Neural Network & Deep Learning

(ICP Assignment # 4)

CS 5720

Name: Satya Ishyanth Kadali

Id: 700735513

CRN: 23259

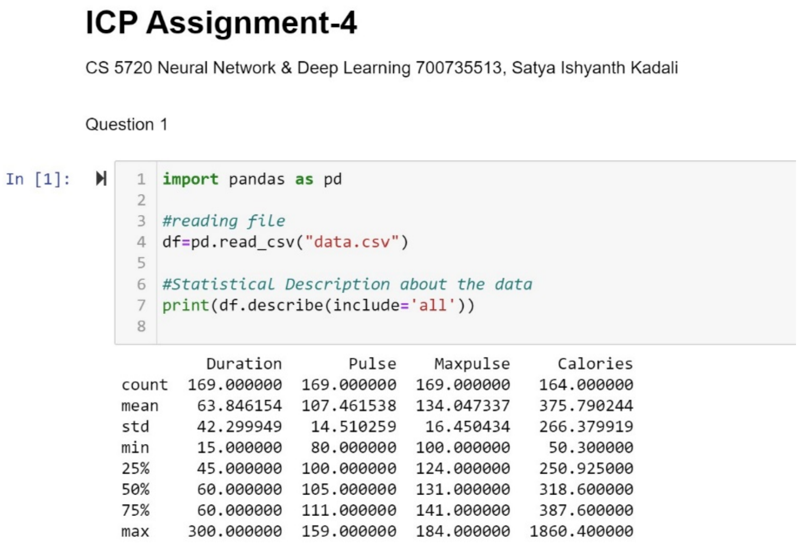
Video link:

https://drive.google.com/file/d/1wBqvJDcnDjkPoFYjO9R3PbcFCtLp76oJ/view?usp=share_link

Github link:

https://github.com/Ishyanth/Deep-Learning-Assignments

Question 1: Data Manipulation



ICP Assignment-4

CS 5720 Neural Network & Deep Learning 700735513, Satya Ishyanth Kadali

Question 1

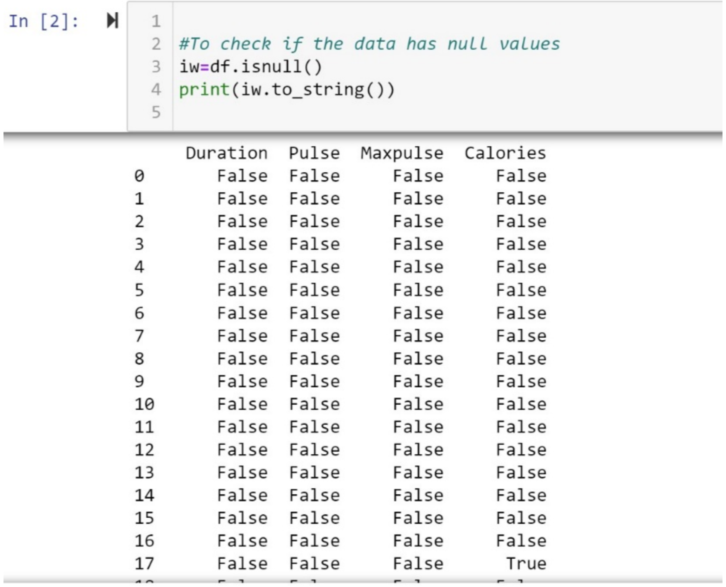
In [1]:

```
import pandas as pd

#reading file
df=pd.read_csv("data.csv")

#Statistical Description about the data
print(df.describe(include='all'))

```

```
         Duration       Pulse    Maxpulse    Calories
count  169.000000  169.000000  169.000000  164.000000
mean    63.846154  107.461538  134.047337  375.790244
std     42.299949   14.510259   16.450434  266.379919
min     15.000000   80.000000  100.000000   50.300000
25%     45.000000  100.000000  124.000000  250.925000
50%     60.000000  105.000000  131.000000  318.600000
75%     60.000000  111.000000  141.000000  387.600000
max    300.000000  159.000000  184.000000 1860.400000
```

Importing pandas to read data into dataframe.

Using describe method to view statistical description about the data.

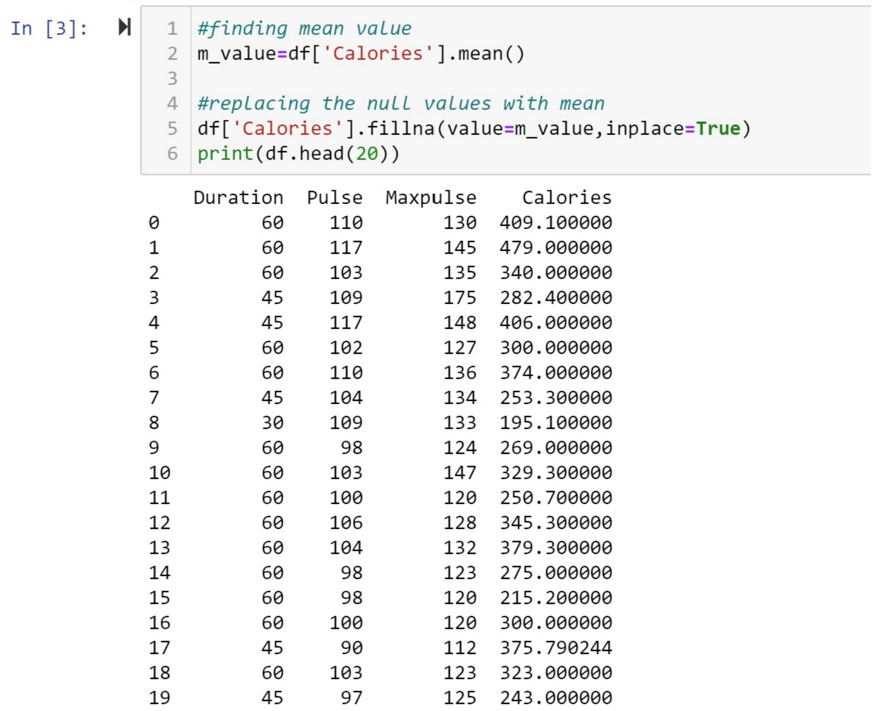


In [2]:

```

#To check if the data has null values
iw=df.isnull()
print(iw.to_string())

```

```
     Duration  Pulse  Maxpulse  Calories
0       False  False     False     False
1       False  False     False     False
2       False  False     False     False
3       False  False     False     False
4       False  False     False     False
5       False  False     False     False
6       False  False     False     False
7       False  False     False     False
8       False  False     False     False
9       False  False     False     False
10      False  False     False     False
11      False  False     False     False
12      False  False     False     False
13      False  False     False     False
14      False  False     False     False
15      False  False     False     False
16      False  False     False     False
17      False  False     False      True
```

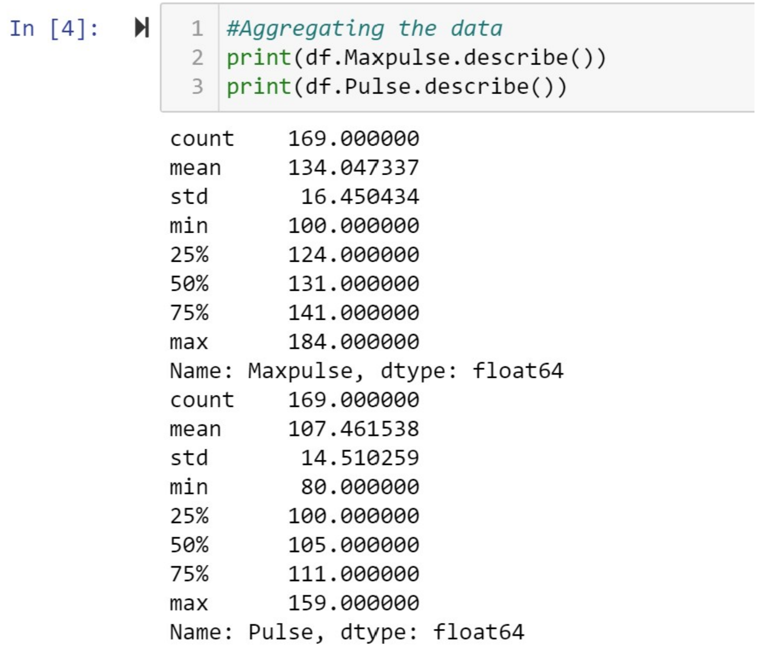
Checking for any null values in the data.



```
In [3]:
#finding mean value
m_value=df['Calories'].mean()

#replacing the null values with mean
df['Calories'].fillna(value=m_value,inplace=True)
print(df.head(20))
```

```
    Duration  Pulse  Maxpulse    Calories
0         60    110       130  409.100000
1         60    117       145  479.000000
2         60    103       135  340.000000
3         45    109       175  282.400000
4         45    117       148  406.000000
5         60    102       127  300.000000
6         60    110       136  374.000000
7         45    104       134  253.300000
8         30    109       133  195.100000
9         60     98       124  269.000000
10        60    103       147  329.300000
11        60    100       120  250.700000
12        60    106       128  345.300000
13        60    104       132  379.300000
14        60     98       123  275.000000
15        60     98       120  215.200000
16        60    100       120  300.000000
17        45     90       112  375.790244
18        60    103       123  323.000000
19        45     97       125  243.000000
```

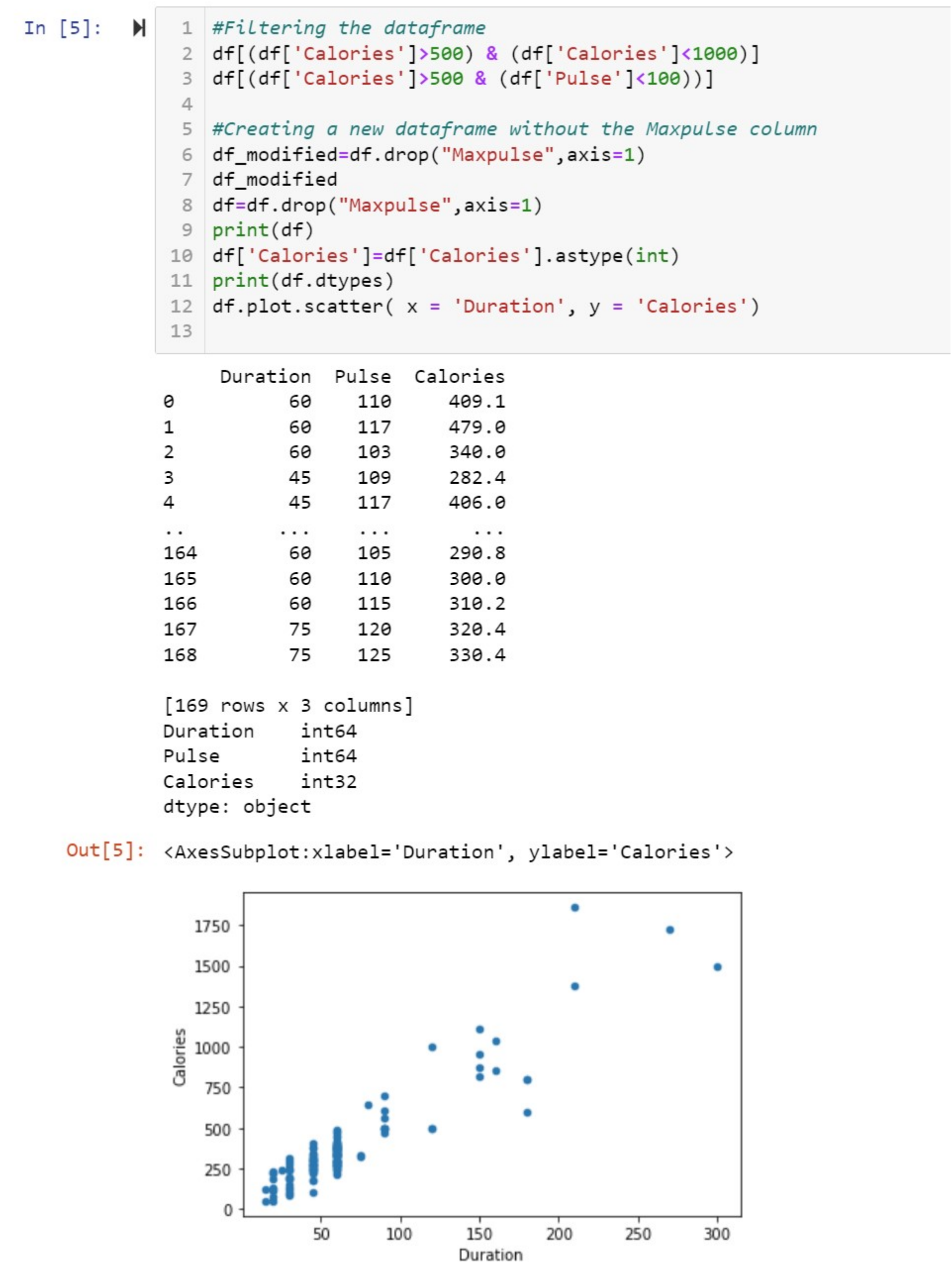
Replacing all null values by mean value of that column.



```
In [4]:
#Aggregating the data
print(df.Maxpulse.describe())
print(df.Pulse.describe())
```

```
count    169.000000
mean     134.047337
std       16.450434
min      100.000000
25%      124.000000
50%      131.000000
75%      141.000000
max      184.000000
Name: Maxpulse, dtype: float64
count    169.000000
mean     107.461538
std       14.510259
min       80.000000
25%      100.000000
50%      105.000000
75%      111.000000
max      159.000000
Name: Pulse, dtype: float64
```

Aggregating the data by using describe method to show count, mean, minimum and maximum values of the Maxpulse and Pulse columns.



In [5]:

```
#Filtering the dataframe
df[(df['Calories']>500) & (df['Calories']<1000)]
df[(df['Calories']>500 & (df['Pulse']<100))]

#Creating a new dataframe without the Maxpulse column
df_modified=df.drop("Maxpulse",axis=1)
df_modified
df=df.drop("Maxpulse",axis=1)
print(df)
df['Calories']=df['Calories'].astype(int)
print(df.dtypes)
df.plot.scatter( x = 'Duration', y = 'Calories')
```

```
     Duration  Pulse  Calories
0          60    110     409.1
1          60    117     479.0
2          60    103     340.0
3          45    109     282.4
4          45    117     406.0
..        ...    ...       ...
164        60    105     290.8
165        60    110     300.0
166        60    115     310.2
167        75    120     320.4
168        75    125     330.4

[169 rows x 3 columns]
Duration    int64
Pulse       int64
Calories    int32
dtype: object
```

Out[5]:

```
<AxesSubplot:xlabel='Duration', ylabel='Calories'>
```

Filtering the dataframe to select the rows with calories values between 500 and 1000.

And also filtering the dataframe to select the rows with calories values > 500 and pulse < 100.

