Importing libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn import metrics
```

Data collection and processing

```
#loading csv to pandas data frame
gold_data = pd.read_csv('/content/gld_price_data.csv')
```

```
#print first five rows
gold_data.head()
```

	Date	SPX	GLD	USO	SLV	EUR/USD
0	1/2/2008	1447.160034	84.860001	78.470001	15.180	1.471692
1	1/3/2008	1447.160034	85.570000	78.370003	15.285	1.474491
2	1/4/2008	1411.630005	85.129997	77.309998	15.167	1.475492
3	1/7/2008	1416.180054	84.769997	75.500000	15.053	1.468299
4	1/8/2008	1390.189941	86.779999	76.059998	15.590	1.557099

```
#print last five data frame
gold_data.tail()
```

	Date	SPX	GLD	USO	SLV	EUR/USD
2285	5/8/2018	2671.919922	124.589996	14.0600	15.5100	1.186789
2286	5/9/2018	2697.790039	124.330002	14.3700	15.5300	1.184722
2287	5/10/2018	2723.070068	125.180000	14.4100	15.7400	1.191753
2288	5/14/2018	2730.129883	124.489998	14.3800	15.5600	1.193118
2289	5/16/2018	2725.780029	122.543800	14.4058	15.4542	1.182033

```
#number of rows and columns
```

```
#getting some basic info about data
gold_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2290 entries, 0 to 2289
Data columns (total 6 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   Date     2290 non-null   object 
 1   SPX      2290 non-null   float64
 2   GLD      2290 non-null   float64
 3   USO      2290 non-null   float64
 4   SLV      2290 non-null   float64
 5   EUR/USD  2290 non-null   float64
dtypes: float64(5), object(1)
memory usage: 107.5+ KB
```

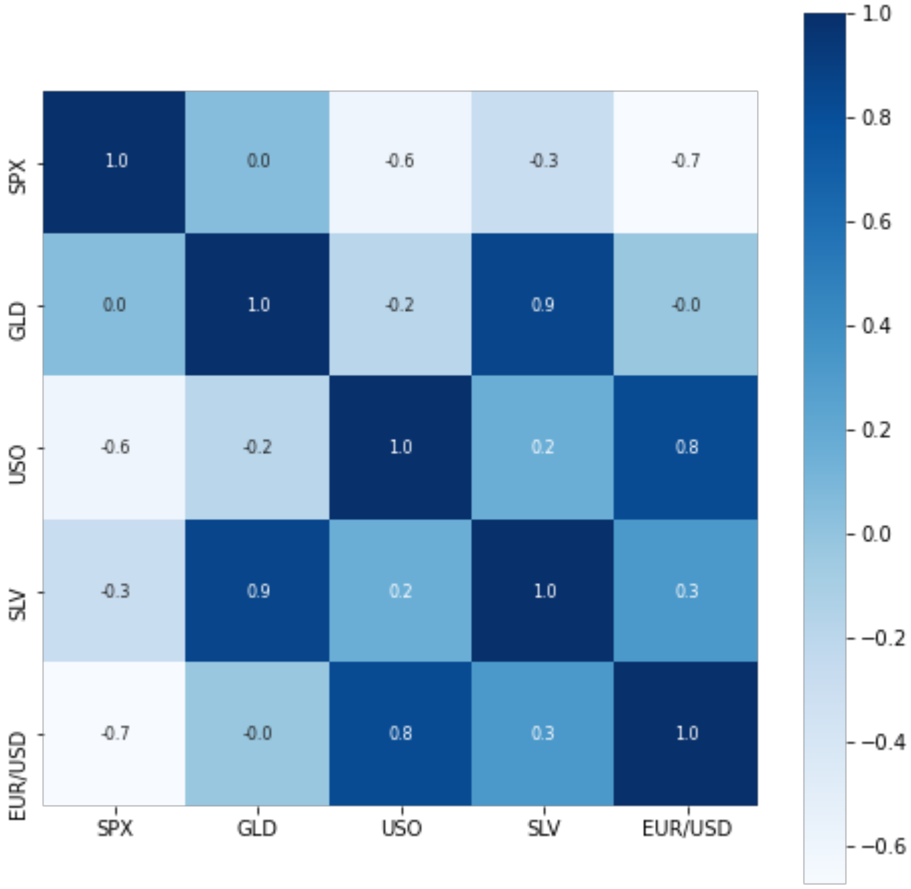
```
#checking number of missing values
gold_data.isnull().sum()
```

```
Date       0
SPX        0
GLD        0
USO        0
SLV        0
EUR/USD    0
dtype: int64
```

```
#getting the statistical measures of the data
gold_data.describe()
```

	SPX	GLD	USO	SLV	EUR/USD
count	2290.000000	2290.000000	2290.000000	2290.000000	2290.000000
mean	1654.315776	122.732875	31.842221	20.084997	1.283653
std	519.111540	23.283346	19.523517	7.092566	0.131547
min	676.530029	70.000000	7.960000	8.850000	1.039047
25%	1239.874969	109.725000	14.380000	15.570000	1.171313
50%	1551.434998	120.580002	33.869999	17.268500	1.303296
75%	2073.010070	132.840004	37.827501	22.882499	1.369971

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc9717b6ed0>
```



```
#correlation values of GLD
```

```
print(correlation['GLD'])
```

```
SPX        0.049345
GLD        1.000000
USO       -0.186360
SLV        0.866632
EUR/USD   -0.024375
Name: GLD, dtype: float64
```

```
#checking the distribution of the GLD price
sns.displot(gold_data['GLD'],color='green')
```

splitting the features and target (gold and date)

```
X=gold_data.drop(['Date','GLD'],axis=1)
Y=gold_data['GLD']
```

```
print(X)
```

```
              SPX        USO      SLV   EUR/USD
0     1447.160034  78.470001  15.1800  1.471692
1     1447.160034  78.370003  15.2850  1.474491
2     1411.630005  77.309998  15.1670  1.475492
3     1416.180054  75.500000  15.0530  1.468299
4     1390.189941  76.059998  15.5900  1.557099
...           ...        ...      ...       ...
2285  2671.919922  14.060000  15.5100  1.186789
2286  2697.790039  14.370000  15.5300  1.184722
2287  2723.070068  14.410000  15.7400  1.191753
2288  2730.129883  14.380000  15.5600  1.193118
2289  2725.780029  14.405800  15.4542  1.182033

[2290 rows x 4 columns]
```

```
print(Y)
```

```
0       84.860001
1       85.570000
2       85.129997
3       84.769997
4       86.779999
          ...
```

```
regressor.fit(X_train,Y_train)
```

```
    RandomForestRegressor(bootstrap=True, ccp_alpha=0.0, criterion='mse',
                          max_depth=None, max_features='auto', max_leaf_nodes=N
                          max_samples=None, min_impurity_decrease=0.0,
                          min_impurity_split=None, min_samples_leaf=1,
                          min_samples_split=2, min_weight_fraction_leaf=0.0,
                          n_estimators=100, n_jobs=None, oob_score=False,
                          random_state=None, verbose=0, warm_start=False)
```

Model evaluation

```
#prediction on Test Data
test_data_prediction=regressor.predict(X_test)
```

```
print(test_data_prediction)
```

```
    [168.68449967  82.02389996 116.17810003 127.66170085 120.82890131
     154.74819774 150.15929927 126.16140027 117.41029883 125.87800101
     116.76950108 171.5890007  141.95409882 168.19929899 115.20049982
     117.73670044 136.97140352 170.11210059 159.4500027  157.45339942
     154.89170025 125.39660044 176.23649954 156.45690328 125.15760044
      93.9111996   77.66950018 120.59830016 118.9816988  167.45359896
      88.27430047 125.25120009  91.15660076 117.75820031 121.0192995
     136.67070118 115.47530089 114.43630084 147.97420017 107.13990098
     104.36130255  87.19379803 126.56630038 117.81100012 153.80609928
```

```
 154.72820079 114.28770122 138.83100151  90.13569812 115.60169971
 114.58449933 123.03300034 121.77270013 165.21610137  92.83459979
 135.127201   121.43629898 120.90700049 104.77760014 140.45060319
 121.83209901 116.64780041 113.58400063 127.01209765 122.82469942
 125.80669935 121.26220039  86.76309869 132.51220131 145.9791022
  92.6725994  159.18219961 159.50600246 126.35929908 164.97949922
 108.86529956 109.54000102 103.90329871  94.38880039 127.80910279
 107.09960039 161.64299963 121.75510044 132.20719989 130.2038021
 160.27409968  90.18799843 176.20870169 127.47270027 126.67839891
  86.36019928 124.4944992  150.60619729  89.55320025 106.85389964
 109.02779995  84.41309903 136.06449885 155.10220289 138.41960359
  74.34940021 152.71120143 126.10559995 126.66710051 127.4386992
 108.7787     156.05550057 114.64670114 116.89190147 125.13259945
 154.00800162 121.35999979 156.38809867  92.8839007  125.44870155
 125.80250019  87.78760045  92.11269925 126.14029967 128.2608033
```

```
#R square error
error_score=metrics.r2_score(Y_test, test_data_prediction)
print("R square error:",error_score)
```

```
R square error: 0.9895186991521377
```

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7fc962c6fd10>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fc962c29350>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7fc962bde9d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fc962c09b90>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7fc962b57710>,
```

top	2009-06-24	NaN	NaN	NaN	NaN
freq	1	NaN	NaN	NaN	NaN
mean	NaN	1654.315776	122.732875	31.842221	20.084997
std	NaN	519.111540	23.283346	19.523517	7.092566
min	NaN	676.530029	70.000000	7.960000	8.850000
25%	NaN	1239.874969	109.725000	14.380000	15.570000
50%	NaN	1551.434998	120.580002	33.869999	17.268500