In [55]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [193]:

```
df = pd.read_excel(r"C:\Users\mdmoh\Downloads\stock analysis.xlsx",sheet_name = "Sheet2",parse_dates=True, )
df.head()
```

Out[193]:

	Date	AAPL	SAVA	CEI	SPCE	CAN
0	2021-01-01	133.520004	6.840000	0.97	23.959999	261.000000
1	2021-02-01	133.750000	20.500000	1.43	47.169998	246.100006
2	2021-03-01	123.750000	51.049999	1.52	38.930000	254.000000
3	2021-04-01	123.660004	46.040001	1.02	31.290001	278.619995
4	2021-05-01	132.039993	46.889999	0.73	21.719999	292.929993

In [194]:

```
df.describe()
```

Out[194]:

	AAPL	SAVA	CEI	SPCE	CAN
count	8.000000	8.000000	8.000000	8.000000	8.000000
mean	131.993752	54.463750	0.933850	33.702500	278.713749
std	8.051985	36.715324	0.376426	9.470574	23.630580
min	123.660004	6.840000	0.490800	21.719999	246.100006
25%	124.747501	39.655001	0.657500	28.527499	259.250000
50%	132.779999	48.969999	0.850000	30.895001	282.099991
75%	134.462502	62.117500	1.122500	40.697500	293.567498
max	147.550003	124.500000	1.520000	47.169998	316.000000

In [195]:

```
heatmap = df.corr()
heatmap
```

Out[195]:

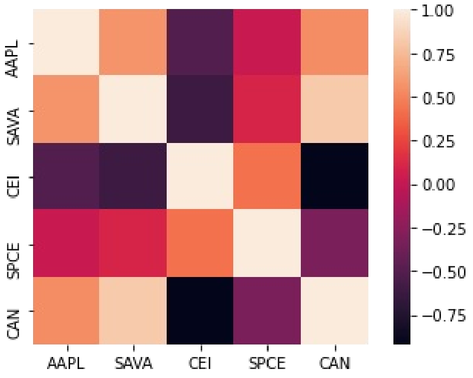
	AAPL	SAVA	CEI	SPCE	CAN
AAPL	1.000000	0.579768	-0.510164	0.032444	0.548574
SAVA	0.579768	1.000000	-0.612567	0.106425	0.833329
CEI	-0.510164	-0.612567	1.000000	0.429439	-0.919821
SPCE	0.032444	0.106425	0.429439	1.000000	-0.321892
CAN	0.548574	0.833329	-0.919821	-0.321892	1.000000

In [196]:

```
sns.heatmap(heatmap, square=True)
```

Out[196]:

```
<AxesSubplot:>
```



In [197]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8 entries, 0 to 7
Data columns (total 6 columns):
 #   Column  Non-Null Count  Dtype         
---  ------  --------------  -----         
 0   Date    8 non-null      datetime64[ns]
 1   AAPL    8 non-null      float64       
 2   SAVA    8 non-null      float64       
 3   CEI     8 non-null      float64       
 4   SPCE    8 non-null      float64       
 5   CAN     8 non-null      float64       
dtypes: datetime64[ns](1), float64(5)
memory usage: 512.0 bytes
```

In [198]:

```
df['AAPL'].std()
```

Out[198]:

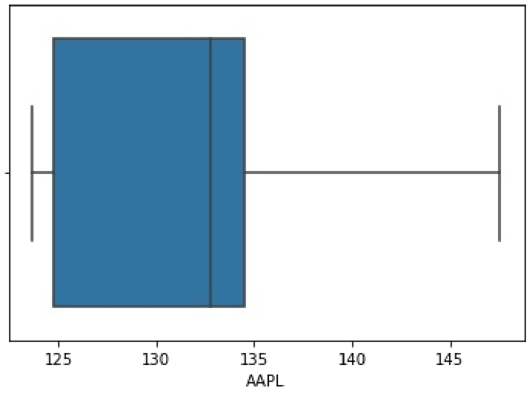
```
8.051985144317007
```

In [199]:

```
sns.boxplot(x='AAPL', data=df)
```

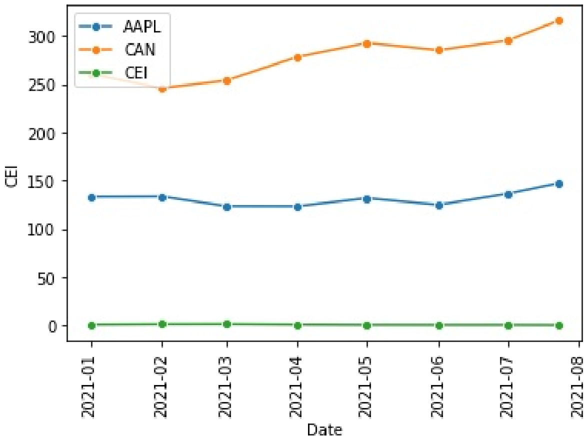
Out[199]:

```
<AxesSubplot:xlabel='AAPL'>
```



In [200]:

```
sns.lineplot(x='Date',y='AAPL',data=df, marker='o',label='AAPL')
sns.lineplot(x='Date',y='CAN',data=df, marker='o',label='CAN')
sns.lineplot(x='Date',y='CEI',data=df, marker='o',label='CEI')
plt.xticks(rotation='vertical')
plt.legend()
plt.show()
```

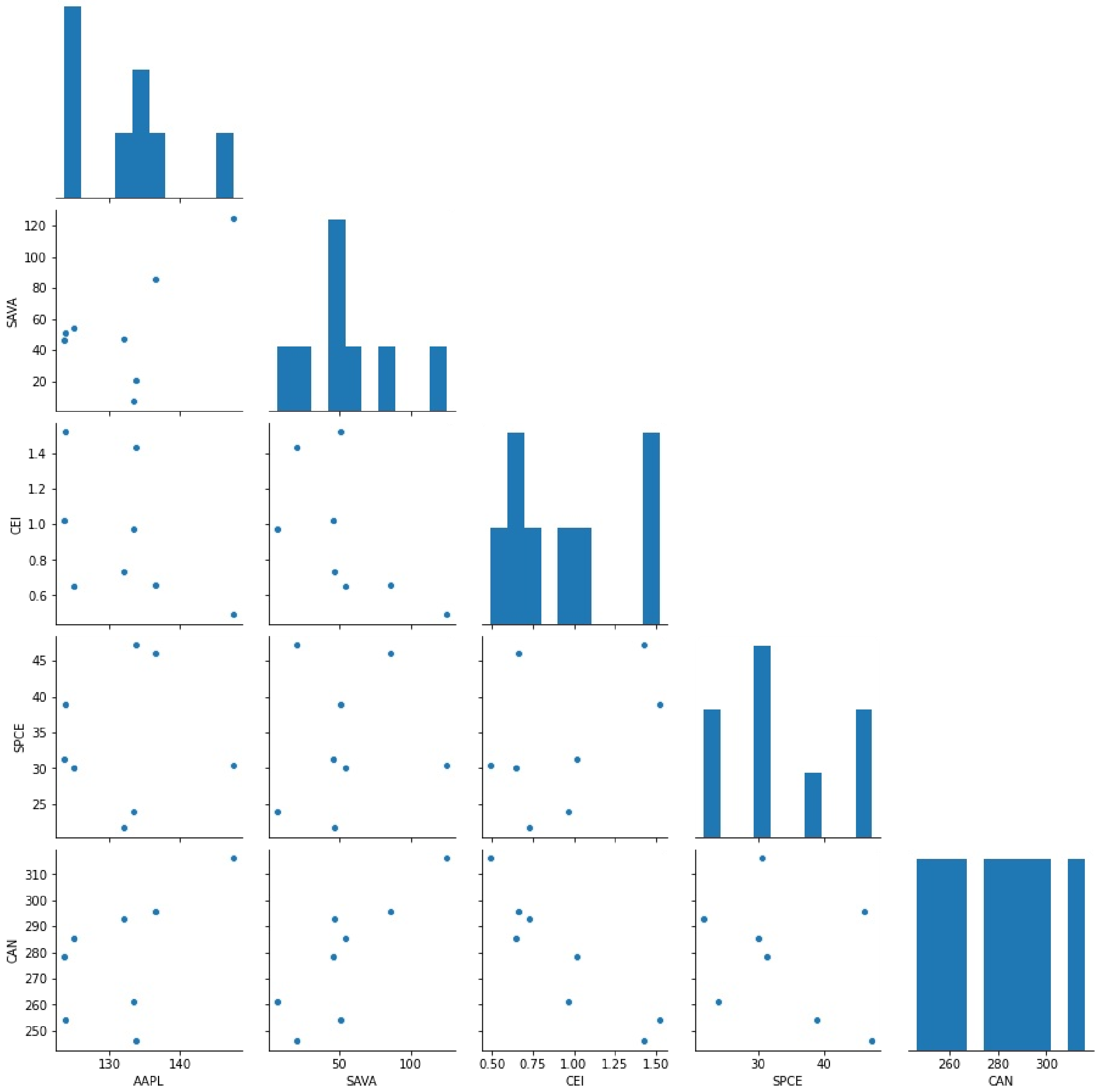


In [201]:

```
sns.pairplot(df,corner=True)
```

Out[201]:

```
<seaborn.axisgrid.PairGrid at 0x2138e540808>
```

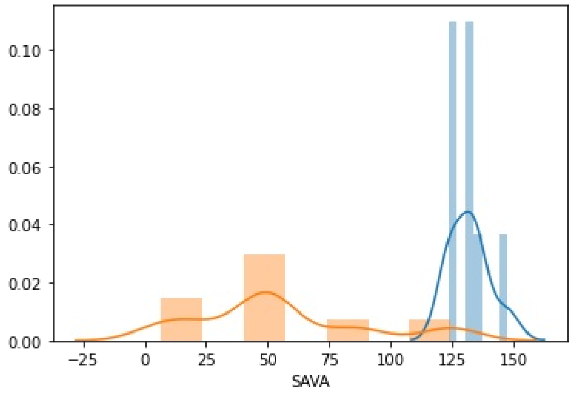


In [202]:

```
sns.distplot(df['AAPL'], bins=7)
sns.distplot(df['SAVA'], bins=7)
```

Out[202]:

```
<AxesSubplot:xlabel='SAVA'>
```

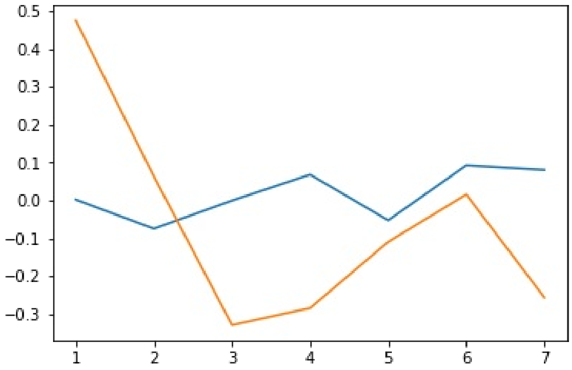


In [203]:

```
df['AAPL'].pct_change().plot()
df['CEI'].pct_change().plot()
```

Out[203]:

```
<AxesSubplot:>
```



In [204]:

```
#you have $1000, you distribute evenly across all 5 stocks, whats your return by the end of the year
```

In [205]:

```
data1= df[df.columns[-5:]]
data1
```

Out[205]:

	AAPL	SAVA	CEI	SPCE	CAN
0	133.520004	6.840000	0.9700	23.959999	261.000000
1	133.750000	20.500000	1.4300	47.169998	246.100006
2	123.750000	51.049999	1.5200	38.930000	254.000000
3	123.660004	46.040001	1.0200	31.290001	278.619995
4	132.039993	46.889999	0.7300	21.719999	292.929993
5	125.080002	54.290001	0.6500	30.049999	285.579987
6	136.600006	85.599998	0.6600	46.000000	295.480011
7	147.550003	124.500000	0.4908	30.500000	316.000000

In [206]:

```
for x in data1:
      print((data1[x].iloc[-1]- data1[x].iloc[0])/data1[x].iloc[0])
```

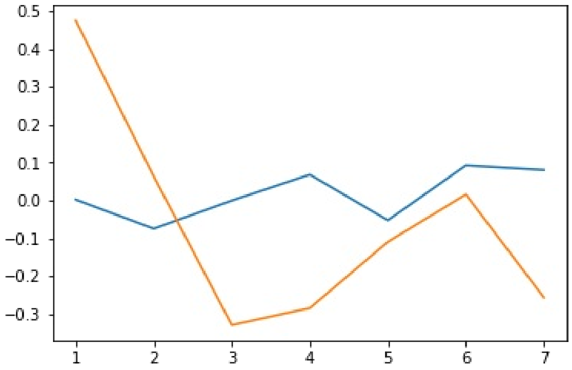
```
0.1050778803152223
17.20175438596491
-0.494020618556701
0.2729549780031293
0.210727969348659
```

In [234]:

```
data1['AAPL'].pct_change().plot()
data1['CEI'].pct_change().plot()
```

Out[234]:

```
<AxesSubplot:>
```



In [222]:

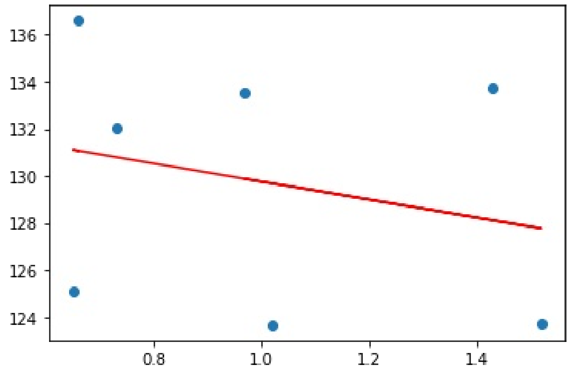
```
from sklearn.linear_model import LinearRegression
```

In [232]:

```
X = df['CEI'].iloc[:7].values.reshape(-1, 1)
Y = df['AAPL'].iloc[:7].values.reshape(-1, 1)
linear_regressor = LinearRegression()
linear_regressor.fit(X, Y)
Y_pred = linear_regressor.predict(X)
```

In [230]:

```
plt.scatter(X, Y)
plt.plot(X, Y_pred, color='red')
plt.show()
```



In [ ]: