Time Series Forecasting using Machine Learning

Anav Dutt

IPHS Senior Research Seminar, Fall 2021, Prof Elkins and Prof Chun

Abstract

Predicting stock prices has always been frowned upon, however as we move into a more technology driven world, it is important to note the potential impact of time series forecasts. This is a theoretical study to see how machine learning interacts with time series data. This project analyses 4 different machine learning models, being XGBoost, ARIMA, Long-Short Term Memory, and Facebook Prophet. My purpose is to look at how these modelling programs would interact with different assetclassesand which ones would be the best predictors for each. I wanted to look at different asset classes so that when analyzing these models I would be able to take into account different levels of volatility, seasonality, fluctuations, and trends. The purpose behind this project is to see how time series forecasting is important, and how machine learning models perform in high pressure situations. Time series is often used by traders, investment bankers, and many others in the financial world, and even a slight change in the data ticks could benefit them greatly. If machine learning models are able to accurately predict and find trends in time series data, then a lot of these high paying jobs will come under scrutiny. Producing accurate and reliable time series models is extremely tough and therefore time series forecasting is something I wanted to explore to see how if in theory we were to implement these models, how they would interact with real world, high scrutiny variables. It is also extremely interesting to see how these models work internally, what they account for, and how they differ from each other. Compared to standard regression models, these are much more advanced, and provide a better approach to understanding data. It is however, important to look at these models in the context for which they were designed. Each has its own parameters and nuances which leads to requiring extreme user expertise when working on tuning these models and understanding the portrayed results. This project involved studying every model in detail to understand how the model worked under the hood, be it the math, tuning parameters, conversions, dealing with missing data, and different variable types, what the models were used for, and what the optimal usage looked like. The next step was collecting all the datasets required, converting them to daily data and checking for missing data. The notebook I used required tweaks for all the models to accommodate different variable types, as well as adjustment for the model parameters to adjust for daily data. The notebook was run 3 separate times for the different asset classes, and the analysis was modeled based on my understanding on the inner working of these models and on the expectations of their performance along with analyzing any reasons for potential anomalies and working through them. The mean squared error, model diagnostics, and the R squared values also added to model understanding and fit.

Models

- XGBoost– This model is the most similar to our classic regression models. The full form of this being extreme gradient boosting. It is a decision tree based ML technique used for regression using the sum of M weak learners to learn F(X) (boosting). This model focuses on computational speed and model performance. An issue with the XGBoost model is that if the parameters are not tuned properly, and it is often cumbersome to tune because of the large number of hyper parameters. XGBoost is integrated with scikit learn for python users and requires time series datasets to be transformed into supervised learning.
- ARIMA (Autoregressive integrated moving average) – This model is a more advanced time series forecaster than XGBoost. The ARIMA model contains multiple building blocks of univariate time series. Most regression packages do not make special allowances for time series data, and in this case, the ARIMA model is built to deal with time series data. This model is an ensemble combining the Autoregressive model with the Moving average model in order to improve the model performance. So by using these two models, part of the future variable can be explained by past values (using the autoregressive model), and the other part can be explained by past errors (using the moving average model)
- LSTM(Long short term memory) – Recurrent Neural Networks (RNNs) suffer from short term memory and may leave out important information. LSTM'swork to fix that issue. They have gates that regulate the flow of information and analyze what data to keep and what to let go. This Gated RNN architecture is designed to mitigate the vanishing and exploding gradient problem. Each LSTM maintains a cell state vector, which is a value or data that the LSTM has edited or kept, and at each time step, the next LSTM can choose to read from it, write over it, or reset it. There are three gates in the LSTM model: 1) The input gate, 2) The forget gate, and 3) The output gate. As the data travels through the LSTM, there is also a tanh function that distributes gradients, assigning weights to values, and helps prevent vanishing or exploding.
- Facebook Prophet – Facebook Prophet is a decomposable time series model. It accommodates seasonality over multiple periods and allows the analyst to make assumptions about trends. The Facebook Prophet is more hands on and has a human in the loop system, where there are a lot of interpretable parameters that an analyst can adjust based on their assumptions of the forecast. The Prophet does not need to remove outliers unlike ARIMA. Prophet was designed for analysts who do not have much of a background in ML, and uses intuitive parameters that can be adjusted without needing to know the inner working of the model.

Visualizations

Fig 1a. – Bitcoin Forecast

```
df_valid[["adj_close", "Forecast_ARIMAX", "Forecast_Prophet", "Forecast_XGBoost"]].plot(figsize=(14,7))
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f79ea77ef90>

Fig 1b. – Bitcoin LSTM forecast

```
plt.figure(figsize=(16,7))
plt.plot(y_test_inv.flatten(), marker='.', label="Actual")
plt.plot(test_predict_inv.flatten(), 'r', marker='.', label="Predicted")
plt.legend()
```

<matplotlib.legend.Legend at 0x7efdd1d0f110>

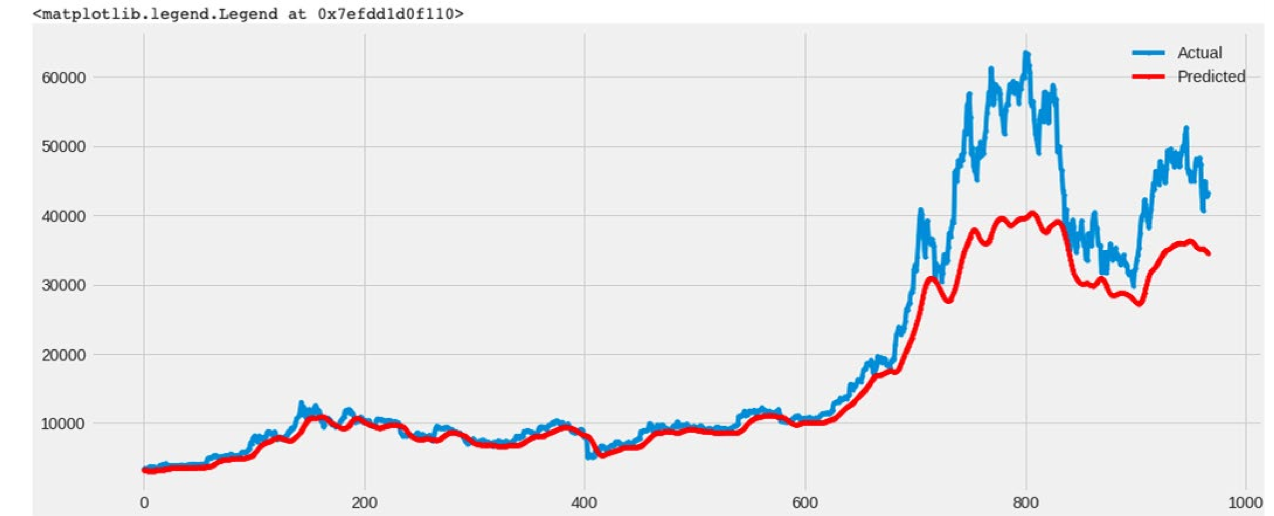


Fig 2a – S&P500 Forecast

```
df_valid[["open", "Forecast_ARIMAX", "Forecast_Prophet", "Forecast_XGBoost"]].plot(figsize=(14,7))
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f504f647d10>

Fig 2b. S&P500 LSTM Forecast

```
plt.figure(figsize=(16,7))
plt.plot(y_test_inv.flatten(), marker='.', label="Actual")
plt.plot(test_predict_inv.flatten(), 'r', marker='.', label="Predicted")
plt.legend()
```

<matplotlib.legend.Legend at 0x7f4fe5142490>

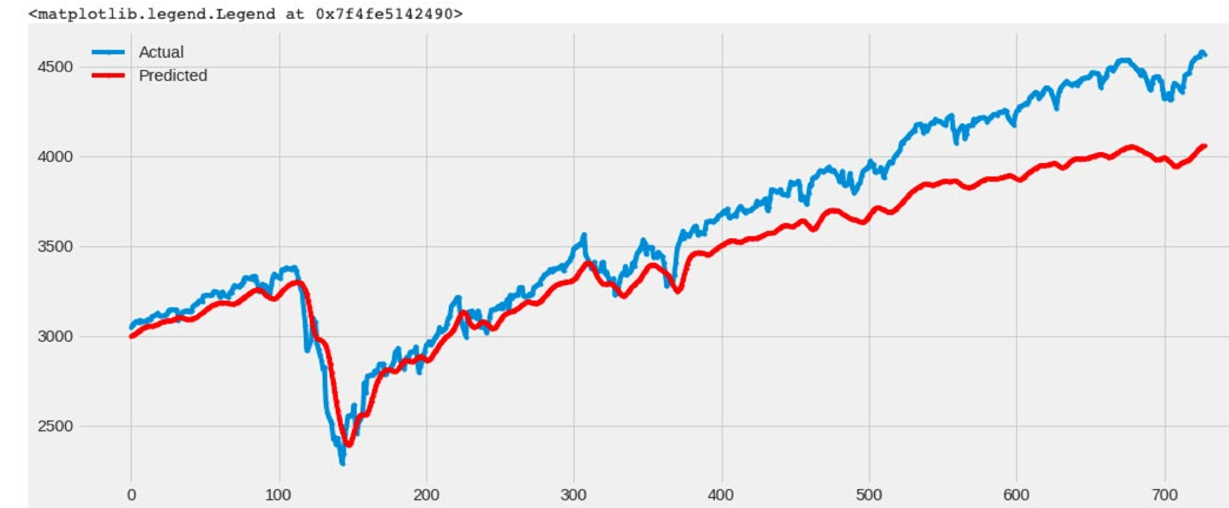


Fig 3a – 10 Year US Treasury Forecast

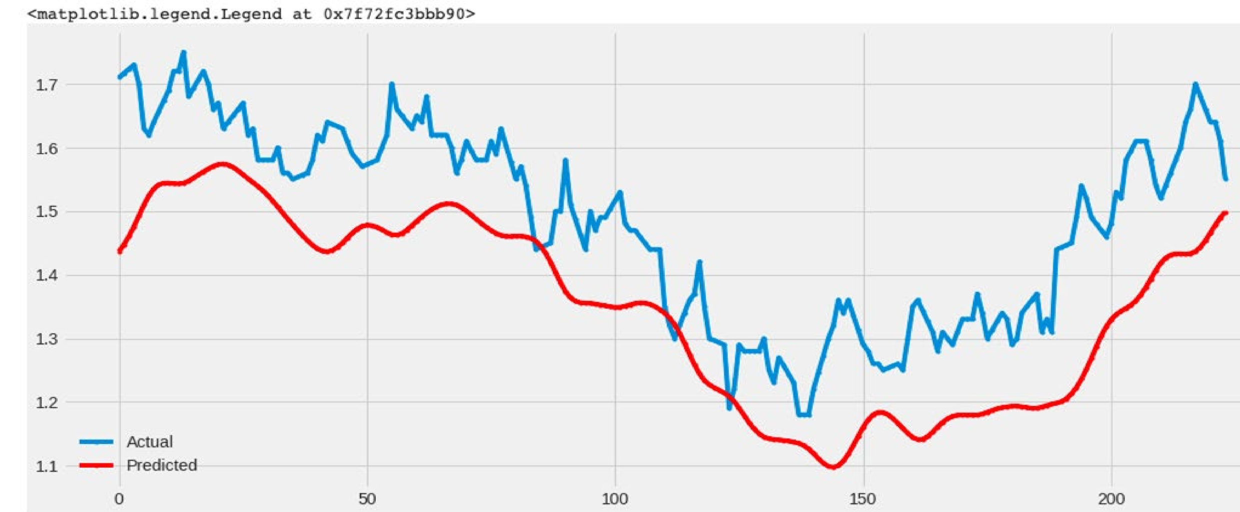
```
df_valid[["open", "Forecast_ARIMAX", "Forecast_Prophet", "Forecast_XGBoost"]].plot(figsize=(14,7))
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f9ac0e30e50>

Fig 3b. – 10 Year US Treasury LSTM Forecast

