Stock Price Forecasting Exponential Smoothing

This Python code performs a time series analysis on stock price data using exponential smoothing to forecast future stock prices. Let's go through each step in detail:

Step 1: Importing necessary libraries

python

Copy code

import pandas as pd

import numpy as np

import statsmodels.api as sm

import matplotlib.pyplot as plt

This step imports the required libraries: pandas for data manipulation, numpy for numerical computations, statsmodels for regression analysis, and matplotlib for data visualization.

NOTE:  
To install all the required imports in the code, you can use the following commands:

1. Pandas: pip install pandas

2. NumPy: pip install numpy

3. Statsmodels: pip install statsmodels

4. Matplotlib: pip install matplotlib

Make sure you have Python and pip installed on your system before running these commands. Once you have installed the required packages, you should be able to run the code without any import-related errors.

Step 2: Data Gathering and Preprocessing

python

Copy code

data = pd.read\_csv("MTNL.csv")

data['Date'] = pd.to\_datetime(data['Date'])

data = data.set\_index('Date')

In this step, the code reads the stock price data from a CSV file named "MTNL.csv" and stores it in a pandas DataFrame called data. The 'Date' column is converted to a datetime format using pd.to\_datetime() and set as the index for the DataFrame using data.set\_index().

Step 3: Calculating Returns

python

Copy code

returns = (data["Adj Close"] - data["Adj Close"].shift(1)) / data["Adj Close"].shift(1)

data["Returns"] = returns

The code calculates the daily returns for the stock prices using the formula: (Current day's adjusted close - Previous day's adjusted close) / Previous day's adjusted close. The calculated returns are added as a new column "Returns" in the DataFrame.

Step 4: Regression Analysis and Estimating Alpha and Beta

python

Copy code

Y = data["Returns"].dropna()

X = data["Adj Close"].shift(1).dropna()

X = sm.add\_constant(X)

model = sm.OLS(Y, X).fit()

alpha = model.params[0]

beta = model.params[1]

In this step, the code prepares the data for regression analysis. The dependent variable Y is the "Returns" column, and the independent variable X is the lagged "Adj Close" column, which represents the stock prices from the previous day.

The code uses the statsmodels library to perform Ordinary Least Squares (OLS) regression with sm.OLS(). The fit() method is called to obtain the regression model. The regression model estimates the intercept (alpha) and slope (beta) coefficients, which are stored in the variables alpha and beta, respectively.

Step 5: Exponential Smoothing and Forecasting

python

Copy code

def exponential\_smoothing(stock\_prices, alpha, beta, periods):

# ... function implementation ...

return smoothed\_values, forecasted\_values

stock\_prices = data["Adj Close"].tolist()

smoothed\_prices, forecasted\_stock\_prices = exponential\_smoothing(stock\_prices, alpha, beta, periods=24)

This step defines a function exponential\_smoothing() that takes the stock prices, alpha, beta, and the number of periods to forecast as inputs. It performs exponential smoothing on the stock prices and returns the smoothed values and the forecasted values.

The exponential\_smoothing() function uses the initial stock price (stock\_prices[0]) as the first smoothed value and iterates through the remaining stock prices to calculate the smoothed values based on the exponential smoothing formula. Then, it uses the same formula to forecast future stock prices for the specified number of periods.

The function returns two lists: smoothed\_values, which contains the smoothed stock prices for the historical data, and forecasted\_values, which contains tuples of (date, forecasted price) for the future periods.

Step 6: Visualize the Results

python

Copy code

plt.figure(figsize=(12, 6))

plt.plot(data.index, data["Adj Close"], label="Actual Stock Prices", color='blue')

plt.plot(data.index, smoothed\_prices[:len(data)], label="Smoothed Stock Prices", color='green', linestyle='dashed')

forecast\_dates = pd.date\_range(start=data.index[-1], periods=25, freq='M')[1:]

forecasted\_prices = [price for \_, price in forecasted\_stock\_prices]

plt.plot(forecast\_dates, forecasted\_prices, label="Forecasted Stock Prices", color='red')

plt.xlabel("Date")

plt.ylabel("Stock Price")

plt.title("Actual vs. Forecasted Stock Prices")

plt.legend()

plt.show()

This step visualizes the actual stock prices, smoothed stock prices, and forecasted stock prices using matplotlib. The plt.plot() function is used to plot the data on a line chart. The historical stock prices are plotted in blue, the smoothed stock prices in green (dashed line), and the forecasted stock prices in red.

The x-axis represents the dates, and the y-axis represents the stock prices. The chart also includes labels, titles, and a legend to provide additional context.

In summary, this code reads historical stock price data, calculates the daily returns, performs a regression analysis to estimate alpha and beta, applies exponential smoothing to forecast future stock prices, and finally visualizes the results using a line chart.