In [181…

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
sns.set_style('darkgrid')
```

In [182…

```
df = pd.read_csv('assets_price.csv')
df.head()
```

Out[182…

	Date	S&P500	Gold	US Gov Bond	USD/JPY
0	2019-01-02	2511.00	1281.000000	114.687500	109.667998
1	2019-01-03	2447.75	1291.800049	115.304688	107.441000
2	2019-01-04	2531.25	1282.699951	114.757812	107.808002
3	2019-01-07	2550.50	1286.800049	114.585938	108.521998
4	2019-01-08	2572.50	1283.199951	114.335938	108.616000

In [183…

```
df.dtypes
```

Out[183…

```
Date             object
S&P500          float64
Gold            float64
US Gov Bond     float64
USD/JPY         float64
dtype: object
```

In [184…

```
df['Date'] = pd.to_datetime(df['Date'])
df.set_index('Date',inplace=True)
df.head()
```

Out[184…

	S&P500	Gold	US Gov Bond	USD/JPY
Date				
2019-01-02	2511.00	1281.000000	114.687500	109.667998
2019-01-03	2447.75	1291.800049	115.304688	107.441000
2019-01-04	2531.25	1282.699951	114.757812	107.808002
2019-01-07	2550.50	1286.800049	114.585938	108.521998
2019-01-08	2572.50	1283.199951	114.335938	108.616000

In [185…

```
df.sort_index()
```

Out[185…

	S&P500	Gold	US Gov Bond	USD/JPY
Date				
2019-01-02	2511.00	1281.000000	114.687500	109.667998
2019-01-03	2447.75	1291.800049	115.304688	107.441000
2019-01-04	2531.25	1282.699951	114.757812	107.808002
2019-01-07	2550.50	1286.800049	114.585938	108.521998
2019-01-08	2572.50	1283.199951	114.335938	108.616000
...	...	...	...	...
2021-05-11	4146.25	1835.900024	124.257812	108.714997
2021-05-12	4058.75	1822.599976	123.968750	109.664000
2021-05-13	4107.00	1823.800049	124.109375	109.515998
2021-05-17	4157.75	1867.500000	124.140625	109.242002
2021-05-18	4123.00	1867.800049	124.187500	108.867993

531 rows × 4 columns

In [186…

```
for i in df.columns:
    df[f'norm_{i}'] = df[i] / df[i].iloc[0] * 100
```

In [187…

```
df
```

Out[187…

	S&P500	Gold	US Gov Bond	USD/JPY	norm_S&P500	norm_Gold	norm_US Gov Bond	norm_USD/JPY
Date								
2019-01-02	2511.00	1281.000000	114.687500	109.667998	100.000000	100.000000	100.000000	100.000000
2019-01-03	2447.75	1291.800049	115.304688	107.441000	97.481083	100.843095	100.538147	97.969327
2019-01-04	2531.25	1282.699951	114.757812	107.808002	100.806452	100.132705	100.061308	98.303975
2019-01-07	2550.50	1286.800049	114.585938	108.521998	101.573078	100.452775	99.911444	98.955028
2019-01-08	2572.50	1283.199951	114.335938	108.616000	102.449223	100.171737	99.693460	99.040743
...	...	...	...	...	...	...	...	...
2021-05-11	4146.25	1835.900024	124.257812	108.714997	165.123457	143.317722	108.344687	99.131012
2021-05-12	4058.75	1822.599976	123.968750	109.664000	161.638789	142.279467	108.092643	99.996354
2021-05-13	4107.00	1823.800049	124.109375	109.515998	163.560335	142.373150	108.215259	99.861399
2021-05-17	4157.75	1867.500000	124.140625	109.242002	165.581442	145.784543	108.242507	99.611558
2021-05-18	4123.00	1867.800049	124.187500	108.867993	164.197531	145.807966	108.283379	99.270521

531 rows × 8 columns

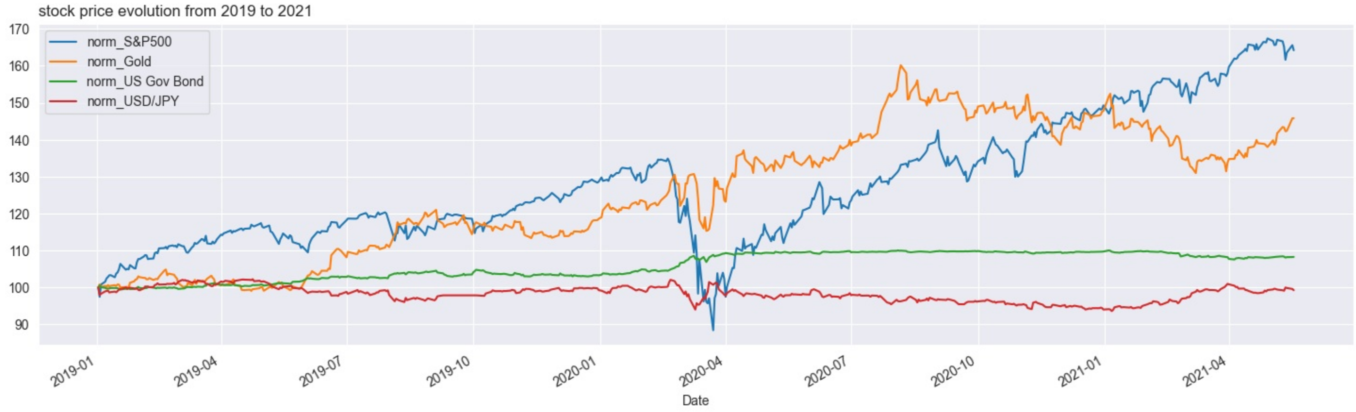
visualisation

In [188…

```
df.iloc[:,4:].plot(kind='line',figsize=(18,5))
plt.title("stock price evolution from 2019 to 2021",loc='left')
```

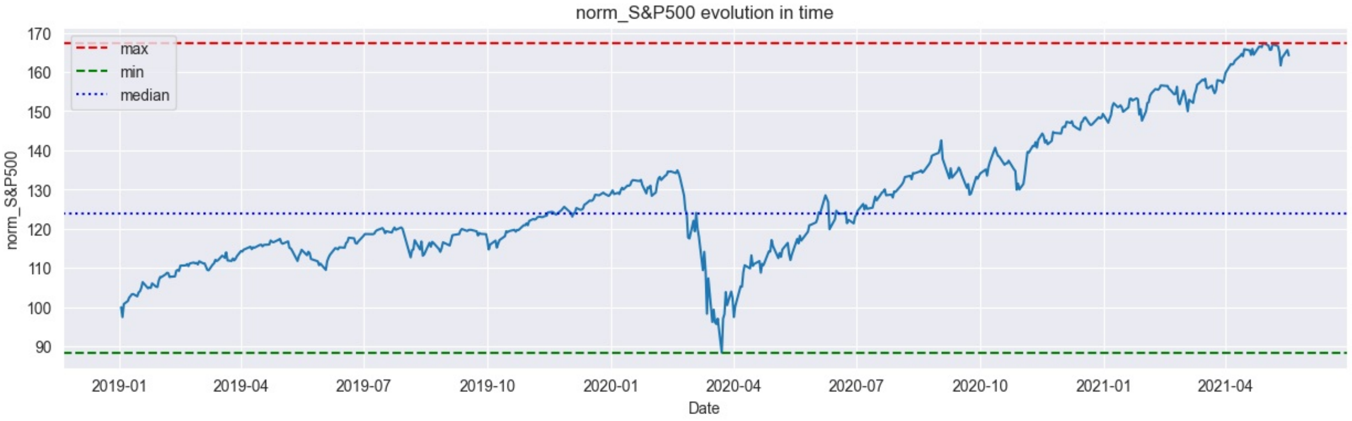
Out[188…

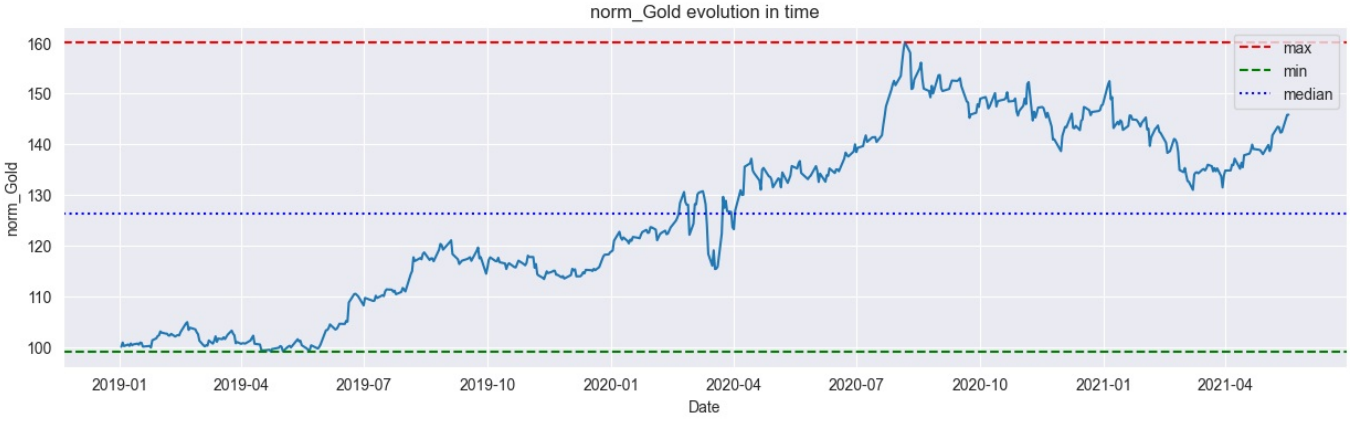
```
Text(0.0, 1.0, 'stock price evolution from 2019 to 2021')
```

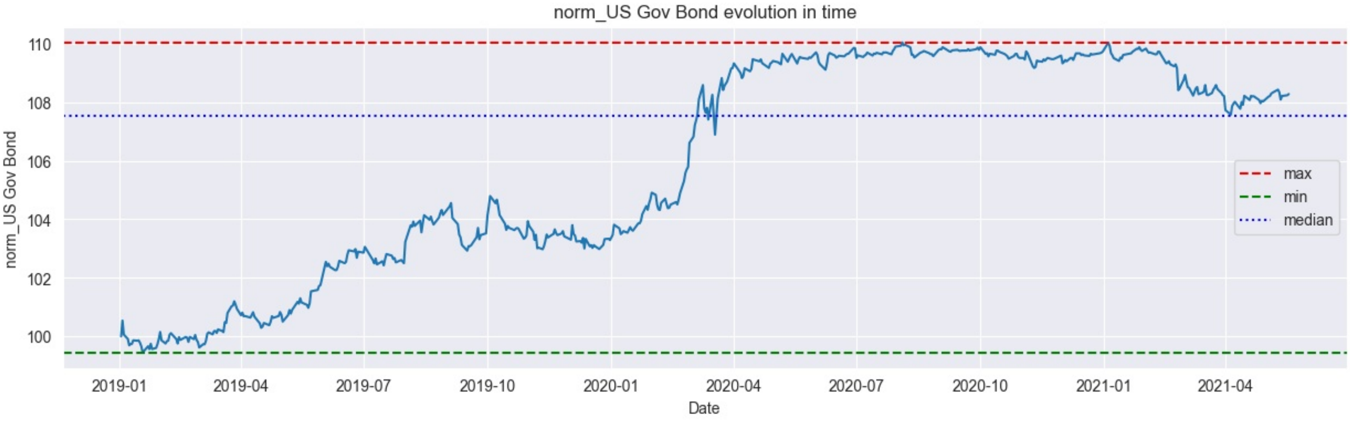


In [273…

```
for i in df.iloc[:,4:7] :
    plt.figure(figsize=(15,4))
    sns.lineplot(data=df,x=df.index,y=i)
    plt.axhline(df[i].max(),c="red",label="max",linestyle='--')
    plt.axhline(df[i].min(),c="green",label="min",linestyle='--')
    plt.axhline(df[i].median(),c="blue",label="median",linestyle=':')
    plt.title(f"{i} evolution in time")
    plt.legend()
    plt.show()
```







In [190…

```
len(df.iloc[:,:4].columns)
```

Out[190…

```
4
```

regression

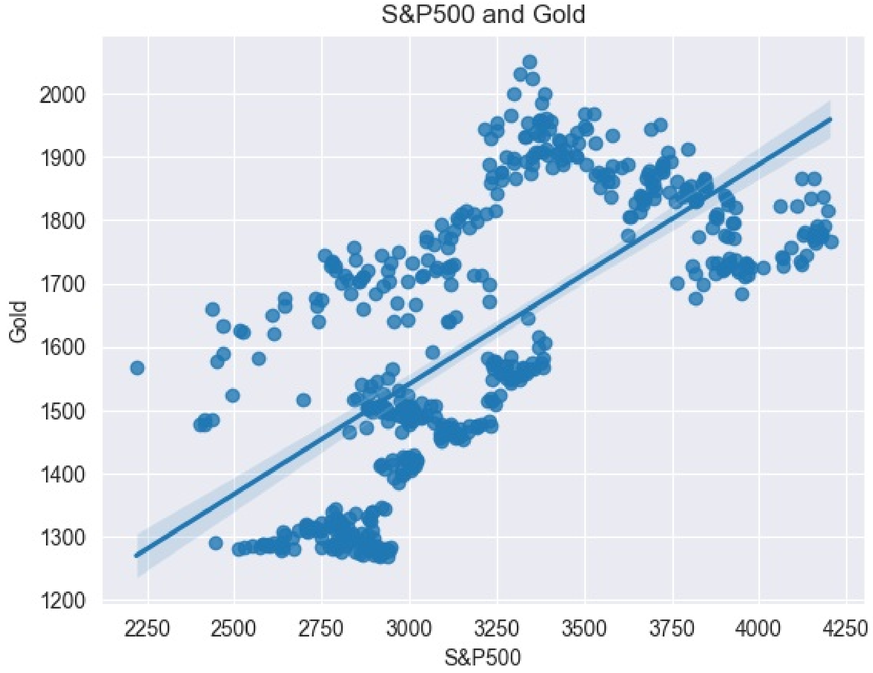
In [ ]:

In [191…

```
from IPython.display import Markdown,display

lr = LinearRegression()

for i in range(len(df.iloc[:,0:3].columns)):
    for j in range(len(df.iloc[:,:4].columns)):
        if i!=j :
            x_col=df.columns[i]
            y_col=df.columns[j]

            sns.regplot(df,x=x_col,y=y_col)
            plt.title(f"{x_col} and {y_col}")
            X = df[[x_col]]
            Y = df[y_col]
            lr.fit( X,Y)
            yprid = lr.predict(X)
            residual = abs(Y-yprid)
            sum_residual = residual.sum()

            markdown_text = f"## {df.columns[i]} and {df.columns[j]}\n"
            markdown_text += f"- The slope: {lr.coef_[0]:.4f}\n"
            markdown_text += f"- The intercept: {lr.intercept_:.4f}\n"
            markdown_text += f"- R-squared: {r2_score(Y, yprid) * 100:.2f}%\n"
            markdown_text += f"- Total sum of residuals: {sum_residual:.4f}\n"



            display(Markdown(markdown_text))
            plt.show()
```

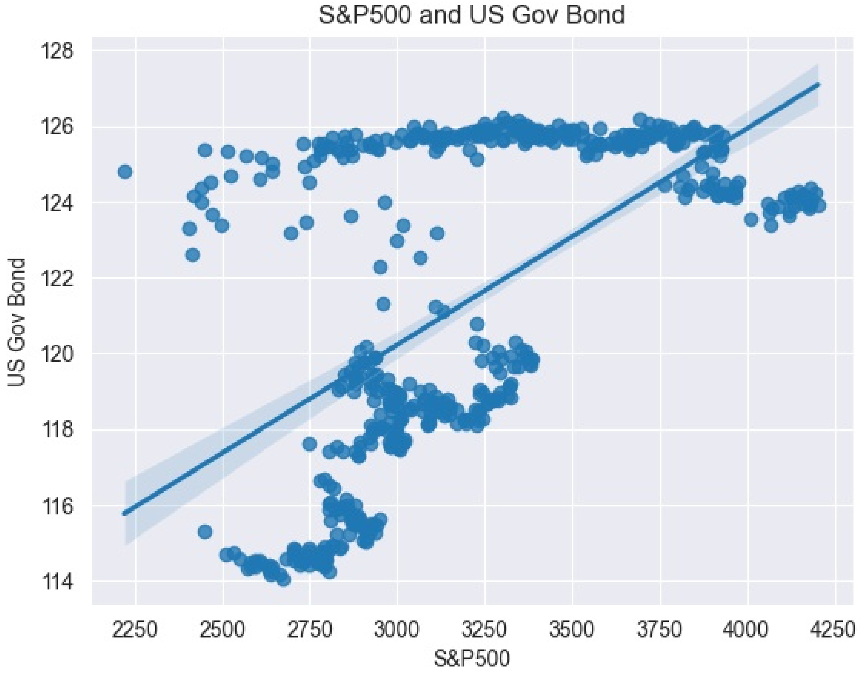
S&P500 and Gold

- The slope: 0.3476
- The intercept: 498.2425
- R-squared: 46.31%
- Total sum of residuals: 74969.6637



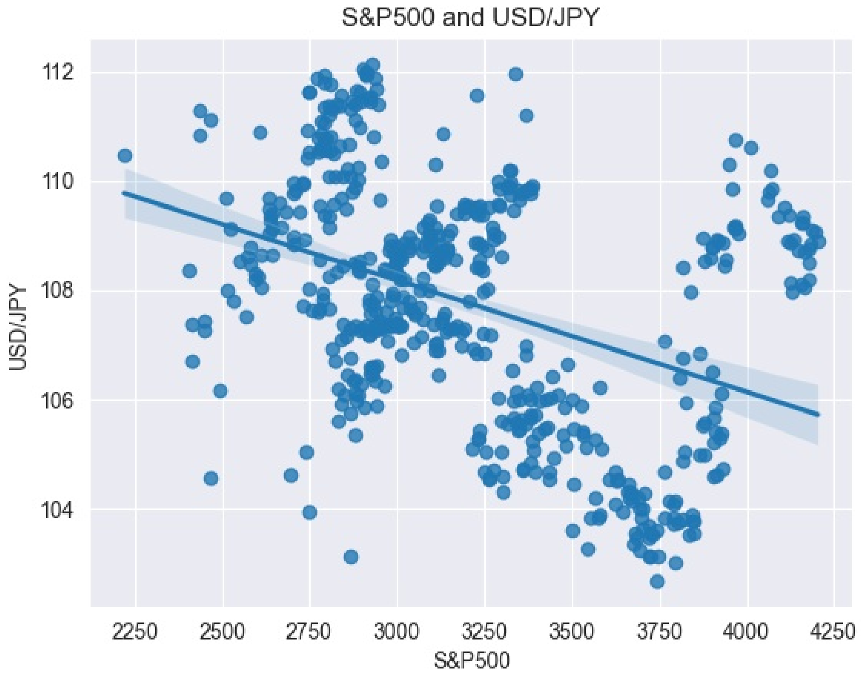
S&P500 and US Gov Bond

- The slope: 0.0057
- The intercept: 103.1062
- R-squared: 34.43%
- Total sum of residuals: 1560.0327



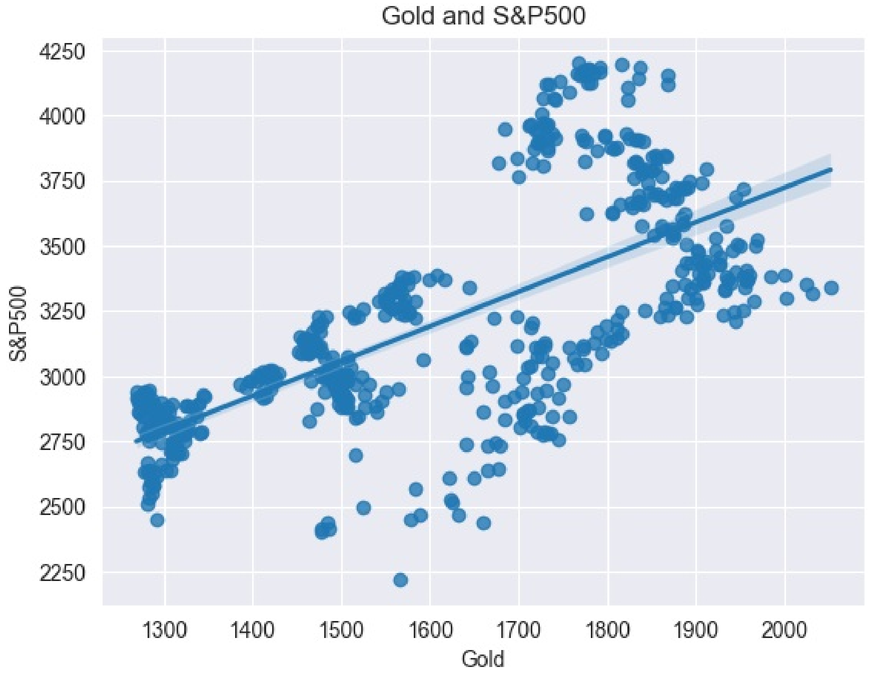
S&P500 and USD/JPY

- The slope: -0.0020
- The intercept: 114.3089
- R-squared: 15.65%
- Total sum of residuals: 917.8194



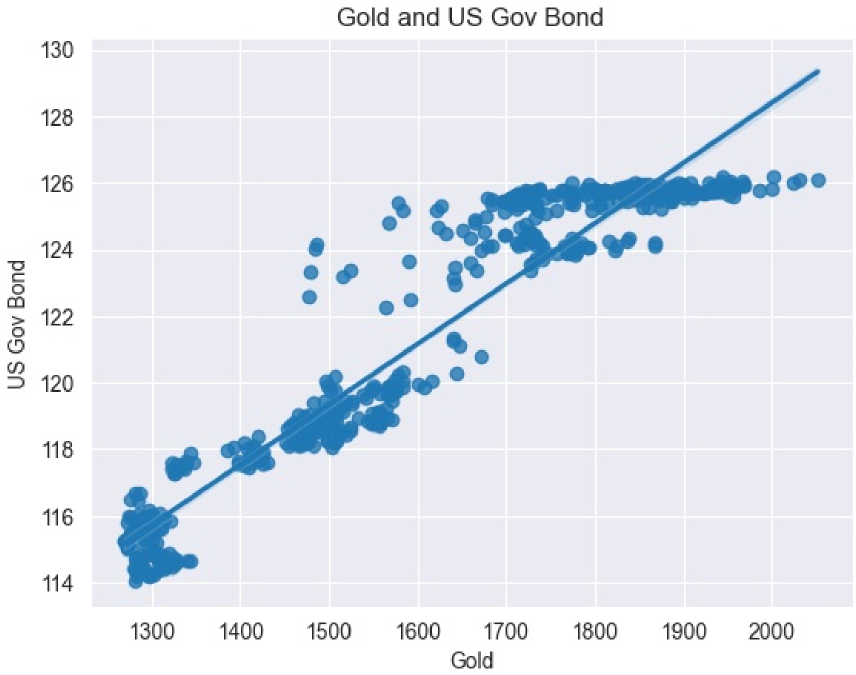
Gold and S&P500

- The slope: 1.3325
- The intercept: 1058.9031
- R-squared: 46.31%
- Total sum of residuals: 128096.2500



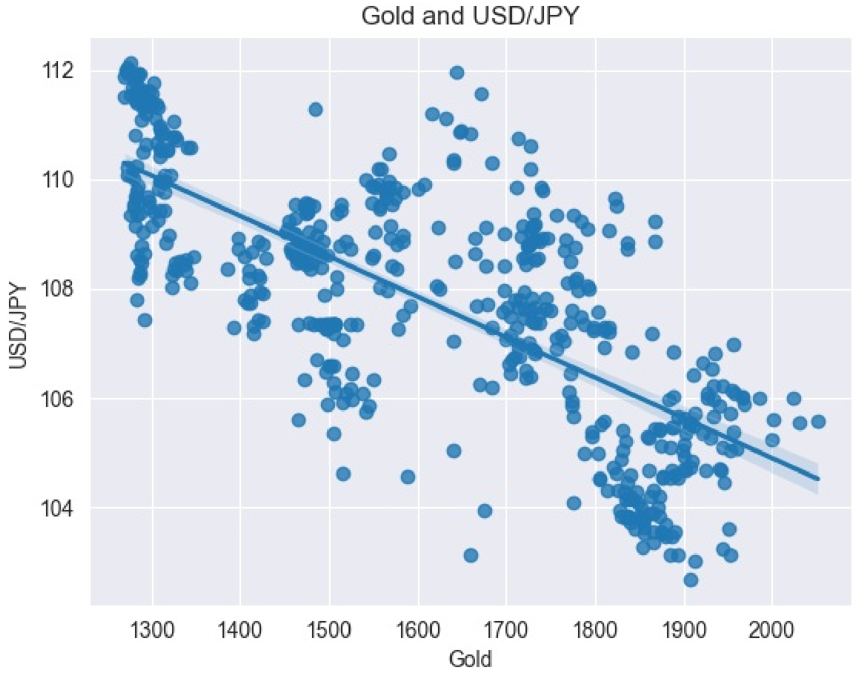
Gold and US Gov Bond

- The slope: 0.0181
- The intercept: 92.1841
- R-squared: 90.54%
- Total sum of residuals: 522.3351



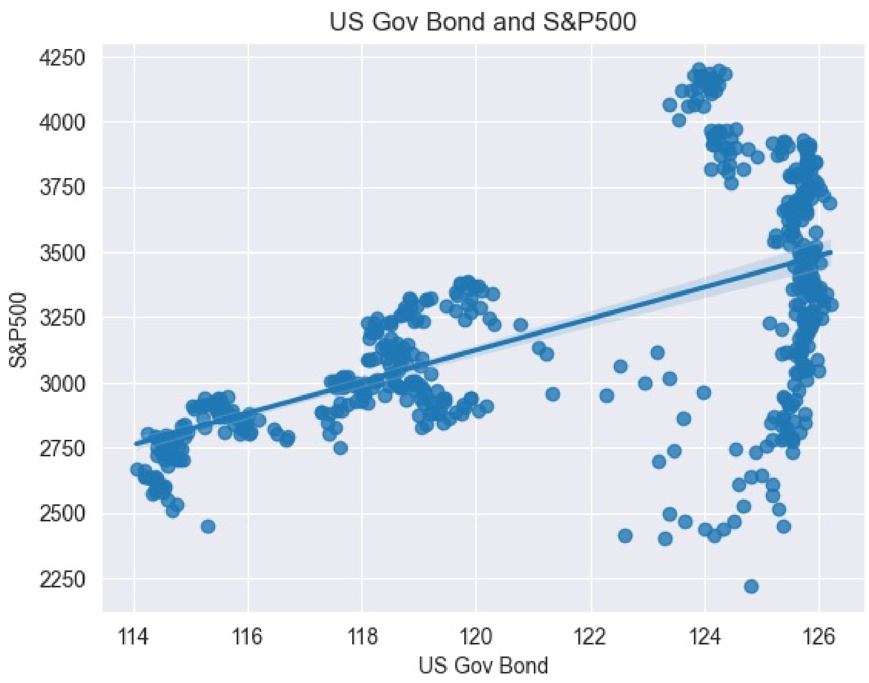
Gold and USD/JPY

- The slope: -0.0074
- The intercept: 119.6688
- R-squared: 53.32%
- Total sum of residuals: 659.1611



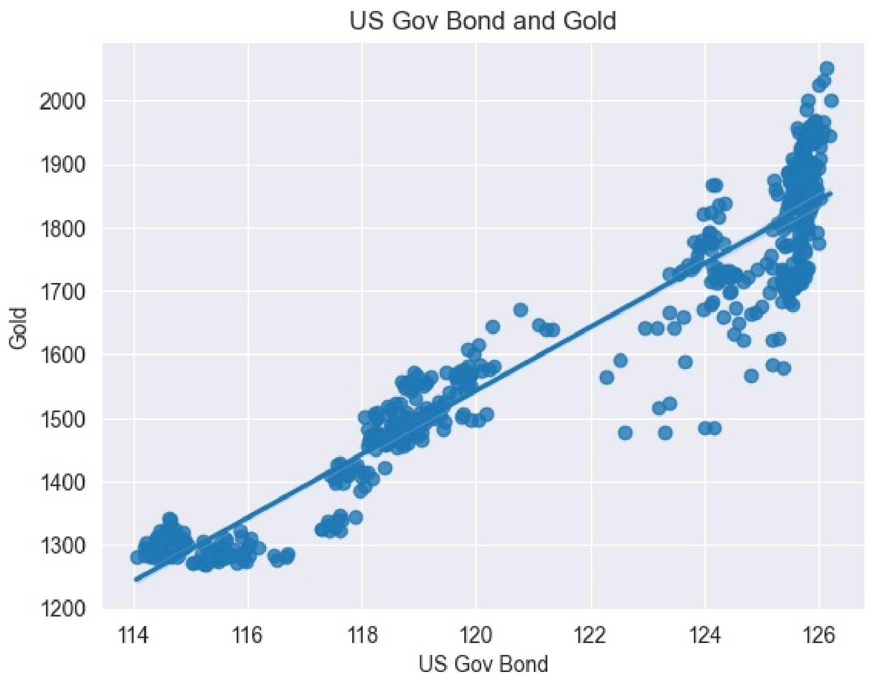
US Gov Bond and S&P500

- The slope: 60.3474
- The intercept: -4117.9392
- R-squared: 34.43%
- Total sum of residuals: 134255.6667



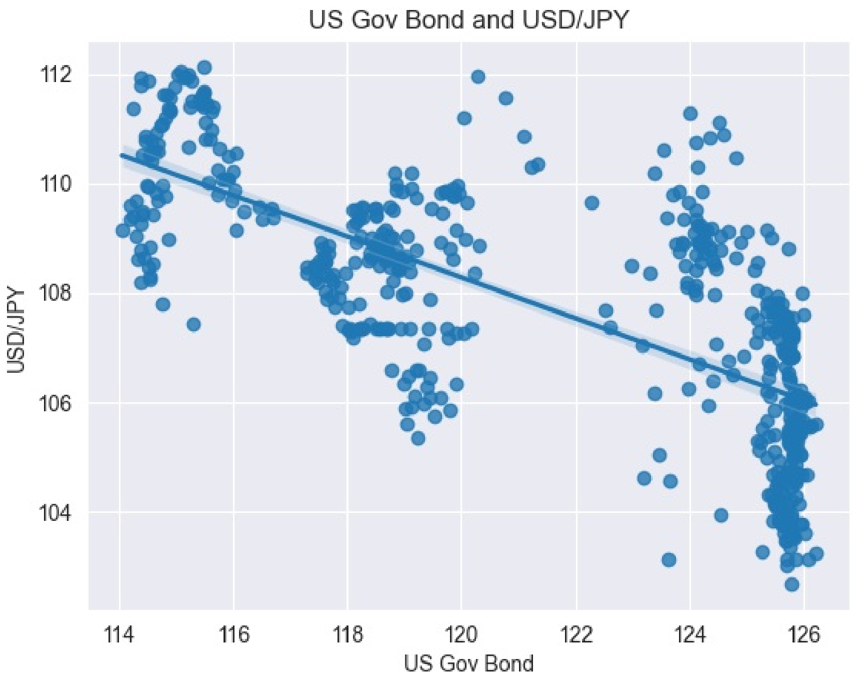
US Gov Bond and Gold

- The slope: 49.9792
- The intercept: -4454.5768
- R-squared: 90.54%
- Total sum of residuals: 27297.4663



US Gov Bond and USD/JPY

- The slope: -0.3748
- The intercept: 153.2520
- R-squared: 49.75%
- Total sum of residuals: 685.0937



volatility

In [201…

```
for c in df.iloc[:,4:7].columns:
    df[f"return_{c}"] = round((df[c].pct_change())*100,2)
```

In [202…

```
df.head()
```

Out[202…

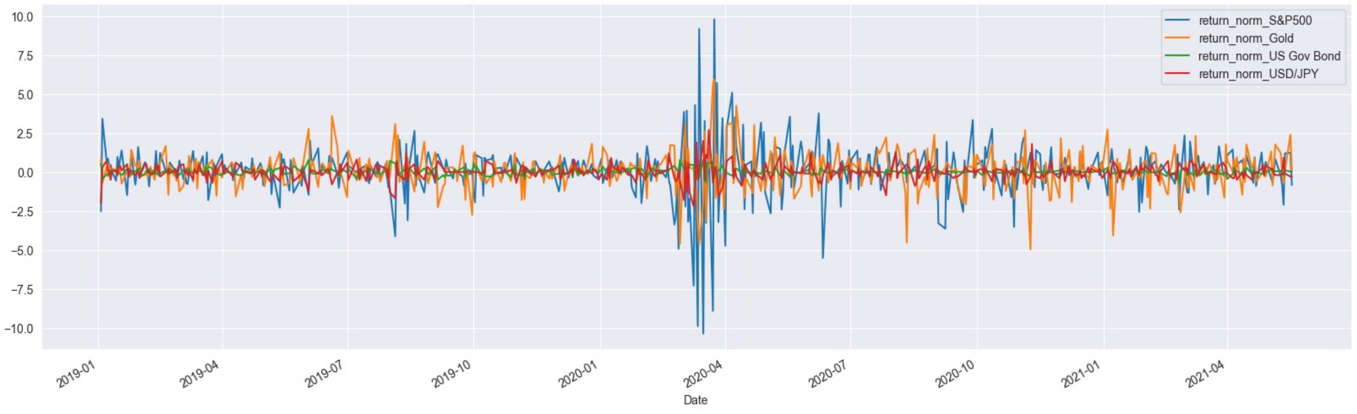
	S&P500	Gold	US Gov Bond	USD/JPY	norm_S&P500	norm_Gold	norm_US Gov Bond	norm_USD/JPY	return_norm_S&P500
Date									
2019-01-02	2511.00	1281.000000	114.687500	109.667998	100.000000	100.000000	100.000000	100.000000	NaN
2019-01-03	2447.75	1291.800049	115.304688	107.441000	97.481083	100.843095	100.538147	97.969327	-2.52
2019-01-04	2531.25	1282.699951	114.757812	107.808002	100.806452	100.132705	100.061308	98.303975	3.41
2019-01-07	2550.50	1286.800049	114.585938	108.521998	101.573078	100.452775	99.911444	98.955028	0.76
2019-01-08	2572.50	1283.199951	114.335938	108.616000	102.449223	100.171737	99.693460	99.040743	0.86

In [254…

```
df[['return_norm_S&P500','return_norm_Gold','return_norm_US Gov Bond','return_norm_USD/JPY']].plot(kind='line',
```

Out[254…

```
<Axes: xlabel='Date'>
```



In [239…

```
std_return = df[['return_norm_S&P500', 'return_norm_Gold', 'return_norm_US Gov Bond','return_norm_USD/JPY']].st
volatility = std_return * np.sqrt(252)
```

In [243…

```
volatility = pd.DataFrame(data=volatility,columns={'volatility':0})
std_return = pd.DataFrame(std_return,columns={'std':0})
```

In [244…

```
return_analysis = pd.concat([std_return,volatility],ignore_index=False,sort=True,axis=1)
```

In [245…

```
return_analysis
```

Out[245…

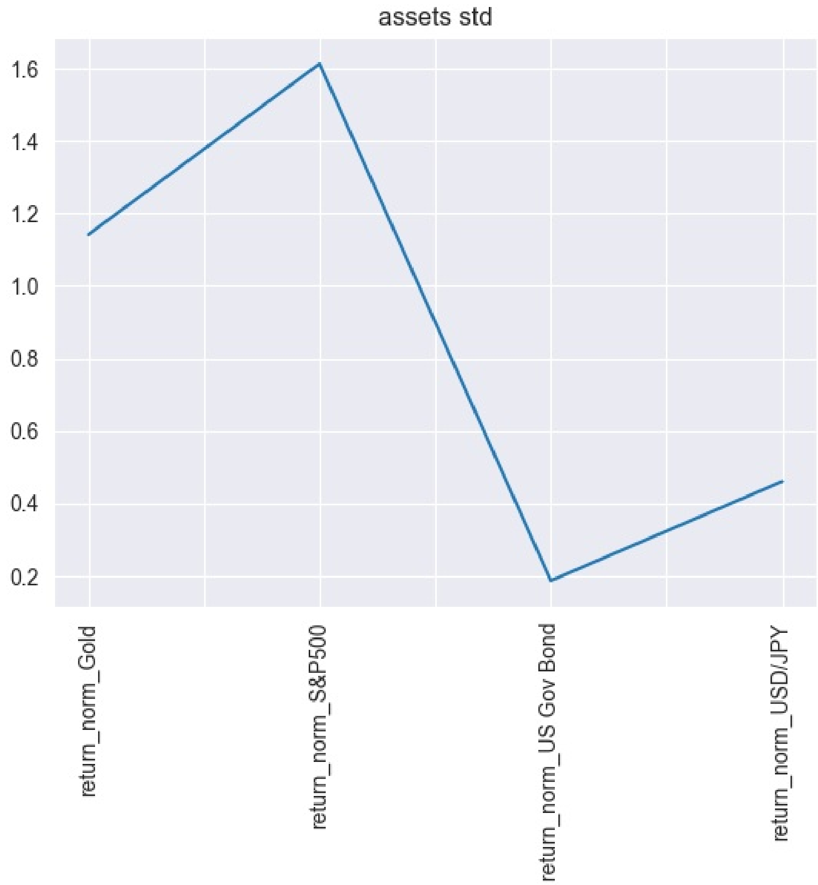
	std	volatility
return_norm_Gold	1.142004	18.128751
return_norm_S&P500	1.613454	25.612781
return_norm_US Gov Bond	0.186744	2.964471
return_norm_USD/JPY	0.460311	7.307208

In [251…

```
return_analysis['std'].plot(kind='line',title="assets std")
plt.xticks(rotation=90)
```

Out[251…

```
(array([-0.5,  0. ,  0.5,  1. ,  1.5,  2. ,  2.5,  3. ,  3.5]),
 [Text(-0.5, 0, ''),
  Text(0.0, 0, 'return_norm_Gold'),
  Text(0.5, 0, ''),
  Text(1.0, 0, 'return_norm_S&P500'),
  Text(1.5, 0, ''),
  Text(2.0, 0, 'return_norm_US Gov Bond'),
  Text(2.5, 0, ''),
  Text(3.0, 0, 'return_norm_USD/JPY'),
  Text(3.5, 0, '')])
```



In [256…

```
df.columns
```

Out[256…

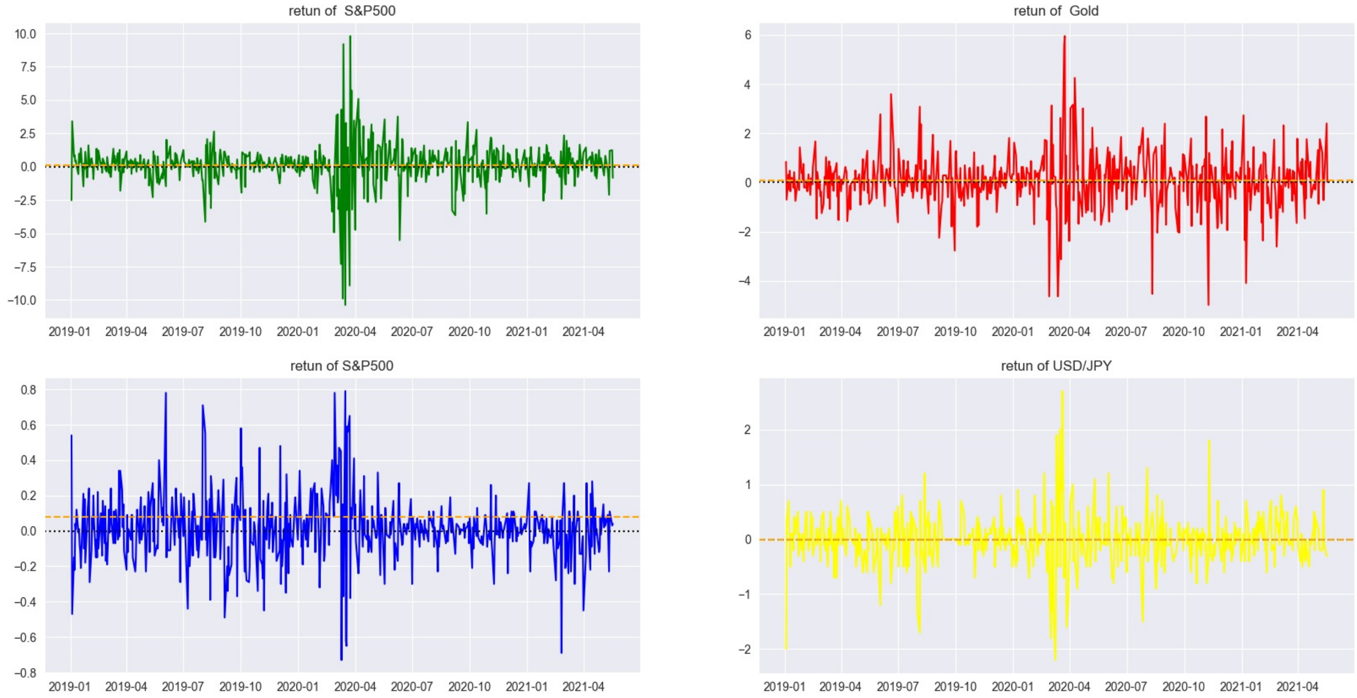
```
Index(['S&P500', 'Gold', 'US Gov Bond', 'USD/JPY', 'norm_S&P500', 'norm_Gold',
       'norm_US Gov Bond', 'norm_USD/JPY', 'return_norm_S&P500',
       'return_norm_Gold', 'return_norm_US Gov Bond', 'return_norm_USD/JPY'],
      dtype='object')
```

In [275…

```
fig ,axs = plt.subplots(2,2,figsize=(20,10))

axs[0,0].plot(df.index,df['return_norm_S&P500'],label='return S&P500',color='green')
axs[0,0].set_title("retun of  S&P500")
axs[0, 0].axhline(y=0, color='black', linestyle=':', label='0')
axs[0, 0].axhline(y=df['return_norm_S&P500'].mean(), color='orange', linestyle='--', label='mean')



axs[0,1].plot(df.index,df['return_norm_Gold'],label='return Gold',color='red')
axs[0,1].set_title("retun of  Gold")
axs[0, 1].axhline(y=0, color='black', linestyle=':', label='0')
axs[0, 1].axhline(y=df['return_norm_Gold'].mean(), color='orange', linestyle='--', label='mean')
```

```
axs[1,0].plot(df.index,df['return_norm_US Gov Bond'],label='return US GOV Bonds',color='blue')
axs[1,0].set_title("retun of S&P500")
axs[1, 0].axhline(y=0, color='black', linestyle=':', label='0')
axs[1, 0].axhline(y=df['return_norm_Gold'].mean(), color='orange', linestyle='--', label='mean')

axs[1,1].plot(df.index,df['return_norm_USD/JPY'],label='return USD/JPY',color='yellow')
axs[1,1].set_title("retun of USD/JPY")
axs[1, 1].axhline(y=0, color='black', linestyle=':', label='0')
axs[1, 1].axhline(y=df['return_norm_USD/JPY'].mean(), color='orange', linestyle='--', label='mean')
```

```
Out[275… <matplotlib.lines.Line2D at 0x217cc0aeac0>
```



Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js