Experiment No. 4
Apply Random Forest Algorithm on Adult Census Income Dataset and analyze the performance of the model
Date of Performance: 14/8/23
Date of Submission: 22/8/23

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.

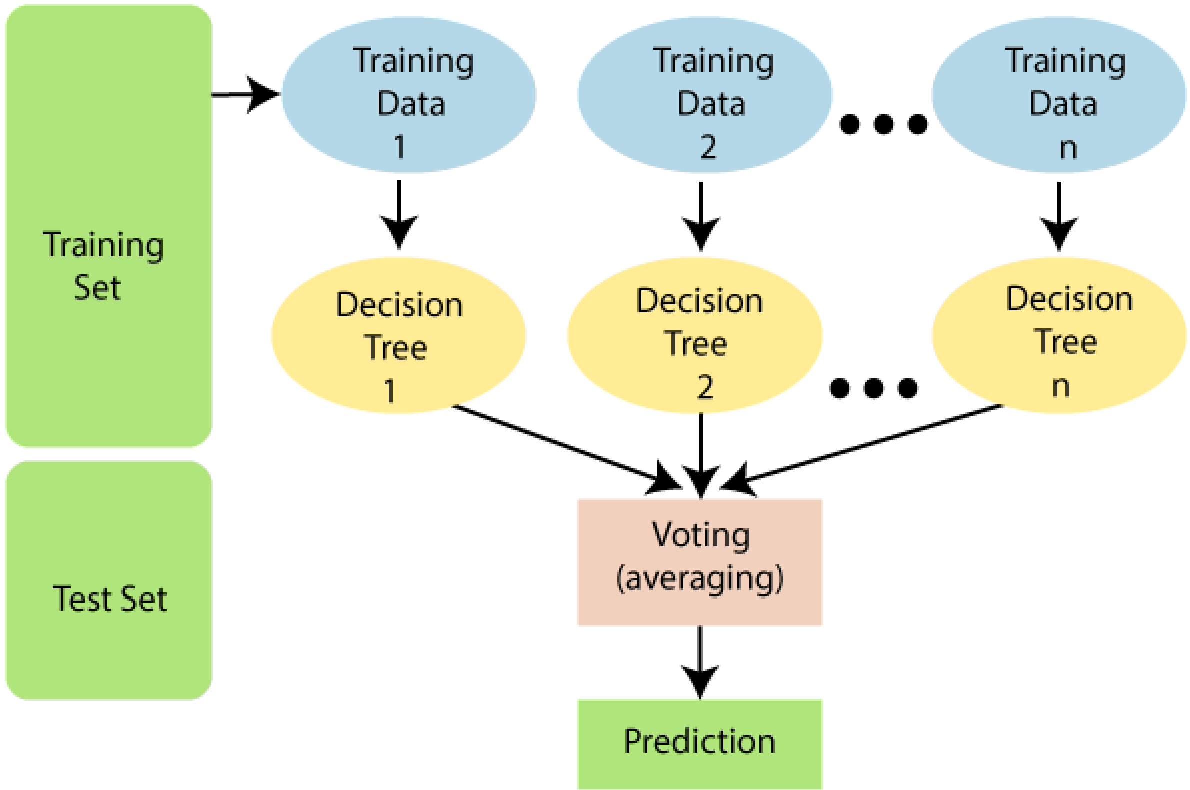
Theory:

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, Married-spouse-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, Outlying-US(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.

Code:

Conclusion:

1. State the observations about the data set from the correlation heat map.
   Most features have low correlations, indicating they offer distinct predictive information, reducing redundancy in the dataset.
2. Comment on the accuracy, confusion matrix, precision, recall and F1 score obtained.
   Accuracy is 84.58%.

3. Compare the results obtained by applying random forest and decision tree algorithm on the Adult Census Income Dataset

   In terms of precision, recall, and F1-Score, Random Forest exhibited a more balanced trade-off between precision and recall, whereas Decision Tree tended to prioritize one metric over the other. Additionally, Random Forest's ensemble approach facilitated better generalization, mitigating overfitting risks compared to Decision Trees, which are more sensitive to data variations. However, Decision Trees trained faster individually, while Random Forest required more time due to its multiple tree ensemble.

ADULT CENSUS INCOME PREDICTION

```
!pip install scikit-plot
!pip install -U seaborn
```

```
Collecting scikit-plot
  Downloading scikit_plot-0.3.7-py3-none-any.whl (33 kB)
Requirement already satisfied: matplotlib>=1.4.0 in /usr/local/lib/python3.10/dist-packages (from scikit-plot) (3.7.1)
Requirement already satisfied: scikit-learn>=0.18 in /usr/local/lib/python3.10/dist-packages (from scikit-plot) (1.2.2)
Requirement already satisfied: scipy>=0.9 in /usr/local/lib/python3.10/dist-packages (from scikit-plot) (1.11.3)
Requirement already satisfied: joblib>=0.10 in /usr/local/lib/python3.10/dist-packages (from scikit-plot) (1.3.2)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=1.4.0->scikit-plot) (1.1.1
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=1.4.0->scikit-plot) (0.12.0)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=1.4.0->scikit-plot) (4.43
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=1.4.0->scikit-plot) (1.4.
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=1.4.0->scikit-plot) (1.23.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=1.4.0->scikit-plot) (23.2)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=1.4.0->scikit-plot) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=1.4.0->scikit-plot) (3.1.1
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=1.4.0->scikit-plot) (2
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn>=0.18->scikit-plot) (
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib>=1.4.0->sciki
Installing collected packages: scikit-plot
Successfully installed scikit-plot-0.3.7
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-packages (0.12.2)
Collecting seaborn
  Downloading seaborn-0.13.0-py3-none-any.whl (294 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 294.6/294.6 kB 3.5 MB/s eta 0:00:00
Requirement already satisfied: numpy!=1.24.0,>=1.20 in /usr/local/lib/python3.10/dist-packages (from seaborn) (1.23.5)
Requirement already satisfied: pandas>=1.2 in /usr/local/lib/python3.10/dist-packages (from seaborn) (1.5.3)
Requirement already satisfied: matplotlib!=3.6.1,>=3.3 in /usr/local/lib/python3.10/dist-packages (from seaborn) (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.3->seaborn) (1.1
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.3->seaborn) (0.12.0)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.3->seaborn) (4.
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.3->seaborn) (1.
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.3->seaborn) (23.2
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.3->seaborn) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.3->seaborn) (3.1
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.3->seaborn)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2->seaborn) (2023.3.post1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib!=3.6.1,>=3.3-
Installing collected packages: seaborn
  Attempting uninstall: seaborn
    Found existing installation: seaborn 0.12.2
    Uninstalling seaborn-0.12.2:
      Successfully uninstalled seaborn-0.12.2
Successfully installed seaborn-0.13.0
```

```
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,GridSearchCV
from sklearn.metrics import confusion_matrix,classification_report,accuracy_score
import scikitplot as skplt
```

```
dataset=pd.read_csv("adult.csv")
```

```
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
```

```
dataset.head()
```

	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss
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

```
#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})
```

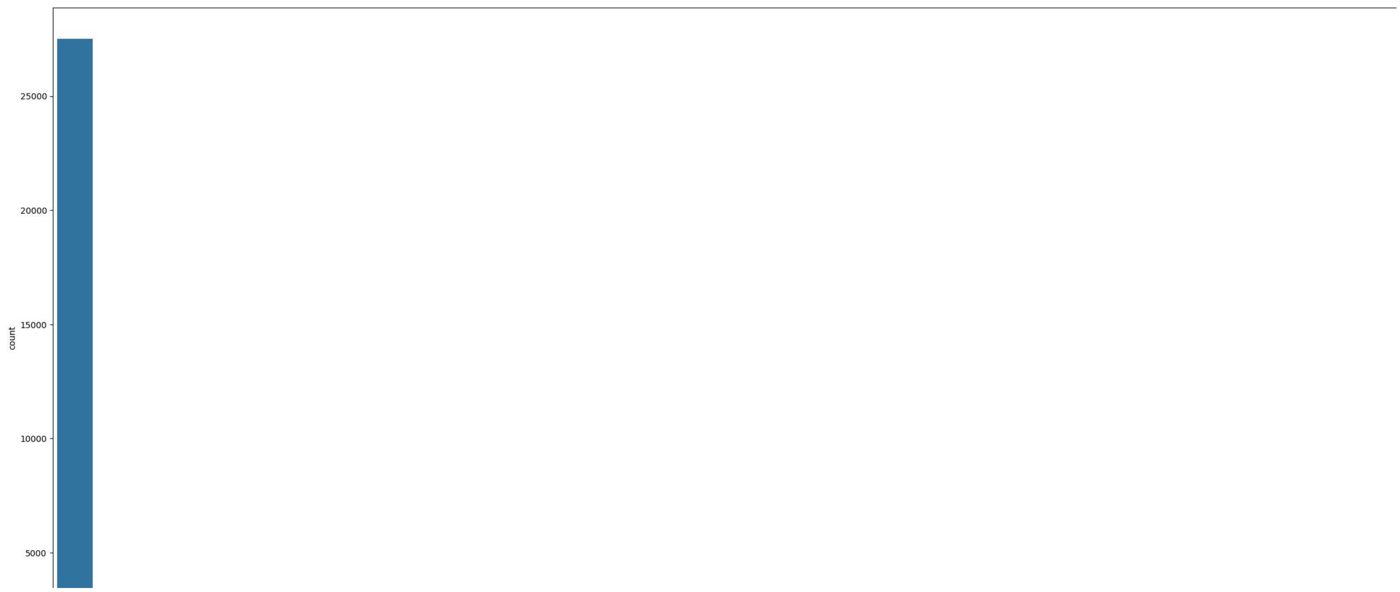
```
<ipython-input-7-39ed73805135>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
  dataset['income']=dataset['income'].map({'<=50K': 0, '>50K': 1})
```

Analyze data

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

Higher the value ,higher the probability of income greater than 50k(Obviously!)

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



Here most people are from the USA,so we can drop this column as it creates unnecessary bias.

Feature Engineering

```
#we can reformat marital.status values to single and married
dataset['marital.status']=dataset['marital.status'].map({'Married-civ-spouse':'Married', 'Divorced':'Single', 'Never-married':'Single', 'Sepa
'Widowed':'Single', 'Married-spouse-absent':'Married', 'Married-AF-spouse':'Married'})
```

Label encoding

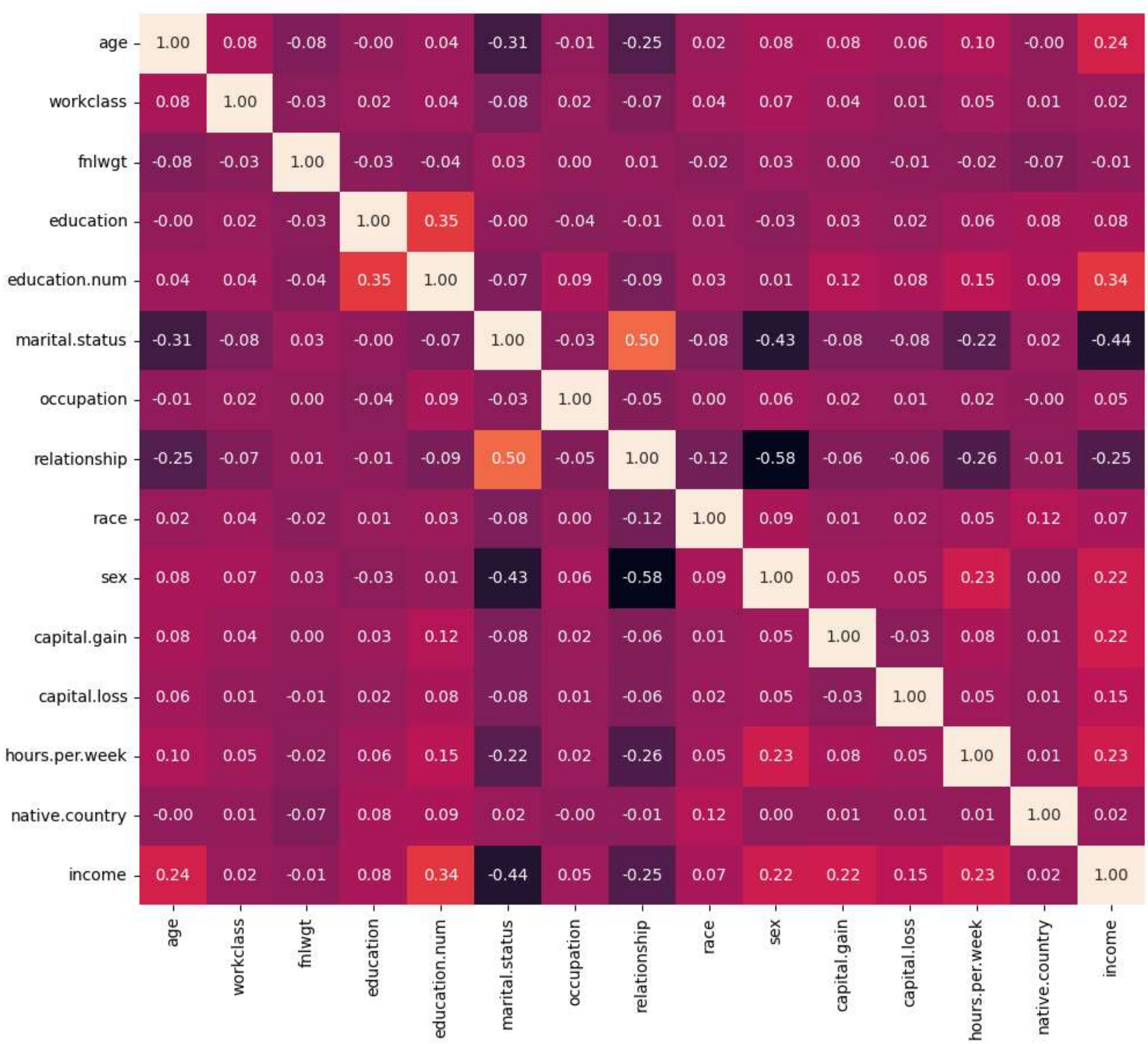
```
for column in dataset:
    enc=LabelEncoder()
    if dataset.dtypes[column]==np.object:
        dataset[column]=enc.fit_transform(dataset[column])
```

```
<ipython-input-11-5d7d7fe4d7c0>:3: DeprecationWarning: `np.object` is a deprecated alias for the builtin `object`. To silence this warn
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  if dataset.dtypes[column]==np.object:
<ipython-input-11-5d7d7fe4d7c0>:3: DeprecationWarning: `np.object` is a deprecated alias for the builtin `object`. To silence this warn
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  if dataset.dtypes[column]==np.object:
<ipython-input-11-5d7d7fe4d7c0>:3: DeprecationWarning: `np.object` is a deprecated alias for the builtin `object`. To silence this warn
```

```
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  if dataset.dtypes[column]==np.object:
<ipython-input-11-5d7d7fe4d7c0>:3: DeprecationWarning: `np.object` is a deprecated alias for the builtin `object`. To silence this warn
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  if dataset.dtypes[column]==np.object:
<ipython-input-11-5d7d7fe4d7c0>:3: DeprecationWarning: `np.object` is a deprecated alias for the builtin `object`. To silence this warn
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  if dataset.dtypes[column]==np.object:
<ipython-input-11-5d7d7fe4d7c0>:3: DeprecationWarning: `np.object` is a deprecated alias for the builtin `object`. To silence this warn
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  if dataset.dtypes[column]==np.object:
<ipython-input-11-5d7d7fe4d7c0>:3: DeprecationWarning: `np.object` is a deprecated alias for the builtin `object`. To silence this warn
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  if dataset.dtypes[column]==np.object:
<ipython-input-11-5d7d7fe4d7c0>:3: DeprecationWarning: `np.object` is a deprecated alias for the builtin `object`. To silence this warn
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  if dataset.dtypes[column]==np.object:
<ipython-input-11-5d7d7fe4d7c0>:3: DeprecationWarning: `np.object` is a deprecated alias for the builtin `object`. To silence this warn
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  if dataset.dtypes[column]==np.object:
```

Correlation using heatmap

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



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

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

Dataset after preprocessing

```
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
```

Split the dataset into predictors and target and make training and testing sets

```
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=False)
```

```
   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
```

```
clf=GaussianNB()
cv_res=cross_val_score(clf,x_train,y_train,cv=10)
print(cv_res.mean()*100)
```

```
76.68213951528749
```

```
clf=DecisionTreeClassifier()
cv_res=cross_val_score(clf,x_train,y_train,cv=10)
print(cv_res.mean()*100)
```

```
74.20353368835151
```

```
clf=RandomForestClassifier(n_estimators=100)
cv_res=cross_val_score(clf,x_train,y_train,cv=10)
print(cv_res.mean()*100)
```

```
76.65793573419688
```

Fit the model with tuned parameters

```
clf=RandomForestClassifier(n_estimators=50,max_features=5,min_samples_leaf=50)
clf.fit(x_train,y_train)
```

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

```
pred=clf.predict(x_test)
pred
```

```
array([1, 1, 1, ..., 0, 0, 0])
```

```
print("Accuracy: %f " % (100*accuracy_score(y_test, pred)))
print(confusion_matrix(y_test,pred))
print(classification_report(y_test,pred))
```

```
Accuracy: 84.589110
[[7520  422]
 [1112  900]]
              precision    recall  f1-score   support

           0       0.87      0.95      0.91      7942
           1       0.68      0.45      0.54      2012

    accuracy                           0.85      9954
   macro avg       0.78      0.70      0.72      9954
weighted avg       0.83      0.85      0.83      9954
```