4/21/2025

Git hub link  
https://github.com/CharlSmuts/MLG382\_CYO\_Project.git

Machine learning Stock prediction report

Matt Takawira 600791

Charl Smuts 600270

Ruan Le Raux 600194

Vunene Khoza 600676

Table of Contents

[Introduction 2](#_Toc196338572)

[Background 2](#_Toc196338573)

[Data preparation 3](#_Toc196338574)

[Details of dataset (overview and origin of data) 3](#_Toc196338575)

[Data cleaning including handling missing values, outliers, and inconsistencies 3](#_Toc196338576)

[Pre-processing including feature scaling, encoding categorical variables, and splitting the data into training and testing sets. 3](#_Toc196338577)

[Exploratory Data Analysis 5](#_Toc196338578)

[Details of data exploration including insights, include screenshots of graphs from the code 5](#_Toc196338579)

[Feature engineering 6](#_Toc196338580)

[Model Building 7](#_Toc196338581)

[Logistic regression 7](#_Toc196338582)

[Random forest 7](#_Toc196338583)

[Evaluation of models 8](#_Toc196338584)

[Neural network 9](#_Toc196338585)

[Model deployment – 11](#_Toc196338586)

[Deployment steps 11](#_Toc196338587)

[Saving models 11](#_Toc196338588)

[Loading data 12](#_Toc196338589)

[Declare app 12](#_Toc196338590)

[Initialise HTML 13](#_Toc196338591)

[Process web app input 14](#_Toc196338592)

[Web application 16](#_Toc196338593)

[Conclusion 17](#_Toc196338594)

[Reflection 18](#_Toc196338595)

## Introduction

Predicting the stock market is a dynamic and complex process that requires examining trends, historical data, and several market indicators. Building a binary classification model to forecast whether the Samsung index will close higher or lower in the upcoming month is the goal of this project.

The dataset makes up weekly historical stock data for Samsung Electronics (005930.KS). It has several years' worth of features, including Open, High, Low, Close, and Volume. Two models for machine learning were trained: Logistic regression - A linear model for binary categorization (price up/down). Random Forest Classifier - An ensemble model that manages feature interactions and nonlinear correlations.

Users will be able to input current stock metrics and receive forecasts by integrating the final trained model into a web application. For the frontend, the application might make use of frameworks like dash.

## Background

Technical indicators, historical stock prices, and trends can all be used to forecast future moves. ML models trained on past data can be used to predict whether the price will rise or fall. Short-term movements influenced by recent closing prices and volatility, and momentum indicators can be captured using lag-based binary classification.

## Data preparation

### Details of dataset (overview and origin of data)

* Source - Local CSV file (005930.KS\_Weekly.csv, 005930.KS.csv, 005930.KS\_monthly.csv
* Features - Date, Open, High, Low, Close, Volume
* Target - Target column indicating whether the next months close is higher or not
* Size and Coverage: Multiyear weekly data

A computer code with text

AI-generated content may be incorrect.

Figure 1 read data

### Data cleaning including handling missing values, outliers, and inconsistencies

* Handling Duplicates – Removed duplicate dates
* Missing Values – Dropped rows with nulls



Figure 2 drop duplicates

### Pre-processing including feature scaling, encoding categorical variables, and splitting the data into training and testing sets.

* Feature Scaling – Applied where necessary
* Encoding – Binary target variable for direction

A screen shot of a computer code

AI-generated content may be incorrect.

Figure 3 split test and train data

A screen shot of a computer

AI-generated content may be incorrect.

Figure 4 defining model

NN Models

Binary Classification Report on Samsung Monthly Data



Figure 5 Binary Classification



## Exploratory Data Analysis

### Details of data exploration including insights, include screenshots of graphs from the code

* Trend Analysis – Weekly close prices show significant volatility and upward long-term movement.

Correlation Matrix – Strong self-correlation across price metrics (Open, High, Low, Close).

Figure 6 Correlation Matrix code

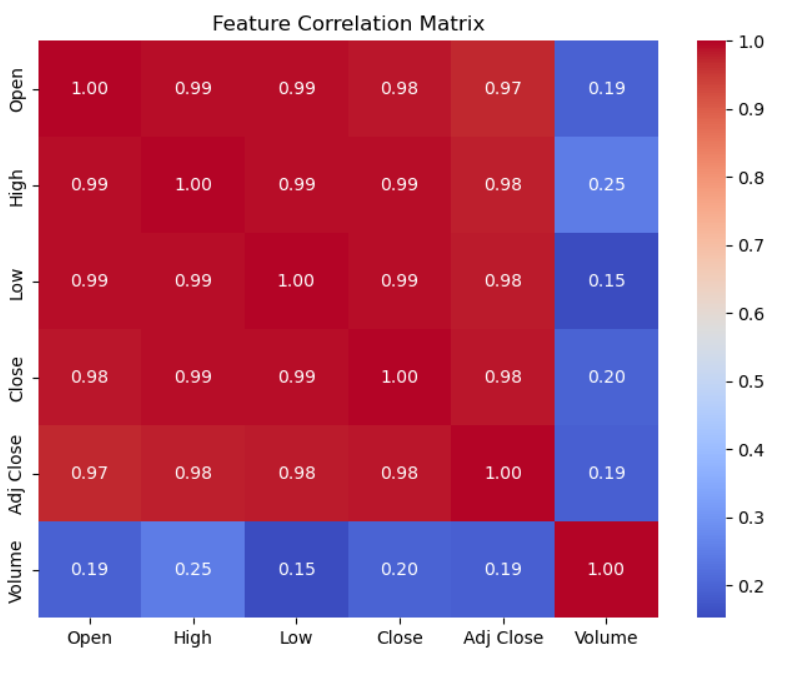


Figure 7 Correlation Matrix graph

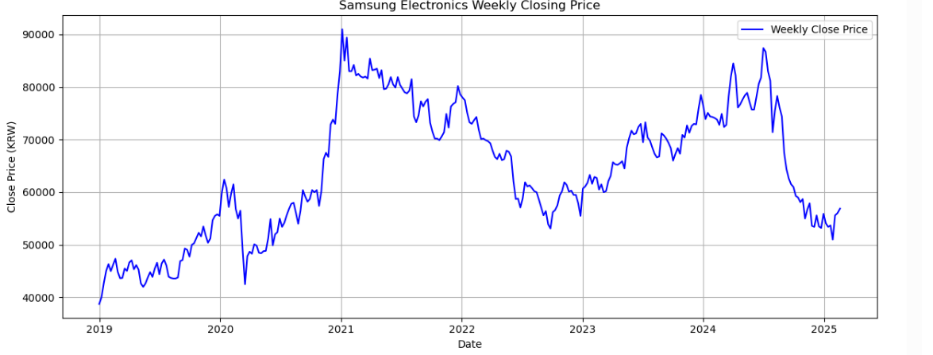


Figure 8 graph of actual closing price

## Feature engineering

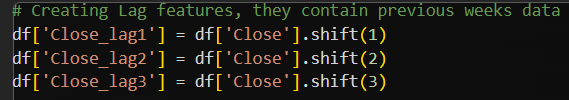
* details of features used and how they work, include screenshots from the code

Figure 9 lag feature

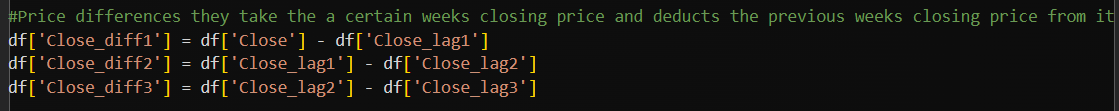


Figure 10 price difference



Figure 11 volume lag

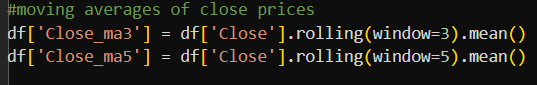


Figure 12 moving average



Figure 13 closing price

A black screen with white text

AI-generated content may be incorrect.

Figure 14 drop NaN value

## Model Building

* details of ml models used and how they work, include screenshots from the code showing the model and details on how the model works and performed

### Logistic regression

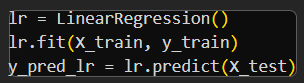


Figure 15 Logistic Regression fitting

### Random forest

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 16 Random Forest fitting

### Evaluation of models

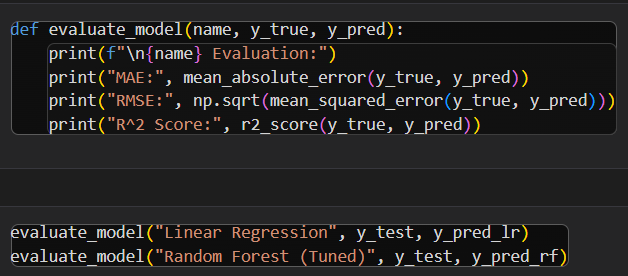


Figure 17 model evaluation

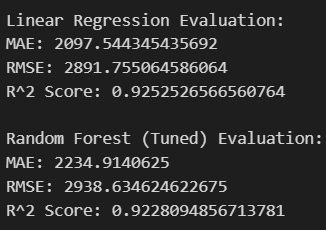


Figure 18 evaluation results

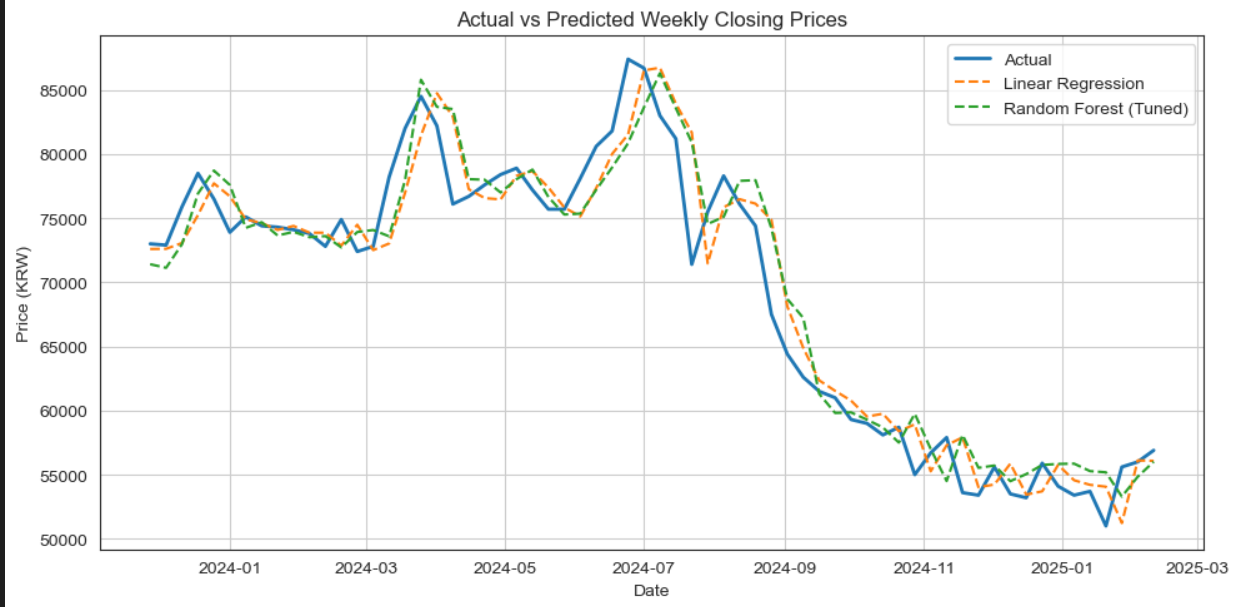


Figure 19 graph of prediction vs actual results

### Neural network

A screenshot of a computer code

AI-generated content may be incorrect.

Figure 20 building neural network

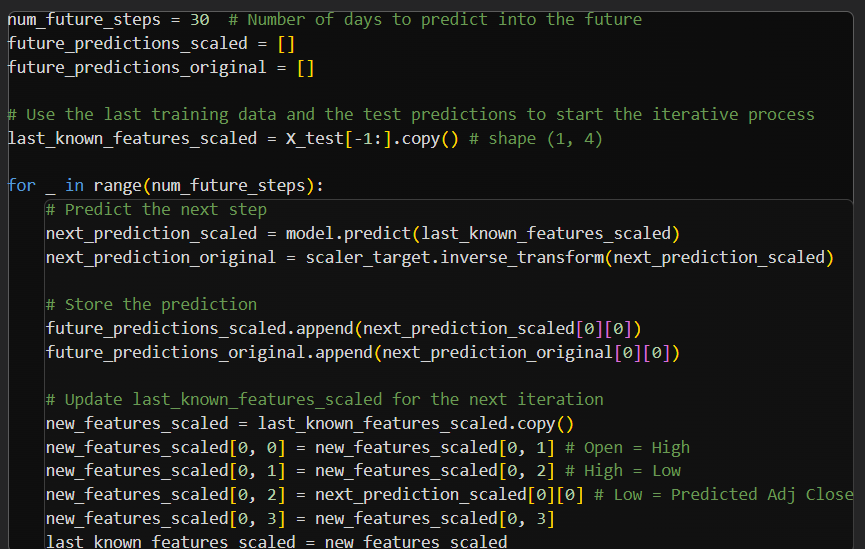


Figure 21 logic of neural network

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 22 Neural network evaluation

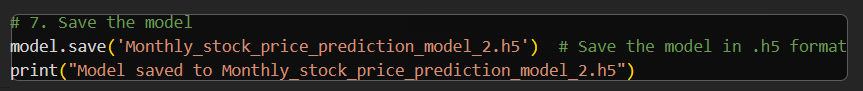


Figure 23 saving neural network model

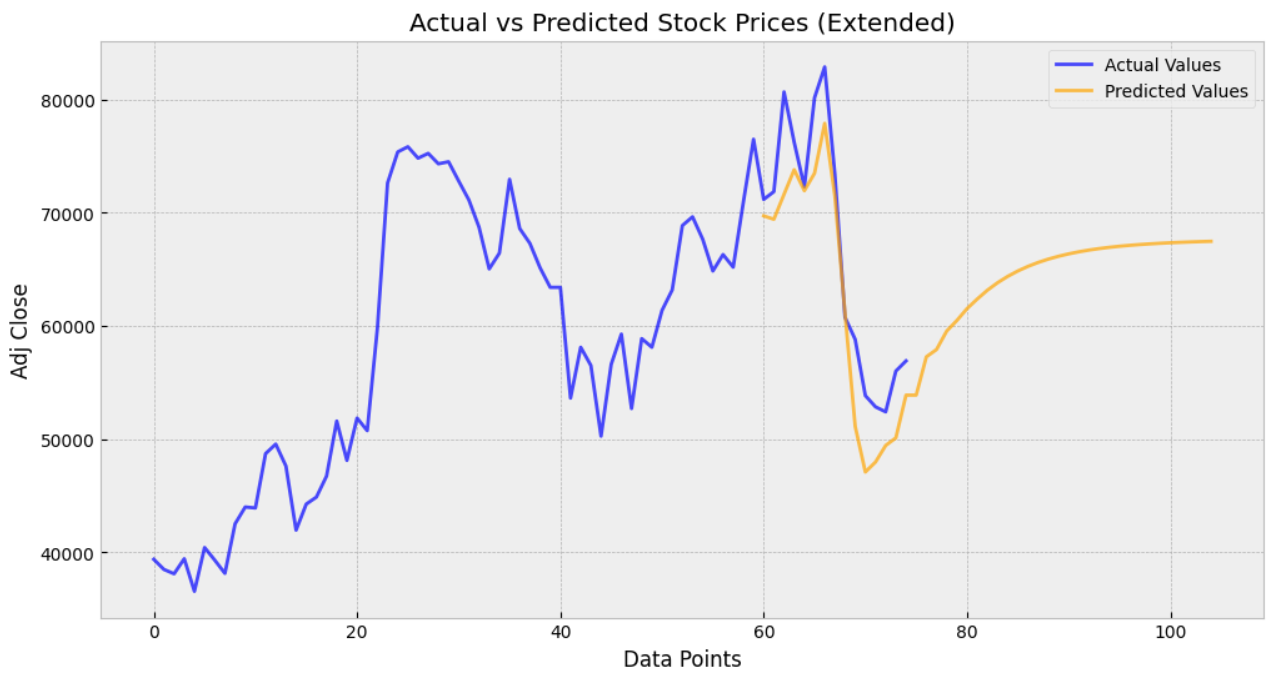


Figure 24 Neural network prediction graph

## Model deployment –

### Deployment steps

1. Save trained models from notebook
2. Load Datasets to be tested in python web application
3. Preprocess loaded data
4. Declare the web application and initialise the html layout of the web app
5. Write code to receive web app input and run it through the model
6. Write code display the output on the web app
7. Include code to run the app server at the end
8. Upload the files to Git with a requirements text file
9. Upload the Git repository to Render
10. Setup the Build
11. Deploy the build
12. The application is ready to use

### Saving models

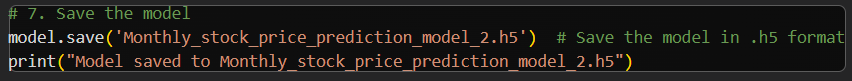


Figure 25 saving monthly model

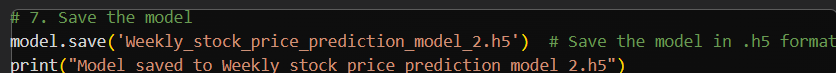


Figure 26 saving weekly

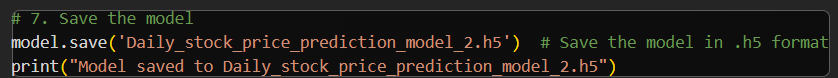


Figure 27 saving daily model

### Loading data

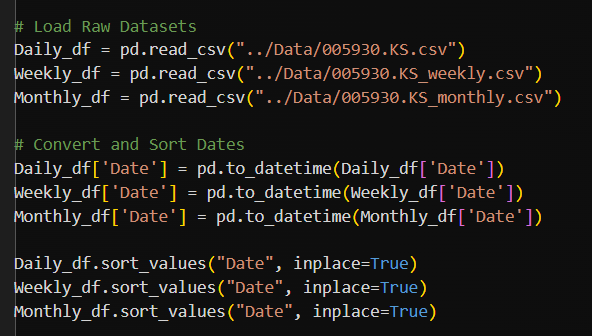


Figure 28 load data

### Declare app

A black background with white text

AI-generated content may be incorrect.

Figure 29 declaring app

### Initialise HTML



Figure 30 HTML code

A computer screen with text and images

AI-generated content may be incorrect.

Figure 31 HTML continued

### Process web app input

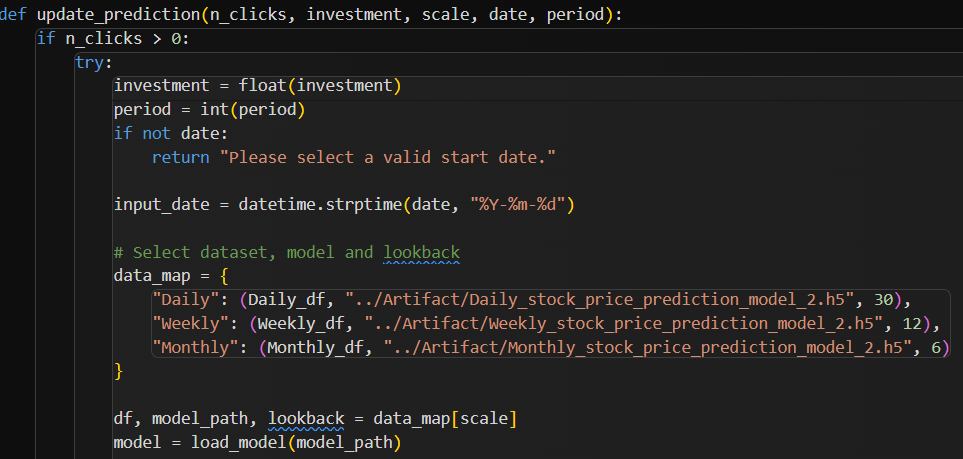


Figure 32 receiving input

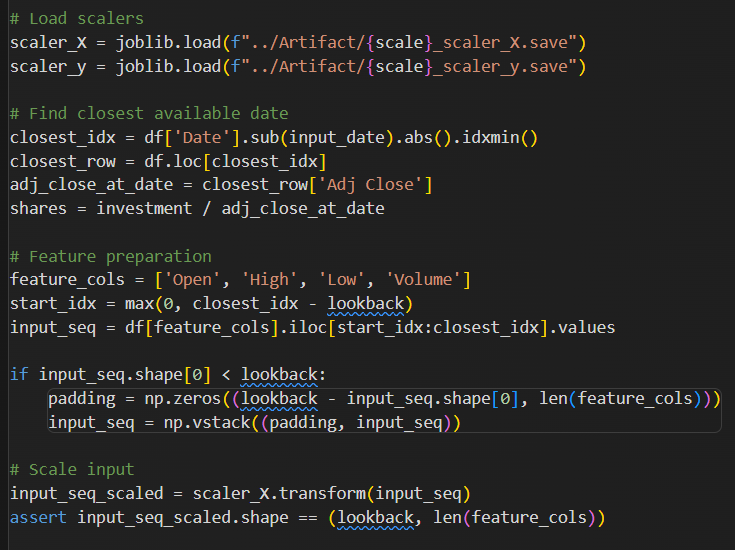


Figure 33 pre-processing input

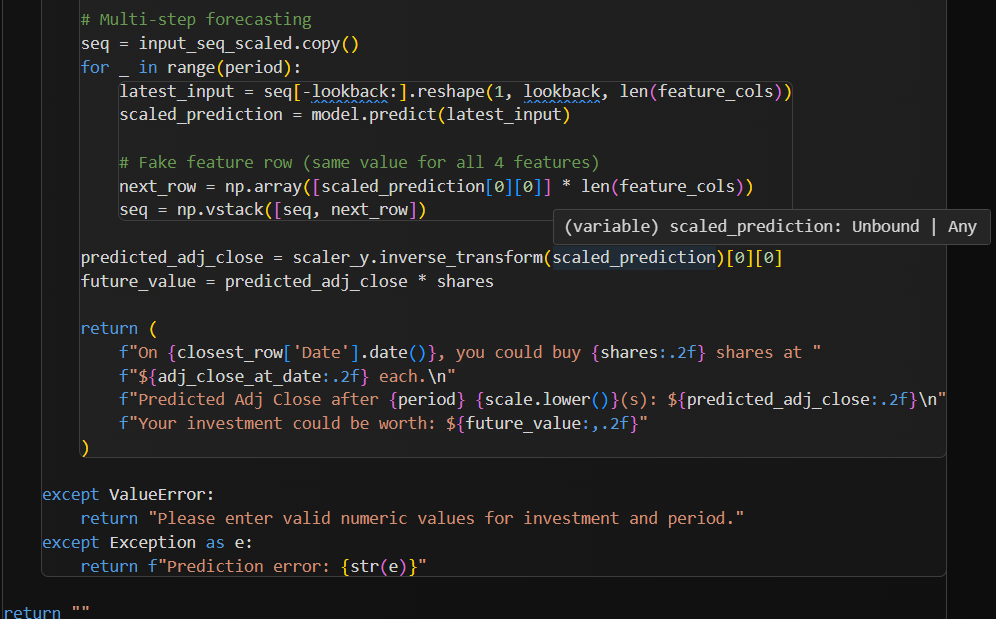


Figure 34 making prediction and return result

### Web application



Figure 35 Web app monthly prediction

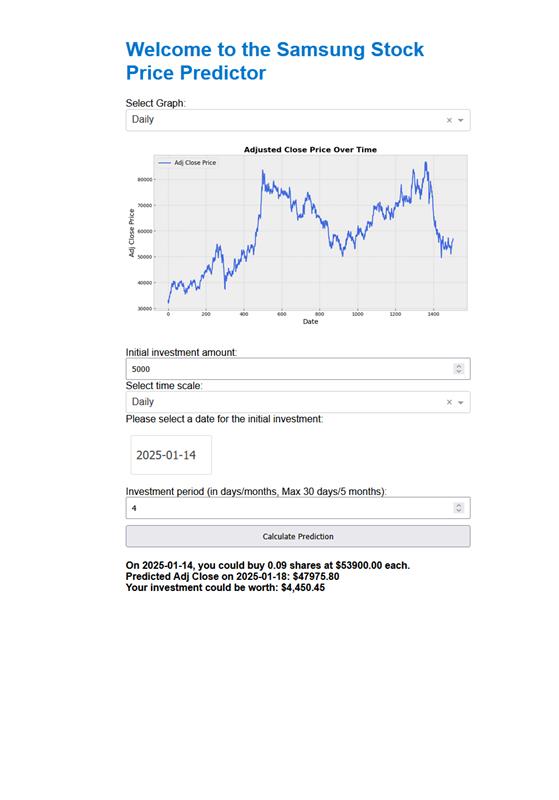


Figure 36 Web app daily prediction

## Conclusion

This project successfully demonstrated the application of machine learning for stock market prediction, focusing on forecasting the Samsung index's monthly direction. Two simple classification models — Logistic Regression and Random Forest — were trained using engineered features derived from historical stock data for weekly data. Among them, the Random Forest model showed better performance due to its ability to handle nonlinear relationships and feature interactions. Neural networks were also used as a more complex model for the daily and monthly data.

The final model was integrated into a web application, providing a user-friendly interface for making predictions based on real-time input. This demonstrates the project’s practical viability and potential for real-world use.

## Reflection

Key challenges faced during the project included issues with model integration, web application deployment, and data mismatches that occasionally impacted model training and prediction accuracy. Through iteration and testing, these issues were gradually resolved, leading to a more stable and consistent application.

Future improvements may involve adding more technical indicators, utilizing advanced models like LSTMs for time series forecasting, enhancing the front-end experience, and incorporating real-time data feeds. The foundation laid by this project offers ample opportunity for continued development and research.