Ananya Singh
Assignment #2

Notes for the grader:
I submitted this assignment in both a notebook and a python file. A lot of the screenshots I put here are from the notebook itself so you can refer to that and it can be easier to follow along with my logic. The .py file is much "cleaner" but more function driven and doesn't have executables that the notebook has. I hope this is not an issue for you since a lot of my data science experience is using jupyter notebooks.

1.
   1) Solution:

```
spy_df["True Label"] = np.where(spy_df["Return"] >= 0, '+', '-')
spy_df.head()[["Date", "Return", "True Label"]]
```

✓ [116] < 10 ms

5 rows × 3 columns

	Date	Return	True Label
0	2016-01-04	0.000000	+
1	2016-01-05	0.001691	+
2	2016-01-06	-0.012614	-
3	2016-01-07	-0.023991	-
4	2016-01-08	-0.010977	-

```
sbux_df["True Label"] = np.where(sbux_df["Return"] >= 0, '+', '-')
sbux_df.head()[["Date", "Return", "True Label"]]
```

✓ [117] < 10 ms

5 rows × 3 columns

	Date	Return	True Label
0	2016-01-04	0.000000	+
1	2016-01-05	0.006694	+
2	2016-01-06	-0.008867	-
3	2016-01-07	-0.024772	-
4	2016-01-08	-0.001058	-

   2) Solution:

```python
training_data_for_spy = spy_df[spy_df["Year"].isin([2016, 2017, 2018])]
size_of_training_data_for_spy = len(training_data_for_spy)
size_of_positive_days_for_spy = len(training_data_for_spy[training_data_for_spy["True Label"] == '+'])
probability_next_day_up_for_spy = size_of_positive_days_for_spy/size_of_training_data_for_spy
print(f"Default probability of up day (p*) for SPY is: {probability_next_day_up_for_spy*100:.2f}%")
```
[118] < 10 ms

```
Default probability of up day (p*) for SPY is: 55.44%
```

```python
training_data_for_sbux = sbux_df[sbux_df["Year"].isin([2016, 2017, 2018])]
size_of_training_data_for_sbux = len(training_data_for_sbux)
size_of_positive_days_for_sbux = len(training_data_for_sbux[training_data_for_sbux["True Label"] == '+'])
probability_next_day_up_for_sbux = size_of_positive_days_for_sbux/size_of_training_data_for_sbux
print(f"Default probability of up day (p*) for SBUX is: {probability_next_day_up_for_sbux*100:.2f}%")
```
[119] < 10 ms

```
Default probability of up day (p*) for SBUX is: 50.93%
```

3) Solution:

   Probability of up day after 1 down days for ticker SPY : 59.52%
   Probability of up day after 1 down days for ticker SBUX : 50.00%
   Probability of up day after 2 down days for ticker SPY : 59.56%
   Probability of up day after 2 down days for ticker SBUX : 49.19%
   Probability of up day after 3 down days for ticker SPY : 63.64%
   Probability of up day after 3 down days for ticker SBUX : 44.68%

4) Solution:

   Probability of up day after 1 up days for ticker SPY : 52.04%
   Probability of up day after 1 up days for ticker SBUX : 51.70%
   Probability of up day after 2 up days for ticker SPY : 50.23%
   Probability of up day after 2 up days for ticker SBUX : 55.33%
   Probability of up day after 3 up days for ticker SPY : 46.79%
   Probability of up day after 3 up days for ticker SBUX : 53.70%

2.

1) I have added 3 extra columns for each of my tickers df. Each column is based on the window (W) provided. I am only going to show you the first 15 rows for each ticker in the screenshots below but if you want to see more please remove .head() from the df when I am iterating the outputs manually.

   Solution:

```python
print("\nSample predictions for SPY (2019):")
print(spy_df[spy_df['Year'] == 2019][['Date', 'Predicted_Label_W2', 'Predicted_Label_W3', 'Predicted_Label_W4']].head(15))
```
✓ [135] 21ms

```
Sample predictions for SPY (2019):
           Date Predicted_Label_W2 Predicted_Label_W3 Predicted_Label_W4
754  2019-01-02                NaN                NaN                NaN
755  2019-01-03                NaN                NaN                NaN
756  2019-01-04                  +                NaN                NaN
757  2019-01-07                  +                  +                NaN
758  2019-01-08                  +                  +                  +
759  2019-01-09                  +                  -                  -
760  2019-01-10                  +                  -                  +
761  2019-01-11                  +                  -                  +
762  2019-01-14                  +                  -                  +
763  2019-01-15                  +                  +                  +
764  2019-01-16                  +                  +                  -
765  2019-01-17                  +                  +                  +
766  2019-01-18                  +                  -                  -
767  2019-01-22                  +                  -                  +
768  2019-01-23                  +                  +                  +
```

```python
print("\nSample predictions for SPY (2020):")
print(spy_df[spy_df['Year'] == 2020][['Date', 'Predicted_Label_W2', 'Predicted_Label_W3', 'Predicted_Label_W4']].head(15))
```
✓ [136] 24ms

```
Sample predictions for SPY (2020):
            Date Predicted_Label_W2 Predicted_Label_W3 Predicted_Label_W4
1006  2020-01-02                  +                  +                  +
1007  2020-01-03                  +                  +                  +
1008  2020-01-06                  +                  +                  +
1009  2020-01-07                  +                  +                  -
1010  2020-01-08                  +                  +                  +
1011  2020-01-09                  +                  +                  +
1012  2020-01-10                  +                  +                  +
1013  2020-01-13                  +                  +                  +
1014  2020-01-14                  +                  +                  -
1015  2020-01-15                  +                  +                  +
1016  2020-01-16                  +                  +                  +
1017  2020-01-17                  +                  +                  +
1018  2020-01-21                  +                  -                  -
1019  2020-01-22                  +                  +                  +
1020  2020-01-23                  +                  +                  -
```

```
Sample predictions for SBUX (2019):
           Date Predicted_Label_W2 Predicted_Label_W3 Predicted_Label_W4
754  2019-01-02                NaN                NaN                NaN
755  2019-01-03                NaN                NaN                NaN
756  2019-01-04                  -                NaN                NaN
757  2019-01-07                  -                  +                NaN
758  2019-01-08                  +                  +                  +
759  2019-01-09                  +                  +                  +
760  2019-01-10                  +                  +                  -
761  2019-01-11                  +                  +                  -
762  2019-01-14                  +                  +                  -
763  2019-01-15                  -                  +                  +
764  2019-01-16                  -                  +                  -
765  2019-01-17                  +                  -                  -
766  2019-01-18                  -                  -                  -
767  2019-01-22                  +                  +                  +
768  2019-01-23                  +                  +                  +
```

```python
print("\nSample predictions for SBUX (2020):")
print(sbux_df[sbux_df['Year'] == 2020][['Date', 'Predicted_Label_W2', 'Predicted_Label_W3', 'Predicted_Label_W4']].head(15))
```
✓ [138] 20ms

```
Sample predictions for SBUX (2020):
            Date Predicted_Label_W2 Predicted_Label_W3 Predicted_Label_W4
1006  2020-01-02                  -                  -                  -
1007  2020-01-03                  +                  +                  +
1008  2020-01-06                  +                  +                  +
1009  2020-01-07                  -                  +                  +
1010  2020-01-08                  -                  -                  -
1011  2020-01-09                  -                  +                  +
1012  2020-01-10                  +                  +                  +
1013  2020-01-13                  +                  +                  +
1014  2020-01-14                  -                  -                  -
1015  2020-01-15                  +                  -                  +
1016  2020-01-16                  -                  -                  -
1017  2020-01-17                  +                  +                  +
1018  2020-01-21                  +                  +                  +
1019  2020-01-22                  +                  +                  -
1020  2020-01-23                  -                  -                  -
```

2) Solution:

SPY Results:

Results for W=2:
Correct predictions: 294
Total predictions: 502

Accuracy: 58.57%

Results for W=3:
Correct predictions: 293
Total predictions: 501
Accuracy: 58.48%

Results for W=4:
Correct predictions: 289
Total predictions: 500
Accuracy: 57.80%

SBUX Results:

Results for W=2:
Correct predictions: 248
Total predictions: 502
Accuracy: 49.40%

Results for W=3:
Correct predictions: 240
Total predictions: 501
Accuracy: 47.90%

Results for W=4:
Correct predictions: 254
Total predictions: 500
Accuracy: 50.80%

3) Solution (after analyzing above numbers):
Best Results:
SPY: W=2 (Accuracy: 58.57%)
SBUX: W=4 (Accuracy: 50.80%)

3.

1) Solutions:

[177] 33ms

Sample predictions for SPY:

11-20 40 rows × 4 columns

	Predicted_Label_W2	Predicted_Label_W3	Predicted_Label_W4	Ensemble_Label
764	+	+	-	+
765	+	+	+	+
766	+	-	-	-
767	+	-	+	+
768	+	+	+	+
769	+	+	-	+
770	+	+	+	+
771	+	-	-	-
772	+	+	+	+
773	+	+	+	+

```
print("Sample predictions for SBUX:")
sbux_df_with_ensemble[sbux_df_with_ensemble['Year'].isin([2019, 2020])][['Predicted_Label_W2', 'Predicted_Label_W3', 'Predicted_Label_W4', 'Ensemble_Label']].head(40)
```

[178] < 10 ms

Sample predictions for SBUX:

1-10 40 rows × 4 columns

	Predicted_Label_W2	Predicted_Label_W3	Predicted_Label_W4	Ensemble_Label
754	NaN	NaN	NaN	NaN
755	NaN	NaN	NaN	NaN
756	-	NaN	NaN	-
757	-	+	NaN	-
758	+	+	+	+
759	+	+	+	+
760	+	+	-	+
761	+	+	-	+
762	+	+	-	+
763	-	+	+	+

2) Solution:

SPY Ensemble Results:

Overall Accuracy: 58.80%

SBUX Ensemble Results:

Overall Accuracy: 48.40%

4. Solution:

```python
print('SPY statistics:')
spy_results.head()
```

✓ [253] 12ms

SPY statistics:

4 rows × 8 columns

	W	TP	FP	TN	FN	Accuracy	TPR	TNR
0	2	294	208	0	0	0.585657	1.000000	0.000000
1	3	245	160	48	48	0.584830	0.836177	0.230769
2	4	235	154	54	57	0.578000	0.804795	0.259615
3	ensemble	270	183	25	24	0.587649	0.918367	0.120192

```python
print('SBUX statistics:')
sbux_results.head()
```

✓ [255] < 10 ms

SBUX statistics:

4 rows × 8 columns

	W	TP	FP	TN	FN	Accuracy	TPR	TNR
0	2	150	126	98	128	0.494024	0.539568	0.437500
1	3	172	156	68	105	0.479042	0.620939	0.303571
2	4	143	113	111	133	0.508000	0.518116	0.495536
3	ensemble	174	156	68	104	0.482072	0.625899	0.303571

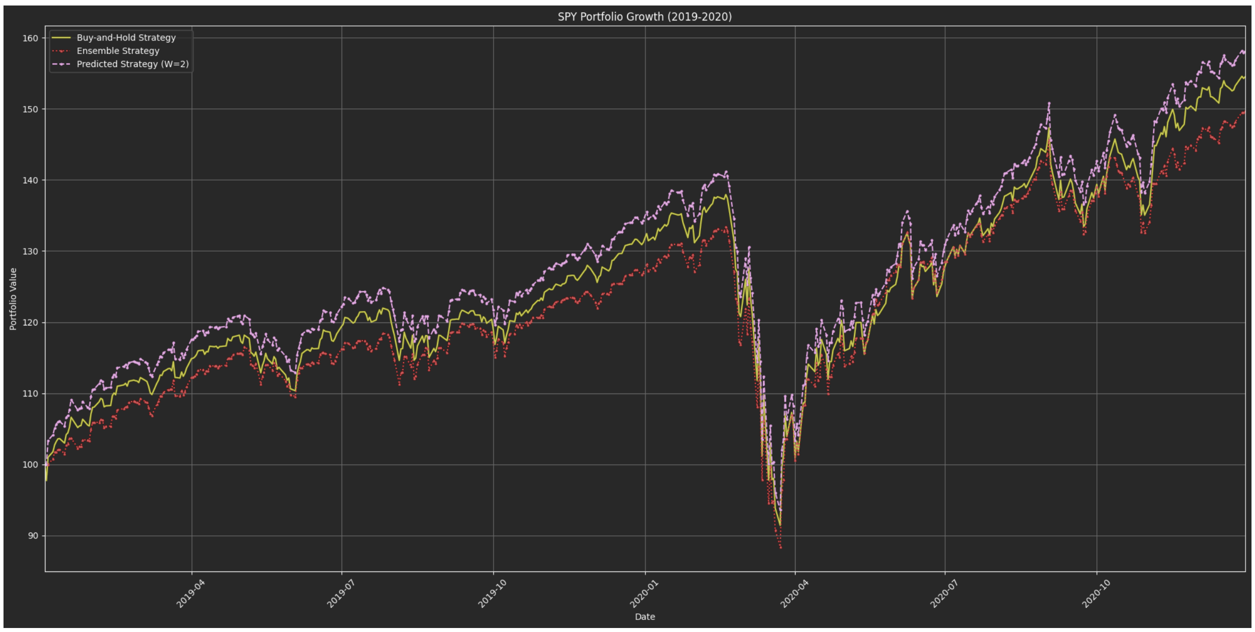
Findings:
SPY:

