CMT307: Coursework 2 - Group Project

Group Number	**8**
Project Title	**Stock Price Prediction**
Supervisor	**Dev Kant**
Word Count	**4458**
Student IDs	22085988 23122046 23077348 23126214 23080077 23025443 23092051 23022937

Introduction

This project will firstly aim to assess the literature of share price prediction, reviewing papers which analyse and review various forecasting techniques and their respective performance. Here we will learn the techniques that have been employed and which have been effective in predicting share prices. We will then go on to build these models which are discussed to assess their performance, where, in some cases, we will employ feature engineering to improve predictive capacity and alter various hyper-parameters to tinker with the architecture of the models to improve performance. We will then look to finally visualise the performance of these models using appropriate error terms to demonstrate their predictive capabilities and then comment on these results.

The insights of this project demonstrate that modern ML methods outperform the traditional forecasting techniques significantly, Naïve and ARIMA modelling are massively outperformed by the other ML models in our analysis. This follows the expectation from the literature, where more traditional techniques have been outperformed by ML techniques when it comes to share price predictions. Machine Learning techniques have shown that SVR, Random Forest, Linear Regression perform very well over short prediction periods whilst LSTM and Stateful LSTM perform better over longer periods but are still outperformed by these other techniques. We found that Linear Regression performed the best across these models and this was likely due to the strong linearity found in the relationship between the predictor variables and the target variable in our analysis.

Literature Review

The methods by which to forecast share prices have changed with the emergence of Machine Learning techniques. These can be categorised into two main approaches. The first is regression based approaches such as Support Vector Regression (SVR)and Random Forest Regression, and the second approach is the Neural Network approach which include Long Short-Term Memory Networks (LSTM) and Recurrent Neural Networks (RNN). These new approaches have challenged more traditional Time-Series forecasting methods (ARIMA, moving averages, exponential smoothing and Holt-Winters) which have historically been used for stock price prediction. The prospect of Machine Learning improving share prices predictions has wide-ranging implications, it promises a future in which firms and individuals can make more informed investment decisions. In which, profits are maximised, and losses minimised. On a larger scale, it helps to stabilise markets by steering investors away from riskier assets through accurate risk management, benefitting not only investors but wider society by limiting risk to the greater economy. This literature review seeks to understand the effective methods which lead to accurate share price prediction, it will compare multiple sources to provide a thorough overview of the various techniques that have been used for this task and their respective efficacies. It will evaluate literature which compare both Time Series and Machine Learning techniques and, it will also compare the efficacy between various Machine Learning models and their respective accuracies.

When comparing the efficiency of ARIMA and traditional statistical techniques against ML models, several studies have found traditional time-series forecasting methods to perform comparatively well against other machine learning techniques. (Ayodele A. Adebiyi, 2015) found that ARIMA models were particularly useful for short-term forecasting of share prices. Similarly The study by (Islam & Nguyen 2020) investigated the performances of popular models used for the stock price prediction and compared them to determine which model was the most effective out of ARIMA, Brownian and ANN models. They found that ARIMA and Brownian models outperformed ANN in short term next day predictions with ARIMA and Brownian models producing almost identical results with any difference being insignificant. The study followed up on previous research comparing the effectiveness of ARIMA, ANN and Brownian models for predicting the price of a stock. Across the studies (Khashei & Bijari 2010; Merh et al 2010; Lee et al 2007) ARIMA models outperformed ANN models in all cases with the exception of hybrid ANN models. Whilst (Rathnayaka et al, 2014) found that stochastic model prediction is more significant than the traditional ARIMA model, however Islam & Nguyen study found there was no significant difference between the two.

Other studies have demonstrated that regression and neural network ML methods have produced more accurate outputs over longer periods of time than traditional time-series forecasting methods. (Sima Siami-Namini, 2018) Found that LSTM models on average were 85% more effective at predicting share prices across a study of 6 shares and 6 funds than ARIMA across years of predictions. (Indronil Bhattacharjee, 2019) conducted an analysis which investigated traditional statistical approaches such as Simple Moving Average (SMA), Weighted Moving Average (WMA), exponential smoothing and the naïve approach against machine learning algorithms. (SLP, MLP and LSTM) The analysis was undertaken using Apple and Tesla previous close prices and found dramatic differences between these statistical techniques and ML ones. The MSE for statistical techniques ranged from 322.4 to 11.2 across both datasets whilst the MSE for ML methods ranged from 3.5 to 10.3. This demonstrates a huge difference in predictive accuracy between ML algorithms and traditional statistical methods.

Furthermore, when LSTM was compared to other ML models it performed better. (Pengfei Yu, 2020) conducted a study comparing ARIMA with multiple ML methods and found that LSTM outperformed all the other methods. (Mahla Nikou, 2019) evaluated the use of deep learning algorithms in ANN and LSTM and compared their performance against SVM and RF. The study concluded that deeper neural networks, specifically LSTMs, were more powerful than other machine learning methods and suggested for further research that different types of LSTM be trialled and tested. (Mehar Vijh, 2020) study focused its attention on ML techniques as these techniques have improved share price prediction efficiencies by 60-86%. During this study he implemented feature engineering to generate a variety of inputs including range (high price-low price) and an array of moving averages (7-day, 14-day and 21-day moving averages) to input to the ANN. The study found that ANN and Random Forest were very successful in predicting stock prices with both demonstrating very low RMSE and MAPE values, however, ANN performed better. Despite numerous papers finding LSTM performed the best in share price prediction, (Sreelekshmy Selvin, 2017) and (M et al. 2018) undertook separate studies looking into the performance of CNN, RNN and LSTM and both concluded that CNN performed the best. Each of their studies highlighted that this is likely due to its ability to capture sudden alterations that occur in the share price increasing its capacity to capture dynamic changes.

To conclude the findings across the literature, whilst research has shown a trend in neural network models such as LSTM and CNN outperforming other models, different studies have produced results on the optimal model for stock price prediction that have been inconsistent or conflicted with the findings of other studies suggesting that there is still no consensus on the optimal model for stock price prediction. This further highlights the importance of our research in comparing the performances of a large selection of time-series, regression and neural network models in predicting stock prices across different time periods in order to provide definitive insights on their comparative performances which will support further research.

Description of the Task / Dataset

The aim of the research is to expand upon existing literature to test, compare and contract different machine learning models' efficacy. The project involves 3 datasets for each of the stocks used in the projects, the idea behind this was to gather data from three different types of stocks with all different structures. Apple shows a rapid increase in the share price, Johnson and Johnson has a relatively stable share price and Tesla has seen a rapid decline in the share price. When comparing these different shares we hope to capture a more accurate representation of the models accuracy across data structures when it comes to share prices. Furthermore, we used short (7-days), medium (30-days) and long term prediction horizons (90 days) to assess whether certain models performed better over shorter horizons or increasingly longer ones. Both of these variations in our analysis were used to provide a comprehensive assessment of the best methods to employ when predicting share prices.

In order to predict the close price we shifted the predictor variables to t-1 and used this information to predict the share price at time t. This is to provide a realistic setting when predicting share prices - in reality an individual would not have access to time t high, low and volume variables to predict close price at time t. By shifting the data backwards one timestep we are able to use the feature variables including the previous day's close price to predict close price at time t.

The Datasets starts from 01/01/2021 and ends at 28/03/24. The entire dataset contains 813 rows containing daily information about the stocks share price. The Initial Dataset contained 6 features: Open, High, Low, Close, Adjusted Close and Volume. We expanded on these features through feature engineering by adding moving averages and exponential moving averages at 7 days, 14 days and 21 days. 7 day standard deviation, relative strength index, price momentum, range and price momentum. We then evaluated the strength of the correlation of these variables with the close price using a correlation matrix and found moving averages were very strongly correlated. The range and standard deviation were also moderately positively correlated with the target variable. We determined the volume, relative strength index and price momentum were not useful parameters and so these were omitted for the analysis.

