**REGRESSION**

**ON**

**PAGE RELEVANCY**

*-By*

*Utkarsh Srivastava*

(u2@buffalo.edu)

CSE-574, Introduction to Machine Learning

**GOAL :**

The goal if the project is to: Train a regression model based on query-url pair datasets , then predict the page relevancy labels for new coming queries. Therefore for a given data set with multiple feature vector, A Regression model, that analyses this set of input data and comes up with a function that can accurately predict the output, must first be implemented.

The Regression model takes a set of feature vectors as input. Data is then parsed into a machine readable format and imported. The Linear Basis function model employed in supervised learning is them applied to the data set.

**Linear Basis function model**

**y(x,w)= Φ(x)w**

Φ0(x) is kept as 1 to promote bias parameter.

By varying the hyper-parameters; with various values of Basis function M, the deviation s and λ for regularization term, We try to realize the most accurate values of w by maximum likelihood solution (via Gaussian/Sigmoid) and Stochastic gradient descent.

We then test our model on test data, document and statistically represent the value of errors

**DATA SET:**

LETOR is a package of benchmark data sets for research on Learning Rank released by Microsoft Research Asia.To

The latest version, 4.0, can be found at

http://research.microsoft.com/en-us/um/beijing/projects/letor/letor4dataset.aspx

(It contains 8 datasets for four ranking settings derived from the two query sets and the Gov2 web page collection).For this project, one dataset of MQ2007 is used (supervised ranking)

**PROCESSING DATA SET**

Given a query and a document, construct a feature vector (normalized between 0 and 1). There are 46 features provided and 69000 records. As part of our Project, we partition data into three subsets

1.Training

2.Validation

3.Testing

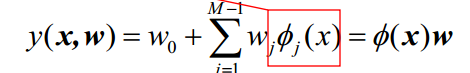
The data is also *parsed* into a machine readable format by splitting the relevancy labels, from the 46 feaure vectoes and pruning the serial np., queryId and docId.

2 ~~qid:10002~~ 1:0.007477 2:0.000000 3:1.000000 4:0.000000 5:0.007470 … 46:0.007042 ~~#docid =~~

~~GX008-86-4444840 inc = 1 prob = 0.086622~~

**Linear Regression (by Gaussian Basis Function)**

Linear Regression by Gaussian method is provided by the formula

Where

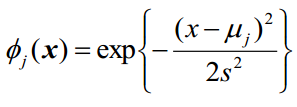
y(x,w) is the output

M is the complexity, which is assumed to be a random number between 1 to 46.

D is the dimensions

W: the weight vector of order M\*1

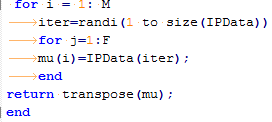
Φ(x) is the linear basis function model variable design matrix, and can be computed by various models. the Gaussian method has been used here .

  
Where

μ is the mean

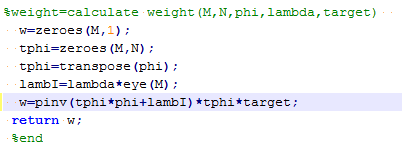
and s is the std. deviation.

Since x is a 46 feature vector here. M instances of data set are taken as mean from input data M and s is also assumed initially to have a value 0.5.



We then compute the value of w and calculate Erms for various values of lambda, s and M with formula



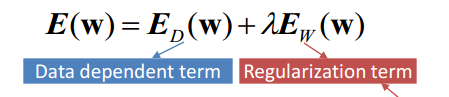


Here, Lambda is the smoothening parameter that, we introduce in order to prevent curve from “over-fitting the data”.

**Over-Fitting**

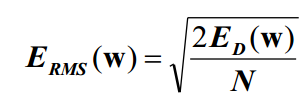
If one continues to train and improve the model according to the training data, The output function might ultimately end up behaving exactly according to the needs of the training data , But may fail to predict the output while testing, which uses a different set of inputs. This is called over-fitting, We use regularization and heavy number of training examples to combat Over-Fitting. This involves introduction of regularisation term to weights while computing error.

(As followed earlier with hyper-parameters and complexity, We have tried to compare the Error (or the level of accuracy) for various values of lambda during validation.)

****

Finally, The root mean square error is computed over the training data as the square root of the data dependent error

**Error Calculation**

****

Once, Erms is computed, the values of hyper parameters , M and λ etc are tuned accordingly.

**RESULTS :**

**\_** =0.1 , ERMS(Training)= 0.540406 ERMS(Testing)= 0.588986 , M=24

**Tables** 

Training Error Validation Error Testing Error

M= 2 0.555443 0.543018 0.617894

M= 3 0.553969 0.535443 0.613223

M= 4 0.549411 0.529073 0.606980

M= 5 0.547693 0.528574 0.606672

M= 6 0.545576 0.522191 0.600586

M= 7 0.545819 0.524433 0.603552

M= 8 0.545162 0.523040 0.601437

M= 9 0.545937 0.524175 0.602322

M= 10 0.544961 0.520315 0.599221

M= 11 0.544393 0.520916 0.599012

M= 12 0.544542 0.520240 0.599312

M= 13 0.542649 0.518041 0.594443

M= 14 0.545142 0.523020 0.600650

M= 15 0.543700 0.519896 0.598705

M= 16 0.544179 0.520367 0.598644

**Gradient descent**

This is another method for weight calculation associated with each feature vector of the input data.

Weight essentially quantifies the emphasis each feature of the input vector has wrt the output.



Where,

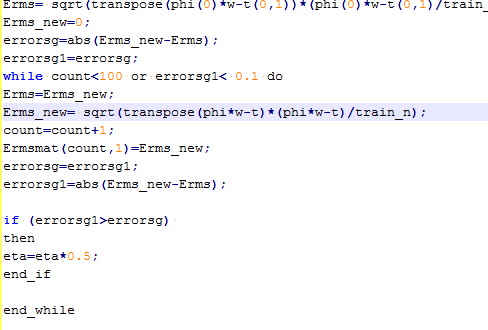
W τ+1 : the improved weight vector of order M\*1

W τ : the current vector , assume some random values initially

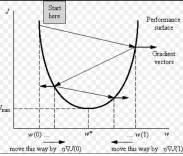
tn  : Target for the n-th row

Φn : M\*1 dimensional, till the n-th row

This is an iterative process where the value of weight is improved using the Backtracking learningrate “eta”. Everytime, we realize that the error is not decreasing , the value of η is reduced by 0.5, to decrease the influence of the expected dependent error on the weight vector’s value.



Upon iteration , we finally realize a stable W τ+1 , which does not show significant improvements in the value. The same is graphically implied as follows



Erms :The final values comes up to be .58863, M =26