**import** **pandas** **as** **pd**

**import** **numpy** **as** **np**

**import** **matplotlib.pyplot** **as** **plt**

**import** **seaborn** **as** **sns**

**from** **sklearn.model\_selection** **import** train\_test\_split

**from** **sklearn** **import** metrics

**from** **sklearn.ensemble** **import** RandomForestRegressor

In [9]:

gold\_data = pd.read\_csv("gold price dataset.csv")

In [10]:

print(gold\_data)

Date SPX GLD USO SLV EUR/USD

0 1/2/2008 1447.160034 84.860001 78.470001 15.1800 1.471692

1 1/3/2008 1447.160034 85.570000 78.370003 15.2850 1.474491

2 1/4/2008 1411.630005 85.129997 77.309998 15.1670 1.475492

3 1/7/2008 1416.180054 84.769997 75.500000 15.0530 1.468299

4 1/8/2008 1390.189941 86.779999 76.059998 15.5900 1.557099

... ... ... ... ... ... ...

2285 5/8/2018 2671.919922 124.589996 14.060000 15.5100 1.186789

2286 5/9/2018 2697.790039 124.330002 14.370000 15.5300 1.184722

2287 5/10/2018 2723.070068 125.180000 14.410000 15.7400 1.191753

2288 5/14/2018 2730.129883 124.489998 14.380000 15.5600 1.193118

2289 5/16/2018 2725.780029 122.543800 14.405800 15.4542 1.182033

[2290 rows x 6 columns]

In [11]:

gold\_data.head()

Out[11]:

|  | **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 |

In [12]:

gold\_data.tail()

Out[12]:

|  | **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 |

In [13]:

gold\_data.shape

Out[13]:

(2290, 6)

In [14]:

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

In [16]:

gold\_data.isnull().sum()

Out[16]:

Date 0

SPX 0

GLD 0

USO 0

SLV 0

EUR/USD 0

dtype: int64

In [17]:

gold\_data.describe()

Out[17]:

|  | **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 | 1.303297 |
| **75%** | 2073.010070 | 132.840004 | 37.827501 | 22.882500 | 1.369971 |
| **max** | 2872.870117 | 184.589996 | 117.480003 | 47.259998 | 1.598798 |

In [18]:

correlation = gold\_data.corr()

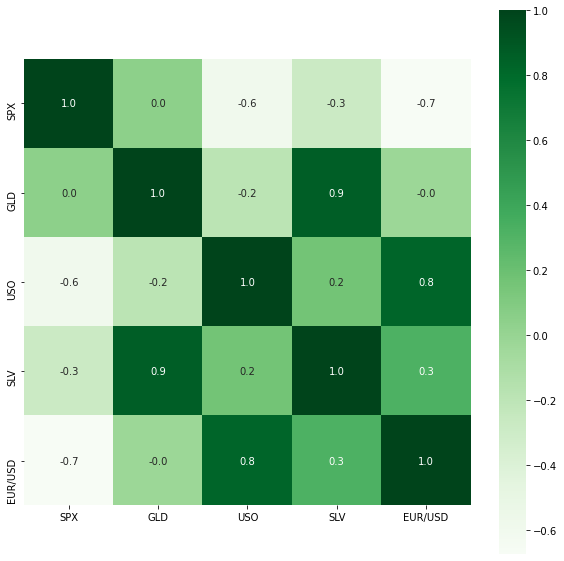
In [21]:

plt.figure(figsize=(10,10))

sns.heatmap(correlation, cbar=**True**, square=**True**, fmt='.1f', annot=**True**, annot\_kws={'size':10}, cmap= 'Greens')

Out[21]:

<AxesSubplot:>



In [22]:

print(correlation['GLD'])

SPX 0.049345

GLD 1.000000

USO -0.186360

SLV 0.866632

EUR/USD -0.024375

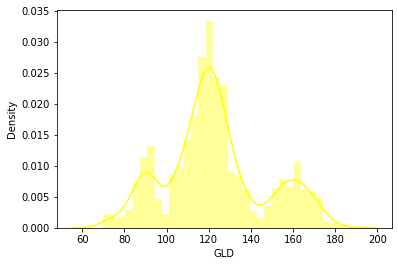
Name: GLD, dtype: float64

In [24]:

sns.distplot(gold\_data['GLD'], color='yellow')

Out[24]:

<AxesSubplot:xlabel='GLD', ylabel='Density'>



In [27]:

x = gold\_data.drop(['Date','GLD'], axis=1)

y = gold\_data['GLD']

In [28]:

print(x)

print(y)

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]

0 84.860001

1 85.570000

2 85.129997

3 84.769997

4 86.779999

...

2285 124.589996

2286 124.330002

2287 125.180000

2288 124.489998

2289 122.543800

Name: GLD, Length: 2290, dtype: float64

In [39]:

x\_train,x\_test,y\_train,y\_test = train\_test\_split(x,y, test\_size=0.20, random\_state=2)

In [40]:

print(x.shape)

print(y.shape)

print(x\_train.shape)

print(x\_test.shape)

(2290, 4)

(2290,)

(1832, 4)

(458, 4)

In [41]:

regressor= RandomForestRegressor(n\_estimators=100)

In [42]:

regressor.fit(x\_train,y\_train)

Out[42]:

RandomForestRegressor()

In [43]:

test\_data\_prediction = regressor.predict(x\_test)

In [44]:

print(test\_data\_prediction)

[168.75469943 81.97799976 116.11560007 127.66290084 121.03300119

154.54459746 150.36669832 126.15739999 117.36239881 126.04430045

116.73220084 171.53600112 141.78509875 167.64729757 115.03380047

117.56490048 138.80860338 170.01690115 159.55250299 158.17139991

155.18579993 125.36309973 175.37999981 157.05340335 125.29240047

93.70210009 77.22110056 120.54250023 119.13759984 167.44749968

88.4247 125.29599999 91.14590088 117.78800008 121.06279883

136.85760138 115.62930146 115.4959007 147.03349926 107.16090099

104.60970267 87.1568978 126.49110088 118.06199959 151.63319873

119.47380006 108.18900029 108.16059816 93.09150034 127.23109732

74.96150018 113.52729941 121.3048002 111.4799991 118.99769892

120.55469964 158.74349895 168.6015007 146.99059674 86.16489883

94.27780027 86.80059862 90.34200039 118.94100061 126.42330039

127.68750047 170.43870046 122.17889932 117.47679912 98.34980046

167.90930147 143.15769815 131.77500307 121.23320193 121.05949953

119.6820008 114.60400158 118.05360061 107.1036008 127.89380043

113.99779955 107.27820005 116.76600064 119.73249909 89.05210042

88.14719855 146.70520217 127.36730015 113.35900034 109.80509838

108.22719896 77.51169884 169.49500153 113.96669895 121.54129922

127.9542016 154.94659854 91.95379925 136.26060177 159.09930445

124.98340075 125.37260037 130.87820209 115.07710137 119.77780037

92.06150002 110.34579898 168.7095998 157.04249894 114.10579936

106.67690126 79.29289999 113.18020029 125.77000078 107.31669923

119.30940102 155.42500306 159.31949944 119.71640019 133.99070295

101.2461999 117.53059823 119.38649988 112.94360068 102.77109924

160.03359818 99.23100042 147.96539952 125.66200122 169.58240007

125.72229865 127.31339755 127.4663017 113.55139937 112.65870054

123.47219869 102.24499926 89.40629946 124.31109958 101.79099948

107.26829898 113.56130062 117.30240074 99.48129963 121.53940093

163.02939907 87.47549858 106.98959992 117.39740025 127.79350087

124.15850059 80.69589911 120.37120049 157.16169863 88.01969966

110.39289925 118.95899894 172.35039879 102.91989913 105.56400088

122.74550034 157.48909818 87.73019837 93.38720068 112.6843004

177.14639958 114.78149989 119.43989962 94.54990091 125.8434

166.11990005 114.89580088 116.80670136 88.26139862 148.49670073

120.34879944 89.50019977 111.97929975 117.35670046 118.84560131

88.21739959 94.04089972 116.86209987 118.59770156 120.09890022

126.61939848 121.86270008 150.13920041 165.34810072 118.61209958

120.32140137 150.00140022 118.67539905 172.56429886 105.24339941

104.91940171 148.47020078 113.85620034 124.89310111 148.13730018

119.6465012 115.42330091 112.14919992 113.65370174 143.0140017

117.75089806 102.93890028 115.86740139 103.90790193 99.12270048

117.33230089 90.55610051 91.60180074 153.21019945 102.64549987

154.84970083 114.40740157 138.60490094 90.08949857 115.50749918

114.71309957 122.86060054 121.81580055 165.39830162 92.97099932

134.77140119 121.36449935 120.6817006 104.61270051 143.73370279

121.94099932 116.64790035 113.44080071 127.00559785 122.59129929

125.92289969 121.2657004 86.85099868 132.15280173 144.39840211

92.67769944 158.6762994 159.17520374 126.04009925 164.667899

108.6749991 110.41640033 103.64049832 94.17200086 127.91580275

107.25950048 162.19309977 121.6953008 132.10620036 130.88230229

160.8989001 90.12639911 174.89520165 127.74190047 126.68299843

86.60109906 124.5339991 150.183697 89.71390011 107.03789971

109.01259992 84.60059914 135.73389918 154.92140274 139.81560324

73.5102005 152.46810067 126.02650059 126.70379982 127.59689888

108.55339958 156.04329993 114.68120126 117.0239015 125.14139947

153.95660192 121.39859955 156.34439882 92.93120057 125.46840116

125.47380019 87.70740012 92.36709921 126.18639941 128.38120407

113.20550031 117.48309735 120.9731001 127.09039791 119.62530113

137.11170155 93.94969948 119.83200036 113.38530128 94.23959932

108.83250002 87.11589928 109.21319928 89.70949956 92.62830003

131.5787027 162.32780039 89.34950015 119.42930106 133.24720188

124.02560021 128.30630236 101.98479874 88.8671987 132.07780071

119.8998004 108.91490003 167.5675005 115.22700038 86.56759854

118.80000069 91.0004994 161.5311001 116.27050019 121.84189998

160.21779753 120.04459913 112.75399949 108.53399851 126.71960015

76.02980006 102.97349983 128.02570282 121.871999 92.63660017

131.63620078 118.2790013 116.07519983 154.33070288 159.29190062

110.07859978 154.46319742 119.26590097 160.38520065 118.68290051

156.80109949 115.19339896 116.55270039 148.38419851 114.71610088

125.47619889 166.38429904 117.78870017 125.08299935 153.10990361

153.48560224 132.09150045 114.68310085 121.36180212 125.17540057

89.78470019 123.13549992 154.47450165 111.71430028 106.94659978

161.80170186 118.33719963 165.73270014 134.15590123 115.22839945

153.04169935 168.66830062 115.29180024 114.12840135 159.41289924

85.31519872 127.20250028 127.7918008 128.86279952 124.08800046

123.85490048 90.38720084 153.16209993 97.10259977 137.45879948

88.94899924 107.28540014 115.09450051 112.67070087 123.93839935

91.45999881 125.42720155 162.47679946 119.95689908 165.11680171

126.62339832 112.37740024 127.50799914 94.78699916 90.68780002

102.78509913 120.85130023 83.22939969 126.34340008 160.34450464

117.26550094 118.25180004 120.13259998 123.00039979 120.09530153

121.58609989 118.30000059 107.08089981 147.88189965 126.12459826

115.64420065 74.08510011 127.74380071 154.36810052 122.30169991

125.63400058 88.91820034 103.29059855 124.16690042 120.24600018

73.23940073 151.34540036 121.27080035 104.75520046 86.47689769

115.07899904 172.09749913 119.93060023 159.59819838 113.1456994

121.31569966 118.89360135 95.86409989 118.86900019 125.8273005

118.68239987 95.7490009 154.11410193 122.15749967 147.78250042

159.40430269 113.8341008 122.49529936 150.24939838 127.03910018

165.85760013 136.11209993 119.69809952 167.55719855 108.34039937

121.93079875 140.15000127 105.83129926]

In [45]:

r\_score = metrics.r2\_score(y\_test, test\_data\_prediction)

In [46]:

print(r\_score)

0.9886222549758875

In [47]:

y\_test = list(y\_test)

In [50]:

plt.plot(y\_test, color='green', label ='Actual Value')

plt.plot(test\_data\_prediction, color='red', label='predicted value')

plt.title('Actual value vs Predicted value')

plt.xlabel('number of values')

plt.ylabel('GLD Prices')

plt.legend()

plt.show()

