Gold_Price Prediction Using ML

[1]:

```
import numpy as np
import pandas as pd
import os
```

[2]:

```
gold=pd.read_csv("gold_price_data.csv")
gold.head()
```

[2]:

```
       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
```

[3]:

```
gold.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
```

[4]:

```
gold.describe()
```

[4]:

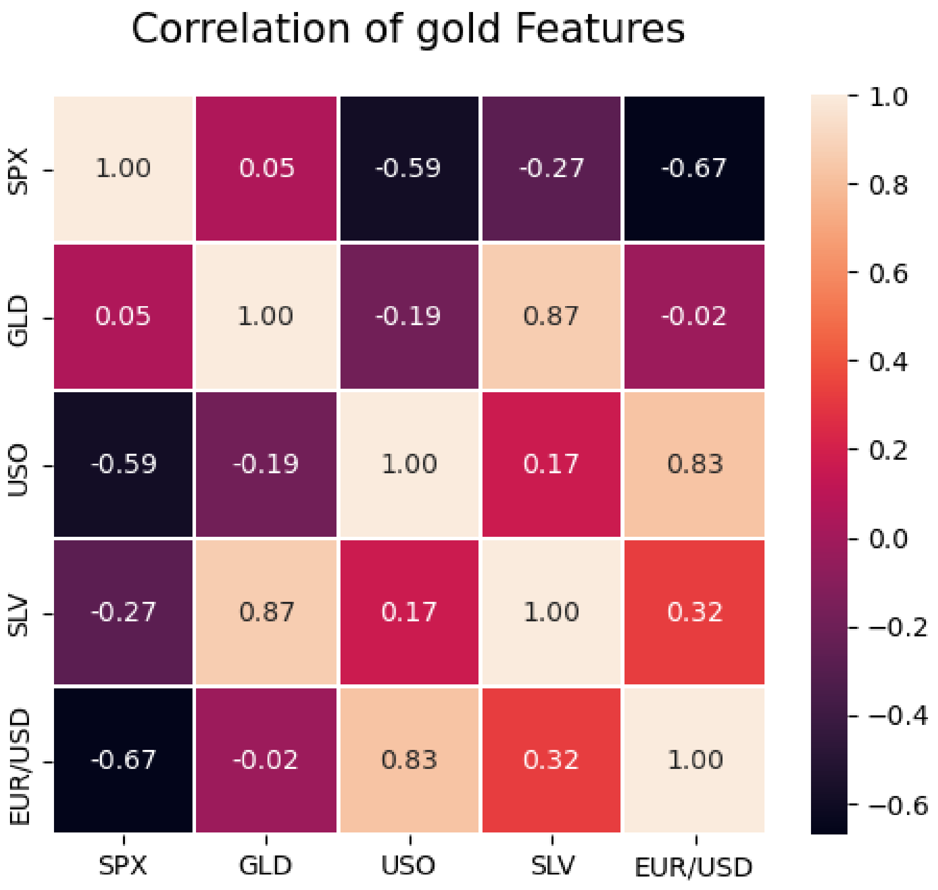
```
               SPX          GLD          USO          SLV      EUR/USD
count  2290.000000  2290.000000  2290.000000  2290.000000  2290.000000
mean   1654.315776   122.732875    31.842221    20.084997     1.283653
std     519.111540    23.283346    19.523517     7.092566     0.131547
min     676.530029    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
```

[5]:
```
import matplotlib.pyplot as plt
import seaborn as sns
corr = gold.corr()
plt.figure(figsize = (6,5))
sns.heatmap(corr,
            xticklabels=corr.columns.values,
            yticklabels=corr.columns.values,
            annot=True,fmt='.2f',linewidths=0.30)
plt.title('Correlation of gold Features',y=1.05,size=15)
```

```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_10332\2123799203.py:3:
FutureWarning: The default value of numeric_only in DataFrame.corr is
deprecated. In a future version, it will default to False. Select only valid
columns or specify the value of numeric_only to silence this warning.
  corr = gold.corr()
```

[5]:
```
Text(0.5, 1.05, 'Correlation of gold Features')
```



[6]:

