In [1]:
```
# import libraries
import boto3, re, sys, math, json, os, sagemaker, urllib.request
from sagemaker import get_execution_role
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from IPython.display import Image
from IPython.display import display
from time import gmtime, strftime
from sagemaker.predictor import csv_serializer

# Define IAM role
role = get_execution_role()
prefix = 'sagemaker/DEMO-xgboost-dm'
my_region = boto3.session.Session().region_name # set the region of the instan

# this line automatically looks for the XGBoost image URI and builds an XGBoos
xgboost_container = sagemaker.image_uris.retrieve("xgboost", my_region, "lates

print("Success - the MySageMakerInstance is in the " + my_region + " region. Y
```

```
Success - the MySageMakerInstance is in the us-west-2 region. You will use th
e 433757028032.dkr.ecr.us-west-2.amazonaws.com/xgboost:latest container for y
our SageMaker endpoint.
```

In [2]:
```
bucket_name = 'raw-sample-file'
s3 = boto3.resource('s3')
try:
    if  my_region == 'us-east-1':
      s3.create_bucket(Bucket=bucket_name)
    else:
      s3.create_bucket(Bucket=bucket_name, CreateBucketConfiguration={ 'Locati
    print('S3 bucket created successfully')
except Exception as e:
```

```
S3 bucket created successfully
```

In [3]:
```
try:
  urllib.request.urlretrieve ("https://raw-sample-file.s3.us-west-2.amazonaws.
  print('Success: downloaded ethereum_price.csv.')
except Exception as e:
  print('Data load error: ',e)

try:
  model_data = pd.read_csv('./ethereum_price.csv',index_col=0)
  print('Success: Data loaded into dataframe.')
except Exception as e:
```

```
  Cell In[3], line 2
    urllib.request.urlretrieve ("https://raw-sample-file.s3.us-west-2.amazona
ws.com/ethereum_price.csv", "ethereum_price.csv)

^
SyntaxError: unterminated string literal (detected at line 2)
```

In [4]:
```
try:
  urllib.request.urlretrieve ("https://raw-sample-file.s3.us-west-2.amazonaws.
  print('Success: downloaded ethereum_price.csv.')
except Exception as e:
  print('Data load error: ',e)

try:
  model_data = pd.read_csv('./ethereum_price.csv',index_col=0)
  print('Success: Data loaded into dataframe.')
except Exception as e:
    print('Data load error: ',e)
```

```
Success: downloaded ethereum_price.csv.
Success: Data loaded into dataframe.
```

In [5]:

```
import pandas as pd
import datetime
import regex as re
import math
import matplotlib.pyplot as plt
import plotly.graph_objects as go
import plotly.express as px
from itertools import cycle
import numpy as np
from sklearn.metrics import mean_squared_error, mean_absolute_error, explained
from sklearn.metrics import mean_poisson_deviance, mean_gamma_deviance, accura
from sklearn.preprocessing import MinMaxScaler

import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.layers import LSTM, GRU

import warnings
warnings.filterwarnings('ignore')
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
Cell In[5], line 14
     11 from sklearn.metrics import mean_poisson_deviance, mean_gamma_devianc
e, accuracy_score
     12 from sklearn.preprocessing import MinMaxScaler
---> 14 import tensorflow as tf
     15 from tensorflow.keras.models import Sequential
     16 from tensorflow.keras.layers import Dense, Dropout

ModuleNotFoundError: No module named 'tensorflow'
```

In [6]:

In [7]:

Out[7]:

	Date	Price	Open	High	Low	Vol.	Change %
0	8-Mar-23	1,553.49	1,561.79	1,569.70	1,548.98	498.57K	-0.53%
1	7-Mar-23	1,561.78	1,565.84	1,580.95	1,536.31	460.10K	-0.26%
2	6-Mar-23	1,565.84	1,564.36	1,581.13	1,555.43	322.16K	0.09%
3	5-Mar-23	1,564.37	1,566.73	1,587.95	1,556.84	313.01K	-0.15%
4	4-Mar-23	1,566.73	1,569.45	1,577.02	1,550.10	247.02K	-0.14%

In [8]:
```
eth['Date'] = pd.to_datetime(eth.Date)

for i in range(len(eth)):
    eth['Price'][i] = float(re.sub(',', '', eth['Price'][i]))
    eth['Open'][i] = float(re.sub(',', '', eth['Open'][i]))
    eth['High'][i] = float(re.sub(',', '', eth['High'][i]))
    eth['Low'][i] = float(re.sub(',', '', eth['Low'][i]))
    eth['Change %'][i] = float(re.sub('%', '', eth['Change %'][i]))
    if eth['Vol.'][i][-1] == 'K':
        eth['Vol.'][i] = int(float(re.sub('K', '', eth['Vol.'][i])) * 1000)
    elif eth['Vol.'][i][-1] == 'M':
        eth['Vol.'][i] = int(float(re.sub('M', '', eth['Vol.'][i])) * 1000000)
```

```
/tmp/ipykernel_7421/2147789252.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  eth['Price'][i] = float(re.sub(',', '', eth['Price'][i]))
/tmp/ipykernel_7421/2147789252.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  eth['Open'][i] = float(re.sub(',', '', eth['Open'][i]))
/tmp/ipykernel_7421/2147789252.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  eth['High'][i] = float(re.sub(',', '', eth['High'][i]))
/tmp/ipykernel_7421/2147789252.py:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  eth['Low'][i] = float(re.sub(',', '', eth['Low'][i]))
/tmp/ipykernel_7421/2147789252.py:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  eth['Change %'][i] = float(re.sub('%', '', eth['Change %'][i]))
/tmp/ipykernel_7421/2147789252.py:10: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  eth['Vol.'][i] = int(float(re.sub('K', '', eth['Vol.'][i])) * 1000)
/tmp/ipykernel_7421/2147789252.py:12: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  eth['Vol.'][i] = int(float(re.sub('M', '', eth['Vol.'][i])) * 1000000)
```

Out[8]:

	Date	Price	Open	High	Low	Vol.	Change %
0	2023-03-08	1553.49	1561.79	1569.7	1548.98	498570	-0.53
1	2023-03-07	1561.78	1565.84	1580.95	1536.31	460100	-0.26
2	2023-03-06	1565.84	1564.36	1581.13	1555.43	322160	0.09
3	2023-03-05	1564.37	1566.73	1587.95	1556.84	313010	-0.15
4	2023-03-04	1566.73	1569.45	1577.02	1550.1	247020	-0.14

In [9]:

```
```

Out[9]:
```
(2555, 7)
```

In [10]:
```
print('Total number of days :', eth.Date.nunique())
```

```
Total number of days : 2555
Total number of fields : 7
```

In [11]:
```
print("Null values :", eth.isnull().values.sum())
```

```
Null values : 0
NA values : False
```

In [12]:
```
print("Starting date :", eth.iloc[-1][0])
print("Ending date :", eth.iloc[0][0])
```

```
Starting date : 2016-03-10 00:00:00
Ending date : 2023-03-08 00:00:00
Duration : 2554 days 00:00:00
```

In [13]:
```
monthwise = eth.groupby(pd.DatetimeIndex(eth.Date).month)[['Open']].mean()
new_order = ['January', 'February', 'March', 'April', 'May', 'June', 'July', '
             'September', 'October', 'November', 'December']
monthwise = monthwise.reset_index()
monthwise['Date'] = new_order
```

Out[13]:

	Date	Open
0	January	1012.926636
1	February	1057.254670
2	March	856.974306
3	April	897.661762
4	May	940.999447
5	June	729.158619
6	July	666.152673
7	August	857.359770
8	September	848.079286
9	October	888.357926
10	November	989.121476
11	December	971.279631

In [14]:
```
fig = go.Figure()

fig.add_trace(go.Bar(
    x = monthwise.Date,
    y = monthwise['Open'],
    name = 'Stock Open Price',
    marker_color = 'pink'
))
fig.update_layout(barmode = 'group', xaxis_tickangle = -45,
                  title = 'Monthwise comparision for Open Prices')
```

In [15]:
```
monthwise_high = eth.groupby(pd.DatetimeIndex(eth.Date).month)['High'].max()
monthwise_high = monthwise_high.reset_index()
monthwise_high['Date'] = new_order

monthwise_low = eth.groupby(pd.DatetimeIndex(eth.Date).month)['Low'].min()
monthwise_low = monthwise_low.reset_index()
```

In [16]:
```
fig = go.Figure()
fig.add_trace(go.Bar(
    x = monthwise_high.Date,
    y = monthwise_high.High,
    name = 'Stock High Price',
    marker_color = 'purple'
))
fig.add_trace(go.Bar(
    x = monthwise_low.Date,
    y = monthwise_low.Low,
    name = 'Stock Low Price',
    marker_color='pink'
))

fig.update_layout(barmode='group', xaxis_tickangle = -45,
                  title=' Monthwise High and Low Price')
```

