Benjamin Rothschild: PS8

Part 1: Biden

```r
suppressWarnings(suppressMessages(library(tidyverse)))
suppressWarnings(suppressMessages(library(forcats)))
suppressWarnings(suppressMessages(library(broom)))
suppressWarnings(suppressMessages(library(modelr)))
suppressWarnings(suppressMessages(library(tree)))
suppressWarnings(suppressMessages(library(randomForest)))
suppressWarnings(suppressMessages(library(stringr)))
suppressWarnings(suppressMessages(library(ISLR)))
suppressWarnings(suppressMessages(library(gridExtra)))
suppressWarnings(suppressMessages(library(grid)))
suppressWarnings(suppressMessages(library(pROC)))
suppressWarnings(suppressMessages(library(gbm)))
devtools::install_github("bensoltoff/ggdendro")
```

```
## Skipping install of 'ggdendro' from a github remote, the SHA1 (9c9c6e7e) has not changed since last 
##   Use `force = TRUE` to force installation
```

```r
suppressWarnings(suppressMessages(library(ggdendro)))
knitr::opts_chunk$set(echo = TRUE)
```

```r
options(digits = 3)
set.seed(1234)
theme_set(theme_minimal())
biden <- read_csv("biden.csv")
```

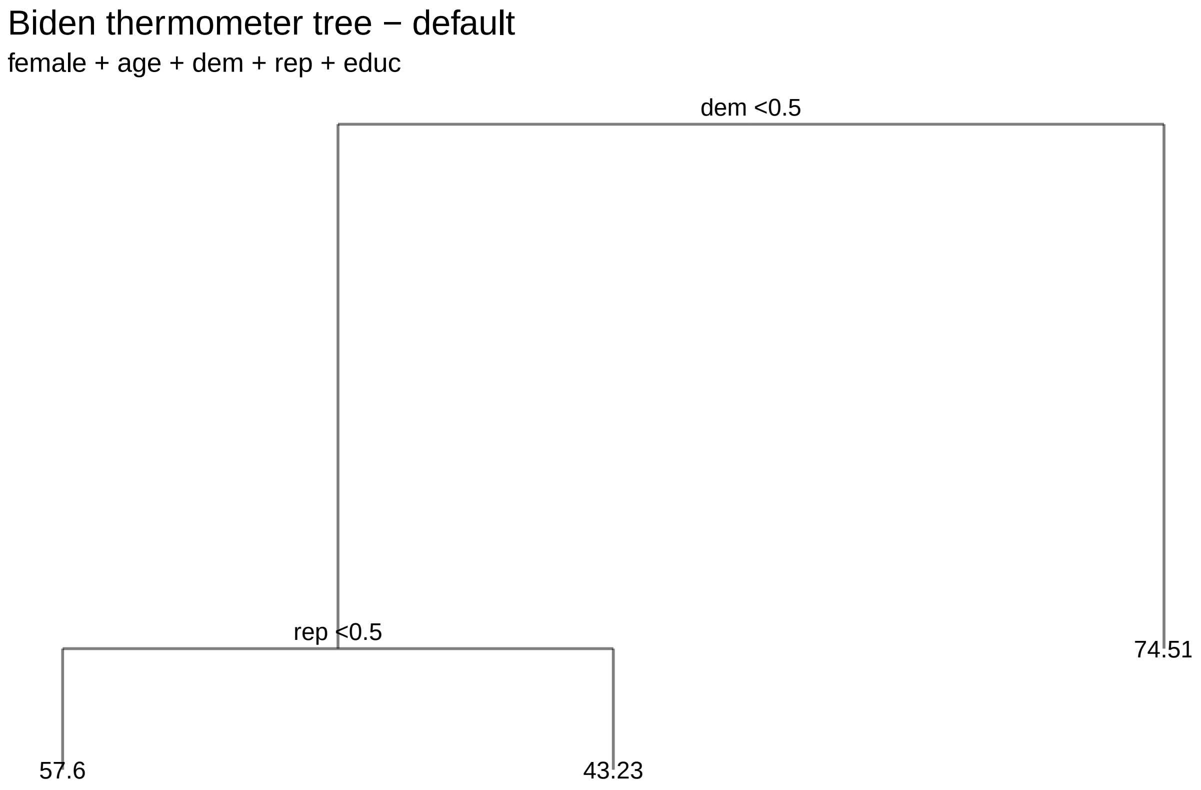
```
## Parsed with column specification:
## cols(
##   biden = col_integer(),
##   female = col_integer(),
##   age = col_integer(),
##   educ = col_integer(),
##   dem = col_integer(),
##   rep = col_integer()
## )
```

```r
biden_split <- resample_partition(biden, c(test = 0.3, train = 0.7))

mse <- function(model, data) {
  x <- modelr:::residuals(model, data)
  mean(x ^ 2, na.rm = TRUE)
}
#Grow tree
biden_tree_default <- tree(biden ~ female + age + dem + rep + educ, data = biden_split$train)

mse(biden_tree_default, biden_split$test)
```

```
## [1] 406
```

1. I split the dataset into two pieces.
2. After fitting a decision tree on the data I got a MSE of 406. I plotted the tree below.

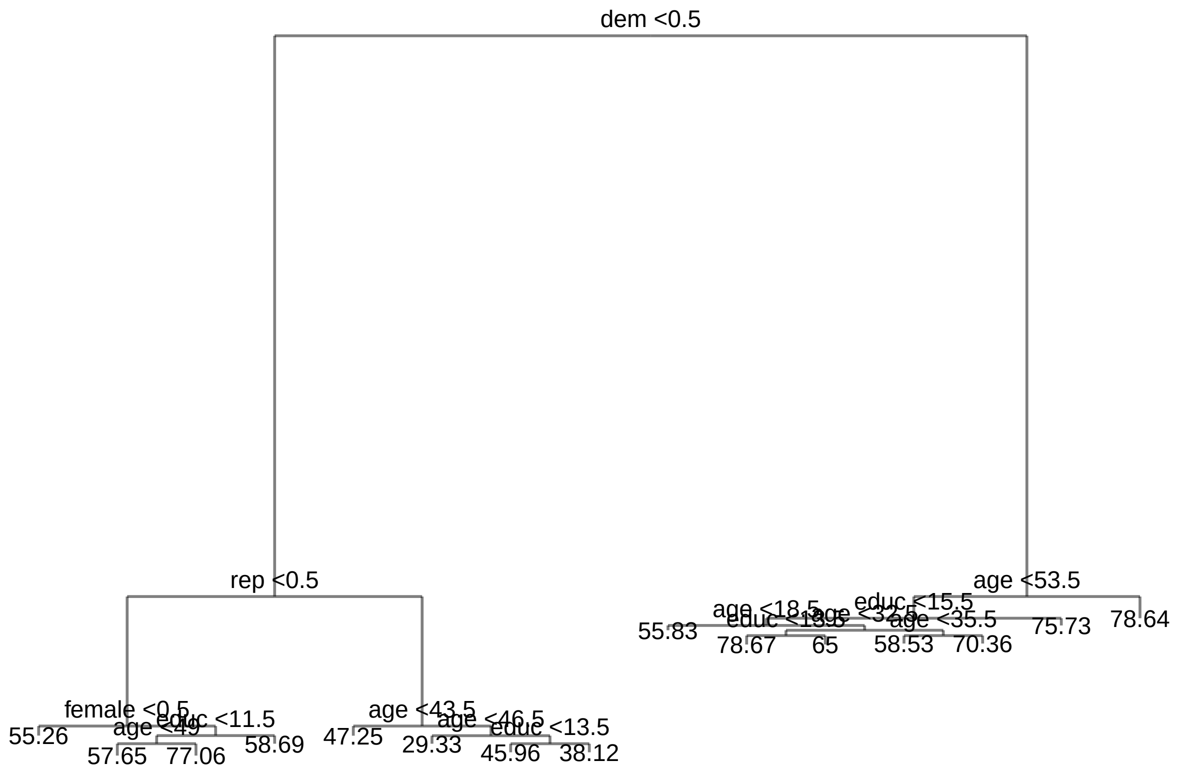
```r
#Plot tree
tree_data <- dendro_data(biden_tree_default)

ggplot(segment(tree_data)) +
  geom_segment(aes(x = x, y = y, xend = xend, yend = yend), alpha = 0.5) +
  geom_text(data = label(tree_data), aes(x = x, y = y, label = label_full), vjust = -0.5, size = 3) +
  geom_text(data = leaf_label(tree_data), aes(x = x, y = y, label = label), vjust = 0.5, size = 3) +
  theme_dendro() +
  labs(title = "Biden thermometer tree - default",
       subtitle = "female + age + dem + rep + educ")
```


Biden thermometer tree − default
female + age + dem + rep + educ
dem <0.5
rep <0.5
74.51
57.6
43.23

```r
options(digits = 3)
set.seed(1234)

fit_2 <- tree(biden ~ female + age + dem + rep + educ, data = biden_split$train, control = tree.control

mse(fit_2, biden_split$test)
```

3. Using this new tree I got a much lower MSE of 481. The plot of the tree is below.

```r
mod <- prune.tree(fit_2, best = 15)
tree_data <- dendro_data(mod)
ggplot(segment(tree_data)) +
  geom_segment(aes(x = x, y = y, xend = xend, yend = yend),
               alpha = 0.5) +
  geom_text(data = label(tree_data),
            aes(x = x, y = y, label = label_full), vjust = -0.5, size = 3) +
  geom_text(data = leaf_label(tree_data),
            aes(x = x, y = y, label = label), vjust = 0.5, size = 3) +
  theme_dendro()
```

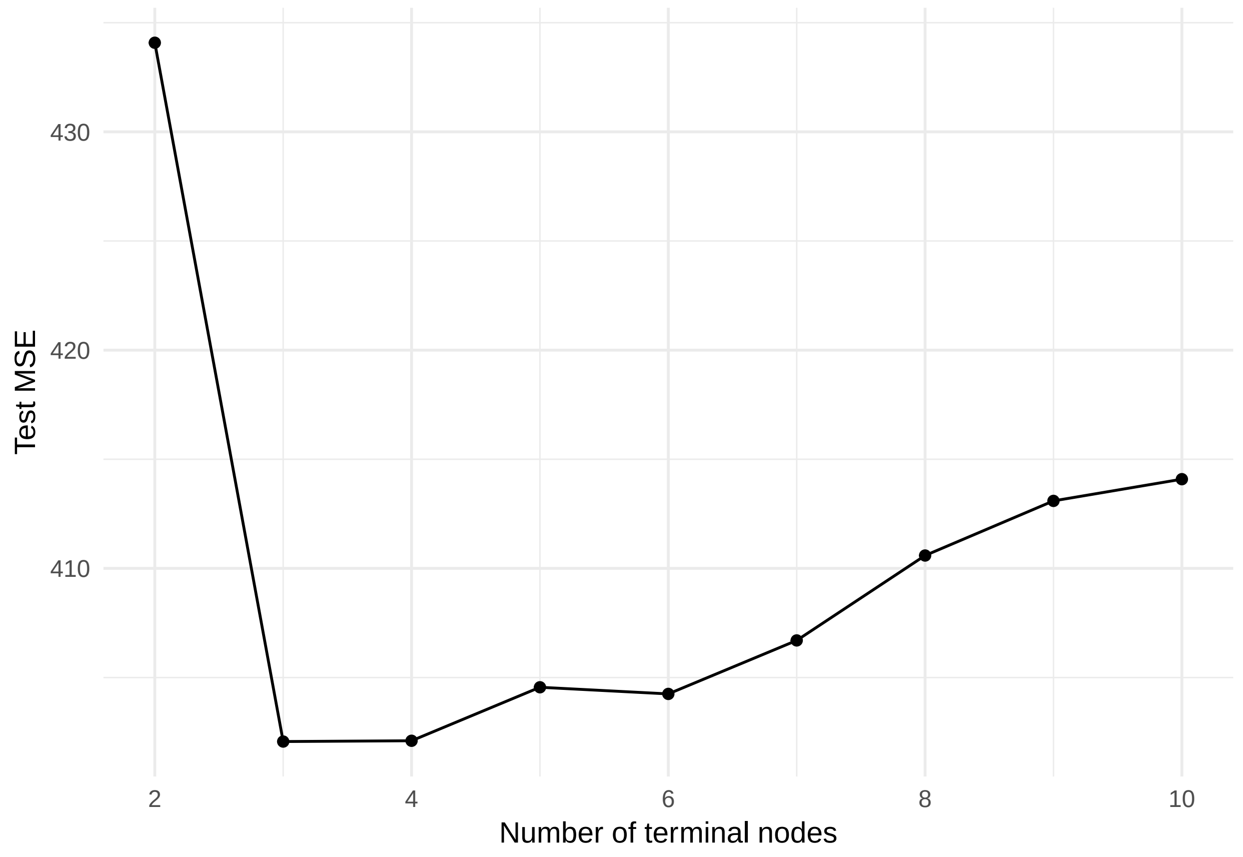


```r
# generate 10-fold CV trees
biden_cv <- crossv_kfold(biden, k = 10) %>%
  mutate(tree = map(train, ~ tree(biden ~ female + age + dem + rep + educ, data = .,
     control = tree.control(nobs = nrow(biden), mindev = 0))))

# calculate each possible prune result for each fold
biden_cv <- expand.grid(biden_cv$.id, 2:10) %>%
  as_tibble() %>%
  mutate(Var2 = as.numeric(Var2)) %>%
  rename(.id = Var1,
         k = Var2) %>%
  left_join(biden_cv) %>%
  mutate(prune = map2(tree, k, ~ prune.tree(.x, best = .y)),
         mse = map2_dbl(prune, test, mse))
```

```
## Joining, by = ".id"
```

```
## Warning in left_join_impl(x, y, by$x, by$y, suffix$x, suffix$y): joining
## character vector and factor, coercing into character vector
```

```r
biden_cv %>%
  select(k, mse) %>%
  group_by(k) %>%
  summarize(test_mse = mean(mse),
            sd = sd(mse, na.rm = TRUE)) %>%
  ggplot(aes(k, test_mse)) +
  geom_point() +
  geom_line() +
  labs(x = "Number of terminal nodes",
       y = "Test MSE")
```

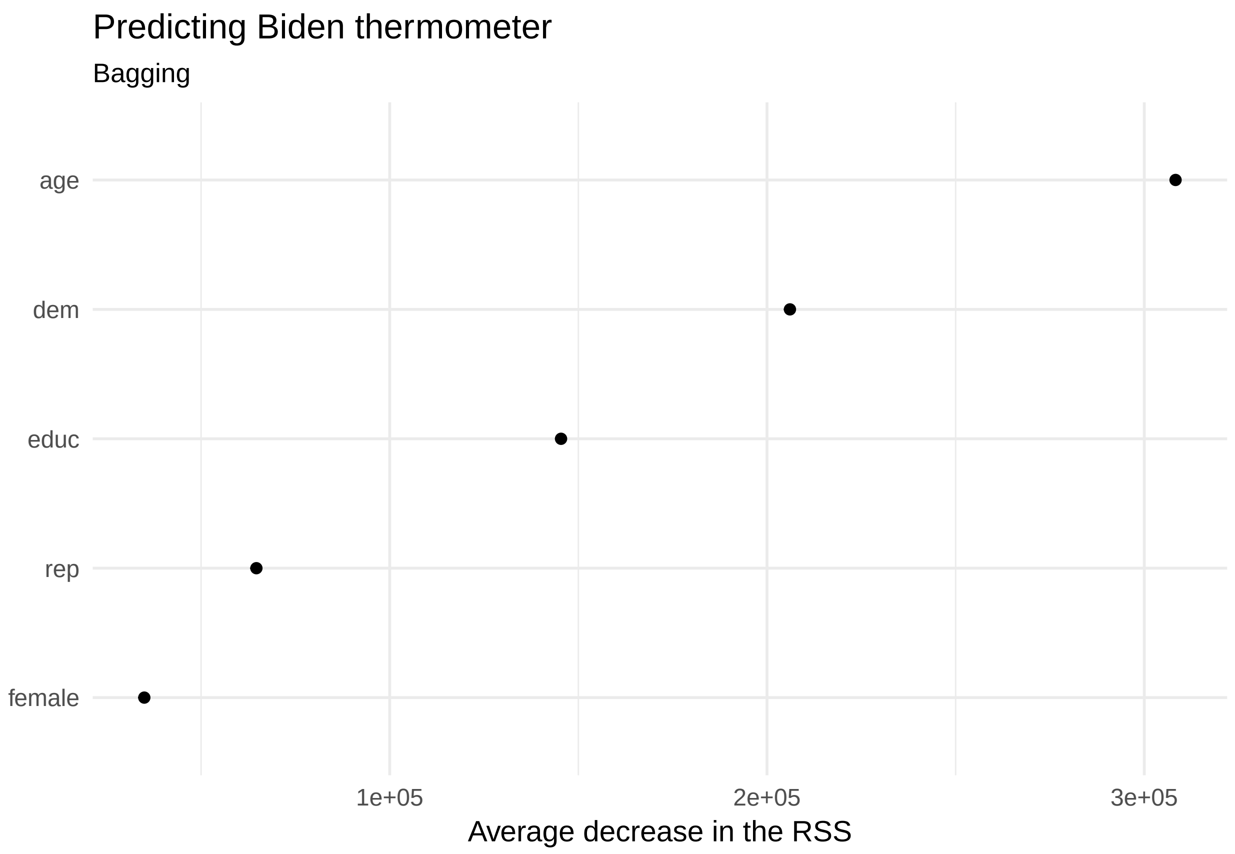


3. From this analysis you can see that the MSE is lowest with three-four terminal nodes. This is awesome and makes for a simple tree structure. The lowest MSE is around 395. This shows that pruning the tree does improve the model's fitness.

```
set.seed(1234)

biden_bag <- randomForest(biden ~ ., data = biden, mtry = 5, ntree = 500)
biden_bag
```

```
##
## Call:
##  randomForest(formula = biden ~ ., data = biden, mtry = 5, ntree = 500)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 5
##
##           Mean of squared residuals: 493
##                     % Var explained: 10.4
```

```
data_frame(var = rownames(importance(biden_bag)),
           MeanDecreaseRSS = importance(biden_bag)[,1]) %>%
  mutate(var = fct_reorder(var, MeanDecreaseRSS, fun = median)) %>%
  ggplot(aes(var, MeanDecreaseRSS)) +
  geom_point() +
  coord_flip() +
  labs(title = "Predicting Biden thermometer",
       subtitle = "Bagging",
       x = NULL,
       y = "Average decrease in the RSS")
```



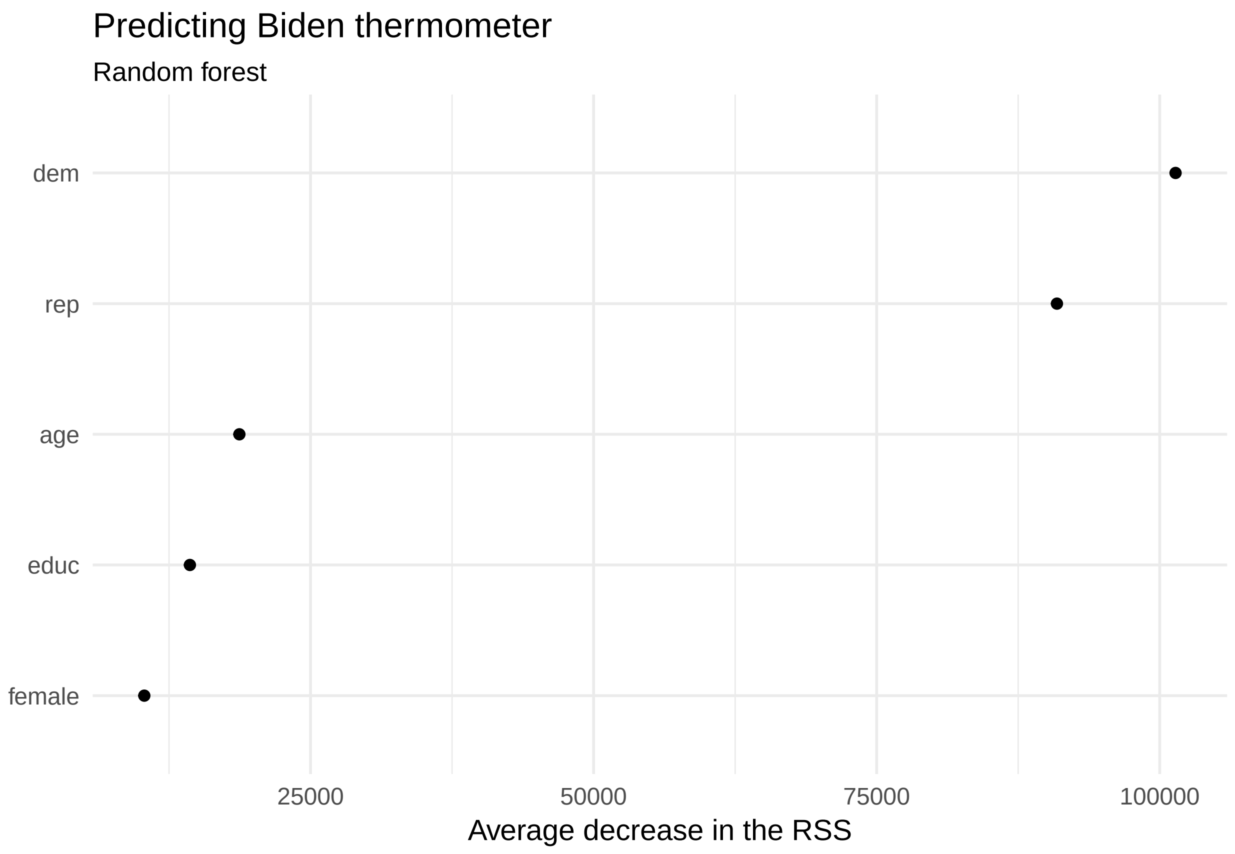
4. With bagging the test MSE is 495. This MSE is higher than what I got with the previous methods. The variable importance graph I made is above. It shows that age, dem, and educ are the most important predictors of the model. After that rep and female.

```
set.seed(1234)

biden_rf <- randomForest(biden ~ ., data = biden, ntree = 500)
biden_rf
```

```
##
## Call:
##  randomForest(formula = biden ~ ., data = biden, ntree = 500)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 1
##
##           Mean of squared residuals: 407
##                     % Var explained: 26
```

```
data_frame(var = rownames(importance(biden_rf)),
           MeanDecreaseRSS = importance(biden_rf)[,1]) %>%
  mutate(var = fct_reorder(var, MeanDecreaseRSS, fun = median)) %>%
  ggplot(aes(var, MeanDecreaseRSS)) +
  geom_point() +
```

```r
  coord_flip() +
  labs(title = "Predicting Biden thermometer",
       subtitle = "Random forest",
       x = NULL,
       y = "Average decrease in the RSS")
```



5. The random first model returned a MSE of 408 which is much lower than the MSE from bagging. The graph above shows the most important predictor is dem and rep and the importance of educ and age is less compared to the bagging model.

```r
set.seed(1234)
biden_split <- resample_partition(biden, c(test = 0.3, train = 0.7))
biden_boost <- gbm(biden ~ ., data = biden_split$train, n.trees = 10000, interaction.depth = 1)
```

```
## Distribution not specified, assuming gaussian ...
```

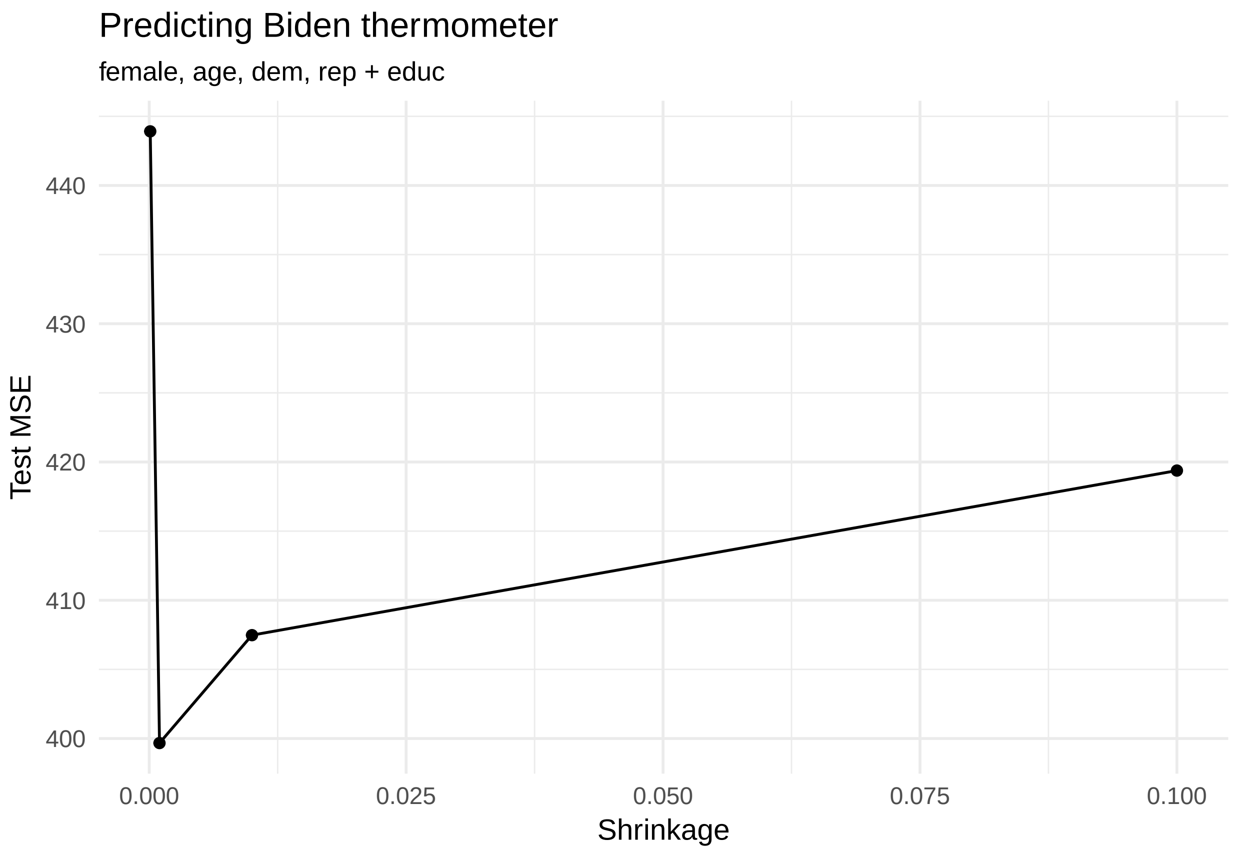
```r
yhat.boost = predict(biden_boost, newdata = biden_split$test, n.trees = 10000)

mean((yhat.boost - biden[biden_split$test[2]$idx, ]$biden)^2)
```

```
## [1] 400
```

```r
mses <- numeric(4)
shrinkages <- numeric(4)
for (s in 1:4){
  shrinkages[s] <- 10^(-s)
  biden_boost <- gbm(biden ~ ., data = biden_split$train, n.trees = 10000, interaction.depth = 1, shrin
  yhat.boost = predict(biden_boost, newdata = biden_split$test, n.trees = 10000)
  mses[s] <- mean((yhat.boost - biden[biden_split$test[2]$idx, ]$biden)^2)
}
```

```
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
```

```
data_frame(mse = mses, shrinkage = shrinkages) %>%
  ggplot(aes(shrinkage, mse)) +
  geom_point() +
  geom_line() +
  labs(title = "Predicting Biden thermometer",
       subtitle = "female, age, dem, rep + educ",
       x = "Shrinkage",
       y = "Test MSE")
```



6. With boosting I got a test MSE of 399 which is one of the lowest I have gotten so far. I tested different shrinkage levels and was able to get the lowest MSE at around 0.001

Part 2

Tree-Based Model

```
voting_data <- read_csv("mental_health.csv")
```

```
## Parsed with column specification:
```

```
## cols(
##   vote96 = col_double(),
##   mhealth_sum = col_double(),
##   age = col_double(),
##   educ = col_double(),
##   black = col_double(),
##   female = col_double(),
##   married = col_double(),
##   inc10 = col_double()
## )
```

```r
mse <- function(model, data) {
  x <- modelr:::residuals(model, data)
  mean(x ^ 2, na.rm = TRUE)
}

vote_val_test <- function(){
  voting_data_split <- resample_partition(voting_data, p = c(test = 0.3, train = 0.7))
  # fit <- tree(vote96 ~ mhealth_sum + age + educ + black + female + married + inc10, data =voting_data
  fit <- tree(vote96 ~ age + educ + black + married + inc10, data=voting_data_split$train)
  mse(fit, voting_data_split$test)
}

val_mse <- data_frame(id = 1:1000, mse = map_dbl(id, ~ vote_val_test()))
mean(val_mse$mse)
```

```
## [1] 0.197
```

```r
# ggplot(val_mse, aes(mse)) + geom_histogram() + geom_vline(xintercept = mean(val_mse$mse))
```

1. To perform this analysis I set up a framework that would shuffle my data and calculate the MSE of my 70/30 train/test set 1,000 times. To find a model of best fit I made up an algorithm I thought would provide me a good answer. First I found the mse of the model between voting and each one of the other variables. After comparing the Average MSE for each of these models I took the variable that produced the lowest MSE (educ) and then tried a two variable model with educ and each one of the remaining variables. I repeated this process 5 times. Though I don't think this algorithm will always produce the best results I think it provides a simple way to a lot of models. The results are below. This table can be a little hard to interpret. Basically the second column is the the MSE of each of the varibles in the 1st column. The Third column is would be the MSE of educ + the variables that appear below the educ in column one. (i.e. the MSE of vote96 ~ educ + age is .196, vote ~ educ + black is .206)

Average MSE over 1,000 Iterations					
educ	0.207				
age	0.206	0.196			
black	0.217	0.206	0.196		
married	0.215	0.206	0.196	0.196	
mhealth_sum	0.212	0.204	0.198	0.198	0.198
female	0.217	0.206	0.196	0.196	0.196
inc10	0.213	0.203	0.197	0.197	0.198

Interestingly the lowest MSE I achieved was .196. This was achieved by only using two variables in my model educ and age. By combining more than two variables I wasn't able to lower my MSE at all.

SVM Model

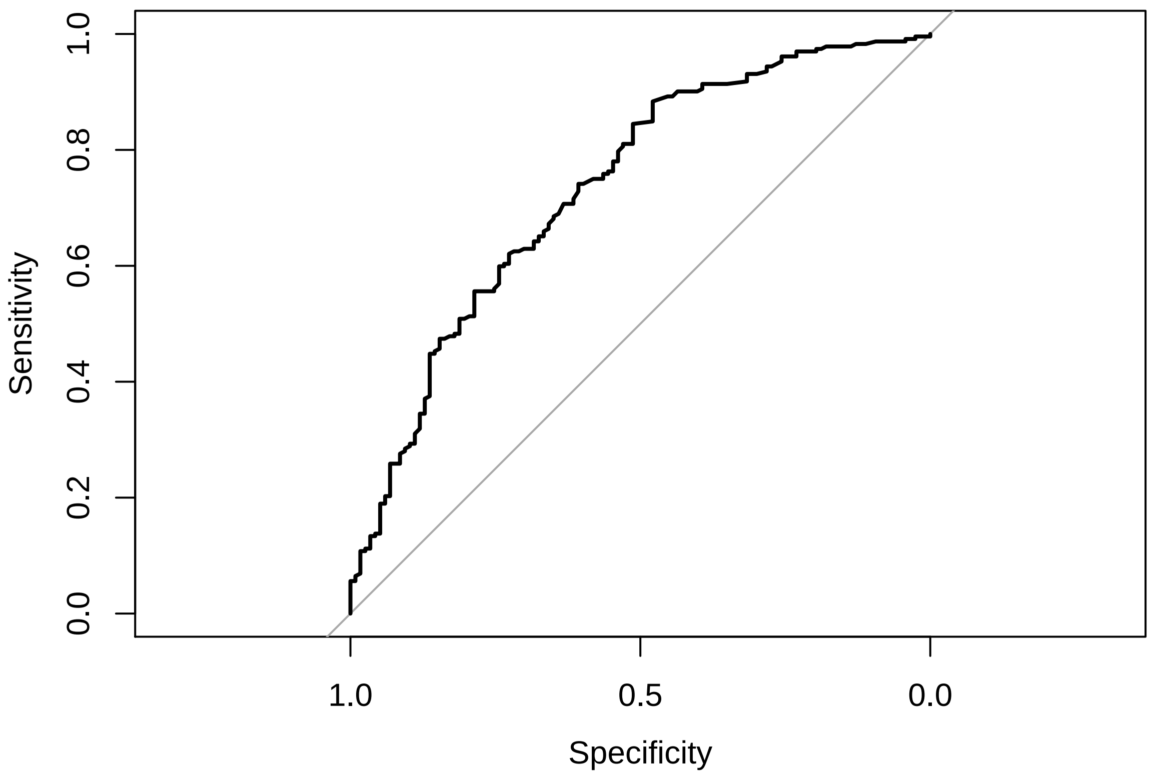
```r
library(e1071)
set.seed(1234)
df_mental <- read_csv("mental_health.csv")
```

```
## Parsed with column specification:
## cols(
##   vote96 = col_double(),
##   mhealth_sum = col_double(),
##   age = col_double(),
##   educ = col_double(),
##   black = col_double(),
##   female = col_double(),
##   married = col_double(),
##   inc10 = col_double()
## )
```

```r
mh_split <- resample_partition(na.omit(df_mental), p = c("test" = .3, "train" = .7))

mh_lin_tune <- tune(svm, vote96 ~ educ + age, data = as_tibble(mh_split$train),
                    kernel = "linear",
                    range = list(cost = c(.001, .01, .1, 1, 5, 10, 100)))
mh_lin <- mh_lin_tune$best.model
fitted <- predict(mh_lin, as_tibble(mh_split$test), decision.values = TRUE) %>%
  attributes

roc_line <- roc(as_tibble(mh_split$test)$vote96, fitted$decision.values)
```

```
## Warning in roc.default(as_tibble(mh_split$test)$vote96, fitted
## $decision.values): Deprecated use a matrix as predictor. Unexpected results
## may be produced, please pass a numeric vector.
```

```r
plot(roc_line)
```



```
auc(roc_line)
```

```
## Area under the curve: 0.736
```

```
real <- na.omit(as.numeric(as_tibble(mh_split$test)$vote96))
E1 <- mean(as.numeric(real != median(real)))
E2 <- 0.2856
PRE <- (E1 - E2) / E1
PRE
```

```
## [1] 0.148
```

To start the analysis I am going to perform the same analysis with the four kernel types that are available in R, linear, polyomial, radial, and sigmoid.

