# EE 239AS
# Special Topics in Signals and Systems
# Project 1
# Regression Analysis
# Winter 2016

Liqiang YU, Kaiming WANG and Jun FENG
904592975, 504592374, 304588434

01-31-2016

## Contents

# 1 Introduction

In this report, we implemented some regression models to fit the data within two datasets, including the linear regression, random forest regression, neural network regression and polynomial regression. We made some comparisons between the prediction results of different models based on root mean squared error(RMSE) and the coefficient of determination($R^2$). Based on the statistics provided by models, we analyzed the significance of different features. In order to avoid overfitting, which often happens in the regression fitting problem, we implemented both cross-validation and regularization techniques. Moreover, since some of the features have no numerical meaning, we formatted them into binary features, which increase the number of features.

The report is organized as follows : In section 2, we preprocessed the data and plotted the data to find some pattern. In section 3, The dataset was from a simulated traffic data on the backup system in a network system. We implemented both the linear regression model and random forest regression model to predict the copy size of the backup file and made some analysis and comparisons of the predictions. In section 4, we implemented the neural network model and polynomial regression model to fit the data and analyzed the impact of different parameters to the results. In section 5, we implement the piecewise linear regression and polynomial regressions. In section 6, we implement all the previous regression on the Boston Housing Dataset. In section 7, we introduce ridge and lasso regression techniques on this Dataset

# 2 Data Preprocessing

## 2.1 Feature Binarization

After checking the data, there are seven features in the dataset named network_backup_ dataset, including week, day of the week, backup start time, work flow ID, file name, size of backup and backup time. The target value is the size of backup, others are features. Therefore, first of all we need to transform non-numerical values into numerical values. However, consider "the day of the week" feature, "Tuesday" is not larger than "Monday", but "2" is larger than "1". So we need to make such kind of features into binary format. For instance, for "day of the week", we need to expand it into seven features, where it is 1 on the specified day and all the others 0. Finally, we have 64 features in total.

## 2.2 Pattern Detection

In order the find the pattern, we plot the figure of the copy size over time. The independent variable is the "Start Time - Hour of Day", the dependent variable is the copy size. There are six sampling times in a day and 20 days as a period. So the x axis is from 1 to 120. The figure is shown in figure 1. From 1 we can see there is clearly a cycle between copy size and time. So we plot the copy size

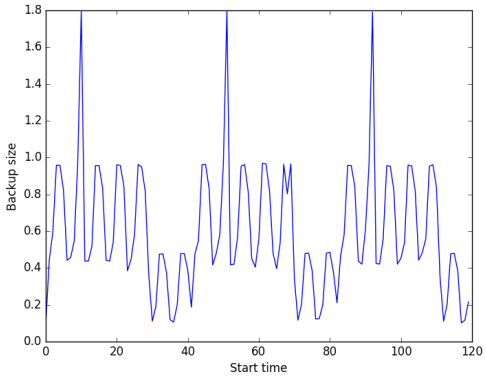


Figure 1: The copy size of backup over start time

over days to find the cycle period. The result is shown in figure 2. Hence we can predict that the cycle period is approximately a week.

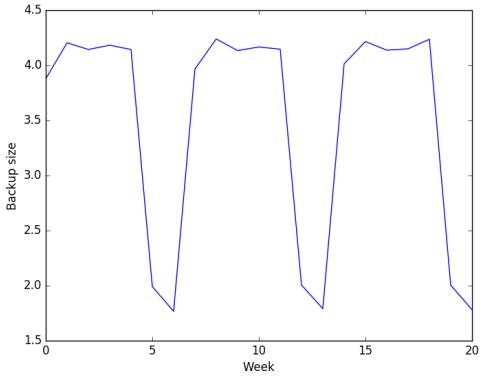


Figure 2: The copy size of backup over week

# 3 Linear and Random Forest Regression

## 3.1 Linear Regression

In the Linear Regression Model, we choose the copy size as the target variable and the other attributes as the features, the fit model is as follow

$$\bar{Y} = X\alpha$$

where $\bar{Y}$ represents the target, $X$ represents the features vector and $\alpha$ is the linear coefficient. We used least square as the penalty function, which is

$$min||Y - \bar{Y}||_2$$

In order to avoid the overfitting problem, we implemented 10-fold cross-validation technique. First do a random shuffling over the data, then split them into 10 folds. Each time choose 9 of 10 as the train set and the other 1 fold as the test set. The Root Square Mean Error (RMSE) is shown in figure 3. The average RMSE is 0.071. To evaluate the fitting accuracy of our model, Fitted values VS

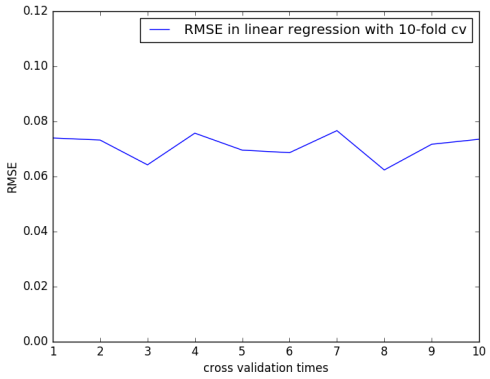


Figure 3: The copy size of backup over week

actual values and residuals vs fitted values are plotted. From figure 4, we can see that most predicted values are close to the actual values. From figure 3.1, we can see that the residuals are distributed randomly around zero axis, which means the model is proper.

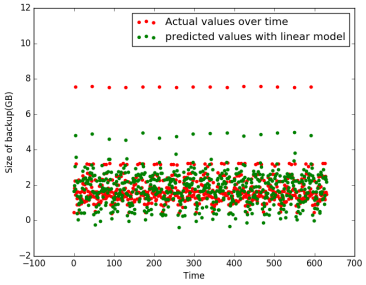


Figure 4: Actual VS predicted

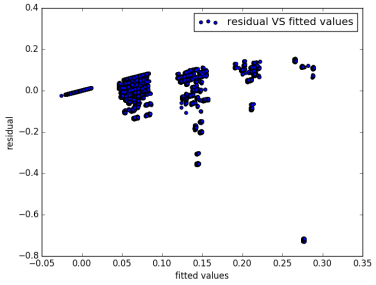


Figure 5: Residual VS fitted values

As for the significance of different variables, we choose p-value as the evaluation

criterion. The p-value of the six features is shown in figure 6.

```
[  7.78443589e-01   8.71062213e-04   2.46112227e-08   1.32748780e-01
   1.57542554e-01   0.00000000e+00]
```

Figure 6: p-value of 6 features

Therefore the most important two features are "Backup start time - Hour of the day" and "Backup Time" because their p-values are close to 0. For 64 features, the p-value matrix is shown below.

```
7.36199210e-001   9.39383338e-001   4.66439149e-001   9.08462632e-001
5.45241529e-001   4.35994229e-001   3.09252234e-001   6.38013695e-001
5.16578878e-001   2.80070834e-001   4.85768453e-001   8.55993609e-001
6.36535181e-001   7.63174401e-001   2.98120865e-001   1.18985257e-004
9.31487572e-004   1.66246279e-001   1.95819809e-001   4.44216789e-001
5.17062882e-001   1.99647310e-024   6.99156954e-009   1.54736893e-004
1.23481345e-002   3.53017379e-014   5.81362330e-002   1.24279127e-001
1.59934030e-033   8.85184242e-006   7.79480962e-034   2.34530310e-060
2.97238246e-100   2.68855252e-006   1.23242625e-005   3.15955319e-004
1.10012870e-005   5.78513377e-005   2.40988217e-007   2.36501171e-003
3.07991642e-001   4.75428019e-002   6.68776922e-002   5.09616461e-002
8.84143253e-001   6.74472700e-005   9.82830391e-006   6.18754039e-006
7.10107131e-003   9.41370915e-007   2.92999661e-010   5.36294108e-008
2.37168301e-008   1.50927326e-011   4.51686551e-009   2.17186539e-010
1.27307840e-007   6.34206490e-013   3.52571891e-010   4.98335413e-016
5.48046529e-013   1.95679737e-017   3.45472611e-014   0.00000000e+000]
```

Figure 7: p-value of 64 features

### 3.2 Random Forest Regression

For random forest regression, we initialize the model with parameters : number of trees 20, depth of trees 4, max features 64. The average RMSE is 0.0297. After tuning the parameters, we have the best RMSE 0.00943. The parameters are : number of trees 32, depth of trees 12, max features 64. The best RMSE is shown in figure 8.

### 3.3 The comparison between two models

The comparison between Linear Regression and Random Forest Regression is shown in figure 9. Random forest is a lot better than linear regression.

The prediction of the random forest model is shown in figure 10. We can see the cycle period is around 42 backup times, which is about a week, so the pattern is the same with the actual values.

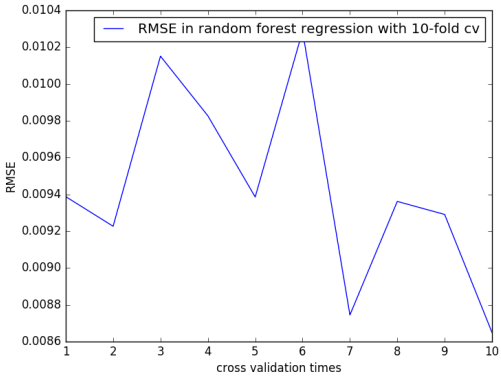


Figure 8: The best RMSE of random forest regression

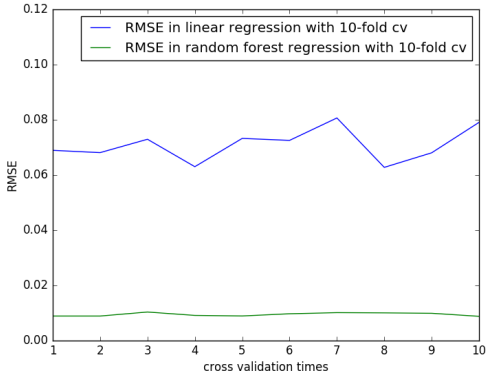


Figure 9: The comparison between Linear and Random Forest Regression

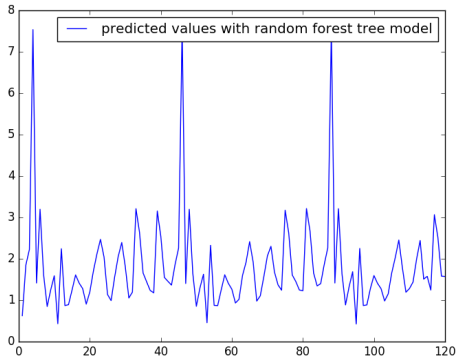


Figure 10: The prediction of the random forest

# 4 Neural Network

In the neural network model, we choose the feedforward network with the back-propagation trainer. The main parameters are the number of layers, the kind of hidden layers and the number of iteration before convergence. After lots of experience, we find that sigmoid layer is proper for hidden layers and linear layer is proper for the output layer. The optimal number of layers is 7. The RMSE reduces when the number of iteration is increasing, however it is too slow with large iteration number. So we set it to be 10. The best RMSE we can get is 0.0827. The RMSE is shown in figure 11.

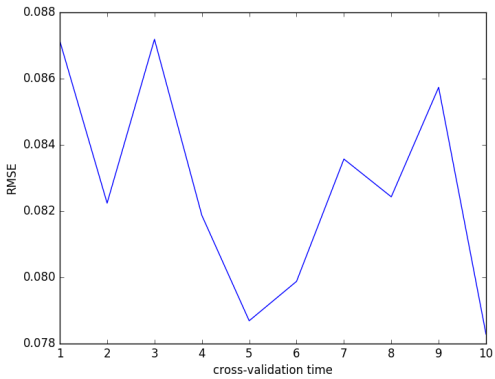


Figure 11: The RMSE of neural network model

# 5 Piece-wise Linear Regression and Polynomial Regression

## 5.1 Piece-wise Linear Regression

We just divide the all tuples according to their work-flow #, then implement the linear regression respectively which is illustrated by figure 12. However, we find that sometimes, piecewise regression does not necessarily benign the RMSE. It is consistent to the most cases: at most time, more data and more features can bring about a more reliable regression.

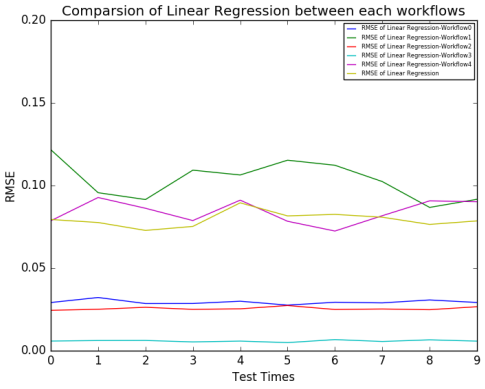


Figure 12: Piecewise Linear Regression

## 5.2 Polynomial Regression

By implying the PolynomialFeatures of Scikit-Learn, we can do polynomial regression. However, when the degree go higher and higher, overfitting may happened. Just as in figure 13. According to the figure, the threshold of the degree is 5. Cross validation can help us to decide the degree of polynomial function.

# 6 Boston Datasets

# 7 Lasso & Ridge Regularization

In machine learning, over fitting is likely to happens when a model is excessively complicated. Under this situation, the machine only memorizes the training data, instead of learning to predict the data from given features. In our problem, we have only about 500 tuple of data with 20 different features. Thus, the chance of our model to memorize data from these features is relatively high. Before,

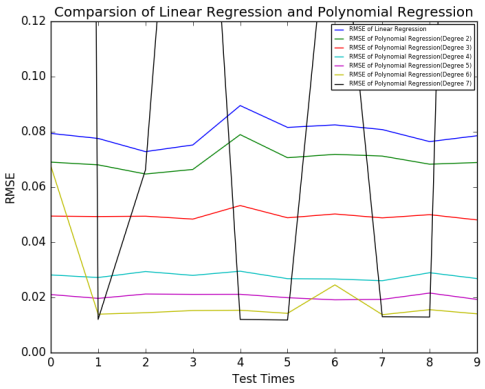


Figure 13: Piecewise Linear Regression

we use cross validation to limit the affection of over fitting during the assessing of a model. In this way, we can only determine if a given model is good or not. Instead, if we want to generate better learning models with little over fitting, we may use method as regularization. In regularization, we introduce a regularization term or so called loss function, which is usually a penalty on the complexity of our model. Therefore, it can improve the generalization ability of our model.

## 7.1 Lasso Regularization

In Lasso regularization, the regularization term is considering the -norm of the regression coefficient vector. And the penalty function has the following form:

$$min\frac{1}{2n}||Y - X\beta||_2^2 + \alpha||\beta||_1$$

For different extent of penalty on the complexity, we can choose various value of . And in our solution, we choose it in the range $0.1, 0.01, 0.001$ as required. The corresponding performance of the regression model is shown in Figure 14, from which we can see that the RMSE is smaller with a little value of $\alpha$. After further research, we know that the result of linear regression almost overlaps the result of $\alpha = 0.001$, which means that the RMSE of pure linear regression is smaller than those of Lasso regression. Thus, in this problem, the control of over fitting does not bring a better model to us. However, this does not mean it is useless. After checking the result of the regression coefficients, we are shocked by its sparsity. When $\alpha = 0.1$, only 3 coefficients are nonzero and all the other 17 are 0. Thus, in this way, we have found a great sparse solution of our problem.

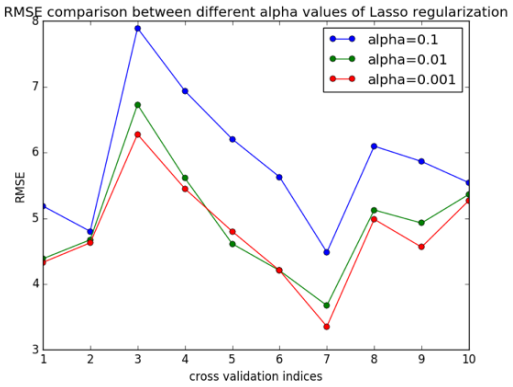


Figure 14: Lasso Regularization of Boston Dataset

## 7.2 Ridge Regularization

In Ridge regularization, the regularization term is considering the $l_2$-norm of regression coefficient vector. And the penalty function has the following form:

$$min\frac{1}{2n}||Y - X\beta||_2^2 + \alpha||\beta||_1$$

Similar as Lasso regularization, we set the value of  in the range {0.1,0.01,0.001} and the performance of our regression models are shown in Figure 15. This time, the curves of RMSE for different  are very close to each other. For a clearer

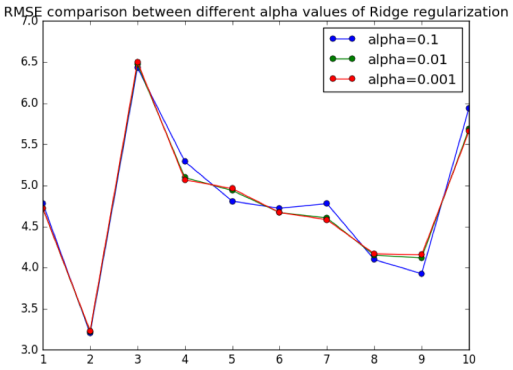


Figure 15: Ridge Regularization of Boston Dataset

comparison of the all RMSE values, we calculate the mean RMSE upon every 10-fold cross validation and we put all of them in Table 1. From the above table, we find out that for Lasso regularization, the RMSE is worsened but the

Table 1: Compare between Lasso regression and Ridge regression

| Value of $\alpha$ | Lasso regularization | Ridge regularization |
|---|---|---|
| 0.1 | 5.8639 | 4.7981 |
| 0.01 | 4.9307 | 4.7686 |
| 0.001 | 4.7861 | 4.7725 |

sparsity of regression coefficients is obtained. And for Ridge regularization, we get the best RMSE result at =0.01, which prove that with the control of over fitting, we may get a better model.