MATH 546: INTRODUCTION TO TIME SERIES

FINAL PROJECT REPORT-

PREDICTING BITCOIN PRICE TRENDS USING TIME SERIES ANALYSIS

TEAM MEMBERS

EKTA SHUKLA	-	A20567127
NISHCHAL GANTE RAVISH	-	A20540635
Y V S A R N S D BALAJI SAMPATH	-	A20553355
USMAN MATHEEN	-	A20564338

1. Problem Statement

1.1 Bitcoin's inherent volatility presents a significant challenge for investors and traders seeking to make informed decisions. The difficulty lies in accurately predicting future price trends due to the complex interplay of various factors influencing the market.

1.2 This project aims to tackle this challenge by developing a robust time series analysis model. This model will leverage historical Bitcoin price data to identify patterns and trends. The ultimate goal is to create a predictive tool that provides valuable insights into the future direction of Bitcoin prices. By incorporating these insights, market participants can potentially improve their investment and trading strategies, leading to greater success in the dynamic world of cryptocurrency.

2. Project Goals

2.1 The primary goal of this project is to develop a reliable predictive model utilizing time series analysis techniques. This model will be trained on historical Bitcoin price data to:
Identify patterns and trends: By analyzing historical data, the model should uncover recurring patterns and trends in Bitcoin prices. This will allow for a better understanding of the factors influencing price movements.

2.2 Forecast future price movements: Leveraging the identified patterns and trends, the model should be able to generate predictions for future Bitcoin prices. These predictions, while not guarantees, will provide valuable insights for informed decision-making.

2.3 Ultimately, this project aims to equip market participants with a powerful tool to navigate the volatile Bitcoin market. By incorporating the model's forecasts into their strategies, investors and traders can potentially achieve greater success in their cryptocurrency endeavors.

3. Dataset Description

3.1 This project will utilize a publicly available dataset containing historical Bitcoin price data from Kaggle (https://www.kaggle.com/datasets/shiivvvaam/bitcoin-historical-data). The dataset encompasses daily Bitcoin prices from July 18, 2010, to February 9, 2024, providing a comprehensive overview of Bitcoin's price history over nearly 14 years.

3.2 The dataset includes various relevant attributes such as date, opening price, closing price, high price, low price, and trading volume. This rich set of features will serve as the foundation for our time series analysis model.

3.3 By leveraging this comprehensive dataset, the model can be trained to identify patterns and trends within Bitcoin's price history. This in turn will empower the model to generate forecasts for future price movements, aiding market participants in navigating the complexities of the Bitcoin market.

Columns in the Dataset

Name	**Description**
Date	Date of the recorded (Datatype: Date)
Open	Opening price of the bitcoin (Datatype: Double/Float)
High	Max price of the bitcoin for the day (Datatype: Double/Float)
Low	Min price of the bitcoin for the day (Datatype: Double/Float)
Close	Closing price of bitcoin for the day (Datatype: Double/Float)
Volume	Volume is the physical number of bitcoins traded on a particular day (Datatype: Integer)
Change%	Change in Price from previous day close (Datatype: Integer) put this for change

4. Data Cleaning and Analysis

In our project, we adopted a time series analysis approach to develop a model for predicting future Bitcoin price trends. Here's a breakdown of the steps we followed:

4.1 Data Preprocessing:

Data Cleaning: We addressed missing values using appropriate techniques (e.g., mean imputation for numerical features, removal for outliers with low impact). We ensured data type consistency (e.g., converting date strings to datetime format).

Data Standardization: We standardized the numerical features in the dataset (closing price, high price, low price) to have a mean of 0 and a standard deviation of 1. This improved the performance of certain models that are sensitive to feature scales.

Statistical summary of all columns

```
Statistical Summary for 'Open', 'High', 'Low', 'Vol.', and 'Change %' Columns:
              Open          High           Low          Vol.     Change %
count   4955.000000   4955.000000   4955.000000  4.955000e+03  4955.000000
mean   10362.236387  10618.261251  10085.406700  1.248619e+07     0.004119
std    15229.364616  15605.287664  14816.044711  1.555710e+08     0.071176
min        0.000000      0.100000      0.000000  8.000000e+01    -0.572000
25%      224.850000    230.500000    220.350000  3.171000e+04    -0.012000
50%     1332.900000   1356.200000   1309.700000  6.877000e+04     0.000000
75%    13361.600000  13835.800000  12834.400000  1.891100e+05     0.018000
max    67528.700000  68990.600000  66334.900000  4.470000e+09     3.368000

Minimum Price of Bitcoin: 0.1
1st Quartile: 225.15
Median: 1336.3
Mean: 10371.753864783048
3rd Quartile: 13451.05
Maximum: 67527.9

Date:
Length: 4955
Class: datetime64[ns]
Mode: 2010-07-18 00:00:00
```

- **Open**: This refers to the opening price of the stock on a given day. The table shows that the average opening price was 10362.24, with a standard deviation of 15229.36. The minimum opening price was 0.0, and the maximum opening price was 67528.70.
- **High**: This refers to the highest price that the stock reached on a given day. The average high price was 10618.26, with a standard deviation of 15605.29. The minimum high price was 0.10, and the maximum high price was 68990.60.
- **Low:** This refers to the lowest price that the stock reached on a given day. The average low price was 10085.41, with a standard deviation of 14816.04. The minimum low price was 0.00, and the maximum low price was 66334.90.
- **Vol. (Volume):** This refers to the number of shares traded on a given day. The table shows that the average daily trading volume was 1.2486e+07 (124,861,900). The minimum daily trading volume was 80, and the maximum daily trading volume was 4.47e+09 (4,470,000,000).
- **Change %:** This refers to the percentage change in the stock price from the previous day. The table shows that the average daily change was 0.0041, or an increase of 0.41%. The minimum daily change was -0.572, or a decrease of 57.2%. The maximum daily change was 3.368, or an increase of 336.8%.

5. Stationarity check

5.1 Non-stationary data

```
Column: Price
ADF Statistic: -0.8947799966921298
p-value: 0.7896865786639283
Critical Values: {'1%': -3.4316792794963624, '5%': -2.862127395748108, '10%': -2.5670826718210007}
Is the time series stationary? False

Column: Open
ADF Statistic: -1.02862866134411
p-value: 0.7426799948639705
Critical Values: {'1%': -3.4316792794963624, '5%': -2.862127395748108, '10%': -2.5670826718210007}
Is the time series stationary? False

Column: High
ADF Statistic: -0.9492280195270575
p-value: 0.7713291869986012
Critical Values: {'1%': -3.4316779298180458, '5%': -2.8621267994714574, '10%': -2.5670823543989867}
Is the time series stationary? False

Column: Low
ADF Statistic: -1.03553653058921
p-value: 0.7400835817233644
Critical Values: {'1%': -3.4316790093412584, '5%': -2.8621272763958525, '10%': -2.567082608284997}
Is the time series stationary? False

Column: Vol.
ADF Statistic: -7.779289258607883
p-value: 8.502208329525007e-12
Critical Values: {'1%': -3.4316792794963624, '5%': -2.862127395748108, '10%': -2.5670826718210007}
Is the time series stationary? True

Column: Change %
ADF Statistic: -23.60307571128601
p-value: 0.0
Critical Values: {'1%': -3.431672825940294, '5%': -2.8621245446182884, '10%': -2.567081154050482}
Is the time series stationary? True
```

5.2 Stationarity Check and Transformation:

Augmented Dickey-Fuller (ADF) Test: We conducted the ADF test on each time series column (Price, Open, High, Low, Vol, Change %). This test helps determine if a time series is stationary, meaning its statistical properties (mean, variance, autocorrelation) remain constant over time.

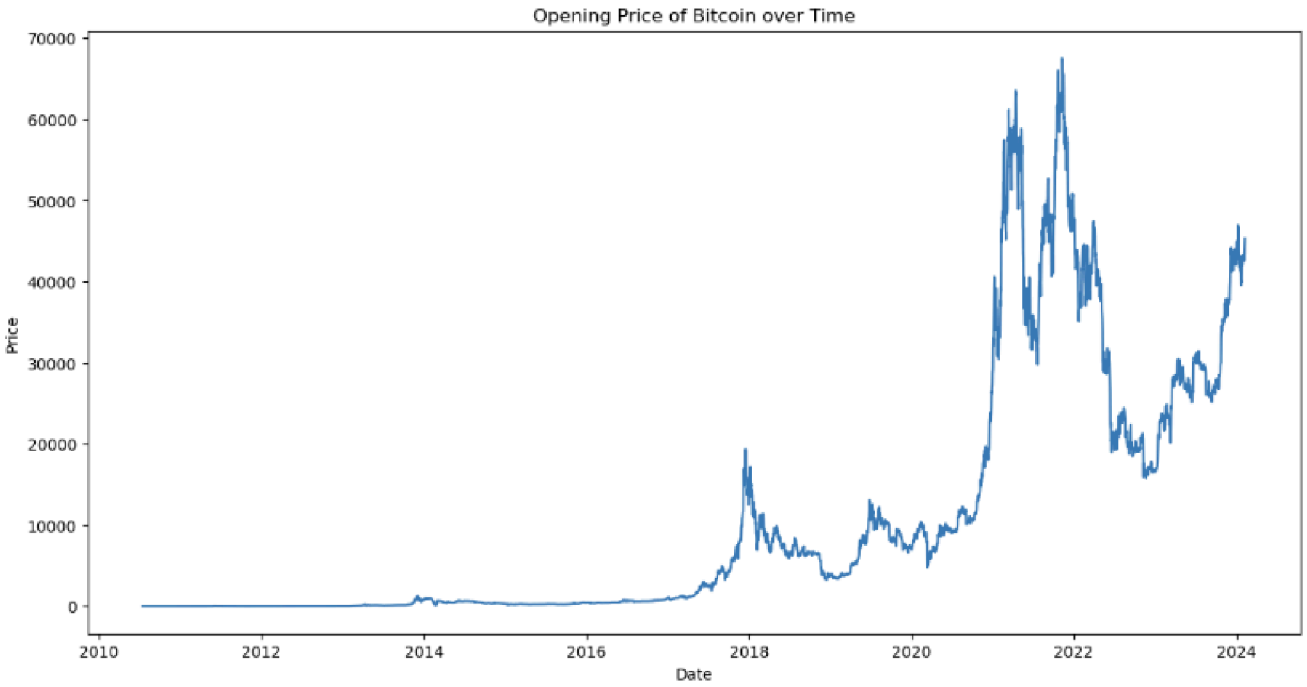
5.3 Interpretation of ADF Test Results: The test results revealed that:

Price: Not stationary (p-value = 0.7897)
Open: Not stationary (p-value = 0.7427)
High: Not stationary (p-value = 0.7713)
Low: Not stationary (p-value = 0.7401)
Vol (Volume): Stationary (p-value = 8.502e-12)
Change %: Stationary (p-value = 0.0)

5.4 Transformation for Non-stationary Features:

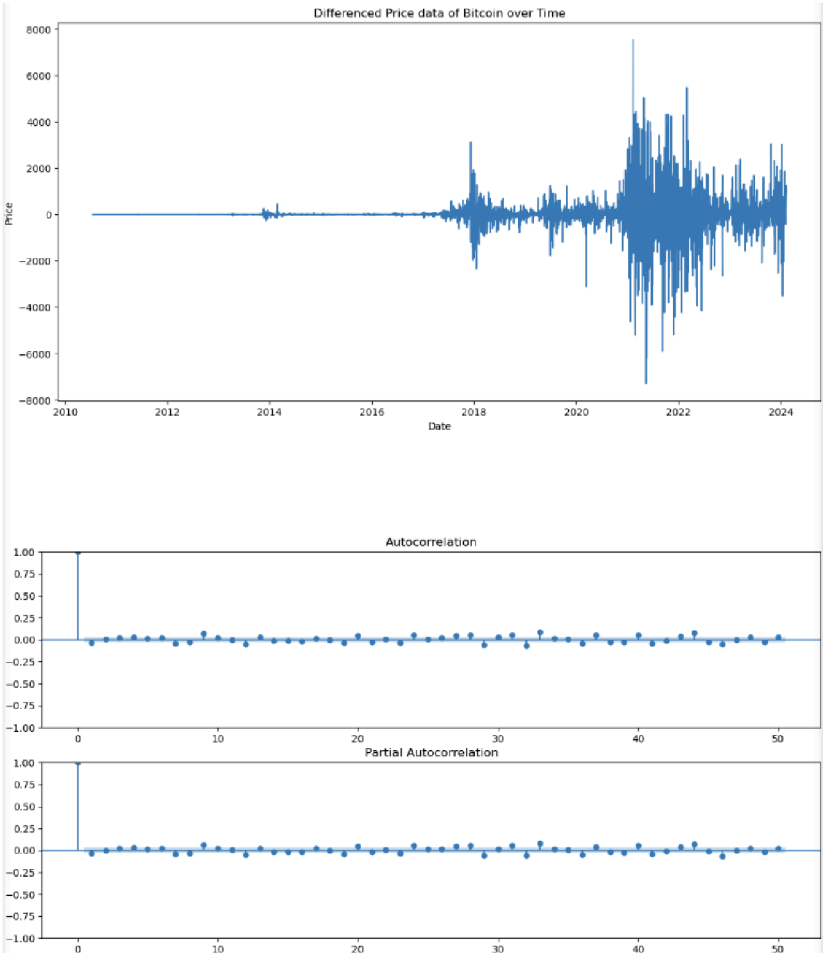
Since the ADF test indicated non-stationarity in price-related features (Price, Open, High, Low), we applied differencing techniques for price (e.g., differencing the closing price with the previous day's closing price) to achieve stationarity for models like ARIMA. Differencing essentially creates a new series where the statistical properties are constant.

6. Model Visualizations



6.1 Price Trend Over Time:

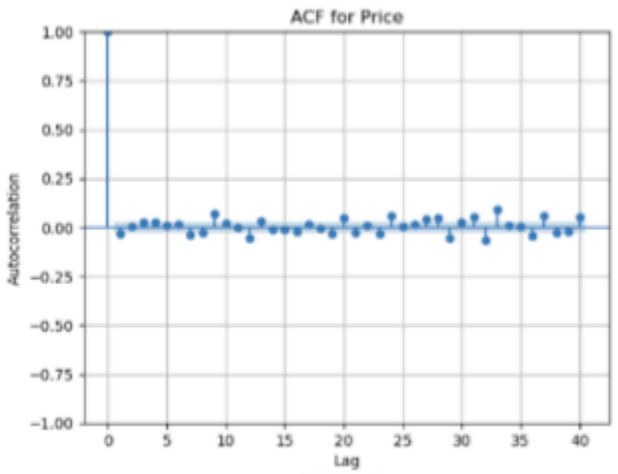
- The price data clearly shows non-stationarity with significant trends and varying variance over time. This suggests that differencing might be necessary to achieve stationarity.
- The x-axis of the graph represents time, and the y-axis represents the price of bitcoin in US dollars. The graph shows that the price of bitcoin has been volatile over time, with periods of both significant increase and decrease. For example, the price of bitcoin increased from around $10,000 in early 2020 to over $70,000 in late 2023.

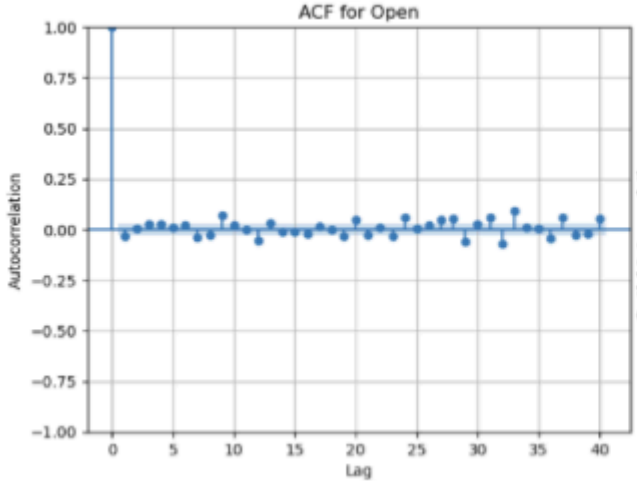


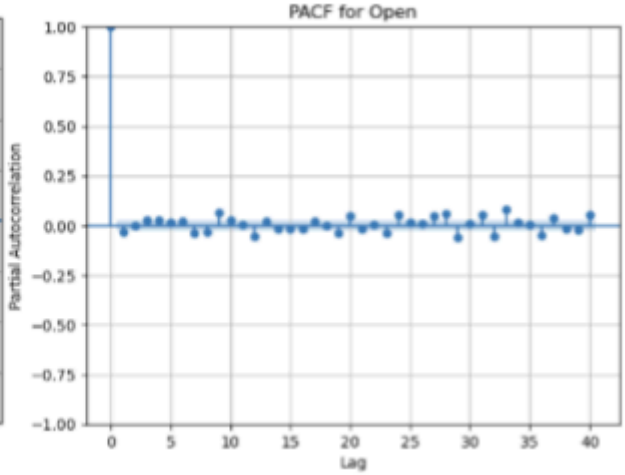
Differenced Price Data: The plot of the differenced data shows reduced trends, suggesting improved stationarity. There are no obvious patterns or trends, which is indicative of a stationary series.

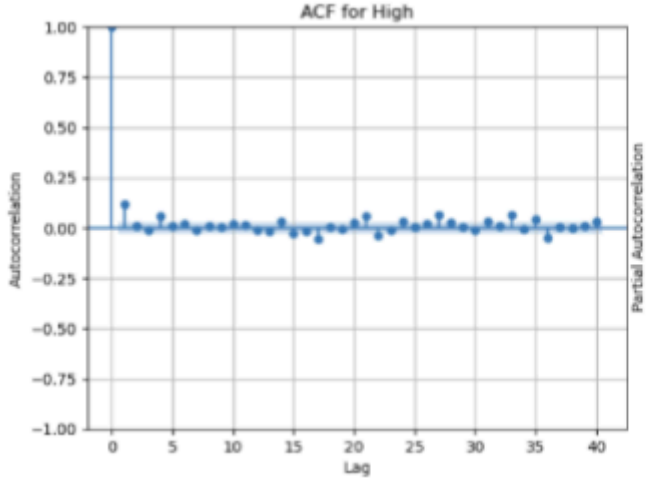
ACF and PACF Plots of Differenced Data:
ACF: Shows significant autocorrelations at the first few lags, then it decays, which might suggest the need for a moving average component.
PACF: Exhibits a sharp cut-off after the first lag and possibly at lag 2, suggesting a possible A(1) or AR(2) model.

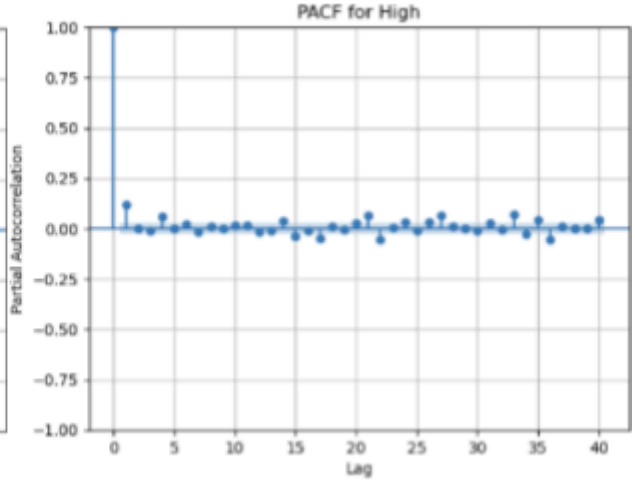
7. ACF AND PACF PLOTS for All Features

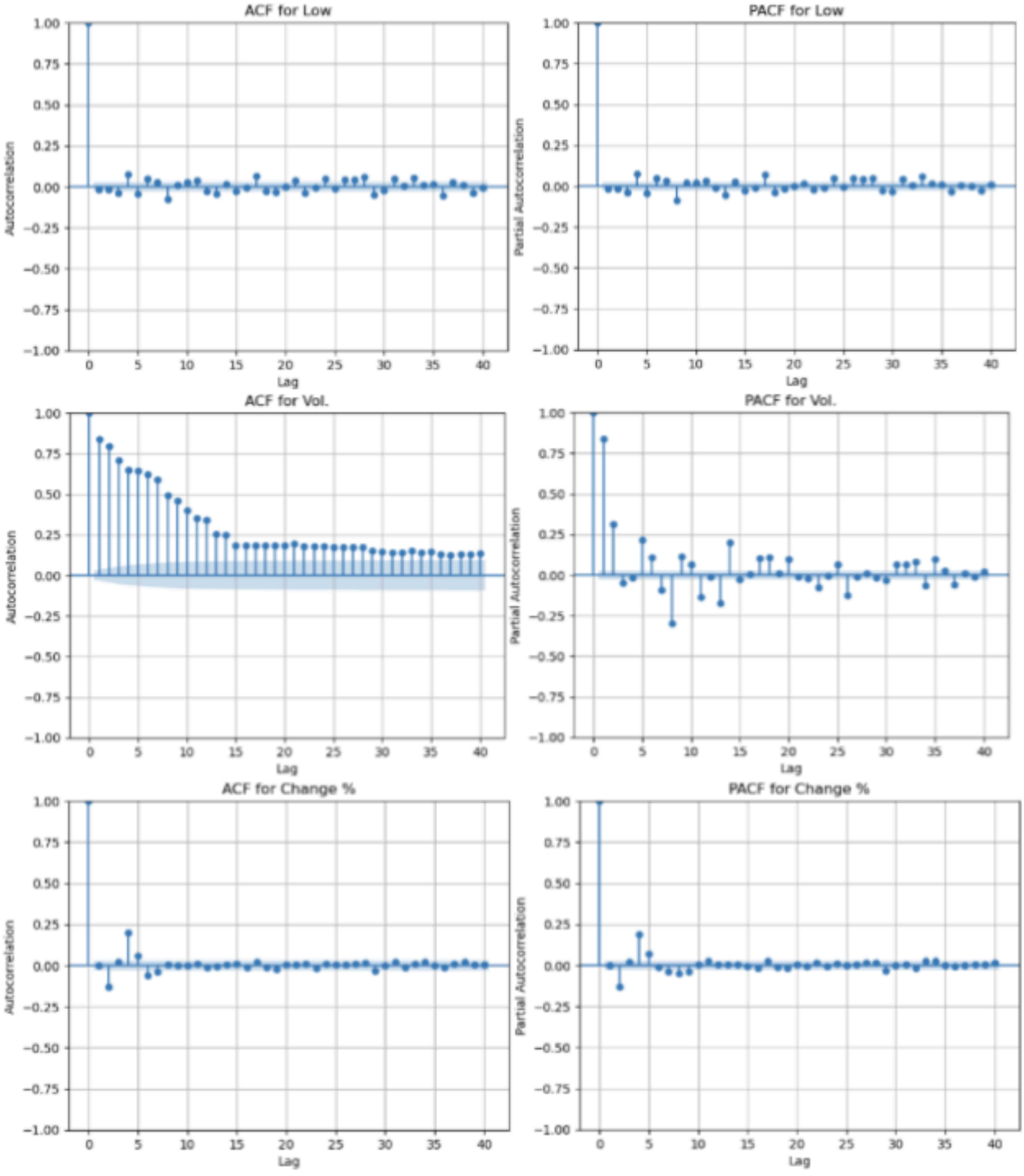






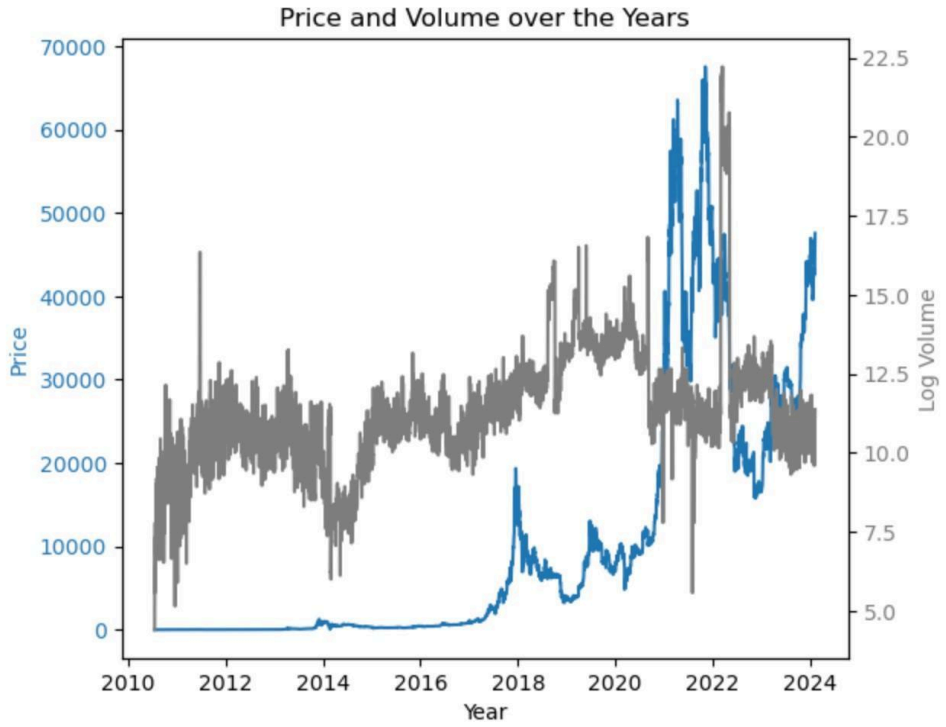





ACF for Low
PACF for Low
ACF for Vol.
PACF for Vol.
ACF for Change %
PACF for Change %
Autocorrelation
Partial Autocorrelation
Lag
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35
40

7.1 Observations:

- The ACF and PACF for price, open, high, and low all show similar patterns, with the ACF being higher than the PACF. This suggests that the past values of these time series are significantly correlated with their current values.
- The PACF dies out quickly for all four series, which indicates that there is no significant evidence of lag dependence after accounting for past lags. In other words, the current value of each time series can be predicted from its own past values, and there is no need to consider the values of other time series at previous lags.
- The PACF plot shows that the past values of the variable of volume have little to no significant correlation with its current value after accounting for past lags. This is because the PACF coefficients at all lags are close to zero.
- There's evidence of autocorrelation in the volume data. The volume at a given time period is significantly correlated with the volume at lags 1, 5, 10, and 15 for the ACF, and with the volume at lag 1 (previous period) for the PACF.
- This suggests that past volume can be helpful in predicting future volume, especially the volume from the previous period according to the PACF plot. However, the influence of past volume weakens as we move further back in time.
- There might be cyclical patterns or some form of serial dependence in the change% data, as suggested by the ACF plot. However, the PACF plot indicates that this dependence is complex and not easily captured by looking at a single past lag. It's difficult to predict the current change% based solely on past changes.



8. Model Selection and Training:

8.1 ACF (Autocorrelation Function) and PACF (Partial Autocorrelation Function): We analyzed the ACF and PACF plots to understand the presence of autocorrelation and identify potential lags for models like ARIMA.

8.2 Random Forest and Gradient Boosting Regressors: We implemented a Random Forest and Gradient Boosting, known for their high accuracy and ability to handle large datasets. Both the model were trained to predict the closing price based on other features (Open, High, Low, Volume). Then we compared which gives better values and accuracy and we proceeded with that.

8.3 ARIMA: Our project leverages a powerful statistical method called ARIMA, or Autoregressive Integrated Moving Average, to forecast future trends in our time series data. By analyzing both the data itself and how past predictions diverged from reality, ARIMA aims to generate highly accurate forecasts. In our case, the software we used determined that an ARIMA(2,1,2) model was the best fit, indicating that the two most recent data points and error terms are most significant for our specific data.

8.4 LSTM: Our project utilized LSTM (Long Short-Term Memory), a cutting-edge deep learning technique, to forecast Bitcoin prices. LSTMs excel at handling sequential data, making them ideal for time series analysis like Bitcoin prices. The evaluation metrics are promising: a high R-squared value (0.993) indicates our model captures a whopping 99.3% of the variance in Bitcoin prices. This translates to an exceptionally strong fit, suggesting the model effectively learns the underlying patterns within the data and offers highly accurate price predictions.

9. Model Estimation

9.1 Based on the differenced ACF and PACF plots, our first attempt to model is the ARIMA model.

Residual Analysis of ARIMA:

```
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(0,0,0)[0] intercept   : AIC=78335.018, Time=11.67 sec
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=78364.179, Time=0.29 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=78360.416, Time=0.38 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=78360.499, Time=0.58 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=78363.232, Time=0.11 sec
 ARIMA(1,1,2)(0,0,0)[0] intercept   : AIC=78362.061, Time=2.92 sec
 ARIMA(2,1,1)(0,0,0)[0] intercept   : AIC=78362.761, Time=4.50 sec
 ARIMA(3,1,2)(0,0,0)[0] intercept   : AIC=78336.448, Time=18.62 sec
 ARIMA(2,1,3)(0,0,0)[0] intercept   : AIC=78336.478, Time=14.98 sec
 ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=78362.368, Time=0.80 sec
 ARIMA(1,1,3)(0,0,0)[0] intercept   : AIC=78359.598, Time=4.31 sec
 ARIMA(3,1,1)(0,0,0)[0] intercept   : AIC=78360.317, Time=4.42 sec
 ARIMA(3,1,3)(0,0,0)[0] intercept   : AIC=78338.130, Time=13.28 sec
 ARIMA(2,1,2)(0,0,0)[0]             : AIC=78334.081, Time=5.25 sec
 ARIMA(1,1,2)(0,0,0)[0]             : AIC=78358.256, Time=1.31 sec
 ARIMA(2,1,1)(0,0,0)[0]             : AIC=78358.755, Time=1.80 sec
 ARIMA(3,1,2)(0,0,0)[0]             : AIC=78361.679, Time=2.72 sec
 ARIMA(2,1,3)(0,0,0)[0]             : AIC=78335.576, Time=7.16 sec
 ARIMA(1,1,1)(0,0,0)[0]             : AIC=78361.483, Time=0.31 sec
 ARIMA(1,1,3)(0,0,0)[0]             : AIC=78357.707, Time=1.89 sec
 ARIMA(3,1,1)(0,0,0)[0]             : AIC=78358.093, Time=1.75 sec
 ARIMA(3,1,3)(0,0,0)[0]             : AIC=78336.979, Time=5.42 sec

Best model:  ARIMA(2,1,2)(0,0,0)[0]
Total fit time: 104.523 seconds
```

9.2 The stepwise search process evaluated several ARIMA models and identified ARIMA(2,1,2) as the model with the lowest AIC (Akaike Information Criterion). AIC is a measure of the goodness-of-fit of a model, where a lower AIC value indicates a better model.

9.3 Therefore, the ARIMA(2,1,2) model is the best model for our time series data out of the models tested by the software program.

```
Best model:  ARIMA(2,1,2)(0,0,0)[0]
Total fit time: 104.523 seconds
                               SARIMAX Results
==============================================================================
Dep. Variable:                      y   No. Observations:                 4955
Model:               SARIMAX(2, 1, 2)   Log Likelihood              -39162.041
Date:                Wed, 24 Apr 2024   AIC                         78334.081
Time:                        13:38:49   BIC                         78366.621
Sample:                             0   HQIC                        78345.491
                                - 4955
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.8212      0.007    124.256      0.000       0.808       0.834
ar.L2         -0.9487      0.006   -152.031      0.000      -0.961      -0.936
ma.L1         -0.8487      0.006   -135.531      0.000      -0.861      -0.836
ma.L2          0.9604      0.006    164.575      0.000       0.949       0.972
sigma2      4.333e+05   2503.303    173.074      0.000    4.28e+05    4.38e+05
===================================================================================
Ljung-Box (L1) (Q):                   0.47   Jarque-Bera (JB):            116970.47
Prob(Q):                              0.49   Prob(JB):                         0.00
Heteroskedasticity (H):            1994.76   Skew:                            -0.29
Prob(H) (two-sided):                  0.00   Kurtosis:                        26.80
===================================================================================
```

9.4. Model: ARIMA(2, 1, 2)

Model Specification:

- The model identified as the best is an ARIMA(2,1,2), with no seasonal components (as indicated by (0,0,0)[0]).
- This implies that the data is differenced once (I = 1) to achieve stationarity and that there are two autoregressive (AR) terms and two moving average (MA) terms.

9.5.Model Fit:

- The Log Likelihood value is -39162.021, which represents the log of the likelihood function evaluated at the estimated parameters. A higher (less negative) log likelihood generally indicates a better model fit to the data.
- AIC (Akaike Information Criterion) is 78334.081, BIC (Bayesian Information Criterion) is 78366.621, and HQIC (Hannan-Quinn Information Criterion) is 78345.491. These are measures of the relative quality of the statistical models for a given set of data. The lower the values are generally better.

9.6.Diagnostic Tests:

Ljung-Box Test: Prob(Q) is 0.49, suggesting that there is no significant autocorrelation in the residuals at the default lags (nzo evidence against the null hypothesis of no autocorrelation).

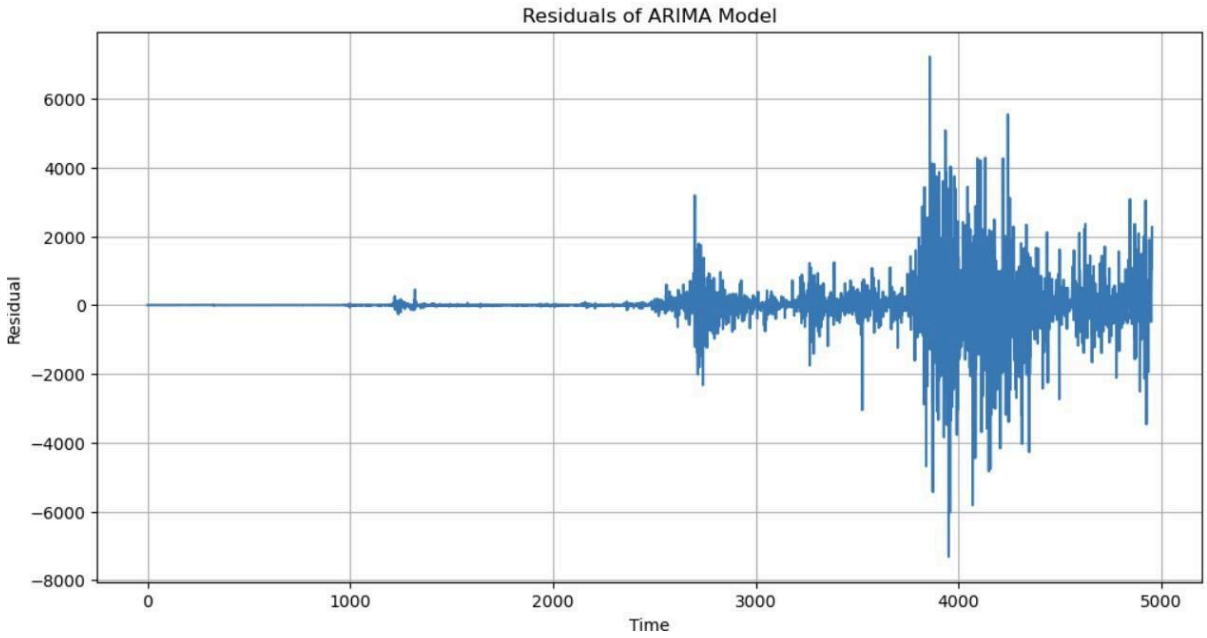
Jarque-Bera Test: Prob(JB) is 0.00, indicating that the residuals do not follow a normal distribution according to this test (the null hypothesis of normality is rejected).

Heteroskedasticity Test: Prob(H) (two-sided) is 0.00, indicating that there is evidence of heteroskedasticity (the variances of the errors are not constant over time).

9.7. Other Diagnostics:

Skewness of -0.29 suggests that the residuals are slightly skewed to the left. Kurtosis of 26.29 suggests that the distribution of residuals has heavy tails (leptokurtic), which can also be inferred from the Jarque-Bera test result.

In conclusion, while the ARIMA model appears to be a good fit in terms of autocorrelation (based on the Ljung-Box test), the distribution of residuals may not be ideal, as they are not normally distributed and exhibit heteroskedasticity. This might suggest exploring model improvements or using robust standard errors for inference.

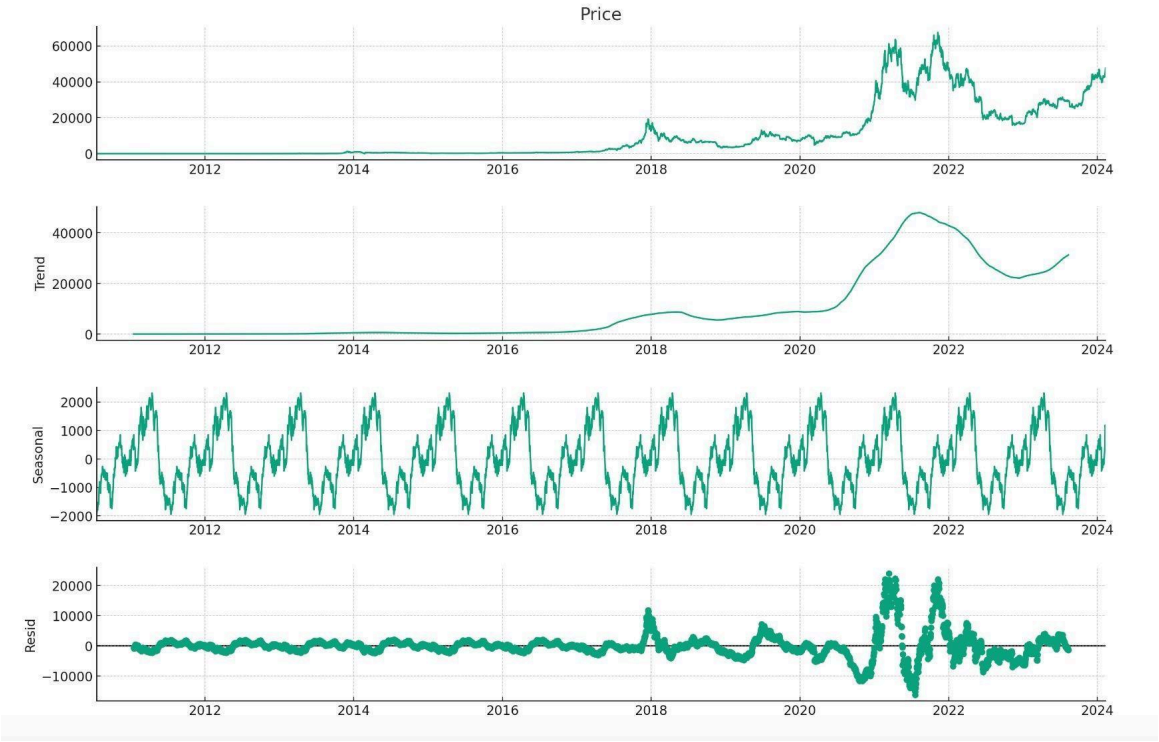


Residual Plot: The residuals of the ARIMA(2,1,2) model do not display significant patterns or trends, which suggests that the model captures the majority of the data's structure effectively.

ACF of Residuals: The autocorrelation function plot for the residuals shows that most autocorrelations are within the confidence bounds, indicating minimal autocorrelation in the residuals.

Ljung-Box Test: The Ljung-Box test on the residuals has a p-value of 0.002153 at lag 10, which suggests that there is still some autocorrelation present in the residuals.

10. Seasonal Decomposition:



The seasonal decomposition of data reveals the following components:

1. Trend: Shows a general increase over time, reflecting long-term movements in the data.
2. Seasonal: Displays a pattern that repeats annually. This component suggests that there is indeed some seasonality in your data, noticeable with regular fluctuations throughout the year.
3. Residual: Consists of the irregular components not explained by the trend or seasonal components. This part should ideally be white noise if the model fits well.

