ASSIGNMENT – 4

NEURAL NETWORKS & DEEP LEARNING

NAME: DEEPIKA IRRINKI
#700741218

GITHUB: https://github.com/Deepikairrinki/NN-DL_Assignment4

RECORDER:https://drive.google.com/file/d/1V1paQVQsJd0tVyJicNg-capwb8sn8bUk/view?usp=drive_link

QUESTION 1:

a. Read the provided CSV file 'data.csv'.
c. Show the basic statistical description about the data.
d. Check if the data has null values.
Replace the null values with the mean

Input:

NNDL_Assignment4_Q2Linear Regression.py | NNDL_Assignment4_Q1.py*

```
#A

#Import pandas
#Read the provided CSV file 'data.csv'.
import pandas as pd
df = pd.read_csv("data.csv")

#C

#Show the basic statistical description about the data.
#df.head()
df.describe()

#D

#Check if NULL values are there in the dataset
df.isnull().sum()
#D(i)

#Fill the NULL values with Mean
df = df.fillna(df.mean())
df
#verifying if there are any NULL values
df.isnull().sum()
```

e. Select at least two columns and aggregate the data using: min, max, count, mean.

Input:

```
#E

#Aggregation functions on 3 columns
result = df[['Duration','Pulse','Calories']].agg(['min', 'max','count','mean'])
print("Min, Max, Count and Mean values ")
result
```

Output:

```
neural')
Min, Max, Count and Mean values
```

Index	Duration	Pulse	Calories
min	15	80	50.3
max	300	159	1860.4
count	169	169	169
mean	63.8462	107.462	375.79

f. Filter the data frame to select the rows with calories values between 500 and 1000.

g. Filter the data frame to select the rows with calories values > 500 and pulse < 100.

```
#F

#Filtering with Calories between 500 and 1000
filter1 = df[(df.Calories >= 500) & (df.Calories <= 1000)]
filter1

#G

#Filtering the dataset with calories greater than 500 and pulse less than 100
filter2 = df[(df.Calories > 500) & (df.Pulse < 100)]
filter2
```

Output:

Filter 1:

Index	Duration	Pulse	Maxpulse	Calories
51	80	123	146	643.1
62	160	109	135	853
65	180	90	130	800.4
66	150	105	135	873.4
67	150	107	130	816
72	90	100	127	700

Filter 2:

filter2 - DataFrame

Index	Duration	Pulse	Maxpulse	Calories
65	180	90	130	800.4
70	150	97	129	1115
73	150	97	127	953.2
75	90	98	125	563.2
99	90	93	124	604.1
103	90	90	100	500.4

h. Create a new "df_modified" dataframe that contains all the columns from df except for "Maxpulse".
i. Delete the "Maxpulse" column from the main df dataframe
j. Convert the datatype of Calories column to int datatype

Input:

```
#H

#Showing all columns except Maxpulse in a new dataframe
df_modified = df.loc[:, df.columns!='Maxpulse']
df_modified

#I

#Deleting Maxpulse from main dataframe
del df["Maxpulse"]
df
#Existing datatypes of columns in dataframe
df.dtypes


#J

#Replacing the calories datatype
df.Calories = df.Calories.astype(int)
df.dtypes
```

Output:

df: Data Frame

df - DataFrame

Index	Duration	Pulse	Calories
0	60	110	409
1	60	117	479
2	60	103	340
3	45	109	282
4	45	117	406
5	60	102	300

df modifier:

df_modified - DataFrame

Index	Duration	Pulse	Calories
0	60	110	409.1
1	60	117	479
2	60	103	340
3	45	109	282.4
4	45	117	406

k. Using pandas create a scatter plot for the two columns (Duration and Calories).

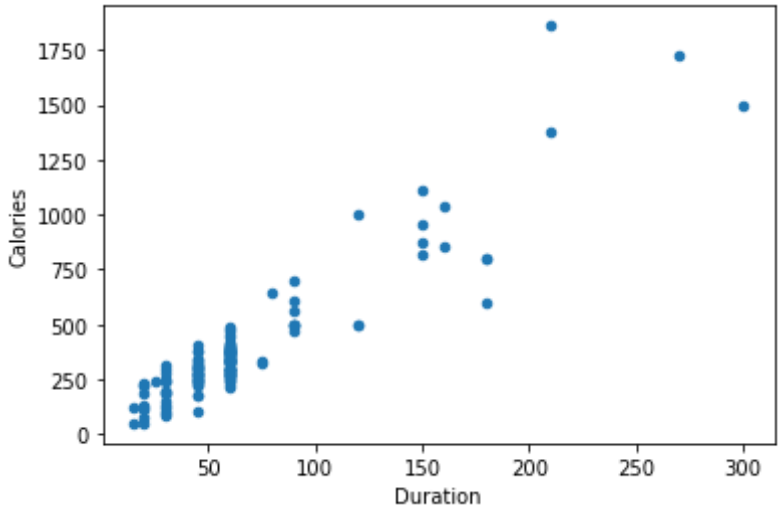
Input:

```
#K

#Visualizing using scatter plot for the two columns (Duration and Calories).
import matplotlib.pyplot as plt
print(df.plot.scatter(x ='Duration', y= 'Calories'))
plt.show()
```

Output:

```
Axes(0.125,0.125;0.775x0.755)
```



QUESTION 2: Linear Regression

a) **Import the given "Salary_Data.csv"**

Input:

C:\Users\irrin\OneDrive\Desktop\neural\NN&DL_Assignment4_Q2Linear Regression.py

NNDL_Assignment4_Q2Linear Regression.py

```python
#2 QUESTION. Linear Regression

# Importing the libraries

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error


# a) Importing the Salarydatasets

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

#excluding last column
X = salaryData.iloc[:, :-1].values
#salary
Y = salaryData.iloc[:, 1].values
salaryData.info()
salaryData.head()

```

Output:

```
In [1]: runfile('C:/Users/irrin/OneDrive/Desktop/neural/NN&DL_Assignment4_Q2Linear Regression.py',
wdir='C:/Users/irrin/OneDrive/Desktop/neural')
<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     int64
dtypes: float64(1), int64(1)
memory usage: 612.0 bytes
```

Index	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2	43525
4	2.2	39891
5	2.9	56642
6	3	60150
7	3.2	54445
8	3.2	64445
9	3.7	57189
10	3.9	63218

b) Split the data in train_test partitions, such that 1/3 of the data is reserved as test subset.

Input:

```
#b) Splitting the dataset into the Training set and Test set

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=1/3, random_state=0)

print("Below is the Split Data:")
print("Train features:")
print(pd.DataFrame(X_train).head())
print("Train targets:")
print(pd.DataFrame(Y_train).head())
print("Test features:")
print(pd.DataFrame(X_test).head())
print("Test targets:")
print(pd.DataFrame(Y_test).head())
```

Output:

```
Below is the Split Data:
Train features:
     0
0  2.9
1  5.1
2  3.2
3  4.5
4  8.2
Train targets:
        0
0   56642
1   66029
2   64445
3   61111
4  113812
Test features:
      0
0   1.5
1  10.3
2   4.1
3   3.9
4   9.5
Test targets:
        0
0   37731
1  122391
2   57081
3   63218
4  116969
```

c) Train and predict the model.

Input:

```
# c) Train and predict the model

model = LinearRegression()
model.fit(X_train, Y_train)
Y_pred = model.predict(X_test)
print("Predicted values:", Y_pred)
```

Output:

```
Predicted values: [ 40835.10590871 123079.39940819  65134.55626083  63265.36777221
 115602.64545369 108125.8914992  116537.23969801  64199.96201652
  76349.68719258 100649.1375447 ]
```

d) Calculate the mean_squared error

Input:

```
#d) Calculate the mean_squared error

mse = mean_squared_error(Y_test, Y_pred)
print("Mean Squared Error:", mse)
```

Output:

```
Mean Squared Error: 21026037.329511296
```

e) Visualize both train and test data using scatter plot.

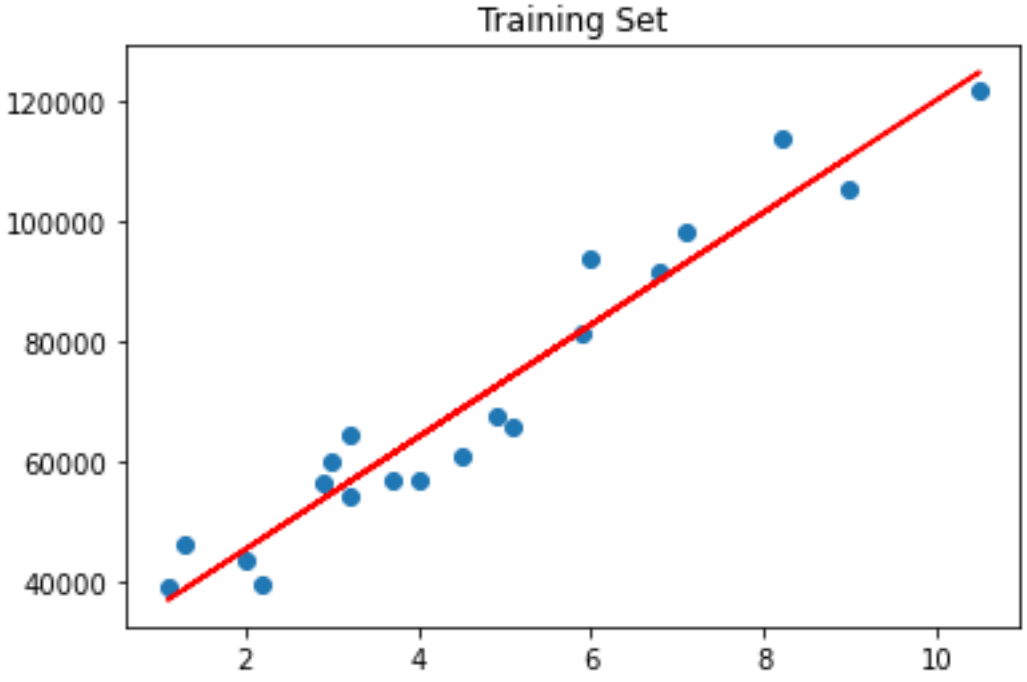
Input:

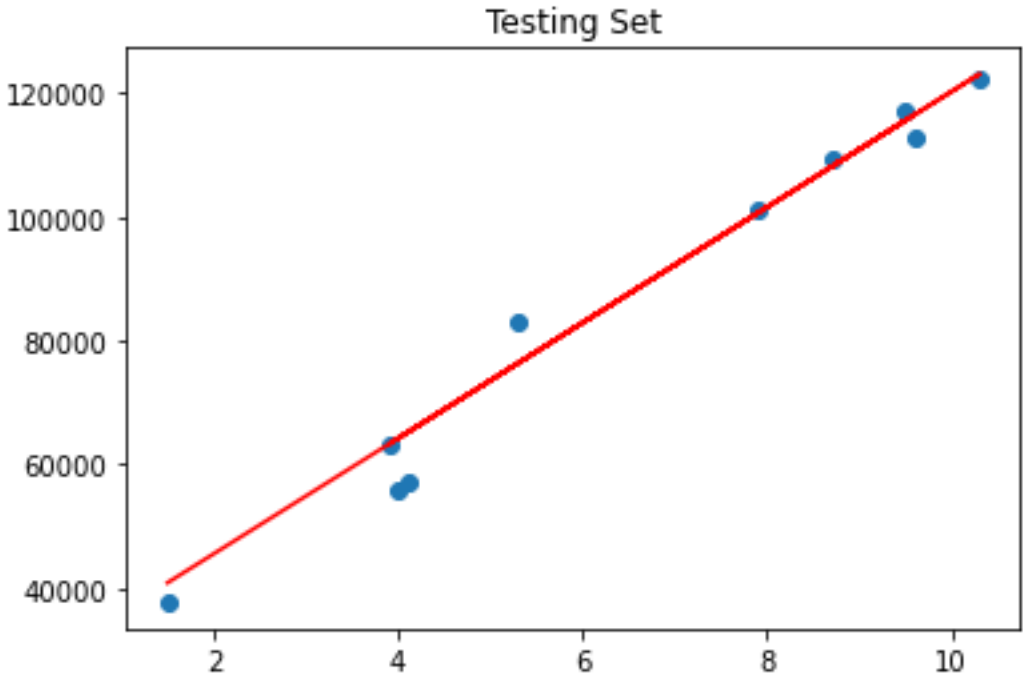
```
#e) Visualize both train and test data using scatter plot.
#The scatter function is used to plot the training data, and the plot function is used to plot the predicted values


# Training Data
plt.scatter(X_train, Y_train)
plt.plot(X_train, model.predict(X_train), color='red')
plt.title('Training Set')
plt.show()

# Testing Data
plt.scatter(X_test, Y_test)
plt.plot(X_test, model.predict(X_test), color='red')
plt.title('Testing Set')
plt.show()
```

Output:


Training Set
120000
100000
80000
60000
40000
2
4
6
8
10


Testing Set
120000
100000
80000
60000
40000
2
4
6
8
10