FLIP ROBO

Project Report On

“Reviews Ratings Prediction”

Submitted by:

NEERAJ KUMAR

ACKNOWLEDGMENT

I would like to express my sincere thanks of gratitude to present this report on "Ratings Prediction" project. Working on this project was a good experience that has given me a basic knowledge about Machine Learning Model with NLP. This project also helped me in doing lots of research wherein I came to know about so many new things.

At the commencement of this project report, I would like to evince my deepest sense of gratitude to SME. Without his guidance, insightful decision, valuable comments and corrections it would not have possible to reach up to this mark.

I would like to draw my gratitude to Flip Robo Technologies and Data Trained for providing me a suitable environment and guidance to complete my work.

Finally, I would like to thank my family and friends who have helped me with their valuable suggestions and guidance and have been very helpful in various stages of project completion. Last but not the least thanks to the brilliant authors from where I have got the idea to carry out the project.

1. INTRODUCTION

Modern E-commerce websites contain heterogeneous sources of information, such as numerical ratings, textual reviews and images. This information can be utilized to assist recommendation. Through textual reviews, a user explicitly expresses her/his affinity towards the item. E-commerce is the daily essential of our life now-a-days. E-commerce is the platform where a seller can sell products and customer can buy the products. That means e-commerce platform connects large number of sellers to customers through online. Providing better services to customers is one of the main keys to be successful as an e-commerce seller.

1.1 Business Problem Framing

In the present system, the e-commerce platform will send a feedback mail to customers after the product is delivered. The customers can give ratings out of 5, also can write down some comments/reviews about the product that he/she has purchased. Using these reviews and ratings, e-commerce platform will rate the products, which helps other people to get the insights about the quality of the product. But many times, customers would not give any ratings or reviews. How to predict the review score that a customer could give? This is the problem in e-commerce business.

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. Use of the user's similarity matrix and applying neighbours' analysis are all part of this method. This method ignores any information from the review text content analysis.

Business Goal: 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.

1.2 Conceptual Background of the Domain Problem

The rise in online shopping has brought a significant rise in the importance of textual customer reviews. There are thousands of review sites online and massive amounts of reviews for each and every product. Nowadays customers have changed their way of shopping. 60-70 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, flipkart and Yelp!

1.3 Review of Literature

The ability to successfully decide whether a review will be helpful to other customers and thus give the product more exposure is important to companies that support these reviews, companies like Flipkart, Amazon and Yelp! The customer will make a decision to buy a product if he or she sees valuable reviews posted by others, especially the user's trusted friend. We believe reviews and reviewers will do help to the rating prediction based on the idea that 5-star ratings may greatly be attached with extremely good reviews. It's also agreed that different people may have different sentimental expression preferences. For example, some users prefer to use "good" to describe an "excellent" product, while others may prefer to use "good" to describe a "just so so" product. User's rating information is not always available on many review websites. Here we will predict the rating of the textual reviews based on the Machine Learning approach. This problem can also be referred to as a text classification problem when solved by machine learning approach.

1.4 Motivation for the Problem Undertaken

The main objective of this study is 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. Many products 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 in turn improve customer experience.

The project which is given by Flip Robo as a part of the internship programme which gives an insight to identify major factors that led to get reviews ratings for different products from different e-commerce websites. The exposure to real world data and the opportunity to deploy my skillset in solving a real time problem has been the primary objective. However, the motivation for taking this project was that it is relatively a new field of research. Here we have many options but less concrete solutions. The main motivation is to classify and predict the ratings for the reviews which were written in the past and they don't have a rating and make a good model which will help us to know and decide to buy the product online.

2. ANALYTICAL PROBLEM FRAMING

2.1 Mathematical/ Analytical Modelling of the Problem:

We need to develop an efficient and effective Machine Learning model which predicts the ratings for the reviews which were written in the past and they don't have a rating. So, "Ratings" is our target variable which is textual in nature. The rating is out 5 stars and it only has 5 options available 1 star, 2 stars, 3 stars, 4 stars, 5 stars. Clearly it is a Multiclass Classification Problem where we need to use classification algorithms to predict the results.

It is a Multiclass Classification Problem as the target columns giving 5 outputs and all independent variables has text so it is clear that it is a supervised machine learning problem where we can use the techniques of NLP and classification-based algorithms of Machine Learning. Here we will use NLP techniques like word tokenization, lemmatization, stemming and TFIDF vectorizer then those processed data will be used to create best model using various classification based supervised ML algorithms like LinearSVC, SGD Classifier, XGB Classifier, Random Forest Classifier, Decision Tree Classifier and Extra Trees Classifier. This project is done on 2 phases:

- **Data Collection Phase:** I have done web scraping to collect the data of customers review ratings from the well-known websites https://www.flipkart.com/ and https://www.amazon.com/. In this section I have scraped the reviews of different laptops, Phones, Headphones, smart watches, Professional Cameras, Printers, Monitors, Home theatre, Router from the above 2 mentioned e-commerce websites.

- **Model Building Phase:** After collecting the data, we need to build a machine learning model. Before model building should do all data pre-processing steps involving NLP. Try different models with different hyper parameters and select the best model.

2.2 Data Sources and their formats

I have collected the dataset from the websites https://www.amazon.com/ www.flipkart.com, which is a web platform where the people can purchase their products. The data is scraped using Web scraping technique and the framework used is Selenium. We scrapped and fetched the data for different products like laptops, smart watches, phones etc. and saved the collected data in excel format. The dimension of the dataset is 77551 rows and 3 columns including target variable "Ratings". The data description is as follows:

- Review_title: Title of the review
- Review_text: Content of the review text
- Ratings : Ratings out of 5 stars

2.3 Data Pre-processing Done

Data pre-processing is the process of converting raw data into a well-readable format to be used by Machine Learning model. Data pre-processing is an integral step in Machine Learning as the quality of data and the useful information that can be derived from it directly affects the ability of our model to learn; therefore, it is extremely important that we pre-process our data before feeding it into our model. I have used following pre-processing steps:

- Importing necessary libraries and loading dataset as a data frame.
- Checked some statistical information like shape, number of unique values present, info, null values, value counts, etc.
- Checked null values and filled them using imputation technique (mode method).
- Done feature engineering and created new columns viz original_length, and Full_review_word_count.
- Done text pre-processing techniques like
    - Text case conversion
    - Removing Punctuations and other special characters
    - Splitting the comments into individual words
    - Removing Stop Words

  - Stemming and Lemmatization
  - Text Standardization-Normalization
- Then created new column as clean_length after cleaning the data. Removed skewness in numerical columns.
- 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.
- Balanced the data using SMOTE method.

2.4 Data Inputs- Logic- Output Relationships

The dataset consists of label and features. The features are independent and label is dependent as the values of our independent variables changes as our label varies. Here the label is "Ratings" which has multiclass.

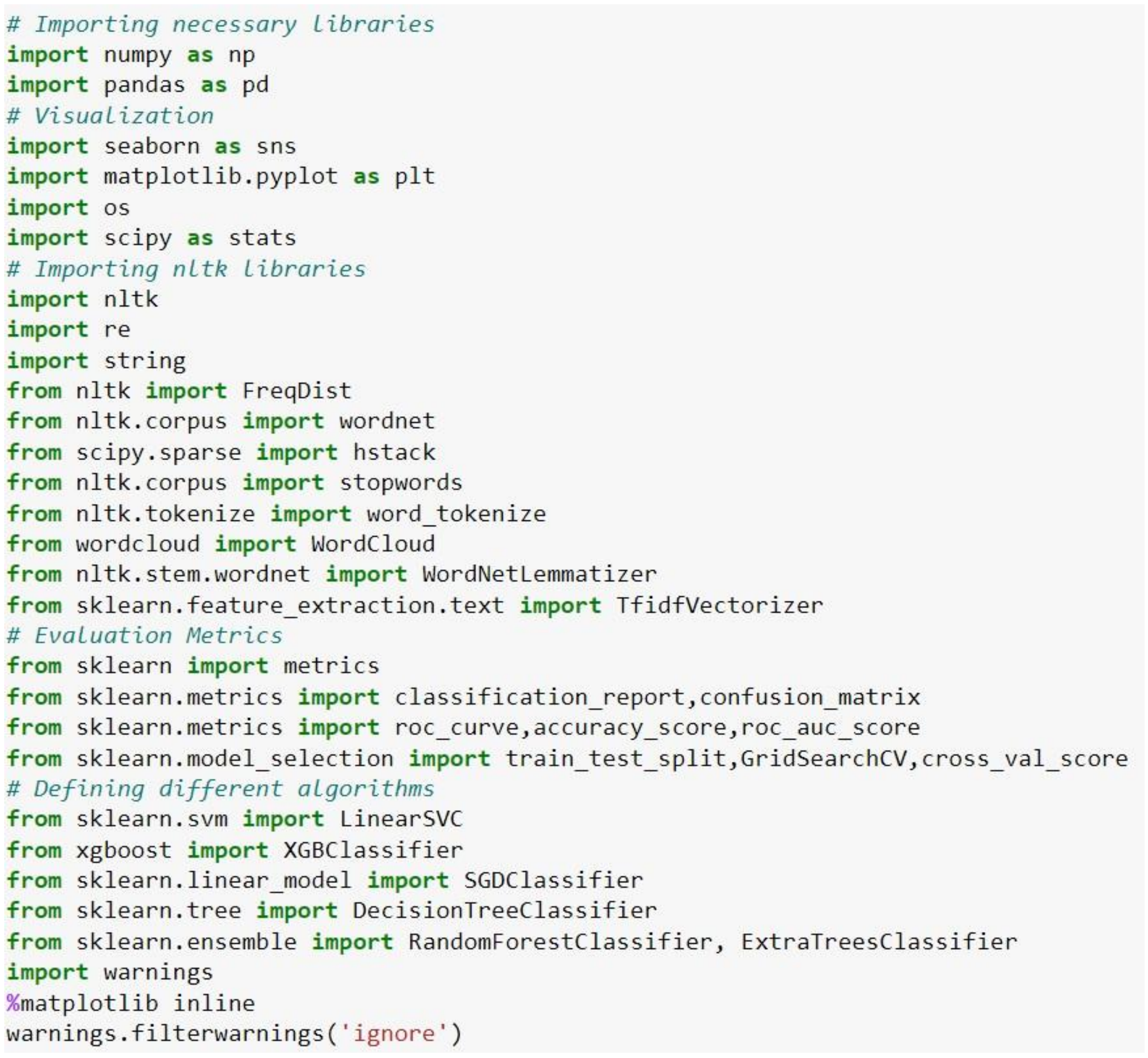
- I checked the distribution of skewness using dist plots and used count plot and pie chart to check the counts available in label column.
- Plotted WordCloud for each rating that is 1 star, 2 stars, 3 stars, 4 stars and 5 stars to get the most frequently used words in the reviews.
- I have checked the correlation between the label and features using heat map.

2.5 Hardware & Software Requirements & Tools Used

To build the machine learning projects it is important to have the following hardware and software requirements and tools.

Hardware	Processor: core i3 RAM: 4 GB ROM/HDD: 512
Software	Distribution: Anaconda Navigator Programming language: Python Browser based language shell: Jupyter Notebook Chrome: To scrape the data

Libraries required:



```python
# Importing necessary libraries
import numpy as np
import pandas as pd
# Visualization
import seaborn as sns
import matplotlib.pyplot as plt
import os
import scipy as stats
# Importing nltk libraries
import nltk
import re
import string
from nltk import FreqDist
from nltk.corpus import wordnet
from scipy.sparse import hstack
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from wordcloud import WordCloud
from nltk.stem.wordnet import WordNetLemmatizer
from sklearn.feature_extraction.text import TfidfVectorizer
# Evaluation Metrics
from sklearn import metrics
from sklearn.metrics import classification_report,confusion_matrix
from sklearn.metrics import roc_curve,accuracy_score,roc_auc_score
from sklearn.model_selection import train_test_split,GridSearchCV,cross_val_score
# Defining different algorithms
from sklearn.svm import LinearSVC
from xgboost import XGBClassifier
from sklearn.linear_model import SGDClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, ExtraTreesClassifier
import warnings
%matplotlib inline
warnings.filterwarnings('ignore')
```

- ✓ **import numpy as np:** It is defined as a Python package used for performing the various numerical computations and processing of the multidimensional and single dimensional array elements. The calculations using Numpy arrays are faster than the normal Python array.
- ✓ **import pandas as pd:** Pandas is a Python library that is used for faster data analysis, data cleaning and data pre-processing. The data-frame term is coming from Pandas only.
- ✓ **import matplotlib.pyplot as plt:** Matplotlib and Seaborn acts as the backbone of data visualization through Python.

  Matplotlib: It is a Python library used for plotting graphs with the help of other libraries like Numpy and Pandas. It is a powerful tool for visualizing data in Python. It is used for creating statical interferences and plotting 2D graphs of arrays.

- ✓ **import seaborn as sns: Seaborn** is also a Python library used for plotting graphs with the help of Matplotlib, Pandas, and Numpy. It is built on the roof of Matplotlib and is considered as a superset of the Matplotlib library. It helps in visualizing univariate and bivariate data.

With the above sufficient libraries, we can perform pre-processing and data cleaning and model building.

3. MODEL/S DEVELOPMENT AND EVALUATION

3.1 Identification of possible Problem-solving approaches (Methods):

In this NLP based project we need to predict the ratings based on reviews given by customers. The Ratings column contains 5 classes hence it is multiclass classification problem. I have collected data from flipkart website.

I have converted text into feature vectors using TF-IDF vectorizer and separated our feature and labels. Also, before building the model, I made sure that the input data is cleaned, balanced and scaled before it was fed into the machine learning models.

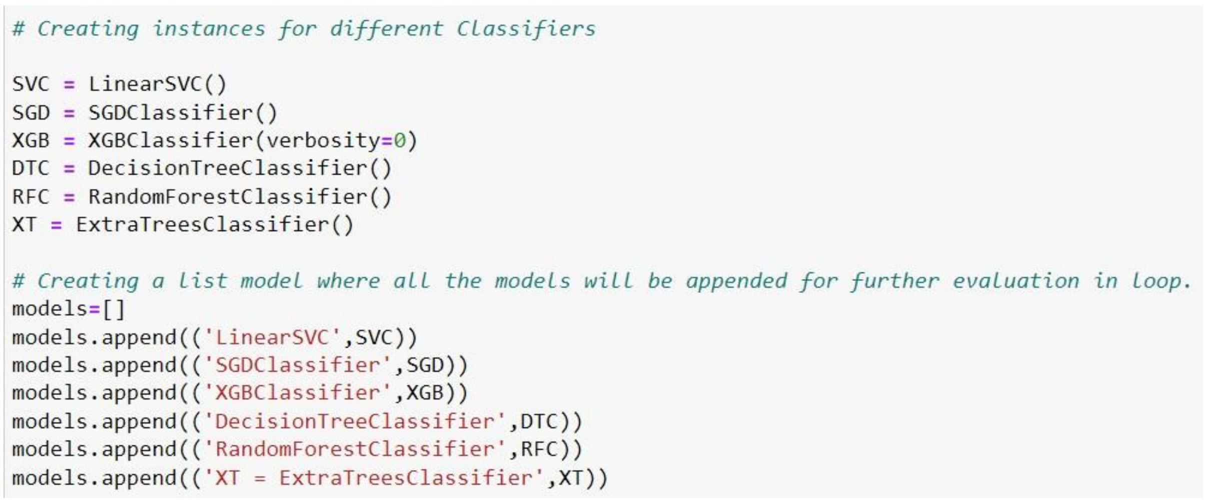
3.2 Testing of Identified Approaches (Algorithms)

Since the target variable is categorical in nature, from this I can conclude that it is a classification type problem hence I have used following classification algorithms.

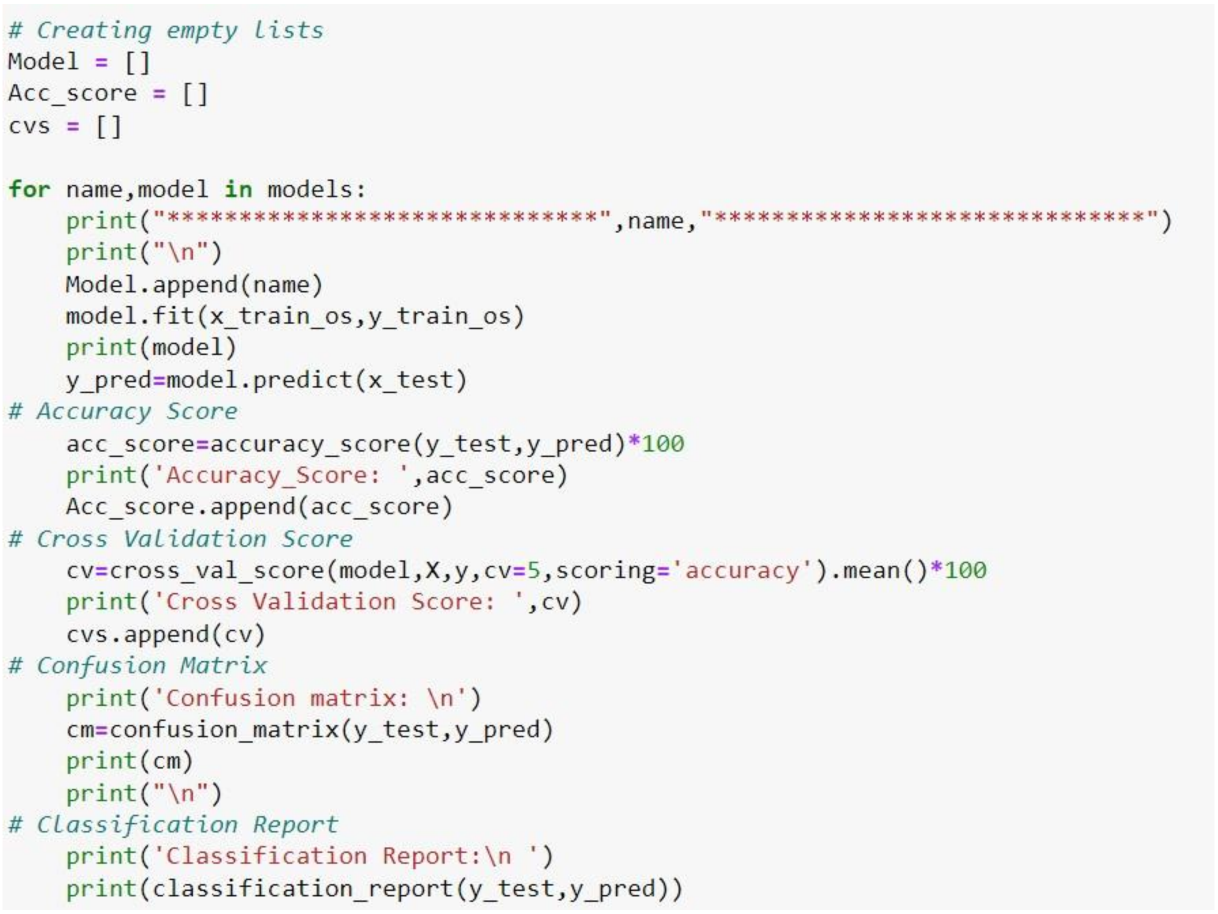
1. LinearSVC
2. SGD Classifier
3. Extreme Gradient Boosting Classifier (XGB)
4. Decision Tree Classifier
5. Random Forest Classifier
6. Extra Trees Classifier

3.3 Run and evaluate selected models

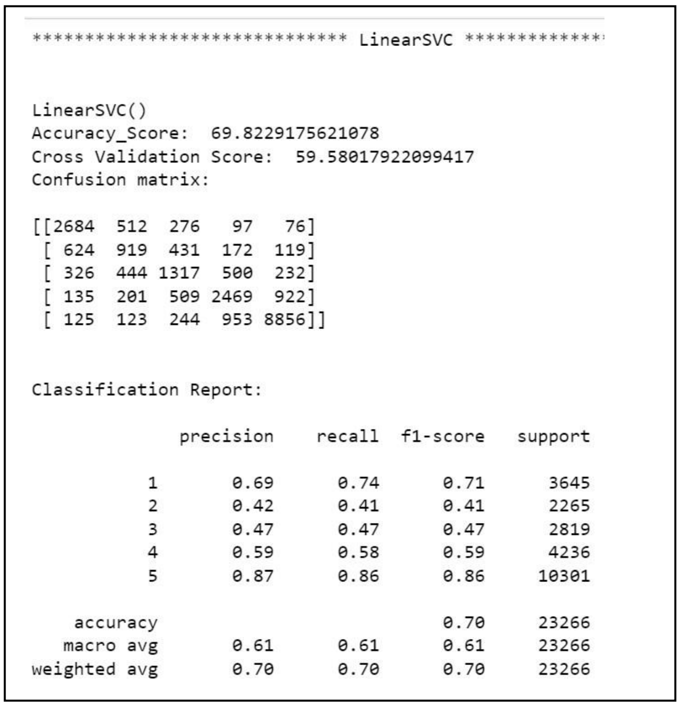
I have used 6 classification algorithms after choosing random state as 67. First, I have created 6 different classification algorithms and are appended in the variable called models. Then, ran a for loop which contained the accuracy of the models along with different evaluation metrics.



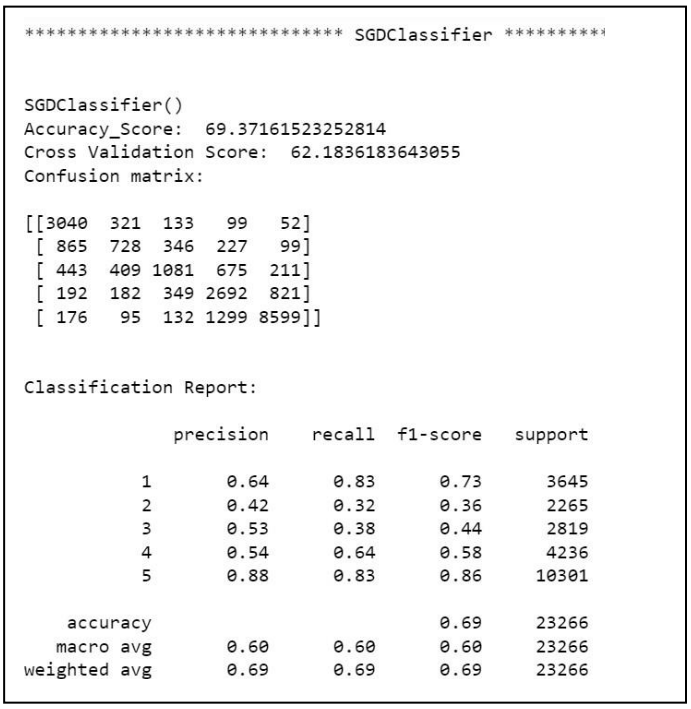
```
# Creating instances for different Classifiers

SVC = LinearSVC()
SGD = SGDClassifier()
XGB = XGBClassifier(verbosity=0)
DTC = DecisionTreeClassifier()
RFC = RandomForestClassifier()
XT = ExtraTreesClassifier()

# Creating a list model where all the models will be appended for further evaluation in loop.
models=[]
models.append(('LinearSVC',SVC))
models.append(('SGDClassifier',SGD))
models.append(('XGBClassifier',XGB))
models.append(('DecisionTreeClassifier',DTC))
models.append(('RandomForestClassifier',RFC))
models.append(('XT = ExtraTreesClassifier',XT))
```



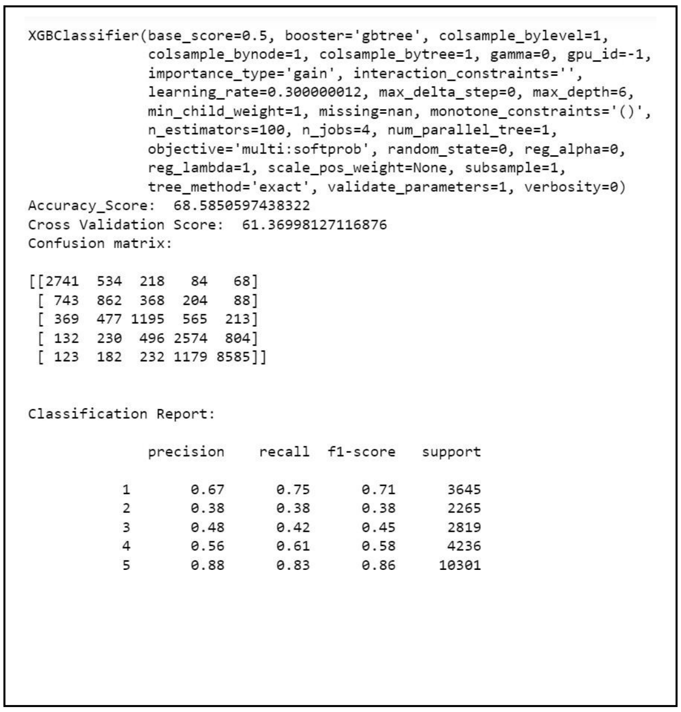
```
# Creating empty lists
Model = []
Acc_score = []
cvs = []

for name,model in models:
    print("******************************",name,"******************************")
    print("\n")
    Model.append(name)
    model.fit(x_train_os,y_train_os)
    print(model)
    y_pred=model.predict(x_test)
# Accuracy Score
    acc_score=accuracy_score(y_test,y_pred)*100
    print('Accuracy_Score: ',acc_score)
    Acc_score.append(acc_score)
# Cross Validation Score
    cv=cross_val_score(model,X,y,cv=5,scoring='accuracy').mean()*100
    print('Cross Validation Score: ',cv)
    cvs.append(cv)
# Confusion Matrix
    print('Confusion matrix: \n')
    cm=confusion_matrix(y_test,y_pred)
    print(cm)
    print("\n")
# Classification Report
    print('Classification Report:\n ')
    print(classification_report(y_test,y_pred))
```



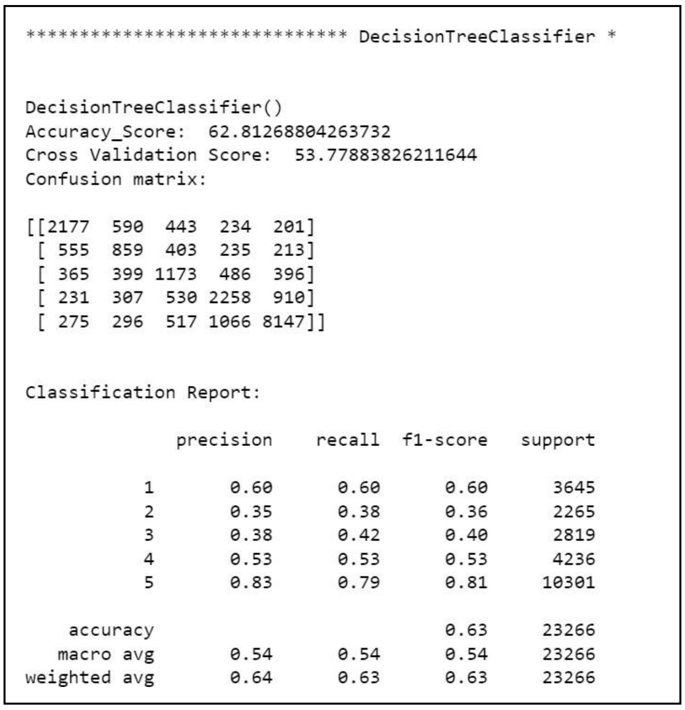
```
****************************** LinearSVC *************


LinearSVC()
Accuracy_Score:  69.8229175621078
Cross Validation Score:  59.58017922099417
Confusion matrix:

[[2684  512  276   97   76]
 [ 624  919  431  172  119]
 [ 326  444 1317  500  232]
 [ 135  201  509 2469  922]
 [ 125  123  244  953 8856]]


Classification Report:

              precision    recall  f1-score   support

           1       0.69      0.74      0.71      3645
           2       0.42      0.41      0.41      2265
           3       0.47      0.47      0.47      2819
           4       0.59      0.58      0.59      4236
           5       0.87      0.86      0.86     10301

    accuracy                           0.70     23266
   macro avg       0.61      0.61      0.61     23266
weighted avg       0.70      0.70      0.70     23266
```



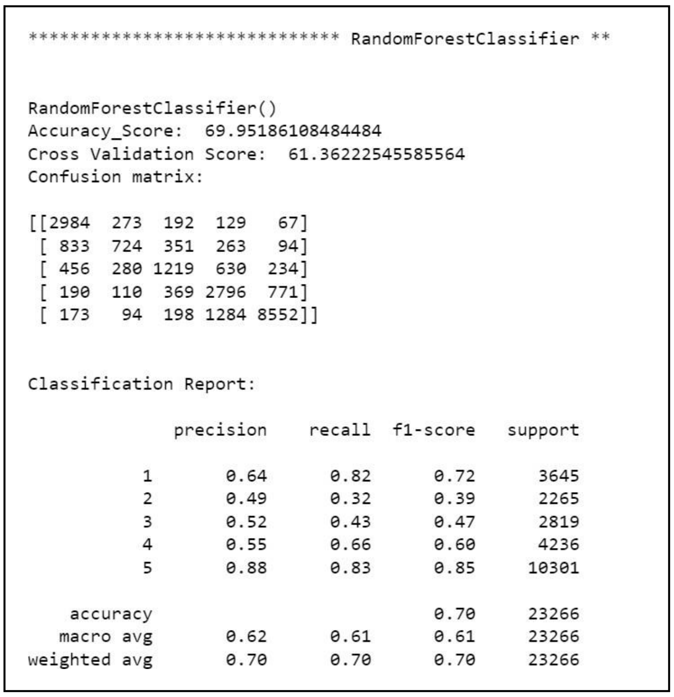
```
****************************** SGDClassifier *********


SGDClassifier()
Accuracy_Score:  69.37161523252814
Cross Validation Score:  62.1836183643055
Confusion matrix:

[[3040  321  133   99   52]
 [ 865  728  346  227   99]
 [ 443  409 1081  675  211]
 [ 192  182  349 2692  821]
 [ 176   95  132 1299 8599]]


Classification Report:

              precision    recall  f1-score   support

           1       0.64      0.83      0.73      3645
           2       0.42      0.32      0.36      2265
           3       0.53      0.38      0.44      2819
           4       0.54      0.64      0.58      4236
           5       0.88      0.83      0.86     10301

    accuracy                           0.69     23266
   macro avg       0.60      0.60      0.60     23266
weighted avg       0.69      0.69      0.69     23266
```



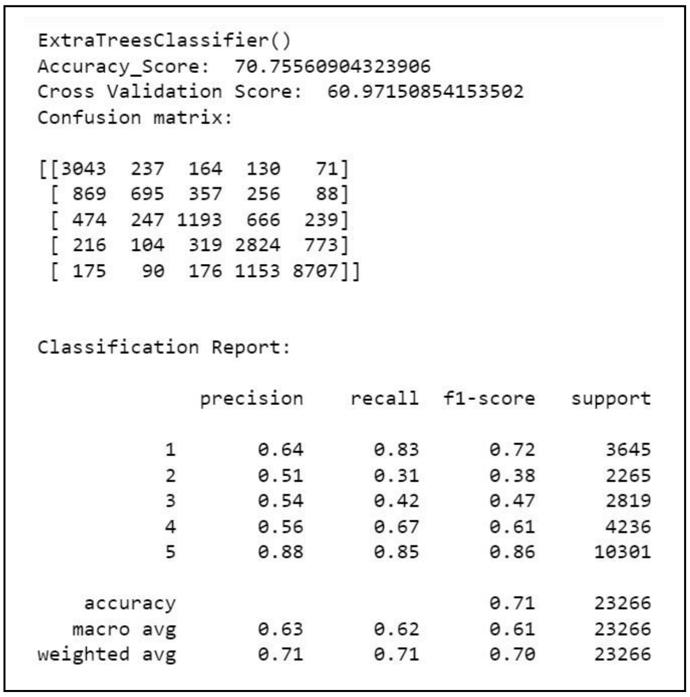
```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
              importance_type='gain', interaction_constraints='',
              learning_rate=0.300000012, max_delta_step=0, max_depth=6,
              min_child_weight=1, missing=nan, monotone_constraints='()',
              n_estimators=100, n_jobs=4, num_parallel_tree=1,
              objective='multi:softprob', random_state=0, reg_alpha=0,
              reg_lambda=1, scale_pos_weight=None, subsample=1,
              tree_method='exact', validate_parameters=1, verbosity=0)
Accuracy_Score:  68.5850597438322
Cross Validation Score:  61.36998127116876
Confusion matrix:

[[2741  534  218   84   68]
 [ 743  862  368  204   88]
 [ 369  477 1195  565  213]
 [ 132  230  496 2574  804]
 [ 123  182  232 1179 8585]]


Classification Report:

              precision    recall  f1-score   support

           1       0.67      0.75      0.71      3645
           2       0.38      0.38      0.38      2265
           3       0.48      0.42      0.45      2819
           4       0.56      0.61      0.58      4236
           5       0.88      0.83      0.86     10301
```



```
****************************** DecisionTreeClassifier *


DecisionTreeClassifier()
Accuracy_Score:  62.81268804263732
Cross Validation Score:  53.77883826211644
Confusion matrix:

[[2177  590  443  234  201]
 [ 555  859  403  235  213]
 [ 365  399 1173  486  396]
 [ 231  307  530 2258  910]
 [ 275  296  517 1066 8147]]


Classification Report:

              precision    recall  f1-score   support

           1       0.60      0.60      0.60      3645
           2       0.35      0.38      0.36      2265
           3       0.38      0.42      0.40      2819
           4       0.53      0.53      0.53      4236
           5       0.83      0.79      0.81     10301

    accuracy                           0.63     23266
   macro avg       0.54      0.54      0.54     23266
weighted avg       0.64      0.63      0.63     23266
```

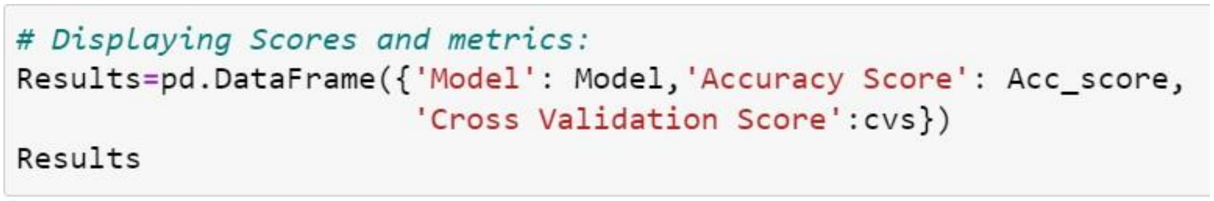


```
****************************** RandomForestClassifier **


RandomForestClassifier()
Accuracy_Score:  69.95186108484484
Cross Validation Score:  61.36222545585564
Confusion matrix:

[[2984  273  192  129   67]
 [ 833  724  351  263   94]
 [ 456  280 1219  630  234]
 [ 190  110  369 2796  771]
 [ 173   94  198 1284 8552]]


Classification Report:

              precision    recall  f1-score   support

           1       0.64      0.82      0.72      3645
           2       0.49      0.32      0.39      2265
           3       0.52      0.43      0.47      2819
           4       0.55      0.66      0.60      4236
           5       0.88      0.83      0.85     10301

    accuracy                           0.70     23266
   macro avg       0.62      0.61      0.61     23266
weighted avg       0.70      0.70      0.70     23266
```

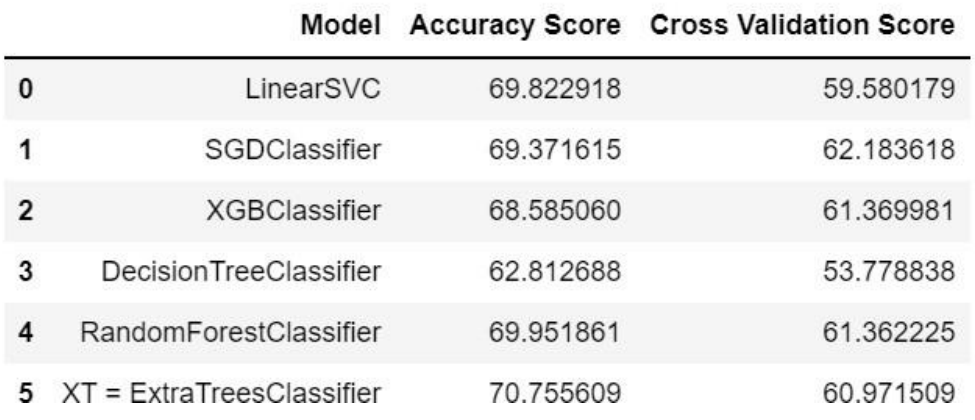


```
ExtraTreesClassifier()
Accuracy_Score:  70.75560904323906
Cross Validation Score:  60.97150854153502
Confusion matrix:

[[3043  237  164  130   71]
 [ 869  695  357  256   88]
 [ 474  247 1193  666  239]
 [ 216  104  319 2824  773]
 [ 175   90  176 1153 8707]]


Classification Report:

              precision    recall  f1-score   support

           1       0.64      0.83      0.72      3645
           2       0.51      0.31      0.38      2265
           3       0.54      0.42      0.47      2819
           4       0.56      0.67      0.61      4236
           5       0.88      0.85      0.86     10301

    accuracy                           0.71     23266
   macro avg       0.63      0.62      0.61     23266
weighted avg       0.71      0.71      0.70     23266
```

Model Selection:



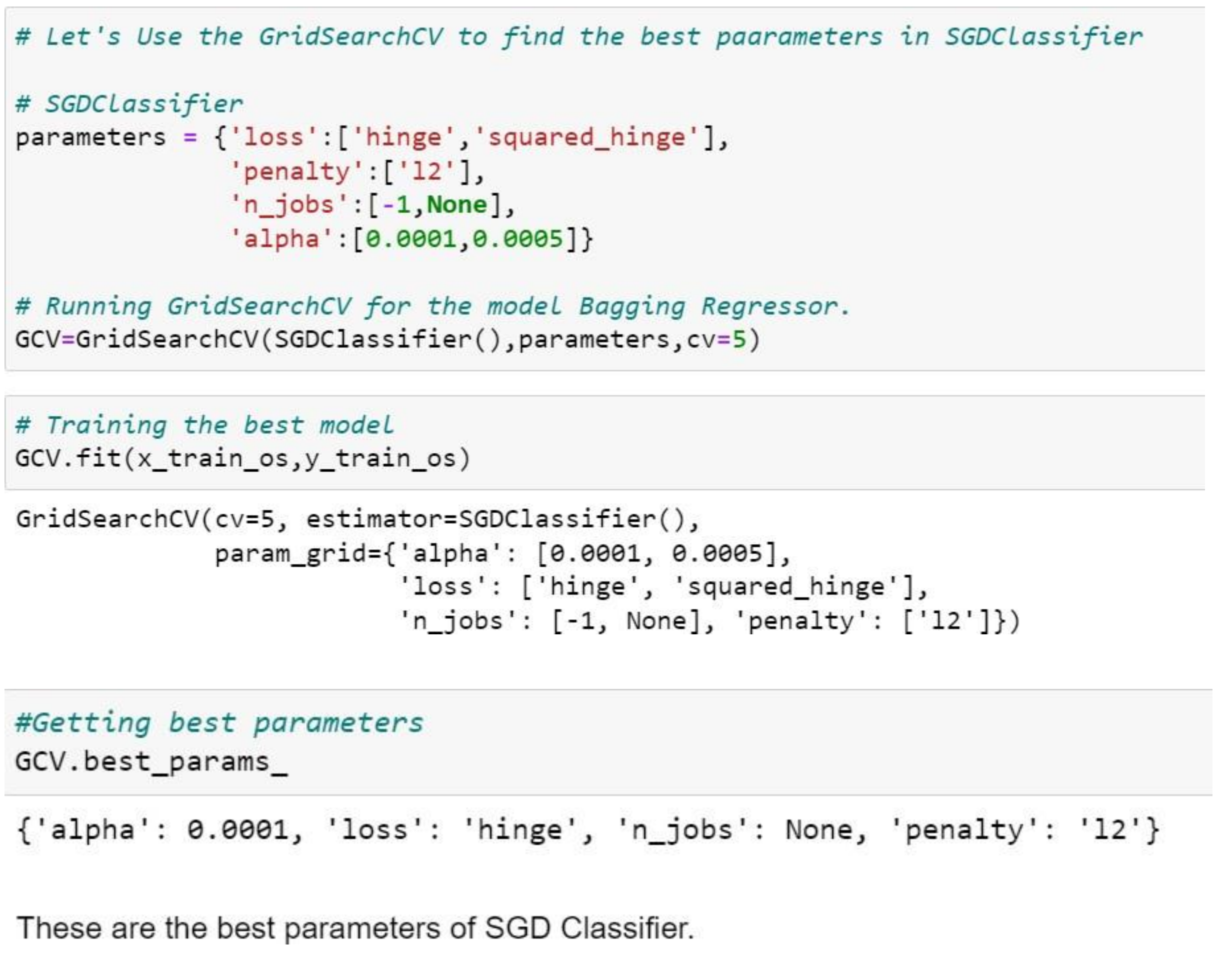
```
# Displaying Scores and metrics:
Results=pd.DataFrame({'Model': Model,'Accuracy Score': Acc_score,
                      'Cross Validation Score':cvs})
Results
```



	Model	Accuracy Score	Cross Validation Score
0	LinearSVC	69.822918	59.580179
1	SGDClassifier	69.371615	62.183618
2	XGBClassifier	68.585060	61.369981
3	DecisionTreeClassifier	62.812688	53.778838
4	RandomForestClassifier	69.951861	61.362225
5	XT = ExtraTreesClassifier	70.755609	60.971509

After creating and training different classification algorithms, we can see that the difference between accuracy and cross validation score is less for "Stochastic Gradient Descent Classifier (SGDClassifier)". On this basis I can conclude that "SGDClassifier" as the best fitting model. Now, we will try Hyperparameter Tuning to find out the best parameters and using them to improve the scores and metrics values.

Hyper Parameter Tuning:



```
# Let's Use the GridSearchCV to find the best paarameters in SGDClassifier

# SGDClassifier
parameters = {'loss':['hinge','squared_hinge'],
              'penalty':['l2'],
              'n_jobs':[-1,None],
              'alpha':[0.0001,0.0005]}

# Running GridSearchCV for the model Bagging Regressor.
GCV=GridSearchCV(SGDClassifier(),parameters,cv=5)
```

```
# Training the best model
GCV.fit(x_train_os,y_train_os)
```

```
GridSearchCV(cv=5, estimator=SGDClassifier(),
             param_grid={'alpha': [0.0001, 0.0005],
                         'loss': ['hinge', 'squared_hinge'],
                         'n_jobs': [-1, None], 'penalty': ['l2']})
```

```
#Getting best parameters
GCV.best_params_
```

```
{'alpha': 0.0001, 'loss': 'hinge', 'n_jobs': None, 'penalty': 'l2'}
```

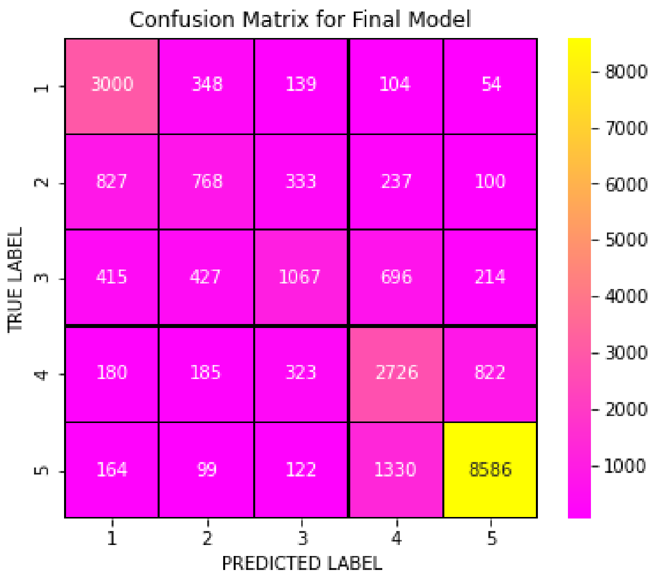
These are the best parameters of SGD Classifier.

I have used 4 SGDClassifier parameters to be saved under the variable "parameters" that will be used in GridSearchCV for finding the best output. Assigned a variable to the GridSearchCV function after entering all the necessary inputs. And we used our training data set to make the GridSearchCV aware of all the hyper parameters that needs to be applied on our best model. After getting the best parameters here I am creating new model to get the predicted values of the model.

Creating Final Model:

```
# Creating final model
ratings_model = SGDClassifier(alpha=0.0001, loss='hinge', n_jobs=None, penalty='l2')
ratings_model.fit(x_train_os, y_train_os)
pred = ratings_model.predict(x_test)
acc_score = accuracy_score(y_test,pred)
print("Accuracy score:", acc_score*100)
print('Confusion Matrix: \n',confusion_matrix(y_test,pred))
print('\n')
print('Classification Report:','\n',classification_report(y_test,pred))
```

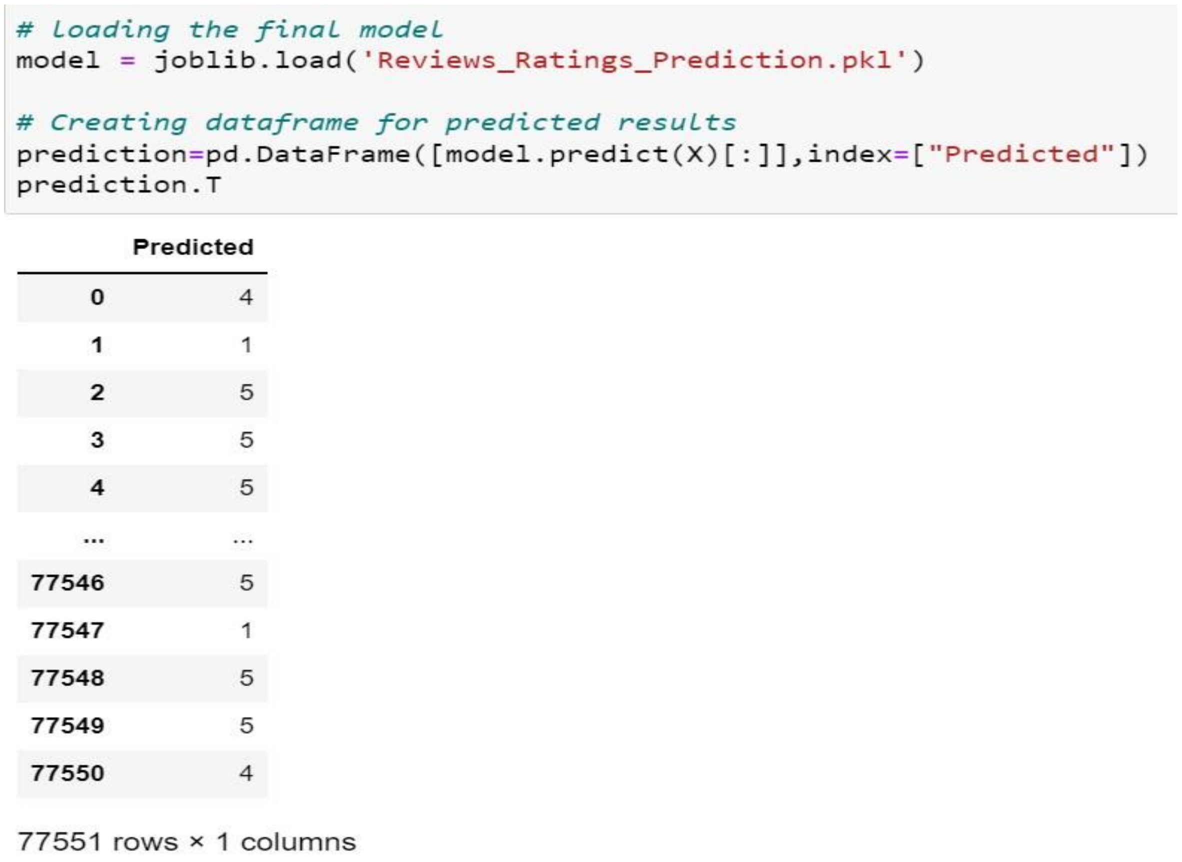
```
Accuracy score: 69.40170205450012
Confusion Matrix:
 [[3000  348  139  104   54]
 [ 827  768  333  237  100]
 [ 415  427 1067  696  214]
 [ 180  185  323 2726  822]
 [ 164   99  122 1330 8586]]


Classification Report:
               precision    recall  f1-score   support

           1       0.65      0.82      0.73      3645
           2       0.42      0.34      0.38      2265
           3       0.54      0.38      0.44      2819
           4       0.54      0.64      0.58      4236
           5       0.88      0.83      0.86     10301

    accuracy                           0.69     23266
   macro avg       0.61      0.60      0.60     23266
weighted avg       0.69      0.69      0.69     23266
```



I have successfully incorporated the hyper parameter tuning using best parameters of SGDClassifier and the accuracy of the model has been increased after hyperparameter tuning and received the accuracy score as 69.40 % which is very good.

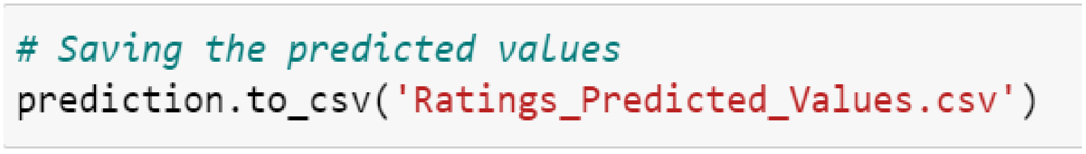
With the help of confusion matrix, we can able to see actual and predicted values for the final model. And also, we can understand the number of times we got the correct outputs and the number of times my model missed to provide the correct prediction.

Saving the final model and predicting the results



```
# Loading the final model
model = joblib.load('Reviews_Ratings_Prediction.pkl')

# Creating dataframe for predicted results
prediction=pd.DataFrame([model.predict(X)[:]],index=["Predicted"])
prediction.T
```

```
       Predicted
    0          4
    1          1
    2          5
    3          5
    4          5
  ...        ...
77546          5
77547          1
77548          5
77549          5
77550          4

77551 rows × 1 columns
```



```
# Saving the predicted values
prediction.to_csv('Ratings_Predicted_Values.csv')
```

I have loaded the saved model to get the predictions for reviews ratings. Using classification model, we have got the predicted values for review ratings. Finally saved the predicted value of review ratings.

3.4 Key Metrics for success in solving problem under consideration

In order to evaluate the performance of each algorithm, several metrics are defined accordingly, and are discussed briefly below.

- **Accuracy score**: This metric measures how many of the comments are labelled correctly. However, in our dataset, where most of comments are not toxic, regardless of performance of model, a high accuracy was achieved. Accuracy is the ratio of number of correct predictions into number of predictions. In binary classification problem, accuracy can be calculated as below,

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

Where TP = True Positive, TN = True Negative, FP = False Positive, FN = False Negative.

- **Precision and Recall:** Precision and recall were designed to measure the model performance in its ability to correctly classify the malignant

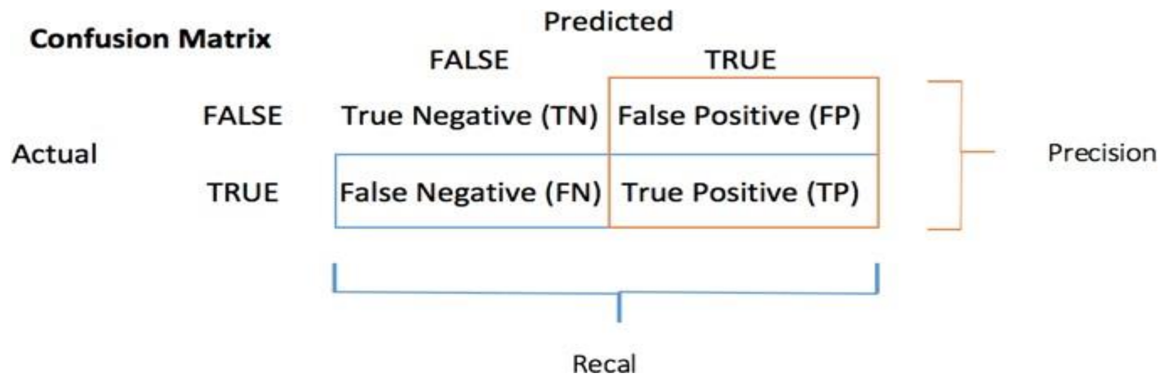
$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive} \qquad \text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

comments. Precision explains what fraction of malignant classified comments are truly malignant, and Recall measures what fraction of malignant comments are labelled correctly.

- **F1 Score** is used to express the performance of the machine learning model (or classifier). It gives the combined information about the precision and recall of a model. This means a high F1-score indicates a high value for both recall and precision.

$$F1\ score = 2 * \frac{Precision\ * Recall}{Precision + Recall}$$

- **Confusion Matrix** is one of the evaluation metrics for machine learning classification problems, where a trained model is being evaluated for



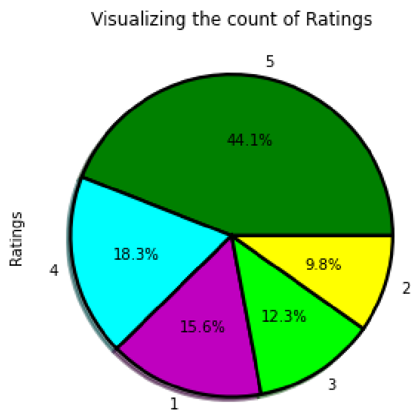
accuracy and other performance measures. And this matrix is called the confusion matrix since it results in an output that shows how the system is confused between the two classes.

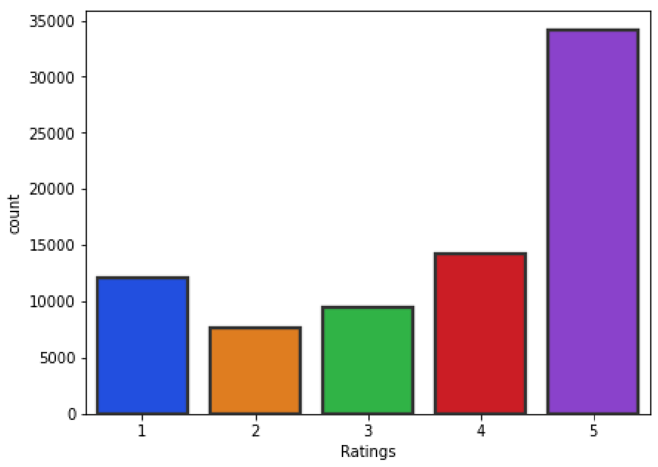
- **Cross Validation Score** is a technique in which we train our model using the subset of the data-set and then evaluate using the complementary subset of the data-set. It is used to protect against overfitting in a predictive model, particularly in a case where the amount of data may be limited. In cross-validation, you make a fixed number of folds (or partitions) of the data, run the analysis on each fold, and then average the overall error estimate. It is used to estimate the performance of ML models.

3.5 Visualizations

I used pandas profiling to get the over viewed visualization on the pre-processed data. Pandas is an open-source Python module with which we can do an exploratory data analysis to get detailed description of the features and it helps in visualizing and understanding the distribution of each variable. I have used wordcloud to get the sense of loud words in the label for each rating.

Count of Target Variable: Ratings



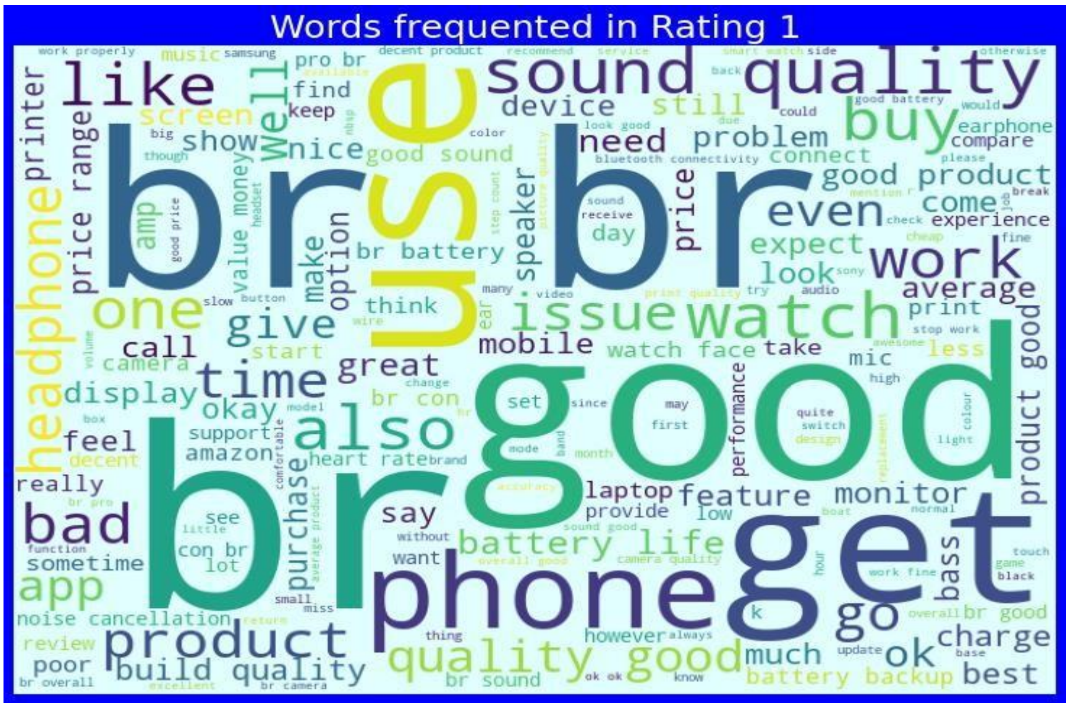


Observations:

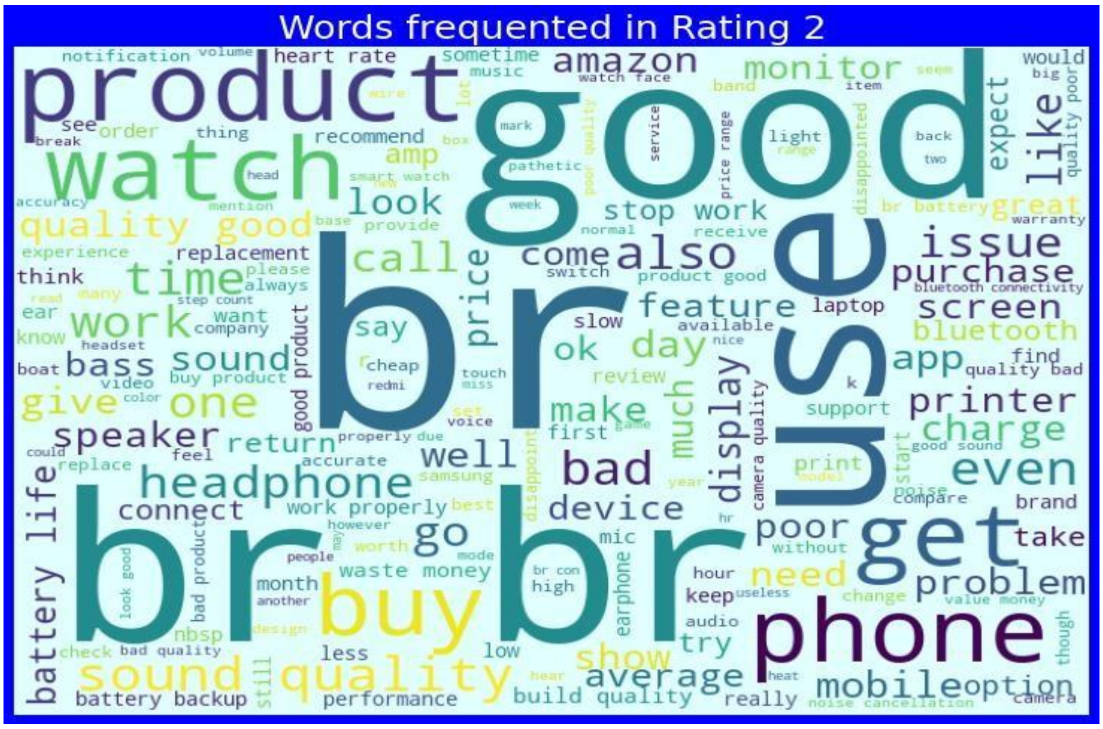
- Looking at the above pie chart and count plot for our target variable "Ratings" we can say that the dataset contains the review text rated as 5 stars compared to other review ratings and very less reviews rated as 2 compared to others. Around 44% of the texts are rated as 5 and only 10% of the texts rated as 2 stars.
- So, we can say that there is "imbalance problem" which I have balanced using SMOTE method.

Plotting WordCloud for each label:

