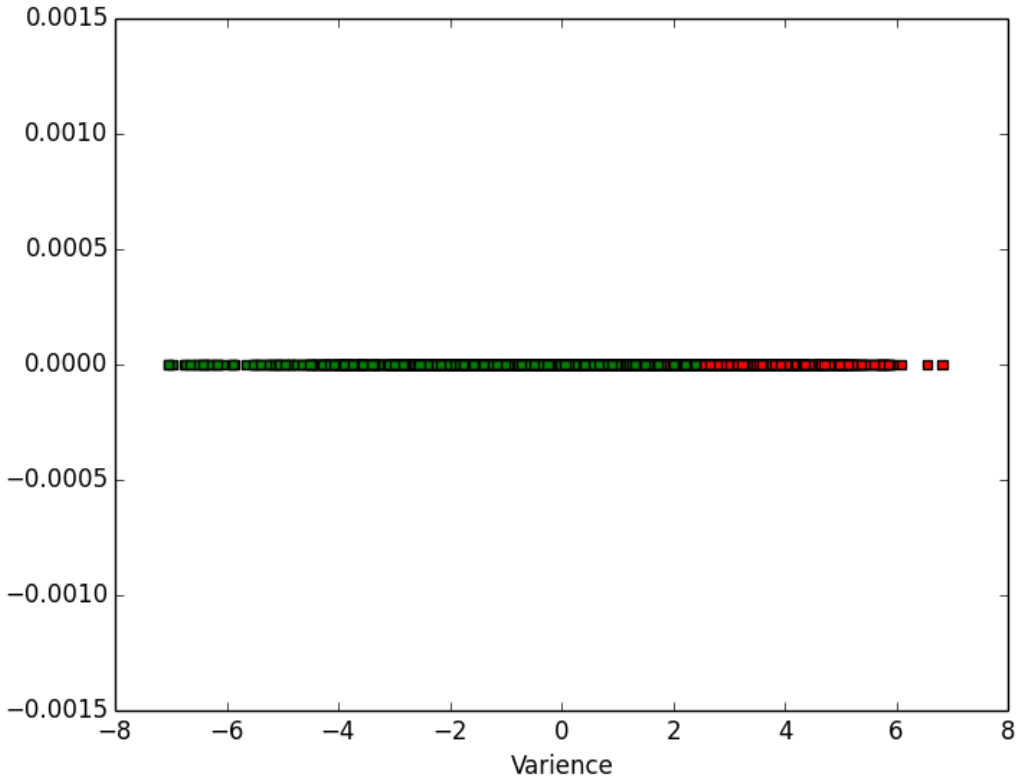
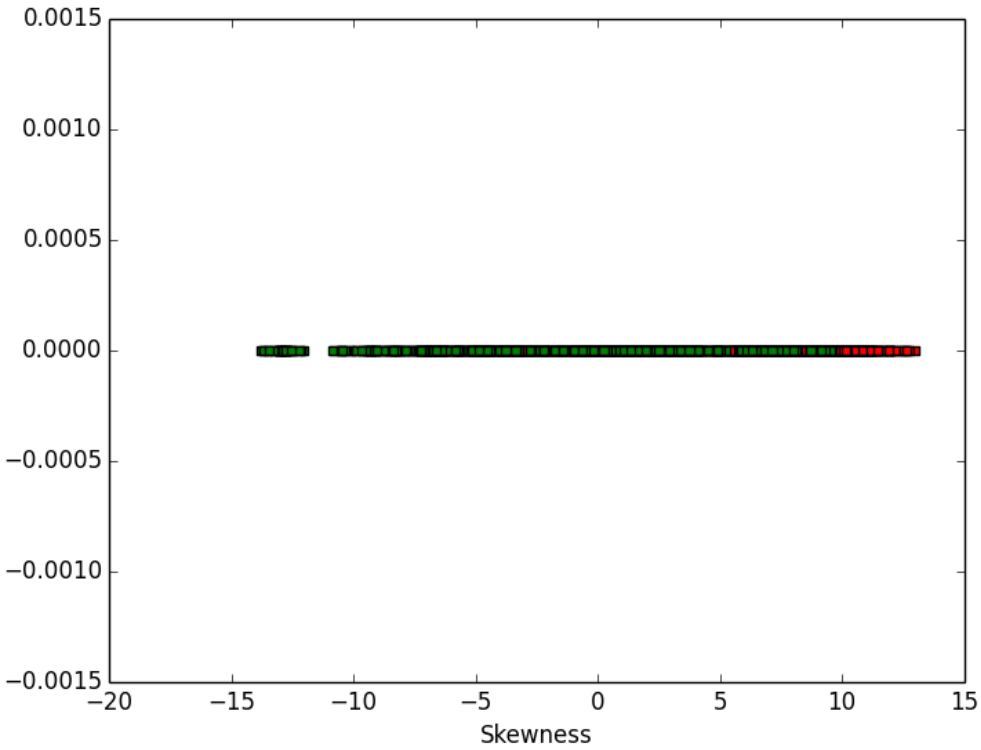
Banknote authentication Dataset

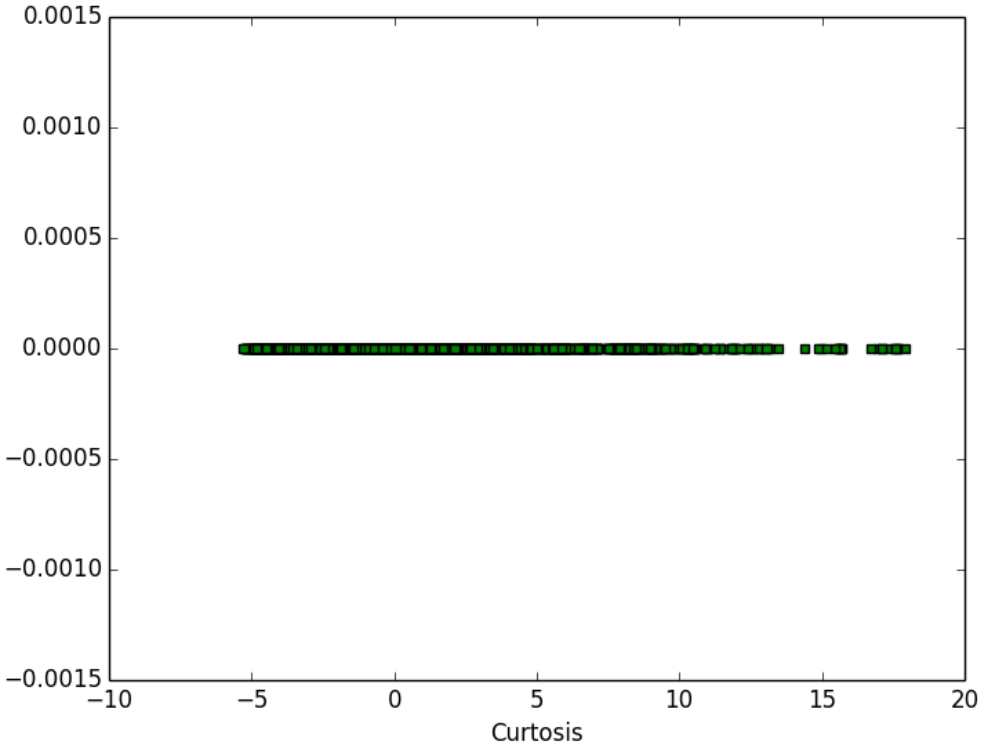
Solution:

(b)

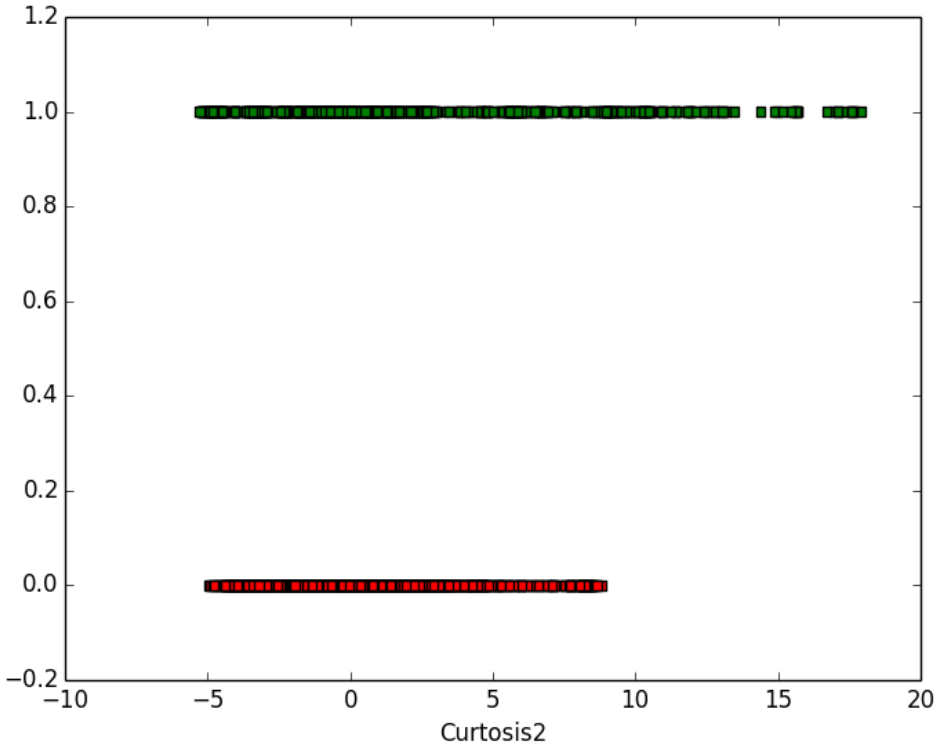
(i)Class 0 is red, Class 1is green.

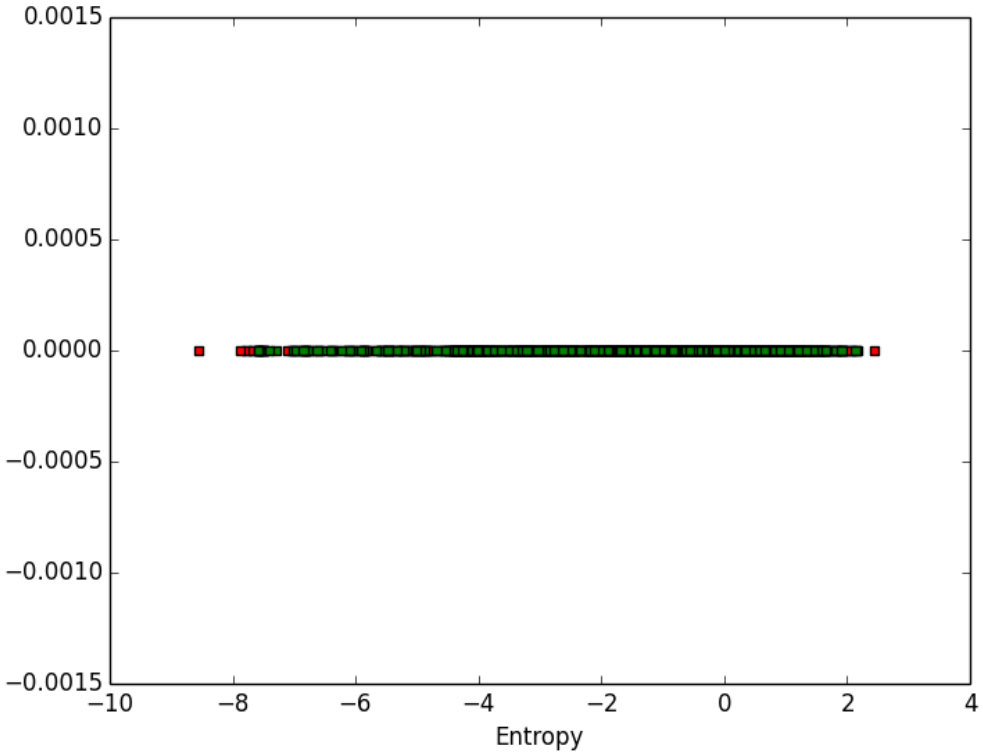



0.0015
0.0010
0.0005
0.0000
−0.0005
−0.0010
−0.0015
−20
−15
−10
−5
0
5
10
15
Skewness


0.0015
0.0010
0.0005
0.0000
−0.0005
−0.0010
−0.0015
−10
−5
0
5
10
15
20
Curtosis

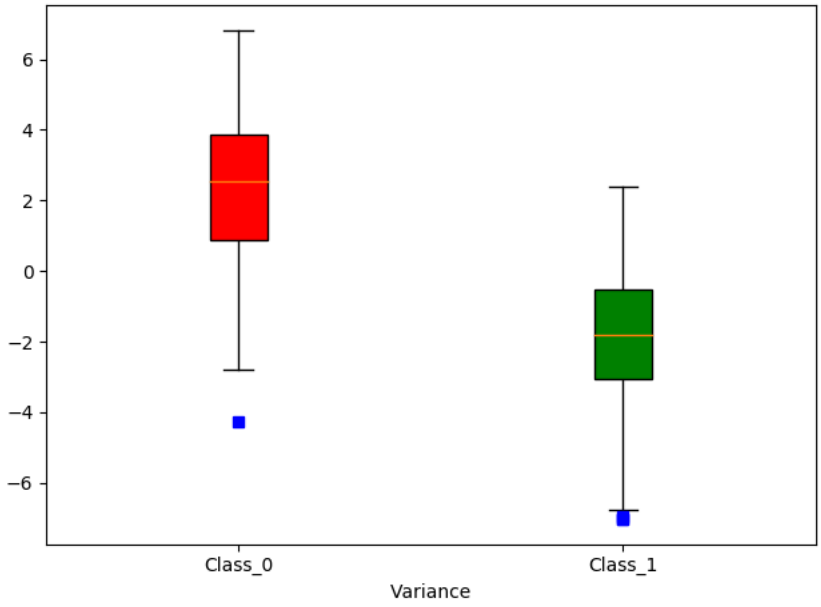
In this picture we can only see Class 1 in green color, because data in Class 0 is totally behind Class 1. So I separate two classes of data and print them in one picture as below:

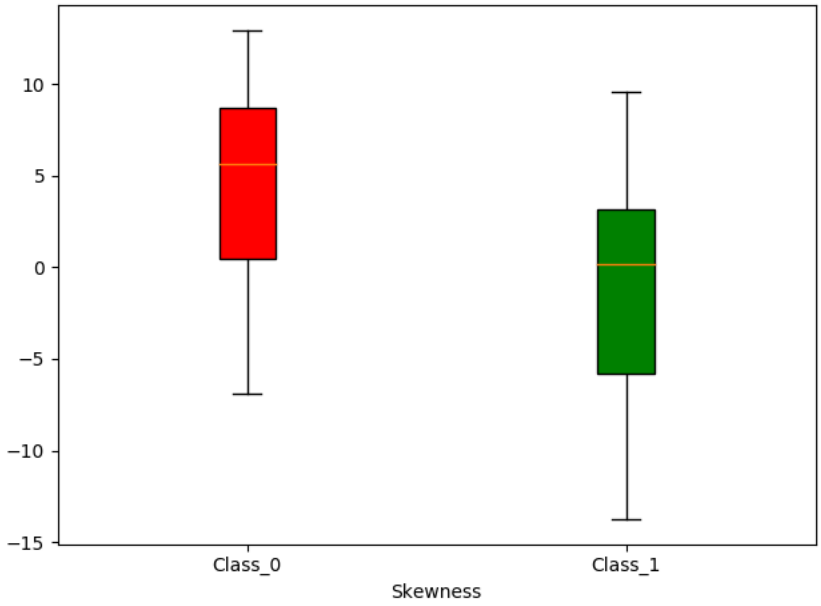


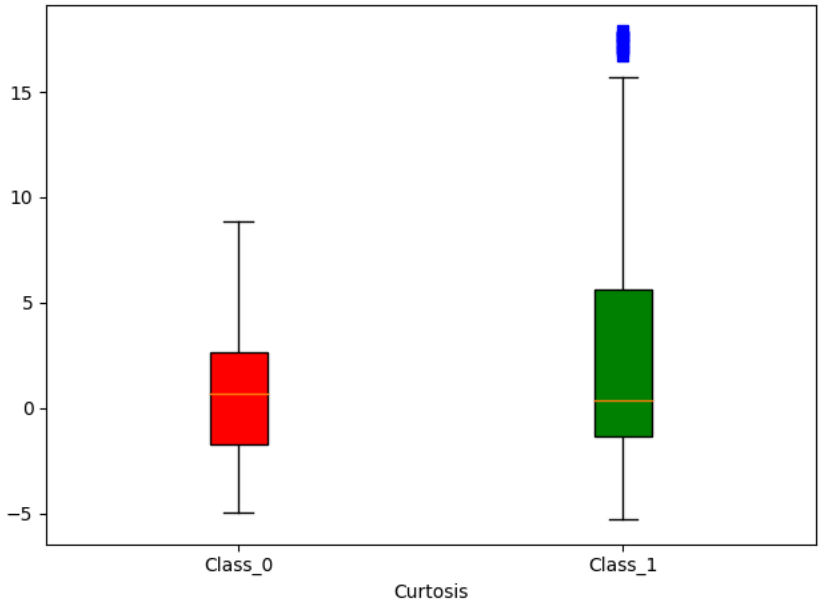


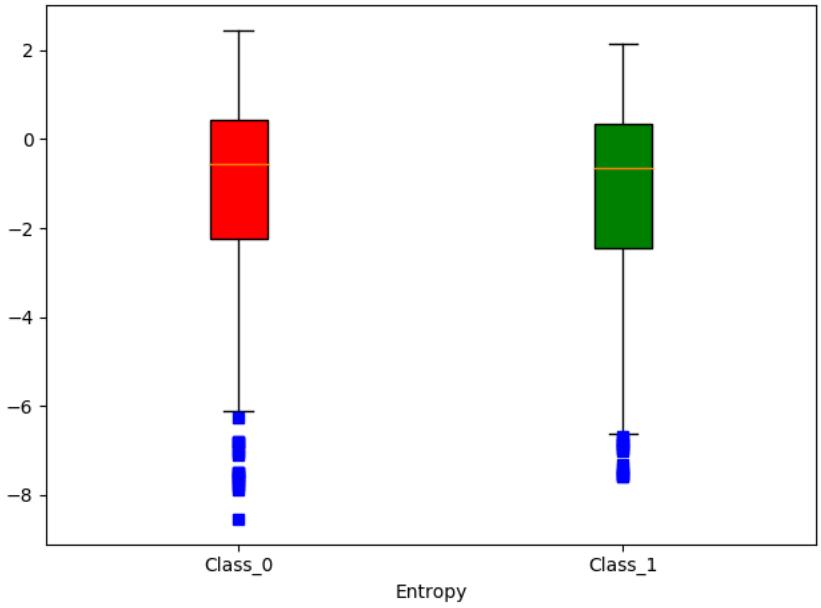
(ii)

Code: “boxplots_1.py”


6
4
2
0
−2
−4
−6
Class_0
Class_1
Variance

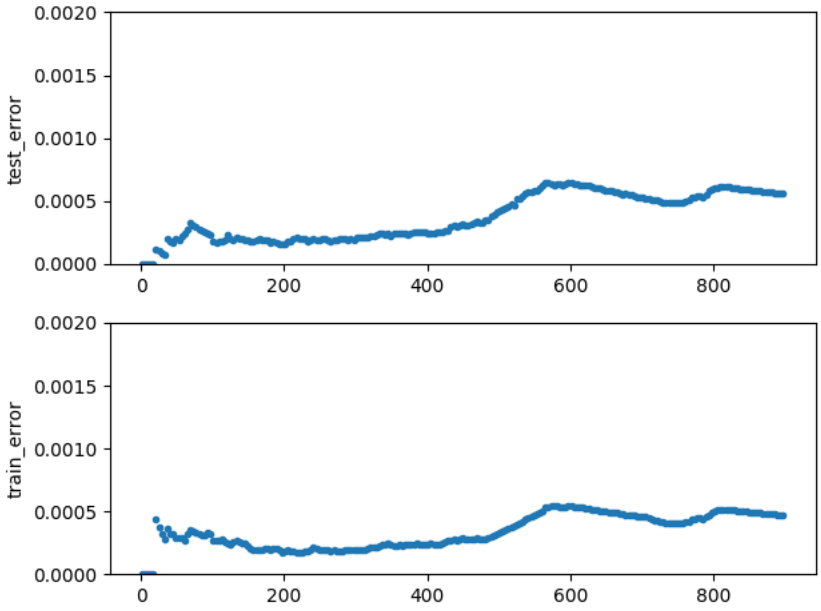

10
5
0
−5
−10
−15
Class_0
Class_1
Skewness





(c)

Code: “KNN_1.py” and “KNN_2.py”



We can see that when K* = 19, it is the most suitable value.

```
>>> confusion_matrix
<function confusion_matrix at 0x1106d8e18>
>>> confusion_matrix(test_y, test_predict)
array([[200,   0],
       [  0, 200]])

>>>
```

We can calculate true positive rate, true negative rate, precision, F-score with confusion matrix given by picture:

true positive rate = 200/(200+0) = 100%

true negative rate = 0/(200+0) = 0%

precision = 200/200 = 100%

F-score = 0%

(d)

(i)

Code: “KNN_3.py”

Test_error:

0.00E+00
0.00E+00
1.19E-04
2.42E-04
6.10E-05
4.90E-05
1.64E-04
2.11E-04
2.78E-04
2.47E-04
2.97E-04
2.93E-04
2.69E-04
3.05E-04
3.01E-04

3.31E-04
3.26E-04
3.51E-04
3.31E-04
3.53E-04
3.86E-04
3.55E-04
3.85E-04
3.57E-04
3.73E-04
3.78E-04
3.83E-04
3.78E-04
3.74E-04
3.78E-04
3.65E-04
3.54E-04
3.82E-04
3.85E-04
3.89E-04

4.13E-04
4.16E-04
4.18E-04
4.20E-04
4.80E-04
4.80E-04
5.05E-04
5.11E-04
5.05E-04
5.10E-04
5.10E-04
5.15E-04
5.36E-04
5.56E-04
5.70E-04
5.99E-04
6.21E-04
6.38E-04
6.26E-04
6.47E-04

6.44E-04
6.55E-04
6.39E-04
6.24E-04
6.22E-04
6.07E-04
5.97E-04
5.88E-04
5.74E-04
5.62E-04
5.49E-04
5.45E-04
5.37E-04
5.36E-04
5.35E-04
5.35E-04
5.70E-04
6.07E-04
6.19E-04
6.24E-04

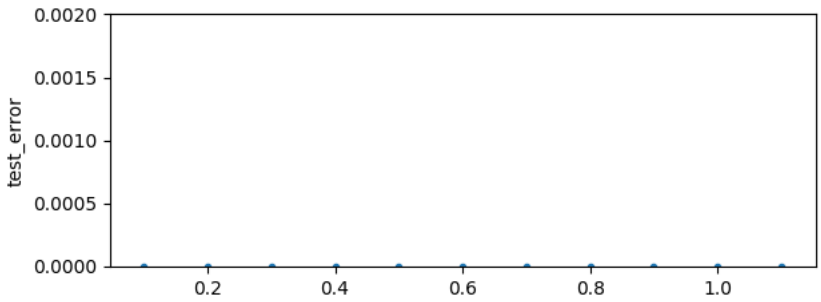
6.19E-04
6.08E-04
6.10E-04
6.05E-04
6.10E-04
6.12E-04
6.07E-04
6.09E-04
6.02E-04
5.95E-04
5.88E-04
5.81E-04
5.74E-04
5.68E-04
5.61E-04

We can see that k* = 11.

(ii)

Code: “KNN_4.py”

When k =11, log10(p) ∈ {0.1,0.2,0 log10(p) ∈ {0.1,0.2,0.3,...,1}, we can print the error in one figure and we can find that the best is log10(1):

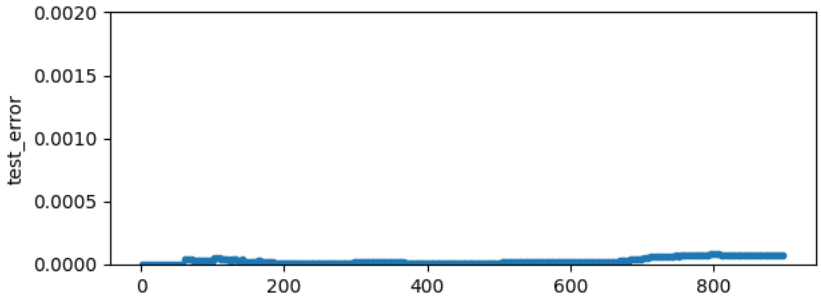


When p = 0.1, it becomes Chebyshev Distance

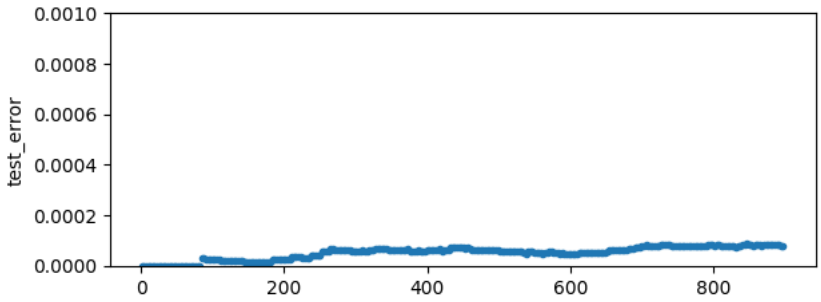
(e)

Code: "KNN_5.py"

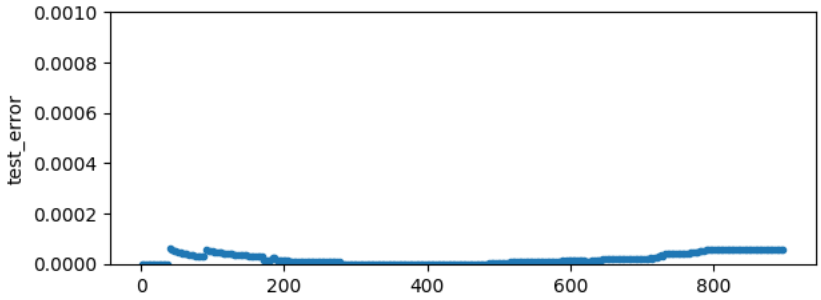
When use Euclidean distance with weighted decision,



When use Manhattan distance with weighted decision,



When use Chebyshev distance with weighted decision,



We can see that the best tese_error are all 0%.

(f)

The lowest training error rate I achieved in this exercise is 0%.