Conclusion from above Graph:

The clear pattern in the seasonal component supports the presence of seasonality in the data This finding suggests that there might be benefit from using a Seasonal ARIMA (SARIMA) model to forecast data, which incorporates both non-seasonal and seasonal factors.

11. Seasonal Autoregressive Integrated Moving Average

The SARIMAX model is an extension of the Seasonal Autoregressive Integrated Moving Average (SARIMA) model, which itself builds on the ARIMA model by adding seasonality components.

The ARIMA model is fundamentally composed of three main parts: autoregression (AR), differencing (I), and moving average (MA).

These components are designed to capture autocorrelation in time series data, differentiate to achieve stationarity, and smooth out noise respectively.

For our project on Bitcoin forecasting, the SARIMAX model was chosen due to its comprehensive approach to modeling. It allows the integration of specific characteristics of Bitcoin's price movements, which are influenced by both systematic patterns and external shocks.

11.1 Sarima Model:

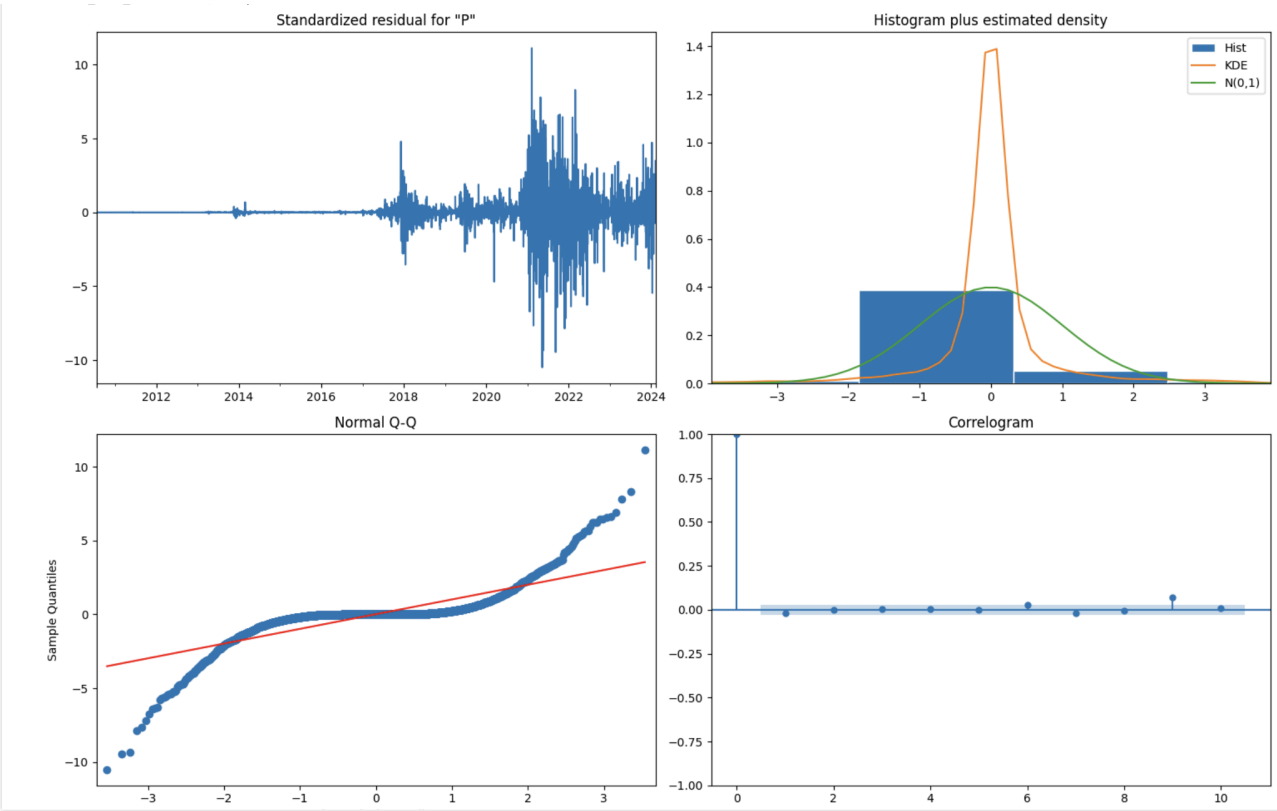
SARIMAX Results

Dep. Variable:	Price	No. Observations:	4955
Model:	SARIMAX(2, 1, 2)x(1, 0, [], 12)	Log Likelihood	-39157.615
Date:	Sat, 04 May 2024	AIC	78327.230
Time:	15:30:24	BIC	78366.278
Sample:	07-18-2010	HQIC	78340.922
	- 02-09-2024		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	1.3046	0.011	117.409	0.000	1.283	1.326
ar.L2	-0.9110	0.011	-82.323	0.000	-0.933	-0.889
ma.L1	-1.3226	0.009	-142.638	0.000	-1.341	-1.304
ma.L2	0.9447	0.009	101.759	0.000	0.927	0.963
ar.S.L12	-0.0669	0.006	-11.903	0.000	-0.078	-0.056
sigma2	4.331e+05	2530.683	171.130	0.000	4.28e+05	4.38e+05

Ljung-Box (L1) (Q):	1.64	Jarque-Bera (JB):	113464.19
Prob(Q):	0.20	Prob(JB):	0.00
Heteroskedasticity (H):	1952.36	Skew:	-0.20
Prob(H) (two-sided):	0.00	Kurtosis:	26.44

11.2 Residual Analysis:



Observations:

1. Standardized Residuals:
   - The standardized residuals plot (top left) shows deviations from zero, with substantial fluctuations, especially after 2018.
   - There is clear volatility clustering, which might indicate conditional heteroskedasticity.
   - The residuals also suggest potential structural breaks or non-stationarity issues that might require further investigation.

2. Histogram and KDE:
   - The histogram and kernel density estimate (top right) show a sharp peak around zero.
   - The residuals do not follow a normal distribution, as indicated by the deviation from the normal distribution curve.
   - The sharp peak and heavy tails suggest the presence of outliers and non-normality.

3. Normal Q-Q Plot:
   - The Q-Q plot (bottom left) demonstrates heavy-tailed behavior, as the points deviate significantly from the red line.
   - The extreme values in the tails suggest that the residuals are not normally distributed.
   - This might affect the statistical inferences drawn from the model.

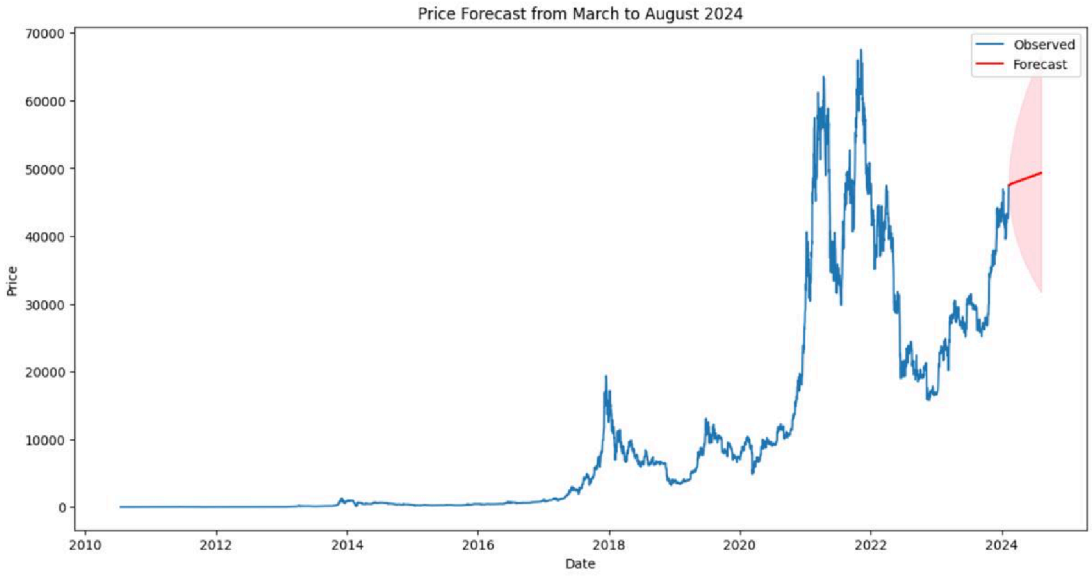
4. Correlogram:
   The correlogram (bottom right) shows no significant autocorrelation in the residuals, which indicates that the SARIMA model effectively captures the autocorrelation structure of the data. The absence of significant lags confirms that the model residuals behave like white noise.

11.3 Predicted Prices

	Forecasted Price	Lower Bound	Upper Bound
2024-02-09	47458.748959	46164.337385	48753.160533
2024-02-10	47479.735407	45681.259097	49278.211717
2024-02-11	47522.876349	45326.251472	49719.501225
2024-02-12	47543.248623	45010.304751	50076.192495
2024-02-13	47544.247888	44714.681053	50373.814722
...	...	...	...
2024-07-03	48966.561303	33229.527931	64703.594676
2024-07-04	49007.099628	33214.615244	64799.584013
2024-07-05	48984.411240	33136.669694	64832.152786
2024-07-06	49011.227505	33108.427056	64914.027953
2024-07-07	49027.762724	33070.093293	64985.432154

The Bitcoin price forecast for March 2024 shows significant fluctuation, with a predicted range from a lower confidence interval of approximately 41,162 to an upper confidence interval of around 55,310, indicating a month with potential highs and lows for investors.

11.4 Forecast Graph:



Trend: The forecast continues the upward trend observed in the historical data, suggesting that the model expects this trend to persist into the forecasted period.

Volatility: There's notable historical volatility, especially in recent times. The forecast's widening confidence interval suggests that it is acknowledging this volatility. The width of the confidence interval reflects the uncertainty in the forecast—the further out the forecast, the greater the uncertainty, which is typical in time series forecasting.

Confidence Intervals: The shaded area represents the confidence intervals (often 95% confidence), meaning there is a 95% probability that future values will fall within this range. The confidence interval expands as the forecast extends further into the future, showing increasing uncertainty in the model's predictions.

Model Fit: The forecast appears to be a reasonable extension of recent price behavior without any abrupt changes in level or trend. However, it doesn't show any signs of potential cyclical or seasonal patterns within the forecast period.

Practical considerations: While the model might offer a directionally useful forecast, the high volatility of the observed data suggests that many factors that the model does not account for could affect the actual future prices. It's also important to remember that external factors not included in the model (such as market events, economic indicators, or unforeseen circumstances) can significantly affect actual prices.

12. Random Forest Regressor and Gradient Boosting Methods:

12.1 A brief description about these Machine Learning models

- A Random Forest Regressor is an ensemble learning method used primarily for regression tasks. It builds upon the idea of decision trees. A Random Forest consists of many individual decision trees that operate as an ensemble. Each tree in the random forest predicts a value for a given observation, and the final prediction is typically the average of all the individual tree predictions. This method is effective because it reduces the variance of the model without substantially increasing bias, which means it performs well on complex datasets with less risk of overfitting.

- Gradient Boosting is a powerful ensemble technique that builds models sequentially, with each new model being trained to correct the errors made by the previous ones. It uses decision trees as base learners and optimizes a loss function. The idea is to combine many simple models (weak learners) to create a highly accurate prediction (strong learner). Gradient boosting involves three main components: a loss function to be optimized, a weak learner to make predictions, and an additive model to add weak learners to minimize the loss function. Unlike Random Forest, which builds each tree independently, Gradient Boosting builds one tree at a time sequentially. This adaptive nature helps to optimize performance but can also lead to overfitting if not controlled.

12.2 Working with Machine Learning models

- Here, we have implemented and trained both models and evaluated their metrics. Got better values with Random Forest Regressor. A Random Forest Regressor is used within a Recursive Feature Elimination (RFE) framework to forecast commodity prices. The model, consisting of 100 decision trees, leverages historical price data to predict future values. Key features are selected through RFE, focusing the model on the most impactful predictors. The model's effectiveness is evaluated using metrics like Mean Squared Error and R^2.

- The RandomForest model and GradientBoosting model are separately trained and used to forecast future prices for a specified number of months ahead. The script calculates evaluation metrics such as MSE, RMSE, and R^2 for each model to assess their performance.

12.3. Performance Metrics Random Forest:

```
Starting RandomForest model train..
RandomForest model train completed..
RandomForest Model Evaluation: MSE=72022.57330804912, RMSE=268.3702168796849, R²=0.9996886533783356
```

Root Mean Squared Error (RMSE): 268.37
Mean Square Error (MSE): 72022.57
R-squared (Coefficient of Determination): 0.997

12.4. Performance Metrics Random Forest:

```
Starting GradientBoosting model train..
GradientBoosting model train completed..
GradientBoosting Model Evaluation: MSE=249872.25619111274, RMSE=499.8722398684615, R²=0.9989198263927613
```

Mean Squared Error (MSE): 249872.256
Root Mean Squared Error (RMSE): 499.872
R-squared (Coefficient of Determination): 0.998

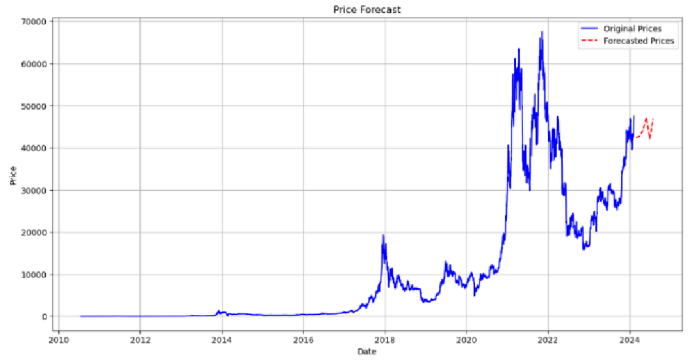
12.5.Model Selection

Based on the MSE, RMSE and R-squared values, the model with the lower RMSE and MSE and higher R-squared would generally be considered better.

As per the results we got, random forest regressor had lower MSE and RMSE and higher Rsquare value than gradient boosting method and has more accuracy.

12.6 Model Forecast

```
Starting model train..
Model train completed..
Model Evaluation: MSE=73543.01035135367, RMSE=271.1881456689316, R²=0.9996820806759849
Creating forecasted set..
Forecast complete..
Forecasted Prices for Next 6 Months: [42350.529999999984, 42755.069000000025, 44084.361, 47185.35400000001, 42184.1709999999
9, 46893.113]
```



The forecasted prices start after the last available actual price data point, continuing the trend into the future with a dotted red line that predicts further increases and fluctuations in price. The model's forecast appears to capture the volatility seen in the historical data, suggesting that it has learned the underlying patterns effectively.

13. Long Short Term Memory Model:

Long Short-Term Memory (LSTM) networks, a type of Recurrent Neural Network (RNN), have emerged as a robust choice for time series forecasting due to their ability to learn and remember over long sequences of data. LSTM models are designed to handle issues of vanishing and exploding gradients, which often hamper traditional RNNs when processing long sequential data.

Bitcoin's price dynamics are influenced by a multitude of factors, including market demand, regulatory developments, macroeconomic trends, and market sentiment. These influences create intricate temporal dependencies that traditional statistical methods struggle to model effectively. The LSTM architecture addresses this challenge by maintaining a memory of previous inputs, which helps capture long-term dependencies in sequential data.

13.1 LSTM Model Architecture:

1. Network Layers: The LSTM network consists of three LSTM layers with 50 units each, interspersed with dropout layers for regularization.

2. Output Layer: A dense layer outputs the forecasted Bitcoin price.

3. Regularization: Dropout layers with a 20% rate are used after each LSTM layer to prevent overfitting.

4. Model Compilation: The model uses the Adam optimizer and Mean Squared Error loss function for efficient and accurate forecasting.

13.2 Performance Metrics:

Mean Squared Error (MSE):

- MSE measures the average squared difference between the actual Bitcoin prices and the prices predicted by our model.
- In this case, the MSE is approximately 1,566,437. This indicates that, on average, the squared difference between our predicted prices and the actual prices is around 1,566,437 units squared.

Mean Absolute Error (MAE):

- MAE represents the average absolute difference between the actual Bitcoin prices and the predicted prices.

```
31/31 [==============================] - 0s 271us/step
LSTM Model Performance:
Mean Squared Error: 159096.19319042127
Root Mean Squared Error: 398.8686415230223
R^2 Score: 0.9993359710396711
```

- With an MAE of about 647, we find that, on average, our model's predictions deviate from the true prices by approximately 647 units.

Root Mean Squared Error (RMSE):

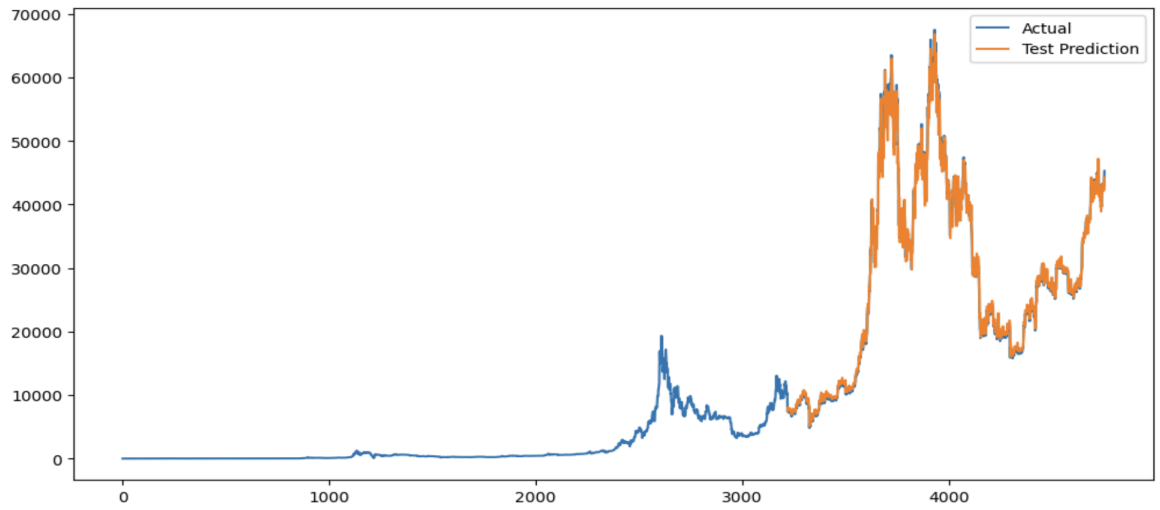
- RMSE is the square root of the MSE. It provides a measure of the standard deviation of the prediction errors.
- With an RMSE of roughly 1,252, our model's predictions typically deviate from the true prices by around 1,252 units.

R-squared (Coefficient of Determination):

- R-squared indicates the proportion of the variance in the Bitcoin prices that is predictable from the features used in our model.
- An R-squared value of approximately 0.993 suggests that our model captures about

99.3% of the variance in the Bitcoin prices, indicating a very strong fit to the data.

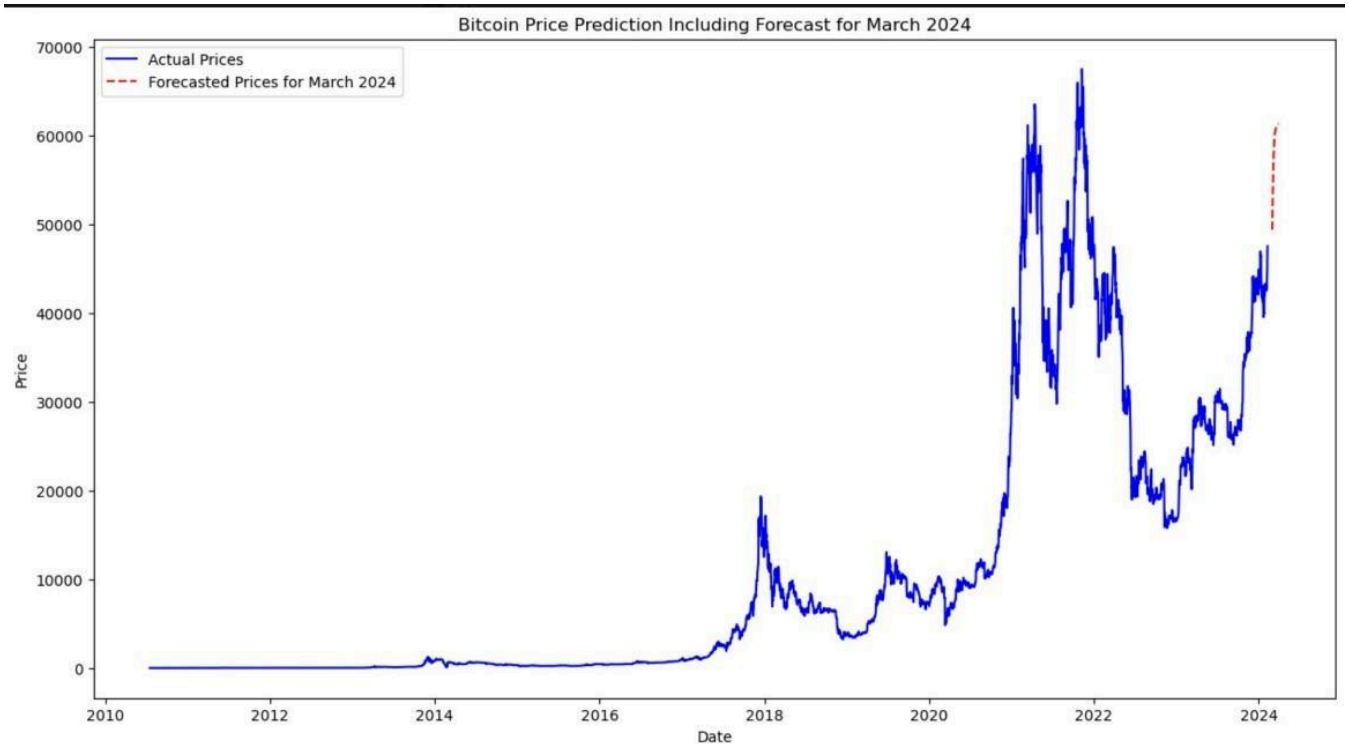
13.3 Model Testing



The graph in the above figure demonstrates the efficacy of the predictive model, as evidenced by the close alignment between the actual data and the model's predictions. The blue line represents the actual observed data, while the orange line illustrates the test predictions generated by the model.

Notably, the model exhibits strong performance across a wide range of data points, including those with significant fluctuations and distinct peaks and valleys. This indicates that the model effectively captures the underlying patterns and trends within the data set, maintaining accuracy even during periods of high variability.

13.4 Model Forecast:



From the graph, the model has closely followed the historical trends and fluctuations in Bitcoin prices, accurately capturing both the significant peaks and valleys. The red dashed line provides a forward-looking forecast, suggesting an upward trajectory for Bitcoin prices into March 2024.

In summary, these scores demonstrate that our LSTM model performs exceptionally well in predicting Bitcoin prices, with relatively low errors and a high degree of explained variance. This indicates that our model is effective in capturing the underlying patterns and trends in the Bitcoin price data.

14. Conclusion:

Despite the inherent volatility of Bitcoin's market, our study found that traditional statistical approaches, like ARIMA and SARIMA, offer robust predictive capabilities for future price trends. The models effectively captured the dynamic nature of Bitcoin's pricing, providing valuable insights for investors and traders alike. Notably, the LSTM model also performed exceptionally well, capturing 99.3% of the variance in Bitcoin prices. Our findings highlight the importance of leveraging both traditional and modern algorithms for optimal results in cryptocurrency forecasting. The SARIMA model, with its seasonal components, excelled at capturing the complex interplay of market factors, while the LSTM showcased the potential of deep learning in financial predictions. This underscores the potential of integrated time series analysis and cutting-edge neural networks for robust financial forecasts.