Contents

[Introduction 2](#_Toc58082212)

[Dataset 2](#_Toc58082213)

[Data pre-processing 2](#_Toc58082214)

[Feature Selection 2](#_Toc58082215)

[Model Preparation 3](#_Toc58082216)

[Logistic Regression 3](#_Toc58082217)

[K-Nearest Neighbours 3](#_Toc58082218)

[Support Vector Machine 4](#_Toc58082219)

[Kernel SVM 4](#_Toc58082220)

[Naïve Bayes 4](#_Toc58082221)

[Decision Tree Classification 4](#_Toc58082222)

[Random Forest Classification 5](#_Toc58082223)

[Conclusion 5](#_Toc58082224)

[Reference 6](#_Toc58082225)

## Introduction

Nowadays, social network advertising plays a significant role in the success of a business. Social network marketing or advertising allows businesses to perform targeted advertisings. Vendors try to predict whether a customer will buy the product or not based on previous purchase history. In this assignment, We picked a dataset with customers' purchase history and implemented various models to classify the customers and predict whether the user will buy a product or not.

## Dataset

In this assignment, we have used the ['Social-network-ads](https://www.kaggle.com/rakeshrau/social-network-ads)' dataset from Kaggle.

*adds = pd.read\_csv('Social\_Network\_Ads.csv')*

Data contains four hundred records and five features.

1. UserID – This holds each person's unique identity.
2. Gender – A person's gender. If the product is gender-based, then customers' gender should be considered.
3. Age - Age of the person. Purchase preferences depend on the age of the customer.
4. EstimatedSalary - The salary of a customer as it can also affect one's shopping.
5. Purchased – This is our target. It contains two values, '0' or '1'. '0' if not purchased, or '1' if purchased.

## Data pre-processing

This dataset has no missing values, so we do not have to perform any imputation.

## Feature Selection

For feature selection, we have used a heatmap from seaborn to visualize the correlation between features. The value of the correlation coefficients ranges from -1 to 1.

*corr\_matrix = adds.corr()*

*addscorr = corr\_matrix['Purchased'].sort\_values(ascending = False)*

*print(addscorr)*

|  |  |
| --- | --- |
| **Feature** | **Correlation** |
| Age | 0.622454 |
| EstimatedSalary | 0.362083 |
| Gender | 0.042469 |
| User ID | 0.007120 |

Based on the correlation matrix, we consider that 'Age' and 'EstimatedSalary' are correlated to 'Purchase.'

*x = adds.iloc[:, [2,3]].values*

*y = adds.iloc[:, -1].values*

*x\_train, x\_test, y\_train, y\_test = train\_test\_split(x,y,test\_size = 0.2)*

Considering the two highly correlated features, we had split the dataset into training and testing data, with an 80:20 ratio, using 'train\_test\_split' from sci-kit learn.

*sc = StandardScaler()*

*x\_train = sc.fit\_transform(x\_train)*

*x\_test = sc.transform(x\_test)*

We have scaled the features using 'StandardScaler' from sci-kit learn to perform regression algorithms.

## Model Preparation

We have created seven models with different algorithms, and their performance is measured using the confusion matrix, accuracy score and classification report.

### Logistic Regression

The predictant variable 'Purchased' in our dataset is binary, i.e. 0 & 1 and the Logistic Regression works better on binary classification. Hence, Logistic Regression is used to create a model.

*classifier = LogisticRegression()*

*classifier.fit(x\_train,y\_train)*

*y\_pred\_train = classifier.predict(x\_train)*

*y\_pred = classifier.predict(x\_test)*

*score\_tr = accuracy\_score(y\_train, y\_pred\_train)*

We got the accuracy of around 86 percent after performing K-fold cross validation.

### K-Nearest Neighbours

By analyzing the data points graphically, it is clear that people with a higher salary and more aged are more likely to purchase. As the data is in cluster form, we used K-Nearest Neighbours to create a model.

*knnclassifier = KNeighborsClassifier(n\_neighbors = 5, metric = 'minkowski', p = 2)*

*knnclassifier.fit(x\_train, y\_train)*

*y\_pred\_train = knnclassifier.predict(x\_train)*

*y\_pred = knnclassifier.predict(x\_test)*

*k\_score\_tr = accuracy\_score(y\_train, y\_pred\_train)*

We got the accuracy of around 89 percent after performing K-fold cross validation.

### Support Vector Machine

When we plotted, the field 'Age', 'EstimatedSalary' and 'Purchased' in the scatter plot. We noticed that the data points are aligned in two clusters, and a line can be drawn between the clusters to separate them. Hence we decided to use the Support Vector Machine to create a model.

*svmclassifier = SVC(kernel = 'linear', random\_state = 0)*

*svmclassifier.fit(x\_train, y\_train)*

*y\_pred\_train = svmclassifier.predict(x\_train)*

*y\_pred = svmclassifier.predict(x\_test)*

*svm\_score\_tr = accuracy\_score(y\_train, y\_pred\_train)*

We got the accuracy of around 90 percent after performing K-fold cross validation.

