NATURAL LANGUAGE PROCESSING AND RECOMMENDER SYSTEMS

Group 1 project

Morris Zuniga

Ankit Mehra

Prashant Sharma

Reggie Brice

CONTENTS

Introduction 2
ABOUT the dataset 2
Data Exploration 3
Data Pre-PROCESSING 8
Models ..... **Error! Bookmark not defined.**
testing summary 10
Phase 2 12
Future work 15
Final Conclusion 16

INTRODUCTION

In this project we will be constructing a sentiment analysis model for products based on customers textual reviews of office products. We will be using the data sets from Amazon product review. We will be releasing the project in two phases, the first focused on data preprocessing and lexicon-based analysis, and the second based on machine learning approach.

ABOUT THE DATASET

The dataset was a collection of amazon reviews of office products, specifically the reduced K-core set. The reduced dataset contains approximately 53258 records and has 9 different columns that define each review. All the features were crucial to analyzing the data, and therefore none were dropped; two columns ("ReviewText" and "summary") will be concatenated for more manageable processing. The number of reviews per product ranged from 5 to 311.

The column "overall" represents the overall rating of the product. A new column "label" was created, and each review was assigned "Positive" label for a rating of 4 or more, "Neutral" for the rating of 3, and "Negative" for rating of 2 or less.

DATA EXPLORATION

Below are samples of how we have chosen to explore and understand the data for the purpose of preprocessing.

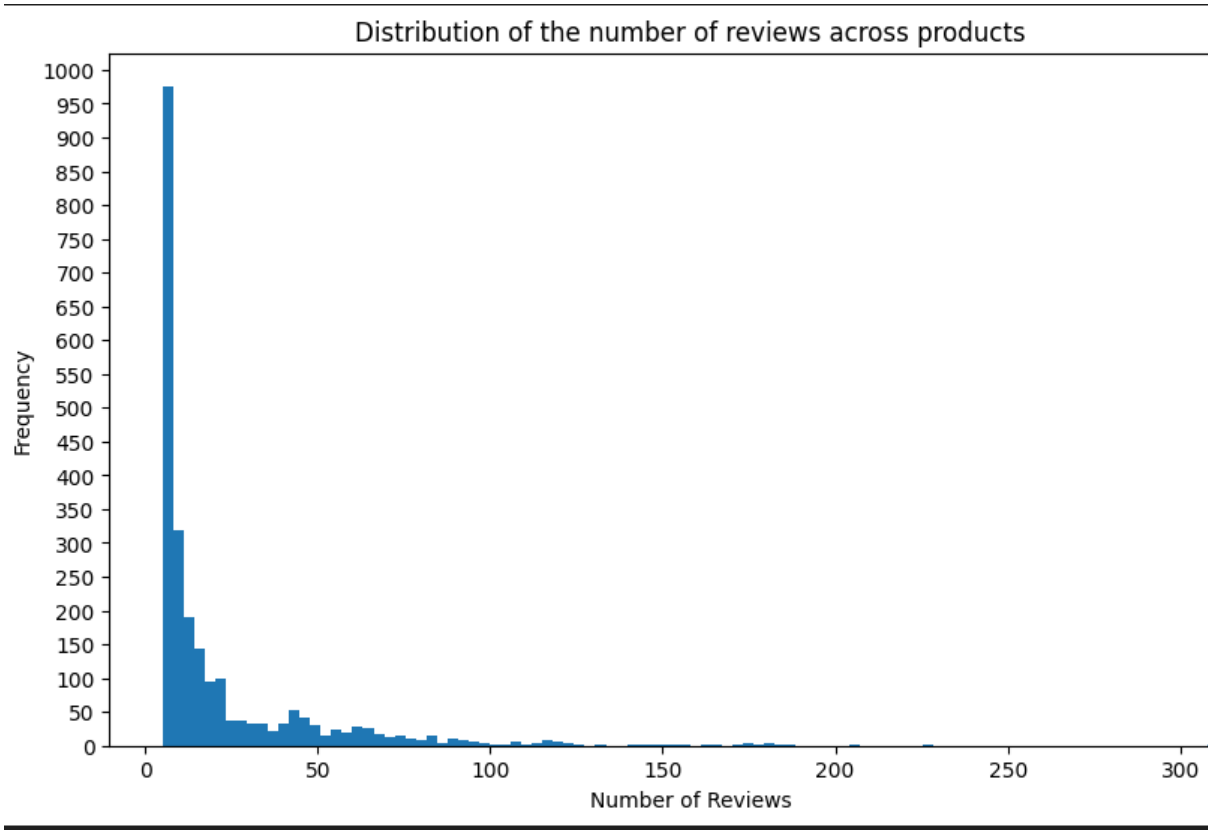


Figure 1 Distribution of number of reviews across products

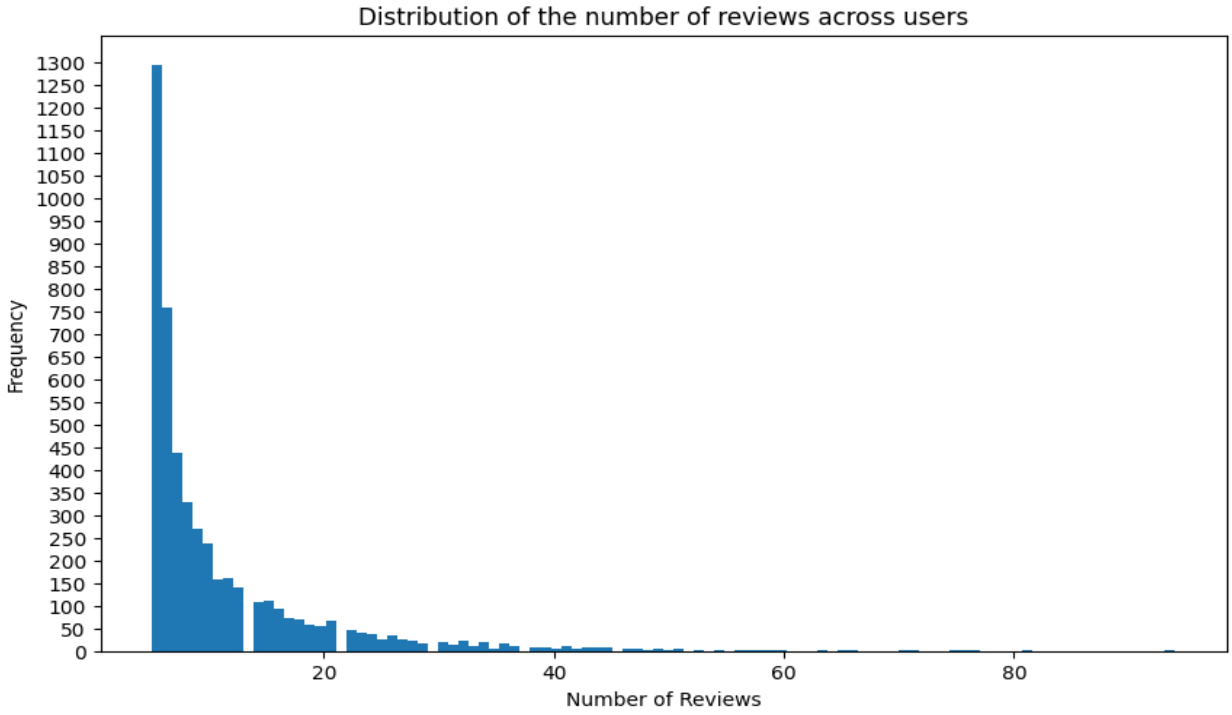


Figure 2 Number of reviews across users

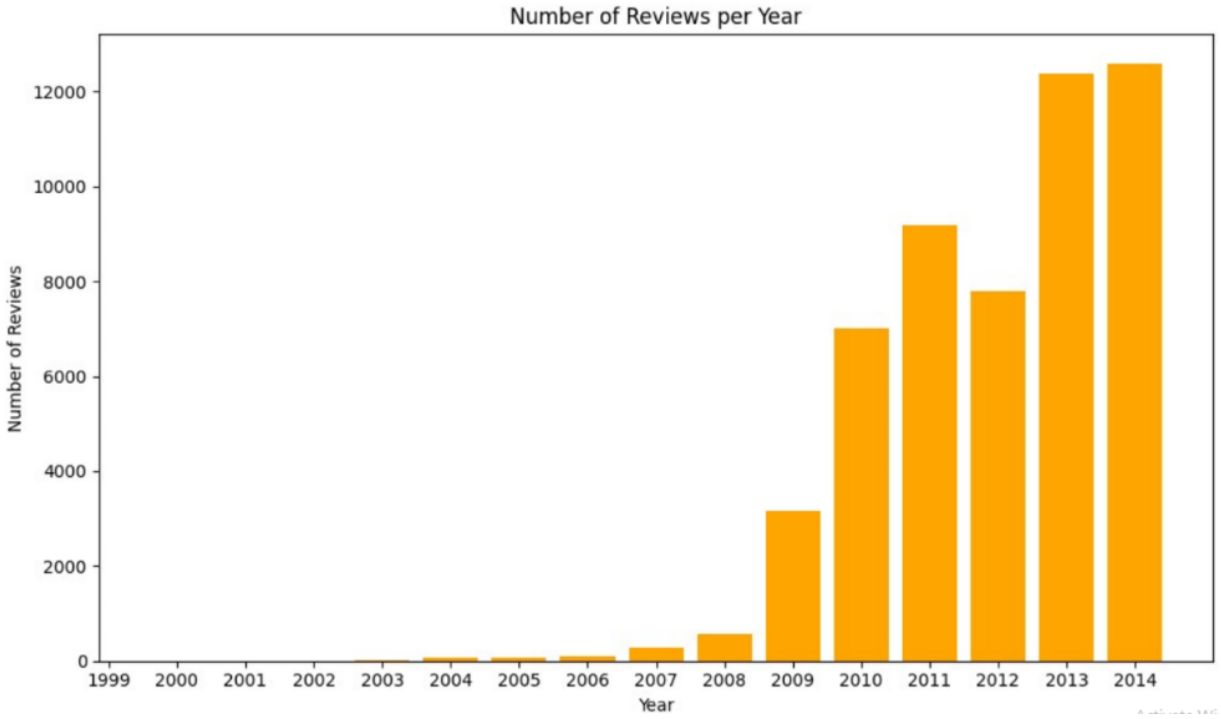


Figure 3 Number of reviews per year

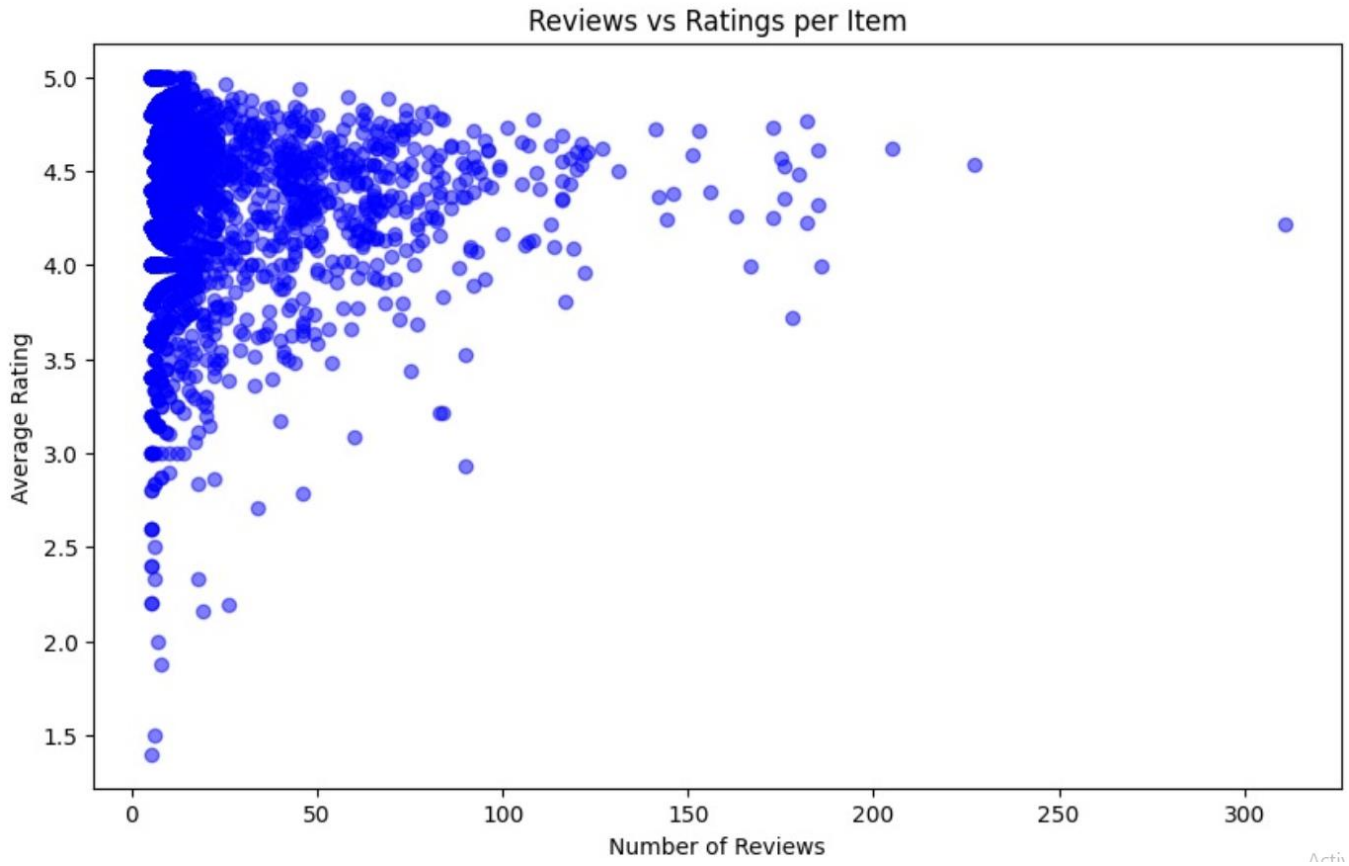


Figure 4 Reviews Vs Rating per Item

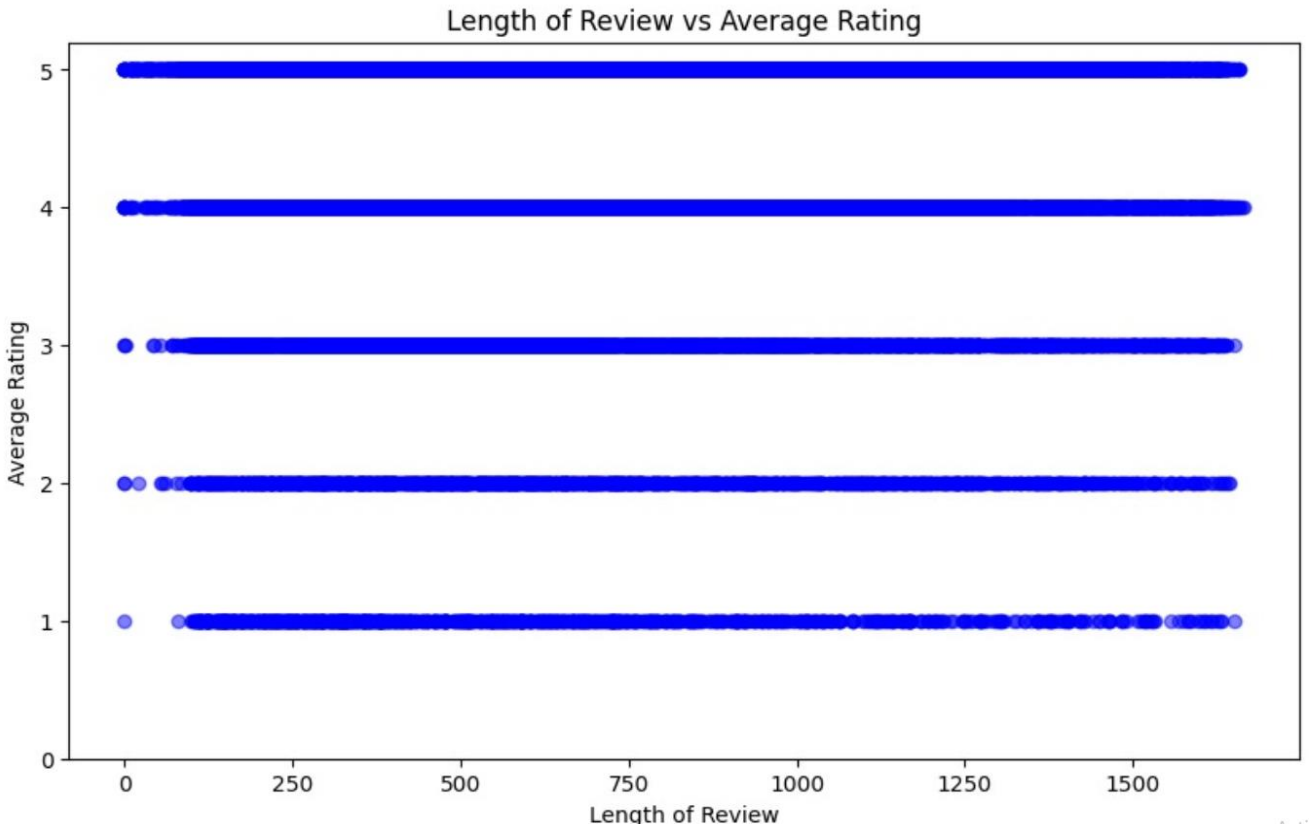


Figure 5 Length of review vs rating

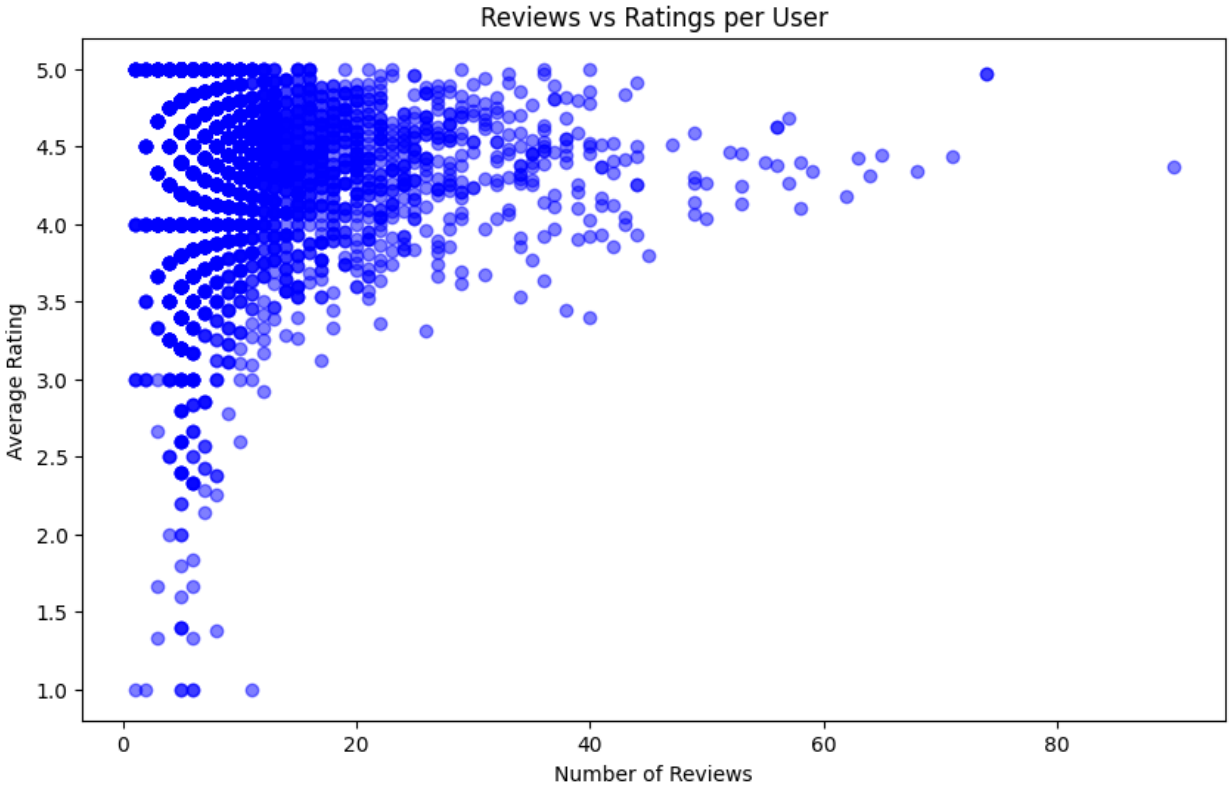


Figure 6 Reviews vs Rating per user

DATA PRE-PROCESSING

After randomly selecting 1000 values from original dataset following steps were taken to clean the data -

