Project Final Report:

Deep Predictions: Harnessing Neural Networks for Stock Price Movement Forecasting

Estevao Bittencourt, Catherine Lennon, & Aidan Mitchell

Kennesaw State University

CS 7357: Neural Networks and Deep Learning

Md Abdullah Al Hafiz Kahn

April 30, 2024

Abstract

This study investigates the integration of Long Short-Term Memory (LSTM) and Feed Forward Neural Network (FFNN) architectures to forecast short-term market movements in the technology sector of the American financial markets. Given the sector's rapid innovation and voluminous data, our model aimed to outperform traditional linear financial forecasting methods. Utilizing a comprehensive dataset sourced from Yahoo Finance and Alpha Vantage, the project applied advanced neural network techniques and rigorous data preprocessing to capture both temporal and non-temporal market dynamics. The model demonstrated partial convergence in training with an implementation of early stopping to prevent overfitting, confirmed by validation loss trends and a confusion matrix analysis. However, the results also indicated challenges, particularly with the volatility and unpredictability of the technology sector, affecting predictive consistency. The study suggests a cautious optimism for the LSTM and FFNN combination in financial forecasting, recommending further model refinements and continuous validation against real-world financial data for future improvements.

I. INTRODUCTION

In the American financial markets, the ability to predict short-term movements presents a significant advantage for traders, investors, funds, and financial analysts. Leveraging advancements in neural network and deep learning architecture, we develop a predictive model capable of forecasting day-to-day market movements. The inherent volatility of the market, driven by systemic factors such as economic indicators, geopolitical events, and market sentiment, as well as idiosyncratic factors unique to individual companies, makes accurate forecasting a challenging yet crucial task. We have chosen to focus on the technology sector since it offers a wide breadth of data, large market caps, and thus stability. By focusing on the technology sector, this project seeks to navigate the complexities of widespread and economic market dynamics while offering idiosyncratic insights that are both precise and actionable.

Traditional financial models, primarily linear in nature, often fail to capture the nonlinear intricacies and the multifaceted relationships within the market data (Kumbure et al, 2022). These models struggle to adapt to the rapid pace of change in financial environments, leading to predictions that quickly become outdated or inaccurate. As the financial industry evolves, there is a pressing need for more advanced analytical methods that can handle the complexity and volatility of the market more effectively.

In recent years, advancements in neural networks and deep learning technologies have shown promising potential to revolutionize the field of financial analytics (Jiang et al, 2021). These technologies are adept at modeling complex, nonlinear relationships and can process vast datasets to uncover hidden patterns and insights. Neural networks, particularly deep learning architectures like Convolutional Neural Networks (CNNs), Feed Forward Neural Networks, and Long Short-Term Memory (LSTM) networks, have demonstrated their capability to enhance predictive accuracy and offer nuanced understandings of market dynamics.

Our research zeroes in on the challenge of predicting short-term market movements, with a focus on the technology sector (Thomas et al, 2017). The technology sector is particularly well-suited for neural network forecasting models due to its inherent characteristics and the nature of the data it produces. This sector is marked by high volatility and significant growth potential, fueled by continuous innovation, product developments, and shifts in consumer demand. These dynamics create complex patterns and relationships in market data, which neural networks are adept at capturing and analyzing due to their ability to model non-linear interactions.

Moreover, technology companies generate vast quantities of data, including metrics on user engagement, financial performance, and market trends. The substantial volume of data provides a rich training ground for neural networks, which can improve their predictive accuracy as the dataset size increases. This aspect is crucial because neural networks, particularly deep learning models, require large amounts of data to learn and identify the underlying patterns effectively.

The sector's complexity, with its myriad of influencing factors such as regulatory changes, technological breakthroughs, and competitive landscape shifts, presents an ideal scenario for neural networks. These models excel in understanding and forecasting outcomes in environments where there are complex relationships and interdependencies(Mustafa et al, 2019). The fast-paced nature of technological innovation means market conditions and company fortunes can change rapidly, necessitating adaptive and flexible forecasting models like neural networks that can quickly incorporate and respond to new information and trends.

Furthermore, the global impact and connectivity of the technology sector mean that events in one region can have significant repercussions worldwide, affecting companies and markets across the globe. Neural networks are capable of integrating and analyzing data from multiple sources and regions, capturing the sector's global dynamics and interconnections. Additionally, the specific performance indicators unique to technology companies, such as user growth rates, subscription numbers, or cloud service revenues, can be effectively integrated into neural network models. This sector-specific focus enhances the models' relevance and accuracy in forecasting market movements.

In essence, the dynamic and data-rich environment of the technology sector, combined with its complex and fast-evolving market conditions, makes it an ideal candidate for applying neural network forecasting models. These models can leverage the sector's characteristics to deliver accurate, nuanced, and actionable insights into market trends and company performance.

The motivation behind this project is twofold. Firstly, there is a significant gap in the application of advanced neural network architectures for short-term market predictions in specific sectors. Secondly, the potential for these technologies to significantly enhance the predictive accuracy of financial models presents an opportunity for both theoretical advancement and practical application in financial decision-making.

Our contribution through this project will be the development of a predictive model that integrates advanced neural network architectures and deep learning methodologies to forecast short-term market movements. By employing a Feed Forward Neural Network our model will be designed to process and analyze large datasets from various sources, including Yahoo Finance and Alpha Vantage. This approach will allow us to capture both systemic and idiosyncratic factors affecting the market, providing a holistic view of the influences on short-term market dynamics.

In conclusion, this project aims to not only enhance the predictive accuracy of financial market movements but also contribute to the broader field of financial analytics by demonstrating the application of advanced neural network and deep learning technologies. Through this research, we aspire to provide actionable insights and develop a model that can adapt to the evolving nature of financial markets, thereby supporting more informed and effective financial decision-making.

II. RELATED WORK

The application of neural networks and machine learning techniques to financial forecasting has garnered significant interest, with various approaches tested across different financial markets and sectors. This review synthesizes findings from twelve studies, highlighting methodologies, performances, and limitations, while proposing a novel approach using a Feedforward Neural Network (FFNN) for the technology sector's financial forecasting.

Overview of Existing Approaches

In the evolving landscape of financial forecasting, neural networks have emerged as a pivotal tool, bridging the gap between historical data analysis and predictive accuracy. Maciel and Ballini's research underscores this evolution by demonstrating the superior accuracy of Artificial Neural Networks (ANNs) over traditional models like the Generalized Autoregressive Conditional Heteroscedasticity (GARCH), a statistical model used for stock market volatility forecasting. Their work, focused on stock markets in North America, Europe, and Brazil, showcases ANNs' capability to adapt to different financial environments (Maciel & Ballini). The journey into neural network efficacy extends into systematic reviews, as seen in the work of Ge et al., who dissected the performance and methodologies of neural network-based financial volatility forecasting. This meta-analysis highlighted the diverse and often disjointed approaches within the field, signaling a need for standardized evaluation frameworks to better understand neural network potentials in financial applications (Ge et al.).

Exploring specific neural network architectures, Lu et al. introduced a model combining Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks. This CNN-LSTM model, tailored for stock price forecasting, outperformed traditional and more simplistic neural network models by capturing both spatial features through CNNs and temporal dependencies through LSTMs, reflecting the dynamic nature of financial markets (Lu et al.). The international perspective of neural network applications in finance was further expanded by Chen, Leung, and Daouk, who applied a probabilistic neural network (PNN) to the Taiwan Stock Index. The PNN model, known for its pattern recognition capabilities, demonstrated strong performance in an emerging market context, offering insights into the adaptability of neural network models across different market dynamics (Chen, Leung, & Daouk, n.d.).

The narrative of neural networks in financial forecasting is enriched by comparative studies like that of Dunis and Jalilov, who evaluated Neural Network Regression (NNR) against more straightforward forecasting methods. Their analysis revealed that NNR models not only enhance forecasting accuracy but also improve trading performance, highlighting the advanced capabilities of neural networks in deciphering complex financial data patterns (Dunis & Jalilov). The exploration of neural networks' potential is not without its challenges, as illustrated by Kwong's study on stock market returns. Here, the struggle to surpass the predictive accuracy of a naive predictor model sheds light on the limitations and areas for improvement in neural network forecasting (Kwong).

Walczak's investigation into the impact of training data size on neural network performance introduces the "Time Series Recency Effect," positing that recent data yields better forecasting results. This finding points to the critical role of data relevance and timeliness in neural network training for financial forecasting (Walczak, n.d.). The diverse array of research, from systematized reviews to empirical studies, paints a comprehensive picture of the current state of neural network applications in financial forecasting. Each study contributes to the narrative, illustrating the nuanced and multifaceted nature of neural network modeling in financial markets, paving the way for future innovations in the sector.

