CS436/536: Introduction to Machine Learning
Homework 4
(Due 3/31 **before 11.59pm**)

Instructions: To solve these problems, you are allowed to consult your classmates, as well as the class textbook (*Learning from Data* by Abu-Mostafa, Magdon-Ismail, and Lin, which we will call LFD) and the slides posted on Brightspace, but no other sources. You are encouraged to collaborate with other students, while respecting the collaboration policy (please see the module on Academic Honesty on Brightspace). Please write the names of all the other students you collaborated with on the homework. Everyone must write up their assignments separately.

Please write clearly and concisely, and use rigorous, formal arguments. Homework is due at the beginning of lecture, and homework turned in later will be considered late and will use up one of your late days. You must use Brightspace to submit the homework as a single neatly typed pdf file. Hand-drawn formulas or figures are okay and may be included as images within the pdf. If a programming assignment calls for plotting the results, axes must be clearly labeled, and its meaning must be obvious to anyone with only a rudimentary knowledge of machine learning and computer science. Emailed copies will not be accepted.

(1) [100 points] LFD Exercise 4.3.

(2) [100 points] LFD Exercise 4.6.

(3) [100 points] LFD Exercise 4.8.

(4) [100 points] LFD Exercise 4.11.

(5) [1000 points] An End-to-End Learning System with Regularization and Validation: Predicting 1s vs. Not 1s.

We revisit the MNIST Handwritten Digits Dataset we worked with in the last homework to solve the problem of predicting whether a given image of a handwritten digit represents either the digit 1 or not the digit 1, i.e., if the n-th example is labeled as being the digit 1, then $y_n = +1$, and otherwise, $y_n = -1$.

First, you must perform the following steps to prepare to solve the problem:

(1) **[Combine Data]** Combine the training and test sets (in `ZipDigits.train` and `ZipDigits.test` respectively) into a single dataset.

(2) **[Compute Features]** Use the algorithms you developed in the previous homework to compute two features for each example in the dataset.

(3) **[Normalize Features]** For each data point, *shift* and then *rescale* the values of each feature across the entire dataset so that for each feature, so that the values of every feature are in the range $[-1, 1]$.

(4) **[Create Input and Test Datasets]** Select 300 data points from the dataset *uniformly at random* (and without replacement) to for the dataset $\mathcal{D}$. Use the remaining data points to form the test dataset $\mathcal{D}_{\text{test}}$. You must leave aside $\mathcal{D}_{\text{test}}$ and not touch it till the end of this exercise when we will compute the final output g of our learning system. We will use $\mathcal{D}_{\text{test}}$ to estimate $E_{\text{out}}(g)$.

For convenience, we will treat this as a regression problem with real valued targets ± 1, until we output our final hypothesis g for classifying handwritten images as either the digit 1 or not the digit 1, at which point we will use $\text{sign}(g(x))$ to predict the class of a test data point x.

The standard polynomial feature transform generates features which are not 'orthogonal', making the columns in the data matrix dependent. This can be problematic for the one-step linear regression algorithm as it requires computing the (pseudo-)inverse of a matrix. An 'orthogonal' polynomial transform is

$$(x_1, x_2) \to (1, L_1(x_1), L_1(x_2), L_2(x_1), L_1(x_1)L_1(x_2), L_2(x_2), L_3(x_1), L_2(x_1)L_1(x_2), \dots),$$

where $L_k(x_i)$ is the k-th order polynomial transform applied to the i-th feature of the input data point x. See LFD Problem 4.3 for a recursive expression that defines the Legendre polynomials which can be implemented as an efficient iterative algorithm.

We will use the one-step (pseudo-inverse) algorithm for linear regression algorithm with weight decay regularization for learning. This corresponds to minimizing the augmented error $E_{\text{aug}}(w) = E_{\text{in}}(w) + \lambda w^T w$, where $E_{\text{in}}(w)$ is the sum of squared errors. The weights that minimize $E_{\text{aug}}(w)$ are $w_{\text{lin}}(\lambda) = (Z^T Z + \lambda I)^{-1} Z^T y$, where Z is the $N \times \tilde{d}$ matrix generated from the polynomial transform of the data points X in the data set $\mathcal{D} = (X, y)$ and w_{lin} is the $\tilde{d} \times 1$ regularized weight vector.

Now, complete the following tasks to arrive at the final hypothesis g.

(Task 1) **[100 points] 10-th order Polynomial Transform.** Use the 10-th order Legendre polynomial feature transform to compute Z. Report the dimensions of Z.

(Task 2) **[100 points] Overfitting.** Plot the decision boundary of the output of the regularized linear regression algorithm without any regularization ($\lambda = 0$). What do you observe, overfitting or underfitting?

(Task 3) **[100 points] Regularization.** Plot the decision boundary of the output of the regularized linear regression algorithm with $\lambda = 3$. Do you observe overfitting or underfitting?

(Task 4) **[200 points] Cross Validation.** Use leave-one-out cross validation to estimate $E_{\text{CV}}(\lambda)$ for $\lambda \in \{0, 0.01, 0.1, 1, 5, 10, 25, 50, 75, 100\}$. Plot E_{CV} versus λ and $E_{\text{test}}(w_{\text{lin}}(\lambda))$ versus λ on the same plot. Comment on the behavior of E_{CV} and E_{test} versus λ. Here, E_{CV} and E_{test} are the regression, sum of squared errors.

(Task 5) **[100 points] Pick λ.** Use the cross validation errors from the previous step to pick the best value of λ, and call it λ^*. Plot the decision boundary corresponding to the weights $w_{\text{lin}(\lambda^*)}$.

(Task 6) **[100 points] Estimate Classification Error.** Use $w_{\text{lin}(\lambda^*)}$ for classification and estimate the *classification* out-of-sample error $E_{\text{out}}(w_{\text{lin}(\lambda^*)})$ for your final hypothesis g. Estimate $E_{\text{out}}(g)$ to distinguishing between digits that are 1s and not 1s (give the 99% error bar).

(Task 7) **[100 points] Is E_{CV} biased?** Comment on whether $E_{\text{CV}}(\lambda^*)$ is an unbiased estimator of $E_{\text{test}}(w_{\text{lin}}(\lambda^*))$(treated as regression error). Why or why not?

(Task 8) **[200 points] Data snooping.** $E_{\text{test}}(w_{\text{lin}}(\lambda^*))$ an unbiased estimator of $E_{\text{out}}(w_{\text{lin}}(\lambda^*))$ (treat them as classification errors)? Why or why not? If not, what could we do differently to fix things, so that it is? Explain.