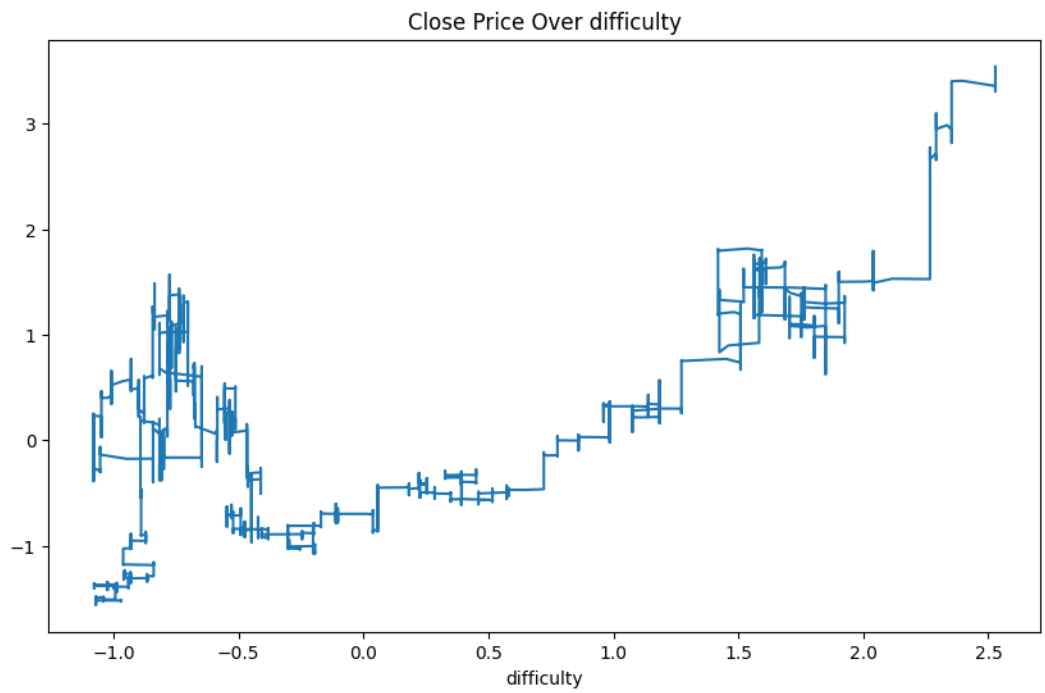
Summary of correlation for each variable with the target variable close_price

Correlation between Variables				
Variable Name	**Target Variable**	**Correlation**	**Relationship**	**Action**
time	close_price	0.566715	Indicates a time-related trend	Retain
difficulty	close_price	0.64	Moderate correlation; significant influence	Retain
estimated_leverage_ratio	close_price	0.158837	Very weak correlation	Drop
addresses_count_active	close_price	-0.000141	no linear relationship	Drop
addresses_count_sender	close_price	-0.1029	weak positive correlation	Drop
addresses_count_receiver	close_price	-0.140818	Very weak correlation	Drop
exchange_whale_ratio	close_price	-0.144744	Very weak correlation	Drop
coinbase_premium_gap	close_price	-0.149693	Very weak correlation	Drop
coinbase_premium_index	close_price	-0.020911	Very weak correlation; likely no meaningful relationship	Drop
coinbase_premium_gap_usdt_adjusted	close_price	0.004622	Insignificant correlation	Drop
coinbase_premium_index_usdt_adjusted	close_price	0.001033	Insignificant correlation	Drop
taker_buy_volume	close_price	0.297314	Weak correlation; test for predictive importance	Retain
taker_sell_volume	close_price	0.300727	Weak correlation; test for predictive importance	Retain
taker_buy_ratio	close_price	-0.038593	Very weak correlation	Drop
taker_sell_ratio	close_price	-0.043311	Very weak correlation	Drop
taker_buy_sell_ratio	close_price	-0.043311	Very weak correlation	Drop
blockreward	close_price	0.309306	Weak correlation; test for predictive importance	Retain
blockreward_usd	close_price	0.561912	Moderately strong correlation; significant influence	Retain
fees_transaction_mean	close_price	-0.101973	Weak correlation	Drop
fees_transaction_mean_usd	close_price	-0.166919	Very weak correlation	Drop
fees_transaction_median	close_price	-0.134659	Very weak correlation	Drop
fees_transaction_median_usd	close_price	-0.177226	Very weak correlation	Drop
miner_supply_ratio	close_price	0.712875	Moderately strong correlation	Retain
addresses_count_inflow	close_price	-0.218503	Weak correlation	Drop
addresses_count_outflow	close_price	0.118641	Weak correlation	Drop
exchange_supply_ratio	close_price	-0.336219	Weak correlation; potential relevance	Retain
transactions_count_inflow_1	close_price	0.026446	Retain???	Drop
transactions_count_outflow_1	close_price	0.287564	Weak correlation; might contribute to the model	Retain
tokens_transferred_total	close_price	-0.038672	Very weak correlation	Drop
tokens_transferred_mean	close_price	-0.056378	Very weak correlation	Drop
tokens_transferred_median	close_price	0.551104	Moderate correlation; potential value	Retain
transactions_count_inflow_2	close_price	-0.044583	Very weak correlation; no meaningful relationship	Drop
transactions_count_outflow_2	close_price	-0.068607	Very weak correlation; no meaningful relationship	Drop
long_liquidations	close_price	-0.044583	Very weak correlation	Drop
short_liquidations	close_price	-0.068607	Very weak correlation	Drop
long_liquidations_usd	close_price	0.023953	Very weak correlation	Drop
short_liquidations_usd	close_price	0.045097	Very weak correlation	Drop
open_price	close_price	0.999906	Redundant with close_price	Retain
high_price	close_price	0.999947	Redundant with close_price	Drop
low_price	close_price	0.999936	Redundant with close_price	Drop
close_price	close_price	1	Target variable	Retain
volume	close_price	0.046385	Very weak correlation	Drop
open_interest	close_price	0.842432	Strong correlation; significant influence	Retain

Single Variable Comparison (close_price)

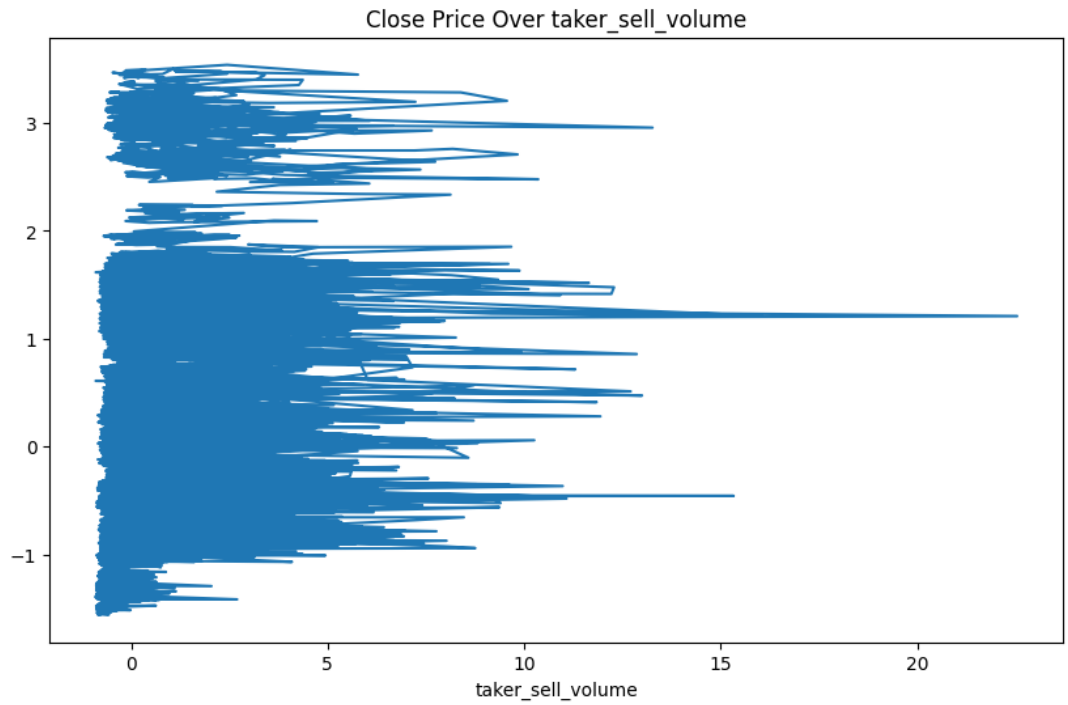
Time

Difficult

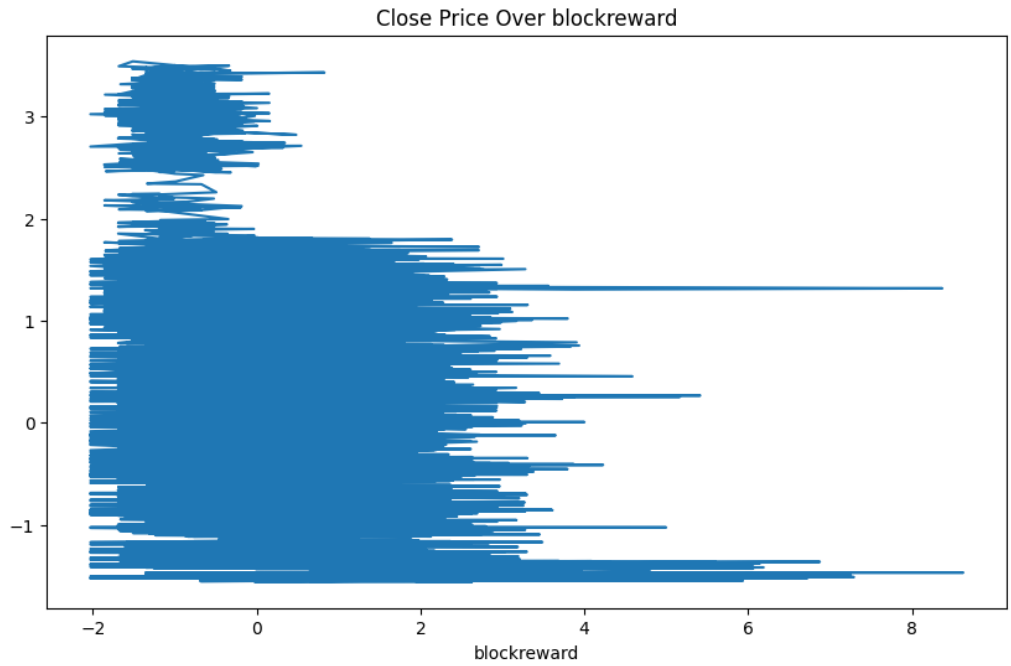


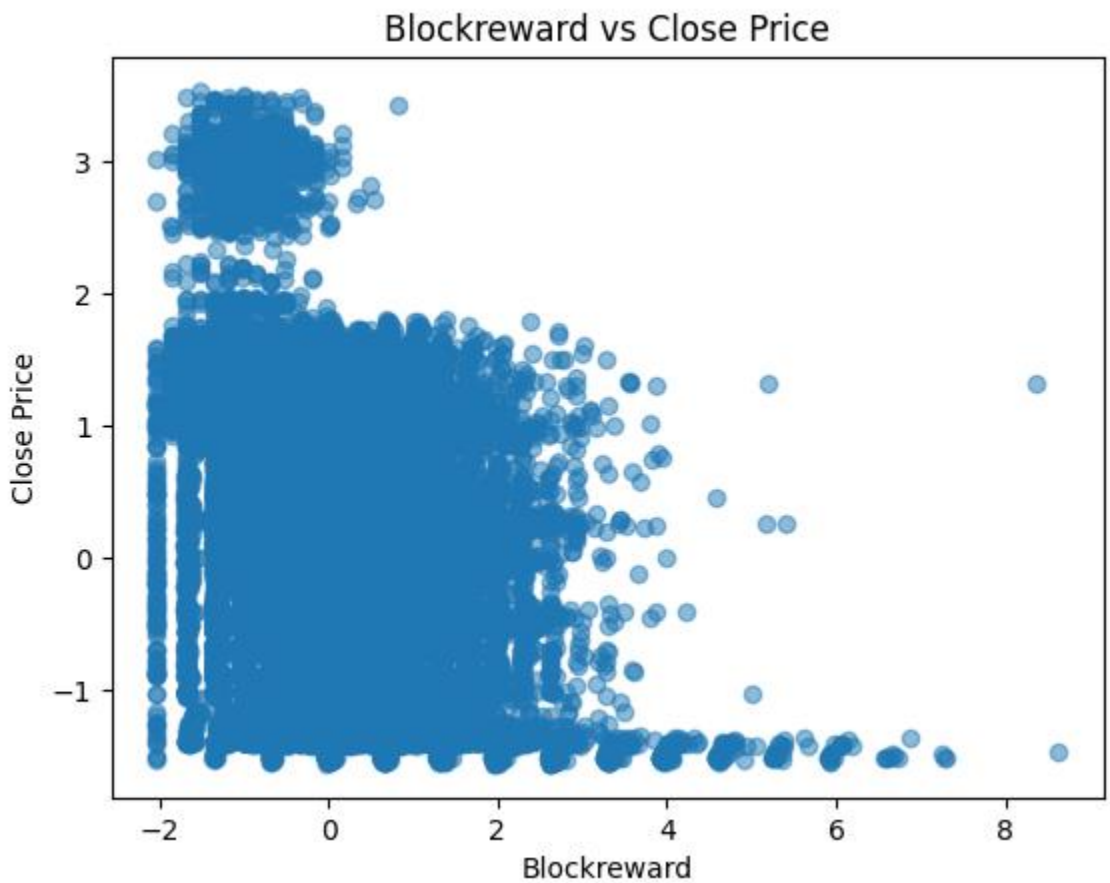
taker_buy_volume

taker_sell_volume

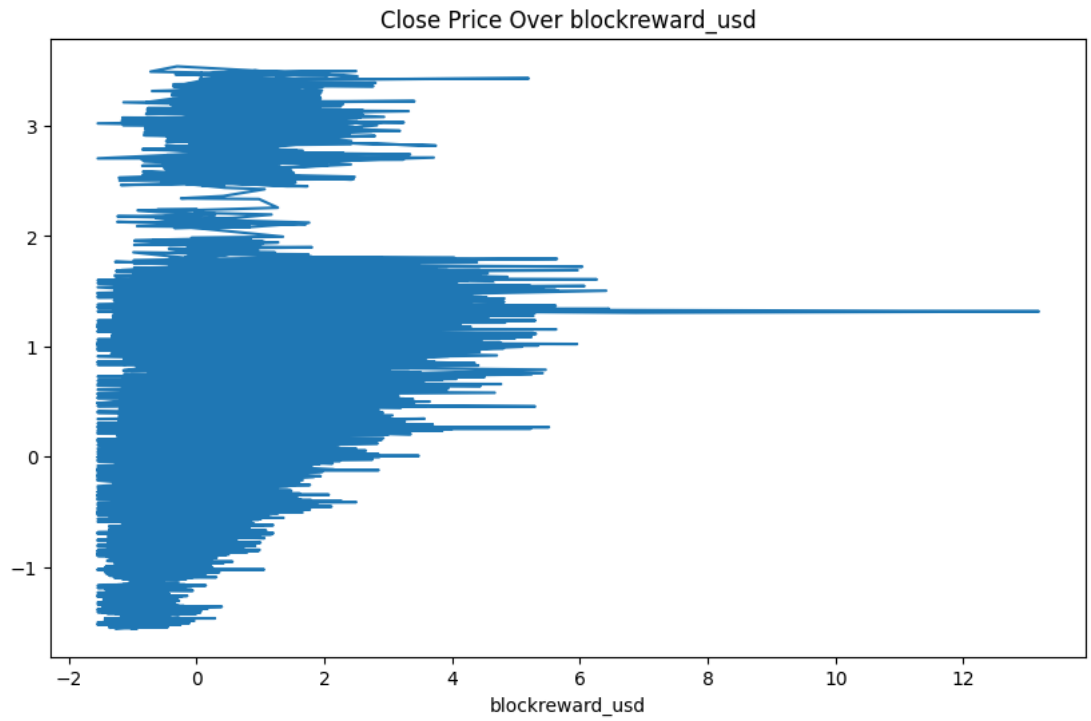


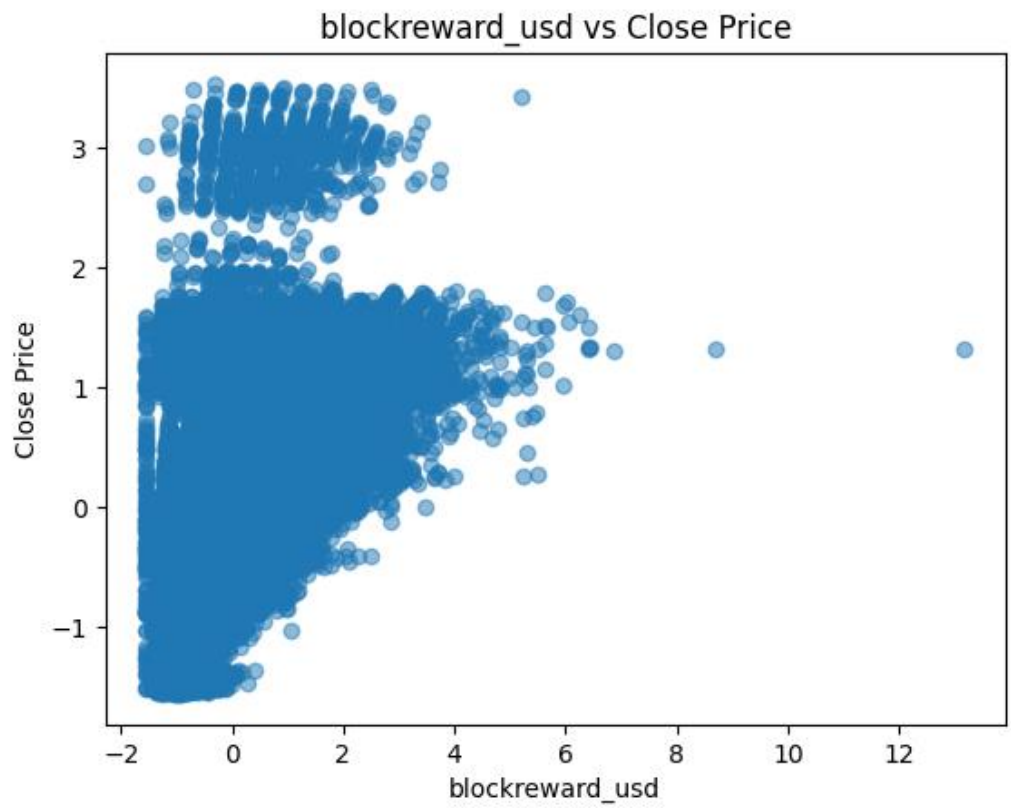
Blockreward



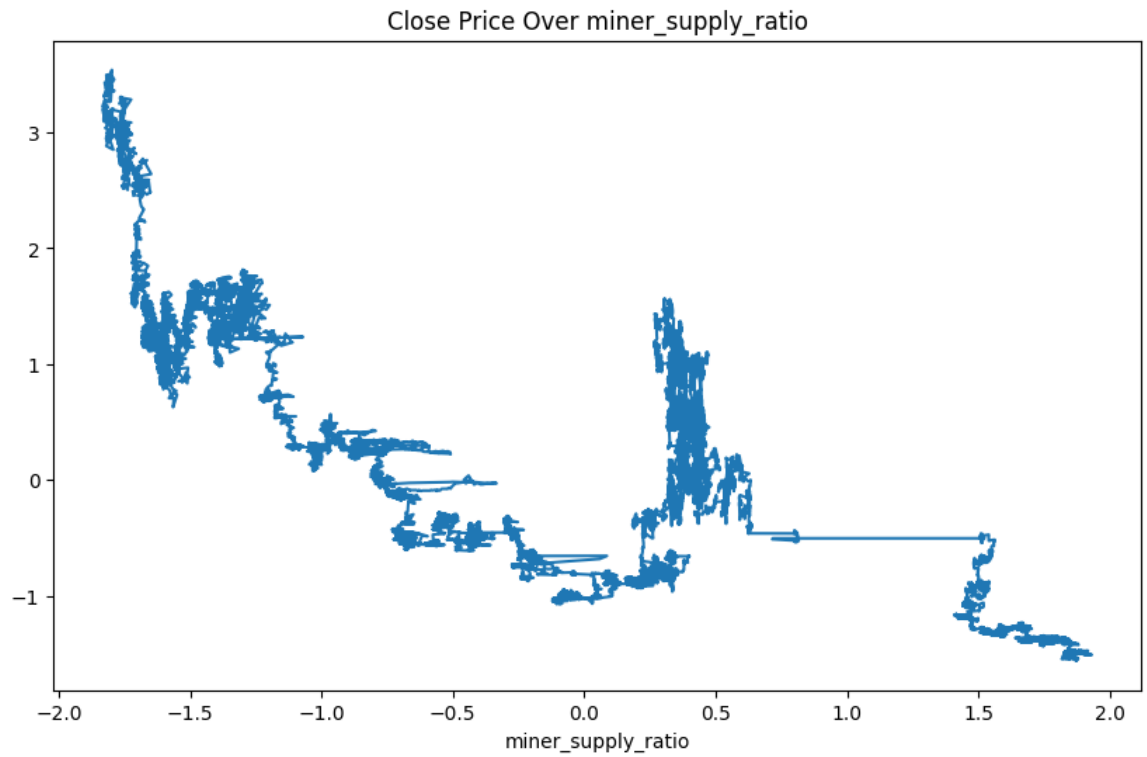


blockreward_usd

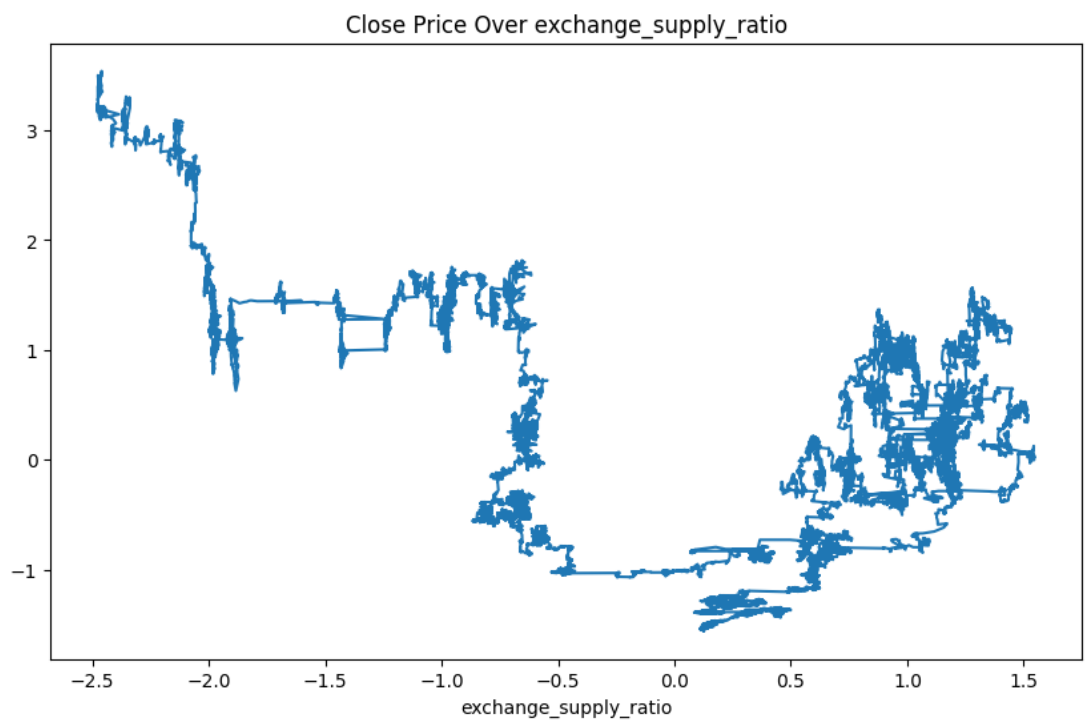




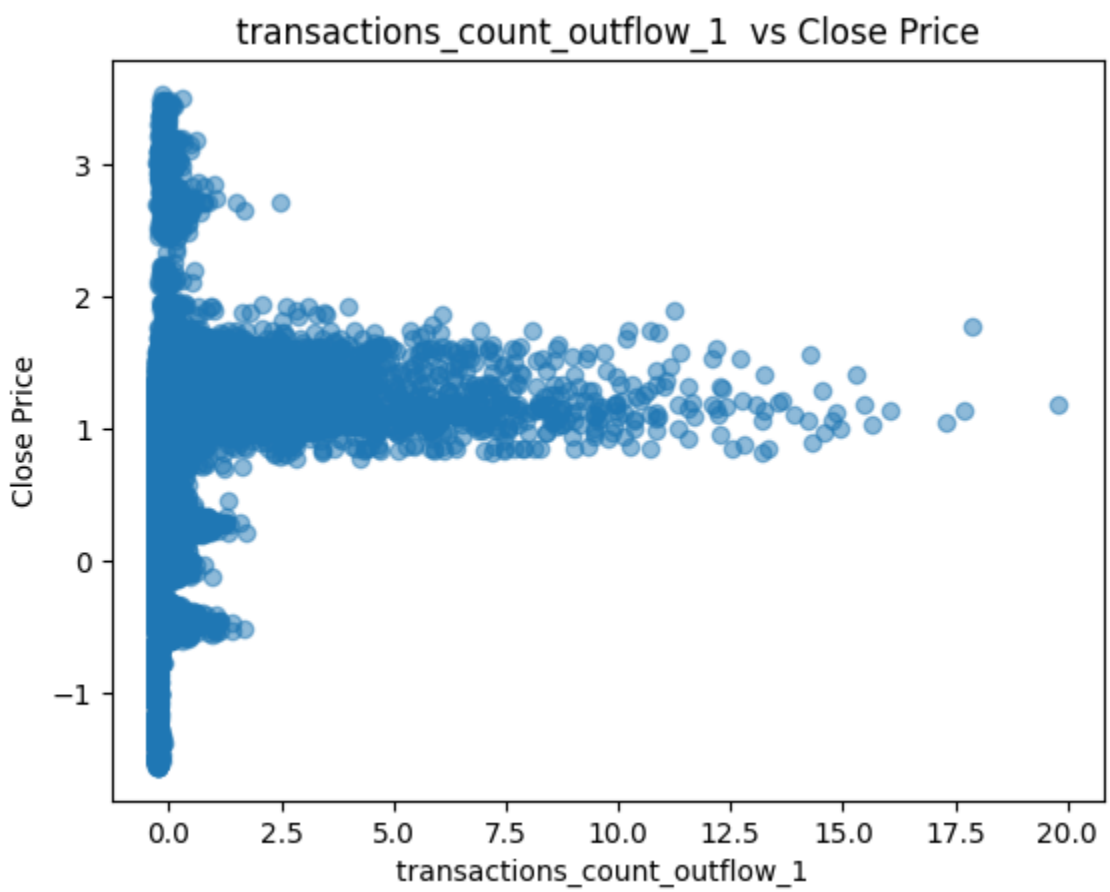
miner_supply_ratio



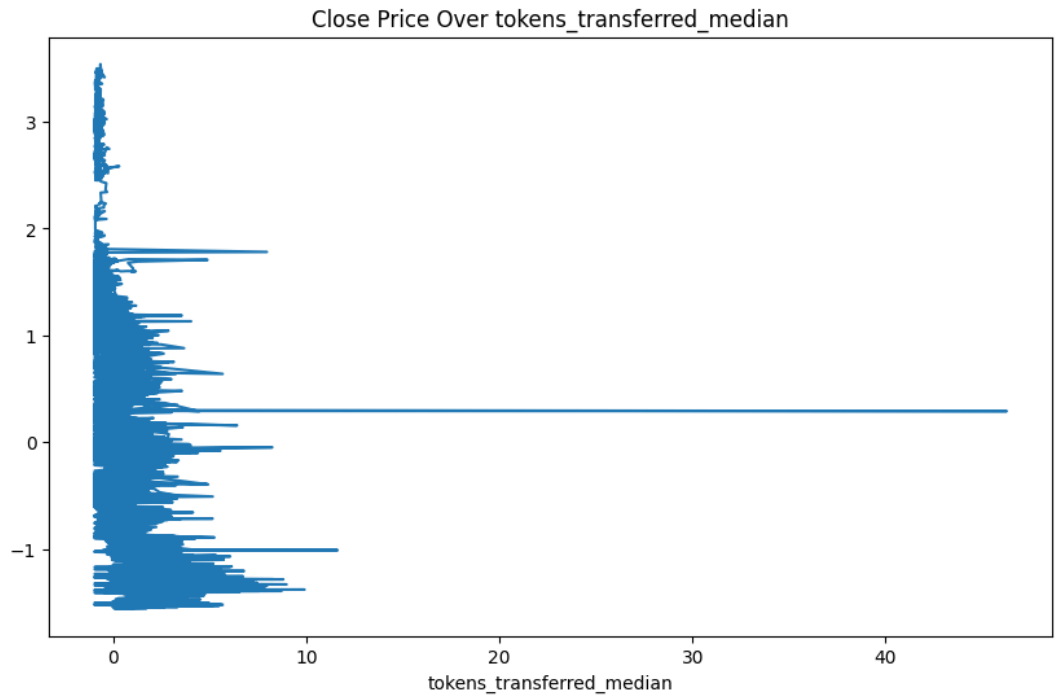
exchange_supply_ratio

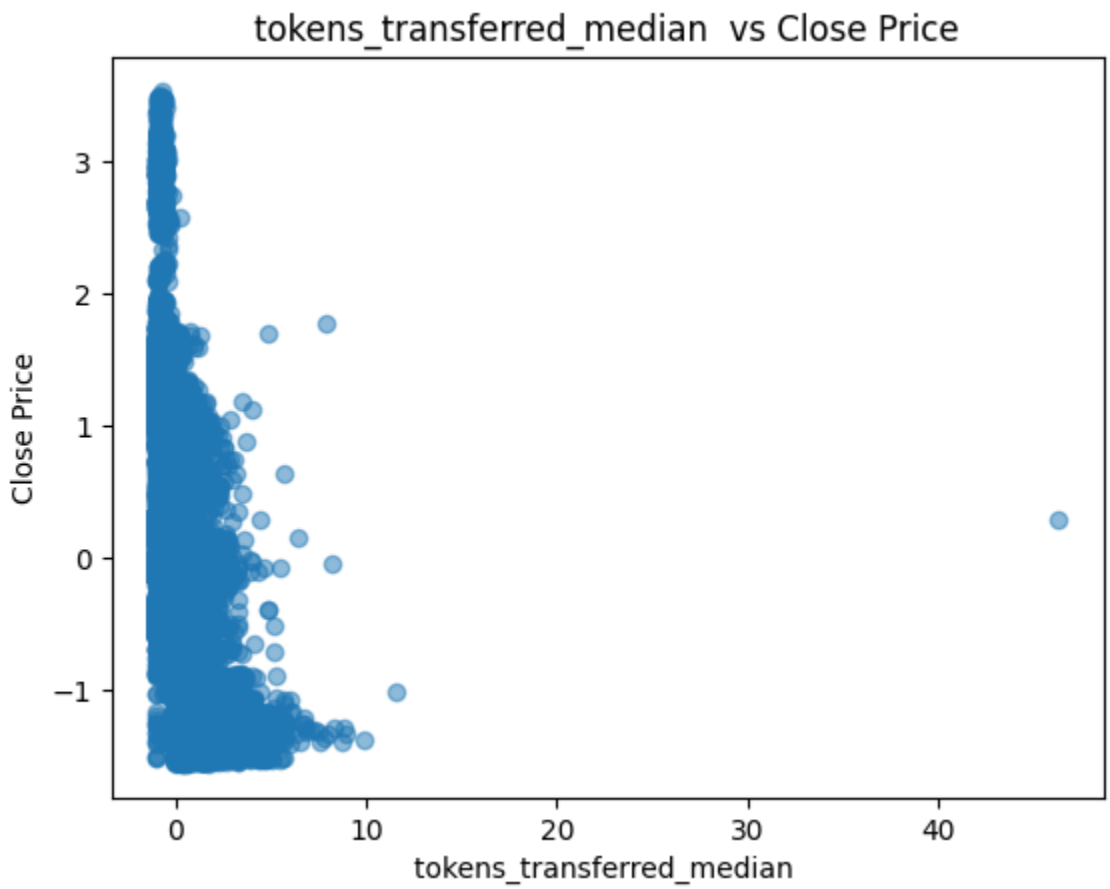


transactions_count_outflow_1

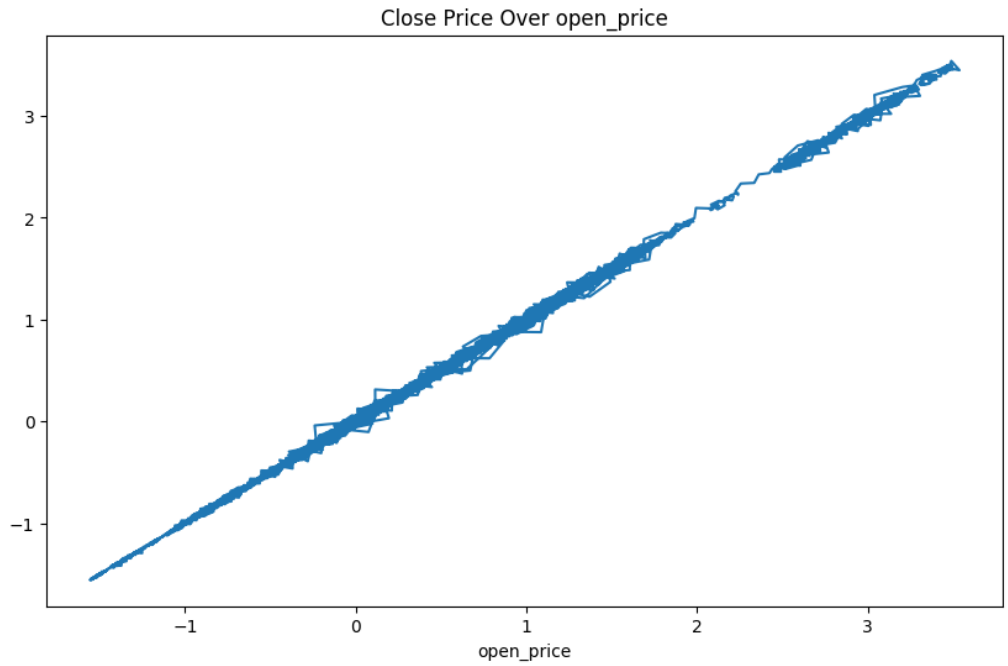


tokens_transferred_median

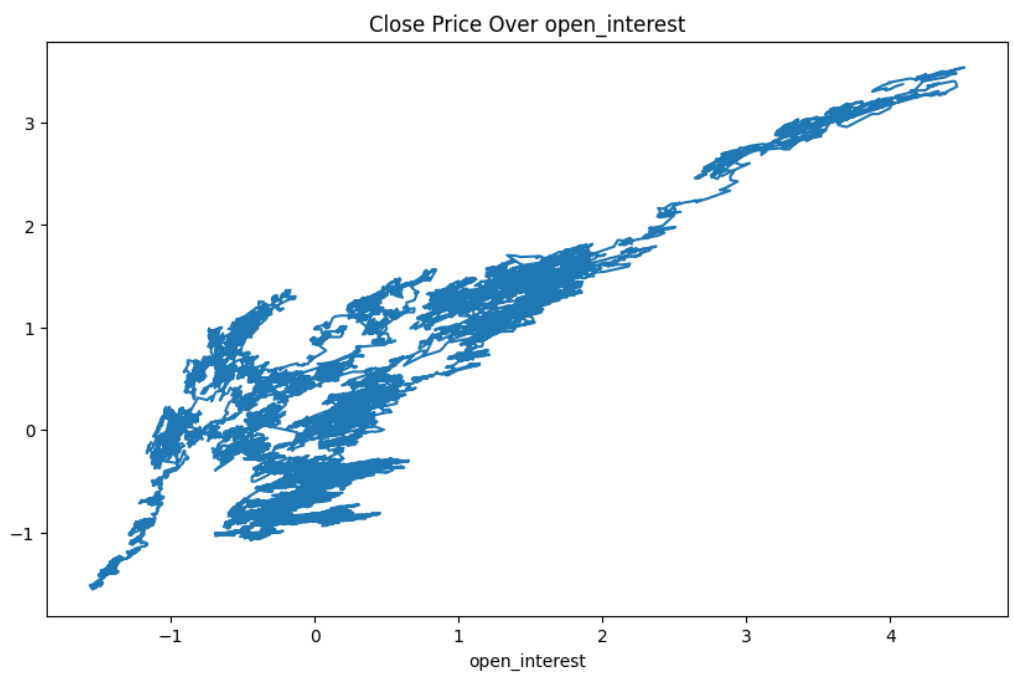




open_price

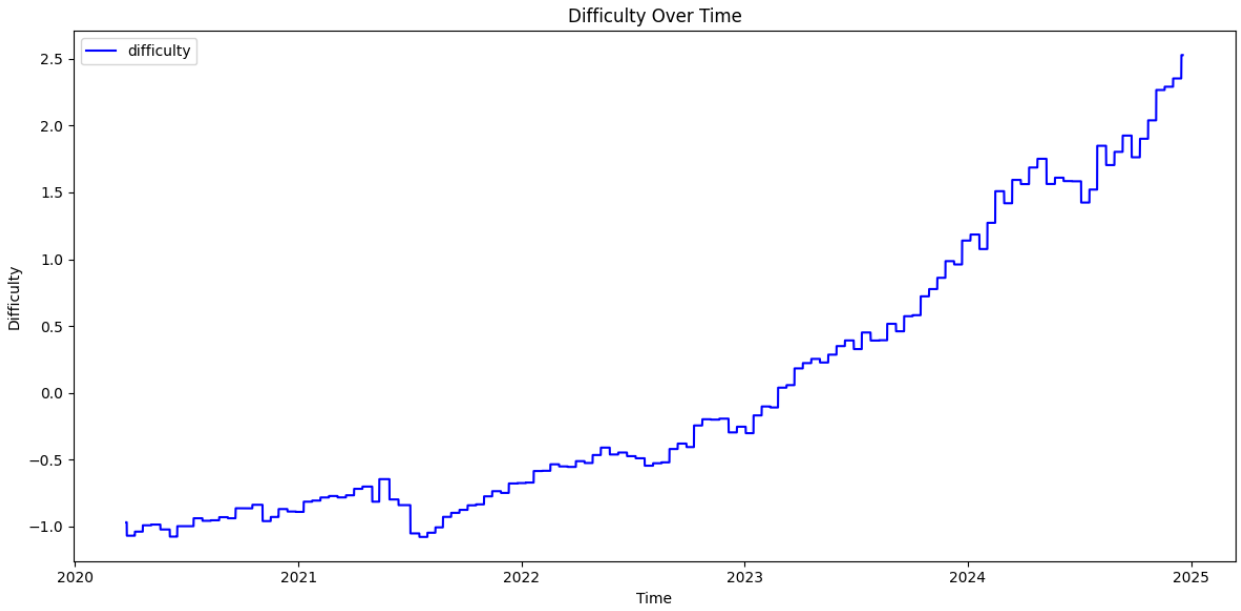


open_interest

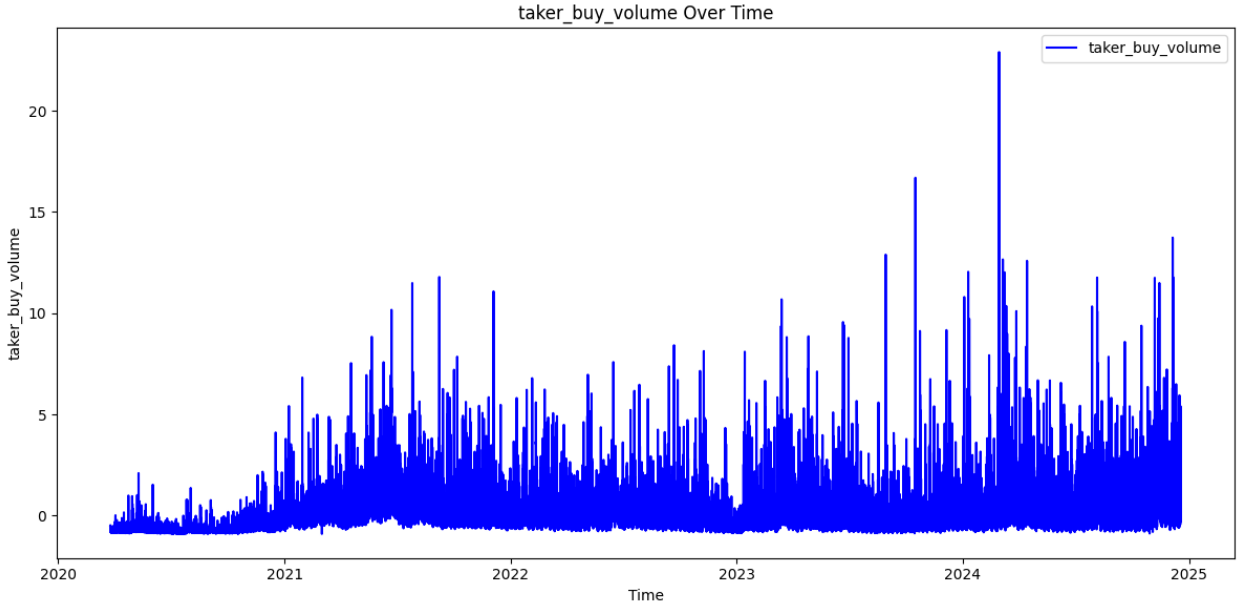


Single Variables Comparison (Time)

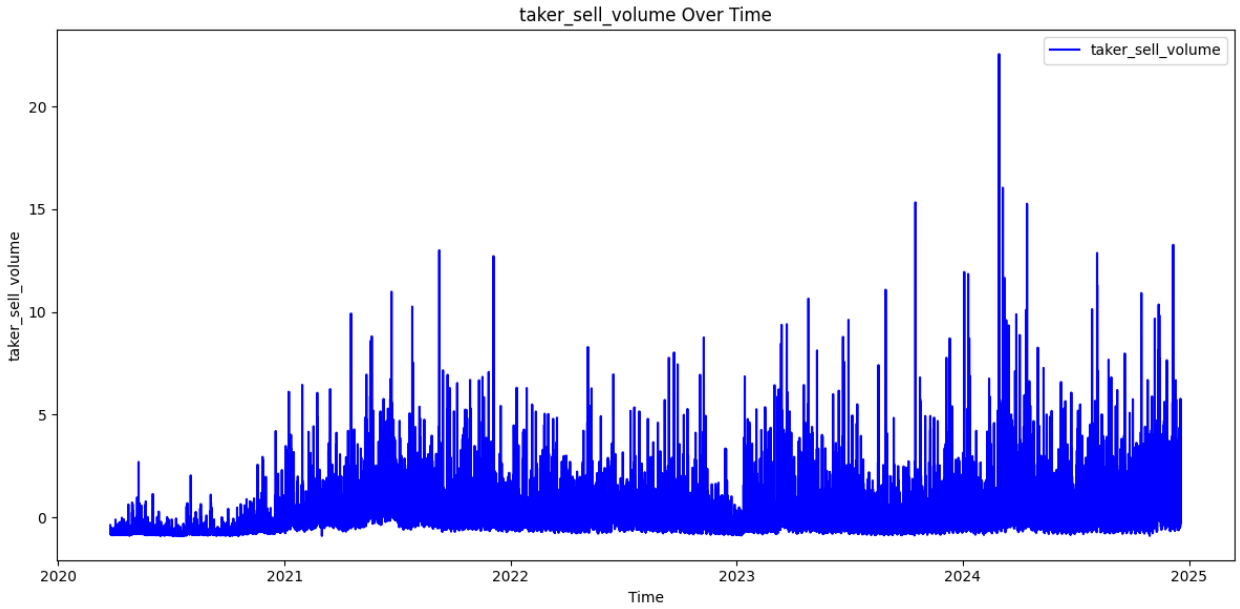
Difficult



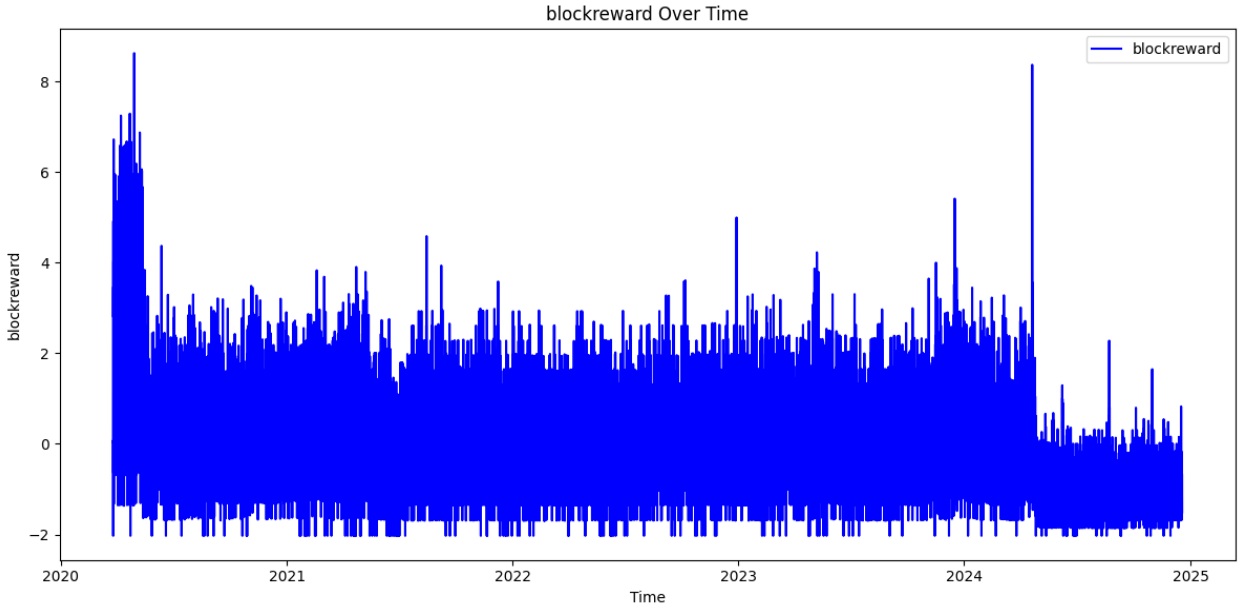
taker_buy_volume



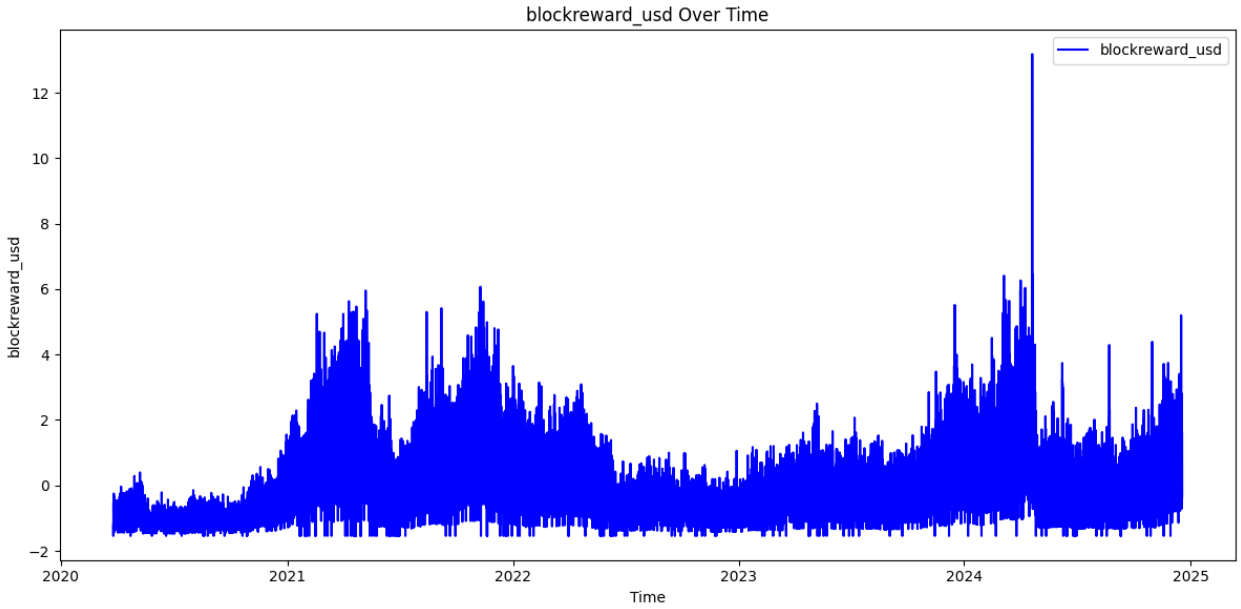
taker_sell_volume



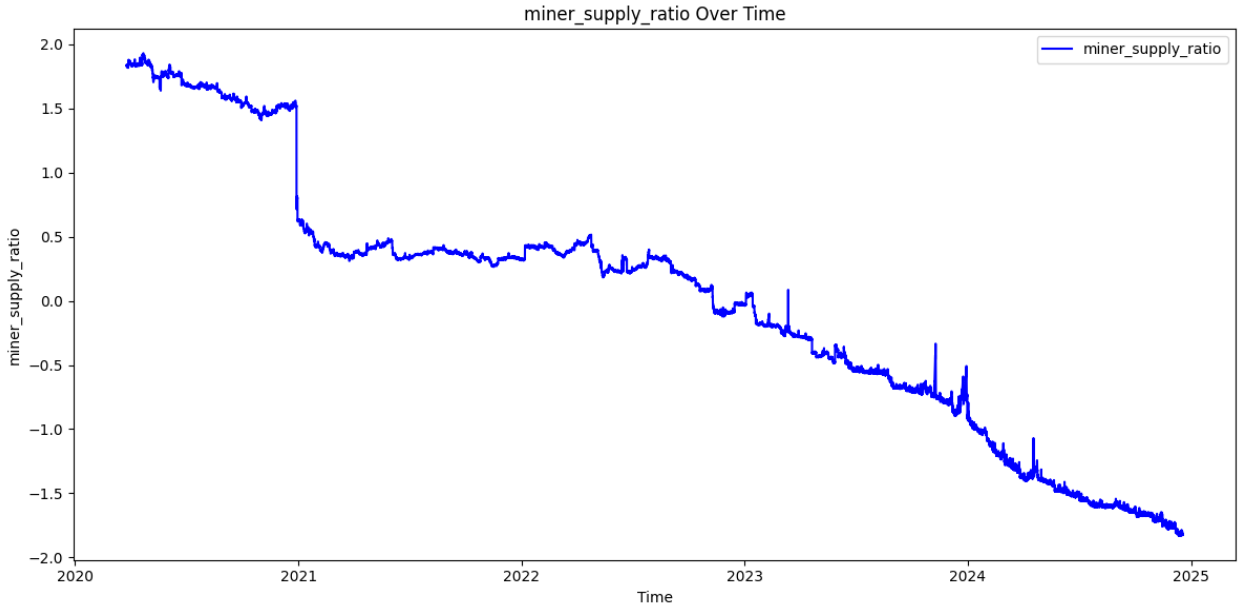
blockreward



blockreward_usd

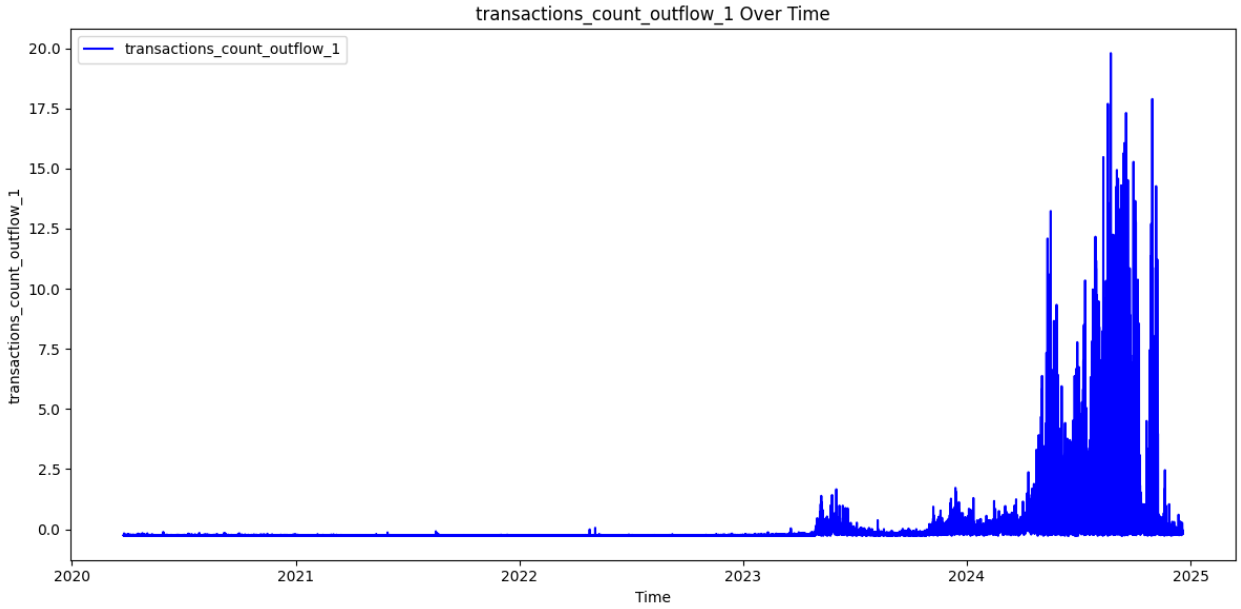


miner_supply_ratio

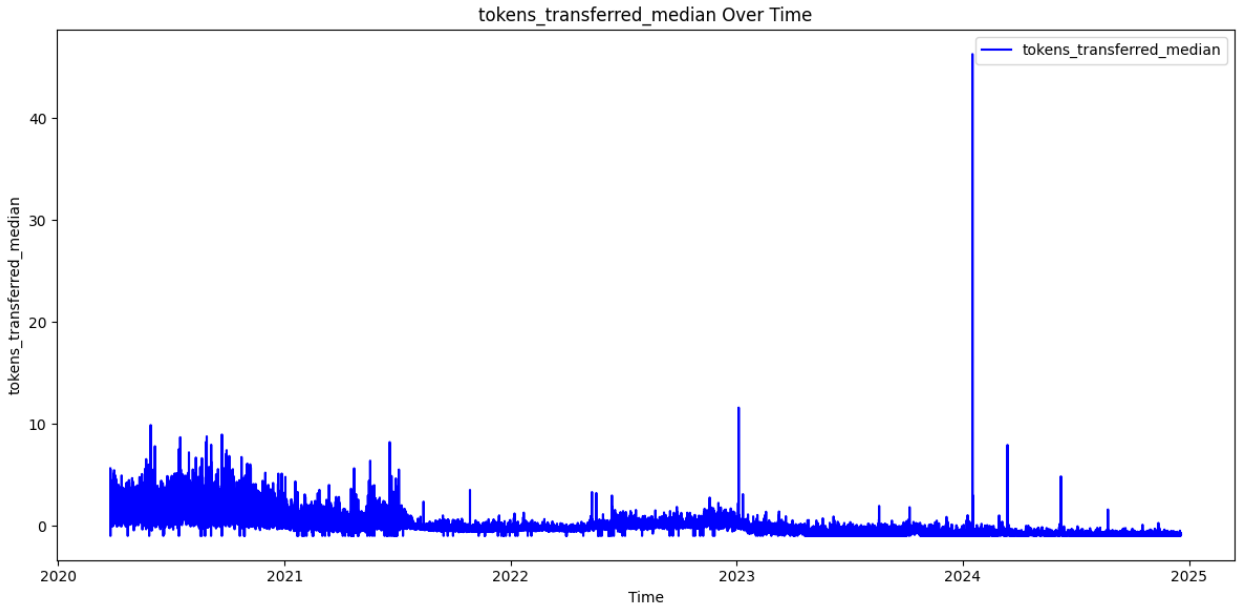


exchange_supply_ratio

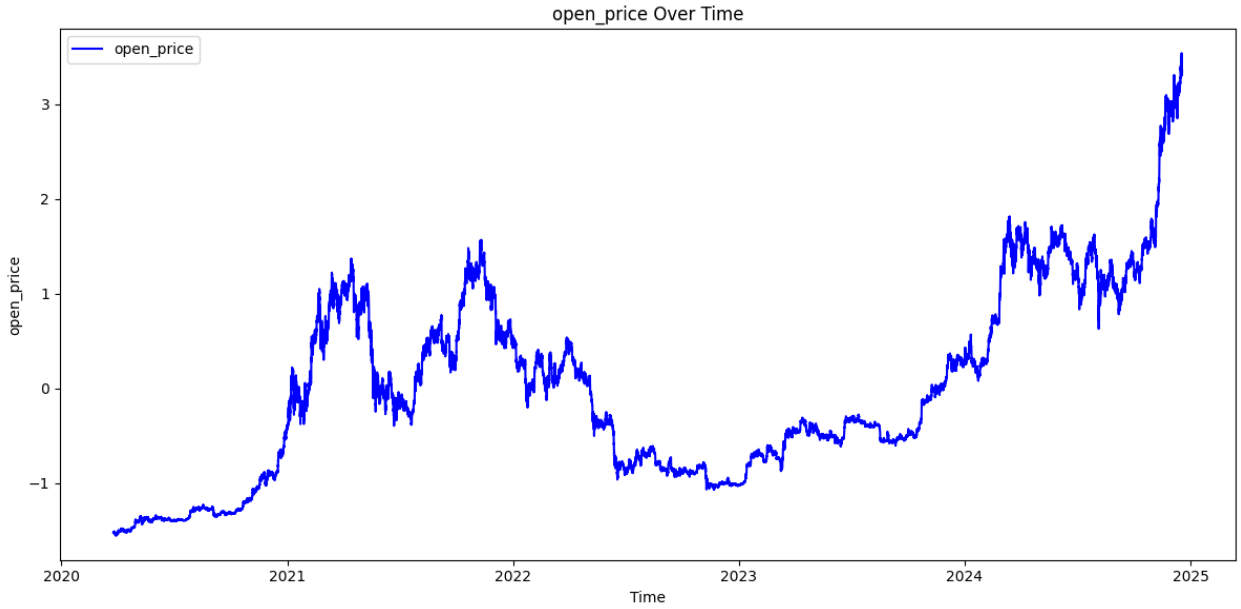
transactions_count_outflow_1



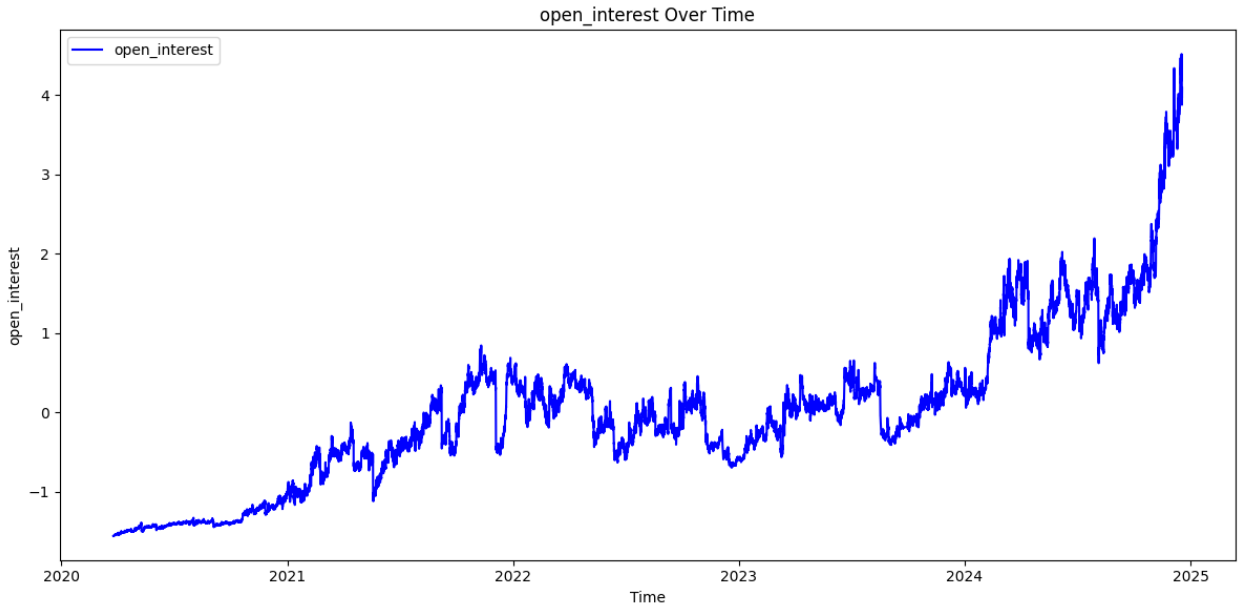
tokens_transferred_median



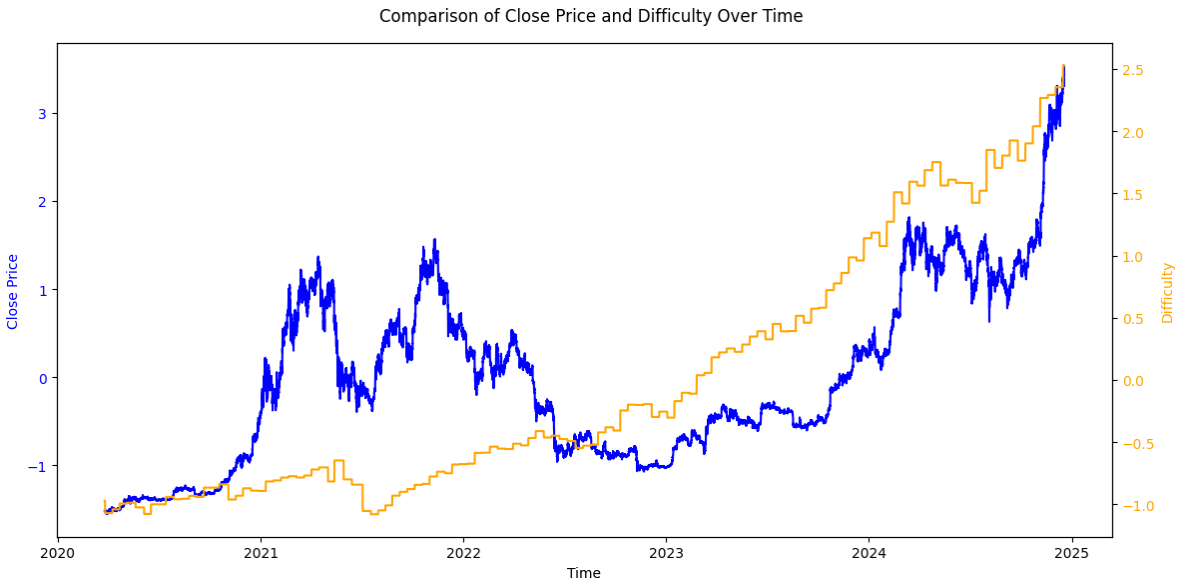
open_price

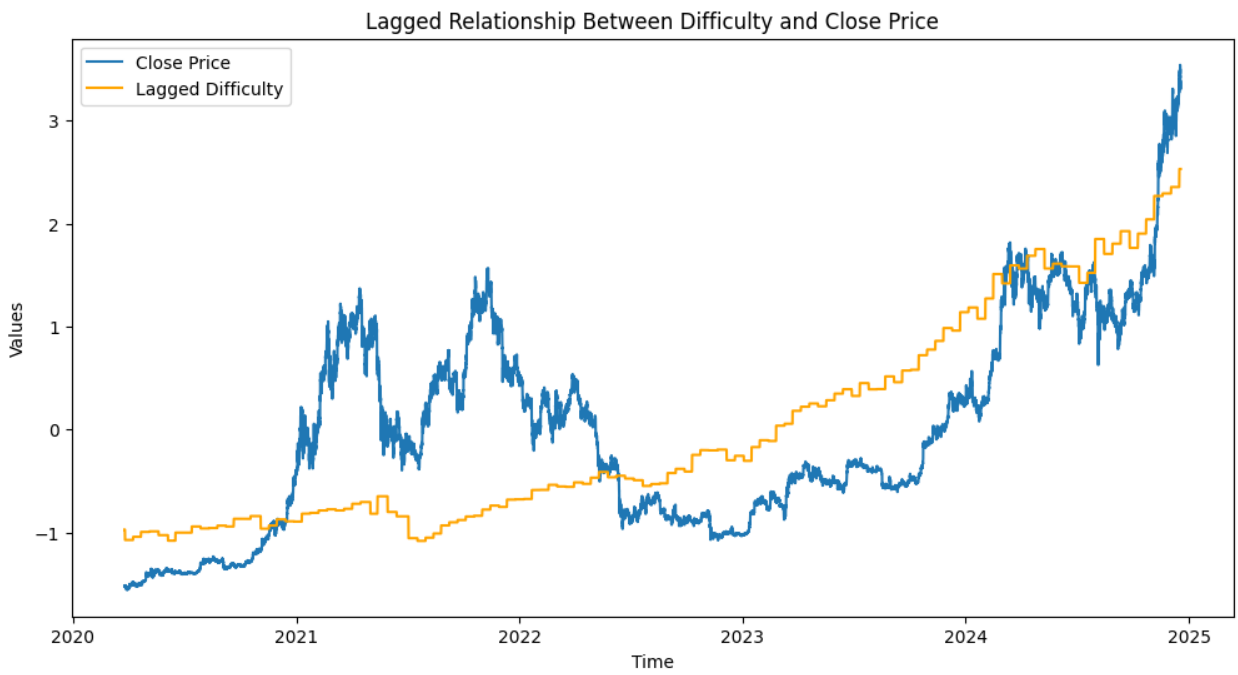


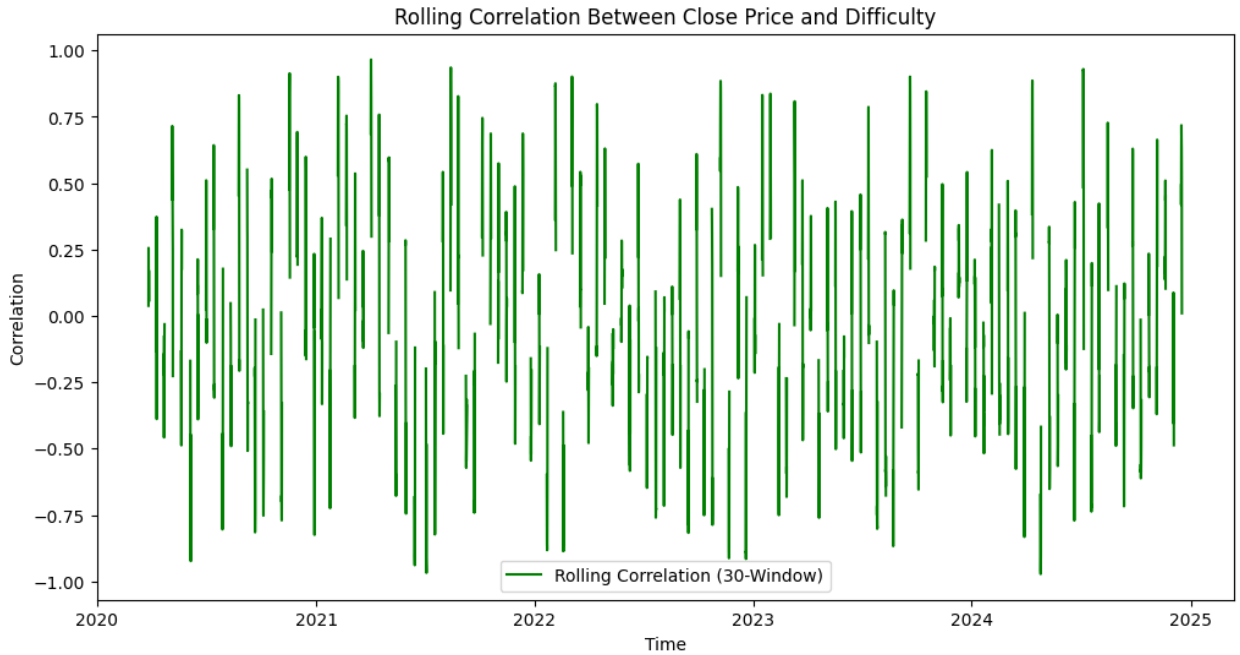
open_interest



Time Series Analysis








```python
finance_data['difficulty_lag'] = finance_data['difficulty'].shift(1)

lagged_corr = finance_data[['close_price', 'difficulty_lag']].corr()
print("Correlation with lagged difficulty:")
print(lagged_corr)
```

[134] ✓ 0.1s Open 'lagged_corr' in Data Wrangler

```
Correlation with lagged difficulty:
                 close_price  difficulty_lag
close_price         1.000000        0.639818
difficulty_lag      0.639818        1.000000
```

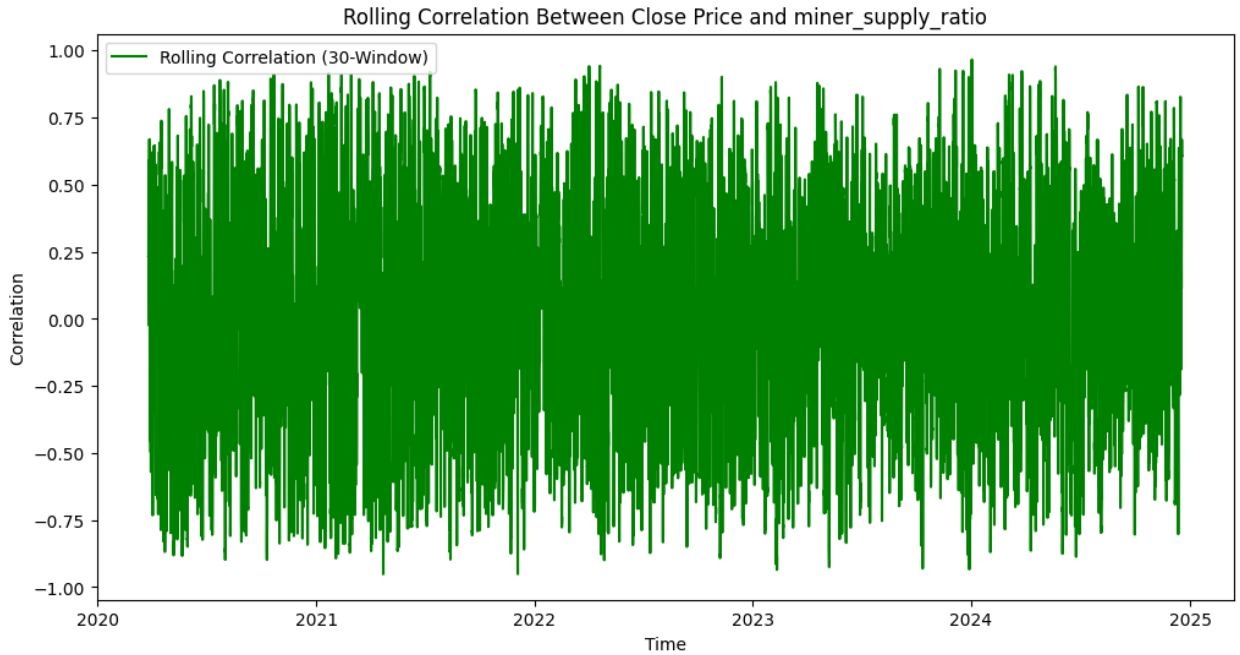
```python
threshold = 2
high_difficulty = finance_data[finance_data['difficulty'] > threshold]
low_difficulty = finance_data[finance_data['difficulty'] <= threshold]

print("High Difficulty Average Close Price:", high_difficulty['close_price'].mean())
print("Low Difficulty Average Close Price:", low_difficulty['close_price'].mean())
```

[143] ✓ 0.1s Pyth

```
High Difficulty Average Close Price: 2.5129945861629803
Low Difficulty Average Close Price: -0.08527752208376452
```

Comparison of Close Price and miner_supply_ratio Over Time
Close Price
miner_supply_ratio
Time
2020
2021
2022
2023
2024
2025

Lagged Relationship Between miner_supply_ratio and Close Price
Close Price
Lagged miner_supply_ratio
Values
Time
2020
2021
2022
2023
2024
2025




```python
finance_data['miner_supply_ratio_lag'] = finance_data['miner_supply_ratio'].shift(1)

lagged_corr = finance_data[['close_price', 'miner_supply_ratio_lag']].corr()
print("Correlation with lagged miner_supply_ratio:")
print(lagged_corr)
```

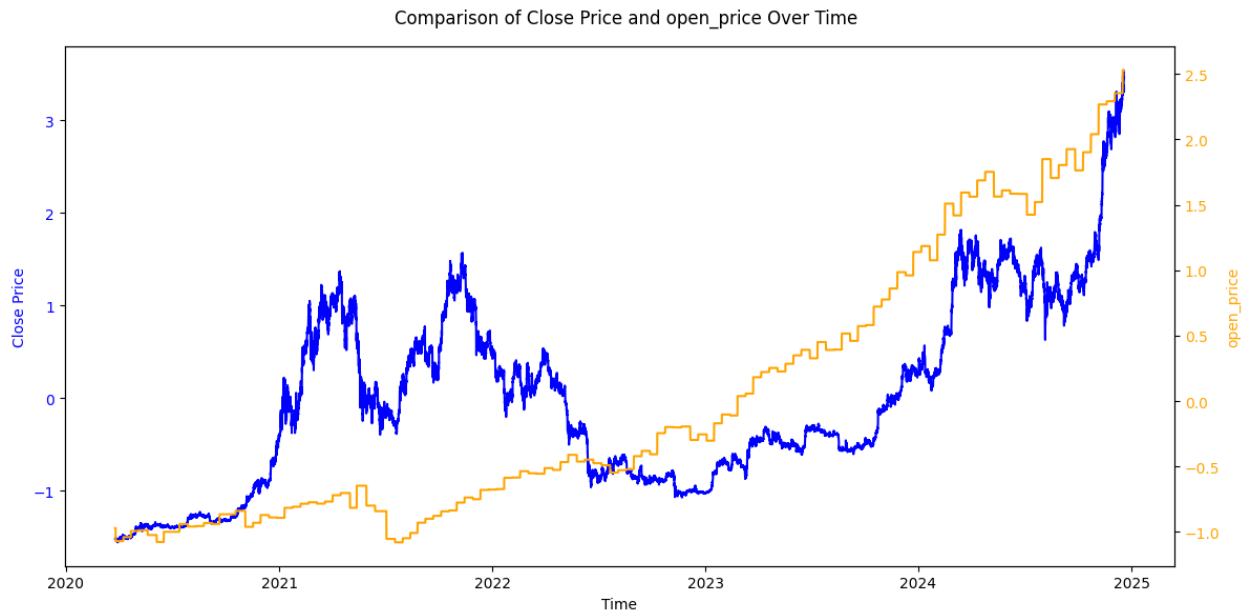
[135] ✓ 0.0s Open 'lagged_corr' in Data Wrangler Python

```
Correlation with lagged miner_supply_ratio:
                        close_price  miner_supply_ratio_lag
close_price                1.000000               -0.712864
miner_supply_ratio_lag    -0.712864                1.000000
```

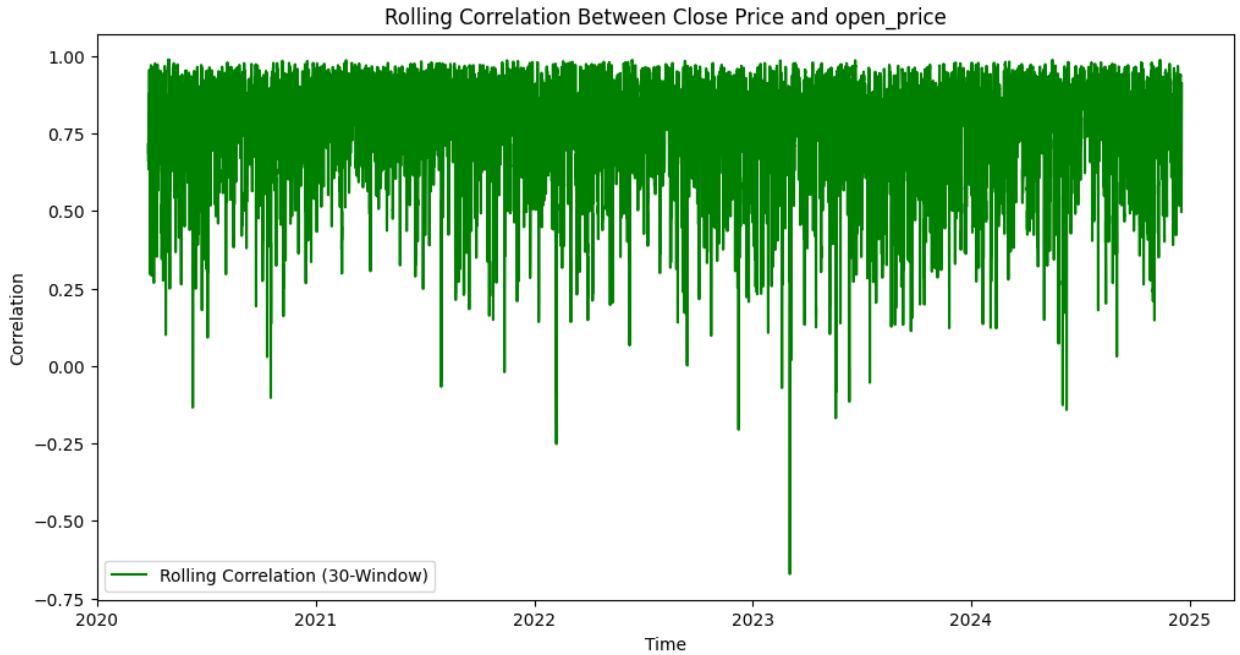
```python
threshold = 2
high_miner_supply_ratio = finance_data[finance_data['miner_supply_ratio'] > threshold]
low_miner_supply_ratio = finance_data[finance_data['miner_supply_ratio'] <= threshold]

print("High miner_supply_ratio Average Close Price:", high_miner_supply_ratio['close_price'].mean())
print("Low miner_supply_ratio Average Close Price:", low_miner_supply_ratio['close_price'].mean())
```

[144] ✓ 0.0s Python

```
High miner_supply_ratio Average Close Price: nan
Low miner_supply_ratio Average Close Price: 1.0958295601871515e-16
```


Comparison of Close Price and open_price Over Time
Close Price
open_price
Time
2020
2021
2022
2023
2024
2025

Lagged Relationship Between open_price and Close Price
Close Price
Lagged open_price
Values
Time
2020
2021
2022
2023
2024
2025




```python
finance_data['open_price_lag'] = finance_data['open_price'].shift(1)

lagged_corr = finance_data[['close_price', 'open_price_lag']].corr()
print("Correlation with lagged open_price:")
print(lagged_corr)
```

[136] ✓ 0.0s Open 'lagged_corr' in Data Wrangler Python

```
Correlation with lagged open_price:
                close_price  open_price_lag
close_price        1.000000        0.999813
open_price_lag     0.999813        1.000000
```

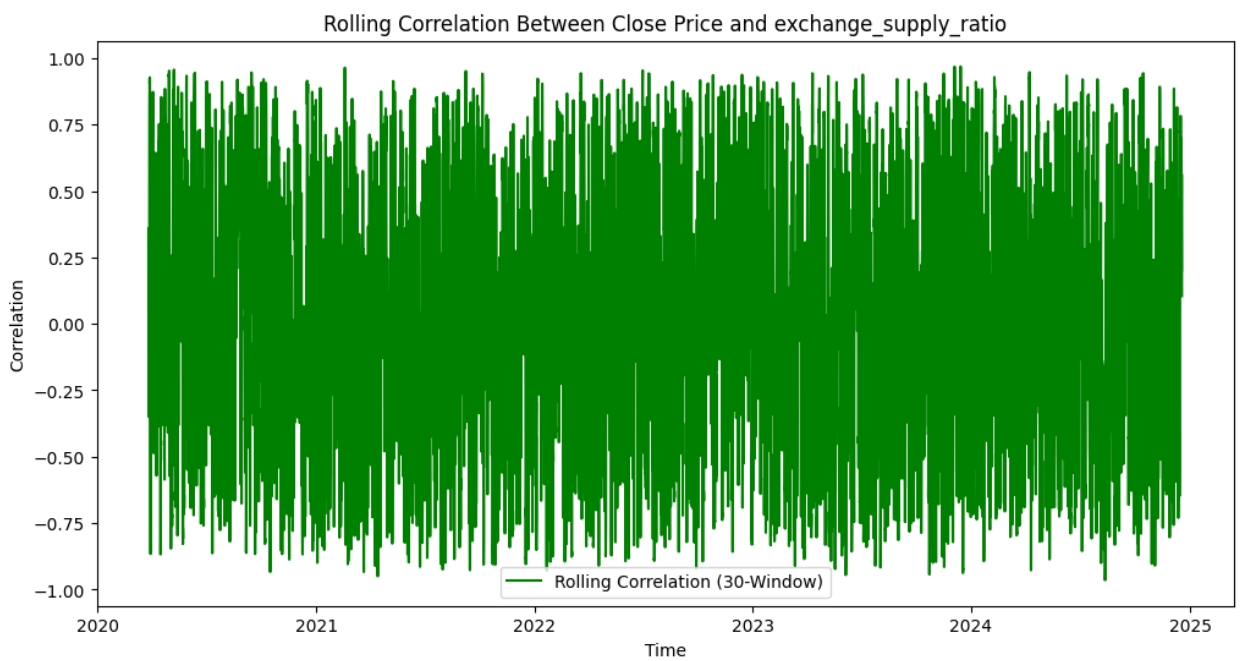
```python
threshold = 2
high_miner_open_price = finance_data[finance_data['open_price'] > threshold]
low_miner_open_price= finance_data[finance_data['open_price'] <= threshold]

print("High miner_open_price Average Close Price:", high_miner_open_price['close_price'].mean())
print("Low miner_open_price Average Close Price:", low_miner_open_price['close_price'].mean())
```

[145] ✓ 0.0s Python

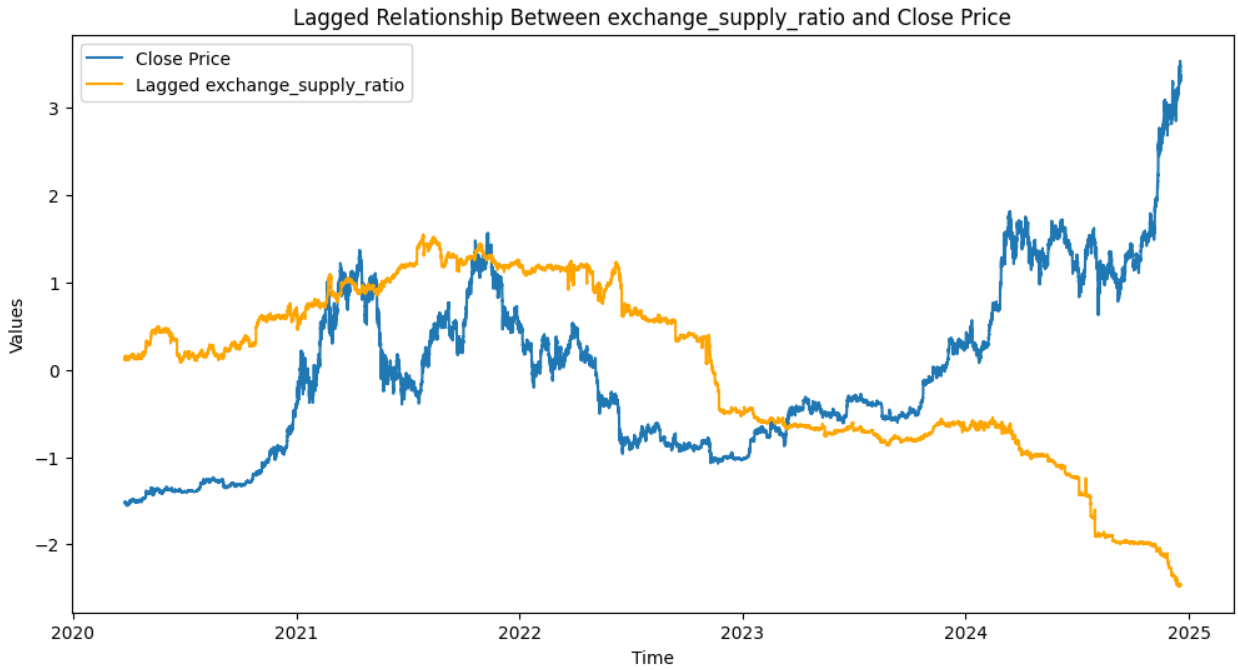
```
High miner_open_price Average Close Price: 2.925607512693099
Low miner_open_price Average Close Price: -0.0662567659174684
```

Additional

Comparison of Close Price and exchange_supply_ratio Over Time
Close Price
exchange_supply_ratio
Time
2020
2021
2022
2023
2024
2025


Rolling Correlation Between Close Price and exchange_supply_ratio
Correlation
Time
Rolling Correlation (30-Window)
2020
2021
2022
2023
2024
2025

Lagged Relationship Between exchange_supply_ratio and Close Price
Close Price
Lagged exchange_supply_ratio
Values
Time
3
2
1
0
−1
−2
2020
2021
2022
2023
2024
2025

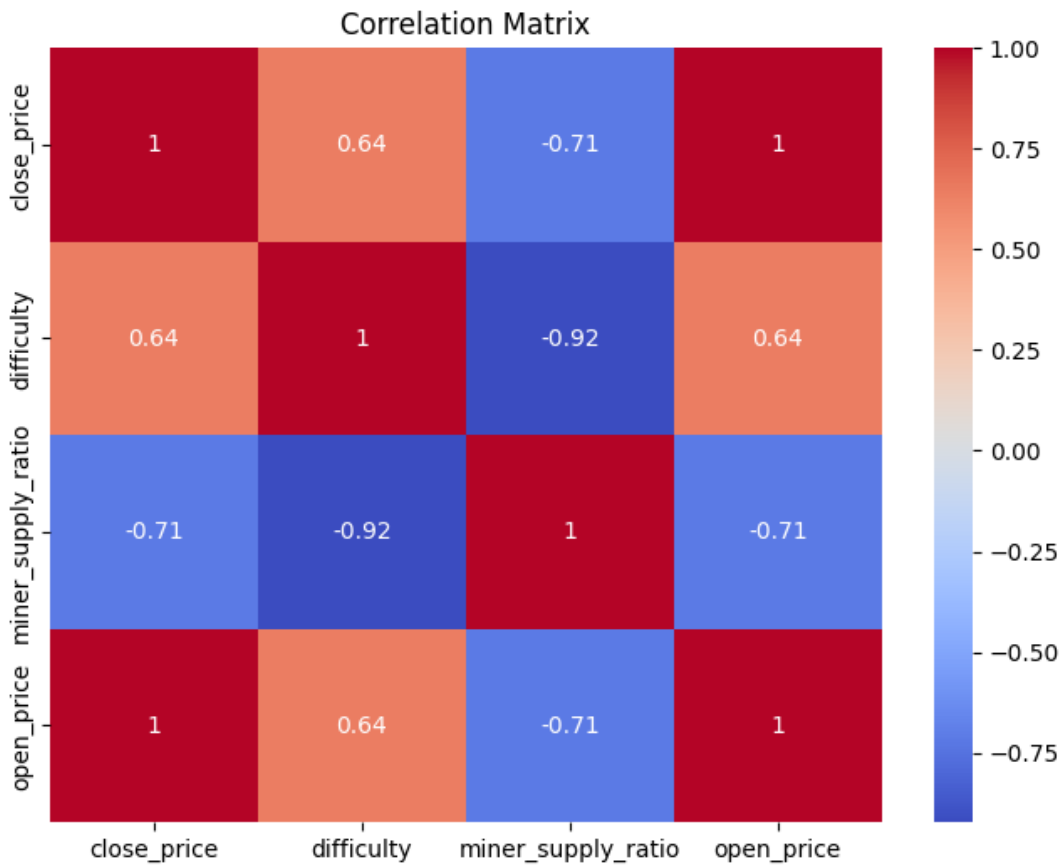

Lagged Relationship Between exchange_supply_ratio and Close Price
Close Price
Lagged exchange_supply_ratio
Values
Time
3
2
1
0
−1
−2
2020
2021
2022
2023
2024
2025

Summary of Time Series Analysis

```python
correlation = finance_data[['close_price', 'difficulty', 'miner_supply_ratio', 'open_price']].corr()
print(correlation)
```

[137] ✓ 0.0s Open 'correlation' in Data Wrangler Python

```
                    close_price  difficulty  miner_supply_ratio  open_price
close_price            1.000000    0.639837           -0.712875    0.999906
difficulty             0.639837    1.000000           -0.922150    0.639787
miner_supply_ratio    -0.712875   -0.922150            1.000000   -0.712865
open_price             0.999906    0.639787           -0.712865    1.000000
```

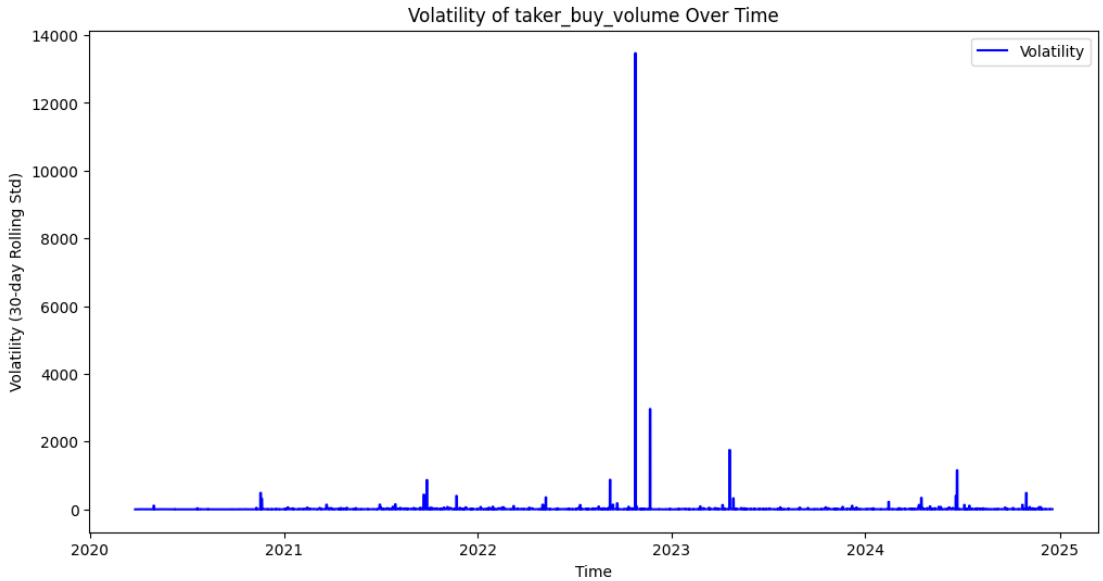


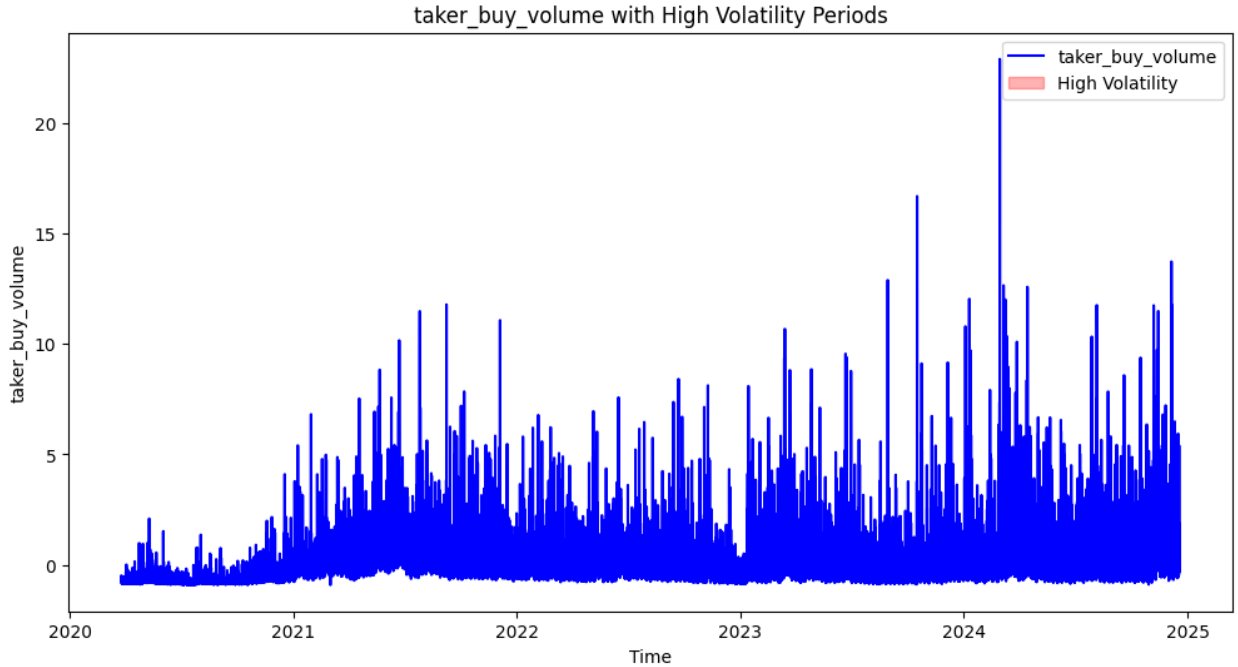

```python
from sklearn.linear_model import LinearRegression

X = finance_data[['difficulty', 'miner_supply_ratio', 'open_price']]
y = finance_data['close_price']

model = LinearRegression()
model.fit(X, y)

print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)
```

[142] ✓ 0.1s Python

```
Coefficients: [2.46137675e-04 7.73251266e-05 9.99803788e-01]
Intercept: -4.338782368973947e-17
```

Volatility Analysis

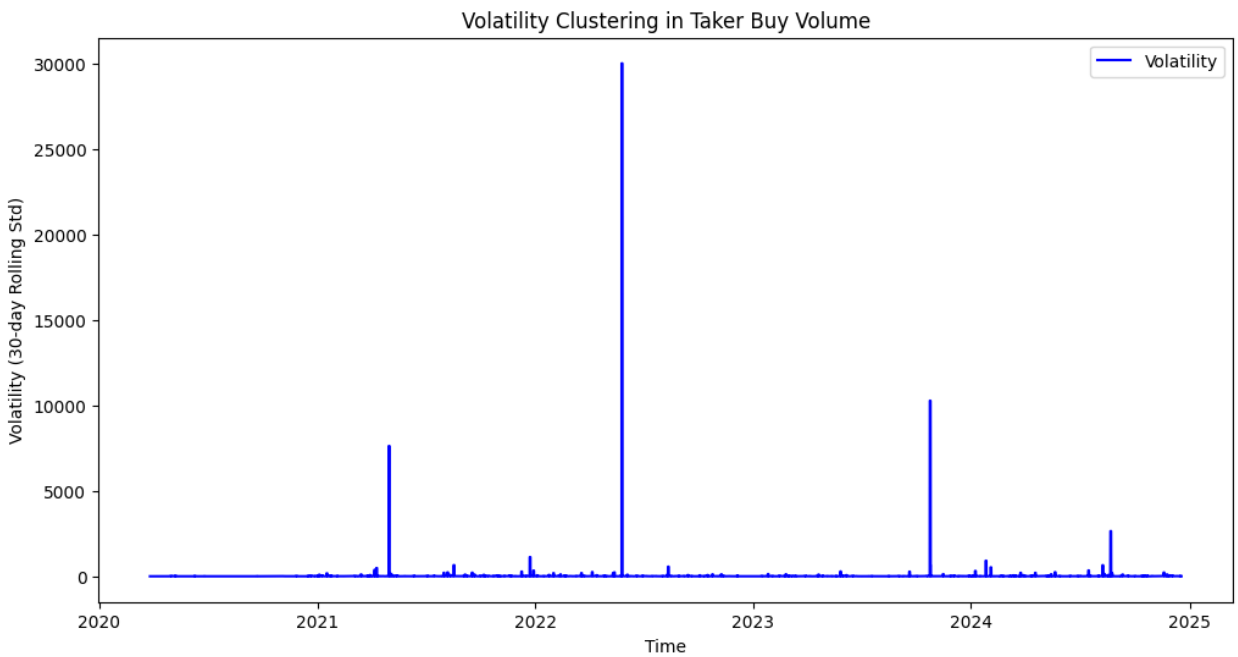


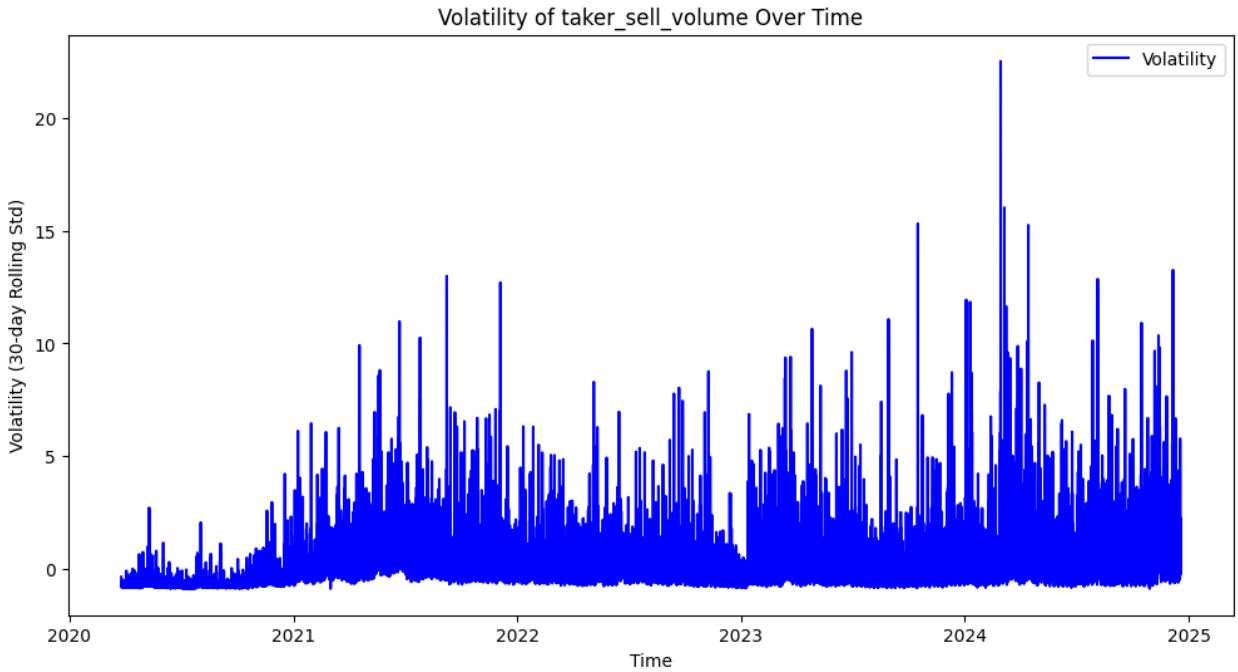


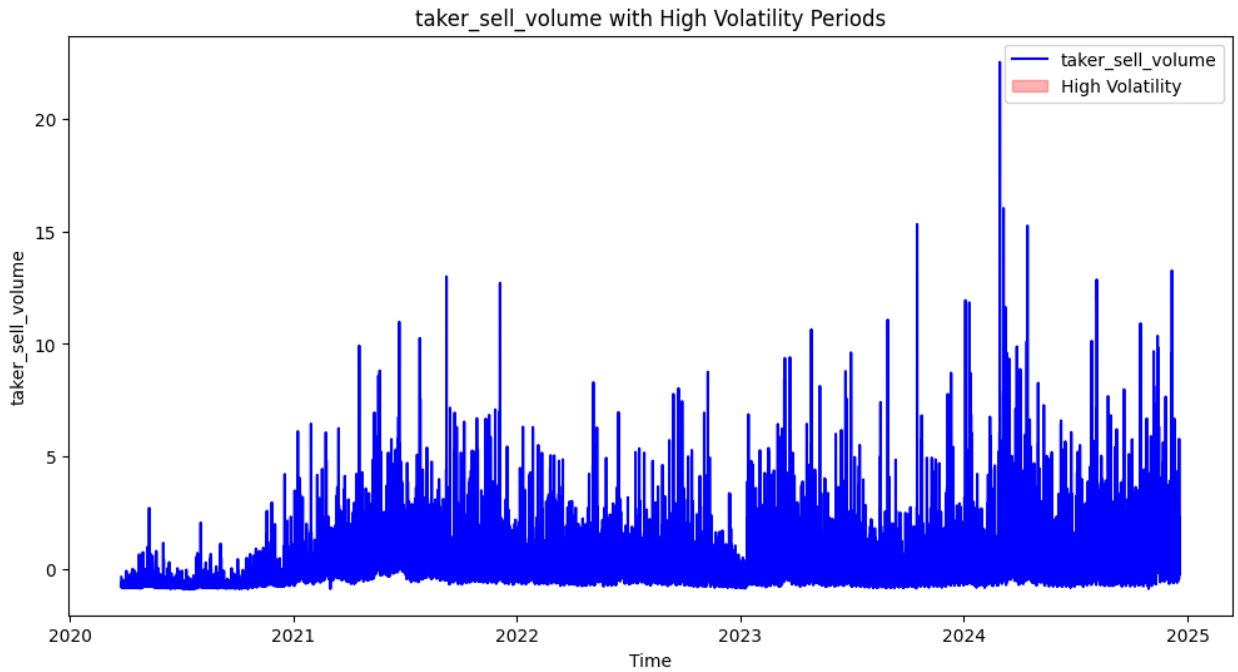
```python
high_volatility = finance_data[finance_data['taker_buy_volatility_category'] == 'High']
low_volatility = finance_data[finance_data['taker_buy_volatility_category'] == 'Low']

print("High Volatility Period Average Taker Buy Volume:", high_volatility['taker_buy_volume'].mean())
print("Low Volatility Period Average Taker Buy Volume:", low_volatility['taker_buy_volume'].mean())
```

[151] ✓ 0.0s Python

```
High Volatility Period Average Taker Buy Volume: 0.1654772236224292
Low Volatility Period Average Taker Buy Volume: -0.0010837016197258542
```



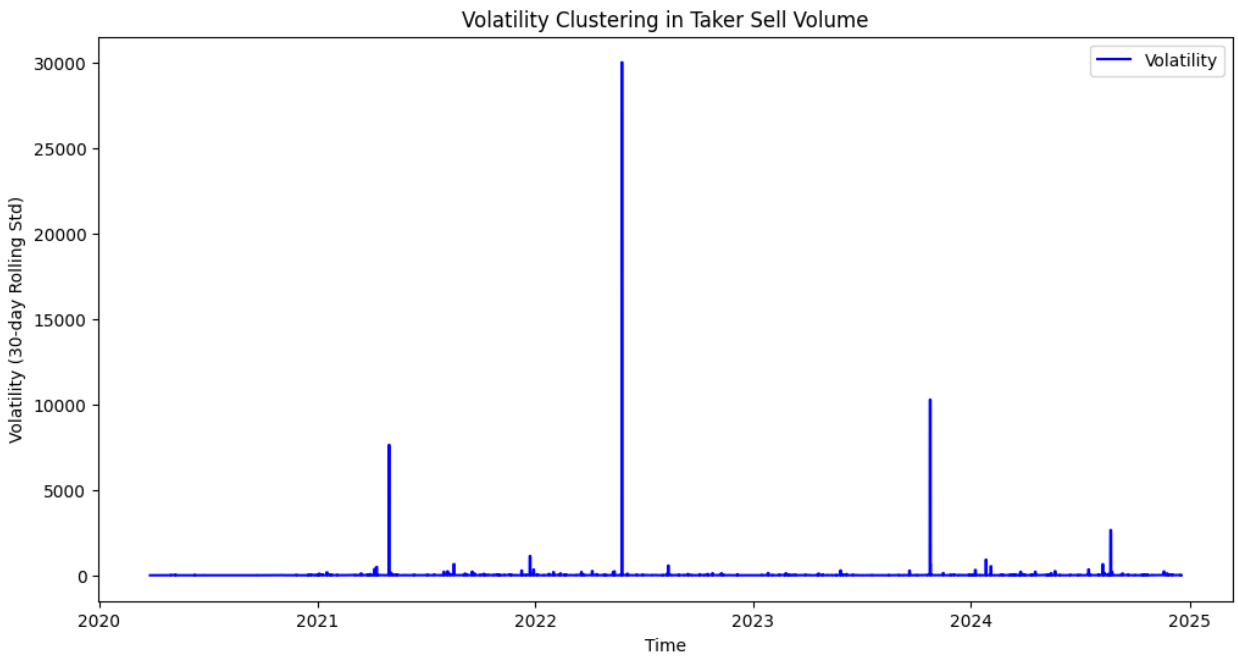

Volatility of taker_sell_volume Over Time
Volatility
Volatility (30-day Rolling Std)
20
15
10
5
0
2020
2021
2022
2023
2024
2025
Time

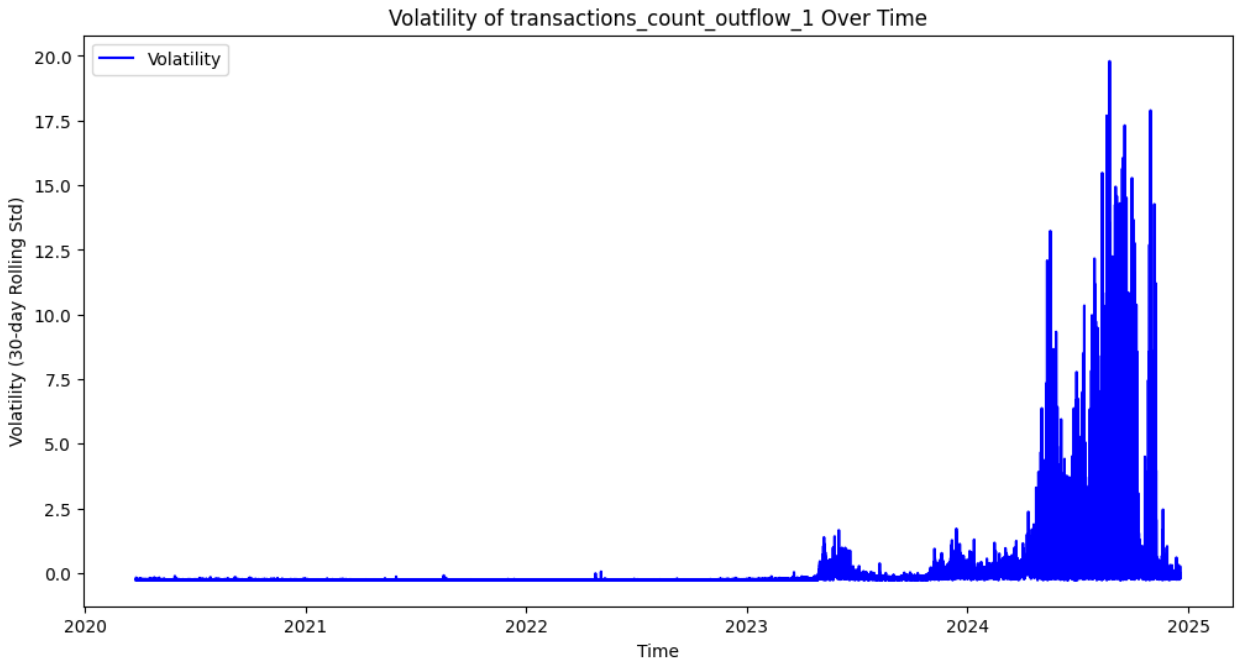

taker_sell_volume with High Volatility Periods
taker_sell_volume
High Volatility
taker_sell_volume
20
15
10
5
0
2020
2021
2022
2023
2024
2025
Time

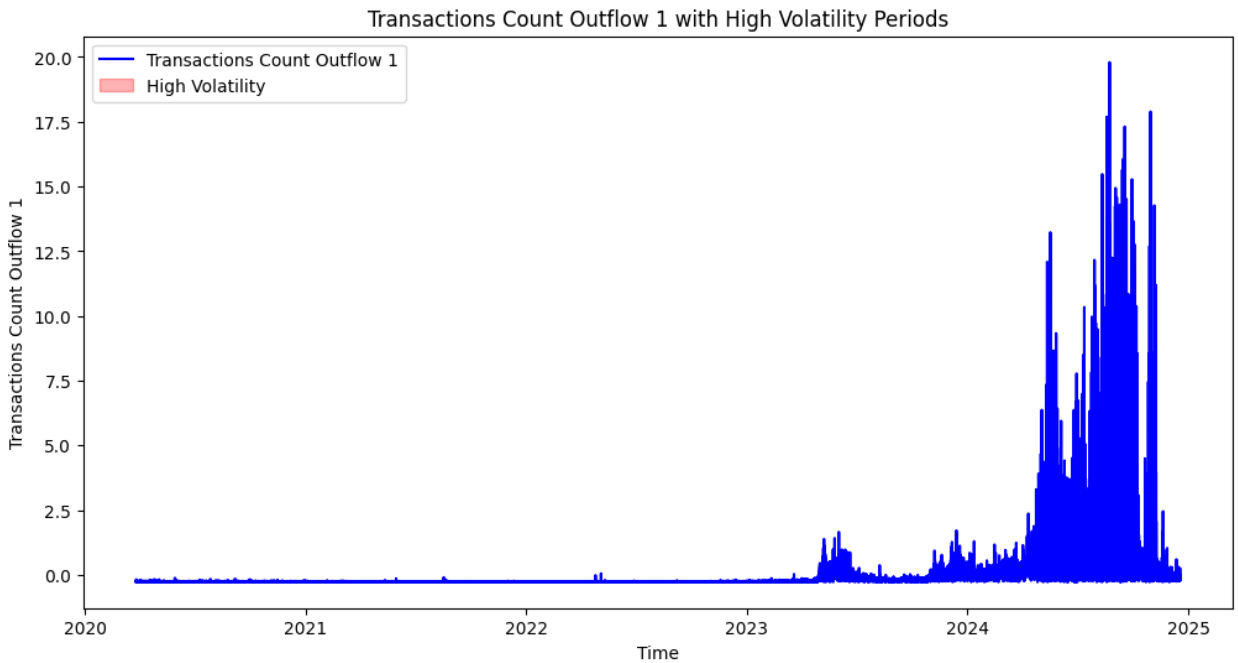
```python
high_volatility = finance_data[finance_data['taker_sell_volume_volatility_category'] == 'High']
low_volatility = finance_data[finance_data['taker_sell_volume_volatility_category'] == 'Low']

print("High Volatility Period Average taker_sell_volume:", high_volatility['taker_sell_volume'].mean())
print("Low Volatility Period Average taker_sell_volume:", low_volatility['taker_sell_volume'].mean())
```

[161] ✓ 0.0s Python

```
High Volatility Period Average taker_sell_volume: 0.41798846862639844
Low Volatility Period Average taker_sell_volume: -0.0015163555744887444
```



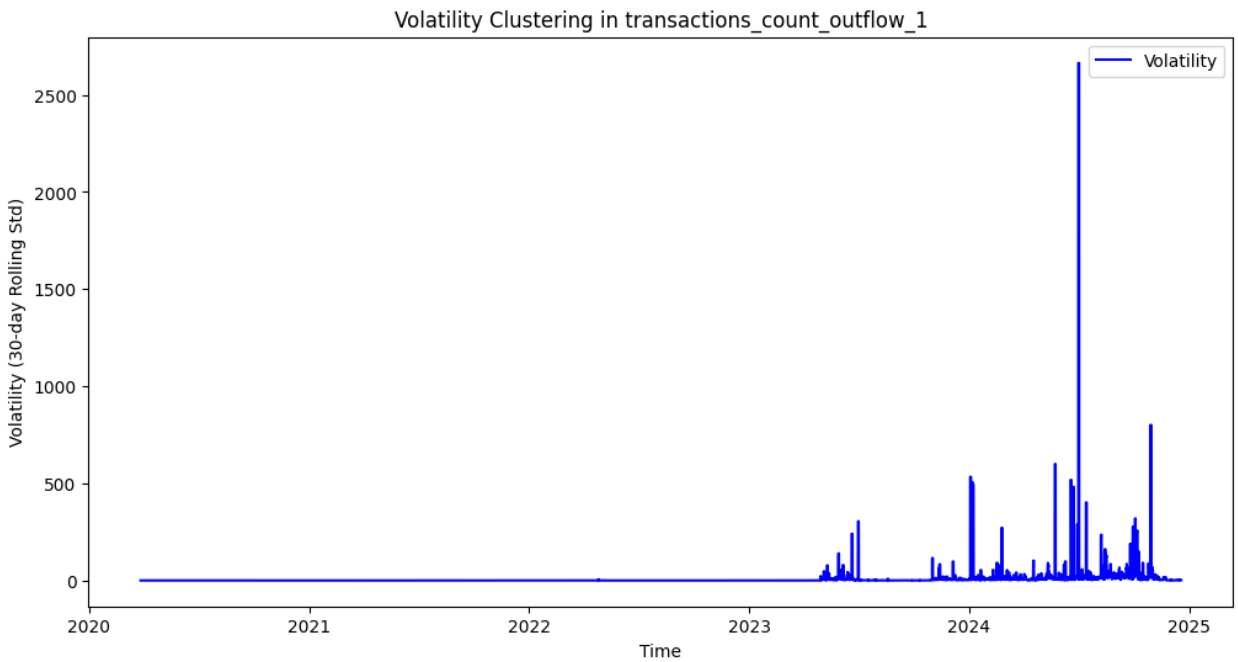

Volatility of transactions_count_outflow_1 Over Time
Volatility
Volatility (30-day Rolling Std)
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
2020
2021
2022
2023
2024
2025
Time


Transactions Count Outflow 1 with High Volatility Periods
Transactions Count Outflow 1
High Volatility
Transactions Count Outflow 1
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
2020
2021
2022
2023
2024
2025
Time

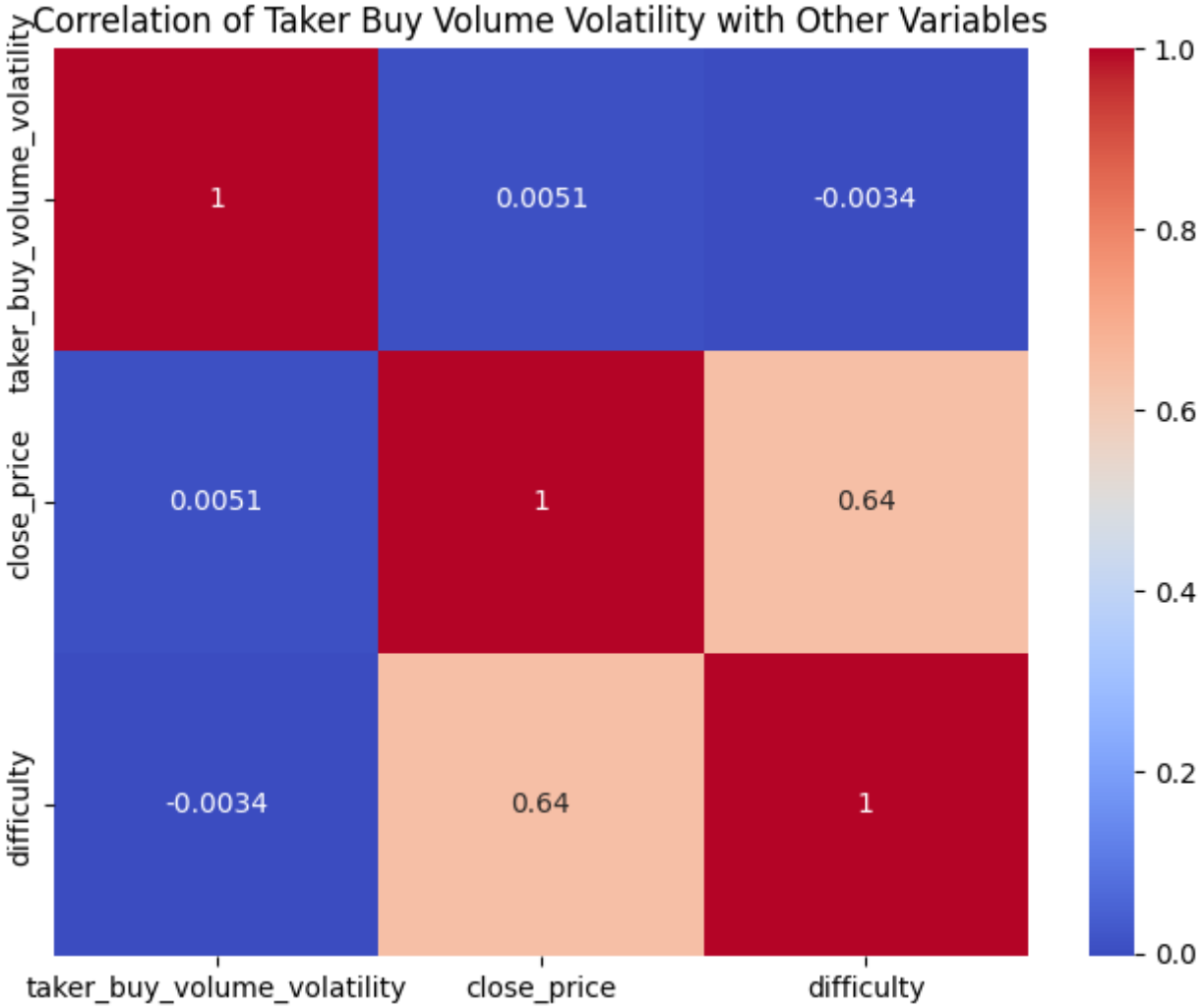
```python
high_volatility = finance_data[finance_data['transactions_outflow_volatility_category'] == 'High']
low_volatility = finance_data[finance_data['transactions_outflow_volatility_category'] == 'Low']

print("High Volatility Period Average transactions_count_outflow_1:", high_volatility['transactions_count_outflow_1'].mean())
print("Low Volatility Period Average transactions_count_outflow_1:", low_volatility['transactions_count_outflow_1'].mean())
```

[173] ✓ 0.0s Python

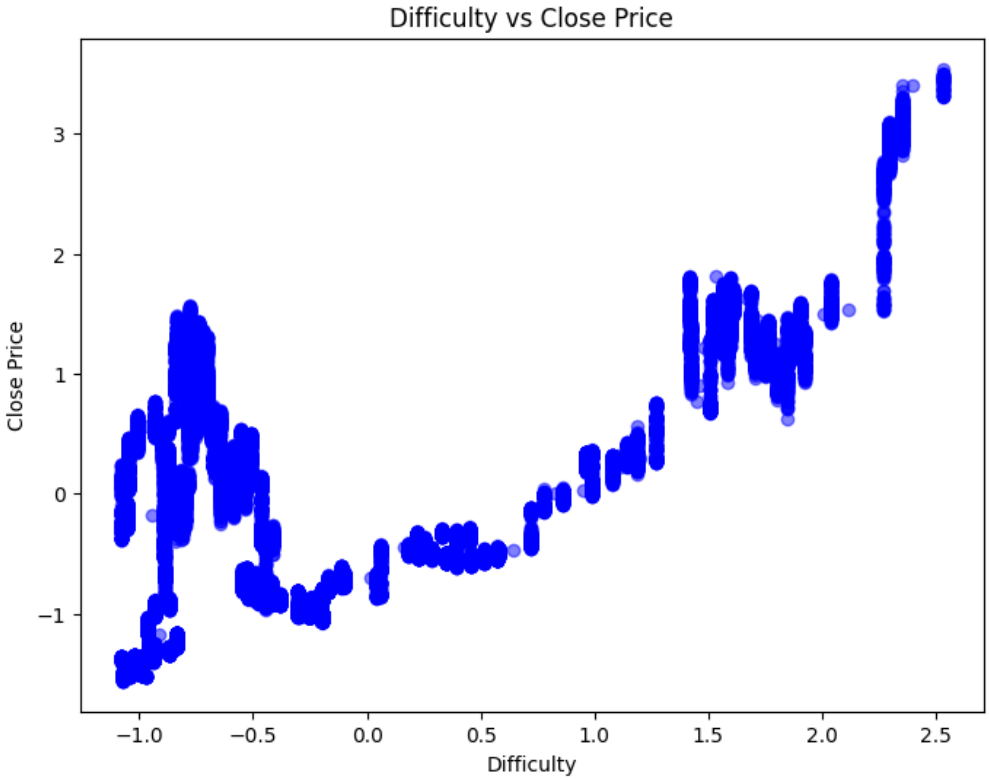
```
High Volatility Period Average transactions_count_outflow_1: 1.0362822606984154
Low Volatility Period Average transactions_count_outflow_1: -0.019851175483686916
```

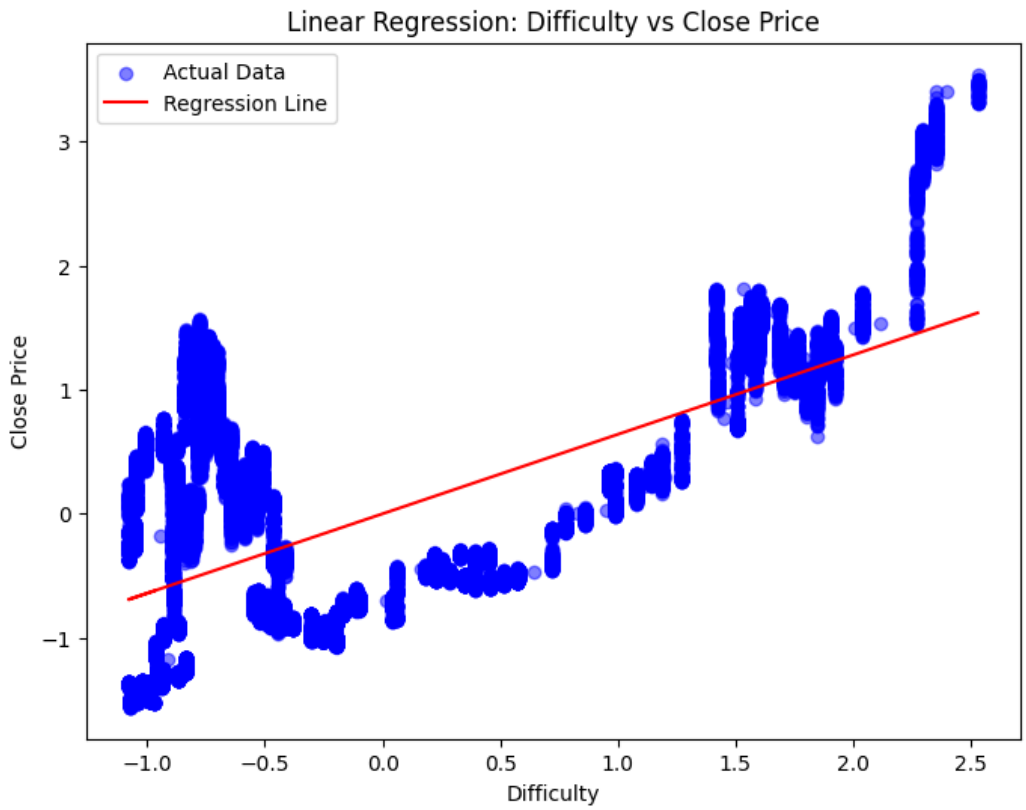


Summary of Volatility Impact



Correlation and Regression Analysis






```python
from sklearn.linear_model import LinearRegression

X = finance_data[['difficulty']]
y = finance_data['close_price']

model = LinearRegression()
model.fit(X, y)

print("Intercept:", model.intercept_)
print("Coefficient for Difficulty:", model.coef_[0])

finance_data['predicted_close_price'] = model.predict(X)
```

[193] ✓ 0.0s Python

```
Intercept: 2.0774424613705122e-16
Coefficient for Difficulty: 0.6398367937383278
```

```
plt.ylabel('Close Price')
plt.legend()
plt.show()
```

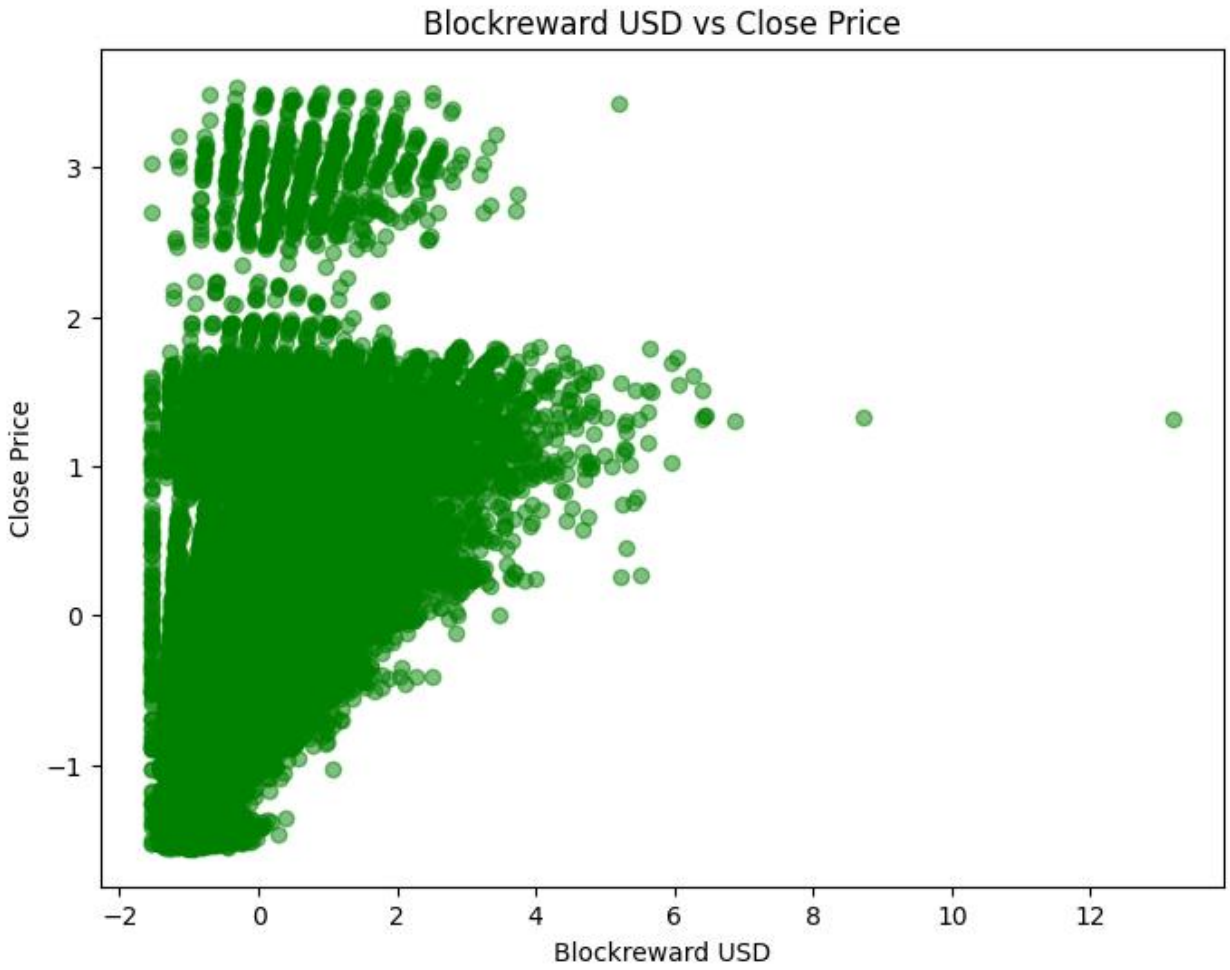
8] ✓ 0.4s Pytho

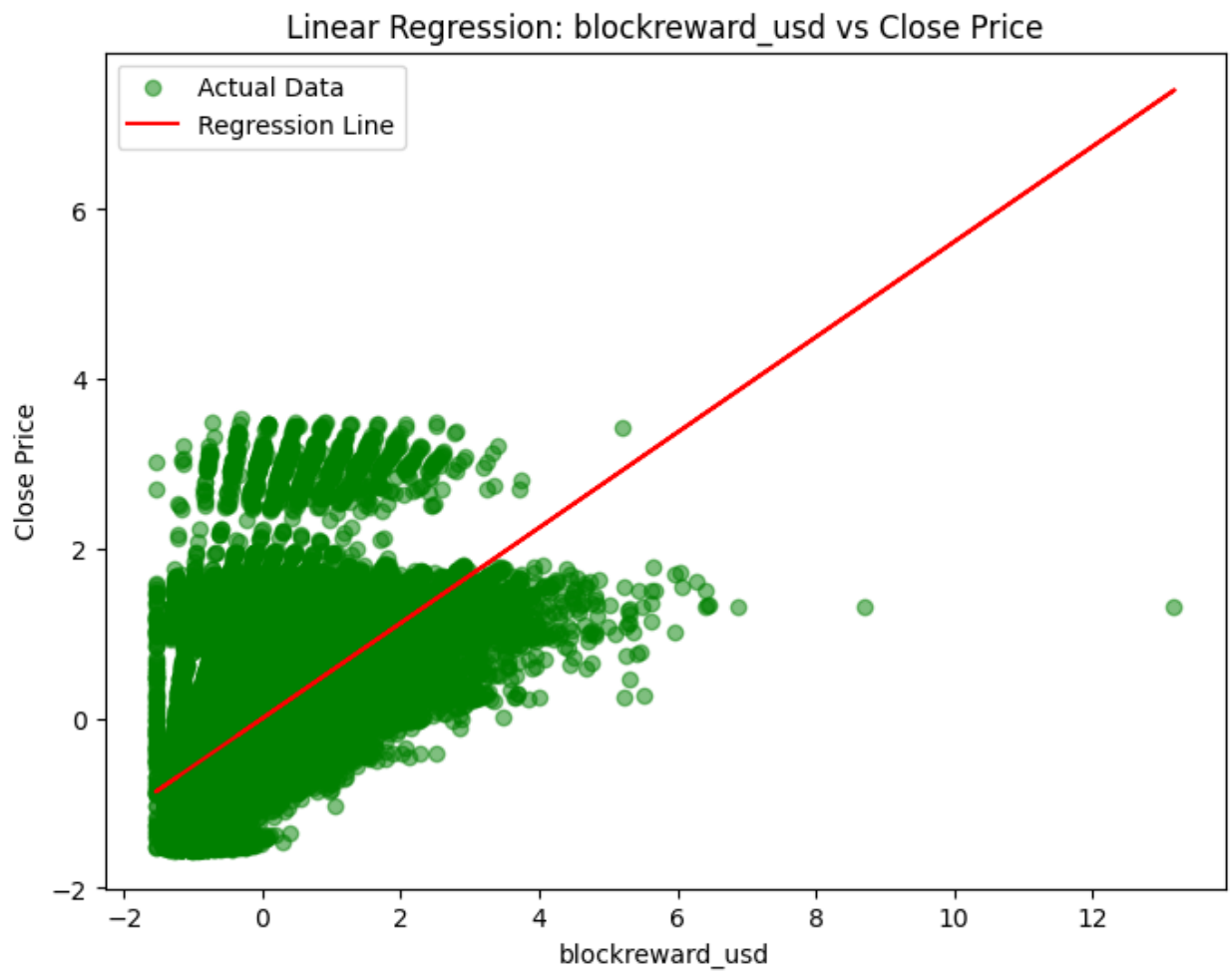
```
Intercept: 1.2261517316895244e-16
Coefficient for Difficulty: 0.29731396249275255
```

```
plt.ylabel('Close Price')
plt.legend()
plt.show()
```

8] ✓ 0.4s Pytho

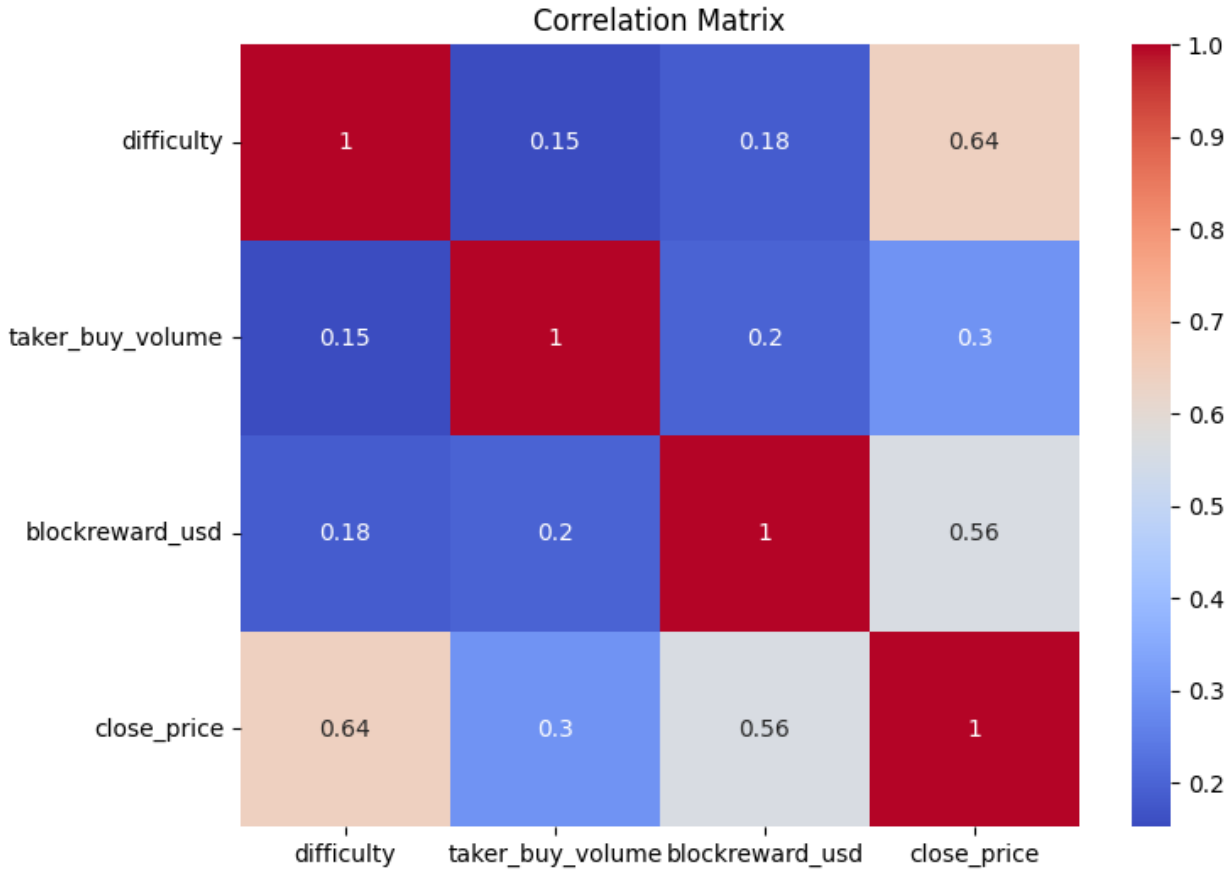
```
Intercept: 1.2261517316895244e-16
Coefficient for Difficulty: 0.29731396249275255
```


Blockreward USD vs Close Price
Close Price
Blockreward USD
3
2
1
0
−1
−2
0
2
4
6
8
10
12




```
from sklearn.linear_model import LinearRegression ···

Intercept: -3.050243341677137e-17
Coefficient for Difficulty: 0.561912158545363
```

Summary of Correlation and Regression Analysis




```python
X = finance_data[['difficulty', 'taker_buy_volume', 'blockreward_usd']]
y = finance_data['close_price']

model = LinearRegression()
model.fit(X, y)

print("Intercept:", model.intercept_)
print("Coefficients:")
feature_names = ['difficulty', 'taker_buy_volume', 'blockreward_usd']
for name, coef in zip(feature_names, model.coef_):
    print(f"{name}: {coef:.4f}")

from sklearn.metrics import r2_score
r2 = r2_score(y, model.predict(X))
print("R-squared:", r2)
```

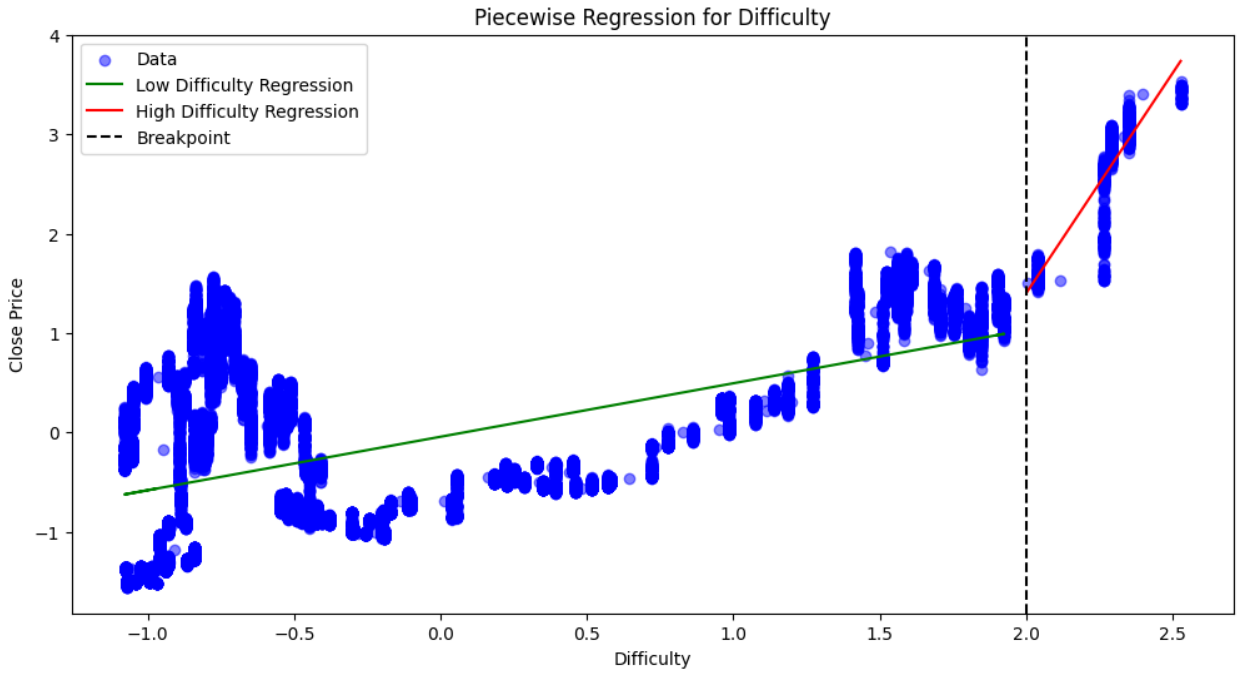
[194] ✓ 0.0s Python

```
Intercept: 8.906484039439459e-17
Coefficients:
difficulty: 0.5395
taker_buy_volume: 0.1295
blockreward_usd: 0.4371
R-squared: 0.6293173742265427
```

Threshold and Breakpoint Analysis

Difficult Variable

```
Difficulty Threshold Analysis:
Above Threshold Avg Close Price: 2.5129945861629803
Below Threshold Avg Close Price: -0.08527752208376452
```

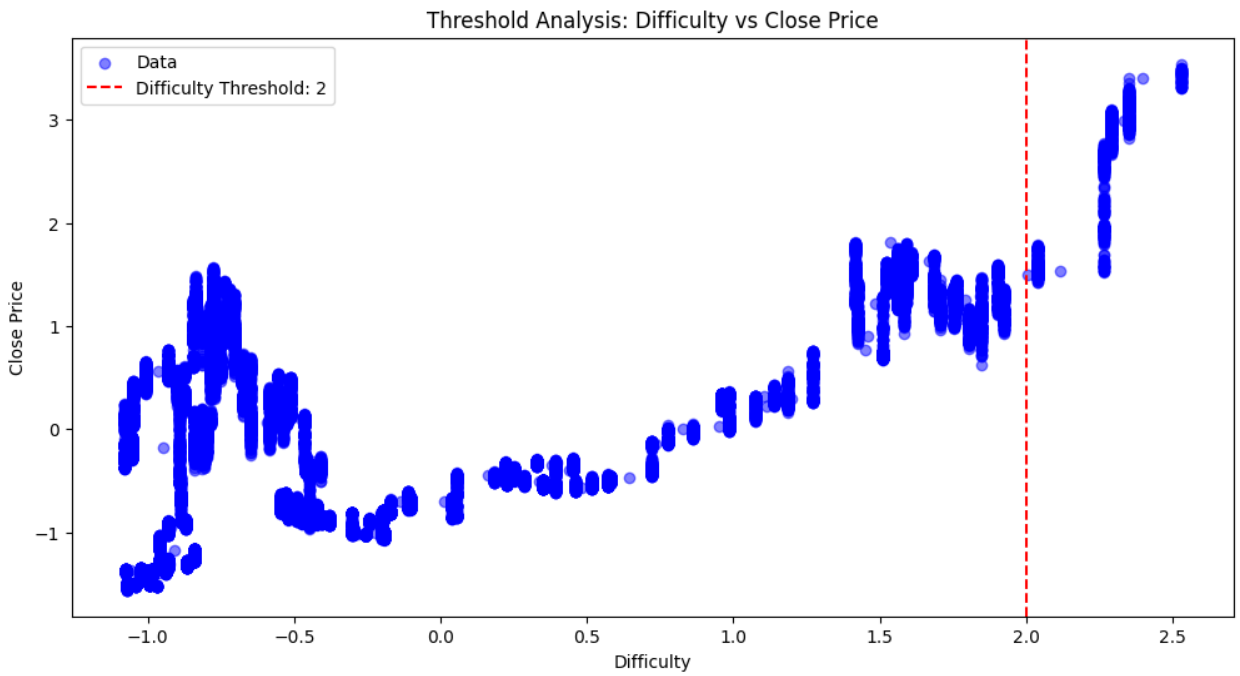



```
Low Difficulty Segment Coefficient: 0.5383137590972802
High Difficulty Segment Coefficient: 4.431418319946328
```

```python
from ruptures import Binseg

signal = finance_data[['difficulty', 'close_price']].to_numpy()

model = Binseg(model="l2").fit(signal)
breakpoints = model.predict(n_bkps=2)

print("Detected Breakpoints:", breakpoints)
```

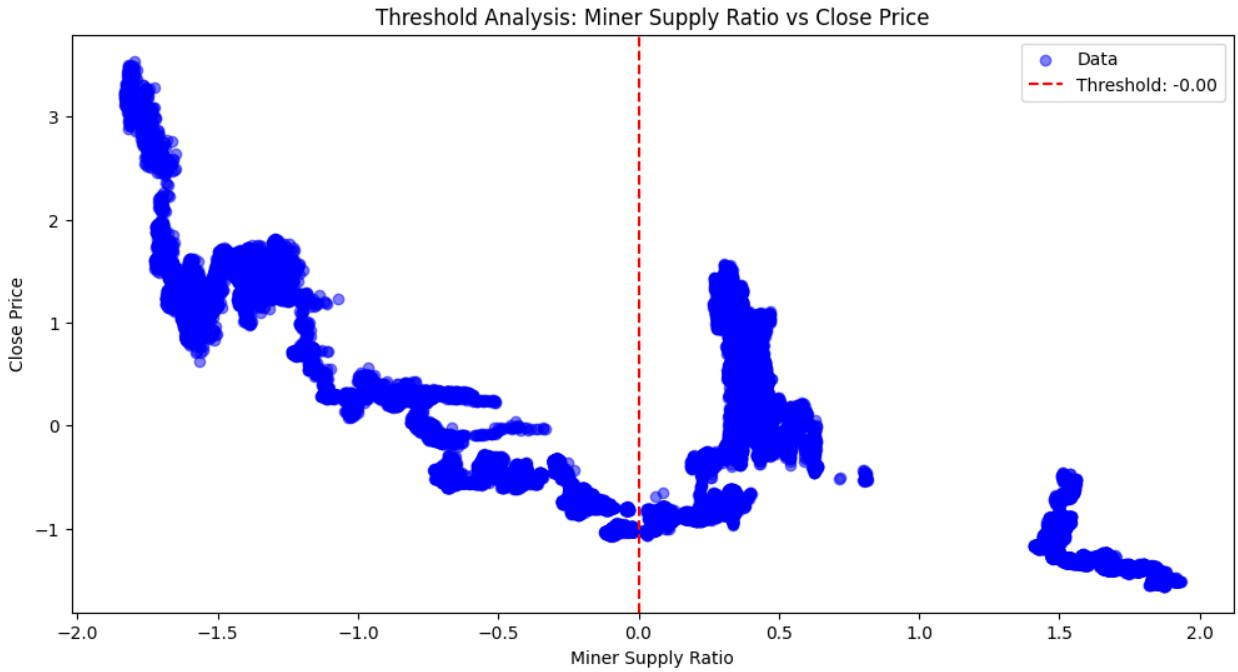
[207] ✓ 7.5s

```
Detected Breakpoints: [6630, 32175, 41498]
```

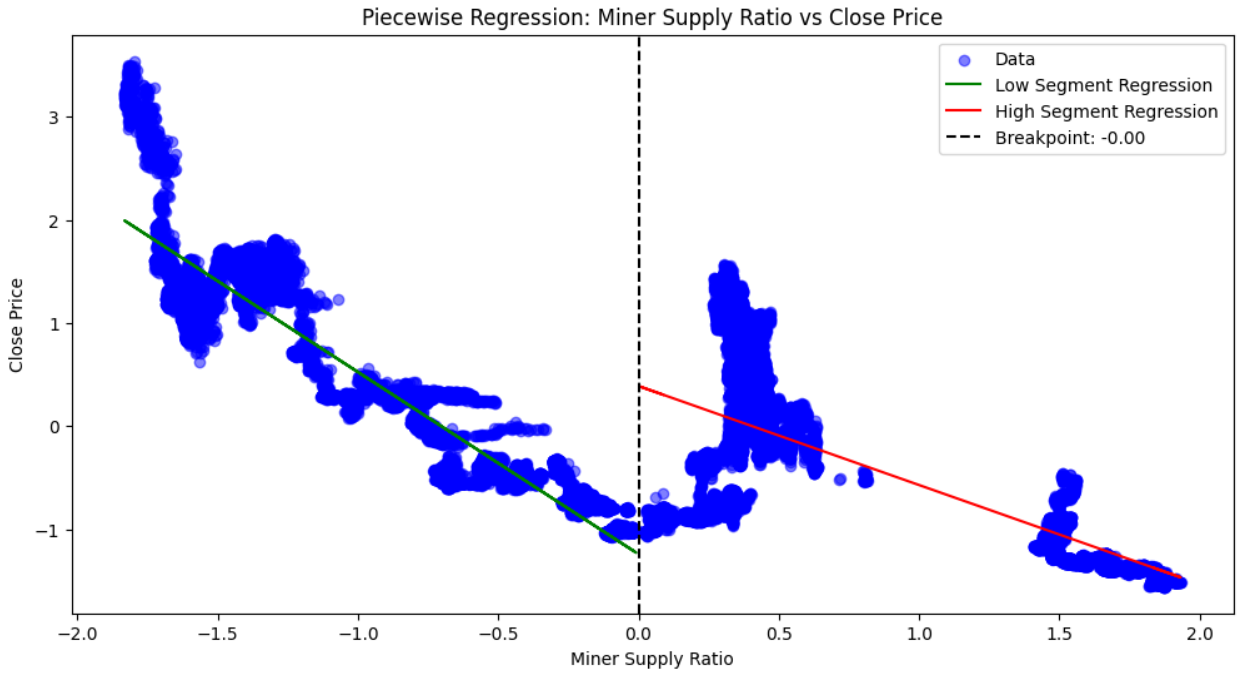


miner_supply_ratio

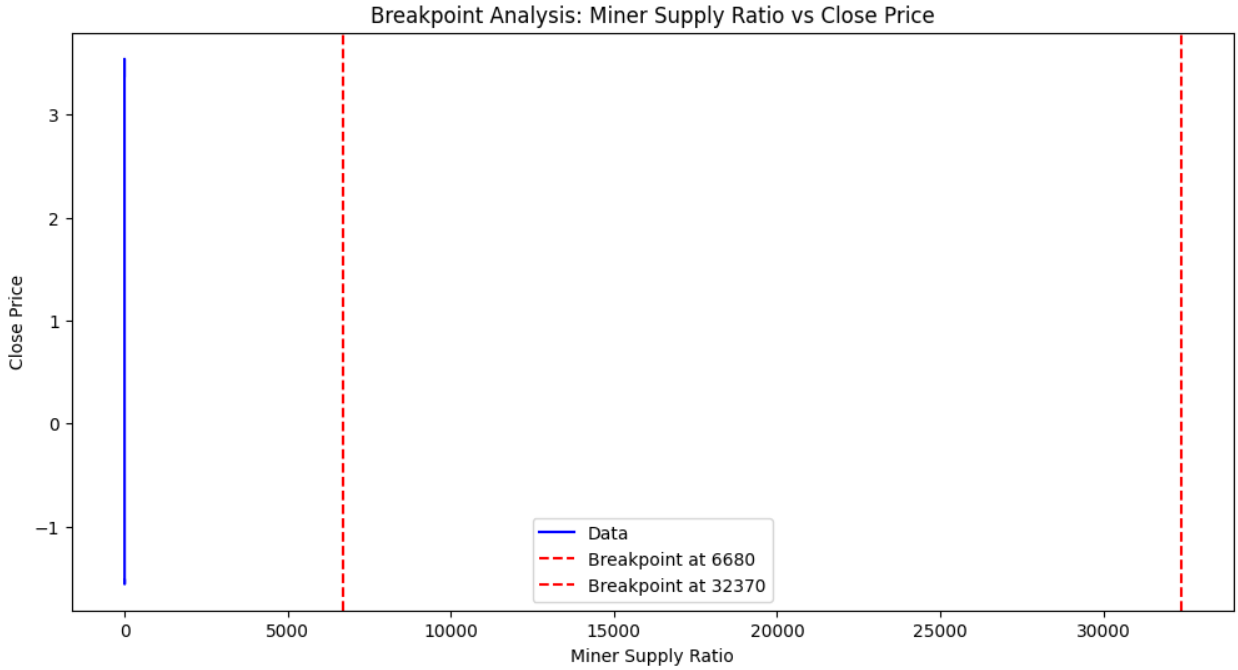
```
Miner Supply Ratio Threshold Analysis:
Above Threshold Avg Close Price: -0.298746599633926
Below Threshold Avg Close Price: 0.38265311409635167
```



```
Low Segment Coefficient: -1.7643911041462281
High Segment Coefficient: -0.9591967816637255
```




```
Detected Breakpoints: [6680, 32370, 41498]
```



Predictive Modeling

Sample Dataset (3 variables with 1 target variable) 80% / 20%

```
selected_data = finance_data[['difficulty', 'taker_buy_volume', 'blockreward_usd', 'close_price']].dropna()

X = selected_data[['difficulty', 'taker_buy_volume', 'blockreward_usd']]
y = selected_data['close_price']

#Split into training and testing sets (80% training, 20% testing)
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

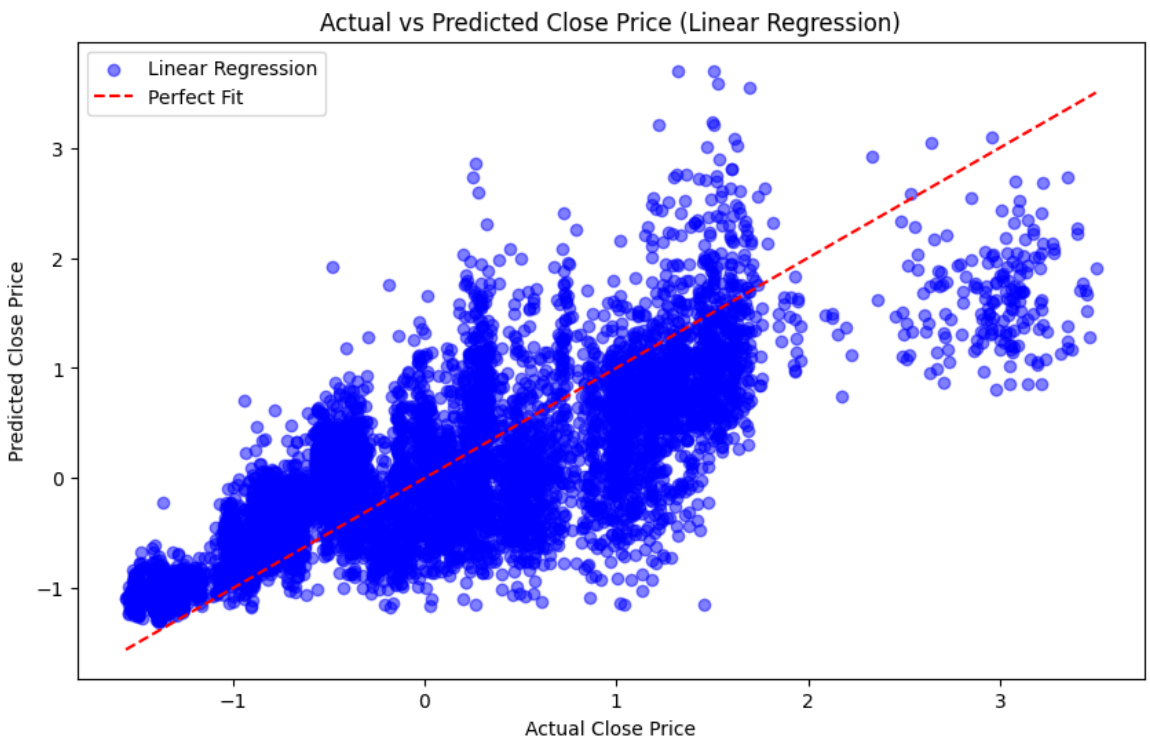
Linear Regression Results:

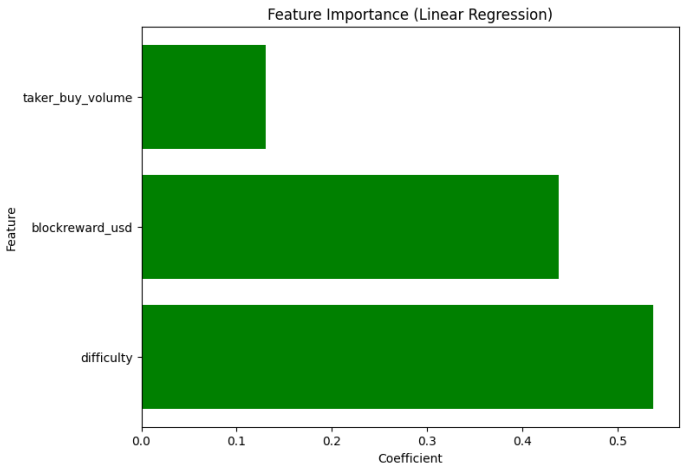
Intercept: -0.000254699597113638

Coefficients: {'difficulty': np.float64(0.5370914951389283), 'taker_buy_volume': np.float64(0.13069739848177586), 'blockreward_usd': np.float64(0.43828848172990287)}

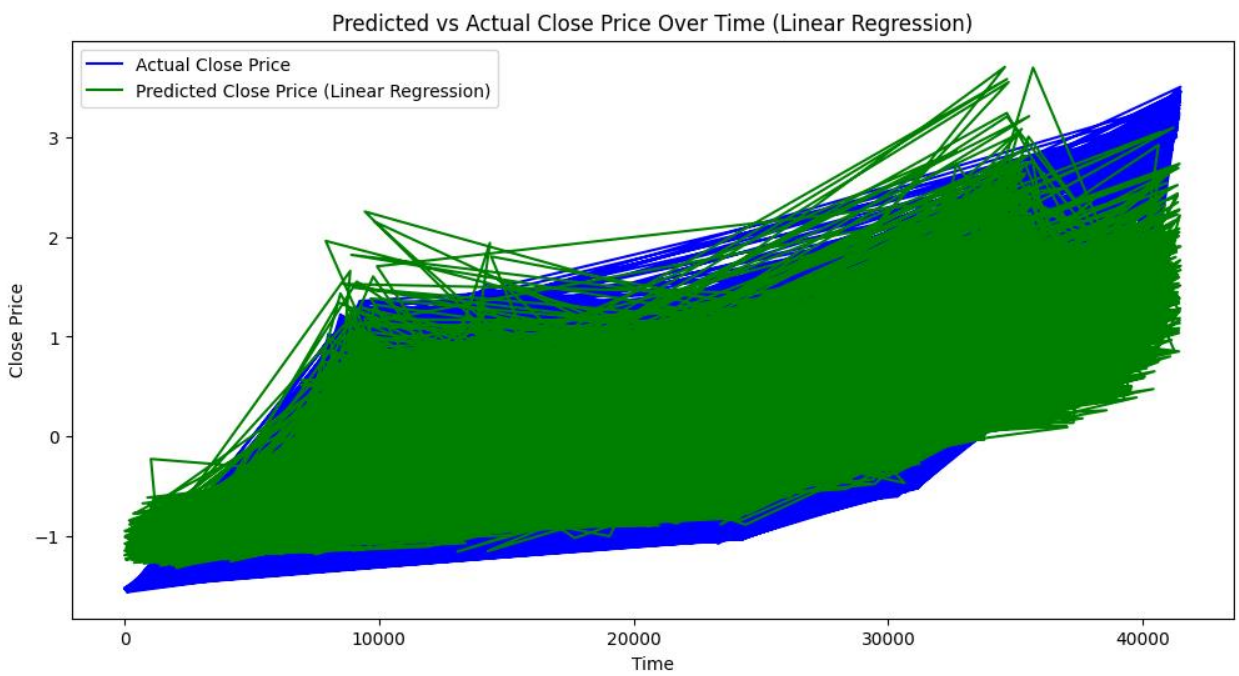
Mean Squared Error (MSE): 0.3757211097171112

R-squared (R^2): 0.6311952914982172





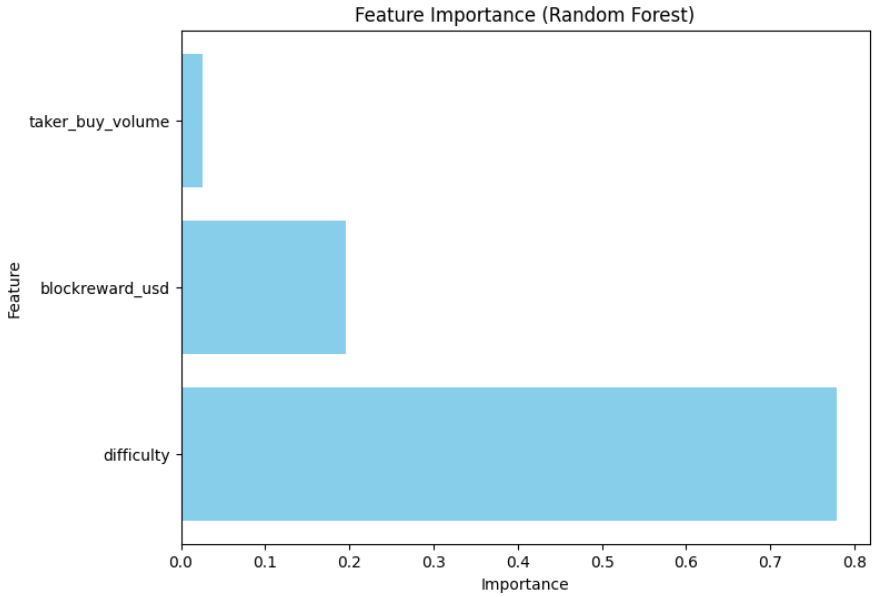
Residual Plot (Linear Regression)
Zero Error
Residuals (Actual - Predicted)
Predicted Close Price
2
1
0
−1
−2
−1
0
1
2
3

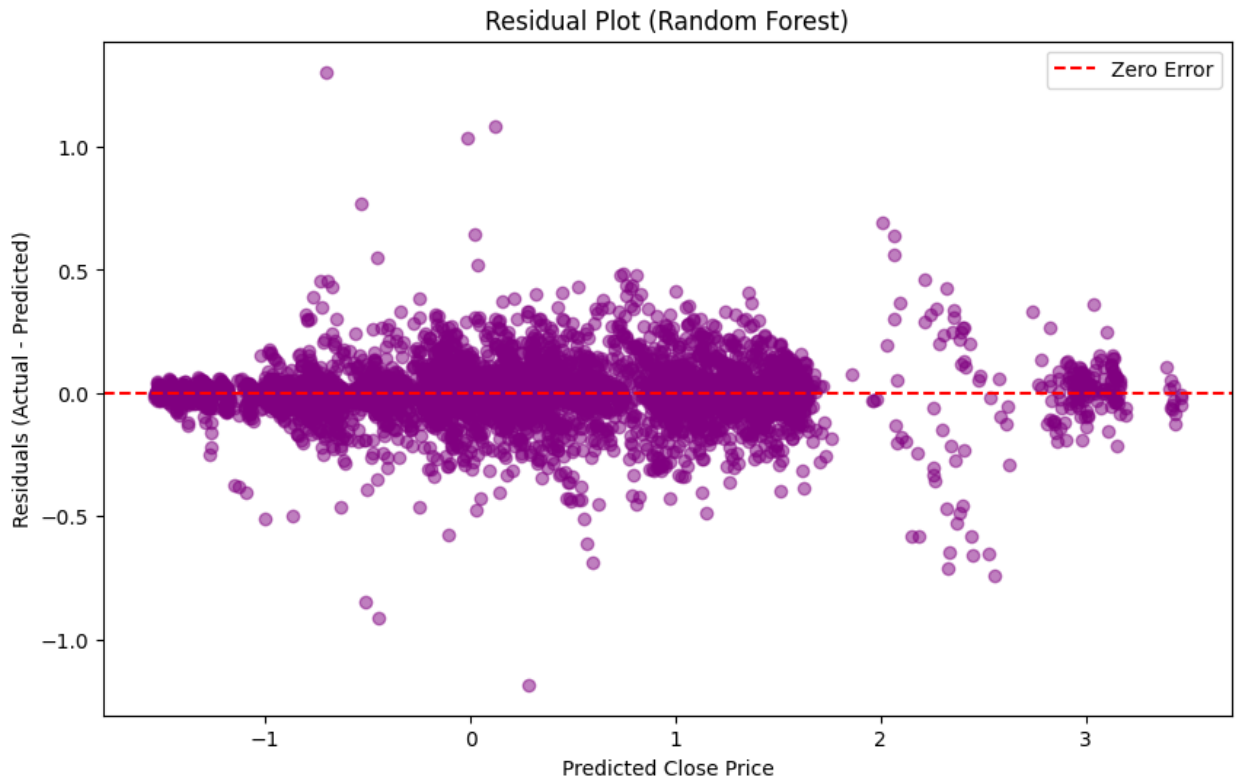

Predicted vs Actual Close Price Over Time (Linear Regression)
Actual Close Price
Predicted Close Price (Linear Regression)
Close Price
Time
3
2
1
0
−1
0
10000
20000
30000
40000

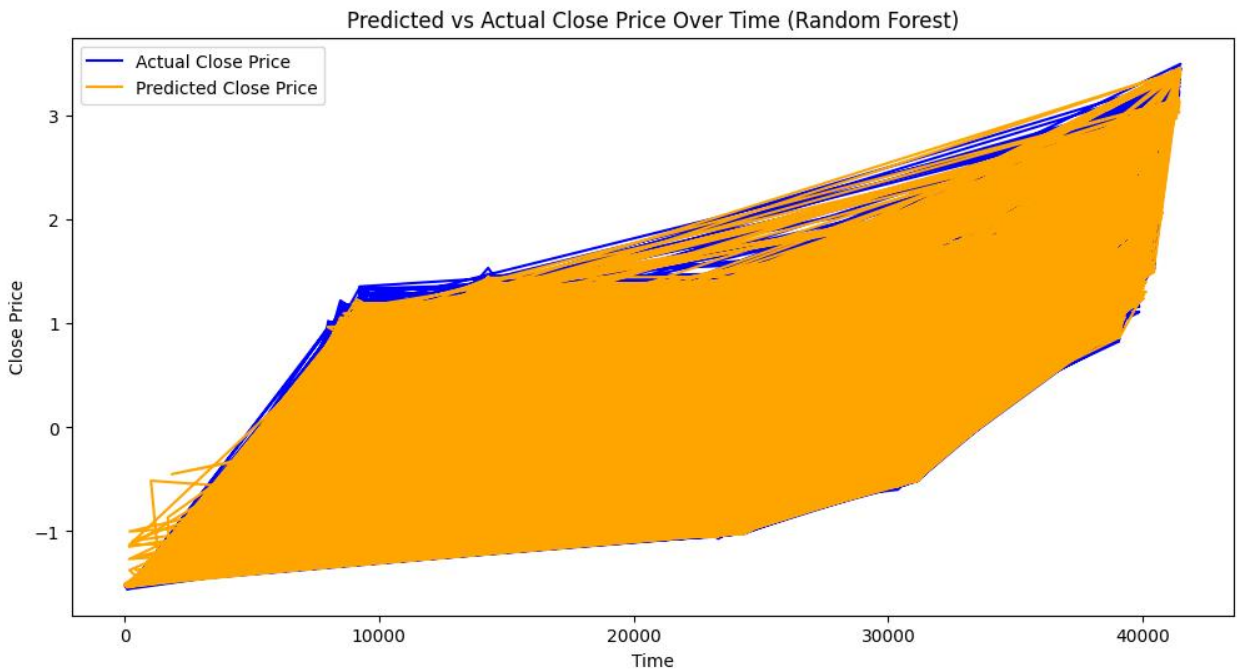
Random Forest Results:

Mean Squared Error (MSE): 0.009400388322936453

R-squared (R^2): 0.9907726572034921



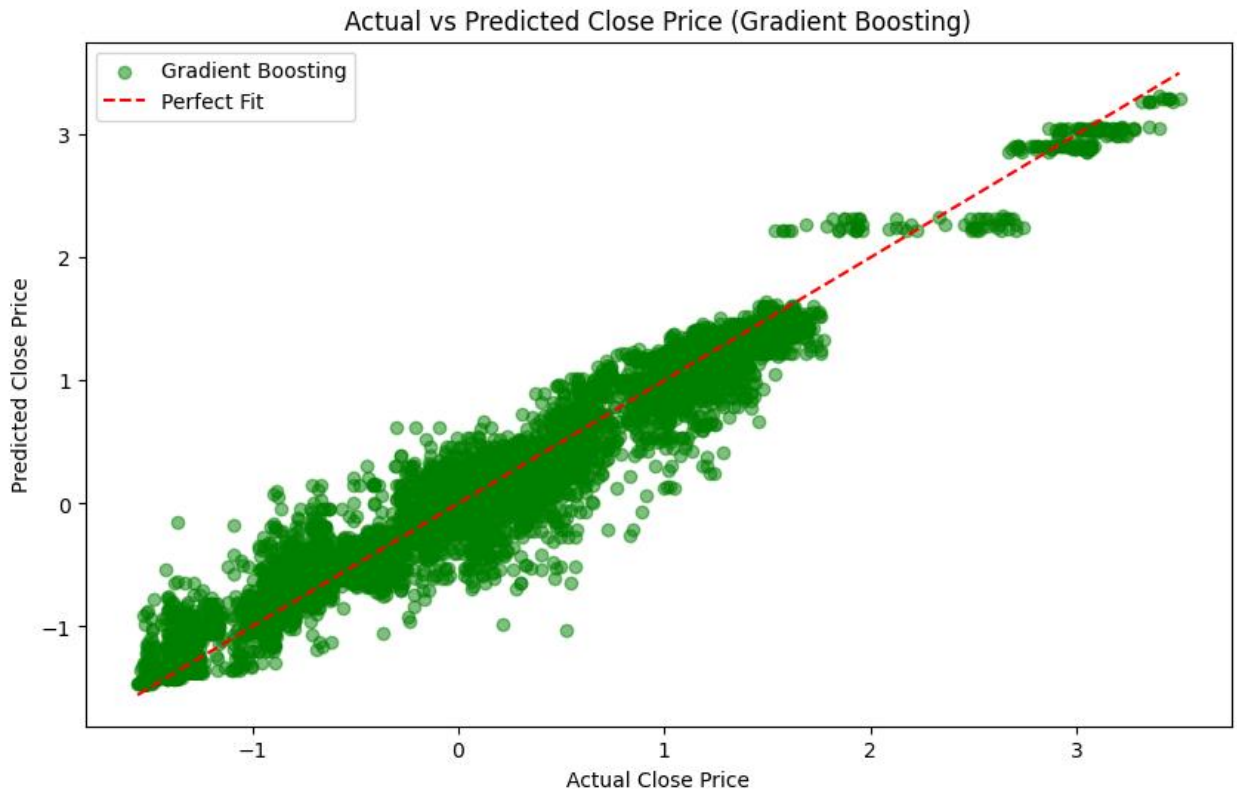

Residual Plot (Random Forest)
Zero Error
Residuals (Actual - Predicted)
Predicted Close Price
1.0
0.5
0.0
−0.5
−1.0
−1
0
1
2
3

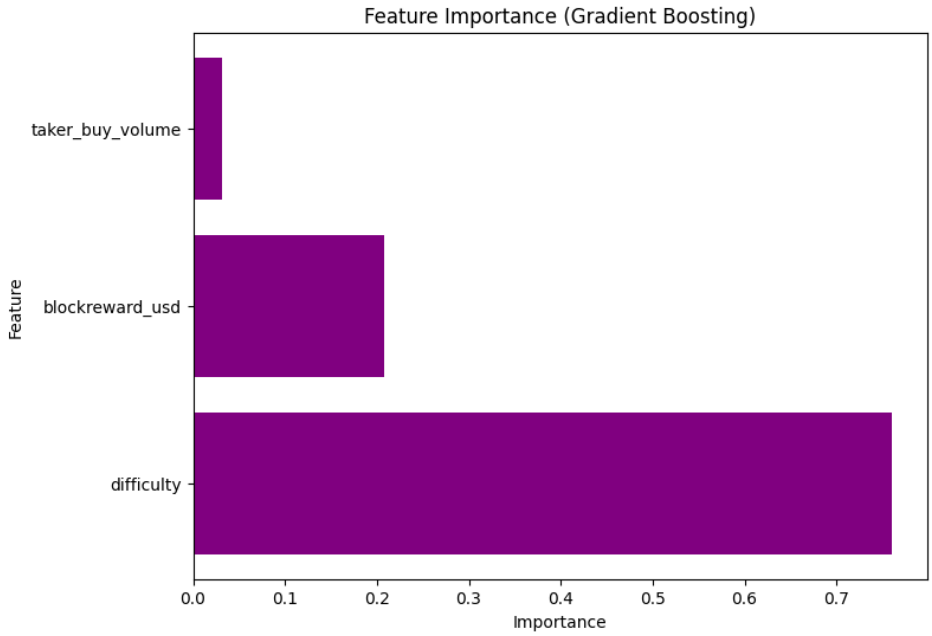

Predicted vs Actual Close Price Over Time (Random Forest)
Actual Close Price
Predicted Close Price
Close Price
Time
3
2
1
0
−1
0
10000
20000
30000
40000

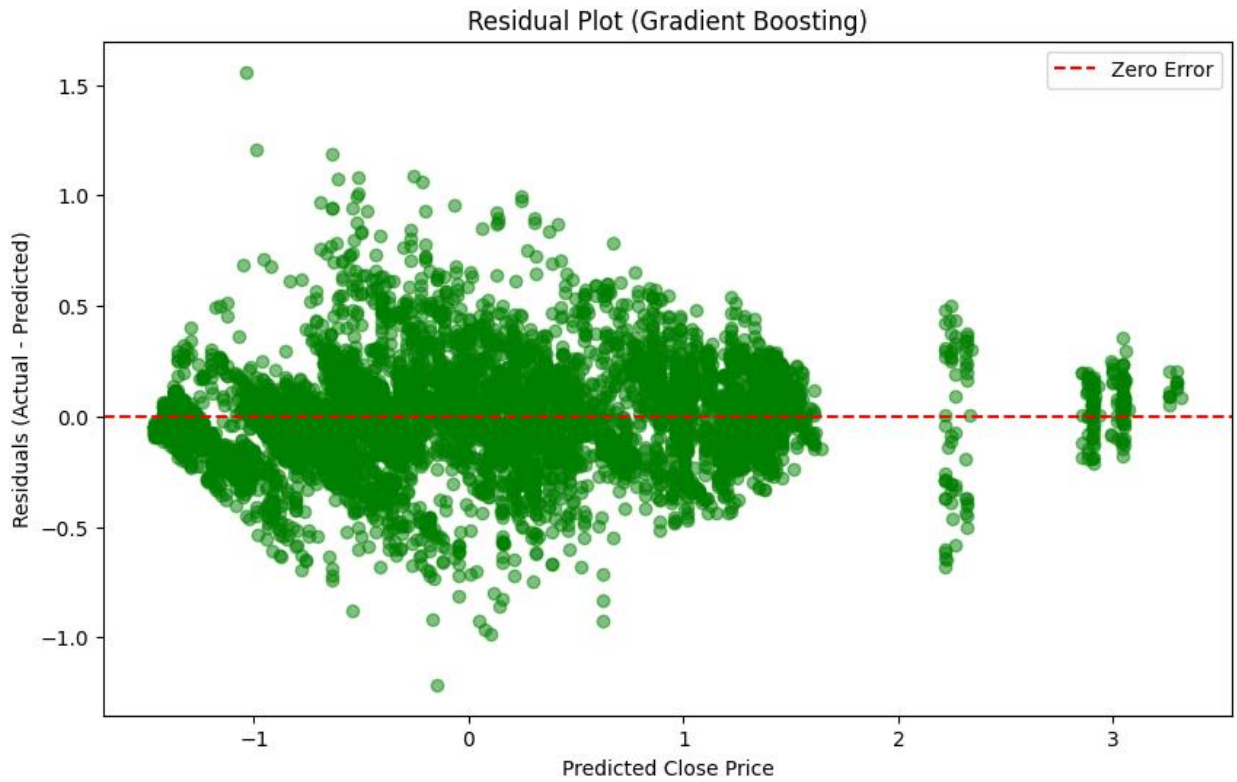
Gradient Boosting Results:

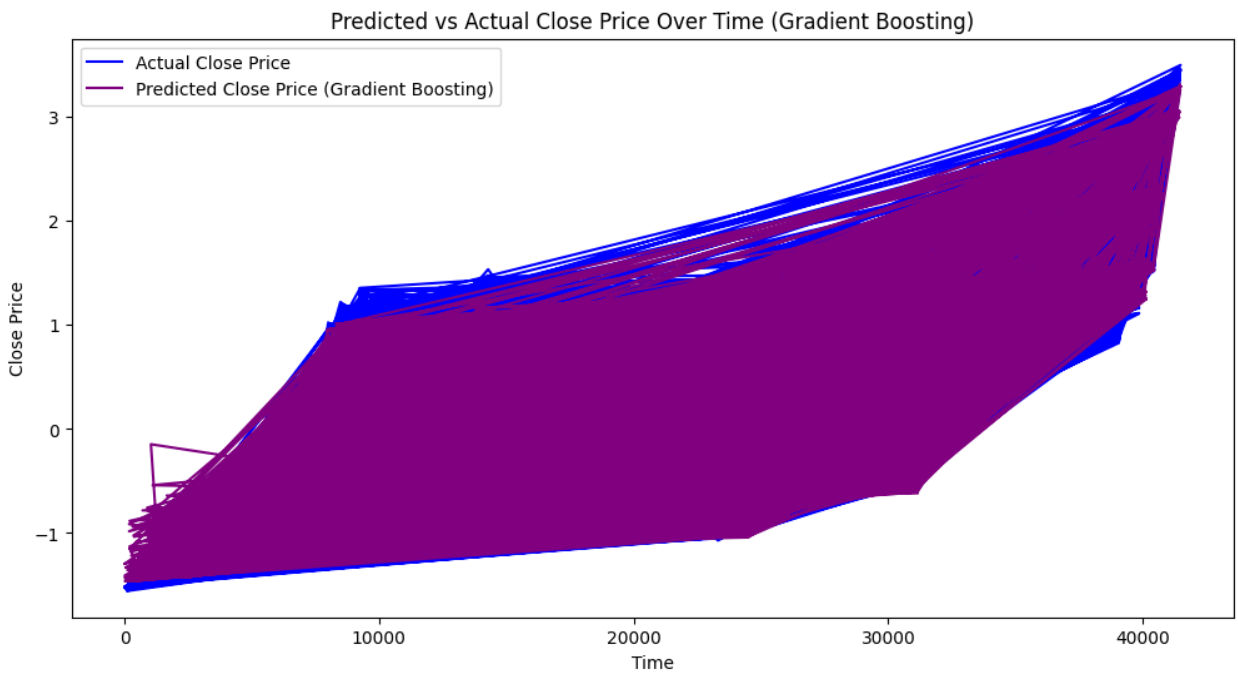
Mean Squared Error (MSE): 0.046626225145128715

R-squared (R^2): 0.9542320861712164

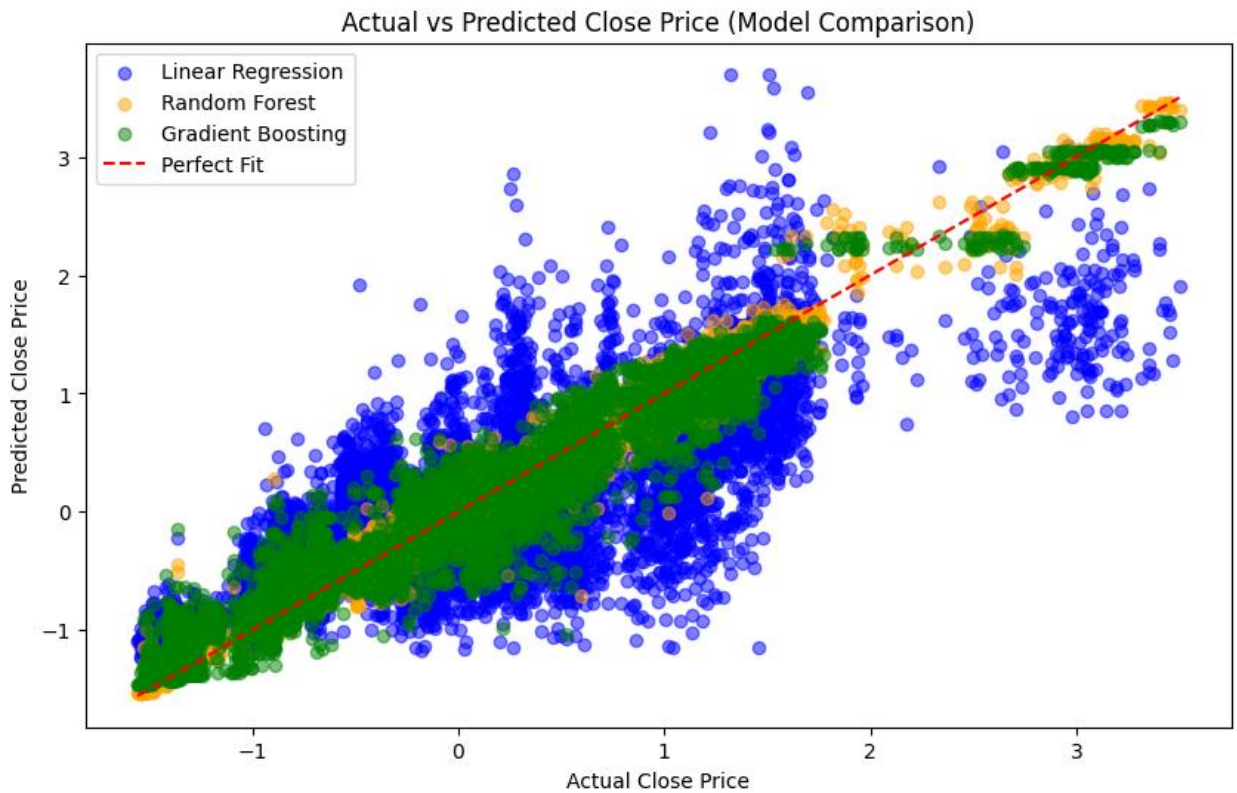


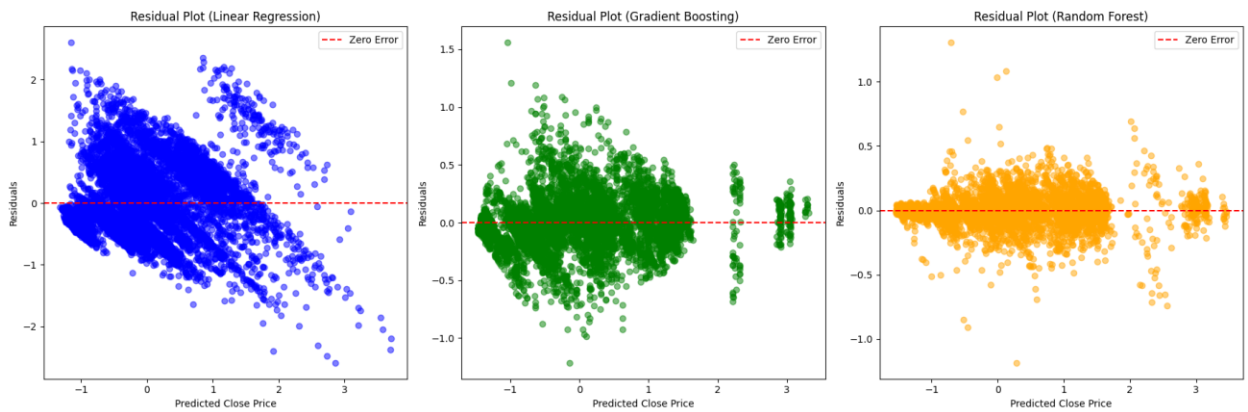


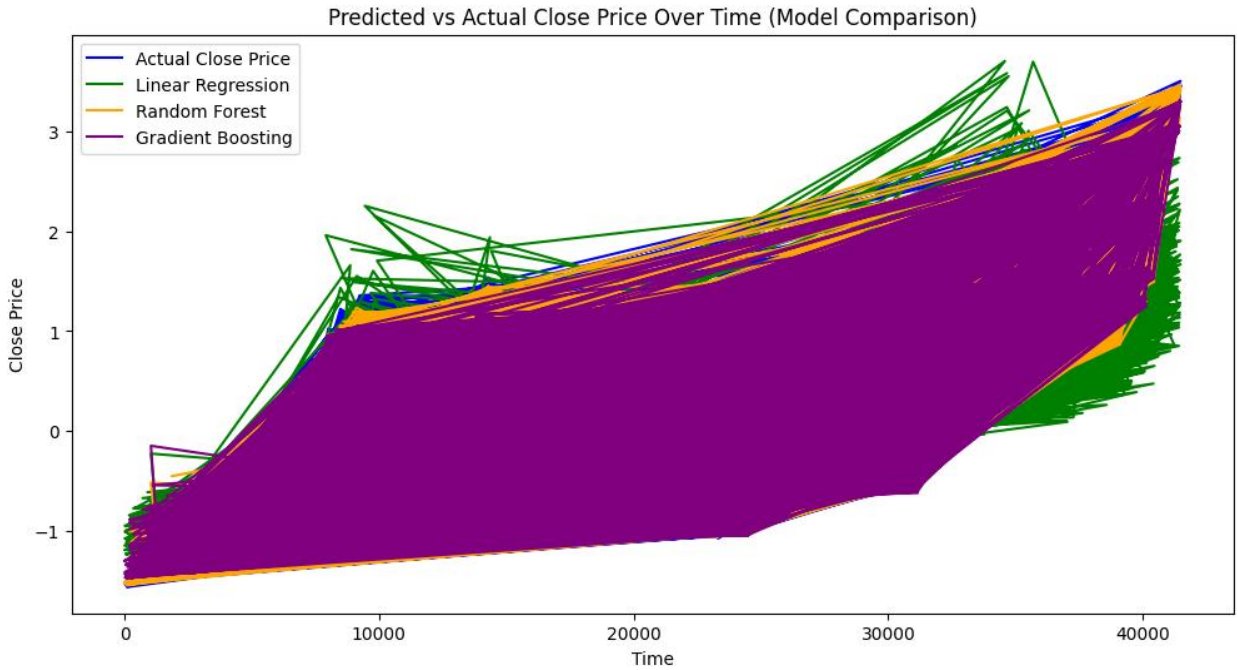

Residual Plot (Gradient Boosting)
Zero Error
Residuals (Actual - Predicted)
Predicted Close Price
1.5
1.0
0.5
0.0
−0.5
−1.0
−1
0
1
2
3


Predicted vs Actual Close Price Over Time (Gradient Boosting)
Actual Close Price
Predicted Close Price (Gradient Boosting)
Close Price
Time
3
2
1
0
−1
0
10000
20000
30000
40000

Summary of Prediction Models (LR & RF & GB)






Predicted vs Actual Close Price Over Time (Model Comparison)
Actual Close Price
Linear Regression
Random Forest
Gradient Boosting
Close Price
Time
3
2
1
0
−1
0
10000
20000
30000
40000