In [17]:
```
names = cycle(['Eth Open Price','Eth High Price','Eth Low Price'])

fig = px.line(eth, x = eth.Date, y = [eth['Open'], eth['High'], eth['Low']],
              labels = {'date': 'Date','value':'Eth value'})
fig.update_layout(title_text = 'Ethereum Price analysis chart', font_size = 15
fig.for_each_trace(lambda t:  t.update(name = next(names)))
fig.update_xaxes(showgrid = False)
fig.update_yaxes(showgrid = False)

fig.show()
```

In [18]:
```
open_eth = eth[['Date', 'Open']]
print(open_eth.shape)
```

```
(2555, 2)
```

Out[18]:

	Date	Open
0	2023-03-08	1561.79
1	2023-03-07	1565.84
2	2023-03-06	1564.36
3	2023-03-05	1566.73
4	2023-03-04	1569.45

In [19]:
```
fig = px.line(open_eth, x = open_eth.Date, y = open_eth.Open,labels = {'date':
fig.update_traces(marker_line_width = 2, opacity = 0.8)
fig.update_layout(title_text = 'Stock close price chart', plot_bgcolor = 'whit
fig.update_xaxes(showgrid = False)
fig.update_yaxes(showgrid = False)
```

In [20]:
```
open_eth = open_eth[open_eth['Date'] > '2022-03-08']
open_stock = open_eth.copy()
```

```
Total data for prediction:  365
```

In [21]:
```
fig = px.line(open_stock, x = open_stock.Date, y = open_stock.Open, labels = {
fig.update_traces(marker_line_width = 2, opacity = 0.8, marker_line_color = 'o
fig.update_layout(title_text = 'Considered period to predict Ethereum close pr
fig.update_xaxes(showgrid = False)
fig.update_yaxes(showgrid = False)
fig.show()
```

In [22]:
```
del open_stock['Date']
scaler = MinMaxScaler(feature_range = (0,1))
open_stock = scaler.fit_transform(np.array(open_stock).reshape(-1,1))
```

```
(365, 1)
```

In [23]:
```
train_size = int(len(open_stock)*0.75)
test_size = len(open_stock) - train_size
train_data , test_data = open_stock[0:train_size, :] ,open_stock[train_size:le
print("Train_data :", train_data.shape)
```

```
Train_data : (273, 1)
Test_data : (92, 1)
```

In [24]:
```
def create_dataset(dataset, time_step = 1):
    dataX, dataY = [], []
    for i in range(len(dataset) - time_step - 1):
        a = dataset[i:(i + time_step), 0]
        dataX.append(a)
        dataY.append(dataset[i + time_step, 0])
```

In [25]:
```
time_step = 15
x_train, y_train = create_dataset(train_data, time_step)
x_test, y_test = create_dataset(test_data, time_step)

print("X_train: ", x_train.shape)
print("y_train: ", y_train.shape)
print("X_test: ", x_test.shape)
```

```
X_train:  (257, 15)
y_train:  (257,)
X_test:  (76, 15)
y_test (76,)
```

In [26]:
```
#Reshaping input to be of format [samples, time steps, features] which is reuq
x_train_lstm = x_train.reshape(x_train.shape[0], x_train.shape[1], 1)
x_test_lstm = x_test.reshape(x_test.shape[0], x_test.shape[1], 1)
```

```
(257, 15, 1) (76, 15, 1)
```

In [27]:
```
tf.keras.backend.clear_session()
model = Sequential()
model.add(GRU(32, return_sequences = True, input_shape = (time_step, 1)))
model.add(GRU(32, return_sequences = True))
model.add(GRU(32))
model.add(Dropout(0.20))
model.add(Dense(1))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[27], line 1
----> 1 tf.keras.backend.clear_session()
      2 model = Sequential()
      3 model.add(GRU(32, return_sequences = True, input_shape = (time_step,
1)))

NameError: name 'tf' is not defined
```

In [28]:
```
model.summary()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[28], line 1
----> 1 model.summary()

NameError: name 'model' is not defined
```

In [29]:
```
history = model.fit(x_train_lstm, y_train, validation_data = (x_test_lstm, y_t
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[29], line 1
----> 1 history = model.fit(x_train_lstm, y_train, validation_data = (x_test_
lstm, y_test), epochs = 200, batch_size = 32, verbose = 1)

NameError: name 'model' is not defined
```

In [30]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[30], line 1
----> 1 x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)

NameError: name 'train_test_split' is not defined
```

In [31]:
```
pd.concat([train_data['y_yes'], train_data.drop(['y_no', 'y_yes'], axis=1)], a
boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.join(prefix,
s3_input_train = sagemaker.inputs.TrainingInput(s3_data='s3://{}/{}/train'.for
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[31], line 1
----> 1 pd.concat([train_data['y_yes'], train_data.drop(['y_no', 'y_yes'], ax
is=1)], axis=1).to_csv('train.csv', index=False, header=False)
      2 boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.joi
n(prefix, 'train/train.csv')).upload_file('train.csv')
      3 s3_input_train = sagemaker.inputs.TrainingInput(s3_data='s3://{}/{}/t
rain'.format(bucket_name, prefix), content_type='csv')

IndexError: only integers, slices (`:`), ellipsis (`...`), numpy.newaxis (`No
ne`) and integer or boolean arrays are valid indices
```

In [32]:
```
pd.concat([train_data['y_yes'], train_data.drop(['y_no', 'y_yes'], axis=1)], a
boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.join(prefix,
s3_input_train = sagemaker.inputs.TrainingInput(s3_data='s3://{}/{}/train'.for
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[32], line 1
----> 1 pd.concat([train_data['y_yes'], train_data.drop(['y_no', 'y_yes'], ax
is=1)], axis=1).to_csv('train.csv', index=False, header=False)
      2 boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.joi
n(prefix, 'train/train.csv')).upload_file('train.csv')
      3 s3_input_train = sagemaker.inputs.TrainingInput(s3_data='s3://{}/{}/t
rain'.format(bucket_name, prefix), content_type='csv')

IndexError: only integers, slices (`:`), ellipsis (`...`), numpy.newaxis (`No
ne`) and integer or boolean arrays are valid indices
```

In [33]:
```
train_data, test_data = np.split(model_data.sample(frac=1, random_state=1729),
```

```
(1788, 6) (767, 6)
```

In [34]:
```
boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.join(prefix,
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
Cell In[34], line 1
----> 1 boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.joi
n(prefix, 'train/train.csv')).upload_file('train.csv')
      2 s3_input_train = sagemaker.inputs.TrainingInput(s3_data='s3://{}/{}/t
rain'.format(bucket_name, prefix), content_type='csv')

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/boto3/s3/inject.p
y:318, in object_upload_file(self, Filename, ExtraArgs, Callback, Config)
    287 def object_upload_file(
    288     self, Filename, ExtraArgs=None, Callback=None, Config=None
    289 ):
    290     """Upload a file to an S3 object.
    291
    292     Usage::
   (...)
    316         transfer.
    317     """
```

In [35]:
```
bucket_name='raw-sample-file'
boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.join(prefix,
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
Cell In[35], line 2
      1 bucket_name='raw-sample-file'
----> 2 boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.joi
n(prefix, 'train/train.csv')).upload_file('train.csv')
      3 s3_input_train = sagemaker.inputs.TrainingInput(s3_data='s3://{}/{}/t
rain'.format(bucket_name, prefix), content_type='csv')

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/boto3/s3/inject.p
y:318, in object_upload_file(self, Filename, ExtraArgs, Callback, Config)
    287 def object_upload_file(
    288     self, Filename, ExtraArgs=None, Callback=None, Config=None
    289 ):
    290     """Upload a file to an S3 object.
    291
    292     Usage::
   (...)
    316         transfer.
    317     """
```

In [36]:
```
import boto3
s3 = boto3.resource('s3')
for key in bucket.objects.all():
```

```
  Cell In[36], line 4
    print 's3://{}/{}.format(bucket,key.key)
          ^
SyntaxError: unterminated string literal (detected at line 4)
```

In [37]:
```
import boto3
s3 = boto3.resource('s3')
for key in bucket.objects.all():
```

```
  Cell In[37], line 4
    print 's3://{}.format(bucket,key.key)
          ^
SyntaxError: unterminated string literal (detected at line 4)
```

In [38]:
```
import boto3
s3 = boto3.resource('s3')
for key in bucket.objects.all():
    print key
```

```
  Cell In[38], line 4
    print key
    ^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print(...)?
```

In [39]:
```
import boto3
s3 = boto3.resource('s3')
for key in bucket.objects.all():
    print(key)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[39], line 3
      1 import boto3
      2 s3 = boto3.resource('s3')
----> 3 for key in bucket.objects.all():
      4   print(key)

NameError: name 'bucket' is not defined
```

In [40]:
```
import boto3
s3 = boto3.resource('s3')
for my_bucket_object in s3.objects.all():
    print(my_bucket_object)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[40], line 3
      1 import boto3
      2 s3 = boto3.resource('s3')
----> 3 for my_bucket_object in s3.objects.all():
      4     print(my_bucket_object)

AttributeError: 's3.ServiceResource' object has no attribute 'objects'
```

In [41]:
```
import boto3
s3 = boto3.resource('s3')
my_bucket = s3.Bucket('raw-sample-file')

for file in my_bucket.objects.all():
    print(file.key)
```

```
ethereum_price.csv
train/
```

In [42]:
```
bucket_name='raw-sample-file'
boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.join(prefix,
s3_input_train = sagemaker.inputs.TrainingInput(s3_data='s3:/train'.format(buc
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
Cell In[42], line 2
      1 bucket_name='raw-sample-file'
----> 2 boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.joi
n(prefix, 'train/train.csv')).upload_file('train.csv')
      3 s3_input_train = sagemaker.inputs.TrainingInput(s3_data='s3:/train'.f
ormat(bucket_name, prefix), content_type='csv')

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/boto3/s3/inject.p
y:318, in object_upload_file(self, Filename, ExtraArgs, Callback, Config)
    287 def object_upload_file(
    288     self, Filename, ExtraArgs=None, Callback=None, Config=None
    289 ):
    290     """Upload a file to an S3 object.
    291
    292     Usage::
   (...)
    316         transfer.
    317     """
```

In [43]:
```
bucket_name='raw-sample-file'
boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.join(prefix,
s3_input_train = sagemaker.inputs.TrainingInput(s3_data='s3://{}/{}'.format(bu
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
Cell In[43], line 2
      1 bucket_name='raw-sample-file'
----> 2 boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.joi
n(prefix, 'train/train.csv')).upload_file('train.csv')
      3 s3_input_train = sagemaker.inputs.TrainingInput(s3_data='s3://{}/{}'.
format(bucket_name, prefix), content_type='csv')

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/boto3/s3/inject.p
y:318, in object_upload_file(self, Filename, ExtraArgs, Callback, Config)
    287 def object_upload_file(
    288     self, Filename, ExtraArgs=None, Callback=None, Config=None
    289 ):
    290     """Upload a file to an S3 object.
    291
    292     Usage::
   (...)
    316         transfer.
    317     """
```

In [44]:
```
train_data, test_data = np.split(model_data.sample(frac=1, random_state=1729),
```

```
(1788, 6) (767, 6)
```

In [45]:
```
bucket_name='raw-sample-file'
boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.join(prefix,
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
Cell In[45], line 2
      1 bucket_name='raw-sample-file'
----> 2 boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.joi
n(prefix, 'train/train.csv')).upload_file('train.csv')
      3 s3_input_train = sagemaker.inputs.TrainingInput(s3_data='s3://{}/{}'.
format(bucket_name, prefix), content_type='csv')

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/boto3/s3/inject.p
y:318, in object_upload_file(self, Filename, ExtraArgs, Callback, Config)
    287 def object_upload_file(
    288     self, Filename, ExtraArgs=None, Callback=None, Config=None
    289 ):
    290     """Upload a file to an S3 object.
    291
    292     Usage::
   (...)
    316         transfer.
    317     """
```

In [ ]:
```
bucket_name='raw-sample-file'
boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.join(prefix,
s3_input_train = sagemaker.inputs.TrainingInput(s3_data='s3://{}/{}'.format(bu
```

In [46]:
```
sess = sagemaker.Session()
xgb = sagemaker.estimator.Estimator(xgboost_container,role, instance_count=1,
xgb.set_hyperparameters(max_depth=5,eta=0.2,gamma=4,min_child_weight=6,subsamp
```

In [47]:
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[47], line 1
----> 1 xgb.fit({'train': s3_input_train})

NameError: name 's3_input_train' is not defined
```

In [48]:
```
sess = sagemaker.Session()
xgb = sagemaker.estimator.Estimator(xgboost_container,role, instance_count=1,
xgb.set_hyperparameters(max_depth=5,eta=0.2,gamma=4,min_child_weight=6,subsamp
```

In [49]:

In [50]:

```
INFO:sagemaker:Creating training-job with name: xgboost-2023-05-30-09-18-24-2
32

---------------------------------------------------------------------------
ResourceLimitExceeded                     Traceback (most recent call last)
Cell In[50], line 1
----> 1 xgb.fit({'train': s3_input_train})

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/sagemaker/workflow
/pipeline_context.py:284, in runnable_by_pipeline.<locals>.wrapper(*args, **k
wargs)
    280         return context
    282     return _StepArguments(retrieve_caller_name(self_instance), run_fu
nc, *args, **kwargs)
--> 284 return run_func(*args, **kwargs)

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/sagemaker/estimato
r.py:1195, in EstimatorBase.fit(self, inputs, wait, logs, job_name, experimen
t_config)
   1192 self._prepare_for_training(job_name=job_name)
   1194 experiment_config = check_and_get_run_experiment_config(experiment_co
nfig)
-> 1195 self.latest_training_job = _TrainingJob.start_new(self, inputs, exper
iment_config)
   1196 self.jobs.append(self.latest_training_job)
   1197 if wait:

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/sagemaker/estimato
r.py:2131, in _TrainingJob.start_new(cls, estimator, inputs, experiment_confi
g)
   2106 """Create a new Amazon SageMaker training job from the estimator.
   2107
   2108 Args:
   (...)
   2127     all information about the started training job.
   2128 """
   2129 train_args = cls._get_train_args(estimator, inputs, experiment_confi
g)
-> 2131 estimator.sagemaker_session.train(**train_args)
   2133 return cls(estimator.sagemaker_session, estimator._current_job_name)

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/sagemaker/session.
py:848, in Session.train(self, input_mode, input_config, role, job_name, outp
ut_config, resource_config, vpc_config, hyperparameters, stop_condition, tag
s, metric_definitions, enable_network_isolation, image_uri, training_image_co
nfig, algorithm_arn, encrypt_inter_container_traffic, use_spot_instances, che
ckpoint_s3_uri, checkpoint_local_path, experiment_config, debugger_rule_confi
gs, debugger_hook_config, tensorboard_output_config, enable_sagemaker_metric
s, profiler_rule_configs, profiler_config, environment, retry_strategy)
    845     LOGGER.debug("train request: %s", json.dumps(request, indent=4))
    846     self.sagemaker_client.create_training_job(**request)
--> 848 self._intercept_create_request(train_request, submit, self.train.__na
me__)
```

```
File ~/anaconda3/envs/python3/lib/python3.10/site-packages/sagemaker/session.
py:5375, in Session._intercept_create_request(self, request, create, func_nam
e)
   5358 def _intercept_create_request(
   5359     self,
   5360     request: typing.Dict,
   (...)
   5363     # pylint: disable=unused-argument
   5364 ):
   5365     """This function intercepts the create job request.
   5366
   5367     PipelineSession inherits this Session class and will override
   (...)
   5373         func_name (str): the name of the function needed intercepting
   5374     """
-> 5375     return create(request)

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/sagemaker/session.
py:846, in Session.train.<locals>.submit(request)
    844 LOGGER.info("Creating training-job with name: %s", job_name)
    845 LOGGER.debug("train request: %s", json.dumps(request, indent=4))
--> 846 self.sagemaker_client.create_training_job(**request)

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/botocore/client.p
y:530, in ClientCreator._create_api_method.<locals>._api_call(self, *args, **
kwargs)
    526     raise TypeError(
    527         f"{py_operation_name}() only accepts keyword arguments."
    528     )
    529 # The "self" in this scope is referring to the BaseClient.
--> 530 return self._make_api_call(operation_name, kwargs)

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/botocore/client.p
y:960, in BaseClient._make_api_call(self, operation_name, api_params)
    958     error_code = parsed_response.get("Error", {}).get("Code")
    959     error_class = self.exceptions.from_code(error_code)
--> 960     raise error_class(parsed_response, operation_name)
    961 else:
    962     return parsed_response

ResourceLimitExceeded: An error occurred (ResourceLimitExceeded) when calling
the CreateTrainingJob operation: The account-level service limit 'ml.m4.xlarg
e for training job usage' is 0 Instances, with current utilization of 0 Insta
nces and a request delta of 1 Instances. Please use AWS Service Quotas to req
uest an increase for this quota. If AWS Service Quotas is not available, cont
act AWS support to request an increase for this quota.
```

In [51]:
```
print(f"{len(eth)} rows")

eth["Date"] = pd.to_datetime(eth['Date'])

last_date = eth["Date"].max()

print(f"Latest row is from {last_date}")
```

```
2555 rows
Latest row is from 2023-03-08 00:00:00
```

Out[51]:

	Date	Price	Open	High	Low	Vol.	Change %
0	2023-03-08	1553.49	1561.79	1569.7	1548.98	498570	-0.53
1	2023-03-07	1561.78	1565.84	1580.95	1536.31	460100	-0.26
2	2023-03-06	1565.84	1564.36	1581.13	1555.43	322160	0.09
3	2023-03-05	1564.37	1566.73	1587.95	1556.84	313010	-0.15
4	2023-03-04	1566.73	1569.45	1577.02	1550.1	247020	-0.14

In [52]:
```
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2555 entries, 0 to 2554
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype         
---  ------    --------------  -----         
 0   Date      2555 non-null   datetime64[ns]
 1   Price     2555 non-null   object        
 2   Open      2555 non-null   object        
 3   High      2555 non-null   object        
 4   Low       2555 non-null   object        
 5   Vol.      2555 non-null   object        
 6   Change %  2555 non-null   object        
dtypes: datetime64[ns](1), object(6)
memory usage: 139.9+ KB
```

In [53]:

```
/tmp/ipykernel_7421/3711191218.py:1: FutureWarning:

Treating datetime data as categorical rather than numeric in `.describe` is d
eprecated and will be removed in a future version of pandas. Specify `datetim
e_is_numeric=True` to silence this warning and adopt the future behavior now.
```

Out[53]:

	Date	Price	Open	High	Low	Vol.	Change %
count	2555	2555.00	2555.00	2555.00	2555.00	2555.0	2555.0
unique	2555	2442.00	2434.00	2436.00	2428.00	2016.0	1333.0
top	2023-03-08 00:00:00	10.16	10.16	10.16	10.16	1420000.0	0.0
freq	1	9.00	8.00	7.00	7.00	10.0	14.0
first	2016-03-10 00:00:00	NaN	NaN	NaN	NaN	NaN	NaN
last	2023-03-08 00:00:00	NaN	NaN	NaN	NaN	NaN	NaN

In [54]:

```
eth.plot(kind="line", x="Date", y="Close", figsize=(12,6))
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File ~/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/core/indexes/base.py:3803, in Index.get_loc(self, key, method, tolerance)
   3802 try:
-> 3803     return self._engine.get_loc(casted_key)
   3804 except KeyError as err:

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/_libs/index.pyx:138, in pandas._libs.index.IndexEngine.get_loc()

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/_libs/index.pyx:165, in pandas._libs.index.IndexEngine.get_loc()

File pandas/_libs/hashtable_class_helper.pxi:5745, in pandas._libs.hashtable.PyObjectHashTable.get_item()

File pandas/_libs/hashtable_class_helper.pxi:5753, in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'Close'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[54], line 1
----> 1 eth.plot(kind="line", x="Date", y="Close", figsize=(12,6))

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/plotting/_core.py:986, in PlotAccessor.__call__(self, *args, **kwargs)
    983             pass
    985 # don't overwrite
--> 986 data = data[y].copy()
    988 if isinstance(data, ABCSeries):
    989     label_name = label_kw or y

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/core/frame.py:3805, in DataFrame.__getitem__(self, key)
   3803 if self.columns.nlevels > 1:
   3804     return self._getitem_multilevel(key)
-> 3805 indexer = self.columns.get_loc(key)
   3806 if is_integer(indexer):
   3807     indexer = [indexer]

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/core/indexes/base.py:3805, in Index.get_loc(self, key, method, tolerance)
   3803     return self._engine.get_loc(casted_key)
   3804 except KeyError as err:
-> 3805     raise KeyError(key) from err
   3806 except TypeError:
   3807     # If we have a listlike key, _check_indexing_error will raise
```

```
3808     #  InvalidIndexError. Otherwise we fall through and re-raise
3809     #  the TypeError.
3810     self._check_indexing_error(key)

KeyError: 'Close'
```

In [55]:

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File ~/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/core/indexe
s/base.py:3803, in Index.get_loc(self, key, method, tolerance)
   3802 try:
-> 3803     return self._engine.get_loc(casted_key)
   3804 except KeyError as err:

