Financial News Classification Model for NLP-based Bond Portfolio Construction

Thanos Konstantinidis, Arnau Bonet Farres, Yao Lei Xu, Tony G. Constantinides, Danilo P. Mandic
Department of Electrical and Electronic Engineering
Imperial College London
London, United Kingdom
Email: {a.konstantinidis16, arnau.bonet-farres18, yao.xu15, a.constantinides, d.mandic}@imperial.ac.uk

***Abstract*—The objective of this paper is to build an accurate classification model for news articles, and to study the effect of different news categories in corporate bond returns. The first step is to show how to classify news articles for traded companies into different defined categories. A classification model using Bag of Words (BoW) and Term Frequency Inverse Document Frequency (TF-IDF) methodology is developed to classify the news into 10 defined categories, and gives an average accuracy of 71%. After web scrapping news articles from 3 different sources to build a dataset of over 550 thousand articles, these are classified into 10 different categories. Natural Language Processing is one of the main techniques used in systematic portfolio construction. Also in the paper, the constructed portfolio based on the sentiment of news articles, by buying the corporate bonds of companies that have positive sentiment in the news, and shorting the bonds of companies with negative sentiment in the news is presented. The effect of each news category in corporate bond returns is studied through these portfolios.**

I. Introduction

Automatic document classification is a topic which has been researched thoroughly in academia and industry. The first classification system in libraries is the Dewey Decimal System [1], where books were classified into 1 of 10 groups of knowledge. There are several ways in which analysts, researchers, or fund managers try to predict corporate bond movements to make money for their bosses or themselves. In this paper, the use of Natural Language Processing as a technique to forecast corporate bond returns is presented. More specifically, the use of Sentiment Analysis as a trading signal is evaluated [16].

This paper aims to present a classification method of financial news articles into different categories. The first objective of this paper is to automate the classification of financial news articles into different categories that will be defined. In other words, the objective is to find the tone of the article, and buy the corporate bonds of companies that are seen positively in the news, and short the bonds of companies that are seen negatively in the news. There is considerable research in this area aiming to forecast stock returns with news articles, or even messages on social media to assess the sentiment of the public towards a specific company, but there is much less published research in the corporate bond space.

The main contribution of this paper is to build a dataset of financial news articles through web scrapping as a first step, while in a second step is to build a classification model that accurately classifies financial news articles into different categories that will be defined. The final objective is to evaluate the performance of portfolios built using sentiment analysis on the articles belonging to each category.

II. Related Work

Literature review has been conducted in different areas including document classification, portfolio construction and supervised machine learning. In literature, Feldman et al., have tried to build a model to classify financial news articles [3]. The objective of their research is to show that if news is selected carefully and only the relevant news are taken into account, there exists a correlation between the tone of the news and stock returns for a company. This system is called The Stock Sonar (TSS) [5], [6] and consists of 4,411 rules to classify news into "identified" (relevant), and "unidentified" (irrelevant) news. The relevant news is then classified into 1 of 14 categories. Using this methodology, classification of the news will be done in 1 of 10 categories using Feldman's categories as basis because they are mutually exclusive, collectively exhaustive (MECE), and able to cover all types of financial news articles.

Another work that has made improvements in this field is in [7], that instead of generating a very large number of rules to classify articles into different categories, it builds a machine learning model and trains it as a naive Bayesian classifier. It is interested in a specific section of a company's annual report: Management's Discussion and Analysis of Financial Condition and Results of Operations (MD&A).

A Bag of Words model (BoW) [11] and Term Frequency Inverse Document Frequency (TF-IDF) model were used in [8] to perform text classification. The model effectively has the objective of classifying a sentence describing a film into one of 35,000 categories. This model starts with the synopsis of 35,000 films which is available in the repository Wiki movie plots [9] in Kaggle.

The same methodology as in [8] will be followed in this paper, where the classification will be performed into news categories. To build the corpus set, the "synopsis" for each category of financial news articles, articles that contain the type of words that appear more often in each category will be needed. For the corpus set collection the data should be as MECE as possible, to make sure the article defining each category is only defining that category.

979-8-3503-3959-8/23/$31.00 ©2023 IEEE

III. Data Collection and Pre-Processing

This section presents the first step concerned the construction of the dataset that will be used. During this process, the work presented in [10] has been used as baseline. A web scraper that downloaded company-specific articles into a csv file for 3 different news sources (i.e., MarketWatch [17], The Motley Fool [19], Reuters [20]) has been developed. For each link we are able to identify the zone where the articles are placed, and process them one by one. Once the article zone is identified, the web scrapping algorithm does the scrolling down to be able to retrieve all the articles for every company. A similar procedure is used for each one of the news sources. Hence, for each news source and each company we are able to retrieve all the articles written, and separately identify the headline, the body and the date of publication. At the end of the scrapping, we have a csv file for each company and each source containing all this information. It is noted that the dataset is manually labelled on the corresponding categories supported by the model.

Some of the companies have articles dating back to 2005 or 2006, but it is not until about 2015 that the news sources follow a similar and consistent pattern of articles, and have a big number of articles per day. The amount of the dataset collected for the experiments is available in the second column of TABLE I. The dataset that will be used consists of 552,598 articles, but not all of those are directly relevant to a company's corporate bond returns. This section presents the natural first step towards classification of company- specific relevant articles which concerns the cleaning of the article dataset using Named Entity Recognition (NER). For this task, a bert-base-NER model which is a ready-to-use model trained on the CoNLL-2003 NER dataset [22] [47] [25] has been used, that is able to identify the different entities being mentioned in the article, and that returns a confidence score for each one of the entities mentioned in the article is required. It has been trained to recognize four types of entities: location, organizations, person and Miscellaneous.

For a given article, the model returns a set of different entities mentioned, and a confidence score for each of them. After testing for different values and trying to find the optimal threshold value, a threshold of 0.98 has been set, meaning that: if the company's article is supposed to be about has a higher score than 0.98, we keep the article, otherwise we reject it. In TABLE I. are the results of running the NER filter through our initial dataset is presented.

TABLE I. Total number of articles scrapped per source

News source	*# of articles pre processing*	*# of articles post NER filtering*
Marketwatch	309,187	236,214
Reuters	38,141	35,741
The Motley Fool	205,270	147,413
Total	**552,598**	**419,368**

The number of articles has been reduced by 24.1% in total. The effect is similar for MarketWatch and The Motley Fool, as they see reductions of just above 25% both.

IV. Classification Model: BoW and TF-IDF

After the clean dataset of articles, it is time to develop the classification model. The scope of this paper is to perform the classification task for 10 categories. Having 10 categories seems a good trade-off between complexity of the model and number of categories. From the 14 categories defined by Feldman [3], I merged some and ended up with the following 10 categories presented in that are: (i) Stock Price Update, (ii)Financials, (iii) Deals, Partnerships and M&A (iv) Legal, (v) Product (vi) Analyst Recommendation (vii) Employment (viii) Facilities (ix) Award (x) Other. The first step is to have, for each category, an article or series of articles describing that category perfectly, with the words that usually appear in the category. The set of articles describing each category is going to be called "Corpus set".

The objective is to find articles defining each category perfectly that have all the important words that usually appear in each category. Also, we want the articles describing a specific category to have no words important to other categories. I have selected between 5 articles per category, for a total of 45 articles in the corpus set (we do not classify into the category Other, so we have 9 categories to define). To avoid bias to certain companies in specific categories, I have built the corpus set with articles from 5 different companies in different sectors including Manufacture (3M [21]), Technology (Apple [22]), Oil, Gas and Energy (BP [23]), Financial Services (Citigroup [24]), and Healthcare (Stryker [25]).

A. Data preprocessing

The pre-processing steps performed are the following: (i) Change in lower case (ii) Punctuation & digits removal (iii) non-alphabetic characters removal (iv) Stop words removal (v) Words with length less than 3 removal (vi) Stemming (vii) Words conversion to a number using an existing English dictionary (viii) Vectorization (ix)TF-IDF calculation.

