Name: Krishna Bhatu More

Class: BTECH

Year: Second

Branch: Computer

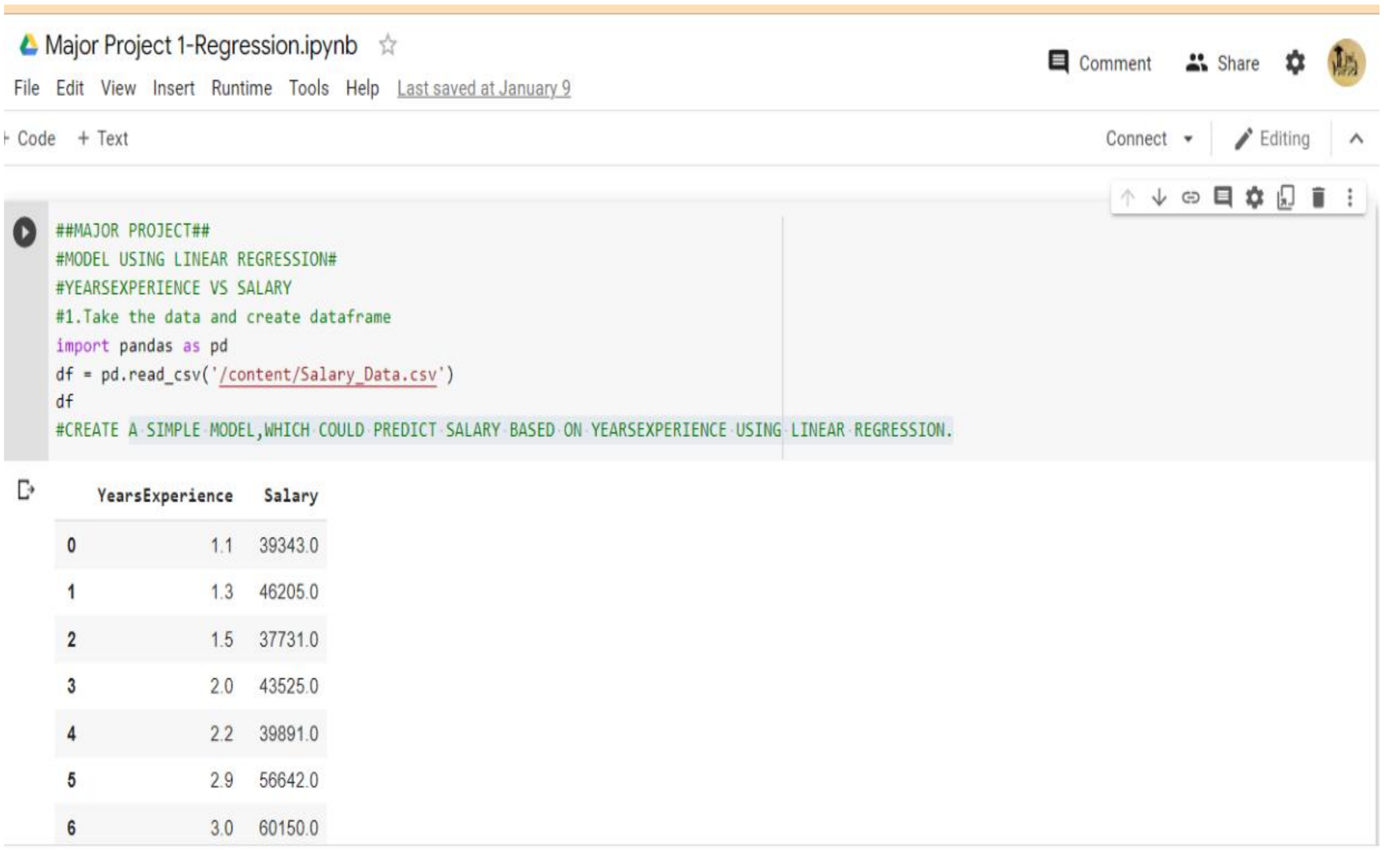
College: R.C.Patel Institute Of Technology,Shirpur

Program: Data Science(Nov-22)

Major Project-1: Chose dataset of your choice and apply regressor

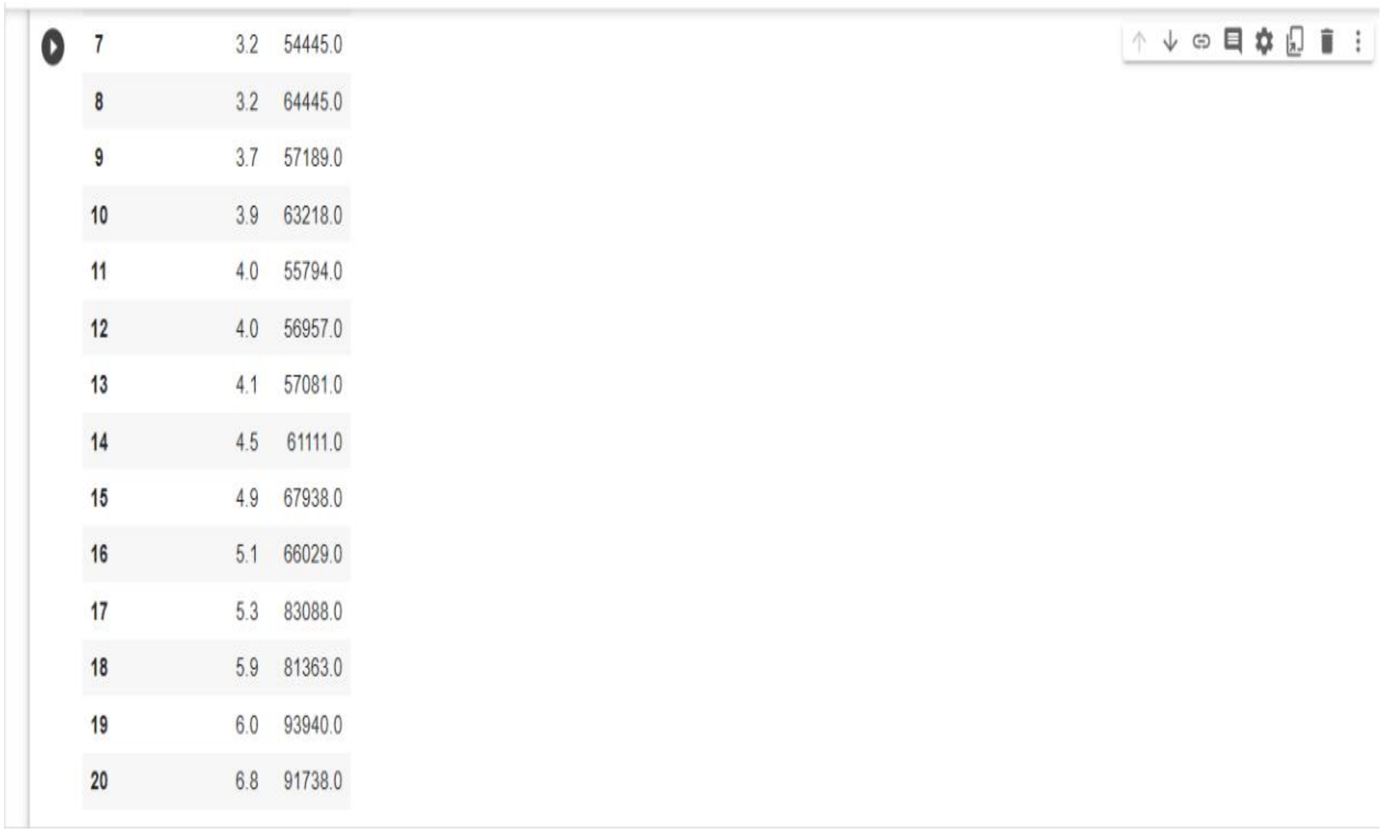
Major Project-2: Choose any dataset of your choice and perform EDA for atleast 15 different facts

Major Project-1: Years vs Salary Model using regressor


Major Project 1-Regression.ipynb
File Edit View Insert Runtime Tools Help Last saved at January 9
Comment
Share
+ Code + Text
Connect
Editing

```
##MAJOR PROJECT##
#MODEL USING LINEAR REGRESSION#
#YEARSEXPERIENCE VS SALARY
#1.Take the data and create dataframe
import pandas as pd
df = pd.read_csv('/content/Salary_Data.csv')
df
#CREATE A SIMPLE MODEL,WHICH COULD PREDICT SALARY BASED ON YEARSEXPERIENCE USING LINEAR REGRESSION.
```

```
    YearsExperience   Salary
0               1.1  39343.0
1               1.3  46205.0
2               1.5  37731.0
3               2.0  43525.0
4               2.2  39891.0
5               2.9  56642.0
6               3.0  60150.0
```

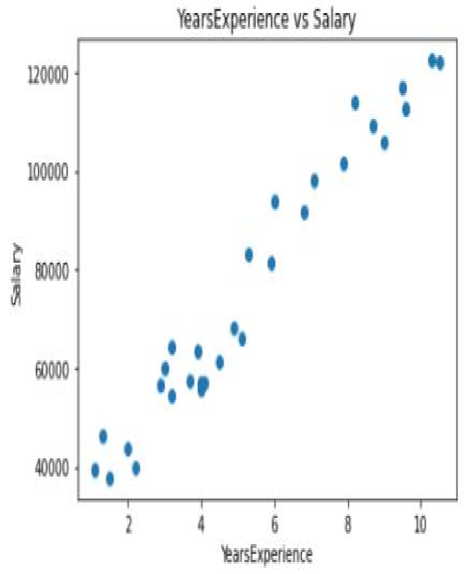


```
7               3.2  54445.0
8               3.2  64445.0
9               3.7  57189.0
10              3.9  63218.0
11              4.0  55794.0
12              4.0  56957.0
13              4.1  57081.0
14              4.5  61111.0
15              4.9  67938.0
16              5.1  66029.0
17              5.3  83088.0
18              5.9  81363.0
19              6.0  93940.0
20              6.8  91738.0
```

21	7.1	98273.0
22	7.9	101302.0
23	8.2	113812.0
24	8.7	109431.0
25	9.0	105582.0
26	9.5	116969.0
27	9.6	112635.0
28	10.3	122391.0
29	10.5	121872.0

```
[ ] #2.We are not performing step no 2 ,for this dataset
    #3.DATA VISUALISATION - CREATION of GRAPHS
    import matplotlib.pyplot as plt
    #plt.scatter(x-axis,y-axis)
    plt.scatter(df['YearsExperience'],df['Salary'])
    plt.title('YearsExperience vs Salary')
    plt.xlabel('YearsExperience')
[ ]
    plt.ylabel('Salary')
```

```
Text(0, 0.5, 'Salary')
```



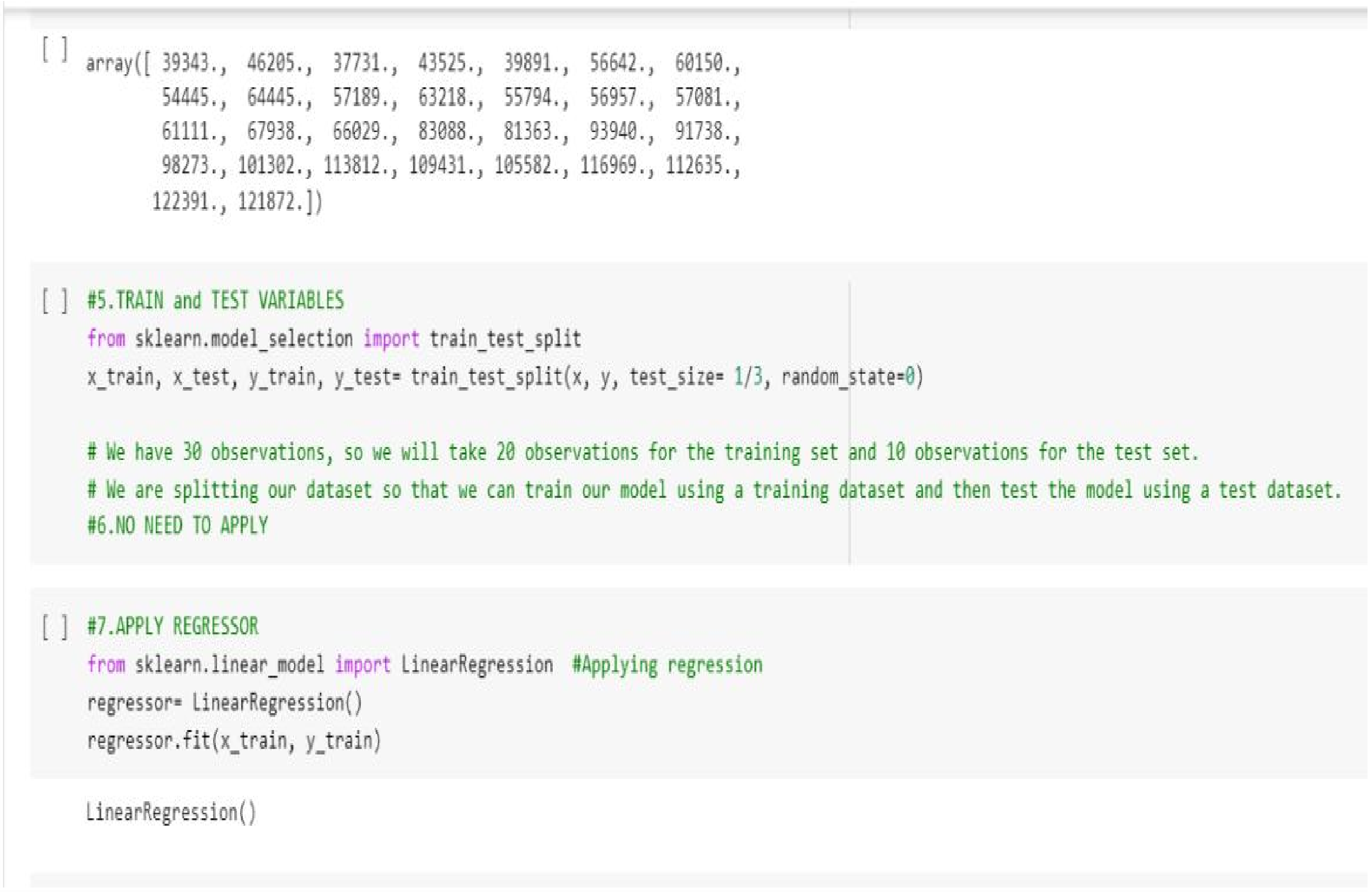
```
[ ] #4.
    #INPUT(x) -YearsExperience
    #OUTPUT - Salary
```

```
#4.DIVIDE THE DATA INTO INPUT and OUTPUT
#INPUT(x) is always 2 dimensional,OUTPUT(y) is always 1 dimensional array
#df.iloc[row slicing,column slicing]
#In df.iloc ,if the column's place has a ':',then array is 2 dimensional
x = df.iloc[0:30,0:1].values
x
#.values converts the dataframe into an array
#OUTPUT - Salary
```

```
array([[ 1.1],
       [ 1.3],
       [ 1.5],
       [ 2. ],
       [ 2.2],
       [ 2.9],
       [ 3. ],
       [ 3.2],
       [ 3.2],
       [ 3.7],
       [ 3.9],
```

```
[ ]        [ 4. ],
           [ 4. ],
           [ 4.1],
           [ 4.5],
           [ 4.9],
           [ 5.1],
           [ 5.3],
           [ 5.9],
           [ 6. ],
           [ 6.8],
           [ 7.1],
           [ 7.9],
           [ 8.2],
           [ 8.7],
           [ 9. ],
           [ 9.5],
           [ 9.6],
           [10.3],
           [10.5]])
```

```
[ ] #In df.iloc ,if the column's place does not  has a ':',then array is 1 dimensiona
    y = df.iloc[0:30,1].values #df.iloc[:,1].values
    #If I want to select all rows or all cols , I can write only :
    y
```

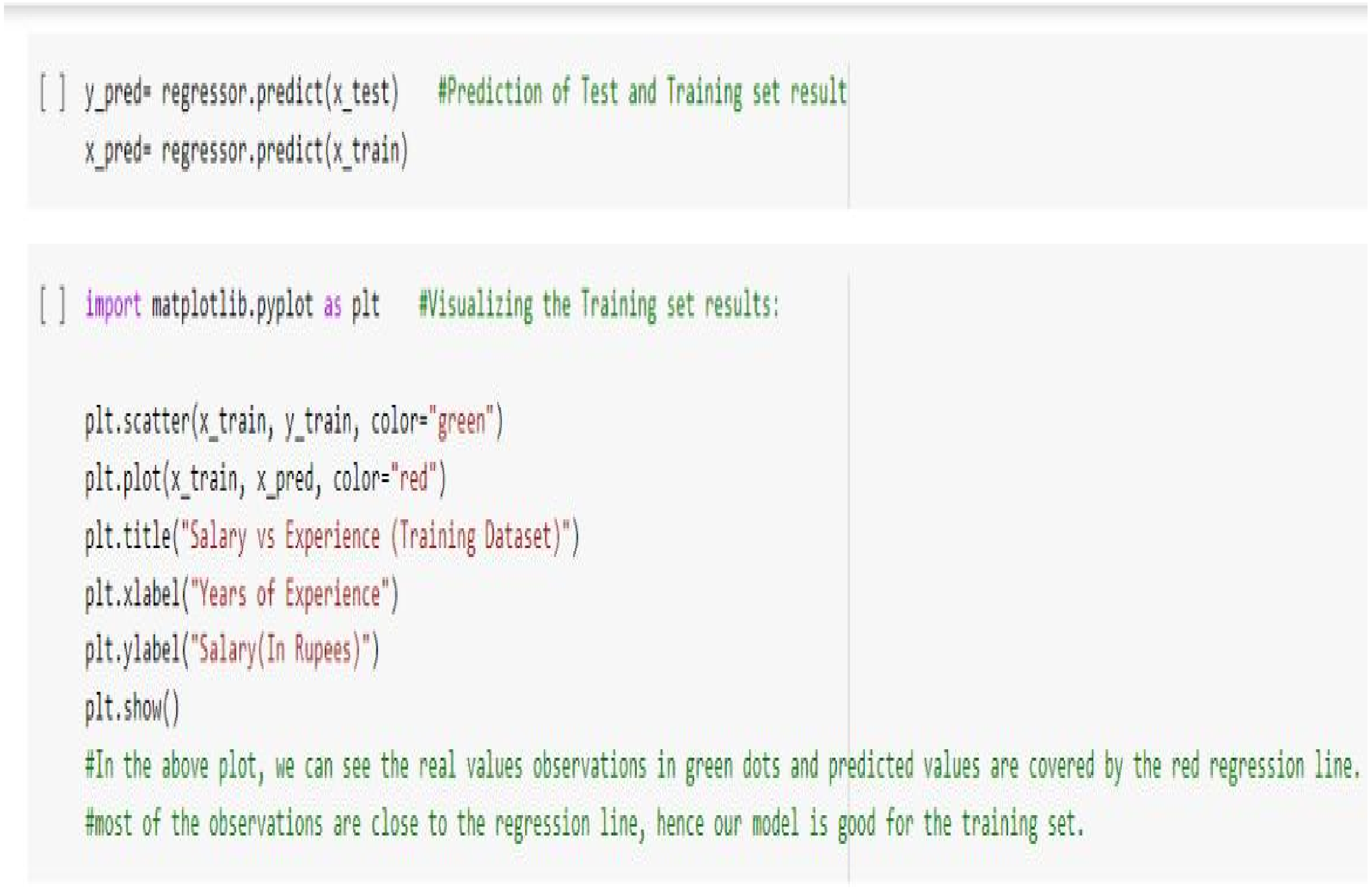


```
array([ 39343.,  46205.,  37731.,  43525.,  39891.,  56642.,  60150.,
        54445.,  64445.,  57189.,  63218.,  55794.,  56957.,  57081.,
        61111.,  67938.,  66029.,  83088.,  81363.,  93940.,  91738.,
        98273., 101302., 113812., 109431., 105582., 116969., 112635.,
       122391., 121872.])
```

```
#5.TRAIN and TEST VARIABLES
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 1/3, random_state=0)

# We have 30 observations, so we will take 20 observations for the training set and 10 observations for the test set.
# We are splitting our dataset so that we can train our model using a training dataset and then test the model using a test dataset.
#6.NO NEED TO APPLY
```

```
#7.APPLY REGRESSOR
from sklearn.linear_model import LinearRegression  #Applying regression
regressor= LinearRegression()
regressor.fit(x_train, y_train)
```

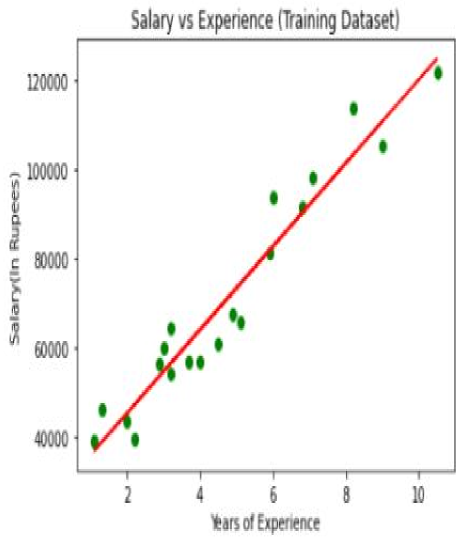
```
LinearRegression()
```



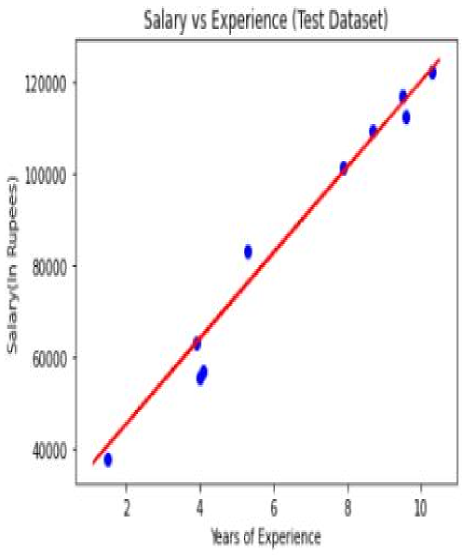
```
y_pred= regressor.predict(x_test)    #Prediction of Test and Training set result
x_pred= regressor.predict(x_train)
```

```
import matplotlib.pyplot as plt    #Visualizing the Training set results:


plt.scatter(x_train, y_train, color="green")
plt.plot(x_train, x_pred, color="red")
plt.title("Salary vs Experience (Training Dataset)")
plt.xlabel("Years of Experience")
plt.ylabel("Salary(In Rupees)")
plt.show()
#In the above plot, we can see the real values observations in green dots and predicted values are covered by the red regression line.
#most of the observations are close to the regression line, hence our model is good for the training set.
```

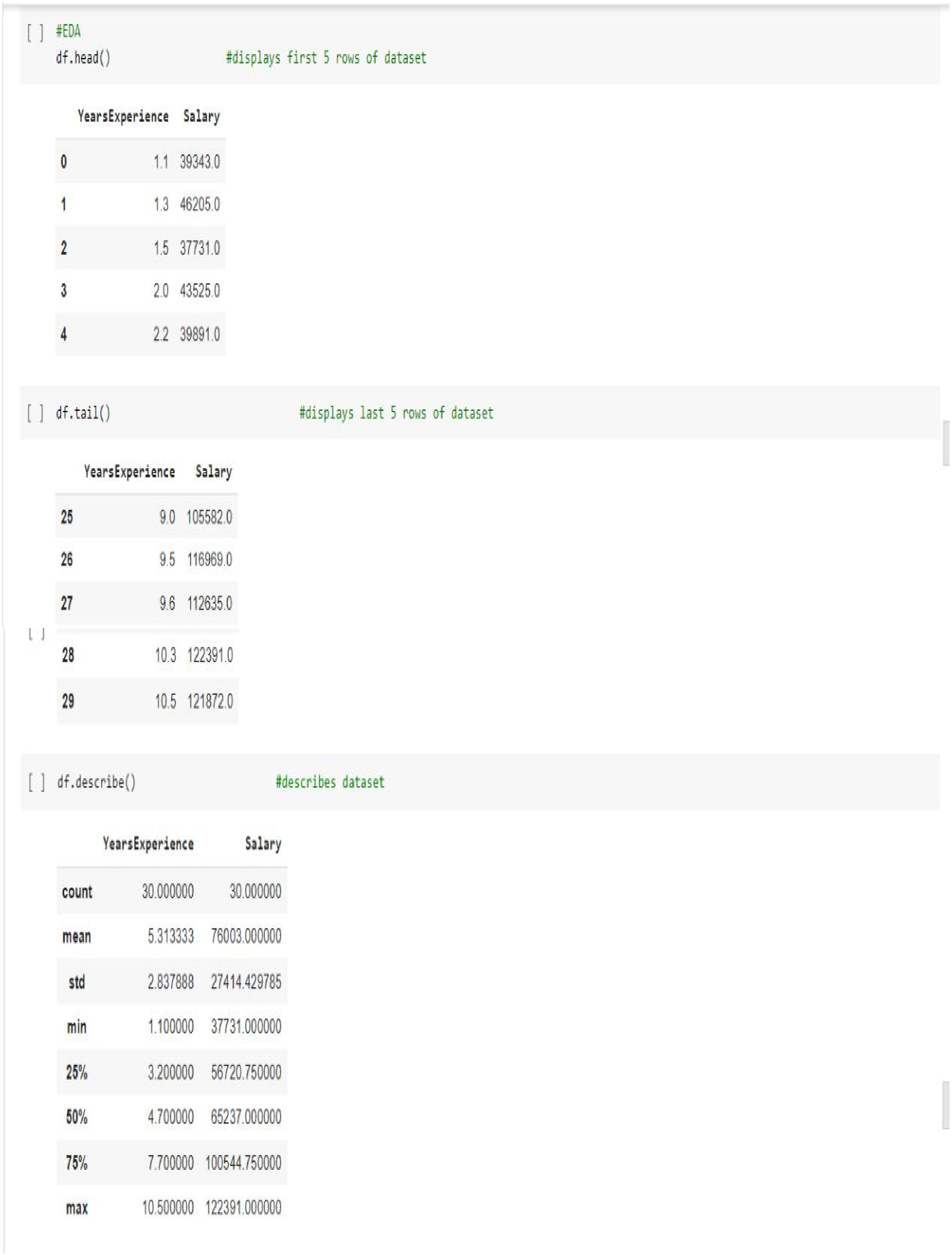
[ ]



```
[ ] #Visualizing the Test set results
    plt.scatter(x_test, y_test, color="blue")
    plt.plot(x_train, x_pred, color="red")
    plt.title("Salary vs Experience (Test Dataset)")
    plt.xlabel("Years of Experience")
    plt.ylabel("Salary(In Rupees)")
    plt.show()
[ ]
    #In the below plot, there are observations given by the blue color, and prediction is given by the red regression line.
    #As we can see,most of the observations are close to the regression line,
    #Hence we can say our Simple Linear Regression is a good model and able to make good predictions.
```



```
[ ] #EDA
```



```
[ ] #EDA
    df.head()                    #displays first 5 rows of dataset
```

```
   YearsExperience   Salary
0              1.1  39343.0
1              1.3  46205.0
2              1.5  37731.0
3              2.0  43525.0
4              2.2  39891.0
```

```
[ ] df.tail()                             #displays last 5 rows of dataset
```

```
    YearsExperience    Salary
25              9.0  105582.0
26              9.5  116969.0
27              9.6  112635.0
28             10.3  122391.0
29             10.5  121872.0
```

```
[ ] df.describe()                     #describes dataset
```

```
       YearsExperience         Salary
count        30.000000      30.000000
mean          5.313333   76003.000000
std           2.837888   27414.429785
min           1.100000   37731.000000
25%           3.200000   56720.750000
50%           4.700000   65237.000000
75%           7.700000  100544.750000
max          10.500000  122391.000000
```

```
[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30 entries, 0 to 29
Data columns (total 2 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   YearsExperience  30 non-null     float64
 1   Salary           30 non-null     float64
dtypes: float64(2)
memory usage: 608.0 bytes
```

