In [1]:

```
import pandas as pd
import os
import joblib as jb
import sklearn
import pydotplus
```

In [2]:

```
train_data=pd.read_excel('Training_Updated.xlsx')
valid_data=test_data=pd.read_excel('Validation_final.xlsx')
test_data=pd.read_excel('Test_final.xlsx')
```

In [3]:

```
train_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32561 entries, 0 to 32560
Data columns (total 15 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   AgeCategory           32561 non-null  int64
 1   Workclass             32561 non-null  int64
 2   Education             32561 non-null  int64
 3   EducationNum          32561 non-null  int64
 4   MaritalStatus         32561 non-null  int64
 5   Occupation            32561 non-null  int64
 6   Relationship          32561 non-null  int64
 7   Sex                   32561 non-null  int64
 8   NativeCountry         32561 non-null  int64
 9   Race                  32561 non-null  int64
 10  FnlwgtCategory        32561 non-null  int64
 11  CapitalGainCategory   32561 non-null  int64
 12  CapitalLossCategory   32561 non-null  int64
 13  HoursPerWeekCategory  32561 non-null  int64
 14  Class                 32561 non-null  int64
dtypes: int64(15)
memory usage: 3.7 MB
```

In [4]:

```
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
```

In [5]:

```
X_train=train_data.drop(['Class'],axis=1)
Y_train=train_data['Class']

X_valid=valid_data.drop(['Class'],axis=1)
Y_valid=valid_data['Class']

X_test=test_data.drop(['Class'],axis=1)
Y_test=test_data['Class']
```

In [8]:

```
from sklearn.preprocessing import StandardScaler
```

In [32]:

```
clf = StandardScaler()
```

In [33]:

```
X_train_scaled = clf.fit_transform(X_train)
X_valid_scaled = clf.fit_transform(X_valid)
X_test_scaled = clf.fit_transform(X_test)
```

In [34]:

```
from sklearn.linear_model import LogisticRegression
```

In [35]:

```
clf = LogisticRegression(random_state=0).fit(X_train_scaled,Y_train)
```

In [36]:

```
Y_train_pred=clf.predict(X_train_scaled)

Y_valid_pred=clf.predict(X_valid_scaled)

Y_test_pred=clf.predict(X_test_scaled)
```

In [37]:

```
from sklearn import metrics,model_selection,preprocessing
wrong_train_pred=(Y_train !=Y_train_pred).sum()
print("Total wrong detected on training data= {}".format(wrong_train_pred))

accuracy_train=metrics.accuracy_score(Y_train,Y_train_pred)
print("Accuracy of this model on training data= {:.3f}".format(accuracy_train))
```

```
Total wrong detected on training data= 5852
Accuracy of this model on training data= 0.820
```

In [38]:

```
wrong_valid_pred=(Y_valid !=Y_valid_pred).sum()
print("Total wrong detected on validation data = {}".format(wrong_valid_pred))

accuracy_valid=metrics.accuracy_score(Y_valid,Y_valid_pred)
print("Accuracy of this model on validation data = {:.3f}".format(accuracy_valid))
```

```
Total wrong detected on validation data = 1469
Accuracy of this model on validation data = 0.820
```

In [39]:

```
wrong_test_pred=(Y_test !=Y_test_pred).sum()
print("Total wrong detected on test data = {}".format(wrong_test_pred))

accuracy_test=metrics.accuracy_score(Y_test,Y_test_pred)
print("Accuracy of this model on test data = {:.3f}".format(accuracy_test))
```

```
Total wrong detected on test data = 1464
Accuracy of this model on test data = 0.820
```

In [40]:

```
clf = LogisticRegression(random_state=0, C=1, fit_intercept = True).fit(X_train_scaled,Y
```

In [41]:

```
Y_train_pred=clf.predict(X_train_scaled)

Y_valid_pred=clf.predict(X_valid_scaled)

Y_test_pred=clf.predict(X_test_scaled)
```

In [42]:

```
from sklearn import metrics,model_selection,preprocessing
wrong_train_pred=(Y_train !=Y_train_pred).sum()
print("Total wrong detected on training data= {}".format(wrong_train_pred))

accuracy_train=metrics.accuracy_score(Y_train,Y_train_pred)
print("Accuracy of this model on training data= {:.3f}".format(accuracy_train))
```

```
Total wrong detected on training data= 5852
Accuracy of this model on training data= 0.820
```

In [43]:

```
wrong_valid_pred=(Y_valid !=Y_valid_pred).sum()
print("Total wrong detected on validation data = {}".format(wrong_valid_pred))

accuracy_valid=metrics.accuracy_score(Y_valid,Y_valid_pred)
print("Accuracy of this model on validation data = {:.3f}".format(accuracy_valid))
```

```
Total wrong detected on validation data = 1469
Accuracy of this model on validation data = 0.820
```

In [29]:

```
wrong_test_pred=(Y_test !=Y_test_pred).sum()
print("Total wrong detected on test data = {}".format(wrong_test_pred))

accuracy_test=metrics.accuracy_score(Y_test,Y_test_pred)
print("Accuracy of this model on test data = {:.3f}".format(accuracy_test))
```

```
Total wrong detected on test data = 1464
Accuracy of this model on test data = 0.820
```

In [ ]: