6.036: Machine Learning

Ryan Lacey <rlacey@mit.edu>

February 11, 2014

6. (c) The range of possible separators is much smaller in part (b) than in part (a), which increases the likelihood of the perceptron algorithm requiring a high number of iterations. Starting with $x^{(2)}$ we do not see this issue because the resulting decision boundary immediately and perfectly separates the data points. Starting with $x^{(1)}$, on the other hand, we see how the smaller changes in the slope of the decision boundary with $x^{(3)}$ at [10, 1] has a large impact on the performance of the algorithm, taking three times as many iterations as with $x^{(3)}$ at [1.5, 1].

   (d) An adversary would have to goals to maximize the amount of mistakes made by the perceptron algorithm. One is minimize the margin between differently labeled points. Since the coordinates of all the points are known, the labels can be assigned in such a way to cluster together different labels while maintaining a system that is linearly separable. The other goal of the adversary is to try to prevent the algorithm from making small incremental adjustments to the boundary and instead try to make the boundary jump. To achieve this we order the traversal of points in increasing distance from the origin. This causes the later points to still potentially have significant sway on θ with the hope being that the jumps will cause misclassification of points that were previously fixed.

7. (a) No, the training procedures do not necessarily converge to the same θ and θ_0. If training points are linearly separable, there are many choices for θ and θ_0.

   As an example take the training points $x^{(1)} = [2, 1]$, $x^{(2)} = [-2, -1]$, and $x^{(3)} = [-1, 5]$ with corresponding labels $y^{(1)} = (+1)$, $y^{(2)} = (-1)$, and $y^{(3)} = (+1)$.

   We define two different decisions boundaries with the following θ and θ_0:

   $\theta^{(1)} = \left[\begin{smallmatrix}0\\1\end{smallmatrix}\right]$ and $\theta_0^{(1)} = 0$

   $\theta^{(2)} = \left[\begin{smallmatrix}1\\1\end{smallmatrix}\right]$ and $\theta_0^{(2)} = -1$

   These decision boundaries linearly separate the data so when the perceptron algorithm is applied to the training data with these parameters the boundaries incur no updates. Thus we have differing results from the perceptron algorithm on the same data.

   (b) The decision boundaries will not necessarily have the same performance on the test data. Let us assume we have some test data that is linearly separable along the x_1 axis. In this case the first decision boundary from part (a) would perfectly classify the data, while the second boundary is apt to incorrectly predict several points.

   (c) $\theta = \left[\begin{smallmatrix}-4\\2\end{smallmatrix}\right]$ and $\theta_0 = -1$

8. (a) i. Such a θ would occur when the dot product is zero.

   ii. PLACEHOLDER

   (b) i. No such family member exists because three of the points are equidistant from the origin and only two are of the same label. Thus the classifier would have to predict these points to all be of the same value, which is incorrect.

ii. The points can be correctly classified with a circle of radius 2 centered at [-1, -1].

iii. Looking at only the negatively labeled points we can conclude that no such family member exists. The points would either lie directly on the decision boundary (thus both being misclassified) or one would lie above and the other below the boundary (thus misclassifying one of the two points).

iv. A line with normal [1,1] and offset of -0.5 would satisfy.

(c) Families (iii) and (iv) are linear classifiers.

9. (a) $A = \begin{bmatrix} \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ 1 & 1 & 1 & -1 & -1 & -1 \end{bmatrix}$

(b) pl

10. (a) The training set might not be representative of the general population of data. If so, then the classifier would be a poor predictor of the correct labels that should be applied to the population data.

Only measuring performance by the training set also leads to a tendency of overfitting to the training data. Because of this, even with training data that is fairly representative of population data, the predictions may not generalize well.

(b) The perceptron algorithm would be ran on $n-1$ points n number of times, each time leaving out one of the training points to use as the test. The goal of cross validation is to test how well the classifier generalizes and about how often it is expected to misclassify data from the test set. If the training data is drawn from and is representative of the population being studied, then cross validation should show errors about as often as test set validation. The advantage, however, is that cross validation requires less data to be collected since data points are reused from training for testing.