Prediction task is to determine whether a person makes over 50K a year by XGBoost.

In [1]:

```python
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

In [2]:

```python
# Importing the dataset
train = pd.read_csv('http://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.d
test = pd.read_csv('http://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.te
```

In [3]:

```python
train.head(5)
```

Out[3]:

	0	1	2	3	4	5	6	7	8	9	10	11
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0

In [4]:

```python
column_names = ['age', 'workclass', 'fnlwgt', 'education', 'educational_num','marital_statu
train.columns = column_names
test.columns = column_names
```

In [5]:

```
train.head(5)
```

Out[5]:

	age	workclass	fnlwgt	education	educational_num	marital_status	occupation	relationship
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife

In [6]:

```
test.head(5)
```

Out[6]:

	age	workclass	fnlwgt	education	educational_num	marital_status	occupation	relationship
0	25	Private	226802	11th	7	Never-married	Machine-op-inspct	Own-child
1	38	Private	89814	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband
2	28	Local-gov	336951	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband
3	44	Private	160323	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband
4	18	?	103497	Some-college	10	Never-married	?	Own-child

In [7]:

```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32561 entries, 0 to 32560
Data columns (total 15 columns):
age                32561 non-null int64
workclass          32561 non-null object
fnlwgt             32561 non-null int64
education          32561 non-null object
educational_num    32561 non-null int64
marital_status     32561 non-null object
occupation         32561 non-null object
relationship       32561 non-null object
race               32561 non-null object
gender             32561 non-null object
capital_gain       32561 non-null int64
capital_loss       32561 non-null int64
hours_per_week     32561 non-null int64
native_country     32561 non-null object
income             32561 non-null object
dtypes: int64(6), object(9)
memory usage: 3.7+ MB
```

In [8]:

```
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16281 entries, 0 to 16280
Data columns (total 15 columns):
age                16281 non-null int64
workclass          16281 non-null object
fnlwgt             16281 non-null int64
education          16281 non-null object
educational_num    16281 non-null int64
marital_status     16281 non-null object
occupation         16281 non-null object
relationship       16281 non-null object
race               16281 non-null object
gender             16281 non-null object
capital_gain       16281 non-null int64
capital_loss       16281 non-null int64
hours_per_week     16281 non-null int64
native_country     16281 non-null object
income             16281 non-null object
dtypes: int64(6), object(9)
memory usage: 1.9+ MB
```

In [9]:

```
print(train.describe())
print('*'*100)
print(test.describe())
```

```
                 age          fnlwgt  educational_num  capital_gain  \
count  32561.000000    3.256100e+04     32561.000000  32561.000000
mean      38.581647    1.897784e+05        10.080679   1077.648844
std       13.640433    1.055500e+05         2.572720   7385.292085
min       17.000000    1.228500e+04         1.000000      0.000000
25%       28.000000    1.178270e+05         9.000000      0.000000
50%       37.000000    1.783560e+05        10.000000      0.000000
75%       48.000000    2.370510e+05        12.000000      0.000000
max       90.000000    1.484705e+06        16.000000  99999.000000

       capital_loss  hours_per_week
count  32561.000000    32561.000000
mean      87.303830       40.437456
std      402.960219       12.347429
min        0.000000        1.000000
25%        0.000000       40.000000
50%        0.000000       40.000000
75%        0.000000       45.000000
max     4356.000000       99.000000
********************************************************************************
**********************
                 age          fnlwgt  educational_num  capital_gain  \
count  16281.000000    1.628100e+04     16281.000000  16281.000000
mean      38.767459    1.894357e+05        10.072907   1081.905104
std       13.849187    1.057149e+05         2.567545   7583.935968
min       17.000000    1.349200e+04         1.000000      0.000000
25%       28.000000    1.167360e+05         9.000000      0.000000
50%       37.000000    1.778310e+05        10.000000      0.000000
75%       48.000000    2.383840e+05        12.000000      0.000000
max       90.000000    1.490400e+06        16.000000  99999.000000

       capital_loss  hours_per_week
count  16281.000000    16281.000000
mean      87.899269       40.392236
std      403.105286       12.479332
min        0.000000        1.000000
25%        0.000000       40.000000
50%        0.000000       40.000000
75%        0.000000       45.000000
max     3770.000000       99.000000
```

From the primary data analysis, we can note that there are both numeric and object variables and there are ? values

In [10]:

```
train = train.replace(' ?',np.nan)
test = test.replace(' ?',np.nan)
```

In [11]:

```python
print(train.isna().sum())
print('*'*100)
print(test.isna().sum())
```

```
age                     0
workclass            1836
fnlwgt                  0
education               0
educational_num         0
marital_status          0
occupation           1843
relationship            0
race                    0
gender                  0
capital_gain            0
capital_loss            0
hours_per_week          0
native_country        583
income                  0
dtype: int64
****************************************************************************
************************
age                     0
workclass             963
fnlwgt                  0
education               0
educational_num         0
marital_status          0
occupation            966
relationship            0
race                    0
gender                  0
capital_gain            0
capital_loss            0
hours_per_week          0
native_country        274
income                  0
dtype: int64
```

In [12]:

```python
# As there are large number of null values we will drop the null values
train = train.dropna()
test = test.dropna()
```

In [13]:

```python
dataset = pd.concat([train,test],axis=0)
```

In [14]:

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 45222 entries, 0 to 16280
Data columns (total 15 columns):
age                45222 non-null int64
workclass          45222 non-null object
fnlwgt             45222 non-null int64
education          45222 non-null object
educational_num    45222 non-null int64
marital_status     45222 non-null object
occupation         45222 non-null object
relationship       45222 non-null object
race               45222 non-null object
gender             45222 non-null object
capital_gain       45222 non-null int64
capital_loss       45222 non-null int64
hours_per_week     45222 non-null int64
native_country     45222 non-null object
income             45222 non-null object
dtypes: int64(6), object(9)
memory usage: 5.5+ MB
```

In [15]:

```
dataset.income.unique()
```

Out[15]:

```
array([' <=50K', ' >50K', ' <=50K.', ' >50K.'], dtype=object)
```

In [16]:

```
dataset['income'] = dataset.income.replace({' <=50K':' <=50K', ' >50K':' >50K', ' <=50K.':'
```

In [17]:

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 45222 entries, 0 to 16280
Data columns (total 15 columns):
age                45222 non-null int64
workclass          45222 non-null object
fnlwgt             45222 non-null int64
education          45222 non-null object
educational_num    45222 non-null int64
marital_status     45222 non-null object
occupation         45222 non-null object
relationship       45222 non-null object
race               45222 non-null object
gender             45222 non-null object
capital_gain       45222 non-null int64
capital_loss       45222 non-null int64
hours_per_week     45222 non-null int64
native_country     45222 non-null object
income             45222 non-null object
dtypes: int64(6), object(9)
memory usage: 5.5+ MB
```

EDA

In [18]:

```python
import seaborn as sns
fig, ((a,b),(c,d)) = plt.subplots(2,2,figsize=(20,15))
plt.xticks(rotation=45)
sns.countplot(dataset['workclass'],hue=dataset['income'],ax=a)
sns.countplot(dataset['relationship'],hue=dataset['income'],ax=b)
sns.countplot(dataset['gender'],hue=dataset['income'],ax=c)
sns.countplot(dataset['race'],hue=dataset['income'],ax=d)
```
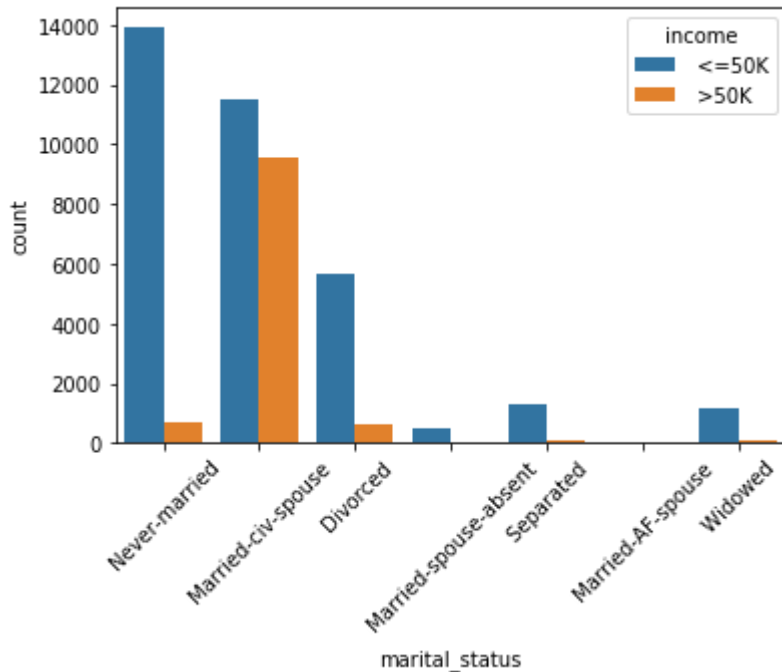
Out[18]:

<matplotlib.axes._subplots.AxesSubplot at 0x16e7697ed30>
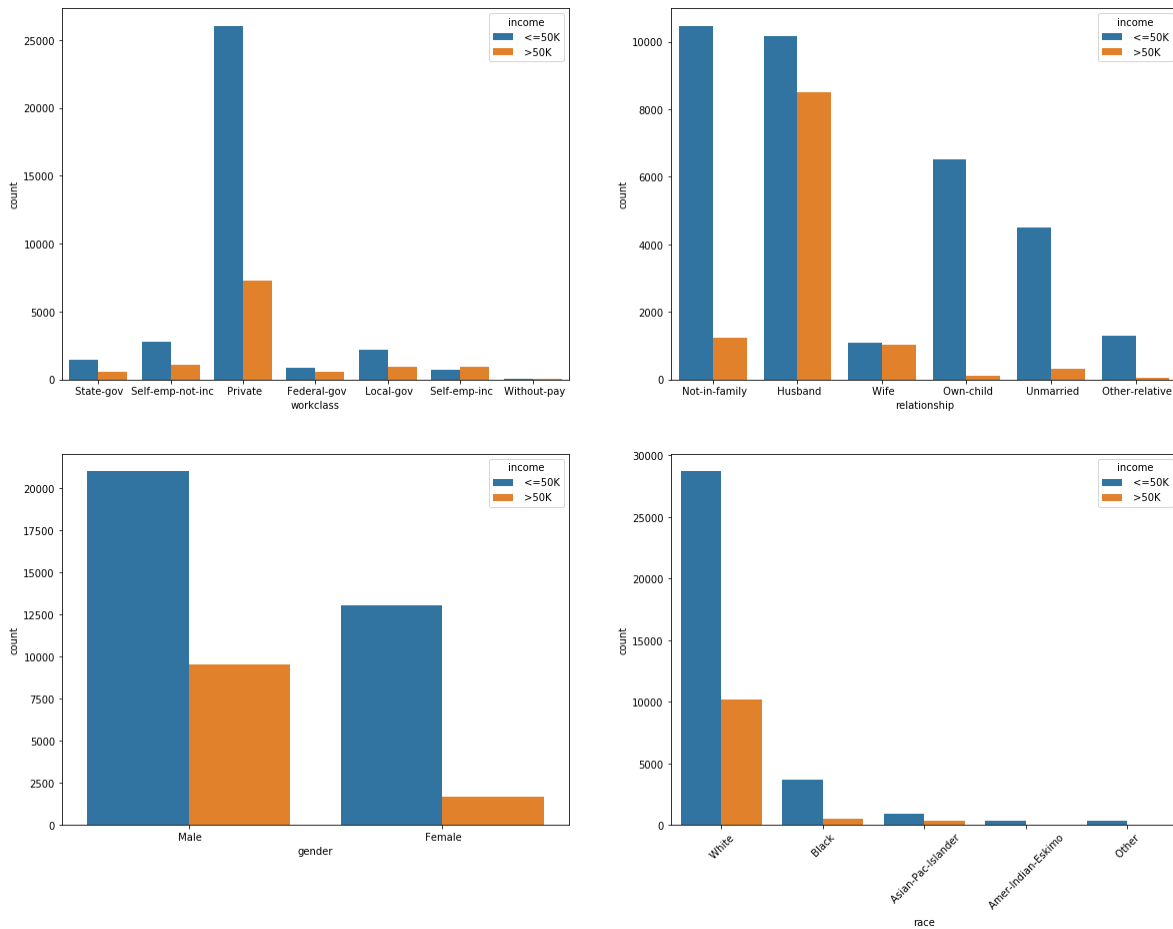
In [19]:

```
plt.xticks(rotation=45)
sns.countplot(dataset['marital_status'],hue=dataset['income'])
```
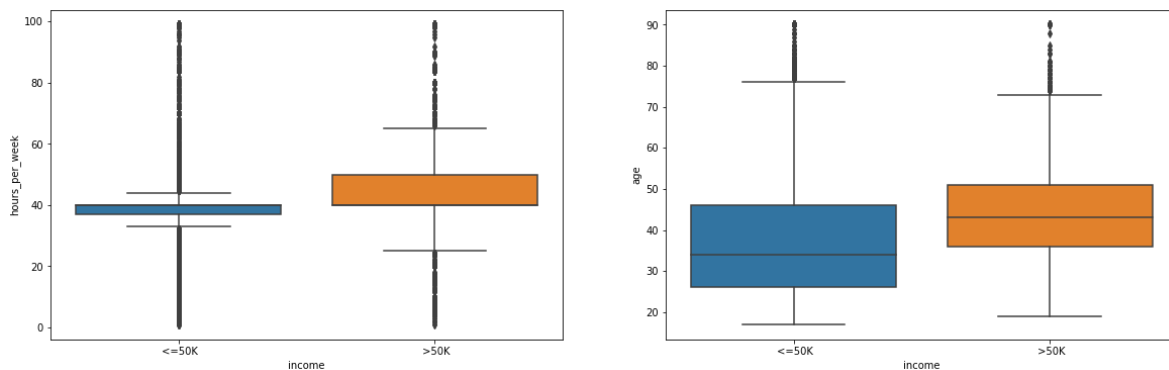
Out[19]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x16e7715a518>
```


In [20]:

```
fig, (a,b)= plt.subplots(1,2,figsize=(20,6))
sns.boxplot(y='hours_per_week',x='income',data=dataset,ax=a)
sns.boxplot(y='age',x='income',data=dataset,ax=b)
```
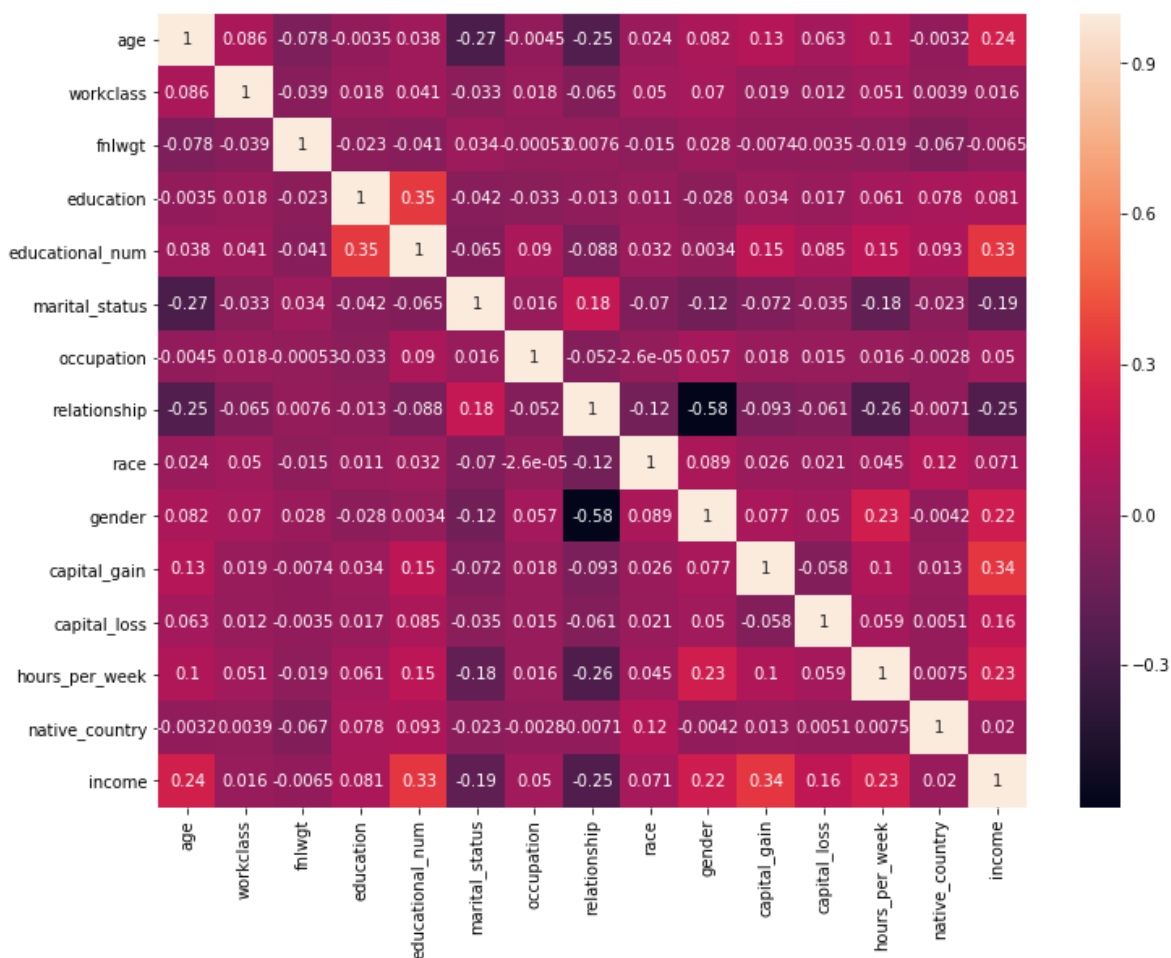
Out[20]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x16e774f8588>
```


Data preprocessing

In [21]:

```python
#Feature Engineering
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
for i in dataset.columns:
    dataset[i]=le.fit_transform(dataset[i])
```

In [22]:

```python
hmap = dataset.corr()
plt.subplots(figsize=(12, 9))
sns.heatmap(hmap,annot=True);
```


In [23]:

```python
#split the dataset
X = dataset.iloc[:,:-1].values
y = dataset.iloc[:,14].values
X_train = X[:train.shape[0]] # Up to the last initial training set row
y_train = y[:train.shape[0]] # Up to the last initial training set row
X_test  = X[train.shape[0]:] # Past the last initial training set row
y_test  = y[train.shape[0]:] # Past the last initial training set row
```

In [24]:

```
print(X_train.shape,'**',y_train.shape,'**', X_test.shape,'**',y_test.shape)
```

(30162, 14) ** (30162,) ** (15060, 14) ** (15060,)

In [25]:

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

C:\Users\mallikarjuna.m\AppData\Local\Continuum\anaconda3\lib\site-packages
\sklearn\utils\validation.py:595: DataConversionWarning: Data with input dty
pe int64 was converted to float64 by StandardScaler.
  warnings.warn(msg, DataConversionWarning)
C:\Users\mallikarjuna.m\AppData\Local\Continuum\anaconda3\lib\site-packages
\sklearn\utils\validation.py:595: DataConversionWarning: Data with input dty
pe int64 was converted to float64 by StandardScaler.
  warnings.warn(msg, DataConversionWarning)
C:\Users\mallikarjuna.m\AppData\Local\Continuum\anaconda3\lib\site-packages
\sklearn\utils\validation.py:595: DataConversionWarning: Data with input dty
pe int64 was converted to float64 by StandardScaler.
  warnings.warn(msg, DataConversionWarning)

In [26]:

```
# Fitting XGBoost to the Training set
from xgboost import XGBClassifier
from sklearn.metrics import confusion_matrix
xgb = XGBClassifier()
xgb.fit(X_train, y_train)
y_pred = xgb.predict(X_test)
cm = confusion_matrix(y_test, y_pred)
Correct = cm[0,0] + cm[1,1]
Total =  cm[0,0] + cm[0,1]+ cm[1,0] + cm[1,1]
xgb_accuracy = Correct / Total
print('XGBoost Accuracy : ', xgb_accuracy*100,'%' )
print(cm)
```

XGBoost Accuracy : 86.15537848605578 %
[[10785   575]
 [1510 2190]]

In [ ]: