Multiple Time Series modeling using Apache Spark and Facebook Prophet

Yogesh Awdhut Gadade
(CWID: 10467214)

05-06-2022

—

BIA 678 Big Data Technologies

—

Professor David Belanger

Introduction

Time series forecasting helps organization in decision making by analysing future predictions based on historical time series data. For example, in case crypto currency daily closing prices if we want to make predictions for the daily close values of bitcoins, we can train a time series model dedicated to learning fluctuations in bitcoins closing prices for past x years and then make predictions for the future closing prices. Now just imagine if I want to make the same predictions for the other crypto currencies too then in that case, I will be ending up creating n separate models (where n is number of crypto currencies data we have) for each crypto currency and the sequential training of each model will take a lot of time computationally.

Similarly, we can consider other examples too like forecasting monthly CPI values for 26 different countries, Products Sales predictions for the chain of stores (it can be thousands of products so those many models), etc. Also, as a data size increases with time the sequential processing becomes time consuming computationally.

Therefore, instead of going sequential why not use big data technologies to make training of models in parallel. In this project I am proposing a different approach of Time Series Forecasting Models development at the same time using parallel processing and computing power of big data technologies. In general, we can use different models in time series forecasting to learn and predict future forecasting. In this project I am using Facebook Prophet. It does not require much prior knowledge of forecasting time series data as it can automatically find seasonal trends with a set of data and offers easy to understand parameters.

The goal of the project is to focus on exploring the ways to develop and train multiple time-series models at the same time in parallel that can be used in case if we want to train thousands of time series models in parallel. It's challenging to train in this way and develop multiple time series models with ever increasing volume of dataset. The purpose of exploring this approach is to employ Spark to do the above task in less amount of time in parallel instead of normal sequential way of training the individual time series models.

Training of Multiple Time Series Models

About Data

Before jumping to training of models let’s look at datasets used for the exploration of sequential and parallel processing of multiple time series models. We have with use following data:

1. Daily closing price of top 4 Crypto Currencies data from 2013-04-28 to 2022-04-24
    a. Shape of the dataset: (10015, 10)
    b. memory usage: 7.8 MB
    c. ‘bitcoin', 'ethereum', 'cardano', 'tether'

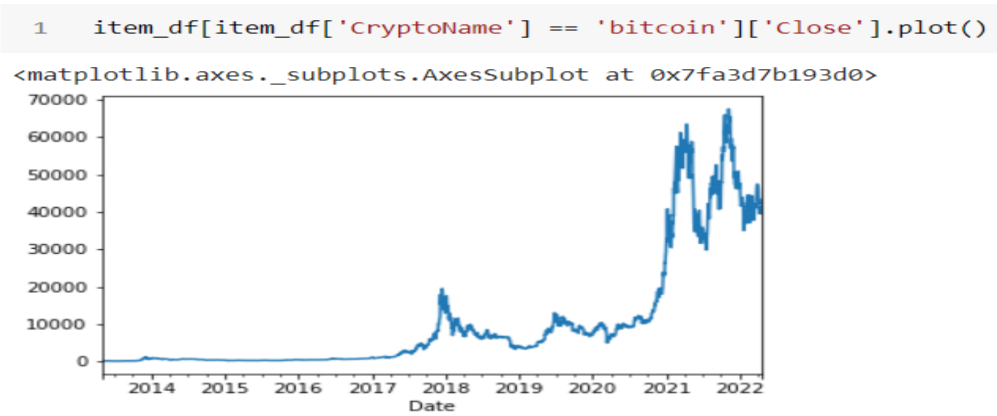

Figure 1 Crypto Currency (bit coin)

2. Monthly CPI values for the 4 countries from 1915-01-01 to 2022-01-01
    a. Shape of the dataset: (2704, 4)
    b. Memory used: 1.9+ MB

```
1    item_df[item_df['LOCATION'] == 'FIN']['Close'].plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc4153f71d0>
```

