

# Healthcare\_Portfolio\_Growth

September 29, 2021

## 1 Healthcare Stocks Growth

```
[1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import math

import warnings
warnings.filterwarnings("ignore")

# yahoo finance data
import yfinance as yf
yf.pdr_override()
```

```
[2]: # input
# Growth Stock
title = "Healthcare Stocks Growth"
symbols = ['UTHR', 'AXSM', 'CCXI']
start = '2020-04-01'
end = '2020-07-09'
```

```
[3]: df = pd.DataFrame()
for s in symbols:
    df[s] = yf.download(s,start,end)['Adj Close']
```

```
[*****100%*****] 1 of 1 completed
[*****100%*****] 1 of 1 completed
[*****100%*****] 1 of 1 completed
```

```
[4]: from datetime import datetime
from dateutil import relativedelta

d1 = datetime.strptime(start, "%Y-%m-%d")
d2 = datetime.strptime(end, "%Y-%m-%d")
delta = relativedelta.relativedelta(d2,d1)
print('How many years of investing?')
print('%s years' % delta.years)
```

How many years of investing?  
0 years

```
[5]: number_of_years = delta.years
```

```
[6]: days = (df.index[-1] - df.index[0]).days  
days
```

```
[6]: 98
```

```
[7]: df.head()
```

```
[7]:
```

	UTHR	AXSM	CCXI
Date			
2020-04-01	93.900002	55.790001	38.580002
2020-04-02	93.529999	55.660000	39.750000
2020-04-03	92.739998	55.020000	38.959999
2020-04-06	95.760002	53.549999	43.500000
2020-04-07	95.059998	49.869999	42.599998

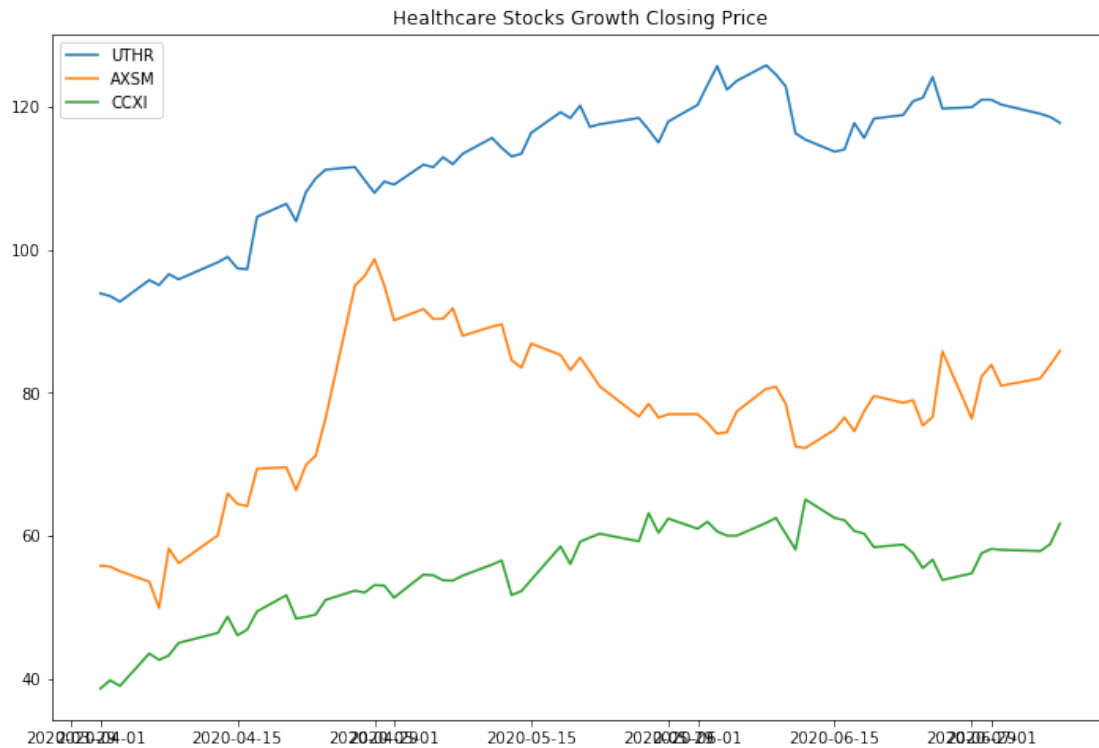
```
[8]: df.tail()
```

```
[8]:
```

	UTHR	AXSM	CCXI
Date			
2020-07-01	121.000000	83.910004	58.150002
2020-07-02	120.330002	80.970001	58.009998
2020-07-06	119.040001	82.010002	57.840000
2020-07-07	118.599998	83.879997	58.799999
2020-07-08	117.779999	85.839996	61.689999

```
[9]: plt.figure(figsize=(12,8))  
plt.plot(df)  
plt.title(title + ' Closing Price')  
plt.legend(labels=df.columns)
```

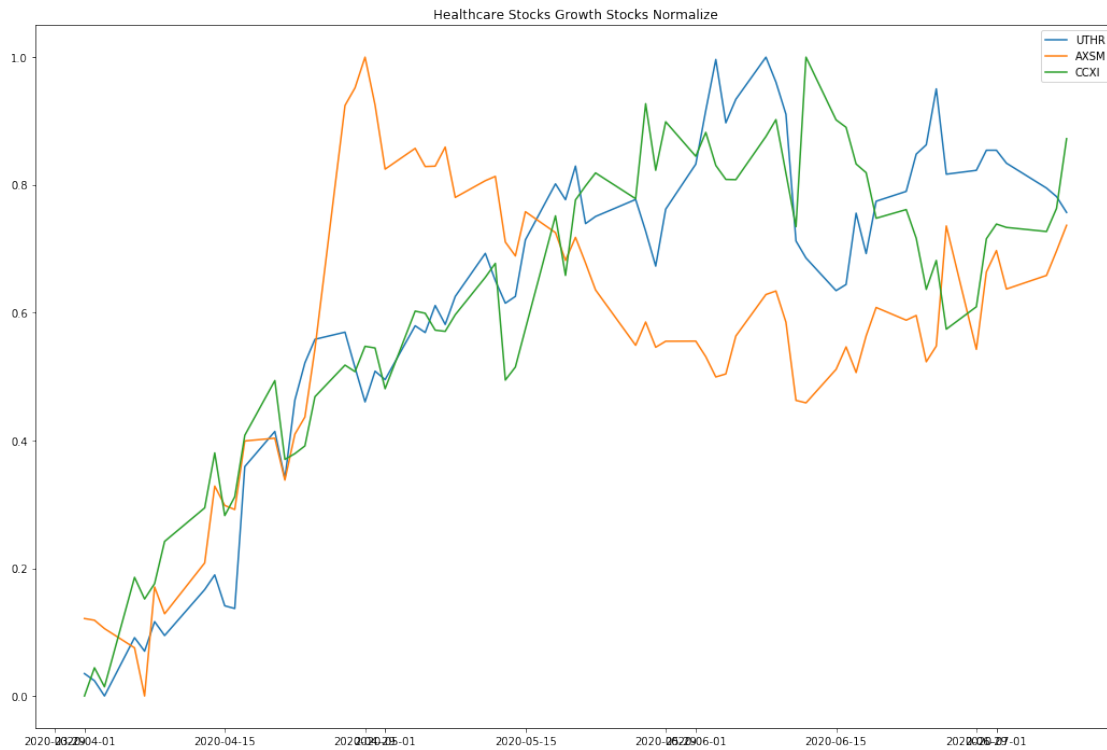
```
[9]: <matplotlib.legend.Legend at 0x2704ed5c668>
```



```
[10]: # Normalize the data
normalize = (df - df.min()) / (df.max() - df.min())
```

```
[11]: plt.figure(figsize=(18,12))
plt.plot(normalize)
plt.title(title + ' Stocks Normalize')
plt.legend(labels=normalize.columns)
```

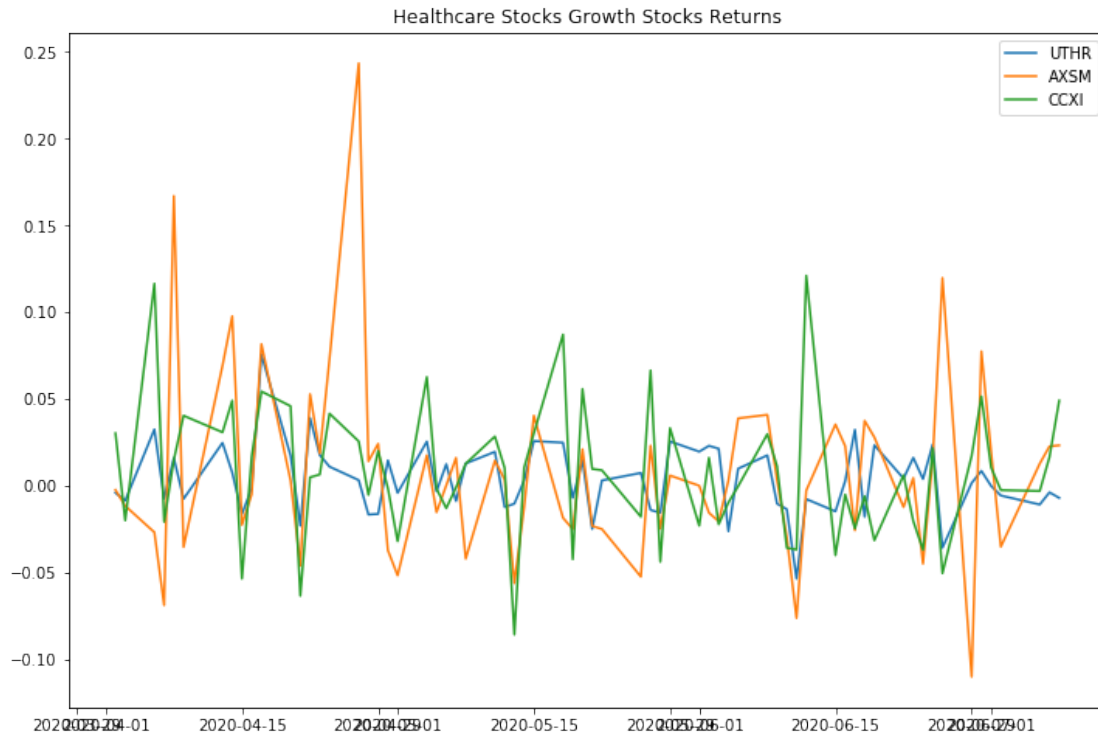
```
[11]: <matplotlib.legend.Legend at 0x2704efd15f8>
```



```
[12]: stock_rets = df.pct_change().dropna()
```

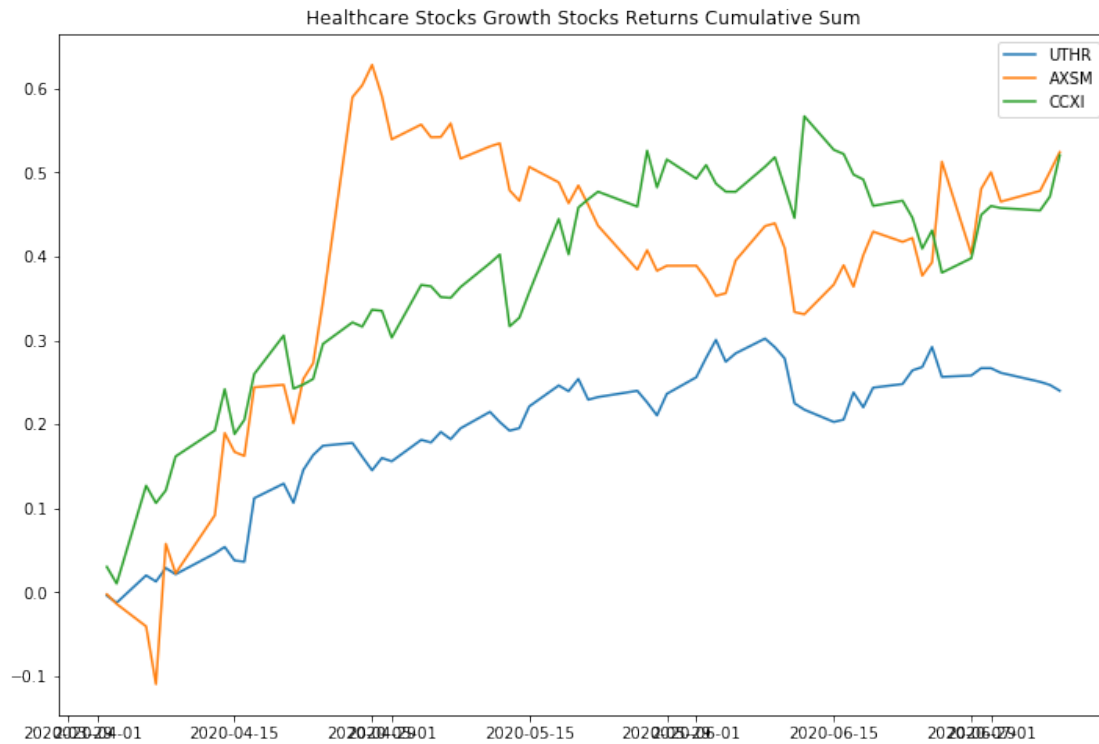
```
[13]: plt.figure(figsize=(12,8))
plt.plot(stock_rets)
plt.title(title + ' Stocks Returns')
plt.legend(labels=stock_rets.columns)
```

```
[13]: <matplotlib.legend.Legend at 0x2704edc9f28>
```



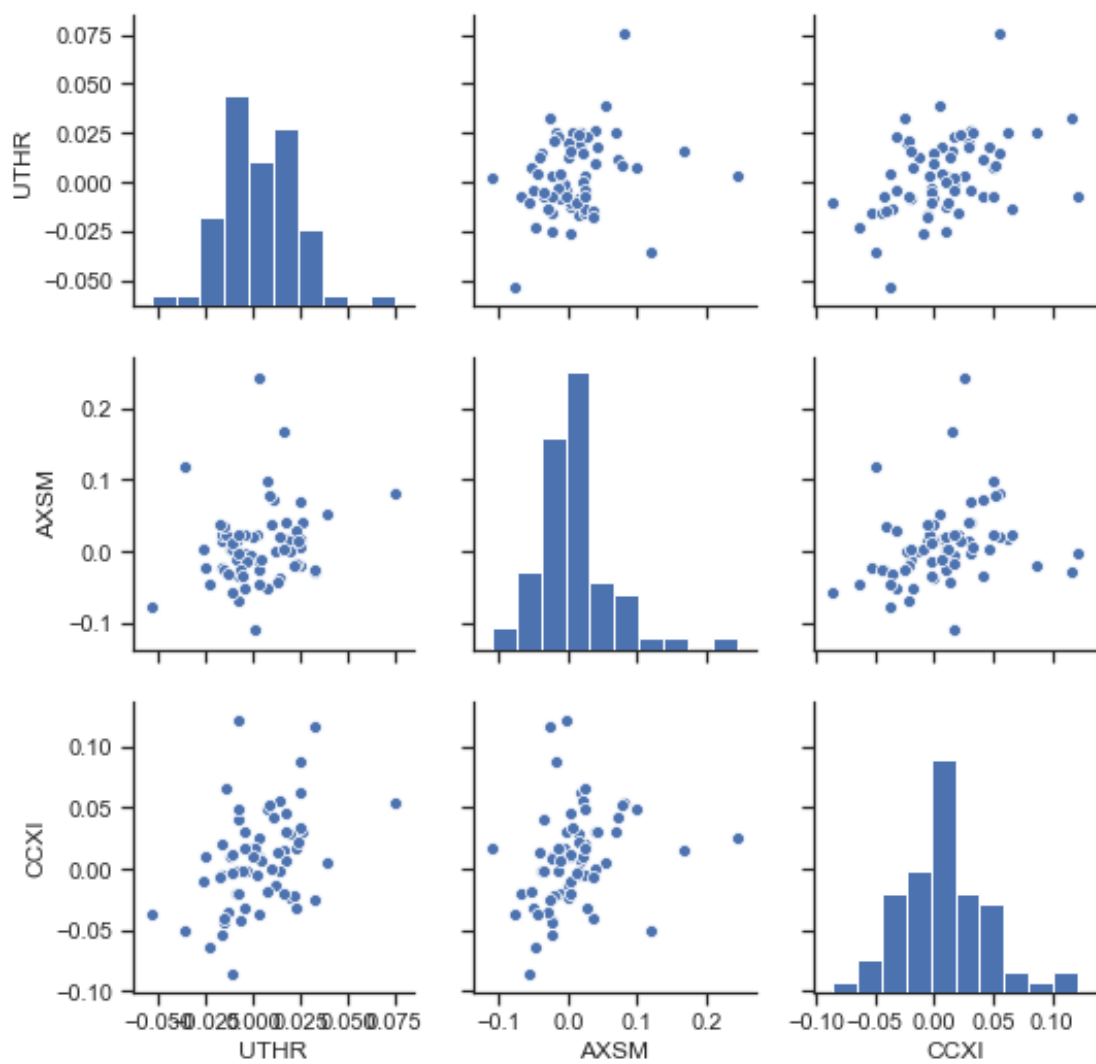
```
[14]: plt.figure(figsize=(12,8))
plt.plot(stock_rets.cumsum())
plt.title(title + ' Stocks Returns Cumulative Sum')
plt.legend(labels=stock_rets.columns)
```

