Final Project

Owen Bolick

2024-10-13

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

summary(cars)

## speed dist   
## Min. : 4.0 Min. : 2.00   
## 1st Qu.:12.0 1st Qu.: 26.00   
## Median :15.0 Median : 36.00   
## Mean :15.4 Mean : 42.98   
## 3rd Qu.:19.0 3rd Qu.: 56.00   
## Max. :25.0 Max. :120.00

## Including Plots

You can also embed plots, for example:



Note that the echo = FALSE parameter was added to the code chunk to prevent printing of the R code that generated the plot.

library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.4 ✔ readr 2.1.5  
## ✔ forcats 1.0.0 ✔ stringr 1.5.1  
## ✔ ggplot2 3.5.0 ✔ tibble 3.2.1  
## ✔ lubridate 1.9.3 ✔ tidyr 1.3.1  
## ✔ purrr 1.0.2   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

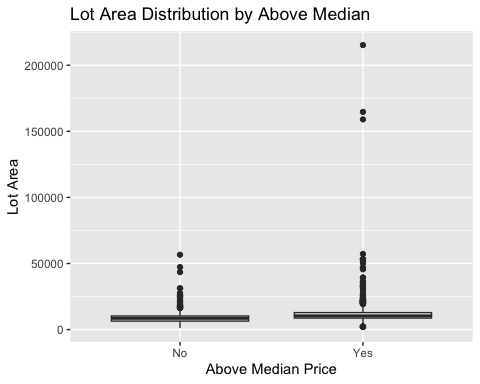
library(readr)  
library(dplyr)  
ames\_student\_1 <- read\_csv("~/Downloads/ames\_student-1.csv")

## Rows: 2053 Columns: 81  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (47): MS\_SubClass, MS\_Zoning, Street, Alley, Lot\_Shape, Land\_Contour, Ut...  
## dbl (34): Lot\_Frontage, Lot\_Area, Year\_Built, Year\_Remod\_Add, Mas\_Vnr\_Area, ...  
##   
## 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.

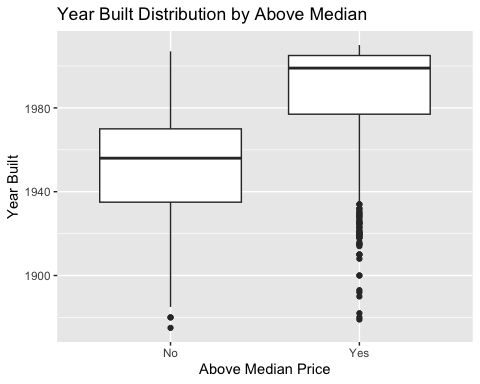
summary(select(ames\_student\_1, where(is.numeric)))

## Lot\_Frontage Lot\_Area Year\_Built Year\_Remod\_Add  
## Min. : 0.00 Min. : 1300 Min. :1875 Min. :1950   
## 1st Qu.: 43.00 1st Qu.: 7500 1st Qu.:1953 1st Qu.:1965   
## Median : 62.00 Median : 9548 Median :1972 Median :1993   
## Mean : 57.38 Mean : 10258 Mean :1971 Mean :1984   
## 3rd Qu.: 78.00 3rd Qu.: 11600 3rd Qu.:2000 3rd Qu.:2004   
## Max. :313.00 Max. :215245 Max. :2010 Max. :2010   
## Mas\_Vnr\_Area BsmtFin\_SF\_1 BsmtFin\_SF\_2 Bsmt\_Unf\_SF   
## Min. : 0.0 Min. :1.00 Min. : 0.00 Min. : 0.0   
## 1st Qu.: 0.0 1st Qu.:3.00 1st Qu.: 0.00 1st Qu.: 226.0   
## Median : 0.0 Median :3.00 Median : 0.00 Median : 460.0   
## Mean : 103.8 Mean :4.21 Mean : 52.57 Mean : 561.2   
## 3rd Qu.: 164.0 3rd Qu.:7.00 3rd Qu.: 0.00 3rd Qu.: 801.0   
## Max. :1600.0 Max. :7.00 Max. :1526.00 Max. :2336.0   
## Total\_Bsmt\_SF First\_Flr\_SF Second\_Flr\_SF Low\_Qual\_Fin\_SF   
## Min. : 0 Min. : 432 Min. : 0.0 Min. : 0.000   
## 1st Qu.: 793 1st Qu.: 882 1st Qu.: 0.0 1st Qu.: 0.000   
## Median : 988 Median :1088 Median : 0.0 Median : 0.000   
## Mean :1055 Mean :1168 Mean : 326.1 Mean : 4.973   
## 3rd Qu.:1304 3rd Qu.:1402 3rd Qu.: 701.0 3rd Qu.: 0.000   
## Max. :5095 Max. :5095 Max. :1862.0 Max. :1064.000   
## Gr\_Liv\_Area Bsmt\_Full\_Bath Bsmt\_Half\_Bath Full\_Bath   
## Min. : 480 Min. :0.0000 Min. :0.00000 Min. :0.000   
## 1st Qu.:1137 1st Qu.:0.0000 1st Qu.:0.00000 1st Qu.:1.000   
## Median :1447 Median :0.0000 Median :0.00000 Median :2.000   
## Mean :1499 Mean :0.4301 Mean :0.05796 Mean :1.564   
## 3rd Qu.:1737 3rd Qu.:1.0000 3rd Qu.:0.00000 3rd Qu.:2.000   
## Max. :5095 Max. :3.0000 Max. :2.00000 Max. :4.000   
## Half\_Bath Bedroom\_AbvGr Kitchen\_AbvGr TotRms\_AbvGrd   
## Min. :0.0000 Min. :0.000 Min. :1.000 Min. : 3.000   
## 1st Qu.:0.0000 1st Qu.:2.000 1st Qu.:1.000 1st Qu.: 5.000   
## Median :0.0000 Median :3.000 Median :1.000 Median : 6.000   
## Mean :0.3751 Mean :2.855 Mean :1.047 Mean : 6.442   
## 3rd Qu.:1.0000 3rd Qu.:3.000 3rd Qu.:1.000 3rd Qu.: 7.000   
## Max. :2.0000 Max. :6.000 Max. :3.000 Max. :15.000   
## Fireplaces Garage\_Cars Garage\_Area Wood\_Deck\_SF   
## Min. :0.000 Min. :0.000 Min. : 0 Min. : 0.00   
## 1st Qu.:0.000 1st Qu.:1.000 1st Qu.: 320 1st Qu.: 0.00   
## Median :1.000 Median :2.000 Median : 478 Median : 0.00   
## Mean :0.603 Mean :1.774 Mean : 472 Mean : 93.52   
## 3rd Qu.:1.000 3rd Qu.:2.000 3rd Qu.: 576 3rd Qu.: 168.00   
## Max. :4.000 Max. :5.000 Max. :1488 Max. :1424.00   
## Open\_Porch\_SF Enclosed\_Porch Three\_season\_porch Screen\_Porch   
## Min. : 0.00 Min. : 0.00 Min. : 0.000 Min. : 0.00   
## 1st Qu.: 0.00 1st Qu.: 0.00 1st Qu.: 0.000 1st Qu.: 0.00   
## Median : 27.00 Median : 0.00 Median : 0.000 Median : 0.00   
## Mean : 48.17 Mean : 23.02 Mean : 2.799 Mean : 16.68   
## 3rd Qu.: 72.00 3rd Qu.: 0.00 3rd Qu.: 0.000 3rd Qu.: 0.00   
## Max. :742.00 Max. :584.00 Max. :407.000 Max. :576.00   
## Pool\_Area Misc\_Val Mo\_Sold Year\_Sold   
## Min. : 0.000 Min. : 0.00 Min. : 1.000 Min. :2006   
## 1st Qu.: 0.000 1st Qu.: 0.00 1st Qu.: 4.000 1st Qu.:2007   
## Median : 0.000 Median : 0.00 Median : 6.000 Median :2008   
## Mean : 1.339 Mean : 60.12 Mean : 6.189 Mean :2008   
## 3rd Qu.: 0.000 3rd Qu.: 0.00 3rd Qu.: 8.000 3rd Qu.:2009   
## Max. :800.000 Max. :17000.00 Max. :12.000 Max. :2010   
## Longitude Latitude   
## Min. :-93.69 Min. :41.99   
## 1st Qu.:-93.66 1st Qu.:42.02   
## Median :-93.64 Median :42.03   
## Mean :-93.64 Mean :42.03   
## 3rd Qu.:-93.62 3rd Qu.:42.05   
## Max. :-93.58 Max. :42.06

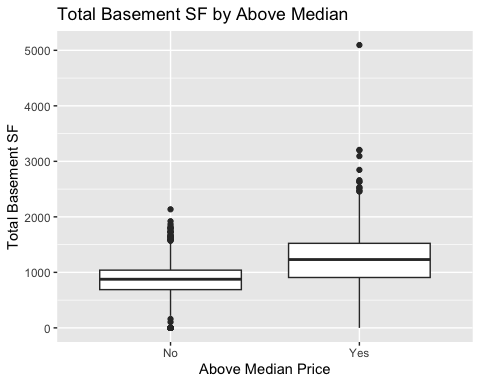
ggplot(ames\_student\_1, aes(x = Above\_Median, y = Lot\_Area)) +  
 geom\_boxplot() +  
 labs(title = "Lot Area Distribution by Above Median",   
 x = "Above Median Price",   
 y = "Lot Area")



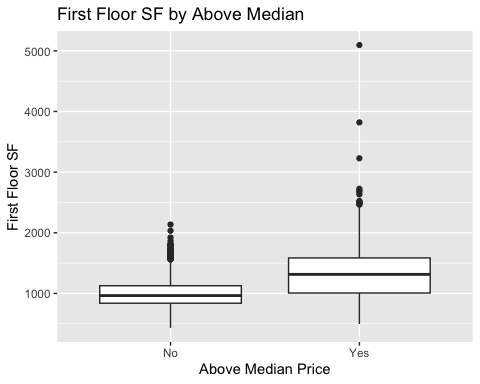
ggplot(ames\_student\_1, aes(x = Above\_Median, y = Year\_Built)) +  
 geom\_boxplot() +  
 labs(title = "Year Built Distribution by Above Median",   
 x = "Above Median Price",   
 y = "Year Built")



ggplot(ames\_student\_1, aes(x = Above\_Median, y = Total\_Bsmt\_SF)) +  
 geom\_boxplot() +  
 labs(title = "Total Basement SF by Above Median",   
 x = "Above Median Price",   
 y = "Total Basement SF")



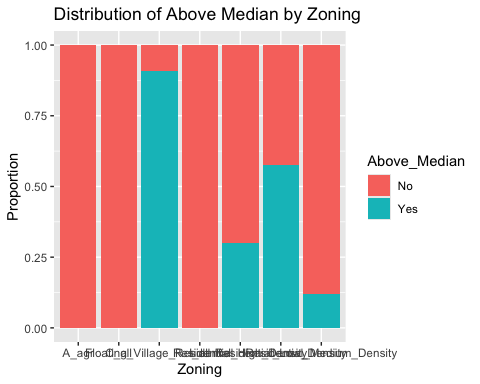
ggplot(ames\_student\_1, aes(x = Above\_Median, y = First\_Flr\_SF)) +  
 geom\_boxplot() +  
 labs(title = "First Floor SF by Above Median",   
 x = "Above Median Price",   
 y = "First Floor SF")



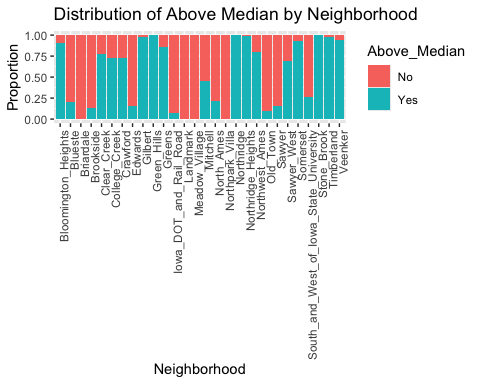
ames\_student\_1$Above\_Median\_Num <- ifelse(ames\_student\_1$Above\_Median == "Yes", 1, 0)  
  
num\_vars <- select(ames\_student\_1, where(is.numeric))  
cor\_matrix <- cor(num\_vars, use = "complete.obs")  
  
cor\_matrix[,"Above\_Median\_Num"]

## Lot\_Frontage Lot\_Area Year\_Built Year\_Remod\_Add   
## 0.091920232 0.189377364 0.582288565 0.529246315   
## Mas\_Vnr\_Area BsmtFin\_SF\_1 BsmtFin\_SF\_2 Bsmt\_Unf\_SF   
## 0.301026422 -0.095715517 0.001246641 0.183641060   
## Total\_Bsmt\_SF First\_Flr\_SF Second\_Flr\_SF Low\_Qual\_Fin\_SF   
## 0.441421509 0.430708299 0.260966734 -0.061578721   
## Gr\_Liv\_Area Bsmt\_Full\_Bath Bsmt\_Half\_Bath Full\_Bath   
## 0.565688190 0.141372151 -0.014161064 0.588631443   
## Half\_Bath Bedroom\_AbvGr Kitchen\_AbvGr TotRms\_AbvGrd   
## 0.297206579 0.111896015 -0.166102822 0.375236118   
## Fireplaces Garage\_Cars Garage\_Area Wood\_Deck\_SF   
## 0.440745636 0.560091912 0.494627712 0.293820057   
## Open\_Porch\_SF Enclosed\_Porch Three\_season\_porch Screen\_Porch   
## 0.271078175 -0.162041135 0.015245812 0.108880884   
## Pool\_Area Misc\_Val Mo\_Sold Year\_Sold   
## 0.014874614 0.017703377 0.015321303 -0.022151039   
## Longitude Latitude Above\_Median\_Num   
## -0.284912353 0.214819594 1.000000000

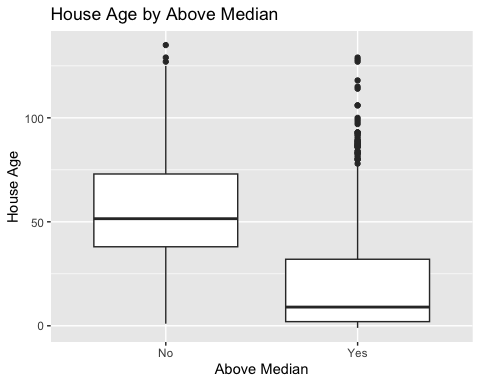
ggplot(ames\_student\_1, aes(x = MS\_Zoning, fill = Above\_Median)) +  
 geom\_bar(position = "fill") +  
 labs(title = "Distribution of Above Median by Zoning",   
 x = "Zoning",   
 y = "Proportion")



ggplot(ames\_student\_1, aes(x = Neighborhood, fill = Above\_Median)) +  
 geom\_bar(position = "fill") +  
 labs(title = "Distribution of Above Median by Neighborhood",   
 x = "Neighborhood",   
 y = "Proportion") +  
 theme(axis.text.x = element\_text(angle = 90, hjust = 1))



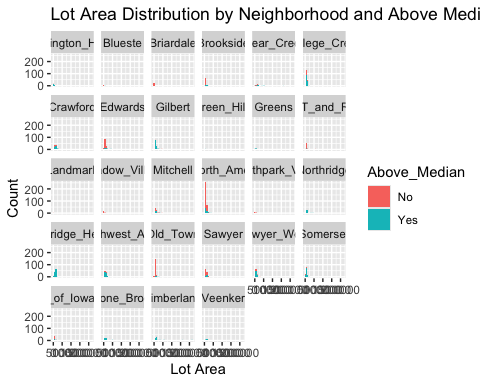
ames\_student\_1$House\_Age <- ames\_student\_1$Year\_Sold - ames\_student\_1$Year\_Built  
  
ggplot(ames\_student\_1, aes(x = Above\_Median, y = House\_Age)) +  
 geom\_boxplot() +  
 labs(title = "House Age by Above Median",   
 x = "Above Median",   
 y = "House Age")



missing\_data <- colSums(is.na(ames\_student\_1))  
missing\_data[missing\_data > 0]

## named numeric(0)

ggplot(ames\_student\_1, aes(x = Lot\_Area, fill = Above\_Median)) +  
 geom\_histogram(bins = 30) +  
 facet\_wrap(~Neighborhood) +  
 labs(title = "Lot Area Distribution by Neighborhood and Above Median",   
 x = "Lot Area",   
 y = "Count")



library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

set.seed(123)  
  
train\_index <- createDataPartition(ames\_student\_1$Above\_Median, p = 0.8, list = FALSE)  
train\_data <- ames\_student\_1[train\_index, ]  
test\_data <- ames\_student\_1[-train\_index, ]  
  
dim(train\_data)

## [1] 1643 83

dim(test\_data)

## [1] 410 83

train\_data$MS\_Zoning <- as.factor(train\_data$MS\_Zoning)  
train\_data$Neighborhood <- as.factor(train\_data$Neighborhood)  
train\_data$Above\_Median <- as.factor(train\_data$Above\_Median)  
  
test\_data$MS\_Zoning <- as.factor(test\_data$MS\_Zoning)  
test\_data$Neighborhood <- as.factor(test\_data$Neighborhood)  
test\_data$Above\_Median <- as.factor(test\_data$Above\_Median)

library(tidymodels)

## ── Attaching packages ────────────────────────────────────── tidymodels 1.2.0 ──

## ✔ broom 1.0.5 ✔ rsample 1.2.1   
## ✔ dials 1.2.1 ✔ tune 1.2.1   
## ✔ infer 1.0.7 ✔ workflows 1.1.4   
## ✔ modeldata 1.4.0 ✔ workflowsets 1.1.0   
## ✔ parsnip 1.2.1 ✔ yardstick 1.3.1   
## ✔ recipes 1.0.10

## ── Conflicts ───────────────────────────────────────── tidymodels\_conflicts() ──  
## ✖ scales::discard() masks purrr::discard()  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ recipes::fixed() masks stringr::fixed()  
## ✖ dplyr::lag() masks stats::lag()  
## ✖ caret::lift() masks purrr::lift()  
## ✖ yardstick::precision() masks caret::precision()  
## ✖ yardstick::recall() masks caret::recall()  
## ✖ yardstick::sensitivity() masks caret::sensitivity()  
## ✖ yardstick::spec() masks readr::spec()  
## ✖ yardstick::specificity() masks caret::specificity()  
## ✖ recipes::step() masks stats::step()  
## • Search for functions across packages at https://www.tidymodels.org/find/

logit\_model <- logistic\_reg() %>%  
 set\_engine("glm") %>%  
 fit(Above\_Median ~ Lot\_Area + Year\_Built + Total\_Bsmt\_SF + First\_Flr\_SF + MS\_Zoning + Neighborhood,   
 data = train\_data)  
  
logit\_model

## parsnip model object  
##   
##   
## Call: stats::glm(formula = Above\_Median ~ Lot\_Area + Year\_Built + Total\_Bsmt\_SF +   
## First\_Flr\_SF + MS\_Zoning + Neighborhood, family = stats::binomial,   
## data = data)  
##   
## Coefficients:  
## (Intercept)   
## -9.534e+01   
## Lot\_Area   
## 1.002e-04   
## Year\_Built   
## 3.732e-02   
## Total\_Bsmt\_SF   
## 5.777e-04   
## First\_Flr\_SF   
## 2.079e-03   
## MS\_ZoningC\_all   
## 3.126e+00   
## MS\_ZoningFloating\_Village\_Residential   
## 5.228e+00   
## MS\_ZoningI\_all   
## -1.229e+00   
## MS\_ZoningResidential\_High\_Density   
## 1.893e+01   
## MS\_ZoningResidential\_Low\_Density   
## 1.955e+01   
## MS\_ZoningResidential\_Medium\_Density   
## 1.859e+01   
## NeighborhoodBlueste   
## -1.731e+01   
## NeighborhoodBriardale   
## -1.718e+01   
## NeighborhoodBrookside   
## -8.291e-01   
## NeighborhoodClear\_Creek   
## -1.104e+00   
## NeighborhoodCollege\_Creek   
## -1.446e+00   
## NeighborhoodCrawford   
## 6.712e-01   
## NeighborhoodEdwards   
## -3.072e+00   
## NeighborhoodGilbert   
## 1.201e+00   
## NeighborhoodGreen\_Hills   
## 1.768e+01   
## NeighborhoodGreens   
## 3.471e-01   
## NeighborhoodIowa\_DOT\_and\_Rail\_Road   
## -1.371e+00   
## NeighborhoodLandmark   
## -1.858e+01   
## NeighborhoodMeadow\_Village   
## -1.728e+01   
## NeighborhoodMitchell   
## -2.502e+00   
## NeighborhoodNorth\_Ames   
## -2.993e+00   
## NeighborhoodNorthpark\_Villa   
## -1.932e+01   
## NeighborhoodNorthridge   
## 1.497e+01   
## NeighborhoodNorthridge\_Heights   
## 6.781e-01   
## NeighborhoodNorthwest\_Ames   
## -1.026e+00   
## NeighborhoodOld\_Town   
## -1.145e+00   
## NeighborhoodSawyer   
## -3.123e+00   
## NeighborhoodSawyer\_West   
## -1.400e+00   
## NeighborhoodSomerset   
## 1.458e+01   
## NeighborhoodSouth\_and\_West\_of\_Iowa\_State\_University   
## -7.778e-01   
## NeighborhoodStone\_Brook   
## 1.499e+01   
## NeighborhoodTimberland   
## 6.148e-01   
## NeighborhoodVeenker   
## -8.511e-01   
##   
## Degrees of Freedom: 1642 Total (i.e. Null); 1605 Residual  
## Null Deviance: 2277   
## Residual Deviance: 1028 AIC: 1104

logit\_preds <- predict(logit\_model, new\_data = test\_data, type = "prob")  
  
logit\_class <- ifelse(logit\_preds$.pred\_Yes > 0.5, "Yes", "No")  
  
confusionMatrix(as.factor(logit\_class), test\_data$Above\_Median)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 182 32  
## Yes 20 176  
##   
## Accuracy : 0.8732   
## 95% CI : (0.837, 0.9038)  
## No Information Rate : 0.5073   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.7465   
##   
## Mcnemar's Test P-Value : 0.1272   
##   
## Sensitivity : 0.9010   
## Specificity : 0.8462   
## Pos Pred Value : 0.8505   
## Neg Pred Value : 0.8980   
## Prevalence : 0.4927   
## Detection Rate : 0.4439   
## Detection Prevalence : 0.5220   
## Balanced Accuracy : 0.8736   
##   
## 'Positive' Class : No   
##

test\_data$MS\_Zoning <- factor(test\_data$MS\_Zoning, levels = levels(train\_data$MS\_Zoning))  
test\_data$Neighborhood <- factor(test\_data$Neighborhood, levels = levels(train\_data$Neighborhood))  
  
test\_data$Above\_Median <- factor(test\_data$Above\_Median, levels = levels(train\_data$Above\_Median))

library(randomForest)

## randomForest 4.7-1.2

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

##   
## Attaching package: 'randomForest'

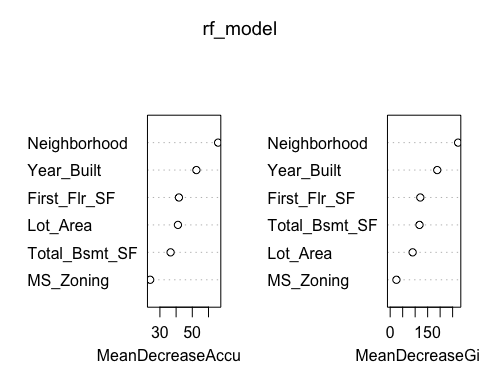
## The following object is masked from 'package:dplyr':  
##   
## combine

## The following object is masked from 'package:ggplot2':  
##   
## margin

rf\_model <- randomForest(Above\_Median ~ Lot\_Area + Year\_Built + Total\_Bsmt\_SF + First\_Flr\_SF + MS\_Zoning + Neighborhood,   
 data = train\_data,   
 importance = TRUE)  
  
print(rf\_model)

##   
## Call:  
## randomForest(formula = Above\_Median ~ Lot\_Area + Year\_Built + Total\_Bsmt\_SF + First\_Flr\_SF + MS\_Zoning + Neighborhood, data = train\_data, importance = TRUE)   
## Type of random forest: classification  
## Number of trees: 500  
## No. of variables tried at each split: 2  
##   
## OOB estimate of error rate: 11.93%  
## Confusion matrix:  
## No Yes class.error  
## No 713 95 0.1175743  
## Yes 101 734 0.1209581

varImpPlot(rf\_model)



rf\_preds <- predict(rf\_model, newdata = test\_data)  
  
confusionMatrix(rf\_preds, test\_data$Above\_Median)

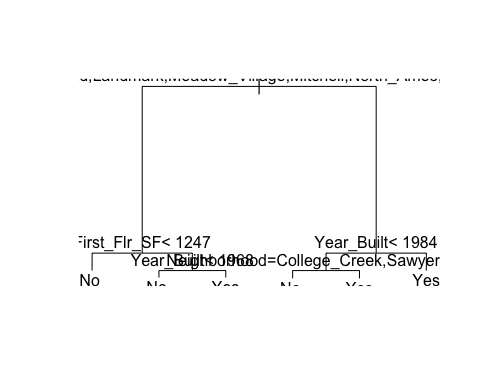
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 183 25  
## Yes 19 183  
##   
## Accuracy : 0.8927   
## 95% CI : (0.8586, 0.9209)  
## No Information Rate : 0.5073   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.7854   
##   
## Mcnemar's Test P-Value : 0.451   
##   
## Sensitivity : 0.9059   
## Specificity : 0.8798   
## Pos Pred Value : 0.8798   
## Neg Pred Value : 0.9059   
## Prevalence : 0.4927   
## Detection Rate : 0.4463   
## Detection Prevalence : 0.5073   
## Balanced Accuracy : 0.8929   
##   
## 'Positive' Class : No   
##

library(rpart)

##   
## Attaching package: 'rpart'

## The following object is masked from 'package:dials':  
##   
## prune

tree\_model <- rpart(Above\_Median ~ Lot\_Area + Year\_Built + Total\_Bsmt\_SF + First\_Flr\_SF + MS\_Zoning + Neighborhood,   
 data = train\_data,   
 method = "class")  
  
plot(tree\_model)  
text(tree\_model, pretty = 1)



tree\_preds <- predict(tree\_model, newdata = test\_data, type = "class")  
  
library(caret)  
confusionMatrix(tree\_preds, test\_data$Above\_Median)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 179 32  
## Yes 23 176  
##   
## Accuracy : 0.8659   
## 95% CI : (0.829, 0.8973)  
## No Information Rate : 0.5073   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.7318   
##   
## Mcnemar's Test P-Value : 0.2807   
##   
## Sensitivity : 0.8861   
## Specificity : 0.8462   
## Pos Pred Value : 0.8483   
## Neg Pred Value : 0.8844   
## Prevalence : 0.4927   
## Detection Rate : 0.4366   
## Detection Prevalence : 0.5146   
## Balanced Accuracy : 0.8661   
##   
## 'Positive' Class : No   
##

tree\_accuracy <- mean(tree\_preds == test\_data$Above\_Median)  
print(paste("Decision Tree Accuracy: ", tree\_accuracy))

## [1] "Decision Tree Accuracy: 0.865853658536585"

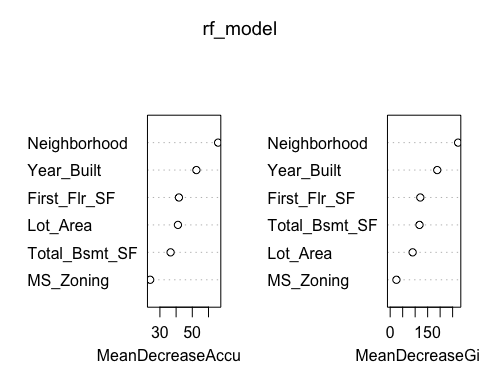
rf\_accuracy <- mean(rf\_preds == test\_data$Above\_Median)  
  
logit\_accuracy <- mean(logit\_class == test\_data$Above\_Median)  
  
tree\_accuracy <- mean(tree\_preds == test\_data$Above\_Median)  
  
print(tibble(  
 Model = c("Random Forest", "Logistic Regression", "Decision Tree"),  
 Accuracy = c(rf\_accuracy, logit\_accuracy, tree\_accuracy)  
))

## # A tibble: 3 × 2  
## Model Accuracy  
## <chr> <dbl>  
## 1 Random Forest 0.893  
## 2 Logistic Regression 0.873  
## 3 Decision Tree 0.866

results <- tibble(  
 Model = c("Random Forest", "Logistic Regression", "Decision Tree"),  
 Accuracy = c(0.9024390, 0.8731707, 0.8658537)  
)  
  
print(results)

## # A tibble: 3 × 2  
## Model Accuracy  
## <chr> <dbl>  
## 1 Random Forest 0.902  
## 2 Logistic Regression 0.873  
## 3 Decision Tree 0.866

varImpPlot(rf\_model)



# Logistic Regression Training Accuracy  
logit\_train\_preds <- predict(logit\_model, new\_data = train\_data, type = "prob")  
logit\_train\_class <- ifelse(logit\_train\_preds$.pred\_Yes > 0.5, "Yes", "No")  
logit\_train\_accuracy <- mean(logit\_train\_class == train\_data$Above\_Median)  
print(paste("Logistic Regression Training Accuracy: ", logit\_train\_accuracy))

## [1] "Logistic Regression Training Accuracy: 0.873402312842362"

# Random Forest Training Accuracy  
rf\_train\_preds <- predict(rf\_model, newdata = train\_data)  
rf\_train\_accuracy <- mean(rf\_train\_preds == train\_data$Above\_Median)  
print(paste("Random Forest Training Accuracy: ", rf\_train\_accuracy))

## [1] "Random Forest Training Accuracy: 0.999391357273281"

# Decision Tree Training Accuracy  
tree\_train\_preds <- predict(tree\_model, newdata = train\_data, type = "class")  
tree\_train\_accuracy <- mean(tree\_train\_preds == train\_data$Above\_Median)  
print(paste("Decision Tree Training Accuracy: ", tree\_train\_accuracy))

## [1] "Decision Tree Training Accuracy: 0.87522824102252"

# Original Random Forest Model  
rf\_model <- randomForest(Above\_Median ~ Lot\_Area + Year\_Built + Total\_Bsmt\_SF + First\_Flr\_SF + MS\_Zoning + Neighborhood,   
 data = train\_data,   
 importance = TRUE)  
  
# Original training accuracy  
rf\_train\_preds <- predict(rf\_model, newdata = train\_data)  
rf\_train\_accuracy <- mean(rf\_train\_preds == train\_data$Above\_Median)  
print(paste("Original Random Forest Training Accuracy: ", rf\_train\_accuracy))

## [1] "Original Random Forest Training Accuracy: 1"

# Original test accuracy  
rf\_test\_preds <- predict(rf\_model, newdata = test\_data)  
rf\_test\_accuracy <- mean(rf\_test\_preds == test\_data$Above\_Median)  
print(paste("Original Random Forest Test Accuracy: ", rf\_test\_accuracy))

## [1] "Original Random Forest Test Accuracy: 0.897560975609756"

# Tuned Random Forest Model - Limiting Tree Depth  
rf\_model\_tuned <- randomForest(Above\_Median ~ Lot\_Area + Year\_Built + Total\_Bsmt\_SF + First\_Flr\_SF + MS\_Zoning + Neighborhood,   
 data = train\_data,   
 importance = TRUE,   
 ntree = 500, # Number of trees  
 maxnodes = 30) # Limit tree depth  
  
# Tuned training accuracy  
rf\_train\_preds\_tuned <- predict(rf\_model\_tuned, newdata = train\_data)  
rf\_train\_accuracy\_tuned <- mean(rf\_train\_preds\_tuned == train\_data$Above\_Median)  
print(paste("Tuned Random Forest Training Accuracy: ", rf\_train\_accuracy\_tuned))

## [1] "Tuned Random Forest Training Accuracy: 0.907486305538649"

# Tuned test accuracy  
rf\_test\_preds\_tuned <- predict(rf\_model\_tuned, newdata = test\_data)  
rf\_test\_accuracy\_tuned <- mean(rf\_test\_preds\_tuned == test\_data$Above\_Median)  
print(paste("Tuned Random Forest Test Accuracy: ", rf\_test\_accuracy\_tuned))

## [1] "Tuned Random Forest Test Accuracy: 0.88780487804878"