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/_libs/inde
x.pyx:138, in pandas._libs.index.IndexEngine.get_loc()

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/_libs/inde
x.pyx:165, in pandas._libs.index.IndexEngine.get_loc()

File pandas/_libs/hashtable_class_helper.pxi:5745, in pandas._libs.hashtable.
PyObjectHashTable.get_item()

File pandas/_libs/hashtable_class_helper.pxi:5753, in pandas._libs.hashtable.
PyObjectHashTable.get_item()

KeyError: 'Close'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[55], line 1
----> 1 eth.plot(kind="line", x="Date", y="Close", figsize=(12,6))

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/plotting/_c
ore.py:986, in PlotAccessor.__call__(self, *args, **kwargs)
    983             pass
    985 # don't overwrite
--> 986 data = data[y].copy()
    988 if isinstance(data, ABCSeries):
    989     label_name = label_kw or y

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/core/frame.
py:3805, in DataFrame.__getitem__(self, key)
   3803 if self.columns.nlevels > 1:
   3804     return self._getitem_multilevel(key)
-> 3805 indexer = self.columns.get_loc(key)
   3806 if is_integer(indexer):
   3807     indexer = [indexer]

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/core/indexe
s/base.py:3805, in Index.get_loc(self, key, method, tolerance)
   3803     return self._engine.get_loc(casted_key)
   3804 except KeyError as err:
-> 3805     raise KeyError(key) from err
   3806 except TypeError:
   3807     # If we have a listlike key, _check_indexing_error will raise
   3808     #  InvalidIndexError. Otherwise we fall through and re-raise
   3809     #  the TypeError.
   3810     self._check_indexing_error(key)
```

```
KeyError: 'Close'
```

In [56]:

```
prophet_data = eth[["Date", "Close"]]

prophet_data = prophet_data.rename(columns = {
    "Date": "ds",
    "Close": "y"
})

prophet_data.head()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[56], line 1
----> 1 prophet_data = eth[["Date", "Close"]]
      3 prophet_data = prophet_data.rename(columns = {
      4     "Date": "ds",
      5     "Close": "y"
      6 })
      8 prophet_data.head()

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/core/frame.
py:3811, in DataFrame.__getitem__(self, key)
   3809     if is_iterator(key):
   3810         key = list(key)
-> 3811     indexer = self.columns._get_indexer_strict(key, "columns")[1]
   3813 # take() does not accept boolean indexers
   3814 if getattr(indexer, "dtype", None) == bool:

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/core/indexe
s/base.py:6113, in Index._get_indexer_strict(self, key, axis_name)
   6110 else:
   6111     keyarr, indexer, new_indexer = self._reindex_non_unique(keyarr)
-> 6113 self._raise_if_missing(keyarr, indexer, axis_name)
   6115 keyarr = self.take(indexer)
   6116 if isinstance(key, Index):
   6117     # GH 42790 - Preserve name from an Index

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/core/indexe
s/base.py:6176, in Index._raise_if_missing(self, key, indexer, axis_name)
   6173     raise KeyError(f"None of [{key}] are in the [{axis_name}]")
   6175 not_found = list(ensure_index(key)[missing_mask.nonzero()[0]].unique
())
-> 6176 raise KeyError(f"{not_found} not in index")

KeyError: "['Close'] not in index"
```

In [57]:

```
from prophet import Prophet

prophet = Prophet(daily_seasonality=True)

prophet.fit(prophet_data)

print("Data fitted")
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
Cell In[57], line 1
----> 1 from prophet import Prophet
      3 prophet = Prophet(daily_seasonality=True)
      5 prophet.fit(prophet_data)

ModuleNotFoundError: No module named 'prophet'
```

In [58]:

```
import matplotlib as mpl
import matplotlib.pyplot as plt

fig = plt.figure(dpi=100)

fig.set_facecolor("white")

prophet_plot_forecast_fig = prophet.plot(forecast, ax=fig.gca());

prophet_plot_forecast_fig.savefig('forecast_details.png')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[58], line 8
      4 fig = plt.figure(dpi=100)
      6 fig.set_facecolor("white")
----> 8 prophet_plot_forecast_fig = prophet.plot(forecast, ax=fig.gca());
     10 prophet_plot_forecast_fig.savefig('forecast_details.png')

NameError: name 'prophet' is not defined


<Figure size 640x480 with 0 Axes>
```

In [ ]: