Rate Prediction Project

Submitted By,
Poovarasi Vijayan

Acknowledgement:

I wish to express my sincere thanks to the following companies, without whom I would have not got opportunity to work on this project; Data Trained Institute and Flip Robo Technology-Bangalore.

Introduction:

Business Problem Facing:

A website has a forum for writing technical reviews of products and consists of repository of 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. An application to predict the rating by seeing the review is required to be built. Therefore, a predictive model to accurately predict a user's rating based on input review is required to be made.

Concept Background of the Domain Problem:

In today's web-based world, virtually everyone is reading online reviews. In fact, 91% of people read them and 84% trust as much as they would a personal recommendation. The effects of reviews are measurable, too. Negative reviews can carry as much weight as positive ones. One study found that 82% of those who read online reviews specifically seek out negative reviews.

That may be sound alarming – this states that only emphasizes that negative reviews aren't going unnoticed – but there are sone benefits: Research indicates that users spend five times as long on sites when interacting with negative reviews, with an 85% increase in conversion rate.

Customers like to see lots of reviews. A single review with few positive words makes up an opinion, but a few dozen that say the same thing make a consensus.

The more reviews, the better, and one study found that consumers want to see at least 40 reviews to justify trusting an average star rating. However, a few reviews are still better than no reviews.

With the vast array of review sites and the level of trust most consumers have in reviews, it's a safe assumption that virtually everyone considering your products, no matter your target demographic, industry, or market, is reading online reviews before making a purchase.

Review of Literature:

"Review-Based Rating Prediction" by Tal Hadad was reviewed and studied to gain insights into the importance of contextual information of user sentiments in determining the rating of products, the role of natural language processing tools and techniques in identifying the user sentiments towards various products based on their reviews and ratings It is learnt that Contextual information about a user's opinion of a product can be explicit or implicit and can be inferred in different ways such as user score ratings and textual reviews. A user may express in his review(s), their satisfaction / dissatisfaction with a product, based on its quality, features performance, and monetary worth and they may then give the product a rating score based on their opinion of it. These reviews have contextual data based on users' experiences with the products and their opinions of them. The user ratings have a strong correlation with the contextual data carried in their reviews. Thus, comparing the similarity in the reviews with the similarity in the scores based on those reviews can be a basis for predicting user ratings based on the context, inference and semantics of their reviews.

Motivation for Problem Undertaken:

Ratings are an important metric in e-commerce application to determine a product's quality, consumer demand, worth and profitability. The sentiment of a user towards a product is reflected in their rating score and their review of the product. This helps determine how the product is perceived by the consumers and in turn gives an idea about the acceptance of the product by the consumers. There is a strong positive correlation between the rating of a product and its consumer demand. Therefore, it is necessary to build a predictive model which can, with good accuracy predict what rating a user might give a particular product based on the user review. This helps understand user sentiment towards a product and determine the product's worth and acceptance by consumers.

Analytical Problem Framing:

Mathematical/Analytical Modelling of the Problem:

Effective Machine Learning model needs to be created which predicts the ratings based on reviews. So 'ratings' is the target variable, here Classification problem with NLP is used to predict the ratings.

Web Scraping is done to collect data from www.flipkart.com

First, the analysis is started with importing the data, this dataset contains Null values as there are no null values, continued with data cleaning and pre-processing.

Once this is all done then data is ready for modelling. Modelling is done with Logistic Regression, Random Forest Classifier, Ada Boost Classifier, Gradient Boosting Classifier, Decision Tree Classifier, Extra Tree Classifer . From this the best model is identified and using Cross Validation technique is checked for overfitting and underfitting and Hypertuning is done to increase accuracy. Then, Finally the best model is saved.

Data Source and their Format:

The data is scrapped using selenium webscraping which is stored in xlsx format.

- Data contains 31480 entries each having 2 variables.
- Here the dataset is divided into independent and target variable.

This project is more about exploration, feature engineering and classification that can be done on this data. Since the data set is huge and includes multi classification of ratings, we can do good amount of data exploration and derive some interesting features using the Review column available. The model is built so that it can predict Ratings of the reviewer.

Data Pre-Processing:

Columns: Unnamed: 0(just a series of numbers) was dropped since it doesn't contribute to building a good model for predicting the target variable values.

• The train and test dataset contents were then converted into lowercase.

• Punctuations, unnecessary characters etc were removed, currency symbols, phone numbers, web URLs, email addresses etc were replaced with single words

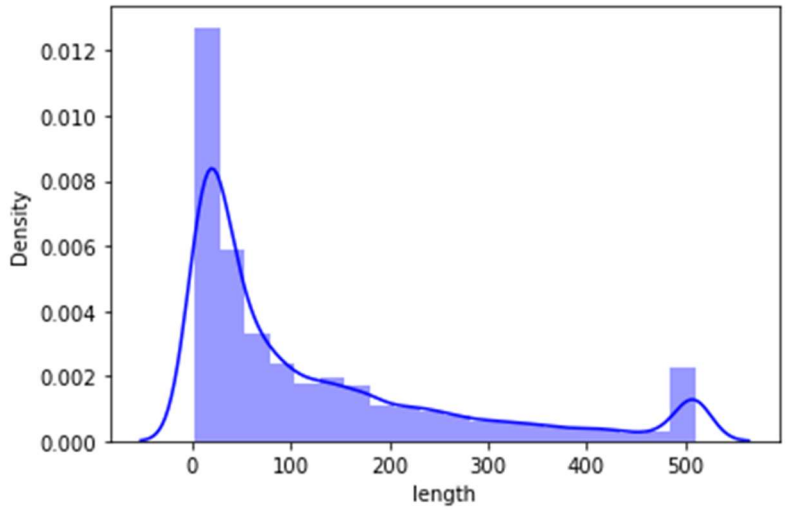
• Tokens that contributed nothing to semantics of the messages were removed as Stop words. Finally retained tokens were lemmatized using WordNetLemmatizer().
• The string lengths of original comments and the cleaned comments were then compared.

Data Input Logic Output Relations:

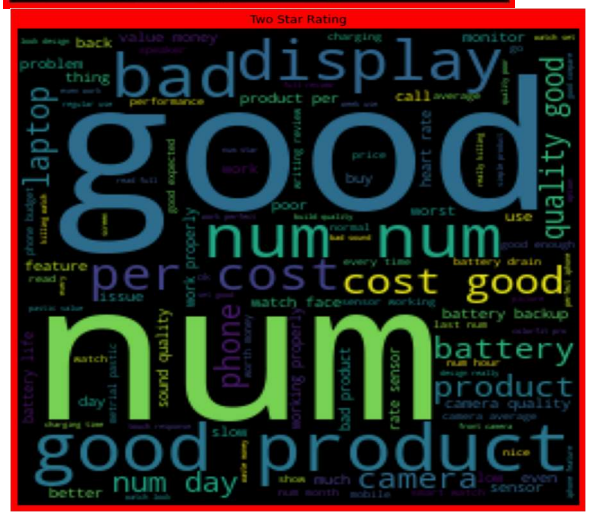
The comment tokens so vectorised using TfidVectorizer are input and the corresponding rating is predicted based on their context as output by classification models.

Data Visualization:

Ratings are an important metric in e-commerce application to determine a product's quality, consumer demand, worth and profitability. The sentiment of a user towards a product is reflected in their rating score and their review of the product. This helps determine how the product is perceived by the consumers and in turn gives an idea about the acceptance of the product by the consumers. There is a strong positive correlation between the rating of a product and its consumer demand. Therefore, it is necessary to build a predictive model which can, with good accuracy predict what rating a user might give a particular product based on the user review. This helps understand user sentiment towards a product and determine the product's worth and acceptance by consumers.



This data set contains upto 500 words with most lies below 100.



Key Observations:

1. These visualizations shows most occuring words in 5,4,3,2,1 star ratings.

2. From the graphs above the following observations are made: Reviews corresponding to 5.0 rating frequently carry words like: ‘good’, ’excellent’, ’quality’, ‘value money’ etc indicating very high customer satisfaction and high quality product.
3. Reviews corresponding to 4.0 rating frequently carry words like: ‘good’, ’great’, ’performance’, ’features’, ‘quality’, ’price’ etc indicating high customer satisfaction and good quality product.
4. Reviews corresponding to 3.0 rating frequently carry words like: ‘good’, ’well’, ’average, ’quality’, ’issue’ etc indicating customer dissatisfaction and average to below average product quality.
5. Reviews corresponding to 2.0 rating frequently carry words like: ‘problem’, ‘bad’, ’issues’, ’waste money’, ’poor quality’ etc indicating high customer dissatisfaction and below average product quality.
6. Reviews corresponding to 1.0 rating frequently carry words like: ‘stopped’, ’working’, ’cheap’, ’return’, ’issue’, ’waste money’, ’poor quality’,’ customer care’, ’bad’, ‘poor build quality’ etc indicate very high customer dissatisfaction and poor quality product.

Tools Used:

1. Python 3.8
2. Numpy
3. Pandas
4. Matplotlib
5. Seaborn
6. Data Science
7. Machine Learning

Model/s Development and Evaluation:

Test of Identified Approaches:

The approaches followed in this project are

1. Find the best random state of a model.
2. Use all the other models to find accuracy score.
3. Then find Cross Validation of all models to find the best accuracy, which is the least difference between r2 score and cv score.
4. With the best model accuracy, we need to do hyper tuning using GridSearchCV.
5. Then finally saving the best model and by keeping this we need to predict the ratings.

Run and evaluate of selected model:

```
acc = 0

for i in range(0,1000):
    x_train,x_test,y_train,y_test = train_test_split(x,y,random_state=i,test_size=.22)
    lr = LogisticRegression()
    lr.fit(x_train,y_train)
    y_pred = lr.predict(x_test)
    temp = accuracy_score(y_test,y_pred)
    if temp>acc:
        acc = temp
        best_rstate = i

print("Accuracy : ",acc*100,"RandomState : ",best_rstate)
```

```
Accuracy :  71.4369980489133 RandomState :  962
```

We got best random state 962 with accuracy 71%

```
for m in model:
    m.fit(x_train,y_train)
    m.score(x_train,y_train)
    predm = m.predict(x_test)

    print("Accuracy Score ",m," is ", accuracy_score(y_test,predm))
    print("Confusion Matrix is \n",confusion_matrix(y_test,predm))
    print("Classification report is \n",classification_report(y_test,predm))
    print("\n\n")
```

```
Accuracy Score  LogisticRegression()  is  0.7114678166427089
Confusion Matrix is
 [[3923  688  325   86   21]
 [ 546 3868  430  117    4]
 [ 261  539 3523  421  234]
 [ 101  219  827 3138  759]
 [  83  112  495  958 3366]]
Classification report is
               precision    recall  f1-score   support

            1       0.80      0.78      0.79      5043
            2       0.71      0.78      0.74      4965
            3       0.63      0.71      0.67      4978
            4       0.66      0.62      0.64      5044
            5       0.77      0.67      0.72      5014

     accuracy                           0.71     25044
    macro avg       0.71      0.71      0.71     25044
 weighted avg       0.71      0.71      0.71     25044
```

Logistic Regression gives 71.11% accuracy

```
Accuracy Score  RandomForestClassifier()  is  0.7889314805941543
Confusion Matrix is
 [[3941  715  288   81   18]
 [ 237 4161  457   96   14]
 [ 109  414 4102  260   93]
 [  40  121  825 3788  270]
 [  40   41  484  683 3766]]
Classification report is
              precision    recall  f1-score   support

           1       0.90      0.78      0.84      5043
           2       0.76      0.84      0.80      4965
           3       0.67      0.82      0.74      4978
           4       0.77      0.75      0.76      5044
           5       0.91      0.75      0.82      5014

    accuracy                           0.79     25044
   macro avg       0.80      0.79      0.79     25044
weighted avg       0.80      0.79      0.79     25044
```

RandomForestClassifier gives 78.8% accuracy

```
Accuracy Score  AdaBoostClassifier()  is  0.4646222648139275
Confusion Matrix is
 [[3939  454  508  109   33]
 [2381 1287 1047  184   66]
 [1297  371 2405  474  431]
 [ 473  134 1568 1574 1295]
 [ 453   82  913 1135 2431]]
Classification report is
              precision    recall  f1-score   support

           1       0.46      0.78      0.58      5043
           2       0.55      0.26      0.35      4965
           3       0.37      0.48      0.42      4978
           4       0.45      0.31      0.37      5044
           5       0.57      0.48      0.52      5014

    accuracy                           0.46     25044
   macro avg       0.48      0.46      0.45     25044
weighted avg       0.48      0.46      0.45     25044
```

AdaBoostClassifier() give 46.46% accuracy score

```
Accuracy Score  GradientBoostingClassifier()  is  0.587645743491455
Confusion Matrix is
 [[3647  835  453   83   25]
 [ 997 3075  696  153   44]
 [ 396  787 2954  517  324]
 [ 146  350 1196 2213 1139]
 [ 140  329  726  991 2828]]
Classification report is
               precision    recall  f1-score   support

            1       0.68      0.72      0.70      5043
            2       0.57      0.62      0.59      4965
            3       0.49      0.59      0.54      4978
            4       0.56      0.44      0.49      5044
            5       0.65      0.56      0.60      5014

     accuracy                           0.59     25044
    macro avg       0.59      0.59      0.59     25044
 weighted avg       0.59      0.59      0.59     25044
```

GradientBoostingClassifier gives 58.78% accuracy score

```
Accuracy Score  DecisionTreeClassifier()  is  0.7805063088963424
Confusion Matrix is
 [[3916  722  302   84   19]
 [ 235 4164  458   99    9]
 [ 107  419 4110  260   82]
 [  45  127  847 3814  211]
 [  52   44  557  818 3543]]
Classification report is
               precision    recall  f1-score   support

            1       0.90      0.78      0.83      5043
            2       0.76      0.84      0.80      4965
            3       0.66      0.83      0.73      4978
            4       0.75      0.76      0.75      5044
            5       0.92      0.71      0.80      5014

     accuracy                           0.78     25044
    macro avg       0.80      0.78      0.78     25044
 weighted avg       0.80      0.78      0.78     25044
```

DecisionTreeClassifier gives 78.05% accuracy

```
Accuracy Score  ExtraTreeClassifier()  is  0.7795879252515573
Confusion Matrix is
 [[3917  717  295   96   18]
 [ 234 4159  459  102   11]
 [ 102  421 4113  264   78]
 [  50  131  843 3788  232]
 [  70   52  543  802 3547]]
Classification report is
              precision    recall  f1-score   support

           1       0.90      0.78      0.83      5043
           2       0.76      0.84      0.80      4965
           3       0.66      0.83      0.73      4978
           4       0.75      0.75      0.75      5044
           5       0.91      0.71      0.80      5014

    accuracy                           0.78     25044
   macro avg       0.79      0.78      0.78     25044
weighted avg       0.80      0.78      0.78     25044
```

ExtraTreeClassifier gives 77.95% accuracy

Cross Validation:

```
Score of  LogisticRegression() is [0.67127527 0.65989518 0.683354
13 0.64387322 0.63878213]
Mean Score of  LogisticRegression() is 0.6594359870227102
Standard Deviation is  0.016620700559442564
```

CV Score of LogisticRegression is 0.6594

```
Score of  RandomForestClassifier() is [0.76386324 0.75692538 0.77
963564 0.73356626 0.74544547]
Mean Score of  RandomForestClassifier() is 0.7558871974045421
Standard Deviation is  0.015722197263250973
```

CV Score of RandomForestClassifier is 0.733

```
Score of  AdaBoostClassifier() is [0.43638632 0.44527078 0.443274
27 0.46064387 0.47057649]
Mean Score of  AdaBoostClassifier() is 0.4512303468929374
Standard Deviation is  0.012506385852456187
```

CV Score of AdaBoostClassifier is 0.4512

```
Score of  GradientBoostingClassifier() is [0.56416272 0.54903918
0.57993511 0.54509608 0.56031944]
Mean Score of  GradientBoostingClassifier() is 0.5597105066134265
Standard Deviation is  0.01230192083104044
```

CV Score of GradientBoostingClassifier is 0.5597

```
Score of  DecisionTreeClassifier() is [0.73616172 0.74100324 0.75
273272 0.7088595  0.71235338]
Mean Score of  DecisionTreeClassifier() is 0.7302221113052159
Standard Deviation is  0.016934532687689748
```

CV Score of DecisionTreeClassifier is 0.7302

```
Score of  ExtraTreeClassifier() is [0.74125281 0.74150237 0.75827
302 0.70995757 0.70741203]
Mean Score of  ExtraTreeClassifier() is 0.7316795607686549
Standard Deviation is  0.019779305565051624
```

