# Sentiment Analysis in the Financial Phrasebank Dataset, an Approach to FinBERT

**Antonio Jose Lopez Roldan**
University Id: 230016805
MSc Data Science
antonio-jose.lopez-roldan@city.ac.uk

## 1 Problem statement and motivation

Overall, economies are governed by people's behaviour, their culture. In the same line, the financial markets are influenced by people's decisions. In this context, digital media has become a catalyst of those decisions speeding up the effect that worldwide events could have on the financial markets. Agile sentiment analysis could play a crucial role in making profitable decisions.

This coursework aims to understand and predict market behaviour. The importance of sentiment analysis in financial markets cannot be overstated. Sentiment analysis has the potential power to decode the vast amount of data generated by news, reports, social media text and so on turning it into actionable insights. Several pieces of literature underscore the importance of this field. For instance, "Time Frame Analysis for Sentiment Prediction of Stock Based on Financial News using Natural Language Processing" emphasizes the direct impact of news on stock prices, suggesting that understanding the dynamics can lead to more accurate predictions and profitable outcomes.

Similarly, "Entropy-Based Measure Sentiment Analysis in the Financial Market" introduces a relatively novel approach to quantifying sentiment and its effect on market ecosystems, providing new perspectives on the power of sentiment analysis. The study showcases how sentiment analysis could be pivotal in uncovering market trends and guiding investment decisions.

## 2 Research hypothesis

The main focus of this paper is to explore sentiment analysis within the financial markets, and compare the performance of traditional sentiment analysis models, such as Support Vector Machine and Naive Bayes, against the transformer-based model, FinBERT.

From the papers I read, I conclude that FinBERT will out perform accurately the more traditional sentiment analysis models. The main reasons for my statement are the following:

a) Transformers models can understand the context whereas traditional models base their predictions on frequency counts of words.

b) In addition, FinBERT is pre-trained on financial text what is an extra point in favour as more traditional models start their analysis from scratch.

c) Besides that, traditional models often rely on the right feature engineering to achieve the best results this requires certain experience to do the best of those models, on the other hand, Transformers models automatically learn to identify suitable features during the training.

## 3 Related work and background

My previous statements as well as the core purpose of this paper has been carefully studied beforehand and the points previously noted are supported by one of the most reputable research papers in the field (Araci, 2019). Specialized financial sentiment lexicons like (Loughran and McDonald, 2011) might appear to be an effective approach because they integrate established financial expertise into text analysis. Nonetheless, these lexicons rely on "word counting" techniques, their ability to decipher the meaning of a given context.

In another section of the same book, it is supported the discussed point "b)", were they propose that pre-trained language models could be more effective in addressing the issue of lacking enough labelled data when analysing

financial text. Even more, the model is initially pre-trained using a substantial corpus relevant to the intended domain, being more specific, they enhance a BERT language model through additional training on a financial corpus(Araci, 2019).

Finally, the same paper points out that traditional machine learning methods, which rely on extracting features from text, often fail to capture the semantic meaning derived from specific word sequences, in contrast, FinBERT is criticized for being overly dependent on large data volumes due to their complex parameter structures which potentially drive to high demands of considerable time and computation resources even in small datasets (Araci, 2019). Although this paper encompasses the FinBERT model itself, it doesn't provide any comparison of the model performance against any traditional machine learning sentiment analysis. To find the best configuration for the traditional models that could have a chance to perform better against the model based on transformers I evaluated a few papers such as (Agarwal, 2020) from where I concluded that TF-IDF could be one of the best feature extraction method.

Initially, I considered VADER, however, according to this other paper (Picasso et al., 2019) it wasn't the best of the choices. Besides, I looked for a versatile feature extraction that allows me to leverage several models based on the same technique to also explore how the feature extraction affected differently to the different traditional models.

In addition, following the positive outcomes from the studies of (Jaca-Madariaga et al., 2022) I tried unsuccessfully to implement Word2Vec.

To finish this section I would like to share that the decision for selecting the mentioned traditional machine learning models, Support Vector Machine, Multilayer Perceptron, Naive Bayes is based on the results from papers such as (Patel et al., 2015a) and (Hájek and Olej, 2015).

### 3.1 Accomplishments

1. Task 1: Getting dataset from the Hugging face - Completed.

2. Task 2: Preprocess dataset - Partially completed; Punctuation not applied. Turned the dataset into a massive dataset with a hight computational cost impossible to manage.

3. Task 3: Statistics and text analysis - Completed.

4. Task 4: Splitting data and feature extraction - Partially completed; Word2Vec couldn't be applied.

5. Task 5: Setting up baseline models - Partially completed; Multilayer Perceptron outputs were odds, this model was discarded during the model selection.

6. Task 6: Experimentation, fine-tuning, model selection - Completed.

7. Task 7: Gathering results for the final comparison - Completed.

## 4 Approach and Methodology

To build the basic architecture of the models that I used as a baseline for a later model selection I took as a reference other pieces of work. Among these references worth mentioning the following; for Multilayer Perceptron I used was (Patel et al., 2015b), for Support Vector Machines read (Hsu et al., 2003), for Multinomial Naive Bayes (Tama et al., 2019) and for the FinBERT model the already mentioned (Araci, 2019), then I applied standard configurations for the hyperparameter to narrow the best configurations towards their optimum results. This approach is reflected in the table I created during my model selection (see Figures 5, 6, 7). The following pipeline was the generalized approach for all the models.

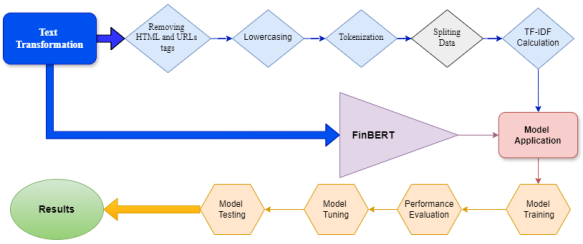


Figure 1: PipeLine

For the traditional models, the approach was the same as we can see in the pipeline above, Figure 1. Applied text preprocessing, and continued with feature extraction to convert the preprocessed

"text data" into numerical features using the Term Frequency-Inverse Document Frequency (TF-IDF). Then split the data, trained the models and performed hyperparameter tuning followed by model evaluation.

Overall, the mentioned steps represent the key components that altogether form a complete sentiment analysis pipeline.

From my readings and understanding, when I was building the models I had expected that Multilayer Percetron would be the model that might have a chance against FinBERT, however, during the hyperparameter tunning the outcomes for the results were quite imbalanced (the dataset is imbalanced, indeed). During the examination of the confusion matrix, every time a whole column was showing "0". I discarded the analysis of this model in the end, although I am attaching it within the code. For the other two traditional machine learning models, Support Vector Machine and Multinomial Naive Bayes, I was limited by my programming skills and time available. I am aware that I could have built a more sophisticated grid search, automatizing the search for the best of the possible models driving to higher accuracy scores. Even though, I managed to improve the baseline models I used. For example, I added a logarithmic scale in the search making sure that both, small and large values are included in the grid search this is important because machine learning models can be sensitive to some range of parameters.

Regarding to FinBERT model, I didn't have any expectations but to perform better than the other models according to what I read. It is a pretty straightforward model to implement, however, I spent some time understanding the different hyperparameters and best configurations for my purpose for what this resource (VincentCodesFinance, 2024) was very helpful.

In summary, to the baseline models used I applied standard configurations, just enough to run the model, from that starting point added extra characteristics to improve the results. As parameters, used what are accepted as standard in the field, when this wasn't available I used Fibonacci sequences as values.

During the implementation of the pipeline, as mentioned I Struggled with the application of Word2Vec. After spending a couple of days looking for a solution I gave up since I couldn't solve errors such as "Creating a tensor from a list of numpy.ndarrays is extremely slow" and other kinds of messages related to the input array, unsuccessfully tried to break down and isolate the issue looking for the root cause, I decided to use only TF-IDF in the end. Using other extraction features would have enriched and broadened the insights of this work.

The libraries used are the following:

- Pandas: To load and manipulate the dataset.
- Numpy: Essential in our work to deal with arrays and vectors from token transformations.
- Scikit-learn: Used for splitting the data, provided the Support Vector Machine model and the Multinomial Naive Bayes, GridsearchCV for hyperparameter tunning, etc.
- Natural language Toolkit (NLTK): Overall is a library for natural language processing tasks, e.g. text preprocessing, tokenization, stop word removal.
- BeautifulSoup is a library for web scraping and parsing HTML/XML doc, used in my code to remove HTML tags from the text.
- Matplotlib and Seaborn: Are libraries for data visualization used for the display of the confusion matrix.
- PyTorch: Deep learning library used for training and building neural networks, used in my code for the construction of the Multilayer Perceptron model as well as to convert the data into tensors, needed to get the inputs for the Support Vector Machine model.

Regarding the external resources used for the code development of this piece of work, I leveraged many resources. I am mentioning some of them and how they helped me, the rest are listed under each section of the code where I used that external knowledge. Said so, here are some of the most relevant; (SEHGAL, 2021) used when creating the basic statistics report, (PyTorch, 2023) to understand how to create the input data for the

models and finally the application of a practical early stopping was possible to (StackOverFlow, 2022).

Overall, the models used in this paper are like a puzzle, made from different pieces, from different sources and tailored to the problem we are working on, and the origin of those pieces are well listed under each section code as mentioned earlier.

## 5 Dataset

The dataset used for the study is the "financial phrasebank" from (Malo et al., 2014).

Before starting to work on the application of any model, understanding the dataset and its context we are working on is crucial for the achievement of our goals. Not doing so involves many risks that would compromise the progress of our work as well as the reliability of the results. Next, I am explaining some reasons to examine our dataset carefully:

- Context: We need to be aware of the words present in our environment and their context. Debt, for instance, in a "regular" text could have a negative connotation nevertheless in a financial context could have a positive/neutral connotation.
- In our particular case, the context of the financial sentiment analysis depends on the economic cycles. These economic cycles create a dynamic environment and in this highly dynamic environment, the financial sentiment can be importantly influenced by current events that are linked to a specific period.
- Class imbalance, one of the worst enemies of sentiment analysis: examining whether or not we have the same number of labels for the different classes is critical to, for example, understand the output, if not, taking actions would possibly be needed to find an approximate balanced dataset among all the possible classes to help our models do their best. At this point, worth mentioning that our Multilayer Perceptron model was discarded due to this fact. Even though, the goal of this work is to compare the considered traditional sentiment analysis against the transformer model FinBERT even though the circumstances are not ideal for the analysis.

Below in Figure 2 basic statistics are displayed, as aforementioned, our dataset has class imbalance and the classes have; 2879 labels for Neutral, 1363 for Positive, and 604 for Negative:

```
Sentiment distribution: 2-Positive, 1-Neutral, 0-Negative,
1    2879
2    1363
0     604
Name: label, dtype: int64
```

Figure 2: Imbalanced Data

Next, Figure 3 displays the basic statistics from our dataset:

```
Statistics by label: 0-Negative, 1-Neutral,  2-Positive


       min  max       mean        var       std
label
0        2   34  13.877483  37.746159  6.143790
1        0   46  12.516151  37.739055  6.143212
2        2   35  14.318415  40.969022  6.400705
```

Figure 3: Statistics By Label

Interpretation of results:

- Mean. As we see, the mean values are relatively close to each other, with positive labels having a slightly higher mean compared to the neutral and negative classes.
- Variance and Standard Deviation. These two metrics are also relatively similar indicating a similar grade of dispersion in the labels within each class.

In addition, the total number of unique words is 11105.

.

From Figure 4, we can appreciate which are the most repeated words among the different classes,

- Positive: Company, Eur, said, Finland, Finnish...
- Neutral: Eur, "mn", profit, net, company, Finnish...
- Negative: Eur, "mn", company, said, Finnish...

"**mn**" = Millions

Earlier, we underlined that the meaning or relevance of certain words relies heavily on the

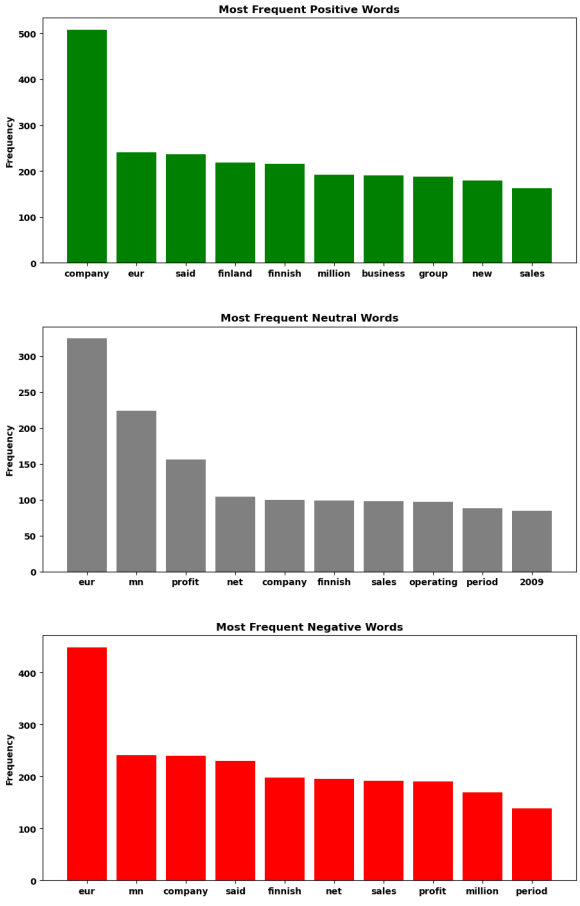


Figure 4: Most Frequent Words For Each Class

context, well, here is the proof that supports that previous statement; among the the most frequent words, appear the people of a country, Finnish within the three bar chart from Figure 4 and their country, Finland, within the most repeated words within the positive class. This makes sense since the corpus used in this dataset is made out of English news on all listed companies in OMX Helsinki. In contrast, if the "Finland"/"Finnish" words were among the most mentioned in a dataset related to the United Kingdom economic newspapers, the importance of these words would be different, that is why context is so important in text analysis.

### 5.1 Dataset preprocessing

Initially intended to use Word2Vec together to TF-IDF. Word2Vec is a technique that turns words into numerical representations. These representations capture the meaning and relationships between words, considering the context where they appear. TF-IDF, which stands for Term Frequency-Inverse Document Frequency, assigns words a certain level of relevance considering how often appear.

When tried to implement Word2Vec faced difficulties. Although I could extract the features, the problem laid on when the models tried to take those as input. I couldn't figure out how to solve it.

The data pre-processing applied was already mentioned and essential to progress on our analysis. By removing the HTML tags and URLs, stop words (common meaningless words), lowercasing words, we cleared out a lot of noise that would interfere in our analysis otherwise. With tokenization standardised text. The main difficulty associated with preprocessing is finding the right balance between removing noise and preserving meaningful information. Removing too much information can lead to loss of context, while keeping irrelevant information can distort the performance of our models.

## 6 Baselines

The baseline models used in this analysis were the same as the final models but with a very basic configuration, with only standard parameters and simple lines of code enough to produce the first results.

This approach saved me time and allowed me to understand since the beginning each part of my code. I continued adding extra parts until reaching a point where any extra improvement happened.

Then I conducted a systematic hyperparameter tuning, testing at least two "opposite" configurations for each model, then I narrowed the figures towards those that were producing the best results recorded in the tables from Figures 5, 6, and 7.

## 7 Results, error analysis

The values from the basic statistics, Figures 2 and 3, might have an impact on our models. SVM is a discriminative classifier that aims to find the optimal hyperplane to separate the classes in the space, hence the similarity in the mean of the classes may not help in the performance. Continuing with the statistics, Multinomial Naive Bayes is a probabilistic classifier that assumes that the features, in these case tokens, are independently

distributed. Therefore, since the mean values are so close to each other across the sentiment classes, the distribution may not help on the classification.

On the other hand, FinBERT, in comparison with the traditional models, would be affected differently by the characteristics of the dataset as is a pre-trained model on a large corpus of financial text. Since it is a pre-trained model in a large corpus, has already learned the contextual meaning of words hence the performance of FinBERT would depend mostly on the fine-tuning process and the specific architecture and hyperparameters used.

**Support Vector Machine**

| Kernel function | Cross Validation | GridSearch Space for "C" | Box Constant | Test accuracy | Computational Cost (Time;Min) |
|---|---|---|---|---|---|
| Baseline | Baseline Model | Baseline Model | Baseline | 72 | 2 |
| Linear | 3 | np.logspace(0.1, 5, 3) | C=1.809255 | 74 | 15 |
| Linear | 3 | np.logspace(1, 20, 3) | C=10.0 | 71 | 13 |
| Linear | 3 | np.logspace(1.5, 2.75,2) | C=133.3521 | 74 | 9 |
| RBF | 3 | np.logspace(0.1, 5, 3) | C=354.8133 | 74 | 15 |
| RBF | 3 | np.logspace(1, 20, 3) | C=10.0 | 74 | 12 |
| RBF | 3 | np.logspace(-15,15,3) | C=10000000 | 70 | 8 |
| | | | Max | 74 | 15 |
| | | | Min | 70 | 2 |
| | | | Avg | 73 | 10.57143 |

Figure 5: Support Vector Machine

**Multinomial Naïve Bayes**

| alpha | alpha2 | fit_prior | class_prior | Cross Validation | Test accuracy | Computational Cost (Time;Min) |
|---|---|---|---|---|---|---|
| Baseline Model | Baseline Model | Baseline | Baseline Model | Baseline | 67 | <1 |
| np.logspace(0.01, 1, 5) | 1.023292992 | [True, False] | | 3 | 68 | <1 |
| np.logspace(0.01, 1, 5) | 1.023292992 | [True,] | | 3 | 67 | <1 |
| np.logspace(0.01, 1, 5) | 1.80925591 | [False] | | 3 | 69 | <1 |
| np.logspace(0.01, 1, 5) | 1.023292992 | [False] | [None, [0.01, 6, 2]] | 3 | 71 | <1 |
| np.logspace(1, 5, 5] | 1.80925591 | [False] | [None, [0.01, 0.1, 0.6, 1]] | 3 | 67 | <1 |
| np.logspace(10, 15, 5] | 10000000000 | [False] | [None, [0.01, 0.1, 0.6, 1]] | 3 | 67 | <1 |
| | | | | Max | 71 | 1 |
| | | | | Min | 67 | 1 |
| | | | | Avg | 68 | 1 |

Figure 6: Multinomial Naive Bayes

**FinBERT**

| learning rate | batch size | epochs | model architecture | max_length | Test accuracy | Computational Cost (Time;Min) |
|---|---|---|---|---|---|---|
| 1 | 10 | 1 | Bert | 5 | 64 | 4 |
| 0.00001 | 5 | 5 | Bert | 5 | 74 | 48 |
| 0.001 | 10 | 3 | Bert | 5 | 93 | 32 |
| 0.01 | 5 | 3 | Bert | 10 | 62 | 53 |
| 0.01 | 10 | 3 | Bert | 10 | 26 | 19 |
| 0.000001 | 15 | 4 | Bert | 30 | 72 | 30 |
| 0.000001 | 20 | 4 | Bert | 25 | 74 | 49 |
| | | | | Max | 93 | 53 |
| | | | | Min | 26 | 4 |
| | | | | Avg | 66 | 33.57143 |

Figure 7: FinBERT

The tables above display the records for the models used. We will examine them next:

1. Support Vector Machine: The best model achieved in the test set an accuracy of 72 per cent and the improved mode of 74 per cent. The computational cost for the baseline model is only 2 minutes whereas the best model with the lowest computational cost takes 9 minutes. The Kernel used in this model is Linear with a C=133.3521432163324

2. Multinomial Naive Bayes: The baseline model obtained an accuracy of 67 per cent, and the tuned model improved up to 71 per cent. In this model, all the computational cost is less than 1 minute.

3. FinBERT: The best model achieved a test accuracy of 93 per cent and it took 32 minutes to run in comparison with the baseline model that only took 4 minutes obtaining a test accuracy of 64 per cent.

Overall the tuned models demonstrate an improvement in the performance compared to their respective baselines. The model with major variability is shown by the transformer FinBERT whereas the most stable model in regards to performance is the SVM.

The robust performance of the SVM suggests that if we would like to see variability in the outcomes, we would need to change the preprocessing for the text data, specifically would be interesting to solve the error given by Word2Vec and use that feature extraction method. On the other hand, the gap between the maximum and minimum accuracy for the FinBERT model is indicative of how important it is to understand the configuration of the model, the purpose and relationship among them, and how it affects the results.

I had to deal with many coding errors. Below in Figure 8, I am showing one of them which shows there is a problem with "class prior" parameter. I am aware that it might seem something obvious but I didn't realise that the "class prior" parameter of MultinomialNB must be an array-like or None, yet got "balanced", from my search means that "class-prior was given a value that isn't acceptable. This is, The "class prior" parameter expects either:

- An array-like object which consists of the prior probabilities of the classes.

- Or "None", in which case the class priors will be estimated from the training data.

In the end, what happened is that I confused "class prior" with the "class weight" which can be set to "balanced" as value.


Figure 8: Multinomial Naive Bayes Error

Finally, we will analyse the case of the Multilayer Perceptron in Figure 9. We observe that there is not classification for the positive class. Considering the scenario in which the code is well written without any mistake/error, this output suggests the model didn't learn to classify positive sentiment in the text. As we earlier observed, data is imbalanced and this could be the reason for this inaccurate performance.

It could be due to not having enough data for that class but is the negative class with less labels while the positive class has an intermediate number of labels therefore this argument is not solid.

The model was tested in the IMDb movie reviews dataset(Maas et al., 2011) with the same preprocessing but with a perfect balance, 25k labels for the negative class 25k labels for the positive class. Performance was decent with around 83 per cent accuracy on the test data set, although as we indirectly mentioned, this data is a binary classification (positive/negative) whereas in this work we are performing a multi-classification. Below we can see the results for the confusion matrix.

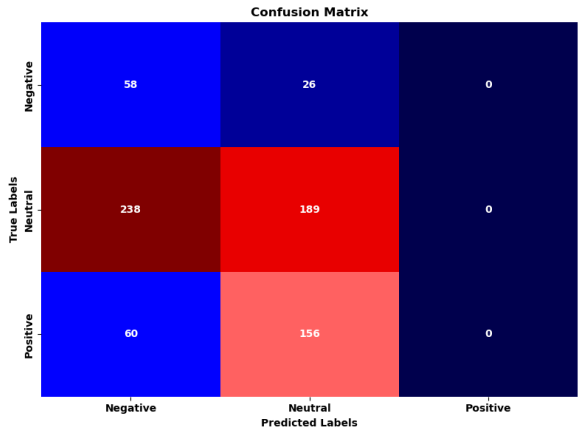


Figure 9: Enter Caption

## 8 Lessons learned and conclusions

Throughout this coursework, I consider that learned many lessons and made important observations. Just the process of applying different sentiment analysis models to the financial text data contributed to enriching my understanding of the whole ecosystem, even more, the experience with my first transformers has been without any hesitation a memorable milestone.

One of the key takeaways is the importance of data preprocessing and feature engineering and the impact on the model performance. Although I missed the opportunity to compare in depth how different feature extraction techniques would affect the different model's outputs.

Another valuable lesson learned is the significance of hyperparameter tuning. We can observe this particular point on the table for the results of the FinBERT model. The impact of the different configurations on the outcomes is critical producing a massive gap (Min Vs Max) between the best and the worst results of the model.

During the project, some challenges and dificulties were encountered. One of the most frustrated challenges was dealing with the effect of the imbalanced data on the Multilayer Perceptron model which I had to discard due to a lack of time for finding a reliable solution. There are techniques that we could have employed to diminish this such as adding/deleting extra labels until we get the same numbers of labels among all the classes.

Another relevant lesson is the impact of the context/economic cycle on the meaning of the words and the effect on the text classification as in the case we already treated, the "Finland" "Finnish" words.

From the results, it became evident that sentiment analysis can provide tangible results if information is addressed diligently. However, it is crucial to consider the limitations and potential biases of the models and to interpret the results with caution.

Looking back, in the traditional models I missed also the opportunity to apply n-grams. I tried to implement the code, however couldn't

manage. N-grams could have potentially helped the traditional models to capture context.

In conclusion:

- Regarding the impact of this experience into my learning journey, it has highlighted the importance of data quality, preprocessing, feature engineering and model selection in achieving the best of possible results.

- Regarding the results obtained from this study, we assert that FinBERT performs better than the traditional models considered in this study, Support Vector Machine, Multinomial Naive Bayes.

## References


Arul Agarwal. 2020. Sentiment analysis of financial news. In *2020 12th International Conference on Computational Intelligence and Communication Networks (CICN)*, pages 312–315. IEEE.

Dogu Araci. 2019. Finbert: Financial sentiment analysis with pre-trained language models. *CoRR*, abs/1908.10063.

Petr Hájek and Vladimír Olej. 2015. Word categorization of corporate annual reports for bankruptcy prediction by machine learning methods. In *Text, Speech, and Dialogue: 18th International Conference, TSD 2015, Pilsen, Czech Republic, September 14-17, 2015, Proceedings 18*, pages 122–130. Springer.

Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. 2003. A practical guide to support vector classification. 2003.

M Jaca-Madariaga, E Zarrabeitia-Bilbao, RM Rio-Belver, and MF Moens. 2022. Sentiment analysis model using word2vec, bi-lstm and attention mechanism. In *The International Conference on Industrial Engineering and Industrial Management*, pages 239–244. Springer.

Tim Loughran and Bill McDonald. 2011. When is a liability not a liability? textual analysis, dictionaries, and 10-ks. *The Journal of finance*, 66(1):35–65.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

P. Malo, A. Sinha, P. Korhonen, J. Wallenius, and P. Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65.

Jigar Patel, Sahil Shah, Priyank Thakkar, and Ketan Kotecha. 2015a. Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert systems with applications*, 42(1):259–268.

Jigar Patel, Sahil R. Shah, Priyank Thakkar, and Ketan Kotecha. 2015b. Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert Syst. Appl.*, 42:259–268.

Andrea Picasso, Simone Merello, Yukun Ma, Luca Oneto, and Erik Cambria. 2019. Technical analysis and sentiment embeddings for market trend prediction. *Expert Systems with Applications*, 135:60–70.

PyTorch. 2023. "torch.tensor".

AKSHAY SEHGAL. 2021. Ultimate guide to pandas groupby and aggregate.

StackOverFlow. 2022. "early stopping in pytorch".

VO Tama, Y Sibaroni, et al. 2019. Labeling analysis in the classification of product review sentiments by using multinomial naive bayes algorithm. In *Journal of Physics: Conference Series*, volume 1192, page 012036. IOP Publishing.

VincentCodesFinance. 2024. Sentiment analysis of financial news in python - 3 ways using dictionary, finbert and llms.