Importing Necessary Packages

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

df=pd.read_csv(r"C:\Users\Arigala.Adarsh\Downloads\Data-sets\census-
income (7).csv")

df.head()
```

```
   age          workclass   fnlwgt   education   education-num  \
0   39          State-gov    77516   Bachelors              13
1   50   Self-emp-not-inc    83311   Bachelors              13
2   38            Private   215646     HS-grad               9
3   53            Private   234721        11th               7
4   28            Private   338409   Bachelors              13

        marital-status           occupation    relationship    race
sex  \
0        Never-married        Adm-clerical   Not-in-family   White
Male
1   Married-civ-spouse      Exec-managerial         Husband   White
Male
2             Divorced   Handlers-cleaners   Not-in-family   White
Male
3   Married-civ-spouse   Handlers-cleaners         Husband   Black
Male
4   Married-civ-spouse      Prof-specialty            Wife   Black
Female

    capital-gain   capital-loss   hours-per-week  native-country

0           2174              0               40   United-States
<=50K
1              0              0               13   United-States
<=50K
2              0              0               40   United-States
<=50K
3              0              0               40   United-States
<=50K
4              0              0               40            Cuba
<=50K
```

Data Preprocessing

```
df.rename(columns={' ':'Anuual-Income'},inplace=True)
df.head()
```

```
   age          workclass   fnlwgt   education   education-num  \
0   39          State-gov    77516   Bachelors              13
1   50   Self-emp-not-inc    83311   Bachelors              13
2   38            Private   215646     HS-grad               9
3   53            Private   234721        11th               7
4   28            Private   338409   Bachelors              13

        marital-status          occupation    relationship    race
sex  \
0        Never-married        Adm-clerical   Not-in-family   White
Male
1   Married-civ-spouse     Exec-managerial         Husband   White
Male
2             Divorced   Handlers-cleaners   Not-in-family   White
Male
3   Married-civ-spouse   Handlers-cleaners         Husband   Black
Male
4   Married-civ-spouse      Prof-specialty            Wife   Black
Female

    capital-gain   capital-loss   hours-per-week  native-country
Anuual-Income
0           2174              0               40   United-States
<=50K
1              0              0               13   United-States
<=50K
2              0              0               40   United-States
<=50K
3              0              0               40   United-States
<=50K
4              0              0               40            Cuba
<=50K
```

```
df.shape
```

```
(32561, 15)
```

```
df.columns
```

```
Index(['age', ' workclass', ' fnlwgt', ' education', ' education-num',
       ' marital-status', ' occupation', ' relationship', ' race', '
sex',
       ' capital-gain', ' capital-loss', ' hours-per-week', ' native-
country',
       'Anuual-Income'],
      dtype='object')
```

```
df.dtypes
```

```
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
Anuual-Income      object
dtype: object
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32561 entries, 0 to 32560
Data columns (total 15 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   age              32561 non-null  int64 
 1    workclass       32561 non-null  object
 2    fnlwgt          32561 non-null  int64 
 3    education       32561 non-null  object
 4    education-num   32561 non-null  int64 
 5    marital-status  32561 non-null  object
 6    occupation      32561 non-null  object
 7    relationship    32561 non-null  object
 8    race            32561 non-null  object
 9    sex             32561 non-null  object
 10   capital-gain    32561 non-null  int64 
 11   capital-loss    32561 non-null  int64 
 12   hours-per-week  32561 non-null  int64 
 13   native-country  32561 non-null  object
 14  Anuual-Income    32561 non-null  object
dtypes: int64(6), object(9)
memory usage: 3.7+ MB
```

```
df.isnull().sum()
```

```
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
Anuual-Income      0
dtype: int64
```

```
df.duplicated()
```

```
0        False
1        False
2        False
3        False
4        False
         ...
32556    False
32557    False
32558    False
32559    False
32560    False
Length: 32561, dtype: bool
```

```
df.duplicated().sum()
```

```
24
```

```
df.drop_duplicates(inplace=True)
```

```
df.shape
```

```
(32537, 15)
```

Exploratory Data analysis-EDA

```
df.describe()
```

```
                age        fnlwgt   education-num   capital-gain  \
count  32537.000000  3.253700e+04    32537.000000   32537.000000
mean      38.585549  1.897808e+05       10.081815    1078.443741
std       13.637984  1.055565e+05        2.571633    7387.957424
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  2.369930e+05       12.000000       0.000000
max       90.000000  1.484705e+06       16.000000   99999.000000
```

```
        capital-loss   hours-per-week
count   32537.000000     32537.000000
mean       87.368227        40.440329
std       403.101833        12.346889
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
```

```
df.columns
```

```
Index(['age', ' workclass', ' fnlwgt', ' education', ' education-num',
       ' marital-status', ' occupation', ' relationship', ' race', '
sex',
       ' capital-gain', ' capital-loss', ' hours-per-week', ' native-
country',
       'Anuual-Income'],
      dtype='object')
```

```
df[' occupation'].unique()
```

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

```
df[' occupation'].nunique()
```

```
15
```

```
df[' native-country'].unique()
```

```
array([' United-States', ' Cuba', ' Jamaica', ' India', ' ?', '
Mexico',
       ' South', ' Puerto-Rico', ' Honduras', ' England', ' Canada',
       ' Germany', ' Iran', ' Philippines', ' Italy', ' Poland',
       ' Columbia', ' Cambodia', ' Thailand', ' Ecuador', ' Laos',
       ' Taiwan', ' Haiti', ' Portugal', ' Dominican-Republic',
       ' El-Salvador', ' France', ' Guatemala', ' China', ' Japan',
       ' Yugoslavia', ' Peru', ' Outlying-US(Guam-USVI-etc)', '
Scotland',
       ' Trinadad&Tobago', ' Greece', ' Nicaragua', ' Vietnam', '
Hong',
       ' Ireland', ' Hungary', ' Holand-Netherlands'], dtype=object)
```

```
df[' native-country'].nunique()
```

```
42
```

How many people are working as tech support and have an annual income greater than 50k ?

```
df[df[' occupation']==' Tech-support']
```

```
       age   workclass   fnlwgt      education   education-num  \
24      59     Private   109015        HS-grad               9
25      56   Local-gov   216851      Bachelors              13
42      24     Private   172987      Bachelors              13
55      43     Private   237993   Some-college              10
64      29     Private   105598   Some-college              10
...    ...         ...      ...            ...             ...
32396   56     Private   135458        HS-grad               9
32457   33     Private   139057        Masters              14
32546   37     Private   198216     Assoc-acdm              12
32553   32     Private   116138        Masters              14
32556   27     Private   257302     Assoc-acdm              12

            marital-status     occupation    relationship  \
24                Divorced   Tech-support       Unmarried
25      Married-civ-spouse   Tech-support         Husband
42      Married-civ-spouse   Tech-support         Husband
55      Married-civ-spouse   Tech-support         Husband
64                Divorced   Tech-support   Not-in-family
...                    ...            ...             ...
32396             Divorced   Tech-support   Not-in-family
32457   Married-civ-spouse   Tech-support         Husband
32546             Divorced   Tech-support   Not-in-family
32553        Never-married   Tech-support   Not-in-family
32556   Married-civ-spouse   Tech-support            Wife

                      race      sex   capital-gain   capital-loss  \
24                   White   Female              0              0
25                   White     Male              0              0
42                   White     Male              0              0
55                   White     Male              0              0
64                   White     Male              0              0
...                    ...      ...            ...            ...
32396                Black   Female              0              0
32457   Asian-Pac-Islander     Male              0              0
32546                White   Female              0              0
32553   Asian-Pac-Islander     Male              0              0
32556                White   Female              0              0

