By DataDreamin

BITCOIN CRYPTO ANALYSIS

Research Contents

1. Factors Influencing the Close Price
2. Trading Activity based on Time-Series Analysis
3. ML Model: Risk Analysis
4. ML Model: Predicting Close Price

What affects the close price of the cryptocurrency that we want to purchase?

NEXT ➡

RESEARCH 1

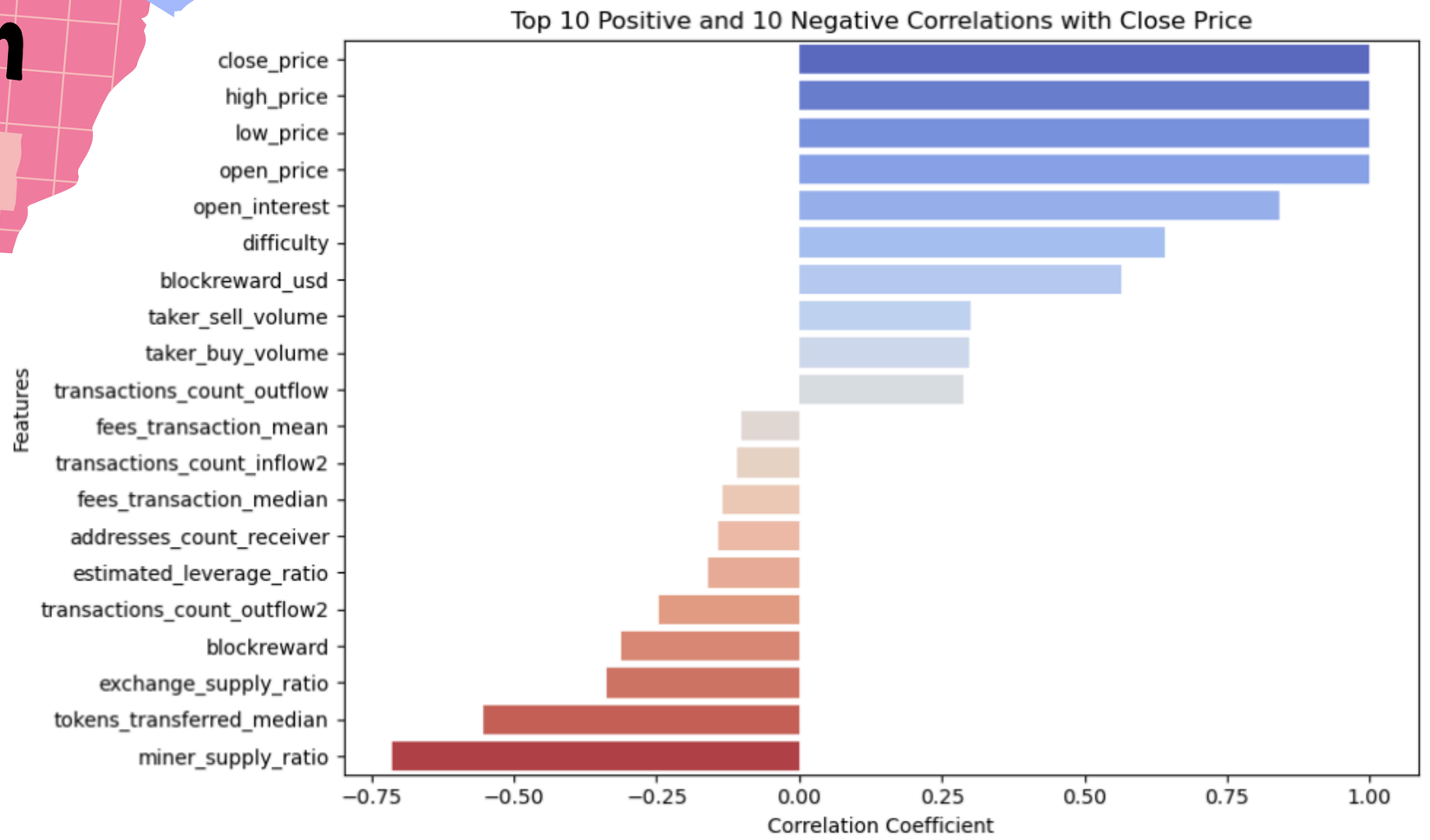
FACTORS INFLUENCING THE CLOSE PRICE

Correlation Matrix

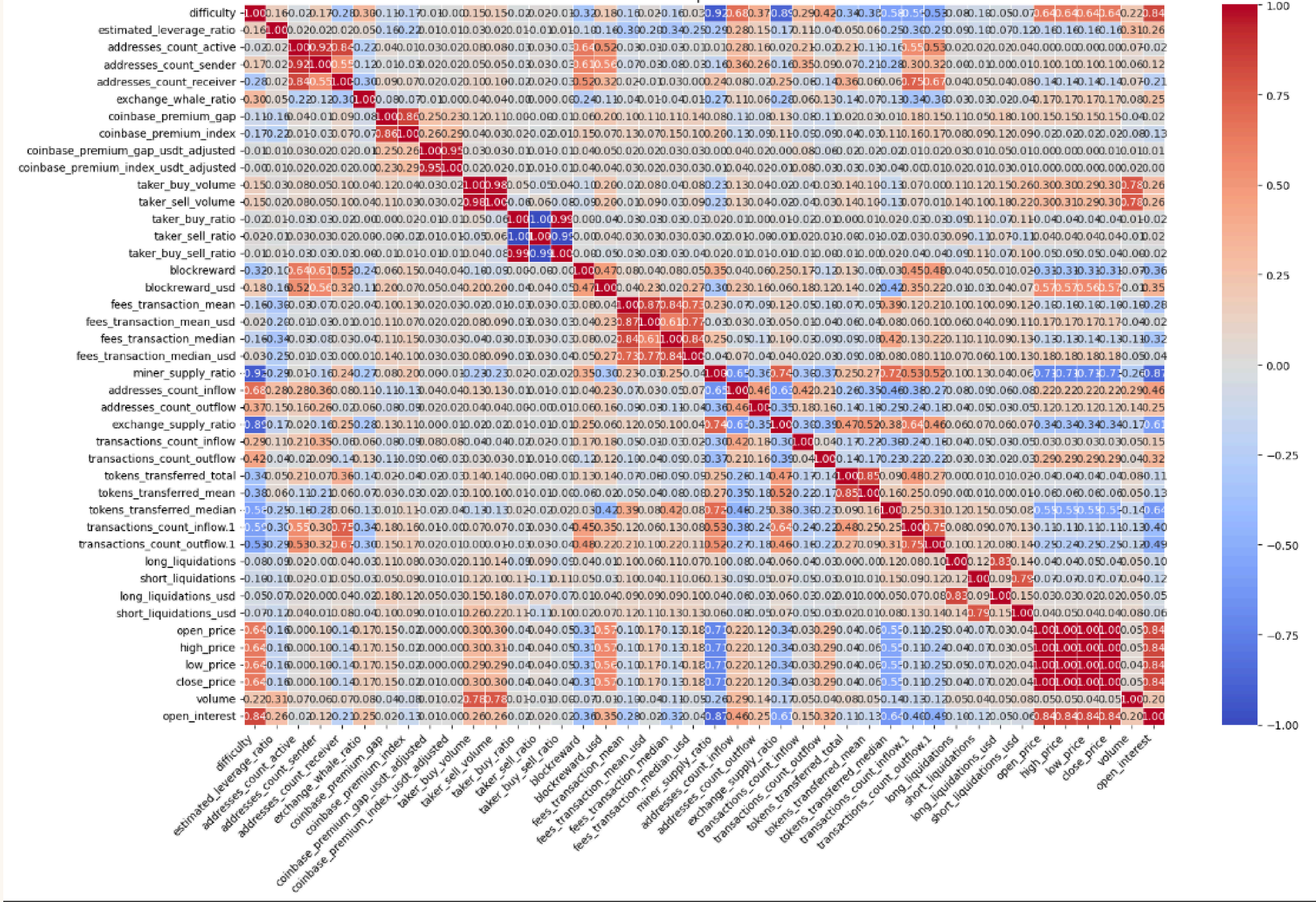
Top Three Features with Highest Correlation:

Open Interest (+)
Difficulty (+)
Miner Supply Ratio (-)

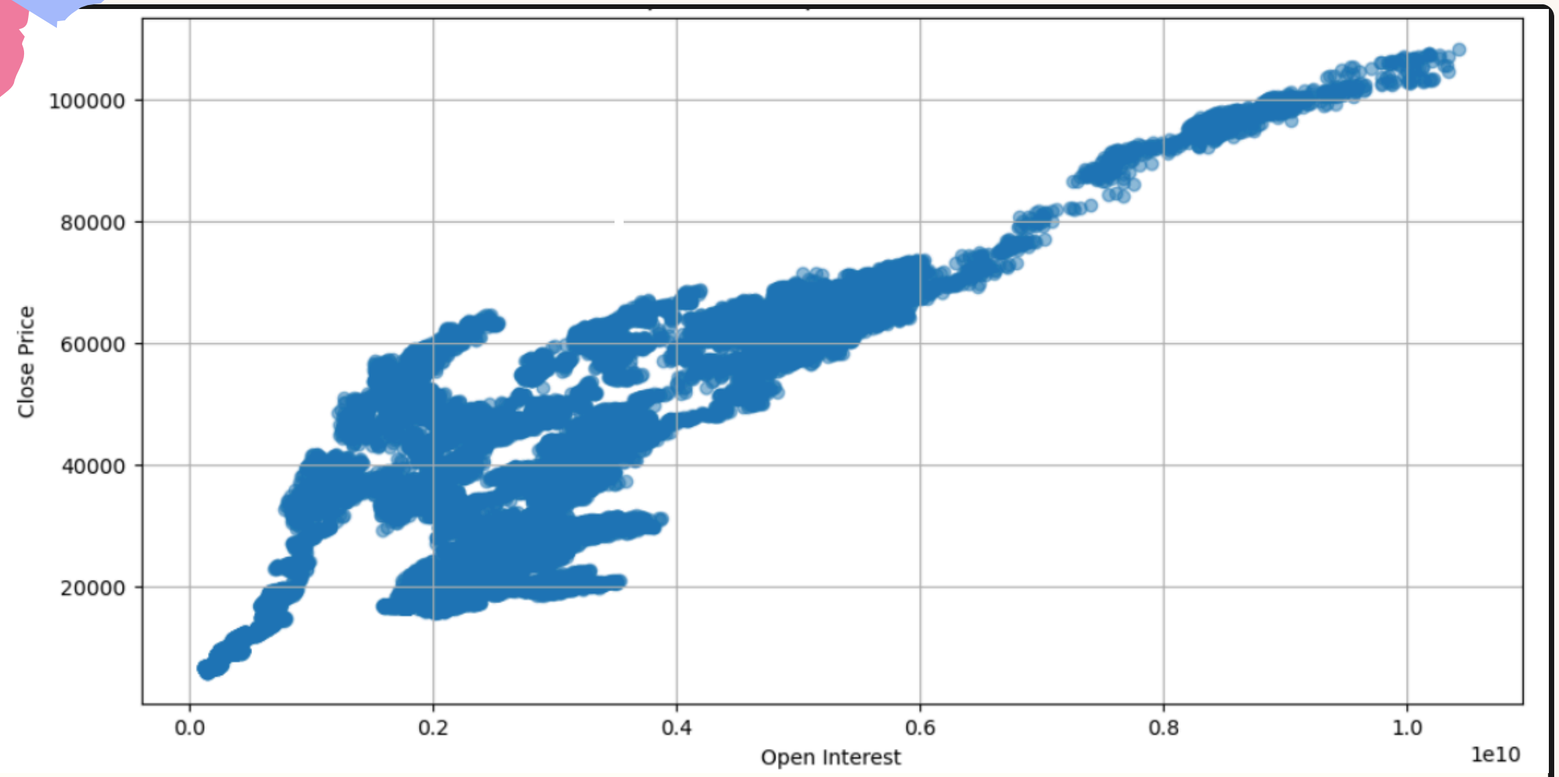
+: Positive Correlation
-: Negative Correlation



Full Correlation Matrix Heatmap



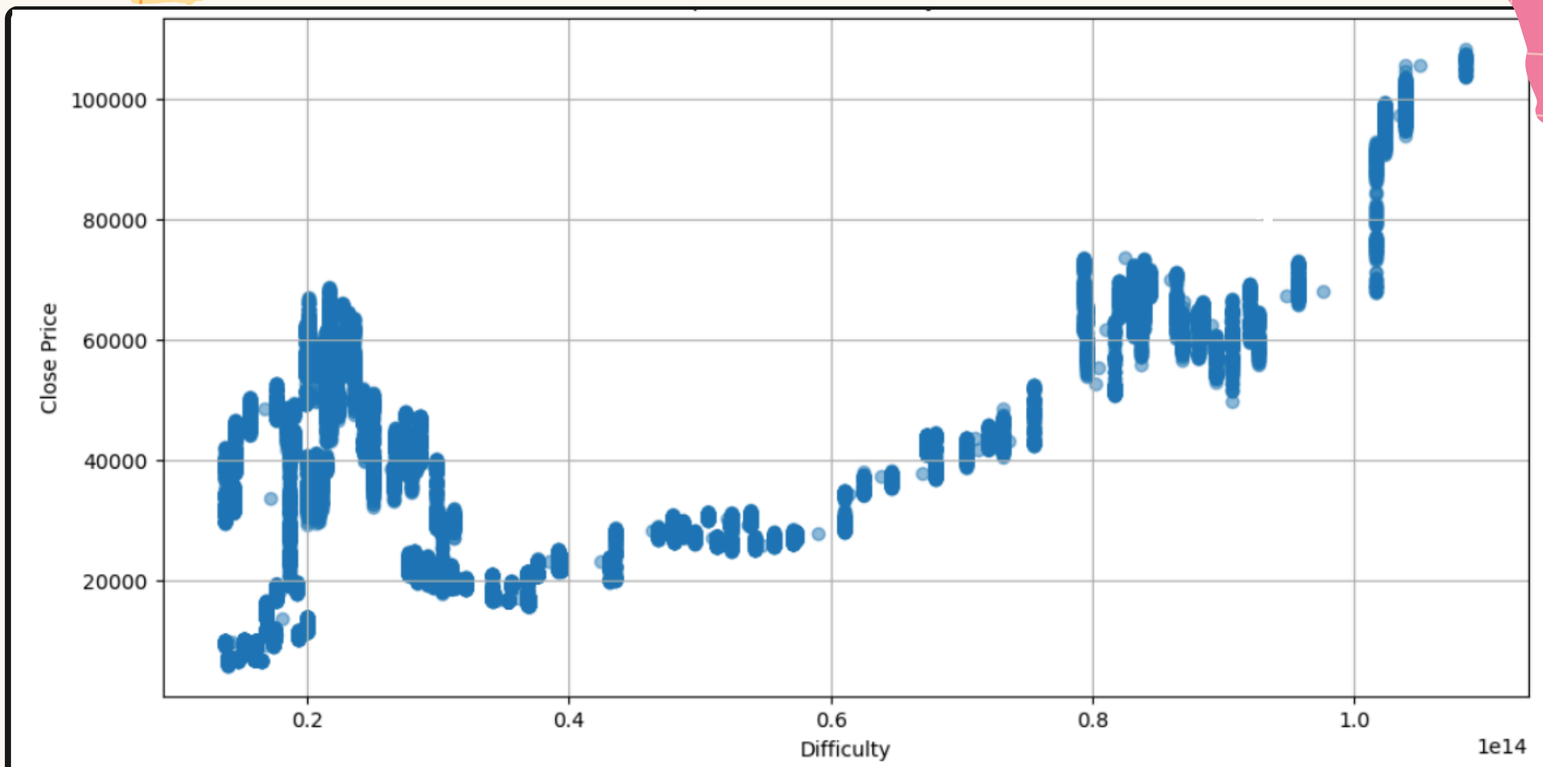
OPEN INTEREST



Open interest possesses a very strong positive correlation with the close price.

Using this information, investors can strategize when to buy, sell, or hold their crypto based on the current interest rate.

DIFFICULTY

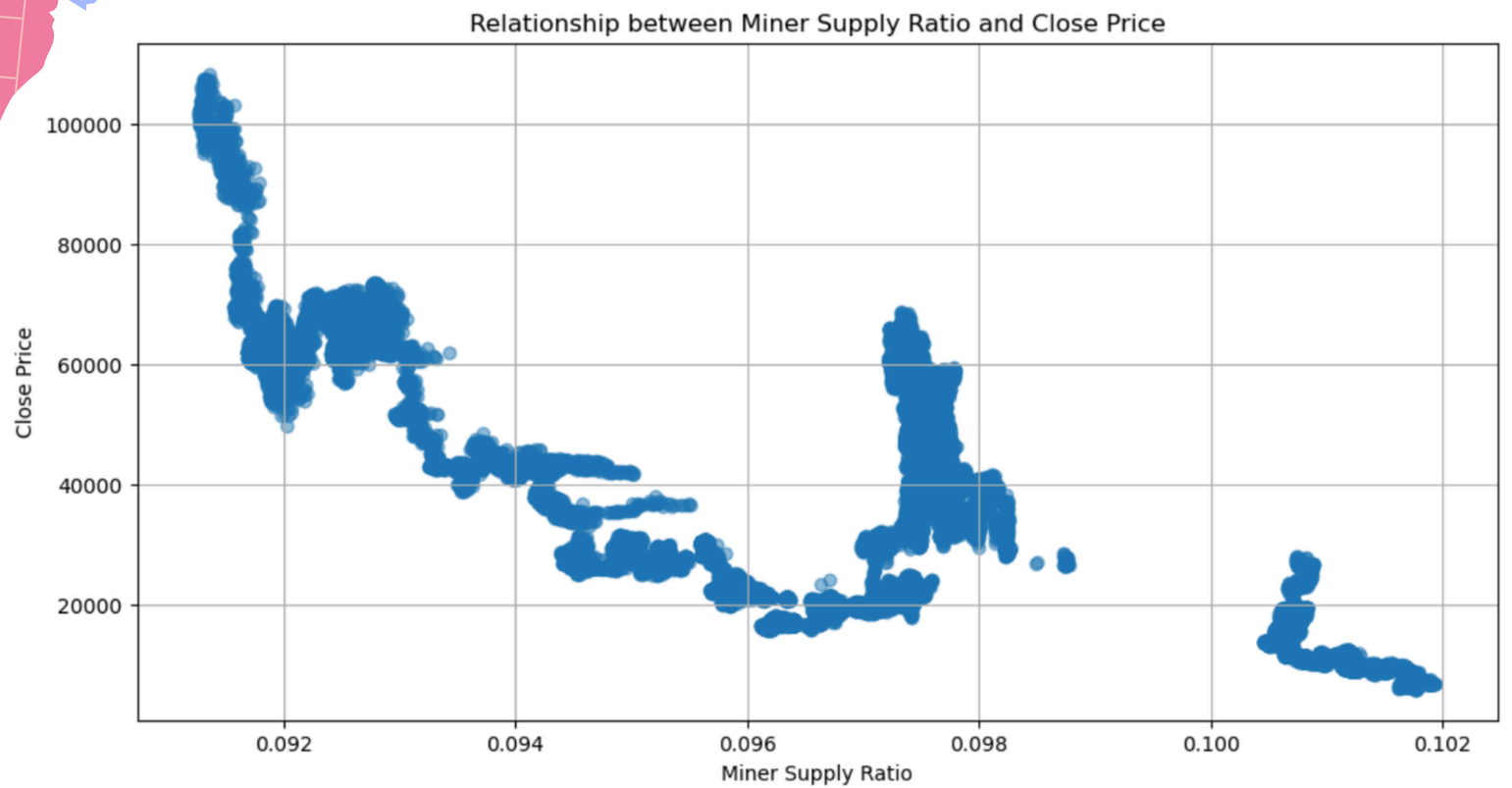


Although not as strong as open interest, this metric still forms a moderately strong positive correlation with the close price.

The more difficult it is to mine for crypto, the higher the close price.

Therefore, investors should be aware and proceed with their purchase when miners are having an easier time digging for cryptos.

MINER SUPPLY RATIO



The slight curve shape that is formed indicates the strong negative correlation between the miner supply ratio and the close price. This implies that crypto should be purchased when the supply of miners are high.

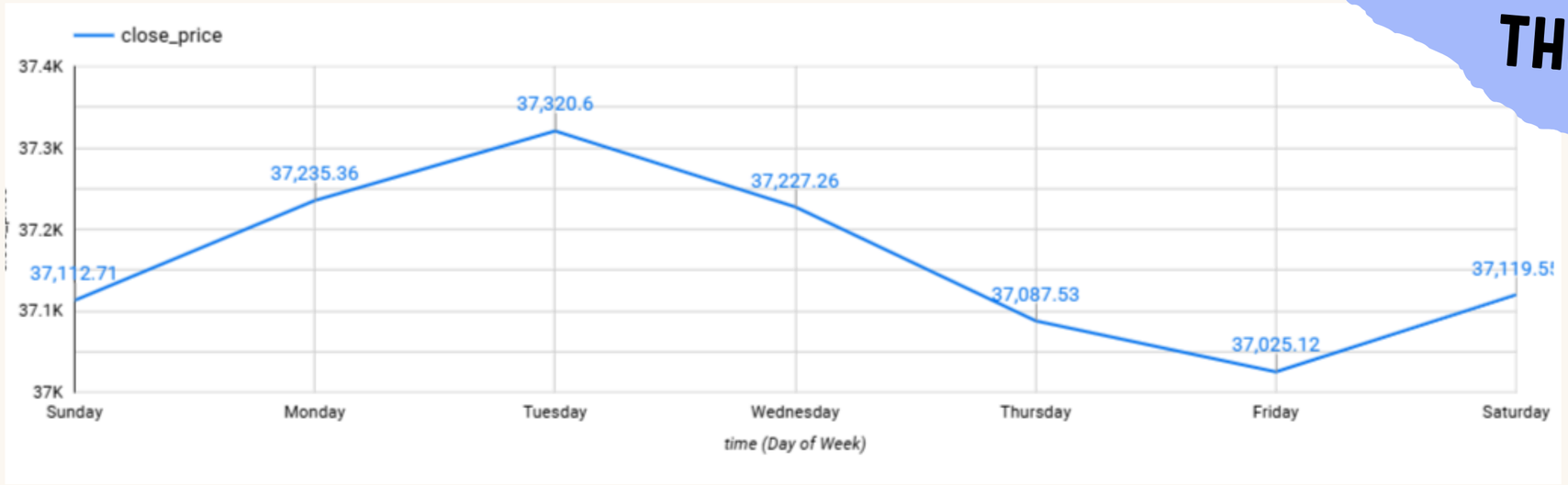
How do we know when is the best and worst time to invest in cryptocurrency?

NEXT ➡

RESEARCH 2

TRADING ACTIVITY BASED ON TIME-SERIES ANALYSIS

AVERAGE CLOSE PRICE PER DAY OF THE WEEK



From the graph, we can see that the highest average close price is on Tuesday which indicates that the market is busy. On the contrary, Friday shows the lowest average close price which could be a result of inactive investors who are taking a break towards the weekend.

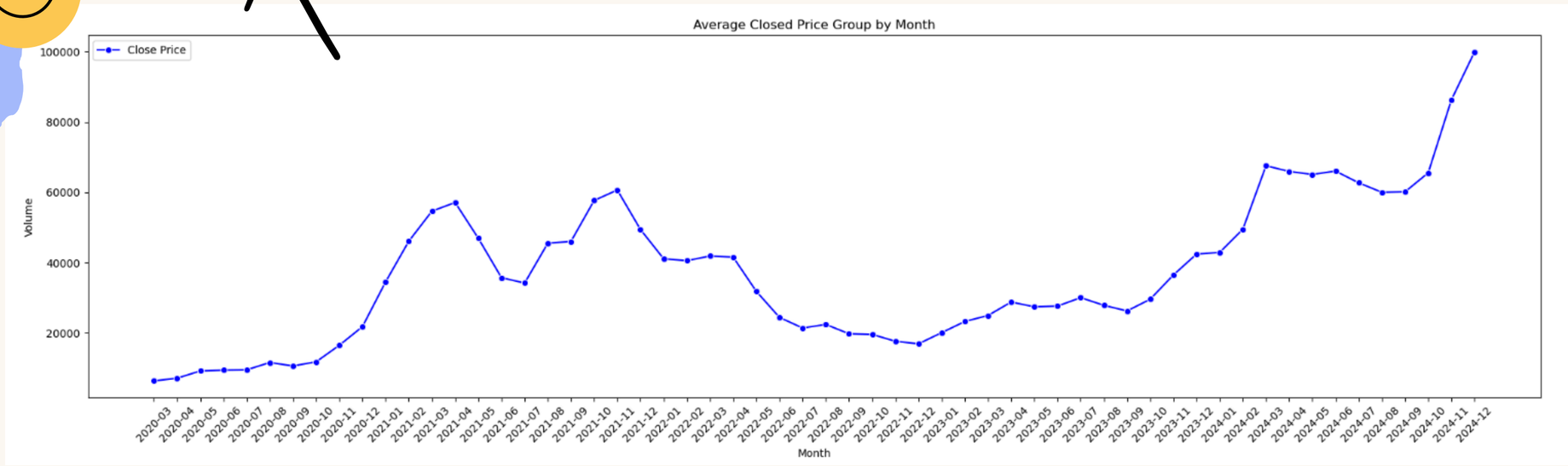
AVERAGE CLOSE PRICE PER DAY OF THE MONTH

In this visualization, it is observed that the close price gets higher towards the middle of the month. However, the trend dissipates drastically after approaching that period. This could be explained by the timing of paychecks in Malaysia. As we get our wages towards the end of the month, we tend to invest aggressively at the start of the new month as our capital has increased. This tight competitions lead to the close prices getting higher and higher. The prices drop towards the end of the month, as the competition is less among investors who do not have a strong capital anymore after investing in the earlier period.

AVERAGE CLOSE PRICE PER MONTH

As large organizations often release their annual earning reports in mid-January, their investors will be more active in the cryptocurrency market. This can be seen in the drastic increase of close price from January to March. In the other end of the year, the festive season brings optimism and hope among investors. They tend to buy cryptocurrencies for multiple reasons such as extra income during the holidays and also to be given as gifts for Christmas.

AVERAGE CLOSE PRICE PER MONTH (OVERALL)



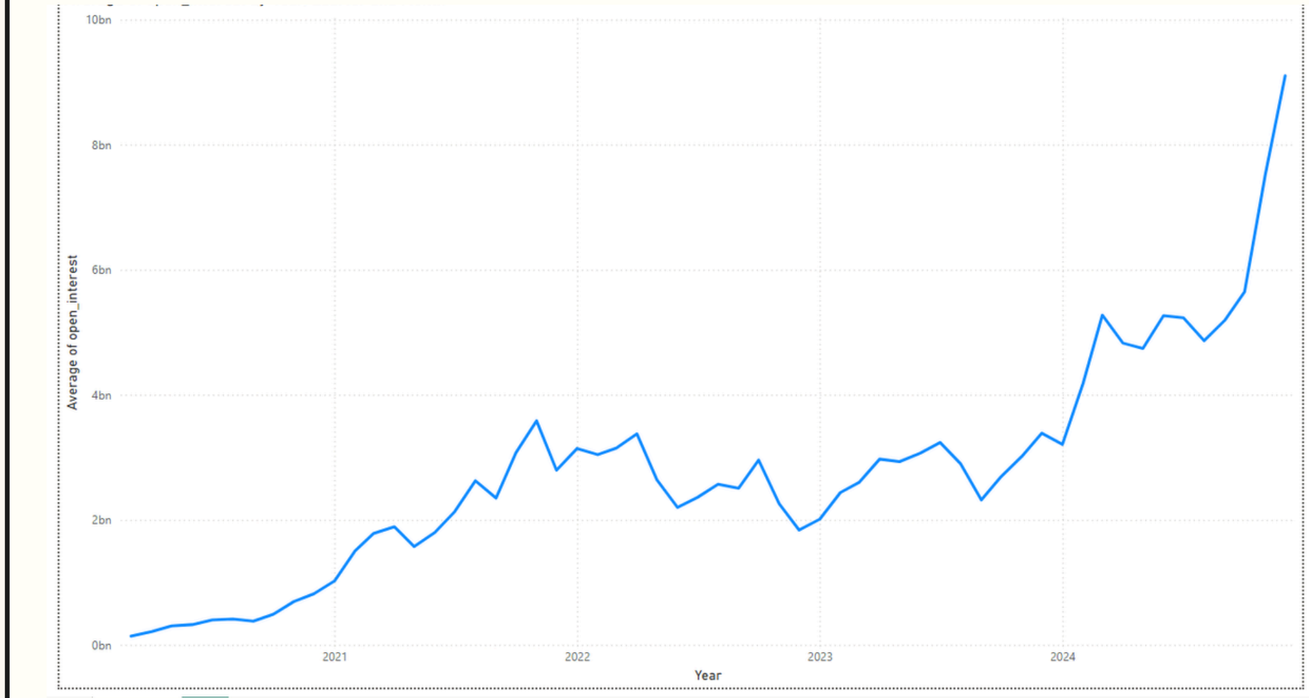
In the span of five years from 2020 to 2024, we can observe two spikes in 2021 and an extreme spike in close price towards the end of 2024.

2021 Spikes - As this period was during the global COVID-19 pandemic, many investors had more time to invest at home during lockdown, which led to higher prices of cryptocurrency

End of 2024 - Recent election results in the USA have heavily influenced the price of cryptocurrency, as President Donald Trump's victory promised a bright future in Bitcioin

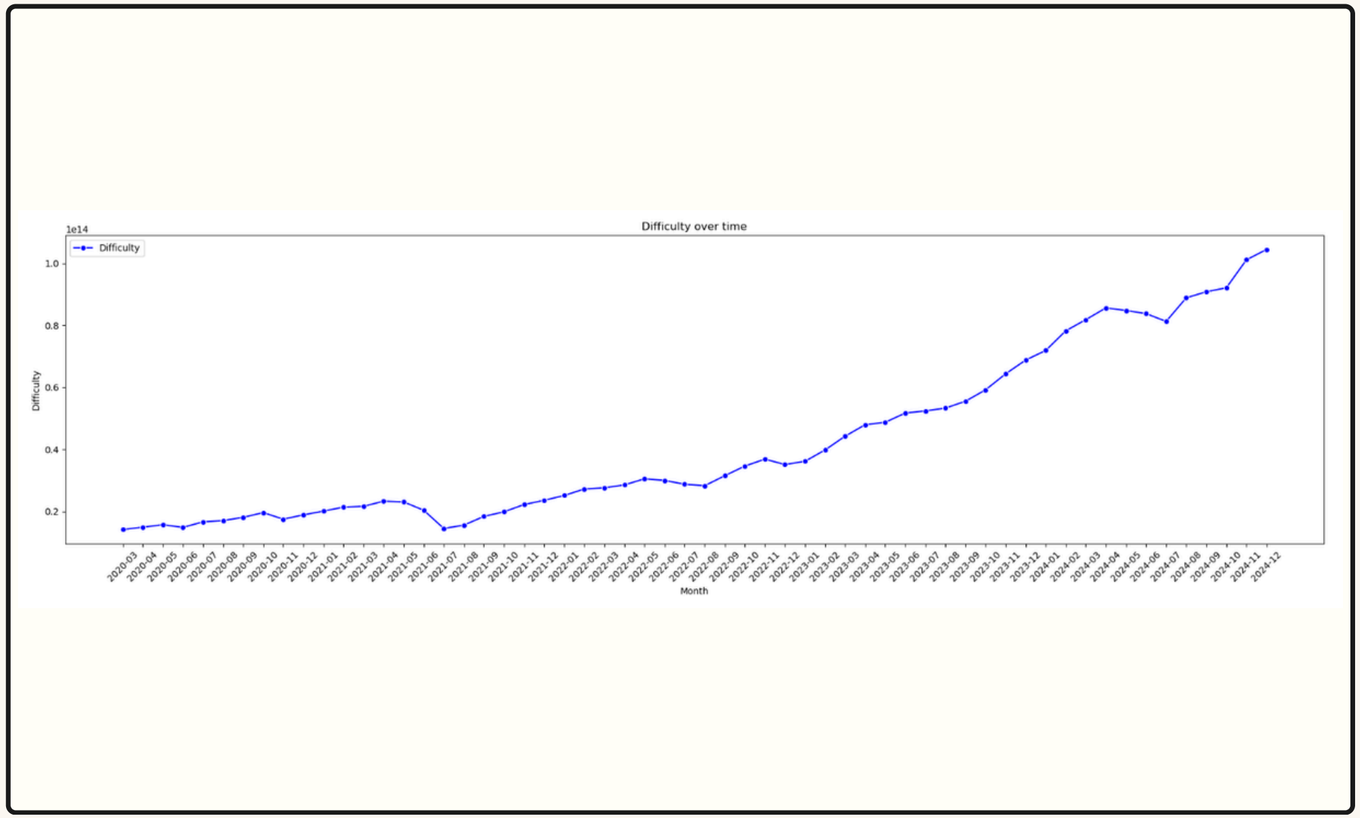
Open Interest against Time

Gradually increasing values show that more and more people are investing in cryptocurrency over time. This trend is expected to continue in coming years as the demand is increasing.



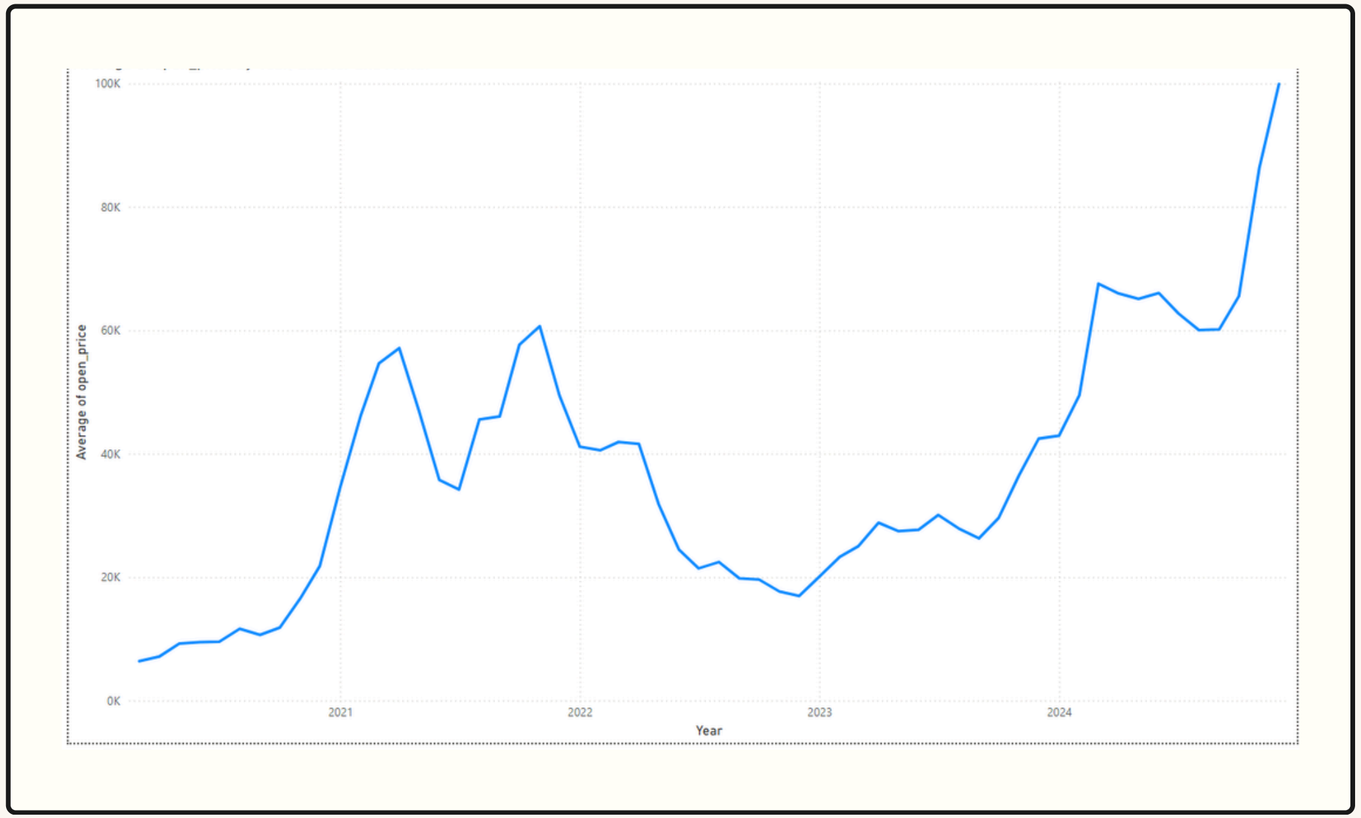
Difficulty against Time

As the number of crypto blocks are finite, the number of unmined bitcoins is reducing rapidly. Because of this, it is getting harder for crypto miners to mine for bitcoin as the years go by.



Open Price against Time

The graph of open price against time shows a similar pattern to the graph of close price, indicating a strong correlation and also another metric that investors can look at when trading.



Time-Series SUMMARY

Days

Tuesday has the highest close price on average, while Friday has the lowest close price on average

Open interest

The demand for cryptocurrency is increasing rapidly overtime, which implies that we should act fast in the market

Dates

The prices increase towards the middle of the month and constantly reduces towards the end of the month

Difficulty

Investors should buy crypto as fast as possible since it will be more difficult to purchase in the future

Months

March causes high close prices due to reveal of annual earnings report, while November also means higher prices due to busy holiday periods

Open Price

Monitoring the open price can help significantly as it is highly correlated with the close price

Are there any specific risk factors we are advised to look into before making an investment?

NEXT ➡

RESEARCH 3

ML MODEL:
RISK ANALYSIS

Import Libraries and Dataset

[1]:
```
# Import required libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
```

[60]:
```
%pip install xgboost
%pip install shap
```

```
Requirement already satisfied: xgboost in c:\users\user\anaconda3\lib\site-packages (2.1.3)
```

[22]:
```
df = pd.read_csv('C:/Users/User/OneDrive - Asia Pacific University/Desktop/UMDAC DATATHON/UM_datathon_2024.csv')
```

[24]:
```
df.head
```

[24]:
```
<bound method NDFrame.head of                     time    difficulty  estimated_leverage_ratio  \
0      2020-03-25 10:00:00  1.655292e+13                  0.070577
1      2020-03-25 11:00:00  1.655292e+13                  0.071966
2      2020-03-25 12:00:00  1.655292e+13                  0.072686
3      2020-03-25 13:00:00           NaN                  0.070549
4      2020-03-25 14:00:00  1.655292e+13                  0.071959
...                    ...           ...                       ...
41493  2024-12-18 07:00:00  1.085226e+14                  0.199751
41494  2024-12-18 08:00:00  1.085226e+14                  0.200089
41495  2024-12-18 09:00:00  1.085226e+14                  0.200362
41496  2024-12-18 10:00:00  1.085226e+14                  0.200380
41497  2024-12-18 11:00:00  1.085226e+14                  0.201131

       addresses_count_active  addresses_count_sender  \
0                       30656                   12888
1                       21379                    8467
```

Before Imputation

[125]:
```
count_long = (df['long_liquidations'] == 0).sum()
print("Number of zeros:", count_long)
count_short = (df['short_liquidations'] == 0).sum()
print("Number of zeros:", count_short)
```

```
Number of zeros: 6628
Number of zeros: 5381
```

[127]:
```
befmean1 = df['long_liquidations'].mean()
befmean2 = df['short_liquidations'].mean()
print(befmean1)
print(befmean2)
```

```
19.86419728661622
15.337658634151044
```

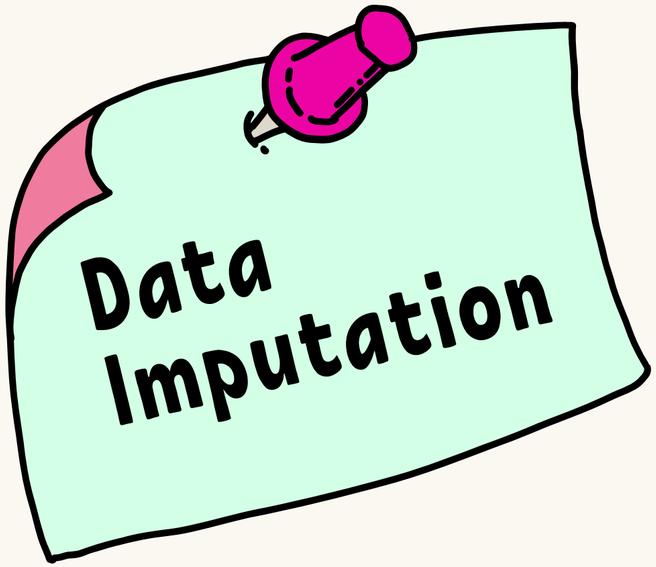
Imputation Process

[129]:
```
mean_value1 = df['long_liquidations'][df['long_liquidations'] != 0].mean()
mean_value2 = df['short_liquidations'][df['short_liquidations'] != 0].mean()

# Replace zeros with the calculated mean
df['long_liquidations'] = df['long_liquidations'].replace(0, mean_value1)
df['short_liquidations'] = df['short_liquidations'].replace(0, mean_value2)
```

After Imputation

[131]:
```
count_zeros1 = (df['long_liquidations'] == 0).sum()
count_zeros2 = (df['short_liquidations'] == 0).sum()
print("Number of zeros:", count_zeros1)
print("Number of zeros:", count_zeros2)
```

```
Number of zeros: 0
Number of zeros: 0
```



Create Feature

[42]:
```
# Interaction Features
df['whale_liquidation_interaction'] = df['exchange_whale_ratio'] * (df['long_liquidations'] + df['short_liquidations'])
```

[46]:
```
# Define high-risk periods
df['high_risk'] = (
    (df['exchange_whale_ratio'] > 0.9) |
    (df['long_liquidations'] > 100) |
    (df['short_liquidations'] > 100)
).astype(int)

# Check target variable distribution
print("High-risk periods:", df['high_risk'].sum())
print("Low-risk periods:", len(df) - df['high_risk'].sum())
```

```
High-risk periods: 19971
Low-risk periods: 21527
```

[50]:

```
# Select features
features = [
    'exchange_whale_ratio', 'close_price', 'volume', 'long_liquidations',
    'short_liquidations', 'whale_liquidation_interaction'
]
X = df[features]
y = df['high_risk']

# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3, random_state=42)

# Check data shapes
print(f"Training set size: {X_train.shape}")
print(f"Testing set size: {X_test.shape}")
```

```
Training set size: (29048, 6)
Testing set size: (12450, 6)
```

XGBoost Algorithm

[56]:
```python
import xgboost as xgb

# Initialize XGBoost model
xgb_model = xgb.XGBClassifier(
    n_estimators=100,
    learning_rate=0.1,
    max_depth=5,
    scale_pos_weight=5,  # Handle class imbalance
    random_state=42,
    use_label_encoder=False
)

# Train the model
xgb_model.fit(X_train, y_train)
```

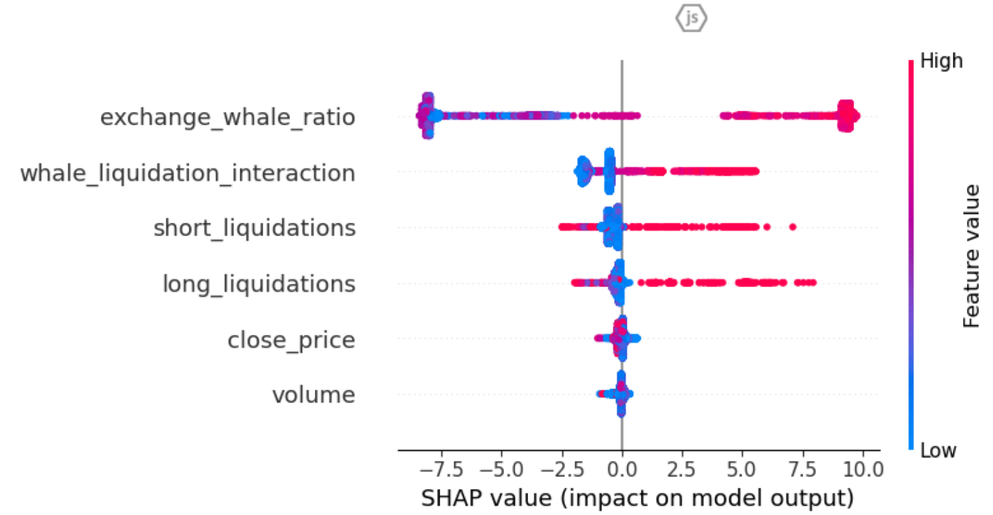
```
C:\Users\User\anaconda3\Lib\site-packages\xgboost\core.py:158: UserWarning: [14:37:40] WARNING: (
ng-group-i-0c55ff5f71b100e98-1\xgboost\xgboost-ci-windows\src\learner.cc:740:
Parameters: { "use_label_encoder" } are not used.

  warnings.warn(smsg, UserWarning)
```

[56]:

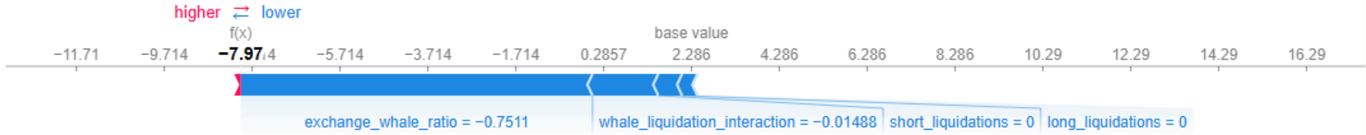
XGBClassifier

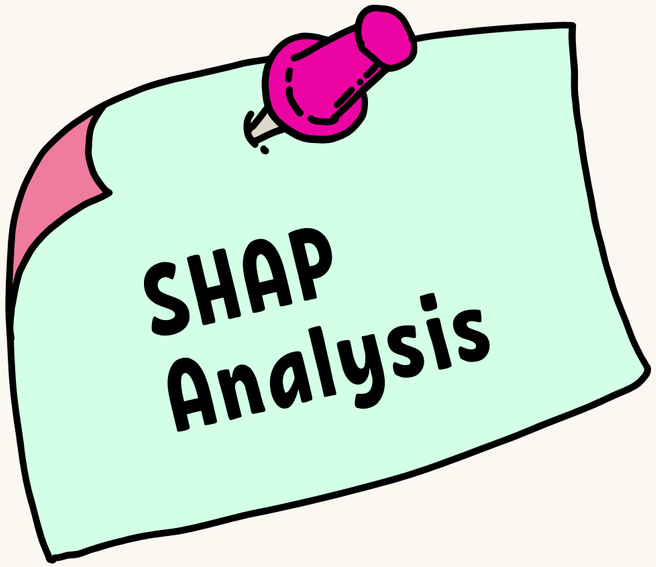
```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=0.1, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=5, max_leaves=None,
              min_child_weight=None, missing=nan, monotone_constraints=None,
              multi_strategy=None, n_estimators=100, n_jobs=None,
              num_parallel_tree=None, random_state=42, ...)
```

143]:
```python
import shap
shap.initjs()

# Initialize SHAP explainer
explainer = shap.TreeExplainer(xgb_model)
shap_values = explainer.shap_values(X_test)

# SHAP Summary Plot
shap.summary_plot(shap_values, X_test, feature_names=features)

# SHAP Force Plot for a single instance
shap.force_plot(explainer.expected_value, shap_values[0, :], X_test[0, :], feature_names=features)
```



143]:





Testing Model

In [166]:
```python
# Simulated data for hypothetical scenarios
simulated_data = pd.DataFrame({
    'exchange_whale_ratio': [0.95, 0.8, 0.5],
    'close_price': [6000, 6200, 6400],
    'volume': [5000, 4500, 4200],
    'long_liquidations': [150, 100, 50],
    'short_liquidations': [120, 90, 60],
    'whale_liquidation_interaction': [0.95 * 270, 0.8 * 190, 0.5 * 110]
})

# Standardize simulated data
simulated_data_scaled = scaler.transform(simulated_data)

# Predict risk probabilities
simulated_risk_proba = xgb_model.predict_proba(simulated_data_scaled)[:, 1]
print("Simulated Risk Probabilities:", simulated_risk_proba)
```

```
Simulated Risk Probabilities: [0.9999765  0.6582136  0.07000975]
```

Explanation

- Before carrying on with the machine learning algorithm, we need to make sure there are no missing values
- The long liquidities and short liquidities seem to have a few empty cells which may affect the result of our analysis
- Since more than 5% of the rows are containing null values, data imputation needs to be carried out
- We did this by replacing the null values in both columns with the mean of the non-null values
- By doing so, all missing values have been imputed and are now ready for analysis
- The XGBoost algorithm was used for this study because it is able to ordinally rank the data points so that we can identify if they are at high, medium or low risk
- After undergoing training, the values are passed on for SHAP analysis

Explanation

- SHAP values show how much each feature affects the final prediction, the significance of each feature compared to others, and the model's reliance on the interaction between features
- In this situation, we see that exchange whale ratio, whale liquidation interaction, long liquidations, and short liquidations play an important role in identifying how risky an investment is
- The red plots represent high risk, while the blue plots represent low risk
- After training the model, we test it new data values to see if the output matches the risk factor
- The expected output is achieved as the values 0.9999, 0.6582, 0.0700 represent high, medium, and low risk respectively

Now that we know the most significant factors, is there a way to predict the close price in cryptocurrency?

NEXT ➡

RESEARCH 4

ML MODEL:
PREDICTING CLOSE PRICE

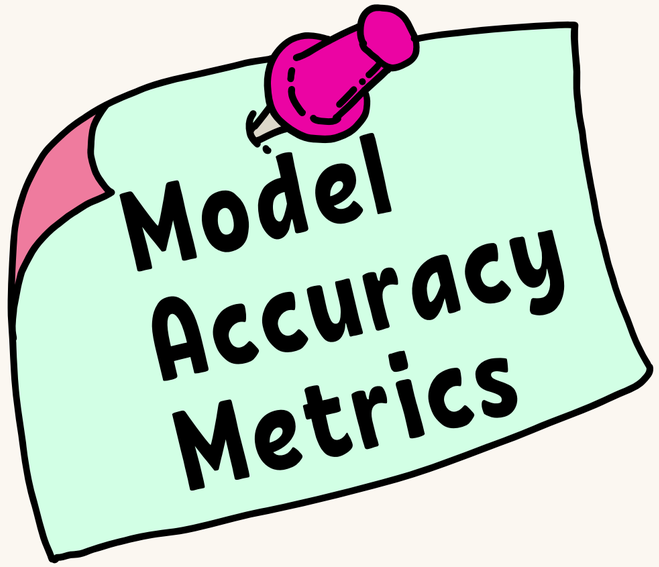
Modelling Procedure

```python
# Independent variables (features)
X = data[['open_interest','difficulty','miner_supply_ratio']].replace([np.inf, -np.inf], np.nan).dropna()
y = data['close_price'].loc[X.index]  # Dependent variable

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Dictionary of models
models = {
    'Linear Regression': LinearRegression(),
    'Random Forest': RandomForestRegressor(random_state=42),
    'Gradient Boosting': GradientBoostingRegressor(random_state=42),
    'Support Vector Regressor': SVR(),
    'Decision Tree': DecisionTreeRegressor(random_state=42)
}

# Training and evaluating models
results = {}
predictions = pd.DataFrame(index=X_test.index)  # Create empty DataFrame for predictions

for model_name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)
    mae = mean_absolute_error(y_test, y_pred)  # Calculate MAE
    r2 = r2_score(y_test, y_pred)
    results[model_name] = {'MSE': mse, 'MAE': mae, 'R2': r2}
    print(f'{model_name} - MSE: {mse}, MAE: {mae}, R2: {r2}')

    # Save predictions to DataFrame
    predictions[model_name] = y_pred

# Concatenate predictions DataFrame with original X_test
X_test_with_predictions = pd.concat([X_test, predictions], axis=1)

# Convert results to DataFrame
results_df = pd.DataFrame(results).T
results_df
```

```
Linear Regression - MSE: 111072941.65831535, MAE: 8069.512561997313, R2: 0.7239434817093391
Random Forest - MSE: 617826.8512784078, MAE: 417.1832925235638, R2: 0.998464476343887
Gradient Boosting - MSE: 4583649.173417388, MAE: 1513.534072807342, R2: 0.9886079704652826
Support Vector Regressor - MSE: 339655425.11331254, MAE: 15205.305751545136, R2: 0.15583316084529197
Decision Tree - MSE: 1017077.2902153573, MAE: 487.91308275528314, R2: 0.9974721942952312
```

[48]:

	MSE	MAE	R2
Linear Regression	1.110729e+08	8069.512562	0.723943
Random Forest	6.178269e+05	417.183293	0.998464
Gradient Boosting	4.583649e+06	1513.534073	0.988608
Support Vector Regressor	3.396554e+08	15205.305752	0.155833
Decision Tree	1.017077e+06	487.913083	0.997472

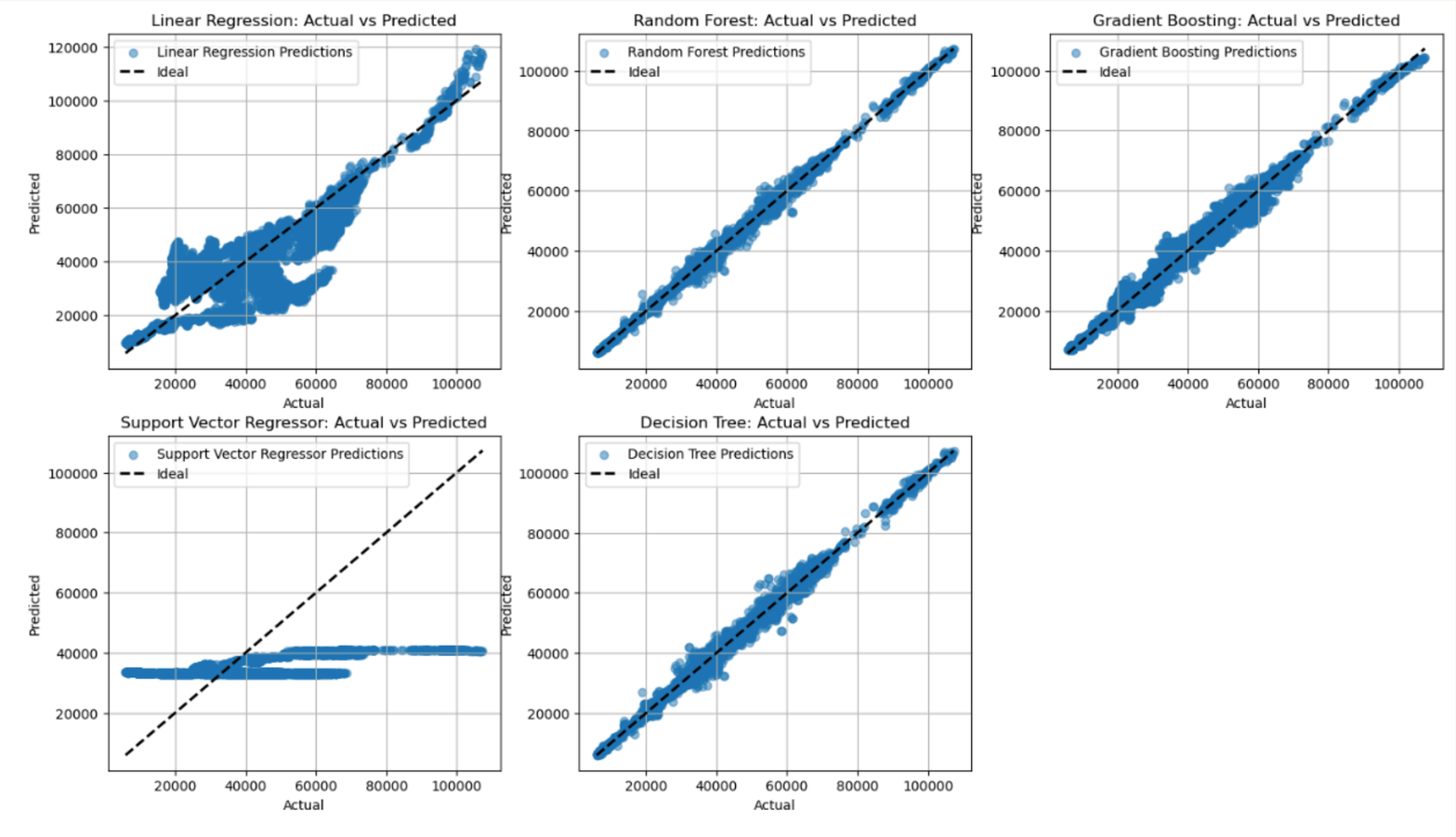


Explanation

- Dataset is split into training set (70%) and testing set (30%)
- The training set is fed to multiple regression models such as Linear Regression and Random Forest
- Models make predictions on the testing set to test their accuracy
- Accuracy metrics such as mean squared error (MSE) and r-squared value are used to determine which model is the best fit for the situation
- Even though there are models with almost perfect accuracy, it is not suitable to use them as it could be a result of overfitting
- Therefore, the best and most suitable model for this situation is the linear regression model

Code for Visualization

[50]:
```
plt.style.use('default')
fig, axes = plt.subplots(nrows=2, ncols=3, figsize=(18,10))
axes = axes.flatten()
for idx, model_name in enumerate(models.keys()):
    axes[idx].scatter(y_test, predictions[model_name], alpha=0.5, label=f'{model_name} Predictions')
    axes[idx].plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=2, label='Ideal')
    axes[idx].set_xlabel('Actual')
    axes[idx].set_ylabel('Predicted')
    axes[idx].set_title(f'{model_name}: Actual vs Predicted')
    axes[idx].legend()
    axes[idx].grid(True)

# Remove the last empty subplot if there's an extra one
if len(models) % 3 != 0:
    fig.delaxes(axes[-1])

plt.show()
```



Actual vs Predicted Values

Explanation

- In order to give a clear representation of each model, proper visualization techniques are performed to convey the results
- The actual values are plotted against the predicted values to see if it fits the ideal line (diagonal dashed line)
- As deduced earlier, the chosen linear regression model shows that most of the points are near the dashed line which further solidifies our statement
- Random forest, decision tree regressor, and gradient boosting regressor models show near perfect results which could lead to them performing poorly on new data
- Support Vector Regressor model is the least accurate model as most of the points are located away from the dashed line

CONCLUSIONS

EXAMINING SIGNIFICANT FACTORS

STRATEGIZING ACCORDING TO TIME

CONDUCTING PROPER RISK ANALYSIS

Open interest, mining difficulty, and miner supply ratio are all crucial indicators in determining the close price of crypto. Balaena Quant can use these findings to teach their new clients about what they should look out for when investing.

Multiple factors such as day, date, and month should be considered by investors when looking at the cryptocurrency market. Balaena Quant should help their clients purchase crypto according to specific times so that they receive maximum profit.

Cryptocurrency is a menacing industry which can either go really good or really bad if investments are made carelessly. Therefore, careful risk analysis should be carried out by Balaena Quant to notify their clients about the current situation of the stock market.

THANK YOU!

DataDreamin