Mini-Project Report

on

Comparing the Performance of Different Classification Algorithms using Social Media Ads Analysis

By

BECOB224	Mohit Nakhale
BECOB225	Rohit Nawale
BECOB236	Kaustubh Patil

Under the guidance of

Prof. Priya Surana

DEPARTMENT OF COMPUTER ENGINEERING,

PIMPRI CHINCHWAD COLLEGE OF ENGINEERING

SECTOR26, NIGDI, PRADHIKARAN

SAVITRIBAI PHULE PUNE UNIVERSITY

Academic Year: 2019-20

SEMESTER I

Report on Mini Project

Title: Comparing the Performance of Different Classification Algorithms using Social Media Ads Analysis.

Introduction: This mini project deals with studying the performance of different Classification Algorithms namely:

Random Forest Classifier
Support Vector Classifier
K-Nearest Neighbour Classifier

Requirements:

For the execution of Program following is required:

Python Interpreter
Sklearn library
Pandas
Social Media Ads Dataset

Theory:

Classification:

Classification is a data mining technique that assigns categories to a collection of data in order to aid in more accurate prediction. Classification is one of several methods intended to make the analysis of very large datasets effective. The goal is to create a set of classification rules that will answer a question, make a decision, or predict behaviour. To start, a set of training data is developed that contains a certain set of attributes as well as the likely outcome.

The job of the classification algorithm is to discover how that set of attributes reaches its conclusion.

It is a two-step process such as:

1. **Learning Step (Training Phase)**: Construction of Classification Model Different Algorithms are used to build a classifier by making the model learn using the training set available. Model has to be trained for prediction of accurate results.
2. **Classification Step**: Model used to predict class labels and testing the constructed model on test data and hence estimate the accuracy of the classification rules.

Different Classifiers used in Machine Learning:

1. Decision Trees
2. Naïve Bayes Classifiers
3. Neural Networks
4. K-Nearest Neighbour
5. Support Vector Machines
6. Random Forest
7. Linear Regression
8. Logistic Regression

Techniques Used:

Random Forest Classifier:

- Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.
- Random forest algorithm can use both for classification and the regression kind of problems.
- Applications :
  - Banking
  - Medical
  - Stock Market
  - E-commerce.

Advantages :

- More accurate solutions can be given as we are comparing data with several decision trees.
- By averaging several trees there is lower risk of overfitting.

Disadvantage :

- The main disadvantage of Random forests is their complexity. They are much harder and time-consuming to construct than decision trees.

Support Vector Machine Classifier :-

- SVM classifies a hyperplane into 2 different classes.
- It simply classifies data.

- Example :-

  Bunch of red and blue balls. If the balls aren't too mixed together, you could take a stick and without moving the balls, separate them with the stick.
  When a new ball is added on the table, by knowing which side of the stick the ball is on, you can predict its colour.
  The balls represent data points, and the red and blue colour represent 2 classes.
  The stick represents the simplest hyperplane which is a line.

Advantages :

- Easy to classify data.
- Perform well on large datasets.
- Works well with even unstructured and semi structured data like text, Images and trees.

Disadvantages :

- Long training time on large datasets.
- Difficult to understand and interpret the final model.

K-Nearest Neighbour Classifier:

- It is classification algorithm, decides result based on nearest neighbours.
- It assumes that similar things are near to each other.
- Calculates distance from nearest 'k' data points from the graph to classify unlabelled data.
- Classifies data based on nearest data points.
- It classifies data based on the probability we get from nearest data points.

Advantages :

- The algorithm is simple and easy to implement.
- The algorithm is versatile. It can be used for classification, regression, and search.

Disadvantage :

- The algorithm gets significantly slower as the number of examples and/or predictors/independent variables increase.

Applying Different Classifiers on Social Media Ads Dataset:

About Social Media Ads Dataset:

The Social Media Ads Dataset has the fields such as user id, gender, age, estimated salary and purchased. The Dataset expresses the various attributes which can be analyzed to estimate weather the user purchases the item seen in the advertisement.

We are classifying the user purchasing item and user who do not purchase item. We divide dataset into 75% training and 25% testing. Total number of rows in the dataset is 400.

We are getting accuracy of Random Forest as 91%.For Support Vector Machine, accuracy is 90% and for K-nearest neighbors, accuracy is 93%.

Source Code:

KNN.py

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
class knn:
    def kn(self):
        print("---KNN Classification Algorithm---")
        # Import the data
        dataset = pd.read_csv('Social_Network_Ads.csv')
        x = dataset.iloc[:, [2,3]].values
        y = dataset.iloc[:, 4].values
        # train-test split
        x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.25, random_state = 0)
        # Feature Scaling
        scalar = StandardScaler()
        x_train = scalar.fit_transform(x_train)
        x_test = scalar.transform(x_test)
        # Perform KNN
        knn = KNeighborsClassifier(n_neighbors = 5, p=2,metric='minkowski')
        knn.fit(x_train, y_train)
        y_pred = knn.predict(x_test)
        # Confusion Matrix
        cm = confusion_matrix(y_test, y_pred)
        print("confusion_matrix")
        print(cm)
        # Calculating the Accuracy
        print("accuracy")
        print(accuracy_score(y_test, y_pred))
        return  accuracy_score(y_test, y_pred)
```

RandomForest.py

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
class randomforest:
    def rf(self):
        print("---RandomForest Classification Algorithm---")
        # Importing the dataset
        dataset = pd.read_csv('Social_Network_Ads.csv')
        X = dataset.iloc[:, [2, 3]].values
        y = dataset.iloc[:, 4].values
        # Splitting 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
        sc = StandardScaler()
        X_train = sc.fit_transform(X_train)
        X_test = sc.transform(X_test)
        # Fitting Random Forest Classification to the Training set
        classifier = RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_state = 0)
        classifier.fit(X_train, y_train)
        # Predicting the Test set results
        y_pred = classifier.predict(X_test)
        # Making the Confusion Matrix
        cm = confusion_matrix(y_test, y_pred)
        print("confusion_matrix")
        print(cm)
        print("accuracy")
        print(accuracy_score(y_pred, y_test))
        return accuracy_score(y_test, y_pred)
```

SupportVector.py

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
class supportvector:
    def svc(self):
        print("---SupportVector Classification Algorithm---")
        dataset = pd.read_csv('Social_Network_Ads.csv')
        x = dataset.iloc[:, [2,3]]
        y = dataset.iloc[:, 4]
        x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.25, random_state = 0)
        sc = StandardScaler()
        x_train = sc.fit_transform(x_train)
        x_test = sc.transform(x_test)
        classifier = SVC(kernel='linear', random_state=0)
        classifier.fit(x_train, y_train)
        y_pred = classifier.predict(x_test)
        cm = confusion_matrix(y_test, y_pred)
        print(cm)
        print("accuracy")
        print(accuracy_score(y_pred, y_test))
        return accuracy_score(y_test, y_pred)
```

dem.py

