Group Project 1 - Project Group 2 (Hongjiang Li, Joanna Oghoore and Prasun Surana)

Introduction

Index funds are a type of mutual fund that aim to match the returns on the underlying index that it tracks. Vanguard Index Funds include both bonds and any components found in the underlying index. There are some risks that can be associated with owning index funds, even though they are generally considered safe and well-diversified investments. An example of risk is the one that the Vanguard funds can choose to track risky stocks compared to the ones that are tracking investment-grade bonds.

Investing in a Vanguard Fund gives investors exposure to returns from hundreds, or thousands, of companies who's shares makes up the fund. When some underperform, the fund is not drastically affected, as it enjoys the benefits of diversification, and some other constituents of the fund may perform well to offset this loss. Diversification is a key benefit of index funds, and appeal to many risk-averse investors who would prefer to take an approach of passive management.

One such investor is Louise Winthorpe, the subject of this project. Louise's strategy consists of starting with a 500 dollar initial investment, followed by further 500 dollar monthly investments. These investments do not remain in a single security, but instead, when Louise observes 2 consecutive months of positive returns in the S&P500, she moves her money to Vanguard 500 Index Fund, also known by its ticker VFINX. Conversely, when she observes 2 consecutive months of negative returns in the underlying index, she moves her investments into Treasury Bonds, which return the risk-free rate.

One conceptual explanation of the suboptimality of Louise's strategy is rooted in the Efficient Markets Hypothesis. The EMH posits that asset prices reflect all information about the asset, implying that it is impossible to beat the market consistently, since asset prices will respond immediately to new information. Hence, Louise's strategy, under EMH, is already inefficient, as she could instead just put her money in a Vanguard Fund, or an equivalent, that would theoretically generate returns at the highest rate.

Another reason that this strategy may be suboptimal is that there is a time-lag between the movement of the market and Louise's response. In her strategy, she will only move her money after 2 consecutive months of negative returns, and she moves it into a security that on average, pays far lower returns than the Vanguard Fund (Treasury Bonds are very safe investments since they are backed by the government). This means that by moving into Treasury Bonds, she has already lost money over 2 months, and is now making lower returns on the bonds. Conversely, she misses out on 2 months of positive returns before transferring her money back into the Vanguard Fund. Naturally, even if she kept her money in the Vanguard Fund

at all times, she would experience negative returns over some months. However, the difference in losses between this strategy and her original strategy may be sizeable.

This project will aim to use data-driven analysis to demonstrate why Louise's investment strategy can be improved. We begin with a few assumptions. Firstly, we assume that Louise starts her investment in 1985 with 500 dollars into the VFINX by default. Secondly, we assume that her switching between VFINX and Treasury Bonds has no additional cost. In real markets, there is a transaction cost which is incurred every time an investor buys or sells a security. For purpose of simplicity, and lack of access to information about commissions and other investment-related costs, we have ommitted these from the project's consideration, and will assume that her move between these two securities is frictionless. Thirdly, we assume that Louise is a risk-averse investor. This signifies that for any two portfolios that have the same return, Louise will choose the portfolio that minimizes risk, which is a common assumption for all rational investors. Further, it is even possible that Louise would choose to forgo higher expected returns as it carries more risk, and instead settle for a lower-risk portfolio.

The project will use data from the S&P500 index, VFINX returns and the Fama-French 3 factor dataset and examine metrics such as total portfolio values, rolling volatilities, rolling Sharpe Ratios and rolling Sortino ratios to illustrate why Louise's investment returns could be easily improved by simply adopting a more naïve strategy which does not involve constant switching between two securities.

Data Analysis

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
# Importing the Data

gspc = pd.read_csv('GSPC.csv', index_col='Date', parse_dates=True).resample('M')

vfinx = pd.read_csv('VFINX.csv', index_col='Date', parse_dates=True).resample('M

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]:
```
gspc['Return'] = gspc['Adj Close'].pct_change()
gspc
```

Out[3]:

Date	Open	High	Low	Close	Adj Close	Volume	Return
1980-01-31	0.00	117.17	113.78	114.16	114.16	65900000	NaN
1980-02-29	0.00	114.12	111.77	113.66	113.66	38810000	-0.004380
1980-03-31	0.00	102.65	100.02	102.09	102.09	35840000	-0.101795
1980-04-30	0.00	106.72	104.50	106.29	106.29	30850000	0.041140
1980-05-31	0.00	111.55	108.87	111.24	111.24	34820000	0.046571
...	...	...	...	...	...	...	...
2020-09-30	3341.21	3393.56	3340.47	3363.00	3363.00	4722530000	-0.039228
2020-10-31	3293.59	3304.93	3233.94	3269.96	3269.96	4840450000	-0.027666
2020-11-30	3634.18	3634.18	3594.39	3621.63	3621.63	6291400000	0.107546
2020-12-31	3733.27	3760.20	3726.88	3756.07	3756.07	3172510000	0.037121
2021-01-31	3778.05	3778.05	3694.12	3714.24	3714.24	6612570000	-0.011137

493 rows × 7 columns

In [4]:
```
vfinx['Return'] = vfinx['Adj Close'].pct_change()
vfinx
```

Out[4]:

Date	Open	High	Low	Close	Adj Close	Volume	Return
1980-01-31	15.510000	15.510000	15.510000	15.510000	5.550337	0	NaN
1980-02-29	15.570000	15.570000	15.570000	15.570000	5.571810	0	0.003869
1980-03-31	13.880000	13.880000	13.880000	13.880000	5.026296	0	-0.097906
1980-04-30	14.480000	14.480000	14.480000	14.480000	5.243568	0	0.043227
1980-05-31	15.280000	15.280000	15.280000	15.280000	5.533269	0	0.055249
...	...	...	...	...	...	...	...
2020-09-30	310.329987	310.329987	310.329987	310.329987	309.145447	0	-0.040059
2020-10-31	302.040009	302.040009	302.040009	302.040009	300.887115	0	-0.026713
2020-11-30	335.070007	335.070007	335.070007	335.070007	333.791046	0	0.109356
2020-12-31	346.600006	346.600006	346.600006	346.600006	346.600006	0	0.038374
2021-01-31	343.059998	343.059998	343.059998	343.059998	343.059998	0	-0.010214

493 rows × 7 columns

In [5]:
```
# Create a new DataFrame with all relevant returns since 1985.

lump = 500
regular = 500
ret = gspc.loc['1985-01-01':'2021-01-31', 'Return']
ret2 = vfinx.loc['1985-01-01':'2021-01-31', 'Return']
rf = ff.loc['1985-01-01':'2021-01-31', 'RF']

df = pd.DataFrame(
    {
        'Date': pd.date_range(start ='1985-01-01', end = '2021-01-31', freq='M')
        'Return_GSPC': ret, 'Return_VFINX': ret2, 'RF': rf}
)
df.set_index('Date', inplace=True)
df.head()
```

Out[5]:

Date	Return_GSPC	Return_VFINX	RF
1985-01-31	0.074085	0.075820	0.65
1985-02-28	0.008629	0.013810	0.58
1985-03-31	-0.002870	0.000053	0.62
1985-04-30	-0.004594	-0.002843	0.72
1985-05-31	0.054051	0.060333	0.66

In [6]:
```
# The column 'Value' is the portfolio values if Louise invested $500 at the star
# additional $500 monthly investments. The 'Portfolio' column will be the portfo
# which involves switching between VFINX and Treasury Bonds, INCLUDING $500 mont

df['Value'] = lump * df['Return_VFINX'].add(1).cumprod()
df['Portfolio'] = np.nan
df['Investment'] = regular
df
```

Out[6]:

Date	Return_GSPC	Return_VFINX	RF	Value	Portfolio	Investment
1985-01-31	0.074085	0.075820	0.65	537.909994	NaN	500
1985-02-28	0.008629	0.013810	0.58	545.338323	NaN	500
1985-03-31	-0.002870	0.000053	0.62	545.366962	NaN	500
1985-04-30	-0.004594	-0.002843	0.72	543.816704	NaN	500
1985-05-31	0.054051	0.060333	0.66	576.626654	NaN	500
...	...	...	...	...	...	...
2020-09-30	-0.039228	-0.040059	0.01	17393.933814	NaN	500
2020-10-31	-0.027666	-0.026713	0.01	16929.282363	NaN	500
2020-11-30	0.107546	0.109356	0.01	18780.607698	NaN	500

Date	Return_GSPC	Return_VFINX	RF	Value	Portfolio	Investment
2020-12-31	0.037121	0.038374	0.01	19501.298249	NaN	500
2021-01-31	-0.011137	-0.010214	0.00	19302.121242	NaN	500

433 rows × 6 columns

In [7]:
```
# VFINX_Value is the column which shows the portfolio values if Louise had inves
# ONLY into VFINX.

Value = np.array([lump*(1 + df['Return_VFINX'][0]) + df['Investment'][0]])
for i in range(1, len(df)):
    Value = np.append(Value, Value[i-1]*(1 + df['Return_VFINX'][i]) + df['Invest

df['VFINX_Value'] = Value
decimals = 2
df['VFINX_Value'] = df['VFINX_Value'].apply(lambda x: round(x, decimals))
df
```

Out[7]:

Date	Return_GSPC	Return_VFINX	RF	Value	Portfolio	Investment	VFINX_Value
1985-01-31	0.074085	0.075820	0.65	537.909994	NaN	500	1037.91
1985-02-28	0.008629	0.013810	0.58	545.338323	NaN	500	1552.24
1985-03-31	-0.002870	0.000053	0.62	545.366962	NaN	500	2052.32
1985-04-30	-0.004594	-0.002843	0.72	543.816704	NaN	500	2546.49
1985-05-31	0.054051	0.060333	0.66	576.626654	NaN	500	3200.13
...	...	...	...	...	...	...	...
2020-09-30	-0.039228	-0.040059	0.01	17393.933814	NaN	500	1701507.88
2020-10-31	-0.027666	-0.026713	0.01	16929.282363	NaN	500	1656554.79
2020-11-30	0.107546	0.109356	0.01	18780.607698	NaN	500	1838209.65
2020-12-31	0.037121	0.038374	0.01	19501.298249	NaN	500	1909249.45
2021-01-31	-0.011137	-0.010214	0.00	19302.121242	NaN	500	1890249.29

433 rows × 7 columns

In [8]:
```
iloc_Portfolio = df.columns.get_loc('Portfolio')
iloc_Return = df.columns.get_loc('Return_GSPC')
iloc_Return2 = df.columns.get_loc('Return_VFINX')
```

```
iloc_RF = df.columns.get_loc('RF')
iloc_Invest = df.columns.get_loc('Investment')
```

In [9]:

```
# Filling out the portfolio values in the 'Portfolio' column, which is Louise's

# Initializing the first value in 'Portfolio':
df.iloc[0, iloc_Portfolio] = lump * (1 + df.iloc[0, iloc_Return2]) + df.iloc[0,

# Here, we use the variable i to indicate the row indices in the DataFrame, and
# between VFINX and Treasury Bonds (RF).
i=1
t=1
while i < len(df):
    j = df.iloc[i-2]['Return_GSPC']
    k = df.iloc[i-1]['Return_GSPC']
    if ((t==1) and (j<0 and k <0)):
        df.iloc[i, iloc_Portfolio] = df.iloc[i-1, iloc_Portfolio]*(1 + df.iloc[i
        t = 0
    elif ((t==1) and (j>0 or k>0)):
        df.iloc[i, iloc_Portfolio] = df.iloc[i-1, iloc_Portfolio]*(1 + df.iloc[i
    elif ((t==0) and (j<0 or k<0)):
        df.iloc[i, iloc_Portfolio] = df.iloc[i-1, iloc_Portfolio]*(1 + df.iloc[i
    elif ((t==0) and (j>0 and k>0)):
        df.iloc[i, iloc_Portfolio] = df.iloc[i-1, iloc_Portfolio]*(1 + df.iloc[i
        t = 1
    df['Portfolio'] = df['Portfolio'].apply(lambda x: round(x, decimals))
    i += 1

df
```

Out[9]:

Date	Return_GSPC	Return_VFINX	RF	Value	Portfolio	Investment	VFINX_Value
1985-01-31	0.074085	0.075820	0.65	537.909994	1037.91	500	1037.91
1985-02-28	0.008629	0.013810	0.58	545.338323	1552.24	500	1552.24
1985-03-31	-0.002870	0.000053	0.62	545.366962	2052.32	500	2052.32
1985-04-30	-0.004594	-0.002843	0.72	543.816704	2546.49	500	2546.49
1985-05-31	0.054051	0.060333	0.66	576.626654	3063.30	500	3200.13
...	...	...	...	...	...	...	...
2020-09-30	-0.039228	-0.040059	0.01	17393.933814	1251361.42	500	1701507.88
2020-10-31	-0.027666	-0.026713	0.01	16929.282363	1218433.28	500	1656554.79
2020-11-30	0.107546	0.109356	0.01	18780.607698	1219055.12	500	1838209.65
2020-12-31	0.037121	0.038374	0.01	19501.298249	1219677.03	500	1909249.45

Date	Return_GSPC	Return_VFINX	RF	Value	Portfolio	Investment	VFINX_Value
2021-01-31	-0.011137	-0.010214	0.00	19302.121242	1207719.83	500	1890249.29

433 rows × 7 columns

In [10]:
```
# RF_Value shows the portfolio value if Louise left all her investments in Treas

df['Value'] = lump * df['RF'].add(1).cumprod()

Value = np.array([lump*(1 + 0.01*df['RF'][0]) + df['Investment'][0]])
for i in range(1, len(df)):
    Value = np.append(Value, Value[i-1]*(1 + 0.01*df['RF'][i]) + df['Investment'

df['RF_Value'] = Value
df.drop(columns = 'Value')
```

Out[10]:

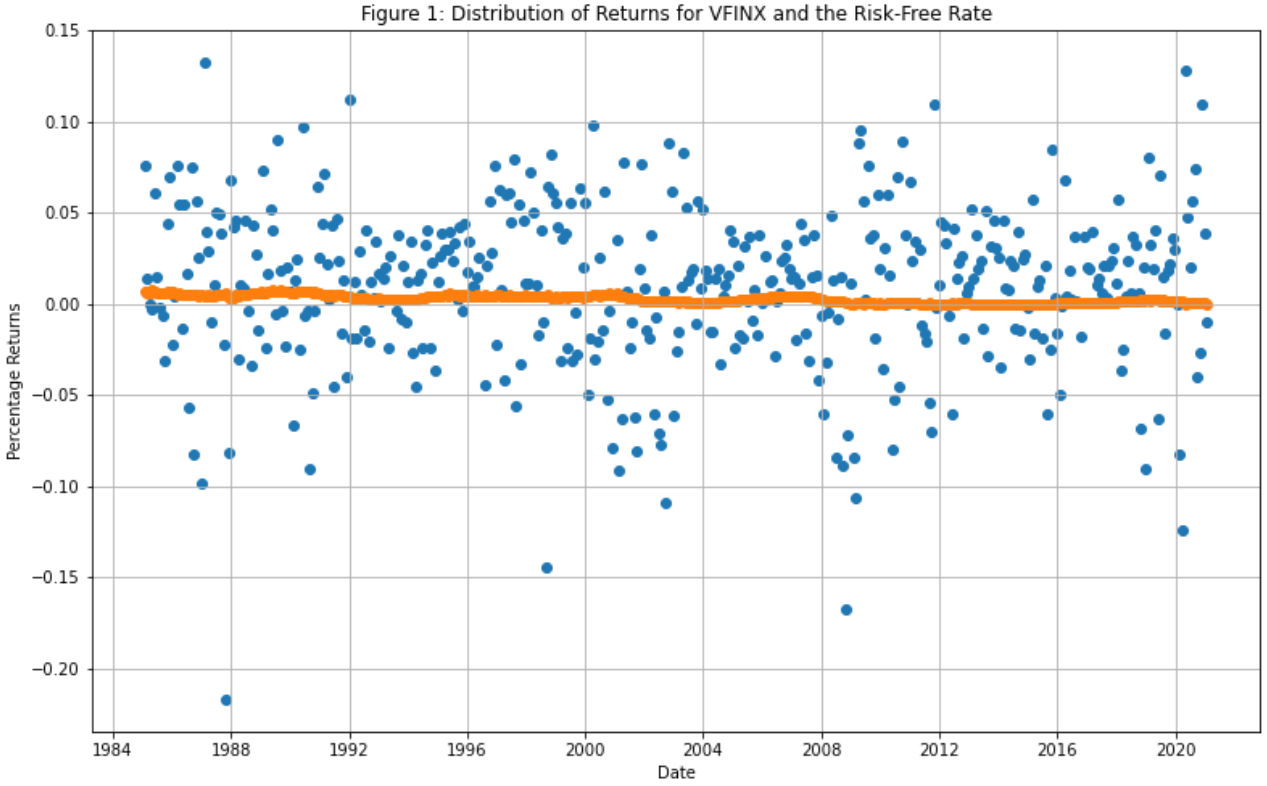
Date	Return_GSPC	Return_VFINX	RF	Portfolio	Investment	VFINX_Value	RF_Value
1985-01-31	0.074085	0.075820	0.65	1037.91	500	1037.91	1003.250000
1985-02-28	0.008629	0.013810	0.58	1552.24	500	1552.24	1509.068850
1985-03-31	-0.002870	0.000053	0.62	2052.32	500	2052.32	2018.425077
1985-04-30	-0.004594	-0.002843	0.72	2546.49	500	2546.49	2532.957737
1985-05-31	0.054051	0.060333	0.66	3063.30	500	3200.13	3049.675258
...	...	...	...	...	...	...	...
2020-09-30	-0.039228	-0.040059	0.01	1251361.42	500	1701507.88	325712.066376
2020-10-31	-0.027666	-0.026713	0.01	1218433.28	500	1656554.79	326244.637583
2020-11-30	0.107546	0.109356	0.01	1219055.12	500	1838209.65	326777.262046
2020-12-31	0.037121	0.038374	0.01	1219677.03	500	1909249.45	327309.939773
2021-01-31	-0.011137	-0.010214	0.00	1207719.83	500	1890249.29	327809.939773

433 rows × 7 columns

We can see from the DataFrame above that the final value of 'VFINX_Value' is much larger than that of 'RF_Value'. Below, we plot the distribution of returns for both VFINX and RF to illustrate

why this is the case.

In [11]:
```
plt.figure(figsize = (13,8))
plt.scatter(df.index, df['Return_VFINX'])
plt.scatter(x=df.index, y=(df['RF'])/100)
plt.title('Figure 1: Distribution of Returns for VFINX and the Risk-Free Rate')
plt.xlabel('Date')
plt.ylabel('Percentage Returns')
plt.grid()
plt.show()
```



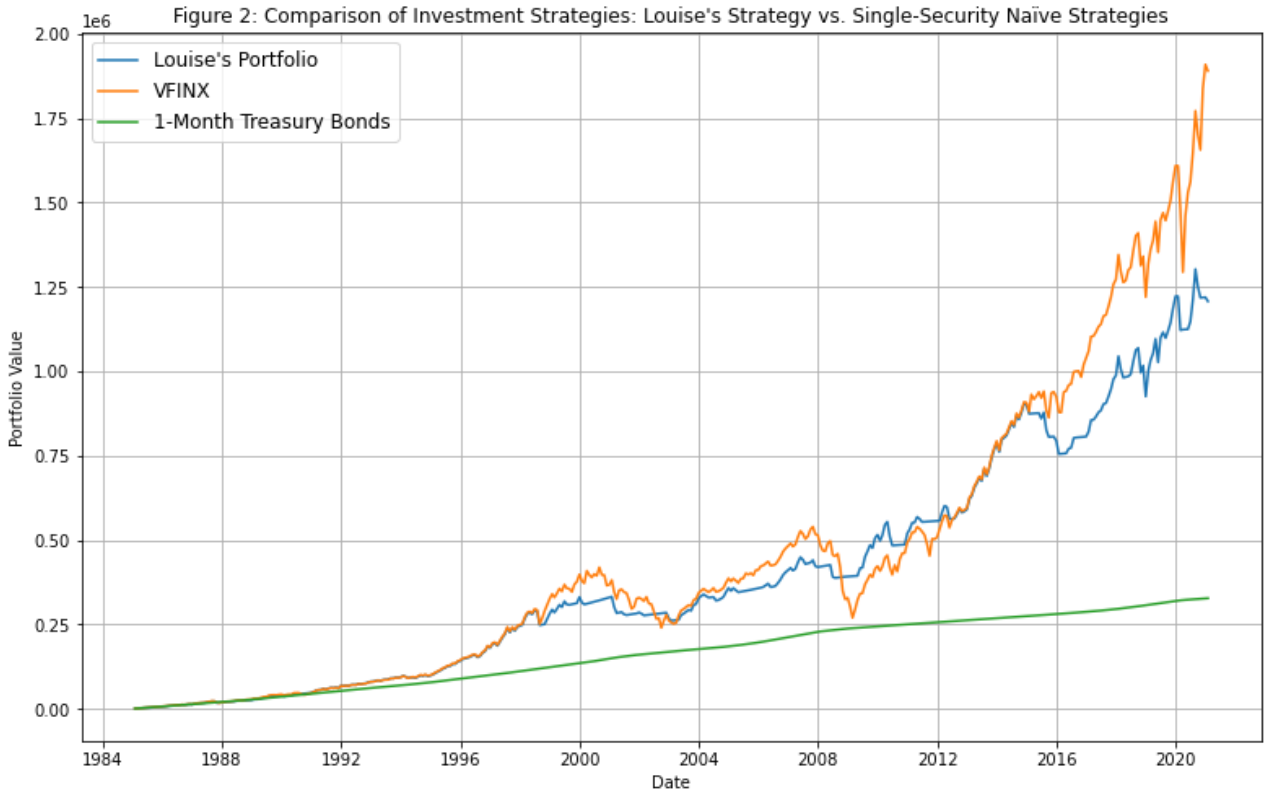
In [12]:
```
print(df['Return_VFINX'].mean())
print((df['RF']/100).mean())
```

```
0.009452265676859943
0.0026325635103926085
```

From the values above, we can see that the mean of the Treasury Bond returns since 1985 is lower than the mean of returns from VFINX since 1985. Since, on average, the risk-free rate is lower than VFINX returns, we can expect lower returns from a naïve strategy where we invest money only into Treasury Bonds over the 36 year period, than if we invested into either only the Vanguard Fund, or a mixture of the Vanguard Fund and Treasury Bonds in varying weights. This can be seen from the DataFrame above, where the final portfolio value for the a naïve strategy involving only VFINX returns approximately USD1.89m at the end of the investment period, and a naïve strategy involving only investments into Treasury Bonds produces approximately USD327,810. The growth of these two portfolios can be seen in Figure 2 below. We will use this idea to explore differently weighted portfolios shortly.

In [13]:
```
plt.figure(figsize = (13,8))
plt.plot(df.index, df['Portfolio'], label = "Louise's Portfolio")
plt.plot(df.index, df['VFINX_Value'], label = 'VFINX')
plt.plot(df.index, df['RF_Value'], label = '1-Month Treasury Bonds')
```

```
plt.title("Figure 2: Comparison of Investment Strategies: Louise's Strategy vs.
plt.xlabel('Date')
plt.ylabel('Portfolio Value')
plt.legend(fontsize = 12)
plt.grid()
plt.show()
```



In [14]:
```
# Final Value for Louise's Portfolio:
df.iloc[432, 4]
```

Out[14]: 1207719.83

In [15]:
```
# Final Value for Naive Portfolio of just VFINX:
df.iloc[432, 6]
```

Out[15]: 1890249.29

In [16]:
```
# How much more money would Louise have if she followed a naïve strategy of keep
df.iloc[432, 6] - df.iloc[432, 4]
```

Out[16]: 682529.46

Comparison of differently-weighted portfolios in VFINX and Treasury Bonds

In [17]:
```
df2 = pd.DataFrame(
    {
        'Date': pd.date_range(start ='1985-01-01', end = '2021-01-31', freq='M')
        'Return_VFINX': ret2, 'RF': rf}
)
df2.set_index('Date', inplace=True)
```

```
df2
```

Out[17]:

Date	Return_VFINX	RF
1985-01-31	0.075820	0.65
1985-02-28	0.013810	0.58
1985-03-31	0.000053	0.62
1985-04-30	-0.002843	0.72
1985-05-31	0.060333	0.66
...	...	...
2020-09-30	-0.040059	0.01
2020-10-31	-0.026713	0.01
2020-11-30	0.109356	0.01
2020-12-31	0.038374	0.01
2021-01-31	-0.010214	0.00

433 rows × 2 columns

We would like to see how portfolio returns are impacted if we invest varying proportions into the Vanguard Fund and Treasury Bonds. We will test this for portfolio weighted 90%, 80%, 70%, 60% and 50% in VFINX. 'w' denotes the weight in VFINX, and consequently the weight in Treasury Bonds would be '1-w' (not shown in the DataFrame below).

In [18]:
```
df2[['w1=0.9','w2=0.8','w3=0.7','w4=0.6','w5=0.5']] = np.nan
df2['Investment'] = 500
df2
```

Out[18]:

Date	Return_VFINX	RF	w1=0.9	w2=0.8	w3=0.7	w4=0.6	w5=0.5	Investment
1985-01-31	0.075820	0.65	NaN	NaN	NaN	NaN	NaN	500
1985-02-28	0.013810	0.58	NaN	NaN	NaN	NaN	NaN	500
1985-03-31	0.000053	0.62	NaN	NaN	NaN	NaN	NaN	500
1985-04-30	-0.002843	0.72	NaN	NaN	NaN	NaN	NaN	500
1985-05-31	0.060333	0.66	NaN	NaN	NaN	NaN	NaN	500
...	...	...	...	...	...	...	...	...
2020-09-30	-0.040059	0.01	NaN	NaN	NaN	NaN	NaN	500
2020-10-31	-0.026713	0.01	NaN	NaN	NaN	NaN	NaN	500
2020-11-30	0.109356	0.01	NaN	NaN	NaN	NaN	NaN	500
2020-12-31	0.038374	0.01	NaN	NaN	NaN	NaN	NaN	500
2021-01-31	-0.010214	0.00	NaN	NaN	NaN	NaN	NaN	500

433 rows × 8 columns

In [19]:
```
iloc_VFINX = df2.columns.get_loc('Return_VFINX')
iloc_RF = df2.columns.get_loc('RF')
iloc_w1 = df2.columns.get_loc('w1=0.9')
iloc_w2 = df2.columns.get_loc('w2=0.8')
iloc_w3 = df2.columns.get_loc('w3=0.7')
iloc_w4 = df2.columns.get_loc('w4=0.6')
iloc_w5 = df2.columns.get_loc('w5=0.5')
iloc_Invest = df2.columns.get_loc('Investment')

# Initializing the first portfolio values for each weight column:
df2.iloc[0, 2] = lump * (1+((df2.iloc[0,0]*0.9) + ((df2.iloc[0, 1]/100)*0.1)))
df2.iloc[0, 3] = lump * (1+((df2.iloc[0,0]*0.8) + ((df2.iloc[0, 1]/100)*0.2)))
df2.iloc[0, 4] = lump * (1+((df2.iloc[0,0]*0.7) + ((df2.iloc[0, 1]/100)*0.3)))
df2.iloc[0, 5] = lump * (1+((df2.iloc[0,0]*0.6) + ((df2.iloc[0, 1]/100)*0.4)))
df2.iloc[0, 6] = lump * (1+((df2.iloc[0,0]*0.5) + ((df2.iloc[0, 1]/100)*0.5)))

decimals = 2

i = 1
while i < len(df2):
    df2.iloc[i, iloc_w1] = df2.iloc[i-1, iloc_w1] * (1+((df2.iloc[i,0]*0.9) + ((
    df2['w1=0.9'] = df2['w1=0.9'].apply(lambda x: round(x, decimals))
    i += 1

i = 1
while i < len(df2):
    df2.iloc[i, iloc_w2] = df2.iloc[i-1, iloc_w2] * (1+((df2.iloc[i,0]*0.8) + ((
    df2['w2=0.8'] = df2['w2=0.8'].apply(lambda x: round(x, decimals))
    i += 1

i = 1
while i < len(df2):
    df2.iloc[i, iloc_w3] = df2.iloc[i-1, iloc_w3] * (1+((df2.iloc[i,0]*0.7) + ((
    df2['w3=0.7'] = df2['w3=0.7'].apply(lambda x: round(x, decimals))
    i += 1

i = 1
while i < len(df2):
    df2.iloc[i, iloc_w4] = df2.iloc[i-1, iloc_w4] * (1+((df2.iloc[i,0]*0.6) + ((
    df2['w4=0.6'] = df2['w4=0.6'].apply(lambda x: round(x, decimals))
    i += 1

i = 1
while i < len(df2):
    df2.iloc[i, iloc_w5] = df2.iloc[i-1,iloc_w5] * (1+((df2.iloc[i,0]*0.5) + ((d
    df2['w5=0.5'] = df2['w5=0.5'].apply(lambda x: round(x, decimals))
    i += 1

df2
```

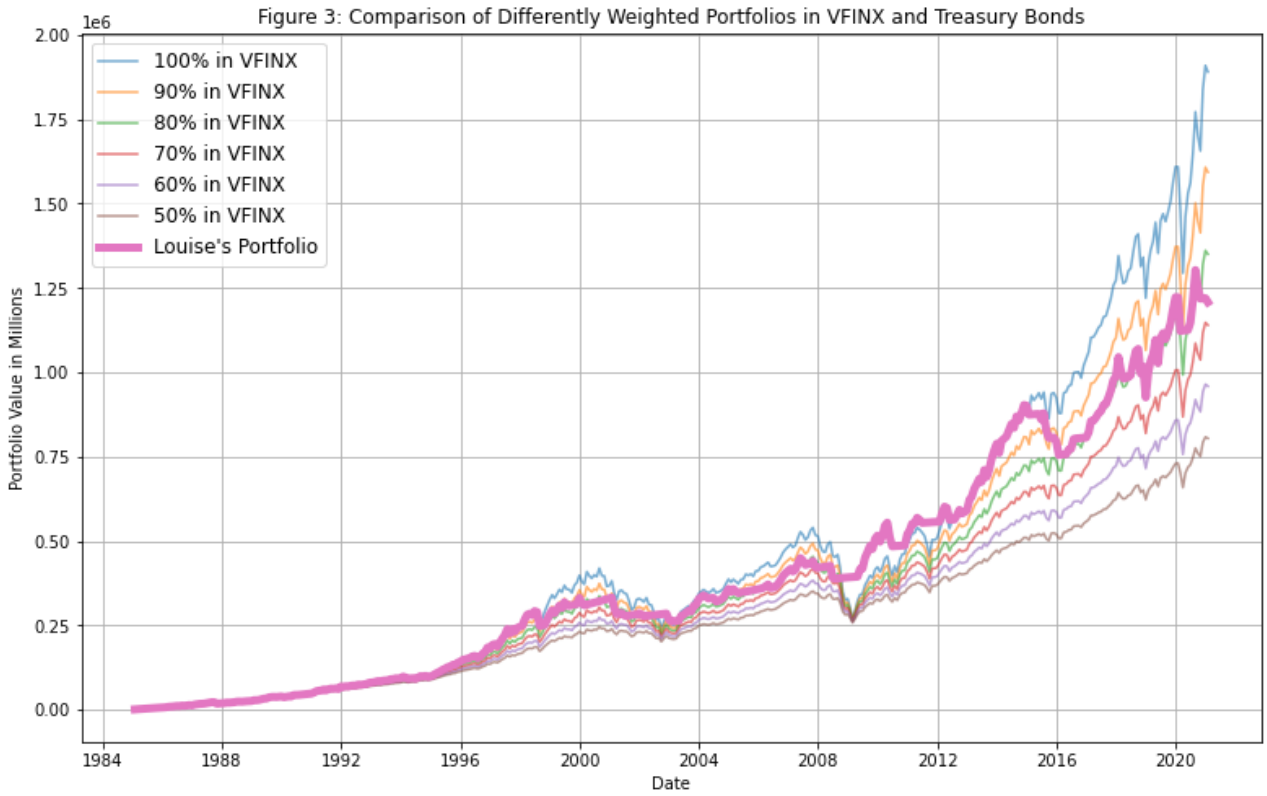
Out[19]:

Date	Return_VFINX	RF	w1=0.9	w2=0.8	w3=0.7	w4=0.6	w5=0.5	Investme
1985-01-31	0.075820	0.65	534.44	530.98	527.51	524.05	520.58	5(

Date	Return_VFINX	RF	w1=0.9	w2=0.8	w3=0.7	w4=0.6	w5=0.5	Investme
1985-02-28	0.013810	0.58	1041.40	1037.46	1033.53	1029.60	1025.68	5(
1985-03-31	0.000053	0.62	1542.09	1538.79	1535.49	1532.19	1528.89	5(
1985-04-30	-0.002843	0.72	2039.26	2037.51	2035.75	2033.99	2032.22	5(
1985-05-31	0.060333	0.66	2651.34	2638.54	2625.76	2612.99	2600.23	5(
...	...	...	...	...	...	...	...	
2020-09-30	-0.040059	0.01	1448056.84	1239053.61	1056267.07	897477.04	760396.93	5(
2020-10-31	-0.026713	0.01	1413757.03	1213098.90	1037047.20	883628.13	750778.55	5(
2020-11-30	0.109356	0.01	1553414.21	1319751.26	1116963.73	942141.71	792367.31	5(
2020-12-31	0.038374	0.01	1607579.65	1360793.16	1147501.04	964371.75	808110.15	5(
2021-01-31	-0.010214	0.00	1593302.50	1350174.36	1139797.02	958961.97	804483.32	5(

433 rows × 8 columns

In [20]:
```
# Plotting the portfolio value growths for each weight:

plt.figure(figsize = (13,8))
plt.plot(df.index, df['VFINX_Value'], label = '100% in VFINX', alpha = 0.6)
plt.plot(df2.index, df2['w1=0.9'], label = "90% in VFINX", alpha = 0.6)
plt.plot(df2.index, df2['w2=0.8'], label = '80% in VFINX', alpha = 0.6)
plt.plot(df2.index, df2['w3=0.7'], label = '70% in VFINX', alpha = 0.6)
plt.plot(df2.index, df2['w4=0.6'], label = '60% in VFINX', alpha = 0.6)
plt.plot(df2.index, df2['w5=0.5'], label = '50% in VFINX', alpha = 0.6)
plt.plot(df.index, df['Portfolio'], label = "Louise's Portfolio", lw = 5)
plt.title("Figure 3: Comparison of Differently Weighted Portfolios in VFINX and
plt.xlabel('Date')
plt.ylabel('Portfolio Value in Millions')
plt.legend(fontsize = 12)
plt.grid()
plt.show()
```



We can see that as the weight of the portfolio in VFINX is higher, the portfolio value will be higher. Since the Risk-Free Rate from 1985 onwards has a much lower mean and variance than the VFINX returns since 1985 (as illustrated by Figure 1), the addition of Treasury Bonds in the portfolio will only serve to 'dampen' the returns that could otherwise be made if 100% of the portfolio had been invested in the Vanguard Fund.

We can see that the endpoint of Louise's portfolio lies somewhere between the endpoint values of the '70% in VFINX' and the '80% in VFINX' portfolios. We now use trial and error to find the weight in VFINX that would give the same final portfolio value as Louise's portfolio in 2021 (approximately USD1.21m), with some tolerance. The weight in VFINX that satisfied this condition was found to be 0.734, implying a 0.266 weight in Treasury Bonds, which yields a final portfolio value of approximately USD1.21m.

In [21]:
```
# Creating a DataFrame for the 73.4% VFINX-weighted portfolio:

df3 = pd.DataFrame(
    {
        'Date': pd.date_range(start ='1985-01-01', end = '2021-01-31', freq='M')
        'Return_VFINX': ret2, 'RF': rf}
)
df3.set_index('Date', inplace=True)

df3['w6=0.734'] = np.nan
df3['Investment'] = 500

iloc_Invest = df3.columns.get_loc('Investment')

df3.iloc[0, 2] = lump * (1+((df3.iloc[0,0]*0.734) + ((df3.iloc[0, 1]/100)*0.266)

i = 1
while i < len(df3):
```

```
    df3.iloc[i, 2] = df3.iloc[i-1,2] * (1+((df3.iloc[i,0]*0.734) + ((df3.iloc[i,
    df3['w6=0.734'] = df3['w6=0.734'].apply(lambda x: round(x, decimals))
    i += 1

df3
```

Out[21]:

Date	Return_VFINX	RF	w6=0.734	Investment
1985-01-31	0.075820	0.65	528.69	500
1985-02-28	0.013810	0.58	1034.87	500
1985-03-31	0.000053	0.62	1536.62	500
1985-04-30	-0.002843	0.72	2036.36	500
1985-05-31	0.060333	0.66	2630.11	500
...	...	...	...	...
2020-09-30	-0.040059	0.01	1115610.24	500
2020-10-31	-0.026713	0.01	1094265.42	500
2020-11-30	0.109356	0.01	1182628.58	500
2020-12-31	0.038374	0.01	1216470.73	500
2021-01-31	-0.010214	0.00	1207851.18	500

433 rows × 4 columns

This means that a portfolio that invests 73.4% into VFINX and 26.6% into Treasury Bonds approximately replicates the returns obtained by Louise's strategy (seen by the last value in the 'w6=0.734' column. We will call this the replicating portfolio.

We will continue under the assumption that Louise is risk-averse. That is to say, for two strategies that have the same return but different standard deviations (i.e. risk), Louise will choose to follow the strategy that has a lower standard deviation. Below, we will compare the volatilities of Louise's portfolio vs. the replicating portfolio.

In [22]:

```
df4 = pd.DataFrame(
    {
        'Date': pd.date_range(start ='1985-01-01', end = '2021-01-31', freq='M')
        'Portfolio': df['Portfolio'], 'w6=0.734': df3['w6=0.734']}
)
df4.set_index('Date', inplace=True)

df4['StDev_Port'] = df4['Portfolio'].rolling(window=12, min_periods=1).std()
df4['StDev_w6'] = df4['w6=0.734'].rolling(window=12, min_periods=1).std()
df4 = df4.apply(lambda x: round(x, decimals))
df4
```

Out[22]:

Date	Portfolio	w6=0.734	StDev_Port	StDev_w6
1985-01-31	1037.91	528.69	NaN	NaN
1985-02-28	1552.24	1034.87	363.69	357.92

Date	Portfolio	w6=0.734	StDev_Port	StDev_w6
1985-03-31	2052.32	1536.62	507.22	503.97
1985-04-30	2546.49	2036.36	648.86	648.70
1985-05-31	3063.30	2630.11	797.71	823.37
...	...	...	...	...
2020-09-30	1251361.42	1115610.24	60144.58	62417.65
2020-10-31	1218433.28	1094265.42	59710.86	63243.43
2020-11-30	1219055.12	1182628.58	60353.86	73846.27
2020-12-31	1219677.03	1216470.73	60226.21	86415.01
2021-01-31	1207719.83	1207851.18	59674.90	94471.54

433 rows × 4 columns

In [23]:
```
print(df4['Portfolio'].std())
print(df4['w6=0.734'].std())
```

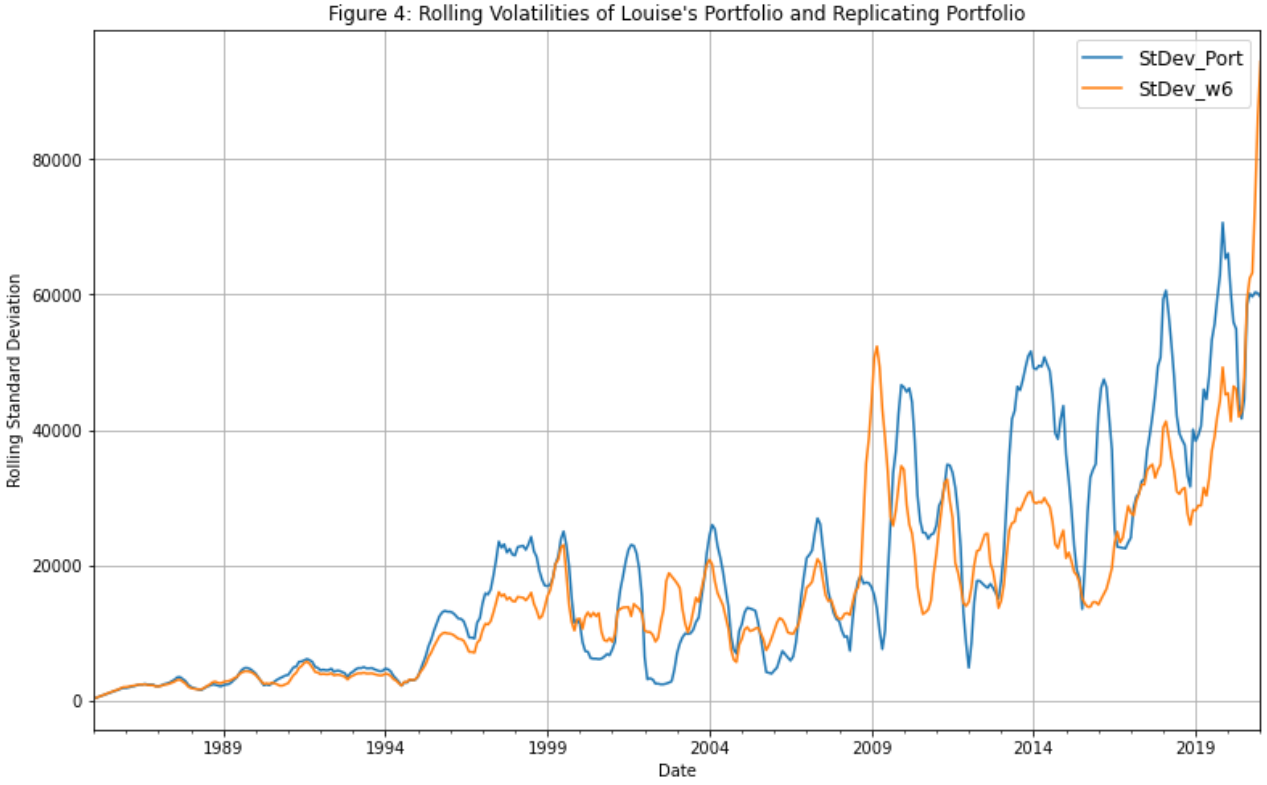
```
334317.9581724983
290408.87298166327
```

As a starting point, we can see that the 73.6% VFINX-weighted portfolio has a lower standard deviation, i.e. lower risk, than Louise's portfolio. However, we would also like to see how the volatilities of these two portfolios change over the entire 36-year period. This will be shown below, where we plot rolling standard deviations, or rolling volatilites for each portfolio.

In [24]:
```
df4[['StDev_Port', 'StDev_w6']].plot(figsize=(13,8), grid = True)
plt.title("Figure 4: Rolling Volatilities of Louise's Portfolio and Replicating
plt.xlabel('Date')
plt.ylabel('Rolling Standard Deviation')
plt.legend(fontsize = 12)
```

Out[24]:
```
<matplotlib.legend.Legend at 0x7ffe18046fa0>
```

Figure 4: Rolling Volatilities of Louise's Portfolio and Replicating Portfolio



From the plot above, we can observe that both these portfolios' volatilities generally rise and fall in tandem. For most of the time period, the standard deviation of the 73.4% weighted portfolio is beneath that of Louise's portfolio. Additionally, the amplitude of volatility change for Louise's portfolio is greater than the 73.4% weighted replicating portfolio, which means from a risk-assessment standpoint, Louise's portfolio is not optimal because it yields the same return as the replicating portfolio, just with higher risk. As a risk-averse investor, it would be prudent of her to instead invest in the 73.4% VFINX-weighted portfolio if she wanted to generate similar returns but by taking less risk.

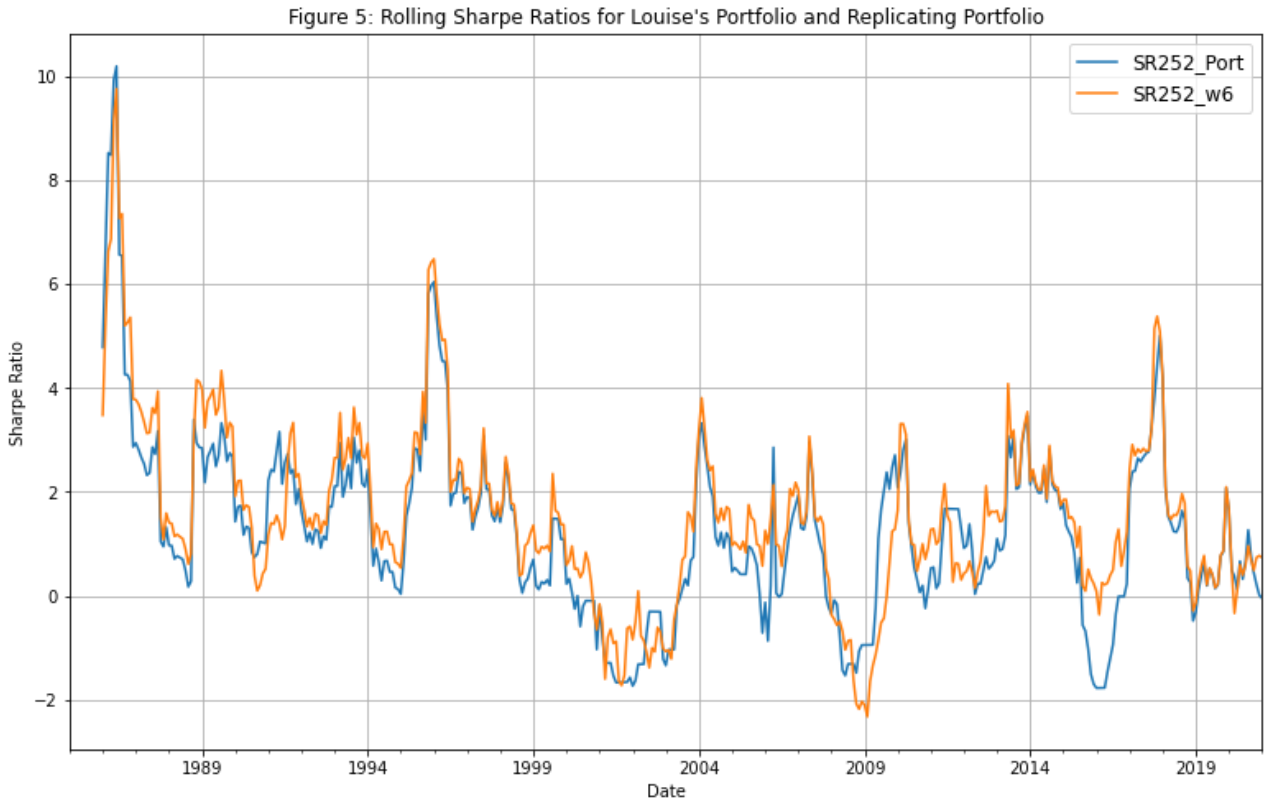
Examining Sharpe Ratios of Louise's Portfolio vs Replicating Portfolio

We can also examine how much return Louise is able to obtain for a given level of risk. The Sharpe Ratio is a common metric to measure this, and is calculated by dividing the excess returns of a portfolio over the risk-free rate by the portfolio's standard deviation.

In [25]:
```
df4['Portfolio_Return'] = df4['Portfolio'].pct_change()
df4['w6_Return'] = df4['w6=0.734'].pct_change()
df4['RF'] = ff['RF']

# ExRet refers to Excess Returns, i.e. how much more the return on the asset mad
df4['Port_ExRet'] = df4['Portfolio_Return'] - (df4['RF']/100)
df4['w6_ExRet'] = df4['w6_Return'] - (df4['RF']/100)

# Adding columns for Annualized Rolling Sharpe Ratios:
df4['SR252_Port'] = np.sqrt(12) * df4['Port_ExRet'].rolling(12, min_periods=1).m
df4['SR252_w6'] = np.sqrt(12) * df4['w6_ExRet'].rolling(12, min_periods=1).mean(

# Plotting the Rolling Sharpe Ratios for both portfolios:
```

```
df4[['SR252_Port','SR252_w6']].plot(figsize=(13, 8), grid = True)
plt.title("Figure 5: Rolling Sharpe Ratios for Louise's Portfolio and Replicatin
plt.xlabel('Date')
plt.ylabel('Sharpe Ratio')
plt.legend(fontsize = 12)
```

Out[25]: <matplotlib.legend.Legend at 0x7ffe18a8cfa0>



(NOTE: Both portfolios' Sharpe Ratios are much higher at the start of the holding period than later. This is because at the start, when the portfolio value is, for instance, only around 1000 for Louise's portfolio, the addition of the 500 monthly investment factors into the returns, which would offset a negative return and instead creating a very high, positive percentage return. As the holding period increases, the 500 monthly investments become an increasingly smaller proportion of the total portfolio value, so the Sharpe Ratios are lower.)

From Figure 5 above, we can see that generally, the Sharpe Ratio for the 73.4% VFINX-weighted portfolio remains above Louise's portfolio, with the notable exceptions in the years of 1990 and 2009. This indicates that per unit risk, the 73.4% VFINX-weighted portfolio is generating higher returns than Louise's portfolio. This corroborates what we observed in Figure 4 with the rolling volatilities, confirming that Louise's strategy is not optimal from a risk-assessment standpoint.

However, not all risk is the same. As investors, we would LIKE to have upside risk, since upside risk essentially means higher returns. The only risk that investors are really wary of is downside risk. In a fashion similar to how we compared Sharpe Ratios for the two portfolios, we can go a step further and also compare Sortino Ratios, which measure the excess return enjoyed by the investor per unit of DOWNSIDE risk. We will plot the rolling Sortino Ratios for Louise's portfolio, a 100% VFINX-weighted portfolio and a 50% VFINX-weighted portfolio.

To illustrate the Sortino Ratio, we have taken simple portfolios that do not include the 500 dollar monthly investments. This is because even if there is a negative return for a particular month, the 500 dollar monthly investment will offset it, changing it to a positive value. For the Sortino Ratio, we want to strictly isolate all the negative returns, so including to 500 dollar monthly investments will skew the results and lead to very few actual negative return values. The method is shown below.

Sortino Ratios

In [26]:
```
df5 = pd.DataFrame(
    {
        'Date': pd.date_range(start ='1985-01-01', end = '2021-01-31', freq='M')
        'Return_GSPC': df['Return_GSPC'], 'Return_VFINX': df['Return_VFINX']}
)
df5.set_index('Date', inplace=True)
df5['RF'] = ff['RF']
df5['Portfolio'] = np.nan
df5['100%_Value'] = np.nan
df5['50%_Value'] = np.nan
lump = 500

# Initializing the first two portfolio values for Louise's portfolio (excluding
df5.iloc[0, 3] = lump * (1 + df5.iloc[0, 1])
df5.iloc[1, 3] = df5.iloc[0,3] * (1 + df5.iloc[1, 1])

# Filling out the remainder of Louise's portfolio column:
i=2
t=1
while i < len(df5):
    j = df5.iloc[i-2]['Return_GSPC']
    k = df5.iloc[i-1]['Return_GSPC']
    if ((t==1) and (j<0 and k <0)):
        df5.iloc[i, 3] = df5.iloc[i-1, 3]*(1 + df5.iloc[i, 2]/100)
        t = 0
    elif ((t==1) and (j>0 or k>0)):
        df5.iloc[i, 3] = df5.iloc[i-1, 3]*(1 + df5.iloc[i, 1])
    elif ((t==0) and (j<0 or k<0)):
        df5.iloc[i, 3] = df5.iloc[i-1, 3]*(1 + df5.iloc[i, 2]/100)
    elif ((t==0) and (j>0 and k>0)):
        df5.iloc[i, 3] = df5.iloc[i-1, 3]*(1 + df5.iloc[i, 1])
        t = 1
    df5['Portfolio'] = df5['Portfolio'].apply(lambda x: round(x, decimals))
    i += 1

# Filling out the column for the 100% VFINX-weighted portfolio values:
df5.iloc[0, 4] = lump * (1+(df5.iloc[0,1]))
i = 1
while i < len(df5):
    df5.iloc[i, 4] = df5.iloc[i-1,4] * (1+(df5.iloc[i,1]))
    df5['100%_Value'] = df5['100%_Value'].apply(lambda x: round(x, decimals))
    i += 1

# Filling out the column for the 50% VFINX-weighted portfolio values:
df5.iloc[0, 5] = lump * (1+((df5.iloc[0,1]*0.5) + ((df5.iloc[0, 2]/100)*0.5)))
i = 1
while i < len(df5):
    df5.iloc[i, 5] = df5.iloc[i-1,5] * (1+((df5.iloc[i,1]*0.5) + ((df5.iloc[i, 2
```

```
    df5['50%_Value'] = df5['50%_Value'].apply(lambda x: round(x, decimals))
    i += 1

df5
```

Out[26]:

Date	Return_GSPC	Return_VFINX	RF	Portfolio	100%_Value	50%_Value
1985-01-31	0.074085	0.075820	0.65	537.91	537.91	520.58
1985-02-28	0.008629	0.013810	0.58	545.34	545.34	525.68
1985-03-31	-0.002870	0.000053	0.62	545.37	545.37	527.32
1985-04-30	-0.004594	-0.002843	0.72	543.82	543.82	528.47
1985-05-31	0.054051	0.060333	0.66	547.41	576.63	546.16
...	...	...	...	...	...	...
2020-09-30	-0.039228	-0.040059	0.01	11137.88	17394.42	5794.63
2020-10-31	-0.027666	-0.026713	0.01	10840.35	16929.76	5717.52
2020-11-30	0.107546	0.109356	0.01	10841.43	18781.14	6030.43
2020-12-31	0.037121	0.038374	0.01	10842.51	19501.85	6146.44
2021-01-31	-0.011137	-0.010214	0.00	10731.77	19302.67	6115.05

433 rows × 6 columns

In [27]:
```
# Before calculating the Sortino Ratio, we must have the returns on the portfoli
df5['Port_Return'] = df5['Portfolio'].pct_change()
df5['100%_Return'] = df5['100%_Value'].pct_change()
df5['50%_Return'] = df5['50%_Value'].pct_change()
df5
```

Out[27]:

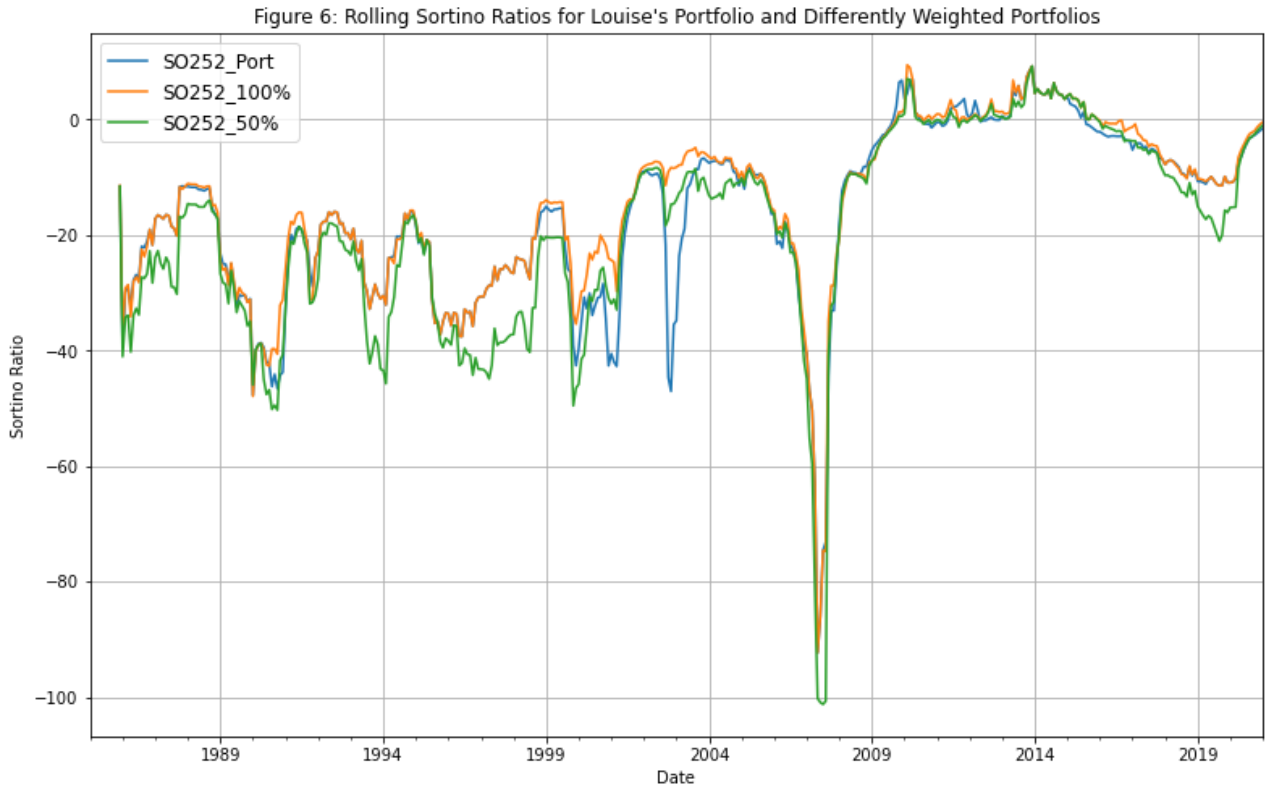
Date	Return_GSPC	Return_VFINX	RF	Portfolio	100%_Value	50%_Value	Port_Return	100%_R
1985-01-31	0.074085	0.075820	0.65	537.91	537.91	520.58	NaN	
1985-02-28	0.008629	0.013810	0.58	545.34	545.34	525.68	0.013813	0.0
1985-03-31	-0.002870	0.000053	0.62	545.37	545.37	527.32	0.000055	0.0(
1985-04-30	-0.004594	-0.002843	0.72	543.82	543.82	528.47	-0.002842	-0.0(
1985-05-31	0.054051	0.060333	0.66	547.41	576.63	546.16	0.006601	0.0(
...	...	...	...	...	...	...	...	
2020-09-30	-0.039228	-0.040059	0.01	11137.88	17394.42	5794.63	-0.040059	-0.0(

Date	Return_GSPC	Return_VFINX	RF	Portfolio	100%_Value	50%_Value	Port_Return	100%_R
2020-10-31	-0.027666	-0.026713	0.01	10840.35	16929.76	5717.52	-0.026713	-0.0
2020-11-30	0.107546	0.109356	0.01	10841.43	18781.14	6030.43	0.000100	0.1
2020-12-31	0.037121	0.038374	0.01	10842.51	19501.85	6146.44	0.000100	0.0
2021-01-31	-0.011137	-0.010214	0.00	10731.77	19302.67	6115.05	-0.010214	-0.0

433 rows × 9 columns

In [28]:
```
# Creating columns for the excess returns and negative excess returns for the po
df5['Port_ExRet'] = df5['Port_Return'] - df['RF']
df5['100%_ExRet'] = df5['100%_Return'] - df['RF']
df5['50%_ExRet'] = df5['50%_Return'] - df['RF']
df5['Port_ExRet_neg'] = np.where(df5['Port_ExRet'] < 0, df5['Port_ExRet'], 0)
df5['100%_ExRet_neg'] = np.where(df5['100%_ExRet'] < 0, df5['100%_ExRet'], 0)
df5['50%_ExRet_neg'] = np.where(df5['50%_ExRet'] < 0, df5['50%_ExRet'], 0)

# Adding columns for Annualized Rolling Sortino Ratios:
df5['SO252_Port'] = np.sqrt(12) * df5['Port_ExRet'].rolling(window=12, min_perio
df5['SO252_100%'] = np.sqrt(12) * df5['100%_ExRet'].rolling(window=12, min_perio
df5['SO252_50%'] = np.sqrt(12) * df5['50%_ExRet'].rolling(window=12, min_periods

# Plotting the Rolling Sortino Ratios for the portfolios:
df5[['SO252_Port','SO252_100%','SO252_50%']].plot(figsize=(13, 8), grid = True)
plt.title("Figure 6: Rolling Sortino Ratios for Louise's Portfolio and Different
plt.xlabel('Date')
plt.ylabel('Sortino Ratio')
plt.legend(fontsize = 12)
```

Out[28]:
```
<matplotlib.legend.Legend at 0x7ffe1ca75e50>
```



From Figure 6 above, we notice that the Sortino Ratio for the 100% portfolio (orange line) is typically above the other Sortino Ratios, and the 50% portfolio (green line) is typically the lowest, with a few exceptions (e.g 2003). We recall that the Sortino Ratio is the excess return enjoyed per unit of DOWNSIDE risk, so this implies that the higher the portfolio weight in VFINX, the greater the return per unit of downside risk. This adds to our findings from Figure 5: that with a naïve strategy that maintains a high weight in VFINX, an investor can expect to make higher returns for the amount of risk that they take, implying that Louise's portfolio, which generates equivalent returns to a 73.4% VFINX-weighted portfolio, can be optimized further for higher returns and a better reward-to-risk ratio.

Tradeoff

We established earlier in Figure 3 that the higher the percentage of the portfolio invested in VFINX, the higher the expected returns. However, there must be some tradeoff; it is not likely that we can get higher returns without some cost. Below, we plot the volatilities of Louise's portfolio, as well as the volatilities of the portfolios that are 100% weighted in VFINX, and 50% weighted in VFINX.

In [29]:
```
df6 = pd.DataFrame(df[['Portfolio','VFINX_Value']])
df6['StDev_Port'] = df['Portfolio'].rolling(window=12, min_periods=1).std()
df6['StDev_VFINX'] = df['VFINX_Value'].rolling(window=12, min_periods=1).std()
df6['StDev_50%'] = df2['w5=0.5'].rolling(window=12, min_periods=1).std()
df6
```

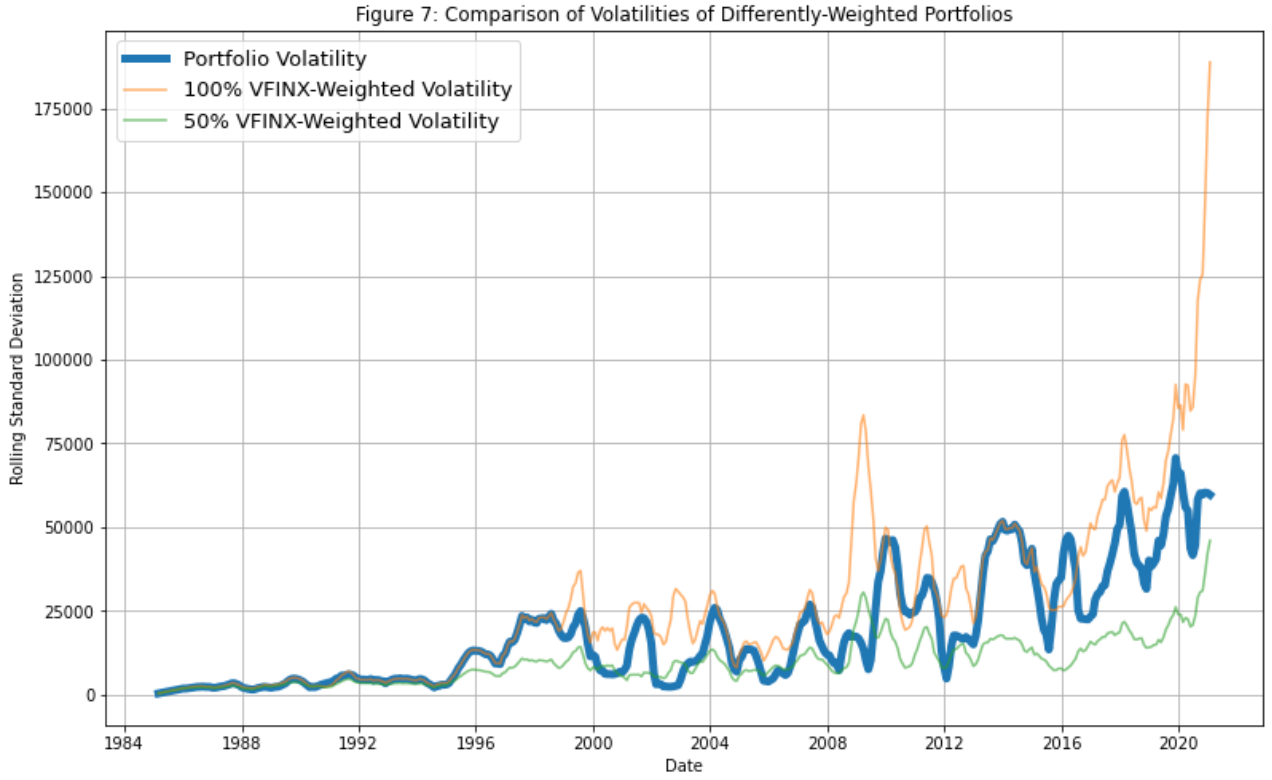
Out[29]:

Date	Portfolio	VFINX_Value	StDev_Port	StDev_VFINX	StDev_50%
1985-01-31	1037.91	1037.91	NaN	NaN	NaN

Date	Portfolio	VFINX_Value	StDev_Port	StDev_VFINX	StDev_50%
1985-02-28	1552.24	1552.24	363.686231	363.686231	357.159635
1985-03-31	2052.32	2052.32	507.221681	507.221681	504.155295
1985-04-30	2546.49	2546.49	648.857565	648.857565	650.420039
1985-05-31	3063.30	3200.13	797.711533	842.248286	817.040736
...	...	...	...	...	...
2020-09-30	1251361.42	1701507.88	60144.580376	123932.423279	30611.117358
2020-10-31	1218433.28	1656554.79	59710.858225	125498.633758	31013.101084
2020-11-30	1219055.12	1838209.65	60353.862884	146719.667431	36157.980785
2020-12-31	1219677.03	1909249.45	60226.211830	172389.087056	42152.054368
2021-01-31	1207719.83	1890249.29	59674.904890	188854.955632	45990.418043

433 rows × 5 columns

In [30]:
```
plt.figure(figsize = (13,8))
plt.plot(df6.index, df6['StDev_Port'], label = "Portfolio Volatility", lw = 5)
plt.plot(df6.index, df6['StDev_VFINX'], label = "100% VFINX-Weighted Volatility"
plt.plot(df6.index, df6['StDev_50%'], label = "50% VFINX-Weighted Volatility", a
plt.title('Figure 7: Comparison of Volatilities of Differently-Weighted Portfoli
plt.xlabel('Date')
plt.ylabel('Rolling Standard Deviation')
plt.grid()
plt.legend(fontsize = 13)
plt.show()
```



We can see from Figure 7 above that the rolling standard deviation, i.e. the volatility, for the portfolios increases when the weight in VFINX increases. Again, the volatilities move in tandem; they all tend to spike and fall at the same time. However, the 50% VFINX-weighted portfolio spikes the least, whereas the 100% VFINX-weighted portfolio tends to see the biggest jumps in volatility, with Louise's portfolio somewhere in between.

Although all investors would like to see higher returns, this will come at a cost, which is enduring greater volatility, or greater risk. One of our earlier assumptions was that Louise is a risk-averse investor. This makes sense as her original strategy was to include some investment into Treasury Bonds, which are considered very safe. For a risk-averse investor, generating higher returns may not be the optimal strategy depending on their personal level of risk aversion. A very conservative investor may choose to weigh their portfolio in VFINX as low as 50-60%, whereas a risk-seeking investor may be willing to endure greater risk for the prospect of gaining much higher return on their investments.

Conclusion

We have shown throughout the course of the project that Louise's strategy is suboptimal. First, we examined how her portfolio values would differ if she adopted naïve investment strategies by changing the weight of her portfolio invested in VFINX. Based on the data in Figure 3, the higher the proportion invested in VFINX, the greater returns she would have enjoyed. We then found a naïve portfolio that would replicate Louise's returns over the 36 year period, and found that even though they produced the same returns, the naïve portfolio had lower risk, shown by Figure 4, which plotted the rolling standard deviations of Louise's portfolio and the replicating portfolio. This is compelling evidence that Louise has not chosen the optimal strategy, since she could produce the same returns for lower risk. This is especially important given our assumption of Louise being a risk-averse investor; a rational investor would not choose her investment strategy in the face of a better alternative. The rolling Sharpe Ratios also confirmed this idea, where the replicating portfolio generated more excess returns per unit risk than Louise's portfolio did. The Sortino ratio had a similar effect in showing that Louise may not be getting appropriately compensated for the risk she is bearing, when there are other portfolios that have a higher reward-to-risk ratio. Furthermore, if our assumption about zero transaction costs is relaxed, the gap between Louise's portfolio and the 100% VFINX-weighted portfolio, for instance, would widen even further, since an investor in the latter is not required to pay any transaction fees.

We must remain cognizant of the fact that there is a tradeoff associated with a change in investment strategy. We illustrated in the final section of the project how despite generating higher returns, a portfolio that has 100% of it's investments in VFINX also carries the higher risk, and this risk decreases as the weight in VFINX decreases. Indeed, the 50% VFINX-weighted portfolio is below the other line graphs. It is therefore important to note that expected returns are not the only attractive quality of a portfolio, but also that a risk-averse investor may choose to forgo higher returns in favour less exposure to risk.

To conclude, we believe that Louise's strategy is still suboptimal, and that she would be better off strictly from a return-oriented point of view by investing in a portfolio predominantly featuring VFINX. Even a small exposure to Treasury Bonds would keep her risk level down, and yet still have higher returns than her current strategy. Deviating from the scope of this project, Louise could also invest in different types of funds that may have many more companies, thereby providing more diversification benefits, or, if she wanted higher risk and higher reward, she could invest in funds such as emerging market funds, or sector-specific funds. Despite its suboptimality, we acknowledge the caveat that depending on the investor's risk profile, they may instead think that the portfolios that generate higher returns are suboptimal and would prefer to settle for both less risk and return.