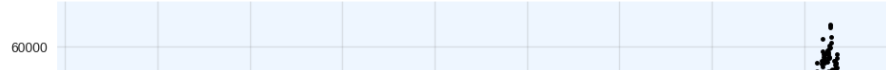


```
1 #Prophet Model

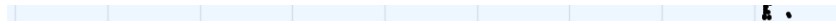
1 import pandas as pd
2 from prophet import Prophet
3
4 # Read the individual CSV files
5 btc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Bitcoin.csv")
6 eth_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Ethereum.csv")
7 xrp_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_XRP.csv")
8 ltc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Litecoin.csv")
9 usdc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_USDCoin.csv")
10
11 # Concatenate the datasets vertically
12 df = pd.concat([btc_df, eth_df, xrp_df, ltc_df, usdc_df])
13
14 # Convert the 'Date' column to a datetime format
15 df['Date'] = pd.to_datetime(df['Date'])
16
17 # Rename the 'Date' and 'Close' columns to 'ds' and 'y', respectively
18 df = df.rename(columns={'Date': 'ds', 'Close': 'y'})
19
20 # Create and fit the model
21 model = Prophet()
22 model.fit(df)
23
24 # Generate future dates for prediction
25 future_dates = model.make_future_dataframe(periods=30) # Adjust the number of future data points as needed
26
27 # Make predictions
28 forecast = model.predict(future_dates)
29
30 # Plot the forecasted values
31 model.plot(forecast)
32
```



```
INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/ubj8kjpg.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/im0sk2uv.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_model/p
17:19:10 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
17:19:12 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
```

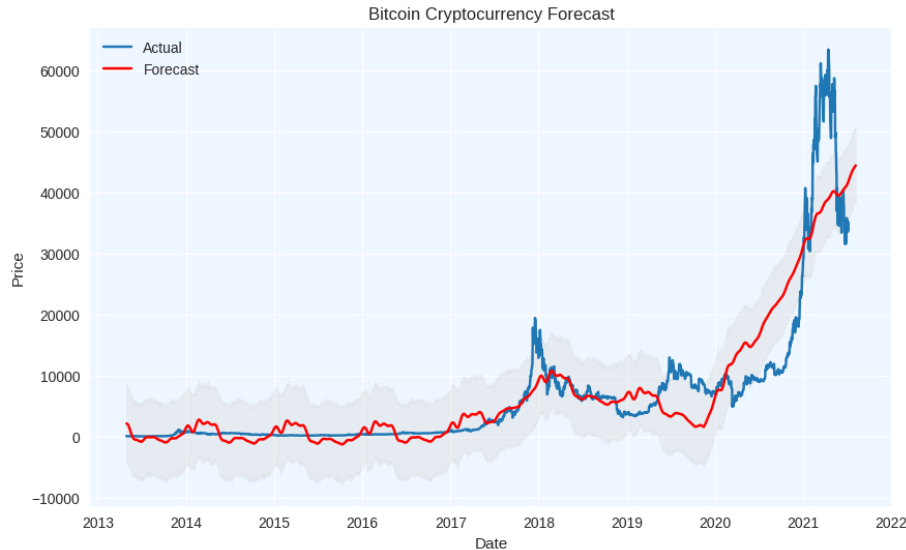


1 #Better representation of Prophet Model



```
1 import pandas as pd
2 from prophet import Prophet
3 import matplotlib.pyplot as plt
4
5 # Read the individual CSV files
6 btc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Bitcoin.csv")
7 eth_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Ethereum.csv")
8 xrp_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_XRP.csv")
9 ltc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Litecoin.csv")
10 usdc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_USDCoin.csv")
11
12 # Create a dictionary of coins and their respective dataframes
13 coins = {
14     'Bitcoin': btc_df,
15     'Ethereum': eth_df,
16     'XRP': xrp_df,
17     'Litecoin': ltc_df,
18     'USDCoin': usdc_df
19 }
20
21 # Set the color palette and background color
22 #colors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd']
23 plt.style.use('seaborn')
24 plt.rcParams['axes.facecolor'] = '#eef6ff'
25
26 # Iterate over the coins dictionary
27 for coin, df in coins.items():
28     # Convert the 'Date' column to a datetime format
29     df['Date'] = pd.to_datetime(df['Date'])
30
31     # Rename the 'Date' and 'Close' columns to 'ds' and 'y', respectively
32     df = df.rename(columns={'Date': 'ds', 'Close': 'y'})
33
34     # Create and fit the model
35     model = Prophet()
36     model.fit(df)
37
38     # Generate future dates for prediction
39     future_dates = model.make_future_dataframe(periods=30) # Adjust the number of future data points as needed
40
41     # Make predictions
42     forecast = model.predict(future_dates)
43
44     # Plot the forecasted values
45     fig = plt.figure(figsize=(10, 6))
46     ax = fig.add_subplot(111)
47     ax.plot(df['ds'], df['y'], color=colors[0], label='Actual')
48     ax.plot(forecast['ds'], forecast['yhat'], color='red', label='Forecast')
49     ax.fill_between(forecast['ds'], forecast['yhat_lower'], forecast['yhat_upper'], color='lightgray', alpha=0.3)
50
51     # Customize the plot
52     plt.title(f'{coin} Cryptocurrency Forecast')
53     plt.xlabel('Date')
54     plt.ylabel('Price')
55     plt.legend()
56     plt.show()
57
```

```
<ipython-input-34-5725250f138f>:23: MatplotlibDeprecationWarning: The seaborn styles ship
plt.style.use('seaborn')
INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to over
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/wrztlda4.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/y7jtkvuv.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_mode
17:19:15 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
17:19:15 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
```



```
INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to over
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/bkv6jfbv.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/xpyxaoqb.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_mode
17:19:17 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
17:19:18 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
```



```
1 import pandas as pd
2 from prophet import Prophet
3 import matplotlib.pyplot as plt
4
5 # Read the individual CSV files
6 btc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Bitcoin.csv")
7 eth_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Ethereum.csv")
8 xrp_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_XRP.csv")
9 ltc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Litecoin.csv")
10 usdc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_USDCoin.csv")
11
12 # Create a dictionary of coins and their respective dataframes
13 coins = {
14     'Bitcoin': btc_df,
15     'Ethereum': eth_df,
16     'XRP': xrp_df,
17     'Litecoin': ltc_df,
18     'USDCoin': usdc_df
19 }
20
21 # Set the color palette and background color
22 colors = ['#1f77b4']
23 plt.rcParams['axes.facecolor'] = '#eef6ff'
24
25 # Create a 3x2 grid of subplots
26 fig, axes = plt.subplots(3, 2, figsize=(12, 12))
27
```

```
28 # Iterate over the coins dictionary and plot the forecast in each subplot
29 for i, (coin, df) in enumerate(coins.items()):
30     row = i // 2
31     col = i % 2
32     ax = axes[row, col]
33
34     # Convert the 'Date' column to a datetime format
35     df['Date'] = pd.to_datetime(df['Date'])
36
37     # Rename the 'Date' and 'Close' columns to 'ds' and 'y', respectively
38     df = df.rename(columns={'Date': 'ds', 'Close': 'y'})
39
40     # Create and fit the model
41     model = Prophet()
42     model.fit(df)
43
44     # Generate future dates for prediction
45     future_dates = model.make_future_dataframe(periods=30) # Adjust the number of future data points as needed
46
47     # Make predictions
48     forecast = model.predict(future_dates)
49
50     # Plot the forecasted values
51     ax.plot(df['ds'], df['y'], color=colors[0], label='Actual')
52     ax.plot(forecast['ds'], forecast['yhat'], color='red', linestyle='dashed', label='Forecast')
53     ax.fill_between(forecast['ds'], forecast['yhat_lower'], forecast['yhat_upper'], color=colors[0], alpha=0.3)
54
55     # Customize the subplot
56     ax.set_title(coin)
57     ax.set_xlabel('Date')
58     ax.set_ylabel('Price')
59     ax.legend()
60
61 # Adjust the spacing between subplots
62 fig.tight_layout()
63
64 # Show the plot
65 plt.show()
66
```

```

INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to overr
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/cvrjm5pf.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/2wu64z51.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_mode
17:19:27 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
17:19:28 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to overr
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/ps6iludh.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/ilp2gvg2.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_mode
17:19:28 - cmdstanpy - INFO - Chain [1] start processing
1 #to predict future values- 365 days
INFO:cmdstanpy:Chain [1] done processing
1 import pandas as pd
2 from prophet import Prophet
3 import matplotlib.pyplot as plt
4
5 # Read the individual CSV files
6 btc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Bitcoin.csv")
7 eth_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Ethereum.csv")
8 xrp_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_XRP.csv")
9 ltc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Litecoin.csv")
10 usdc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_USDCoin.csv")
11
12 # Create a dictionary of coins and their respective dataframes
13 coins = {
14     'Bitcoin': btc_df,
15     'Ethereum': eth_df,
16     'XRP': xrp_df,
17     'Litecoin': ltc_df,
18     'USDCoin': usdc_df
19 }
20
21 # Set the color palette and background color
22 colors = ['#1f77b4']
23 plt.rcParams['axes.facecolor'] = '#eef6ff'
24
25 # Create a 3x2 grid of subplots
26 fig, axes = plt.subplots(3, 2, figsize=(12, 12))
27
28 # Iterate over the coins dictionary and plot the forecast in each subplot
29 for i, (coin, df) in enumerate(coins.items()):
30     row = i // 2
31     col = i % 2
32     ax = axes[row, col]
33
34     # Convert the 'Date' column to a datetime format
35     df['Date'] = pd.to_datetime(df['Date'])
36
37     # Rename the 'Date' and 'Close' columns to 'ds' and 'y', respectively
38     df = df.rename(columns={'Date': 'ds', 'Close': 'y'})
39
40     # Create and fit the model
41     model = Prophet()
42     model.fit(df)
43
44     # Generate future dates for prediction
45     future_dates = model.make_future_dataframe(periods=365) # Adjust the number of future data points as needed
46
47     # Make predictions
48     forecast = model.predict(future_dates)
49
50     # Plot the forecasted values
51     ax.plot(df['ds'], df['y'], color=colors[0], label='Actual')
52     ax.plot(forecast['ds'], forecast['yhat'], color='red', linestyle='dashed', label='Forecast')
53     ax.fill_between(forecast['ds'], forecast['yhat_lower'], forecast['yhat_upper'], color=colors[0], alpha=0.3)
54
55     # Customize the subplot
56     ax.set_title(coin)
57     ax.set_xlabel('Date')
58     ax.set_ylabel('Price')

```

```
59     ax.legend()
60
61 # Adjust the spacing between subplots
62 fig.tight_layout()
63
64 # Show the plot
65 plt.show()
66
```

```
1 #Evaluation of Model

1 import pandas as pd
2 from prophet import Prophet
3 from sklearn.metrics import r2_score
4 import matplotlib.pyplot as plt
5
6 # Read the individual CSV files
7 btc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Bitcoin.csv")
8 eth_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Ethereum.csv")
9 xrp_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_XRP.csv")
10 ltc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Litecoin.csv")
11 usdc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_USDCoin.csv")
12
```

```

13 # Create a dictionary of coins and their respective dataframes
14 coins = {
15     'Bitcoin': btc_df,
16     'Ethereum': eth_df,
17     'XRP': xrp_df,
18     'Litecoin': ltc_df,
19     'USDCoin': usdc_df
20 }
21
22 # Set the color palette and background color
23 colors = ['#1f77b4']
24 plt.rcParams['axes.facecolor'] = '#eef6ff'
25
26 # Create a 3x2 grid of subplots
27 fig, axes = plt.subplots(3, 2, figsize=(12, 12))
28
29 # Iterate over the coins dictionary and plot the forecast in each subplot
30 for i, (coin, df) in enumerate(coins.items()):
31     row = i // 2
32     col = i % 2
33     ax = axes[row, col]
34
35     # Convert the 'Date' column to a datetime format
36     df['Date'] = pd.to_datetime(df['Date'])
37
38     # Rename the 'Date' and 'Close' columns to 'ds' and 'y', respectively
39     df = df.rename(columns={'Date': 'ds', 'Close': 'y'})
40
41     # Create and fit the model
42     model = Prophet(daily_seasonality=False) # Disable daily seasonality
43     model.fit(df)
44
45     # Generate future dates for prediction
46     future_dates = model.make_future_dataframe(periods=30) # Adjust the number of future data points as needed
47
48     # Make predictions
49     forecast = model.predict(future_dates)
50
51     # Evaluate the model
52     df_eval = forecast.set_index('ds')[['yhat']].join(df.set_index('ds')['y']).reset_index()
53     df_eval.fillna(df_eval.mean(), inplace=True) # Fill missing values with mean
54     mse = ((df_eval['yhat'] - df_eval['y']) ** 2).mean()
55     rmse = mse ** 0.5
56     mae = (df_eval['yhat'] - df_eval['y']).abs().mean()
57     r2 = r2_score(df_eval['y'], df_eval['yhat'])
58
59     # Plot the forecasted values
60     ax.plot(df['ds'], df['y'], color=colors[0], label='Actual')
61     ax.plot(forecast['ds'], forecast['yhat'], color='red', linestyle='dashed', label='Forecast')
62     ax.fill_between(forecast['ds'], forecast['yhat_lower'], forecast['yhat_upper'], color=colors[0], alpha=0.3)
63
64     # Customize the subplot
65     ax.set_title(coin)
66     ax.set_xlabel('Date')
67     ax.set_ylabel('Price')
68     ax.legend()
69
70     # Display evaluation metrics
71     textstr = f"MSE: {mse:.2f}\nRMSE: {rmse:.2f}\nMAE: {mae:.2f}\nR2 Score: {r2:.2f}"
72     ax.text(0.05, 0.95, textstr, transform=ax.transAxes, fontsize=10, verticalalignment='top')
73
74 # Adjust the spacing between subplots
75 fig.tight_layout()
76
77 # Show the plot
78 plt.show()
79

```

```

DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/ltccs77g.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/_2n5i67r.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_model/
17:31:57 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
17:31:58 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
<ipython-input-40-5b2b048c9bd2>:53: FutureWarning: DataFrame.mean and DataFrame.median with
df_eval.fillna(df_eval.mean(), inplace=True) # Fill missing values with mean
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/pauquku5i.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/wppbkog3.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_model/
17:31:59 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
17:32:00 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
<ipython-input-40-5b2b048c9bd2>:53: FutureWarning: DataFrame.mean and DataFrame.median with
df_eval.fillna(df_eval.mean(), inplace=True) # Fill missing values with mean
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/8mz72u5s.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/pr9wtrlx.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_model/
17:32:01 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
17:32:02 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
<ipython-input-40-5b2b048c9bd2>:53: FutureWarning: DataFrame.mean and DataFrame.median with
df_eval.fillna(df_eval.mean(), inplace=True) # Fill missing values with mean
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/hg5i3j8r.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/w086wz7z.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_model/
17:32:03 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
17:32:04 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
<ipython-input-40-5b2b048c9bd2>:53: FutureWarning: DataFrame.mean and DataFrame.median with
df_eval.fillna(df_eval.mean(), inplace=True) # Fill missing values with mean
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/xsl1hdvm.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/tkeqyvou.json
DEBUG:cmdstanpy:idx 0

1 import pandas as pd
2 from prophet import Prophet
3 from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
4
5 # Read the individual CSV files
6 btc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Bitcoin.csv")
7 eth_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Ethereum.csv")
8 xrp_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_XRP.csv")
9 ltc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Litecoin.csv")
10 usdc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_USDCoin.csv")
11
12 # Create a dictionary of coins and their respective dataframes
13 coins = {
14     'Bitcoin': btc_df,
15     'Ethereum': eth_df,
16     'XRP': xrp_df,
17     'Litecoin': ltc_df,
18     'USDCoin': usdc_df
19 }
20
21 # Initialize an empty DataFrame to store the evaluation metrics
22 metrics_df = pd.DataFrame(columns=['MSE', 'RMSE', 'MAE', 'R2 Score'])
23
24 # Iterate over the coins dictionary and evaluate the model for each coin
25 for coin, df in coins.items():
26     # Convert the 'Date' column to a datetime format
27     df['Date'] = pd.to_datetime(df['Date'])
28
29     # Rename the 'Date' and 'Close' columns to 'ds' and 'y', respectively
30     df = df.rename(columns={'Date': 'ds', 'Close': 'y'})
31
32     # Create and fit the model
33     model = Prophet()

```



```

34     model.fit(df)
35
36     # Generate future dates for prediction
37     future_dates = model.make_future_dataframe(periods=30) # Adjust the number of future data points as needed
38
39     # Make predictions
40     forecast = model.predict(future_dates)
41
42     # Evaluate the model
43     df_eval = forecast[['ds', 'yhat']].merge(df[['ds', 'y']], on='ds', how='inner').dropna()
44     mse = mean_squared_error(df_eval['y'], df_eval['yhat'])
45     rmse = mse ** 0.5
46     mae = mean_absolute_error(df_eval['y'], df_eval['yhat'])
47     r2 = r2_score(df_eval['y'], df_eval['yhat'])
48
49     # Add the evaluation metrics to the DataFrame
50     metrics_df.loc[coin] = [mse, rmse, mae, r2]
51
52 # Display the table of evaluation metrics
53 print(metrics_df)
54

```

INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.

DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/8r76dxb2.json

DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/by9i89ri.json

DEBUG:cmdstanpy:idx 0

DEBUG:cmdstanpy:running CmdStan, num_threads: None

DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_model/prophet_model.bin', 'random', 'seed=79782',

17:44:51 - cmdstanpy - INFO - Chain [1] start processing

INFO:cmdstanpy:Chain [1] start processing

17:44:51 - cmdstanpy - INFO - Chain [1] done processing

INFO:cmdstanpy:Chain [1] done processing

INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.

DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/m8haq3z1.json

DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/bph5miy0.json

DEBUG:cmdstanpy:idx 0

DEBUG:cmdstanpy:running CmdStan, num_threads: None

DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_model/prophet_model.bin', 'random', 'seed=49837',

17:44:52 - cmdstanpy - INFO - Chain [1] start processing

INFO:cmdstanpy:Chain [1] start processing

17:44:53 - cmdstanpy - INFO - Chain [1] done processing

INFO:cmdstanpy:Chain [1] done processing

INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.

DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/dtnp1tzq.json

DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/9di9l0u2.json

DEBUG:cmdstanpy:idx 0

DEBUG:cmdstanpy:running CmdStan, num_threads: None

DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_model/prophet_model.bin', 'random', 'seed=32572',

17:44:54 - cmdstanpy - INFO - Chain [1] start processing

INFO:cmdstanpy:Chain [1] start processing

17:44:56 - cmdstanpy - INFO - Chain [1] done processing

INFO:cmdstanpy:Chain [1] done processing

INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.

DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/b5ab6at8.json

DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/rosccsy03.json

DEBUG:cmdstanpy:idx 0

DEBUG:cmdstanpy:running CmdStan, num_threads: None

DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_model/prophet_model.bin', 'random', 'seed=71320',

17:44:58 - cmdstanpy - INFO - Chain [1] start processing

INFO:cmdstanpy:Chain [1] start processing

17:44:59 - cmdstanpy - INFO - Chain [1] done processing

INFO:cmdstanpy:Chain [1] done processing

INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.

DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/yeaeu9pc.json

DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/vz72c5og.json

DEBUG:cmdstanpy:idx 0

DEBUG:cmdstanpy:running CmdStan, num_threads: None

DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_model/prophet_model.bin', 'random', 'seed=59313',

17:45:00 - cmdstanpy - INFO - Chain [1] start processing

INFO:cmdstanpy:Chain [1] start processing

17:45:01 - cmdstanpy - INFO - Chain [1] done processing

INFO:cmdstanpy:Chain [1] done processing

	MSE	RMSE	MAE	R2 Score
Bitcoin	2.275776e+07	4770.509099	2895.874572	0.821655
Ethereum	3.165409e+04	177.915951	108.106014	0.912347
XRP	3.865966e-02	0.196621	0.110671	0.662744
Litecoin	8.686634e+02	29.473096	19.794538	0.782727
USDCoin	1.746138e-05	0.004179	0.002638	0.624399

▼ Here is the analysis for each coin based on the evaluation metrics:

1. **Bitcoin:**

- MSE: 2.275776e+07
- RMSE: 4770.509099
- MAE: 2895.874572
- R2 Score: 0.821655

Analysis: The model achieves a relatively high R2 score of 0.821655, indicating that it captures a significant portion of the variance in Bitcoin's price. However, the MSE and RMSE values are quite large, suggesting that there are still considerable prediction errors. The MAE is also relatively high, indicating that the average difference between the predicted and actual values is significant.

2. **Ethereum:**

- MSE: 3.165409e+04
- RMSE: 177.915951
- MAE: 108.106014
- R2 Score: 0.912347

Analysis: The model performs well for Ethereum, as indicated by the high R2 score of 0.912347. The MSE, RMSE, and MAE values are relatively small, suggesting that the model's predictions are generally accurate and close to the actual values.

3. **XRP:**

- MSE: 3.865966e-02
- RMSE: 0.196621
- MAE: 0.110671
- R2 Score: 0.662744

Analysis: The model achieves a moderate R2 score of 0.662744 for XRP, indicating that it captures a reasonable portion of the variance in XRP's price. The MSE, RMSE, and MAE values are quite small, suggesting that the model's predictions are relatively accurate for XRP.

4. **Litecoin:**

- MSE: 8.686634e+02
- RMSE: 29.473096
- MAE: 19.794538
- R2 Score: 0.782727

Analysis: The model performs reasonably well for Litecoin, as indicated by the R2 score of 0.782727. The MSE, RMSE, and MAE values are relatively small, suggesting that the model's predictions are generally accurate and close to the actual values.

5. **USDCoin:**

- MSE: 1.746138e-05
- RMSE: 0.004179
- MAE: 0.002638
- R2 Score: 0.624399

Analysis: The model achieves a moderate R2 score of 0.624399 for USDCoin, indicating that it captures a reasonable portion of the variance in USDCoin's price. The MSE, RMSE, and MAE values are very small, suggesting that the model's predictions are quite accurate for USDCoin.

Overall, the model performs well for most coins, with high R2 scores and relatively small error metrics (MSE, RMSE, MAE). However, there is room for improvement, especially for Bitcoin, where the errors are relatively larger. Further analysis and refinement of the model may be required to improve its performance for Bitcoin.

1 #For each refinement step applied to each coin separately:

```
1 import pandas as pd
2 from prophet import Prophet
3
4 # Read the individual CSV files
5 btc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Bitcoin.csv")
6 eth_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Ethereum.csv")
7 xrp_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_XRP.csv")
8 ltc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Litecoin.csv")
9 usdc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_USDCoin.csv")
```

```

10
11 # Create a dictionary of coins and their respective dataframes
12 coins = {
13     'Bitcoin': btc_df,
14     'Ethereum': eth_df,
15     'XRP': xrp_df,
16     'Litecoin': ltc_df,
17     'USDCoin': usdc_df
18 }
19
20 # Iterate over the coins dictionary and refine the model for each coin
21 for coin, df in coins.items():
22     # Feature Engineering
23     # Example: Include additional features like Volume and Market Cap
24     df['Volume'] = df['Volume'].fillna(0)
25     df['MarketCap'] = df['Close'] * df['Volume']
26
27     # Model Hyperparameter Tuning
28     model = Prophet(changepoint_prior_scale=0.05, seasonality_prior_scale=10)
29     # Update the hyperparameters according to the desired configuration
30
31     # Data Quality and Preprocessing
32     # Example: Handle missing values and outliers
33     df = df.fillna(method='ffill')
34     df = df.dropna()
35     # You can add additional preprocessing steps here if required
36
37     # Print the refined model for each coin
38     print(f"Refined Model for {coin}: {model}")
39
40
41 Refined Model for Bitcoin: <prophet.forecaster.Prophet object at 0x7fdcab8bb50>
42 Refined Model for Ethereum: <prophet.forecaster.Prophet object at 0x7fdcaa5fc220>
43 Refined Model for XRP: <prophet.forecaster.Prophet object at 0x7fdcaa5fecb0>
44 Refined Model for Litecoin: <prophet.forecaster.Prophet object at 0x7fdcaa5fc220>
45 Refined Model for USDCoin: <prophet.forecaster.Prophet object at 0x7fdcaa5fead0>

```

```

1 import pandas as pd
2 from prophet import Prophet
3 import matplotlib.pyplot as plt
4
5 # Read the individual CSV files
6 btc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Bitcoin.csv")
7 eth_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Ethereum.csv")
8 xrp_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_XRP.csv")
9 ltc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_Litecoin.csv")
10 usdc_df = pd.read_csv("https://raw.githubusercontent.com/Amarpreet3/CIND-820-CAPSTONE/main/Dataset/coin_USDCoin.csv")
11
12 # Create a dictionary of coins and their respective dataframes
13 coins = {
14     'Bitcoin': btc_df,
15     'Ethereum': eth_df,
16     'XRP': xrp_df,
17     'Litecoin': ltc_df,
18     'USDCoin': usdc_df
19 }
20
21 # Iterate over the coins dictionary and refine the model for each coin
22 for coin, df in coins.items():
23     # Rename the columns to match the expected format
24     df = df.rename(columns={'Date': 'ds', 'Close': 'y'})
25
26     # Feature Engineering
27     # Example: Include additional features like Volume and Market Cap
28     df['Volume'] = df['Volume'].fillna(0)
29     df['MarketCap'] = df['y'] * df['Volume']
30
31     # Model Hyperparameter Tuning
32     model = Prophet(changepoint_prior_scale=0.05, seasonality_prior_scale=10)
33     # Update the hyperparameters according to the desired configuration
34
35     # Data Quality and Preprocessing
36     # Example: Handle missing values and outliers
37     df = df.fillna(method='ffill')
38     df = df.dropna()
39     # You can add additional preprocessing steps here if required
40
41

```

```
41 # Fit the model
42 model.fit(df)
43
44 # Generate future dates for prediction
45 future_dates = model.make_future_dataframe(periods=30) # Adjust the number of future data points as needed
46
47 # Make predictions
48 forecast = model.predict(future_dates)
49
50 # Plot the forecasted values
51 model.plot(forecast)
52 plt.title(f"Forecasted Prices for {coin}")
53 plt.xlabel("Date")
54 plt.ylabel("Price")
55 plt.show()
56
```

```
INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to over
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/wggjgbsj.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpzeuacr7o/ybcjztqj.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_mode
18:20:59 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
18:20:59 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
```

Forecasted Prices for Bitcoin

--	--	--	--	--	--	--	--	--	--