Amazon Product Reviews Sentiment Classification

Konstantinos Kostis

Abstract

The amazon product reviews contains numerous, more than 500k, customer reviews for amazon products. The main task is to answer whether it is possible to build a sentiment classifier for the reviews with a test accuracy greater than 80%. Based on the Jupyter analysis done, it is shown that **Logistic-Regression** algorithm provides an accuracy of **91%**. The methodology followed, includes text cleaning (html, stop-words, punctuation), tokenization, sentiment assignment according to score, TF-IDF vectorization and a comparison between Logistic-Regression and SVM based on test accuracy score.

Motivation

What i would like to do with the amazon-product-reviews dataset is to investigate the possibility of creating a sentiment analysis classifier for product reviews. Creating sentiment analysis tools is a very interesting problem and it benefits very much e-commerce companies. With such a tool there is no need for humans to waste hours to understand product reviews but simply delegate the task to a tool, so the workforce of the company can put in hours for other more important things.

Dataset(s)

The dataset used is the Amazon Product Reviews dataset and it can be downloaded from Kaggle. The data is a **single CSV file**. It contains **568454 reviews** of different amazon products and the number of **attributes is 10**. More specifically the fields include: **a record id, the id of product being reviewed, the id of the reviewer, the name of the reviewer, the numerator and the denominator for helpfulness of review, the rating/score of the review, a summary of the review and the full review text**.

It is important to note that the dataset contains no missing values.

Data Preparation and Cleaning

At a high level the data preparation and cleaning process included: dropping reviews with score 3, cleanup from html tags, lower-casing words, tokenization using NLTK WhitespaceTokenizer, removing of stop-words and punctuation, stripping punctuation, creation of new columns (Tokens and Sentiment).

I encountered a problem with the reviews of score 3 as some seemed positive, some seemed negative and some inconclusive and this is why i dropped these.

Research Questions

What kind of reviews does the dataset contain? Mostly bad? Mostly good?

Using sampling, how does a word cloud of positive reviews look like?

Using sampling, how does a negative reviews word cloud look like? Are there any clear differences from the positives world cloud?

Is it possible to create a sentiment classifier for reviews with an acceptable, >=80% (on the test set) accuracy?

Methods

After the data preprocessing (described in previous slide) a train-test split with 20% for test size was used. The classification metric picked was accuracy. Then a feature-extraction approach was used, using scikit-learn’s TF-IDF Vectorizer and a matrix was constructed to try different parameter combinations to tune the vectorizer. During this experimentation LogisticRegression and SVM classifier from scikit-learn were used to identify the method (Vectorizer + Classifier) that produces the best test accuracy score.

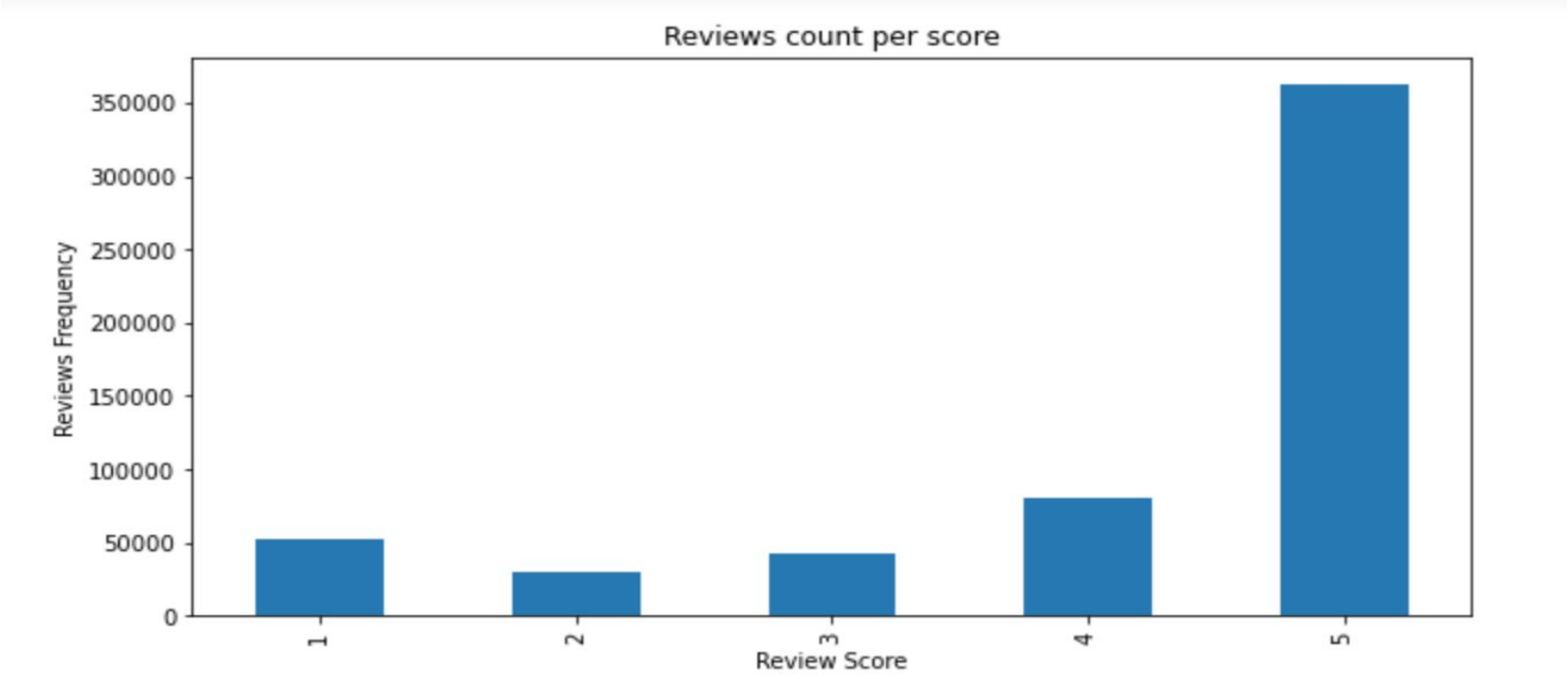
Methods (continued)

Test accuracy score along with training accuracy score are reported, in order to identify if the algorithms overfit the data. Confusion matrices are also given for further exploration.

For the visual representation of positives and negative reviews word clouds are created, using equal number of samples (40k) for both positives and negatives.

A simple bar plot helped in visualizing what kind of reviews exist in the dataset.

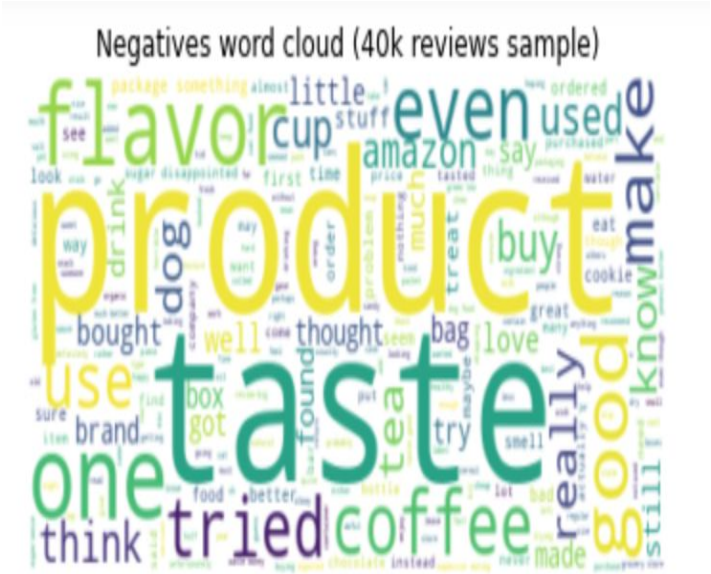
Findings (I - Plot)



Findings (I - Explanation)

So, it is evident that the dataset contains mostly positive reviews. In fact, reviews with score 4 and score 5, sum to 443777 which means they represent about 78% of the dataset since all the reviews are 568454.

Findings (II - Word clouds)



Findings (II - Explanation)

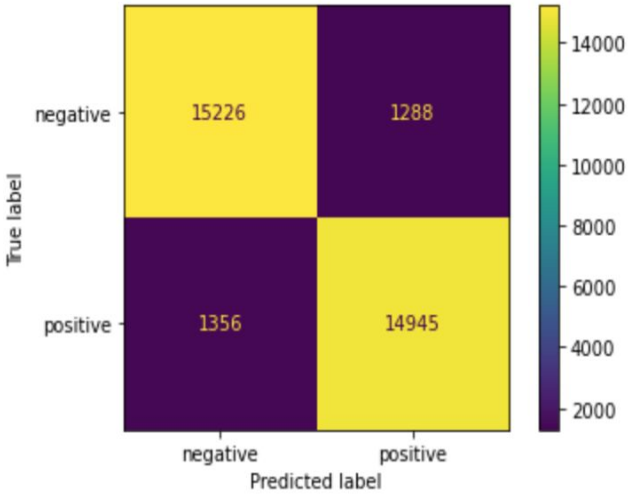
So if we look carefully we can see that in the positives word cloud we can find the words: 'best', 'delicious' which do not seem visually present in the negatives cloud. Still, words like 'taste', ‘love’, 'good', 'flavor', 'product' and others exist in both with different intensities, so the word-clouds do not help that much..

Perhaps, a better idea (for future improvement) is no to use sampling, but use all the positive reviews and all the negatives to form word-clouds.

Findings (III - Visuals)

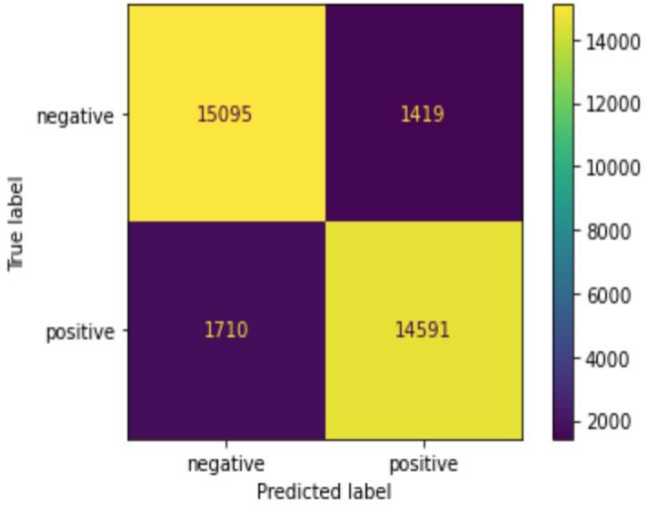
Out[15]:

	Test Accuracy	Train Accuracy
LogisticRegression	0.919427	0.942922



Out[16]:

	Test Accuracy	Train Accuracy
SGDClassifier	0.904647	0.914962



Findings (III - Explanation)

So it is evident that Logistic-Regression outperforms SVM classifier (SGD classifier with a **hinge** loss gives a linear SVM) with **91.94% vs 90.46%** regarding the chosen metric for the binary classification task which was **accuracy**, on the test set.

It can be also seen, that both algorithms have similar confusion matrices. For example in both algorithms the **False Negatives are more than the False Positives**. LR (1356 > 1288) and SVM (1710 > 1419). [But in general this is good if you think about it because no one wants bad reviews to be seen as good. This could hurt the popularity of an ecommerce site.]

Limitations

Reviews with score 3 have been excluded due to difficulty in interpretation (some seem positive, some seem negative, some are inconclusive).

For word clouds, sampling is used (40k for positives, 40k for negatives) and 5k words for each class were used for visuals, to not overload memory.

For classification, positives and negatives were balanced by dropping a big portion of positives, since negatives were fewer.

Conclusions

In the dataset, **most reviews are positive** representing ~78% of all the reviews.

The **word-clouds** of positive and negative reviews **did not give a good idea** about how these categories are represented, **possibly due to sampling**. In the future, all the reviews should be used along with incorporating reviews with score 3 as well, either as a neutral category or pushing them to negatives side.

Logistic Regression outperformed SVM with **91.94% vs 90.46%** test accuracy.

Acknowledgements

No one gave me feedback.

No other informal analysis took place. All steps and thought-process are documented in the Jupyter analysis.

I was inspired by a datacamp tutorial (link on references) regarding the setup of word clouds.

For the sentiment classification, i tried approaches mentioned in an SMU Scholar article for a comparative analysis of sentiment classification on amazon product reviews (link on references)

References

- https://www.datacamp.com/tutorial/wordcloud-python for word cloud
- https://scholar.smu.edu/cgi/viewcontent.cgi?article=1051&context=datasciencereview for sentiment classification
- https://scikit-learn.org/stable/modules/classes.html for scikit-learn documentation
- https://numpy.org/doc/stable/ for numpy documentation
- https://pandas.pydata.org/docs/reference/index.html for pandas documentation