Predicting Cryptocurrency Prices Using Various Time Series ML Models

A report submitted in partial fulfilment of the

requirement for the award of certification of

IN-HOUSE SUMMER TRAINING

in

Data Science

By

Anshul Kannaujia - 00611502722

Navya Sharma - 04311502722

Vaibhav Gupta - 05511502722

under the guidance of

Mr. Arpit Shrotriya

Ms. Jyotika Bhati

Trainee, Veeyo Tech

Computer Science and Engineering

Bharati Vidyapeeth's College of Engineering,

New Delhi – 110063, INDIA

August 2024

Table of contents

List of Figures

List of Tables

About Company

Veeyotech is a company that promotes technological advancement by offering cutting-edge tools and a knowledgeable faculty that enable programmers and developers to push the boundaries of their respective professions.

Apart from its remarkable technological capabilities, Veeyotech is renowned for fostering a cooperative and encouraging atmosphere. The business fosters innovation and lifelong learning, creating an environment conducive to the success of technicians and researchers. This encouraging atmosphere is enhanced by a group of professionals who are constantly available to provide technical support and direction, guaranteeing that users can overcome any obstacles they have while working on their projects.

For programmers and developers looking to push the limits of technology, Veeyotech is a great partner because of its commitment to creating an innovative and supportive culture. Veeyotech contributes significantly to the advancement of technical research and development by offering the required instruments, resources, and experience, enabling the realization of inventive concepts.

CERTIFICATE

This is to certify that **<u>Navya Sharma, Anshul Kannaujia and Vaibhav Gupta</u>** of **CSE-1** has successfully completed the Data Science Project under the guidance of **Mr. Arpit Shrotriya and Ms. Jyotika Bhati** (Faculty of Veeyotech) as prescribed **by Guru Gobind Singh Indraprastha University (GGSIPU)**

Signature

Acknowledgements

First and foremost, we want to express our sincere gratitude to Mr. Arpit Shrotriya and Ms. Jyotika Bhati. Their extensive knowledge, unwavering compassion, and remarkable tolerance greatly enhanced our research experience. Throughout the course of our study, they provided us with invaluable advice, insightful guidance, and consistent encouragement, all of which played a crucial role in keeping us motivated and focused. Their dedication to fostering a positive and productive learning environment has left a lasting impression on us, and we are deeply appreciative of their support.

We would especially like to extend our heartfelt thanks to Veeyotech for their generous provision of tools and a supportive environment for our study. The development of this project was significantly aided by the availability of their comprehensive library and extensive database. Veeyotech's resources allowed us to access a wealth of information and materials that were instrumental in our research. Additionally, their commitment to fostering innovation and learning provided us with an inspiring and conducive atmosphere to explore and develop our ideas.

Furthermore, we are grateful for the technical assistance we received from the Veeyotech team. Their expertise and willingness to assist with any technical challenges ensured that our research proceeded smoothly. The collaborative spirit within Veeyotech was vital in overcoming obstacles and achieving our project goals.

Abstract

In recent years, the volatility and rapid growth of cryptocurrency markets have spurred significant interest in the development of predictive models to forecast cryptocurrency prices. The creation of predictive models to estimate cryptocurrency values has attracted substantial interest in recent years due to the markets' volatility and explosive growth. With an emphasis on their approaches, efficacy, and potential for practical use, this study examines and assesses a number of cryptocurrency datasets such as Bitcoin, Solana and Ethereum, and uses machine learning models used to forecast cryptocurrency values, such as, Prophet, neural-Prophet and ARIMA . Deep learning models, such as Auto Regressive Integrated Moving Average (ARIMA),is given special attention. An extensive grasp of the difficulties and factors involved in predicting bitcoin prices is provided by discussing data preparation techniques, feature selection, and model evaluation metrics. According to the results, while no single machine learning model is perfect ARIMA model proves to be one of the most efficient.

Introduction

Due to their extreme volatility and decentralized character, cryptocurrencies have become a revolutionary asset class that has drawn the interest of investors, researchers, and financial institutions. For market participants, the sharp price swings of cryptocurrencies like Ethereum, Solana, and Bitcoin offer both opportunities and challenges. Making wise investing decisions, managing risk, and creating successful trading methods all depend on accurate price forecast. In this extremely unpredictable market, traditional financial models frequently prove to be insufficient, which has led to the investigation of more advanced methods like machine learning and statistical models.

The Auto Regressive Integrated Moving Average (ARIMA) model has become well-known among time series forecasting models due to its ease of use and efficiency in addressing seasonality and linear patterns in financial data. Furthermore, more recent models have been created to handle intricate patterns and non-linear trends, such as Facebook's Prophet and its variation Neural Prophet. Although these models are widely used in traditional banking, there hasn't been much research done on applying them to bitcoin markets. In light of the particulars of cryptocurrency markets, this research aims to close this gap by examining the predictive capabilities of ARIMA, Prophet, and Neural Prophet models.

We make use of past price data from Solana, Ethereum, and Bitcoin—three of the most popular cryptocurrencies. The first section of the paper gives a thorough introduction to the ARIMA, Prophet, and Neural Prophet models, outlining their theoretical foundations as well as the procedures for choosing and fine-tuning their parameters. The preprocessing methods required to get cryptocurrency price data ready for these models are then covered, including how to deal with non-stationarity, noise, and volatility. The empirical analysis compares each model's accuracy and resilience in predicting cryptocurrency prices in order to assess each one's predictive performance.

Objective of Project

This project's main goal is to create and assess prediction models for predicting the prices of significant cryptocurrencies, namely Solana, Ethereum, and Bitcoin. The goal is to ascertain how well the various time series forecasting methods—namely, ARIMA, Prophet, and Neural Prophet—capture the distinctive features of fluctuations in cryptocurrency prices by utilizing their respective strengths. The project's particular goals are:

Model Development: To put into practice the Prophet, Neural Prophet, and ARIMA models for predicting bitcoin prices while making sure each model is set up and optimized for the dataset.

Data Preprocessing: In order to prepare the historical price data of Solana, Bitcoin, and Ethereum for reliable time series forecasting, it is necessary to handle non-stationarity, noise, and volatility.

Performance Evaluation: Using pertinent measures like Mean ,Root Mean Squared Error (RMSE), assess the predictive performance of the ARIMA, Prophet, and Neural Prophet models.

Comparative Analysis: To determine which model, or combination of models, offers the best forecasting accuracy, this study compares the benefits and drawbacks of the ARIMA, Prophet, and Neural Prophet models in the context of bitcoin price prediction.

Generating Insights: To shed light on how well these models work for actual trading and investing tactics while pointing out possible directions for development and use in cryptocurrency markets.

Project Description

The objective of this research is to use the capabilities of multiple time series forecasting approaches to construct and assess predictive models for predicting the prices of the three main cryptocurrencies: Ethereum, Solana, and Bitcoin. Because of the extreme volatility and distinct market features of cryptocurrencies, conventional financial models frequently fail to produce reliable forecasts. This project is focused on addressing these issues and enhancing forecast accuracy through the use of cutting-edge models such as ARIMA, Prophet, and Neural Prophet.

The wide range of features offered by the library yfinance made it possible for us to effectively organize and retrieve massive amounts of data, which was necessary for the in-depth research and modeling that our study required. The correctness and dependability of our data, which served as the basis for the predictive models created and assessed in this study, were guaranteed by utilizing yfinance.

There were many machine learning models used in our study such as Prophet, Neural-prophet and ARIMA, for best results, stationary data is necessary for the ARIMA model, which is well-known for its efficiency in time series forecasting. The original Solana, Ethereum, and Bitcoin datasets, however, were not stationary. To convert the data into a stationary format, therefore, a number of preprocessing operations were carried out.

The machine learning models used are as follows:

- **ARIMA**

A well-liked statistical technique for examining and projecting time series data is the ARIMA (Auto Regressive Integrated Moving Average) model. It is composed of three parts: moving average (MA), differencing (I), and autoregression (AR). In the MA portion, the error term is modeled as a linear combination of previous error terms, but in the AR component, the variable is regressed on its own previous values. The time series is made stationary using differencing, which ensures that statistical parameters like mean and variance don't change over time. The formula for the ARIMA model is ARIMA(p,d,q), where p is the number of lag observations (AR terms) that are included in the model, d is the number of difference operations performed on the raw observations, and q is the moving average window size.

