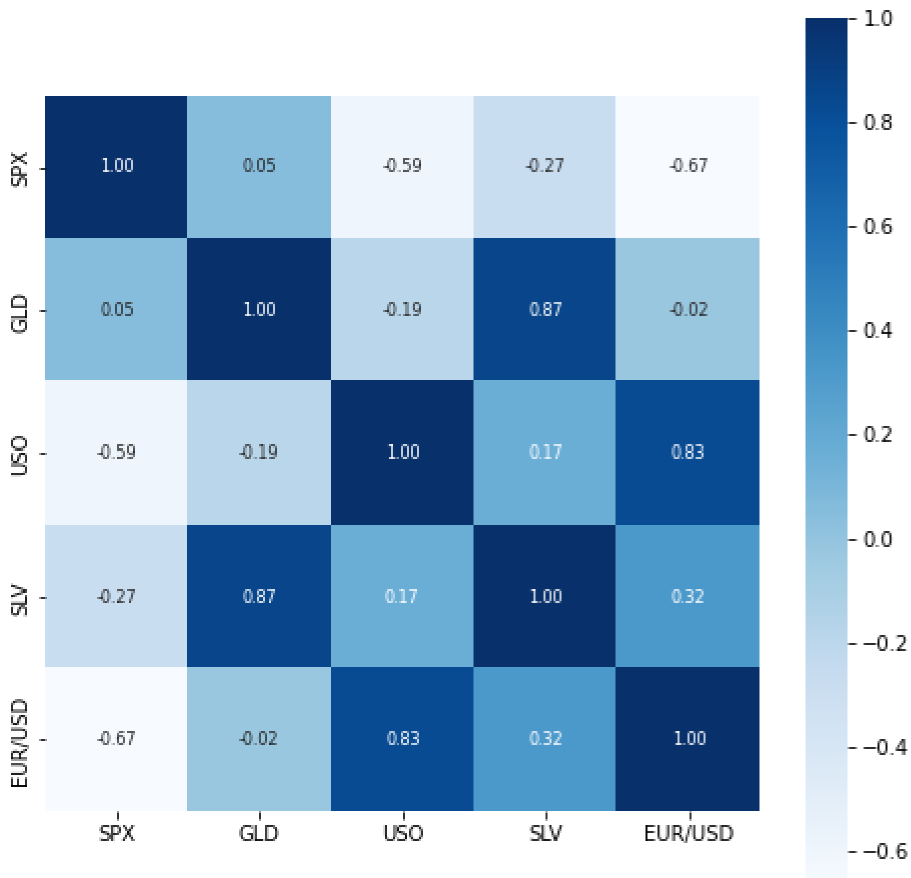
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn import metrics
```

```
gld_data=pd.read_csv('/content/gld_price_data.csv')
```

```
correlation=gld_data.corr()
```

```
plt.figure(figsize=(8,8))
sns.heatmap(correlation,cbar=True,square=True,fmt='.2f',annot=True,annot_kws={'size':8},cmap=
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7efead9e12e0>
```




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

```
SPX        0.049345
GLD        1.000000
USO       -0.186360
SLV        0.866632
```

```
EUR/USD   -0.024375
Name: GLD, dtype: float64
```

```
x=gld_data.drop(['GLD','Date'],axis=1)
```

```
gld_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

```
y=gld_data['GLD']
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=2)
```

```
regressor = RandomForestRegressor(n_estimators=100)
```

```
regressor.fit(x_train,y_train)
regressor.get_params(deep=True) #here sometimes by default,the hyperparameters of the N estim
```

```
{'bootstrap': True,
 'ccp_alpha': 0.0,
 'criterion': 'squared_error',
 'max_depth': None,
 'max_features': 'auto',
 'max_leaf_nodes': None,
 'max_samples': None,
 'min_impurity_decrease': 0.0,
 '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}
```

```
test_data_prediction=regressor.predict(x_test)
```

```
print(test_data_prediction)
```

```
 165.95510022 114.91870106 116.71030126  88.36189882 148.96340117
 120.37939926  89.37029968 111.88669995 116.98360048 118.81110104
  88.11879928  94.12389997 116.89599982 118.59260156 119.97440012
 126.7673982  121.92479995 149.78960021 165.19320025 118.57769961
 120.27830159 149.9038995  118.57649909 172.47759891 105.50379931
 105.10150115 149.70990141 113.91160118 124.9129009  147.79539904
 119.58390128 115.45830041 112.21219969 113.56570188 141.65960139
 117.61569789 103.00880057 115.98080108 103.5045017   98.49940032
 117.34240066  90.52810027  91.51730049 153.4386993  102.76759989
 154.75090142 114.41700164 138.43810118  90.2881982  115.46079947
 114.67269973 122.8838999  121.81580016 165.66500105  92.8595992
 135.64120166 121.26649931 120.9022004  104.61630024 142.38730244
 121.51909916 116.66050035 113.60490124 127.39489742 122.70349963
 125.75169926 121.24500006  86.87759896 132.14160163 143.9464023
  92.70399931 157.72839945 159.84750268 126.30429914 165.10949924
 108.75699931 109.39250096 103.73009824  94.00220111 127.86920298
 106.67960076 161.60209967 121.77540023 132.20319974 130.68240146
 159.98090037  90.28859884 176.53270258 127.78430018 126.88829839
  86.38669933 124.55979941 150.30489708  89.65690008 106.63169976
 108.90929979  83.71009935 135.79800025 155.29090274 139.72600285
  73.91830042 151.96150051 125.8643     126.72800016 127.53529932
 108.60659936 156.40680033 114.71070113 116.81430146 125.38439942
 154.12160135 121.34860008 156.4071988   92.94410064 125.49020158
 126.12240073  87.9038003   92.08249935 126.19589935 128.09350263
 113.14190005 117.61719746 120.92920026 127.52489772 119.67990118
 135.96920045  93.96779938 119.83770064 113.05620107  94.35839942
 108.61419961  87.29399918 109.29819933  89.7856997   92.41139996
 131.86610306 162.23210032  89.35299993 119.6053009  133.30530187
 123.84669993 128.20810188 101.94189859  88.95909859 131.4527004
 120.01020049 108.5580996  167.01120044 115.19960059  86.65879869
 118.91760056  91.13199998 161.61720098 116.55330039 121.50850017
 160.38489774 120.03859933 112.8331998  108.40219856 126.7670992
  76.37410032 102.98999989 127.80370303 121.77629912  92.62189982
 132.20560021 118.06850105 115.87919974 154.81150296 159.88380082
 109.9013996  156.38039836 119.24000052 160.67030045 118.7417006
 158.76959954 115.08979951 116.46960015 150.01609954 114.90550066
 125.66969895 165.27879961 117.79360014 125.06049906 153.61440358
 153.36050257 132.24300072 114.93320018 121.19300219 124.86110056
  89.73830056 123.44260003 155.24460187 111.82140038 106.53749999
 161.21680086 118.54179979 165.64610058 134.16280111 114.8801997
 152.91549845 168.61920074 114.99160036 113.81830145 158.54859913
  85.21789896 127.13280033 127.9342002  128.79499963 124.28440086
 123.99430073  90.74510071 153.45790021  97.18369975 137.32499926
  89.12129942 107.04769991 115.06080064 112.72820069 124.31319939
  91.39659884 125.20400103 162.27079795 120.10809883 165.18120068
 126.69459805 112.42009997 127.61759926  94.72289927  90.82439978
 103.58849902 120.84270029  82.96509923 126.34989954 159.65050398
 117.22630095 118.30479965 120.19529974 122.79579974 120.12210135
 121.54540005 118.24210007 106.89599996 148.6816003  126.31729841
 115.74300079  73.94360026 127.74640094 154.49310112 122.85749979
 125.61540093  88.80040069 103.34999839 124.40340063 120.18640012
  73.40100079 151.86419982 121.22650026 104.65020005  86.55919758
 115.11759926 172.28849794 119.93880049 160.74389755 113.19889935
 121.60450003 118.54540142  95.9379999  118.58149989 125.7147002
```

```
 118.49229978  95.83150058 154.29780199 122.20310046 147.64110045
 159.52530216 114.05040012 122.51229953 149.95689767 127.32530051
 165.72240055 135.30390044 119.7800992  166.22709821 108.5292991
 121.65439888 138.78310119 106.77699887]
```

```
error=metrics.r2_score(y_test,test_data_prediction)
```

```
print("the squared error is:",error)
```

```
the squared error is: 0.9892816690399935
```

```
y_test=list(y_test)
plt.plot(y_test,color='Blue',label='actualvalue')
plt.plot(test_data_prediction,color='red',label='predicted values')
plt.xlabel('number of values')
plt.ylabel('gld price')
plt.legend()
plt.show()
```