Project Short Paper

Emma Choueiry (choueir1), Josh Keeney (keeneyjo), Logan Langmeyer (langme10), Ricky Li (liricky1), Skanda Vijaykumar (vijayku2)

Abstract

This paper outlines a machine learning project aimed at developing a predictive trading strategy for the stock market. Utilizing historical stock data, our model will predict optimal moments to buy, hold, or sell stocks. We will explore various machine learning algorithms to achieve precise predictions, enhancing trading strategies and financial outcomes.

1 Introduction

The stock market is constantly changing, and nearly impossible to predict if you don't know what you're doing. Investors and traders alike strive to maximize profits and make well-informed decisions based on experience, but the market can sometimes be unpredictable. This fluctuation provides an interesting challenge when trying to find the best time to change up your stock portfolio. Machine learning techniques, however, have demonstrated promising capabilities in recognizing patterns within large datasets and making data-driven predictions.

This project will focus on finding the best moments to buy, sell, or hold certain stocks. We will investigate the performance of various ML models, including deep learning architectures, in forecasting stock prices.

2 Dataset(s)

Stock prices were obtained using finance from Yahoo Finance, which provides historical market data for various publicly traded companies. The dataset consists of historical stock market data for 24 stocks, capturing daily trading activity from 2020 to 2024. Each record represents a trading day for a particular stock.

Open: The price of the stock at the beginning of the trading session (numerical).

Close: The price at which the stock closed for the day (numerical).

High: The highest price reached during the trading day (numerical).

Low: The lowest price observed (numerical).

Volume: The number of shares traded during the trading day (numerical).

Date: Trading day (timestamp).

3 Model(s)

3.1 Logistic Regression:

Classification: Predict whether a stock should be bought, held, or sold.

Will output probabilities for each class (buy, hold, sell), and the action with the highest probability will be chosen as the decision.

3.2 Neural Network Model:

Extract relevant features and identifies important trends/correlations in stock behavior.

Make prediction for trading of stock.

Tune hyperparameters to reduce loss and improve performance.

3.3 Evaluation:

Precision: measures how many of the predicted buys/ holds/ sells were correct.

Recall: measures how much of the buys/ holds/sells/ were correctly predicted.

Accuracy: overall percentage of correct predictions for all three classes.

4 Related Works

Extensive research has been conducted in the realm of stock market predictions, utilizing a wide range of datasets and methodologies. A significant portion of the existing literature focuses on understanding market volatility and identifying the most profitable stocks for investment. These studies have primarily aimed at forecasting price movements and identifying high-return opportunities based on historical data trends.

Unlike these prior efforts, our project adopts a distinctive approach by developing a machine learning model tailored to support individual trading strategies for any chosen stock. Instead of merely predicting stock prices or identifying profitable stocks, our model will provide actionable recommendations on whether to buy, sell, or hold a stock based on real-time data analysis and historical trends. This approach ensures that our system is versatile and directly applicable to traders' specific needs, offering a strategic tool that enhances decision-making across diverse market conditions and stock portfolios.

5 Researcher Bios

Logan Langmeyer - langme10 - Computer Science - Senior

Emma Choueiry - choueir1- Computational Data Science- Junior

Skanda Vijaykumar - vijayku2 - Computer Science - Junior

Josh Keeney - keeneyjo - Data Science - Senior

Ricky Li - liricky1 - Data Science - Senior

6 Work Delegation

Cleaning and Preprocessing: Logan, Emma

Baseline model (Logistic Regression): Logan, Skanda

Neural Network Model: All

Hyperparameter Tuning/Model Optimization: Josh Keeney, Ricky Li

Model evaluation and visualizations: Emma Choueiry, Josh Keeney

We have divided the tasks among our team members based on individual strengths and expertise. Logan and Emma will focus on the cleaning and preprocessing of the data, ensuring that it is properly prepared for modeling. Logan and Skanda will then develop the baseline model using Logistic Regression, establishing a foundation for comparison with more advanced models. All team members will contribute to the implementation of a Neural Network model, collaborating on its design and execution. Josh Keeney and Ricky Li will handle the hyperparameter tuning and model optimization, refining the models for better performance. Lastly, Emma Choueiry and Josh Keeney will take responsibility for model evaluation and visualizations, presenting the results in an insightful and comprehensible way. This division of labor leverages the unique skills of each team member to efficiently complete the project.

7 Data Overview

7.1 Data Cleaning and Preprocessing:

There are a significant number of missing values in the dataset due to:

- Stock market holidays and weekends: The market is closed on non-trading days.
- Suspended trading days: Certain stocks might not have been traded on specific days.
- Not traded until a specific date: Certain stocks might not have opened until a later date

To maintain consistency in the dataset and avoid data loss, missing values were treated as follows:

- Non-trading days (weekends) were dropped, as no market activity occurs on these days.
- Missing Open, Close, High, and Low values were imputed using the average of the previous and next trading day.
- Missing Volume values were filled using the median, as trading volumes can have a high variance, and using a mean could skew results.

No duplicate rows were found in the dataset.

7.2 Data Splitting:

The dataset was split in the following way to ensure a robust model:

- 70% Training Set: Used for model training and learning patterns from historical data.
- 15% Testing Set: Used to assess model performance on unseen data.
- 15% Validation Set: Used to fine-tune hyperparameters and prevent overfitting.

7.3 Feature Scaling:

Stock prices and trading volumes vary significantly in magnitude. To prevent models from being biased toward larger numerical values, feature scaling was applied. The StandardScale transformation was used to normalize the numerical features to have a mean of zero and a standard deviation of one. This ensures that the logistic regression model performs efficiently.

8 Models

In this project, we will develop and implement a logistic regression model and a neural network model to predict the optimal trading actions. The logistic regression model will serve as the simple baseline model and the neural network model will be used as an advanced model. The logistic regression model was chosen as the baseline due to the model's interpretability for binary classification problems, which for our case is yes/no for each action. The neural network is used as the complex model as the deep learning architecture is able to capture the complex relations and patterns in financial data, allowing the model to pick up on intricate market trends that simpler models might overlook. By comparing the performance of these two models, we aim to assess the benefits of deep learning over traditional statistical methods in financial forecasting.

8.1 Logistic Regression Model:

We started the project by implementing a logistic regression model to try to predict if the Apple stock would increase or decrease for the next day. We wanted the model to try to predict the probability of an increase in value, and use a softmax function to give us the output with a 1 signifying an increase in value, and a 0 being a decrease.

After running the model, we achieved the following results:

Validation Accuracy: 96.93%
Validation Recall for Decrease: 98%
Validation Recall for Increase: 51%
Validation Precision for Decrease: 98%
Validation Precision for Increase: 52%

To evaluate the performance of our trained FNN model, we conduct testing on the test set, achieving the following results:

Test Accuracy: 96.91%
Test Recall for Decrease: 98%
Test Recall for Increase: 49%
Test Precision for Decrease: 98%
Test Precision for Increase: 52%

For a more detailed view on the current Logistic Regression model's validation/testing results, refer to the "Logistic Regression" section in the jupyter notebook located in the team's Github repository.

A problem we found when evaluating the confusion matrix for this model is that our data is very skewed to where we have a lot more days where the stock decreases versus when it increases, so it does very well predicting a decrease, but not as well for increases.

8.2 Neural Network Model:

We have currently implemented a simple feedforward neural network (FNN) using tensorflow's keras library to predict Apple's stock price increasing or decreasing for the next day. This neural network architecture consists of three dense layers: Input Layer: Processes and extracts high-dimensional representations from the input stock data. This layer consists of 64 neurons and utilizes the ReLU activation function. Hidden Layer: Compresses and refines the features and patterns extracted from the input layer. This layer consists of 32 neurons and utilizes the ReLU activation function. Output Layer: Produces a final probability between 0 and 1. Utilizes the sigmoid activation function.

For the neural network compilation, we used Adam for the optimizer, binary cross-entropy for the loss function, and accuracy as the model evaluation metric.

After training the model for 20 epochs, we achieved the following results:

- Training Accuracy: 96.87%
- Training Loss: 0.1361
- Validation Accuracy: 96.81%
- Validation Loss: 0.1377

To evaluate the performance of our trained FNN model, we conduct testing on the test set, achieving the following results:

- Testing Accuracy: 96.77%
- Testing Loss: 0.1393

For a more detailed view on the current FNN model's architecture and training/testing results, refer to the "Neural Network Model" section in the jupyter notebook located in the team's Github repository.

Moving forward, our group will need to implement a more complex neural network for predicting optimal trading action as the feed-forward neural network we currently have is too simple. Possible options for other architectures include RNNs, LSTMs, or even CNNs.

9 Overall Initial Experimental Results

When implementing our models, we found that we could accurately predict whether the stock for one company will increase or decrease for the next day with almost a 97% accuracy for both of our models. Moving forward in the project, we will be able to further implement this information to have the model output what should be done with a specific stock for the maximum profit.
Having this information will allow the user to look at the stock of their choice, and have the model give them a recommendation of whether they should buy, sell, or hold the stock based on the given day. The model will use past patterns of the stock to hopefully predict the best option for a specific stock. Ideally, the model will also be able to predict when the best point to buy or sell a stock is to ease the process of working with the stock market.