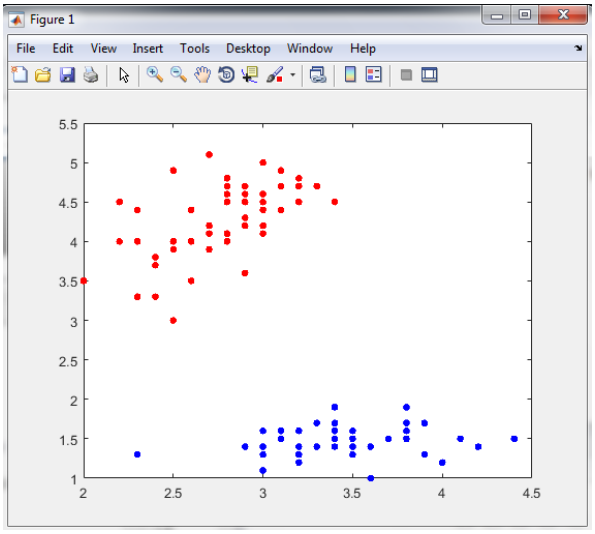
Data exploration

In order to complete the knn task effectively we must first explore the given data, the iris data set although not normalised, the range of each attribute is acceptable. Looking at the graphs of the attributes we can see a clear separation of classes, in most of the graphs which means our error percentage should be quite low.

V1	V2	V3	V4
Min. :4.300	Min. :2.000	Min. :1.000	Min. :0.100
1st Qu.:5.000	1st Qu.:2.800	1st Qu.:1.500	1st Qu.:0.200
Median :5.400	Median :3.050	Median :2.450	Median :0.800
Mean :5.471	Mean :3.094	Mean :2.862	Mean :0.785
3rd Qu.:5.900	3rd Qu.:3.400	3rd Qu.:4.325	3rd Qu.:1.300
Max. :7.000	Max. :4.400	Max. :5.100	Max. :1.800



Looking at the ionosphere data set, we can see the value of the second attribute is always the same therefore it will not contribute to our distance, if the data set was larger it would be best to remove this attribute in order to reduce the time taken for computation. The data set is also already normalised.

We can also see that neither data sets contain categorical attributes, which would be more difficult to deal with when using a distance measure.

Method

In order to complete this task I first separated the data set into training and test sets, I then computed the Euclidean distance from each point in the training set to 1 point in the test storing this as a column in a matrix, I did this calculation for each point in the test set in the same way as the built in Euclidean distance function.

I then combined the training set, the class labels and the distance associated into a new matrix in order to sort them by distance. To sort the data I picked the minimum distance and found the corresponding row in the matrix, placing this value in a new matrix followed by deleting that row from the matrix. I kept doing this until the new matrix was fully populated and the old matrix was empty.

I allowed the user to set the k value and then picked the first k rows of the new matrix, I used a built in function to count the frequency of each class and picked the class with the highest frequency to be the class label, for each point in the test set. I did not account for situations where there may be a draw although this is normally done at random.

Finally I compared the predictions with the true class labels, counting both the correctly classified instances and the incorrectly instances, and calculated the percentages associated with both.

Results

Iris:

After running my script for k=1 and k=3 on the iris data set I obtained the following results.

	Correctly classified	Misclassified
K=1	100	0
K=3	100	0

Although these results are highly unusual, looking at the plots for the attributes we can see this is a likely occurrence.

Ionosphere:

After running my script for k=1 and k=3 on the ionosphere data set I obtained the following results.

	Correctly classified	Misclassified
K=1	92.0530	7.9470
K=3	93.3775	6.6225

The ionosphere predictions seem to show an accurate portrayal of the classifier and increasing the value of k produced a better rate of correct classifications. I experimented with different values of k and found that when k=10 I produced the same amount of correctly classified instances.