Confusion Matrix in Machine Learning

The confusion matrix:
The confusion matrix (Kohavi and Provost, 1998) contains details about what is real and predicted sections made by classification systems. The effectiveness of such programs is often tested using data in the matrix.One of the most used tools for evaluating the output and understanding the effectiveness of a binary or categorical classifier is the Confusion Matrix.Confusion matrix, also known as an error matrix.

It has four dimensions:
1. True Positive (TP)
2. True Negative (TN)
3. False Positive (FP)
4. False Negative (FN)

True Positives (TP) – It is the case when both actual class & predicted class of data point is 1.
True Negatives (TN) – It is the case when both actual class & predicted class of data point is 0.
False Positives (FP) – It is the case when the actual class of data point is 0 & predicted class of data point is 1.
False Negatives (FN) – It is the case when the actual class of data point is 1 & predicted class of data point is 0.

Measure Terms in Confusion Matrix:
• **Accuracy: –** It is how close a measured value to the actual (True) value.
• **Precision: –** It is how close the measured values are to each other.

Precision = TP / Predicted Yes

• **Recall: –** It is the ratio of all correctly predicted positive predictions

Recall = TP / Actual Yes

• **Error Rate: –** It is calculated as the number of all incorrect predictions divided by the total number of the datasets. The best error rate is 0.0 – The worst error rate is 1.0.

Error Rate = 1 - Accuracy = (FN + FP) / Total

Implement confusion matrix:
Implement confusion matrix on diabetes dataset to classify the outcome based on binary classification (i.e. 1= has diabetes, 0= has no diabetes) as follows,

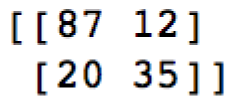
Import necessary packages and dataset
Such as sklearn.linear_model,sklearn.datasets,pandas,sklearn.model_selection,etc.

```
from sklearn.linear_model import LogisticRegression
from sklearn.datasets import load_diabetes
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
data=pd.read_csv('diabetes.csv')#available on Kaggle
data.head()
x=data.iloc[:,:-1]
y=data.iloc[:,-1]
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.
20,random_state=32)
```

Build Model, Predict and Generate the confusion matrix:
Train and test the LogisticRegression model for given dataset and plot confusion matrix as follows,

```
#Fit the model
Model = LogisticRegression()
Model.fit(x_train,y_train)
#Generate predictions with the model using our x_test values
y_pred = Model.predict(x_test)
#Get the confusion matrix
cnf_matrix = confusion_matrix(y_test, y_pred)
print(cnf_matrix)
```

Output:

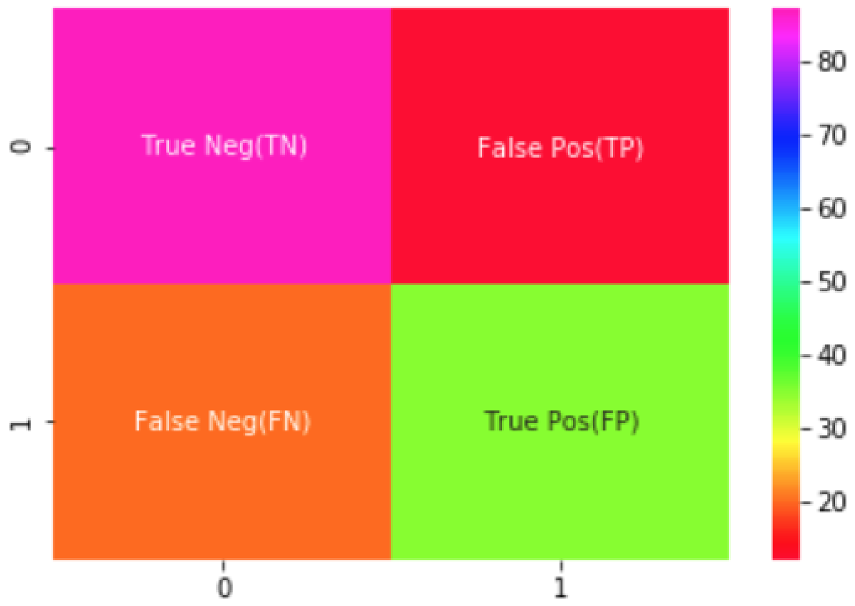


```
[[87 12]
 [20 35]]
```

Plot confusion matrix using seaborn for better visualization and understating purpose.

```
import seaborn as sns
import numpy as np
labels = ['True Neg(TN)','False Pos(TP)','False Neg(FN)','True
Pos(FP)']
labels = np.asarray(labels).reshape(2,2)
sns.heatmap(cnf_matrix, annot=labels,
fmt='',cmap='gist_rainbow')
```

Output:



Evaluation:

Finding accuracy, precision , recall etc, using classification_report as follows,

```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

Output:

```
              precision    recall  f1-score   support

           0       0.81      0.88      0.84        99
           1       0.74      0.64      0.69        55

    accuracy                           0.79       154
   macro avg       0.78      0.76      0.77       154
weighted avg       0.79      0.79      0.79       154
```