Time Series Project

Submitted by: Roy Azuelos 205736879, Amir Steiner 302342498

Technion Israel institute of technology

Part 1 – Data introduction – Bitcoin price

Bitcoin, the first decentralized cryptocurrency, operates on a blockchain-based infrastructure that ensures secure and transparent transactions. Unlike traditional fiat currencies regulated by governments or central banks, Bitcoin's value is determined by market forces, making it highly volatile. Its price behavior reflects global adoption, .investor sentiment, technological advancements, and regulatory shifts
In this analysis, we aim to explore the weekly opening prices of Bitcoin in USD over the past decade (February 2015 to February 2025). Additionally, we integrate an exogenous variable—the cost per computational unit (dollar per hash rate)—to investigate how the cost of mining hardware affects Bitcoin's price. This variable reflects the economics of mining, a critical component of Bitcoin's ecosystem, and serves as an external factor that influences Bitcoin's supply and market dynamics

Dataset overview

The dataset consists of 523 weekly observations of Bitcoin's closing prices (USD) from February 2015 to February 2025. The data is clean, with no missing values, and offers a robust foundation for time series analysis. The high variability in prices highlights Bitcoin's volatile nature, which is crucial to understanding market dynamics. The data was originally with daily observations, but we divided the data into weekly observation

Key statistics:

Mean Price: $21,147
Median Price: $10,167
Standard Deviation: $23,590
Minimum Price: $210
Maximum Price: $106,029

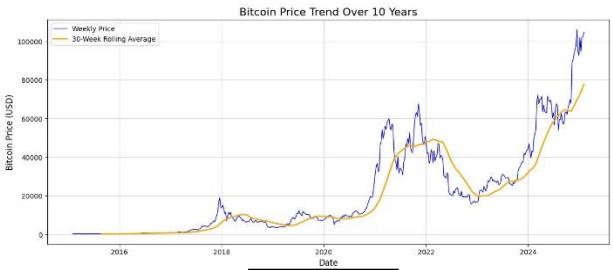


Figure 1

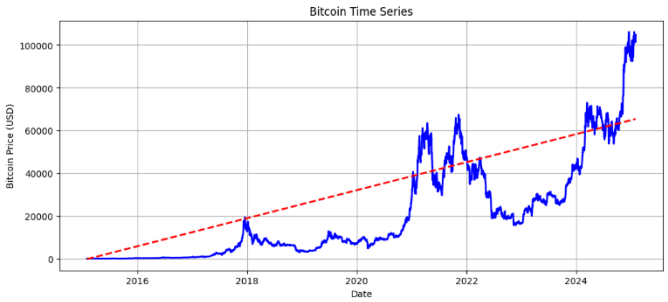


Figure 2

Trend Analysis:

Figures 1 & 2: Bitcoin Price Trend Over 10 Years

The graphs highlight Bitcoin's long-term upward trend, with notable surges in 2017, late 2020, and 2024. Figure 1 emphasizes this trend with a 30-week rolling average (orange line), smoothing out short-term volatility for clearer trend visualization. In Figure 4, the linear trend line (red dashed line) reinforces the consistent growth trajectory, despite significant deviations during speculative peaks and market corrections, such as in 2018

Data exploration:
In order to expand our analysis, we used *statmodel* Python library in order to find the seasonal decomposed trends, as can be seen in figure 3.

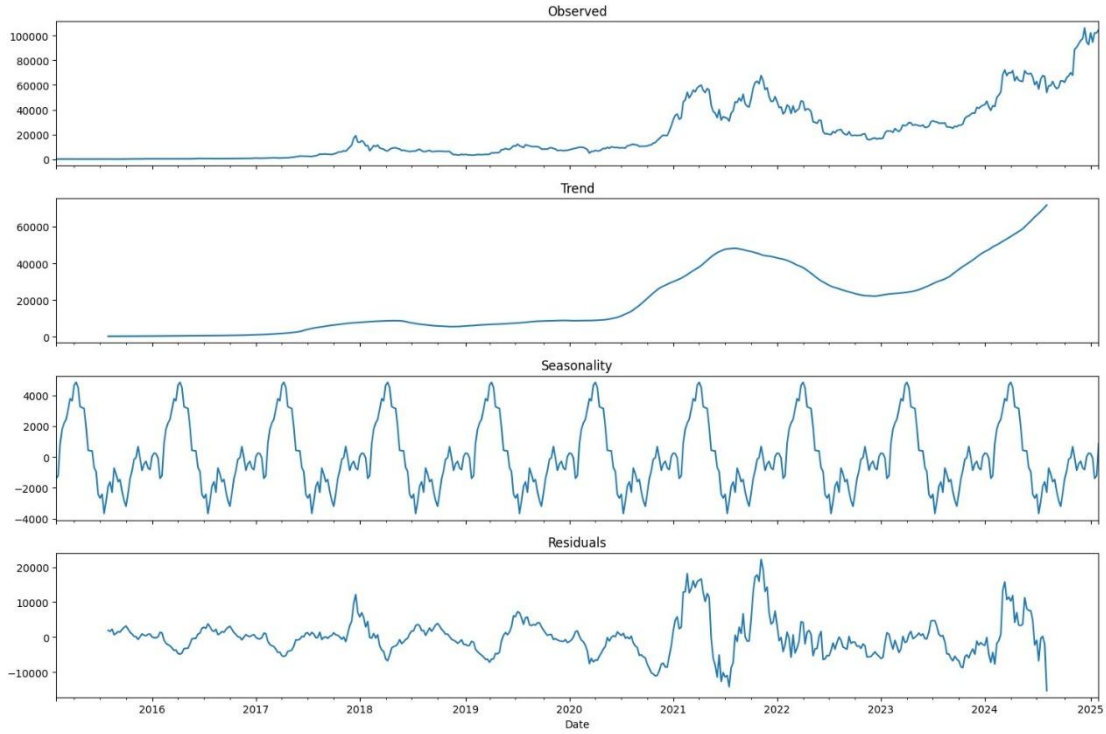


Figure 3: Decomposition of Bitcoin Prices into Trend, Seasonality, and Residuals

Seasonal Decomposition

Observed: The raw data shows high volatility with significant surges in 2017, 2020–2021, .and 2024

Trend: A clear upward trajectory is visible, reflecting long-term market adoption and .value growth

Seasonality: Repeated annual cycles indicate price increases early in the year, with .corrections mid-year, likely tied to market behavior and external factors

Residuals: Significant spikes, particularly in 2017 and 2021, highlight external shocks such as regulatory announcements or speculative surges not explained by the trend or seasonality

Seasonal differences

The graph indicates the overall first, seasonal and combined differences from the original price series:

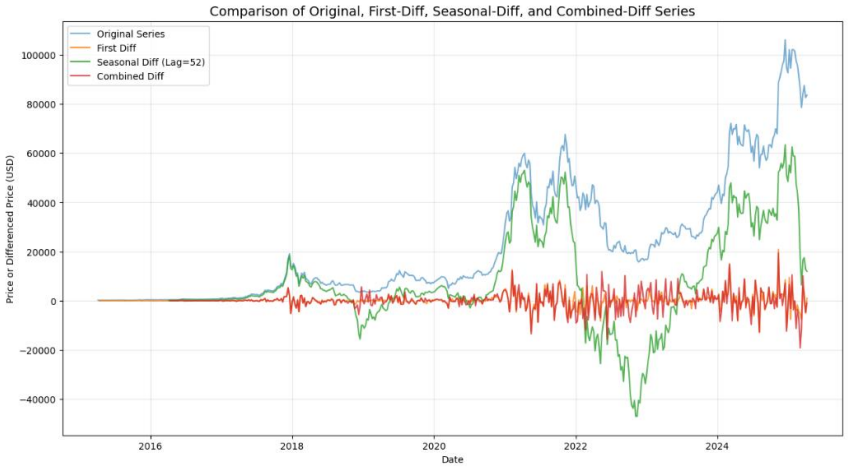


Figure 4: Time difference between original dataset, daily differences, seasonal differences and overall difference

As we can refer from figure 4, there are constant differences

Autocorrelation and Partial Autocorrelation

The graph reveals a clear 52-week seasonal cycle, with recurring annual patterns of price increases early in the year and mid-year corrections. This confirms the need to incorporate seasonality and trend in forecasting models.

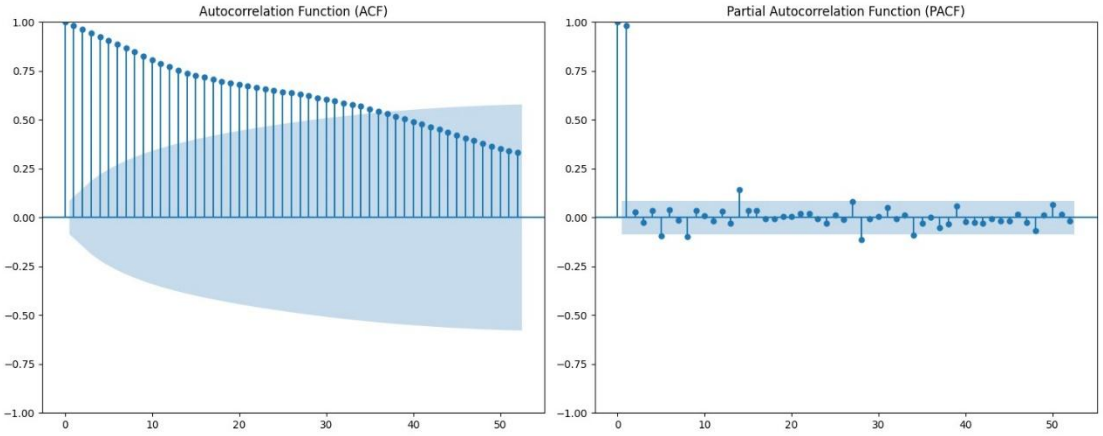


Figure 5: ACF and PACF Plots for original dataset

The ACF plot displays a slow decay, confirming the presence of a strong trend and non-stationarity in the Bitcoin time series. The PACF plot shows significant correlations at .the first lag, indicating that differencing may help stabilize the series

The Augmented Dickey-Fuller (ADF) test results support this conclusion

ADF Statistic: 0.7160

P-Value: 0.9902 ($p \geq 0.05$ indicates non-stationarity)

To address this, applying differencing—such as seasonal differencing ($Y_t - Y_{t-52}$) to achieve stationary

Incorporating an Exogenous Variable

To enrich the analysis, we incorporate dollar per hash rate, a metric representing the cost efficiency of Bitcoin mining. Mining costs are a key economic factor influencing Bitcoin prices, as they impact miner profitability and the supply side of the cryptocurrency market.

Part 2 – Prediction models

Model 1: SARIMAX

We decided to use $((p,d,q)) \times \left((P,D,Q)_{\{s=52\}}\right) = (2,1,1) \times (3,1,0,52)$

In order to find the preferred parameters (p,d,q), $(P,D,Q)_S$, we first needed to make the time series stationary. To achieve this, we applied a **first-order difference**, and tested the result using the Augmented Dickey-Fuller (ADF) test. The test returned an **ADF statistic of -7.2345** and a **p-value of 0**, indicating that the differenced series is likely stationary. Based on this result, we set **d = 1**.

In addition, we removed **seasonality** with a lag of 52 (identified in Part 1 of the analysis), which corresponds to weekly data with yearly seasonality. To account for this, we applied a **seasonal difference** and set **D = 1**.

After ensuring that the series was approximately stationary, we proceeded to fit the SARIMAX model.

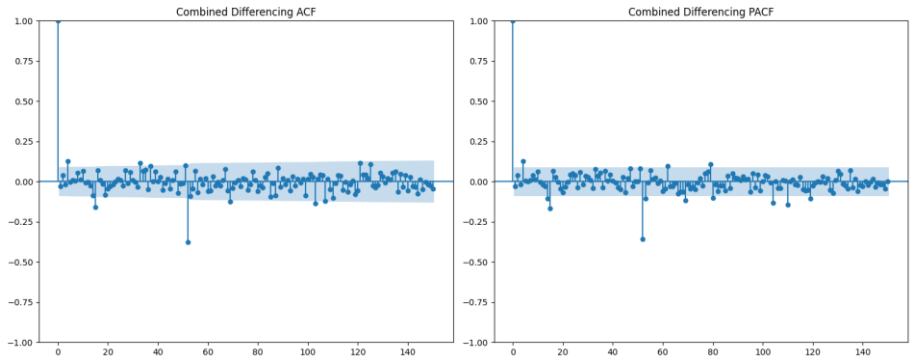


Figure 6: ACF and PACF Plots for combined difference series

- Since we confirmed that the time series was stationary after applying both a first-order difference and a seasonal difference (lag = 52), we examined the **ACF and PACF plots** of the seasonally differenced series to guide our parameter selection. we calculated ACF & PACF in order to check the distribution. Here are the results:

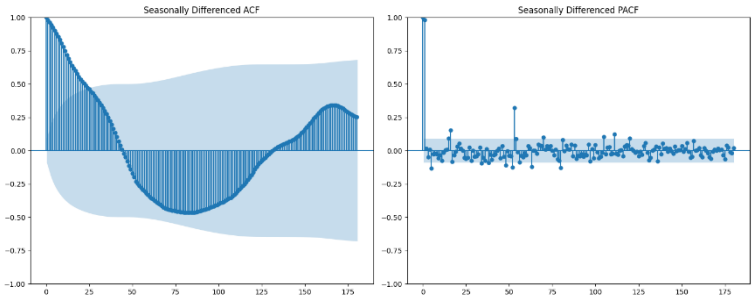
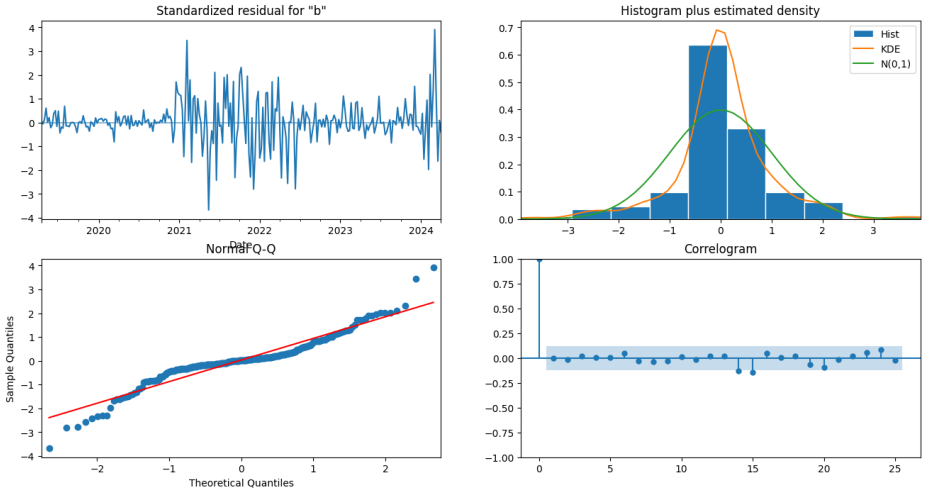


