MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

Skill Based Mini Project Report

on

Gold Price Prediction Using Python and ML

Submitted By:

Vaibhav Khatri

0901CS201132

Faculty Mentor:

Dr. Ranjeet Kumar Singh

Assistant professor,CSE

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE
GWALIOR - 474005 (MP) est. 1957

MAY-JUNE 2022

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

CERTIFICATE

This is certified that **Vaibhav Khatri** (0901CS201132) has submitted the project report titled **Gold Price Prediction Using Python and ML** under the mentorship of **Dr. Ranjeet Kumar Singh**, in partial fulfilment of the requirement for the award of degree of Bachelor of Technology in Computer Science and Engineering from Madhav Institute of Technology and Science, Gwalior.

R.K. Singh

Dr. Ranjeet Kumar Singh
Faculty Mentor
Assistant professor
Computer Science and Engineering

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

DECLARATION

I hereby declare that the work being presented in this project report, for the partial fulfilment of requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering at Madhav Institute of Technology & Science, Gwalior is an authenticated and original record of my work under the mentorship of **Dr. Ranjeet Kumar Singh**, **Assistant professor**, **Computer Science and Engineering**

I declare that I have not submitted the matter embodied in this report for the award of any degree or diploma anywhere else.

Vaibhav Khatri
0901CS201132
II Year, 4thSem

Computer Science and Engineering

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

ACKNOWLEDGEMENT

The full semester project has proved to be pivotal to my career. I am thankful to my institute, **Madhav Institute of Technology and Science** to allow me to continue my disciplinary/interdisciplinary project as a curriculum requirement, under the provisions of the Flexible Curriculum Scheme (based on the AICTE Model Curriculum 2018), approved by the Academic Council of the institute. I extend my gratitude to the Director of the institute, **Dr. R. K. Pandit** and Dean Academics, **Dr. Manjaree Pandit** for this.

I would sincerely like to thank my department, **Department of Computer Science and Engineering, for allowing** me to explore this project. I humbly thank **Dr. Manish Dixit**, Professor and Head, Department of Computer Science and Engineering, for his continued support during the course of this engagement, which eased the process and formalities involved.

I am sincerely thankful to my faculty mentors. I am grateful to the guidance of **Dr. Ranjeet Kumar Singh**,Assistant Professor, Computer Science and Engineering, for his continued support and guidance throughout the project. I am also very thankful to the faculty and staff of the department.

Vaibhav Khatri
0901CS201132
II Year, 4th Sem

Computer Science and Engineering

ABSTRACT

The Project titled 'GOLD PRICE PREDICTION' predicts the gold EFT price based on the previous year's gold price data. The main goal of this project is to forecast the rise and fall in the daily gold rates, that can help investors to decide when to buy or sell the gold. Inventory forecasting plays a crucial role in the financial success of the business. The price of gold is calculated by looking at the dataset that contains the previous year's gold price. Rise in gold value coupled with volatility and falling prices from other markets such as capital markets and real estate markets has attracted more and more investors to gold as an attractive investment. There's a fear that those high prices will be sustainable and that the prices will reverse. Although there are a number of studies that analyze the correlation between the gold price and certain economic variables. We have applied machine learning technique to predict financial variables and we have focused on predicting the gold price ETF using a linear regression algorithm as our dataset is a numerical dataset

Keyword: Gold ETF, Machine Learning, Supervised Learning, Linear Regression, Python.

TABLE OF CONTENTS

Chapter 1: INTRODUCTION

Gold is one of the precious metals. It has been used as currency, for jewellery and other purposes. It is used as medium for money or exchange because of its limited supply and high value. This metal's scarcity and difficulty in extraction made it a valuable commodity. It also reflects the country's economic strength and hence many companies and individuals started to invest in gold reserves. Due to its increasing value, many people considered gold as an attractive investment.

Gold is preferred as protective asset by investors because of their negative expectations regarding the current situation in the foreign exchange and capital markets. Investors also consider gold as an asset to rely on, when the desirable profits are not achieved by the world capital markets. Since gold is stored and accumulated over years, the influence of a year's production on its price is less. The price of gold depends on currency fluctuations and other economic variables. The raise of gold prices and fall of prices in other markets has attracted more investors to invest in gold market. These changes in the price of gold made the investments risky and a fear has been developed that these prices would decrease.

There are several numbers of studies analysing the relation between the gold price and other economic variables. Understanding the relation between these variables helps the investors to take better decisions. Hence, we use machine learning algorithms such as multiple linear regression, random forest and gradient boosting for analysing the relation between the variables and predict the gold price.

