

```
In [2]: import pandas as pd
```

```
In [3]: df = pd.read_csv("shirts.csv")  
df.head()
```

Out[3]:

	Height	Weight	Size
0	158	58	M
1	158	59	M
2	158	63	M
3	160	59	M
4	160	60	M

```
In [4]: df.isnull().sum()
```

Out[4]: Height 0  
Weight 0  
Size 0  
dtype: int64

```
In [5]: df.dtypes
```

Out[5]: Height int64  
Weight int64  
Size object  
dtype: object

```
In [6]: x = df[["Height", "Weight"]]  
x.head()
```

Out[6]:

	Height	Weight
0	158	58
1	158	59
2	158	63
3	160	59
4	160	60

```
In [7]: # target  
y = df["Size"]  
y.head()
```

Out[7]: 0 M  
1 M  
2 M  
3 M  
4 M  
Name: Size, dtype: object

```
In [9]: y1 = pd.get_dummies(df["Size"])
y1
```

Out[9]:

	L	M
0	0	1
1	0	1
2	0	1
3	0	1
4	0	1
5	0	1
6	0	1
7	1	0
8	1	0
9	1	0
10	1	0
11	1	0
12	1	0
13	1	0
14	1	0
15	1	0
16	1	0
17	1	0

```
In [8]: from sklearn.neighbors import KNeighborsClassifier
```

```
In [12]: knn = KNeighborsClassifier()
```

```
In [13]: knn.fit(x,y1["L"])
```

```
Out[13]: KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                               metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                               weights='uniform')
```

```
In [14]: # test the model
y_pred = knn.predict(x)
y_pred
```

```
Out[14]: array([0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1], dtype=uint8)
```

```
In [7]: from sklearn.metrics import accuracy_score, confusion_matrix
```



```
Out[16]: array([[5.1, 3.5, 1.4, 0.2],
 [4.9, 3. , 1.4, 0.2],
 [4.7, 3.2, 1.3, 0.2],
 [4.6, 3.1, 1.5, 0.2],
 [5. , 3.6, 1.4, 0.2],
 [5.4, 3.9, 1.7, 0.4],
 [4.6, 3.4, 1.4, 0.3],
 [5. , 3.4, 1.5, 0.2],
 [4.4, 2.9, 1.4, 0.2],
 [4.9, 3.1, 1.5, 0.1],
 [5.4, 3.7, 1.5, 0.2],
 [4.8, 3.4, 1.6, 0.2],
 [4.8, 3. , 1.4, 0.1],
 [4.3, 3. , 1.1, 0.1],
 [5.8, 4. , 1.2, 0.2],
 [5.7, 4.4, 1.5, 0.4],
 [5.4, 3.9, 1.3, 0.4],
 [5.1, 3.5, 1.4, 0.3],
 [5.7, 3.8, 1.7, 0.3],
 [5.1, 3.6, 1.5, 0.2],
 [5.2, 3.7, 1.6, 0.3],
 [5.3, 3.8, 1.7, 0.4],
 [5.4, 3.9, 1.8, 0.5],
 [5.5, 4.0, 1.9, 0.6],
 [5.6, 4.1, 2.0, 0.7],
 [5.7, 4.2, 2.1, 0.8],
 [5.8, 4.3, 2.2, 0.9],
 [5.9, 4.4, 2.3, 1.0],
 [6.0, 4.5, 2.4, 1.1],
 [6.1, 4.6, 2.5, 1.2],
 [6.2, 4.7, 2.6, 1.3],
 [6.3, 4.8, 2.7, 1.4],
 [6.4, 4.9, 2.8, 1.5],
 [6.5, 5.0, 2.9, 1.6],
 [6.6, 5.1, 3.0, 1.7],
 [6.7, 5.2, 3.1, 1.8],
 [6.8, 5.3, 3.2, 1.9],
 [6.9, 5.4, 3.3, 2.0],
 [7.0, 5.5, 3.4, 2.1],
 [7.1, 5.6, 3.5, 2.2],
 [7.2, 5.7, 3.6, 2.3],
 [7.3, 5.8, 3.7, 2.4],
 [7.4, 5.9, 3.8, 2.5],
 [7.5, 6.0, 3.9, 2.6],
 [7.6, 6.1, 4.0, 2.7],
 [7.7, 6.2, 4.1, 2.8],
 [7.8, 6.3, 4.2, 2.9],
 [7.9, 6.4, 4.3, 3.0],
 [8.0, 6.5, 4.4, 3.1],
 [8.1, 6.6, 4.5, 3.2],
 [8.2, 6.7, 4.6, 3.3],
 [8.3, 6.8, 4.7, 3.4],
 [8.4, 6.9, 4.8, 3.5],
 [8.5, 7.0, 4.9, 3.6],
 [8.6, 7.1, 5.0, 3.7],
 [8.7, 7.2, 5.1, 3.8],
 [8.8, 7.3, 5.2, 3.9],
 [8.9, 7.4, 5.3, 4.0],
 [9.0, 7.5, 5.4, 4.1],
 [9.1, 7.6, 5.5, 4.2],
 [9.2, 7.7, 5.6, 4.3],
 [9.3, 7.8, 5.7, 4.4],
 [9.4, 7.9, 5.8, 4.5],
 [9.5, 8.0, 5.9, 4.6],
 [9.6, 8.1, 6.0, 4.7],
 [9.7, 8.2, 6.1, 4.8],
 [9.8, 8.3, 6.2, 4.9],
 [9.9, 8.4, 6.3, 5.0],
 [10.0, 8.5, 6.4, 5.1],
 [10.1, 8.6, 6.5, 5.2],
 [10.2, 8.7, 6.6, 5.3],
 [10.3, 8.8, 6.7, 5.4],
 [10.4, 8.9, 6.8, 5.5],
 [10.5, 9.0, 6.9, 5.6],
 [10.6, 9.1, 7.0, 5.7],
 [10.7, 9.2, 7.1, 5.8],
 [10.8, 9.3, 7.2, 5.9],
 [10.9, 9.4, 7.3, 6.0],
 [11.0, 9.5, 7.4, 6.1],
 [11.1, 9.6, 7.5, 6.2],
 [11.2, 9.7, 7.6, 6.3],
 [11.3, 9.8, 7.7, 6.4],
 [11.4, 9.9, 7.8, 6.5],
 [11.5, 10.0, 7.9, 6.6],
 [11.6, 10.1, 8.0, 6.7],
 [11.7, 10.2, 8.1, 6.8],
 [11.8, 10.3, 8.2, 6.9],
 [11.9, 10.4, 8.3, 7.0],
 [12.0, 10.5, 8.4, 7.1],
 [12.1, 10.6, 8.5, 7.2],
 [12.2, 10.7, 8.6, 7.3],
 [12.3, 10.8, 8.7, 7.4],
 [12.4, 10.9, 8.8, 7.5],
 [12.5, 11.0, 8.9, 7.6],
 [12.6, 11.1, 9.0, 7.7],
 [12.7, 11.2, 9.1, 7.8],
 [12.8, 11.3, 9.2, 7.9],
 [12.9, 11.4, 9.3, 8.0],
 [13.0, 11.5, 9.4, 8.1],
 [13.1, 11.6, 9.5, 8.2],
 [13.2, 11.7, 9.6, 8.3],
 [13.3, 11.8, 9.7, 8.4],
 [13.4, 11.9, 9.8, 8.5],
 [13.5, 12.0, 9.9, 8.6],
 [13.6, 12.1, 10.0, 8.7],
 [13.7, 12.2, 10.1, 8.8],
 [13.8, 12.3, 10.2, 8.9],
 [13.9, 12.4, 10.3, 9.0],
 [14.0, 12.5, 10.4, 9.1],
 [14.1, 12.6, 10.5, 9.2],
 [14.2, 12.7, 10.6, 9.3],
 [14.3, 12.8, 10.7, 9.4],
 [14.4, 12.9, 10.8, 9.5],
 [14.5, 13.0, 10.9, 9.6],
 [14.6, 13.1, 11.0, 9.7],
 [14.7, 13.2, 11.1, 9.8],
 [14.8, 13.3, 11.2, 9.9],
 [14.9, 13.4, 11.3, 10.0],
 [15.0, 13.5, 11.4, 10.1],
 [15.1, 13.6, 11.5, 10.2],
 [15.2, 13.7, 11.6, 10.3],
 [15.3, 13.8, 11.7, 10.4],
 [15.4, 13.9, 11.8, 10.5],
 [15.5, 14.0, 11.9, 10.6],
 [15.6, 14.1, 12.0, 10.7],
 [15.7, 14.2, 12.1, 10.8],
 [15.8, 14.3, 12.2, 10.9],
 [15.9, 14.4, 12.3, 11.0],
 [16.0, 14.5, 12.4, 11.1],
 [16.1, 14.6, 12.5, 11.2],
 [16.2, 14.7, 12.6, 11.3],
 [16.3, 14.8, 12.7, 11.4],
 [16.4, 14.9, 12.8, 11.5],
 [16.5, 15.0, 12.9, 11.6],
 [16.6, 15.1, 13.0, 11.7],
 [16.7, 15.2, 13.1, 11.8],
 [16.8, 15.3, 13.2, 11.9],
 [16.9, 15.4, 13.3, 12.0],
 [17.0, 15.5, 13.4, 12.1],
 [17.1, 15.6, 13.5, 12.2],
 [17.2, 15.7, 13.6, 12.3],
 [17.3, 15.8, 13.7, 12.4],
 [17.4, 15.9, 13.8, 12.5],
 [17.5, 16.0, 13.9, 12.6],
 [17.6, 16.1, 14.0, 12.7],
 [17.7, 16.2, 14.1, 12.8],
 [17.8, 16.3, 14.2, 12.9],
 [17.9, 16.4, 14.3, 13.0],
 [18.0, 16.5, 14.4, 13.1],
 [18.1, 16.6, 14.5, 13.2],
 [18.2, 16.7, 14.6, 13.3],
 [18.3, 16.8, 14.7, 13.4],
 [18.4, 16.9, 14.8, 13.5],
 [18.5, 17.0, 14.9, 13.6],
 [18.6, 17.1, 15.0, 13.7],
 [18.7, 17.2, 15.1, 13.8],
 [18.8, 17.3, 15.2, 13.9],
 [18.9, 17.4, 15.3, 14.0],
 [19.0, 17.5, 15.4, 14.1],
 [19.1, 17.6, 15.5, 14.2],
 [19.2, 17.7, 15.6, 14.3],
 [19.3, 17.8, 15.7, 14.4],
 [19.4, 17.9, 15.8, 14.5],
 [19.5, 18.0, 15.9, 14.6],
 [19.6, 18.1, 16.0, 14.7],
 [19.7, 18.2, 16.1, 14.8],
 [19.8, 18.3, 16.2, 14.9],
 [19.9, 18.4, 16.3, 15.0],
 [20.0, 18.5, 16.4, 15.1],
 [20.1, 18.6, 16.5, 15.2],
 [20.2, 18.7, 16.6, 15.3],
 [20.3, 18.8, 16.7, 15.4],
 [20.4, 18.9, 16.8, 15.5],
 [20.5, 19.0, 16.9, 15.6],
 [20.6, 19.1, 17.0, 15.7],
 [20.7, 19.2, 17.1, 15.8],
 [20.8, 19.3, 17.2, 15.9],
 [20.9, 19.4, 17.3, 16.0],
 [21.0, 19.5, 17.
```

```
Out[17]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
                2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
                2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

```
In [18]: import pandas as pd
y1 = pd.get_dummies(iris_data.target)
y1
```

Out[18]:

	0	1	2
0	1	0	0
1	1	0	0
2	1	0	0
3	1	0	0
4	1	0	0
5	1	0	0
6	1	0	0
7	1	0	0
8	1	0	0
9	1	0	0
10	1	0	0
11	1	0	0
12	1	0	0
13	1	0	0
14	1	0	0
15	1	0	0
16	1	0	0
17	1	0	0
18	1	0	0
19	1	0	0
20	1	0	0
21	1	0	0
22	1	0	0
23	1	0	0
24	1	0	0
25	1	0	0
26	1	0	0
27	1	0	0
28	1	0	0
29	1	0	0
...	...	...	...
120	0	0	1
121	0	0	1

	0	1	2
122	0	0	1
123	0	0	1
124	0	0	1
125	0	0	1
126	0	0	1
127	0	0	1
128	0	0	1
129	0	0	1
130	0	0	1
131	0	0	1
132	0	0	1
133	0	0	1
134	0	0	1
135	0	0	1
136	0	0	1
137	0	0	1
138	0	0	1
139	0	0	1
140	0	0	1
141	0	0	1
142	0	0	1
143	0	0	1
144	0	0	1
145	0	0	1
146	0	0	1
147	0	0	1
148	0	0	1
149	0	0	1

150 rows × 3 columns

```
In [25]: kn1 = KNeighborsClassifier(n_neighbors=2)
```

