Jack Dumanski, Ryan O'Neill

Dr. Perry

STAT 351 001

6 December 2025

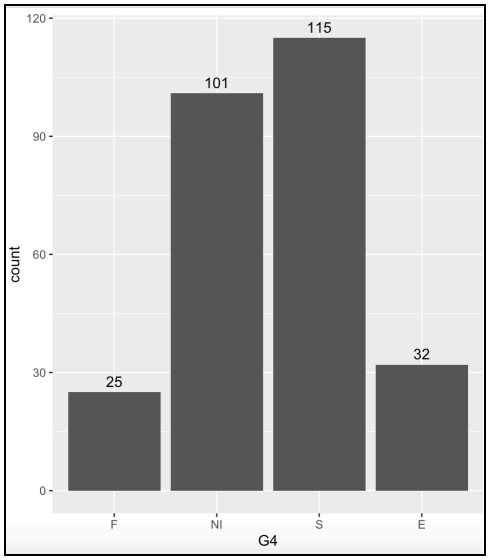
Math Performance Final Paper

Introduction

In this project, we attempted to predict math students' grades based on several predictors on their schooling and personal living situations. None of the predictors were continuous variables, and the response variable, labeled "G4," was a categorical variable with 4 levels ranging from "Fail" to "Needs Improvement" to "Satisfactory" to "Excellent." Our goal was to create a tree model that most accurately predicted this variable for the students, and we utilized several methods to make a model with maximized predictive ability. After we came to a conclusion that our model was sufficiently accurate, we recorded our results for predicting the MathScorestest.csv dataset, which was missing the G4 variable.

Data

Looking at the distribution of our response variable G4, most of the students fall into the middle grade outcomes of "Needs Improvement" and "Satisfactory" with much fewer in the "Fail" and "Excellent" outcomes. This suggests that it may be more difficult for a classification tree to predict the "F" and "E" outcomes because the sample size for those values are much smaller. The response variable having four possible outcomes means we might expect a lower accuracy for our model than if we had just two possible outcomes to predict from.



Distribution of the response variable of the dataset G4

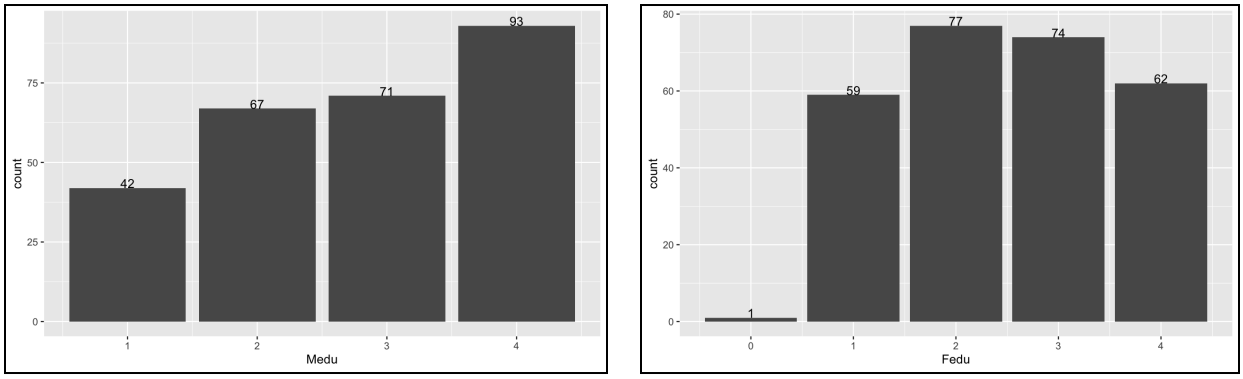
Many of the explanatory variables in the dataset are binary with outcomes of either "yes" or "no". These variables were treated as factors so that R knew that they were binary categorical variables. Other variables were ordinal on a scale from 1 to 5 such as freetime, Walc, or health. We did not treat these variables as factors because the values are ordered from very low to very high. Treating these as numerical variables allows R to split for either greater than or less than certain values rather than treating them as separate categories without order.

Some variables such as Mjob, Fjob, reason, and guardian were categorical variables that had more than 2 possible outcomes. This meant that each outcome of the variable might have a small sample size in each grade outcome of our response variable. For example, using the variable Fjob, in the table below we can see that there are only 12 students whose father's job is in health, and none of those students had a failing grade. This might make it difficult to use this variable as a predictor because there is a small sample size to work with in each outcome. We will try to create models with and without the variables Mjob and Fjob and compare which models have the better accuracy.

	at_home	health	other	services	teacher
F	3	0	14	7	1
NI	5	7	58	28	3
S	4	3	65	36	7
E	2	2	12	9	7

Table of G4 grade outcomes vs Fjob categories

Finally, the variables Medu and Fedu, representing the highest level of education of the students' mother and father, had numerical values but were treated as factors. Each numerical value represented an education level such as "secondary education" or "higher education". Since these values represented categorical outcomes, we thought it was best to treat these as categorical variables.



Distribution of predictor variables Medu and Fedu

Methods

Before attempting to create a tree model, we split the dataset into a training set and testing set. The training dataset had 191 observations which was about 70% of the total observations, and the testing dataset had 82. We used the training dataset to create a model that would be able to predict the outcome of the response variable G4 on the unseen test data. We created a tree out of this dataset using all predictors except for ID, which was a meaningless predictor for our purpose. We then attempted to cross-validate the tree using the cv() tree function, but our results were unexpected. The cv-SSE plot showed a constant increase in SSE,

showing the optimal splits to be around 2 nodes, which indicated to us that a normal tree model with all predictors may not be effective. We then attempted to prune the tree and note the accuracy with many combinations of nodes.

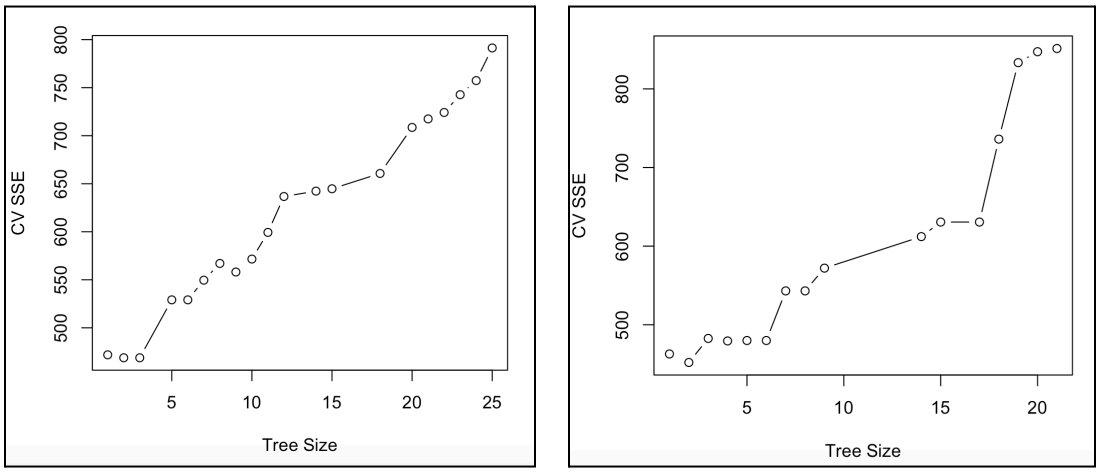
After experimenting with a simple tree, we moved on to a random forest to try to improve our accuracy. The random forest takes several trees with predictors selected at random and allows the most important predictors to be most prevalent in the resulting forest. This way, we didn't have to guess which predictors to include or exclude, the random forest's algorithm displayed it for us. For final checks to gauge the effectiveness of this forest model, we edited the "mtry" and "ntree" arguments of the random forest function which allow us to set the number of variables sampled and trees aggregated.

Results

Initially, with the simple tree model that contained all possible predictor variables except ID, we received a model that used 15 of the variables for splits in the tree and had 25 terminal nodes. It had a misclassification rate of .2356, which means an accuracy of .7644. Testing the tree on the testing data, we found it had a much lower accuracy of .3902. The low accuracy and large size of the tree suggests that the model was overfitting based on the training data and did not do as good of a job on new data that it had not seen. Using the same method after removing the variables Mjob and Fjob, we received training and testing accuracies of .6806 and .4390. We removed these variables due to their type of variable, where most of the other predictors were either binary or meaningfully factored, but we suggested that these were not. This tree did significantly worse at predicting G4 for the training data, but over 4% better on the testing data, which shows that since the training and testing data were closer together, the predictive ability

excluding these variables may have been slightly better. The accuracy, however, was still lower than the trees produced by the other methods.

