Rating Prediction Project

Submitted By:- Sujay Nimbalkar

Submitted To:- Swati Mahaseth (Flip Robo Technologies)

Acknowldgement

Here we have scrapped the Data from “Flipkart com” in order to scrape the reviews of the various products which were necessary for the project phase which helped us and gave us the idea to make a dataset and finally we can work on that dataset to build a model and prepare the solution for the given problem statement

Introduction:-

We have a client who has a website where people write different reviews for technical products. Now they are adding a new feature to their website i.e. The reviewer will have to add stars(rating) as well with the review. The rating is out 5 stars and it only has 5 options available 1 star, 2 stars, 3 stars, 4 stars, 5 stars. Now they want to predict ratings for the reviews which were written in the past and they don’t have a rating. So, we have to build an application which can predict the rating by seeing the review

Conceptional Backgroud:

Now a days Ecommerce platforms are the most convenient way for the people to get the things they want hence there is a huge competation between the various Ecommerce platforms among themselves People buy they want and rate the products as well which helps the buyer to get the information of the products in brief looking at the rating of the product.

Review of Litrature:

This project is all about the prediction of the rating given by the customers on the Ecommerce platform which will give a detailed idea The data Is been scapped from one of the Ecommerce platform to study and the reviews have been collected of various usefull products

Motivation of the Problem Undertaken

Here we have taken initiative at our fullest to solve the problem. We have scrapped the data from Flipkart com of the various useful products as you can see in the dataset along with the reviews of the customer which is gonna help us We are going to use this data set along with the machine learning algorithms with various parameters to get an best accuracy score so we are going to see various algorithms in this project applied which are going to give us the best accuracy score

Steps followed for model Building Phase are

1. Scrapping the data

ut[36]:

	Rating	Review_head	Full_review
0	2	Could be way better	Price is high.
1	5	Brilliant	Superb quality nice panels good working safe l...
2	5	Simply awesome	Fine product
3	4	Worth the money	Almost good product
4	5	Wonderful	Nice product...
...	...	...	...
115688	5	Highly recommended	Super for proficition photo grapher
115689	3	Just okay	in the description it was mentioned as as alum...
115690	5	Worth every penny	nice products
115691	5	Terrific purchase	this is the best stepilizer for low angle shot...
115692	5	Great product	Very Good Product....Fast Shipping....I hope D...

115693 rows × 3 columns

As we can see that the data have been scrapped from Flipkart along with the customer reviews.

After deleting the data which is not required here finally we have got the new dataset with us

87]:

	Rating	Review_head	Full_review
17	1	did not meet expectations	poor bas
32	1	very poor	bluetoo bad
41	1	worthless	poor qual
45	1	useless product	purchas produc
48	1	worthless	nyc sound bass smal ok pric
...	...	...	...
7649	5	wonderful	model
7650	5	wonderful	sup produc ❤
7655	5	terrific	🤩this nic produc lov n s work prop n smoo mus...
7656	5	highly recommended	superb also look awesom charg ind nt show mac...
7657	5	mindblowing purchase	' nic purchas lik year warranty get produc ...

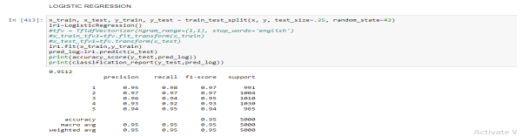
20000 rows × 3 columns

Further we imported necessary Libraries required

```
|: from sklearn.model_selection import train_test_split
   from sklearn.metrics import accuracy_score,classification_report
   from sklearn.linear_model import LogisticRegression
```

Further we went for Logistic Regression,

We got the score of 0.9512 as you will get to see in the below picture



Support Vector Machine:-

With Support Vector Machine there is no major difference in the score as you will able to see in the below picture

SUPPORT VECTOR MACHINE(LinearSVC)

```
In [414]: from sklearn.svm import LinearSVC
          svm=LinearSVC(class_weight='balanced')
          svm.fit(x_train,y_train)
          pred_svm=svm.predict(x_test)
          print(accuracy_score(y_test,pred_svm))
          print(classification_report(y_test,pred_svm))
```

```
0.9516
              precision    recall  f1-score   support

           1       0.95      0.98      0.97       991
           2       0.97      0.98      0.97      1004
           3       0.96      0.94      0.95      1010
           4       0.93      0.92      0.92      1030
           5       0.94      0.95      0.94       965

    accuracy                           0.95      5000
   macro avg       0.95      0.95      0.95      5000
weighted avg       0.95      0.95      0.95      5000
```

With KNEIGHBOURS CLASSIFIER the score has been changed and you can see the score has been increased to 0.9476.

KNEIGHBORS CLASSIFIER

```
In [415]: from sklearn.neighbors import KNeighborsClassifier
          knc=KNeighborsClassifier(n_neighbors=3)
          knc.fit(x_train,y_train)
          pred_knc=knc.predict(x_test)
          print(accuracy_score(y_test,pred_knc))
          print(classification_report(y_test,pred_knc))
```

```
0.9476
              precision    recall  f1-score   support

           1       0.93      0.98      0.95       991
           2       0.97      0.98      0.97      1004
           3       0.97      0.93      0.95      1010
           4       0.93      0.91      0.92      1030
           5       0.94      0.95      0.94       965

    accuracy                           0.95      5000
   macro avg       0.95      0.95      0.95      5000
weighted avg       0.95      0.95      0.95      5000
```

With DecisionTree Classifier the score has again increased to 0.9516 as it is shown in the picture below

DECISION TREE CLASSIFIER

```
In [417]: from sklearn.tree import DecisionTreeClassifier
          dtc=DecisionTreeClassifier(criterion='gini')
          dtc.fit(x_train,y_train)
          pred_dtc=dtc.predict(x_test)
          print(accuracy_score(y_test,pred_dtc))
          print(classification_report(y_test,pred_dtc))
```

```
0.9516
              precision    recall  f1-score   support

           1       0.95      0.99      0.97       991
           2       0.98      0.98      0.98      1004
           3       0.96      0.94      0.95      1010
           4       0.93      0.92      0.92      1030
           5       0.94      0.94      0.94       965

    accuracy                           0.95      5000
   macro avg       0.95      0.95      0.95      5000
weighted avg       0.95      0.95      0.95      5000
```

With Random Forest classifier we got the accuracy score of 0.9516.

Further we done the crossvalidation with all the algorithms.

All the algorithms performed well except the kneighbours classifier

Thus we further proceed to ROC-AUC curve scoring to get more better results.

The Random Forest Classifier has been selected as the best model as it has the lowest difference between the accuracy_score and the cross_val_score,a very good f-1 score and the highest roc_auc_score. The f-1 score being the most important metric in reaching this conclusion. Hyperparameter Tuning the Random Forest Classifier. The LinearSVC algorithm cannot be used here as it does not have the predict_proba method.

Finally proceding with the gridsearchCV we got a best accuracy score and finally we saved and concluded to save our model with RandomForestModel with Count vectorizer

Below is the Accuracy and classification Report

```
In [509]: #Accuracy score and classification report
          print(accuracy_score(x['Rating'],x['Predicted']))
          print(classification_report(x['Rating'],x['Predicted']))
```

```
0.9343637395868334
              precision    recall  f1-score   support

           1       0.95      0.98      0.96     15032
           2       0.93      0.99      0.96      4015
           3       0.85      0.96      0.91      7105
           4       0.82      0.95      0.88     14799
           5       0.99      0.91      0.95     45358
```

Thank You.