In [17]:
```
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import StandardScaler
```

In [27]:
```
df=pd.read_csv("f:/dataset/classification/online_shop.csv")
```

In [34]:
```
en=LabelEncoder()
df.Gender=en.fit_transform(df.Gender)
X=df.iloc[:,:-1].values
y=df.iloc[:,-1].values
df
```

Out[34]:

	Gender	Age	EstimatedSalary	Purchased
0	1	19	19000	0
1	1	35	20000	0
2	0	26	43000	0
3	0	27	57000	0
4	1	19	76000	0
...	...	...	...	...
395	0	46	41000	1
396	1	51	23000	1
397	0	50	20000	1
398	1	36	33000	0
399	0	49	36000	1

400 rows × 4 columns

In [23]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=1)
```

In [25]:
```
sc=StandardScaler()
X_train_new=sc.fit_transform(X_train)
X_test_new=sc.transform(X_test)
model=KNeighborsClassifier(n_neighbors=7)
model.fit(X_train_new,y_train)
pred_train=model.predict(X_train_new)
pred_test=model.predict(X_test_new)
print("Train Score:",accuracy_score(y_train,pred_train))
print("Test Score:",accuracy_score(y_test,pred_test))
```

```
Train Score: 0.9266666666666666
Test Score: 0.89
```

In [43]:
```
#new sample
import numpy as np
g=input("enter gender['male','female']:").capitalize()
if(g=="Male" or g=="Female"):
    g=en.transform(np.array([g]))
    a=float(input("enter age:"))
    s=float(input("enter sal:"))
    sample=np.append(g,[a,s])
```

```
    sample=sc.transform([sample])
    print(model.predict(sample))
else:
    print("invalid")
```

```
[1]
```

In [32]:
```
df
```

Out[32]:

	Gender	Age	EstimatedSalary	Purchased
0	Male	19	19000	0
1	Male	35	20000	0
2	Female	26	43000	0
3	Female	27	57000	0
4	Male	19	76000	0
...	...	...	...	...
395	Female	46	41000	1
396	Male	51	23000	1
397	Female	50	20000	1
398	Male	36	33000	0
399	Female	49	36000	1

400 rows × 4 columns

In [44]:
```
from sklearn.datasets import load_wine
```

In [45]:
```
wine=load_wine()
print(type(wine)) #bunch object(dict like)
```

```
<class 'sklearn.utils._bunch.Bunch'>
```

In [46]:
```
wine.keys()
```

Out[46]:
```
dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names'])
```

In [48]:
```
print(wine['feature_names'])
print(wine.feature_names)
```

```
['alcohol', 'malic_acid', 'ash', 'alcalinity_of_ash', 'magnesium', 'total_phenols', 'fla
vanoids', 'nonflavanoid_phenols', 'proanthocyanins', 'color_intensity', 'hue', 'od280/od
315_of_diluted_wines', 'proline']
['alcohol', 'malic_acid', 'ash', 'alcalinity_of_ash', 'magnesium', 'total_phenols', 'fla
vanoids', 'nonflavanoid_phenols', 'proanthocyanins', 'color_intensity', 'hue', 'od280/od
315_of_diluted_wines', 'proline']
```

In [51]:
```
X=wine.data
y=wine.target
```

In [52]:
```
print(wine.target_names)
```

```
['class_0' 'class_1' 'class_2']
```

In [53]:
```
print(wine.DESCR)
```

```
.. _wine_dataset:
```

Wine recognition dataset

Data Set Characteristics:

    :Number of Instances: 178
    :Number of Attributes: 13 numeric, predictive attributes and the class
    :Attribute Information:
                - Alcohol
                - Malic acid
                - Ash
                - Alcalinity of ash
                - Magnesium
                - Total phenols
                - Flavanoids
                - Nonflavanoid phenols
                - Proanthocyanins
                - Color intensity
                - Hue
                - OD280/OD315 of diluted wines
                - Proline

    - class:
            - class_0
            - class_1
            - class_2

    :Summary Statistics:

    ============================== ==== ===== ======= =====
                                    Min   Max   Mean   SD
    ============================== ==== ===== ======= =====
    Alcohol: 11.0 14.8   13.0   0.8
    Malic Acid: 0.74 5.80   2.34 1.12
    Ash: 1.36 3.23   2.36 0.27
    Alcalinity of Ash: 10.6 30.0   19.5   3.3
    Magnesium: 70.0 162.0   99.7 14.3
    Total Phenols: 0.98 3.88   2.29 0.63
    Flavanoids: 0.34 5.08   2.03 1.00
    Nonflavanoid Phenols: 0.13 0.66   0.36 0.12
    Proanthocyanins: 0.41 3.58   1.59 0.57
    Colour Intensity: 1.3 13.0   5.1   2.3
    Hue: 0.48 1.71   0.96 0.23
    OD280/OD315 of diluted wines: 1.27 4.00   2.61 0.71
    Proline: 278 1680   746   315
    ============================== ==== ===== ======= =====

    :Missing Attribute Values: None
    :Class Distribution: class_0 (59), class_1 (71), class_2 (48)
    :Creator: R.A. Fisher
    :Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)
    :Date: July, 1988

This is a copy of UCI ML Wine recognition datasets.
https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data

The data is the results of a chemical analysis of wines grown in the same
region in Italy by three different cultivators. There are thirteen different
measurements taken for different constituents found in the three types of
wine.

Original Owners:

Forina, M. et al, PARVUS -
An Extendible Package for Data Exploration, Classification and Correlation.

```
Institute of Pharmaceutical and Food Analysis and Technologies,
Via Brigata Salerno, 16147 Genoa, Italy.

Citation:

Lichman, M. (2013). UCI Machine Learning Repository
[https://archive.ics.uci.edu/ml]. Irvine, CA: University of California,
School of Information and Computer Science.

.. topic:: References

  (1) S. Aeberhard, D. Coomans and O. de Vel,
  Comparison of Classifiers in High Dimensional Settings,
  Tech. Rep. no. 92-02, (1992), Dept. of Computer Science and Dept. of
  Mathematics and Statistics, James Cook University of North Queensland.
  (Also submitted to Technometrics).

  The data was used with many others for comparing various
  classifiers. The classes are separable, though only RDA
  has achieved 100% correct classification.
  (RDA : 100%, QDA 99.4%, LDA 98.9%, 1NN 96.1% (z-transformed data))
  (All results using the leave-one-out technique)

  (2) S. Aeberhard, D. Coomans and O. de Vel,
  "THE CLASSIFICATION PERFORMANCE OF RDA"
  Tech. Rep. no. 92-01, (1992), Dept. of Computer Science and Dept. of
  Mathematics and Statistics, James Cook University of North Queensland.
  (Also submitted to Journal of Chemometrics).
```

In [57]:
```
X.shape
```

Out[57]:
```
(178, 13)
```

In [58]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=1)
model=KNeighborsClassifier()
model.fit(X_train,y_train)
pred_train=model.predict(X_train)
pred_test=model.predict(X_test)
print("Train Score:",accuracy_score(y_train,pred_train))
print("Test Score:",accuracy_score(y_test,pred_test))
```

```
Train Score: 0.8270676691729323
Test Score: 0.6888888888888889
```

In [59]:
```
sc=StandardScaler()
X_train_new=sc.fit_transform(X_train)
X_test_new=sc.transform(X_test)
model=KNeighborsClassifier()
model.fit(X_train_new,y_train)
pred_train=model.predict(X_train_new)
pred_test=model.predict(X_test_new)
print("Train Score:",accuracy_score(y_train,pred_train))
print("Test Score:",accuracy_score(y_test,pred_test))
```

```
Train Score: 0.9849624060150376
Test Score: 0.9777777777777777
```

In [60]:
```
from sklearn.feature_selection import f_classif
```

In [62]:
```
fvalue,pvalue=f_classif(X,y)
```

In [63]:
```
fvalue
```

```
array([135.07762424,  36.94342496,  13.3129012 ,  35.77163741,
```

Out[63]:
```
        12.42958434,  93.73300962, 233.92587268,  27.57541715,
        30.27138317, 120.66401844, 101.31679539, 189.97232058,
       207.9203739 ])
```

In [64]:
```
print(wine.feature_names)
```

```
['alcohol', 'malic_acid', 'ash', 'alcalinity_of_ash', 'magnesium', 'total_phenols', 'fla
vanoids', 'nonflavanoid_phenols', 'proanthocyanins', 'color_intensity', 'hue', 'od280/od
315_of_diluted_wines', 'proline']
```

In [73]:
```
X1=X[:,[0,5,6,9,10,11,12]]
```

In [74]:
```
X1.shape
```

Out[74]:
```
(178, 7)
```

In [75]:
```
X_train,X_test,y_train,y_test=train_test_split(X1,y,random_state=1)
sc=StandardScaler()
X_train_new=sc.fit_transform(X_train)
X_test_new=sc.transform(X_test)
model=KNeighborsClassifier()
model.fit(X_train_new,y_train)
pred_train=model.predict(X_train_new)
pred_test=model.predict(X_test_new)
print("Train Score:",accuracy_score(y_train,pred_train))
print("Test Score:",accuracy_score(y_test,pred_test))
```

```
Train Score: 0.9849624060150376
Test Score: 0.9777777777777777
```

In [ ]:
```

```