FLIP ROBO

Customer Rating Classifier Model

Submitted by:

NIHAL SINGH

ACKNOWLEDGMENT

Customer Rating Classifier Model Project includes the details about the review of customers for different products like Mobile, Smart watch, laptops etc. on different shopping websites. This Comment text are expression of people in from of words. In that, some of comments and rating are good for the product and some are bad ratings.

Customer Rating Classifier Model is a Natural language processing Model, which is used to predict the type of rating and reviews given by Customer after purchasing the product on e-commerce websites. The selenium web scrapping is used to scrap both the data i.e. review and ratings of the customer from e-commerce websites on different products.

INTRODUCTION

- Business Problem Framing

  The Customer Rating Classifier model is related to e-commerce websites where customers gives feedback on different products.it is basically related to consumers to express their opinions widely on product in their comments and reviews. These comments consist of good rating and bad rating of products from 1 to 5.

- Conceptual Background of the Domain Problem

  The model is related to the E-commerce domain and consist of comments and ratings of customers to express their opinions widely online on e-commerce websites products.

  The model is basically a natural language processing model where model is build on based on understanding reviews.

- Motivation for the Problem Undertaken

  The Customer Rating Classifier Model is consist of only text comments which has been encoded in to binary format using multiple method like bag of words, TF/IDF, Stopwords in Natural language processing Model building. These methods used to filter the comments and then using steaming or lemmatizing to convert this words into binary format.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

  The Dataset basically is in csv , which consist is scrapped from e-commerce web sites. Analysing the data, there data type, which columns is impotent. Plotting the multiple plots like bar plot,heat map to analyse the data and which features are important. After that using multiple methods to filter the review comments and then using lemmanizing to covert word into vectors.

- Data Sources and their formats

  Data is in the CSV format.the data has been imported into the data frame using the python library.

```
#Loading the dataset
df=pd.read_csv('Rating1.csv',index_col=False)
df.head()
```

	Review	Rating
0	I am writing this review after using this mobi...	5
1	This product was absolutely good and it is ver...	5
2	Everything is better of the phone except of fr...	5
3	It's a value for money product.. But if you r ...	4
4	This phone is pretty decentBattery backup and ...	4

```
#checking the shape of dataframe
df.shape
```

```
(24113, 2)
```

The data having the different datatypes i.e. int, and object.

```
#pd.set_option('display.max_rows',None)
```

```
# check information of dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24113 entries, 0 to 24112
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   Review  24113 non-null  object
 1   Rating  24113 non-null  int64 
dtypes: int64(1), object(1)
memory usage: 376.9+ KB
```

```
#checking the null values in dataset
df.isnull().sum()
```

```
Review    0
Rating    0
dtype: int64
```

```
#there is no null values available in the  dataset.
```

- Data Preprocessing Done

  The Data Preprocessing consist of below mentioned points:

  1. Fetch the data frame from the scraped data for the Data pre-processing.
  2. Checking the null values form the dataset, if null value is available them replace them with the suitable values.
  3. Describing the data.
  4. The train Data basically consist of only review as feature which having only text data in form of people comments.
  5. Using multiple methods to filter the comments. Using stopwords, lower then lemmanizing to filter the words.
  6. After that using TF-IDF method to convert words into the vectors i.e basically in the binary format to build the model.
  7. This operation has been done on both train and test data separately.
  8. Plotting the multiple plots like bar plot,heat map to check the data and type of comments used for different words.
  9. There are multiple targets so combing all the target as one label and using that label to build the model.
  10. This is basically a NLP model so there is no need to check skewness and outliers.
  11. Separating the features and target to build the model.
  12. Checking the data imbalanced issue on the target variable using value counts.
  13. Here imbalanced issue is present and using upsampling SMOTE to solve the data imbalanced issue.

- Data Inputs- Logic- Output Relationships

  There are only one feature is available i.e review which consist of comments in text format and multiple labels which has been converted to one target. So using the comment doing the prediction type on comments.

- Hardware and Software Requirements and Tools Used

  The Tool used to build the model is anaconda jupyter.

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  Multiple problem faced during the model building. All are listed bellows.

