

REPORT

Parzen Window	1
Purpose	1
Approach	1
Interpretation	2
KNN	4
The KNN with 1 observation	4
Approach	4
Interpretation	6
The second KNN with different attributes	6
Approach	6
Interpretation	7

Parzen Window

Purpose

Using different heights and number of training samples to generate plots and classify test observations.

Approach

For this lab, we used three different widths, and four different numbers of samples to test the classification and plots using Parzen Window and KNN. With each combination of width and samples, we generated training samples and testing samples in order to approximate the PDF.

The testing sample is predefined in the range from -2 to 2 with the increment of 0.1. For the training samples, we get n number of points from a normal distribution with the mean = 0, and the standard deviation equal to 1.

For each testing observations, we calculated the Parzen Window ($p(x)$) by using all training observations versus each training sample. We then calculated the Parzen Window by taking the average of the single testing sample. MatLab has a cool syntax

that allow us to subtract a single value from an array of number. That is why we do not need an additional for loop to subtract each training samples from the single testing observations.

In plotting, each plot represents a specific n number of samples against each width comprised of the parzen window of each test observation.

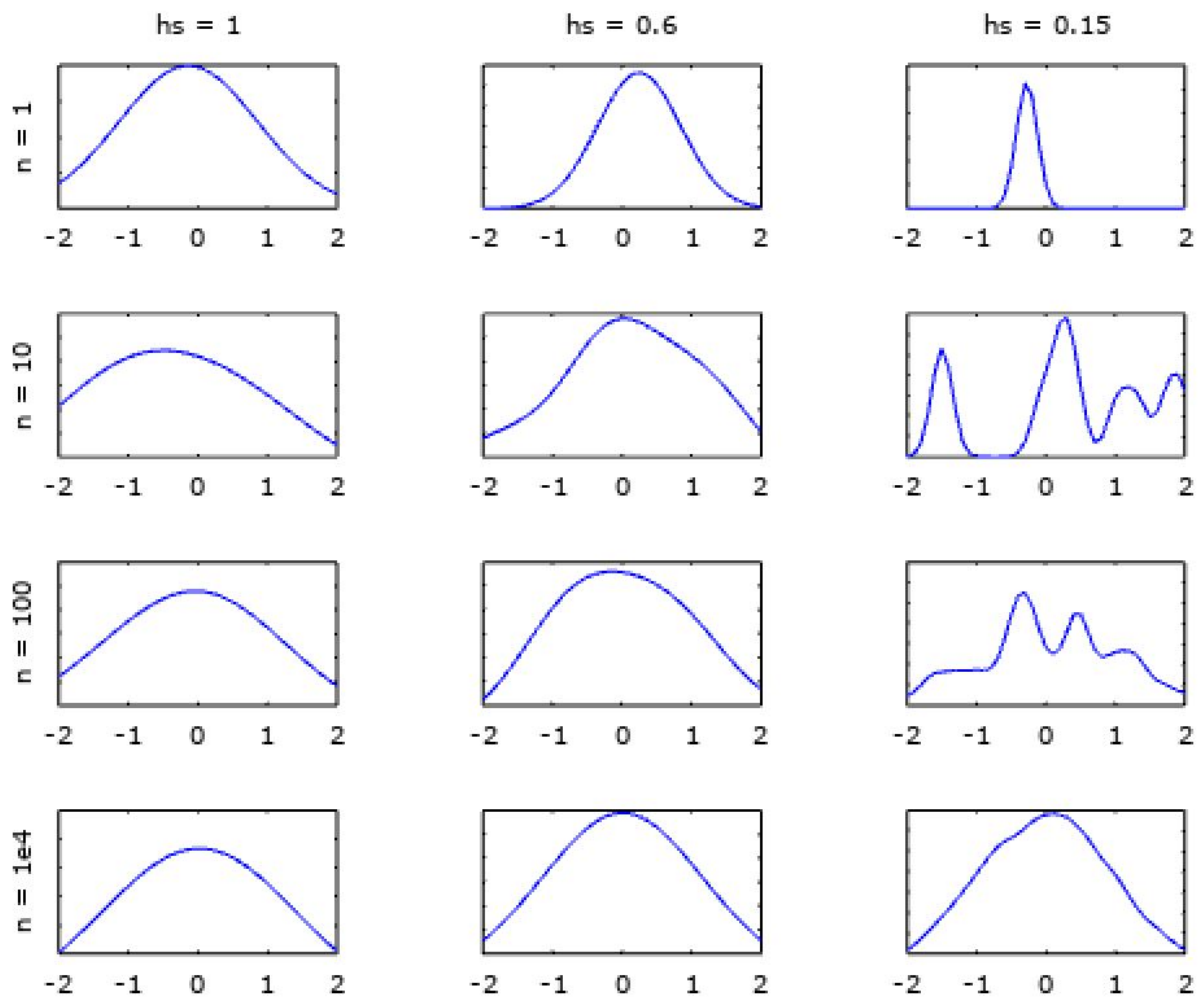
Interpretation

Below is the plot that we generated when using different h and w . How well the plot approximate a normal distribution according to the Central Limit Theorem depends on the width and the number of training observations. Looking at the plot, I can conclude that as the number of observations becomes larger, the better the plot approximate the normal distribution.

In addition, the window width also affect the width of the plot, which represent the standard deviation. I can conclude that as the width increase, the plot tends to be spread further (increasing standard deviation). This is because the test observation is trying to capture more points around it, because of the increase width. So, we see there is more variation. However, as the width become smaller, the standard deviation becomes smaller as well (tighter).

An interesting point of observation is that; as the width become smaller, the normal distribution at that point also varies with the distribution at other points as well. Whereas in other cases, all the points were unable to connect to each other and approximate a normal distribution.

Figure 1: Plots that show the combination of different h and w



KNN

In this lab, I used KNN in two places. One is in the Parzen Window file, in which it is just an example from where only one test observation is used. I decided to include it along with the actual KNN with nine test observations for reference. The second being in the KNN portion of the assignment.

The KNN with 1 observation

Approach

The approach is the same as the example in which the data is loaded from the data set and saved into a variable. From that variable, the 3rd and 4th feature (example) are selected and plotted into a scatter matrix by using those two features.

After adding the new test point and marking it on the plot, I continued to use the method `KNNSearch` that returns two variables, `n`, and `d`, in which `n` is the index of the neighbor in the variable that has the data set, and `d` is the euclidian distance of those points to the test observations.

Then I drew a circle that encompasses the 10 neighbors around the test observation and use the `tabulate` function to calculate the percentage of this test observations belong to the specified class. The class with the highest percentage is the classified observation of the test observation.

Figure 2: KNN search on a single test observation of the 3rd and 4th feature

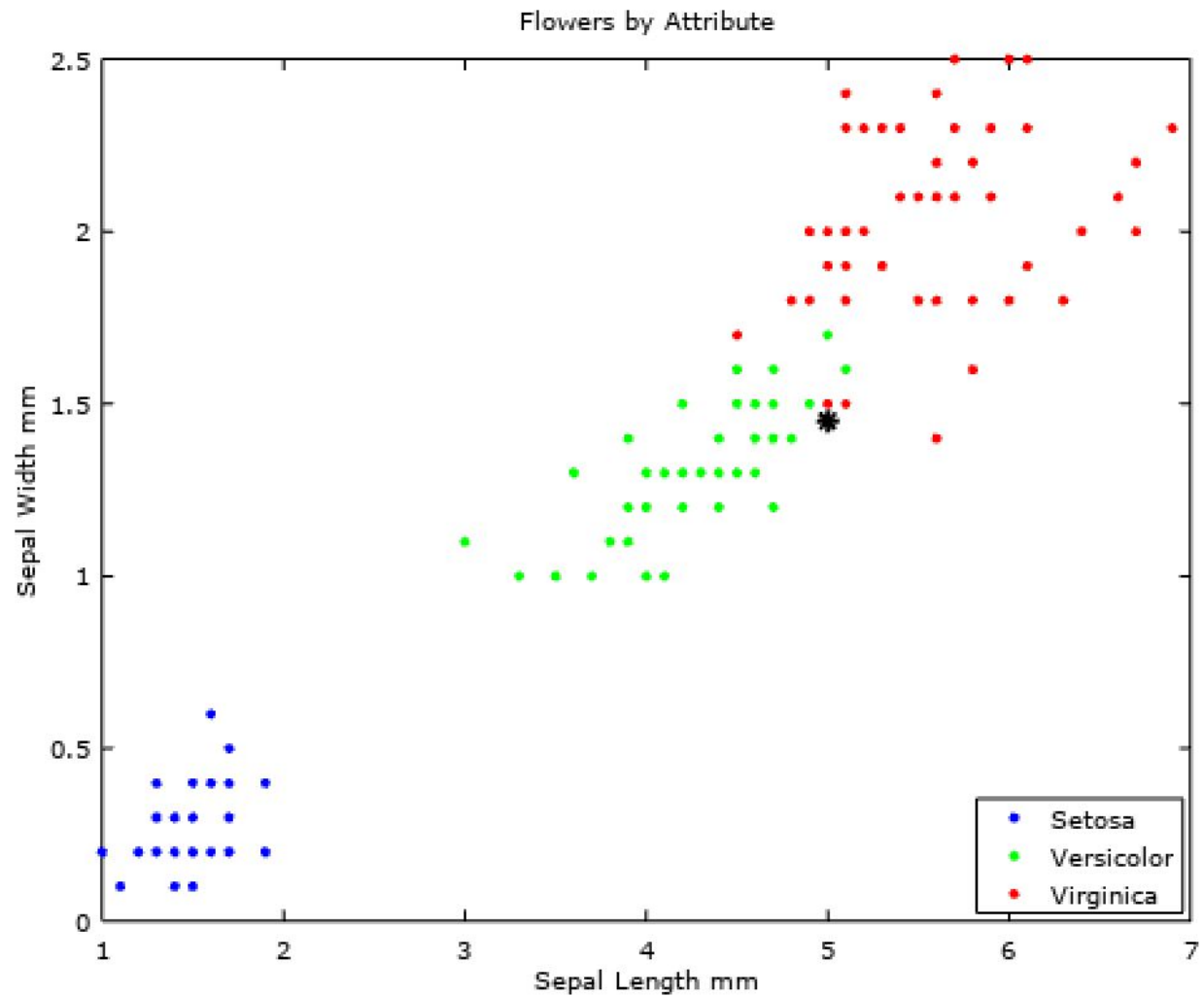
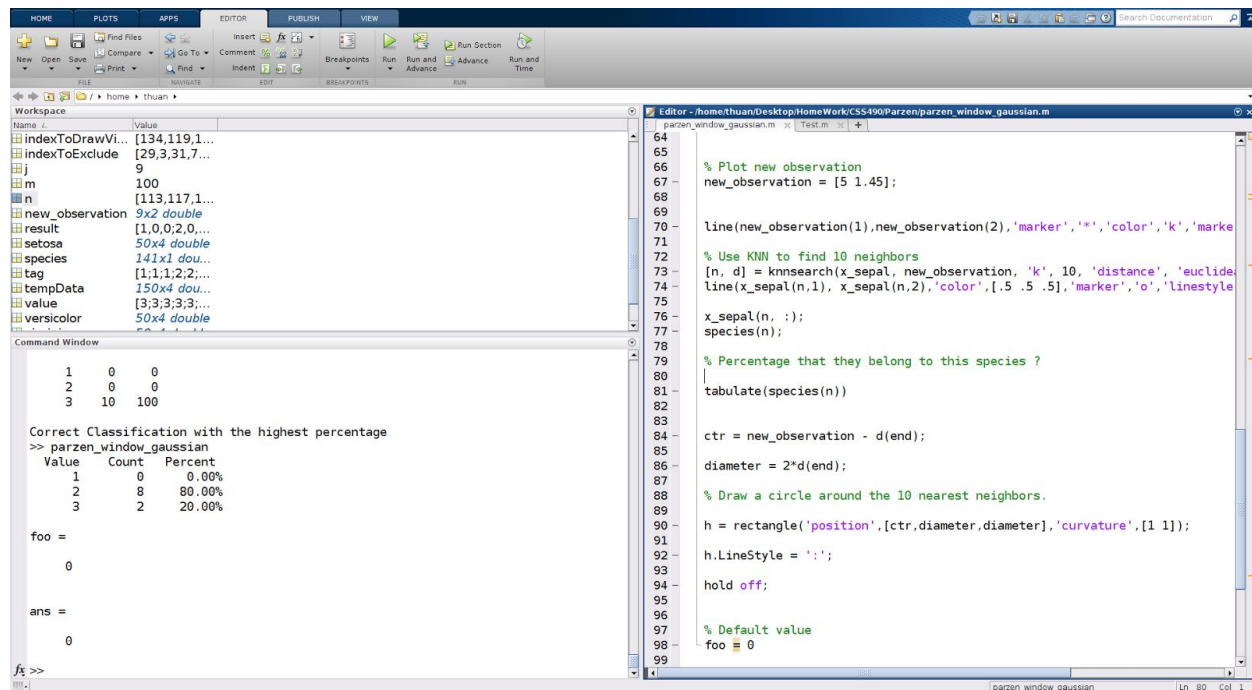


Figure 3: Classification results of first KNN



Interpretation

Looking at the result, we can determine that the test observation belongs to the second class since it has the highest percentage. Most of the points in the 10 neighbors belong to class 2.

