Experiment No. 4
Apply Random Forest Algorithm on Adult Census Income Dataset and analyze the performance of the model
Date of Performance: 07/09/2023
Date of Submission: 14/09/2023

Aim: Apply Random Forest Algorithm on Adult Census Income Dataset and analyze the performance of the model.

Objective: Able to perform various feature engineering tasks, apply Random Forest Algorithm on the given dataset and maximize the accuracy, Precision, Recall, F1 score.

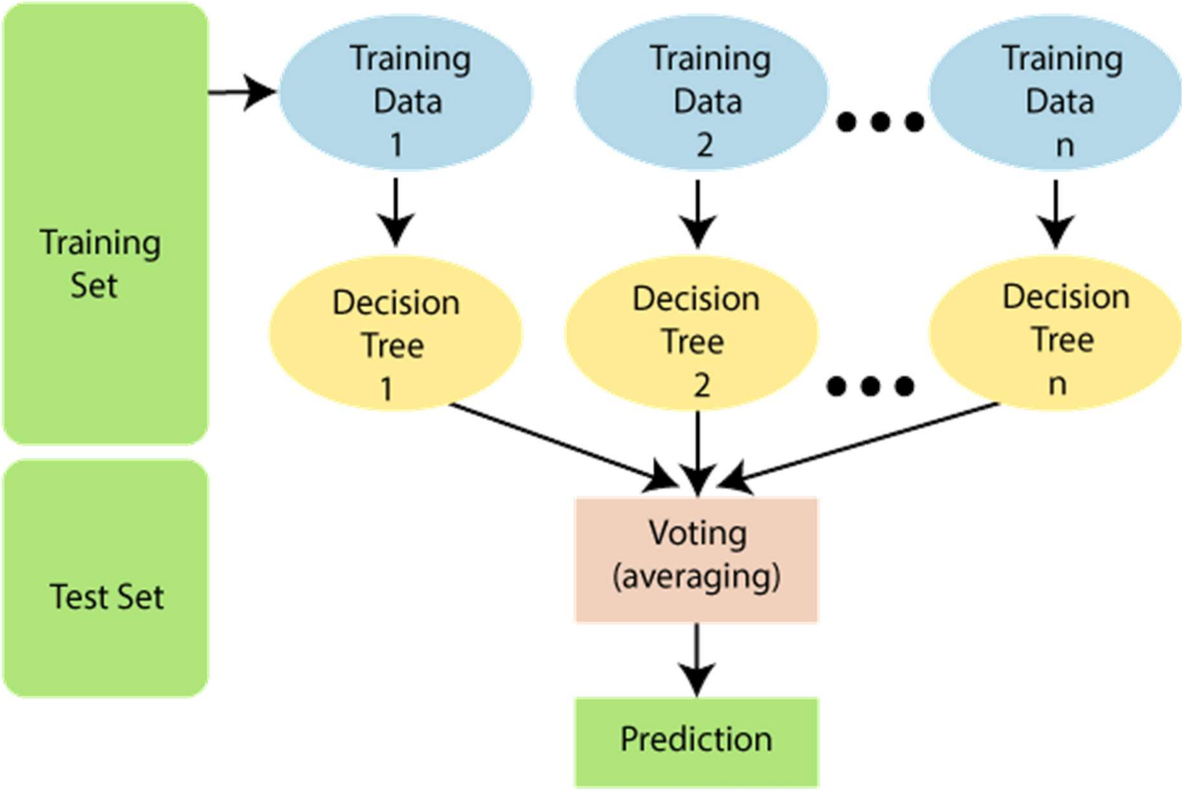
Theory:

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

The below diagram explains the working of the Random Forest algorithm:



Dataset:

Predict whether income exceeds $50K/yr based on census data. Also known as "Adult" dataset.

Attribute Information:

Listing of attributes:

>50K, <=50K. age:

continuous.

workclass: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked. fnlwgt: continuous.

education: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool. education-num: continuous.

marital-status: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Marriedspouse-absent, Married-AF-spouse.

occupation: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces.

relationship: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.

race: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black. sex: Female,

Male. capital-gain: continuous. capital-loss: continuous.

hours-per-week: continuous.

native-country: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, OutlyingUS(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad &Tobago, Peru, Hong, Holand-Netherlands.

In [4]:
```python
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import train_test_split,cross_val_score,KFold,Gr
from sklearn.metrics import confusion_matrix,classification_report,accuracy_sc
import scikitplot as skplt
```

In [5]:
```python
dataset=pd.read_csv("adult.csv")
```

In [6]:
```python
print(dataset.isnull().sum())
print(dataset.dtypes)
```

```
age               0
workclass         0
fnlwgt            0
education         0
education.num     0
marital.status    0
occupation        0
relationship      0
race              0
sex               0
capital.gain      0
capital.loss      0
hours.per.week    0
native.country    0
income            0
dtype: int64
age                int64
workclass         object
fnlwgt             int64
education         object
education.num      int64
marital.status    object
occupation        object
relationship      object
race              object
sex               object
capital.gain       int64
capital.loss       int64
hours.per.week     int64
native.country    object
income            object
dtype: object
```

]:

In [7

```python
dataset.head()
```

Out[7]:

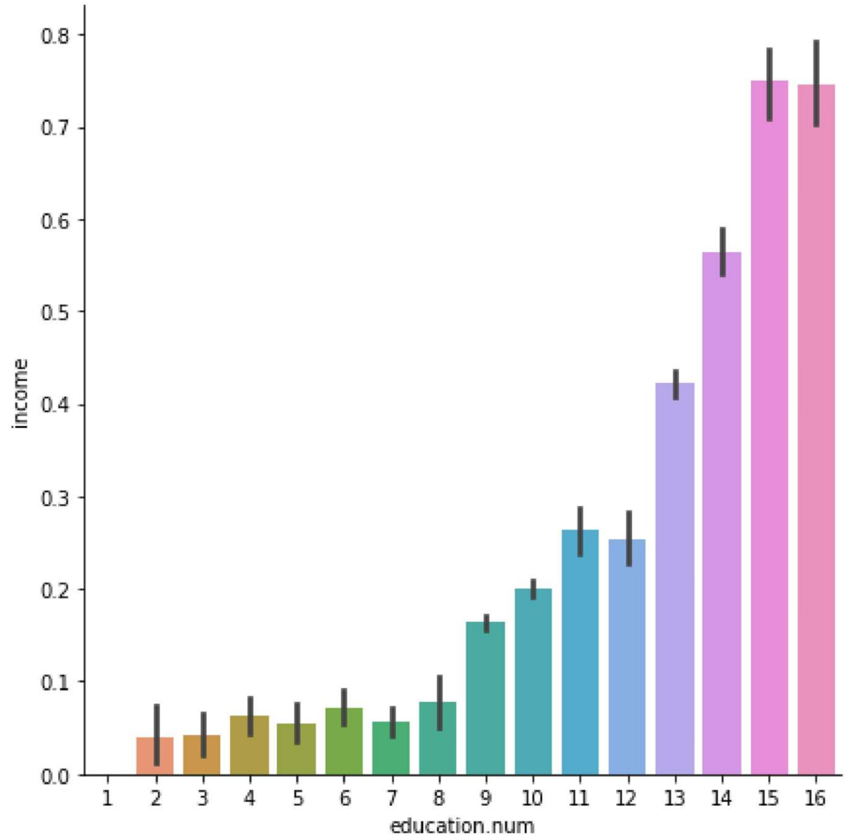
	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	
0	90	?	77053	HS-grad	9	Widowed	?	Not-in-family	V
1	82	Private	132870	HS-grad	9	Widowed	Exec-managerial	Not-in-family	V
2	66	?	186061	Some-college	10	Widowed	?	Unmarried	E
3	54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct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

In [8]:

```python
#removing '?' containing rows
dataset = dataset[(dataset != '?').all(axis=1)]
#label the income objects as 0 and 1
dataset['income']=dataset['income'].map({'<=50K': 0, '>50K': 1})
```

In [9]:

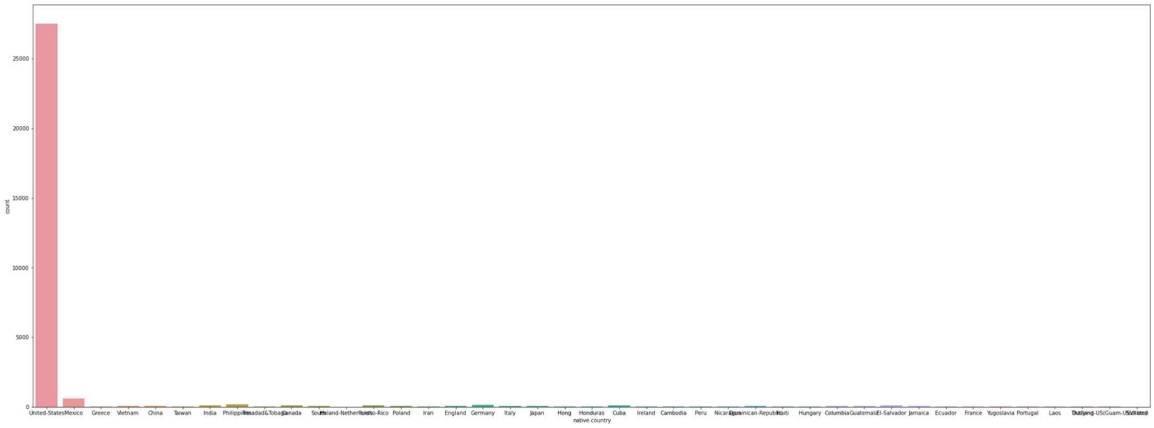
```python
sns.catplot(x='education.num',y='income',data=dataset,kind='bar',height=6)
plt.show()
```



In [1]:

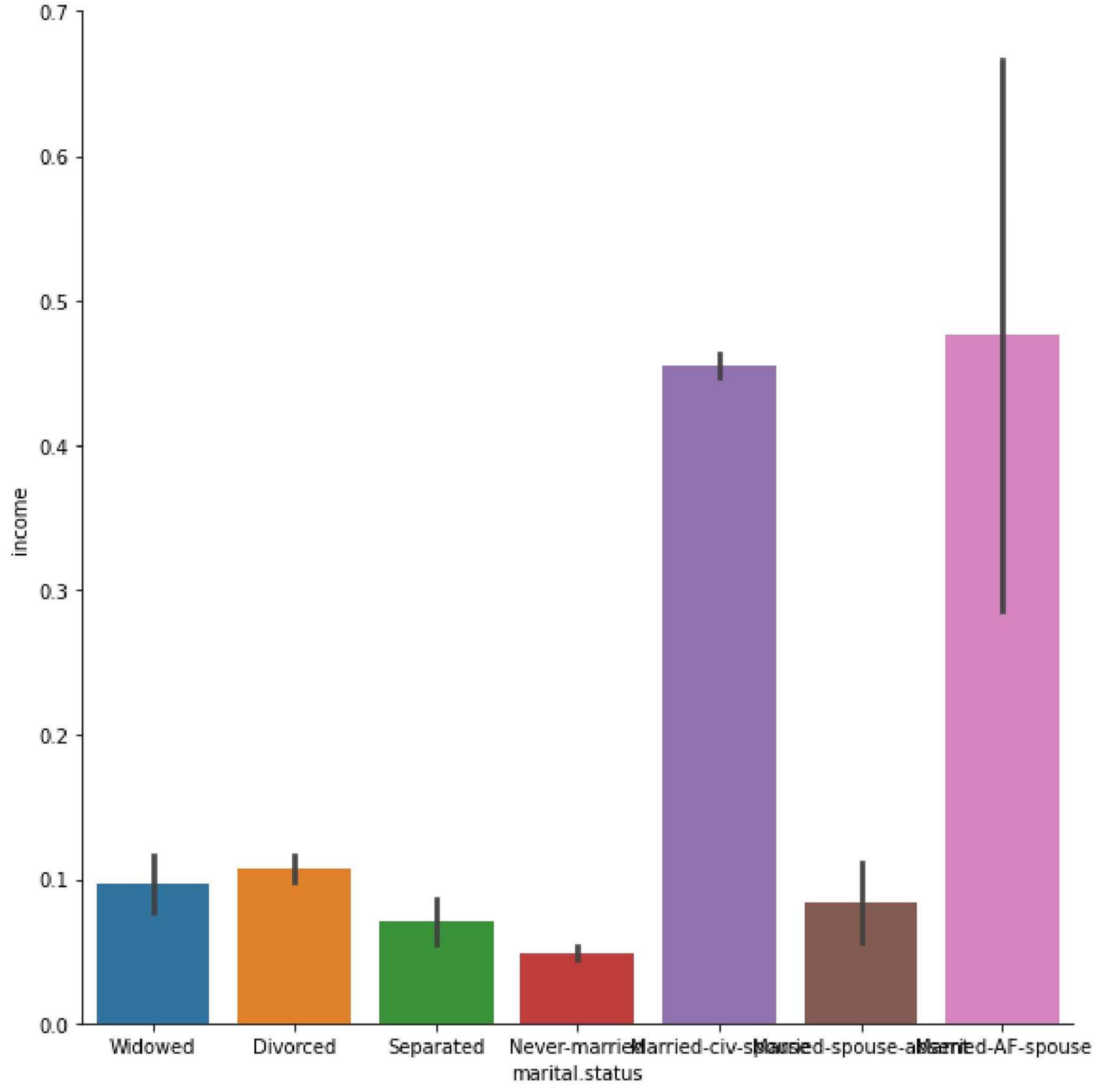
0

```
#explore which country do most people belong
plt.figure(figsize=(38,14))
sns.countplot(x='native.country',data=dataset)
plt.show()
```



```
1   #marital.status vs income
    sns.factorplot(x='marital.status',y='income',data=dataset,kind='bar',height=8)
    plt.show()
```

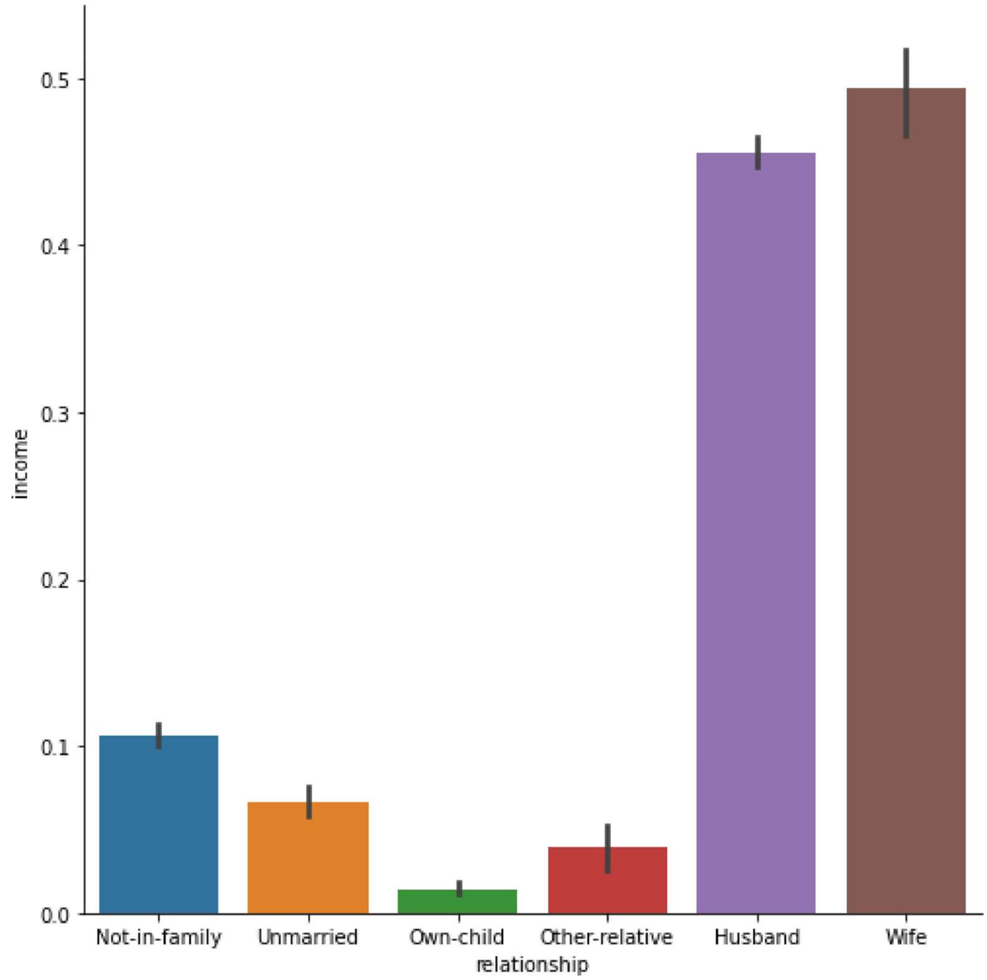
```
C:\Users\Amruta\anaconda3\lib\site-packages\seaborn\categorical.py:3714: Use
rWarning: The `factorplot` function has been renamed to `catplot`. The origi
nal name will be removed in a future release. Please update your code. Note
that the default `kind` in `factorplot` (`'point'`) has changed `'strip'` in
`catplot`.
  warnings.warn(msg)
```



2

```
#relationship vs income
sns.factorplot(x='relationship',y='income',data=dataset,kind='bar',size=7)
plt.show()
```

```
C:\Users\Amruta\anaconda3\lib\site-packages\seaborn\categorical.py:3714: Use
rWarning: The `factorplot` function has been renamed to `catplot`. The origi
nal name will be removed in a future release. Please update your code. Note
that the default `kind` in `factorplot` (`'point'`) has changed `'strip'` in
`catplot`.
  warnings.warn(msg)
C:\Users\Amruta\anaconda3\lib\site-packages\seaborn\categorical.py:3720: Use
rWarning: The `size` parameter has been renamed to `height`; please update y
our code.
  warnings.warn(msg, UserWarning)
```



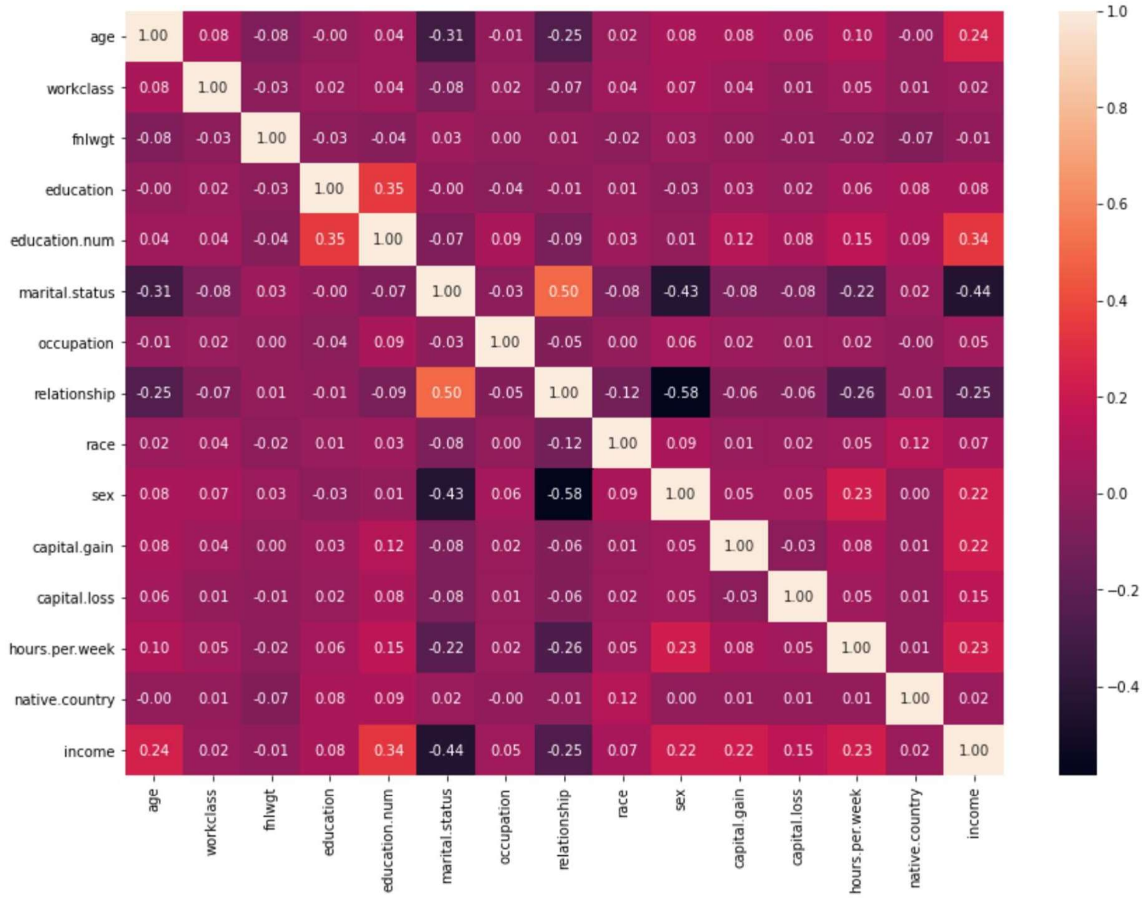
In [13]:

```
#we can reformat marital.status values to single and married
dataset['marital.status']=dataset['marital.status'].map({'Married-civ-spouse'
'Widowed':'Single', 'Married-spouse-absent':'Married', 'Married-AF-spouse':'Ma
```

In [14]:
```
for column in dataset:
    enc=LabelEncoder()
    if dataset.dtypes[column]==np.object:
        dataset[column]=enc.fit_transform(dataset[column])
```

```
ecated alias for the builtin `object`. To silence this warning, use `object`
by itself. Doing this will not modify any behavior and is safe.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/d
evdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/rel
ease/1.20.0-notes.html#deprecations)
  if dataset.dtypes[column]==np.object:
```

In [15]:
```
plt.figure(figsize=(14,10))
sns.heatmap(dataset.corr(),annot=True,fmt='.2f')
plt.show()
```



In [16]:
```
dataset=dataset.drop(['relationship','education'],axis=1)
```

In [17]:
```
dataset=dataset.drop(['occupation','fnlwgt','native.country'],axis=1)
```

In [18]:
```python
print(dataset.head())
```

```
   age  workclass  education.num  marital.status  race  sex  capital.gain  \
1   82          2              9               1     4    0             0   
3   54          2              4               1     4    0             0   
4   41          2             10               1     4    0             0   
5   34          2              9               1     4    0             0   
6   38          2              6               1     4    1             0   

   capital.loss  hours.per.week  income  
1          4356              18       0  
3          3900              40       0  
4          3900              40       0  
5          3770              45       0  
6          3770              40       0  
```

In [19]:
```python
X=dataset.iloc[:,0:-1]
y=dataset.iloc[:,-1]
print(X.head())
print(y.head())
x_train,x_test,y_train,y_test=train_test_split(X,y,test_size=0.33,shuffle=Fals
```

```
   age  workclass  education.num  marital.status  race  sex  capital.gain  \
1   82          2              9               1     4    0             0   
3   54          2              4               1     4    0             0   
4   41          2             10               1     4    0             0   
5   34          2              9               1     4    0             0   
6   38          2              6               1     4    1             0   

   capital.loss  hours.per.week  
1          4356              18  
3          3900              40  
4          3900              40  
5          3770              45  
6          3770              40  
1    0
3    0
4    0
5    0
6    0
Name: income, dtype: int64
```

In [20]:
```python
clf=GaussianNB()
cv_res=cross_val_score(clf,x_train,y_train,cv=10)
print(cv_res.mean()*100)
```

```
76.68213951528749
```

In [21]:
```python
clf=DecisionTreeClassifier()
cv_res=cross_val_score(clf,x_train,y_train,cv=10)
print(cv_res.mean()*100)
```

```
74.25305088648399
```

In [22]:
```
clf=RandomForestClassifier(n_estimators=100)
cv_res=cross_val_score(clf,x_train,y_train,cv=10)
print(cv_res.mean()*100)
```

```
76.70739904272466
```

In [23]:
```
clf=RandomForestClassifier(n_estimators=50,max_features=5,min_samples_leaf=50)
clf.fit(x_train,y_train)
```

Out[23]:
```
RandomForestClassifier

RandomForestClassifier(max_features=5, min_samples_leaf=50, n_estimators=5
0)
```

In [24]:
```
pred=clf.predict(x_test)
pred
```

Out[24]:
```
array([1, 1, 1, ..., 0, 0, 0], dtype=int64)
```

In [25]:
```
print("Accuracy: %f " % (100*accuracy_score(y_test, pred)))
```

```
Accuracy: 84.729757
```

In [37]:
```
print(confusion_matrix(y_test, pred))
```

```
[[7521  421]
 [1065  947]]
```

In [38]:
```
print(classification_report(y_test, pred))
```

```
              precision    recall  f1-score   support

           0       0.88      0.95      0.91      7942
           1       0.69      0.47      0.56      2012

    accuracy                           0.85      9954
   macro avg       0.78      0.71      0.74      9954
weighted avg       0.84      0.85      0.84      9954
```

Conclusion:

1. From the correlation heat map, it was observed that there exists a high positive correlation between education and education.num as well as between martital.status and relationship. Hence relationship and education attributes were drop to improve accuracy of model
2. The Accuracy score obtained by our decision tree model on the testing data is 0.84 which means our model is 84% accurate on the testing data.
3. Confusion matrix is used to assess the performance of a classification model, in our case the no. of TP is 947, no. of TN is 7521, no. of FP is 421 and no. of FN are 1065 which means our model is better in predicting negative cases than the positive cases.
4. Precision measures the accuracy of the positive predictions and the precision score obtained by our model is 0.88
5. Recall measures the ability of the model to correctly identify all relevant instances and the Recall score obtained by our model is 0.95
6. F1-score is the harmonic mean of precision and recall and provides a balance between the 2 metrics and the F1-score obtained by our model is 0.91
7. In the decision tree algorithm, the accuracy, precision, recall and F1-score obtained respectively is 83%, 85%, 96%, 90% and the accuracy, precision, recall and F1-score obtained by random forest algorithm respectively is 84%, 88%, 95%, 91%. Thus we can conclude that random forest algorithm is slightly better than the decision tree algorithm.