EE 510: Machine Learning Theory & Algorithms **Spring 2020**

Homework 2
Due: April 17, 2020, 11:59PM PT

Student Name: *Instructor Name: John Lipor*

Problem 1 Multiclass Logistic Regression Classifier (5 pts, 5 pts, 5 pts, 2 pts)

In this problem, you will derive and implement a multiclass linear classifier based on *logistic regression* (LR, see Section 9.3 of UML). For binary classification, LR has two interpretations. The first, which is given in UML, is that it performs ERM with the logistic loss, i.e., using

$$\ell(h_w, (x, y)) = \log\left(1 + e^{-y(w^T x)}\right),$$

where $x, w \in \mathbb{R}^d$, $y \in \{-1, 1\}$, and

$$h_w(x) = \text{sign}\left(w^T x\right).$$

This results in the ERM objective function

$$J_1(w) = \frac{1}{m}\sum_{i=1}^{m} \log\left(1 + e^{-y_i(w^T x_i)}\right). \tag{1}$$

The second interpretation of (binary) LR is that it models/learns the conditional distribution $\mathbb{P}[Y = y \mid x]$ and then uses this model to estimate the Bayes classifier. In particular, LR assumes that

$$\mathbb{P}[Y = y \mid x, w] \approx \sigma(w^T x)$$

where

$$\sigma(t) = \frac{1}{1 + e^{-t}}$$

is called the *sigmoid* or *logistic* function. You can check that $\sigma(w^T x) \in [0, 1]$ and goes to 1 when $w^T x$ is large and 0 when $w^T x$ is small. The vector w can be learned using maximum likelihood estimation, which results in minimizing the negative log-likelihood

$$J_2(w) = -\frac{1}{m}\sum_{i=1}^{m} y_i \log\left(\sigma(w^T x_i)\right) + (1 - y_i)\log\left(1 - \sigma(w^T x_i)\right), \tag{2}$$

where now we set $y_i \in \{0, 1\}$. The learned w is then used in $\sigma(t)$ above to estimate the Bayes classifier.

The other benefit of the probabilistic interpretation of LR is that it easily extends to the multiclass case. In this case, the conditional probability is modeled as

$$\mathbb{P}[Y = k \mid x, w_1, \ldots, w_K] \approx \text{softmax}(k, w_1^T x, \ldots, w_K^T x),$$

where

$$\text{softmax}(k, t_1, \ldots, t_K) = \frac{e^{t_k}}{\sum_{j=1}^{K} e^{t_j}}$$

is the *softmax* function, which is the generalization of the sigmoid to multiple classes. We learn K weight vectors (one for each class) by minimizing the cost function

$$J(w_1, \ldots, w_K) = -\sum_{i=1}^{m}\sum_{k=1}^{K} \mathbb{1}\{y_i = k\} \log\left(\text{softmax}(k, w_1^T x_i, \ldots, w_K^T x_i)\right). \tag{3}$$

The above is minimized using (stochastic) gradient descent and is known as multiclass LR/multinomial LR/softmax regression.

(a) Verify that (1) and (2) are equivalent cost functions. *Hint:* Start with (2) and use algebraic manipulations to show that it is equivalent to (1).

(b) Verify that the gradient of (3) with respect to one weight vector is

$$\nabla_{w_k} J(w_1, \ldots, w_K) = \sum_{i=1}^{m} x_i \left(\text{softmax}(k, w_1^T x, \ldots, w_K^T x) - \mathbb{1}\left\{ y_i = k \right\} \right).$$

(c) Complete the script `prob1.py` by minimizing (3) using stochastic gradient descent and then using the learned weight vectors to predict on the MNIST dataset (use the data from Homework 1). Use a learning rate of $\mu = 10^{-2}$ and run ten full passes through the data (you can play with these if you want). **Turn in** your code, as well as the classification error on the training and test sets. *Hint:* I've given you code to create a smaller dataset for debugging that selects only three digits. Your final training and test errors should both be below 10%.

(d) What do you notice about the differences between LR and the multiclass ridge regression classifier from Homework 1?

Problem 2 DSS: Visualizing Data (3 pts, 5 pts, 5 pts)

(*DSS rules apply.*) Another important tool for any data scientist is knowing how to visualize results. Two popular approaches to visualizing high-dimensional datasets are *t-distributed stochastic neighbor embedding* (t-SNE) and *Uniform Manifold Approximation and Projection for Dimension Reduction* (UMAP). In this problem, you will use UMAP to embed the MNIST **test** dataset and create an interactive plot showing which digits were misclassified. To complete this problem, you will hack the tutorial here, which requires the following packages: `pandas, seaborn, bokeh, umap`.
Hint: Start by grabbing a subset of the digits to reduce computation time while debugging.

(a) Use UMAP to create a scatter plot of the MNIST test dataset embedded into two dimensions, with the color of each point in your plot corresponding to the true class of the image. Note that the 'Digits' dataset is not the same as the MNIST dataset. **Turn in** your plot.

(b) Hack the code in the linked tutorial to create an interactive plot that allows you to view the image and true class of each point in the MNIST test dataset when you hover over it. **Turn in** a screenshot of your plot with your mouse hovering over a point to show the tooltip.

(c) An important method for troubleshooting and displaying results is understanding what types of examples are misclassified by your predictor. First, classify the MNIST test dataset using `LogisticRegression` from `sklearn`. Next, hack the code from part (b) so that the points are colored based on whether they are classified correctly. **Turn in** your code, a screenshot of your plot with your mouse hovering over a point to show the tooltip, and some observations of what you learned about which types of points are incorrectly classified.

Problem 3 SLT (10 pts)

(*SLT rules apply.*) UML, Ch. 3, Exercise 7. State how long you worked on the problem before looking at the solution.

Problem 4 SLT (10 pts)

(*SLT rules apply.*) UML, Ch. 4, Exercise 2. State how long you worked on the problem before looking at the solution.