In [1]:
```
# import all required packages

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

In [2]:
```
# read the data
df = pd.read_csv('Wine.csv')
# print(df.head())
# print(len(df.columns))
```

In [3]:
```
# decide x and y
x = df.drop('Customer_Segment', axis=1)
y = df['Customer_Segment']
```

LDA

In [4]:
```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

lda = LinearDiscriminantAnalysis(n_components=2)

# extract features from train
x = lda.fit_transform(x, y)
# print(x)
```

In [5]:
```
# split the data into train and test
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, train_size=0.8)
```

data cleansing process

In [6]:
```
# look out the missing data
# use the correct data type
# add new columns if needed
# drop existing columns if needed

# scale the features
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

# scale the x_train features
x_train = scaler.fit_transform(x_train)

# scale the x_test features
x_test = scaler.fit_transform(x_test)
```

Logistic Regression

In [7]:
```
from sklearn.linear_model import LogisticRegressionCV

# create a model
model = LogisticRegressionCV(max_iter=1000)

# train the model
model.fit(x_train, y_train)

# predict the values from x_test
y_prediction = model.predict(x_test)

# evaluation
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score

print(confusion_matrix(y_test, y_prediction))
```

```
[[ 9  0  0]
 [ 0 14  0]
 [ 0  0 13]]
```

In [8]:
```
print(classification_report(y_test, y_prediction))
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00         9
           2       1.00      1.00      1.00        14
           3       1.00      1.00      1.00        13

    accuracy                           1.00        36
   macro avg       1.00      1.00      1.00        36
weighted avg       1.00      1.00      1.00        36
```

In [9]:
```
print(accuracy_score(y_test, y_prediction))
```

```
1.0
```

In [ ]:

In [ ]: