—----------import libraries—----------------
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import KFold,cross_val_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report,accuracy_score
from sklearn.preprocessing import StandardScaler,RobustScaler
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix,classification_report

—---------read dataset—-----------------
data1 = pd.read_csv('Downloads/SalaryData_Train(1).csv')
data1.head(3)

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	sex	capitalgain	capitalloss	hoursperweek	native	Salary
0	39	State-gov	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50K
1	50	Self-emp-not-inc	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50K
2	38	Private	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=50K

data2 = pd.read_csv('Downloads/SalaryData_Test(1).csv')
data2.head(3)

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	sex	capitalgain	capitalloss	hoursperweek	native	Salary
0	25	Private	11th	7	Never-married	Machine-op-inspct	Own-child	Black	Male	0	0	40	United-States	<=50K
1	38	Private	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	White	Male	0	0	50	United-States	<=50K
2	28	Local-gov	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	White	Male	0	0	40	United-States	>50K

—---------shape—----------

```
data1.shape
```

```
(30161, 14)
```

```
data2.shape
```

```
(15060, 14)
```

—--------info—---------------------
data1.info()

```
RangeIndex: 30161 entries, 0 to 30160
Data columns (total 14 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   age            30161 non-null  int64
 1   workclass      30161 non-null  object
 2   education      30161 non-null  object
 3   educationno    30161 non-null  int64
 4   maritalstatus  30161 non-null  object
 5   occupation     30161 non-null  object
 6   relationship   30161 non-null  object
 7   race           30161 non-null  object
 8   sex            30161 non-null  object
 9   capitalgain    30161 non-null  int64
 10  capitalloss    30161 non-null  int64
 11  hoursperweek   30161 non-null  int64
 12  native         30161 non-null  object
 13  Salary         30161 non-null  object
dtypes: int64(5), object(9)
memory usage: 3.2+ MB
```

—----------describe—-----------------

data1.describe()

	age	educationno	capitalgain	capitalloss	hoursperweek
count	30161.000000	30161.000000	30161.000000	30161.000000	30161.000000
mean	38.438115	10.121316	1092.044064	88.302311	40.931269
std	13.134830	2.550037	7406.466611	404.121321	11.980182
min	17.000000	1.000000	0.000000	0.000000	1.000000
25%	28.000000	9.000000	0.000000	0.000000	40.000000
50%	37.000000	10.000000	0.000000	0.000000	40.000000

data2.info()

```
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   age            15060 non-null  int64
 1   workclass      15060 non-null  object
 2   education      15060 non-null  object
 3   educationno    15060 non-null  int64
 4   maritalstatus  15060 non-null  object
 5   occupation     15060 non-null  object
 6   relationship   15060 non-null  object
 7   race           15060 non-null  object
 8   sex            15060 non-null  object
 9   capitalgain    15060 non-null  int64
 10  capitalloss    15060 non-null  int64
 11  hoursperweek   15060 non-null  int64
 12  native         15060 non-null  object
 13  Salary         15060 non-null  object
dtypes: int64(5), object(9)
memory usage: 1.6+ MB
```

data2.describe()

	age	educationno	capitalgain	capitalloss	hoursperweek
count	15060.000000	15060.000000	15060.000000	15060.000000	15060.000000
mean	38.768327	10.112749	1120.301594	89.041899	40.951594
std	13.380676	2.558727	7703.181842	406.283245	12.062831
min	17.000000	1.000000	0.000000	0.000000	1.000000
25%	28.000000	9.000000	0.000000	0.000000	40.000000
50%	37.000000	10.000000	0.000000	0.000000	40.000000
75%	48.000000	13.000000	0.000000	0.000000	45.000000

data1.workclass.unique()

```
array([' State-gov', ' Self-emp-not-inc', ' Private', ' Federal-gov',
       ' Local-gov', ' Self-emp-inc', ' Without-pay'], dtype=object)
```

data1.workclass.value_counts()

```
 Private             22285
 Self-emp-not-inc     2499
 Local-gov            2067
 State-gov            1279
 Self-emp-inc         1074
 Federal-gov           943
 Without-pay            14
Name: workclass, dtype: int64
```

data1.occupation.unique()

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

data1.occupation.value_counts()

```
 Prof-specialty       4038
 Craft-repair         4030
 Exec-managerial      3992
 Adm-clerical         3721
 Sales                3584
 Other-service        3212
 Machine-op-inspct    1965
```

data1.native.unique()

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

data1.native.value_counts()

```
United-States                       27504
Mexico                                610
Philippines                           188
Germany                               128
Puerto-Rico                           109
Canada                                107
El-Salvador                           100
India                                 100
Cuba                                   92
England                                86
Jamaica                                80
South                                  71
Italy                                  68
China                                  68
```

numerical = [var for var in data1.columns if data1[var].dtypes!='0']
print('numerical values are',numerical)

x = data1.drop(['Salary'],axis=1)
y = data1['Salary']

—-----------traintest—----------------------
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
x_train.shape,x_test.shape,y_train.shape,y_test.shape

```
((21112, 13), (9049, 13), (21112,), (9049,)
```

x_train.dtypes

```
age               int64
workclass        object
education        object
educationno       int64
maritalstatus    object
occupation       object
relationship     object
race             object
sex              object
capitalgain       int64
capitalloss       int64
hoursperweek      int64
native           object
dtype: object
```

x_test.dtypes

```
age               int64
workclass        object
education        object
educationno       int64
maritalstatus    object
occupation       object
relationship     object
race             object
sex              object
capitalgain       int64
capitalloss       int64
```

categorical = [col for col in x_train.columns if x_train[col].dtypes=='0']
categorical

```
['workclass',
 'education',
 'maritalstatus',
 'occupation',
 'relationship',
 'race',
 'sex',
 'native']
```

numerical = [col for col in x_train.columns if x_train[col].dtypes=='0']
numerical

```
['age', 'educationno', 'capitalgain', 'capitalloss', 'hoursperweek']
```

x_train[categorical].isnull().mean()

```
workclass        0.0
education        0.0
maritalstatus    0.0
occupation       0.0
relationship     0.0
race             0.0
sex              0.0
native           0.0
dtype: float64
```

```
for col in categorical:
    if x_train[col].isnull().mean()>0:
        print(col,(x_train[col].isnull().mean()))
for df2 in [x_train,x_test]:
    df2['workclass'].fillna(x_train['workclass'].model()[0],inplace=True)
    df2['occupation'].fillna(x_train['occupation'].model()[0],inplace=True)
    df2['native'].fillna(x_train['native'].model()[0],inplace=True)
```

x_train[categorical].isnull().sum()

```
workclass        0
education        0
maritalstatus    0
occupation       0
relationship     0
race             0
sex              0
native           0
dtype: int64
```

x_test[categorical].isnull().sum()

```
workclass        0
education        0
maritalstatus    0
occupation       0
relationship     0
race             0
sex              0
native           0
dtype: int64
```

x_train.isnull().sum()

```
age              0
workclass        0
education        0
educationno      0
maritalstatus    0
occupation       0
relationship     0
race             0
sex              0
capitalgain      0
capitalloss      0
hoursperweek     0
native           0
dtype: int64
```

x_test.isnull().sum()

```
age              0
workclass        0
education        0
educationno      0
maritalstatus    0
occupation       0
relationship     0
race             0
sex              0
capitalgain      0
capitalloss      0
hoursperweek     0
native           0
dtype: int64
```

pip install category_encoders

```
Collecting category_encoders
  Downloading category_encoders-2.3.0-py2.py3-none-any.whl (82 kB)
Requirement already satisfied: patsy>=0.5.1 in c:\users\dell\anaconda3\lib\site-packages (from category_encoders) (0.5.1)
Requirement already satisfied: scikit-learn>=0.20.0 in c:\users\dell\anaconda3\lib\site-packages (from category_encoders) (0.2
4.1)
Requirement already satisfied: statsmodels>=0.9.0 in c:\users\dell\anaconda3\lib\site-packages (from category_encoders) (0.12.
2)
Requirement already satisfied: scipy>=1.0.0 in c:\users\dell\anaconda3\lib\site-packages (from category_encoders) (1.6.2)
Requirement already satisfied: pandas>=0.21.1 in c:\users\dell\anaconda3\lib\site-packages (from category_encoders) (1.2.4)
Requirement already satisfied: numpy>=1.14.0 in c:\users\dell\anaconda3\lib\site-packages (from category_encoders) (1.20.1)
Requirement already satisfied: pytz>=2017.3 in c:\users\dell\anaconda3\lib\site-packages (from pandas>=0.21.1->category_encoder
s) (2021.1)
Requirement already satisfied: python-dateutil>=2.7.3 in c:\users\dell\anaconda3\lib\site-packages (from pandas>=0.21.1->catego
ry_encoders) (2.8.1)
Requirement already satisfied: six in c:\users\dell\anaconda3\lib\site-packages (from patsy>=0.5.1->category_encoders) (1.15.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\dell\anaconda3\lib\site-packages (from scikit-learn>=0.20.0->ca
tegory_encoders) (2.1.0)
Requirement already satisfied: joblib>=0.11 in c:\users\dell\anaconda3\lib\site-packages (from scikit-learn>=0.20.0->category_e
ncoders) (1.0.1)
Installing collected packages: category-encoders
Successfully installed category-encoders-2.3.0
Note: you may need to restart the kernel to use updated packages.
```

```
import category_encoders as ce
encoder =
ce.OneHotEncoder(cols=['workclass','education','maritalstatus','occupation','relationship','race','s
ex','native'])
x_train = encoder.fit_transform(x_train)
x_test = encoder.fit_transform(x_test)
```

```
x_train.head()
```

	age	workclass_1	workclass_2	workclass_3	workclass_4	workclass_5	workclass_6	workclass_7	education_1	education_2
8166	54	1	0	0	0	0	0	0	1	0
7138	21	0	1	0	0	0	0	0	1	0
437	30	0	1	0	0	0	0	0	0	1
5436	42	0	1	0	0	0	0	0	0	1
6541	37	0	0	1	0	0	0	0	0	1

rows x 102 columns

```
x_train.shape
```

```
(21112, 102)
```

```
x_test.head()
```

	age	workclass_1	workclass_2	workclass_3	workclass_4	workclass_5	workclass_6	workclass_7
25338	21	0	1	0	0	0	0	0
18840	21	0	1	0	0	0	0	0
8391	56	0	1	0	0	0	0	0
18258	43	1	0	0	0	0	0	0

```
cols = x_train.columns

scaler = RobustScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.fit_transform(x_test)

x_train = pd.DataFrame(x_train,columns=[cols])

x_test = pd.DataFrame(x_test,columns=[cols])

x_train.head()
```

—----------gaussian—---------------------

```
gnb = GaussianNB()
gnb.fit(x_train,y_train)
```

```
GaussianNB()
```

ypredict = gnb.predict(x_test)
ypredict

```
array([' >50K', ' <=50K', ' <=50K', ..., ' >50K', ' <=50K', ' <=50K'],
      dtype='<U6')
```

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

```
Model accuracy score:{0:0.4f} 0.7515747596419494
```

ypredict_train = gnb.predict(x_train)
ypredict_train

```
array([' >50K', ' >50K', ' <=50K', ..., ' <=50K', ' <=50K', ' <=50K'],
      dtype='<U6')
```

print('Training - set accuracy score:{0:0.4f}',format(accuracy_score(y_train,ypredict_train)))

```
Training - set accuracy score:{0:0.4f} 0.7975085259568018
```

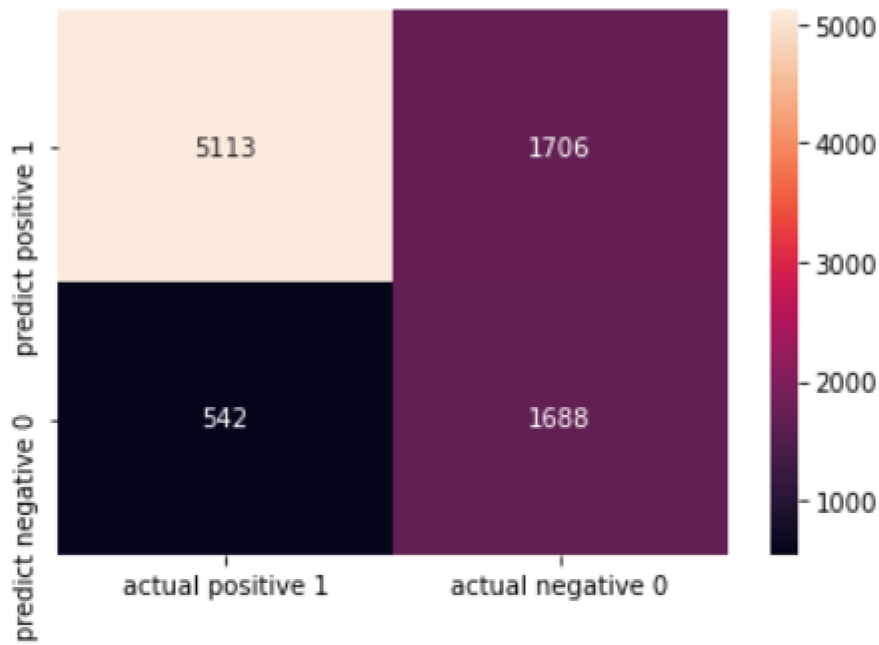
```
y_test.value_counts()
```

```
 <=50K    6819
 >50K     2230
Name: Salary, dtype: int64
```

cm = confusion_matrix(y_test,ypredict)
print('Confusion Matrix \n\n',cm)
print('\n True Positives(TP)=',cm[0,0])
print('\n True Negatives(TN)=',cm[1,1])
print('\n False Positives(FP)=',cm[0,1])
print('\n False Negatives(FN)=',cm[1,0])

```
[[5113 1706]
 [ 542 1688]]

True Positives(TP)= 5113

True Negatives(TN)= 1688

False Positives(FP)= 1706

False Negatives(FN)= 542
```

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')



print(classification_report(y_test,ypredict))

<=50K	0.90	0.75	0.82	6819
>50K	0.50	0.76	0.60	2230
accuracy			0.75	9049
macro avg	0.70	0.75	0.71	9049
weighted avg	0.80	0.75	0.77	9049

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

classification_accuracy = (TP+TN)/float(TP+TN+FP+FN)
print('classification accuracy:{0:0.4f}',format(classification_accuracy))

```
classification accuracy:{0:0.4f} 0.7515747596419494
```

classification_error = (FP+FN)/float(TP+TN+FP+FN)
print('classification error:{0:0.4f}',format(classification_error))

```
classification error:{0:0.4f} 0.2484252403580506
```

precision = TP/float(TP+FP)
print('precision:{0:0.4f}',format(precision))

```
precision:{0:0.4f} 0.749816688664027
```

recall = TP/float(TP+FN)
print('recall:{0:0.4f}',format(recall))

```
recall:{0:0.4f} 0.9041556145004421
```

true_positive_rate = TP/float(TP+FN)
print('true_positive_rate:{0:0.4f}',format(true_positive_rate))

```
true_positive_rate:{0:0.4f} 0.9041556145004421
```

false_positive_rate = FP/float(FP+TN)
print('false_positive_rate:{0:0.4f}',format(false_positive_rate))

```
false_positive_rate:{0:0.4f} 0.502651738361815
```

specificity = TN/float(FP+TN)
print('specificity:{0:0.4f}',format(specificity))

```
specificity:{0:0.4f} 0.497348261638185
```

ypredict_prob = gnb.predict_proba(x_test)[0:10]
ypredict_prob

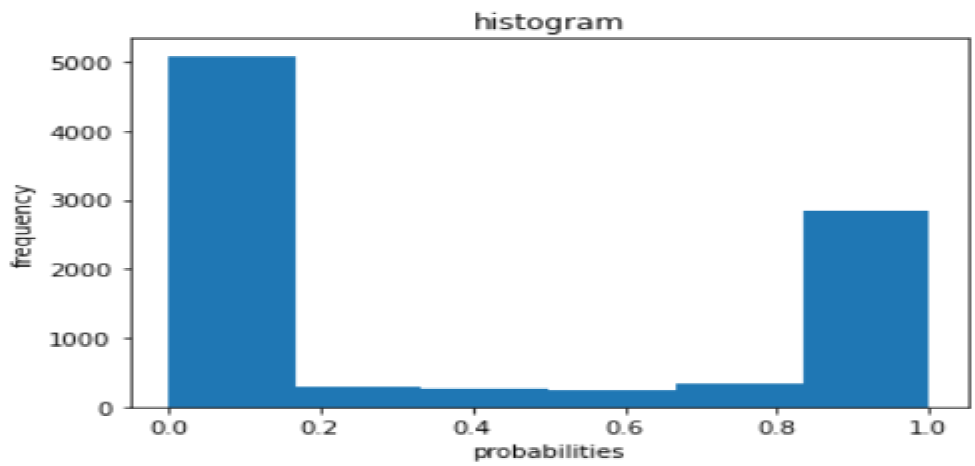
```
array([[5.95373694e-02, 9.40462631e-01],
       [9.99992594e-01, 7.40639066e-06],
       [9.99865551e-01, 1.34448847e-04],
       [9.99997881e-01, 2.11928102e-06],
       [9.97914068e-01, 2.08593214e-03],
       [9.99609179e-01, 3.90821247e-04],
       [2.23216260e-02, 9.77678374e-01],
       [1.47320254e-02, 9.85267975e-01],
       [1.16408255e-01, 8.83591745e-01],
       [9.99999197e-01, 8.02904032e-07]])
```

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

```
array([9.40462631e-01, 7.40639066e-06, 1.34448847e-04, 2.11928102e-06,
       2.08593214e-03, 3.90821247e-04, 9.77678374e-01, 9.85267975e-01,
       8.83591745e-01, 8.02904032e-07])
```

ypredict1 = gnb.predict_proba(x_test)[:,1]

plt.hist(ypredict1,bins=6)
plt.title('histogram')
plt.xlabel('probabilities')
plt.ylabel('frequency')



scores = cross_val_score(gnb,x_train,y_train,cv=6,scoring='accuracy')
print('cross validation scores:{}'.format(scores))

```
cross validation scores:[0.79681728 0.79227053 0.80591077 0.8076158  0.79562251 0.78538943]
```

print('avergae cross validation scores:{}'.format(scores.mean()))

```
avergae cross validation scores:0.7972710529477408
```