In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

In [2]:
```
import warnings

warnings.filterwarnings('ignore')
```

In [3]:
```
data = 'C:/Users/surya/Downloads/sistem-cerdas/adult.csv'

df = pd.read_csv(data, header=None, sep=',\s')
```

In [4]:
```
df.shape
```

Out[4]:
```
(32561, 15)
```

In [5]:
```
df.head()
```

Out[5]:

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40

In [6]:
```
col_names = ['age', 'workclass', 'fnlwgt', 'education', 'education_num', 'marital_st
             'race', 'sex', 'capital_gain', 'capital_loss', 'hours_per_week', 'nativ

df.columns = col_names

df.columns
```

Out[6]:
```
Index(['age', 'workclass', 'fnlwgt', 'education', 'education_num',
       'marital_status', 'occupation', 'relationship', 'race', 'sex',
```

```
       'capital_gain', 'capital_loss', 'hours_per_week', 'native_country',
       'income'],
      dtype='object')
```

In [7]:
```
df.head()
```

Out[7]:

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

In [8]:
```
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  income          32561 non-null  object
dtypes: int64(6), object(9)
memory usage: 3.7+ MB
```

In [9]:
```
categorical = [var for var in df.columns if df[var].dtype=='O']

print('There are {} categorical variables\n'.format(len(categorical)))

print('The categorical variables are :\n\n', categorical)
```

```
There are 9 categorical variables

The categorical variables are :

 ['workclass', 'education', 'marital_status', 'occupation', 'relationship', 'race',
'sex', 'native_country', 'income']
```

In [10]:
```
df[categorical].head()
```

Out[10]:

	workclass	education	marital_status	occupation	relationship	race	sex	native_country	in
0	State-gov	Bachelors	Never-married	Adm-clerical	Not-in-family	White	Male	United-States	<
1	Self-emp-not-inc	Bachelors	Married-civ-spouse	Exec-managerial	Husband	White	Male	United-States	<
2	Private	HS-grad	Divorced	Handlers-cleaners	Not-in-family	White	Male	United-States	<
3	Private	11th	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	United-States	<
4	Private	Bachelors	Married-civ-spouse	Prof-specialty	Wife	Black	Female	Cuba	<

In [11]:

```
df[categorical].isnull().sum()
```

Out[11]:

```
workclass         0
education         0
marital_status    0
occupation        0
relationship      0
race              0
sex               0
native_country    0
income            0
dtype: int64
```

In [12]:

```
for var in categorical:

    print(df[var].value_counts())
```

```
Private             22696
Self-emp-not-inc     2541
Local-gov            2093
?                    1836
State-gov            1298
Self-emp-inc         1116
Federal-gov           960
Without-pay            14
Never-worked            7
Name: workclass, dtype: int64
HS-grad         10501
Some-college     7291
Bachelors        5355
Masters          1723
Assoc-voc        1382
11th             1175
Assoc-acdm       1067
10th              933
7th-8th           646
Prof-school       576
9th               514
12th              433
Doctorate         413
5th-6th           333
1st-4th           168
Preschool          51
Name: education, dtype: int64
Married-civ-spouse       14976
Never-married            10683
Divorced                  4443
```

```
Separated                 1025
Widowed                    993
Married-spouse-absent      418
Married-AF-spouse           23
Name: marital_status, dtype: int64
Prof-specialty       4140
Craft-repair         4099
Exec-managerial      4066
Adm-clerical         3770
Sales                3650
Other-service        3295
Machine-op-inspct    2002
?                    1843
Transport-moving     1597
Handlers-cleaners    1370
Farming-fishing       994
Tech-support          928
Protective-serv       649
Priv-house-serv       149
Armed-Forces            9
Name: occupation, dtype: int64
Husband           13193
Not-in-family      8305
Own-child          5068
Unmarried          3446
Wife               1568
Other-relative      981
Name: relationship, dtype: int64
White                 27816
Black                  3124
Asian-Pac-Islander     1039
Amer-Indian-Eskimo      311
Other                   271
Name: race, dtype: int64
Male      21790
Female    10771
Name: sex, dtype: int64
United-States                 29170
Mexico                          643
?                               583
Philippines                     198
Germany                         137
Canada                          121
Puerto-Rico                     114
El-Salvador                     106
India                           100
Cuba                             95
England                          90
Jamaica                          81
South                            80
China                            75
Italy                            73
Dominican-Republic               70
Vietnam                          67
Guatemala                        64
Japan                            62
Poland                           60
Columbia                         59
Taiwan                           51
Haiti                            44
Iran                             43
Portugal                         37
Nicaragua                        34
Peru                             31
Greece                           29
France                           29
Ecuador                          28
Ireland                          24
Hong                             20
```

```
Cambodia                        19
Trinadad&Tobago                 19
Laos                            18
Thailand                        18
Yugoslavia                      16
Outlying-US(Guam-USVI-etc)      14
Hungary                         13
Honduras                        13
Scotland                        12
Holand-Netherlands               1
Name: native_country, dtype: int64
<=50K    24720
>50K      7841
Name: income, dtype: int64
```

In [13]:

```python
for var in categorical:

    print(df[var].value_counts()/np.float(len(df)))
```

```
Private             0.697030
Self-emp-not-inc    0.078038
Local-gov           0.064279
?                   0.056386
State-gov           0.039864
Self-emp-inc        0.034274
Federal-gov         0.029483
Without-pay         0.000430
Never-worked        0.000215
Name: workclass, dtype: float64
HS-grad         0.322502
Some-college    0.223918
Bachelors       0.164461
Masters         0.052916
Assoc-voc       0.042443
11th            0.036086
Assoc-acdm      0.032769
10th            0.028654
7th-8th         0.019840
Prof-school     0.017690
9th             0.015786
12th            0.013298
Doctorate       0.012684
5th-6th         0.010227
1st-4th         0.005160
Preschool       0.001566
Name: education, dtype: float64
Married-civ-spouse       0.459937
Never-married            0.328092
Divorced                 0.136452
Separated                0.031479
Widowed                  0.030497
Married-spouse-absent    0.012837
Married-AF-spouse        0.000706
Name: marital_status, dtype: float64
Prof-specialty       0.127146
Craft-repair         0.125887
Exec-managerial      0.124873
Adm-clerical         0.115783
Sales                0.112097
Other-service        0.101195
Machine-op-inspct    0.061485
?                    0.056601
Transport-moving     0.049046
Handlers-cleaners    0.042075
Farming-fishing      0.030527
Tech-support         0.028500
Protective-serv      0.019932
Priv-house-serv      0.004576
```

```
Armed-Forces         0.000276
Name: occupation, dtype: float64
Husband           0.405178
Not-in-family     0.255060
Own-child         0.155646
Unmarried         0.105832
Wife              0.048156
Other-relative    0.030128
Name: relationship, dtype: float64
White                 0.854274
Black                 0.095943
Asian-Pac-Islander    0.031909
Amer-Indian-Eskimo    0.009551
Other                 0.008323
Name: race, dtype: float64
Male      0.669205
Female    0.330795
Name: sex, dtype: float64
United-States                 0.895857
Mexico                        0.019748
?                             0.017905
Philippines                   0.006081
Germany                       0.004207
Canada                        0.003716
Puerto-Rico                   0.003501
El-Salvador                   0.003255
India                         0.003071
Cuba                          0.002918
England                       0.002764
Jamaica                       0.002488
South                         0.002457
China                         0.002303
Italy                         0.002242
Dominican-Republic            0.002150
Vietnam                       0.002058
Guatemala                     0.001966
Japan                         0.001904
Poland                        0.001843
Columbia                      0.001812
Taiwan                        0.001566
Haiti                         0.001351
Iran                          0.001321
Portugal                      0.001136
Nicaragua                     0.001044
Peru                          0.000952
Greece                        0.000891
France                        0.000891
Ecuador                       0.000860
Ireland                       0.000737
Hong                          0.000614
Cambodia                      0.000584
Trinadad&Tobago               0.000584
Laos                          0.000553
Thailand                      0.000553
Yugoslavia                    0.000491
Outlying-US(Guam-USVI-etc)    0.000430
Hungary                       0.000399
Honduras                      0.000399
Scotland                      0.000369
Holand-Netherlands            0.000031
Name: native_country, dtype: float64
<=50K    0.75919
>50K     0.24081
Name: income, dtype: float64
```

In [14]:

```
df.workclass.unique()
```

Out[14]:

```
array(['State-gov', 'Self-emp-not-inc', 'Private', 'Federal-gov',
```

```
       'Local-gov', '?', 'Self-emp-inc', 'Without-pay', 'Never-worked'],
      dtype=object)
```

In [15]:
```
df.workclass.value_counts()
```

Out[15]:
```
Private             22696
Self-emp-not-inc     2541
Local-gov            2093
?                    1836
State-gov            1298
Self-emp-inc         1116
Federal-gov           960
Without-pay            14
Never-worked            7
Name: workclass, dtype: int64
```

In [16]:
```
df['workclass'].replace('?', np.NaN, inplace=True)
```

In [17]:
```
df.workclass.value_counts()
```

Out[17]:
```
Private             22696
Self-emp-not-inc     2541
Local-gov            2093
State-gov            1298
Self-emp-inc         1116
Federal-gov           960
Without-pay            14
Never-worked            7
Name: workclass, dtype: int64
```

In [18]:
```
df.occupation.unique()
```

Out[18]:
```
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)
```

In [19]:
```
df.occupation.value_counts()
```

Out[19]:
```
Prof-specialty       4140
Craft-repair         4099
Exec-managerial      4066
Adm-clerical         3770
Sales                3650
Other-service        3295
Machine-op-inspct    2002
?                    1843
Transport-moving     1597
Handlers-cleaners    1370
Farming-fishing       994
Tech-support          928
Protective-serv       649
Priv-house-serv       149
Armed-Forces            9
Name: occupation, dtype: int64
```

In [20]:
```
df['occupation'].replace('?', np.NaN, inplace=True)
```

In [21]:
```
df.occupation.value_counts()
```

Out[21]:
```
Prof-specialty       4140
Craft-repair         4099
Exec-managerial      4066
Adm-clerical         3770
Sales                3650
Other-service        3295
Machine-op-inspct    2002
Transport-moving     1597
Handlers-cleaners    1370
Farming-fishing       994
Tech-support          928
Protective-serv       649
Priv-house-serv       149
Armed-Forces            9
Name: occupation, dtype: int64
```

In [22]:
```
df.native_country.unique()
```

Out[22]:
```
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)
```

In [23]:
```
df.native_country.value_counts()
```

Out[23]:
```
United-States                 29170
Mexico                          643
?                               583
Philippines                     198
Germany                         137
Canada                          121
Puerto-Rico                     114
El-Salvador                     106
India                           100
Cuba                             95
England                          90
Jamaica                          81
South                            80
China                            75
Italy                            73
Dominican-Republic               70
Vietnam                          67
Guatemala                        64
Japan                            62
Poland                           60
Columbia                         59
Taiwan                           51
Haiti                            44
Iran                             43
Portugal                         37
Nicaragua                        34
Peru                             31
Greece                           29
France                           29
Ecuador                          28
Ireland                          24
Hong                             20
Cambodia                         19
Trinadad&Tobago                  19
Laos                             18
Thailand                         18
```

```
Yugoslavia                       16
Outlying-US(Guam-USVI-etc)       14
Hungary                          13
Honduras                         13
Scotland                         12
Holand-Netherlands                1
Name: native_country, dtype: int64
```

In [24]:

```
df['native_country'].replace('?', np.NaN, inplace=True)
```

In [25]:

```
df.native_country.value_counts()
```

Out[25]:

```
United-States                 29170
Mexico                          643
Philippines                     198
Germany                         137
Canada                          121
Puerto-Rico                     114
El-Salvador                     106
India                           100
Cuba                             95
England                          90
Jamaica                          81
South                            80
China                            75
Italy                            73
Dominican-Republic               70
Vietnam                          67
Guatemala                        64
Japan                            62
Poland                           60
Columbia                         59
Taiwan                           51
Haiti                            44
Iran                             43
Portugal                         37
Nicaragua                        34
Peru                             31
Greece                           29
France                           29
Ecuador                          28
Ireland                          24
Hong                             20
Trinadad&Tobago                  19
Cambodia                         19
Thailand                         18
Laos                             18
Yugoslavia                       16
Outlying-US(Guam-USVI-etc)       14
Hungary                          13
Honduras                         13
Scotland                         12
Holand-Netherlands                1
Name: native_country, dtype: int64
```

In [26]:

```
df[categorical].isnull().sum()
```

Out[26]:

```
workclass         1836
education            0
marital_status       0
occupation        1843
relationship         0
race                 0
sex                  0
native_country     583
```

```
income            0
dtype: int64
```

In [27]:

```
for var in categorical:

    print(var, ' contains ', len(df[var].unique()), ' labels')
```

```
workclass  contains  9  labels
education  contains  16  labels
marital_status  contains  7  labels
occupation  contains  15  labels
relationship  contains  6  labels
race  contains  5  labels
sex  contains  2  labels
native_country  contains  42  labels
income  contains  2  labels
```

In [28]:

```
numerical = [var for var in df.columns if df[var].dtype!='O']

print('There are {} numerical variables\n'.format(len(numerical)))

print('The numerical variables are :', numerical)
```

```
There are 6 numerical variables

The numerical variables are : ['age', 'fnlwgt', 'education_num', 'capital_gain', 'ca
pital_loss', 'hours_per_week']
```

In [29]:

```
df[numerical].head()
```

Out[29]:

	age	fnlwgt	education_num	capital_gain	capital_loss	hours_per_week
0	39	77516	13	2174	0	40
1	50	83311	13	0	0	13
2	38	215646	9	0	0	40
3	53	234721	7	0	0	40
4	28	338409	13	0	0	40

In [30]:

```
df[numerical].isnull().sum()
```

Out[30]:

```
age               0
fnlwgt            0
education_num     0
capital_gain      0
capital_loss      0
hours_per_week    0
dtype: int64
```

In [31]:

```
X = df.drop(['income'], axis=1)

y = df['income']
```

In [32]:

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_st
```

In [33]:
```
X_train.shape, X_test.shape
```

Out[33]:
```
((22792, 14), (9769, 14))
```

In [34]:
```
X_train.dtypes
```

Out[34]:
```
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
dtype: object
```

In [35]:
```
categorical = [col for col in X_train.columns if X_train[col].dtypes == 'O']

categorical
```

Out[35]:
```
['workclass',
 'education',
 'marital_status',
 'occupation',
 'relationship',
 'race',
 'sex',
 'native_country']
```

In [36]:
```
numerical = [col for col in X_train.columns if X_train[col].dtypes != 'O']

numerical
```

Out[36]:
```
['age',
 'fnlwgt',
 'education_num',
 'capital_gain',
 'capital_loss',
 'hours_per_week']
```

In [37]:
```
X_train[categorical].isnull().mean()
```

Out[37]:
```
workclass         0.055985
education         0.000000
marital_status    0.000000
occupation        0.056072
relationship      0.000000
race              0.000000
sex               0.000000
native_country    0.018164
dtype: float64
```

In [38]:
```
for col in categorical:
    if X_train[col].isnull().mean()>0:
        print(col, (X_train[col].isnull().mean()))
```

```
workclass 0.055984555984555984
occupation 0.05607230607230607
native_country 0.018164268164268166
```

In [39]:

```
for df2 in [X_train, X_test]:
    df2['workclass'].fillna(X_train['workclass'].mode()[0], inplace=True)
    df2['occupation'].fillna(X_train['occupation'].mode()[0], inplace=True)
    df2['native_country'].fillna(X_train['native_country'].mode()[0], inplace=True)
```

In [40]:

```
X_train[categorical].isnull().sum()
```

Out[40]:

```
workclass         0
education         0
marital_status    0
occupation        0
relationship      0
race              0
sex               0
native_country    0
dtype: int64
```

In [41]:

```
X_test[categorical].isnull().sum()
```

Out[41]:

```
workclass         0
education         0
marital_status    0
occupation        0
relationship      0
race              0
sex               0
native_country    0
dtype: int64
```

In [42]:

```
X_train.isnull().sum()
```

Out[42]:

```
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
dtype: int64
```

In [43]:

```
X_test.isnull().sum()
```

Out[43]:

```
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
dtype: int64
```

In [44]:
```
categorical
```

Out[44]:
```
['workclass',
 'education',
 'marital_status',
 'occupation',
 'relationship',
 'race',
 'sex',
 'native_country']
```

In [45]:
```
X_train[categorical].head()
```

Out[45]:

	workclass	education	marital_status	occupation	relationship	race	sex	native_country
32098	Private	HS-grad	Married-civ-spouse	Craft-repair	Husband	White	Male	United-States
25206	State-gov	HS-grad	Divorced	Adm-clerical	Unmarried	White	Female	United-States
23491	Private	Some-college	Married-civ-spouse	Sales	Husband	White	Male	United-States
12367	Private	HS-grad	Never-married	Craft-repair	Not-in-family	White	Male	Guatemala
7054	Private	7th-8th	Never-married	Craft-repair	Not-in-family	White	Male	Germany

In [48]:
```
import category_encoders as ce
```

In [49]:
```
encoder = ce.OneHotEncoder(cols=['workclass', 'education', 'marital_status', 'occupa
                                 'race', 'sex', 'native_country'])

X_train = encoder.fit_transform(X_train)

X_test = encoder.transform(X_test)
```

In [50]:
```
X_train.head()
```

Out[50]:

	age	workclass_1	workclass_2	workclass_3	workclass_4	workclass_5	workclass_6	workclas
32098	45	1	0	0	0	0	0	
25206	47	0	1	0	0	0	0	
23491	48	1	0	0	0	0	0	
12367	29	1	0	0	0	0	0	
7054	23	1	0	0	0	0	0	

5 rows × 105 columns

In [51]:
```
X_train.shape
```

Out[51]: (22792, 105)

In [52]:
```
X_test.head()
```

Out[52]:

	age	workclass_1	workclass_2	workclass_3	workclass_4	workclass_5	workclass_6	workclas
22278	27	1	0	0	0	0	0	
8950	27	1	0	0	0	0	0	
7838	25	1	0	0	0	0	0	
16505	46	1	0	0	0	0	0	
19140	45	1	0	0	0	0	0	

5 rows × 105 columns

In [53]:
```
X_test.shape
```

Out[53]: (9769, 105)

In [54]:
```
cols = X_train.columns
```

In [55]:
```
from sklearn.preprocessing import RobustScaler

scaler = RobustScaler()

X_train = scaler.fit_transform(X_train)

X_test = scaler.transform(X_test)
```

In [56]:
```
X_train = pd.DataFrame(X_train, columns=[cols])
```

In [57]:
```
X_test = pd.DataFrame(X_test, columns=[cols])
```

In [58]:
```
X_train.head()
```

Out[58]:

	age	workclass_1	workclass_2	workclass_3	workclass_4	workclass_5	workclass_6	workclass_7
0	0.40	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.50	-1.0	1.0	0.0	0.0	0.0	0.0	0.0
2	0.55	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	-0.40	0.0	0.0	0.0	0.0	0.0	0.0	0.0

	age	workclass_1	workclass_2	workclass_3	workclass_4	workclass_5	workclass_6	workclass_7
4	-0.70	0.0	0.0	0.0	0.0	0.0	0.0	0.0

5 rows × 105 columns

In [59]:
```
from sklearn.naive_bayes import GaussianNB

gnb = GaussianNB()

gnb.fit(X_train, y_train)
```

Out[59]:
```
GaussianNB()
```

In [60]:
```
y_pred = gnb.predict(X_test)

y_pred
```

Out[60]:
```
array(['<=50K', '<=50K', '>50K', ..., '>50K', '<=50K', '<=50K'],
      dtype='<U5')
```

In [61]:
```
from sklearn.metrics import accuracy_score

print('Model accuracy score: {0:0.4f}'. format(accuracy_score(y_test, y_pred)))
```

```
Model accuracy score: 0.8083
```

In [62]:
```
y_pred_train = gnb.predict(X_train)

y_pred_train
```

Out[62]:
```
array(['>50K', '<=50K', '>50K', ..., '<=50K', '>50K', '<=50K'],
      dtype='<U5')
```

In [63]:
```
print('Training-set accuracy score: {0:0.4f}'. format(accuracy_score(y_train, y_pred
```

```
Training-set accuracy score: 0.8067
```

In [64]:
```
print('Training set score: {:.4f}'.format(gnb.score(X_train, y_train)))

print('Test set score: {:.4f}'.format(gnb.score(X_test, y_test)))
```

```
Training set score: 0.8067
Test set score: 0.8083
```

In [65]:
```
y_test.value_counts()
```

Out[65]:
```
<=50K    7407
>50K     2362
Name: income, dtype: int64
```

In [66]:
```
null_accuracy = (7407/(7407+2362))

print('Null accuracy score: {0:0.4f}'. format(null_accuracy))
```

```
Null accuracy score: 0.7582
```

In [67]:

```python
from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test, y_pred)

print('Confusion matrix\n\n', cm)

print('\nTrue Positives(TP) = ', cm[0,0])

print('\nTrue Negatives(TN) = ', cm[1,1])

print('\nFalse Positives(FP) = ', cm[0,1])

print('\nFalse Negatives(FN) = ', cm[1,0])
```

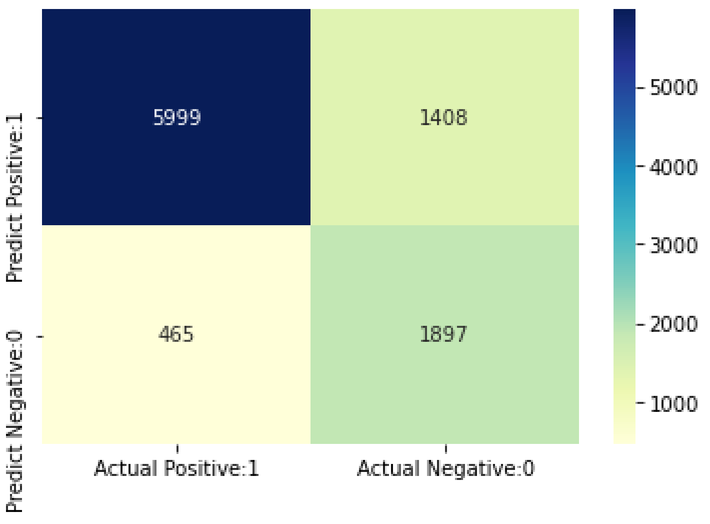
```
Confusion matrix

 [[5999 1408]
 [ 465 1897]]

True Positives(TP) =  5999

True Negatives(TN) =  1897

False Positives(FP) =  1408

False Negatives(FN) =  465
```

In [68]:

```python
cm_matrix = pd.DataFrame(data=cm, columns=['Actual Positive:1', 'Actual Negative:0']
                                 index=['Predict Positive:1', 'Predict Negative:0'])

sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')
```

Out[68]:

```
<AxesSubplot:>
```



In [69]:

```python
from sklearn.metrics import classification_report

print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

        <=50K       0.93      0.81      0.86      7407
         >50K       0.57      0.80      0.67      2362

     accuracy                           0.81      9769
    macro avg       0.75      0.81      0.77      9769
```

```
weighted avg       0.84      0.81      0.82      9769
```

In [70]:

```
TP = cm[0,0]
TN = cm[1,1]
FP = cm[0,1]
FN = cm[1,0]
```

In [71]:

```
classification_accuracy = (TP + TN) / float(TP + TN + FP + FN)

print('Classification accuracy : {0:0.4f}'.format(classification_accuracy))
```

```
Classification accuracy : 0.8083
```

In [72]:

```
classification_error = (FP + FN) / float(TP + TN + FP + FN)

print('Classification error : {0:0.4f}'.format(classification_error))
```

```
Classification error : 0.1917
```

In [73]:

```
precision = TP / float(TP + FP)

print('Precision : {0:0.4f}'.format(precision))
```

```
Precision : 0.8099
```

In [74]:

```
recall = TP / float(TP + FN)

print('Recall or Sensitivity : {0:0.4f}'.format(recall))
```

```
Recall or Sensitivity : 0.9281
```

In [75]:

```
true_positive_rate = TP / float(TP + FN)

print('True Positive Rate : {0:0.4f}'.format(true_positive_rate))
```

```
True Positive Rate : 0.9281
```

In [76]:

```
false_positive_rate = FP / float(FP + TN)

print('False Positive Rate : {0:0.4f}'.format(false_positive_rate))
```

```
False Positive Rate : 0.4260
```

In [77]:

```
specificity = TN / (TN + FP)

print('Specificity : {0:0.4f}'.format(specificity))
```

```
Specificity : 0.5740
```

In [78]:

```
y_pred_prob = gnb.predict_proba(X_test)[0:10]

y_pred_prob
```

Out[78]:

```
array([[9.99999426e-01, 5.74152436e-07],
       [9.99687907e-01, 3.12093456e-04],
       [1.54405602e-01, 8.45594398e-01],
       [1.73624321e-04, 9.99826376e-01],
       [8.20121011e-09, 9.99999992e-01],
```

```
        [8.76844580e-01, 1.23155420e-01],
        [9.99999927e-01, 7.32876705e-08],
        [9.99993460e-01, 6.53998797e-06],
        [9.87738143e-01, 1.22618575e-02],
        [9.99999996e-01, 4.01886317e-09]])
```

In [79]:

```
y_pred_prob_df = pd.DataFrame(data=y_pred_prob, columns=['Prob of - <=50K', 'Prob of

y_pred_prob_df
```

Out[79]:

	Prob of - <=50K	Prob of - >50K
0	9.999994e-01	5.741524e-07
1	9.996879e-01	3.120935e-04
2	1.544056e-01	8.455944e-01
3	1.736243e-04	9.998264e-01
4	8.201210e-09	1.000000e+00
5	8.768446e-01	1.231554e-01
6	9.999999e-01	7.328767e-08
7	9.999935e-01	6.539988e-06
8	9.877381e-01	1.226186e-02
9	1.000000e+00	4.018863e-09

In [80]:

```
gnb.predict_proba(X_test)[0:10, 1]
```

Out[80]:

```
array([5.74152436e-07, 3.12093456e-04, 8.45594398e-01, 9.99826376e-01,
       9.99999992e-01, 1.23155420e-01, 7.32876705e-08, 6.53998797e-06,
       1.22618575e-02, 4.01886317e-09])
```

In [81]:

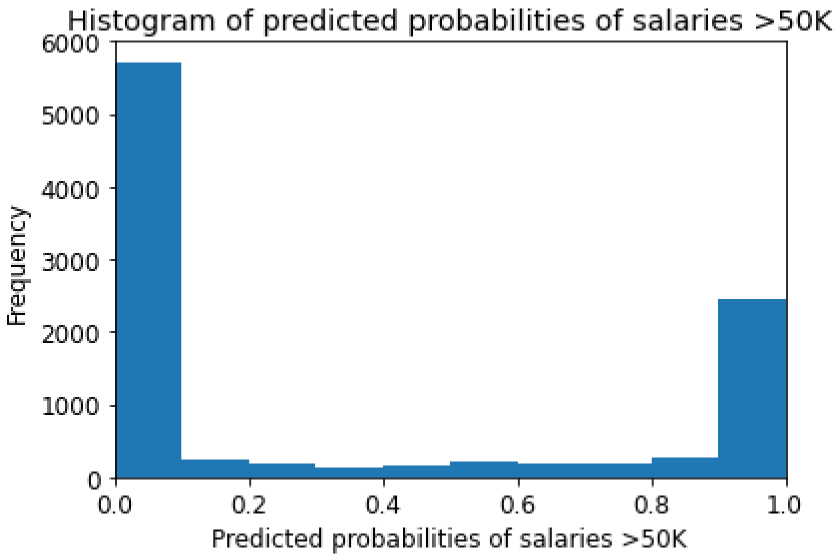
```
y_pred1 = gnb.predict_proba(X_test)[:, 1]
```

In [82]:

```
# adjust the font size
plt.rcParams['font.size'] = 12

# plot histogram with 10 bins
plt.hist(y_pred1, bins = 10)

# set the title of predicted probabilities
plt.title('Histogram of predicted probabilities of salaries >50K')

# set the x-axis limit
plt.xlim(0,1)

# set the title
plt.xlabel('Predicted probabilities of salaries >50K')
plt.ylabel('Frequency')
```

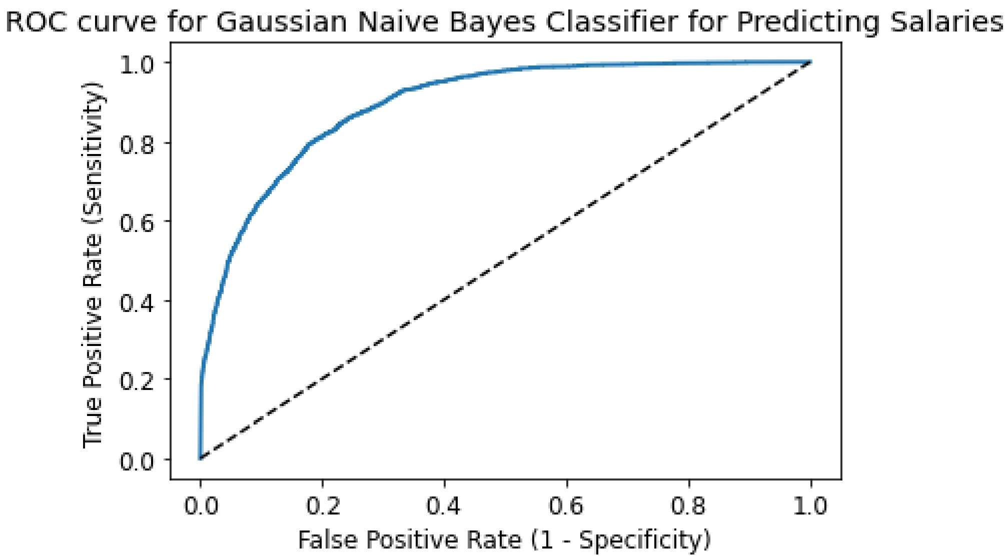
Out[82]:

```
Text(0, 0.5, 'Frequency')
```



In [83]:

```
# plot ROC Curve

from sklearn.metrics import roc_curve

fpr, tpr, thresholds = roc_curve(y_test, y_pred1, pos_label = '>50K')

plt.figure(figsize=(6,4))

plt.plot(fpr, tpr, linewidth=2)

plt.plot([0,1], [0,1], 'k--' )

plt.rcParams['font.size'] = 12

plt.title('ROC curve for Gaussian Naive Bayes Classifier for Predicting Salaries')

plt.xlabel('False Positive Rate (1 - Specificity)')

plt.ylabel('True Positive Rate (Sensitivity)')

plt.show()
```



In [84]:

```
# compute ROC AUC

from sklearn.metrics import roc_auc_score
```

```
ROC_AUC = roc_auc_score(y_test, y_pred1)

print('ROC AUC : {:.4f}'.format(ROC_AUC))
```

```
ROC AUC : 0.8941
```

In [85]:

```
# calculate cross-validated ROC AUC

from sklearn.model_selection import cross_val_score

Cross_validated_ROC_AUC = cross_val_score(gnb, X_train, y_train, cv=5, scoring='roc_

print('Cross validated ROC AUC : {:.4f}'.format(Cross_validated_ROC_AUC))
```

```
Cross validated ROC AUC : 0.8938
```

In [86]:

```
# Applying 10-Fold Cross Validation

from sklearn.model_selection import cross_val_score

scores = cross_val_score(gnb, X_train, y_train, cv = 10, scoring='accuracy')

print('Cross-validation scores:{}'.format(scores))
```

```
Cross-validation scores:[0.81359649 0.80438596 0.81175954 0.8056165  0.79596314 0.79
684072
 0.81044318 0.81175954 0.80210619 0.81044318]
```

In [87]:

```
# compute Average cross-validation score

print('Average cross-validation score: {:.4f}'.format(scores.mean()))
```

```
Average cross-validation score: 0.8063
```

In [ ]: