BAN 502 Course Project

Mario G. Rodriguez

2022-06-25

# Phase 1

#### Libraries

#### Read in the data.

ames = read\_csv("ames\_student.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.

### Examined the structure, summary, and missingness of the data set.

# str(ames)  
# summary(ames)  
# skim(ames) #check for missingness

## Mutations

All categorical variables were converted to “factors”. Additionally, “First\_Flr\_SF”, “Second\_Flr\_SF, and Total Bsmt\_SF were combined to make up a new variable, Total\_SF.

ames2 = ames %>% mutate\_if(is.character,as\_factor) %>%  
 mutate(Total\_SF = First\_Flr\_SF + Second\_Flr\_SF + Total\_Bsmt\_SF) %>% # a new variable   
 mutate(Bsmt\_Full\_Bath = as\_factor(Bsmt\_Full\_Bath)) %>%  
 mutate(Bsmt\_Half\_Bath = as\_factor(Bsmt\_Half\_Bath)) %>%  
 mutate(Half\_Bath = as\_factor(Half\_Bath)) %>%  
 mutate(Full\_Bath = as\_factor(Full\_Bath)) %>%  
 mutate(Bedroom\_AbvGr = as\_factor(Bedroom\_AbvGr)) %>%  
 mutate(Kitchen\_AbvGr = as\_factor(Kitchen\_AbvGr)) %>%  
 mutate(TotRms\_AbvGrd = as\_factor(TotRms\_AbvGrd)) %>%  
 mutate(Fireplaces = as\_factor(Fireplaces)) %>%  
 mutate(Garage\_Cars = as\_factor(Garage\_Cars)) %>%  
 mutate(Mo\_Sold = as\_factor(Mo\_Sold))

## Data Exploration

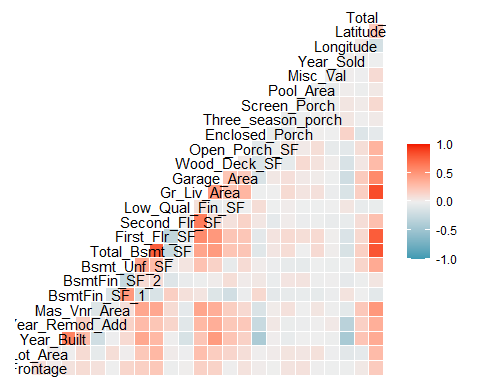
Examining the correlation between the variables, a strong correlation can be seen between “Total\_Bsmt\_SF” and “First\_Flr\_SF”, which was expected since there are approximately 57 homes with no basement in our data set and descriptive statistics between them are very similar.

ames2 %>% group\_by(BsmtFin\_Type\_1) %>% summarize(freq = n()) %>% arrange(desc(freq)) #there are 57 homes in "BSmtFin\_Type\_1" listed with "No\_Basement

## # A tibble: 7 × 2  
## BsmtFin\_Type\_1 freq  
## <fct> <int>  
## 1 Unf 602  
## 2 GLQ 578  
## 3 ALQ 298  
## 4 Rec 216  
## 5 BLQ 196  
## 6 LwQ 106  
## 7 No\_Basement 57

Examined correlation between quantitative variables.

ggcorr(ames2,label = FALSE)



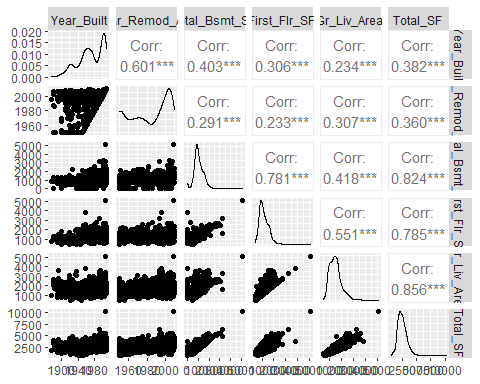
Correlation Matrix

Also created a database called “numeric\_vars” using all numeric variables from the “ames2” database.

numeric\_vars = dplyr::select\_if(ames2, is.numeric) #creates a new data frame called "numeric\_vars" of all numeric columns  
  
# str(numeric\_vars)   
# summary(numeric\_vars)  
# glimpse(numeric\_vars)

“numeric\_vars” allowed me to build a great scatter plot matrix to visualize possible multicollinearity and assist me to logically disregard certain columns. The matrix helps identify and visualize patterns in the given data. The below scatter plot uses those variables found to be most correlated with other variables.

ggpairs(numeric\_vars, columns = c(3, 4, 9, 10, 13, 25))



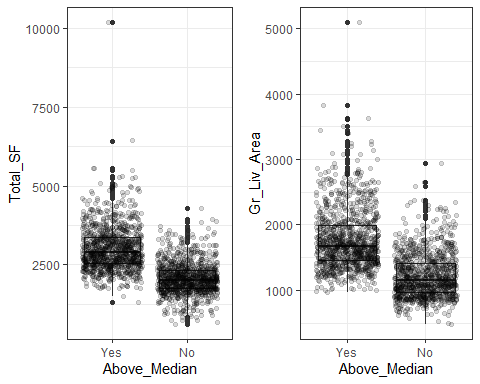
Great scatter plot matrix demonstrating multicollinearity

“Total\_Bsmt\_SF”, “First\_Flr\_SF” and “Second\_Flr\_SF” will be removed since they were used to create “Total\_SF”. However, “Gr\_Liv\_Area” is correlated with “Total\_SF” now.

### Demonstrating Weaker Predictors

“Gr\_Liv\_Area” seems to be a better predictor than the created “Total\_SF” and has a better box plot without the outliers present in “Total\_SF”. “Gr\_Liv\_Area” refers to all living square feet in a home that is above ground. I will analyze both variables later to determine which would be best in predicting “Above\_Median”.

p1 = ggplot(ames2, aes(x=Above\_Median, y= Total\_SF)) + geom\_boxplot() + geom\_jitter(alpha = 0.15) + theme\_bw()  
p2 = ggplot(ames2, aes(x=Above\_Median, y= Gr\_Liv\_Area)) + geom\_boxplot() + geom\_jitter(alpha = 0.15) + theme\_bw()  
grid.arrange(p1,p2,ncol=2)



Discovering Weaker Predictors

### Removal of Columns

Removed columns that did not provide much predictive value logically when analyzing their summaries and correlation matrix. Most importantly, some columns’ categories are not populated with many data points.

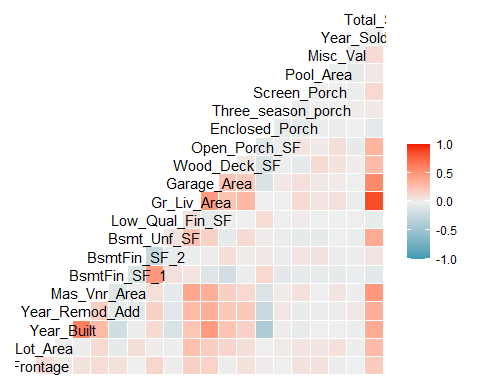
ames3 = ames2 %>% dplyr::select(-c(First\_Flr\_SF, Second\_Flr\_SF, Total\_Bsmt\_SF, Street, Alley, Utilities, Roof\_Style, Roof\_Matl, Condition\_1, Condition\_2, Bldg\_Type, Roof\_Style, Roof\_Matl, Exter\_Cond, Bsmt\_Cond, Heating, Central\_Air, Electrical, Functional, Garage\_Cond, Paved\_Drive, Pool\_QC, Fence, Misc\_Feature, Sale\_Type, Sale\_Condition, Lot\_Shape, Land\_Slope, Bldg\_Type, Land\_Contour, Garage\_Qual, Longitude, Latitude))

Reexamined the structure and summary of the data set after factor conversions.

#str(ames3)  
#summary(ames3)

Reexamined the correlation between the quantitative variables and everything seems good.

ggcorr(ames3,label = FALSE)



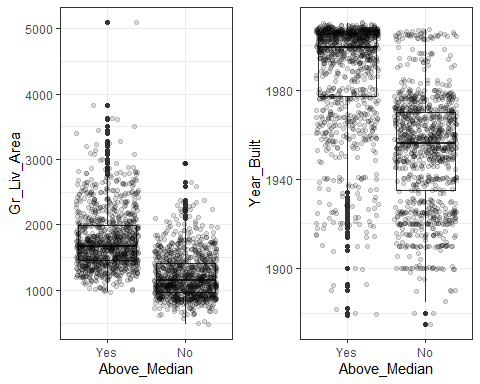
Correlation Matrix after Restructuring.

## Visuals

### Very Good Predictors

Below box plots demonstrate how “Gr\_Living\_Area” and “year the was house”Year\_Built” could possibly be great predictors of homes “Above\_Median”. Most homes that are above median are approximately 2900 square feet while homes below median are around 2000 square feet. Moreover, homes built after 2000 are more likely to be above median too.

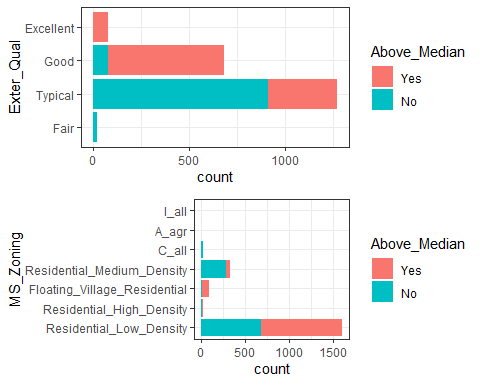
p3 = ggplot(ames3, aes(x=Above\_Median, y= Gr\_Liv\_Area)) + geom\_boxplot() + geom\_jitter(alpha = 0.15) + theme\_bw()  
p4 = ggplot(ames3, aes(x=Above\_Median, y= Year\_Built)) + geom\_boxplot() + geom\_jitter(alpha = 0.15) + theme\_bw()  
grid.arrange(p3,p4,ncol=2)



Gross Living Area and Year Built Box Plots

Below bar charts have great predictive qualities too. An excellent or good exterior quality of a home are very predictive.

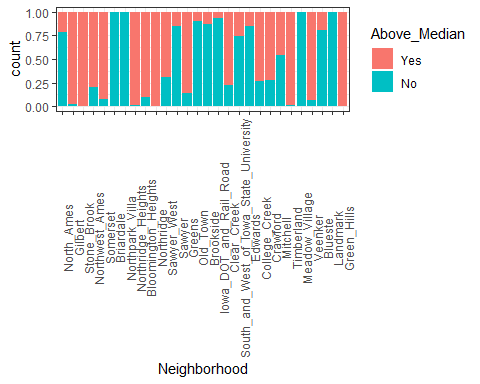
ames3 = ames3 %>% mutate(Exter\_Qual = fct\_relevel(Exter\_Qual, "Fair", "Typical", "Good", "Excellent"))  
  
p5 = ggplot(ames3, aes(x=Exter\_Qual, fill = Above\_Median)) + geom\_bar() + theme\_bw() + coord\_flip()  
p6 = ggplot(ames3, aes(x=MS\_Zoning, fill = Above\_Median)) + geom\_bar() + theme\_bw() + coord\_flip()  
grid.arrange(p5,p6,ncol=1)



Exterior Quality and MS Zoning Bar Charts

Stacked Neighborhoods also demonstrate how strong some categories are in predicting “Above\_Median”. Many neighborhoods exclusively have homes above median, while others do not.

ggplot(ames3, aes(x=Neighborhood, fill = Above\_Median)) + geom\_bar(position="fill") + theme\_bw() + theme(axis.text.x = element\_text(angle = 90))



Proportional Contribution of Neighborhoods

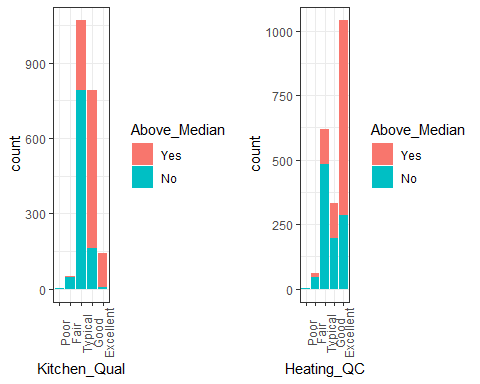
The below data demonstrates there are some neighborhoods with less than one percent of the observations. Smaller neighborhoods will be collapsed later in a recipe.

ames3 %>% group\_by(Neighborhood) %>% summarize(freq = n()) %>% arrange(desc(freq))

## # A tibble: 28 × 2  
## Neighborhood freq  
## <fct> <int>  
## 1 North\_Ames 327  
## 2 College\_Creek 183  
## 3 Old\_Town 181  
## 4 Edwards 129  
## 5 Somerset 119  
## 6 Gilbert 109  
## 7 Sawyer 109  
## 8 Northridge\_Heights 105  
## 9 Northwest\_Ames 95  
## 10 Sawyer\_West 82  
## # … with 18 more rows

We can see there are some predictive qualities to kitchen quality and heating QC.

ames3 = ames3 %>% mutate(Kitchen\_Qual = fct\_relevel(Kitchen\_Qual, "Poor", "Fair", "Typical", "Good", "Excellent")) %>%  
 mutate(Heating\_QC = fct\_relevel(Heating\_QC, "Poor", "Fair", "Typical", "Good", "Excellent"))  
  
p7 = ggplot(ames3, aes(x=Kitchen\_Qual, fill = Above\_Median)) + geom\_bar() + theme\_bw() + theme(axis.text.x = element\_text(angle = 90))  
p8 = ggplot(ames3, aes(x=Heating\_QC, fill = Above\_Median)) + geom\_bar() + theme\_bw() + theme(axis.text.x = element\_text(angle = 90))  
grid.arrange(p7,p8,ncol=2)



Kitchen Quality and Heating QC

### Various Tables Exploring Variables

Overall Quality is much more predictive than Overall Condition, Basement Exposure, and House Style.

t1 = table(ames3$Above\_Median, ames3$Overall\_Qual) #create a table object  
prop.table(t1, margin = 2 ) #crosstab with proportions

##   
## Above\_Average Average Good Very\_Good Excellent Below\_Average  
## Yes 0.50193050 0.14650767 0.88321168 0.98734177 0.98571429 0.04142012  
## No 0.49806950 0.85349233 0.11678832 0.01265823 0.01428571 0.95857988  
##   
## Fair Poor Very\_Excellent Very\_Poor  
## Yes 0.00000000 0.00000000 1.00000000 0.00000000  
## No 1.00000000 1.00000000 0.00000000 1.00000000

Overall Condition is not so predictive

t2 = table(ames3$Above\_Median, ames3$Overall\_Cond) #create a table object  
prop.table(t2, margin = 2 ) #crosstab with proportions

##   
## Average Above\_Average Good Poor Very\_Good Below\_Average  
## Yes 0.67804024 0.31914894 0.30419580 0.14285714 0.26530612 0.19178082  
## No 0.32195976 0.68085106 0.69580420 0.85714286 0.73469388 0.80821918  
##   
## Excellent Fair Very\_Poor  
## Yes 0.60714286 0.08571429 0.00000000  
## No 0.39285714 0.91428571 1.00000000

Basement Exposure

t3 = table(ames3$Above\_Median, ames3$Bsmt\_Exposure) #create a table object  
prop.table(t3, margin = 2 ) #crosstab with proportions

##   
## Gd No Av Mn No\_Basement  
## Yes 0.8442211 0.4395192 0.6408451 0.5586592 0.1333333  
## No 0.1557789 0.5604808 0.3591549 0.4413408 0.8666667

House Style

t4 = table(ames3$Above\_Median, ames3$House\_Style) #create a table object  
prop.table(t4, margin = 2 ) #crosstab with proportions

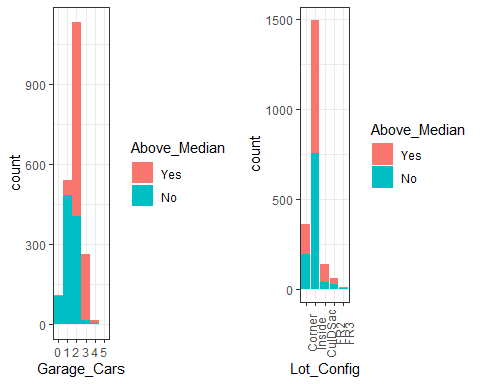
##   
## One\_Story Two\_Story One\_and\_Half\_Fin SLvl SFoyer One\_and\_Half\_Unf  
## Yes 0.4752852 0.7016949 0.2133333 0.5888889 0.2678571 0.0000000  
## No 0.5247148 0.2983051 0.7866667 0.4111111 0.7321429 1.0000000  
##   
## Two\_and\_Half\_Unf Two\_and\_Half\_Fin  
## Yes 0.4736842 0.6666667  
## No 0.5263158 0.3333333

### Other Variables Considered

* Variables not found to be as predictive as earlier variables.

Garage\_Cars & Lot Configuration

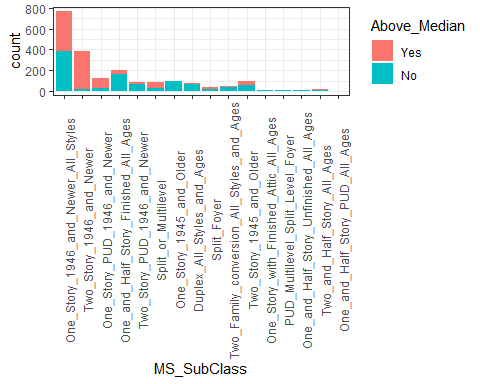
p9 = ggplot(ames3, aes(x=Garage\_Cars, fill = Above\_Median)) + geom\_bar() + theme\_bw()   
p10 = ggplot(ames3, aes(x=Lot\_Config, fill = Above\_Median)) + geom\_bar() + theme\_bw() + theme(axis.text.x = element\_text(angle = 90))  
grid.arrange(p9,p10,ncol=2)



Variables not so predictive.

MS\_Subclass

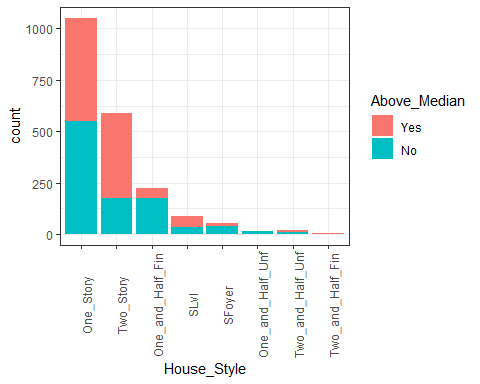
ggplot(ames3, aes(x=MS\_SubClass, fill = Above\_Median)) + geom\_bar() + theme\_bw() + theme(axis.text.x = element\_text(angle = 90))



MS\_Subclass Box Plots.

House Style

ggplot(ames3, aes(x=House\_Style, fill = Above\_Median)) + geom\_bar() + theme\_bw() + theme(axis.text.x = element\_text(angle = 90))



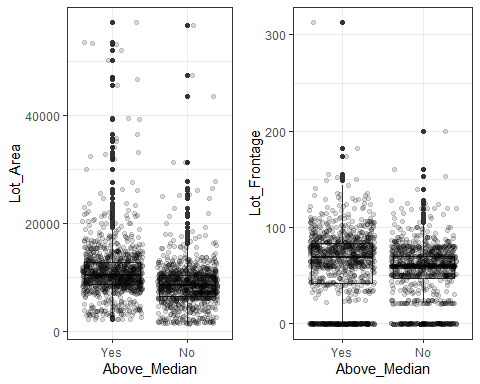
House Style Bar Chart.

### Not Good Predictors

“Lot\_Area” and “Lot\_Frontage”

Lot\_Area - There were some outliers removed and filtered to less than 65000. Still not a good predictor.

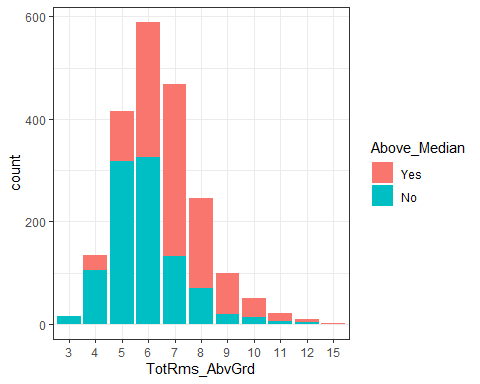
ames4 = ames3 %>% filter(Lot\_Area < 65000)  
p10 = ggplot(ames4, aes(x=Above\_Median, y= Lot\_Area)) + geom\_boxplot() + geom\_jitter(alpha = 0.15) + theme\_bw()  
p11 = ggplot(ames4, aes(x=Above\_Median, y= Lot\_Frontage)) + geom\_boxplot() + geom\_jitter(alpha = 0.15) + theme\_bw()  
grid.arrange(p10,p11,ncol=2)



Lot Area and Frontage not good predictors.

TotRms\_AbvGrd - Good predictive value for less than four and more than seven rooms.

ggplot(ames3, aes(x=TotRms\_AbvGrd, fill = Above\_Median)) + geom\_bar() + theme\_bw()



Total Rooms Above Ground.

# Phase 2

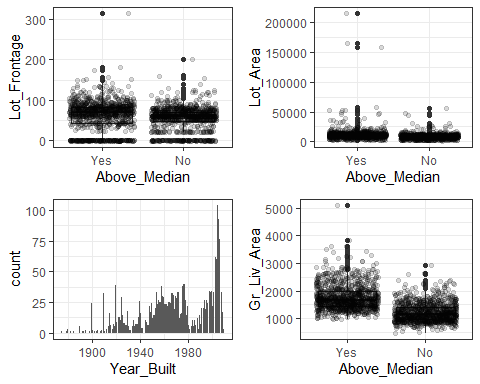
## Manipulating & Restructuring Data

Before splitting the data into Train/Test sets with k-fold cross validation , outliers will be removed.

### Outliers and Variables Analyzed

Outliers are present in Lot Frontage, Lot Area, Year Built and Gross Living Area. Outliers were removed on all variables.

p12 = ggplot(ames3, aes(x=Above\_Median, y= Lot\_Frontage)) + geom\_boxplot() + geom\_jitter(alpha = 0.15) + theme\_bw() #outliers greater than 150  
p13 = ggplot(ames3, aes(x=Above\_Median, y= Lot\_Area)) + geom\_boxplot() + geom\_jitter(alpha = 0.15) + theme\_bw() #outliers greater than 19000  
p14 = ggplot(ames3, aes(x=Year\_Built)) + geom\_bar()+ theme\_bw() #outliers less than 1910  
p15 = ggplot(ames3, aes(x=Above\_Median, y= Gr\_Liv\_Area)) + geom\_boxplot() + geom\_jitter(alpha = 0.15) + theme\_bw() #outliers greater than 2750  
grid.arrange(p12,p13,p14,p15,ncol=2)



Variables with Outliers

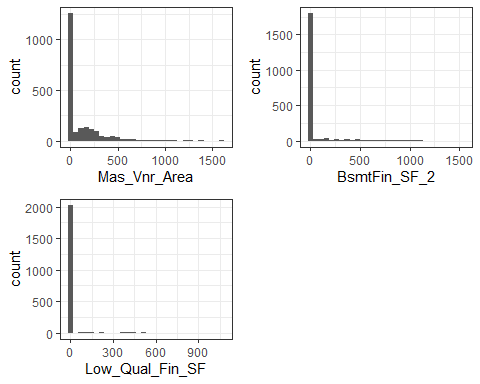
ames4 = ames3 %>% filter(Lot\_Frontage < 150 & Lot\_Area < 19000 & Year\_Built > 1910 & Gr\_Liv\_Area < 3750)

#### Variables with No Predicitive Value Analyzed and Removed

Masonry Veneer Area, Basement Finish SF, & Low Quality Finish SF

p16 = ggplot(ames3, aes(x=Mas\_Vnr\_Area)) + geom\_histogram()+ theme\_bw()   
p17 = ggplot(ames3, aes(x=BsmtFin\_SF\_2)) + geom\_histogram()+ theme\_bw()   
p18 = ggplot(ames3, aes(x=Low\_Qual\_Fin\_SF)) + geom\_histogram()+ theme\_bw()  
grid.arrange(p16,p17,p18,ncol=2)

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

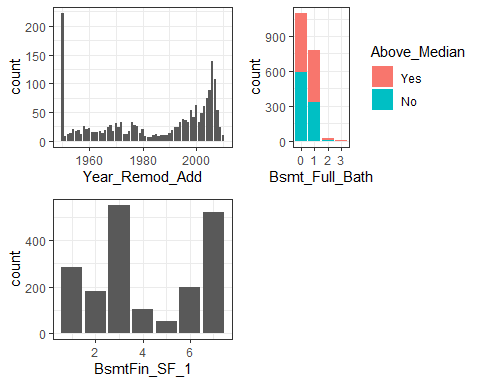


Variables with low predictive values.

ames5 = ames4 %>% dplyr::select(-c(Mas\_Vnr\_Area, BsmtFin\_SF\_2, Low\_Qual\_Fin\_SF))

#### Visuals

p19 = ggplot(ames5, aes(x=Year\_Remod\_Add)) + geom\_bar()+ theme\_bw()  
p20 = ggplot(ames5, aes(x=Bsmt\_Full\_Bath, fill = Above\_Median)) + geom\_bar()+ theme\_bw()  
p21 = ggplot(ames5, aes(x=BsmtFin\_SF\_1)) + geom\_bar()+ theme\_bw()  
grid.arrange(p19,p20,p21,ncol=2)



Visuals.

### Split

Before proceeding to the split, the levels (categories) in the response variable should be reordered. The “Yes” variable should be listed second.

levels(ames5$Above\_Median)

## [1] "Yes" "No"

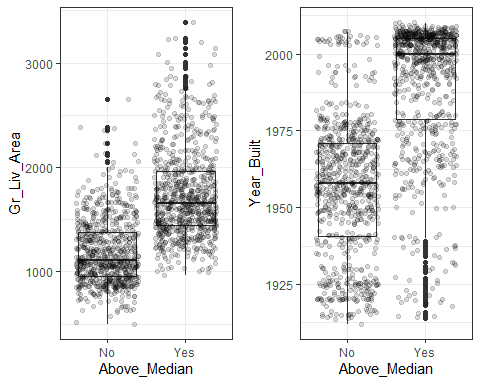
ames5 = ames5 %>% mutate(Above\_Median = fct\_relevel(Above\_Median, c("No","Yes")))

The data was split into training and testing sets with 80% of the data to training. The random split was stratified by the response variable, “Above\_Median”.

set.seed(123)  
ames\_split = initial\_split(ames5, prop = 0.80, strata = Above\_Median)  
train = training(ames\_split)  
test = testing(ames\_split)

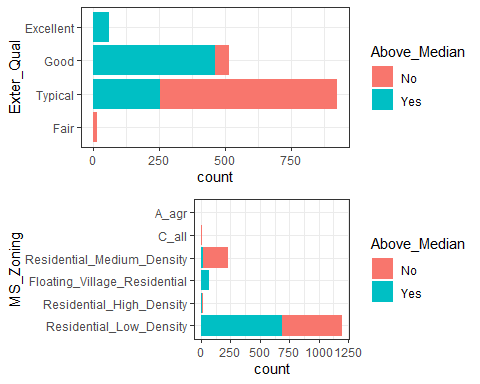
### Visualize the Training Set

p22 = ggplot(train, aes(x=Above\_Median, y= Gr\_Liv\_Area)) + geom\_boxplot() + geom\_jitter(alpha = 0.15) + theme\_bw()  
p23 = ggplot(train, aes(x=Above\_Median, y= Year\_Built)) + geom\_boxplot() + geom\_jitter(alpha = 0.15) + theme\_bw()  
grid.arrange(p22,p23,ncol=2)



Gross Living Area and Year Built Box Plots.

p24 = ggplot(train, aes(x=Exter\_Qual, fill = Above\_Median)) + geom\_bar() + theme\_bw() + coord\_flip()  
p25 = ggplot(train, aes(x=MS\_Zoning, fill = Above\_Median)) + geom\_bar() + theme\_bw() + coord\_flip()  
grid.arrange(p24,p25,ncol=1)



Exterior Quality and MS Zoning Bar Charts

### K-fold

Then, a k-fold cross-validation of ten was applied. The k-fold approach helps find a good value for lambda for a regression model. Here, 10 folds (the standard) was used.

folds = vfold\_cv(train, v = 10)

## Model and Predictions

### Logistic Regression

#### Logistic regression model with Gross Living Area predicting homes selling Above Median.

ames\_recipe = recipe(Above\_Median ~ Gr\_Liv\_Area, train)   
   
ames\_model =   
 logistic\_reg(mode = "classification") %>% #note the use of logistic\_reg  
 set\_engine("glm") #standard logistic regression engine is glm  
  
logreg\_wf = workflow() %>%  
 add\_recipe(ames\_recipe) %>%   
 add\_model(ames\_model)  
  
ames\_fit = fit(logreg\_wf, train)

summary(ames\_fit$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -3.3711 -0.6516 0.0605 0.7698 2.0934   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -6.4941790 0.3443244 -18.86 <2e-16 \*\*\*  
## Gr\_Liv\_Area 0.0045867 0.0002412 19.02 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 2099.3 on 1514 degrees of freedom  
## Residual deviance: 1391.7 on 1513 degrees of freedom  
## AIC: 1395.7  
##   
## Number of Fisher Scoring iterations: 5

The AIC of the model above using Gross Living Area is 1395.7. This model will be quickly compared to neighborhood to determine which variable should be used first.

#### Logistic regression model with Neighborhood instead.

ames\_recipe2 = recipe(Above\_Median ~ Neighborhood, train) %>%  
 step\_other(Neighborhood, threshold = 0.01) %>% #collapses small categories into an "Other" group on variables with many categories but less than 21  
 step\_dummy(all\_nominal(), -all\_outcomes()) #makes the factors categorical  
   
   
logreg\_wf2 = workflow() %>%  
 add\_recipe(ames\_recipe2) %>%   
 add\_model(ames\_model)  
  
ames\_fit2 = fit(logreg\_wf2, train)

summary(ames\_fit2$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.98861 -0.60757 0.00013 0.66276 2.46760   
##   
## Coefficients:  
## Estimate Std. Error  
## (Intercept) -1.2715 0.1554  
## Neighborhood\_Gilbert 5.6282 1.0183  
## Neighborhood\_Stone\_Brook 19.8375 1304.5277  
## Neighborhood\_Northwest\_Ames 2.6754 0.3364  
## Neighborhood\_Somerset 3.6694 0.4007  
## Neighborhood\_Briardale -17.2946 1537.4007  
## Neighborhood\_Northridge\_Heights 5.7258 1.0177  
## Neighborhood\_Bloomington\_Heights 4.0440 1.0424  
## Neighborhood\_Northridge 19.8375 1102.5272  
## Neighborhood\_Sawyer\_West 2.2523 0.3171  
## Neighborhood\_Sawyer -0.3246 0.3230  
## Neighborhood\_Old\_Town -1.7243 0.4839  
## Neighborhood\_Brookside -1.3312 0.5410  
## Neighborhood\_Iowa\_DOT\_and\_Rail\_Road -1.1264 0.6227  
## Neighborhood\_South\_and\_West\_of\_Iowa\_State\_University 0.6355 0.4406  
## Neighborhood\_Edwards -0.6863 0.3453  
## Neighborhood\_College\_Creek 2.2400 0.2404  
## Neighborhood\_Crawford 2.3701 0.3560  
## Neighborhood\_Mitchell 1.0747 0.3215  
## Neighborhood\_Timberland 19.8375 983.3248  
## Neighborhood\_Meadow\_Village -17.2946 1390.6312  
## Neighborhood\_other 1.5126 0.3245  
## z value Pr(>|z|)   
## (Intercept) -8.180 2.83e-16 \*\*\*  
## Neighborhood\_Gilbert 5.527 3.26e-08 \*\*\*  
## Neighborhood\_Stone\_Brook 0.015 0.987867   
## Neighborhood\_Northwest\_Ames 7.954 1.80e-15 \*\*\*  
## Neighborhood\_Somerset 9.158 < 2e-16 \*\*\*  
## Neighborhood\_Briardale -0.011 0.991025   
## Neighborhood\_Northridge\_Heights 5.626 1.84e-08 \*\*\*  
## Neighborhood\_Bloomington\_Heights 3.879 0.000105 \*\*\*  
## Neighborhood\_Northridge 0.018 0.985645   
## Neighborhood\_Sawyer\_West 7.103 1.22e-12 \*\*\*  
## Neighborhood\_Sawyer -1.005 0.315003   
## Neighborhood\_Old\_Town -3.563 0.000366 \*\*\*  
## Neighborhood\_Brookside -2.461 0.013866 \*   
## Neighborhood\_Iowa\_DOT\_and\_Rail\_Road -1.809 0.070472 .   
## Neighborhood\_South\_and\_West\_of\_Iowa\_State\_University 1.442 0.149188   
## Neighborhood\_Edwards -1.987 0.046888 \*   
## Neighborhood\_College\_Creek 9.317 < 2e-16 \*\*\*  
## Neighborhood\_Crawford 6.658 2.78e-11 \*\*\*  
## Neighborhood\_Mitchell 3.343 0.000829 \*\*\*  
## Neighborhood\_Timberland 0.020 0.983905   
## Neighborhood\_Meadow\_Village -0.012 0.990077   
## Neighborhood\_other 4.661 3.15e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 2099.3 on 1514 degrees of freedom  
## Residual deviance: 1136.0 on 1493 degrees of freedom  
## AIC: 1180  
##   
## Number of Fisher Scoring iterations: 17

The AIC of the model above using “Neighborhood” to predict “Above Median” was better than the AIC of 1395.7 presented by “Gross Living Area” providing an insight that location is more important than Gross Living Area. There is a significant number of neighborhoods that are not significant predictors; however, an AIC of 1180 presented by “Neighborhood” is better than Gr\_Liv\_Area. This model will be used instead to compare with other models.

#### Logistic regression model with Neighborhood and Gross Living Area.

ames\_recipe3 = recipe(Above\_Median ~ Neighborhood + Gr\_Liv\_Area, train) %>%  
 step\_other(Neighborhood, threshold = 0.01) %>% #collapses small categories into an "Other" group on variables with many categories but less than 21  
 step\_dummy(all\_nominal(), -all\_outcomes()) #makes the factors categorical   
   
logreg\_wf3 = workflow() %>%  
 add\_recipe(ames\_recipe3) %>%   
 add\_model(ames\_model)  
  
ames\_fit3 = fit(logreg\_wf3, train)

summary(ames\_fit3$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.91757 -0.37589 0.00005 0.32616 2.74920   
##   
## Coefficients:  
## Estimate Std. Error  
## (Intercept) -7.092e+00 4.789e-01  
## Gr\_Liv\_Area 4.265e-03 3.117e-04  
## Neighborhood\_Gilbert 5.025e+00 1.029e+00  
## Neighborhood\_Stone\_Brook 1.868e+01 1.068e+03  
## Neighborhood\_Northwest\_Ames 1.928e+00 3.922e-01  
## Neighborhood\_Somerset 3.182e+00 4.231e-01  
## Neighborhood\_Briardale -1.651e+01 1.498e+03  
## Neighborhood\_Northridge\_Heights 4.400e+00 1.033e+00  
## Neighborhood\_Bloomington\_Heights 4.118e+00 1.060e+00  
## Neighborhood\_Northridge 1.616e+01 9.433e+02  
## Neighborhood\_Sawyer\_West 1.823e+00 3.737e-01  
## Neighborhood\_Sawyer -2.649e-01 3.992e-01  
## Neighborhood\_Old\_Town -2.915e+00 6.019e-01  
## Neighborhood\_Brookside -1.469e+00 6.012e-01  
## Neighborhood\_Iowa\_DOT\_and\_Rail\_Road -1.733e+00 6.956e-01  
## Neighborhood\_South\_and\_West\_of\_Iowa\_State\_University -5.819e-01 6.024e-01  
## Neighborhood\_Edwards -5.989e-01 3.923e-01  
## Neighborhood\_College\_Creek 2.401e+00 2.971e-01  
## Neighborhood\_Crawford 1.295e+00 4.199e-01  
## Neighborhood\_Mitchell 1.286e+00 3.977e-01  
## Neighborhood\_Timberland 1.905e+01 8.975e+02  
## Neighborhood\_Meadow\_Village -1.598e+01 1.338e+03  
## Neighborhood\_other 1.227e+00 3.965e-01  
## z value Pr(>|z|)   
## (Intercept) -14.809 < 2e-16 \*\*\*  
## Gr\_Liv\_Area 13.682 < 2e-16 \*\*\*  
## Neighborhood\_Gilbert 4.885 1.03e-06 \*\*\*  
## Neighborhood\_Stone\_Brook 0.017 0.986050   
## Neighborhood\_Northwest\_Ames 4.915 8.87e-07 \*\*\*  
## Neighborhood\_Somerset 7.520 5.47e-14 \*\*\*  
## Neighborhood\_Briardale -0.011 0.991203   
## Neighborhood\_Northridge\_Heights 4.258 2.06e-05 \*\*\*  
## Neighborhood\_Bloomington\_Heights 3.884 0.000103 \*\*\*  
## Neighborhood\_Northridge 0.017 0.986333   
## Neighborhood\_Sawyer\_West 4.879 1.07e-06 \*\*\*  
## Neighborhood\_Sawyer -0.663 0.507015   
## Neighborhood\_Old\_Town -4.844 1.28e-06 \*\*\*  
## Neighborhood\_Brookside -2.444 0.014524 \*   
## Neighborhood\_Iowa\_DOT\_and\_Rail\_Road -2.491 0.012746 \*   
## Neighborhood\_South\_and\_West\_of\_Iowa\_State\_University -0.966 0.334067   
## Neighborhood\_Edwards -1.527 0.126814   
## Neighborhood\_College\_Creek 8.083 6.30e-16 \*\*\*  
## Neighborhood\_Crawford 3.084 0.002042 \*\*   
## Neighborhood\_Mitchell 3.234 0.001219 \*\*   
## Neighborhood\_Timberland 0.021 0.983066   
## Neighborhood\_Meadow\_Village -0.012 0.990471   
## Neighborhood\_other 3.094 0.001972 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 2099.33 on 1514 degrees of freedom  
## Residual deviance: 832.94 on 1492 degrees of freedom  
## AIC: 878.94  
##   
## Number of Fisher Scoring iterations: 17

The AIC of the model improved to 878.94 by adding Gross Living Area. Gross Living Area is a significant predictor. There is a small concern for multicollinearity (correlation between predictor variables). The Variance Inflation Factor will be used.

#### Logistic regression model with Neighborhood, Gross Living Area, and Year Built.

ames\_recipe4 = recipe(Above\_Median ~ Neighborhood + Gr\_Liv\_Area + Year\_Built, train) %>%  
 step\_other(Neighborhood, threshold = 0.01) %>% #collapses small categories into an "Other" group on variables with many categories but less than 21  
 step\_dummy(all\_nominal(), -all\_outcomes()) #makes the factors categorical   
   
logreg\_wf4 = workflow() %>%  
 add\_recipe(ames\_recipe4) %>%   
 add\_model(ames\_model)  
  
ames\_fit4 = fit(logreg\_wf4, train)

summary(ames\_fit4$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.92224 -0.35751 0.00004 0.30538 2.68576   
##   
## Coefficients:  
## Estimate Std. Error  
## (Intercept) -1.036e+02 1.490e+01  
## Gr\_Liv\_Area 4.186e-03 3.142e-04  
## Year\_Built 4.929e-02 7.571e-03  
## Neighborhood\_Gilbert 3.067e+00 1.071e+00  
## Neighborhood\_Stone\_Brook 1.706e+01 1.039e+03  
## Neighborhood\_Northwest\_Ames 1.228e+00 4.086e-01  
## Neighborhood\_Somerset 9.821e-01 5.384e-01  
## Neighborhood\_Briardale -1.709e+01 1.502e+03  
## Neighborhood\_Northridge\_Heights 2.177e+00 1.088e+00  
## Neighborhood\_Bloomington\_Heights 1.872e+00 1.112e+00  
## Neighborhood\_Northridge 1.449e+01 9.457e+02  
## Neighborhood\_Sawyer\_West 2.227e-01 4.554e-01  
## Neighborhood\_Sawyer -4.566e-01 3.941e-01  
## Neighborhood\_Old\_Town -1.543e+00 6.413e-01  
## Neighborhood\_Brookside -9.595e-02 6.207e-01  
## Neighborhood\_Iowa\_DOT\_and\_Rail\_Road -7.591e-01 7.454e-01  
## Neighborhood\_South\_and\_West\_of\_Iowa\_State\_University 6.364e-01 6.150e-01  
## Neighborhood\_Edwards -1.177e+00 4.504e-01  
## Neighborhood\_College\_Creek 7.309e-01 4.057e-01  
## Neighborhood\_Crawford 2.482e+00 5.027e-01  
## Neighborhood\_Mitchell 1.249e-01 4.386e-01  
## Neighborhood\_Timberland 1.745e+01 8.588e+02  
## Neighborhood\_Meadow\_Village -1.658e+01 1.338e+03  
## Neighborhood\_other 4.459e-01 4.151e-01  
## z value Pr(>|z|)   
## (Intercept) -6.953 3.57e-12 \*\*\*  
## Gr\_Liv\_Area 13.325 < 2e-16 \*\*\*  
## Year\_Built 6.510 7.51e-11 \*\*\*  
## Neighborhood\_Gilbert 2.863 0.00419 \*\*   
## Neighborhood\_Stone\_Brook 0.016 0.98690   
## Neighborhood\_Northwest\_Ames 3.005 0.00266 \*\*   
## Neighborhood\_Somerset 1.824 0.06815 .   
## Neighborhood\_Briardale -0.011 0.99092   
## Neighborhood\_Northridge\_Heights 2.000 0.04550 \*   
## Neighborhood\_Bloomington\_Heights 1.684 0.09220 .   
## Neighborhood\_Northridge 0.015 0.98777   
## Neighborhood\_Sawyer\_West 0.489 0.62483   
## Neighborhood\_Sawyer -1.159 0.24657   
## Neighborhood\_Old\_Town -2.407 0.01609 \*   
## Neighborhood\_Brookside -0.155 0.87715   
## Neighborhood\_Iowa\_DOT\_and\_Rail\_Road -1.018 0.30850   
## Neighborhood\_South\_and\_West\_of\_Iowa\_State\_University 1.035 0.30082   
## Neighborhood\_Edwards -2.614 0.00894 \*\*   
## Neighborhood\_College\_Creek 1.802 0.07161 .   
## Neighborhood\_Crawford 4.938 7.91e-07 \*\*\*  
## Neighborhood\_Mitchell 0.285 0.77581   
## Neighborhood\_Timberland 0.020 0.98379   
## Neighborhood\_Meadow\_Village -0.012 0.99011   
## Neighborhood\_other 1.074 0.28276   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 2099.33 on 1514 degrees of freedom  
## Residual deviance: 783.79 on 1491 degrees of freedom  
## AIC: 831.79  
##   
## Number of Fisher Scoring iterations: 17

The AIC of the model improved to 831.79, still a significant drop-off by adding Year Built. Another strong predictor will be added to see if there are any improvements to the model.

#### Logistic regression model with Neighborhood, Gross Living Area, Year Built, and Exterior Quality.

ames\_recipe5 = recipe(Above\_Median ~ Neighborhood + Gr\_Liv\_Area + Year\_Built + Exter\_Qual, train) %>%  
 step\_other(Neighborhood, threshold = 0.01) %>% #collapses small categories into an "Other" group on variables with many categories but less than 21  
 step\_dummy(all\_nominal(), -all\_outcomes()) #makes the factors categorical  
   
logreg\_wf5 = workflow() %>%  
 add\_recipe(ames\_recipe5) %>%   
 add\_model(ames\_model)  
  
ames\_fit5 = fit(logreg\_wf5, train)

summary(ames\_fit5$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.80847 -0.35421 0.00001 0.28764 2.71476   
##   
## Coefficients:  
## Estimate Std. Error  
## (Intercept) -8.149e+01 1.594e+01  
## Gr\_Liv\_Area 4.070e-03 3.218e-04  
## Year\_Built 3.840e-02 8.077e-03  
## Neighborhood\_Gilbert 3.223e+00 1.078e+00  
## Neighborhood\_Stone\_Brook 1.742e+01 1.725e+03  
## Neighborhood\_Northwest\_Ames 1.428e+00 4.147e-01  
## Neighborhood\_Somerset 5.490e-01 5.598e-01  
## Neighborhood\_Briardale -1.785e+01 2.480e+03  
## Neighborhood\_Northridge\_Heights 1.420e+00 1.110e+00  
## Neighborhood\_Bloomington\_Heights 1.295e+00 1.128e+00  
## Neighborhood\_Northridge 1.488e+01 1.569e+03  
## Neighborhood\_Sawyer\_West 1.087e-03 4.808e-01  
## Neighborhood\_Sawyer -3.530e-01 3.987e-01  
## Neighborhood\_Old\_Town -1.741e+00 6.499e-01  
## Neighborhood\_Brookside -2.531e-01 6.227e-01  
## Neighborhood\_Iowa\_DOT\_and\_Rail\_Road -9.392e-01 7.610e-01  
## Neighborhood\_South\_and\_West\_of\_Iowa\_State\_University 5.560e-01 6.107e-01  
## Neighborhood\_Edwards -1.087e+00 4.631e-01  
## Neighborhood\_College\_Creek 5.566e-01 4.295e-01  
## Neighborhood\_Crawford 2.101e+00 5.078e-01  
## Neighborhood\_Mitchell 4.238e-01 4.476e-01  
## Neighborhood\_Timberland 1.839e+01 1.351e+03  
## Neighborhood\_Meadow\_Village -1.735e+01 2.213e+03  
## Neighborhood\_other 3.553e-01 4.273e-01  
## Exter\_Qual\_Typical -7.477e-01 1.223e+00  
## Exter\_Qual\_Good 4.500e-01 1.249e+00  
## Exter\_Qual\_Excellent 1.737e+01 1.099e+03  
## z value Pr(>|z|)   
## (Intercept) -5.113 3.17e-07 \*\*\*  
## Gr\_Liv\_Area 12.649 < 2e-16 \*\*\*  
## Year\_Built 4.754 1.99e-06 \*\*\*  
## Neighborhood\_Gilbert 2.990 0.002788 \*\*   
## Neighborhood\_Stone\_Brook 0.010 0.991942   
## Neighborhood\_Northwest\_Ames 3.444 0.000572 \*\*\*  
## Neighborhood\_Somerset 0.981 0.326754   
## Neighborhood\_Briardale -0.007 0.994256   
## Neighborhood\_Northridge\_Heights 1.280 0.200577   
## Neighborhood\_Bloomington\_Heights 1.148 0.250780   
## Neighborhood\_Northridge 0.009 0.992437   
## Neighborhood\_Sawyer\_West 0.002 0.998196   
## Neighborhood\_Sawyer -0.885 0.375969   
## Neighborhood\_Old\_Town -2.680 0.007370 \*\*   
## Neighborhood\_Brookside -0.406 0.684459   
## Neighborhood\_Iowa\_DOT\_and\_Rail\_Road -1.234 0.217150   
## Neighborhood\_South\_and\_West\_of\_Iowa\_State\_University 0.911 0.362525   
## Neighborhood\_Edwards -2.348 0.018899 \*   
## Neighborhood\_College\_Creek 1.296 0.195037   
## Neighborhood\_Crawford 4.137 3.51e-05 \*\*\*  
## Neighborhood\_Mitchell 0.947 0.343757   
## Neighborhood\_Timberland 0.014 0.989142   
## Neighborhood\_Meadow\_Village -0.008 0.993744   
## Neighborhood\_other 0.832 0.405678   
## Exter\_Qual\_Typical -0.611 0.541019   
## Exter\_Qual\_Good 0.360 0.718547   
## Exter\_Qual\_Excellent 0.016 0.987384   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 2099.33 on 1514 degrees of freedom  
## Residual deviance: 756.13 on 1488 degrees of freedom  
## AIC: 810.13  
##   
## Number of Fisher Scoring iterations: 18

The AIC of the model slightly improved to 810.13. Additionally, all Exterior Quality’s levels are not significant. Exterior Quality will be omitted from the model and replaced with MS Zoning.

#### Logistic regression model with Neighborhood, Gross Living Area, Year Built, and MS Zoning.

ames\_recipe6 = recipe(Above\_Median ~ Neighborhood + Gr\_Liv\_Area + Year\_Built + MS\_Zoning, train) %>%  
 step\_other(Neighborhood, MS\_Zoning, threshold = 0.01) %>% #collapses small categories into an "Other" group on variables with many categories  
 step\_dummy(all\_nominal(), -all\_outcomes()) #makes the factors categorical   
   
logreg\_wf6 = workflow() %>%  
 add\_recipe(ames\_recipe6) %>%   
 add\_model(ames\_model)  
  
ames\_fit6 = fit(logreg\_wf6, train)

summary(ames\_fit6$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.94986 -0.35979 0.00004 0.28817 2.85709   
##   
## Coefficients:  
## Estimate Std. Error  
## (Intercept) -1.256e+02 1.603e+01  
## Gr\_Liv\_Area 4.164e-03 3.158e-04  
## Year\_Built 6.056e-02 8.140e-03  
## Neighborhood\_Gilbert 2.608e+00 1.078e+00  
## Neighborhood\_Stone\_Brook 1.668e+01 1.033e+03  
## Neighborhood\_Northwest\_Ames 1.048e+00 4.120e-01  
## Neighborhood\_Somerset 7.036e-01 1.113e+00  
## Neighborhood\_Briardale -1.528e+01 1.503e+03  
## Neighborhood\_Northridge\_Heights 1.646e+00 1.098e+00  
## Neighborhood\_Bloomington\_Heights 1.797e+00 1.127e+00  
## Neighborhood\_Northridge 1.409e+01 9.466e+02  
## Neighborhood\_Sawyer\_West -1.604e-01 4.710e-01  
## Neighborhood\_Sawyer -4.338e-01 3.904e-01  
## Neighborhood\_Old\_Town 2.666e-01 7.458e-01  
## Neighborhood\_Brookside 1.177e+00 6.980e-01  
## Neighborhood\_Iowa\_DOT\_and\_Rail\_Road 1.360e+00 9.183e-01  
## Neighborhood\_South\_and\_West\_of\_Iowa\_State\_University 9.536e-01 6.373e-01  
## Neighborhood\_Edwards -9.868e-01 4.421e-01  
## Neighborhood\_College\_Creek 4.544e-01 4.221e-01  
## Neighborhood\_Crawford 2.849e+00 5.083e-01  
## Neighborhood\_Mitchell 6.956e-03 4.458e-01  
## Neighborhood\_Timberland 1.713e+01 8.463e+02  
## Neighborhood\_Meadow\_Village -1.478e+01 1.338e+03  
## Neighborhood\_other 4.222e-01 4.274e-01  
## MS\_Zoning\_Floating\_Village\_Residential -2.904e-01 1.108e+00  
## MS\_Zoning\_Residential\_Medium\_Density -1.972e+00 5.073e-01  
## MS\_Zoning\_other -1.643e+00 9.668e-01  
## z value Pr(>|z|)   
## (Intercept) -7.837 4.60e-15 \*\*\*  
## Gr\_Liv\_Area 13.186 < 2e-16 \*\*\*  
## Year\_Built 7.439 1.01e-13 \*\*\*  
## Neighborhood\_Gilbert 2.420 0.015506 \*   
## Neighborhood\_Stone\_Brook 0.016 0.987122   
## Neighborhood\_Northwest\_Ames 2.545 0.010924 \*   
## Neighborhood\_Somerset 0.632 0.527118   
## Neighborhood\_Briardale -0.010 0.991888   
## Neighborhood\_Northridge\_Heights 1.499 0.133964   
## Neighborhood\_Bloomington\_Heights 1.594 0.110935   
## Neighborhood\_Northridge 0.015 0.988122   
## Neighborhood\_Sawyer\_West -0.341 0.733365   
## Neighborhood\_Sawyer -1.111 0.266537   
## Neighborhood\_Old\_Town 0.357 0.720754   
## Neighborhood\_Brookside 1.686 0.091883 .   
## Neighborhood\_Iowa\_DOT\_and\_Rail\_Road 1.480 0.138753   
## Neighborhood\_South\_and\_West\_of\_Iowa\_State\_University 1.496 0.134559   
## Neighborhood\_Edwards -2.232 0.025628 \*   
## Neighborhood\_College\_Creek 1.076 0.281734   
## Neighborhood\_Crawford 5.605 2.08e-08 \*\*\*  
## Neighborhood\_Mitchell 0.016 0.987552   
## Neighborhood\_Timberland 0.020 0.983847   
## Neighborhood\_Meadow\_Village -0.011 0.991186   
## Neighborhood\_other 0.988 0.323250   
## MS\_Zoning\_Floating\_Village\_Residential -0.262 0.793139   
## MS\_Zoning\_Residential\_Medium\_Density -3.887 0.000102 \*\*\*  
## MS\_Zoning\_other -1.700 0.089164 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 2099.33 on 1514 degrees of freedom  
## Residual deviance: 764.59 on 1488 degrees of freedom  
## AIC: 818.59  
##   
## Number of Fisher Scoring iterations: 17

The AIC of the model only did not improve. The above model performed worse than Exterior Quality. Most importantly, MS Zoning’s P-values also demonstrated they are not significant. The only category significant to predict “Above\_Median” is Medium Density MS Zoning; however, there are few observations for such MS Zoning. MS Zoning should not be used on the model too.

The best regression model examined was Neighborhood, Gr\_Liv\_Area, Year\_Built to predict Above\_Median. The below are this model’s predictions.

##### Predictions on Training and Test Sets

Regression Model Predictions on Training Set

predLR = predict(ames\_fit4, train)  
head(predLR)

## # A tibble: 6 × 1  
## .pred\_class  
## <fct>   
## 1 No   
## 2 No   
## 3 No   
## 4 No   
## 5 No   
## 6 No

Confusion matrix

confusionMatrix(predLR$.pred\_class, train$Above\_Median, positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 657 96  
## Yes 82 680  
##   
## Accuracy : 0.8825   
## 95% CI : (0.8652, 0.8983)  
## No Information Rate : 0.5122   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.765   
##   
## Mcnemar's Test P-Value : 0.3299   
##   
## Sensitivity : 0.8763   
## Specificity : 0.8890   
## Pos Pred Value : 0.8924   
## Neg Pred Value : 0.8725   
## Prevalence : 0.5122   
## Detection Rate : 0.4488   
## Detection Prevalence : 0.5030   
## Balanced Accuracy : 0.8827   
##   
## 'Positive' Class : Yes   
##

There was a 0.8825 accuracy using this regression model on the test set. Now, this regression model will be compared to Lasso Regression Model.

Regression Model Predictions on Training Set

predLRT = predict(ames\_fit4, test)  
head(predLRT)

## # A tibble: 6 × 1  
## .pred\_class  
## <fct>   
## 1 No   
## 2 Yes   
## 3 Yes   
## 4 No   
## 5 Yes   
## 6 No

confusionMatrix(predLRT$.pred\_class, test$Above\_Median, positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 158 18  
## Yes 27 177  
##   
## Accuracy : 0.8816   
## 95% CI : (0.8448, 0.9123)  
## No Information Rate : 0.5132   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.7627   
##   
## Mcnemar's Test P-Value : 0.233   
##   
## Sensitivity : 0.9077   
## Specificity : 0.8541   
## Pos Pred Value : 0.8676   
## Neg Pred Value : 0.8977   
## Prevalence : 0.5132   
## Detection Rate : 0.4658   
## Detection Prevalence : 0.5368   
## Balanced Accuracy : 0.8809   
##   
## 'Positive' Class : Yes   
##

Check Multicollinearity via a statistic known as the Variance Inflation Factor (VIF).

car::vif(ames\_fit6$fit$fit$fit)

## Gr\_Liv\_Area   
## 1.162809   
## Year\_Built   
## 4.224833   
## Neighborhood\_Gilbert   
## 1.117995   
## Neighborhood\_Stone\_Brook   
## 1.000000   
## Neighborhood\_Northwest\_Ames   
## 1.288538   
## Neighborhood\_Somerset   
## 8.002071   
## Neighborhood\_Briardale   
## 1.000000   
## Neighborhood\_Northridge\_Heights   
## 1.158026   
## Neighborhood\_Bloomington\_Heights   
## 1.141367   
## Neighborhood\_Northridge   
## 1.000000   
## Neighborhood\_Sawyer\_West   
## 1.760535   
## Neighborhood\_Sawyer   
## 1.214885   
## Neighborhood\_Old\_Town   
## 1.634647   
## Neighborhood\_Brookside   
## 1.346391   
## Neighborhood\_Iowa\_DOT\_and\_Rail\_Road   
## 1.608098   
## Neighborhood\_South\_and\_West\_of\_Iowa\_State\_University   
## 1.193340   
## Neighborhood\_Edwards   
## 1.297850   
## Neighborhood\_College\_Creek   
## 2.085489   
## Neighborhood\_Crawford   
## 1.455354   
## Neighborhood\_Mitchell   
## 1.390471   
## Neighborhood\_Timberland   
## 1.000000   
## Neighborhood\_Meadow\_Village   
## 1.000000   
## Neighborhood\_other   
## 1.271019   
## MS\_Zoning\_Floating\_Village\_Residential   
## 6.895305   
## MS\_Zoning\_Residential\_Medium\_Density   
## 1.971377   
## MS\_Zoning\_other   
## 1.064292

There is minor collinearity.

A k-fold approach will be done to help find a good value for lambda for a lasso regression model.

#### Lasso Regression Model

ames\_recipe7 = recipe(Above\_Median ~., train) %>% #add all variables via ~.  
 step\_other(MS\_Zoning, Fireplace\_Qu, Neighborhood, House\_Style, Overall\_Cond, Mas\_Vnr\_Type, Foundation, threshold = 0.01) %>% #collapses small categories into an "Other" group on variables with many categories but less than 21  
 step\_dummy(all\_nominal(), -all\_outcomes()) %>% #makes the factors categorical  
 step\_center(all\_predictors()) %>% #centers the predictors  
 step\_scale(all\_predictors()) #scales the predictors  
  
lasso\_model = #give the model type a name  
 logistic\_reg(penalty = tune(), mixture = 1) %>% #mixture = 1 sets up Lasso, 0 sets up Ridge; tune indicates I am trying to select the best lambda value  
 set\_engine("glmnet") #specify the specify type of linear tool we want to use  
  
#try different lambda values ranging from 0 to 10000 in increments of 100  
#you may need to tweak this range  
lambda\_grid = expand.grid(penalty = seq(0,10000,100)) #consider a sequence of values from 0 to 10000 by 100  
  
lasso\_wflow =  
 workflow() %>%  
 add\_model(lasso\_model) %>%  
 add\_recipe(ames\_recipe7)  
  
lasso\_res = lasso\_wflow %>%  
 tune\_grid(  
 resamples = folds, #new line  
 grid = lambda\_grid  
 )

What is the exact best value?

best\_accuracy = lasso\_res %>%  
 select\_best("accuracy")  
best\_accuracy

## # A tibble: 1 × 2  
## penalty .config   
## <dbl> <chr>   
## 1 0 Preprocessor1\_Model001

final\_lasso = lasso\_wflow %>% finalize\_workflow(best\_accuracy)

Shows the model performance on the testing set.

last\_fit(  
 final\_lasso,  
 ames\_split) %>%  
 collect\_metrics()

## # A tibble: 2 × 4  
## .metric .estimator .estimate .config   
## <chr> <chr> <dbl> <chr>   
## 1 accuracy binary 0.892 Preprocessor1\_Model1  
## 2 roc\_auc binary 0.955 Preprocessor1\_Model1

There is an accuracy of 0.8921 on the testing set.

#### Classification Trees

Now that we have the split data, let’s build a classification tree. Here we use caret to manage the model building.

ames8\_recipe = recipe(Above\_Median ~ ., train) %>%  
 step\_other(Neighborhood, threshold = 0.01) %>% #collapses small categories into an "Other" group on variables with many categories but less than 21  
 step\_dummy(all\_nominal(), -all\_outcomes()) #makes the factors categorical  
#   
#   
# tree\_model = decision\_tree() %>%  
# set\_engine("rpart", model = TRUE) %>% #don't forget the model = TRUE flag  
# set\_mode("classification")  
#   
# ames\_wflow =  
# workflow() %>%  
# add\_model(tree\_model) %>%  
# add\_recipe(ames8\_recipe)  
#   
# ames\_fit8 = fit(ames\_wflow, train)

Save the model to a file to load later

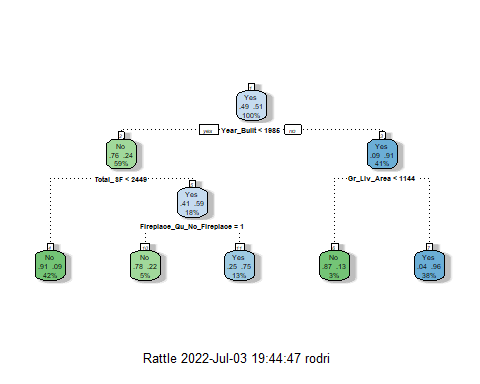
# saveRDS(ames\_fit8, "ames\_fit8.rds")

Load the model

ames\_fit8 = readRDS("ames\_fit8.rds")

tree = ames\_fit8 %>%  
 extract\_fit\_parsnip() %>%  
 pluck("fit")

fancyRpartPlot(tree)



ames\_fit8$fit$fit$fit$cptable

## CP nsplit rel error xerror xstd  
## 1 0.63464141 0 1.0000000 1.0000000 0.02632706  
## 2 0.06901218 1 0.3653586 0.3802436 0.02047199  
## 3 0.05953992 2 0.2963464 0.3491204 0.01979825  
## 4 0.03788904 3 0.2368065 0.2746955 0.01794172  
## 5 0.01000000 4 0.1989175 0.2097429 0.01596190

Predictions on training set,

treepred = predict(ames\_fit8, train, type = "class")  
head(treepred)

## # A tibble: 6 × 1  
## .pred\_class  
## <fct>   
## 1 No   
## 2 No   
## 3 No   
## 4 No   
## 5 No   
## 6 No

Caret confusion matrix and accuracy, etc. calcs.

confusionMatrix(treepred$.pred\_class,train$Above\_Median,positive="Yes") #predictions first then actual

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 670 78  
## Yes 69 698  
##   
## Accuracy : 0.903   
## 95% CI : (0.8869, 0.9174)  
## No Information Rate : 0.5122   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.8059   
##   
## Mcnemar's Test P-Value : 0.5094   
##   
## Sensitivity : 0.8995   
## Specificity : 0.9066   
## Pos Pred Value : 0.9100   
## Neg Pred Value : 0.8957   
## Prevalence : 0.5122   
## Detection Rate : 0.4607   
## Detection Prevalence : 0.5063   
## Balanced Accuracy : 0.9031   
##   
## 'Positive' Class : Yes   
##

Predictions on testing set.

treepred\_test = predict(ames\_fit8, test, type = "class")  
head(treepred\_test)

## # A tibble: 6 × 1  
## .pred\_class  
## <fct>   
## 1 No   
## 2 Yes   
## 3 Yes   
## 4 No   
## 5 Yes   
## 6 No

Caret confusion matrix and accuracy, etc. calcs

confusionMatrix(treepred\_test$.pred\_class,test$Above\_Median,positive="Yes") #predictions first then actual

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 164 18  
## Yes 21 177  
##   
## Accuracy : 0.8974   
## 95% CI : (0.8624, 0.926)  
## No Information Rate : 0.5132   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.7945   
##   
## Mcnemar's Test P-Value : 0.7488   
##   
## Sensitivity : 0.9077   
## Specificity : 0.8865   
## Pos Pred Value : 0.8939   
## Neg Pred Value : 0.9011   
## Prevalence : 0.5132   
## Detection Rate : 0.4658   
## Detection Prevalence : 0.5211   
## Balanced Accuracy : 0.8971   
##   
## 'Positive' Class : Yes   
##

#### Building Random Forest Model with Tidymodels

# rf\_model = rand\_forest() %>%  
# set\_engine("ranger") %>%  
# set\_mode("classification")  
#   
# ames\_wflow2 =  
# workflow() %>%  
# add\_model(rf\_model) %>%  
# add\_recipe(ames8\_recipe)  
#   
# set.seed(123)  
# ames\_fit9 = fit(ames\_wflow2, train)

Save the model to a file to load later

# saveRDS(ames\_fit9, "ames\_fit9.rds")

Load the model

ames\_fit9 = readRDS("ames\_fit9.rds")

Check out random forest details

ames\_fit9

## ══ Workflow [trained] ══════════════════════════════════════════════════════════  
## Preprocessor: Recipe  
## Model: rand\_forest()  
##   
## ── Preprocessor ────────────────────────────────────────────────────────────────  
## 2 Recipe Steps  
##   
## • step\_other()  
## • step\_dummy()  
##   
## ── Model ───────────────────────────────────────────────────────────────────────  
## Ranger result  
##   
## Call:  
## ranger::ranger(x = maybe\_data\_frame(x), y = y, num.threads = 1, verbose = FALSE, seed = sample.int(10^5, 1), probability = TRUE)   
##   
## Type: Probability estimation   
## Number of trees: 500   
## Sample size: 1515   
## Number of independent variables: 222   
## Mtry: 14   
## Target node size: 10   
## Variable importance mode: none   
## Splitrule: gini   
## OOB prediction error (Brier s.): 0.0597031

Predictions

predRF = predict(ames\_fit9, train)  
head(predRF)

## # A tibble: 6 × 1  
## .pred\_class  
## <fct>   
## 1 No   
## 2 No   
## 3 No   
## 4 No   
## 5 No   
## 6 No

Confusion matrix

confusionMatrix(predRF$.pred\_class, train$Above\_Median, positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 730 8  
## Yes 9 768  
##   
## Accuracy : 0.9888   
## 95% CI : (0.9821, 0.9935)  
## No Information Rate : 0.5122   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9775   
##   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.9897   
## Specificity : 0.9878   
## Pos Pred Value : 0.9884   
## Neg Pred Value : 0.9892   
## Prevalence : 0.5122   
## Detection Rate : 0.5069   
## Detection Prevalence : 0.5129   
## Balanced Accuracy : 0.9888   
##   
## 'Positive' Class : Yes   
##

Set up folds for cross-validation again.

set.seed(123)  
rf\_folds = vfold\_cv(train, v = 5)

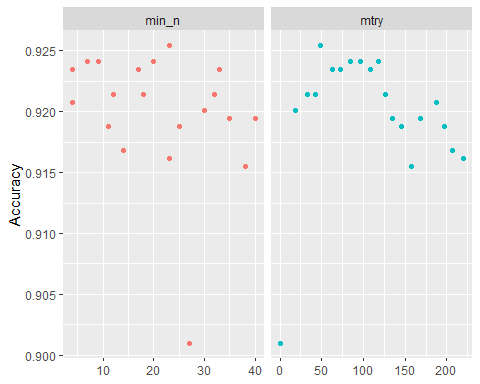
Random forest with an R-defined tuning grid (this model took about 5 minutes to run)

rf\_model = rand\_forest(mtry = tune(), min\_n = tune(), trees = 500) %>% #add tuning of mtry and min\_n parameters  
 #setting trees to 100 here should also speed things up a bit, but more trees might be better  
 set\_engine("ranger", importance = "permutation") %>% #added importance metric  
 set\_mode("classification")  
  
ames\_wflow3 =  
 workflow() %>%  
 add\_model(rf\_model) %>%  
 add\_recipe(ames8\_recipe)  
  
set.seed(123)  
rf\_res = tune\_grid(  
 ames\_wflow3,  
 resamples = rf\_folds,  
 grid = 20 #try 20 different combinations of the random forest tuning parameters  
)

## i Creating pre-processing data to finalize unknown parameter: mtry

Look at parameter performance (borrowed from <https://juliasilge.com/blog/sf-trees-random-tuning/>)

rf\_res %>%  
 collect\_metrics() %>%  
 filter(.metric == "accuracy") %>%  
 dplyr::select(mean, min\_n, mtry) %>%  
 pivot\_longer(min\_n:mtry,  
 values\_to = "value",  
 names\_to = "parameter"  
 ) %>%  
 ggplot(aes(value, mean, color = parameter)) +  
 geom\_point(show.legend = FALSE) +  
 facet\_wrap(~parameter, scales = "free\_x") +  
 labs(x = NULL, y = "Accuracy")



Parameter performance.

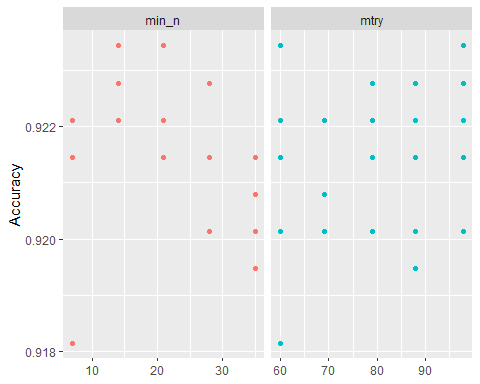
Refining the parameters

ames\_wflow3

## ══ Workflow ════════════════════════════════════════════════════════════════════  
## Preprocessor: Recipe  
## Model: rand\_forest()  
##   
## ── Preprocessor ────────────────────────────────────────────────────────────────  
## 2 Recipe Steps  
##   
## • step\_other()  
## • step\_dummy()  
##   
## ── Model ───────────────────────────────────────────────────────────────────────  
## Random Forest Model Specification (classification)  
##   
## Main Arguments:  
## mtry = tune()  
## trees = 500  
## min\_n = tune()  
##   
## Engine-Specific Arguments:  
## importance = permutation  
##   
## Computational engine: ranger

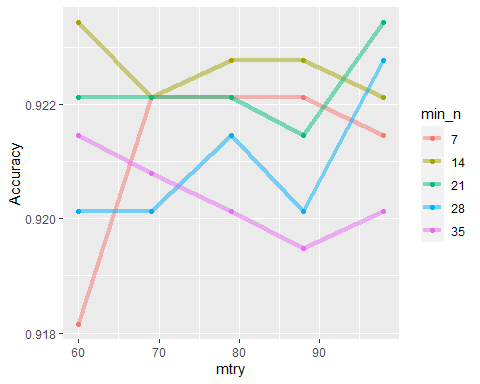
rf\_grid = grid\_regular(  
 mtry(range = c(60, 98)), #these values determined through significant trial and error  
 min\_n(range = c(7, 35)), #these values determined through significant trial and error  
 levels = 5  
)  
  
set.seed(123)  
rf\_res\_tuned = tune\_grid(  
 ames\_wflow3,  
 resamples = rf\_folds,  
 grid = rf\_grid #use the tuning grid  
)

rf\_res\_tuned %>%  
 collect\_metrics() %>%  
 filter(.metric == "accuracy") %>%  
 dplyr::select(mean, min\_n, mtry) %>%  
 pivot\_longer(min\_n:mtry,  
 values\_to = "value",  
 names\_to = "parameter"  
 ) %>%  
 ggplot(aes(value, mean, color = parameter)) +  
 geom\_point(show.legend = FALSE) +  
 facet\_wrap(~parameter, scales = "free\_x") +  
 labs(x = NULL, y = "Accuracy")



An alternate view of the parameters

rf\_res\_tuned %>%  
 collect\_metrics() %>%  
 filter(.metric == "accuracy") %>%  
 mutate(min\_n = factor(min\_n)) %>%  
 ggplot(aes(mtry, mean, color = min\_n)) +  
 geom\_line(alpha = 0.5, size = 1.5) +  
 geom\_point() +  
 labs(y = "Accuracy")



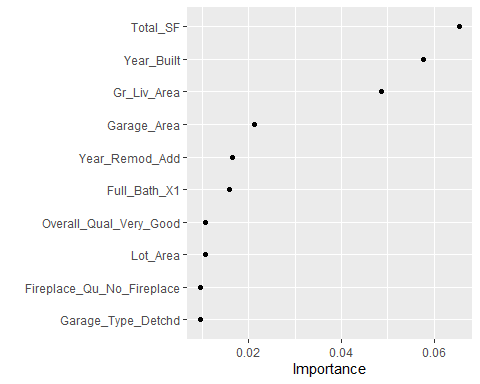
best\_rf = select\_best(rf\_res\_tuned, "accuracy")  
  
final\_rf = finalize\_workflow(  
 ames\_wflow3,  
 best\_rf  
)  
  
final\_rf

## ══ Workflow ════════════════════════════════════════════════════════════════════  
## Preprocessor: Recipe  
## Model: rand\_forest()  
##   
## ── Preprocessor ────────────────────────────────────────────────────────────────  
## 2 Recipe Steps  
##   
## • step\_other()  
## • step\_dummy()  
##   
## ── Model ───────────────────────────────────────────────────────────────────────  
## Random Forest Model Specification (classification)  
##   
## Main Arguments:  
## mtry = 60  
## trees = 500  
## min\_n = 14  
##   
## Engine-Specific Arguments:  
## importance = permutation  
##   
## Computational engine: ranger

#fit the finalized workflow to our training data  
final\_rf\_fit = fit(final\_rf, train)

Check out variable importance

final\_rf\_fit %>% extract\_fit\_parsnip() %>% vip(geom = "point")



Predictions

trainpredrf = predict(final\_rf\_fit, train)  
head(trainpredrf)

## # A tibble: 6 × 1  
## .pred\_class  
## <fct>   
## 1 No   
## 2 No   
## 3 No   
## 4 No   
## 5 No   
## 6 No

Confusion matrix

confusionMatrix(trainpredrf$.pred\_class, train$Above\_Median,  
positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 732 10  
## Yes 7 766  
##   
## Accuracy : 0.9888   
## 95% CI : (0.9821, 0.9935)  
## No Information Rate : 0.5122   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9775   
##   
## Mcnemar's Test P-Value : 0.6276   
##   
## Sensitivity : 0.9871   
## Specificity : 0.9905   
## Pos Pred Value : 0.9909   
## Neg Pred Value : 0.9865   
## Prevalence : 0.5122   
## Detection Rate : 0.5056   
## Detection Prevalence : 0.5102   
## Balanced Accuracy : 0.9888   
##   
## 'Positive' Class : Yes   
##

Predictions on test

testpredrf = predict(final\_rf\_fit, test)  
head(testpredrf)

## # A tibble: 6 × 1  
## .pred\_class  
## <fct>   
## 1 No   
## 2 Yes   
## 3 Yes   
## 4 No   
## 5 Yes   
## 6 No

confusionMatrix(testpredrf$.pred\_class, test$Above\_Median,  
 positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 172 16  
## Yes 13 179  
##   
## Accuracy : 0.9237   
## 95% CI : (0.8922, 0.9483)  
## No Information Rate : 0.5132   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.8473   
##   
## Mcnemar's Test P-Value : 0.7103   
##   
## Sensitivity : 0.9179   
## Specificity : 0.9297   
## Pos Pred Value : 0.9323   
## Neg Pred Value : 0.9149   
## Prevalence : 0.5132   
## Detection Rate : 0.4711   
## Detection Prevalence : 0.5053   
## Balanced Accuracy : 0.9238   
##   
## 'Positive' Class : Yes   
##

## Summary

Among the different models created (Logistic Regression, Lasso Regression, Classification Trees, and Random Forest), the best model was examined when using Random Forest with an accuracy of 0.9835 on the training set and 0.9237 on the test set. As suspected, Year Built was the strongest predictor. Further analysis demonstrated Gross Living Area was not as important to Year

tab = as.table(rbind(c(0.8825, 0.8816), c(NA, 0.8921), c(0.903, 0.8974), c(0.9835, 0.9237)))  
dimnames(tab) = list(Model = c("Logistic Regression", "Lasso Regression", "Classification Trees", "Random Forest"),  
 Tests = c("Train", "Test"))  
tab

## Tests  
## Model Train Test  
## Logistic Regression 0.8825 0.8816  
## Lasso Regression 0.8921  
## Classification Trees 0.9030 0.8974  
## Random Forest 0.9835 0.9237

Save the model to a file to load later

# saveRDS(final\_rf\_fit, "final\_rf\_fit.rds")

Load the model

# final\_rf\_fit = readRDS("final\_rf\_fit.rds")