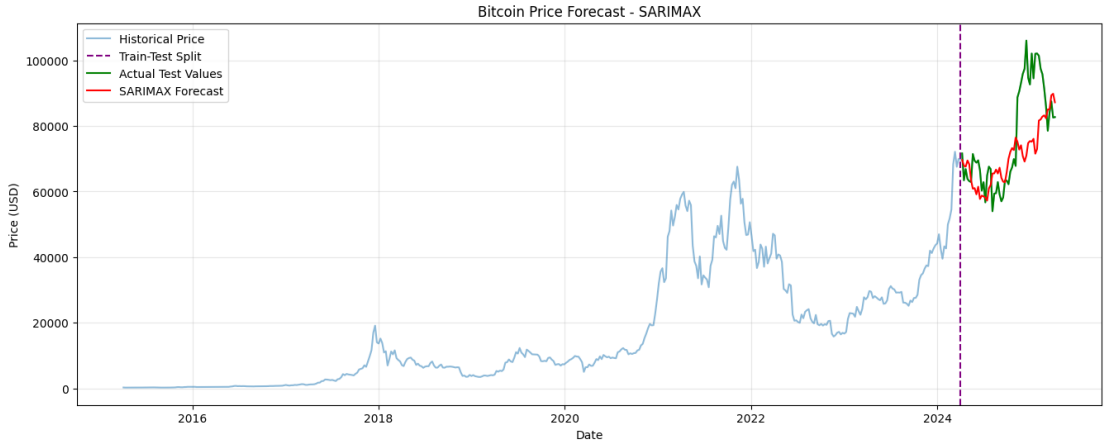
Figure 7: ACF and PACF Plots for seasonal difference series

From the **PACF plot** (right side of Figure 7), we observed two significant spikes at **lags 1 and 2**, indicating an autoregressive component with order **p = 2**. We also noticed a spike at **lag 52**, which reflects the annual seasonality, suggesting a seasonal autoregressive component of **P = 2**. However, after some trial and error, we found that **P = 3** resulted in better prediction performance.

In the **ACF plot** (left side of Figure 7), the wave-shaped pattern without prominent early spikes, along with a gradual decay toward zero, suggests a **moving average order of q = 0 or 1**, and **Q = 0** for the seasonal part. We ultimately chose **q = 1** based on model performance.

SARIMAX Results:





Mean Absolute Error (MAE): 10076.87
Root Mean Squared Error (RMSE): 13237.96

Model 2: Prophet

In the Prophet model for Bitcoin price forecasting, we set changepoint_prior_scale=0.2 to allow moderate trend flexibility and changepoint_range=0.95 to capture changes over most of the historical data. We replaced the built-in weekly seasonality with a custom version for better control. To enhance predictive accuracy, we added two technical indicators—7-period moving average (MA) and Relative Strength Index (RSI)—as external regressors. These features helped the model capture short-term trends and momentum, significantly improving forecasting performance.

We implemented the model in the following python code:

```
# Ensure technical indicators are added to the DataFrame
train_data['ma_7'] = train_data['bitcoin_price'].rolling(window=7, min_periods=1).mean()
train_data['rsi'] = compute_rsi(train_data['bitcoin_price'])

test_data['ma_7'] = test_data['bitcoin_price'].rolling(window=7, min_periods=1).mean()
test_data['rsi'] = compute_rsi(test_data['bitcoin_price'])

# Prophet expects a DataFrame with columns 'ds', 'y' and any regressors
prophet_train = pd.DataFrame({
    'ds': train_data.index,
    'y': train_data['bitcoin_price'].values,
    'ma_7': train_data['ma_7'].values,
    'rsi': train_data['rsi'].values
})
# ---- 5.1) Fit Prophet model ----
prophet_model = Prophet(
    changepoint_prior_scale=0.2,
    changepoint_range=0.95,
    weekly_seasonality=False  # We'll add it manually
)
# Add technical indicators as external regressors
prophet_model.add_regressor('ma_7')
prophet_model.add_regressor('rsi')
# Custom seasonality (avoid duplication)
prophet_model.add_seasonality(name='monthly', period=30.5, fourier_order=5)
prophet_model.add_seasonality(name='yearly', period=365.25, fourier_order=15)
prophet_model.add_seasonality(name='weekly', period=7, fourier_order=10)
prophet_model.fit(prophet_train)
```

The results:

Mean Absolute Error (MAE): 6645.84

Root Mean Squared Error (RMSE): 8344.63

As we can see, even that the MAE is lower for prophet compared to SARIMAX, the overall trend for prophet is more accurate, therefore is the better model in this case.

Model 3: LSTM

We implemented the LSTM neural network in python, after tuning in our hyperparameters:

```python
# LSTModel definition
class LSTModel(nn.Module):
    def __init__(self, input_size=1, hidden_size=32, num_layers=2, output_size=1, dropout_rate=0.3):
        super().__init__()
        self.input_size = input_size
        self.hidden_size = hidden_size
        self.output_size = output_size
        self.num_layers = num_layers
        self.lstm = nn.LSTM(input_size, hidden_size, num_layers,
                            batch_first=True, dropout=dropout_rate)
        self.linear = nn.Linear(hidden_size, output_size)
        self.relu = nn.ReLU()
        self.sigmoid = nn.Sigmoid()
        self.dropout = nn.Dropout(dropout_rate)

    def forward(self, x):
        h0 = torch.zeros(self.num_layers, x.size(0), self.hidden_size).to(device)
        c0 = torch.zeros(self.num_layers, x.size(0), self.hidden_size).to(device)
        out, _ = self.lstm(x, (h0, c0))
        out = out[:, -1, :]
        out = self.dropout(out)
        out = self.sigmoid(out)
        out = self.linear(out)
        return out

# Create model
model = LSTModel(input_size=1, hidden_size=128, num_layers=2).to(device)

# Define loss function and optimizer
criterion = nn.MSELoss()
optimizer = torch.optim.Adam(model.parameters(), lr=0.001)
```

and got the following:

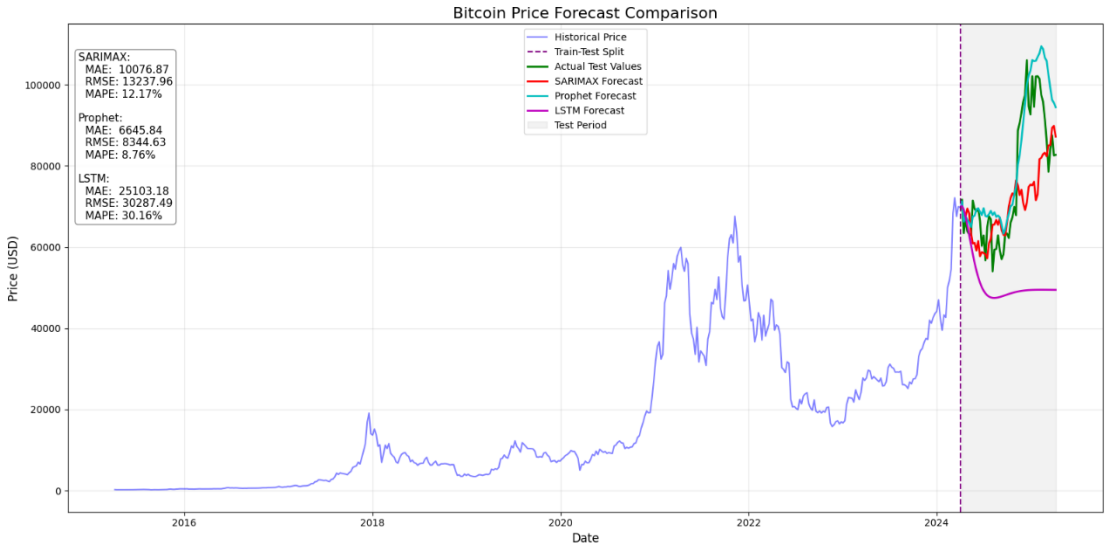
Mean Absolute Error (MAE): 25103.18

Root mean Squared Error (RMSE): 30287.49

As can be seen, the prediction is far less precise compared to the first 2 model we used to predict the results. The MAE is 3 times bigger, indicating that this model is ineffective for price prediction.
We test more architecture CNN-LSTM and attention-based LSTM they improved our MAE/RMSE but failed to capture trends or didn't improve the forecast. We can see the impact of each architecture in the graph below
The results:

Model	MAE	RMSE	Trend
LSTM	25103.18	30287.49	yes
CNN-LSTM	13366.19	18035.99	no
attention- LSTM	35709.72	38802.30	yes

Model Comparison:



Model Performance Comparison and Forecast Evaluation:

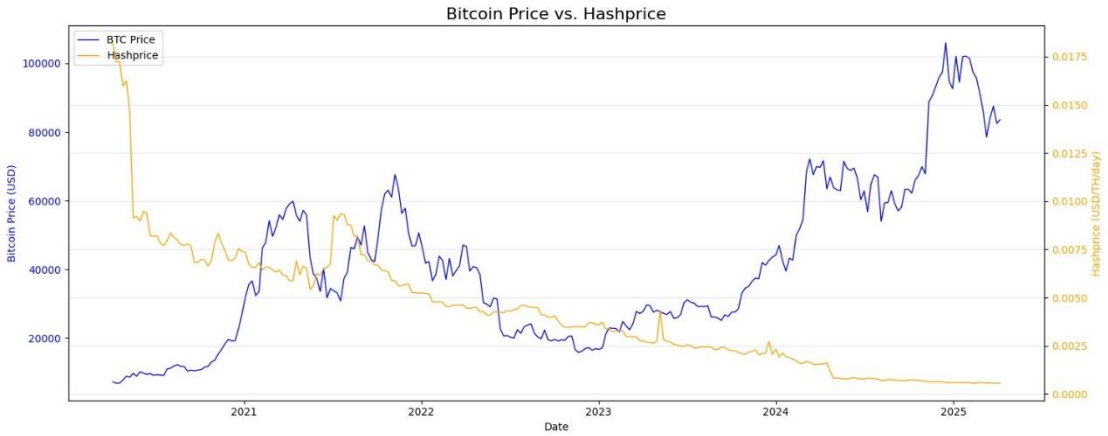
The plot above presents a visual and quantitative comparison between SARIMAX, Prophet, and LSTM models. Among the three, **Prophet** achieved the best overall performance, with the **lowest MAE (6645.84)**, **RMSE (8344.63)**, and **MAPE (8.76%)**, closely aligning with the actual values throughout the forecast period. **SARIMAX** also performed relatively well, especially in capturing general trend patterns, with **MAPE of 12.17%** and errors moderately higher than Prophet. On the other hand, the **LSTM model** produced the **least accurate forecast**, with **significantly higher errors** (MAPE = 30.16%), and showed a tendency to underestimate prices and flatten over time. This indicates that while LSTM may struggle with volatile and rapidly changing data like Bitcoin, classical statistical models (SARIMAX) and hybrid approaches (Prophet) can offer better goodness of fit and robustness in this context.

Part 3 – Exogenous variables

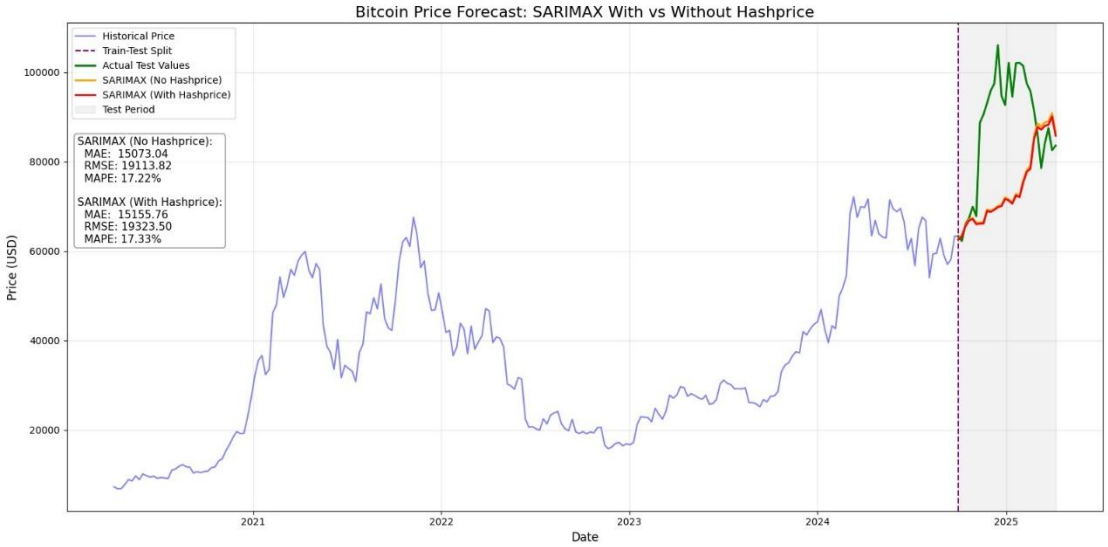
We decided to test two exogenous variables – BTC Hashprice index & US dollar index.

Hashprice index: This variable represent the average value that a bitcoin miner receive on a daily basis. We wanted to research if this index may change the model's behaviour.

Below is a graph that shows the index value over time, compared to Bitcoin price:



When we ran the SARIMAX model with this exogenous variable, and compared it to the original SARIMAX model, we got those results:

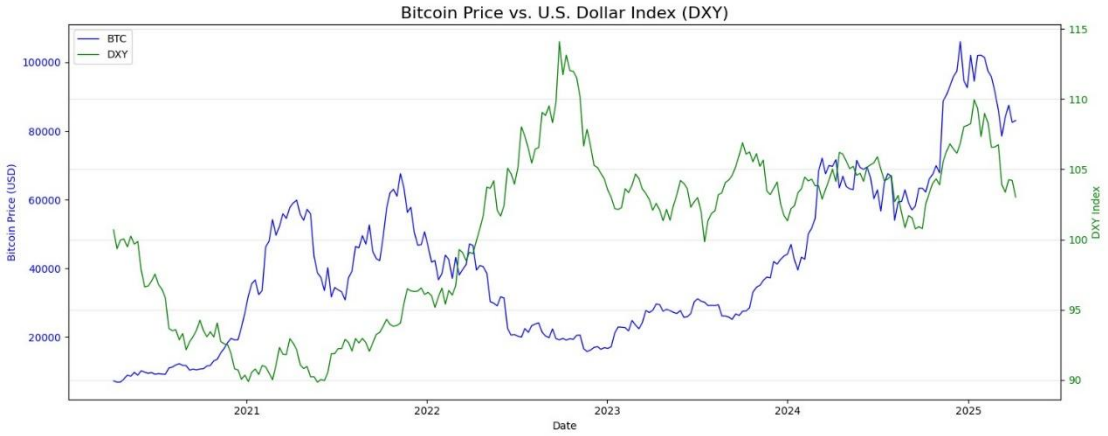


As we can understand from this graph, it seem that there isn't any significant impact on the original results, indicating that this variable has low correlations to Bitcoin prices.

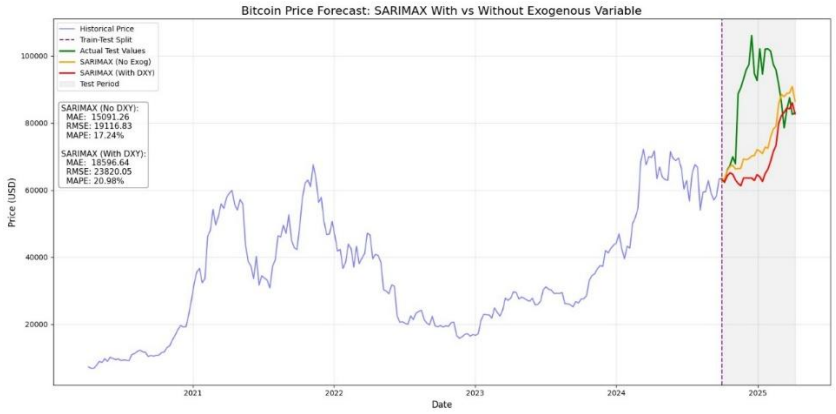
US Dollar index:

The dollar index, often known as the U.S. Dollar Index (USDX), is **a tool for assessing the strength or weakness of the US dollar in relation to a basket of major currencies**. This basket often contains currencies such as the British Pound, Euro, Japanese Yen, Swedish Krona, Swiss Franc, and Canadian Dollar. We decided to test this variable due to the increasing impact of the Bitcoin, and we wanted to check if the traditional currency has an impact on Bitcoin.

Below, there is a comparison between US$ index and Bitcoin price:



As we can see, there is an opposite relation between the two variable – if one is increased, the other decrease, and so on.
When we tested the impact of the index in the SARIMAX model, we got those results:



As we can see, the Dollar index made the model to undervalue its original prediction, indicating that the dollar index has an inversed correlation to bitcoin price.

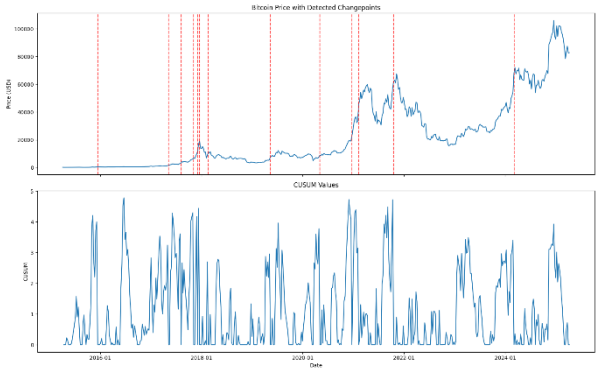
Part 4 – Change point detection

In the previous parts of the project, we tried to predict the future values of Bitcoin using classical and machine learning-based time series forecasting models. However, due to Bitcoin's extreme price volatility and sensitivity to external shocks, the models exhibited reduced accuracy during high-volatility periods. Therefore, we turned to **change point detection** to try identifying significant structural shifts in the distribution of Bitcoin prices over time.

To capture these turning points, we implemented in python two statistical tools: the **CUSUM algorithm**, and the **Shewhart Control Chart**

1. CUSUM Change Point Detection
The algorithm detects significant shifts in the mean of standardized Bitcoin returns. This method is especially effective for identifying gradual, sustained changes in the price-generating process.
Parameter Selection: To balance sensitivity and robustness, we selected a threshold of 5, which ensures that only significant and persistent shifts are flagged, effectively filtering out short-term noise. Additionally, we set the drift parameter to 0.05, allowing the model to tolerate minor fluctuations without falsely triggering a changepoint.