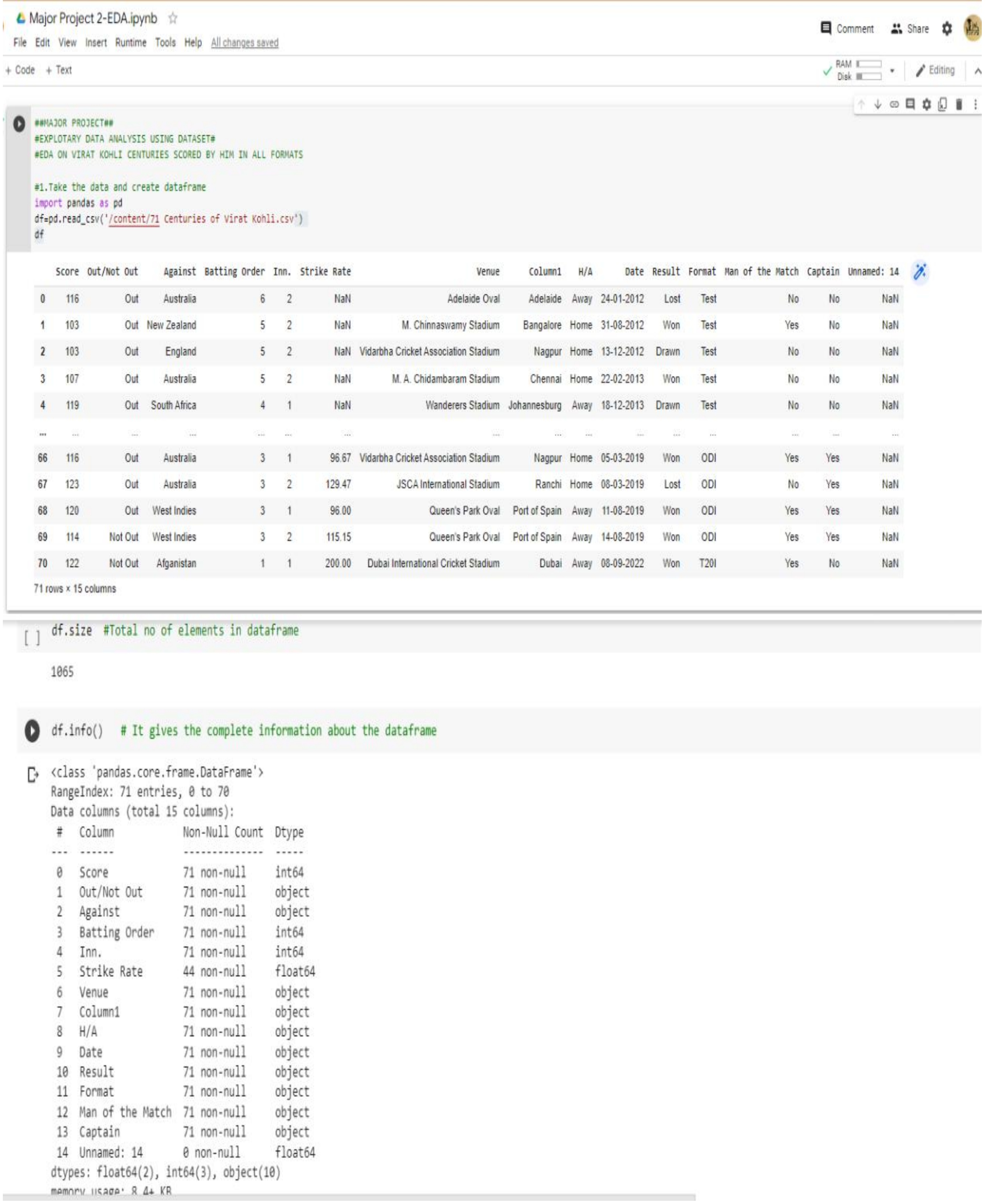
```
[ ] df.shape                #30 rows and 2 column
```

```
(30, 2)
```

```
[ ] df.size                 #check the size of dataset
```

```
60
```

Major Project-2: EDA Using 71 Century dataset of Virat Kohli



Major Project 2-EDA.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing

```
##MAJOR PROJECT##
#EXPLOTARY DATA ANALYSIS USING DATASET#
#EDA ON VIRAT KOHLI CENTURIES SCORED BY HIM IN ALL FORMATS

#1.Take the data and create dataframe
import pandas as pd
df=pd.read_csv('/content/71 Centuries of Virat Kohli.csv')
df
```

	Score	Out/Not Out	Against	Batting Order	Inn.	Strike Rate	Venue	Column1	H/A	Date	Result	Format	Man of the Match	Captain	Unnamed: 14
0	116	Out	Australia	6	2	NaN	Adelaide Oval	Adelaide	Away	24-01-2012	Lost	Test	No	No	NaN
1	103	Out	New Zealand	5	2	NaN	M. Chinnaswamy Stadium	Bangalore	Home	31-08-2012	Won	Test	Yes	No	NaN
2	103	Out	England	5	2	NaN	Vidarbha Cricket Association Stadium	Nagpur	Home	13-12-2012	Drawn	Test	No	No	NaN
3	107	Out	Australia	5	2	NaN	M. A. Chidambaram Stadium	Chennai	Home	22-02-2013	Won	Test	No	No	NaN
4	119	Out	South Africa	4	1	NaN	Wanderers Stadium	Johannesburg	Away	18-12-2013	Drawn	Test	No	No	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
66	116	Out	Australia	3	1	96.67	Vidarbha Cricket Association Stadium	Nagpur	Home	05-03-2019	Won	ODI	Yes	Yes	NaN
67	123	Out	Australia	3	2	129.47	JSCA International Stadium	Ranchi	Home	08-03-2019	Lost	ODI	No	Yes	NaN
68	120	Out	West Indies	3	1	96.00	Queen's Park Oval	Port of Spain	Away	11-08-2019	Won	ODI	Yes	Yes	NaN
69	114	Not Out	West Indies	3	2	115.15	Queen's Park Oval	Port of Spain	Away	14-08-2019	Won	ODI	Yes	Yes	NaN
70	122	Not Out	Afganistan	1	1	200.00	Dubai International Cricket Stadium	Dubai	Away	08-09-2022	Won	T20I	Yes	No	NaN

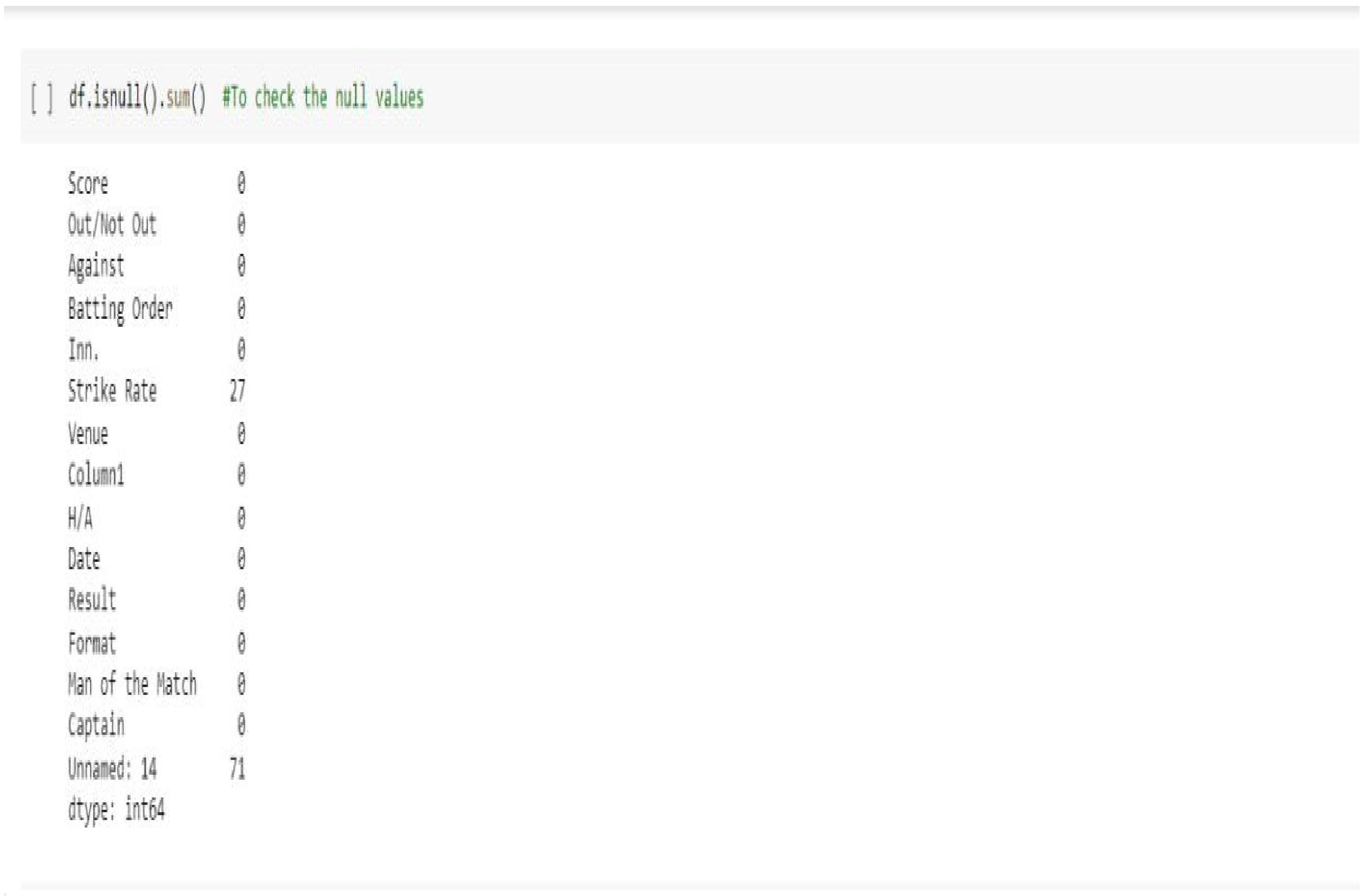
71 rows × 15 columns

```
[ ] df.size  #Total no of elements in dataframe
```

```
1065
```

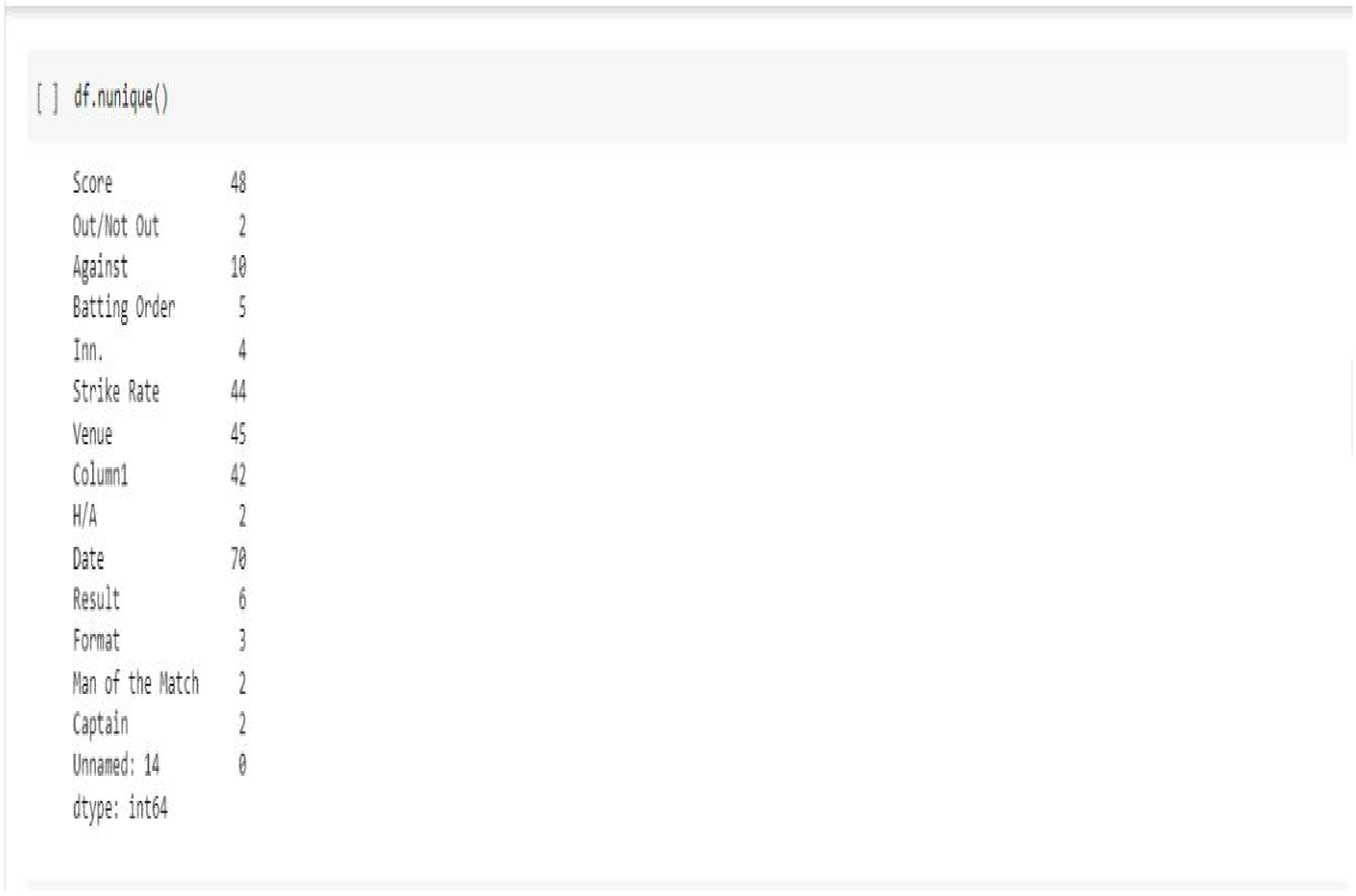
```
df.info()  # It gives the complete information about the dataframe
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 71 entries, 0 to 70
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Score             71 non-null     int64
 1   Out/Not Out       71 non-null     object
 2   Against           71 non-null     object
 3   Batting Order     71 non-null     int64
 4   Inn.              71 non-null     int64
 5   Strike Rate       44 non-null     float64
 6   Venue             71 non-null     object
 7   Column1           71 non-null     object
 8   H/A               71 non-null     object
 9   Date              71 non-null     object
 10  Result            71 non-null     object
 11  Format            71 non-null     object
 12  Man of the Match  71 non-null     object
 13  Captain           71 non-null     object
 14  Unnamed: 14       0 non-null      float64
dtypes: float64(2), int64(3), object(10)
memory usage: 8.4+ KB
```



```
[ ] df.isnull().sum()  #To check the null values
```

```
Score              0
Out/Not Out        0
Against            0
Batting Order      0
Inn.               0
Strike Rate       27
Venue              0
Column1            0
H/A                0
Date               0
Result             0
Format             0
Man of the Match   0
Captain            0
Unnamed: 14       71
dtype: int64
```

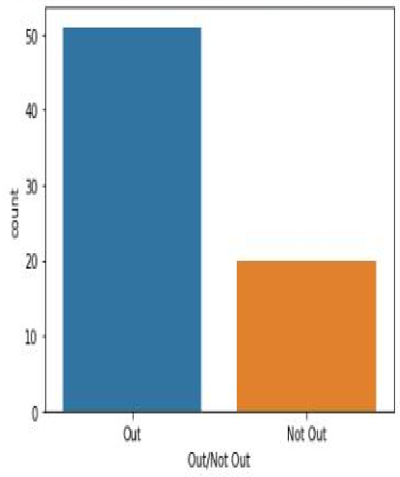


```
[ ] df.nunique()
```

```
Score             48
Out/Not Out        2
Against           10
Batting Order      5
Inn.               4
Strike Rate       44
Venue             45
Column1           42
H/A                2
Date              70
Result             6
Format             3
Man of the Match   2
Captain            2
Unnamed: 14        0
dtype: int64
```

```
[ ] import seaborn as sns                #Visualisation-using seaborn
    sns.countplot(x='Out/Not Out',data=df)   #This Visualisation tells about how many times did virat was remained Notout after hitting century
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8e01ceabb0>
```



```
[ ] df['Out/Not Out'].value_counts()              #No.of Out/Notout after hitting century
```

```
Out        51
Not Out    20
Name: Out/Not Out, dtype: int64
```

```
[ ] df['Result'].value_counts()                  #Result when Virat scored a Century
```

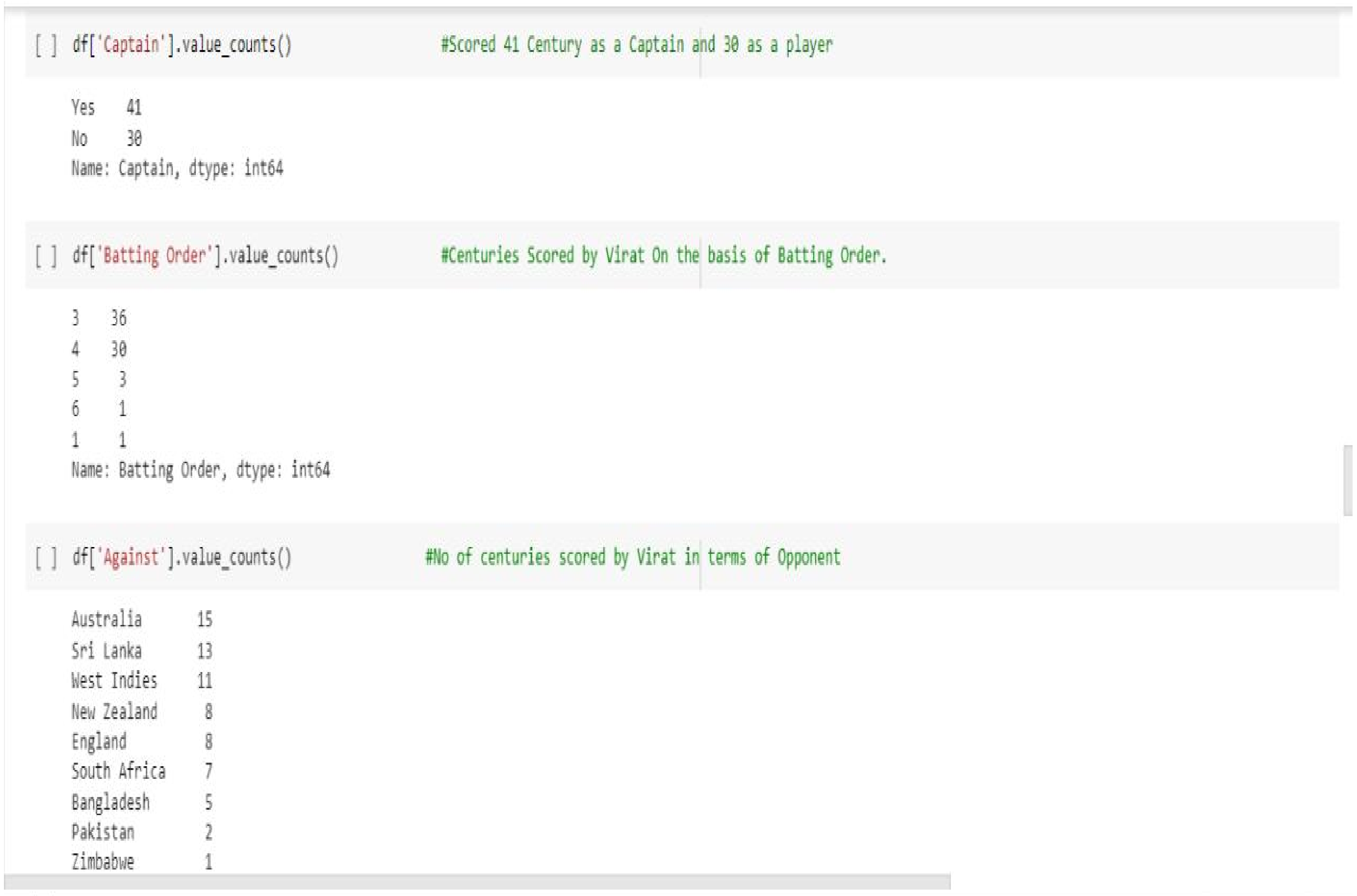
```
Won          48
Lost         13
Drawn         7
Lost (D/L)    1
Won (D/L)     1
Tied          1
Name: Result, dtype: int64
```

```
[ ] df['H/A'].value_counts()                   #Scored 39 centuries in India(Home) and 32 in other country(Away)
```

```
Away    39
Home    32
Name: H/A, dtype: int64
```

```
[ ] df['Format'].value_counts()                #No of Centuries scored in terms of Format
```

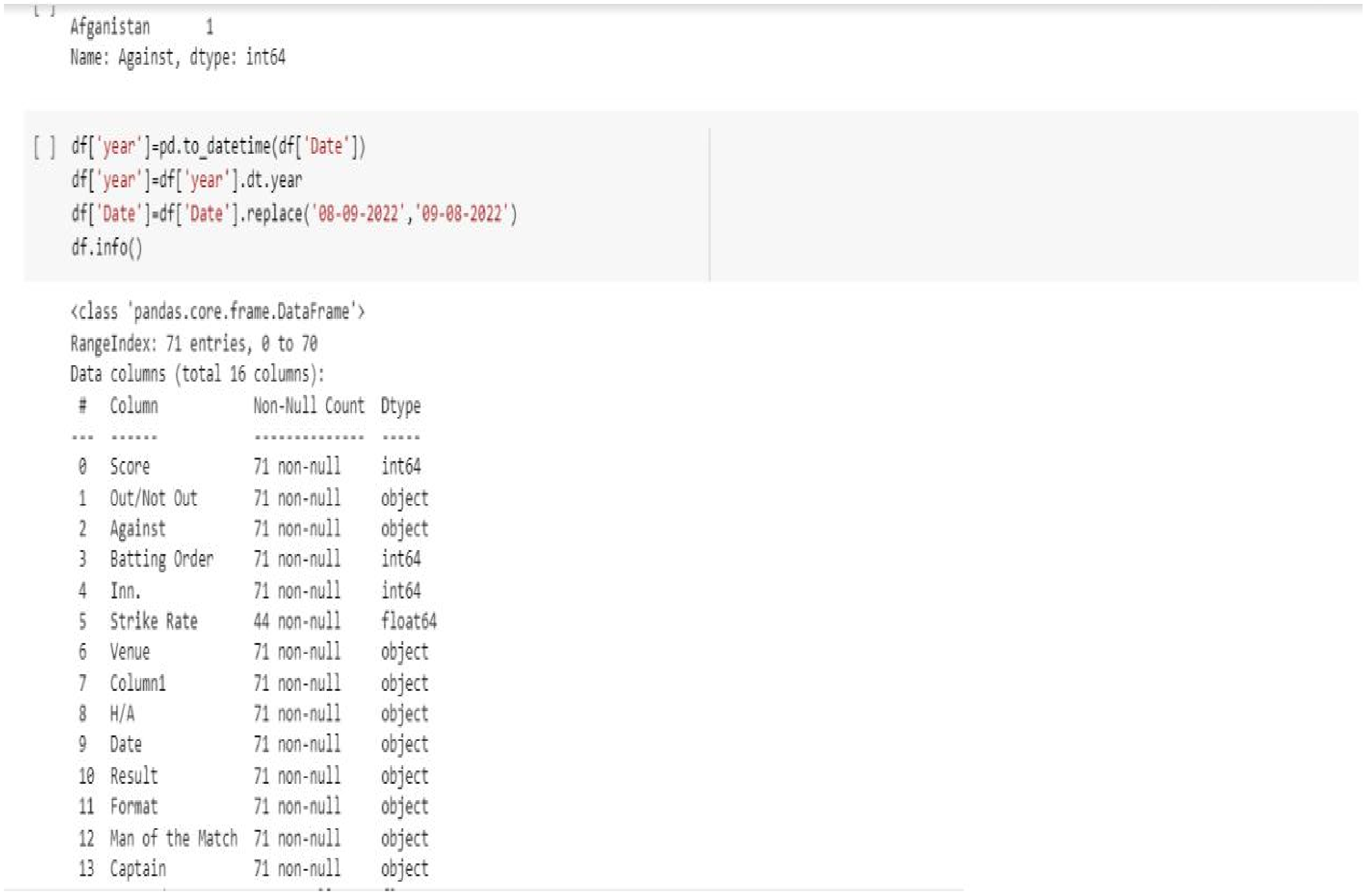
```
ODI     43
Test    27
T20I     1
Name: Format, dtype: int64
```



```
[ ] df['Captain'].value_counts()                #Scored 41 Century as a Captain and 30 as a player

    Yes    41
    No     30
    Name: Captain, dtype: int64
```

```
[ ] df['Batting Order'].value_counts()          #Centuries Scored by Virat On the basis of Batting Order.

    3    36
    4    30
    5     3
    6     1
    1     1
    Name: Batting Order, dtype: int64
```

```
[ ] df['Against'].value_counts()               #No of centuries scored by Virat in terms of Opponent

    Australia       15
    Sri Lanka       13
    West Indies     11
    New Zealand      8
    England          8
    South Africa     7
    Bangladesh       5
    Pakistan         2
    Zimbabwe         1
```



```
    Afganistan       1
    Name: Against, dtype: int64
```

```
[ ] df['year']=pd.to_datetime(df['Date'])
    df['year']=df['year'].dt.year
    df['Date']=df['Date'].replace('08-09-2022','09-08-2022')
    df.info()

    <class 'pandas.core.frame.DataFrame'>
    RangeIndex: 71 entries, 0 to 70
    Data columns (total 16 columns):
     #   Column            Non-Null Count  Dtype
    ---  ------            --------------  -----
     0   Score             71 non-null     int64
     1   Out/Not Out       71 non-null     object
     2   Against           71 non-null     object
     3   Batting Order     71 non-null     int64
     4   Inn.              71 non-null     int64
     5   Strike Rate       44 non-null     float64
     6   Venue             71 non-null     object
     7   Column1           71 non-null     object
     8   H/A               71 non-null     object
     9   Date              71 non-null     object
     10  Result            71 non-null     object
     11  Format            71 non-null     object
     12  Man of the Match  71 non-null     object
     13  Captain           71 non-null     object
```

```
[ ]  14  Unnamed: 14      0 non-null      float64
     15  year             71 non-null     int64
    dtypes: float64(2), int64(4), object(10)
    memory usage: 9.0+ KB
```

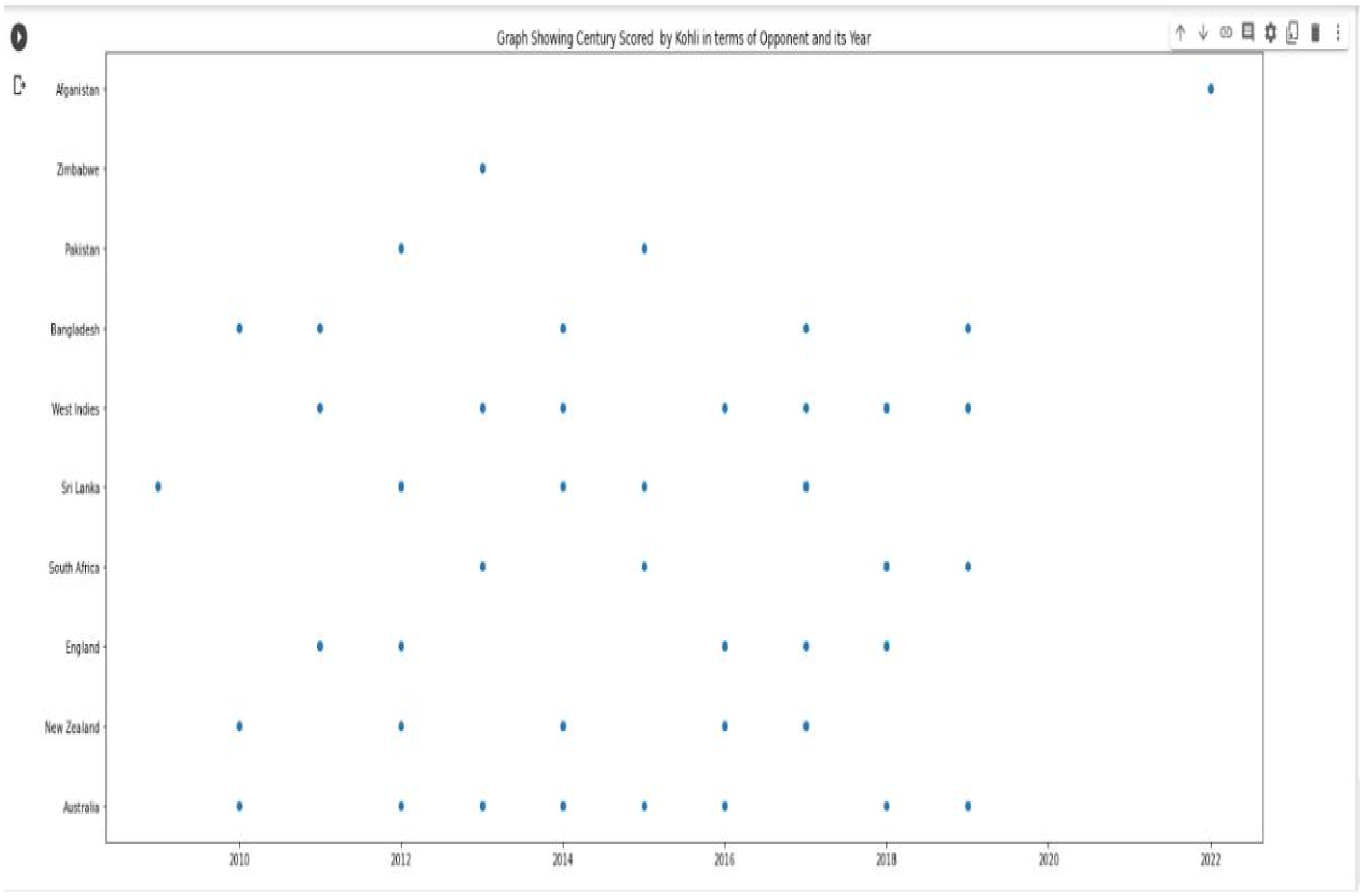
```
[ ] df.describe()
```

	Score	Batting Order	Inn.	Strike Rate	Unnamed: 14
count	71.000000	71.000000	71.000000	44.000000	0.0
mean	132.140845	3.521127	1.732394	114.019545	NaN
std	35.911119	0.714326	0.675230	25.257567	NaN
min	100.000000	1.000000	1.000000	84.900000	NaN
25%	107.000000	3.000000	1.000000	96.632500	NaN
50%	119.000000	3.000000	2.000000	108.935000	NaN
75%	139.500000	4.000000	2.000000	120.787500	NaN
max	254.000000	6.000000	4.000000	200.000000	NaN

```
[ ] df.groupby(['Against','year'],as_index=False)['Score'].max().sort_values(by='Score',ascending=False).reset_index(drop=True).head()   #Kohli Scores Highest (254)
```

	Against	year	Score
0	South Africa	2019	254
1	Sri Lanka	2017	243
2	England	2016	235
3	New Zealand	2016	211
4	Bangladesh	2017	204

```
[ ] df_new = df[['year','Against']]  #SEPARATE DATA FOR YEAR AND AGAINST COLUMN
    df_new
    import matplotlib.pyplot as plt
    plt.figure(figsize = (30,15))
    plt.scatter(df_new['year'],df_new['Against'])


    plt.title("Graph Showing Century Scored  by Kohli in terms of Opponent and its Year")

    plt.show()
```



Graph Showing Century Scored by Kohli in terms of Opponent and its Year
Afganistan
Zimbabwe
Pakistan
Bangladesh
West Indies
Sri Lanka
South Africa
England
New Zealand
Australia
2010
2012
2014
2016
2018
2020
2022