仮想通貨に関しての時系列解析

Ran YI

Department of Mathematical Sciences
Ritsumeikan University

January 10, 2025

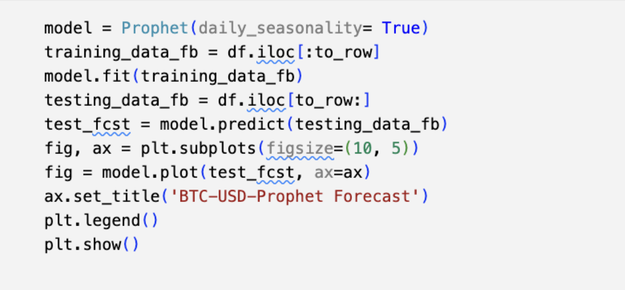
データの準備

2018 年 1 月 1 日から 2025 年 1 月 1 日までの BTC-USD データを使っている

```
df = pd.read_csv('bitcoin_data.csv')
df = df[['Date','Close']]
df = df.rename(columns={'Date':'ds', 'Close':'y'})
to_row = int(len(df)*0.9)
training_data = list(df['y'][:to_row])  # トレーニングデータ
testing_data = list(df['y'][to_row:])     # テストデータ
plt.figure(figsize=(10,6))
plt.grid(True)
plt.xlabel('ds')
plt.ylabel('y')
plt.plot(df['y'][:to_row],'green',label='Train data')
plt.plot(df['y'][to_row:],'blue',label = 'Test data')
plt.legend()
plt.show()
```

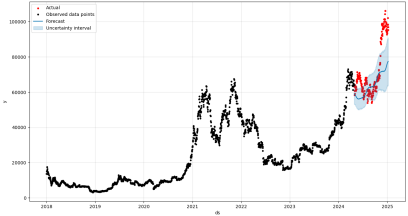
Facebook Prophet モデルの設定

Facebook Prophet の時系列解析モデル
(詳細: https://facebook.github.io/prophet/)

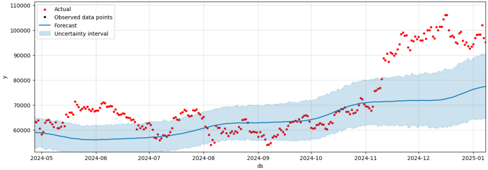


```
model = Prophet(daily_seasonality= True)
training_data_fb = df.iloc[:to_row]
model.fit(training_data_fb)
testing_data_fb = df.iloc[to_row:]
test_fcst = model.predict(testing_data_fb)
fig, ax = plt.subplots(figsize=(10, 5))
fig = model.plot(test_fcst, ax=ax)
ax.set_title('BTC-USD-Prophet Forecast')
plt.legend()
plt.show()
```

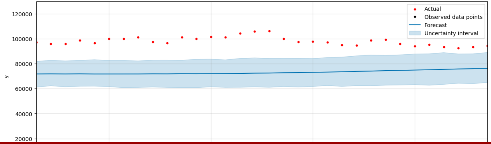
一般化予測



Forecast vs Actuals



12/24~01/25 Forecast vs Actuals



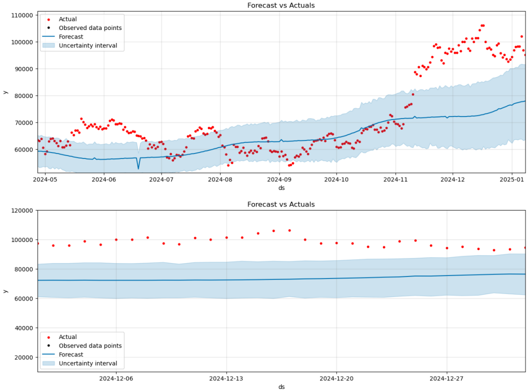
年月日のトレンド


90000
85000
80000
75000
70000
65000
60000
trend
2024-05
2024-06
2024-07
2024-08
2024-09
2024-10
2024-11
2024-12
2025-01
ds
60
40
20
0
−20
−40
weekly
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Day of week
6000
4000
2000
0
−2000
−4000
yearly
January 1
March 1
May 1
July 1
September 1
November 1
January 1
Day of year
1000
0
−1000
−2000
−3000
daily
00:00:00
03:25:42
06:51:25
10:17:08
13:42:51
17:08:34
20:34:17
00:00:00
Hour of day

モデルの評価

- MSE Value(平均二乗誤差): 12724.452579794137
- MAE Value(平均絶対誤差): 9289.974856441853
- MAPE Value(平均絶対パーセント誤差): 11.582111797512047

祝日含みでのモデル訓練



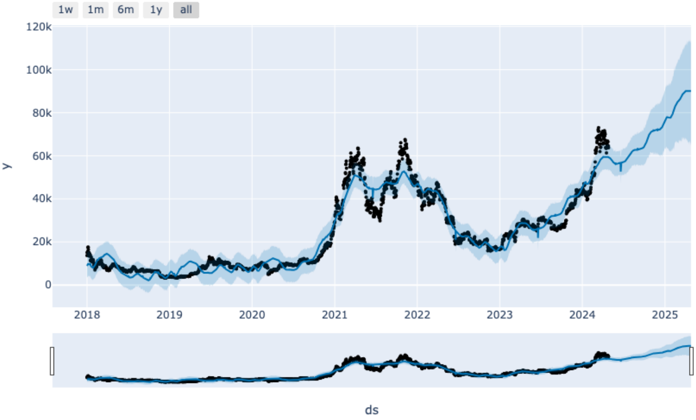
祝日含み版


trend
90000
85000
80000
75000
70000
65000
60000
55000
2024-05
2024-06
2024-07
2024-08
2024-09
2024-10
2024-11
2024-12
2025-01
ds
holidays
0
−1000
−2000
−3000
−4000
2024-05
2024-06
2024-07
2024-08
2024-09
2024-10
2024-11
2024-12
2025-01
ds
weekly
60
40
20
0
−20
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Day of week
6000
4000
2000

祝日版 Prophet モデル評価

- SE Value(平均二乗誤差): 12550.97617585529
- MAE Value(平均絶対誤差): 9187.666839253901
- MAPE Value(平均絶対パーセント誤差): 11.468800365741414

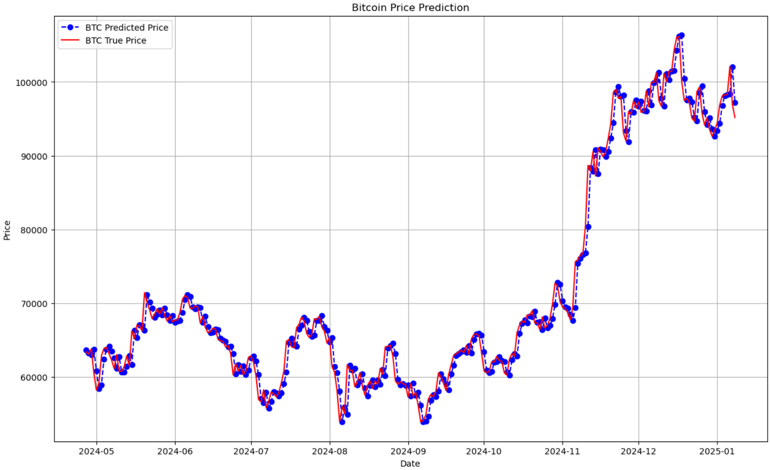
未来１年の予測


1w
1m
6m
1y
all
120k
100k
80k
60k
40k
20k
0
y
2018
2019
2020
2021
2022
2023
2024
2025
ds

訓練データとテストデータの大分け

訓練データ: 全体データの 0.9
テストデータ: 全体データ 0.1

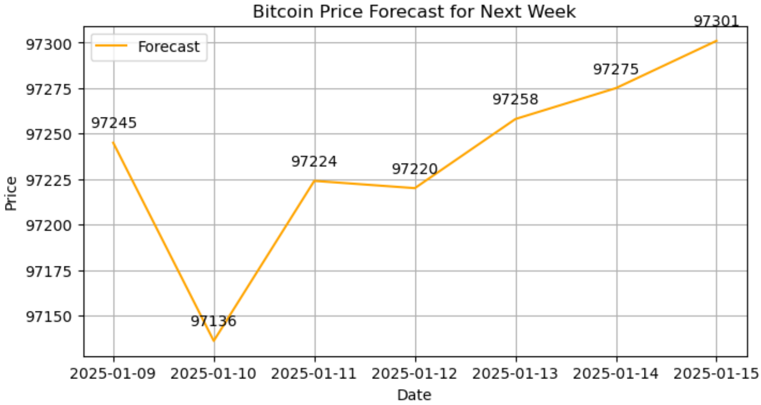
訓練した ARIMA モデルとテストデータの対比



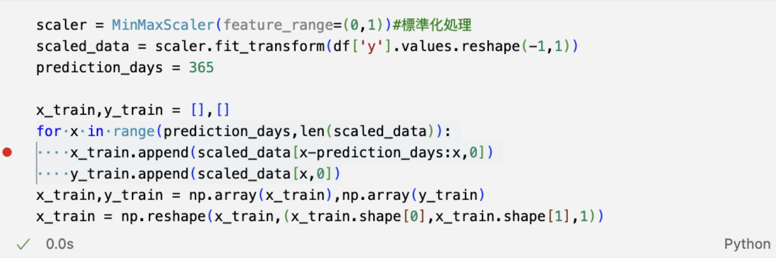
ARIMA モデルの評価

- MAPE Value :1.94、すなわち、正確率は 0.9806 である

ARIMA モデルでの未来 1 週間の価格予測



データの標準化

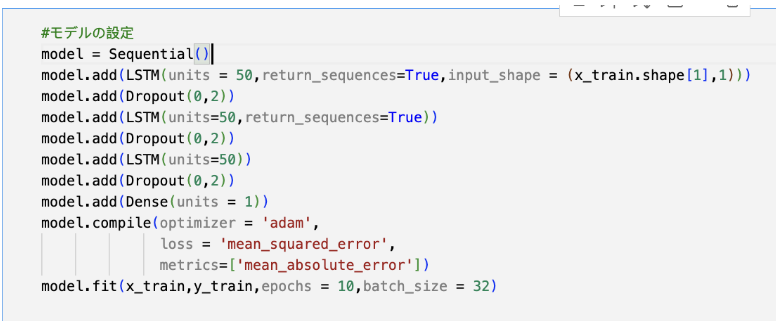


```python
scaler = MinMaxScaler(feature_range=(0,1))#標準化処理
scaled_data = scaler.fit_transform(df['y'].values.reshape(-1,1))
prediction_days = 365

x_train,y_train = [],[]
for x in range(prediction_days,len(scaled_data)):
    x_train.append(scaled_data[x-prediction_days:x,0])
    y_train.append(scaled_data[x,0])
x_train,y_train = np.array(x_train),np.array(y_train)
x_train = np.reshape(x_train,(x_train.shape[0],x_train.shape[1],1))
```

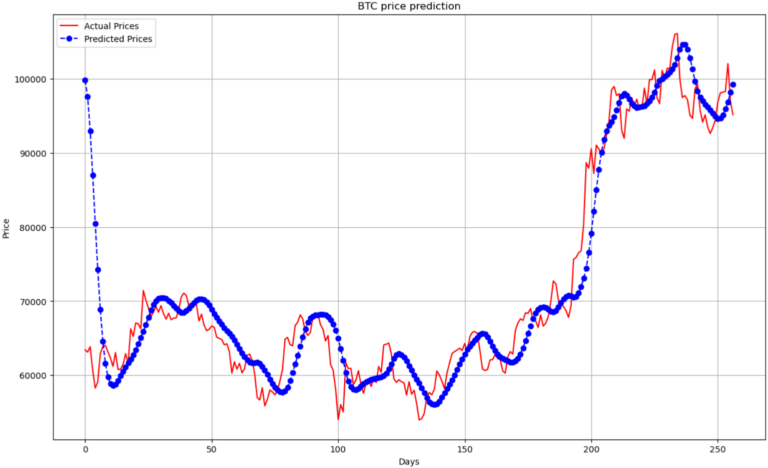
✓ 0.0s Python

LSTM モデルの設定



```
#モデルの設定
model = Sequential()
model.add(LSTM(units = 50,return_sequences=True,input_shape = (x_train.shape[1],1)))
model.add(Dropout(0,2))
model.add(LSTM(units=50,return_sequences=True))
model.add(Dropout(0,2))
model.add(LSTM(units=50))
model.add(Dropout(0,2))
model.add(Dense(units = 1))
model.compile(optimizer = 'adam',
              loss = 'mean_squared_error',
              metrics=['mean_absolute_error'])
model.fit(x_train,y_train,epochs = 10,batch_size = 32)
```

訓練した LSTM モデルとテストデータの対比



LSTM モデルの評価

- MAPE Value:20、すなわち、正確率は 0.8 である

- Prophet モデルの紹介は (https://www.skillupai.com/blog/tech/prophet/no1) を参考してください
- Prophet の原理は (https://zhuanlan.zhihu.com/p/463183142) とコードの原理 (https://zhuanlan.zhihu.com/p/52330017) を参考してください
- ARIMA モデルの原理の紹介は (https://datastudy.gonna.jp/arima/) を参考してください
- LSTM モデルの紹介は (https://qiita.com/KojiOhki/items/89cd7b69a8a6239d67ca) を参考してください