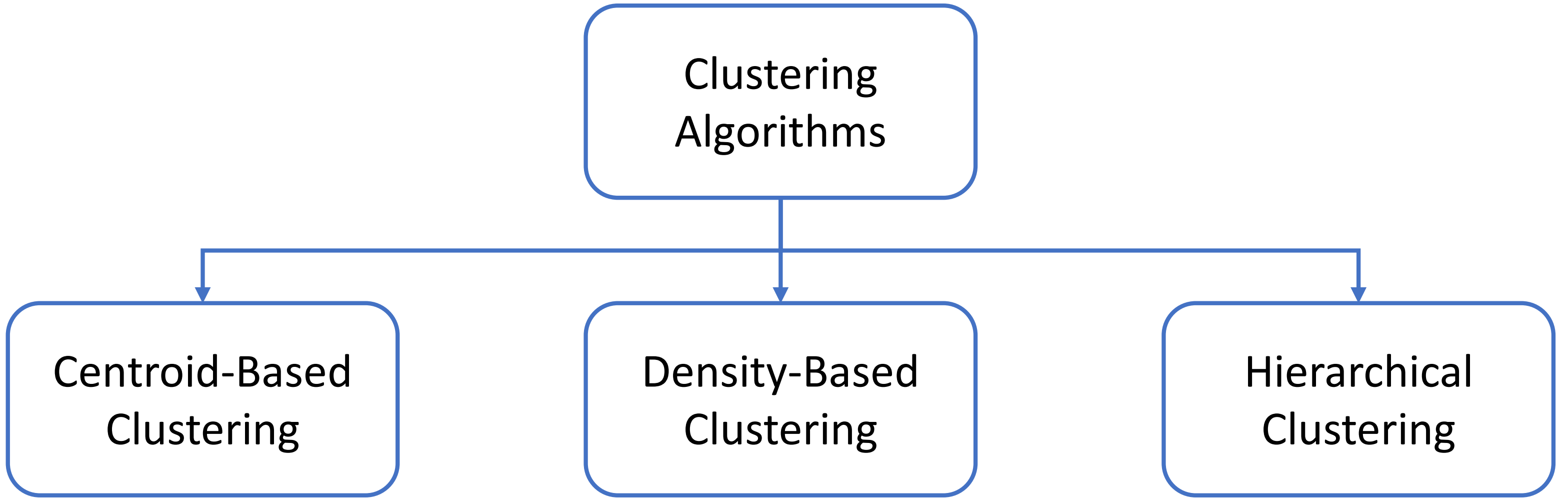

Fig 5.8: Types of Clustering algorithm

Unsupervised and Advanced Machine Learning

Clustering - General Overview

Key concepts used in clustering

- Distance Metrics
- Cluster Evaluation

Applications of clustering

- Customer segmentation
- Image segmentation
- Anomaly Detection

Unsupervised and Advanced Machine Learning

K-Means Clustering Theory

Overview of K-Means Clustering Algorithm

- K-Means is a centroid-based clustering algorithm that partitions data into K clusters by minimizing the variance within each cluster.
- The goal is to assign data points to clusters such that the sum of squared distances (Euclidean distance) between data points and their assigned cluster centroids is minimized.

Unsupervised and Advanced Machine Learning

K-Means Clustering Theory

Algorithm Steps:

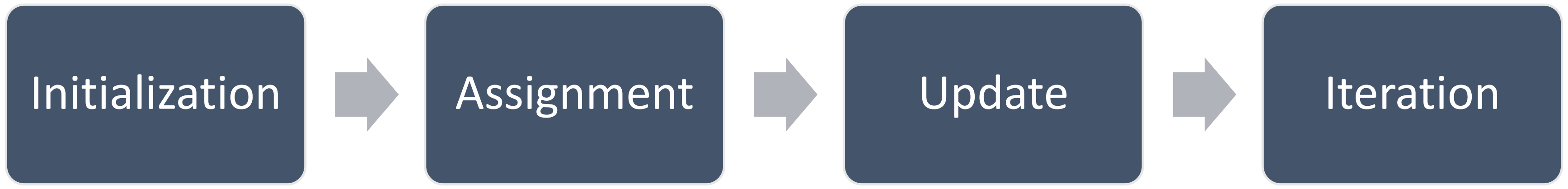

Fig 5.9:K-Means Clustering Algorithm Steps

Key Concepts of K-Means Clustering

- Centroid Initialization
- Convergence
- Complexity

Unsupervised and Advanced Machine Learning

K-Means Clustering coding

1. Load the dataset
2. Data preprocessing and feature engineering
3. Choose the value of k
4. Initialize the model

```
from sklearn.cluster import Kmeans

kmeans = KMeans(n_clusters=k, random_state=0)
```

Unsupervised and Advanced Machine Learning

K-Means Clustering coding

5. Fit the model

   kmeans.fit(X)

6. Predict cluster labels

   y_kmeans = kmeans.predict(X)

5. Visualize the results

6. Plot the cluster centers

   *centers = kmeans.cluster_centers_*

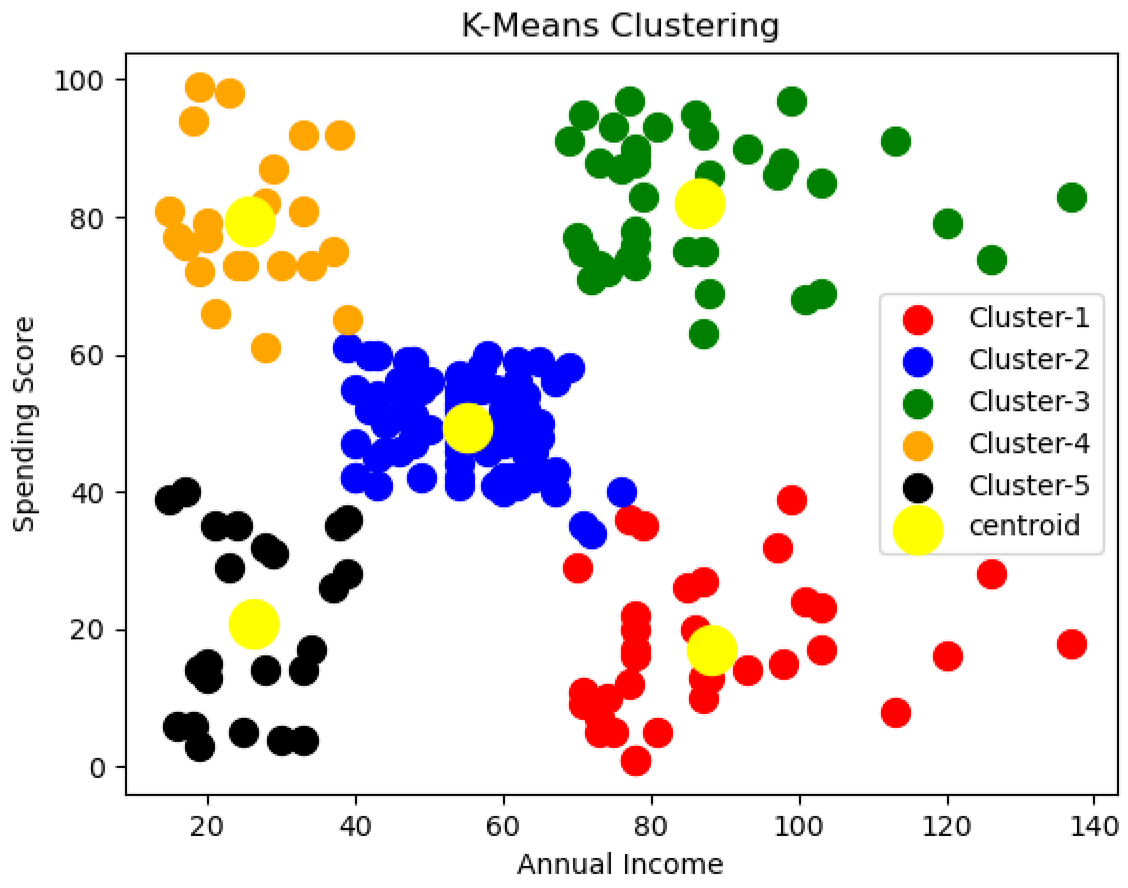

Fig 5.10 : K-Means Clustering

Unsupervised and Advanced Machine Learning

K-Means Color Quantization

Color Quantization

- Color quantization is the process of reducing the number of distinct colors in an image. It's commonly used in image compression to reduce file size while maintaining visual similarity.
- Purpose of color quantization is to reduce the number of colors in an image to decrease memory usage and file size, and to simplify the representation for various image processing tasks.

Unsupervised and Advanced Machine Learning

K-Means Color Quantization

- K-Means clustering can be applied to the color space of an image to identify a set of representative colors (centroids) that approximate the original colors.
- **Process:**
    - Each pixel in the image is treated as a data point in the color space (e.g., RGB).
    - K-Means clusters these pixels into K clusters, where each cluster represents a color in the quantized image.

Unsupervised and Advanced Machine Learning

K-Means Color Quantization

Steps for Implementing K-Means Color Quantization

1. Import Libraries
2. Load and Preprocess the Image:
3. Apply K-Means Clustering:
4. Replace Colors with Cluster Centroids:
5. Display the Original and Quantized Images

Unsupervised and Advanced Machine Learning

K-Means Color Quantization

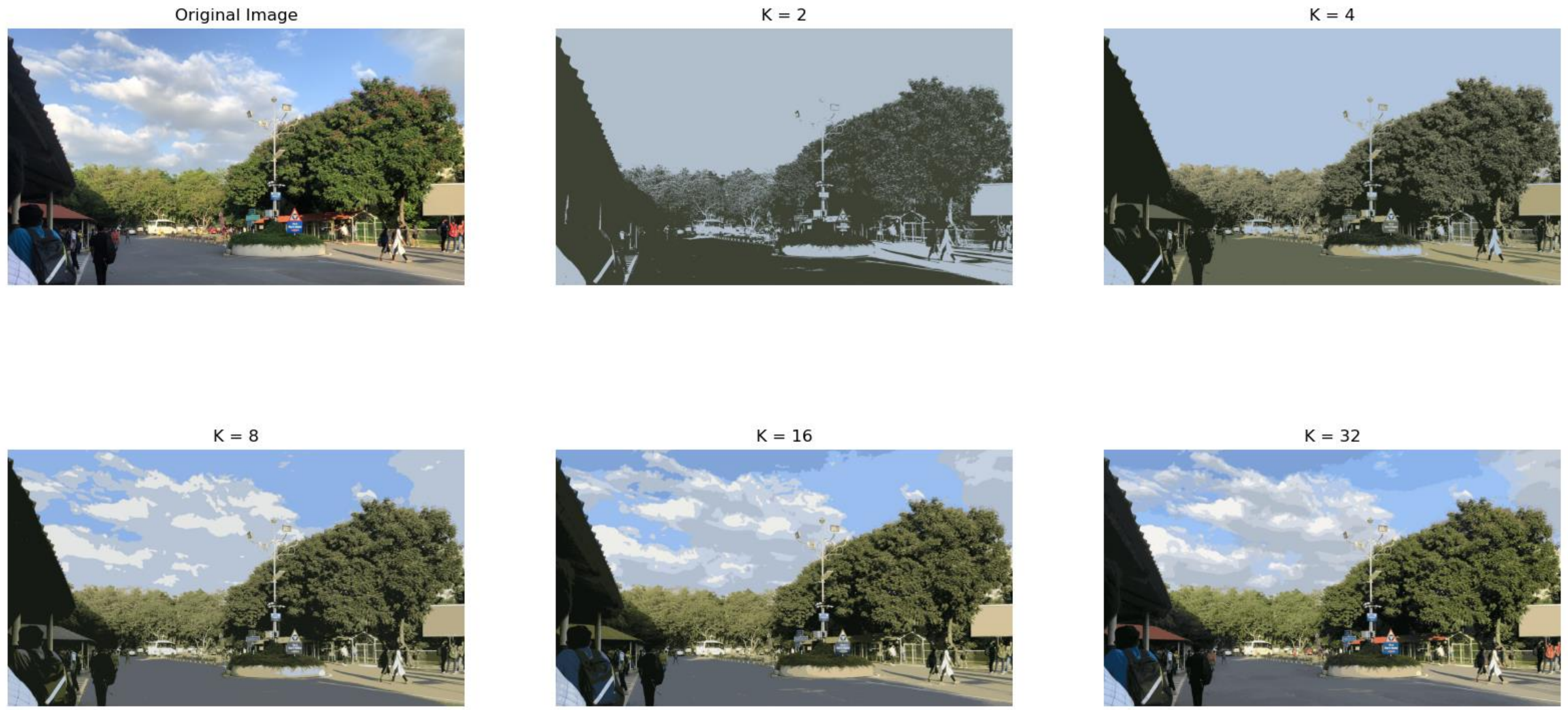

Fig 5.11 : Color Quantization using K-Means algorithm

Unsupervised and Advanced Machine Learning

K-Means Color Quantization

Knowledge Check

In K-Means Color Quantization, what does the value of K represent?

A. The number of distinct colors in the resulting image.

B. The total number of pixels in the image.

C. The image's resolution.

D. The intensity threshold for colors.

Unsupervised and Advanced Machine Learning

K-Means Color Quantization

Knowledge Check

In K-Means Color Quantization, what does the value of K represent?

A. The number of distinct colors in the resulting image.

B. The total number of pixels in the image.

C. The image's resolution.

D. The intensity threshold for colors.

Unsupervised and Advanced Machine Learning

K-Means Clustering Algorithm Project Overview

Problem Statement:

A mall wants to better understand its customer base and tailor its marketing strategies to different customer segments. use the K-means clustering algorithm to segment its customers into distinct groups based on their characteristics and spending behavior.

