Predicting House Prices

Kaggle Competition

This Kaggle competition is about predicting house prices based on a set of around 80 predictor variables. Please read the brief description of the project and get familiar with the various predictors. We will have to do some initial cleaning to successfully work with these data. Overall, we (in teams) will use the provided training dataset to built a multiple linear regression model for predicting house prices. Once we have settled on a final model, we will use it with the predictors available in the testing dataset to predict house prices. The goal of the competition mentions that our predictions $\hat{y}_i$ for the houses in the testing data are compared to the (withheld) true selling prices y_i^{test} via $\sum_i (\log \hat{y}_i - \log y_i^{\text{test}})^2$. Because selling prices are typically right-skewed, I think as a first step we will log-transform the selling prices of the houses in the training data to obtain a more bell-shaped distribution. However, although we will built a model for the log-prices, we will still have to submit the price of a house (and not the log-price) to Kaggle, together with the ID of the house.

Loading and inspecting the train and test datasets

```r
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.0.5
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.2     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
```

```
## Warning: package 'ggplot2' was built under R version 4.0.5
```

```
## Warning: package 'tibble' was built under R version 4.0.5
```

```
## Warning: package 'tidyr' was built under R version 4.0.5
```

```
## Warning: package 'readr' was built under R version 4.0.5
```

```
## Warning: package 'dplyr' was built under R version 4.0.5
```

```
## Warning: package 'forcats' was built under R version 4.0.5
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(tidyr)
## Load Training Data
path_traindata <- 'https://raw.githubusercontent.com/bklingen/Price-Prediction/main/train.csv'
train <- read_csv(path_traindata)
```

```
##
## -- 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
## )
## i Use `spec()` for the full column specifications.
```

```r
dim(train)
```

```
## [1] 1460   81
```

```r
## Load Testing Data
path_testdata <- 'https://raw.githubusercontent.com/bklingen/Price-Prediction/main/test.csv'
test <- read_csv(path_testdata)
```

```
##
## -- 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
```

```
## )
## i Use `spec()` for the full column specifications.
```

```
dim(test)
```

```
## [1] 1459   80
```

This makes sense: We have one less column in test data because of the missing house prices.

But, are the column names the same? Let's find the "difference" between two sets: All the column names that are in the test data but not in the train data:

```
setdiff(colnames(test), colnames(train))
```

```
## character(0)
```

OK, good, and now the other way around:

```
setdiff(colnames(train), colnames(test))
```

```
## [1] "SalePrice"
```

OK, great. So no surprises there. All predictors that exist in the train data set also appear in the test dataset.

Let's see how many quantitative and how many categorical predictors we have in the training dataset, at least at face value:

```
train_quantPredictors = train %>% select(where(is.numeric)) %>% select(-SalePrice)
train_catPredictors = train %>% select(where(is.character))
dim(train_quantPredictors)
```

```
## [1] 1460   37
```

```
dim(train_catPredictors)
```

```
## [1] 1460   43
```

Let's quickly do the same split for the test data:

```
test_quantPredictors = test %>% select(where(is.numeric))
test_catPredictors = test %>% select(where(is.character))
```

Let's transform the categorical predictors into factors, which should make it easier to combine categories, create a category like "other", etc.

```
train_catPredictors = train_catPredictors %>% transmute_all(as.factor)
test_catPredictors = test_catPredictors %>% transmute_all(as.factor)
```

First, let's see the category names and frequency for each variable:

```
for(i in 1:ncol(train_catPredictors)) {
  print(colnames(train_catPredictors)[i])
  print("----")
  print(as.data.frame(fct_count(unlist(train_catPredictors[,i]))))
  print("--------------")
}
```

```
## [1] "MSZoning"
## [1] "----"
##          f    n
## 1 C (all)   10
## 2      FV   65
## 3      RH   16
```

```
## 4      RL 1151
## 5      RM  218
## [1] "--------------"
## [1] "Street"
## [1] "----"
##      f    n
## 1 Grvl    6
## 2 Pave 1454
## [1] "--------------"
## [1] "Alley"
## [1] "----"
##      f    n
## 1 Grvl   50
## 2 Pave   41
## 3 <NA> 1369
## [1] "--------------"
## [1] "LotShape"
## [1] "----"
##     f   n
## 1 IR1 484
## 2 IR2  41
## 3 IR3  10
## 4 Reg 925
## [1] "--------------"
## [1] "LandContour"
## [1] "----"
##     f    n
## 1 Bnk   63
## 2 HLS   50
## 3 Low   36
## 4 Lvl 1311
## [1] "--------------"
## [1] "Utilities"
## [1] "----"
##        f    n
## 1 AllPub 1459
## 2 NoSeWa    1
## [1] "--------------"
## [1] "LotConfig"
## [1] "----"
##         f    n
## 1  Corner  263
## 2 CulDSac   94
## 3     FR2   47
## 4     FR3    4
## 5  Inside 1052
## [1] "--------------"
## [1] "LandSlope"
## [1] "----"
##     f    n
## 1 Gtl 1382
## 2 Mod   65
## 3 Sev   13
## [1] "--------------"
```

```
## [1] "Neighborhood"
## [1] "----"
##          f   n
## 1  Blmngtn  17
## 2  Blueste   2
## 3   BrDale  16
## 4  BrkSide  58
## 5  ClearCr  28
## 6  CollgCr 150
## 7  Crawfor  51
## 8  Edwards 100
## 9  Gilbert  79
## 10  IDOTRR  37
## 11 MeadowV  17
## 12 Mitchel  49
## 13   NAmes 225
## 14 NoRidge  41
## 15 NPkVill   9
## 16 NridgHt  77
## 17  NWAmes  73
## 18 OldTown 113
## 19  Sawyer  74
## 20 SawyerW  59
## 21 Somerst  86
## 22 StoneBr  25
## 23   SWISU  25
## 24  Timber  38
## 25 Veenker  11
## [1] "--------------"
## [1] "Condition1"
## [1] "----"
##        f    n
## 1 Artery   48
## 2  Feedr   81
## 3   Norm 1260
## 4   PosA    8
## 5   PosN   19
## 6   RRAe   11
## 7   RRAn   26
## 8   RRNe    2
## 9   RRNn    5
## [1] "--------------"
## [1] "Condition2"
## [1] "----"
##        f    n
## 1 Artery    2
## 2  Feedr    6
## 3   Norm 1445
## 4   PosA    1
## 5   PosN    2
## 6   RRAe    1
## 7   RRAn    1
## 8   RRNn    2
## [1] "--------------"
```

```
## [1] "BldgType"
## [1] "----"
##        f    n
## 1   1Fam 1220
## 2 2fmCon   31
## 3 Duplex   52
## 4  Twnhs   43
## 5 TwnhsE  114
## [1] "--------------"
## [1] "HouseStyle"
## [1] "----"
##        f   n
## 1 1.5Fin 154
## 2 1.5Unf  14
## 3 1Story 726
## 4 2.5Fin   8
## 5 2.5Unf  11
## 6 2Story 445
## 7 SFoyer  37
## 8   SLvl  65
## [1] "--------------"
## [1] "RoofStyle"
## [1] "----"
##         f    n
## 1    Flat   13
## 2   Gable 1141
## 3 Gambrel   11
## 4     Hip  286
## 5 Mansard    7
## 6    Shed    2
## [1] "--------------"
## [1] "RoofMatl"
## [1] "----"
##         f    n
## 1 ClyTile    1
## 2 CompShg 1434
## 3 Membran    1
## 4   Metal    1
## 5    Roll    1
## 6 Tar&Grv   11
## 7 WdShake    5
## 8 WdShngl    6
## [1] "--------------"
## [1] "Exterior1st"
## [1] "----"
##          f   n
## 1  AsbShng  20
## 2  AsphShn   1
## 3  BrkComm   2
## 4  BrkFace  50
## 5   CBlock   1
## 6  CemntBd  61
## 7  HdBoard 222
## 8  ImStucc   1
```

```
## 9  MetalSd 220
## 10 Plywood 108
## 11   Stone   2
## 12  Stucco  25
## 13 VinylSd 515
## 14 Wd Sdng 206
## 15 WdShing  26
## [1] "--------------"
## [1] "Exterior2nd"
## [1] "----"
##          f   n
## 1  AsbShng  20
## 2  AsphShn   3
## 3  Brk Cmn   7
## 4  BrkFace  25
## 5   CBlock   1
## 6  CmentBd  60
## 7  HdBoard 207
## 8  ImStucc  10
## 9  MetalSd 214
## 10   Other   1
## 11 Plywood 142
## 12   Stone   5
## 13  Stucco  26
## 14 VinylSd 504
## 15 Wd Sdng 197
## 16 Wd Shng  38
## [1] "--------------"
## [1] "MasVnrType"
## [1] "----"
##         f   n
## 1  BrkCmn  15
## 2 BrkFace 445
## 3    None 864
## 4   Stone 128
## 5    <NA>   8
## [1] "--------------"
## [1] "ExterQual"
## [1] "----"
##    f   n
## 1 Ex  52
## 2 Fa  14
## 3 Gd 488
## 4 TA 906
## [1] "--------------"
## [1] "ExterCond"
## [1] "----"
##    f    n
## 1 Ex    3
## 2 Fa   28
## 3 Gd  146
## 4 Po    1
## 5 TA 1282
## [1] "--------------"
```

```
## [1] "Foundation"
## [1] "----"
##        f   n
## 1 BrkTil 146
## 2 CBlock 634
## 3  PConc 647
## 4   Slab  24
## 5  Stone   6
## 6   Wood   3
## [1] "--------------"
## [1] "BsmtQual"
## [1] "----"
##      f   n
## 1   Ex 121
## 2   Fa  35
## 3   Gd 618
## 4   TA 649
## 5 <NA>  37
## [1] "--------------"
## [1] "BsmtCond"
## [1] "----"
##      f    n
## 1   Fa   45
## 2   Gd   65
## 3   Po    2
## 4   TA 1311
## 5 <NA>   37
## [1] "--------------"
## [1] "BsmtExposure"
## [1] "----"
##      f   n
## 1   Av 221
## 2   Gd 134
## 3   Mn 114
## 4   No 953
## 5 <NA>  38
## [1] "--------------"
## [1] "BsmtFinType1"
## [1] "----"
##      f   n
## 1  ALQ 220
## 2  BLQ 148
## 3  GLQ 418
## 4  LwQ  74
## 5  Rec 133
## 6  Unf 430
## 7 <NA>  37
## [1] "--------------"
## [1] "BsmtFinType2"
## [1] "----"
##      f    n
## 1  ALQ   19
## 2  BLQ   33
## 3  GLQ   14
```

```
## 4  LwQ   46
## 5  Rec   54
## 6  Unf 1256
## 7 <NA>   38
## [1] "--------------"
## [1] "Heating"
## [1] "----"
##       f    n
## 1 Floor    1
## 2  GasA 1428
## 3  GasW   18
## 4  Grav    7
## 5  OthW    2
## 6  Wall    4
## [1] "--------------"
## [1] "HeatingQC"
## [1] "----"
##    f   n
## 1 Ex 741
## 2 Fa  49
## 3 Gd 241
## 4 Po   1
## 5 TA 428
## [1] "--------------"
## [1] "CentralAir"
## [1] "----"
##   f    n
## 1 N   95
## 2 Y 1365
## [1] "--------------"
## [1] "Electrical"
## [1] "----"
##       f    n
## 1 FuseA   94
## 2 FuseF   27
## 3 FuseP    3
## 4   Mix    1
## 5 SBrkr 1334
## 6  <NA>    1
## [1] "--------------"
## [1] "KitchenQual"
## [1] "----"
##    f   n
## 1 Ex 100
## 2 Fa  39
## 3 Gd 586
## 4 TA 735
## [1] "--------------"
## [1] "Functional"
## [1] "----"
##      f    n
## 1 Maj1   14
## 2 Maj2    5
## 3 Min1   31
```

```
## 4 Min2   34
## 5  Mod   15
## 6  Sev    1
## 7  Typ 1360
## [1] "--------------"
## [1] "FireplaceQu"
## [1] "----"
##      f   n
## 1   Ex  24
## 2   Fa  33
## 3   Gd 380
## 4   Po  20
## 5   TA 313
## 6 <NA> 690
## [1] "--------------"
## [1] "GarageType"
## [1] "----"
##         f   n
## 1  2Types   6
## 2  Attchd 870
## 3 Basment  19
## 4 BuiltIn  88
## 5 CarPort   9
## 6  Detchd 387
## 7    <NA>  81
## [1] "--------------"
## [1] "GarageFinish"
## [1] "----"
##      f   n
## 1  Fin 352
## 2  RFn 422
## 3  Unf 605
## 4 <NA>  81
## [1] "--------------"
## [1] "GarageQual"
## [1] "----"
##      f    n
## 1   Ex    3
## 2   Fa   48
## 3   Gd   14
## 4   Po    3
## 5   TA 1311
## 6 <NA>   81
## [1] "--------------"
## [1] "GarageCond"
## [1] "----"
##      f    n
## 1   Ex    2
## 2   Fa   35
## 3   Gd    9
## 4   Po    7
## 5   TA 1326
## 6 <NA>   81
## [1] "--------------"
```

```
## [1] "PavedDrive"
## [1] "----"
##   f    n
## 1 N   90
## 2 P   30
## 3 Y 1340
## [1] "--------------"
## [1] "PoolQC"
## [1] "----"
##      f    n
## 1   Ex    2
## 2   Fa    2
## 3   Gd    3
## 4 <NA> 1453
## [1] "--------------"
## [1] "Fence"
## [1] "----"
##       f    n
## 1 GdPrv   59
## 2  GdWo   54
## 3 MnPrv  157
## 4  MnWw   11
## 5  <NA> 1179
## [1] "--------------"
## [1] "MiscFeature"
## [1] "----"
##      f    n
## 1 Gar2    2
## 2 Othr    2
## 3 Shed   49
## 4 TenC    1
## 5 <NA> 1406
## [1] "--------------"
## [1] "SaleType"
## [1] "----"
##       f    n
## 1   COD   43
## 2   Con    2
## 3 ConLD    9
## 4 ConLI    5
## 5 ConLw    5
## 6   CWD    4
## 7   New  122
## 8   Oth    3
## 9    WD 1267
## [1] "--------------"
## [1] "SaleCondition"
## [1] "----"
##         f    n
## 1 Abnorml  101
## 2 AdjLand    4
## 3  Alloca   12
## 4  Family   20
## 5  Normal 1198
```

```
## 6 Partial  125
## [1] "--------------"
```

Handle Categorical Features

MSZoning (Mei)

There are no null/missing values in the training set, but there are a few in the test set

```
sum(is.na(train$MSZoning))
```

```
## [1] 0
```

```
sum(is.na(test$MSZoning))
```

```
## [1] 4
```

Although there are 8 potential categories for this variable, there only exist 5 unique ones in the training and test set.

```
fct_count(train$MSZoning)
```

```
## # A tibble: 5 x 2
##   f           n
##   <fct>   <int>
## 1 C (all)    10
## 2 FV         65
## 3 RH         16
## 4 RL       1151
## 5 RM        218
```

```
fct_count(test$MSZoning)
```

```
## # A tibble: 6 x 2
##   f           n
##   <fct>   <int>
## 1 C (all)    15
## 2 FV         74
## 3 RH         10
## 4 RL       1114
## 5 RM        242
## 6 <NA>        4
```

```
mszoning.collapse <- function(x) fct_collapse(x,
                   "FV" = c("FV"),
                   "RL" = c("RL", "RP"),
                   "RO" = c("RM", "RH"),
                   other_level = "other")

train <- train %>% mutate(MSZoning = as.factor(MSZoning), MSZoning = mszoning.collapse(MSZoning))
test <- test %>% mutate(MSZoning = as.factor(MSZoning), MSZoning = mszoning.collapse(MSZoning))
```

```
fct_count(train$MSZoning)
```

```
## # A tibble: 4 x 2
##   f         n
##   <fct> <int>
## 1 FV       65
## 2 RO      234
```

```
## 3 RL     1151
## 4 other    10
```

MSSubClass (Mei)

There are no null/missing values

```
sum(is.na(train$MSSubClass))
```

```
## [1] 0
```

```
sum(is.na(test$MSSubClass))
```

```
## [1] 0
```

Assuming the 1/2 story refers to a basement level as "(un)finished" terminology typically refers to, the categories will be split as follows (counts in parenthesis): - 1-STORY 1946 & NEWER single-family (536) - 1-STORY single-family other - 30 1-STORY 1945 & OLDER (69) - 40 1-STORY W/FINISHED ATTIC ALL AGES (4) - 45 1-1/2 STORY - UNFINISHED ALL AGES (12) - 50 1-1/2 STORY FINISHED ALL AGES (144) - multi-level single-family non PUD - 60 2-STORY 1946 & NEWER (299) - 70 2-STORY 1945 & OLDER (60) - 75 2-1/2 STORY ALL AGES (16) - 80 SPLIT OR MULTI-LEVEL (58) - 85 SPLIT FOYER (20) - other - 90 DUPLEX - ALL STYLES AND AGES (52) - 120 1-STORY PUD (Planned Unit Development) - 1946 & NEWER (87) - 150 1-1/2 STORY PUD - ALL AGES - 160 2-STORY PUD - 1946 & NEWER (63) - 180 PUD - MULTILEVEL - INCL SPLIT LEV/FOYER (10) - 190 2 FAMILY CONVERSION - ALL STYLES AND AGES (30)

```
mssubclass.collapse <- function(x) fct_collapse(x,
                  "1-story single-family 1946 & newer" = c("20"),
                  "1-story single-family other" = c("30", "40", "45", "50"),
                  "multi-level single-family non PUD" = c("60", "70", "75", "80", "85"),
                  other_level = "other")

train <- train %>% mutate(MSSubClass = as.factor(MSSubClass), MSSubClass = mssubclass.collapse(MSSubCla
test <- test %>% mutate(MSSubClass = as.factor(MSSubClass), MSSubClass = mssubclass.collapse(MSSubClass)
```

```
fct_count(train$MSSubClass)
```

```
## # A tibble: 4 x 2
##   f                                      n
##   <fct>                              <int>
## 1 1-story single-family 1946 & newer   536
## 2 1-story single-family other          229
## 3 multi-level single-family non PUD    453
## 4 other                                242
```

Condition1/Condition2 (Mei)

There are no null/missing values

```
sum(is.na(train$Condition1))
```

```
## [1] 0
```

```
sum(is.na(test$Condition1))
```

```
## [1] 0
```

```
sum(is.na(train$Condition2))
```

```
## [1] 0
```

```
sum(is.na(test$Condition2))
```

```
## [1] 0
```

Collapse similar locations together: - All the railroad related locations - All the park related locations - All the street related locations This results in only 4 categories: - Normal - Near railroad - Near park - Near arterial or feeder street

```
condition.collapse <- function(x) fct_collapse(x,
                  RR = c("RRNn", "RRAn", "RRNe", "RRAe"),
                  Pos = c("PosN", "PosA"),
                  St = c("Artery", "Feedr"))

train <- train %>% mutate_at(vars(Condition1, Condition2), condition.collapse)
test <- test %>% mutate_at(vars(Condition1, Condition2), condition.collapse)
```

```
fct_count(train$Condition1)
```

```
## # A tibble: 4 x 2
##   f         n
##   <fct> <int>
## 1 St      129
## 2 Norm   1260
## 3 Pos      27
## 4 RR       44
```

Richard's Features

RoofStyle

combine flat, shed as other; gambrel, mansard, gable as gable; leave others as is

```
roof_price <- train %>% group_by(RoofStyle) %>% summarize(count=n(),
  mean(SalePrice), sd(SalePrice))

roof_price
```

```
## # A tibble: 6 x 4
##   RoofStyle count `mean(SalePrice)` `sd(SalePrice)`
##   <chr>     <int>             <dbl>           <dbl>
## 1 Flat         13           194690           62523.
## 2 Gable      1141           171484.          66331.
## 3 Gambrel      11           148909.          67014.
## 4 Hip         286           218877.         111550.
## 5 Mansard       7           180568.          58058.
## 6 Shed          2           225000           49497.
```

```
train$RoofStyle <- fct_collapse(train$RoofStyle, Other = c("Flat", "Shed"))
train$RoofStyle <- fct_collapse(train$RoofStyle, Gable = c("Gable", "Gambrel", "Mansard"))
```

Let's do the same on the testing dataset:

```
test$RoofStyle <- fct_collapse(test$RoofStyle, Other = c("Flat", "Shed"))
test$RoofStyle <- fct_collapse(test$RoofStyle, Gable = c("Gable", "Gambrel", "Mansard"))
```

BldgType

Combine 2FmCon, Duplex as multifamily; leave others as is

```
bldg_price <- train %>% group_by(BldgType) %>% summarize(count=n(),
  mean(SalePrice), sd(SalePrice))

bldg_price
```

```
## # A tibble: 5 x 4
##   BldgType count `mean(SalePrice)` `sd(SalePrice)`
##   <chr>    <int>             <dbl>           <dbl>
## 1 1Fam      1220           185764.          82649.
## 2 2fmCon      31           128432.          35459.
## 3 Duplex      52           133541.          27833.
## 4 Twnhs       43           135912.          41013.
## 5 TwnhsE     114           181959.          60626.
```

```
train$BldgType <- fct_collapse(train$BldgType, MultiFam = c("2fmCon", "Duplex"))
```

Let's do the same on the testing dataset:

```
test$BldgType <- fct_collapse(test$BldgType, MultiFam = c("2fmCon", "Duplex"))
```

HouseStyle

Combine 1.5Fin, 1Story, split foyer, split level as less than 2 story; 2.5fin, 2Story as two story or greater; leave 1.5Unf and 2.5Unf as is since they drag down property values

```
style_price <- train %>% group_by(HouseStyle) %>% summarize(count=n(),
  mean(SalePrice), sd(SalePrice))

style_price
```

```
## # A tibble: 8 x 4
##   HouseStyle count `mean(SalePrice)` `sd(SalePrice)`
##   <chr>      <int>             <dbl>           <dbl>
## 1 1.5Fin       154           143117.          54278.
## 2 1.5Unf        14           110150           19036.
## 3 1Story       726           175985.          77056.
## 4 2.5Fin         8           220000          118212.
## 5 2.5Unf        11           157355.          63934.
## 6 2Story       445           210052.          87339.
## 7 SFoyer        37           135074.          30481.
## 8 SLvl          65           166703.          38305.
```

```
train$HouseStyle <- fct_collapse(train$HouseStyle, Less2story = c("1Story", "1.5Fin", "SFoyer", "SLvl"))
train$HouseStyle <- fct_collapse(train$HouseStyle, EqMore2story = c("2Story", "2.5Fin"))
```

And on the test data:

```
test$HouseStyle <- fct_collapse(test$HouseStyle, Less2story = c("1Story", "1.5Fin", "SFoyer", "SLvl"))
test$HouseStyle <- fct_collapse(test$HouseStyle, EqMore2story = c("2Story", "2.5Fin"))
```

```
## Warning: Unknown levels in `f`: 2.5Fin
```

Kyle:

```
cleanpool <- as.character(train_catPredictors$PoolQC)
cleanpool[is.na(cleanpool)] <- "none"
cleanpool <- as.factor(cleanpool)
```

```
cleanfence <- as.character(train_catPredictors$Fence)
cleanfence[is.na(cleanfence)] <- "none"
cleanfence <- as.factor(cleanfence)
```

```
cleanfunc <- as.character(train_catPredictors$Functional)
cleanfunc[cleanfunc == 'Min1' | cleanfunc == 'Min2'] <- "Minor"
cleanfunc[cleanfunc == 'Maj1' | cleanfunc == 'Maj2'] <- "Major"
cleanfunc[cleanfunc == 'Sev' | cleanfunc == 'Sal'] <- "Severe"
cleanfunc <- as.factor(cleanfunc)
```

```
train$PoolQC <- cleanpool
train$Fence <- cleanfence
train$Functional <- cleanfunc
```

We need to do the same for the test dataset, so I just copied the code block and replaced "train" by "test":

```
cleanpool <- as.character(test_catPredictors$PoolQC)
cleanpool[is.na(cleanpool)] <- "none"
cleanpool <- as.factor(cleanpool)
```

```
cleanfence <- as.character(test_catPredictors$Fence)
cleanfence[is.na(cleanfence)] <- "none"
cleanfence <- as.factor(cleanfence)
```

```
cleanfunc <- as.character(test_catPredictors$Functional)
cleanfunc[cleanfunc == 'Min1' | cleanfunc == 'Min2'] <- "Minor"
cleanfunc[cleanfunc == 'Maj1' | cleanfunc == 'Maj2'] <- "Major"
cleanfunc[cleanfunc == 'Sev' | cleanfunc == 'Sal'] <- "Severe"
cleanfunc <- as.factor(cleanfunc)
```

```
test$PoolQC <- cleanpool
test$Fence <- cleanfence
test$Functional <- cleanfunc
```

Mileva: Heating, Electrical, FireplaceQu, HeatingQC, CentralAir

The processing for the Heating, Electrical, and FireplaceQu predictors is below. The HeatingQC and CentralAir predictors did not require any additional processing.

```
# Heating: Collapsed categores with low frequencies into "other"
heating <- as.factor(train_catPredictors$Heating)
heating <- fct_other(heating, keep=c("GasA", "GasW"))
train$Heating <- heating
```

```
# Electrical: Collapsed similar categories together and handled missing values
electrical <- as.character(train_catPredictors$Electrical)

electrical <- fct_collapse(electrical, Fuse=c("FuseA", "FuseF", "FuseP"))
electrical <- fct_collapse(electrical, Other=c("Mix"))
electrical[is.na(electrical)] <- "Other"

train$Electrical <- electrical
```

```r
# Fireplace: Handled missing values
fireplace <- as.character(train_catPredictors$FireplaceQu)
fireplace[is.na(fireplace)] <- "none"
train$FireplaceQu <- as.factor(fireplace)
```

