FLIP ROBO

RATING PREDICATION BASED ON REVIEWS USING NLP

Submitted by:

YASH BHARDWAJ

ACKNOWLEDGMENT

I whole heartedly thank our SME Sapna Verma, flip robo technologies for their support towards me to complete this project.

I also thank my family for supporting me.

Yash Bhardwaj

Contents

INTRODUCTION

Business Problem Framing

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.

Conceptual Background of the Domain Problem

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!

In this paper, we use different Machine Learning Classifier approach to predict product rating by providing reviews of the products.

Review of Literature

According to the Lackermair, Kailer and Kanmaz (2013), product reviews and ratings represent an important source of information for consumers and are helpful tools in order to support their buying decisions. They also found out that consumers are willing to compare both positive and negative reviews when searching for a specific product. The authors argue that customers need compact and concise information about the products.

Therefore, consumers first need to pre-select the potential products matching their requirements. With this aim in mind, consumers use the star ratings as an indicator for selecting products. Later, when a limited number of potentials products have been chosen, reading the associated text review will reveal more details about the products and therefore help consumers making a final decision. 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 (Baccianella, Esuli & Sebastiani, 2009).

Chevalier and Mayzlin (2006) also analyse the distribution of ratings in online reviews and come to the same conclusion: the resulting data presents an asymmetric bimodal distribution, where reviews are overwhelmingly positive.

Pang, Lee and Vaithyanathan (2002) approach this predictive task as an opinion mining problem enabling to automatically distinguish between positive and negative reviews. In order to determine the reviews polarity, the authors use text classification techniques by training and testing binary classifiers on movie reviews containing 36.6% of negative reviews and 63.4% of positive reviews. On the top of that, they also try to identify appropriate features to enhance the performance of the classifiers.

Motivation for the Problem Undertaken

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.

Data was required to be scraped from E-commerce platform and data cleaning operation over it. Features derived from textual reviews are used to predict its corresponding star ratings. To accomplish it, the prediction problem is transformed into a multi-class classification task to classify reviews to one of the five classes corresponding to its star rating. Getting an overall sense of a textual review could in turn improve consumer experience. However, the motivation for taking this project was that it is relatively a new field of research.

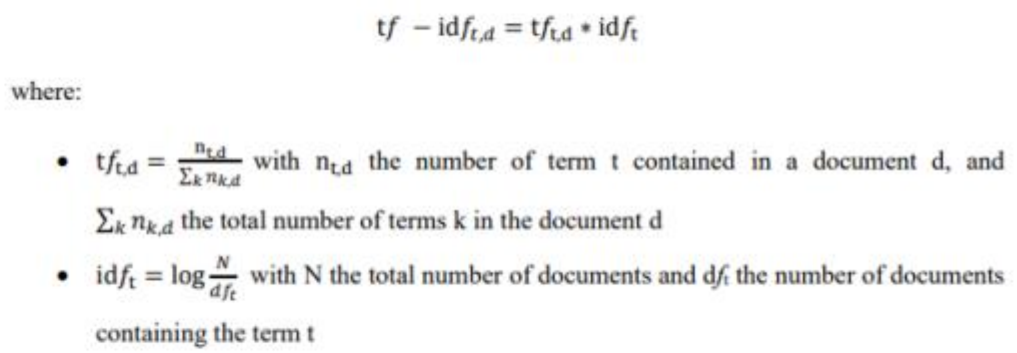
Analytical Problem Framing

Mathematical/ Analytical Modeling of the Problem

In order to apply text classification, the unstructured format of text has to be converted into a structured format for the simple reason that it is much easier for computer to deal with numbers than text. This is mainly achieved by projecting the textual contents into Vector Space Model, where text data is converted into vectors of numbers.

In the field of text classification, documents are commonly treated like a Bag-of-Words (BoW), meaning that each word is independent from the others that are present in the document. They are examined without regard to grammar neither to the word order. In such a model, the term frequency (occurrence of each word) is used as a feature in order to train the classifier.

However, using the term frequency implies that all terms are considered equally important. As its name suggests, the term frequency simply weights each term based on their occurrence frequency and does not take the discriminatory power of terms into account. To address this problem and penalize words that are too frequent, each word is given a term frequency inverse document frequency (tf-idf) score which is defined as follow:



$$tf\ -\mathrm{id}f_{t,d} = \mathrm{t}f_{\mathrm{t,d}} * \mathrm{id}f_{\mathrm{t}}$$

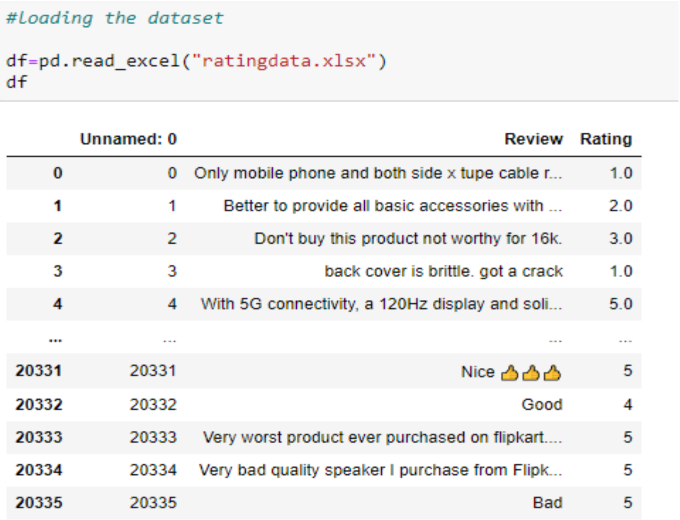
where:

- $\mathrm{t}f_{\mathrm{t,d}} = \frac{\mathrm{n_{t,d}}}{\sum_k n_{k,d}}$ with $\mathrm{n_{t,d}}$ the number of term t contained in a document d, and $\sum_k n_{k,d}$ the total number of terms k in the document d
- $\mathrm{id}f_{\mathrm{t}} = \log\frac{N}{df_t}$ with N the total number of documents and df_t the number of documents containing the term t

Data Sources and their formats

So, for this project data was scraped using selenium from e-commerce websites amazon.com and flipkart.com and saved in excel file. Around 20000 reviews with their ratings are collected.

The dataset looks like:



```
#loading the dataset

df=pd.read_excel("ratingdata.xlsx")
df
```

	Unnamed: 0	Review	Rating
0	0	Only mobile phone and both side x tupe cable r...	1.0
1	1	Better to provide all basic accessories with ...	2.0
2	2	Don't buy this product not worthy for 16k.	3.0
3	3	back cover is brittle. got a crack	1.0
4	4	With 5G connectivity, a 120Hz display and soli...	5.0
...	...	...	...
20331	20331	Nice 👍👍👍	5
20332	20332	Good	4
20333	20333	Very worst product ever purchased on flipkart....	5
20334	20334	Very bad quality speaker I purchase from Flipk...	5
20335	20335	Bad	5

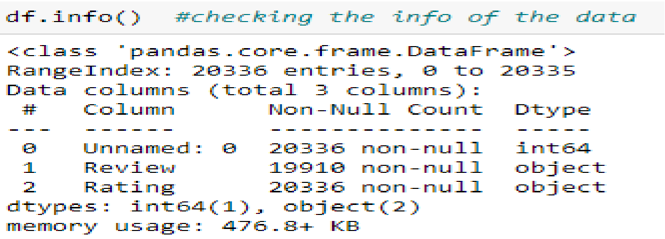
This is multi-classification problem and Rating is our target feature class to be predicated in this project.

Data Preprocessing Done

The dataset is large and it may contain some data error. In order to reach clean, error free data some data cleaning & data pre-processing performed data.

Firstly, an unnamed column should be dropped from the dataset as it represents the index only of the rows. Afterwards,

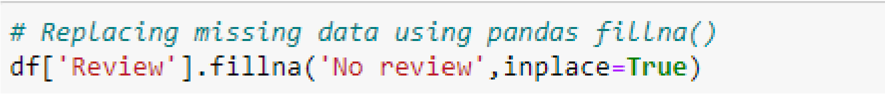
Data integrity check –



```
df.info()   #checking the info of the data
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20336 entries, 0 to 20335
Data columns (total 3 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   Unnamed: 0   20336 non-null  int64 
 1   Review       19910 non-null  object
 2   Rating       20336 non-null  object
dtypes: int64(1), object(2)
memory usage: 476.8+ KB
```

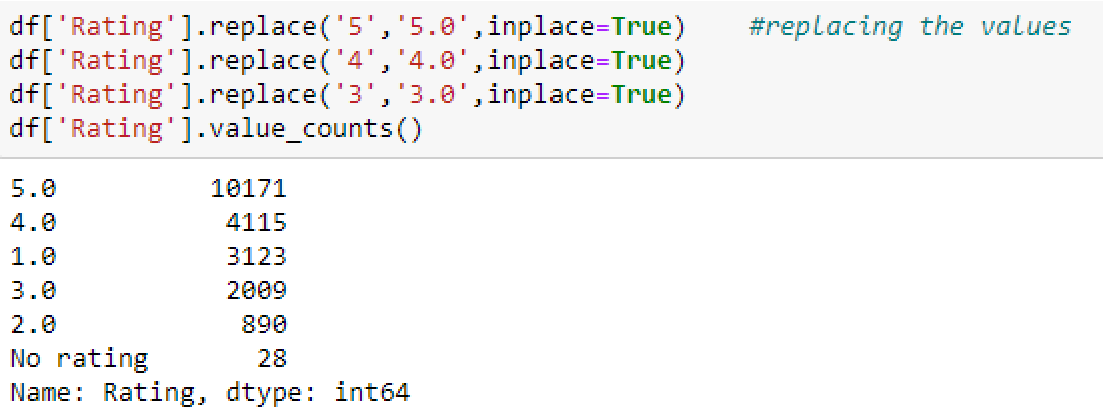
The Review variable has some missing values present which are going to be replaced by ‘No review’:



```
# Replacing missing data using pandas fillna()
df['Review'].fillna('No review',inplace=True)
```

Conversion of target variable to numeric datatype –

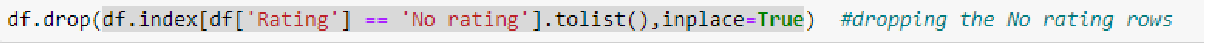
Initially, the datatype of target variable is of object or categorical type. Also the target variable has some values as 5, 3, 4 and some as 5.0, 3.0, 4.0, so we have to replace the first one with the latter one.



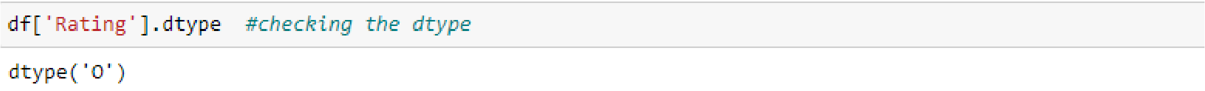
```
df['Rating'].replace('5','5.0',inplace=True)    #replacing the values
df['Rating'].replace('4','4.0',inplace=True)
df['Rating'].replace('3','3.0',inplace=True)
df['Rating'].value_counts()
```

```
5.0          10171
4.0           4115
1.0           3123
3.0           2009
2.0            890
No rating       28
Name: Rating, dtype: int64
```

The target variable also has some values as ‘No rating’ and as the count is very less for this class let’s drop these values from the dataset and convert the variable into numeric one.

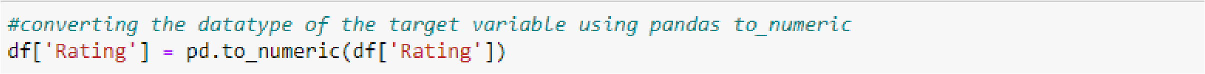


```
df.drop(df.index[df['Rating'] == 'No rating'].tolist(),inplace=True)  #dropping the No rating rows
```



```
df['Rating'].dtype  #checking the dtype
```

```
dtype('O')
```



```
#converting the datatype of the target variable using pandas to_numeric
df['Rating'] = pd.to_numeric(df['Rating'])
```

Now the data is pre-processed using the following techniques:

- Removing inverted commas and other special characters
- Removing punctuations
- Removing stop-words
- Lemmatizing
- Converting into vectors

- Hardware and Software Requirements and Tools Used

Hardware used:

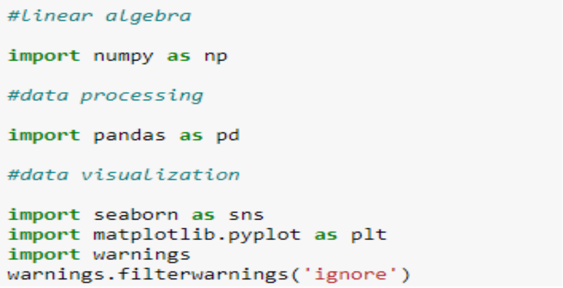
1. Processor — AMD A9-9425 RADEON R5, 5 COMPUTE CORES 2C+3G 3.10 GHz
2. RAM — 4 GB
3. GPU — AMD Radeon(TM) R5 Graphics

Software utilised –

1. Anaconda – Jupyter Notebook
2. Selenium – Web scraping

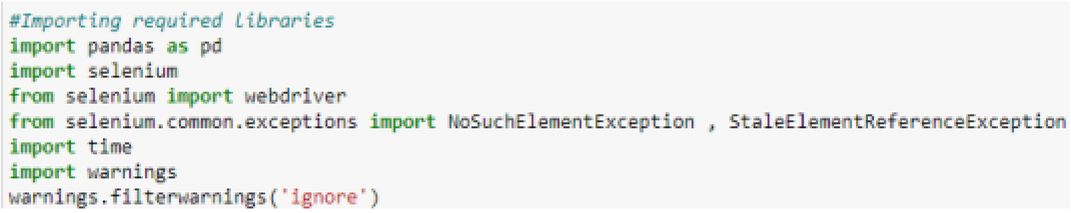
Libraries Used –

General library for data wrangling & visualisation:



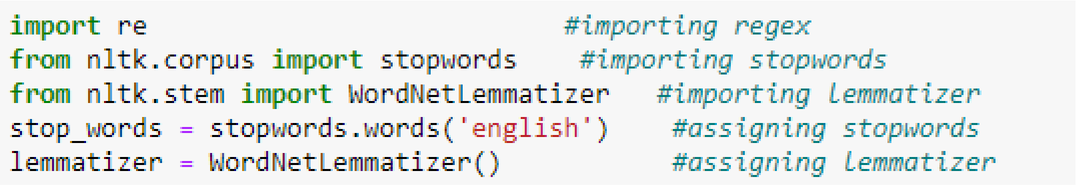
```
#linear algebra

import numpy as np

#data processing

import pandas as pd

#data visualization

import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

Libraries used for web scraping data from e-commerce website:



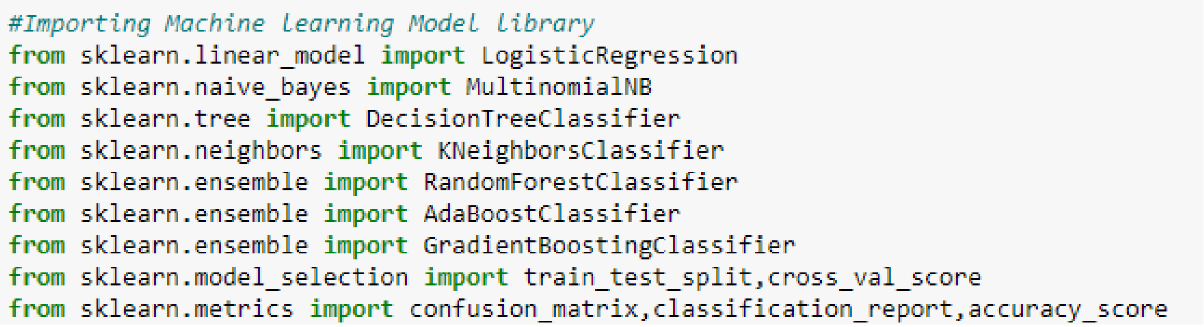
```
#Importing required libraries
import pandas as pd
import selenium
from selenium import webdriver
from selenium.common.exceptions import NoSuchElementException , StaleElementReferenceException
import time
import warnings
warnings.filterwarnings('ignore')
```

Libraries used for text pre-processing:



```
import re                                  #importing regex
from nltk.corpus import stopwords           #importing stopwords
from nltk.stem import WordNetLemmatizer     #importing lemmatizer
stop_words = stopwords.words('english')     #assigning stopwords
lemmatizer = WordNetLemmatizer()            #assigning lemmatizer
```

Libraries used for model building and its metrics:



```
#Importing Machine Learning Model library
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn.metrics import confusion_matrix,classification_report,accuracy_score
```

Model/s Development and Evaluation

Identification of possible problem-solving approaches (methods)

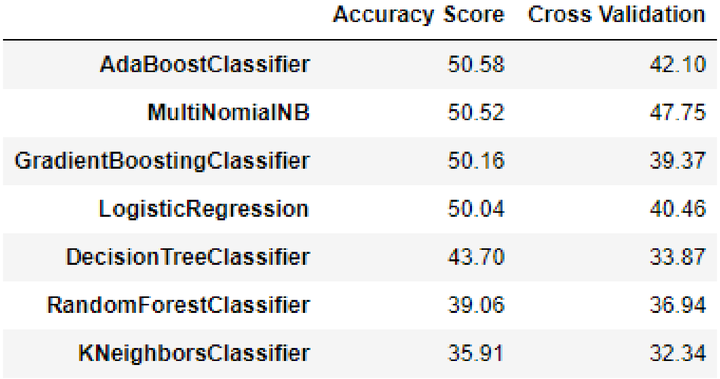
First part of problem solving is to scrape data from e-commerce websites like amazon.in and flipkart.com which we have already done. Second is performing text mining operation to convert textual review in ML algorithm useable form. Third part of problem building machine learning model to predict rating on review. This problem can be solve using classification-based machine learning algorithm like logistic regression. Further, hyperparameter tuning is performed to build a more accurate model out of best model.

Testing of Identified Approaches (Algorithms)

The different ML algorithms used are:

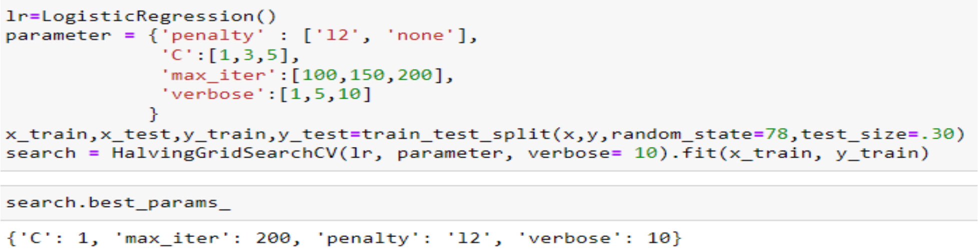
- Logistic Regression
- KNeighbors Classifier
- Decision Tree Classifier
- MultinomialNB
- Ada boost Classifier
- Gradient Boosting Classifier
- Random Forest Classifier

Best Model?



	Accuracy Score	Cross Validation
AdaBoostClassifier	50.58	42.10
MultiNomialNB	50.52	47.75
GradientBoostingClassifier	50.16	39.37
LogisticRegression	50.04	40.46
DecisionTreeClassifier	43.70	33.87
RandomForestClassifier	39.06	36.94
KNeighborsClassifier	35.91	32.34

- We are going to select Logistic Regression model as our best model because it is giving good cross validation score and good recall as well for all the classes of target variable when compared to other algorithm and their model.
- Below is the code and its output for the hyperparameter tuning of the logistic regression model:



```
lr=LogisticRegression()
parameter = {'penalty' : ['l2', 'none'],
             'C':[1,3,5],
             'max_iter':[100,150,200],
             'verbose':[1,5,10]
            }
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=78,test_size=.30)
search = HalvingGridSearchCV(lr, parameter, verbose= 10).fit(x_train, y_train)
```

```
search.best_params_
```

```
{'C': 1, 'max_iter': 200, 'penalty': 'l2', 'verbose': 10}
```

Key Metrics for success in solving problem under consideration

▪ Precision can be seen as a measure of quality; higher precision means that an algorithm returns more relevant results than irrelevant ones.

▪ Recall is used as a measure of quantity and high recall means that an algorithm returns most of the relevant results.

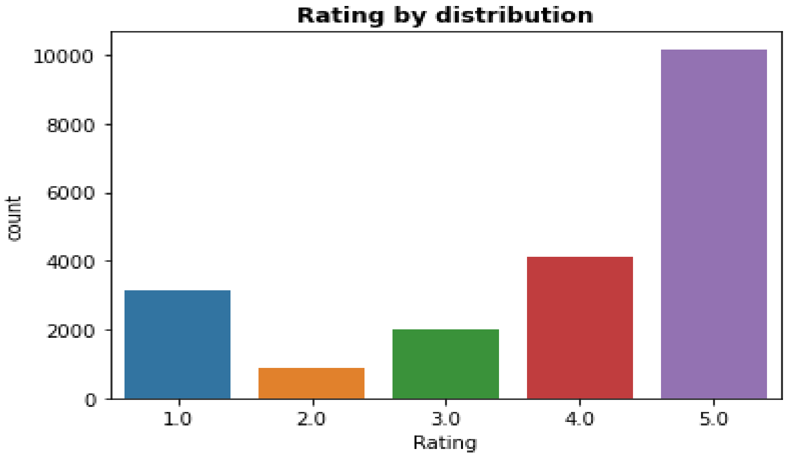
▪ Accuracy score is used when the True Positives and True negatives are more important. Accuracy can be used when the class distribution is similar.

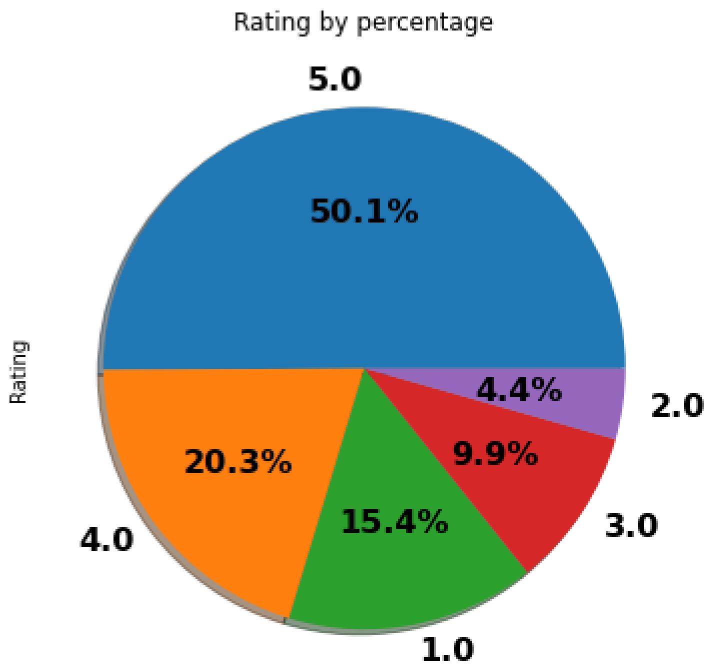
▪ F1-score is used when the False Negatives and False Positives are crucial. While F1-score is a better metric when there are imbalanced classes.

▪ Cross validation Score: To run cross-validation on multiple metrics and also to return train scores, fit times and score times. Get predictions from each split of cross-validation for diagnostic purposes. Make a scorer from a performance metric or loss function.

▪ Used Accuracy Score and Cross validation score as key parameter for model evaluation in this project since balancing of data is perform.

Visualizations



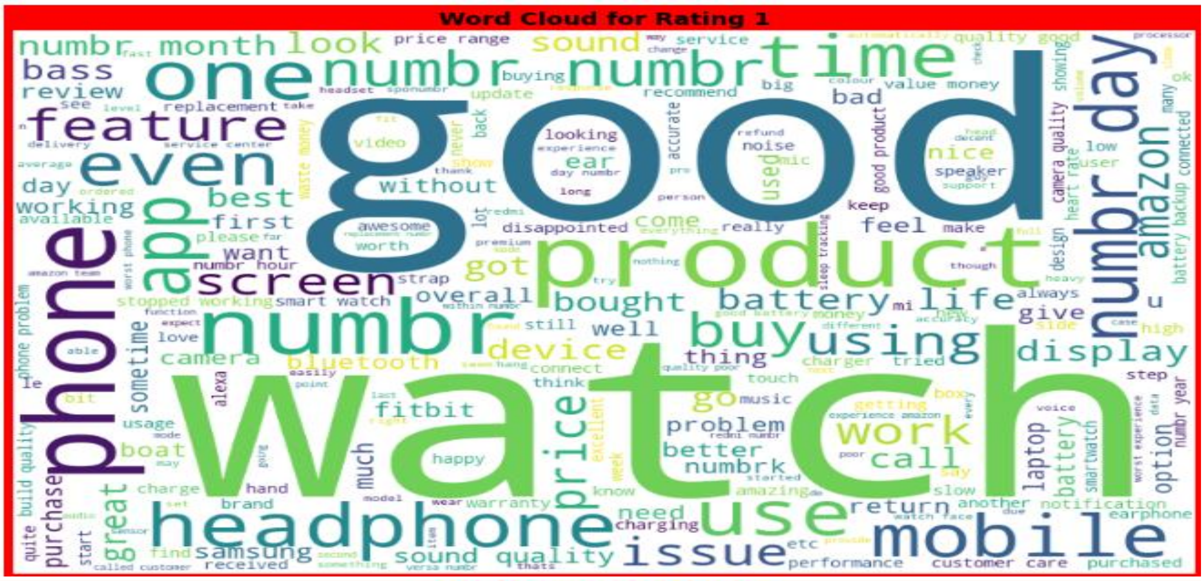


Observations:

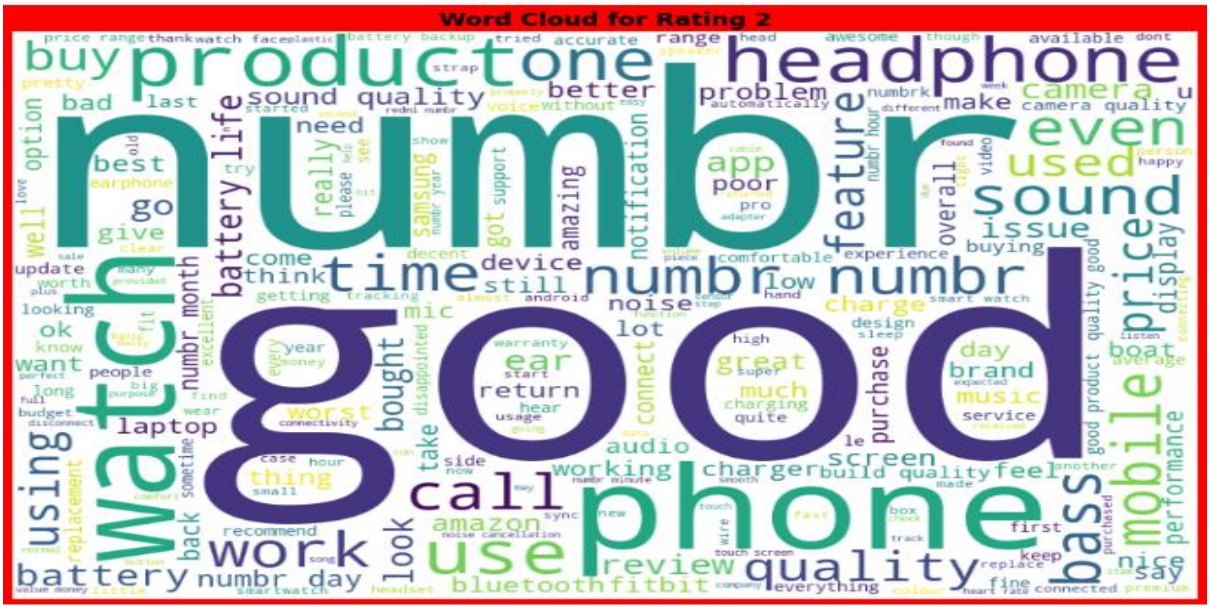
- More than 50% customers have given 5-star rating
- Only 4% customers have given 2-star rating

Word Cloud:

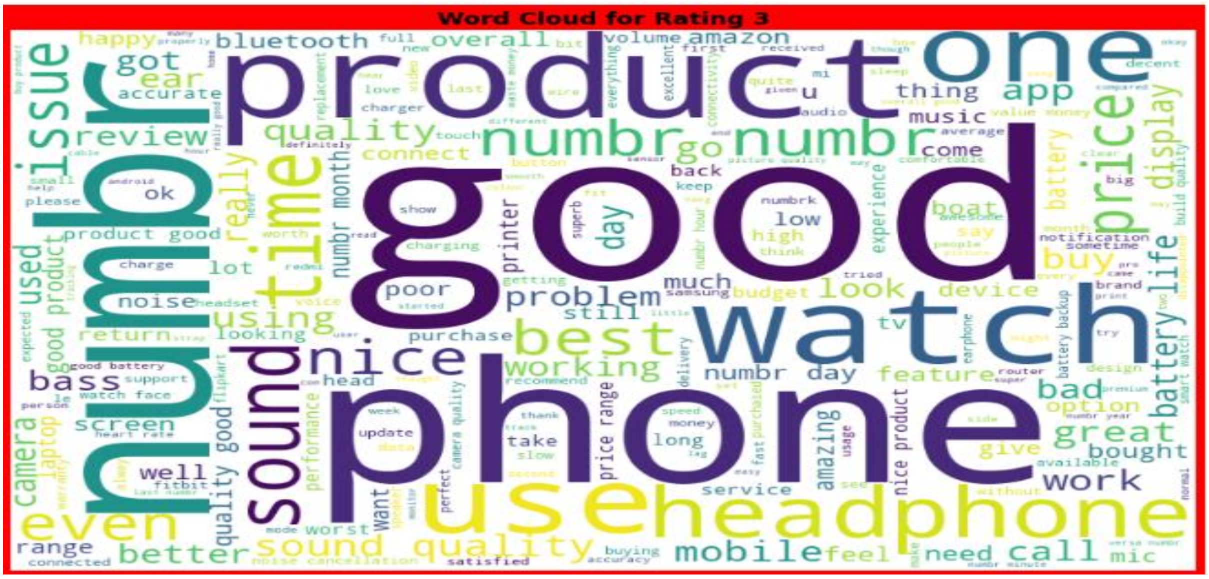
- Word Cloud is a visualization technique for text data wherein each word is picturized with its importance in the context or its frequency.
- The more commonly the term appears within the text being analysed, the larger the word appears in the image generated.
- The enlarged texts are the greatest number of words used there and small texts are the smaller number of words used.



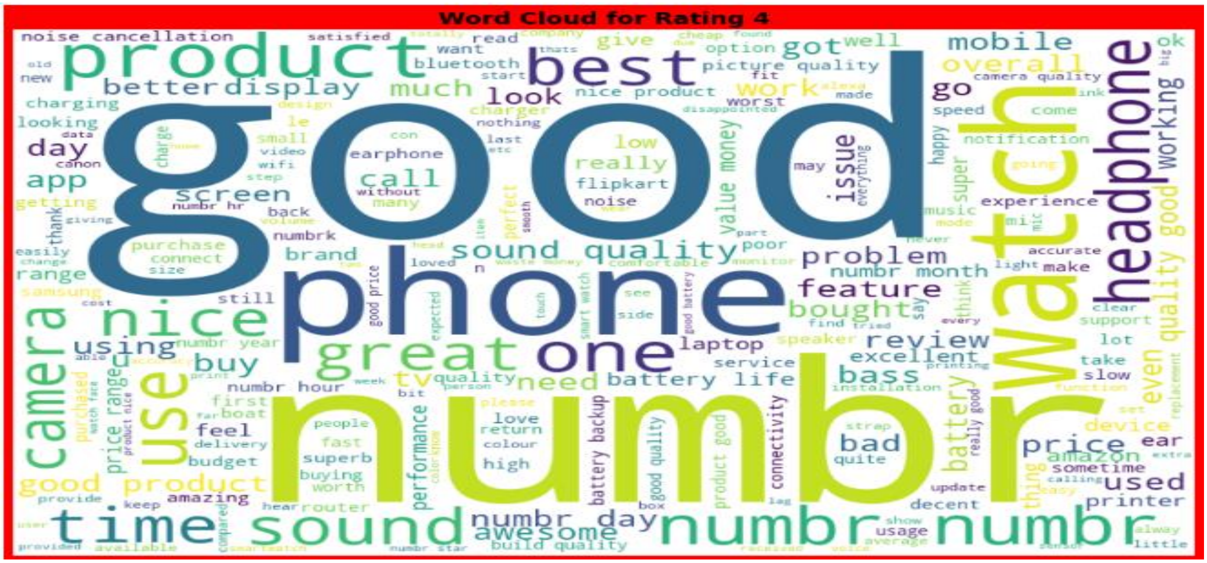
Word Cloud for Rating 1



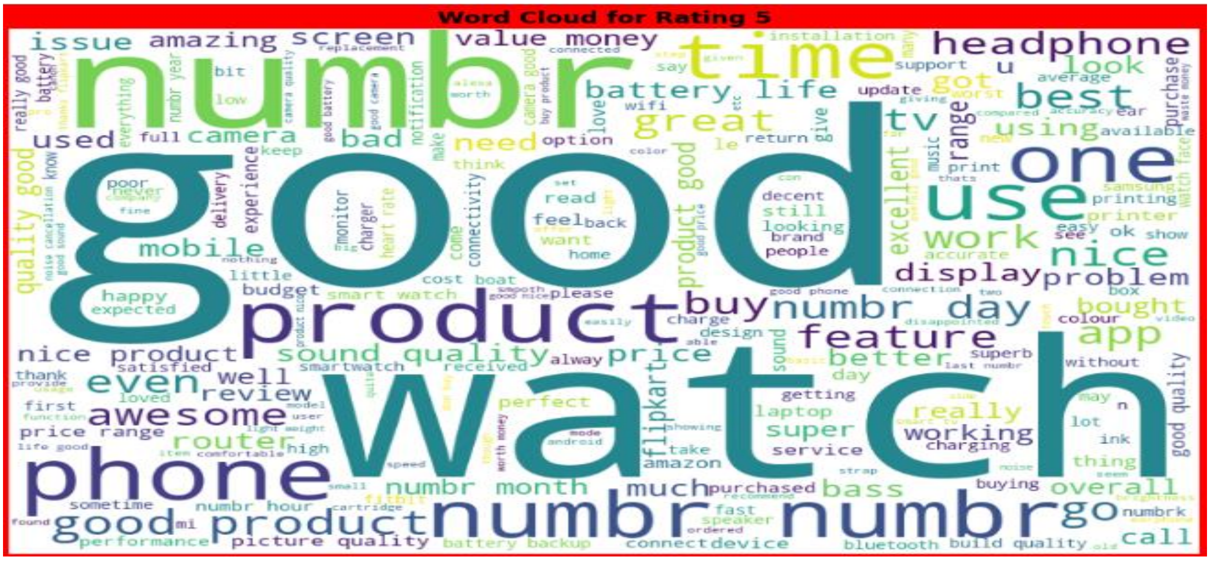
Word Cloud for Rating 2



Word Cloud for Rating 3



Word Cloud for Rating 4

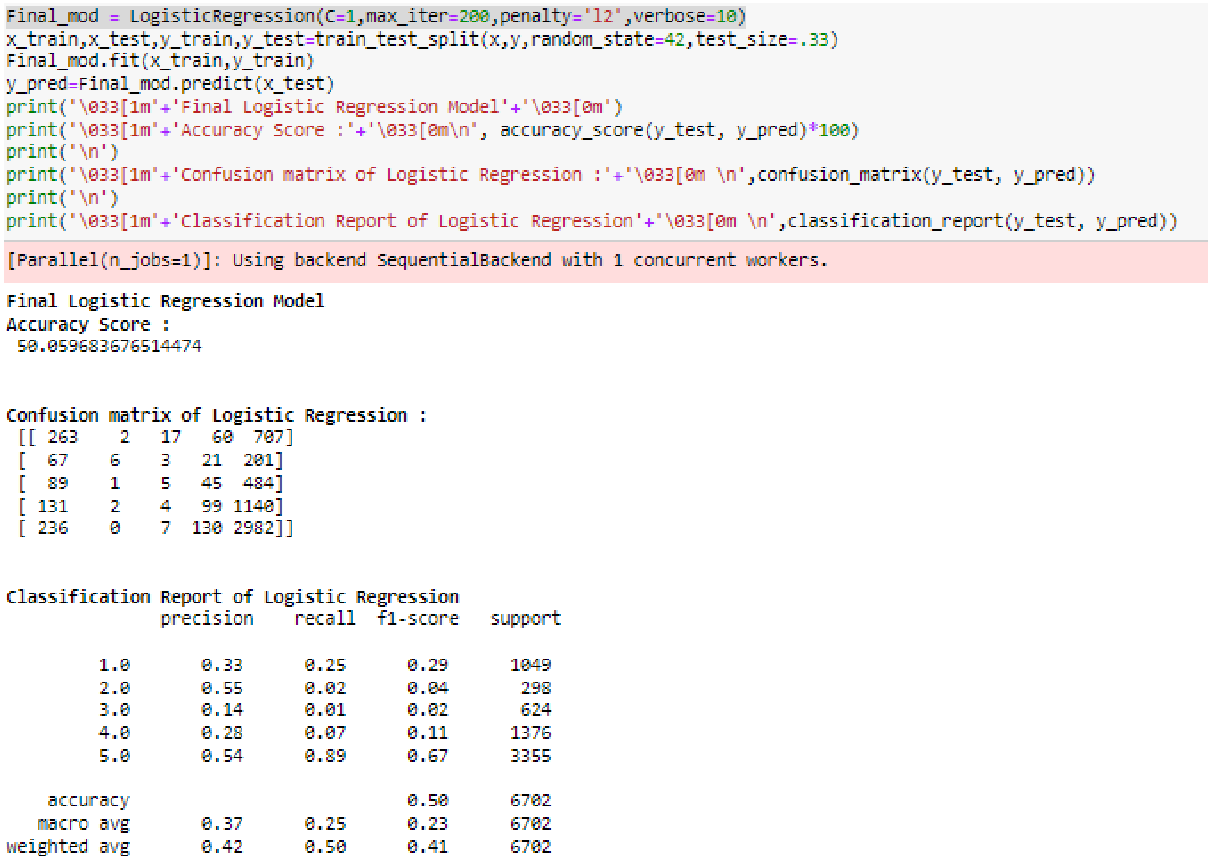


Word Cloud for Rating 5

CONCLUSION

Key Findings and Conclusions of the Study

- Below is the final model:



```
Final_mod = LogisticRegression(C=1,max_iter=200,penalty='l2',verbose=10)
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=42,test_size=.33)
Final_mod.fit(x_train,y_train)
y_pred=Final_mod.predict(x_test)
print('\033[1m'+'Final Logistic Regression Model'+'\033[0m')
print('\033[1m'+'Accuracy Score :'+'\033[0m\n', accuracy_score(y_test, y_pred)*100)
print('\n')
print('\033[1m'+'Confusion matrix of Logistic Regression :'+'\033[0m \n',confusion_matrix(y_test, y_pred))
print('\n')
print('\033[1m'+'Classification Report of Logistic Regression'+'\033[0m \n',classification_report(y_test, y_pred))
```

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
```

```
Final Logistic Regression Model
Accuracy Score :
 50.059683676514474


Confusion matrix of Logistic Regression :
 [[ 263    2   17   60  707]
 [  67    6    3   21  201]
 [  89    1    5   45  484]
 [ 131    2    4   99 1140]
 [ 236    0    7  130 2982]]


Classification Report of Logistic Regression
               precision    recall  f1-score   support

         1.0       0.33      0.25      0.29      1049
         2.0       0.55      0.02      0.04       298
         3.0       0.14      0.01      0.02       624
         4.0       0.28      0.07      0.11      1376
         5.0       0.54      0.89      0.67      3355

    accuracy                           0.50      6702
   macro avg       0.37      0.25      0.23      6702
weighted avg       0.42      0.50      0.41      6702
```

- The accuracy can be low due to some dataset issues.
- One of the reason for such low accuracy can be the imbalanced target variable.
- One can opt for any of the resample methods to fix this issue for a better accuracy.

Learning Outcomes of the Study in respect of Data Science

- In this project we were able to learn various Natural language processing techniques like lemmatization, stemming, removal of Stop words.
- This project has demonstrated the importance of sampling effectively, modelling and predicting data.

Limitations of this work and Scope for Future Work

- More input features can be scrap to build predication model.
- There is scope for application of advanced deep learning NLP tool to enhanced text mining operation which eventually help in building more accurate model with good cross validation score.
- Extensive hyperparameter tuning can result in better model.

References

- https://www.google.com/
- https://towardsdatascience.com/review-rating-prediction-a-combined-approach-538c617c495c
- https://levelup.gitconnected.com/how-to-fine-tune-an-nlp-classification-model-with-transformers-and-huggingface-1a2c0ea79c2
- https://medium.com/data-science-lab-spring-2021/amazon-review-rating-prediction-with-nlp-28a4acdd4352