Sentiment Analysis on Amazon Reviews
Machine Learning for Natural Language Processing 2020

Noémie Happi Nono
ENSAE IP Paris
noemie.happi.nono@ensae.fr

Solène Cochennec
ENSAE IP Paris
solene.cochennec@ensae.fr

Abstract

The aim of our project is to analyse the sentiment of Amazon reviews based both on full review and comment summary. We mainly used supervised models, including Deep NLP Learning algorithms, and achieved at most 85% accuracy.

1 Problem Framing

Alongside with the rise of the internet and the emergence of evaluation platforms, such as TripAdvisor or MediEval4i (to rate your doctor), retail giant Amazon is no exception to the rule and allows its customers to rate the products they offer, as well as leave a comment. Our approach shifted throughout the project from an unsupervised viewpoint to a supervised one. We first aimed at reevaluating rates given by customers, to establish more accurate ratings. However, given the unbalanced dataset and in order to efficiently compare the performance of several algorithms, we decided to perform traditional sentiment analysis classification to evaluate whether a review is positive, neutral or negative.

2 Experiments Protocol

Data (train and evaluation): We worked on the *"Art, Craft Sewing"* database composed of 494485 reviews and two variables of interest tokenized with several methods available in the NLTK library: `Summary` (equivalent of a title review) and `Reviewtext`, a denser text variable.

Model used: We first spotted possible trends and clusters with K-means algorithm. We then performed general and finetuned logistic regressions. We eventually decided not to include the SOTA Bert method as we implemented it in the lab sessions for sentiment analysis. We rather focused on other deep learning models. We started with a simple CBOW model (one layer, using GlobalAveragePooling1D), then added additional layers in hope to improve our results. As word order is very relevant when analysing sentiments, we updated our model by using a LSTM layer to improve our results. Finally, we used a pre-trained embedding for our last model with an extract of Glove vectors[1].

Implementation: We used simple and more complex embedding methods for the models: Count Vectorizer for K-means and Tf-idf for Logistic Regression. For the Deep Learning models, we used tokenizer and embedding models from Keras.

Model training: 60% of the data was training set for the Logistic regression and we used the whole data for the general one but a 10% sample for the finetuned model. For the Deep Learning models, we used only 10% of the dataset, and divided our sample dataset with 60% for train set, 10% for validation set and 40% for test set.

Please go to our github repository for more details.

3 Results

Our dataset being widely unbalanced (75% of positive reviews), we also considered other metrics: F1-score, precision and recall (cf notebook colab). The performance of the finetuned Logistic Regression (oversampling method to reach a more balanced dataset over the classes) was quite disappointing with overfitting effects.

After computing the algorithms above, we notice that the Logistic Regression is the most per-

[1] http://nlp.stanford.edu/projects/glove/

Algorithm	ReviewText	Summary
Logistic Regression	0.81	0.85
Finetuned Logistic Regression	0.67	0.72
CBOW	0.79	0.83
CBOW modified	0.78	0.81
With LSTM	0.75	0.80
Pre-trained model	0.76	0.76

Table 1: Accuracy obtained for each algorithm

forming model with the highest accuracy score. This is quite a surprising result as we could have expected the models using deep learning architecture to perform better. It may have been relevant to implement the state of the art BERT method to see whether or not it stood out and outperformed.

4 Discussion/Conclusion

Considering how dense the dataset was, we used a smaller dataset to run most of our algorithms. We can state that simpler models perform better than the most complex one.

To go further, we could have :

- Include the variable ”vote” in our models. Indeed, as part of our exploratory analysis, we noticed that the vote variable could be a strong asset : relevant reviews get higher votes. The challenge would be the high percentage of missing data (more than 80%).
- Compare with other datasets and see if the models built generalize well with reviews of other products.
- We could also implement other complementary and useful tasks using the same dataset like topic modelling (LDA model for example). The preprocessing and exploratory analysis we did (see in the colab) could have helped as we plotted the most frequent words.

References

Keras documentation . *https://keras.io*

Scikit-learn documentation. *https://scikit-learn.org*

Muller Benjamin. Machine Learning for NLP

SOTA embedding, *https://towardsdatascience.com*