Need to do the same for test dataset:

```r
# Heating: Collapsed categores with low frequencies into "other"
heating <- as.factor(test_catPredictors$Heating)
heating <- fct_other(heating, keep=c("GasA", "GasW"))
test$Heating <- heating
```

```r
# Electrical: Collapsed similar categories together and handled missing values
electrical <- as.character(test_catPredictors$Electrical)

electrical <- fct_collapse(electrical, Fuse=c("FuseA", "FuseF", "FuseP"))
electrical <- fct_collapse(electrical, Other=c("Mix"))
```

```
## Warning: Unknown levels in `f`: Mix
```

```r
electrical[is.na(electrical)] <- "Other"
```

```
## Warning in `[<-.factor`(`*tmp*`, is.na(electrical), value = "Other"): invalid
## factor level, NA generated
```

```r
test$Electrical <- electrical
```

```r
# Fireplace: Handled missing values
fireplace <- as.character(test_catPredictors$FireplaceQu)
fireplace[is.na(fireplace)] <- "none"
test$FireplaceQu <- as.factor(fireplace)
```

Thomas: RoofMatl, Exterior1st/Exterior2nd, SaleType

RoofMatl - Dropped

1434/1460 entries in the training set are CompShg.
The off-materials aren't meaningfully different price-wise as an 'other' group. Wood Shingles ('wdshngl') does contain 2 houses in the 99th percentile sale price, but with only 6 entries I don't think it's safe to include.
I think we're better off dropping this one.

```r
train <- select(train, -c(RoofMatl))
test <- select(test, -c(RoofMatl))
```

Exterior1st/2nd

Fixed the following label mis-matches between columns:Exterior1st - WdShing,CemntBd,BrkComm, Exterior2nd - Wd Shng,CmentBd,Brk Cmn

~90% of these two variables matched. In the ~10% that didn't match, Exterior1st is generally a better predictor of sale price than Exterior2nd. I converted Exterior2nd into a boolean, TRUE if Exterior1st!=Exterior2nd.

I combined the bottom half of Exterior1st's categories into an 'Other' category. (This leaves 7, but Brick Face/Cement Board seem to be decent categories for predicting sale price, so I didn't want to drop them.)

```r
train$Exterior2nd[train$Exterior2nd=='Wd Shng'] <- 'WdShing'
train$Exterior2nd[train$Exterior2nd=='CmentBd'] <- 'CemntBd'
train$Exterior2nd[train$Exterior2nd=='Brk Cmn'] <- 'BrkComm'
```

```r
train$Exterior2nd <- train$Exterior1st!=train$Exterior2nd
train$Exterior1st <- fct_collapse(train$Exterior1st, Other = c("AsbShng","AsphShn","CBlock","ImStucc","

test$Exterior2nd[test$Exterior2nd=='Wd Shng']<- 'WdShing'
test$Exterior2nd[test$Exterior2nd=='CmentBd']<- 'CemntBd'
test$Exterior2nd[test$Exterior2nd=='Brk Cmn']<- 'BrkComm'
test$Exterior2nd <- test$Exterior1st!=test$Exterior2nd
test$Exterior1st <- fct_collapse(test$Exterior1st, Other = c("AsbShng","AsphShn","CBlock","ImStucc","Br
```

```
## Warning: Unknown levels in `f`: ImStucc, Stone
```

SaleType

WD, New, and Court deed/estate were the three most common categories, and all 3 were significant when using SaleType as sole predictor. Combined the other categories into 'Other'.

```r
train$SaleType <- fct_collapse(train$SaleType, Other = c("ConLD", "ConLw", "ConLI", "CWD", "Oth", "Con"
test$SaleType <- fct_collapse(test$SaleType, Other = c("ConLD", "ConLw", "ConLI", "CWD", "Oth", "Con"))
```

Marina: Neighborhood, GarageType, GarageFinish, GarageQual, GarageCond

```r
### Neighborhood ###
# Collapse categores with low frequencies into "other"

#Explore counts
train_catPredictors %>% count(Neighborhood, sort = TRUE)
```

```
## # A tibble: 25 x 2
##    Neighborhood     n
##    <fct>        <int>
##  1 NAmes          225
##  2 CollgCr        150
##  3 OldTown        113
##  4 Edwards        100
##  5 Somerst         86
##  6 Gilbert         79
##  7 NridgHt         77
##  8 Sawyer          74
##  9 NWAmes          73
## 10 SawyerW         59
## # ... with 15 more rows
```

```r
#Factorize
neighborhood <- as.factor(train_catPredictors$Neighborhood)

#Convert to "Other" any category that represents less than 2% of the data
neighborhood <- neighborhood %>%
  fct_lump(prop=0.03, other_level='Other')

levels(neighborhood) #New levels of the factor
```

```
##  [1] "BrkSide" "CollgCr" "Crawfor" "Edwards" "Gilbert" "Mitchel" "NAmes"
##  [8] "NridgHt" "NWAmes"  "OldTown" "Sawyer"  "SawyerW" "Somerst" "Other"
```

```r
#Update column with new values
train$Neighborhood <- neighborhood
```

Need to do the same on test data:

```
#Factorize
neighborhood <- as.factor(test_catPredictors$Neighborhood)

#Convert to "Other" any category that represents less than 2% of the data
neighborhood <- neighborhood %>%
  fct_lump(prop=0.03, other_level='Other')

levels(neighborhood) #New levels of the factor
```

```
##  [1] "BrkSide" "CollgCr" "Crawfor" "Edwards" "Gilbert" "IDOTRR"  "Mitchel"
##  [8] "NAmes"   "NridgHt" "NWAmes"  "OldTown" "Sawyer"  "SawyerW" "Somerst"
## [15] "Other"
```

```
#Update column with new values
test$Neighborhood <- neighborhood
```

Anyone sees the issue??

```
table(train$Neighborhood)
```

```
##
## BrkSide CollgCr Crawfor Edwards Gilbert Mitchel   NAmes NridgHt  NWAmes OldTown
##      58     150      51     100      79      49     225      77      73     113
##  Sawyer SawyerW Somerst   Other
##      74      59      86     266
```

```
table(test$Neighborhood)
```

```
##
## BrkSide CollgCr Crawfor Edwards Gilbert  IDOTRR Mitchel   NAmes NridgHt  NWAmes
##      50     117      52      94      86      56      65     218      89      58
## OldTown  Sawyer SawyerW Somerst   Other
##     126      77      66      96     209
```

```
### GarageType ###

#Explore counts
train_catPredictors %>% count(GarageType, sort = TRUE)
```

```
## # A tibble: 7 x 2
##   GarageType     n
##   <fct>      <int>
## 1 Attchd       870
## 2 Detchd       387
## 3 BuiltIn       88
## 4 <NA>          81
## 5 Basment       19
## 6 CarPort        9
## 7 2Types         6
```

```
#Handle NAs
#According to the data description, NA means no garage.
#Change NA category to "none" to avoid issues.
garageType <- as.character(train_catPredictors$GarageType)
garageType[is.na(garageType)] <- "none"
```

```r
garageType <- as.factor(garageType)
```

```r
#Collapse into "Other" categries that represent less than 5% of the data
garageType <- garageType %>%
  fct_lump(prop=0.05, other_level='Other')

#levels(garageType) #New levels of the factor
```

```r
#Update column with new values
train$GarageType <- garageType
```

Attention!! Need to do the same on the test data:

```r
### GarageFinish ###

#Explore counts
train_catPredictors %>% count(GarageFinish, sort = TRUE)
```

```
## # A tibble: 4 x 2
##   GarageFinish     n
##   <fct>        <int>
## 1 Unf            605
## 2 RFn            422
## 3 Fin            352
## 4 <NA>            81
```

```r
#Handle NAs
#According to the data description, NA means no garage.
#Change NA category to "none" to avoid issues.
garageFinish <- as.character(train_catPredictors$GarageFinish)
garageFinish[is.na(garageFinish)] <- "none"
garageFinish <- as.factor(garageFinish)

#No need to collapse categories
```

```r
#Update column with new values
train$GarageFinish <- garageFinish
```

Need to do the same for the test data:

```r
#Handle NAs
#According to the data description, NA means no garage.
#Change NA category to "none" to avoid issues.
garageFinish <- as.character(test_catPredictors$GarageFinish)
garageFinish[is.na(garageFinish)] <- "none"
garageFinish <- as.factor(garageFinish)
#No need to collapse categories
```

```r
#Update column with new values
test$GarageFinish <- garageFinish
```

```r
### GarageQual ###

#Explore counts
train_catPredictors %>% count(GarageQual, sort = TRUE)
```

```
## # A tibble: 6 x 2
##   GarageQual     n
```

```
##   <fct>      <int>
## 1 TA          1311
## 2 <NA>          81
## 3 Fa            48
## 4 Gd            14
## 5 Ex             3
## 6 Po             3
```

```
#Handle NAs
#According to the data description, NA means no garage.
#Change NA category to "none" to avoid issues.
garageQual <- as.character(train_catPredictors$GarageQual)
garageQual[is.na(garageQual)] <- "none"
garageQual <- as.factor(garageQual)
```

```
#Collapse categories:
# - Let's collapse Ex   (Excellent) and Gd  (Good) into 1 category: Gd
# - Let's collapse Fa   (Fair) and Po   (Poor) into 1 category: Po
# - None and TA remains the same

garageQual <- fct_collapse(garageQual, Gd = c("Ex","Gd"))
garageQual <- fct_collapse(garageQual, Po = c("Fa","Po"))
```

```
#Update column with new values
train$GarageQual <- garageQual
```

