# Learning to Rank using Linear Regression

Haoquan Zhao

50189594

haoquanz@buffalo.edu

# 1. Abstract

The main goal of this project is to use machine learning linear regression techniques to solve the Learning to Rank (LeToR) problems on a real dataset and a synthetic dataset. We train our model using the training set which comprises of 80% of the given data set and validate it using the 10 % validation data set and finally test it using the 10% testing data set.

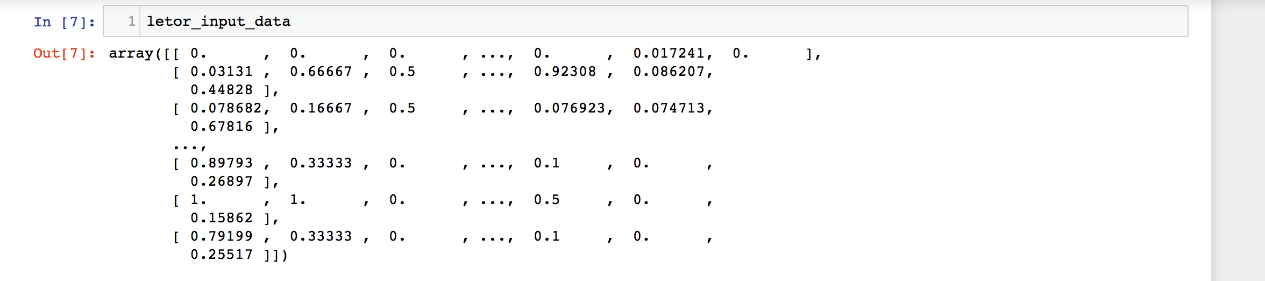
# 2.Introduction.

In this project, we apply linear regression on a synthetic dataset from microsoft LETOR 4.0, train our model on part of it and evaluate the performance on another part, then by tuning Hyper-Parameters, we get to know with which value of k can we get the minimum error. And apply this into test part of dataset. And finally, by using Root Mean Square error to evaluate the solution on a test part of the dataset, we could find out how dose the solution work.

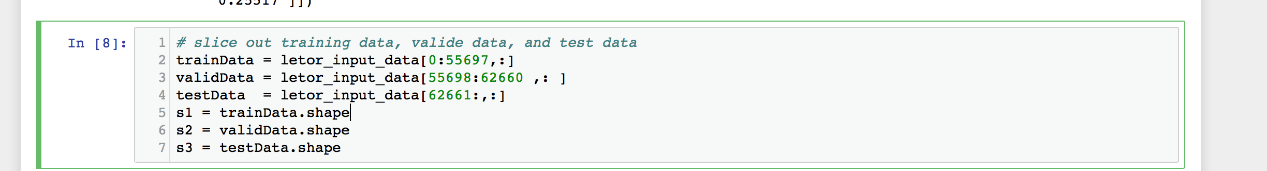
# 2. LeToR Dataset



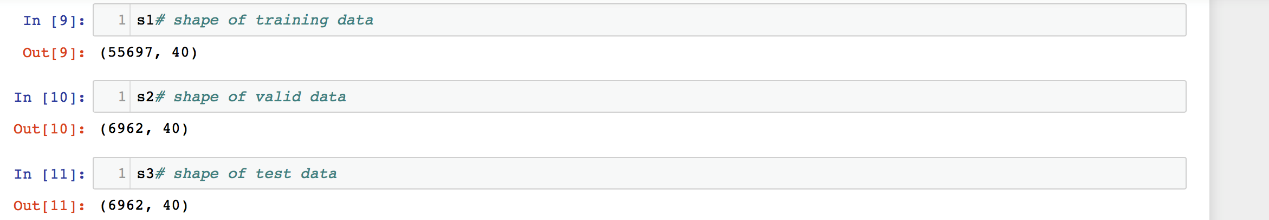
In this part, we use numpy function to deal with the dataset we need to use to do LeToR. By using genfromtxt function in numpy, we could generate pure dataset from the orignal dataset.

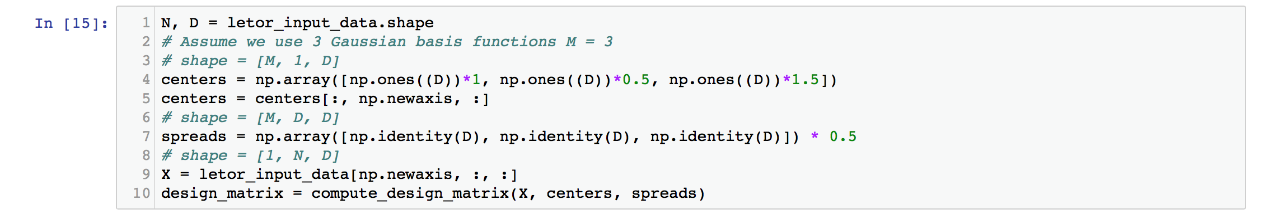


After dealing with the dataset, what we need to do is, do partition on the dataset, so that we could use these data to train and test our model.

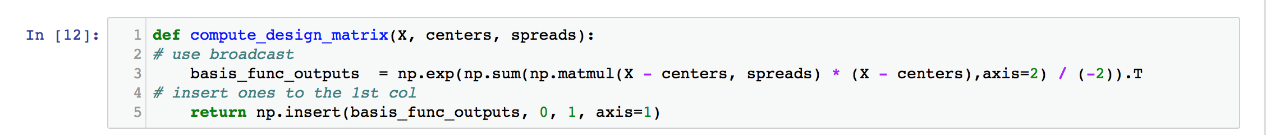


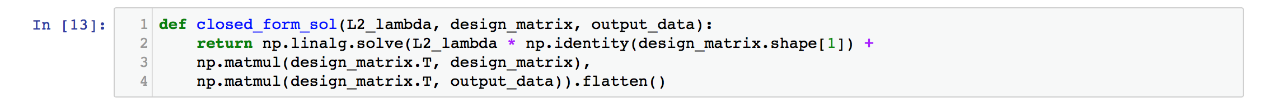
In this part, we do Partition to dicide the whole dataset into three parts. The first one is used to train our model, so that we could generate the optimal parameter set as expected. The second part is called valid part, which is used as a confrimation that the model we build by training the train part dataset is reliable. The last part is reserved as test dataset, with which we could evaluate the efficiency of our model.



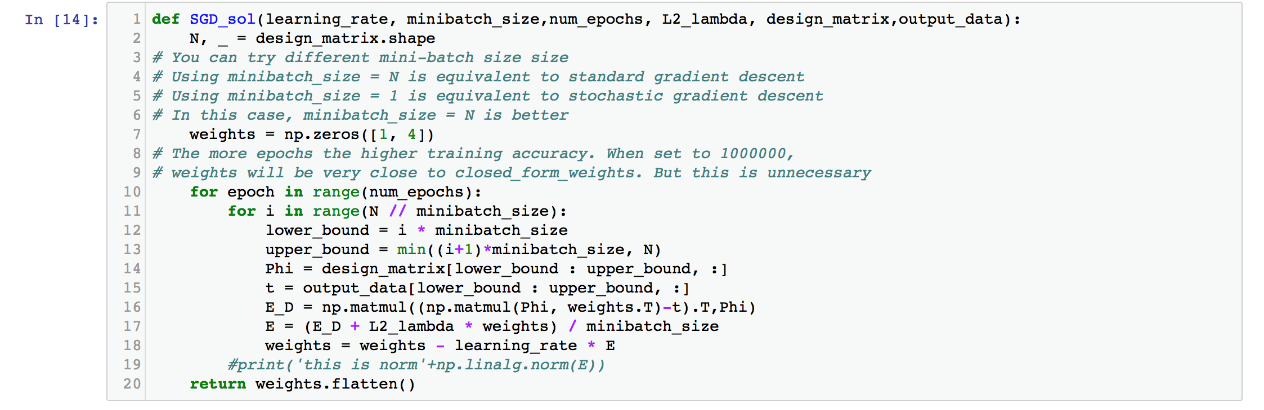
80% of the dataset is used as train data, 10% of the dataset is used as valid data, and the rest 10% is used for test. In the dataset, there are 69623 query-document pairs each consisting of 46features.

In this part, we randomly pick up 3 basis functions, and set centers and spreads of them. This process could also be accomplished by K-means function.

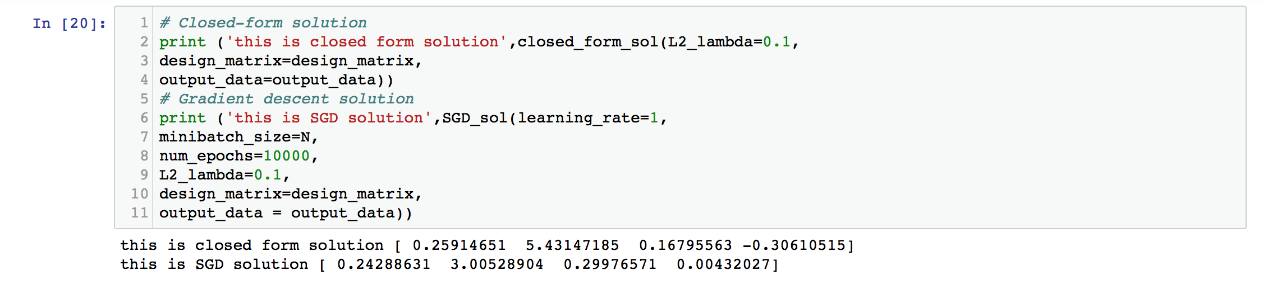




Then we compute the design\_matrix to get the variance, and define closed\_form\_solution and apply them onto our train data.



Then we define how dose SGD works in the dataset, firstly, wetake a random initial value and by using , we get to know how those values changes in the whole dataset. In which, we also get , which is called weight update, it gose against with direction of gradient of the error.



Then we print out the result generated by the two different solutions, and we could compare them with each other.

# 3. Synthetic Data

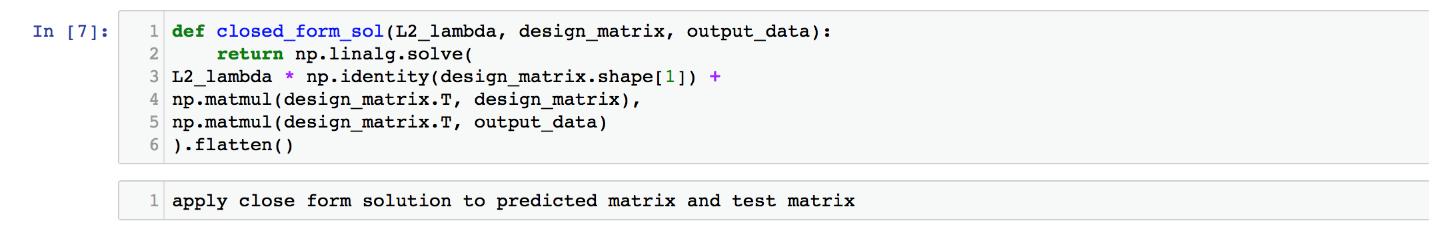
# Processing the data from the data set:

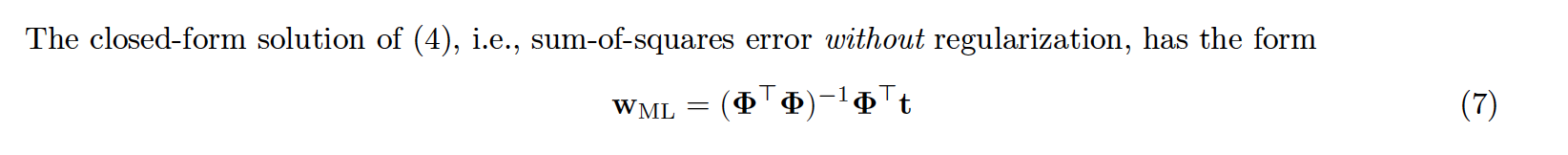


# Compute the design matrix /Users/Juno/Desktop/Screen Shot 2017-10-23 at 9.35.41 PM.png

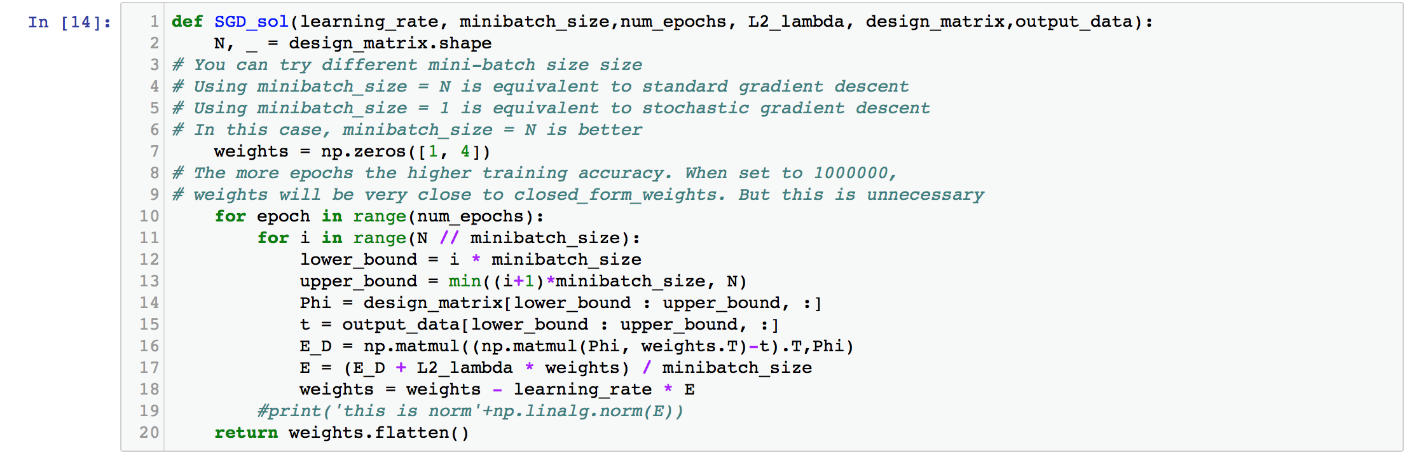
In order to predict the result, each basis take one data point, but for computation efficiency，we need to perform row-wise dot product. So this function stack all centers and spreads together and use broadcast to compute the design matrix in one statement.

# Compute closed form solution

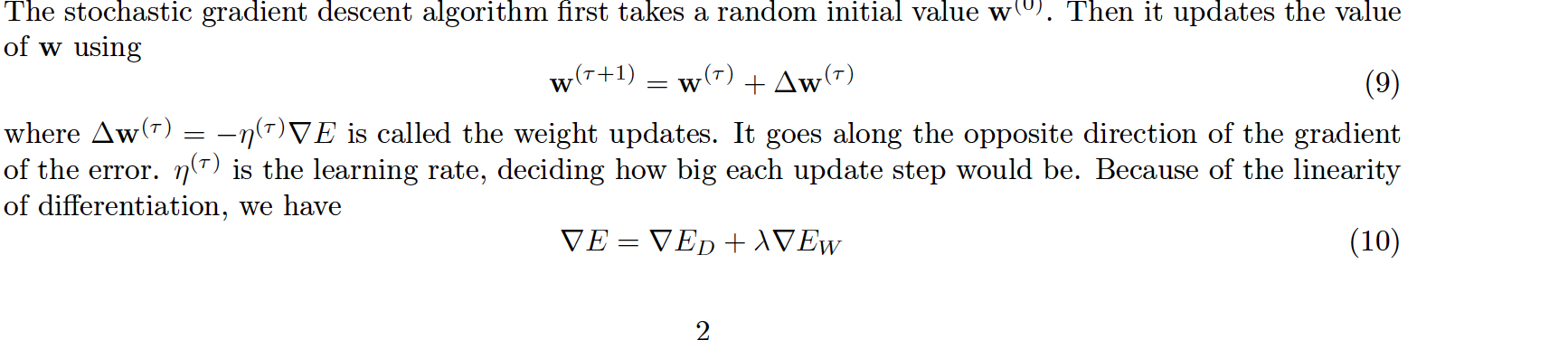


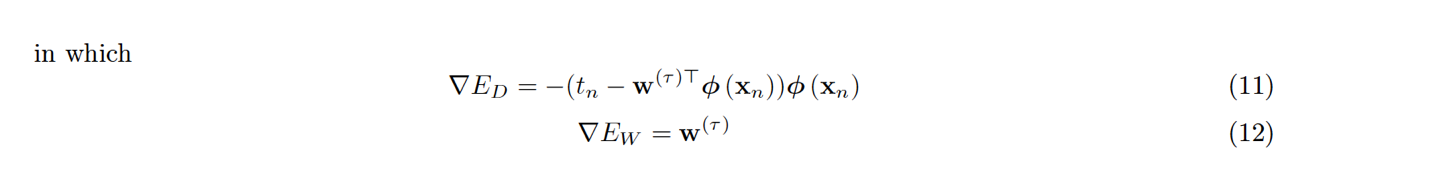
the closed form solution is given as 

# Compute Gradient descent solution

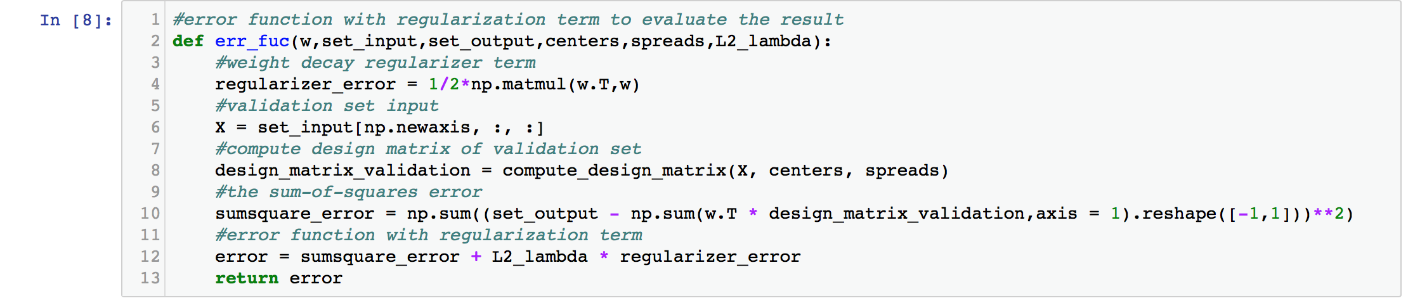


and the form is given as :

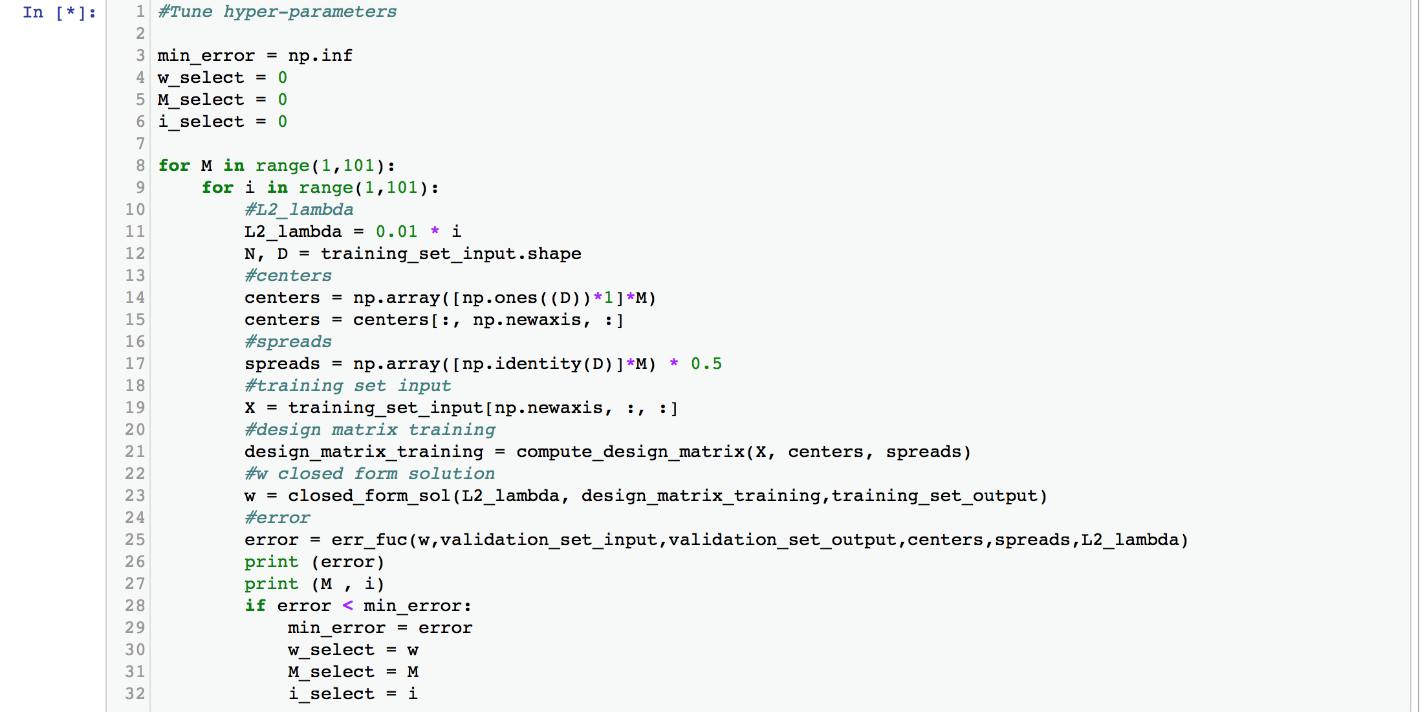




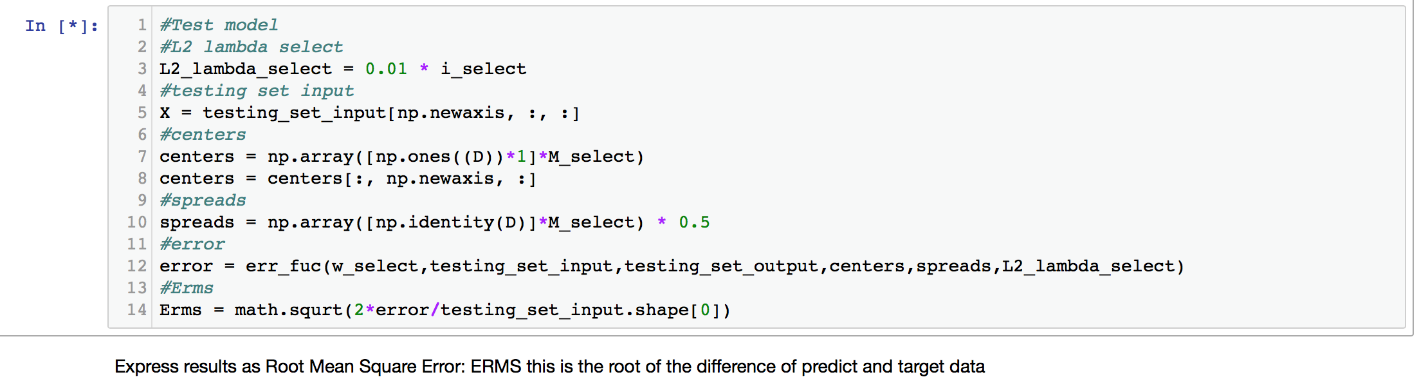
# Then we compute the error with input and output



# and compute tune hyper parameters



and also the ERMS



# Validation

After we decide the model, then we could could choose parameters. Then we could get the best performance and results.

We should choose:

*mu, s, M*

# The Intuitive Choice of Parameters

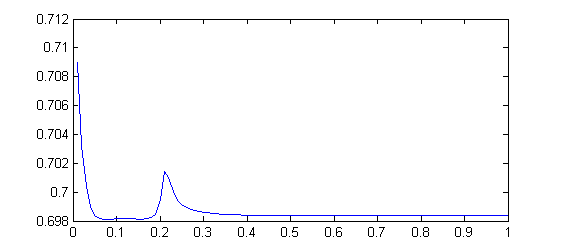
***mu*:** govern the locations of the basis functions. Ideas to choose mu:

* 1. Uniformly choose the value between 0 and 1. For example mu = [0, 0.5, 1] when M = 3.
  2. Choose the average values of some column of the feature vectors.

***s*:** governs the spatial scale. Ideas to choose s:

1. Choose a fix value.
2. For each column choose the variance of the data of this column. So *s* is a D dimension vector. Each feature vector use different s. For this case, D = 46.
3. Calculate from Hyperparameter

Choose the average variance of all feature



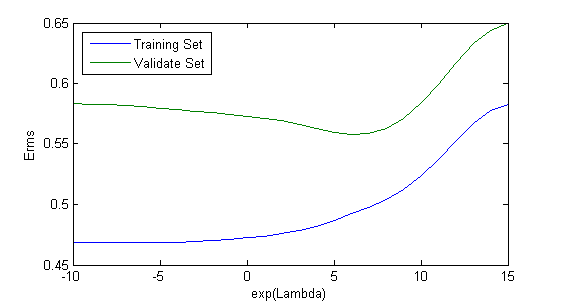
this figure shows that when s changes, how the ERMS change. If we want a highest ERMS, the s is around 0.08. So choose the variance around it.

M:

This is the Model complexity. If we don’t consider over-fitting, the bigger M the better result.

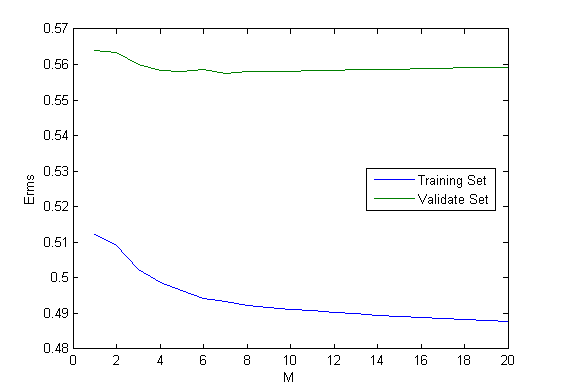
# Over-fitting Avoid:

We need to avoid the over-fitting issue. So we need an appropriate M and lambda which make the best performance of validate data.



this figure shows that how ERMS goes with the lambda. So for the best performance, we could choose exp(6).

Then



for a good M value is 7.

So after running the whole model

### Model Performance

The final *ERMS* of the 3 data set is:

|  |  |  |
| --- | --- | --- |
| Training data | Validation data | Test data |
| 0.4942 | 0.5568 | 0.5196 |