Sentiment Predictor

By Haseeb Siddiqi

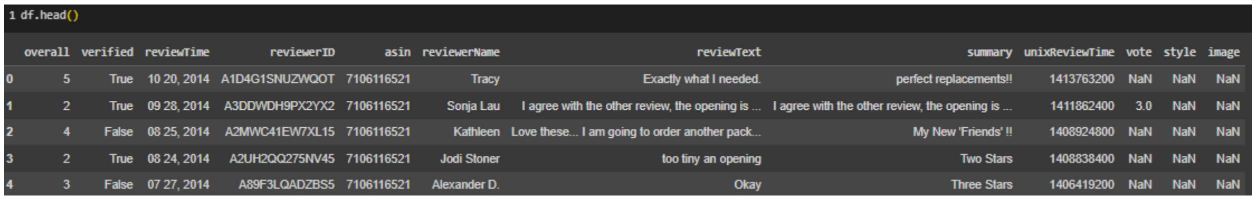
Introduction

The purpose of this project is to create a sentiment analysis model for customer reviews using both a lexicon approach and a machine learning approach.

Customer reviews vary significantly; some are short and concise, while others are lengthy and detailed. Often, customers praise a product in their review but give it a rating that does not reflect their positive feedback. Other times, people criticize the product but provide a higher rating. Being able to predict the sentiment of a review is useful because companies can adjust ratings to normalize reviews and better understand customer feedback.

This report outlines the preprocessing steps taken and highlights the differences between lexicon-based models and machine learning models. It also evaluates which approach is more suitable for sentiment analysis. The dataset used is Amazon_Fashion, which contains reviews on fashion products from Amazon.

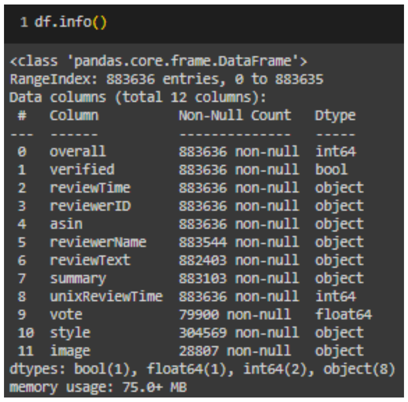
Dataset exploration



```python
df.head()
```

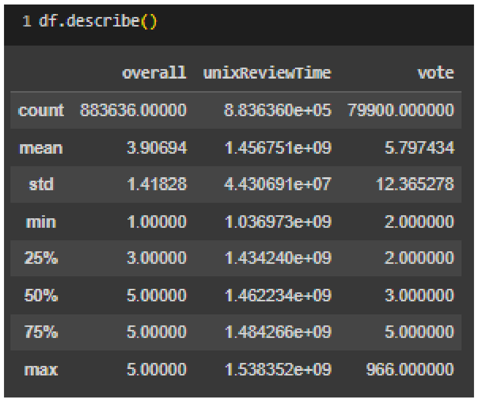
	overall	verified	reviewTime	reviewerID	asin	reviewerName	reviewText	summary	unixReviewTime	vote	style	image
0	5	True	10 20, 2014	A1D4G1SNUZWQOT	7106116521	Tracy	Exactly what I needed.	perfect replacements!!	1413763200	NaN	NaN	NaN
1	2	True	09 28, 2014	A3DDWDH9PX2YX2	7106116521	Sonja Lau	I agree with the other review, the opening is ...	I agree with the other review, the opening is ...	1411862400	3.0	NaN	NaN
2	4	False	08 25, 2014	A2MWC41EW7XL15	7106116521	Kathleen	Love these... I am going to order another pack...	My New 'Friends' !!	1408924800	NaN	NaN	NaN
3	2	True	08 24, 2014	A2UH2QQ275NV45	7106116521	Jodi Stoner	too tiny an opening	Two Stars	1408838400	NaN	NaN	NaN
4	3	False	07 27, 2014	A89F3LQADZBS5	7106116521	Alexander D.	Okay	Three Stars	1406419200	NaN	NaN	NaN

Column Name	Description
reviewerID	ID of the reviewer, e.g. A2SUAM1J3GNN3B
asin	ID of the product, e.g. 0000013714
reviewerName	Name of the reviewer
vote	Helpful votes of the review
style	A dictionary of the product metadata, e.g., "Format" is "Hardcover"
reviewText	Text of the review
overall	Rating of the product
summary	Summary of the review
unixReviewTime	Time of the review (unix time)
reviewTime	Time of the review (raw)
image	Images that users post after they have received the product



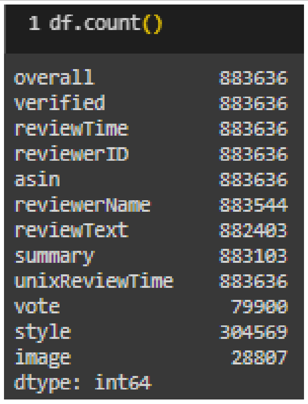
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 883636 entries, 0 to 883635
Data columns (total 12 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   overall         883636 non-null  int64
 1   verified        883636 non-null  bool
 2   reviewTime      883636 non-null  object
 3   reviewerID      883636 non-null  object
 4   asin            883636 non-null  object
 5   reviewerName    883544 non-null  object
 6   reviewText      882403 non-null  object
 7   summary         883103 non-null  object
 8   unixReviewTime  883636 non-null  int64
 9   vote            79900 non-null   float64
 10  style           304569 non-null  object
 11  image           28807 non-null   object
dtypes: bool(1), float64(1), int64(2), object(8)
memory usage: 75.0+ MB
```



```python
df.describe()
```

	overall	unixReviewTime	vote
count	883636.00000	8.836360e+05	79900.000000
mean	3.90694	1.456751e+09	5.797434
std	1.41828	4.430691e+07	12.365278
min	1.00000	1.036973e+09	2.000000
25%	3.00000	1.434240e+09	2.000000
50%	5.00000	1.462234e+09	3.000000
75%	5.00000	1.484266e+09	5.000000
max	5.00000	1.538352e+09	966.000000

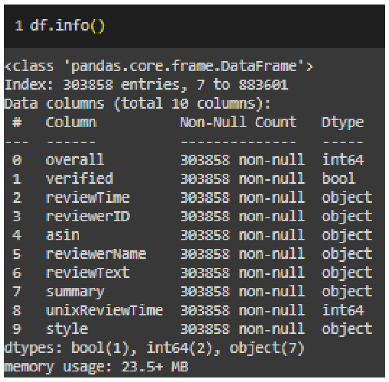


```python
df.count()
```

```
overall           883636
verified          883636
reviewTime        883636
reviewerID        883636
asin              883636
reviewerName      883544
reviewText        882403
summary           883103
unixReviewTime    883636
vote               79900
style             304569
image              28807
dtype: int64
```

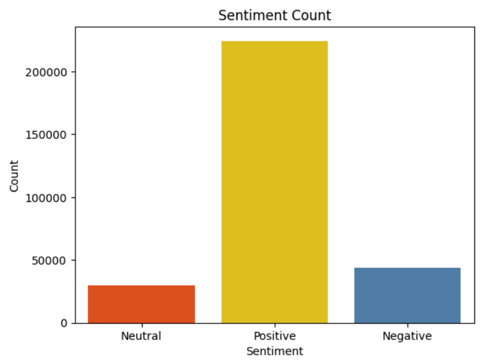
Dataset Preprocessing

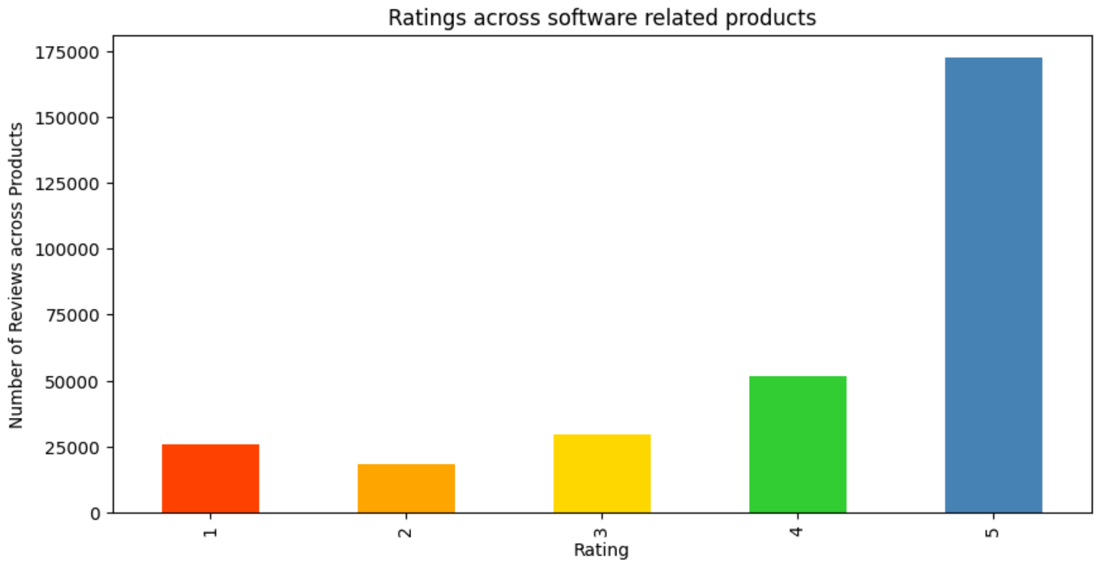
- **Removed Columns**: The image column was dropped as it was not relevant for text analysis. The vote column was also excluded due to excessive missing data.
- **Missing Values**: Rows with missing values were removed to ensure consistency.
- **Duplicate Reviews**: Duplicates were identified by checking if the same user reviewed the same product (reviewerID and asin).
- **Sentiment Column Creation**:
    - Reviews with a rating ≥ 4 were classified as positive.
    - Reviews with a rating ≤ 2 were classified as negative.
    - Reviews with a rating of 3 were classified as neutral.
- **Unverified Reviews**: Removed to prevent potential bias from bot-generated reviews.



```
df.info()
```

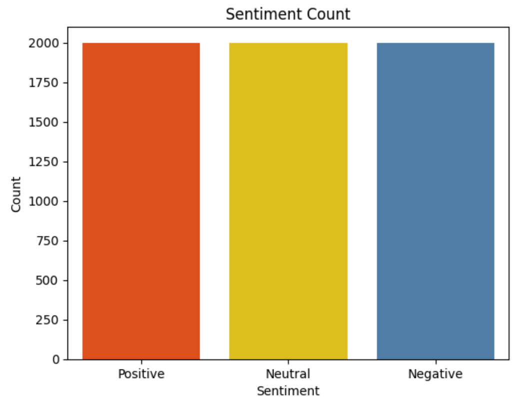
```
<class 'pandas.core.frame.DataFrame'>
Index: 303858 entries, 7 to 883601
Data columns (total 10 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   overall         303858 non-null  int64
 1   verified        303858 non-null  bool
 2   reviewTime      303858 non-null  object
 3   reviewerID      303858 non-null  object
 4   asin            303858 non-null  object
 5   reviewerName    303858 non-null  object
 6   reviewText      303858 non-null  object
 7   summary         303858 non-null  object
 8   unixReviewTime  303858 non-null  int64
 9   style           303858 non-null  object
dtypes: bool(1), int64(2), object(7)
memory usage: 23.5+ MB
```

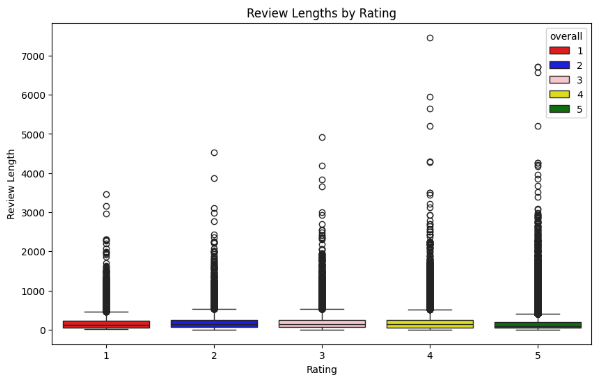


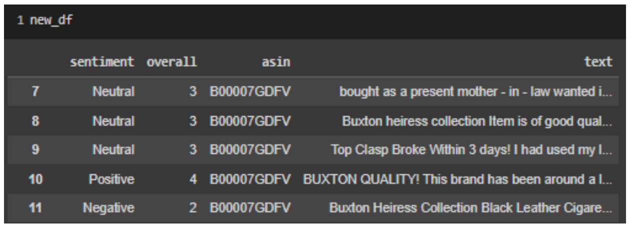


Text Preprocessing

- Combined reviewText and summary into a new column called text.
- Added a text_length column to compare review length with given ratings.
- Balanced data by taking 2000 samples from each sentiment class.
- Cleaned the text by removing URLs, hashtags, digits, and other unnecessary elements.
- Removed extra columns







```
1 new_df
```

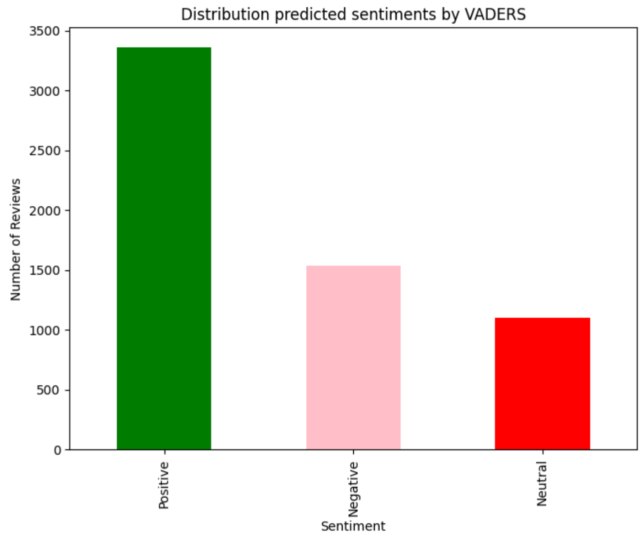
	sentiment	overall	asin	text
7	Neutral	3	B00007GDFV	bought as a present mother - in - law wanted i...
8	Neutral	3	B00007GDFV	Buxton heiress collection Item is of good qual...
9	Neutral	3	B00007GDFV	Top Clasp Broke Within 3 days! I had used my l...
10	Positive	4	B00007GDFV	BUXTON QUALITY! This brand has been around a l...
11	Negative	2	B00007GDFV	Buxton Heiress Collection Black Leather Cigare...

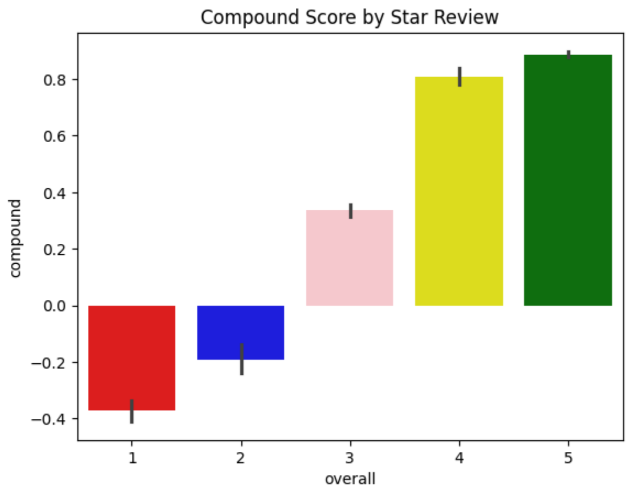
Lexicon Models

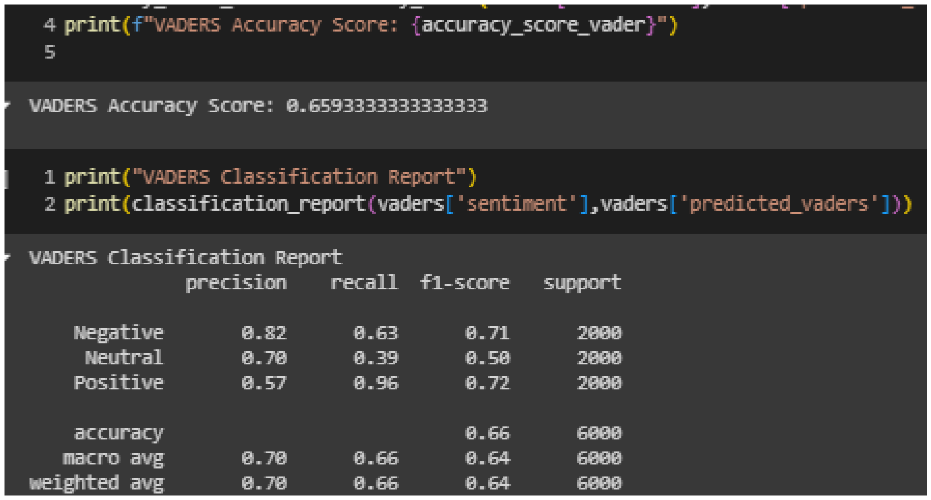
The lexicon models chosen for this project are VADER and TextBlob. VADER is extremely useful because it is tailored toward customer reviews and feedback analysis. It is context-sensitive, meaning it checks sentence structure to better understand the context and determine the sentiment. VADER outputs four different scores: positive, negative, neutral, and compound. If a sentence has a positive sentiment, it receives a higher positive score. This also applies to negative and neutral sentiments, which are reflected in their respective scores. The compound score is the final output, combining the positive, negative, and neutral scores to determine whether the sentiment is positive, negative, or neutral.

To further enhance VADER's capabilities, I experimented with adding specific keywords that affect the sentiment score. For example, if the phrase "five star" appears in a review, it marks the sentiment as positive, even if the compound score doesn't reach the positive threshold. This adjustment accounts for the Amazon review dataset, where customers often mention their ratings in the review text. Afterward, the sentiment is classified based on the compound score. However, due to overlaps between categories, the VADER model sometimes loses accuracy.

TextBlob is also used for sentiment analysis in product reviews. It employs techniques such as tokenization, n-grams, word inflection, and lemmatization. Like VADER, TextBlob provides negative, neutral, and positive scores, along with a final sentiment score. However, TextBlob tends to be less accurate in predicting the final sentiment due to even more significant overlaps between categories, leading to a higher rate of false predictions.





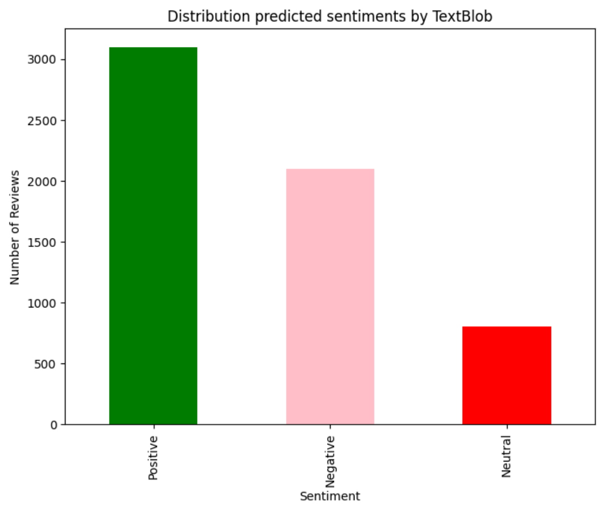


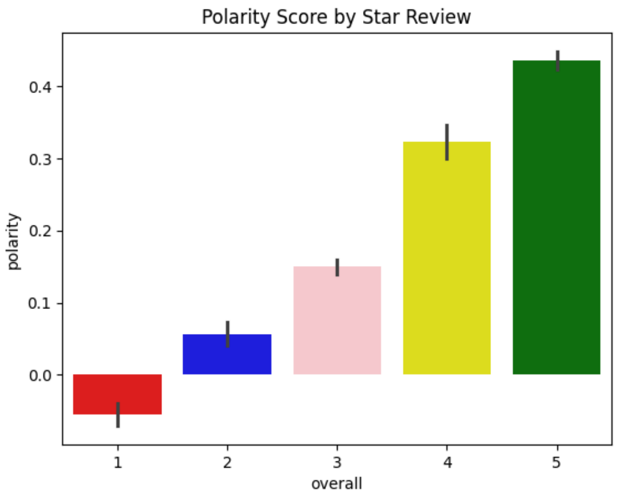
```
print(f"VADERS Accuracy Score: {accuracy_score_vader}")

```

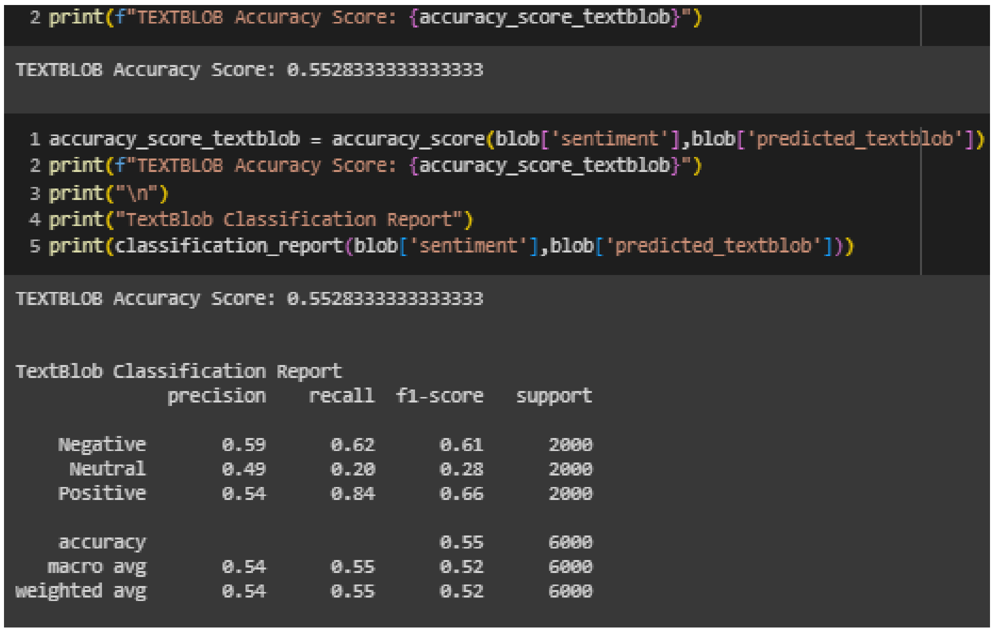
```
VADERS Accuracy Score: 0.6593333333333333
```

```
print("VADERS Classification Report")
print(classification_report(vaders['sentiment'],vaders['predicted_vaders']))
```

```
VADERS Classification Report
              precision    recall  f1-score   support

    Negative       0.82      0.63      0.71      2000
     Neutral       0.70      0.39      0.50      2000
    Positive       0.57      0.96      0.72      2000

    accuracy                           0.66      6000
   macro avg       0.70      0.66      0.64      6000
weighted avg       0.70      0.66      0.64      6000
```







```
print(f"TEXTBLOB Accuracy Score: {accuracy_score_textblob}")
```

```
TEXTBLOB Accuracy Score: 0.5528333333333333
```

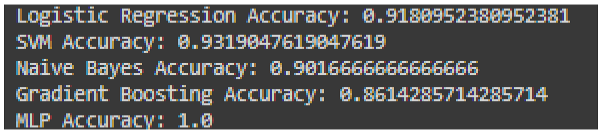
```
accuracy_score_textblob = accuracy_score(blob['sentiment'],blob['predicted_textblob'])
print(f"TEXTBLOB Accuracy Score: {accuracy_score_textblob}")
print("\n")
print("TextBlob Classification Report")
print(classification_report(blob['sentiment'],blob['predicted_textblob']))
```

```
TEXTBLOB Accuracy Score: 0.5528333333333333


TextBlob Classification Report
              precision    recall  f1-score   support

    Negative       0.59      0.62      0.61      2000
     Neutral       0.49      0.20      0.28      2000
    Positive       0.54      0.84      0.66      2000

    accuracy                           0.55      6000
   macro avg       0.54      0.55      0.52      6000
weighted avg       0.54      0.55      0.52      6000
```

Machine learning models

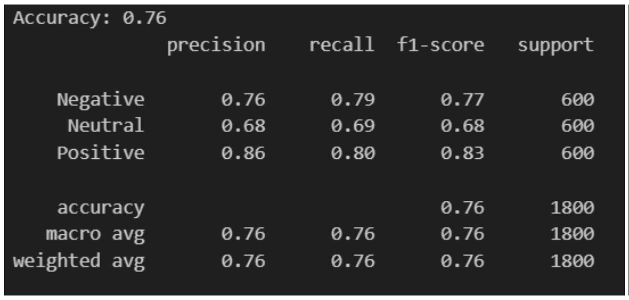
To get the data ready for the machine learning models, it first needs to be vectorized. The vectorization method chosen for this project is TF-IDF vectorization. TF-IDF is similar to Bag of Words but it reduces the weight of common words and increases the weight of rare ones. This method is beneficial for reviews since it decreases the weight of regular words, while words that indicate sentiment, being rarer, are weighted higher.

There are many different ways to build a machine learning model. To decide which models to use, multiple models were trained, and those with the highest training scores were selected. The models chosen were Logistic Regression, Support Vector Machine, and Multi-Layer Perceptron.

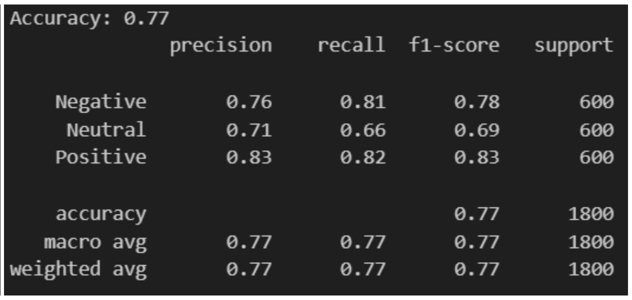


```
Logistic Regression Accuracy: 0.9180952380952381
SVM Accuracy: 0.9319047619047619
Naive Bayes Accuracy: 0.9016666666666666
Gradient Boosting Accuracy: 0.8614285714285714
MLP Accuracy: 1.0
```

To get the best score with the Support Vector Machine Model, Grid Search Cross validation was used. GridSearchCV tests out different combinations of hyperparameters in order to get this best result.

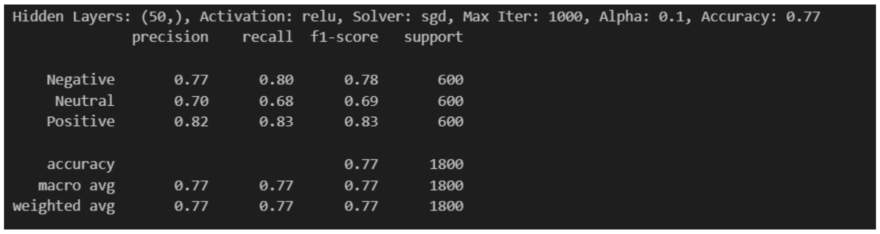


```
Accuracy: 0.76
              precision    recall  f1-score   support

    Negative       0.76      0.79      0.77       600
     Neutral       0.68      0.69      0.68       600
    Positive       0.86      0.80      0.83       600

    accuracy                           0.76      1800
   macro avg       0.76      0.76      0.76      1800
weighted avg       0.76      0.76      0.76      1800
```



```
Accuracy: 0.77
              precision    recall  f1-score   support

    Negative       0.76      0.81      0.78       600
     Neutral       0.71      0.66      0.69       600
    Positive       0.83      0.82      0.83       600

    accuracy                           0.77      1800
   macro avg       0.77      0.77      0.77      1800
weighted avg       0.77      0.77      0.77      1800
```

The logistic regression model is only slightly more accurate than the SVM model. To get the best parameters for the Multi layered perceptron model loops were used to loop through the hyper parameters in the end the final accuracy was the same as Logistic Regression model



```
Hidden Layers: (50,), Activation: relu, Solver: sgd, Max Iter: 1000, Alpha: 0.1, Accuracy: 0.77
              precision    recall  f1-score   support

    Negative       0.77      0.80      0.78       600
     Neutral       0.70      0.68      0.69       600
    Positive       0.82      0.83      0.83       600

    accuracy                           0.77      1800
   macro avg       0.77      0.77      0.77      1800
weighted avg       0.77      0.77      0.77      1800
```

Conclusion

In conclusion, the machine learning models significantly outperform the lexicon-based models, achieving a highest accuracy of 77%. While this may seem relatively low, it is essential to recognize that these models aim to uncover the true sentiment of the reviews, which may sometimes contradict the given rating. For instance, a reviewer might assign a product a 3-star rating while their review text expresses a strongly negative sentiment, revealing a genuine negative sentiment. Such discrepancies naturally impact accuracy metrics.

These techniques are scalable and can be applied to larger and more complex datasets, providing a solid foundation for further research in sentiment analysis. Additionally, experimenting with advanced vectorization techniques, such as Bag of Words and Word2Vec, could further enhance performance and results.

Overall, machine learning models demonstrate greater viability for sentiment analysis on this dataset. However, lexicon-based models may still prove advantageous for other datasets. By experimenting with both approaches, researchers can identify the most effective model for a given task. Future work on this project could include condensing reviews and implementing automatic responses for reviews that pose questions, thereby improving usability and applicability.