Across the models a train-test split was performed with the training using the first 70% of the dataset, the next 15% was used for the validation set and the final 15% was used for the test set in order of date. A Validation set was included to test models performance and tune parameters

before introducing an unseen test set and is also useful for comparisons for testing model performances on different datasets.

Methodology

In order to choose the most effective model for the prediction of stock prices across different time periods we built several Machine Learning models which were mentioned in the literature and used them to predict prices for the next 7, 30 and 90s across 3 stocks and determined which models performed the best using MSE, RMSE and MAE. In this section we will provide an overview of the models, provide an explanation about them and discuss the hyperparameters used

ANN: In the development of this ANN, various models were created with adjustments in each which were then evaluated against each of their respective evaluation metrics. The use of additional features was explored and it was found that limiting these to 'close', '12-day-EMA' & '26-day-EMA' (with those values chosen as backed by literature) proved most beneficial – although comparably similar to only using close values. To enhance the model, an ensemble approach was employed to incorporate an XGBRegressor to predict the residuals between the actual and predicted values from the model and then add these corrections onto the initial predictions. An EarlyStopping callback and different batch sizes were also explored to prevent overfitting while a LearningRateScheduler with a step decay function reducing the learning rate throughout the epochs was experimented with, yet no significant improvement to the model was evidenced. The addition of LeakyReLU and Dropout layers were also tested but were both marginally outperformed by the model ultimately developed. The final model used 2 hidden 'dense' layers with ReLU activation functions and an adam optimizer using the aforementioned features resulting in the best performing model across all prediction periods.

Random Forest: Random Forest is an ensemble method that uses multiple decision trees and bagging technique (bootstrap samples) (bagging) from the dataset to predict the Close Price of stocks with the results of each tree aggregated to provide overall predictions. Random Forest was selected to be one of the models tested due to its use of bootstrap samples and multiple decision trees which can achieve a lower variance and less at risk of overfitting as the bootstrap samples each use a different sample of data resulting in a more generalised model. At each decision tree node a sub selection of 2 of the 4 predictor features for this particular model and chooses the feature that provides the best split of the data using metrics such as mean squared error. In the model two hyperparameters were used with the the n_estimators hyperparameter set to 250, after using grid search to find the optimal number of decision trees that produces the lowest RMSE, and the Random State hyperparameter was set to 42, which is standard practice in data science, so that each bootstrap sample is randomly selected but the sampling is reproducible when rerunning the model as the same data points and features would be selected.

Ridge Regression: Ridge Regression is a form of linear regression that applies a L2 regularisation penalty to the loss function. This is an appropriate choice for the dataset due to the model only using a few strongly correlated features, which is referred to as multicollinearity, that can lead to

overfitting. Ridge Regression applies a L2 penalty which reduce the impact of multicollinearity and achieve smaller coefficients as the model is less sensitive to individual variables and reduces overfitting leading to improved generalisation performance with the model being able to be accurately applied to unseen data as the penalty reduces the variance of coefficient estimates. Ridge Regression has one hyperparameter denoted as alpha which controls regularisation strength and for the model and after testing was set to 1 in order to provide a good balance between the model complexity and the generalisation performance to avoid overfitting.

SVR: Support Vector Regression is a machine learning algorithm based on the Support Vector Machines classification method that is used to predict a continuous target variable and uses several hyperparameters. The Kernel function determines the type of decision boundary that the model can learn. During the grid search, Linear, Polynomial and Radial Basis Function were tested to see which performed the best for stock price prediction with Linear Kernel being selected via grid search as exploratory analysis showed a strong linear relationship between the target and explanatory variables. This means that the data is not mapped to a higher dimensional space to handle nonlinear relationships, but instead finds the best linear fit in the original input space.The expression for the linear kernel function is as follows K(x,yi)=xTyi. Where, x and y_i are the feature and support vectors, respectively.This represents the model trying to find a linear hyperplane in the feature space that can be as close as possible to the target value of all training samples to maximise the separation of different categories of data points, i. e., the rise and drop of stock prices.

The target is to minimise the epsilon-insensitive loss function, which defines the tolerance for what the model considers errors, while maintaining the prediction error within a certain range. The grid search returned the optimal value for the epsilon hyperparameter as 0.5 which promotes a lower tolerance for what the model considers to be an error. The last hyperparameter used in the model is C which determines the margin size and penalties for errors during training. For the model the value of 1 was selected via grid search for C providing a more generalised model that applies lower penalties and less overfitting during the training set. Using a small C and Epsilon ensures that the model is less concerned with minimising the training errors but also has a lower tolerance for what it perceives to be errors. This helps maintain the model's accuracy when the C hyperparameter is set to a lower penalty for correcting errors to prevent overfitting.

Linear Regression: Linear Regression models the relationship between an explanatory and response feature using the linear function Y = mX + b **.** It is considered to be a relatively simplistic model and was selected for the project to serve as a baseline for comparison and evaluation of other more complex regression models. Whilst a linear regression model may not be able to capture the full complexity of price movements and other models perform better it generally still produces reliable results and serves as a strong foundation for the initial model development process.

ARIMA Model: ARIMA looks to use a value's previous lags to predict future values. We can control the number of lags that are part of the model, the levels of differencing applied (this attempts to stabilise the mean and remove seasonality and trends from the data) and the moving average order that is applied. The hyper-parameters that were used as part of the grid-search were autoregressive and moving average. As part of this project ARIMA will be taking the previous close prices values to

predict close price at time t, autoregressive parameter establishes the number of lags to introduce into the model and moving average is used for including x many forecast errors into the model.

Naive: Naïve forecasting takes the last value in the training set and projects that forward for the length of the test set. This technique is generally used as a benchmark of performance – if the results are worse or similar to this forecast we deem the forecasting performance of the model as poor.r

Stateful LSTM & LSTM: LSTM looks to build upon the simple RNN model by increasing its ability to understand long-term dependencies in data. It does this by introducing a cell state which contains an input gate, forget gate and output gate. These gates determine what data should be retained within the cell state through timesteps. Input gate controls the volume of new information that is introduced, the forget gate decides what information should be discarded from the cell state at time t and lastly the output gate determines the volume of information that should be discarded from the cell state to move on to the next timestep at t+1. This cell state in addition to the hidden state builds upon RNN by reducing the possibility of the exploding/vanishing gradient problem to capture long-term dependencies more-easily in data.

LSTM for this project looks to take the information from 11 features using a variety of timesteps that have been determined by a grid search to predict the close price at time t. This timestep is a crucial hyper-parameter in the LSTM model which determines how many time periods we are going to look back upon the predictor variables to determine the target variable at time t. We have used a deep neural network with two LSTM layers of 512 neurons each respectively and with one dense layer with 1 neuron for output. The LSTM model resets the internal states at time t+1 after batch has taken place at time t. This therefore treats each batch independently and internal states are reset from scratch per batch. When using stateful LSTM's, we can capture long-term dependencies more easily as the initial states are not reset between batches.

CNN: CNNs known as Convolution Neural Networks are trained for classification and computer vision tasks. CNN has three layers Convolutional layer, a Pooling layer and a fully-connected layer. In this project PyTorch's nn.Conv1d layer is represented as the convolution layer, with in_channels=1, out_channels=30 and kernel_size=10 for predicting the closing price of the stocks. The pooling layer is represented by nn.AdaptiveMaxPool1d function and fully-connected layer is represented by nn.Linear.

In the preprocessing phase, the function creates a sequence of data from across a moving window. The sequence is normalised by dividing the reference value and subtracting by 1.
In the training phase, gradients are computed to update weights and biases, here Huber loss function is used since it is less sensitive to outliers.

RNN:RNNs known as Recurrent Neural Networks are trained to handle sequential data like words, sentences and time-series data. RNNs are made of neurons, each neuron has an input, output and hidden layers. Information is processed in the input layer, an output layer provides the result, and data analysis and prediction happens in the hidden layer.In this project PyTorch's simple RNN model nn.RNN is used, with input_size, hidden_size and num_layers as its parameters.While training, each batch of input sequences is passed through to the model for forward propagation to get the output sequences. Backpropagation is used to compute gradients and Adam optimizer is used to update the

model parameters to minimise the loss function.RNN's ability to learn long-term dependencies is hindered by vanishing and exploding gradients

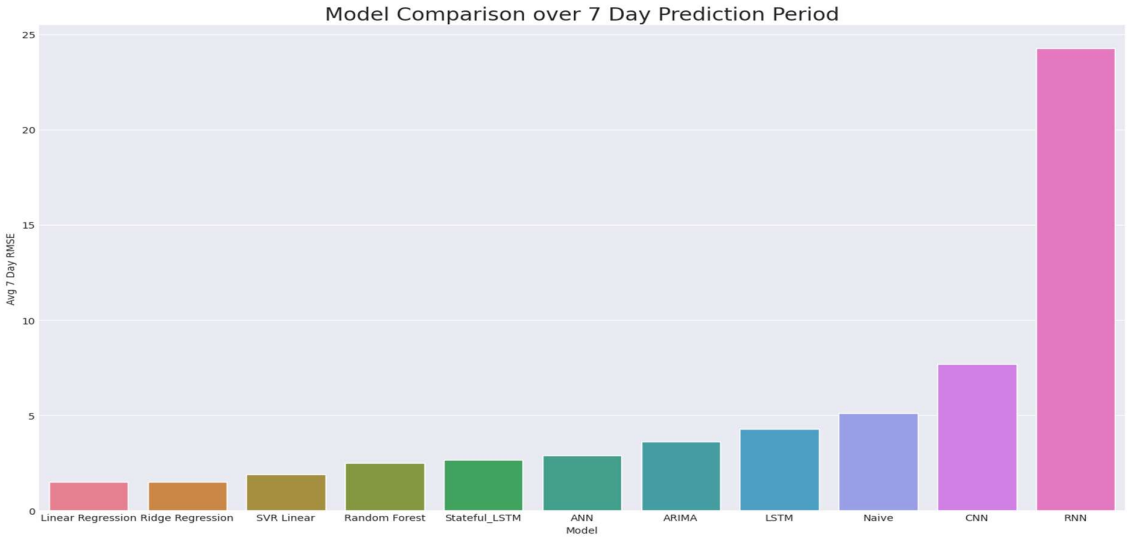
Experimental Setting

The development set represents 15% of the total data and was used as a reference to check the accuracy of the models before using them on the test set. In order to establish the most suitable hyper-parameters, a number of the models used grid-search to work out which values for the parameters are most accurate. In order to do this, we used analysis on mean squared error to work out which were the most suitable. Those particular hyper-parameters are discussed in the methodology.

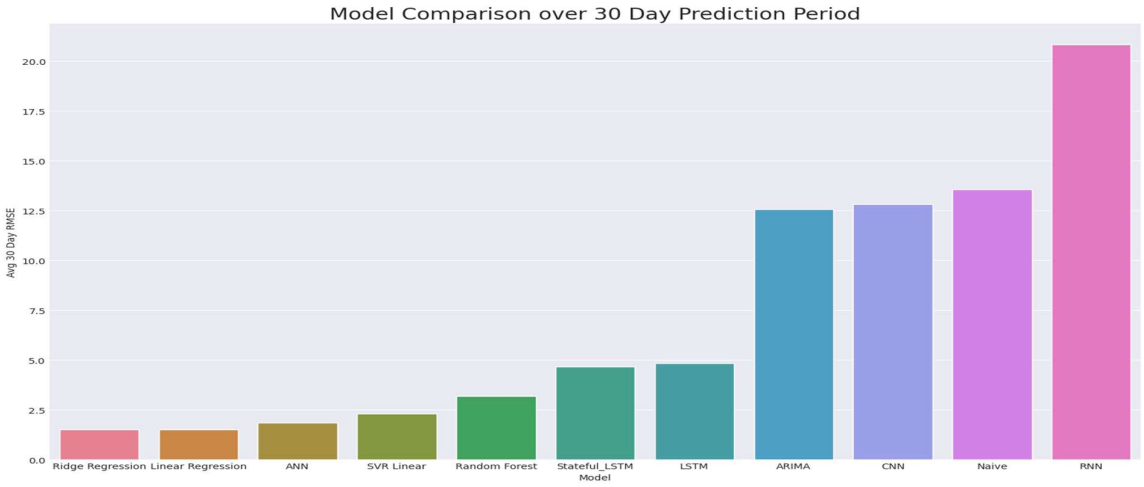
Results

The results provided below show the data frames that were created from the notebook alongside the graphical output of the results. These representations demonstrate an average of the models Test RMSE across all 3 different shares. From our analysis, we wanted to review whether the distribution of the data makes a difference to the predictive capability of the model. There were no notable differences between the results across models between shares and the ones shown below we therefore will not be commenting on these here. Below you will find the mean of the Root Mean Squared Errors across various prediction periods across all the shares (Apple, Johnson & Johnson and Tesla). You will find the data frames associated with the visualisations mentioned below in the notebook.

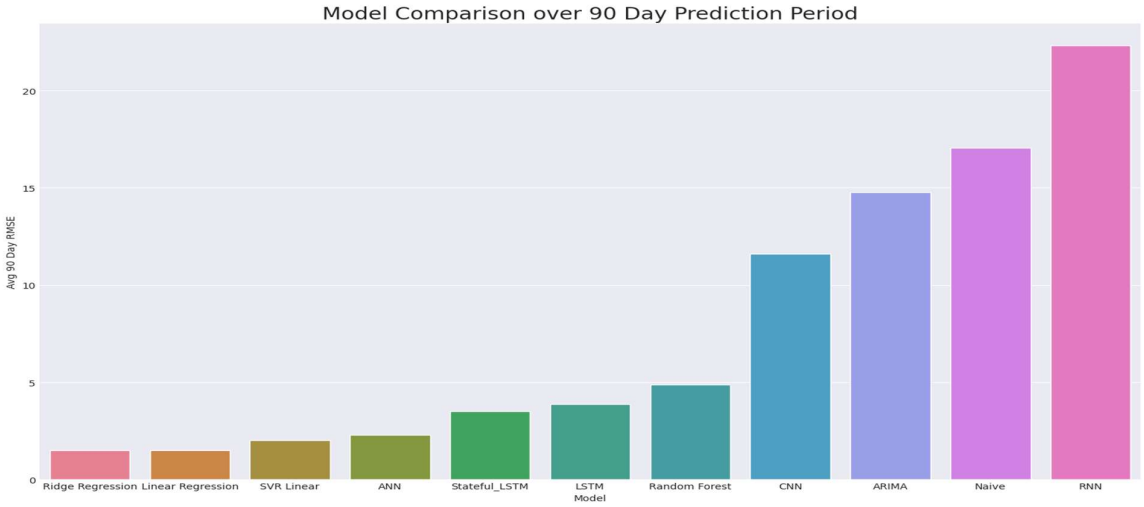
Short-Term Prediction (7 Days)



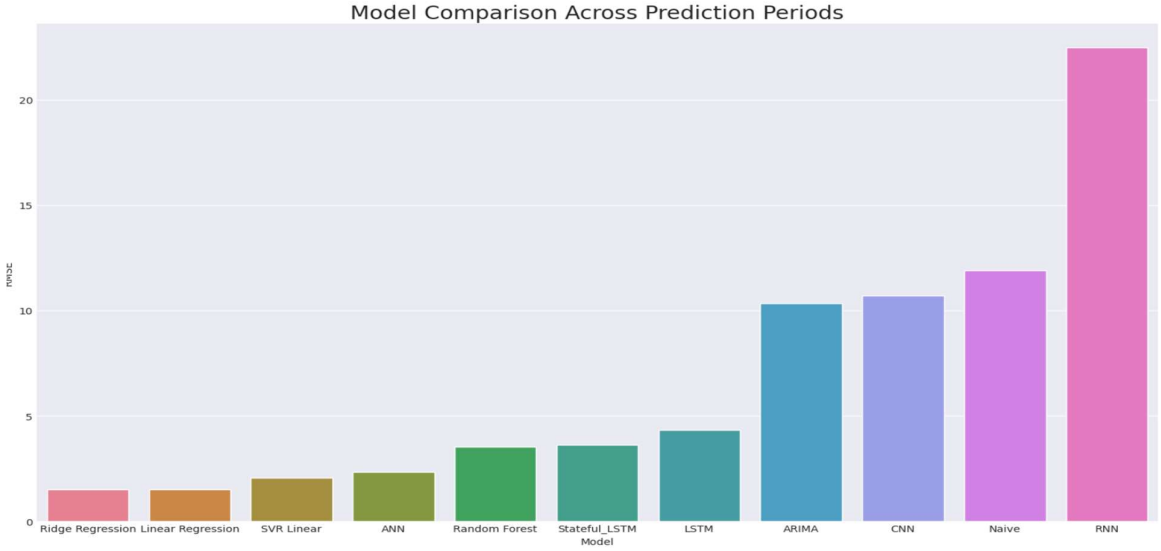
Medium-Term Prediction (30 Days)



Long-Term Prediction (90 Days)



Overall Model Performance



Analysis

From our analysis it's clear that most of the machine learning methods perform very well in attempting to predict the close price. Relative to the baseline naive model, only RNN lagged behind on its predictions. We observe that regression (ridge and linear) performs very well alongside SVR and ANN across all the prediction periods. LSTM accuracy remained relatively consistent across prediction periods, where a stateful LSTM model performs marginally better over shorter, medium and long-term predictions. ARIMA performed relatively well over a shorter prediction period which aligns with the review of the literature that we undertook but struggled over longer-term periods but still performed better than the naive model overall and for all prediction periods. CNN performed poorly across all the prediction periods and only marginally performed better than the naïve baseline model.

During our assessment we did encounter issues with the Random Forest analysis as the scaling undertaken upon the training set prevented the predictions from being able to exceed the values dictated by the scaling due to extrapolation. This did affect the analysis for a few of the predictions that were made and therefore reduced the accuracy of the model. Aside from this we didn't encounter other issues in the performance of the models and the code.

Conclusion

The outcome of our results clearly demonstrate that modern Machine Learning techniques are far superior to baseline models such as the naive model and the time series method in ARIMA. This is an outcome that we expected given the numerous sources that suggested the same output. We observed that between ML models there is great variability when it comes to accuracy, there was poor performance in the RNN model particularly over longer time horizons which was expected given the issues surrounding exploding gradients but surprisingly it didn't perform better than the naive model. CNN also struggled with only a marginal improvement to the naïve model and worse predictive capability than ARIMA. The adaptation from RNN to LSTM performed markedly better, likely due to the introduction of the cell state which eliminates the vanishing / exploding gradient issue and enables the model to capture temporal relationships more effectively. Our analysis demonstrated that using a stateful system which increases the level of memory retained from previous time sequences due to not resetting the output values from batch to batch is better for predictions. This highlights that increasing the retention of information for previous values increases predictive capacity when predicting the close price.

Random Forest performed relatively well but was impacted by scaling issues previously mentioned. We experienced stronger results in ANN using a more basic neural network compared to LSTM. These results suggest that introducing multiple lags to the predictor variables to form part of the predictive process for the next days close price isn't as impactful as we thought. It suggests that simply using the previous days variables at t-1 can be more impactful than stretching to t-n timesteps to predict close price at time t.

Linear and ridge regression performed the best amongst all the models and given our exploratory data analysis at the beginning of the project this is perhaps unsurprising. Regression works particularly well when predictor variables are all linearly related - during our exploratory analysis it was clear that all the predictor variables used as part of the analysis were all almost perfectly linearly related to the close price. (please refer to the pair plot provided in the notebook) When introducing non-linear data we would expect that SVR, ANN and LSTM would outperform these linear models. (Linear Regression) Even after undertaking feature engineering we still observed strong linear relationships between these features and the target variables and collectively this appears to have created a strong position for linear regression to perform so well.

Recommendations for Future Research

To build upon this research individuals could look into web scraping and using sentiment analysis from Twitter data. This could involve filtering the accounts which involve the word “news” in their bio to reduce the degree of poor information being fed into the model and looking at the ratio of positive words to negative ones and evaluate how this impacts the share price. This would provide an arguably more comprehensive model which would likely introduce non-linearity to the model and enable more advanced complex models such as LSTM to really excel. This use of Natural Language Processing (NLP) could be expanded upon further by analysing individual company reports by brokers and company accounts to extract relevant features such as P/E ratios, % increase in revenue and Earnings per share for example. Further additions could also more simply introduce macroeconomic data such as inflation, interest rates and GDP as predictor variables to add further rigour to the model. Further research could also look to predict over a longer horizon such as 365 days to evaluate how errors develop, we observed increasing performance for the LSTM model over longer horizons and we may observe over a longer horizon that the model performs markedly better.

Bibliography

Ayodele A. Adebiyi, A. O. (2015). Stock Price Prediction Using the ARIMA Model . *IEEE*, 1-7.

Indronil Bhattacharjee. (2019). The proposed methods by which to forecast share prices have changed with the evolution of Machine Learning techniques ranging from Support Vector Machines (more specifically Support Vector Regression), Random Forests to Long Short-Term Memory (LSTM) model. *Research Gate*, 1-7.

Islam, M.R.; Nguyen, N. Comparison of Financial Models for Stock Price Prediction. *J. Risk Financial Manag.* **2020**, *13*, 181. https://doi.org/10.3390/jrfm13080181

J. Jagwani, M. Gupta, H. Sachdeva and A. Singhal, "Stock Price Forecasting Using Data from Yahoo Finance and Analysing Seasonal and Nonseasonal Trend," *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, Madurai, India, 2018, pp. 462-467, doi: 10.1109/ICCONS.2018.8663035.

Lei li, Y. W. (2017). Research on machine learning algorithms and feature extraction for time series. *IEEE International Symposium* , 1-5.

M., H., E.A., G., Menon, V.K. and K.P., S. 2018. NSE Stock Market Prediction Using Deep-Learning Models. Procedia Computer Science 132, pp. 1351–1362. doi: 10.1016/j.procs.2018.05.050.

Mahla Nikou, G. M. (2019). Stock Price Prediction using DEEP Learning algortithm and its comparison with machine learning algorithms. *Wiley*, 1-12.

Mehar Vijh, D. C. (2020). Stock Closing Price Prediction using Machine Learning Techniques. *Elsevier* , 1-8.

Pengfei Yu, X. Y. (2020). Stock price prediction based on deep neural networks. *Deep Learning for Big Data Analytics*, 1-20.

Sima Siami-Namini, N. T. (2018). A Comparison of ARIMA and LSTM in Forecasting Time Series. *IEEE*, 1-8.

Sreelekshmy Selvin, V. R. (2017). Stock Price Prediction Using LSTM, RNN & CNN-Sliding Window Model. *Research Gate*, 1-6.