**Data Analytics Group Project Report**

*Zicheng Yan – Group 9*

**Questions to Explore:**

Question 1: What features are important for stock prediction?

Based on past experience with the Weekly dataset, the last closing and opening price tend to have higher contributions. And according to my past experience, technical indicators like MACD or volume are also useful

Question 2: The two data sets in this project, TSLA and KO are both stock prices, but has demonstrated very different trend in the past. Would same model have very different performances on two data sets? Or would the performances be similar since both data sets are stock prices?

I expect same model to have vary results for different stock.

**Data Description**

The two data sets used in this project are retrieved from yahoo!finance. They include the Date, Opening price, Highest price Lowest price, Closing price, Adjusted Closing price, and Volume of two stocks, Tesla, Inc. (TSLA) and The Coca-Cola Company (KO), for the period of 5 years from Dec 12, 2018 to Dec 08, 2023. The links are as follows:

<https://finance.yahoo.com/quote/KO/history?p=KO>

<https://finance.yahoo.com/quote/TSLA/history?p=TSLA>

**Analysis Methods and Results**

First of all, I created a few functions for future use, including: adding lagged features, calculate mean absolute percentage error, and adding technical indicators MACD as features. Then, use the functions created, add lagged features and technical indicators to dataset. Extract day, month and year from date. Drop all rows with na, since the calculation of MACD and lagged features require previous observations, there should be some na rows. Take away all non-lagged variables so that no future data is leaked to model. Perform train test split. Now the preprocessing is completed for our first model.

The first model I selected was tree regression. Because I want to start with a simple one as benchmark. The processed training data has discrete variables like day and month in it, so regression may not be appropriate. After fitting the models with two training data sets and use previously defined functions to calculate the metric, I got two trees and one table, as shown below:

Table 1 Tree Model Metrics

|  |  |  |  |
| --- | --- | --- | --- |
|  | MSE | RMSE | MAPE |
| KO | 1.276 | 1.129 | 1.582% |
| TSLA | 949.266 | 30.810 | 13.165% |



Figure 2 KO Tree

Figure TSLA Tree

The only two variables in the tree graphs are close\_lag1 and high\_lag\_1. This partially validated my assumption that last day prices are among the most important features. And from the table we can see that although both data sets are consisted of stock prices and data, the same model yield very different results on them. The difference between MAPE is larger than 10%!

Next, I decided to try a model with similar mechanism but yield more complexity. Thus, I went with random forest model, which is basically the combination of numerous tree regression models using different sets of predictors. To train a machine learning model like random forest model, a train-test split must be conducted. In this case I used a split ratio of 0.2.

Table 2 RF Model Metrics

|  |  |  |  |
| --- | --- | --- | --- |
|  | MSE | RMSE | MAPE |
| KO | 0.471 | 0.686 | 0.951% |
| TSLA | 68.196 | 8.258 | 3.060% |

The random forest model in R is really smart with a good set of parameters so I did not made too much adjustment. As expected, the results are much better compared to tree model, especially for TSLA, which is more volatile and less seasonal. And I had the model passed feature importance, so I was able to visualize it. The feature importance is measured by percentage increase in MSE based on perturbation.

After a tree-based model is trained, the values of a particular feature are randomly shuffled across the observations in the dataset. This shuffling breaks the relationship between that feature and the target variable. When the relationship between an important feature and the target is broken, the model's predictions become less accurate, leading to an increase in MSE. The more important the feature is, the larger the increase in MSE when it's shuffled. This increase is often expressed as a percentage relative to the baseline MSE (without any shuffling).

As shown below, ‘Month’ is the third and fourth important feature in the two models, respectively. Such result is unexpected. Although I did expect ‘Month’ to have some contribution to the model, I never thought it could be this much. I suspect this is because public listed companies usually file their quarterly and yearly financial reports in the same months of a year. Technical indicators Hist and MACD are more important for KO predictions than for TSLA predictions. A possible explanation is the price of Tesla is more affected by factors and events outside the stock market.





Next, I tried Recurrent Neural Network model on the datasets. Before transformer, the RNN is probably the most popular deep learning model for analyzing sequential data. With RNN, I can define a time window for the model to learn from past data, so there is no need to add lagged features. Unlike random forest, RNN is very sensitive to the scale of input, so I first need to normalize all the variables. Also, given the cyclical nature of variables ‘Month’ and ‘Days’, I did something different. I used trigonometric to encode them while keeping the cyclical information.

After completing the pre-processing, I trained the model multiple times and tuned the parameters as well as the structure of the neural network accordingly. The idea was to use long-term short-term memory layers with large numbers of neurons for the first layer to capture as much information and features as possible. Then shrinking the number of neurons layer by layer while also drop a significant portion to reduce overfitting. However, the results are not as good as I expected. As shown in the table below, for KO prices, the MAPE and MSE are larger than the simple tree model. For TSLA prices, the metrics are only slightly better than the tree model and still much worse compared to random forest.

Table 3 RNN Model Metrics

|  |  |  |  |
| --- | --- | --- | --- |
|  | MSE | RMSE | MAPE |
| KO | 2.120 | 1.456 | 1.895% |
| TSLA | 737.899 | 27.164 | 7.351% |

By analyzing the learning curve of the RNN models, I inferred that the main reason of the poor performances might be saturation, which is very likely to happen when dealing with long sequences and having variables scaled to one. With the current scaling and tanh as activation function, the neurons in the network squashes the input values into a very narrow range, leading to gradients that are either too small (vanishing gradient problem) or too large (exploding gradient problem). Although using gated structure with LTSM layers should solve a portion of the problem, the learning curves still show a very unstable loss on validation sets. Saturation is the most reasonable knowledge according to my knowledge.

图表, 直方图

描述已自动生成图表, 折线图

描述已自动生成

**Conclusion**

Based on the analysis, I have gained sufficient knowledge of the data sets to answer the questions in the first part of the report. As I expected, technical indicators like MACD and closing prices of last one to two days are useful when predicting stock prices. Surprisingly, month is also a crucial factor. The performances of different models are largely different on the two data sets, and they all tend to perform much better on KO price prediction. So when working on quantitative strategies based on ML or DL techniques, one might want to start with stocks with less volatility and more observable seasonal trends.

**Appendix**

Figure KO Tree



Figure TSLA Tree



Figure KO RF



Figure TSLA RF

