# Experiment - 5

**Student Name:** Aniket Kumar
**UID:** 20BCS5306
**Branch:** CSE
**Section/Group:** 20BCS-WM-703/B
**Semester:** 5th
**Date of Performance:** 30/08/2022
**Subject Name:** ML LAB
**Subject Code:** 20-CSP-317

1. **Aim/Overview of the practical:** Implement Support Vector Machine on any data set and analyze the accuracy with Logistic regression

2. **Task to be done/ Which logistics used:**

- Import all the required library
- Import the dataset which you want to implement
- Split data into x and y and perform some task.
- Split data into training set and testing set
- Feature Scaling
- Predict The test set result
- Check the accuracy score by using different kernel
- Plot the train data
- Plot the test data
- Predicting the test set result
- Plot data points
- Create the hyperplane
- Plot the hyperplane

## 3. Code and Output:

[75]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
#Importing the datasets
df=pd.read_csv("Social_Network_Ads.csv")
df.head()
```

[75]:

| | User ID | Gender | Age | EstimatedSalary | Purchased |
|---|---|---|---|---|---|
| 0 | 15624510 | Male | 19 | 19000 | 0 |
| 1 | 15810944 | Male | 35 | 20000 | 0 |
| 2 | 15668575 | Female | 26 | 43000 | 0 |
| 3 | 15603246 | Female | 27 | 57000 | 0 |
| 4 | 15804002 | Male | 19 | 76000 | 0 |

[8]:
```
df.shape
```

[8]:
```
(400, 5)
```

[11]:
```
x=df.iloc[:,[2,3]]#Independent variable
y=df.iloc[:, 4]#Dependent variable
```

[12]:
```
x.head()
```

[12]:

| | Age | EstimatedSalary |
|---|---|---|
| **0** | 19 | 19000 |
| **1** | 35 | 20000 |
| **2** | 26 | 43000 |
| **3** | 27 | 57000 |
| **4** | 19 | 76000 |

[13]:
```
y.head()
```

[13]:
```
0    0
1    0
2    0
3    0
4    0
Name: Purchased, dtype: int64
```

[18]:
```
#Splitting the dataset into the Training set and Test set
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)
```

[19]:
```
print("Training data : ",X_Train.shape)
print("Training data : ",X_Test.shape)
```

```
Training data :  (300, 2)
Training data :  (100, 2)
```

[26]:
```
#Feature Scaling
from sklearn.preprocessing import StandardScaler
sc_X=StandardScaler()
X_Train=sc_X.fit_transform(X_Train)
X_Test=sc_X.transform(X_Test)
```

[31]:
```
from sklearn.svm import SVC
classifier =SVC(kernel = "linear", random_state = 0)
classifier.fit(X_Train, Y_Train)
#Predicting the test set results
Y_pred =classifier.predict(X_Test)
```

[32]:
```
Y_pred
```

[32]:
```
array([0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
       0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
       1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1], dtype=int64)
```

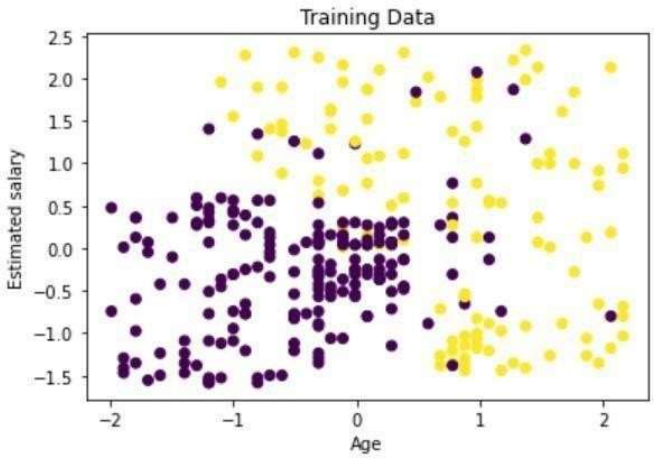
[35]:
```
from sklearn import metrics
print("Accuracy score: with linear kernel")
print(metrics.accuracy_score( Y_Test,Y_pred))
```

```
Accuracy score: with linear kernel
0.9
```

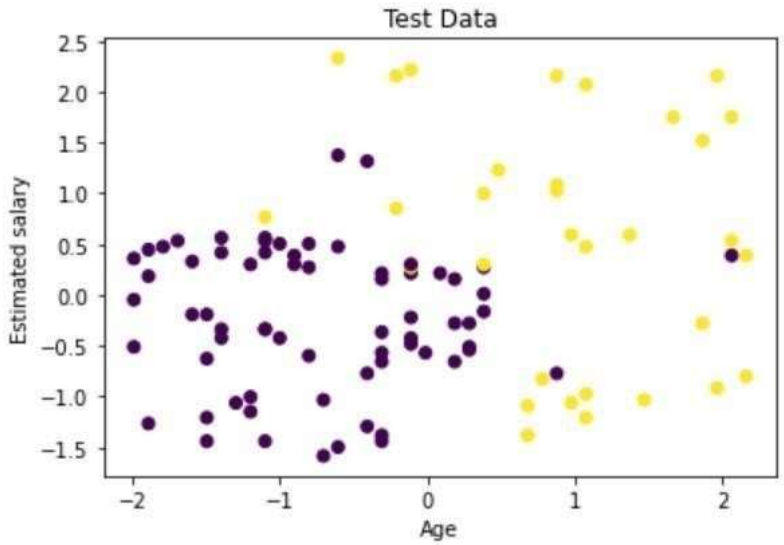
[44]:
```
from sklearn.svm import SVC
classifier =SVC(kernel = 'rbf')#radial basic function
classifier.fit (X_Train, Y_Train)
#Predicting the test set results
Y_pred = classifier.predict(X_Test)
print("Accuracy Score: with default rbf kernel")
print(metrics.accuracy_score(Y_Test,Y_pred))
```

```
Accuracy Score: with default rbf kernel
0.93
```

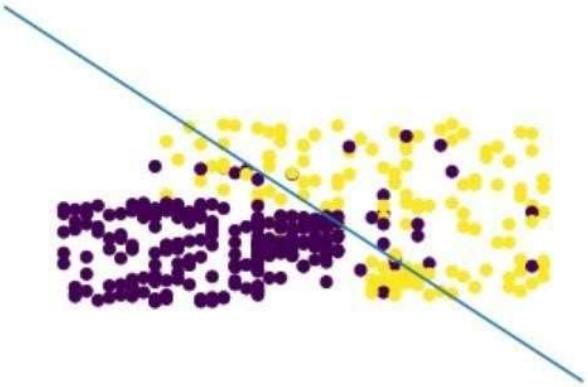
[58]:
```
import matplotlib.pyplot as plt
plt.scatter (X_Train [:, 0], X_Train[:, 1],c=Y_Train)
plt.xlabel('Age')
plt.ylabel('Estimated salary')
plt.title("Training Data")
plt.show()
```



[59]:
```
import matplotlib.pyplot as plt
plt.scatter (X_Test [:, 0], X_Test[:, 1],c=Y_Test)
plt.xlabel('Age')
plt.ylabel('Estimated salary')
plt.title("Test Data")
plt.show()
```




[76]:
```
from sklearn.svm import SVC
classifier= SVC(kernel='linear', random_state = 0)
classifier.fit(X_Train, Y_Train)
#Predicting the test set results
Y_pred=classifier.predict(X_Test)
#Plot data points
plt.scatter(X_Test[:, 0], X_Test[:, 1],c=Y_Test)
plt.scatter(X_Train[:, 0],X_Train[:, 1],c=Y_Train)
#Create the hyperplane
w = classifier.coef_[0]
a=-w[0]/w[1]
xx= np.linspace(-2.5, 2.5)
yy= a*xx -(classifier. intercept_[0])/w[1]
#Plot the hyperplane
plt.plot(xx, yy)
plt.axis("off"),plt.show();
```



## 4. Learning outcomes (What I have learnt):

1. Learn about SVM.
2. How to split dataset into train and test.
3. How to draw hyperplane

## 5. Evaluation Grid :

| s.no | Parameters | Marks Obtained | Maximum Marks |
|---|---|---|---|
| 1. | **Student Performance (Conduct of experiment) objectives/Outcomes.** | | **12** |
| 2. | **Viva Voce** | | **10** |
| 3. | **Submission of Work Sheet (Record)** | | **8** |
| | **Total** | | **30** |