Rating Prediction of the App

Objective: Predict the star rating based on the Review Text

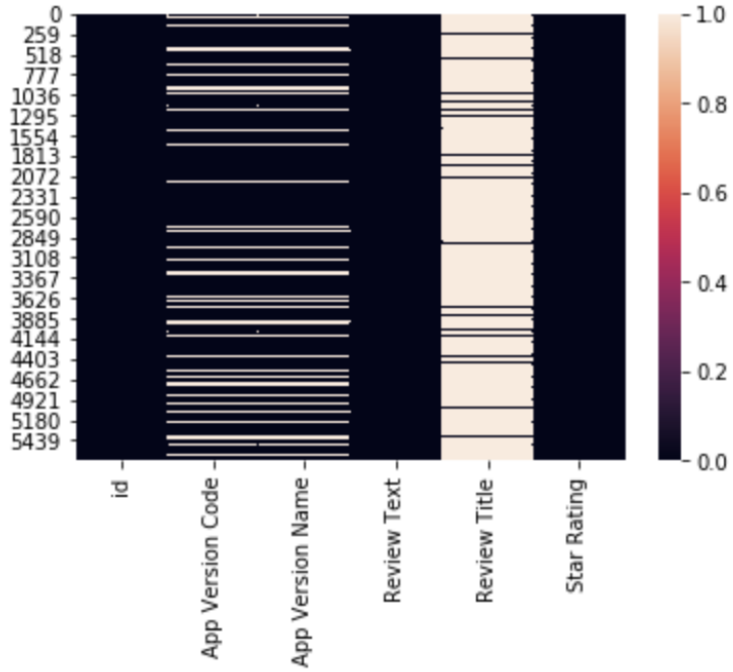
Understanding: The aim is to classify the review text into different star rating [1-5]. The review text is the user feedback given along with columns as: app name, app version , user id.

Feature Engineering:

There are 6 columns:
Id, App Version, App Name, Review Text, Review Title, Star Rating.

1. **Removing the feature having too many null values:**
   App Version, App Name, Review Title

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e7b093f9b0>
```

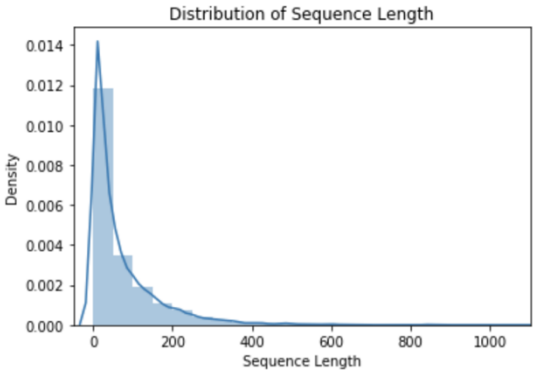


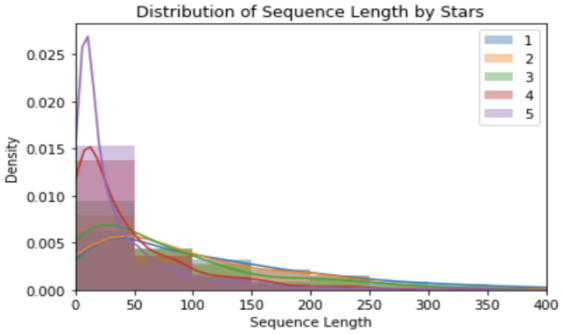
2. **Removing the not necessary feature:**
   Id: unique for every rating. Not needed in prediction

3. **Dropping a single null value in Review Text**

4. **Adding the “length” Feature for review text field.**

   Visualize the length of text. Also star rating wise length visualization





Cleaning the data:

When dealing with numerical data, data cleaning often involves removing null values and duplicate data, dealing with outliers, etc. With text data, there are some common data cleaning techniques, which are also known as text pre-processing techniques.

Common data cleaning steps on all text:

1. Make text all lower case
2. Remove punctuation
3. Remove numerical values
4. Tokenize text
5. Remove stop words

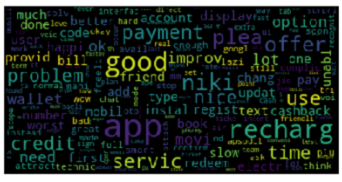
More data cleaning steps after tokenization:

1. Stemming / lemmatization
2. Create bi-grams or tri-grams

5. **Analyzed Common words in each star rating using Word Cloud:**

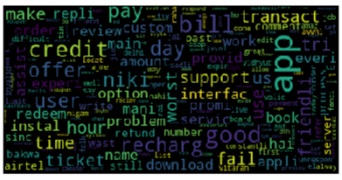
Word Cloud for Rating 1

Word Cloud for Rating 3



Word Cloud for Rating 5

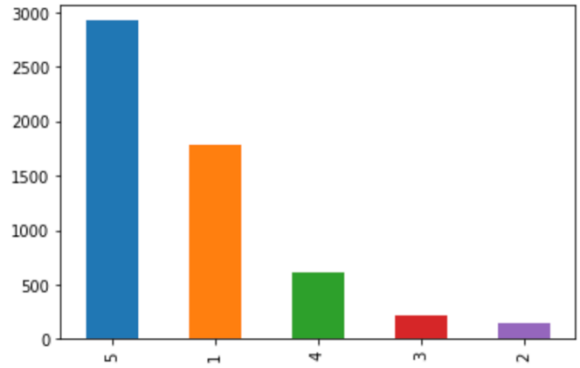
Word Cloud for Rating 2

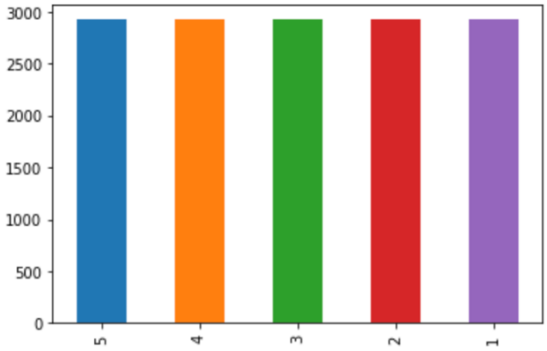


Word Cloud for Rating 4

6. **Balancing the dataset using SMOTE:**

Before **After**





Modelling:

Different classification algo's are used to determine which performs best on our dataset.

1. Multinomial Naive Bayes
2. Logistic Regression
3. SVM Classifier
4. KNN Classifier
5. Decision Tree Classifier
6. Random Forest Classifier
7. XGboost Classifier: (Didn't run because it was too much time. Surely, It can increase the F1 score by 2-5 %)

Evaluation Metric:

F-Score: F-measure has a parameter that sets the tradeoff between recall and precision. The standard F-measure is F1, which gives equal importance to recall and precision:

$$F_1 = (2 \cdot recall \cdot precision) / (recall + precision)$$

Weighted F-Score: If we give more weight to recall or to precision, use the Weighted F-measure:

$$F_\beta = (1+\beta_2) \cdot recall \cdot precision / recall + \beta_2 \cdot precision$$

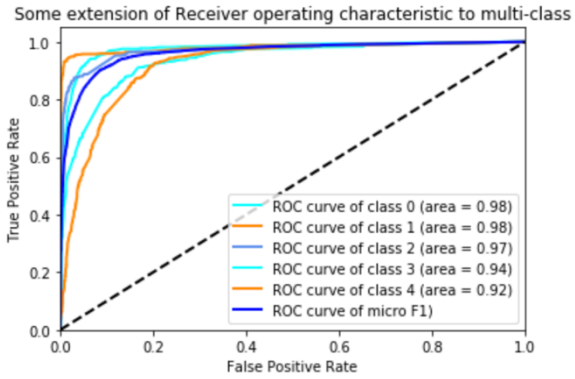
Result:

After running these models, the best F1 Score was obtained by using **Random Forest Classifier (82 %),** Giving equal weightage to Precision and Recall.

```
Confusion Matrix
 [[508  16  10  12  34]
 [  9 531  11   9   4]
 [  7   9 503  45  23]
 [ 15   8  19 459 112]
 [ 55   4  12 126 382]]
Classification Report
              precision    recall  f1-score   support

           1       0.86      0.88      0.87       580
           2       0.93      0.94      0.94       564
           3       0.91      0.86      0.88       587
           4       0.71      0.75      0.73       613
           5       0.69      0.66      0.67       579

    accuracy                           0.82      2923
   macro avg       0.82      0.82      0.82      2923
weighted avg       0.82      0.82      0.82      2923
```



What can be done better:

Following things can be done with more timing and research.

1. As we can observe the text contains reviews in **Hindi** language also.
   So, if we can **identify the language** in each review and translate it to English language (more common language), We can definitely obtain better results:

2. Modelling wise, we can use **Neural Network**: especially LSTM for achieving better results.