Linear Kernel: Above I first tried using a linear kernel and got an error rate of 28%. The AUC was .746 and the PRE was 14.81%. This model performs pretty well (decreasing error rate by 14.81% compared to a random model)

Polynomial Kernel: Performing the same analysis with a polynomial kernel I get error rate of 30.15% AUC .7016 and PRE of 14.8%.

Radial Kernel: Performing the same analysis with a radial kernel I get error rate of 31.87% AUC 0.735 and PRE of 12.9%

Sigmoid Kernel: Performing the same analysis with a radial kernel I get error rate of 31.87% AUC .6029 and PRE of 14.8%.

From these results I believe that the linear model performs best with an error rate of 28%. From here I will try another version of the linear kernel.

From the first part of my analysis I found that age and educ were the most important variables and adding more did not help reduce the MSE. So I will try a Linear Kernel SVM with these two variables. Here I get AUC of .7356 and PRE of 14.8%. This did not improve the SVM model.

Part 3: O.J. Simpson

```
simpson_data <- read_csv("simpson_coded.csv")
```

```
## Warning: Missing column names filled in: 'X4' [4]
```

```
## Parsed with column specification:
## cols(
##   guilt = col_integer(),
##   dem = col_integer(),
##   rep = col_integer(),
##   X4 = col_integer(),
##   ind = col_integer(),
##   age = col_integer(),
##   educ = col_integer(),
##   female = col_integer(),
##   black = col_integer(),
##   hispanic = col_integer(),
##   income = col_integer()
## )
```

```
summary(simpson_data)
```

```
##      guilt           dem              rep               X4              ind
##  Min.   :0.0   Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0
##  1st Qu.:0.0   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:1.000   1st Qu.:0
##  Median :1.0   Median :0.000   Median :0.000   Median :1.000   Median :0
##  Mean   :0.7   Mean   :0.489   Mean   :0.391   Mean   :0.881   Mean   :0
##  3rd Qu.:1.0   3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:0
##  Max.   :1.0   Max.   :1.000   Max.   :1.000   Max.   :1.000   Max.   :0
##  NA's   :153
##       age             educ           female           black
##  Min.   :18.0   Min.   : 5.0   Min.   :0.000   Min.   :0.000
##  1st Qu.:32.0   1st Qu.:12.0   1st Qu.:0.000   1st Qu.:0.000
##  Median :41.0   Median :14.0   Median :1.000   Median :0.000
##  Mean   :44.7   Mean   :13.1   Mean   :0.584   Mean   :0.194
##  3rd Qu.:56.0   3rd Qu.:16.0   3rd Qu.:1.000   3rd Qu.:0.000
##  Max.   :99.0   Max.   :16.0   Max.   :1.000   Max.   :1.000
##                 NA's   :6
##     hispanic          income
##  Min.   :0.000   Min.   :  7500
##  1st Qu.:0.000   1st Qu.: 22500
##  Median :0.000   Median : 40000
##  Mean   :0.068   Mean   : 41185
##  3rd Qu.:0.000   3rd Qu.: 62500
##  Max.   :1.000   Max.   :100000
##                  NA's   :92
```

```
fit <- glm(formula = guilt ~ dem + age + educ + female + black + hispanic + income, family = binomial, 
summary(fit)
```

```
##
## Call:
## glm(formula = guilt ~ dem + age + educ + female + black + hispanic +
##     income, family = binomial, data = simpson_data)
##
```

```
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.407  -0.535   0.520   0.664   2.272
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.06e-01   4.00e-01   -1.26  0.20614
## dem         -2.96e-01   1.51e-01   -1.97  0.04925 *
## age          1.83e-02   4.65e-03    3.94    8e-05 ***
## educ         9.57e-02   2.51e-02    3.81  0.00014 ***
## female      -3.32e-01   1.51e-01   -2.19  0.02845 *
## black       -2.92e+00   1.94e-01  -15.01  < 2e-16 ***
## hispanic    -1.75e-01   2.60e-01   -0.67  0.50146
## income       7.53e-06   3.21e-06    2.35  0.01889 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1646.2  on 1340  degrees of freedom
## Residual deviance: 1210.1  on 1333  degrees of freedom
##   (228 observations deleted due to missingness)
## AIC: 1226
##
## Number of Fisher Scoring iterations: 4
```

1. In order to investigate the relationship between belief of guilt and the other variables I decided to use a logistic regression because the output will give a classification of 1 or 0 (1 - belief of guilt or 0 - belief of innosence). If I used a linaer regression I would obtain predicted values that do not make sense in this instance, for example any value that is not 0 or 1.

When I took a look at the policital party data I noticed that there are no indpendents in the dataset. In the dataset a person can either be democrat or republican (not both) and 88% or respondents are democrat or republican while the rest did not respond. I should only include one of the party ids as an indepent variable because the other has a linear relationship of the other which we don't want in a regression analysis.

Next I noticed that the educ variable is a string which won't help our model. There are four possibilities SOME COLLEGE(TRADE OR BUSINESS), REFUSED, NOT A HIGH SCHOOL GRAD, HIGH SCHOOL GRAD, COLLEGE GRAD AND BEYOND. Following the example of the voting dataset I think I will convert these into numbers which represent the number of years of formal education completed by the respondant. I could introduce these as categorical variables but I think there is some inherant order which could be interesting to have in my model. I will code them as follows. This variable I will call educ_num.

NA REFUSED 5 NOT A HIGH SCHOOL GRAD 12 HIGH SCHOOL GRAD 14 SOME COLLEGE(TRADE OR BUSINESS) 16 COLLEGE GRAD AND BEYOND

Similarly for income I will code the income as the mean value in the range and NA for refused. For the largest bin I coded as $100,000.

The logistic regression model had three significant variables at the .001 level: aage, educaiton, and black. Three others were significant at the .1 level: dem, female, income.

Taking a look at the specific question of the relationship between race and if one thinks OJ was guilty I can see that since the black variables was significant at the .001 level there is a relationship between one's race and if they think OJ is guilty. The variable for hispanic is not significant in this model. Furthremore since the sign of coeficient for black is negative this means that a black person is less likely to think that OJ is guilty than a non-black, non-hispanic person. I can also see older and richer people think OJ is guilty. Very interesting!

```
fit <- glm(formula = guilt ~ dem + age + educ + female + black + hispanic + income, family = binomial, d
summary(fit)
```

```
##
## Call:
## glm(formula = guilt ~ dem + age + educ + female + black + hispanic +
##     income, family = binomial, data = simpson_data)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.407  -0.535   0.520   0.664   2.272
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.06e-01   4.00e-01   -1.26  0.20614
## dem         -2.96e-01   1.51e-01   -1.97  0.04925 *
## age          1.83e-02   4.65e-03    3.94    8e-05 ***
## educ         9.57e-02   2.51e-02    3.81  0.00014 ***
## female      -3.32e-01   1.51e-01   -2.19  0.02845 *
## black       -2.92e+00   1.94e-01  -15.01  < 2e-16 ***
## hispanic    -1.75e-01   2.60e-01   -0.67  0.50146
## income       7.53e-06   3.21e-06    2.35  0.01889 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1646.2  on 1340  degrees of freedom
## Residual deviance: 1210.1  on 1333  degrees of freedom
##   (228 observations deleted due to missingness)
## AIC: 1226
##
## Number of Fisher Scoring iterations: 4
```

```
mse <- function(model, data) {
  x <- modelr:::residuals(model, data)
  mean(x ^ 2, na.rm = TRUE)
}

simp_val_test <- function(){
  simpson_data_split <- resample_partition(simpson_data, p = c(test = 0.3, train = 0.7))
  # fit <- tree(vote96 ~ mhealth_sum + age + educ + black + female + married + inc10, data =voting_data
  fit <- glm(formula = guilt ~ dem + age + educ + female + black + hispanic + income, family = binomial
  mse(fit, simpson_data_split$test)
}

val_mse <- data_frame(id = 1:1000, mse = map_dbl(id, ~ simp_val_test()))
mean(val_mse$mse)
```

```
## [1] 1.51
```

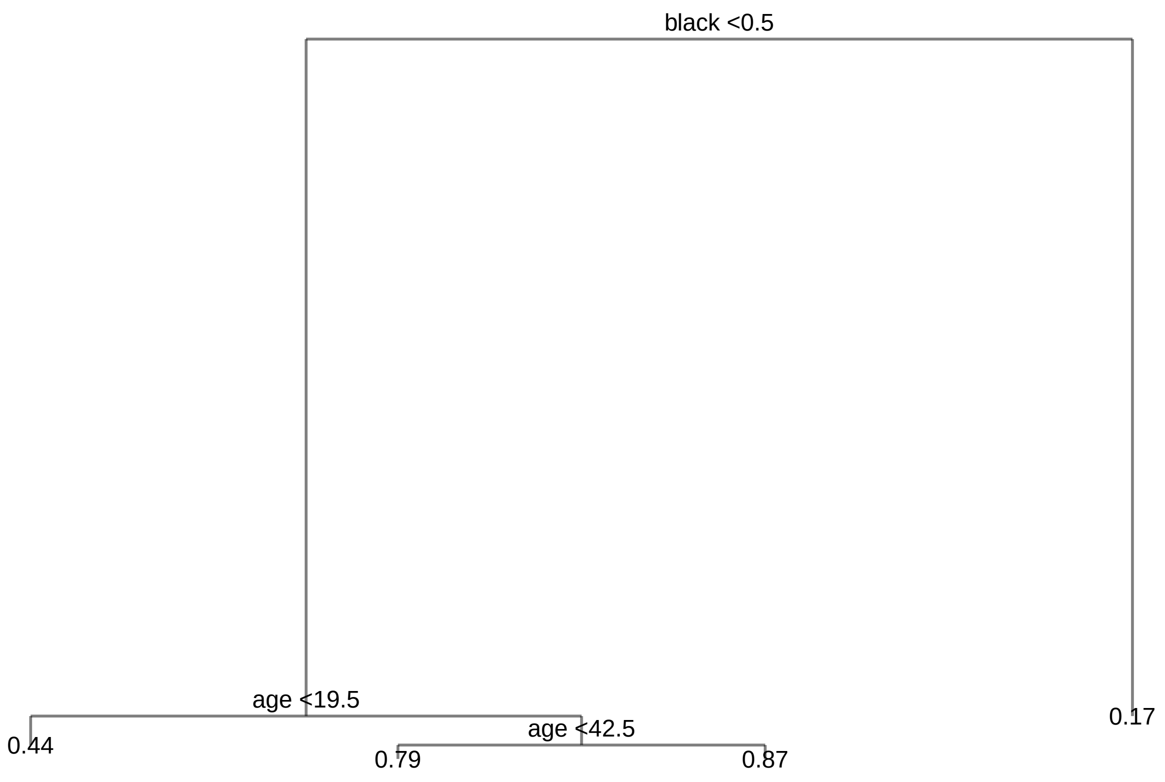
2. Next I will try to predict if someone will think OJ is guilty usin the same dataset. Since I found some interesting relationships with my logistic regression I will use this model as a baseline and try to improve it using a tree-based model. I think a tree-based model will be a good idea because it can easily be

used to predict categorical variables like the one I have here. For all testing I will calculate the average MSE over 1,000 different random 70/30 training test datasets. I have 1569 observations in total so I think this will leave me with ample data to train my model with. From above I find that a logistic regression gives a MSE of 1.5319

```
simpson_tree <- tree(guilt ~ dem + age + educ + female + black + hispanic + income, data = simpson_data
     control = tree.control(nobs = nrow(simpson_data), mindev = 0))

mod <- prune.tree(simpson_tree, best = 4)

# plot tree
tree_data <- dendro_data(mod)

simp_val_test <- function(){
  simpson_data_split <- resample_partition(simpson_data, p = c(test = 0.3, train = 0.7))
  # fit <- tree(vote96 ~ mhealth_sum + age + educ + black + female + married + inc10, data =voting_data

  simpson_tree <- tree(guilt ~ dem + age + educ + female + black + hispanic + income, data = simpson_da
     control = tree.control(nobs = nrow(simpson_data), mindev = 0))

  mod <- prune.tree(simpson_tree, best = 4)

#  fit <- glm(formula = guilt ~ dem + age + educ + female + black + hispanic + income, family = binomia
  mse(mod, simpson_data_split$test)
}

val_mse <- data_frame(id = 1:1000, mse = map_dbl(id, ~ simp_val_test()))
mean(val_mse$mse)
```

```
## [1] 0.146
```

```
ggplot(segment(tree_data)) +
  geom_segment(aes(x = x, y = y, xend = xend, yend = yend),
               alpha = 0.5) +
  geom_text(data = label(tree_data),
            aes(x = x, y = y, label = label_full), vjust = -0.5, size = 3) +
  geom_text(data = leaf_label(tree_data),
            aes(x = x, y = y, label = label), vjust = 0.5, size = 3) +
  theme_dendro()
```

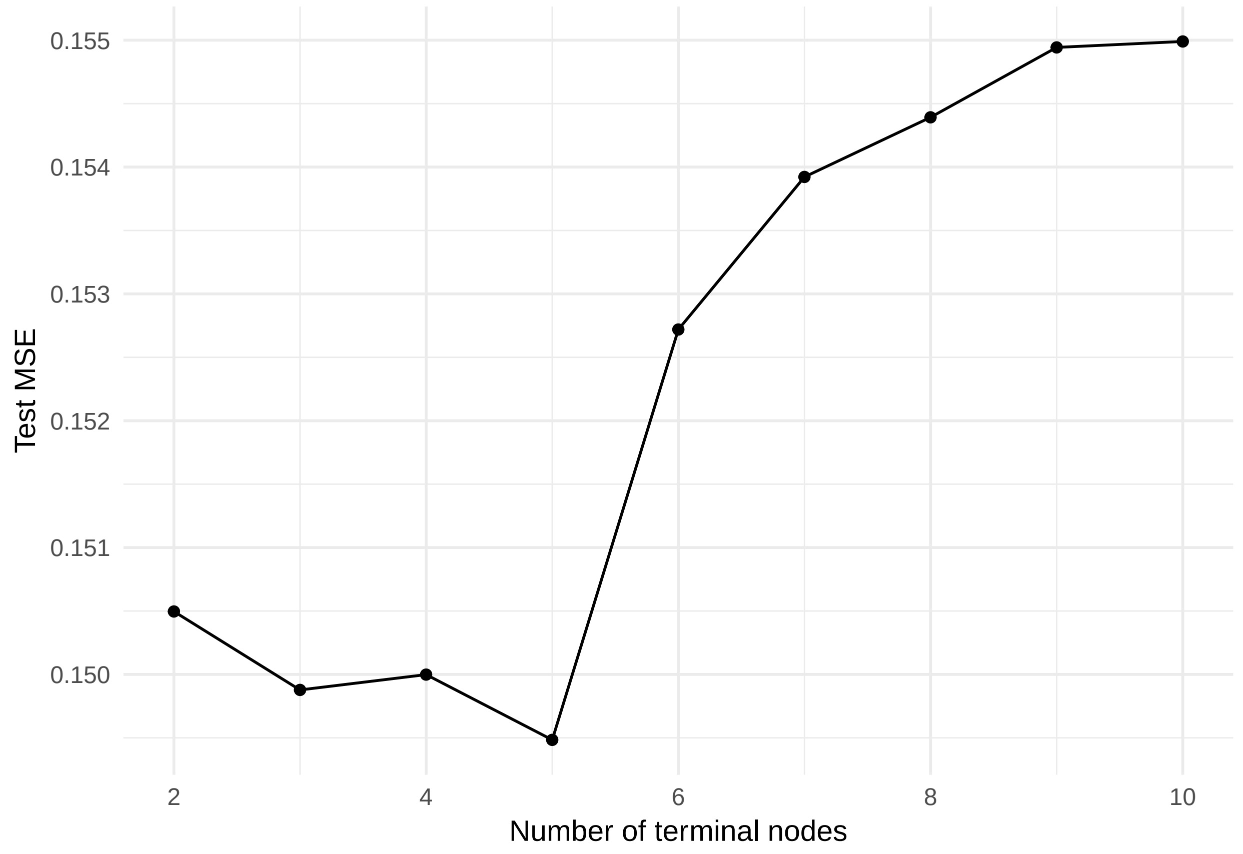


Next I try to find the optimal number of tree branches by using 10-fold CV trees across the whole dataset. I plotted the average MSE and number of terminal branches below. It can be seen that 5 branches produced the lowest MSE (around .148). I plotted this tree below.

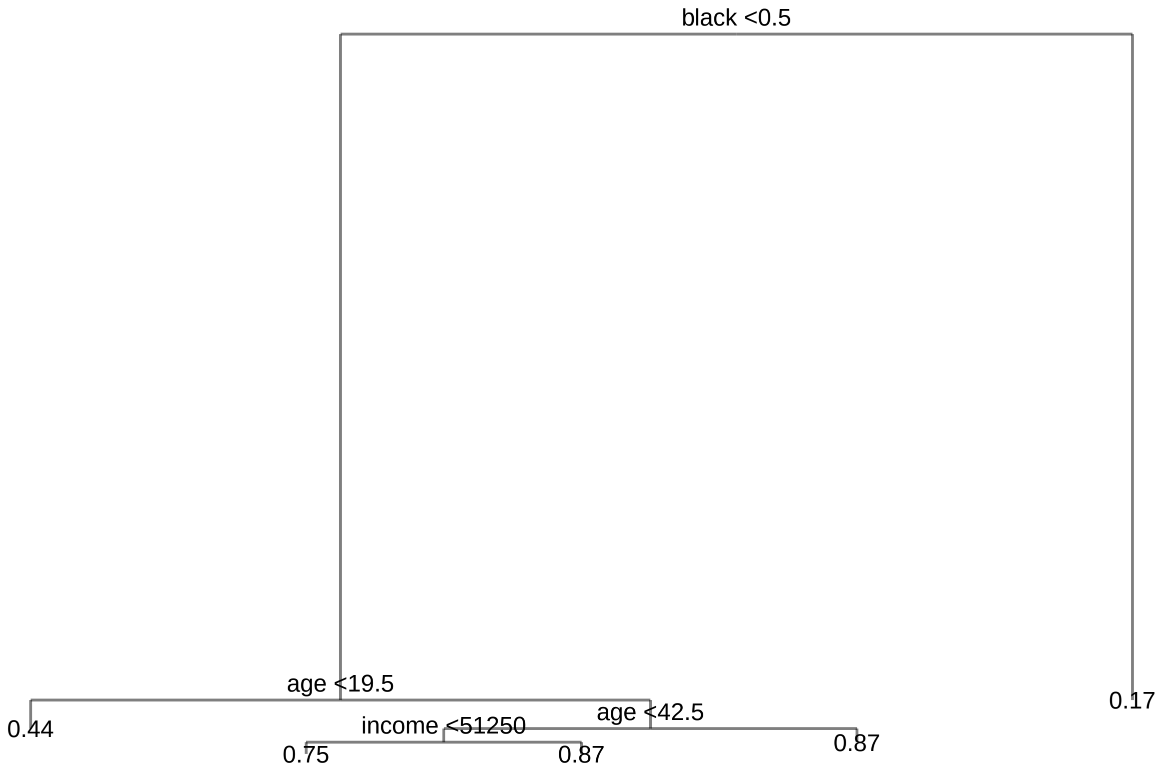
```r
# generate 10-fold CV trees
simpson_cv <- crossv_kfold(simpson_data, k = 10) %>%
  mutate(tree = map(train, ~ tree(guilt ~ dem + age + educ + female + black + hispanic + income, data =
    control = tree.control(nobs = nrow(simpson_data), mindev = 0))))

# calculate each possible prune result for each fold
simpson_cv <- expand.grid(simpson_cv$.id, 2:10) %>%
  as_tibble() %>%
  mutate(Var2 = as.numeric(Var2)) %>%
  rename(.id = Var1,
         k = Var2) %>%
  left_join(simpson_cv) %>%
  mutate(prune = map2(tree, k, ~ prune.tree(.x, best = .y)),
         mse = map2_dbl(prune, test, mse))
```

```
## Joining, by = ".id"

## Warning in left_join_impl(x, y, by$x, by$y, suffix$x, suffix$y): joining
## character vector and factor, coercing into character vector
```

```r
simpson_cv %>%
  select(k, mse) %>%
  group_by(k) %>%
  summarize(test_mse = mean(mse),
            sd = sd(mse, na.rm = TRUE)) %>%
  ggplot(aes(k, test_mse)) +
  geom_point() +
  geom_line() +
  labs(x = "Number of terminal nodes",
       y = "Test MSE")
```



```
mod <- prune.tree(simpson_tree, best = 5)

# plot tree
tree_data <- dendro_data(mod)
ggplot(segment(tree_data)) +
  geom_segment(aes(x = x, y = y, xend = xend, yend = yend),
               alpha = 0.5) +
  geom_text(data = label(tree_data),
            aes(x = x, y = y, label = label_full), vjust = -0.5, size = 3) +
  geom_text(data = leaf_label(tree_data),
            aes(x = x, y = y, label = label), vjust = 0.5, size = 3) +
  theme_dendro()
```



Lastly I calculate the MSE from this model and get 0.1456716. This is an improvement over the logistic regression. :)