FLIP ROBO

RATING PREDICTION

Submitted by:

Amritesh Kumar.

Acknowledgement

I would like to express my gratitude to my guide Shubham Yadav (SME, Flip Robo) for his constant guidance, continuous encouragement and unconditional help towards the development of the project. It was he who helped me whenever I got stuck somewhere in between. The project would have not been completed without his support and confidence he showed towards me.

Lastly, I would l like to thank all those who helped me directly or indirectly toward the successful completion of the project.

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. As a Data Scientist, we have to apply our analytical skills to give findings and conclusions in detailed data analysis written in Jupyter notebook.

Conceptual Background of the Domain Problem

Created model can be used to predict ratings of reviews, it might be a good tool for online shopping sites such as Flipkart.com, Amazon.in etc and manufacturer companies who might look into their product ratings and reviews so that they can make their investment according to demand of customers , which might help them to save time and earn more profits.

Review of Literature

Now a days people buy more products from online sites, So by using this model, rating prediction and positive or negative review prediction of a product is easy and less time consuming.

Motivation for the Problem Undertaken

Data Science help us to make predictions at areas like health sectors, auto industry, education, media etc. For our project we decided to implement Prediction of Rating of reviews. We have to create a model which will help online sites and manufacturer of a product if they have to modify their product or not according to customers requirement

Analytical Problem Framing

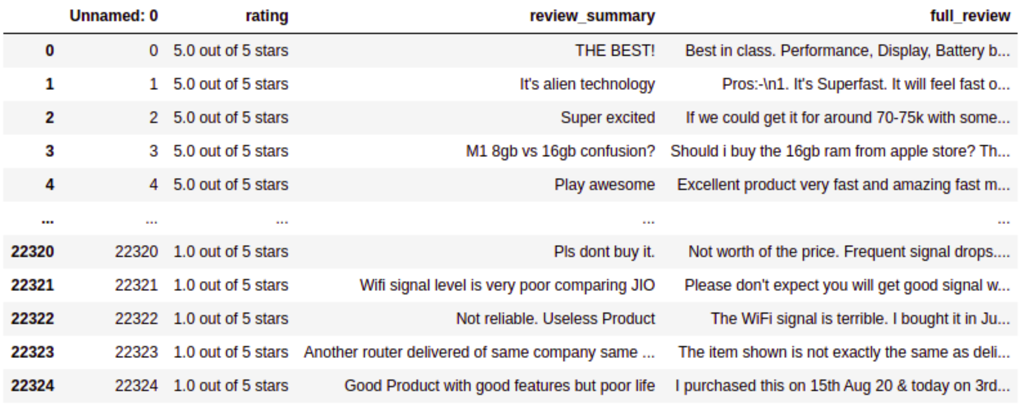
Mathematical/ Analytical Modelling of the Problem

We would perform one type of supervised learning algorithms: classification. While it seems more reasonable to perform classification since we have 5 types of Rating i.e., 1, 2, 3, 4, 5. The prediction will base on the dataset which is scrapped from Flipkart and Amazon technical product. Scrapping dataset contains reviews and ratings of different products.

Data Sources and their formats

We scraped more than 20000 rows of data. We can scrape more data as well. More the data better the model. In this section we need to scrape the reviews of different laptops, Phones, Headphones, smart watches, Professional Cameras, Printers, Monitors, Home theatre, Router from different e-commerce websites. Basically, we need these columns
1) reviews of the product.
2) rating of the product. We fetched other data as well. Our columns are –
1. Rating
2. Review
3. Long Review

2]:



	Unnamed: 0	rating	review_summary	full_review
0	0	5.0 out of 5 stars	THE BEST!	Best in class. Performance, Display, Battery b...
1	1	5.0 out of 5 stars	It's alien technology	Pros:-\n1. It's Superfast. It will feel fast o...
2	2	5.0 out of 5 stars	Super excited	If we could get it for around 70-75k with some...
3	3	5.0 out of 5 stars	M1 8gb vs 16gb confusion?	Should i buy the 16gb ram from apple store? Th...
4	4	5.0 out of 5 stars	Play awesome	Excellent product very fast and amazing fast m...
...	...	...	...	...
22320	22320	1.0 out of 5 stars	Pls dont buy it.	Not worth of the price. Frequent signal drops....
22321	22321	1.0 out of 5 stars	Wifi signal level is very poor comparing JIO	Please don't expect you will get good signal w...
22322	22322	1.0 out of 5 stars	Not reliable. Useless Product	The WiFi signal is terrible. I bought it in Ju...
22323	22323	1.0 out of 5 stars	Another router delivered of same company same ...	The item shown is not exactly the same as deli...
22324	22324	1.0 out of 5 stars	Good Product with good features but poor life	I purchased this on 15th Aug 20 & today on 3rd...

Data Pre-processing Done

Cleaned the data from junk values. Replace multiple spaces with single space So that it'll easy to classify it.

Hardware and Software Requirements and Tools Used

The system requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software product. The requirements can be obvious or hidden, known or unknown, expected or unexpected from client's point of view. System requirements are all of the requirements at the system level that describe the functions which the system as a whole should fulfil to satisfy the stakeholder needs and requirements, and is expressed in an appropriate combination of textual statements, views, and non-functional requirements; the latter expressing the levels of safety, security, reliability, etc., that will be necessary.

Hardware requirements: -

1. Processor — core i5 and above
2. RAM — 8 GB or above
3. SSD — 250GB or above

Software requirements: -

Anaconda

Libraries: -

From sklearn.preprocessing import StandardScaler

As these columns are different in **scale**, they are **standardized** to have common **scale** while building machine learning model. This is useful when you want to compare data that correspond to different units.

from sklearn.preprocessing import Label Encoder

Label Encoder and One Hot Encoder. These two encoders are parts of the SciKit Learn library in Python, and they are used to convert categorical data, or text data, into numbers, which our predictive models can better understand.

From sklearn.model_selection import train_test_split,cross_val_score

Train_test_split is a function in Sklearn model selection for splitting data arrays into two subsets: for training data and for testing data. With this function, you don't need to divide the dataset manually. By default, Sklearn train_test_split will make random partitions for the two subsets.

The algorithm is trained and tested K times, each time a new set is used as testing set while remaining sets are used for training. Finally, the result of the K-Fold Cross-Validation is the average of the results obtained on each set.

from sklearn.neighbors import KNeighborsClassifier

K Nearest Neighbor(KNN) is a very simple, easy to understand, versatile and one of the topmost machine learning algorithms. KNN used in the variety of applications such as finance, healthcare, political science, handwriting detection, image recognition and video recognition

from sklearn.linear_model import LogisticRegression

The library sklearn can be used to perform logistic regression in a few lines as shown using the LogisticRegression class. It also supports multiple features. It requires the input values to be in a specific format hence they have been reshaped before training using the fit method.

from sklearn.tree import DecisionTreeClassifier

Decision Tree is a white box type of ML algorithm. It shares internal decision-making logic, which is not available in the black box type of algorithms such as Neural Network. Its training time is faster compared to the neural network algorithm. The time complexity of decision trees is a function of the number of records and number of attributes in the given data. The decision tree is a distribution-free or non-parametric method, which does not depend upon probability distribution assumptions. Decision trees can handle high dimensional data with good accuracy.

Model/s Development and Evaluation

Identification of possible problem-solving approaches (methods)

Just make the comments more appropriate so that we’ll get less word to process and get more accuracy. Removed extra spaces, converted email address into email keyword, likely wise phone number etc. Tried to make Comments small and more appropriate as much as it was possible.

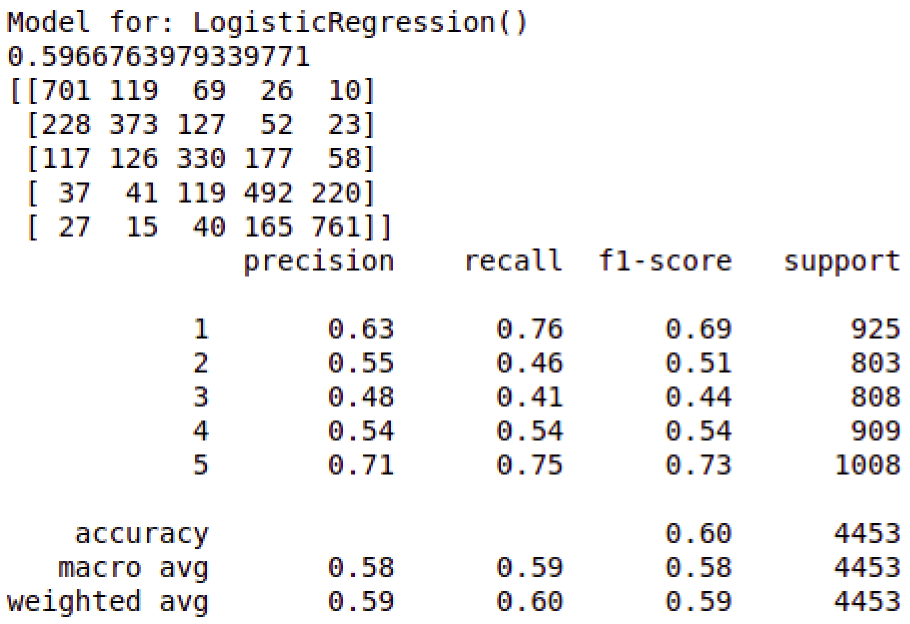
Testing of Identified Approaches (Algorithms)

Listing down all the algorithms used for the training and testing.

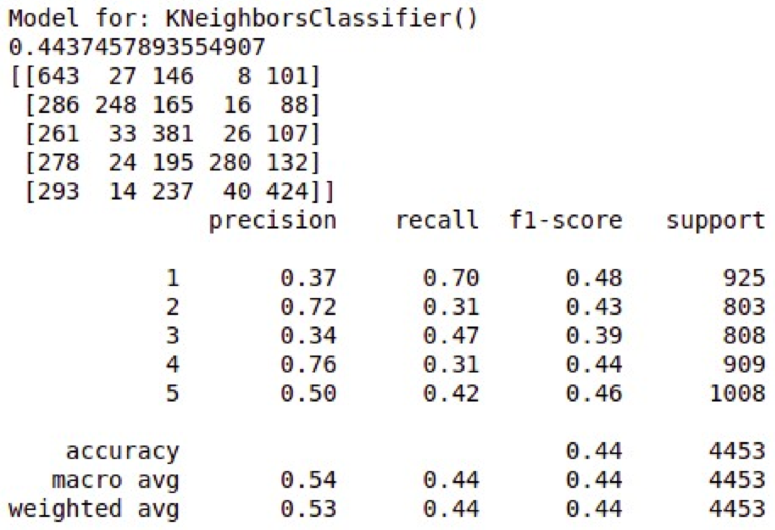
- KNeighborsClassifier()
- LogisticRegression()
- BernoulliNB()
- DecisionTreeClassifier()
- RandomForestClassifier()

I applied all these algorithms in the dataset.

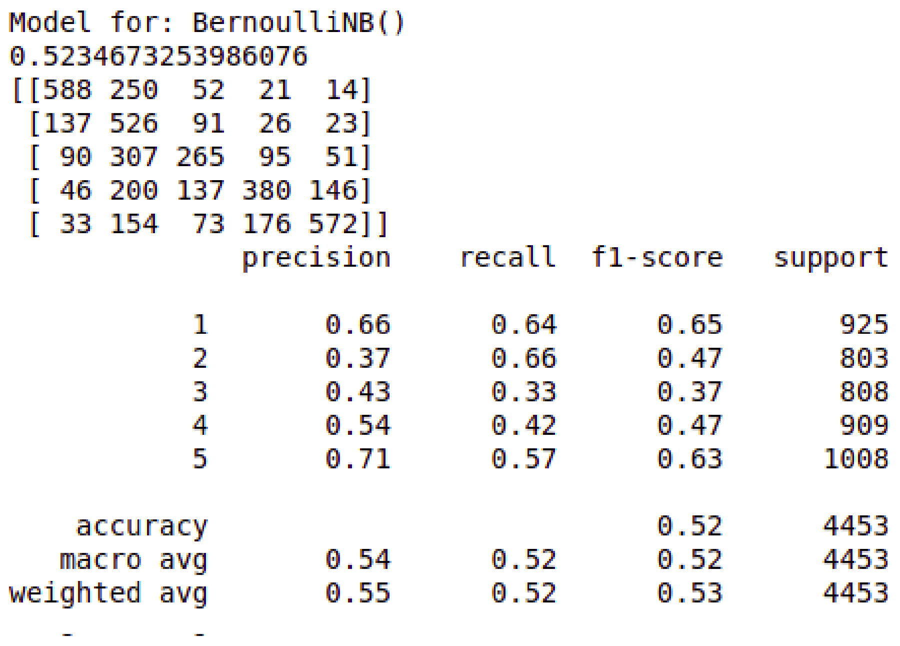
Run and Evaluate selected models



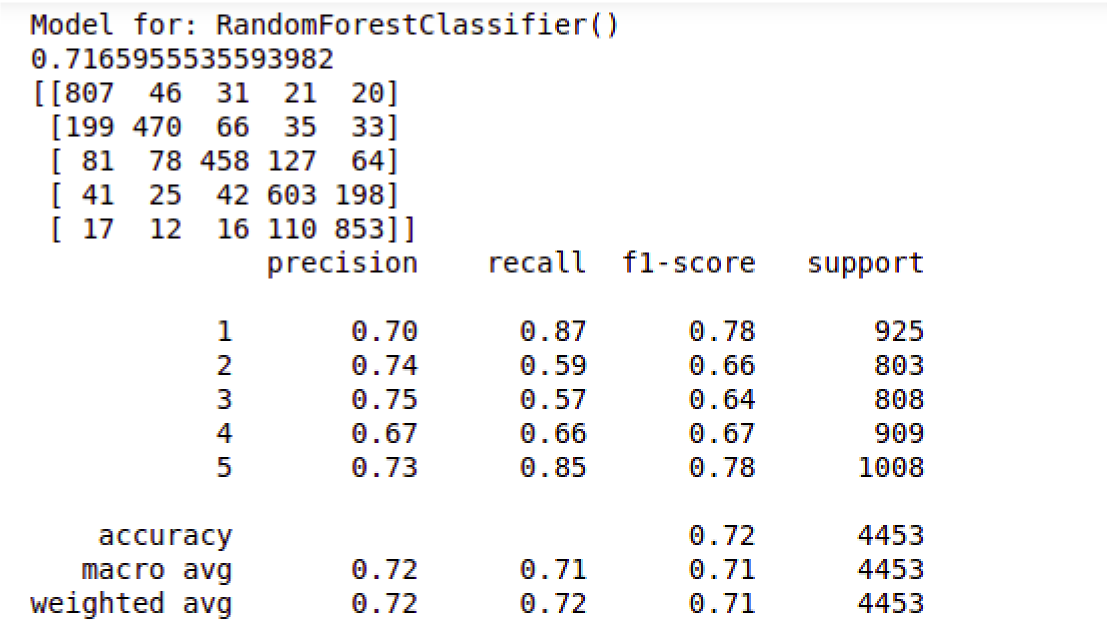
```
Model for: LogisticRegression()
0.5966763979339771
[[701 119  69  26  10]
 [228 373 127  52  23]
 [117 126 330 177  58]
 [ 37  41 119 492 220]
 [ 27  15  40 165 761]]
              precision    recall  f1-score   support

           1       0.63      0.76      0.69       925
           2       0.55      0.46      0.51       803
           3       0.48      0.41      0.44       808
           4       0.54      0.54      0.54       909
           5       0.71      0.75      0.73      1008

    accuracy                           0.60      4453
   macro avg       0.58      0.59      0.58      4453
weighted avg       0.59      0.60      0.59      4453
```



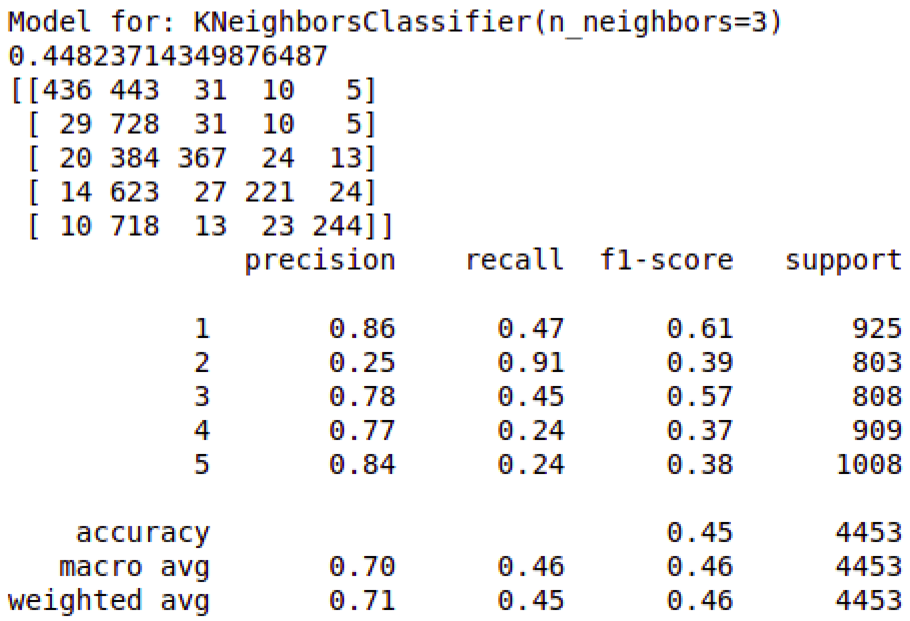
```
Model for: KNeighborsClassifier()
0.4437457893554907
[[643  27 146   8 101]
 [286 248 165  16  88]
 [261  33 381  26 107]
 [278  24 195 280 132]
 [293  14 237  40 424]]
              precision    recall  f1-score   support

           1       0.37      0.70      0.48       925
           2       0.72      0.31      0.43       803
           3       0.34      0.47      0.39       808
           4       0.76      0.31      0.44       909
           5       0.50      0.42      0.46      1008

    accuracy                           0.44      4453
   macro avg       0.54      0.44      0.44      4453
weighted avg       0.53      0.44      0.44      4453
```



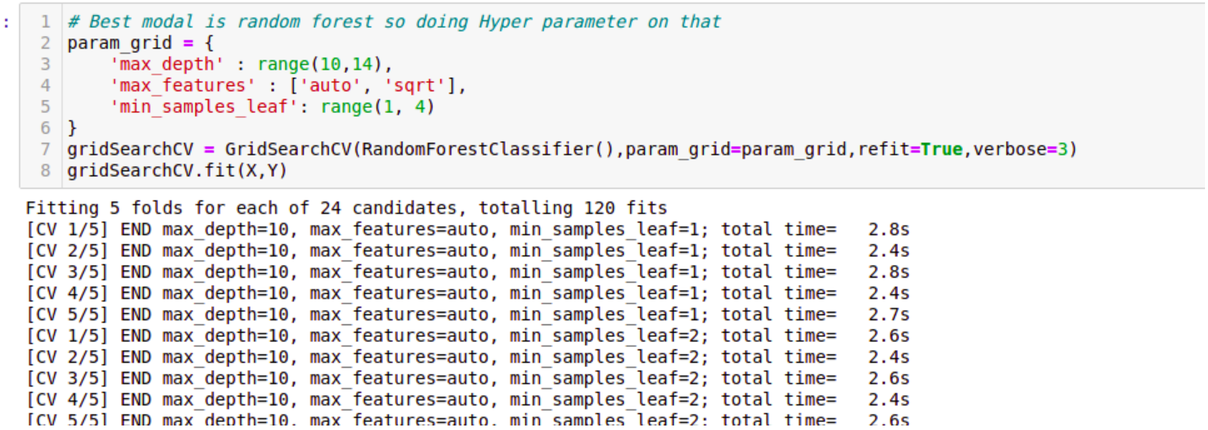
```
Model for: BernoulliNB()
0.5234673253986076
[[588 250  52  21  14]
 [137 526  91  26  23]
 [ 90 307 265  95  51]
 [ 46 200 137 380 146]
 [ 33 154  73 176 572]]
              precision    recall  f1-score   support

           1       0.66      0.64      0.65       925
           2       0.37      0.66      0.47       803
           3       0.43      0.33      0.37       808
           4       0.54      0.42      0.47       909
           5       0.71      0.57      0.63      1008

    accuracy                           0.52      4453
   macro avg       0.54      0.52      0.52      4453
weighted avg       0.55      0.52      0.53      4453
   -       -
```



```
Model for: RandomForestClassifier()
0.7165955535593982
[[807  46  31  21  20]
 [199 470  66  35  33]
 [ 81  78 458 127  64]
 [ 41  25  42 603 198]
 [ 17  12  16 110 853]]
              precision    recall  f1-score   support

           1       0.70      0.87      0.78       925
           2       0.74      0.59      0.66       803
           3       0.75      0.57      0.64       808
           4       0.67      0.66      0.67       909
           5       0.73      0.85      0.78      1008

    accuracy                           0.72      4453
   macro avg       0.72      0.71      0.71      4453
weighted avg       0.72      0.72      0.71      4453
```



```
Model for: KNeighborsClassifier(n_neighbors=3)
0.44823714349876487
[[436 443  31  10   5]
 [ 29 728  31  10   5]
 [ 20 384 367  24  13]
 [ 14 623  27 221  24]
 [ 10 718  13  23 244]]
              precision    recall  f1-score   support

           1       0.86      0.47      0.61       925
           2       0.25      0.91      0.39       803
           3       0.78      0.45      0.57       808
           4       0.77      0.24      0.37       909
           5       0.84      0.24      0.38      1008

    accuracy                           0.45      4453
   macro avg       0.70      0.46      0.46      4453
weighted avg       0.71      0.45      0.46      4453
```



