Project 1 Report

Model Choose:

The task is to use a linear model to predict the rank value based on the feature vector. There are 3 models to choose and they only differ on the basis functions. The intuitive way is to compare all 3 models and choose the one get best performance.

Following is the graph shows the performance of all 3 models. Note that *lambda* = 0 and in order to eliminate discrimination, the *mu* in both Gaussian and Sigmoid is chose uniform value between [0, 1] (by obtaining the data, all the value of the feature vector is between 0 and 1). And *s* is the average variance in both Gaussian and Sigmoid.

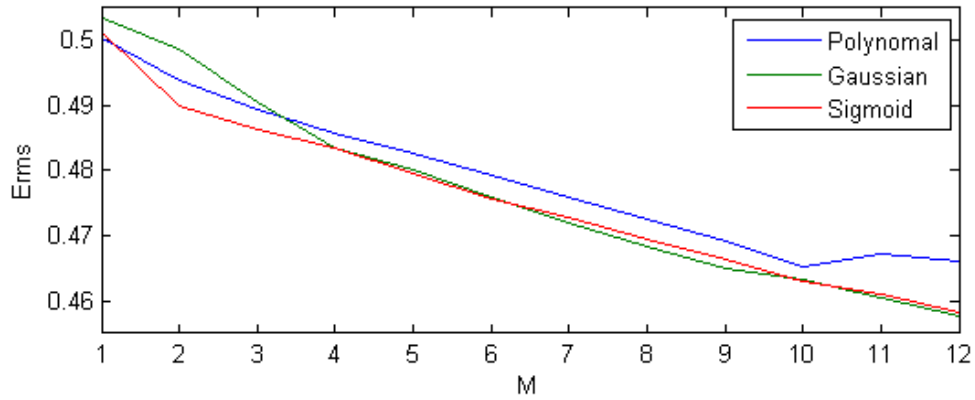

Fig.1

Figure 1 show that Gaussian and sigmoid get similar performance. So I just compute both Gaussian and sigmoid every time and choose the better one.

Validating Process:

After choosing the model, we should find the value of the parameters in the model in order to obtain best performance. The parameters waiting for choosing are:

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 sigmoid value of the uniformly value which makes the mu get higher density around 1 and 0. mu = 1./(1+exp(-(-0.5:1/M:0.5)/0.1)).
3) Choose the average values of some column of the feature vectors.

After tried all these method, it seems that the uniformly sigmoid choices is the best one.

***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) Choose the average variance of all feature values.

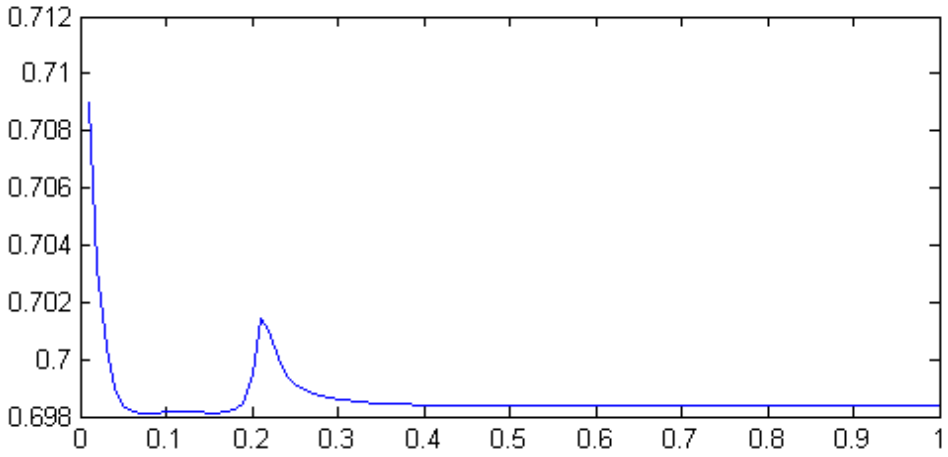

Fig. 2

Figure 2 shows how the E_{RMS} change when s varies. We can see from the table that around 0.08 the model performs best. And it is the value very close to the average variance of the data. So I just choose the average variance for convenience.

***M*:** Model complexity. We can see from Fig. 1 that without concerning about over-fitting, the bigger the M the better the results.

Over-Fitting Avoid

Though the performance seems increases when M increases but when it comes to validate data the over-fitting problem will get more and more severe. In order to avoid over-fitting, we should choose appropriate M and lambda which make the best performance of validate data.

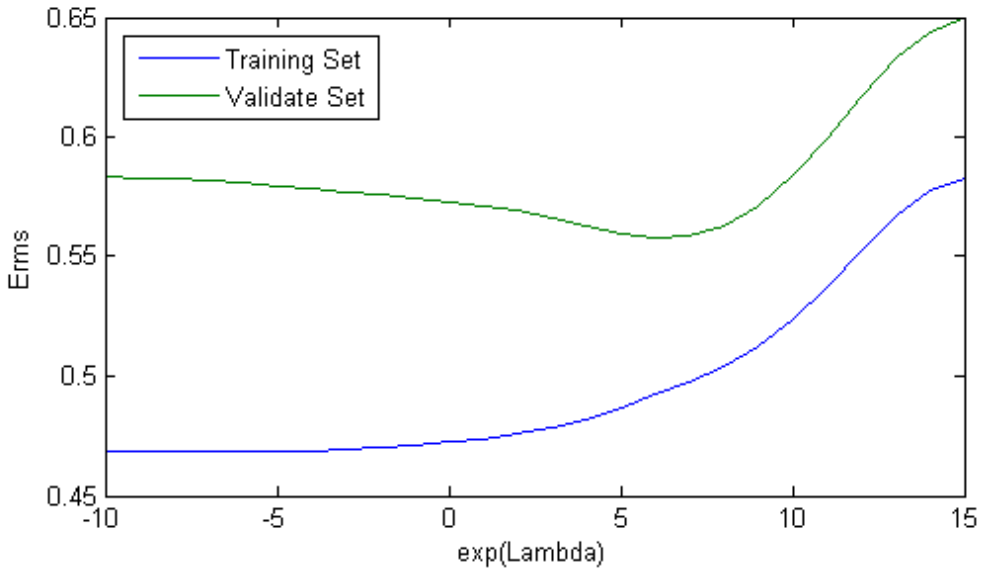

Fig. 3

Figure 3 shows how the E_{RMS} changes when lambda varies only. We can choose *Lambda* = *exp*(6) from the figure.

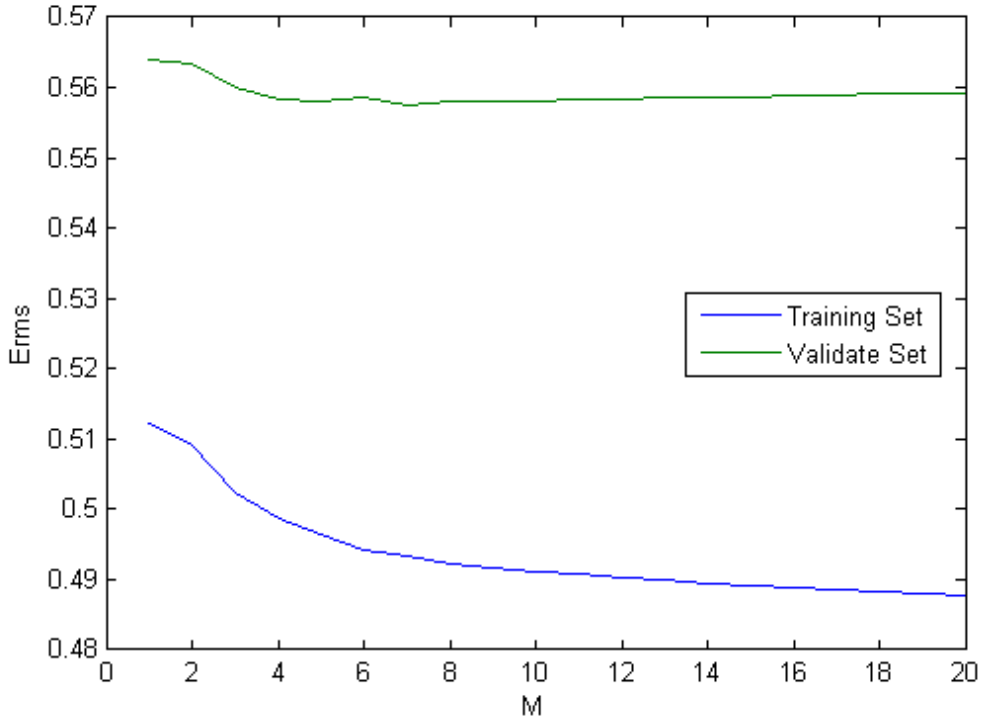

Fig. 4

Figure 4 shows how the E_{RMS} changes when M varies only. Choose M = 7 according to the figure.

Final Result:

Model Performance

The final E_{RMS} of the 3 data set is:

Training data	Validation data	Test data
0.4942	0.5568	0.5196

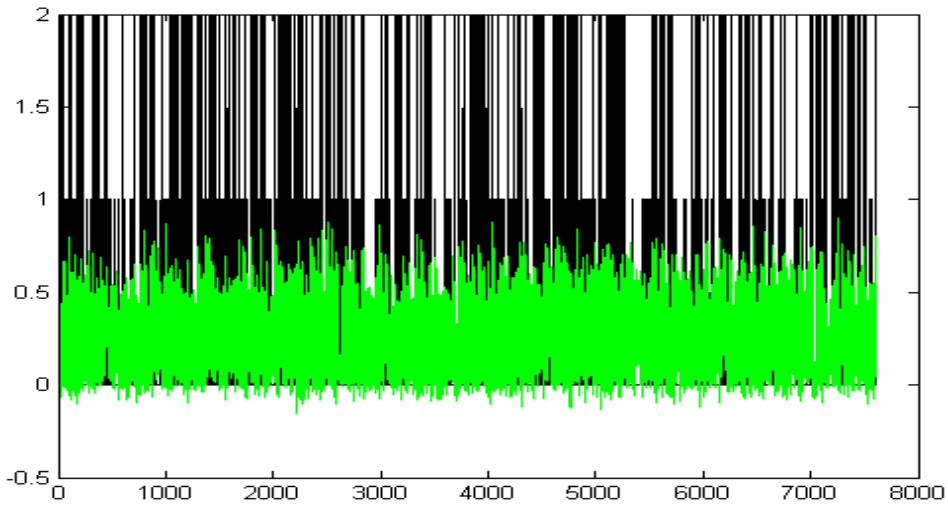

Fig. 5

Figure 5 is the comparison of the target values and predicted values.

Usage:

Copy the following code to the command window and run it.

```
% initialize
clear
clc
load('project1_data.mat');

% data pre-process.
% column 7,8,9,10,11,44 all zeros, eliminate.
data = [data(:,1:6),data(:,12:43),data(:,45:end)];
trainN = floor(size(data,1)*0.4);
valiN = floor(size(data,1)*0.1);
trainDataX = data(1:trainN,2:end);
trainDataT = data(1:trainN,1);
valiDataX = data(trainN+1:trainN+valiN,2:end);
valiDataT = data(trainN+1:trainN+valiN,1);
testDataX = data(trainN+valiN+1:end,2:end);
testDataT = data(trainN+valiN+1:end,1);

[w,M,choise,lambda] = train(trainDataX,trainDataT);
[Y,ErmsTest] = predict(w,M,choise,testDataX,testDataT);
nn_model;

fprintf('the model complexity M for the linear regression model is %d\n',
M);
fprintf('the regularization parameters lambda for the linear regression
model is %f\n', lambda);
fprintf('the root mean square error for the linear regression model
is %f\n', ErmsTest);
fprintf('the root mean square error for the neural network model is %f\n',
ErmsTestNN);
```

Conclusion:

The model is not very sensitive for some of the parameter. The performance varies too slightly. And the choice of some parameter seems did not affect another. For example no matter which M you have chosen, the minimal of lambda did not change notable. So we can have some parameter fixed and one of it vary to choose the best.

Note that every time run the neural network it will get different results but the linear model is fixed without randomness.