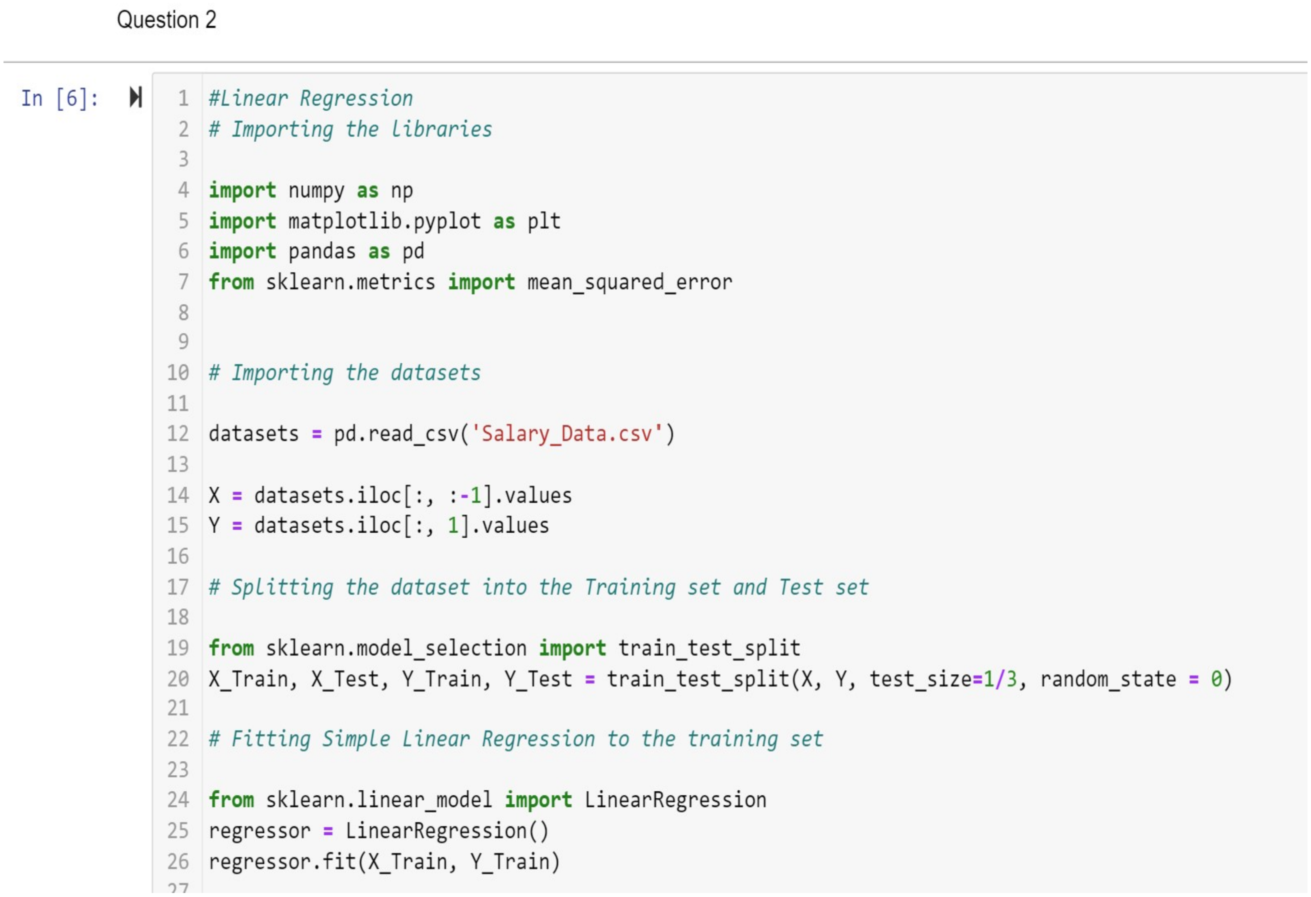
Creating a new "df_modified" dataframe that contains all the columns from df except for "Maxpulse".

Dropping the "Maxpulse" column from the main df dataframe

Converting the datatype of Calories column to int datatype.

Using scatter plot for the two columns (Duration and Calories) to plot the graph.

Question 2: Linear Regression



Question 2

In [6]:

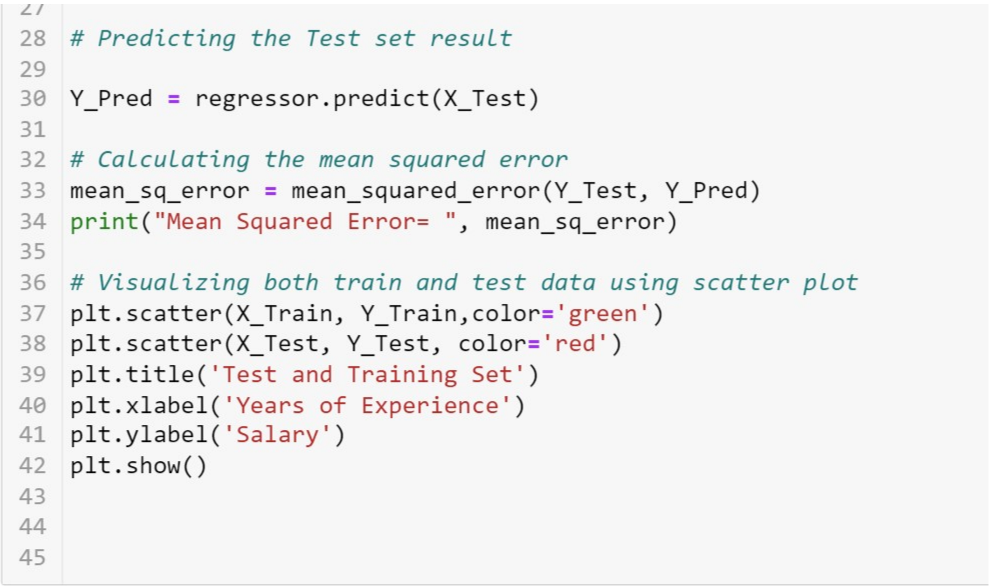
```
#Linear Regression
# Importing the libraries

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.metrics import mean_squared_error


# Importing the datasets

datasets = pd.read_csv('Salary_Data.csv')

X = datasets.iloc[:, :-1].values
Y = datasets.iloc[:, 1].values

# Splitting the dataset into the Training set 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=1/3, random_state = 0)

# Fitting Simple Linear Regression to the training set

from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_Train, Y_Train)
```

Importing libraries and reading the dataset.

Splitting the dataset into 1/3 for test set and remaining for training set.

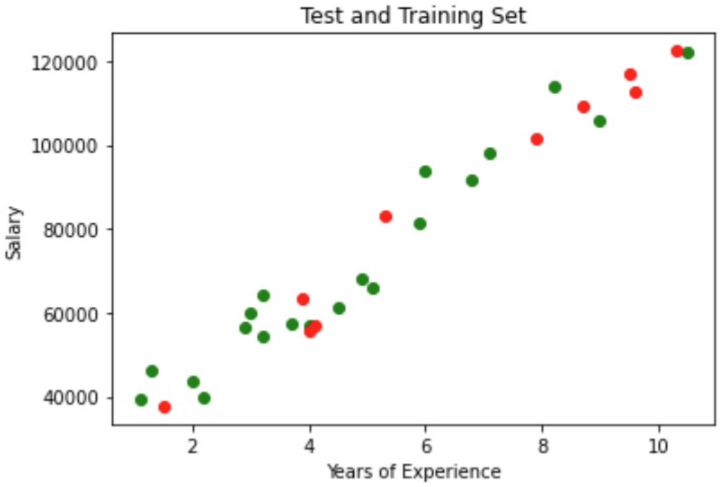
Now using sklearn importing linear Regression function.

Fitting the Linear regression to the training set.



```
# Predicting the Test set result

Y_Pred = regressor.predict(X_Test)

# Calculating the mean squared error
mean_sq_error = mean_squared_error(Y_Test, Y_Pred)
print("Mean Squared Error= ", mean_sq_error)

# Visualizing both train and test data using scatter plot
plt.scatter(X_Train, Y_Train,color='green')
plt.scatter(X_Test, Y_Test, color='red')
plt.title('Test and Training Set')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.show()
```

```
Mean Squared Error=  21026037.329511296
```



Predicting the test result by using regressor.predict function.

By sklearn.metrics library, we calculating mean_squared_error.

Now using scatter plot to visualize both train and test data.