351 - Final Project Code

Ryan O’Neill, Jack Dumanski

2025-12-06

```
library(readr)
MathScores <- read_csv("MathScores.csv")
```

```
## Rows: 273 Columns: 32
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (18): school, sex, address, famsize, Pstatus, Mjob, Fjob, reason, guardi...
## dbl (14): ID, age, Medu, Fedu, traveltime, studytime, failures, famrel, free...
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
MathScores$G4 = as.factor(MathScores$G4)
MathScores$G4 = factor(MathScores$G4, levels=c("F", "NI", "S", "E"))
MathScores$school = as.factor(MathScores$school)
MathScores$sex = as.factor(MathScores$sex)
MathScores$address = as.factor(MathScores$address)
MathScores$famsize = as.factor(MathScores$famsize)
MathScores$school = as.factor(MathScores$school)
MathScores$Pstatus = as.factor(MathScores$Pstatus)
MathScores$Mjob = as.factor(MathScores$Mjob)
MathScores$Fjob = as.factor(MathScores$Fjob)
MathScores$reason = as.factor(MathScores$reason)
MathScores$guardian = as.factor(MathScores$guardian)
MathScores$schoolsup = as.factor(MathScores$schoolsup)
MathScores$famsup = as.factor(MathScores$famsup)
MathScores$paid = as.factor(MathScores$paid)
MathScores$activities = as.factor(MathScores$activities)
MathScores$nursery = as.factor(MathScores$nursery)
MathScores$higher = as.factor(MathScores$higher)
MathScores$internet = as.factor(MathScores$internet)
MathScores$romantic = as.factor(MathScores$romantic)
```

Splitting the Dataset:

```
set.seed(98)
idx = sample(1:nrow(MathScores), nrow(MathScores)*.7)
MathTrain = MathScores[idx,]
MathTest = MathScores[-idx,]
```

Simple tree:

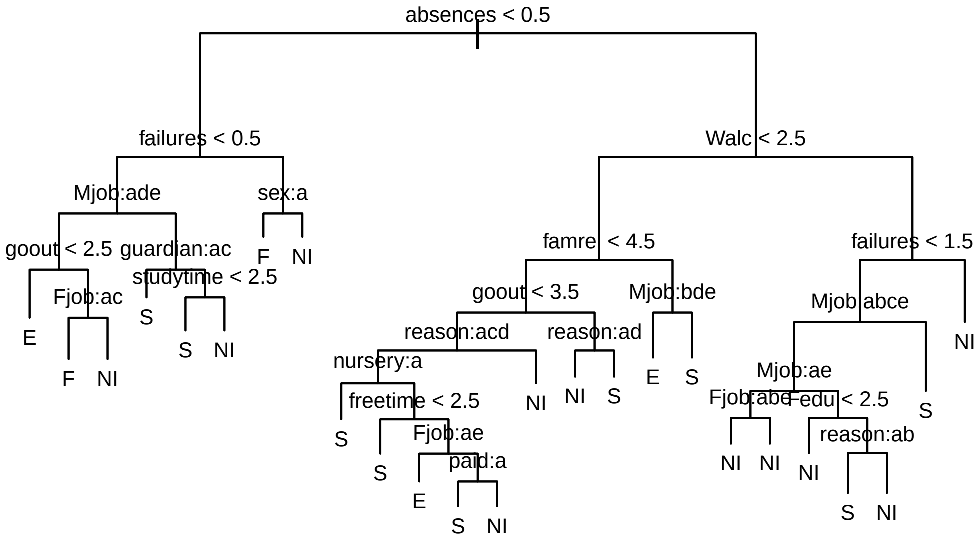
```r
library(tree)
InitialTree = tree(G4~.-ID, MathTrain)
summary(InitialTree)
```

```
##
## Classification tree:
## tree(formula = G4 ~ . - ID, data = MathTrain)
## Variables actually used in tree construction:
##  [1] "absences"  "failures"  "Mjob"      "goout"     "Fjob"      "guardian"
##  [7] "studytime" "sex"       "Walc"      "famrel"    "reason"    "nursery"
## [13] "freetime"  "paid"      "Fedu"
## Number of terminal nodes:  25
## Residual mean deviance:  1.136 = 188.5 / 166
## Misclassification error rate: 0.2356 = 45 / 191
```

```r
TestPredict = predict(InitialTree, MathTest, type="class")
(accuracy.test = mean(TestPredict == MathTest$G4))
```

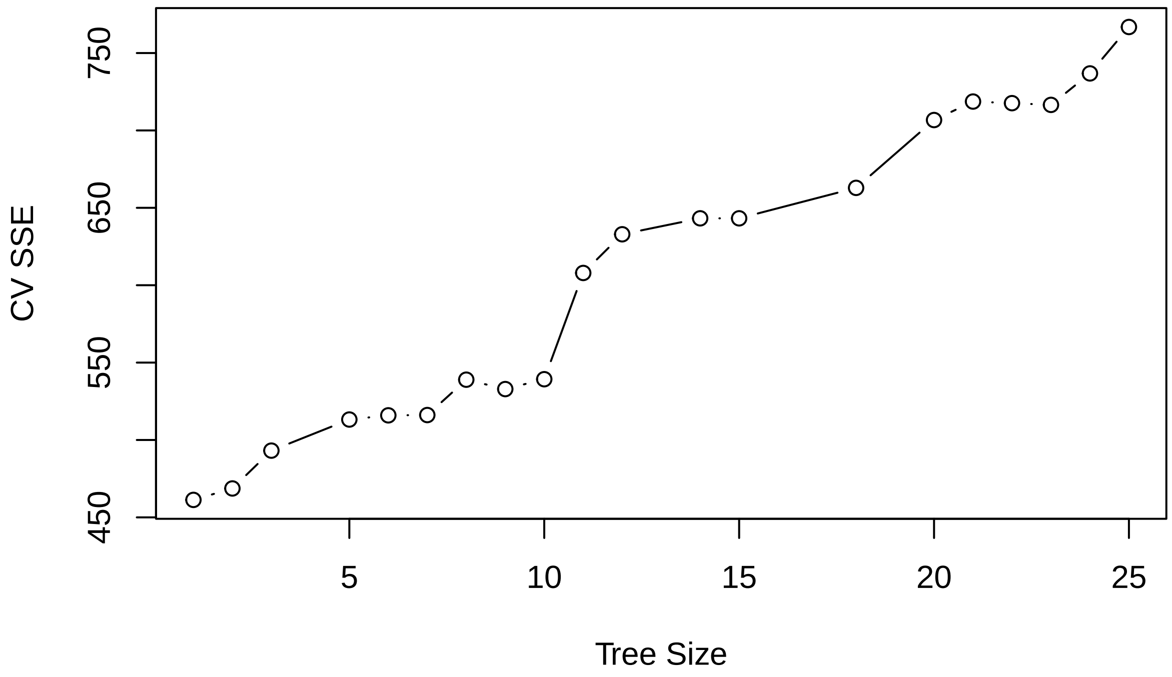
```
## [1] 0.3902439
```

```r
plot(InitialTree)
text(InitialTree, cex=.65)
```



Cross-Validation and Pruning:

```
MathCV = cv.tree(InitialTree)
plot(MathCV$size, MathCV$dev, type="b", xlab = "Tree Size", ylab = "CV SSE")
```



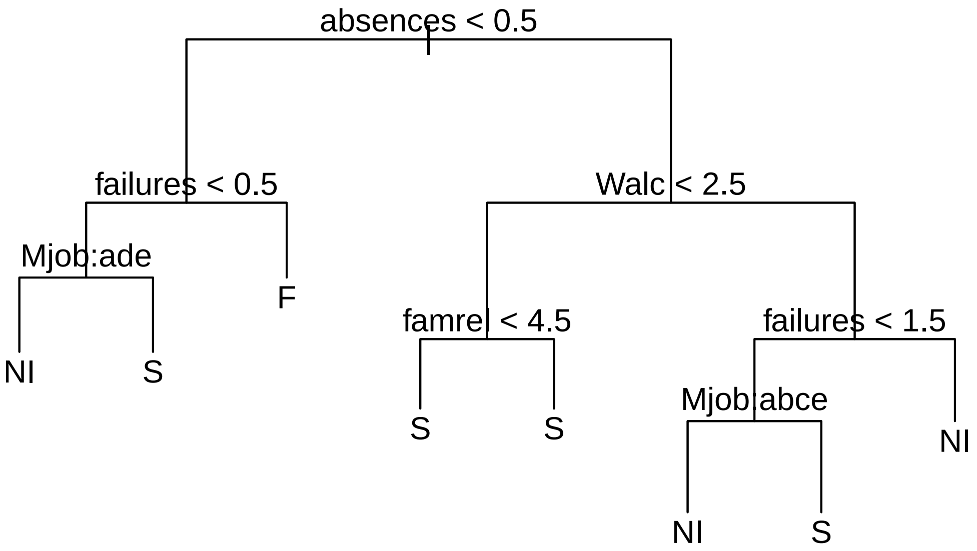
```
PruneMath = prune.tree(InitialTree, best=8)
summary(PruneMath)
```

```
##
## Classification tree:
## snip.tree(tree = InitialTree, nodes = c(5L, 9L, 28L, 12L, 13L,
## 8L))
## Variables actually used in tree construction:
## [1] "absences" "failures" "Mjob"     "Walc"     "famrel"
## Number of terminal nodes:  8
## Residual mean deviance:  1.759 = 321.9 / 183
## Misclassification error rate: 0.3822 = 73 / 191
```

```
TestPrunePredict = predict(PruneMath, MathTest, type="class")
(accuracy.train = mean(TestPrunePredict == MathTest$G4))
```

```
## [1] 0.4390244
```

```
plot(PruneMath)
text(PruneMath)
```



