Sentiment Analysis : predicting the impact of a financial newspaper headlines

Jérémy Berloty, Amaury Dreux

Description of the dataset

Our dataset consists of 4845 headlines of financial newspapers. They were anotated by experts in three different classes : "neutral", "negative" and "positive". We transform them in numerical variable for further processing. To describe the dataset, we will check the Zipf's law, the words that occur the most and finally the words that are the most associated with a negative or positive review. To do so we tokenise the sentences with the tweet tokeniser technique. In the first graph of the notebook we clearly see that the occurences are inversly proportional to the ranking of the words (in terms of occurencies). Therefore our dataset follows the Zipf's law. After only keeping the nouns and adjectives, we can plot the words according their occurences with a wordcloud, as shown below. Some of the words that occur the most are "EUR", "company", "Sales", "Profit", "Year", "Period" ...

We built the index *scale* = (*(appearences in positive reviews)+(appearences in negative reviews)/(total appearences)*). Thus it is between -1 (for the most negative ones) and 1 (for the most positive ones). The most negative words are for instance, "warning", "temporarily", "decreased", and more surprisingly "jobs", "workers", "staff" or "employment". The most positive words are for instance "narrowed", "grew", "efficient", "awarded" ... On average the words are more associated with positive reviews (0.2 approximately), which can be explained by the fact the data is unbalanced : there are **604** negative reviews, **1363** positive reviews and **2872** neutral ones. The unbalanced data could also be a problem for the prediction algorithm developed after.

Multinomial Naive Bayes for Bag of Words representation

The first prediction algorithm we are testing is a multinomial Naive Bayes algorithm, with Bag of Words representation for words. Bag of Words representation associates the words with its number of occurences in our corpus. We have also tried bi-gram and tri-gram, which means we don't take into acount occurences of individual words, but of n-sequence of words (respectively 2 and 3). In terms of global accuracy the three different models have similar results (between 0.74 and 0.75), but bi-gram and tri-gram have better precisions for the negative and positive reviews (see table below). Giving that datas are unbalanced, we can maintain that bi-gram and tri-gram give more robust results.

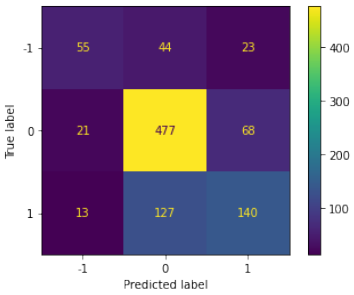
	Negative	**Neutral**	**Positive**
1-gram	0.78	0.77	0.65
bi-gram	0.84	0.75	0.68
tri-gram	0.85	0.75	0.72

TABLE 1 – Precision of the different Multinomial NB tested

Logistic Regression with SVD and Glove embeddings

We have tested Logistic Regression for prediction, and compared two embeddings : Singular Value Decomposition embedding and Glove Embedding. Interestingly enough, with Glove Embedding, the model predicts only neutral reviews. As for the SVD emebdding it gives better

accuracy (69%) and precision (as shown in the confusion matrix below). However it is less accurate than for the Multinomial Naive Bayes model.



LSTM with Fast Text Embeddings

We also choose to test a LSTM. The embedding of words is done from a pretrained vectors from Fast Text. The LSTM takes as input a sentence of embedded words (of dimension 300) and outputs a vector of size 268*(batch size). Then a fully connected layer gives 3 outputs (the three classes), which is fed into a softmax function which gives the final output. The final output is three probabilities, for each class, and the predicted class of the input will be the highest probability. To fit the model we use a cross entropy function, and we use an Adam Gradient Descent with a learning rate of 0.001. To avoid overfitting we add a dropout function. The dataset is split in a train dataset (80% of the dataset), and an eval and test dataset (10% each).

Unfortunately the model only predicts "neutral" reviews. To check whether the poor result came from the dataset, we tested the model by resampling the "neutral" label, and also by removing the "neutral" labels (we used another fucntion, SentimentRNN very close to the previous one, also using a Binary Cross Entropy instead), but both approach gave same results. This lead us to conclude that the problem did not come from the dataset. Therefore we identified two leads of improvement. First the problem could come from the embedding. Second it could come from the LSTM : indeed recurrent neural networks struggle to get long term relationships and have vanishing gradient issues. A transformer model like BERT could prove itself more efficient. We decided to focus on the second lead.

Prediction with BERT

BERT gives a training loss of 0.22 and a validation accuracy of 83% ! Below is shown the confusion matrix on the test dataset. The results are very satisfying and outperform all the other models.