Machine Learning Engineer Nanodegree

Capstone Proposal

Quang Vu

Investment and Trading Capstone Project

Domain Background

Stock market price is both interesting and challenging to predict. Traditionally, it is analyzed with statistical model, then tested with series of back-testing, which is a process of applying the model with historical data to measure the performance of the model. This project aims to use a LSTM (Long Short Term Memory) model to automate that process, and measure how much information can be captured by the LSTM, comparing to statistical model.

The project's implementation will apply the method mention on this paper: Financial Market Time Series Prediction with RNN. And the statistical method used for benchmark is Autoregressive Integrated Moving Averate (ARIMA) model.

Problem Statement

The stock market has some properties on its data for each day: Open price, Close price, Adjusted price. In this project, I will use a RNN with LSTM to process a window interval (7 days, 30 days, etc. . .), and try to predict the stock price for the next day. A naive prediction would be taking an average of all the previous price in the window as a prediction for the next day's price:

```
price[t] = avarage(price[t-1] + price[t-2] + price[t-3]...)
```

An improve of that method is a ARIMA model, which takes into consideration a coefficient for each day price:

```
price[t] = a1*price[t-1] + a2*price(t-2] + a3*price[t-3] ...
```

We will see if an LSTM model can make a better prediction than a naive guess and an ARIMA model. Also, we will see which properties or which hyperparameter can be optimzed to help the performance of the model.

Datasets and Inputs

The New York Stock Exchange (NYSE) in 2010-2016 will be used for this project.
The dataset can be downloaded on Kaggle (https://www.kaggle.com/dgawlik/nyse/data).

The data is chosen because of the record of Open/Close/Low/High price for each day on the market during 2010-2016. In addition, there is also extra information about the securities price, splited-price, and the information about the company's earning, expense, profit... that can be served as extra parameters for the model.

Solution Statement

An LSTM model will look at a sequence of data in an interval (for example: 7 days), and try to capture the trend of the price and try to make the prediction for the next day. The accuracy of the prediction will be base on the actual price from the data for that day. The testing data will be split at the end of the actual data, to simulate predicting future price given historical prices. The training data will be taken from the data, with its validation data will be the price of next day.

Benchmark Model

The naive guess and the ARIMA model makes prediction base on previous price data, and compare them to the actual data obtain. Their metrics are obtain from how closely the predictions follow the actual price:

[Figure below]

Evaluation Metrics

While the metric of the benchmark model is well-defined, there are different metrics and less measurable metrics for using the LSTM model. The most important performance for evaluating the LSTM model, would be whether the model correctly capture the trend of the price. For example, if the model is able to predict the downward or upward trend of the price. Secondly, how much difference between the model's prediction and the actual result. The direction of the difference should also be taken into account. Thirdly, how lag the prediction is, ex: how many days after the price change, the model would be able to reflect/predict those changes.

Project Design

The project will be implemented follows these outline steps:

Exploring the data

- How large is the data?
- How many companies involve in the data?

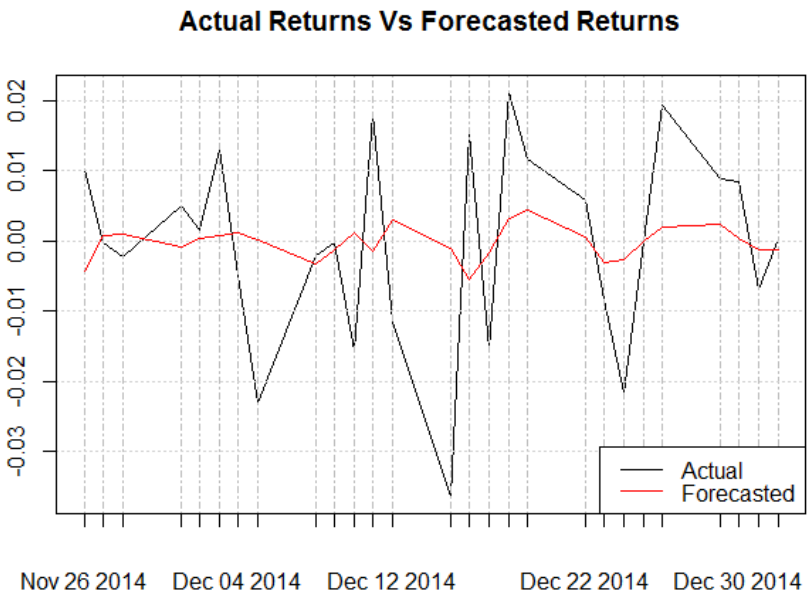

Figure 1: ARIMA visual result

	Actual_series	Forecasted	Accuracy
2014-11-26	0.0101377233	-0.0043090616	0
2014-11-27	-0.0003219135	0.0007807852	0
2014-11-28	-0.0023441737	0.0010150580	0
2014-12-01	0.0049120615	-0.0009070759	0
2014-12-02	0.0016375258	0.0003435073	1
2014-12-03	0.0131590684	0.0008073216	1
2014-12-04	-0.0049711114	0.0011228101	0
2014-12-05	-0.0230689605	0.0001129290	0
2014-12-08	-0.0020447544	-0.0032902765	1
2014-12-09	-0.0002495719	-0.0012435422	1
2014-12-10	-0.0156834851	0.0012498747	0
2014-12-11	0.0182084822	-0.0014586629	0
2014-12-12	-0.0115992224	0.0029173522	0
2014-12-15	-0.0365085594	-0.0010131041	1
2014-12-16	0.0154268090	-0.0055505608	0
2014-12-17	-0.0154268090	-0.0015995673	1
2014-12-18	0.0212679508	0.0031471618	1
2014-12-19	0.0116987764	0.0043570056	1
2014-12-22	0.0058169635	0.0005555327	1
2014-12-23	-0.0083976341	-0.0030760477	1
2014-12-24	-0.0216700763	-0.0026127616	1
2014-12-25	0.0000000000	-0.0001160376	0
2014-12-26	0.0193149604	0.0019859033	1
2014-12-29	0.0089410911	0.0024058764	1
2014-12-30	0.0084193215	0.0002595307	1
2014-12-31	-0.0069530750	-0.0013568468	1
2015-01-01	0.0003854973	-0.0013269785	0

[1] 55.55556

Figure 2: ARIMA Data

- Is price on each company in the same range, or vastly different?
- How frequently is the data collected?
- Is there any anomily in the data?

Prepare the data

- Load the data
- Can the data be processed on my machine? Can the data be processed on my GPU? Do I need to sample the data to make it smaller?
- Does the data require normalize/denormalize?
- Prepare the training and testing data
- Does the data need to be converted to particular shape?

Building the model

- LSTM will be used, and mainly rely on Keras implementation for converting the LSTM model to Tensorflow backend.
- Determine how many LSTM layer is needed
- Determine if Dropout is needed for LSTM each layer, and what rate is best for the model.

Train the model

- How many epoch is needed? And how many epoch my machine can run. Will more epoch always better?
- If the training takes very long time, determine how to capture the training and resume if needed, without restart from scratch.
- Is there a way to visualize the on-training process?
- Is there a method to make model run faster?
- Is there a parameter to improve the above metrics of the model?

Visualize the result

- Make a naive prediction as a baseline benchmark.
- How is the result from the LSTM model compare to the benchmark?

Conclusion

- How well is the performance of the model?
- Is there any extra info that could help the model?
- In which criteria or use case, does the model provide more benefit can other statistical model?