```
from KNN import knn
from RandomForest import randomforest
from SupportVector import supportvector
knn=knn()
a=knn.kn()
rf=randomforest()
b=rf.rf()
sv=supportvector()
c=sv.svc()
print("Accuracy of KNN is : "+str(a))
print("Accuracy of RandomForest is : "+str(b))
print("Accuracy of SupportVector is : "+str(c))
```

DATASET:

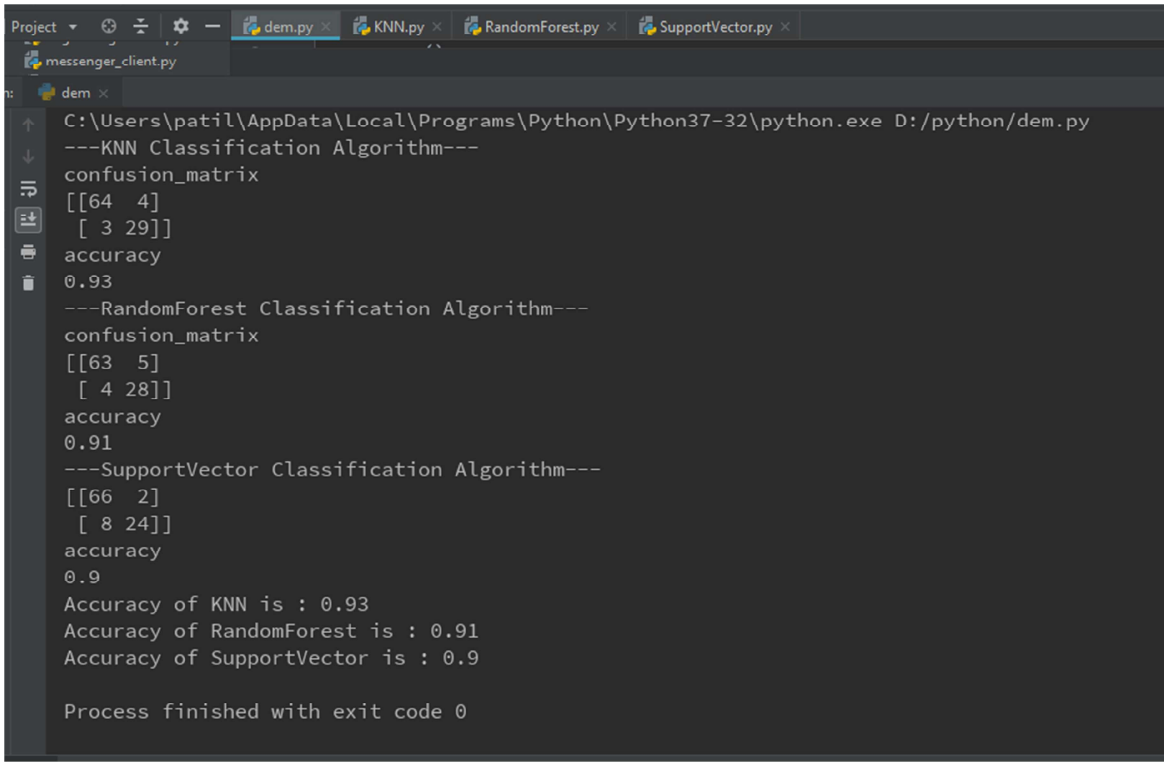
```
C:\Users\patil\AppData\Local\Programs\Python\Python37-32\python.exe D:/python/dem.py
            User ID         Age  EstimatedSalary   Purchased
count  4.000000e+02  400.000000       400.000000  400.000000
mean   1.569154e+07   37.655000     69742.500000    0.357500
std    7.165832e+04   10.482877     34096.960282    0.479864
min    1.556669e+07   18.000000     15000.000000    0.000000
25%    1.562676e+07   29.750000     43000.000000    0.000000
50%    1.569434e+07   37.000000     70000.000000    0.000000
75%    1.575036e+07   46.000000     88000.000000    1.000000
max    1.581524e+07   60.000000    150000.000000    1.000000

Process finished with exit code 0
```

```
dem
C:\Users\patil\AppData\Local\Programs\Python\Python37-32\python.exe D:/python/dem.py
    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
5  15728773    Male   27            58000          0
6  15598044  Female   27            84000          0
7  15694829  Female   32           150000          1
8  15600575    Male   25            33000          0
9  15727311  Female   35            65000          0

Process finished with exit code 0
```

OUTPUT:



CONCLUSION :-

Thus we studied how to apply different classifiers to classify data.
We find out the difference between accuracies of KNN, SVM and Random Forest algorithms on social media ads Dataset to predict ads purchase.
We compare these three algorithms with the help of social media ads dataset.