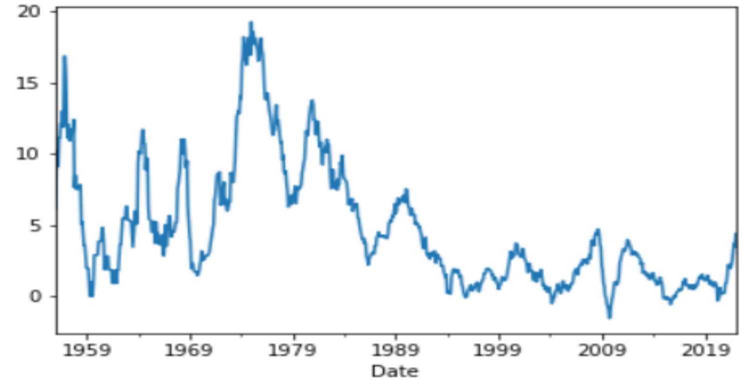

Figure 2 Example CPI data for Finland

Time Series Data Preparation-Trend, Seasonality, Stationarity

For most of the time series models it's assumed that the data should be stationary. Therefore, under preprocessing we will perform the test to check time series data is stationary or not. There are different tests available to do so I am using Time Series plot of data and one more test Augmented Dickey–Fuller test (ADF) tests as follows:

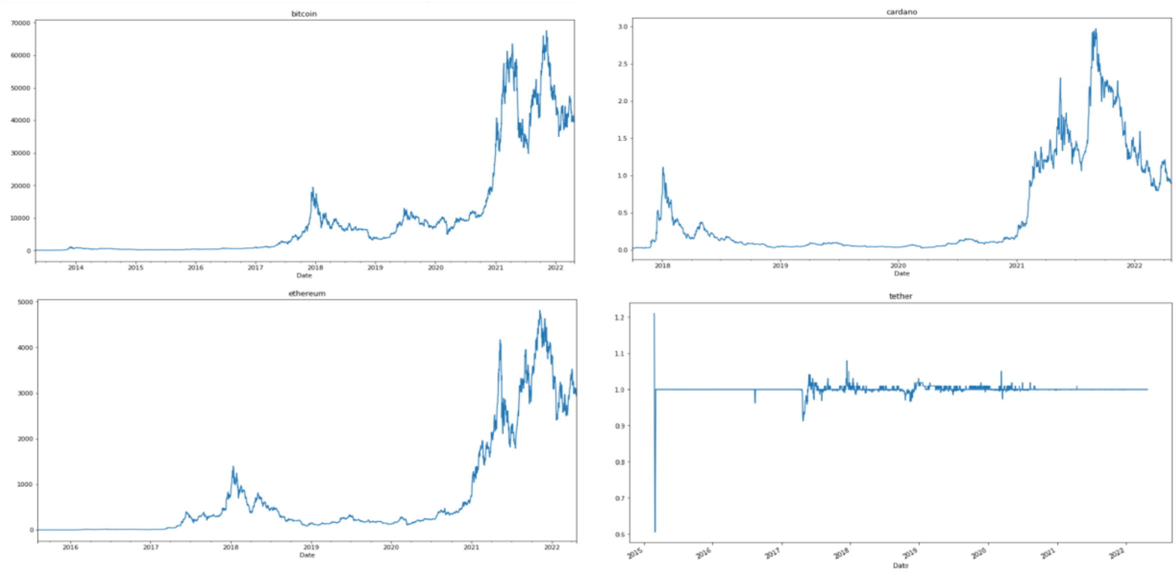

Figure 3 Crypto Currency Data with No or very little Trend or Seasonality

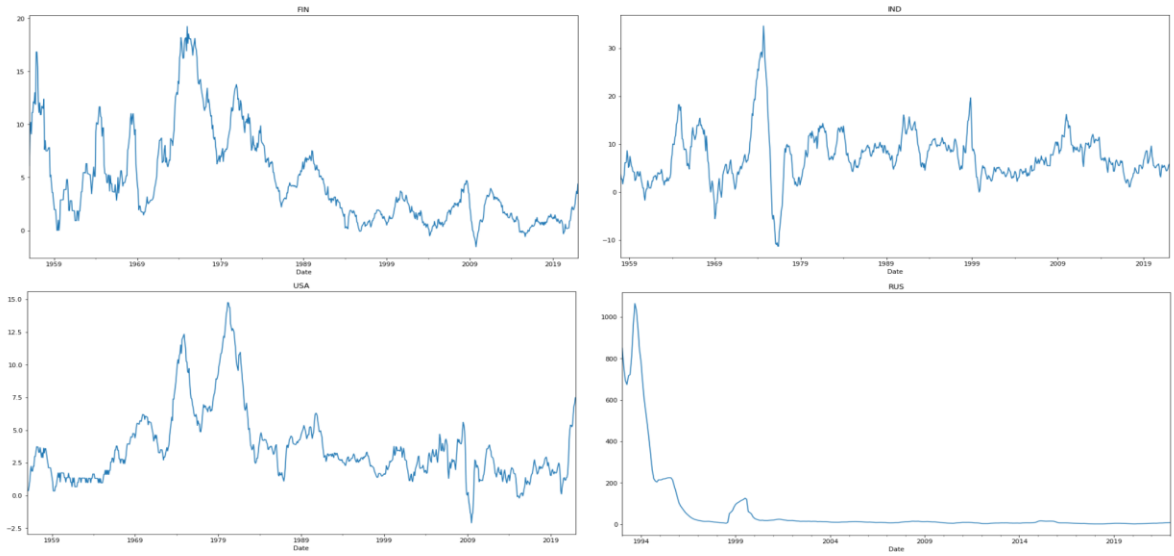

Figure 4 Monthly CPI data with No trend or seasonality

As it can be seen in both datasets, we do not have significant recognisable trend and seasonality. So, in this test it seems we have stationary data set with less signs of trend and seasonality. Now let’s see what ADF test indicates about the datasets.

```python
# Augmented Dickey-Fuller test (ADF) tests
for Location in list(df['Location'].unique()):
  print("\nLocation is: ", Location)
  adfuller_test(item_df[item_df['Location'] == Location]['Close'])
```

```
Location is:  FIN
ADF Test Statistic : -3.235734997696385
p-value : 0.018007166116157873
#Lags Used : 21
Number of Observations Used : 771
strong evidence against the null hypothesis(Ho), reject the null hypothesis. Data has no unit root and is stationary

Location is:  USA
ADF Test Statistic : -3.078298753191706
p-value : 0.02820142106666191
#Lags Used : 16
Number of Observations Used : 776
strong evidence against the null hypothesis(Ho), reject the null hypothesis. Data has no unit root and is stationary

Location is:  IND
ADF Test Statistic : -4.821815739849388
p-value : 4.939388364062457e-05
#Lags Used : 13
Number of Observations Used : 755
strong evidence against the null hypothesis(Ho), reject the null hypothesis. Data has no unit root and is stationary

Location is:  RUS
ADF Test Statistic : -5.19947005572882
p-value : 8.81401281228848e-06
#Lags Used : 17
Number of Observations Used : 331
strong evidence against the null hypothesis(Ho), reject the null hypothesis. Data has no unit root and is stationary
```

Figure 5 ADF test on CPI dataset indicating Stationary data

As the ADF tests indicating the Monthly CPI data is stationary in nature for all the locations. On the other hand, same ADF test done on Crypto currency dataset showing some of the crypto currency data is not stationary in nature. But looking at the ADF readings and test 1 plots both the tests indicate I can ignore the non-stationary cases in case of Crypto Currency. With this let's check out the model and training implementation.

Time Series Model - Facebook Prophet implementation

FB Prophet is a forecasting package in both R and Python that was developed by Facebook's data science research team. The goal of the package is to give business users a powerful and easy-to-use tool to help forecast business results without needing to be an expert in time series analysis. The underlying algorithm is a generalized additive model that is decomposable into three main components: trend, seasonality, and holidays. But seasonality and trend are two important, but difficult to quantify, components of a time series analysis and FB Prophet does a great job capturing both. Because it is a decomposable model, it is relatively easy to extract the coefficients of the model to understand the impact of seasonality, trend, holidays, and other regressor variables. Prophet does not perform well on non-stationary data because it is difficult to find the actual seasonality and trend of the data if the patterns are inconsistent.

Performance Comparison

Let’s look at how training of multiple time series models will get influenced when we use traditional way of sequential training of individual models

Execution Time:

Time Series Models	Normal Execution Time	Pyspark (Parallel processing) time
4 models for Crypto Currencies Daily Closing values forecasted	**1 min 68** seconds	**0.0058** seconds
4 models for Monthly CPI values	**61.714** seconds	**0.823** seconds

There is a clear difference in the time of executions between the two normal training of models in sequential manner and training models in parallel using Pyspark.

Accuracy:

Used Mean Absolute Error (MAE) to measure the accuracy of the models. The MAE is defined as the average of the absolute difference between forecasted and true values. Where y_i is the expected value and x_i is the actual value (shown below formula). The letter n represents the total number of values in the test set.

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

The performance of each model is same in both way of executions:

	Normal Execution Time	PySpark (Parallel processing) time
4 models for Crypto Currencies Daily Closing values forecasted	bitcoin : MAE: 2339.380 ethereum: MAE: 106.012 cardano : MAE: 0.045 tether : MAE: 0.004	bitcoin : MAE: 2339.380 ethereum : MAE: 106.012 cardano : MAE: 0.045 tether : MAE: 0.004
4 models for Monthly CPI values	FIN : MAE: 0.986 USA : MAE: 0.734 IND : MAE: 3.540 RUS : MAE: 7.358	RUS : MAE: 0.986 RUS : MAE: 0.734 RUS : MAE: 3.540 RUS : MAE: 7.358

Technologies used

Python, Time series forecasting using Facebook Prophet, Big Data Technology: Spark (Pyspark), Graphical presentation of data: Matplotlib module, Databricks

Output for each model

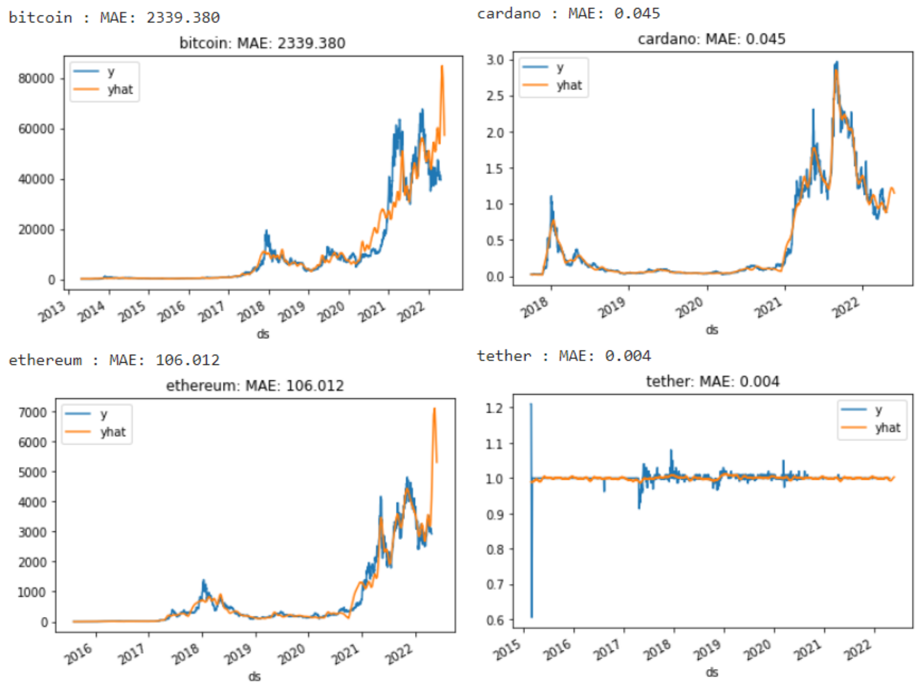

Figure 6Crypto Currency Forecasting output

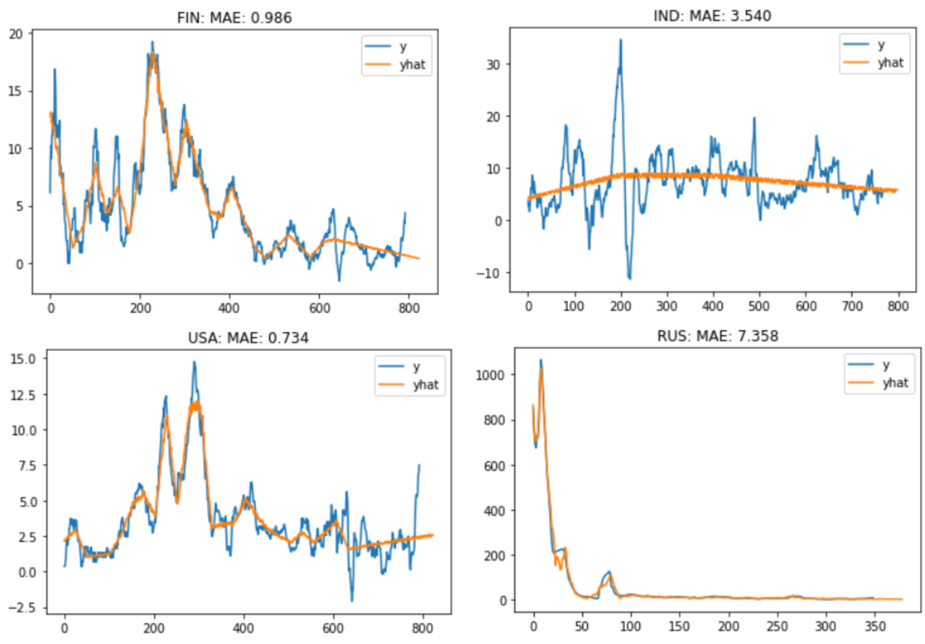

Figure 7 Monthly CPI forecasting for 4 countries

Conclusion

In the implementation of multiple time series forecasting I learned following things:

1. Employing Pyspark in Python to train multiple time series models
2. Understanding sequential implementation and parallel processing implementation in models training
3. Handling challenges in setting up environment for the Spark implementation on the Python. Required packages of Java, Pyspark, Facebook Prophet
4. Explored Pyspark and Facebook Prophet features in training multiple time series models
5. Data can be further prepared made stationary and tried out with different time series models such as SARIM, ARIMA, ARMA-GARCH Model, etc.

In conclusion it can be said that to train multiple time series models the parallel processing wins with clear difference in execution time. This is specifically important since if there are thousands of time series models we want to train and forecast for the future values this approach will surely derive the benefit the big data technologies powerful parallel computing approach.

Also, if noticed the same approach can be employed to train multiple machine learning or deep learning models too by using parallel computing on cluster of machines on cloud platform. This approach also proves that if implemented correctly it will maintain the same performance of models which is seen in sequential approach. This project proves that there is huge difference in training time for at the same time maintain the same performance of the models.

References and Appendix

1. Facebook prophet:

   https://facebook.github.io/prophet/docs/installation.html#python

2. https://www.tessellationtech.io/facebook-prophet-tutorial-time-series-forecasting/

3. Performance measurements: https://analyticsindiamag.com/a-guide-to-different-evaluation-metrics-for-time-series-forecasting-models/

4. Evaluation matrix for time series: https://analyticsindiamag.com/a-guide-to-different-evaluation-metrics-for-time-series-forecasting-models/

Code repository is created on Github:

1. Monthly CPI Multiple Time Series Forecasting for countries:

   https://github.com/Yogeshnaik1190/Big-Data-Technologies/blob/main/Project-Multiple%20Time%20Series%20using%20Spark%20and%20Facebook%20Prophet/BDT_CPI_time_series_analysis_forecasting.ipynb

2. Daily Crypto Currency Time Series Forecasting for Cryptocurrencies:

   https://github.com/Yogeshnaik1190/Big-Data-Technologies/blob/main/Project-Multiple%20Time%20Series%20using%20Spark%20and%20Facebook%20Prophet/BDT_Cryptocurrency_time_series_forecasting.ipynb