        hours-per-week  native-country Anuual-Income
24                  40   United-States         <=50K
25                  40   United-States          >50K
42                  50   United-States         <=50K
55                  40   United-States          >50K
64                  58   United-States         <=50K
...                ...             ...           ...
32396               40   United-States         <=50K
```

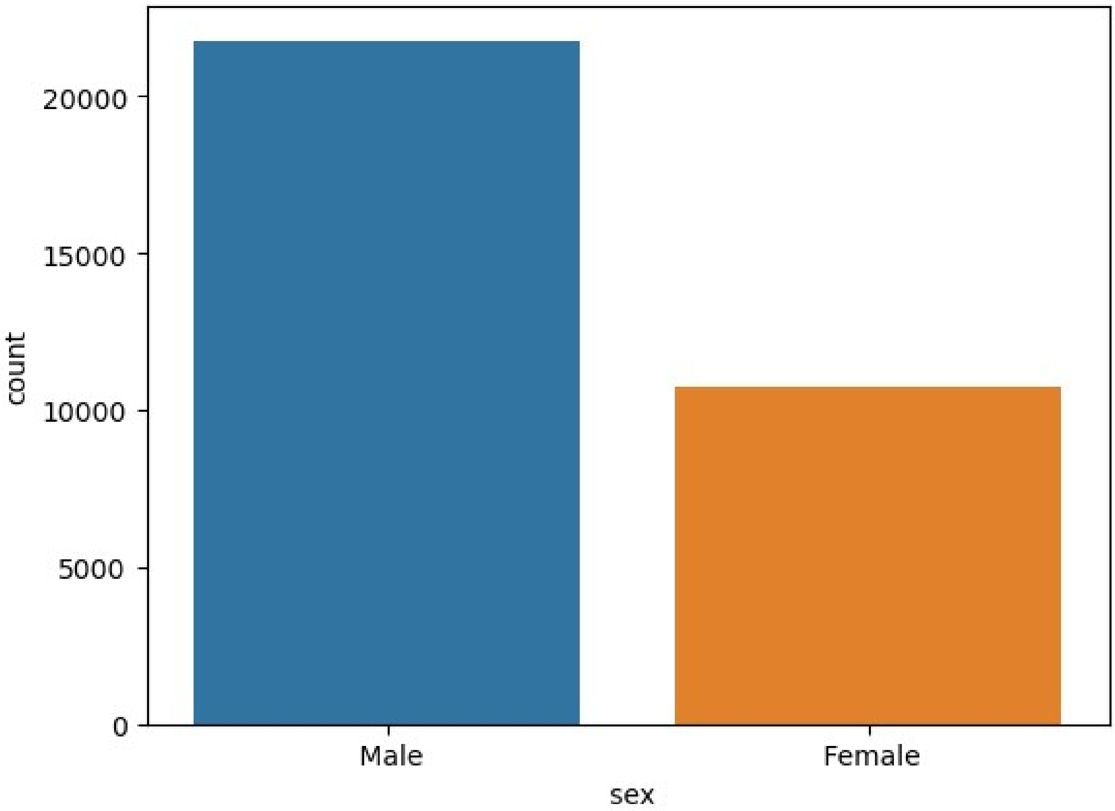
```
32457                  50   United-States          >50K
32546                  40   United-States         <=50K
32553                  11          Taiwan         <=50K
32556                  38   United-States         <=50K

[927 rows x 15 columns]
```

```
df[df[' occupation']==' Tech-support'].count()
```

```
age                   927
 workclass            927
 fnlwgt               927
 education            927
 education-num        927
 marital-status       927
 occupation           927
 relationship         927
 race                 927
 sex                  927
 capital-gain         927
 capital-loss         927
 hours-per-week       927
 native-country       927
Anuual-Income         927
dtype: int64
```

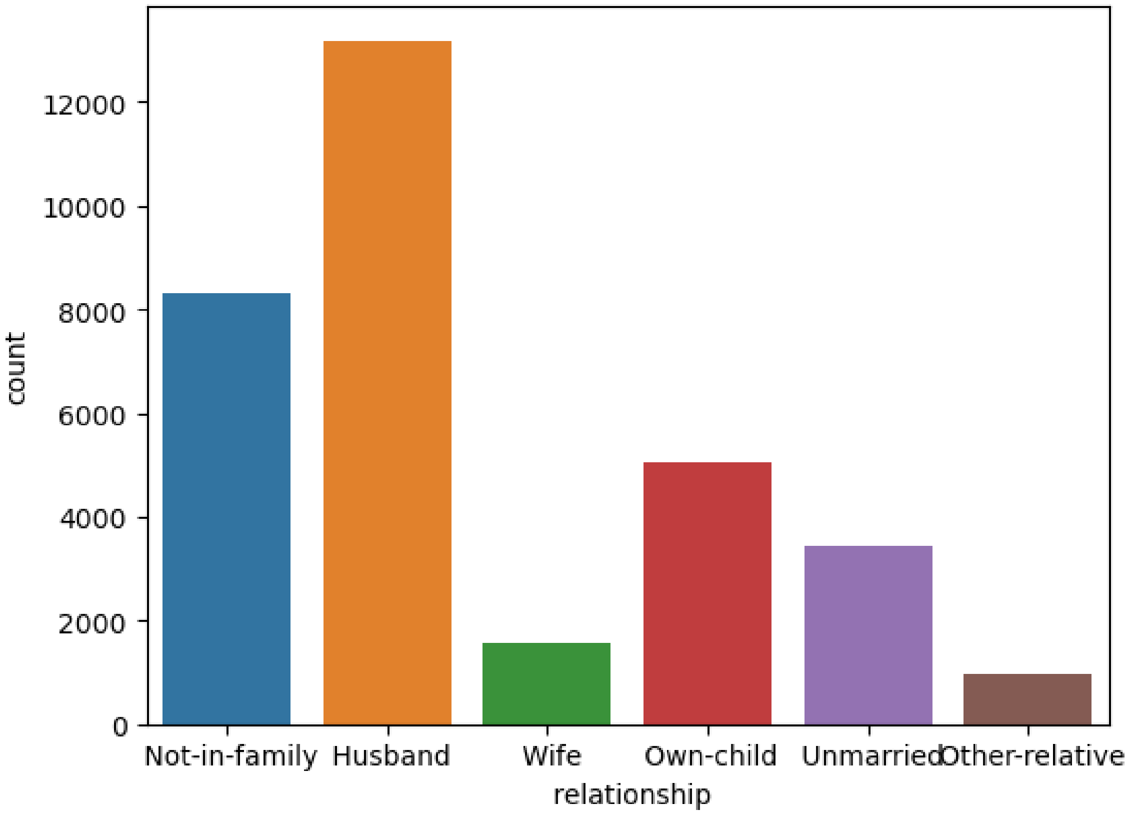
```
#No of  Males and females in the data
sns.countplot(df[' sex'])
plt.show()
```

```
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\seaborn\
_decorators.py:36: FutureWarning: Pass the following variable as a
keyword arg: x. From version 0.12, the only valid positional argument
will be `data`, and passing other arguments without an explicit
keyword will result in an error or misinterpretation.
  warnings.warn(
```



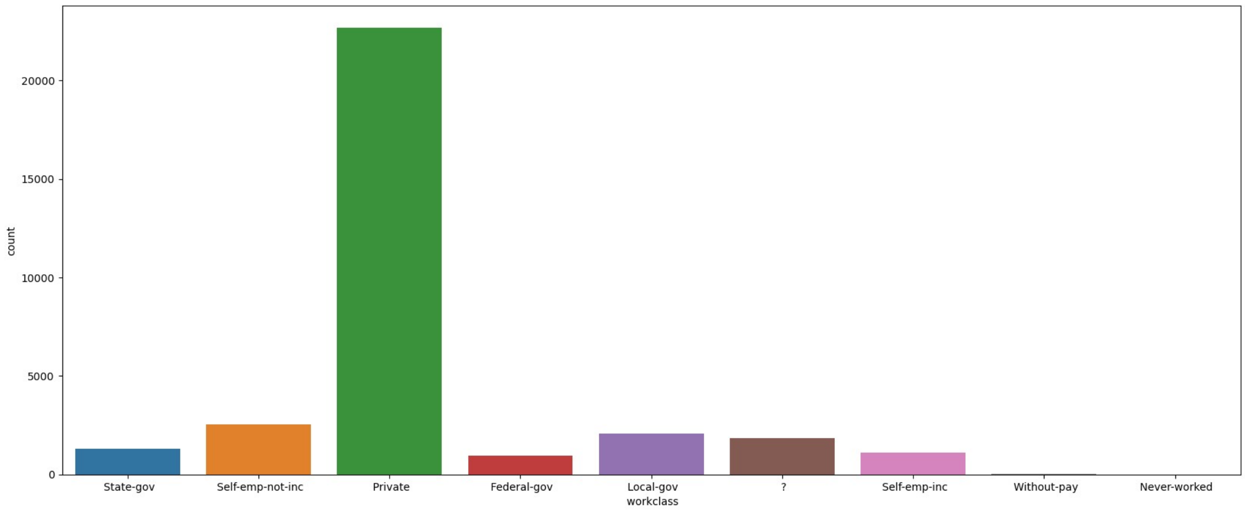
```
#How manay people are married and un married in the censues dataset

sns.countplot(df[' relationship'])
plt.show()
```

```
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\seaborn\
_decorators.py:36: FutureWarning: Pass the following variable as a
keyword arg: x. From version 0.12, the only valid positional argument
will be `data`, and passing other arguments without an explicit
keyword will result in an error or misinterpretation.
  warnings.warn(
```



```
#How many people are working in different sectors
plt.figure(figsize=(20,8))
sns.countplot(df[' workclass'])
plt.show()
```

```
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\seaborn\
_decorators.py:36: FutureWarning: Pass the following variable as a
keyword arg: x. From version 0.12, the only valid positional argument
will be `data`, and passing other arguments without an explicit
keyword will result in an error or misinterpretation.
  warnings.warn(
```



```
#Types of Education
df[' education'].unique()

array([' Bachelors', ' HS-grad', ' 11th', ' Masters', ' 9th',
       ' Some-college', ' Assoc-acdm', ' Assoc-voc', ' 7th-8th',
       ' Doctorate', ' Prof-school', ' 5th-6th', ' 10th', ' 1st-4th',
       ' Preschool', ' 12th'], dtype=object)
```

How many people are having private work classes and are not from the United States of America?

```
df[' workclass'].unique()

array([' State-gov', ' Self-emp-not-inc', ' Private', ' Federal-gov',
       ' Local-gov', ' ?', ' Self-emp-inc', ' Without-pay',
       ' Never-worked'], dtype=object)

df[(df[' workclass']==' Private')&(df[' native-country']!=' United-
States')]

       age  workclass   fnlwgt   education   education-num  \
4       28    Private   338409   Bachelors              13
6       49    Private   160187         9th               5
14      40    Private   121772   Assoc-voc              11
15      34    Private   245487     7th-8th               4
35      48    Private   242406        11th               7
...    ...        ...      ...         ...             ...
32508   45    Private   155093        10th               6
32510   39    Private   107302     HS-grad               9
32533   54    Private   337992   Bachelors              13
32547   43    Private   260761     HS-grad               9
32553   32    Private   116138     Masters              14

               marital-status         occupation    relationship  \
4          Married-civ-spouse     Prof-specialty            Wife
6       Married-spouse-absent      Other-service   Not-in-family
```

```
14         Married-civ-spouse        Craft-repair         Husband
15         Married-civ-spouse    Transport-moving         Husband
35              Never-married   Machine-op-inspct       Unmarried
...                       ...                 ...             ...
32508                Divorced       Other-service   Not-in-family
32510      Married-civ-spouse      Prof-specialty         Husband
32533      Married-civ-spouse     Exec-managerial         Husband
32547      Married-civ-spouse   Machine-op-inspct         Husband
32553           Never-married        Tech-support   Not-in-family

                      race      sex   capital-gain   capital-loss  \
4                    Black   Female              0              0
6                    Black   Female              0              0
14      Asian-Pac-Islander     Male              0              0
15      Amer-Indian-Eskimo     Male              0              0
35                   White     Male              0              0
...                    ...      ...            ...            ...
32508                Black   Female              0              0
32510                White     Male              0              0
32533   Asian-Pac-Islander     Male              0              0
32547                White     Male              0              0
32553   Asian-Pac-Islander     Male              0              0

        hours-per-week       native-country Anuual-Income
4                   40                 Cuba         <=50K
6                   16              Jamaica         <=50K
14                  40                    ?          >50K
15                  45               Mexico         <=50K
35                  40          Puerto-Rico         <=50K
...                ...                  ...           ...
32508               38   Dominican-Republic         <=50K
32510               45                    ?          >50K
32533               50                Japan          >50K
32547               40               Mexico         <=50K
32553               11               Taiwan         <=50K

[2554 rows x 15 columns]
```

```
df[(df[' workclass']==' Private')&(df[' native-country']!=' United-States')][' workclass'].count()
```

```
2554
```

How many people are either having Annual Income(last column) less than or equal to 50k or their working hours is greater than or equal to 40 hrs

```
df[(df['Anuual-Income']==' <=50K')&(df[' hours-per-week']>40)]
```

```
       age          workclass   fnlwgt      education   education-num
\
13      32            Private   205019     Assoc-acdm              12
```

```
15      34            Private   245487        7th-8th               4
18      38            Private    28887           11th               7
28      39            Private   367260        HS-grad               9
30      23          Local-gov   190709     Assoc-acdm              12
...    ...                ...      ...            ...             ...
32537   30            Private   345898        HS-grad               9
32543   45          Local-gov   119199     Assoc-acdm              12
32548   65   Self-emp-not-inc    99359    Prof-school              15
32550   43   Self-emp-not-inc    27242   Some-college              10
32552   43            Private    84661      Assoc-voc              11

            marital-status         occupation    relationship  \
13           Never-married              Sales   Not-in-family
15      Married-civ-spouse   Transport-moving         Husband
18      Married-civ-spouse              Sales         Husband
28                Divorced    Exec-managerial   Not-in-family
30           Never-married    Protective-serv   Not-in-family
...                    ...                ...             ...
32537        Never-married       Craft-repair   Not-in-family
32543             Divorced     Prof-specialty       Unmarried
32548        Never-married     Prof-specialty   Not-in-family
32550   Married-civ-spouse       Craft-repair         Husband
32552   Married-civ-spouse              Sales         Husband

                      race      sex   capital-gain   capital-loss  \
13                   Black     Male              0              0
15      Amer-Indian-Eskimo     Male              0              0
18                   White     Male              0              0
28                   White     Male              0              0
30                   White     Male              0              0
...                    ...      ...            ...            ...
32537                Black     Male              0              0
32543                White   Female              0              0
32548                White     Male           1086              0
32550                White     Male              0              0
32552                White     Male              0              0

        hours-per-week  native-country Anuual-Income
13                  50   United-States         <=50K
15                  45          Mexico         <=50K
```

```
18                    50   United-States         <=50K
28                    80   United-States         <=50K
30                    52   United-States         <=50K
...                  ...             ...           ...
32537                 46   United-States         <=50K
32543                 48   United-States         <=50K
32548                 60   United-States         <=50K
32550                 50   United-States         <=50K
32552                 45   United-States         <=50K

[5721 rows x 15 columns]
```

```
df[(df['Anuual-Income']==' <=50K')&(df[' hours-per-week']>40)].count()
```

```
age                  5721
 workclass           5721
 fnlwgt              5721
 education           5721
 education-num       5721
 marital-status      5721
 occupation          5721
 relationship        5721
 race                5721
 sex                 5721
 capital-gain        5721
 capital-loss        5721
 hours-per-week      5721
 native-country      5721
Anuual-Income        5721
dtype: int64
```

Popoulation According to the country

```
l=df[' native-country'].unique()
print(l)
```

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

```
d={}
for i in l:
    pop=df[df[' native-country']==i][' native-country'].count()
    d[i]=pop
population=pd.DataFrame(list(d.items()),columns=['Native
country','Population'])
population.head(60)
```

```
                    Native country  Population
0                    United-States       29153
1                             Cuba          95
2                          Jamaica          81
3                            India         100
4                                ?         582
5                           Mexico         639
6                            South          80
7                      Puerto-Rico         114
8                         Honduras          13
9                          England          90
10                          Canada         121
11                         Germany         137
12                            Iran          43
13                     Philippines         198
14                           Italy          73
15                          Poland          60
16                        Columbia          59
17                        Cambodia          19
18                        Thailand          18
19                         Ecuador          28
20                            Laos          18
21                          Taiwan          51
22                           Haiti          44
23                        Portugal          37
24              Dominican-Republic          70
25                     El-Salvador         106
26                          France          29
27                       Guatemala          62
28                           China          75
29                           Japan          62
30                      Yugoslavia          16
31                            Peru          31
32    Outlying-US(Guam-USVI-etc)          14
33                        Scotland          12
34                  Trinadad&Tobago          19
35                          Greece          29
36                       Nicaragua          34
37                         Vietnam          67
38                            Hong          20
39                         Ireland          24
40                         Hungary          13
41              Holand-Netherlands           1
```

Observations from table

- United-States has Highest Population

- Holand-Netherlands has lowest Population

```
df['Anuual-Income'].unique()
```

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

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder
```

```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

# Fit and transform the 'workclass' column
df[' workclass'] = le.fit_transform(df[' workclass'])
df[' fnlwgt'] = le.fit_transform(df[' fnlwgt'])
df[' education'] = le.fit_transform(df[' education'])
df[' marital-status'] = le.fit_transform(df[' marital-status'])
df[' occupation'] = le.fit_transform(df[' occupation'])
df[' relationship'] = le.fit_transform(df[' relationship'])
df[' sex'] = le.fit_transform(df[' sex'])
df[' race'] = le.fit_transform(df[' race'])
df[' native-country'] = le.fit_transform(df[' native-country'])
df['Anuual-Income'] = le.fit_transform(df['Anuual-Income'])
```

```
df.head()
```

```
   age   workclass   fnlwgt   education   education-num   marital-
status  \
0   39           7     2671           9              13
4
1   50           6     2926           9              13
2
2   38           4    14086          11               9
0
3   53           4    15336           1               7
2
4   28           4    19355           9              13
2

    occupation   relationship   race   sex   capital-gain   capital-
loss  \
0            1              1      4     1           2174
0
1            4              0      4     1              0
0
2            6              1      4     1              0
0
3            6              0      2     1              0
0
4           10              5      2     0              0
```

```
0

    hours-per-week   native-country  Anuual-Income
0               40               39              0
1               13               39              0
2               40               39              0
3               40               39              0
4               40                5              0
```

```
df.drop(['age'],axis=1,inplace=True)
```

```
df
```

```
        workclass   fnlwgt   education   education-num   marital-
status  \
0               7     2671           9              13   
4  
1               6     2926           9              13   
2  
2               4    14086          11               9   
0  
3               4    15336           1               7   
2  
4               4    19355           9              13   
2  
...           ...      ...         ...             ...              ..
.  
32556           4    16528           7              12   
2  
32557           4     8080          11               9   
2  
32558           4     7883          11               9   
6  
32559           4    12881          11               9   
4  
32560           5    17825          11               9   
2  

        occupation   relationship   race   sex   capital-gain  
capital-loss  \
0                1              1      4     1           2174   
0  
1                4              0      4     1              0   
0  
2                6              1      4     1              0   
0  
3                6              0      2     1              0   
0  
4               10              5      2     0              0   
0  
```

```
...            ...            ...    ...   ...             ...
...
32556           13              5      4     0               0
0
32557            7              0      4     1               0
0
32558            1              4      4     0               0
0
32559            1              3      4     1               0
0
32560            4              5      4     0           15024
0

        hours-per-week   native-country   Anuual-Income
0                   40               39               0
1                   13               39               0
2                   40               39               0
3                   40               39               0
4                   40                5               0
...                ...              ...             ...
32556               38               39               0
32557               40               39               1
32558               40               39               0
32559               20               39               0
32560               40               39               1

[32537 rows x 14 columns]
```

Segregration of Dependent and Independent Columns

```
x=df.drop('Anuual-Income',axis=1)
y=df['Anuual-Income']

x.head(20)

     workclass   fnlwgt   education   education-num   marital-
status  \
0            7     2671           9              13                 4

1            6     2926           9              13                 2

2            4    14086          11               9                 0

3            4    15336           1               7                 2

4            4    19355           9              13                 2
```

```
5            4    17700          12              14                2

6            4     8536           6               5                3

7            6    13620          11               9                2

8            4     1318          12              14                4

9            4     8460           9              13                2

10           4    17530          15              10                2

11           7     7077           9              13                2

12           4     5772           9              13                4

13           4    13217           7              12                4

14           4     5722           8              11                2

15           4    15963           5               4                2

16           6    10163          11               9                4

17           4    11257          11               9                4

18           4      314           1               7                2

19           6    17974          12              14                0


     occupation   relationship   race   sex   capital-gain   capital-
loss  \
0             1              1      4     1           2174
0
1             4              0      4     1              0
0
2             6              1      4     1              0
0
3             6              0      2     1              0
0
4            10              5      2     0              0
0
5             4              5      4     0              0
0
6             8              1      2     0              0
0
7             4              0      4     1              0
0
8            10              1      4     0          14084
0
```

```
9             4              0      4     1           5178
0
10            4              0      2     1              0
0
11           10              0      1     1              0
0
12            1              3      4     0              0
0
13           12              1      2     1              0
0
14            3              0      1     1              0
0
15           14              0      0     1              0
0
16            5              3      4     1              0
0
17            7              4      4     1              0
0
18           12              0      4     1              0
0
19            4              4      4     0              0
0

     hours-per-week   native-country
0                40               39
1                13               39
2                40               39
3                40               39
4                40                5
5                40               39
6                16               23
7                45               39
8                50               39
9                40               39
10               80               39
11               40               19
12               30               39
13               50               39
14               40                0
15               45               26
16               35               39
17               40               39
18               50               39
19               45               39
```

```
y.head(20)
```

```
0     0
1     0
2     0
```

```
3     0
4     0
5     0
6     0
7     1
8     1
9     1
10    1
11    1
12    0
13    0
14    1
15    0
16    0
17    0
18    0
19    1
Name: Anuual-Income, dtype: int32
```

Logistic Regression Model

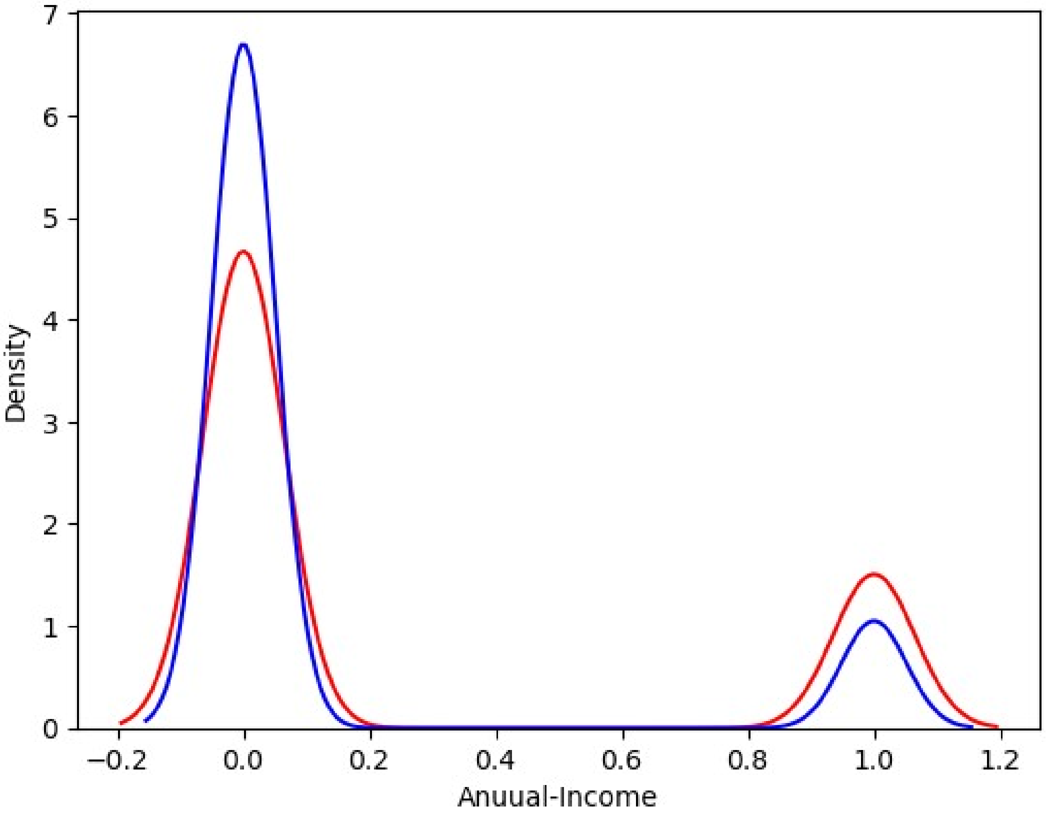
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, 
test_size=0.4, random_state=42)
from sklearn.linear_model import LogisticRegression
model=LogisticRegression()
model.fit(x_train,y_train)
pred=model.predict(x_test)from sklearn.linear_model import 
LogisticRegression
model=LogisticRegression()
model.fit(x_train,y_train)
pred=model.predict(x_test)
```

```
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
linear_model\_logistic.py:460: ConvergenceWarning: lbfgs failed to
converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as 
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
from sklearn.metrics import *
print("Accuracy",accuracy_score(y_test,pred))
print("recall",recall_score(y_test,pred))
```

```
Accuracy 0.8086822896657703
recall 0.3857052896725441
```

```
sns.distplot(y_test,hist=False,color="red")
sns.distplot(pred,hist=False,color="blue")
plt.show()
```

```
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\seaborn\
distributions.py:2619: FutureWarning: `distplot` is a deprecated
function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar
flexibility) or `kdeplot` (an axes-level function for kernel density
plots).
  warnings.warn(msg, FutureWarning)
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\seaborn\
distributions.py:2619: FutureWarning: `distplot` is a deprecated
function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar
flexibility) or `kdeplot` (an axes-level function for kernel density
plots).
  warnings.warn(msg, FutureWarning)
```



_________Hyperparameter Tunning of Logistic Regression _________

```
logistic_hyperparameter = {
    'C': [0.001, 0.01, 0.1, 1, 10, 100],
    'penalty': ['l1', 'l2'],
    'solver': ['liblinear', 'saga']
}

logistic_hyper= RandomizedSearchCV(
    estimator=LogisticRegression(random_state=31),
    param_distributions=logistic_hyperparameter,
    n_jobs=-1,
    n_iter=100,
    cv=3,
    verbose=3,
    scoring='accuracy',
    random_state=0
)

logistic_hyper.fit(x_train, y_train)
```

```
print("Logistic Regression Best Score:", logistic_hyper.best_score_)
print("Logistic Regression Best Params:", logistic_hyper.best_params_)
```

```
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
model_selection\_search.py:307: UserWarning: The total space of
parameters 24 is smaller than n_iter=100. Running 24 iterations. For
exhaustive searches, use GridSearchCV.
  warnings.warn(
```

```
Fitting 3 folds for each of 24 candidates, totalling 72 fits
Logistic Regression Best Score: 0.8249155079142386
Logistic Regression Best Params: {'solver': 'liblinear', 'penalty':
'l1', 'C': 1}
```

```
model=LogisticRegression(solver='liblinear',penalty='l1',C=1)
model.fit(x_train,y_train)
pred=model.predict(x_test)
print("Accuracy",accuracy_score(y_test,pred))
print("recall",recall_score(y_test,pred))
```

```
Accuracy 0.8237418363426815
recall 0.4474181360201511
```

Decision Tree Model

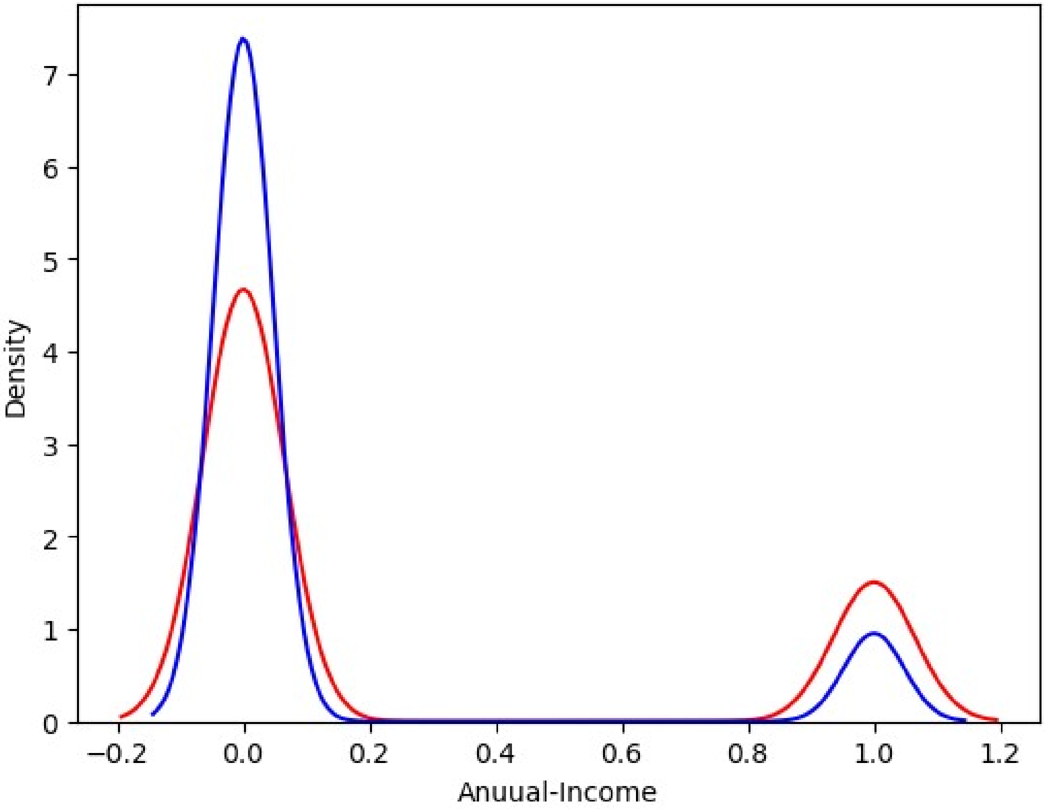
```
from sklearn.tree import DecisionTreeClassifier
model=DecisionTreeClassifier()
model.fit(x_train,y_train)
pred=model.predict(x_test)

print("Accuracy",accuracy_score(y_test,pred))
print("recall",recall_score(y_test,pred))
```

```
Accuracy 0.8026892047637342
recall 0.6026448362720404
```

```
sns.distplot(y_test,hist=False,color="red")
sns.distplot(pred,hist=False,color="blue")
plt.show()
```

```
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\seaborn\
distributions.py:2619: FutureWarning: `distplot` is a deprecated
function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar
flexibility) or `kdeplot` (an axes-level function for kernel density
plots).
  warnings.warn(msg, FutureWarning)
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\seaborn\
distributions.py:2619: FutureWarning: `distplot` is a deprecated
```

```
function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar
flexibility) or `kdeplot` (an axes-level function for kernel density
plots).
  warnings.warn(msg, FutureWarning)
```



_________Hyperparameter Tunning of Decision Tree Classifier _________

```
from sklearn.model_selection import RandomizedSearchCV
decisionhyperparameter={
    'max_depth':[None,range(1,20)],
    'min_samples_split':[2,5,10],
    'min_samples_leaf':[1,2,4],
    'max_features':['auto','sqrt','log2'],
    'criterion': ['gini', 'entropy']

}
decision_hyper=RandomizedSearchCV(estimator=DecisionTreeClassifier(ran
dom_state=31),
param_distributions=decisionhyperparameter,
                               n_jobs=-1,
                               n_iter=100,
                               cv=3,
                                verbose=3,
                                scoring='accuracy',
                                random_state=0
                            )
decision_hyper.fit(x_train,y_train)
print(decision_hyper.best_score_)
print(decision_hyper.best_params_)
```

```
Fitting 3 folds for each of 100 candidates, totalling 300 fits
0.8438170001278626
{'min_samples_split': 10, 'min_samples_leaf': 4, 'max_features':
'log2', 'max_depth': None, 'criterion': 'gini'}
```

```
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
model_selection\_validation.py:425: FitFailedWarning:
204 fits failed out of a total of 300.
The score on these train-test partitions for these parameters will be
set to nan.
If these failures are not expected, you can try to debug them by
setting error_score='raise'.

Below are more details about the failures:
----------------------------------------------------------------------
----------
150 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
model_selection\_validation.py", line 729, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
base.py", line 1145, in wrapper
    estimator._validate_params()
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
base.py", line 638, in _validate_params
    validate_parameter_constraints(
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
utils\_param_validation.py", line 95, in
validate_parameter_constraints
    raise InvalidParameterError(
sklearn.utils._param_validation.InvalidParameterError: The 'max_depth'
parameter of DecisionTreeClassifier must be an int in the range [1,
inf) or None. Got range(1, 20) instead.

----------------------------------------------------------------------
----------
34 fits failed with the following error:
```

```
Traceback (most recent call last):
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
model_selection\_validation.py", line 729, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
base.py", line 1145, in wrapper
    estimator._validate_params()
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
base.py", line 638, in _validate_params
    validate_parameter_constraints(
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
utils\_param_validation.py", line 95, in
validate_parameter_constraints
    raise InvalidParameterError(
sklearn.utils._param_validation.InvalidParameterError: The
'max_features' parameter of DecisionTreeClassifier must be an int in
the range [1, inf), a float in the range (0.0, 1.0], a str among
{'sqrt', 'log2'} or None. Got 'auto' instead.

----------------------------------------------------------------------
----------
20 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
model_selection\_validation.py", line 729, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
base.py", line 1145, in wrapper
    estimator._validate_params()
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
base.py", line 638, in _validate_params
    validate_parameter_constraints(
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
utils\_param_validation.py", line 95, in
validate_parameter_constraints
    raise InvalidParameterError(
sklearn.utils._param_validation.InvalidParameterError: The
'max_features' parameter of DecisionTreeClassifier must be an int in
the range [1, inf), a float in the range (0.0, 1.0], a str among
{'log2', 'sqrt'} or None. Got 'auto' instead.

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
model_selection\_search.py:979: UserWarning: One or more of the test
scores are non-finite: [       nan 0.81917842 0.83920705        nan
0.84156318        nan
        nan        nan 0.84156318        nan 0.83597968 0.82952564
        nan        nan 0.83101104        nan 0.82952564 0.81733414
        nan        nan        nan        nan 0.843817          nan
        nan        nan 0.83920705        nan        nan        nan
 0.84356099        nan        nan        nan        nan        nan
        nan 0.81016295        nan        nan 0.81088001 0.84294637
        nan        nan 0.84356099        nan        nan        nan
 0.84156318 0.843817          nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
        nan 0.82998695        nan        nan 0.83992418        nan
        nan        nan        nan        nan        nan        nan
 0.83992418 0.82665717 0.81917842        nan        nan        nan
        nan 0.84356099 0.84294637 0.82998695        nan 0.81088001
 0.83597968 0.84156318        nan        nan        nan        nan
        nan        nan 0.82665717        nan 0.83413573        nan
        nan 0.84356099        nan        nan]
  warnings.warn(
```

```
model=DecisionTreeClassifier(min_samples_split= 10, min_samples_leaf=
4, max_features= 'log2', max_depth= None, criterion= 'gini')
model.fit(x_train,y_train)
pred=model.predict(x_test)
print("Accuracy",accuracy_score(y_test,pred))
print("recall",recall_score(y_test,pred))
```

```
Accuracy 0.8341913177103343
recall 0.5639168765743073
```

Random Forest Model

```
from sklearn.ensemble import RandomForestClassifier
model=RandomForestClassifier()
model.fit(x_train,y_train)
pred=model.predict(x_test)
```

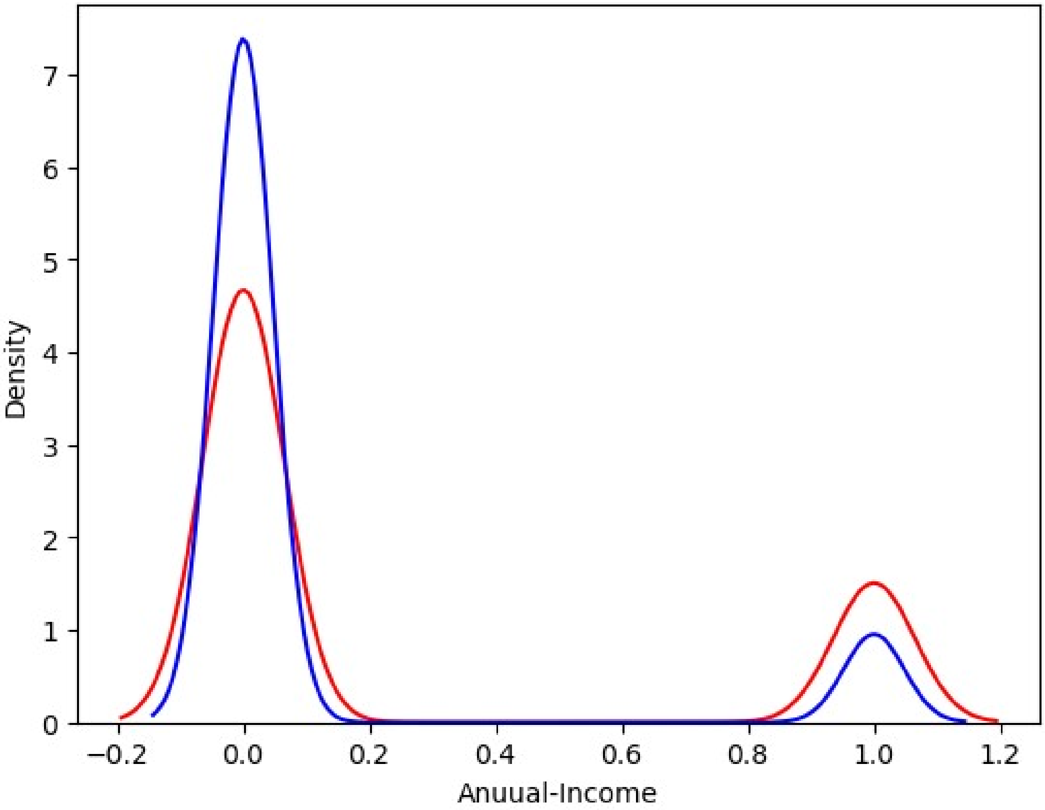
```
print("Accuracy",accuracy_score(y_test,pred))
print("recall",recall_score(y_test,pred))
```

```
Accuracy 0.8375720322704572
recall 0.6001259445843828
```

```
sns.distplot(y_test,hist=False,color="red")
sns.distplot(pred,hist=False,color="blue")
plt.show()
```

```
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\seaborn\
distributions.py:2619: FutureWarning: `distplot` is a deprecated
function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar
flexibility) or `kdeplot` (an axes-level function for kernel density
plots).
  warnings.warn(msg, FutureWarning)
```

```
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\seaborn\
distributions.py:2619: FutureWarning: `distplot` is a deprecated
function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar
flexibility) or `kdeplot` (an axes-level function for kernel density
plots).
  warnings.warn(msg, FutureWarning)
```



________Hyperparameter Tunning of Random Forest Classifier ________

```
from sklearn.model_selection import RandomizedSearchCV
Randomforesthyperparameter={
    'n_estimators':[50,100,200,300,400,500],
    'max_depth':[None,range(1,20)],
    'min_samples_split':[2,5,10],
    'min_samples_leaf':[1,2,4],
    'max_features':['auto','sqrt','log2']



}
```

```
Random_hyper=RandomizedSearchCV(estimator=RandomForestClassifier(rando
m_state=31),

param_distributions=Randomforesthyperparameter,
                               n_jobs=-1,
                               n_iter=100,
                               cv=3,
                                verbose=3,
                                scoring='accuracy',
                                random_state=0
                            )
Random_hyper.fit(x_train,y_train)
print(Random_hyper.best_score_)
print(Random_hyper.best_params_)
```

```
Fitting 3 folds for each of 100 candidates, totalling 300 fits

C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
model_selection\_validation.py:425: FitFailedWarning:
192 fits failed out of a total of 300.
The score on these train-test partitions for these parameters will be
set to nan.
If these failures are not expected, you can try to debug them by
setting error_score='raise'.

Below are more details about the failures:
----------------------------------------------------------------------
----------
138 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
model_selection\_validation.py", line 729, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
base.py", line 1145, in wrapper
    estimator._validate_params()
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
base.py", line 638, in _validate_params
    validate_parameter_constraints(
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
utils\_param_validation.py", line 95, in
validate_parameter_constraints
    raise InvalidParameterError(
sklearn.utils._param_validation.InvalidParameterError: The 'max_depth'
parameter of RandomForestClassifier must be an int in the range [1,
inf) or None. Got range(1, 20) instead.

----------------------------------------------------------------------
----------
40 fits failed with the following error:
```

```
Traceback (most recent call last):
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
model_selection\_validation.py", line 729, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
base.py", line 1145, in wrapper
    estimator._validate_params()
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
base.py", line 638, in _validate_params
    validate_parameter_constraints(
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
utils\_param_validation.py", line 95, in
validate_parameter_constraints
    raise InvalidParameterError(
sklearn.utils._param_validation.InvalidParameterError: The
'max_features' parameter of RandomForestClassifier must be an int in
the range [1, inf), a float in the range (0.0, 1.0], a str among
{'log2', 'sqrt'} or None. Got 'auto' instead.

-----------------------------------------------------------------------
----------
14 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
model_selection\_validation.py", line 729, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
base.py", line 1145, in wrapper
    estimator._validate_params()
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
base.py", line 638, in _validate_params
    validate_parameter_constraints(
  File "C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
utils\_param_validation.py", line 95, in
validate_parameter_constraints
    raise InvalidParameterError(
sklearn.utils._param_validation.InvalidParameterError: The
'max_features' parameter of RandomForestClassifier must be an int in
the range [1, inf), a float in the range (0.0, 1.0], a str among
{'sqrt', 'log2'} or None. Got 'auto' instead.

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
model_selection\_search.py:979: UserWarning: One or more of the test
scores are non-finite: [        nan 0.85477914        nan        nan
0.86067002 0.85908201
 0.85923566        nan 0.85421566 0.85949179        nan        nan
 0.85903079        nan        nan 0.85488156        nan        nan
 0.85892832 0.84714657        nan 0.85687933 0.85887709        nan
 0.86010657        nan        nan        nan        nan        nan
        nan        nan        nan        nan        nan        nan
 0.84724912        nan 0.85682817 0.85339608 0.85841608        nan
 0.85856984 0.85324241 0.85979927 0.86036274 0.8593895         nan
        nan        nan        nan        nan 0.84786382 0.85682817
        nan 0.86010657        nan 0.84765895        nan        nan
        nan        nan 0.858826          nan        nan        nan
        nan        nan        nan        nan        nan        nan
        nan        nan 0.85892832 0.85856984 0.85938935        nan
 0.85687933        nan 0.85918446        nan        nan        nan
 0.85928688        nan        nan 0.86000413 0.85713553        nan
        nan        nan 0.85687938        nan        nan        nan
        nan        nan        nan        nan]
  warnings.warn(

0.8606700152109527
{'n_estimators': 200, 'min_samples_split': 10, 'min_samples_leaf': 2, 
'max_features': 'log2', 'max_depth': None}
```

```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, recall_score

# Assuming x_train, y_train, x_test, and y_test are defined

model = RandomForestClassifier(n_estimators=200, min_samples_split=10, 
min_samples_leaf=2, max_features='log2', max_depth=None)
model.fit(x_train, y_train)
pred = model.predict(x_test)

print("Accuracy:", accuracy_score(y_test, pred))
print("Recall:", recall_score(y_test, pred))
```

```
Accuracy: 0.8559354590856704
Recall: 0.5954030226700252
```

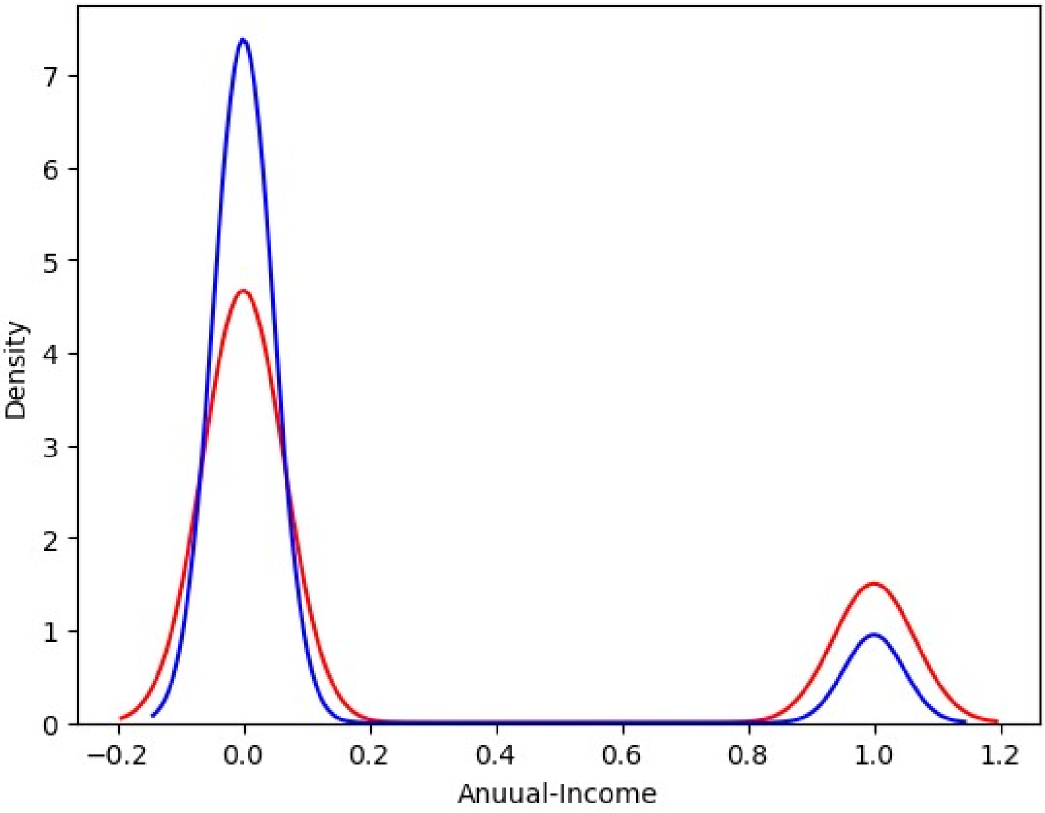
Naive Bayes Model

```python
from sklearn.naive_bayes import GaussianNB

model=GaussianNB()
model.fit(x_train,y_train)
pred=model.predict(x_test)

print("Accuracy",accuracy_score(y_test,pred))
print("recall",recall_score(y_test,pred))
```

```
Accuracy 0.7975412985017287
recall 0.3195843828715365
```

```
sns.distplot(y_test,hist=False,color="red")
sns.distplot(pred,hist=False,color="blue")
plt.show()
```

```
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\seaborn\
distributions.py:2619: FutureWarning: `distplot` is a deprecated
function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar
flexibility) or `kdeplot` (an axes-level function for kernel density
plots).
  warnings.warn(msg, FutureWarning)
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\seaborn\
distributions.py:2619: FutureWarning: `distplot` is a deprecated
function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar
flexibility) or `kdeplot` (an axes-level function for kernel density
plots).
  warnings.warn(msg, FutureWarning)
```



KNN Model

```
from sklearn.neighbors import KNeighborsClassifier
```

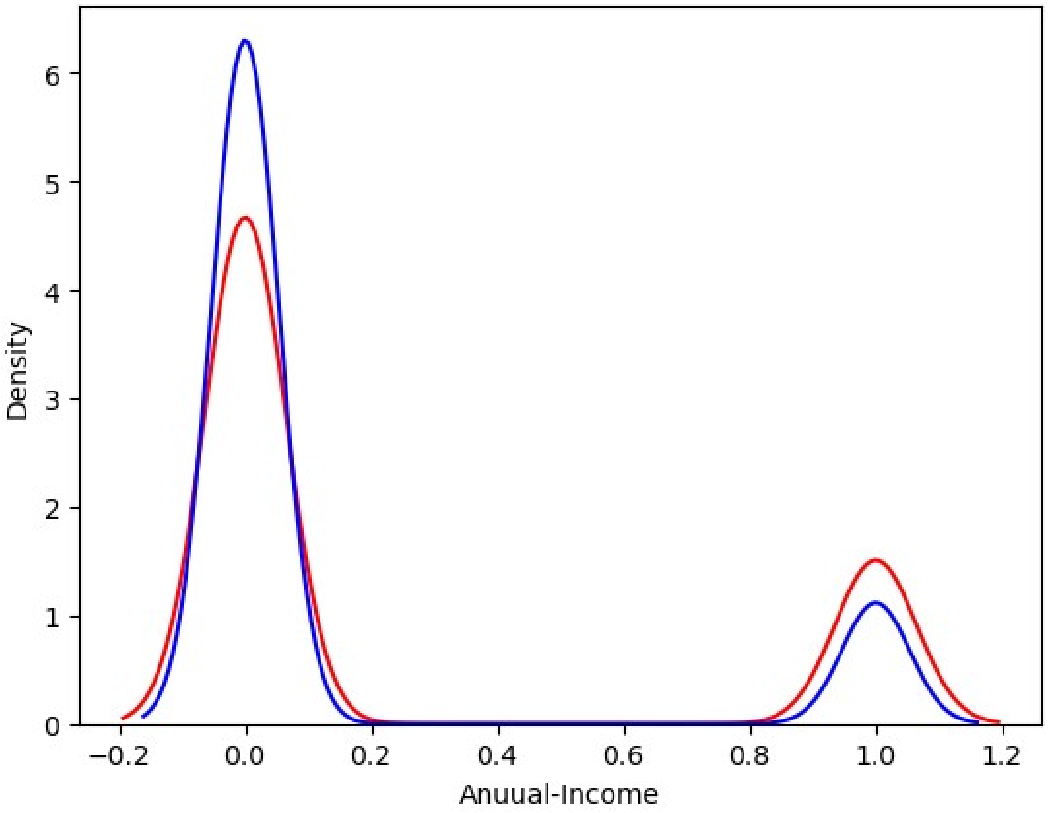
```
model=KNeighborsClassifier(n_neighbors=3)
model.fit(x_train,y_train)
prediction=model.predict(x_test)
```

```
print("Accuracy",accuracy_score(y_test,pred))
print("recall",recall_score(y_test,pred))
```

```
Accuracy 0.8237418363426815
recall 0.4474181360201511
```

```
sns.distplot(y_test,hist=False,color="red")
sns.distplot(pred,hist=False,color="blue")
plt.show()
```

```
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\seaborn\
distributions.py:2619: FutureWarning: `distplot` is a deprecated
function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar
flexibility) or `kdeplot` (an axes-level function for kernel density
plots).
  warnings.warn(msg, FutureWarning)
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\seaborn\
distributions.py:2619: FutureWarning: `distplot` is a deprecated
function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar
flexibility) or `kdeplot` (an axes-level function for kernel density
plots).
  warnings.warn(msg, FutureWarning)
```



Hyperparameter Tunning of KNN

```
from sklearn.model_selection import RandomizedSearchCV
knnhyperparameter={
    'n_neighbors': [3,5,7,9,11],
    'weights': ['uniform', 'distance'],
    'metric': ['euclidean', 'manhattan']

}
knn_hyper=RandomizedSearchCV(estimator=KNeighborsClassifier(),
                               param_distributions=knnhyperparameter,
                               n_jobs=-1,
                               n_iter=100,
                               cv=3,
                                verbose=3,
                                scoring='accuracy',
                                random_state=0
                            )
knn_hyper.fit(x_train,y_train)
print(knn_hyper.best_score_)
print(knn_hyper.best_params_)
```

```
C:\Users\Arigala.Adarsh\anaconda3\lib\site-packages\sklearn\
model_selection\_search.py:307: UserWarning: The total space of
parameters 20 is smaller than n_iter=100. Running 20 iterations. For
exhaustive searches, use GridSearchCV.
  warnings.warn(

Fitting 3 folds for each of 20 candidates, totalling 60 fits
0.8093433695835167
{'weights': 'distance', 'n_neighbors': 11, 'metric': 'manhattan'}
```

```
model=KNeighborsClassifier(weights= 'distance', n_neighbors= 11,
metric= 'manhattan')
model.fit(x_train,y_train)
prediction=model.predict(x_test)
print("Accuracy",accuracy_score(y_test,pred))
print("recall",recall_score(y_test,pred))
```

```
Accuracy 0.8237418363426815
recall 0.4474181360201511
```

- **Among above Models Random forest gives best accuracy than other Models**