Assignment2

Coffy Andrews-Guo

2023-03-28

## The GOOD, The BAD & The Ugly of Using Decision Trees

[*GitHub Source Code*](https://github.com/candrewxs/D622)

### Introduction

Decision trees are a machine learning algorithm used for classification and regression tasks. They are supervised learning algorithms that build a model in a tree, where each internal node represents a decision based on a feature, and each leaf node represents a predicted outcome or a decision. As illustrated in Figure 1, it shows a decision tree-like structure.

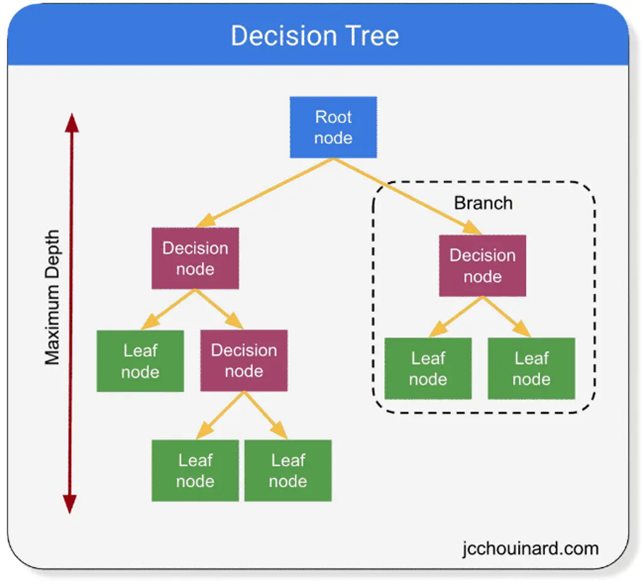


Figure 1: Structure of a decision tree

There are several types of decision tree algorithms based on the different programming languages, including:

1. ID3 (Iterative Dichotomiser 3): This algorithm builds a decision tree by selecting the feature that provides the most information gain at each step.
2. C4.5: This algorithm is an extension of ID3 that can handle both continuous and categorical features. It also uses an information gain ratio instead of information gain to handle biases towards features with many values.
3. CART (Classification and Regression Trees): This algorithm can be used for classification and regression tasks. It builds binary trees, where each internal node represents a split on a feature, and each leaf node represents a predicted value.
4. Random Forest: This algorithm is an ensemble learning method that uses multiple decision trees to improve accuracy and reduce overfitting.
5. Gradient Boosted Decision Trees: This algorithm builds a decision tree ensemble by iteratively adding trees that minimize the residual errors.

In the Kaggle dataset, [Shop Customer Data](https://www.kaggle.com/datasets/datascientistanna/customers-dataset?select=Customers.csv), a business owner is looking to understand its customers. The customer’s information is retrieved from their membership cards. We aim to build a decision tree algorithm to determine a qualitative response (class outcome) using a classification tree.

* In Part I, a classification tree model is fitted using two different features and compares results.
* In Part II, a Random Forest algorithm is fitted with a regression tree model, and we analyze the results.
* Then, in Part III, a brief discussion on handling the “The GOOD, The BAD & The Ugly of Using Decision Tree”.

### PART I: Building a Classification Tree Model

We use the CART decision tree algorithm to analyze the **Shop Customer Data** dataset. The dataset has 2000 observations and 8 variables. We are interested in predicting *gender* and *annual\_income* based on the other variables in the dataset. Predicting an outcome on the customer’s behavior from a selected characteristic (the dependent variable or class) and how likely they make a purchase.

customerid gender age annual\_income   
 Min. : 1.0 Length:2000 Min. : 0.00 Min. : 0   
 1st Qu.: 500.8 Class :character 1st Qu.:25.00 1st Qu.: 74572   
 Median :1000.5 Mode :character Median :48.00 Median :110045   
 Mean :1000.5 Mean :48.96 Mean :110732   
 3rd Qu.:1500.2 3rd Qu.:73.00 3rd Qu.:149093   
 Max. :2000.0 Max. :99.00 Max. :189974   
 spending\_score profession work\_experience family\_size   
 Min. : 0.00 Length:2000 Min. : 0.000 Min. :1.000   
 1st Qu.: 28.00 Class :character 1st Qu.: 1.000 1st Qu.:2.000   
 Median : 50.00 Mode :character Median : 3.000 Median :4.000   
 Mean : 50.96 Mean : 4.103 Mean :3.768   
 3rd Qu.: 75.00 3rd Qu.: 7.000 3rd Qu.:5.000   
 Max. :100.00 Max. :17.000 Max. :9.000

The response variables for the classification decision tree algorithms are: **gender** and **annual\_income**.

In Table 2, Summary Statistics, each response variable of interest is a different type: qualitative and quantitative. The gender variable is a categorical variable that measures the quantity of each individual, whereas the annual\_income is a quantitative variable whose values are either countable or have infinite possibilities. Now, we perform the following pre-processing steps on the dataset.

* Decision Tree | Classification with *gender variable and annual\_income*
* The dataset was prepared and manipulated to resolve issues such as:
  + Changing the column names to lowercase
  + Changing the column names to remove white space between words
  + Changing numeric variables that are known to be factor variables
  + Treating inconsistencies, where the minimum values for *age* are illogical. All individuals eligible for membership must be at or above 13. Therefore, values less than 13 are treated as missing data by setting them to NA.

customerid gender age annual\_income   
 Min. : 1.0 Female:1186 Min. :13.0 Min. : 0   
 1st Qu.: 500.8 Male : 814 1st Qu.:32.0 1st Qu.: 74572   
 Median :1000.5 Median :54.0 Median :110045   
 Mean :1000.5 Mean :54.6 Mean :110732   
 3rd Qu.:1500.2 3rd Qu.:77.0 3rd Qu.:149093   
 Max. :2000.0 Max. :99.0 Max. :189974   
 NA's :230   
 spending\_score profession work\_experience family\_size   
 Min. : 0.00 Artist :612 Min. : 0.000 Min. :1.000   
 1st Qu.: 28.00 Healthcare :339 1st Qu.: 1.000 1st Qu.:2.000   
 Median : 50.00 Entertainment:234 Median : 3.000 Median :4.000   
 Mean : 50.96 Engineer :179 Mean : 4.103 Mean :3.768   
 3rd Qu.: 75.00 Doctor :161 3rd Qu.: 7.000 3rd Qu.:5.000   
 Max. :100.00 (Other) :440 Max. :17.000 Max. :9.000   
 NA's : 35

* Additionally, in **Part II**, the annual\_income variable was recoded from a continuous variable to a categorical variable. The new variable, **high\_income**, takes on a value of Yes if the annual\_income variable exceeds the median value of 110045 and No if not.
* We split the dataset randomly into training and testing subsets in a 75:25 ratio to build each classification model.

#### Modeling

We use the rpart library to build the model and specify “method = class” since we are dealing with a classification dataset.

* 1st target variable, ***"gender***" model:

custhw\_mod1 <- rpart(  
 gender ~ .,   
 method = "class",  
 data = custhw\_train)

* 2nd target variable ***"annual\_income (aka "high\_income")"*** model:

#Creates a new binary variable, high\_income.  
high\_income <- ifelse(custhw$annual\_income <= mean(custhw$annual\_income), "No", "Yes")  
#Add high\_income to the data set.  
custhw1 <- data.frame(custhw, high\_income)  
#Remove the annual\_income variable from the data.  
custhw1 <- custhw1[, -4]  
#Code high\_income as a factor variable  
custhw1$high\_income <- as.factor(custhw1$high\_income)  
class(custhw1$high\_income)

custhw1\_mod <- rpart(  
 high\_income ~ .,   
 method = "class",  
 data = custhw1\_train  
)

#### Classification Rules

Figures 2 & 3 show the **rules** decision tree model used to make the classifications. Nwanganga et al. states that a decision tree model shows the variable (feature) most predictive of the target outcome (or class). Based on our trees (Figures 2 & 3), these are the following top three rules :

* Model 1: target variable ‘gender’