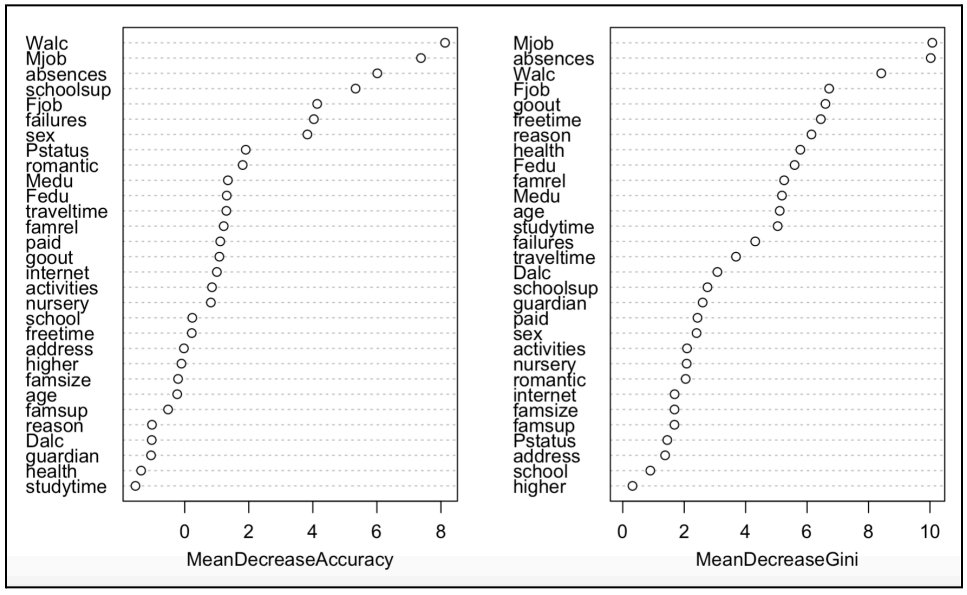
Next, we checked the results of the pruned tree based on the cross-validation method. Experimenting with the tree sizes, we found the best tree to have a size 12 terminal nodes. Selecting an optimal size greater than 12 slightly increased the training accuracy but did not help the testing accuracy, and lowering the size only hurt both the training and testing accuracy. The optimal 12 node tree had a training accuracy of .6911 and testing accuracy of .4634. After eliminating Mjob and Fjob, we found the best tree had a size of 8 terminal nodes with a .5916 training accuracy and .5122 testing accuracy. This size of tree had the closest accuracies between the training and testing. This was also the highest testing accuracy we had found so far, but the training accuracy had decreased substantially.



Plot of deviation by tree size. Left: including all variables, Right: excluding Mjob, Fjob

Finally, we determined that the random forest produced our best possible tree model for predicting G4. Every time the random forest process is run, the results will be slightly different due to which variables are randomly selected for each of the trees in the forest. Using this method the training accuracy always came out to equal a perfect 1.00. The testing accuracy varied, but the highest we were able to obtain was .5367, slightly higher than the pruned tree

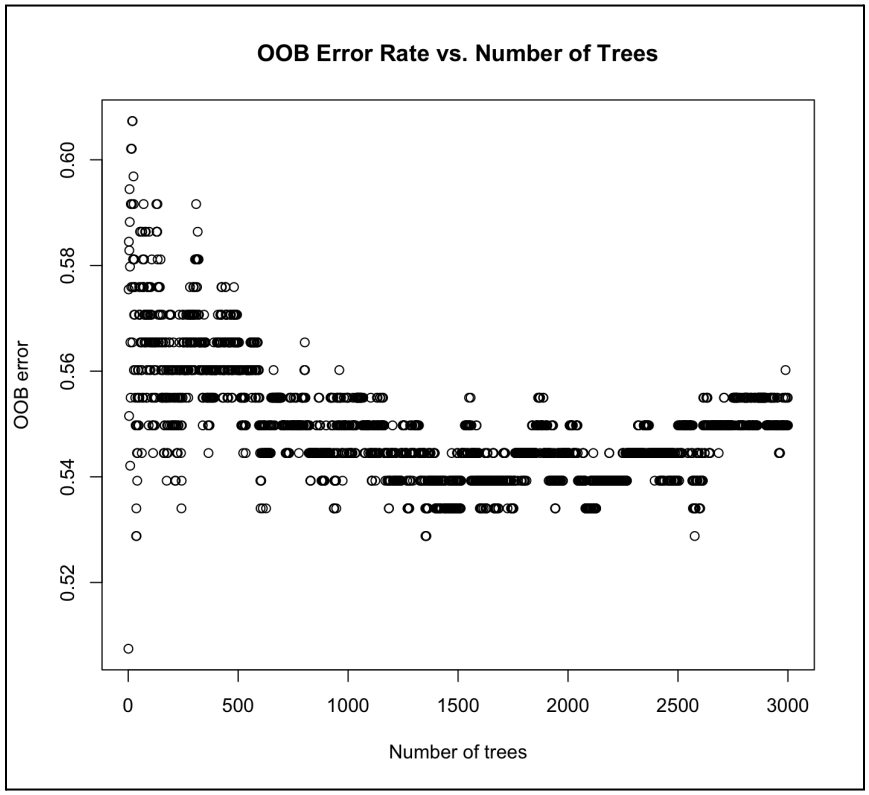
from the cross validation method. Unlike the previous methods, the random forest method did not have a very noticeable change in the test accuracy depending on if we included the variables Mjob and Fjob or not. This could be because the random forest automatically removes variables that are not significant in the classification tree. For this reason, it makes sense to leave all possible variables in the set for the random forest to select from because if the variable is not significant, it will simply not be used. We can then make a variable importance plot to analyze which variables were the most significant in all of the trees sampled in the random forest. The plot showed that variables such as Walc, Mjob, absences, and schoolsup were the most important variables used in the process.



Variance Importance Plot based on the random forest method

Our final check with the random forest model was to modify the "mtry" and "ntree" arguments which control the number of variables per tree and number of trees in the algorithm, respectively. Like our original pruned tree, the number of variables that resulted in the highest accuracy of the testing data were at 12. The accuracy at an "mtry" of 12 with the default "ntree" argument was 0.5367, as previously mentioned as our original maximum accuracy for the

random forest model. However, by changing the "ntree" argument to a larger number of trees, the accuracy was able to increase all the way to 0.5610, our highest yet. This accuracy was found by selecting 2000 trees to be made in our random forest, and we saw a trend of a steady increase until around 2000, and then a decline. A value of 1 for the "ntree" argument is equivalent to conducting a single tree model, and by running this a number of times, the accuracy was often less than 0.5. By increasing the number of trees, only the important predictors are most prevalent, which is why we see an increase in accuracy. We struggled to get any further increase because, as shown in the below plot, increase in number of trees does not always result in less error attached to the model, in fact, at around 2250, the error begins to increase once again. This is how we decided that we optimized our number of nodes and number of trees, and our accuracy of 0.5610 was around the highest we would get with the random forest model and overall.



Plot of Out Of Bag Error (OOB) versus number of trees in Random Forest

Conclusion/Future Work

Throughout our process, we tried many different models to get the highest predictive accuracy for students' math scores. Since the response variable had four levels, it was difficult for any of our models to predict results accurately, as the models had to precisely sort their predictions into different groups. Since none of the predictors were continuous variables, it was even more difficult to make splits based on other factored variables that may or may not have had importance, resulting in relatively low accuracy throughout the process. Randomness played a large part in the original construction of our tree models, as any given tree could have vastly different accuracy from another based on the selection of variables. Even in the random forest algorithm with thousands of trees, without a set seed, the accuracy was noticeably different in each forest. Therefore, we cannot confidently say that our accuracy we ultimately arrived at, 0.5610, was necessarily close to the true maximum predictive ability of a random forest model. We made checks based on the number of variables sampled and number of trees in the forest, but still, the randomness of the process is a major confounding factor.

We believe that we were relatively close to what the highest accuracy of a random forest for our data would be, but we only attempted random forests with every predictor included into our model. We did this to ensure that no potentially significant predictor would be left out, but we could have experimented more with different combinations of predictors. We also could have treated the variables differently originally, as we primarily made almost all of the categorical variables into factors when there potentially could be interesting results depending on other types of treatment. Other potential work could include the usage of Extra Trees (extremely randomized trees) which choose random split points for trees. This could be helpful due to the nature of our predictors where there was little meaning attached to some of the values of our factored

categorical variables. Still, for our methods, the random forest algorithm we utilized resulted in the highest accuracy, and our manipulation of the arguments within the random forest ensured that we verified that we accounted for as much as we could.

Division of Labor

We worked very fluidly together, collaborating on many of the ideas to create our models. We individually coded out each model and discussed how we could improve our accuracy. In terms of the documentation of our work, we didn't have to discuss who was doing what parts, we simply filled in the blanks and appended each other's work. Specifically, Ryan began the EDA by making visualizations of each variable, and Jack helped finish the process. For the final paper, Jack did the introduction, method, and conclusion section, Ryan did the data, the bulk of the results section, and uploaded the code. We never felt that either one of us was doing a disproportionate amount of work, and we communicated very well when it was necessary.

Final Document:

https://drive.google.com/file/d/1BIspl9BpJrpDv5E7SWFBiC8ieV3kWt2H/view?usp=share_link

Predicted Outcomes csv:

https://drive.google.com/file/d/1WXtrhkmuIrAJJciczDa4pevuMAAiF_7J/view?usp=share_link