Random Forest:

```
library(randomForest)
```

```
## randomForest 4.7-1.2
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

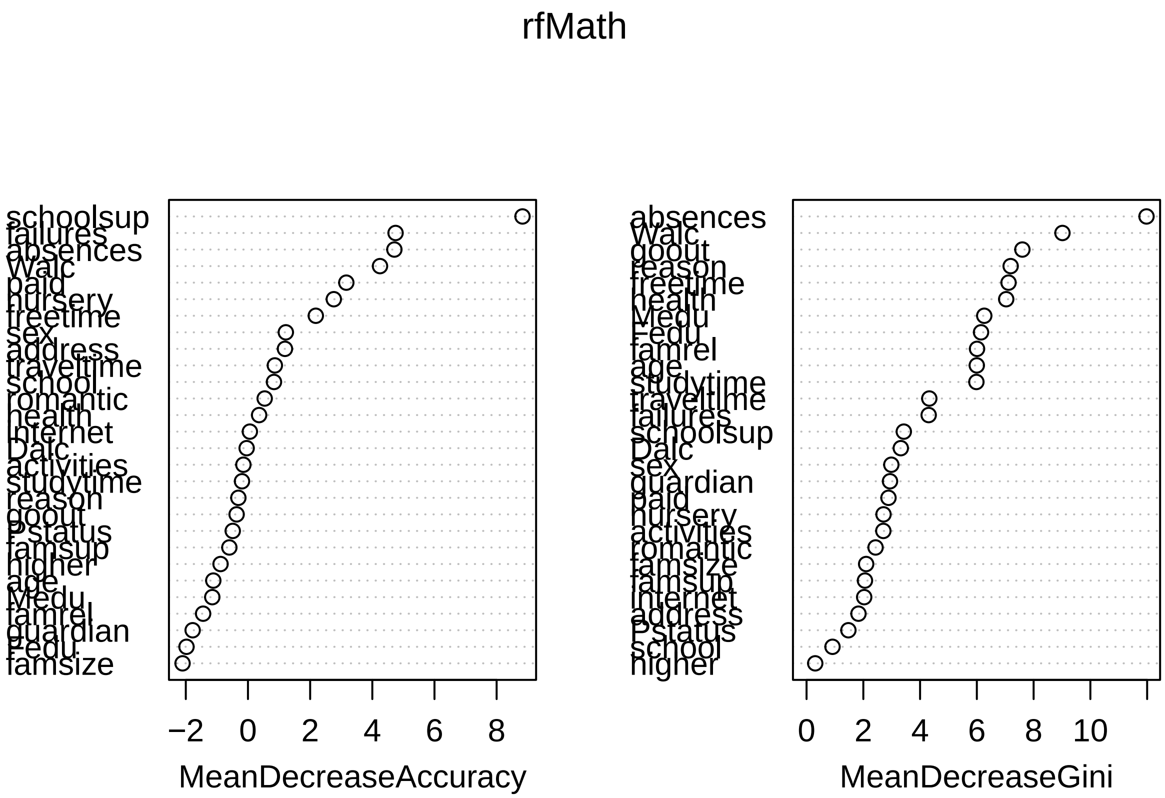
```
rfMath = randomForest(G4~.-ID-Fjob-Mjob, MathTrain, mtry=8, importance=TRUE)
G4PredictTrain = predict(rfMath, MathTrain, type="class")
(accuracy.full= mean(G4PredictTrain == MathTrain$G4))
```

```
## [1] 1
```

```
G4PredictTest = predict(rfMath, MathTest, type="class")
(accuracy.full.test= mean(G4PredictTest == MathTest$G4))
```

```
## [1] 0.5365854
```

```
varImpPlot(rfMath)
```



*variance importance plot might look slightly different than plot in our paper based on variation in random forest process

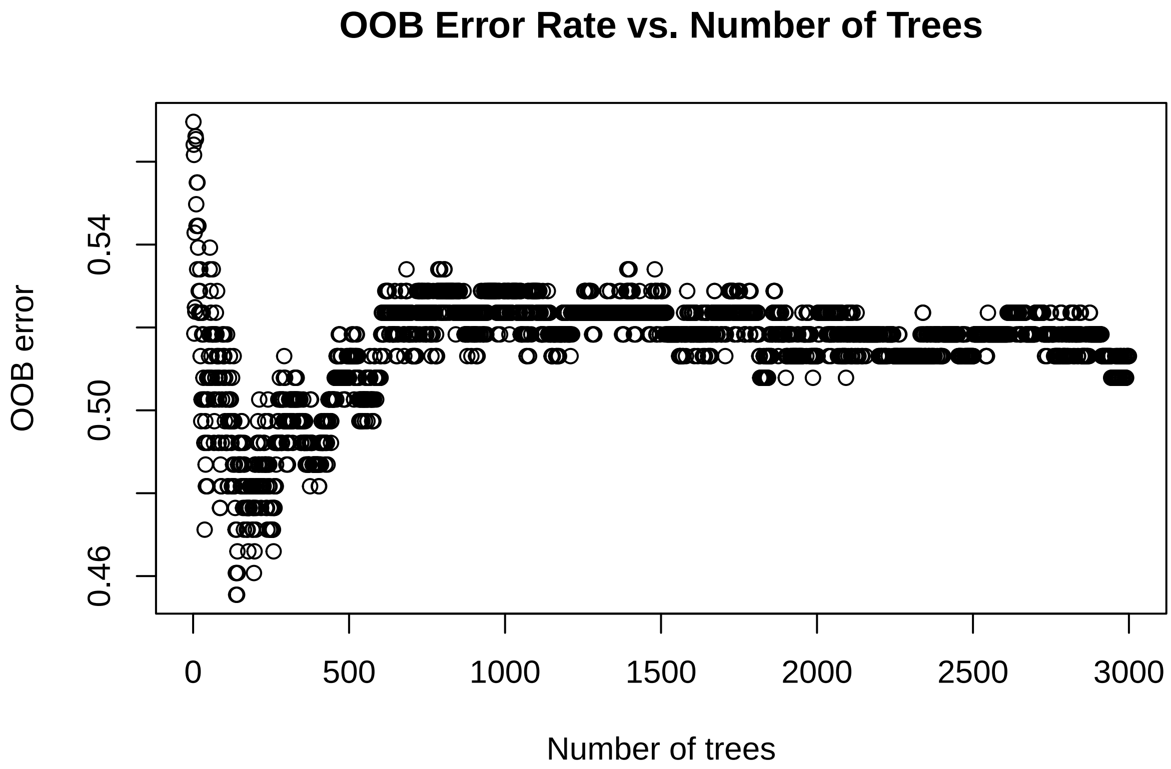
```
rf.scores = randomForest(G4 ~ . - ID, data = MathTrain, mtry = 12, ntree = 2000, importance = TRUE)
scorespredictfulltrn = predict(rf.scores,MathTrain,type = "class")
(accuracy.full3= mean(scorespredictfulltrn == MathTrain$G4))
```

```
## [1] 1
```

```
scorespredictfulltst = predict(rf.scores,MathTest,type = "class")
(accuracy.full4= mean(scorespredictfulltst == MathTest$G4))
```

```
## [1] 0.5
```

```
rf.scores = randomForest(G4 ~ . - ID, data = MathTrain, mtry = 12, ntree = 3000, importance = TRUE)
plot(rf.scores$err.rate[,1], main = "OOB Error Rate vs. Number of Trees", xlab = "Number of trees", ylab
```



Predicting on New Data

```
library(readr)
Results <- read_csv("MathScorestest.csv")
```

```
## Rows: 119 Columns: 31
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (18): ID, school, sex, address, famsize, Pstatus, Mjob, Fjob, reason, gu...
## dbl (13): age, Medu, Fedu, traveltime, studytime, failures, famrel, freetime...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
Results$school = as.factor(Results$school)
Results$sex = as.factor(Results$sex)
Results$address = as.factor(Results$address)
Results$famsize = as.factor(Results$famsize)
Results$school = as.factor(Results$school)
Results$Pstatus = as.factor(Results$Pstatus)
Results$Mjob = as.factor(Results$Mjob)
Results$Fjob = as.factor(Results$Fjob)
Results$reason = as.factor(Results$reason)
Results$guardian = as.factor(Results$guardian)
Results$schoolsup = as.factor(Results$schoolsup)
```

```r
Results$famsup = as.factor(Results$famsup)
Results$paid = as.factor(Results$paid)
Results$activities = as.factor(Results$activities)
Results$nursery = as.factor(Results$nursery)
Results$higher = as.factor(Results$higher)
Results$internet = as.factor(Results$internet)
Results$romantic = as.factor(Results$romantic)

predict_rf <- predict(rf.scores, newdata = Results, type = "class")
PredictedOutcomes = data.frame(Results$ID, predict_rf)
write.csv(PredictedOutcomes, file = "Dumanski_Oneill_MathScorePrediction.csv")
```