FINA 6333

Wenyi Wei, Shang Yang, Ze Zhang

Market Timing Report

Introduction

Louise, a family friend, believes that her strategy can produce return better than the market return. To prove her wrong, this report includes both conceptual explanation and data-driven explanation. Conceptual explanation includes efficient market hypothesis, the change in risk free rate, and why no strategy can outperform the market in the long term. Data-driven explanation includes the money growth, the volatilities and the Sharpe ratios of the two strategies.

Conceptual Explanation

#1 According to the efficient market hypothesis, we cannot use past performance to predict future price movements. Louise's strategy focuses on catching the momentum of the price uptrend and gaining the short-term profit. According to the hypothesis, stocks prices are trading at fair value, making it impossible to outperform overall market by stock selection, or in Louise's case, market timing method.

#2 Risk-free rate became really low in the 2000s. Louise's strategy may stay in the bond market for a long period of time if the stock market did not have two consecutive positive return when stock return was superior to bond return in the 2000s.

#3 Set aside EMH, even if Louise's strategy can provide decent returns in the short-term, there is no guarantee that it will provide long-term profits. Market transparency means that successful trading strategies will be duplicated by more investors, causing the profitable opportunity to disappear very soon. There is no way to use the same strategy and maintain a high rate of return in the long-term.

Data-driven Explanation

In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
gspc = pd.read_csv('GSPC.csv', index_col='Date', parse_dates=True).resample
vfinx = pd.read_csv('VFINX.csv', index_col='Date', parse_dates=True).resamp

ff = pd.read_csv(

    'F-F_Research_Data_Factors.csv',

    index_col=0,

    skiprows=3,

    nrows=12*(2020 - 1927 + 1) + 6 + 1

)

ff.index = pd.to_datetime(ff.index, format='%Y%m') + pd.offsets.MonthEnd(0)
```

In [3]:
```
# Combine the SP500 and the risk-free return into the same table
df = gspc.join(ff, how='inner')
```

In [4]:
```
df = df[['Adj Close', 'RF']]
```

In [5]:
```
# Table of SP500 adjusted close price and the risk-free rate.
df
```

Out[5]:

	Adj Close	RF
1980-01-31	114.16	0.80
1980-02-29	113.66	0.89
1980-03-31	102.09	1.21
1980-04-30	106.29	1.26
1980-05-31	111.24	0.81
...	...	...
2020-09-30	3363.00	0.01
2020-10-31	3269.96	0.01
2020-11-30	3621.63	0.01
2020-12-31	3756.07	0.01
2021-01-31	3714.24	0.00

493 rows × 2 columns

In [6]:
```
# Create a return column and calculate monthly return of SP500
df['return'] = df['Adj Close']/df['Adj Close'].shift(1) - 1
```

In [7]:
```
# Create a position column, 1 means holding SP500, 0 means holding risk-fre
df.loc[(df['return'] > 0) & (df['return'].shift(1) > 0), 'pos'] = 1
df.loc[(df['return'] < 0) & (df['return'].shift(1) < 0), 'pos'] = 0
```

In [8]:
```
# This table adds the return of SP500 and the postion of each month.
df
```

Out[8]:

	Adj Close	RF	return	pos
1980-01-31	114.16	0.80	NaN	NaN
1980-02-29	113.66	0.89	-0.004380	NaN
1980-03-31	102.09	1.21	-0.101795	0.0
1980-04-30	106.29	1.26	0.041140	NaN
1980-05-31	111.24	0.81	0.046571	1.0
...	...	...	...	...
2020-09-30	3363.00	0.01	-0.039228	NaN
2020-10-31	3269.96	0.01	-0.027666	0.0
2020-11-30	3621.63	0.01	0.107546	NaN
2020-12-31	3756.07	0.01	0.037121	1.0
2021-01-31	3714.24	0.00	-0.011137	NaN

493 rows × 4 columns

In [9]:
```
# Set the first month's position to 1, and fill the position with previous
df.iloc[0,3] = 1
df['pos'] = df['pos'].fillna(method='ffill')
df
```

Out[9]:

	Adj Close	RF	return	pos
1980-01-31	114.16	0.80	NaN	1.0
1980-02-29	113.66	0.89	-0.004380	1.0
1980-03-31	102.09	1.21	-0.101795	0.0
1980-04-30	106.29	1.26	0.041140	0.0
1980-05-31	111.24	0.81	0.046571	1.0
...	...	...	...	...
2020-09-30	3363.00	0.01	-0.039228	1.0
2020-10-31	3269.96	0.01	-0.027666	0.0
2020-11-30	3621.63	0.01	0.107546	0.0
2020-12-31	3756.07	0.01	0.037121	1.0
2021-01-31	3714.24	0.00	-0.011137	1.0

493 rows × 4 columns

In [10]:
```
# Because position changes after two consecutive returns occur, so we need
df['pos'] = df['pos'].shift(1)
df
```

Out[10]:

	Adj Close	RF	return	pos
1980-01-31	114.16	0.80	NaN	NaN
1980-02-29	113.66	0.89	-0.004380	1.0
1980-03-31	102.09	1.21	-0.101795	1.0
1980-04-30	106.29	1.26	0.041140	0.0
1980-05-31	111.24	0.81	0.046571	0.0
...	...	...	...	...
2020-09-30	3363.00	0.01	-0.039228	1.0
2020-10-31	3269.96	0.01	-0.027666	1.0
2020-11-30	3621.63	0.01	0.107546	0.0
2020-12-31	3756.07	0.01	0.037121	0.0
2021-01-31	3714.24	0.00	-0.011137	1.0

493 rows × 4 columns

In [11]:
```
# This table includes the ETF and risk-free rate.
df1 = vfinx.join(ff, how='inner')
```

In [12]:
```
df1 = df1[['Adj Close','RF']]
```

In [13]:
```
df1
```

Out[13]:

	Adj Close	RF
1980-01-31	5.550337	0.80
1980-02-29	5.571810	0.89
1980-03-31	5.026296	1.21
1980-04-30	5.243568	1.26
1980-05-31	5.533269	0.81
...	...	...
2020-09-30	309.145447	0.01
2020-10-31	300.887115	0.01
2020-11-30	333.791046	0.01
2020-12-31	346.600006	0.01
2021-01-31	343.059998	0.00

493 rows × 2 columns

In [14]:
```
df1['pos'] = df['pos']
df1['RF'] = df1['RF']/100
```

In [15]:
```
# Set the first month's position to 1
df1.iloc[0,2] = 1
```

In [16]:
```
# Calculate monthly return
df1['return'] = df1['Adj Close']/df1['Adj Close'].shift(1) - 1
```

In [17]:
```
df1['Naive'] = np.nan
```

In [18]:
```
# Find out each column's loc for later use.
df1['investment'] = 500
iloc_naive = df1.columns.get_loc('Naive')
iloc_Return = df1.columns.get_loc('return')
iloc_Invest = df1.columns.get_loc('investment')
iloc_RF = df1.columns.get_loc('RF')
```

In [19]:
```
df1
```

Out[19]:

	Adj Close	RF	pos	return	Naive	investment
1980-01-31	5.550337	0.0080	1.0	NaN	NaN	500
1980-02-29	5.571810	0.0089	1.0	0.003869	NaN	500
1980-03-31	5.026296	0.0121	1.0	-0.097906	NaN	500
1980-04-30	5.243568	0.0126	0.0	0.043227	NaN	500
1980-05-31	5.533269	0.0081	0.0	0.055249	NaN	500
...	...	...	...	...	...	...
2020-09-30	309.145447	0.0001	1.0	-0.040059	NaN	500
2020-10-31	300.887115	0.0001	1.0	-0.026713	NaN	500
2020-11-30	333.791046	0.0001	0.0	0.109356	NaN	500
2020-12-31	346.600006	0.0001	0.0	0.038374	NaN	500
2021-01-31	343.059998	0.0000	1.0	-0.010214	NaN	500

493 rows × 6 columns

In [20]:
```
# Calculate each month's ending balance of the naive strategy
df1.iloc[0,iloc_naive] = 500
for i in range(1, len(df1)):
    df1.iloc[i, iloc_naive] = (df1.iloc[i-1, iloc_naive] + df1.iloc[i, iloc
```

In [21]:
```
df1.round(2)
```

Out[21]:

	Adj Close	RF	pos	return	Naive	investment
1980-01-31	5.55	0.01	1.0	NaN	500.00	500
1980-02-29	5.57	0.01	1.0	0.00	1003.87	500
1980-03-31	5.03	0.01	1.0	-0.10	1356.63	500
1980-04-30	5.24	0.01	0.0	0.04	1936.89	500
1980-05-31	5.53	0.01	0.0	0.06	2571.52	500
...	...	...	...	...	...	...
2020-09-30	309.15	0.00	1.0	-0.04	3051644.42	500
2020-10-31	300.89	0.00	1.0	-0.03	2970611.21	500
2020-11-30	333.79	0.00	0.0	0.11	3296021.23	500
2020-12-31	346.60	0.00	0.0	0.04	3423022.55	500
2021-01-31	343.06	0.00	1.0	-0.01	3388556.31	500

493 rows × 6 columns

In [22]:
```
# Create a new column "new return", this is the monthly return of Louise's
df1.loc[df['pos'] == 1, 'New Return'] = df1['return']
df1.loc[df['pos'] == 0, 'New Return'] = df1['RF']
```

In [23]:
```
df1
```

Out[23]:

	Adj Close	RF	pos	return	Naive	investment	New Return
1980-01-31	5.550337	0.0080	1.0	NaN	5.000000e+02	500	NaN
1980-02-29	5.571810	0.0089	1.0	0.003869	1.003869e+03	500	0.003869
1980-03-31	5.026296	0.0121	1.0	-0.097906	1.356631e+03	500	-0.097906
1980-04-30	5.243568	0.0126	0.0	0.043227	1.936888e+03	500	0.012600
1980-05-31	5.533269	0.0081	0.0	0.055249	2.571523e+03	500	0.008100
...	...	...	...	...	...	...	...
2020-09-30	309.145447	0.0001	1.0	-0.040059	3.051644e+06	500	-0.040059
2020-10-31	300.887115	0.0001	1.0	-0.026713	2.970611e+06	500	-0.026713
2020-11-30	333.791046	0.0001	0.0	0.109356	3.296021e+06	500	0.000100
2020-12-31	346.600006	0.0001	0.0	0.038374	3.423023e+06	500	0.000100
2021-01-31	343.059998	0.0000	1.0	-0.010214	3.388556e+06	500	-0.010214

493 rows × 7 columns

In [24]:
```
df1['Strategy'] = np.nan
iloc_new = df1.columns.get_loc('New Return')
iloc_str = df1.columns.get_loc('Strategy')
```

In [25]:
```
# Calculate each month's ending balance of Louise's strategy
df1.iloc[0,-1] = 500
for i in range(1, len(df)):
    df1.iloc[i, iloc_str] = (df1.iloc[i-1, iloc_str] + df1.iloc[i, iloc_Inv
```

In [26]:
```
df1.round(2)
```

Out[26]:

	Adj Close	RF	pos	return	Naive	investment	New Return	Strategy
1980-01-31	5.55	0.01	1.0	NaN	500.00	500	NaN	500.00
1980-02-29	5.57	0.01	1.0	0.00	1003.87	500	0.00	1003.87
1980-03-31	5.03	0.01	1.0	-0.10	1356.63	500	-0.10	1356.63
1980-04-30	5.24	0.01	0.0	0.04	1936.89	500	0.01	1880.02
1980-05-31	5.53	0.01	0.0	0.06	2571.52	500	0.01	2399.30
...	...	...	...	...	...	...	...	...
2020-09-30	309.15	0.00	1.0	-0.04	3051644.42	500	-0.04	2123549.46
2020-10-31	300.89	0.00	1.0	-0.03	2970611.21	500	-0.03	2067308.84
2020-11-30	333.79	0.00	0.0	0.11	3296021.23	500	0.00	2068015.62
2020-12-31	346.60	0.00	0.0	0.04	3423022.55	500	0.00	2068722.47
2021-01-31	343.06	0.00	1.0	-0.01	3388556.31	500	-0.01	2048088.41

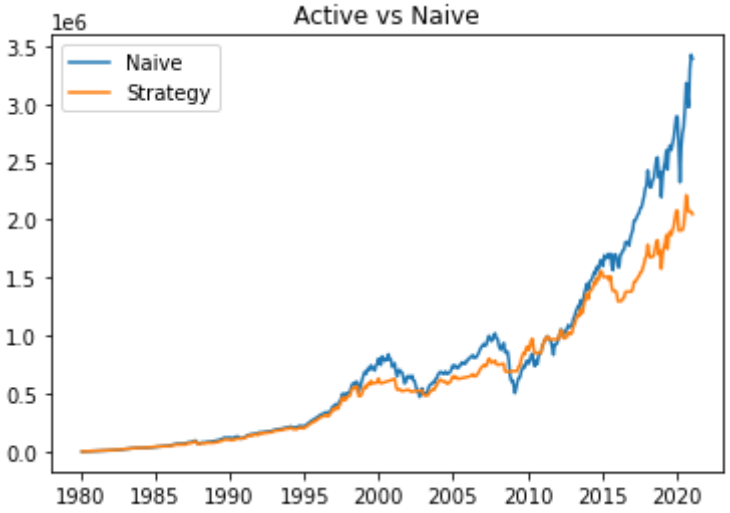
493 rows × 8 columns

In [27]:
```
# Calculate each strategy's rolling 12 months' standard deviation
df1['Naive std'] = df1['return'].rolling(window = 12).std()
df1['Strategy std'] = df1['New Return'].rolling(window = 12).std()
```

In [28]:
```
# Calculate each strategy's rolling 12 months' sharpe ratio
df1['Naive Excess Return'] = df1['return'] - df1['RF']
df1['Strategy Excess Return'] = df1['New Return'] - df1['RF']
df1['Naive sharpe'] = df1['Naive Excess Return'].rolling(window = 12).mean(
df1['Strategy sharpe'] = df1['Strategy Excess Return'].rolling(window = 12)
```

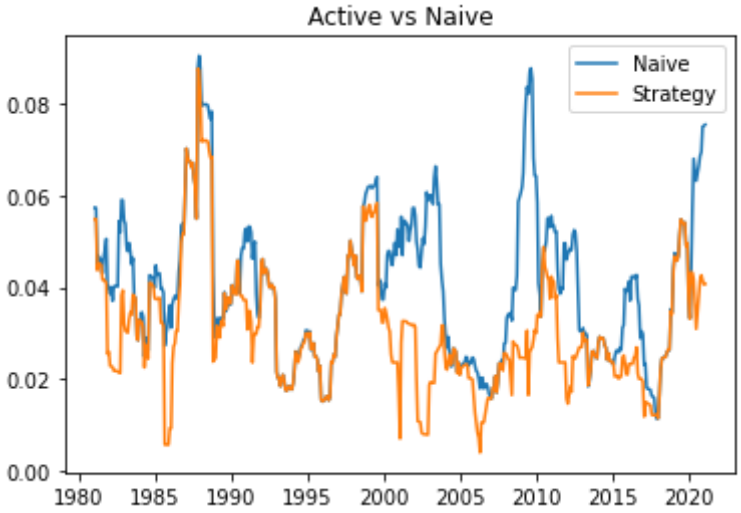
In [29]:
```
import matplotlib.pyplot as plt
```

In [30]:
```
# Plot each strategy's money growth
plt.plot(df1['Naive'],label = 'Naive')
plt.plot(df1['Strategy'],label = 'Strategy')
plt.title('Active vs Naive')
plt.legend();
```



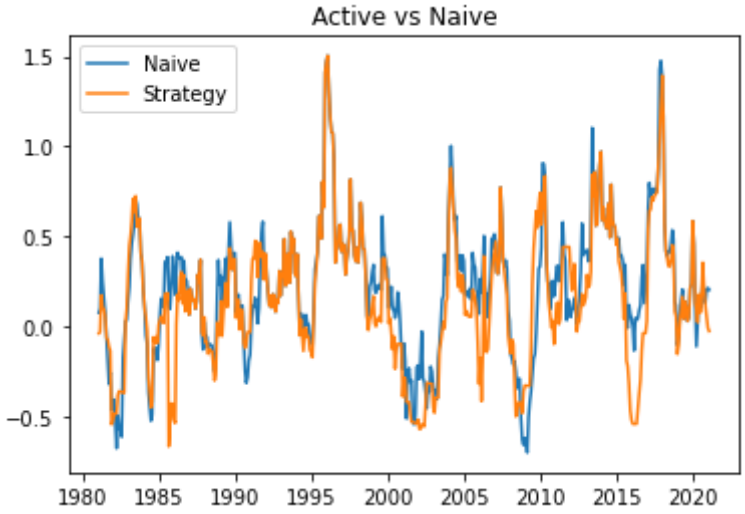
From the money growth we can see that the naive strategy outperformed Louise's strategy most of the time. Louise's strategy outperformed the naïve strategy only during 2008 financial crisis. The difference between two strategies becomes evident after 2015. From df1 table, we can find out that Louise's strategy earned much less compared to the naïve strategy.

In [31]:
```
# Plot each strategy's rolling 12 months' standard deviation
plt.plot(df1['Naive std'],label = 'Naive')
plt.plot(df1['Strategy std'],label = 'Strategy')
plt.title('Active vs Naive')
plt.legend();
```



From the standard deviation graph we can see that because Louise' strategy included bond, her strategy was less volatile.

In [32]:
```
