Baldeep Dhada - 38150397

2023-12-05

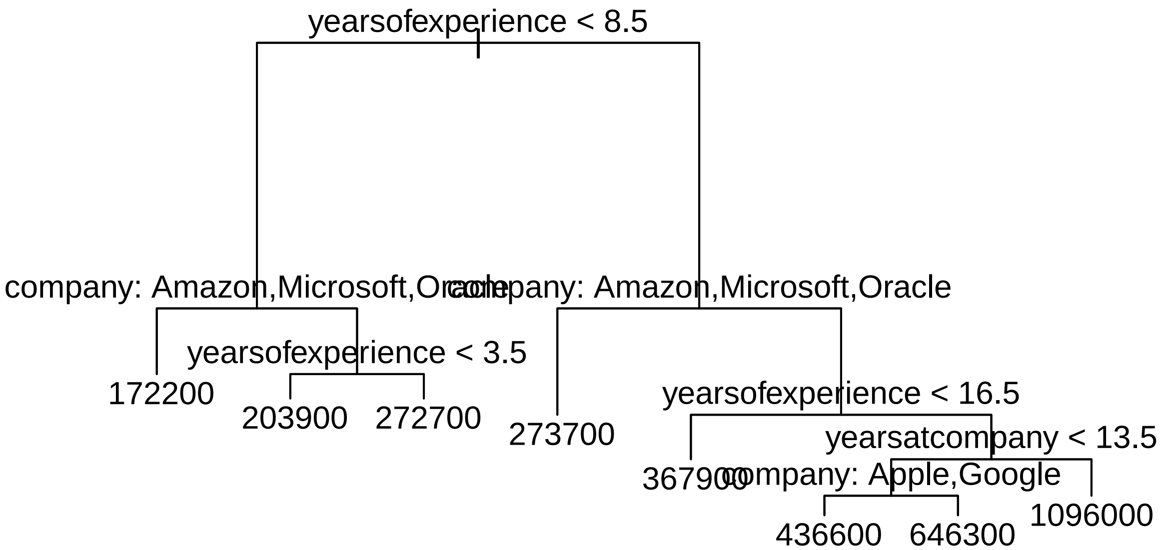
1 a)

Create a regression tree for a data science/STEM salaries given the remainder of the variables in the data set. Provide the tree, including labels — using the command text(treename, pretty=0) will provide a (somewhat) more understandable split labelling for the questions that follow.

```
library(tree)
datasalaries <- read.csv("datasalaries.csv", stringsAsFactors=TRUE)
head(datasalaries)
```

```
##     company totalyearlycompensation yearsofexperience yearsatcompany gender
## 1    Google                  400000                 5              5   Male
## 2 Microsoft                  136000                 3              2   Male
## 3    Google                  337000                 6              6   Male
## 4 Microsoft                  222000                 4              4   Male
## 5    Amazon                  310000                15              3   Male
## 6    Amazon                  620000                19              7   Male
##          Race         Education
## 1       Asian               PhD
## 2 Two Or More Bachelor's Degree
## 3       Asian Bachelor's Degree
## 4       Asian   Master's Degree
## 5       Asian Bachelor's Degree
## 6       Asian Bachelor's Degree
```

```
datasalaries_tree <- tree(totalyearlycompensation~., data=datasalaries)
plot(datasalaries_tree)
text(datasalaries_tree, pretty=0)
```



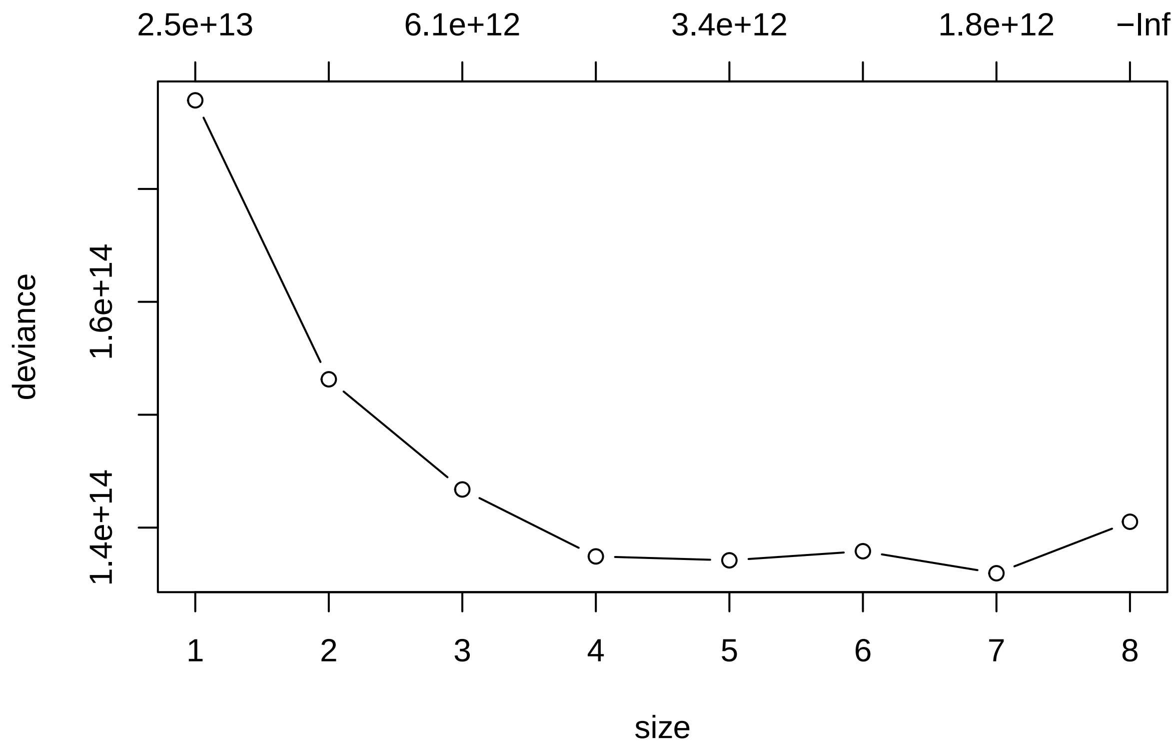
1 b) Based only on the tree outputted in the previous question, which companies included in this data set would you prefer to work for? Why?

I would prefer to work at a company thats not Amazon, Microsoft, Oracle, Apple, or Google because they offer the highest salaries based on the data on the tree. The branch on the left of Apple/Google offers 436600 and the one on the right offers 646300 which is a higher salary.

1 c)

Using set.seed(51341), perform 10-fold cross-validation using cv.tree. Plot the resulting object. How many terminal nodes does cross-validation suggest?

```
set.seed(51341)
cv.datasalaries_tree <- cv.tree(datasalaries_tree, K = 10)
plot(cv.datasalaries_tree, type="b")
```

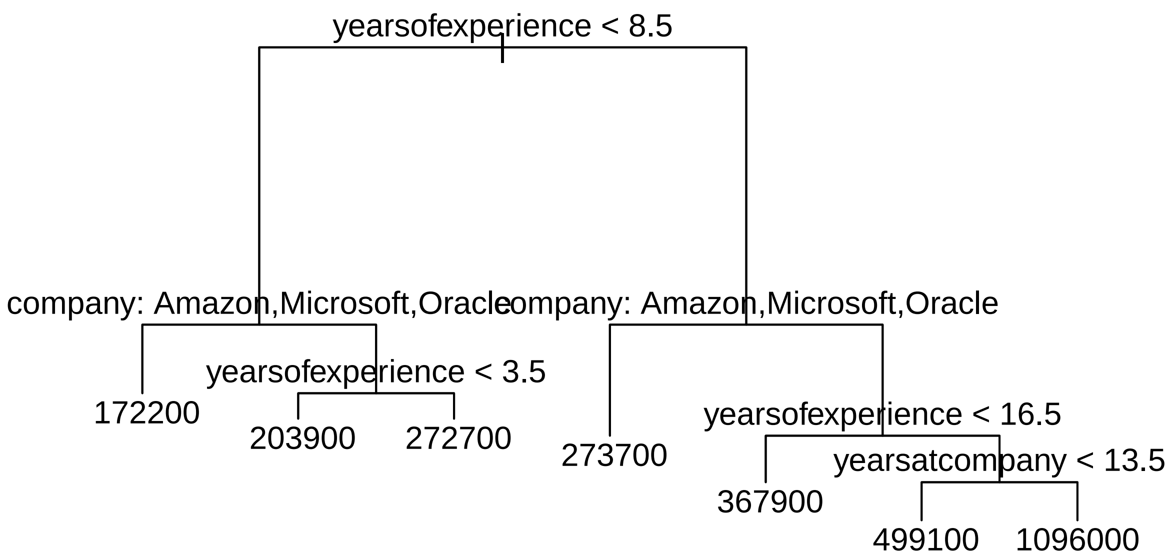


Cross Validation suggests 7 terminal nodes

1 d)

Prune your original tree. Give the predicted salary for a self-identified Asian female with a PhD working at Google with 10 years of experience and 10 years at that company. Use the predict() function to do this, but PLEASE double-check with your tree diagram and brain. My warning is to pay careful attention to how the character vectors are factored, and note that you will have to setup the entry as a 'data.frame'. You will likely find this finicky. . . but it is good practice for real life data science messiness.

```
p.datasalaries_tree <- prune.tree(datasalaries_tree, best=7)
plot(p.datasalaries_tree)
text(p.datasalaries_tree, pretty=0)
```




```r
new_data <- data.frame(
  company = factor("Google", levels = levels(datasalaries$company)),
  yearsofexperience = 10,
  yearsatcompany = 10,
  gender = factor("Female", levels = levels(datasalaries$gender)),
  Race = factor("Asian", levels = levels(datasalaries$Race)),
  Education = factor("PhD", levels = levels(datasalaries$Education))
)

predicted_salary <- predict(p.datasalaries_tree, newdata = new_data)
cat("The predicted salary is:", predicted_salary)
```

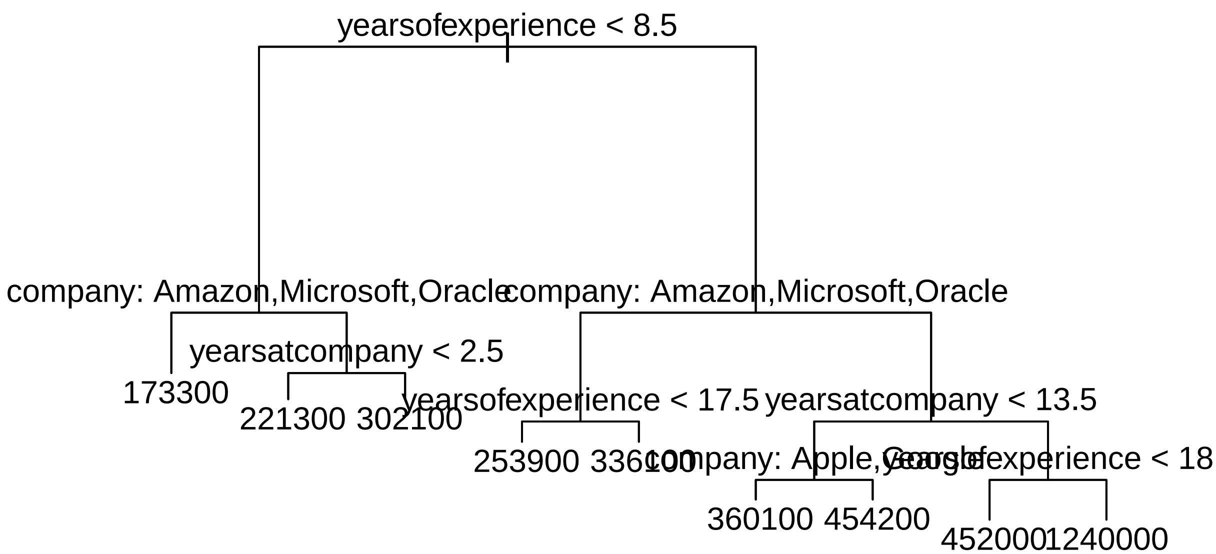
```
## The predicted salary is: 367866.8
```

#1 e) Use the following commands to setup a training and testing set:

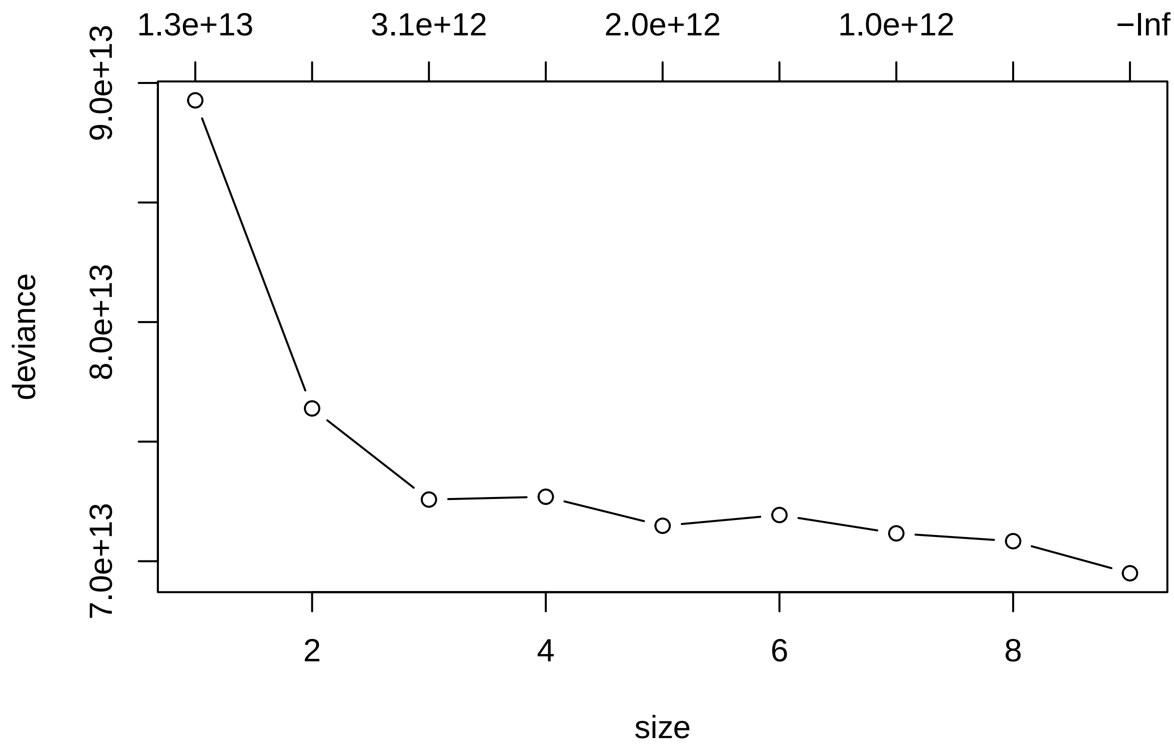
```r
set.seed(763)
dsindex <- sample(1:nrow(datasalaries), 4000)
dstrain <- datasalaries[dsindex, ]
dstest <- datasalaries[-dsindex, ]
```

Now fit a tree to the training set, prune via 10-fold CV, and once again give the predicted salary for the individual from part (d) via the predict function.

```r
dstrain.tree <- tree(totalyearlycompensation~., data=dstrain)
dstest.tree <- tree(totalyearlycompensation~., data=dstest)
plot(dstrain.tree)
text(dstrain.tree, pretty=0)
```

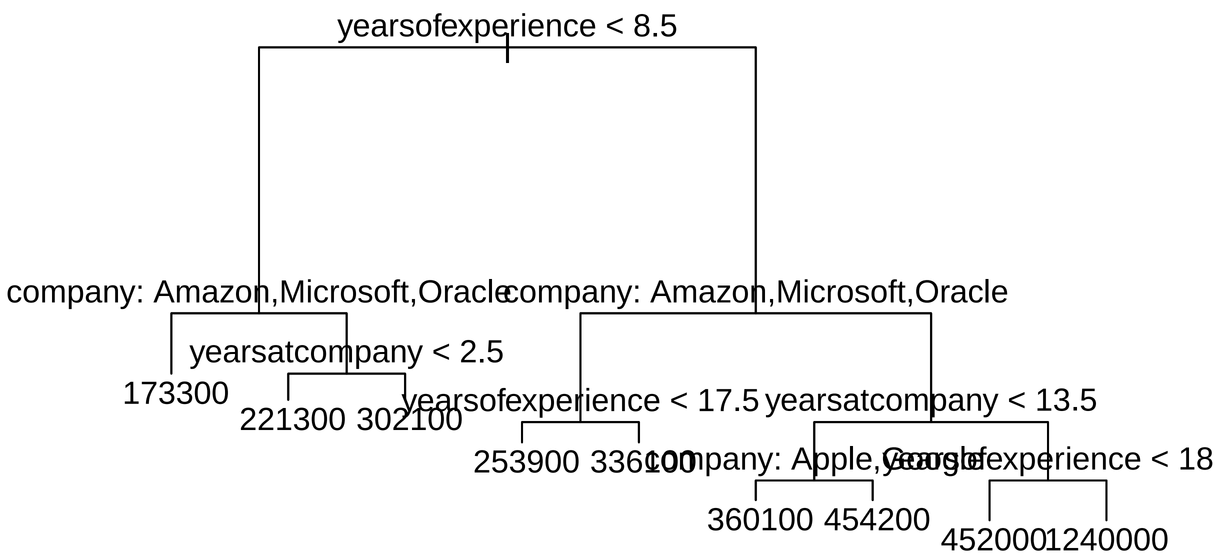



```r
cv.dstrain.tree <- cv.tree(dstrain.tree, K = 10)
plot(cv.dstrain.tree, type="b")
```




```
cv.dstest.tree <- cv.tree(dstest.tree, K = 10)
```

```
p.dstrain.tree <- prune.tree(dstrain.tree, best=9)
p.dstest.tree <- prune.tree(dstest.tree, best = 9)
plot(p.dstrain.tree)
text(p.dstrain.tree, pretty=0)
```




```r
predicted_salary_dstrain <- predict(p.dstrain.tree, newdata = new_data)
cat("The predicted salary for the training set is:", predicted_salary_dstrain, "\n")
```

```
## The predicted salary for the training set is: 360079.9
```

```r
predicted_salary_dstest <- predict(p.dstrain.tree, newdata = dstest)
```

#1 f) Provide the estimated MSE of the model in part (e) — that is, calculate the MSE of the test set. Is the MSE of the test set close to the expected MSE from the 10-fold CV from question (c)?

```r
# MSE for part c

MSE_org <- min(cv.datasalaries_tree$dev)/length(datasalaries$totalyearlycompensation)
MSE_org
```

```
## [1] 18087026057
```

```r
# MSE for test set
MSE_test <- mean((dstest$totalyearlycompensation - predicted_salary_dstest)^2)
MSE_test
```

```
## [1] 19324364849
```

They are close to one another, the testing MSE is higher.

#2) Here I'll run you through some code that could seem aggravating/confusing at first. Pause and consider what you're asking the computer to do for each of these estimates of the MSE. Which estimate is more believable as a long-run estimate of the MSE? Why?

```
library(randomForest)
```

```
## randomForest 4.7-1.1
```

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

```
library(gclus)
```

```
## Loading required package: cluster
```

```
data(body)
set.seed(02139)
bodrun <- randomForest(Weight~Height+Gender, data=body)
bodrun
```

```
##
## Call:
##  randomForest(formula = Weight ~ Height + Gender, data = body)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 1
##
##           Mean of squared residuals: 82.15451
##                     % Var explained: 53.78
```

```
MSE1 <- bodrun$mse[500]
MSE2 <- sum((body$Weight-predict(bodrun))^2)/length(body$Weight)
MSE3 <- sum((body$Weight-predict(bodrun, newdata=body))^2)/length(body$Weight)
MSE1
```

```
## [1] 82.15451
```

```
MSE2
```

```
## [1] 82.15451
```

```
MSE3
```

```
## [1] 74.8239
```

The first mean squared error (MSE) is directly obtained from the random forest model, while the second one is manually calculated using the same formula. Interestingly, the third MSE is computed based on the dataset used for training the model. It is noticeable that the third estimate of MSE is lower than the first two. In evaluating these estimates, the one derived directly from the model summary or by computing residuals and predictions on the original dataset (second approach) appears more reliable as a long-term measure of MSE. This is attributed to its ability to gauge the model's performance on unseen data, which is crucial for assessing its generalization to new and unseen observations. On the contrary, the third estimate, using the same data for prediction as used in training, tends to underestimate the true error. This is because the model has already learned from that particular dataset during training, potentially leading to overfitting and not accurately reflecting its performance on new and diverse data.

#3)

On Canvas (in the assignment area), you will find a data set (insurance.csv) on individuals' health insurance charges in the US along with some demographic information. Note that the data includes both categorical and numeric measures. Provide a thorough regression analysis attempting to predict the 'charges' variable using the remainder of the predictors in the data set. At minimum, trees, boosting, linear models, random forests, and lasso should be used… with appropriate diagnostics, sensible training/testing split, cross-validation, etc. Which model is most likely to provide the lowest MSE in the long-run? Which model would you choose if you were consulting with an insurance company on this data set? If they don't match, explain why.

```
insurance <- read.csv("insurance.csv", stringsAsFactors=TRUE)
head(insurance)
```

```
##   age    sex    bmi children smoker    region   charges
## 1  19 female 27.900        0    yes southwest 16884.924
## 2  18   male 33.770        1     no southeast  1725.552
## 3  28   male 33.000        3     no southeast  4449.462
## 4  33   male 22.705        0     no northwest 21984.471
## 5  32   male 28.880        0     no northwest  3866.855
## 6  31 female 25.740        0     no southeast  3756.622
```

Linear Model:

```
is.index <- sample(1:nrow(insurance), 0.7 * nrow(insurance))
is.train <- insurance[is.index, ]
is.test <- insurance[-is.index, ]
```

```
train.insurance.lm <- lm(charges~., data = is.train)
test.insurance.lm <- lm(charges~., data = is.test)

train.predictions <- predict(train.insurance.lm, newdata = is.train)

train.residuals <- is.train$charges - train.predictions

train.mse <- mean(train.residuals^2)

test.predictions <- predict(test.insurance.lm, newdata = is.test)

test.residuals <- is.test$charges - test.predictions

test.mse <- mean(test.residuals^2)

# Display MSE for training and test sets
cat("Training MSE:", train.mse, "\n")
```

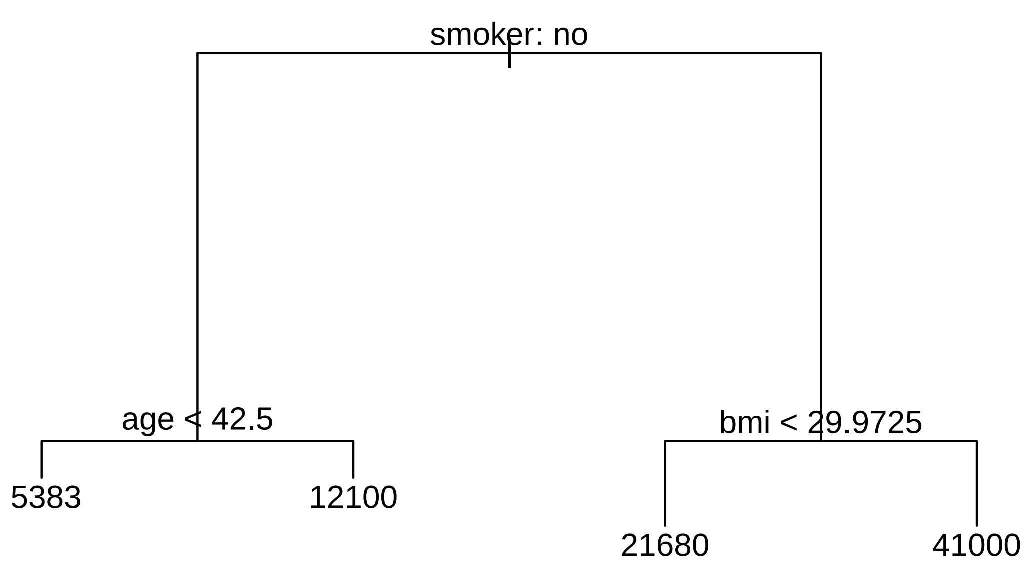
```
## Training MSE: 35373024
```

```r
cat("Test MSE:", test.mse, "\n")
```

```
## Test MSE: 38597290
```

Trees:

```r
insurance_tree <- tree(charges~., data=is.train)
plot(insurance_tree)
text(insurance_tree, pretty=0)
```



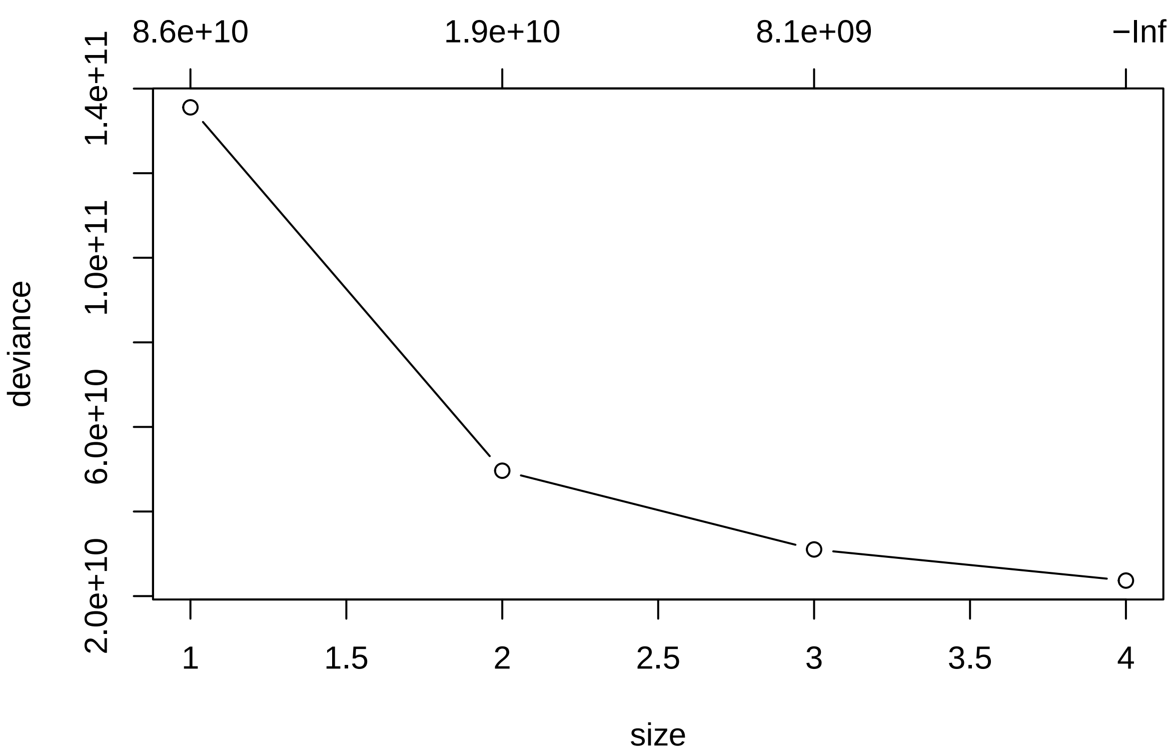

```r
summary(insurance_tree)
```

```
##
## Regression tree:
## tree(formula = charges ~ ., data = is.train)
## Variables actually used in tree construction:
## [1] "smoker" "age"    "bmi"
## Number of terminal nodes:  4
## Residual mean deviance:  24300000 = 2.265e+10 / 932
```

```
## Distribution of residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   -8854   -3027   -1035       0    1122   24480
```

Residual mean deviance = 19830000

```r
cv.insurance_tree <- cv.tree(insurance_tree, K = 10)
plot(cv.insurance_tree, type="b")
```



Pruning is not necessary

```r
training_tree_MSE <- min(cv.insurance_tree$dev)/nrow(insurance)
cat("The Training MSE is", training_tree_MSE)
```

```
## The Training MSE is 17693726
```

```r
test_predict <- predict(insurance_tree, newdata = is.test)
test_MSE <- mean((is.test$charges - test_predict)^2)
cat( "The Testing MSE is", test_MSE)
```

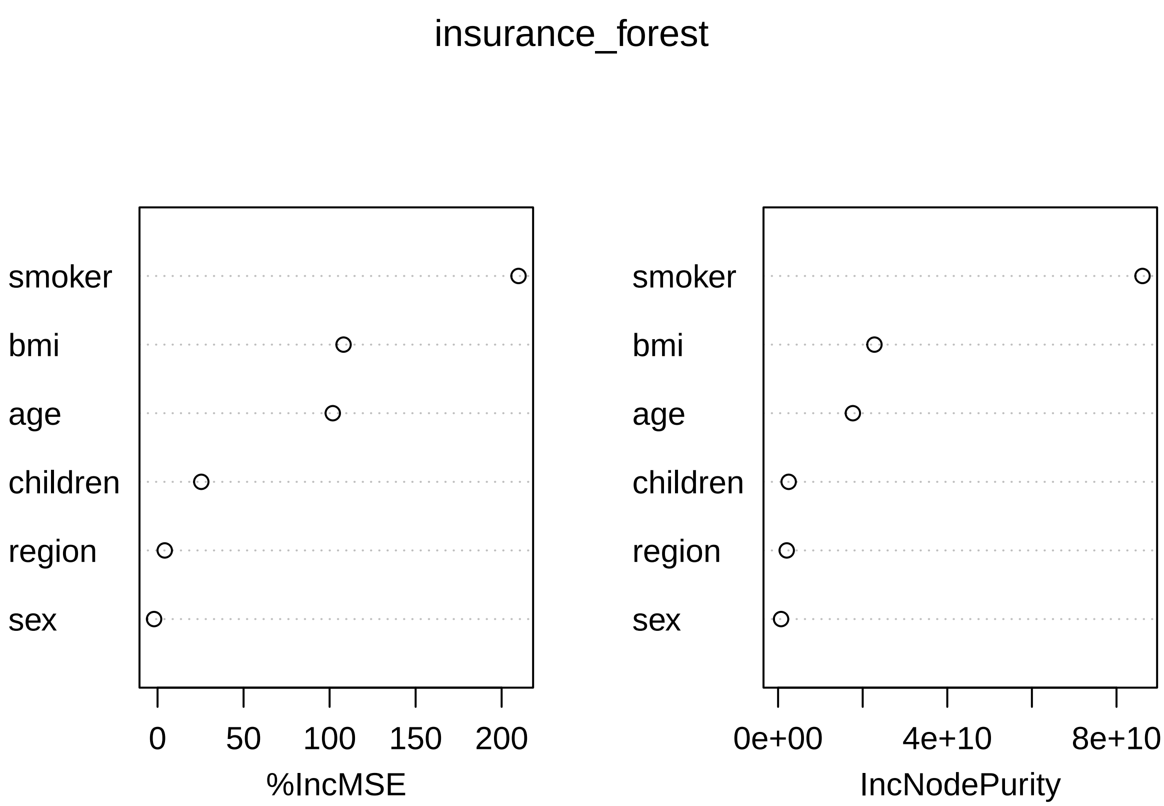
```
## The Testing MSE is 29004119
```

Random Forest

```r
library("randomForest")
insurance_forest <- randomForest(charges~., data=is.train, mtry=4, importance=TRUE)
print(insurance_forest)
```

```
##
## Call:
##  randomForest(formula = charges ~ ., data = is.train, mtry = 4,      importance = TRUE)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##           Mean of squared residuals: 21135832
##                     % Var explained: 85.39
```

```r
varImpPlot(insurance_forest)
```




```r
insurance_forest_train_MSE <- insurance_forest$mse[500]
cat("The training MSE is", insurance_forest_train_MSE)
```

```
## The training MSE is 21135832
```

```r
insurance_forest_predict <- predict(insurance_forest, newdata = is.test)
insurance_test_MSE <- mean((is.test$charges - insurance_forest_predict)^2)
cat("The testing MSE is", insurance_test_MSE)
```

```
## The testing MSE is 24901400
```

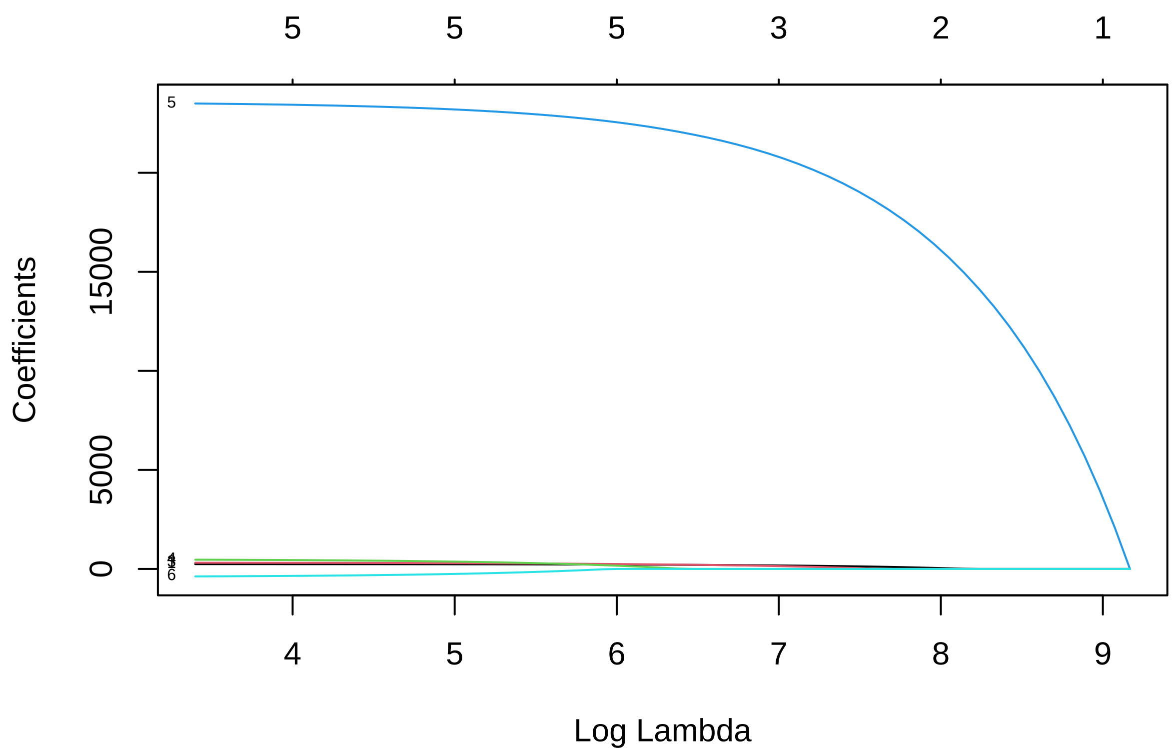
Lasso

```r
library(glmnet)
```

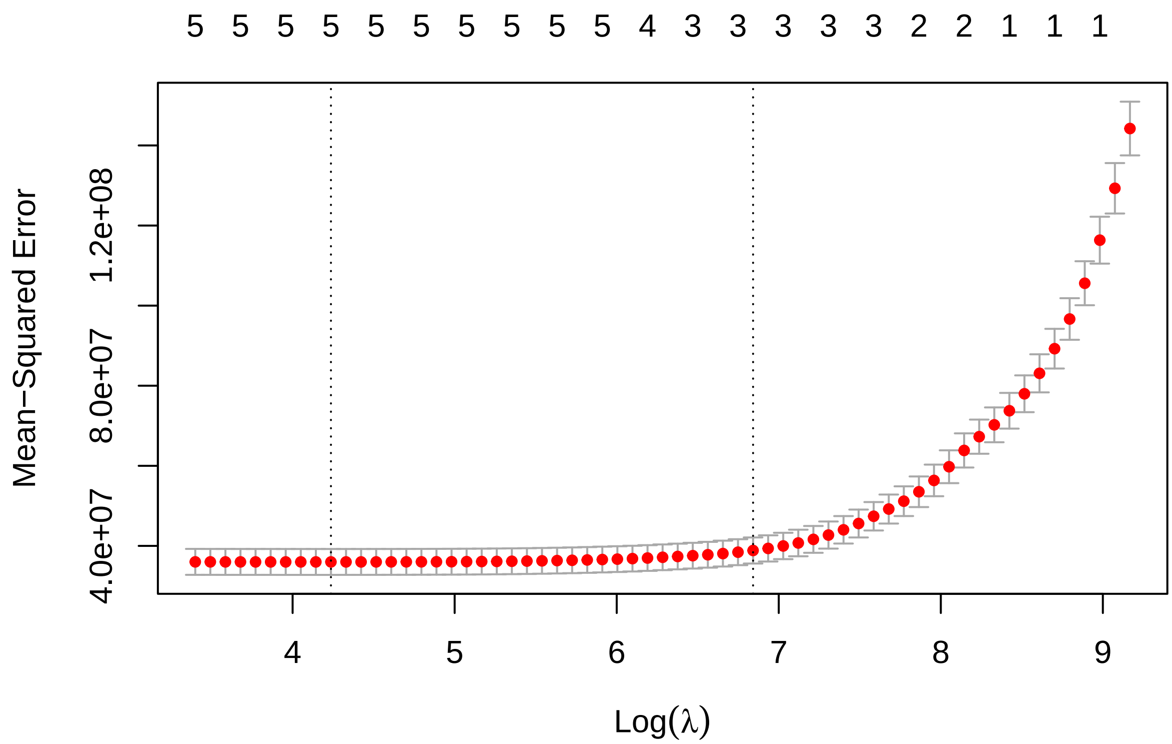
```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-8
```

```r
x <- data.matrix(is.train[,c('age', 'sex', 'bmi', 'children', 'smoker', 'region')])
y <- is.train$charges
insurance_lasso <- cv.glmnet(x, y, alpha=1)
plot(insurance_lasso$glmnet.fit, label=TRUE, xvar="lambda")
```



```r
plot(insurance_lasso)
```




```r
lambda_value <- insurance_lasso$lambda.min
lambda_value
```

```
## [1] 69.16733
```

```r
library(glmnet)

lasso_prediction <- predict(insurance_lasso, s="lambda.min", newx=data.matrix(is.test[,c('age', 'sex',

lasso_MSE <- mean((lasso_prediction - is.test$charges)^2)
lasso_MSE
```

```
## [1] 39523161
```

Boosting

```r
library(gbm)
```

```
## Loaded gbm 2.1.8.1
```

```r
insurance_boosting <- gbm(charges ~ ., data = is.train, distribution = "gaussian", n.trees = 5000, cv.fo
insurance_boosting
```

```
## gbm(formula = charges ~ ., distribution = "gaussian", data = is.train,
##     n.trees = 5000, interaction.depth = 2, shrinkage = 0.1, cv.folds = 10)
## A gradient boosted model with gaussian loss function.
## 5000 iterations were performed.
## The best cross-validation iteration was 83.
## There were 6 predictors of which 6 had non-zero influence.
```

```r
insurance_boosting_predictions <- predict(insurance_boosting, newdata = is.test)
```

```
## Using 83 trees...
```

```r
boosting_MSE <- mean((is.test$charges - insurance_boosting_predictions)^2)
cat("The testing MSE is", boosting_MSE)
```

```
## The testing MSE is 23368149
```

Explanation:

The lowest testing MSE is given by the Boosting model. This is the best model and it will provide the lowest MSE in the long run. The decision tree is the best model to choose if consulting with an insurance company because its the most simple one and the easiest to explain however its not the most reliable model.