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 may praise a product in their review but give it a rating that does not reflect their positive feedback. Other times people will be critical of the product but give it a higher rating. Being able to predict the sentiment of a review is useful because companies can use this to adjust the ratings of their reviews to normalize them which helps them understand the customers feedback better.

This report will go through the preprocessing steps taken and then highlight the differences between lexicon models and machine learning models and which one is more suitable for this task. The dataset that is being used is called Amazon_fashion for reviews on fashion products on amazon.

Dataset exploration

```
1 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 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

```
1 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
```

```
1 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

```
1 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
```

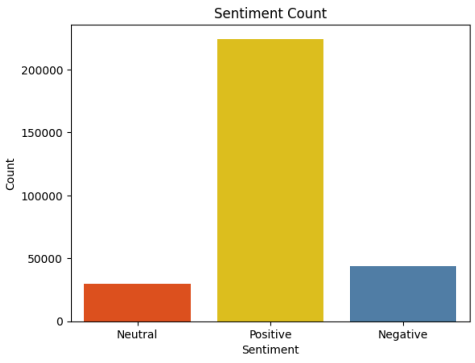
Dataset Preprocessing

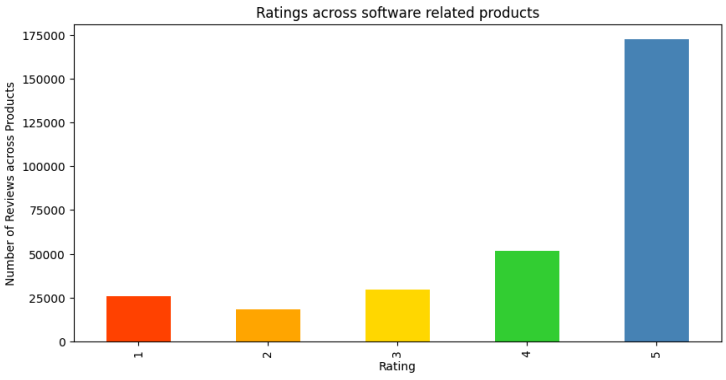
The image column was dropped because we are only interested in processing the text reviews. The vote column was also dropped because there were too many missing rows in that column which will reduce our sample size by too much. After this we removed all other missing values so each column has the same number of values.

```
1 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
```

To check for duplicate reviews we checked if a user made multiple reviews on the same product using the ReviewerID column and the asin column.

Next to create a Sentiment column the values from the overall column were taken and separated into 3 different groups depending on the value. If the review was 4 or higher it was classified as a positive review in the sentiment column, under 2 it was classified as a negative review and if the value was 3 it was classified as a neutral review. The dataset contained far more positive reviews than negative or neutral.





Any reviews that were made by unverified users were also removed. This is so we can guarantee there are no reviews from bots skewing the data.

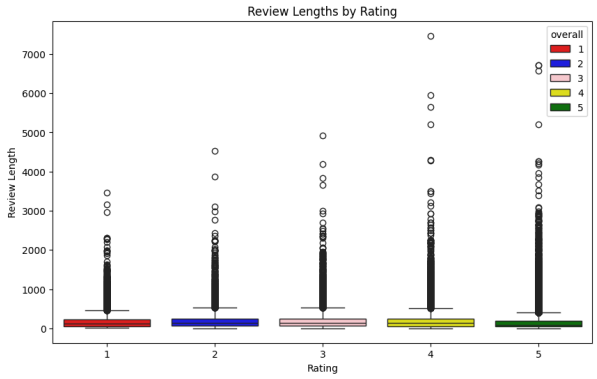
Text Preprocessing

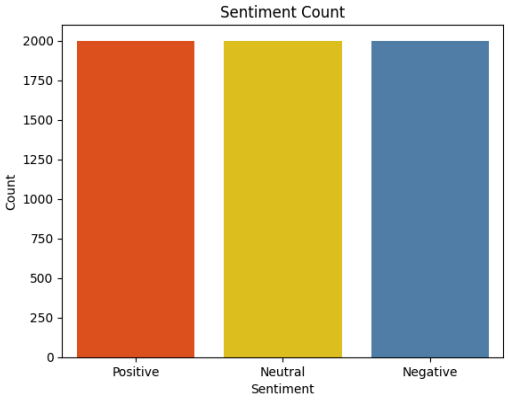
Next the unnecessary columns were removed and the summary and reviewText column were combined into a new column called text so we could preprocess them at the same time.

```
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...

Another column called text length was added to compare the review length with the given rating. This graph shows positive reviews tend to get longer reviews. Next to balance the data for the lexicon and machine learning models. 2000 samples were taken from each sentiment type this ensures the data isn't biased towards one sentiment.





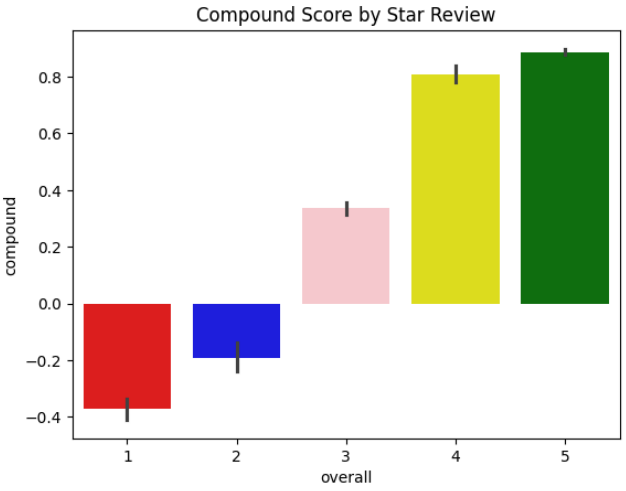
Next we remove urls, hashtags, digits and more to clean up the text before we use the lexicon models on it.

Lexicon Models

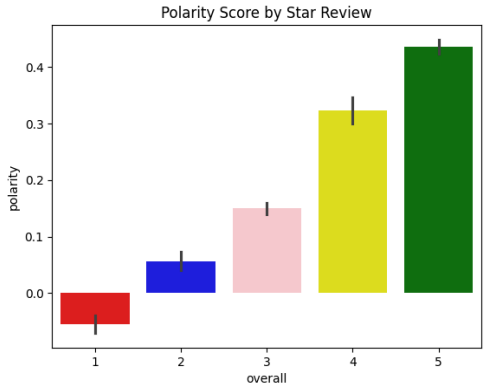
The lexicon models chosen for this project are Vaders and TextBlob. Vader's is extremely useful because it is tailored towards customer reviews and feedback analysis. It is context-sensitive, which means it checks the sentence structure to better understand the context of the sentence and determine its sentiment. Vaders, outputs four different scores, positive, negative, neutral and compound score. If the sentence has a positive sentiment it has a higher positive score, this also applies to the negative and neutral sentiments, which receive corresponding scores. The compound score is the final score; it combines the positive, negative and neutral score and gives the final score which determines if the sentence has a positive, negative or neutral sentiment.

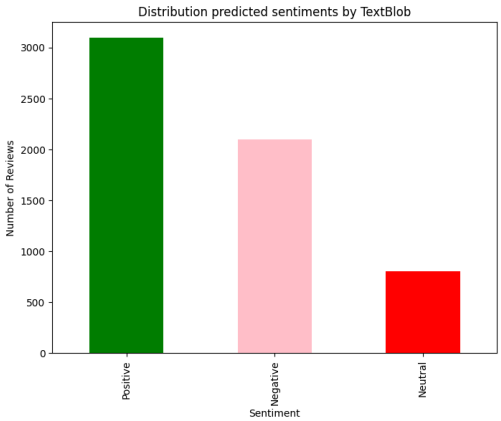
To further improve Vader's capabilities I experimented with adding some keywords that will affect the score if they appear in the review text. For example if the words five star were mentioned in the review it will mark the sentiment as positive even if the total compound score does not meet that threshold. This is because the data we are working with are amazon reviews and customers often mention the rating that they give in their review. Next we classify the sentiment depending on the compound score. But because there is overlap between the different categories the Vaders model loses accuracy

```
4 print(f"VADERS Accuracy Score: {accuracy_score_vader}")
5

VADERS Accuracy Score: 0.6593333333333333

1 print("VADERS Classification Report")
2 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
```



Text blob is also used for sentiment analysis in product reviews; it uses tokenization, n-grams and word inflection and lemmatization. Text blob also gives a negative, neutral and positive score and then a final score to determine the sentiment. Textblob wasn't very accurate in predicting the final score. It suffers from having even more overlap which leads to more false predictions.






```
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      0.28      0.36      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 perform significantly better than the lexicon-based models. The highest accuracy achieved is 77%. While this may seem a bit low, it's important to consider that the purpose of these models is to determine the true sentiment of the reviews, which might sometimes contradict the given rating. For example, some reviewers may rate a product as 3 stars, but their review itself could be very negative, indicating a true negative sentiment. Therefore, it's natural for the accuracy to not be exceptionally high.
These techniques can be applied to larger and more complex datasets. Additionally, other vectorization techniques, such as Bag of Words and Word2Vec, can also be experimented with to improve results.

Overall, machine learning models are more viable for sentiment analysis for this dataset. However, different datasets may benefit from lexicon models. Experimenting with both approaches helps us choose the most effective model for the job. This project can be expanded on by shortening the reviews down and adding automatic responses to reviews that leave questions.