# Plot each strategy's rolling 12 months' sharpe ratio
plt.plot(df1['Naive sharpe'],label = 'Naive')
plt.plot(df1['Strategy sharpe'],label = 'Strategy')
plt.title('Active vs Naive')
plt.legend();
```



From the graph we can see that naive strategy had a similar sharpe ratio with Louise's stratgy through out the years.

In [33]:
```
Naive_best = df1.nlargest(5,['return'])
Naive_best.reset_index(drop=True, inplace=True)
```

In [34]:
```
Louise_best = df1.nlargest(5,['New Return'])
Louise_best.reset_index(drop=True, inplace=True)
```

In [35]:
```
Best5 = pd.DataFrame()
Best5['Naive'] = Naive_best['return']
Best5['Louise'] = Louise_best['New Return']
Best5
```

Out[35]:

	Naive	Louise
0	0.132674	0.132674
1	0.128138	0.112472
2	0.124268	0.111968
3	0.112472	0.111328
4	0.111968	0.109645

Naive strategy's five best months were better than Louise's strategy's.

In [36]:
```
Naive_worst = df1.nsmallest(5,['return'])
Naive_worst.reset_index(drop=True, inplace=True)
```

In [37]:
```
Louise_worst = df1.nsmallest(5,['New Return'])
Louise_worst.reset_index(drop=True, inplace=True)
```

In [38]:
```
Worst5 = pd.DataFrame()
Worst5['Naive'] = Naive_worst['return']
Worst5['Louise'] = Louise_worst['New Return']
```

In [39]:
```
Worst5
```

Out[39]:

	Naive	Louise
0	-0.217270	-0.217270
1	-0.167924	-0.144736
2	-0.144736	-0.098788
3	-0.124013	-0.097906
4	-0.108880	-0.091377

Naive strategy's five worst months were worse than Louise's strategy's.

Based on data-driven analysis, we can conclude that naïve strategy outperformed Louise's strategy over the past 40 years. Even though Louise is now a millionaire, she could have earned more. Naïve strategy produced $1 million more than Louise's strategy did. On the other side, Louise's strategy had a lower volatility compared to Naïve strategy, so Louise did not take as much risk as the market portfolio did.

Other Trade-offs

#1 Transaction costs. The data-driven explanation did not consider transaction costs. Retail investors had to pay for transaction costs until recent. This means Louise should have earned even less compared to the naive strategy

#2 Risk and return. Louise tried to avoid big losses, as a result, she could also miss some big returns. If the SP500 had two small losses and a big return in the following month, Louise's strategy would have missed it.

#3 Monitoring. Apparently Louise's strategy requires monitoring monthly. Monitoring for five months may seem easy, but monitoring for forty years could be troublesome. On the other side, the naive strategy required no monitoring.

Conclusion

From both conceptual evidence and data-driven evidence, we can conclude that Louise's strategy did not outperform the market and she could have earned more money by using the naïve strategy. Louise's strategy did not take as much risk as SP500 ETF did, therefore, she cannot earn a superior return. There are many tradeoffs to consider, but overall passive investing performs better than active investing.