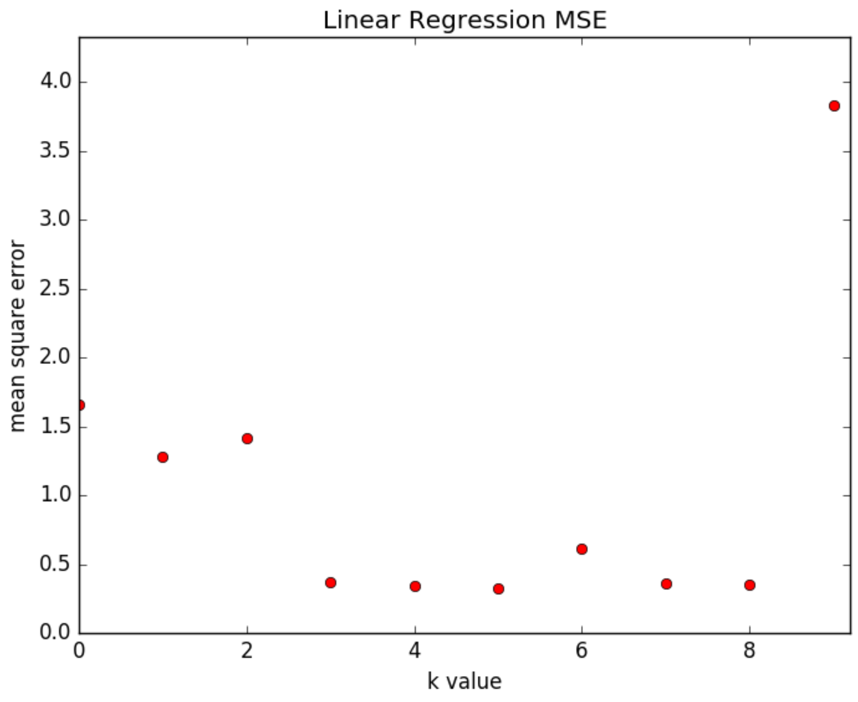
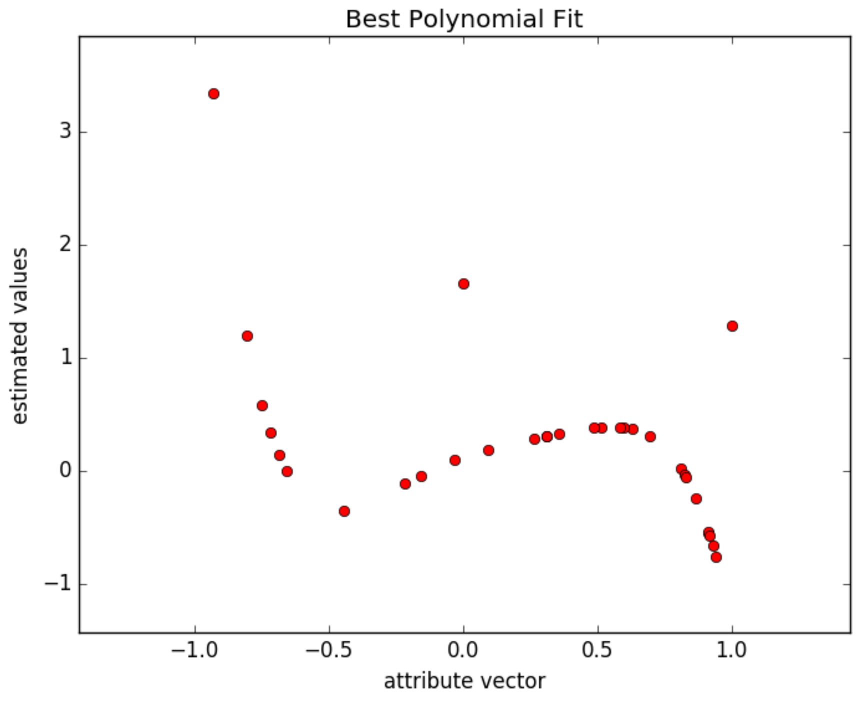
Problem One

A. Implemented using n = 3

I chose n=3 in order to balance the variance within both the training and testing sets. Because 30 data points is not that large of a sample, I wanted to ensure that both training and testing had a sufficient number of data points to provide grounded statistics.



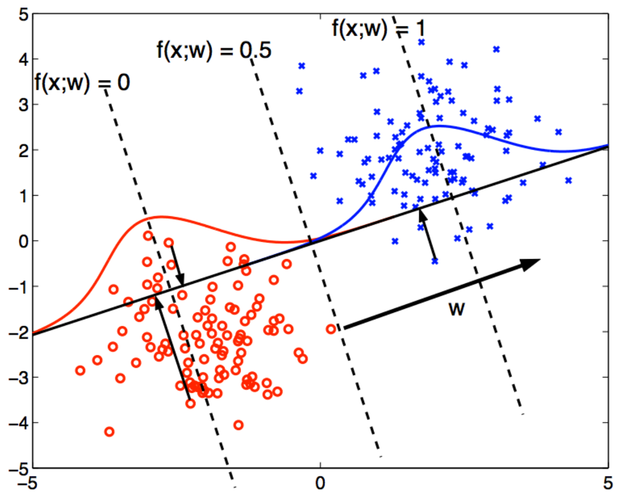


B. The k that yielded the best result was 5 with the lowest error rate 0.325665168273.

C. If this model was given a new query x=3, I would be concerned about the accuracy of the prediction. The best-fitted line is based off of a training data set X where all |x| ∈ X is less than or equal to one. In turn, the value 3 is far beyond what the best-fitted line was trained with. As a result, predicting x=3 under this model would be assuming that the function's behavior accurately extends beyond the training set's range. While this trend may hold true for values beyond the given range, making the best-fitted line a good predictive measure, it is likely that the model is specific to the training set. Thus, I would be tentative in applying this best-fitted function to values far beyond the training range, such as x=3.

Problem Two

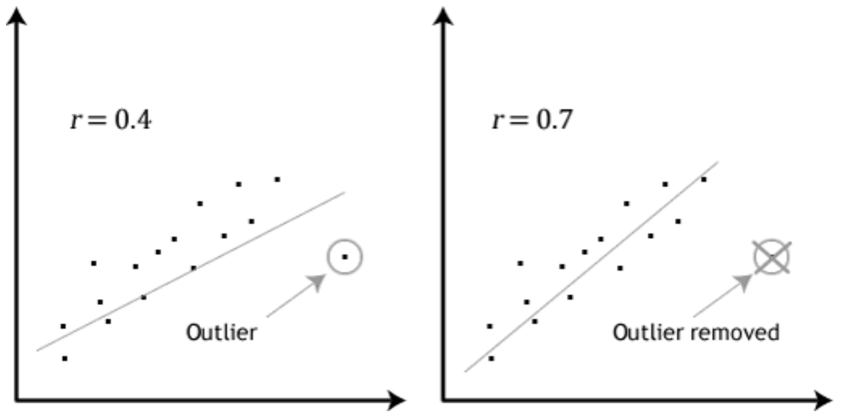
A. Regression can be utilized in order to classify data. Classification implies that the resulting variable is casted as a class. In turn, the way to classify using regression is that you label each class a number, formulate a best-fit regression line, and then round all output to their closest label number. For the sake of this explanation, we will assume we are in a binary system such that the outputted variable must be classified as a 0 or 1. Regression results in a continuous variable. Meaning, a value will be plugged into a polynomial function that will return a label that is not necessarily 0 or 1. In order to deal with this continuity, one must leverage a decision boundary. A decision boundary will indicate whether or not the input should be classified as 0 or 1 based on the outputted value. If there are several classes, then the best approach is often to round the resulting number to the closest number associated to a given label. In a binary space, an example of a decision boundary is if f(x,w) > .5 then it will be classified as 1, otherwise it will be classified as 0.



Source: http://www.ai.mit.edu/courses/6.867-f04/lectures/lecture-5-ho.pdf

In the above graph, the decision boundary is based on .5. When the polynomial produces values greater than .5, the input is classified as "blue", otherwise it is classified as "red". Another concept would be if 1=dog, 2=cat, and 3=mouse. If the regression outputs 2.7, it would be classified as a mouse, but if it outputs .2 it is classified as a dog. In the end, classification via regression takes results of a best-fit function and essentially casts them in the class with the closest associated numeric label.

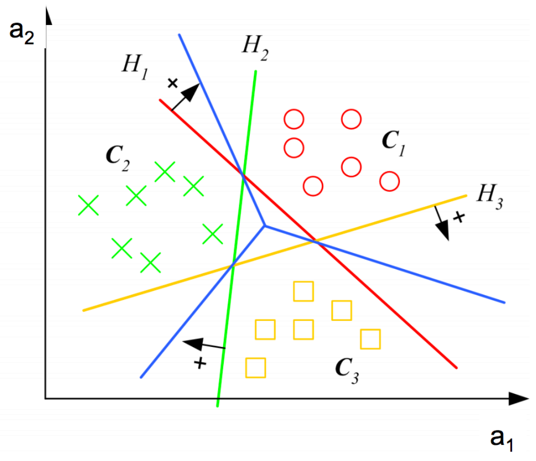
B. The regression algorithm is dependent on the relationship between inputs and estimated values. If there isn't a strong correlation between the two values, then this approach will be highly compromised. Similarly, if there are a few outliers, they will greatly affect the best-fit function, and in turn compromise the accuracy of the regression's predictions. The following graph clearly demonstrates this notion. With all of the data (on the left) the best-fit function takes the outliers into account, which alters the slope away from the trend of the majority of data points. When the outliers are taken out, the best-fit function estimates values much more accurately. In turn, being at the will of outliers is a weakness of classification via regression. Similarly, if data cannot be best-fit using a linear model, a regression algorithm would not be a good approach.



Source: https://statistics.laerd.com/statistical-guides/pearson-correlation-coefficient-statistical-guide-2.php

Problem Three

A. When there are more than 3 classes to be distinguished, LDA is more accurate in cases where the data cannot be separated linearly. In ordered to be separated linearly, there must be singly connected convex regions. When this is not the case, the data can still be separated using n discriminant functions, in turn separating the space into n convex regions. An example of an instance where data is not linearly separable, but can be separated under the LDA model is as follows.

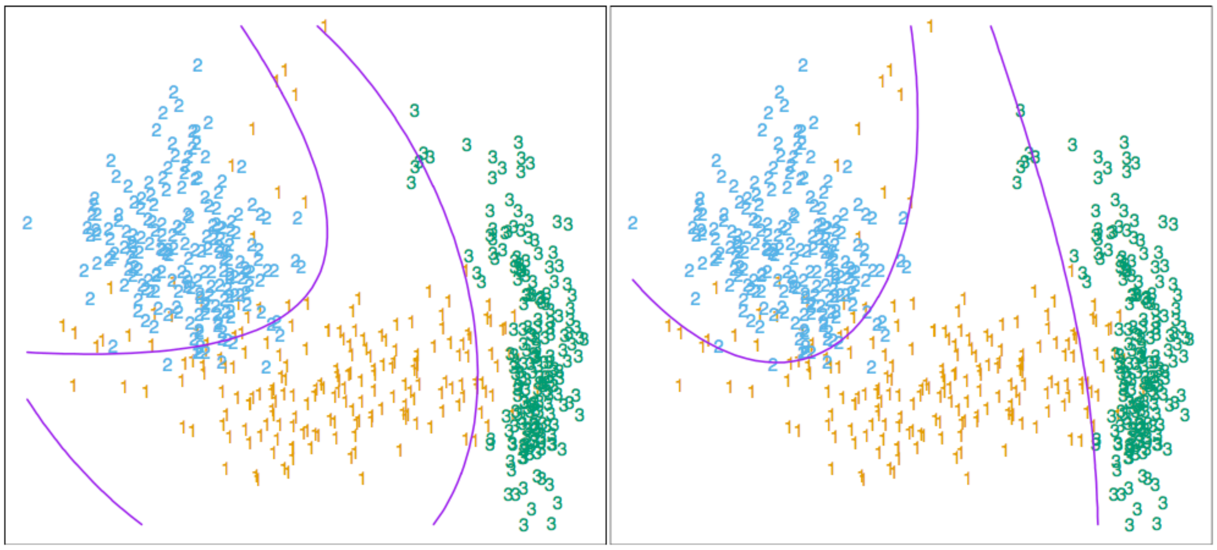


Source: Class Linear Discriminants Lecture Slides

B. The LDA method assumes that the classes are normally distributed, meaning there are Gaussian densities. Thus, the method's effectiveness is rooted in the assumption that classes share the same covariance matrix. Due to this assumption, as explained on pg.127 of the textbook, the normalization factors and quadratic exponents cancel, making each decision boundary linear.

C. According to the book, QDA is very similar to LDA. However, in QDA, a different covariance matrix must be calculated per class. This is contrary to LDA, which assumes all classes share the same covariance matrix. The downside of this assumption is that when p is large, the number of parameters greatly increases under the QDA model. The decision boundaries are rooted in the parameters' densities, making the model a bit more complex. A large reason LDA and QDA are so successful with large data sets is that they have extremely stable predictions due to their Gaussian models and simplicity of boundaries (linear and quadratic). This advantage is slightly lost in QDA, as it does not assume one shared covariance matrix, and the number of parameters drastically increases.

D.



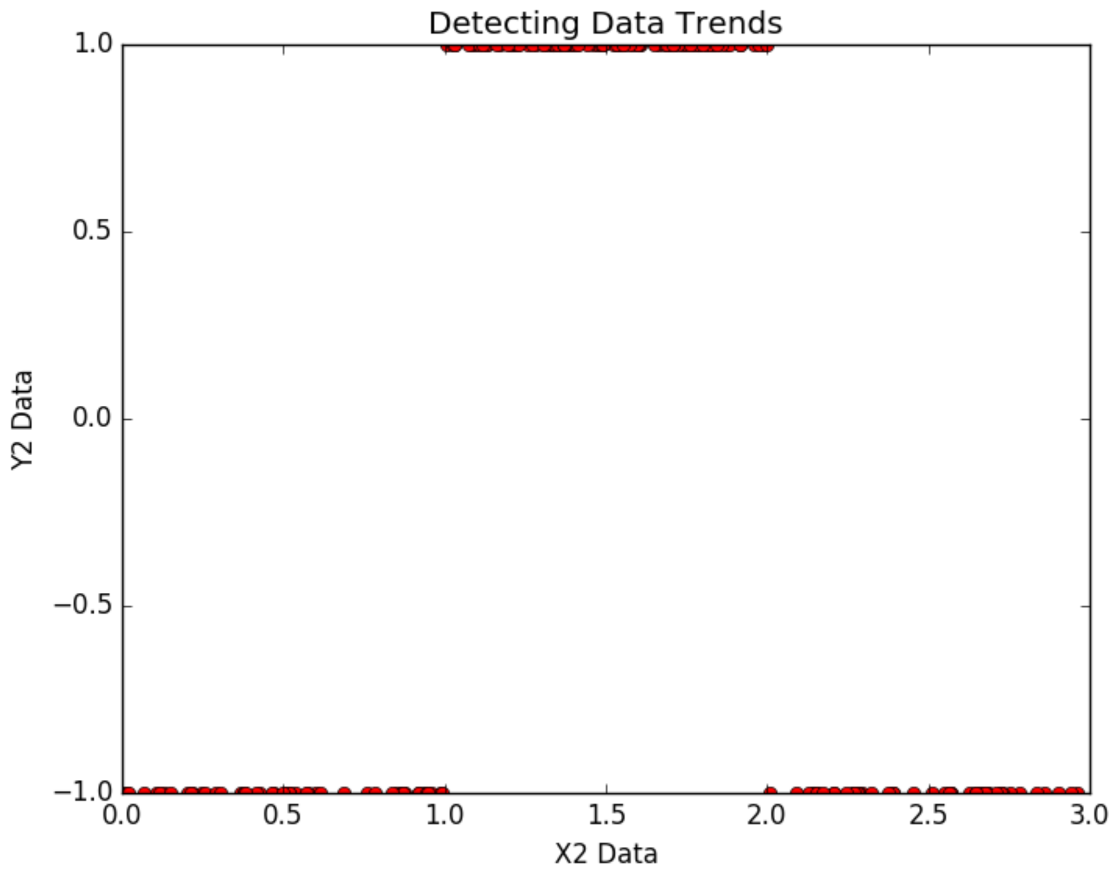
Textbook, pg. 111

In order to find a decision surface that can be modeled with a polynomial using LDA and not QDA, one must find a transformation that enables the data to be linearly separable. As shown in the above diagram (QDA represented on the left and LDA on the right) data that is not linearly separable in its original raw form, can still be represented with LDA. In the above scenario, the data is transformed to be in a five dimensional space, and as such can be divided appropriately. As such, you can determine a decision surface that can be modeled utilizing LDA but determining the appropriate transformation that needs to be applied on the data set.

Problem Four

A. It took 6 epochs to correctly classify X1.
The parameter vector is: [-11.0, 4.01152]

B. Using the same sequential algorithm as in part A, the data cannot be correctly classified. In turn, the program keeps running the while loop over and over again, and does not terminate with a solution. Because the perceptron algorithm is a linear classifier, my instinct was that the data does not fit this model, and that the algorithm needs to first apply transformation to be linearly separable. In order to affirm this hypothesis, I plotted X2, Y2 data to detect the general pattern (graph below: X2Data.png). After plotting these data points, it is clear that this dataset is not linearly separable. The code for the graph below can be found commented out at the bottom of perceptrona.py.



C. It took 685 epochs to correctly classify X2.
The parameter vector is: [-49.0, 73.3556525, -24.44140367]

In order to classify X2, we have to make it linearly separable. In turn, I transformed the data by making it two-dimensional. As a result, the w vector now has three elements, and the second dimension of x is x^2. Once this transformation took place, the data shifted into a linear graph with a positive slope, where the data was partitioned on either side of a given point. As a result, this transformation enabled the set to be linearly divisible and classifiable under the algorithm.