The second KNN with different attributes

Approach

The here approach is similar to the above example, where the data is loaded into variables.

The only difference is that this time, the different attributes of the data set need to be combined. We have four attributes, and I did the combination as follows: 1 2, 2,3 ,1 3,..etc.

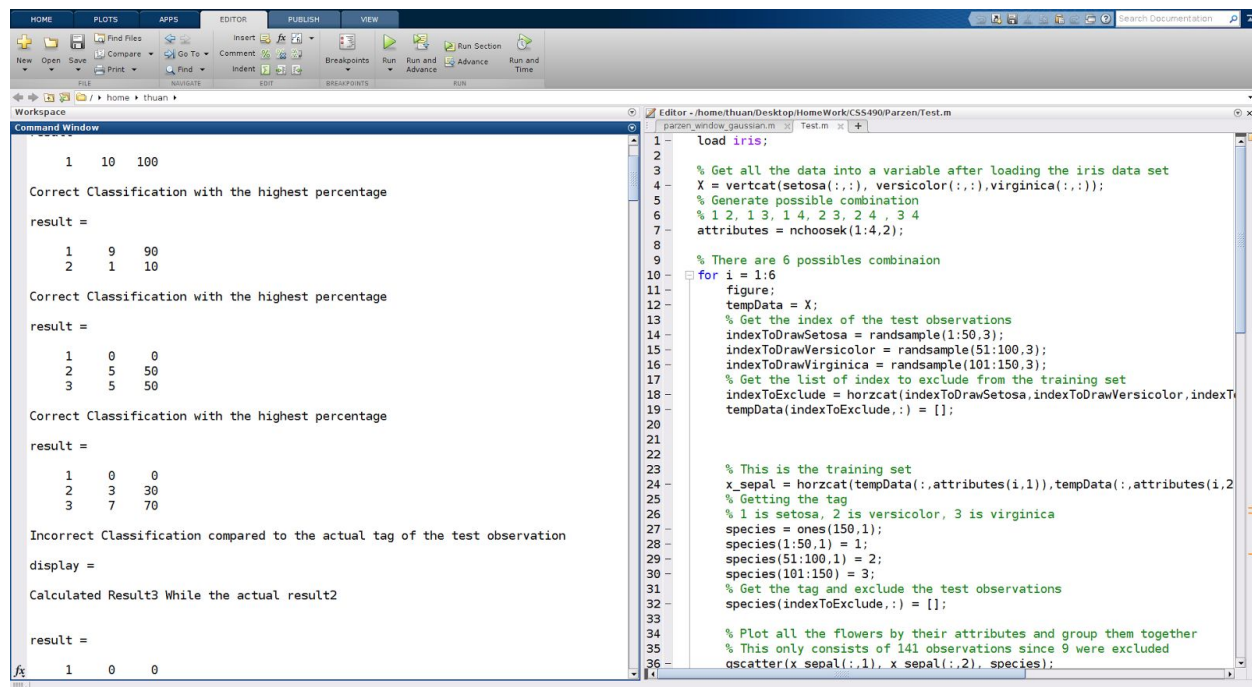
For each attributes combination, I exclude nine random observations as test observations and only use the remaining 141 observations as the training data (each

run uses different test observations and training samples since I randomly select new points each time).

Then, for each test observations, I used the KNNSearch function and generate the result and mark the point in the plot. Below, Figure 4 refers to the result when the script is ran. It will print out if the classification is correct (by using the class of the highest percentage) and the actual class tag

If it is not correct, then it will display not correct and what is the correct class

Figure 4: Result of classification using KNN with various attributes and test observations



```
1 load iris;
2
3 % Get all the data into a variable after loading the iris data set
4 X = vertcat(setosa(:,1:4), versicolor(:,1:4), virginica(:,1:4));
5 % Generate possible combination
6 % 1 2, 1 3, 1 4, 2 3, 2 4, 3 4
7 attributes = nchoosek(1:4,2);
8
9 % There are 6 possible combination
10 for i = 1:6
11     figure;
12     tempData = X;
13     % Get the index of the test observations
14     indexToDrawSetosa = randsample(1:50,3);
15     indexToDrawVersicolor = randsample(51:100,3);
16     indexToDrawVirginica = randsample(101:150,3);
17     % Get the list of index to exclude from the training set
18     indexToExclude = horzcat(indexToDrawSetosa, indexToDrawVersicolor, indexToDrawVirginica);
19     tempData(indexToExclude,:) = [];
20
21
22
23 % This is the training set
24 x_sepal = horzcat(tempData(:,attributes(i,1)), tempData(:,attributes(i,2)));
25 % Getting the tag
26 % 1 is setosa, 2 is versicolor, 3 is virginica
27 species = ones(150,1);
28 species(1:50,1) = 1;
29 species(51:100,1) = 2;
30 species(101:150,1) = 3;
31 % Get the tag and exclude the test observations
32 species(indexToExclude,1) = [];
33
34 % Plot all the flowers by their attributes and group them together
35 % This only consists of 141 observations since 9 were excluded
36 scatter(x_sepal(:,1), x_sepal(:,2), species);
```

Command Window

```
1 10 100
Correct Classification with the highest percentage
result =
1 9 90
2 1 10
Correct Classification with the highest percentage
result =
1 0 0
2 5 50
3 5 50
Correct Classification with the highest percentage
result =
1 0 0
2 3 30
3 7 70
Incorrect Classification compared to the actual tag of the test observation
display =
Calculated Result3 While the actual result2
result =
1 0 0
```

Interpretation

Looking at the results, I can see that knn did well with classifying. Mostly because it works in a similar way to the Parzen Window in that it get the neighbors among the test observations and chooses the one that is most likely to be the observations class. I noticed that in some cases the data was classified wrong between versicolor and virginica. This is most likely because these two classes are clustered close toward each other. The test observations might be in an area that is not close, or related to its flower, but more toward the other types of flowers. So, when we use KNN on those points, it may produce inaccurate result since it captures more points that belong to the other classes instead of the actual class that this observation belongs to.

Below are some figures that represent the KNN of some test observations using the combination of 2 different features

Note that 1 represent setosa, 2 represent versicolor and 3 represent virginica

Figure 5: KNN with Petal Width and Petal Length

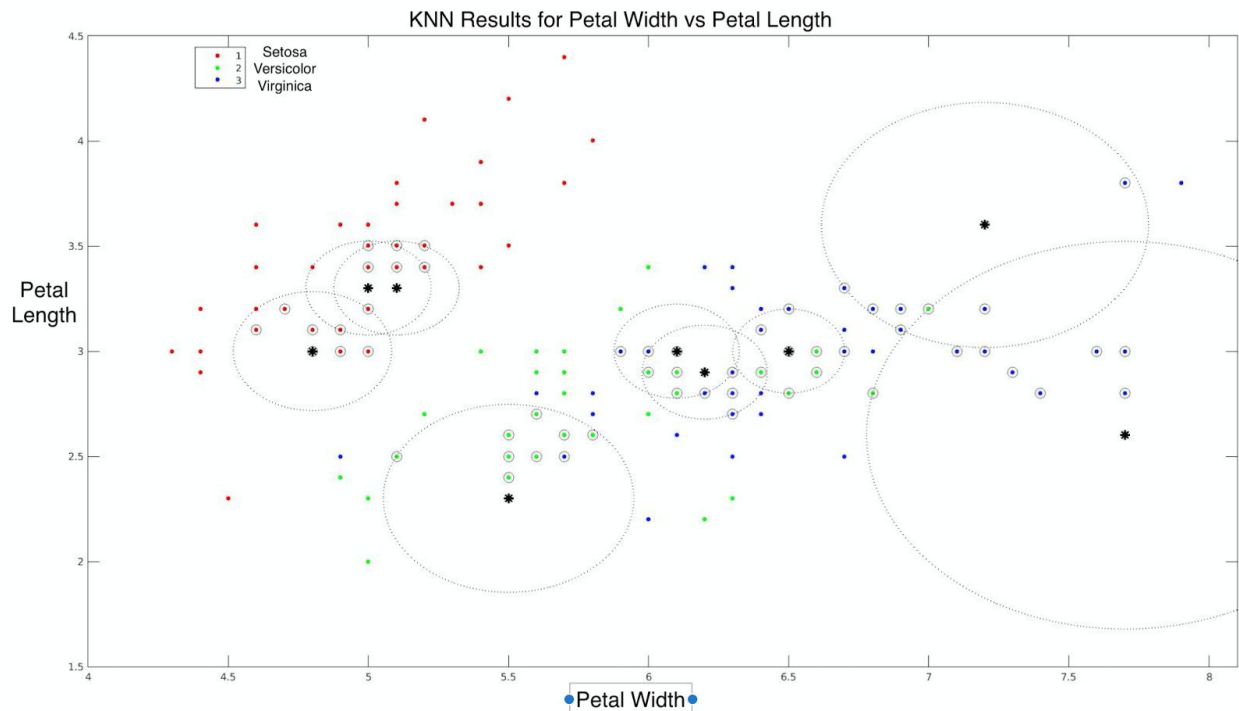


Figure 6: KNN with Petal Width and Sepal Width

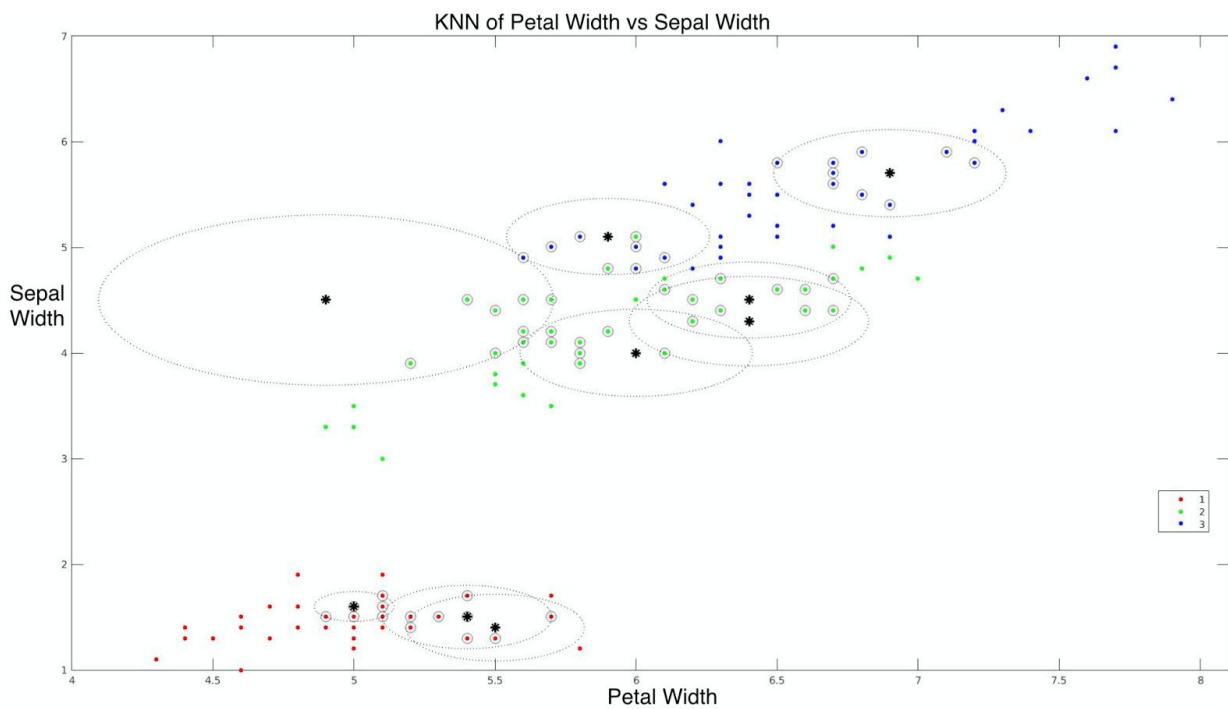


Figure 7: KNN of Sepal Length and Petal Width

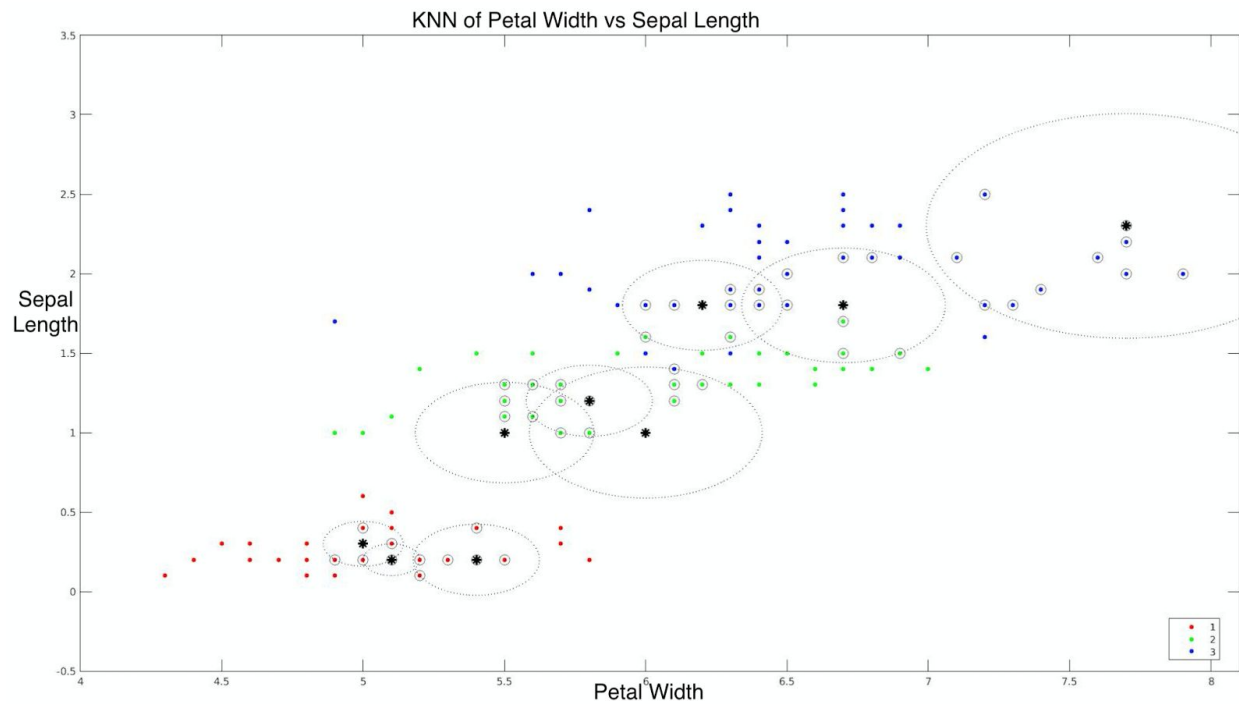


Figure 8: KNN of Petal Length and Sepal Width

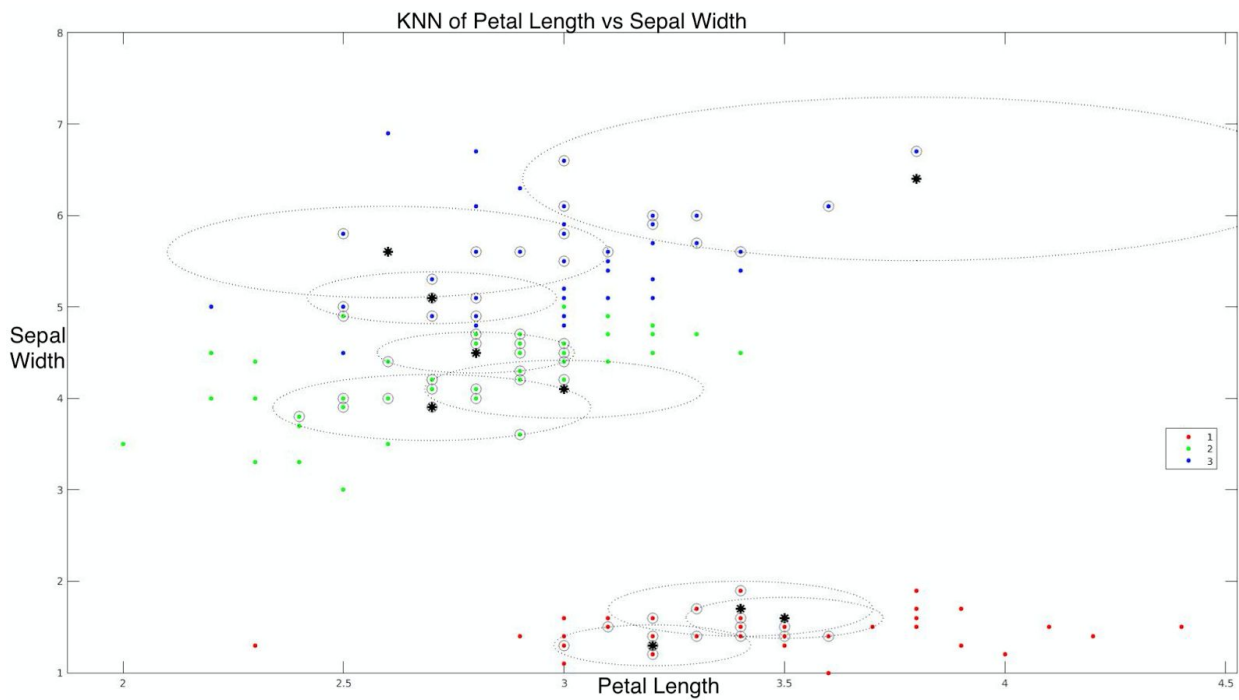


Figure 9: KNN of Petal Length and Sepal Length

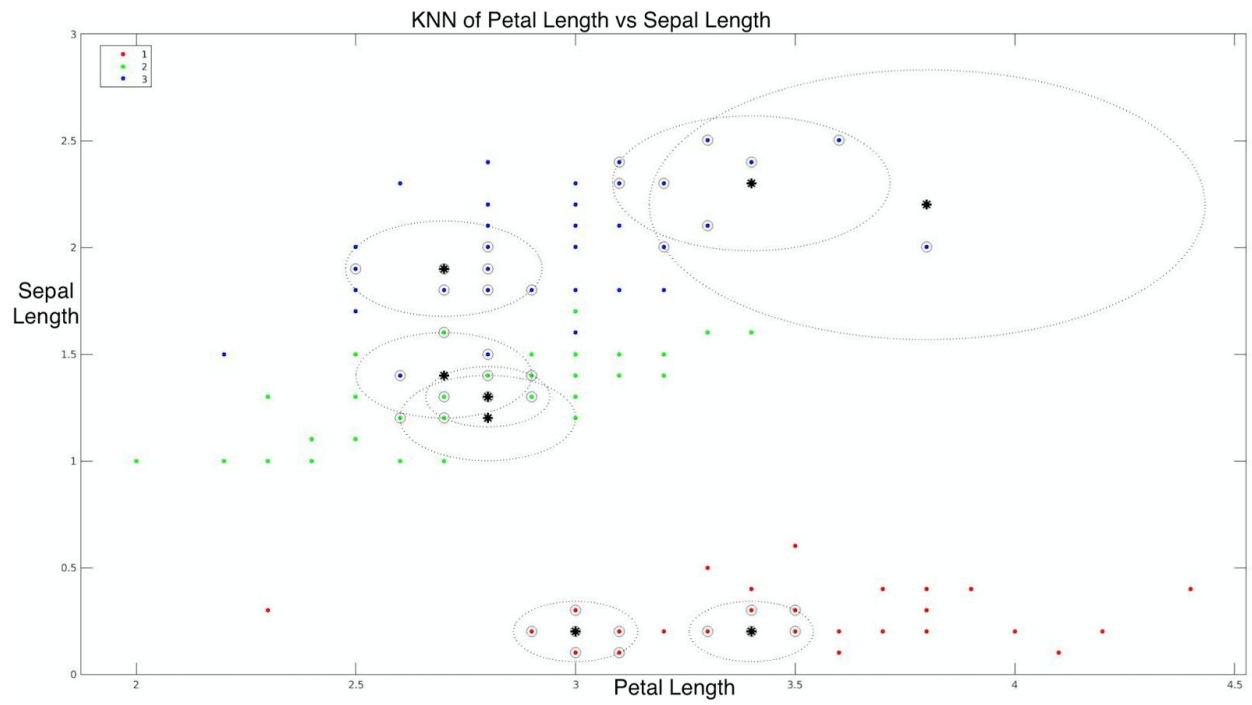


Figure 10: KNN of Sepal Width and Sepal Length

