In the given strategy , I have applied Bollinger Band mean regression to assign signals ,volatility comparison as an Indicator and RVI(Relative Volatility Index) also as an indicator
Since Volatility Itself is not Feasible to be used to Buy/Sell , but more important to consolidate the amount to be traded or the number of shares to be bought.

The Dataset used for the implementation is of APPLE , from 1 Jan 2023 to 1 Jan 2024.

I first created a function to calculate the volatility per data. I first found the Standard deviation for the data with a rolling window of 10 periods, then I replaced the first NaN values by the average of all the deviations and then I divided the standard deviation by the period, which is 10 to get the Volatility for each data and created a column in the dataset for
it.

```
def calculate_vol(data,period):

    std=data.Close.rolling(period,min_periods=1).std()
    mean=std.mean()
    std.fillna(mean, inplace=True)
    vol=std/period

    return vol
```

Then I calculated the mean volatility and standard deviation of the volatilities to use them later and also I defined two columns for change in two successive closing prices and for
mod(change in two successive closing prices) to use further.

```
df["Volatility"]=calculate_vol(df,10)
avg_vol=df["Volatility"].mean()
print(avg_vol)
dev_vol=df["Volatility"].std()
print(dev_vol)
df["price_change"]=df['Close'].diff()
df["price_change"].fillna(0, inplace=True)
df['AbsChange'] = abs(df['price_change'])
df["AbsChange"].fillna(0, inplace=True)
```

Now , to write the function for the RVI Indicator , I first defined the period of two weeks and found out the standard deviation for the data for which the price change is positive and
separately found out the deviation for the data for which the price change is negative . Then , according the formula of RVI-(std positive)/(std positive + std negative) * 100, I found the
RVI for each data point.

```
def calculate_rvi(data, period=14):
    data['RVI'] = np.nan

    for i in range(period, len(data)):
        positive_changes = data['price_change'].iloc[i-period+1:i+1][data['price_change'].iloc[i-period+1:i+1] > 0].sum()
        negative_changes = abs(data['price_change'].iloc[i-period+1:i+1][data['price_change'].iloc[i-period+1:i+1] < 0].sum())

        if positive_changes + negative_changes != 0:
            rvi = (positive_changes / (positive_changes + negative_changes)) * 100
            data.loc[data.index[i], 'RVI'] = rvi

    return data['RVI']
```

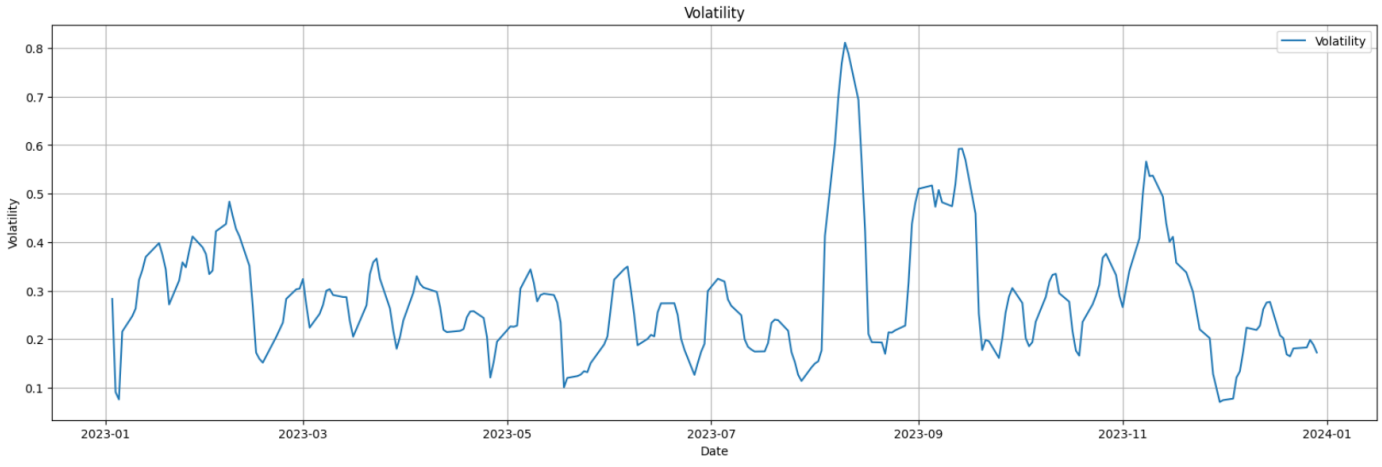
Now , coming to our main strategy, I have defined the moving average line for a window period of 10 days and the upper and lower line as (Moving average + 2*std deviation) and
(Moving average - 2*std deviation) respectively.

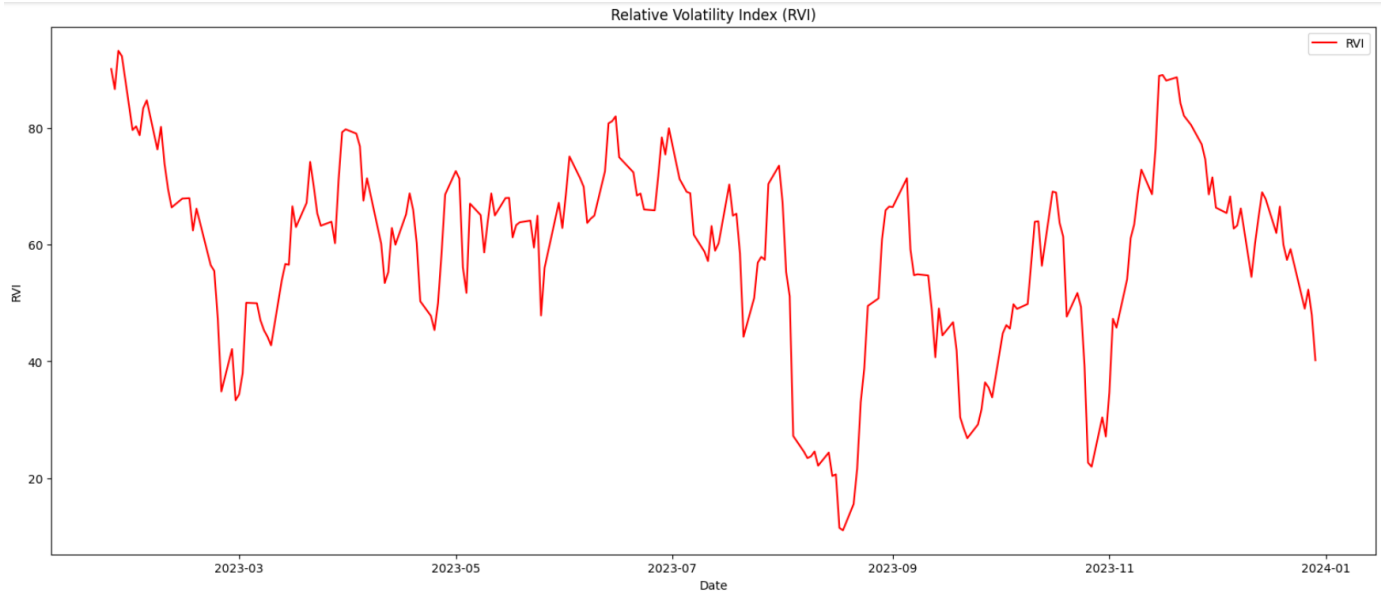
```
win=10
df['MA']=df['Close'].rolling(window=win).mean()
df['STD']=df['Close'].rolling(window=win).std()
df['Upper']=df['MA']+2*df['STD']
df['Lower']=df['MA']-2*df['STD']
```

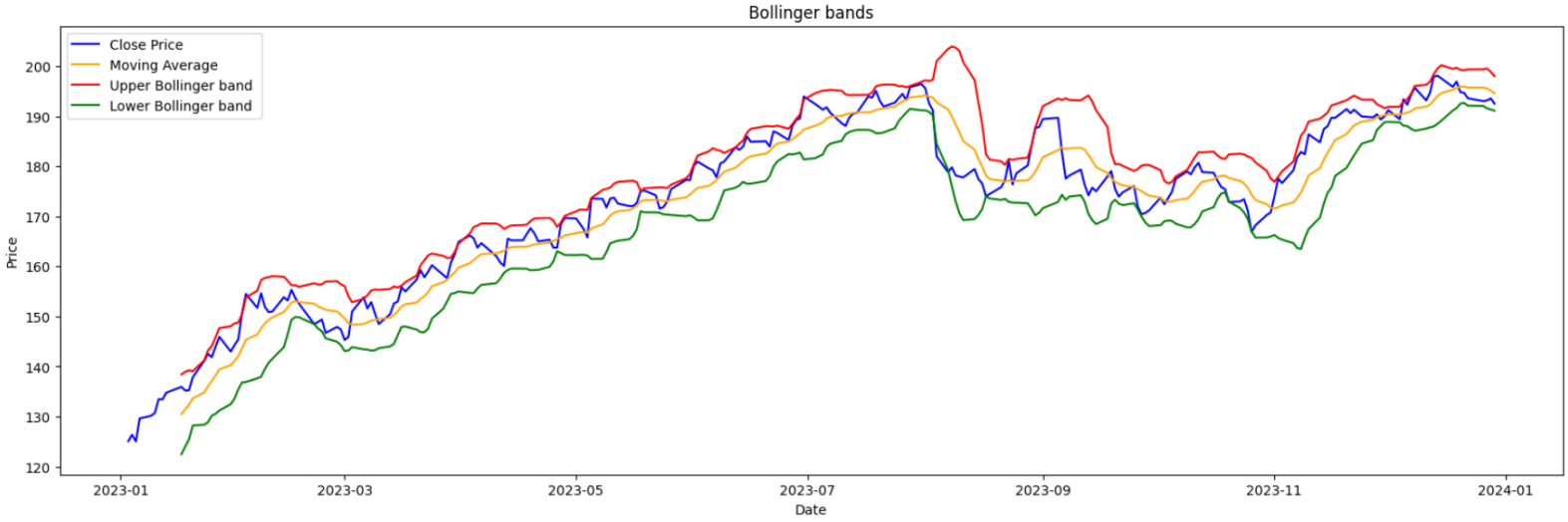
Finally, I created a function to assign signals according to the Bollinger bands. I assigned a sell signal if the closing price went above the upper band or if the closing price peaked between the MA line and the upper line . Similarly, I assigned a Buy signal if the closing price went below the Lower band or if the closing price troughed between the MA line and the Lower line.

```
def BBMR(data):
    data['signal_bbmr']=0
    for i in range(1,len(data)-2):
        # Sell
        if(data['Close'][i] > data['Upper'][i]):
          data.loc[data.index[i], 'signal_bbmr'] = 1
        if((data['Close'][i]<data['Upper'][i])and(data['Close'][i]>data['MA'][i])and(data['Close'][i]<data['Close'][i-1])and(data['Close'][i-1]>data['Close'][i-2])):
          data.loc[data.index[i], 'signal_bbmr'] = 1
        # Buy
        if(data['Close'][i] < data['Lower'][i]):
          data.loc[data.index[i], 'signal_bbmr'] = -1
        if((data['Close'][i]>data['Lower'][i])and(data['Close'][i]<data['MA'][i])and(data['Close'][i-1]<data['Close'][i])and(data['Close'][i-1]<data['Close'][i-2])):
          data.loc[data.index[i], 'signal_bbmr'] = -1
    return data
BBMR(df)
```

I plotted the Bollinger band data(MA line , upper line, lower line and the closing price . On separate plots, I plotted volatility and RVI respectively.








```
plt.figure(figsize=(20,6))
plt.plot(df['Close'], label='Close Price', color='blue')
plt.plot(df['MA'], label='Moving Average', color='orange')
plt.plot(df['Upper'], label='Upper Bollinger band', color='red',)
plt.plot(df['Lower'], label='Lower Bollinger band ', color='green',)
plt.title('Bollinger bands')
plt.xlabel('Date')
plt.ylabel('Price')
plt.legend()
plt.show()

plt.figure(figsize=(20, 6))
plt.plot(df["Volatility"], label="Volatility")
plt.title('Volatility')
plt.xlabel('Date')
plt.ylabel('Volatility')
plt.legend()
plt.grid(True)
plt.show()

plt.figure(figsize=(20,8))
plt.plot(df["RVI"], label='RVI',color='red')
plt.title('Relative Volatility Index (RVI)')
plt.xlabel('Date')
plt.ylabel('RVI')
plt.legend()
plt.show()
```

Now, coming to use of the indicators , I first assigned the total capital to be 10000 .Now whatever I am doing further is relative of my risk appetite :

I am considering volatility as a primary indicator and RVI as a secondary Indicator and my logic goes-
1)if my volatility is greater than average volatility + std dev , I am not very interested to invest large capital in case of a buy signal , and will not sell the entire shares in case of a sell signal.
2)if my volatility is greater than average volatility, I am moderately interested to invest in case of a buy signal , and will sell considerable shares in case of a sell signal
3)if the volatility is lesser than the mean volatility , I will invest the entire capital or sell all the shares in case of a buy or sell signal respectively.

Now , as my risk appetite is not much, I also use RVI to put a cap on the capital when:-

1)->(case1)(RVI<=40 or 50<=RVI<=60)- I will sell only 15 shares at max and buy only 4000 worth of stocks at max.
->(case2)(not case1)-I will be extra cautious as RVI did not indicate ,so I will sell only 10 shares at max or buy 3000 worth of stocks at max.

2)->(case1)(RVI<=60 or 40<=RVI<=50)-I will sell only 20 shares at max and buy only 6000 worth of stocks at max.
->(case2)(not case1)-I will be extra cautious as RVI did not indicate ,so I will sell only 15 shares at max or buy 5000 worth of stocks at max.

3)I will invest the entire capital or sell the entire shares.

The returns for the given strategy are 11.39%.

```
def returns(data,mean_vol,dev):
  amount=10000
  position=0
  noofshare=0
  min=0
  max=0
  vol=data['Volatility']
  rvi=data["RVI"]
  sl=0
  for i in range(len(data)):
    sig=df['signal_bbmr'][i]
    price=df['Close'][i]

    if(sig==1):
      if(vol[i]>avg_vol+dev and noofshare>0 and ((rvi[i]>=50 and rvi[i]<=60) or (rvi[i]<=40))):
        if(noofshare<15):
          amount+=noofshare*price
          noofshare=0
        if(noofshare>15):
          amount+=15*price
          noofshare-=15

      if(vol[i]>avg_vol+dev and noofshare>0 and (not(rvi[i]>=50 and rvi[i]<=60) or (rvi[i]<=40))):
        if(noofshare<10):
          amount+=noofshare*price
          noofshare=0
        if(noofshare>10):
          amount+=10*price
          noofshare-=10

      if(vol[i]>avg_vol and noofshare>0 and (not(rvi[i]>=50 and rvi[i]<=60) or (rvi[i]<=40))):
        if(noofshare<15):
          amount+=noofshare*price
          noofshare=0
        if(noofshare>15):
          amount+=15*price
          noofshare-=15
```

```
    if(vol[i]>avg_vol and noofshare>0 and ((rvi[i]>=50 and rvi[i]<=60) or (rvi[i]<=40))):
      if(noofshare<20):
        amount+=noofshare*price
        noofshare=0
      if(noofshare>20):
        amount+=20*price
        noofshare-=20

    if(vol[i]<=avg_vol and noofshare>0):
      amount+=noofshare*price
      noofshare=0

  if(sig==-1):
    if(vol[i]>avg_vol+dev and ((rvi[i]>=40 and rvi[i]<=50) or (rvi[i]>=60))):
      if(amount<4000):
        noofshare+=(amount//price)
        amount-=(amount//price)*price
      if(amount>=4000):
        noofshare+=4000//price
        amount-=4000

    if(vol[i]>avg_vol+dev and (not(rvi[i]>=40 and rvi[i]<=50) or (rvi[i]>=60))):
      if(amount<3000):
        noofshare+=(amount//price)
        amount-=(amount//price)*price
      if(amount>=3000):
        noofshare+=3000//price
        amount-=3000

    if(vol[i]>avg_vol and ((rvi[i]>=40 and rvi[i]<=50) or (rvi[i]>=60))):
      if(amount<6000):
        noofshare+=(amount//price)
        amount-=(amount//price)*price
      if(amount>=6000):
        noofshare+=6000//price
        amount-=6000

    if(vol[i]>avg_vol and (not(rvi[i]>=40 and rvi[i]<=50) or (rvi[i]>=60))):
      if(amount<5000):
        if(amount>=3000):
          noofshare+=3000//price
          amount-=3000

        if(vol[i]>avg_vol and ((rvi[i]>=40 and rvi[i]<=50) or (rvi[i]>=60))):
          if(amount<6000):
            noofshare+=(amount//price)
            amount-=(amount//price)*price
          if(amount>=6000):
            noofshare+=6000//price
            amount-=6000

        if(vol[i]>avg_vol and (not(rvi[i]>=40 and rvi[i]<=50) or (rvi[i]>=60))):
          if(amount<5000):
            noofshare+=(amount//price)
            amount-=(amount//price)*price
          if(amount>=5000):
            noofshare+=5000//price
            amount-=5000

        if(vol[i]<=avg_vol and amount>0):
          noofshare+=(amount//price)
          amount-=(amount//price)*price



  amount+=noofshare*price
  return (amount-10000)/100



x=returns(df,avg_vol,dev_vol)
x
```

```
11.39569839477539
```