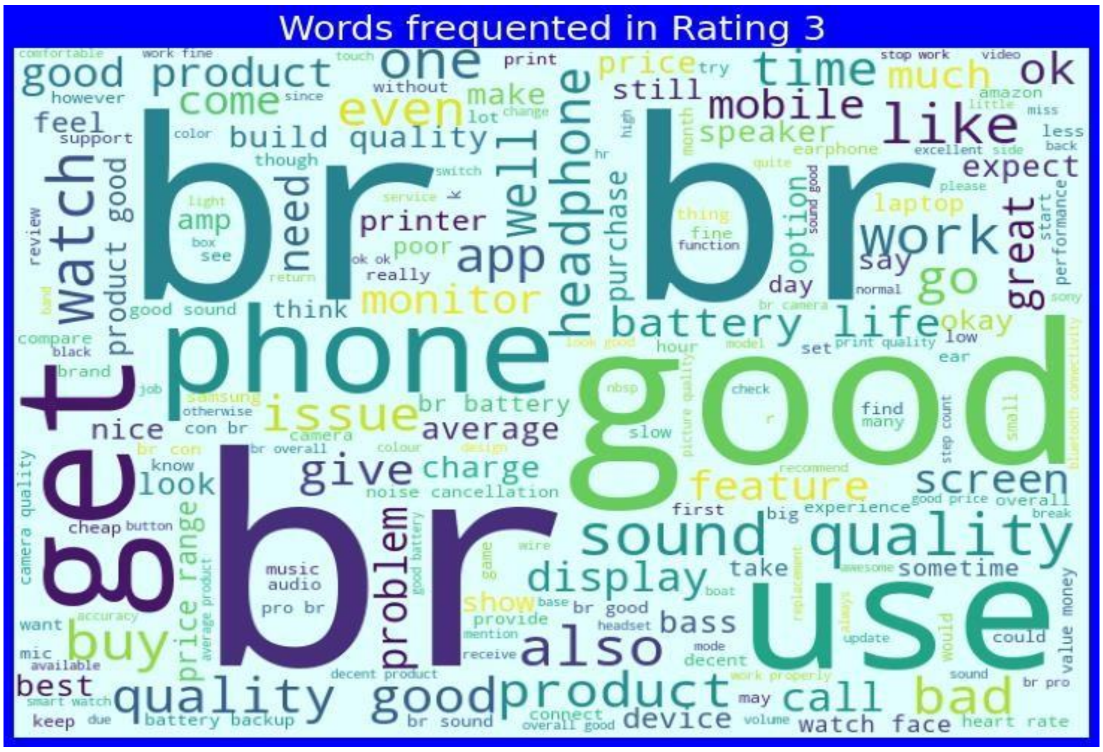
Words for rating = 1



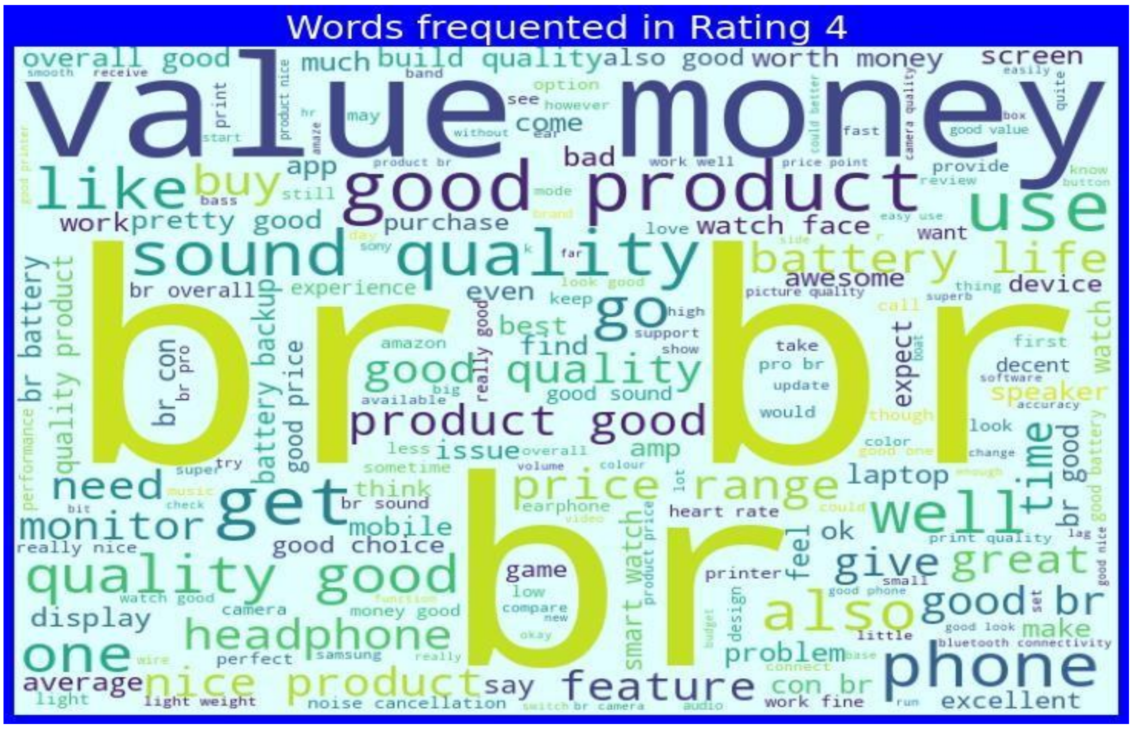
Words for rating = 2



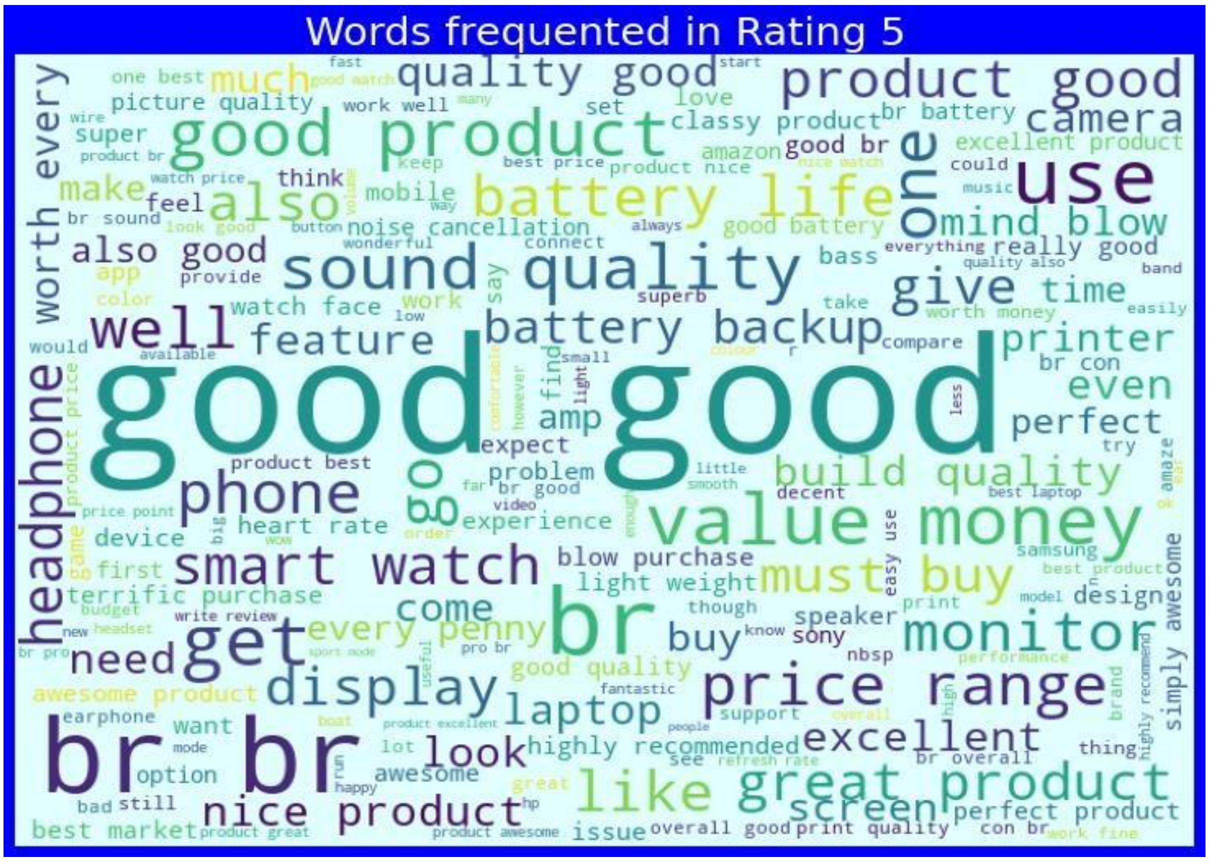
Words for rating = 3



Words for rating = 4



Words for rating = 5



Observations:

- From the above plots we can clearly see the sense of loud words in the reviews. The enlarged words in the above plots are the most used texts for ratings 1, 2, 3, 4, and 5 stars.
- Here most frequent words used for each rating is displayed in the word cloud based on label and also when all the values are present.

3.6 Interpretation of the Results

<u>Visualizations:</u> I have used distribution plot to visualize how the data has been distributed. Used count plots and pie charts to check the count of particular category for target feature. The heat map helped me to understand the correlation between dependent and independent features. Also, heat map helped to detect the multicollinearity problem and feature importance. With the help of WordClouds I would able to sense the loud words in each class of label.

<u>Pre-processing:</u> The dataset should be cleaned and scaled to build the ML models to get good predictions. I have performed few NLP text processing steps which I have already mentioned in the pre-processing steps where the important features are present in the dataset and ready for model building.

<u>Model building:</u> After cleaning and processing data, I performed train test split to build the model. I have built multiple classification models to get the accurate accuracy score, and evaluation metrics like precision, recall, confusion matrix, f1 score and cross validation score. I got SGD Classifier as the best model which gives 69.37% accuracy score. I checked the cross-validation score ensuring there will be no overfitting. After tuning the best model SGD Classifier, I got 69.40 % accuracy score. Finally, I saved my final model and got the good predictions results for reviews ratings.

4. <u>CONCLUSION</u>

4.1 Key Findings and Conclusions of the Study

From the above analysis the below mentioned results were achieved which depicts the conditions of a reviews ratings;

- ✓ With the increasing popularity of e-commerce websites more and more people consume feeds from social media and buy products on the basis of reviews and ratings of the products given by others.
- ✓ In this project I have collected data from of reviews ratings for different products from amazon.in and flipkart.com. Then I have done some text pre-processing for review column and chose maximum value count of text from each rating class and applied SMOTE method to eliminate problem of imbalance.
- ✓ By doing different EDA steps I have analysed the text where I got that the dataset contains more review text rated as 5 stars compared to other review ratings. Around 44% of the texts are rated as 5 and only 10% of the texts rated as 2 stars. And checked frequently occurring words in each ratings using word cloud plotting. After all these steps I have built function to train and test different algorithms and using various evaluation metrics.

4.2 Learning Outcomes of the Study in respect of Data Science

While working on this project we learned many things and gains new techniques and ways to deal with uncleaned text data. Found how to deal with multiclassification problem. Tools used for visualizations gives a better understanding of dataset. We have used a lot of algorithms and, SGDClassifier gives better results compared to others.

It is possible to classify the comments content into the required categories of authentic and however, using this kind of project we can easily get an idea to buy the products based on the ratings given to the product reviews.

4.3 Limitations of this work and scope for future work

Limitations: This project was amazing to work on, it creates new ideas to think about but there were some limitations in this project like unbalanced dataset. Textual reviews for different products and services are abundant. Still, when trying to make a buy decision, getting sufficient and reliable information can be a daunting task. Every effort has been put on it for perfection but nothing is perfect and this project is of no exception. There are certain areas which can be enhanced. As we know the context of text in reviews is totally depends on the reviewer and they rate differently which is totally depends on that particular person. So, it is difficult to predict the ratings based on the reviews with higher accuracies.

Future work: In future work, we can focus on performance and error analysis of the model as lots of text were present. Previous work has achieved success using various algorithms on data in English language but in future, we can consider having data in regional languages. We have represented 9 algorithms chosen for their simplicity of implementation and run out of time efficiency. Review ratings detection is an emerging research area with few public datasets. Though we have not got better accuracy but we can improve our accuracy by fetching more data and by doing proper text processing and hyperparameter tuning.