COMP6713

Stock Whisperers

Neeraj Mirashi (z5316619)

Priyal Jain (z5512958)

Mark Wang (z5429626)

Reynah Hsu (z5245032)

Table of Contents

1. Introduction

In today's fast paced financial markets, investors are presented with a deluge of information, which they need to sift through to extrapolate meaning. This information overload can not only obscure valuable information, but it can also be highly inefficient. The complication of accurately and swiftly gauging market sentiments is crucial to an ever-competitive market environment. The challenge for many investors today is not only analysing and processing substantial amounts of data but doing so in a way that is efficient and extracting meaningful insights. Recognising this challenge, our team has developed a solution inspired by the transformative capabilities of machine learning models which exist today. The core of our experimental analysis employs statistical models and machine learning algorithms including TF-IDF and SVM, followed by advanced techniques such as BERT and FinBERT for fine-tuning. This report presents a thorough analysis of the models used, their reasoning and their efficacy in helping solve the aforementioned problem.

2. Experiment Setup: Dataset Collection

2.1 Financial PhraseBank Dataset (Hugging Face)

For our initial dataset, we opted for the Financial PhraseBank provided by Hugging Face. This choice was primarily driven by its accessibility and the richness of financial text data it offered.

2.2 Custom Dataset Creation

To tailor our analysis tool to the nuances of stock analysis, we embarked on creating a bespoke dataset.

We initiated the dataset creation process by integrating NewsAPI to gather news articles' URLs related to ASX companies. However, we faced constraints due to the free tier's limitation of 100 API requests per account and the requirement that articles be within the last 30 days. Despite employing generic word lists to filter out irrelevant URLs, some unrelated articles slipped through the cracks. This was particularly problematic for stocks with lower search volumes, leading to the inclusion of non-relevant articles in the dataset. In our ideal stock analysis tool, we would continuously hit the NewsAPI to get the newest information as it comes out.

Leveraging Selenium, we navigated through the collected URLs to extract their content. This process proved challenging due to the diverse structural layouts of websites. Additionally, the presence of popup ads posed a significant obstacle, as their inadvertent inclusion could skew sentiment analysis results. Scraping content from a large number of URLs proved to be a time-intensive process, taking approximately 900 minutes to complete. Manual cleanup of the dataset was impractical due to its sheer size.

Using Longformer for content summarization introduced its own set of challenges. Some summaries lacked relevance to the corresponding stock due to articles discussing multiple stocks simultaneously, undermining the dataset's integrity. Dataset labelling was performed using BERT. While GPT-3.5 Turbo would have been preferable due to its capabilities in labelling and generating summaries, cost constraints prevented its adoption.

While the creation of our custom dataset posed notable challenges, it also paved the way for valuable insights into the intricacies of stock analysis data acquisition. Despite facing hurdles such as limited API access, diverse website structures, and the presence of irrelevant content, the dataset holds immense potential with the right refinements.

With further cleanup efforts and refinement strategies, the dataset stands poised to become an invaluable asset for our stock analysis tool. The challenges encountered serve as learning opportunities, guiding us toward more effective data gathering and preprocessing methodologies in the future.

Through further innovation, we are confident that the dataset will soon achieve its full potential, providing a robust foundation for comprehensive and insightful stock analysis.

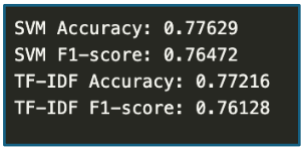
3. Statistical models

3.1 TF-IDF

Term Frequency-Inverse Document Frequency (TF-IDF) is a statistical measure used to evaluate the significance of a word within a document in relation to in relation to its frequency across various documents. The word that occurs frequently in a single document is considered important, so it will be given a high score. However, if a word appears across many documents, it is considered less distinctive and therefore receive lower score. Common terms like “the” that have higher frequency in many documents will be scaled down.

We leverage TF-IDF for extracting information of stock news summaries as it helps in focusing the analysis on the features most relevant to making sense of the sentiment expressed in financial news articles. It can further be applied to linear classifier potentially enhancing the efficiency of prediction in stock news context.

