U18ISI6204 – Machine Learning Techniques

LAB EXPERIMENT 4

NAME: Aaron Mathew

ROLL_NO: 20BIS001

Write a Program to implement Logistic Regression by plotting the decision boundary and use it to classify spam mail

INTRODUCTION

In this experiment, we have to perform Logistic regression on the covid dataset.

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, **logistic regression** (or **logit regression**) is estimating the parameters of a logistic model (a form of binary regression).

Linear Regression Equation:

$y = \beta 0 + \beta 1X1 + \beta 2X2 + \ldots\ldots + \beta nXn$

Where, y is dependent variable and x1, x2 ... and Xn are explanatory variables.

Sigmoid Function:

$p = 1/1 + e^{-y}$

Apply Sigmoid function on linear regression:

$$p = 1/1 + e^{-(\beta 0+\beta 1X1+\beta 2X2\ldots\ldots\beta nXn)}$$

OBJECTIVE OF THE EXERCISE/EXPERIMENT

To perform Logistic regression on the given dataset, using scikit library

ACQUISITION PROCEDURE:

STEP-1: Start the program.

STEP-2: import all the necessary libraries

i) Numpy – array manipulation
ii) Pandas – dataframe manipulation
iii) Matplotlib and seaborn – for data visualization
iv) Sklearn.model_selection – train test data split
v) Sklearn.metrics – f1 score.
vi) Sklearn,linear_model – for logistic regression

STEP-3: Loading the dataset using read_csv method in pandas module.

STEP-4: Analyze the dataset using info method, which gives its data types and number of non- null values in each columns.

STEP-5: Perform basic statistic operation using describe() method.

STEP-6: Use heatmaps, correlation matrix, regression plots and pairplots in seaborn to find the relationship between features.

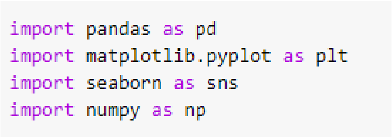
STEP-7: Implement Logistics regression(logreg) with all variable and calculate the f1 score.

STEP-8: Stop the program.

PROGRAM:

Importing libraries

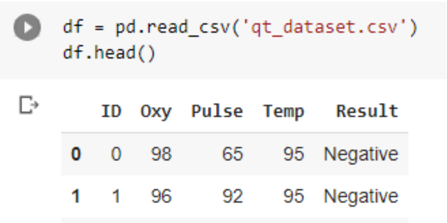
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```



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

Loading dataset

```
df = pd.read_csv('qt_dataset.csv')
df.head()
```



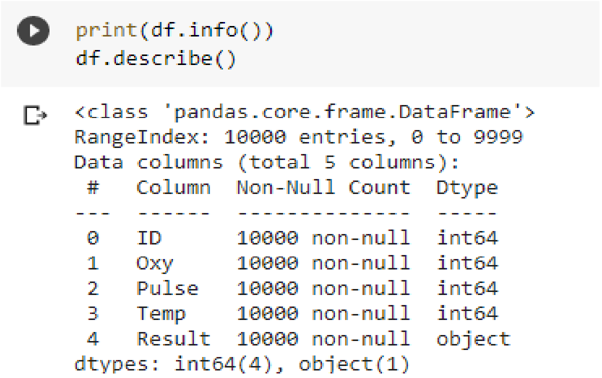
```
df = pd.read_csv('qt_dataset.csv')
df.head()
```

	ID	Oxy	Pulse	Temp	Result
0	0	98	65	95	Negative
1	1	96	92	95	Negative

Basic statistics operations

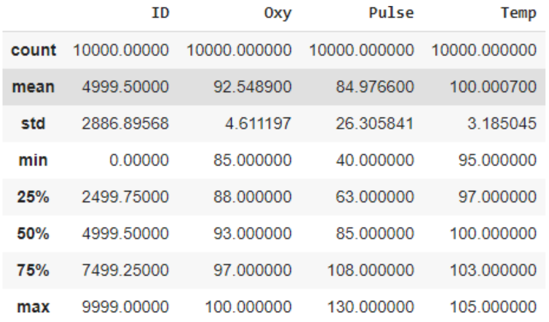
Printf(df.info())

df.describe()



```
print(df.info())
df.describe()
```

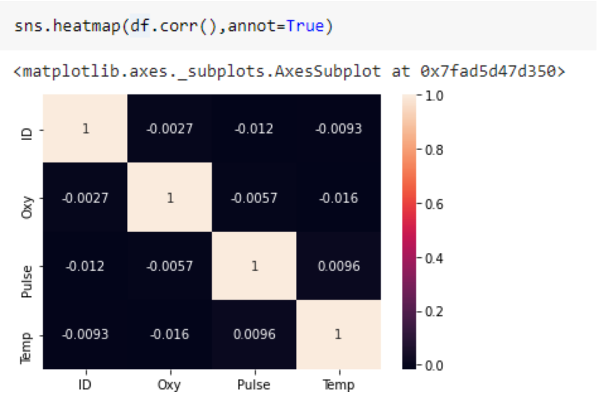
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   ID      10000 non-null  int64 
 1   Oxy     10000 non-null  int64 
 2   Pulse   10000 non-null  int64 
 3   Temp    10000 non-null  int64 
 4   Result  10000 non-null  object
dtypes: int64(4), object(1)
```



	ID	Oxy	Pulse	Temp
count	10000.00000	10000.000000	10000.000000	10000.000000
mean	4999.50000	92.548900	84.976600	100.000700
std	2886.89568	4.611197	26.305841	3.185045
min	0.00000	85.000000	40.000000	95.000000
25%	2499.75000	88.000000	63.000000	97.000000
50%	4999.50000	93.000000	85.000000	100.000000
75%	7499.25000	97.000000	108.000000	103.000000
max	9999.00000	100.000000	130.000000	105.000000

Correlation between columns

Sns.heatmap(df.corr(), annot = True)



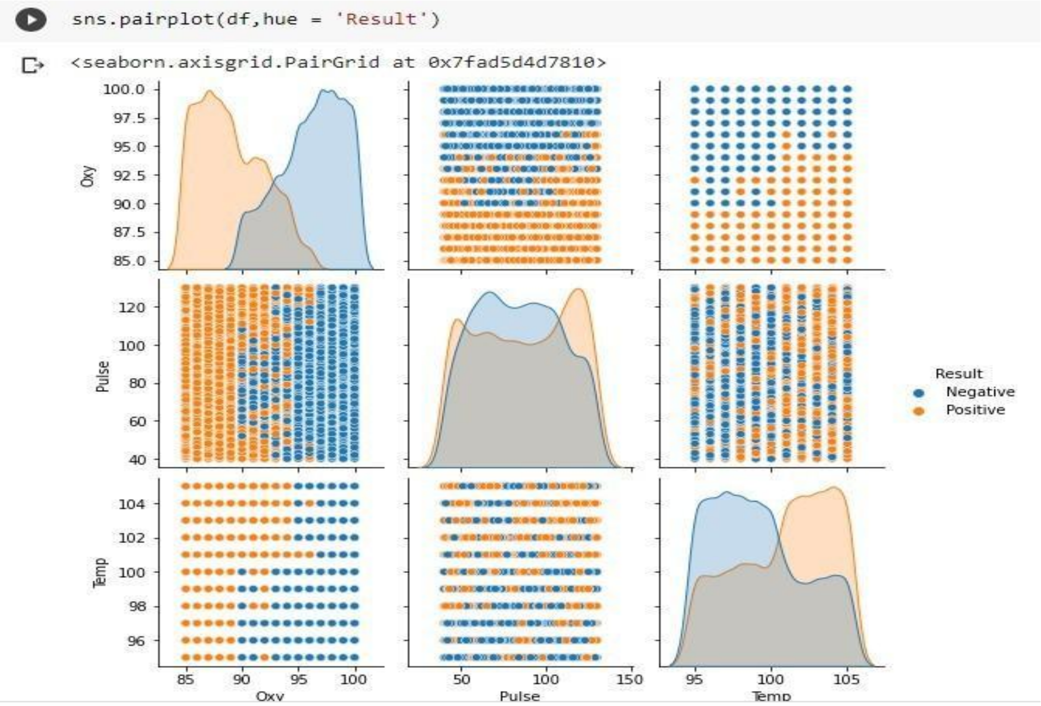
```
sns.heatmap(df.corr(),annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fad5d47d350>
```

	ID	Oxy	Pulse	Temp
ID	1	-0.0027	-0.012	-0.0093
Oxy	-0.0027	1	-0.0057	-0.016
Pulse	-0.012	-0.0057	1	0.0096
Temp	-0.0093	-0.016	0.0096	1

Pairplots.

Sns.pairplot(df,hue = ‘Result’)



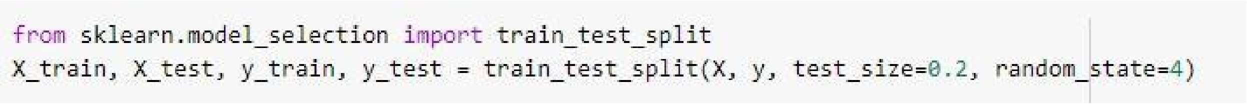
```
sns.pairplot(df,hue = 'Result')
```

```
<seaborn.axisgrid.PairGrid at 0x7fad5d4d7810>
```

Train test split.

from sklearn.model_selection import train_test_split

X_train, X_test, y_train,y_test = train_test_split(X,y,test_size=0.2, random_state =4)

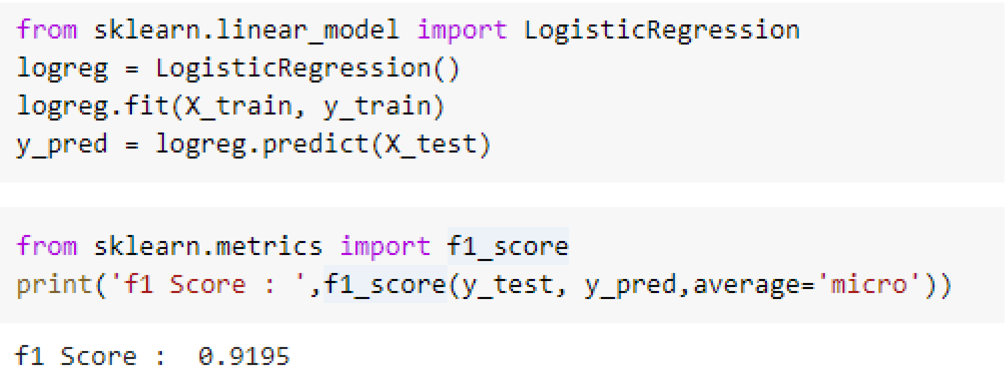


```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=4)
```

Logistic regression :

From sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
y_pred = logreg.predict(X_test)

from sklearn.metrics import f1_score
print(‘f1 Score :’ ,f1_score(y_test,y_pred,average =’micro’))



```
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
y_pred = logreg.predict(X_test)
```

```
from sklearn.metrics import f1_score
print('f1 Score : ',f1_score(y_test, y_pred,average='micro'))
```

```
f1 Score :  0.9195
```