PUNE INSTITUTE OF COMPUTER TECHNOLOGY

DEPARTMENT OF COMPUTER ENGINEERING

A MINIPROJECT REPORT ON

PREDICTING PURCHASE PATTERNS USING SVM CLASSIFIER

SUBMITTED BY

Prathamesh Sonawane 41166
Yuvraj Tambe 41169

IN PARTIAL FULFILLMENT

OF

BACHELOR OF ENGINEERING

COMPUTER ENGINEERING

UNIVERSITY OF PUNE

Abstract:

The main aim of our project to study buying patterns of people using Support Vector Machines for Classification.

The dataset we used consists of the following features: age, salary and records of whether the person buys an item or not. We have used these features to predict whether a person of certain age and with given income will purchase an item or not. We used a linear kernel for the purpose of classification.

Objective:

The purposes of this project are:
1. To learn how to do data preprocessing.
2. To apply SVM classification algorithm.

Introduction:

In classification, the idea is to predict the target class by analyzing the training dataset. This could be done by finding proper boundaries for each target class. In a general way of saying, use the training dataset to get better boundary conditions which could be used to determine each target class. Once the boundary conditions are determined, the next task is to predict the target class. The whole process is, therefore, known as classification.

In machine learning, support vector machines (SVMs, or support vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

Data:

This is our input dataset containing 3 columns and 400 rows and ‘purchased’ is the dependent variable.

	A	B	C	D	E	F
1	User ID	Gender	Age	EstimatedSalary	Purchased	
2	15624510	Male	19	19000	0	
3	15810944	Male	35	20000	0	
4	15668575	Female	26	43000	0	
5	15603246	Female	27	57000	0	
6	15804002	Male	19	76000	0	
7	15728773	Male	27	58000	0	
8	15598044	Female	27	84000	0	
9	15694829	Female	32	150000	1	
10	15600575	Male	25	33000	0	
11	15727311	Female	35	65000	0	
12	15570769	Female	26	80000	0	
13	15606274	Female	26	52000	0	
14	15746139	Male	20	86000	0	
15	15704987	Male	32	18000	0	

Fig. Dataset

Design/Implementation:

We first clean the data and than divide the dataset into training and testing sets. Then, we train the model and apply the classification model.

Source Code:

```
#Support Vector Machine (SVM)
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import cross_validate
from sklearn.model_selection import train_test_split

#Import the dataset
dataset = pd.read_csv('Social_Network_Ads.csv')
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, 4].values

# Split the dataset into the Training set and Test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)

#Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

#Fitting SVM to the Training set
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)

#Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)

#View the Training set results
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step
= 0.01),
```

```
                np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
         alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
            c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('SVM (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()

#Visualise the Test set results
from matplotlib.colors import ListedColormap
X_set, y_set = X_test, y_test
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step
= 0.01),
                np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
         alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
            c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('SVM (Test set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

Output:

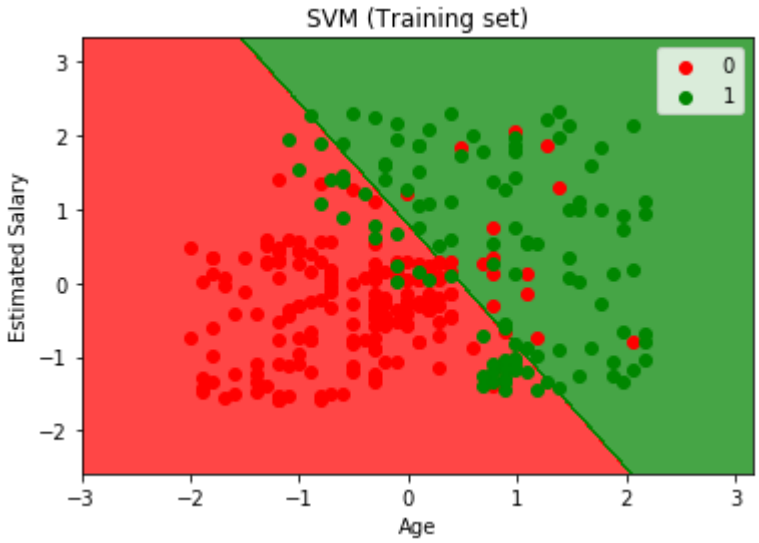

Fig: Training set

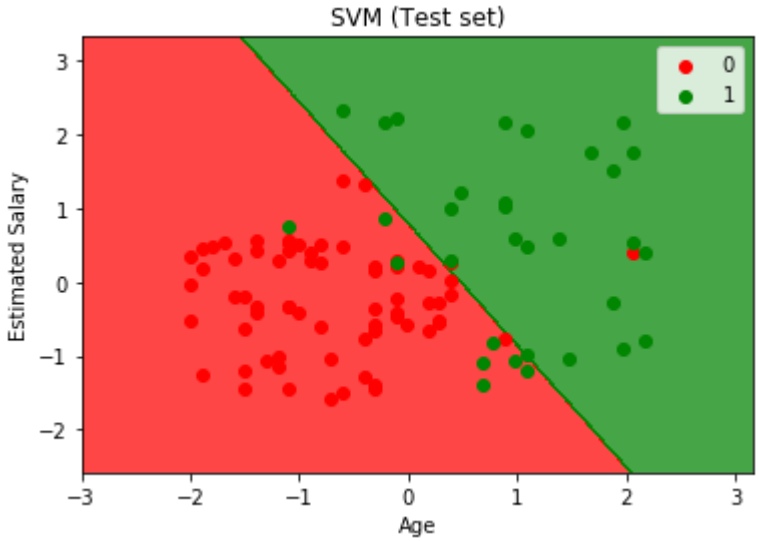

Fig: Testing set output

Conclusion and Future Enhancement:

To conclude, we have successfully implemented the project ‘Predicting Purchase Patterns using SVM classifier’.

For fututre work, more such classification algorithms can be studied and their accuracies can be compared with the this method.