Steps for Classification Model with Sklearn

import library

In [1]:

```
import numpy as np
import pandas as pd
```

import CSV as DataFrame

In [2]:

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

Get the first five rows of Dataframe

In [3]:

```
df.head()
```

Out[3]:

	Fruit Category	Fruit Name	Fruit Weight	Fruit Width	Fruit Length	Fruit Colour Score
0	1	Apple	192	8.4	7.3	0.55
1	1	Apple	180	8.0	6.8	0.59
2	1	Apple	176	7.4	7.2	0.60
3	1	Apple	178	7.1	7.8	0.92
4	1	Apple	172	7.4	7.0	0.89

Get Information of Dataframe

In [4]:

```
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
```

Get the Summary Statistics

In [5]:

```
df.describe()
```

Out[5]:

	Fruit Category	Fruit Weight	Fruit Width	Fruit Length	Fruit Colour Score
count	59.000000	59.000000	59.000000	59.000000	59.000000
mean	1.949153	141.796610	7.105085	7.693220	0.762881
std	0.775125	67.335951	0.816938	1.361017	0.076857
min	1.000000	58.000000	5.800000	4.000000	0.550000
25%	1.000000	82.000000	6.600000	7.200000	0.720000
50%	2.000000	154.000000	7.200000	7.600000	0.750000
75%	3.000000	167.000000	7.500000	8.200000	0.810000
max	3.000000	362.000000	9.600000	10.500000	0.930000

Get Shape of Dataframe

In [6]:

```
df.shape
```

Out[6]:

```
(59, 6)
```

Get Columns Names

In [7]:

```
df.columns
```

Out[7]:

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

Get Unique Values(class or Label) in y Variable

In [8]:

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

Out[8]:

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

In [9]:

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

Out[9]:

Fruit Category	Fruit Weight	Fruit Width	Fruit Length	Fruit Colour Score
1	165.052632	7.457895	7.342105	0.783684
2	170.333333	7.220833	7.195833	0.776250
3	71.375000	6.512500	8.856250	0.718125

Define y(dependent or label or target variable) and X(independent or features or attribute Variable)

In [10]:

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

In [11]:

```
y.shape
```

Out[11]:

```
(59,)
```

In [12]:

```
y
```

Out[12]:

```
0     1
1     1
2     1
3     1
4     1
5     1
6     1
7     1
8     1
9     1
10    1
11    1
12    1
13    1
14    1
15    1
16    1
17    1
18    1
19    2
20    2
21    2
22    2
23    2
24    2
25    2
26    2
27    2
28    2
29    2
30    2
31    2
32    2
33    2
34    2
35    2
36    2
37    2
38    2
39    2
40    2
41    2
42    2
43    3
44    3
45    3
46    3
47    3
48    3
49    3
50    3
51    3
52    3
53    3
54    3
```

```
55    3
56    3
57    3
58    3
Name: Fruit Category, dtype: int64
```

In [13]:

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

In [14]:

```
X = df.drop(['Fruit Category','Fruit Name'], axis=1)
```

In [15]:

```
X.shape
```

Out[15]:

```
(59, 4)
```

In [16]:

```
X
```

Out[16]:

	Fruit Weight	Fruit Width	Fruit Length	Fruit Colour Score
0	192	8.4	7.3	0.55
1	180	8.0	6.8	0.59
2	176	7.4	7.2	0.60
3	178	7.1	7.8	0.92
4	172	7.4	7.0	0.89
5	166	6.9	7.3	0.93
6	172	7.1	7.6	0.92
7	154	7.0	7.1	0.88
8	164	7.3	7.7	0.70
9	152	7.6	7.3	0.69
10	156	7.7	7.1	0.69
11	156	7.6	7.5	0.67
12	168	7.5	7.6	0.73
13	162	7.5	7.1	0.83
14	162	7.4	7.2	0.85
15	160	7.5	7.5	0.86
16	156	7.4	7.4	0.84
17	140	7.3	7.1	0.87
18	170	7.6	7.9	0.88
19	86	6.2	4.7	0.80
20	84	6.0	4.6	0.79
21	80	5.8	4.3	0.77
22	80	5.9	4.3	0.81
23	76	5.8	4.0	0.81
24	342	9.0	9.4	0.75
25	356	9.2	9.2	0.75
26	362	9.6	9.2	0.74
27	204	7.5	9.2	0.77
28	140	6.7	7.1	0.72
29	160	7.0	7.4	0.81
30	158	7.1	7.5	0.79
31	210	7.8	8.0	0.82
32	164	7.2	7.0	0.80
33	190	7.5	8.1	0.74

	Fruit Weight	Fruit Width	Fruit Length	Fruit Colour Score
34	142	7.6	7.8	0.75
35	150	7.1	7.9	0.75
36	160	7.1	7.6	0.76
37	154	7.3	7.3	0.79
38	158	7.2	7.8	0.77
39	144	6.8	7.4	0.75
40	154	7.1	7.5	0.78
41	180	7.6	8.2	0.79
42	154	7.2	7.2	0.82
43	97	7.2	10.3	0.70
44	70	7.3	10.5	0.72
45	93	7.2	9.2	0.72
46	80	7.3	10.2	0.71
47	98	7.3	9.7	0.72
48	87	7.3	10.1	0.72
49	66	5.8	8.7	0.73
50	65	6.0	8.2	0.71
51	58	6.0	7.5	0.72
52	59	5.9	8.0	0.72
53	60	6.0	8.4	0.74
54	58	6.1	8.5	0.71
55	58	6.3	7.7	0.72
56	58	5.9	8.1	0.73
57	76	6.5	8.5	0.72
58	59	6.1	8.1	0.70

Get Train Test Split

In [17]:

```
from sklearn.model_selection import train_test_split
```

In [18]:

```
X_train,X_test,y_train,y_test = train_test_split(X,y, test_size = 0.3, random_state=2527)
```

In [19]:

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

Out[19]:

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

Get Model Train

In [20]:

```
from sklearn.linear_model import LogisticRegression
```

In [21]:

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

In [22]:

```
model.fit(X_train,y_train)
```

Out[22]:

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

Get Model prediction

In [23]:

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

In [24]:

```
y_pred.shape
```

Out[24]:

```
(18,)
```

In [25]:

```
y_pred
```

Out[25]:

```
array([1, 2, 2, 3, 2, 2, 3, 2, 1, 2, 3, 2, 3, 2, 2, 2, 2, 3], dtype=int64)
```

Get Probability of Each Predicted Class

In [26]:

```
model.predict_proba(X_test)
```

Out[26]:

```
array([[7.23681943e-01, 2.76313802e-01, 4.25561303e-06],
       [1.53278000e-01, 8.46722000e-01, 6.27408228e-14],
       [2.78026327e-01, 7.21905037e-01, 6.86359051e-05],
       [5.39138800e-03, 7.67942375e-04, 9.93840670e-01],
       [2.48086478e-01, 7.51661185e-01, 2.52337034e-04],
       [3.39645937e-01, 6.60159459e-01, 1.94603589e-04],
       [5.42029491e-03, 9.77024714e-03, 9.84809458e-01],
       [4.39377245e-01, 5.60243234e-01, 3.79521145e-04],
       [5.75887906e-01, 4.23535002e-01, 5.77092478e-04],
       [4.45714562e-01, 5.54163571e-01, 1.21866709e-04],
       [7.23751518e-02, 2.04706462e-02, 9.07154202e-01],
       [1.82769229e-01, 8.17201254e-01, 2.95171210e-05],
       [2.89539753e-03, 6.40223274e-03, 9.90702370e-01],
       [3.24418625e-01, 6.75553718e-01, 2.76560565e-05],
       [1.60776110e-01, 8.39163268e-01, 6.06211689e-05],
       [2.67374666e-01, 7.32377243e-01, 2.48091176e-04],
       [2.99848194e-01, 6.99675536e-01, 4.76269559e-04],
       [2.81049181e-02, 8.02066540e-03, 9.63874417e-01]])
```

Get Model Evaluation

In [27]:

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

In [28]:

```
print(confusion_matrix(y_test,y_pred))
```

```
[[2 7 0]
 [0 4 0]
 [0 0 5]]
```

In [29]:

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

```
              precision    recall  f1-score   support

           1       1.00      0.22      0.36         9
           2       0.36      1.00      0.53         4
           3       1.00      1.00      1.00         5

    accuracy                           0.61        18
   macro avg       0.79      0.74      0.63        18
weighted avg       0.86      0.61      0.58        18
```

Get Future Predictions

lets select a random sample from existing dataset as new value

Steps to follow
1.Extract a random row using sample function
2.Separate X and y
3.Predict

In [30]:

```
df_new = df.sample(1)
```

In [31]:

```
df_new
```

Out[31]:

	Fruit Category	Fruit Name	Fruit Weight	Fruit Width	Fruit Length	Fruit Colour Score
51	3	Lemon	58	6.0	7.5	0.72

In [32]:

```
X_new = df_new[['Fruit Weight','Fruit Width','Fruit Length','Fruit Colour Score']]
```

In [34]:

```
X_new.shape
```

Out[34]:

```
(1, 4)
```

In [36]:

```
y_pred_new = model.predict(X_new)
```

In [37]:

```
y_pred_new
```

Out[37]:

```
array([3], dtype=int64)
```

In [38]:

```
model.predict_proba(X_new)
```

Out[38]:

```
array([[0.00542029, 0.00977025, 0.98480946]])
```

In [ ]:

```

```