Project 1 Regression Analysis

Group members: Huang xinxin 104589081

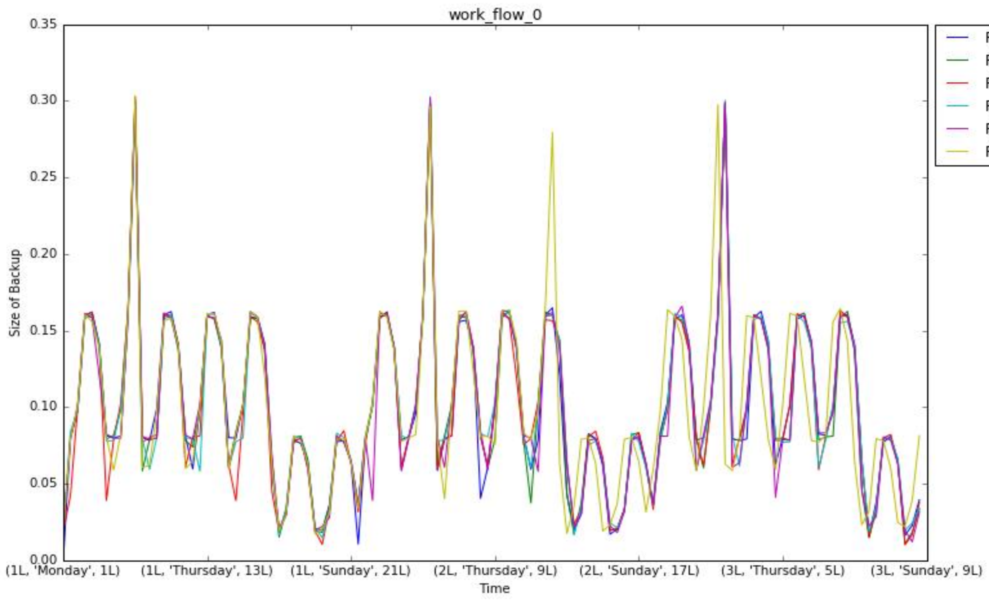
Lu qiujing 704617222

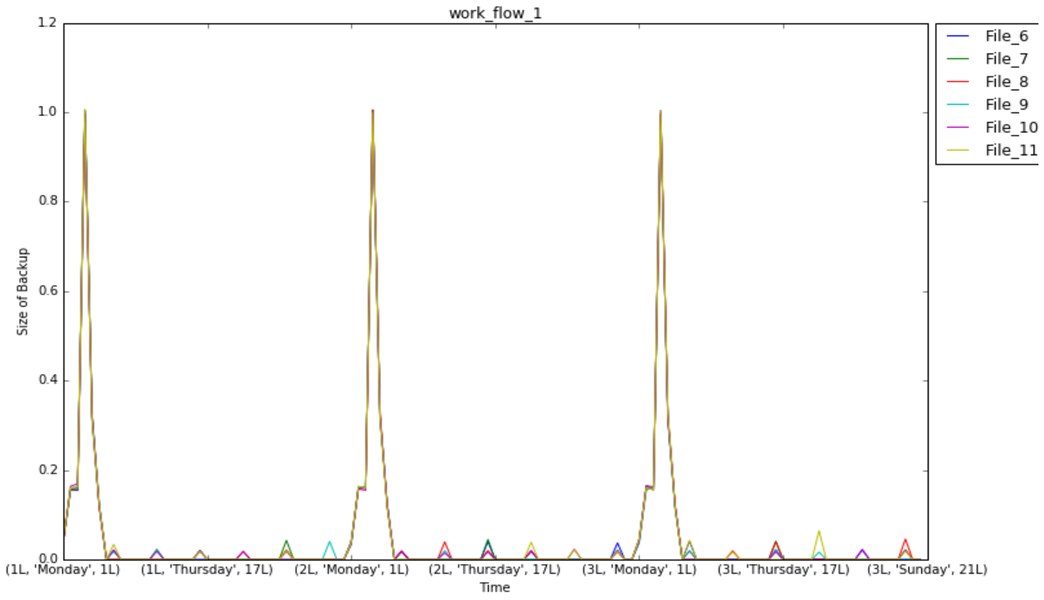
Niu longjia 304590762

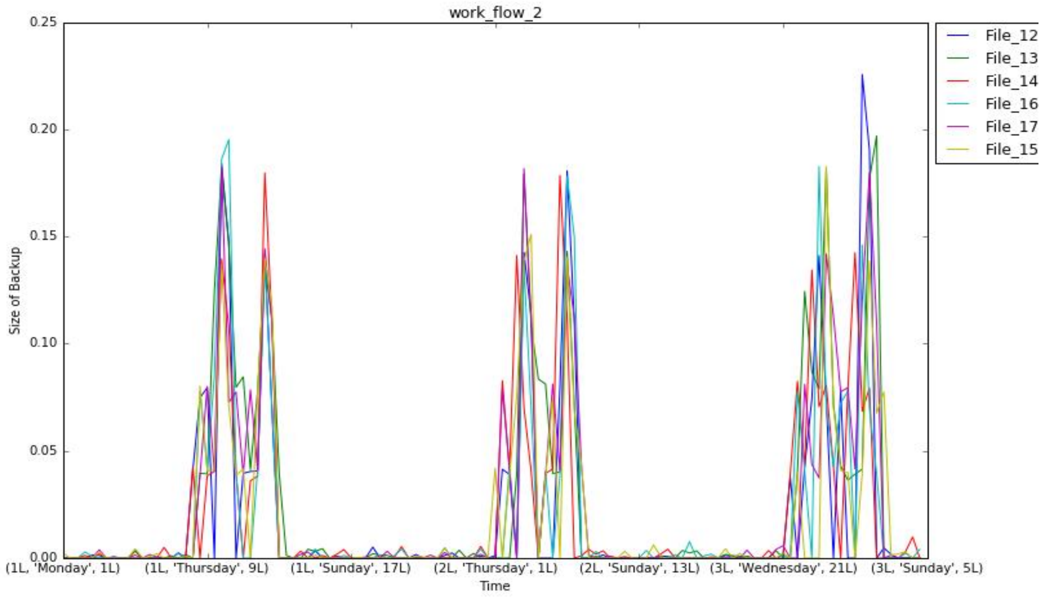
I. Network backup Dataset

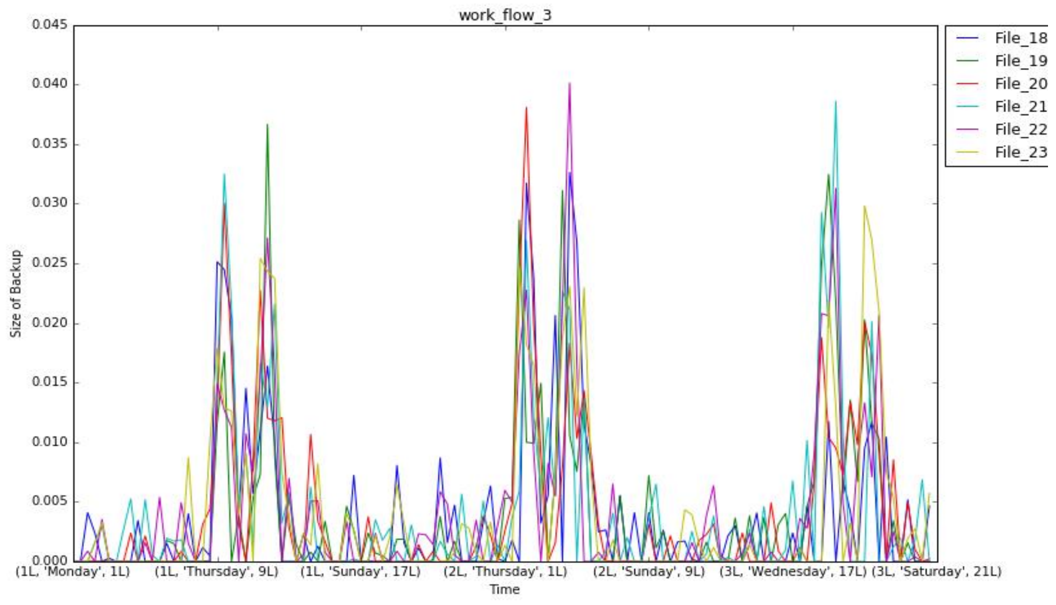
Problem1

For each workflow, we plot the backup sizes of all files on a time period of 20 days. Figures are as follows.




work_flow_1
Size of Backup
Time
File_6
File_7
File_8
File_9
File_10
File_11
(1L, 'Monday', 1L)
(1L, 'Thursday', 17L)
(2L, 'Monday', 1L)
(2L, 'Thursday', 17L)
(3L, 'Monday', 1L)
(3L, 'Thursday', 17L)
(3L, 'Sunday', 21L)


work_flow_2
Size of Backup
Time
File_12
File_13
File_14
File_16
File_17
File_15
(1L, 'Monday', 1L)
(1L, 'Thursday', 9L)
(1L, 'Sunday', 17L)
(2L, 'Thursday', 1L)
(2L, 'Sunday', 13L)
(3L, 'Wednesday', 21L)
(3L, 'Sunday', 5L)



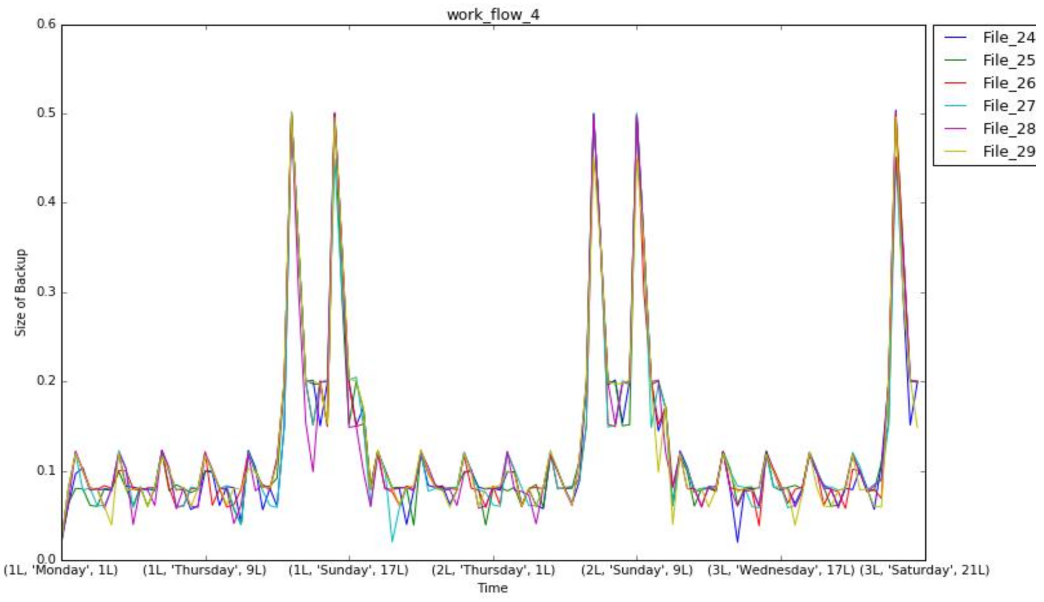


Fig 1. Figures of all files in all workflows

As we can see from the figures, backup sizes of files belong to the same workflow have similar changing patterns. Backup sizes of all the files are changing periodically to time and share the same period of 7 days.

Problem2 a)

1) Preparing the data

In the linear regression problem, we are trying to fit the input features to predict reasonable output by training our linear model. In this problem, we have 6 attributes that can be used as features, and one target variable: the backup size. To train our model we need to minimize the error between fitted results and actual values, so we need to prepare our data as numbers for calculation. For the first tries, we transform the week into 0-6 integers, filenames into 0-29 integers, Work-Flow-ID into 0-5 integers, the rest keep the same.

However, the preprocessing of data will pose a heavy influence on the model's coefficient and the whole understanding of the model. As a subjective training, we don't want to introduce any artificial information about the data, such as input order, relations between variables, importance in name. So in the next stage, we process the data in a different way which is more reasonable for fitting. For the week#, we made this column which contains 15 different numbers into 15 separate columns representing 15 different numbers. In each column, there contains only 0 and 1(0 represent not this week, 1 represents that this data belong to this week). For file names, we change them into 30 columns, for day of weeks 7 columns, for backup start time, 6 columns, for work flow 5 columns. For backup time, 5 columns. So finally, our input data set contains 68 features.

2) Using the linear regression model to fit the training data

To compare the significance of the 7 coefficients, normalizing the input data is a necessary step.

For the normalized input, the coefficients show that when the variable changes one unit, how the output changes. The larger its absolute value is, the more difference it makes in the output.

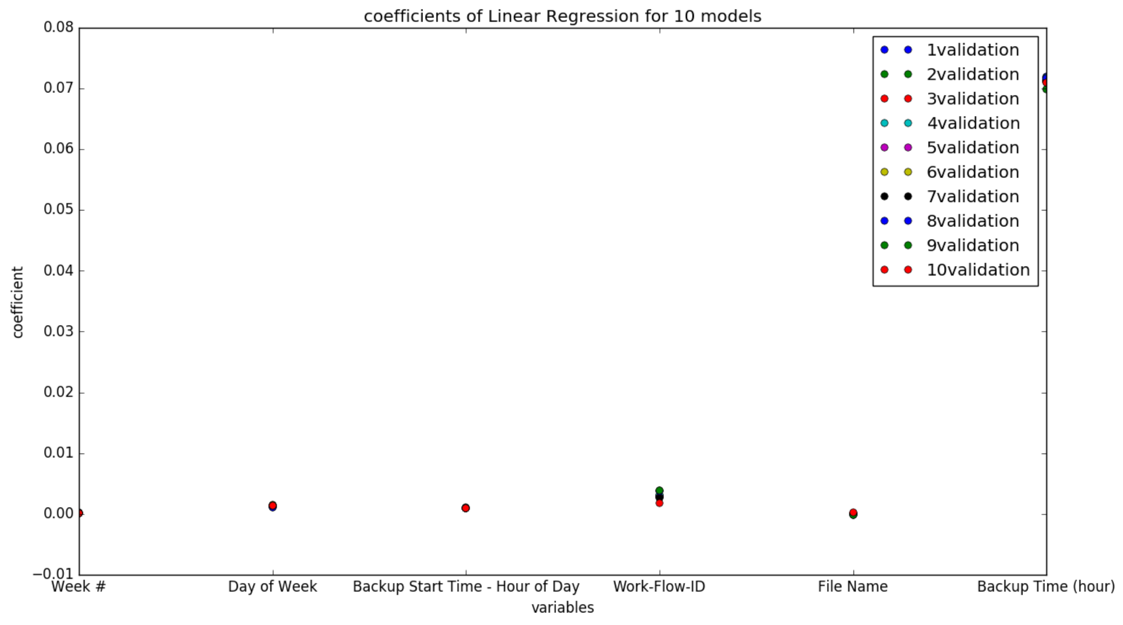


Fig 2a.1. Coefficients of linear regression in 10 folds cross validation (6 inputs)

As shown in the picture, the coefficient of ‘Backup Time’ effects the output most and in the 10 times training, coefficients stay on the similar ranges, which means that the model is relatively steady.

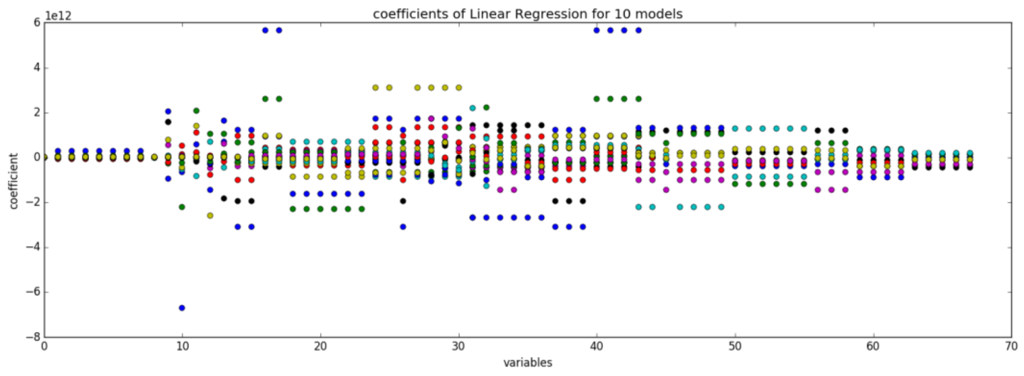


Fig 2a.2. Coefficients of linear regression in 10 folds cross validation (68 inputs)

x=['Week #_1', 'Week #_2', 'Week #_3', 'Week #_4', 'Week #_5', 'Week #_6', 'Week #_7', 'Week #_8', 'Week #_9', 'Week #_10', 'Week #_11', 'Week #_12', 'Week #_13', 'Week #_14', 'Week #_15', 'Friday', 'Monday', 'Saturday', 'Sunday', 'Thursday', 'Tuesday', 'Wednesday', 'BST_1', 'BST_5', 'BST_9', 'BST_13', 'BST_17', 'BST_21', 'work_flow_0', 'work_flow_1', 'work_flow_2', 'work_flow_3', 'work_flow_4', 'File_0', 'File_1', 'File_10', 'File_11', 'File_12', 'File_13', 'File_14', 'File_15', 'File_16', 'File_17', 'File_18', 'File_19', 'File_2', 'File_20', 'File_21', 'File_22', 'File_23', 'File_24', 'File_25', 'File_26', 'File_27', 'File_28', 'File_29', 'File_3', 'File_4', 'File_5', 'File_6', 'File_7', 'File_8', 'File_9', 'BT_h_0', 'BT_h_1', 'BT_h_2', 'BT_h_3', 'BT_h_4']

In this situation, we have 68 features as input resulting better fitting for the original data, shown in the scores: for the first try, the average score for the 10 folds cross validation is 0.42, but for this process, the average score is 0.59, increasing about 40%. However, it seems difficult to directly find the important variables: the 68 corresponding coefficients fluctuate among 10 validations. Although we can observe, ‘Friday’, ‘Monday’ and ’work flow’ are the variables that have occurred large values, it changes over different models. This result shows the unsteady nature of linear model with multiple variables, any abnormal values can greatly change the coefficients of our model.

3) Analyze the model

RMSE for the 10-folds cross validations for 6 inputs

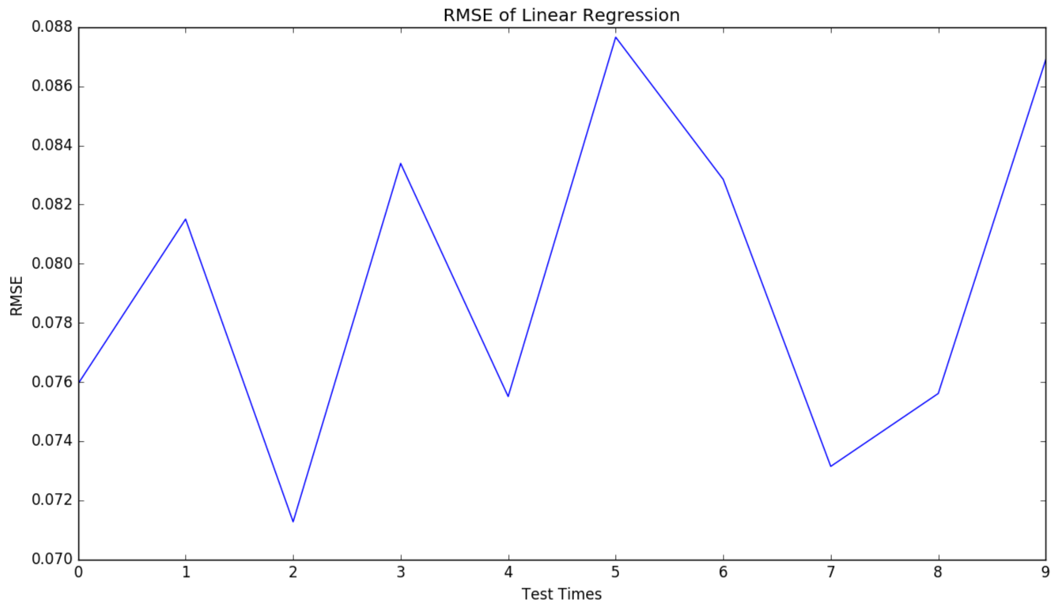


Fig 2a.3. RMSE of linear regression in 10 folds cross validation (6 inputs)

RMSE is 0.079378388728

RMSE for the 10-folds cross validations for 68 inputs

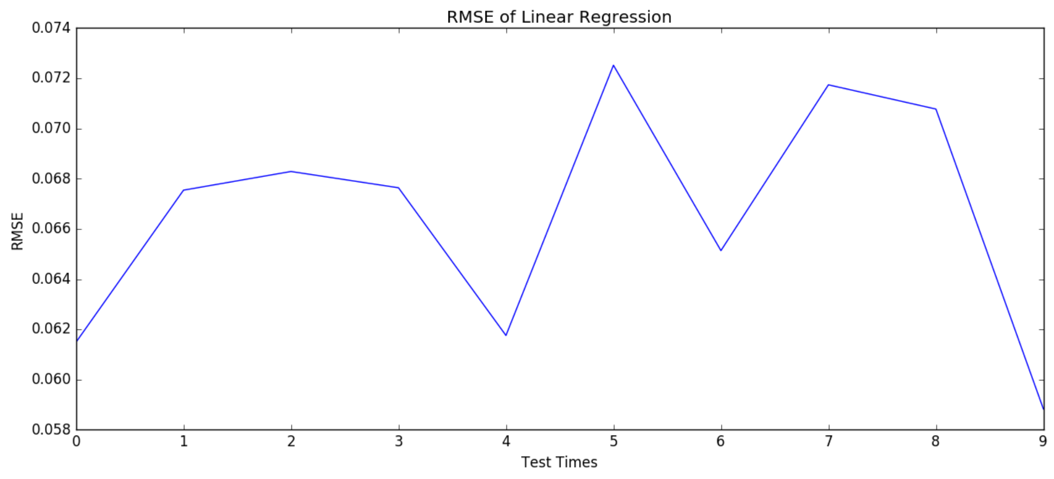


Fig 2a.4. RMSE of linear regression in 10 folds cross validation (68 inputs)

RMSE is: 0.0665702516332

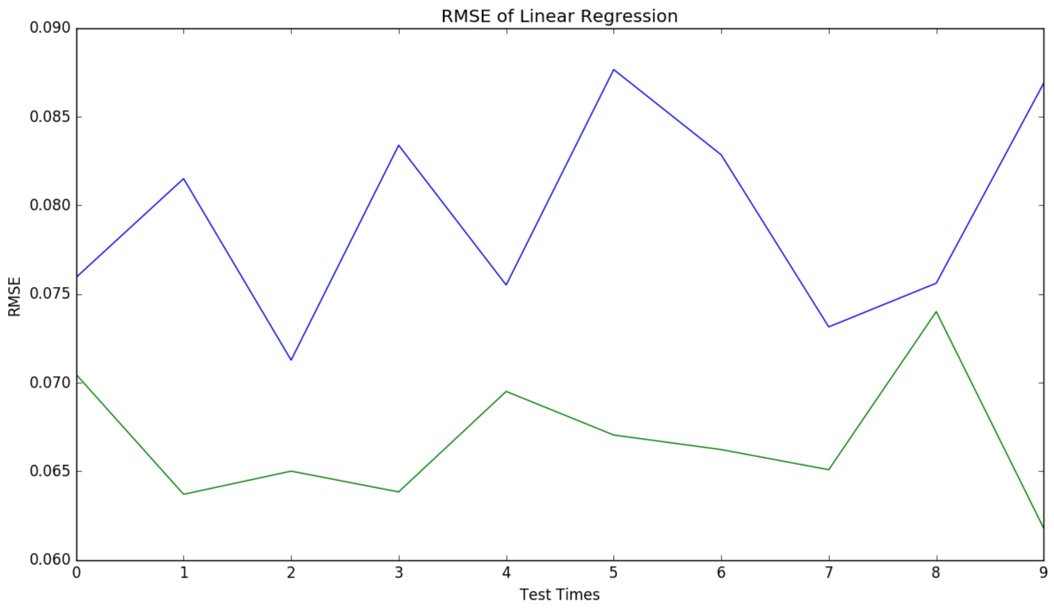
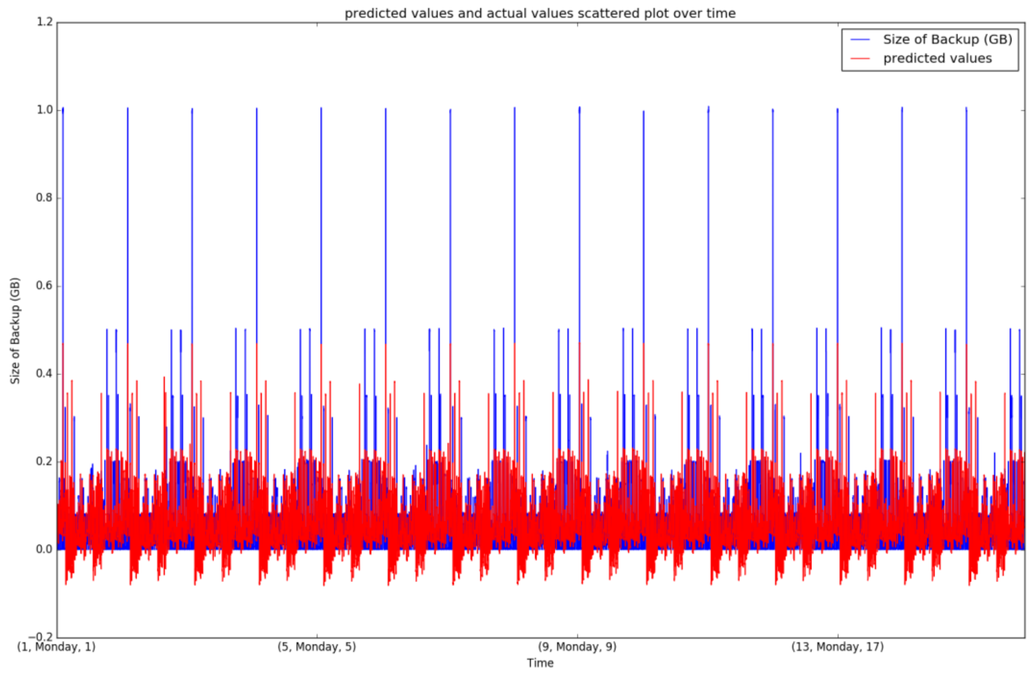
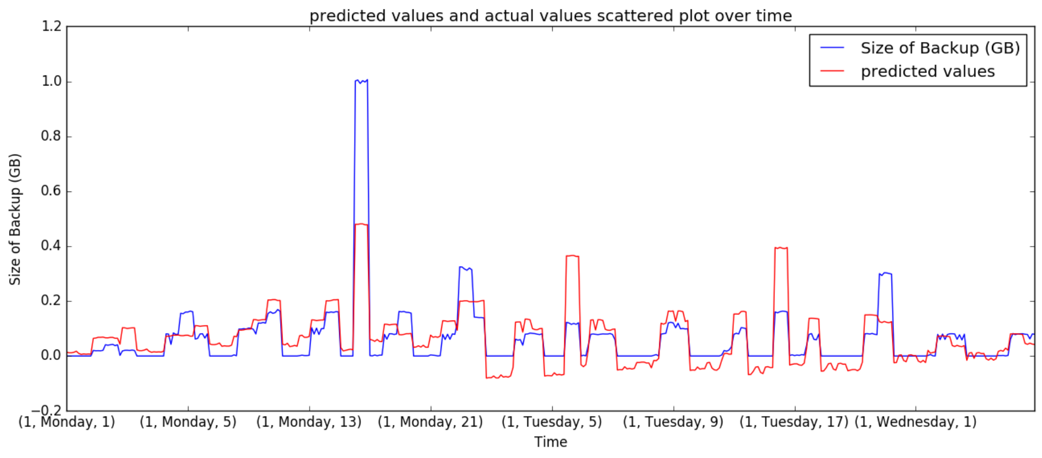


Fig 2a.5. RMSE comparison between the two models in 10 folds cross validation

Since the RMSE for the 10-folds cross validations doesn’t change greatly, any of 10 trained models can represent the model get from the whole dataset. In addition, since the second model performs better than the first one, we use it to do the following analysis. Pick the random one among them to show the” model’s fitted values and actual values scattered” over time.

“Fitted values and actual values scattered plot over time”


predicted values and actual values scattered plot over time
Size of Backup (GB)
predicted values
Size of Backup (GB)
Time
(1, Monday, 1)
(5, Monday, 5)
(9, Monday, 9)
(13, Monday, 17)
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2


predicted values and actual values scattered plot over time
Size of Backup (GB)
predicted values
Size of Backup (GB)
Time
(1, Monday, 1)
(1, Monday, 5)
(1, Monday, 13)
(1, Monday, 21)
(1, Tuesday, 5)
(1, Tuesday, 9)
(1, Tuesday, 17)
(1, Wednesday, 1)
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

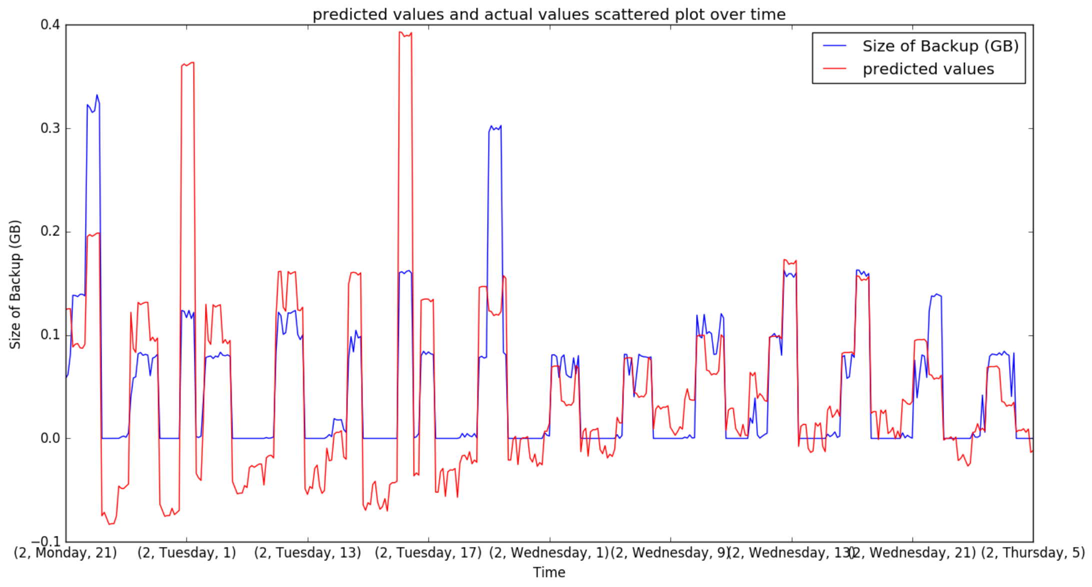


Fig 2a.6. Fitted values and actual values scattered plot over time (whole result and partial region)

As shown in the above 3 pictures, the model has a good fit for backup size between 0-0.2 and shows periodic pattern, but it hardly fits data with large values even they have a pattern. The linear model is not good enough to describe the hidden pattern.

“Residuals versus fitted values plot”.

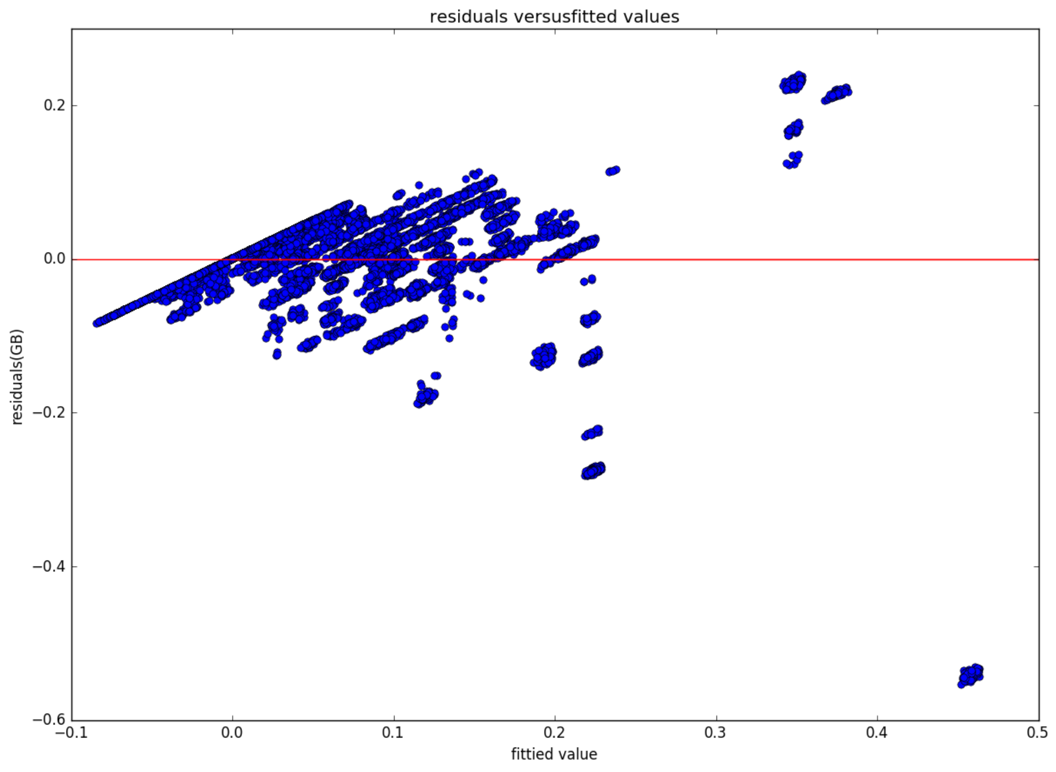


Fig 2a.7. Residual versus fitted values

Residual = Observed – Predicted

For the residuals versus fitted values plot, y goes below 0 means the fitted model underestimates the actual value and when y is above 0 the fitted value is higher than actual one. In this picture, the residual has a relatively random pattern around y=0, but there are two problems limiting it as a decent enough model.

1. As actual value increases, the absolute value of residual increases, which shows a weak fit for large backup size. It shows that linear model is not strong enough to accurately describe the relations behind the 68 features.

2. There are negative values for fitted model, but in the actual situation, there won't be any negative backup size, so this linear model does poorly for few results with small backup size.

To improve results, we may transform variables into a better form, add new relations resulted in backup size, and introduce new variables describing hidden relations between variables.

Problem2 b)

In this part, we use the random forest regression model. We also implement the same two data preprocessing schemes as in part a).

With inputs = 6 and 68, the best RMSE values we can get is 0.00957255768834 and 0.00952052265835, respectively. Detailed parameters and experiment results are as follows,

Table 2b.1 RMSE of Random Forest Regress or with different parameters

No. of Trees	Max Depth	Max No. of Features	RMSE
20	4	6	0.0296047335874
		68	0.0427490772841
30	4	6	0.0296011376063
		68	0.0430224266212
20	6	6	0.0165130026881
		68	0.0252371201141
30	6	6	0.0164943807357
		68	0.0253073474016
20	8	6	0.010868771088
		68	0.0163384527275
30	8	6	0.0108293576788
		68	0.0164058103989
20	12	6	0.00963975039345

		68	0.00955401236959
30	12	6	0.00957255768834
		68	0.00952052265835

As shown in the table, we mainly analyze the impact of the three parameters on the model performance. It shows that the influence of tree number is much smaller than the max depth, no matter how many inputs we have. As the trees getting deeper, the predict values are more precise. It also can be seen that when depth of each tree max is small, the model perform better on the first scheme, which have less attributes. But as the max depth becoming larger, the performance on RMSE of the two schemes converges to the same value.

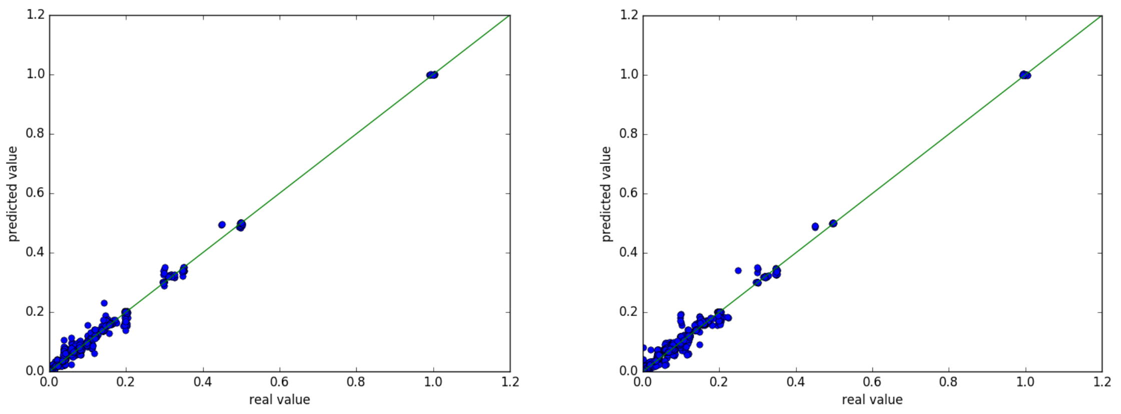


Fig 2b.1. Predicted values vs. real values at best RMSE with input = 6 and 68

In the following part, we analyze which feature provides the most information in prediction. It can be obtained by the feature importance attribute of module random forest regression. In the Figure 2b.2, Backup time and Day of week are much more important than other attributes. To some degree, our target, Size of Backup, is in direct proportion to Backup time. This agrees with our intuition. When backup time equals to 0, the backup process does not happen. The Size of Backup must be zero. And the larger the backup size is, the more time it will take to transmit. The pattern observed in question one can also be identified in this part. The second most important attribute is Day of Week. This agrees the conclusion we get in question 1. In question 1, we find that backup sizes of all the files are changing periodically to time and share the same period of 7 days. So, the records sent at same day in each week are more likely to have the same backup size.

In Figure 2b.3, more details of feature importance can be observed. In Backup size column, 0 is the most frequent value. When backup size is zero, backup time is also zero. So BT_h_0 (Backup Time = 0) is much more important than the other features. Similarly, most ‘0’ Backup size appear on Monday, which results that Monday becomes the most important feature. Size of files in workflow 0, workflow 1 and workflow 4 are changing more regularly to time than these in workflow 1 and workflow 2 which can also be observed on the figure.

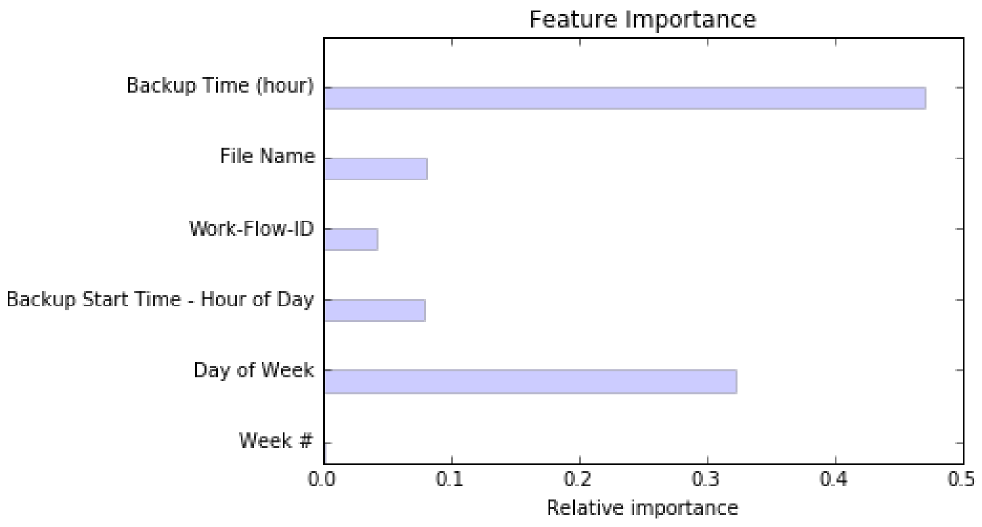


Fig 2b.2. Feature importance with 6 features

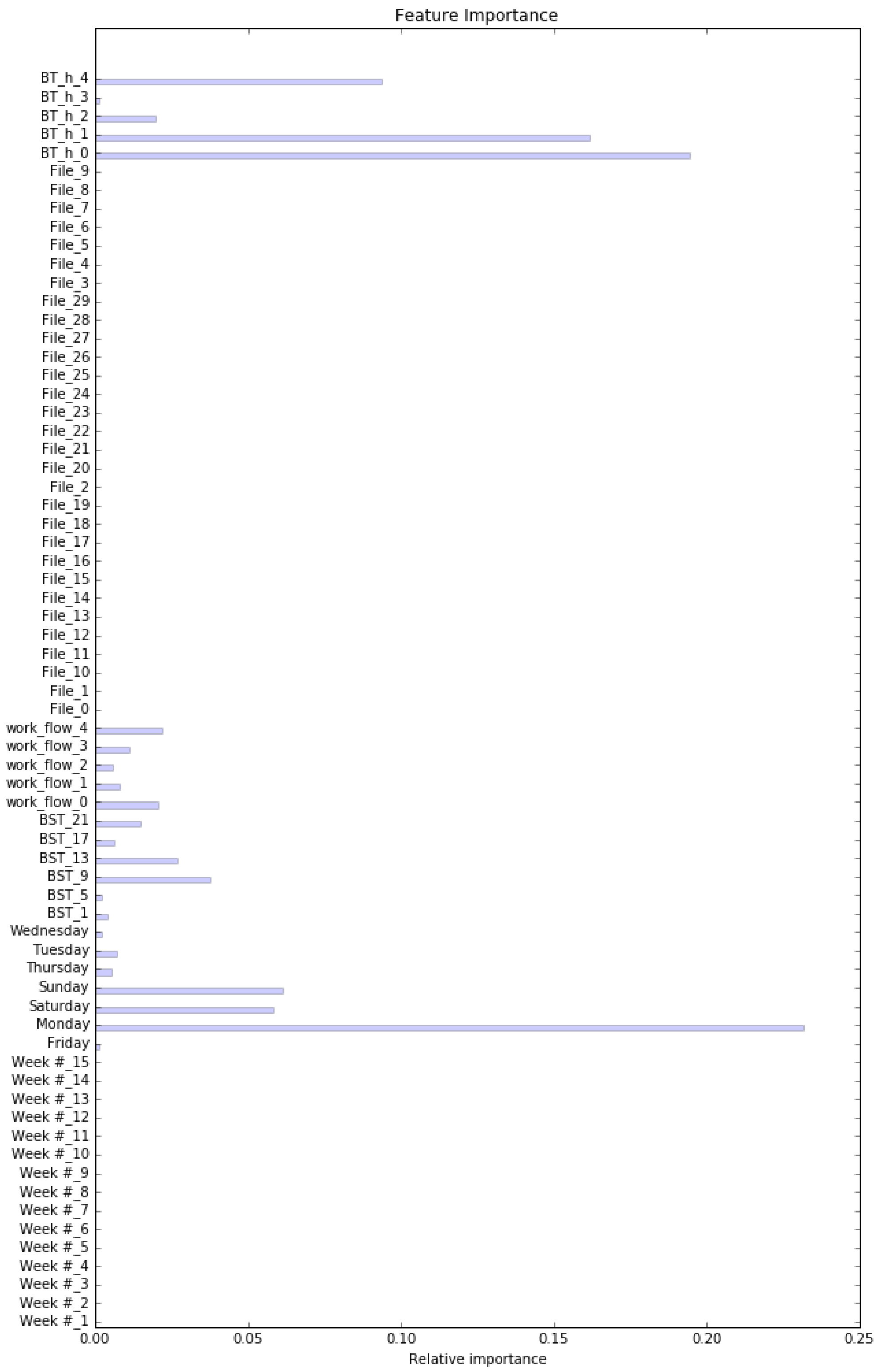


Fig 2b.3. Feature Importance with 68 features

Comparing with the linear regression, Random forest regressor has much more better performance in RMSE. And are more stable for different test sets.

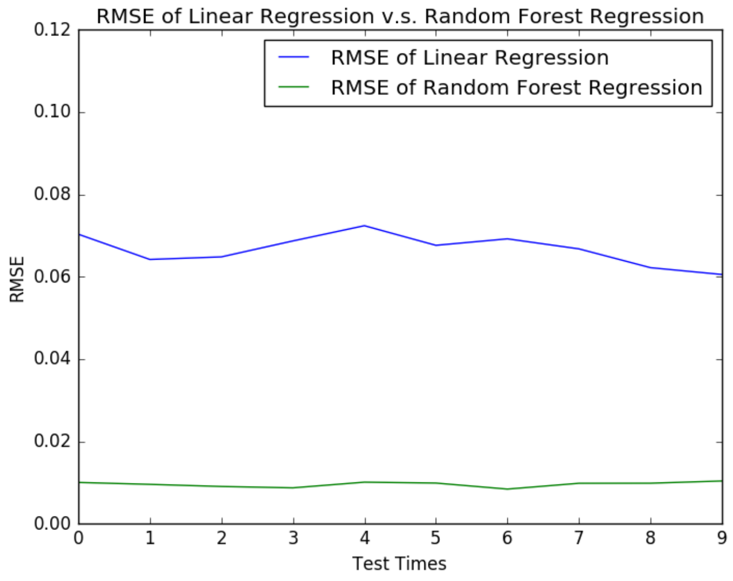


Fig 2b.4. Performance in RMSE compared with the linear regression model

Problem2 c)

In this part, we implement Feed-Forward Network model to make prediction and back propagation algorithm to train the model.

1) Preprocessing the data set.

We use the same two schemes as part a) to preprocess the data set. In the first scheme, we change all the categorical variables into real variables. In the Day of Week attribute, we map {Monday, Tuesday,..., Sunday} to {0,1,...,6}. Similarly, the workflow ID and File Name are mapped to {0, 1, 2, 3, 4} and {0, 1,..., 29}. The inputs of the model are the six attributes and the target is backup size. Since all the real-valued variables are take discrete values, it is reasonable to consider these variables as categorical variables. So in our second scheme, we consider all the variables as categorical variables. For a variable with D categories, we convert them into D dimensional (0/1) variables. The inputs of the model become to 68 attributes and the target still is only one.

2) Experiment

Our major parameters of the model are number of hidden layers and number of hidden units (we call it NHU for short) of each hidden layers. We also compared the performance of the models with two different numbers of inputs. We implement 10-fold Cross-validation to test our model and consider the average RMSE of 10 tests as the final result. The figures of predicted value vs. real value are plotted based on the results of 10th test of each experiment.

We use the pybrain module to build network. Since the data set is large and the train period takes a long time, we try to shorten the training time by limit the maximum epochs. For our data set, we verify that there is no difference between the output when maximum epoch takes 40 and 100. So, in our experiment, maximum epoch equals to 40, which will not influence the accuracy of the output and will greatly shorten the time.

i) One hidden layer

In this part, for two different types of inputs, the number of hidden units is changing from small to large. Since the number of input in second type of data is about 10 times larger than that in first type, we design different numbers for these two experiments in case of overfitting.

Results and figures of our experiments are as follows. When there is only one hidden layer, the minimal RMSE of 6 inputs is around 0.0594200377607 associated, associated with NHU taking the value around 100. When NHU equals to 2, the model almost can't predict anything since the output are all around 0. As NHU increasing, the performance is getting more precise. For 68 inputs, we can see the same the discipline. The minimal RMSE is 0.0511611196481, associated with NHU taking the values around 700.

Table 1 RMSE with number of inputs = 6

number of input	NHU	RMSE
6	2	0.103941702333
6	3	0.0924635178063
6	6	0.0794219384714
6	20	0.0764959516833
6	30	0.0713037503197
6	45	0.0670731180751
6	80	0.0669149000462

6	100	0.0594200377607
6	120	0.0595996387572
6	150	0.0630575262345
6	300	0.0708889504532
6	500	0.0941152450134

Table 2. RMSE with number of inputs = 68

number of input	NHU	RMSE
68	20	0.0819660974381
68	50	0.0720212415407
68	100	0.0696732229549
68	150	0.0622442570474
68	250	0.0612778316431
68	300	0.062792282257
68	350	0.0629670176909
68	400	0.0620882524956
68	500	0.0618893930552
68	700	0.0511611196481
68	800	0.0557012062076

68	1000	0.139625907454

According to the results, when there is only one hidden layer, we can infer that the number of hidden units can influence the precision of prediction, which especially shows on the predicted results of infrequent values (size around 1.0 in this case). However, it is not in a linear relation. The results will be imprecise when NHU is too small. When NHU is too large, the RMSE of ten tests become unstable. The range extends from 0.025 to 0.04 (as shown in Fig 2c.3). Comparing the results of two different types of input, it is reasonable to conclude that it is the ratio of hidden units number and input number, instead of 'absolute' number of hidden units, which influence the performance. We can generally conclude that the best RMSE can be obtained when ratio of NHU over number of inputs is taken values between 10 and 20.

Besides, 68 inputs and 6 inputs have almost the same best RMSE. As can be seen in Fig 3 and 4, in the 68 inputs case, RMSE of 10 tests are more stable compared with those in 6 inputs case. With more input, it is possible to get a more complex network structure with more parameters. It is possible to find more relation between attributes and target, which can lead to a more stable model and more precise results. However, it is more likely to get overfitting with large inputs. In this case, we have 16279 records in training set, which is much larger than the number of attributes. So it is reliable to use our second preprocessing scheme.

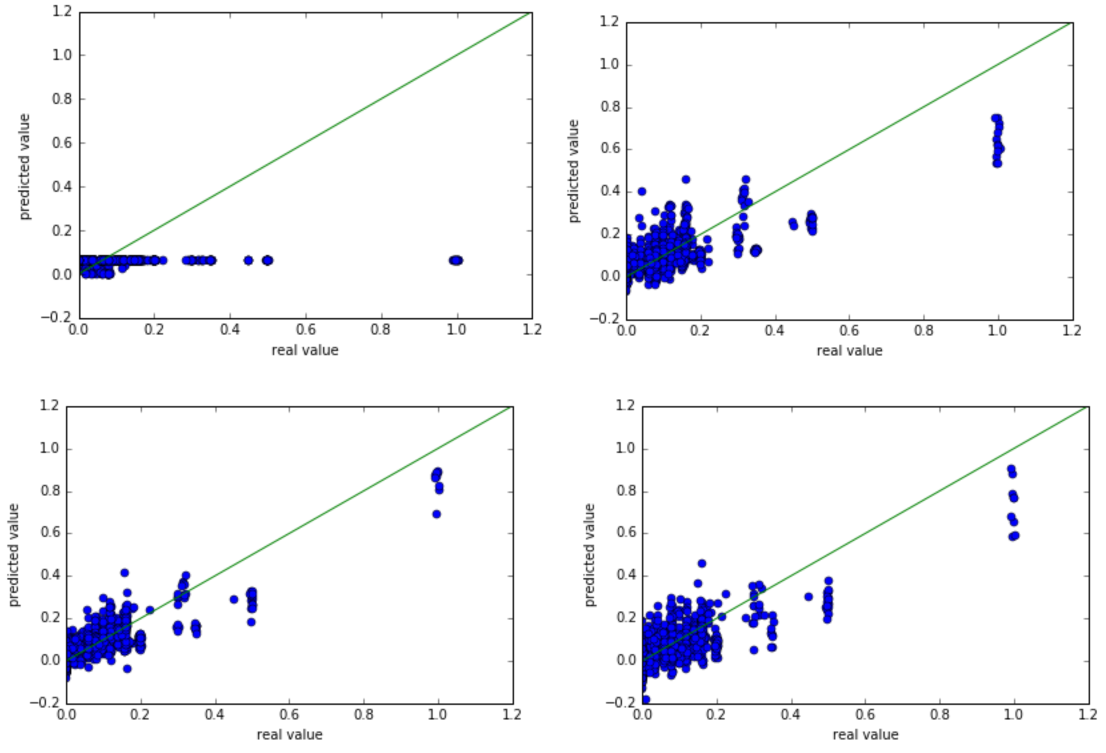


Fig 2c.1. Predicted results from one hidden layer, with number of input = 6, NHU = 2,30,100,300

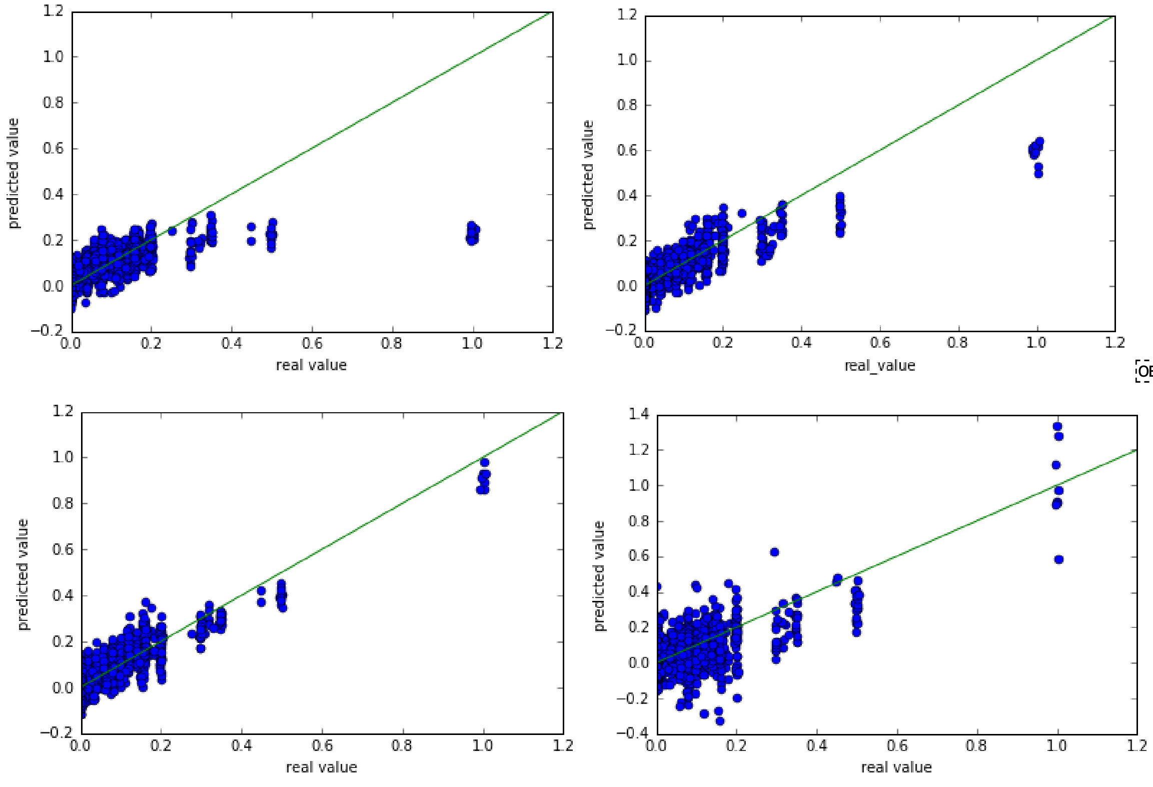


Fig 2c.2. Predicted results from one hidden layer neural network with number of input = 68, NHU =20,100,700,1000

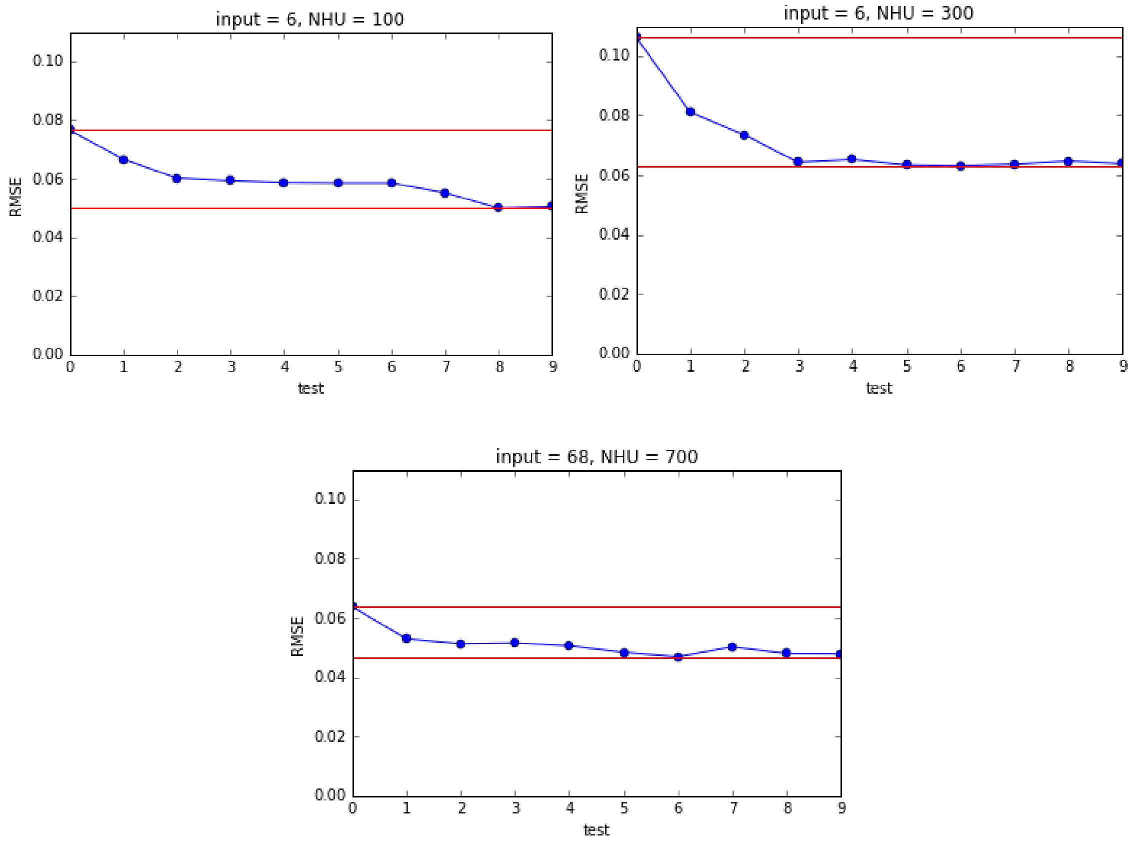


Fig 2c.3. Figures of RMSE in 10 test with number of input = 6 with NHU = 100, 300 and input = 68 with NHU = 700

ii) Two hidden layers

In this part, we add one more hidden layer. According to the figures below, the predicted results of infrequent values (such as 1.0) become more accurate and the variance of predicted values associated with close real values become smaller. After we add one more layer, the performance is improved in general, with minimal RMSE decreasing from 0.06 to 0.04. Meanwhile, the number of hidden units on each layer doesn't make obvious difference on the RMSE. It is hard to decide the relation between RMSE and the number of hidden units on each layer. As the number of hidden units on each layer increasing, the network structure become more complicated, which lead to longer training time and larger probability to get overfitting.

Table 2c.1. RMSE of 2 hidden layers network with number of inputs = 6

# of input	NHU of layer 1	NHU of layer 2	RMSE

6	50	25	0.0472515233912
6	50	50	0.0417471999952
6	50	100	0.0411425377174
6	100	30	0.0362188593537
6	100	50	0.0365146599854
6	100	100	0.0437623646733
6	100	150	0.0475202217249
6	150	75	0.0429635014287
6	150	150	0.0456357160211

Table 2c.2. RMSE of 2 hidden layers network with number of inputs = 68

# of input	NHU of layer 1	NHU of layer 2	RMSE
68	100	50	0.0493185012137
68	100	100	0.0414014881661
68	100	150	0.0462256932171
68	150	75	0.0469119030716
68	150	150	0.0442919339162
68	150	225	0.0411316792785
68	200	100	0.0430905779956

68	200	200	0.0449389816899
68	200	300	0.0440403361516
68	250	125	0.0410723008964
68	250	250	0.0482466568617

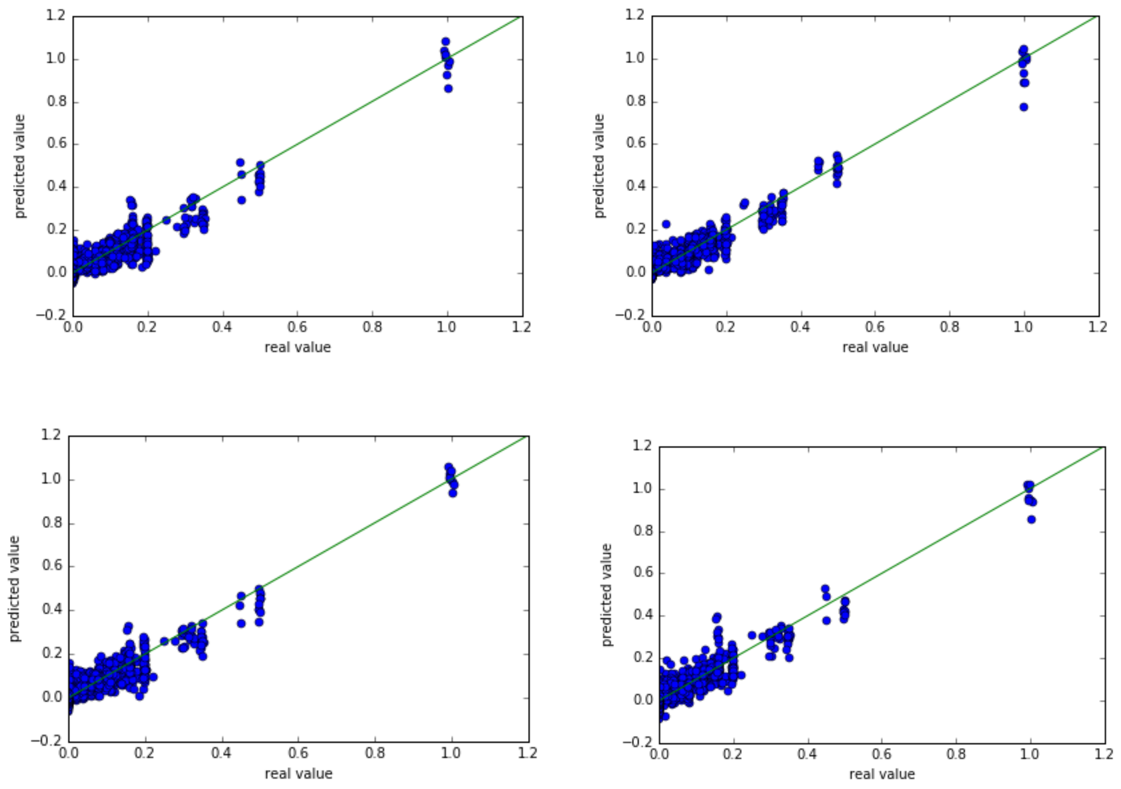


Fig 2c.4. Predicted results from two hidden layers neural network, with number of input = 6, with NHU = (50,25),(100,30),(100,100),(100,150)

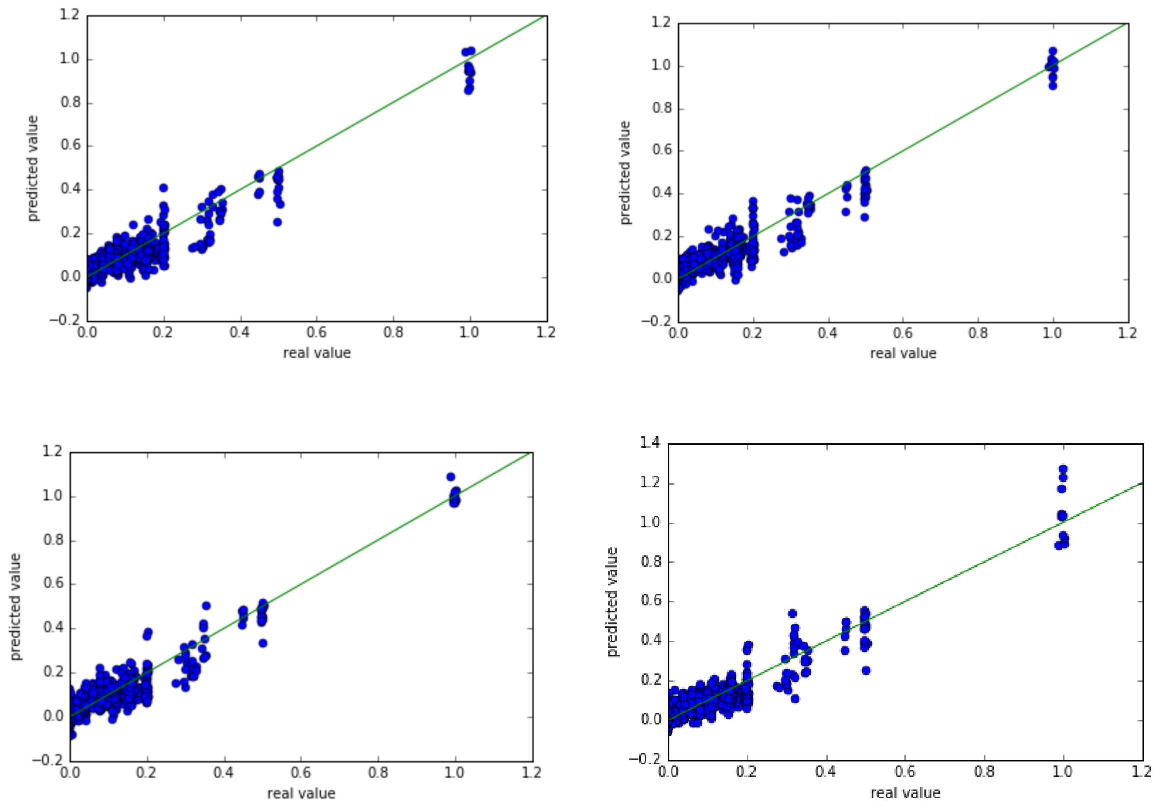


Fig 2c.5. Predicted results from two hidden layers neural network with number of input = 68 and NHU = (100,50),(150,225),(250,125),(250,250)

We also tried the neural network with three hidden layers. In our limited experiment time, we only test the a few sets of parameters and didn't find better RMSE. In theory, the more layers there is, the more precise predicted results we can get. We hope can finish our experiment on 3-hidden-layers network in the future.

Problem3

1) Fitting a piecewise linear regression model

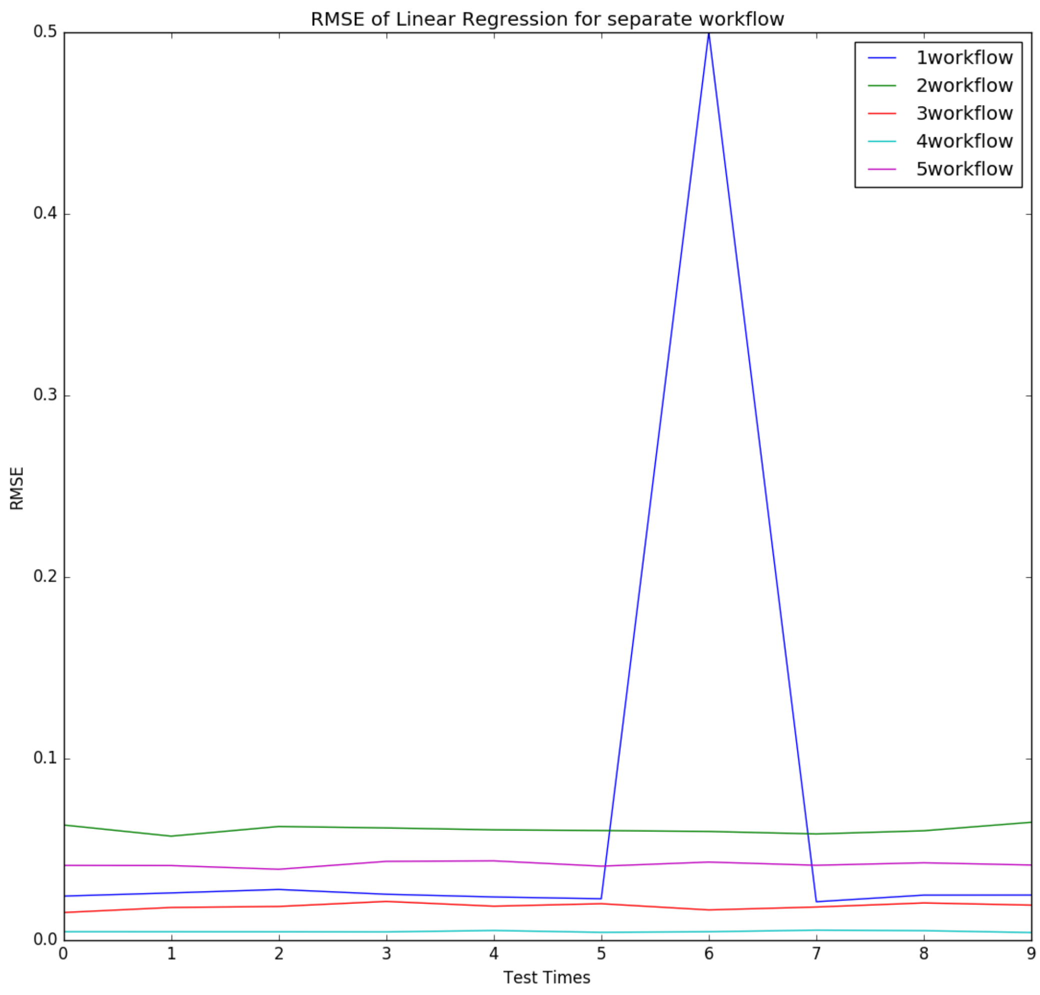


Figure 3-1 RMSE of linear regression model for separate workflows

(To show in one picture, cut that strangely large RMSE of work flow 0 into 0.5.)

The fitting has improved compared to the linear model in 2a.As shown in the picture, the RMSE for fitting the curve separately decreases nearly for every training except in work flow 0, comparing to training in whole data set(0.067). The average RMSE for 10 folds cross validation are 949263357.117, 0.0609124074618, 0.0185991173438, 0.00470450640717, 0.0416733456342.

In fact, there is always a big RMSE for the work flow 0. It should not be deduced that this problem lies in this specific model. As shown in the plot of actual copy sizes of all files on a time period of 20 days, the file 5 always deviate the pattern. When fitting for the whole data set, its impact can be adjusted by other data, but when training separately, the problem arises.

In all, the piecewise linear model performs better which shows that the prior understanding of data(such as finding the pattern between week and backup size) do help a lot in building a more reasonable, suitable fitting model.

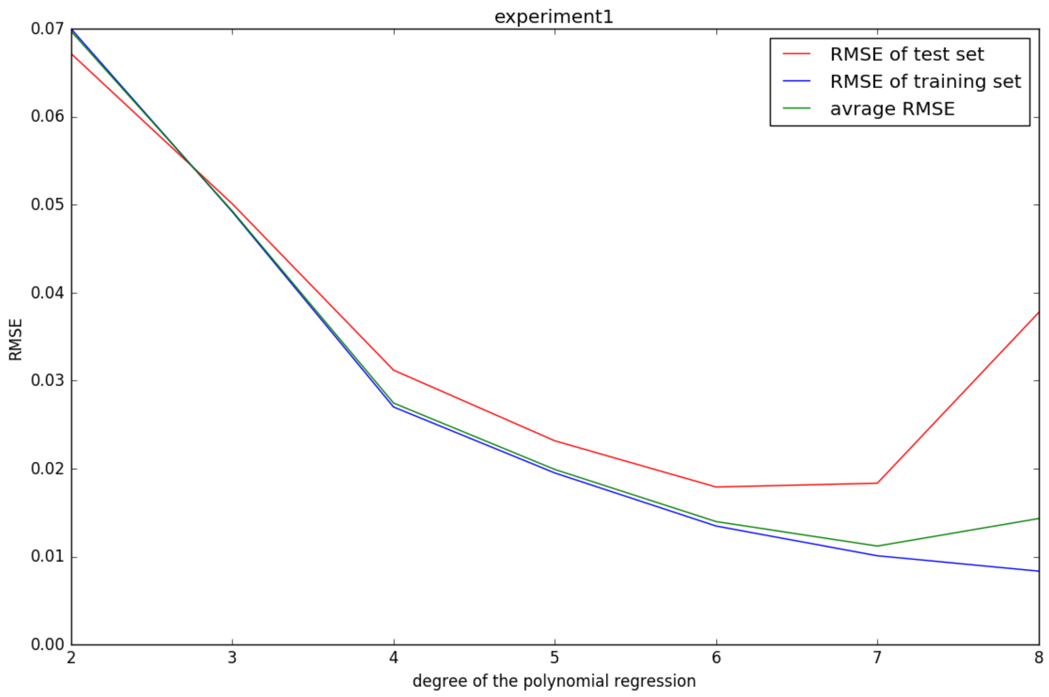
2) Polynomial model

When using polynomial function to fit the data, we introduce high degree variables as input. For example, when fitting function whose degree is 2, we input 27 variables.

$$X = [X_1, X_2, X_3, X_4, X_5, X_6] \Rightarrow X = \left[X_1, \ldots X_6, X_1X_2, \ldots X_5X_6, X_1^2, \ldots X_6^2 \ \right]$$

Then we are going to find the coefficient for each input.

To get a general idea of how the model's degree affects our fitting results, we first fix the training data set as the first 9 parts of the 10 parts in whole data sets, the rest 1 part as testing data set. Then training 2-8 degrees model through this training set, and test them through testing set. In addition, we build another group of fixed data sets: last 9 parts of the 10 parts in whole data sets as training data set, first 1 part as testing data set and test our models through it.



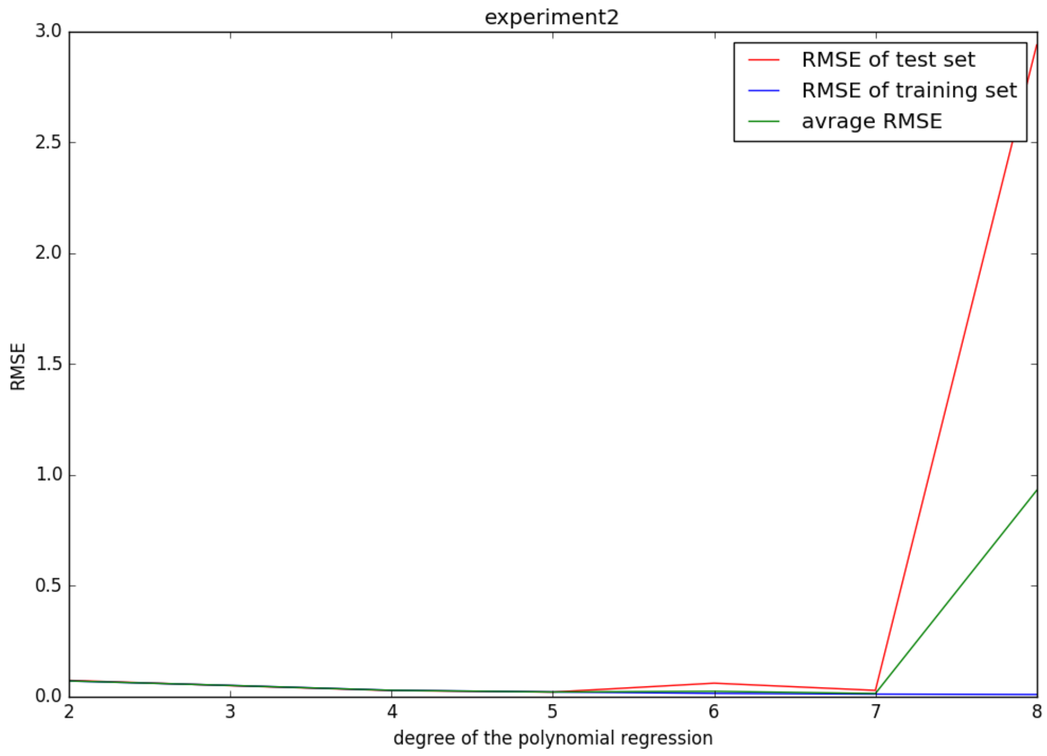


Figure 3.2 two experiments for the RMSE of different degrees of polynomial regression models in fixed training and testing data set

As we can see from the picture, as the degree increases, the RMSE of Training set decreases, which shows that model fits the given data set better when it comes complex. It is reasonable because when the model has more dimensions, it have more chance to get closer to every data.

However, for the test set, there is an obvious increasing in RMSE for degree>=6 in our two experiments. So it is definitely existing overfitting problem for degree>=6 when the model has tremendous difference between training results and testing results.

Moreover, from the first picture, when degree>2,the RMSE of testing set is larger than that of training set, but since they are all below 0.5, it is hard to decide whether there is an overfitting problem. We do need more experiments to test and observe our model.

In the next stage, we use 10-folds cross validation to train and test our models.

10 folds training and test process gives us the chance to get a relatively general performance of the model compare to the fixed training and test data set. When we analyze whether a model is too complex to overfit the data or too simple to be unable to describe the relations between input and output, we need the model's general performance. From the fixed data sets experiments we

know that the RMSE of training set is decreasing as degree increases, so we only need to keep track of the RMSE of testing set to analyze the model’s performance.

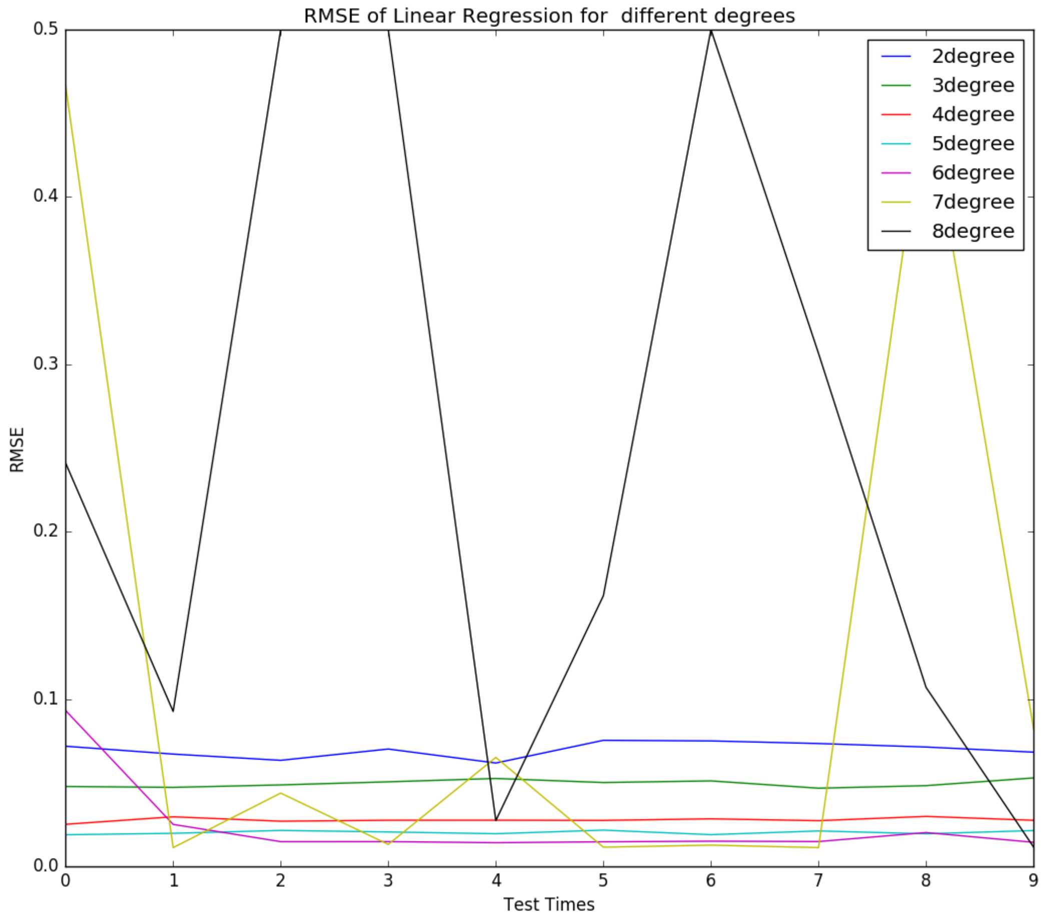


Figure 3.3 RMSE of Linear Regression for different degrees polynomial regression model in 10 folds cross validations(to better show in one graph, cut RMSE>0.5 parts into 0.5)

In the picture, only degree 7,8 show unsteady RMSE during 10 tests. Degree 6 has a large RMSE once. For degree<6, there are steady and decreasing RMSE.

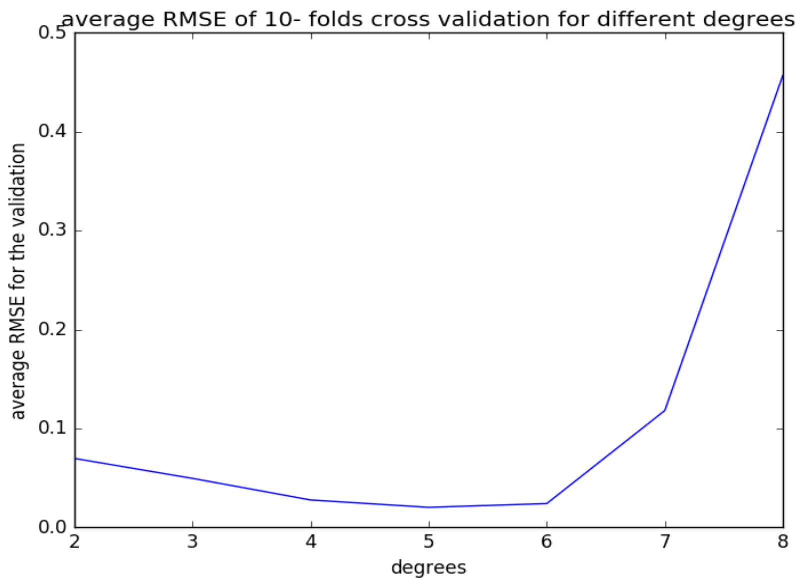


Figure 3.4. Average RMSE for different degrees polynomial regression model in 10 folds cross validations

In this average RMSE picture, degree 5 has the lowest RMSE. Combined with the observation from first picture, we can say that degree 5 is a suitable polynomial model for this problem.

So in this experiment, we control our model’s complexity by avoiding overfitting problem, which can be observed from the large gap between RMSE of training set and testing set. 10 folds cross validation gives random training and test data set, which helps us get the average RMSE of the model, which shows the model’s general performance.

This model has an obvious problem, its input dimension will increase as power function of variables, and exponential function of degree. Though it may finally find a suitable model through trying across possible degrees, it needs large calculations and wastes time in getting results from several meaningless inputs. Moreover, as its scale comes larger, it requires larger training and testing data set to reduce the overfitting issue, which may be demanding for practical problem.

Conclusion:

Comparing these fitting models for regression, linear model is not good enough to fit the hidden pattern. However, after we utilize the observed relations behind workflows, a piecewise linear model improves its performance. Furthermore, if we advance the linear model into polynomial

Ones, effects of combined variables on the output has been taken into model, it fits the data better. For neural network regression, its performance is mainly decided by its structure.

Although it can achieve low RMSE for specific structure, it takes time and skill to find the structure. The best regression result is achieved by Random Forest Model. It is fast and has obvious meaning for the trees, convenient for us to interpret and explore the hidden patterns behind the data set.

To control the complexity of our models, in this part, we use the 10 folds cross validation and for the following parts, we use the method of regularization.

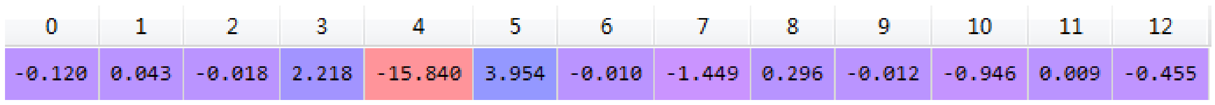
II. Boston Housing Dataset

Problem 4

Task 1: Analyze the significance of different variables with the statistics obtained from your model.

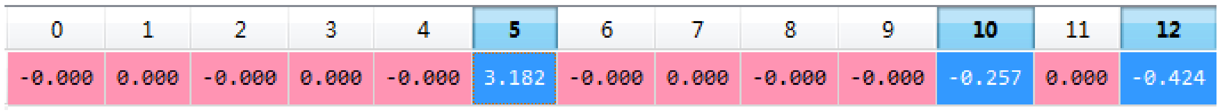
By performing a 10 fold cross validation, the coefficients of different variables is shown below. Indexes from 0 to 12 indicate features from “CRIM” to “LSTAT”. The coefficients indicate the significance of different variables and their corresponding features. If we compare the coefficients generated from Linear Regression in Table. 4.1 and those generated from Lasso Regression in Table. 4.2, we may find that the 6th, 11th and 13th elements of coefficient array have main impacts on the feature of “MEDV”, those features are “RM”,”PTRATIO” and ”LSTAT”.

Table 4.1. Coefficients Corresponding to 13 Features in Linear Regression



0	1	2	3	4	5	6	7	8	9	10	11	12
-0.120	0.043	-0.018	2.218	-15.840	3.954	-0.010	-1.449	0.296	-0.012	-0.946	0.009	-0.455

Table 4.2. Coefficients Corresponding to 13 Features in Lasso Regression



0	1	2	3	4	5	6	7	8	9	10	11	12
-0.000	0.000	-0.000	0.000	-0.000	3.182	-0.000	0.000	-0.000	-0.000	-0.257	0.000	-0.424

Task 2: Report your obtained Root Mean Square Error.

RMSE of Linear Regression versus test times is shown in Fig. 4.1. The maximum value of RMSE is 7.0456, the minimum value is 3.8976 and the average value is 4.7587.

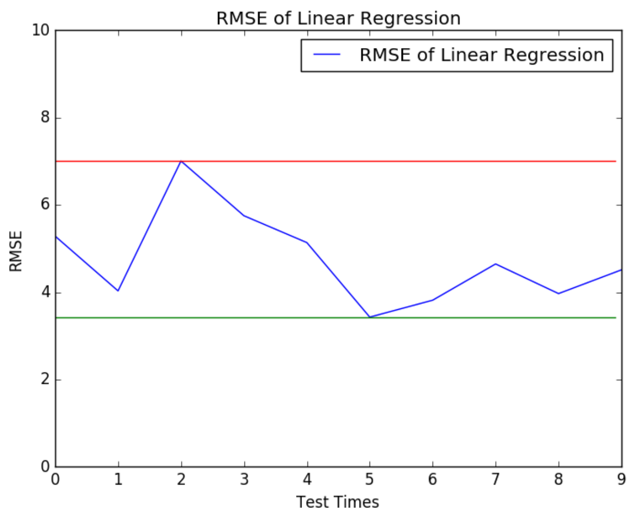


Figure 4.1: RMSE versus Test Times in Linear Regression

Task 3: Provide “Fitted values and actual values scattered plot over time”.

Task 4: Provide “Residuals versus fitted values plot”.

We also provide graphs to evaluate how well linear model fits the data. As shown in Fig. 4.2 (Left), the actual values and their corresponding fitted values locate around y=x, which is the line for ideal fitting model. The other graph shown in Fig. 4.2 (Right), depicts the residuals located around zero line, which is also the line for ideal fitting model. Since most of the fitted data locates around y=x line, and most residuals cluster around zero line, we can confirm that the linear regression model fits the data with reasonably accurate performance.

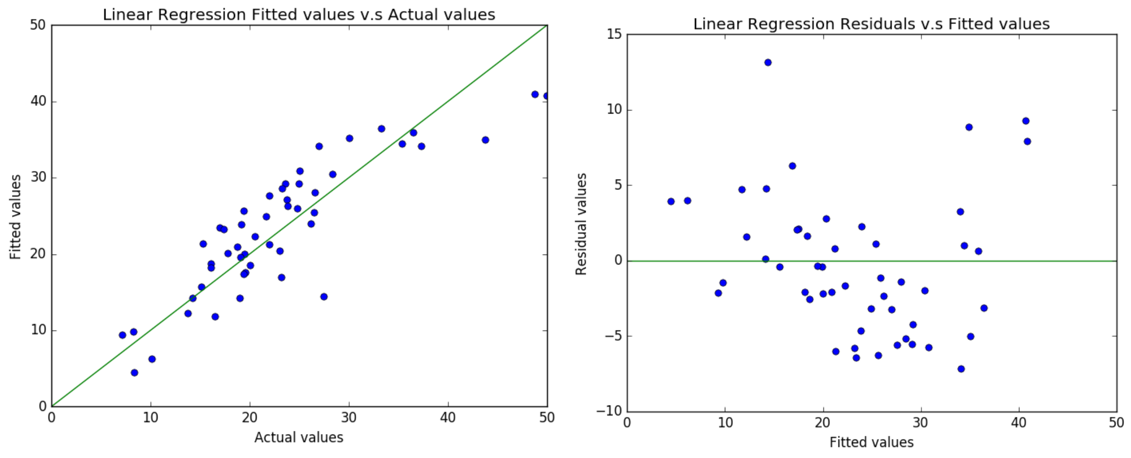


Figure 4.2. Fitted Values versus Actual Values (Left) and Residuals versus Fitted Values (Right)

Task 5: For a polynomial regression function, find the optimal degree of fit as in part 3.

We use the same Polynomial Regression model to analyze Boston dataset, just like in part 3. Fig .4.3 shows the relationship of RMSE of Polynomial Regression versus different polynomial degrees. We can clearly see that RMSE remains low when polynomial degree equals to 1 or 2. However, as polynomial degree increases, we spot a sudden increase of RMSE at polynomial degree 3. After that, RMSE gradually deteriorates as the polynomial degree increases. The detailed RMSE versus polynomial degree data is listed in Table.4.3.

Table 4.3. Detailed Statistics of RMSE versus Polynomial Degree

Poly Degree	RMSE
1	3.29
2	8.72
3	1849.38
4	65.73
5	72.45
6	85.38

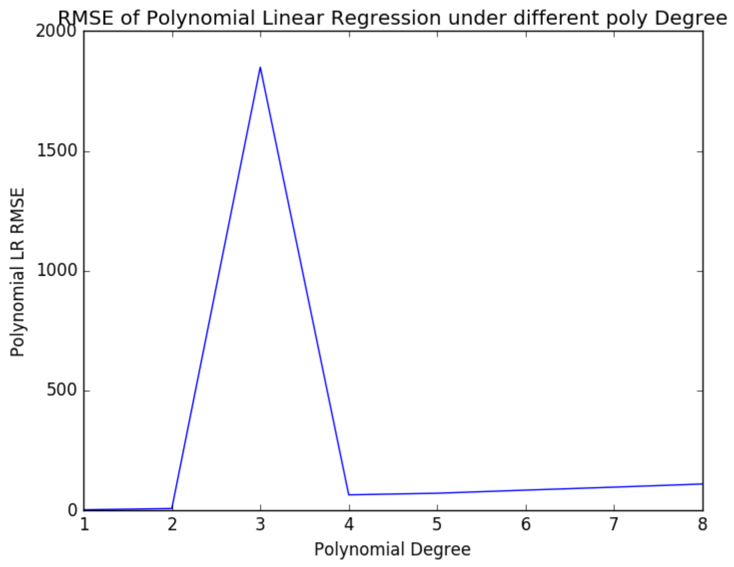


Figure 4.3. RMSE of Polynomial Regression versus Different Polynomial Degrees

The average RMSE versus polynomial degree with 10-fold cross validation is depicted in Fig .4.4. Similar to the graph and table above, the average RMSE achieves its lowest at polynomial degree 2, rockets at polynomial degree 3, and gradually deteriorates as the polynomial degree increases. The minimum RMSE achieved at polynomial degree 2 is 3.75. Detailed information of average RMSE versus polynomial degree data is listed in Table .4.4.

Table 4.4. Detailed Statistics of RMSE versus Polynomial Degree

Poly Degree	RMSE
1	4.84
2	3.75
3	1985.45
4	42.09
5	51.39
6	34.73

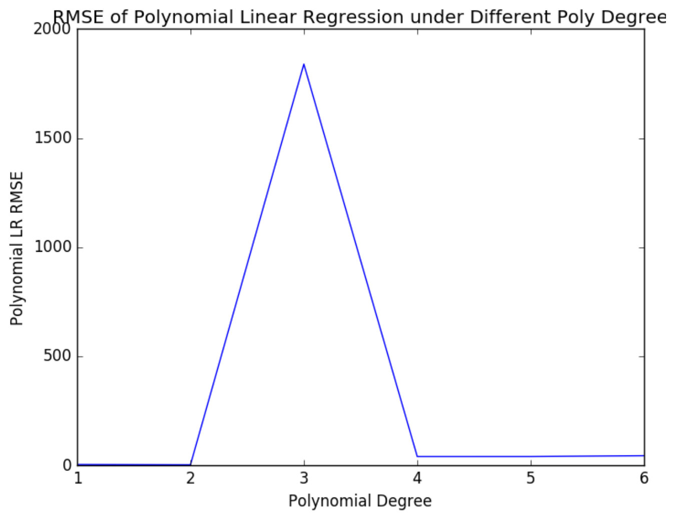


Figure 4.4. Average RMSE of Polynomial Regression versus Polynomial Degree

From the discussion above, we can get the optimal degree of fit happens at degree equals to 2. And we set polynomial degree 2 as the threshold on the degree of the fitted polynomial beyond which the generalization error of our model gets worse.

Problem 5

Lasso and Ridge Regularization

Introduction: In statistics and machine learning, overfitting occurs when a statistical model describes random error or noise instead of the underlying relationship. A model that has been overfit will generally have poor predictive performance, as it will exaggerate minor fluctuations in the data. In the parts above, we use cross validation to limit the affection of overfitting. We can determine if a given model is good or bad in that way. In this part, we introduce loss function in the method of regularization to achieve little overfitting.

Task 1: Tune the complexity parameter α of the Ridge Regression below in the range {0.1,0.01,0.001} and report the best RMSE obtained via 10-fold cross validation.

Ridge regression addresses some of the problems of ordinary least squares by imposing a penalty on the size of coefficients. The ridge coefficients minimize a penalized residual sum of squares:

$$\min \parallel Y - X\beta \parallel_2^2 + \alpha \parallel \beta \parallel_2^2$$

Here, $\alpha \geq 0$ is a complexity parameter that controls the amount of shrinkage: the larger the value of , the greater the amount of shrinkage and thus the coefficients become more robust to collinearity.

We set the value of α in the range {0.1, 0.01, 0.001} and the performance of Ridge Regression model is shown in Fig. 4.5. The curves with different α values stay close to each other. The best RMSE spotted for Ridge Regression is 3.04, which is achieved with α=0.1, confirming that with the control of overfitting, we may achieve better performance than Linear Regression and Polynomial Regression model.

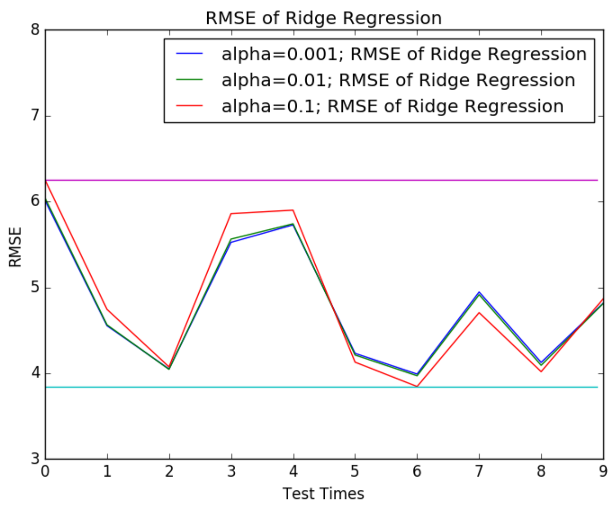
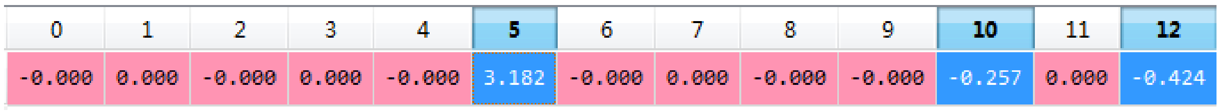


Figure 4.5. RMSE of Ridge Regression versus Test Times under Different α

Task 2: The **Lasso** is a linear model that estimates sparse coefficients. It is useful in some contexts due to its tendency to prefer solutions with fewer parameter values, effectively reducing the number of variables upon which the given solution is dependent.

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

We also set α in the range {0.1, 0.01, 0.001} and the performance of Lasso Regression model is shown in Fig. 4-6. The curves with different α values have clear distinction. Generally the RMSE is smaller with smaller α value and vice versa. The best RMSE is 3.49, which is achieved at α=0.001. The best RMSE is better than that of Linear Regression. And if we look into the coefficients of different features generated by Lasso Regression, as in Table. 4.5, we may notice that when α=0.1, 10 out of 13 coefficients are zeros, and the 3 nonzero coefficients correspond to features of "RM","PTRATIO" and "LSTAT", which are exactly the three main factors that have impacts the "MEDV" feature.



0	1	2	3	4	5	6	7	8	9	10	11	12
-0.000	0.000	-0.000	0.000	-0.000	3.182	-0.000	0.000	-0.000	-0.000	-0.257	0.000	-0.424

Figure 4.5. Coefficients Corresponding to 13 Features in Lasso Regression

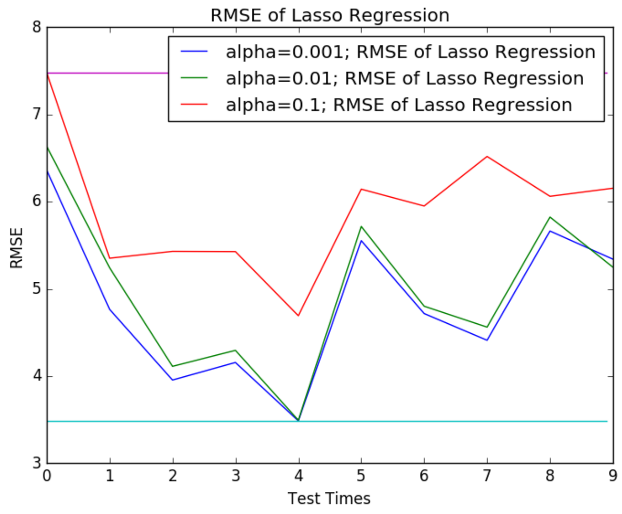


Figure 4.6. RMSE of Lasso Regression versus Test Times under different α