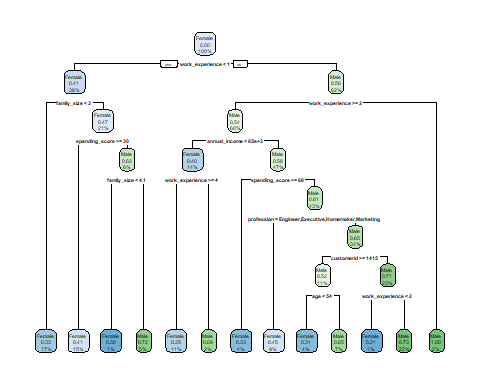
n= 2388   
  
node), split, n, loss, yval, (yprob)  
 \* denotes terminal node  
  
 1) root 2388 1194 Female (0.5000000 0.5000000)   
 2) work\_experience< 1.003925 906 370 Female (0.5916115 0.4083885)   
 4) family\_size< 3.00759 398 130 Female (0.6733668 0.3266332) \*  
 5) family\_size>=3.00759 508 240 Female (0.5275591 0.4724409)   
 10) spending\_score>=29.95927 364 150 Female (0.5879121 0.4120879) \*  
 11) spending\_score< 29.95927 144 54 Male (0.3750000 0.6250000)   
 22) family\_size< 4.057795 33 10 Female (0.6969697 0.3030303) \*  
 23) family\_size>=4.057795 111 31 Male (0.2792793 0.7207207) \*  
 3) work\_experience>=1.003925 1482 658 Male (0.4439946 0.5560054)   
 6) work\_experience>=1.989374 1436 658 Male (0.4582173 0.5417827)   
 12) annual\_income< 84942.5 323 130 Female (0.5975232 0.4024768)   
 24) work\_experience>=3.964972 274 96 Female (0.6496350 0.3503650) \*  
 25) work\_experience< 3.964972 49 15 Male (0.3061224 0.6938776) \*  
 13) annual\_income>=84942.5 1113 465 Male (0.4177898 0.5822102)   
 26) spending\_score>=87.5 92 30 Female (0.6739130 0.3260870) \*  
 27) spending\_score< 87.5 1021 403 Male (0.3947111 0.6052889)   
 54) profession=Engineer,Executive,Homemaker,Marketing 211 95 Female (0.5497630 0.4502370) \*  
 55) profession=Artist,Doctor,Entertainment,Healthcare,Lawyer 810 287 Male (0.3543210 0.6456790)   
 110) customerid>=1414.681 269 128 Male (0.4758364 0.5241636)   
 220) age< 53.5 100 31 Female (0.6900000 0.3100000) \*  
 221) age>=53.5 169 59 Male (0.3491124 0.6508876) \*  
 111) customerid< 1414.681 541 159 Male (0.2939002 0.7060998)   
 222) work\_experience< 2.012046 24 5 Female (0.7916667 0.2083333) \*  
 223) work\_experience>=2.012046 517 140 Male (0.2707930 0.7292070) \*  
 7) work\_experience< 1.989374 46 0 Male (0.0000000 1.0000000) \*

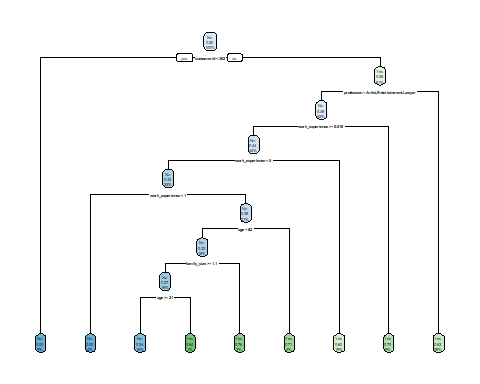
* Model 2: target variable high\_income

n= 2984   
  
node), split, n, loss, yval, (yprob)  
 \* denotes terminal node  
  
 1) root 2984 1492 No (0.500000000 0.500000000)   
 2) customerid< 302.4575 258 1 No (0.996124031 0.003875969) \*  
 3) customerid>=302.4575 2726 1235 Yes (0.453044754 0.546955246)   
 6) profession=Artist,Entertainment,Lawyer 1548 745 No (0.518733850 0.481266150)   
 12) work\_experience>=0.01772836 1321 585 No (0.557153671 0.442846329)   
 24) work\_experience< 4.962301 748 244 No (0.673796791 0.326203209)   
 48) work\_experience< 0.9993159 108 0 No (1.000000000 0.000000000) \*  
 49) work\_experience>=0.9993159 640 244 No (0.618750000 0.381250000)   
 98) age< 81.99102 543 175 No (0.677716390 0.322283610)   
 196) family\_size>=1.088011 489 134 No (0.725971370 0.274028630)   
 392) age>=21 464 113 No (0.756465517 0.243534483) \*  
 393) age< 21 25 4 Yes (0.160000000 0.840000000) \*  
 197) family\_size< 1.088011 54 13 Yes (0.240740741 0.759259259) \*  
 99) age>=81.99102 97 28 Yes (0.288659794 0.711340206) \*  
 25) work\_experience>=4.962301 573 232 Yes (0.404886562 0.595113438) \*  
 13) work\_experience< 0.01772836 227 67 Yes (0.295154185 0.704845815) \*  
 7) profession=Doctor,Engineer,Executive,Healthcare,Homemaker,Marketing 1178 432 Yes (0.366723260 0.633276740) \*

#### Visualizing the Rules

Visualizing the many classifications in a graph provides a simpler view.

**Model 1: target variable - gender** 

**Model 2: target variable - high\_income** 

**The structure of the decision tree tells us:**

* the order in which features are evaluated within the tree are significant and predictive of the final outcome. Nwanaganga et al., states that pathways (or branches) towards the left represent agreement with the split criteria, while pathways toward the right represent disagreement with the split criteria. Let’s review the left pathways.
  + Model 1 (gender response): root node begins when the customer is female, splits when the female has less than one year of work experience (*female = work\_experience < 1*), whose family\_size is less than three (*female = family\_size < 3*), and terminates to predict a *17%* merchandise purchase. The
  + Model 2 (high\_income response): root node begins when the customer do not have a high income, splits when the customer id is less than 302 (*no = customer\_id < 302*), and terminates to predict customers with low income will purchase *5%* of merchandise.

#### Predicting and Evaluating the Models Using the Test Data

We have built the decision tree model with the training data to predict the response variable using the test data. Now, we use the predict() function, pass it in the testing subset, and specify the type as **class**. The first ten results for the models are shown below:

* Model 1: target variable - gender

1 2 3 4 5 6 7 8 9 10   
Female Female Male Female Female Female Female Female Female Female   
Levels: Female Male

* Model 2: target variable - high\_income (annual\_income)

1 3 5 7 8 9 12 14 18 24   
No No No No No No No No No No   
Levels: No Yes

How do we determine these results are good predictions? We can use the widely known **Confusion Matrix** to evaluate classification models.

Let’s take a look at the results:

* **Model 1**: target variable - gender

Actual  
Prediction Female Male  
 Female 214 165  
 Male 69 52

Model 1 generates 214 true negative (0’s), 52 true positives (1’s), while there are 69 false negatives and 165 false positives.

* **Model 2**: target variable - high\_income (annual\_income)

Actual  
Prediction No Yes  
 No 106 27  
 Yes 154 213

Model 2 generates 110 true negative (0’s), 207 true positives (1’s), while there are 150 false negatives and 33 false positives.

**Results Discussion** *The results show that Model 1 has a predictive accuracy of 53.2 percent and Model 2 has a predictive accuracy of 63.8 percent against their respective dataset. However, the high\_income (annual\_income) response variable provides the best performance.*

### PART II: Building a Random Forest Model with a Regression Tree

In a regression model, the decision tree is a popular bagging ensemble method that consists of many decision trees called a forest.

We are revisiting the [Shop Customer Data](https://www.kaggle.com/datasets/datascientistanna/customers-dataset?select=Customers.csv) to predict **annual\_income** based on the other variables in the dataset. Annual\_income is a continuous variable - numeric.

The data is prepared and manipulated to resolve issues indicated in *Part I - Building a Classification Tree Model*. However, the random forest algorithm requires all NA’s to be replaced.

The data has 0 missing values, and the summary statistics as follow:

customerid gender age annual\_income   
 Min. : 1.0 Female:1186 Min. :13.00 Min. : 0   
 1st Qu.: 500.8 Male : 814 1st Qu.:25.00 1st Qu.: 74572   
 Median :1000.5 Median :48.00 Median :110045   
 Mean :1000.5 Mean :49.82 Mean :110732   
 3rd Qu.:1500.2 3rd Qu.:73.00 3rd Qu.:149093   
 Max. :2000.0 Max. :99.00 Max. :189974   
   
 spending\_score profession work\_experience family\_size   
 Min. : 0.00 Artist :612 Min. : 0.000 Min. :1.000   
 1st Qu.: 28.00 Healthcare :339 1st Qu.: 1.000 1st Qu.:2.000   
 Median : 50.00 Entertainment:234 Median : 3.000 Median :4.000   
 Mean : 50.96 Engineer :179 Mean : 4.103 Mean :3.768   
 3rd Qu.: 75.00 Doctor :161 3rd Qu.: 7.000 3rd Qu.:5.000   
 Max. :100.00 Executive :153 Max. :17.000 Max. :9.000   
 (Other) :322

**Split the dataset to train and test.**

The regression trees, use the argument *mtry = (p/3)* variables when growing a random forest.

Call:  
 randomForest(formula = annual\_income ~ ., data = custhw2\_train, mtry = 3, importance = TRUE, ntrees = 50)   
 Type of random forest: regression  
 Number of trees: 500  
No. of variables tried at each split: 3  
  
 Mean of squared residuals: 1588001646  
 % Var explained: 22.95

The number of trees is 500 in the model and no. of variable tried at each split are 3. The model with the lowest test mean squared error (MSE) used 343 trees. The average difference between the predicted value for annual\_income and the actual observed value is 3.9819421^{4}.

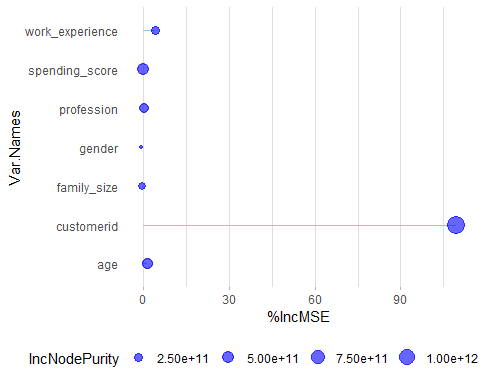
**Visualizing the Random Forest Tree**

A random forests trees are a random subset of the features, that is built on multipledecision trees.

**Variables with the most predictive power**

%IncMSE IncNodePurity  
customerid 109.5470602 1.281382e+12  
gender -0.7795383 5.689156e+10  
age 1.3901320 3.911375e+11  
spending\_score 0.1144302 4.210733e+11  
profession 0.4919933 2.848944e+11  
work\_experience 4.3288175 2.322238e+11  
family\_size -0.1915973 1.910287e+11

In the above table, the variable of importance is reported as “(1) the mean decrease of accuracy in predictions on the out of bag samples when a given variable is permuted, and (2) a measure of th total decrease in node impurity that results from splits over that variable, averaged over all trees” (James, G. et al).

Let’s view the plot of these important measures. We see that across all of the tree in the random forest model, the customerid, and work\_experience are the top most important variables.

### PART III: Solving Real Problems with Decision Trees

In the article, [The Good, The Bad, & The Ugly](https://decizone.com/blog/the-good-the-bad-the-ugly-of-using-decision-trees), perceptions on how to resolve the **bad and ugly** aspects of decision trees involves several steps. Explaining the decision-making process to stakeholders, will help build trust in the model and provide transparency into the decision-making process. Addressing concerns such as the potential bias or inaccuracies to reduce concerns and build confidence in the model. It is important to provide ongoing support and updates, including training to help identify any issues and make improvements over time.

### References

[R for Statistical Learning](https://daviddalpiaz.github.io/r4sl/trees.html)

[An Introduction to Statistical Learning with Applications in R](https://hastie.su.domains/ISLR2/ISLRv2_website.pdf)

[An Introduction to Recursive Partitioning Using the rpart Routines](https://cran.r-project.org/web/packages/rpart/vignettes/longintro.pdf)

[Plotting rpart trees with the rpart.plot package](http://www.milbo.org/doc/prp.pdf)

[KoalaTea | Random Forest in R](https://koalatea.io/r-random-forest-regression/)

## Appendix: All code for this report

knitr::opts\_chunk$set(echo = FALSE, comment = NA)  
  
#Load required libraries  
suppressPackageStartupMessages({  
 library("tidyverse")  
 library("DMwR") #package for imbalance data  
 library("rpart") #model data  
 library("rpart.plot")  
 library("caret")  
 library("randomForest")  
 library("ggalt")  
 library("data.table")  
 library("reprtree")  
 })  
  
#Load the data  
dfhw<- read\_csv("https://raw.githubusercontent.com/candrewxs/D622/main/Customers.csv")  
  
#View the data  
glimpse(dfhw)  
#Change column names to lower case  
names(dfhw) <- tolower(names(dfhw))  
#Change column names  
names(dfhw)[4] <- "annual\_income"  
names(dfhw)[5] <- "spending\_score"  
names(dfhw)[7] <- "work\_experience"  
names(dfhw)[8] <- "family\_size"  
#View the dataset statistical summary  
summary(dfhw)  
#Change variables, gender and profession, data type to factor  
custhw <- dfhw %>%  
 mutate(gender = as.factor(gender)) %>%  
 mutate(profession = as.factor(profession)) %>%  
 mutate(age = ifelse(age <13, NA, age))  
#View the dataset revised summary statistics  
summary(custhw)  
#Split the data set into a training and testing   
set.seed(1234)  
sample\_set <- sample(nrow(custhw), round(nrow(custhw)\*.75), replace = FALSE)  
custhw\_train <- custhw[sample\_set, ]  
custhw\_test <- custhw[-sample\_set, ]  
  
#View the target distribution (gender) in the dataset  
round(prop.table(table(select(custhw, gender), exclude = NULL)), 4) \* 100  
  
#View the target distribution (gender) in the training set  
round(prop.table(table(select(custhw\_train, gender), exclude = NULL)), 4) \* 100  
  
#View the target (gender) distribution in the testing set  
round(prop.table(table(select(custhw\_test, gender), exclude = NULL)), 4) \* 100  
  
#Use SMOTE function to resolve imbalance data (gender - response variable)  
set.seed(1234)  
custhw\_train <- SMOTE(gender ~ ., data.frame(custhw\_train), perc.over = 100, perc.under = 200)  
  
#View balance data (response variable: gender)  
round(prop.table(table(select(custhw\_train, gender), exclude = NULL)), 4) \* 100  
  
custhw\_mod1 <- rpart(  
 gender ~ .,   
 method = "class",  
 data = custhw\_train)  
  
summary(custhw$annual\_income)  
  
#Creates a new binary variable, high\_income.  
high\_income <- ifelse(custhw$annual\_income <= mean(custhw$annual\_income), "No", "Yes")  
#Add high\_income to the data set.  
custhw1 <- data.frame(custhw, high\_income)  
#Remove the annual\_income variable from the data.  
custhw1 <- custhw1[, -4]  
#Code high\_income as a factor variable  
custhw1$high\_income <- as.factor(custhw1$high\_income)  
class(custhw1$high\_income)  
  
#Split the data set into a training and testing   
set.seed(1234)  
sample2\_set <- sample(nrow(custhw1), round(nrow(custhw1)\*.75), replace = FALSE)  
custhw1\_train <- custhw1[sample2\_set, ]  
custhw1\_test <- custhw1[-sample2\_set, ]  
  
#View the target distribution (high\_income) in the dataset  
round(prop.table(table(select(custhw1, high\_income), exclude = NULL)), 4) \* 100  
  
#View the target distribution (high\_income) in the training set  
round(prop.table(table(select(custhw1\_train, high\_income), exclude = NULL)), 4) \* 100  
  
round(prop.table(table(select(custhw1\_test, high\_income), exclude = NULL)), 4) \* 100  
  
#Use SMOTE function to resolve imbalance data (high\_income - response variable)  
set.seed(1234)  
custhw1\_train <- SMOTE(high\_income ~ ., data.frame(custhw1\_train), perc.over = 100, perc.under = 200)  
  
#View balance data (response variable: high\_income)  
round(prop.table(table(select(custhw1\_train, high\_income), exclude = NULL)), 4) \* 100  
  
custhw1\_mod <- rpart(  
 high\_income ~ .,   
 method = "class",  
 data = custhw1\_train  
)  
  
#Model - Rules for response variable: gender  
custhw\_mod1  
  
#Model - Rules for response variable: high\_income  
custhw1\_mod  
  
#png(file = "Model1.png")  
  
#Plot model1 - response variable "gender"  
rpart.plot(custhw\_mod1)  
  
#Saving the file  
#dev.off()  
  
  
#Plot model2 - response variable "high\_income"  
rpart.plot(custhw1\_mod)  
  
#Predict Model 1 - response variable: gender  
custhw\_pred <- predict(custhw\_mod1, custhw\_test, type = "class")  
head(custhw\_pred, 10)  
  
#Predict Model 2 - response variable: high\_income  
custhw1\_pred <- predict(custhw1\_mod, custhw1\_test, type = "class")  
head(custhw1\_pred, 10)  
  
custhw\_pred\_table <- table(Prediction = custhw\_pred, Actual = custhw\_test$gender)  
custhw\_pred\_table  
  
acc1 <- sum(diag(custhw\_pred\_table)) / nrow(custhw\_test)  
acc1  
  
custhw\_pred\_table2 <- table(Prediction = custhw1\_pred, Actual = custhw1\_test$high\_income)  
custhw\_pred\_table2  
  
acc2 <- sum(diag(custhw\_pred\_table2)) / nrow(custhw1\_test)  
acc2  
  
#Replace NAs with string  
custhw2 <- dfhw %>%  
 mutate(profession = replace(profession, is.na(profession), "Missing"))  
#Change variables, gender and profession, data type to factor  
#Replace age < 13 with a value = 13  
custhw2 <- custhw2 %>%  
 mutate(gender = as.factor(gender)) %>%  
 mutate(profession = as.factor(profession)) %>%  
 mutate(age = ifelse(age <13, 13, age))  
summary(custhw2)  
  
  
#Split the data set into a training and testing   
set.seed(1234)  
sample\_set3 <- sample(nrow(custhw2), round(nrow(custhw2)\*.75), replace = FALSE)  
custhw2\_train <- custhw2[sample\_set3, ]  
custhw2\_test <- custhw2[-sample\_set3, ]  
  
  
set.seed (1234)  
rf\_mod <- randomForest(annual\_income ~ ., data = custhw2\_train,   
 mtry = 3, importance = TRUE, ntrees = 50)  
  
rf\_mod  
  
  
which.min(rf\_mod$mse)  
  
  
sqrt(rf\_mod$mse[which.min(rf\_mod$mse)])  
  
  
options(repos='http://cran.rstudio.org')  
have.packages <- installed.packages()  
cran.packages <- c('devtools','plotrix','randomForest','tree')  
to.install <- setdiff(cran.packages, have.packages[,1])  
if(length(to.install)>0) install.packages(to.install)  
  
library(devtools)  
if(!('reprtree' %in% installed.packages())){  
 install\_github('araastat/reprtree')  
}  
for(p in c(cran.packages, 'reprtree')) eval(substitute(library(pkg), list(pkg=p)))  
  
  
reprtree:::plot.getTree(rf\_mod)  
  
  
importance(rf\_mod)  
  
  
# Get variable importance from the model fit  
ImpData <- as.data.frame(importance(rf\_mod))  
ImpData$Var.Names <- row.names(ImpData)  
  
ggplot(ImpData, aes(x=Var.Names, y=`%IncMSE`)) +  
 geom\_segment( aes(x=Var.Names, xend=Var.Names, y=0, yend=`%IncMSE`), color="skyblue") +  
 geom\_point(aes(size = IncNodePurity), color="blue", alpha=0.6) +  
 theme\_light() +  
 coord\_flip() +  
 theme(  
 legend.position="bottom",  
 panel.grid.major.y = element\_blank(),  
 panel.border = element\_blank(),  
 axis.ticks.y = element\_blank()  
 )