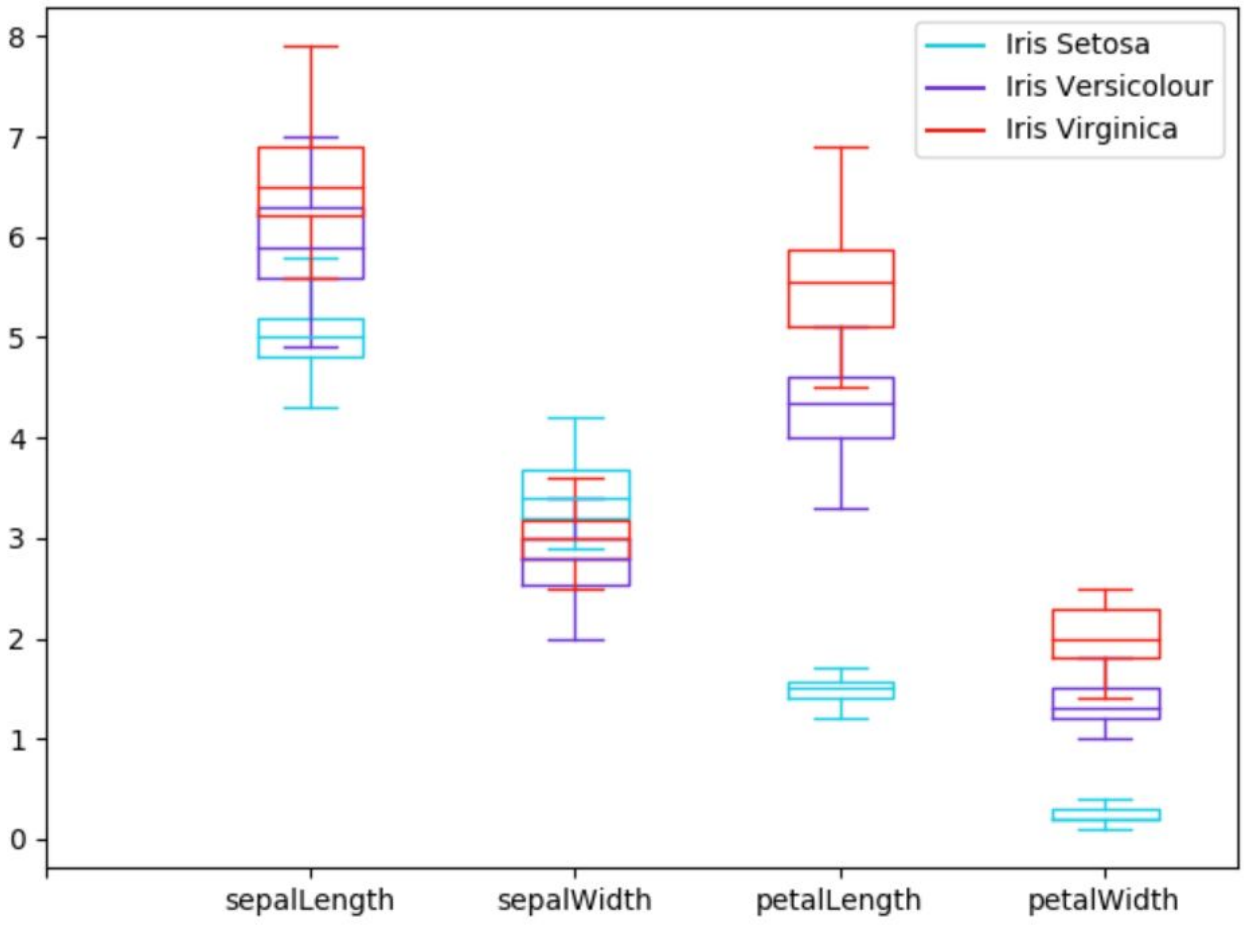
1: A-C: See Code

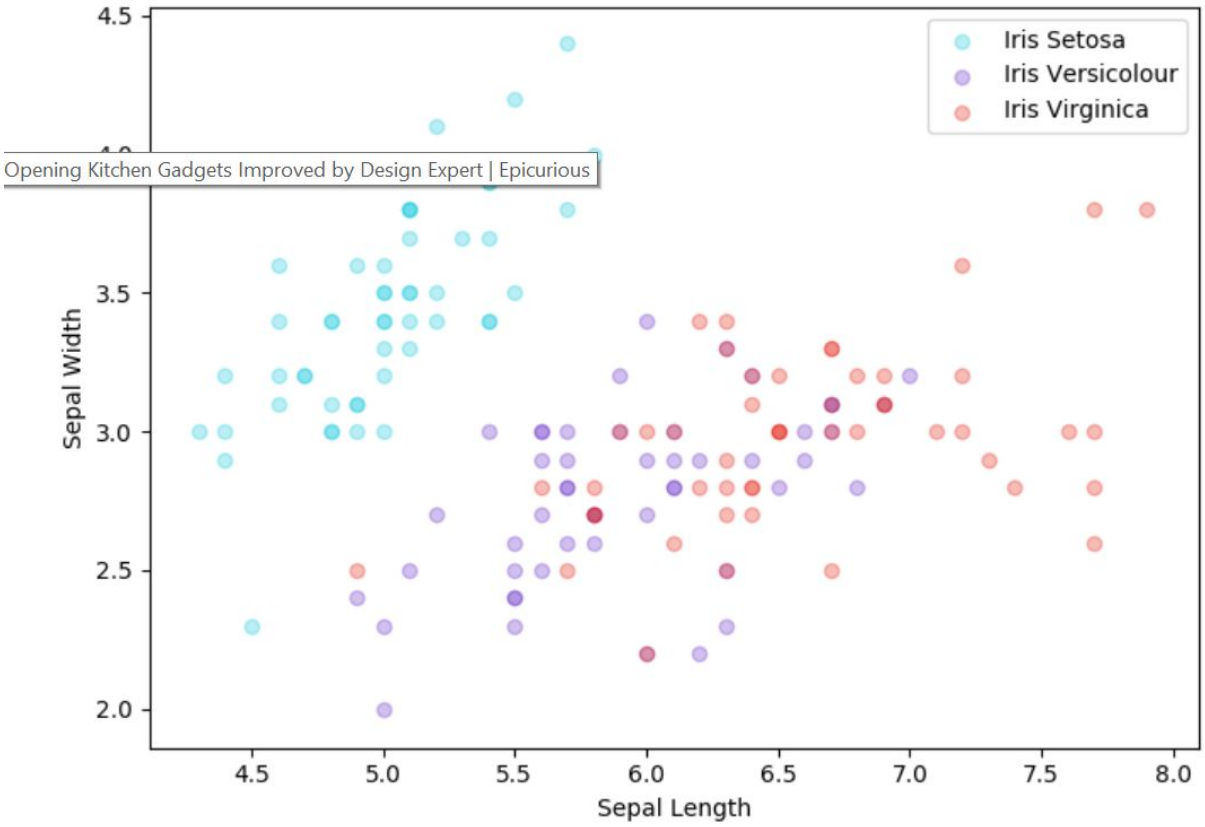
1D: I ran the program about five times and the difference of time elapse between the approaches was about 1.24 units

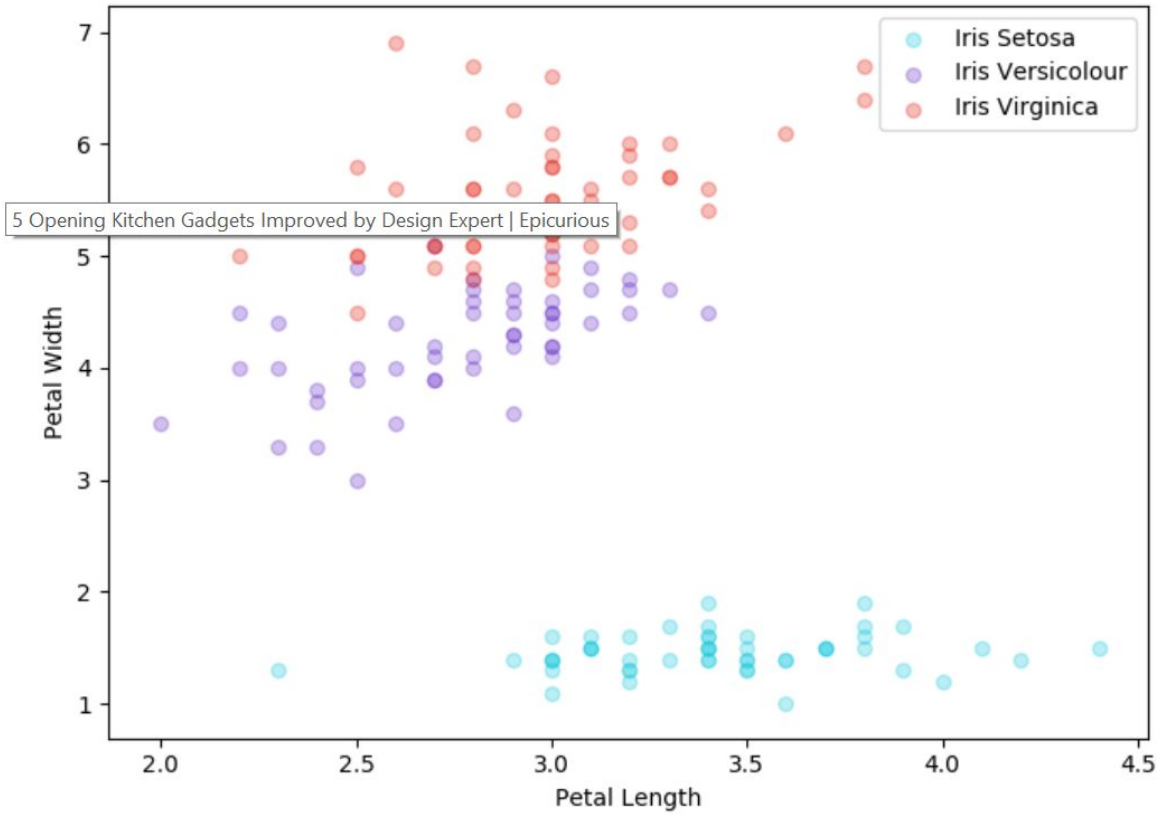
2: A: See code

2B:



2C

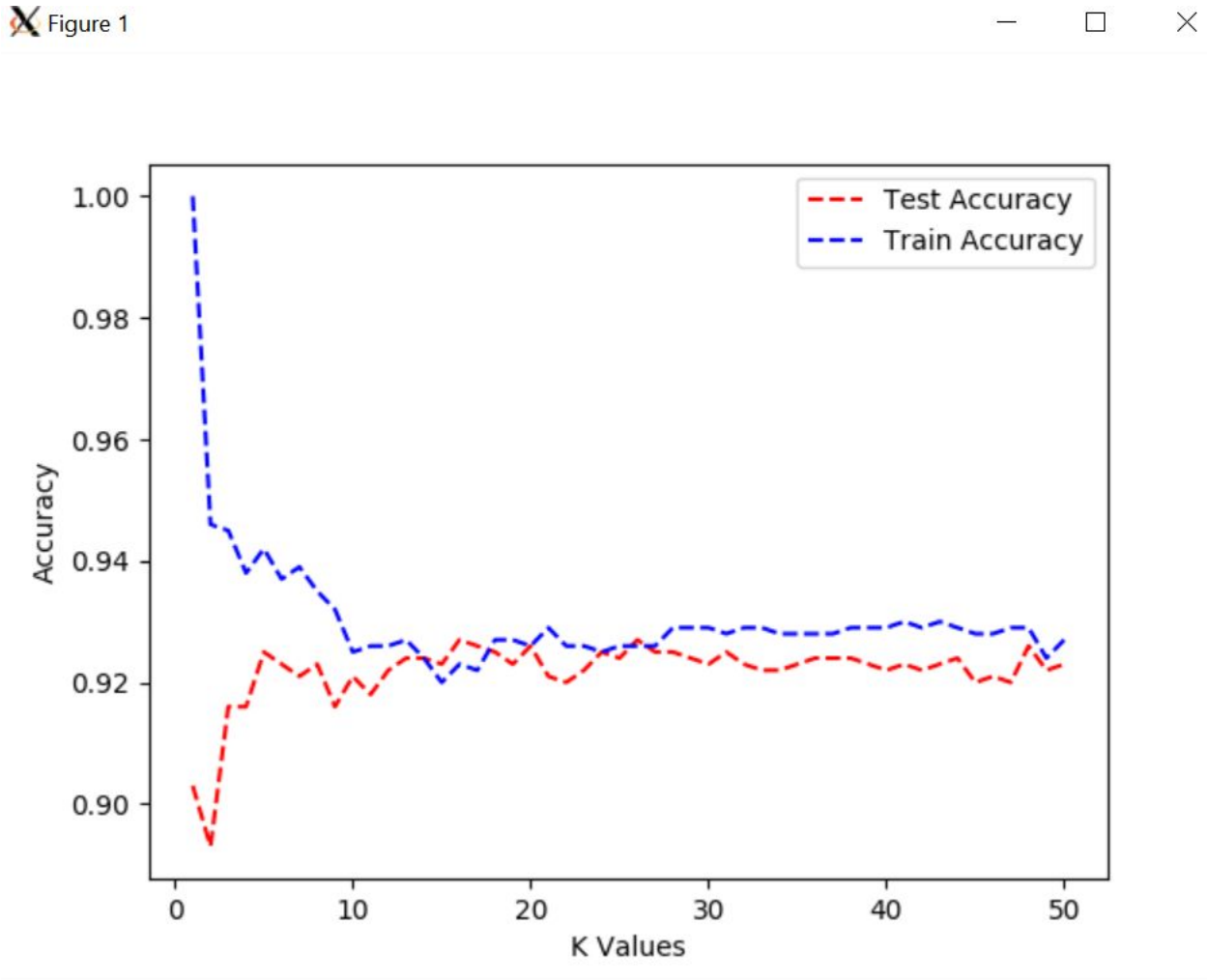

Iris Setosa
Iris Versicolour
Iris Virginica
Opening Kitchen Gadgets Improved by Design Expert | Epicurious
Sepal Width
Sepal Length
4.5
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0



2D Some rules I came up with to classify species type is first by looking at sepal and petal length vs width for each of the species type. Iris Setosa has larger sepal width and larger petal length compared to the other species type. Also Iris Virginica has the largest petal width among the species type

3: A-C: See Code

3D



3E: The computational complexity is O[N * ((N * D) + NlogN + 2K)]. Going through my predict function my first loop goes through the xFeat which has a size of N. Then inside the loop the findNeighbor function. In that function I go through the training data array with size N and then inside that loop I calculate the euclidean distance, which the number of operations scale by d features. The D operations are squared but then you square root the D squared giving you D operations. Next I quicksorted the distances which has a runtime of nlogn, after that I had to loops that have a length of K giving you runtime of 2K.

4: A-C: See Code

4D For most values when you scale the data you get better accuracy than not processing the data. Also the knn algorithm can't handle irrelevant features, the accuracy usually is lower than all the other methods but sometimes the accuracy gets skewed.