CSCE 489: Machine Learning (Spring 2019)

Homework #2
Due 2/18/2019

---

1. You need to submit a report in hard-copy before lecture and your code to eCampus. Your hard-copy report should include (1) answers to the non-programming part, and (2) results of the programming part. Your submission to eCampus should be your code files ONLY. Please put all your code files into a compressed file named "HW#_FirstName_LastName.zip"
2. Hard-copy is due in class before lecture, and code files are due 9:10AM to eCampus on the due date.
3. Unlimited number of submissions are allowed on eCampus and the latest one will be graded. If you make a resubmission after the due date, it will be considered late.
4. LFD refers to the textbook "Learning from Data".
5. Please read and follow submission instructions. No exception will be made to accommodate incorrectly submitted files/reports.
6. All students are highly encouraged to typeset their reports using Word or LaTeX. In case you decide to hand-write, please make sure your answers are clearly readable.

---

1. (10 points) Exercise 3.6 (page 92) in LFD.
2. (20 points) Recall the objective function for linear regression can be expressed as

$$E(w) = \frac{1}{N}\|Xw - y\|^2,$$

as in Equation (3.3) of LFD. Minimizing this function with respect to w leads to the optimal w as $(X^TX)^{-1}X^Ty$. This solution holds only when X^TX is nonsingular. To overcome this problem, the following objective function is commonly minimized instead:

$$E_2(w) = \|Xw - y\|^2 + \lambda\|w\|^2,$$

where $\lambda > 0$ is a user-specified parameter. Please do the following:

   (a) (10 points) Derive the optimal w that minimize $E_2(w)$.
   (b) (10 points) Explain how this new objective function can overcome the singularity problem of X^TX.

3. (35 points) In logistic regression, the objective function can be written as

$$E(w) = \frac{1}{N}\sum_{n=1}^{N} \ln\left(1 + e^{-y_n w^T x_n}\right).$$

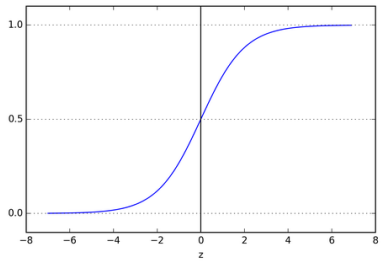
Please

(a) (10 points) Compute the first-order derivative $\nabla E(w)$. You will need to provide the intermediate steps of derivation.

(b) (10 points) Once the optimal w is obtain, it will be used to make predictions as follows:

$$\text{Predicted class of } x = \begin{cases} 1 & \text{if } \theta(w^T x) \geq 0.5 \\ -1 & \text{if } \theta(w^T x) < 0.5 \end{cases}$$

where the function $\theta(z) = \frac{1}{1+e^{-z}}$ looks like



Explain why the decision boundary of logistic regression is still linear, though the linear signal $w^T x$ is passed through a nonlinear function θ to compute the outcome of prediction.

(c) (5 points) Is the decision boundary still linear if the prediction rule is changed to the following? Justify briefly.

$$\text{Predicted class of } x = \begin{cases} 1 & \text{if } \theta(w^T x) \geq 0.9 \\ -1 & \text{if } \theta(w^T x) < 0.9 \end{cases}$$

(d) (10 points) In light of your answers to the above two questions, what is the essential property of logistic regression that results in the linear decision boundary?

4. (35 points) **Logistic Regression for Handwritten Digits Recognition**: Implement logistic regression for classification using gradient descent to find the best separator. The handwritten digits files are in the "data" folder: train.txt and test.txt. The starting code is in the "code" folder. In the data file, each row is a data example. The first entry is the digit label ("1" or "5"), and the next 256 are grayscale values between -1 and 1. The 256 pixels correspond to a 16×16 image. You are expected to implement your solution based on the given codes. The only file you need to modify is the "solution.py" file. You can test your solution by running "main.py" file. Note that code is provided to compute a two-dimensional feature (symmetry and average intensity) from each digit image; that is, each digit image is represented by a two-dimensional vector before being augmented with a "1" to form a three-dimensional vector as discussed in class. These features along with the corresponding labels should serve as inputs to your logistic regresion algorithm.

Note: Please write all your code between comments "### YOUR CODE HERE" and "### END YOUR CODE".

(a) (5 points) Complete the ***_third_order()*** function to transform the features into 3rd order polynomial Z-space.

(b) (15 points) Complete the ***_gradient()*** and the ***fit()*** methods in the ***LogisticRegression()*** class for training the logistic regression model.

(c) (5 points) Complete the ***predict()*** method in the ***LogisticRegression()*** class.

(d) (10 points) Run "main.py" to see the classification results. As your final deliverable to a customer, would you use the linear model with or without the 3rd order polynomial transform? Briefly explain your reason.