Learning to Rank using Linear Regression

Dhayanidhi Gunasekaran
University at Buffalo
dhayanid@buffalo.edu

Abstract

To solve Learning to Rank (LeToR) problem using linear regression.

1 Project Statement

To return the correct match of document for the given input query. This is an information retrieval problem in which queries are matched against the document from the Gov2 web page collection. LeToR dataset is obtained from the Microsoft which consists 46 input features and the output are classified into three categories. The objective is to use linear regression with basis function to train the model and classify the input into the target categories. There are two tasks while training the model, one is to use Closed form solution and the other one is to use stochastic gradient descent to optimize the model. The target value should be scalar which should be of the form 0,1,2.

2 Solution

The first step is to get the input data and process it so that it can be used in training the linear regression. The given dataset has 46 input features with some unwanted features such as query ID, Doc Id and so on. The first step is to cluster the input features using K Means clustering Algorithm. So that the input size is reduced to the cluster size say M. The next step is to convert input features x0,x1..Xm into a scalar value. This is achieved by using the M basis function. The linear regression function formula is

$$y(\mathbf{x}, \mathbf{w}) = \mathbf{w}^{\top} \boldsymbol{\phi}(\mathbf{x})$$

Gradient radial basis function is used to convert the given x vectors into a scalar value (phi).

$$\phi_j(\mathbf{x}) = \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_j)^{\top} \Sigma_j^{-1} (\mathbf{x} - \boldsymbol{\mu}_j)\right)$$

Where sigma is the variance between the points and Mu is the center of the cluster.

Once the phi matrix is generated then the weight can be computed. There are two ways in which the weight is computed in this solution.

1.Closed form solution.

2. Stochastic Gradient descent.

2.1 Closed Form Solution

The closed form solution of w is nothing is but the sum of squared errors. Which is of the following form

$$\mathbf{w}^* = (\lambda \mathbf{I} + \mathbf{\Phi}^\top \mathbf{\Phi})^{-1} \mathbf{\Phi}^\top \mathbf{t}$$

We use Moore-Penrose pseudo inversion of the matrix(phi) to find the w, where lambda is the regularization parameter.

2.1 Stochastic Gradient Descent solution

The stochastic gradient descent takes a random initial value for w. then the value is updated by using the below formula

$$w(\tau+1) = w(\tau) + \Delta w(\tau)$$

where $\Delta w(\tau) = -\eta(\tau) rE$ is called the weight updates and $\eta(\tau)$ is the learning rate.

$$\nabla E = \nabla E_D + \lambda \nabla E_W$$

in which

$$\nabla E_D = -(t_n - \mathbf{w}^{(\tau)\top} \phi(\mathbf{x}_n)) \phi(\mathbf{x}_n)$$
$$\nabla E_W = \mathbf{w}^{(\tau)}$$

3 Conceptual Understanding

Input data which consists nearly 69000 datasets which is divided into training, validation and test sets. The training set comprises of 80% of raw data while the validation and test data consist 10% each. The initial data processing is done by removing unwanted data such as query id, doc id and other unwanted columns.

In order to train the data in linear regression model, the input feature has to be scalar, however the given dataset has 41 features which needs to be scaled down to a scalable value.

3.1 K – Means Clustering

Since the input data in unlabeled use K – Means Clustering to group the data. It is a type of unsupervised learning which aims in finding the groups in an unlabeled data input. It works in a way to assign each data into a specific group. The center of M clusters are used as label to the input data.

After using K – Means algorithm we have M centers in the input feature set.

The graph showing 10 centroids is provided below.

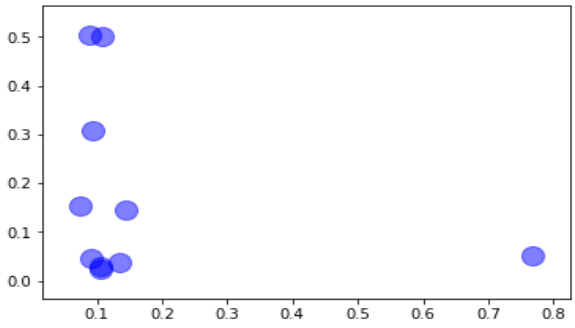

Figure 1: K means clustering with 10 Centroids

3.2 M Basis function

Now that we have the dataset labelled based on the centroids. We need to convert the input feature vector X into a scalable value which can be achieved by using Basis Function.

The input dataset is of the shape 69000 * 41 which is visualized as

	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	...	0.33	0.34	
count	69622.000000	69622.000000	69622.000000	69622.000000	69622.000000	69622.0	69622.0	69622.0	69622.0	69622.0	...	69622.000000	69622.000000	696
mean	0.161690	0.142535	0.252800	0.146084	0.165257	0.0	0.0	0.0	0.0	0.0	...	0.635138	0.569768	
std	0.234703	0.255635	0.341221	0.311805	0.234518	0.0	0.0	0.0	0.0	0.0	...	0.269526	0.272595	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.0	0.0	0.0	0.0	0.0	...	0.000000	0.000000	
25%	0.019157	0.000000	0.000000	0.000000	0.021931	0.0	0.0	0.0	0.0	0.0	...	0.470615	0.383290	
50%	0.063291	0.000000	0.000000	0.000000	0.067821	0.0	0.0	0.0	0.0	0.0	...	0.693540	0.613330	
75%	0.190467	0.200000	0.500000	0.000000	0.196013	0.0	0.0	0.0	0.0	0.0	...	0.851457	0.782597	
max	1.000000	1.000000	1.000000	1.000000	1.000000	0.0	0.0	0.0	0.0	0.0	...	1.000000	1.000000	

8 rows × 46 columns

Figure 2: Showing the raw data

Radial basis function is a real function which determines the distance of each point from the center, in this problem the center is the centroids found using K Means Clustering. Also it converts the input vector x into a scalable value.

The formula of linear regression with radial basis function is provided below.

$$y(\mathbf{x}, \mathbf{w}) = \mathbf{w}^{\top} \boldsymbol{\phi}(\mathbf{x})$$

Where w is the weight which needs to be computed by training the model while φ is the vector of M basis function. The φj(x) computation formula is provided in the previous section.

After computing the Phi value, the data set is of the shape 69000 * 10, where 10 is the M of this basis function.

Now that we have Input feature set, and an output target vector. We need to compute the Weight of each input feature.

The weight computation can be done by two methods.

1. Closed form solution
2. Stochastic gradient descent.

3.3 Closed form Solution

The closed form solution for computing the weight formula is provided in the section 2.1. Now that the weight is computed, the noise in the function has to be found and reduced which is achieved by assuming the output function has a normal distribution, the noise can be reduced by finding the sum of squared differences.

$$E_D(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{t_n - \mathbf{w}^{\top} \boldsymbol{\phi}(\mathbf{x}_n)\}^2$$

Where t is the target vector and W is the weight computed using Moore Penrose Pseudo Inverse Matrix. After computing the weight, the hyperparameters can be tweaked to improve the efficiency of the model

3.3.1 Tuning M value

After computing the weight, the efficiency can be calculated using the Root Mean Square (RMS). The Efficiency of various values of M is shown in the graph below.

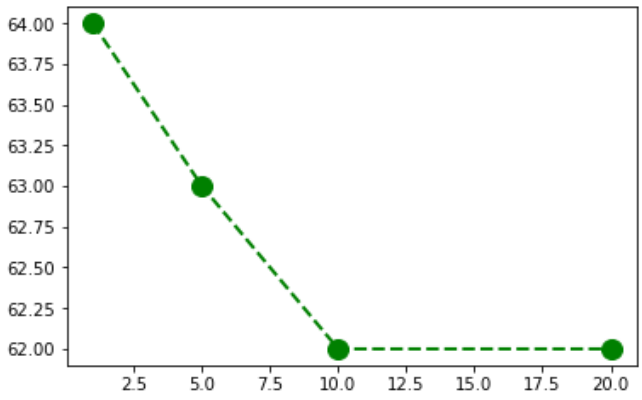

Figure 3: M values Vs Efficiency

For the given dataset the M value reaches the threshold at the value of 10. The Efficiency of 62 % is achieved at this value without tuning the hyper parameters such as Learning rate, sigma value.

M (no of clusters)	**Testing Efficiency (%)**
1	64
5	63
10	62
20	62

Table 1: No of clusters Vs Efficiency

3.3.2 Tuning sigma value

The next hyper parameter to tune is the sigma value. The sigma value governs the spatial scale of the dataset, higher the value of sigma broader the basis function would be. Hence the sigma is computed by finding the variance of x and scaled with a lower number say $1/10^{th}$.

The efficiency found across various sigma value is plotted as a graph and shown below.

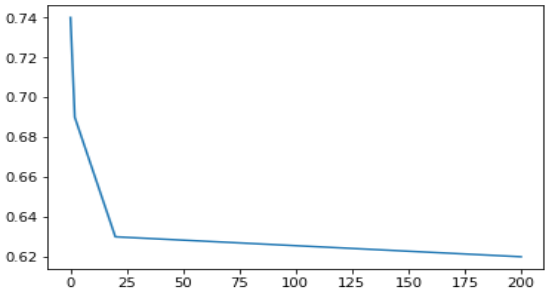

Figure 4: Sigma vs Efficiency

Efficiency is comparatively higher when sigma value is a fraction of x than then the larger sigma value. The values for each sigma value for constant M and learning rate is shown below.

Sigma	Testing Efficiency (%)
200	0.62
20	0.64
2	0.69
0.1	0.74

Table 2: Sigma vs Efficiency

3.3.3 Tuning Regularization coefficient

The regularization coefficient (λ) is used to avoid overfitting and provide generalization. The regularization coefficient shrinks the coefficient estimate to zero. The error function with regularization coefficient is given below.

$$E(\mathbf{w}) = E_D(\mathbf{w}) + \lambda E_W(\mathbf{w})$$

Where Ew is the weight decay regularizer. The model is trained using various values of λ and the output is given in the graph below.

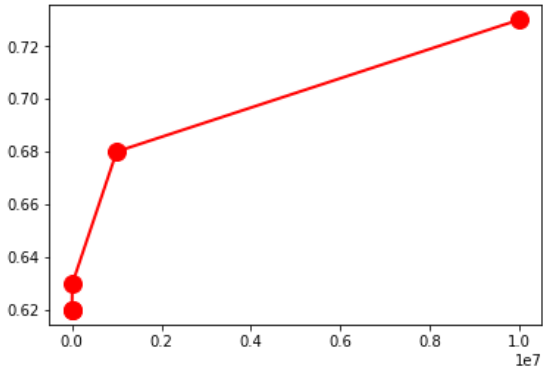

Figure 5: Lambda vs Efficiency

The values for various lambda value with M =10 and sigma = 200 are given in the table below.

Lambda	Testing Efficiency (%)
0.1	0.62
100	0.62
10000	0.63
1000000	0.68
100000000	0.73

Table 3: lambda vs Efficiency

3.3.4 Tuning Regularization coefficient

After tuning the various hyper parameters, the final value which gives the best efficiency is as follows.

Hyper paramer	Value
M	10
sigma	0.1
Lambda	1000000

Table 4: Final Parameter value

3.4 Stochastic Gradient Descent

Stochastic gradient descent is another method for finding the weight, in which the initial weight is set to some random value. On subsequent iterations, it computes the difference between the predicted value and the actual output and determines the weight based on the learning rate. The iteration is repeating till the model reaches the convergence point beyond which there would not be drastic change in the cost improvement.

The formula used for computing the weight by stochastic gradient descent is given in the section 2.1. Now the model is trained by changing the learning rate and calculating the accuracy.

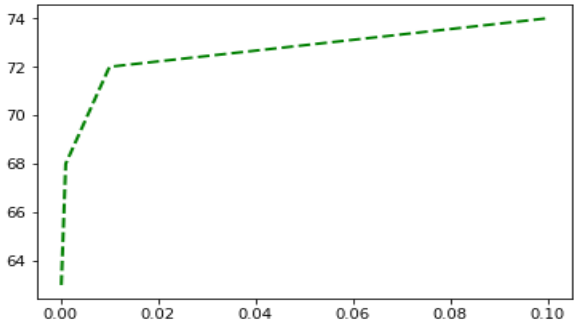

Figure 6: Learning rate vs Accuracy

Learning Rate(λ)	Testing Efficiency (%)
0.0001	63
0.001	68
0.01	72
0.1	74

Table 5: Learning rate vs Accuracy

References

[1] https://cedar.buffalo.edu/~srihari/CSE574/Chap3/3.1-Regression-BasisFns.pdf, Linear Model of Regression, by Sargur Srihari

[2] https://matplotlib.org/api/api_overview.html

[3] https://www.datascience.com/blog/k-means-clustering

[4] https://en.m.wikipedia.org/wiki/Moore–Penrose_inverse

[5] https://en.wikipedia.org/wiki/Radial_basis_function_kernel