NAME : Mehul Uttam
PRN : 1032222936
TY CSE AIDS - A (A1 Batch)

Assignment 04

Title: Support Vector Machine (SVM) Classifier and comparison with Tree based classifier
Aim: To perform SVM classification using Python and compare it with tree based classifier.
Objectives: To implement SVM classifier and perform comparison with Decision Tree.

Problem Statement: Implementation of SVM. Comparison with tree based classifier.

Algorithm: SVM, Decision Tree

SVM:
Support vector machine is highly preferred by many as it produces significant accuracy with less computation power. Support Vector Machine, abbreviated as SVM can be used for both regression and classification tasks. But, it is widely used in classification objectives.

- The line that maximizes the minimum margin is a good best.
- This maximum-margin separator is determined by a subset of the data points.
  - Data points in this subset are called “support vectors”.
  - It will be useful computationally if only a small fraction of the data points are support vectors
  - The support vectors are used to decide which side of the separator a test case is on.

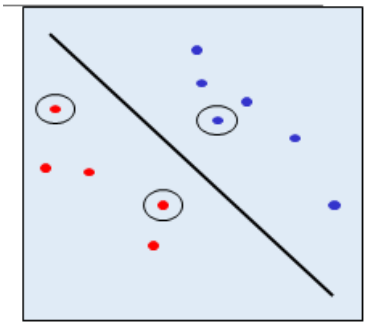

Fig.1 Support vectors are indicated by circles around them

Hyper-planes are decision boundaries that help classify the data points. Data points falling on either side of the hyperplane can be attributed to different classes. Also, the dimension of the hyperplane depends upon the number of features. If the number of input features is 2, then the hyperplane is just a line. If the number of input features is 3, then the hyperplane becomes a two-dimensional plane. It becomes difficult to imagine when the number of features exceeds 3.

Main Idea behind the kernel function:
1. Starts with data in low dimension.
2. Moves the data into a higher dimension if it is not linearly separable by a hyperplane.
3. Finds the Support Vector Classifier (hyperplane) that separates the higher dimensional data into two groups.

SVM (Support Vector Machines) and decision tree algorithms are both commonly used in machine learning for classification tasks. However, they differ in their approach to classification and their strengths and weaknesses.
Here are some of the key differences between SVM and decision tree algorithms:
Approach to classification:
SVM: SVM is a discriminative classifier that separates the data points into different classes using a hyperplane. It tries to find the hyperplane that maximizes the margin between the different classes.
Decision Tree: Decision tree is a type of classification algorithm that builds a tree-like model of decisions and their possible consequences. It uses a tree structure to partition the data into different classes based on the values of the input features.
Handling non-linearly separable data:
SVM: SVM can handle non-linearly separable data by using a technique called kernel trick, which maps the input data to a higher-dimensional feature space where the data becomes separable.
Decision Tree: Decision tree may not be able to handle non-linearly separable data very well. It may require pre-processing or feature engineering to make the data separable.
Interpretability:
SVM: SVM is not very interpretable. It provides a black-box model that is difficult to understand and explain.
Decision Tree: Decision tree is more interpretable than SVM. It provides a tree-like structure that can be easily visualized and understood.
Overfitting:
SVM: SVM is less prone to overfitting because it tries to maximize the margin between the different classes, which leads to a simpler decision boundary.
Decision Tree: Decision tree is more prone to overfitting because it can create complex decision boundaries that may not generalize well to new data.

Input:
https://github.com/psvm-tallman/ML-LAB/blob/main/Social_Network_Ads.csv

Platform: Jupyter Notebook

Output:

In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
df = pd.read_csv('Social_Network_Ads.csv')
x = df.iloc[:,:-1].values
y = df.iloc[:,-1].values
```

In [5]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.25, random_state = 42)
```

In [6]:
```
print(X_train)
```

```
[[    57 122000]
 [    39  71000]
 [    47  25000]
 [    24  19000]
 [    36  50000]
 [    32 150000]
 [    48  29000]
 [    30 107000]
 [    60  34000]
 [    38  61000]
 [    33  31000]
 [    39  71000]
 [    55  39000]
 [    49  39000]
 [    43 112000]
 [    27  20000]
 [    26  17000]
 [    37  93000]
 [    42  54000]
 [    35  61000]
 [    29  75000]
```

```
 [    49  36000]
 [    45  22000]
 [    35  72000]
 [    24  27000]
 [    26  35000]
 [    43 133000]
 [    39  77000]
 [    32  86000]]
```

In [13]:
```
from sklearn.svm import SVC
classifier = SVC(kernel = 'linear', C=1.0,  random_state = 42)
classifier.fit(X_train, y_train)
```

Out[13]:
```
SVC(kernel='linear', random_state=42)
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [14]:
```
y_pred = classifier.predict(X_test)
```

In [15]:
```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

```
[[59  4]
 [10 27]]
```

In [16]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[16]:
```
0.86
```

To compare SVM and decision tree algorithms, we can conduct experiments on a dataset and evaluate their performance on various metrics such as accuracy, Mean square error. Here is an example of how we can compare the performance of SVM and decision tree algorithms on a dataset:

We will use the breast cancer dataset from scikit-learn library. This is a binary classification problem where we need to predict whether a patient has malignant or benign breast cancer based on various features

Conclusion:
In summary, SVM is a good choice when the data is separable and you want a black-box model that is less prone to overfitting. Decision tree is a

good choice when the data is not separable and you want a more interpretable model.

FAQs:

1. Compare SVM with Decision Tree classifiers.

Feature	SVM (Support Vector Machine)	Decision Tree
Type	Non-parametric, linear/non-linear	Non-parametric, tree-based
Working Principle	Maximizes the margin between classes using a hyperplane	Recursive partitioning based on feature splits
Data Type	Works well with continuous and categorical data with numerical features	Handles both continuous and categorical data easily
Interpretability	Harder to interpret due to the complexity of hyperplanes	Easy to interpret and visualize
Handling Non-linearity	Uses kernel functions to handle non-linear problems	Handles non-linear problems through complex tree structures

Overfitting	Less prone to overfitting, especially with the right kernel	More prone to overfitting, needs pruning and regularization
Training Speed	Slower for larger datasets	Faster training, especially for small/medium datasets
Performance on Large Datasets	May struggle with very large datasets	Handles large datasets better, but may require pruning
Feature Importance	Does not provide inherent feature importance	Provides feature importance based on splits
Robustness to Noise	Sensitive to noise, especially in overlapping classes	More robust to noise

2. Describe various types of kernel functions.

- **Linear Kernel**: The simplest kernel, which is used when the data is linearly separable. It computes the dot product between data points, i.e., K(x,y)=x⋅yK(x, y) = x \cdot yK(x,y)=x⋅y.
    - **Application**: Suitable for problems where the data can be separated by a straight line (linear classification tasks).
- **Polynomial Kernel**: It represents the similarity of vectors in a feature space over polynomials of the original variables. The polynomial kernel of degree d is given by: K(x,y)=(x⋅y+c)dK(x, y) = (x \cdot y + c)^dK(x,y)=(x⋅y+c)d.
    - **Application**: Effective in cases where the relationship between classes is more complex than linear, but still manageable with polynomial terms.
- **Radial Basis Function (RBF) Kernel**: Also known as the Gaussian kernel, it maps the input space into a higher-dimensional space. The RBF kernel is defined as: K(x,y)=exp(−γ∥x−y∥2)K(x, y) = \exp(-\gamma \|x - y\|^2)K(x,y)=exp(−γ ∥ x−y ∥ 2), where γ\gammaγ is a parameter that defines how far the influence of a single training example reaches.
    - **Application**: Commonly used for non-linear data. Suitable for cases where the decision boundary is highly curved.
- **Sigmoid Kernel**: It is similar to the activation function used in neural networks and is defined as:
  K(x,y)=tanh(αx⋅y+c)K(x, y) = \tanh(\alpha x \cdot y + c)K(x,y)=tanh(αx⋅y+c).
    - **Application**: Sometimes used in neural network algorithms and models where relationships between classes resemble a sigmoid function.

3. Give the applications of SVM classifiers.

- **Text and Hypertext Categorization**: SVM is widely used in **text classification** problems due to its ability to handle high-dimensional spaces efficiently.
- **Image Classification**: SVM is employed in **image recognition** tasks, such as **handwriting recognition**, **facial expression recognition**, and **object detection**.
- **Bioinformatics**: It is used in **protein classification**, **gene classification**, and **cancer diagnosis** tasks.
- **Handwriting Recognition**: SVMs are applied in recognizing handwritten characters in OCR systems.
- **Intrusion Detection**: In **cybersecurity**, SVM is used for detecting **anomalies and attacks** in network traffic data.
- **Spam Detection**: SVM classifiers are implemented in email filtering systems to detect **spam** versus **non-spam** emails.