Notebook

November 7, 2019

Question 2a Recall the optimal value of β should minimize our loss function. One way we've approached solving for β is by taking the derivative of our loss function with respect to β, like we did in HW6.

In the space below, use LaTeX to write/compute the following values: * $R(\mathbf{x}, \mathbf{y}, \beta_1, \beta_2)$: our loss function, the empirical risk/mean squared error * $\frac{\partial R}{\partial \beta_1}$: the partial derivative of R with respect to θ_1 * $\frac{\partial R}{\partial \beta_2}$: the partial derivative of R with respect to θ_2

$R(x, y, \beta_1, \beta_2) = \frac{1}{n} \sum_{i=0}^{n} (y_i - (\beta_1 x_i + sin(\beta_2 x_i)))^2$ | $\frac{\partial R}{\partial \beta_1} = -2\frac{1}{n} \sum_{i=0}^{n} x(y_i - (\beta_1 x_i + sin(\beta_2 x_i)))$ | $\frac{\partial R}{\partial \beta_2} = -2\frac{1}{n} \sum_{i=0}^{n} xcos(\beta_2 x)(y_i - (\beta_1 x_i + sin(\beta_2 x_i)))$

In 1-2 sentences, describe what you notice about the path that beta takes with a static learning rate vs. a decaying learning rate. In your answer, refer to either pair of plots above (the 3d plot or the contour plot).

Looking at both plots, I notice that beta takes a shorter path for the decaying learning rate. This is evident in both the contour plot and 3d plot. The static learning rate took much more steps going back and forth, whereas the decaying learning rate took much less steps.

In the cell below, explain why we got the error above when trying to calculate the analytical solution for our one-hot encoded `tips` dataset.

We are getting an error when trying to calculate the analytical solution because the shapes of our tips and one_hot_X are vastly different. The shape of tips is (244,) and the shape of one_hot_X is (244,12), and because of this, we are not able to do the dot product of one_hot_X transpose and tips becuase they are not conducive.

Why did removing redundancies in our one-hot encoding fix the problem we had in 6a?

After removing the redundancies, the shape of our one_hot_X_revised is now (244,8). From removing the redundancies, we are now able to get the inverse of the transpose from the dot product of X. Removing the redundancies also gets rid of the overlaps in one_hot_X, making it possible to find these dot products.

Our precision is fairly high, while our recall is a bit lower. Why may we observe these results?

Hint: Consider the distribution of true/false values in our data.

If our precision is higher than our recall, this means that we have more false negatives compared to false positives. This means that we have more instances where we say a tumor is malignant when it is actually benign. And we have less instances where we say a tumor is benign when it is actually malignant. This means that our model is more inclined to report true for a tumor being malignant.