A

Report On

Rating Prediction

Submitted By:

Rohit Kachhal

ACKNOWLEDGMENT

I would like to express my special gratitude to “Flip Robo” team, who has given me this opportunity to deal with a beautiful dataset and it has helped me to improve my analyzation skills.

A huge thanks to my academic team “Data trained” who are the reason behind what I am doing today.

INTRODUCTION

- Business Problem Framing
  - ❖ Rating prediction is a well-known recommendation task aiming to predict a user's rating for those items which were not rated yet by her. Predictions are computed from users' explicit feedback, i.e. their ratings provided on some items in the past. Another type of feedback are user reviews provided on items which implicitly express users' opinions on items. Recent studies indicate that opinions inferred from users' reviews on items are strong predictors of user's implicit feedback or even ratings and thus, should be utilized in computation.
  - ❖ The rise in E-commerce has brought a significant rise in the importance of customer reviews. There are hundreds of review sites online and massive amounts of reviews for every product. Customers have changed their way of shopping and according to a recent survey, 70 percent of customers say that they use rating filters to filter out low rated items in their searches. The ability to successfully decide whether a review will be helpful to other customers and thus give the product more exposure is vital to companies that support these reviews, companies like Google, Amazon and Yelp!. There are two main methods to approach this problem. The first one is based on review text content analysis and uses the principles of natural language process (the NLP method). This method lacks the insights that can be drawn from the relationship between costumers and items. The second one is based on recommender systems, specifically on collaborative filtering, and focuses on the reviewer's point of view.
  - ❖ E-commerce website like flip-kart, amazon etc people write different reviews for technical products. Now these type of farm are like to 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, I have to build an application which can predict the rating by seeing the review.

- Conceptual Background of the Domain Problem
  - ❖ Rating prediction is a well-known recommendation task aiming to predict a user's rating for those items which were not rated yet by customers. Predictions are computed from users' explicit feedback i.e., their ratings provided on some items in the past. Another type of feedback are user reviews provided on items which implicitly express users' opinions on items. Recent studies indicate that opinions inferred from users' reviews on items are strong predictors of user's implicit feedback or even ratings and thus, should be utilized in computation. As far as we know, all the recent works on recommendation techniques utilizing opinions inferred from users' reviews are either focused on the item recommendation task or use only the opinion information, completely leaving users' ratings out of consideration. The approach proposed in this project is filling this gap, providing a simple, personalized and scalable rating prediction framework utilizing both ratings provided by users and opinions inferred from their reviews. Experimental results provided on dataset containing user ratings and reviews from the real-world Amazon and Flipkart Product Review Data show the effectiveness of the proposed framework.

- Review of Literature
  - ❖ In real life, people's decision is often affected by friends action or recommendation. How to utilize social information has been extensively studied. Yang et al. propose the concept of "Trust Circles" in social network based on probabilistic matrix factorization. Jiang et al. propose another important factor, the individual preference. some websites do not always offer structured information, and all of these methods do not

leverage user's unstructured information, i.e. reviews, explicit social networks information is not always available and it is difficult to provide a good prediction for each user. For this problem the sentiment factor term is used to improve social recommendation.

- Motivation for the Problem Undertaken
  - ❖ Every day we come across various products in our lives, on the digital medium we swipe across hundreds of product choices under one category. It will be tedious for the customer to make selection. Here comes 'reviews' where customers who have already got that product leave a rating after using them and brief their experience by giving reviews. As we know ratings can be easily sorted and judged whether a product is good or bad. But when it comes to sentence reviews, we need to read through every line to make sure the review conveys a positive or negative sense. In the era of artificial intelligence, things like that have got easy with the Natural Language Processing (NLP) technology. Therefore, it is important to minimize the number of false positives our model produces, to encourage all constructive conversation. Our model also provides beneficence for the platform hosts as it replaces the need to manually moderate discussions, saving time and resources. Employing a machine learning model to predict ratings promotes easier way to distinguish between products qualities, costs and many other features.
  - ❖ Many product reviews are not accompanied by a scale rating system, consisting only of a textual evaluation. In this case, it becomes daunting and time-consuming to compare different products in order to eventually make a choice between them. Therefore, models able to predict the user rating from the text review are critically important. Getting an overall sense of a textual review could in turn improve consumer experience.