[14]: <matplotlib.legend.Legend at 0x2704edcde48>

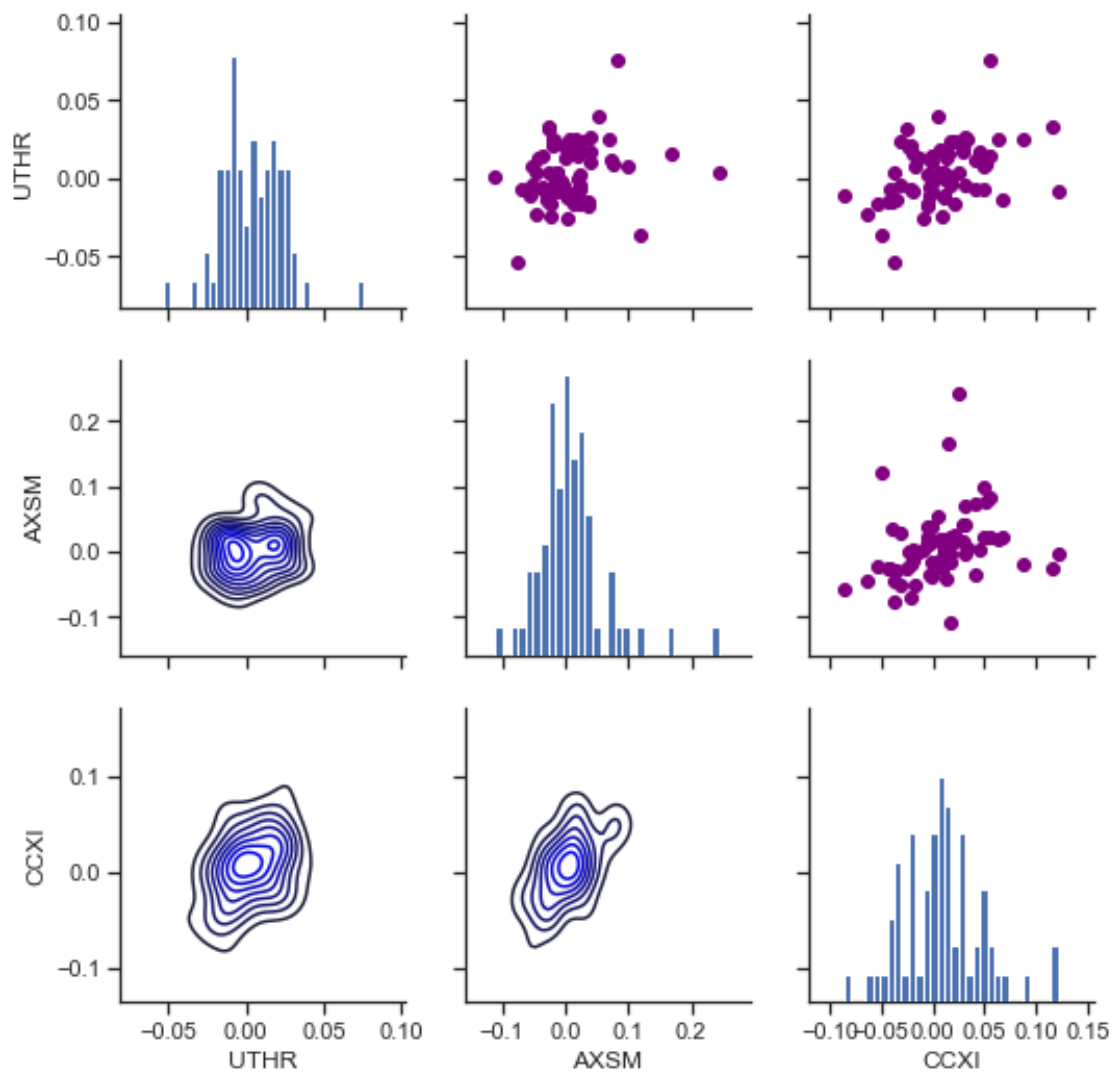


```
[15]: sns.set(style='ticks')
ax = sns.pairplot(stock_rets, diag_kind='hist')

nplot = len(stock_rets.columns)
for i in range(nplot) :
    for j in range(nplot) :
        ax.axes[i, j].locator_params(axis='x', nbins=6, tight=True)
```



```
[16]: ax = sns.PairGrid(stock_rets)
ax.map_upper(plt.scatter, color='purple')
ax.map_lower(sns.kdeplot, color='blue')
ax.map_diag(plt.hist, bins=30)
for i in range(nplot) :
    for j in range(nplot) :
        ax.axes[i, j].locator_params(axis='x', nbins=6, tight=True)
```

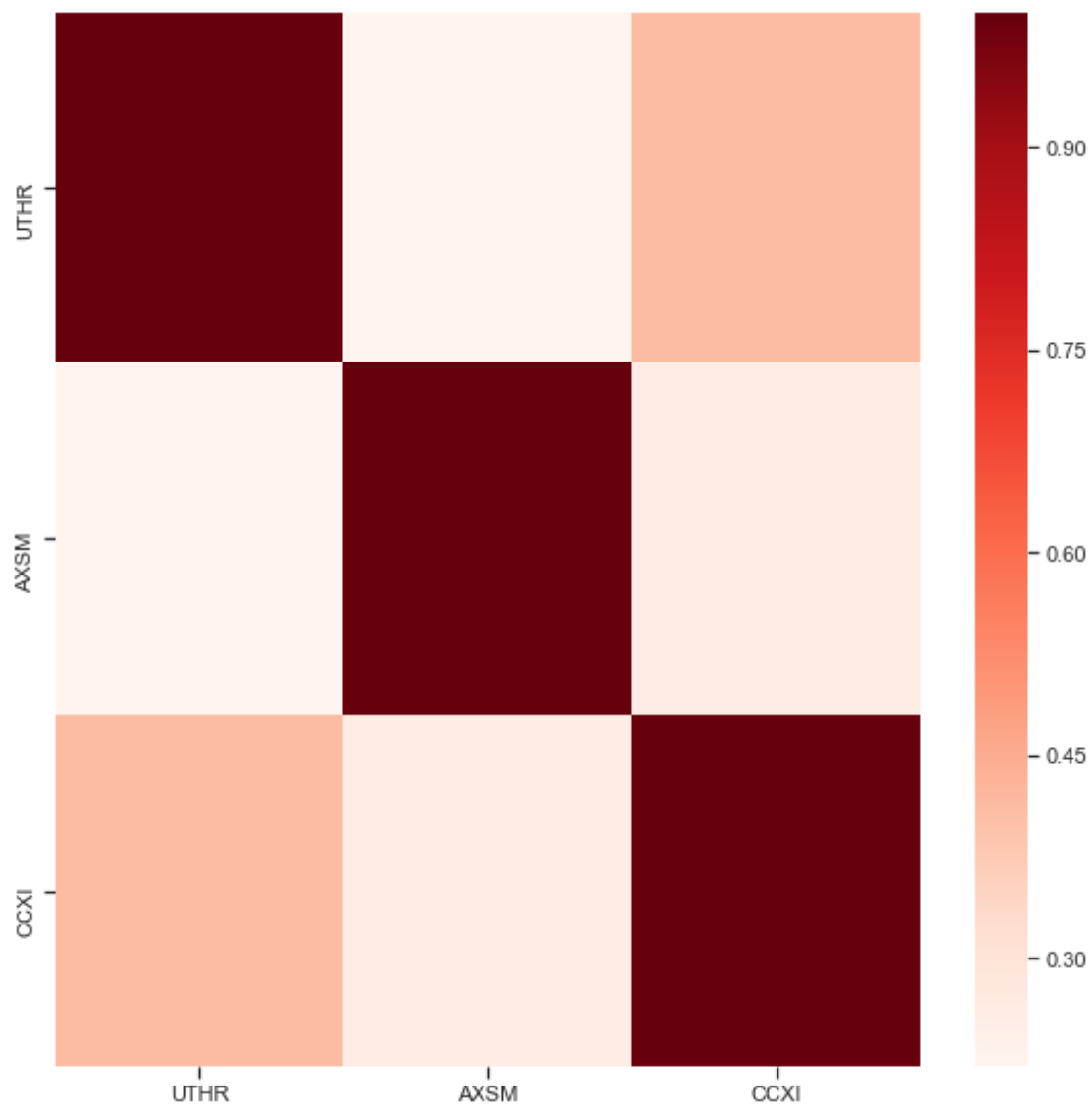


```
[17]: plt.figure(figsize=(10,10))
      corr = stock_rets.corr()

      # plot the heatmap
      sns.heatmap(corr,
                  xticklabels=corr.columns,
                  yticklabels=corr.columns,
                  cmap="Reds")
```

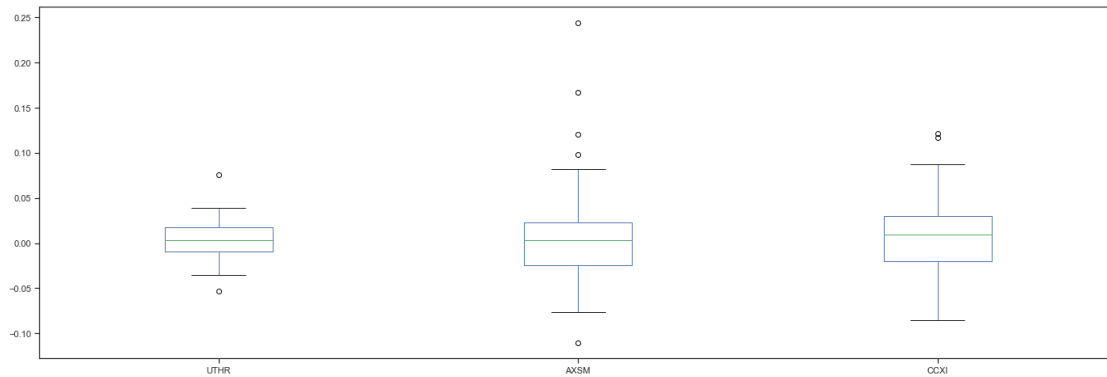
```
[17]: <matplotlib.axes._subplots.AxesSubplot at 0x27050dabeb8>
```





```
[18]: # Box plot  
stock_rets.plot(kind='box',figsize=(24,8))
```

```
[18]: <matplotlib.axes._subplots.AxesSubplot at 0x27050fee898>
```

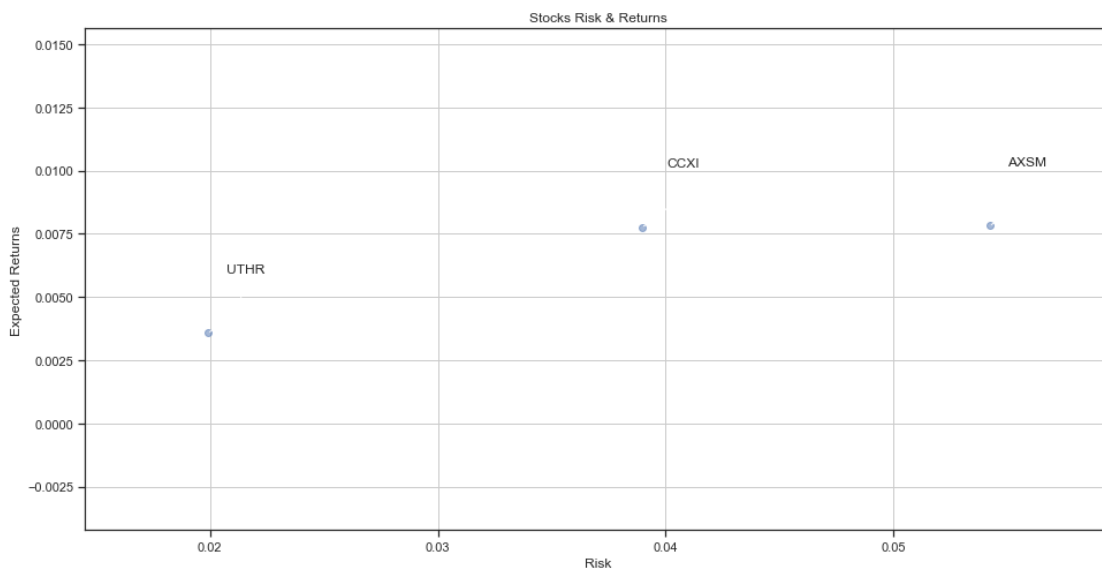


```
[19]: rets = stock_rets.dropna()

plt.figure(figsize=(16,8))
plt.scatter(rets.std(), rets.mean(),alpha = 0.5)

plt.title('Stocks Risk & Returns')
plt.xlabel('Risk')
plt.ylabel('Expected Returns')
plt.grid(which='major')

for label, x, y in zip(rets.columns, rets.std(), rets.mean()):
    plt.annotate(
        label,
        xy = (x, y), xytext = (50, 50),
        textcoords = 'offset points', ha = 'right', va = 'bottom',
        arrowprops = dict(arrowstyle = '-', connectionstyle = 'arc3,rad=-0.3'))
```



```
[20]: rets = stock_rets.dropna()
area = np.pi*20.0

sns.set(style='darkgrid')
plt.figure(figsize=(16,8))
plt.scatter(rets.std(), rets.mean(), s=area)
plt.xlabel("Risk", fontsize=15)
plt.ylabel("Expected Return", fontsize=15)
plt.title("Return vs. Risk for Stocks", fontsize=20)

for label, x, y in zip(rets.columns, rets.std(), rets.mean()) :
    plt.annotate(label, xy=(x,y), xytext=(50, 0), textcoords='offset points',
        arrowprops=dict(arrowstyle='-',
        ↪connectionstyle='bar,angle=180,fraction=-0.2'),
        bbox=dict(boxstyle="round", fc="w"))
```



```
[21]: def annual_risk_return(stock_rets):
    tradeoff = stock_rets.agg(["mean", "std"]).T
    tradeoff.columns = ["Return", "Risk"]
    tradeoff.Return = tradeoff.Return*252
    tradeoff.Risk = tradeoff.Risk * np.sqrt(252)
    return tradeoff
```

```
[22]: tradeoff = annual_risk_return(stock_rets)
tradeoff
```

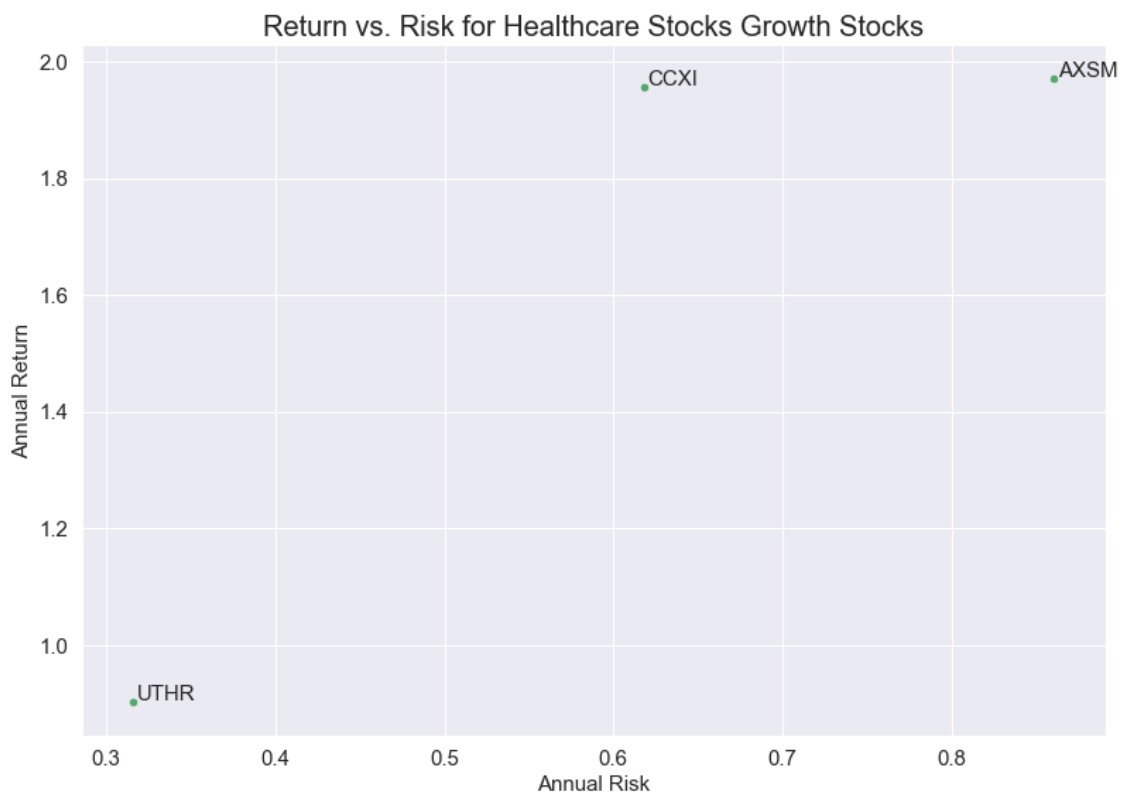
```
[22]:
```

	Return	Risk
UTHR	0.902482	0.315917
AXSM	1.971030	0.860686
CCXI	1.956847	0.618366

```
[23]: import itertools

colors = itertools.cycle(["r", "b", "g"])

tradeoff.plot(x = "Risk", y = "Return", kind = "scatter", figsize = (13,9), s = 20,
              fontsize = 15, c='g')
for i in tradeoff.index:
    plt.annotate(i, xy=(tradeoff.loc[i, "Risk"]+0.002, tradeoff.loc[i, "Return"]+0.002), size = 15)
plt.xlabel("Annual Risk", fontsize = 15)
plt.ylabel("Annual Return", fontsize = 15)
plt.title("Return vs. Risk for " + title + " Stocks", fontsize = 20)
plt.show()
```



```
[24]: rest_rets = rets.corr()
pair_value = rest_rets.abs().unstack()
pair_value.sort_values(ascending = False)
```

```
[24]: CCXI  CCXI    1.000000
      AXSM  AXSM    1.000000
      UTHR  UTHR    1.000000
      CCXI  UTHR    0.414333
      UTHR  CCXI    0.414333
      CCXI  AXSM    0.260518
      AXSM  CCXI    0.260518
           UTHR    0.219479
      UTHR  AXSM    0.219479
      dtype: float64
```

```
[25]: # Normalized Returns Data
      Normalized_Value = ((rets[:] - rets[:].min()) / (rets[:].max() - rets[:].min()))
      Normalized_Value.head()
```

```
[25]:          UTHR      AXSM      CCXI
Date
2020-04-02  0.383490  0.304462  0.560852
2020-04-03  0.348547  0.278518  0.318051
2020-04-06  0.666565  0.235450  0.977783
2020-04-07  0.357360  0.116586  0.314107
2020-04-08  0.538855  0.783739  0.486838
```

```
[26]: Normalized_Value.corr()
```

```
[26]:          UTHR      AXSM      CCXI
UTHR  1.000000  0.219479  0.414333
AXSM  0.219479  1.000000  0.260518
CCXI  0.414333  0.260518  1.000000
```

```
[27]: normalized_rets = Normalized_Value.corr()
      normalized_pair_value = normalized_rets.abs().unstack()
      normalized_pair_value.sort_values(ascending = False)
```

```
[27]: CCXI  CCXI    1.000000
      AXSM  AXSM    1.000000
      UTHR  UTHR    1.000000
      CCXI  UTHR    0.414333
      UTHR  CCXI    0.414333
      CCXI  AXSM    0.260518
      AXSM  CCXI    0.260518
           UTHR    0.219479
      UTHR  AXSM    0.219479
      dtype: float64
```

```
[28]: print("Stock returns: ")
      print(rets.mean())
```

```
print('-' * 50)
print("Stock risks:")
print(rets.std())
```

```
Stock returns:
UTHR    0.003581
AXSM    0.007822
CCXI    0.007765
dtype: float64
```

```
-----
Stock risks:
UTHR    0.019901
AXSM    0.054218
CCXI    0.038953
dtype: float64
```

```
[29]: table = pd.DataFrame()
      table['Returns'] = rets.mean()
      table['Risk'] = rets.std()
      table.sort_values(by='Returns')
```

```
[29]:      Returns      Risk
      UTHR  0.003581  0.019901
      CCXI  0.007765  0.038953
      AXSM  0.007822  0.054218
```

```
[30]: table.sort_values(by='Risk')
```

```
[30]:      Returns      Risk
      UTHR  0.003581  0.019901
      CCXI  0.007765  0.038953
      AXSM  0.007822  0.054218
```

```
[31]: rf = 0.01
      table['Sharpe Ratio'] = (table['Returns'] - rf) / table['Risk']
      table
```

```
[31]:      Returns      Risk  Sharpe Ratio
      UTHR  0.003581  0.019901    -0.322534
      AXSM  0.007822  0.054218    -0.040179
      CCXI  0.007765  0.038953    -0.057369
```

```
[32]: table['Max Returns'] = rets.max()
```

```
[33]: table['Min Returns'] = rets.min()
```

```
[34]: table['Median Returns'] = rets.median()
```

```
[35]: total_return = stock_rets[-1:].transpose()
      table['Total Return'] = 100 * total_return
      table
```

```
[35]:      Returns      Risk  Sharpe Ratio  Max Returns  Min Returns  \
UTHR  0.003581  0.019901   -0.322534    0.075563   -0.053394
AXSM  0.007822  0.054218   -0.040179    0.243455   -0.109920
CCXI  0.007765  0.038953   -0.057369    0.121123   -0.085633

      Median Returns  Total Return
UTHR           0.003095   -0.691399
AXSM           0.003098    2.336670
CCXI           0.009041    4.914965
```

```
[36]: table['Average Return Days'] = (1 + total_return)**(1 / days) - 1
      table
```

```
[36]:      Returns      Risk  Sharpe Ratio  Max Returns  Min Returns  \
UTHR  0.003581  0.019901   -0.322534    0.075563   -0.053394
AXSM  0.007822  0.054218   -0.040179    0.243455   -0.109920
CCXI  0.007765  0.038953   -0.057369    0.121123   -0.085633

      Median Returns  Total Return  Average Return Days
UTHR           0.003095   -0.691399           -0.000071
AXSM           0.003098    2.336670            0.000236
CCXI           0.009041    4.914965            0.000490
```

```
[37]: initial_value = df.iloc[0]
      ending_value = df.iloc[-1]
      table['CAGR'] = ((ending_value / initial_value) ** (252.0 / days)) - 1
      table
```

```
[37]:      Returns      Risk  Sharpe Ratio  Max Returns  Min Returns  \
UTHR  0.003581  0.019901   -0.322534    0.075563   -0.053394
AXSM  0.007822  0.054218   -0.040179    0.243455   -0.109920
CCXI  0.007765  0.038953   -0.057369    0.121123   -0.085633

      Median Returns  Total Return  Average Return Days      CAGR
UTHR           0.003095   -0.691399           -0.000071  0.790787
AXSM           0.003098    2.336670            0.000236  2.028307
CCXI           0.009041    4.914965            0.000490  2.343430
```

```
[38]: table.sort_values(by='Average Return Days')
```

```
[38]:      Returns      Risk  Sharpe Ratio  Max Returns  Min Returns  \
UTHR  0.003581  0.019901   -0.322534    0.075563   -0.053394
AXSM  0.007822  0.054218   -0.040179    0.243455   -0.109920
```

CCXI	0.007765	0.038953	-0.057369	0.121123	-0.085633
------	----------	----------	-----------	----------	-----------

	Median Returns	Total Return	Average Return Days	CAGR
UTHR	0.003095	-0.691399	-0.000071	0.790787
AXSM	0.003098	2.336670	0.000236	2.028307
CCXI	0.009041	4.914965	0.000490	2.343430