My research question is "Can machine learning models accurately predict stock price movements based on historical market data?” . This is a significant question that can have many positive repercussions as much of the world’s actions influences what happens in the stock market and vice versa. There are a multitude of reasons as to why anyone should care about my question. One reason includes more informed investment decision making. If machine learning models can accurately predict the movement of stock prices, then these predictions would allow investors to make more informed financial decisions regarding the stock market. The use of machine learning models can help with identifying more volatile periods in the market, thus aiding investors in improved risk management. Certain investors, more specifically hedge funds, utilise algorithmic strategies. The predictions of the machine learning models, provided the predictions are accurate, could enhance the performance of these algorithms. Market analysis can be broadened using machine learning models’ predictions as the analysis of historical market data can identify relevant patterns or trends.

My dataset consists of historical stock market data from 10 companies. I found this dataset on Kaggle.com, and it consists of a CSV file containing 25,161 rows, where each row represents the stock market data for a given company on a given date. The information was obtained through web scraping the official NASDAQ website. The dataset contains seven columns, and these columns are labelled as “Company”, “Date”, “Close/Last”, “Volume”, “Open”, “High” and “Low”. The dataset contains ten companies, which include Apple, Starbucks, Microsoft, Cisco Systems, Qualcomm, Meta, Amazon, Tesla, Advanced Micro Devices (AMD), and Netflix, and their stock market data from 2013 to 2023.

As shown above, preprocessing was undertaken to ensure that the data was ready to be used by the models. The steps I took to preprocess the data include converting the dates to datetime format, formatting the ‘Close/Last’, ‘Open’, ‘High’ and ‘Low’ columns to remove the dollar sign within them as well as dropping any potential NA values and dropping the "Company" column to prevent the models from overfitting with any companies.

The machine learning technique I used is regression. I chose regression as my machine learning technique as it makes most sense for the type of data I am working with, and regression would produce the most usable predictions as opposed to classification and clustering, which cannot predict continuous values.

I have decided to use three machine learning models for my data analysis. One model is a linear regression model, one is a neural network (multi-layer perceptron) regressor model and the other is a random forest regressor model. A linear regression model is a type of supervised learning algorithm that is used for regression problems. It typically works when there is a linear relationship between the input features and the target feature. A multi-layer perceptron (MLP) regressor model is a type of neural network that is used for regression tasks. Unlike a linear regression model, it is more suited to tasks where there is a non-linear relationship between the input features and the target feature. Efficient hyperparameter tuning ensures more optimised performance. A random forest regressor model is a type of ensemble machine learning, which means it utilises multiple algorithms and combines them to obtain more accurate predictions, that is used for regression tasks. Just like the MLP regressor, efficient hyperparameter tuning will ensure a more optimised performance.

The image on the left shows how I trained the models. I used the sklearn library, which provides me with the various models that I can use. In the image shown, I use a lambda function to apply the timestamp method to every value in the ‘Date’ column. Without this line of code, I was running into errors when running the code when trying to train the models. The neural network model is the only model where I adjusted the hyperparameters. Without the adjustment of the hyperparameters, the performance of the neural network model was very poor. I then split the data into input and output sets by using the iloc function from pandas. The train\_test\_split function from sklearn is then used to split the input and output sets into training and testing sets. I set the test\_size as 0.3, which means the training/testing split is 70/30. Random\_state is set to 1 to ensure that the same sets of data will be produced with each run, ensuring result reproducibility. I used the StandardScaler from sklearn’s preprocessing library, where I normalised the input training and testing data for the neural network model. The results produced from the model were very poor, and the normalisation of the data was a key step in solving this issue. The models are then trained using the fit() functions, passing the training data as parameters.

The image on the left displays the R-squared (R2) scores of the three models. The R2 score is calculated by subtracting the sum squared regression divided by the total sum of squares from 1. The score ranges from values 0 to 1, where 0 signifies that the model does not explain any variability in the target variable, whereas 1 signifies that the model explains all the variability in the target variable. As shown, all of the values are very close together. They are all above 0.999, with the LR model having the highest accuracy at 0.99981 (to 5 decimal places), then the NN model (MLP regressor) having an R2 score of 0.99979, and the RF regressor model having an accuracy of 0.99975. The values being so close to 1 suggests that all of these models are performing extremely well on the dataset.

As shown through the graphs, all of these models produced results with a strong positive correlation. The graphs help visualise how accurate the models are. The strong positive correlation shows that the models are making very accurate predictions as most of the points are mapping the predicted values to actual values.

Throughout the course of this project, I have learnt how to find datasets, preprocess data, train and test machine learning models and allowed myself to learn more about machine learning techniques, more specifically regression. The key takeaways are that machine learning models that utilise regression can effectively predict stock price movements based on historical data. In the real world, the values may be influenced by many other factors, from financial factors to global events.

There are a few limitations on my data and the methods I used to obtain my results. Firstly, the dataset only contained data from ten companies. As the companies are mostly tech or social media related companies (with the exception of Starbucks), it could cause the model to have industry bias. Due to the dataset only being from ten companies, it may not be an accurate representation of the whole market. Another limitation is the fact that the dataset only contains stock prices over the past ten years. Limited historical representation, including events such as the 2008 financial crisis, failure to represent stock market before the impact of technological advancements and climate & environmental considerations to name a few of the consequences. These can all be prevented by training the model on a larger, more diverse dataset, containing more companies from multiple sectors and from a wider time range.

Limitations on my methods include only using one method of testing the data (only checking R2 scores), only using three models, not adjusting hyperparameters for random forests model and not using a different training/testing split to compare results.