In our project, the accuracy of TF-IDF reaches 0.77 and F1-score achieves 0.76. This is a baseline model and serves as a foundation for further improvements. Our next steps involve integrating more natural language processing method and testing advanced machine learning algorithms. capture the intricacies of market sentiment and ultimately aid in more informed decision-making processes.



3.2 SVM

The SVM (Support Vector Machine) model for Financial News Sentiment Analysis is designed to categorize financial news articles into positive, negative, or neutral sentiments based on their textual content. Utilizing a linear kernel, this model processes inputs through TF-IDF (Term Frequency-Inverse Document Frequency) vectorization, which transforms text into numerical data reflecting word importance within the corpus. This setup allows the SVM to construct a hyperplane in high-dimensional space, optimally classifying the sentiment of financial news.

During training, the model learns from a dataset that merges generic financial phrases from Financial PhraseBank and the dataset was collected using NewsAPI to gather news articles related to ASX companies, with the process involving data preparation (handling missing values and vectorizing text), training on the TF-IDF transformed data, and evaluating model performance using accuracy and F1-score metrics on a validation set. The SVM model is noted for its simplicity and effectiveness in high-dimensional spaces, making it particularly suitable for linear classification tasks.

However, it faces limitations in scalability and dependency on the choice of the kernel, which might hinder performance with large datasets or complex nonlinear relationships. The SVM achieved an accuracy of 0.77629 and an F1-score of 0.76472. We anticipate even better performance in subsequent analyses using BERT and FinBERT models, which are designed to handle larger datasets and more complex relationships more effectively.

```
SVM Accuracy: 0.77629
SVM F1-score: 0.76472
```

4. Fine Tuned Models

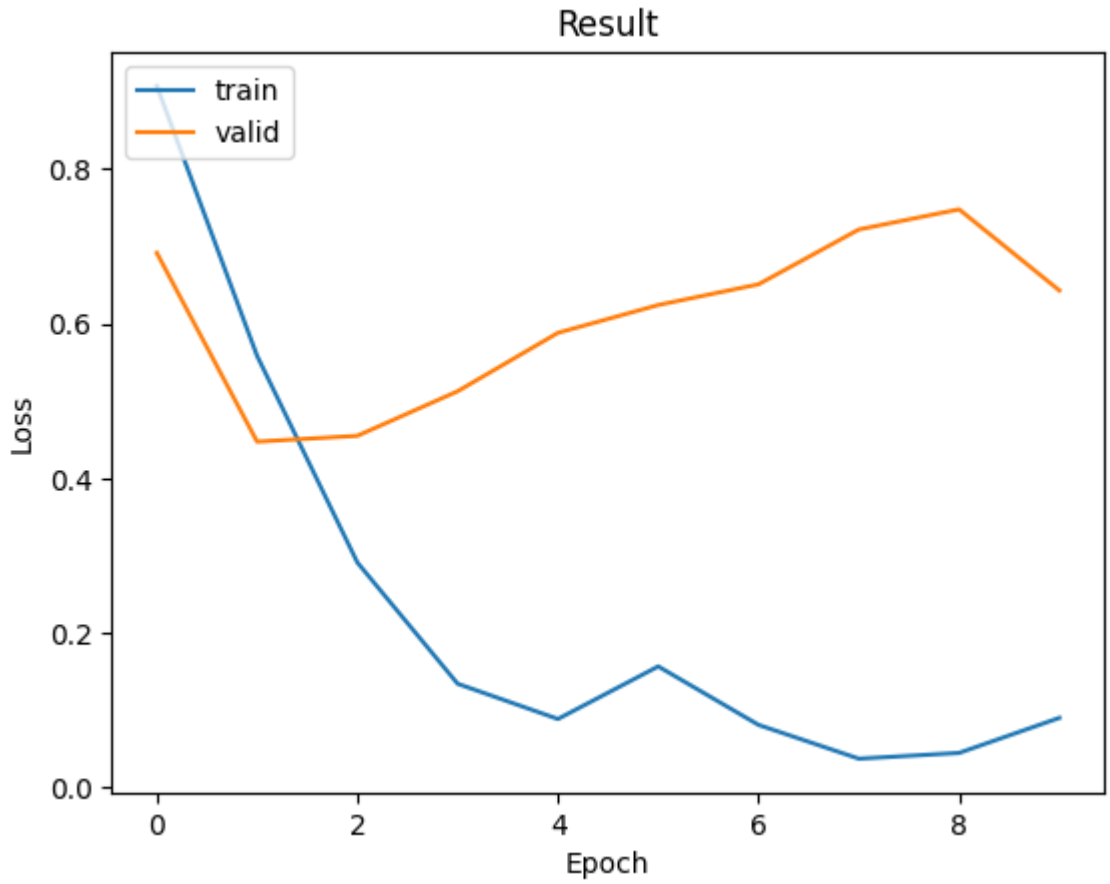
4.1 Extended BERT model:

