Importing the libraries

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

Importing the dataset

```
df = pd.read_csv('https://github.com/YBI-Foundation/Dataset/raw/main/Fruits.csv')
```

Get Information of Dataframe

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 59 entries, 0 to 58
Data columns (total 6 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Fruit Category      59 non-null     int64
 1   Fruit Name          59 non-null     object
 2   Fruit Weight        59 non-null     int64
 3   Fruit Width         59 non-null     float64
 4   Fruit Length        59 non-null     float64
 5   Fruit Colour Score  59 non-null     float64
dtypes: float64(3), int64(2), object(1)
memory usage: 2.9+ KB
```

```
df.head()
```

	Fruit Category	Fruit Name	Fruit Weight	Fruit Width	Fruit Length	Fruit Colour
0	1	Apple	192	8.4	7.3	
1	1	Apple	180	8.0	6.8	
2	1	Apple	176	7.4	7.2	
3	1	Apple	178	7.1	7.8	
4	1	Apple	172	7.4	7.0	

```
df.isnull().sum()
```

```
Fruit Category        0
Fruit Name            0
Fruit Weight          0
Fruit Width           0
Fruit Length          0
Fruit Colour Score    0
dtype: int64
```

```
df.columns
```

```
Index(['Fruit Category', 'Fruit Name', 'Fruit Weight', 'Fruit Width',
       'Fruit Length', 'Fruit Colour Score'],
      dtype='object')
```

Get Unique Values in y variable

```
df['Fruit Category'].value_counts()
```

```
2    24
1    19
3    16
Name: Fruit Category, dtype: int64
```

```
df.groupby('Fruit Category').mean()
```

Setting X and y

```
X= df[['Fruit Weight', 'Fruit Width',
       'Fruit Length', 'Fruit Colour Score']]
```

```
y=df['Fruit Category']
```

```
X.shape, y.shape
```

```
((59, 4), (59,))
```

Splitting the dataset into the Training set and Test set

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, stratify=y,
```

```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

```
((41, 4), (18, 4), (41,), (18,))
```

Logistic Regression

```
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression(max_iter=500)
lr.fit(X_train,y_train)
```

```
LogisticRegression(max_iter=500)
```

Model Prediction

```
y_pred=lr.predict(X_test)
```

```
y_pred.shape
```

```
(18,)
```

Model Evaluation

Get probability of each predicted class

```
lr.predict_proba(X_test)
```

```
array([[3.97531652e-01, 6.01975007e-01, 4.93340951e-04],
       [6.40972747e-01, 3.58153192e-01, 8.74061564e-04],
       [5.99059554e-01, 3.98219888e-01, 2.72055750e-03],
       [5.08218422e-01, 4.91126505e-01, 6.55072619e-04],
       [2.28091620e-03, 2.48572924e-03, 9.95233355e-01],
       [3.43865475e-01, 6.55819262e-01, 3.15262913e-04],
       [2.01059195e-03, 1.86389543e-03, 9.96125513e-01],
       [5.10734580e-01, 4.89077807e-01, 1.87612639e-04],
       [3.86558629e-01, 6.13416900e-01, 2.44715384e-05],
       [4.64441327e-01, 5.35007547e-01, 5.51125754e-04],
           [2.91798339e-03, 3.30343472e-03, 9.93778582e-01],
           [3.44370280e-01, 6.53457074e-01, 2.17264597e-03],
           [4.42913215e-01, 5.56751653e-01, 3.35132571e-04],
           [3.51031882e-03, 4.38853246e-04, 9.96050828e-01],
           [2.83625877e-01, 7.16254466e-01, 1.19657564e-04],
           [6.19207072e-01, 3.80074729e-01, 7.18198770e-04],
           [2.53288399e-01, 6.95010914e-01, 5.17006872e-02],
           [5.55774841e-03, 5.32253484e-03, 9.89119717e-01]])
```

```python
from sklearn.metrics import confusion_matrix, classification_report
```

```python
confusion_matrix(y_test,y_pred)
```

```
array([[4, 2, 0],
       [1, 6, 0],
       [0, 0, 5]])
```

```python
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           1       0.80      0.67      0.73         6
           2       0.75      0.86      0.80         7
           3       1.00      1.00      1.00         5

    accuracy                           0.83        18
   macro avg       0.85      0.84      0.84        18
weighted avg       0.84      0.83      0.83        18
```