An ARIMA model's general formula is as follows:

$$Y_t = c + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \ldots + \phi_p Y_{t-p} + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots + \theta_q \epsilon_{t-q} + \epsilon_t$$

where

σ t is the error term,

c is a constant,

Y_t is the value at time t,

ϕ represents the coefficients for the AR terms,

θ represents the coefficients for the MA terms

For short-term forecasting, ARIMA models are frequently utilized in a variety of disciplines, including finance and economics.

- **PROPHET**

Facebook created the Prophet model, an open-source program for predicting time series data with trends, seasonality, and holiday impacts. It works especially well with datasets that have missing data and significant seasonal patterns, which makes it appropriate for a variety of uses, including predicting bitcoin prices. Trend, seasonality, and holidays are the three primary components into which the Prophet model divides time series data. In order to capture the non-periodic changes in the data, the trend component might be either logistic or linear. To represent periodic variations like daily, weekly, or annual cycles, seasonality is approximated using Fourier series. The impact of particular days on the time series is taken into consideration by the holiday component.

Formula for the model is:

$$y(t)=g(t)+s(t)+h(t)+\epsilon t$$

where,

ϵ t is the error term,

s(t) is the seasonal component,

h(t) is the holiday effect,

y(t) is the expected value at time t. g(t) represents the trend function.

Prophet is a useful tool for assessing and projecting cryptocurrency trends since it is simple to use, requires little adjustment of the parameters, and performs well for forecasting tasks.

- **Neural Prophet**

A sophisticated time series forecasting tool, the Neural Prophet model incorporates neural network architectures—which are specifically engineered to enhance accuracy and flexibility in catching intricate patterns—into Facebook's Prophet model. In order to better capture non-linear linkages and interactions in the data, Neural Prophet incorporates neural network capabilities into the fundamental elements of the original Prophet model, including trend, seasonality, and holidays. The autoregressive neural network components are used by the model to capture lagged dependencies once it breaks down the time series data into additive parts. The general formula for Neural Prophet is:

$$y(t)=g(t)+s(t)+h(t)+AR(yt-1,yt-2,...,yt-p)+\epsilon$$

where,

g(t) is the trend,

s(t) uses Fourier series to capture seasonality,

h(t) takes into account holiday effects,

$y(t)$ is the predicted value at time t,

$AR(y_{t-1}, y_{t-2}, ..., y_{t-p})$ denotes the autoregressive component modeled by neural networks, ϵ_t is the error term.

Because of its adaptive and user-friendly architecture, Neural Prophet is an effective tool for predicting in complicated and turbulent markets, such as cryptocurrency markets, where it is critical to identify detailed patterns and trends.

Each model is trained on the pre-processed historical price data of Bitcoin, Ethereum, and Solana. The models' performance will be evaluated using metrics such as Root Mean Squared Error (RMSE), and r-squared values.

Implementation Procedure

Auto Regressive Integrated Moving Average (ARIMA):

First we import the dataset through the library yfinance.

```
import yfinance as yf
btc_ticker=yf.Ticker("BTC-USD")
btc=btc_ticker.history(period="max")
```

The given code snippet shows how the dataset for bitcoin is loaded into the program.

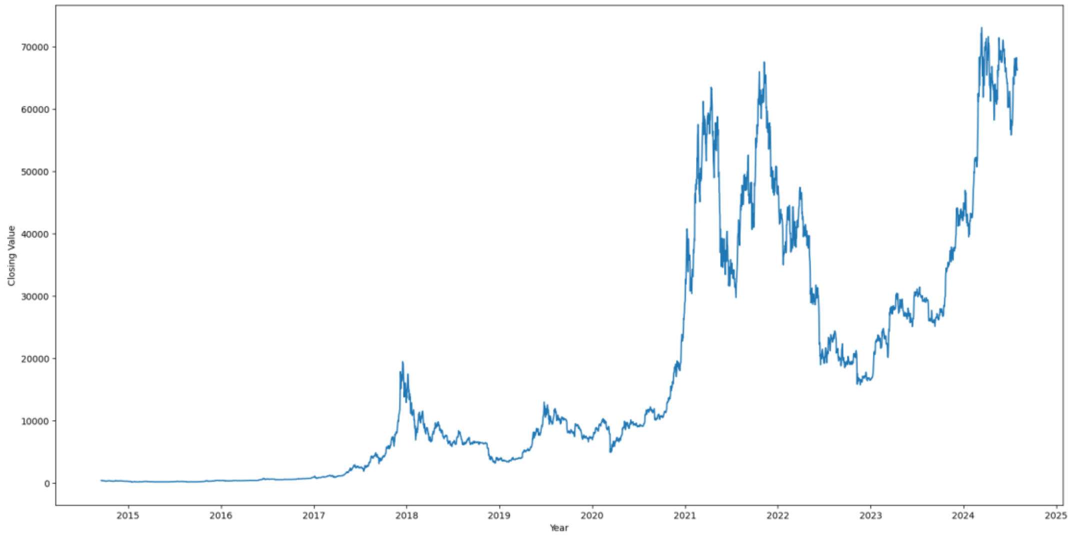
Now we import the required libraries

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
```

Here matplotlib is used to plot the graphs, numpy is used for effective arithmetic calculations, pandas is used to work with structured data, our dataset in this case, and seaborn is used for data visualization.

After importing the required libraries we plot the values of bitcoin with respect to each year and analyse the trends in the dataset through a line graph.

```
btc=btc['Close'].copy()
plt.figure(figsize=(20,10))
plt.xlabel("Year")
plt.ylabel("Closing Value")
plt.plot(btc)
```

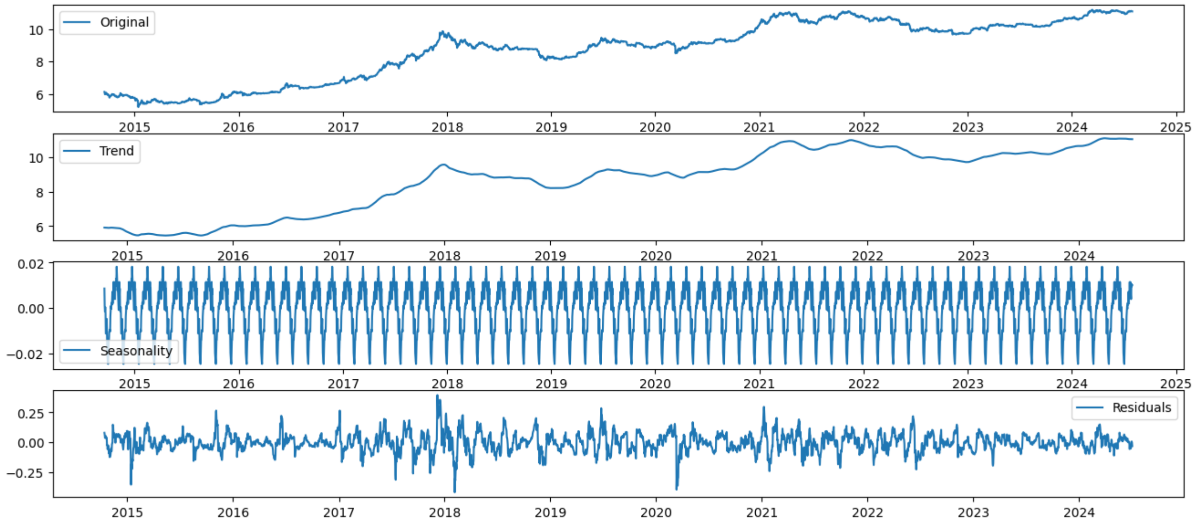


As the data present in the dataset was already plotted with respect to equal time intervals we needn't do any preprocessing to analyse the data, but to make the dataset stabilized we used log function in the numpy library, this makes the data more homoscedastic as the very volatile nature of bitcoin prices make its data very heteroscedastic in nature.

```
btc=np.log(btc)

from statsmodels.tsa.seasonal import seasonal_decompose
decomposition = seasonal_decompose(btc,period=54)

trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid

plt.figure(figsize=(16,7))
fig = plt.figure(1)

plt.subplot(411)
plt.plot(btc, label='Original')
plt.legend(loc='best')
plt.subplot(412)
plt.plot(trend, label='Trend')
plt.legend(loc='best')
plt.subplot(413)
plt.plot(seasonal,label='Seasonality')
plt.legend(loc='best')
plt.subplot(414)
plt.plot(residual, label='Residuals')
plt.legend(loc='best')

train = btc.iloc[:len(btc)-365]
test = btc.iloc[len(btc)-365:]
```

The decomposition function is used to break down a time series into its constituent components, typically including trend, seasonality, and residual (noise).

This is useful for understanding the underlying patterns in the data and for improving forecasting accuracy by modelling each component separately, and then the graph plotted is as follows:



The given graph shows the trends in which the prices of bitcoin changes.

After Preprocessing the data and before implementing the ARIMA model we need to justify the p, q and d values, for that purpose we have the function auto_arima.

```
from pmdarima import auto_arima
import warnings
warnings.filterwarnings('ignore')
stepwise_fit=auto_arima(train,trace=True,suppress_warnings=True)
stepwise_fit.summary()
```

From the above code we get,

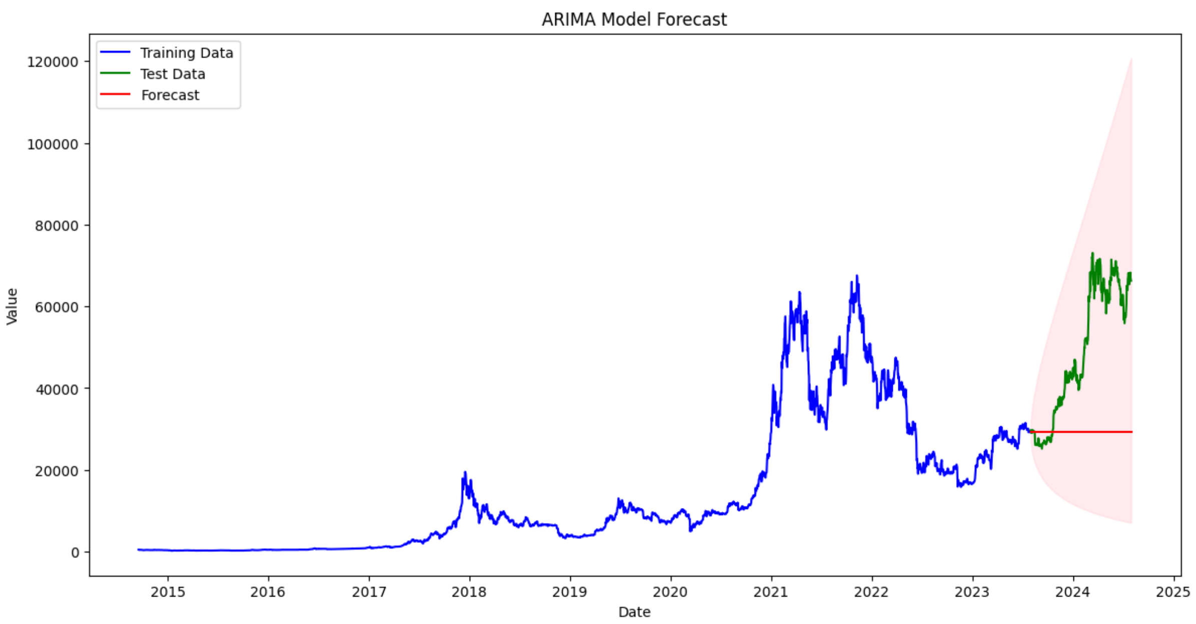
```
Best model:  ARIMA(0,1,0)(0,0,0)[0] intercept
```

Now we train the **ARIMA model** on our dataset.

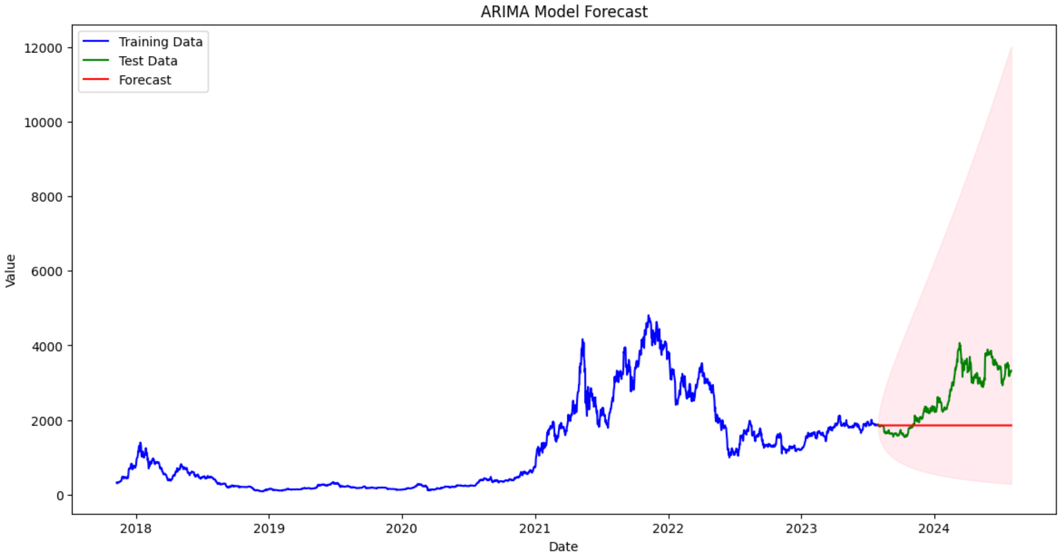
First we import the model from the library statsmodels and train it on the p, q and d values found out before, then the function get forecast is used to forecast the predicted values.

```
from statsmodels.tsa.arima.model import ARIMA
model=ARIMA(train, order=(0,1,0))
results=model.fit()

forecast = results.get_forecast(steps=len(test))
forecast_index = test.index
forecast_mean = forecast.predicted_mean
forecast_conf_int = forecast.conf_int()
# Plotting+
plt.figure(figsize=(14, 7))

# Plot training data
plt.plot(train.index, np.exp(train), label='Training Data', color='blue')

# Plot test data
plt.plot(test.index, np.exp(test), label='Test Data', color='green')

# Plot forecasts
plt.plot(test.index, np.exp(forecast_mean), label='Forecast', color='red')
plt.fill_between(forecast_index, np.exp(forecast_conf_int.iloc[:, 0]), np.exp(forecast_conf_int.iloc[:, 1]), color='pink', alpha=0.3)
plt.title('ARIMA Model Forecast')
plt.xlabel('Date')
plt.ylabel('Value')
plt.legend()
plt.show()
```

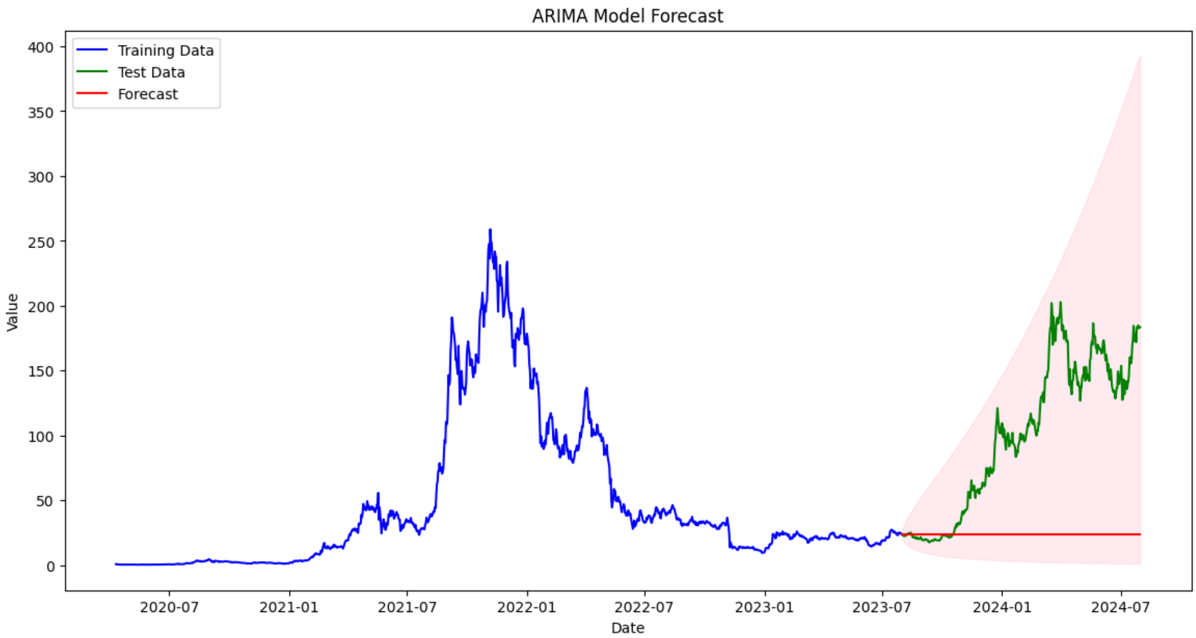
The plotted graph is as follows, the blue line chart shows the training data, the green line chart shows the testing data and the red line shows the forecasting data.



Similarly for the Ethereum dataset set get,



And for Solana's dataset we get,



Neural Prophet:

Similar to the ARIMA model we first import the yfinance dataset to the program so as to access the Bitcoin's dataset,

```
import yfinance as yf
btc_ticker=yf.Ticker("BTC-USD")
btc=btc_ticker.history(period="max")
```

Now we first import the required libraries,

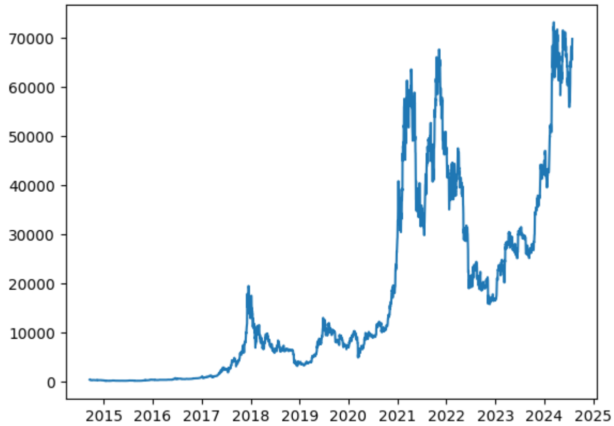
```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
```

Here matplotlib is used to plot the graphs, numpy is used effective arithmetic calculations, pandas is used to work with structured data, our dataset in this case, and seaborn is used for data visualization.

Now from the library neuralprophet we import NeuralProphet to train our given dataset on the neural prophet model, and we first plot our dataset so as to show the trends in the prices of bitcoin.

```
from neuralprophet import NeuralProphet
plt.plot(btc_reset['Date'],btc_reset['Close'])
plt.show()
```

The plotted graph is plotted with the time interval in relation to there prices,



We now split the training and testing data,

```
btc_train=btc_final.iloc[:len(btc_final)-365]
btc_test=btc_final.iloc[len(btc_final)-365:]
```

the split is based on the present and previous years data.

Now we fit the given dataset into the neural prophet model, and give it a epoch or train it for a thousand times.

```
model_np=NeuralProphet(batch_size=13)
model_np.fit(btc_train,freq='D',epochs=1000)
```

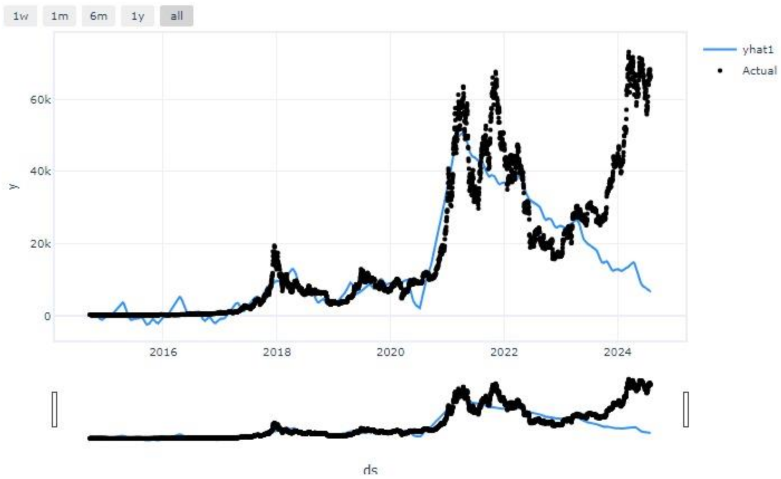
After fitting the data we apply the predict function,

```
forecast=model_np.predict(btc_final)
forecast.head()
```

Now we plot the forecasted values,

```
plot1 = model_np.plot(forecast)
plot1.show()
```

the plotted graph is,



Similarly for the dataset of Ethereum we get,

For Solana's dataset we get,

Prophet:

Similar to the Neural prophet model we first import the yfinance dataset to the program so as to access the Bitcoin's dataset,

```
import yfinance as yf
btc_ticker=yf.Ticker("BTC-USD")
btc=btc_ticker.history(period="max")
```

Now we import the prophet library to gain access to the prophet dataset and their functions,

```
from prophet.plot import plot_plotly, plot_components_plotly
from prophet import Prophet
```

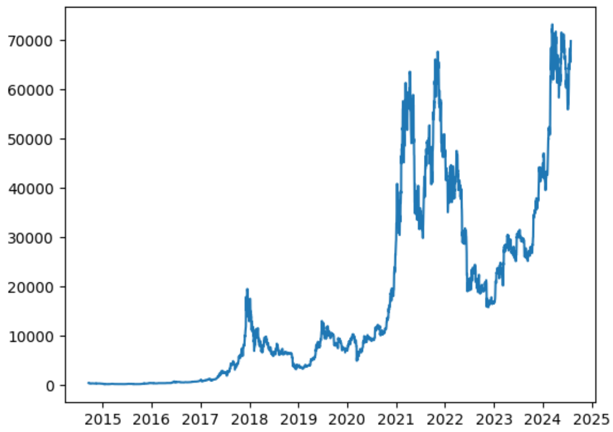
here the prophet library is used to import the Prophet model and its components from the prophet.plot library to plot their components on forecasting.

Now the rest of the required libraries are imported.

```python
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
```

Here matplotlib is used to plot the graphs, numpy is used effective arithmetic calculations, pandas is used to work with structured data, our dataset in this case, and seaborn is used for data visualization.

We now plot our dataset so as to show the trends in the prices of bitcoin.



We now split the training and testing data,

```python
btc_train=btc_final.iloc[:len(btc_final)-365]
btc_test=btc_final.iloc[len(btc_final)-365:]
```

the split is based on the present and previous years data.

Now we fit our dataset into the Prophet model using the code,

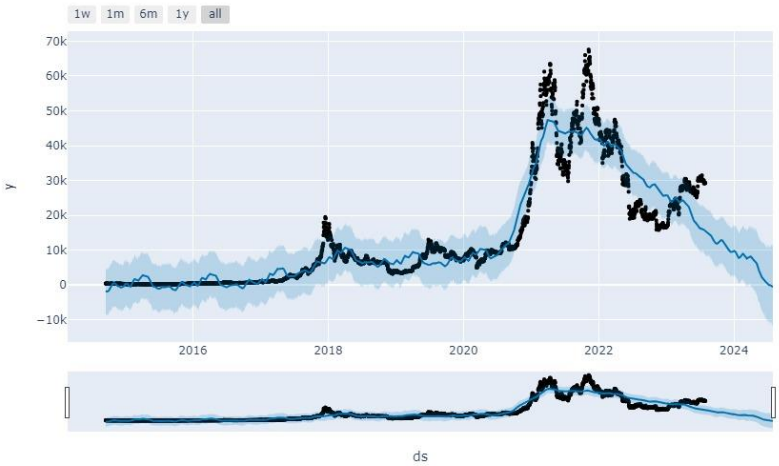
```python
model_p= Prophet()
model_p.fit(btc_train)
```

Now we use the predict function to predict the values,

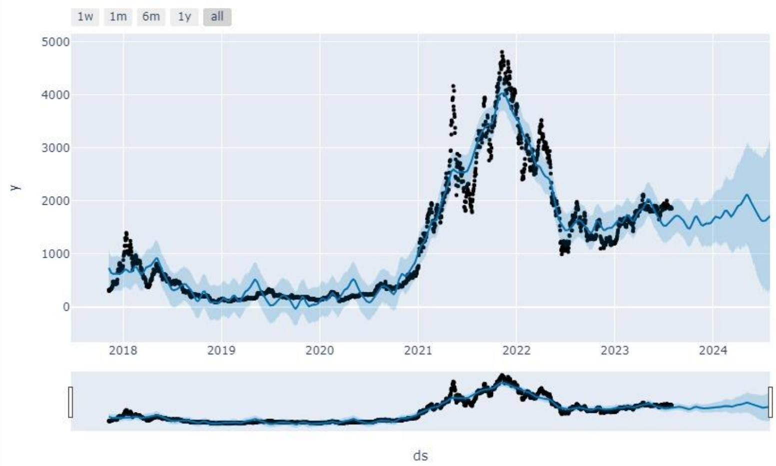
```
future = model_p.make_future_dataframe(periods=365)
forecast = model_p.predict(future)
```

This line generates a DataFrame that extends the original time series data into the future, creating a placeholder for future dates.

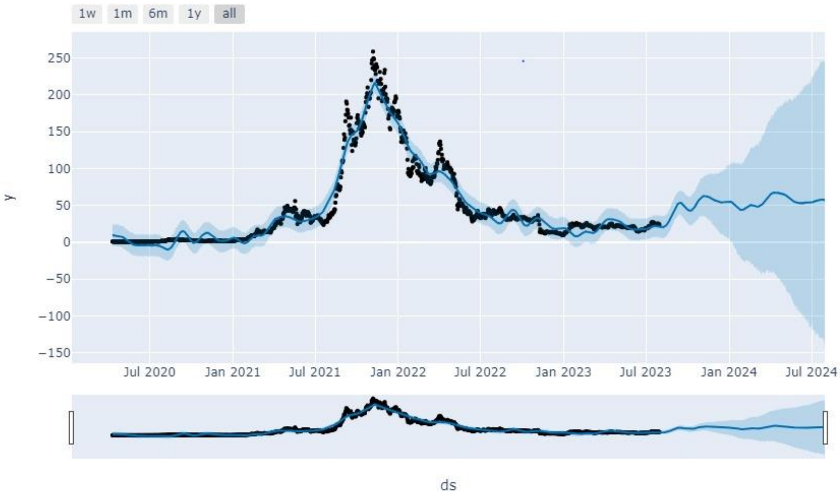
Now we plot the values for the found predictions,



Similarly for the Ethereum dataset we get,



Now for the Solana dataset,



Project Visualization:

To effectively showcase our project's outcomes, we employed two key visualization tools: 'Streamlit' and 'Power BI'. Our Streamlit app demonstrates the model's functionality, providing interactive visualizations and insights, while our Power BI dashboard offers a comprehensive analysis of the data, including detailed charts and metrics.

Let's delve into each of these visualizations in detail to understand how they enhance our project's presentation and insights.

Streamlit web application

Streamlit is a powerful and user-friendly framework that allows for the rapid development of interactive web applications specifically tailored for data science and machine learning projects. With its intuitive syntax and real-time updates, Streamlit makes it easy to build and deploy data-driven applications, enabling users to visualize data, interact with models, and gain insights effortlessly.

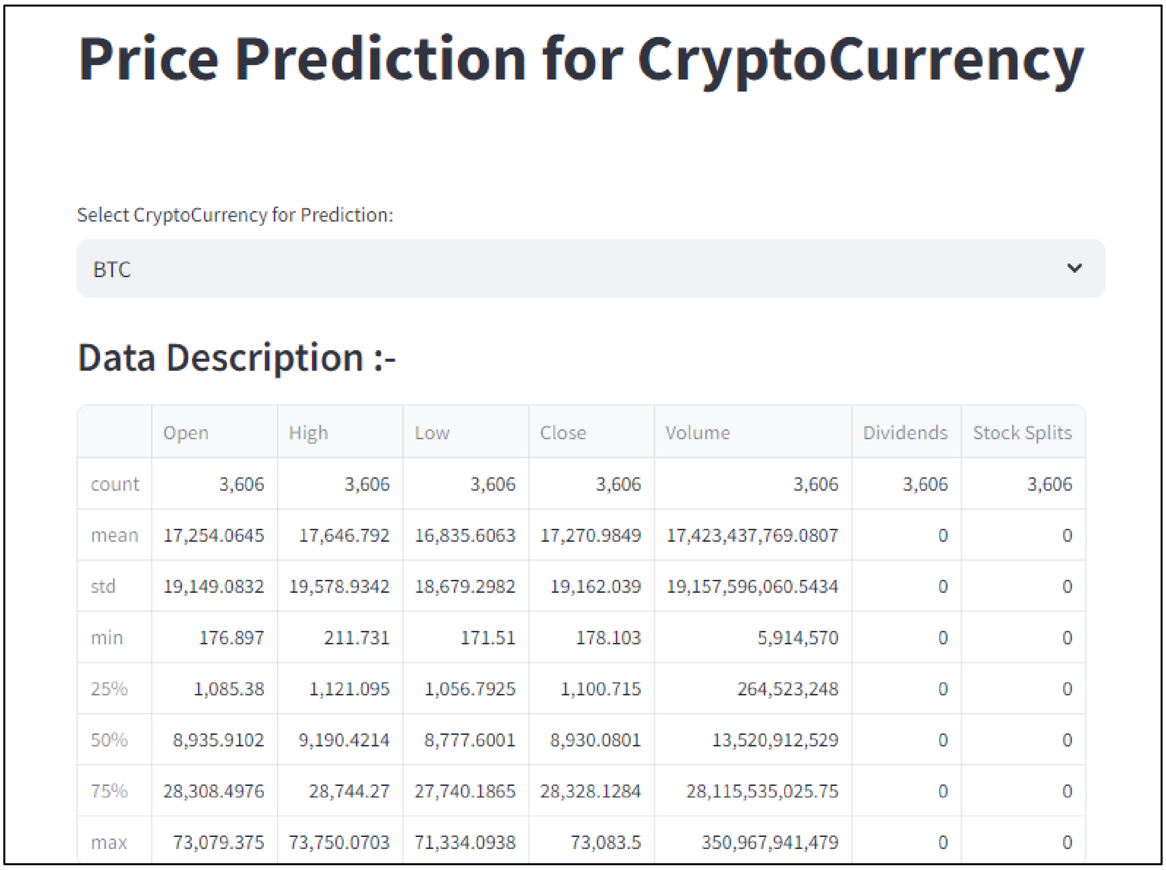
To work with it, we first imported the following packages :-

```
import streamlit as st
from prophet.plot import plot_plotly
import plotly.graph_objects as go
```

'Streamlit' library contains all the important functionality for creating interactive web applications, enabling users to visualize and interact with data seamlessly.

'plot_plotly' allows for interactive visualizations of time series forecasts using Prophet and 'plotly.graph_objects' provides tools for creating customizable and detailed plots.

Our Application first asks user to select the Crypto about which they want to know the predictions. Then it shows user the description of the dataset of that Crypto, which it will use to train the ARIMA model. We chose ARIMA model for our web app as its predictions had the least error among all the models.



Price Prediction for CryptoCurrency

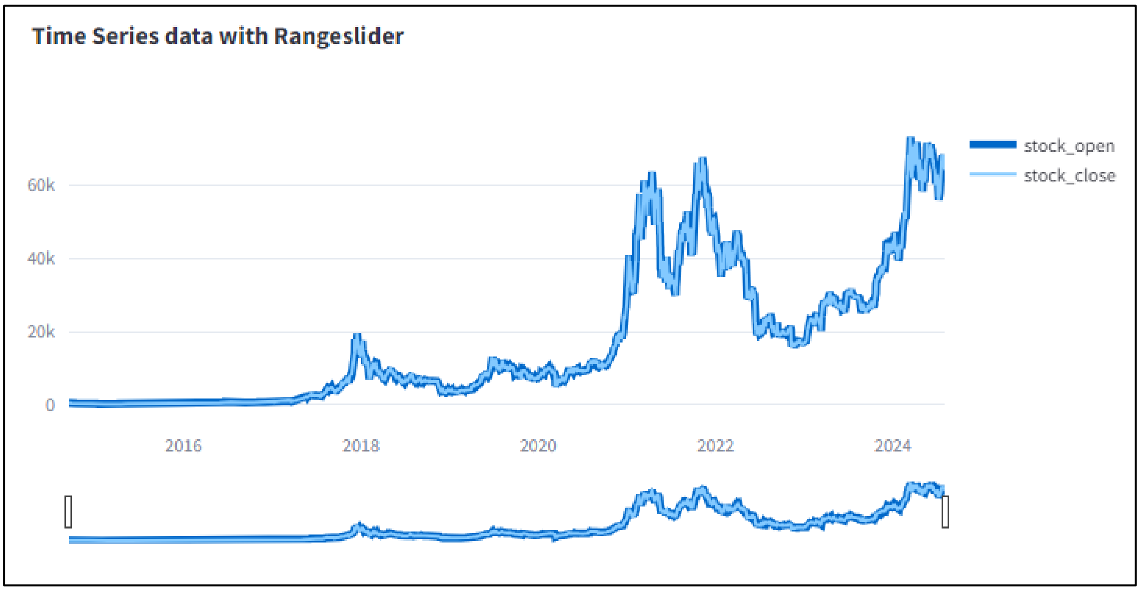
Select CryptoCurrency for Prediction:

BTC

Data Description :-

	Open	High	Low	Close	Volume	Dividends	Stock Splits
count	3,606	3,606	3,606	3,606	3,606	3,606	3,606
mean	17,254.0645	17,646.792	16,835.6063	17,270.9849	17,423,437,769.0807	0	0
std	19,149.0832	19,578.9342	18,679.2982	19,162.039	19,157,596,060.5434	0	0
min	176.897	211.731	171.51	178.103	5,914,570	0	0
25%	1,085.38	1,121.095	1,056.7925	1,100.715	264,523,248	0	0
50%	8,935.9102	9,190.4214	8,777.6001	8,930.0801	13,520,912,529	0	0
75%	28,308.4976	28,744.27	27,740.1865	28,328.1284	28,115,535,025.75	0	0
max	73,079.375	73,750.0703	71,334.0938	73,083.5	350,967,941,479	0	0

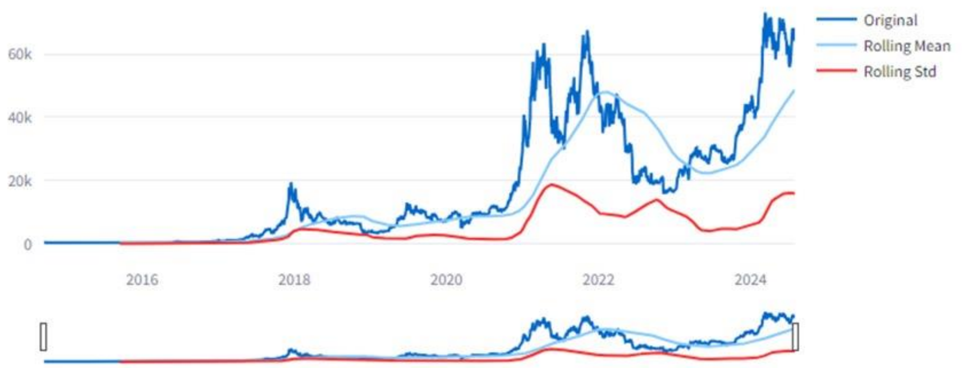
Then our application shows the time series plot of opening and closing prices of that Crypto.



Our application also explains the need for Log Transformation by plotting rolling mean and rolling standard deviation of the dataset before and after transformation.

Before Transformation:

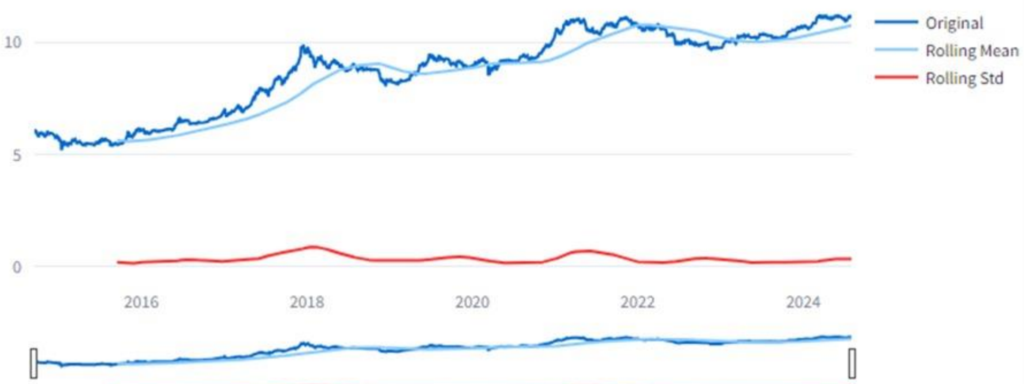
Plotting Rolling Mean and Standard Deviation to check Stationarity



We can see that Mean and Standard Deviation of data is not constant, so data is not stationary. To make data stationary we will be doing log-transformation of the data and further try to plot Rolling Mean and Standard Deviation:

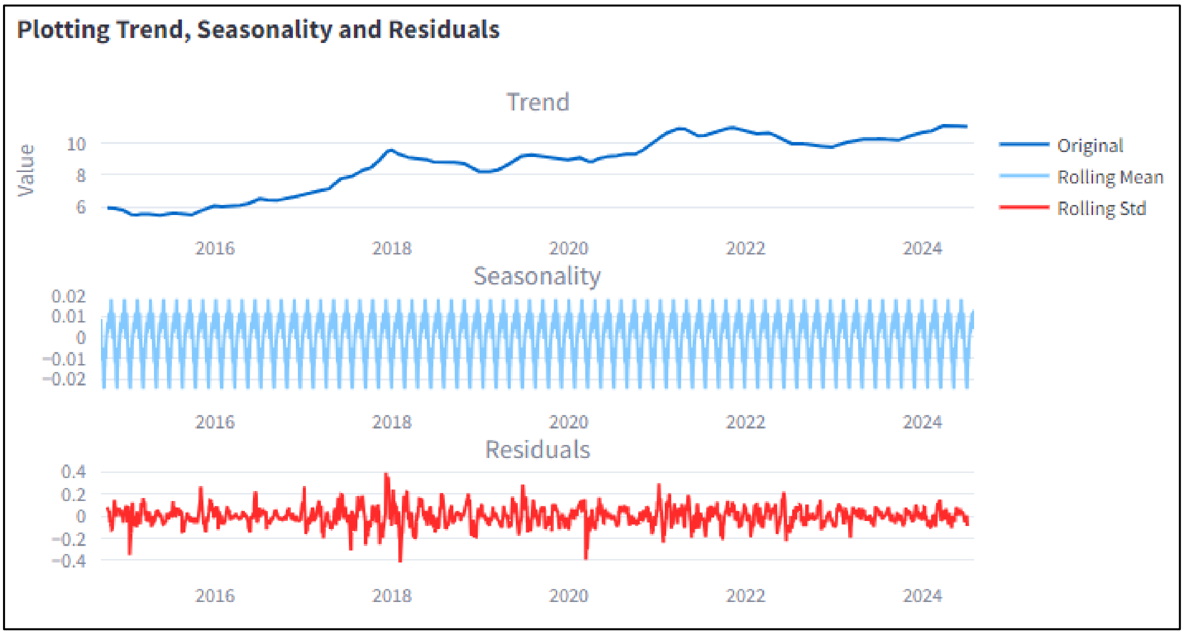
After Transofrmation:

Plotting Rolling Mean and Standard Deviation to check Stationarity



We can see that Mean and Standard deviation are somewhat constant now in comparison to previous one. So, we will proceed with using the log-transformed data for all subsequent analyses and modeling.

As knowing about trend, seasonality and residuals in dataset is important, so then our application presents plots to user that clearly depict each of these components.

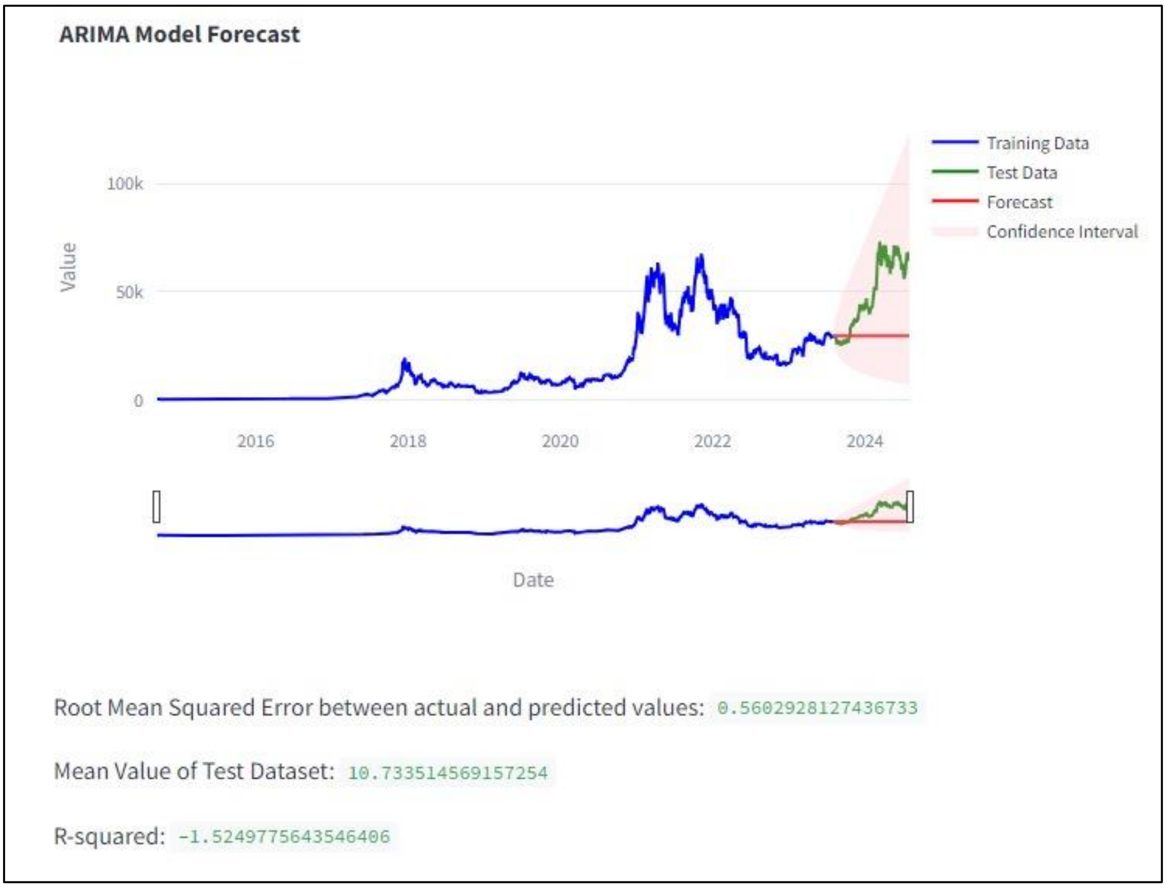


It also shows the user outcomes of 'Dickey-Fuller Test' on the data which also explains about the stationarity of data.

Dickey-Fuller Test on train data :

	Metric	Value
0	Test Statistic	-1.0378
1	p-value	0.7392
2	Lags Used	10
3	No. of Obs	3,230
4	Critical Value (1%)	-3.4324
5	Critical Value (5%)	-2.8624
6	Critical Value (10%)	-2.5672

At last, our application presents the user predicted values along with a 95% confidence interval, which gives an idea of the range that the actual future values are anticipated to fall inside. Additionally, it displays key performance metrics including RMSE, mean, and R^2 values, offering insights into the accuracy and quality of the predictions.

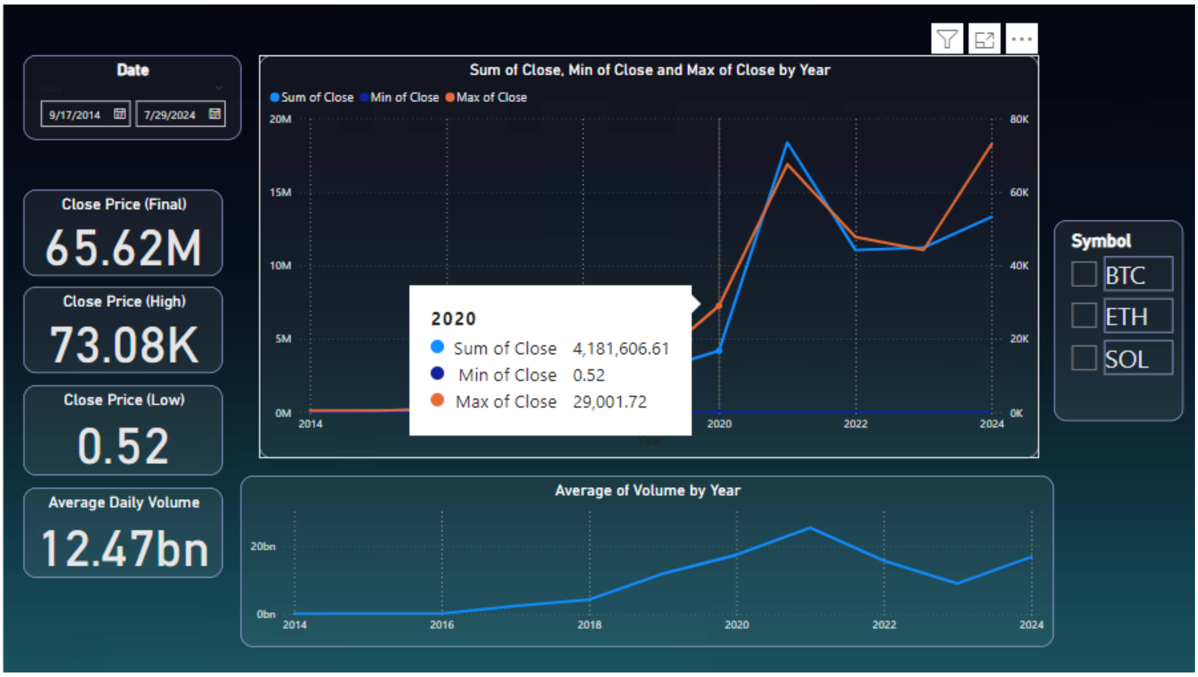


PowerBI :

Following is the face dashboard of Cryptocurrency Analysis.

The following image shows how the dates are selected for analysing the dataset,

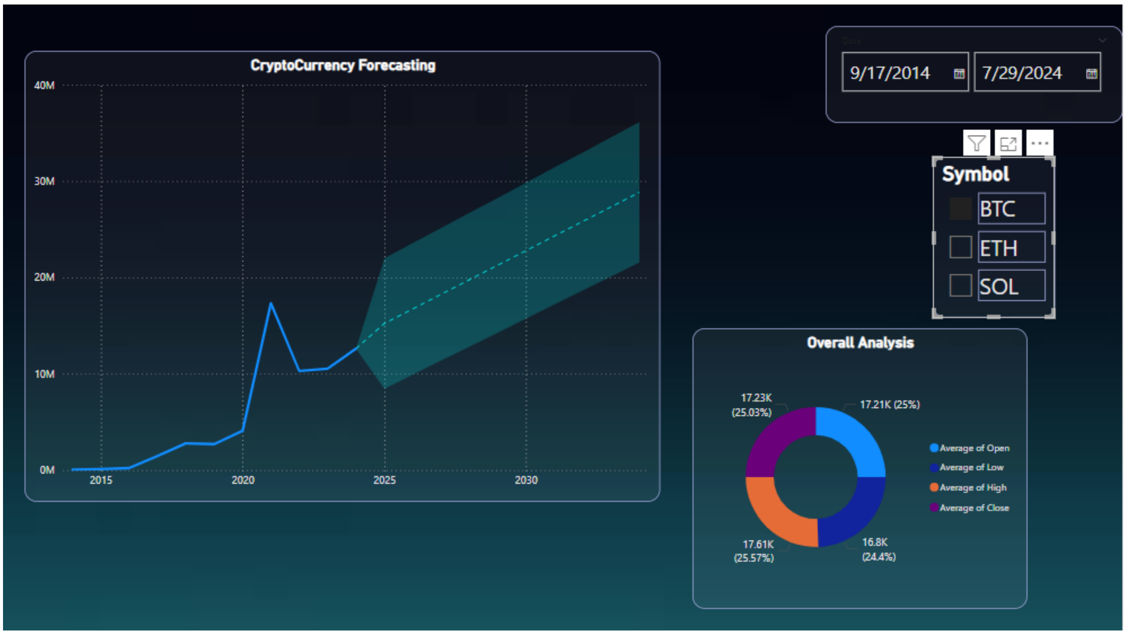
The following dashboard in powerbi shows the changes in the prices of the cryptocurrency with respect to time, it also highlights the highest and lowest price on yearly and its respective closing price.



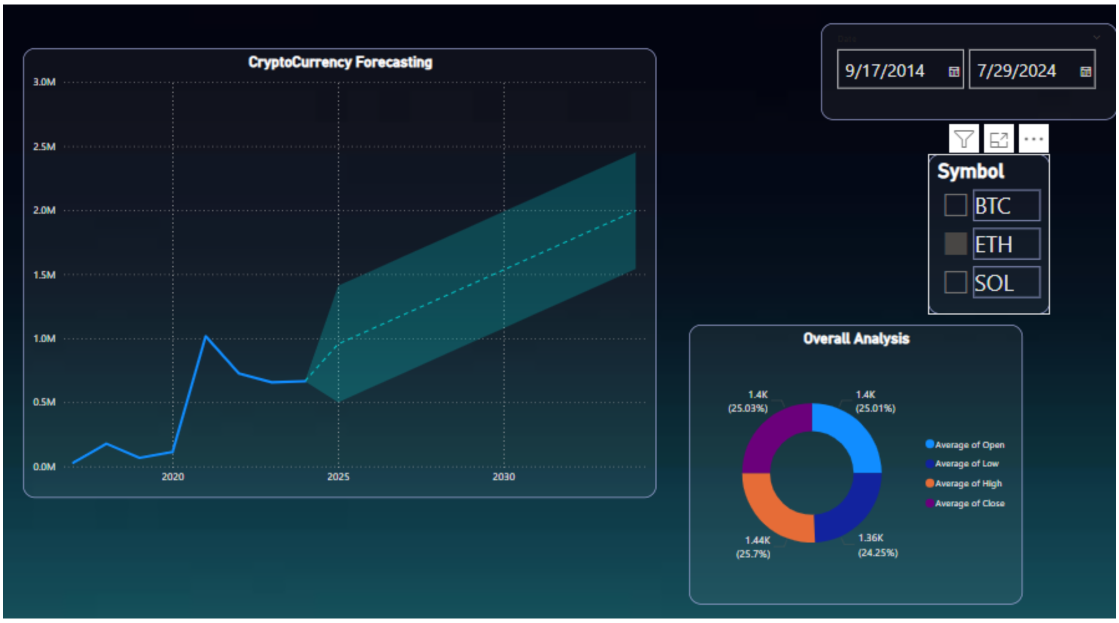
The following dashboard shows the price comparison table for the various cryptocurrencies, namely Bitcoin, Ethereum and Solana. The metrics it measures are the percent change in their price according to various time intervals, the sum of closing prices and the average volume. Bar graph shows the best and worst performance of respective currencies in last 30 and 365 days.

The following dashboard shows the forecasted values of the cryptocurrencies, and the overall analysis shows the average of open, low, high and closing prices.

For the Bitcoin,



For the Ethereum,



For Solana,


CryptoCurrency Forecasting
60K
40K
20K
0K
-20K
2020
2022
2024
2026
2028
2030
2032
2034
9/17/2014
7/29/2024
Symbol
BTC
ETH
SOL
Overall Analysis
58.08... (25....)
57.98... (24....)
60.48... (26.0...)
55.52... (23....)
Average of Open
Average of Low
Average of High
Average of Close

Conclusion

This study has investigated the use of ARIMA, Prophet, and Neural Prophet—three sophisticated time series forecasting models—to anticipate the prices of the three main cryptocurrencies, Solana, Ethereum, and Bitcoin. The inherent non-stationarity of bitcoin price data was efficiently handled with substantial data preprocessing, which included techniques like differencing and logarithmic transformations. This improved the predicted accuracy of the models and ensured their applicability.

While each model has advantages and disadvantages, the comparative study of the ARIMA, Prophet, and Neural Prophet models showed that they are all useful tools for understanding the intricate, non-linear patterns that define cryptocurrency markets. The utilization of the yfinance library played a crucial role in acquiring dependable historical price data, which enabled an extensive examination and sturdy model construction.

According to our research, integrating these models with further developments in data preprocessing and machine learning can enhance the precision and dependability of bitcoin price forecasts.

The project's conclusion emphasizes the value of complex forecasting models and careful data pretreatment in boosting the capacity to predict cryptocurrency values.

Result Analysis

	RSME			Mean			R-squared		
	Bitcoin	*Ethereum*	*Solana*	*Bitcoin*	*Ethereum*	*Solana*	*Bitcoin*	*Ethereum*	*Solana*
ARIMA	0.57065	0.42476	1.42312	10.731	7.8256	4.3458	-1.6148	-0.9962	-2.1837
Neural Prophet	39432.53	2224.571	92.566	48472.62	2609.368	98.6486	-5.11023	-7.56844	-1.5884
Prophet	44845.42	1126.277	70.8144	48566.11	2617.063	99.5023	-6.9175	-1.1994	-0.5128

RMSE:

When compared to the Neural Prophet and Prophet models, the ARIMA model performs better in terms of prediction accuracy because it displays the lowest RMSE values for Bitcoin, Ethereum, and Solana.
The RMSE values of the Prophet and Neural Prophet models are much greater, indicating lower prediction accuracy.

Mean:

In terms of average prediction errors, the ARIMA model performs better than the Neural Prophet and Prophet models, as evidenced by its significantly lower Mean values.

R-squared:

Poor performance is indicated by all models' negative R-squared values for all cryptocurrencies. Models with negative R-squared values are deemed inferior to a horizontal line, which represents the mean of the data.

Summary

The goal of this project is to create and assess prediction models for predicting the prices of the three main cryptocurrencies: Solana, Ethereum, and Bitcoin. Utilizing cutting-edge time series forecasting methods—more especially, ARIMA, Prophet, and Neural Prophet models—is the main goal. The yfinance library was utilized to gather historical pricing data for several cryptocurrencies, resulting in a dependable and effective data retrieval process.

The non-stationary cryptocurrency datasets required a great deal of preprocessing because the ARIMA model performs best with stationary data. The data was stabilized using methods including logarithmic transformation and differencing, which made it possible to use the ARIMA model with accuracy.

Model building, data preprocessing, performance assessment, comparison analysis, insight production, and optimization recommendations are all part of the project's goal. By fulfilling these goals, the project hopes to provide insightful analysis and useful tools for predicting bitcoin prices, assisting researchers, financial analysts, and investors in navigating the erratic cryptocurrency market.

References

[1] Oyedele AA, Ajayi AO, Oyedele LO, Bello SA, Jimoh KO (2023) Performance evaluation of deep learning and boosted trees for cryptocurrency closing price prediction. Expert Systems With Applications 213:119233. https://doi.org/10.1016/j.eswa.2022.119233

[2] Kang CY, Lee CP, Lim KM (2022) Cryptocurrency Price Prediction with Convolutional Neural Network and Stacked Gated Recurrent Unit. Data 7(11):149. https://doi.org/10.3390/data7110149

[3] Kim G, Shin D-H, Choi JG, Lim S (2022) A Deep Learning-Based cryptocurrency price prediction model that uses On-Chain data. IEEE Access 10:56232–56248. https://doi.org/10.1109/access.2022.3177888

[4] Shahbazi Z, Byun Y-C (2021) Improving the cryptocurrency price prediction performance based on reinforcement learning. IEEE Access 9:162651–162659. https://doi.org/10.1109/access.2021.3133937

[5] Tanwar S, Patel NP, Patel SN, Patel JR, Sharma G, Davidson IE, Sharma G, Tanwar S (2021) Deep Learning-Based Cryptocurrency Price Prediction Scheme with Inter-Dependent relations

[6] Chen M, Narwal N, Schultz M, Stanford University (2017) Predicting price changes in Ethereum

[7] Poongodi M, Vijayakumar V, Naveen Chilamkurti (2020) Bitcoin price prediction using ARIMA mode

[8] Dahham AZD, Ibrahim AA (2020) Effects of volatility and trend indicator for improving price prediction of cryptocurrency. IOP Conference Series Materials Science and Engineering 928(3):032043. https://doi.org/10.1088/1757-899x/928/3/032043

[9] Patel J, Kalariya V, Parmar P, Tanwar S, Neeraj Kumar, Mamoun Alazab (2020) Stochastic Neural Networks for Cryptocurrency Price Prediction

[10] Maleki N, Nikoubin A, Rabbani M, Zeinali Y (2020) Bitcoin price prediction based on other cryptocurrencies using machine learning and time series analysis. Scientia Iranica 0(0):0. https://doi.org/10.24200/sci.2020.55034.4040

[11] Salman MK, Ibrahim AA (2020) PRICE PREDICTION OF DIFFERENT CRYPTOCURRENCIES USING TECHNICAL TRADE INDICATORS AND MACHINE LEARNING. IOP Conference Series Materials Science and Engineering 928(3):032007. https://doi.org/10.1088/1757-899x/928/3/032007

[12] Vo A-D (2019) Sentiment analysis of news for effective cryptocurrency price prediction. International Journal of Knowledge Engineering 5(2):47–52. https://doi.org/10.18178/ijke.2019.5.2.116

[13] Valencia F, Gómez-Espinosa A, Valdés-Aguirre B (2019) Price movement prediction of cryptocurrencies using sentiment analysis and machine learning. Entropy 21(6):589. https://doi.org/10.3390/e21060589

[14] Almasri E, Computer Engineering Department, Istanbul University, Arslan E, Computer Engineering Department, Istanbul University (2018) Predicting Cryptocurrencies Prices with Neural Networks

[15] Mittal R, Arora S, Bhatia MPS (2018) AUTOMATED CRYPTOCURRENCIES PRICES PREDICTION USING MACHINE LEARNING. DOAJ (DOAJ: Directory of Open Access Journals). https://doi.org/10.21917/ijsc.2018.0245