6.036: Machine Learning

Ryan Lacey <rlacey@mit.edu>

February 13, 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 maximizes potential errors with the following strategy (maximizing $\frac{R^2}{\gamma^2}$):
   - Spread the points out over a large area, thereby increasing R.
   - Assign labels to points in a manner that minimizes the margin between differently labeled points, thereby reducing γ.
   - Traverse the points in order from further from the origin to nearest. This will cause the large magnitude vectors to have a strong influence at the beginning of the algorithm making the boundary line jump. Then the near vectors, which have much less of an effect on θ due to their comparably small magnitude, will have difficultly in nudging the decision boundary to an acceptable position, ideally causing further iterations of the algorithm.

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) $\theta = \left[\begin{smallmatrix} -3 \\ 2 \end{smallmatrix}\right]$ and $\theta_0 = -2$

8. (a) i. No such θ could exist because the dot product with x would always be zero, thus making the second constraint unsatisfiable.

   ii. Yes, the addition of the offset parameter would allow us to satisfy both constraints.

(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) $A = \begin{bmatrix} \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & -\frac{1}{3} & -\frac{1}{3} & -\frac{1}{3} \end{bmatrix}$

(b) placeholderB

(c) placeholderC

(d) placeholderD

(e) placeholderE

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.

(c) A large number of cross validation errors with a low number of training set errors suggests that the boundary decision resulting from the learning algorithm overfit to the training set. One possible solution would be to take the best decision boundary that resulted from the data points subsets of the cross validation runs.