K-NEAREST NEIGHBORS

1. IMPORT LIBRARY

```
import pandas as pd
from sklearn.utils import shuffle
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report
```

2. READ DATASET (CSV)

```
data = pd.read_csv('ruspini.csv')
```

```
<bound method NDFrame.head of      #   X   Y  CLASS
0     1   4  53      1
1     2   5  63      1
2     3  10  59      1
3     4   9  77      1
4     5  13  49      1
..   ..  ..  ..    ...
70   71  66  23      4
71   72  61  25      4
72   73  76  27      4
73   74  72  31      4
74   75  64  30      4
```

3. SPLIT DATA 80% FOR TRAINING AND 20% FOR TESTING

```
train1=data[0:16]
test1=data[16:20]
train2=data[20:34]
test2=data[34:37]
train3=data[37:55]
test3=data[55:60]
train4=data[60:72]
test4=data[72:75]
```

Data train1

```
[75 rows x 4 columns]>
      #   X   Y  CLASS
0     1   4  53      1
1     2   5  63      1
2     3  10  59      1
3     4   9  77      1
4     5  13  49      1
5     6  13  69      1
6     7  12  88      1
7     8  15  75      1
8     9  18  61      1
9    10  19  65      1
10   11  22  74      1
11   12  27  72      1
12   13  28  76      1
13   14  24  58      1
14   15  27  55      1
15   16  28  60      1
```

Data test1

```
      #   X   Y  CLASS
16   17  30  52      1
17   18  31  60      1
18   19  32  61      1
19   20  36  72      1
```

4. COMBINE DATA TRAIN AND DATA TEST

```
train=train1.append([train2,train3,train4])
test=test1.append([test2,test3,test4])
```

Data train

	#	X	Y	CLASS
0	1	4	53	1
1	2	5	63	1
2	3	10	59	1
3	4	9	77	1
4	5	13	49	1
5	6	13	69	1
6	7	12	88	1
7	8	15	75	1
8	9	18	61	1
9	10	19	65	1
10	11	22	74	1
11	12	27	72	1
12	13	28	76	1
13	14	24	58	1
14	15	27	55	1
15	16	28	60	1
20	44	86	132	2
21	45	85	115	2
22	46	85	96	2
23	47	78	94	2
24	48	74	96	2

Data test

	#	X	Y	CLASS
16	17	30	52	1
17	18	31	60	1
18	19	32	61	1
19	20	36	72	1
34	58	111	126	2
35	59	115	117	2
36	60	117	115	2
55	39	52	152	3
56	40	55	155	3
57	41	54	124	3
58	42	60	136	3
59	43	63	139	3
72	73	76	27	4
73	74	72	31	4
74	75	64	30	4

5. SHUFFLE DATA TRAIN AND DATA TEST

```
train=shuffle(train)
test=shuffle(test)
```

Data train

	#	X	Y	CLASS
65	66	78	16	4
9	10	19	65	1
8	9	18	61	1
63	64	61	15	4
25	49	97	122	2
43	27	38	145	3
71	72	61	25	4
48	32	44	149	3
47	31	44	156	3
69	70	69	21	4
23	47	78	94	2
50	34	46	142	3
40	24	33	154	3
51	35	47	149	3
24	48	74	96	2
11	12	27	72	1
49	33	44	143	3
46	30	34	141	3
33	57	108	116	2
70	71	66	23	4
15	16	28	60	1
0	1	4	53	1

Data test

	#	X	Y	CLASS
73	74	72	31	4
16	17	30	52	1
17	18	31	60	1
57	41	54	124	3
18	19	32	61	1
55	39	52	152	3
35	59	115	117	2
58	42	60	136	3
56	40	55	155	3
72	73	76	27	4
19	20	36	72	1
36	60	117	115	2
74	75	64	30	4
59	43	63	139	3
34	58	111	126	2

6. REMOVE LABEL FROM DATA TRAIN AND DATA TEST FOR INPUT

```
train_in = train.drop(columns=['CLASS','#'])
test_in  = test.drop(columns=['CLASS','#'])
```

Input data train

	X	Y
15	28	60
64	69	15
9	19	65
53	50	142
25	97	122
32	110	111
33	108	116
47	44	156
27	98	124
39	35	153
48	44	149
5	13	69
43	38	145
10	22	74
2	10	59
6	12	88
22	85	96
42	41	150

Input data test

	X	Y
16	30	52
35	115	117
19	36	72
36	117	115
55	52	152
73	72	31
34	111	126
57	54	124
59	63	139
17	31	60
18	32	61
72	76	27
74	64	30
58	60	136
56	55	155

7. REMOVE LABEL FROM DATA TRAIN AND DATA TEST FOR OUTPUT

```
train_out = train['CLASS']
test_out =  test['CLASS']
```

Output data train

33	2
2	1
32	2
4	1
39	3
64	4
26	2
9	1
47	3
43	3
62	4
0	1
51	3
37	3
46	3
71	4
52	3
25	2
41	3
65	4

Output data test

34	2
73	4
56	3
18	1
57	3
35	2
74	4
72	4
16	1
19	1
59	3
55	3
17	1
36	2
58	3

7. KNN CLASSIFIER (K=5)

```
classifier = KNeighborsClassifier(n_neighbors=5)
classifier.fit(train_in, train_out)
```

8. PREDICT DATA TEST USING KNN

```
pred_out = classifier.predict(test_in)
```

The predict data test

```
[1 3 1 4 3 3 1 1 3 4 2 2 3 4 2]
```

Real Output data test

16	1
55	3
18	1
73	4
59	3
58	3
17	1
19	1
56	3
74	4
34	2
36	2
57	3
72	4
35	2

9. CLASSIFICATION REPORT

```
print(classification_report(test_out, pred_out))
```

The predict data test

	precision	recall	f1-score	support
1	1.00	1.00	1.00	4
2	1.00	1.00	1.00	3
3	1.00	1.00	1.00	5
4	1.00	1.00	1.00	3
accuracy			1.00	15
macro avg	1.00	1.00	1.00	15
weighted avg	1.00	1.00	1.00	15

ANALYSIS OF KNN (K=5)

After using KNN with K=5, the result is

- Label “1” precision 100%, recall 100% and f1-score 100%
- Label “2” precision 100%, recall 100% and f1-score 100%
- Label “3” precision 100%, recall 100% and f1-score 100%
- Label “4” precision 100%, recall 100% and f1-score 100%

The result of using KNN with K=5 in this case is excellent, there is no error of each labels. This happens because the dataset for training is good.

However, not all cases can produce a great result with K equals 5 or bigger because the optimum value of K is different each cases