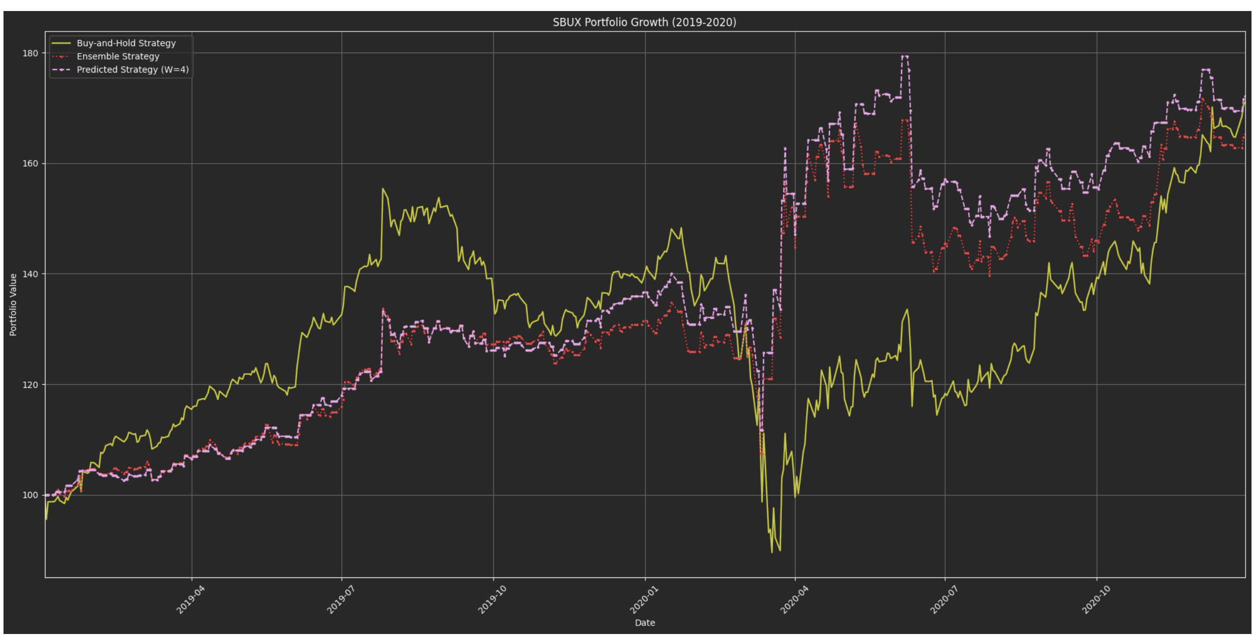
I think from what we can gather from the statistics table is that the Ensemble approach gives the best "accuracy" overall since it has the highest accuracy (58.7%) out of all 4 models given. Ensemble approach does however how a low TNR (0.12). For the window sizes, starting with 2. This was interesting because it suggests the model is biased towards predicting and labels. The accuracy is 58.6% because all + labels were correctly predicted. For window 3, TNR dropped to 0.84 but TNR increased to 0.23 and the overall accuracy. For window 4, the accuracy is around 57.8% and recognize that as the window length increases, the model gains ability to recognize the down labels.

SBUX:

I think from what we can gather from the statistics table is that the Windows size 4 provides the best accuracy whereas the ensemble approach is better for predicting + labels. With TPR of 0.63 and TNR of 0.30, ensemble approach shows that its very good at predicting + labels, the catch being that its overall accuracy is 48.2%. For the window sizes, starting with 2, TPR was 0.54 and TNR was 0.44 which indicates it leans slightly towards + predictions. For window 3, TPR increases to 0.62, while TNR drops to 0.30, suggesting that the model is now better at predicting + labels, at the expense of identifying - labels correctly and the accuracy is lower than W2 where it is at 47.9%. For window 4, this was the best model per say since TPR decreased to 0.52 and TNR increased to 0.50 where that showed more balance between + and - predictions. The accuracy is at its highest at 50.8%.

5.





Observations for SPY:

Buy-and-hold is the baseline here so we will be comparing W=2 and ensemble approach with it. W=2 closely follows Buy-and-hold but it has some deviation because it catches the uptrend but is not as quickly responsive. The ensemble approach also follows the buy-and-hold with close by ups and downs and it does deviate now and then but quickly returns to follow buy-and-hold. In 2019, we saw minor deviations but in 2020, it was more volatile. Both approaches deviated a bit more than what we saw in 2019.

Observations for SBUX:

Buy-and-hold is the baseline here so we will be comparing W=4 and ensemble approach with it. W=4 seems “flat” here which means that the model’s predictions are not really following buy-and-hold approach. The ensemble approach is extremely volatile (a lot more fluctuations) but it does seem to react similarly to buy-and-hold. Not as much possible growth though. In 2019, the W=4 strategy for SBUX remains flat, indicating less responsiveness to buy-and-hold approach, while the ensemble approach shows more fluctuations but somewhat mirrors Buy-and-Hold. In 2020, both strategies deviate further. W=4 doesn’t capture the recovery and the ensemble approach, lags behind Buy-and-Hold approach.