Rating prediction project

Submitted by:

Vivek kumar singh

ACKNOWLEDGMENT

I take great pleasure to thank and acknowledge the help provided by **Flip Robo Technologies**. I extend whole hearted thanks to Mr. Shwetank Mishra who become my Mentor and with whom I worked and learned a lot and for enlightening me with his knowledge and experience to grow with the corporate working. Her guidance at every stage of the Project enabled me to successfully complete this Project which otherwise would not have been possible without her consent encouragement and motivation. Without the support it was not possible for me to complete the report with fullest endeavour.

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

  With the explosion of online user reviews, review rating prediction has become a research focus in natural language processing. Existing review rating prediction methods only use a single model to capture the sentiments of review texts, ignoring users who express the sentiment and products that are evaluated, both of which have great influences on review rating prediction. In order to solve the issue, we propose a review rating prediction method based on user context and product context by incorporating user information and product information into review texts. Our method firstly models the user context information of reviews, and then models the product context information of reviews. Finally, a review rating prediction method that is based on user context and product context is proposed. Our method consists of three main parts. The first part is a global review rating prediction model, which is shared by all users and all products, and it can be learned from training datasets of all users and all products. The second part is a user-specific review rating prediction model, which represents the user's personalized sentiment information, and can be learned from training data of an individual user. The third part is a product-specific review rating prediction model, which uses training datasets of an individual product to learn parameter of the model. Experimental results on four datasets show that our

proposed methods can significantly outperform the state-of-the-art baselines in review rating prediction.

- ## Review of Literature

  The dataset used for this task is the one available as a part of the flipkart dataset challenge (www.flipkart.com). Most of the predictive tasks previously performed on this dataset have rating predictions primarily based on user and business attributes. However. research has been carried out, not just in the general area of text mining and sentiment analysis, but in text mining for predictive tasks in review and rating systems. The impact of text derived information has been previously studied at the sentence level, with the help of the topic information on various datasets. Various methods have been adopted in the past, including regression, bag of opinions method and classification .In technical product reviews, it has been observed that Naive Bayes' had a slightly better accuracy than the SVM method. However, this was in combination with other features of the dataset. Hence, the results differ from the ones chosen here. The flipkart dataset has been extensively studied as well. Attempts have been made to gauge information from the review text by predicting what the user felt about various aspects of the business, such as service and quality . If the user experience can be divided into various aspects, then a function of these can be used to predict the overall rating. Another approach that has been taken is to classify 3 stars , 4 and 5 stars , in order to gauge the general opinion of the user. However, this falsely increases the accuracy of rating prediction. It is an analysis of the user's sentiment but should not be used for rating prediction tasks. Work has been done to take this a step further as well. If a feature for some customization of a user is included, we can treat the reviews of each user as separate entities. There is expected to be a uniformity in the reviews that a user writes and different users have different ways of expressing the same emotion. Other measures of evaluation such as precision and recall have been used for baseline comparison

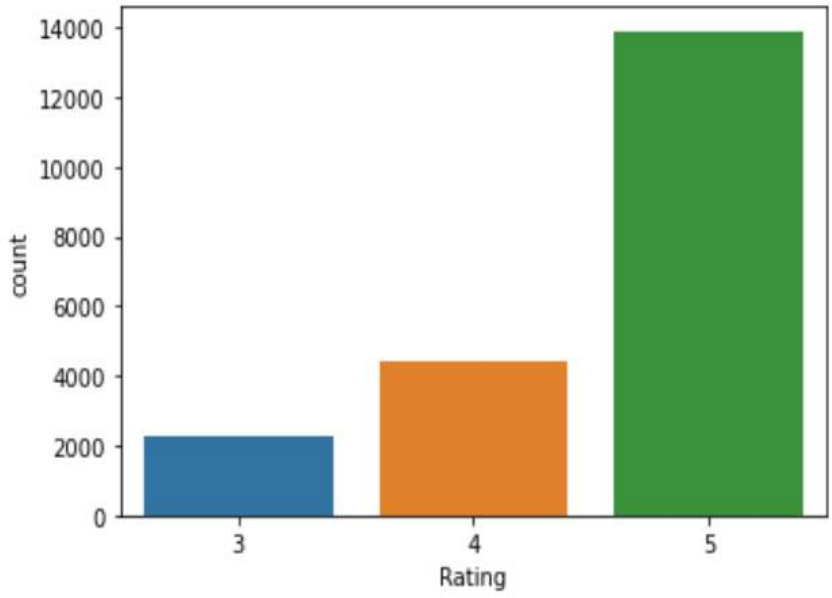
Analytical Problem Framing

- Mathematical/ Analytical Modelling of the Problem

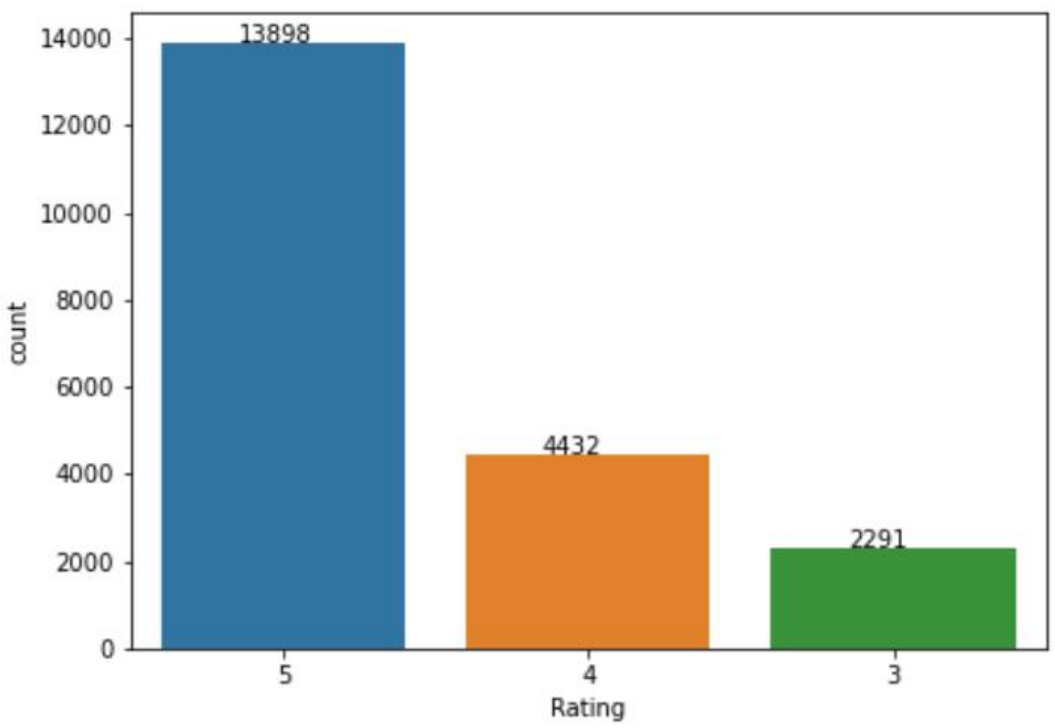
  The dataset has 20621 rows and 2 columns

```
[316]: sb.countplot(df['Rating'])
```

```
:[316]: <AxesSubplot:xlabel='Rating', ylabel='count'>
```



Here , Rating is one of the columns which is target columns. 3 star,4 star, 5 star are the target variable although data is little unbalanced.



5star- 13898

4 star-4432

5 star- 2291

These are the rows which are divided with columns

- Data Sources and their formats

In first step data is scrapped from flipkart.com . Data is scrapped from different types of technical product.Here review prediction data is scrapped from led tv, aneroid phone, I phones, wireless headset and laptops .

All the dataset is scrapped and has two columns

1. Review columns :- text reviews are here , in format of string.

2.Rating columns:- 3star, 4star and 5 star are present, which we use as target variable.

```
10]: df.shape
     # now here we see the there are 20621 rows and 2 columns

10]: (20621, 2)
```

- Data Preprocessing Done
  First we import the data, and then check it has null value or not

```
]: df.isnull().sum()

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

We find out that rating and review data has no null values and we can proceed to next step.
After that we are using some EDA technique for Data visualization.
After that we use some of text cleaning process which are:-

```
: df['Review']=df['Review'].apply(nfx.remove_currency_symbols)
  df['Review']=df['Review'].apply(nfx.remove_emails)
  df['Review']=df['Review'].apply(nfx.remove_emojis)
  df['Review']=df['Review'].apply(nfx.remove_hashtags)
  df['Review']=df['Review'].apply(nfx.remove_non_ascii)
  df['Review']=df['Review'].apply(nfx.remove_punctuations)
  df['Review']=df['Review'].apply(nfx.remove_special_characters)
  df['Review']=df['Review'].apply(nfx.remove_numbers)
  df['Review']=df['Review'].apply(nfx.remove_phone_numbers)
```

```
: df['Review']=df['Review'].apply(nfx.remove_stopwords)
```

These are the done to clean the review text which is filled with punctuation, special character, numbers, phone numbers, emojis , emails, hashtags , currency symbols.To clean all this data I used the NEAT TEXT package of python

Before cleaning the text we see,

```
[321]: df['before_cleaning']=df['Review'].map(lambda Review :len(Review) )

[322]: df.head()
```

[322]:

	Rating	Review	before_cleaning
0	5	Well while switching from android to iOS the f...	457
1	5	awesome phone to have. it has got many smart f...	243
2	5	I can say I'm damn impressed with iPhone 11. A...	510
3	5	I am using this phone for 5 days and its one o...	422
4	5	After 1 month use I found camera quality best ...	213

Here, I added one columns to show how much the word are present before the cleaning the text.

```
In [333]: df['after_cleaning']=df['Review'].map(lambda Review :len(Review))

In [334]: df
```

Out[334]:

	Rating	Review	before_cleaning	after_cleaning
0	5	switching android iOS thing need careful loose...	457	274
1	5	awesome phone got smart features apart mind bl...	243	153
2	5	Im damn impressed iPhone says iPhone toughest ...	510	296
3	5	phone days best camera screen oled coming ipho...	422	214
4	5	month use found camera quality best compared p...	213	154

Again I added one more columns after cleaning the text it shows how much text is removed and how much is available for the prediction.

- ## Hardware and Software Requirements and Tools Used

  The project is done into laptop with i5 processor with quad core with 8gb of RAM with GTX 1650 GPU with Anaconda and juypter notebook.

  Some of important libraries are also used which are :-

```
In [305]: # Importing the important packages:-
          import pandas as pd
          import numpy as np
          import matplotlib.pyplot as plt
          import seaborn as sb
          from nltk import word_tokenize,sent_tokenize,regexp_tokenize
          from nltk.stem.wordnet import WordNetLemmatizer
          from nltk.stem import PorterStemmer,LancasterStemmer,SnowballStemmer
          from nltk.stem.wordnet import WordNetLemmatizer
          from nltk.corpus import stopwords
          import string
          import warnings
          warnings.filterwarnings('ignore')
          import wordcloud
          from PIL import Image
          from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
```

  Here, are some major tool which are used for visualization and some are for pre-processing. There are also some of tool imported for prediction , like creating models

```
In [348]: from sklearn.svm import LinearSVC
          from sklearn.naive_bayes import MultinomialNB
          from sklearn.linear_model import LogisticRegression
          from lightgbm import LGBMClassifier
          from sklearn.linear_model import SGDClassifier
          from sklearn.ensemble import RandomForestClassifier

          from sklearn.model_selection import train_test_split, GridSearchCV
          from sklearn.metrics import f1_score,precision_score, multilabel_confusion_matrix, accuracy_score,jaccard_score, recall_score,
          from sklearn.multiclass import OneVsRestClassifier
          from sklearn.model_selection import cross_val_score
```

Above libraries imported for creating the models. All these packages are the algorithm which is used for prediction.

Before creating the models we vectorized the models , imported libraries

```
: from sklearn.feature_extraction.text import TfidfVectorizer
tfidf=TfidfVectorizer(max_features = 14000, stop_words='english')
```

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

Problem solving approaches are :-

- Removing special character
- Removing emojis and emails
- Removing punctuation
- Removing stop words
- Removing numbers.
- Using lemmatize and stemming
- Using vectorizer (tfidf vectorizer) for numerical values
- Splitting train and test data
- And at the end create models
- Check the accuracy
- Use cross validation for overfitting and underfitting
- Saving the best performance model

- Testing of Identified Approaches (Algorithms)

  The following algorithm are used for testing: -

```
]: from sklearn.svm import LinearSVC
   from sklearn.naive_bayes import MultinomialNB
   from sklearn.linear_model import LogisticRegression
   from lightgbm import LGBMClassifier
   from sklearn.linear_model import SGDClassifier
   from sklearn.ensemble import RandomForestClassifier
```

```
svc = LinearSVC()
lr = LogisticRegression(solver='lbfgs')
mnb = MultinomialNB()
lgb = LGBMClassifier()
sgd = SGDClassifier()
rf = RandomForestClassifier()
```

  These are the algorithm used for testing approachs.

- Run and Evaluate selected models

```
]: def print_score(y_pred,clf):
       print('classifier:',clf)
       print("Jaccard score: {}".format(jaccard_score(y_test,y_pred,average='micro')))
       print("Accuracy score: {}".format(accuracy_score(y_test,y_pred)))
       print("f1_score: {}".format(f1_score(y_test,y_pred,average='micro')))
       print("Precision : ", precision_score(y_test,y_pred,average='micro'))
       print("Recall: {}".format(recall_score(y_test,y_pred,average='micro')))
       print("Hamming loss: ", hamming_loss(y_test,y_pred))
       print("Confusion matrix:\n ", multilabel_confusion_matrix(y_test,y_pred))
       print('========================================\n')
```

```
0]: for classifier in [svc,lr,mnb,sgd,lgb,rf]:
        clf = OneVsRestClassifier(classifier)
        clf.fit(x_train,y_train)
        y_pred = clf.predict(x_test)
        print_score(y_pred, classifier)
```

```
classifier: LinearSVC()
Jaccard score: 0.9219455366244357
Accuracy score: 0.9593877860281861
f1_score: 0.9593877860281861
Precision :  0.9593877860281861
Recall: 0.9593877860281861
Hamming loss:  0.040612213971813 91
Confusion matrix:
  [[[5866    0]
  [   0  733]]

 [[5181    0]
  [ 268 1150]]

 [[1883  268]
  [   0 4448]]]
=========================================
```

```
classifier: LogisticRegression()
Jaccard score: 0.9219455366244357
Accuracy score: 0.9593877860281861
f1_score: 0.9593877860281861
Precision :  0.9593877860281861
Recall: 0.9593877860281861
Hamming loss:  0.040612213971813 91
Confusion matrix:
  [[[5866    0]
  [   0  733]]

 [[5181    0]
  [ 268 1150]]

 [[1883  268]
  [   0 4448]]]
=========================================
```

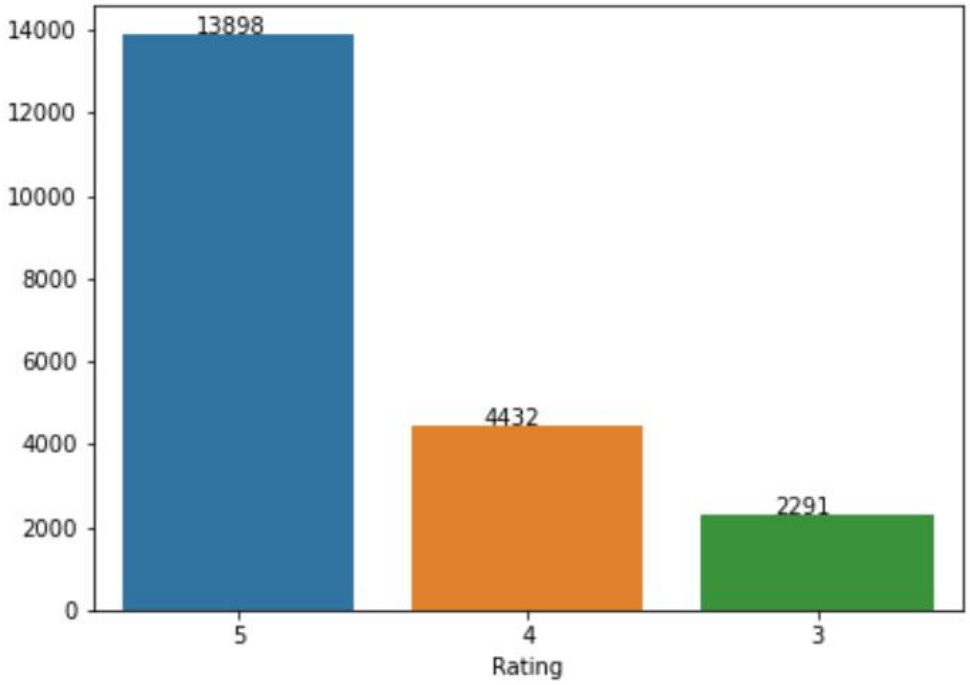
```
classifier: MultinomialNB()
Jaccard score: 0.9219455366244357
Accuracy score: 0.9593877860281861
f1_score: 0.9593877860281861
Precision :  0.9593877860281861
Recall: 0.9593877860281861
Hamming loss:  0.040612213971813 91
Confusion matrix:
  [[[5866    0]
  [   0  733]]

 [[5181    0]
  [ 268 1150]]

 [[1883  268]
  [   0 4448]]]
========================================
```

```
classifier: SGDClassifier()
Jaccard score: 0.9219455366244357
Accuracy score: 0.9593877860281861
f1_score: 0.9593877860281861
Precision :  0.9593877860281861
Recall: 0.9593877860281861
Hamming loss:  0.040612213971813 91
Confusion matrix:
  [[[5866    0]
  [   0  733]]

 [[5181    0]
  [ 268 1150]]

 [[1883  268]
  [   0 4448]]]
========================================
```

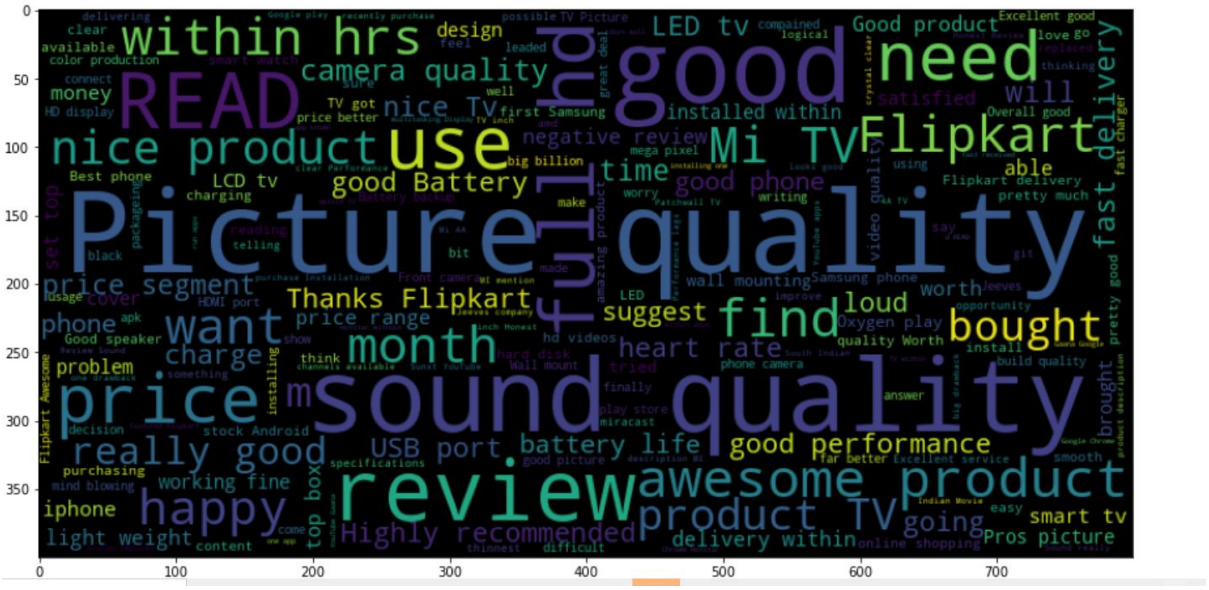
```
classifier: LGBMClassifier()
Jaccard score: 0.9219455366244357
Accuracy score: 0.9593877860281861
f1_score: 0.9593877860281861
Precision :  0.9593877860281861
Recall: 0.9593877860281861
Hamming loss:  0.04061221397181391
Confusion matrix:
  [[[5866    0]
  [   0  733]]

 [[5181    0]
  [ 268 1150]]

 [[1883  268]
  [   0 4448]]]
========================================
 classifier: RandomForestClassifier()
 Jaccard score: 0.9219455366244357
 Accuracy score: 0.9593877860281861
 f1_score: 0.9593877860281861
 Precision :  0.9593877860281861
 Recall: 0.9593877860281861
 Hamming loss:  0.04061221397181391
 Confusion matrix:
   [[[5866    0]
   [   0  733]]

  [[5181    0]
   [ 268 1150]]

  [[1883  268]
   [   0 4448]]]
 ========================================
```

- Key Metrics for success in solving problem under consideration

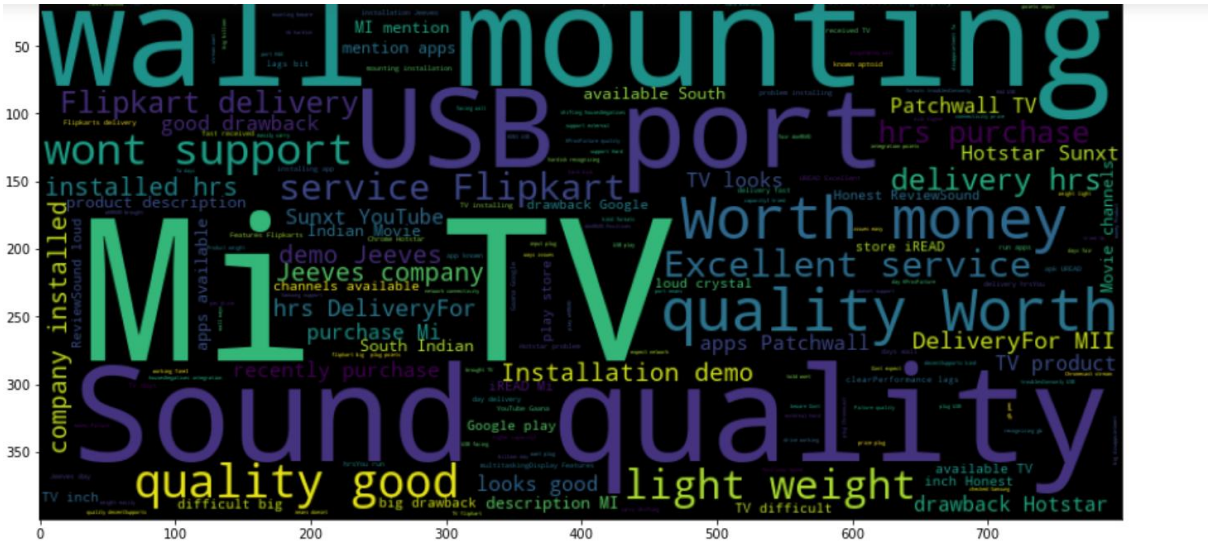
  In basis of accuracy score, matrix and cross validation. Logistic regression selected as final models.
- Visualizations



.

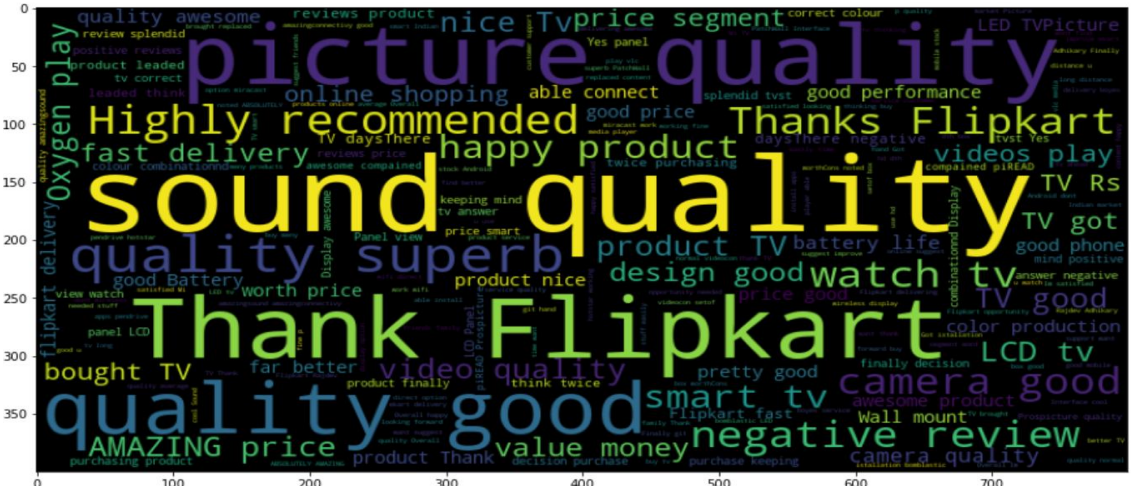


Before pre-processing how feature columns look like



Here,rating == 3, how the review looks like ,in feature columns

Here, rating ==4 looks like , in feature columns



Here, Rating ==5 look like in feature columns

- Interpretation of the Results

We interpreted the result logistic regression give best accuracy .