Limitations of Existing Approaches

Common limitations across these studies include the challenge of model overfitting, the need for extensive data preprocessing, and the difficulty in generalizing findings across different financial markets or sectors. Additionally, many studies rely on historical data, which may not fully capture future market dynamics due to unforeseen economic events or trends.

Comparative Analysis with Feedforward Neural Networks

Feedforward Neural Networks (FFNNs), with their straightforward architecture, offer a robust alternative for financial forecasting. Unlike more complex models like RNNs or CNNs, FFNNs can mitigate overfitting through regularization techniques and simpler structures, making them efficient for capturing underlying market trends without excessive computational demands. However, a base FFNN might lack in capturing temporal dependencies in time series data, which is crucial for accurate financial forecasting, and we thus incorporate long short term memory (LSTM) architecture into our model to compensate for this.

Novel Approach for Technology Sector Forecasting

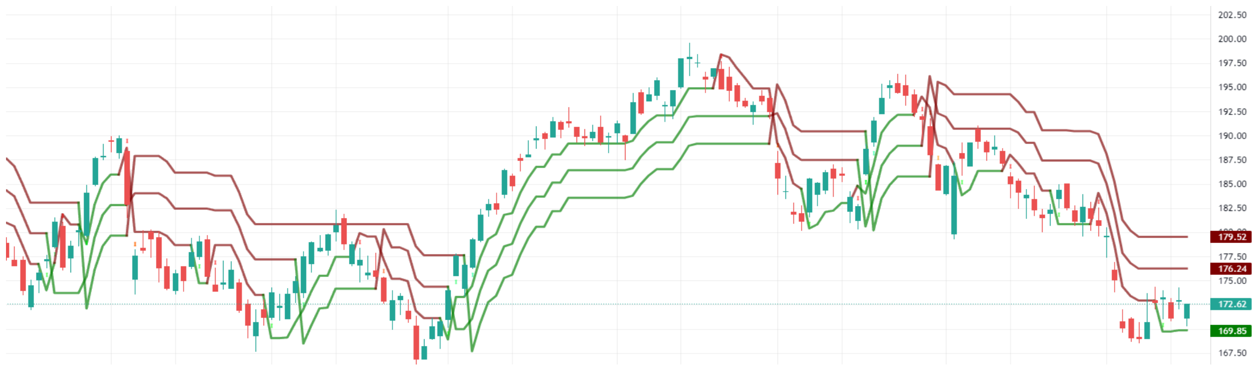
For the technology sector, where rapid innovation and market dynamics play significant roles, a hybrid FFNN model could be developed. This model would combine the strengths of FFNNs in handling non-temporal data with elements of time series analysis, perhaps through feature engineering or integration with time-sensitive layers, to better predict market movements. Incorporating sector-specific indicators, such as patent filings, product launch cycles, and R&D spending, into the FFNN could address the limitations of traditional financial forecasting models and provide more accurate predictions for the technology sector.

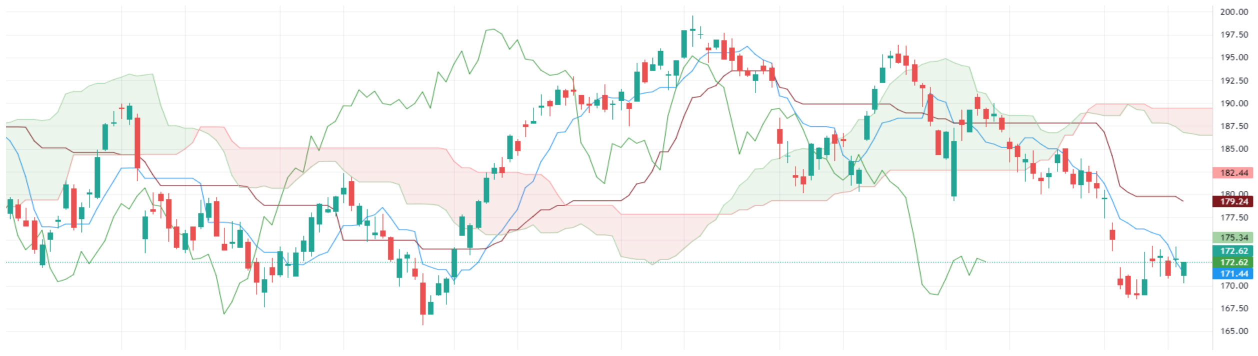
III. METHODOLOGY

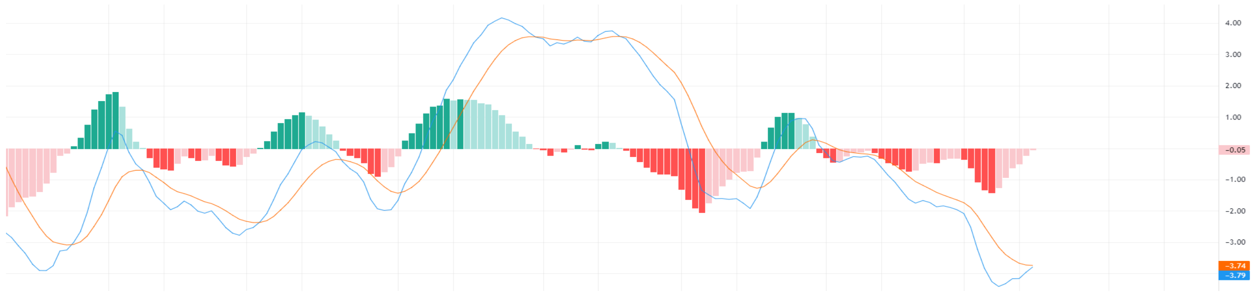
a. Data Preprocessing and Feature Engineering

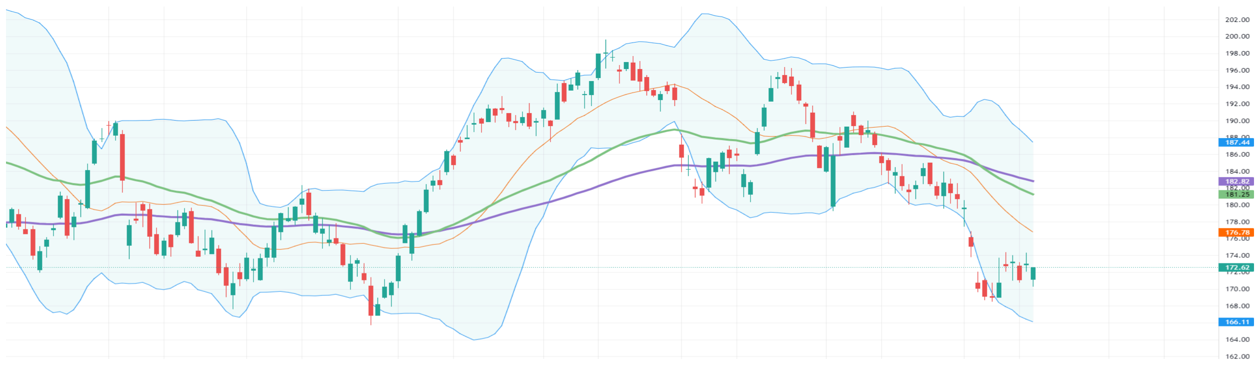
The implementation began with the gathering and preprocessing of a dataset focused on Apple Inc. (AAPL) stock, covering a period of 10 years with daily trading data, which fetched historical price information such as opening, closing, high, and low prices, volume, and stock splits. Subsequently, various technical indicators commonly used in financial trading were computed and appended to the dataset using the 'pandas_ta' library. These indicators included the Relative Strength Index (RSI), Exponential Moving Averages (EMA) of various lengths, Commodity Channel Index (CCI), Volume Weighted Average Price (VWAP), Moving Average Convergence Divergence (MACD), Stochastic RSI, and many more. The idea behind providing technical indicators to the model is that the technical indicators generate new non-linear features combined from other features, and these indicators are used by other traders, which may allow the model to learn how the market will fluctuate based on other traders. For example, markets may fluctuate greatly based on a certain indicator because large institutional investment holders or many smaller investors trade using that indicator, thus making it a good idea to provide the model with a chance to compete with these traders with some insight into their strategies.

Below are some examples of trading charts with some technical analysis trading indicators added to give an idea of what these generated features might look like.


Triple SuperTrend Technical Indicator


Ichimoku Cloud Technical Indicator


MACD Technical Indicator


Double EMA & Bollinger Bands Technical Indicators

b. Feature Standardization and Selection

We employed standardizing scalars to normalize the dataset, subtracting the mean and dividing by the standard deviation for each feature to ensure all input features contributed equally to the analysis, avoiding any bias towards variables with higher magnitudes. Additionally, to manage outliers and skewed distributions, we applied quantile transformations which reshaped the data distribution to follow a uniform or normal distribution, effectively mitigating outlier effects. While the logarithmic transform could stabilize variance and normalize data with exponential growth patterns or heavy-tailed distributions, we ultimately decided not to use it for our final model because the features it generated were not deemed effective by the feature reduction techniques that we used.

Initially, we experimented with Recursive feature elimination with cross-validation (RFECV) using Random Forest Regressors, as well as Random Forest Classifier algorithms as feature selection tools, aiming to leverage their ability to rank the importance of various features through the construction of decision trees. However, despite their potential to reduce the impurity of splits, we ultimately found that it did not provide the expected benefits in our specific context. Consequently, we decided not to include it in the final model. Instead, we focused on Principal Component Analysis (PCA) to manage the high dimensionality of our dataset. PCA effectively reduced the dataset's complexity by transforming it into a set of uncorrelated variables, retaining only those components that accounted for 99% of the variance. This method enhanced computational efficiency and model focus, enabling more effective training and reducing the risk of overfitting.

c. Model Selection

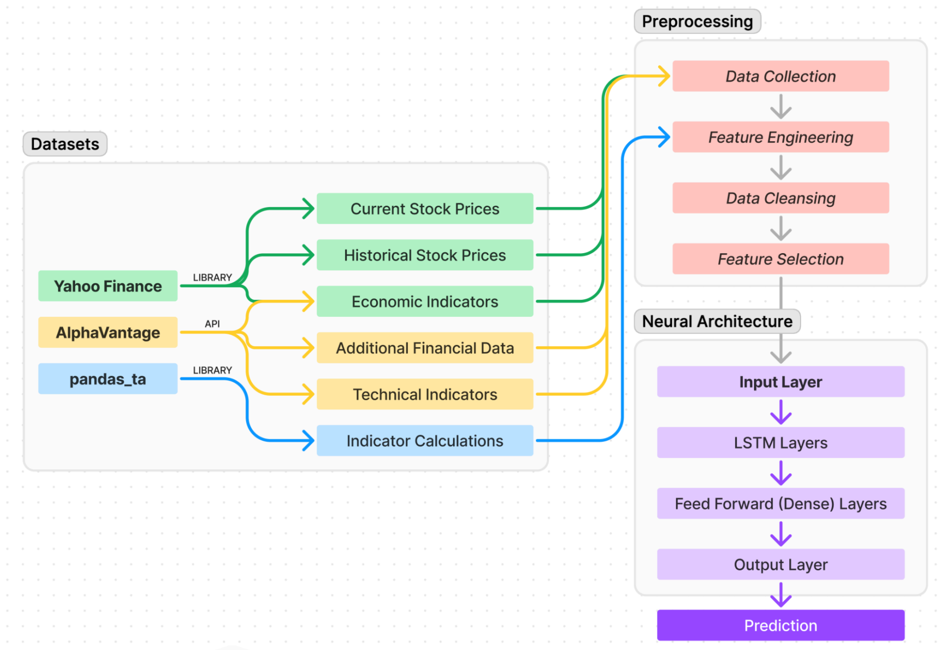
For the model architecture, various configurations were explored to determine the most effective structure for predicting stock price movements based on time series data:

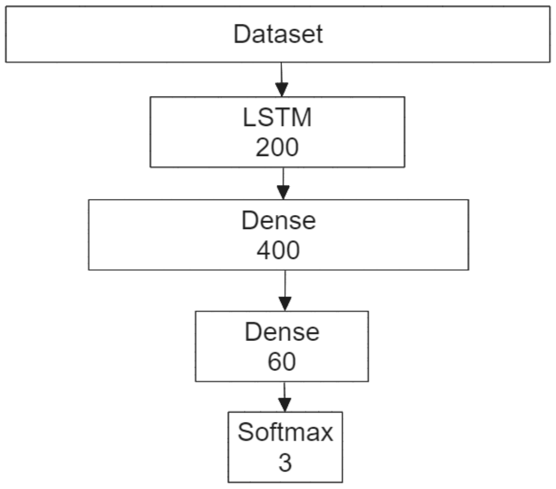
- Base Pure LSTM: This was the model included in the final iteration. It utilizes an LSTM layer to capture temporal trends and dependencies and extract key features over time. This layer then fed into a multilayer dense structure to further allow the model to classify the samples in the datasets.
- Smaller Pure LSTM: A more compact version of the LSTM network aiming to reduce complexity and overfitting present in the base model while still capturing essential time series trends.
- Pure Convolution: Employed convolutional layers to extract features from structured time series data without the use of recurrent structures, focusing on local feature detection.
- Convolution then LSTM: Combined convolutional layers upfront to extract and compress spatial features from the data, followed by LSTM layers to analyze temporal dependencies from the convolved features.
- Mixed Convolution and LSTM: Integrated both convolutional and LSTM layers in a mixed approach, allowing the model to simultaneously process spatial and temporal features effectively.
- RNN: Standard recurrent neural network layers were tested to benchmark their performance against more complex architectures.

- Transformer: Incorporated Transformer architecture, utilizing attention mechanisms to handle dependencies in the data, which allows the model to focus on important time steps dynamically.
- Simultaneous Convolution and LSTM: Configured to run convolution and LSTM operations in parallel to independently process and then merge spatial and temporal insights, enhancing the model's ability to discern relevant patterns.
- Grid Search for Model Configuration: Conducted extensive testing with 224 different models, exploring a variety of configurations and parameters to identify the optimal architecture. This approach provided a comprehensive understanding of how different models and settings impacted predictive accuracy.

d. Model Architecture

For the final model architecture, the model begins with several LSTM layers capturing the temporal trends and dependencies within the data, which then feeds into a dense fully-connected feed-forward network. The idea is that the LSTM structure will condense the information from the dataset into useful information pertaining to time series trends in the data for each feature. The data fed to the LSTM structure will be a sliding window of the past x days of data for the asset in question. From there, the LSTM layer's output can be used as the input for a normal dense feed-forward neural network that will be able to determine likelihoods of various price movements given the information condensed by the LSTM structure. The output would be the likelihood of the price of the stock moving to each of several percent changes.





e. Loss Functions

Various loss functions were explored with the end goal of optimizing performance, particularly with respect to precision. Initial trials included Categorical Cross Entropy, Weighted Categorical Cross Entropy, and Doubly Weighted Categorical Cross Entropy, each adding a level of complexity in handling class imbalance by assigning different weights to the classes. Additionally, we tested Focal Loss, designed to focus learning on hard examples and mitigate the overwhelming influence of easy negatives in the training set. Despite these explorations, these loss functions did not align with our strategic goals, primarily because they did not adequately prioritize the precision of predicting positive and negative market movements over the recall or overall accuracy. This precision is critical in financial applications where the cost of false positives—incorrect predictions of market movement—can be economically significant.

Therefore, we opted for Doubly Weighted Focal Loss, which refined the model's focus, ensuring it concentrated on cases where it was highly confident and cases where the consequences if wrong were more significant. This approach is based on the principle that "if you make a guess, it better be right," which emphasizes our operational strategy to engage in trades only when the model exhibits high confidence in the prediction of a market change.

f. *Training Stages*

With the end goal of achieving a more precise and accurate result, we manually iterated through training parameters using a variety of loss functions. We additionally adjusted the balance of class samples and varied the training epochs, all with the goal of achieving a more stable and robust model across different settings. This training on different architectures helped us to optimize the parameters and improve our accuracy.

IV. EXPERIMENTAL SETUP

a. *Implementation*

The dataset of ten years of daily trading data for Apple Inc. (AAPL) was pulled using the 'pandas_ta' library. We enriched this data with various technical indicators like Relative Strength Index (RSI), Exponential Moving Averages (EMA) of various lengths, Commodity Channel Index (CCI), and others by calculating them directly from the stock prices. These indicators capture market trends and volatility, and our intention was to provide the network with predictive signals about future price movements. We normalized this data using the StandardScaler to ensure that all features contributed equally to the model without bias towards variables with higher magnitude. We also applied the QuantileTransformer to manage outliers effectively, reshaping the distribution of features to follow a more uniform or normal distribution, which is beneficial for training stability.

We implemented multiple LSTM (Long Short-Term Memory) layers to capture temporal dependencies within the data. LSTM structures were chosen for their ability to model temporally dependent data, as this makes them well-suited for financial data like stock prices, which are inherently sequential. We then compiled and trained the network using the Adam optimizer, and custom loss functions were implemented to focus the training on specific aspects of the prediction task, such as precision in financial scenarios where the cost of incorrect predictions can be significant.

Throughout the development of this project, we conducted numerous testing iterations to refine the network architecture and optimize parameters to identify the most effective configurations that balance complexity and performance. Each iteration involved adjusting layers, neurons, rates, and learning parameters within the neural network model. These changes were methodically tested to evaluate their impact on the model's predictive accuracy.

All computational tasks, including model training, validation, and testing, were executed on a local machine:

- CPU: AMD Ryzen 9 7900X (12-Core, 24 Logical Processors, 5.2 GHz)
- Memory: 32GB 6000MHz dual-dim
- GPU: NVIDIA RTX 3060Ti 8Gb

b. *Data Sources*

In our model, both Alpha Vantage and YFinance are pivotal in sourcing the financial data essential for our market analysis, investment decision-making processes, and the development of algorithmic trading strategies. Their integration into our model provides a robust foundation for accessing and analyzing financial market data, underscoring their significance in the realm of financial research.

Alpha Vantage

Alpha Vantage represents a comprehensive source of APIs that facilitate access to both historical and real-time financial data, encompassing stock prices, foreign exchange rates, and cryptocurrency information. The platform is renowned for its extensive provision of financial metrics and analytical indicators, rendering it a valuable asset for financial analysts and developers within the economic sector. It furnishes users with a plethora of data types, including time series, technical indicators, and fundamental corporate data. The acclaim for Alpha Vantage is attributed to its straightforward integration, the broad spectrum of financial data it encompasses, and its capabilities in delivering extensive historical financial datasets alongside real-time data, which are pivotal for conducting thorough financial analyses and supporting algorithmic trading endeavors.

YFinance

YFinance, colloquially recognized as the unofficial API of Yahoo Finance, is a Python library designed to facilitate the acquisition of historical market data from Yahoo Finance. Despite the absence of an official API from Yahoo Finance for data extraction, YFinance addresses this gap by scraping data from the website and structuring it into an accessible format. This library is extensively utilized for the procurement of historical stock price information, financial summaries, and other market-related data that are crucial for investors and financial analysts. The utility of YFinance is underscored by its user-friendly interface and the comprehensive scope of data it can retrieve, such as stock prices, dividends, and financial metrics, thereby serving as an indispensable tool for conducting financial analysis and formulating investment strategies.

The neural framework integrates insights from both systematic and idiosyncratic factors through a multi-input model where different types of data are processed in parallel before being combined to make a prediction.

Systematic factors include market-wide influences such as changes in interest rates, economic growth indicators, geopolitical events. These impact sector-specific market movements.

- This exploration involves analyzing historical data to identify patterns and correlations between these systemic factors and market behavior.
- Techniques such as feature importance analysis in machine learning models will be employed to quantify the impact of systemic factors on stock prices.

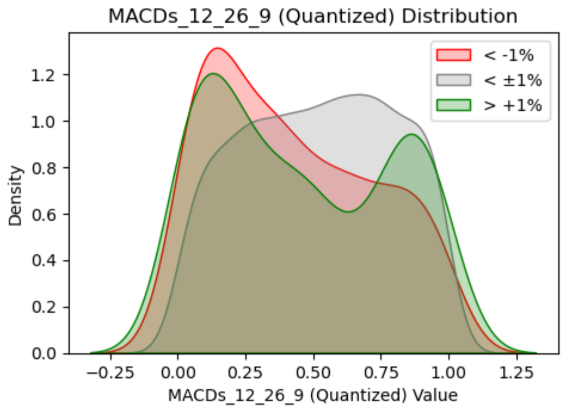
Idiosyncratic factors include company-specific factors such as earnings reports, dividend announcements, management changes, and launches. Strategy may include sentiment analysis on news articles and financial reports, alongside time series analysis of financial metrics.

Thus, from AlphaVantage and YFinance, we acquire the following data points:

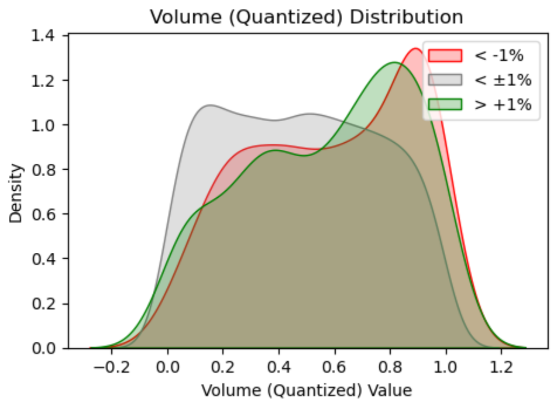
- Historical Stock Prices: Data from sources like Yahoo Finance, which provides daily stock prices, trading volumes, and other market-related data for technology companies. This dataset will form the base of our analysis, allowing us to study price movements and trading patterns.
- Economic Indicators: Datasets from platforms like the Federal Reserve Economic Data (FRED through AlphaVantage) that include broader economic indicators such as interest rates, inflation rates, and employment figures, which affect the market conditions and investor sentiment in the technology sector:
    - Forex rates (e.g., EUR, CNY, JPY)
    - Commodity prices (e.g., copper, aluminum, crude oil, natural gas)
    - Key economic figures like inflation rates, CPI, retail sales, durable goods orders, unemployment rates, non-farm payroll data, and different Treasury yield rates.
- Financial Statements: Quarterly and annual financial reports from the companies, available on financial databases like Quandl or directly from the companies' investor relations websites. These documents provide insights into the companies' financial health, including revenue, expenses, profits, and cash flows.

To help further our understanding of the dataset we used multi-class distribution graphs for each feature to give a better representation of how different features may help the model classify samples.

For example, the following plot shows the distributions of the dataset with each class being a different color. With this graph we can see that if a sample had a value of about 0.75 to 1.00 for this feature, the sample was more likely to be a neutral or positive case than a negative case.



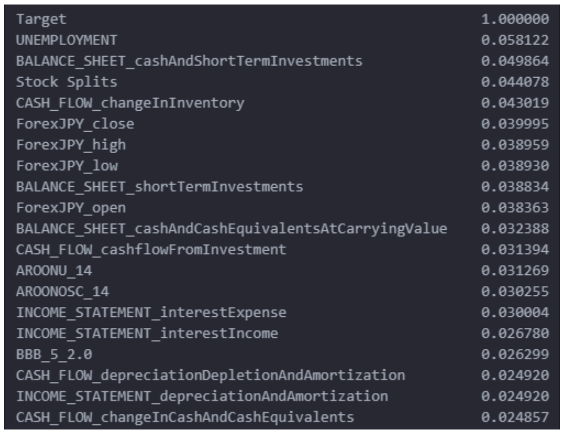
Another example would be this plot, which demonstrates that if a sample had a value of about 0.60 to 1.10 for this feature then the sample would more likely be a positive or negative case than a neutral case.



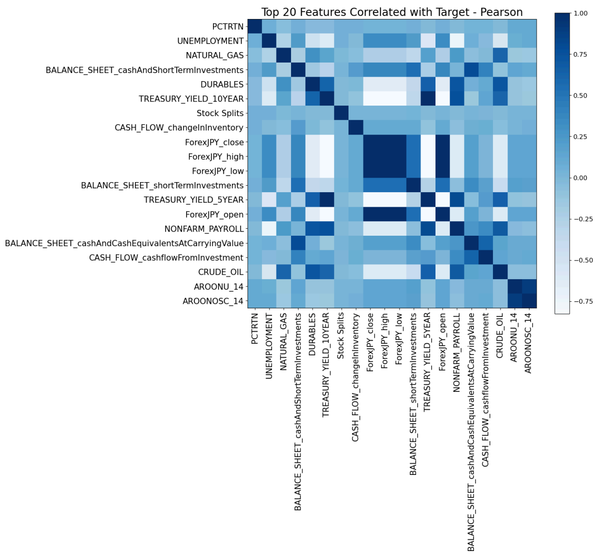
With many combinations of features like this over time a model can begin to predict more accurately what class a sample falls under.

In order to reduce the number of features provided to the model to save the model from overfitting or using poor features, we used several methods to determine the fit of a feature. First we tried using pure Pearson or Spearman correlation, but both of these proved to provide poor feature reduction.

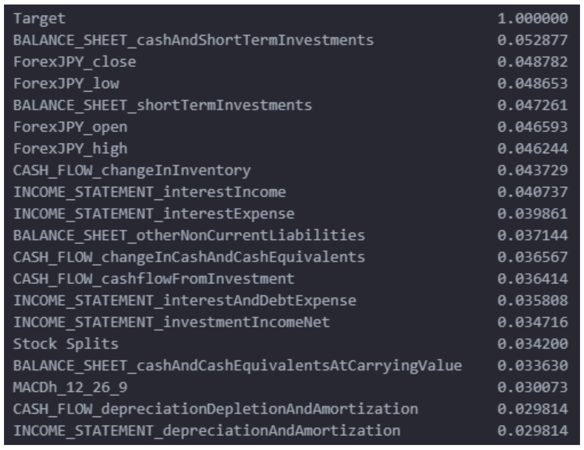
Here is the correlation for Pearson:



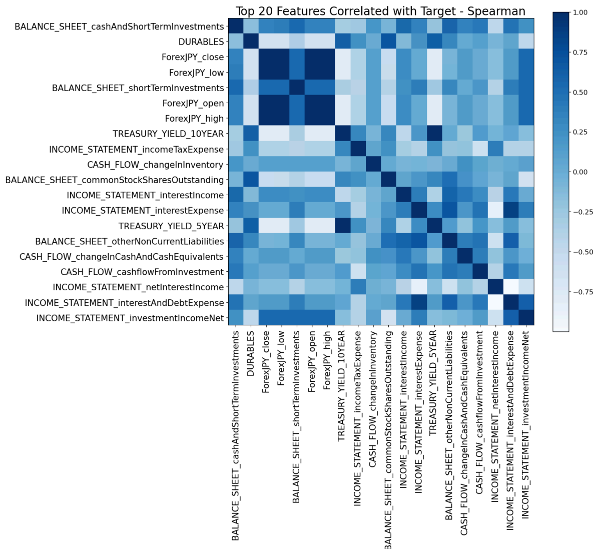
```
Target                                              1.000000
UNEMPLOYMENT                                        0.058122
BALANCE_SHEET_cashAndShortTermInvestments           0.049864
Stock Splits                                        0.044078
CASH_FLOW_changeInInventory                         0.043019
ForexJPY_close                                      0.039995
ForexJPY_high                                       0.038959
ForexJPY_low                                        0.038930
BALANCE_SHEET_shortTermInvestments                  0.038834
ForexJPY_open                                       0.038363
BALANCE_SHEET_cashAndCashEquivalentsAtCarryingValue 0.032388
CASH_FLOW_cashflowFromInvestment                    0.031394
AROONU_14                                           0.031269
AROONOSC_14                                         0.030255
INCOME_STATEMENT_interestExpense                    0.030004
INCOME_STATEMENT_interestIncome                     0.026780
BBB_5_2.0                                           0.026299
CASH_FLOW_depreciationDepletionAndAmortization      0.024920
INCOME_STATEMENT_depreciationAndAmortization        0.024920
CASH_FLOW_changeInCashAndCashEquivalents            0.024857
```



And here is the Spearman correlation:

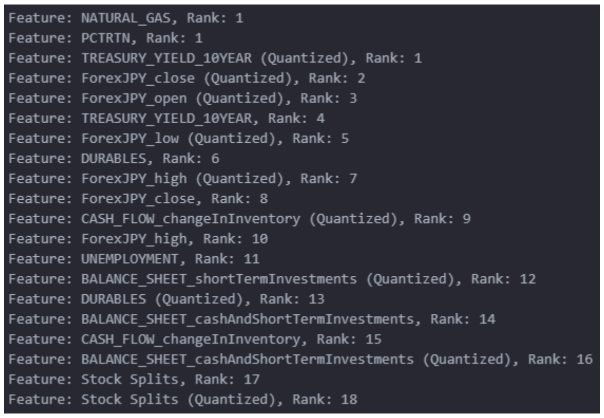


```
Target                                              1.000000
BALANCE_SHEET_cashAndShortTermInvestments           0.052877
ForexJPY_close                                      0.048782
ForexJPY_low                                        0.048653
BALANCE_SHEET_shortTermInvestments                  0.047261
ForexJPY_open                                       0.046593
ForexJPY_high                                       0.046244
CASH_FLOW_changeInInventory                         0.043729
INCOME_STATEMENT_interestIncome                     0.040737
INCOME_STATEMENT_interestExpense                    0.039861
BALANCE_SHEET_otherNonCurrentLiabilities            0.037144
CASH_FLOW_changeInCashAndCashEquivalents            0.036567
CASH_FLOW_cashflowFromInvestment                    0.036414
INCOME_STATEMENT_interestAndDebtExpense             0.035808
INCOME_STATEMENT_investmentIncomeNet                0.034716
Stock Splits                                        0.034200
BALANCE_SHEET_cashAndCashEquivalentsAtCarryingValue 0.033630
MACDh_12_26_9                                       0.030073
CASH_FLOW_depreciationDepletionAndAmortization      0.029814
INCOME_STATEMENT_depreciationAndAmortization        0.029814
```

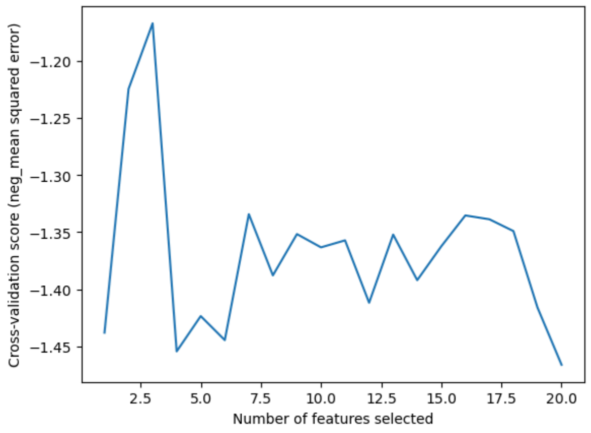


As you can probably see the results of these correlation tables were not confidence-inspiring, and so we tried using RFECV (Recursive feature elimination with cross-validation) to select several of the best features, whose output can be seen below:

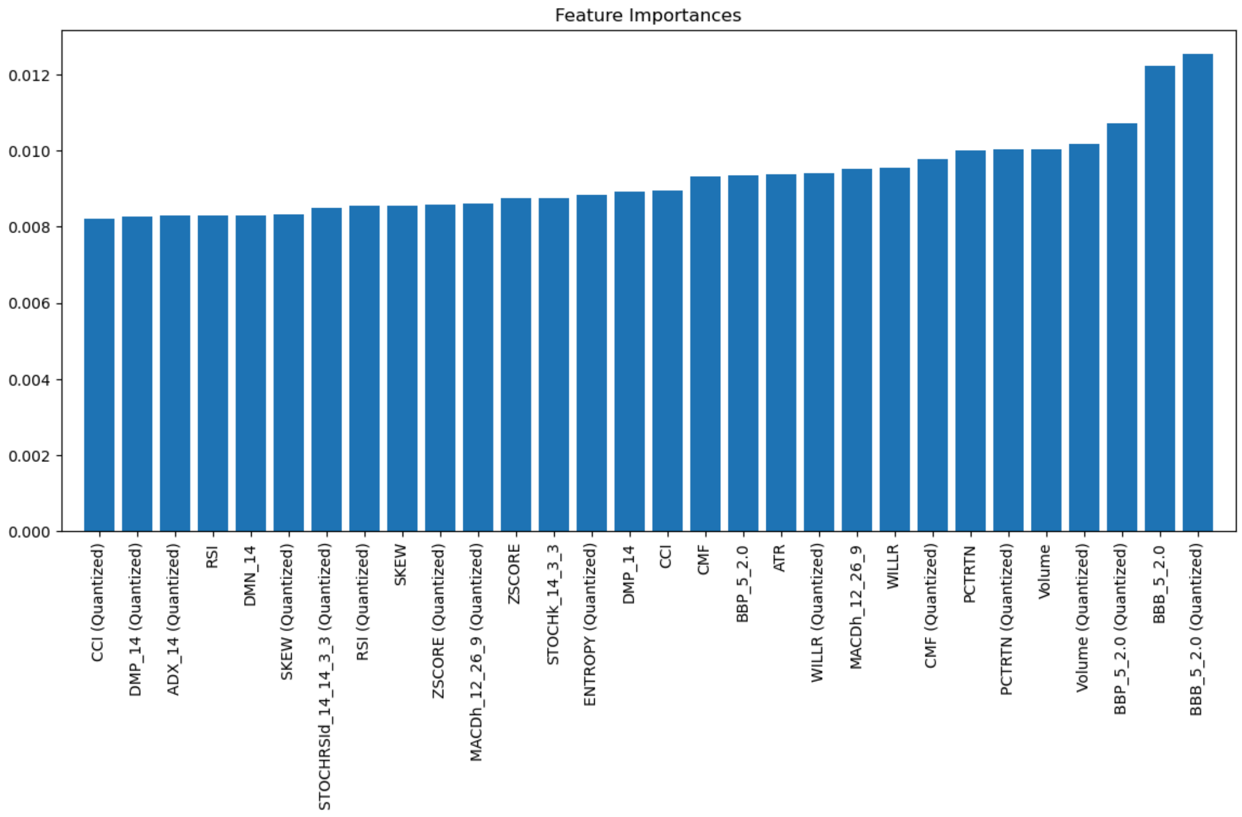
Recursive feature elimination with cross-validation results:



```
Feature: NATURAL_GAS, Rank: 1
Feature: PCTRTN, Rank: 1
Feature: TREASURY_YIELD_10YEAR (Quantized), Rank: 1
Feature: ForexJPY_close (Quantized), Rank: 2
Feature: ForexJPY_open (Quantized), Rank: 3
Feature: TREASURY_YIELD_10YEAR, Rank: 4
Feature: ForexJPY_low (Quantized), Rank: 5
Feature: DURABLES, Rank: 6
Feature: ForexJPY_high (Quantized), Rank: 7
Feature: ForexJPY_close, Rank: 8
Feature: CASH_FLOW_changeInInventory (Quantized), Rank: 9
Feature: ForexJPY_high, Rank: 10
Feature: UNEMPLOYMENT, Rank: 11
Feature: BALANCE_SHEET_shortTermInvestments (Quantized), Rank: 12
Feature: DURABLES (Quantized), Rank: 13
Feature: BALANCE_SHEET_cashAndShortTermInvestments, Rank: 14
Feature: CASH_FLOW_changeInInventory, Rank: 15
Feature: BALANCE_SHEET_cashAndShortTermInvestments (Quantized), Rank: 16
Feature: Stock Splits, Rank: 17
Feature: Stock Splits (Quantized), Rank: 18
```

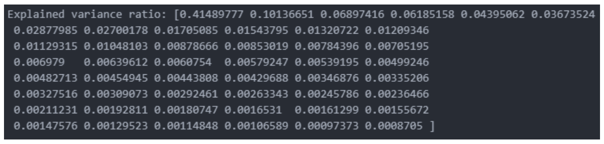


After this we tried a Random Forest Classifier as well:

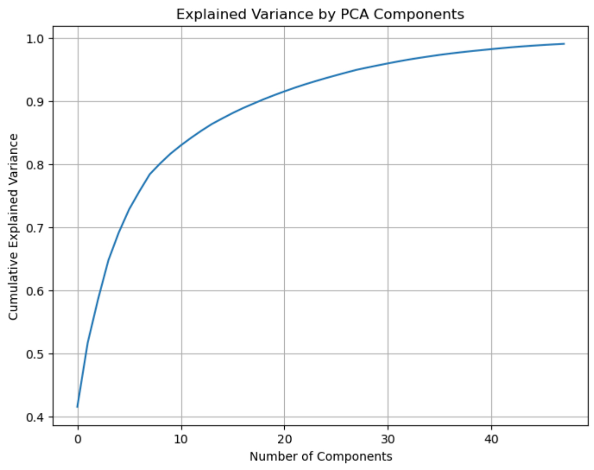


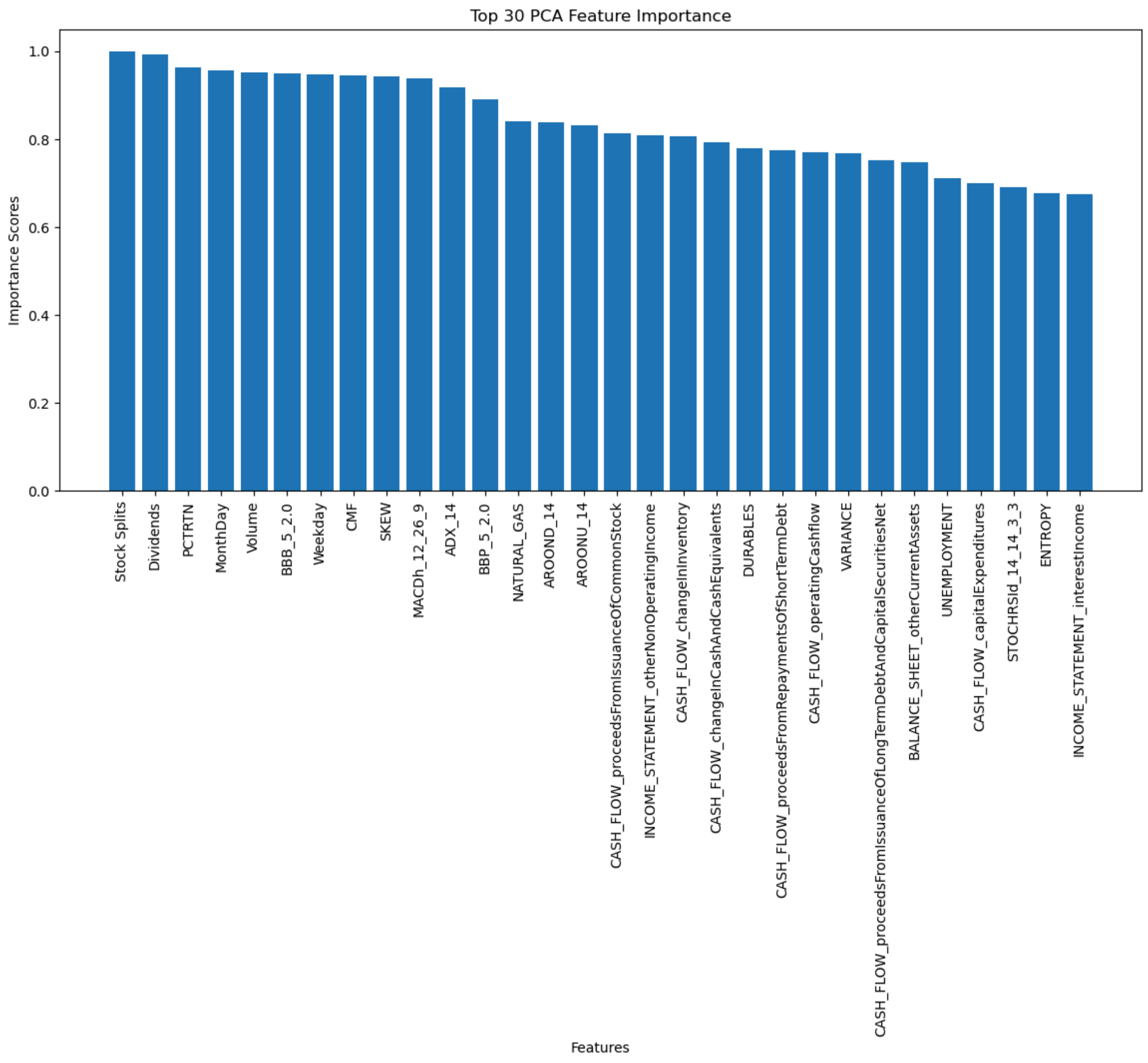
But both of these techniques resulted in poor reduced feature sets when compared to principal component analysis:

Principal Component Analysis results:

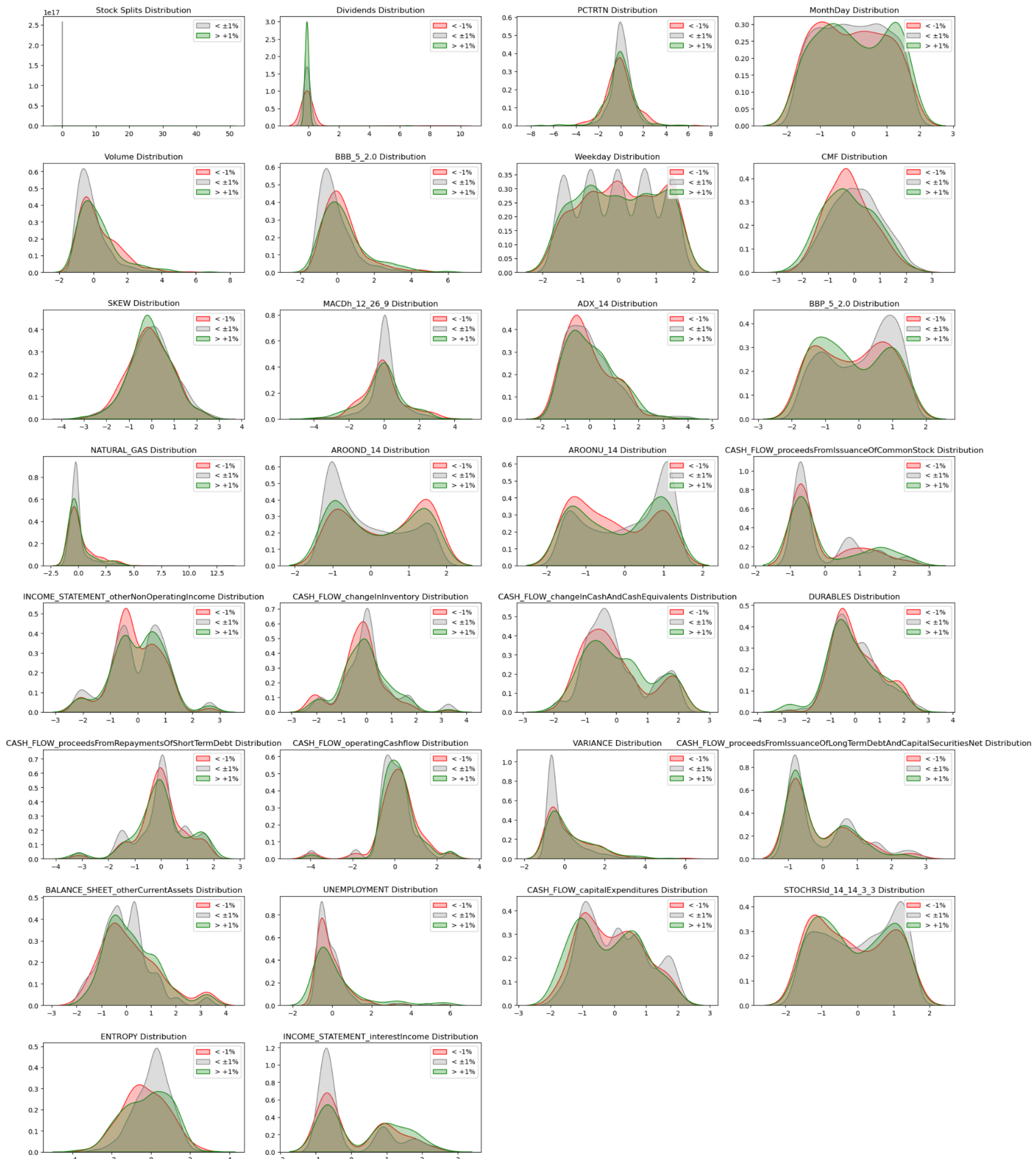


```
Explained variance ratio: [0.41489777 0.10136651 0.06897416 0.06185158 0.04395062 0.03673524
 0.02877985 0.02700178 0.01705085 0.01543795 0.01320722 0.01209346
 0.01129315 0.01048103 0.00878666 0.00853019 0.00784396 0.00705195
 0.006979   0.00639612 0.0060754  0.00579247 0.00539195 0.00499246
 0.00482713 0.00454945 0.00443808 0.00429688 0.00346876 0.00335206
 0.00327516 0.00309073 0.00292461 0.00263343 0.00245786 0.00236466
 0.00211231 0.00192811 0.00180747 0.0016531  0.00161299 0.00155672
 0.00147576 0.00129523 0.00114848 0.00106589 0.00097373 0.0008705 ]
```

