HW Assignment 2 (Due by 10:30am on Oct 5)

1 Implementation (90 points)

In this exercise, you are asked to implement de gradient descent algorithm and use it to train linear regression models for the Athens houses dataset and an artificial dataset with and without L2 regularization. The input data is available at `http://ace.cs.ohio.edu/~razvan/courses/ml4900/hw02.zip`. Make sure that you organize your code in folders as shown in the table below. Write code only in the Python files indicated in bold.

ml4900/
hw02/
`code/`
univariate.py
multivariate.py
polyfit.py
train_test_line.png
`data/`
`univariate/`
train.txt, test.txt
`multivariate/`
train.txt, test.txt
`polyfit/`
train.txt, test.txt, devel.txt

1. [**Feature Scaling**, 10 points]
   To improve the convergence of gradient descent, it is important that the features are scaled so that they have similar ranges. Implement the standard scaling method using the skeleton code in `scaling.py`.

2. [**Univariate Regression**, 20 points]
   Train a univariate linear regression model to predict house prices as a function of their floor size, by running gradient descent for 200 epochs with a learning rate of 0.1. Use the dataset from the folder `hw02/data/univariate`. Plot $J(\mathbf{w})$ vs. the number of epochs in increments of 10 (i.e. after 0 epochs, 10 epochs, 20 epochs, ...). After training print the parameters and compare with solution from normal equations. Plot the training using the default blue circles and test examples using lime green triangles. On the same graph also plot the linear approximation.

3. [**Multivariate Regression**, 20 points]
   Train a multivariate linear regression model to predict house prices as a function of their floor size, number of bedrooms, and year. Run gradient descent for 500 epochs with a learning rate of 0.1. Use the the dataset from the folder `hw02/data/multivariate`. Plot $J(\mathbf{w})$ vs. the number of epochs in increments of 10 (i.e. after 0 epochs, 10 epochs, 20 epochs, ...). After training print the parameters and compare with solution from normal equations.

4. [**Polynomial Curve Fitting**, 40 points]
   In this exercise, you are asked to use gradient descent to train a linear regression model, with and without regularization, on the artificial dataset from the folder `hw02/data/polyfit`.

   (a) Select 30 values for $x \in [0, 1]$ uniformly spaced, and generate corresponding t values according to $t(x) = sin(2\pi x) + x(x + 1)/4 + \epsilon$, where $\epsilon = N(0, 0.005)$ is a zero mean Gaussian with variance 0.005. Save and plot all the values. Done in `dataset.txt`.

   (b) Split the 30 samples (x_n, t_n) in three sets: 10 samples for training, 10 samples for validation, and 10 samples for testing. Save and plot the 3 datasets separately. Done in `train.txt, test.txt, devel.txt`.

   (c) Consider a linear regression model with polynomial basis functions, trained with the objective shown below:

   $$J(\mathbf{w}) = \frac{1}{2N} \sum_{n=1}^{N} (h(x_n, \mathbf{w}) - t_n)^2 + \frac{\lambda}{2} ||\mathbf{w}||^2$$

   Show the gradient update used for minimizing $J(\mathbf{w})$.

   (d) Train and evaluate the linear regression model in the following scenarios:

      1. Without regularization: Using for M the value that obtained the lowest test RMSE in the first homework, run gradient descent with a learning rate that is tuned on the training data using the following consecutive powers of 10: $\{0.0001, 0.001, 0.01, 0.1, 1, 10\}$. Plot $J(\mathbf{w})$ vs. epochs and select the largest learning rate that leads to a smooth, decreasing behavior of $J(\mathbf{w})$. If convergence appears to be too slow, you may consider consecutive powers of 2 in the same range for the learning rate. Once the learning rate is selected, run gradient descent as long as $J(\mathbf{w})$ decreases by at least 1e-10 after each epoch. Report and compare the RMSE and the trained parameters with the solution from the normal equations for the same degree M.
      2. With regularization: Fixing $M = 9$ and λ to the value tuned in the first homework, repeat the experiments above, this time with regularization. Report and compare the RMSE and the trained parameters with the solution from the normal equations for the same degree M and λ.

5. [**Stochastic Gradient Descent (*)**, 20 points]
   Implement SGD and run it for problems 2, 3, and 4 above, using the same hyperparameters (learning rate, number of epochs, M, and *lambda*). Compare the SGD solution to the batch GD solution. Does SGD need fewer or more epochs to arrive at a similar RMSE as batch GD? If fewer, how many epochs are sufficient?

2 Submission

Turn in a hard copy of your homework report at the beginning of class on the due date. Electronically submit on Blackboard a hw02.zip file that contains the hw02 folder in which your code is in the 3 required files.

On a Linux system, creating the archive can be done using the command:
`> zip -r hw02.zip hw02`.

Please observe the following when handing in homework:

1. Structure, indent, and format your code well.
2. Use adequate comments, both block and in-line to document your code.
3. On the theory assignment, **clear and complete explanations and proofs of your results are as important as getting the right answer**.
4. Make sure your code runs correctly when used in the directory structure shown above.