Changlin Yi-cy2578

FRE-6991 HW1

1. Setup and Imports

In [3]:
```
import pandas as pd
import numpy as np
import yfinance as yf
import statsmodels.api as sm
import datetime as dt
```

2. Define Date Ranges

In [5]:
```
# Daily range
daily_start = dt.datetime(2024, 3, 31)
daily_end   = dt.datetime(2025, 3, 31)

# Weekly range
weekly_start = dt.datetime(2019, 3, 31)
weekly_end   = dt.datetime(2025, 3, 31)

# Monthly range
monthly_start = dt.datetime(2018, 3, 31)
monthly_end   = dt.datetime(2025, 3, 31)
```

3. Function to Download Data and Calculate Returns

In [7]:
```
def get_returns(ticker, start_date, end_date, freq='D'):

    # Download data
    data = yf.download([ticker, 'SPY'], start=start_date, end=end_date, prog

    # In case data is a Series (if single ticker), make sure it is a DataFra
    if isinstance(data, pd.Series):
        data = data.to_frame()

    # Rename columns if needed
    data.columns = [f"{col}" for col in data.columns]

    # Resample
    if freq == 'D':
        # Already daily by default, so just forward-fill if missing
        data = data.asfreq('B').fillna(method='ffill')  # 'B' = business day
    elif freq == 'W':
        data = data.resample('W-FRI').last()  # For weekly, take last price
    elif freq == 'M':
```

```
        data = data.resample('M').last()      # For monthly, take last price

    # Calculate returns
    df = data.pct_change().dropna()
    df.columns = ['Stock', 'SPY'] if len(df.columns) == 2 else df.columns

    # Rename columns to reflect returns
    df.rename(columns={df.columns[0]: 'Stock_Return',
                       df.columns[1]: 'SPY_Return'}, inplace=True)

    return df
```

4. Function to Regress Stock Returns on SPY Returns

In [9]:
```
def regress_stock_on_spy(df_returns):
    """
    Performs a linear regression of Stock_Return on SPY_Return.
    Returns alpha, beta, and R-squared.
    """
    X = df_returns[['SPY_Return']]  # Independent variable
    y = df_returns['Stock_Return']  # Dependent variable

    # Add a constant term to the regression
    X = sm.add_constant(X)

    # Fit the regression model
    model = sm.OLS(y, X).fit()

    alpha = model.params['const']
    beta = model.params['SPY_Return']
    r_squared = model.rsquared

    return alpha, beta, r_squared, model
```

5. Output Data for Apple

In [11]:
```
# AAPL Daily
aapl_daily = get_returns('AAPL', daily_start, daily_end, freq='D')
alpha_d, beta_d, r2_d, model_d = regress_stock_on_spy(aapl_daily)

print("AAPL Daily Regression (3/31/2024 - 3/31/2025)")
print(f"Alpha: {alpha_d:.6f}")
print(f"Beta:  {beta_d:.6f}")
print(f"R^2:   {r2_d:.6f}")

# AAPL Weekly
aapl_weekly = get_returns('AAPL', weekly_start, weekly_end, freq='W')
alpha_w, beta_w, r2_w, model_w = regress_stock_on_spy(aapl_weekly)

print("AAPL Weekly Regression (3/31/2019 - 3/31/2025)")
print(f"Alpha: {alpha_w:.6f}")
print(f"Beta:  {beta_w:.6f}")
print(f"R^2:   {r2_w:.6f}")
```

```
# AAPL Monthly
aapl_monthly = get_returns('AAPL', monthly_start, monthly_end, freq='M')
alpha_m, beta_m, r2_m, model_m = regress_stock_on_spy(aapl_monthly)

print("AAPL Monthly Regression (3/31/2018 - 3/31/2025)")
print(f"Alpha: {alpha_m:.6f}")
print(f"Beta:  {beta_m:.6f}")
print(f"R^2:   {r2_m:.6f}")
```

```
YF.download() has changed argument auto_adjust default to True
/var/folders/9n/q6k7950n7xg1s3hmhsgf07fh0000gn/T/ipykernel_11861/2537916345.
py:36: FutureWarning: DataFrame.fillna with 'method' is deprecated and will
raise in a future version. Use obj.ffill() or obj.bfill() instead.
  data = data.asfreq('B').fillna(method='ffill')  # 'B' = business day
AAPL Daily Regression (3/31/2024 - 3/31/2025)
Alpha: 0.000777
Beta:  0.965103
R^2:   0.297749
AAPL Weekly Regression (3/31/2019 - 3/31/2025)
Alpha: 0.002702
Beta:  1.085479
R^2:   0.526366
AAPL Monthly Regression (3/31/2018 - 3/31/2025)
Alpha: 0.010212
Beta:  1.238905
R^2:   0.514004
/var/folders/9n/q6k7950n7xg1s3hmhsgf07fh0000gn/T/ipykernel_11861/2537916345.
py:40: FutureWarning: 'M' is deprecated and will be removed in a future vers
ion, please use 'ME' instead.
  data = data.resample('M').last()      # For monthly, take last price of th
e month
```

6. Output Data for PFE

In [13]:

```
# PFE Daily
pfe_daily = get_returns('PFE', daily_start, daily_end, freq='D')
alpha_pd, beta_pd, r2_pd, model_pd = regress_stock_on_spy(pfe_daily)

print("PFE Daily Regression (3/31/2024 - 3/31/2025)")
print(f"Alpha: {alpha_pd:.6f}")
print(f"Beta:  {beta_pd:.6f}")
print(f"R^2:   {r2_pd:.6f}")

# PFE Weekly
pfe_weekly = get_returns('PFE', weekly_start, weekly_end, freq='W')
alpha_pw, beta_pw, r2_pw, model_pw = regress_stock_on_spy(pfe_weekly)

print("PFE Weekly Regression (3/31/2019 - 3/31/2025)")
print(f"Alpha: {alpha_pw:.6f}")
print(f"Beta:  {beta_pw:.6f}")
print(f"R^2:   {r2_pw:.6f}")

# PFE Monthly
pfe_monthly = get_returns('PFE', monthly_start, monthly_end, freq='M')
```

```
alpha_pm, beta_pm, r2_pm, model_pm = regress_stock_on_spy(pfe_monthly)

print("PFE Monthly Regression (3/31/2018 - 3/31/2025)")
print(f"Alpha: {alpha_pm:.6f}")
print(f"Beta:  {beta_pm:.6f}")
print(f"R^2:   {r2_pm:.6f}")
```

```
/var/folders/9n/q6k7950n7xg1s3hmhsgf07fh0000gn/T/ipykernel_11861/2537916345.
py:36: FutureWarning: DataFrame.fillna with 'method' is deprecated and will
raise in a future version. Use obj.ffill() or obj.bfill() instead.
  data = data.asfreq('B').fillna(method='ffill')  # 'B' = business day
PFE Daily Regression (3/31/2024 - 3/31/2025)
Alpha: -0.000102
Beta:  0.207161
R^2:   0.016229
PFE Weekly Regression (3/31/2019 - 3/31/2025)
Alpha: -0.001597
Beta:  0.540766
R^2:   0.165488
PFE Monthly Regression (3/31/2018 - 3/31/2025)
Alpha: -0.004636
Beta:  0.595104
R^2:   0.162786
/var/folders/9n/q6k7950n7xg1s3hmhsgf07fh0000gn/T/ipykernel_11861/2537916345.
py:40: FutureWarning: 'M' is deprecated and will be removed in a future vers
ion, please use 'ME' instead.
  data = data.resample('M').last()      # For monthly, take last price of th
e month
```

7. Output Data for TSLA

In [15]:
```
# TSLA Daily
tsla_daily = get_returns('TSLA', daily_start, daily_end, freq='D')
alpha_td, beta_td, r2_td, model_td = regress_stock_on_spy(tsla_daily)

print("TSLA Daily Regression (3/31/2024 - 3/31/2025)")
print(f"Alpha: {alpha_td:.6f}")
print(f"Beta:  {beta_td:.6f}")
print(f"R^2:   {r2_td:.6f}")

# TSLA Weekly
tsla_weekly = get_returns('TSLA', weekly_start, weekly_end, freq='W')
alpha_tw, beta_tw, r2_tw, model_tw = regress_stock_on_spy(tsla_weekly)

print("TSLA Weekly Regression (3/31/2019 - 3/31/2025)")
print(f"Alpha: {alpha_tw:.6f}")
print(f"Beta:  {beta_tw:.6f}")
print(f"R^2:   {r2_tw:.6f}")

# TSLA Monthly
tsla_monthly = get_returns('TSLA', monthly_start, monthly_end, freq='M')
alpha_tm, beta_tm, r2_tm, model_tm = regress_stock_on_spy(tsla_monthly)

print("TSLA Monthly Regression (3/31/2018 - 3/31/2025)")
print(f"Alpha: {alpha_tm:.6f}")
```

```
print(f"Beta:  {beta_tm:.6f}")
print(f"R^2:   {r2_tm:.6f}")
```

```
/var/folders/9n/q6k7950n7xg1s3hmhsgf07fh0000gn/T/ipykernel_11861/2537916345.
py:36: FutureWarning: DataFrame.fillna with 'method' is deprecated and will
raise in a future version. Use obj.ffill() or obj.bfill() instead.
  data = data.asfreq('B').fillna(method='ffill')  # 'B' = business day
TSLA Daily Regression (3/31/2024 - 3/31/2025)
Alpha: 0.000045
Beta:  0.113756
R^2:   0.318853
TSLA Weekly Regression (3/31/2019 - 3/31/2025)
Alpha: 0.000716
Beta:  0.159514
R^2:   0.318101
TSLA Monthly Regression (3/31/2018 - 3/31/2025)
Alpha: 0.005775
Beta:  0.114315
R^2:   0.225451
/var/folders/9n/q6k7950n7xg1s3hmhsgf07fh0000gn/T/ipykernel_11861/2537916345.
py:40: FutureWarning: 'M' is deprecated and will be removed in a future vers
ion, please use 'ME' instead.
  data = data.resample('M').last()      # For monthly, take last price of th
e month
```

In [ ]: