# CSE474/574: Introduction to Machine Learning(Fall 2018)
## Project 1.2: Learning to Rank using Linear Regression

### Objective

The goal of this project is to use machine learning to solve a problem that arises in Information Retrieval, one known as the Learning to Rank (LeToR) problem. We formulate this as a problem of linear regression where we map an input vector x to a real-valued scalar target y(x;w).
There are two tasks:
1. Train a linear regression model on LeToR dataset using a closed-form solution.
2. Train a linear regression model on the LeToR dataset using stochastic gradient descent (SGD).

### Introduction

Notations included: $x^i$ and $y^i$ to denote (i) th input variable and target variable respectively. So, the pair ($x^i$, $y^i$) denotes the (i) th training example. Each training example consists of n features.

When the target variable we have to predict a continuous real value, the learning problem is called Regression. It is an example of supervised learning problem. In this project, we will develop linear model for Learning to Rank problem and use Gradient Descent to learn parameters of the model and also derive a closed form solution for the parameters.

### Linear Regression

In linear regression, a linear hypothesis function hθ(x) which approximates target variable y.

$$y \approx h_\theta(x) = \theta_0 + \theta_1 x_1 + \dots + \theta_n x_n$$

Here θ's are the parameters of the model. When the context of parameters is clear, we can drop θ in $h_\theta(x)$.

To simplify the notation, we define $x_0 = 1$, so

$$h(x) = \sum_{i=1}^{n} \theta_i x_i = \theta^T x$$

Job of our learning problem is to learn these parameters. Obvious method is to choose parameters which make h(x) as close to y as possible for the training examples provided to us.

Cost function:

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

**Gradient Descent**

We want to choose θ which minimizes J(θ). A general strategy would be to start with some random θ and keep changing θ to reduce our objective function J(θ).

More specifically, Gradient descent starts with some initialization of θ and repeatedly performs the following update until convergence:

$$\theta_i := \theta_i - \alpha \frac{\partial J(\theta)}{\partial \theta_i} \qquad \forall i \in [0, 1, 2, 3, .., n]$$

Here, α is the learning rate which is a hyperparameter and we will have to tune it. In order to implement this algorithm, we will first have to calculate the partial derivative.

$$\begin{aligned}\frac{\partial J(\theta)}{\partial \theta_j} &= \frac{\partial}{\partial \theta_j} \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 \\ &= \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}\end{aligned}$$

This method looks at all training examples for every iteration of the loop.

Closed Form Solution

Another method to find parameters is to set derivative of objective function to zero and obtain required parameters without resorting to iterative algorithm. Before doing this, we quickly introduce notations for matrix.

Given a training set, we define the design matrix X to be m × n matrix as

$$X = \begin{bmatrix} (x^{(1)})^T \\ (x^{(2)})^T \\ \vdots \\ (x^{(m)})^T \end{bmatrix}$$

Similarly, let Y be the m dimensional vector containing the target labels as

$$Y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

Writing J(θ) in matrix representation, we get

$$J(\theta) = \frac{1}{2}(X\theta - Y)^T (X\theta - Y)$$

Finally, we take derivative of J(θ) with respect to θ and set it to zero.

$$\nabla_\theta J(\theta) = X^T (X\theta - Y) = 0$$

Thus, the value of θ which minimizes J(θ) is given in closed form by

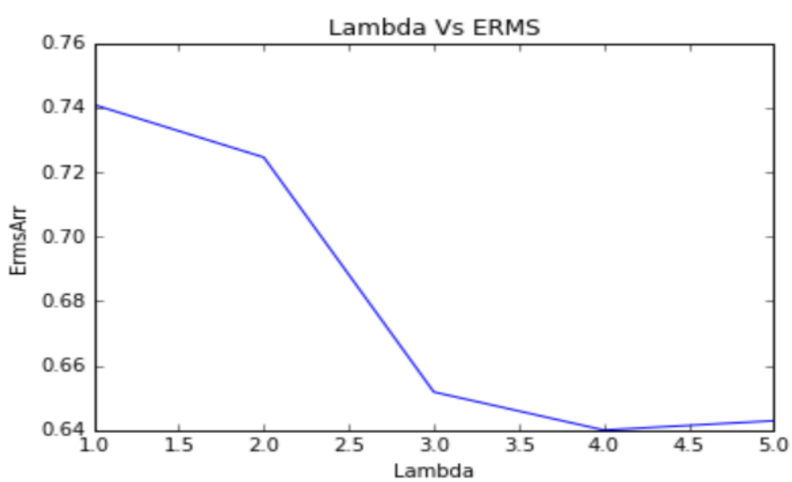
$$\theta = (X^T X)^{-1} X^T Y$$

**Experiments with the parameters for Closed form Solution:**

**1.**

| No. of Clusters (M) | Lambda | ERMS Training | ERMS Testing | ERMS Validation | Accuracy Testing |
|---|---|---|---|---|---|
| **1** | **0.1** | 0.6427529035300575 | 0.7408882410461395 | 0.6282628133169708 | 70.23416175836805 |
| **1** | **2** | 0.6291726989697519 | 0.7245673889895873 | 0.6145247378828577 | 69.71699468467175 |
| **1** | **30** | 0.5739418763580454 | 0.6518339490640811 | 0.5626546708919288 | 70.23416175836805 |
| **1** | **400** | 0.5647041154115904 | 0.6400319301310835 | 0.5539445044177406 | 70.23416175836805 |
| **1** | **5000** | 0.5649843730740433 | 0.6429119421989584 | 0.553827046136877 | 70.23416175836805 |

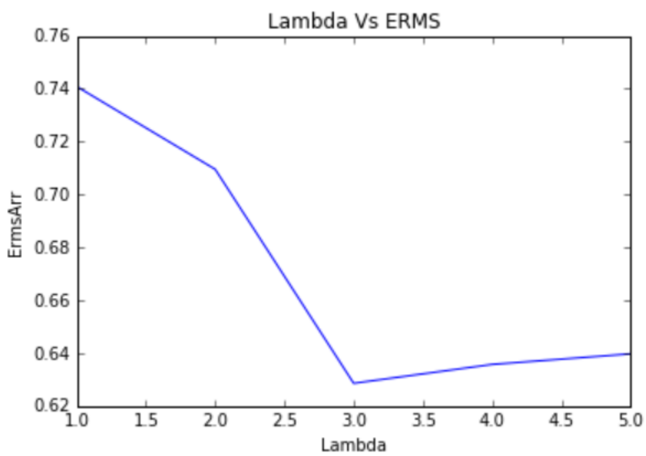
ERMS Vs Lambda Graph depicting the above readings:



**2.**

| No. of Clusters (M) | Lambda | ERMS Training | ERMS Testing | ERMS Validation | Accuracy Testing |
|---|---|---|---|---|---|
| **11** | **0.1** | 0.6427528984352141 | 0.7408882386822628 | 0.6282628125486431 | 70.23416175836805 |
| **11** | **2** | 0.6149166326086989 | 0.7095542618621802 | 0.6015941509841954 | 68.84068380979744 |
| **11** | **30** | 0.5521891466386234 | 0.628611221054567 | 0.5424701890860169 | 70.11923574199109 |
| **11** | **400** | 0.5599630818461021 | 0.6357550888641107 | 0.5488095825078251 | 70.23416175836805 |
| **11** | **5000** | 0.564400540511381 | 0.6397451836761452 | 0.5536664657038907 | 70.23416175836805 |

ERMS Vs Lambda Graph depicting the above readings:

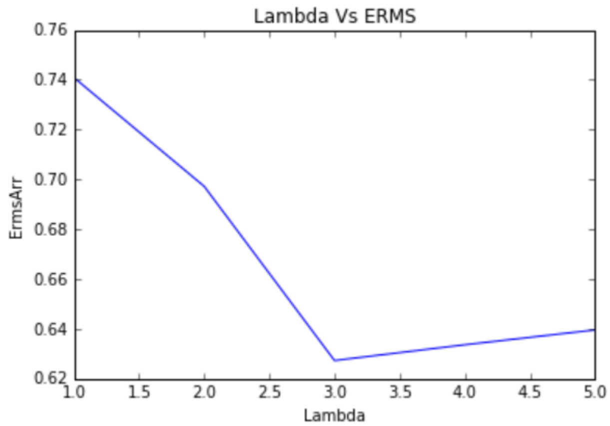



**3.**

| No. of Clusters (M) | Lambda | ERMS Training | ERMS Testing | ERMS Validation | Accuracy Testing |
|---|---|---|---|---|---|
| **21** | **0.1** | 0.642752856308148 | 0.7408882023425261 | 0.6282627430125443 | 70.23416175836805 |
| **21** | **2** | 0.6066306247051635 | 0.6972247151220713 | 0.595192864176806 | 68.82631805775033 |
| **21** | **30** | 0.5494465037368979 | 0.6273179156700758 | 0.5404984784263781 | 69.6164344203419 |
| **21** | **400** | 0.55725110939963 | 0.6336409609502813 | 0.5459526128392471 | 70.23416175836805 |

| 11 | 5000 | 0.5643879979352604 | 0.6395961142963466 | 0.5536753551556489 | 70.23416175836805 |
|---|---|---|---|---|---|

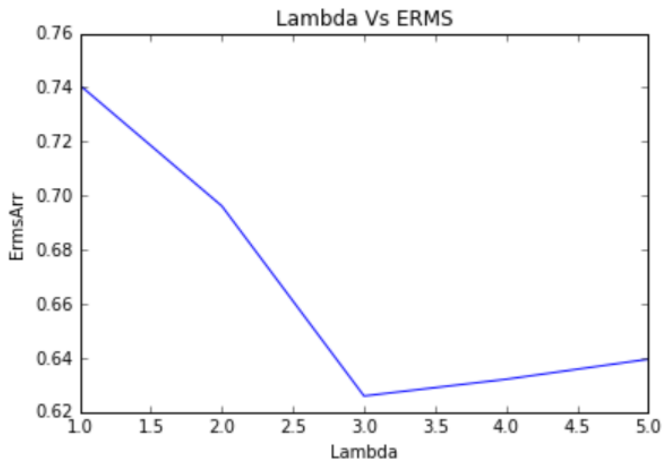
ERMS Vs Lambda Graph depicting the above readings:



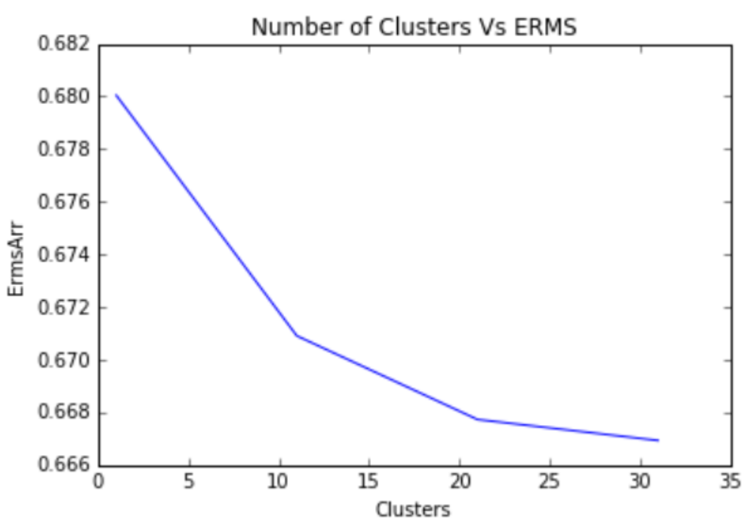

**4.**

| No. of Clusters (M) | Lambda | ERMS Training | ERMS Testing | ERMS Validation | Accuracy Testing |
|---|---|---|---|---|---|
| **31** | **0.1** | 0.6427526456427277 | 0.7408877073328477 | 0.628262648148756 | 70.23416175836805 |
| **31** | **2** | 0.6045979058074382 | 0.6961726952065399 | 0.5930665114795357 | 68.76885504956185 |
| **31** | **30** | 0.5468893499553777 | 0.6259465577763952 | 0.539374114865649 | 69.58770291624766 |
| **31** | **400** | 0.5555316953032732 | 0.6395449437346774 | 0.553682855936783 | 70.23416175836805 |
| **31** | **5000** | 0.5643850807473036 | 0.632132218388237 | 0.5443296773299638 | 70.23416175836805 |

ERMS Vs Lambda Graph depicting the above readings:




Graph Depicting ERMS Vs No.of clusters:

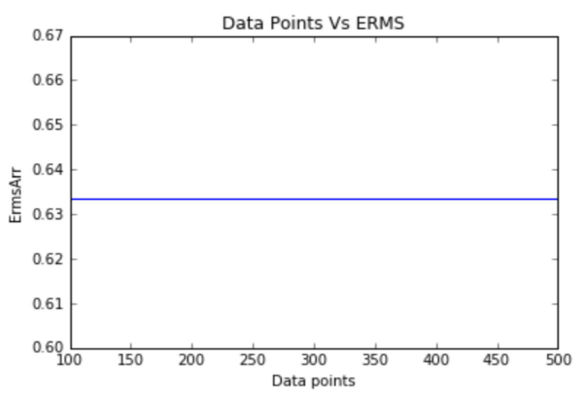


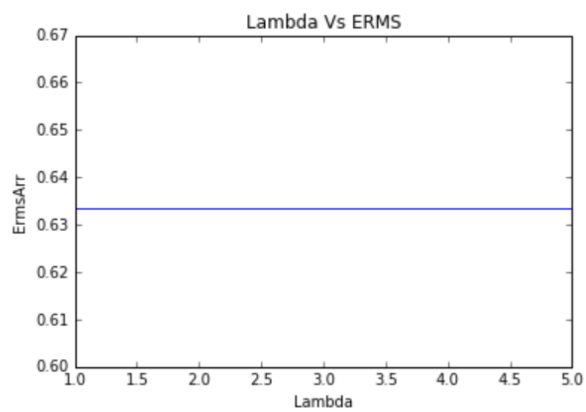

Arooshi Avasthy arooshia@buffalo.edu

**Experiments with the parameters for Gradient Descent:**

.

| No. of Data points | Lambda | Learning Rate | ERMS Training | ERMS Validation | ERMS Testing |
|---|---|---|---|---|---|
| **100** | **1** | 0.1 | 0.56255 | 0.55183 | 0.63338 |
| **200** | **2** | 0.02 | 0.56255 | 0.55183 | 0.63338 |
| **300** | **3** | 0.003 | 0.56255 | 0.55183 | 0.63338 |
| **400** | **4** | 0.0004 | 0.56255 | 0.55183 | 0.63338 |
| **500** | **5** | 0.00005 | 0.56255 | 0.55183 | 0.63338 |




Arooshi Avasthy
arooshia@buffalo.edu