In [1]:

```
import numpy as np
import pandas as pd
```

In [2]:

```
dataset=pd.read_csv('adult.csv',header=None)
x=dataset.iloc[:,:-1].values
y=dataset.iloc[:,-1].values
```

In [3]:

```
print(x)
```

```
[[39 ' State-gov' 77516 ... 0 40 ' United-States']
 [50 ' Self-emp-not-inc' 83311 ... 0 13 ' United-States']
 [38 ' Private' 215646 ... 0 40 ' United-States']
 ...
 [58 ' Private' 151910 ... 0 40 ' United-States']
 [22 ' Private' 201490 ... 0 20 ' United-States']
 [52 ' Self-emp-inc' 287927 ... 0 40 ' United-States']]
```

In [5]:

```
from sklearn.impute import SimpleImputer
imputer=SimpleImputer(missing_values=np.nan,strategy='most_frequent')
imputer.fit(x[:,1:])
x[:,1:]=imputer.transform(x[:,1:])
```

In [7]:

```
from sklearn.preprocessing import LabelEncoder
le1=LabelEncoder()
le3=LabelEncoder()
le5=LabelEncoder()
le6=LabelEncoder()
le7=LabelEncoder()
le8=LabelEncoder()
le9=LabelEncoder()
le13=LabelEncoder()
le=LabelEncoder()
x[:,1]=le1.fit_transform(x[:,1])
x[:,3]=le3.fit_transform(x[:,3])
x[:,5]=le5.fit_transform(x[:,5])
x[:,6]=le6.fit_transform(x[:,6])
x[:,7]=le7.fit_transform(x[:,7])
x[:,8]=le8.fit_transform(x[:,8])
x[:,9]=le9.fit_transform(x[:,9])
x[:,13]=le13.fit_transform(x[:,13])
y=le.fit_transform(y)
```

In [8]:

```
print(x)
```

```
[[39 7 77516 ... 0 40 39]
 [50 6 83311 ... 0 13 39]
 [38 4 215646 ... 0 40 39]
 ...
 [58 4 151910 ... 0 40 39]
 [22 4 201490 ... 0 20 39]
 [52 5 287927 ... 0 40 39]]
```

In [9]:

```
print(y)
```

```
[0 0 0 ... 0 0 1]
```

In [11]:

```
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=0)
```

In [12]:

```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
X_train=sc.fit_transform(X_train)
X_test=sc.transform(X_test)
```

In [14]:

```
from xgboost import XGBClassifier
classifier=XGBClassifier()
classifier.fit(X_train,Y_train)
```

Out[14]:

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
              importance_type='gain', interaction_constraints='',
              learning_rate=0.300000012, max_delta_step=0, max_depth=6,
              min_child_weight=1, missing=nan, monotone_constraints='()',
              n_estimators=100, n_jobs=0, num_parallel_tree=1, random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
              tree_method='exact', validate_parameters=1, verbosity=None)
```

In [16]:

```
from sklearn.metrics import confusion_matrix,accuracy_score
y_pred=classifier.predict(X_test)
cm=confusion_matrix(Y_test,y_pred)
print(cm)
accuracy_score(Y_test,y_pred)
```

```
[[4559  359]
 [ 527 1068]]
```

Out[16]:

```
0.8639643789344388
```

In [17]:

```
print(y_pred)
```

```
[0 0 0 ... 1 0 1]
```

In [19]:

```
from sklearn.model_selection import cross_val_score
accuracies=cross_val_score(estimator=classifier,X=X_train,y=Y_train,cv=10)
print('Accuracy: {:.2f} Standard Deviation: {:.2f}'.format(accuracies.mean()*100,a
```

```
Accuracy: 87.08 Standard Deviation: 0.60
```

In [20]:

```
print(np.concatenate((y_pred.reshape(len(y_pred),1),Y_test.reshape(len(Y_test),1)
```

```
[[0 0]
 [0 0]
 [0 0]
 ...
 [1 1]
 [0 0]
 [1 1]]
```

In [21]:

```
result=classifier.predict(sc.transform([[40,4,80000,9,9,0,4,0,4,1,0,1000,50,39]])
if result==[0]:
  print('Person makes Below 50K/year')
else:
  print('Person makes Above 50K/year')
```

```
Person makes Below 50K/year
```

In [ ]: