MIE 1628H - BIG DATA

FINAL PROJECT REPORT

Apple Stock Price Prediction

Naga Raja Paidimarri

April 13th, 2019

raja.paidimarri@mail.utoronto.ca

1004636717

MEng in Mechanical and Industrial Engineering

1. Introduction & Literature Review

Time series is simply a time-ordered sequence of observations (Wei, 2013), examples include data sets such as daily closing prices, monthly unemployment figures, annual birth-rates, daily rainfall amounts, and goods shipped from a factory. Timeseries has examples in many diverse fields ranging from economics, business, engineering and sciences. By relying on time series analysis, the business and organizations can forecast future demand to plan operations, manage resources to save cost and improve profits. In this project our team performed time series analysis on Apple stock price. By predicting the stock price, a trader can leverage this information by going long or short position.

The most fundamentally distinguishing feature in time series analysis is that the observations are dependent or correlated. Time series analysis requires different methods and commonly used statistical methods based on random samples may not be applicable. For instance, we cannot used cross validation with random split. Traditionally the statistical methods used for time series are:

- Random Walks
- Simple and Damped Exponential Smoothing.
- Holt's linear trend
- Autoregressive Integration Moving Average (ARIMA)
- Simple Moving Average

But over the time, machine learning methods are increasingly seen as an alternative to this traditional **methods and have more accuracy** (Spyros Makridakis, 2018) **but, they are computationally more expensive in comparison to statistical methods**. It is always advantageous to apply both methods because statistical methods could be a simple benchmark for our machine learning models to check If whether the extra effort for constructing the model and the computational requirement is giving us any better predictions. Our group have used Simple moving average as a bench mark model. There are many machine learning models used for timeseries (Nesreen K. Ahmed, 2010) such as Bayesian and generalized regression neural networks, K-nearest neighbour regression, regression trees and SVR.

In general, every time series can be decomposed broadly in to three components

- Trend - Consistent upwards or downwards slope of a time series (Figure below)
- Seasonality - Clear Periodic pattern such as a sine function
- Noise – Erratic or random or unpredictable

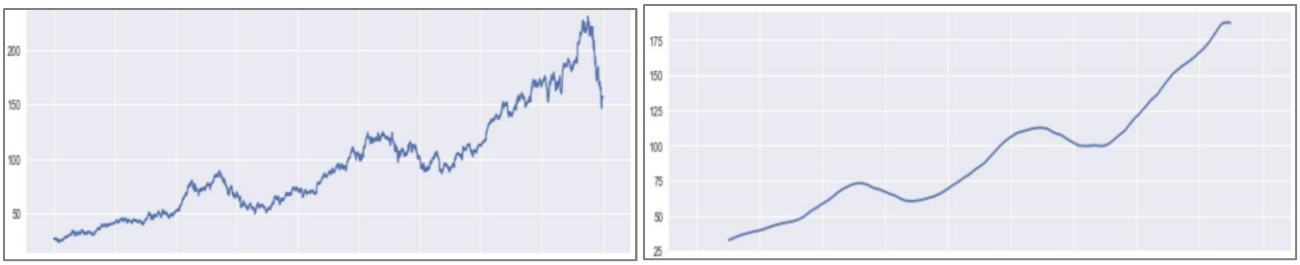


Figure 1: Adjusted closing price of Apple on left and Trend part of the decomposition on the right

2. Business Problem

The aim of our project is to construct and compare machine learning models for stock price prediction of Apple for different time horizons. By precisely predicting the stock prices we can anticipate the expected movement and take a position in the market. The continuing aim is to develop a trading strategy to see the profitability of the classification model.

3. Target Variable

This is a supervised machine learning approach and the target variable or independent variable is adjusted closing price of Apple stock (please assume unless mentioned I am referring to adjusted prices). **The features are selected such that the correlation, time dependent nature of the series will be taken in to account.** The models considered are Moving Average, Linear regression, Random Forests, Gradient Boosted trees and logistic regression (classification for trading). The reason for limiting only to this simple regression models is due to the easy implementation of them in the Spark ML Lib library.

4. Data set

We used data set derived from Quandl website and **the main reason is to get the data of adjusted prices.** The following plot illustrate the difference in prices from both data sources. The yahoo data only provides the Adjusted Close prices where as Quandl provides Adjusted prices of open, high, low, close and volume. The difference in adjusted and actual close price is due to the nature of the market. It happens when a company announces a dividend, or the stock of the company is split. The apple stock is split four times with the most recent and biggest split happened on June 9th, 2014 with 7:1 basis (apple).

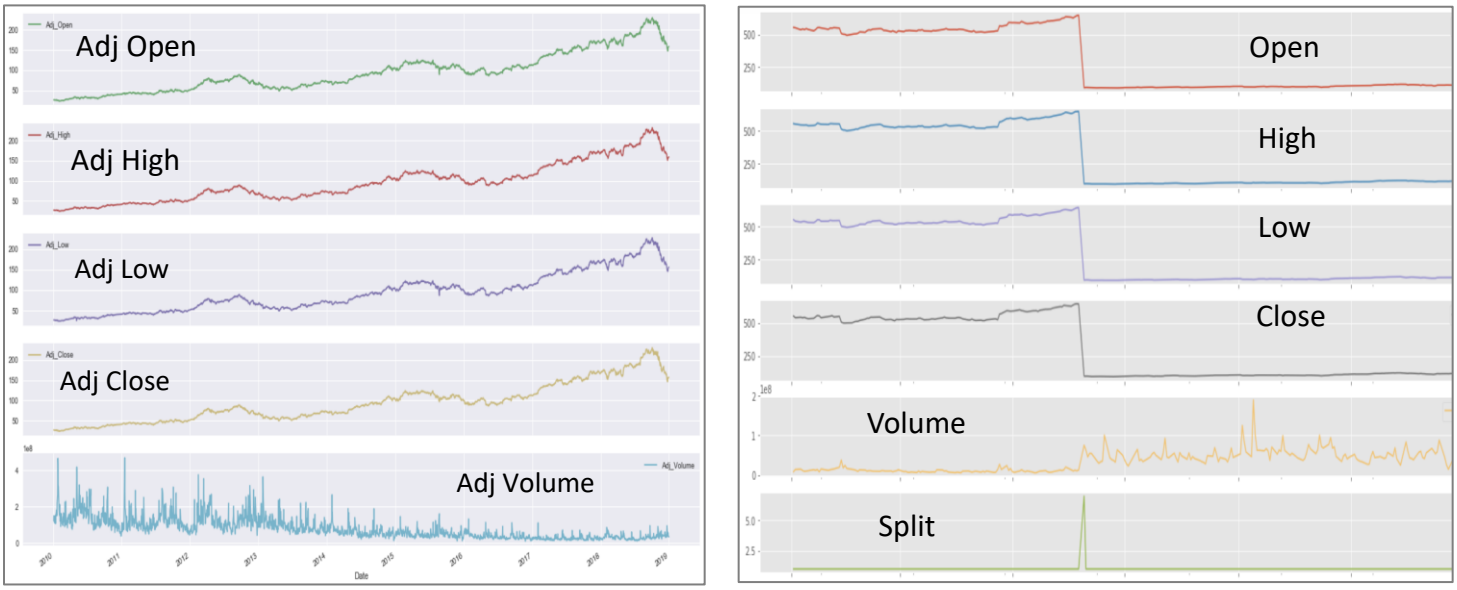


Figure 2 Comparison of Quandl data (left) with Yahoo finance (right)

During the recent split, closing price decreased from 645$ on June 6th, 2014 to 93$ on June 9th, 2014. If we were to train a machine learning model it would be biased and try to account this decrease in price to some attributes unnecessarily. We can always calculate back the Actual close price from adjusted price (investopedia)

5. Moving Average: Benchmark Model

The statistical model we used for benchmarking is simple moving average. Simple moving average is nothing but an arithmetic mean of recent closing prices in a window. If we assume a window of '1' we are saying that tomorrow's price is same as today and if the window size is '3' we are saying that tomorrow's price is average of today and the two days before closing prices. The biggest drawback of SMA is it cannot capture the acute changes in previous day prices.

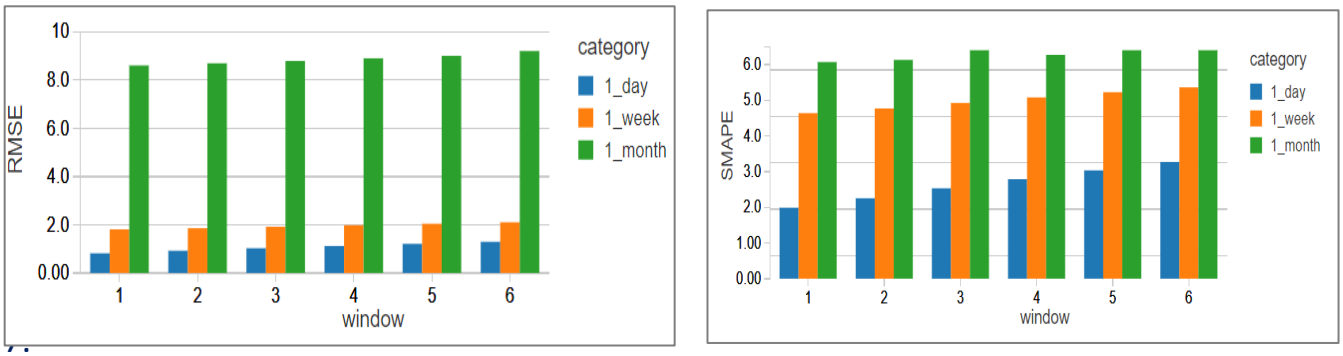


Figure 3 Moving Average model for different window periods. RMSE on left and SMAPE on right

We can see that as period of window is increasing from 1 day to 6 days the RMSE and SMAPE values are increasing for all the three different time horizons. We can say that the moving average is not a good model for longer window periods. Tomorrows price is **explained more** by today's price rather than average of last six days. This also makes sense intuitively and better version of Moving average is exponential smoothing where the weightage given to past prices decreases exponentially. Also, to note is that the moving average is performing worse for longer time horizons.

6. Feature Engineering

The following table represents the features for predicting the closing price. We selected features to account for different parts in a time series as explained earlier trend, seasonality and volatility.

	Actual	Trend	Seasonality	Volatility
1	Adj_Open	Adj_Open – Adj_Close	DAYOFWEEK	StdClose_5
2	Adj_Low	Adj_High – Adj_Low	DAYOFMONTH	StdClose_10
3	Adj_High	Avg_Close_5	WEEKOFYEAR	StdClose_21
4	Adj_Close	Avg_Close_10	MONTH	StdClose_84
5	Volume	Avg_Close_21	YEAR	MinClose_5
6		Avg_Close_0_1_3_5		MaxClose_5
7		Avg_Close_0_1_5_15		MinClose_21
8		Avg_Close_1_5_21_84		MaxClose_84

Figure 4 Features created

I will explain all features by taking a date as example. If we assume, we want to predict the adjusted close price for November 28th, 2018 then the first column "Actual" features correspond to adjusted prices on November 27th, 2018. In "trend" column "Avg_Close_5" corresponds to average of closing prices of 5 previous trading days to Nov 28th which are Nov 21 – 23, 26 -27 and "Avg_Close_0_1_3_5" corresponds to average of prices of 27th, 26th, 22nd and 20th (the large gap is because Saturday and Sunday no market). The seasonality column captures the seasonal variations because every Monday there would be more trading because of market opening. The standard deviation of different window periods such as last 21 and 84 trading days capture the volatility of the stock.

7. Feature Importance

We used linear regression and random forests to identify important features for all time horizons. Our team observed that the importance of features has come out to be same in both models.

	Actual	Trend	Seasonality	Volatility
1	Adj_Open	Adj_Open – Adj_Close	DAYOFWEEK	StdClose_5
2	Adj_Low	Adj_High – Adj_Low	DAYOFMONTH	StdClose_10
3	Adj_High	Avg_Close_5	WEEKOFMONTH	StdClose_21
4	Adj_Close	Avg_Close_10	WEEKOFYEAR	StdClose_84
5	Volume	Avg_Close_21	MONTH	MinClose_5
6		Avg_Close_0_1_3_5	YEAR	MaxClose_5
7		Avg_Close_0_1_5_15		MinClose_21
8		Avg_Close_1_5_21_84		MaxClose_84

Figure 5 Important Features 1 Day

	Actual	Trend	Seasonality	Volatility
1	Adj_Open	Adj_Open – Adj_Close	DAYOFWEEK	StdClose_5
2	Adj_Low	Adj_High – Adj_Low	DAYOFMONTH	StdClose_10
3	Adj_High	Avg_Close_5	WEEKOFYEAR	StdClose_21
4	Adj_Close	Avg_Close_10	WEEKOFMONTH	StdClose_84
5	Volume	Avg_Close_21	YEAR	MinClose_5
6		Avg_Close_0_1_3_5	MONTH	MaxClose_5
7		Avg_Close_0_1_5_15		MaxClose_10
8		Avg_Close_1_5_21_84	MinClose_10	MinClose_80

Figure 6 Important Features 1 Week

	Actual	Trend	Seasonality	Volatility
1	Adj_Open	Adj_Open – Adj_Close	DAYOFWEEK	StdClose_5
2	Adj_Low	Adj_High – Adj_Low	DAYOFMONTH	StdClose_10
3	Adj_High	Avg_Close_5	WEEKOFYEAR	StdClose_21
4	Adj_Close	Avg_Close_10	MONTH	StdClose_84
5	Volume	Avg_Close_21	YEAR	MinClose_5
6		Avg_Close_84	WEEKOFMONTH	MaxClose_5
7		Avg_Close_0_1_5_15		MaxClose_21
8		Avg_Close_1_5_21_84	MaxClose_10	MinClose_84

Figure 7 Important Features 2 weeks

	Actual	Trend	Seasonality	Volatility
1	Adj_Open	Adj_Open – Adj_Close	DAYOFWEEK	StdClose_5
2	Adj_Low	Adj_High – Adj_Low	DAYOFMONTH	StdClose_10
3	Adj_High	Avg_Close_5	WEEKOFYEAR	MinClose_84
4	Adj_Close	Avg_Close_10	MONTH	StdClose_84
5	Volume	Avg_Close_21	YEAR	MinClose_5
6		Avg_Close_84	WEEKOFMONTH	MaxClose_5
7		Avg_Close_0_1_5_15		MaxClose_10
8		Avg_Close_1_5_21_84		MaxClose_84

Figure 8 Important Features 1 month

	Actual	Trend	Seasonality	Volatility
1	Adj_Open	Adj_Open – Adj_Close	DAYOFWEEK	StdClose_5
2	Adj_Low	Adj_High – Adj_Low	DAYOFMONTH	StdClose_10
3	Adj_High	Avg_Close_5	WEEKOFYEAR	StdClose_21
4	Adj_Close	Avg_Close_60	MONTH	StdClose_84
5	Volume	Avg_Close_84	YEAR	MinClose_5
6		Avg_Close_0_1_3_5		MaxClose_5
7		Avg_Close_0_1_5_15		MinClose_21
8		Avg_Close_1_5_21_84		MaxClose_84

Figure 9 Important Features 4 months

1-day prediction means predicting the price for next trading day and 4 months means predicting the price after 4 months (here we used exact dates for making the data so 4 months corresponds to 84 trading days). We can clearly see a pattern, as time horizon is increasing the most predictive features are moving towards volatility. This is expected because for instance when predicting tomorrow's price recent prices are more important than the previous 84 days volatility.

8. Evaluation Metrics

Through out the project we used two evaluation metrics:

1. SMAPE (Symmetric mean absolute percentage error) : Based on relative errors.
2. RMSE (root-mean-square error): It is simply the square root the second sample moment of the differences between predicted and actual values or simply, quadratic mean of these differences.

Both are good because we are dealing with regression problem and they measure how close our prediction is to the actual value. Since we are interested in correctly predicting the price these are right metrics to choose. **The main advantage of SMAPE is, it is a normalized error metric** so we can compare across time horizons because this is normalized for the price and share prices usually follow an increasing trend. Due to this reason we choose SMAPE for the Final results table. Both metrics do not distinguish with the direction of prediction (either predicting more or less that the actual).

9. Model improvement

There are many ways of improving the model performance and our group approached it in two perspectives first, **finding the right amount of training data and second, hyper parameter tuning.**

Train – Test Split:

First, we started with linear regression. We increased the training data for a fixed test data (2 months, 4 months and 36 months) and calculated both SMAPE and RMSE. We can see in the plot below that as we increase the training data linear regression is performing better and this nature is same for all the test sizes. **Providing more training data will improve the performance of linear regression**. Another point to note is increasing the test size, evaluation metrics are improving.

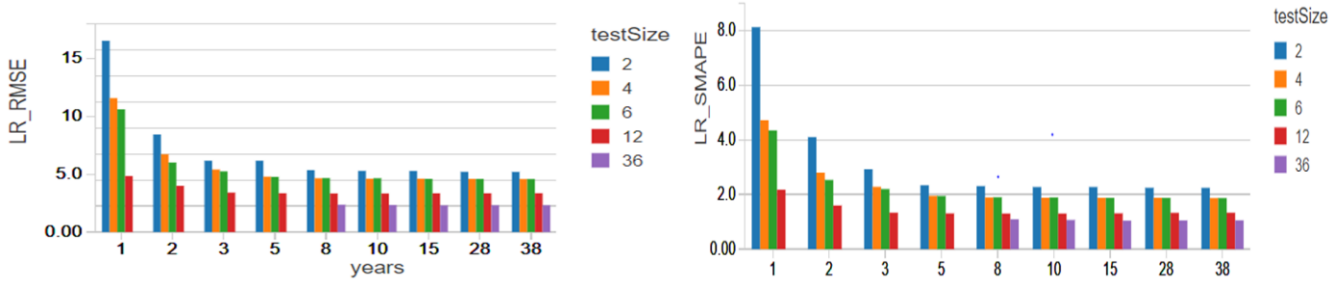


Figure 10: 1-day Prediction Linear Regression

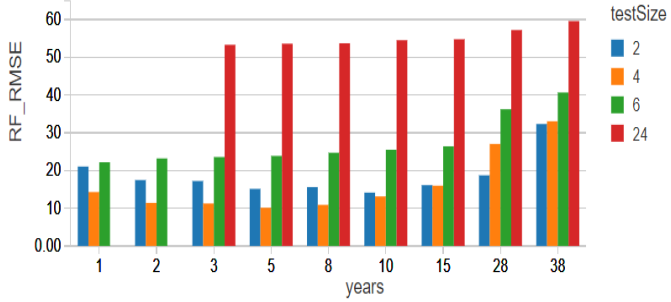


Figure 11: 1-day Prediction – Random Forests Results

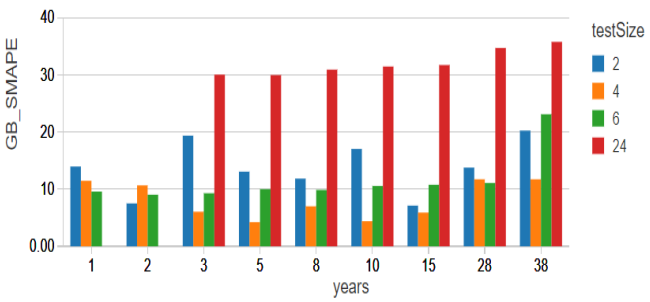


Figure 12:1-day Prediction -Gradient Booosting Results

For random forests there is a trade-off and we can determine the optimum amount of train data which gives best results. For Gradient boosting there is no clear pattern, but we can say that increasing the training data is not improving the performance. I will explain the reasons for this behaviour in the results and conclusion section.

Model Optimization

We developed our own user defined functions for performing fixed window rolling forecast cross validation with 10 folds and grid search hyperparameter tuning. The function "crossval" takes any spark data frame as input and splits it to 10 equal parts using the rank function on the data column. Now, I merge using union function 1^{st},2^{nd},3^{rd} splits as train and 4^{th} split as test and send them as arguments to "tuning_lr" function which would perform grid search on the list of parameters (another argument) and stores the results. Next, I merge 2^{nd},3^{rd},4^{th} as training and 5^{th} split as test and repeat this until the 10^{th} split becomes the test by each time moving forward the window by 1 split. I take all results and find out the average of results on test data.

Model	Hyper Parameter	Hyper Parameter
Linear Regression	'C' value : 0.0001	Regressor: L2
Random Forests	numTress : 20	maxDepth : 5
Gradient boosting	maxIter : 20	maxDepth : 5

These are the tuned hyperparameters for each of the model for 1-day predictions. We only tuned for 1-day predictions due to limitations and 1-day predictions models were the best performing in comparison to other time horizons. In case of random forests and Gradient boosting regressors there are many more hyperparameters (spark) such as "mininstancespernode" and "subsamplingrate" but we only choose to work on two most important ones and it is assumed others would not have much effect due to less number of features (around 15).

10. Results and Conclusions

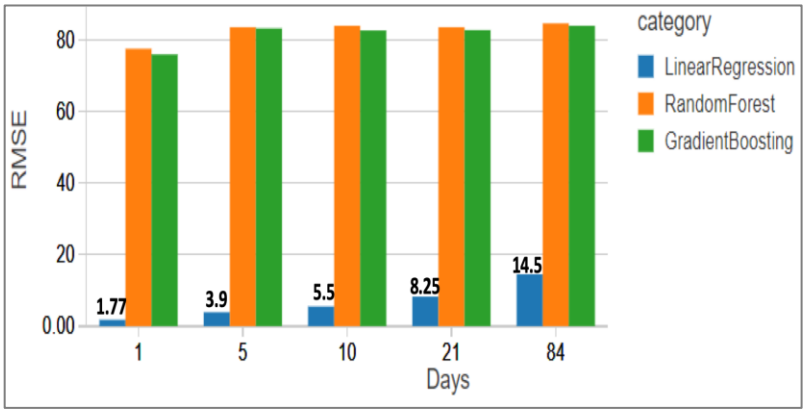
Our project compared three different machine learning models with a bench mark model for time series prediction of Apple Stock. The final SMAPE values for all models at different time horizons after identifying the **best features, train test split and performing hyperparameter tuning** (1-day prediction) are given in the table below.

The simpler model Linear regression performed better in comparison to Random forests, Gradient boost regressor and moving average. As expected, the machine learning has capability to explain the behaviour of time series.

Model	1day	1 week	2 weeks	1 month	4 months
Moving Average	1.98	4.63	6.06	11.09	15.43
Linear Regression	1.04	2.54	3.68	6.64	8.28
Random Forests	4.92	9.14	10.63	10.15	10.91
Gradient Boosting	4.22	8.76	9.94	10.43	10.98

Figure 13: Final SMAPE values for all models

One of the reasons for poor performance of random forests which are based on decision trees is **random forests do not work well with time series data which has increasing trend**. This is also illustrated in this blog (medium) very neatly. **After training the random forests on a training data when it is tested on validation data which it has never seen before the random forests can never predict values which are bigger than the training samples. But stock prices generally have increasing trend and the test data would have on average higher value than the train. The random forests simply put calculates good average of the new data from training data.** Another point to note is that all models are performing better in smaller time horizons and this is expected because as stock price after 4 months can be influenced more by external factors



When we take the whole 38 years dataset and train on 80% and test on 20% these are the results obtained. **We can the RMSE values for Random forests and GBT is very high comparison and this can be explained by the above reason. Providing to less data the model cannot generalized well and providing more training data the random forests can never predict upward trend, so this is the possible reason for random forests there is a trade off in train test split for obtaining good results (figure 11).**

We have only developed the classification model but not implemented the trading strategy. I believe for trading strategies classification methods give better returns because if we predict the stock price tomorrow to decrease to 9.8$ from 10$ and sold the stock today but if the actual stock price increases to 10.1$ we have lost money though, the SMAPE is good. constructing a model for predicting if tomorrow's price will go up or down should be a better strategy.

11. Recommendations

1. Do the hyper parameter tuning for more range of values and also for more parameters to improve the accuracy if random forests
2. Try to remove the trend in the series by some statistical transformations such as differencing and then apply random forests and gradient boosted trees
3. Our group has not explored advanced algorithms such as neural networks and this could potentially perform better than linear regression
4. Do multivariate analysis by adding external factors such as currency rate and sentiment

12. Mentors Recommendations

Our mentor Reza Farahani was very helpful and approachable. His recommendations were mainly in the side of feature engineering. Especially the features such as average of adjusted close of alternate days and average of prices with a window of week (Avg_Close_0_1_5_15), maximum price in last 5, 10 and 84 training days were his recommendations. It is evident that these features were proved to be important in longer time horizons. He also suggested us to not to focus much effort on classical models like ARIMA as machine learning models would give better performance. He answered a lot of our questions and guided us to good time series kernels in Kaggle. We referred to some of them but due to limited functionality in spark we didn't apply all of them.

13. Challenges:

We faced multiple limitations during the project and some of them are

1. Lack of native libraries for performing tasks such as rolling window cross validation
2. Lack of native libraries for statistical models such as ARIMA. There was a link provided to spark-ts library which had all the stats models that leverage spark are developed by a developer named Sandy Ryza (github) but the documentation was hard to infer and with our limited knowledge in Java and Scala we decided not to move a head.
3. The community account with data bricks has limited computational capacity and due to this the cross validation for model like random forests took hours to execute and many a times it has stopped running in between throwing a memory error ("Connect Exception error: This is often caused by an OOM error")

14. Contribution

My main contribution was in developing the code for performing train test splits, cross validation, hyper parameter tuning and running them for different scenarios. Group discussion, contributing to the analysis and presentation. Segregating all the teams work in the end.

15. Databricks

Although the community edition of data bricks has limited computational capacity it had many good features. I could see the power of spark and its parallel processing. For running a cross validation of

random forests, it made more than 10,000 spark jobs and I could also see the job numbers its details and DAG graph (I didn't understand it completely).

Bibliography

(n.d.). Retrieved from https://www.quandl.com/

(n.d.). Retrieved from https://investor.apple.com/investor-relations/faq/default.aspx

(n.d.). Retrieved from https://www.investopedia.com/ask/answers/06/adjustedclosingprice.asp

(n.d.). Retrieved from https://github.com/sryza/spark-timeseries

(n.d.). Retrieved from https://spark.apache.org/docs/2.1.0/api/python/pyspark.ml.html#pyspark.ml.classification.RandomForestRegressor

(n.d.). Retrieved from https://medium.com/datadriveninvestor/why-wont-time-series-data-and-random-forests-work-very-well-together-3c9f7b271631

Nesreen K. Ahmed, A. F.-S. (2010). An Empirical Comparison of Machine Learning Models for Time Series Forecasting. *Econometric Reviews*.

Spyros Makridakis, E. S. (2018). Statistical and Machine Learning forecasting methods: Concerns and ways forward.

Wei, W. W. (2013). *Time Series Analysis.* Retrieved from http://www.oxfordhandbooks.com/view/10.1093/oxfordhb/9780199934898.001.0001/oxfordhb-9780199934898-e-022?print=pdf