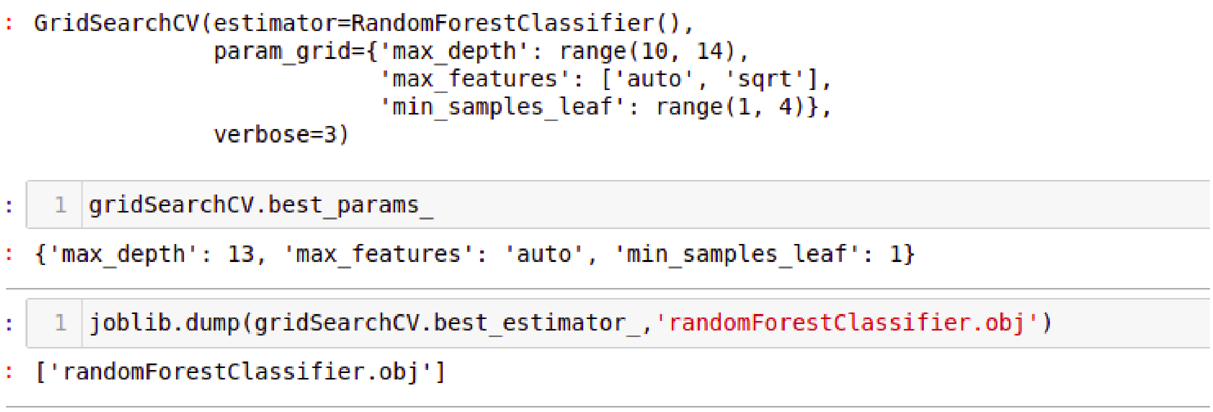
```python
# Best modal is random forest so doing Hyper parameter on that
param_grid = {
    'max_depth' : range(10,14),
    'max_features' : ['auto', 'sqrt'],
    'min_samples_leaf': range(1, 4)
}
gridSearchCV = GridSearchCV(RandomForestClassifier(),param_grid=param_grid,refit=True,verbose=3)
gridSearchCV.fit(X,Y)
```

```
Fitting 5 folds for each of 24 candidates, totalling 120 fits
[CV 1/5] END max_depth=10, max_features=auto, min_samples_leaf=1; total time=   2.8s
[CV 2/5] END max_depth=10, max_features=auto, min_samples_leaf=1; total time=   2.4s
[CV 3/5] END max_depth=10, max_features=auto, min_samples_leaf=1; total time=   2.8s
[CV 4/5] END max_depth=10, max_features=auto, min_samples_leaf=1; total time=   2.4s
[CV 5/5] END max_depth=10, max_features=auto, min_samples_leaf=1; total time=   2.7s
[CV 1/5] END max_depth=10, max_features=auto, min_samples_leaf=2; total time=   2.6s
[CV 2/5] END max_depth=10, max_features=auto, min_samples_leaf=2; total time=   2.4s
[CV 3/5] END max_depth=10, max_features=auto, min_samples_leaf=2; total time=   2.6s
[CV 4/5] END max_depth=10, max_features=auto, min_samples_leaf=2; total time=   2.4s
[CV 5/5] END max_depth=10, max_features=auto, min_samples_leaf=2; total time=   2.6s
```

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.



```
: GridSearchCV(estimator=RandomForestClassifier(),
               param_grid={'max_depth': range(10, 14),
                           'max_features': ['auto', 'sqrt'],
                           'min_samples_leaf': range(1, 4)},
               verbose=3)
```

```
: gridSearchCV.best_params_
```

```
: {'max_depth': 13, 'max_features': 'auto', 'min_samples_leaf': 1}
```

```
: joblib.dump(gridSearchCV.best_estimator_,'randomForestClassifier.obj')
```

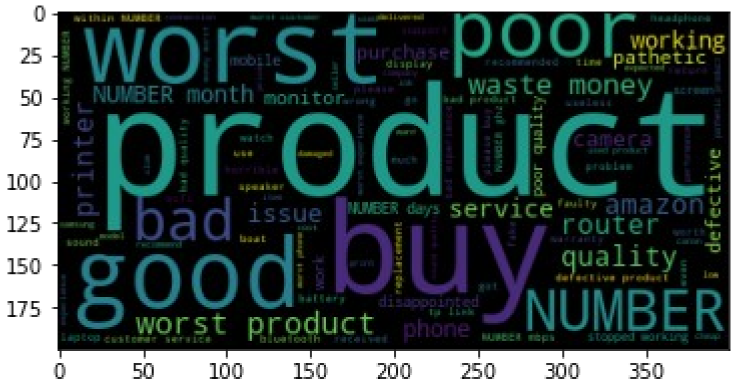
```
: ['randomForestClassifier.obj']
```

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

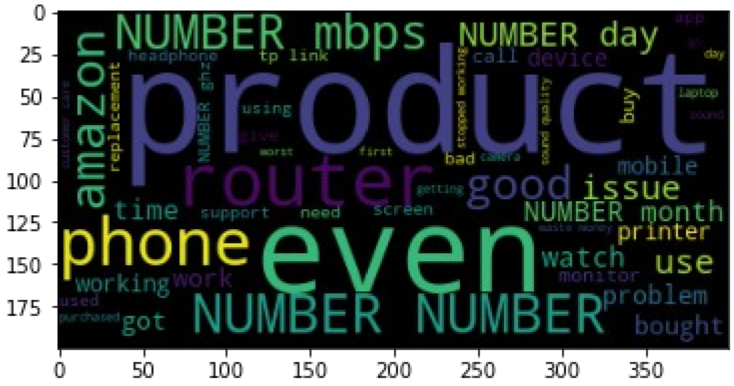
Visualizations

Word Clouds -

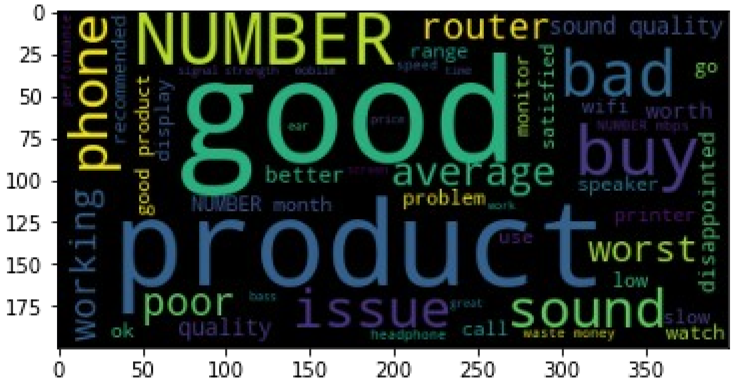
One Ratting(Review Title)



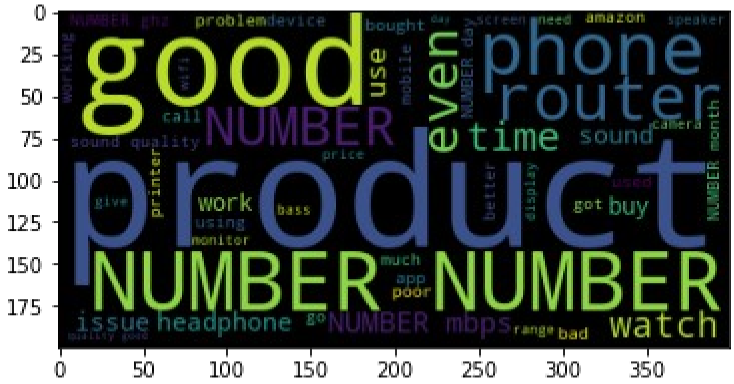
One Ratting(Review Desc)



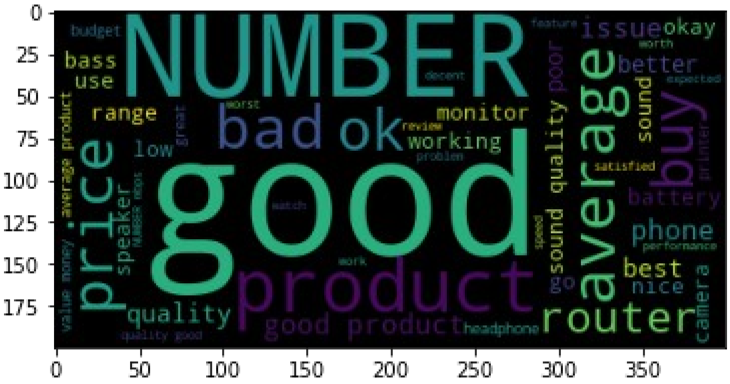
For 2 rattings (Review Title)



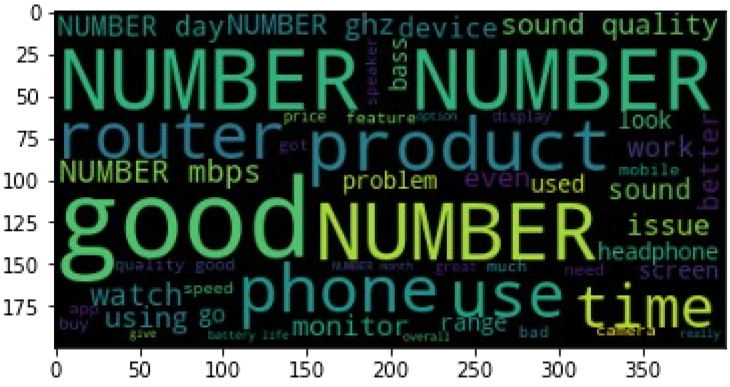
For 2 Ratting (Review Desc)



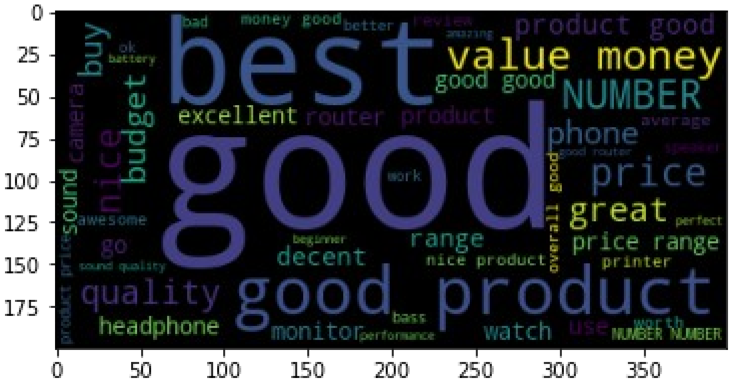
For 3 rattings(For Title)



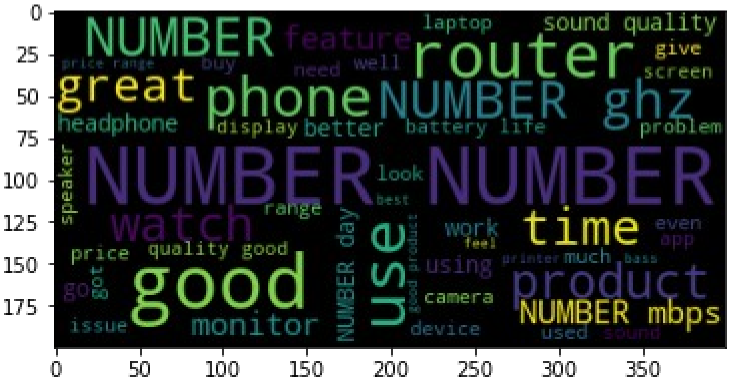
For 3 Ratting (For Desc)



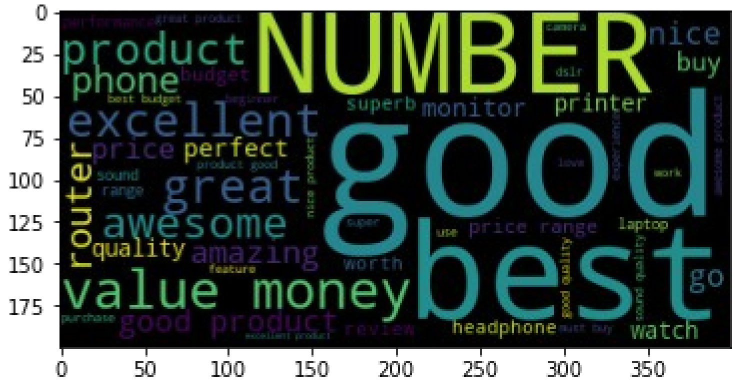
For 4 rattings(For title)



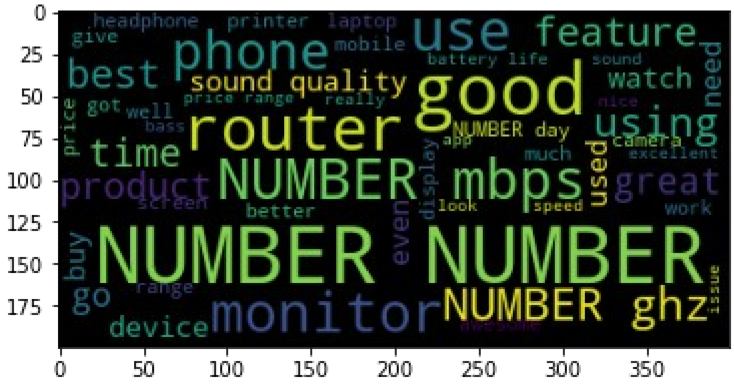
For 4 Ratting (For Desc)



For 5 rattings(For title)



For 5 rattings(For Review)



Threat

Abuse

Interpretation of the Results

From Random Forest Classifier R2 score was 71.64%,

Higher the R2 score means the model is well fit for the data. However, if R2 score is very high, it might be a case of overfitting. Other metrics Mean Absolute Error, Mean Squared Error and Root Mean Squared Error, with gradient boosting these scores are less then compared to other models. If these errors are less that means the model shows less errors.

Conclusion

Key Findings and Conclusions of the Study

First, we scrap Reviews and Ratings data of different technical products from flipkart.com using selenium web driver. After scrapping, we save this dataset in a csv file named “flipkartreviews.csv”. Then we pre-process data by cleaning duplicates, noise and unnecessary words which are not helpful for this project. Then we convert text into vectors by using tfidfvectorizer as we know our ml model understands only integers. We choose our best random_state. Then we build our model. Though our dataset is imbalanced, we clearly see that f1 score of SVC is good compared to all other algorithms. So, we check croos_val_score of this model to check if it is overfitting or not. We confirm and conclude that this is the best model. We performed hyperparameter tuning by using GridSearchCV. We save the model by using joblib

Limitations of this work and Scope for Future Work

As we know data is increasing in every second in our day today life. So more the data better the model. If we make this dataset for sentiment analysis, we choose ratings 3 or more as our threshold for being helpful reviews or good or positive reviews and below 3 we choose reviews is not helpful or bad reviews or negative reviews. For example: two people give their reviews on a product as ‘a nice product. But their ratings are ‘4’ and ‘5’ respectively. This is where the model fails to predict whether to choose rating ‘4’ or’5’ Due to increase in data in our daily basics, this model can be used to predict ratings of reviews. It might be a good tool for online shopping sites and manufacturer companies who may predict their customers ratings so that they can make their investment according to the demand of customers, which might help them to save time and earn more profits.