### Kernel SVM

As few data points in the two clusters overlap with each other, and a linear hyperplane of SVM cannot separate the data efficiently. So we decided to add a dimension to SVM by using a kernel.

*ksvmclassifier = SVC(kernel = 'rbf', random\_state = 0)*

*ksvmclassifier.fit(x\_train, y\_train)*

*y\_pred\_train = ksvmclassifier.predict(x\_train)*

*y\_pred = ksvmclassifier.predict(x\_test)*

*ksvm\_score\_tr = accuracy\_score(y\_train, y\_pred\_train)*

We got the accuracy of around 90 percent after performing K-fold cross validation.

### Naïve Bayes

From the scatter plot, we can observe that the data points with vale 0 are closely grouped in circle form, and the data point with value 1 surrounds it. It is clear that we can draw a Gaussian curve to separate the cluster. Hence we choose Gaussian Naïve Bayes for our model.

*nbclassifier = GaussianNB()*

*nbclassifier.fit(x\_train, y\_train)*

*y\_pred\_train = nbclassifier.predict(x\_train)*

*y\_pred = nbclassifier.predict(x\_test)*

*nb\_score\_tr = accuracy\_score(y\_train, y\_pred\_train)*

We got the accuracy of around 93 percent after performing K-fold cross validation.

### Decision Tree Classification

Eventhough accuracy is higher for random forest model, as far as the comprehensibility of the model is taken into account, we planned to go for decision tree algorithm in the next iteration.

*dtclassifier = DecisionTreeClassifier(criterion = 'entropy', random\_state = 0)*

*dtclassifier.fit(x\_train, y\_train)*

*y\_pred\_train = dtclassifier.predict(x\_train)*

*y\_pred = dtclassifier.predict(x\_test)*

*dt\_score\_tr = accuracy\_score(y\_train, y\_pred\_train)*

We got the accuracy of around 86 percent after performing K-fold cross validation.

### Random Forest Classification

As the K-fold validation accuracy score of decision tree model is only 86 percent, we tried to do the classification with random forest classifier.

*rfclassifier = RandomForestClassifier(n\_estimators = 10, criterion = 'entropy', random\_state = 0)*

*rfclassifier.fit(x\_train, y\_train)*

*y\_pred\_train = rfclassifier.predict(x\_train)*

*y\_pred = rfclassifier.predict(x\_test)*

*rfc\_score\_tr = accuracy\_score(y\_train, y\_pred\_train)*

We got the accuracy of around 90 percent after performing K-fold cross validation.

## Conclusion

We have developed 7 models to predict the online purchasing trend of individuals through different social network advertisements. The performance measurement comparison is provided in the below table.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ***Classification Algorithm*** | ***class*** | ***Accuracy*** | | ***Precision*** | ***Recall*** | ***F1-score*** |
| ***Normal method*** | ***K-fold cross validation*** |
| Logistic Regression | 0 | 0.88 | 0.86 | 0.89 | 0.92 | 0.9 |
| 1 | 0.85 | 0.79 | 0.82 |
| K-Nearest Neighbours | 0 | 0.91 | 0.89 | 0.94 | 0.92 | 0.93 |
| 1 | 0.87 | 0.9 | 0.88 |
| Support Vector Machine | 0 | 0.85 | 0.90 | 0.85 | 0.92 | 0.89 |
| 1 | 0.84 | 0.72 | 0.78 |
| Kernel SVM | 0 | 0.94 | 0.90 | 0.96 | 0.94 | 0.95 |
| 1 | 0.9 | 0.93 | 0.92 |
| Naïve Bayes | 0 | 0.90 | 0.93 | 0.92 | 0.92 | 0.92 |
| 1 | 0.86 | 0.86 | 0.86 |
| Decision Tree classifier | 0 | 0.88 | 0.86 | 0.89 | 0.92 | 0.9 |
| 1 | 0.85 | 0.79 | 0.82 |
| Random Forest classifier | 0 | 0.91 | 0.90 | 0.92 | 0.94 | 0.93 |
| 1 | 0.89 | 0.86 | 0.88 |

Thus, it is clear from the table that accuracy score is higher for Naïve Bayes and Kernel SVM models and both of them can be considered as the best suited algorithm for this dataset. K-fold cross validation technique was introduced in the model so as to enhance the accuracy of the model and to reduce the overfitting problem.

## Reference

* [https://www.slideshare.net/CharlesVestur/building-a-performing-machine-learning-model-from-a-to-z](https://www.slideshare.net/CharlesVestur/building-a-performing-machine-learning-model-from-a-to-z%20)
* <https://www.datacamp.com/community/tutorials/decision-tree-classification-python>
* <https://developer.ibm.com/technologies/data-science/tutorials/learn-classification-algorithms-using-python-and-scikit-learn/>