ML - Lab Assignment 4
Naive Bayes

[1]:

```
#preprocessing
#model develop
#model_Train
#confusion_matrix, classificatrion report
#save the model
#load the model
#test the model with the new data
```

[2]:

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

[3]:

```
df = pd.read_csv("D:\MIT ADT\Third Year - Sem 2\ML LAB\Assign 4\Company_Data.
 ↪csv")
```

[4]:

```
df.head()
```

[4]:

```
   Sales  CompPrice  Income  Advertising  Population  Price ShelveLoc  Age  \
0   9.50        138      73           11         276    120       Bad   42
1  11.22        111      48           16         260     83      Good   65
2  10.06        113      35           10         269     80    Medium   59
3   7.40        117     100            4         466     97    Medium   55
4   4.15        141      64            3         340    128       Bad   38

   Education Urban   US
0         17   Yes  Yes
1         10   Yes  Yes
2         12   Yes  Yes
3         14   Yes  Yes
4         13   Yes   No
```

[5]:

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

[5]:

```
array(['Yes', 'No'], dtype=object)
```

[6]:

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

```
[6]: array(['Yes', 'No'], dtype=object)
```

```
[7]: df['Urban']=df['Urban'].replace('Yes',1)
     df['Urban']=df['Urban'].replace('No',0)

     df['US']=df['US'].replace('Yes',1)
     df['US']=df['US'].replace('No',0)
```

```
[8]: from sklearn.preprocessing import LabelEncoder
     lbl_enconder = LabelEncoder()

     df['ShelveLoc']=lbl_enconder.fit_transform(df['ShelveLoc'])
```

```
[9]: df.head()
```

```
[9]:    Sales  CompPrice  Income  Advertising  Population  Price  ShelveLoc  Age  \
     0   9.50        138      73           11         276    120          0   42
     1  11.22        111      48           16         260     83          1   65
     2  10.06        113      35           10         269     80          2   59
     3   7.40        117     100            4         466     97          2   55
     4   4.15        141      64            3         340    128          0   38

        Education  Urban  US
     0         17      1   1
     1         10      1   1
     2         12      1   1
     3         14      1   1
     4         13      1   0
```

```
[10]: df.shape
```

```
[10]: (400, 11)
```

```
[11]: # Random Sampling
      US0=df[df['US']==0]
      US1=df[df['US']==1]

      print("US0: ", US0.shape)
      print("US1: ", US1.shape)

     US0:  (142, 11)
     US1:  (258, 11)
```

```
[12]: no_sample=US1.sample(n=142)
```

```
[13]: no_sample.shape
```

[13]:
```
(142, 11)
```

[14]:
```
sampled_df=pd.concat([no_sample,US0],axis=0)
```

[15]:
```
sampled_df.shape
```

[15]:
```
(284, 11)
```

[16]:
```
#outliers

def outliers(df, ft):
    Q1 = df[ft].quantile(0.25)
    Q3 = df[ft].quantile(0.75)
    IQR = Q3-Q1

    lb = Q1 - 1.5*IQR
    ub = Q3 + 1.5*IQR

    ls = df.index[(df[ft]<lb) | (df[ft]>ub)]

    return ls
```

[17]:
```
index_list = []
for feature in ['Sales', 'Advertising', 'Price', 'Age', 'Education', 'Urban']:
    index_list.extend(outliers(df,feature))
```

[18]:
```
index_list
```

[18]:
```
[316, 376, 42, 125, 165, 174, 367]
```

[19]:
```
def remove(df,ls):
    ls = sorted(set(ls))
    df = df.drop(ls)
    return df
```

[20]:
```
df_cleaned = remove(df,index_list)
```

[21]:
```
df_cleaned.shape
```

[21]:
```
(393, 11)
```

[22]:
```
X = df_cleaned.drop('US',axis=1)
y = df_cleaned['US']
```

[23]:
```
from sklearn.feature_selection import mutual_info_classif
# determine the mutual information
mutual_info = mutual_info_classif(X, y)
```

```
mutual_info
```

[23]:
```
array([0.03501297, 0.        , 0.        , 0.37553371, 0.        ,
       0.02475734, 0.        , 0.0367367 , 0.04202818, 0.01177197])
```

[24]:
```
mutual_info = pd.Series(mutual_info)
mutual_info.index = X.columns
mutual_info.sort_values(ascending=False)
```

[24]:
```
Advertising    0.375534
Education      0.042028
Age            0.036737
Sales          0.035013
Price          0.024757
Urban          0.011772
CompPrice      0.000000
Income         0.000000
Population     0.000000
ShelveLoc      0.000000
dtype: float64
```

[25]:
```
X = X.drop(['ShelveLoc','Population', 'CompPrice','Income'],axis=1)
```

[26]:
```
X.head()
```

[26]:
```
   Sales  Advertising  Price  Age  Education  Urban
0   9.50           11    120   42         17      1
1  11.22           16     83   65         10      1
2  10.06           10     80   59         12      1
3   7.40            4     97   55         14      1
4   4.15            3    128   38         13      1
```

[27]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.
 ↪2,random_state=0)
```

[28]:
```
print(X_train.shape, y_train.shape)
print(X_test.shape, y_test.shape)
```

```
(314, 6) (314,)
(79, 6) (79,)
```

[29]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

X_train_std=scaler.fit_transform(X_train)
X_test_std=scaler.transform(X_test)
```

[30]:

```
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
y_pred = gnb.fit(X_train_std, y_train).predict(X_test_std)
print("Number of mislabeled points out of a total %d points : %d" % (X_test_std.
 ↪shape[0], (y_test != y_pred).sum()))
```

```
Number of mislabeled points out of a total 79 points : 7
```

[31]:

```
gnb.score(X_train_std,y_train)
```

[31]:

```
0.8662420382165605
```

[32]:

```
gnb.score(X_test_std,y_test)
```

[32]:

```
0.9113924050632911
```

[33]:

```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.83      0.93      0.88        27
           1       0.96      0.90      0.93        52

    accuracy                           0.91        79
   macro avg       0.90      0.91      0.90        79
weighted avg       0.92      0.91      0.91        79
```

[ ]:

```

```