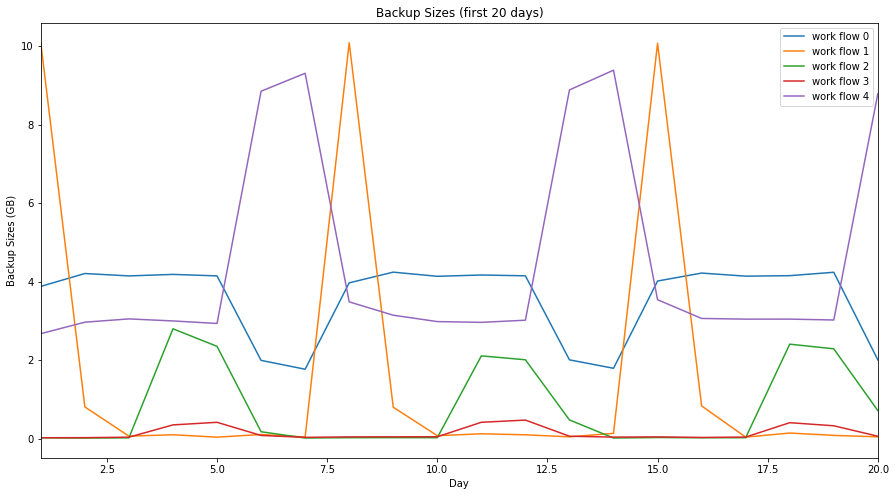
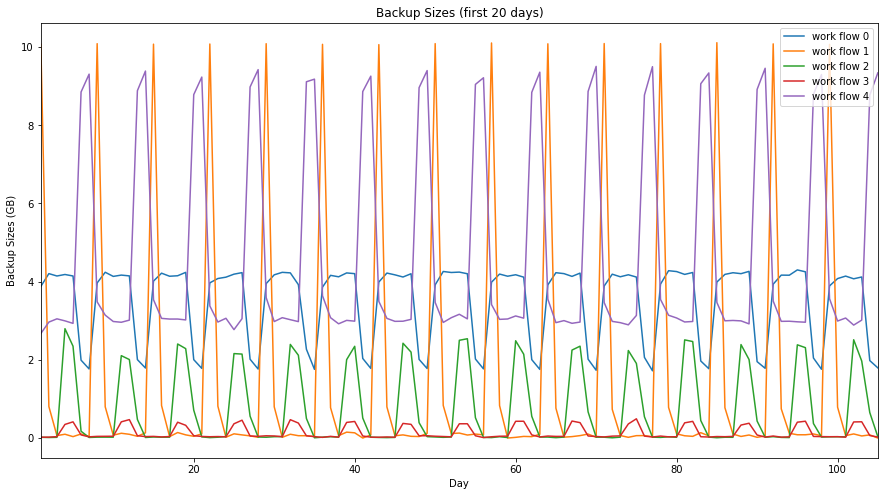
**Dataset 1**

**1 (a)**



**1 (b)**

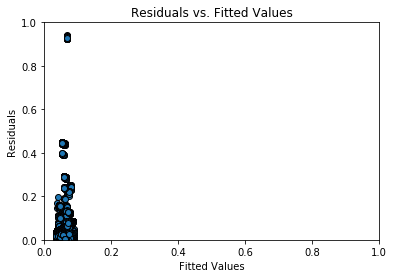
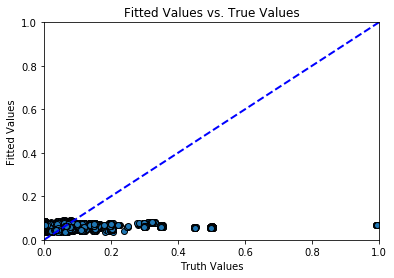


**1 (c)**

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.

**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: [0.10324315757540427, 0.10396677791654478, 0.10322579905879994, 0.1039464291963174, 0.10319511299798963, 0.10393838251390547, 0.10320263162336227, 0.10393638595896813, 0.10320098997274793, 0.10399160058862011]

Test RMSE: [0.10671805207226825, 0.10018461438631411, 0.106849773893365, 0.10036709167781363, 0.10711585431838347, 0.10044533654683797, 0.10705026882029617, 0.1004666455097133, 0.10707418586815101, 0.09994712086112156]

Average training RMSE: 0.10358539364277801

Average test RMSE: 0.1036758476759903

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.

**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: [0.060125191073332696, 0.06110392544040279, 0.060158586514289715, 0.0602406142474856, 0.06014710842758177, 0.06096165657626343, 0.060120692975535565, 0.05936499728108565, 0.060028754199416064, 0.060771666775195354]

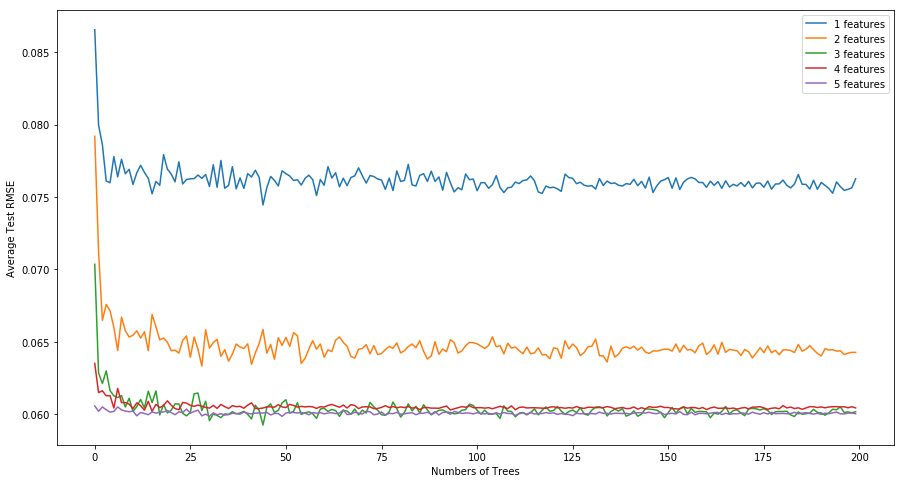
Test RMSE: [0.06762686196824738, 0.052765728240840803, 0.06748062002350932, 0.05221889388750287, 0.06745128450507824, 0.05395264408239399, 0.06774005732500292, 0.051273909032480455, 0.0672297812038687, 0.052754133692846025]

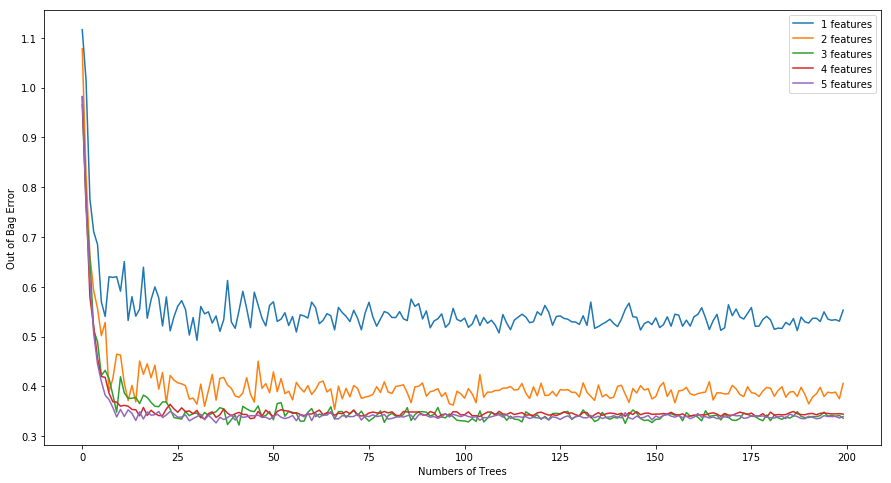
Average training RMSE: 0.060304279372425544

Average test RMSE: 0.060513797017857966

Out of bag error: 0.34068005991710215

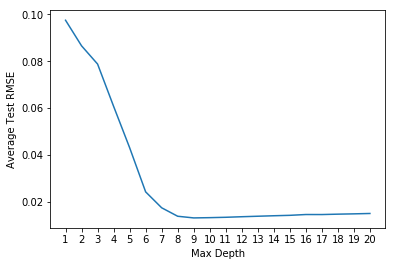
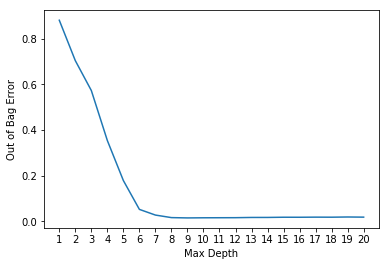
Next, we try to obtain the best combination of the hyperparameters for our model. We sweep number of trees from 1 to 200 and maximum number of features from 1 to 5, and plot the average test RMSE and Out of Bag error vs. number of trees using each of the five features.





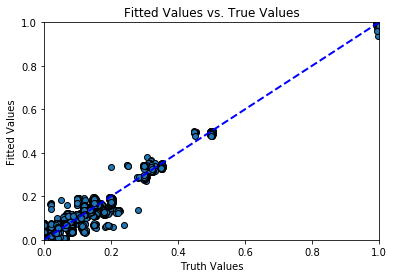
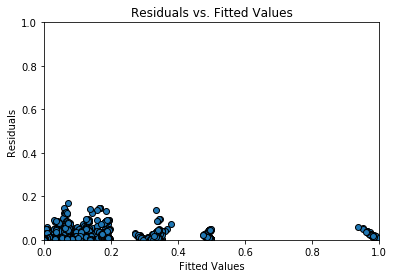
From the above plots, we observe that 4 and 5 features would yield the smallest average test RMSE and out of bag error. Furthermore, for out of bag errors, using 4 features would give a less fluctuated out of bag error compared to using 5 features. Additionally, for both average test RMSE and out of 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: [0.01297769875433942, 0.012402136472288279, 0.013470502215884412, 0.012482127819047745, 0.012774162798932641, 0.012281469601279413, 0.013041059325531368, 0.0127027571590768, 0.013083246597064785, 0.012741397850654061]

Test RMSE: [0.014930080919357559, 0.014151709019461606, 0.012049665849944461, 0.015120655339493977, 0.011680293741677323, 0.01518993591766826, 0.012820629546908682, 0.013811504215742806, 0.01205258414631431, 0.014075596077955957]

Average training RMSE: 0.012800192022537741

Average test RMSE: 0.013648016869604906

Out of bag error: 0.017713496896837655

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 , and the Out of Bag error is improved by .

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

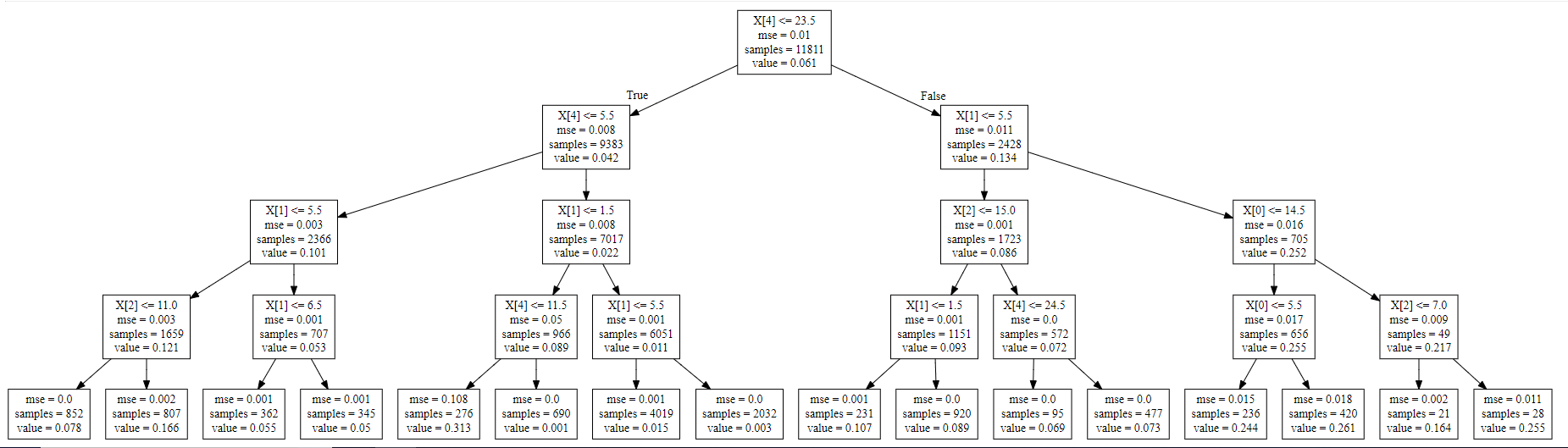
Feature 4 Importance = 0.2084943295705703

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.

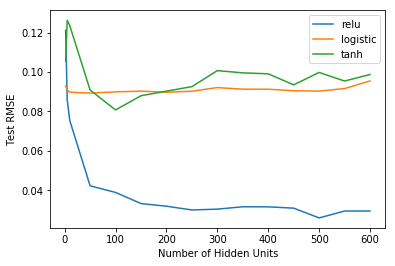
Finally, we visualize our decision tree using graphviz.



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.

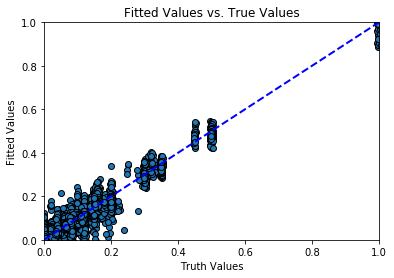
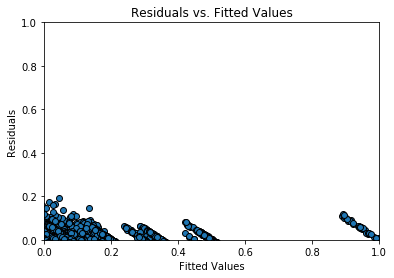
**2 (c)**

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: [0.014964935923638515, 0.013632019390229469, 0.01579240527404731, 0.013710585903069738, 0.015620649061258294, 0.01405486011316085, 0.015018761099988398, 0.014119997699301553, 0.014724413742444289, 0.015466858427300572]

Test RMSE: [0.030161042672223446, 0.02098999765600241, 0.027522310858475094, 0.0411714432122026, 0.02406834955709214, 0.0348112040033641, 0.03077249981695051, 0.027415445382418642, 0.030990851978511688, 0.023270115480557862]

Average training RMSE: 0.014729816188841486

Average test RMSE: 0.029655393731562826

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 are 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.

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 |