

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.

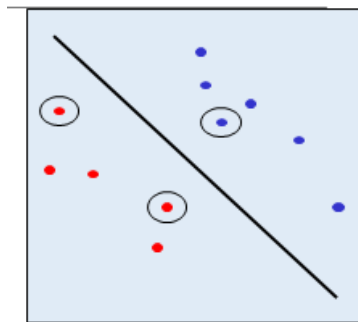


Fig.1 Support vectors are indicated by circles around them

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: Dataset

Platform:

Output:

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

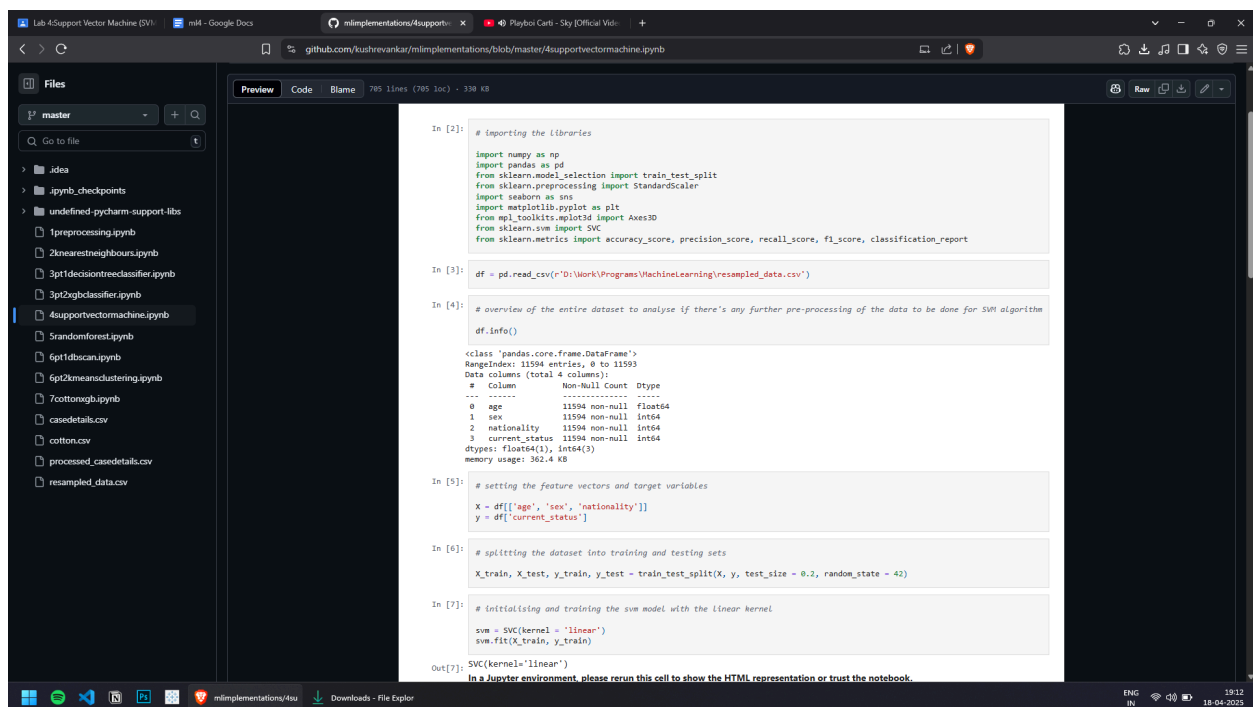
Conclusion:

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. What is a Classifier?
2. Compare SVM with decision tree classifiers.
3. How would you tune SVM parameters?
4. Describe various types of kernel functions.
5. Give the applications of SVM classifiers.
6. State the significance of the kernel function.

Code & Output:



```
In [2]: # importing the libraries
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
import seaborn as sns
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classification_report

In [3]: df = pd.read_csv(r'D:\Work\Programs\MachineLearning\resampled_data.csv')

In [4]: # overview of the entire dataset to analyse if there's any further pre-processing of the data to be done for SVM algorithm
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11594 entries, 0 to 11593
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype  
---  --
 0   age           11594 non-null  float64
 1   sex           11594 non-null  int64  
 2   nationality   11594 non-null  int64  
 3   current_status 11594 non-null  int64  
dtypes: float64(1), int64(3)
memory usage: 362.4 KB

In [5]: # setting the feature vectors and target variables
X = df[['age', 'sex', 'nationality']]
y = df['current_status']

In [6]: # splitting the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)

In [7]: # initialising and training the svm model with the linear kernel
svm = SVC(kernel = 'linear')
svm.fit(X_train, y_train)

Out[7]: SVC(kernel='linear')

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
```

