UCLA ECE219 - Project 4

March 6th, 2019

Authors

Ning An - 205034447
Zaurbek Tsorojev - 805029443
Endi Xu - 005030030
Minya Yao - 704161217

Introduction

The goal of this project was to learn about regression analysis. We worked on the Network Backup Dataset and applied several regression models to estimate the relationship between a target variable and a set of potentially relevant variables. Cross-validation was used to handle over-fitting, and regularization was used to penalize overly complex models.

Dataset 1

Question 1 (a)

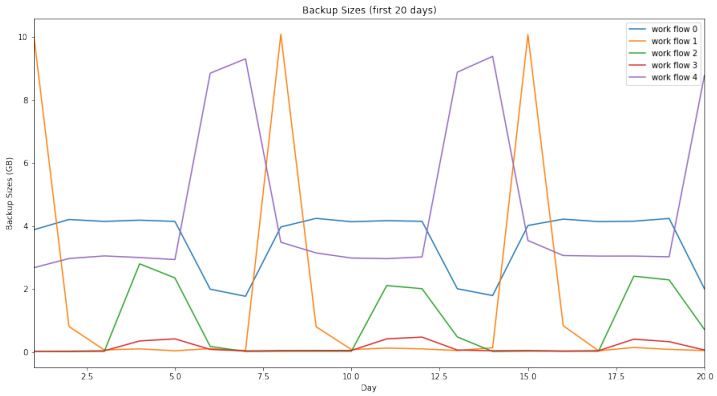


FIGURE 1 – Backup sizes for the first 20-day period

Question 1 (b)

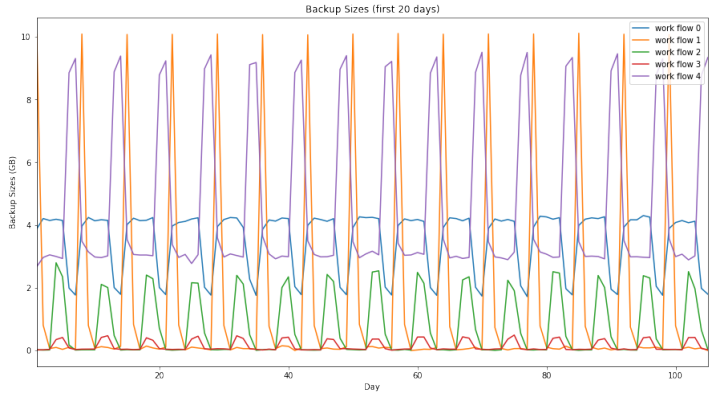


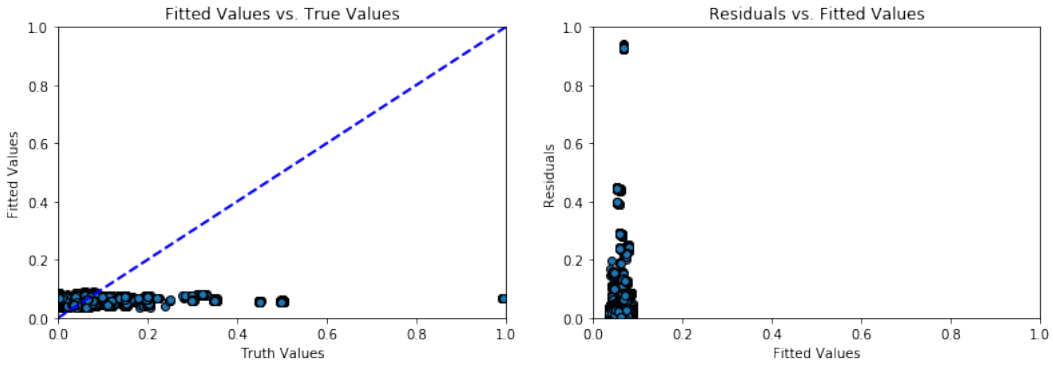
FIGURE 2 – Backup sizes for the first 105-day period

Question 1 (c)

The workflows in both 20-day and 105-day plots are periodic with a period for each workflow to be about 7 days. Workflow_1 always have the largest backup size, followed by workflow_4, workflow_0, workflow_2, and workflow_3.

Question 2 (a)

For this part, we fit a linear regression model to the dataset, and plot fitted values vs. true values and residuals vs. fitted values as scatter plots to visualize how well the model fits the dataset.



Training RMSE	Test RMSE
0.1032	0.1067
0.1039	0.1001
0.1032	0.1068
0.1039	0.1004
0.1032	0.1071
0.1039	0.1004
0.1032	0.1071
0.1039	0.1004
0.1032	0.1071
0.1039	0.0999

Average training RMSE : 0.10358
Average test RMSE : 0.10367

From the plots, we observe that the predicted value using the basic linear regression model centered at around 0.061 and the residuals centered at around 0.064. This implies that our model does not predict very well, as ideally we would want the points from the first plot to lie roughly at the line y = x, and the points (residuals) from the second plot to be as close to 0 as possible.

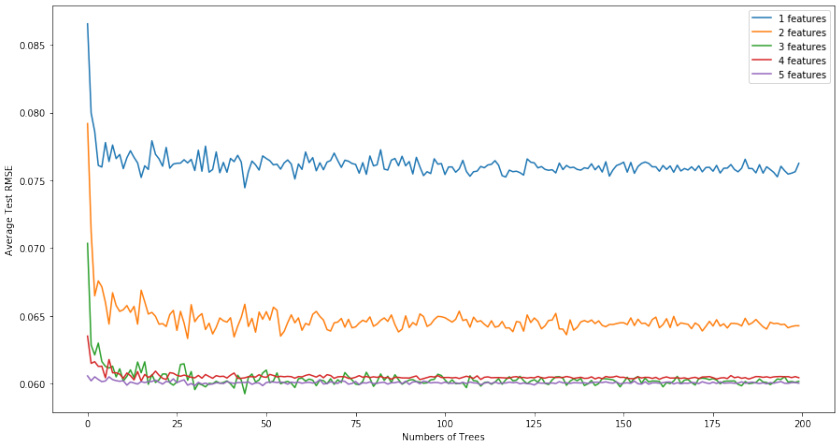
Question 2 (b)

For this part, we fit a random forest regression model to the dataset. We report the training and average test RMSE from 10 fold cross validation and the Out of Bag errors for the initial setting (using number of trees = 20, depth of each tree = 4, and maximum number of features = 5) below.

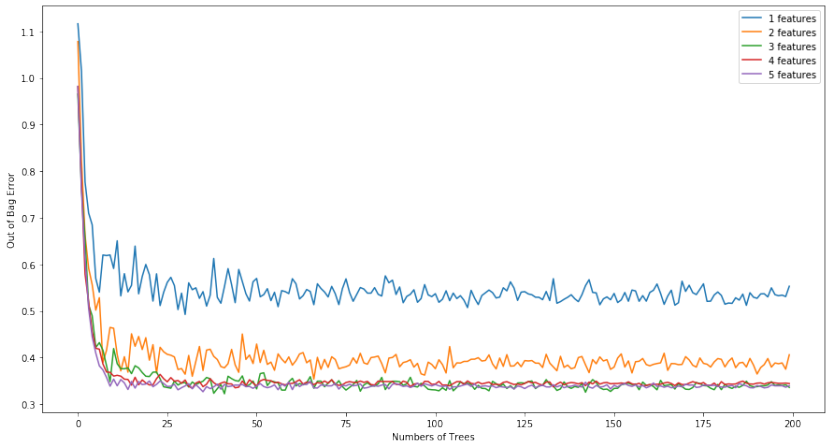
Training RMSE	Test RMSE
0.0601	0.06762
0.0611	0.05276
0.06015	0.06748
0.0602	0.05221
0.0601	0.06745
0.06096	0.05395
0.06012	0.06774
0.05936	0.05127
0.06002	0.06722
0.06077	0.05275

Average training RMSE : 0.060304
Average test RMSE : 0.060514
Out of bag error : 0.340681

