AMS 561 Project
Amazon Product Reviews Analysis based on NLP

Zhilin Yang

May 1, 2023

1 Project Objectives

The main purpose of this project is to learn Natural Language Processing techniques. As a graduate student majoring in AMS, the data I come into contact with are all in the form of numbers. However, with the rapid development of technology today, data already exists in various forms, such as numbers, texts and images. With the popularity of ChatGPT all over the world, NLP technology began to enter my field of vision. So I want to gain experience working with natural language data, which differs from the numerical data commonly encountered in AMS.

The specific objectives of the project can be divided into three parts. First part is data cleaning and preparation which includes removed missing values, text cleaning and text tokenized. The second part is Exploratory Data Analysis, including the plots of the reviews' scores distribution, frequent words and word clouds. Final part is Sentiment Analysis for different models, such as Logistic Regression, SVM, Naive Bayes and LSTM.

I did this project on my own. Unfortunately, due to personal time issue and lack of computer performance, there is still some content that I wanted to do that was not implemented, like Topic Modeling and Word Embeddings. I will continue to finish those parts when I have spare time.

2 Techniques and Tools

In this project, I utilized various natural language processing techniques and tools for text preprocessing, feature extraction, and sentiment analysis. The following is a brief overview of the techniques and tools used:

2.1 Text Preprocessing Techniques

- **Expand contractions**: I used the contractions package in Python to expand contractions in the text data, such as "can't" to "cannot".
- **Replace abbreviations**: I used a custom script to replace common abbreviations found in the Amazon product reviews dataset, such as "w/" for "with".
- **Remove HTML tags**: I used regular expressions to remove HTML tags from the text data, which often appear in product reviews.
- **Remove URLs**: I used regular expressions to remove URLs from the text data, as these do not contribute to sentiment analysis.
- **Normalize text**: I used the unicodedata package to normalize text by removing non-ASCII characters.
- **Remove special characters**: I used regular expressions to remove special characters, such as punctuation marks, from the text data.

2.2 Feature Extraction Techniques

- **Tokenization**: I used the nltk package to tokenize the preprocessed text data into words.
- **Stop word removal**: I used the nltk package to remove stop words, such as "and" and "the", from the tokenized data.
- **Stemming**: I used the nltk package to apply the Porter stemming algorithm to the tokenized data, reducing words to their root form.

- **Bag-of-words representation**: I used the CountVectorizer class from scikit-learn to convert the preprocessed text data into a bag-of-words representation, where each row represents a review and each column represents a unique word in the corpus.
- **Tf-idf representation**: I used the TfidfVectorizer class from scikit-learn to convert the preprocessed text data into a tf-idf representation, where each row represents a review and each column represents a unique word in the corpus weighted by its importance in the document.

2.3 Sentiment Analysis Techniques

- **Logistic regression**: I used the LogisticRegression class from scikit-learn to train a logistic regression model on the bag-of-words features and predict the sentiment of each review.
- **Support vector machines**: I used the SVC class from scikit-learn to train a support vector machine (SVM) model on the Tf-idf features and predict the sentiment of each review.
- **Naive Bayes**: I used the MultinomialNB class from scikit-learn to train a naive Bayes model on the bag-of-words features and predict the sentiment of each review.
- **LSTM**: I used the tensorflow package to implement a long short-term memory (LSTM) neural network for sentiment analysis on the preprocessed text data.

Overall, the use of these techniques and tools allowed me to preprocess the Amazon product reviews data, extract relevant features, and classify the sentiment of the reviews with high accuracy.

3 Results

My analysis of Amazon product reviews using various feature extraction techniques and machine learning algorithms yielded the following results:

Table 1: Accuracy comparison of different models

Model	**Accuracy**
Logistic Regression	0.7297644388930915
Logistic Regression(Scaled)	0.7431609872807557
SVM	0.7829987861302183
Naive Bayes	0.7244339672431082
LSTM	0.0912073627114296

Machine learning models are evaluated on a dataset divided into 80% training set and 20% test set

These results indicate that SVM was the most effective model for sentiment analysis of the Amazon product reviews, achieving an accuracy of 78.30%. Logistic Regression after scaled was the second most effective model, achieving an accuracy of 74.32%. Naive Bayes and Logistic Regression without scaling achieved moderate accuracies of 72.44% and 72.98%, respectively. The LSTM model, despite its potential for deep learning analysis, performed poorly, achieving an accuracy of only 9.12%.

4 Conclusions

My project demonstrates the potential of NLP techniques for sentiment analysis of Amazon product reviews. The results suggest that SVM is the most effective model for classifying reviews as positive or negative, followed by Logistic Regression after scaled. The Naive Bayes and Logistic Regression models also performed reasonably well.

The most frequent words and the word cloud of the reviews provides insight into the key factors that influence customers' opinions. This information could be valuable for businesses seeking to improve their products or services.

Overall, my project underscores the importance of NLP techniques for un-

derstanding unstructured textual data and demonstrates the potential for sentiment analysis to provide valuable insights into customer opinions. In the Future, I will continue this project, add Topic Modeling and Word Embedding contents, deal with the class imbalance problem.

A Graphics and Outputs

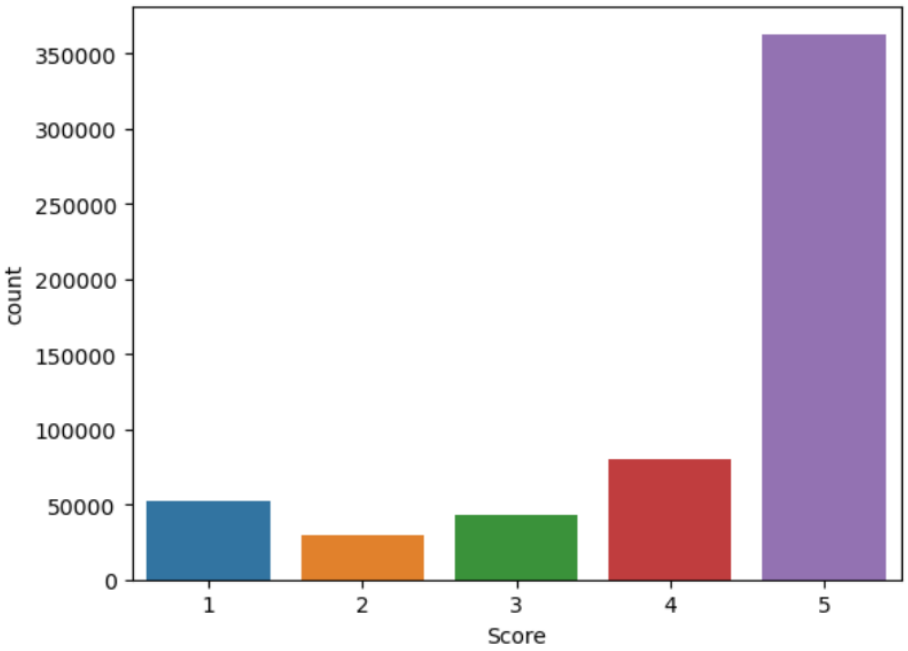

Figure 1: Scores Distribution Figure

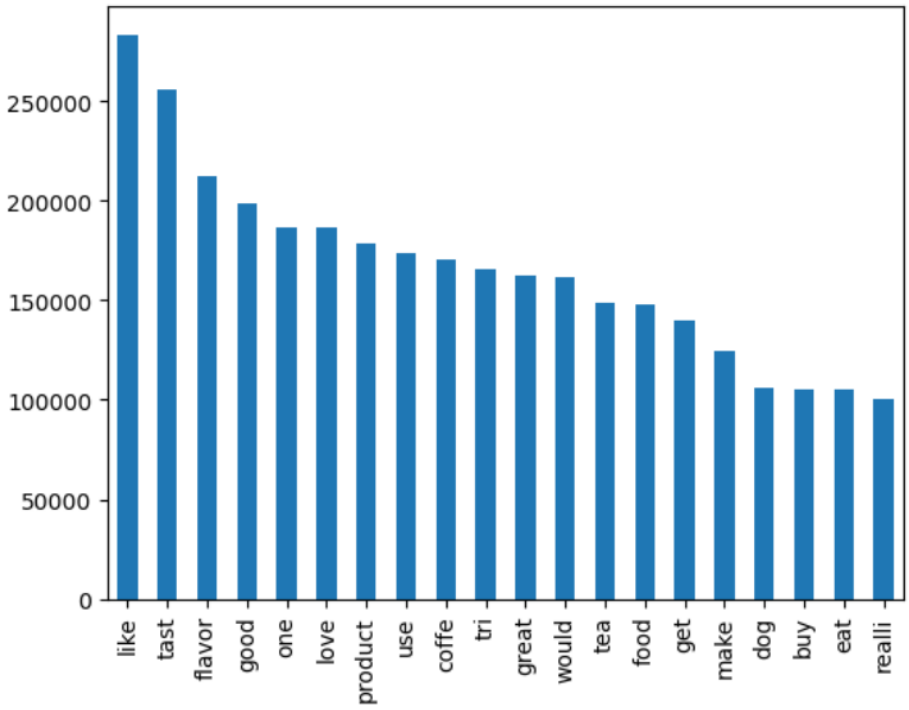

Figure 2: Frequent Words Figure

Figure 3: Word Cloud Figure

```
Accuracy: 0.7297644388930915
Confusion matrix:
[[ 6513   608   443   206  2599]
 [ 1514  1140   918   358  1970]
 [  818   639  2501  1167  3517]
 [  341   274  1120  3586 10777]
 [  758   191   656  1848 69224]]
Classification report:
              precision    recall  f1-score   support

           1       0.65      0.63      0.64     10369
           2       0.40      0.19      0.26      5900
           3       0.44      0.29      0.35      8642
           4       0.50      0.22      0.31     16098
           5       0.79      0.95      0.86     72677

    accuracy                           0.73    113686
   macro avg       0.56      0.46      0.48    113686
weighted avg       0.69      0.73      0.69    113686
```

Figure 4: Logistic Regression Output

```
Accuracy: 0.7431609872807557
Confusion matrix:
[[ 6769  1004   654   404  1538]
 [  895  2887   664   459   995]
 [  670   695  4284  1045  1948]
 [  494   569  1125  7204  6706]
 [ 1428  1094  1892  4920 63343]]
Classification report:
              precision    recall  f1-score   support

           1       0.66      0.65      0.66     10369
           2       0.46      0.49      0.48      5900
           3       0.50      0.50      0.50      8642
           4       0.51      0.45      0.48     16098
           5       0.85      0.87      0.86     72677

    accuracy                           0.74    113686
   macro avg       0.60      0.59      0.59    113686
weighted avg       0.74      0.74      0.74    113686
```

Figure 5: Logistic Regression(Scaled) Output

```
Accuracy: 0.7829987861302183
Classification report:
              precision    recall  f1-score   support

           1       0.72      0.74      0.73     10369
           2       0.65      0.35      0.46      5900
           3       0.62      0.42      0.50      8642
           4       0.61      0.37      0.46     16098
           5       0.83      0.96      0.89     72677

    accuracy                           0.78    113686
   macro avg       0.69      0.57      0.61    113686
weighted avg       0.76      0.78      0.76    113686
```

Figure 6: SVM Output

Accuracy: 0.7829987861302183

Classification report:

	precision	recall	f1-score	support
1	0.72	0.74	0.73	10369
2	0.65	0.35	0.46	5900
3	0.62	0.42	0.50	8642
4	0.61	0.37	0.46	16098
5	0.83	0.96	0.89	72677
accuracy			0.78	113686
macro avg	0.69	0.57	0.61	113686
weighted avg	0.76	0.78	0.76	113686

Figure 7: Naive Bayes Output

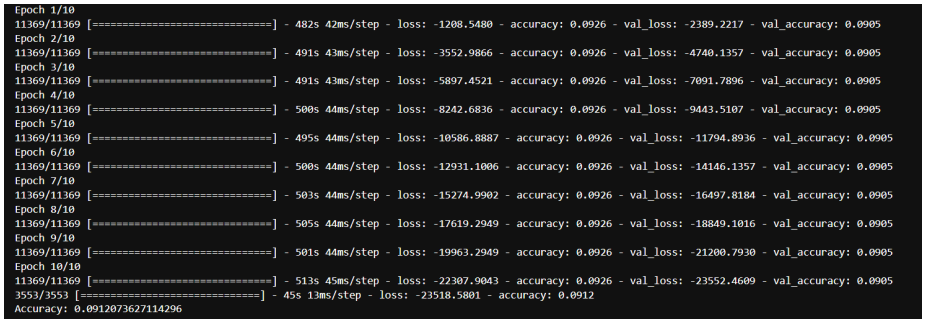

Figure 8: LSTM Output

B Reference

Data set's link

https://www.kaggle.com/datasets/snap/amazon-fine-food-reviews

Useful links

https://www.nltk.org/

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.svm

https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html#sklearn.feature_extraction.text.TfidfVectorizer

https://www.tensorflow.org/api_docs/python/tf/keras/layers/LSTM