Classification of Amazon reviews
Machine Learning for Natural Language Processing 2022

Jules Kuperminc
jules.kupermin@ensae.fr

Etienne Maeght
etienne.maeght@ensae.fr

Abstract

We worked with data sets composed of thousands of reviews for different Amazon products. We performed binary and five-classes classification to predict whether a review was positive or not based on its content. We used two data sets to train our models: the Amazon Alexa Reviews and the Amazon Magazines Reviews. Our first goal was to identify the best performing models by evaluating the performance on classification of several models (Machine learning models applied to the sparse word count matrix, Neural networks). The second objective was to test the best performance models on the other data set and then try to perform some transfer learning using previously trained models. The small size of our data sets (with a dimension similar to the number of features) and the strongly unbalanced aspect of the two data sets (the majority of reviews are positive ones) were the two main constraints that we had to consider.

1 Problem Framing

Using different available methods, we wanted to classify a data set of Amazon reviews and identify the most suited method for this simple problem. We split the Amazon Magazines Reviews data set into train set and test set and perform several classification methods on the subset. The following step was to tune each model's hyperparameters and to plot the accuracy for each model to conclude.

2 Experiments Protocol

The data was preprocessed in three ways. First, a count based vector representation of each review was computed, along with a TfIdf representation. Second, a vectorized representation was used based on a Word2Vec model (fasttext), pretrained on the english Wikipedia dataset. Third, a representation was made using the Bert tokenizer. We used *matplotlib* and *seaborn* to explore the key characteristics of data sets. The two sets are heavily unbalanced, with a majority of positive reviews. Plotting word cloud and lengths of reviews based on their class highlights the specificity of the different classes of reviews. Given the unbalance aspect of our data sets, we introduce a weighted accuracy that over-penalizes misclassification of negative reviews. The performance of the model is also evaluated using the f1 score, the unweighted accuracy score and plotting the confusion matrix on the test set. For each configuration and model, we used GridSearch or RandomizedSearch methods to tune the parameters.
A first set of methods use the sparse word count matrix to perform bianry classification:

- **Regular random forest classification:** we first tried a random forest classifier, using *RandomSearchCV* to iterate on the number of estimators, the depth and the number of leafs.
- **Random forest with rebalanced set:** to adapt the unbalanced aspect of the training set, we created a rebalanced training set with only a fraction of the positive reviews. Using the hyperparameters from the previous method, we identified the best ratio between positive and negative reviews in the training set.
- **Weighted random forest:** using the unbalanced set, we performed random forest classification with different weights.
- **Random forest with a restricted number of features:** we then trained the weighted model on data sets with a limited number of training features. Features were eliminated based on their frequency in positive or negative comments.

- **Linear discriminant analysis:** this method yielded interesting results in classification on the test data set

- **Linear discriminant analysis with restricted number of features:** we were able to slightly improve the results of the LDA

- **Support vector classification:** we also tried to used SVC as those methods are commonly used in sparse matrix classification problems.

- **Support vector classification with restricted number of features:** trying different limits for the number of features did not yield interesting results

A second set of methods:

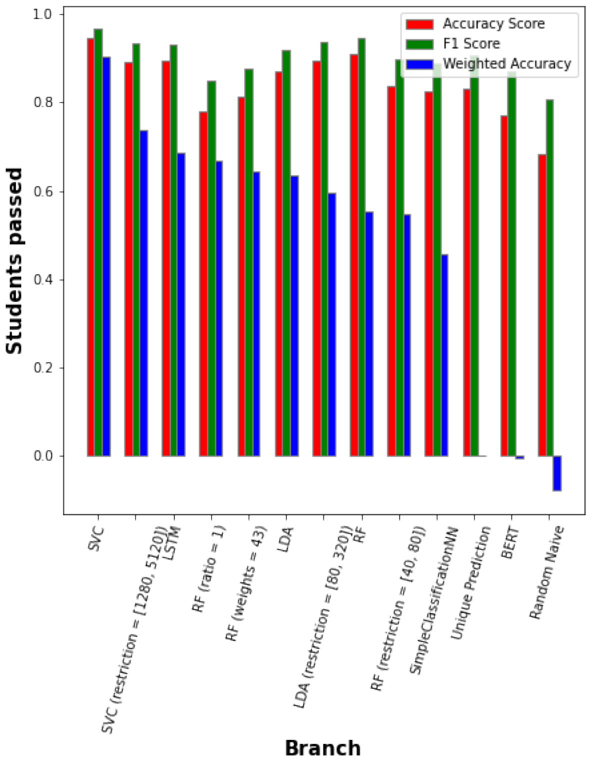
- **Neural Network with one hidden layer** The ReLU activation function was used, along with a softmax for classification and an average over non-zero tokens.

- **LSTM** A LSTM was used and the outputs were concatenated, followed by a linear layer.

- **BERT + finetuning** The BERT model was used with some finetuning.

3 Results

Our results are summarized in the following plot. We considered 3 accuracy metrics (accuracy score, f1 score and weighted accuracy score).



4 Discussion/Conclusion

We were able to explore several classic classification methods to identify the most efficient one on our data set. Due to time constraint and technical constraints, we were not able to perform advanced parameter tuning through GridSearch and RandomSearch. For this specific data set, support vector classification on the wordcount matrix and LSTM provide interesting results for the weighted accuracy. We can also observe that BERT provides non satisfying results for the weighted accuracy but still predicts with high unweighted accuracy and f1 score. The other models perform well on the data set. Additional considerations could have include testing the efficiency of our best models on an other similar data set and exploring more advanced parameter tuning.