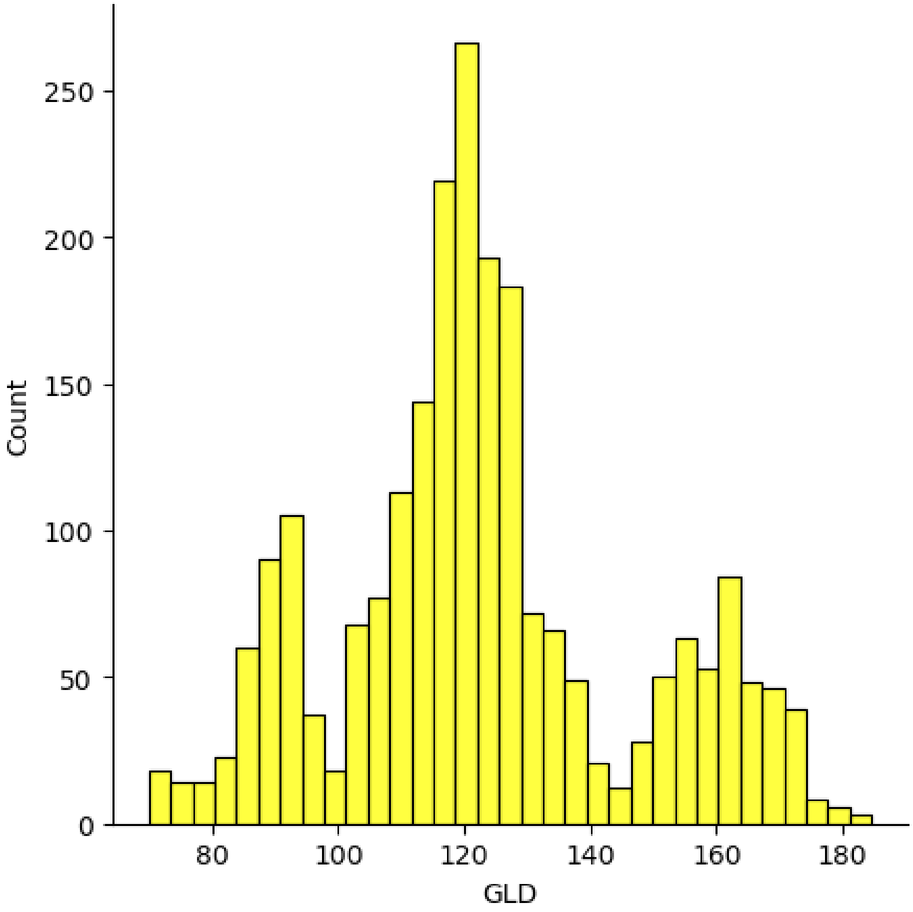
```python
#Lets look the correlation score
print(corr['GLD'].sort_values(ascending=False),'\n')
```

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

[7]:

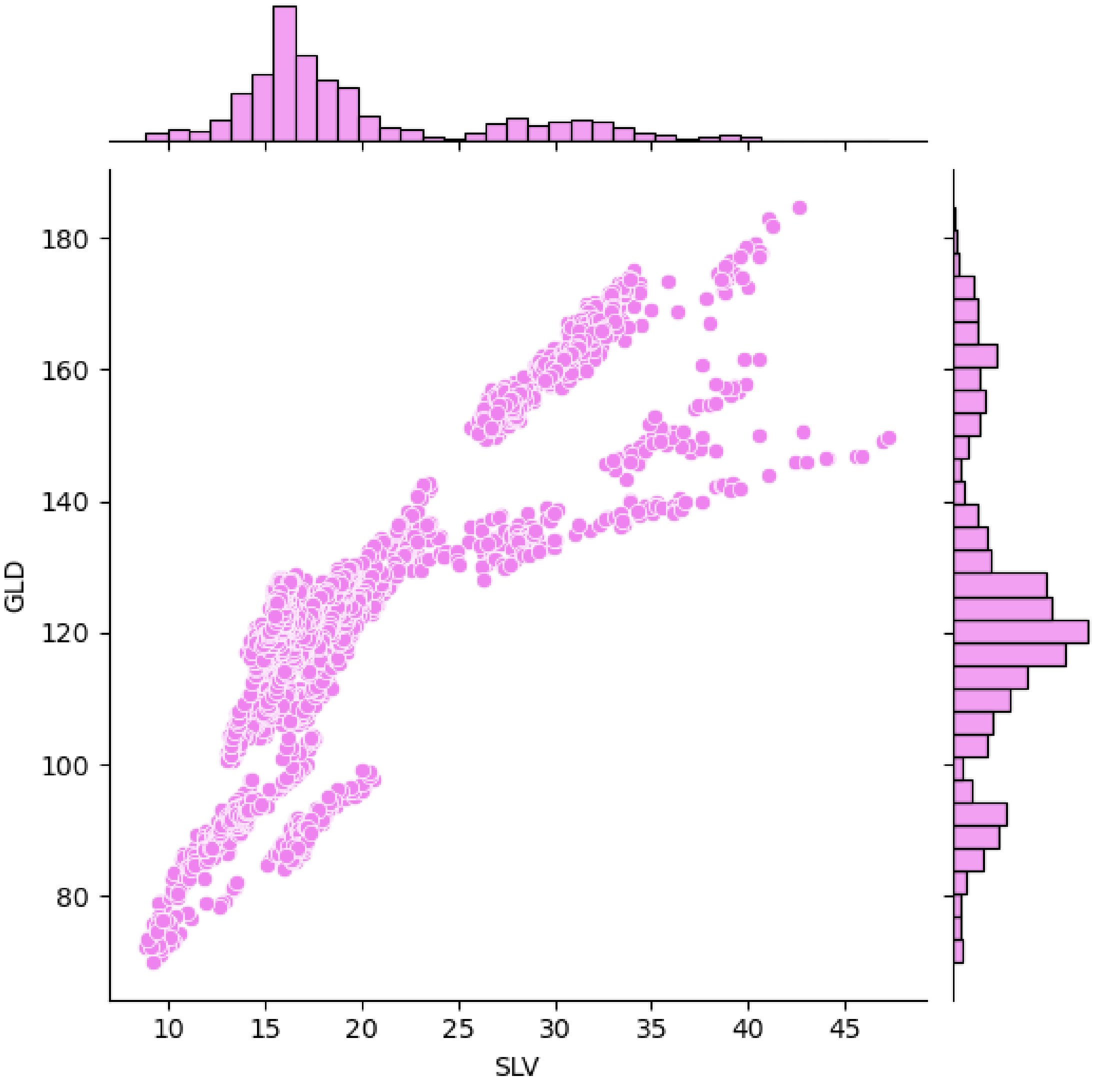
```python
#Lets check our targets variable
sns.displot(gold['GLD'],color = 'yellow')
print('Skewness: %f', gold['GLD'].skew())
print("Kurtosis: %f" % gold['GLD'].kurt())
```

```
Skewness: %f 0.3341383472692508
Kurtosis: -0.275081
```



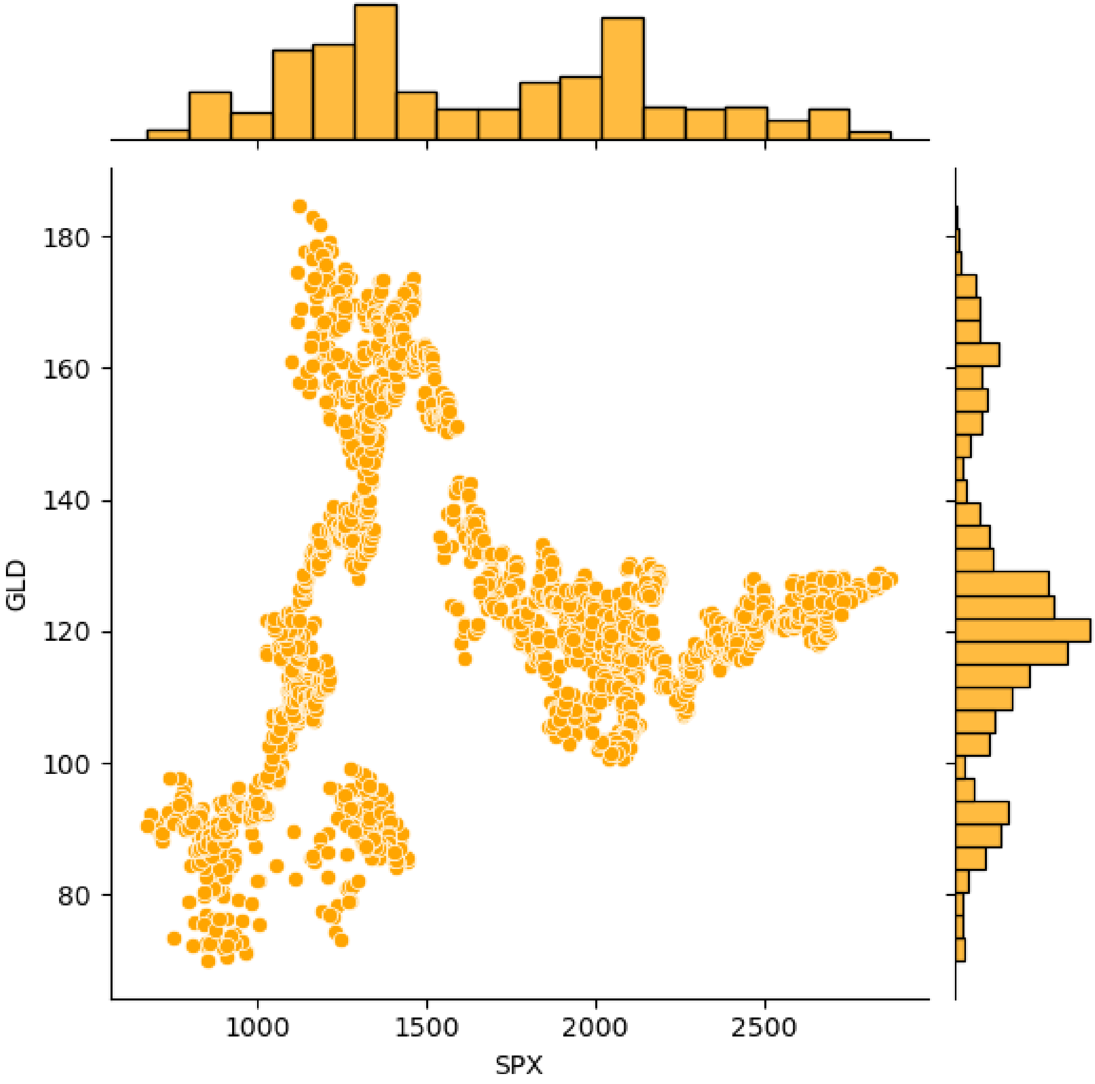
[8]:
```
#now we  check  the  relation  with  GLD  variable
sns.jointplot(x =gold['SLV'], y = gold['GLD'], color = 'violet')
```

[8]:
```
<seaborn.axisgrid.JointGrid at 0x25d84c2c160>
```



[9]:
```
#Now we  check  the  relation  with  gld  variable
sns.jointplot(x =gold['SPX'], y = gold['GLD'], color = 'orange')
```

[9]:
```
<seaborn.axisgrid.JointGrid  at  0x25dff2fe8c0>
```



[10]:

```
#Now Lets create a ml model
#Now lets take our matrix of Feature and Target
x_trail = gold[['SPX','USO','SLV','EUR/USD']]
x =  x_trail.iloc[:, :].values
y = gold.iloc[:,2].values
```

[11]:

```
#spliting the dataset into training and test set
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size = 0.2,
 random_state = 0)
```

[12]:

```
#Now fitting the Random forest regression to the traning set
from sklearn.ensemble import RandomForestRegressor
```

```
regressor = RandomForestRegressor(n_estimators = 100, random_state = 0)
regressor.fit(x_train, y_train)
```

[12]:

```
RandomForestRegressor(random_state=0)
```

[13]:

```
#Now predicting the test set result
y_pred  =  regressor.predict(x_test)
```

[14]:

```
#Now check the error for the regression
from sklearn import metrics
print("MAE :"," ", metrics.mean_absolute_error(y_test,y_pred))
print("MSE :"," ", metrics.mean_squared_error(y_test,y_pred))
print("RMAE :"," ",  np.sqrt(metrics.mean_squared_error(y_test,y_pred)))
```

```
MAE :    1.297793151724892
MSE :    5.16257387057774
RMAE :    2.272129809358994
```

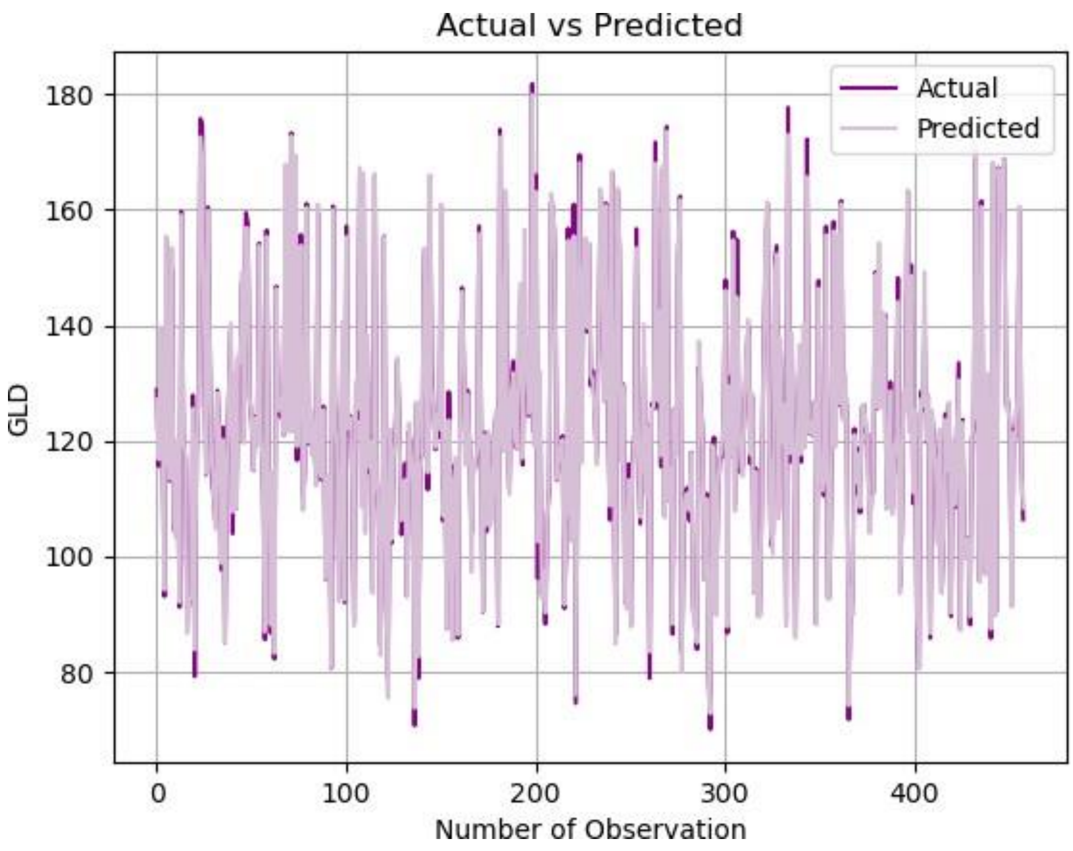
[15]:

```
#Now Lets check the training and test set Accuracy
accuracy_train = regressor.score(x_train, y_train)
accuracy_test = regressor.score(x_test, y_test)
print(accuracy_train)
print(accuracy_test)
```

```
0.9984324726699736
0.9899648553789232
```

[16]:

```
#Visualising the Accuracy of Predicted result
plt.plot(y_test, color = 'darkmagenta', label = 'Actual')
plt.plot(y_pred, color = 'thistle', label = 'Predicted')
plt.grid(0.3)
plt.title('Actual vs Predicted')
plt.xlabel('Number of Observation')
plt.ylabel('GLD')
plt.legend()
plt.show()
```



[17]:
```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
x_train = scaler.fit_transform(x_train)
x_test =  scaler.transform(x_test)
```

[18]:
```
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
lm.fit(x_train,y_train)
```

[18]:
```
LinearRegression()
```

[19]:
```
#Now predicting the test set results
y_pred  =  lm.predict(x_test)
```

[20]:
```
#Now check the error for the regression
from sklearn import metrics
print("MAE :"," ", metrics.mean_absolute_error(y_test,y_pred))
print("MSE :"," ", metrics.mean_squared_error(y_test,y_pred))
print("RMAE :"," ",  np.sqrt(metrics.mean_squared_error(y_test,y_pred)))
```

```
MAE :    6.1122323620399905
MSE :    75.61760337571633
RMAE :    8.69583827906869
```

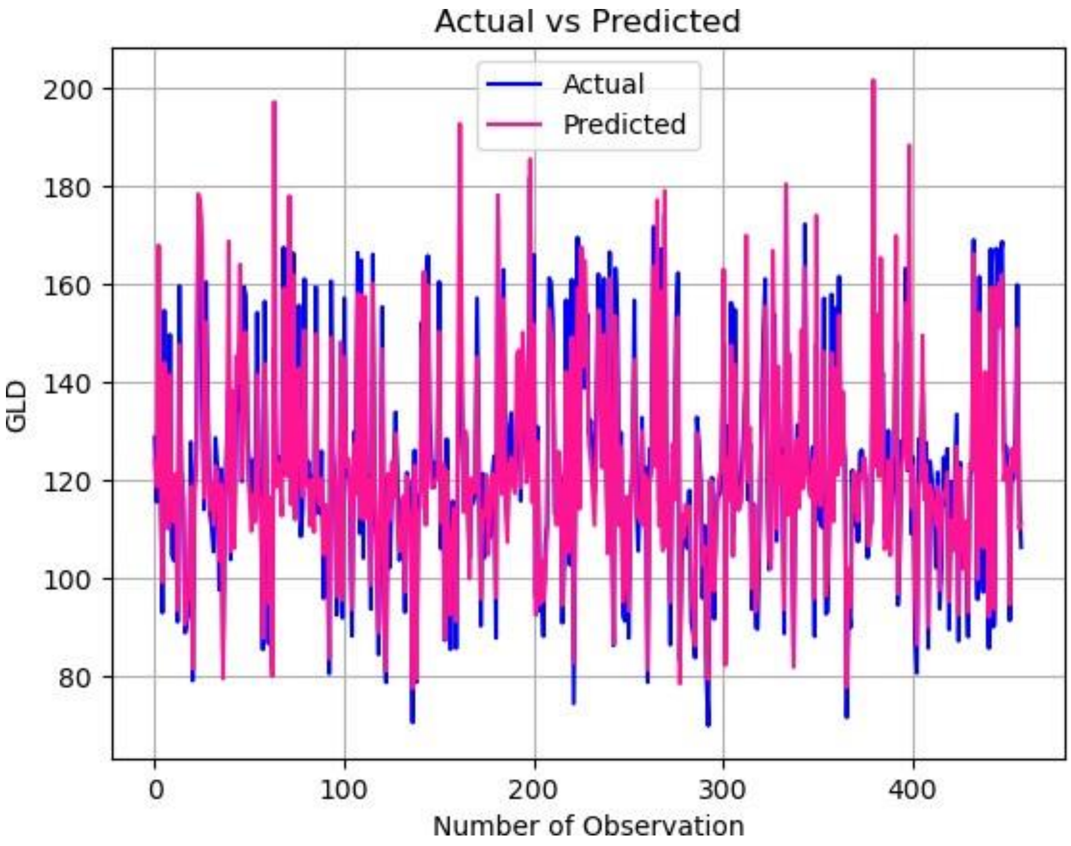
[21]:

```
#Now Lets check the training and test set Accuracy
accuracy_train2 = lm.score(x_train, y_train)
accuracy_test2 = lm.score(x_test, y_test)
print(accuracy_train2)
print(accuracy_test2)
```

```
0.8877758904855643
0.8530125466873725
```

[22]:

```
#Visualising the Accuracy of Predicted result
plt.plot(y_test, color = 'blue', label = 'Actual')
plt.plot(y_pred, color = 'deeppink', label = 'Predicted')
plt.grid(0.3)
plt.title('Actual vs Predicted')
plt.xlabel('Number of Observation')
plt.ylabel('GLD')
plt.legend()
plt.show()
```



[23]:

```
scores = [accuracy_train,accuracy_train2]
algorithms = ["Random forest","Logistic Regression"]

for i in range(len(algorithms)):
    print("The accuracy score "+algorithms[i]+" is: "+str(scores[i])+"%")
```

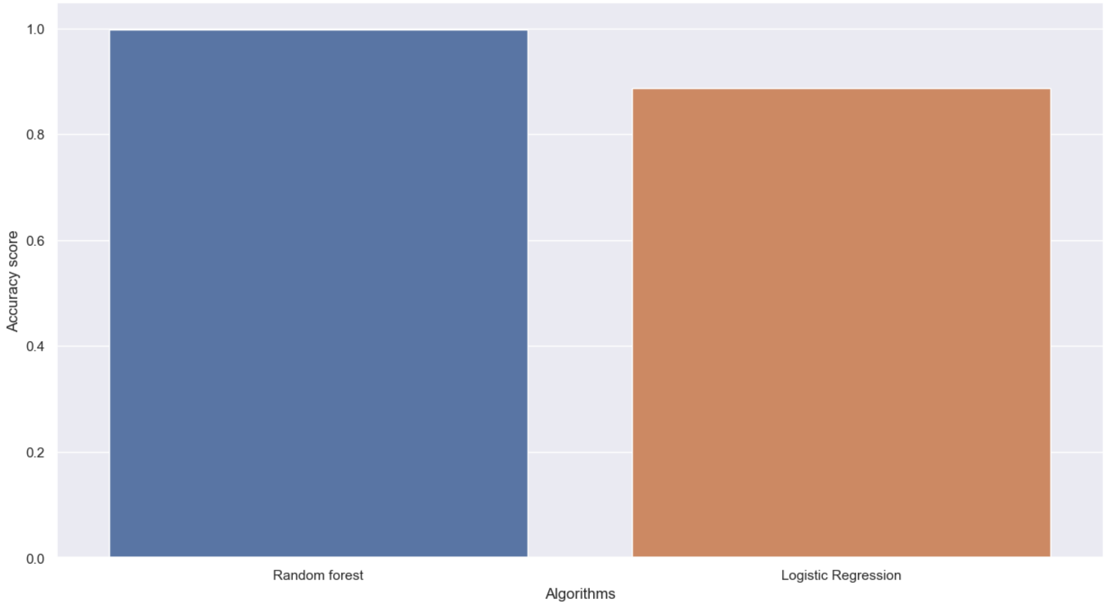
```
The accuracy score Random forest is: 0.9984324726699736%
The accuracy score Logistic Regression is: 0.8877758904855643%
```

[24]:

```
sns.set(rc={'figure.figsize':(15,8)})
plt.xlabel("Algorithms")
plt.ylabel("Accuracy score")
sns.barplot(x=algorithms,y=scores)
```