Results: 13 changepoints between 2015 and 2024, many of which align with major external events



some notable occurrences that were made during those dates:

Date	Event
2015-12-14	*Inside Bitcoins Conference & Expo*: Held from December 9-11, 2015, at the Korea International Exhibition Center (KINTEX), this event was one of the largest fintech and Bitcoin/blockchain technology conferences in Asia at the time.
2017-08-7	*Bitcoin's Independence Day*: the Bitcoin network underwent a User Activated Soft Fork (UASF), leading to the activation of Segregated Witness (SegWit).
2020-05-04	Bitcoin underwent its third halving event, reducing the block reward from 12.5 to 6.25 BTC

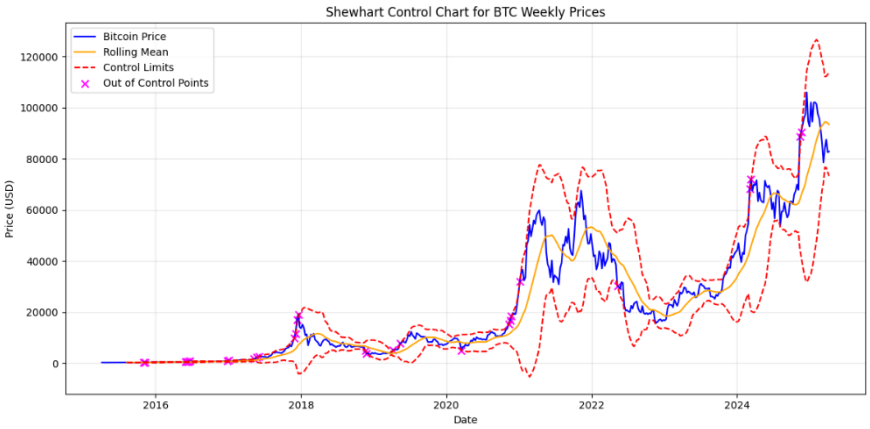
2. Shewhart Control Chart

To complement the CUSUM analysis, we implemented a Shewhart Control Chart on the raw weekly Bitcoin price series. This method identifies changepoints by comparing each data point to control limits based on a rolling mean and standard deviation. It is particularly effective at catching sharp, short-term deviations, such as price spikes and sudden drops.

Parameter Selection: We selected a rolling window of 19 weeks, which captures medium-term trends while smoothing out weekly noise, and set the control limit to 2.55 standard deviations, slightly below the traditional ±3σ, to increase sensitivity to meaningful deviations in a highly volatile asset. This configuration was chosen after testing multiple combinations and provided a balanced detection of significant outliers without excessive false positives.

Results: The Shewhart chart flagged 29 changepoints, many of which occurred in close proximity to those identified by the CUSUM method, reinforcing the validity of these structural shifts. Examples include:

Day	Event
2017-12-04	Peak of the Bitcoin bubble
2020-03-16	Market bottom due to the COVID-19 crash
2024-11-11	First trading week after Trump's re-election



Each X mark represents a change point date along the timeframe, and the control limits represent the upper and lower bound that once crossed, are considered change points.

Conclusion:

In this project, we explored various time series forecasting techniques to model and predict Bitcoin prices. Due to Bitcoin's highly volatile and non-stationary nature, forecasting its future values presented a significant challenge, yet also provided a valuable learning experience.

We evaluated three main models: SARIMAX, Prophet, and LSTM. Among them, Prophet demonstrated the best forecasting performance, achieving the lowest MAE, RMSE, and MAPE, while effectively capturing both trend and seasonality. SARIMAX also performed well after proper differencing and parameter tuning. The LSTM model, although capable of capturing nonlinear patterns, showed limitations in handling Bitcoin's extreme fluctuations and underperformed compared to the classical models.

We further tested more advanced deep learning architectures, such as CNN-LSTM and Attention-based LSTM, which improved MAE and RMSE but still failed to capture the true dynamics of the market. Additionally, We analyzed two exogenous variables - Hash Price Index and the US Dollar Index - to evaluate their contribution to Bitcoin forecasting. Our findings showed that neither variable significantly improved predictive accuracy. While the former showed weak correlation with Bitcoin prices, the latter exhibited an inverse relationship, suggesting a potential macroeconomic influence. While these variables may have theoretical relevance, they did not enhance the models' ability to predict future Bitcoin prices in our case

Lastly, we applied change point detection methods such as CUSUM and the Shewhart Control Chart. The CUSUM method detected major structural shifts around early 2021 and late 2022, which coincided with known market shocks (e.g., post-COVID recovery and macroeconomic tightening). These results demonstrate the importance of using adaptive models that can adjust to sudden changes in market dynamics.

References:

BTC Prices, Hash Price index, US Dollar index