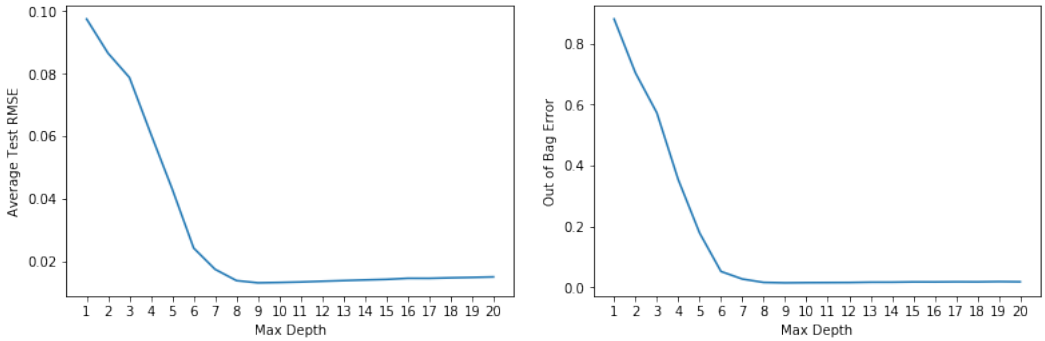
Next, we try to obtain the best combination of the hyperparameters for our model. We sweep number of trees from 1 to 200 and maximum number of features from 1 to 5, and plot the average test RMSE and Out of Bag error vs. number of trees using each of the five features.



From the above plots, we observe that 4 and 5 features would yield the smallest average test RMSE and out of bag error. Furthermore, for out of bag errors, using 4 features would give a less fluctuated out of bag error compared to using 5 features. Additionally, for both average test RMSE and out of bag errors, the values decrease to steady state errors when the number of trees takes values greater than or equal to 20.

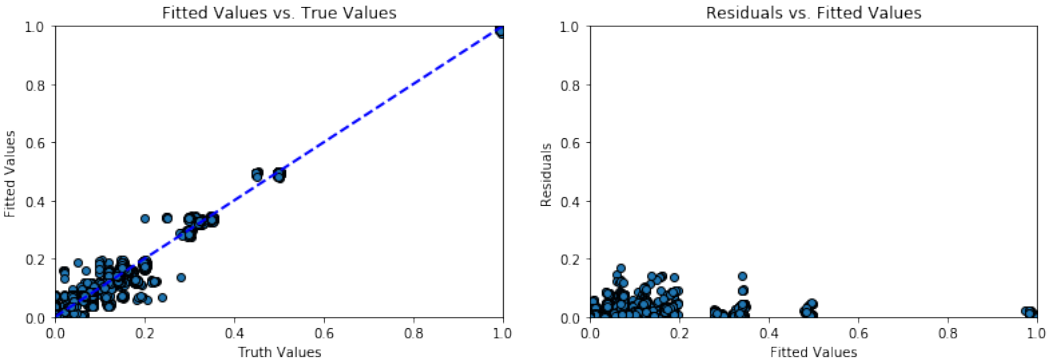


We also experiment on the maximum depth of each tree, for values from 1 to 20, while keeping the number of trees fixed at 25 and number of features fixed at 4. We plot the average test RMSE and Out of Bag error vs. maximum depth below.



From the plots, we observe that the average test RMSE and Out of Bag error decrease to their steady state values for max depths greater than or equal to 8. Therefore, the parameters that achieves the best performance is tree number = 25, number of features = 4, max depth = 8.

Using the above parameters, we yield the following scatter plots :



From the plots, we observe that the predicted value using the random forest regression model predicts better than the basic linear regression model, because the fitted values vs. true values lie rougly at the line y = x. We now compute the average test RMSE and Out of Bag error for this setting and report the results below.

Training RMSE	Test RMSE
0.01297	0.01493
0.01240	0.01415
0.01347	0.01204
0.01248	0.01512
0.01277	0.01168
0.01228	0.01518
0.01304	0.01282
0.01270	0.01381
0.01308	0.01205
0.01274	0.01407

Average training RMSE : 0.0128002
Average test RMSE : 0.0136481
Out of bag error : 0.0177135

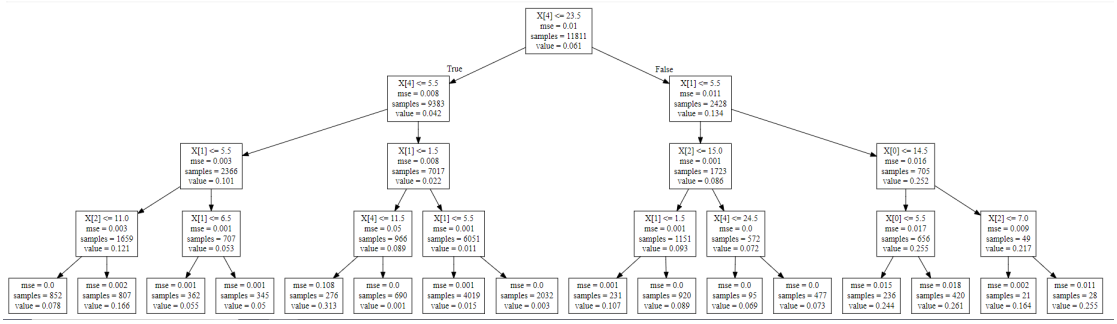
Compared to the average test RMSE and Out of Bag error we obtained from the initial setting (using number of trees = 20, depth of each tree = 4, and maximum number of features = 5), we see that the average test RMSE is improved by $\frac{(0.0605-0.0136)}{0.0605} = 77.52\%$, and the Out of Bag error is improved by $\frac{(0.3407-0.0177)}{0.3407} = 0.9480\%$.

Next, we obtain the order of feature importance among the five features, namely, day of the week, hour of the day, work-flow-ID, file name, and week number. We obtain the following results :

```
Feature 2 Importance = 0.38933001041675014
Feature 1 Importance = 0.23359502384072703
Feature 4 Importance = 0.2084943295705703
Feature 3 Importance = 0.16679964892361154
Feature 0 Importance = 0.0017809872483410417
```

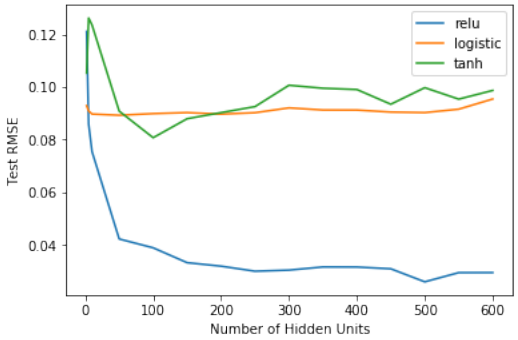
Based on the importance ranks, we see that the most important features are hour of the day, day of the week, file name, work-flow-ID, and week number, with descending order of feature importance.
Finally, we visualize our decision tree using graphviz.

The root node in the decision tree corresponds to file name, which is not the most important feature we found in the previous part. This happens because the decision tree we realized in this part is an instance of many trees that are obtained using the random forest algorithm, in which the algorithm uses the many trees and votes on the most important feature based on these results.



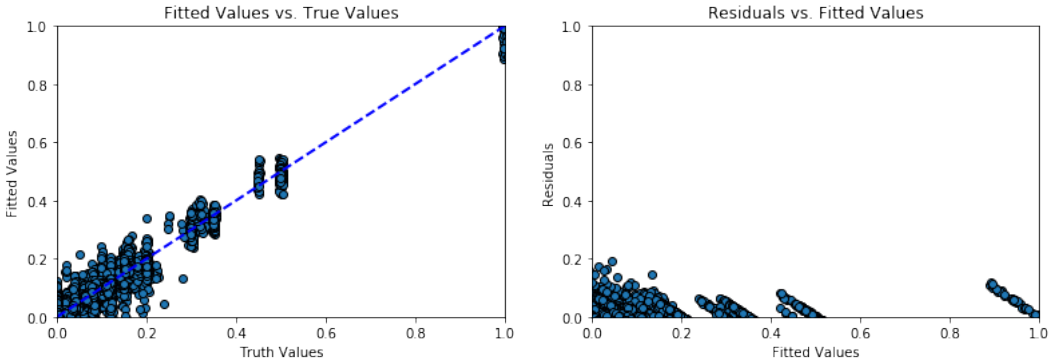
Question 2 (c)

For this part, we fit a neural network regression model with one hidden layer to the dataset. We sweep the number of hidden units from values [2, 5, 10, 50, 100, 150, 200, 250, 300, 350, 400, 450, 500, 550, 600], and plot the test RMSE vs. number of hidden units used for the hidden layer in our neural network, using the activation functions ReLU, logistic, and tanh.



We observe from the plot that using 500 hidden units and the ReLU activation function, the regression model yields the lowest test RMSE.

Now, we use this setting to fit the dataset and plot fitted values vs. true values and residuals vs. fitted values as scatter plots to visualize how well the model fits the dataset.



Training RMSE	Test RMSE
0.01496	0.03016
0.01363	0.02098
0.01579	0.02752
0.01371	0.04117
0.01562	0.02407
0.01405	0.03481
0.01502	0.03077
0.01411	0.02742
0.01472	0.03099
0.01546	0.02327

Average training RMSE : 0.0147298
Average test RMSE : 0.0296554

From the plots, we observe that although the points from the first plot center roughly around the line y=x, there are many points that do not lie on the line. From the second plot, the residuals or errors of our predicted values range from 0 to a little less than 1. We conclude that the neural network regression model works better than the basic linear regression model, but worse than the random forest regression model, as ideally we would want the points from the first plot to lie roughly at the line y = x, and the points (residuals) from the second plot to be as close to 0 as possible.

The table below shows the performance summary :

	Average Training RMSE	**Average Test RMSE**
Linear Regression	0.1036	0.1037
Random Forest Regression (number of trees = 25, number of features = 4, max depth = 8)	0.0128	0.0136
Neural Network Regression (number of hidden units = 500, activation function = ReLU)	0.0147	0.0297

Question 2 (d)

i - Linear Regression

We use a linear regression model to predict the backup size, but this time, we do it on each of the workflows separately.

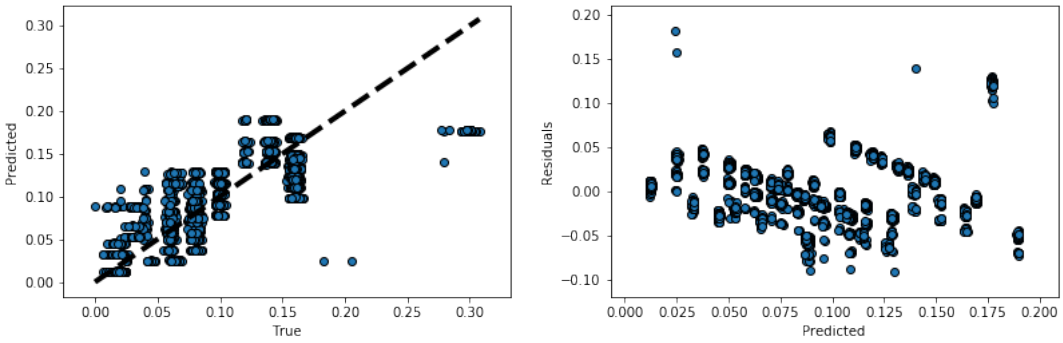


FIGURE 3 – Workflow 0

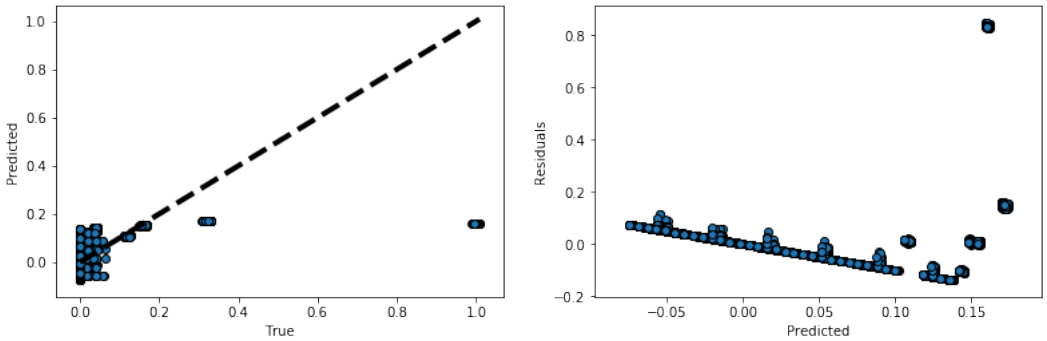


FIGURE 4 – Workflow 1

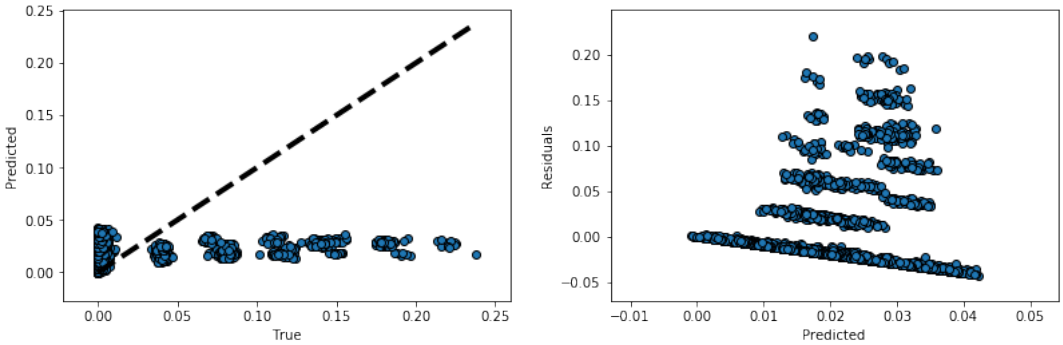


FIGURE 5 – Workflow 2

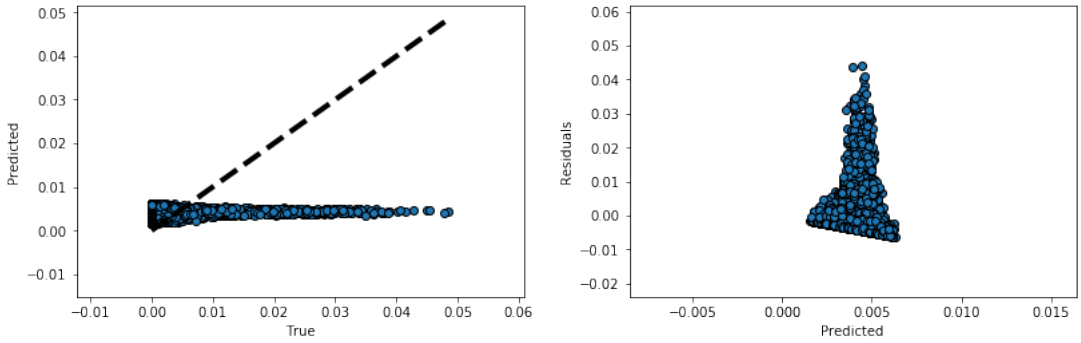


FIGURE 6 – Workflow 3

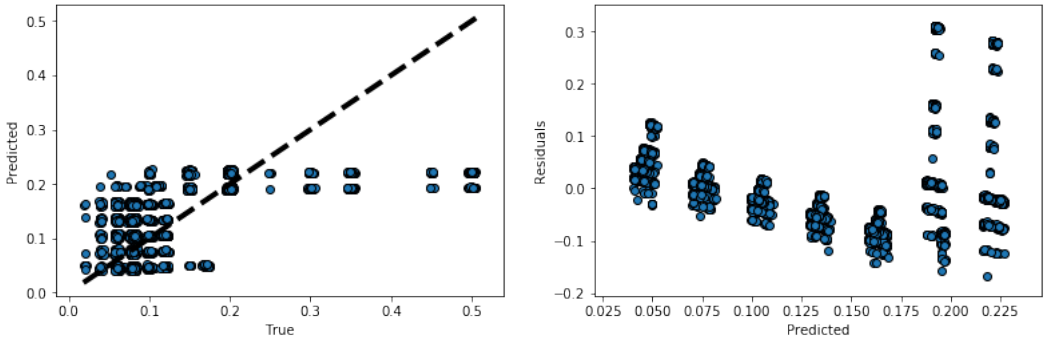


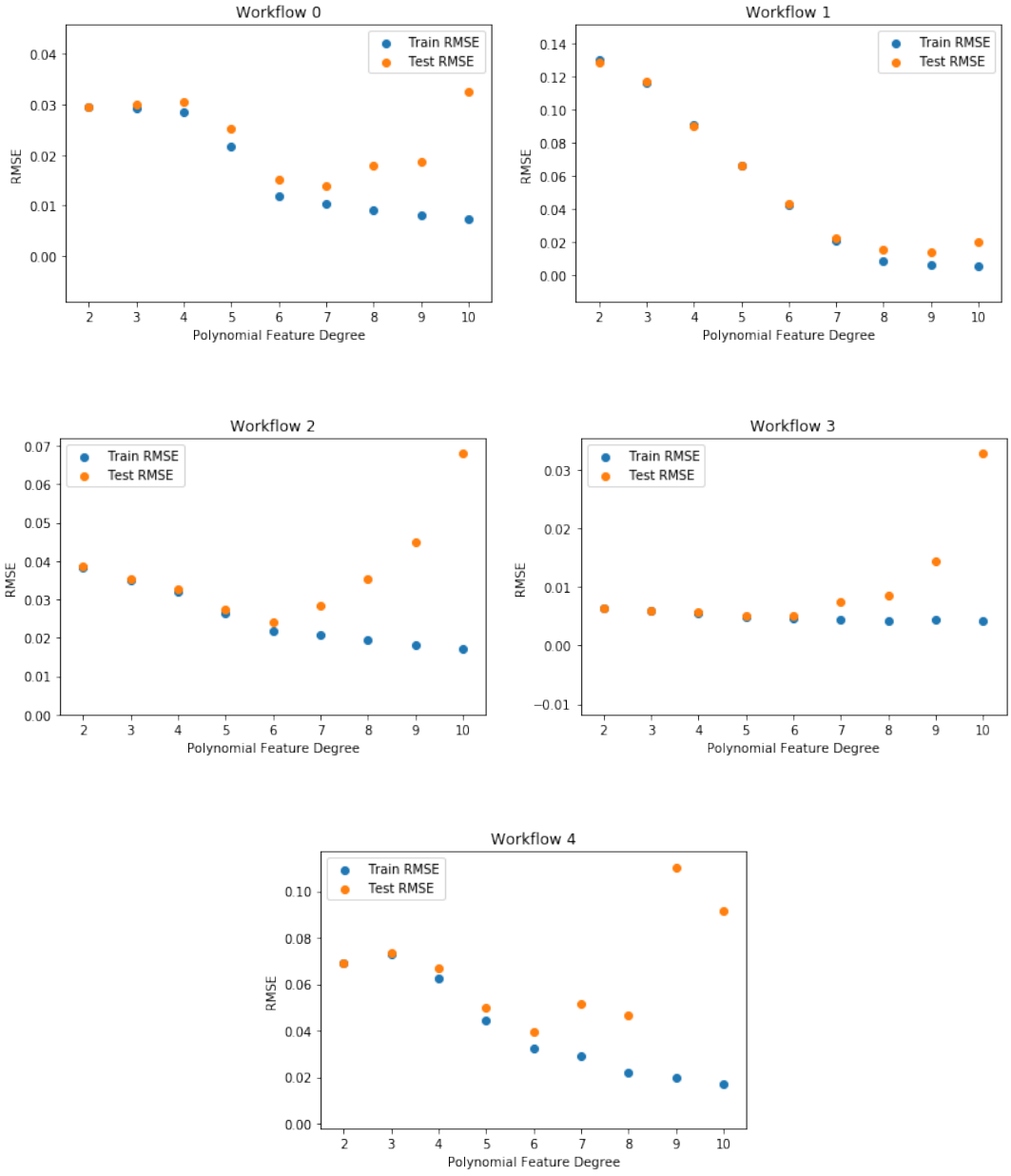
FIGURE 7 – Workflow 4

	Average Train RMSE	Average Test RMSE
Workflow 0	0.03583	0.03586
Workflow 1	0.14874	0.14709
Workflow 2	0.04290	0.04257
Workflow 3	0.00724	0.00719
Workflow 4	0.08591	0.08534

Except for workflow 1, overall we seem to get better results than in point 2(a). Using a linear regression model on each workflow separetely seems to yield lower RMSE.

ii - Polynomial Function

A polynomial function can have more degrees of freedom, which can lead to a better fit of the data. We plot the average train and test RMSE of the trained model against the degree of the polynomial for each workflow.



We notice that overall after the threshold degree of 7, the generalization error of the model gets worse. This is probably due to overfitting as we keep increase the degree to force the polynomial model to fit the data.

Cross validation allows us to detect overfitting when there is a large difference between the RMSE of the train and test dataset. It thus allows us to improve the model without overfitting.

For the optimal degree of 7, we plot on figure 8 the fitted values against true values, as well as the residuals versus fitted values. We get the following error values :

