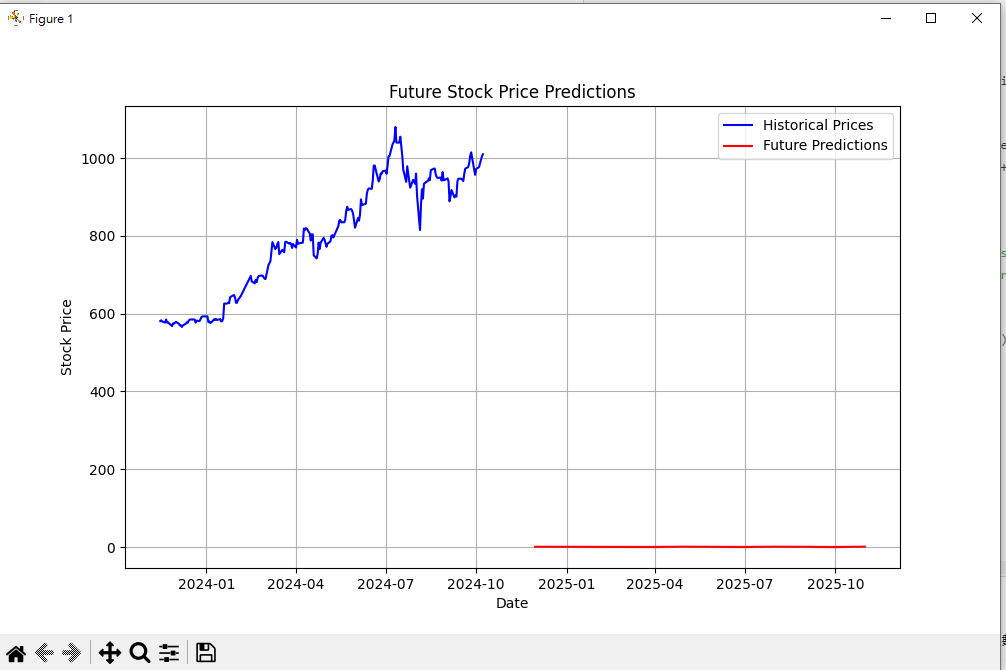
根據檔案2330-training.csv，用python去 solve "auto regression" problem，要照著CRISP-datamining(DM)的步驟，1.)將檔案2330-training.csv中的A column 當作附圖中的x軸並標示年分，2.)用檔案2330-training.csv中的B到G column各別代表的x1到x6當作"auto regression" problem中的輸入，並預測出y軸，也就是附圖的當日收盤價，做出像附圖一樣的圖，並要包含 1. feature selection 2. model evaluation 3. web implementation deployment 4. 預測未來12個月分別的收盤價並產生預測範圍的漏斗圖 (websim)   
  
為甚麼我的預測圖中沒有2023年份的資料  
  
為甚麼我的未來預測曲線(紅線)的部分是一條平行線，並沒有顯示該有的曲折線  
  
跑出來的圖示這樣的，並沒有漏斗圖，而且未來預測也在0元  
  
import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error

import matplotlib.pyplot as plt

# 讀取資料集

data = pd.read\_csv('你的資料集.csv')

# 移除數字中的逗號，並轉換為浮點數

data[['x1', 'x2', 'x3', 'x4', 'x5', 'x6']] = data[['x1', 'x2', 'x3', 'x4', 'x5', 'x6']].replace({',': ''}, regex=True).astype(float)

# 確保日期欄位為 datetime 格式

data['Date'] = pd.to\_datetime(data['Date'], format='%m/%d/%Y')

# 定義自變量和應變量

X = data[['x1', 'x2', 'x3', 'x4', 'x5', 'x6']]

y = data['x1'] # 假設 'x1' 是目標變量，根據你的需求可以調整

# 將資料集分割為訓練集和測試集

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# 訓練模型

model = LinearRegression()

model.fit(X\_train, y\_train)

# 預測並評估模型

y\_pred = model.predict(X\_test)

rmse = mean\_squared\_error(y\_test, y\_pred, squared=False)

print(f'RMSE: {rmse}')

# 繪製結果圖

plt.figure(figsize=(10, 6))

plt.plot(data['Date'], data['x1'], color='blue', label='Historical Prices') # 繪製歷史資料

# 預測未來的12個月數據（假設每個月有一個觀測值）

future\_dates = pd.date\_range(start=data['Date'].iloc[-1] + pd.DateOffset(months=1), periods=12, freq='M')

future\_X = data[['x1', 'x2', 'x3', 'x4', 'x5', 'x6']].iloc[-1].values.reshape(1, -1) # 使用最新資料進行預測

future\_predictions = []

# 進行未來12個月的預測

for \_ in range(12):

future\_pred = model.predict(future\_X)

future\_predictions.append(future\_pred[0])

# 這裡假設你會根據未來預測的值更新 x1 並進行下一步預測，具體邏輯可以根據需求修改

future\_X[0][0] = future\_pred[0] # 假設 x1 是你想要預測的未來數值

# 繪製未來預測

plt.plot(future\_dates, future\_predictions, color='red', label='Future Predictions')

# 圖形標題和標籤

plt.title('Future Stock Price Predictions')

plt.xlabel('Date')

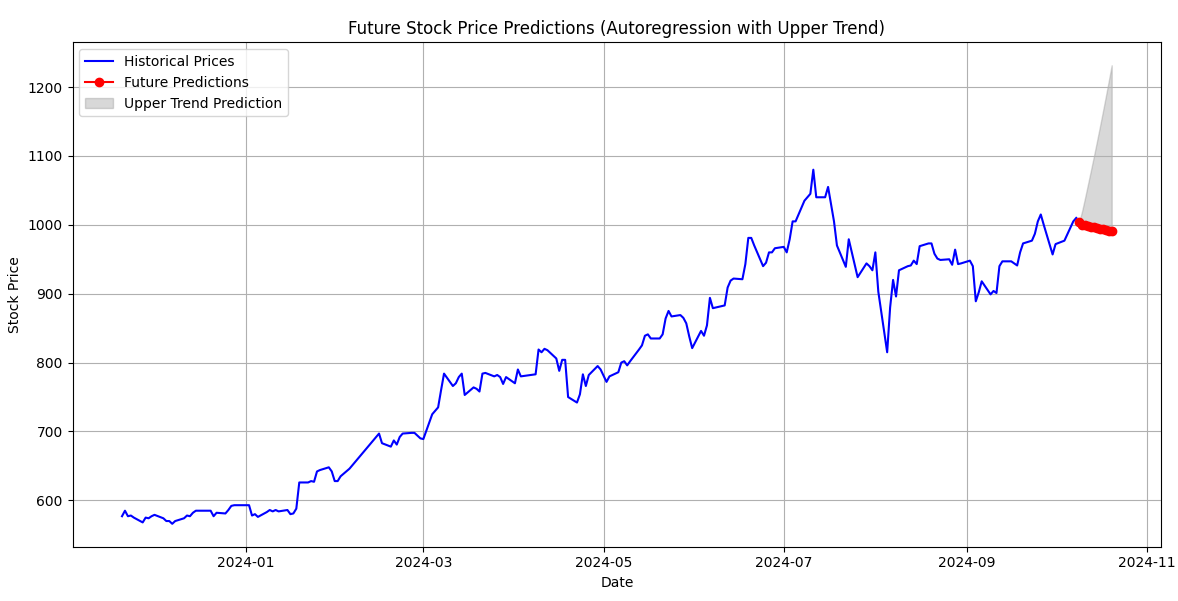
plt.ylabel('Stock Price')

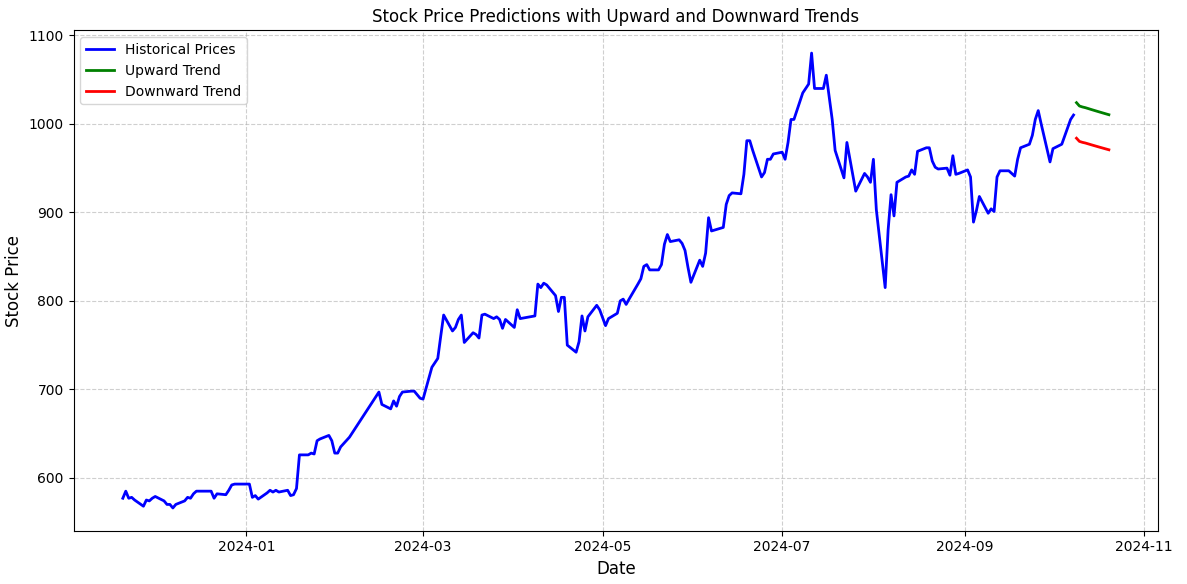
plt.legend()

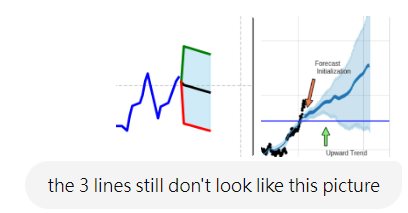
# 顯示圖表

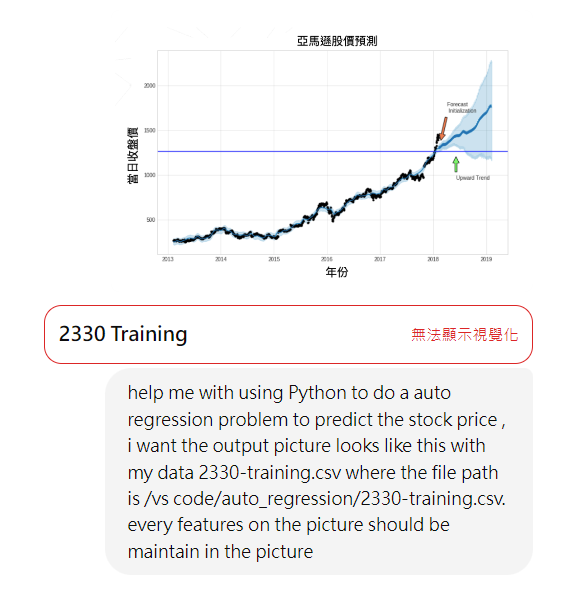
plt.show()

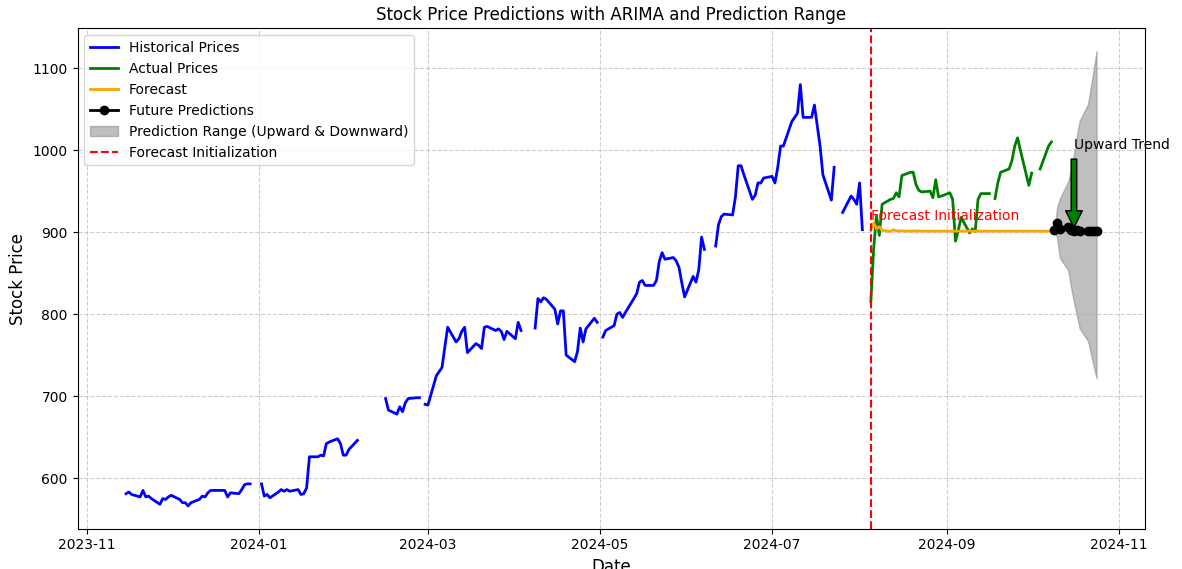
help me with using Python to do a auto regression problem to predict the stock price here is my code:  
  
i want to change the prediction part, i want the gray area become the upper trend prediction

  
this is the result, but i want it looks like the second picture  
  
can i have the upper trend like the second picture on my last question,which the upper trend is cover by the area of two prediction line,one is the stock price rise line at the top and the second line is the downward stock price line at the bottom

i want to change my question, i want 2 line after the historical line : 1. the upward trend 2. the downward trend  
  
why the two line is straight , i want this two line be the prediction lines  
  
  
this is the result, i wand the green line looks like the tail part in second picture which green line(upward trend line) is the upper bound of the blue part of the tail part in second picture , and the red line(downward trend line) is the lower bound of the blue part of the tail part in second picture  
  
i want the green、black、red line start at the same start point after the historical line which is the initialization forecast

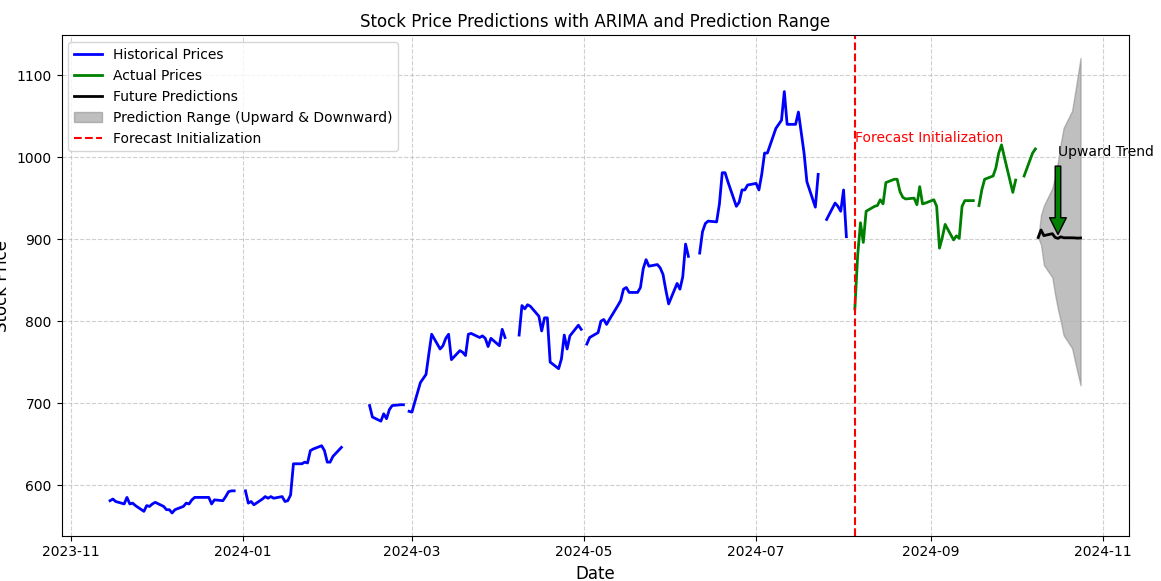






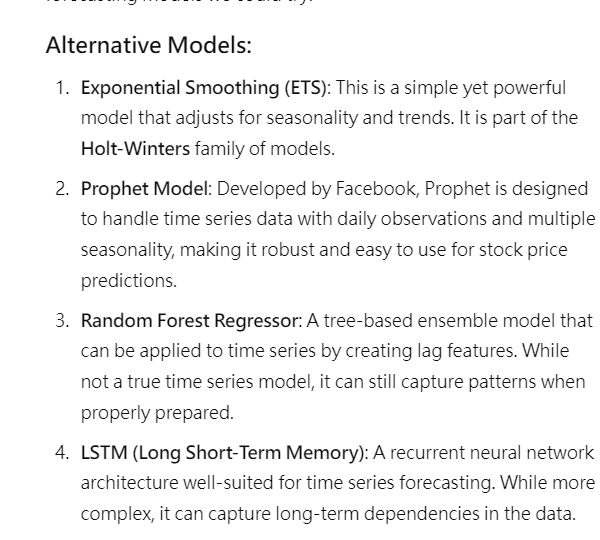
what's the difference between historical and actual price

