1.) Pull in Data and Convert ot Monthly

In [2]:
```
import yfinance as yf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [3]:
```
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 [5]:
```
df['Diff'] = df.diff().shift(-1)
```

In [6]:
```
df['Target'] = np.sign(df['Diff'])
```

In [7]:
```
df['Premium'] = 0.08 * df['Adj Close']
```

In [8]:
```
df.head()
```

Out[8]:

Date	Adj Close	Diff	Target	Premium
1980-12-31	0.117887	-0.020296	-1.0	0.009431
1981-01-31	0.097592	-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

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

In [10]:
```
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 [11]:
```
X = pd.read_csv("Xdata.csv", index_col="Date", parse_dates=["Date"])
```

In [15]:
```
y = df.loc[:"2023-09-30","Target"].copy()
df = df.loc[:'2023-09-30',:].copy()
```

In [13]:
```
logreg = LogisticRegression()
logreg.fit(X, y)
y_pred = logreg.predict(X)
```

In [ ]:
```

```

4.) Add columns, prediction and profits.

In [16]:
```
df['Predictions'] = y_pred
```

In [22]:
```
df['Profits'] = 0
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File ~\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3653, in Index.get_loc
(self, key)
   3652 try:
-> 3653     return self._engine.get_loc(casted_key)
   3654 except KeyError as err:

File ~\anaconda3\Lib\site-packages\pandas\_libs\index.pyx:147, in pandas._libs.index.
IndexEngine.get_loc()

File ~\anaconda3\Lib\site-packages\pandas\_libs\index.pyx:176, in pandas._libs.index.
IndexEngine.get_loc()

File pandas\_libs\hashtable_class_helper.pxi:7080, in pandas._libs.hashtable.PyObject
HashTable.get_item()

File pandas\_libs\hashtable_class_helper.pxi:7088, in pandas._libs.hashtable.PyObject
HashTable.get_item()

KeyError: 'Premiums'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[22], line 2
      1 df['Profits'] = 0
----> 2 df['Premiums']

File ~\anaconda3\Lib\site-packages\pandas\core\frame.py:3761, in DataFrame.__getitem_
_(self, key)
   3759 if self.columns.nlevels > 1:
   3760     return self._getitem_multilevel(key)
-> 3761 indexer = self.columns.get_loc(key)
   3762 if is_integer(indexer):
   3763     indexer = [indexer]

File ~\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3655, in Index.get_loc
(self, key)
   3653     return self._engine.get_loc(casted_key)
   3654 except KeyError as err:
-> 3655     raise KeyError(key) from err
   3656 except TypeError:
   3657     # If we have a listlike key, _check_indexing_error will raise
   3658     #  InvalidIndexError. Otherwise we fall through and re-raise
   3659     #  the TypeError.
   3660     self._check_indexing_error(key)

KeyError: 'Premiums'
```

In [28]:
```
df.loc[(df['Predictions'] == 1) & (df['Target'] == 1), 'Profits'] = df['Premium']
df.loc[(df['Predictions'] == 1) & (df['Target'] == -1), 'Profits'] = df['Diff']*100 +
```

In [27]:
```
df.head(20)
```

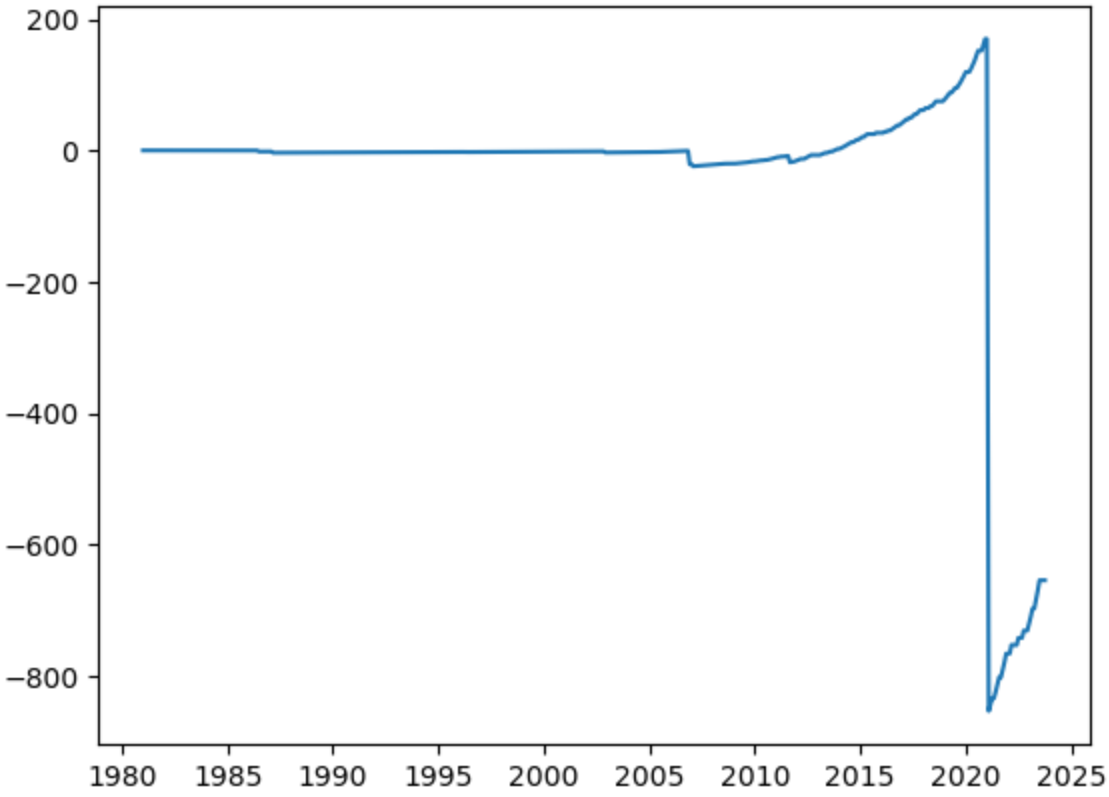
Out[27]:

Date	Adj Close	Diff	Target	Premium	Predictions	Profits
1980-12-31	0.117887	-0.020296	-1.0	0.009431	-1.0	0.000000
1981-01-31	0.097592	-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
1981-05-31	0.114432	-0.024614	-1.0	0.009155	-1.0	0.000000
1981-06-30	0.089818	-0.003454	-1.0	0.007185	-1.0	0.000000
1981-07-31	0.086364	-0.016841	-1.0	0.006909	-1.0	0.000000
1981-08-31	0.069523	-0.016842	-1.0	0.005562	-1.0	0.000000
1981-09-30	0.052682	0.016410	1.0	0.004215	1.0	0.004215
1981-10-31	0.069092	-0.004751	-1.0	0.005527	-1.0	0.000000
1981-11-30	0.064341	0.012091	1.0	0.005147	1.0	0.005147
1981-12-31	0.076432	-0.006045	-1.0	0.006115	-1.0	0.000000
1982-01-31	0.070387	-0.007341	-1.0	0.005631	-1.0	0.000000
1982-02-28	0.063046	-0.004750	-1.0	0.005044	-1.0	0.000000
1982-03-31	0.058296	-0.007341	-1.0	0.004664	-1.0	0.000000
1982-04-30	0.050955	-0.002591	-1.0	0.004076	-1.0	0.000000
1982-05-31	0.048364	-0.004318	-1.0	0.003869	-1.0	0.000000
1982-06-30	0.044046	0.002591	1.0	0.003524	1.0	0.003524
1982-07-31	0.046637	0.015545	1.0	0.003731	1.0	0.003731

5.) Plot profits over time

In [29]:
```
plt.plot(np.cumsum(df['Profits']))
```

Out[29]:
```
[<matplotlib.lines.Line2D at 0x1eca5de34d0>]
```



5.5

The primary skills I have learned from the MQE, which could help in Mr.Liu's ventures are forecasting, incentives and optimal decision making, for users, and statistical modeling.

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 [ ]: