Logistic Regression Tutorial
COMS3007

Benjamin Rosman

1. For this week's lab we will begin by generating data again. In this case we will be generating data for a classification problem.

   An easy way to generate classification data is to define and sample from a distribution for each class. We commonly use Gaussian distributions. In your favourite language, perform the following tasks:

   (a) Define two 2D Gaussian distributions: one with centre at $(1, -1)$ and the other at $(-1, 1)$. Let them both have variances of 1 in each dimension. We'll treat these as the generators of data from classes 0 and 1 respectively.

   (b) Draw 20 data points from each distribution (use the `randn` function in Matlab or Python). Plot the points from each class in a different colour. How easily separable are these classes?

   (c) Repeat the process above with variances of 3. How separable is this data?

   (d) Repeat the process, with different means and variances. Which datasets do you expect to be easier to classify?

2. For this question, use the dataset you generated for question 1(b) above. We are now going to train a logistic regression model to classify this data.

   (a) This data is 2D, in x_1 and x_2. We therefore need to learn three parameters for the model, as we augment them with a bias parameter θ_0, so we need to learn $\theta = (\theta_0, \theta_1, \theta_2)$. Choose random initial values for these parameters, in $(-0.5, 0.5)$, and plot this line. This line is the dividing line between the two classes called the decision boundary, and corresponds to the line where the value of the logistic function $= 0.5$, i.e. the area where the predictor is completely uncertain.

   (b) What is the error of this random classifier on your data? Describe the error both in terms of the log-likelihood error function, and a confusion matrix.

   (c) Write out the weight update equations for each parameter. Choose a learning rate $\alpha = 0.01$.

   (d) Perform a single weight update by hand for two of your datapoints (choose one from each class). Make sure you understand how the update works. Repeat this in code to check you have the right answer.

   (e) In code, cycle through all training datapoints, and use the three weight update equations to adjust the parameters. Draw the decision boundary. What is the error now?

   (f) Repeat the update in a loop. We usually use two terminating conditions: the first is to look at the normed difference between the parameter vector between two successive iterations and we terminate when this is small, i.e. $||\theta_{new} - \theta_{old}|| < \epsilon$, with $\epsilon = 0.05$. We also usually set a maximum number of iterations, e.g. 1000, just in case. Run the learning until convergence and plot the decision boundary. What do you notice? What is the error on the training data?

(g) Generate 20 more datapoints from each of your two Gaussians. This will be our validation data. Classify them using your trained model and tabulate the results in a confusion matrix. Compare the error on the training data to the error on the validation data. What do you notice?

(h) Change the hyperparameters: the learning rate α and termination threshold ϵ. What effect do these have on learning? For each different setting of the hyperparameters, retrain your model on the *training data* and evaluate it on the *validation data*.

(i) Keep the best values of your hyperparameters. Now generate 20 more datapoints from each of your two Gaussians. This will be our testing data. Classify them using your trained model and tabulate the results in a confusion matrix. This is the final performance of the classifier with optimised hyperparameters!

(j) Why is it important to have the three data sets: training, validation, and testing?