CSE 574: Project 2

Project Report

Submission Date: Nov 2 2015

Suramrit Singh/5016 9918/suramrit@buffalo.edu

1. Overview

Given a real world data set, and a synthetic data set(generated with gaussian), the project required implementation of linear regression to solve regression problem and further evaluate the performance of the method used.
The project required the following four tasks:

1. Train a linear regression model on real dataset using batch method.
2. Train a linear regression model on real dataset using stochastic gradient descent.
3. Train a linear regression model on synthetic dataset using batch method and evaluate its performance.
4. Train a linear regression model on synthetic dataset using stochastic gradient descent and evaluate its performance.

1.1 Background

Terms and Concepts:

Linear Regression Model:

Our linear regression function y(x, w) has the form

$$\boldsymbol{y}(\mathbf{x}, \mathbf{w}) = \boldsymbol{w^T}\, \emptyset(\boldsymbol{x}) \qquad (1)$$

where *w = (w0, w1, .., wM−1)* is a weight vector to be learnt from training samples and *φ = (φ0, ..,φM−1)* is a vector of M basis functions.
Each basis function *φj (x)* converts the input vector x into a scalar value.
In this project, the Gaussian radial basis functions was used as the basis function

$$\emptyset(\boldsymbol{x}) = \mathbf{exp}(-\frac{\mathbf{1}}{\mathbf{2}} * (\boldsymbol{x} - \mu\mathbf{j})^{\boldsymbol{T}}(\boldsymbol{\Sigma j^{-1}})(\boldsymbol{x} - \mu\mathbf{j})) \qquad (2)$$

where μj is the center of the basis function and Σj decides how broadly the basis function spreads.

Maximum Likelihood Solution:

Maximizing the likelihood is equivalent to minimizing the sum-of-squares error:

$$\boldsymbol{E(w) = 1/2 \sum_{n=1}^{N} \{t - W^T \emptyset(x_n)\}^2} \quad (3)$$

Regularization to contain over-fitting:

We append a quadratic regularization term in the error function

$$\boldsymbol{E(w) = E_d(w) + \lambda E_w(w)} \quad (4)$$

to avoid over-fitting, where

$$\boldsymbol{E_d(w) = 1/2 \sum_{n=1}^{M-1} |w_j|^2} \quad (5)$$

in which the coefficient λ governs the relative importance of the regularization term.

Solution:

After choosing the basis functions, we solve linear regression and evaluate the optimal $W_{ml}$by setting the likelihood's gradient w.r.t w equal to zero. There are two ways to solve linear regression, closed-form solution and stochastic gradient descent

i] Closed-form Maximum Likelihood Solution for w

Solution to maximum likelihood solution with quadratic regularization has the form

$$\boldsymbol{w_{ML} = \left(\lambda I + \phi^T \phi\right)^{-1} \phi^T t} \quad (6)$$

where **t = {t1, .., tN }** is the outputs in the training data and Φ is the design matrix:

$$\begin{bmatrix} \boldsymbol{\phi_0(x1)} & \cdots & \boldsymbol{\phi_{M-1}(x1)} \\ \vdots & \ddots & \vdots \\ \boldsymbol{\phi_0(xN)} & \cdots & \boldsymbol{\phi_{0M-1}(xN)} \end{bmatrix} \quad (7)$$

ii] Stochastic Gradient Descent Solution for w

The stochastic gradient descent algorithm first takes a random initial value w(0). Then it updates the value of w using

$$\boldsymbol{w^{(\tau+1)} = w^{(\tau)} + \Delta w^{(\tau)}} \quad (8)$$

Where $\Delta \mathbf{w}(\tau) = -\boldsymbol{\eta}(\tau)\nabla \mathbf{E}$ **(9)** is called the weight updates. It goes along the opposite direction of the gradient of the error. $\boldsymbol{\eta}(\tau)$ is the learning rate

Evaluation

The solution is then evaluated on a separate validation dataset using Root Mean Square (RMS) error, defined as

$$E_{RMS} = \left(\frac{2E(w^*)}{N_v}\right)^{1/2} \quad (8)$$

where w∗ is the solution and NV is the size of the validation dataset

2. Tools Used

Matlab and related libraries were used for the computation and analysis of the data. Wolfram alpha was also intermittently used for computations, plotting and analysis.

3. Tasks

3.1 Real World Dataset

3.1.1 Extracting feature values and labels from the data:

The original real world data set was converted into an excel sheet and then parsed to convert it to a MATLAB matrix. The matrix contains the feature vector s and a MATLAB vector that contains the labels corresponding to the data points of the set.

3.1.2 Data Partition

The data was then further partitioned into 3 separate disjoint data sets *Training, Validation and Test set*. The sizes of the matrices are tabulated below

Training Set	55698x46
Validation Set	6963x46
Test Set	6962x46

Tab 1

3.1.3 Setting of Model Parameters and tuning Hyper-parameters

Once the training set is obtained, we set the values of the parameters **M, μj, Σj , λ**, in order to train the model to obtain the parameter '**w'** on the training set.
The parameters were obtained and tuned so as to give better performance on the validation set.

Several Different combinations of the hyper-parameters were used so as to obtain the values which would give minimum RMS Error on the validation set

3.1.3 Result

The hyper-parameters , E-rms and the weight values found after implementation are tabulated below.

M	6
μj	[156; 2351; 3431; 15644; 30009; 52344]
λ	1

Tab 2

w0	0.307469312862584
w1	1.51765860022150e-21
w2	-5.15238819465962e-21
w3	2.08558498949146e-21
w4	-1.80544822478133e-20
w5	3.20147537710040e-19

Tab 3

Validation RMS	0.553431074
Training RMS	0.564312049

Tab 4

3.2 Synthetic Dataset

3.2.1 Extracting feature values and labels from the data

The synthetic dataset contains a matrix that contains the feature vectors and a vector that contains the labels. This is the public data generated by classified stochastic mechanism

3.2.2 Data Partition

The data was then further partitioned into 3 separate disjoint data sets *Training, Validation and Test set*. The sizes of the matrices are tabulated below

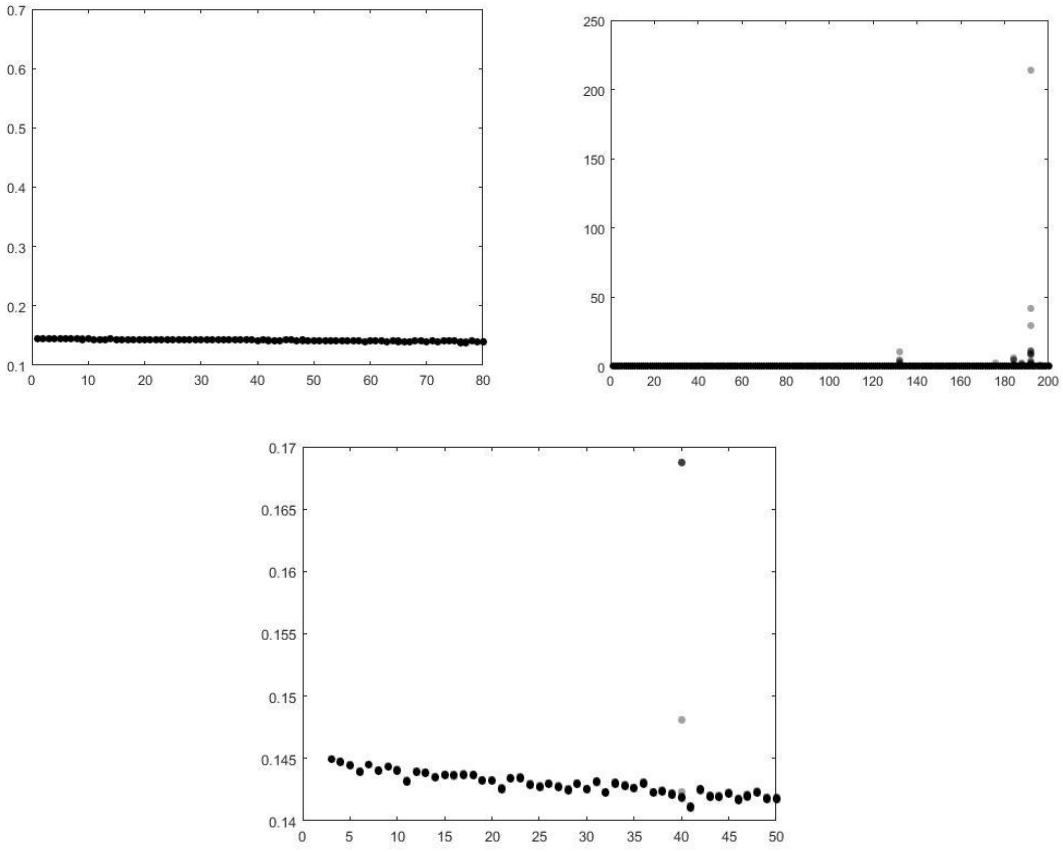
Training Set	1600x10
Validation Set	400x10

Tab 5

3.2.3 Setting of Model Parameters and tuning Hyper-parameters

While in 3.1 I started by taking a manual value of μj, M, and λ and then further changing them to obtain a valid value**.** For synthetic data I used *grid search* **[1]** to analyse the performance of the model for various different combinations of M and λ.

The plot for such combinations were then analysed to find the range of M which would give the minimum difference between E-rms for validation and training set **[2]**



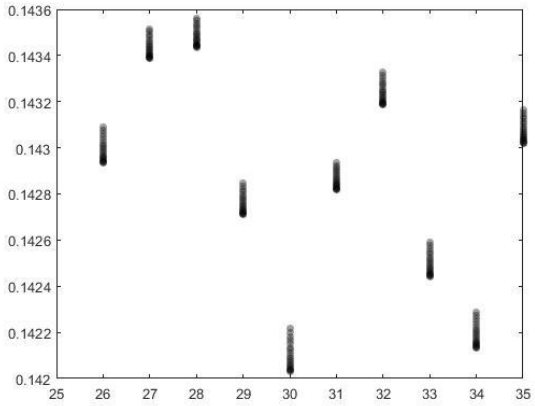
The plot for M vs E-rms(training) and the corresponding minimum E-rms(validation) values are depicted above with the y-axis containing the E-rms(training) values and x-axis containing the M values.

The corresponding E-rms(validation) for eachrange were as follows:

M Range	Erms value(validation)
0-40	.1536
0-20	.15511*
0-30	.15391
0-50	.15405
0-200	.15494

Tab 6(*possible deviation due to noise)

Now the system was tuned to change the range of M from 25-35with the value of lambda for which the E-rms was minimum



Using the modified range, now the weights were calculated.

w0	0.635081892019371
w1	-0.000328595598760719
w2	-0.000175403666864663
w3	3.91878825133068e-05
w4	4.04575887832062e-06
w5	4.47639693026679e-06
w6	8.10033875351255e-05
w7	-6.15626149731732e-05
w8	-4.63803731954695e-05
w9	-5.13755288958996e-06
w10	-7.19820311594204e-05
w11	1.15191344613161e-05
w12	9.88802256452939e-05
w13	-3.78612135885287e-06
w14	-1.07302187031705e-05
w16	-0.000155165769986356

w17	1.31626307675151e-06
w18	-9.43384443208004e-05
w19	1.30370045211377e-05
w20	-3.34096891548182e-06
w21	-2.14124203427712e-05
w22	3.88363370911566e-06
w23	2.35432777056557e-05
w24	5.50777984206264e-06
w25	7.38272021644758e-07
w26	7.25834878722068e-06
w27	-1.00724614845580e-05
w28	4.02590353963395e-05

Tab 7

3.3 Formalization of method

In section 3.2 we used a grid method in order to determine the value of hyper parameters that would give us the minimum value of Erms(validation) and Erms(training), where in section 3.1 we manually tuned and determined these parameters by looking through the results obtained.
The method of section 3.2 is a formal approach to the linear regression problem.

3.4 Matlab Implementation

3.4.1 Closed form solution

In our real world data there were 46 features for over 62000 data points. After the label values and feature matrix were extracted we take M random data points as **μ**. We then calculate the variance of each column of feature from the data and store it in Sigma. Thus the dimension of Sigma will be 46x46 with the diagonal elements containing the variance values.

Once the parameters were set, then the design matrix populated using:

```
transpose(training_set_matrix(i,1:46))-mu1(:,j);
and exp((-0.5)*(transpose(diff))*(Sigma1_inv)*diff);
```

which is the matlab implementation of **(2)**

The weights were then calculated by :

```
(inv(1*eye(M1,M1)+(transpose(design_mat)*design_mat)))*(transpose(design_mat)*transpose(training_set_label));
```

Where 1 is the value of lambda, M1 is no of basis functions. This corresponds to **(6)**

Erms was calculated using:

```
for i = 1:55698
E_dw = E_dw + ((training_set_label(i)-(transpose(w_ml)*transpose(design_mat(i,:))))^2);
end

E_rms = sqrt(E_dw/55698);
```

This corresponds to **(8)**

These implementations were used both for training and validation sets of the data in both Real World Dataset and the Synthetic Dataset.

3.4.2 Issues faced with Stochastic Gradient Descent Solution for *w*

While we got the weights from the closed form solution for both real world and synthetic data, I could not implement it for finding the solution through SGD. There were several issued faced while developing the sgd solution, some of them are listed below:

- Unable to get real valued weights: Using matlab implementation of (9), the weights were found to be exponentially growing with each updation of the weights.

- Updation of learning rate: When using a constant learning rate, the weights still remained exponentially growing with the weight values near to infinity within several update iterations.

4. Conclusion and Further Work.

Using matlab we were able to get a closed form solution for the regression problem and were able to learn and modify the prameters based on observations of the E-rms values obtained.

We were able to analyse different values of hyper-parameters and successively find corresponding values of the weight labels. While the closed form solution gave a valid result, finding the solution through stochastic gradient descent could not be implemented as the weight values did not turn out to be correct.

Thus I was only able to partially complete the requirements of the project and the model needs to be developed so that the solution can be found through stochastic gradient descent as well.

5.References

1] https://en.wikipedia.org/wiki/Hyperparameter_optimization
2] Christopher M. Bishop, Pattern Recognition and Machine Learning, Page 8-10.
3] https://en.wikipedia.org/wiki/Hyperparameter_optimization#Grid_search
4] Random Search for Hyper-Parameter Optimization ,James Bergstra Yoshua Bengio *et al. Journal of Machine Learning Research 13 (2012) 281-305*