**Comparing the predictive power of different models**

Hyung-Ju Cha

Info- I 390

**Motivation**

**Goal of project**

The goal of this project is to compare the accuracy between time series and RNN models in predicting stock prices. I will use three models which are LSTM, Arima, and Sarima to predict stock prices. I will compare the MSE and MAE between the predicted stock prices of Samsung, Apple, and Amazon using data within 3 years, 1 year, and 1 month period.

**Methods**

* Data Collection and preprocessing

To collect stock price values, I used the the FinanceDataReader library on python in order to collect stock data. For Samsung, I have collected data after 2018 since there was a stock split on May 2018. I have collected the stock data by changing the stock code. For Samsung, the stock code was “005930” because it is the Samsung stock code on KOSDAQ and the stock code for Apple and Amazon are “AAPL” and “AMZN”.

In order to preprocess collected stock data, I used the MinMaxScaler. The MinMaxScaler is a method to scale the data into values between 0 to 1. This method was used to reduce the calculation time of the deep learning model since smaller values require less computing time. The equation to scale the value is the equation below.

(X- Minimum value of X)/ (Maximum value of X – Minimum value of X)

In order to prevent the error of dividing the value into 0, I added a very small value (1e-7) to the denominator.

After that, I preprocessed the data so that I could use the OHLV(Open, High, Low, Volume) data for the past 10 days to get the Closing price for the next day. In order to do this, I have assigned the the 10 days worth of OHLV data into \_x and assigned \_y to the Closing price for next day. For example, if we suppose that the KOSPI or NASDAQ opens on the weeknds, we could predict the closing price of June 11th from OHLV data from June 1st to June 10th.

In order to split the preprocessed data set into train and test data, I assigned the data set into a 7:3 ratio using the np.array function.

After that, I have imported Keras from Tensorflow in order to develop a prediction model. The input\_shape is the length of the data that is inputted into the LSTM Model. The length of the data decides how the LSTM is going to be generated. Since we have 10 days worth of data, the length of the data should be 10. The input\_shape dimension is 4 because we are inputting four data points ten times. This means that the LSTM model will get 4 dimensional numbers 10 times. In order to make the LSTM model more complicated, I first used the return\_sequences=True on the first LSTM in order to run one more LSTM. I have also inputted the Droupout after adding the LSTM model in order to enhance learning accuracy. The Droupout value could be any value between 0 and 1 and the value I have used was 0.1. After that, I have compiled the model by first setting the optimizer as ‘adam’ and the loss and mean squared error. I have ran 70 epochs and the batch size was 30.

To run the Arima model, I first split the data into train and test data into 7:3 ratio using Sklearn. Then I used the tsa plot to find the Autocorrelation and partial Autocorrelation of data. After checking the correlation of the data, I differenced the raw data in order to make the data stationary. This is because the arima prediction model could be used on stationary data. After that, I have inputted the differenced data into the auto arima model. The auto arima model is a model that can automatically find the ideal p,d,q value for the arima model. In an Arima model, p is the number of autoregressive terms required for stationarity, d is the number of nonseasonal differences required, and In the prediction equation, q is the number of lagged forecast errors.In the arima model, I have also set the the month interval to 12 months and seasonality to true. After that, I have made a prediction using the auto arima model and visualized the predicted value and actual value.

For the Sarima model, I made an algorithm that could find the optimal p,d,q value where the AIC is the lowest. The Akaike information criterion (AIC) is a methematical method for evaluating how well the model fits the raw data. AIC is used to compare different possible models and determine which one is the fits the data the best. I have set the range of p,d,q values and have also added seasonal pdq values which are the pdq values and one value for seasonality terms.

* Models

There are three different methods used to predict stock prices. The first method I used is the Recurrent Neural Network(RNN) which is a type of deep learning method. The Arima method is a method that integrates the Autoregression function and the Moving Average function. It is a model that detects a series of different standard structures in a time series data. Autoregression is a model that uses the dependent relationship between an observation and several numbers of lagged observations. The moving average is a model that uses the the dependency between an observation and a residual error from a moving average model applied to lagged observations. These two models are integrated differencing raw observations to make the time series stationary. The Sarima (Seasonal Autoregressive integrated Moving Average) is a method is a method that adds seasonality to the Arima model by adding an additional parameter for the period of seasonality.

* Prediction models

When a person is having a conversation, he interprets words while understanding the previous context. For example, in the bold text on the left, you read "People~" followed by "Conversation~" and then "When I do~" started reading "When I do~" sequentially and read "~ Interpret" while understanding the overall context from the front to the back. In other words, you read the sentence sequentially from left to right, understanding the previous context. In other words, deep learning could be used to put the concept of time so that data could be input sequentially to understand the overall flow of data.   
  
RNNs are sequential data with time concepts. Thus it is appropriate for dealing with time series data. For example, for Stock Price Data, we could enter stock price data on June 1st, stock price data on June 2nd, and stock price data on June 3rd as input sequentially and ask RNN the stock price on June 4th. RNN is the previous context. Here, the term "stock price flow" would be more appropriate than context, and RNN is predicting what the next stock price will be like from the previous stock price flow.

In circular neural networks, neurons are called cells, and recurrent in circular neural networks means that the state information of cells learned from previous data, as shown in the figure below, is reused when learning using the next data. Therefore, the cyclic neural network is suitable for processing time series data and can be spread out as shown below.

텍스트, 시계, 표지판이(가) 표시된 사진

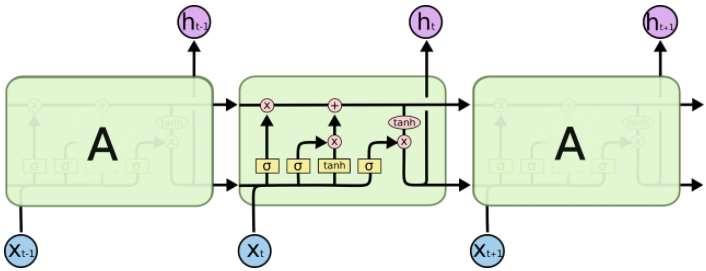
자동 생성된 설명

The formula for the RNN algorithm is the function below

ℎ𝑡=Activation function (𝑊(ℎ𝑡−1,𝑥𝑡)+𝑏)  
ℎ1=Activation function (𝑊(ℎ0,𝑥1)+𝑏)

The activation function on the formula above could be relu or tanh.

The output from the cell, 𝑡𝑡, is called a hidden state. W and b are weights and biases that converge to optimal values when the model is trained, as is the equation for linear regression (y=W𝑥+ b)).  
  
In the figure above, x0 represents the first input, x1 represents the second input, x2 represents the third input, and then it continues to be inputted and then one day the xt which is the t-th input enters the input.  
  
ht is the output of the RNN at the t-th input. The way we calculate ht is the same as the formula above, and ht is not only getting input at point xt, but also getting input at point ht-1, which is why we can understand the past context using ht. It accumulates past values and continues to calculate them.   
  
 For it's stock price data, x is stock price data. Let's say, let's make a stock price predictor that predicts the next day's stock price data from the three days' stock price data. For example, you can enter the stock price data for June 1, enter the stock price data for June 2, enter the stock price data for June 2, and enter the stock price data for June 3 for 22, and ask RNN how much is the stock price data for June 4th.



The LSTM is a more complementary version to improve the performance of the previously learned RNN. LSTM is generally used rather than RNN because it usually performs better than RNNs. Although LSTM also has a more complex formula internally, its use is no different from RNN.

**Compare performance (Explain MSE and MAE)**

The mean squared error (MSE) is a measure of how close a fitted line is to the data points. It measures the distance between the actual data points and predicted line by finding the average distance from each point to the predicted line. The MSE formula is presented below.

MSE = (1/n) \* Σ(actual – forecast)2  
Where:

* n = number of items,
* Actual = original or observed y-value,
* Forecast = y-value from regression.

The mean absolute error (MAE) is a statistic that measures the difference in errors between paired observations describing the same phenomena. Comparisons of predicted against observed, subsequent time versus starting time, and one measuring technique versus another measurement technique are examples of Y versus X. The MAE is calculated by the following formula

텍스트이(가) 표시된 사진

자동 생성된 설명

**Results**

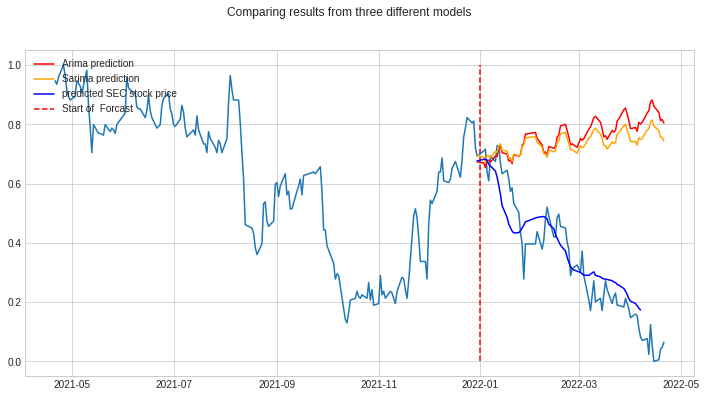
In order to compare the accuracy between different models, I have calculated the MSE and MAE between predicted stock values and actual stock values. I first split the data into train and test data with a ratio of 7:3. After that, I have calculated the predicted value using the RNN, Arima, and Sarima model. After that I calculated the MSE and MAE between actual data and predicted data. The MAE between predicted stock price using RNN and actual stock price was 0.0234 and the MSE was 0.0009706. The MAE between predicted stock price using arima model was 0.295 and MSE was 0.1117 and the MAE between predicted stock price using Sarima model was 0.425 and MSE was 0.226. From this result, we could say that the RNN method is a lot more accurate in predicting Samsung stock prices from 2019 to 2022 since the scaled MSE is about 27% lower than arima and 40% lower than sarima. The MAE is about 10% lower than the arima model and more than 20% lower than the sarima model. The arima model was the second most accurate model out of the three models as its MSE is 13% lower and MAE is 11% lower than the sarima model.

I have also used stock data of Samsung from 2021 to 2022 in order to test the accuracy of different models in a smaller window. The MAE for the RNN model was 0.0724 and the MSE was 0.0077. The MAE for the arima model was 0.419 and MSE was 0.238 and the MAE for sarima model was 0.393 and MSE was 0.207 which was respectively about 2 percent lower than the arima model which shows that the Sarima model is more accurate.

In addition, stock data from Apple and Amazon was also used to test the accuracy of three models. For Apple stock data in one year period, the MAE was 0.178 and MSE was 0.041. The MAE and MSE for arima model was 0.278 and 0.113 respectively and the MAE and MSE were 0.117 and 0.019 for the sarima model. When I increased the window size to three years, the MAE for RNN was 0.06 and MSE was 0.005. The MAE for Arima model was 0.14 and MSE was 0.025. The MAE for Sarima model was 0.115 and MSE 0.017. The RNN was the most accurate model as its MAE and RNN was the lowest in both window sizes and the which shows that the RNN is the most accurate model. The Sarima model was the second most accurate model since its MAE and MSE are about 10 to 15% lower than Arima model.

For Amazon data in three year period, the MAE for RNN model was 0.06 and MSE was 0.005. The MAE for Arima model was 0.139 and MSE was 0.034 and the MAE for Sarima model was 0.141 and MSE was 0.036.

Moreover, the accuracy differs more when daily stock prices were predicted. For Samsung, the predicted stock price for April 22nd was 66000 . For apple it was $155.264 which is 8 dollars lower than the actual price whereas



The graph above is the comparison between the three models based on Samsung stock prices during 2021 and 2022. The red line is represents the Arima prediction and yellow line represents the Sarima prediction and the blue line represents the prediction based of RNN. The red vertical line shows the point where the prediction starts. As you can see, the RNN based prediction is almost identical to the actual stock data whereas the Arima and Sarima data is quite off from the actual data.

**Discussion**

Looking at the results, we can say that the LSTM or RNN are the most accurate model to predict stock prices since prediction based on LSTM model was the most accurate in all different settings. The LSTM prediction was about 10 to 20 percent accurate than Arima Sarima model even though different window size and three different stock data were used. This is because RNN is suitable for time series data because the cell that trained the previous data is repeatedly used when calculating the next data. Moreover, the Arima model predicts the future value using the autoregressive and moving average function, however, since the prediction is made on a whole period instead of repeating small sections within a period, the prediction can only show the general trend of future stock prices and cannot predict daily stock prices. Moreover, Sarima model seemed to be more accurate than Arima model for predicting Samsung stock prices but was less accurate when predicting Amazon prices. This is because Samsung stock prices have more seasonality since it regularly releases new products unlike Amazon.

**Conclusion**

In conclusion, the RNN model seemed to be the most accurate model to predict daily or monthly stock prices because it repeats learning based on small chunks of the data whereas the Arima or Sarima model uses the whole data as one instance to develop a learning model. Since it is important to predict daily stock prices when investing stocks, RNN or LSTM is a more accurate method to predict stock prices since it is more suitable to predict the daily stock prices in detail rather than getting the overall monthly or seasonal trend. This kind of analysis is not conclusive since it is not systematic. In order to improve the credibility of this statement, we need to test on more different observation windows or add more parameters.

**References**

<https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/#:~:text=ARIMA%20is%20an%20acronym%20that,structures%20in%20time%20series%20data>.

<https://www.statisticshowto.com/probability-and-statistics/statistics-definitions/mean-squared-error/>