MACHINE LEARNING (IMC-4302C)

LAB 1: LINEAR REGRESSION

School Year

2017-2018

About your tutor and labs

- Slim BEN AMOR: Slim.ben-amor@inria.fr
  - Bachelor from Ecole Polytechnique of Tunisia + Master from Paris-Saclay
  - Phd Student at INRIA
  - Online courses achievements (Machine Learning from Stanford and AI from Berkley University)

- 10 Labs:
  - 2 labs per week
  - 1 report per week (your code for 2 labs + 2-3 pages for your comments/observations on each question)
  - Grade: 5 reports 50% + Final project 50%

Notation

$$X = \begin{bmatrix} x_{1,0} & x_{1,1} & \cdots & x_{1,n-1} \\ x_{2,0} & x_{2,1} & \cdots & x_{2,n-1} \\ \vdots & \vdots & \cdots & \vdots \\ x_{m,0} & x_{m,1} & \cdots & x_{m,n-1} \end{bmatrix}$$

$$x_i = \begin{bmatrix} x_{i,0} & x_{i,1} & \dots & x_{i,n-1} \end{bmatrix}$$

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_{n-1} \end{bmatrix}$$

Where:
- m is number of samples
- n is number of features including y-intercept (bias)

$$h_\theta(x_i) = \theta_0 + \theta_1 x_{i,1} + \theta_2 x_{i,2} + \cdots + \theta_{n-1} x_{i,n-1} = \sum_{j=0}^{n-1} \theta_j x_{i,j} = x_i \theta$$

$$h_\theta(X) = X\theta$$

Cost Function: Mean Square Error (MSE)

- Cost Function

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2$$

- Partial derivative

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i) x_j$$

$$j = 0 \ldots n - 1$$

Minimizing Cost Function: Gradient descent Algorithm

- Update equation for parameters θ_j

  Where : α steplength or learning rate

$$\theta_j = \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j}$$

- The update of θ_j should be made simultaneously. Because if we update θ_1 the vector θ will change. Hence, when updating θ_2 we will calculate the partial derivative of J(θ) on a different point θ as the initial θ.
- Simultaneously update

$$\theta_j^{temp} = \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j} \quad for\ all\ j = 0 \ldots n-1$$

$$then\ \theta_j = \theta_j^{temp}$$

Vectorized Implementation

- Cost function

$$J(\theta)) = \frac{1}{2m}(X\theta - y)^\top(X\theta - y)$$

- Cost function gradient

$$\nabla J(\theta) = \begin{bmatrix} \frac{\partial J(\theta)}{\partial \theta_0} \\ \vdots \\ \frac{\partial J(\theta)}{\partial \theta_{n-1}} \end{bmatrix} = \frac{1}{m}X^\top(X\theta - y)$$

- Gradient descent update

$$\theta = \theta - \alpha \nabla J(\theta)$$

Feature normalization

- Replace x feature in the matrix X by x_norm:

$$x_{norm} = \frac{(x - \bar{x})}{\sigma_x}$$

Where : $\bar{x}$ is the mean of x

σ_x is the standard deviation