LOVELY PROFESSIONAL UNIVERSITY

Course Name: Machine Learning

Course Code: CAP 781

Lab Evaluation: 1

Submitted to: Dr. Amar Singh

Submitted by: Menka Kalsi

Reg. No.: 12103299

Roll No.: RDE543B47

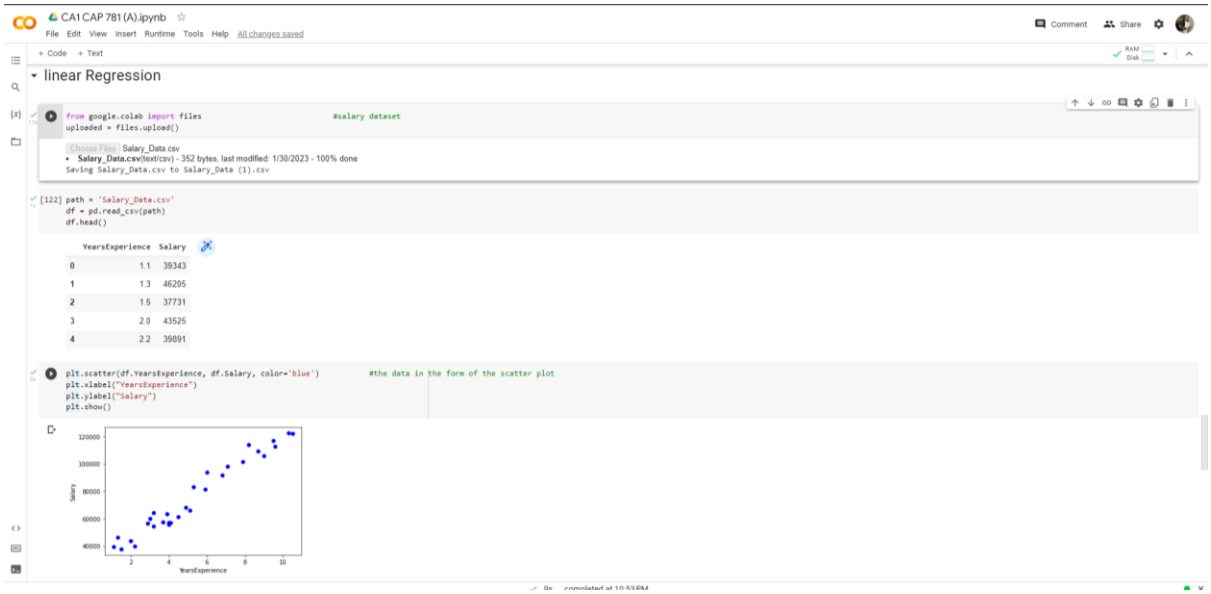
Paper Code: A

Que. 1: Apply following operations on Salary dataset.

1. **Plot the outputs with respect to input and identify that the dataset is linear or nonlinear.**
2. **Print Maximum, minimum, and average Salaries.**

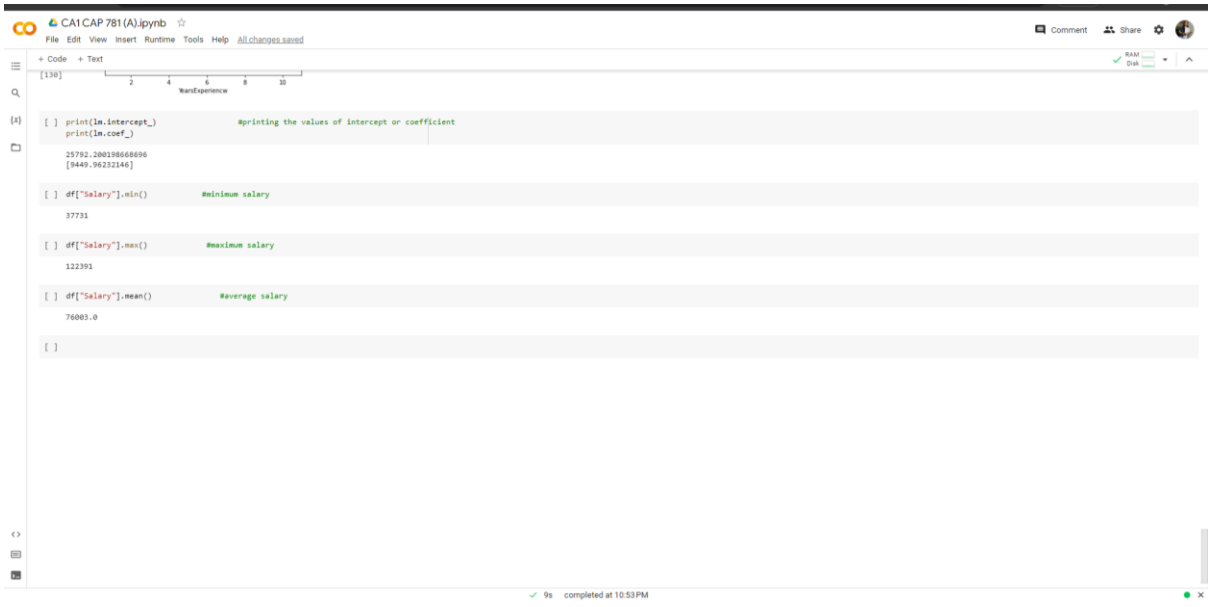
1.

- Importing the Salary dataset and the input in the experience or output is the salary.
- After importing plotted the scattered graph and finding that data is scattered in one direction.
- Here, we can say that dataset is linear.



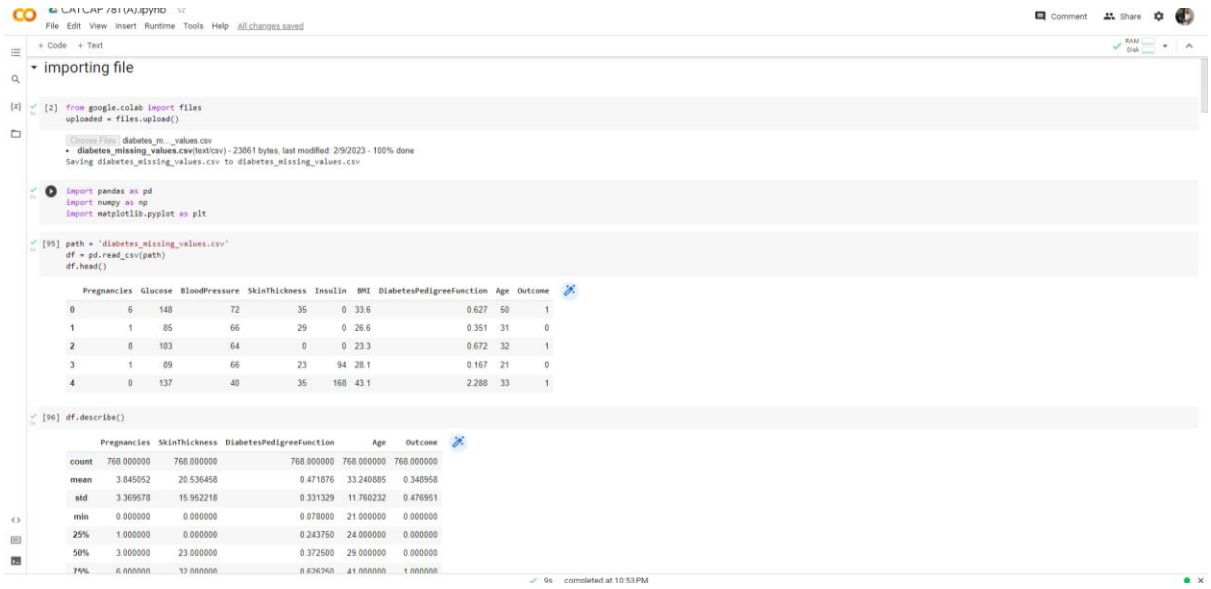
- Initially imported a file named linear regression from inbuilt library SKLEARN for finding the best possible solution.
- Assigned the variable to the input data or output data and after we fitted the linear model.
- Then, calculated intercept and coefficient of the dataset
- Then after writing the suitable lines of code the best fitted line for the Salary dataset is plotted into the graph

2. Printing the maximum, minimum and average salary.



Que. 2. Identify the missing values from diabetes dataset and take the appropriate actions against missing values. Also identify the first three highly correlated columns with the outcome column.

- For missing values using the isnull method and for special symbols using replace function



CA1 CAP 781 (A).ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

importing file

```
[2] from google.colab import files
    uploaded = files.upload()
```

Choose Files diabetes_m...values.csv

- diabetes_missing_values.csv(text/csv) - 23861 bytes, last modified: 2/9/2023 - 100% done

Saving diabetes_missing_values.csv to diabetes_missing_values.csv

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

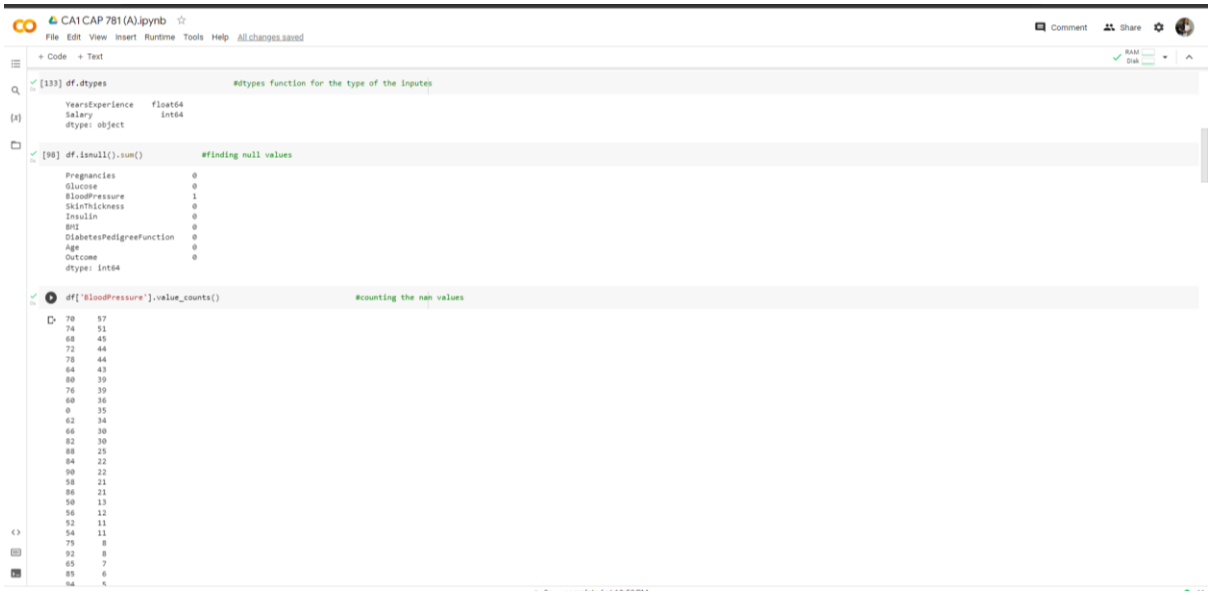
```
[95] path = 'diabetes_missing_values.csv'
     df = pd.read_csv(path)
     df.head()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

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

	Pregnancies	SkinThickness	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	20.536458	0.471876	33.240885	0.348958
std	3.369578	15.952218	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	0.000000	0.243750	24.000000	0.000000
50%	3.000000	23.000000	0.372500	29.000000	0.000000
75%	6.000000	32.000000	0.626250	41.000000	1.000000

9s completed at 10:53 PM



CA1 CAP 781 (A).ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

```
[133] df.dtypes                    #dtypes function for the type of the inputes
```

```
YearsExperience    float64
Salary               int64
dtype: object
```

```
[98] df.isnull().sum()          #finding null values
```

```
Pregnancies                 0
Glucose                     0
BloodPressure               1
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

```
df['BloodPressure'].value_counts()                    #counting the nan values
```

```
70    57
74    51
68    45
72    44
78    44
64    43
80    39
76    39
60    36
0     35
62    34
66    30
82    30
88    25
84    22
90    22
58    21
86    21
50    13
56    12
52    11
54    11
75     8
92     8
65     7
85     6
```

CA1 CAP 781 (A).ipynb

File Edit View Insert Runtime Tools Help All changes saved

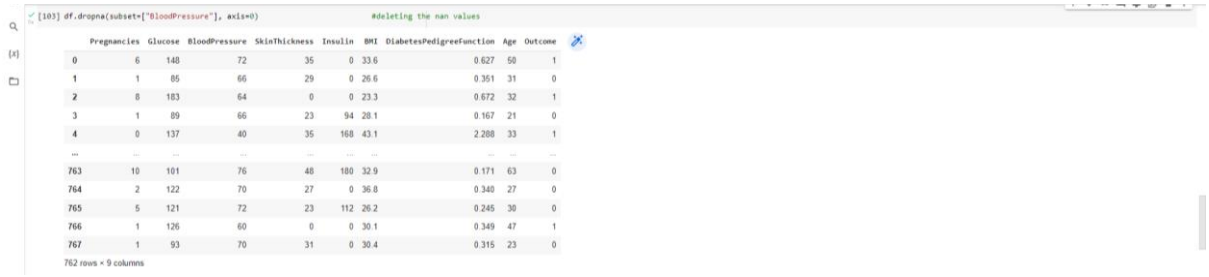
Comment Share

+ Code + Text

```
[100] df.replace("?", np.nan, inplace = True)          #replacing the spacial characters with nan values
      df.head()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

- For deleting using the dropna function

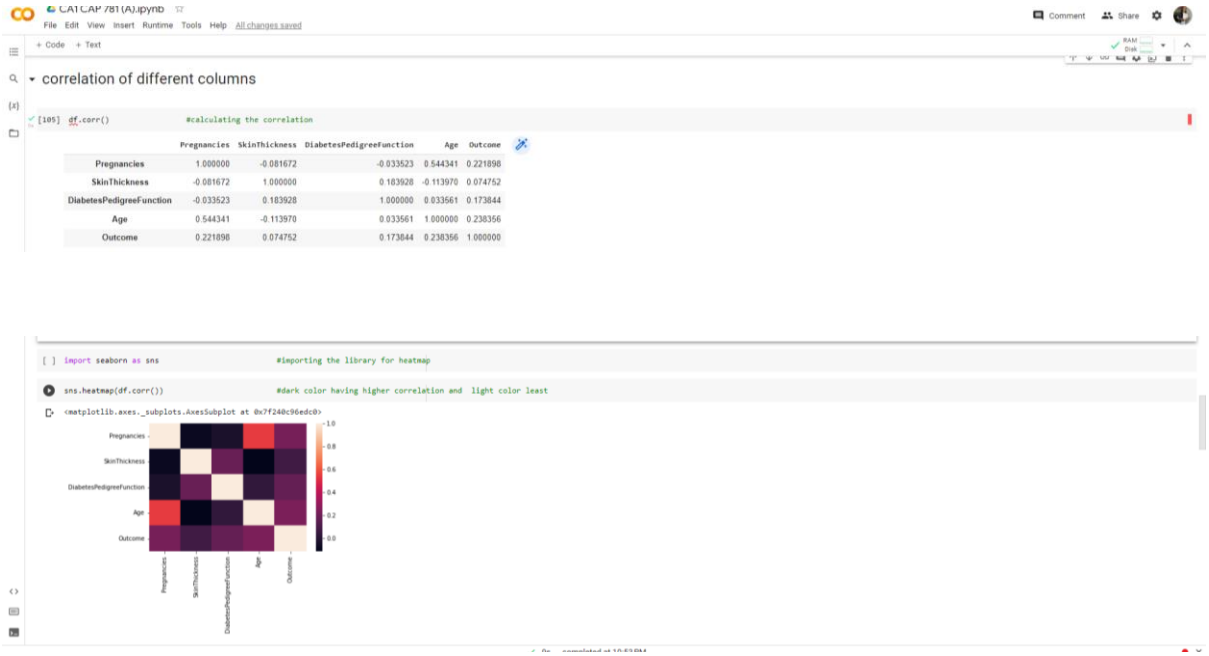


```
[103] df.dropna(subset=["BloodPressure"], axis=0)                    #deleting the nan values
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

762 rows × 9 columns

- Calculating Correlation



CAT CAP 78T (A).ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk

correlation of different columns

```
[105] df.corr()                    #calculating the correlation
```

	Pregnancies	SkinThickness	DiabetesPedigreeFunction	Age	Outcome
Pregnancies	1.000000	-0.081672	-0.033523	0.544341	0.221898
SkinThickness	-0.081672	1.000000	0.183928	-0.113970	0.074752
DiabetesPedigreeFunction	-0.033523	0.183928	1.000000	0.033561	0.173844
Age	0.544341	-0.113970	0.033561	1.000000	0.238356
Outcome	0.221898	0.074752	0.173844	0.238356	1.000000

```
[ ] import seaborn as sns                    #importing the library for heatmap

sns.heatmap(df.corr())                    #dark color having higher correlation and  light color least

<matplotlib.axes._subplots.AxesSubplot at 0x7f240c96edc0>
```


Que.3. Normalize diabetes dataset using minmax and feature scaling methods.

- Normalizing the dataset using Feature scaling and minmax method.

normalizing the data set using min-max and feature scalling method

```
Z = df[['BloodPressure', 'BMI', 'Age']]                #normalizing the data using min-max and feature scallingmethod

[110] featureset = df[['BloodPressure', 'BMI', 'Age']]

[132] from sklearn.preprocessing import MinMaxScaler
      x = featureset.values                                    #calculating the min - max  and feature scalling
      min_max_scaler = MinMaxScaler()
      feature_mtx = min_max_scaler.fit_transform(x)
      Z = feature_mtx
      feature_mtx [0:5]

      array([[0.59016393, 0.50074516, 0.48333333],
             [0.54098361, 0.39642325, 0.16666667],
             [0.52459016, 0.34724292, 0.18333333],
             [0.54098361, 0.41877794, 0.        ],
             [0.32786885, 0.64232489, 0.2       ]])
```