Project Report

Project 1: *Linear Regression with Basis Functions*

CSE 574

Subhendu Saha

UBit: subhendu

UB# 5009 7223

Overview

This project is to implement regression on web search relevance data set LETOR 4.0. We are given a dataset of vectors and target values. There are 46 features for each sample in the dataset. We need to learn the weight of each feature which can give us the best result given a new sample. We use the technique of regression to learn these weights by using the samples given. We use the Gaussian basis function and vary the degree or complexity of the model until it gives us the closest prediction we can get on an unseen test dataset.

Objective

The project is uses two linear regression models - maximum likelihood solution and stochastic gradient descent to rank web search data. In both methods, regularization is used to reduce the effect of over-fitting. Dataset used for the project is Microsoft LETOR. It has 46 feature vector for each data point.

Models Used

The first model uses Maximum Likelihood Estimator using the Least-square method.

$$y(x, w) = w_0 + \sum_{j=1}^{M-1} w_j \Phi_j(x)$$

where the basis function Φ is of the form $\Phi_j = \exp\left(\frac{-(x-\mu_j)^2}{2s^2}\right)$

In our case ***x*** is not a single value but a vector with **46** dimensions.

In order to solve we need to determine ***w*** vector and apply the first equation to predict ***y*** values. Using least-square method with regularization coefficient we can calculate ***w*** vector using the following formula.

$$w = (\varphi^T \varphi + \lambda I)^{-1} \varphi^T t$$

Now we need to calculate Φ. For that we need to come out with a model complexity value ***M.***

Depending on the value of M, our Φ matrix will be of dimension ***N x (M*D + 1)***. N being the total number of data points and D being the dimension of feature vector. The one extra column is because $\Phi_0(x) = 1$ for all values of *x*.

For our second model, we have used stochastic gradient descent. Instead of learning about the whole dataset at once, it fine tunes the knowledge of ***w*** by learning about one data-point at a time.

$$w^{(\tau+1)} = w^{(\tau)} + \eta\left(t_n - w^{(\tau)^T}\varphi_n\right)\varphi_n$$

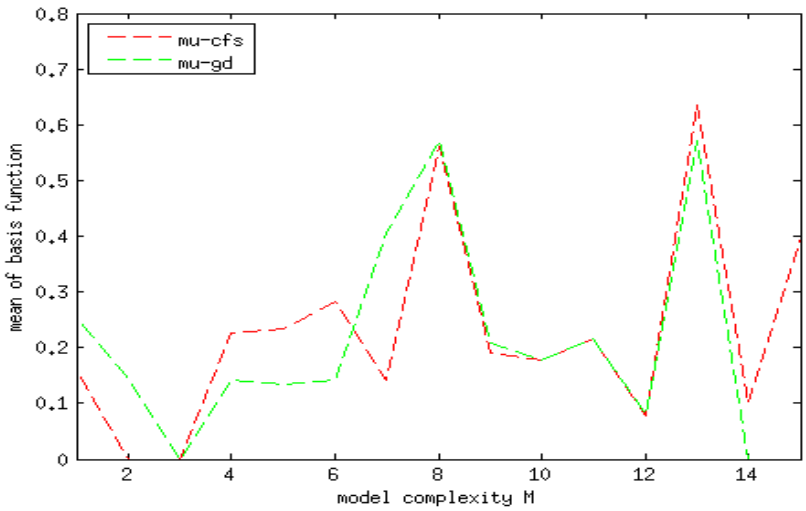
we need to specify a ***threshold*** value which upon reaching, the model will quit gradient descent and is presumed to at a local minimum.

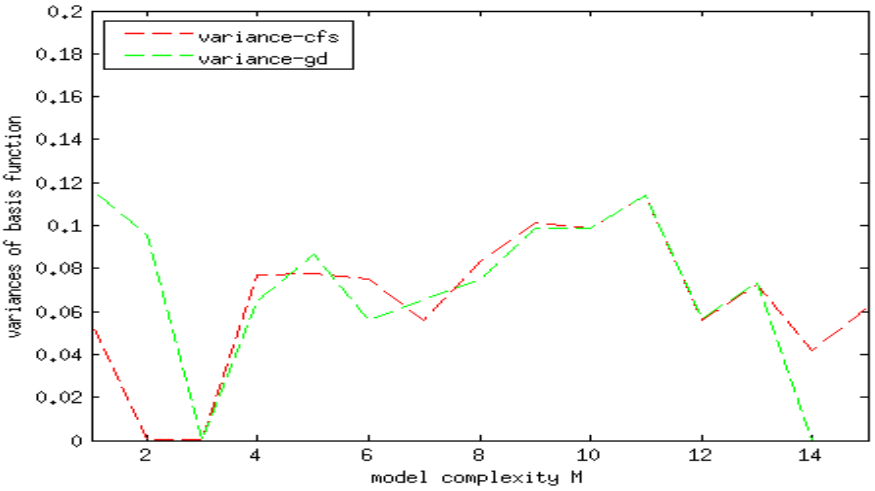
Dataset

For our project, we have used the Microsoft LETOR 4.0 dataset. For our supervised learning algorithm we have used the "*Querylevelnorm.txt*" . The file contains 69623 rows of formatted and normalized data. Next we partitioned the dataset into 8:1:1 ratio between training set, validation set and testing set.
Our goal is learn the ***w*** vector by applying our model to the training dataset. Then fine tune (remove overfitting, select model-complexity, adjust hyperparameters) the vector using our validation set. And finally predict the values in testing set using our refined model and report the rms error.

Selecting Model Parameters μ_j and σ_j

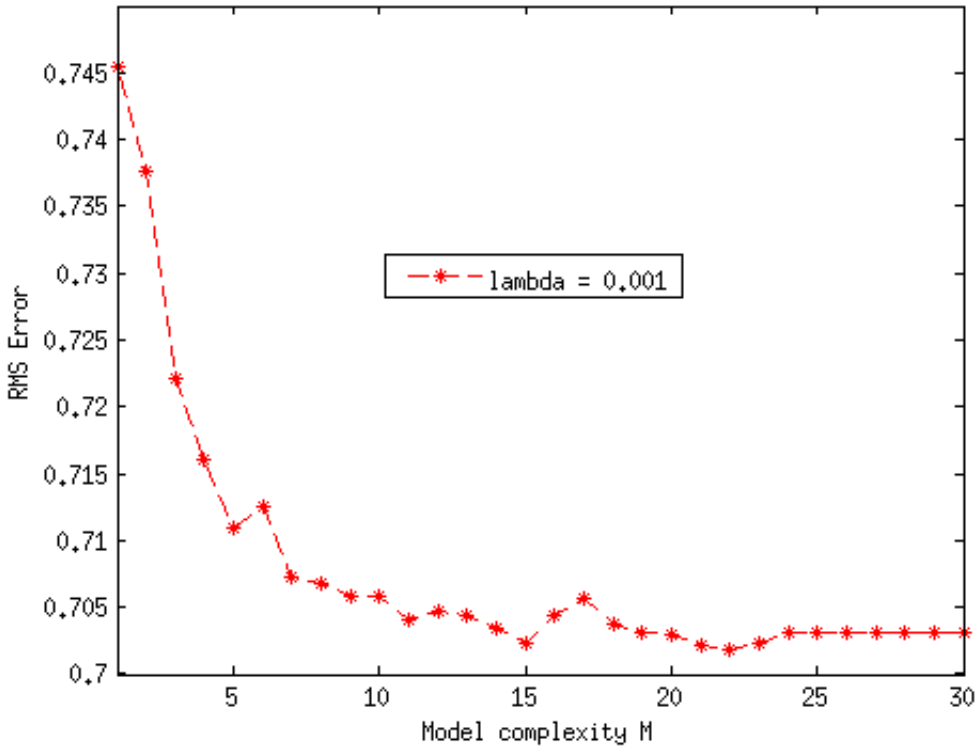
One important aspect of the project is to choose correct values for ***mu*** and ***sigma*** for the basis functions. For this case, we have partitioned the feature matrix ***X*** into ***M*** partitions. Then we have randomly selected a feature from each partition and computed its ***mu*** and ***sigma.*** In case of ***sigma,*** we added a small offset value because there are features in the dataset for which the value is **zero** for all data points.



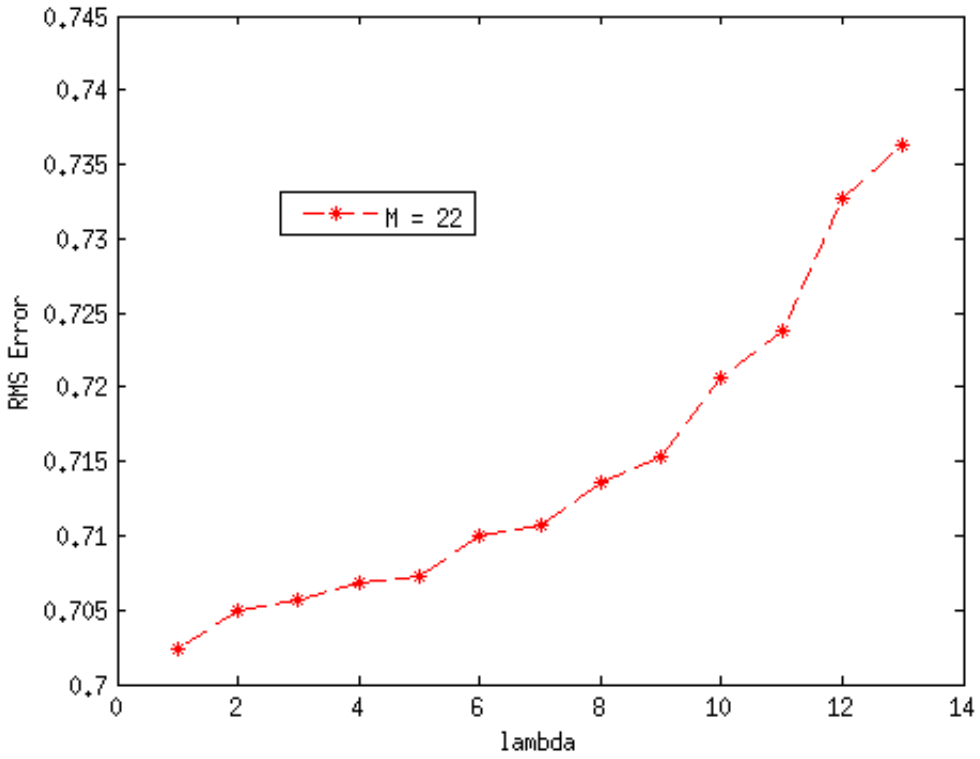


Selecting M and λ

At first we vary ***M*** and keep ***λ*** fixed. Then we can observe the following trend. We see the RMS Error reaches a minimum value at ***M = 22.*** If we increase the value of M then the error also increase and becomes constant. If we decrease M, then also the error keeps increasing.



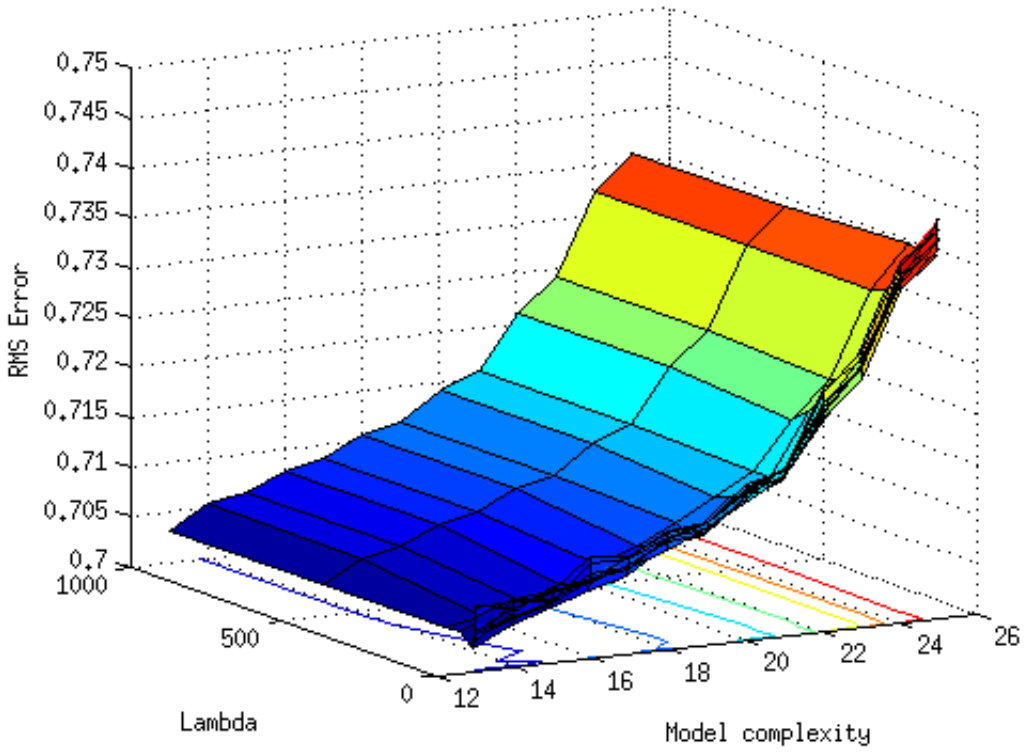
Next we keep ***M*** fixed at 22 and vary ***λ***. We can observe the following trend. As we keep on increasing ***λ,*** the RMS Error also increases. We get the smallest RMS Error at ***λ*** = ***0.001.***



Thus from above discussions we can see that our model reaches optimal value at ***M*** = ***22*** and ***λ*** = ***0.001***

Using similar reasoning in case of gradient descent model we find that it is optimal at ***M*** = ***13*** and ***λ*** = ***0.01***

Following surface plot gives an idea when both ***M*** and ***λ*** are varied together.



Result Reporting and Comparison

The outcome of the learning model is reported in terms of estimated error. The rms error is calculated using the formula $E_{rms}(w)=\sqrt{\frac{2E_d}{N}}$

Following are the estimated results

$E_{rms-validation}\,cfs = 0.70$ $E_{rms-testing}\,cfs = 0.81$

$E_{rms-validation}\,gd = 0.81$ $E_{rms-testing}\,gd = 0.92$

We can see that the closed form solution does a little better that the gradient descent model. But time requirement for closed form solution is more than gradient descent. Also gradient descent can be applied to cases when we do not have the full data available for processing since it is an incremental learning algorithm.