Assignment 12

Alberto Mejia
RIN: 661514960

CSCI 4100 - Machine Learning from Data

December 8, 2019

1. (300) Neural Networks and Backpropagation

(a) Backpropagation
D := Dataset
$E_{in}(w) = \frac{1}{4}(h(x, w) - y)^2$ when $|D| = 1$
Identity
Layer 1: [[-0.032, -0.032], [-0.032, -0.032], [-0.032, -0.032]]
Layer 2: [[-0.216], [-0.137], [-0.137]]

$\theta(s) = \tanh(s)$
Identity
Layer 1: [[-0.027, -0.027], [-0.027, -0.027], [-0.027, -0.027]]
Layer 2: [[-0.18], [-0.114], [-0.114]]

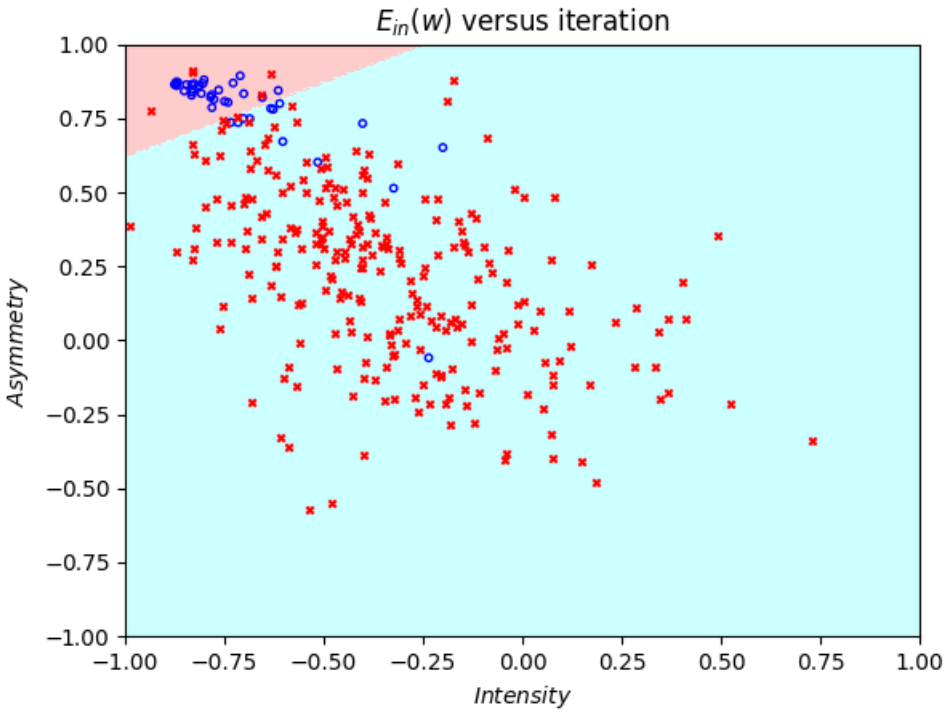
(b) Gradient Approximation
D := Dataset
$E_{in}(w) = \frac{1}{4}(h(x, w) - y)^2$ when $|D| = 1$
Identity
Layer 1: [[-0.032, -0.032], [-0.032, -0.032], [-0.032, -0.032]]
Layer 2: [[-0.216], [-0.137], [-0.137]]

$\theta(s) = \tanh(s)$
Identity
Layer 1: [[-0.026, -0.026], [-0.026, -0.026], [-0.026, -0.026]]
Layer 2: [[-0.17], [-0.113], [-0.113]]

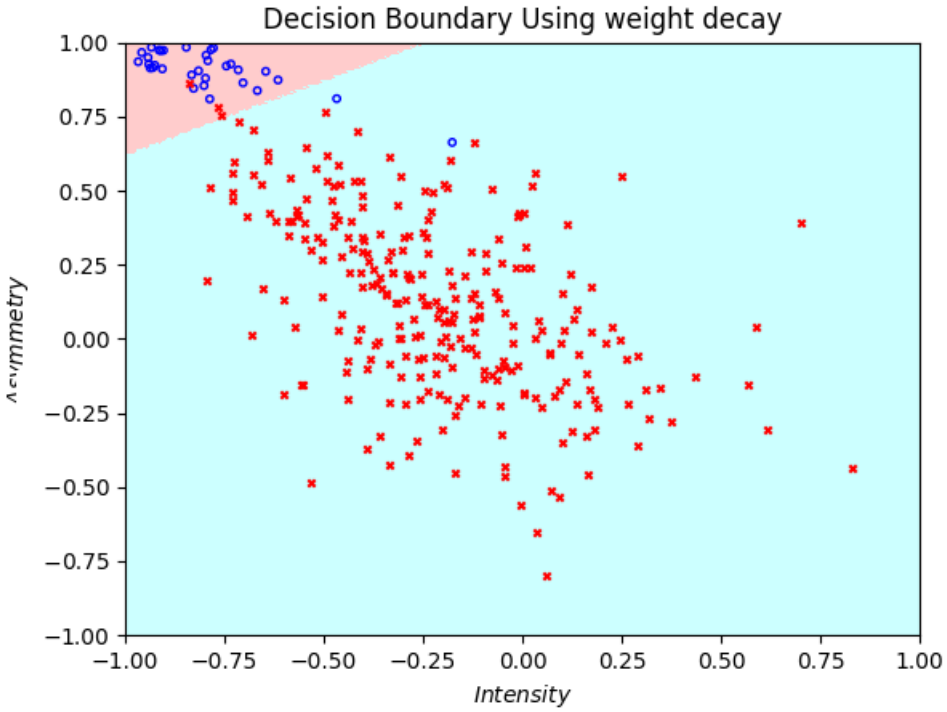
Overall, there is not much change when comparing the two. This helps us verify our backpropagation gradient calculation as these results are supposed to be similar. If they weren't, then there would be something wrong with our backpropagation gradient calculation.

2. (600) Neural Network for Digits

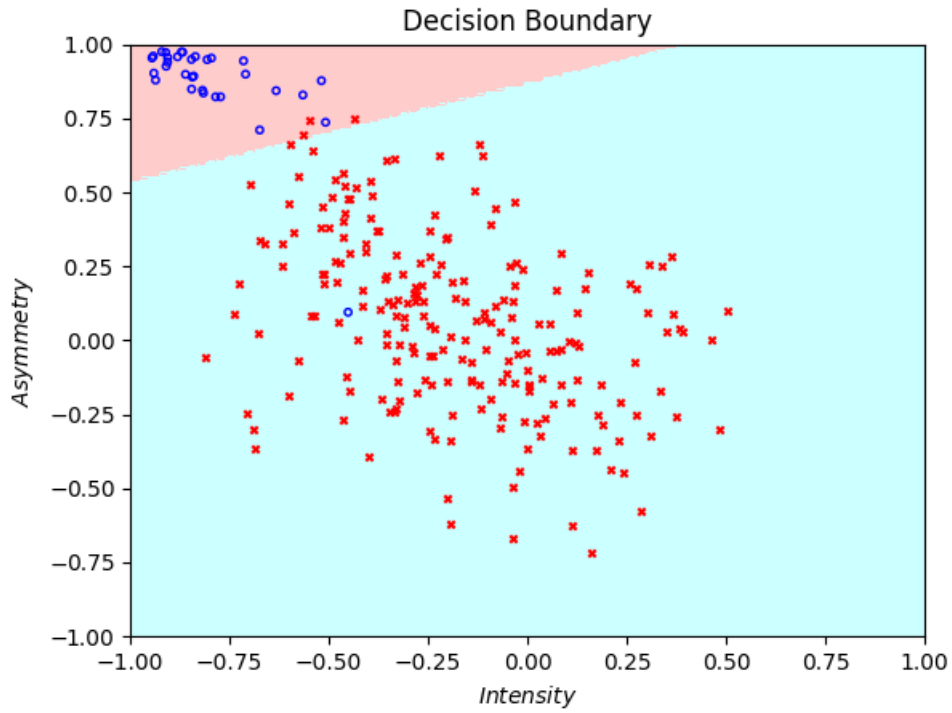
(a)



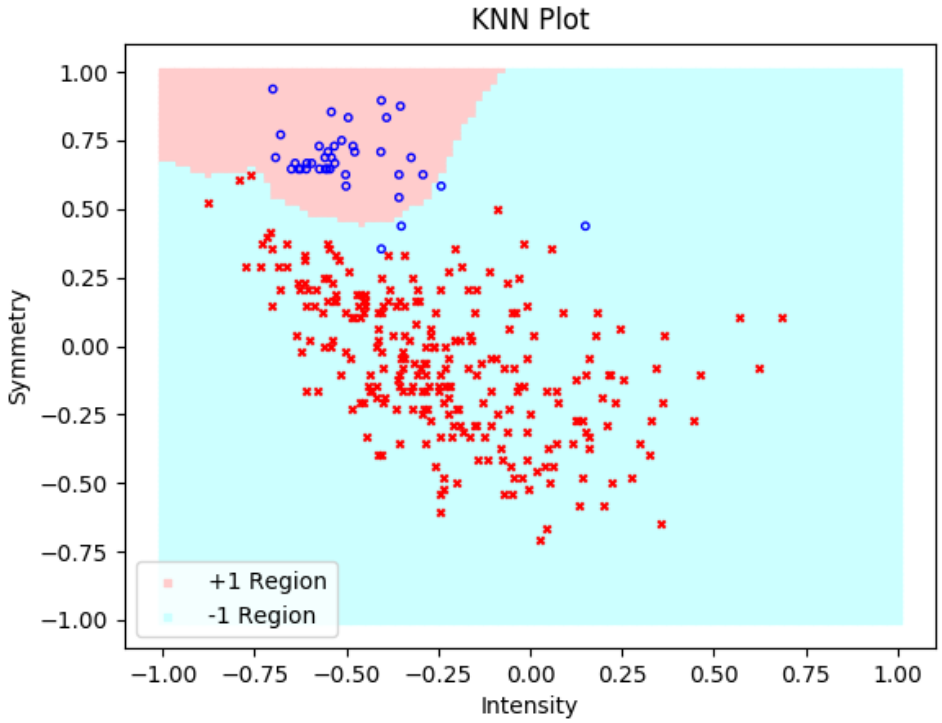
(b)



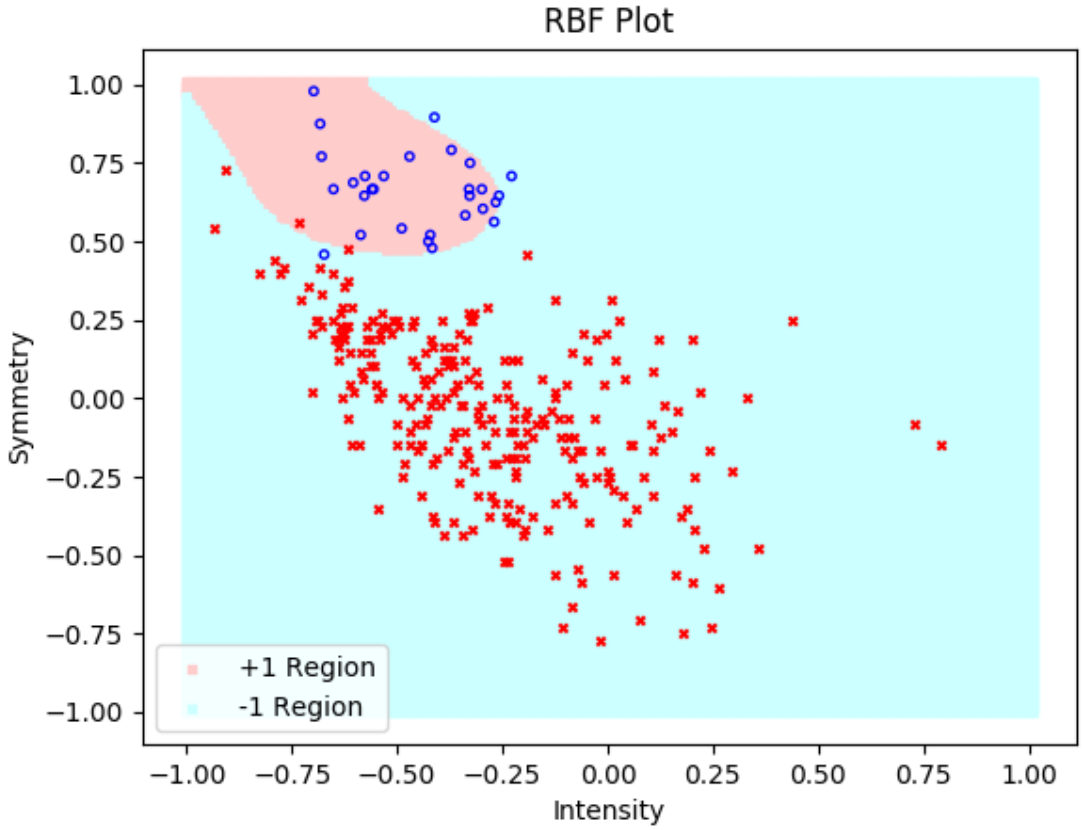
(c) Early Stopping



Using the two features and data developed in a previous assignment and the 300 training data points selected in assignment #9:



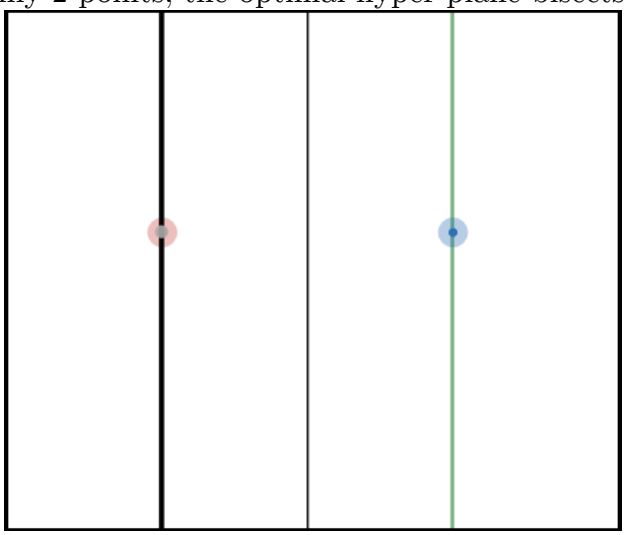
$E_{test} = 0.012891753723049567$



$E_{test} = 0.007890642364969993$

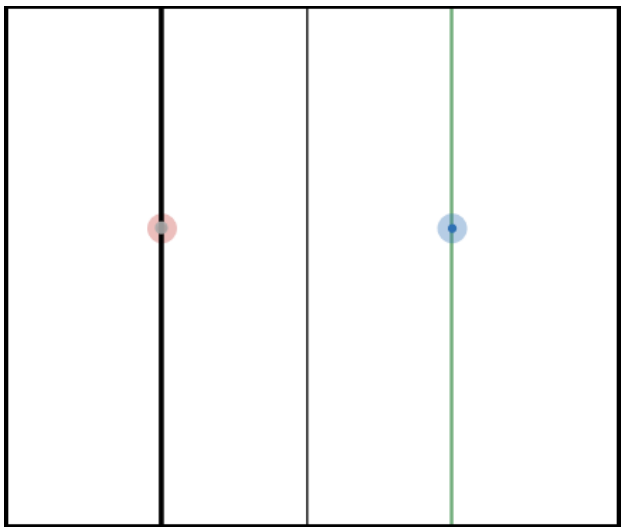
3. (300) Support Vector Machines

(a) Since there are only 2 points, the optimal hyper-plane bisects the origin.



Red: (-1, 0)
Blue: (1,0)
Hyperplane: (0,0)

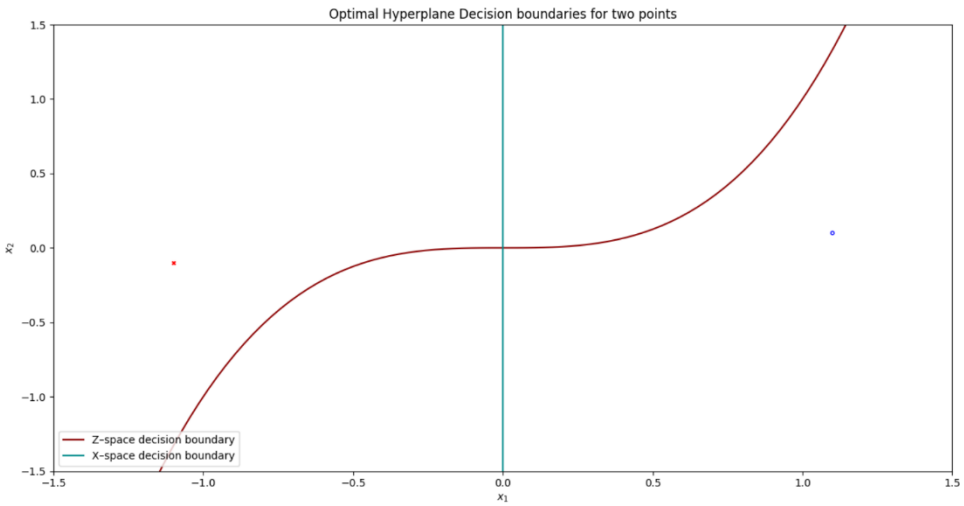
(b)



Red: (-1, 0)
Blue: (1,0)
Hyperplane: (0,0)

The data points and the hyperplane look identical. That is because our X plane is the same in Z direction, giving us a line $Z_1 = 0$.

(c)

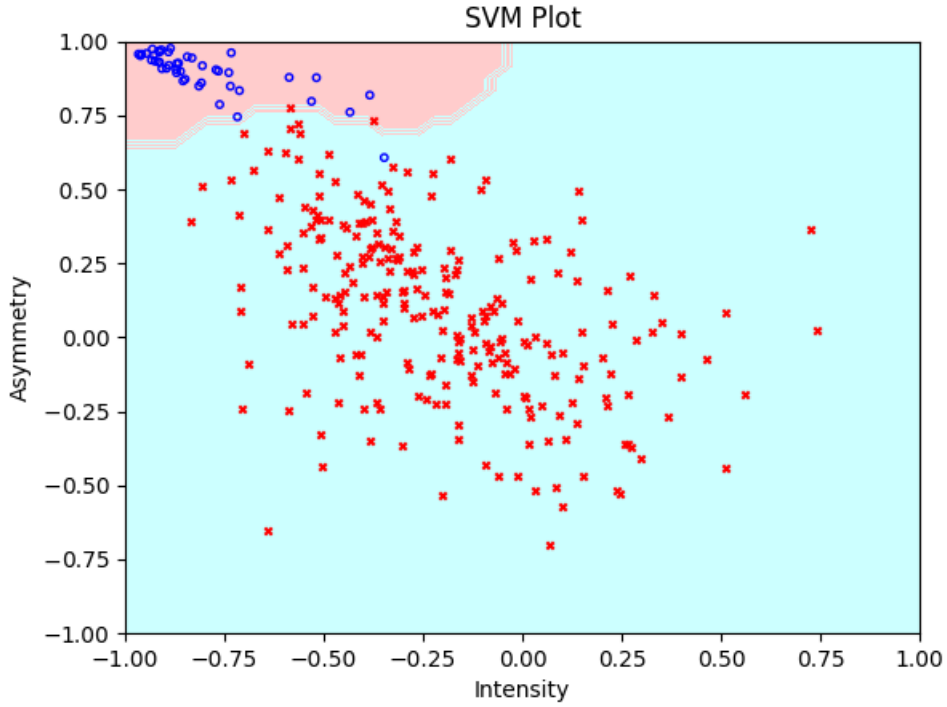


(d) Expression for kernel function in terms of x and y
$K(x,y) = (x_1^3 - x_2)(y_1^3 - y_2) + x_1x_2y_1y_2$

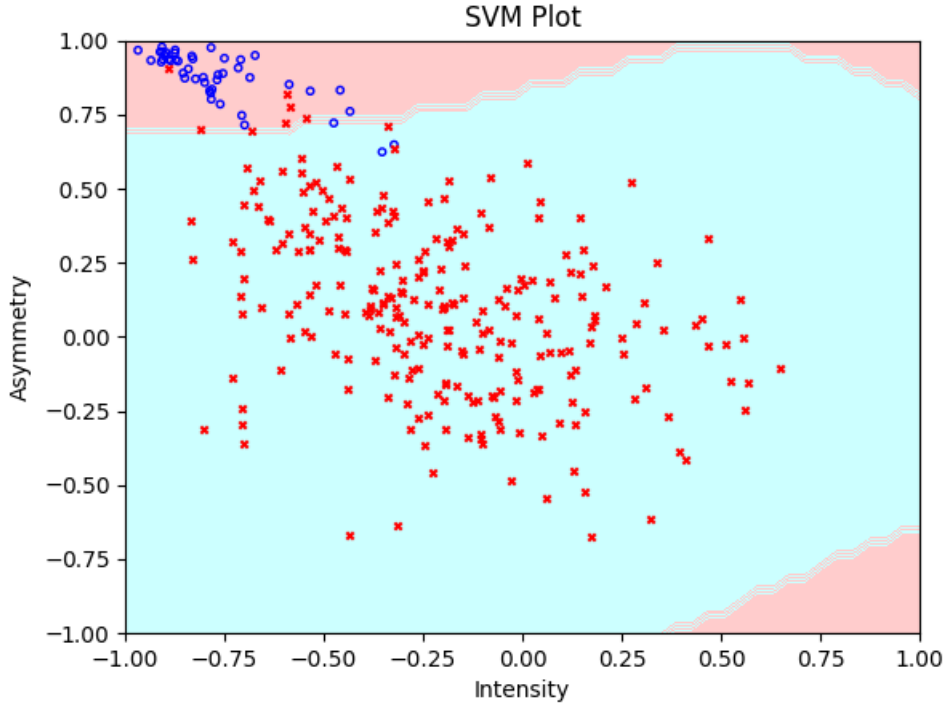
(e) Explicit Form
$g(x) = \text{sign}(x_1^3 - y)$

4. (600) SVM with digits data

(a)
Small C = 10



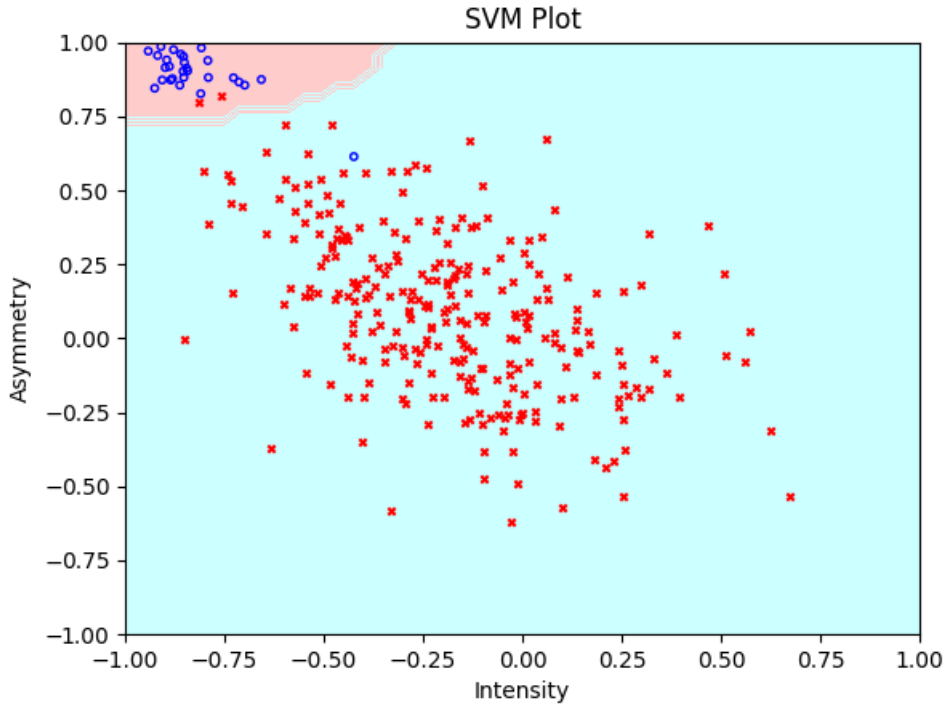
Large C = 100



(b)
As we can see, the larger the C value, the more complex the decision boundary is.

The boundary is tighter, fitting better to our dataset. This makes sense as there should be a higher penalty for violating margins AKA misclassified samples. This should give us more correctly classified samples, thus improving our accuracy. However, I expect overfitting to occur once C become very large due to the fact that we might end up with a over-complex model.

(c)



5. (200) Compare Methods: Linear, k-NN, RBF-network, Neural Network, SVM

In terms of performance, the Neural Networks had the least percentage of errors compared to the other methods. The biggest drawback however, is that the neural networks took much longer to run compared to these other methods. SVM on the other hand ran exceptionally quick with decent/average performance. Also, using an SVM would be better than a neural network if our data was always changing. In essence, it comes down once again to our use case. Some problems are better suited for the SVM and others a neural network. For example, previously we worked on classifying digits and k-NN worked the best for that. There is also the time vs space tradeoff.