```
classifier: LogisticRegression()
Jaccard score: 0.9219455366244357
Accuracy score: 0.9593877860281861
f1_score: 0.9593877860281861
Precision :  0.9593877860281861
Recall: 0.9593877860281861
Hamming loss:  0.040612213971813 91
Confusion matrix:
  [[[5866    0]
  [   0  733]]

 [[5181    0]
  [ 268 1150]]

 [[1883  268]
  [   0 4448]]]
========================================
```

Here the accuracy is 95 % which quite good. Also comparing with other as score card given below shows random forest also quite effective model and shows as same accuracy score. As per I preferred to select the logistic regression.

	model	acc_score	Jaccard score	f1_score	Hamming loss	cross val score
0	linearSVC	0.959388	0.921946	0.910138	0.040612	0.904646
1	logistic	0.959388	0.921946	0.910138	0.040612	0.904646
2	multinomial	0.959388	0.912754	0.910138	0.040612	0.880396
3	sdg	0.959388	0.912754	0.910138	0.040612	0.904646
4	LGBM	0.959388	0.921946	0.910138	0.040612	0.904646
5	Random forest	0.959388	0.921946	0.910138	0.040612	0.904646

CONCLUSION

- ## Key Findings and Conclusions of the Study

  In this project we see that the commercial e commerce website uses the rating system for rate its product as we are here working on technical product, we see that most of the rating we have done classified into 3 categories. We made use of natural language processing and create the models. We select and save best model fitted. As per here logistic regression is best selected model.

- ## Learning Outcomes of the Study in respect of Data Science

  Through this project we are able to learn Various Natural language processing technique such as lemmatization and stemming, removal of stop word
  This project has shown us importance of sampling, modelling and prediction of data.
  With the help of different types of tool, we use to clean the dataset, cleaning null values. And also vectorized the text data into numeric data.
  With the help of tfidf we use vectorizer.
  Also, we use here cross validation for analysing the model is overfitted or not.

- ## Limitations of this work and Scope for Future Work

  The models were trained on a highly imbalanced dataset where the total malignant comments formed only 10% of the entire available data, which seriously affected the training and accuracy of the models. By training the models on more diverse data sets, longer comments, and a more balanced dataset, more accurate and efficient classification models can be built