HKS SUP-135 Lab 6: Predicting Social Mobility using Trees and Regression

Matt Khinda

3/31/2023

Question 1: Why Divide Data?

Dividing data allows us to both train a model on a large enough sample size while also reserving the ability to test it on out-of-sample data it has never seen before. Doing this acts as a barometer for how the model would likely perform with new data for which we don't have the validating information. If we did not divide the data, and instead trained a model on the full sample, it would end up perfectly predicting the the training data but being useless in the real world.

Question 2: Random Assignment

```r
HUID <- 41531460
set.seed(HUID)
```

2a: Setting the Random Seed

```r
# assign random number
mobility$rand_num <- runif(length(mobility$cz))
mobility$train_flag <- ifelse(mobility$rand_num>= 0.5, 1, 0)
```

2b: Assigning Data to Training Set The training dataset contains 364 observations, while the test dataset contains 377.

Question 3: Subsetting the Data

```r
training_set <- subset(mobility, mobility$train_flag == 1)
test_set <- subset(mobility, mobility$train_flag ==0)
```

Question 4: Using Linear Regression to Predict Upward Mobility

```r
model_1 <- lm(kfr_pooled_pooled_p25 ~ mean_commutetime2000
              + poor_share2000 + popdensity2000, data=training_set)
# Display model results
summary(model_1)
```

4a: Multivariate Regression

```
##
## Call:
## lm(formula = kfr_pooled_pooled_p25 ~ mean_commutetime2000 + poor_share2000 +
##     popdensity2000, data = training_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -25.5901  -2.9916  -0.2726   2.6887  23.9151
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)           6.129e+01  1.457e+00  42.071  < 2e-16 ***
## mean_commutetime2000 -7.245e-01  6.290e-02 -11.518  < 2e-16 ***
## poor_share2000       -1.732e+01  4.845e+00  -3.575 0.000398 ***
## popdensity2000        1.551e-03  9.185e-04   1.689 0.092147 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.079 on 360 degrees of freedom
## Multiple R-squared:  0.3158, Adjusted R-squared:  0.3101
## F-statistic: 55.38 on 3 and 360 DF,  p-value: < 2.2e-16
```

```r
#subset data for milwaukee
milwaukee <- subset(mobility, cz == 24100)
#use previous model to predict mobility
milwaukee_pred = 61.29 +
                 (-0.7245*milwaukee$mean_commutetime2000) +
                 (-17.32*milwaukee$poor_share2000) +
                 (0.001551*milwaukee$popdensity2000)
#calc prediction error
mil_pred_error = milwaukee$kfr_pooled_pooled_p25 - milwaukee_pred
```

4b: Predicting Mobility in Milwaukee, WI Using the multivariate regression above to predict economic mobility in Milwaukee, WI results in a prediciton error of -4.5169863.

```r
y_train_predictions_ols <- predict(model_1, newdata=training_set)
```

4c: OLS Predicitons for Training Data

```
y_test_predictions_ols <- predict(model_1, newdata=test_set)
```

4d: OLS Predicitons for Test Data

```
OLS_performance_train <- (training_set$kfr_pooled_pooled_p25 - y_train_predictions_ols)^2
OLS_performance_test <- (test_set$kfr_pooled_pooled_p25 - y_test_predictions_ols)^2

# calculate the root mean squared prediction error
rmspe_train_ols <- sqrt(mean(OLS_performance_train, na.rm=TRUE))
rmspe_test_ols <- sqrt(mean(OLS_performance_test, na.rm=TRUE))
```

4e: RMSPE in Training and Test Datasets The root mean squared prediction error in the training dataset is 5.0513186, while in the test dataset it is 5.1642457.

4f: Comparing Prediction Errors When comparing the root mean squared prediction errors, we can see that the difference is minimal. This suggests that the model predicts almost equally well for the training and the test datasets, meaning that it has not been overfit.

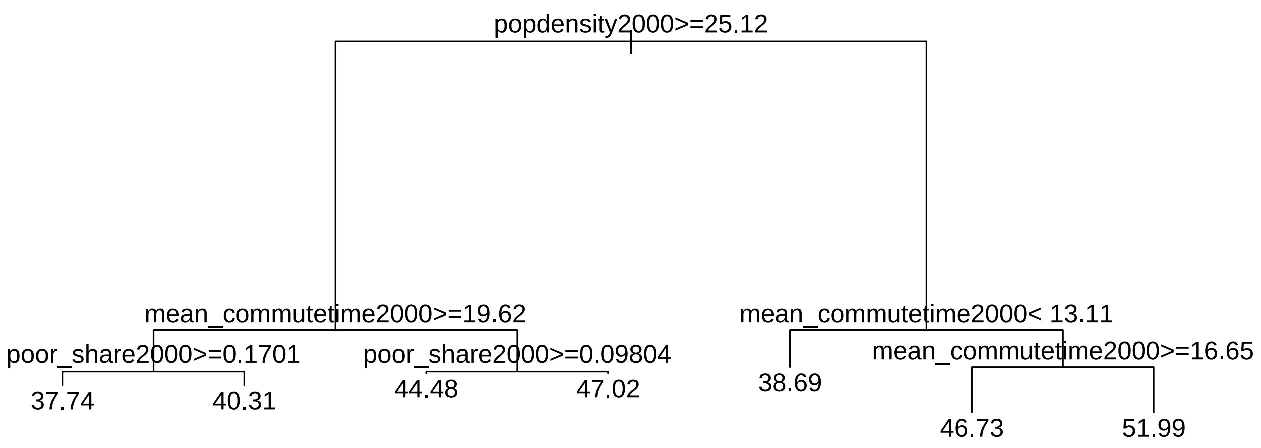
Question 5: Using a Decision Tree to Predict Upward Mobility

```
mobilitytree <- rpart(kfr_pooled_pooled_p25 ~ mean_commutetime2000 +
                      poor_share2000 + popdensity2000,
                      data=training_set,
                      maxdepth = 3,
                      cp=0)
```

5a: Creating the Decision Tree

```
# Visualize tree with labels
plot(mobilitytree, margin = 0.2)
text(mobilitytree, cex = 1)
```

5b: Visualizing the Tree and Using it for Milwaukee Prediction



Based on the values for Milwaukee, WI we can follow the binary tree to arrive at a predicted value. To do that, first we look at popdensity2000 which for Milwaukee is 575.3881836 so we follow the tree to the left because it is greater than the threshold. Then we look at mean_commutetime2000 which for Milwaukee is 23.5812187 so we again follow the tree to the left because it is greater than the threshold. Finally, we look at poor_share2000 which for Milwaukee is 0.0977476 and go to the right this time because it is below the threshold. This path leads us to the predicted value of 40.31 which

```
y_train_predictions_tree <- predict(mobilitytree, newdata=training_set)
```

5c: Decision Tree Predictions for Training Dataset

```
y_test_predictions_tree <- predict(mobilitytree, newdata=test_set)
```

5d: Decision Tree Predictions for Test Dataset

```
tree_performance_train <- (training_set$kfr_pooled_pooled_p25 - y_train_predictions_tree)^2
tree_performance_test <- (test_set$kfr_pooled_pooled_p25 - y_test_predictions_tree)^2

# calculate the root mean squared prediction error
rmspe_train_tree <- sqrt(mean(tree_performance_train, na.rm=TRUE))
rmspe_test_tree <- sqrt(mean(tree_performance_test, na.rm=TRUE))
```

5e: The root mean squared prediction error in the training dataset is 4.1133228, while in the test dataset it is 4.6326436.

5f: RMSPE in Training and Test Datasets When comparing the root mean squared prediction errors from the decision tree, we can see that the prediction error for the test is slightly larger than for the training set.

Question 6: Overfitting

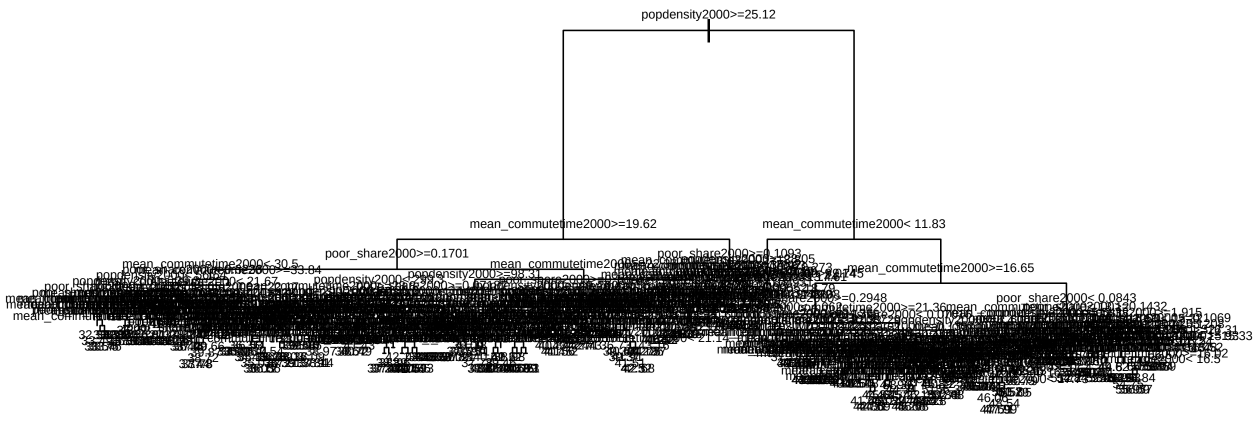
```r
big_tree <-rpart(kfr_pooled_pooled_p25 ~ mean_commutetime2000 +
                 poor_share2000 + popdensity2000,
                 data=training_set,
                 maxdepth = 30,
                 cp=0,
                 minsplit = 1,
                 minbucket = 1)


y_train_predictions_big_tree <- predict(big_tree, newdata=training_set)
y_test_predictions_big_tree <- predict(big_tree, newdata=test_set)

big_tree_performance_train <- (training_set$kfr_pooled_pooled_p25 - y_train_predictions_big_tree)^2
big_tree_performance_test <- (test_set$kfr_pooled_pooled_p25 - y_test_predictions_big_tree)^2

# calculate the root mean squared prediction error
rmspe_train_big_tree <- sqrt(mean(big_tree_performance_train, na.rm=TRUE))
rmspe_test_big_tree <- sqrt(mean(big_tree_performance_test, na.rm=TRUE))

# Visualize tree with labels
plot(big_tree, margin = 0.2)
text(big_tree, cex = 0.5)
```



In this tree of depth 30, the root mean squared prediction error in the training dataset is 0, while in the test dataset it is 5.1007488. Here we can see that the model perfectly predicts the training data, but has a larger prediction error in the test data than the smaller tree used above. This is an example of overfitting.

Question 7: Comparing Predictive Models

By looking at the root mean squared prediction errors of the three different models used, we can determine that, unsurprisingly, the decision tree of depth 30 was the most accurate predictor for the training dataset. However, the decision tree of depth 3 was the best predictor for the data in the test set.