Average training RMSE : 0.0405979
Average test RMSE : 0.0411963

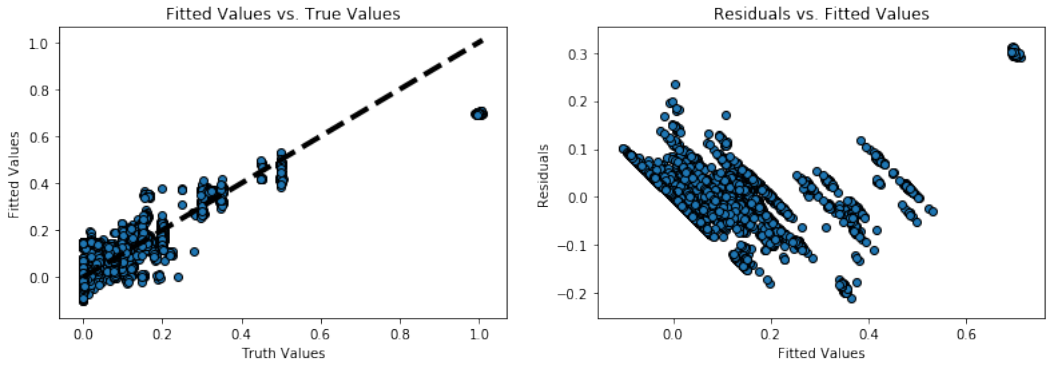


FIGURE 8 – Scatter plots

Question 2 (e)

Finally, we use the k-nearest neighbor regression. To find the best parameter, we plot the average train and test RMSE against the number of neighbors k. The graph is represented on figure 9.

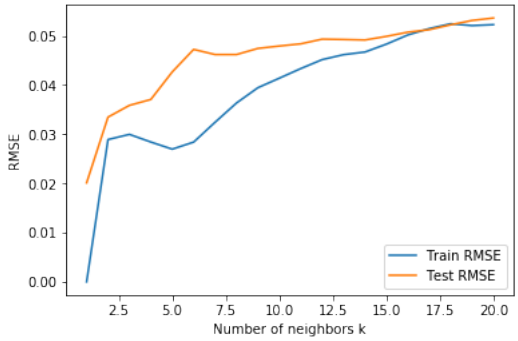


FIGURE 9 – RMSE vs number of neighbors

The best RMSE is obtained when k = 2. We get the following error values :

Average training RMSE : 0.028907
Average test RMSE : 0.033467

The plots of the fitted values against true values, as well as the residuals versus fitted values are represented on figure 10.

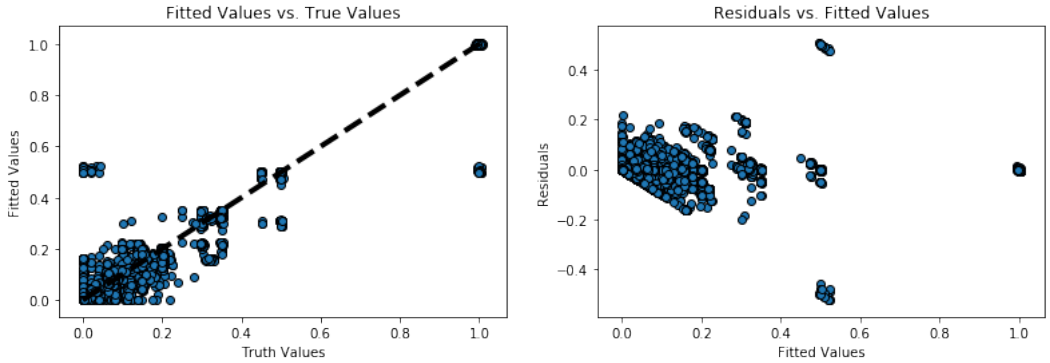


FIGURE 10 – Scatter plots

Question 3

The best overall performance is obtained with the Random Forest Regression model with a max_depth of 8. It has the lowest RMSE value of all : 0.0128

1. Linear Regression : produced the worst results (average RMSE of 0.1036). However, when we used a linear regression model on each workflow separetely, we got much better results for all workflows except workflow 1 for which the average RMSE didn't improve. This regression is the simplest. It has the advantage to be the fastest.

2. Random Forest Regression Model : produced the best results (average RMSE of 0.0128). This method is best for handling categorical features. On top of that, it can evaluate the importance of each feature and calculate the error rate of the algorithm using out of bag data.

3. Neural Network Regression Model : good at handling sparse features as it was mainly dealing with the sparse matrix with all features one-hot encoded. This model works better than the basic linear regression model, but worse than the random forest regression model.

4. Polynomial Function Regression Model : better than linear regression to predict the backup size for the workflows. It works well with numerical features. We need to make sure to select the right degree as threshold to avoid overfitting.

5. K-Nearest Neighbor Regression Model : it provides decent results once we select the right k, even comparable to the best results obtained with polynomial regression. It is not good to handle sparse features.