Importing Necessary Libraries

In [22]:

```
import pandas as pd
from pandas.plotting import scatter_matrix
import numpy as np
import matplotlib.pyplot as plt
import os
from imblearn.over_sampling import ADASYN
from collections import Counter
import seaborn as sn
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import BernoulliNB
from sklearn import metrics
%matplotlib inline
sn.set_style("ticks")
sn.set_palette("BuGn_r")

import matplotlib.pyplot as plt
import numpy as np
from sklearn import metrics

def plot_confusion_matrix(cm, classes, title, cmap):
    "plotting confusion matrix"
    plt.clf()
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    classnames = classes
    plt.title(title)
    plt.ylabel('True')
    plt.xlabel('Predicted')
    tick_marks = np.arange(len(classnames))
    plt.xticks(tick_marks, classnames, rotation=0)
    plt.yticks(tick_marks, classnames)
    s = [['TN','FP'], ['FN', 'TP']]

    for i in range(2):
        for j in range(2):
            plt.text(j,i, str(s[i][j])+" = "+str(cm[i][j]))
    plt.show()
```

Importing Data into Dataset and Read the CSV Data Info

In [2]:

```
df = pd.read_csv("https://datahub.io/machine-learning/creditcard/r/creditcard.csv")
print('The dataset contains {0} rows and {1} columns.'.format(df.shape[0], df.shape[1]))
print('Normal transactions count: ', df['Class'].value_counts().values[0])
print('Fraudulent transactions count: ', df['Class'].value_counts().values[1])
```

```
The dataset contains 284807 rows and 31 columns.
Normal transactions count:  284315
Fraudulent transactions count:  492
```

Preparing The Dataset

In [4]:

```
X = df.iloc[:, :-1]
y = df['Class']
scaler = StandardScaler()
scaled_X = scaler.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(scaled_X, y, test_size=0.20, random_st
ate=42)
```

Resampling Dataset Shape

In [5]:

```
ada = ADASYN(random_state=42)
X_res, y_res = ada.fit_sample(X_train, y_train)
```

In [6]:

```
X_train, y_train = X_res, y_res
# Regression
LGR_Classifier = LogisticRegression()
LGR_Classifier.fit(X_train, y_train);
```

```
C:\Users\prana\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
32: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify
a solver to silence this warning.
  FutureWarning)
```

Model

In [7]:

```
modl = [('LogisticRegression', LGR_Classifier)]
models = [md for md in modl]
```

Printing Results

In [23]:

```python
print()
for i,v in models:
    scores = cross_val_score(v, X_train, y_train, cv=10)
    accuracy = metrics.accuracy_score(y_train, v.predict(X_train))
    confusion_matrix = metrics.confusion_matrix(y_train, v.predict(X_train))
    classification = metrics.classification_report(y_train, v.predict(X_train))
    print('===== Logistic Regression =====')
    print()
    print ("Mean Score: ", '{}%'.format(np.round(scores.mean(), 3) * 100))
    print()
    print ("Model Accuracy: ", '{}%'.format(np.round(accuracy, 3) * 100))
    print()
    print("Confusion Matrix:" "\n", confusion_matrix)
    print()
    print("Classification Report:" "\n", classification)
    print()
```

```
C:\Users\prana\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
32: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify
a solver to silence this warning.
  FutureWarning)
C:\Users\prana\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
32: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify
a solver to silence this warning.
  FutureWarning)
C:\Users\prana\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
32: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify
a solver to silence this warning.
  FutureWarning)
C:\Users\prana\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
32: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify
a solver to silence this warning.
  FutureWarning)
C:\Users\prana\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
32: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify
a solver to silence this warning.
  FutureWarning)
C:\Users\prana\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
32: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify
a solver to silence this warning.
  FutureWarning)
C:\Users\prana\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
32: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify
a solver to silence this warning.
  FutureWarning)
C:\Users\prana\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
32: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify
a solver to silence this warning.
  FutureWarning)
C:\Users\prana\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
32: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify
a solver to silence this warning.
  FutureWarning)
C:\Users\prana\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
32: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify
a solver to silence this warning.
  FutureWarning)
```

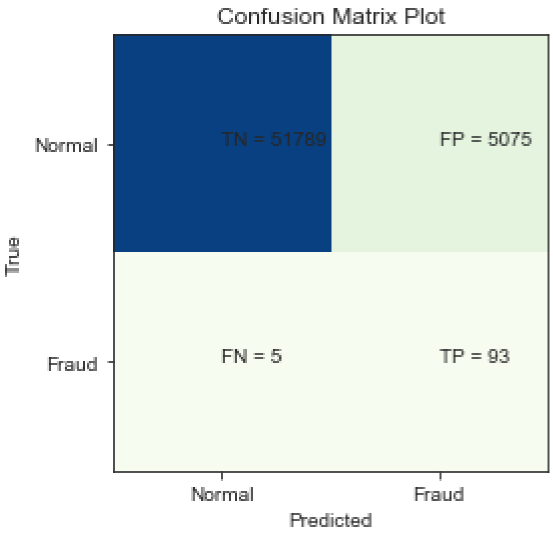
```
===== Logistic Regression =====

Mean Score:  87.8%

Model Accuracy:  89.9%

Confusion Matrix:
 [[207559  19892]
 [ 26089 201369]]

Classification Report:
               precision    recall  f1-score   support

           0       0.89      0.91      0.90    227451
           1       0.91      0.89      0.90    227458

    accuracy                           0.90    454909
   macro avg       0.90      0.90      0.90    454909
weighted avg       0.90      0.90      0.90    454909
```

Testing Model and Plotting Graph

In [24]:

```
classnf = {'Normal':0, 'Fraud':1}
print()
print('************************** Model Test Results **************************' "\n")

for i, v in models:
    accuracy = metrics.accuracy_score(y_test, v.predict(X_test))
    confusion_matrix = metrics.confusion_matrix(y_test, v.predict(X_test))
    classification = metrics.classification_report(y_test, v.predict(X_test))
    print('********** Logistic Regression **********')
    print ("Model Accuracy: ",  '{}%'.format(np.round(accuracy, 3) * 100))
    print()
    print("Confusion Matrix:" "\n", confusion_matrix)
    print()
    print("Matrix Plot : ")
    plot_confusion_matrix(confusion_matrix, classes = list(classnf.keys()), title='Confusion Matrix Plot', cmap=plt.cm.GnBu)
    print()
    print("Classification Report:" "\n", classification)
    print()
```

```
************************** Model Test Results **************************

********** Logistic Regression **********
Model Accuracy:  91.10000000000001%

Confusion Matrix:
 [[51789  5075]
 [    5    93]]

Matrix Plot :
```



```
Classification Report:
               precision    recall  f1-score   support

           0       1.00      0.91      0.95     56864
           1       0.02      0.95      0.04        98

    accuracy                           0.91     56962
   macro avg       0.51      0.93      0.49     56962
weighted avg       1.00      0.91      0.95     56962
```