[24]:

```
<Axes: xlabel='Algorithms', ylabel='Accuracy score'>
```



[25]:

```
pip install gradio
```

```
Requirement already satisfied: gradio in c:\users\admin\anaconda2\lib\site-
packages (3.25.0)
Requirement already satisfied: ffmpy in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (0.3.0)
Requirement already satisfied: pillow in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (9.4.0)
Requirement already satisfied: pandas in c:\users\admin\anaconda2\lib\site-
```

```
packages (from gradio) (1.5.3)
Requirement already satisfied: pydub in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (0.25.1)
Requirement already satisfied: pyyaml in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (6.0)
Requirement already satisfied: semantic-version in
c:\users\admin\anaconda2\lib\site-packages (from gradio) (2.10.0)
Requirement already satisfied: httpx in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (0.24.0)
Requirement already satisfied: websockets>=10.0 in
c:\users\admin\anaconda2\lib\site-packages (from gradio) (11.0.1)
Requirement already satisfied: gradio-client>=0.0.8 in
c:\users\admin\anaconda2\lib\site-packages (from gradio) (0.1.0)
Requirement already satisfied: typing-extensions in
c:\users\admin\anaconda2\lib\site-packages (from gradio) (4.4.0)
Requirement already satisfied: aiohttp in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (3.8.4)
Requirement already satisfied: aiofiles in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (23.1.0)
Requirement already satisfied: markdown-it-py[linkify]>=2.0.0 in
c:\users\admin\anaconda2\lib\site-packages (from gradio) (2.2.0)
Requirement already satisfied: uvicorn in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (0.21.1)
Requirement already satisfied: python-multipart in
c:\users\admin\anaconda2\lib\site-packages (from gradio) (0.0.6)
Requirement already satisfied: markupsafe in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (2.1.1)
Requirement already satisfied: pydantic in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (1.10.7)
Requirement already satisfied: matplotlib in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (3.7.0)
Requirement already satisfied: huggingface-hub>=0.13.0 in
c:\users\admin\anaconda2\lib\site-packages (from gradio) (0.13.4)
Requirement already satisfied: requests in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (2.28.1)
Requirement already satisfied: jinja2 in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (3.1.2)
Requirement already satisfied: orjson in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (3.8.10)
Requirement already satisfied: numpy in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (1.23.5)
Requirement already satisfied: mdit-py-plugins<=0.3.3 in
c:\users\admin\anaconda2\lib\site-packages (from gradio) (0.3.3)
Requirement already satisfied: altair>=4.2.0 in
c:\users\admin\anaconda2\lib\site-packages (from gradio) (4.2.2)
Requirement already satisfied: fastapi in c:\users\admin\anaconda2\lib\site-
packages (from gradio) (0.95.0)
Requirement already satisfied: entrypoints in c:\users\admin\anaconda2\lib\site-
```

```
packages (from altair>=4.2.0->gradio) (0.4)
Requirement already satisfied: toolz in c:\users\admin\anaconda2\lib\site-
packages (from altair>=4.2.0->gradio) (0.12.0)
Requirement already satisfied: jsonschema>=3.0 in
c:\users\admin\anaconda2\lib\site-packages (from altair>=4.2.0->gradio) (4.17.3)
Requirement already satisfied: fsspec in c:\users\admin\anaconda2\lib\site-
packages (from gradio-client>=0.0.8->gradio) (2022.11.0)
Requirement already satisfied: packaging in c:\users\admin\anaconda2\lib\site-
packages (from gradio-client>=0.0.8->gradio) (22.0)
Requirement already satisfied: filelock in c:\users\admin\anaconda2\lib\site-
packages (from huggingface-hub>=0.13.0->gradio) (3.9.0)
Requirement already satisfied: tqdm>=4.42.1 in
c:\users\admin\anaconda2\lib\site-packages (from huggingface-
hub>=0.13.0->gradio) (4.64.1)
Requirement already satisfied: mdurl~=0.1 in c:\users\admin\anaconda2\lib\site-
packages (from markdown-it-py[linkify]>=2.0.0->gradio) (0.1.2)
Requirement already satisfied: linkify-it-py<3,>=1 in
c:\users\admin\anaconda2\lib\site-packages (from markdown-it-
py[linkify]>=2.0.0->gradio) (2.0.0)
Requirement already satisfied: python-dateutil>=2.8.1 in
c:\users\admin\anaconda2\lib\site-packages (from pandas->gradio) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in
c:\users\admin\anaconda2\lib\site-packages (from pandas->gradio) (2022.7)
Requirement already satisfied: async-timeout<5.0,>=4.0.0a3 in
c:\users\admin\anaconda2\lib\site-packages (from aiohttp->gradio) (4.0.2)
Requirement already satisfied: frozenlist>=1.1.1 in
c:\users\admin\anaconda2\lib\site-packages (from aiohttp->gradio) (1.3.3)
Requirement already satisfied: multidict<7.0,>=4.5 in
c:\users\admin\anaconda2\lib\site-packages (from aiohttp->gradio) (6.0.4)
Requirement already satisfied: charset-normalizer<4.0,>=2.0 in
c:\users\admin\anaconda2\lib\site-packages (from aiohttp->gradio) (2.0.4)
Requirement already satisfied: attrs>=17.3.0 in
c:\users\admin\anaconda2\lib\site-packages (from aiohttp->gradio) (22.1.0)
Requirement already satisfied: aiosignal>=1.1.2 in
c:\users\admin\anaconda2\lib\site-packages (from aiohttp->gradio) (1.3.1)
Requirement already satisfied: yarl<2.0,>=1.0 in
c:\users\admin\anaconda2\lib\site-packages (from aiohttp->gradio) (1.8.2)
Requirement already satisfied: starlette<0.27.0,>=0.26.1 in
c:\users\admin\anaconda2\lib\site-packages (from fastapi->gradio) (0.26.1)
Requirement already satisfied: httpcore<0.18.0,>=0.15.0 in
c:\users\admin\anaconda2\lib\site-packages (from httpx->gradio) (0.17.0)
Requirement already satisfied: idna in c:\users\admin\anaconda2\lib\site-
packages (from httpx->gradio) (3.4)
Requirement already satisfied: certifi in c:\users\admin\anaconda2\lib\site-
packages (from httpx->gradio) (2022.12.7)
Requirement already satisfied: sniffio in c:\users\admin\anaconda2\lib\site-
packages (from httpx->gradio) (1.2.0)
Requirement already satisfied: contourpy>=1.0.1 in
```

```
c:\users\admin\anaconda2\lib\site-packages (from matplotlib->gradio) (1.0.5)
Requirement already satisfied: kiwisolver>=1.0.1 in
c:\users\admin\anaconda2\lib\site-packages (from matplotlib->gradio) (1.4.4)
Requirement already satisfied: fonttools>=4.22.0 in
c:\users\admin\anaconda2\lib\site-packages (from matplotlib->gradio) (4.25.0)
Requirement already satisfied: pyparsing>=2.3.1 in
c:\users\admin\anaconda2\lib\site-packages (from matplotlib->gradio) (3.0.9)
Requirement already satisfied: cycler>=0.10 in
c:\users\admin\anaconda2\lib\site-packages (from matplotlib->gradio) (0.11.0)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
c:\users\admin\anaconda2\lib\site-packages (from requests->gradio) (1.26.14)
Requirement already satisfied: click>=7.0 in c:\users\admin\anaconda2\lib\site-
packages (from uvicorn->gradio) (8.0.4)
Requirement already satisfied: h11>=0.8 in c:\users\admin\anaconda2\lib\site-
packages (from uvicorn->gradio) (0.14.0)
Requirement already satisfied: colorama in c:\users\admin\anaconda2\lib\site-
packages (from click>=7.0->uvicorn->gradio) (0.4.6)
Requirement already satisfied: anyio<5.0,>=3.0 in
c:\users\admin\anaconda2\lib\site-packages (from
httpcore<0.18.0,>=0.15.0->httpx->gradio) (3.5.0)
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!=0.17.2,>=0.14.0 in
c:\users\admin\anaconda2\lib\site-packages (from
jsonschema>=3.0->altair>=4.2.0->gradio) (0.18.0)
Requirement already satisfied: uc-micro-py in c:\users\admin\anaconda2\lib\site-
packages (from linkify-it-py<3,>=1->markdown-it-py[linkify]>=2.0.0->gradio)
(1.0.1)
Requirement already satisfied: six>=1.5 in c:\users\admin\anaconda2\lib\site-
packages (from python-dateutil>=2.8.1->pandas->gradio) (1.16.0)
```

[26]:

```
import gradio as gr
```

[27]:

```
def gold_price(SPX,USO,SLV,EUR_USD):
    x  =  np.array([SPX,USO,SLV,EUR_USD])
    x = np.array(x).reshape(1,-1)
    prediction = lm.predict(x)
    prediction = float(prediction)
    return prediction
```

[28]:

```
#Creating user  interface
app = gr.Interface(fn=gold_price,
            inputs = [gr.inputs.Number(label = "SPX"),
                   gr.inputs.Number(label = "USO"),
                   gr.inputs.Number(label = "SLV"),
                   gr.inputs.Number(label = "EUR_USD")
                   ],
                 outputs  = "label",
                 title = "Devolping an ML Model for Gold_Price Prediction"
```

```
                     )
```

```
C:\Users\ADMIN\anaconda2\lib\site-packages\gradio\inputs.py:59: UserWarning:
Usage of gradio.inputs is deprecated, and will not be supported in the future,
please import your component from gradio.components
  warnings.warn(
C:\Users\ADMIN\anaconda2\lib\site-packages\gradio\deprecation.py:40:
UserWarning: `optional` parameter is deprecated, and it has no effect
  warnings.warn(value)
```

[33]:

```
app.launch(show_error=True)
```

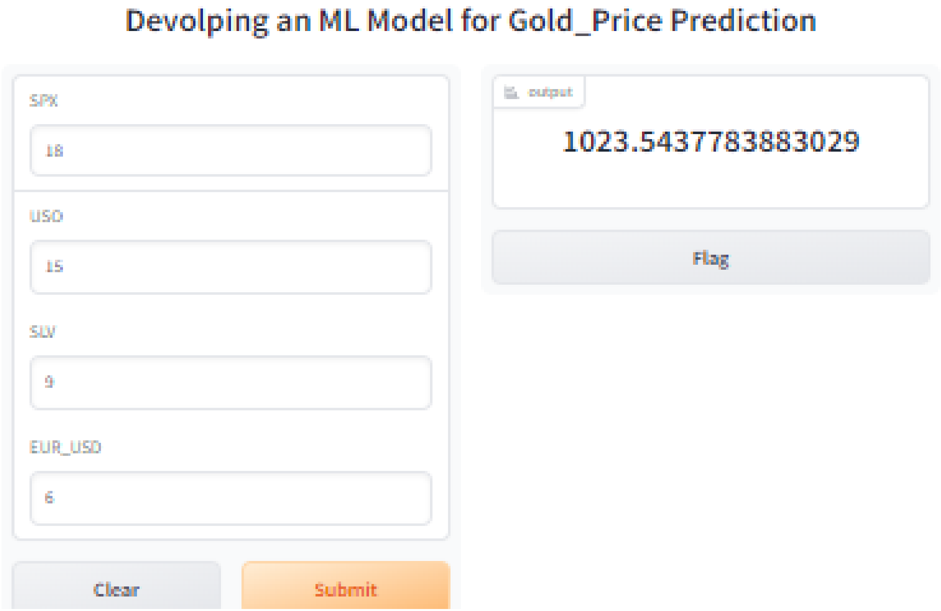
```
Rerunning server... use `close()` to stop if you need to change `launch()`
parameters.
----

Running on local URL:  http://127.0.0.1:7860

To create a public link, set `share=True` in `launch()`.
```

```
<IPython.core.display.HTML object>
```

[33]:



Devolping an ML Model for Gold_Price Prediction

SPX: 18

USO: 15

SLV: 9

EUR_USD: 6

output: 1023.5437783883029

[33]:

[34]:

```
import pickle as pkl

filename = "save_model.sav"
pkl.dump(lm, open(filename, 'wb'))
```

[35]:

```
model_load = pkl.load(open(filename, 'rb'))
```

[32]:

```
model_load.score(x_test,y_test)
```

```
[32]: 0.8530125466873725
```

```
[ ]:
```