Sentiment analysis of Amazon product reviews. Machine Learning for Natural Language Processing 2020

Dimitri CONDORIS
ENSAE Paris
dimitri.condoris@ensae.fr

Elias MOALLA
ENSAE Paris
elias.moalla@ensae.fr

Abstract

It is interesting to study Amazon product reviews on the internet as it might lead to interesting concrete applications. Indeed, if we can predict the sentiment of a particular user over a particular product, we can provide better services and more specific advertising hence redirecting users to more valuable content (for them) faster. To test out our sentiment analysis method, we used the dataset from the Amazon Alexa product reviews [1]. This dataset contains thousands of Amazon customers reviews for Alexa Echo, Firestick, Echo Dot and similar products.

1 Problem Framing

One could wonder why we would make such a model (saying if a review has a positive sentiment or a negative sentiment) when users can already leave star ratings for each of the product. The answer to this is that the model is mainly developed to extract features from the product that gives the positive sentiment for users allowing us to expose how good a product is to a certain demand using its description and those extracted textual features. This can therefore allow a website such as amazon to create faster and more pertinent searching bars conducting to a better user experience on the platform overall.

2 Experiments Protocol

Let us attempt to summarize in a modest but concrete way how to implement textual classification methods (to determine weither the sentiment is postive or negative) based on Machine Learning. The steps of sentimental product categorization can be classified into data preprocessing, model estimation, validation, optimization, and model evaluation. The pre-processing of text data is very different from conventional data. Text data must first be converted into a numerical representation before the algorithms are applied to it. The methods that allow this pre-processing are text vectorization techniques that are coupled with other pre-processing steps such as n-grams, removal of empty words of interest, TF-IDF [2] method and even denser techniques such as the FastText [3] word embedding. These steps also exclude punctuation marks and other inconsequential words such as articles, prepositions, and other conjunctions (for, among, etc.). Additionally, in this particular dataset, another step was necessary to remove emojis from the text reviews. Finally, stemming and lemmatization trim a word to its root: the plural becomes singular and various tense variants are reduced to their simple form. Once every of those transformations are done, we can plot the most important words we observe for each of the sentiment we are interested in (positive or negative):

Figure 1: Most important words (after preprocessing) for the positive reviews.

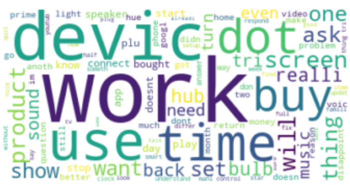

Figure 2: Most important words (after preprocessing) for the negative reviews.

Once the text data has been converted into a numerical representation, it is ready to be processed in classification models. The machine learning algorithms we have chosen for our project are: Naive Bayes, logistic regression, support vector machines (SVMs) and FastText (classification). Naïve Bayes is a probability-based classification method. It calculates the conditional probability for a block of words (comment) to belong to a particular class (sentimental category). A category (positive or negative) is assigned to each of the product labels for which the conditional probability is highest. Logistic regression can be used to classify an observation into one of two classes or into one of many classes (multinomial logistic regression). The system is trained using gradient descent, which is a method of simply minimizing the residual, i.e., the difference between the predicted feature and the prediction made. SVMs (Support Vector Machines) are a generalization of linear classifiers. The aim is to obtain THE best linear separator from the data. They seek to maximize the margin between the separator hyper-plane and the closest data (support vectors). Finally, FastText (classification), developed by the Facebook Research Lab, is a popular library for text classification. The library is an open-source project on GitHub and is quite active. The library also provides predefined templates for text classification. The Naive Bayes and SVM methods have several model hyper-parameters that can be adjusted. The performance / accuracy of the models depends very much on the right parameter values. In this step, a range of values is provided for each parameter and several models are created to see which parameter values give the best result. This method, called grid searching, is essential for refining and finding the optimal values of the model parameters. It allows us to obtain the best results and improve the accuracy (F1-score) of our methods.

3 Results

The evaluation of a machine learning algorithm is an essential part of any project. We used accuracy scores and mostly F1-scores (as the data is imbalanced) to measure the performance of our model. The different models trained and tested during this project ended up giving us not so satisfactory scores. Indeed, many of the trained models (at least for the ones using embedding) were not conclusive enough as the results were purely what we could have expect from a model with such an imbalance in labels (90% of good reviews) inducing constant outputs.

4 Discussion/Conclusion

It is also relevant to note the advantages and disadvantages of each model. This is especially true since none of the methods used have similar characteristics. Moreover, by comparing the models based on their response to the amount of data, it can easily be established that Naive Bayes and logistic regression are quicker to perform with small data (the US database in our case), while FastText (which could explain the observed outputs) performs better with larger data. The SVM method is efficient with both types of data. The SVMs also excel on the criterion of learning time, an area in which FastText, with its instantaneousness, has largely come out on top. Finally, it should be noted that the good score of Naive Bayes is based on a significant amount of pre-processing of the data and did not really repeat itself on the different testing procedures. Conversely, the FastText (embedding method) was applied with too little model structures and giving very poor results (like the other failing models: constant predictions): we did not explore all the possibilities of this promising model and this could hence be an interesting axis to pursue this work.

References

[1] Kaggle dataset link;
[2] Official TF-IDF wesbiste;
[3] Enriching Word Vectors with Subword Information by P. Bojanowski, E. Grave, A. Joulin and T. Mikolov, 2017, arXiv.