DSL Project Report

Arnab Kumar Gogoi

19BM61K27

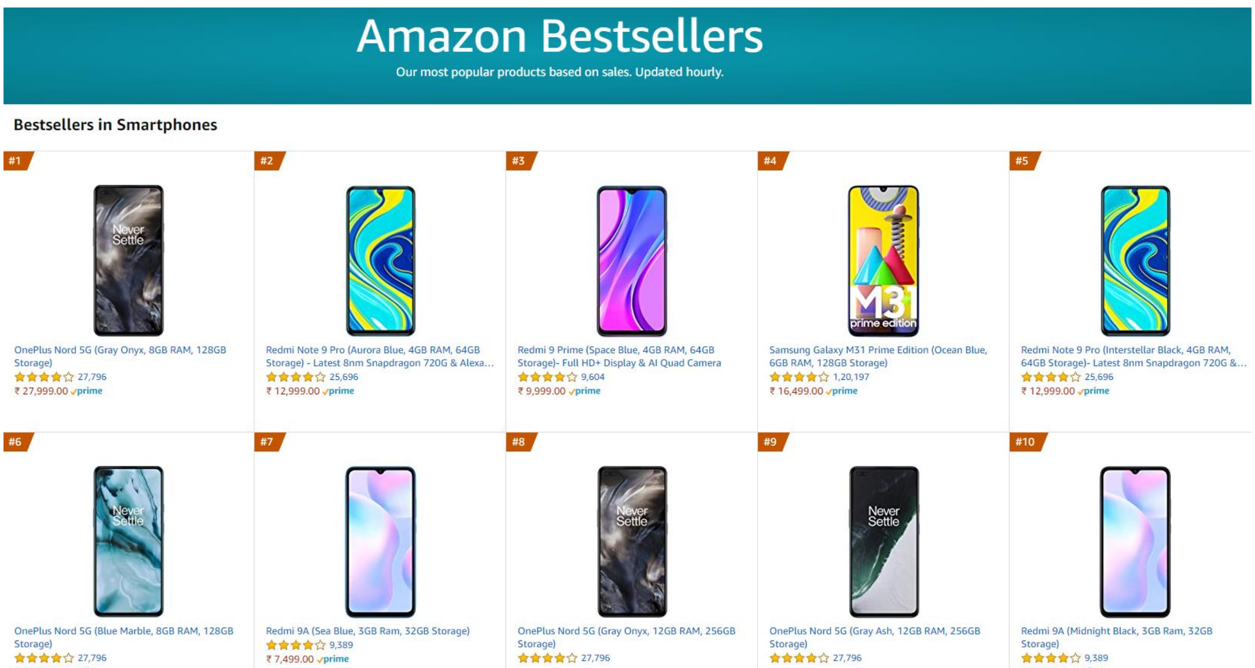
EMBA 2019-2022

Introduction

Online product reviews are helpful for potential customers to evaluate the products before making a decision. These reviews can be based on facts or opinions depending on how individuals have rated the products and the social status of the reviewers. Amazon as an online retail place has built a community and allows the community to rate products and post reviews on their website. This type of information sharing helps to reduce the uncertainty of customers deciding to buy a quality product in their marketplace.

While buying a product from the Amazon store, a customer can check the rating of the product as displayed as starts between 1 to 5. When more detailed user feedback is required, customers can consider going through some reviews as the reviews carry much more information than the rating points.

The problem



This is a page on Amazon India front page that lists the top 100 bestselling smartphones from rank 1 to 100 which changes every day. The link to the page is here.

In this project, my goal is to study what makes a review a good review and what makes it a bad review. Besides, by using NLP, I will develop a prediction model to tell whether a review indicates a positive rating or negative rating.

Data scraping

We have tried 2 methods of data scraping:

1. First method: Using BeautifulSoup HTML parser:

Using this method, one review page worth of information can be extracted, not all the pages. As each amazon page contains 10 reviews each, this is not a viable method, because it is not automated.

This method was tested on the OnePlus 8 product page on amazon.

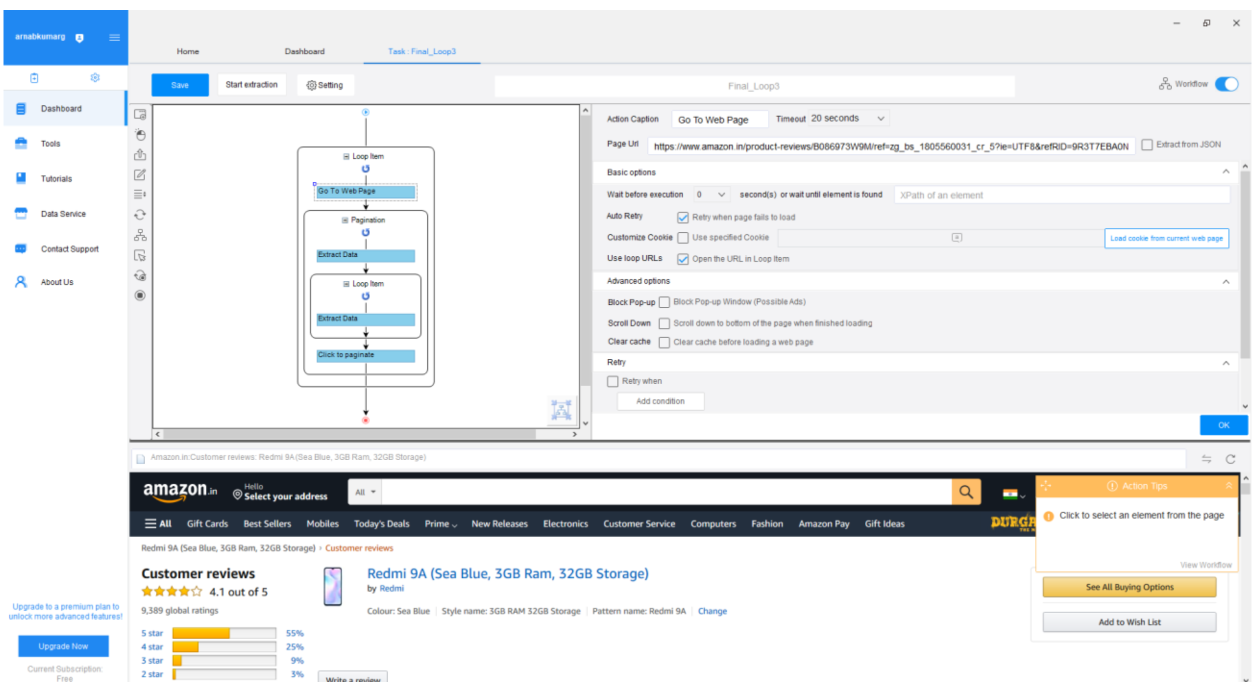
Code file:

https://colab.research.google.com/github/arnabkumargogoi/DSL2020Project/blob/main/01_Web_Scrapping_Amazon_Product_Reviews_OnePlus_8.ipynb

Extracted data csv:
https://github.com/arnabkumargogoi/DSL2020Project/blob/main/csv/01%20Web%20Scrapping%20Amazon%20Product%20Reviews%20OnePlus%208%20using%20Beautifulsoup.csv

2. Second method: Using Octoparse

Octoparse is a standalone software that allows mass web scraping.



Flow:

1. Click on each review link on the Amazon bestseller page
2. Loop next button

3. Loop select reviews and extract elements
4. Cancel after 1000 reviews each product
5. Delete duplicates

Using the above flow, we extracted 2 lists:

1. **Amazon 100 Bestselling smartphone list:** (The link to the page is here.)
https://github.com/arnabkumargogoi/DSL2020Project/blob/main/csv/02%20Amazon%20100%20Bestesellers%20list.csv

2. **Amazon Bestseller Smartphones all review list**:

https://github.com/arnabkumargogoi/DSL2020Project/blob/main/csv/03%20Amazon%20Bestseller%20Smartphones%20all%20reviews.csv

The data

Analysis of Amazon 100 Bestselling smartphone list

Using the 1st list (**Amazon 100 Bestselling smartphone list**), we tried to do some simple analysis using the **code file** :

https://colab.research.google.com/github/arnabkumargogoi/DSL2020Project/blob/main/02_Amazon100Bestesellers.ipynb and found the following information:

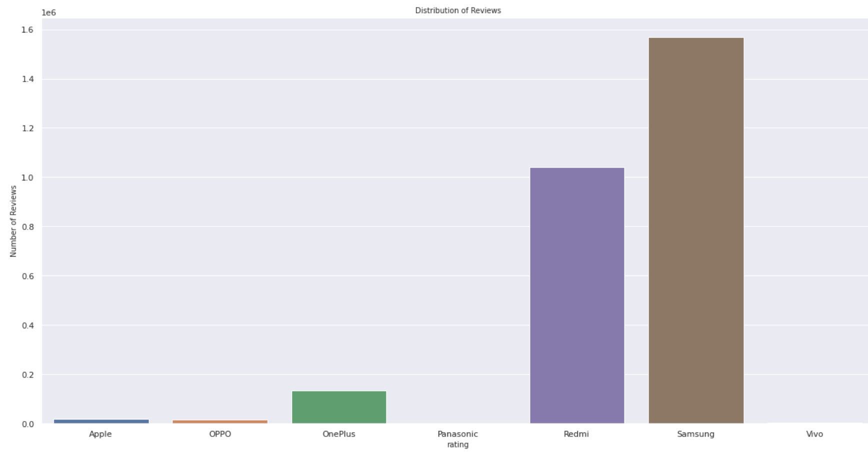
1. No of total products: 100

Unique Models that share reviews: 33

Unique Brands: 7

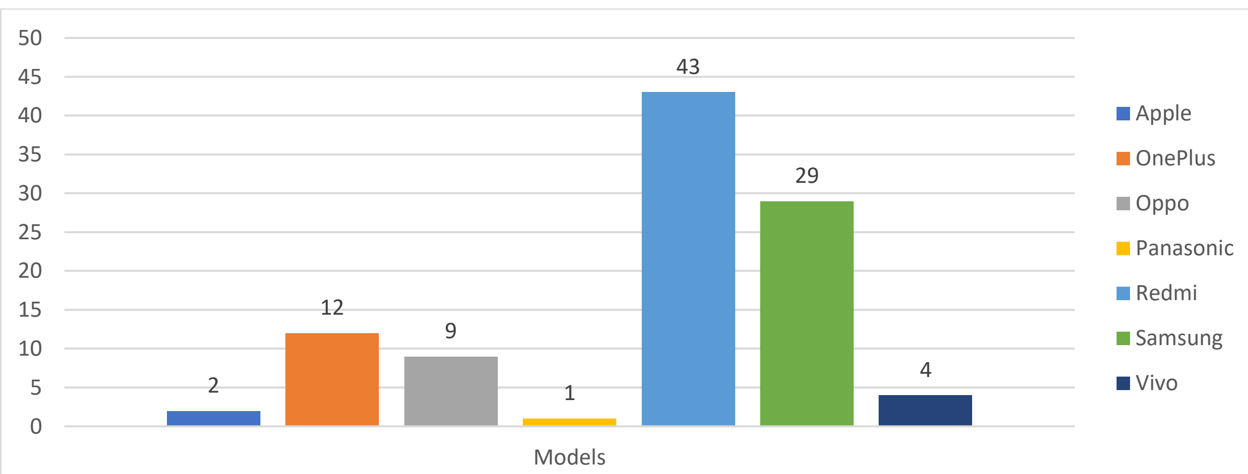
2. Number of reviews per brand:

1. Apple: 20431
2. Oppo: 16890
3. OnePlus: 133689
4. Panasonic: 677
5. Redmi: 1041009
6. Samsung: 1569569
7. Vivo: 2956



3. Number of models in the top 100 per brand:

1. Apple: 2
2. Oppo: 12
3. OnePlus: 9
4. Panasonic: 1
5. Redmi: 43
6. Samsung: 29
7. Vivo: 4



As we can see, Samsung has the overall highest total reviews and Redmi has the highest total number of models in the top 100 spots.

The 2nd list **Amazon Bestseller Smartphones all review list**, includes **31,673** reviews

Table 1

Column Names and Their Explanation

Feature	**Explanation**

Id	Row Id
name	Product Name
customer	Profile name of the user
rating	Rating between 1 and 5
title	Heading of the review
date	The timestamp of the review
review	Text of the review
Total	**8 rows**

Initial Analysis of Amazon Bestseller Smartphones all review list

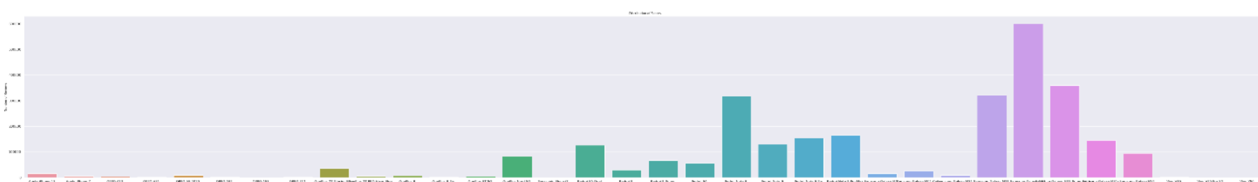
We tried to do an initial analysis using the **code file**: https://colab.research.google.com/github/arnabkumargogoi/DSL2020Project/blob/main/03_DSL_Project_Analysis.ipynb and found the following information:

1. Review count and models:

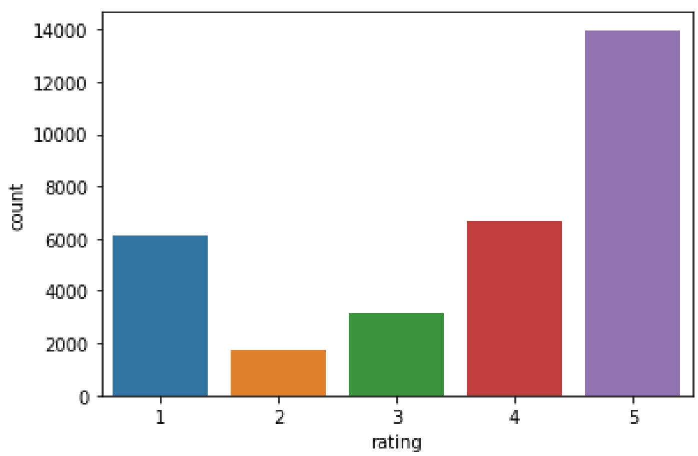
Highest review: Samsung Galaxy M31, M31 Prime, and M21

2nd Highest review: Redmi Note 8, Note 9

3rd Highest review: OnePlus Nord 5G



2. Distribution of the rating column: We can see that maximum reviews correspond to rating 5.



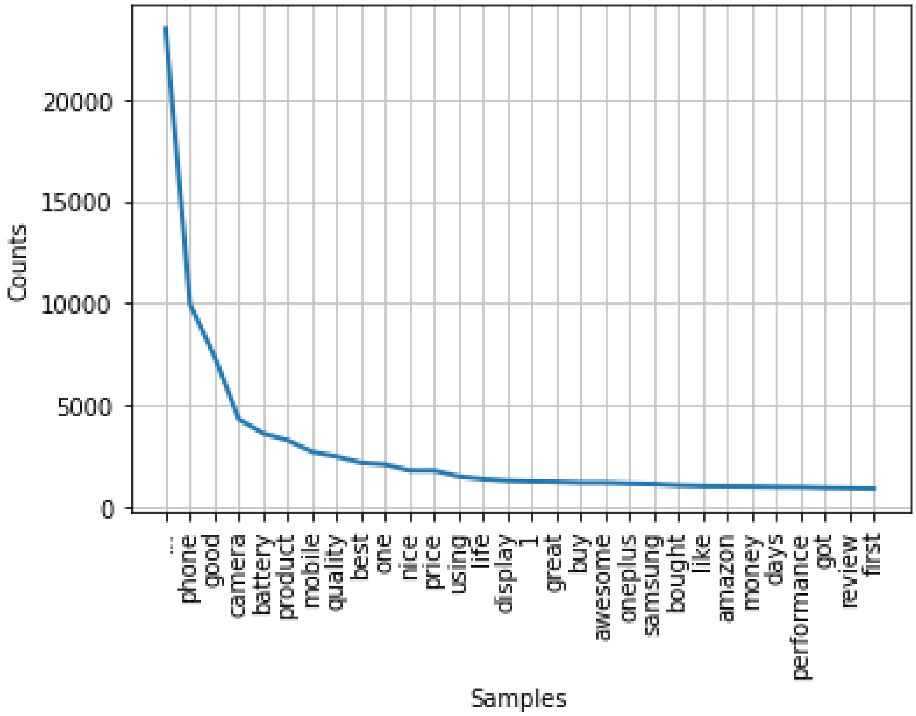
3. Text analysis with NLTK and Vader Sentiment analyzer: **Positive reviews**

Wordcloud for positive reviews shows words best, product, quality, beauty, money, grabbed, apple, etc.

4. Text analysis with NLTK and Vader Sentiment analyzer: Negative Reviews

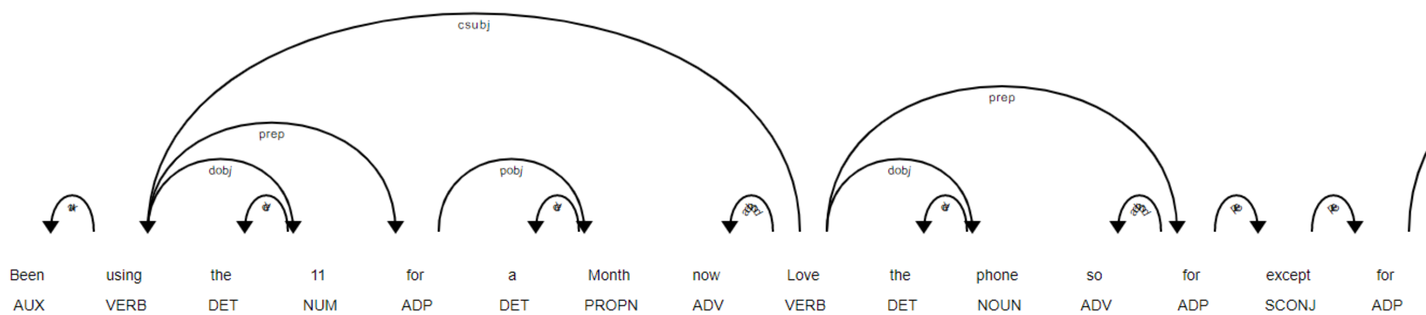
Wordcloud for negative reviews shows words worst, camera, quality, defective, disappointed, etc.

5. Text analysis with NLTK and Vader Sentiment analyzer



Frequency Distribution: Top tokens are product, good, camera, battery, etc.

6. Review text analysis with the spacy library



dependency parse tree or named entity visualization

7. Analyzing with TextBlob

```
[ ] print("negative reviews")
    most_negative = reviews_df[reviews_df.polarity == -1].review.head()
    print(most_negative)
    print("positive reviews")
    most_positive = reviews_df[reviews_df.polarity == 1].review.head()
    print(most_positive)
```

```
negative reviews
535     It is defective phone camera not working on is...
770     Screen was damaged means scratched no support ...
1379    Got a defective try to return it but not allow...
1702    One of the worst products I ve received It s t...
1755    Please don t buy iphone 7 in Amazon it s worst...
Name: review, dtype: object
positive reviews
0                        Best product everValue for money
9       iphone 11 is the perfect iphone best mixture ...
95                                               Awesome
125                          best color among iphone11s
127                      Best product and value for money
Name: review, dtype: object
```

8. Text analysis with gensim and word2vec

Using the trained model to find the similarity of a certain word.

Few examples:

```
model = Word2Vec(reviewsVec,min_count=1, size=32)
model.most_similar('sale')
```

```
[('festival', 0.9314939379692078),
 ('maid', 0.8834912776947021),
 ('diwali', 0.874427080154419),
 ('ordered', 0.8655676245689392),
 ('gift', 0.8557988405227661),
 ('discount', 0.8511460423469543),
 ('9th', 0.8491923213005066),
 ('finally', 0.845089852809906),
 ('addt', 0.8419834971427917),
 ('offer', 0.8409298658370972)]
```

```
model = Word2Vec(reviewsVec,min_count=1, size=32)
model.most_similar('deal')
```

```
[('1300', 0.8136439919471741),
 ('13000', 0.8054893016815186),
 ('7999', 0.7999066114425659),
 ('10000', 0.7996455430984497),
 ('indian', 0.790290296077283),
 ('21500', 0.7864016890525818),
 ('bargain', 0.7844116687774658),
 ('festival', 0.7839363813400269),
 ('12700', 0.7828549146652222),
 ('9999', 0.782375693321228)]
```

```
[103] model = Word2Vec(reviewsVec,min_count=1, size=32)
      model.most_similar('flipkart')
```

```
[('cheating', 0.9255399703979492),
 ('shop', 0.9104970097541809),
 ('portal', 0.9103248119354248),
 ('american', 0.9043139219284058),
 ('fooling', 0.9026511907577515),
 ('strategy', 0.9019997715950012),
 ('stores', 0.8984902501106262),
 ('thanku', 0.898440420627594),
 ('marketing', 0.895793080329895),
 ('fool', 0.8932040929794312)]
```

```
model = Word2Vec(reviewsVec,min_count=1, size=32)
model.most_similar('discount')
```

```
[('hdfc', 0.9596805572509766),
 ('500', 0.95299232006073),
 ('discounts', 0.9300386905670166),
 ('offer', 0.9290621280670166),
 ('1500', 0.9267088770866394),
 ('debit', 0.9156883955001831),
 ('credit', 0.9103200435638428),
 ('emi', 0.908906102180481),
 ('cashback', 0.9007534980773926),
 ('600', 0.8978833556175232)]
```

Final Analysis of Amazon Bestseller Smartphones all review list

The final analysis is provided in the **code file**: https://colab.research.google.com/github/arnabkumargogoi/DSL2020Project/blob/main/04_DSL2020Project.ipynb#scrollTo=UQ67amRYamFu

The flow of study:

1. **Data cleaning and feature extraction:**
   Data cleaning is the first step of every text analysis as the text contains a lot of non-meaningful information. This is a mandatory step that helps us extract the features of the data.
   1.1. Loading and cleaning data:

The CSV file is loaded to google drive and imported to Colab using pandas. The very first step was to look for missing values if any and drop them. We found 9 missing data out of 31673, so we dropped them.

1.2. Basic Feature Extraction – 1

Normally the feature extraction is carried out after the data cleaning process. In this NLP study, as there is some part of the data is deleted during text cleaning, some features are not possible to obtain after the data cleaning. Therefore, the below features are extracted in this process:

1.2.1. Number of stopwords

A stop word is a commonly used word (such as “the”, “a”, “an”, ‘in”) that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query. In python’s **nltk** package, there are 127 English stop words default. These 127 words were ignored.

1.2.2. Number of hashtag characters

One more interesting feature which we can extract from a review is the number of hashtags. After data cleaning, it will not be possible to obtain this feature.

1.2.3. Number of numeric

Having the number of numeric characters in the reviews can be useful. At least, it does not hurt to have such data.

1.2.4. Number of Uppercase words

Anger or rage is quite often expressed by writing in uppercase words which makes this feature extraction a necessary operation

1.3. Text cleaning techniques

Before applying NLP techniques to the data, we first need to clean and prepare the data. If this process is not done properly, it can ruin the analysis part.

1.3.1. Make all text lower case:

The first pre-processing step was transforming the reviews into lower case. This avoids having multiple copies of the same words. For example, while calculating the word count, ‘Analytics’ and ‘analytics’ will be taken as different words if we ignore this transformation.

1.3.2. Removing Punctuation:

Punctuations create noise in the data, should be cleared. For now, there is no meaningful way to analyze punctuations. Thus, they were removed from the text data. With this step, these characters were removed: [!"#$%&'()*+,- J<=>?@M4_{}-]

1.3.3. Removal of Stop Words:

With this step, | removed all default English stop words in the nltk package.

1.3.4. Removing URLs

URLs are another noise in the data that were removed.

1.3.5. Remove HTML tags:

HTML is used extensively on the Internet. But HTML tags themselves are not helpful when processing text.

1.3.6. Removing Emojis:

Emojis can be an indicator of some emotions that can be related to being customer satisfaction. Unfortunately, we need to remove the emojis in our text analysis because for now, it’s not possible to analyze emojis with NLP.

1.3.7. Remove Emoticons:

What is the difference between emoji and emoticons?

:-) is an emoticon.

😊is an emoji.

1.3.8. Spell Correction:

On Amazon reviews, there are a plethora of spelling mistakes. Product reviews are sometimes filled with hastily sent reviews that are barely legible at times. In that regard, spelling correction is a useful pre-processing step because this also will help us in reducing multiple copies of words. For example, “Analytics” and “analytics” will be treated as different words even if they are used in the same sense.

1.4. Basic Feature Extraction – 2

After text cleaning, more feature extraction was done. These features were extracted after text cleaning because they are more meaningful to obtain at this phase. At this point, | tried to extract as many as features. We did not have to worry about whether the features will be useful in the future or not because having extra features cannot harm the text analysis in any way.

1.4.1. Number of Words: This feature tells how many words there are in the review**.**

1.4.2. Number of characters: How many letters are contained in the review

1.4.3. Average Word Length: Average number of letters in the words in a review

1.4.4. Data cleaning processes on 'title' column: Applying round 1 and round 2 data cleaning processes on the 'title' column. Keep in mind that round1 operations make text lowercase, remove text in square brackets, remove punctuation, and remove words containing numbers. And, round2 operations get rid of some additional punctuation and non-sensical text that was missed the first time around

1.5. Adding own stopwords:

Besides, after checking the most frequent works, “even”, “also” words appeared among the top 50 frequent words. Their words were also ignored.

2. Data Visualization

While exploring the data, we will look at the different combinations of features with the help of visuals. Visual will support us to understand our data better and discover hidden patterns in data. We can begin by checking descriptive statistics about the data. The descriptive statistic gives brief information about the numerical data and can help us to understand the shape of the data (Table 1).

Table 1

Table 1: Descriptive Information about features

	Id	rating	stopwords	punctuation	hastags	numerics	upper	word_count	char_count	avg_word
count	31666.0	31666.0	31666.0	31666.0	31666.0	31666.0	31666.0	31666.0	31666.0	31666.0
mean	15837.0	4.0	20.0	10.0	0.0	1.0	2.0	32.0	210.0	6.0
std	9144.0	2.0	30.0	15.0	0.0	2.0	6.0	44.0	286.0	3.0
min	1.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
25%	7919.0	3.0	3.0	1.0	0.0	0.0	0.0	7.0	47.0	5.0
50%	15838.0	4.0	11.0	5.0	0.0	0.0	0.0	19.0	126.0	6.0
75%	23756.0	5.0	25.0	12.0	0.0	1.0	2.0	40.0	261.0	6.0
max	31673.0	5.0	939.0	377.0	14.0	79.0	516.0	1443.0	9090.0	555.0

2.1. Loading saved processed file and Cleaning Outliers

2.2. Distribution analysis

2.2.1. Distribution of stopwords values:

The number of Stopwords is a feature that could only be obtained before data cleaning. From Figures 1 & 2, we see that most of the reviews have less than 500 stopwords. We can easily assume that more than 500 stopwords are extreme cases. For the stopwords feature, 25 is the third quantile value. So, I checked those reviews that have more than 300 stopwords (Figure 2). After checking reviews that have 300 reviews, there appeared still many same reviews. As a result, these reviews were considered as bad data and dropped from the data.

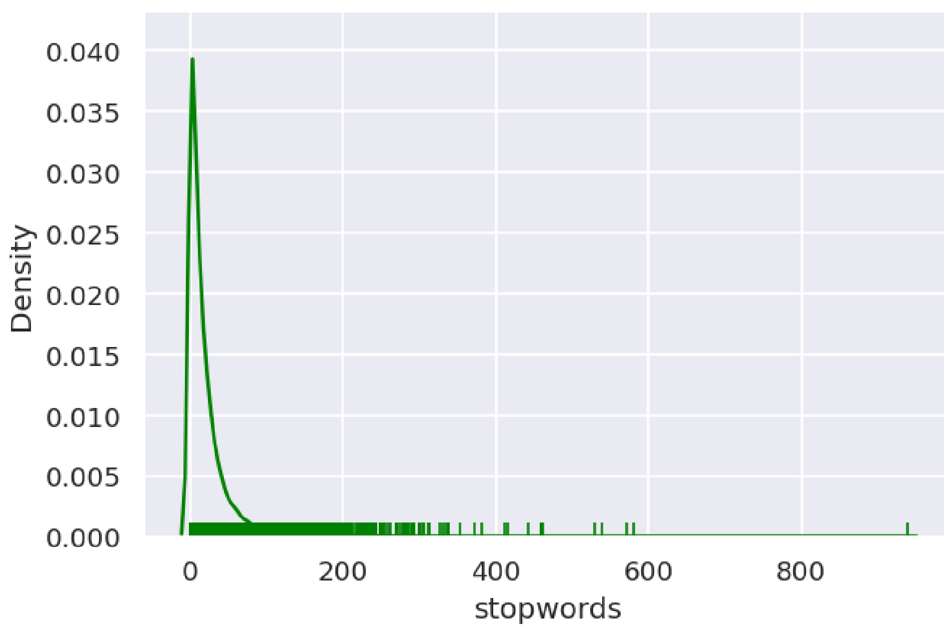


Figure 1 Density plot for stopwords

```
[ ]  df.loc[df.stopwords >= 500].review
```

```
11759    going big review detailed specs find anywhere ...
13698    exchanged redmi note said done think shouldnt ...
22885    buyed youtube channel review good phone review...
28023    samsung galaxy reviewover years samsung figure...
29684    started search samsung phone replace well perf...
Name: review, dtype: object
```

Figure 2 Reviews with more than 500 stopwords

```
[ ]  df.loc[df.stopwords >= 300].review
```

```
2045     reading lot geeky reviews youtube portals boug...
2368     using mi past years loved software experience ...
2620     oneplus excellent device different oneplus cou...
2916     using device since launched ie october review ...
2981     lengthy brutally honest review worth read plan...
3839     going long review looking summary beast phone ...
5644     oneplus uses snapdragon soc supports life gami...
6203     flagship killer flagship journey oneplus commu...
7429     first didnt purchased amazon ordered oneplusin...
7939     owning oneplus almost three years still rememb...
8711     review big blunder done oneplus amazoni ordere...
11582    moment unbox device feel high tier device one ...
11759    going big review detailed specs find anywhere ...
13698    exchanged redmi note said done think shouldnt ...
20479    xiaomis redmi series phones always offered gre...
21773    got note pro galcier white colour first sale s...
22885    buyed youtube channel review good phone review...
26558    galaxy reviewfirst similar device galaxy galax...
27865    lets start first strong point samsung galaxy d...
28023    samsung galaxy reviewover years samsung figure...
29280    reviews honest unbiased ill live smartphone re...
29539    first attached pics theyll exact ones youve se...
29684    started search samsung phone replace well perf...
30072    review days usageas easily excitable person ke...
30088    equipment delivered earlier stipulated time de...
Name: review, dtype: object
```

Figure 3 Reviews with more than 300 stopwords

2.2.2. Distribution of 'punctuation'

Similarly, in the case of punctuations, we choose 200 as the cut-off point for the 'punctuation' feature.

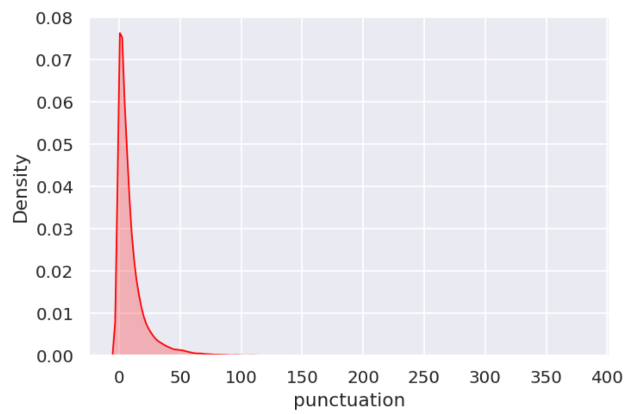


Figure 4 Density plot for punctuations

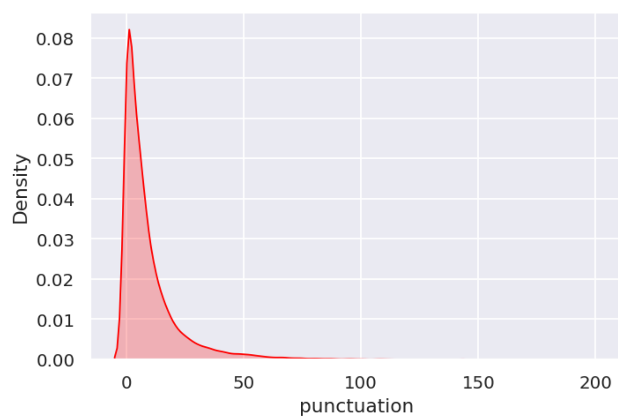


Figure 5 Density plot after removing punctuations above 200

2.2.3. Distribution of 'hashtags' values

Here, around 29000 reviews have 0 hashtags. As we can see from the bar graph, hashtags can be dropped because the dominant portion of the reviews has no hashtag.

