Winton Stock Market Challenge

Afif Mazhar **Namin Rahman**

Project Background

“Do you laugh (and then get down to work) in the face of terabytes of noisy, non-stationary data? Winton Capital is looking for data scientists who excel at finding the hidden signal in the proverbial haystack, and who are excited by creating novel statistical modelling and data mining techniques.

In this recruiting competition, Winton challenges you to take on the very difficult task of predicting the future (stock returns). Given historical stock performance and a host of masked features, can you predict intra and end of day returns without being deceived by all the noise?

Research scientists at Winton have crafted this competition to be challenging and fun for the community while providing a taste of the types of problems they work on everyday. They're excited to connect with Kagglers who bring a unique background and creative approach to the competition.

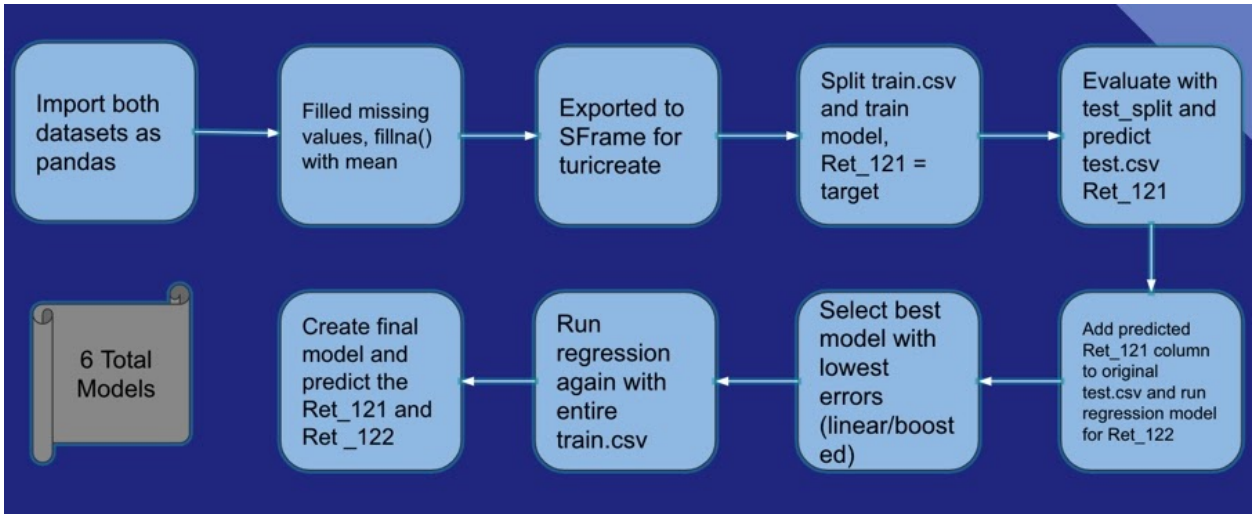
Winton is offering cash prizes to winning teams as a reward for their work, but the intent of the competition is not commercial. The intellectual property you create remains your own and will be evaluated in the context of suitability for employment.“ - Winton Capital

The Winton Stock Market Challenge focuses on generating a prediction model for stock price fluctuation over a series of 5 days. Stock prices tend to fluctuate from a matter of seconds to hours across various days and it is imperative to understand this irregularity through a series of analyses. Winton conducted this challenge to determine the fastest, yet most accurate model for prediction.

The data consists of 5 days of times, days D-2, D-1, D, D+1, D+2. We are given return of days in D-2, D-1, and part of day D. The goal is to predict the rest of day D, and D+1, and D+2. 180 minutes of data for day D (labeled as Ret_2 through Ret_180) is given in the training set, while the test set has only the first 120 minutes. 25 features are presented, labeled as Feature_1 to Feature_25. In order to accommodate for the class, we predicted only the next two minutes (Ret_120 and Ret_121) after the tested 120 minutes to predict the stock price variation.

Methodology

We used the Pandas, MatPlotLib, Numpys, and TuriCreate packages to create the analysis for the dataset. Shown below is the logical step by step process used to design the model.



First, we used Pandas to import both datasets into dataframes so that we could fill the empty values in the columns with their respective mean values. Afterwards, we had to export them to .csv files so that they could be transferred to sFrames using TuriCreate. Both the train.csv and test.csv files were filled. The code is listed below.

```
train_df = pd.read_csv('train.csv')
test_df = pd.read_csv('test_2.csv')

train_df = train_df.fillna(train_df.mean())
train_df.to_csv('train_df.csv')
test_df = test_df.fillna(test_df.mean())
test_df.to_csv('test_2df.csv')

train = tc.SFrame('train_df.csv')
test = tc.SFrame('test_2df.csv')
```

Afterwards, the train.csv file was split (70/30) to train the model and subsequently test the accommodated variables. We used all 25 features of the dataset to generate a regression model that would be accurate in predicting Ret_121 for day D. In order to find the best fit model,

turicreate has a function listed below that showed us the top two regression techniques to use: linear regression and boosted regression.

bestfit_model = tc.regression.create(train_splitted, target = 'Ret_121', features = my_features)

After splitting the variables and finding the best regression techniques, we conducted a linear regression with Ret_121 as the target variable on the train.csv file test data. Then, we conducted the linear regression on the test.csv file and added the predicted column to the original dataset to conduct another linear regression to find Ret_122. Below is the RMSE value of the first and second linear regression conducted.

LR Ret_121 → {'max_error': 0.017016736962887608, 'rmse': 0.0010693758180243516}
LR Ret_122 → {'max_error': 0.012569670489037295, 'rmse': 0.0010997925308388551}

The same procedure was conducted for boosted regression models, beginning with splitting the variables (70/30) and using Ret_121 as the target variable on the test data for the train.csv file. Then, we used the model to find the Ret_121 column on the test.csv file and reiterated the column to predict the Ret_122 column. Below are the error values for both boosted regression models.

BR Ret_121 → {'max_error': 0.034839583560824394, 'rmse': 0.014192864682949889}
BR Ret_122 → {'max_error': 0.028529010713100433, 'rmse': 0.014161963262727216}

We concluded that the linear regression model is the better technique, which corroborates with turicreate's prediction of the specific regression model to use. Selecting the linear regression model, we conducted the model analysis once more, this time using all of the data in the train.csv file for the model creation. The error values are listed below.

Final Ret_121 → {'max_error': 0.015999181242122528, 'rmse': 0.0010123129800176513}
Final Ret_122 → {'max_error': 0.022732735071063777, 'rmse': 0.0010811399566760813}

Analysis

Figure 1

This graph is our first linear regression prediction of Ret_121 compared to the Ret_121 from our train dataset. We decided to compare our prediction to the train, so we can see how similar it is and see if there is any discrepancy. When creating the graph, we decided to run 60 samples to see if there are any huge jumps. Overall, our prediction was very similar to the train's Ret_121. The first discrepancy was around the 10th sample, the train's Ret_121 stock return had a sudden spike, while our prediction did not. The second most noticeable discrepancy is in between the 20th and 30th samples. While the train's Ret_121 stock return dropped, the predicted Ret_121's stock return rose. One thing we noticed there was that before the train dropped the predicted one dropped also, then rose. This is a good sign, because there is still a common pattern between these 2. The third discrepancy we noticed was when near the 40th sample, our predicted Ret_121's stock return decreased, while the train's Ret_121 stock return increased. The final discrepancy we noticed was in between the 50th and 60th sample, the train's Ret_121 decreased drastically while our predicted value stayed normal. Overall, our prediction was similar to the train values, obviously there are some major differences, but the stock market is spontaneous.

Figure 2

Figure 2 is our first boosted tree regression compared to the Ret_121 from the train dataset. As you can see that there are hardly any similarities. Through 60 samples the boosted tree regression kept on a constant pace. Overall we can conclude, using the boosted tree regression is not an optimal solution to predict the stock market. Another thing we can conclude is that the boosted

tree regression did not have specific features and that it did not have specific data to help its regression.

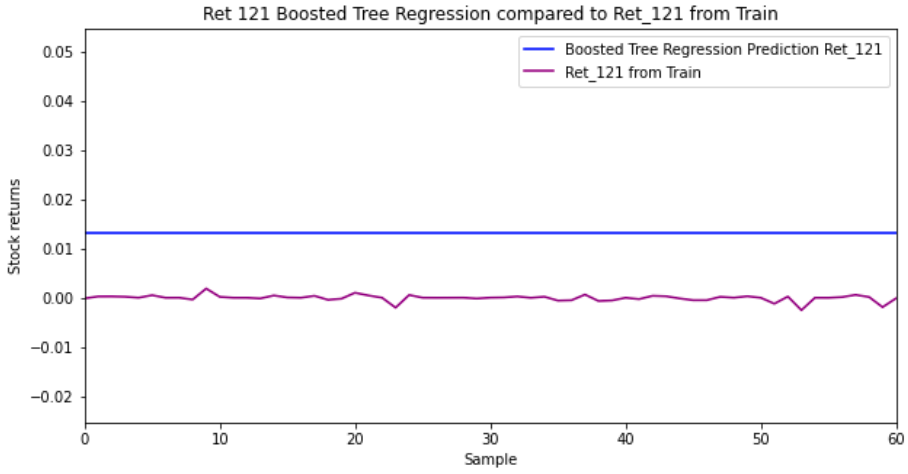


Figure 3

Figure 3 is our first Ret_122 regression model compared to the Ret_122 from the train dataset. Out of all of our models, this prediction had the smallest value for max error and we can see that on the graph. Once again we ran 60 samples to compare both datas. Overall, both data were pretty similar, as we should expect from our max error. There are only a few discrepancies while looking at the graph. The first one comes around the 10th sample, where we can see the train's Ret_122 decrease and our predicted Ret_122 value increase, but afterwards they are almost exact. Our next discrepancy is the 20th sample mark, our prediction increases while the train's Ret_122 decreases drastically again, but it does not get smoother. The next discrepancy is in between the 20th sample and 30th sample. Our prediction decreases while the train increases. In a short span our predictions Ret_122 and train's Ret_122 have some glaring inconsistencies, but soon they start being consistent. Our final discrepancy is in between the 50th and 60th sample, where the train's Ret_122 jumped dramatically while our prediction stayed on a constant pace. Overall this is our most accurate prediction out of all the ones we did.

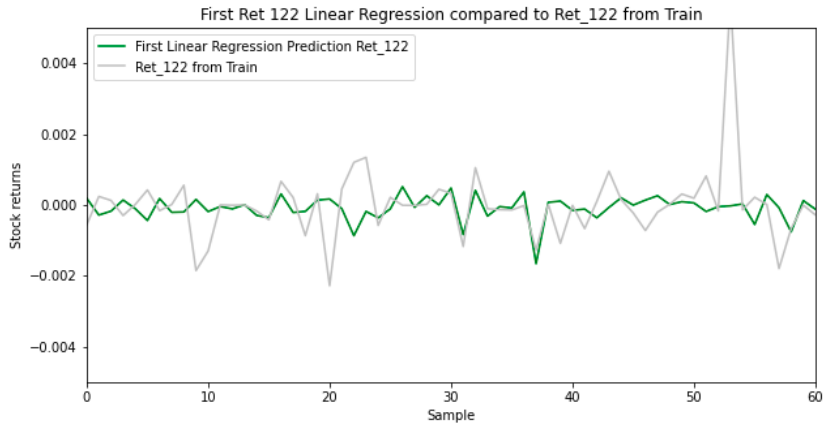


Figure 4

Figure 4 is our Ret_122 boosted tree diagram prediction compared to the Ret_122 from the train dataset. As you can see there are no similarities between the two. The predicted Ret_122 stays at a constant pace throughout the 60 samples, while the Ret_122 from the train dataset shows change. We can conclude the same thing as we did with our Ret_121 boosted tree regression with our Ret_122 boosted tree diagram. Using a boosted tree diagram to predict stock returns is not efficient and should not be used. It does not take in account any specific features like the linear regression does.

