Assignment 2

Aim:- The aim of this assignment is to explore linear regression and its application in predicting numerical outcomes.

Theory:- Linear regression is a supervised learning algorithm used to predict numerical outcomes based on a set of input features. It works by fitting a line to the data that minimizes the sum of the squared errors between the predicted and actual values. The slope and intercept of the line represent the coefficients of the linear regression model.

The key concepts covered in this assignment would include:

* Understanding the difference between simple linear regression and multiple linear regression.
* Learning how to measure the performance of a linear regression model using metrics like R-squared and Mean Squared Error.
* Understanding the concept of regularization and how it can be used to prevent overfitting.

Case study:- The dataset considered for the study is named ‘gld\_price\_data’. It has 2,290 rows and 6 columns. It contains data regarding prices of Gold, Silver, Euro and USD. The aim is to predict the price of Gold given the prices of Silver, Euro, U.S $.

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

data = pd.read\_csv("C:/Users/Lenovo/Desktop/College/Sem 6/ML Ass/gld\_price\_data.csv")

data.shape (2290, 6)

print(data.head) print(data.dtypes)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| <bound USO | method NDFrame.head of SLV EUR/USD | | Date | | SPX | GLD |
| 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]> Date object

SPX float64

GLD float64

USO float64

SLV float64

EUR/USD float64 dtype: object

data.isnull().sum() Date 0

SPX 0

GLD 0

USO 0

SLV 0

EUR/USD 0

dtype: int64 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 | 1.303297 |
| 75% | 2073.010070 132.840004 | 37.827501 | 22.882500 | 1.369971 |
| max | 2872.870117 184.589996 | 117.480003 | 47.259998 | 1.598798 |

*#Checking different percentiles* pd.DataFrame(data['USO']).describe(percentiles=(1,0.99,0.9,0.75,0.5,0. 3,0.1,0.01))

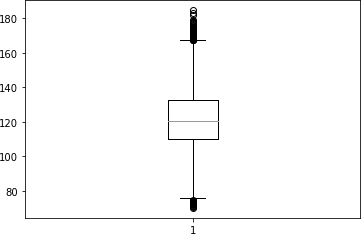
USO

|  |  |
| --- | --- |
| count | 2290.000000 |
| mean | 31.842221 |
| std | 19.523517 |
| min | 7.960000 |
| 1% | 8.927800 |
| 10% | 10.568000 |
| 30% | 18.650000 |
| 50% | 33.869999 |
| 75% | 37.827501 |
| 90% | 41.333002 |
| 99% | 106.696399 |
| 100% | 117.480003 |
| max | 117.480003 |

*#checking boxplot for Age column*

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

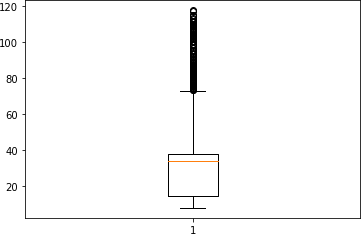
%matplotlib inline plt.boxplot(data['GLD']) plt.show()



*#checking boxplot for Age column*

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

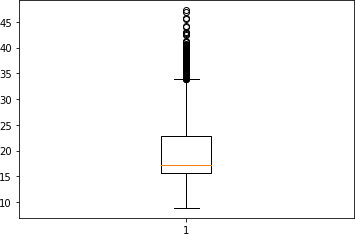
%matplotlib inline plt.boxplot(data['USO']) plt.show()



*#checking boxplot for Age column*

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

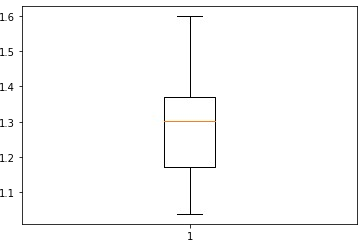
%matplotlib inline plt.boxplot(data['SLV']) plt.show()



*#checking boxplot for Age column*

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

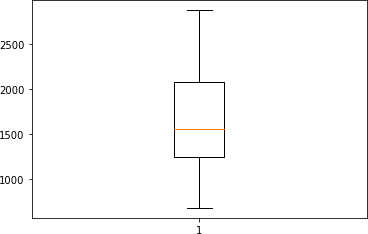
%matplotlib inline plt.boxplot(data['EUR/USD']) plt.show()



*#checking boxplot for Age column*

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

%matplotlib inline plt.boxplot(data['SPX']) plt.show()



*#Checking Outlier by definition and treating outliers*

*#getting median Age*

GLD\_col\_df = pd.DataFrame(data['GLD']) GLD\_median = GLD\_col\_df.median()

*#getting IQR of Age column*

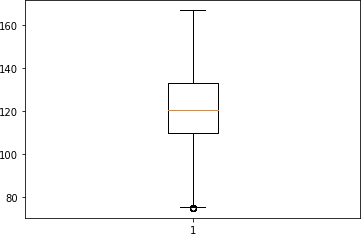
Q3 = GLD\_col\_df.quantile(q=0.75) Q1 = GLD\_col\_df.quantile(q=0.25)

IQR = Q3-Q1

*#Deriving boundaries of Outliers* IQR\_LL = int(Q1 - 1.5\*IQR) IQR\_UL = int(Q3 + 1.5\*IQR)

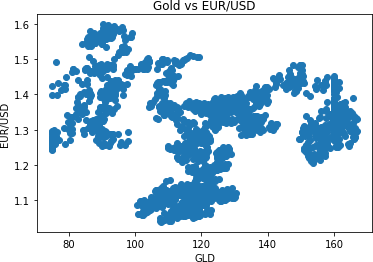
*#Finding and treating outliers - both lower and upper end* data.loc[data['GLD']>IQR\_UL , 'GLD'] = int(GLD\_col\_df.quantile(q=0.90)) data.loc[data['GLD']<IQR\_LL , 'GLD'] = int(GLD\_col\_df.quantile(q=0.01))

*#Check max age value now* max(data['GLD']) plt.boxplot(data['GLD']) plt.show()



x = data["GLD"]

y= data["EUR/USD"] plt.scatter(x, y) plt.title('Gold vs EUR/USD') plt.xlabel('GLD') plt.ylabel('EUR/USD') plt.show()

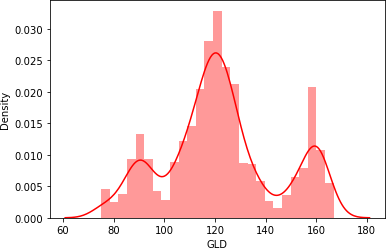


**import** seaborn **as** sns sns.distplot(data['GLD'], color='red')

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\ distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).

warnings.warn(msg, FutureWarning)

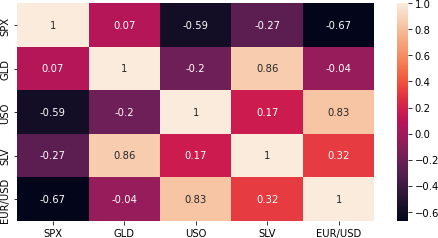
<AxesSubplot:xlabel='GLD', ylabel='Density'>



*#check correltion matrix - to check the strength of variation bwtween two variables*

correlation\_matrix= data.corr().round(2) fgr, ax = plt.subplots(figsize =(8, 4)) **import** seaborn **as** sns

c = sns.heatmap(data=correlation\_matrix, annot=True) fgr.savefig("myimage.png")



X = data.drop(['Date', 'GLD'], axis=1) *#Features*

Y = data['GLD'] *#Target*

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

**from** sklearn.linear\_model **import** LinearRegression

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

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

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y,test\_size = 0.1, random\_state = 0)

model = LinearRegression() model.fit(X\_train, Y\_train)

accuracy = model.score(X\_test,Y\_test) print(accuracy\*100,'%')

83.33354918388221 %

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

y\_pred = model.predict(X\_test)

print('R2\_score : ',metrics.r2\_score(y\_pred,Y\_test)) print('MSE : ',metrics.mean\_squared\_error(y\_pred,Y\_test))

R2\_score : 0.8236381589294605

MSE : 79.28293640338522

X\_test *#the test data - predictors*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | SPX | USO | SLV | EUR/USD |
| 1255 | 1692.770020 | 36.880001 | 20.980000 | 1.351698 |
| 2100 | 2425.530029 | 9.270000 | 14.990000 | 1.147394 |
| 711 | 1309.660034 | 42.049999 | 36.119999 | 1.410596 |
| 1328 | 1838.699951 | 33.689999 | 19.510000 | 1.361804 |
| 53 | 1298.420044 | 82.290001 | 18.250999 | 1.563893 |
| ... | ... | ... | ... | ... |
| 1827 | 2066.659912 | 9.600000 | 14.340000 | 1.140498 |
| 746 | 1316.280029 | 39.279999 | 35.770000 | 1.409900 |
| 962 | 1338.349976 | 35.610001 | 27.420000 | 1.283203 |
| 703 | 1296.390015 | 40.910000 | 35.009998 | 1.393456 |
| 1146 | 1560.699951 | 33.599998 | 27.799999 | 1.301236 |

[229 rows x 4 columns]

Y\_test *#the actual values in test data - target column*

|  |  |
| --- | --- |
| 1255 | 128.789993 |
| 2100 | 115.620003 |
| 711 | 139.220001 |
| 1328 | 120.930000 |
| 53 | 93.040001 |
|  | ... |
| 1827 | 116.940002 |
| 746 | 148.589996 |

|  |  |
| --- | --- |
| 962 | 151.330002 |
| 703 | 138.860001 |
| 1146 | 154.000000 |

Name: GLD, Length: 229, dtype: float64 y\_pred

|  |  |  |  |
| --- | --- | --- | --- |
| array([123.35331704, | 119.18599536, | 164.36402287, | 120.65019774, |
| 99.32394587, | 141.9060249 , | 124.6855032 , | 109.91047834, |
| 139.58298859, | 113.20660895, | 112.07380269, | 121.42430603, |
| 94.10114749, | 145.27505014, | 122.976612 , | 116.33197208, |
| 91.35406032, | 95.8308489 , | 96.87223714, | 118.60835987, |
| 82.6145119 , | 118.04205769, | 122.41090392, | 174.22466353, |
| 172.46016873, | 164.75568691, | 116.74022561, | 149.68225406, |
| 126.68549616, | 114.35090573, | 119.77474914, | 111.33477599, |
| 122.56239537, | 122.02995765, | 103.83263203, | 118.96785316, |
| 80.71992464, | 116.73577287, | 122.46195605, | 165.11227343, |
| 105.52126712, | 136.56264706, | 106.18546221, | 143.12398937, |
| 138.067021 , | 160.76649987, | 120.87455343, | 147.17513425, |
| 147.68892033, | 123.00962034, | 116.44289364, | 109.41781661, |
| 117.12227205, | 111.14466862, | 139.59885451, | 116.29611705, |
| 115.43754832, | 88.39798234, | 141.76072786, | 121.02626917, |
| 95.4924892 , | 116.74077559, | 81.18500389, | 191.87569815, |
| 118.50396608, | 122.83802011, | 121.14248752, | 112.36313841, |
| 156.15969383, | 123.54797999, | 121.29901229, | 173.81477412, |
| 114.51689967, | 158.74247796, | 111.65683718, | 128.04699074, |
| 140.87664925, | 118.99087147, | 116.86341941, | 148.26575127, |
| 118.93831865, | 119.48443112, | 110.45865473, | 119.91560476, |
| 109.28383677, | 147.51884729, | 118.01639847, | 115.4490512 , |
| 120.11935179, | 104.69328043, | 115.35773295, | 112.28614588, |
| 84.80008621, | 146.84237985, | 124.21120819, | 118.8209946 , |
| 96.55093867, | 100.6014964 , | 145.79346224, | 95.97390302, |
| 142.95773085, | 122.89430534, | 124.24770161, | 116.39234537, |
| 94.40052371, | 126.64161971, | 116.97358153, | 155.09968158, |
| 112.88969959, | 154.8937577 , | 112.53383396, | 154.56801816, |
| 112.67055445, | 119.67121275, | 98.76657011, | 157.17925852, |
| 121.22923374, | 105.8928713 , | 89.82456313, | 118.46879435, |
| 144.69381947, | 91.81478744, | 82.35591678, | 121.66912268, |
| 112.49199422, | 123.80756201, | 115.96518731, | 128.54676502, |
| 111.71646849, | 105.56298422, | 105.9517089 , | 117.12681889, |
| 97.55395782, | 121.37985263, | 114.910814 , | 108.8529887 , |
| 78.8976093 , | 123.50085139, | 80.67340794, | 114.08754462, |
| 114.86888524, | 143.63280942, | 159.3135827 , | 110.63680256, |
| 156.88107428, | 126.34646274, | 118.35438236, | 119.52738658, |
| 120.46818007, | 121.14604121, | 148.01979507, | 112.11532785, |
| 123.54628679, | 88.15393449, | 121.1334763 , | 108.44894565, |
| 93.25766544, | 108.91579775, | 114.68995756, | 91.78931381, |
| 137.95983365, | 187.67564685, | 146.10661119, | 113.86233347, |
| 129.63034739, | 118.43356291, | 100.35495653, | 120.76221839, |
| 118.28454313, | 117.68613801, | 143.04868576, | 123.00527638, |
| 96.37844223, | 113.34617325, | 113.06314566, | 121.45021337, |

|  |  |  |  |
| --- | --- | --- | --- |
| 104.95419623, | 115.17070977, | 121.73555596, | 120.99914638, |
| 96.5838649 , | 173.99638 , | 121.4400176 , | 117.30414 , |
| 154.11912998, | 114.80951599, | 107.36060366, | 122.96392927, |
| 123.98837017, | 119.09511621, | 117.33804646, | 143.06648787, |
| 144.20178 , | 119.00239459, | 147.6158906 , | 140.24769391, |
| 119.40825137, | 121.83192178, | 180.85291394, | 114.97636358, |
| 149.21210165, | 92.62256135, | 126.61562941, | 95.29406816, |
| 103.54469313, | 95.93372695, | 101.68744288, | 109.76485743, |
| 152.09727524, | 148.05724613, | 144.60781849, | 121.23614109, |
| 110.61807636, | 125.839362 , | 120.53693006, | 95.05296625, |
| 111.4796053 , | 140.22500374, | 105.26721178, | 112.24198177, |
| 146.7461144 , | 84.05268936, | 120.24755163, | 156.2693208 , |
| 114.6465138 , | 164.03482121, | 141.15214467, | 161.36671441, |
| 144.20282805]) |  |  |  |

print(model.intercept\_ ) *#checking the intercept of model equation*

print(model.coef\_) *#Checking the coefficients of model equation*

69.81906580324966

[ 7.15821702e-03 -2.41908638e-01 3.00933825e+00 -9.46756573e+00]

Y\_test = [1, 2, 3]

y\_pred = [1, 2, 3] metrics.r2\_score(Y\_test, y\_pred)

1.0

Conclusion:- In conclusion, this assignment demonstrates the importance of linear regression in predicting numerical outcomes. By understanding the theory behind linear regression and how to apply it to a dataset, we can make accurate predictions and gain insights from the data.