ECE 219 Project4

Evelyn Chen UID: 704332587
Jack Gong UID: 005025415
Jiuru Shao UID:204288539
Haoxiang Zhang UID:104278461

March 5th 2018

1 Introduction

In this project, we explore some basic regression models on Network backup Dataset, which consist of simulated traffic data on a backup system over a network. The system monitors files and the size of data and during are logged. The dataset has 18,000 data points with variables: week index, data of week, backup start time, workflow ID, file name, backup size, and backup time. The regression models we use include linear regression model, random forest regression model, neural network regression model, and KNN regression model. We also explored techniques to handle overfitting, such as using cross-validation and regularization.

For each regression, we report training and test RMSE from 10 fold cross validation. In addition, we plot fitted values against true values over number of data points and residuals vs fitted values over number of data points for each regression.

2 Q1

2.1 1a

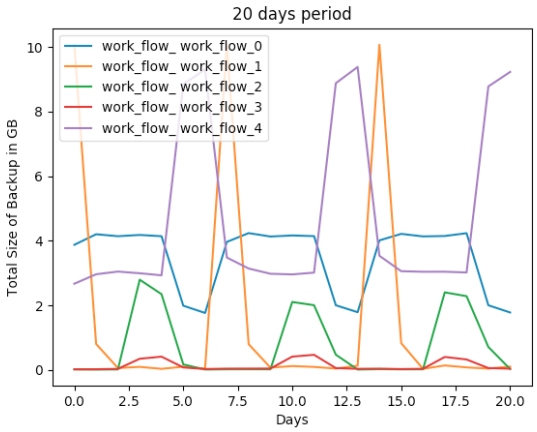


Figure 1: **20 days period**

2.2 1b

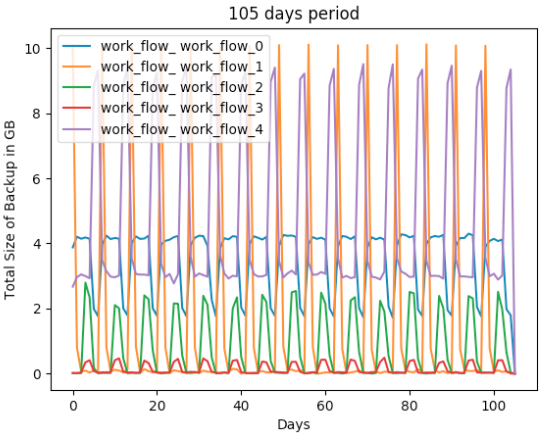


Figure 2: **105 days period**

As the figures shows above, all work flows have predictable pattern - although the 20 days period is not as obvious - the individual work flow peaks regularly, with the period of seven days

3 2a

3.1 i

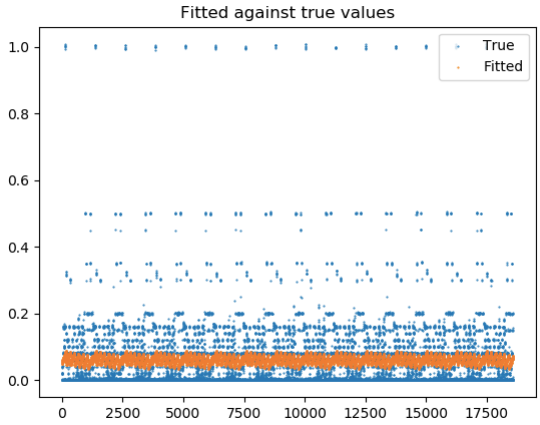


Figure 3: **Fitted values against true values over number of data points**

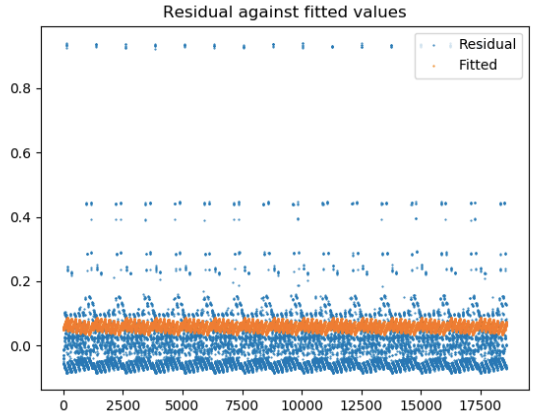


Figure 4: **Residuals vs fitted values over number of data points**

	Train RMSE	Test RMSE
Error	0.103585	0.103676

Table 1: **Metrics of linear regression model (Scalar Encoding)**

The fitted against true value plot and residual plot suggest that the linear regression does not fit very well.

3.2 ii

We standardize all numerical features, then fit and test the model.

	Train RMSE	Test RMSE
Error	0.103585	0.103676

Table 2: **Metrics of linear regression model (Standardize)**

After standardization, the test and train RMSE are still the same as before standardization. After we plot the fitted against true value and residual, the plots look the same as before standardization as well. With the standardization for linear regression, we get the exact same model, and thus same accuracy. This is why we get same test RMSE, train RMSE, and same graphs.

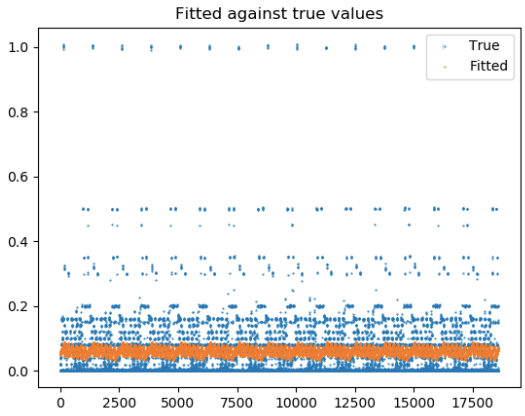


Figure 5: **Fitted values against true values over number of data points**

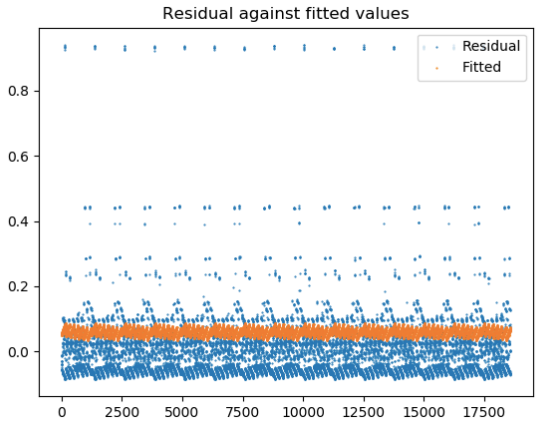


Figure 6: **Residuals vs fitted values over number of data points**

3.3 iii

We use f regression and mutual information regression to select the three most important variables respectively. For f regression for all features, we get:

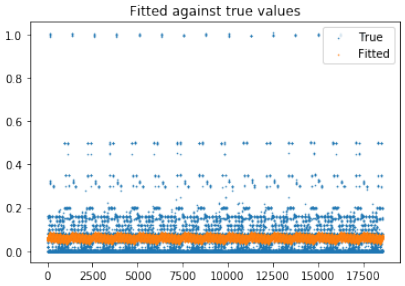
[8.45e-03 3.88e+01 1.51e+02 2.61e+01 2.53e+01]

This means the second, third, and forth features are the most important according to f regression.The second, third, and forth features are day of week, backup start time - hour of day, and work flow ID. We then used these three features to train a new linear model and reported the test RMSE and train RMSE.

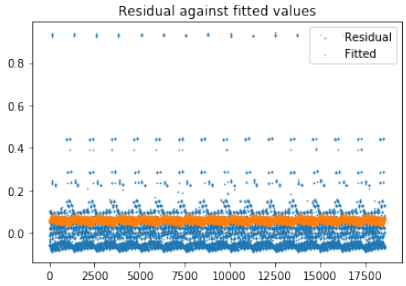
	Train RMSE	Test RMSE
Error	0.103585	0.103676

Table 3: **Metrics of linear regression model (f_regression)**

Followings are plots of best f regression model:



(a) Fitted against true



(b) Residual against fitted

Figure 7: **Scatter plots (Linear Regression, f regression)**

For mutual information regression for all features, we get:

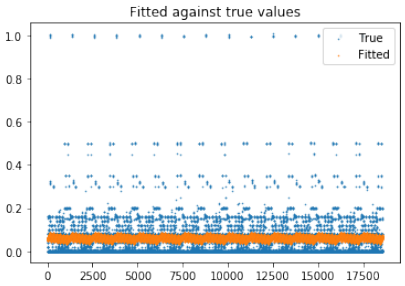
[0.00467879 0.23592485 0.29983709 0.77272196 0.76678531]

This means the third, fourth, and fifth features are the most important according to mutual information.The third, fourth, and fifth features are backup start time - hour of day, work flow ID, and file name. We then used these three features to train a new linear model and reported the test RMSE and train RMSE.

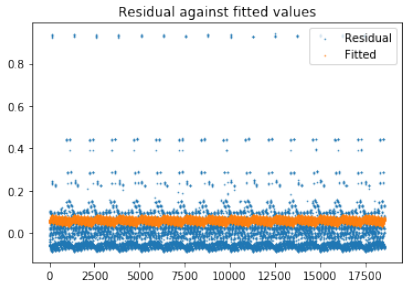
	Train RMSE	Test RMSE
Error	0.10369	0.10377

Table 4: **Metrics of linear regression model (Mutual information)**

Followings are plots of best mutual information model:



(a) Fitted against true



(b) Residual against fitted

Figure 8: **Scatter plots (Linear Regression, mutual information)**

3.4 iv

There are 32 possible combinations of encoding the five categorical variables. Here, we used a mask as a parameter to OneHotEncoder, where [True, True, True, True, True] means all variables are encoded with one hot encoder and [False, False, False, False, False] means no variable is encoded with one hot encoder. We then iterate the 32 possibilities, where all false will be 0. With only the last variable true will be 1, and all true will be 31.

Below, we plot the average training RMSE and test RMSE for each combination in range 1 to 32. Note that the first 16 values are extremely close to 0 (not 0) because the y-axis has very large scale.

By printing out the 32 test and train RMSE, we found out the smallest test RMSE is 0.08850649080774672, which is when the mask is [False, True, True, False, True].

The train RMSE is 0.08833804812051183 with this mask. This essentially is when day of week, backup start time-hour of day, and file name are encoded with one hot encoding. This combination provides the best performance because of the lowest test RMSE. Some variables perform better when encoded in one hot encoding because these variables may be misleading when they are encoded in scalar. For example, Monday and Sunday are just one day apart, but in scalar encoding, they are 1 and 7, which are 6 apart.

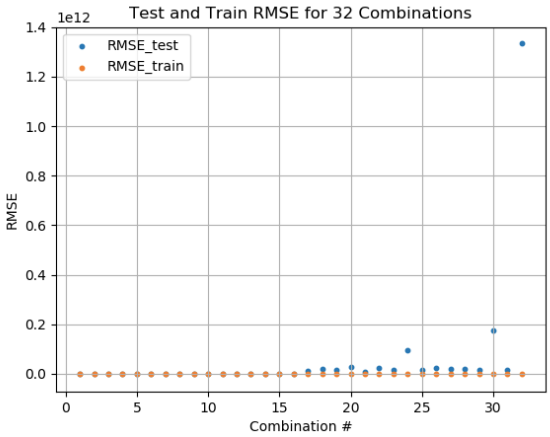
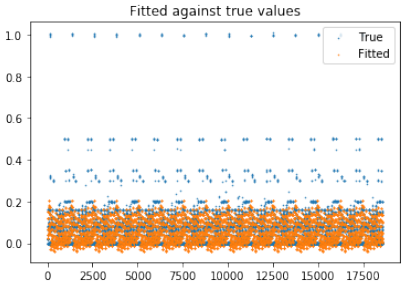
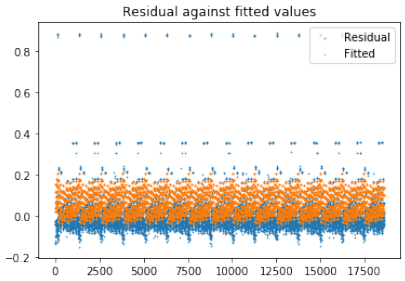


Figure 9: **Test and Train RMSE for 32 Combinations**

Followings are scatter plots of best combination:



(a) Fitted against true



(b) Residual against fitted

Figure 10: **Scatter plots (Best combination)**

3.5 v

We can observe that there's an obvious increase in test RMSE compared to train RMSE in some combinations. Specifically, we found out that test RMSE has a huge increase when the "week" variable is encoded to true. This is due to overfitting.

We can use regularizations to control overfitting. There are three types of regularizers that we will explore here: ridge regularizer, lasso regularizer, and elastic net regularizer.

Ridge Regularizer:

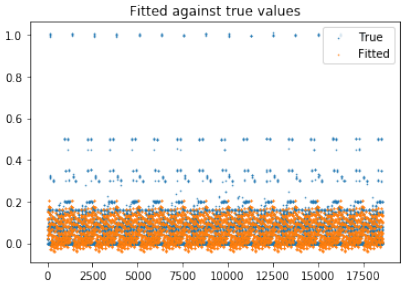
	Train RMSE	Test RMSE
Error	0.0883	0.0885

Table 5: **Metrics of linear regression model (Ridge Regularizer)**

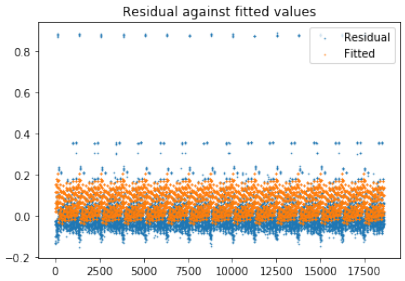
We got 0.0885 for the lowest test RMSE. Train RMSE is 0.0883 with the same parameter setting. This is using [False, True, True, True, False] as the mask for one hot encoding and $\alpha = 10$ as the parameter.

coefficients = [3.91670522e-02 -1.26890387e-02 -2.02009842e-02 -5.51500158e-03 -5.63482941e-03 3.34645903e-03 1.52634264e-03 -2.02300803e-02 -2.09194346e-02 7.10873531e-03 3.39712668e-02 -2.29534913e-03 2.36486187e-03 3.90672096e-02 -1.22296336e-02 -4.07184431e-02 -5.69482599e-02 7.08291270e-02 1.16835718e-05 2.65856761e-05]

Followings are scatter plots of best ridge regularizer model:



(a) Fitted against true



(b) Residual against fitted

Figure 11: **Scatter plots (Ridge Regularizer)**

Lasso Regularizer:

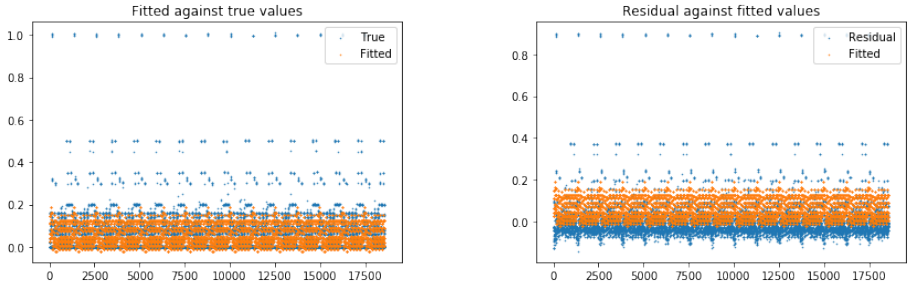
	Train RMSE	Test RMSE
Error	0.0887	0.0889

Table 6: **Metrics of linear regression model (Lasso Regularizer)**

We got 0.0889 for the lowest test RMSE. Train RMSE is 0.0887 with the same setting. This is using [True, True, True, True, True] as the mask for one hot encoding and $\alpha = 0.001$ as the parameter.

coefficients = [-0. 0. 0. -0. 0. -0. 0. 0. 0. 0. -0. 0. -0. -0. 0. 0.03593511 -0.00210386 -0.01025112 -0. -0. 0. 0. -0.01451855 -0.01481428 0.00117583 0.02810024 -0. 0. 0.04624227 -0. -0.02369402 -0.03985359 0.07862426 0. 0. 0. 0. 0. 0. -0. 0. -0. -0. -0. -0. -0. -0. -0. -0. -0. -0. -0. -0. -0. -0. -0. -0. 0. 0. 0. 0. 0. 0.]

Followings are scatter plots of best lasso regularizer model:



(a) Fitted against true (b) Residual against fitted

Figure 12: **Scatter plots (Lasso Regularizer)**

Elastic Net Regularizer:

	Train RMSE	Test RMSE
Error	0.0883	0.0885

Table 7: **Metrics of linear regression model (Elastic Net Regularizer)**

We got 0.0885 for the lowest test RMSE. Train RMSE is 0.0883 with the same setting. This is using [False, True, True, True, False] as the mask for one hot encoding and $\alpha = 0.001$ and l1_ratio = 0. This means we are using $\lambda 2 = 0.001$ and $\lambda 1 = 0$ since $\lambda 1 = \alpha$ * l1_ratio and $\lambda 2$ = alpha*(1-l1_ratio).

coefficients = [3.90603926e-02 -1.26530858e-02 -2.01474828e-02 -5.50084179e-03 -5.61767385e-03 3.33680620e-03 1.52188742e-03 -2.01826542e-02 -2.08670846e-02 7.09198022e-03 3.38908431e-02 -2.29341635e-03 2.36030780e-03 3.91536462e-02 -1.21225593e-02 -4.06375834e-02 -5.69185581e-02 7.05250255e-02 1.16747707e-05 4.03639379e-05]

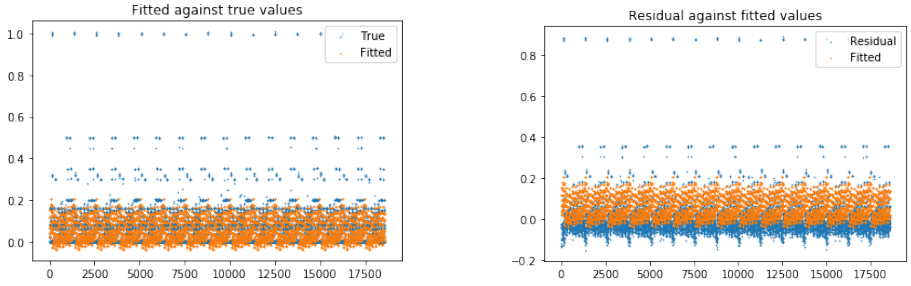
For the unregularized best model that has [False, True, True, False, True] as mask for one hot encoding, we get coefficients as:

[4.50082784e+10 4.50082784e+10 4.50082784e+10 4.50082784e+10 4.50082784e+10 4.50082784e+10 4.50082784e+10 4.50082784e+10 4.72867903e+09 4.72867903e+09 4.72867903e+09 4.72867903e+09 4.72867903e+09 4.72867903e+09 -4.34533516e+08 -4.34533516e+08 -4.34533516e+08 -4.34533516e+08 -4.34533516e+08 -4.34533516e+08 -3.80555631e+08 -3.80555631e+08 -3.80555631e+08 -3.80555631e+08 -3.80555631e+08 -3.80555631e+08 -3.26577746e+08 -3.26577746e+08 -3.26577746e+08 -3.26577746e+08 -3.26577746e+08 -3.26577746e+08 -2.72599861e+08 -2.72599861e+08 -2.72599861e+08 -2.72599861e+08 -2.72599861e+08 -2.72599861e+08 -2.18621977e+08 -2.18621977e+08 -2.18621977e+08 -2.18621977e+08 -2.18621977e+08 -2.18621977e+08 -6.00814819e-05 -5.39778848e+07]

By comparing the coefficients of the regularized good models and the unregularized best model, we see that the coefficients of regularized models are much smaller than the unregularized best model.

Comparing the best test RMSE for the three regularization methods, we see that Ridge and Elastic Net regularizer have similar test RMSE while Lasso has similar but slightly higher RMSE than the one we got with best mask for one hot encoding in 2a(iv). These test RMSE for these regularizers, which is around 0.0885, are all smaller than the test RMSE from 2a(i) and 2a(ii), which is around 0.104, with the help of the best combination of one hot encoding. These three regularizers have controlled overfitting as shown from the difference in the coefficients. By plotting out the fitted against true and residual graph, we can also see that fitted and true values match similarly well compare to 2a(iv), and much better than 2a(i) and 2a(ii).

Followings are scatter plots of best elastic net regularizer model:



(a) Fitted against true

(b) Residual against fitted

Figure 13: **Scatter plots (Elastic Net Regularizer)**

4 2b

4.1 i

	Train RMSE	Test RMSE	OOB error
Error	0.0603	0.0605	0.336

Table 8: **Metrics of Initial Random Forest Model**

Comparing train and test RMSEs with linear regression model, our initial random forest model has better performance.

4.2 ii

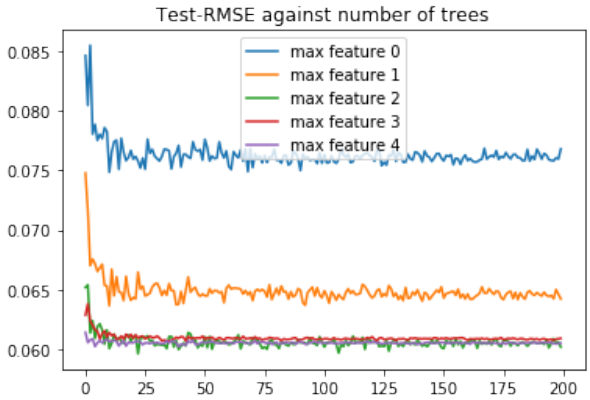


Figure 14: **Test RMSE (y axis) against number of trees (x axis)**

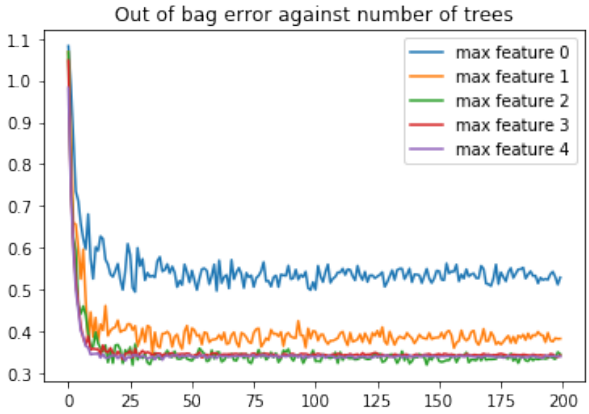


Figure 15: **Out of bag error (y axis) against number of trees (x axis)**

Above are plots for experiments over max number of trees and max number of features. According to the test RMSE metric, the best number of trees is 23 and the best number of features is 3.

4.3 iii

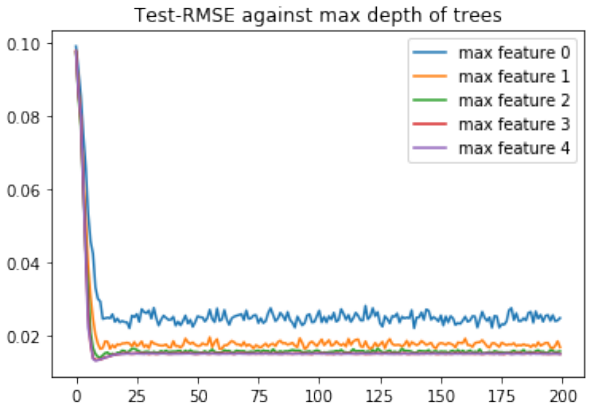


Figure 16: **Test RMSE (y axis) against depth of trees (x axis)**

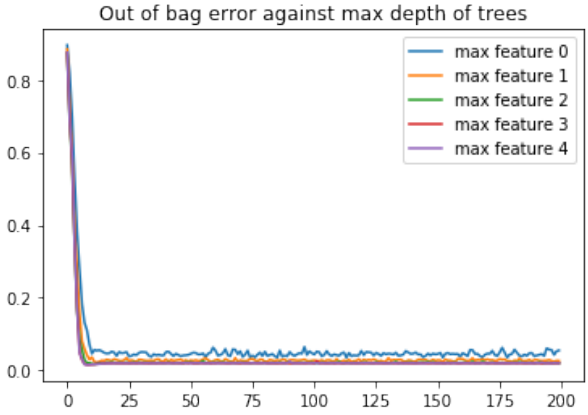


Figure 17: **Out of bag error (y axis) against depth of trees (x axis)**

We choose number of tree depths as another parameter to experiment on. Again, we experiment the tree depth on max number of features from 1 to 5, and above are resulted plots of test RMSE and out of bag errors. According to the test RMSE, the best tree depths is 10 and best number of features is 3.

4.4 iv

From our experients in part ii and part iii, we build our best random forest regression model with 23 as tree number, 3 as feature number and 10 as the tree depth. We report the feature importances in following figure.

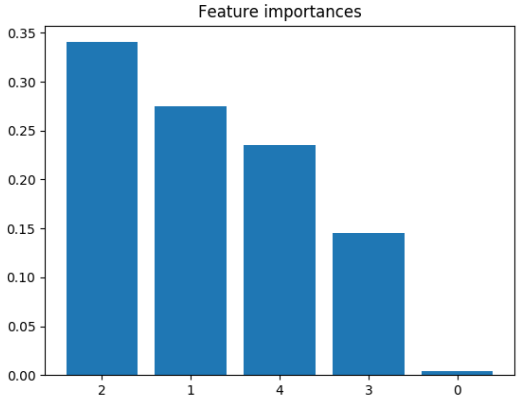


Figure 18: **Feature Importances**

Feature 2 (backup start time, or hour of day), feature 1 (day of week) and feature 4 (file name) are three most important features. Feature 3 (work flow id) is the fourth important feature. Feature 0 (week number) is the least important one.

4.5 v

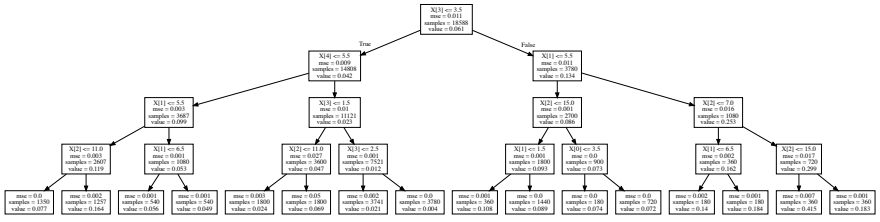


Figure 19: **Visualized Decision Tree**

We pick one tree in best random forest of depth 4 (max features is 3 and max number of trees is 23), and then visualize the decision tree. The root node in this decision tree is feature 3 (work flow id). Howere, it is not the most important feature according to the feature importance reported by the regressor. Reported

feature importance is [3.181e-04 3.189e-01 1.195e-01 2.580e-01 3.034e-01], where feature 1 is the most important one, and feature 3 is the third important one.

4.6

Following plots and metrics are from our best random forest model (max features is 3, max number of tree is 23 and max number of depth is 10). This model has the least test RMSE (best performance) so far.

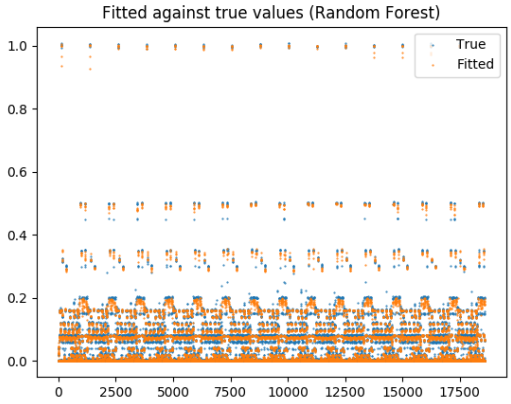


Figure 20: **Fitted values against true values (Best Random Forest)**

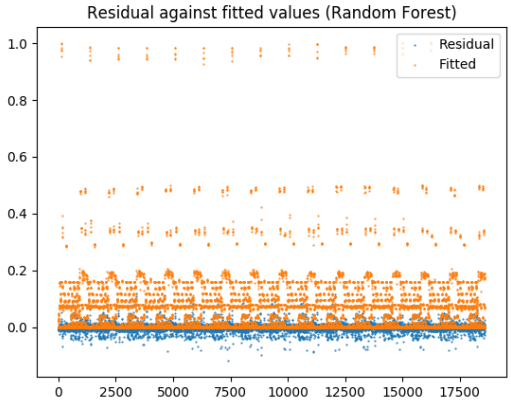


Figure 21: **Residual values against fitted values (Best Random Forest)**

	Train RMSE	Test RMSE
Error	0.01157	0.01369

Table 9: **Metrics of Best Random Forest Model**

5 2c

We scan through 10 to 250, with a step size of 5. So the best number of hidden units reported below is an approximation in each 5 numbers. For relu activation function, the best number of hidden units is 240 from 10 to 250(test RMSE = 0.029, train RMSE = 0.017), for logistic aactivation function, the best number of hidden units is 35 from 10 to 250(test RMSE = 0.089, train RMSE = 0.088), and for tanh activation function, the best number of hidden units is 95 from 10 to 250(test RMSE = 0.073, train RMSE = 0.064).

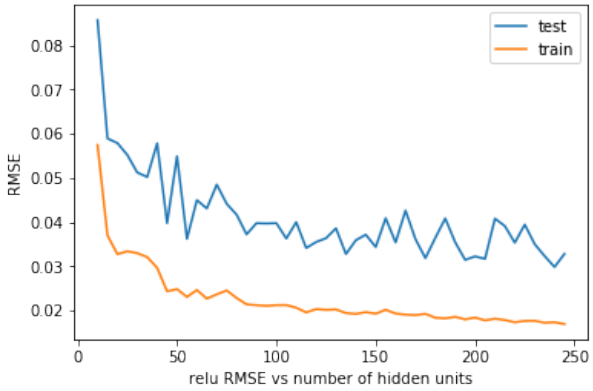


Figure 22: **Train and test RMSE vs number of hidden units using relu. The best number of hidden units is 240 from 10 to 250.**

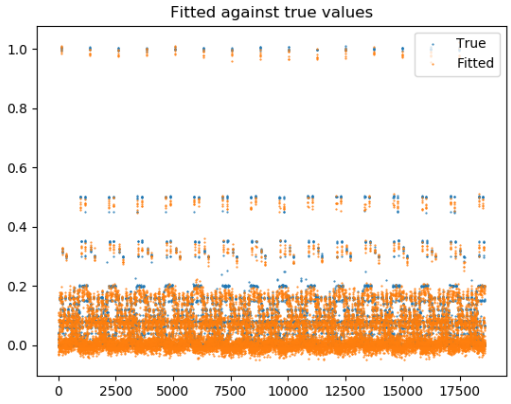


Figure 23: **Fitted values against true values using Neural Network regression model(relu activation function) (with the optimal number of hidden units we find.)**

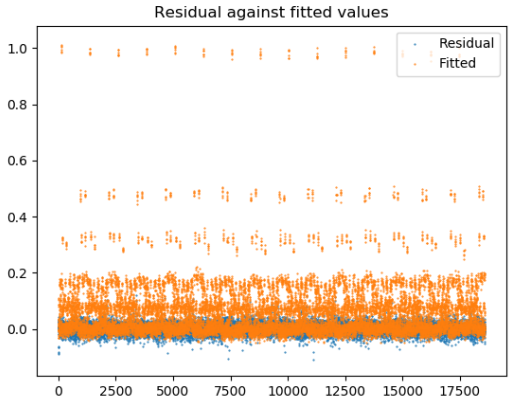


Figure 24: **Residuals vs fitted values using Neural Network regression model(relu activation function) (with the optimal number of hidden units we find.)**

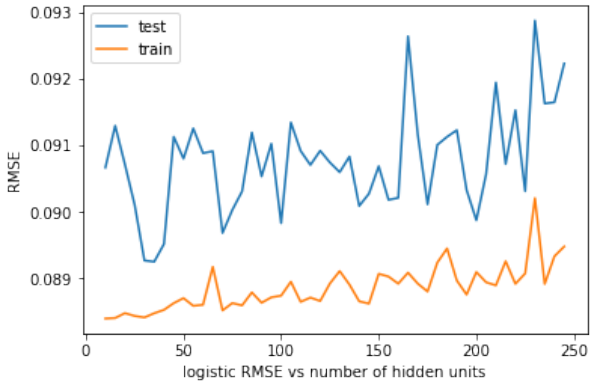


Figure 25: **Train and test RMSE vs number of hidden units using logistic. The best number of hidden units is 35 from 10 to 250.**

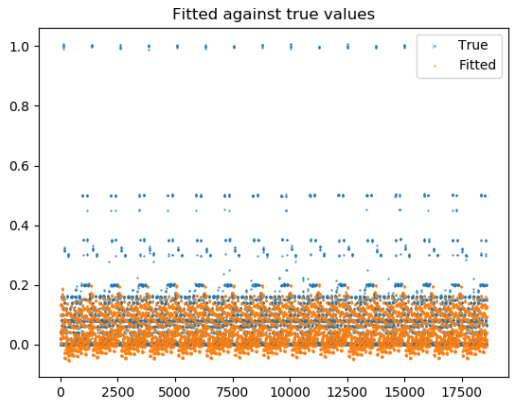


Figure 26: **Fitted values against true values using Neural Network regression model(logistic activation function) (with the optimal number of hidden units we find.)**

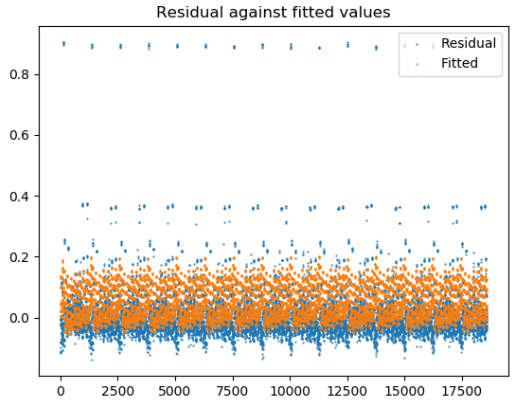


Figure 27: **Residuals vs fitted values using Neural Network regression model(logistic activation function) (with the optimal number of hidden units we find.)**

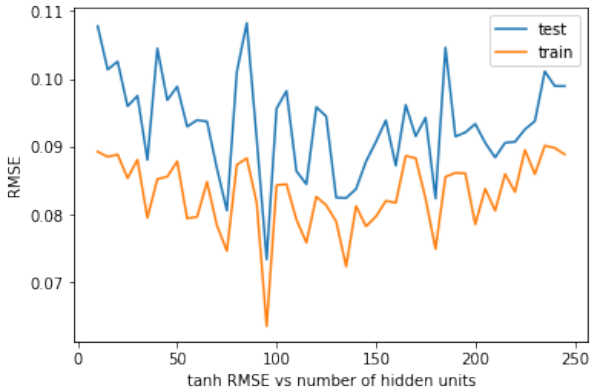


Figure 28: **Train and test RMSE vs number of hidden units using tanh. The best number of hidden units is 95 from 10 to 250.**

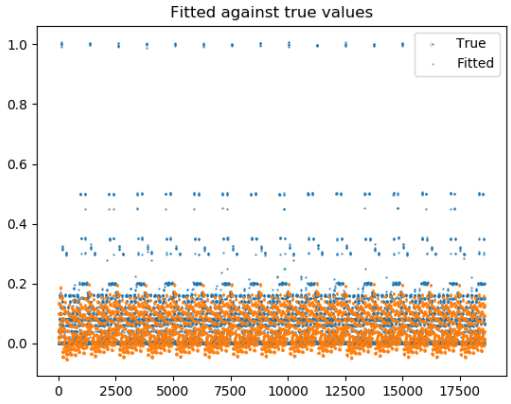


Figure 29: **Fitted values against true values using Neural Network regression model(tanh activation function) (with the optimal number of hidden units we find.)**

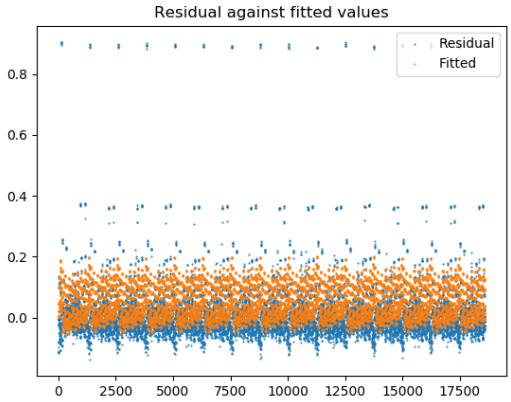
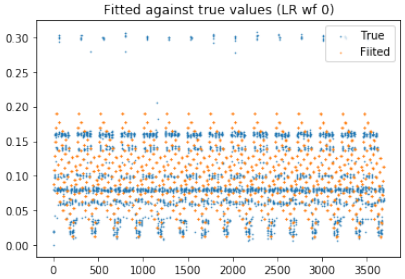


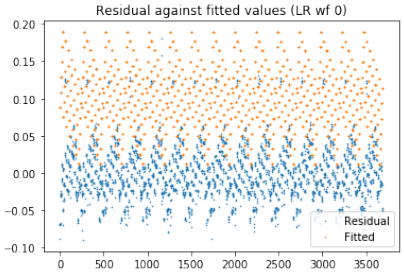
Figure 30: **Residuals vs fitted values using Neural Network regression model(tanh activation function) (with the optimal number of hidden units we find.)**

6 2d

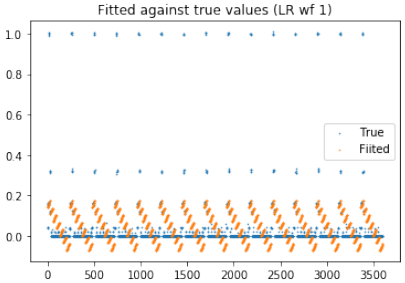
6.1 i



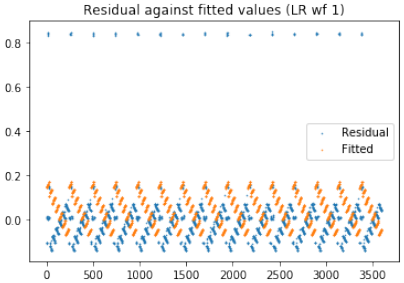
(a) Fitted against true (work flow 0)



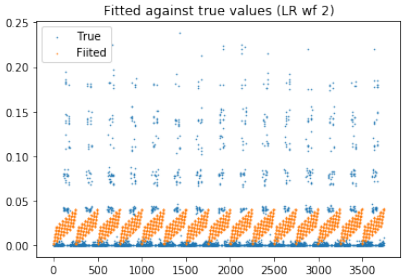
(b) Residual against fitted (work flow 0)



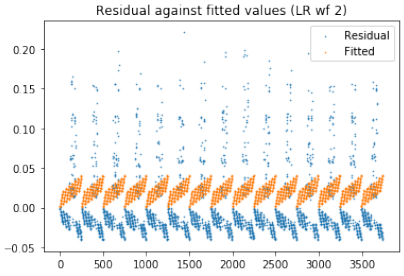
(c) Fitted against true (work flow 1)



(d) Residual against fitted (work flow 1)

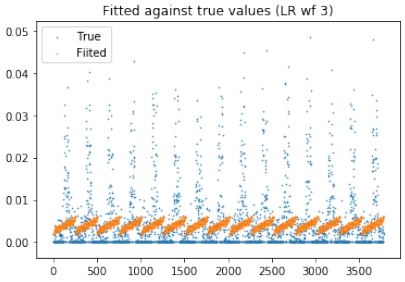


(e) Fitted against true (work flow 2)

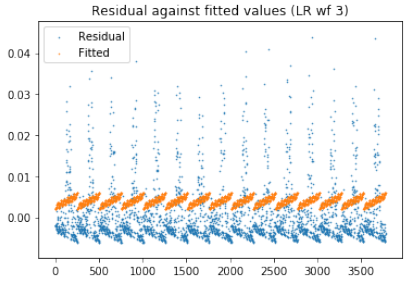


(f) Residual against fitted (work flow 2)

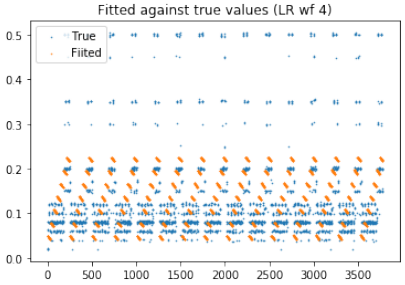
Figure 31: **Scatter plots for work flow 0, 1 and 2 (Linear Regression)**



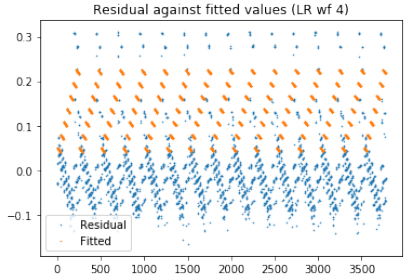
(a) Fitted against true (work flow 3)



(b) Residual against fitted (work flow 3)



(c) Fitted against true (work flow 4)



(d) Residual against fitted (work flow 4)

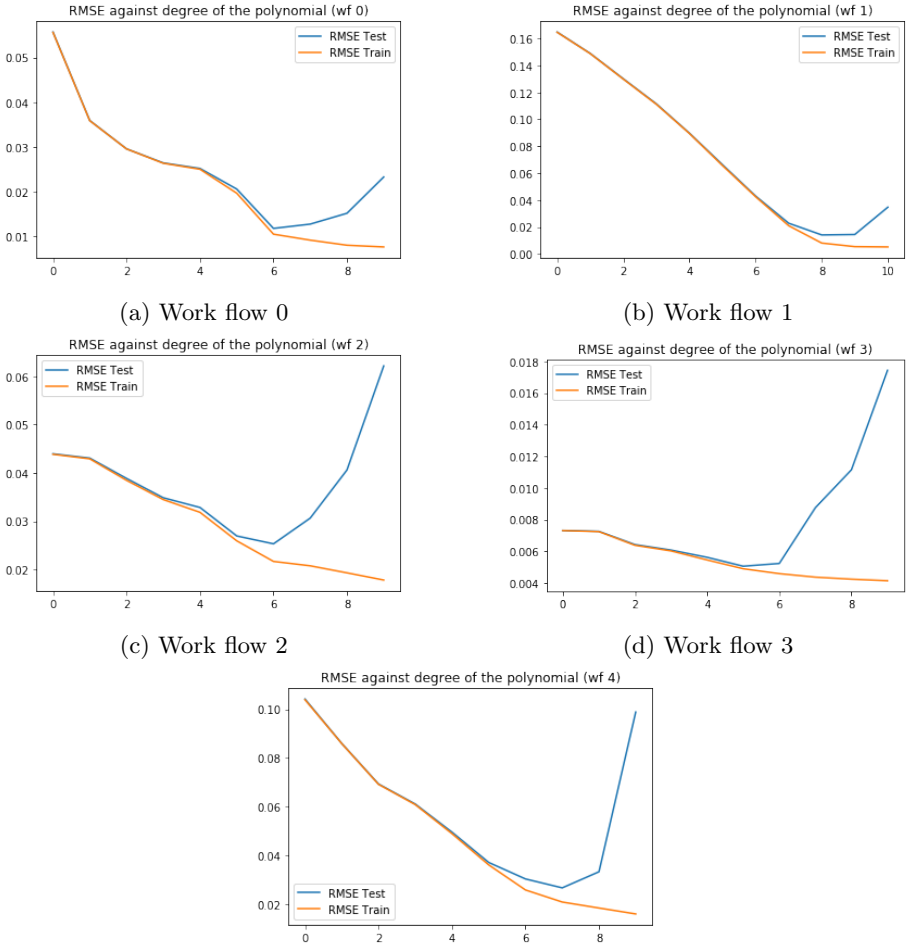
Figure 32: **Scatter plots for work flow 3 and 4 (Linear Regression)**

Work flow	0	1	2	3	4
Test RMSE	0.035888	0.148919	0.043065	0.007261	0.085991
Train RMSE	0.035836	0.148766	0.042909	0.007244	0.085922

Table 10: **Train and test RMSE for each of the workflows (Linear Regression)**

When predicting the backup size for all the workflows, the train RMSE is 0.103585 and the test RMSE is 0.103676. As we can see from the scatter plots and train/test RMSEs of each workflow, the fit is improved when predicting for each of the work flows separately, except for work flow 1 whose performance is decreased.

6.2 ii



(a) Work flow 0

(b) Work flow 1

(c) Work flow 2

(d) Work flow 3

(e) Work flow 4

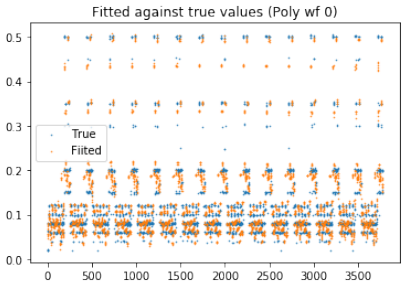
Figure 33: **RMSE against degree of the polynomial for each work flow**

As we can see from above plots, there are obvious thresholds beyond which the generalization error of our model gets worse. The threshold is 6 for work flow 0, 2 and 3, 8 for work flow 1 and 7 for work flow 4.

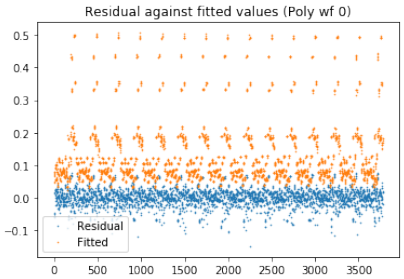
By applying cross validation methods, test errors will increase a lot when the model becomes too complex (although train RMSE keeps decreasing as model becomes more complex) ,and thus helps us identify an overfitting problem. If we do not apply cross validation methods, it would be hard to identify an overfitting problem and thus resulting in a model that has very low training error

but high generalization error.

Then, we apply above best degree of the polynomial to each work flow and obtain following scatter plots and RMSE table.

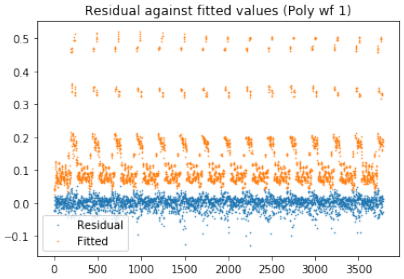


(a) Fitted against true (work flow 0)



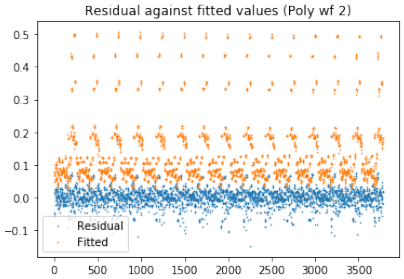
(b) Residual against fitted (work flow 0)

(c) Fitted against true (work flow 1)



(d) Residual against fitted (work flow 1)

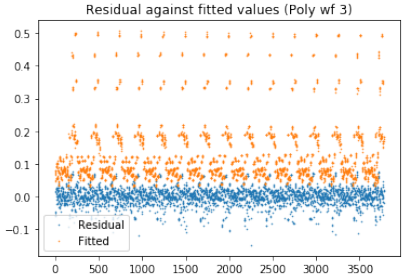
(e) Fitted against true (work flow 2)



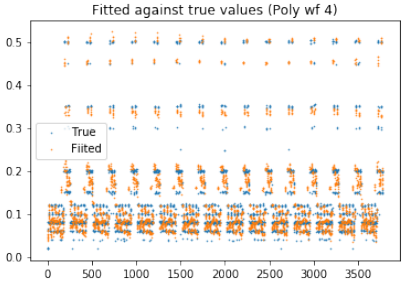
(f) Residual against fitted (work flow 2)

Figure 34: **Scatter plots for work flow 0, 1 and 2 (Polynomial)**

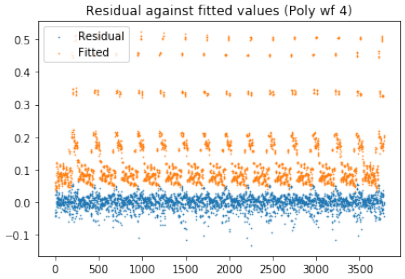
(a) Fitted against true (work flow 3)



(b) Residual against fitted (work flow 3)



(c) Fitted against true (work flow 4)



(d) Residual against fitted (work flow 4)

Figure 35: **Scatter plots for work flow 3 and 4 (Polynomial)**

Work flow	0	1	2	3	4
Test RMSE	0.03036	0.03330	0.03036	0.03036	0.02669
Train RMSE	0.02583	0.01842	0.02583	0.02583	0.02088

Table 11: **Train and test RMSE for each of the workflows (Linear Regression)**

When predicting back up size for each of the workflow separately by using polynomial functions, the best models demonstrate much better performances than linear regression.

7 2e

We use k-nearest neighbor regression for the data and find the best parameter.

We run KNN from n_neighbors = 1 to n_neighbors = 100. As the plot and code result show, lowest test RMSE = 0.03380836443745873, and it is when n_neighbors = 4. The train RMSE along with n_neighbors = 4 is 0.028017391631920175.

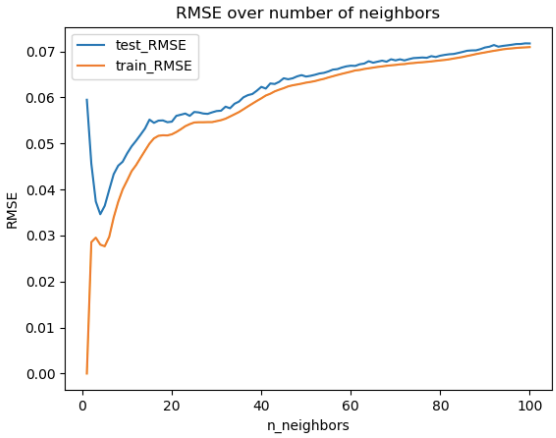


Figure 36: **Train and Test RMSE over number of neighbors for KNN**

Graphs below show fitted against true values over number of data points and residual against fitted values over number of data points. As fitted against true value graph for KNN shows, the predicted values match pretty

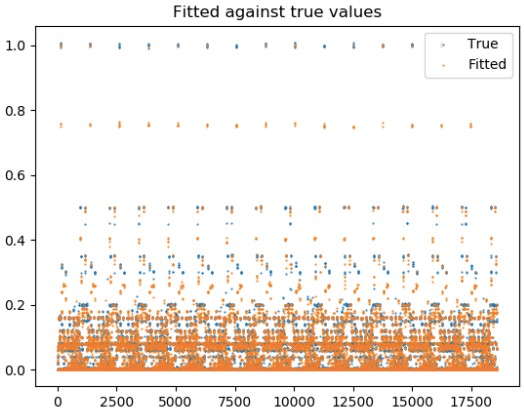


Figure 37: **Fitted against true value for KNN**

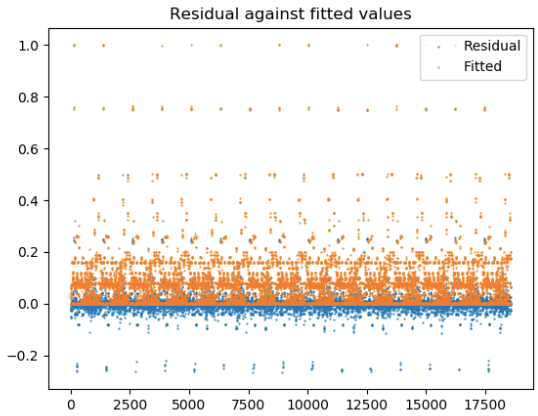


Figure 38: **Residual against fitted value for KNN**

8 Q3

In part 2(a), we trained and cross validated the linear regression model with scalar encoding. The result is poor RMSE = 0.103, indicating this is not a good combination. Then we fit the linear regression model with data from all 32 different combination encoding(in part 3.4), and the best test result has RMSE 0.088, which is still a poor performance. So we conclude linear regression is a poor model for this task.

We then trained a random forest model and tuned the hyperparameters(section 4.6). The best parameters: max features = 6, max number of tree = 23, max depth = 10. The test RMSE is 0.01369, which is the best model for scalar encoding.

The third model is neural network regression model with one hidden layer with three different activation functions. The data are sparse(all features one-hot encoded). The best combination is 'relu' activation functions with 240 hidden units(section 5), and the test RMSE is 0.029. Therefore, this combination is the best model we found so far for processing sparse data.

Then, we apply both linear regression and polynomial regression models to predict back up size for each of the workflows separately. When workflows were predicted separately, the average testing RMSE were largely decreased (i.e. performance largely increased).

The final model is k-nearest neighbor regression. The best KNN model has test RMSE of 0.0338.

The nueral network regression model is better at taking care of sparse features, whereas the polynomial regressions model is better at predicting for each workflow separately (categorical features).

The random forest regression model is good at handling categorical features.

The random forest regression model overall generates the best results.

9 Conclusion

To conclude, we explore regression analysis in linear regression, random forest regression, neural network regression, and KNN regression. We predict the backup size of the file using week index, day of the week, backup start time, workflow ID, and file name. We experiment with different 32 combinations of one hot encoding and found out that certain combination provides lowest RMSE. This is because some variables will be better encoded as one hot encoding by the nature of the dataset. Moreover, we experimented with regularization to control overfitting. Lastly, we compare the performance of different regressions.