Objective:

To utilize the K-means clustering algorithm to segment the customer base of a mall into distinct groups based on their demographic characteristics and spending behavior based on "Mall_Customers.csv".

Unsupervised and Advanced Machine Learning

K-Means Clustering Algorithm Project Overview

Steps:

1. **Data Collection and Exploration**

   Load the dataset and identify the key features and target variable.

2. **Data Preprocessing**

   Clean and prepare the data for modeling.

   - Handle missing values
   - Encode categorical variables
   - Normalize numerical features

3. **Feature Engineering**

   Create additional features to improve the predictive power of the model.

Unsupervised and Advanced Machine Learning

K-Means Clustering Algorithm Project Overview

Steps:

4. **Implementing K-Means Clustering**

```
from sklearn.cluster import KMeans
algorithm = (KMeans(n_clusters=4,init='k-means++',n_init = 10,
        max_iter=300, tol=0.0001,  random_state= 111  , algorithm='elkan’) )
algorithm.fit(X1)
labels1 = algorithm.labels_
centroids1 = algorithm.cluster_centers_
```

5. **Model Evaluation**

Assess the performance of the K-Means model using appropriate metrics such as silhouette score, inertia, or Davies–Bouldin index.

Unsupervised and Advanced Machine Learning

Introduction to Hierarchical Clustering

- Hierarchical clustering is a method of cluster analysis which seeks to build a hierarchy of clusters. It is used to group similar objects into clusters where each object belongs to a single cluster.
- Hierarchical clustering creates a tree-like structure called a dendrogram, which represents the nested grouping of patterns and similarity levels at which groups merge or split.

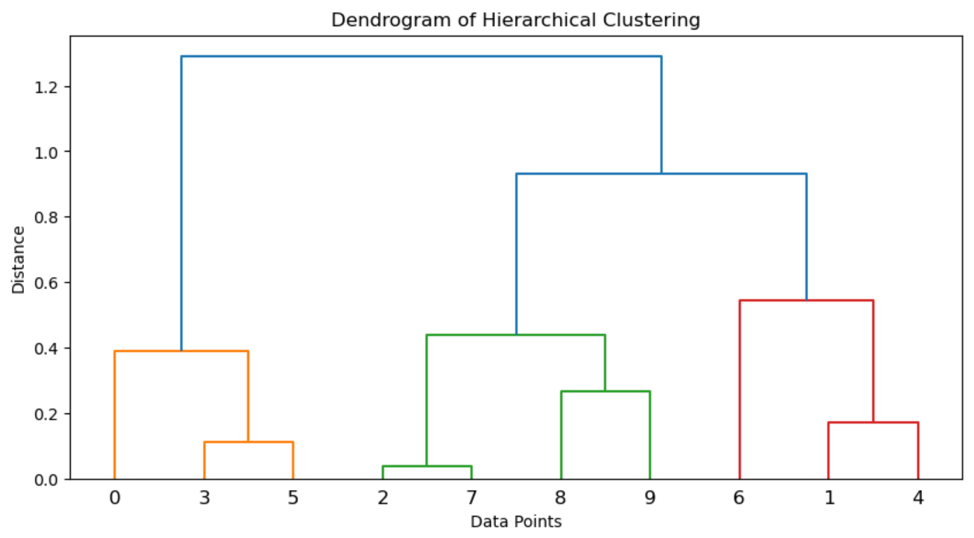

Fig 5.11 : Dendrogram of a Hierarchical Clustering

Unsupervised and Advanced Machine Learning

Hierarchical Clustering - Theory and Intuition

- There are two main types of hierarchical clustering: Agglomerative and Divisive.

Agglomerative Hierarchical Clustering (bottom-up) :

- **Process:**
  - **Step 1- Initialization:**
    - Start with each data point as a single cluster.
  - **Step 2- Iterative Merging:**
    - At each iteration, merge the two closest clusters until all points are merged into a single cluster or a stopping criterion is met.

Unsupervised and Advanced Machine Learning

Hierarchical Clustering - Theory and Intuition

Divisive Hierarchical Clustering (top-down):

- **Process:**
  - **Step 1- Initialization:**
    - Start with all data points in a single cluster.
  - **Step 2- Iterative Splitting:**
    - At each iteration, split the cluster with the highest dissimilarity until each data point is in its own cluster or a stopping criterion is met.

Unsupervised and Advanced Machine Learning

Hierarchical Clustering - Theory and Intuition

Linkage Criteria:

- **Single Linkage (Minimum Linkage):**
  - Distance between the closest points of the clusters.
- **Complete Linkage (Maximum Linkage):**
  - Distance between the farthest points of the clusters.
- **Average Linkage:**
  - Average distance between points of the clusters.
- **Ward's Method:**
  - Minimizes the variance within clusters.

Unsupervised and Advanced Machine Learning

Hierarchical Clustering - Theory and Intuition

Knowledge Check

When performing hierarchical clustering, which linkage criterion is specifically designed to create clusters with the smallest possible increase in total within-cluster variance at each step of the clustering process?

A. The method that calculates the distance between the nearest points of two clusters.

B. The approach that measures the distance between the farthest points of two clusters.

C. The technique that computes the average distance between all pairs of points in two clusters.

D. The strategy that aims to minimize the sum of squared differences within each cluster.

Unsupervised and Advanced Machine Learning

Hierarchical Clustering - Theory and Intuition

Knowledge Check

When performing hierarchical clustering, which linkage criterion is specifically designed to create clusters with the smallest possible increase in total within-cluster variance at each step of the clustering process?

A. The method that calculates the distance between the nearest points of two clusters.

B. The approach that measures the distance between the farthest points of two clusters.

C. The technique that computes the average distance between all pairs of points in two clusters.

D. The strategy that aims to minimize the sum of squared differences within each cluster.

Unsupervised and Advanced Machine Learning

Data Visualization with Scikit-Learn

Steps:

1. Import Libraries
2. Load data
3. Pre process data
4. **Data Visualization with PCA:**
   - Use Principal Component Analysis (PCA) for dimensionality reduction to visualize high-dimensional data.