Analytical Problem Framing

- Mathematical/ Analytical Modelling of the Problem

  For the given problem, I have one dataset. The data is scrapped from flip-kart & amazon. The data set contains 3 columns. After reading this dataset, I can say that I have multi-classification problem. According to this dataset target variables is ‘Ratings’. There are 5 scale of ‘Ratings’. The scale is 1,2,3,4 & 5.

  I used different visualization plot to see the trend of data like distplot, barplot, imgshow & countplot to see the trend, distribution, skewness and outliers in dataset. From these plots, I can observe outliers and treat them accordingly. After that, data pre-processing, cleaning & text processing also performed. I have also done hyperparameter tuning on final model to improve performance of model but default parameter gives me best result. After that, I saved the model and compared the performance of model.

- Data Sources and their formats

  - **DataFrame**

	Review_Title	Review_Text	Ratings
0	Think twice before going for it	\n Pros ----- - Very light weight...	2.0 out of 5 stars
1	Overpriced	\n Overpriced for this mediocre product with ...	2.0 out of 5 stars
2	Build quality is not as good as shown in the Ad	\n Did not meet expectations: Takes signif...	3.0 out of 5 stars
3	Speed and Visuals makes this a power machine!	\n The speakers and sound quality are patheti...	3.0 out of 5 stars
4	Overpriced and unreliable	\n For just i5 11th gen this laptop is way ov...	3.0 out of 5 stars

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 114491 entries, 0 to 114490
Data columns (total 3 columns):
 #   Column        Non-Null Count   Dtype
---  ------        --------------   -----
 0   Review_Title  98878 non-null   object
 1   Review_Text   100649 non-null  object
 2   Ratings       98880 non-null   object
dtypes: object(3)
memory usage: 2.6+ MB
```

- Data Pre-processing Done
  - In dataset ‘Unnamed: 0’ is unnecessary column so need to drop it.
  - In dataset rest columns are of object data types.
  - Dataset contains null values.
  - As all columns are of object type, so I replace all null values in the column by taking the mode of column.
  - After that in ‘Ratings’ column some ratings are in the form of ‘ 2.0 out of 5 stars’ so, I take it as 2 and replaced rest by ‘ ‘.
  - Then I combined ‘Review_Title’ & ‘Review_Text’ column into ‘Review’ column by using map function.
  - Text pre-processing technique like convert all text in lower case, replaced full form of short form words, cleaning the dataset like removing punctuation and other special characters, Removing Stop Words, Stemming and Lemmatization, Normalization-Standardization.

- ❖ After that added two new columns in dataset with name 'Word_Count' & 'Character_Count' which contains the length of word & character.
- ❖ Outliers is also present in the dataset so i deal with it using z-score.
- ❖ After getting clean data used TF-IDF vectorizer. It'll help to transform the text data to feature vector which can be used as input in our modelling. It is a common algorithm to transform text into numbers. It measures the originality of a word by comparing the frequency of appearance of a word in a document with the number of documents the words appear in.

- Data Inputs- Logic- Output Relationships
  - ❖ I used count plot to visualize the trend.
  - ❖ All data now are of object type.

- Hardware and Software Requirements and Tools Used
  - ❖ Hardware:
    - Processor: I3 or above
    - RAM: 4 GB or above
    - Storage: 250 GB or above
  - ❖ Software:
    - Anaconda
    - Jupyter notebook etc
  - ❖ Libraries:
    - numpy
    - pandas
    - matplotlib
    - seaborn
    - model_selection: train_test_split, cross_val_score, GridSearchCV
    - score: Accuracy score, recall, precision, confusion metrics

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  First of all, I dropped the unnecessary and unwanted columns from the data frame by using **drop ()**. After that, I replaced null values in the column by taking mode of the column using **replace().** After that, in 'Ratings' column some ratings are in the form of ' 2.0 out of 5 stars' so I take it as 2 and replace rest by ' ' using **replace().** Then I combined 'Review_Title' & 'Review_Text' column into 'Review' column by using **map()**.I get 5 different scale in 'Ratings' column. After that, text pre-processing and cleaning the data like converting the text into lower case by using **str.lower()**, Text pre-processing technique like replace full form of short form words, cleaning the dataset like removing punctuation and other special characters using user-defined function **decon()**, Removing Stop Words, Stemming and Lemmatization, Normalization-Standardization. After that, added two new columns in dataset with name 'Word_Count' & 'Character_Count' which contains the length of word & character. It is observed that outliers is also present so dead it by using **z-score().** After getting a cleaned data used TF-IDF vectorizer. It'll help to transform the text data to feature vector which can be used as input in our modelling. It is a common algorithm to transform text into numbers. It measures the originality of a word by comparing the frequency of appearance of a word in a document with the number of documents the words appear in. Finally, using **train_test_split** split the dataset and initiate the model.

- Testing of Identified Approaches (Algorithms)

  As the problem is regression problem. In machine learning there are several algorithms for regression problem. I used total 5 algorithm to build best model on this dataset. The algorithms are:

  - ❖ Logistic Regression
  - ❖ Decision Tree Classifier
  - ❖ Random Forest Classifier
  - ❖ Support Vector Classifier
  - ❖ SDG Classifier

- Run and Evaluate selected models

  Here are the coding and insights of all models:

  ❖ Logistic Regression

```
lg =LogisticRegression()
lg.fit(X_train,y_train)
lg_pred = lg.predict(X_test)
lg_accuracy = accuracy_score(y_test,lg_pred)
lg_cf = classification_report(y_test,lg_pred)
lg_cm = confusion_matrix(y_test,lg_pred)
lg_train_score = lg.score(X_train,y_train)
lg_test_score = lg.score(X_test,y_test)

print('Logistic Regression')
print('-----------------------------------------------------------------\n')
print('The Score on train set is   :',lg_train_score)
print('The Score on test set is    :',lg_test_score)
print('The Accuracy on test set is :',lg_accuracy)
print('The Classification report is :\n',lg_cf)
print('The Confusion matrix is :\n',lg_cm)
print('\n-----------------------------------------------------------------')
```

```
Logistic Regression
-----------------------------------------------------------------

The Score on train set is   : 0.8464313145392997
The Score on test set is    : 0.7339392687461268
The Accuracy on test set is : 0.7339392687461268
The Classification report is :
              precision    recall  f1-score   support

           1       0.71      0.77      0.74      5022
           2       0.49      0.51      0.50      3359
           3       0.56      0.50      0.53      4274
           4       0.61      0.69      0.65      6147
           5       0.91      0.86      0.88     15085

    accuracy                           0.73     33887
   macro avg       0.66      0.66      0.66     33887
weighted avg       0.74      0.73      0.74     33887

The Confusion matrix is :
 [[ 3854   709   266   125    68]
 [  815  1697   522   246    79]
 [  456   665  2116   803   234]
 [  145   235   571  4226   970]
 [  137   151   309  1510 12978]]

-----------------------------------------------------------------
```

a). Cross Validation on Logistic Regression

```
lg_cv_score = cross_val_score(lg,X_train,y_train,cv=3)
print('The cross validation score is :',lg_cv_score.mean())
```

```
The cross validation score is : 0.8114815016246544
```

❖ Decision Tree Classifier

```
dt = DecisionTreeClassifier()
dt.fit(X_train,y_train)
dt_pred = dt.predict(X_test)
dt_accuracy = accuracy_score(y_test,dt_pred)
dt_cf = classification_report(y_test,dt_pred)
dt_cm = confusion_matrix(y_test,dt_pred)
dt_train_score = dt.score(X_train,y_train)
dt_test_score = dt.score(X_test,y_test)

print('Decision Tree Classifier')
print('----------------------------------------------------------------------\n')
print('The Score on train set is   :',dt_train_score)
print('The Score on test set is    :',dt_test_score)
print('The Accuracy on test set is :',dt_accuracy)
print('The Classification report is :\n',dt_cf)
print('The Confusion matrix is :\n',dt_cm)
print('\n----------------------------------------------------------------------')
```

```
Decision Tree Classifier
----------------------------------------------------------------------

The Score on train set is   : 0.9929792620828948
The Score on test set is    : 0.7071443326349337
The Accuracy on test set is : 0.7071443326349337
The Classification report is :
              precision    recall  f1-score   support

           1       0.68      0.66      0.67      5022
           2       0.48      0.52      0.50      3359
           3       0.54      0.55      0.54      4274
           4       0.61      0.63      0.62      6147
           5       0.87      0.84      0.85     15085

    accuracy                           0.71     33887
   macro avg       0.63      0.64      0.64     33887
weighted avg       0.71      0.71      0.71     33887

The Confusion matrix is :
 [[ 3339   718   435   290   240]
 [  654  1732   410   318   245]
 [  431   507  2347   603   386]
 [  248   354   609  3901  1035]
 [  270   300   575  1296 12644]]

----------------------------------------------------------------------
```

b). Cross validation on Decision Tree Classifier

```
dt_cv_score = cross_val_score(dt,X_train,y_train,cv=3)
print('The cross validation score is :',dt_cv_score.mean())
```

```
The cross validation score is : 0.7580863375515019
```

❖ Random Forest Classifier

```
rc = RandomForestClassifier()
rc.fit(X_train,y_train)
rc_pred = rc.predict(X_test)
rc_accuracy = accuracy_score(y_test,rc_pred)
rc_cf = classification_report(y_test,rc_pred)
rc_cm = confusion_matrix(y_test,rc_pred)
rc_train_score = rc.score(X_train,y_train)
rc_test_score = rc.score(X_test,y_test)

print('Random Forest Classifier')
print('------------------------------------------------------------------\n')
print('The Score on train set is   :',rc_train_score)
print('The Score on test set is    :',rc_test_score)
print('The Accuracy on test set is :',rc_accuracy)
print('The Classification report is :\n',rc_cf)
print('The Confusion matrix is :\n',rc_cm)
print('\n------------------------------------------------------------------')
```

```
Random Forest Classifier
------------------------------------------------------------------

The Score on train set is   : 0.9929792620828948
The Score on test set is    : 0.7669902912621359
The Accuracy on test set is : 0.7669902912621359
The Classification report is :
              precision    recall  f1-score   support

           1       0.71      0.84      0.77      5022
           2       0.63      0.52      0.57      3359
           3       0.66      0.54      0.59      4274
           4       0.62      0.75      0.68      6147
           5       0.91      0.87      0.89     15085

    accuracy                           0.77     33887
   macro avg       0.71      0.70      0.70     33887
weighted avg       0.77      0.77      0.77     33887

The Confusion matrix is :
 [[ 4220   357   173   190    82]
 [  824  1737   397   295   106]
 [  488   388  2288   879   231]
 [  205   143   375  4598   826]
 [  180   111   219  1427 13148]]

------------------------------------------------------------------
```

c). Cross validation on Random Forest Classifier

```
rc_cv_score = cross_val_score(rc,X_train,y_train,cv=3)
print('The cross validation score is :',rc_cv_score.mean())
```

```
The cross validation score is : 0.883446758822633
```

❖ Support Vector Classifier

```
svc = LinearSVC()
svc.fit(X_train,y_train)
svc_pred = svc.predict(X_test)
svc_accuracy = accuracy_score(y_test,svc_pred)
svc_cf = classification_report(y_test,svc_pred)
svc_cm = confusion_matrix(y_test,svc_pred)
svc_train_score = svc.score(X_train,y_train)
svc_test_score = svc.score(X_test,y_test)

print('Support Vector Classifier')
print('-----------------------------------------------------------------\n')
print('The Score on train set is   :',svc_train_score)
print('The Score on test set is    :',svc_test_score)
print('The Accuracy on test set is :',svc_accuracy)
print('The Classification report is :\n',svc_cf)
print('The Confusion matrix is :\n',svc_cm)
print('\n-----------------------------------------------------------------')
```

```
Support Vector Classifier
-----------------------------------------------------------------

The Score on train set is   : 0.9668648479503883
The Score on test set is    : 0.756838905775076
The Accuracy on test set is : 0.756838905775076
The Classification report is :
              precision    recall  f1-score   support

           1       0.73      0.78      0.75      5022
           2       0.56      0.55      0.55      3359
           3       0.59      0.57      0.58      4274
           4       0.66      0.68      0.67      6147
           5       0.90      0.88      0.89     15085

    accuracy                           0.76     33887
   macro avg       0.69      0.69      0.69     33887
weighted avg       0.76      0.76      0.76     33887

The Confusion matrix is :
 [[ 3893   593   314   130    92]
 [  706  1841   481   240    91]
 [  424   516  2446   638   250]
 [  158   216   597  4164  1012]
 [  126   149   325  1182 13303]]

-----------------------------------------------------------------
```

d). Cross validation on Support Vector Classifier

```
svc_cv_score = cross_val_score(svc,X_train,y_train,cv=3)
print('The cross validation score is :',svc_cv_score.mean())
```

```
The cross validation score is : 0.9117288516653463
```

❖ SDG Classifier

```
sdg = SGDClassifier()
sdg.fit(X_train,y_train)
sdg_pred = sdg.predict(X_test)
sdg_accuracy = accuracy_score(y_test,sdg_pred)
sdg_cf = classification_report(y_test,sdg_pred)
sdg_cm = confusion_matrix(y_test,sdg_pred)
sdg_train_score = sdg.score(X_train,y_train)
sdg_test_score = sdg.score(X_test,y_test)

print('SDG Classifier')
print('------------------------------------------------------------------\n')
print('The Score on train set is   :',sdg_train_score)
print('The Score on test set is    :',sdg_test_score)
print('The Accuracy on test set is :',sdg_accuracy)
print('The Classification report is :\n',sdg_cf)
print('The Confusion matrix is :\n',sdg_cm)
print('\n------------------------------------------------------------------')
```

```
SDG Classifier
------------------------------------------------------------------

The Score on train set is   : 0.7991636560179786
The Score on test set is    : 0.7223419010239915
The Accuracy on test set is : 0.7223419010239915
The Classification report is :
              precision    recall  f1-score   support

           1       0.64      0.85      0.73      5022
           2       0.53      0.38      0.45      3359
           3       0.64      0.40      0.49      4274
           4       0.56      0.73      0.63      6147
           5       0.90      0.84      0.87     15085

    accuracy                           0.72     33887
   macro avg       0.66      0.64      0.63     33887
weighted avg       0.73      0.72      0.72     33887

The Confusion matrix is :
 [[ 4267   358   149   171    77]
 [ 1236  1289   363   369   102]
 [  682   511  1701  1139   241]
 [  250   158   288  4510   941]
 [  226    95   159  1894 12711]]

------------------------------------------------------------------
```

e). Cross validation on SDG Classifier

```
sdg_cv_score = cross_val_score(sdg,X_train,y_train,cv=3)
print('The cross validation score is :',sdg_cv_score.mean())
```

```
The cross validation score is : 0.7729641783116062
```

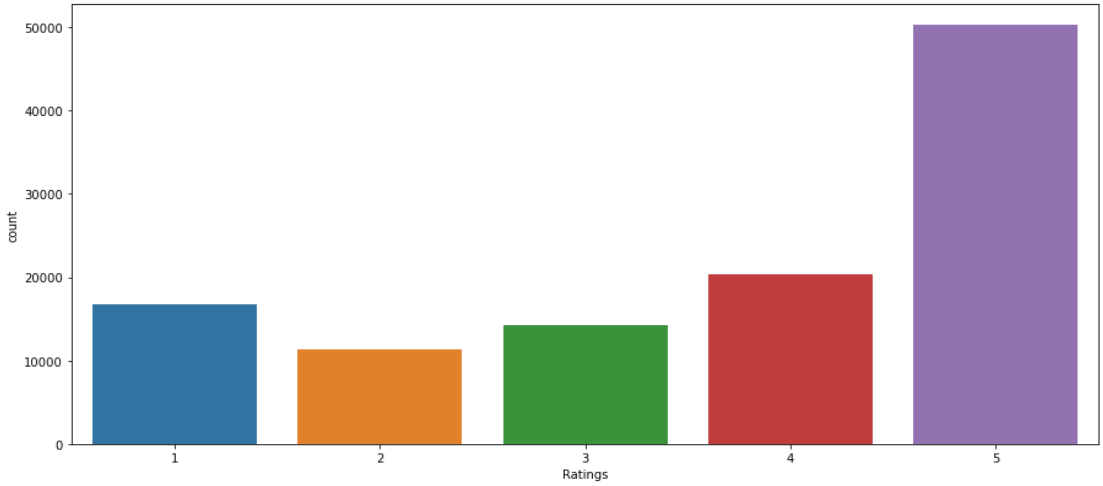
- Key Metrics for success in solving problem under consideration

  Following are the metrics to observe good model:

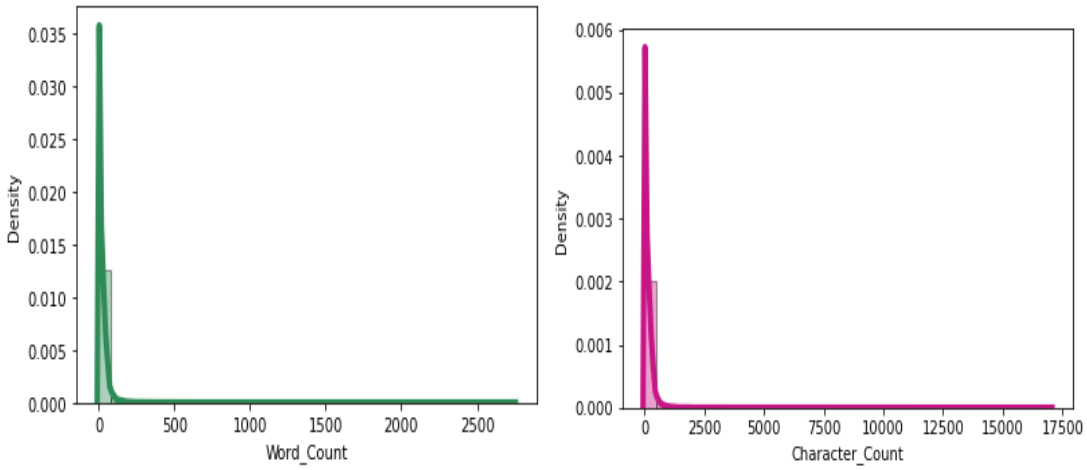
  - ❖ Accuracy simply measures how often the classifier correctly predicts. Accuracy can be defined as the ratio of the number of correct predictions and the total number of predictions.
  - ❖ Precision explains how many of the correctly predicted cases actually turned out to be positive. Precision is useful in the cases where False Positive is a higher concern than False Negatives.
  - ❖ Recall explains how many of the actual positive cases we were able to predict correctly with our model. It is a useful metric in cases where False Negative is of higher concern than False Positive.
  - ❖ It gives a combined idea about Precision and Recall metrics. It is maximum when Precision is equal to Recall.
  - ❖ Cross_val_score: Cross-validation is a resampling method that uses different portions of the data to test and train a model on different iterations. It is mainly used in settings where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice.

- Visualizations

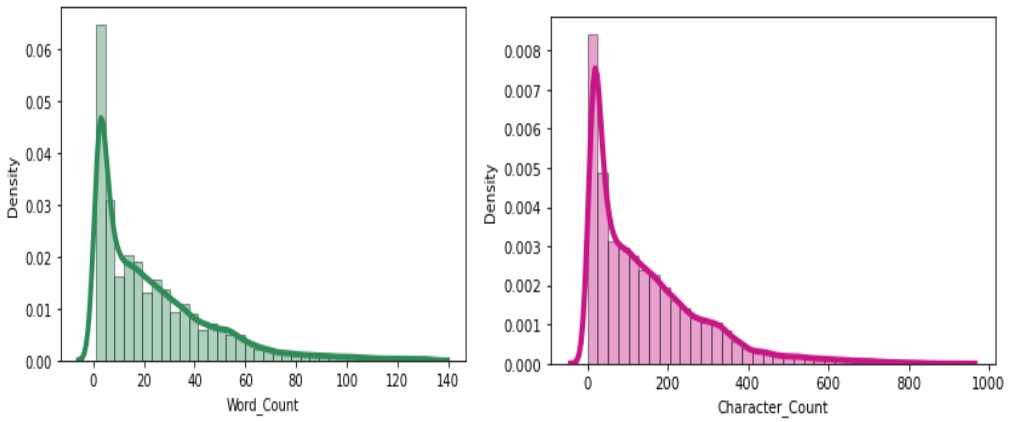
Rating Analysis:



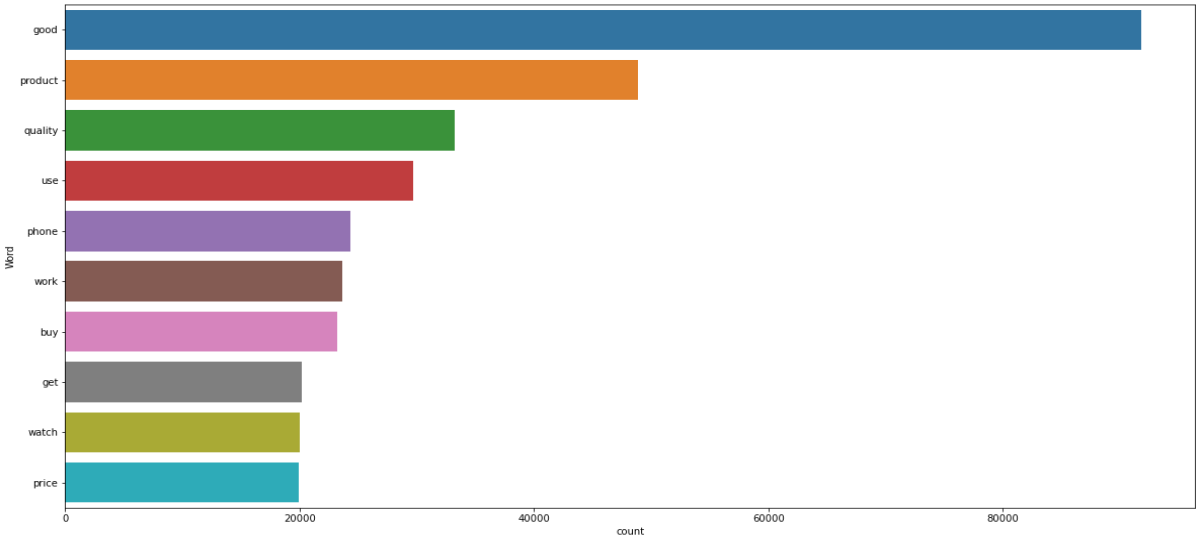
Word & Character Count:



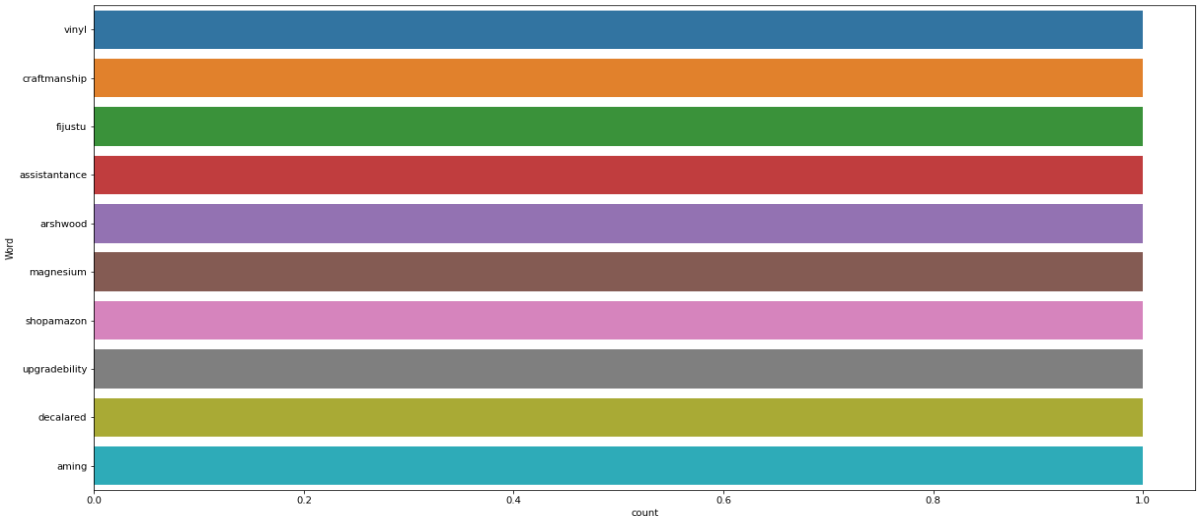
Word & Character Count After Removal Of Outliers:



Top 10 Frequently Used Word:



Top 10 Rarely Used Word:



Word Cloud Rating Wise:

- Interpretation of the Results
  - ❖ Score of all model

	Model Name	Train Score	Test Score	Accuracy Score	Cross Validation Score
0	Logistic Regression	0.846431	0.733939	0.733939	0.811482
1	Decision Tree Classifier	0.992979	0.707144	0.707144	0.758086
2	Random Forest Classifier	0.992979	0.766990	0.766990	0.883447
3	Support Vector Classifier	0.966865	0.756839	0.756839	0.911729
4	SDG Classifier	0.799164	0.722342	0.722342	0.772964

Outcome:

SDG classifier performed well on this dataset as SDG Classifier has good accuracy, comparatively less cross-validation score. So, final model on this dataset is SDG Classifier.

❖ Hyperparameter Tuning

```
{'loss': 'squared_hinge', 'n_jobs': -1, 'penalty': 'l1'}

 SDG Classifier
 ------------------------------------------------------------------------

 The Score on train set is    : 0.8392000682729781
 The Score on test set is     : 0.6850119514858205
 The Accuracy on test set is  : 0.6850119514858205
 The Classification report is :
                precision    recall  f1-score   support

             1       0.66      0.66      0.66      5022
             2       0.43      0.44      0.44      3359
             3       0.47      0.47      0.47      4274
             4       0.57      0.58      0.58      6147
             5       0.86      0.85      0.86     15085

     accuracy                           0.69     33887
    macro avg       0.60      0.60      0.60     33887
 weighted avg       0.69      0.69      0.69     33887

 The Confusion matrix is :
  [[ 3319   770   492   261   180]
  [  742  1489   607   333   188]
  [  497   609  2017   722   429]
  [  239   344   719  3560  1285]
  [  204   259   465  1329 12828]]

 ------------------------------------------------------------------------
```

Outcome:

- ✓ After hyperparameter tuning, I am not able to improve the performance of this model. So, default parameter gives better result.

CONCLUSION

- Key Findings and Conclusions of the Study
  - ❖ This research evaluated the rating of a product classification using machine learning and deep learning techniques. Using real data, we compared the various machine learning algorithms' accuracy by performing detailed experimental analysis while classifying the text into 5 categories.

- Learning Outcomes of the Study in respect of Data Science
  - ❖ After working on this project, I learnt various Natural Language Processing techniques like lemmatization, stemming, removal of Stop Words etc.
  - ❖ According to the study I came to know the importance of sampling, modelling and predicting data. By the use of different visualization tools, I was able to analyse and interpretation different insights of the dataset. The challenges I faced while working on this dataset are:
    - Imbalanced Dataset
    - Lots of Text data
  - ❖ To deal with imbalanced dataset, I used oversampling method of smote algorithm. Then I converted text data into vectors with the help of TfidfVectorizer.

- Limitations of this work and Scope for Future Work
  - ❖ As we know the content of text in reviews is totally depends on the reviewer and they may rate differently which is totally depends on that particular person. So it is difficult to predict ratings based on the reviews with higher accuracies. Still we can improve our accuracy by fetching more data and by doing extensive hyperparameter tuning.

- As the model accuracy near about 72.23 % and I could not reach out goal of maximum accuracy in this project. While building any machine learning project there will be a scope of for improvement .
- We can improve rating prediction system with the addition of more algorithm into it.