ReadMe

The Title:
BUILD A TRADING BOT TO BUY AND SELL STOCKS TO OUTPERFORM THE S&P 500

The Goal:
To program a supervised machine learning trading bot, using algorithmic trading code fromt, to outperform the S&P 500 human traders

The Resources:
In order to run this stock machine learning trading model, we used the following machine learning modules as resources for code: Module 14, Module 10 and Import S&P 500 Dataset
We also used Google Search, scikit-learn, sklearn.svm.SVC, Google Colab and Jupyter Lab, matplot lib

The Approach:
The approach we took was to write a program that uses supervised machine learning to analyze the profitability of Apple stock price (labeled AAPL) from January 1, 2021 to April 7, 2022 and calculate its return of investment, then compare it to the return of investment of the S&P 500 stock price for the same time period and compare the results to see which performed better in terms of optimized profitability.

The Methods:
The first method of supervised learning used was to come up with buy, hold or sell signals. The project team decided to set a buy signal if the current day's close price was less than the previous day's close price. The sell signal would be triggered if the current day's close price was 5% higher than the previous day's close price. If none of these conditions were met the trade signal would be to hold.

We also ran a different version of the model where we would buy if today's close price is 2% lower than the previous day, sell if today's close price is 2% above the previous day and hold if neither of these conditions were met.

```
import hvplot.pandas
AAPL_df.hvplot.line()
AAPL_df["trade_type"]=np.nan
previous_price=0

for index,row in AAPL_df.iterrows():
    if previous_price==0:
        AAPL_df.loc[index,"trade_type"]="buy"

    elif row["Close"] < previous_price:
            AAPL_df.loc[index,"trade_type"]="buy"


    elif row["Close"] > (1.005*previous_price):
            AAPL_df.loc[index,"trade_type"]="sell"
    else:
         AAPL_df.loc[index, "trade_type"] = "hold"

previous_price=row["Close"]

if index == AAPL_df.index[-1]:
    AAPL_df.loc[index, "trade_type"] = "sell"

AAPL_df.head(15)
```

```
for index, row in AAPL_df.iterrows():

    # buy if the previous_price is 0, in other words, buy on the first day
    if previous_price == 0:
        AAPL_df.loc[index, "trade_type"] = "buy"

        # calculate the cost of the trade by multiplying the current day's price
        # by the share_size, or number of shares purchased
        AAPL_df.loc[index, "cost/proceeds"] = -(row["Close"] * share_size)

        # add the number of shares purchased to the accumulated shares
        accumulated_shares += share_size

    # buy if the current day's price is less than the previous day's price
    elif row["Close"] < previous_price:
        AAPL_df.loc[index, "trade_type"] = "buy"

        # calculate the cost of the trade by multiplying the current day's price
        # by the share_size, or number of shares purchased
        AAPL_df.loc[index, "cost/proceeds"] = -(row["Close"] * share_size)

        # add the number of shares purchased to the accumulated shares
        accumulated_shares += share_size
    elif row["Close"] > (1.005*previous_price):
        AAPL_df.loc[index, "trade_type"] = "sell"
```

```
[25] exit = signals_df[signals_df['Entry/Exit'] == -1.0]['Close'].hvplot.scatter(
         color='yellow',
         marker='v',
         size=200,
         legend=False,
         ylabel='Price in $',
         width=1000,
         height=400
     )
     entry = signals_df[signals_df['Entry/Exit'] == 1.0]['Close'].hvplot.scatter(
         color='purple',
         marker='^',
         size=200,
         legend=False,
         ylabel='Price in $',
         width=1000,
         height=400
     )
```

```
[25] # Visualize close price for the investment
     security_close = signals_df[['Close']].hvplot(
         line_color='lightgray',
         ylabel='Price in $',
         width=1000,
         height=400
     )

     # Visualize moving averages
     moving_avgs = signals_df[['SMA50', 'SMA100']].hvplot(
         ylabel='Price in $',
         width=1000,
         height=400
     )

     # Create the overlay plot
     entry_exit_plot = security_close * moving_avgs * entry * exit

     # Show the plot
     entry_exit_plot.opts(
         title="Apple - SMA50, SMA100, Entry and Exit Points"
     )
```

Instructions: Put a screen shot of Code showing trading signals.

Instructions: Put screen shots of dataframes and charts with detailed explanations.

The results were the expected return on investment for the supervised machine learning model, where we gave instructions to our trading bot to either buy or sell based on the stock price increasing or decreasing by 2% from the mean AAPL stock price for the time period of January 1, 2021 to April 7, 2022.

The expected return was 1.51%. This is an annualized return of 552%.

Instructions: More screen shots of code with explanation.

Challenges:

Next steps: Further development of this project would be to test different variations of the various trading signal variables. One example would be to change the buy/sell threshold. Another example would be to set the buy/sell signal based on a popular metric called the Relative Strength Index (RSI).

Additionally we ran the code on Google Colab to demonstrate that the trading bot that we built runs on the cloud. We ran the full code of the Supervised trading bot on Google Colab. This can be useful if we wanted to deploy the code for commercial purposes.

Based on the current results our bot trading AAPL did not outperform the S&P 500. Further testing could be to test an S&P 500 buy and hold strategy versus the active trading bot we built.

Conclusion:
Perhaps from this ROI comparison Apple is not a good stock candidate to compare with the S&P 500 given the time range parameter that we chose, since January 1, 2021 to April 7, 2022 was a rather volatile time for equity stocks, namely Apple.

However, for next steps, we would like to use our machine learning trading bot to buy and sell S&P 500 stock since the ROI and profit index was so good.