MASTER OF TECHNOLOGY IN INTELLIGENT SYSTEMS:
Intelligent Reasoning Systems Project

FINANCIAL SENTIMENT RECOMMENDER

TEAM: IS06PT-GRP-SenTEAMent

Team Members Name:	NUS Student ID
Ng Ying Wee	A0111692L
Yeo Li Ying	A0292287R
Lai Kah Hoe	A0292131N
Tan Kok Tong	A0155101Y

Table of Content

1. Executive Summary

The financial stock market is very important in the modern day world. It helps countries, institutional and retail investors in capital formation. Therefore, the skills or tools to better understand the stock market are relevant and highly sought after.

Understanding human decision-making in stock investment is pivotal for maximising profits in the stock market. Sentiment analysis is one of many tools that an investor can apply in their trading strategy to attempt to gain foresight into future price movements (Deveikyte et al., 2022)[1]. While humans are incredibly good at distinguishing between positive and negative sentiment, it becomes time consuming to undertake such analysis over a big number of news data sources. As more news data sources are published, the effort to analyse the sentiments will only continue to increase and may result in missed opportunities. Hence, this project aims to solve this problem by making the most important news accessible and removing the need of sentiment analysis from the investors.

The team believes this project can provide investors valuable insights into the market sentiment trends, enabling more informed decision-making in their market investments/trades and potentially higher returns on investment.

The team has designed a full system architecture using a domain-specific knowledge discovery via big data mining techniques like API pull. The news data are sourced from traditional and social media platforms, coupled with historical stock price movements. The data then goes through further knowledge refinement to form the knowledge base. Then, the knowledge base is fed into the cognitive systems like natural language processing (NLP) models to gain insights. The insights are then fed for machine learning techniques with another set of knowledge base to produce an output of financial sentiment and estimated price movement for a particular stock requested by the user. Due to the wide range of natural language models and machine learning techniques available, performance testing was carried out to give the highest performance score for a combination of natural language model and machine learning technique. Through the performance testing, the team has developed a combination of the financial NLP model **distilroberta** and **hypertuned random forest classification** with an accuracy score of **0.91** and f1 score of **0.76**.

2. Business Problem Background

Stock price fluctuations especially in the short term are inherently volatile and are often influenced by a myriad of factors, including industry performance, political dynamics, economic indicators, company reputation, corporate news and company performance, therefore predicting the price movement trends of stocks becomes a challenging and interesting task. Real-time news usually encodes the first clues of major changes affecting a company, hence sentiment, essentially a gauge of herd mentality, can be considered a driver of price momentum.

Social media financial news:

In 2021, Reddit caught the attention of people around the world due to one event: The short squeeze of Gamestop's stock price (Smith, 2021)[2]. Gamestop is a brick-and-mortar store where their business model revolves around selling consoles and games in physical form like CDs. Their business was waning due to the proliferation of online game purchases. As a result, GameStop's stock price declined, leading many institutional investors to believe it would continue falling, thus short-selling the stock. In January 2021, the shares of GameStop experienced a dramatic increase in price due to a short squeeze initiated by retail investors on social media platforms like Reddit's WallStreetBets. It resulted in a 600% rise in its stock value over the next few days. Beyond GameStop, other "meme stocks" like AMC Entertainment had experienced significant price movements driven by retail investor sentiment and social media hype. The short squeeze of AMC Entertainment occurred in 2021 when private investors united in Reddit and Twitter social networks to buy up the shares of struggling AMC Entertainment Holdings Inc. These price swings were not related to the companies' fundamentals, but rather to the market sentiment.

Traditional media financial news:

Transocean, a Switzerland-based offshore drilling contractor, was doing well until oil slumped following the OPEC meeting that wrapped up on 25 May 2017. The shares of Transocean plunged 7.6 percent that day, leaving them down 4.5 percent for one year, compared with the S&P 500's rise of 16 percent during the period.

If Transocean shares were traded based purely on news sentiment, buying when the news sentiment was positive and selling when negative, this could have generated a gain once optimised as of May 25, instead of the loss from a simple buy-and-hold strategy. That could be achieved by going long when the news sentiment crossed above 0.225 and going short when it dropped below -0.225.

The above examples demonstrate how market sentiment, driven by factors such as social media discussions, economical and political factors can lead to substantial movements in stock prices. Hence, it is important for investors to be aware of the potential impact of the sentiment on the markets driven by the aforementioned factors.

Key players and user needs in the industry:

Bloomberg Terminal, a professional financial data and analytics platform, primarily designed for institutional investors and financial professionals, provides a news sentiment analysis tool that leverages on machine-learning techniques to determine news and social sentiment from text, generated in real time.

However, for most retail investors, the Bloomberg terminal's cost and complexity may be prohibitive. Moreover, even with knowledge on the financial sentiments, it may still not be enough for retail and institutional investors alike to make financial market decisions. Therefore, they may need to find more cost-effective solutions to better suit their needs.

3. Project Objective and Overview

The objective of this project is **to design a recommendation system based on financial sentiment analysis against the historical stock price. This will equip users with valuable insights and the knowledge to anticipate market shifts**. The solution provides the following:

1. Real-time access to news articles that significantly impacts stock performances,
2. Algorithmic computation of sentiment scores and labelling of the news articles,
3. An actionable recommendation to capitalise on market opportunities promptly.

The project aims to create a Minimum Viable Product (MVP) with a company, Tesla, (TSLA) as a preliminary prototype with the following outline (Detailed justifications of our method are explained in **5.2 Design Justification**):

Step 1: Historical news collection

To analyse stock patterns, REST API is used to fetch the historical news in JSON formatted files.

Step 2: Data preparation

The JSON formatted raw files are pre-processed using a python script to extract the relevant details - - date and time of the publication, title of the article, ticker name based on relevance score higher than 0.8. A similar approach is conducted to extract the data from Reddit social media platform and a data processing script is employed to clean the data. This helps to structure and prepare the data for sentiment analysis.

Step 3: Perform News Sentiment Analysis

Pre-trained models from HuggingFace open sourced library- ProsusAI/finbert, distilroberta, yiyanghkust/finbert-tone and soleimanian/financial-roberta-large-sentiment, are deployed to perform sentiment analysis. The aim is to obtain sentiment scores i.e. positive, negative, neutral of the historical news articles which will be part of the feature set for supervised learning in later part. We first utilised the above NLP models on the articles from both traditional finance and the posts from r/wallstreetbets.

Step 4: Feature selection, Data scaling and Price classification

We select **six** features for training & testing of the classification model:

1. Aggregate positive sentiment score
2. Aggregate positive sentiment article counts
3. Aggregate negative sentiment score
4. Aggregate negative sentiment article counts
5. Aggregate neutral sentiment score
6. Aggregate neutral sentiment article counts

We then apply standard scalar function in the python library to rescale the feature values as the features values have a wide range of numbers.

From yahoo finance, we obtained daily closing prices from January to December 2023 and calculated the day-on-day percentage change% (pct% change). We also prepared two target variables: a Day-0 (same day as news outlet date) and a Day-1 (next day vs news outlet date). We initialise the target feature with 3 classification groups under different pct% change: 'Buy', 'Sell', 'Hold'.

For train test split, 80% of data is used for training, and 20% for testing.

Step 5: Perform supervised learning to predict price movement

The six features selected in **step 4** are used to train our models using machine learning classification algorithms - Linear SVC, Random Forest classifier, KNN, decision tree, Naïve Bayes Gaussian, etc.

We carry out a series of iterations to address 3 key areas:

1. Data performance for: Traditional news vs Traditional + Social news
2. Optimal classification pct% range for 'Buy', 'Sell', 'Hold'.
3. Optimal hyperparameters

The results are compared and the model with the highest accuracy and F1 score is selected.

4. Knowledge modelling

Knowledge modelling can be decomposed into three main stages, namely

4.1 Knowledge identification
4.2 Knowledge specification
4.3 Knowledge refinement

4.1 Knowledge Identification

The table below addresses the main sources of data used in the project. They were identified to be crucial for our architecture.

Table 1: Knowledge source and Acquisition technique

No	Source of information	Knowledge acquisition technique	Insights from information source
1	Mboum website	API call	The API call obtains real time information on new feeds and stock prices based on the ticker selected by the user.
2	Alpha Vantage website	API call	The API call extracts the raw historical stock market new feed from January to December 2023 for sentiment analysis and training of the classification model.
3	Reddit	Torrent	The data download is from Jan to Dec 2023 and used for training of the classification model.
4	Yahoo finance	API call	The yfinance python library extracts the historical stock price from Jan to Dec 2023 for training of the classification model.

4.2 Knowledge Specification

4.2.1 Social finance articles/chatters (Reddit)

Reddit offers a free API which allows a user to retrieve data from the posts/comments. However there is a limitation. A user is unable to pull data based on a fixed timeframe and the articles are often only classified into "Rising", "Hot", "New" and "Top". This limits the user's ability to study "less hot" entries.

As such, a member of the community from subreddit "r/pushshift" (Watchful1, 2024)[3], has helped to compile years of Reddit submissions (first post) and comments into .zst files for public usage (Watchful1, 2024)[4]. As such, the Reddit data used here in this report for preliminary model training is first retrieved by torrenting of .zst files, data extraction into .csv for subreddit "r/WallStreetBets" via pre-built python script (Watchful1, 2024)[5] then process through Jupyter Notebook (Figure 1 below).

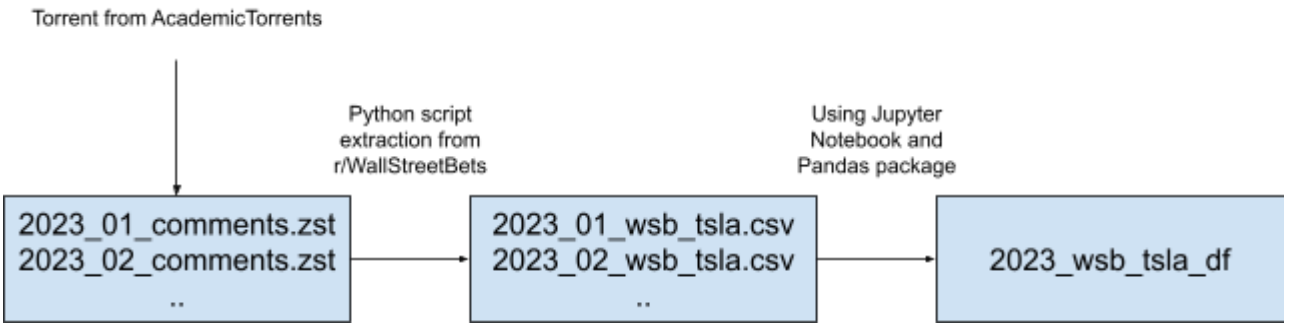


Figure 1: Retrieval flow of Reddit posts on TSLA

4.2.2 Traditional Finance News

Traditional Financial News are identified to be one of the important sources for data for financial sentiment analysis. Large variety of relevant news from different sources can be easily obtained using API software. As such, the Alpha Vantage software was used to pull historical traditional financial news structured data via JSON format. Alpha Vantage offers multiple filters capability like tickers, time from to time to, pull limit, which the team utilises to store proper labelled data in JSON format. Further refinement of these data could be done, which will be explained in **4.3 Knowledge Refinement**.

4.2.3 Stock Prices

Stock prices are obtained from Yahoo Finance (Ranaroussi, 2021)[6]. It is done by a simple data pull request via a free API. The API is highly flexible, allowing the user to extract

desired dataset via ticker, various time intervals (e.g. "5m", "1d") and time periods. This is similar to the demands of a user when he/she manoeuvres around the charts.

In this report, the ticker "TSLA" (Tesla) 's Close Price was extracted with a "1d" time interval over the market opening days in the year of 2023. A day-on-day pct price change was derived with the formula as follows:

$$Percentage\ Price\ Change\%\ (Day\ 0)\ =\ \frac{Close\ Price(Day\ 1) - Close\ Price(Day\ 0)}{Close\ Price(Day\ 0)} \times 100$$

An initial classification of signal was also introduced so as to kickstart the model learning exercise to find an optimal classification for percentage price change (see section 7).

Table 2: A sample of TSLA stock price (post data wrangling)

	Date	Close	pct_change	signal
0	2023-01-04	113.639999	0.051249	buy
1	2023-01-05	110.339996	-0.029039	neutral
2	2023-01-06	113.059998	0.024651	neutral
3	2023-01-09	119.769997	0.059349	buy
4	2023-01-10	118.849998	-0.007681	neutral

4.3 Knowledge Refinement

For traditional financial news data, the API pull via Alpha Vantage generates important relevance scores using their in-built model based on the input ticker filter. Hence, some of the data might not be relevant based on low relevance scores. These irrelevant data mentions the ticker symbol or the company itself, without providing much actual sentiment on it. Hence, these data should be removed, else they may longer the computational duration and effort required during model training to even negatively affect the final performance. The team uses Alpha Vantage built-in NLP model to obtain the relevance scores of each traditional news and filters them via the minimum relevance score. Based on He, 2013[7], the optimal relevance score is **0.80** for best performance based on different models. Attached are the figures of models' performance against the threshold t value:

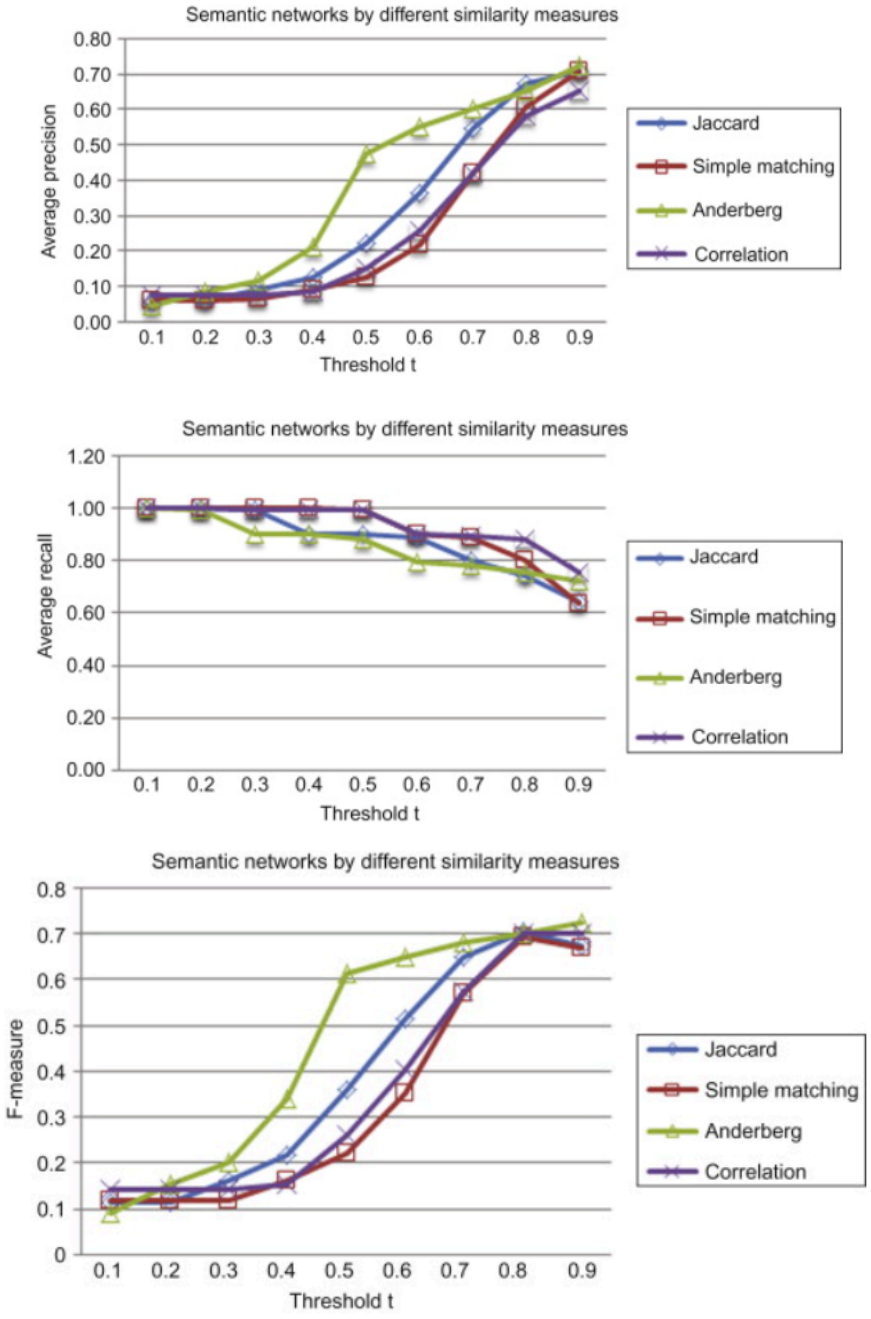


Figure 2: Different performance metrics vs Threshold t (He, 2013)[7]

Using a relevance score of **0.80 and above**, there is an estimated **88%** of data loss for a year's worth (2023) collected data (See Figure 3). Despite this loss, the retained data is sufficient for machine learning. Based on our training features of **6 variables** (to be explained in **5.2 Design Justification**), the minimum amount of data required would be **10 times** of the variables (Smolic, 2022)[8], which is **60**. A relevance score of at least 0.80 and above would yield around **1416** numbers of data. This allows the team to generate refined quality data with sufficient quantity. Moving forward, the team will look into collecting up to 5, 10 and even 15 years of data in future developments.

Figure 3: Amount of data based on different relevance score

5. System Design

5.1 Architecture design

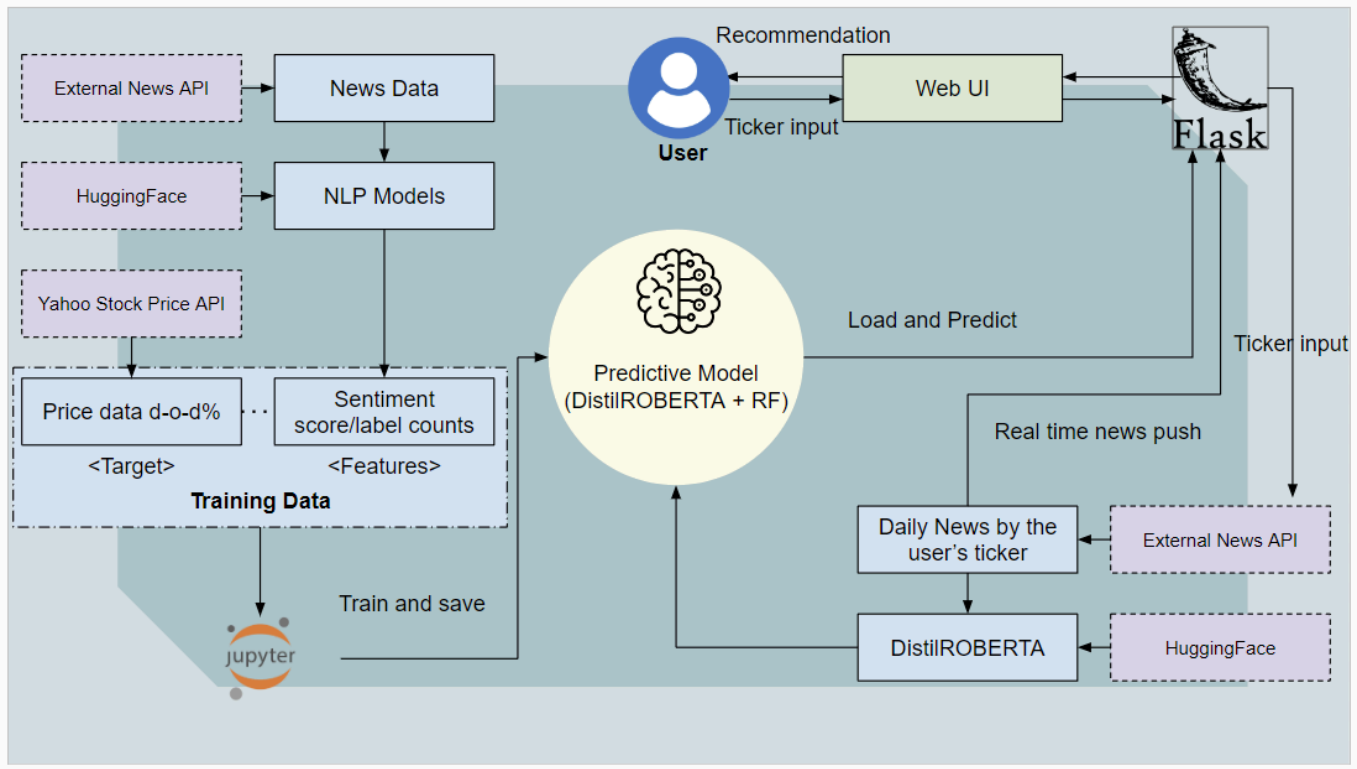


Figure 4: Diagram illustrate the process of model training and serving with prediction using real time data from external API

The project system architecture is broken down into **3** key areas:

A. Pre-training of a classification model to reduce latency

First, a flow is set up to pre-train a model:

1. Financial news data from an external API pull.
2. From HuggingFace, a range of Natural Language Processing models that were trained on financial news were selected in the iterations and performance comparison.
3. The news data were then fed into the models to obtain both sentiment scores and labelling. We aggregated the score and article counts for further usage later.
4. From Yahoo Finance API, the stock prices of TSLA were obtained and processed into a day-on-day percentage price change data.
5. Together, the sentiment data plus with the price data, a complete training dataset that was required as created. Sentiment data are the features while the price data are the target variables. The price data were then grouped and initialized with 3 classes of 'Buy', 'Sell', 'Hold' and iterated later on in the training to fine tune the %price range.
6. The training data is fed into a series of classification models and further compared later with each NLP and classification model combination.

7. The training was done on Jupyter Notebook and our final predictive model is saved as a pickle file for deployment.

B. Consumer UI and backend APIs integration

For a consumer UI point of view:

1. Flask was used to create the web UI with some basic chart, table and graphing techniques:
2. A user first has to access the web UI to provide a stock ticker input.
3. The input goes through Flask and is fed into the API pull for daily news based on the selected ticker.
4. The daily news is fed into the distilroberta NLP model used in the final predictive model.
5. With the sentiment scores and label counts, the trained model is used together with the new inputs to predict the day's signal.

C. Delivery of Recommendations

There are **three** things which will be fed back to the user:

1. The daily news data feed with the respective sentiment scores and the coresponding stock prices will be passed back to Flask. Thirdly, the recommendation will also be sent back through Flask.
2. Thereafter, the user can view the recommendation, stock prices, stock news and sentiment scores all in one on the web UI.

5.2 Design Justification

Since the financial news are influential towards the stock market (SOON, 2010)[9], with examples like analysts providing bull/bear cases with calculated insights and fundamental analysis of the companies' balance sheets, financial news is one of the most important pieces of data to collect and train with. Financial NLP models can be used to train on such financial news to give an overall sentiment score for each positive, neutral and negative category.

When it comes to financial NLP models, there are definitely a few that come to mind. FinBERT, XLP and Roberta, etc. The team will look at performance metrics like accuracy but possibly focus more on precision, recall & f1 scores. This is due to the possibility of imbalanced data (to be addressed in **7. Results and Progress**). According to Arslan et al., 2021, tests have been completed for the top few financial models in table 3. To further narrow it down, our team will look into the top two NLP models as highlighted in table 4 (Arslan et al., 2021)[10].

Table 3: Performance results based on the 4 different models (Arslan et al., 2021)[10]

Epoch		1			3			5		
Datasets	Model	P	R	F1	P	R	F1	P	R	F1
BBC	BERT	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97
	DistilBERT	0.97	0.97	0.97	0.97	0.97	0.97	0.98	0.98	0.98
	RoBERTa	0.98	0.98	0.98	**0.99**	**0.99**	**0.99**	0.98	0.98	0.98
	XLM	0.89	0.88	0.88	0.97	0.97	0.97	0.97	0.97	0.97
	XLNet	0.97	0.97	0.97	0.98	0.98	0.98	0.98	0.98	0.98
20News	BERT	0.85	0.85	0.85	0.92	0.92	0.92	**0.93**	**0.93**	**0.93**
	DistilBERT	0.82	0.82	0.82	0.90	0.90	0.90	0.91	0.91	0.91
	RoBERTa	0.84	0.84	0.84	0.92	0.91	0.91	**0.93**	**0.93**	**0.93**
	XLM	0.89	0.89	0.89	0.92	0.92	0.92	**0.93**	**0.93**	**0.93**
	XLNet	0.85	0.85	0.85	0.91	0.91	0.91	**0.93**	**0.93**	**0.93**
Proprietary-1	BERT	0.86	0.88	0.87	**0.88**	**0.89**	**0.88**	**0.88**	**0.89**	**0.88**
	DistilBERT	0.86	0.88	0.87	**0.88**	**0.89**	**0.88**	**0.88**	**0.89**	**0.88**
	RoBERTa	0.83	0.86	0.83	**0.88**	**0.89**	**0.88**	0.87	0.88	0.87
	XLM	0.81	0.84	0.81	**0.88**	**0.89**	**0.88**	0.83	0.86	0.83
	XLNet	0.83	0.86	0.82	**0.88**	**0.89**	**0.88**	**0.88**	**0.89**	**0.88**
Proprietary-2	BERT	0.70	0.81	0.74	0.95	0.95	0.95	0.96	0.96	0.95
	DistilBERT	0.72	0.84	0.78	0.96	0.96	0.96	**0.97**	0.96	0.96
	RoBERTa	0.89	0.89	0.87	0.95	0.95	0.95	**0.97**	**0.97**	**0.97**
	XLM	0.71	0.70	0.70	0.91	0.91	0.91	0.94	0.94	0.93
	XLNet	0.91	0.90	0.89	0.94	0.94	0.93	**0.97**	0.96	0.96

Table 4: FinBERT vs RoBERTa Performance (Arslan et al., 2021)[10]

Epoch		1			3			5		
Datasets	Model	P	R	F1	P	R	F1	P	R	F1
BBC	RoBERTa	0.98	0.98	0.98	**0.99**	**0.99**	**0.99**	0.98	0.98	0.98
	FinBERT	0.96	0.96	0.96	0.97	0.96	0.96	0.96	0.96	0.96
20News	RoBERTa	0.84	0.84	0.84	0.92	0.91	0.91	**0.93**	**0.93**	**0.93**
	FinBERT	0.86	0.86	0.86	0.92	0.92	0.92	**0.93**	**0.93**	**0.93**
Proprietary-1	RoBERTa	0.83	0.86	0.83	**0.88**	**0.89**	**0.88**	0.87	0.88	0.87
	FinBERT	0.86	0.88	0.87	**0.88**	**0.89**	**0.88**	**0.88**	**0.89**	**0.88**
Proprietary-2	RoBERTa	0.89	0.89	0.87	0.95	0.95	0.95	**0.97**	**0.97**	**0.97**
	FinBERT	0.74	0.82	0.76	0.96	0.95	0.95	**0.97**	0.96	0.96

The financial news that was previously pulled and pre-processed in **4. Knowledge Modelling** will be fed into the financial NLP models to obtain the sentiment score for positive, neutral and negative, along with the total counts of articles that were deemed positive, neutral and negative. This will ultimately yield 6 feature columns that will be used in model training for the second part of the architecture design.

For the second part, the 6 feature columns data generated from the NLP models will be trained together with the target variable of a stock's historical price movements. The 6 feature columns will be applied with standardscalar. This is to avoid overfitting of data during the training and testing of the model. The stock price movements will be categorised into buy/hold/sell signals for the MVP, which may be further expanded into strong buy/sell signals in the next phase of the project. Justification of the range of stock price movements to fall into the buy/hold/sell category will be tested and optimised, along with a machine learning model during training, testing and hyperparameter tuning, to yield the best performance model. The three different categories of buy/hold/sell will be encoded into classes of 0,1 & 2

to enable effective interpretation of the categories in the machine learning model training and testing phase.

Lastly, the team wants to design a machine learning model that can find a relationship or correlation between the sentiment scores and the historical price movements. Based on figure 5, the guide allows us to narrow down our modelling method down to classification (Cournapeau, 2007)[11]. This is because we have labelled data from yahoo finance, Alpha Vantage, etc.

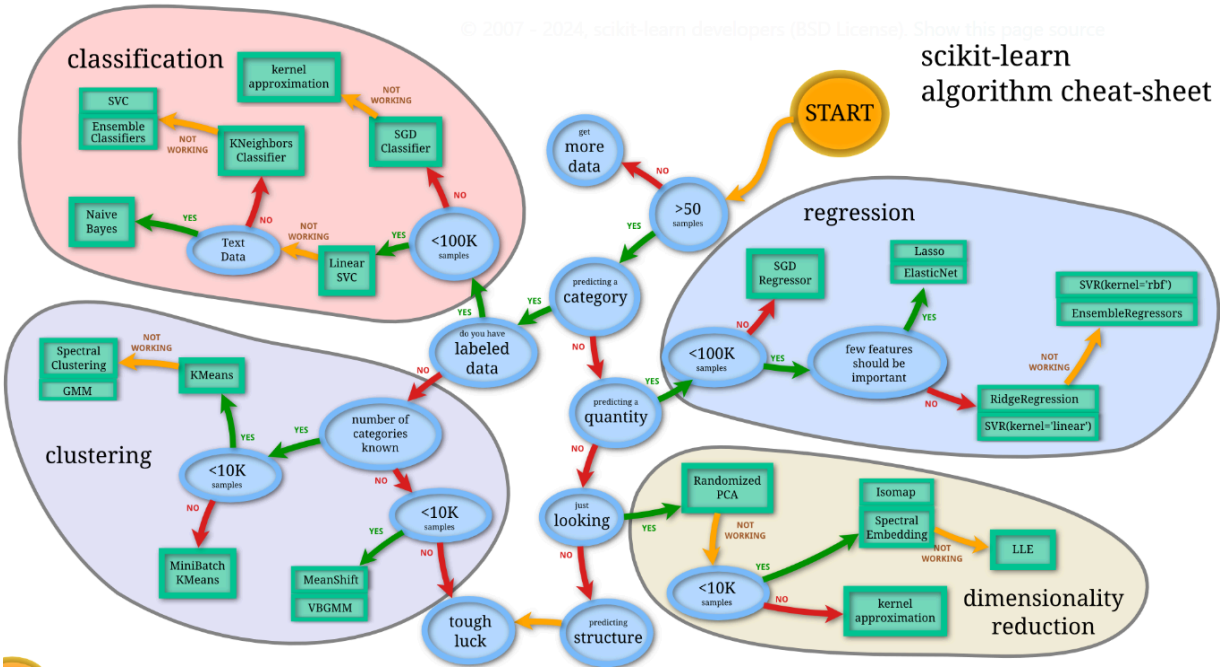


Figure 5: Scikit-learn machine learning techniques selection guide mindmap[11]

Therefore, the team aims to find the best combination of Fin-NLP models and the classification models through training and testing. The best performance combination will be selected as the final design model. This will be explained in **7. Results and Progress**.

6. System Development

6.1 System Implementation

For the main system process, python was decided due to the availability of extensive libraries and rigid frameworks, along with strong community support and high level of scalability if needed. A large number of trained financial NLP models are available on github and huggingface. Using python allows easy translation and deployments of such models. Preprocessing of the API-pulled data was done only via python jupyter as the amount of data is not too large. Nevertheless, the team will consider adopting Pyspark for preprocessing in the future developments when the data gets larger.

The Transformers library was also imported for the pipeline processing of the texts, along with AutoTokenizer and AutoModelForSequenceClassification to use specific tokenizer packages meant for each NLP and for looping of the different models for performance testing. This library is definitely easy to use, given the user-friendly connectivity between the sources and python, as well as the strong performance due to the high level tuning and large training done previously by the experts. Therefore, the team is confident of this implementation even in real-world scales.

The Scikit-learn package was highly utilised too, mainly the different classification models, LabelEncoder, train_test_split and the performance metrics for the team to compare and decide the best combinations of models. Scikit-learn was chosen due to its simplicity and yet, high performance models with smaller datasets. For future developments of the project, where more ticker symbols are considered with a longer period of data pulls (~5, 10, 15 years), the team will consider more complex packages like Tensorflow and pyTorch, etc for large-scale production and scalability, as well as the capability to build complex models to fit and predict the huge variety of tickers' prices and sentiment data.

In summary, the following tools was used during the system implementation phase:

1. VSCode
2. Anaconda
3. Jupyter Notebook
4. Github
5. HuggingFace/AutoTransformer
6. Docker/Docker Hub

In summary, the following runtime/library was used during the system implementation phase:

1. Python 3.11
2. Scikit-learn 1.3.0
3. Flask 2.2.2
4. Numpy 1.24.3
5. Pandas 2.0.3
6. Plotly 5.9.0

7. Joblib 1.2.0
8. Jquery 3.7.1

The following External API was used to retrieve real time data

1. Yahoo Finance API (https://developer.yahoo.com/api/)
2. RapidAPI (https://rapidapi.com/)

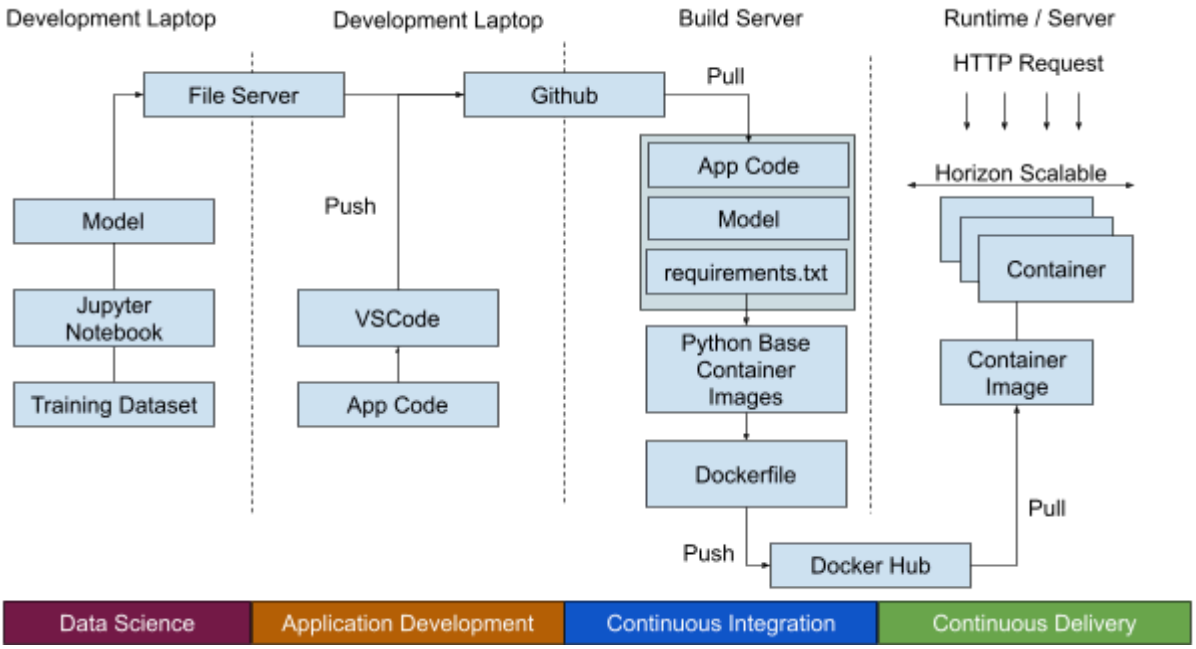


Figure 6 Diagram illustrates the various tools used during the development lifecycle to collaborate between team members and deployment.

6.2 Project

Share code snippets, screenshots, or demos of your system, connecting them to academic and/or market relevance

6.2.1 Leverage on Pre-trained model

The project leverage on pre-trained model with the help of transformer framework from hugging face.

```
print('Loading Tokenizer : ' , config.TOKENIZER)
tokenizer = AutoTokenizer.from_pretrained(config.TOKENIZER)
print('Loading Model : ' , config.MODEL)
model = AutoModelForSequenceClassification.from_pretrained(config.MODEL)
self._sentiment_analysis  = pipeline('sentiment-analysis', model=model, tokenizer=tokenizer)
```

Figure 8 Diagram showing code snippet for load model using transformers

6.2.2 Experiment with various algorithms.

Various techniques have been applied to find the best algorithm to correlate the sentiment score and actual share price movement. The experiment stage includes using various

classification models from scikit-learn library which was taught in class including LinearSVC, DecisionTreeClassifier and more. Performance of each classification model was captured and documented.

```python
#Linear SVC model
from sklearn.svm import LinearSVC
from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, f1_score

from sklearn.model_selection import GridSearchCV
def linear_svc(X_train, y_train, X_test, y_test):
    param_grid = {'C': [0.001, 0.01, 0.1, 1, 10, 100]}
    grid_search = GridSearchCV(LinearSVC(), param_grid, cv=5, n_jobs=-1)
    grid_search.fit(X_train, y_train)
    best_C = grid_search.best_params_['C']

    final_model_1 = LinearSVC(C=best_C)
    final_model_1.fit(X_train, y_train)
    y_pred = final_model_1.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    f1 = f1_score(y_test,y_pred, average="weighted")

#     print("F1 Score:", f1, "C:", best_C)
#     print("Accuracy on test set: {:.3f}%".format(final_model_1.score(X_test, y_test)*100))
    return f1, accuracy*100

f1, accuracy = linear_svc(X_train, y_train, X_test, y_test)
print("F1 Score : ", f1 )
print("Accuracy : ", accuracy)
```

Figure 9 Diagram showing code snippet for experimenting performance using LinearSVC

```python
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt

# Assuming merged_df is your DataFrame containing 'Positive', 'Neutral', 'Negative', and 'pct_change_category_encoded' columns

def decision_tree(X_train, y_train, X_test, y_test):
    # Define and train the Decision Tree model
    tree_model = DecisionTreeClassifier(max_depth=6,criterion='gini',random_state=0)  # You can adjust the max depth as needed
    tree_model.fit(X_train, y_train)

    # Make predictions on the testing set
    y_pred = tree_model.predict(X_test)

    # Evaluate the model's performance
    accuracy = accuracy_score(y_test, y_pred)
    f1 = f1_score(y_test, y_pred, average="weighted")
    return f1, accuracy *100
f1, accuracy = decision_tree(X_train, y_train, X_test, y_test)
print("F1 Score : ", f1 )
print("Accuracy : ", accuracy)
```

Figure 10 Diagram showing code snippet for experimenting performance using DecisionTreeClassifier

```python
def random_forest_classifier(X_train, y_train, X_test, y_test):
    # Initialize the Random Forest classifier
    rf_classifier = RandomForestClassifier(random_state=42)

    # Define the hyperparameters grid
    param_grid = {
        'n_estimators': [100, 200, 300, 400,500],
        'max_depth': [2, 10, 20],
        'min_samples_split': [2, 5, 10],
        'min_samples_leaf': [1, 2, 4],
        'max_features': ['sqrt'],
        'criterion': ['gini', 'entropy']
    }

     # Initialize GridSearchCV
    grid_search = GridSearchCV(estimator=rf_classifier, param_grid=param_grid, cv=5, n_jobs=-1)

    # Fit the grid search to the data
    grid_search.fit(X, y)

    best_params = grid_search.best_params_
    best_score = grid_search.best_score_

    # Use the best parameters to create the final Random Forest classifier
    best_rf_classifier = RandomForestClassifier(**grid_search.best_params_, random_state=42)

    # Train the classifier using the best parameters
    best_rf_classifier.fit(X_train, y_train)

    # Make predictions on the test set
    y_pred = best_rf_classifier.predict(X_test)

    # Make predictions on the testing set
    accuracy = accuracy_score(y_test, y_pred)
    f1 = f1_score(y_test,y_pred,  average="weighted")
    return f1, (accuracy * 100)
```

Figure 10 Diagram showing code snippet for experimenting performance using RandomForestClassifier

6.2.3 Saving and serving final trained model

The trained model with the desired fitness was saved using joblib for higher performance and smaller footprint.

```python
import joblib
# Save the trained model to a file
joblib.dump(model, 'model.joblib')


model = joblib.load(model.joblib')
```

Figure 7 Diagram showing code snippet for dumb and load model using joblib

6.3 Technical Challenges

6.3.1 Vast number of NLP models to choose from

Despite the large access to multiple financial NLP models from the communities, many financial NLP models were stated to have the best performance scores compared to the others, each having been trained with their own set of sources and fine-tuning. For example, BERT alone has many variations. BERT NLP has Bidirectional Encoder Representation capability, and there is even FinBERT, BERT NLP trained specifically with financial corpus. Then, there are also different specialised BERTs like RoBERTa, which is robustly optimised with training on BookCorpus, Wikipedia Data, CC-News, OpenWebText and Stories Datasets (Kanz, 2023)[12]. Therefore, further comparisons between the top few performance NLP models have to be carried out during the model design and training phase to filter out the best suited NLP for our classification model. Further performance tests were done with a variety of combinations of NLPs and classification models to give the best accuracy and f1 scores. Details are mentioned in **7.2 Performance Testing**.

6.3.2 Data handling for articles from Reddit

In terms of the limitations from on Reddit, a user is unable to pull data based on a fixed timeframe and the articles are often only classified into fixed categories that limits the visibility of "less hot" entries.

The second limitation is computational limitations. Even though the community from subreddit did compile years of Reddit entries into .zst files for public usage, such compressed files have huge amounts of data and require larger computational power to repeatedly extract from multiple subreddit threads.

The third limitation is that the articles often contain inappropriate/promotional articles which are irrelevant to the topic of our interests. We will need a stronger filtering mechanism to omit these noises.

Nevertheless, the team did a reasonable amount of filtering and tested the performance of the final model with Reddit data. However, the team found that with the inclusion of Reddit data, the performance is still not doing well and hence, we chose to omit the data at this stage.

6.3.3 Price classification

Even though financial news articles have an impact on the stocks' prices movement (Shynkevich et al., 2015)[13], it is still extremely difficult to predict the actual stock prices change percentage just with the results of the sentiment analysis of related articles. Therefore, to simplify the machine learning training and prediction, the team has decided to label the set of price changes into three main categories: buy, neutral and sell. These three categories are then encoded into multi-classes target variables of 0,1,2 for model training and testing purposes. As such, the predicted output would be in the values of 0,1 & 2, meaning

buy, hold and sell. The price percentage change detail for each category would be displayed along with predicted output for the user to make further considerations in his investment/trading decisions.

7. Results & Progress

7.1 Assumptions & Considerations

The team has managed to find the best combination of financial NLP models with different classification models. There were a few points of considerations that the team has taken when designing the architecture:

a. Applying Auto Tokenizer to fit each NLP model as the team believes each NLP model is better suited with its own set of tokenizer choice, leaving no room for complications
b. **There is a need to label what is considered positive (buy), neutral (hold) and negative (sell) in the stock market world to simplify the recommendation for users.** The team has decided to justify these with percentage gains/loss of a stock price. Optimization will be performed to find the best gain/loss percentage that gives the best model performance.
c. After optimization of the percentage gain/loss of stock price, encoding was done to change the buy/hold/sell into counts of 0,1 & 2 for training purposes.
d. The 6 features columns are applied with standardscaler to have a normalised balanced scale. **This is to avoid having data with high standard deviation which may interfere with the training of model and performance.**
e. Train Test Split is chosen instead of cross-fold validation due to the possibility of having an imbalanced data class.
f. The different combination of NLP models and classification models were tested at the same buy/hold/sell label with percentage gain/loss of stock price fixed at **2%**. After finding the best combination, the stock price percentage gain/loss range for buy/hold/sell categories determination will be optimised along with the best combination of models during the fine tuning stage.
g. The categorization of a stock price movement into the buy/hold/sell signals are based on a selected stock, TSLA for the MVP. This is due to the huge price fluctuation caused by varying financial sentiments with a significant amount of sources. The team believes this is effective for the model training. **For the next phase of the project, the desired stock as input by the user shall be trained with that selected stock's historical price movements.**
h. Due to the stock price percent change labelling optimization and encoding, at a higher stock price percentage change range set (E.g. >+4.75% = Buy, <-4.75% = Sell), **the target variable of price signal encoding, would have a class imbalance of ratio >4 (Majority class:Minority class). Therefore, the team has considered prioritising the models' performance by their f1 scores with average set as macro.** (More details in **7.2 Performance Testing**)

7.2 Performance Testing

Based on the above considerations, the team worked on to find the best performing combination of financial NLP and classification models. Table 5 below shows the variety of combinations the team has tried to obtain the results:

Table 5: Accuracy & f1 score of different NLP-Classification Models combination

		Day 0 price change w/o social finance		Day 1 price change w/o social finance		Day 0 price change with social finance	
NLP Model	**Classification Model**	**Accuracy**	**f1_score**	**Accuracy**	**f1_score**	**Accuracy**	**f1_score**
Distilroberta	Linear SVC	0.622	0.446	0.530	0.370	0.520	0.349
Distilroberta	**K-NN**	**0.600**	**0.552**	0.510	0.360	0.580	0.501
Distilroberta	**Decision Tree**	**0.600**	**0.576**	0.42	0.360	0.480	0.424
Distilroberta	**Random Forest**	**0.644**	**0.596**	0.490	0.360	0.580	0.499
yiyanghkust/finbert-tone	Linear SVC	0.511	0.469	0.568	0.468	0.440	0.332
yiyanghkust/finbert-tone	K-NN	0.556	0.522	0.432	0.347	0.380	0.435
yiyanghkust/finbert-tone	Decision Tree	0.511	0.321	0.546	0.503	0.460	0.428
yiyanghkust/finbert-tone	Random Forest	0.511	0.428	0.568	0.532	0.500	0.381
yiyanghkust/finbert-tone	Kernel Approximation	0.600	0.470	0.591	0.485	0.460	0.348
ProsusAI/finbert	Linear SVC	0.5668	0.4369	0.467	0.537	0.520	0.417
ProsusAI/finbert	K-NN	0.5111	0.4486	0.556	0.457	0.400	0.4643
ProsusAI/finbert	Decision Tree	0.5778	0.529	0.556	0.408	0.520	0.405
ProsusAI/finbert	**Random Forest**	**0.6222**	**0.7342**	0.538	0.667	0.512	0.578
soleimanian/financial-roberta-large-sentiment	Linear SVC	0.58	0.477	0.580	0.476	0.522	0.431
soleimanian/financial-roberta-large-sentiment	K-NN	0.53	0.477	0.490	0.427	0.422	0.405
soleimanian/financi	Decision Tree	0.56	0.732	0.560	0.483	0.495	0.443

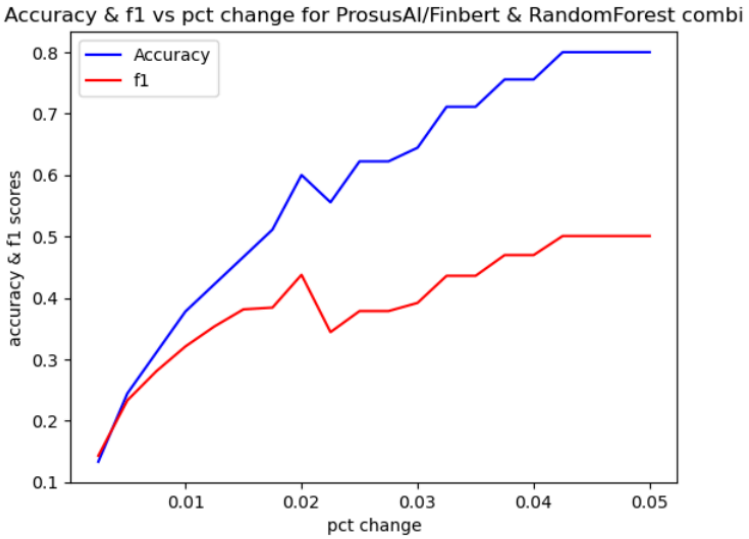
al-roberta-large-sentiment							
soleimanian/financial-roberta-large-sentiment	Random Forest	0.34	0.534	0.530	0.449	0.391	0.510

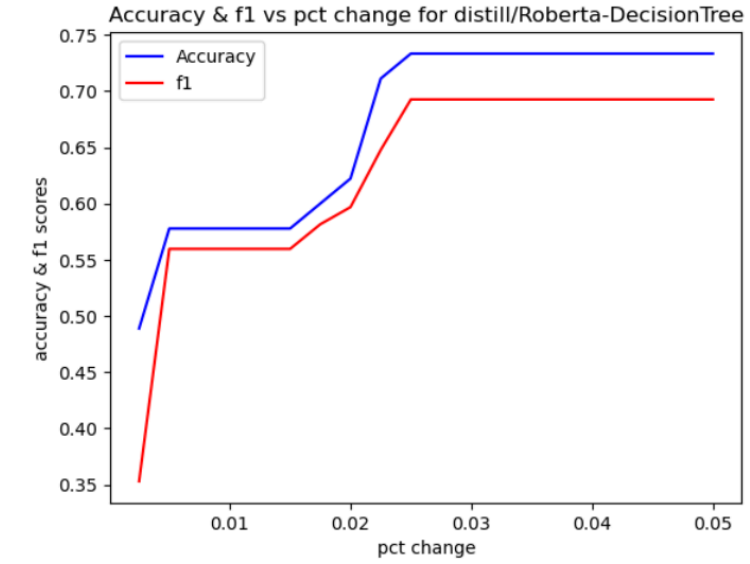
The table's results were compared with that of social media news in the model training.

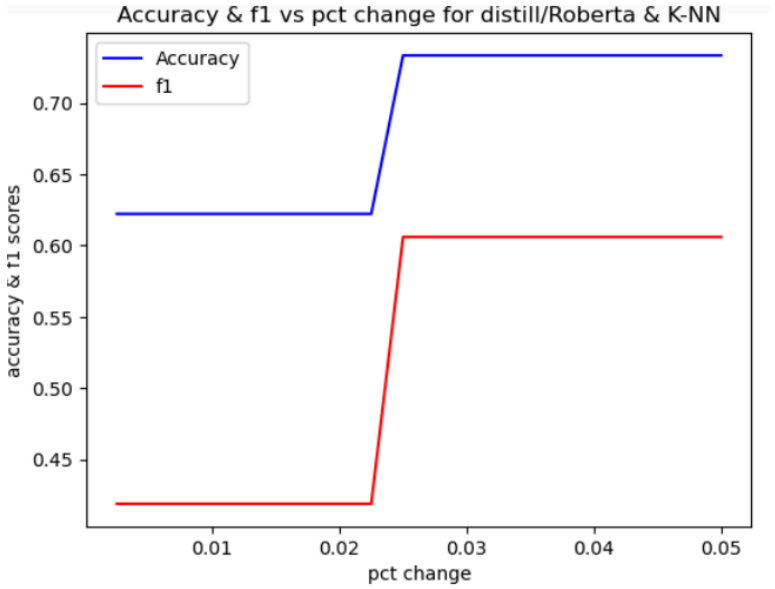
1. Generally, **the model training with only traditional finance news performed better** compared to the ones that was trained with traditional finance together with social media news.
2. Stock price percentage change was based on day 0 price change, i.e. (today price - yesterday price)/today price instead of day 1 price change, i.e. (tomorrow price - today price)/tomorrow price as **the performance with day 0 price change fared better.**

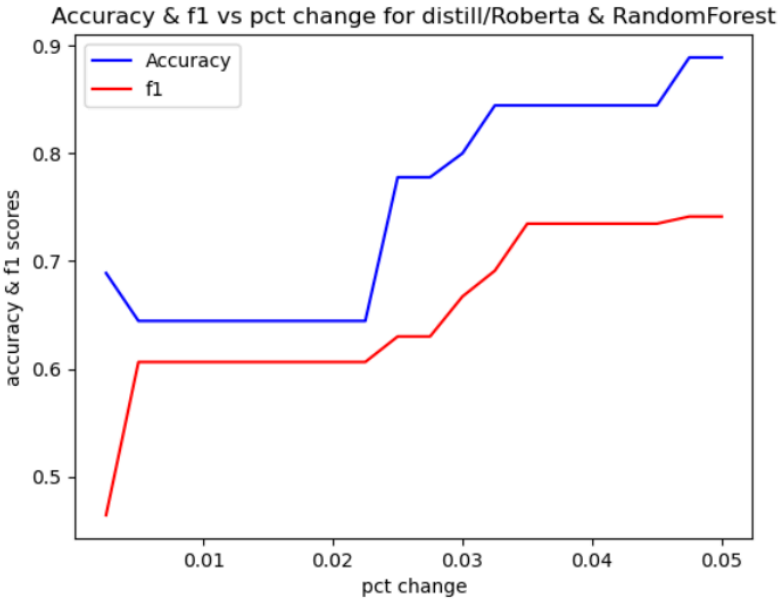
Based on the four combinations with the highest accuracy and f1 scores (highlighted in yellow and bold), the team ran further iterations with the different stock price percentage change to label the buy/hold/sell signal. This is to obtain the optimal stock price percentage change for the best model performance. For each classification model, hyper tuning of parameters as well at each stock price percentage loop.

For example, the random forest model was tuned based on the split criterion of gini or entropy, number of estimators of 200/300/400, max_depth of tree being none/10/20, minimum samples split in 5/10/20 and minimum samples leaf of 1/2/4. Figure 8 below shows the performance of each model while varying the stock price percentage change.




Accuracy & f1 vs pct change for distill/Roberta-DecisionTree
Accuracy
f1
accuracy & f1 scores
pct change
0.75
0.70
0.65
0.60
0.55
0.50
0.45
0.40
0.35
0.01
0.02
0.03
0.04
0.05


Accuracy & f1 vs pct change for distill/Roberta & K-NN
Accuracy
f1
accuracy & f1 scores
pct change
0.70
0.65
0.60
0.55
0.50
0.45
0.01
0.02
0.03
0.04
0.05


Accuracy & f1 vs pct change for distill/Roberta & RandomForest
Accuracy
f1
accuracy & f1 scores
pct change
0.9
0.8
0.7
0.6
0.5
0.01
0.02
0.03
0.04
0.05

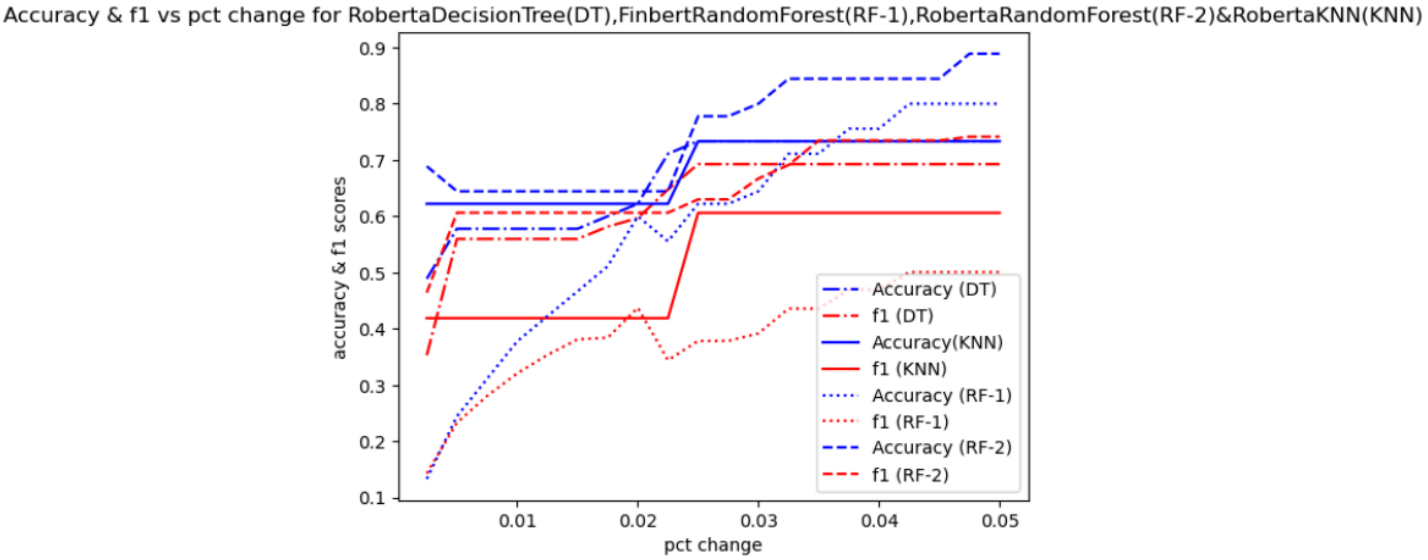


Figure 8: Accuracy and f1 scores of the models against varying stock price percent change

Table 6: Price percent change and accuracy, f1_scores and majority-Minority ratio for target variables for DistilRoberta and RandomForest model combination

Price percent change	Accuracy	F1 scores (Macro)	Majority-Minority Ratio
0.025%	0.711	0.479	0.124
0.050%	0.644	0.584	0.348
0.075%	0.644	0.584	0.554
0.10%	0.622	0.584	0.837
0.125%	0.622	0.624	1.17
0.150%	0.622	0.624	1.57
0.175%	0.622	0.624	2.27
0.200%	0.622	0.624	2.64
0.225%	0.622	0.624	3.65
0.250%	0.778	0.677	3.97
0.275%	0.778	0.677	4.14
0.300%	0.778	0.677	5.07
0.325%	0.778	0.735	6.71
0.350%	0.844	0.735	8.30
0.375%	0.844	0.735	9.50

0.400%	0.844	0.735	10.9
0.425%	0.844	0.735	12.0
0.450%	0.844	0.735	13.0
0.475%	**0.911**	**0.762**	**13.3**
0.500%	0.911	0.762	17.5

Based on Figure 8, after performing iterations with the top four different model combinations based on different stock price percent change labelling and classification model hyper tuning, our team has arrived at an optimal model combination of **Distilroberta** for the NLP model and the classification machine learning model as **Random Forest Classifier**. The best parameters for the Random Forest Classifer are as follows:

- the split criterion as **gini**,
- number of estimators at **200**,
- max tree depth at **20** and minimum sample split at **5**, minimum sample leafs at **1**.

Last but not least, the best stock price change label is set as

- **Buy signal: >+4.75%**
- **Neutral: between +4.75% to -4.75**
- **Sell signal: <-4.75%**.

Based on Table 5, these settings gave an optimal accuracy score of **0.91** and f1 score (average macro due to imbalance data) of **0.76**, with a majority to minority class ratio of **13.3**.

8. Future Developments

Although the product developed by the team is at the minimum viable product stage, there are still many developments that can be considered to further improve the product.

In the next phase of development, the user will be given a feedback option to rate and comment on the predicted outcome by the product. The user can refer to the displayed news articles' link, as well as look at the actual stock prices changes to give an overall feedback on whether the product has predicted accurately. The feedback will be collected and trained with the classification model to further improve the model.

Each stock price fluctuates differently. Some stocks prices vary more due to their sectors being more volatile to economies and even geopolitical events (Davis, 2022)[13]. It will not be accurate to use one model to predict all stock prices. Therefore, we need to consider an independent model for each stock to train their stock prices movement and better predict the outcome.

Each financial news article might not have the same impact on the stock prices due to a variety of reasons. Therefore, to improve the correlation of the financial news articles towards stock prices, the news articles can be categorised into different groups that exhibit different correlation weights to the stock prices. Shynkevich et al., 2015, proposed to split the news articles into five categories into the targeted stock, sub industry, industry, group industry and sector[14]. Then, the team could further filter out by the relevance score with a suitable threshold, followed by applying NLP models to get the sentiment scores. Then, methods like multiple kernel learning techniques or even neural networks could be performed to find the correlation weights of each category to the stock prices movement.

9. References

1. Deveikyte, J., Geman, H., Piccari, C., & Provetti, A. (2022). A sentiment analysis approach to the prediction of market volatility. *Frontiers in Artificial Intelligence*, *5*. https://doi.org/10.3389/frai.2022.836809
2. Smith, A. (2021, April 13). The reddit revolt: Gamestop and the impact of social media on institutional investors. The TRADE. https://www.thetradenews.com/the-reddit-revolt-gamestop-and-the-impact-of-social-media-on-institutional-investors/
3. Wafchful1. (2024). *R/pushshift on reddit: Reddit dump files through the end of 2023*. Reddit dump files through the end of 2023. https://www.reddit.com/r/pushshift/comments/194k9y4/reddit_dump_files_through_the_end_of_2023/
4. Watchful1, W. (2024). *Reddit comments/submissions 2005-06 to 2023-12*. Academic Torrents. https://academictorrents.com/details/9c263fc85366c1ef8f5bb9da0203f4c8c8db75f4
5. Watchful1, W. (2024b). *WATCHFUL1/push shift dumps: Example scripts for the pushshift dump files*. GitHub. https://github.com/Watchful1/PushshiftDumps/tree/master
6. Ranaroussi, R. (2021). *Ranaroussi/yfinance: Download market data from Yahoo! Finance's API*. Download market data from Yahoo! Finance's API. https://github.com/ranaroussi/yfinance
7. He, P. (2013). Emerging Trends in ICT Security. https://www.sciencedirect.com/science/article/abs/pii/B978012411474600027X
8. Smolic, H. (2022, December 15). *How much data is needed for machine learning?* . Graphite Note. https://graphite-note.com/how-much-data-is-needed-for-machine-learning/#:~:text=Generally%20speaking%2C%20the%20rule%20of,100%20rows%20for%20optimal%20results.davis
9. SOON, Y. C. (2010). *News which Moves the Market: Assessing the Impact of Published Financial News on the Stock Market* . Ink.library.smu.edu.sg. https://ink.library.smu.edu.sg/cgi/viewcontent.cgi?article=1057&context=etd_coll
10. Arslan, Y., Allix, K., Veiber, L., Lothritz, C., Bissyandé, T. F., Klein, J., & Goujon, A. (2021). A comparison of pre-trained language models for multi-class text classification in the financial domain. *Companion Proceedings of the Web Conference 2021*. https://doi.org/10.1145/3442442.3451375
11. Cournapeau, D. (2007). *Choosing the right estimator*. scikit. https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html
12. Kanz, J. (2023, January 18). FinBERT and FinRoBERTa models for financial sentiment analysis. WOLFRAM COMMUNITY. https://community.wolfram.com/groups/-/m/t/2792518
13. Davis, M. (2022). *The 8 most volatile sectors*. Investopedia. https://www.investopedia.com/financial-edge/0712/the-8-most-volatile-sectors.aspx
14. Shynkevich, Y., McGinnity, T. M., Coleman, S., & Belatreche, A. (2015). Predicting stock price movements based on different categories of news articles. *2015 IEEE Symposium Series on Computational Intelligence*. https://doi.org/10.1109/ssci.2015.107

Appendix

Appendix A: Project Proposal

Date of proposal:

16 March 2024

Project Title:

Intelligent Sentiment Trading Recommender

Sponsor/Client: *(Name, Address, Telephone No. and Contact Name)*

Institute of Systems Science (ISS) at 25 Heng Mui Keng Terrace, Singapore
NATIONAL UNIVERSITY OF SINGAPORE (NUS)
Contact: Mr. GU ZHAN / Lecturer & Consultant
Telephone No.: 65-6516 8021
Email: zhan.gu@nus.edu.sg

Background/Aims/Objectives:

1. *Background*

- In 2023, the Fed had implemented a series of interest rate hikes throughout the year to slow down inflation which had risen well above their goal of 2%. **With the rising cost of living caused by inflation, it becomes inevitable that a more efficient trading analysis tool is required to help investors**. Whilst traditional analysis of stocks focuses on hard numbers, markets are also driven by emotions. Investor sentiment including fear, greed, and optimism, significantly affects stock prices. **Sentiment analysis helps to gauge these emotions by analyzing news articles, social media posts, and other financial data to identify shifts in investor confidence before they are reflected in the actual price movements**. This allows investors to potentially make informed decisions ahead of the curve. Sentiment analysis helps to quantify the "softer" side of the market, providing a more comprehensive view.

- There are multiple available pre-trained Natural Language Processing (NLP) models in the market that could facilitate sentiment tagging. Some examples are **FinBERT** (ProsusAI, 2022)[1], which focuses solely on financial news articles and **X-roBERTa-base for Sentiment Analysis** (X formerly known as Twitter) which looks into the social media tweets analysis (Fan et al., 2019)[2]. NLP libraries like spaCy, NLTK have made sentiment analysis more possible and accurate.

- While there are plenty of common sentiment analysis models, **these models tend to face difficulty in negation handling, domain**

dependence, spam detection, and ambiguity in the form of abbreviations or sarcasm. Improved tools like **PyFin-sentiment** appear to better reflect "market sentiments" rather than "general sentiments"[3] (but fail to work as issues were not debugged by the code owner).

- On the financial side, there are alot of algorithm-based trading bots and **these bots execute trades based on data trends and a set of predefined rules**. **They do not consider the sentiments of the market, as well as any social, economic, geopolitical events**, etc (Medha et al.,2021)[4].

- **From a global point of view, in 2024, the emotion recognition and sentiment analysis software market size is projected to increase by USD 797.17 million, at a CAGR of 14.15% between between 2023 and 2028**[5]. Specific to the finance applications, the team has identified that there are increasing demands in finance markets for market insights and sentiment extractions. **The NLP in global finance markets is estimated to be growing at a CAGR of 27.6%, worth USD18.8 billion by 2028.** (Marketsandmarkets, 2023)[6]. Key players in the market include Microsoft, Google, IBM, AWS, Oracle, etc.

- On the social media front, Reddit, out of top 100 groups, 3 of which are related to finance/crypto with a total of 29M subscribers[7]. Of which, **r/WallStreetsBets was popularized from the Gamestop (GME) short squeeze saga in 2021 and even the traditional media recognizes the community effect on sentiment trading[8].**

- This project concept continues to be attractive for businesses to retail consumers as both groups are always keen on profit-generating ideas. Online bots have poor reviews[9] and there seems to be a lack of widely-discussed existing recommenders out in the market.

2. *Final Goal:*

- **To design an intelligent sentiment trading recommender that aims to provide market entry suggestions to the user.**
- The software will push notifications based on the user's subscribed company, industry and favorites.
- To push a notification signal for a recommended action (e.g strong buy), the underlying model will label news articles when they are first released. If it hits the right combination of articles and sentiment polarities with the right weightage between traditional finance (tradfi) and social finance (socialfi), a notification will be sent.
- The notification will consist of information (see *Roadmap*) to educate the user on the market sentiments before making any informed trading/investment decisions.

3. *Milestones/Objectives:*

- To leverage on **pre-trained NLP models with financial news' texts/social media inputs and categorize into different polarities of sentiments and company name/industry.**
- **To clean and prepare financial data from stock prices sources, converting into h-o-h or d-o-d %change time series**.
- **To train a model using supervised classification machine learning techniques using pre-trained NLP sentiment outputs and pre-processed financial data.** The model will be trained by breaking up different periods of trading years as train, validate and test dataset. The output will provide **recommendations to users to make investment decisions.**

4. *Roadmap:*

- **Phase 1 MVP:**
  A prediction software that is able to take user's inputs, read market sentiments and provide suggestions on trading actionables. The outputs from the software contain:
  a. Types of recommended actions (Strong Buy, Buy, Neutral, Sell, Strong Sell)
  b. Company/Sector
  c. Expected profit taking period (e.g Day 0, Day +1)
  d. A link to the article related to the stock.

  Phase 2 (Not included in this phase)
- Provide a confidence level for each recommendation to the users.
- Allow the user to rate the sentiments on the articles and feedback to the model to self-learn.
- To include Technical Analysis features in model training.
- To use image analysis machine learning software to scan for chart shapes and patterns to suggest entry points.

References:

1. ProsusAI. (2022). ProsusAI/finBERT: Financial sentiment analysis with bert. GitHub. https://github.com/ProsusAI/finBERT
2. Fan, R., Talavera, O., & Tran, V. (2019). *(PDF) Social Media Bots and stock markets*. Social media bots and stock markets. https://www.researchgate.net/publication/331639758_Social_media_bots_and_stock_markets
3. Wilksch, M., & Abramova, O. (2023a). Pyfin-sentiment: Towards a machine-learning-based model for deriving sentiment from financial tweets. International Journal of Information Management Data Insights, 3(1), 100171. https://doi.org/10.1016/j.jjimei.2023.100171
4. Medha Mathur, M., Mhadalekar, S., Mhatre, S. S., & Mane, V. (2021, August 1). Algorithmic Trading Bot. Researchgate. Retrieved August 3, 2024, from https://www.researchgate.net/publication/353770459_Algorithmic_Trading_Bot
5. Emotion Recognition and Sentiment Analysis Software Market Analysis North America, Europe, APAC, South America, Middle East and Africa - US, China, Japan, UK, Germany - Size and Forecast 2024-2028 https://www.technavio.com/report/emotion-recognition-and-sentiment-analysis-software-market-industry-analysis
6.MarketsandMarkets. (2023, April). *NLP in Finance Market Size & Share Analysis - Industry Research Report - Growth Trends*. MarketsandMarkets. https://www.marketsandmarkets.com/Market-Reports/nlp-in-finance-market-21737879.html
7. Top communities on reddit - page 1. (n.d.-c). https://www.reddit.com/best/communities/1/

8. The GameStop Episode: What Happened and What Does It Mean?. Cato.org. (n.d.). https://www.cato.org/cato-journal/fall-2021/gamestop-episode-what-happened-what-does-it-mean#game-stop-goes-crazy-in-an-interesting-way
8. 6 live sentiment analysis trading bots using Python | Udemy. (n.d.-a). https://www.udemy.com/course/sentiment-trading-python/

Requirements Overview:

- **Research ability**: To extract/interpret financial market sentiments based on financial/social media/event-driven news outlets.
- **Programming ability**: To perform sentiment analysis of stock's traditional and social media data.
- **System integration ability**: To combine financial news' sentiment and stock price actions to give user overall advices to make informed trading/investing decisions

Resource Requirements (please list Hardware, Software and any other resources)

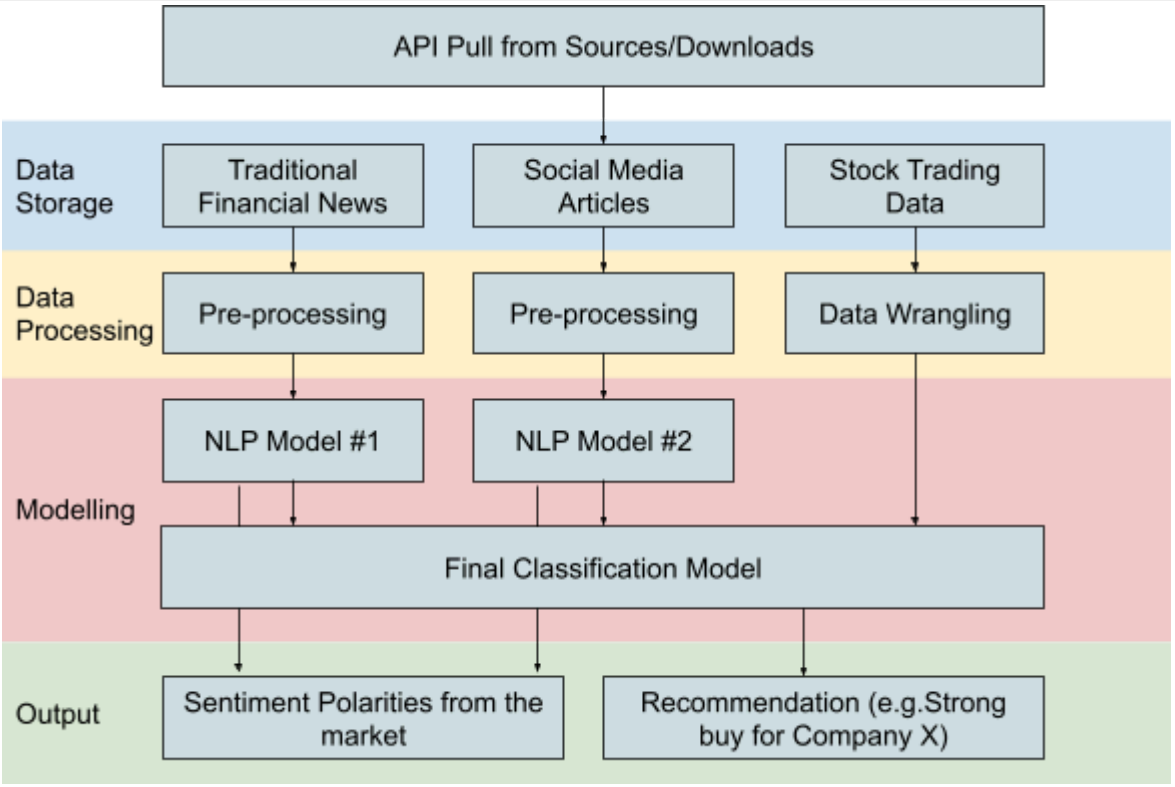
1. **Data Collection/Preparation**
   a. **News**
   - Open News API, PRAW, python-twitter
   - Social Media API (Twitter / X , Facebook / Meta, Reddit, StockTwits, LinkedIn)
   - Reuters, CNBC, Yahoo Finance News, The Business Times, Bloomberg

   b. **Stock screener**
   - FinViz
   - Yahoo Finance / Yahoo Finance API
   - Other (Alpha Vantage, Quandl, IEXCloud, Tiingo)

2. **Verification bots/Teaching assistant**
   - ChatGPT/Prof guidance

3. **System Design**
   a. **Hardware considerations:**
   - GPU, Laptop

   b. **Software considerations:**
   - spaCy, NLTK, Gensim
   - BERT, FinBERT, roBERTa, DistilBERT, LSTM (long short-term memory networks), VADER
   - sklearn, Decision Tree, KNN, SVM, RandomForest, Fuzzy Logic
   - Backtesting, Ta-lib, bt
   - Flask, Django, Bootstrap, jQuery, HTML, CSS, JS



Number of Learner Interns required: (Please specify their tasks if possible)

4 Learner Interns:

Learners	**Tasks**
Kenn Ng Ying Wee	Team Leader, NLP modelling (socialfi), Trading data wrangling, UI/UX, Video Editing, Report
Tan Kok Tong	NLP modelling (tradfi), Final Model compilation, UAT, Report
Lai Kah Hoe	NLP modelling (socialfi), UI/UX, UAT, Report
Yeo Li Ying	NLP modelling (tradfi), Trading data wrangling, Final Model compilation, Report

Methods and Standards:

Procedures	Objective	Key Activities
Requirement Gathering and Analysis	The team should meet with ISS to scope the details of project and ensure the achievement of business objectives.	1. Gather & Analyze Requirements 2. Define application design 3. Prioritize & Consolidate Requirements 4. Establish Functional Baseline
Technical Construction	· To develop the source code in accordance to the design. · To perform unit testing to ensure the quality before the components are integrated as a whole project	1. Setup Development Environment 2. Understand the System Context, Design 3. Perform Coding 4. Conduct Unit Testing (NLP, final classification model)
Integration Testing and acceptance testing	To ensure interface compatibility and confirm that the integrated system hardware and system software meets requirements and is ready for acceptance testing.	1. Prepare System Test Specifications 2. Prepare for Test Execution 3. Conduct System Integration Testing 4. Evaluate Testing 5. Establish Product Baseline
Acceptance Testing	To obtain ISS user acceptance that the system meets the requirements.	1. Plan for user acceptance testing 2. Conduct training for acceptance Testing 3. Prepare for Acceptance Test Execution 4. ISS Evaluate Testing 5. Obtain Customer Acceptance Sign-off
Delivery	To deploy the system (ISS standalone server) environment.	1. Software must be packed by following ISS's standard 2. Deployment guideline must be provided in ISS production (ISS standalone server) format 3. Production (ISS standalone server) support and troubleshooting process must be defined.

\

Appendix B: Mapped System Functionalities

Modular course	Description
Machine Reasoning	• **Knowledge Representation - Business rules** (e.g. If price pct% change > X = 'Buy'..) • **Analogical Reasoning**: The k-nearest neighbors (KNN) algorithm can be used for analogical reasoning by leveraging its ability to classify the price movement of the real time data feed, based on its similarity to historical trained data. • **Probabilistic Reasoning:** The Naive Bayes algorithm is used for probabilistic reasoning by calculating the probabilities of different target classes (buy/hold/neutral) given the sentiment features and selecting the target class with the highest posterior probability. • **Inductive Reasoning**: Decision trees use inductive reasoning to learn patterns and rules from specific observations in the training data. They generalize these patterns to make predictions or classifications of different target classes (buy/hold/neutral) for the real time data feed. • **Supervised Machine Learning**: Metrics such as accuracy score, F1-score are used to evaluate the performance of classification models. • **Document Database**: Data mining via API call from host sites. The data obtained are in the form of JSON format, which the team translated into dataframes in Python. • **Long term memory**: The trained model was dumped into a pickle file so that it can be utilized at any point of time without re-processing.
Reasoning System	• **Competing Experts system**: We use a series of NLP models with a range of classification models to get the best performance combination, voted by accuracy and F1 score. • **Similarity based Reasoning**: The model adopted by the team learns to look sentiment distribution characteristics to group them into target classification: "Buy", "Sell", "Hold"
Cognitive Systems	• **Natural Language Cognition**: The team utilizes pretrained LLM like distilroBERTa, finbert, which trains on multiple financial news articles to predict the sentiment on each new articles in 3 classes 'positive', 'negative' and 'neutral'.

Appendix C: Installation and User Guide

See attached.

Appendix D: Individual Project Report

Project Report: Tan Kok Tong, A0155101Y

(1) Personal contribution

- Researching in suitable data modelling method like API pull, i.e. Alpha vantage API for the collect proper traditional finance news with useful features like ticker filter, period and relevance score, etc
- Data modelling of JSON files into dataframe for python program processing for model training and testing
- Decision to perform a relevance score filtering to select more relevant news, based on justified research for the best performance models
- Researched in different NLP models for sentiment analysis, proposed a few financial-trained NLP models like yiyanghkust/finbert-tone and soleimanian/financial-roberta-large-sentiment
- Proposed to standardscalar of the training features and encoding of the target variables for train, test and split during machine learning model training. Experimented with different combinations of training features to find our best combination to be scaled sentiment scores and article counts.
- Tested a combination of yiyanghkust/finbert-tone and set of classification models like linearSVC, K-NN, Decision Tree, Random Forest, Kernel Approximation, and NavieBayes to find the best performance combination of NLP and classification model
- Performed hypertuning of the random forest model to get the best parameters with the highest accuracy and f1 score
- Optimized the stock price percentage change labelling for sentiment signal (Buy/Hold/Sell) that gives the highest accuracy and f1 score for the final model
- Researched into multiple research papers to justify the project report and decisions made during the model selection and training phase. Example like comparing between FinBERT and RoBERTa and financial domain related inferences

(2) What was learnt that was the most useful?

The whole process of building the MVP together with the team was extremely useful and eye-opening for me. It allows me to experience how a full-stack team would function. The most useful part was actually to discuss and brainstorm the knowledge modelling for our product, not only does it allow me to understand some financial domain knowledge, it also helps me in learning some data refinement knowledge for the next step of model training. The determination of NLP and machine learning classification models was also very useful. I have learnt some cognitive systems knowledge by researching on the NLP models and learning how to apply them to obtain the teams' desired output. The whole process of classification model training and testing allows me to grasp the machine learning knowledge and application.

(3) How can i apply knowledge and skills in other situations or workplaces

I believe the knowledge and skills that I have learnt in this project can be grouped by the modular courses. Machine Reasoning methods like Analogical Reasoning

and Knowledge Representation allows me to come out with logical and suitable categories or metrics in a given project or task. Reasoning System will allow me use methods like competing systems and computational theory like GAs for more complicated business problems that I want to solve in the future. Cognitive System definitely allows me to equip me with the knowledge to utilise the power of LLMs for many functions like, textual response outputs, intent classification and even image and sound processing that future complicated projects may require. Therefore, it opens up many possibility for me to tackle them. In my workplace, these knowledge that I have learnt can be applied in automation of my production plant. If sensors are sufficiently installed, data collection will be sufficient, which will allow me to apply business representation and analogical reasoning for knowledge modelling. The refined data can be used in Reasoning System to find similarity correlation between different parameters. Last but not least, the cognitive systems can definitely be used like an AI Agent for my operations team on resolving manufacturing plant issues and optimization ideas.

Project Report: Yeo Li Ying, A0292287R

(1) Personal contribution

- Research and experiment with various techniques for market data extraction such as API pull and web scraping to acquire news feed and stock price for traditional news media.

- Research and experiment with different sentiment analysis techniques such as nltk vader, which utilised a predefined dictionary of words and their associated sentiment scores, versus FinBERT which is a pre-trained transformer from huggingface specifically trained for the financial domain.

- Research and test the user interface and API call to capture the real time news data and price and integrate backend workflow to calculate sentiment scores and render the display of news data and corresponding calculated sentiment scores, stock price movement.

- Implement python training script to perform data preprocessing of raw JSON files to extract the relevant details like date and time of the publication, title of the article, ticker name. Implement sentiment analysis using the ProsusAI/FinBERT model using the historical data. Test and analyse the performance metrics with different classification models like linearSVC, K-NN, Decision Tree, Random Forest, Kernel Approximation, and NavieBayes to find the best performance classification model.

- Install, configure and test the end to end deployment using Docker on a Ubuntu environment to validate the package encapsulates all required Python libraries and dependencies.

- Write and review the project report, installation and user guide.

(2) What was learnt that was the most useful?

I gained an understanding of how to engineer a solution that leverages on market data API pull for retrieving stock news feed and price data and calculation of sentiment scores using pre-trained transformer models from huggingface. I also learnt how to render visualisation on the user interface using built in plot libraries to demonstrate the correlation between news sentiments and stock price performance. Additionally, the experience of training and testing various different classification models was instrumental in enhancing my grasp of machine learning concepts and their practical applications.

(3) How can i apply knowledge and skills in other situations or workplaces

Machine learning algorithms like KNN, decision tree, Naive Bayes, SVC can be used to predict trade settlement failure by analysing historical trades that encounter reconciliation and settlement failure and identifying various conditions (eg, wrong

PSET , exotic currencies, wrong reference) patterns that indicate a higher likelihood of failure. By leveraging these algorithms, business can proactively manage the risk of settlement failure and missed/delay payments which could lead to regulatory penalty to improve the overall efficiency and reliability of trade processing system.

Project Report: Ng Ying Wee, A0111692L

(1) Personal contribution

- Led the team in brainstorming on the concepts, flow and key milestones to hit every week.
- Plan and assign each member a scope to handle week-on-week. Helped to manage deliverable timelines and set goals for the team.
- Social News data: Research on reddit forums and understanding its data structure. Source for Reddit non-API data and learn to extract from .zst files. Apply data cleansing and feed into several NLP models to obtain sentiment results.
- Extracted finance stocks data via API and modified into usable format, on a day-on-day percentage price increase.
- Research on the different kinds of NLP models we can utilise and read-up on how the community have been growing in sentiment training.
- Construct a plan for the team to rigorously test the various kinds of NLP and classification models to determine the final selection combinations.
- Ran through a series of hyperparameter tunings and iterations on several different models like Decision Tree, Random Forest, KNN etc as part of the rigorous tests.
- Oversee the changes required for the user UI and suggest changes for better experience.
- Create a structure for report writing and provide overall guidance and review for reporting.
- Developed the 2 videos for product marketing and system design.

(2) What was learnt that was the most useful?

- I learned a lot more on how transformers work behind the scenes (the math and the structure) and how sentiment labelling works on articles. I also understand the birth of transformers was pivotal and an enabler into the creation of ChatGPT, Gemini and Grok today.

(3) How can i apply knowledge and skills in other situations or workplaces

- In my workplace Shopee, I lead a team to manage Policy, Automation, Logic and UIUX in the area of Returns and Refunds operations. I feel that sentiment analysis can be deployed to understand the degree of dissatisfaction of a buyer and the system or agents can judge and address the cases differently.
- Furthermore, we can also explore the area of accurate evidence detection by each return reason so that programmatically, cases can be concluded in a much quicker and cost effective manner.

Project Report: Lai Kah Hoe, A0292131N

(1) Personal contribution

- Research on various sources available for data.
- Gather data from yahoo API, Reddit, Alpha Vantage for model training.
- Setup and facilitate file server for dataset and Jupyter Notebook sharing.
- Preprocess data and model training.
- Experiment on various algorithms and capture results for analysis.
- Discussion with the team to conclude the best fit model.
- Implement and code the application to retrieve data from yahoo API, perform data transformation and invoke model for inference.
- Implement and code the web UI to display the result from the model.
- Implement Dockerfile and package the final program as container images, push the image to the docker registry.
- Create and configure server, install docker runtime and deploy the container image.

(2) What was learnt that was the most useful?

I learnt to use various python libraries during different stages of the project. I learnt basic libraries like pandas and numpy to perform feature engineering. I learnt to use the scikit-learn library to apply different algorithms for machine learning, technique to test against test data. I also learnt to load models from huggingface, taking the result and serving the model via API and web UI..

(3) How can i apply knowledge and skills in other situations or workplaces

Ability to source for data and model the training data help me to understand how sentiment analysis works in the financial sector. I can apply the similar technique in different domains. Working on time series data, natural language processing and sentiment data help me to have better decision making in advising clients on the proper approach for solutioning AI/ML project.