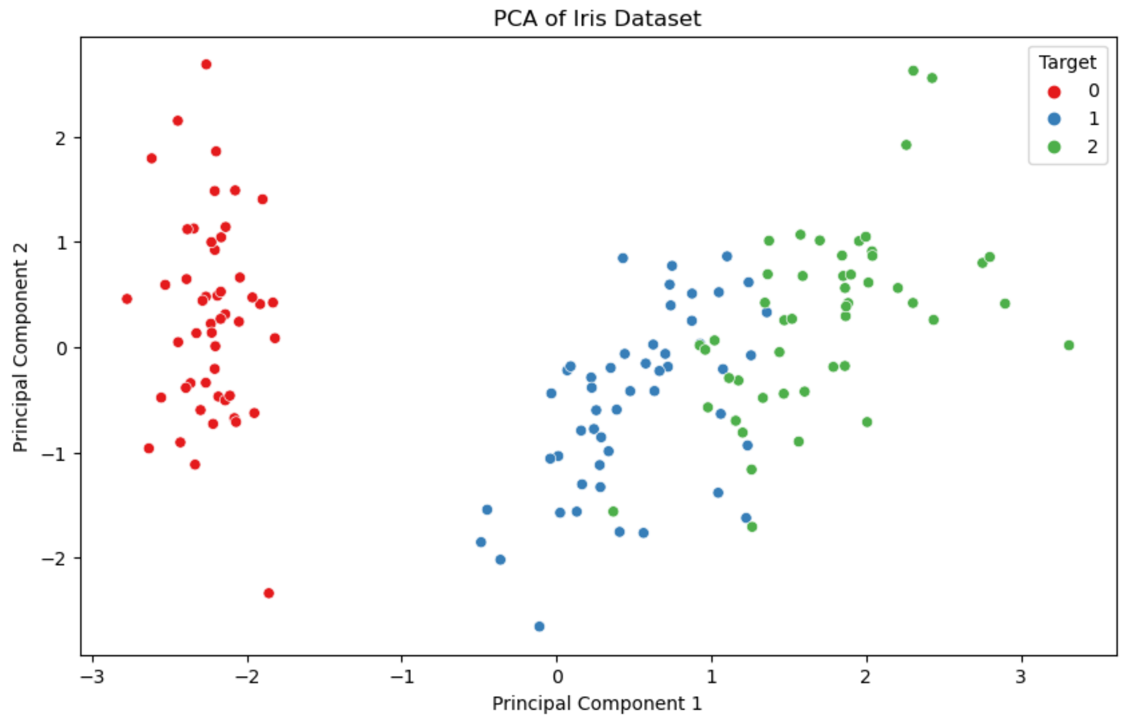

Fig 5.12: Data Visualization with PCA

Unsupervised and Advanced Machine Learning

Data Visualization with Scikit-Learn

5. Data Visualization with Clustering

- Use clustering algorithms like KMeans to visualize clusters.

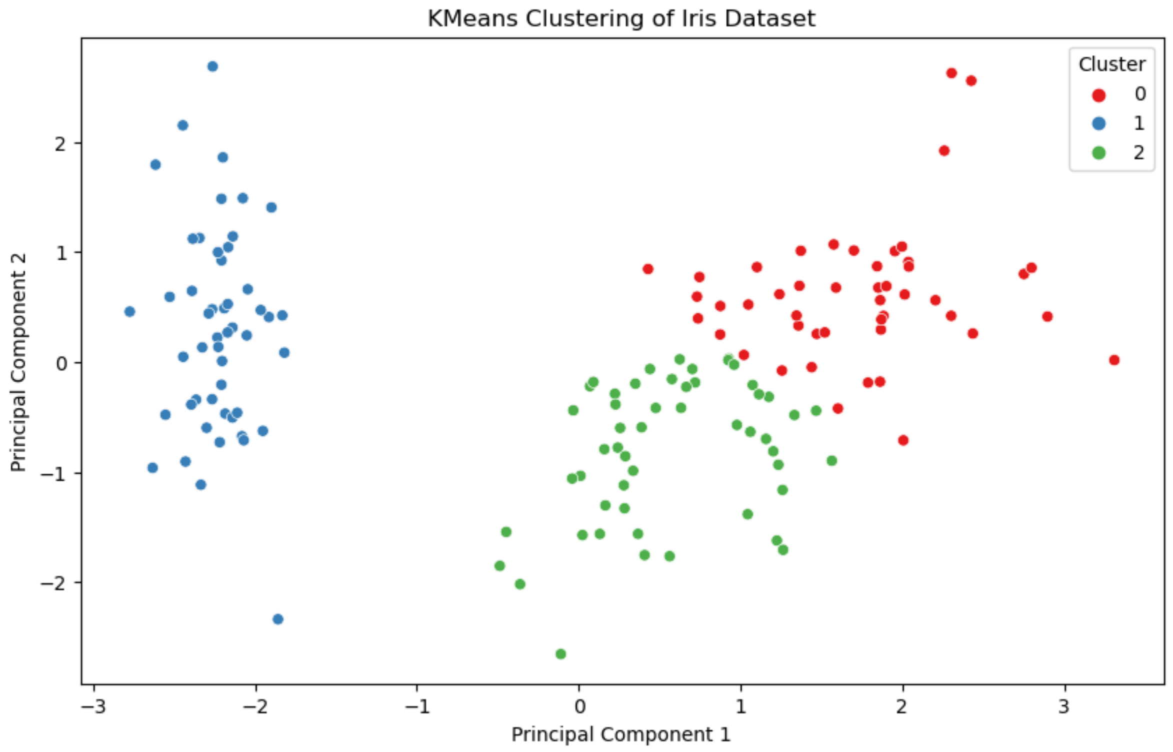

Fig 5.13: Data Visualization with Clustering

Unsupervised and Advanced Machine Learning

Data Visualization with Scikit-Learn

5. Visualize Feature Importances

For example, using a Random Forest classifier.

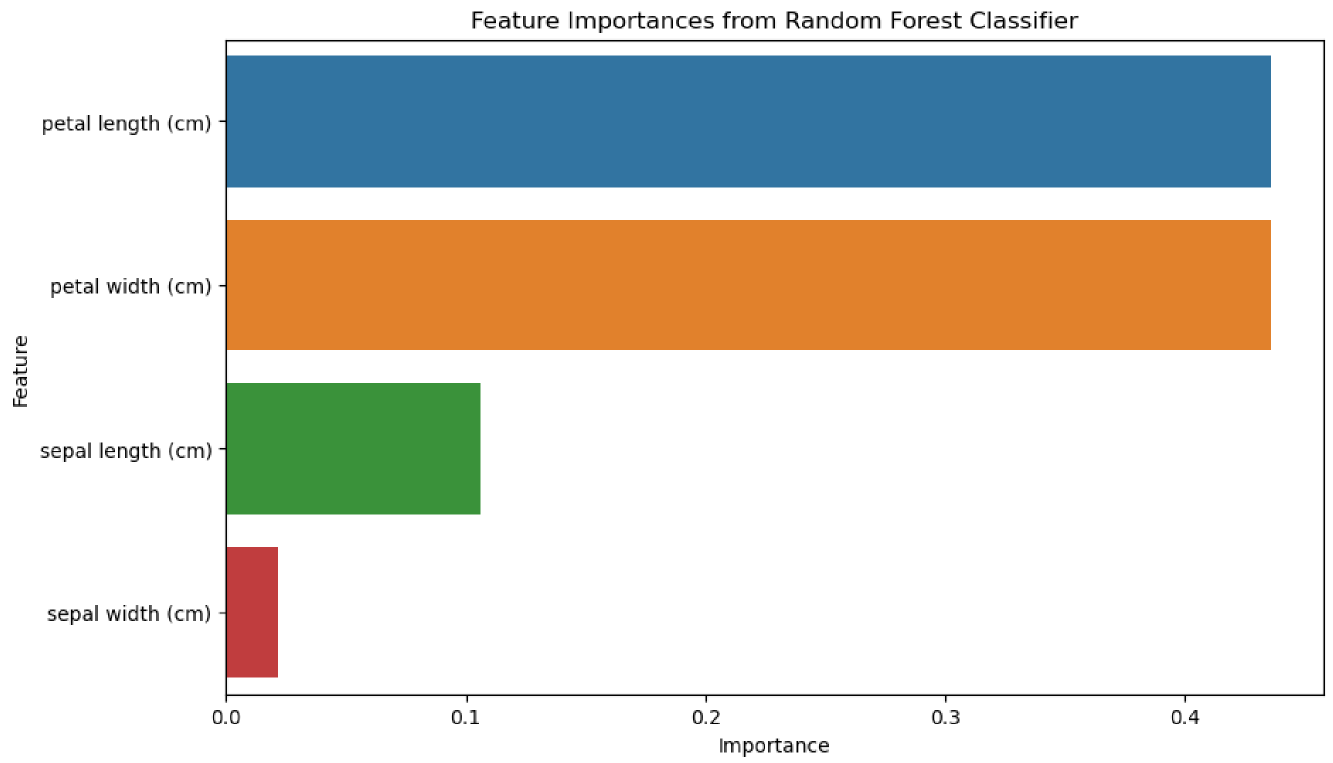

Fig 5.14: Visualizing Feature importance

Unsupervised and Advanced Machine Learning

Introduction to Principal Component Analysis (PCA)

Definition of PCA

- PCA is a dimensionality reduction technique used to reduce the number of variables in a dataset while preserving as much information as possible.
- It transforms the data into a new set of variables, the principal components (PCs).

Unsupervised and Advanced Machine Learning

Introduction to Principal Component Analysis (PCA)

Advantages of PCA

- Simplifies the complexity of high-dimensional data.
- Reduces noise and redundancy.
- Facilitates data visualization.
- Enhances the performance of machine learning algorithms by reducing overfitting.

Unsupervised and Advanced Machine Learning

Introduction to Principal Component Analysis

Knowledge Check

Which of the following is a key benefit of applying Principal Component Analysis (PCA) in handling datasets?

A. It always produces the most accurate machine learning models.

B. It helps manage the intricacies of datasets with many variables.

C. It completely prevents any loss of information.

D. It speeds up the process of data collection.

Unsupervised and Advanced Machine Learning

Introduction to Principal Component Analysis

Knowledge Check

Which of the following is a key benefit of applying Principal Component Analysis (PCA) in handling datasets?

A. It always produces the most accurate machine learning models.

B. It helps manage the intricacies of datasets with many variables.

C. It completely prevents any loss of information.

D. It speeds up the process of data collection.

Unsupervised and Advanced Machine Learning

PCA - Theory and Intuition

Core Concepts

- **Variance:** Measure of how much the data points differ from the mean.
- **Covariance:** Measure of how much two variables change together.
- **Eigenvectors and Eigenvalues:** Determine the direction and magnitude of the new axes (principal components).

Unsupervised and Advanced Machine Learning

PCA - Manual Implementation in Python

Steps in PCA Manual Implementation

- Standardize the data.
- Compute the covariance matrix.
- Calculate the eigenvalues and eigenvectors of the covariance matrix.
- Sort eigenvectors by eigenvalues in descending order.
- Select the top k eigenvectors to form a new feature space.
- Transform the original data into the new feature space.

Unsupervised and Advanced Machine Learning

PCA - Manual Implementation in Python

Steps in PCA Manual Implementation

- Standardize the data.

```
from sklearn.preprocessing import StandardScaler
 X_standardized = StandardScaler().fit_transform(X)
```

- Compute the covariance matrix.

```
import numpy as np
covariance_matrix = np.cov(X_standardized.T)
```

- Calculate the eigenvalues and eigenvectors of the covariance matrix.

```
eigenvalues, eigenvectors = np.linalg.eig(covariance_matrix)
```

Unsupervised and Advanced Machine Learning

PCA - Manual Implementation in Python

Steps in PCA Manual Implementation

- Sort and select the top k eigenvectors to form a new feature space.

```
sorted_index = np.argsort(eigenvalues)[::-1]

sorted_eigenvectors = eigenvectors[:, sorted_index]

k = 2

eigenvector_subset = sorted_eigenvectors[:, :k]
```

- Transform the original data into the new feature space.

```
X_reduced = np.dot(X_standardized, eigenvector_subset)
```

Unsupervised and Advanced Machine Learning

PCA - Manual Implementation in Python

Steps in PCA Manual Implementation

- Sort and select the top k eigenvectors to form a new feature space.

```
sorted_index = np.argsort(eigenvalues)[::-1]

sorted_eigenvectors = eigenvectors[:, sorted_index]

 k = 2

eigenvector_subset = sorted_eigenvectors[:, :k]
```

- Transform the original data into the new feature space.

```
X_reduced = np.dot(X_standardized, eigenvector_subset)
```

Unsupervised and Advanced Machine Learning

PCA - SciKit-Learn

Steps in PCA implementation using SciKit-Learn

- Implementation

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
X_reduced = pca.fit_transform(X_standardized)
```

- Variance ratio

```
print(pca.explained_variance_ratio_)
```

?

Thank You !!!