Need to do the same for test data:

```
#Handle NAs
#According to the data description, NA means no garage.
#Change NA category to "none" to avoid issues.
garageQual <- as.character(test_catPredictors$GarageQual)
garageQual[is.na(garageQual)] <- "none"
garageQual <- as.factor(garageQual)
```

```
#Collapse categories:
# - Let's collapse Ex   (Excellent) and Gd  (Good) into 1 category: Gd
# - Let's collapse Fa   (Fair) and Po   (Poor) into 1 category: Po
# - None and TA remains the same

garageQual <- fct_collapse(garageQual, Gd = c("Ex","Gd"))
```

```
## Warning: Unknown levels in `f`: Ex
```

```
garageQual <- fct_collapse(garageQual, Po = c("Fa","Po"))
```

```
#Update column with new values
test$GarageQual <- garageQual
```

```
### GarageCond ###

#Explore counts
train_catPredictors %>% count(GarageCond, sort = TRUE)
```

```
## # A tibble: 6 x 2
##   GarageCond     n
##   <fct>      <int>
## 1 TA          1326
```

```
## 2 <NA>          81
## 3 Fa            35
## 4 Gd             9
## 5 Po             7
## 6 Ex             2
```

```
#Handle NAs
#According to the data description, NA means no garage.
#Change NA category to "none" to avoid issues.
garageCond <- as.character(train_catPredictors$GarageCond)
garageCond[is.na(garageCond)] <- "none"
garageCond <- as.factor(garageCond)
```

```
#Collapse categories:
# - Let's collapse Ex   (Excellent) and Gd  (Good) into 1 category: Gd
# - Let's collapse Fa   (Fair) and Po   (Poor) into 1 category: Po
# - None and TA remains the same

garageCond <- fct_collapse(garageCond, Gd = c("Ex","Gd"))
garageCond <- fct_collapse(garageCond, Po = c("Fa","Po"))
```

```
#Update column with new values
train$GarageCond <- garageCond
```

Need to do the same with test data:

```
#Handle NAs
#According to the data description, NA means no garage.
#Change NA category to "none" to avoid issues.
garageCond <- as.character(test_catPredictors$GarageCond)
garageCond[is.na(garageCond)] <- "none"
garageCond <- as.factor(garageCond)
```

```
#Collapse categories:
# - Let's collapse Ex   (Excellent) and Gd  (Good) into 1 category: Gd
# - Let's collapse Fa   (Fair) and Po   (Poor) into 1 category: Po
# - None and TA remains the same

garageCond <- fct_collapse(garageCond, Gd = c("Ex","Gd"))
garageCond <- fct_collapse(garageCond, Po = c("Fa","Po"))
```

```
#Update column with new values
test$GarageCond <- garageCond
```

Paul: LotShape, LotConfig, LandContour

Fortunately there are no NA values in either the test or train sets.

```
sum(is.na(train$LotShape))
```

```
## [1] 0
```

```
sum(is.na(test$LotShape))
```

```
## [1] 0
```

```
sum(is.na(train$LotConfig))
```

```
## [1] 0
```

```r
sum(is.na(test$LotConfig))
```

```
## [1] 0
```

```r
sum(is.na(train$LandContour))
```

```
## [1] 0
```

```r
sum(is.na(test$LandContour))
```

```
## [1] 0
```

```r
fct_count(train$LotShape)
```

```
## # A tibble: 4 x 2
##   f         n
##   <fct> <int>
## 1 IR1     484
## 2 IR2      41
## 3 IR3      10
## 4 Reg     925
```

```r
fct_count(test$LotShape)
```

```
## # A tibble: 4 x 2
##   f         n
##   <fct> <int>
## 1 IR1     484
## 2 IR2      35
## 3 IR3       6
## 4 Reg     934
```

```r
fct_count(train$LotConfig)
```

```
## # A tibble: 5 x 2
##   f           n
##   <fct>   <int>
## 1 Corner    263
## 2 CulDSac    94
## 3 FR2        47
## 4 FR3         4
## 5 Inside   1052
```

```r
fct_count(test$LotConfig)
```

```
## # A tibble: 5 x 2
##   f           n
##   <fct>   <int>
## 1 Corner    248
## 2 CulDSac    82
## 3 FR2        38
## 4 FR3        10
## 5 Inside   1081
```

```r
fct_count(train$LandContour)
```

```
## # A tibble: 4 x 2
##   f         n
##   <fct> <int>
```

```
## 1 Bnk      63
## 2 HLS      50
## 3 Low      36
## 4 Lvl    1311
```

```
fct_count(test$LandContour)
```

```
## # A tibble: 4 x 2
##   f         n
##   <fct> <int>
## 1 Bnk      54
## 2 HLS      70
## 3 Low      24
## 4 Lvl    1311
```

All of these variables are highly imbalanced. In each there is one category that represents a "regular" shape, configuration, or land contour, which amount for ~2/3 or more of the total instances. Thus, I collapsed all of the less represented "irregular" categories into one.

```
train$LotShape <- fct_collapse(train$LotShape, Irregular = c("IR1", "IR2", "IR3"))
train$LotConfig <- fct_collapse(train$LotConfig, Other = c("Corner","CulDSac", "FR2", "FR3"))
train$LandContour <- fct_collapse(train$LandContour, NonLvl = c("Bnk", "HLS", "Low"))
```

```
fct_count(train$LotShape)
```

```
## # A tibble: 2 x 2
##   f             n
##   <fct>     <int>
## 1 Irregular   535
## 2 Reg         925
```

```
fct_count(train$LotConfig)
```

```
## # A tibble: 2 x 2
##   f          n
##   <fct>  <int>
## 1 Other    408
## 2 Inside  1052
```

```
fct_count(train$LandContour)
```

```
## # A tibble: 2 x 2
##   f          n
##   <fct>  <int>
## 1 NonLvl   149
## 2 Lvl     1311
```

Need to do the same for the test data:

```
test$LotShape <- fct_collapse(test$LotShape, Irregular = c("IR1", "IR2", "IR3"))
test$LotConfig <- fct_collapse(test$LotConfig, Other = c("Corner","CulDSac", "FR2", "FR3"))
test$LandContour <- fct_collapse(test$LandContour, NonLvl = c("Bnk", "HLS", "Low"))
```