i want to delete the forecast line( orange line) and change the future predicting line( black line with dots) into black line without dots

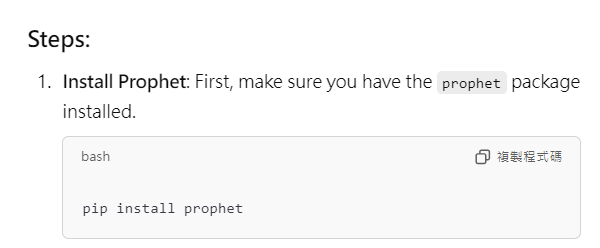


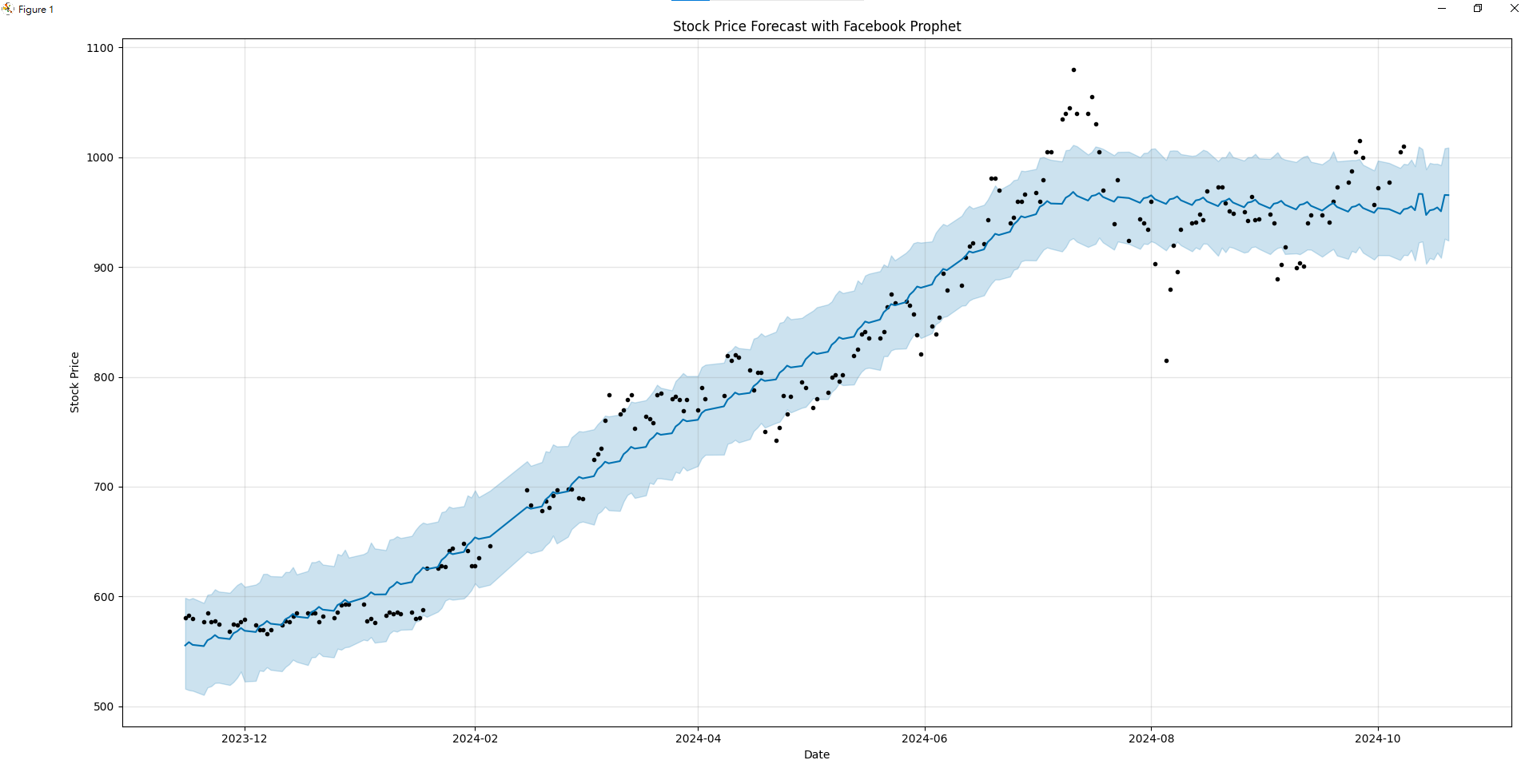
why the historical line and actual price line of my result have space interval

Can we try a different forecasting model?



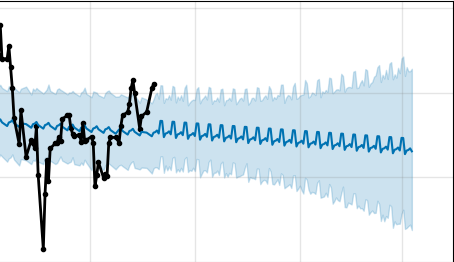
Let's try the Facebook Prophet model.





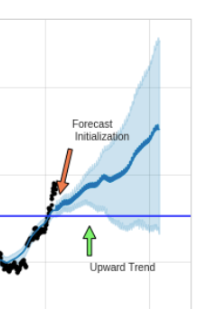
can you line all the black dots

i want to forecast 5 months after the data

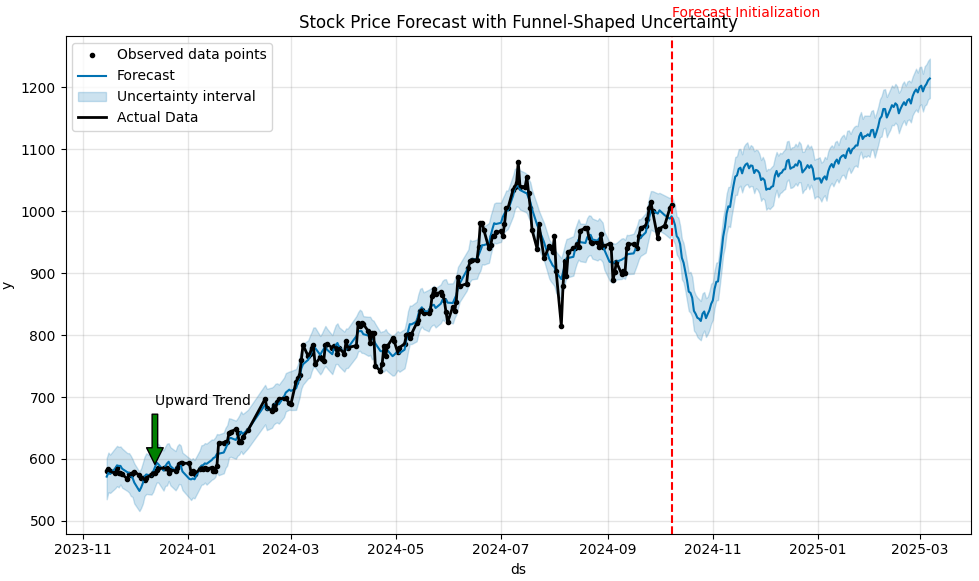


why the forecast part looks like the same

i want the uncertainty interval has the feature which the closer the time is to now, the prediction interval will be more convergent; conversely, the further the time is from now, the larger the prediction interval will be, indicating that the uncertainty will be higher.



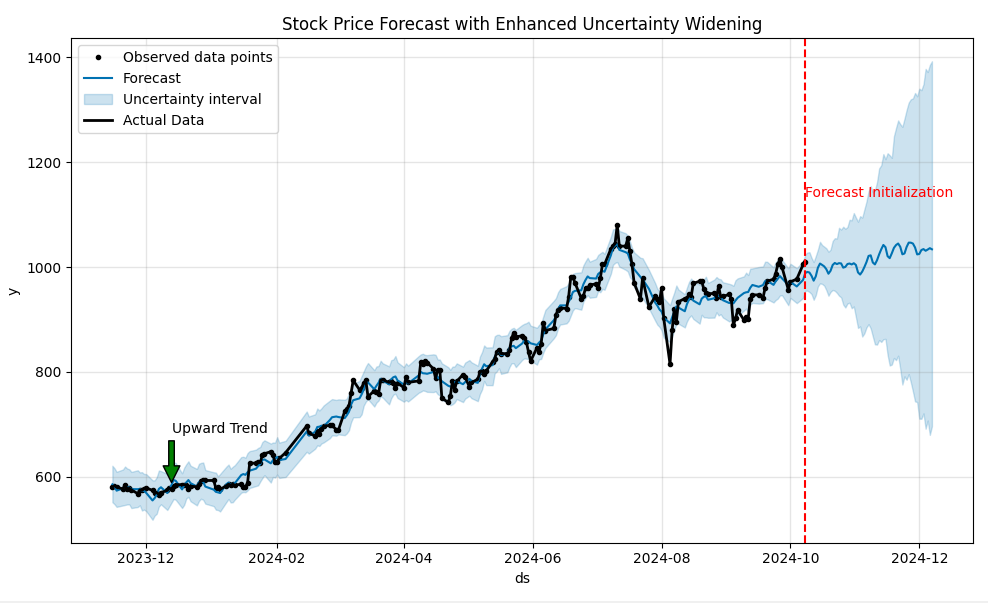
i want the uncertainty looks like the funnel like this picture



this is my result, why the uncertainty interval doesn't have this feature :The closer the time is to now, the prediction interval will be more convergent; conversely, the further the time is from now, the larger the prediction interval will be, indicating that the uncertainty will be higher.

model = Prophet(interval\_width=0.95, changepoint\_prior\_scale=0.5, uncertainty\_samples=10) what's every parameters in the code

Final Result:

  
  
import pandas as pd

import matplotlib.pyplot as plt

from prophet import Prophet

# Load the data

file\_path = '/vs code/auto\_regression/2330-training.csv'

data = pd.read\_csv(file\_path)

# Convert 'Date' to datetime

data['Date'] = pd.to\_datetime(data['Date'], format='%m/%d/%Y')

# Clean data: Remove commas from numeric columns and convert them to float

for col in ['x1', 'x2', 'x3', 'x4', 'x5', 'x6']:

data[col] = data[col].replace({',': ''}, regex=True).astype(float)

# Prepare the data for Prophet model (Prophet requires 'ds' and 'y' columns)

data\_prophet = pd.DataFrame()

data\_prophet['ds'] = data['Date'] # Prophet requires the date column to be named 'ds'

data\_prophet['y'] = data['x1'] # 'y' is the target column we want to forecast (e.g., closing price 'x1')

# Initialize Prophet model with higher uncertainty sampling and changepoint sensitivity

model = Prophet(interval\_width=0.95, changepoint\_prior\_scale=0.5, uncertainty\_samples=500)

model.add\_seasonality(name='monthly', period=30.5, fourier\_order=5) # Adding monthly seasonality

# Fit the model to the historical data

model.fit(data\_prophet)

# Make a DataFrame to hold future dates (2 months = ~60 days)

future\_dates = model.make\_future\_dataframe(periods=60)

# Predict future values

forecast = model.predict(future\_dates)

# Plot the forecasted data with uncertainty intervals

fig, ax = plt.subplots(figsize=(10, 6)) # Single plot

model.plot(forecast, ax=ax)

ax.set\_title('Stock Price Forecast with Enhanced Uncertainty Widening')

# Overlay the historical data (black dots) as a line for continuity

ax.plot(data\_prophet['ds'], data\_prophet['y'], color='black', lw=2, label='Actual Data')

# Add annotations for clarity (e.g., Forecast Initialization and Upward Trend)

ax.axvline(x=data\_prophet['ds'].iloc[-1], color='red', linestyle='--') # Forecast Initialization

ax.text(data\_prophet['ds'].iloc[-1], forecast['yhat'].iloc[-1] + 100, 'Forecast Initialization', color='red')

ax.annotate('Upward Trend', xy=(future\_dates.iloc[20], forecast['yhat'].iloc[20]),

xytext=(future\_dates.iloc[20], forecast['yhat'].iloc[20] + 100),

arrowprops=dict(facecolor='green', shrink=0.05))

# Add legend for clarity

ax.legend()

# Show the plot

plt.tight\_layout()

plt.show()