ML 512 Project Choice 2 - Explore Dataset(3-3)

In [1]:
```
import pandas as pd
import numpy as np
from sklearn import tree
from sklearn import metrics
from sklearn.model_selection import train_test_split
from sklearn.model_selection import KFold, cross_val_score
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
```

In [2]:
```
df = pd.read_csv('adult.csv', header = 0)
df.head()
```

Out[2]:

	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	h
0	90	?	77053	HS-grad	9	Widowed	?	Not-in-family	White	Female	0	4356	
1	82	Private	132870	HS-grad	9	Widowed	Exec-managerial	Not-in-family	White	Female	0	4356	
2	66	?	186061	Some-college	10	Widowed	?	Unmarried	Black	Female	0	4356	
3	54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	Unmarried	White	Female	0	3900	
4	41	Private	264663	Some-college	10	Separated	Prof-specialty	Own-child	White	Female	0	3900	

In [3]:
```
df.rename(columns={'native.country':'nativecountry'}, inplace=True)
df = df[(df.workclass != '?')]
df = df[(df.nativecountry != '?')]
df = df[(df.occupation != '?')]
df = df.drop('education',1)
df.occupation.unique()
```

Out[3]:
```
array(['Exec-managerial', 'Machine-op-inspct', 'Prof-specialty',
       'Other-service', 'Adm-clerical', 'Transport-moving', 'Sales',
       'Craft-repair', 'Farming-fishing', 'Tech-support',
       'Protective-serv', 'Handlers-cleaners', 'Armed-Forces',
       'Priv-house-serv'], dtype=object)
```

In [4]:
```
d = {'Private' : 1, 'Self-emp-not-inc' : 2, 'Self-emp-inc' : 3, 'Federal-gov' : 4, 'Local-gov' : 5,
     'State-gov' : 6, 'Without-pay' : 7, 'Never-worked' : 8}
df['workclass'] = df['workclass'].map(d)
d = {'Married-civ-spouse' : 1, 'Divorced' : 2, 'Never-married' : 3, 'Separated' : 4,
     'Widowed' : 5, 'Married-spouse-absent' : 6, 'Married-AF-spouse' : 7}
df['marital.status'] = df['marital.status'].map(d)
d = {'Tech-support' : 1, 'Craft-repair' : 2, 'Other-service' : 3, 'Sales' : 4, 'Exec-managerial' : 5,
     'Prof-specialty' : 6, 'Handlers-cleaners' : 7, 'Machine-op-inspct' : 8, 'Adm-clerical' : 9,
     'Farming-fishing' : 10, 'Transport-moving' : 11, 'Priv-house-serv' : 12, 'Protective-serv' : 13,
'Armed-Forces' : 14}
df['occupation'] = df['occupation'].map(d)
d = {'Wife' : 1, 'Own-child' : 2, 'Husband' : 3, 'Not-in-family' : 4, 'Other-relative' : 5, 'Unmarried'
 : 7}
df['relationship'] = df['relationship'].map(d)
d = {'White' : 1, 'Asian-Pac-Islander' : 2, 'Amer-Indian-Eskimo' : 3, 'Other' : 4, 'Black' : 5}
df['race'] =df['race'].map(d)
d = {'Female' : 1, 'Male' : 2}
df['sex'] = df['sex'].map(d)
d = {'United-States' : 1, 'Mexico' : 2, 'Greece' : 3, 'Vietnam' : 4, 'China' : 5, 'Taiwan' : 6,
       'Holand-Netherlands' : 7, 'Puerto-Rico' : 8, 'Poland' : 9, 'Iran' : 10, 'England' : 11,
       'Germany' : 12, 'Italy' : 13, 'Japan' : 14, 'Hong' : 15, 'Honduras' : 16, 'Cuba' : 17, 'Ireland'
 : 18,
       'Cambodia' : 19, 'Peru' : 20, 'Nicaragua' : 21, 'Dominican-Republic' : 22, 'Haiti' : 23,
       'Hungary' : 24, 'Columbia' : 25, 'Guatemala' : 26, 'El-Salvador' : 27, 'Jamaica' : 28,
       'Ecuador' : 29, 'France' : 30, 'Yugoslavia' : 31, 'Portugal' : 32, 'Laos' : 33, 'Thailand' : 34,
       'Outlying-US(Guam-USVI-etc)' : 35, 'Scotland' : 36,
       'India' : 35, 'Philippines' : 36, 'Trinadad&Tobago' : 37, 'Canada' : 38, 'South' : 39}
df['nativecountry'] = df['nativecountry'].map(d)
d = {'>50K' : 1, '<=50K' : 2}
df['income'] = df['income'].map(d)
```

In [5]:
```
features = list(df.columns[:13])
features
```

Out[5]:
```
['age',
 'workclass',
 'fnlwgt',
 'education.num',
 'marital.status',
 'occupation',
 'relationship',
 'race',
 'sex',
 'capital.gain',
 'capital.loss',
 'hours.per.week',
 'nativecountry']
```

In [6]:
```
y = df["income"]
x = df[features]
Tree = tree.DecisionTreeClassifier()
Tree = Tree.fit(x,y)
```

In [7]:
```
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=.4, random_state=0)
```

In [8]:
```
kf = KFold(n_splits=10, shuffle=False)
print('KFold CrossValScore Using Decision Tree %s' % cross_val_score(Tree,x, y, cv=5).mean())
```

```
KFold CrossValScore Using Decision Tree 0.7634782927801101
```

In [9]:
```
rf = Tree.fit(X_train, y_train)
y_pred = rf.predict(X_test)
metrics.accuracy_score(y_test, y_pred)
```

Out[9]:
```
0.8101118939079983
```

In [10]:
```
print(accuracy_score(y_test, y_pred))
print(f1_score(y_test, y_pred))
```

```
0.8101118939079983
0.6313757039420758
```