# 1.) Pull in Data and Convert ot Monthly

In [72]:
```
import yfinance as yf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [73]:
```
apple_data = yf.download('AAPL')
df = apple_data.resample("M").last()[["Adj Close"]]
```

```
[**********************100%%**********************]  1 of 1 completed
```

# 2.) Create columns.

- Current Stock Price, Difference in stock price, Whether it went up or down over the next month, option premium

In [74]:
```
df['Diff']=df['Adj Close'].diff().shift(-1)
df['Target']=np.sign(df['Diff'])
df['Permium']=0.08*df['Adj Close']
```

In [75]:
```
df
```

Out[75]:

| Date | Adj Close | Diff | Target | Permium |
|---|---|---|---|---|
| **1980-12-31** | 0.117887 | -0.020296 | -1.0 | 0.009431 |
| **1981-01-31** | 0.097591 | -0.006045 | -1.0 | 0.007807 |
| **1981-02-28** | 0.091546 | -0.006909 | -1.0 | 0.007324 |
| **1981-03-31** | 0.084637 | 0.013386 | 1.0 | 0.006771 |
| **1981-04-30** | 0.098023 | 0.016409 | 1.0 | 0.007842 |
| **...** | ... | ... | ... | ... |
| **2023-09-30** | 170.984741 | -0.439423 | -1.0 | 13.678779 |
| **2023-10-31** | 170.545319 | 19.404678 | 1.0 | 13.643625 |
| **2023-11-30** | 189.949997 | 2.580002 | 1.0 | 15.196000 |
| **2023-12-31** | 192.529999 | -9.850006 | -1.0 | 15.402400 |
| **2024-01-31** | 182.679993 | NaN | NaN | 14.614399 |

518 rows × 4 columns

In [76]:
```
df['Target']
```

Out[76]:
```
Date
1980-12-31   -1.0
1981-01-31   -1.0
1981-02-28   -1.0
1981-03-31    1.0
1981-04-30    1.0
             ...
2023-09-30   -1.0
2023-10-31    1.0
2023-11-30    1.0
2023-12-31   -1.0
2024-01-31    NaN
Freq: M, Name: Target, Length: 518, dtype: float64
```

# 3.) Pull in X data, normalize and build a LogReg on column 2

In [77]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
```

In [78]:
```
X = pd.read_csv("Xdata.csv", index_col="Date", parse_dates=["Date"])
```

In [79]:
```
y = df.loc[:"2023-09-30","Target"].copy()
df=df.loc[:"2023-09-30", :].copy()
```

In [80]:
```
y
```

Out[80]:
```
Date
1980-12-31   -1.0
1981-01-31   -1.0
1981-02-28   -1.0
1981-03-31    1.0
1981-04-30    1.0
             ...
2023-05-31    1.0
2023-06-30    1.0
2023-07-31   -1.0
2023-08-31   -1.0
2023-09-30   -1.0
Freq: M, Name: Target, Length: 514, dtype: float64
```

In [81]:
```
X
```

Out[81]:

| Date | VAR1 |
|---|---|
| **1980-12-31** | 0.163261 |
| **1981-01-31** | 0.437449 |
| **1981-02-28** | -0.334994 |
| **1981-03-31** | 2.550820 |
| **1981-04-30** | 3.170655 |
| **...** | ... |
| **2023-05-31** | 2.330573 |
| **2023-06-30** | 3.033257 |
| **2023-07-31** | 1.007072 |
| **2023-08-31** | 0.504651 |
| **2023-09-30** | 0.669328 |

514 rows × 1 columns

In [82]:
```
log_reg=LogisticRegression()
log_reg.fit(X,y)
y_pred=log_reg.predict(X)
```

# 4.) Add columns, prediction and profits.

In [83]:
```
df['prediction']=y_pred
```

In [84]:
```
df['profit']=0.
#TP
df.loc[(df['prediction']==1)&(df['Target']==1),'profit']=df['Permium']
#FP
df.loc[(df['prediction']==1)&(df['Target']==-1),'profit']=100*df['Diff']+df['Permium']
```

In [85]:
```
df
```

Out[85]:

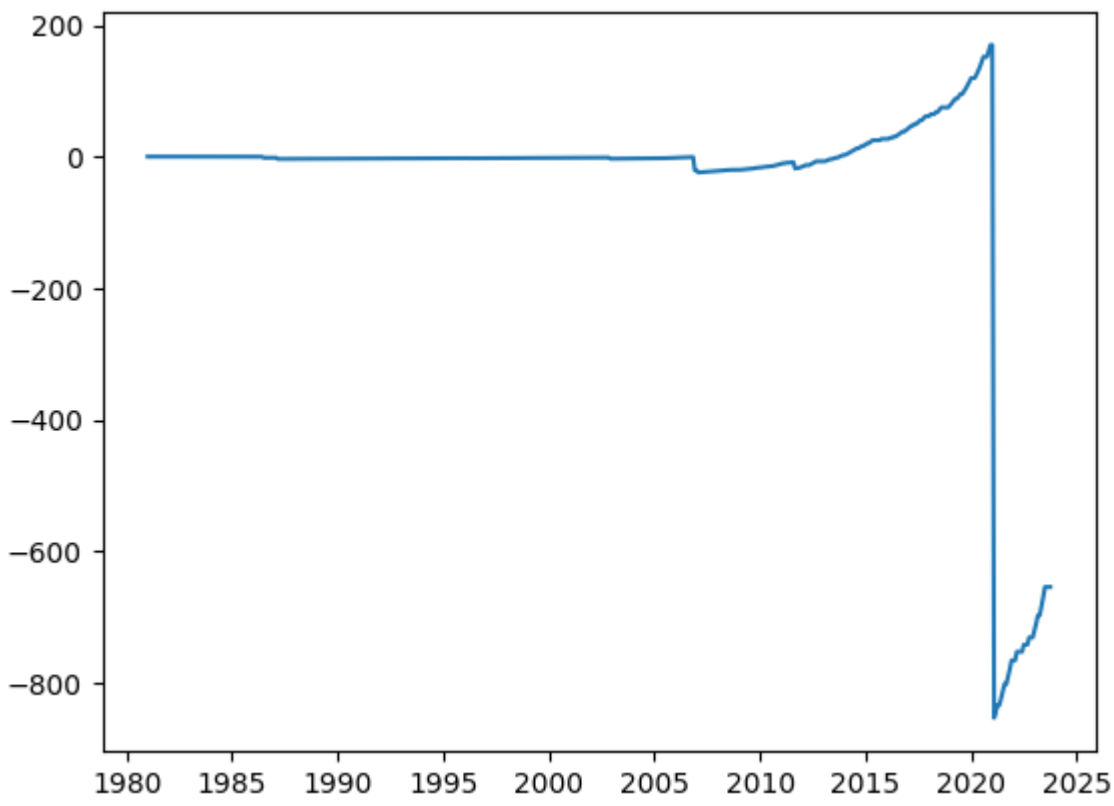
| Date | Adj Close | Diff | Target | Permium | prediction | profit |
|---|---|---|---|---|---|---|
| **1980-12-31** | 0.117887 | -0.020296 | -1.0 | 0.009431 | -1.0 | 0.000000 |
| **1981-01-31** | 0.097591 | -0.006045 | -1.0 | 0.007807 | -1.0 | 0.000000 |
| **1981-02-28** | 0.091546 | -0.006909 | -1.0 | 0.007324 | -1.0 | 0.000000 |
| **1981-03-31** | 0.084637 | 0.013386 | 1.0 | 0.006771 | 1.0 | 0.006771 |
| **1981-04-30** | 0.098023 | 0.016409 | 1.0 | 0.007842 | 1.0 | 0.007842 |
| **...** | ... | ... | ... | ... | ... | ... |
| **2023-05-31** | 176.778076 | 16.675476 | 1.0 | 14.142246 | 1.0 | 14.142246 |
| **2023-06-30** | 193.453552 | 2.473404 | 1.0 | 15.476284 | 1.0 | 15.476284 |
| **2023-07-31** | 195.926956 | -8.304138 | -1.0 | 15.674156 | -1.0 | 0.000000 |
| **2023-08-31** | 187.622818 | -16.638077 | -1.0 | 15.009825 | -1.0 | 0.000000 |
| **2023-09-30** | 170.984741 | -0.439423 | -1.0 | 13.678779 | -1.0 | 0.000000 |

514 rows × 6 columns

## 5.) Plot profits over time

In [86]:
```
plt.plot(np.cumsum(df['profit']))
```

Out[86]: [<matplotlib.lines.Line2D at 0x176118fd0>]




Mr. Luis has established a system for individuals who purchase tickets to gain access to exclusive chat rooms. This access enables them to interact with key opinion leaders (KOLs) or "stars" within intimate group settings. It offers a platform for individuals with shared interests to engage in conversations. From our end, the expertise we've developed from the MQE program can be applied to utilize the data obtained from the website StarArena or from content creators. This data can be organized and processed through models specifically designed for tailored interactions. The information collected will aid in future decision-making processes, and we can employ logistic regression to identify and target the specific interests of website users.

## 6.) Create a loop that stores total profits over time

In [ ]:
```

```

## 7.) What is the optimal threshold and plot the total profits for this model.

In [ ]:
```

```