Chapter 2: Project Objective

Here we proposed predictive models that are adaptive, flexible, and scalable, using the advantages of proposed computationally smart neural network models to enhance the training learning process and enhance faster convergence. The proposed research provides the highest likelihood of achieving high training rate prediction precision for the considered gold EFT price. Generally speaking, this work is performed to suggest suitable predictor models to effectively show the deemed gold in the different scenario with the datasets deemed from their respective databases of previous years. This present's aim is to present correctly the future modified closing price of Gold ETF in the future for a specified period of time. In this project, supervised Machine Learning Algorithms and the solution model were used to determine whether or not to buy Gold ETF using a dataset of past values.

The main objectives of the project are:

1. This project is based on the applicability of the proposed machine learning algorithms that had demonstrated their efficiency to predict gold prices with a better predictive rate.
2. To apply the best appropriate Machine Learning procedure.
3. We proposed the development of a prediction model for predicting future gold prices using Linear Regression (LR).

Chapter 3: DATA AND METHODOLOGY

A. Dataset:

The data was sourced from the kaggle website consisting of ten years data from January 2008 to Decmeber 2018. It consists of the variables date, silver price, stock profit exchange, gold price ,US dollar rate and united states oil ETF. The dataset consists of 2290 records.

B. Machine Learning Algorithms:

For developing the model, we use the algorithms such as Multiple linear regression, Random forest and Gradient boosting.

Multiple Linear Regression is a statistical technique that uses multiple independent variables for predicting the outcome of the target variable. The mathematical expression is

$$Y = a + bX1 + cX2 + dX3 + \epsilon$$

where Y is the dependent variable, X1 and X2 and X3 are independent variables, a is the intercept, b ,c and d are slope values and ϵ is the error.

Random forest is a supervised learning algorithm which performs both classification and regression tasks. This algorithm operates by constructing multiple decision trees during training time and outputting the mean prediction of individual trees.

Gradient Boosting is a machine learning algorithm for classificiation and regression problems. This algorithm produces a prediction model in the form of an ensemble of weak prediction models, which are typically the decision trees. It builds the model in stage-wise fashion and generalizes them by enabling the optimization of an arbitary differentiable loss function. The gradient boosting trees usually outperforms random forest, but are prone to overfitting in some problems as the performance of this model improves over iterations.

Chapter 4: IMPLEMENTATION

To predict the gold price, we need to build a machine learning model which includes the following steps.

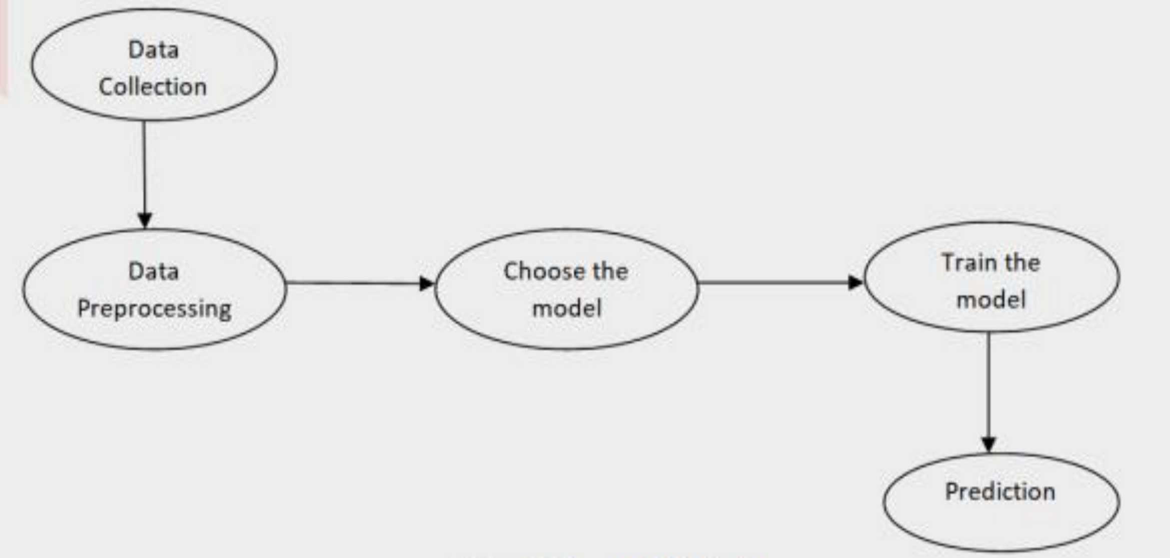


Figure 1 Proposed Method

1. Data Collection:

The first thing required while building a machine learning model is the data. The data is collected from kaggle website consisting of 2290 records and 6 attributes.

2. Data Preprocessing:

Data preprocessing is required when the data is incomplete, inconsistent or noisy. The data collected was noisy, so we performed outlier analysis and removed the noisy data. The data transformation is also done by performing normalization in which the data in each attribute is scaled between the range 0 to 1.

3. Choose the model:

Prediction of gold price is a regression task, so we consider the regression algorithms such as Multiple Linear Regression, Random Forest Regressor and Gradient Boosting for building the model.

4. Training the model: The model is trained by importing the required model and by passing the training data to it. The dataset is splitted into train and test data with test_size=0.20. The linear model is imported from

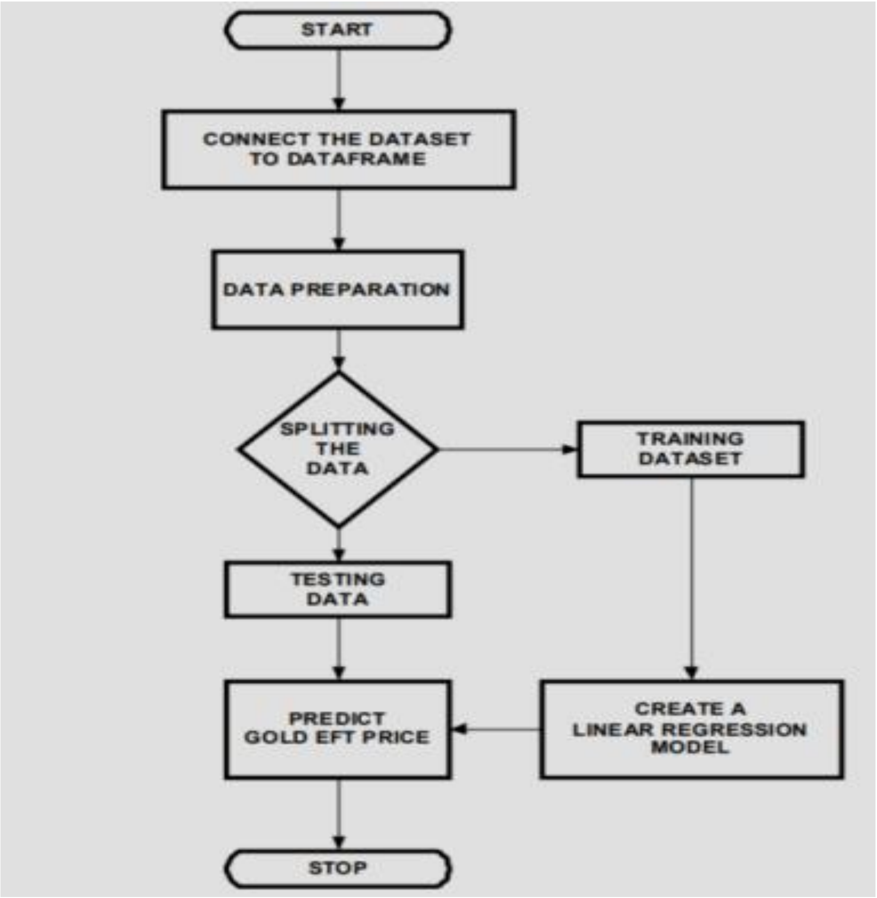
5.

sklearn and the Random forest regressor and Gradient boosting regressor modules are imported from sklearn.ensemble. These models are trained by passing the train data.

While conducting training, it is also important to record the metrics of each training process. The metrics that are tested are mean absolute error, root mean square error and r2 score.

5. Prediction: The trained model is checked by predicting the test data of the dependent variable using the test data of the independent variables.

5: Working Procedure



5.1 ALGORITHM

STEP 1: Gathering the data from y-finances library and preparing the data by removing the missing values

STEP 2: Now we split the gathered data into training and testing dataset.

STEP 3: Now using training data we create a linear regression model.

STEP 4: Using the testing data we test the created linear regression model.

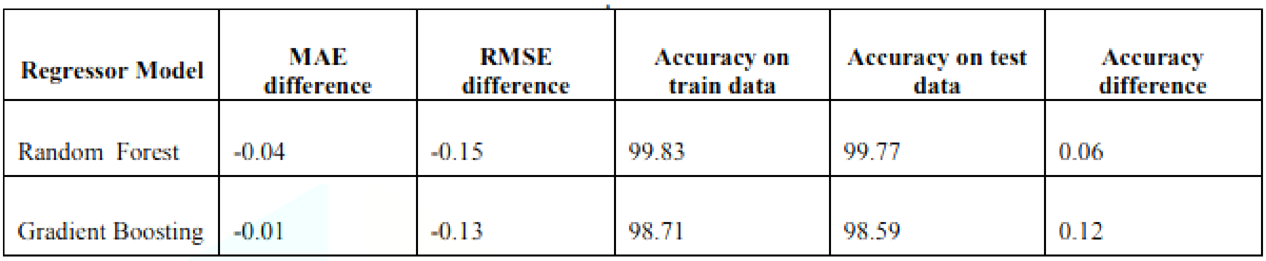
STEP 5: Using the model now we predict daily gold ETF price.

5.2 Linear Regression

Linear regression in machine learning helps you find out patterns and relationships in data and make an educated decision or prediction. It is one of the most well-known and well understood algorithms in statistics and machine learning. But before knowing that -What linear regression actually is, let us get ourselves accustomed to regression. Regression is a method of modeling a target value based on independent predictions. This method is used for forecasting and finding out the cause and efficient relationship between variables. Usually, the regression techniques mostly differ based on the number of independent variables and the types of relationships between the independent and dependent variables. Simple linear regression is a type of regression analysis where the number of independent variables is one and there is a linear relationship between the independent(x) and dependent(y) variables.Based on the given data points, we try to plot a line that models the points the best. The line can be modeled based on the linear regression equation which is y = a_0 + a_1 * x.

Chapter 6: Result

After applying different regression techniques on the data, the results are as follows: When multiple regression is applied on the data, the accuracy (r2 score) obtained is 91% and RMSE is 5.56 which is high.



Regressor Model	MAE difference	RMSE difference	Accuracy on train data	Accuracy on test data	Accuracy difference
Random Forest	-0.04	-0.15	99.83	99.77	0.06
Gradient Boosting	-0.01	-0.13	98.71	98.59	0.12

In random forest model, the accuracy obtained for training data is 99.83% and the accuracy obtained for test data is 99.77%. The accuracy difference is very less. The RMSE (-0.15) and MAE (-0.04) have only slight differences between the train and test data. For gradient boosting model, the accuracy obtained for training data is 98.71% and the accuracy obtained for test data is 98.59%. The accuracy difference is very less. The RMSE (-0.13) and MAE (0.01) have only slight differences between the train and test data.

Hence Random Forest and Gradient Boosting best suited to this data. However, the accuracy is higher for random forest and the accuracy difference is also very less compared to gradient boosting. Hence, random forest regressor model is considered.

Chapter 7: CONCLUSION

7.1 CONCLUSION

As we saw in this project, we'll create a machine learning linear regression model. We first train this machine learning model by giving information from past gold ETF prices. Then we use this trained model for prediction. Similarly, any model can be made much more precise by feeding a very large dataset to get a very accurate score. While forecasting the rate of gold is not very easy, it will allow investors and central banks to determine better when to sell and buy them and thus maximize their income

7.2 FUTURE SCOPE

For future work, we can improve the results and predict the price more accurately by incorporating the other factors such as gold production, crude oil price, platinum price,inflation to the data and by using deep learning.

REFRENCES

[1] Xiaohui Yang, "The Prediction of Gold Price Using ARIMA Model", 2nd International Conference on Social Science, Public Health and Education 2019.

[2] Manjula K. A., Karthikeyan P, "Gold Price Prediction using Ensemble based Machine Learning Techniques", Third International Conference on Trends in Electronics and Informatics, 2019

[3] R. Hafezi*, A. N. Akhavan, "Forecasting Gold Price Changes: Application of an Equipped Artificial Neural Network", AUT Journal of Modeling and Simulation, 2018.

[4] Shian-Chang Huang and Cheng-Feng Wu, Energy Commodity Price Forecasting with Deep Multiple Kernel Learning, MDPI Journal, 2018.

[5] Wedad Ahmed Al-Dhuraibi and Jauhar Ali, "Using Classification Techniques to Predict Gold Price Movement", 4th International Conference on Computer and Technology Applications, 2018.

Appendices

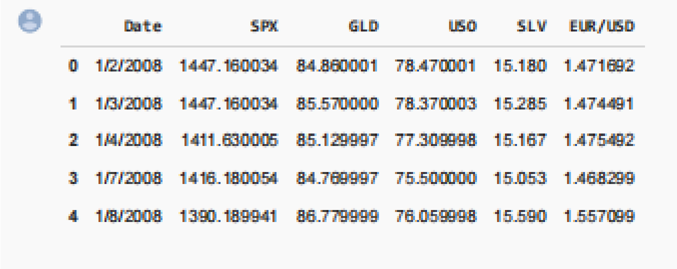
Importing the Libraries

```
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

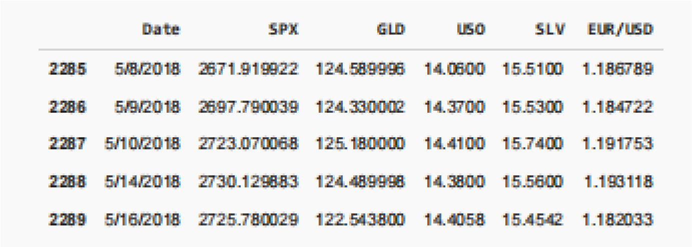
```
# loading the csv data to a Pandas DataFrame

gold_data = pd.read_csv('/content/gold price dataset.csv')
```

```
# print first 5 rows in the dataframe

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

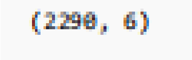
```
# print last 5 rows of the dataframe

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

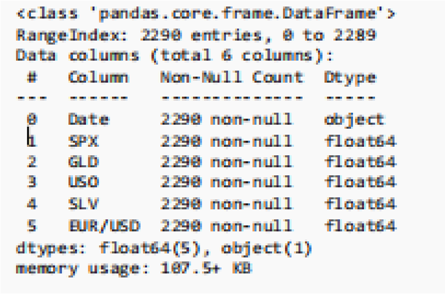
gold_data.shape



```
(2290, 6)
```

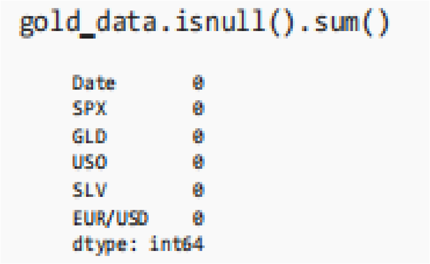
# getting some basic informations about the data

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
```

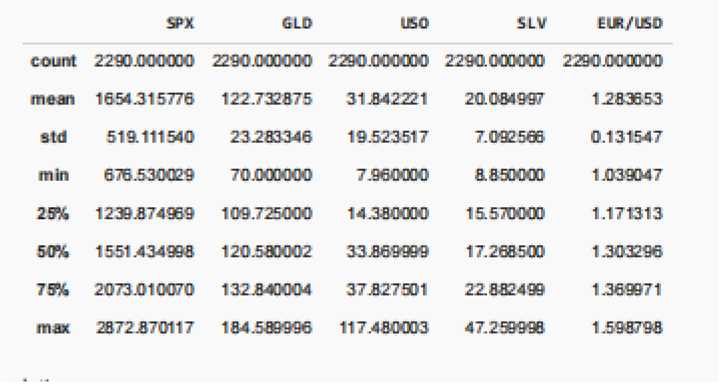
# checking the number of missing values



```
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()



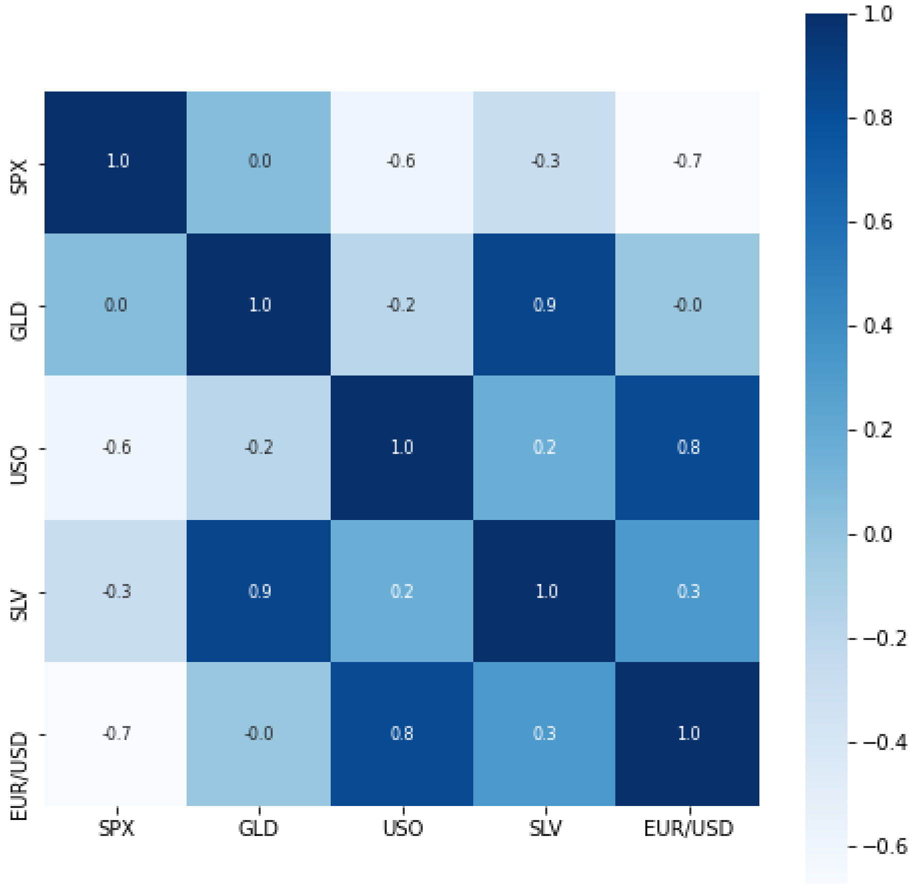
	SPX	GLD	USO	SLV	EUR/USD
count	2290.000000	2290.000000	2290.000000	2290.000000	2290.000000
mean	1654.315776	122.732875	31.842221	20.084997	1.283653
std	519.111540	23.283346	19.523517	7.092566	0.131547
min	676.530029	70.000000	7.960000	8.850000	1.039047
25%	1239.874969	109.725000	14.380000	15.570000	1.171313
50%	1551.434998	120.580002	33.869999	17.268500	1.303296
75%	2073.010070	132.840004	37.827501	22.882499	1.369971
max	2872.870117	184.589996	117.480003	47.259998	1.598798

Correlation:

1. Positive Correlation

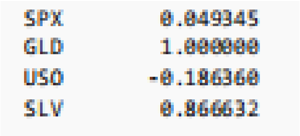
2. Negative Correlation

```
correlation = gold_data.corr()
# constructing a heatmap to understand the correlatiom
plt.figure(figsize = (8,8))
sns.heatmap(correlation, cbar=True, square=True, fmt='.1f',annot=True, annot_kws={'size
<matplotlib.axes._subplots.AxesSubplot at 0x7ff32443b350>
```

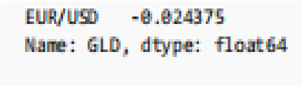


# correlation values of GLD

print(correlation['GLD'])



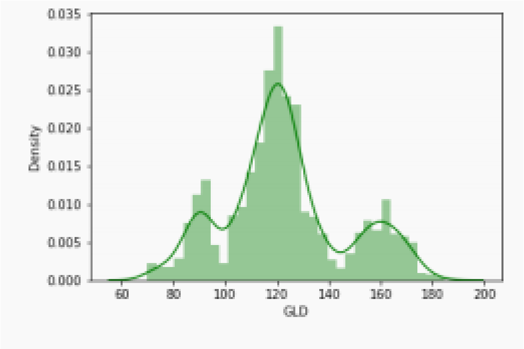
```
SPX          0.049345
GLD          1.000000
USO         -0.186360
SLV          0.866632
```



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

# checking the distribution of the GLD Price

sns.distplot(gold_data['GLD'],color='green')

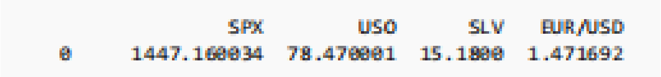


Splitting the Features and Target

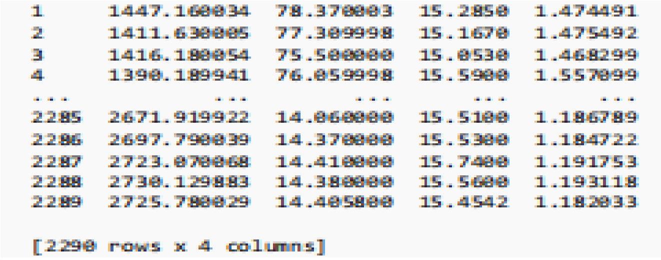
X = gold_data.drop(['Date','GLD'],axis=1)

Y = gold_data['GLD']

print(X)

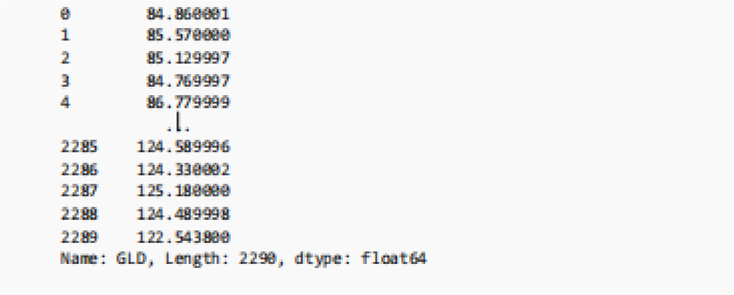


```
                SPX         USO       SLV   EUR/USD
0       1447.160034   78.470001   15.1800  1.471692
```



```
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
            ...
2285    124.589996
2286    124.330002
2287    125.180000
2288    124.489998
2289    122.543800
Name: GLD, Length: 2290, dtype: float64
```

Splitting into Training data and Test Data

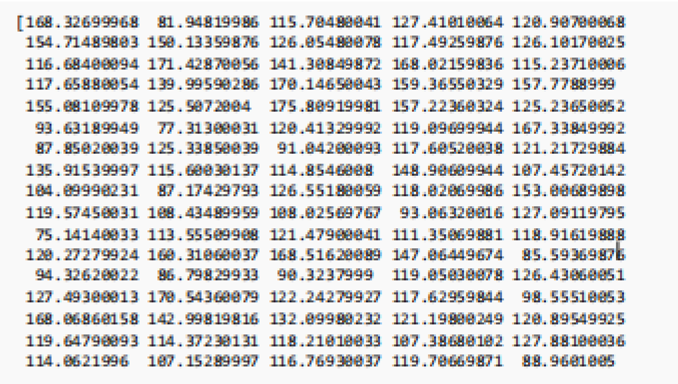
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state
```

Model Training: Random Forest Regressor

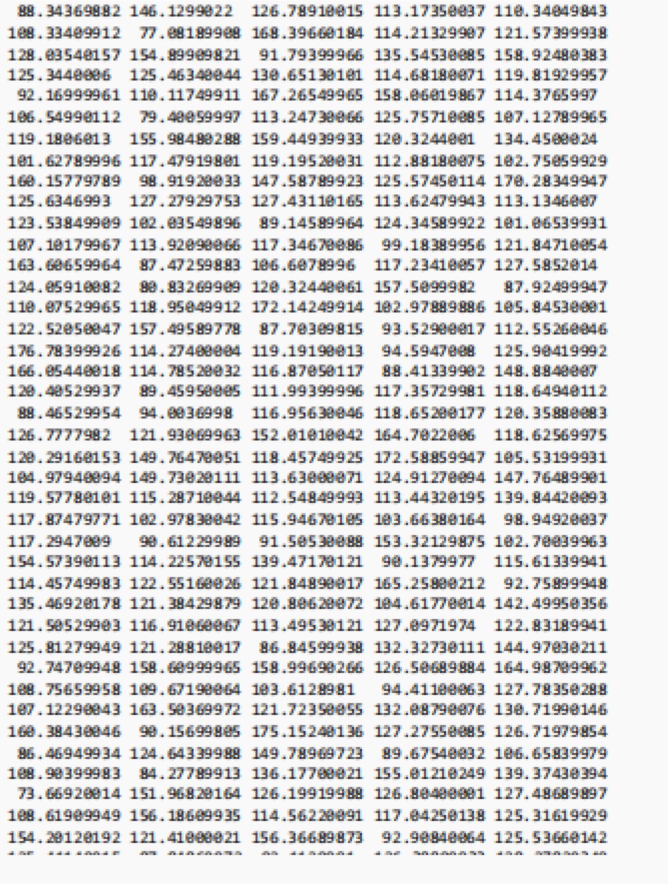
```
regressor = RandomForestRegressor(n_estimators=100)

# training the model
regressor.fit(X_train,Y_train)

RandomForestRegressor(bootstrap=True, ccp_alpha=0.0, criterion='mse',
max_depth=None, max_features='auto', max_leaf_nodes=None,
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)
```



```
[168.32699968  81.94819986 115.70480041 127.41010064 120.90700068
 154.71489803 150.13359876 126.05480078 117.49259876 126.10170025
 116.68400094 171.42870056 141.30849872 168.02159836 115.23710006
 117.65880054 139.99590286 170.14650043 159.36550329 157.7788999
 155.08109978 125.5072004  175.80919981 157.22360324 125.23650052
  93.63189949  77.31300031 120.41329992 119.09699944 167.33849992
  87.85020039 125.33850039  91.04200093 117.60520038 121.21729884
 135.91539997 115.60030137 114.8546008  148.90609944 107.45720142
 104.09990231  87.17429793 126.55180059 118.02069986 153.00689898
 119.57450031 108.43489959 108.02569767  93.06320016 127.09119795
  75.14140033 113.55509908 121.47900041 111.35069881 118.91619888
 120.27279924 160.31060037 168.51620089 147.06449674  85.59369876
  94.32620022  86.79829933  90.3237999  119.05030078 126.43060051
 127.49300013 170.54360079 122.24279927 117.62959844  98.55510053
 168.06860158 142.99819816 132.09980232 121.19800249 120.89549925
 119.64790093 114.37230131 118.21010033 107.38680102 127.88100036
 114.0621996  107.15289997 116.76930037 119.70669871  88.9601005
```



```
  88.34369882 146.1299022  126.78910015 113.17350037 110.34049843
 108.33409912  77.08189908 168.39660184 114.21329907 121.57399938
 128.03540157 154.89909821  91.79399966 135.54530085 158.92480383
 125.3440006  125.46340044 130.65130101 114.68180071 119.81929957
  92.16999961 110.11749911 167.26549965 158.06019867 114.3765997
 106.54990112  79.40059997 113.24730066 125.75710085 107.12789965
 119.1806013  155.98480288 159.44939933 120.3244001  134.4500024
 101.62789996 117.47919801 119.19520031 112.88180075 102.75059929
 160.15779789  98.91920033 147.58789923 125.57450114 170.28349947
 125.6346993  127.27929753 127.43110165 113.62479943 113.1346007
 123.53849909 102.03549896  89.14589964 124.34589922 101.06539931
 107.10179967 113.92090066 117.34670086  99.18389956 121.84710054
 163.60659964  87.47259883 106.6078996  117.23410057 127.5852014
 124.05910082  80.83269909 120.32440061 157.5099982   87.92499947
 110.07529965 118.95049912 172.14249914 102.97889886 105.84530001
 122.52050047 157.49589778  87.70309815  93.52900017 112.55260046
 176.78399926 114.27400004 119.19190013  94.5947008  125.90419992
 166.05440018 114.78520032 116.87050117  88.41339902 148.8840007
 120.40529937  89.45950005 111.99399996 117.35729981 118.64940112
  88.46529954  94.0036998  116.95630046 118.65200177 120.35880083
 126.7777982  121.93069963 152.01010042 164.7022006  118.62569975
 120.29160153 149.76470051 118.45749925 172.58859947 105.53199931
 104.97940094 149.73020111 113.63000071 124.91270094 147.76489901
 119.57780101 115.28710044 112.54849993 113.44320195 139.84420093
 117.87479771 102.97830042 115.94670105 103.66380164  98.94920037
 117.2947009   90.61229989  91.50530088 153.32129875 102.70039963
 154.57390113 114.22570155 139.47170121  90.1379977  115.61339941
 114.45749983 122.55160026 121.84890017 165.25800212  92.75899948
 135.46920178 121.38429879 120.80620072 104.61770014 142.49950356
 121.50529903 116.91060067 113.49530121 127.0971974  122.83189941
 125.81279949 121.28810017  86.84599938 132.32730111 144.97030211
  92.74709948 158.60999965 158.99690266 126.50689884 164.98709962
 108.75659958 109.67190064 103.6128981   94.41100063 127.78350288
 107.12290043 163.50369972 121.72350055 132.08790076 130.71990146
 160.38430046  90.15699805 175.15240136 127.27550085 126.71979854
  86.46949934 124.64339988 149.78969723  89.67540032 106.65839979
 108.90399983  84.27789913 136.17700021 155.01210249 139.37430394
  73.66920014 151.96820164 126.19919988 126.80400001 127.48689897
 108.61909949 156.18609935 114.56220091 117.04250138 125.31619929
 154.20120192 121.41000021 156.36689873  92.90840064 125.53660142
```

# R squared error

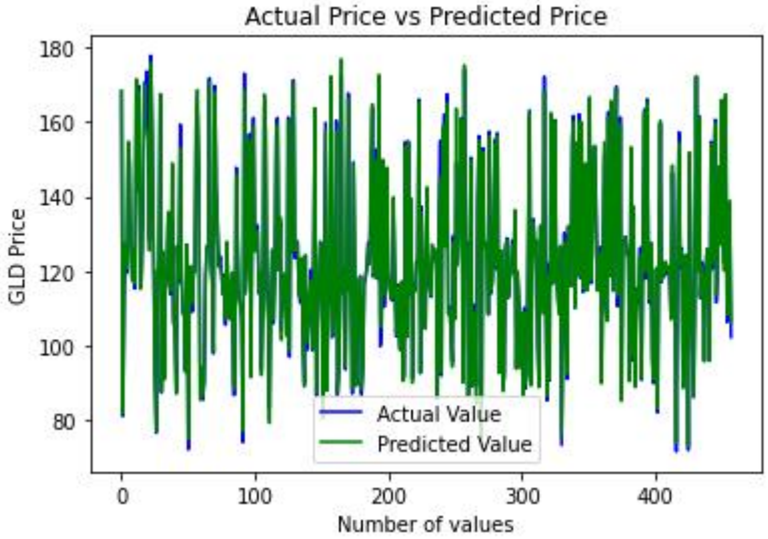
error_score = metrics.r2_score(Y_test, test_data_prediction)

print("R squared error : ", error_score)

R squared error : 0.9887338861925125

Compare the Actual Values and Predicted Values in a Plot

```
Y_test = list(Y_test)

plt.plot(Y_test, color='blue', label = 'Actual Value')
plt.plot(test_data_prediction, color='green', label='Predicted Value')
plt.title('Actual Price vs Predicted Price')
plt.xlabel('Number of values')
plt.ylabel('GLD Price')
plt.legend()
plt.show()
```


Actual Price vs Predicted Price
GLD Price
Actual Value
Predicted Value
Number of values