**U19EC046 | ML | LAB 8**

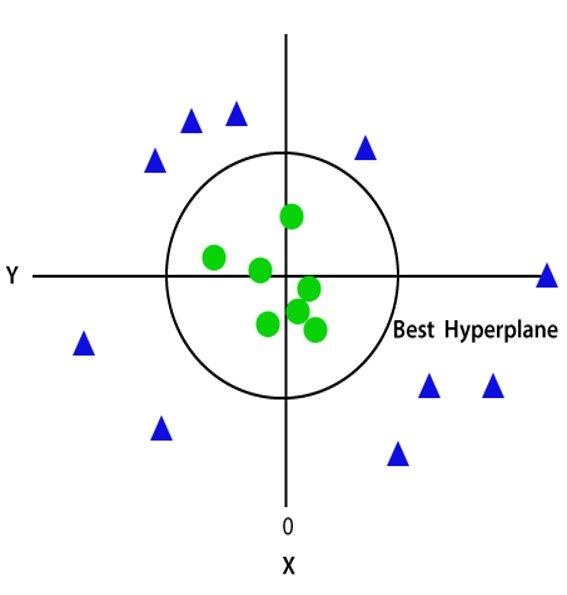
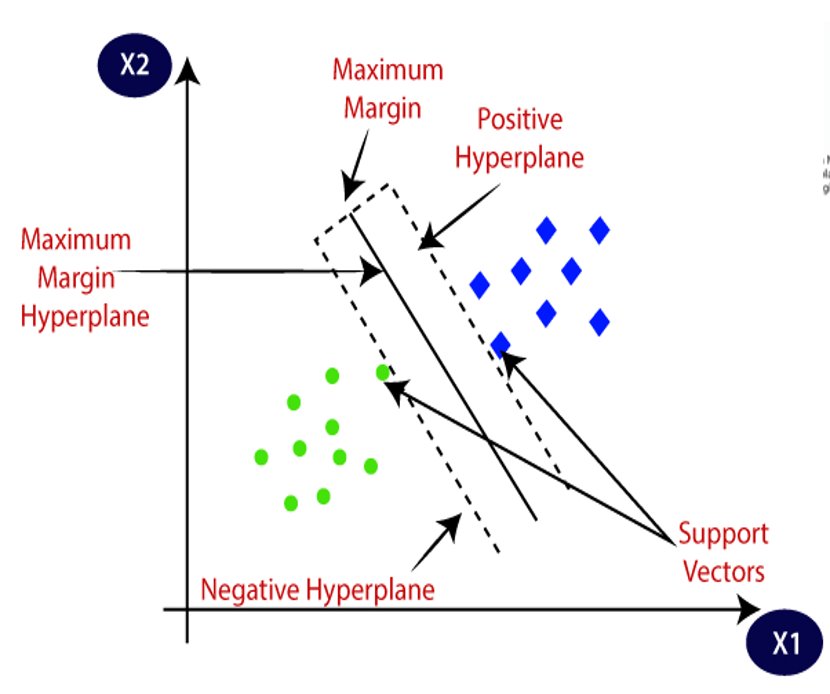
**AIM**

Write a program to implement the support vector machine for a sample training data set stored as a .CSV file. Compute the accuracy of the classifier, considering few test data sets.

**THEORY:**

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems.​ The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.​

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. SVM algorithm can be used for **Face detection, image classification, text categorization,** etc.



**Types of SVM:**

***Linear SVM:*** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.​

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

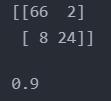
**CODE**

|  |
| --- |
| **import numpy as np**  **import matplotlib.pyplot as plt**  **import pandas as pd** |
| **dataset = pd.read\_csv('Social\_Network\_Ads.csv')**  **X = dataset.iloc[:, :-1].values**  **y = dataset.iloc[:, -1].values**  **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 = 0)**  **from sklearn.preprocessing import StandardScaler**  **sc = StandardScaler()**  **X\_train = sc.fit\_transform(X\_train)**  **X\_test = sc.transform(X\_test)**  **from sklearn.svm import SVC**  **classifier = SVC(kernel = 'linear', random\_state = 0)**  **classifier.fit(X\_train, y\_train)**  **from sklearn.metrics import confusion\_matrix, accuracy\_score**  **cm = confusion\_matrix(y\_test, y\_pred)**  **print(cm)**  **accuracy\_score(y\_test, y\_pred)** |

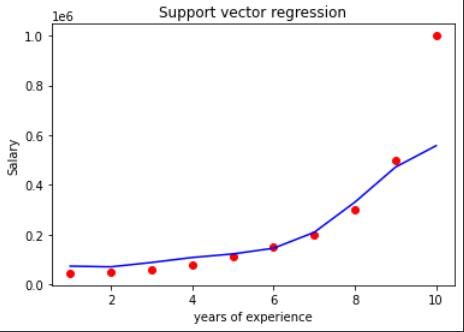
|  |
| --- |
| **import numpy as np**  **import matplotlib.pyplot as plt**  **import pandas as pd**  **dataset = pd.read\_csv('Position\_Salaries.csv')**  **X = dataset.iloc[:, 1:-1].values**  **y = dataset.iloc[:, -1].values**  **y = y.reshape(len(y),1)**  **from sklearn.preprocessing import StandardScaler**  **sc\_X = StandardScaler()**  **sc\_y = StandardScaler()**  **X = sc\_X.fit\_transform(X)**  **y = sc\_y.fit\_transform(y)**  **from sklearn.svm import SVR**  **regressor = SVR(kernel = 'rbf')**  **regressor.fit(X, y)**  **from sklearn.metrics import r2\_score, mean\_squared\_error**  **Y\_model = sc\_y.inverse\_transform(regressor.predict(X).reshape(-1,1))**  **score = r2\_score(Y\_model, sc\_y.inverse\_transform(y))**  **print(f"score: {round(score\*100, 3)} %")**  **mse = mean\_squared\_error(y\_true=y, y\_pred=Y\_model)**  **print(f"mean squared error: {round(mse, 3)}")*## Predicting a new result***  **plt.scatter(sc\_X.inverse\_transform(X), sc\_y.inverse\_transform(y), color = 'red')**  **plt.plot(sc\_X.inverse\_transform(X), sc\_y.inverse\_transform(regressor.predict(X).reshape(-1, 1)), color = 'blue')**  **plt.title('Support vector regression')**  **plt.xlabel('years of experience')**  **plt.ylabel('Salary')**  **plt.show()**  **X\_grid = np.arange(min(sc\_X.inverse\_transform(X)), max(sc\_X.inverse\_transform(X)), 0.1)**  **X\_grid = X\_grid.reshape((len(X\_grid), 1))**  **plt.scatter(sc\_X.inverse\_transform(X), sc\_y.inverse\_transform(y), color = 'red')**  **plt.plot(X\_grid, sc\_y.inverse\_transform(regressor.predict(sc\_X.transform(X\_grid)).reshape(-1,1)), color = 'blue')**  **plt.title('Truth or Bluff (SVR)')**  **plt.xlabel('Position level')**  **plt.ylabel('Salary')**  **plt.show()** |

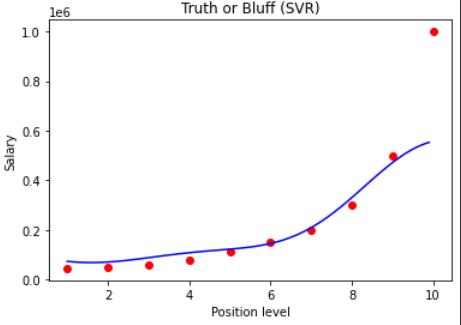
**OUTPUT**

1. SVM Classifier



1. SVM regression





**CONCLUSION**

Hence, we have studied and implemented SVM classifier and regression using scikit learn library in python.