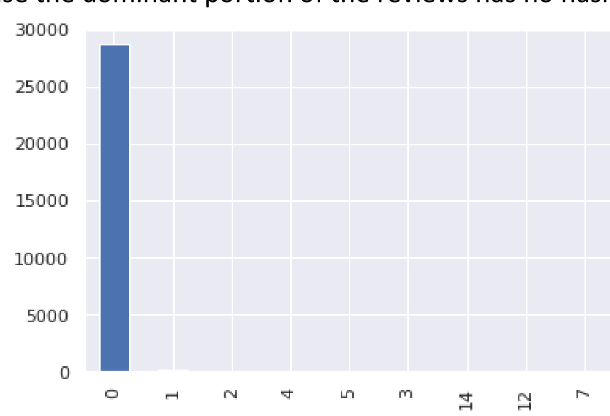


Figure 6 Number of # vs reviews

2.2.4. Distribution of 'numerics'

By looking at this plot, we see that most of the reviews have 6 or fewer numerics. However, we didn't drop any data here.

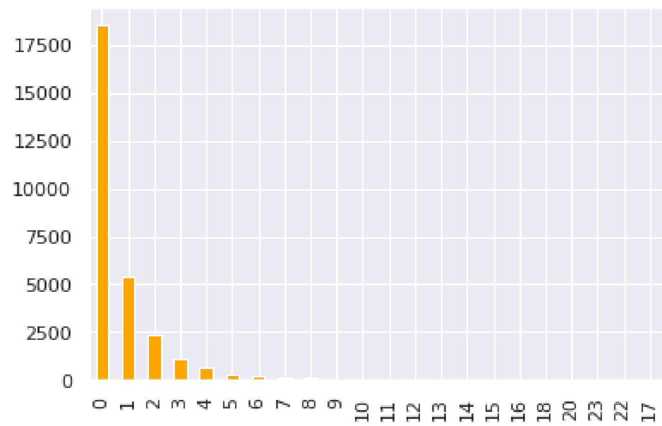


Figure 7 Distribution of numerics

2.2.5. Distribution of 'upper' values

There are 4 reviews with more than 150 upper case characters, Nothing suspicious here.

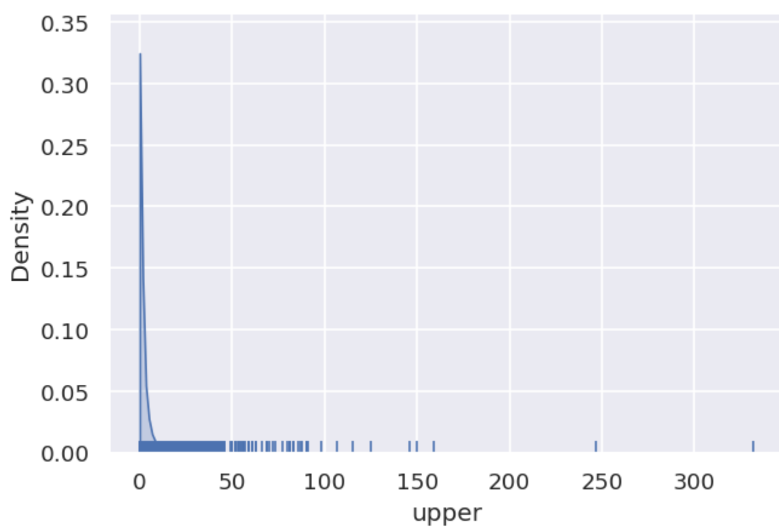


Figure 8 Distribution of upper values

2.2.6. Distribution of word_count values

There are 13 reviews with a word count of 400+ and 6 reviews with a word count of 450+. No need to drop anything.

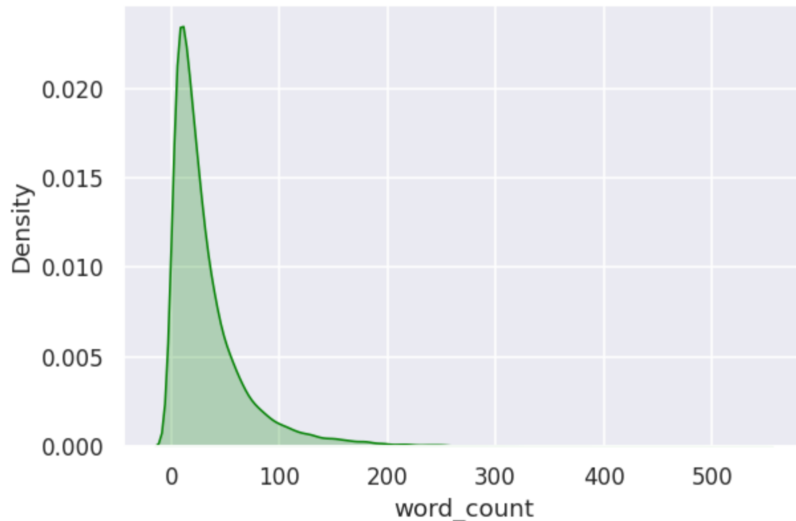


Figure 9 Distribution of word count

2.2.7. Distribution of char_count values

There are 6 reviews with a character count of 3000. No need to drop anything.

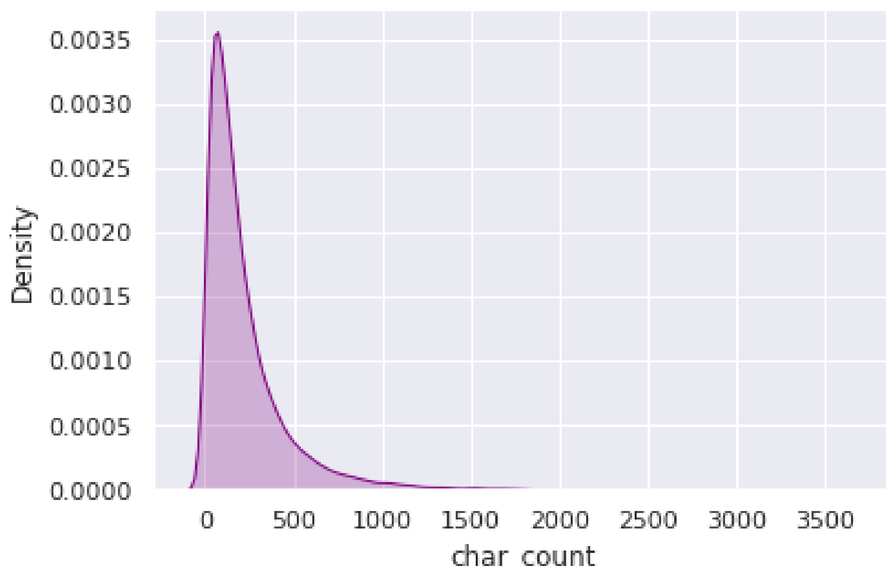


Figure 10 Distribution of char_count values

2.2.8. Distribution of avg_word length

We dropped reviews with an average word length of 15 (5 reviews) because they were not making any sense.

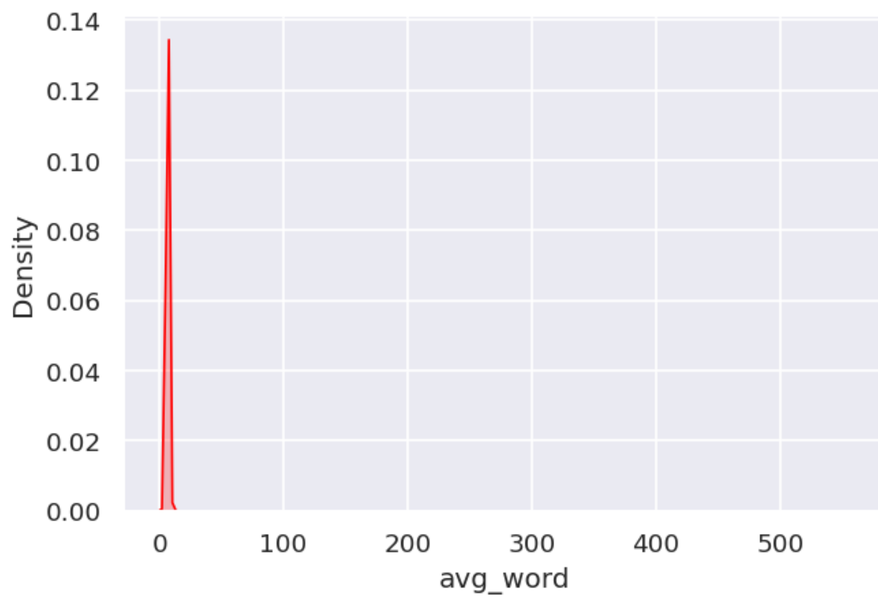
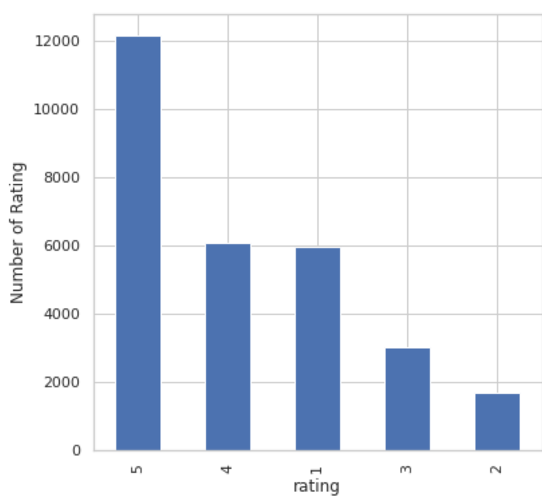
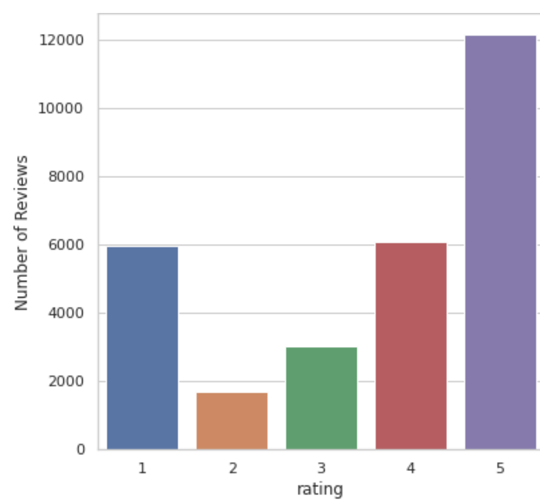


Figure 11 Distribution of average word length

2.2.9. Distribution of rating

After the deduction of reviews, we are left with 28897 reviews. Now we look at the Distribution of ratings 1 to 5

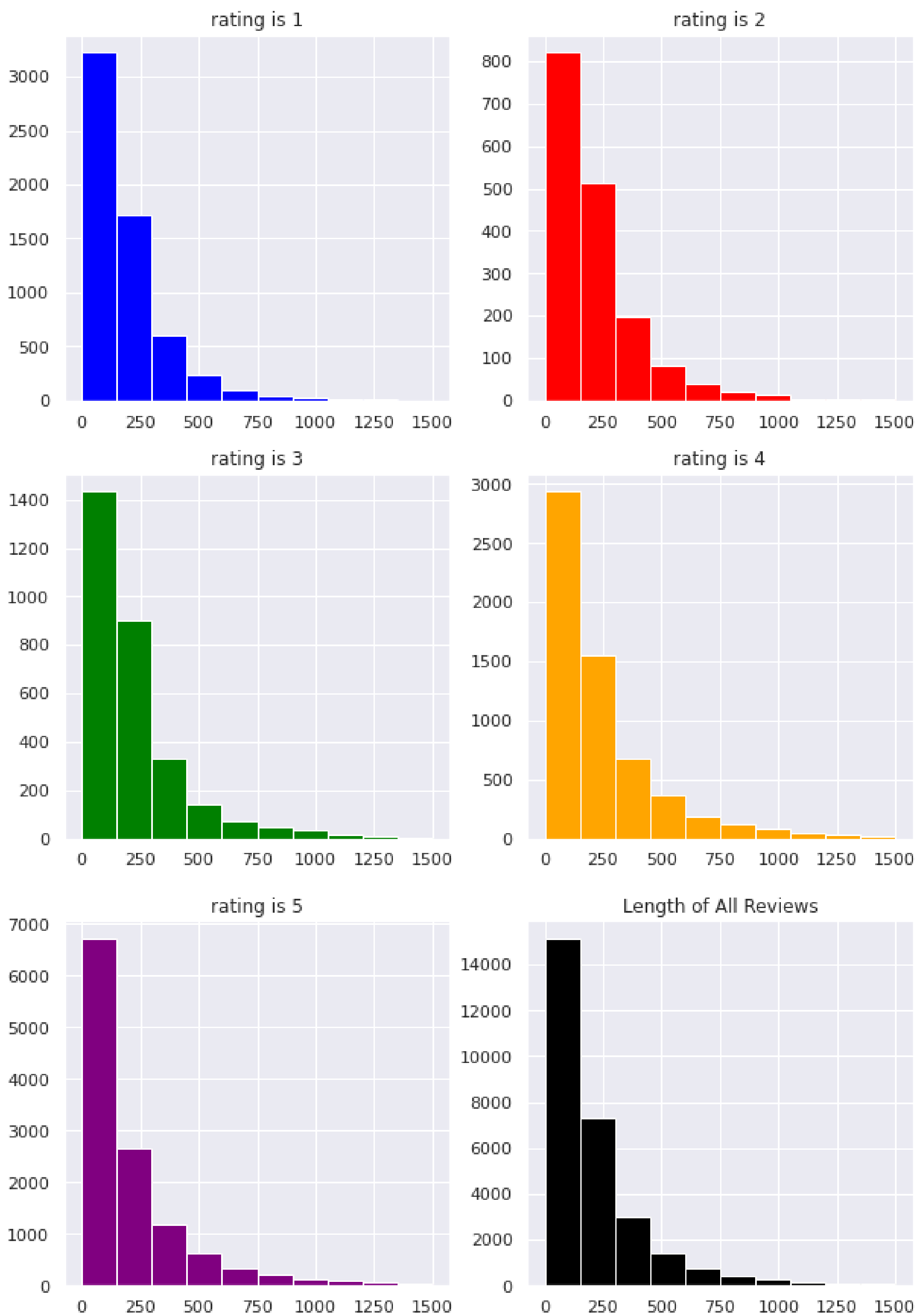




Most of the reviews are 5 stars. This is an unbalanced distribution, but as the database is taken from bestselling smartphones, the graph representation is accurate.

2.3. Number of Characters in Reviews

2.3.1. Number of Characters in review for each rating



We do not see any difference between ratings 1 to 5.

2.3.2. Distribution of good reviews

After this point, we will convert 'review' to a binary feature.

Score values 1, 2, and 3 will be coded as 0 (zero)

Score values 4 and 5 will be coded as 1

The good reviews column is created from the rating column. Examining distribution good reviews may help us to understand how this column was shaped. From Figure 12, it can be seen that this is unbalanced data. (which is obvious because it is a bestselling smartphone data) This is an important point to keep in mind because while making predictions, model performance evaluation is different in balanced and unbalanced data.

Figure 12 Distribution of good reviews

2.3.3. Distribution of Number of Words for Reviews

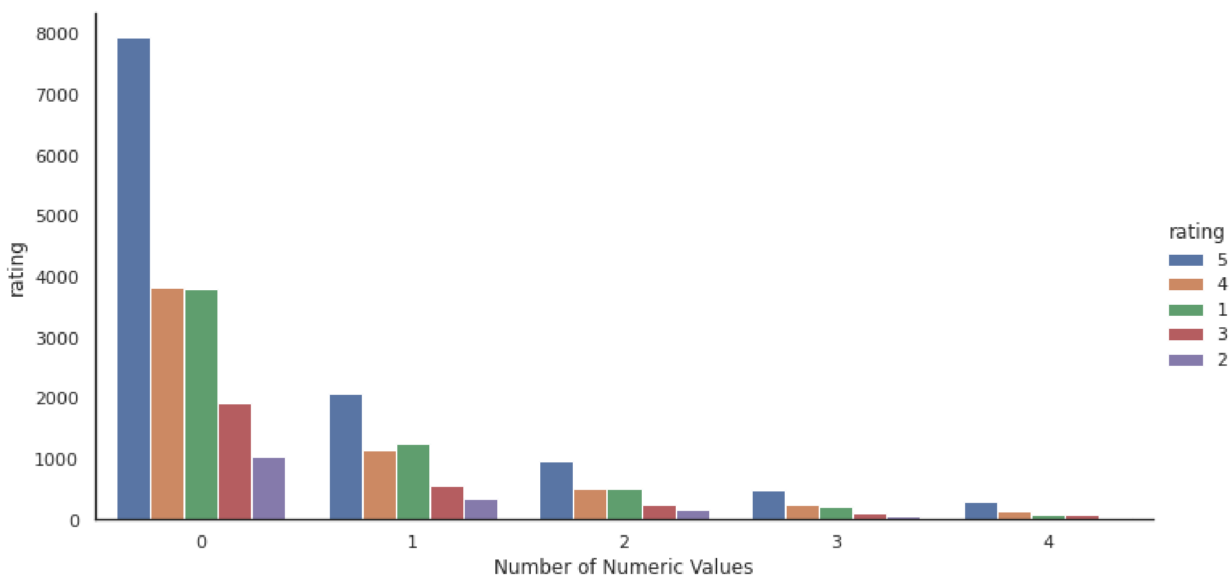


Figure 13 Number of numeric values depending on Ratings

This is a blowup of Figure 7 but divided into ratings. We can see that reviews with at least 1 numeric value can be distributed by ratings 5-1-4-3-2. Similarly, we have blown up Figure 7 into Good and Bad reviews.

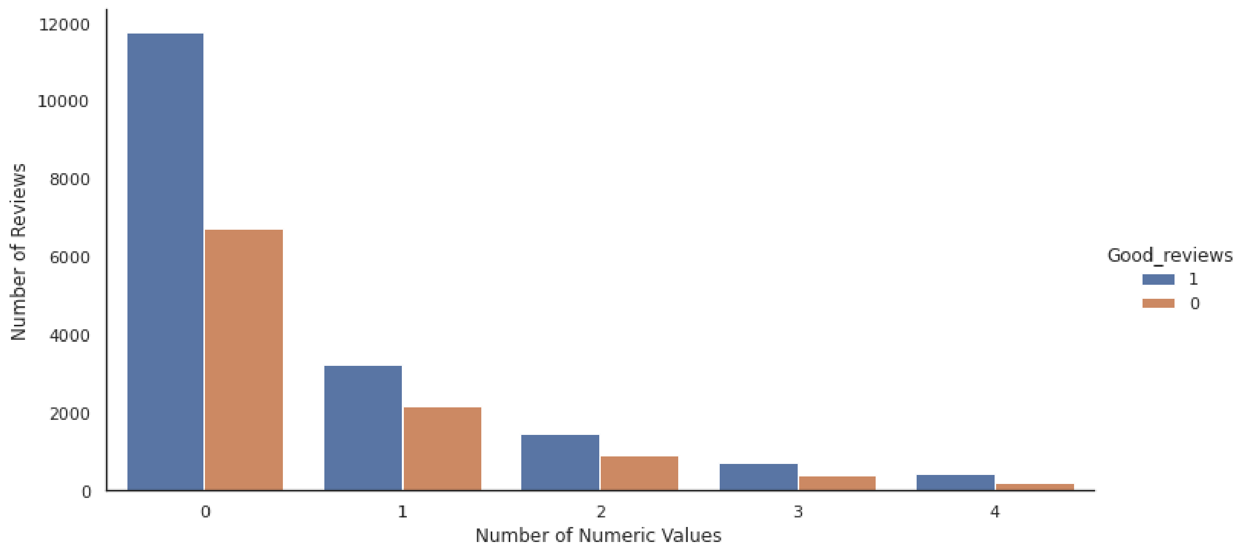


Figure 14 Distribution of numeric values depending on Good Reviews category

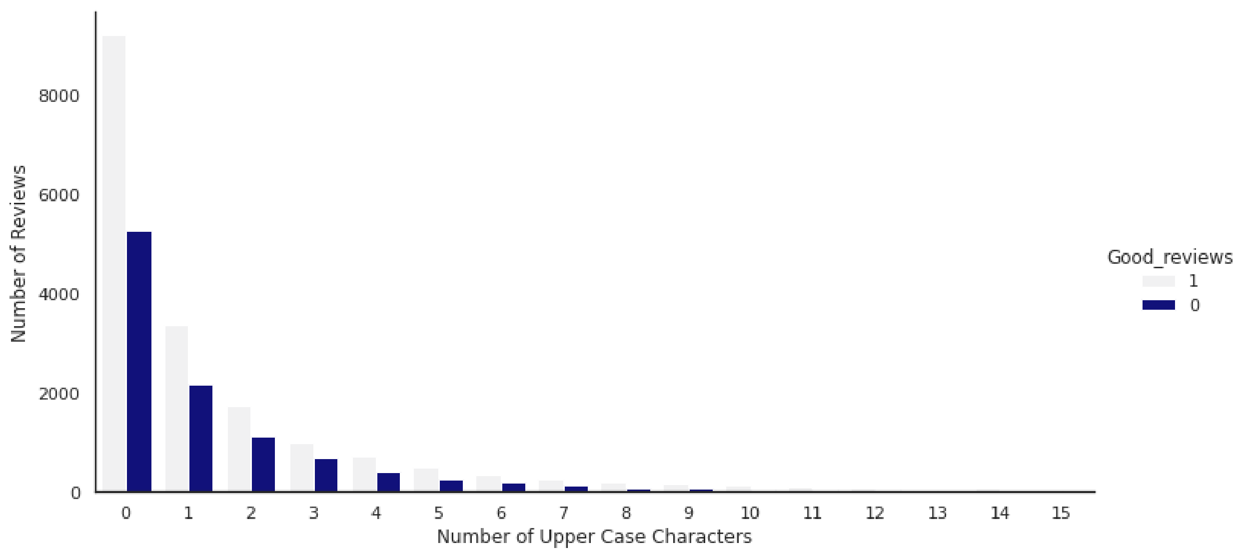


Figure 15 Number of upper-case characters for reviews

Figure 15 implies that most people do not pay much attention to grammar rules while writing a review.

2.4. Historic distribution of reviews

2.4.1. Month-wise distribution

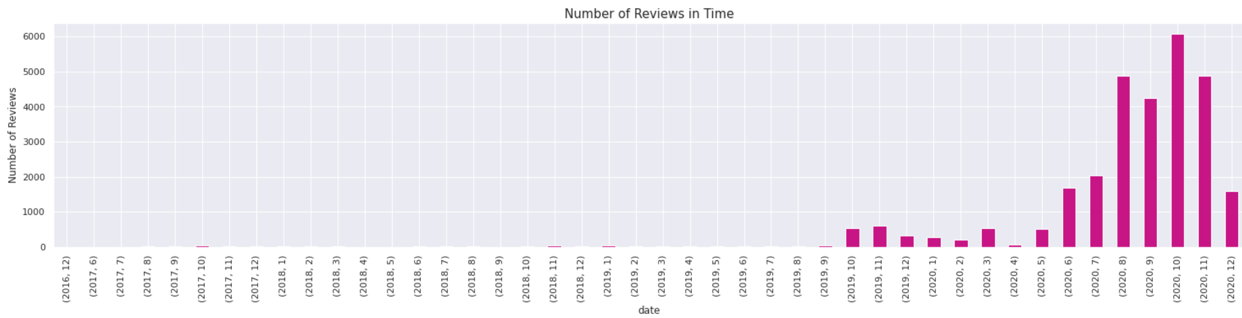


Figure 16 Month-wise distribution of all reviews

The reviews are distributed from December of 2016 to December of 2020. This is odd as the bestselling smartphones listed are the new models launched in the last 2 years.

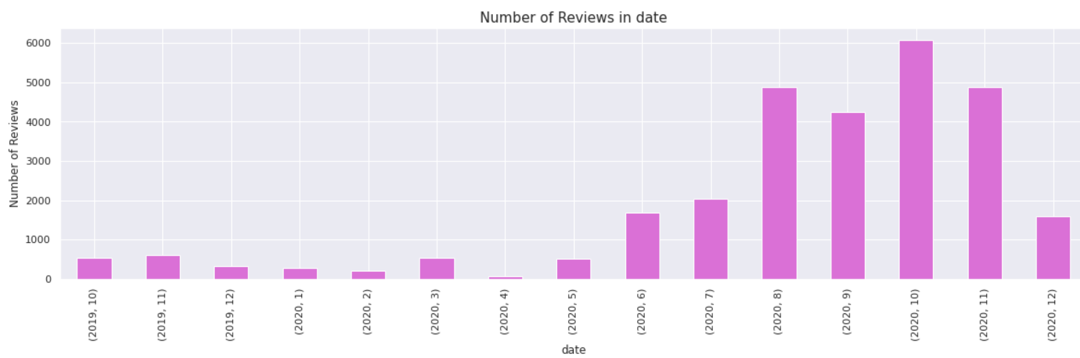


Figure 17 Month wise review distribution between 2019 Oct to present

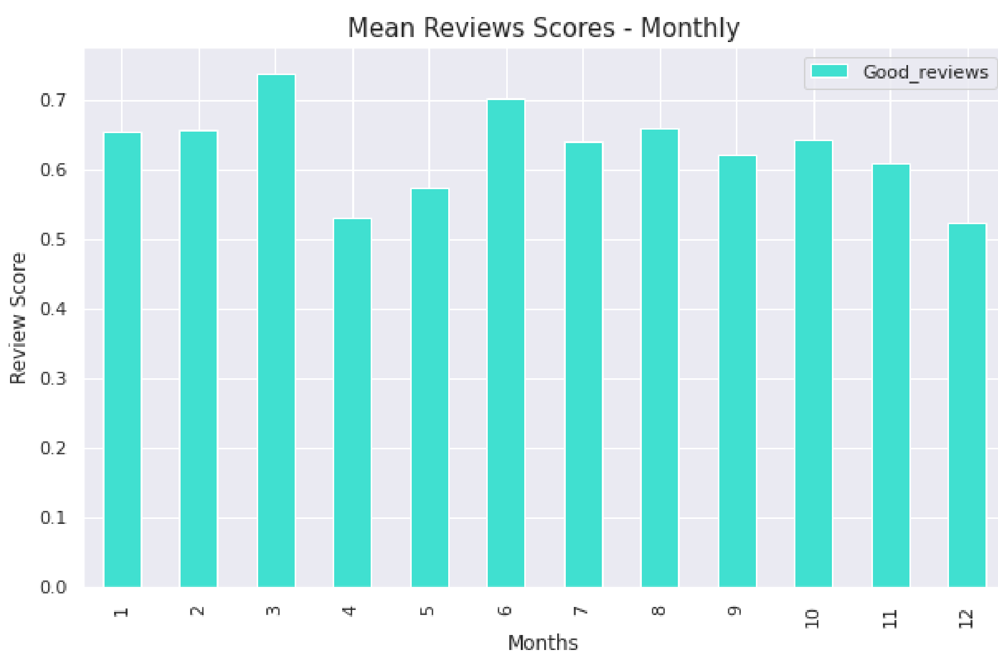


Figure 18 Mean good review score depending on Months

As we can see, the mean of "good reviews' for each month varies. But more of the months have a good review score between 0.6 to 0.7. April, May, and December have scored less than 0.6, on the other hand, March has a score above 0.7

2.4.2. Mean of the Scores over time

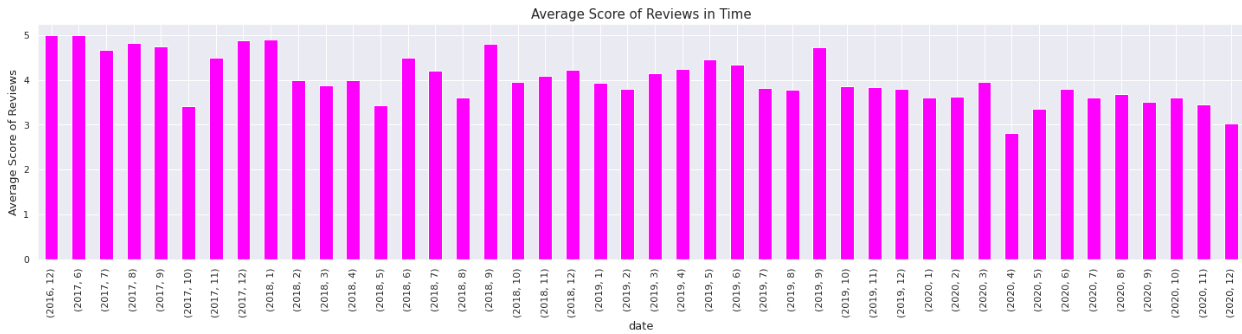


Figure 19 Mean good review score in time

Here, the mean of the score has decreased gradually with time.

2.4.3. Most frequent words

Figure 20 represents the most frequent 100 words in the reviews. When we examine, mobile, phone, battery, life appears most as the data is about 100 bestselling smartphones on Amazon.

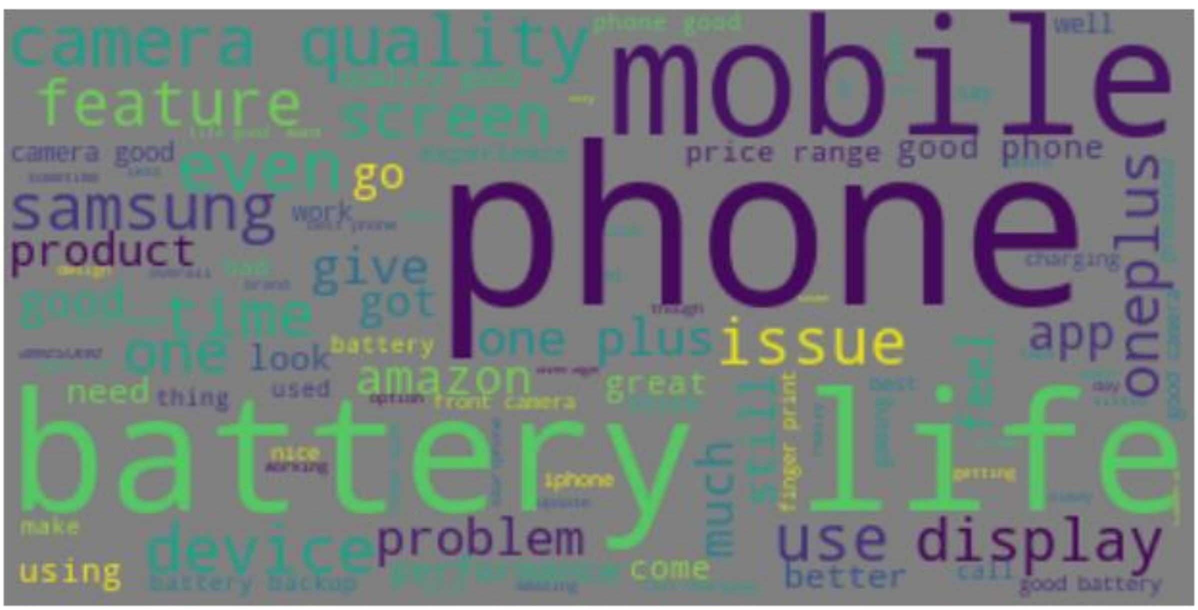


Figure 20 Most frequent 100 words in reviews

3. Sentiment Analysis

Before making predictions based on machine learning models, we need to understand the data better. The reviews are unstructured. In other words, the text is unorganized. Sentiment analysis, however, helps us make sense of all this unstructured text by automatically tagging it. Sentiment analysis helps us to process huge amounts of data efficiently and cost-effectively. That's why sentiment analysis was applied to the text data.

In this study, two main sentiment classifiers were used:

1. Polarity
2. Subjectivity

The TextBlob package for Python is a convenient way to perform many Natural Language Processing (NLP) tasks. For this study, the TextBlob package was used for sentiment analysis. When calculating sentiment for a single word, TextBlob takes an average for the entire text. For heteronym words, TextBlob does not negotiate with different meanings. In the other words, only the most common meaning of a word in the entire text is taken into consideration. For making all these models, TextBlob uses the WordNet Database of Princeton University. WordNet is a large lexical database of English. In this database, nouns, verbs, adjectives, and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked through conceptual-semantic and lexical relations (Fellbaum, 1998). This is the brief background information about how sentiment analysis of this study works. Now, here is the sentiment analysis

3.1. Creating 'Subjectivity' and 'Polarity' Scores

Polarity

Polarity is a float that lies in the range of [-1,1] where 1 means a positive statement and -1 means a negative statement.

Subjectivity

Subjective sentences generally refer to personal opinion, emotion, or judgment whereas objective refers to factual information. Subjectivity is also a float that lies in the range of [0,1].

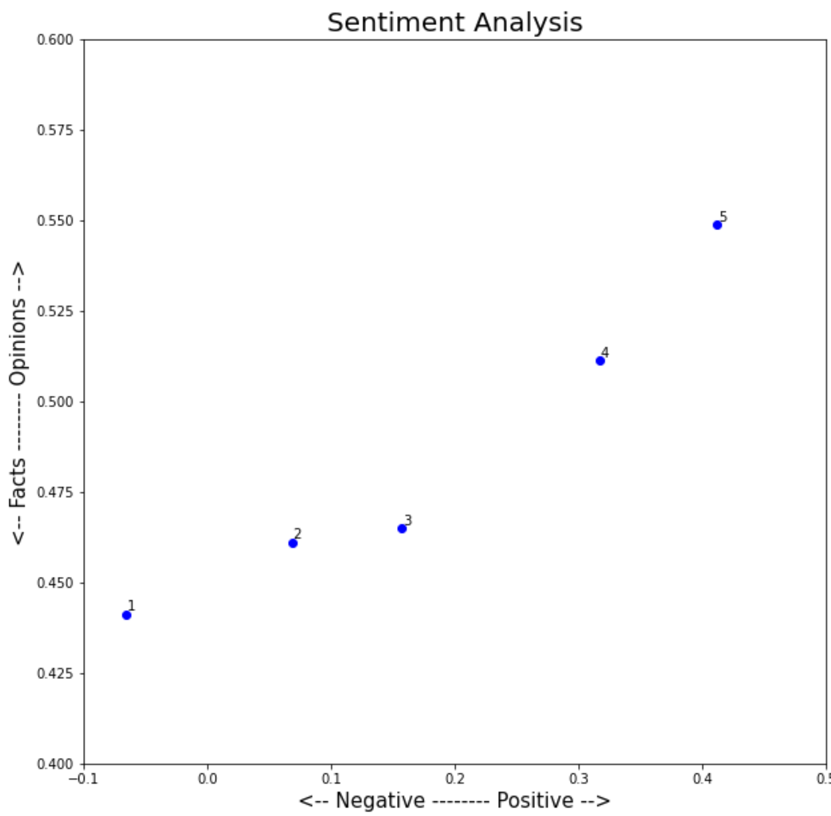


Figure 21 Mean value of polarity and subjectivity scores for review ratings

When we check the mean value of polarity and subjectivity score for review ratings, we can see that the mean subjectivity score and polarity are both higher for higher ratings.

3.2. Examining polarity

Figure 22 shows the distribution of polarity scores in reviews. Most of the reviews are on the positive side of the plot.

In Figure 23, it can be observed that good reviews have higher polarity compared to bad reviews.

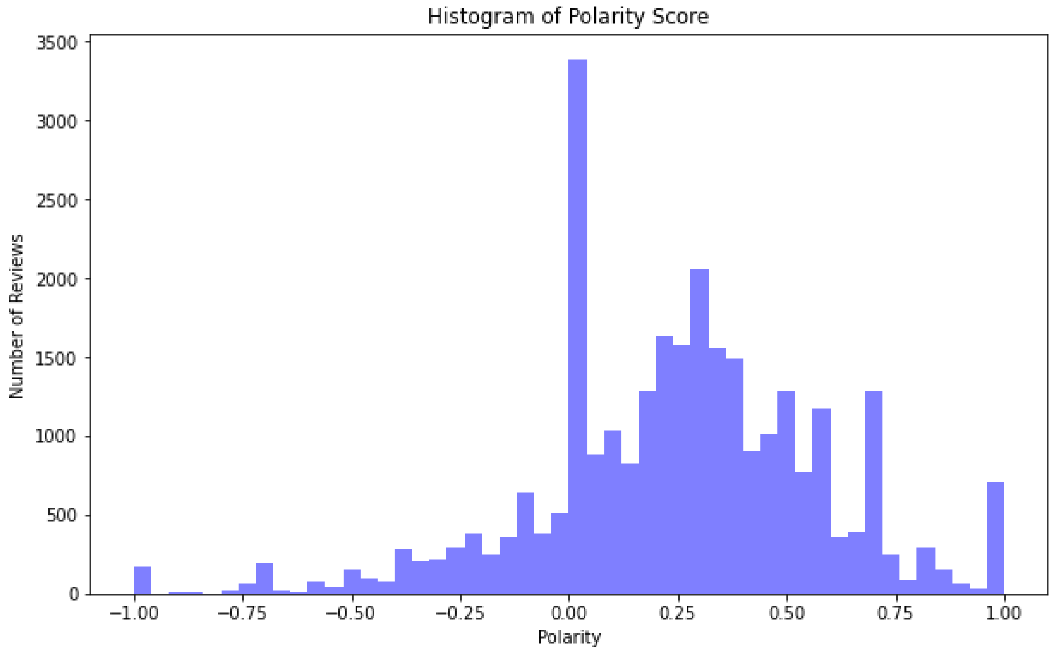


Figure 22 Distribution of polarity score for reviews

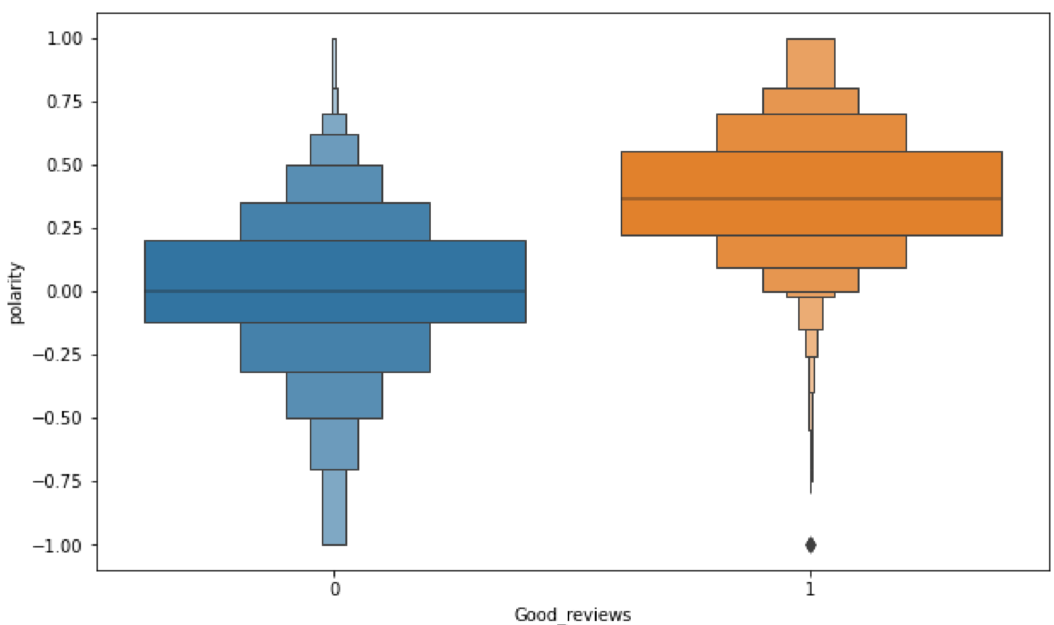


Figure 23 Boxplot of distribution of polarity regarding good or bad review

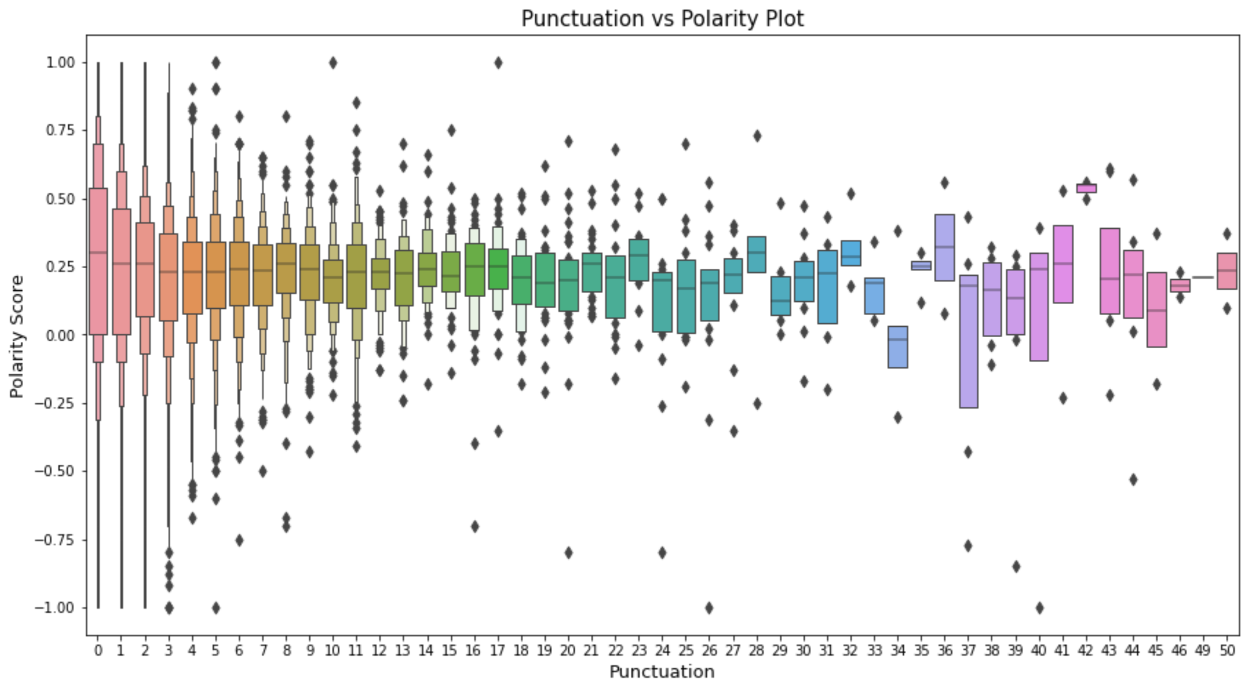


Figure 24 Number of punctuation and polarity score

Punctuation vs Polarity: From figure 24, we can see that when the number of punctuations is low, polarity is higher. A possible explanation for this is people who are paying more attention to punctuation tend to be more balanced in their product evaluation. Despite outliers, the average polarity score is almost a line and it is around 0.25.

3.3. Polarity vs Number of Words

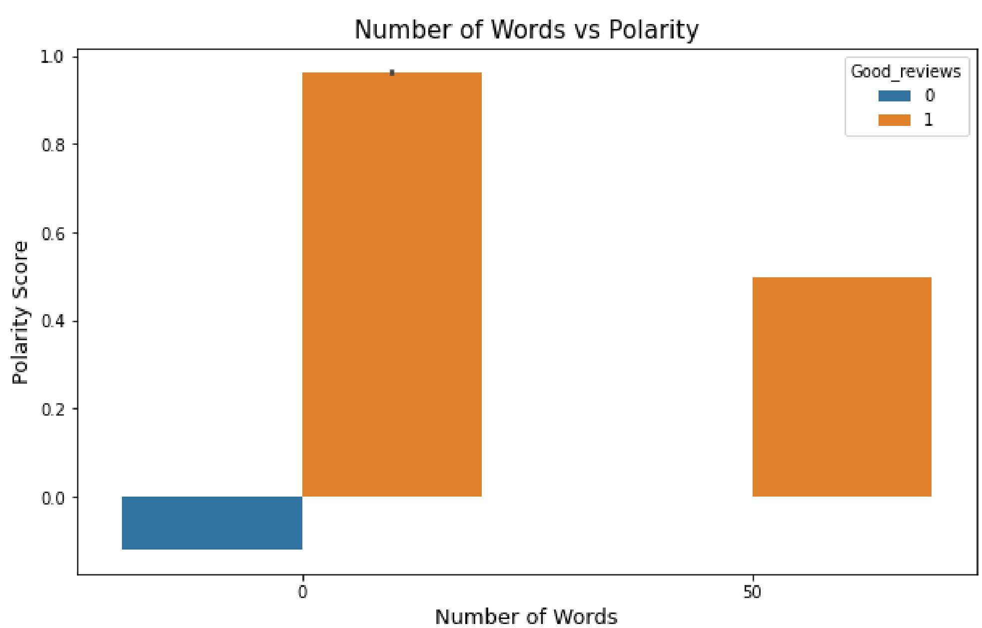


Figure 25 Polarity score and number of words hued to good reviews category

The number of words represents how many words exist in the review. In this aspect, Figure 25 offers a very interesting understanding. In Figure 25, we categorized the number of words into 2 categories. If the number of words is more than 50, we don't have any bad reviews. The reason behind this is it looks like people do not bother to spend effort on writing too long for a bad review. That can be the reason why we don't have long bad reviews (more than 50 words).

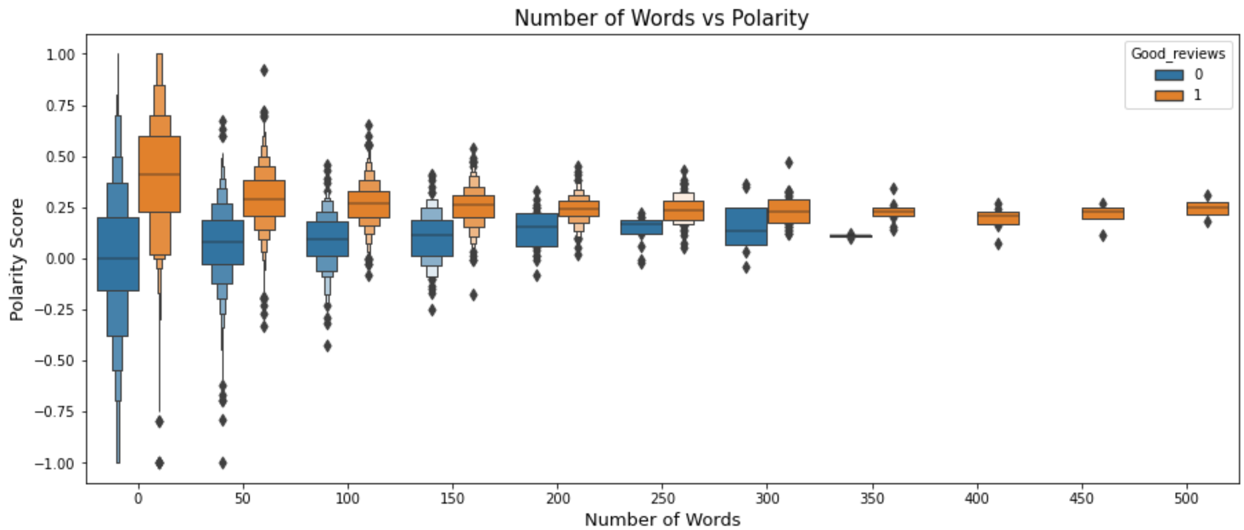


Figure 26 Number of words vs a polarity

Figure 26 is a blowup of Figure 23, where we break the polarity of good and bad reviews by the number of words. There are no bad reviews having words more than 300.

3.4. Examining Subjectivity

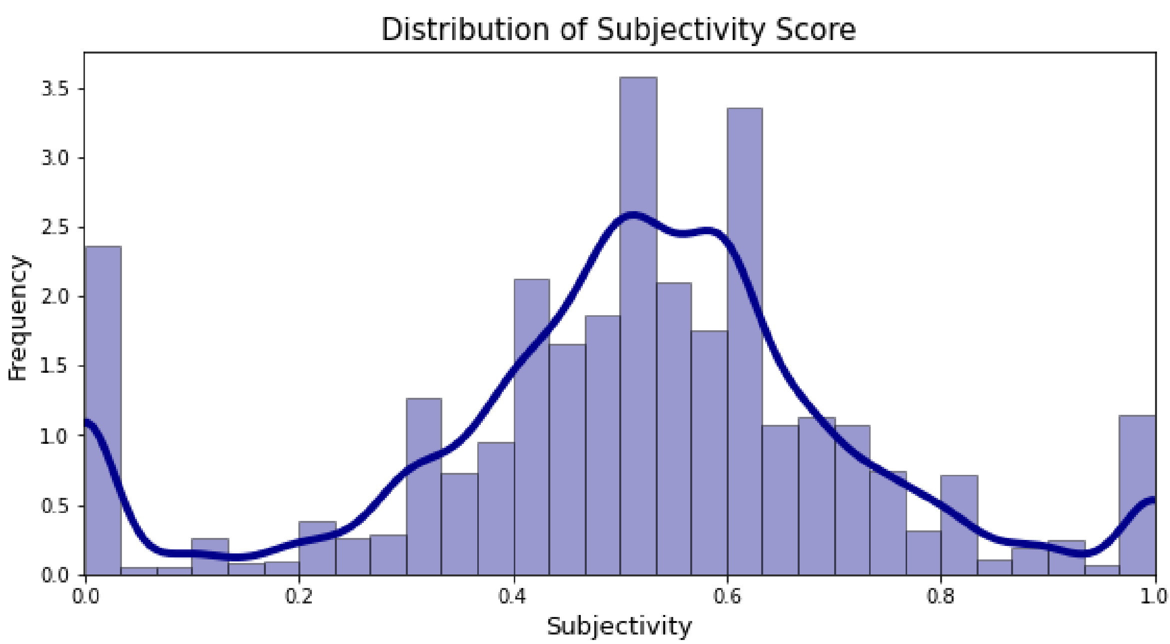


Figure 27 Distribution of subjectivity scores in Amazon reviews

In sentiment analysis, subjectivity is also a float which lies in the range of [0,1]. When it is close to 0, it is more about facts. When subjectivity increases, it comes close to be an opinion. In the dataset, the distribution of subjectivity scores for the reviews is similar to the normal distribution (Fig. 27). When we examined the relationship between subjectivity, polarity, and Good Reviews features we can see that subjectivity and polarity shows a funneling pattern (Fig. 28). It can also be observed that low subjectivity score reviews are also neutral reviews in terms of polarity.

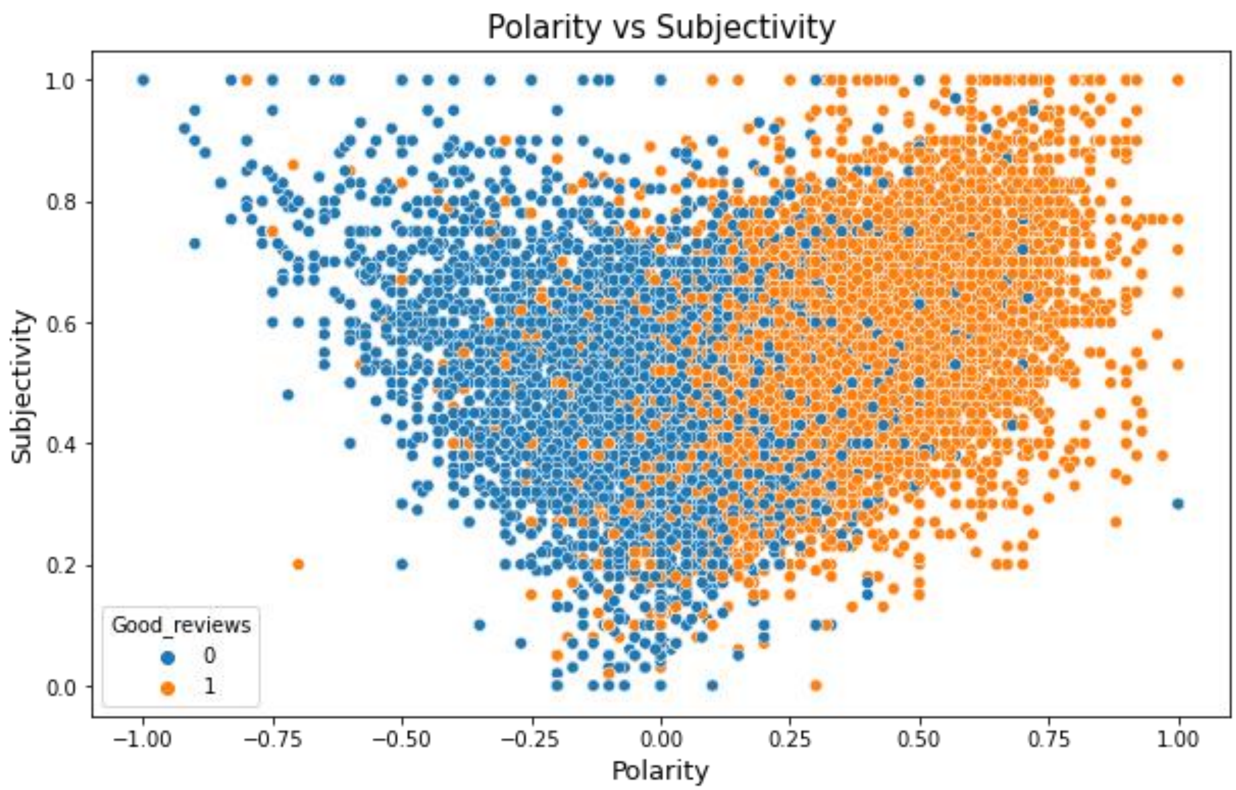


Figure 28 Subjectivity and polarity scores hues to good reviews category

```
# Top 10 reviews that have highest polarity (most positive sentiment) but 'good review' value is 0, and most
df.loc[(df["Good_reviews"] == 0) & (df.polarity == 1 ) & (df.subjectivity ==1), "review"].head(10).tolist()
```

```
['perfect size phone hand battery backup mark',
 'heating issue using greatest cameras',
 'everything awesome phone display problem make flop phone year',
 'excellent device except camera',
 'phone excellent aspects except camera pixel iphone user camera unacceptable',
 'battery drain highbattery consuming power didnt use phonebattery consuming sleepi want charge phone times dayif battery improvenord superb',
 'everything perfect ref tint issue started appearing months use',
 'price superb',
 'gorila glass provided manufacturer camera goodhalf prize oppo gb ram excellent camera',
 'perfect mobile beginner level smartphone battery backup problem processer otherwise theres issues']
```

Figure 29 Reviews that have a polarity of 1 (most positive sentiment) and subjectivity (opinion) of 1 and bad rating (good review is 0)

Figure 29 represents an extreme case where the test is positive, but the rating is bad. These tweets are hard to score for the sentiment analysis algorithm.

```
# 5 sample reviews that have highest polarity (most positive sentiment) and 'good review' value is 1, and most subjective (opinion):
df.loc[(df["Good_reviews"] == 1) & (df.polarity == 1 ) & (df.subjectivity ==1), "review"].sample(5).tolist()
```

```
['battery superb excellent brand mi evergreen mi compan',
 'excellent performancebest price',
 'superb phone oneplus',
 'awesome device used around two months working perfectl',
 'awesome phone pricerange']
```

```
# Top 10 reviews that have lowest polarity (most negative sentiment) but 'good review' value is 1, and most subjective (opinion):
df.loc[(df["Good_reviews"] == 1) & (df.polarity == -1 ) & (df.subjectivity ==1), "review"].head(10).tolist()
```

```
['battery horrible',
 'camera quality worst think oneplus need work',
 'finger print sensor worst couldnt find adaptor type c pin jack box mentioned description',
 'camera picture quality worst recommended']
```

Figure 30 Good rating post and most subjective reviews with the highest and lowest polarity

As we can see, some of these reviews got negative sentiment but good review scores.

4. Topic Modeling

Topic modeling is another popular text analysis technique. The goal of topic modeling to find a theme across reviews and discover hidden topics. Each document in the corpus will be made up of at least one topic, if not multiple topics.

In this part, I will be covering the results of Latent Dirichlet Allocation (LDA), which is one of many topic modeling techniques. LDA was specifically designed for text data.

To use a topic modeling technique, you need to provide (1) a document-term matrix and (2) the number of topics you would like the algorithm to pick up. In topic modeling, | will create different models and compare them. In the end, I will choose the topic model that makes the most sense.

During LDA Topic modeling, we are the one who decides the number of groups in the output. However, we do not know what the best number of groups is. Therefore, we will obtain different numbers of groups. Then, the one that makes the most sense will be chosen among all groups. Once the topic modeling technique is applied, the researcher's job as a human is to interpret the results and see if the mix of words in each topic makes sense. If they don't make sense, we can try changing up the number of topics, the terms in the document- term matrix, model parameters, or even try a different model.

4.1. Importing the data

Data is imported after the sentiment analysis.

4.2. Topic Modeling with Good reviews (Good reviews =1)

In this part, topic models in good ratings will be examined.

4.2.1. Topic Modeling - Attempt #1 (with all review data)

No text filtering was applied in this process. By looking at Table 2, it can be said that Topic group 0 from Modeling with 3 Topics make the most sense, which contains 7.3% “good” followed by 4.4% “camera”, 3.3% “phone”, 3.1% “battery”, 3% “quality.

Table 2 Topic modeling with all text data

```python
# Now that we have the corpus (term-document matrix) and id2word (dictionary of location: term),
# we need to specify two other parameters as well - the number of topics and the number of passes
lda = models.LdaModel(corpus=corpus, id2word=id2word, num_topics=2, passes=10)
lda.print_topics()
```

```
[(0,
  '0.022*"camera" + 0.022*"phone" + 0.022*"good" + 0.017*"battery" + 0.009*"display" + 0.009*"quality" + 0.008*"oneplus" + 0.008*"one" + 0.007*"fast" + 0.007*"life"'),
 (1,
  '0.041*"phone" + 0.036*"good" + 0.016*"product" + 0.014*"camera" + 0.014*"mobile" + 0.014*"best" + 0.013*"price" + 0.013*"one" + 0.013*"quality" + 0.012*"nice"')]
```

```python
# LDA for num_topics = 3
lda = models.LdaModel(corpus=corpus, id2word=id2word, num_topics=3, passes=10)
lda.print_topics()
```

```
[(0,
  '0.073*"good" + 0.044*"camera" + 0.033*"phone" + 0.031*"battery" + 0.030*"quality" + 0.017*"life" + 0.016*"nice" + 0.015*"mobile" + 0.013*"price" + 0.011*"fast"'),
 (1,
  '0.024*"phone" + 0.017*"camera" + 0.015*"good" + 0.011*"battery" + 0.011*"oneplus" + 0.009*"display" + 0.009*"one" + 0.007*"like" + 0.007*"screen" + 0.006*"great"'),
 (2,
  '0.036*"phone" + 0.021*"one" + 0.019*"product" + 0.018*"amazon" + 0.017*"best" + 0.012*"plus" + 0.009*"delivery" + 0.009*"mobile" + 0.008*"price" + 0.008*"buy"')]
```

```python
# LDA for num_topics = 4
lda = models.LdaModel(corpus=corpus, id2word=id2word, num_topics=4, passes=10)
lda.print_topics()
```

```
[(0,
  '0.026*"value" + 0.016*"mone" + 0.012*"money" + 0.005*"bes" + 0.003*"segmen" + 0.003*"quali" + 0.003*"worth" + 0.003*"nd" + 0.003*"phn" + 0.002*"produc"'),
 (1,
  '0.033*"product" + 0.030*"phone" + 0.027*"good" + 0.022*"amazon" + 0.019*"mobile" + 0.018*"nice" + 0.015*"one" + 0.011*"delivery" + 0.010*"price" + 0.008*"got"'),
 (2,
  '0.037*"phone" + 0.018*"oneplus" + 0.013*"best" + 0.010*"like" + 0.010*"go" + 0.008*"iphone" + 0.008*"one" + 0.007*"price" + 0.007*"buy" + 0.006*"phones"'),
 (3,
  '0.038*"good" + 0.033*"camera" + 0.026*"phone" + 0.024*"battery" + 0.017*"quality" + 0.011*"life" + 0.010*"display" + 0.009*"one" + 0.009*"fast" + 0.008*"performance"')]
```

4.2.2. Topic Modeling - Attempt #2 (Nouns Only)

In this step, only nouns were used for creating topics by using the LDA method.

Table 3 Topic modeling with Nouns

```python
# Let's start with 2 topics
ldan = models.LdaModel(corpus=corpusn, num_topics=2, id2word=id2wordn, passes=10)
ldan.print_topics()
```

```
[(0,
  '0.076*"phone" + 0.036*"camera" + 0.013*"display" + 0.011*"price" + 0.010*"issue" + 0.010*"use" + 0.009*"quality" + 0.008*"performance" + 0.008*"device" + 0.008*"day"'),
 (1,
  '0.056*"battery" + 0.046*"camera" + 0.037*"quality" + 0.035*"phone" + 0.032*"life" + 0.026*"product" + 0.020*"price" + 0.014*"performance" + 0.011*"display" + 0.010*"value"')]
```

```python
# Let's try topics = 3
ldan = models.LdaModel(corpus=corpusn, num_topics=3, id2word=id2wordn, passes=10)
ldan.print_topics()
```

```
[(0,
  '0.086*"phone" + 0.050*"camera" + 0.044*"battery" + 0.037*"quality" + 0.027*"life" + 0.022*"product" + 0.017*"price" + 0.011*"performance" + 0.010*"amazon" + 0.009*"iphone"'),
 (1,
  '0.037*"phone" + 0.028*"product" + 0.027*"value" + 0.020*"price" + 0.017*"money" + 0.012*"mone" + 0.012*"range" + 0.010*"mobile" + 0.009*"produc" + 0.006*"box"'),
 (2,
  '0.049*"phone" + 0.044*"camera" + 0.022*"display" + 0.022*"battery" + 0.016*"quality" + 0.013*"performance" + 0.012*"price" + 0.011*"day" + 0.011*"experience" + 0.010*"device"')]
```

```python
# Let's try 4 topics
ldan = models.LdaModel(corpus=corpusn, num_topics=4, id2word=id2wordn, passes=10)
ldan.print_topics()
```

```
[(0,
  '0.031*"iphone" + 0.015*"apple" + 0.011*"smartphone" + 0.009*"card" + 0.009*"phones" + 0.008*"products" + 0.007*"mi" + 0.007*"ios" + 0.006*"box" + 0.005*"user"'),
 (1,
  '0.045*"value" + 0.036*"money" + 0.034*"battery" + 0.030*"life" + 0.028*"camera" + 0.025*"quality" + 0.022*"product" + 0.020*"print" + 0.018*"finger" + 0.015*"mone"'),
 (2,
  '0.095*"phone" + 0.056*"product" + 0.022*"amazon" + 0.021*"delivery" + 0.012*"price" + 0.010*"thanks" + 0.010*"produc" + 0.010*"features" + 0.009*"problem" + 0.008*"service"'),
 (3,
  '0.068*"phone" + 0.056*"camera" + 0.034*"battery" + 0.028*"quality" + 0.019*"display" + 0.018*"price" + 0.015*"life" + 0.015*"performance" + 0.011*"day" + 0.010*"use"')]
```

4.2.3. Topic Modeling - Attempt #3 (Nouns and Adjectives)

In this step, nouns and adjectives were used for creating topics by using the LDA method.

Table 4 Topic modeling with nouns and adjectives

```
# Let's start with 2 topics
ldana = models.LdaModel(corpus=corpusna, num_topics=2, id2word=id2wordna, passes=10)
ldana.print_topics()

[(0,
  '0.047*"phone" + 0.028*"camera" + 0.024*"battery" + 0.023*"good" + 0.013*"quality" + 0.012*"great" + 0.011*"life" + 0.011*"best" + 0.011*"display" + 0.010*"oneplus"'),
 (1,
  '0.066*"good" + 0.024*"camera" + 0.022*"phone" + 0.020*"product" + 0.018*"nice" + 0.018*"mobile" + 0.017*"quality" + 0.011*"price" + 0.006*"awesome" + 0.006*"amazon"')]
```

```
# Let's try 3 topics
ldana = models.LdaModel(corpus=corpusna, num_topics=3, id2word=id2wordna, passes=10)
ldana.print_topics()

[(0,
  '0.038*"good" + 0.034*"camera" + 0.030*"phone" + 0.026*"battery" + 0.015*"quality" + 0.013*"life" + 0.011*"display" + 0.010*"screen" + 0.009*"great" + 0.008*"performance"'),
 (1,
  '0.045*"product" + 0.033*"good" + 0.027*"mobile" + 0.019*"amazon" + 0.017*"phone" + 0.014*"nice" + 0.013*"delivery" + 0.010*"value" + 0.008*"iphone" + 0.008*"time"'),
 (2,
  '0.077*"phone" + 0.034*"good" + 0.028*"price" + 0.026*"camera" + 0.025*"best" + 0.019*"quality" + 0.013*"range" + 0.011*"nice" + 0.010*"oneplus" + 0.010*"great"')]
```

```
# Let's try 4 topics
ldana = models.LdaModel(corpus=corpusna, num_topics=4, id2word=id2wordna, passes=10)
ldana.print_topics()

[(0,
  '0.025*"camera" + 0.021*"phone" + 0.014*"good" + 0.012*"oneplus" + 0.012*"display" + 0.011*"screen" + 0.011*"battery" + 0.009*"device" + 0.009*"great" + 0.008*"day"'),
 (1,
  '0.044*"mobile" + 0.038*"value" + 0.025*"good" + 0.024*"money" + 0.019*"product" + 0.013*"mone" + 0.009*"nice" + 0.007*"amazon" + 0.006*"exchange" + 0.005*"gud"'),
 (2,
  '0.069*"good" + 0.062*"phone" + 0.040*"camera" + 0.033*"battery" + 0.028*"quality" + 0.017*"price" + 0.016*"life" + 0.011*"performance" + 0.011*"nice" + 0.011*"best"'),
 (3,
  '0.044*"phone" + 0.026*"product" + 0.022*"amazon" + 0.018*"delivery" + 0.016*"best" + 0.013*"iphone" + 0.009*"apple" + 0.008*"time" + 0.008*"great" + 0.007*"worth"')]
```

Now, in the final stage, Table 11, Table 12, and Table 13 must be evaluated. We need to ask our self 'which group makes more sense?'. Out of the 9 examined topic models, nouns only, a 3-topic group made the most sense to me (Table 11). | see three distinct groups here: (1) pet items, (2) cookies and snacks, and (3) beverage. Keep in mind that this dataset consists of only food reviews. So, it quite normal to see that the groups are related to foods.
So, let's pull that down here and run it through some more iterations to get more fine-tuned topics.

Table 5 Final topic modeling with tuned parameters with nouns only

```
# Let's start with 2 topics
ldana = models.LdaModel(corpus=corpusna, num_topics=2, id2word=id2wordna, passes=10)
ldana.print_topics()

[(0,
  '0.047*"phone" + 0.028*"camera" + 0.024*"battery" + 0.023*"good" + 0.013*"quality" + 0.012*"great" + 0.011*"life" + 0.011*"best" + 0.011*"display" + 0.010*"oneplus"'),
 (1,
  '0.066*"good" + 0.024*"camera" + 0.022*"phone" + 0.020*"product" + 0.018*"nice" + 0.018*"mobile" + 0.017*"quality" + 0.011*"price" + 0.006*"awesome" + 0.006*"amazon"')]
+ Code  + Text
```

```
# Let's try 3 topics
ldana = models.LdaModel(corpus=corpusna, num_topics=3, id2word=id2wordna, passes=10)
ldana.print_topics()

[(0,
  '0.038*"good" + 0.034*"camera" + 0.030*"phone" + 0.026*"battery" + 0.015*"quality" + 0.013*"life" + 0.011*"display" + 0.010*"screen" + 0.009*"great" + 0.008*"performance"'),
 (1,
  '0.045*"product" + 0.033*"good" + 0.027*"mobile" + 0.019*"amazon" + 0.017*"phone" + 0.014*"nice" + 0.013*"delivery" + 0.010*"value" + 0.008*"iphone" + 0.008*"time"'),
 (2,
  '0.077*"phone" + 0.034*"good" + 0.028*"price" + 0.026*"camera" + 0.025*"best" + 0.019*"quality" + 0.013*"range" + 0.011*"nice" + 0.010*"oneplus" + 0.010*"great"')]
```

```
# Let's try 4 topics
ldana = models.LdaModel(corpus=corpusna, num_topics=4, id2word=id2wordna, passes=10)
ldana.print_topics()

[(0,
  '0.025*"camera" + 0.021*"phone" + 0.014*"good" + 0.012*"oneplus" + 0.012*"display" + 0.011*"screen" + 0.011*"battery" + 0.009*"device" + 0.009*"great" + 0.008*"day"'),
 (1,
  '0.044*"mobile" + 0.038*"value" + 0.025*"good" + 0.024*"money" + 0.019*"product" + 0.013*"mone" + 0.009*"nice" + 0.007*"amazon" + 0.006*"exchange" + 0.005*"gud"'),
 (2,
  '0.069*"good" + 0.062*"phone" + 0.040*"camera" + 0.033*"battery" + 0.028*"quality" + 0.017*"price" + 0.016*"life" + 0.011*"performance" + 0.011*"nice" + 0.011*"best"'),
 (3,
  '0.044*"phone" + 0.026*"product" + 0.022*"amazon" + 0.018*"delivery" + 0.016*"best" + 0.013*"iphone" + 0.009*"apple" + 0.008*"time" + 0.008*"great" + 0.007*"worth"')]
```

In Table 13, | see similar groups in different orders. By considering all the steps of topic analysis with the LDA method, it can be concluded that good reviews (Good Reviews is 1) can be categorized into three main topics: (1) beverages, (2) pet items, and (3) cookies and snacks.

For this study, the same steps of topic modeling also run for bad reviews (Good reviews is Q). In the end, out of 9 topics, nouns only, 2 topics model made the most sense. The prevailing topics are (1) pet items, and (2) beverages. As a result of the topic modeling,
it can be seen that reviewers are complaining and praising for almost the same products because prevailing topics are the same for both good reviews and bad reviews.

5. Prediction Model Creation

This section used machine learning models for data analysis. The data set is labeled data. Therefore, predictions will be done by using supervised machine learning models which refers to fitting a model of dependent variables to the independent variables, accurately predict the dependent variable for future observations, or understand the relationship between the variables. Concerning the data set, the below-listed methods can be appropriate for creating prediction models:

1. Gaussian Naive Bayes
2. Multinomial Naive Bayes
3. Bernoulli Naive Bayes
4. Complement Naive Bayes
5. Logistic Regression

We will apply four different methods in the Naive Bayes Model. Evaluating these different types of Naive Bayes models are suggested if time permits. Therefore, evaluate different Naive Bayes models can help us to find a better prediction model. As for performance evaluation metrics, accuracy, precision, and recall rate will be used. Receiving Operating Characteristic (ROC) Curve will be drawn and models' performance will be compared based on the Area Under Curve (AUC) metric.

Interpretation of Metrics

Before starting to create models, here is a brief definition for model evaluation metrics according to the Scikit-Learn website:

Accuracy: It is a classification score. In multilabel classification, the accuracy function computes subset accuracy: the set of labels predicted for a sample must exactly match the corresponding set of labels in y_true.

Precision: Intuitively the ability of the classifier not to label as positive a sample that is negative. Precision is the estimated probability that a randomly selected retrieved document is relevant.

Recall: Intuitively the ability of the classifier to find all the positive samples. The recall is the estimated probability that a randomly selected relevant document is retrieved in a search.

F1 Score: The F1 score can be interpreted as a weighted average of precision and recall, where an F1 score reaches its best value at 1 and the worst score at 0. It is an easier way of evaluating recall and precision at the same time.

The precision-recall curve: This shows the tradeoff between precision and recall for different thresholds. A high area under the curve represents both high recall and high precision, where high precision relates to a low false-positive rate, and high recall relates to a low false-negative rate. High scores for both show that the classifier is returning accurate results (high precision), as well as returning a majority of all positive results (high recall).

5.1. Naive Bayes with Gaussian Method

This model assumes that the likelihood of the features is to be a normal distribution. For our dataset, this assumption does not hold. On the other hand, checking the Gaussian Naive Bayes model does not hurt our study. Figure 31 presents the model performance of the model. As we can see, the accuracy of Gaussian Naive Bayes is very low for our unbalanced data.

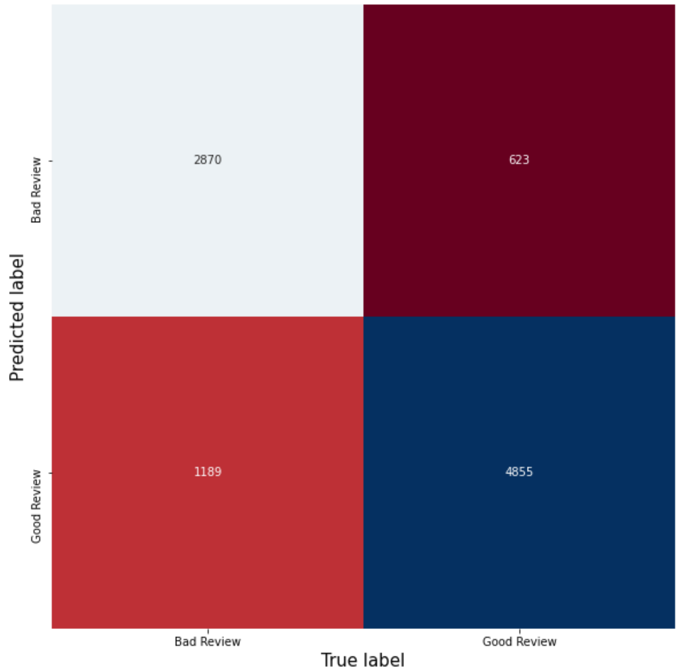


Figure 31: Confusion matrix for Gaussian NB model

```
Accuracy score: 0.81
Precision score: 0.89
Recall score: 0.80
F1 score: 0.81
```

5.2. Multinomial Naive Bayes

Multinomial Naive Bayes implements the naive Bayes algorithm for multinomially distributed data and is one of the two classic naive Bayes variants used in text classification where the data are typically represented as word vector counts, although tt-idf vectors are also known to work well in practice.

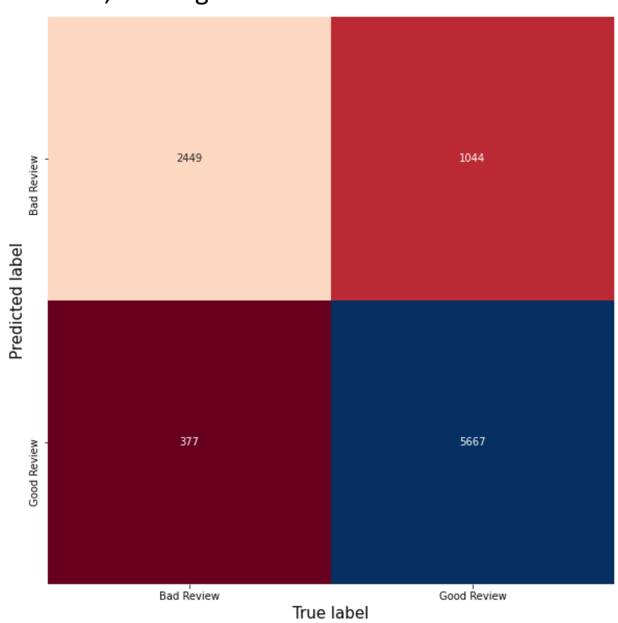


Figure 32: Confusion matrix for Multinomial NB model

```
Accuracy score: 0.85
Precision score: 0.84
Recall score: 0.94
F1 score: 0.85
```

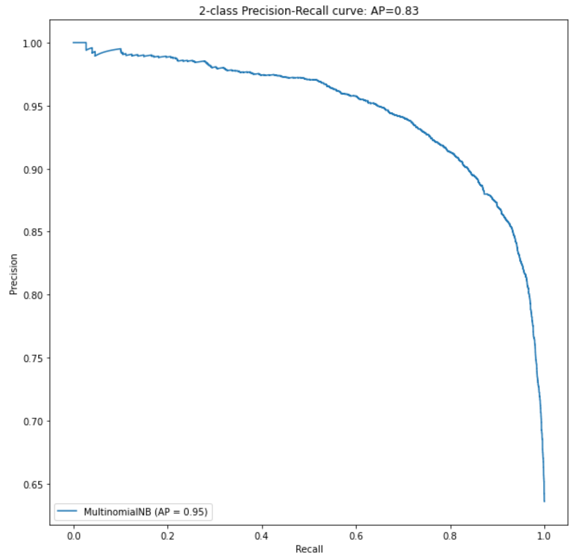


Figure 33 Precision-recall curve for Multinomial NB

5.3. Bernoulli Naive Bayes

Bernoulli Naive Bayes implements the naive Bayes training and classification algorithms tor data that is distributed according to multivariate Bernoulli distributions; i-e., there may be multiple features but each one is assumed to be a binary-valued (Bernoulli, boolean) variable. Therefore, this class requires samples to be represented as binary-valued feature vectors. In this aspect, we can expect that Bernoulli Naive Bayes model can show a good performance for our dataset. Figure 34 supports our expectations to a certain degree. We have good precision and recall rates but a low f1 rate.

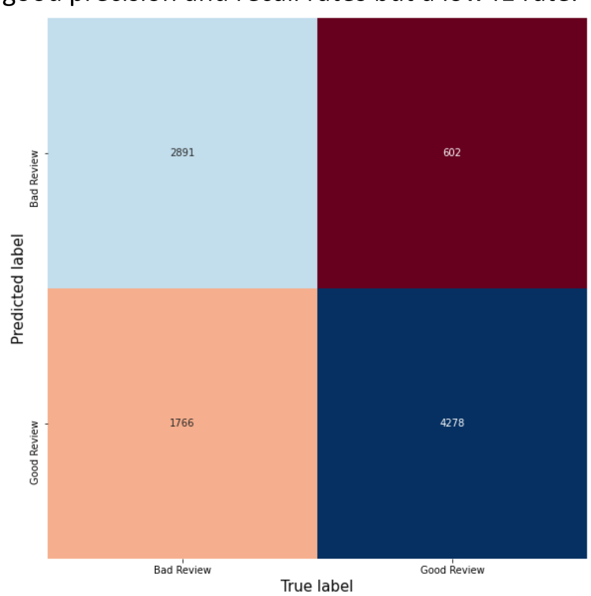


Figure 34 Confusion matrix for Bernoulli NB model

```
Accuracy score: 0.75
Precision score: 0.88
Recall score: 0.71
F1 score: 0.75
```

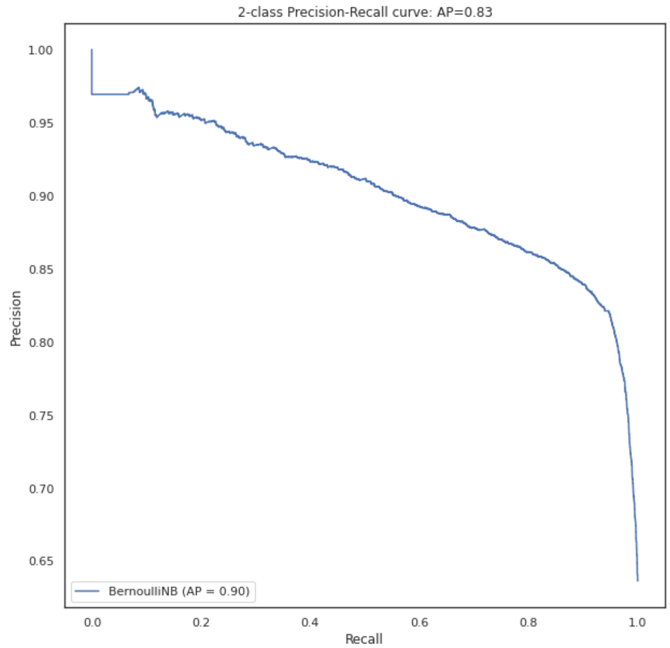


Figure 35 Precision-recall curve for Bernoulli NB

5.4. Complement Naive Bayes

The Complement Naive Bayes classifier was designed to correct the "severe assumptions" made by the standard Multinomial Naive Bayes classifier. It is particularly suited for imbalanced data sets

