CAPSTONE PROJECT #3 REPORT AMAZON REVIEW POLARITY CLASSIFICATION

Mike (Xiangnan) Shi

PROBLEM STATEMENT

1. Project Background and Goal

Customer reviews are always a good way to get a sense of the success of certain products and collect feedback for product improvement. Most review systems have a star rating function that allows for quick polarity analysis. But for those review systems without such star rating, manually reading all reviews to get the sentiment would be costly and time consuming. Therefore, a tool that's able to automatically analyze customer reviews can be of great value to a business. The goal of this project is to develop a model that's able to identify the sentiment of customer reviews.

2. Project Overview

This project is conducted using Python in a Jupyter Notebook environment. The source code can be found at Github[1]. The dataset is from a Kaggle competition[2].

The project follows a Data Science Method that consists of 6 steps as below:

1. Problem identification
2. Data wrangling
    a. Examine the data quality and fix missing values
3. Exploratory data analysis (EDA)
    a. Explore the statistics of the dataset
    b. Explore the potential relationship between target data and relevant features (texts in this case)
4. Pre-processing and Training Data Development
    a. Convert the text data to count vectors
    b. Apply stop words, filter out infrequent words, apply n-gram
    c. Generate train and test data set
5. Modeling
    a. Build different models and train them based on training dataset
    b. Predict sentiment based on testing dataset
    c. Evaluate the model performance
6. Documentation

1 Github link: https://github.com/stonewatertx/Springboard_GitHub/tree/main/Capstone_Project_3_Amazon_Reviews

2 Data link: https://www.kaggle.com/kritanjalijain/amazon-reviews

DATA WRANGLING

1. Data background

The original dataset contains 2 million amazon reviews. Each review has a review title, review body and label (either negative or positive). As language processing is computing intensive, this study only takes 30,000 reviews as a subset for analysis.

2. Data cleaning

The data appears to have a decent quality in the first place. There's only one missing data entry, which is dropped. Misspellings are observed in the dataset. Since spelling correction can a complex task, it's not part of this study. But further exploration is definitely recommended to see its impact on the model performance.

EXPLORATORY DATA ANALYSIS

An Exploratory Data Analysis (EDA) was conducted to help us better understand the data, explore the relations between features and target value and see if there's initial findings of pattern in the data.

1. Review polarity distribution

Of the 30,000 reviews in the dataset, almost equal numbers of positive and negative reviews were observed. This is likely because the data provider purposedly selected balanced dataset for research purpose.

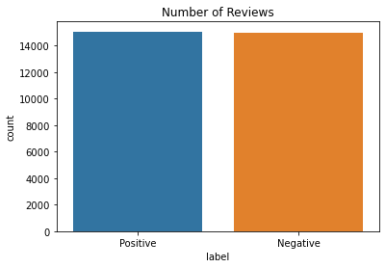

Figure 1. Review polarity distribution

2. Number of words in reviews

Most review titles are 2-6 words long, while review bodies are 30-110 words long. Negative review bodies tend to be a little longer than positive review bodies, but this trend doesn't seem to exist in review titles.

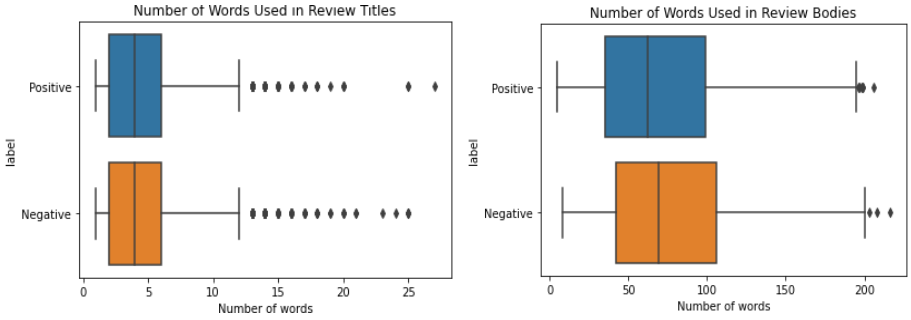

Figure 2. Number of words in reviews

3. Scatter text plot

A scatter text plot shows how frequent certain words appear in positive and negative reviews. Words to the bottom right means those frequently appearing in negative reviews but less so in positive reviews, such as ‘disappointing’. Words to the upper left means the opposite, such as ‘awesome’. Most words actually line up around the central diagonal line, which means they’re neutral in terms of frequency in both types of reviews.

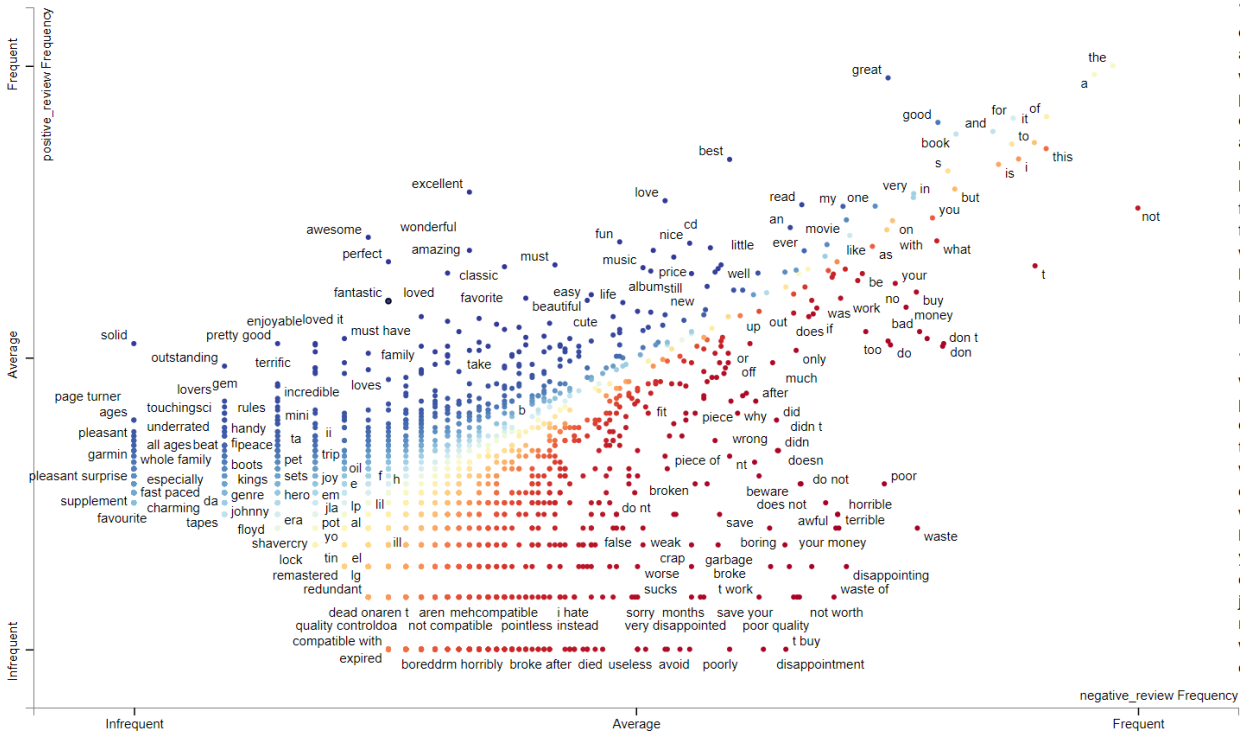

Figure 3. Scatter text plot

4. Word cloud

Two word cloud maps are generated to show the 50 most frequent words in positive and negative review titles. Unlike scatter text plot above, the word cloud maps don’t show the relative frequency of a word in positive reviews comparing to negative reviews. Some words just appear a lot in both types of reviews, such as book, good, buy, read, etc.

Word Cloud of Positive Reviews

Word Cloud of Negative Reviews

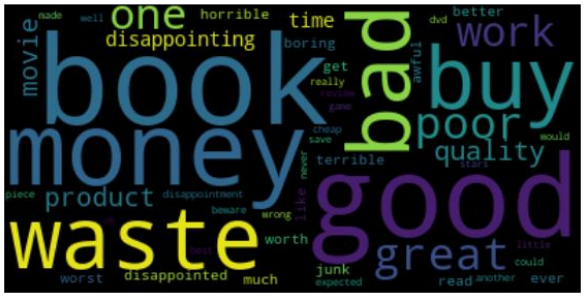

Figure 4. Word cloud of positive and negative reviews

MODELING

In this session, three different inputs including review titles, review bodies and a combination of both were used to predict whether a review is negative or positive. Four types of models were used in the study, which are Naïve Bayes, Logistic Regression, Random Forest and RNN.

Since most machine learning methods don't directly work with text data, the data has to be converted to some form of numeric format. Count vectors are used in this case. A count vector is simply a sparse vector that shows how many times each word appears in an observation (one review in this case). So, the first step is to convert all data into count vectors using CountVectorizer function from SciKit-Learn.

Several arguments can be defined when initializing a count vector function. The most common ones are whether or not to use stop words, the minimum frequency of a word in order to be included in the vectors, and how many words in sequency to be counted (n-gram). After several trials, it's found that including stop words as inputs seem to improve the prediction accuracy. The reason is probably that such stop words like "not" often change the sentiment of a sentence and can be important inputs to keep. Defining a minimum frequency of a word to be included in the vectors seem to reduce the input dimension significantly. The model performance doesn't necessarily drop much if the threshold is picked carefully. 3-gram seems to improve the prediction accuracy based on review bodies, but less so using review titles.

Count vectorized data was then used for inputs to train using Naïve Bayes, Logistic Regression and Random Forest. Most of the models just used default parameter settings without much tuning in this study. After the models were trained, they were then used to predict based on testing dataset. The predicted results were compared to actual results to get the accuracy.

The study also tried using TF-IDF instead of count vectors as model inputs. The results didn't show obvious improvement in this case. Thus count vectors were used for the most parts.

The data inputs for RNN model are a little different than the other three models. Firstly, all sentences were tokenized into words. Then tokenized words were represented by sequenced indices. A padding was then applied to transform all rows to the same length. An embedding layer was defined as the first layer in the RNN model, which converts integer indices in previous step into dense vectors. A bidirectional LSTM layer and a dense layer were then added. 8 epochs were specified for the sake of time.

PERFORMANCE SUMMARY

Below is a summary table of the prediction accuracy using different inputs and models. The accuracy ranges from 0.79 to 0.90. Using review titles for prediction seem to be the least accurate. Adding review bodies as inputs improves the accuracy significantly. The performance of different models is very close without much tuning. Even though Random Forest appears to underperform somewhat than other models, this might change if more hyperparameter tuning effort had been done.

	Model_Accuracy (Title)	Model_Accuracy (Body)	Model_Accuracy (Title&Body)	Model_Run_Time (secs)
Naive_Bayes	0.79988	0.866187	0.883888	0.212183
Logistic_Regression	0.805681	0.874487	0.89539	12.955616
Random_Forest	0.794279	0.832683	0.855586	78.578343
RNN	0.805381	0.862886	0.89529	Depending on epochs

TAKEAWAYS

- Several machine learning models including Native Bayes, Logistic Regression, Random Forest and RNN can be used for Amazon review sentiment analysis. Without much hyperparameter tuning effort, the accuracy can reach 0.86 to 0.9 depending the model selection.
- Combining review titles and bodies together seem to perform better then just using one of them alone.
- When creating count vectors as model inputs, keeping stop words seem to be favorable and result in higher accuracy in this case.
- Another technique to improve prediction accuracy is to specify n-gram. It seems to have more positive impact when using review bodies than titles.

FUTURE RESEARCH

- Additional data preprocessing is definitely worth exploring, such as misspelling correction, stemming and lemmatizing, etc. Often times those processing work can be computing intensive.
- Hyperparameter tuning isn't implemented in this study. As a result, the performance achieved is likely not close to optimal. It's strongly recommended to incorporate tuning as a future exploration topic.
- It's recommended to extend the training set to the original data set that has 2 million reviews, which is a lot more than what this study uses. It's expected that the accuracy can increase to some extent with more training data.