CV Score of ExtraTreeClassifier is 0.7316

HyperTuning:

```
from sklearn.model_selection import GridSearchCV
param_grid = [{'n_estimators':[20,50,100],
               'criterion':['gini','entropy'],
               'max_features':['auto','sqrt','log2']}]

rf = RandomForestClassifier(random_state=42,n_jobs=-1)
```

```
rf_random = GridSearchCV(rf,param_grid,cv=5)
```

```
rf_random.fit(x_train,y_train)
```

```
GridSearchCV(cv=5, estimator=RandomForestClassifier(n_jobs=-1, random_state=42),
             param_grid=[{'criterion': ['gini', 'entropy'],
                          'max_features': ['auto', 'sqrt', 'log2'],
                          'n_estimators': [20, 50, 100]}])
```

```
rf_random.best_params_
```

```
{'criterion': 'gini', 'max_features': 'log2', 'n_estimators': 100}
```

```
rf_random.best_score_
```

```
0.7828859142766145
```

```
preds = cross_val_predict(rf_random.best_estimator_,x_train,y_train,cv=5,n_jobs=-1)
```

```
preds
```

```
array([4, 4, 4, ..., 3, 4, 2], dtype=int64)
```

```
predy = rfc.predict(x_test)
predy
```

```
array([4, 3, 1, ..., 2, 4, 3], dtype=int64)
```

```
classifier_final_model = rf_random.best_estimator_
y_pred = classifier_final_model.predict(x_test)
```

After Hypertuning we have 78% accuracy score which is improved.

Saving the best model:

The best model is saved using pickle.

```
import joblib
joblib.dump(predy,"RatePredictionProject.pkl")
```

```
['RatePredictionProject.pkl']
```

Key Metrics for success in solving problem under consideration:

Some of key metrics for the evaluation of this project are

1. AUC ROC Curve
2. Cross_validation
3. Hypertuning

AUC ROC Curve:

AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve (think integral calculus) from (0,0) to (1,1). Figure 5. AUC (Area under the ROC Curve). AUC provides an aggregate measure of performance across all possible classification thresholds.

Cross Validation:

Cross-validation aids in determining the model's overfitting and underfitting. The model is constructed to run on several subsets of the dataset in cross validation, resulting in numerous measurements of the model. If we fold the data five times, it will be separated into five parts, each representing 20% of the whole dataset. During the Cross-validation, the first part (20%) of the 5 parts will be left out as a holdout set for validation, while the rest of the data will be utilised for training. We'll acquire the initial estimate of the dataset's model quality this way.

Further rounds are produced in the same way for the second 20% of the dataset, which is kept as a holdout set while the remaining four portions are utilised for training data during the process. We'll acquire the second estimate of the dataset's model quality this way. During the cross-validation procedure, these stages are repeated to obtain the remaining estimate of model quality.

Hypertuning:

Hyperparameters in Machine learning are those parameters that are explicitly defined by the user to control the learning process. These hyperparameters are used to improve the learning of the model, and their values are set before starting the learning process of the model.

Conclusion:

Key Findings and Conclusions of the Study:

In this project data is collected for reviews and ratings for different products from www.flipkart.com. Then the Ratings is detected from commercial websites on a scale of 1 to 5 on the basis of the reviews given by the users. By using of natural language processing and machine learning algorithms the problem is solved. Then checked frequently occurring words in our data as well as rarely occurring words. After all these

steps built function is used to train and test different algorithms and using various evaluation metrics I have selected Random Forest Classifier for our final model. Finally, by doing hyperparameter tuning we got optimum parameters for the final model.

Learning Outcomes of the Study in respect of Data Science:

Data cleaning was a very important step in removing null values from the dataset. Visualising data helped identify class composition of label column and the most frequently occurring words in reviews corresponding to each of the rating scores. The various data pre-processing and feature engineering steps in the project lent cognizance to various efficient methods for processing textual data. The NLTK suite is very useful in pre-processing text-based data and building classification models.

Limitations of this work and Scope for Future Work:

First drawback is scrapping the data as it is a fluctuating process. Followed by raw data which is not in format to analyse. Also, this study will not cover all Regression algorithms instead, it is focused on the chosen algorithm, starting from the basic ensemble techniques to the advanced ones.

References:

A few external references have been used to complete this project successfully. Below are the sources

1. https://scikit-learn.org/
2. https://github.com/
3. https://analyticsvidhya.com/