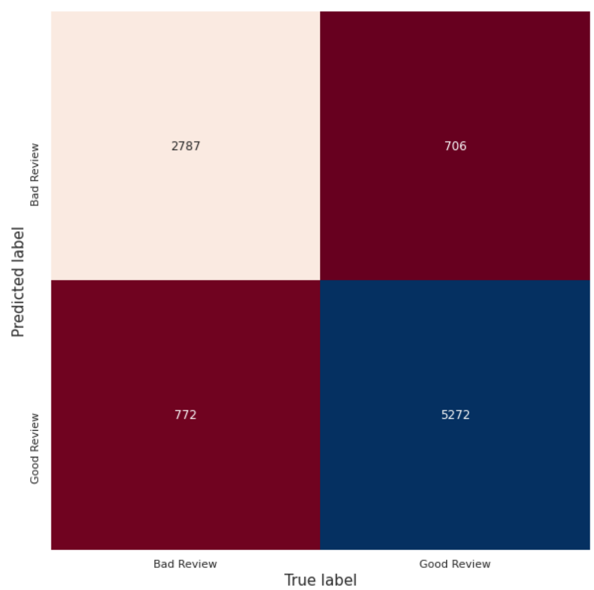


Figure 36 Confusion matrix for complement NB model

```
Accuracy score: 0.85
Precision score: 0.88
```

```
Recall score: 0.87
F1 score: 0.85
```

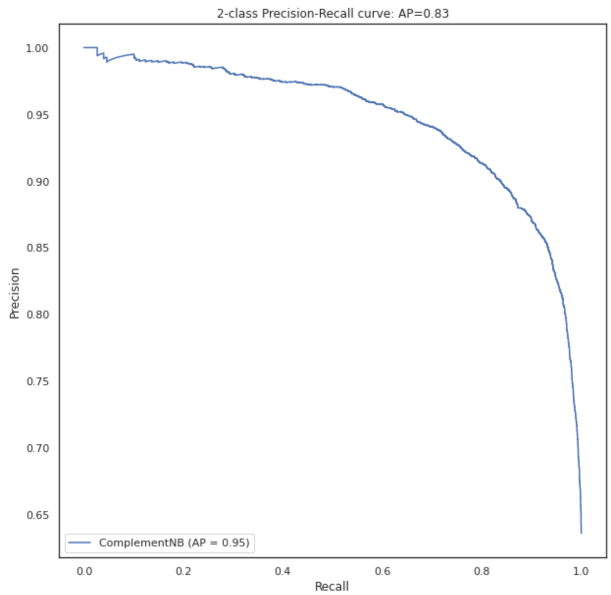


Figure 37 Precision-recall curve for complement NB

5.5. Logistic Regression Model Fitting

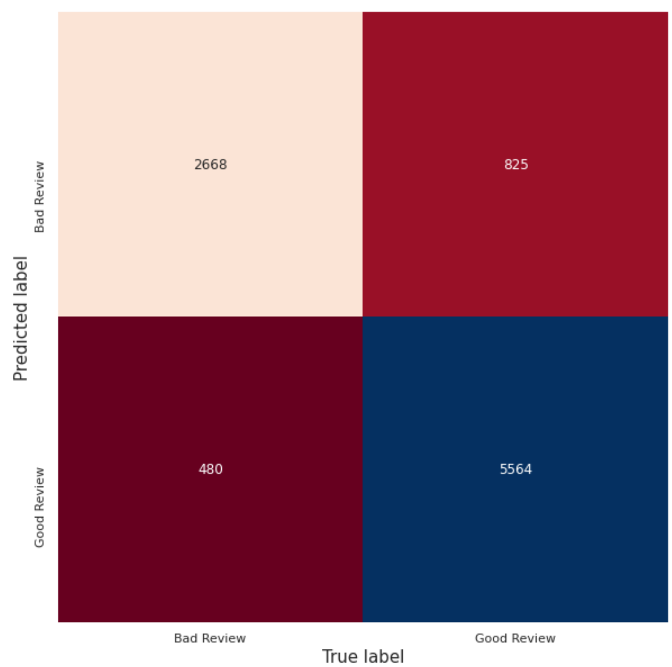


Figure 38 Confusion matrix for the logistic regression model

```
Accuracy score: 0.86
Precision score: 0.87
Recall score: 0.92
F1 score: 0.86
```

5.6. Model performance evaluation

When we examine Table 6, the Logistic Regression Model shows a good classification performance overall. Performance of Naive Bayes models also close to logistic regression. But model performance comparison will be done later. At this point, the Receiver Operating Characteristic (ROC) curve will be investigated. Final model selection will be done based on the ROC curve.

Table 6 Comparison of Model Performance

Measure (%) / Model	Gaussian NB	Multinomial NB	Bernoulli NB	Complement NB	Logistic Reg
Accuracy	0.81	0.85	0.75	0.85	0.86
Precision	0.89	0.84	0.88	0.88	0.87
Recall	0.8	0.94	0.71	0.87	0.92
f1	0.81	0.85	0.75	0.85	0.86

Receiver Operating Characteristic (ROC) Curve

This metric evaluates classifier output quality. ROC curves typically feature a true positive rate on the Y-axis and a false positive rate on the X-axis. This means that the top left corner of the plot is the "ideal" point - a false positive rate of zero, and a true positive rate of one. This is not very realistic, but it does mean that a larger area under the curve (AUC) is usually better.

Table 7 Area under the ROC Curve (AUC)

Evaluation	Score (%)
Gaussian NB	86
Multinomial NB	92
Bernoulli NB	87
Complement NB	92
Logistic Reg	93

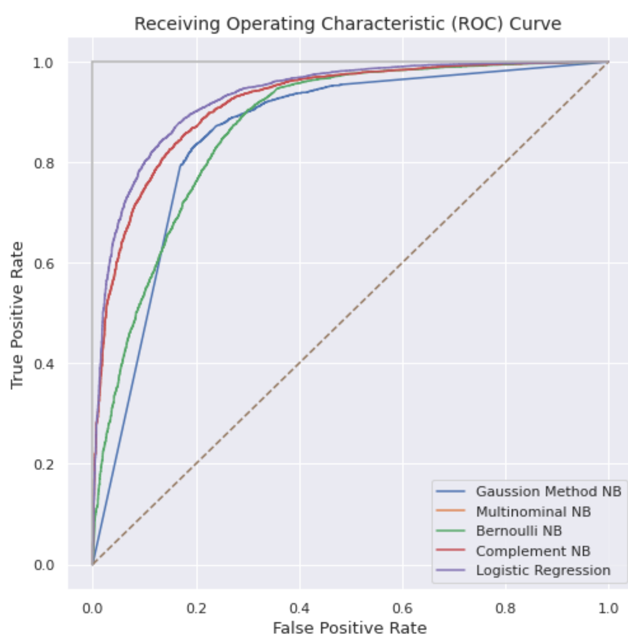


Figure 39 ROC curve for all models used in classification

Now, the interpretation of the ROC curve helps us to determine which model is the best. In Figure 39, the “steepness” of ROC curves is important, since it is ideal to maximize the true positive rate while minimizing the false positive rate. In this figure, it can be observed that the highest true positive rate and the lowest false positive rate are performed by a logistic regression model. This can also be seen by comparing AUC scores. The logistic regression model has the highest AUC score: 93 (Table 7).

Multinomial and Complement Naive Bayes models’ performance are the same. Their Area Under Curve (AUC) scores is 92 (Table 7). That’s why they overlap on the plot (Fig. 39). We also see the color of the Complement NB because of overlapping.

5.7. Sanity check

Running sanity checks is an important part of the data analysis. The final analysis is only as accurate as your data, which means it’s worth it to spend some time to validate the data’s accuracy and completeness. For sanity check, | will take random samples from data and check their predicted and actual Good Reviews values. While making a sanity check, the best performing model, logistic regression, was used for the prediction. In other words, predicted values were gathered from a logistic regression model.

```
check_df.loc[(check_df.actual_label ==1) & (check_df.prediction==0)].review.sample(50).to_list()
```

```
['battery much better',
 'waste products ear speaker damaged working replacement opction',
 'got hands product august beginning finger print senson bad camera good battery life optimization bad one morning
 'feels like low quality displaybut good',
 'battery working properlyand sometimes automatically changes silent mode',
 'box charger like original samsung',
 'delivery ordered product august supposed deliver august item shipped delivered september customer care service p
 'oneplus fan still disappointed green magenta tint display real whenever something grey screen lot use dark mode
 'vry bad camera',
 'used mobile weeks writing reviewthe mobile battery good long lasting normal use hours day reduces battery fully
 'speed working dont need wait pressing button',
 'phone working properly sound apps like skype mxplayer come properlyonly hear one earphoneother one us nit workin
 'phone goodbut buy mistake u lost thing like cable adapter etcu cant find whereall time u see stock',
 'one problem found phone automatically call recording option available phone happ',
 'bad phone good face problems',
 'second phone oneplus dont regret buying',
 'using phone week describe life ok ok like day face lock mp camera zoom optionover phone good',
 'using since last month last week speaker wasnt working properly stopped workingin lockdown situation cant go ser
 'finger scanner working properly defective product good product please replace give another device oppo',
 'using several daysi noticed image processing taking timemore clicking image need keey hand stable otherwise imag
 'redmi note ordered december received whoever wish budget phone go need think twiceso u get via flash sale made f
 'first time buying mobile online platform much impressed amazon service prompt delivery mobile ok value paid thou
 'fingerprint sensor unlock ok connectivity browsing fast quality camera quality poor condition rear camera could
 'purchased phone student poor family help online class',
 'facing heating issue iphone bought recent sale updating device getting hot lot heavy use got solved factory rese
 'defective product received amazon replaced raising issue wish amazon continue future',
 'disappointed camera quality though quad arraythe zoom almost always pixelated especially video recordingof combi
 'display good oneplus bought think good price phone battery life disappointing getting battery life oneplus yrs u
 'offering technician visit charger working corona sitution expecting return product one charger service desk put
 'good product samsungbut think samsung going way cheap china mobiles selling data maybe tackle competitionfirst t
 'months usage find finger print scanner side phone absolutely waste mm width scanner read finger print quickly u
 'honestly speaking great device camera one main highlight device wrap charge extremely fast charge phone hour one
 'bad product dont bu',
 'opinion weeks usageproscamerabattery lifedisplaygood processor compared samsung phones priceconsheavyweight with
 'mobile working guddisplay quality superbbbbattery charging fastusing mobile one week nowstill problemmall workin
 'got phone daysbattery goodcamera goodos touch smoothno problem hanging issueonly issue charging time due charger
 'paid models ear phones amount sending pl asis possible send mobile phone quality check give backup thanq',
```

Figure 40 Example of reviews that are mistakenly predicted as negative

```
check_df.loc[(check_df.actual_label ==0) & (check_df.prediction==1)].review.sample(50).to_list()
```

```
['bought mobile time launch rs thereafter days big billion sale price mobile reduced rs discount hdfc card upto guess price mobil
 'battery quality superb rate long lastingcamera quality good thoughtaverage cameradisplay perfect useable gamersaged people useo
 'camera quality good power button hard',
 'good product site',
 'phone great one issue fingerprint sensor working properl',
 'usage weekpros great finger print sensormindblowing face unlock even complete darkness worksuperb displaygood rear camera comes
 'camera good compare phones range priceits badand battery drain normal use hrsif use gamingyoutube means drain like hrs onlyfing
 'taking battery consuption high due default samsung apps cannot deleted reducing possible delete even download app samsung store
 'product overall good get issue inserting sim using sim tool ejector try many times battery life good discharges rapidly overall
 'bought product great indian festival sale october really doubt product original even year battery life non existent camera refu
 'phone little bit hanging since first month buying didnt recommend one thing liked advantage use wifi hotspot time ie share data
 'nothings worth price fake',
 'phone looks good heavy lagging lot switch apps charging capacity good',
 'phone good display battery life camera etc found issue far buy unfortunate competitive indian brand market left chinese phones'
 'first thing claim otg cable used reality otg doesnt work device doesnt recognise cablesecond thing three months works fine star
 'phone look good price range temper cover come thisi bought oct delivered days sim slots working',
 'charger given phone slow q charging takes provide fast charger phone',
 'smartphone best value money budget phone watch youtube videos details compair',
 'shouldnt gone ahead warp discharging users dont like itfine display look user experience',
 'happy phone ram processing fast lot accessibility issues find brands battery falls like flowing water runs quick charges back w
 'good phoneit good display camera etcbut unboxing wasnt greatthe screen protector dusty many indentations display fine phone pri
 'dont know costlier phone doesnt finger print face id secure recognise face even close eyes thats big drawback like one plus unl
 'best phone camera good struggle lot low light front camera comes mirror mode u cant turn shutter speed slow takes second click
 'nice phone price range spend little bit get betterregarding phine battary upto mark still hardly cover full day normal use wher
 'battery gon',
 'phone goodbut camera quality phone goodone plus pro camera quality better thisphotos clear phones camera',
 'avg battery performancedrains fast even didnt find major difference dont worry battery definitely buy performance wise beastcam
 'इस फन क आग क कमर थड बहत अचछ ह लकन फट फटत ह जम करन क बद इसलए हम आपस अनरध करग क कमर क कवलट बड ह सबस बड बत ह पह
 'redmi prominent smartphone brand given quality product pocketfriendly price range lot expectation xiaomi used almost smartphone
 'buyed phone sale day received next day liked phone battery backup camera features rest things pretty common like competitors ph
 'snapdragon price range totally looting customers processor camera maybe bad atleast snapdragon series mus',
 'camera place',
 'happens sometimes oneplus gb varian',
 'sound sucks big time day played songs sound quality awesome next day gone dogs sound problem big dnt bu',
 'phone use secondaryfor aged personscreen size little bit small regular use',
 'upto found better phone',
 'bought oneplus recently overall performance fine accidentally found issue shake mobile could hear sound loose part inside near
 'phone worth price brands price better features using months found slowed speed intermittent hanging phone camera really poor',
 'using phone days find extremely slow thought speed one selling points nord using gaming see visible lag scrolling moving one sc
 'good hype iphone nothing great itmy realms pro equally good new features good camera',
 'good phone outset gb ram exaggerating works like ram experienced nagging downloads hanging times minimum storage load device pr
 'value money produc',
```

Figure 41 Examples of Reviews that are Mistakenly Predicted as Positive

In Table 40, we see that buyers sometimes give bad ratings even though they are satisfied with the purchase. This can cause errors in prediction in text analysis. In Table 41, we also observe that even though buyers write a negative review, they are sometimes forgiving and give good ratings. In some cases, they are changing their rating if they get a refund, but the review stays still negative. Another common reason for misclassification of the model is that the reviews can contain both positive and negative sentiments. Text analysis algorithms can have a seriously hard time dealing with this kind of complicated review.

In English, some words can be used for both good and bad things such as ‘awful’, ‘crazy’ etc. We observed that the classification model could not understand the contextual meaning of this kind of word and made the wrong prediction. Lastly, the sarcastic language was another reason for the misclassification. In sarcasm, buyers meant the opposite meaning of the words. However, the NLP tools that were used in this study are vulnerable to the sarcastic usage of words.

Limitations of the Study

During data cleaning, all the emojis and emoticons were cleaned. However, emojis and emoticons may provide a substantial understanding of the text analysis. Linguistic and socio-linguistic interpretation of emojis may be useful in sentiment analysis. In this aspect, there is room for improvement at this point in

the analysis. A study that analyzes emojis and emoticons may reach a better prediction model than this one.
Another suggestion for further studies is the usage of stemming and lemmatization. For grammatical reasons, documents are going to use different forms of a word, such as organize, organizes, and organizing. Additionally, there are families of derivationally related words with similar meanings, such as democracy, democratic, and democratization. But, of course, stemming and lemmatization have a big drawback: they change the text itself. This may hurt some studies seriously. For this study, we didn't prefer to apply these processes. However, for the further step, stemming and lemmatization can be applied, and results can be compared.
For dealing with misspelled words, we used TextBlob.correct() method during data cleaning. However, there were still misspelled words in the data. There can be room for improvement for spelling correction. With better spelling correction tools, the analysis can give better results.
Lastly, we believe that it is possible to get good classification by using features other than Text in the dataset such as the number of words, subjectivity, polarity. We did my predictions only by using text data and focused on NLP methods. However, random forest, lasso, ridge, xgboost models may also give a good classification by using other features available in the dataset.
Finally, the data is taken from 100 bestselling smartphone reviews, which are supposed to have more positive ratings than negative. In future review analysis projects from Amazon, we recommend considering taking reviews from products from the general area, not from the top seller list.