house\_price\_pre\_3

Ravi Hela

22/03/2020

## Load libraries

library(tidymodels)

## -- Attaching packages --------------

## v broom 0.5.6 v recipes 0.1.12  
## v dials 0.0.6 v rsample 0.0.6   
## v dplyr 0.8.5 v tibble 3.0.1   
## v ggplot2 3.3.0 v tune 0.1.0   
## v infer 0.5.1 v workflows 0.1.1   
## v parsnip 0.1.1 v yardstick 0.0.6   
## v purrr 0.3.4

## -- Conflicts -----------------------  
## x purrr::discard() masks scales::discard()  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()  
## x ggplot2::margin() masks dials::margin()  
## x recipes::step() masks stats::step()

library(readr)

##   
## Attaching package: 'readr'

## The following object is masked from 'package:yardstick':  
##   
## spec

## The following object is masked from 'package:scales':  
##   
## col\_factor

library(tidyverse)

## -- Attaching packages --------------

## v tidyr 1.0.3 v forcats 0.5.0  
## v stringr 1.4.0

## -- Conflicts -----------------------  
## x readr::col\_factor() masks scales::col\_factor()  
## x purrr::discard() masks scales::discard()  
## x dplyr::filter() masks stats::filter()  
## x stringr::fixed() masks recipes::fixed()  
## x dplyr::lag() masks stats::lag()  
## x ggplot2::margin() masks dials::margin()  
## x readr::spec() masks yardstick::spec()

library(lubridate)

##   
## Attaching package: 'lubridate'

## The following objects are masked from 'package:dplyr':  
##   
## intersect, setdiff, union

## The following objects are masked from 'package:base':  
##   
## date, intersect, setdiff, union

library(skimr)  
library(DataExplorer)  
library(tidyquant)

## Loading required package: PerformanceAnalytics

## Loading required package: xts

## Loading required package: zoo

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

##   
## Attaching package: 'xts'

## The following objects are masked from 'package:dplyr':  
##   
## first, last

##   
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:graphics':  
##   
## legend

## Loading required package: quantmod

## Loading required package: TTR

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

## Version 0.4-0 included new data defaults. See ?getSymbols.

## == Need to Learn tidyquant? ========  
## Business Science offers a 1-hour course - Learning Lab #9: Performance Analysis & Portfolio Optimization with tidyquant!  
## </> Learn more at: https://university.business-science.io/p/learning-labs-pro </>

library(corrr)

##   
## Attaching package: 'corrr'

## The following object is masked from 'package:skimr':  
##   
## focus

library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following objects are masked from 'package:yardstick':  
##   
## precision, recall, sensitivity, specificity

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

library(vip)

##   
## Attaching package: 'vip'

## The following object is masked from 'package:utils':  
##   
## vi

library(Hmisc)

## Loading required package: survival

##   
## Attaching package: 'survival'

## The following object is masked from 'package:caret':  
##   
## cluster

## Loading required package: Formula

##   
## Attaching package: 'Hmisc'

## The following object is masked from 'package:quantmod':  
##   
## Lag

## The following object is masked from 'package:parsnip':  
##   
## translate

## The following objects are masked from 'package:dplyr':  
##   
## src, summarize

## The following objects are masked from 'package:base':  
##   
## format.pval, units

## define some useful functions

#function to help filter corelated variables  
split\_arrange\_names <- function(x){  
   
 return(paste0(sort(unlist(str\_split(x, "\_"))), collapse = ""))  
}  
  
#get simplifeid correlation dataframe  
get\_corr\_df\_simple <- function(df, cut\_off = NA) {  
   
 simpl\_df <-  
 df %>% select\_if(is.numeric) %>% correlate(use = "pairwise.complete.obs") %>%  
 pivot\_longer(  
 -rowname,  
 names\_to = "features",  
 values\_to = "corr",  
 values\_drop\_na = T  
 ) %>%  
 mutate(t1 = paste(rowname, features, sep = "\_")) %>%  
 mutate(t2 = sapply(t1, split\_arrange\_names)) %>%  
 arrange(t2) %>%  
 group\_by(t2) %>%  
 mutate(rank = row\_number(t2)) %>%  
 ungroup %>% filter(rank == 1) %>%  
 select(rowname, features, corr) %>%  
 arrange(desc(abs(corr)))   
   
 if(is.na(cut\_off)){  
 return(simpl\_df)  
 }else{  
   
 return(simpl\_df %>% filter(abs(corr) >= cut\_off))  
 }  
   
}  
  
#get model metrics in a dataframe  
get\_model\_metrics <- function(model, df){  
   
 #model\_type <- str\_replace(class(model)[[1]] , "\_", "")  
 model\_type <- model[[2]]  
   
 #print(model)  
 # class(model)  
 # print(model[[1]])  
 # print(model[[2]])  
   
 #print(deparse(substitute(model)))  
  
 #print(names({{model}}))  
 df %>%  
 bind\_cols(predict(model[[1]], df)) %>%  
 select(log\_sale\_price, predicted = .pred) %>%  
  
 #add squared error  
 mutate(  
 sq\_error\_log = (predicted - log\_sale\_price) ^ 2,  
 sale\_price = exp(log\_sale\_price),  
 predicted\_sp = exp(predicted),  
 sq\_error = predicted\_sp - sale\_price  
 ) %>%  
  
 #plot erro  
 #ggplot(aes(x = sq\_error, y = predicted\_lm)) + geom\_point() + theme\_light()  
 mutate(truth = log(sale\_price),  
 estimate = log(predicted\_sp)) %>%  
  
 #get model metrics  
 metrics(truth, estimate) %>%  
 select(-.estimator) %>%  
 mutate(m\_type = model\_type) %>%  
 filter(.metric == "rmse")  
 }

## Read data

train <- read\_csv("train.csv")

## Parsed with column specification:  
## cols(  
## .default = col\_character(),  
## Id = col\_double(),  
## MSSubClass = col\_double(),  
## LotFrontage = col\_double(),  
## LotArea = col\_double(),  
## OverallQual = col\_double(),  
## OverallCond = col\_double(),  
## YearBuilt = col\_double(),  
## YearRemodAdd = col\_double(),  
## MasVnrArea = col\_double(),  
## BsmtFinSF1 = col\_double(),  
## BsmtFinSF2 = col\_double(),  
## BsmtUnfSF = col\_double(),  
## TotalBsmtSF = col\_double(),  
## `1stFlrSF` = col\_double(),  
## `2ndFlrSF` = col\_double(),  
## LowQualFinSF = col\_double(),  
## GrLivArea = col\_double(),  
## BsmtFullBath = col\_double(),  
## BsmtHalfBath = col\_double(),  
## FullBath = col\_double()  
## # ... with 18 more columns  
## )

## See spec(...) for full column specifications.

test <- read\_csv("test.csv")

## Parsed with column specification:  
## cols(  
## .default = col\_character(),  
## Id = col\_double(),  
## MSSubClass = col\_double(),  
## LotFrontage = col\_double(),  
## LotArea = col\_double(),  
## OverallQual = col\_double(),  
## OverallCond = col\_double(),  
## YearBuilt = col\_double(),  
## YearRemodAdd = col\_double(),  
## MasVnrArea = col\_double(),  
## BsmtFinSF1 = col\_double(),  
## BsmtFinSF2 = col\_double(),  
## BsmtUnfSF = col\_double(),  
## TotalBsmtSF = col\_double(),  
## `1stFlrSF` = col\_double(),  
## `2ndFlrSF` = col\_double(),  
## LowQualFinSF = col\_double(),  
## GrLivArea = col\_double(),  
## BsmtFullBath = col\_double(),  
## BsmtHalfBath = col\_double(),  
## FullBath = col\_double()  
## # ... with 17 more columns  
## )  
## See spec(...) for full column specifications.

names(train) <- make.names(names(train))  
names(test) <- make.names(names(test))

## combinie test and train for eda

lapply(list(train = train, test = test), skim)

## $train  
## -- Data Summary ------------------------  
## Values  
## Name X[[i]]  
## Number of rows 1460   
## Number of columns 81   
## \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
## Column type frequency:   
## character 43   
## numeric 38   
## \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
## Group variables None   
##   
## -- Variable type: character ----------------------------------------------------  
## # A tibble: 43 x 8  
## skim\_variable n\_missing complete\_rate min max empty n\_unique whitespace  
## \* <chr> <int> <dbl> <int> <int> <int> <int> <int>  
## 1 MSZoning 0 1 2 7 0 5 0  
## 2 Street 0 1 4 4 0 2 0  
## 3 Alley 1369 0.0623 4 4 0 2 0  
## 4 LotShape 0 1 3 3 0 4 0  
## 5 LandContour 0 1 3 3 0 4 0  
## 6 Utilities 0 1 6 6 0 2 0  
## 7 LotConfig 0 1 3 7 0 5 0  
## 8 LandSlope 0 1 3 3 0 3 0  
## 9 Neighborhood 0 1 5 7 0 25 0  
## 10 Condition1 0 1 4 6 0 9 0  
## 11 Condition2 0 1 4 6 0 8 0  
## 12 BldgType 0 1 4 6 0 5 0  
## 13 HouseStyle 0 1 4 6 0 8 0  
## 14 RoofStyle 0 1 3 7 0 6 0  
## 15 RoofMatl 0 1 4 7 0 8 0  
## 16 Exterior1st 0 1 5 7 0 15 0  
## 17 Exterior2nd 0 1 5 7 0 16 0  
## 18 MasVnrType 8 0.995 4 7 0 4 0  
## 19 ExterQual 0 1 2 2 0 4 0  
## 20 ExterCond 0 1 2 2 0 5 0  
## 21 Foundation 0 1 4 6 0 6 0  
## 22 BsmtQual 37 0.975 2 2 0 4 0  
## 23 BsmtCond 37 0.975 2 2 0 4 0  
## 24 BsmtExposure 38 0.974 2 2 0 4 0  
## 25 BsmtFinType1 37 0.975 3 3 0 6 0  
## 26 BsmtFinType2 38 0.974 3 3 0 6 0  
## 27 Heating 0 1 4 5 0 6 0  
## 28 HeatingQC 0 1 2 2 0 5 0  
## 29 CentralAir 0 1 1 1 0 2 0  
## 30 Electrical 1 0.999 3 5 0 5 0  
## 31 KitchenQual 0 1 2 2 0 4 0  
## 32 Functional 0 1 3 4 0 7 0  
## 33 FireplaceQu 690 0.527 2 2 0 5 0  
## 34 GarageType 81 0.945 6 7 0 6 0  
## 35 GarageFinish 81 0.945 3 3 0 3 0  
## 36 GarageQual 81 0.945 2 2 0 5 0  
## 37 GarageCond 81 0.945 2 2 0 5 0  
## 38 PavedDrive 0 1 1 1 0 3 0  
## 39 PoolQC 1453 0.00479 2 2 0 3 0  
## 40 Fence 1179 0.192 4 5 0 4 0  
## 41 MiscFeature 1406 0.0370 4 4 0 4 0  
## 42 SaleType 0 1 2 5 0 9 0  
## 43 SaleCondition 0 1 6 7 0 6 0  
##   
## -- Variable type: numeric ------------------------------------------------------  
## # A tibble: 38 x 11  
## skim\_variable n\_missing complete\_rate mean sd p0 p25  
## \* <chr> <int> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 Id 0 1 730. 422. 1 366.  
## 2 MSSubClass 0 1 56.9 42.3 20 20   
## 3 LotFrontage 259 0.823 70.0 24.3 21 59   
## 4 LotArea 0 1 10517. 9981. 1300 7554.  
## 5 OverallQual 0 1 6.10 1.38 1 5   
## 6 OverallCond 0 1 5.58 1.11 1 5   
## 7 YearBuilt 0 1 1971. 30.2 1872 1954   
## 8 YearRemodAdd 0 1 1985. 20.6 1950 1967   
## 9 MasVnrArea 8 0.995 104. 181. 0 0   
## 10 BsmtFinSF1 0 1 444. 456. 0 0   
## 11 BsmtFinSF2 0 1 46.5 161. 0 0   
## 12 BsmtUnfSF 0 1 567. 442. 0 223   
## 13 TotalBsmtSF 0 1 1057. 439. 0 796.  
## 14 X1stFlrSF 0 1 1163. 387. 334 882   
## 15 X2ndFlrSF 0 1 347. 437. 0 0   
## 16 LowQualFinSF 0 1 5.84 48.6 0 0   
## 17 GrLivArea 0 1 1515. 525. 334 1130.  
## 18 BsmtFullBath 0 1 0.425 0.519 0 0   
## 19 BsmtHalfBath 0 1 0.0575 0.239 0 0   
## 20 FullBath 0 1 1.57 0.551 0 1   
## 21 HalfBath 0 1 0.383 0.503 0 0   
## 22 BedroomAbvGr 0 1 2.87 0.816 0 2   
## 23 KitchenAbvGr 0 1 1.05 0.220 0 1   
## 24 TotRmsAbvGrd 0 1 6.52 1.63 2 5   
## 25 Fireplaces 0 1 0.613 0.645 0 0   
## 26 GarageYrBlt 81 0.945 1979. 24.7 1900 1961   
## 27 GarageCars 0 1 1.77 0.747 0 1   
## 28 GarageArea 0 1 473. 214. 0 334.  
## 29 WoodDeckSF 0 1 94.2 125. 0 0   
## 30 OpenPorchSF 0 1 46.7 66.3 0 0   
## 31 EnclosedPorch 0 1 22.0 61.1 0 0   
## 32 X3SsnPorch 0 1 3.41 29.3 0 0   
## 33 ScreenPorch 0 1 15.1 55.8 0 0   
## 34 PoolArea 0 1 2.76 40.2 0 0   
## 35 MiscVal 0 1 43.5 496. 0 0   
## 36 MoSold 0 1 6.32 2.70 1 5   
## 37 YrSold 0 1 2008. 1.33 2006 2007   
## 38 SalePrice 0 1 180921. 79443. 34900 129975   
## p50 p75 p100 hist   
## \* <dbl> <dbl> <dbl> <chr>  
## 1 730. 1095. 1460 <U+2587><U+2587><U+2587><U+2587><U+2587>  
## 2 50 70 190 <U+2587><U+2585><U+2582><U+2581><U+2581>  
## 3 69 80 313 <U+2587><U+2583><U+2581><U+2581><U+2581>  
## 4 9478. 11602. 215245 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 5 6 7 10 <U+2581><U+2582><U+2587><U+2585><U+2581>  
## 6 5 6 9 <U+2581><U+2581><U+2587><U+2585><U+2581>  
## 7 1973 2000 2010 <U+2581><U+2582><U+2583><U+2586><U+2587>  
## 8 1994 2004 2010 <U+2585><U+2582><U+2582><U+2583><U+2587>  
## 9 0 166 1600 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 10 384. 712. 5644 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 11 0 0 1474 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 12 478. 808 2336 <U+2587><U+2585><U+2582><U+2581><U+2581>  
## 13 992. 1298. 6110 <U+2587><U+2583><U+2581><U+2581><U+2581>  
## 14 1087 1391. 4692 <U+2587><U+2585><U+2581><U+2581><U+2581>  
## 15 0 728 2065 <U+2587><U+2583><U+2582><U+2581><U+2581>  
## 16 0 0 572 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 17 1464 1777. 5642 <U+2587><U+2587><U+2581><U+2581><U+2581>  
## 18 0 1 3 <U+2587><U+2586><U+2581><U+2581><U+2581>  
## 19 0 0 2 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 20 2 2 3 <U+2581><U+2587><U+2581><U+2587><U+2581>  
## 21 0 1 2 <U+2587><U+2581><U+2585><U+2581><U+2581>  
## 22 3 3 8 <U+2581><U+2587><U+2582><U+2581><U+2581>  
## 23 1 1 3 <U+2581><U+2587><U+2581><U+2581><U+2581>  
## 24 6 7 14 <U+2582><U+2587><U+2587><U+2581><U+2581>  
## 25 1 1 3 <U+2587><U+2587><U+2581><U+2581><U+2581>  
## 26 1980 2002 2010 <U+2581><U+2581><U+2585><U+2585><U+2587>  
## 27 2 2 4 <U+2581><U+2583><U+2587><U+2582><U+2581>  
## 28 480 576 1418 <U+2582><U+2587><U+2583><U+2581><U+2581>  
## 29 0 168 857 <U+2587><U+2582><U+2581><U+2581><U+2581>  
## 30 25 68 547 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 31 0 0 552 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 32 0 0 508 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 33 0 0 480 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 34 0 0 738 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 35 0 0 15500 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 36 6 8 12 <U+2583><U+2586><U+2587><U+2583><U+2583>  
## 37 2008 2009 2010 <U+2587><U+2587><U+2587><U+2587><U+2585>  
## 38 163000 214000 755000 <U+2587><U+2585><U+2581><U+2581><U+2581>  
##   
## $test  
## -- Data Summary ------------------------  
## Values  
## Name X[[i]]  
## Number of rows 1459   
## Number of columns 80   
## \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
## Column type frequency:   
## character 43   
## numeric 37   
## \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
## Group variables None   
##   
## -- Variable type: character ----------------------------------------------------  
## # A tibble: 43 x 8  
## skim\_variable n\_missing complete\_rate min max empty n\_unique whitespace  
## \* <chr> <int> <dbl> <int> <int> <int> <int> <int>  
## 1 MSZoning 4 0.997 2 7 0 5 0  
## 2 Street 0 1 4 4 0 2 0  
## 3 Alley 1352 0.0733 4 4 0 2 0  
## 4 LotShape 0 1 3 3 0 4 0  
## 5 LandContour 0 1 3 3 0 4 0  
## 6 Utilities 2 0.999 6 6 0 1 0  
## 7 LotConfig 0 1 3 7 0 5 0  
## 8 LandSlope 0 1 3 3 0 3 0  
## 9 Neighborhood 0 1 5 7 0 25 0  
## 10 Condition1 0 1 4 6 0 9 0  
## 11 Condition2 0 1 4 6 0 5 0  
## 12 BldgType 0 1 4 6 0 5 0  
## 13 HouseStyle 0 1 4 6 0 7 0  
## 14 RoofStyle 0 1 3 7 0 6 0  
## 15 RoofMatl 0 1 7 7 0 4 0  
## 16 Exterior1st 1 0.999 6 7 0 13 0  
## 17 Exterior2nd 1 0.999 5 7 0 15 0  
## 18 MasVnrType 16 0.989 4 7 0 4 0  
## 19 ExterQual 0 1 2 2 0 4 0  
## 20 ExterCond 0 1 2 2 0 5 0  
## 21 Foundation 0 1 4 6 0 6 0  
## 22 BsmtQual 44 0.970 2 2 0 4 0  
## 23 BsmtCond 45 0.969 2 2 0 4 0  
## 24 BsmtExposure 44 0.970 2 2 0 4 0  
## 25 BsmtFinType1 42 0.971 3 3 0 6 0  
## 26 BsmtFinType2 42 0.971 3 3 0 6 0  
## 27 Heating 0 1 4 4 0 4 0  
## 28 HeatingQC 0 1 2 2 0 5 0  
## 29 CentralAir 0 1 1 1 0 2 0  
## 30 Electrical 0 1 5 5 0 4 0  
## 31 KitchenQual 1 0.999 2 2 0 4 0  
## 32 Functional 2 0.999 3 4 0 7 0  
## 33 FireplaceQu 730 0.500 2 2 0 5 0  
## 34 GarageType 76 0.948 6 7 0 6 0  
## 35 GarageFinish 78 0.947 3 3 0 3 0  
## 36 GarageQual 78 0.947 2 2 0 4 0  
## 37 GarageCond 78 0.947 2 2 0 5 0  
## 38 PavedDrive 0 1 1 1 0 3 0  
## 39 PoolQC 1456 0.00206 2 2 0 2 0  
## 40 Fence 1169 0.199 4 5 0 4 0  
## 41 MiscFeature 1408 0.0350 4 4 0 3 0  
## 42 SaleType 1 0.999 2 5 0 9 0  
## 43 SaleCondition 0 1 6 7 0 6 0  
##   
## -- Variable type: numeric ------------------------------------------------------  
## # A tibble: 37 x 11  
## skim\_variable n\_missing complete\_rate mean sd p0 p25 p50  
## \* <chr> <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 Id 0 1 2190 421. 1461 1826. 2190   
## 2 MSSubClass 0 1 57.4 42.7 20 20 50   
## 3 LotFrontage 227 0.844 68.6 22.4 21 58 67   
## 4 LotArea 0 1 9819. 4956. 1470 7391 9399   
## 5 OverallQual 0 1 6.08 1.44 1 5 6   
## 6 OverallCond 0 1 5.55 1.11 1 5 5   
## 7 YearBuilt 0 1 1971. 30.4 1879 1953 1973   
## 8 YearRemodAdd 0 1 1984. 21.1 1950 1963 1992   
## 9 MasVnrArea 15 0.990 101. 178. 0 0 0   
## 10 BsmtFinSF1 1 0.999 439. 455. 0 0 350.  
## 11 BsmtFinSF2 1 0.999 52.6 177. 0 0 0   
## 12 BsmtUnfSF 1 0.999 554. 437. 0 219. 460   
## 13 TotalBsmtSF 1 0.999 1046. 443. 0 784 988   
## 14 X1stFlrSF 0 1 1157. 398. 407 874. 1079   
## 15 X2ndFlrSF 0 1 326. 421. 0 0 0   
## 16 LowQualFinSF 0 1 3.54 44.0 0 0 0   
## 17 GrLivArea 0 1 1486. 486. 407 1118. 1432   
## 18 BsmtFullBath 2 0.999 0.434 0.531 0 0 0   
## 19 BsmtHalfBath 2 0.999 0.0652 0.252 0 0 0   
## 20 FullBath 0 1 1.57 0.555 0 1 2   
## 21 HalfBath 0 1 0.378 0.503 0 0 0   
## 22 BedroomAbvGr 0 1 2.85 0.830 0 2 3   
## 23 KitchenAbvGr 0 1 1.04 0.208 0 1 1   
## 24 TotRmsAbvGrd 0 1 6.39 1.51 3 5 6   
## 25 Fireplaces 0 1 0.581 0.647 0 0 0   
## 26 GarageYrBlt 78 0.947 1978. 26.4 1895 1959 1979   
## 27 GarageCars 1 0.999 1.77 0.776 0 1 2   
## 28 GarageArea 1 0.999 473. 217. 0 318 480   
## 29 WoodDeckSF 0 1 93.2 128. 0 0 0   
## 30 OpenPorchSF 0 1 48.3 68.9 0 0 28   
## 31 EnclosedPorch 0 1 24.2 67.2 0 0 0   
## 32 X3SsnPorch 0 1 1.79 20.2 0 0 0   
## 33 ScreenPorch 0 1 17.1 56.6 0 0 0   
## 34 PoolArea 0 1 1.74 30.5 0 0 0   
## 35 MiscVal 0 1 58.2 631. 0 0 0   
## 36 MoSold 0 1 6.10 2.72 1 4 6   
## 37 YrSold 0 1 2008. 1.30 2006 2007 2008   
## p75 p100 hist   
## \* <dbl> <dbl> <chr>  
## 1 2554. 2919 <U+2587><U+2587><U+2587><U+2587><U+2587>  
## 2 70 190 <U+2587><U+2585><U+2582><U+2581><U+2581>  
## 3 80 200 <U+2583><U+2587><U+2581><U+2581><U+2581>  
## 4 11518. 56600 <U+2587><U+2582><U+2581><U+2581><U+2581>  
## 5 7 10 <U+2581><U+2581><U+2587><U+2585><U+2581>  
## 6 6 9 <U+2581><U+2581><U+2587><U+2585><U+2581>  
## 7 2001 2010 <U+2581><U+2582><U+2583><U+2586><U+2587>  
## 8 2004 2010 <U+2585><U+2582><U+2582><U+2583><U+2587>  
## 9 164 1290 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 10 754. 4010 <U+2587><U+2582><U+2581><U+2581><U+2581>  
## 11 0 1526 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 12 798. 2140 <U+2587><U+2586><U+2582><U+2581><U+2581>  
## 13 1305 5095 <U+2587><U+2587><U+2581><U+2581><U+2581>  
## 14 1382. 5095 <U+2587><U+2583><U+2581><U+2581><U+2581>  
## 15 676 1862 <U+2587><U+2583><U+2582><U+2581><U+2581>  
## 16 0 1064 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 17 1721 5095 <U+2587><U+2587><U+2581><U+2581><U+2581>  
## 18 1 3 <U+2587><U+2586><U+2581><U+2581><U+2581>  
## 19 0 2 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 20 2 4 <U+2581><U+2587><U+2587><U+2581><U+2581>  
## 21 1 2 <U+2587><U+2581><U+2585><U+2581><U+2581>  
## 22 3 6 <U+2581><U+2583><U+2587><U+2582><U+2581>  
## 23 1 2 <U+2581><U+2581><U+2587><U+2581><U+2581>  
## 24 7 15 <U+2585><U+2587><U+2583><U+2581><U+2581>  
## 25 1 4 <U+2587><U+2587><U+2581><U+2581><U+2581>  
## 26 2002 2207 <U+2582><U+2587><U+2581><U+2581><U+2581>  
## 27 2 5 <U+2585><U+2587><U+2582><U+2581><U+2581>  
## 28 576 1488 <U+2583><U+2587><U+2583><U+2581><U+2581>  
## 29 168 1424 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 30 72 742 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 31 0 1012 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 32 0 360 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 33 0 576 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 34 0 800 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 35 0 17000 <U+2587><U+2581><U+2581><U+2581><U+2581>  
## 36 8 12 <U+2585><U+2586><U+2587><U+2583><U+2583>  
## 37 2009 2010 <U+2587><U+2587><U+2587><U+2587><U+2583>

test$SalePrice <- NA  
test$set <- "test"  
train$set <- "train"  
house\_comb <- bind\_rows(train, test)  
skim(house\_comb)

Data summary

|  |  |
| --- | --- |
| Name | house\_comb |
| Number of rows | 2919 |
| Number of columns | 82 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Column type frequency: |  |
| character | 44 |
| numeric | 38 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Group variables | None |

**Variable type: character**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| skim\_variable | n\_missing | complete\_rate | min | max | empty | n\_unique | whitespace |
| MSZoning | 4 | 1.00 | 2 | 7 | 0 | 5 | 0 |
| Street | 0 | 1.00 | 4 | 4 | 0 | 2 | 0 |
| Alley | 2721 | 0.07 | 4 | 4 | 0 | 2 | 0 |
| LotShape | 0 | 1.00 | 3 | 3 | 0 | 4 | 0 |
| LandContour | 0 | 1.00 | 3 | 3 | 0 | 4 | 0 |
| Utilities | 2 | 1.00 | 6 | 6 | 0 | 2 | 0 |
| LotConfig | 0 | 1.00 | 3 | 7 | 0 | 5 | 0 |
| LandSlope | 0 | 1.00 | 3 | 3 | 0 | 3 | 0 |
| Neighborhood | 0 | 1.00 | 5 | 7 | 0 | 25 | 0 |
| Condition1 | 0 | 1.00 | 4 | 6 | 0 | 9 | 0 |
| Condition2 | 0 | 1.00 | 4 | 6 | 0 | 8 | 0 |
| BldgType | 0 | 1.00 | 4 | 6 | 0 | 5 | 0 |
| HouseStyle | 0 | 1.00 | 4 | 6 | 0 | 8 | 0 |
| RoofStyle | 0 | 1.00 | 3 | 7 | 0 | 6 | 0 |
| RoofMatl | 0 | 1.00 | 4 | 7 | 0 | 8 | 0 |
| Exterior1st | 1 | 1.00 | 5 | 7 | 0 | 15 | 0 |
| Exterior2nd | 1 | 1.00 | 5 | 7 | 0 | 16 | 0 |
| MasVnrType | 24 | 0.99 | 4 | 7 | 0 | 4 | 0 |
| ExterQual | 0 | 1.00 | 2 | 2 | 0 | 4 | 0 |
| ExterCond | 0 | 1.00 | 2 | 2 | 0 | 5 | 0 |
| Foundation | 0 | 1.00 | 4 | 6 | 0 | 6 | 0 |
| BsmtQual | 81 | 0.97 | 2 | 2 | 0 | 4 | 0 |
| BsmtCond | 82 | 0.97 | 2 | 2 | 0 | 4 | 0 |
| BsmtExposure | 82 | 0.97 | 2 | 2 | 0 | 4 | 0 |
| BsmtFinType1 | 79 | 0.97 | 3 | 3 | 0 | 6 | 0 |
| BsmtFinType2 | 80 | 0.97 | 3 | 3 | 0 | 6 | 0 |
| Heating | 0 | 1.00 | 4 | 5 | 0 | 6 | 0 |
| HeatingQC | 0 | 1.00 | 2 | 2 | 0 | 5 | 0 |
| CentralAir | 0 | 1.00 | 1 | 1 | 0 | 2 | 0 |
| Electrical | 1 | 1.00 | 3 | 5 | 0 | 5 | 0 |
| KitchenQual | 1 | 1.00 | 2 | 2 | 0 | 4 | 0 |
| Functional | 2 | 1.00 | 3 | 4 | 0 | 7 | 0 |
| FireplaceQu | 1420 | 0.51 | 2 | 2 | 0 | 5 | 0 |
| GarageType | 157 | 0.95 | 6 | 7 | 0 | 6 | 0 |
| GarageFinish | 159 | 0.95 | 3 | 3 | 0 | 3 | 0 |
| GarageQual | 159 | 0.95 | 2 | 2 | 0 | 5 | 0 |
| GarageCond | 159 | 0.95 | 2 | 2 | 0 | 5 | 0 |
| PavedDrive | 0 | 1.00 | 1 | 1 | 0 | 3 | 0 |
| PoolQC | 2909 | 0.00 | 2 | 2 | 0 | 3 | 0 |
| Fence | 2348 | 0.20 | 4 | 5 | 0 | 4 | 0 |
| MiscFeature | 2814 | 0.04 | 4 | 4 | 0 | 4 | 0 |
| SaleType | 1 | 1.00 | 2 | 5 | 0 | 9 | 0 |
| SaleCondition | 0 | 1.00 | 6 | 7 | 0 | 6 | 0 |
| set | 0 | 1.00 | 4 | 5 | 0 | 2 | 0 |

**Variable type: numeric**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| skim\_variable | n\_missing | complete\_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
| Id | 0 | 1.00 | 1460.00 | 842.79 | 1 | 730.5 | 1460.0 | 2189.5 | 2919 | ▇▇▇▇▇ |
| MSSubClass | 0 | 1.00 | 57.14 | 42.52 | 20 | 20.0 | 50.0 | 70.0 | 190 | ▇▅▂▁▁ |
| LotFrontage | 486 | 0.83 | 69.31 | 23.34 | 21 | 59.0 | 68.0 | 80.0 | 313 | ▇▃▁▁▁ |
| LotArea | 0 | 1.00 | 10168.11 | 7887.00 | 1300 | 7478.0 | 9453.0 | 11570.0 | 215245 | ▇▁▁▁▁ |
| OverallQual | 0 | 1.00 | 6.09 | 1.41 | 1 | 5.0 | 6.0 | 7.0 | 10 | ▁▂▇▅▁ |
| OverallCond | 0 | 1.00 | 5.56 | 1.11 | 1 | 5.0 | 5.0 | 6.0 | 9 | ▁▁▇▅▁ |
| YearBuilt | 0 | 1.00 | 1971.31 | 30.29 | 1872 | 1953.5 | 1973.0 | 2001.0 | 2010 | ▁▂▃▆▇ |
| YearRemodAdd | 0 | 1.00 | 1984.26 | 20.89 | 1950 | 1965.0 | 1993.0 | 2004.0 | 2010 | ▅▂▂▃▇ |
| MasVnrArea | 23 | 0.99 | 102.20 | 179.33 | 0 | 0.0 | 0.0 | 164.0 | 1600 | ▇▁▁▁▁ |
| BsmtFinSF1 | 1 | 1.00 | 441.42 | 455.61 | 0 | 0.0 | 368.5 | 733.0 | 5644 | ▇▁▁▁▁ |
| BsmtFinSF2 | 1 | 1.00 | 49.58 | 169.21 | 0 | 0.0 | 0.0 | 0.0 | 1526 | ▇▁▁▁▁ |
| BsmtUnfSF | 1 | 1.00 | 560.77 | 439.54 | 0 | 220.0 | 467.0 | 805.5 | 2336 | ▇▅▂▁▁ |
| TotalBsmtSF | 1 | 1.00 | 1051.78 | 440.77 | 0 | 793.0 | 989.5 | 1302.0 | 6110 | ▇▃▁▁▁ |
| X1stFlrSF | 0 | 1.00 | 1159.58 | 392.36 | 334 | 876.0 | 1082.0 | 1387.5 | 5095 | ▇▃▁▁▁ |
| X2ndFlrSF | 0 | 1.00 | 336.48 | 428.70 | 0 | 0.0 | 0.0 | 704.0 | 2065 | ▇▃▂▁▁ |
| LowQualFinSF | 0 | 1.00 | 4.69 | 46.40 | 0 | 0.0 | 0.0 | 0.0 | 1064 | ▇▁▁▁▁ |
| GrLivArea | 0 | 1.00 | 1500.76 | 506.05 | 334 | 1126.0 | 1444.0 | 1743.5 | 5642 | ▇▇▁▁▁ |
| BsmtFullBath | 2 | 1.00 | 0.43 | 0.52 | 0 | 0.0 | 0.0 | 1.0 | 3 | ▇▆▁▁▁ |
| BsmtHalfBath | 2 | 1.00 | 0.06 | 0.25 | 0 | 0.0 | 0.0 | 0.0 | 2 | ▇▁▁▁▁ |
| FullBath | 0 | 1.00 | 1.57 | 0.55 | 0 | 1.0 | 2.0 | 2.0 | 4 | ▁▇▇▁▁ |
| HalfBath | 0 | 1.00 | 0.38 | 0.50 | 0 | 0.0 | 0.0 | 1.0 | 2 | ▇▁▅▁▁ |
| BedroomAbvGr | 0 | 1.00 | 2.86 | 0.82 | 0 | 2.0 | 3.0 | 3.0 | 8 | ▁▇▂▁▁ |
| KitchenAbvGr | 0 | 1.00 | 1.04 | 0.21 | 0 | 1.0 | 1.0 | 1.0 | 3 | ▁▇▁▁▁ |
| TotRmsAbvGrd | 0 | 1.00 | 6.45 | 1.57 | 2 | 5.0 | 6.0 | 7.0 | 15 | ▁▇▂▁▁ |
| Fireplaces | 0 | 1.00 | 0.60 | 0.65 | 0 | 0.0 | 1.0 | 1.0 | 4 | ▇▇▁▁▁ |
| GarageYrBlt | 159 | 0.95 | 1978.11 | 25.57 | 1895 | 1960.0 | 1979.0 | 2002.0 | 2207 | ▂▇▁▁▁ |
| GarageCars | 1 | 1.00 | 1.77 | 0.76 | 0 | 1.0 | 2.0 | 2.0 | 5 | ▅▇▂▁▁ |
| GarageArea | 1 | 1.00 | 472.87 | 215.39 | 0 | 320.0 | 480.0 | 576.0 | 1488 | ▃▇▃▁▁ |
| WoodDeckSF | 0 | 1.00 | 93.71 | 126.53 | 0 | 0.0 | 0.0 | 168.0 | 1424 | ▇▁▁▁▁ |
| OpenPorchSF | 0 | 1.00 | 47.49 | 67.58 | 0 | 0.0 | 26.0 | 70.0 | 742 | ▇▁▁▁▁ |
| EnclosedPorch | 0 | 1.00 | 23.10 | 64.24 | 0 | 0.0 | 0.0 | 0.0 | 1012 | ▇▁▁▁▁ |
| X3SsnPorch | 0 | 1.00 | 2.60 | 25.19 | 0 | 0.0 | 0.0 | 0.0 | 508 | ▇▁▁▁▁ |
| ScreenPorch | 0 | 1.00 | 16.06 | 56.18 | 0 | 0.0 | 0.0 | 0.0 | 576 | ▇▁▁▁▁ |
| PoolArea | 0 | 1.00 | 2.25 | 35.66 | 0 | 0.0 | 0.0 | 0.0 | 800 | ▇▁▁▁▁ |
| MiscVal | 0 | 1.00 | 50.83 | 567.40 | 0 | 0.0 | 0.0 | 0.0 | 17000 | ▇▁▁▁▁ |
| MoSold | 0 | 1.00 | 6.21 | 2.71 | 1 | 4.0 | 6.0 | 8.0 | 12 | ▅▆▇▃▃ |
| YrSold | 0 | 1.00 | 2007.79 | 1.31 | 2006 | 2007.0 | 2008.0 | 2009.0 | 2010 | ▇▇▇▇▃ |
| SalePrice | 1459 | 0.50 | 180921.20 | 79442.50 | 34900 | 129975.0 | 163000.0 | 214000.0 | 755000 | ▇▅▁▁▁ |

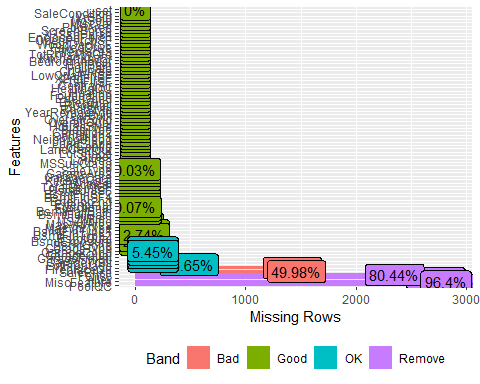
## Eda

Doing an EDA for data quality checks. We first check for missing values

profile\_missing(house\_comb) %>%  
 arrange(desc(pct\_missing))

## # A tibble: 82 x 3  
## feature num\_missing pct\_missing  
## <fct> <int> <dbl>  
## 1 PoolQC 2909 0.997   
## 2 MiscFeature 2814 0.964   
## 3 Alley 2721 0.932   
## 4 Fence 2348 0.804   
## 5 SalePrice 1459 0.500   
## 6 FireplaceQu 1420 0.486   
## 7 LotFrontage 486 0.166   
## 8 GarageYrBlt 159 0.0545  
## 9 GarageFinish 159 0.0545  
## 10 GarageQual 159 0.0545  
## # ... with 72 more rows

plot\_missing(house\_comb)

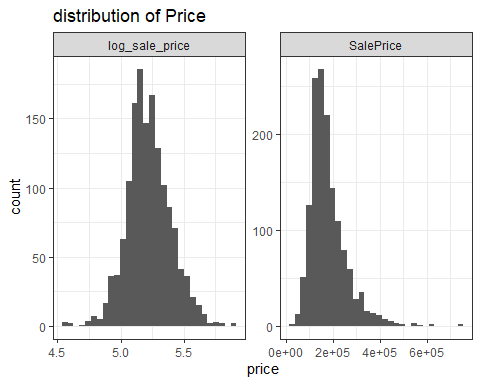


The missing values is result of absence of a particular feature in the property. We will code the missing values as with approproate categorical variable later on.

# Further Data Exploration

#check distribution of repsonse variabes  
  
house\_comb %>%   
 filter(set == "train") %>%  
 mutate(log\_sale\_price = log10(SalePrice)) %>%  
 select(SalePrice, log\_sale\_price) %>%  
 pivot\_longer(everything(), names\_to = "Sale", values\_to = "price") %>%  
 ggplot(aes(x = price)) +  
 geom\_histogram() + facet\_wrap(Sale~., scales = "free") +  
 labs(title = "distribution of Price") +  
 theme\_bw()

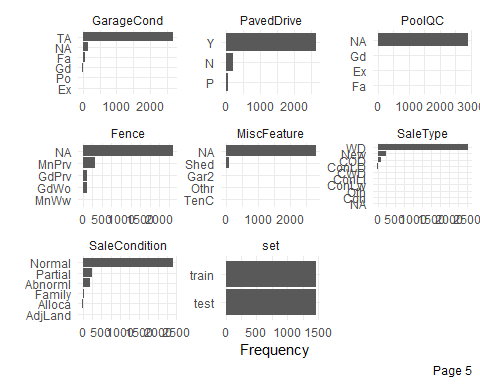
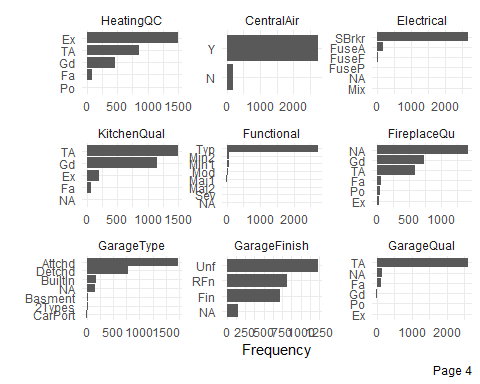
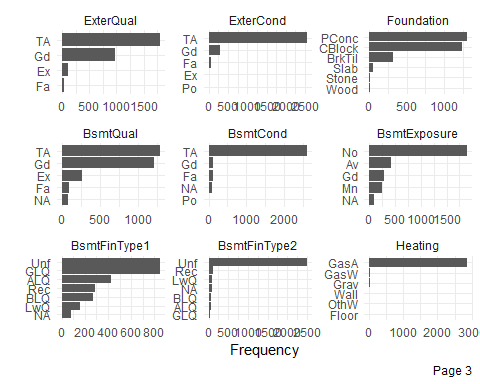
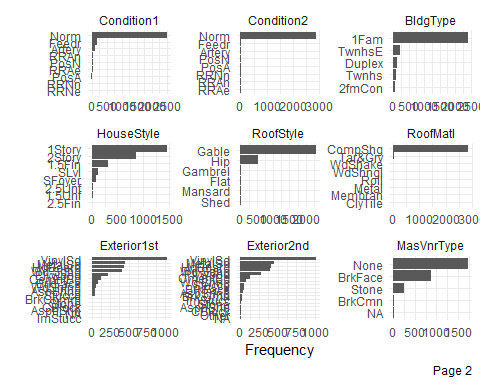
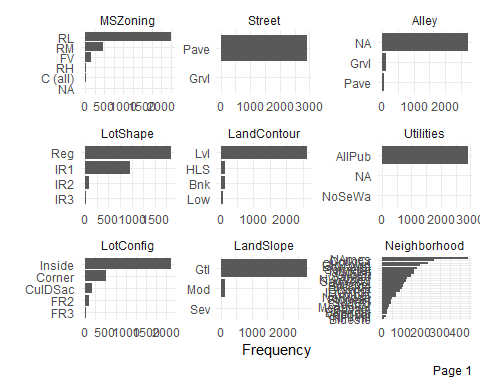
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



Sale Price distribtion is right skewed however log transformation is Normal and is a candidate for Linear Regression. Lets explore other features.

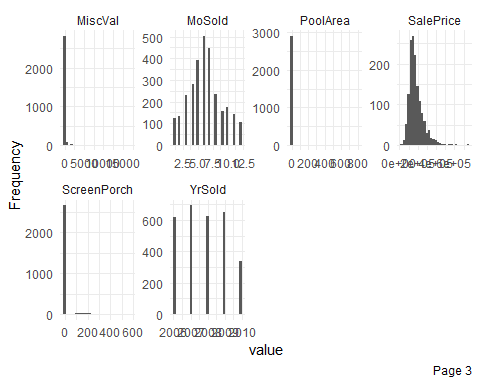
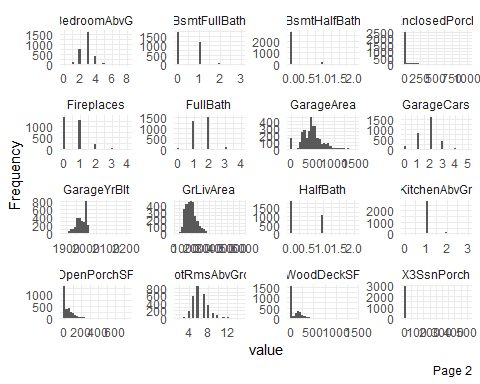
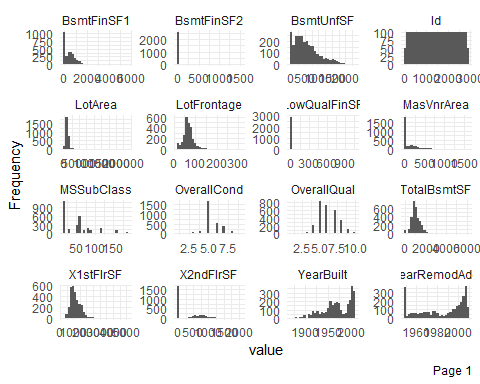
# Check distribution of categorical variables

plot\_bar(house\_comb, ggtheme = theme\_minimal())



# Check distribution of Numeric features

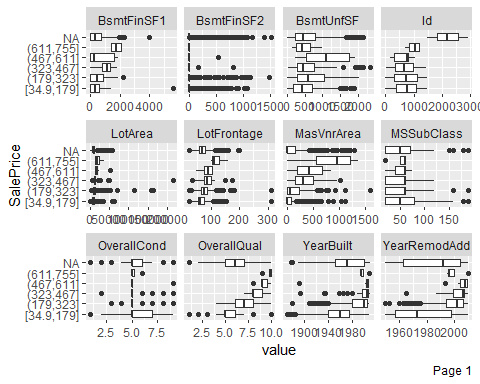
plot\_histogram(house\_comb, ggtheme = theme\_minimal())



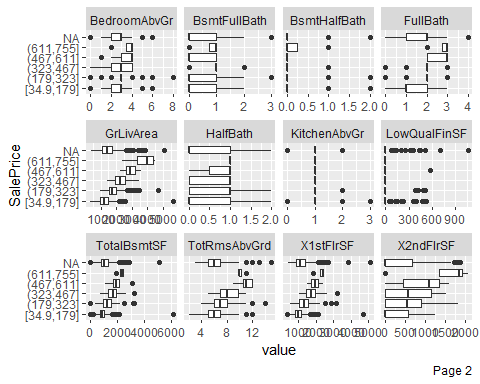
# Explore categorical variables against Sale price.

house\_comb %>% mutate(SalePrice = SalePrice/1000) %>%  
plot\_boxplot(by = "SalePrice")

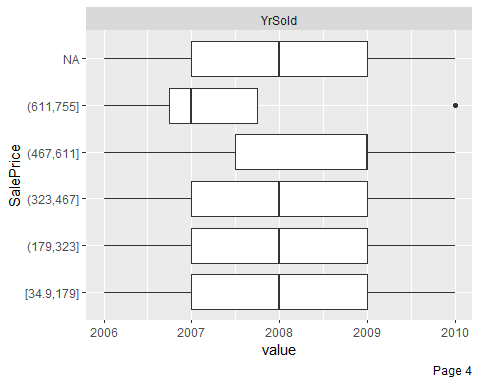
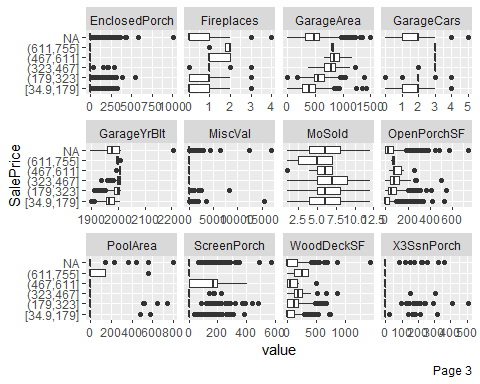
## Warning: Removed 512 rows containing non-finite values (stat\_boxplot).



## Warning: Removed 5 rows containing non-finite values (stat\_boxplot).



## Warning: Removed 161 rows containing non-finite values (stat\_boxplot).



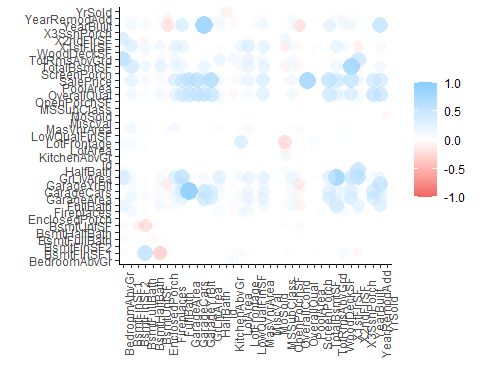
# Checking correlation among variables

house\_comb %>%  
 select\_if(is.numeric) %>%  
 corrr::correlate() %>%  
 rearrange() %>%  
 shave() %>% rplot + theme(axis.text.x = element\_text(angle = 90))

##   
## Correlation method: 'pearson'  
## Missing treated using: 'pairwise.complete.obs'

## Registered S3 method overwritten by 'seriation':  
## method from   
## reorder.hclust gclus

## Don't know how to automatically pick scale for object of type noquote. Defaulting to continuous.



house\_comb %>% get\_corr\_df\_simple(cut\_off = 0.6)

##   
## Correlation method: 'pearson'  
## Missing treated using: 'pairwise.complete.obs'

## # A tibble: 18 x 3  
## rowname features corr  
## <chr> <chr> <dbl>  
## 1 GarageCars GarageArea 0.890  
## 2 YearBuilt GarageYrBlt 0.835  
## 3 GrLivArea TotRmsAbvGrd 0.808  
## 4 TotalBsmtSF X1stFlrSF 0.802  
## 5 OverallQual SalePrice 0.791  
## 6 GrLivArea SalePrice 0.709  
## 7 BedroomAbvGr TotRmsAbvGrd 0.670  
## 8 X2ndFlrSF GrLivArea 0.655  
## 9 YearRemodAdd GarageYrBlt 0.652  
## 10 GarageCars SalePrice 0.640  
## 11 BsmtFinSF1 BsmtFullBath 0.639  
## 12 GrLivArea FullBath 0.630  
## 13 GarageArea SalePrice 0.623  
## 14 TotalBsmtSF SalePrice 0.614  
## 15 YearBuilt YearRemodAdd 0.612  
## 16 X2ndFlrSF HalfBath 0.611  
## 17 X1stFlrSF SalePrice 0.606  
## 18 OverallQual GarageCars 0.601

# Check for near zero variables

house\_comb[, nearZeroVar(house\_comb)] %>%  
 mutate\_if(is.character, as.factor) %>%  
 summary

## Street LandContour Utilities LandSlope Condition2 RoofMatl   
## Grvl: 12 Bnk: 117 AllPub:2916 Gtl:2778 Norm :2889 CompShg:2876   
## Pave:2907 HLS: 120 NoSeWa: 1 Mod: 125 Feedr : 13 Tar&Grv: 23   
## Low: 60 NA's : 2 Sev: 16 Artery : 5 WdShake: 9   
## Lvl:2622 PosA : 4 WdShngl: 7   
## PosN : 4 ClyTile: 1   
## RRNn : 2 Membran: 1   
## (Other): 2 (Other): 2   
## BsmtCond BsmtFinType2 BsmtFinSF2 Heating LowQualFinSF   
## Fa : 104 ALQ : 52 Min. : 0.00 Floor: 1 Min. : 0.000   
## Gd : 122 BLQ : 68 1st Qu.: 0.00 GasA :2874 1st Qu.: 0.000   
## Po : 5 GLQ : 34 Median : 0.00 GasW : 27 Median : 0.000   
## TA :2606 LwQ : 87 Mean : 49.58 Grav : 9 Mean : 4.694   
## NA's: 82 Rec : 105 3rd Qu.: 0.00 OthW : 2 3rd Qu.: 0.000   
## Unf :2493 Max. :1526.00 Wall : 6 Max. :1064.000   
## NA's: 80 NA's :1   
## KitchenAbvGr Functional GarageQual GarageCond OpenPorchSF   
## Min. :0.000 Typ :2717 Ex : 3 Ex : 3 Min. : 0.00   
## 1st Qu.:1.000 Min2 : 70 Fa : 124 Fa : 74 1st Qu.: 0.00   
## Median :1.000 Min1 : 65 Gd : 24 Gd : 15 Median : 26.00   
## Mean :1.045 Mod : 35 Po : 5 Po : 14 Mean : 47.49   
## 3rd Qu.:1.000 Maj1 : 19 TA :2604 TA :2654 3rd Qu.: 70.00   
## Max. :3.000 (Other): 11 NA's: 159 NA's: 159 Max. :742.00   
## NA's : 2   
## EnclosedPorch X3SsnPorch ScreenPorch PoolArea   
## Min. : 0.0 Min. : 0.000 Min. : 0.00 Min. : 0.000   
## 1st Qu.: 0.0 1st Qu.: 0.000 1st Qu.: 0.00 1st Qu.: 0.000   
## Median : 0.0 Median : 0.000 Median : 0.00 Median : 0.000   
## Mean : 23.1 Mean : 2.602 Mean : 16.06 Mean : 2.252   
## 3rd Qu.: 0.0 3rd Qu.: 0.000 3rd Qu.: 0.00 3rd Qu.: 0.000   
## Max. :1012.0 Max. :508.000 Max. :576.00 Max. :800.000   
##   
## MiscVal   
## Min. : 0.00   
## 1st Qu.: 0.00   
## Median : 0.00   
## Mean : 50.83   
## 3rd Qu.: 0.00   
## Max. :17000.00   
##

# Curious to know if street type makes a difference

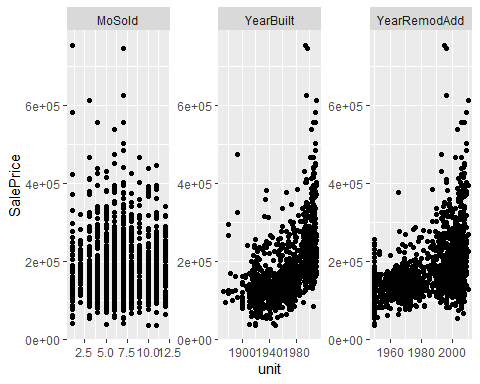
house\_comb %>% filter(set == "train") %>%  
 group\_by(Street) %>%  
 summarise(sumry = list(summary(SalePrice))) %>%   
 mutate(sumry = map(sumry, ~ data.frame(t(.)))) %>%  
 unnest(sumry) %>% pivot\_wider(names\_from = Street, values\_from = Freq)

## # A tibble: 6 x 4  
## Var1 Var2 Grvl Pave  
## <fct> <fct> <dbl> <dbl>  
## 1 A Min. 55993 34900   
## 2 A 1st Qu. 88250 130000   
## 3 A Median 114250 163000   
## 4 A Mean 130190. 181131.  
## 5 A 3rd Qu. 169650 214000   
## 6 A Max. 228950 755000

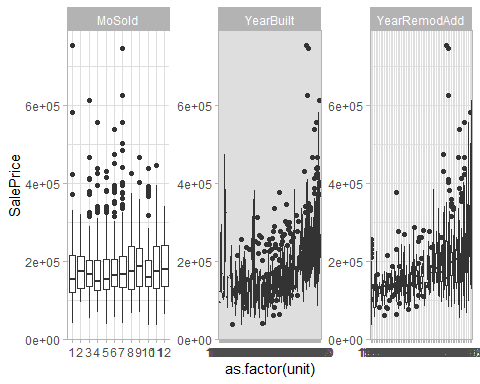
In general we do see House in pavement have higher Sale price than the Gravel.

# Checking affect of time over Sale Price

house\_comb %>%   
 filter(set == "train") %>%  
 select\_at(vars(starts\_with("Year"), starts\_with("MoSold"), SalePrice)) %>%  
 pivot\_longer(-SalePrice, names\_to = "time\_metric", values\_to = "unit") %>%  
 ggplot(aes(y = SalePrice, x = unit )) + geom\_point() + facet\_wrap(.~time\_metric, scales = "free")



house\_comb %>%   
 filter(set == "train") %>%  
 select\_at(vars(starts\_with("Year"), starts\_with("MoSold"), SalePrice)) %>%  
 pivot\_longer(-SalePrice, names\_to = "time\_metric", values\_to = "unit") %>%  
 ggplot(aes(y = SalePrice, x = as.factor(unit) )) + geom\_boxplot() + facet\_wrap(.~time\_metric, scales = "free") + theme\_light()



There is no particular effect of month sold on Sale price. However, In general we dprices does increase Year on Year which makes common sense.

# Based on our explorations above lets will do some preliminary data cleaning on the train data set

The code below will fix few missing values which are actually features not present in a property. These features are related to . Alley . Bsmt . Garage . FireplaceQu . PoolQC . Fence . MiscFeatures . MasVnr

We also create a categorical bucketing variable out of YearSold and Yr Modelled. We create new variabe ‘log\_sale\_price’ that is log of Sale Price. This new variable will be used as response variables.

Last we remove Id, Sale Price and set variable that will not contribute to the model.

#pp3  
house <- train %>%  
   
 #converting sales to log scale  
 mutate(log\_sale\_price = log(SalePrice)) %>%  
   
   
 #fill in missing values NA for factor variables as per data description  
   
 #Alley  
 mutate(Alley = if\_else(is.na(Alley), "No Alley", Alley)) %>%  
   
 #Bsmt  
 mutate(  
 BsmtCond = if\_else(is.na(BsmtCond), "No Bsmnt", BsmtCond),  
 BsmtExposure = if\_else(is.na(BsmtExposure), "No Bsmnt", BsmtExposure),  
 BsmtQual = if\_else(is.na(BsmtQual), "No Bsmnt", BsmtQual),  
 BsmtFinType1 = if\_else(is.na(BsmtFinType1), "No Bsmnt", BsmtFinType1),  
 BsmtFinType2 = if\_else(is.na(BsmtFinType2), "No Bsmnt", BsmtFinType2),  
 BsmtFinSF1 = if\_else(is.na(BsmtFinSF1), 0, BsmtFinSF1),  
 BsmtFinSF2 = if\_else(is.na(BsmtFinSF2), 0, BsmtFinSF2),  
 BsmtUnfSF = if\_else(is.na(BsmtUnfSF), 0, BsmtUnfSF),  
 TotalBsmtSF = if\_else(is.na(TotalBsmtSF), 0, TotalBsmtSF),  
 BsmtFullBath = if\_else(is.na(BsmtFullBath), 0, BsmtFullBath),  
 BsmtHalfBath = if\_else(is.na(BsmtHalfBath), 0, BsmtHalfBath)  
   
 ) %>%  
   
 #Garage  
 #GarageType, GarageType, GarageFinish, GarageQual, GarageCond, 'GarageYrBlt', 'GarageArea', 'GarageCars'  
 mutate(  
 GarageType = if\_else(is.na(GarageType), "No Garage", GarageType),  
 GarageFinish = if\_else(is.na(GarageFinish), "No Garage", GarageFinish),  
 GarageQual = if\_else(is.na(GarageQual), "No Garage", GarageQual),  
 GarageCond = if\_else(is.na(GarageCond), "No Garage", GarageCond),  
 GarageYrBlt = if\_else(is.na(GarageYrBlt), 0, GarageYrBlt),  
 GarageArea = if\_else(is.na(GarageArea), 0, GarageArea),  
 GarageCars = if\_else(is.na(GarageCond), 0, GarageCars)  
 ) %>%  
   
 mutate(MasVnrType = if\_else(is.na(MasVnrType), "No MasVnrType", MasVnrType)) %>%  
 mutate(MasVnrArea = if\_else(is.na(MasVnrArea), 0, MasVnrArea)) %>%  
   
   
 #FireplaceQu, PoolQC,Fence, MiscFeature  
 mutate(  
 FireplaceQu = if\_else(is.na(FireplaceQu), "No Fireplc", FireplaceQu),  
 PoolQC = if\_else(is.na(PoolQC), "No Pool", PoolQC),  
 Fence = if\_else(is.na(Fence), "No Fence", Fence),  
 MiscFeature = if\_else(is.na(MiscFeature), "None", MiscFeature)  
 ) %>%  
   
 mutate(YrBuilt\_cat = cut2(YearBuilt, cuts = seq(min(YearBuilt), max(YearBuilt), 5))) %>%  
   
 mutate(YearRemodAdd = if\_else(is.na(YearRemodAdd), YearBuilt, YearRemodAdd)) %>%  
   