  1. Scrapping the data from the websites for model building.
  2. Checking for the unwanted columns analyse the relation with target and removal of that features.
  3. Checking the null values and replace it.
  4. Filtering the text using the stopwords and lemmanization.
  5. Using TP-ICP to convert cleaned word data into vector data.
  6. Checking of Data imbalanced issue on the target and then solve the data imbalnced suing upsampling SMOTE.

- Testing of Identified Approaches (Algorithms)

  This is a Natural language processing classification problem so multiple classification model has been used for the model building and basically training and testing the data on multiple models to select the best model.

  The multiple models are used as given below:

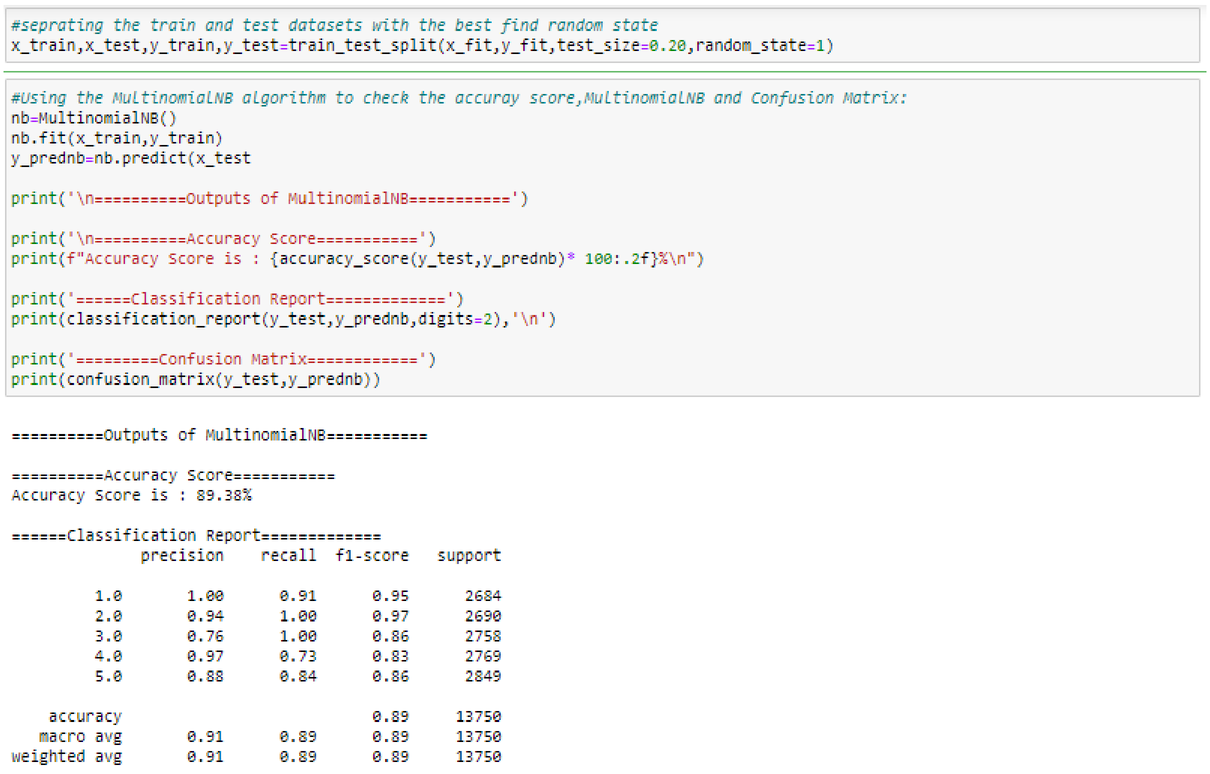
  1. MultinomialNB
  2. Random forest classifier
  3. Logistics Regression
  4. Decision Tree classifier
  5. Ada boost classifier
  6. KNN Classifiers

- Run and Evaluate selected models

  The model is a Natural language processing classification model so all the classification model building algorithm has been used to build the model and across all the model the best one has been selected as final model.

  The different models which has been used has been listed below:

  1. MultinomialNB Classifier: The multinomial Naive Bayes classifier is suitable for classification with discrete features (e.g., word counts for text classification). The multinomial distribution normally requires integer feature counts.



```
#seprating the train and test datasets with the best find random state
x_train,x_test,y_train,y_test=train_test_split(x_fit,y_fit,test_size=0.20,random_state=1)
```

```
#Using the MultinomialNB algorithm to check the accuray score,MultinomialNB and Confusion Matrix:
nb=MultinomialNB()
nb.fit(x_train,y_train)
y_prednb=nb.predict(x_test

print('\n==========Outputs of MultinomialNB===========')

print('\n==========Accuracy Score===========')
print(f"Accuracy Score is : {accuracy_score(y_test,y_prednb)* 100:.2f}%\n")

print('======Classification Report=============')
print(classification_report(y_test,y_prednb,digits=2),'\n')

print('=========Confusion Matrix============')
print(confusion_matrix(y_test,y_prednb))
```

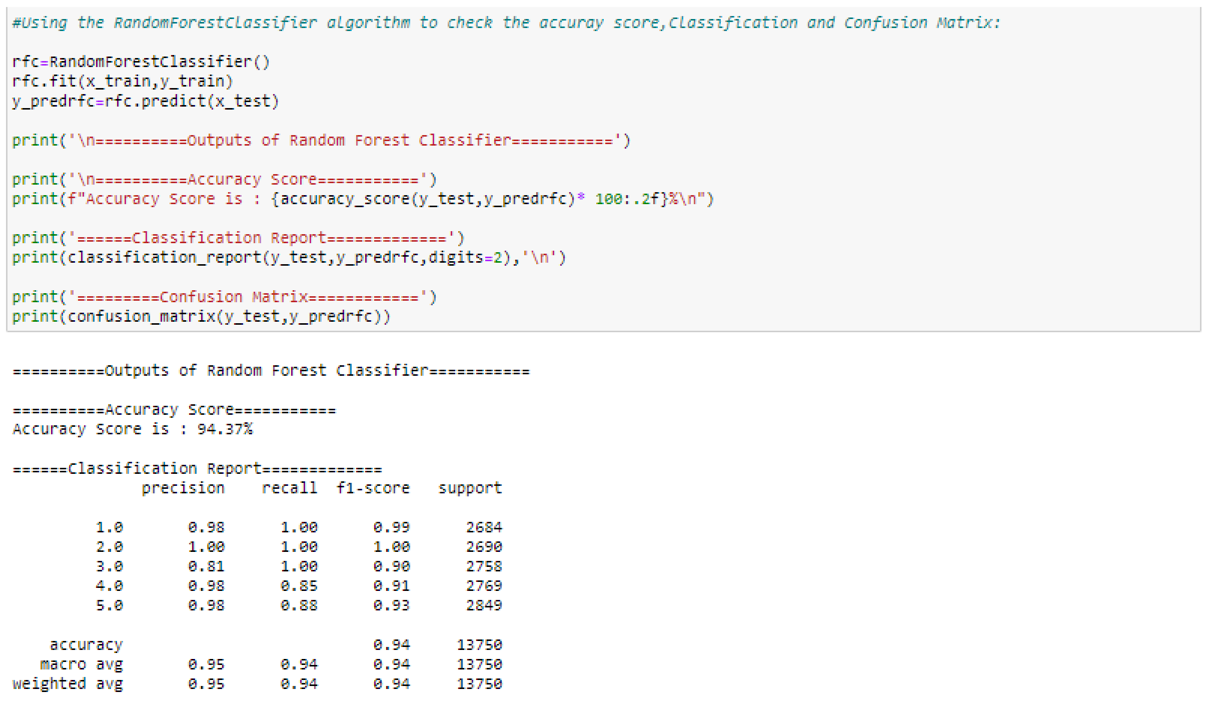
```
==========Outputs of MultinomialNB===========

==========Accuracy Score===========
Accuracy Score is : 89.38%

======Classification Report=============
              precision    recall  f1-score   support

         1.0       1.00      0.91      0.95      2684
         2.0       0.94      1.00      0.97      2690
         3.0       0.76      1.00      0.86      2758
         4.0       0.97      0.73      0.83      2769
         5.0       0.88      0.84      0.86      2849

    accuracy                           0.89     13750
   macro avg       0.91      0.89      0.89     13750
weighted avg       0.91      0.89      0.89     13750
```

  After training the data in MultinomialNB the accuracy score of the model is 89.38%.

  The Classification Report consist of all the details related to accuracy score, precision, recall and f1- score of MultinomialNB model.

It also consist of Confusion matrix values i.e True Positive, False Positive, True Negative and False Negative values MultinomialNB model.

2. Random forest Classifier: Random forests is a supervised learning algorithm. Random forests creates decision trees on randomly selected data samples, gets prediction from each tree and selects the best solution by means of voting.



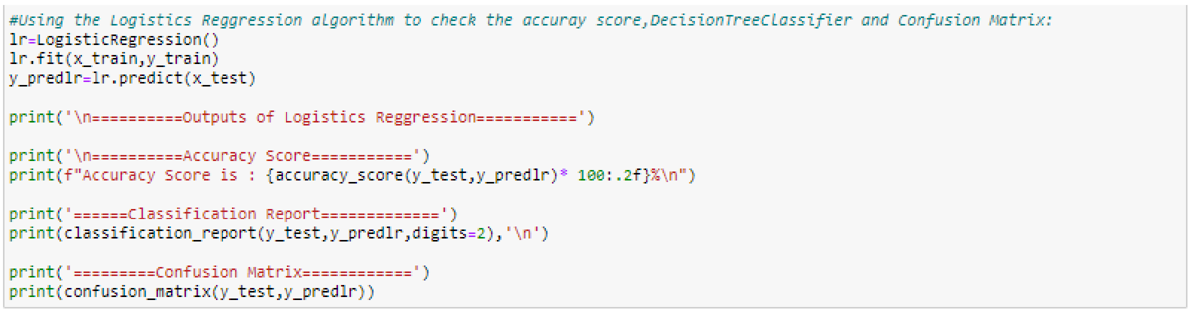
```
#Using the RandomForestClassifier algorithm to check the accuray score,Classification and Confusion Matrix:

rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
y_predrfc=rfc.predict(x_test)

print('\n==========Outputs of Random Forest Classifier===========')

print('\n==========Accuracy Score===========')
print(f"Accuracy Score is : {accuracy_score(y_test,y_predrfc)* 100:.2f}%\n")

print('======Classification Report=============')
print(classification_report(y_test,y_predrfc,digits=2),'\n')

print('=========Confusion Matrix============')
print(confusion_matrix(y_test,y_predrfc))
```

```
==========Outputs of Random Forest Classifier===========

==========Accuracy Score===========
Accuracy Score is : 94.37%

======Classification Report=============
              precision    recall  f1-score   support

         1.0       0.98      1.00      0.99      2684
         2.0       1.00      1.00      1.00      2690
         3.0       0.81      1.00      0.90      2758
         4.0       0.98      0.85      0.91      2769
         5.0       0.98      0.88      0.93      2849

    accuracy                           0.94     13750
   macro avg       0.95      0.94      0.94     13750
weighted avg       0.95      0.94      0.94     13750
```

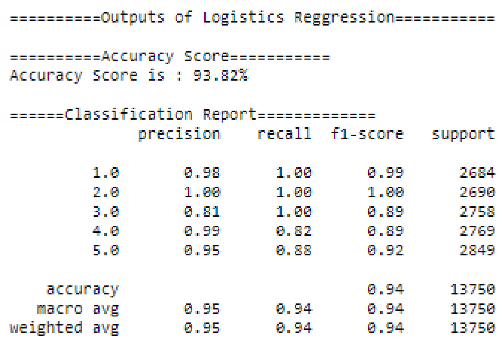
After training, the data in Random forest Classifier the accuracy score of the model is 94.37%.
The Classification Report consist of all the details related to accuracy score, precision, recall and f1- score of Random forest Classifier model.
It also consist of Confusion matrix values i.e True Positive, False Positive, True Negative and False Negative values of Random forest Classifier model.

The different models which has been used has been listed below:

3. Logistics Regression : Logistic Regression is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes or No, Pass or fail etc) or 0.



```
#Using the Logistics Reggression algorithm to check the accuray score,DecisionTreeClassifier and Confusion Matrix:
lr=LogisticRegression()
lr.fit(x_train,y_train)
y_predlr=lr.predict(x_test)

print('\n==========Outputs of Logistics Reggression===========')

print('\n==========Accuracy Score===========')
print(f"Accuracy Score is : {accuracy_score(y_test,y_predlr)* 100:.2f}%\n")

print('======Classification Report=============')
print(classification_report(y_test,y_predlr,digits=2),'\n')

print('=========Confusion Matrix============')
print(confusion_matrix(y_test,y_predlr))
```



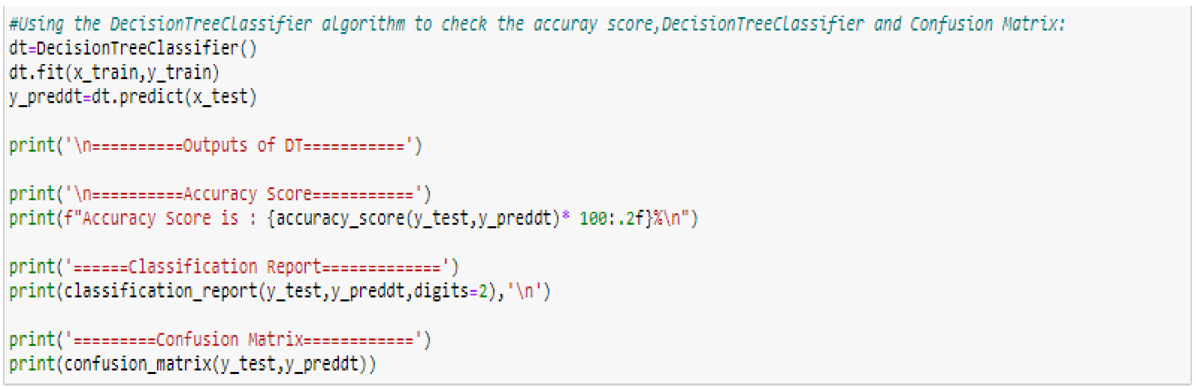
```
==========Outputs of Logistics Reggression===========

==========Accuracy Score===========
Accuracy Score is : 93.82%

======Classification Report=============
              precision    recall  f1-score   support

         1.0       0.98      1.00      0.99      2684
         2.0       1.00      1.00      1.00      2690
         3.0       0.81      1.00      0.89      2758
         4.0       0.99      0.82      0.89      2769
         5.0       0.95      0.88      0.92      2849

    accuracy                           0.94     13750
   macro avg       0.95      0.94      0.94     13750
weighted avg       0.95      0.94      0.94     13750
```

After training the data in Logistic Regression the accuracy score of the model is 93.42%.

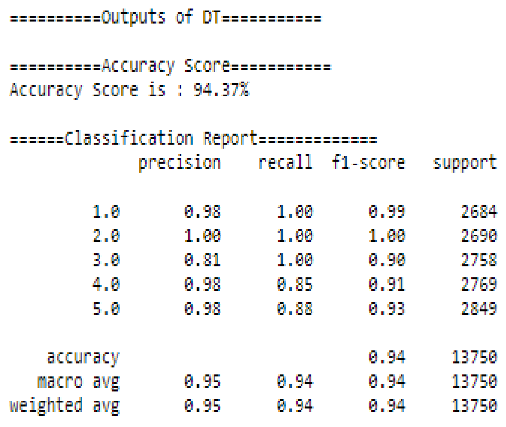
The Classification Report consist of all the details related to accuracy score, precision, recall and f1- score of Logistics regression model.

It also consist of Confusion matrix values i.e True Positive, False Positive, True Negative and False Negative values of logistic regression model.

4. Decision Tree Classifier: Decision Trees are a non-parametric supervised learning method used for classification and regression. Decision tree basically consist of root node, branch and leaf node concept.



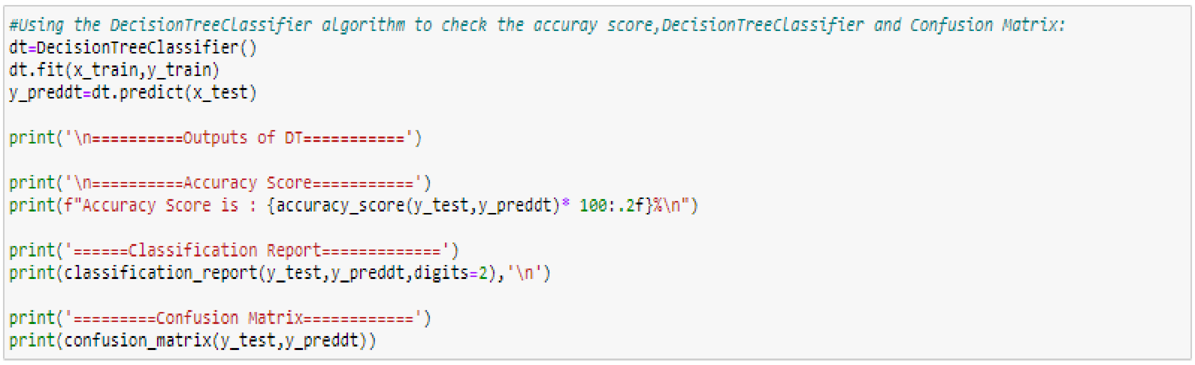
```
#Using the DecisionTreeClassifier algorithm to check the accuray score,DecisionTreeClassifier and Confusion Matrix:
dt=DecisionTreeClassifier()
dt.fit(x_train,y_train)
y_preddt=dt.predict(x_test)

print('\n==========Outputs of DT===========')

print('\n==========Accuracy Score===========')
print(f"Accuracy Score is : {accuracy_score(y_test,y_preddt)* 100:.2f}%\n")

print('======Classification Report=============')
print(classification_report(y_test,y_preddt,digits=2),'\n')

print('=========Confusion Matrix===========')
print(confusion_matrix(y_test,y_preddt))
```



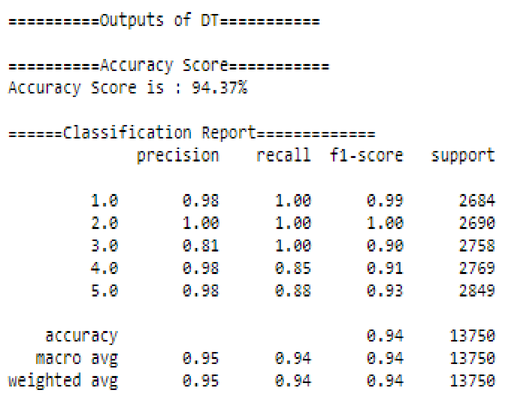
```
==========Outputs of DT===========

==========Accuracy Score===========
Accuracy Score is : 94.37%

======Classification Report=============
              precision    recall  f1-score   support

         1.0       0.98      1.00      0.99      2684
         2.0       1.00      1.00      1.00      2690
         3.0       0.81      1.00      0.90      2758
         4.0       0.98      0.85      0.91      2769
         5.0       0.98      0.88      0.93      2849

    accuracy                           0.94     13750
   macro avg       0.95      0.94      0.94     13750
weighted avg       0.95      0.94      0.94     13750
```

After training the data in Decision Tree Classifier the accuracy score of the model is 94.37%.
The Classification Report consist of all the details related to accuracy score, precision, recall and f1- score of Decision Tree Classifier model.
It also consist of Confusion matrix values i.e True Positive, False Positive, True Negative and False Negative values of Decision Tree Classifier model.

5. Ada boost Classifier: AdaBoost Classifier, short for Adaptive Boosting, is a Boosting technique used as an Ensemble Method in Machine Learning. It combines multiple classifiers to increase the accuracy of classifiers. AdaBoost is an iterative ensemble method.



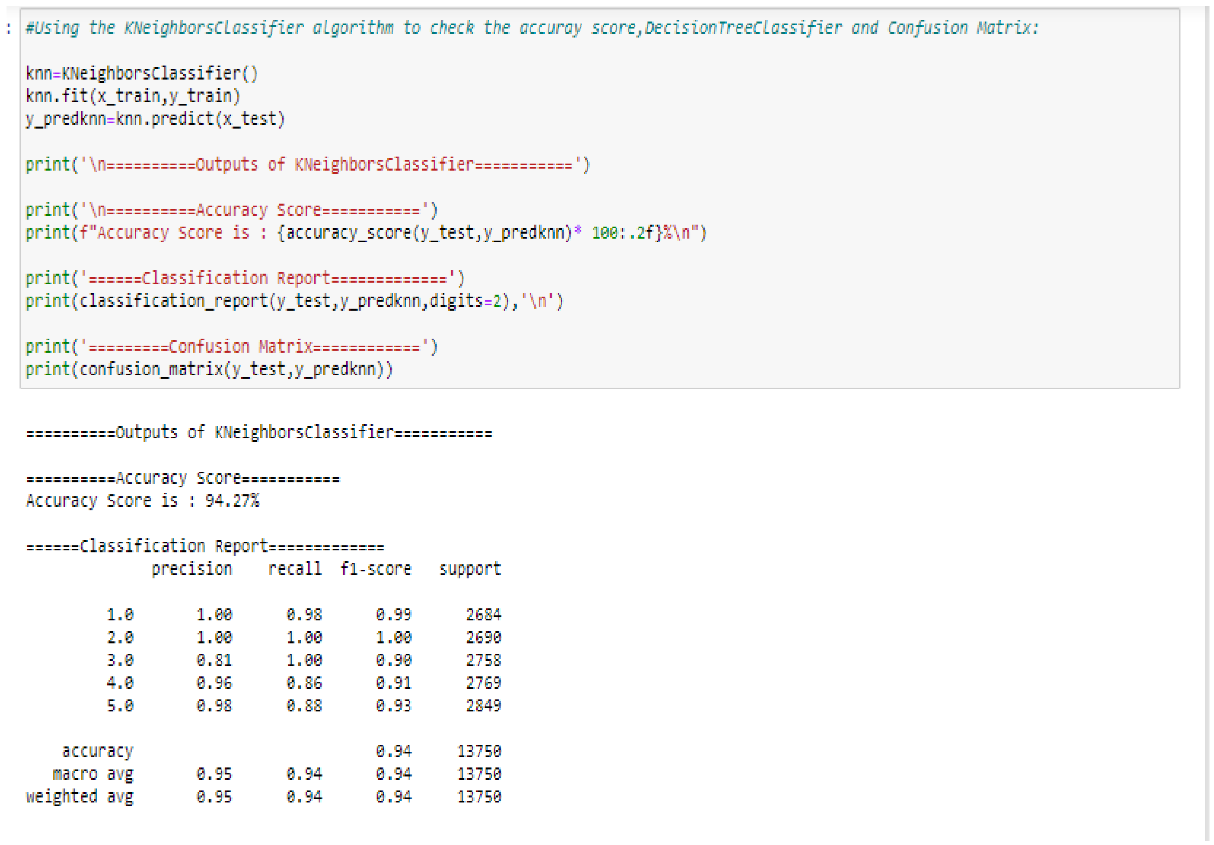
```
#Using the DecisionTreeClassifier algorithm to check the accuray score,DecisionTreeClassifier and Confusion Matrix:
dt=DecisionTreeClassifier()
dt.fit(x_train,y_train)
y_preddt=dt.predict(x_test)

print('\n==========Outputs of DT===========')

print('\n==========Accuracy Score===========')
print(f"Accuracy Score is : {accuracy_score(y_test,y_preddt)* 100:.2f}%\n")

print('======Classification Report=============')
print(classification_report(y_test,y_preddt,digits=2),'\n')

print('=========Confusion Matrix============')
print(confusion_matrix(y_test,y_preddt))
```



```
==========Outputs of DT===========

==========Accuracy Score===========
Accuracy Score is : 94.37%

======Classification Report=============
              precision    recall  f1-score   support

         1.0       0.98      1.00      0.99      2684
         2.0       1.00      1.00      1.00      2690
         3.0       0.81      1.00      0.90      2758
         4.0       0.98      0.85      0.91      2769
         5.0       0.98      0.88      0.93      2849

    accuracy                           0.94     13750
   macro avg       0.95      0.94      0.94     13750
weighted avg       0.95      0.94      0.94     13750
```

After training the data in Ada boost Classifier the accuracy score of the model is 94.37%.
The Classification Report consist of all the details related to accuracy score, precision, recall and f1- score of Ada boost Classifier model.
It also consist of Confusion matrix values i.e True Positive, False Positive, True Negative and False Negative values of Ada boost Classifier model.

6. KNN Classifier: KNN also known as K-nearest neighbour is a supervised and pattern classification learning algorithm which helps us find which class the new input(test value) belongs to when k nearest neighbours are chosen and distance is calculated between them.



```
#Using the KNeighborsClassifier algorithm to check the accuray score,DecisionTreeClassifier and Confusion Matrix:

knn=KNeighborsClassifier()
knn.fit(x_train,y_train)
y_predknn=knn.predict(x_test)

print('\n==========Outputs of KNeighborsClassifier===========')

print('\n==========Accuracy Score===========')
print(f"Accuracy Score is : {accuracy_score(y_test,y_predknn)* 100:.2f}%\n")

print('======Classification Report=============')
print(classification_report(y_test,y_predknn,digits=2),'\n')

print('=========Confusion Matrix============')
print(confusion_matrix(y_test,y_predknn))
```

```
==========Outputs of KNeighborsClassifier===========

==========Accuracy Score===========
Accuracy Score is : 94.27%

======Classification Report=============
              precision    recall  f1-score   support

         1.0       1.00      0.98      0.99      2684
         2.0       1.00      1.00      1.00      2690
         3.0       0.81      1.00      0.90      2758
         4.0       0.96      0.86      0.91      2769
         5.0       0.98      0.88      0.93      2849

    accuracy                           0.94     13750
   macro avg       0.95      0.94      0.94     13750
weighted avg       0.95      0.94      0.94     13750
```

After training, the data in KNN Classifier the accuracy score of the model is 94.27%.
The Classification Report consist of all the details related to accuracy score, precision, recall and f1- score of KNN Classifier model.
It also consist of Confusion matrix values i.e True Positive, False Positive, True Negative and False Negative values of KNN Classifier model.

- Key Metrics for success in solving problem under consideration

  There are multiple points which impact the final outcome for the model.

  1. Scrapping the data from web sites.
  2. Converting the text comments into vectors using multiple methods.
  3. Combining multiple targets into single label and then build the NLP model.
  4. Performing the data pre-processing on the both train and test data separately.
  5. Checking the data imbalanced issue and solve it using upsampling SMOTE method.

- Visualizations

  Visualization is basically finding some outcomes after visualizing the data in form of some graph or some plots. Different visualizing methods has been use the to do the analysis on the data. In this model multiple plots has been used to do analysis on the provided data.

  1. Bar Plot – To check the relation between feature and target.
  2. Heat Map – To check the null count and multicolliniarity issue between the features.

- Interpretation of the Results

  From visualization, pre-processing and modelling the result should indicates label can be 1,2,3,4,5, where these value denotes the rating of the customers.

CONCLUSION

- Key Findings and Conclusions of the Study

  Main finding in the models are using multiple NLP methods to covert the word features to the vectors and using these vectors features building the NLP model.

- Learning Outcomes of the Study in respect of Data Science

  There are only one feature column which is consist of the text data.so the main learning is to analyse this data and cleaning the data using multiple method like stopwords, lowering the text, lemmanizing. Using TF-IDF method to convert cleaned data into vectors and using this vectors building the model. From the visualization the data imbalanced issue has been solved form the target columns using over sampling method called as SMOTE.