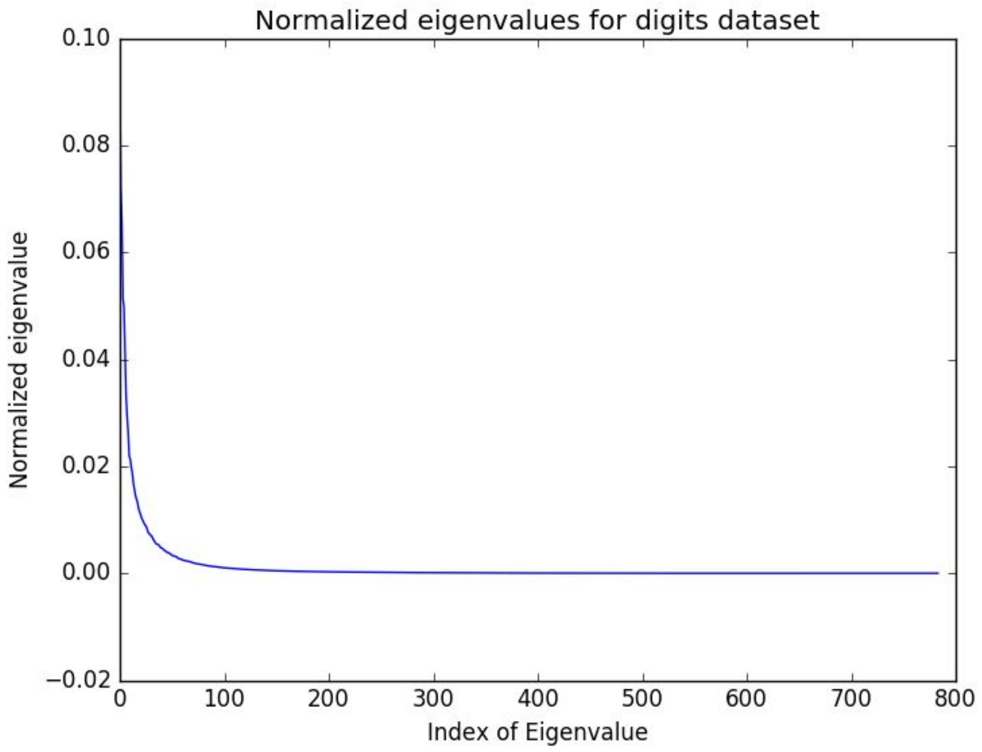
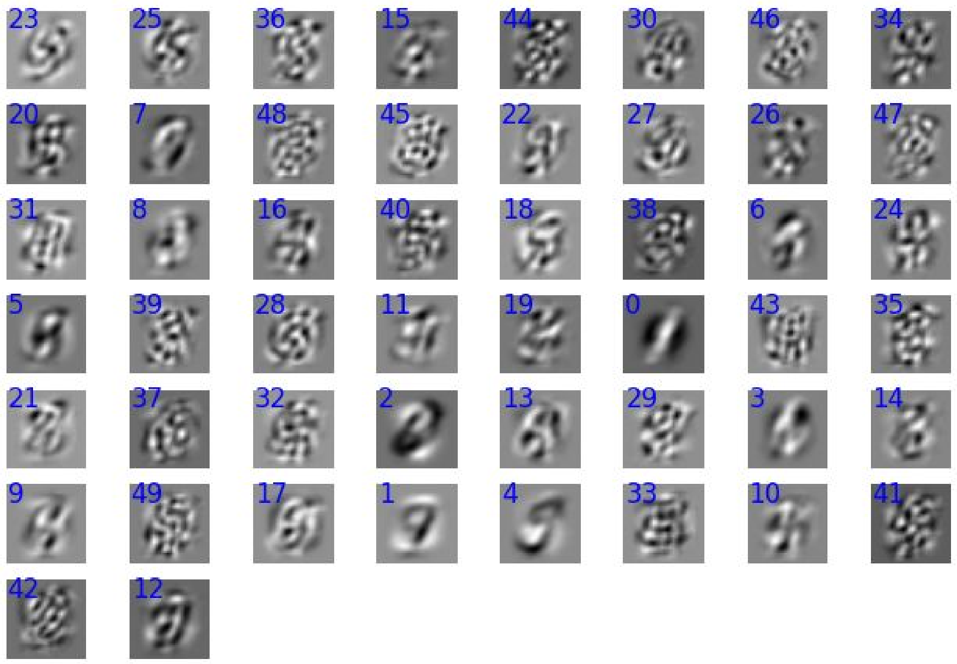
WU1: Plot of the eigenvalues for the first dataset is depicted below

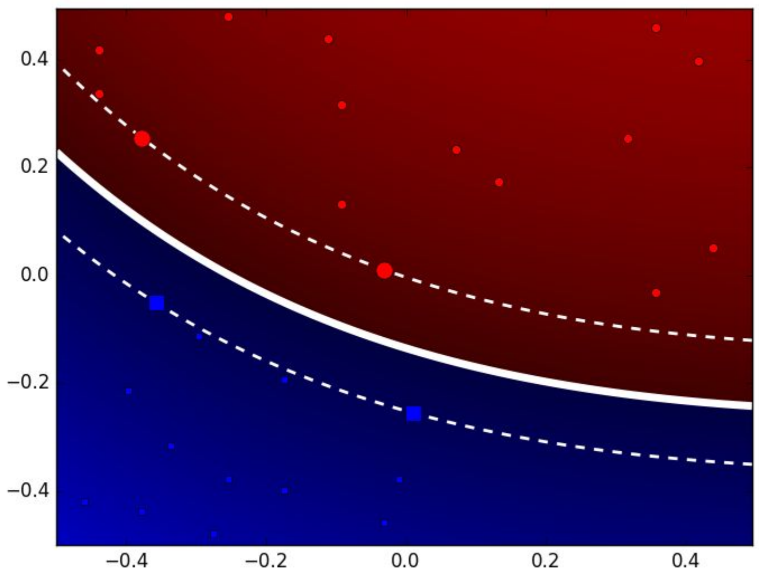


In order to determine how many eigenvalues are needed to account for 90% or 95% of the variance, we first compute the sum of all the eigenvalues, which is total variance. Then we compute the cumulative sum of the eigenvalues and divide all the cumulative sums by the total variance. Then, we find the first index such that the percent variance exceeds .90 or .95. Carrying out this procedure using the same eigenvalues that appear in the plot above, we find that 82 eigenvalues are needed to account for 90% of the variance and 136 eigenvalues are needed to account for 95% of the variance.

WU2: The digits data has been plotted below. Some of the images appear to faintly resemble digits. This result is expected because by projecting the data onto its 50 most “significant” parts, we extract information that is only a partial description of the data. Using the method described in WU1 above, we have plotted data that represents roughly 82% (top 50 eigenvalues) of the variance in the data. So, it is understable that the image data appears fuzzy since we are not representing the finer details of the data.

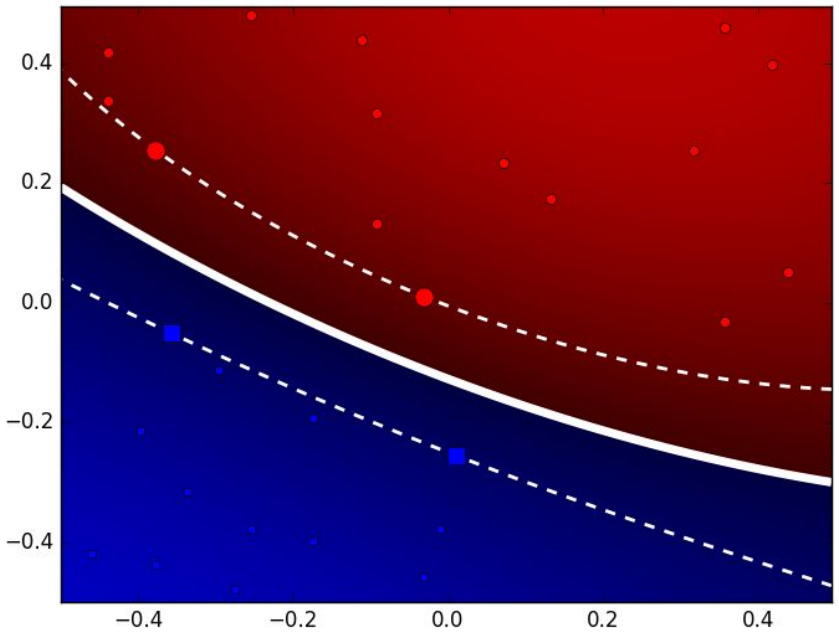

23 25 36 15 44 30 46 34
20 7 48 45 22 27 26 47
31 8 16 40 18 38 6 24
5 39 28 11 19 0 43 35
21 37 32 2 13 29 3 14
9 49 17 1 4 33 10 41
42 12

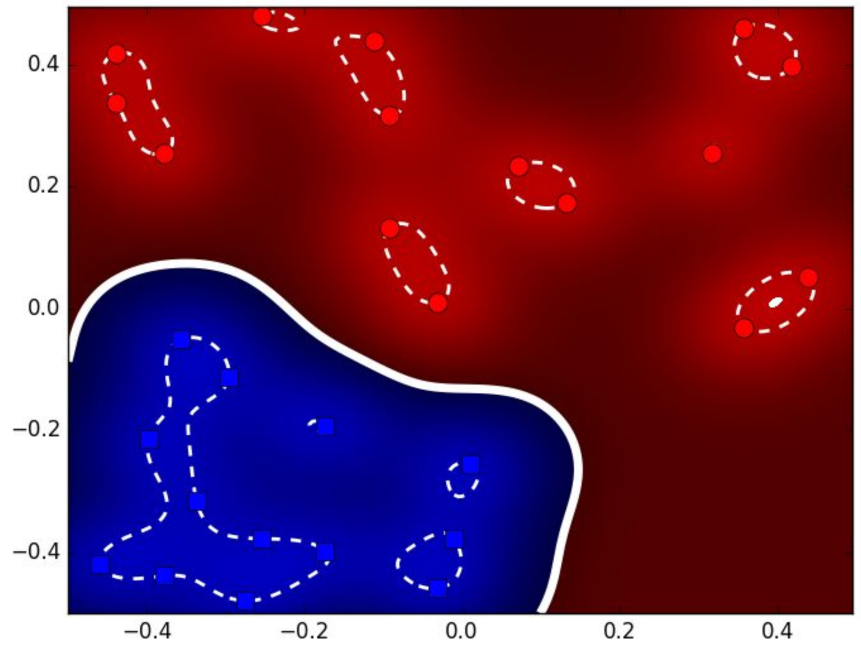
WU3:


0.4
0.2
0.0
−0.2
−0.4
−0.4
−0.2
0.0
0.2
0.4

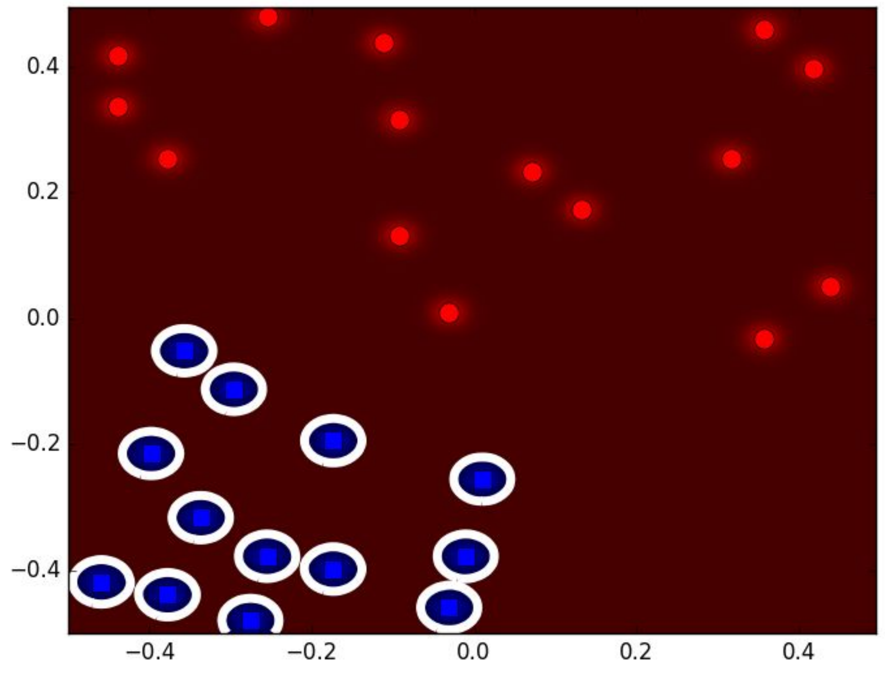
Yes, fewer than 3 support vectors is possible, but the number of support vectors must be odd. If in the case of two support vectors, there would be a middle space in which examples would be unclassified. With 1, or just the decision boundary, examples can still all be classified.

WU4:





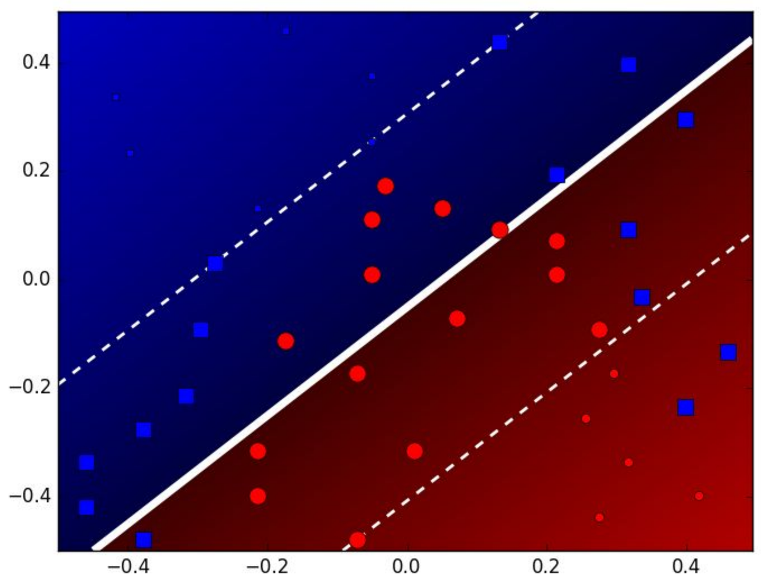
Above: Initial, g=100



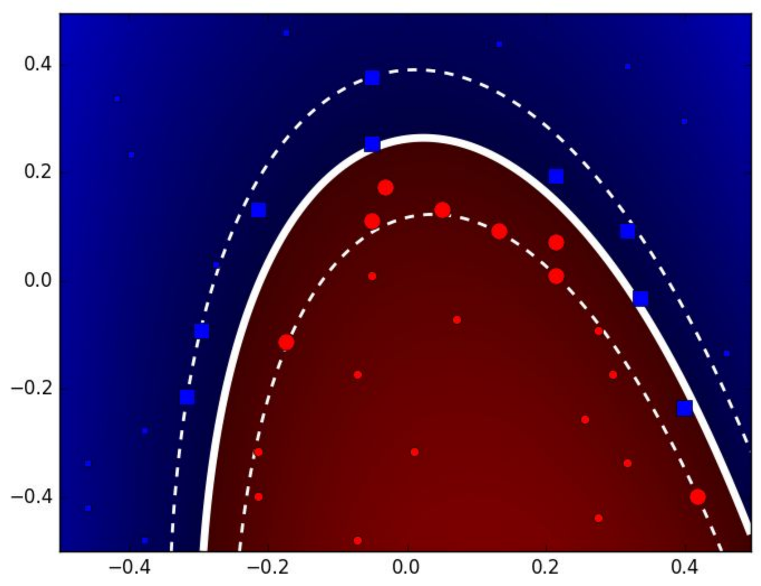
Above: Final, g=2200

The little blobs result from increasing the gamma, which helps discern and create decision boundaries. A higher gamma implies higher sensitivities for the algorithm for classifying examples, which can overfit the data. With a lower gamma, “the blob” was connected between the blue examples, and points between said examples can still be classified as the same. With a decision boundary around a single example, the inbetween spaces to not classify as the same as the larger blob. Through incremental trial and error, a gamma of roughly 2200 separated the examples in the data to individual decision boundaries.

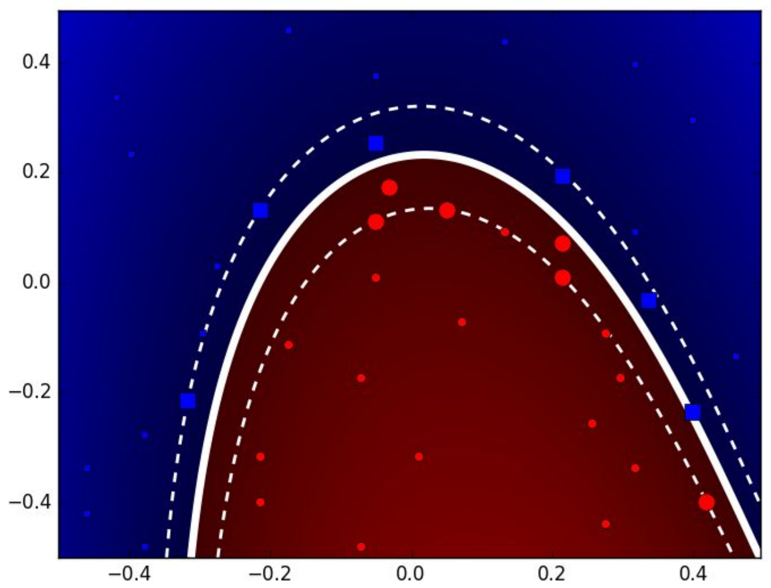
WU5: There are a lot of red support vectors on the blue side of the decision boundary because this dataset is not linearly separable. Because it is not linearly separable, the decision boundary is not able to perfectly divide the classes. So, in order to minimize loss, the algorithm must allow many red data points to be on the wrong side of the data set.



WU6:Based on the data, the 0/1 loss on the training data is 1, not 0. This is because there is one blue data point, located in the lower right of the plot below, which lies on the red side of the decision boundary. This point is classified incorrectly, so a loss value of 1 is incurred because the total loss is the sum of the losses of each example. The hinge loss on the training data is also nonzero. The same blue point above is classified incorrectly, so the true (correct) label and the (incorrect) label outputted below are of opposite signs, so the value of 1 minus the product of the predicted label and true label is positive, making the loss nonzero.



WU7: A gamma value of 0.80 is the smallest value for a good decision boundary because it is the smallest value where all the support vectors are on the correct side of the decision boundary. So, the algorithm does not have to take the penalty for moving points to the wrong side of the decision boundary in order to minimize the loss.


0.4
0.2
0.0
−0.2
−0.4
−0.4
−0.2
0.0
0.2
0.4