1. Removing outliers(removed in the original dataset itself)

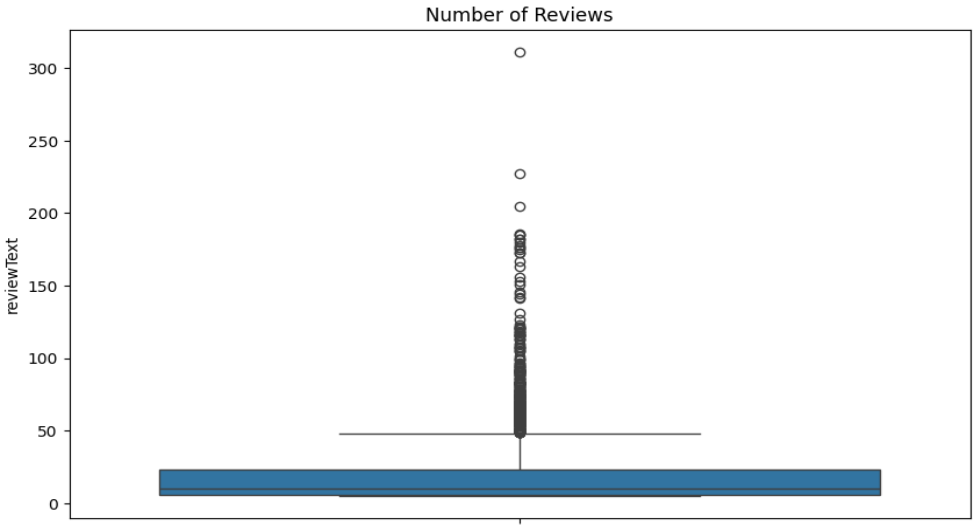


Figure 1 Data with outliers

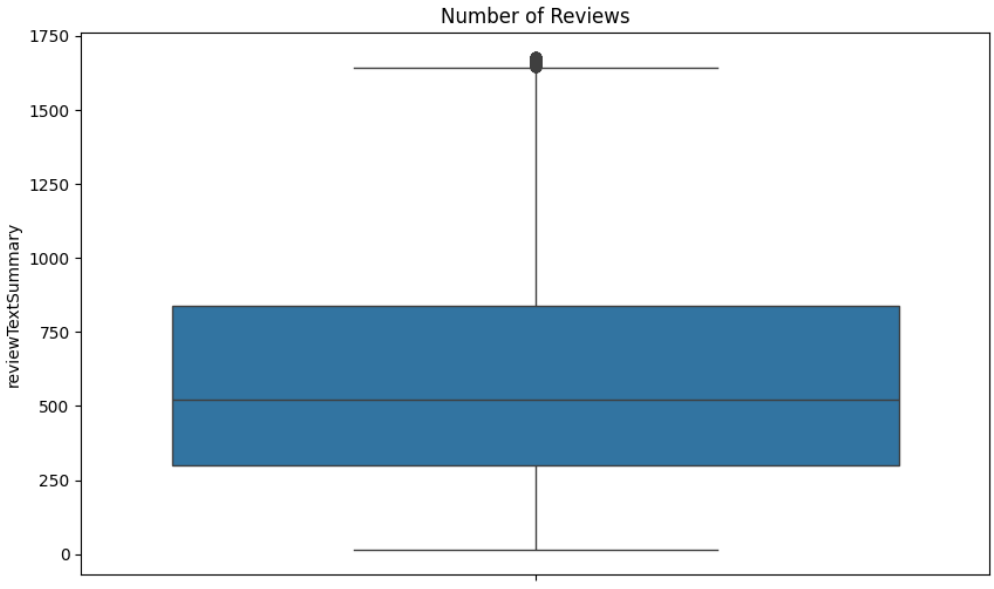


Figure 2 Data after removing outliers

2. Check if there are any rows with no- English words using “langdetect” library

```
# check for non-english reviews
from langdetect import detect
lang_list = df_select["reviewText"].apply(detect)
lang_list.value_counts()
```

```
reviewText
en    1000
Name: count, dtype: int64
```

   as you can see all 1000 rows are in English so no need to remove any rows
3. Removed all special characters from the “reviewTextSummay” column.
4. Converting the text in “reviewTextSummay” in lowercase.
5. Removed all the stop words except the negative words which might have some sentimental connotations.
6. Removed all the punctations.

MODELS: PHASE 1

For the first, lexicon-based, analysis phase, two models were chosen: **TextBlob and VADR (Valence Aware Dictionary and Sentiment Reasoner).**

TextBlob is a Python library that provides a simple and easy-to-use interface for performing various natural language processing (NLP) tasks. It is built on top of the popular NLTK library and offers additional features such as spelling correction, translation, and language detection. When a sentence is passed into TextBlob for sentiment analysis, it provides two outputs: polarity and subjectivity. Polarity is a value between -1 and 1, where -1 represents negative sentiment and 1 represents positive sentiment. Subjectivity is a value between 0 and 1, representing the degree of opinion or judgment in the text.

VADER (Valence Aware Dictionary and sentiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically designed to analyze sentiments expressed in social media. It is therefore more adept at handling emojis, slangs, and social media diction. VADER relies on a sentiment lexicon, which is a list of lexical features (e.g., words) labeled with their semantic

orientation (positive or negative). It also uses a set of grammatical and syntactical rules to handle complex sentiment expressions. 1

VADER is better suited for tasks were understanding the context, including the influence of things like capitalization and punctuation, is important, while TextBlob is a good choice for simpler tasks where context and nuance are less critical. Both tools have their strengths and weaknesses, and their effectiveness can vary depending on the specific characteristic of the dataset. 2

TESTING SUMMARY

The reviewtext, a concatenation, in each case, of the original review and its summary, was cleansed of special characters, capitalization, punctuation, and stop words (except for the negation words such as “no” or “not”).

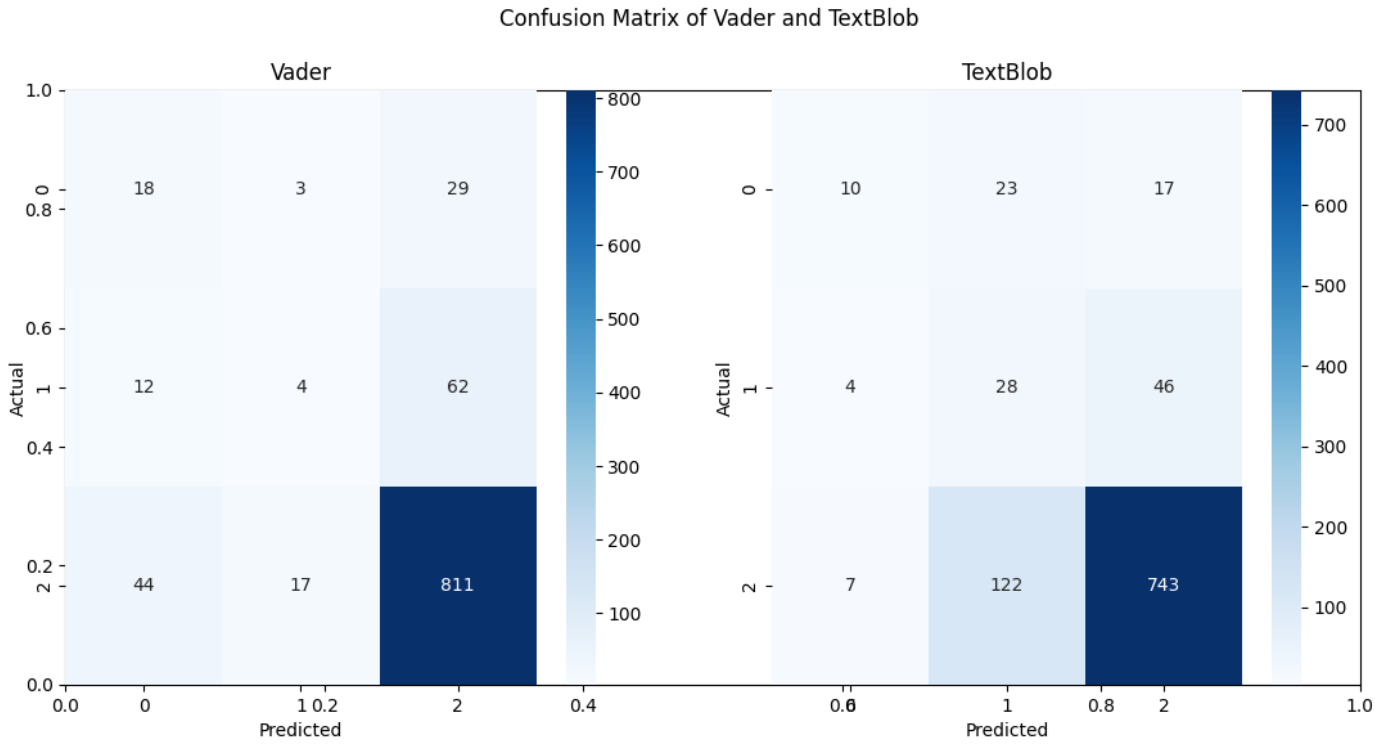
For TextBlob, a label of “Positive” was assigned for polarity more than 0.1, “Negative” for polarity less than -0.1, and “Neutral otherwise. The same labels were assigned in the case of VADER for “compound” scores of -0.1, 0.1, or otherwise.

Comparing the predicted VADER and TextBlob labels against the true labels, we obtained the following accuracy scores and confusion matrix:

Metric	TextBlob	Vader
Accuracy	0.781000	0.833000
Precision	0.840275	0.809189
Recall	0.781000	0.833000
F1-score	0.803711	0.817919

We see that both models perform reasonably well, although VADER tends to outperform TextBlob in terms of accuracy, recall and F1 scores. TextBlob has a higher precision, indicating that it is more conservative in making positive predictions. This may reflect the fact that VADER is trained on social media, and therefore is more attuned than TextBlob to the informal, slang-ridden, online discourse of Amazon product reviews.

[1] https://www.geeksforgeeks.org/python-sentiment-analysis-using-vader/
[2] https://www.linkedin.com/pulse/textblob-vader-sentiment-analysis-abdul-hadi-mohamad/



Comparison Between Vader and Textblob Confusion Matrix

Textblob positive class:

Higher false negatives compared to vader. This suggests that textblob may struggle with correctly identifying positive sentiment in some cases.

Vader positive class:

Higher true positives and lower false negatives compared to textblob. This indicates that vader performs better at identifying positive sentiment.

Textblob negative class:

Higher true positives compared to vader. This suggests that textblob may be better at identifying negative sentiment.

Vader negative class:

Lower false negatives compared to textblob. This indicates that vader is more effective at identifying negative sentiment.

Neutral class (both):

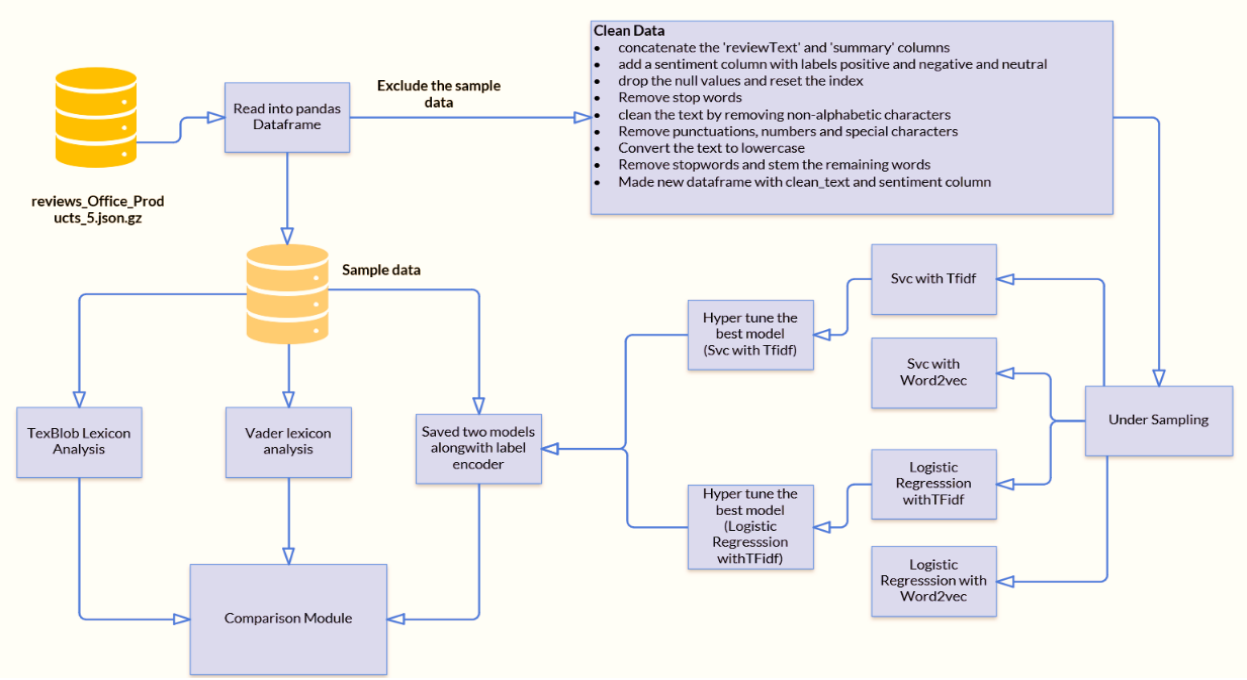
Both models seem to have some difficulty in correctly identifying neutral sentiment, as indicated by the higher false negative rates.

Overall:

We can not that choice between textblob and vader may depend on the specific requirements of the task. Each model may perform better in different contexts or with different types of text data.

PHASE 2

For the second phase of the project, more preprocessing steps were taken. These steps are summarized in the following flow diagram:



The dataset (review of office products) was extracted from the database into a pandas dataframe. A sample of data was extracted from the dataframe and set aside for the comparison module, where the results from all the models—both lexicon-based and machine-learning-based—could be compared. The remaining data were then cleaned and then under-sampled for the sake of balanced classes. The reviews were then vectorized with two text representation methods, word2vec and Tfidf vectorizer.

MODELS: PHASE 2

For the second, machine-learning based, analysis phase, two models were chosen:

SVC (Support vector classifier) and logistic regression. Each of these models were then fed with two different vector representation of review texts: one based on **word2vec**, and the other based on **Tfidf vectorizer**. Best model out of the four were finetuned to extract the most optimal parameters. In each case, the best estimator was chosen.

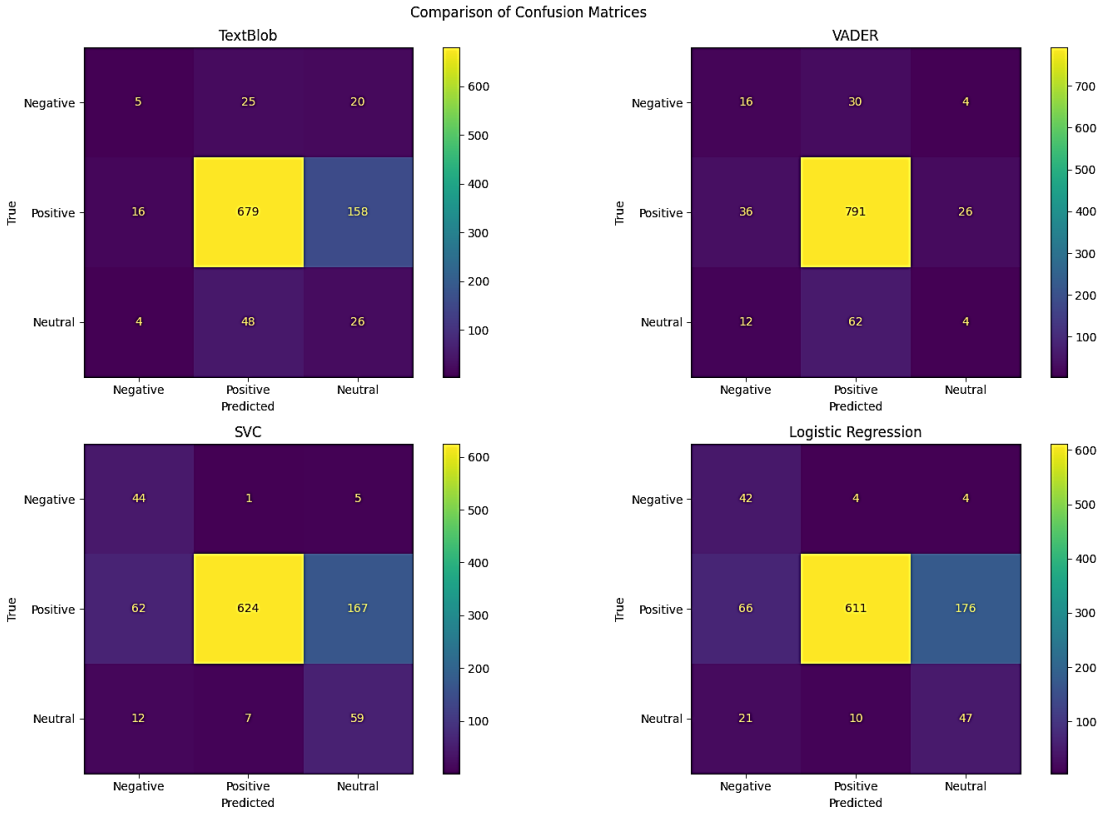
COMPARISON OF ALL MODELS

The following table shows the comparative output of all models, including the machine learning based models, in the second phase of the project:

	Algorithm	Accuracy	Precision	Recall	F1
0	Texblob	0.723751	0.805440	0.723751	0.757164
1	Vader	0.826707	0.801022	0.826707	0.812372
2	SVC	0.732926	0.897230	0.732926	0.782023
3	Logistic	0.713558	0.883101	0.713558	0.767337

- The Vader algorithm has the highest accuracy and F1 score, indicating that overall, it correctly identifies the sentiment of reviews most often and has the best balance between precision and recall.
- The SVC (Support Vector Classifier) has the highest precision, suggesting that when it predicts a review to be positive or negative, it is correct most of the time.
- Texblob and Logisoutperformedn have similar accuracies and F1 scores, with Texblob slightly outperforming Logistic Regression.
- Logistic Regression, despite having a lower accuracy and F1 score, has a relatively high precision which is only slightly less than that of SVC.

The following shows the confusion matrices for the four models:



- VADER seems to be the most reliable for positive sentiment detection and has the best balance between recognizing negative and positive sentiments.
- SVC is precise in identifying negative sentiments but is prone to miss positive sentiments (higher false negatives).
- Logistic Regression tends to over-classify reviews as positive, resulting in a high number of false positives.
- TextBlob appears to be the weakest overall, with the lowest number of true negatives and high misclassification rates

We ran SVC model again with two features: the clean text(cleaned text of concatenated columns of reviewText and summary) and Helpfulness. Here helpfulness denotes the number of “helpful” votes divided by total number of votes per review. The following table shows an example of “helpfulness” calculation.

FUTURE WORK

After reviewing the attached paper "Recommender systems based on user reviews: the state of the art", there is much more that can be done with the dataset and the information it provides to further enhance the recommender system. The paper list different avenues that can be taken to preprocess the data and exploit the review elements to enhance the performance. One aspect that can be explored is considering review helpfulness. This is where a prospective consumer of a product considers a previous review to be helpful, which can be seen as secondary rating. Giving a positive vote for a review further validates or authenticates the rating given.

$Helpfullness$= ($Number\ of\ helpful\ votes$) / ($Total\ No.\ of\ votes$)

Consider the following pseudo code:

```
Calculate the quality score for each review

# Quality score = Number of helpful votes / Total number of votes

# For reviews with no votes, the quality score will be NaN or 0 (we will handle this later)

data['helpfullness'] = data['helpful_votes'] / data['total_votes']

data['helpfullness'] = data['helpfullness'].fillna(0)

# Display the updated dataframe

data.head()
```

More preprocessing was applied on the "helpfulness" column: 0 scores were replaced the average and scaled by the standard scaler.

Following is a screenshot of data frame after introducing helpfulness column.

	clean_text	helpful	sentiment	helpful_votes	total_votes	helpfullness
0	bought first hp c serv faith lost travel searc...	[3, 4]	Positive	3	4	0.750000
1	belat review feel oblig share view old workhor...	[7, 9]	Positive	7	9	0.777778
2	hp gx kick twenti year hp year old still flawl...	[3, 3]	Negative	3	3	1.000000
3	start financ stuff recent went look good time ...	[7, 8]	Positive	7	8	0.875000
4	simpl calcul discount cash flow one still best...	[0, 0]	Positive	0	0	0.000000

	Algorithm	Accuracy	Precision	Recall	F1
0	Texblob	0.723751	0.805440	0.723751	0.757164
1	Vader	0.826707	0.801022	0.826707	0.812372
2	SVC	0.732926	0.897230	0.732926	0.782023
3	Logistic	0.713558	0.883101	0.713558	0.767337
4	State SVC	0.708461	0.889461	0.708461	0.764565

- The last row represents the new SVC model, which slightly underperforms the previous SVC model. There could be several reasons for this underperformance. The helpfulness could be more indicative of the review's usefulness to other users rather than its sentiment. We could have used **probabilistic matrix factorization (MF)** framework for performing the rating prediction.

FINAL CONCLUSION

As this project comes to a close, through thorough exploration of the dataset and steps taken with the recommender system it is clear the VADER model was most accurate. While more can be done to this recommender system should more data be available as there is in a real-world application, VADER was seen as most accurate overall. Of course, there are different approaches that can be taken to handle the data depending on the information and dataset structure available within it. There was much learned through the uses of each model and each of its capacities handling this type of dataset, Amazon Reviews.

Agenda Items

2:00PM – 2:10PM	Get to know your team
2:10PM– 2:25PM	discuss what the project is about
2:25PM– 2:40PM	review project requirements and explore the dataset
2:40PM – 3:00PM	Discuss what approaches will be taken handling the data
2:55PM– 3:15PM	Delegate project workload among all team members

Additional information

After the meeting we agreed to keep in touch informally through social media application WhatsApp, to discuss project requirements or if any team member needs assistance with their workload portion throughout phase one completion.