Deccan Education Society's

FERGUSSON COLLEGE (AUTONOMOUS), PUNE-4

Department of Data Science

A

Project Report

On

Amazon Reviews Using Sentiment Analysis

By

1. Lobhana Kirme 11105

2. Neha Bavdhankar 11107

3. Tushar Bhor 11122

[2020-2021]

Deccan Education Society's
Fergusson College (Autonomous), Pune
Department Of Computer Science

CERTIFICATE

This is to certify that the project entitled

_____ submitted by

1. _____
2. _____
3. _____

in partial fulfillment of the requirement of the completion of M.Sc. (Data Science)-II [Semester-III], has been carried out by them under our guidance satisfactorily during the academic year 2020-2021.

Place: Pune

Date: / /2021

Head of Department
Department Of Computer Science
Fergusson College, Pune

Project Guide:

1. _____

Examiners Name **Sign**

1. _____ _____

2. _____ _____

INDEX

PROJECT DESCRIPTION

The world we see nowadays is becoming more digitalized. In this digitalized world e-commerce is taking the ascendancy by making products available within the reach of customers where the customer doesn't have to go out of their house. As now a day's people are relying on online products so the importance of a review is going higher. For selecting a product, a customer needs to go through thousands of reviews to understand a product. But in this prospering day of machine learning, going through thousands of reviews would be much easier if a model is used to polarize those reviews and learn from it. We used supervised learning method on a large-scale amazon dataset to polarize it and get satisfactory accuracy.

The amazon review dataset for Mobile Phones were considered. The reviews and ratings given by the user to different products as well as reviews about user's experience with the product(s) were also considered.

Product reviews are becoming more important with the evolution of traditional brick and mortar retail stores to online shopping. Consumers are posting reviews directly on product pages in real time. With the vast amount of consumer reviews, this creates an opportunity to see how the market reacts to a specific product. We will be attempting to see if we can predict the sentiment of a product review using machine learning tools.

The dataset we will be using is scrapped from Amazon website using web scrapping. We have Implemented Web Scraping in Python with BeautifulSoup.

PROBLEM STATEMENT

The goal is to develop a model to predict user's sentiment (i.e., Positive, Negative or Neutral) from their comments, usefulness of review and recommendation of most similar items to users based on collaborative filtering.

Which product categories has lower reviews / maybe inferior products? (i.e., Mobile Phones) Which product have higher reviews / maybe superior products?

Which products should be kept, dropped from Amazon's product roster (which ones are junk?) Also: can we associate positive and negative words/sentiments for each product in Amazon's Catalogue By using Sentiment analysis, can we predict scores for reviews based on certain words.

Following are the area's which we have worked on,

- We have compared the ratings percentage of different mobile brands using Visualization.
- We have also analysed that which colour of mobile brand was mostly loved by the customers using their reviews.
- We have also visualized the comments of consumers using text classification in WorldCloud.
  It shows major insight in terms of seller's perspective.

Various visualizations have been used for all the above scenarios.

RESEARCH PAPERS REFERRED

https://www.researchgate.net/publication/325756171_Sentiment_analysis_on_large_scale_Amazon_product_reviews

https://ieeexplore.ieee.org/document/8376299

DATA SOURCE

We have not used the dataset directly from any website.

We have scraped the data from Amazon Website using Web Scraping.

Steps involved in web scraping:

1) Send an HTTP request to the URL of the webpage you want to access. The server responds to the request by returning the HTML content of the webpage. For this task, we will use a third-party HTTP library for python-requests.
2) Once we have accessed the HTML content, we are left with the task of parsing the data. Since most of the HTML data is nested, we cannot extract data simply through string processing. One needs a parser which can create a nested/tree structure of the HTML data. There are many HTML parser libraries available but the most advanced one is html5lib.
3) Now, all we need to do is navigating and searching the parse tree that we created, i.e., tree traversal. For this task, we will be using another third-party python library, Beautiful Soup. It is a Python library for pulling data out of HTML and XML file.

DATA DESCRIPTION

This is a list of over 30K consumer reviews for Amazon products Mobile Phones.

The attributes are as follows:

Attributes	Description
Name	Name of customer.
Date	Date of purchase.
Star	Star given to product by customer (i.e., 5, 4, 3, 2, 1)
Country	Country of customer.
Comment_Title	Short comment given by the customer.
Colour	Colour of Mobile.
Rom	Internal Storage(Memory) in GB
Size	RAM in GB.
Verified	The customer had actually bought the product or not.
Helpful_People_Count	Count of agreed customer on that particular comment.
Brand	Model of mobile.
Price	Price of mobile in Rs.

TECHNOLOGIES USED

- Python: Used for various charts and graphic visualizations as well as predictions.

  Libraries Used:

  Pandas : Working with data files

  Numpy : For Scientific Calculation

  Matplotlib and Seaborn : Data Visualization

  Plotly : Advance Visualization

  Math : For Complex Mathematical Functions

  Worldcloud: Used for Presenting Text data

  Countvectorizer : Used to convert a collection of text documents to a vector

- Beautiful Soup :For Web Scrapping

DATA PREPROCESSING

Data Preprocessing is the process of making data suitable for use while training a machine learning model. The dataset initially provided for training might not be in a ready-to-use state, for e.g. it might not be formatted properly, or may contain missing or null values.

While scrapping we preprocessed the data as follows,

1.Data was in raw form so we converted it into text and extract the data which was required.

2.By removing null spaces

3.Splits the data in the way we can use for further requirment.

We import libraries like numpy, pandas, Matplotlib, etc. for further use. To load the data, we use the traditional ‘read_csv’ method mentioning the URL of our dataset as the file path.

Data Scraping:

Import the required packages for scrapping the data from the website

We can get the html page using urllib, and use BeautifulSoup to parse the html page, and it looks like that we have to generate file to be read from BeautifulSoup.

```
[ ]  from urllib.request import urlopen as ureq
     from bs4 import BeautifulSoup as soup
     import csv
     import pandas as pd
     import re
```

```
[ ]  i=2
     my_url="https://www.amazon.in/Redmi-Note-Pro-Interstellar-Snapdragon/product-reviews/B08696XB3V/ref=c
     uclient=ureq(my_url)
     page_html=uclient.read()
     uclient.close()
     page_soup=soup(page_html,"html.parser")
```

```
[ ]  #d=page_soup.find_all("div")
     #print(d)
     containers=page_soup.find_all("div",{"class":"a-section review aok-relative"})
     len(containers)
```

```
10
```

Here we can see that there are 10 observations in 1st container so we need pagination to scrap more data from this website

In next picture we can see the raw form of data so we converted it into text and extract the data which was required

```
container
```

```
<div class="a-section review aok-relative" data-hook="review" id="RQE15S6E97FHD"><div class="a-row a-spacing-none" id="RQE15S6E97FHD-review-card">
<span>lovely phone and beautiful design</span>
</a></div><span class="a-size-base a-color-secondary review-date" data-hook="review-date">Reviewed in India on 23 September 2020</span><div class=
<span>
  just got the phone delivered today, amazing quick service by Amazon, i am using the phone since morning and did not find any lagging issue and c
</span>
</span></div><div class="a-row review-comments comments-for-RQE15S6E97FHD"><div aria-live="polite" class="a-row a-expander-container a-expander-ir
<div class="a-row a-spacing-small"><span class="a-size-base a-color-tertiary cr-vote-text" data-hook="helpful-vote-statement">9 people found this
<span class="a-button a-button-base"><span class="a-button-inner"><a class="a-button-text" data-hook="vote-helpful-button" href="https://www.amazc
        Helpful</div>
</a></span></span></div>
</span><span class="cr-footer-line-height">
<span><i aria-label="|" class="a-icon a-icon-text-separator" role="img"></i><span class="a-declarative" data-action="cr-popup" data-cr-popup='{"wi
<div aria-expanded="false" class="a-expander-content a-spacing-top-base a-spacing-large a-expander-inline-content a-expander-inner" style="displa
<div class="a-row a-spacing-none a-grid-vertical-align a-grid-center"><div class="a-column a-span6"><span class="a-size-base a-viewoptions-list-la
</div><div class="a-section a-spacing-extra-large a-spacing-top-medium a-text-center comment-load-error aok-hidden"><div aria-live="assertive" cla
```

From above raw data we extracted the data required for analysis

For this we used pyhton function like find_all, split, strip, join, etc. to convert the data in useable form.

Here are some examples of it

```
[ ]  #Rating
     Rating = container.find_all("span",{"class":"a-icon-alt"})
     Star = Rating[0].text.split(" ")[0]
     print(Star)

     5.0

[ ]  #Auther
     Auther = container.find_all("span",{"class":"a-profile-name"})
     Name = Auther[0].text
     print(Name)

     Saif

[ ]  #Country
     country_container = container.find_all("span",{"class":"a-size-base a-color-secondary review-date"})
     text = country_container[0].text.strip()
     Country = "".join(text.split(" ")[2])
     Date = " ".join(text.split(" ")[-3:])
     print(Country)
     print(Date)

     India
     23 September 2020
```



```
[ ]  #Comment_Title
     comment_title_container = container.find_all("a",{"class":"a-size-base a-link-normal review-title a-color-base review-title-content a-
     comment_title = comment_title_container[0].find("span").text
     print(comment_title)

     lovely phone and beautiful design

[ ]  #main_comment
     comment_container = container.find_all("span",{"class":"a-size-base review-text review-text-content"})
     comment = comment_container[0].find("span").text.strip()
     print(comment)

     just got the phone delivered today, amazing quick service by Amazon, i am using the phone since morning and did not find any lagging is
```

We extract data like name of the customer, rating given by the customer to certain product, country of customer, comment given to product, product type and its details.

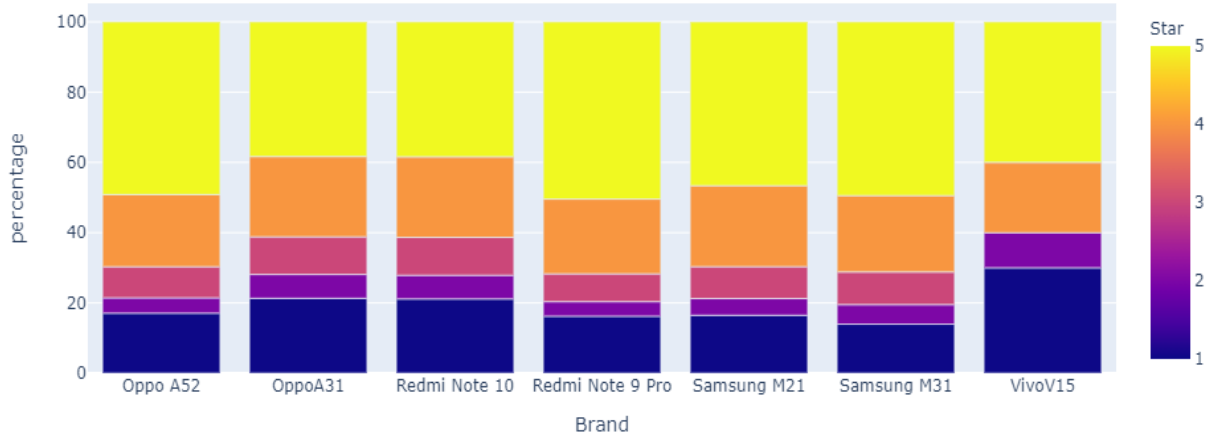
After this we store the data in certain variable and created data frame of this data, and exported this data frame as csv.

VISUAL EXPLORATION

Here we have compared the ratings percentage of different mobile brands using Visualization.

Star Level	General Meaning
★	I hate it.
★★	I don't like it.
★★★	It's okay.
★★★★	I like it.
★★★★★	I love it.

We have calculated the percentage of stars (i.e. for 1-5) for each mobile brand and then we visualize it.



Comment :

From above plot we observed that the mobile Redmi Note 9 Pro and Samsung M31 has highest 5 Star ratings with percentage 50.44 and 49.43 respectively.

And the lowest 5 Star ratings was observed for Oppo A31 and Redmi note 10 with percentage 38.33 and 38.41 respectively.

Also the rating 1 Star is highest for Vivo V15 and lowest for Samsung M31.

Hence we can conclude that the most liked mobile phone by customers is Redmi Note 9 Pro.

Positive Sentiment Analysis of Redmi Note 9 Pro Using World Cloud

Comment

The word cloud from good rating reviews for the above product. It shows major insight in terms of customers perspective. It indicates most of the positive customers agree with “Best Camera”, ”value for money”, ”battery life”.

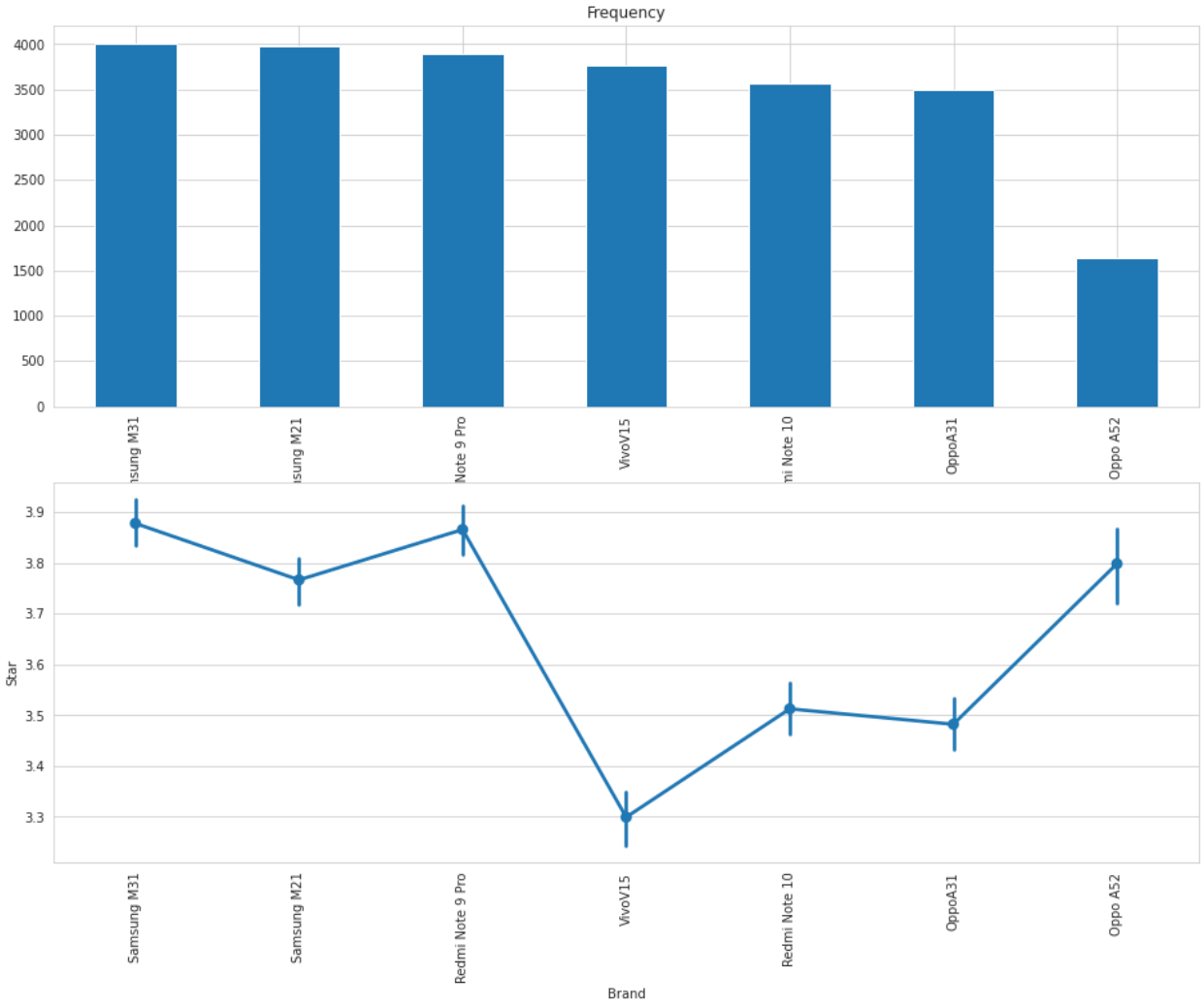
Negative Sentiment Analysis of Vivo V15 Using World Cloud

Comment

The word cloud from good rating reviews for the above product. It shows major insight in terms of customers perspective. It indicates most of the customers agree with “Poor Camera” (specially Front Camera) , “Settings Problem” and “Face Unlock Problem”.

Comparative Study

i)Here we have plotted a combined plot for number of observations with respect to their average star rating for each mobile brand.



Comment:

Here we observed that highest average star rating is for Samsung M31 and Redmi Note 9 Pro approximately 3.9

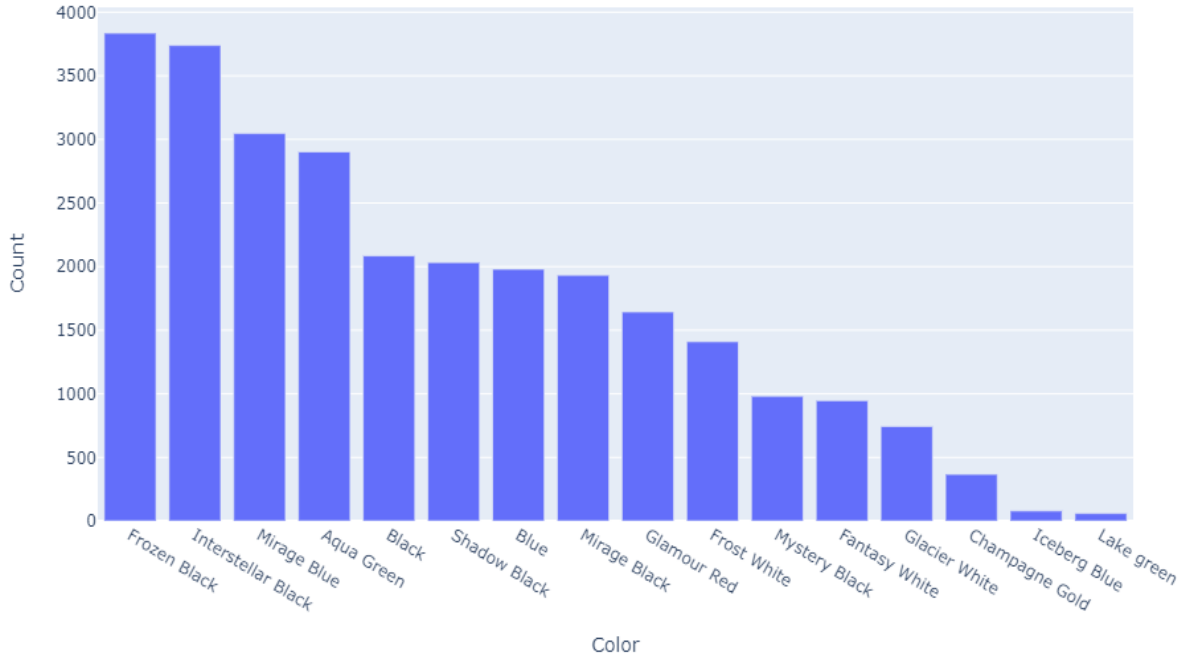
And lowest average star rating is for Vivo V15 which is 3.3.

ii) Customer Rating and Price of Mobile

Comment:

From the analysis of above plot, we can see that products with lower reviews are not significant enough to predict these lower rated products are inferior. On the other hand, products that are highly rated are considered superior products, which also performs well and should continue to sell at a high level.

Popular Colours of Mobiles



Comment:

- Based on the bar graph for Mobile Colour, we see that Black color (Frozen Black, Interstellar Black , Mirage Black) have significantly more count than other colour, which may indicate a higher sale in those specific colors.

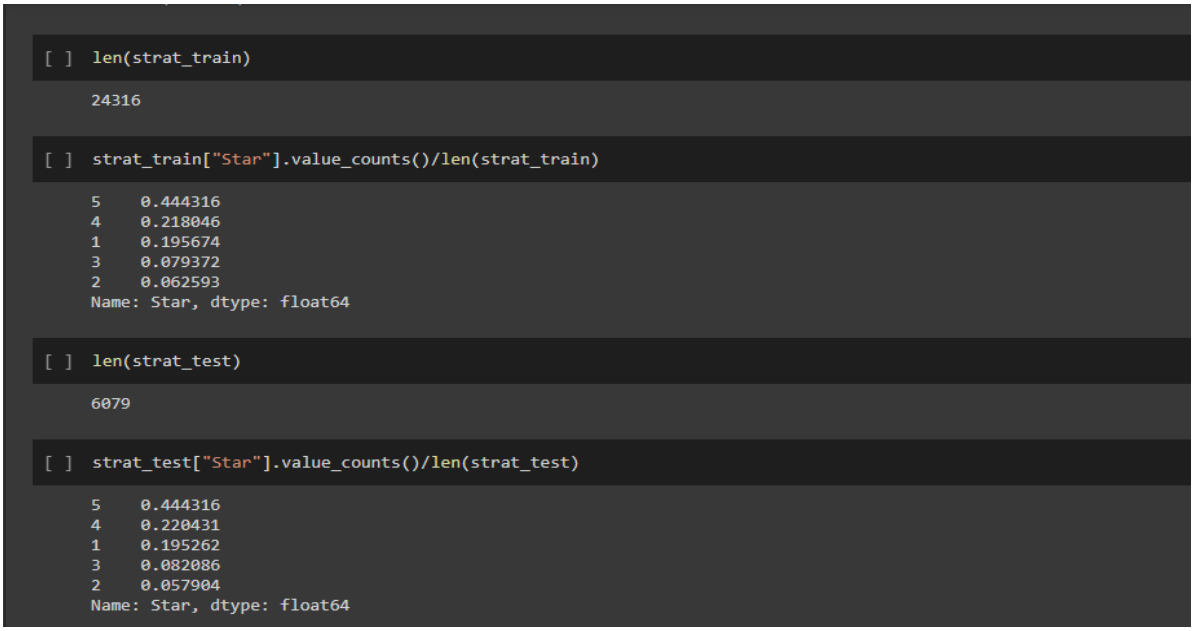
.

MODEL BUILDING:

Split into Train/Test Data

- Before we explore the dataset we're going to split it into training set and test sets
- Our goal is to eventually train a sentiment analysis classifier.
- Since the majority of reviews are positive (5 stars), we will need to do a stratified split on the reviews score to ensure that we don't train the classifier on imbalanced data.
- To use sklearn's Stratified Shuffle Split class, we're going to remove all samples that have NAN in review score, then covert all review scores to integer data type.

Check to see if train/test sets were stratified proportionately in comparison to raw data.

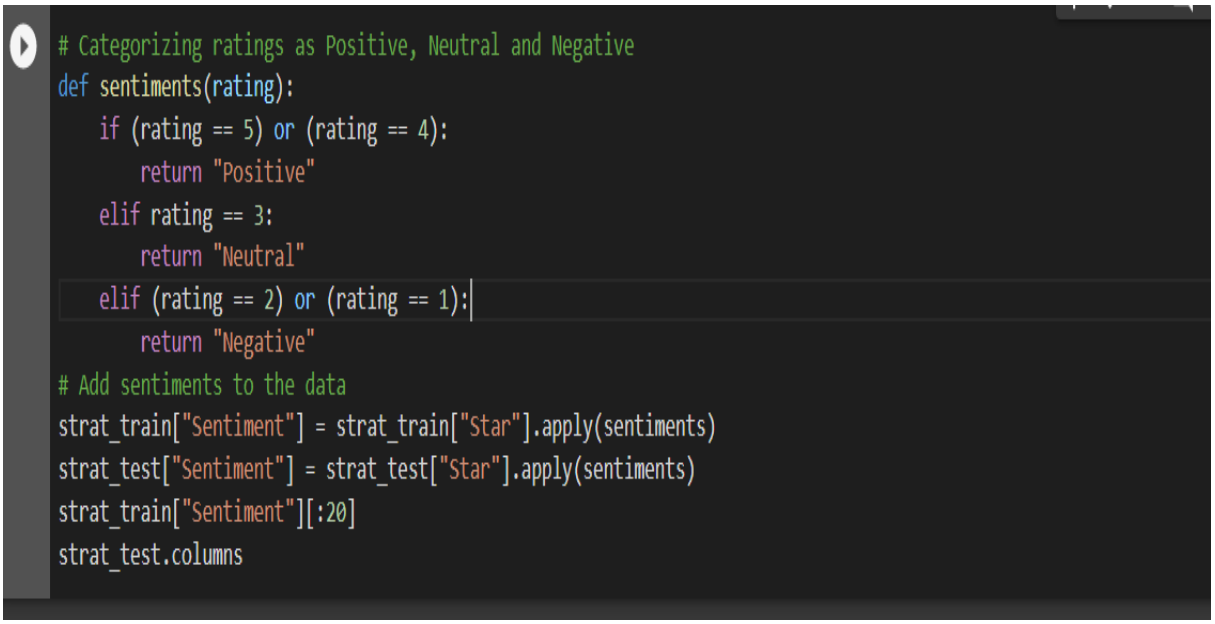


```
[ ] len(strat_train)

    24316

[ ] strat_train["Star"].value_counts()/len(strat_train)

    5    0.444316
    4    0.218046
    1    0.195674
    3    0.079372
    2    0.062593
    Name: Star, dtype: float64

[ ] len(strat_test)

    6079

[ ] strat_test["Star"].value_counts()/len(strat_test)

    5    0.444316
    4    0.220431
    1    0.195262
    3    0.082086
    2    0.057904
    Name: Star, dtype: float64
```

Sentiment Analysis

Using the features in place, we will build a classifier that can determine a review's sentiment.

Set Target Variable (Sentiments) Segregate ratings from 5-4 into positive, 3-neutral, and 1-2 into negative.



```
# Categorizing ratings as Positive, Neutral and Negative
def sentiments(rating):
    if (rating == 5) or (rating == 4):
        return "Positive"
    elif rating == 3:
        return "Neutral"
    elif (rating == 2) or (rating == 1):
        return "Negative"
# Add sentiments to the data
strat_train["Sentiment"] = strat_train["Star"].apply(sentiments)
strat_test["Sentiment"] = strat_test["Star"].apply(sentiments)
strat_train["Sentiment"][:20]
strat_test.columns
```

Extract Features

- Here we will turn content into numerical feature vectors using the Bag of Words strategy:
- Assign fixed integer id to each word occurrence (integer indices to word occurrence dictionary).
  X [i,j] where i is the integer indices, j is the word occurrence, and X is an array of words (our training set).

In order to implement the Bag of Words strategy, we will use SciKit-Learn's CountVectorizer to performs the following:

- Text preprocessing:
  i) Tokenization (breaking sentences into words)
  ii) Stopwords (filtering "the", "are", etc)

- Occurrence counting (builds a dictionary of features from integer indices with word occurrences)
- Feature Vector (converts the dictionary of text documents into a feature vector)

```
[ ]  # Replace "nan" with space
     X_train = X_train.fillna(' ')
     X_test = X_test.fillna(' ')
     X_train_targetSentiment = X_train_targetSentiment.fillna(' ')
     X_test_targetSentiment = X_test_targetSentiment.fillna(' ')
```

```
     # Text preprocessing and occurance counting
     from sklearn.feature_extraction.text import CountVectorizer
     count_vect = CountVectorizer()
     X_train_counts = count_vect.fit_transform(X_train)
     X_train_counts.shape
```

```
     (24316, 22467)
```

Here we have 24316 training samples and 22467 distinct words in our training sample.
Also, with longer documents, we typically see higher average count values on words that carry very little meaning, this will overshadow shorter documents that have lower average counts with same frequencies, as a result, we will use **TfidfTransformer** to reduce this redundancy.

- Term Frequencies (**Tf**) divides number of occurrences for each word by total number of words.
- Term Frequencies times Inverse Document Frequency (**Tfidf**) downscales the weights of each word (assigns less value to unimportant stop words ie. "the", "are", etc)

Model Building Using Various Machine Learning Algorithms:

The model building process involves setting up ways of collecting data, understanding and paying attention to what is important in the data to answer the questions you are asking, finding a statistical, mathematical or a simulation model to gain understanding and make predictions.

- Multinominal Naive Bayes

- Multinominal Naive Bayes is most suitable for word counts where data are typically represented as word vector counts (number of times outcome number X[i,j] is observed over the n trials), while also ignoring non-occurrences of a feature i
- Naive Bayes is a simplified version of Bayes Theorem, where all features are assumed conditioned independent to each other (the classifiers), P(x|y) where x is the feature and y is the classifier

- Test Model

```
[246] import numpy as np
      predictedMultiNB = clf_multiNB_pipe.predict(X_test)
      np.mean(predictedMultiNB == X_test_targetSentiment)

      0.8138662316476346
```

Here we see that our Multinominal Naive Bayes Classifier has 81.38% accuracy level based on the features.

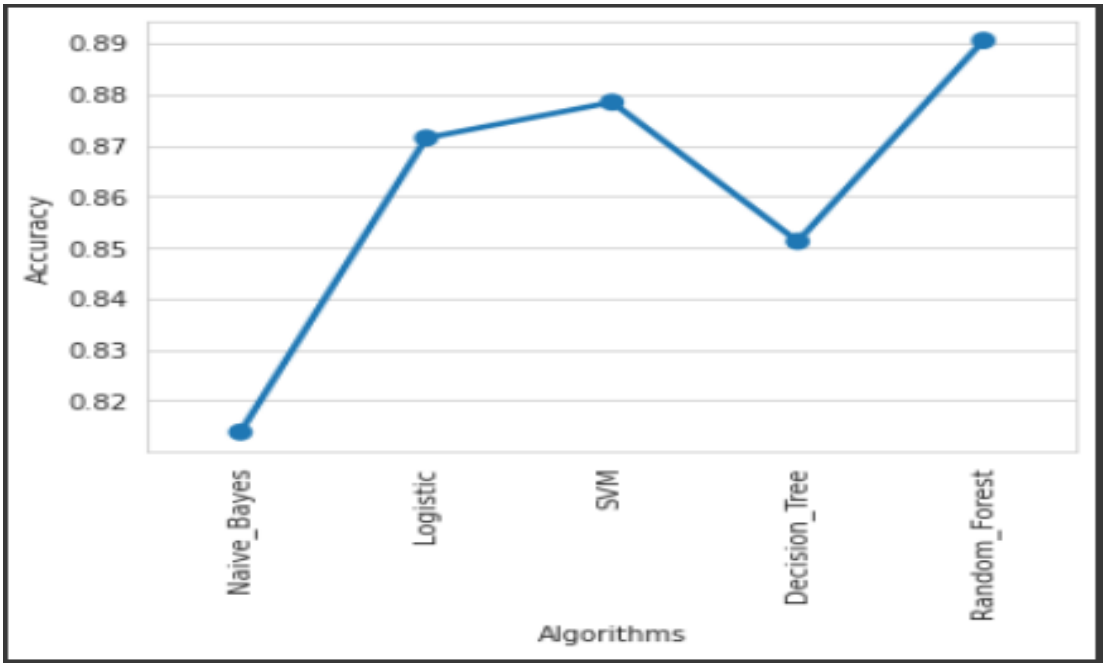
```
[233] List=['Poor backup',
            'Best Mobile under 14k INR !',
            'Average product']
      List

      ['Poor backup', 'Best Mobile under 14k INR !', 'Average product']

[239] clf_multiNB_pipe.predict(List)

      array(['Positive', 'Positive', 'Positive'], dtype='<U8')
```

Here, manually we have taken one of the example of each positive, negative and neutral comment and checked its sentiment. We observed that the sentiment of 'Poor backup' is negative but it predicted as positive i.e. it predicted wrongly and also the sentiment of 'Average Product' comment is neutral but it predicted wrongly as positive. So, we cannot use this model for further predictions.

Similarly , we performed for other models.

Testing Other Models:

Algorithm	Accuracy in Percentage
Multinominal Naive Bayes	81.38
Logistic Regression	87.14
Support Vector Machine Classifier	87.84
Decision Tree Classifier	85.12
Random Forest Classifier	89.05



From above table and graph of accuracy of model versus respective algorithm we observed that the algorithm Random Forest has highest accuracy around 89%.

Similarly, we have taken one example for each kind of sentiment and predicted sentiment for remaining algorithms.

```
[240] ##Multinominal Naive Bayes
      clf_multiNB_pipe.predict(List)

      array(['Positive', 'Positive', 'Positive'], dtype='<U8')

[241] ##Logistic Regression
      clf_logReg_pipe.predict(List)

      array(['Negative', 'Positive', 'Positive'], dtype=object)

[243] ##Support Vector Machine Classifier
      clf_linearSVC_pipe.predict(List)

      array(['Negative', 'Positive', 'Positive'], dtype=object)

[244] ##Decision Tree Classifier
      clf_decisionTree_pipe.predict(List)

      array(['Negative', 'Positive', 'Neutral'], dtype=object)

[245] ##Random Forest Classifier
      clf_randomForest_pipe.predict(List)

      array(['Negative', 'Positive', 'Neutral'], dtype=object)
```

From above image we observed that for algorithm Decision Tree Classifier and Random Forest Classifier the sentiments were predicted correctly, But, Random Forest has more accuracy as compared to Decision Tree so we can use Random Forest Classifier for further sentiment analysis.

Analysis

- From the analysis above in the classification report, we can see that products with lower reviews are not significant enough to predict these lower rated products are inferior. On the other hand, products that are highly rated are considered superior products, which also performs well and should continue to sell at a high level.

- As a result, we need to input more data in order to consider the significance of lower rated product, in order to determine which products should be dropped from Amazon's product roster.

- The good news is that despite the skewed dataset, we were still able to build a robust Sentiment Analysis machine learning system to determine if the reviews are positive or negative. This is possible as the machine learning system was able to learn from all the positive, neutral and negative reviews, and fine tune the algorithm in order to avoid bias sentiments.

- In conclusion, although we need more data to balance out the lower rated products to consider their significance, however we were still able to successfully associate positive, neutral and negative sentiments for each product in Amazon's Catalog.

Observations

- From analysis we can say that Redmi Note 9 Pro and Samsung M31 has highest 5 star rating so we can suggest this phone to customers. And Amazon can continue these products in their Catalog.

- Redmi Note 10 and Oppo A31 has lowest 5 star percentage as well as highest 1 star rating so customer should avoid this phones.

- From World Cloud Map we observed that why customers avoid that particular phone and why they liked that particular phone using their comments on that product.

- Using Comparative study

  i) Between Average star rating with respect to product we selected Samsung M31 is best mobile on the basis of rating.

  ii) Between colors of mobile , Black color (Frozen Black, Interstellar Black , Mirage Black) have significantly more count than other color, which may indicate a higher sale in those specific colors.

  iii) Between average star rating and price , price have no effect on rating of that product.

- For Sentiment Analysis we tried various types of Machine Learning Classification Algorithms. From all we selected Random Forest Classifier as best on the basis of its accuracy.

- Manually we have taken one of the example of each type of comment to check the sentiment. And we got the perfect result for Random Forest Classifier so we used this model for further deployment.

Future Enhancement and Conclusion

Future Enhancemnet

The project has a very vast scope in future. The project can be implemented on intranet in future. Project can be updated in near future as and when requirement for the same arises, as it is very flexible in terms of expansion.

The result from all experiments implies that both approaches give higher accuracies when they are being applied on the summaries of the reviews. The possible explanation for this result might be the nature of the reviews. The reviews itself contain a large amount of words, which can lead to sparsity in bag of words features. As a result we see that the accuracies of the algorithms for all experiments are higher when applied on the summaries which are more informative and contain limited number of words.

There is also another challenge that should be addressed in the problem of sentiment classification which is identification of negation and its effect on the semantic understanding of sentences. Future studies could fruitfully explore this issue further by developing approaches to tackle this issue.

Conclusion

- From the analysis above in the classification report, we can see that products with lower reviews are not significant enough to predict these lower rated products are inferior. On the other hand, products that are highly rated are considered superior products, which also perform well and should continue to sell at a high level.
- As a result, we need to input more data in order to consider the significance of lower rated product, in order to determine which products should be dropped from Amazon's product roster.
- The good news is that despite the skewed dataset, we were still able to build a robust Sentiment Analysis machine learning system to determine if the reviews are positive or negative. This is possible as the machine learning system was able to learn from all the positive, neutral and negative reviews, and fine tune the algorithm in order to avoid bias sentiments.
- In conclusion, although we need more data to balance out the lower rated products to consider their significance, however we were still able to successfully associate positive, neutral and negative sentiments for each product in Amazon's Catalog.

BIBLIOGRAPHY

https://www.digitalocean.com/community/tutorials/how-to-scrape-web-pages-with-beautiful-soup-and-python-3

https://www.datacamp.com/community/tutorials/web-scraping-using-python

https://www.geeksforgeeks.org/generating-word-cloud-python/

https://plotly.com/python/bar-charts/

https://developer.ibm.com/technologies/data-science/tutorials/learn-classification-algorithms-using-python-and-scikit-learn/