LFD Problem Set 12

John Cohen

December 9, 2024

Neural Networks and Backpropagation

(a) Tanh Transformation: $G^{(1)} = \begin{bmatrix} 0.06314459 & 0.06314459 \\ 0.12628918 & 0.12628918 \\ 0.06314459 & 0.06314459 \end{bmatrix}$ $G^{(2)} = \begin{bmatrix} 0.59159231 \\ 0.3177144 \\ 0.3177144 \end{bmatrix}$

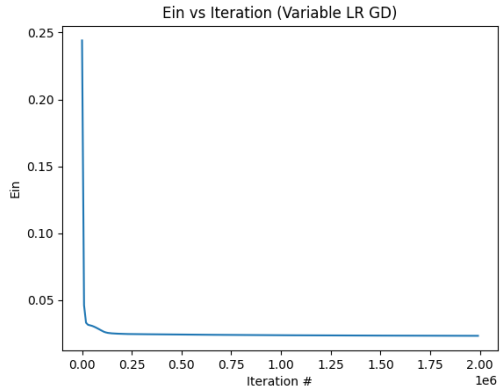
Identity Transformation: $G^{(1)} = \begin{bmatrix} 0.06997201 & 0.06997201 \\ 0.13994403 & 0.13994403 \\ 0.06997201 & 0.06997201 \end{bmatrix}$ $G^{(2)} = \begin{bmatrix} 0.65555744 \\ 0.35206684 \\ 0.35206684 \end{bmatrix}$

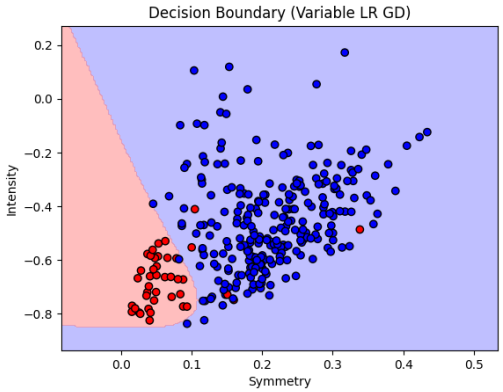
(b) Tanh Transformation: $G^{(1)}_{\text{num}} = \begin{bmatrix} 0.06314459 & 0.06314459 \\ 0.12628918 & 0.12628918 \\ 0.06314459 & 0.06314459 \end{bmatrix}$ $G^{(2)}_{\text{num}} = \begin{bmatrix} 0.59159231 \\ 0.3177144 \\ 0.3177144 \end{bmatrix}$

Identity Transformation: $G^{(1)}_{\text{num}} = \begin{bmatrix} 0.06997201 & 0.06997201 \\ 0.13994403 & 0.13994403 \\ 0.06997201 & 0.06997201 \end{bmatrix}$ $G^{(2)}_{\text{num}} = \begin{bmatrix} 0.65555744 \\ 0.35206684 \\ 0.35206684 \end{bmatrix}$

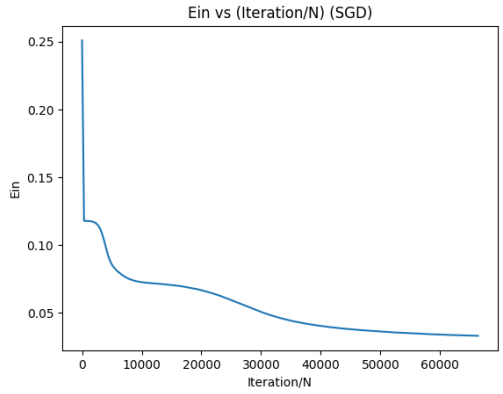
Neural Network for Digits

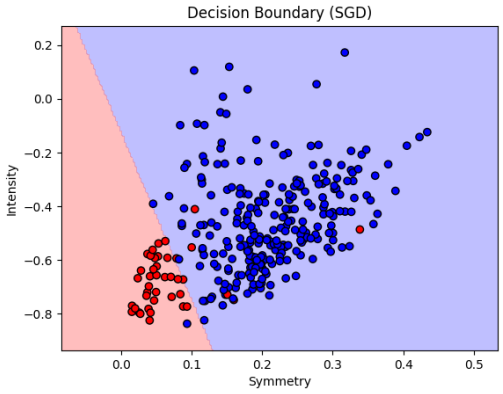
(a) Plot $E_{in}(w)$ versus iteration for the variable learning rate gradient descent heuristic and 2×10^6 iterations. Show the decision boundary for the resulting classifier.





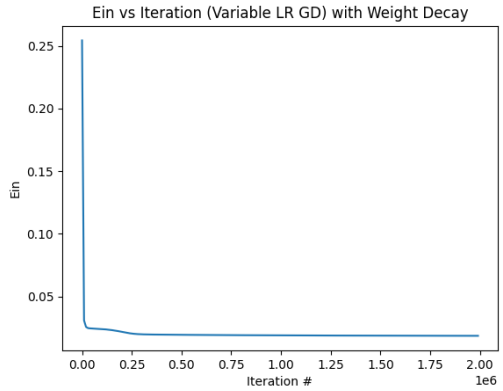
(b) Plot $E_{in}(w)$ versus iteration /N for stochastic gradient descent and 2×10^7 iterations. Show the decision boundary for the resulting classifier. Why divide iteration by N?

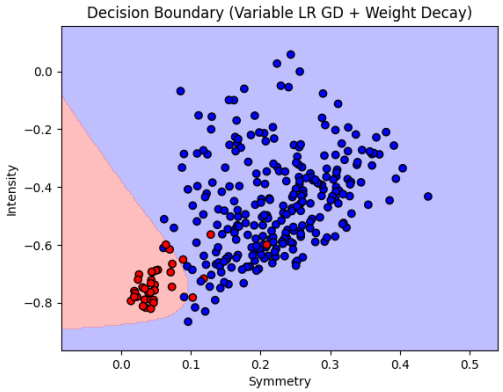


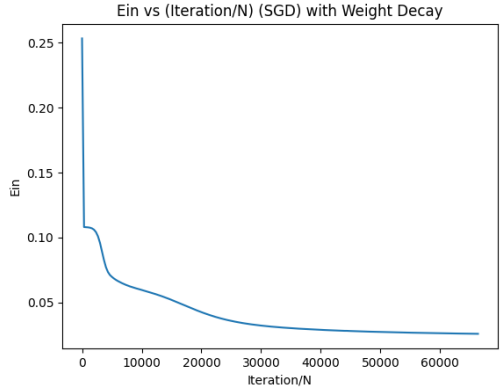


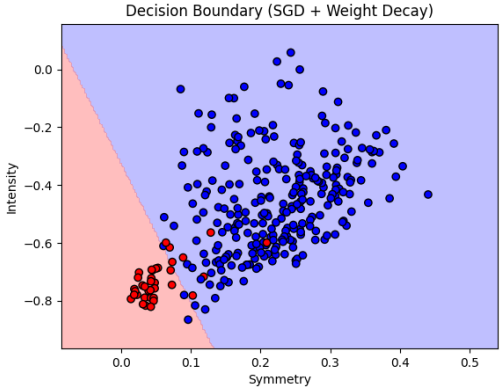
You divide because we update the weights every propagation for each point. Therefore we make N updates after running on N points.

(c) Repeat (a)–(b) to minimize the augmented error with weight decay and $\lambda = 0.01/N$.

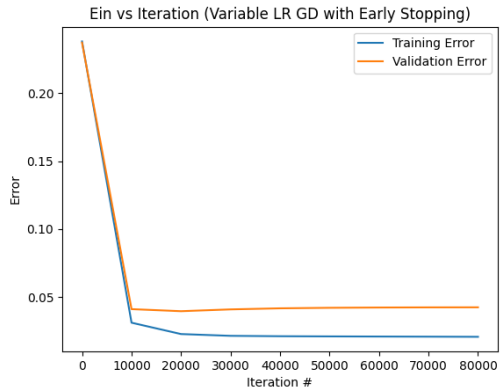


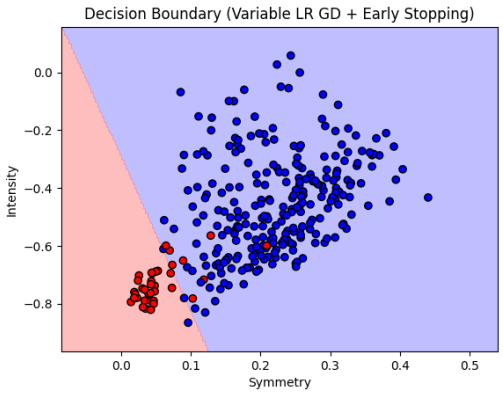


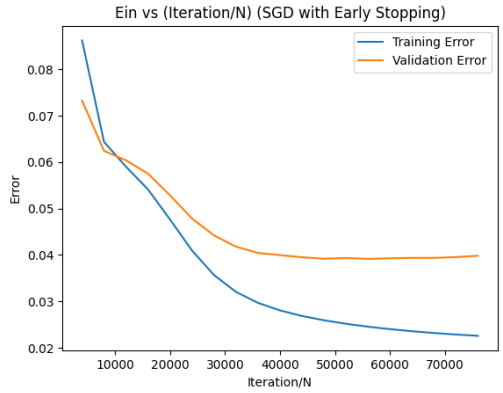


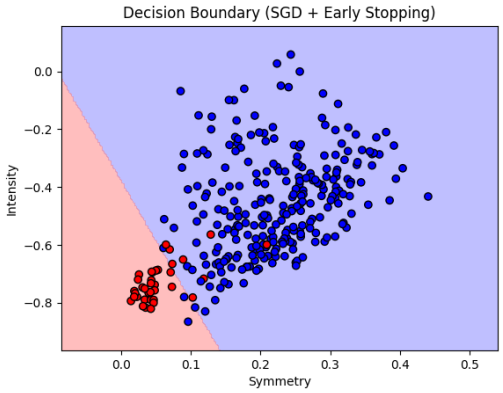


(d) Repeat (a)–(b) using early stopping with a validation set of size 50 and training set of size 250. Show the classifier with minimum validation error.









Support Vector Machines

(a) For the points $x_1 = (1, 0), y_1 = +1, \text{and} x_2 = (-1, 0), y_2 = -1$, the optimal separating hyperplane is the one that maximizes the margin while correctly classifying the points. Geometrically, the optimal hyperplane is the perpendicular bisector of the line segment connecting the two points. The midpoint of the line segment is $(0, 0)$, and since the line segment lies along the x-axis, the perpendicular bisector is vertical, passing through the midpoint. The equation of this hyperplane is $x_1 = 0$, which maximally separates the two points and aligns with the definition of the optimal separating hyperplane.

(b.i)

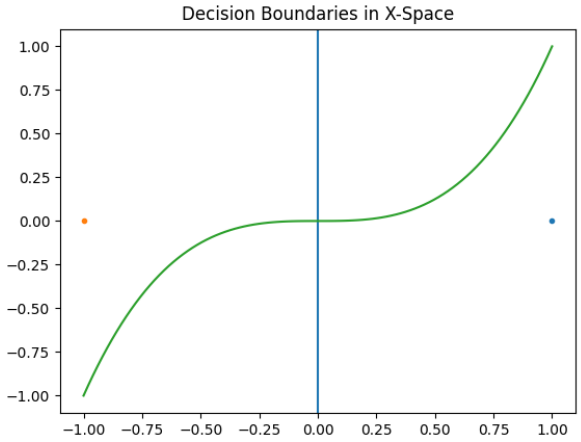
$$z(x_1) = \begin{bmatrix} 1^3 - 0 \\ 1 * 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$z(x_2) = \begin{bmatrix} (-1)^3 - 0 \\ -1 * 0 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$$

(b.ii)

$$w_z = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad b_z = 0$$

(c) See below



(d)

$$K(x,y) = z(x) * z(y) = \begin{bmatrix} x_1^3 - x_2 \\ x_1 x_2 \end{bmatrix} * \begin{bmatrix} y_1^3 - y_2 \\ y_1 y_2 \end{bmatrix}$$

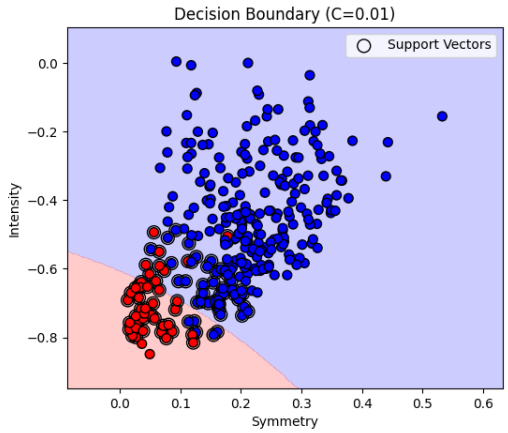
$$K(x,y) = (x_1^3 - x_2)(y_1^3 - y_2) + (x_1 x_2)(y_1 y_2)$$

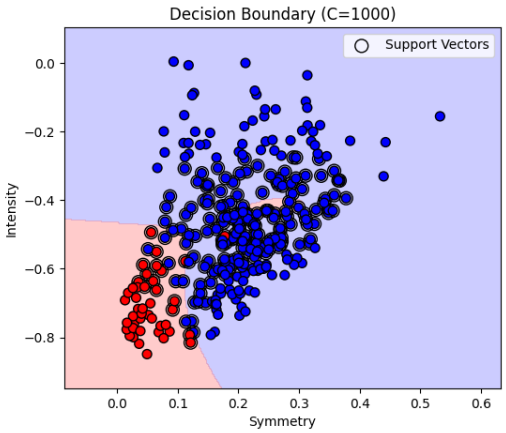
(e)

$$h(x) = \text{sign}(K(w,x) + b) = \text{sign}((w_1^3 - w_2) * (x_1^3 - x_2) + (w_1 w_2) * (x_1 x_2) + b)$$

SVM with digits data

(a) see below

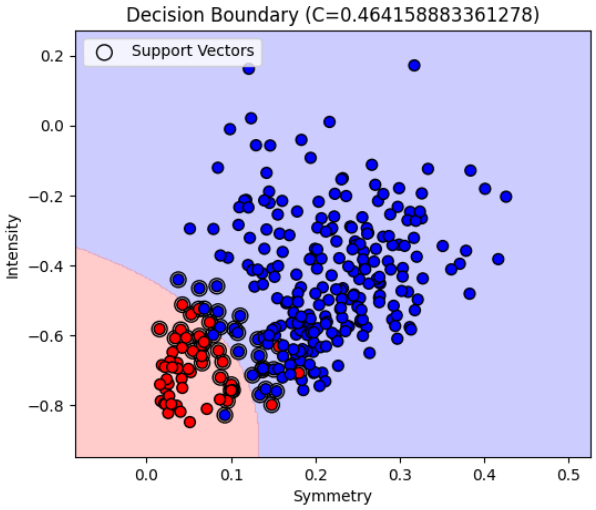




(b) The large C value boundary was much more complex than for the small C value. Large C values are less regularized and such the smaller values equate to more regularization and thus less complex decision boundaries. That is why we see more of a straight line for C = 0.01 and more wiggles for C = 1000.

(c)

$$E_{cv} = 0.26 \quad E_{\text{test}} = 0.03389642142698377$$



Compare Methods: Linear, k-NN, RBF-network, Neural Network, SVM

The results demonstrate that the choice of features has a significantly greater impact on test error than the choice of model. Comparing test errors among classmates using the same models but different features revealed stark differences, underscoring the critical role of feature engineering. Across the models, such as the regularized linear model, k-NN, RBF-network, neural network, and SVM, test errors were relatively consistent when similar features were used. However, SVMs showed a clear strength in handling separable data with well-defined boundaries. Reflecting on this experience, I would prioritize spending more time on feature selection and engineering in future assignments, as this appears to be the most influential factor in achieving better performance, regardless of the model.