# CSCE 5063-001: Assignment 1

Due 11:59pm Friday, September 17, 2021

## 1 Linear regression

In this problem, you will implement linear regression to predict the death rate from several features, including annual precipitation, temperature, population, income, pollution, etc. The data for this assignment is given in file `data.txt`, which contains the data description, 17 columns (features) and 60 rows (examples). In the data matrix, columns 2-16 are input features, and column 17 is the output target. Note that Column 1 is the index and should not be used in the regression. You will implement gradient descent for learning the linear regression model.

### 1.1 Feature normalization

By looking at the feature values in the data, you may notice that some features are about 1000 times the other features. When features differ by orders of magnitude, first performing feature normalization can make gradient descent converge much more quickly. There are different ways to do the feature normalization. For simplicity, we use the following method: (1) subtract the minimal value of each feature; (2) divide the feature values by their ranges of values. Equivalently, it is given by the following equation:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Similarly normalize the target $Y$. Note that to make prediction for a new data point, you also need to first normalize the features as what you did previously for the training dataset.

### 1.2 Gradient descent with quadratic regularization

To recap, the loss function for linear regression with quadratic regularization is given by

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left( y^{(i)} - h_\theta(x^{(i)}) \right)^2 + \frac{\lambda}{2m} \sum_{j=1}^{n} \theta_j^2, \tag{1}$$

where $h_\theta(x^{(i)}) = \theta^T x^{(i)}$ is the linear regression model. To minimize this function, we first obtain the gradient with respect to each parameter $\theta_j$ as:

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m} \sum_{i=1}^{m} \left( h_\theta(x^{(i)}) - y^{(i)} \right) x_j^{(i)} + \frac{\lambda}{m} \theta_j.$$

Then, the (batch) gradient descent algorithm is given as:

**Algorithm 1:** Batch Gradient Descent

```
k = 0;
while convergence criteria not reached do
    for j = 1, ..., n do
        Update θ_j ← θ_j − α ∂J(θ)/∂θ_j;
    Update k ← k + 1;
```

where $\alpha$ is the learning rate. The *convergence criteria* for the above algorithm is $\Delta_{\%cost} < \epsilon$, where

$$\Delta_{\%cost} = \frac{|J_{k-1}(\theta) - J_k(\theta)| \times 100}{J_{k-1}(\theta)},$$

where $J_k(\theta)$ is the value of Eq. (1) at $k$th iteration, and $\Delta_{\%cost}$ is computed at the end of each iteration of the while loop. Initialize $\theta = \mathbf{0}$ and compute $J_0(\theta)$ with these values.

**Important:** You must simultaneously update $\theta_j$ for all $j$.

**Task:** Load the dataset in `data.txt`, split it into 80% / 20% training/test (the dataset is already shuffled so you can simply use the first 80% examples for training and the remaining 20% for test.), and learn the linear regression model using the training data. Plot the value of loss function $J_k(\theta)$ vs. the number of iterations ($k$). After the training completes, compute the squared loss (w/o regularization function) on the test data.

## 1.3 Gradient descent with lasso regularization

The loss function for linear regression with lasso regularization is given by

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left( y^{(i)} - h_\theta(x^{(i)}) \right)^2 + \frac{\lambda}{2m} \sum_{j=1}^{n} |\theta_j|. \tag{2}$$

To minimize the loss function, you will need to derive the gradient by yourself. The gradient descent algorithm is the same as the above.

**Hint:** For simplicity, you can consider $\frac{\partial |\theta_j|}{\theta_j} = 1$ when $\theta_j = 0$.

**Task:** Load the dataset in `data.txt`, split it into 80% / 20% training/test, and learn the linear regression model using the training data. Plot the value of loss function $J_k(\theta)$ vs. the number of iterations ($k$). After the training completes, compute the squared loss (w/o regularization function) on the test data.

## 1.4 Comparing quadratic and lasso regularization

As we learned in class, lasso regularization tends to produce more zero parameters than quadratic regularization. To verify this claim, let's examine the values of parameters. Round each parameter to 0 if its absolute value is smaller than 0.01. Compare the number of zero parameters of the linear regression models with quadratic and lasso regularization.

## 1.5 What to submit

In this assignment, use $\alpha = 0.01$ and $\epsilon = 0.001$. Use $\lambda = 5$ for quadratic regularization, and use $\lambda = 1$ for lasso regularization.

1. Plot the value of loss function $J_k(\theta)$ vs. the number of iterations ($k$) for Section 1.2, report the squared loss on the test data for Section 1.2.
2. Equation for the gradient of Eq. (2).
3. Plot the value of loss function $J_k(\theta)$ vs. the number of iterations ($k$) for Section 1.3, report the squared loss on the test data for Section 1.3.
4. Numbers of zero parameters of the models obtained in Sections 1.2 and 1.3.
5. The source code (e.g., .java, .py, or .m files).