Based on the previous models, we have further implemented the BERT model, which utilizes a unique attention mechanism through masked language model (MLM) training. Compared to the TF-IDF approach, the BERT model can conduct a more comprehensive analysis due to its effective weighting and embedding characteristics. We apply fine-tuning methods to better adapt the model to our specific dataset.

Initially, we use a dataset provided by Financial PhraseBank for fine-tuning, splitting it into an 80% training set and a 20% validation set. Although the experimental results yielded approximately 80% accuracy on the validation set, we observed that the model did not perform as well with actual financial text data. This discrepancy highlights a domain shift problem between the training dataset and the real-world financial dataset. As a result, we decided to manually augment the dataset to optimize model performance.

To collect more appropriate textual data in the financial field, we compiled 3,000 news articles and converted them into summaries. These newly labelled data were added to the training set to enhance the model's generalization capabilities. Later, we applied Warmup and Cosine Annealing to dynamically modify the learning rate, enabling the model to better converge to the optimal solution. We successfully increased the accuracy on the validation set to 83%, significantly enhancing the model's capability to process actual financial-related sentences.

```
              0       0.69      0.84      0.76       110
              1       0.89      0.85      0.87       571
              2       0.79      0.79      0.79       289

    accuracy                            0.83       970
   macro avg       0.79      0.83      0.81       970
weighted avg       0.84      0.83      0.83       970

Train Loss: 0.08992 Train Acc: 0.97285| Valid Loss: 0.64284 Valid Acc: 0.83196| Best Acc: 0.83814
```



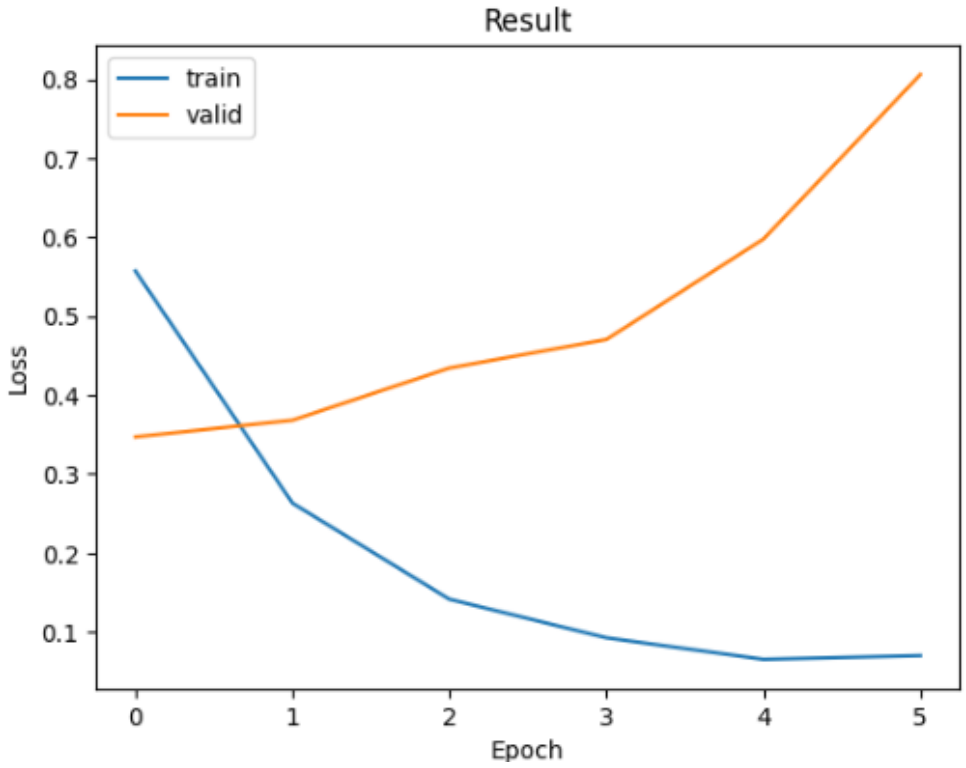
4.2 FinBert

The FinBERT model, specifically tailored for financial sentiment analysis, utilizes the advanced BERT architecture pre-trained on financial texts to enhance predictive accuracy. This model processes a combined dataset, through TF-IDF vectorization before being fine-tuned across multiple epochs. The workflow begins with seed initialization for consistency, involves data merging, and addresses missing values before splitting into training and validation subsets.

FinBERT undergoes fine-tuning where textual data is vectorized via TF-IDF before training through PyTorch's computational frameworks. The fine-tuning adjusts the model's weights specifically for financial sentiment, refining its understanding of sector-specific lexicon and context. Each epoch's loss and accuracy metrics reflect the model's learning trajectory, with performance validated against unseen data to ensure robust generalization. Notably, accuracy and F1-score metrics assess its applicability in enhancing algorithmic trading and investment strategies.

FinBERT's domain-specific pre-training and fine-tuning deliver high performance, distinguishing it from generic models. Despite its computational demands, strategic training with early stopping confirms its efficacy for real-time financial analytics. However, the need for substantial computational resources remains a notable limitation. Overall, FinBERT shows significant potential in transforming financial analytics with nuanced sentiment categorization and robust real-world performance.

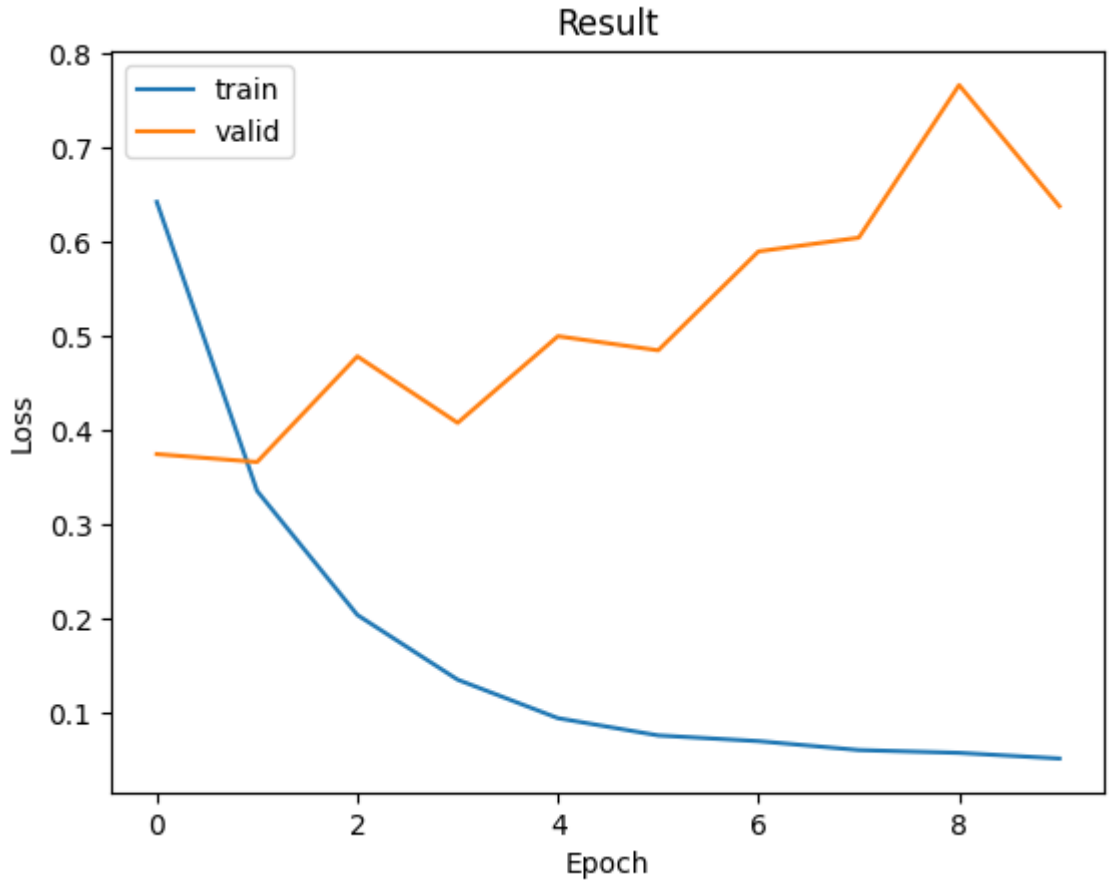
```
Train Loss: 0.06541 Train Acc: 0.97919| Valid Loss: 0.59757 Valid Acc: 0.85464| Best Acc: 0.87320

  0%|          | 0/385 [00:00<?, ?it/s]C:\Users\user\AppData\Local\Temp\ipykernel_4700\2777210971.py:6: DeprecationWarning: a
n integer is required (got type float).  Implicit conversion to integers using __int__ is deprecated, and may be removed in a
future version of Python.
  y = {'labels': torch.LongTensor(y)} # Convert labels into a tensor
Train Epoch 5: 100%|██████████| 385/385 [01:17<00:00,  4.95it/s, loss=0.015]
Valid Epoch 5: 100%|██████████| 61/61 [00:01<00:00, 43.00it/s, loss=0.929]
              precision    recall  f1-score   support

           0       0.87      0.72      0.79       110
           1       0.83      0.96      0.89       571
           2       0.89      0.66      0.76       289

    accuracy                           0.84       970
   macro avg       0.86      0.78      0.81       970
weighted avg       0.85      0.84      0.84       970
```



4.3 Fine-tuned Model (XLNet)

We used the Hugging Face financial phrasebank dataset to train another model- XLNetForSequenceClassification pre-trained model. Preprocessing involved tokenization of sentences using XLNet tokenizer. The model achieved an accuracy 87% after 10 epochs.
XLNet demonstrated strong performance in sentiment analysis on financial data.
Further improvements and fine-tuning on our own dataset could potentially enhance the model's accuracy.

```
Train Epoch 9: 100%|██████████| 243/243 [18:07<00:00,  4.48s/it, loss=0.069]
Valid Epoch 9: 100%|██████████| 61/61 [00:27<00:00,  2.24it/s, loss=0.573]
Train Loss: 0.05145 Train Acc: 0.98504| Valid Loss: 0.63741 Valid Acc: 0.86701| Best Acc: 0.87423
```



5. Comparisons and Conclusions

Feature	SVM & TF-IDF Model	Bert & FinBERT Model
Model Type	Traditional Machine Learning	Deep Learning, Transformer-based
Text Representation	Bag of words with TF-IDF scoring	Contextual embeddings from pre-trained language model
Training Time	Relatively fast on moderate-sized datasets	Can be extensive due to model complexity
Scalability	Good for small to medium datasets	Scales better with larger datasets
Interpretability	High (features and their weights are interpretable)	Low (complex model internals are hard to interpret)
Feature Engineering	Necessary (TF-IDF vectorization, n-grams, etc.)	Minimal (learns features from data)
Accuracy	Varies (often lower than deep learning models)	Generally high (benefits from contextual understanding)
F1-Score	Varies (balanced F1-scores require class-weighting)	Generally high (better at handling class imbalances)
Performance on Unseen Data	Moderate (depends on feature representation)	Generally robust (adapts well to new data)
Use Case	Baseline or when explainability is key	Complex tasks where state-of-the-art performance is needed
Real-world Application	Quick prototyping and simple classification tasks	Advanced applications like sentiment analysis and market prediction

This table serves as a guide to understand how SVM & TF-IDF model compares to a sophisticated model like Bert & FinBERT in the context of financial prediction. SVM & TF-IDF models are great for establishing a baseline and for scenarios where interpretability is a must, whereas Bert & FinBERT excels in scenarios requiring the highest performance, despite the increased computational cost and complexity.

6. Conclusion

In the fast-paced realm of financial markets, efficient extraction of meaningful insights from the deluge of available information is paramount. Our endeavour to address this challenge led us to develop a sophisticated stock analysis tool harnessing the transformative capabilities of machine learning models.

We began with the meticulous collection of datasets, including the readily available Financial PhraseBank from Hugging Face and a bespoke dataset tailored to the nuances of stock analysis. Despite encountering hurdles such as limited API access, diverse website structures, and the presence of irrelevant content, our custom dataset creation process provided valuable insights into data acquisition complexities.

Our experimental analysis delved into the efficacy of statistical models such as TF-IDF and SVM, laying the groundwork for subsequent advancements with BERT, FinBERT, and XLNet. These models, ranging from traditional machine learning to state-of-the-art deep learning architectures, showcased varying strengths and trade-offs in accuracy, interpretability, and computational demands.

In comparing these models, we highlighted their suitability for different use cases, emphasizing the importance of selecting the appropriate tool for the task at hand. While SVM & TF-IDF models offer interpretability and efficiency for baseline analyses, sophisticated models like BERT & FinBERT excel in complex tasks requiring state-of-the-art performance.

7. Project Scope Table

PART A: Problem Defini1on Minimum: 10 credits			**Your Project**
	Points	**Descrip1on**	
NLP (Natural Language Processing) Problem	5 credits per problem	Examples: Question answering, Sentiment Analysis, etc. If your project uses an NLP problem for another downstream task (say, stock market prediction), please count only the NLP problem.	Stock Market Prediction using sentiment analysis of news articles
Text Source/Domain	5 credits per text source/ domains	Select one among: News Articles, Research papers, social media posts, sentences, etc.	News Articles
PART B: Dataset Selection Minimum: 20 credits			
Use one existing dataset	10 credits per dataset	This refers to publicly available annotated datasets	Financial PhraseBank by Hugging Face
Create your own labelled dataset	20 credits	Factors to consider here are the selection of labels, inter annotator agreement	Created our own dataset – final-data-updated.xlsx
Use an existing lexicon (for evaluation)	10 credits	Examples: WordNet, medical ontology	
PART C: Modelling			
Implement a rule-based or statistical model as a baseline	-	Essential.	TFIDF and SVM
Minimum: 30 credits			
Use an existing pretrained/finetune d model	5 credits per model	You must compare the performance of models if you are only using fine-tuned models	XLNet
Fine-tune a model based on a dataset	20 credits per out-of-the-box fine-tuning method	Examples: fine-tune BERT (Bidirectional Encoder Representations from Transformers), prefix-tune LLaMA	BERT FinBert

Extend a finetuning method	30 credits per extended finetuning method	Example of extension: modification of the loss function	BERT
Integrate a language model with external tools	20 credits	Usage of a library like LangChain	
PART D: Evalua1on Minimum: 20 credits			
Quantitative Evaluation	10 credits	Relevant metrics such as Fscore, Precision@5	F-score Accuracy
Qualitative Evaluation	5 credits	Examine mis-classified instances and produce common error types	
Command line testing	5 credits	Interface to test out the system. This can be executed using an input argument or input file.	
Demo	10 credits	A simple demo through Gradio (or equivalent).	Implemented a demo using Gradio