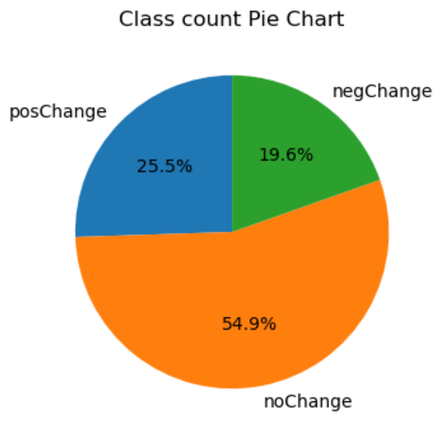




Thus with feature reduction completed, here were the dataset distributions for all of the final features selected:

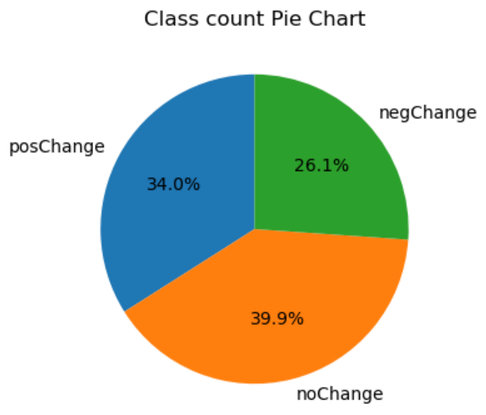


But there was still an issue left, which was the imbalance in class distributions, which caused the model to prefer predicting the neutral output far more than any other class. This imbalance can be seen in the pie chart below:



To counteract the skew from an excessive number of neutral cases in our dataset, we adjusted the class distribution by reducing these instances. This was essential to prevent the model from favoring neutral predictions disproportionately and to enhance its ability to accurately detect significant positive or negative market movements.

Below is the class distribution after performing these reductions:



Lastly, we needed the ability to better teach the model what we wanted for it to do. First we started with categorical cross-entropy loss, but because of the difficulty of this classification task the models would fail to converge and produce usable results because of the disparity in importance of the classes and the number of samples of each class type. To solve this issue, we developed several loss functions to combat this, but ultimately settled on a modified focal loss function, which introduced categorical multi-class cross-entropy and custom weighting for the different possible model predictions and ground truths.

This modified function allowed us to weigh different predictions and ground truth cases differently. For example, we weighed cases where the model predicted a positive case but the true case was negative as being 10x more important than if the model predicted a positive case and it was truly a positive case. This allowed us to get the model to learn that we cared more about precision than recall, and we cared more about precision and recall with the positive and negative cases than we cared about the precision and recall with neutral cases.

Specifically, the breakdown of how this function works is below:

1. Cross Entropy Component

$$CE(y_{\text{true}}, y_{\text{pred}}) = -\sum_{i=1}^{C} y_{\text{true},i} \log(y_{\text{pred},i})$$

2. Predicted class probabilities

$$p_t = \sum_{i=1}^{C} y_{\text{true},i} \times y_{\text{pred},i}$$

3. Focal Loss Modulating Factor

$$\text{modulating factor} = (1 - p_t)^{\gamma}$$

4. Class Weights Application

$$\text{weight factor} = \text{einsum}('ij, bi- > b', \text{weights}, y_{\text{true}})$$

5. Weighted Focal Loss

$$\text{weighted focal loss} = \text{weight factor} \times \sum_{i=1}^{C} (\text{modulating factor} \times CE_i)$$

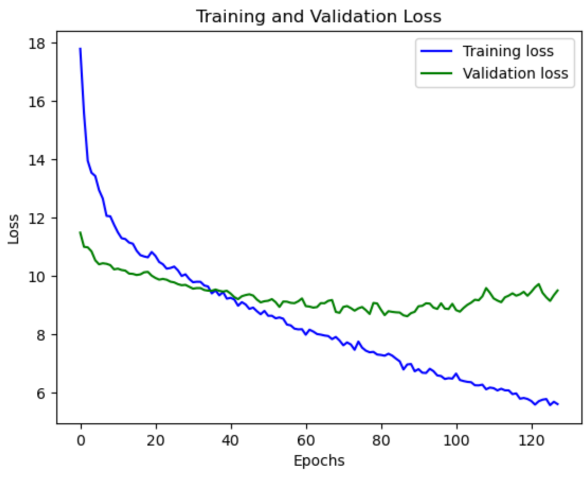
V. RESULTS

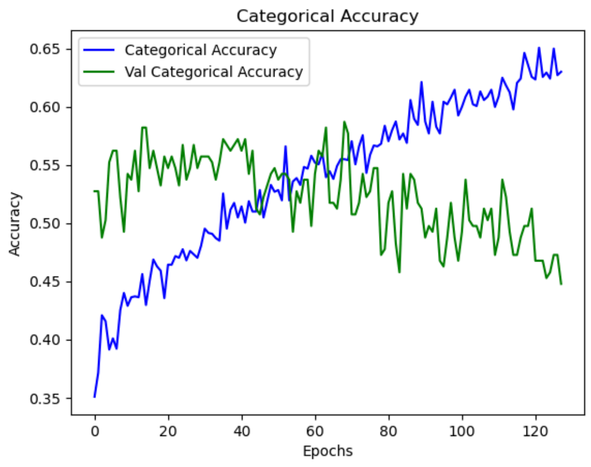
The results from our primitive model showed that predicting stock prices using data from a recent past can be tricky and cumbersome. Although data collected from the past can be useful for anticipating the future, it is not as reliable and realistic due to many reasons, political, geographic, and socioeconomic changes can lead you to false positives when making predictions that do not account for the global incidents. Even having a reliable dataset, free from biases and well diversified, your predictions may be off by as much as 50% plus or minus. When training and testing our model, being careful to avoid overfitting to our datasets, we could make very precise foreshadowings regarding defense companies and contractors stock markets, for example, when we tried to predict the stock market of Lockheed Martin at some point of the testing phase, it would often fail to give a reasonable answer due to tensions in the middle east, a.k.a Israel-Hamas war.

To assess the performance of our neural network model in forecasting market movements, we use the following evaluation metrics:

- Accuracy: In the context of stock price direction prediction, accuracy refers to the percentage of correctly predicted price movements (up or down) compared to the total predictions made.
- Precision: Assesses the model's accuracy in predicting positive labels by calculating the ratio of true positives to the sum of true and false positives. In financial trading, a high precision means that when the model predicts an increase in stock price, it is likely correct, minimizing erroneous trades.
- Recall (Sensitivity): Measures the model's ability to identify all relevant instances correctly. In financial terms, recall ensures that the model captures as many actual price movements as possible. High recall in predicting stock price movements means the model successfully identifies most of the true upward or downward movements, so as not to pass on profitable trades.

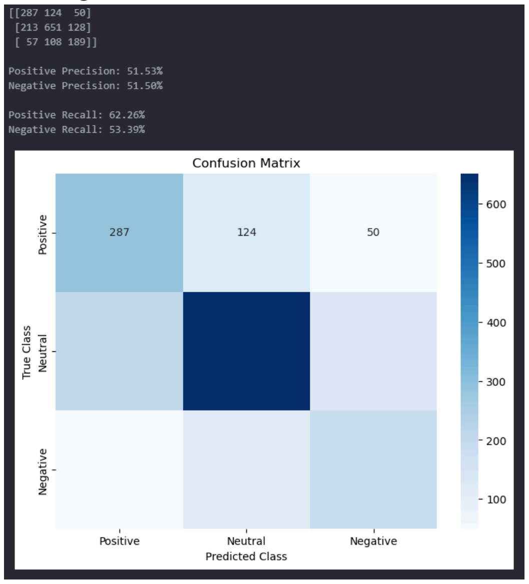
The following shows the training and validation loss and categorical accuracy of the final model as it trains, which demonstrates a partial convergence in the loss, and although it may show that the model began to overfit, this was mitigated using keras's early stopping callback which was able to save the best performing model weights and load those back once it as sure that the model was failing to improve the validation loss meaningfully.





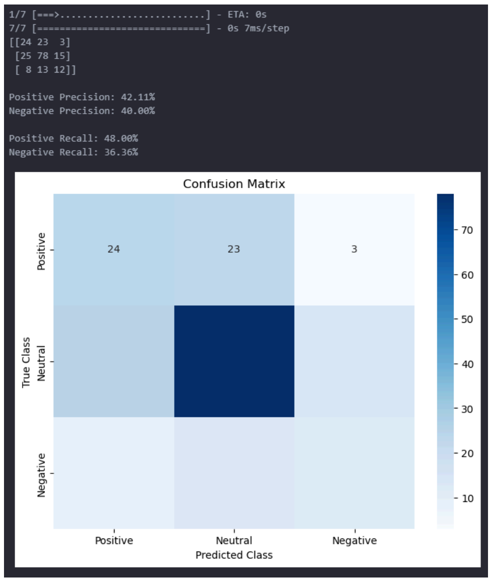
The following is the confusion matrix of the final model for both the training and validation datasets.

Training Dataset:



```
[[287 124  50]
 [213 651 128]
 [ 57 108 189]]

Positive Precision: 51.53%
Negative Precision: 51.50%

Positive Recall: 62.26%
Negative Recall: 53.39%
```

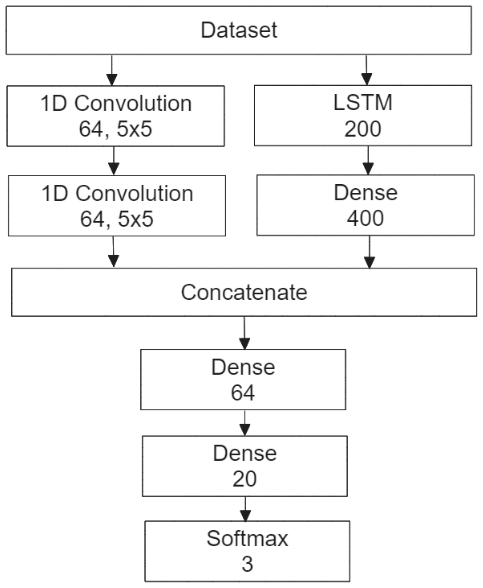
Validation Dataset:

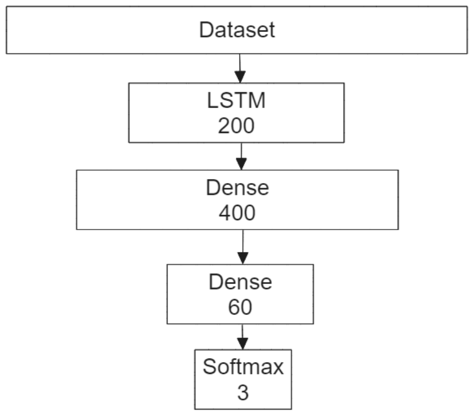


```
1/7 [===>..........................] - ETA: 0s
7/7 [==============================] - 0s 7ms/step
[[24 23  3]
 [25 78 15]
 [ 8 13 12]]

Positive Precision: 42.11%
Negative Precision: 40.00%

Positive Recall: 48.00%
Negative Recall: 36.36%
```

Note that we didn’t think it as important to evaluate the model using F1 Score or precision and recall on all the cases because we deemed neutral cases as generally unimportant.

Models Not Used





VI. CONCLUSION

In conclusion, this project has explored the application of neural network architectures, specifically combining Feed Forward Neural Networks with LSTM layers, to forecast short-term market movements within the technology sector of the American financial markets. Our approach aimed to leverage the unique dynamics of this sector, characterized by rapid technological advancements and significant data availability, to develop a predictive model that improves upon traditional financial forecasting methods.

Throughout our research, we utilized a comprehensive dataset from sources such as Yahoo Finance and Alpha Vantage. We applied sophisticated neural network techniques and rigorous data preprocessing to attempt to identify relevant patterns that could predict market movements. The integration of LSTM with FFNN layers was intended to harness both temporal and non-temporal data effectively, potentially offering a balanced approach to capture the complex dynamics of the technology sector.

However, our results have shown that while the model demonstrates promise in certain aspects, it also faces significant challenges and limitations. One of the primary issues encountered was the model's sensitivity to the volatility and unpredictability inherent in the technology sector, which at times led to inconsistencies in predictive accuracy. Additionally, the need for large and diverse datasets to train the model effectively highlighted the challenges in modeling financial markets where external factors can significantly influence outcomes.

Our findings suggest cautious optimism about the potential of integrating LSTM and FFNN architectures for financial forecasting. While the model has demonstrated some capability to predict market movements, the results also underscore the necessity for further research and refinement. Future work should focus on expanding the dataset, experimenting with different neural network configurations, and incorporating more granular data related to specific market drivers.

It is also imperative to continuously validate and adjust the model against real-world financial outcomes to ensure its relevance and accuracy over time. This iterative process is crucial in a field as dynamic and complex as financial forecasting.

This project contributes to the ongoing dialogue in the field of financial analytics by highlighting both the potential and the challenges of applying advanced neural network technologies to market prediction tasks. It underscores the importance of a cautious and iterative approach to developing and deploying predictive models in the financial sector. Through this research, we hope to encourage further exploration and critical evaluation of machine learning techniques in economic forecasting.

References

1. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=bcbb8ca9d6a6ce6017710ebf6143da76b6ed
2. https://dl.acm.org/doi/pdf/10.1145/3483596
https://www.researchgate.net/profile/Steven-Walczak-2/publication/2398947_An_Empirical_Analysis_of_Data_Requirements_for_Financial_Forecasting_with_Neural_Networks/links/569d3c2608ae16fdf0796592/An-Empirical-Analysis-of-Data-Requirements-for-Financial-Forecasting-with-Neural-Networks.pdf?origin=journalDetail&_tp=eyJwYWdlIjoiam91cm5hbERldGFpbCJ9
3. https://www.researchgate.net/profile/Uzochukwu-Onwuachu/publication/295261436_A_Neural_Network_Approach_to_Financial_Forecasting/links/620c7a90634ff774f4d1562c/A-Neural-Network-Approach-to-Financial-Forecasting.pdf
4. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=10d1f52113df1d52926f7271e2c35076e571bd08
5. https://dl.acm.org/doi/pdf/10.1145/3483596
6. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=1cdbed6fba31b7298b7fff96f1c028a5c6c15e8a
7. https://eds.p.ebscohost.com/eds/pdfviewer/pdfviewer?vid=1&sid=e637027a-f8b1-410c-a73a-6d5500c52577%40redis
8. https://work.caltech.edu/paper/Abu-Mostafa1996forcast.pdf
9. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=e7ce06dc8611f415dc2a609784d2bc579441b34d
10. https://www.cse.unr.edu/~harryt/CS773C/Project/895-1697-1-PB.pdf
11. https://link.springer.com/article/10.1007/s41060-021-00279-9#Sec1
12. Kumbure, M., Lohrmann, C., Luukka, P., & Porras, J. (2022). Machine learning techniques and data for stock market forecasting: A literature review. Retrieved from https://www.sciencedirect.com/science/article/pii/S0957417422001452
13. Jiang, W. (2021). Retrieved from https://www.sciencedirect.com/science/article/abs/pii/S0957417421009441
14. Fischer, T., & Krauss, C. (2017). Retrieved from https://www.sciencedirect.com/science/article/abs/pii/S0377221717310652
15. Mustafa, Z., Yaseen, A., Naeem, H., & Aftab, M. (2019). Stock market prediction with the integration of daily news sentiments. Retrieved from https://ieec.neduet.edu.pk/2019/Papers_IEEC_2019/IEEC_2019_37.pdf