```
plt.figure(figsize=(16,7))
plt.plot(y_test_inv.flatten(), marker='.', label="Actual")
plt.plot(test_predict_inv.flatten(), 'r', marker='.', label="Predicted")
plt.legend()
```

<matplotlib.legend.Legend at 0x7f72fc3bbb90>



Results

The results for these graphs vary in degree as a result of the asset classes chosen. The asset classes chosen here represent a variance in time series with regard to volatility. The first dataset analyzed was Bitcoin. In our world today, with the growing popularity of cryptocurrency, it was important to look at something with high volatility and popularity, to see just how accurate the models would react to such a volatile class. The second asset class is the S&P500. The S&P500 historically has been known to be pretty stable, and would provide a good analysis to how the models perform with stable and near constant data. The last asset class was 10 year US treasury bonds. It was also important to look at something that did not show a constant trend, but also was not extremely volatile. The treasury bonds also look at an asset class that is pretty removed from the likes of Bitcoin and the S&P500.

- XGBoost (green) - As visible in Fig 1a. And Fig 3a. The XGBoost model appears to perform pretty poorly. The issue with this model as can be seen in these figures because both of these assets appear to have volatile time series, which is why this model fails to capture the spike in values. The flat line in the prediction occurs because of the regression is unable to predict as the data is not identically and independently distributed, and in this case, cannot predict the regression coefficients. Therefore, it reverts back to the mean from the data that it was trained on, called auto sample forecast, which is why we see the flat line. In the S&P500 forecast however, XGBoost was able to capture the trend and predict the general outline to a good level, but not as accurately as possible. This could be related to the fact that the beta value that the model had captured was not large enough to keep up with the test data.
- Arima (red) – In Fig 1a. The ARIMA is unable to capture the high volatility in the Bitcoin dataset. A large change in the values results in the model using the past values and error terms of the training set to predict the future values and was unable to capture this high level of volatility. However, in the dataset for the treasury bonds (Fig3a.), the ARIMA does a great job predicting the values, this is because even though there is slight volatility, the values only fluctuate between a very small distance. This shows that the ARIMA is ideal for datasets with low fluctuations, and it is able to sniff out general trends. This idea is reinforced in Fig 2a. Where for the most part, the ARIMA does a good job understanding the trends and volatility, the only reason it isn't able to capture the entire spread is because there is a large change in values and the model was not able to capture the accelerated increase.
- Facebook Prophet – As seen in all the figures, the Facebook prophet model appears to outperform all the other models. There are multiple reasons for this. The main reason being that the prophet makes its prediction using a rolling window, where it uses its last predictions and uses the parameters that have been adjusted to react to the train and test dataset. This rolling window is the main reason it outperforms the other models. There is a large component of human in the loop for this system, which means that the human is able to induct domain knowledge into the problem. The human is able to input priors on seasonality and how often to expect change points in the data. The reason it is able to predict the Bitcoin variable so well compared to the others is a result of the model being able to remove the iid noise, so the data is easier to interpret for the model.
- LSTM– this model is the closest competitor to the Facebook prophet. Fig 1a 2a. 3a. all depict the LSTM's predictive prowess. Although it isn't able to map the Bitcoin data as well as the prophet, it does a great job following the trends, an that is credited to the gated system of the LSTM The graphs for the LSTM is smoother compared to the other models, however, this is not an issue. As we seen in the S&P and treasury data, it is evident that it is able to model the trends, adjust to the volatility and not fall prey to the diminishing gradient problem as the gated channels are effective in keeping the valuable data and assigning weights that are relevant to the dataset. This model was trained over 50 epochs, which means it goes through the training dataset 50 times to accurately assign these weights.

Takeaways

The results are pretty clear cut in displaying how the models perform. The regression models are not able to keep up with the predictive power of the ML models. This is not to say that the regressive models are not useful. They are incorporated into the ML models and improved to make more accurate predictions. As mentioned earlier, these are three different asset classes with different levels of volatilities and fluctuations that help analyze how these different models interact with these distinct asset classes The Prophet appears to outperform most of the models, however, in the S&P model, the LSTM forecast is extremely close if not better at predicting the values. The takeaway from this is the data that possesses a steady trend, even if it's actual beta value fluctuates over time, the LSTM is a better predictor for that type of data. The Facebook Prophet on the other hand is very good at forecasting large variations and fluctuations because of the rolling window, which is why it does well for the treasury and Bitcoin but doesn't perform as well as for the S&P. While XGBoost is an extremely powerful computational tool, I don't believe it was able to capture the beta values accurately in order to accurately forecast trends and reverted to the auto sample training mean. The ARIMA model is a powerful ensemble model that according to the results appears to be extremely efficient at predicting low value data sets, unable to accurately forecast extremely volatile data, and does well with regard to trend prediction when it comes to a steady dataset. It is important to consider that these models were all designed for different purposes, but this project analyzes their interaction with time series data of different categories in order to see which one works best with which kind of data. The key takeaway is that even though the prediction models are not strong enough to compete in the financial markets at this moment in time, their evolution is continuous and as they become more powerful, they might be able to forecast even the smallest changes in data minutes before it happens. However a lot of financial analysis is confidential and an advantage such as that may bring about absolute returns for those who have access to it.

Conclusion

As visible from the results, time series forecasting, while not perfected, is getting closer and closer to accurate models. Especially when analyzing Facebook Prophet and the LSTM model, we can see the potential future use. This project makes it clear that in the future ML models have the potential to become so powerful that they will be able to forecast ticks by the second. This is an essential for any financier who wishes to turn a profit, which is why companies spend billions of dollars on terminals that give them a slight trading advantage. The reason these models have not taken over the roles of analysts at this moment in time is because they are not as accurate and reliable as they should be and money is a valuable asset that needs to be invested with certainty. There are also certain factors that a model may not be able to see for example the models may not be able to understand financial history and how events in the past shape the behaviors of people today. People are emotional beings, and the market is often related to human behavior, sometimes humans can be irrational and therefore affect the market in a way models may not be able to see. However the hope for ML models in the future is to use it not just for profit with regard to predictions in the market, but also predictability for the government and those who can take measures to prevent slumps in the economy, see booms or busts coming and change policies accordingly, and use these models so that they may better allocate resources. Time series modeling is a vast subfield and has varying uses, this project captures one of them and shows how effective and powerful these models are. Future work with regard to this project and time series forecasting would be to look at more datasets with more observations, and see how newer models compare to the existing ones and what parameters could be improved or added to make a more intuitive and accurate model. As more models spring up, the competition to gain access to these models will also increase, however, it would be interesting to see if there was a model that was accurate and reliable and available to the public, and what the effect of that would be on the market. Using these 4 models was ideal because it looks at an entire spectrum of time series forecasting tools, and using these 3 asset classes allows us to see how these different tools react in different environments. Moving forward, people can engineer their models based on results that they can see from this report and decide which models parameters best fit their needs.

Citations

- Jon Chun, Kenyon College, chunj@kenyon.edu
- Katherine Elkins, Kenyon College, elkins1@kenyon.edu
- https://www.kaggle.com/akashmathur2212/bitcoin-price-prediction-arima-xgboost-lstm-fbprop/notebook#LSTM
- https://link-springer-com.libproxy.kenyon.edu/chapter/10.1007/978-1-4842-7150-6_7
- https://learning.oreilly.com/library/view/machine-learning-and/9781492073048/ch05.html
- https://peerj.com/preprints/3190/
- Ritvik Math – youtube
- https://blog.ml-quant.com/p/history-of-machine-learning-in-finance
- https://xgboost.readthedocs.io/en/stable/
- https://github.com/tatsath/fin-ml
- https://github.com/Apress/advanced-forecasting-python
- https://www.google.com/url?q=https://learning.oreilly.com/library/view/advanced-forecasting-with/9781484271506/&sa=D&source=docs&ust=1635435980556000&usg=AOvVaw1CQh28ztOtu1eKsv_WWOpJ
- https://www.youtube.com/watch?v=_ZQ-lQrK9Rg&t=602s
- https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2701346
- Marketwatch.com
- Yahoo Finance
- http://colah.github.io/posts/2015-08-Understanding-LSTMs/
- https://www.netsuite.com/portal/resource/articles/financial-management/financial-forecast-machine-learning.shtml