Using the TF-IDF, the terms that define that category exclusively are given a higher importance. Therefore, I have the 9 vectors defining each of the categories, where each of these has a length of 50,000. If we stack all the vectors on top of each other, we form a matrix with 9 rows, each of them containing the vector defining each category, called big matrix. Each article is preprocessed using aforementioned processes, and we end up with a vector $x = [a1, a2, \ldots .a50000]$ that contains the term frequency of the different terms in the article. Where ai is the frequency of the term i in the article. We multiply matrix A by column vector x to obtain the vector giving the correlation of vector x to every row of matrix $Ax = c = [c1, c2, \ldots .c9]$, where c_i is the correlation of vector x with row i of the matrix A. In intuitive terms, the value c_i is the similarity between the article to be classified and the article describing category i. The article will be classified in the category with which it has the highest correlation (similarity).

B. Model testing

Before going into the test results, let's remind that the benchmark for the classification model accuracy is 63%, obtained by Li [7], which is the highest disclosed accuracy obtained in an article classification model. To test the model, I took the labelled set of about 2,000 labelled articles and classified into categories with the big classification matrix. These are the accuracies obtained for each category.

Clearly some categories are very easy to correctly classify into, mainly Stock Price Update. This is because most of the articles in that category are AI-generated, so they use the same, or very similar words every day, which makes them very easy to identify for our model. The good accuracy of Stock Price Update is what pushes the weighted accuracy up to 85.5%, but some categories have very low accuracy, and the average accuracy is at 59.7%, below the benchmark of accuracy obtained by Li of 63%. The model is especially inaccurate for categories: Financials, Deals and Employment, with 40% accuracy or lower. Once I had this model and this accuracy, I started to think of ways to improve the model, they are outlined in the next section.

Aiming to improve the accuracy of the model, two approaches were followed: (1) evaluating the relative importance of information and (2) noise reduction. By now, the model is constructed, it turns the article into a bag of words [11], so the first sentence of the article has the same importance than the last one. This could be misleading because normally at the end of the article, the writer has changed topic slightly and is talking about different issues. Following the first approach, the input of the model was only the first n characters of each article (both in the corpus set and the articles to be classified). TABLE II. presents the accuracy results when only taking the first n characters in the article.

TABLE II. TOTAL NUMBER OF ARTICLES SCRAPPED PER SOURCE

Number of characters	*Average accuracy*	*Weighted accuracy*
Full Article	59.7%	85.5%
1000	60.9%	88.0%
500	70.9%	90.0%
300	**71.3%**	**91.1%**

We can see that the sweet spot is found at 300 characters of article length. Also, reducing the number of characters in each article improves the speed of the model as there is less information to process. The weighed accuracy is improved but the main improvement is found in the average accuracy. Now the worst performing categories are Employment and Analyst Recommendation, with an accuracy below 60%, almost the same as the previous average accuracy. The performance is much more homogeneous across the different categories, and the improved accuracy is 71%. TABLE III. Presents also the accuracy of the model in each category and in general, using the full article and in the case of using the first 300 characters.

TABLE III. MODEL ACCURACY TEST RESULTS

Category	*Accuracy (full articles)*	*Accuracy (first 300 characters)*
Stock Price Update	97 %	97.5 %
Financials	40.4 %	85.2 %
Deals, Partnerships and M&A	33.8 %	66.1 %
Legal	86.6 %	62.7 %
Product	48.2 %	64.2 %
Analyst Recommendation	74.6 %	59.3 %
Employment	33.3 %	52.3 %
Facilities	63.6 %	71.4 %
Award	60 %	83.3 %
Overall accuracy of the model		
Average accuracy	59.7%	71.3%
Weighted accuracy	85.5%	91.1%

The second approach, concerns the change is the rank of the classification matrix. The matrix dimensions are: 9 times the length of the dictionary (approximately 50,000), and the rank of the untouched matrix is 9 as all the rows are linearly independent. In order to achieve the reduction of the matrix's rank, SVD decomposition was followed [12]. Reducing the rank of the matrix is normally aimed at reducing the noise in the matrix. Using rank reduction in this context is usually known as Latent Semantic Indexing [13], [14], [26] because it reduces the noise of the matrix and helps find latent relationships between words and better assess their similarity. TABLE IV. presents the accuracy results of using this technique to reduce the rank of the matrix and testing on the labelled set.

TABLE IV. ACCURACY OF THE MODEL WITH RESPECT TO THE MATRIX RANK

Matrix rank	*Average accuracy*	*Weighted accuracy*
9	**71.3%**	**91.1%**
8	68.4%	89.8%
7	65.1%	88.5%
6	61.0%	87.7%

V. PORTFOLIO CONSTRUCTION USING CATEGORIZED DATASETS

This section presents the construction of the portfolio for each set of news. Each article has a date, a company name, a headline and a body of the article. The portfolios have been constructed to start trading on February 2015 until June 2021. For each article, the sentiment is calculated using the polarity metric. When we look at the number of "articles" per category, it is actually the number of sentiment data points, not articles.

The portfolios per category will be constructed with the top/bottom 10%, 20%, 30%, 40%, and 50% of sentiments, which will be called L/S 10%, L/S 20% etc., L/S standing for Long-Short. With the 50% portfolio we basically execute trades on all companies for which there is an article, because we long the corporate bonds of the top 50% of companies and we short the corporate bonds of the bottom 50% of companies. Indicatively, the cumulative chart for one category is presented in this paper, due to page limitations.

The portfolios built using Stock Price Update news have very similar performance to the unclassified portfolio. This category has the most sentiment data points, with over 1/3 of the total sentiment data points, so it seems natural that it is quite similar to the unclassified portfolio. The L/S 40% and L/S 50% do not perform so well as the other 3, and have negative returns over most of the trading period. They have Sharpe Ratios of only 0.26 and 0.11, below any of the benchmarks (Fig. 1). The low number of sentiment data points makes it impossible to make investment decisions during the first years, and this is why the portfolios are flat a lot of days. Also, when we do have enough pieces of news to trade, we are entering a very small number of trades, which means that there is almost no diversification and we have total exposure to the movement of a few corporate bonds, so the volatility of the portfolio is very big. This is why we need more sentiment data points.

VI. CONCLUSION

The classification model using Bag of Words and TF-IDF that has been presented has an average accuracy of 71%, beating the model in [15], which is the best available model, by 8%. The classification model successfully classifies articles into the different categories. The model is simple but effective,

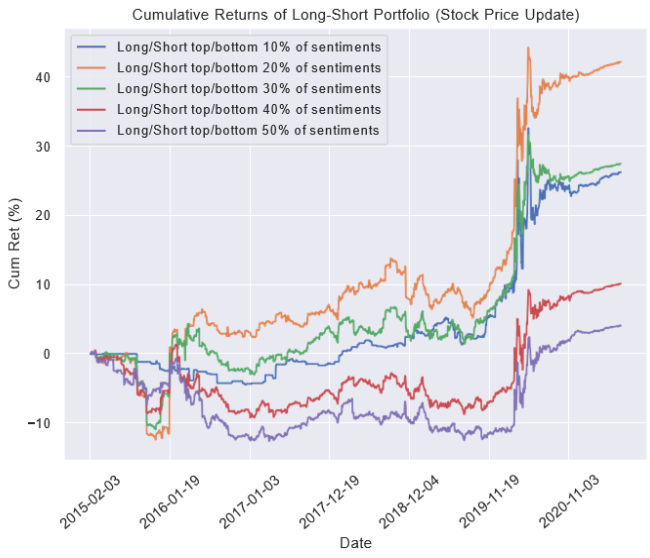

Fig. 1. Cumulative Returns of the Long-Short Portfolios using Stock Price Update news

it requires some smart pre-processing of the data to make sure it only keeps the important information, but apart from that it is very straightforward, fast and accurate. A first issue that should be mentioned concerns the errors in classification. In Fig. 2 it can be seen, for every single category, the percentage of articles that go into each category. For example, the articles of Stock Price Update are classified correctly into Stock Price Update in 97% of the instances, as we saw in the accuracy metrics, then 2% are wrongly classified in the Financials Category, and the remaining 1% is spread between the other categories and doesn't appear because of decimal points, but the column sums to 1.

There are some interesting findings: 19% of articles belonging to the Analyst recommendation category are wrongly classified into the Financials category. This is due to the fact that analyst recommendations are very often based on the financials of the companies so very often the first characters of the analyst recommendation articles will also contain financial information on the companies. Also, 24% of the articles in the employment category are wrongly classified into the Financials category. The last 2 or 3 categories have big values outside the correct category because there are not many articles in the labelled dataset, so the few that are incorrectly classified represent a big percentage of the total articles in that category.

To solve the problem of one category being systematically confused with another, what has to be done is change the defining articles for the big classification matrix in the classification model. The articles defining each category have to be MECE, so that there is minimal or no overlap between categories and all the articles still belong to one category. Furthermore, we have seen that the model is very good at classifying into different categories, but some categories end up with a much lower number of articles in them. The consequences are seen in the portfolios built using the news in these categories: a lot of days the portfolios remain flat simply because there are not enough sentiment data points to make an investment decision: let's say we are in the top 10 portfolio and we have 6 sentiment data points, then we cannot execute trades in the top and bottom 10, because the 10% of 6 is 0.6, so no trades are executed, as we can see in the portfolios built using Facilities news in Fig. 3. Also, on the days that we do have articles they are usually in the tens, not hundreds, so we execute trades for a very few numbers of companies in the 10% and 20% portfolio, we are exposed to the fluctuation of those bonds exclusively, so the portfolios are more volatile and riskier.

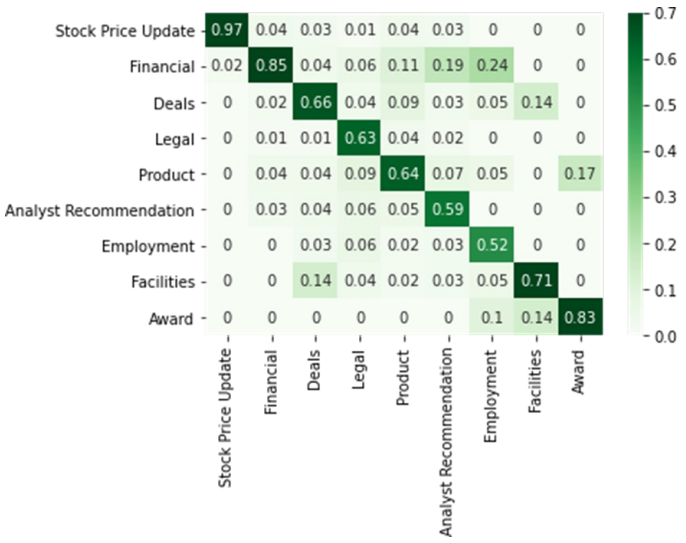

Fig. 2. Error matrix of the classification model

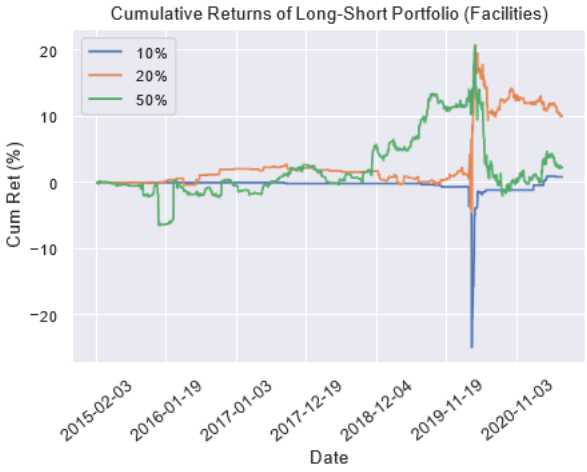

Fig. 3. Cumulative Returns of the Long-Short Portfolio using Facilities news

The easiest solution is to have more news on those categories, by scrapping from more web sources and creating more sentiment data points for companies. Another solution is to merge some of these categories. However, such a merging seems rather unnatural, because, for example, Facilities and Awards have nothing to do with each other. Moreover in the classification model it would become difficult to identify which are the most common words in these articles.

REFERENCES

[1] Axis Technical Group. “History of Document Classification”. https://axistechnical.com/document-classification-artificial-intelligence/

[2] A. Galántai, ”Rank reduction: theory and applications.” Advances in Mathematics Research 3 (2003): 49.

[3] M. Lamba, and M. Madhusudhan, “Text Mining for Information Professionals”, Springer. https://doi.org/10.1007/978-3-030-85085-2, 2022.

[4] J. Boudiukh, R. Feldman, S. Kogan, M. Richardson, “Which News Moves the Stock Prices? A Textual Analysis”, No. w18725. National Bureau of Economic Research, 2013.

[5] R. Feldman, R. Benjamin, B. H. Roy, and M. Fresko, “The stock sonar—sentiment analysis of stocks based on a hybrid approach”, Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 25. No. 2. 2011.

[6] HTMW team. ”Stock Sonar”. https://education.howthemarketworks.com/stock-sonar/

[7] F. Li, “The Determinants and Information Content of the Forward-looking State- ments in Corporate Filings - A Naive Bayesian Machine Learning Approach”, Journal of Accounting Research 48.5,pp. 1049-1102.

[8] E. Fontana, “BOW + TF-IDF in Python for unsupervised learning task”. medium.com. https://medium.com/betacom/bow-tf-idf-in-python-for-unsupervised-learning-task-88f3b63ccd6d

[9] Wikipedia Movie Plots Dataset. Kaggle, 2018. https://www.kaggle.com/datasets/jrobischon/wikipedia-movie-plots

[10] R. K. Jain, “Natural Language Based Recommender Systems for Financial Modelling”, June 2021.

[11] W. Abdulazeez Qader, “An Overview of Bag of Words;Importance, Implementa- tion, Applications, and Challenges”, Conference: 2019 International Engineering Conference (IEC)At: Erbil, Iraq

[12] Dow Jones. “Dow Jones Landing Page”. https://www.dowjones.com/

[13] B. Rosario, “Latent Semantic Indexing: An overview”, https://www.cse.msu.edu/ cse960/Papers/LSI/LSI.pdf

[14] E. Fontana. ”Latent Semantic Indexing in Python”. medium.com. https://medium.com/betacom/latent-semantic-indexing-in-python-85880414b4de

[15] J. Boudiukh, R. Feldman, S. Kogan, and M. Richardson, “Which News Moves the Stock Prices? A Textual Analysis”, (No. w18725). National Bureau of Economic Research, 2013.

[16] Oxford Dictionnary, “Sentiment Analysis Definition”. https://languages.oup.com/

[17] M. Lamba, and M. Madhusudhan, “Text Mining for Information Professionals”, Springer, 2022. https://doi.org/10.1007/978-3-030-85085-2.

[18] Marketwatch. ”Marketwatch landing page”. https://www.marketwatch.com/

[19] The motley fool. ”The Motley Fool landing page”. https://www.fool.com/

[20] Reuters. ”Reuters landing page”. https://www.reuters.com/

[21] Three M. ”3M website”. https://www.3m.com/

[22] Apple. ”Apple website”. https://www.apple.com/

[23] BP. ”BP website”. https://www.bp.com/es/es/spain/home.html

[24] Citigroup. ”Citigroup website”. www.citigroup.com/citi/

[25] Stryker. ”Stryker website”. www.stryker.com

[26] T. Hofmann. (2017). Probabilistic Latent Semantic Indexing. Available: dl.acm.org/doi/pdf/10.1145/3130348.3130370