In [1]:
```
# import the libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [6]:
```
salary_train=pd.read_csv("C:\\Users\\nishi\\Desktop\\Assignments\\Naive_Bayes\\Sa
```

In [7]:
```
salary_train
```

Out[7]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	
0	39	State-gov	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	M
1	50	Self-emp-not-inc	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	M
2	38	Private	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	M
3	53	Private	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	M
4	28	Private	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Fem
...	...	...	...	...	...	...	...	...	
30156	27	Private	Assoc-acdm	12	Married-civ-spouse	Tech-support	Wife	White	Fem
30157	40	Private	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	White	M
30158	58	Private	HS-grad	9	Widowed	Adm-clerical	Unmarried	White	Fem
30159	22	Private	HS-grad	9	Never-married	Adm-clerical	Own-child	White	M
30160	52	Self-emp-inc	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	White	Fem

30161 rows × 14 columns

In [9]:
```
salary_test=pd.read_csv("C:\\Users\\nishi\\Desktop\\Assignments\\Naive_Bayes\\Sal
```

In [10]:
```
salary_test
```

Out[10]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	
0	25	Private	11th	7	Never-married	Machine-op-inspct	Own-child	Black	
1	38	Private	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	White	
2	28	Local-gov	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	White	
3	44	Private	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	Black	
4	34	Private	10th	6	Never-married	Other-service	Not-in-family	White	
...	...	...	...	...	...	...	...	...	
15055	33	Private	Bachelors	13	Never-married	Prof-specialty	Own-child	White	
15056	39	Private	Bachelors	13	Divorced	Prof-specialty	Not-in-family	White	Fe
15057	38	Private	Bachelors	13	Married-civ-spouse	Prof-specialty	Husband	White	
15058	44	Private	Bachelors	13	Divorced	Adm-clerical	Own-child	Asian-Pac-Islander	
15059	35	Self-emp-inc	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	

15060 rows × 14 columns

In [11]:
```
salary_train.columns
```

Out[11]:
```
Index(['age', 'workclass', 'education', 'educationno', 'maritalstatus',
       'occupation', 'relationship', 'race', 'sex', 'capitalgain',
       'capitalloss', 'hoursperweek', 'native', 'Salary'],
      dtype='object')
```

In [13]:
```
salary_test.columns
```

Out[13]:
```
Index(['age', 'workclass', 'education', 'educationno', 'maritalstatus',
       'occupation', 'relationship', 'race', 'sex', 'capitalgain',
       'capitalloss', 'hoursperweek', 'native', 'Salary'],
      dtype='object')
```

In [14]:
```
string_columns=['workclass','education','maritalstatus','occupation','relationshi
```

In [15]:
```
from sklearn import preprocessing
label_encoder=preprocessing.LabelEncoder()
for i in string_columns:
    salary_train[i]=label_encoder.fit_transform(salary_train[i])
    salary_test[i]=label_encoder.fit_transform(salary_test[i])
```

In [17]:
```
col_names=list(salary_train.columns)
train_X=salary_train[col_names[0:13]]
train_Y=salary_train[col_names[13]]
test_x=salary_test[col_names[0:13]]
test_y=salary_test[col_names[13]]
```

In [18]:
```
col_names=list(salary_train.columns)
train_X=salary_train[col_names[0:13]]
train_Y=salary_train[col_names[13]]
test_x=salary_test[col_names[0:13]]
test_y=salary_test[col_names[13]]
```

In [20]:
```

from sklearn.naive_bayes import GaussianNB
Gmodel=GaussianNB()
train_pred_gau=Gmodel.fit(train_X,train_Y).predict(train_X)
test_pred_gau=Gmodel.fit(train_X,train_Y).predict(test_x)
```

In [21]:
```
train_acc_gau=np.mean(train_pred_gau==train_Y)
train_acc_gau#0.795
```

Out[21]: 0.7953317197705646

In [22]:
```
test_acc_gau=np.mean(test_pred_gau==test_y)

test_acc_gau#0.794
```

Out[22]: 0.7946879150066402

In [24]:
```
from sklearn.naive_bayes import MultinomialNB
Mmodel=MultinomialNB()
train_pred_multi=Mmodel.fit(train_X,train_Y).predict(train_X)
test_pred_multi=Mmodel.fit(train_X,train_Y).predict(test_x)
```

In [25]:
```
train_acc_multi=np.mean(train_pred_multi==train_Y)
train_acc_multi
```

Out[25]: 0.7729186698053778

In [26]:
```
test_acc_multi=np.mean(test_pred_multi==test_y)
#0.772
test_acc_multi#0.774
```

Out[26]:
```
0.7749667994687915
```

In [28]:
```
salary_train=pd.read_csv("C:\\Users\\nishi\\Desktop\\Assignments\\Naive_Bayes\\Sa
salary_test=pd.read_csv("C:\\Users\\nishi\\Desktop\\Assignments\\Naive_Bayes\\Sal
salary_train.columns
salary_test.columns
string_columns=['workclass','education','maritalstatus','occupation','relationshi

from sklearn import preprocessing
label_encoder=preprocessing.LabelEncoder()
for i in string_columns:
    salary_train[i]=label_encoder.fit_transform(salary_train[i])
    salary_test[i]=label_encoder.fit_transform(salary_test[i])

col_names=list(salary_train.columns)
train_X=salary_train[col_names[0:13]]
train_Y=salary_train[col_names[13]]
test_x=salary_test[col_names[0:13]]
test_y=salary_test[col_names[13]]

######### Naive Bayes ##############

#Gaussian Naive Bayes

from sklearn.naive_bayes import GaussianNB
Gmodel=GaussianNB()
train_pred_gau=Gmodel.fit(train_X,train_Y).predict(train_X)
test_pred_gau=Gmodel.fit(train_X,train_Y).predict(test_x)

train_acc_gau=np.mean(train_pred_gau==train_Y)
test_acc_gau=np.mean(test_pred_gau==test_y)
train_acc_gau#0.795
test_acc_gau#0.794

#Multinomial Naive Bayes

from sklearn.naive_bayes import MultinomialNB
Mmodel=MultinomialNB()
train_pred_multi=Mmodel.fit(train_X,train_Y).predict(train_X)
test_pred_multi=Mmodel.fit(train_X,train_Y).predict(test_x)

train_acc_multi=np.mean(train_pred_multi==train_Y)
test_acc_multi=np.mean(test_pred_multi==test_y)
train_acc_multi#0.772
test_acc_multi#0.774
```

Out[28]:
```
0.7749667994687915
```