   
 #remove non contributing features and IDs  
 select(-Id,-SalePrice, -set)

# Creating a Training and Validation set using rsample

set.seed(1234)  
house\_split <- initial\_split(house)  
  
train\_split <- training(house\_split)

# Create recepies with series of preprocessing step on training set

house\_recipe <- train\_split %>%  
 recipe(log\_sale\_price ~ .) %>%  
 step\_string2factor(all\_nominal(), -all\_outcomes()) %>%  
 step\_num2factor(MoSold, levels = as.character(unique(train\_split$MoSold))) %>%  
 step\_num2factor(YrSold, levels = as.character(unique(train\_split$YrSold))) %>%  
 step\_num2factor(MSSubClass, levels = as.character(unique(train\_split$MSSubClass))) %>%  
 step\_num2factor(OverallCond, levels = as.character(unique(train\_split$OverallCond))) %>%  
 step\_dummy(all\_nominal(), -all\_outcomes()) %>%  
 step\_center(all\_predictors(), -all\_outcomes()) %>%  
 step\_scale(all\_predictors(), -all\_outcomes()) %>%  
 step\_corr(all\_predictors()) %>%  
 step\_nzv(all\_predictors()) %>%  
 step\_knnimpute(all\_predictors(), -all\_outcomes())   
 # step\_corr(all\_predictors()) %>%  
 # step\_nzv(all\_predictors()) %>%  
   
doParallel::registerDoParallel()  
house\_prep <- prep(house\_recipe)

## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA

## Warning in cor(x, use = use, method = method): the standard deviation is zero

## Warning: The correlation matrix has missing values. 18 columns were excluded  
## from the filter.

## Warning: The correlation matrix has sporadic missing values. Some columns were  
## excluded from the filter.

house\_recipe\_treebased <- train\_split %>%  
 recipe(log\_sale\_price ~ .) %>%  
 step\_string2factor(all\_nominal(), -all\_outcomes()) %>%  
 step\_num2factor(MoSold, levels = as.character(unique(train\_split$MoSold))) %>%  
 step\_num2factor(YrSold, levels = as.character(unique(train\_split$YrSold))) %>%  
 step\_num2factor(OverallCond, levels = as.character(unique(train\_split$OverallCond))) %>%  
 step\_num2factor(MSSubClass, levels = as.character(unique(train\_split$MSSubClass))) %>%  
 step\_dummy(all\_nominal(), -all\_outcomes()) %>%  
 # step\_center(all\_numeric(), -all\_outcomes()) %>%  
 # step\_scale(all\_numeric(), -all\_outcomes()) %>%  
 step\_corr(all\_numeric()) %>%  
 step\_nzv(all\_numeric()) %>%  
 step\_knnimpute(all\_predictors(), -all\_outcomes())  
  
doParallel::registerDoParallel()   
house\_treebased\_prep <- prep(house\_recipe\_treebased)

## Warning: There are new levels in a factor: NA

## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA

## Warning in cor(x, use = use, method = method): the standard deviation is zero

## Warning: The correlation matrix has missing values. 18 columns were excluded  
## from the filter.

## Warning: The correlation matrix has sporadic missing values. Some columns were  
## excluded from the filter.

# Applying preprocessing on training and validation(testing) set

house\_train <- juice(house\_prep)  
house\_test <- bake(house\_prep, testing(house\_split))

## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA

#splits for tree based algo  
house\_train\_treebased <- juice(house\_treebased\_prep)  
house\_test\_treebased <- bake(house\_treebased\_prep, testing(house\_split))

## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA  
  
## Warning: There are new levels in a factor: NA

## Train models

# linear regression

#linear model  
lm\_model <- #recipe(log\_sale\_price ~ . , data = house\_train) %>%  
 linear\_reg() %>%  
 set\_mode("regression") %>%  
 set\_engine("lm") %>%  
 fit(log\_sale\_price ~ . , data = house\_train)

# random forest

#create recepie on the preped house train data  
rf\_rec <-  
 recipe(log\_sale\_price ~. , data = house\_train\_treebased)  
  
#give model spec  
rf\_mod <-  
 rand\_forest(mtry = tune(), min\_n = tune()) %>%  
 set\_engine("ranger") %>%  
 set\_mode("regression")  
  
#create Search grid  
rf\_grid <-  
 grid\_regular(mtry(range = c(15,40)), min\_n(range= c(10, 2)), levels = 5)  
  
#create samples for cross validation  
folds <- vfold\_cv(house\_train\_treebased, v = 10)  
  
  
doParallel::registerDoParallel()  
  
#create models with grid search  
rf\_res <-  
 tune\_grid(model = rf\_mod, rf\_rec,resamples = folds , grid = rf\_grid)

## Warning: `tune\_grid.recipe()` is deprecated as of lifecycle 0.1.0.  
## The first argument to `tune\_grid()` should be either a model or a workflow. In the future, you can use:  
## tune\_grid(rf\_mod, rf\_rec, resamples = folds, grid = rf\_grid)  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_warnings()` to see where this warning was generated.

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

final\_mtry <- select\_best(rf\_res, "rmse", maximize = FALSE)$mtry

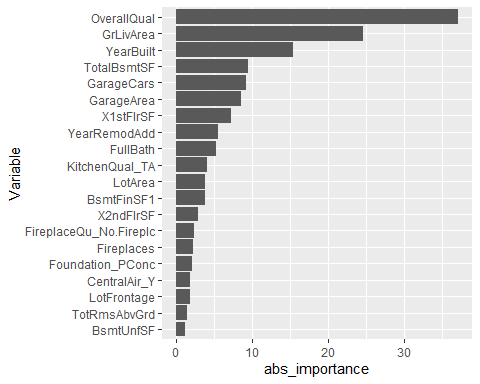
## Warning: The `maximize` argument is no longer needed. This value was ignored.

final\_min\_node <- select\_best(rf\_res, "rmse", maximize = FALSE)$min\_n

## Warning: The `maximize` argument is no longer needed. This value was ignored.

#random forest  
rf\_model <- rand\_forest(mtry = final\_mtry, min\_n = final\_min\_node) %>%  
 set\_mode("regression") %>%  
 set\_engine("ranger",importance = 'impurity') %>%  
 fit(log\_sale\_price ~ . , data = house\_train\_treebased)  
  
#Visualise Feature Importance  
vi(rf\_model$fit) %>% mutate(Variable = fct\_reorder(Variable, abs(Importance)), abs\_importance = abs(Importance)) %>% arrange(desc(abs\_importance)) %>% top\_n(20) %>%  
ggplot(aes(x = Variable, y = abs\_importance)) + geom\_col() + coord\_flip()

## Selecting by abs\_importance



# xgboost

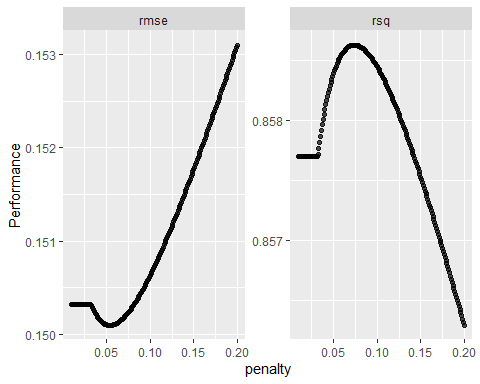
doParallel::registerDoParallel()  
#xgboost  
xg\_model <- boost\_tree() %>%  
 set\_mode("regression") %>%  
 set\_engine("xgboost") %>%  
 fit(log\_sale\_price ~ . , data = house\_train\_treebased)

# linear model ridge

ridge\_rec <-  
 recipe(log\_sale\_price ~. , data = house\_train)  
  
ridge\_mod <-  
 linear\_reg(penalty = tune(), mixture = tune()) %>%  
 set\_engine("glmnet")  
   
  
ridge\_grid <- expand.grid(penalty = seq(from = 0.01, to = 0.2, by=0.001), mixture = 0)  
  
folds <- vfold\_cv(house\_train, v = 10)  
  
doParallel::registerDoParallel()  
  
ridge\_res <-  
 tune\_grid(ridge\_rec, model = ridge\_mod, resamples = folds , grid = ridge\_grid)

## Warning: `tune\_grid.recipe()` is deprecated as of lifecycle 0.1.0.  
## The first argument to `tune\_grid()` should be either a model or a workflow. In the future, you can use:  
## tune\_grid(ridge\_mod, ridge\_rec, resamples = folds, grid = ridge\_grid)  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_warnings()` to see where this warning was generated.

autoplot(ridge\_res)

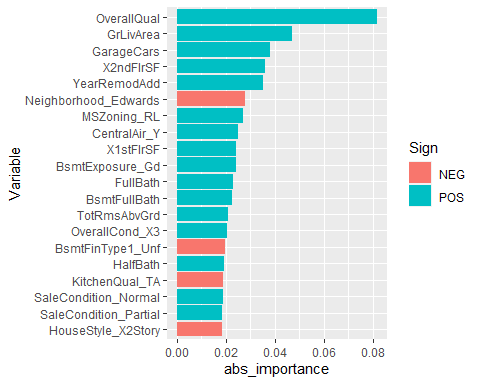


final\_penalty <- select\_best(ridge\_res, "rmse", maximize = FALSE)$penalty

## Warning: The `maximize` argument is no longer needed. This value was ignored.

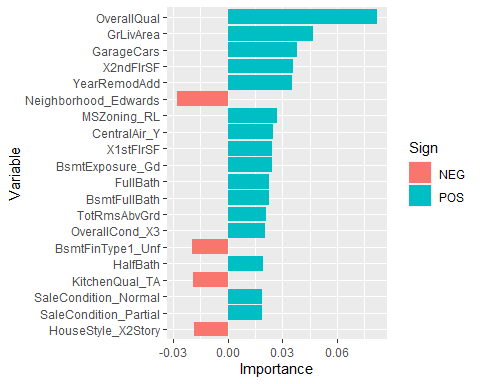
#apply best tuning parameter linear model ridge  
lm\_ridge\_model <- linear\_reg(penalty = final\_penalty , mixture = 0) %>%  
 set\_mode("regression") %>%  
 set\_engine("glmnet") %>%  
 fit(log\_sale\_price ~. , data = house\_train)  
  
#Visualise Feature Importance  
vi(lm\_ridge\_model$fit) %>% mutate(Variable = fct\_reorder(Variable, abs(Importance)), abs\_importance = abs(Importance)) %>% arrange(desc(abs\_importance)) %>% top\_n(20) %>%  
ggplot(aes(x = Variable, y = abs\_importance, fill = Sign)) + geom\_col() + coord\_flip()

## Selecting by abs\_importance

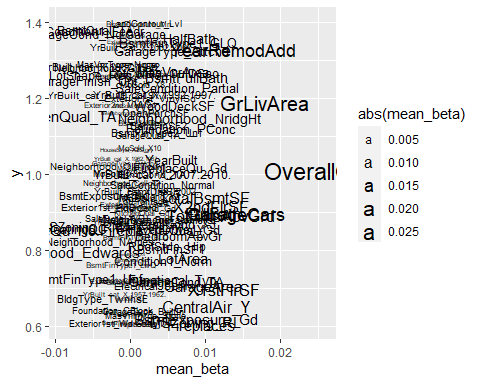


vi(lm\_ridge\_model$fit) %>% mutate(Variable = fct\_reorder(Variable, abs(Importance)), abs\_importance = abs(Importance)) %>% arrange(desc(abs\_importance)) %>% top\_n(20) %>% ggplot(aes(x = Variable, y = Importance, fill = Sign)) + geom\_col() + coord\_flip()

## Selecting by abs\_importance



#Visulaise beta strength  
as.data.frame(as.matrix(lm\_ridge\_model$fit$beta)) %>% rownames\_to\_column() %>% pivot\_longer(-rowname) %>% group\_by(rowname) %>% summarise(mean\_beta = mean(value), se\_beta = sd(value)) %>% ungroup %>% ggplot(aes(y = 1, x = mean\_beta, , size = abs(mean\_beta)), alpha = 0.2) + geom\_text(aes(label = rowname), position = position\_jitter())



# linear model lasso

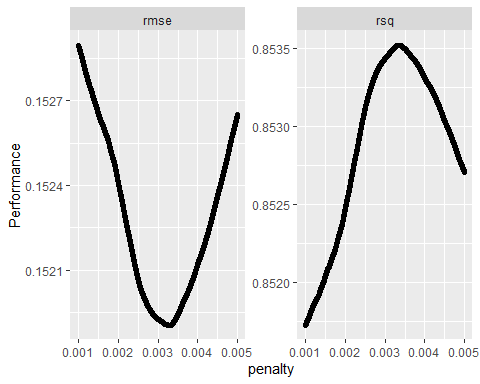
lasso\_rec <-  
 recipe(log\_sale\_price ~. , data = house\_train)  
  
lasso\_mod <-  
 linear\_reg(penalty = tune(), mixture = tune()) %>%  
 set\_engine("glmnet")  
   
lasso\_grid <- expand.grid(penalty = seq(from = 0.001, to = 0.005, by = 0.000001), mixture = 1)  
  
folds\_lasso <- vfold\_cv(house\_train, v = 10)  
  
lasso\_res <-  
 tune\_grid(lasso\_rec, model = lasso\_mod, resamples = folds\_lasso , grid = lasso\_grid)

## Warning: `tune\_grid.recipe()` is deprecated as of lifecycle 0.1.0.  
## The first argument to `tune\_grid()` should be either a model or a workflow. In the future, you can use:  
## tune\_grid(lasso\_mod, lasso\_rec, resamples = folds\_lasso, grid = lasso\_grid)  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_warnings()` to see where this warning was generated.

lambda\_final <-  
 select\_best(lasso\_res, metric = "rmse", maximize = FALSE) %>% select(penalty) %>% unlist

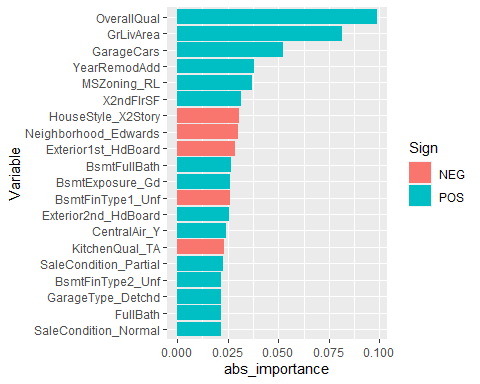
## Warning: The `maximize` argument is no longer needed. This value was ignored.

autoplot(lasso\_res)



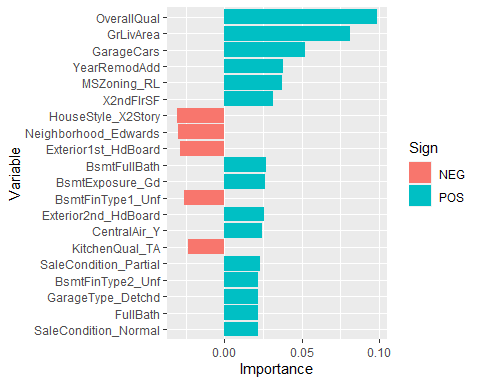
#apply best tuning parameter linear model ridge  
lm\_lasso\_model <- linear\_reg(penalty = lambda\_final, mixture = 1) %>%  
 set\_mode("regression") %>%  
 set\_engine("glmnet") %>%  
 fit(log\_sale\_price ~. , data = house\_train)  
  
#Visualise Feature Importance  
vi(lm\_lasso\_model$fit) %>% mutate(Variable = fct\_reorder(Variable, abs(Importance)),  
 abs\_importance = abs(Importance)) %>% arrange(desc(abs\_importance)) %>% top\_n(20) %>%  
 ggplot(aes(x = Variable, y = abs\_importance, fill = Sign)) + geom\_col() + coord\_flip()

## Selecting by abs\_importance

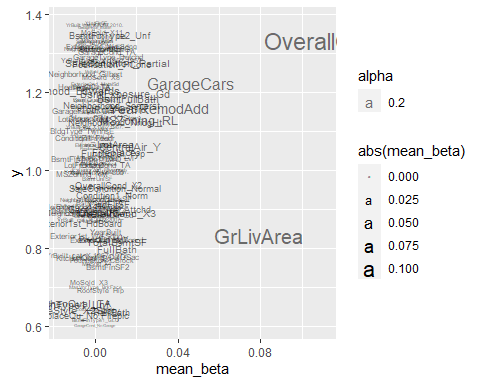


vi(lm\_lasso\_model$fit) %>% mutate(Variable = fct\_reorder(Variable, abs(Importance)),  
 abs\_importance = abs(Importance)) %>% arrange(desc(abs\_importance)) %>% top\_n(20) %>% ggplot(aes(x = Variable, y = Importance, fill = Sign)) + geom\_col() + coord\_flip()

## Selecting by abs\_importance



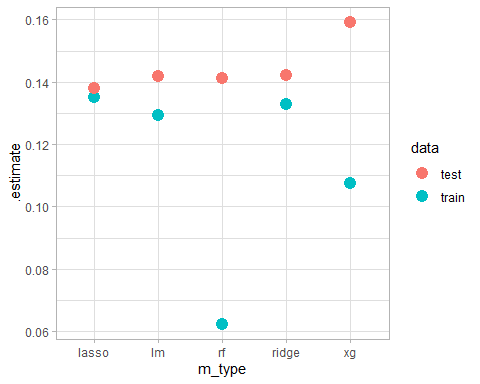
#Visulaise beta strength  
as.data.frame(as.matrix(lm\_lasso\_model$fit$beta)) %>% rownames\_to\_column() %>% pivot\_longer(-rowname) %>% group\_by(rowname) %>% summarise(mean\_beta = mean(value), se\_beta = sd(value)) %>% ungroup %>% ggplot(aes(  
 y = 1,  
 x = mean\_beta,  
 alpha = 0.2,  
 size = abs(mean\_beta)  
)) + geom\_text(aes(label = rowname), position = position\_jitter())



# evaluate and consolidate model metrics

(  
 map2\_df(  
 list(  
 lm = list(lm = lm\_model,name = "lm"),  
 rf = list(rf = rf\_model, name = "rf"),  
 xg = list(xg = xg\_model, name ="xg"),  
 ridge = list(lm\_ridge = lm\_ridge\_model, name = "ridge"),  
 lasso = list(lm\_lasso = lm\_lasso\_model, name = "lasso")  
 ),  
 list(  
 house\_train,  
 house\_train\_treebased,  
 house\_train\_treebased,  
 house\_train,  
 house\_train  
 ),  
 get\_model\_metrics  
 ) %>% mutate(data = "train")  
) %>%  
   
 bind\_rows(  
   
 map2\_df(  
 list(  
 lm = list(lm = lm\_model,name = "lm"),  
 rf = list(rf = rf\_model, name = "rf"),  
 xg = list(xg = xg\_model, name ="xg"),  
 ridge = list(lm\_ridge = lm\_ridge\_model, name = "ridge"),  
 lasso = list(lm\_lasso = lm\_lasso\_model, name = "lasso")  
 ),  
 list(house\_test, house\_test\_treebased, house\_test\_treebased, house\_test, house\_test),  
 get\_model\_metrics  
 ) %>% mutate(data = "test")  
 ) %>%  
 # pivot\_wider(names\_from = data, values\_from = .estimate) %>%  
 # unnest %>%  
   
 #plot the metrics  
 ggplot(aes(y = .estimate, x = m\_type, color = data)) + geom\_point(size = 4) + theme\_light()

## Warning in predict.lm(object = object$fit, newdata = new\_data, type =  
## "response"): prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(object = object$fit, newdata = new\_data, type =  
## "response"): prediction from a rank-deficient fit may be misleading



The lasso model gives the best test RMSE value.