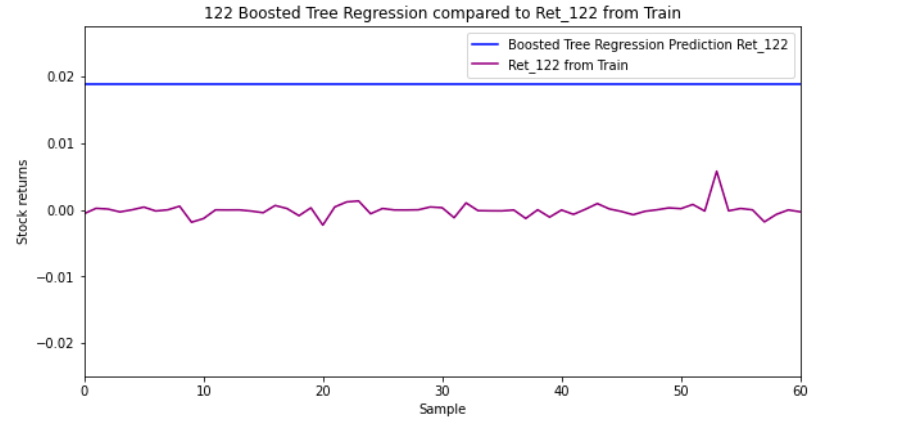


Figure 5

For figure 5 we are comparing our final linear regression for Ret_121 to our Ret_121 from the train dataset. This Ret_121 does not take the data we splitted for the previous Ret_121 linear regression. It takes into account everything. As we can see, that final linear regression data and our Ret_121 from the train are very similar. They have a few discrepancies, but throughout the

60 samples they are pretty consistent. While looking at the graph we can see 4 major discrepancies, the first one being at the 10th sample mark, the train Ret_121 stock return increased by a lot, but our prediction barely increased. The second one is in between the 20th and 30th samples, the Ret_121 from the train data set decreased but our prediction increased. The third one is near the 40th sample mark, where our prediction of and the train's Ret_121 basically did the opposite. While our prediction decreased the train's Ret_121 increased, but soon after the consistency improved. Our final major discrepancy is in between the 50th and 60th sample, where the Ret_121 from the train dataset stock returns plummets, but our prediction stayed consistent from the previous samples. Other than those discrepancies, our prediction was very similar from the train dataset.

Figure 6

Figure 6 is based on our final linear regression of Ret_122 compared to the Ret_122 from the dataset. The difference between the first linear regression for Ret_122 and the final linear regression for Ret_122, is that we did not split for Ret_122. While observing the graph, we can see that there are some consistencies, but glaring discrepancies. It is expected because the max error value was pretty big. There are about 5 huge discrepancies in the graph, the first one comes around the 20th sample. As we can see, the predicted Ret_122 slightly increased from the previous sample, but the Ret_122 from the train dataset is decreasing. Soon after a little consistency is being shown, but not a lot. The next major inconsistency is around the 30th sample mark, where our predicted Ret_122's stock return drastically plummeted, but the train's Ret_122 is increasing in stock return, but once again some consistency is being shown in future samples. The next major inconsistency is in between the 40th sample mark and the 50th sample mark. We can see our prediction's stock return slowly decrease, but the train's Ret_122 stock return increases. The fourth discrepancy is in between the 50th sample and 60th sample, we can see that the train's Ret_122 stock return had a drastic increase, while our predicted Ret_122 stock returns stayed consistent with the previous samples. The final discrepancy is near the 60th sample, we can see that our predicted stock return is increasing while the Ret_122 from the train

dataset is decreasing. Overall there were consistencies between these 2, but there were a lot more glaring inconsistencies.

Figure 7

For figure 7, we wanted to compare both regression predictions for Ret_121. The main reason why is to see how big a difference a split makes for the prediction. Unfortunately, we could not reach our conclusion for this model. As you can see our prediction for the Ret_121 with splitted and our non splitted Ret_121 are exactly the same.

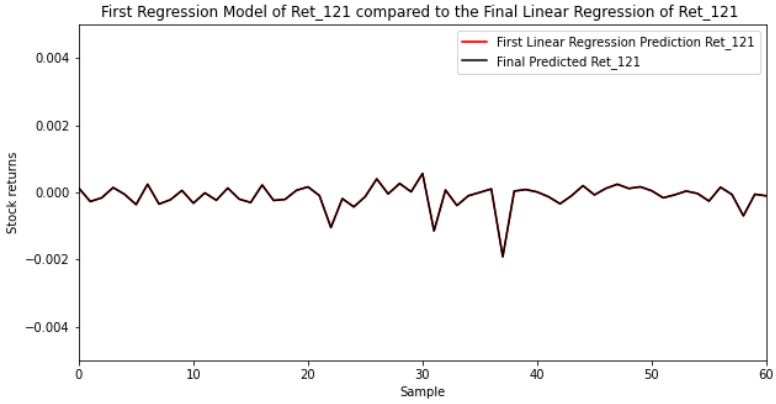
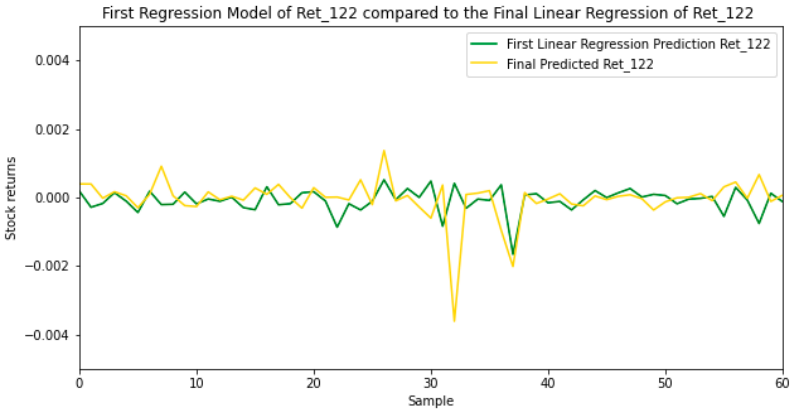


Figure 8

Figure 8 is our first linear regression prediction of Ret_122 compared to our final linear regression prediction of Ret_122. Fortunately, the predictions were different unlike Ret_121. While analyzing the graph, we can see consistencies throughout the 60 samples. There are only a couple of discrepancies while looking at the graph, which tells us there are not that many differences from splitting the data beforehand, but the splitted Ret_122 had the lowest max error. The first major discrepancy we can see in between the 30th and 40th sample. We can see our first

linear regression model's stock returns increasing, while the final one is decreasing by a huge amount. Soon after the future samples become consistent again until the final eye catching discrepancy, which is near the 60th sample. In this one we can see that the final regression prediction for Ret_122's stock return is increasing, but the first prediction's stock return is decreasing. It is not a huge one like the previous ones, but it is still something that will catch your eyes. Overall both predictions were consistent with which other, but these few differences made a huge impact on the max error. The differences between these two are .012 of each other and we can see that the first linear regression prediction of Ret_122 was much more consistent with the Ret_122 from the train dataset than the final linear regression prediction.



Conclusion

Linear regression was the best fit model in comparison to boosted tree regression. The error value for linear regression was smaller than the boosted tree regression which is most likely attributed to the selection of features. The linear regression model had a more specific version of the features with target variables in comparison to a boosted tree regression model that didn't specify which features to use. For Day D, it was a lot more difficult to interpret 2 minutes of data and the predictions behind that timeframe because of the rapidly evolving nature of stock prices. A dip within one minute is sometimes so minute that it doesn't reflect a change in the stock price. Therefore, as the graphs visualize, there are a lot of discrepancies to analyzing stock price prediction.

For a better prediction, finding the most common features amongst the 25 present, such as top 10 or 15, may reduce the max error value slightly more, but the multi-causal elements of the features 1-25 make it more definitive. More features means a more precise result, particularly for stock price prediction. The best model, Ret_122, had the lowest rate of error. We used the original Ret_121 linear model and the prediction for the Ret_121 column to predict Ret_122; thus, we had to add another feature to the model for Ret_122 (26 total features) which concluded that more variables meant a better model.

Turicreate was able to predict the model efficiently. But, if given unlimited access to machine learning techniques, or optimization, then maybe using a different technique onto the prediction model would work better.