Linearly and Non-Linearly Separable Data

Michael Reinhart
Student Number: 20001556
CISC 271

ABSTRACT

PURPOSE:

The purpose of this lab was to understand how to find clusters of data and then find good fitting hyperplanes for linearly separable and non-linearly separable data. The data given was for the iris flower data set and a data set visually clustered into the 4 quadrants of a 2 dimensional graph.

METHODS:

The methods I used for the already separated data was first a kmeans clustering method which separates the data into k clusters based on location relative to the other data points. Then I used a perception algorithm to find the hyperplane and an SVM method to again to find a hyperplane to see which one works better. For the non-linearly separated data I used embedding to separate the data and then the SVM method to find a hyperplane.

RESULTS:

The results were quite good, as I found clear hyperplanes that separated the data well as well as the embedding algorithm used made the data very easy to separate. However the kmeans clustering was not very effective of grouping the data set correctly.

CONCLUSIONS:

The SVM method to find a hyperplane looks much better as it creates a plane that is equidistant to the sides of the clusters. Also the non-linearly separable data separated very clearly using embedding.

INTRODUCTION

For this lab I was given a data set of the Iris Flower data the goal was to find a way to cluster the data. I did this using kmeans clustering. After I used the perceptron algorithm to develop a good hyperplane that splits the two correctly clustered data clusters such that each side of the plane has only positive labeled data points or negatively labeled data points. Then I used a different method called a SVM to again find a hyperplane and compared the two different methods and results.

I was also give a data set that was not linearly separable with the same goal of separating the data into two distinct clusters that could then be used to find a good fitting hyperplane. To make the data linearly separable I used a method called embedding from there I found a hyperplane using a SVM.

METHODS

The first method used was called kmeans. kmeans takes the data given and separates it into k clusters which in this case was only 2. It clusters based on picking a point per cluster and based on the Euclidean distance between the two points either adds it into the cluster or not.

The second method used was called the perceptron algorithm. This algorithm works by looping through the data and checking to see if it was correctly classified. If it is not then it subtracts the values of the misclassified data from a weight vector that is used to make the

hyperplane. After there is no more misclassified data the weight vector left over will be the hyperplane that separates the data into two groups successfully. If the data given to the algorithm was not separable then the algorithm would run forever because every iteration would not bring it closer to finding a plane that separates the data when one does not exist.

The last method used in the first part of the assignment was the SVM or support vector machine. This finds the hyperplane and puts in so that is is equidistant between two support vectors which are the two closest vectors on either side of the clusters to the hyperplane.

The last method I used was embedding which is a method to separate linearly inseparable data by adding a dimension that will uniquely separate the data into two distinct clusters. My embedding method consisted of taking x1 * x2 to get x3. This would make the data is on either the positive or negative side of the z-axis based on the data given.

RESULTS

The number of wrongly classified data is: 23
These are the data points that have wrong data labels

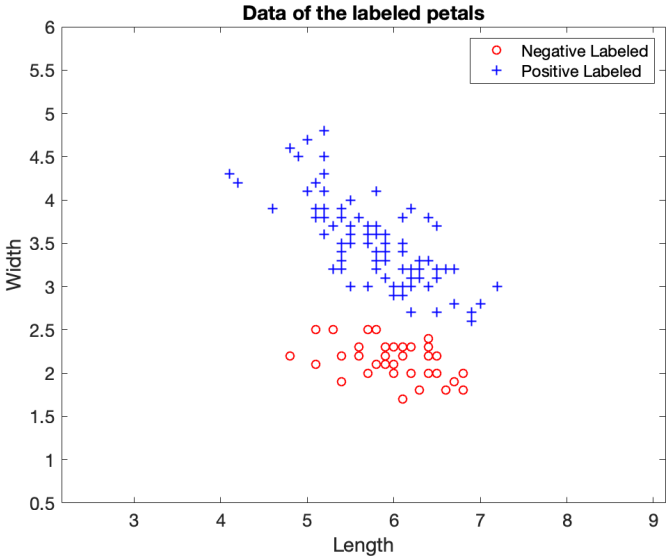
wrongClassIndex =

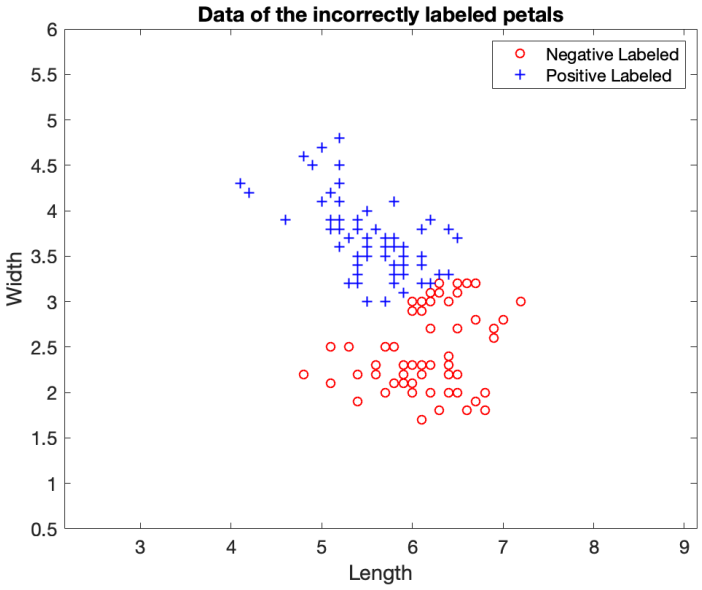
6.0000	**2.9000**
7.2000	**3.0000**
6.3000	**3.1000**
6.0000	**2.9000**
6.0000	**3.0000**
6.6000	**3.2000**
6.1000	**2.9000**
6.3000	**3.2000**
6.7000	**3.2000**
6.2000	**2.7000**
6.2000	**3.1000**
6.6000	**3.2000**
6.1000	**3.0000**
6.5000	**2.7000**
6.5000	**3.1000**
6.4000	**3.0000**
6.5000	**3.2000**
6.2000	**3.1000**
6.2000	**3.0000**
7.0000	**2.8000**
6.9000	**2.6000**
6.7000	**2.8000**
6.9000	**2.7000**

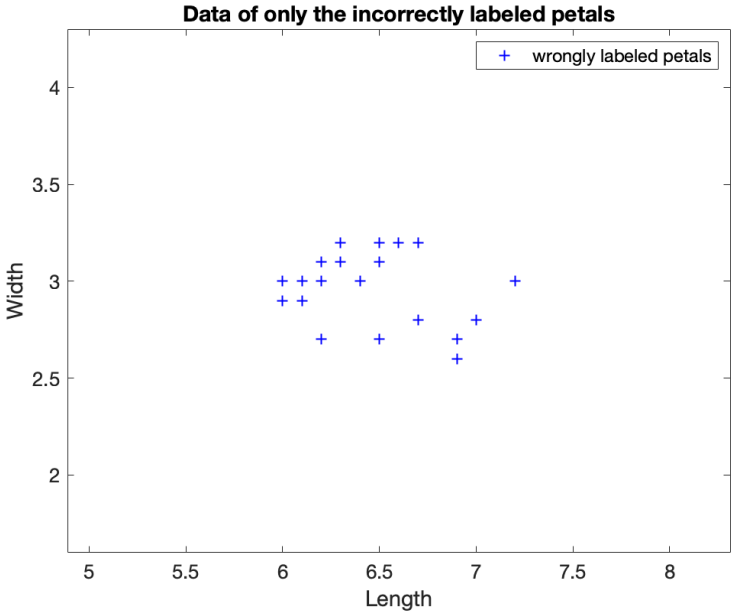
These are the indicies that have wrong data labels

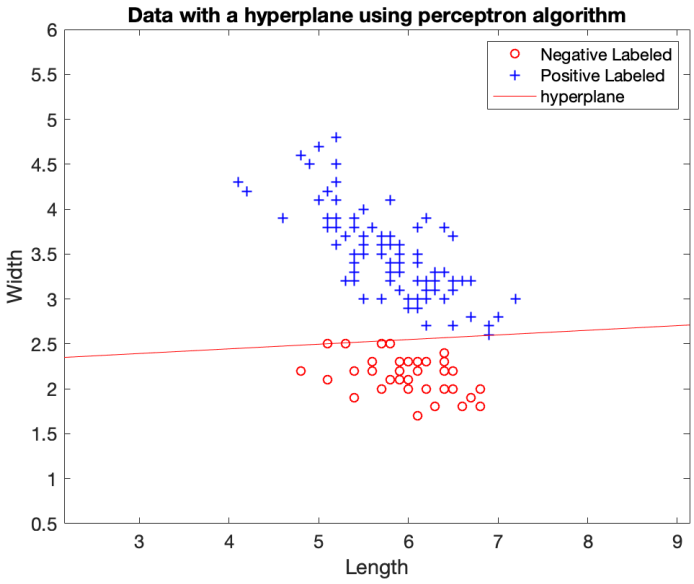
DataOfWrong =

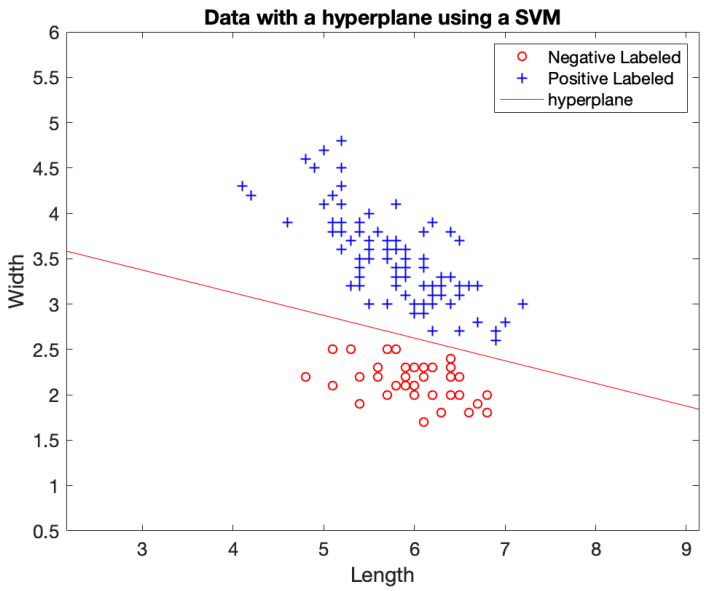
1
5
9
14
27
29
33
39
40
42
48
51
53
72
91
93
102
108
124
128
139
143
149

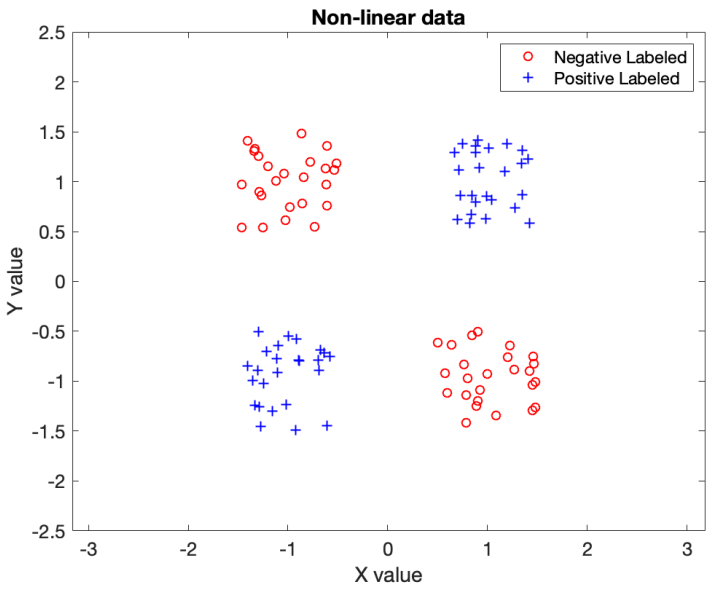

Data of the labeled petals
Negative Labeled
Positive Labeled
Width
Length

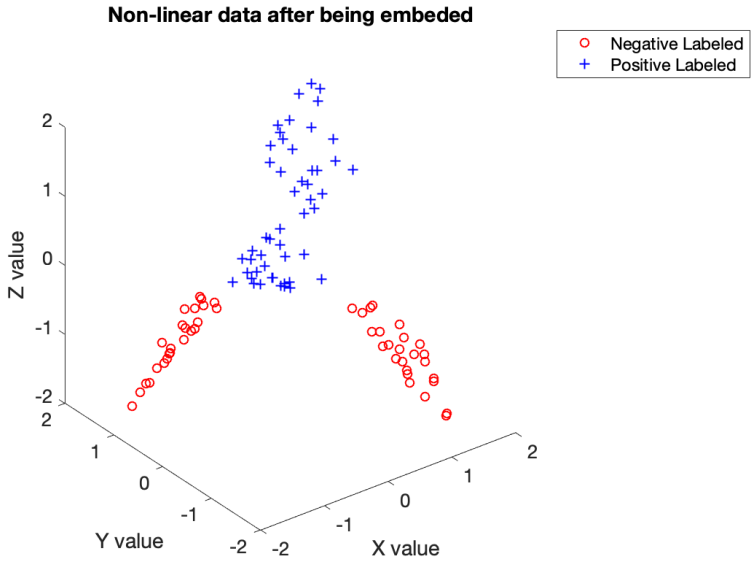

Data of the incorrectly labeled petals
Negative Labeled
Positive Labeled
Width
Length

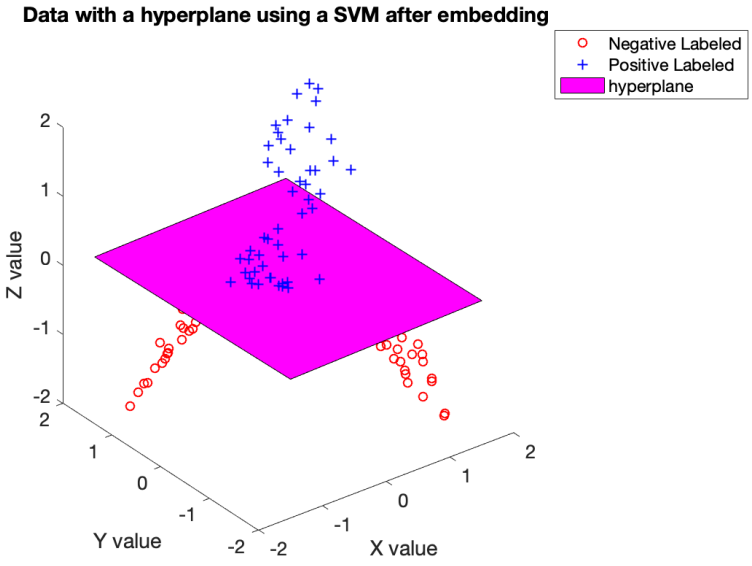

Data of only the incorrectly labeled petals
wrongly labeled petals
Width
Length


Data with a hyperplane using perceptron algorithm
Negative Labeled
Positive Labeled
hyperplane
Width
Length


Data with a hyperplane using a SVM
Negative Labeled
Positive Labeled
hyperplane
Width
Length


Non-linear data
Negative Labeled
Positive Labeled
Y value
X value


Non-linear data after being embeded
Negative Labeled
Positive Labeled
Z value
Y value
X value


Data with a hyperplane using a SVM after embedding
Negative Labeled
Positive Labeled
hyperplane
Z value
Y value
X value

DISCUSSION

The kmeans clustering data was not very successful and misclassified 23 data points. This is because to find the clusters it only decided based on the data points around it instead of on the point globally. This leads to clusters not starting in the correct spots which makes uneven clusters when visually there is an obvious better clustering.

The perceptron algorithm worked well but was not as pleasing of a result as the SVM. The perceptron algorithm was interesting to see develop as each iteration of the loops improved the weight vector until it was perfect and did not allow any misclassifications of the data. There is danger of using this algorithm on data that is not linearly separable due to it continuously looping for an answer that does not exist.

The SVM method for finding hyperplanes is also better than the perception algorithm due to it making the hyperplane perfectly in-between the two clusters based on the support vectors on each side of the hyperplane.

Embedding also worked very well but this is due to the data given being very easy to separate. The data was split between being positive or negative across the z-axis which was an added dimension that was made to do the embedding.

REFRENCES

[1] IEEE: MathWorks. Linked from
https://www.mathworks.com/matlabcentral