Team5_HW4

Tyler Brassfield

2024-09-20

```
# Required packages
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(dplyr)
library(forcats)
library(ggplot2)
library(knitr)
library(mice)
```

```
##
## Attaching package: 'mice'
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
library(moments)
library(EnvStats)
```

```
##
## Attaching package: 'EnvStats'
##
## The following objects are masked from 'package:moments':
##
##     kurtosis, skewness
```

```
##
## The following objects are masked from 'package:stats':
##
##     predict, predict.lm
```

```
library(car)
```

```
## Loading required package: carData
##
## Attaching package: 'car'
##
## The following object is masked from 'package:EnvStats':
##
##     qqPlot
##
## The following object is masked from 'package:dplyr':
##
##     recode
##
## The following object is masked from 'package:purrr':
##
##     some
```

— Question 1: Data Quality Report —

```
########## (a) ##########

df <- tibble(read.csv('housingData.csv'))
df <- df %>%
  dplyr::mutate(age = YrSold - YearBuilt,
                ageSinceRemodel = YrSold - YearRemodAdd,
                ageofGarage = YrSold - GarageYrBlt)

    # Created 3 columns as functions of 3 existing columns within the data
    # based on the above equations

# (b) Create a tibble named "housingNumeric" containing all numeric variables
#     from the original housing data

housingNumeric <- df %>%
  dplyr::select(where(is.numeric))

# (c) Create a tibble named "housingNumeric" containing all non-numeric
#     variables from the original housing data

housingFactor <- df %>%
  dplyr::transmute(across(where(is.character), as.factor))

# (d) Glimpse into the newly created tables

# glimpse(housingNumeric) # new tibble made up of only "int" data types
# glimpse(housingFactor)  # new tibble made up of only factors

head(housingNumeric)
```

```
## # A tibble: 6 x 39
##      Id MSSubClass LotFrontage LotArea OverallQual OverallCond YearBuilt
##   <int>      <int>       <int>   <int>       <int>       <int>     <int>
## 1     1         20          NA   11000           5           6      1966
## 2     2         20          NA   36500           5           5      1964
## 3     3         20          57    9764           5           7      1967
## 4     4         70          NA    7500           6           7      1942
## 5     5         20          80    9200           6           6      1965
## 6     6         60          72   11317           7           5      2003
## # i 32 more variables: YearRemodAdd <int>, MasVnrArea <int>, BsmtFinSF1 <int>,
## #   BsmtFinSF2 <int>, BsmtUnfSF <int>, TotalBsmtSF <int>, X1stFlrSF <int>,
## #   X2ndFlrSF <int>, LowQualFinSF <int>, GrLivArea <int>, BsmtFullBath <int>,
## #   BsmtHalfBath <int>, FullBath <int>, HalfBath <int>, BedroomAbvGr <int>,
## #   KitchenAbvGr <int>, TotRmsAbvGrd <int>, Fireplaces <int>,
## #   GarageYrBlt <int>, GarageCars <int>, GarageArea <int>, WoodDeckSF <int>,
## #   OpenPorchSF <int>, EncPorchSF <int>, PoolArea <int>, MiscVal <int>, ...
```

```
head(housingFactor)
```

```
## # A tibble: 6 x 38
##   MSZoning Alley LotShape LandContour LotConfig LandSlope Neighborhood
##   <fct>    <fct> <fct>    <fct>       <fct>     <fct>     <fct>
## 1 RL       <NA>  IR1      Lvl         CulDSac   Gtl       NAmes
## 2 RL       <NA>  IR1      Low         Inside    Mod       ClearCr
## 3 RL       <NA>  IR1      Lvl         other     Gtl       Sawyer
## 4 RL       <NA>  IR1      Bnk         Inside    Gtl       Crawfor
## 5 RL       <NA>  Reg      Lvl         Inside    Gtl       NAmes
## 6 RL       <NA>  Reg      Lvl         Inside    Gtl       CollgCr
## # i 31 more variables: Condition1 <fct>, BldgType <fct>, HouseStyle <fct>,
## #   RoofStyle <fct>, Exterior1st <fct>, Exterior2nd <fct>, MasVnrType <fct>,
## #   ExterQual <fct>, ExterCond <fct>, Foundation <fct>, BsmtQual <fct>,
## #   BsmtCond <fct>, BsmtExposure <fct>, BsmtFinType1 <fct>, BsmtFinType2 <fct>,
## #   Heating <fct>, HeatingQC <fct>, CentralAir <fct>, Electrical <fct>,
## #   KitchenQual <fct>, Functional <fct>, FireplaceQu <fct>, GarageType <fct>,
## #   GarageFinish <fct>, GarageQual <fct>, GarageCond <fct>, ...
```

```
# (e) Create functions for calculating 1st and 3rd quartiles

Q1 <- function(x, na.rm = TRUE) {
      quantile(x, na.rm = na.rm)[2]
}

Q3 <- function(x, na.rm = TRUE) {
      quantile(x, na.rm = na.rm)[4]
}

    # The above functions compute the quantiles for numeric variables
    # within the data. Because the quantile() function returns a list
    # of 5 numbers, to return the 1st and 3rd quartiles, we extract the
    # 2nd and 4th elements from the list. Additionally, we are choosing
    # to exclude missing values from the quantile calculations.

# (f) Create a summary for our numeric variables
```

```
myNumericSummary <- function(x){
  c(length(x), n_distinct(x), sum(is.na(x)), mean(x, na.rm = TRUE),
  min(x, na.rm = TRUE), Q1(x, na.rm = TRUE), median(x, na.rm = TRUE),
  Q3(x, na.rm = TRUE), max(x, na.rm = TRUE), sd(x, na.rm = TRUE))
}

# (g) Create a tibble of summary statistics for the numerical data

numericSummary <- housingNumeric %>%
  summarize(
    across(everything(), myNumericSummary)
  )
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
# (h) Create column names for clarity/tidiness

numericSummary <- cbind(
                  Statistic = c("n", "unique","missing","mean","min",
                                "Q1","median","Q3","max","sd"),
                  numericSummary)
glimpse(numericSummary)
```

```
## Rows: 10
## Columns: 40
## $ Statistic       <chr> "n", "unique", "missing", "mean", "min", "Q1", "median~
## $ Id              <dbl> 1000.0000, 1000.0000, 0.0000, 500.5000, 1.0000, 250.75~
## $ MSSubClass      <dbl> 1000.00000, 13.00000, 0.00000, 57.18500, 20.00000, 20.~
## $ LotFrontage     <dbl> 1000.00000, 102.00000, 207.00000, 68.74527, 21.00000, ~
## $ LotArea         <dbl> 1000.000, 760.000, 0.000, 10424.881, 1477.000, 7500.00~
## $ OverallQual     <dbl> 1000.000000, 10.000000, 0.000000, 5.979000, 1.000000, ~
## $ OverallCond     <dbl> 1000.000000, 8.000000, 0.000000, 5.638000, 2.000000, 5~
## $ YearBuilt       <dbl> 1000.00000, 108.00000, 0.00000, 1969.83600, 1875.00000~
## $ YearRemodAdd    <dbl> 1000.00000, 61.00000, 0.00000, 1984.10800, 1950.00000,~
## $ MasVnrArea      <dbl> 1000.00000, 249.00000, 4.00000, 95.41767, 0.00000, 0.0~
## $ BsmtFinSF1      <dbl> 1000.0000, 490.0000, 0.0000, 438.6860, 0.0000, 0.0000,~
## $ BsmtFinSF2      <dbl> 1000.000, 107.000, 0.000, 44.296, 0.000, 0.000, 0.000,~
## $ BsmtUnfSF       <dbl> 1000.0000, 598.0000, 0.0000, 535.0780, 0.0000, 208.000~
## $ TotalBsmtSF     <dbl> 1000.0000, 549.0000, 0.0000, 1018.0600, 0.0000, 793.00~
## $ X1stFlrSF       <dbl> 1000.0000, 581.0000, 0.0000, 1131.2510, 334.0000, 868.~
## $ X2ndFlrSF       <dbl> 1000.0000, 306.0000, 0.0000, 346.2790, 0.0000, 0.0000,~
## $ LowQualFinSF    <dbl> 1000.00000, 15.00000, 0.00000, 4.99100, 0.00000, 0.000~
## $ GrLivArea       <dbl> 1000.000, 664.000, 0.000, 1482.521, 334.000, 1110.750,~
## $ BsmtFullBath    <dbl> 1000.0000000, 3.0000000, 0.0000000, 0.4270000, 0.00000~
## $ BsmtHalfBath    <dbl> 1000.0000000, 2.0000000, 0.0000000, 0.0590000, 0.00000~
## $ FullBath        <dbl> 1000.0000000, 4.0000000, 0.0000000, 1.5290000, 0.00000~
```

```
## $ HalfBath        <dbl> 1000.0000000, 3.0000000, 0.0000000, 0.3840000, 0.00000~
## $ BedroomAbvGr    <dbl> 1000.0000000, 7.0000000, 0.0000000, 2.8650000, 0.00000~
## $ KitchenAbvGr    <dbl> 1000.0000000, 3.0000000, 0.0000000, 1.0410000, 1.00000~
## $ TotRmsAbvGrd    <dbl> 1000.000000, 11.000000, 0.000000, 6.410000, 2.000000, ~
## $ Fireplaces      <dbl> 1000.0000000, 4.0000000, 0.0000000, 0.6180000, 0.00000~
## $ GarageYrBlt     <dbl> 1000.00000, 94.00000, 53.00000, 1976.93770, 1906.00000~
## $ GarageCars      <dbl> 1000.0000000, 5.0000000, 0.0000000, 1.7200000, 0.00000~
## $ GarageArea      <dbl> 1000.0000, 353.0000, 0.0000, 458.3290, 0.0000, 318.750~
## $ WoodDeckSF      <dbl> 1000.0000, 226.0000, 0.0000, 94.5550, 0.0000, 0.0000, ~
## $ OpenPorchSF     <dbl> 1000.00000, 169.00000, 0.00000, 43.61000, 0.00000, 0.0~
## $ EncPorchSF      <dbl> 1000.0000, 122.0000, 0.0000, 40.6410, 0.0000, 0.0000, ~
## $ PoolArea        <dbl> 1000.00000, 3.00000, 0.00000, 1.22400, 0.00000, 0.0000~
## $ MiscVal         <dbl> 1000.0000, 14.0000, 0.0000, 27.2100, 0.0000, 0.0000, 0~
## $ MoSold          <dbl> 1000.000000, 12.000000, 0.000000, 6.207000, 1.000000, ~
## $ YrSold          <dbl> 1000.00000, 5.00000, 0.00000, 2007.91900, 2006.00000, ~
## $ SalePrice       <dbl> 1000.00, 477.00, 0.00, 174560.61, 39300.00, 130000.00,~
## $ age             <dbl> 1000.00000, 115.00000, 0.00000, 38.08300, 1.00000, 10.~
## $ ageSinceRemodel <dbl> 1000.00000, 61.00000, 0.00000, 23.81100, 0.00000, 6.00~
## $ ageofGarage     <dbl> 1000.00000, 97.00000, 53.00000, 30.97254, 0.00000, 9.0~
```

```
# (i) Pivot the data and add computed values


numericSummaryFinal <- numericSummary %>%
    pivot_longer("Id":"ageofGarage", names_to ="variable",
                 values_to = "value") %>%
    pivot_wider(names_from = Statistic, values_from = value) %>%
    mutate(missing_pct = 100*missing/n,
           unique_pct = 100*unique/n) %>%
    select(variable, n, missing, missing_pct, unique, unique_pct, everything())
options(digits = 3)
options(scipen = 99)
numericSummaryFinal %>% kable()
```

variable	n	missing	missing_pct	unique	unique_pct	mean	min	Q1	median	Q3	max	sd
Id	1000	0	0.0	1000	100.0	500.500	1	251	500	750.2	1000	288.819
MSSubClass	1000	0	0.0	13	1.3	57.185	20	20	50	70.0	190	41.875
LotFrontage	1000	207	20.7	102	10.2	68.745	21	58	68	80.0	313	23.198
LotArea	1000	0	0.0	760	76.0	10424.881	1477	7500	9422	11423.5	215245	9940.619
OverallQual	1000	0	0.0	10	1.0	5.979	1	5	6	7.0	10	1.310
OverallCond	1000	0	0.0	8	0.8	5.638	2	5	5	6.0	9	1.114
YearBuilt	1000	0	0.0	108	10.8	1969.836	1875	1954	1971	1998.0	2009	29.119
YearRemodAdd	1000	0	0.0	61	6.1	1984.108	1950	1967	1992	2002.0	2010	20.116
MasVnrArea	1000	4	0.4	249	24.9	95.418	0	0	0	146.2	1600	177.318
BsmtFinSF1	1000	0	0.0	490	49.0	438.686	0	0	400	700.0	1880	405.837
BsmtFinSF2	1000	0	0.0	107	10.7	44.296	0	0	0	0.0	1127	150.493
BsmtUnfSF	1000	0	0.0	598	59.8	535.078	0	208	441	779.2	2153	417.944
TotalBsmtSF	1000	0	0.0	549	54.9	1018.060	0	793	962	1223.5	3206	403.641
X1stFlrSF	1000	0	0.0	581	58.1	1131.251	334	868	1060	1327.2	3228	350.862
X2ndFlrSF	1000	0	0.0	306	30.6	346.279	0	0	0	735.0	1872	426.395
LowQualFinSF	1000	0	0.0	15	1.5	4.991	0	0	0	0.0	528	45.295
GrLivArea	1000	0	0.0	664	66.4	1482.521	334	1111	1442	1735.0	4316	490.566

variable	n	missing	missing_pct	unique	unique_pct	mean	min	Q1	median	Q3	max	sd
BsmtFullBath	1000	0	0.0	3	0.3	0.427	0	0	0	1.0	2	0.509
BsmtHalfBath	1000	0	0.0	2	0.2	0.059	0	0	0	0.0	1	0.236
FullBath	1000	0	0.0	4	0.4	1.529	0	1	2	2.0	3	0.531
HalfBath	1000	0	0.0	3	0.3	0.384	0	0	0	1.0	2	0.501
BedroomAbvGr	1000	0	0.0	7	0.7	2.865	0	2	3	3.0	6	0.791
KitchenAbvGr	1000	0	0.0	3	0.3	1.041	1	1	1	1.0	3	0.203
TotRmsAbvGrd	1000	0	0.0	11	1.1	6.410	2	5	6	7.0	12	1.562
Fireplaces	1000	0	0.0	4	0.4	0.618	0	0	1	1.0	3	0.642
GarageYrBlt	1000	53	5.3	94	9.4	1976.938	1906	1960	1977	1999.0	2009	23.592
GarageCars	1000	0	0.0	5	0.5	1.720	0	1	2	2.0	4	0.714
GarageArea	1000	0	0.0	353	35.3	458.329	0	319	470	572.0	1356	197.780
WoodDeckSF	1000	0	0.0	226	22.6	94.555	0	0	0	168.0	857	127.144
OpenPorchSF	1000	0	0.0	169	16.9	43.610	0	0	22	64.0	547	61.915
EncPorchSF	1000	0	0.0	122	12.2	40.641	0	0	0	0.0	508	82.139
PoolArea	1000	0	0.0	3	0.3	1.224	0	0	0	0.0	648	27.403
MiscVal	1000	0	0.0	14	1.4	27.210	0	0	0	0.0	3500	190.707
MoSold	1000	0	0.0	12	1.2	6.207	1	4	6	8.0	12	2.626
YrSold	1000	0	0.0	5	0.5	2007.919	2006	2007	2008	2009.0	2010	1.318
SalePrice	1000	0	0.0	477	47.7	174560.603	39300	130000	160000	205000.0	755000	69329.319
age	1000	0	0.0	115	11.5	38.083	1	10	37	55.0	135	29.109
ageSinceRemodel	1000	0	0.0	61	6.1	23.811	0	6	16	41.2	60	20.033
ageofGarage	1000	53	5.3	97	9.7	30.973	0	9	30	48.0	102	23.563

```
# (j) Create report for the factor data

#### Create mode name and frequency retrieval functions ####

getmodenames <- function(v, type = 1) {

  tbl <- table(v)
  m1 <- which.max(tbl)

  if (type == 1) {
        return (names(m1))                    # 1st mode
  }
  else if (type == 2) {
        return (names(which.max(tbl[-m1])))   # 2nd mode
  }
  else if (type == -1) {
        return (names(which.min(tbl)))        # least common mode
  }
  else {
        stop("Invalid type selected")
  }
}

getmodes <- function(v, type = 1) {

  tbl <- table(v)
  m1 <- which.max(tbl)
```

```
  if (type == 1) {
        return (max(tbl))          # 1st mode frequency
  }
  else if (type == 2) {
        return (max(tbl[-m1]))      # 2nd mode frequency
  }
  else if (type == -1) {
        return (min(tbl))         # least common frequency
  }
  else {
    stop("Invalid type selected")
  }
}

#############################################################

myFactorSummary <- function(x){
    c(length(x), n_distinct(x), sum(is.na(x)), getmodenames(x, type = 1),
      getmodes(x, type = 1), getmodenames(x, type = 2), getmodes(x, type = 2),
      getmodenames(x, type = -1), getmodes(x, type = -1))
}

factorSummary <- housingFactor %>%
    summarize(
      across(everything(), myFactorSummary)
  )
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
factorSummary <- cbind(Statistic = c("n","unique","missing",
                                     "1st mode","1st mode freq",
                                     "2nd mode","2nd mode freq",
                                     "least common","least common freq"),
                                      factorSummary)

glimpse(factorSummary)
```

```
## Rows: 9
## Columns: 39
## $ Statistic    <chr> "n", "unique", "missing", "1st mode", "1st mode freq", "2~
## $ MSZoning     <chr> "1000", "4", "0", "RL", "803", "RM", "151", "RH", "10"
## $ Alley        <chr> "1000", "3", "938", "Grvl", "40", "Pave", "22", "Pave", "~
## $ LotShape     <chr> "1000", "4", "0", "Reg", "633", "IR1", "330", "IR3", "7"
## $ LandContour  <chr> "1000", "4", "0", "Lvl", "905", "Bnk", "40", "Low", "26"
## $ LotConfig    <chr> "1000", "4", "0", "Inside", "711", "Corner", "179", "othe~
## $ LandSlope    <chr> "1000", "3", "0", "Gtl", "946", "Mod", "48", "Sev", "6"
## $ Neighborhood <chr> "1000", "18", "0", "NAmes", "167", "CollgCr", "113", "Tim~
```

```
## $ Condition1   <chr> "1000", "6", "0", "Norm", "871", "Feedr", "51", "PosA", "~
## $ BldgType     <chr> "1000", "5", "0", "1Fam", "837", "TwnhsE", "81", "2fmCon"~
## $ HouseStyle   <chr> "1000", "8", "0", "1Story", "488", "2Story", "310", "2.5F~
## $ RoofStyle    <chr> "1000", "3", "0", "Gable", "795", "Hip", "184", "other", ~
## $ Exterior1st  <chr> "1000", "8", "0", "VinylSd", "328", "HdBoard", "175", "Ce~
## $ Exterior2nd  <chr> "1000", "9", "0", "VinylSd", "320", "HdBoard", "159", "Br~
## $ MasVnrType   <chr> "1000", "5", "4", "None", "617", "BrkFace", "313", "BrkCm~
## $ ExterQual    <chr> "1000", "3", "0", "Avg", "657", "AboveAvg", "336", "Below~
## $ ExterCond    <chr> "1000", "3", "0", "Avg", "880", "AboveAvg", "103", "Below~
## $ Foundation   <chr> "1000", "4", "0", "CBlock", "463", "PConc", "414", "other~
## $ BsmtQual     <chr> "1000", "4", "31", "AboveAvg", "488", "Avg", "459", "Belo~
## $ BsmtCond     <chr> "1000", "4", "31", "Avg", "903", "AboveAvg", "37", "Below~
## $ BsmtExposure <chr> "1000", "5", "32", "No", "668", "Av", "140", "Mn", "76"
## $ BsmtFinType1 <chr> "1000", "7", "31", "GLQ", "273", "Unf", "265", "LwQ", "52"
## $ BsmtFinType2 <chr> "1000", "7", "32", "Unf", "853", "Rec", "36", "ALQ", "11"
## $ Heating      <chr> "1000", "2", "0", "GasA", "974", "other", "26", "other", ~
## $ HeatingQC    <chr> "1000", "3", "0", "AboveAvg", "664", "Avg", "300", "Below~
## $ CentralAir   <chr> "1000", "2", "0", "Y", "936", "N", "64", "N", "64"
## $ Electrical   <chr> "1000", "5", "1", "SBrkr", "908", "FuseA", "72", "FuseP",~
## $ KitchenQual  <chr> "1000", "3", "0", "Avg", "534", "AboveAvg", "439", "Below~
## $ Functional   <chr> "1000", "6", "0", "Typ", "924", "Min2", "26", "Maj2", "4"
## $ FireplaceQu  <chr> "1000", "4", "466", "AboveAvg", "250", "Avg", "240", "Bel~
## $ GarageType   <chr> "1000", "7", "53", "Attchd", "601", "Detchd", "280", "2Ty~
## $ GarageFinish <chr> "1000", "4", "53", "Unf", "434", "RFn", "291", "Fin", "22~
## $ GarageQual   <chr> "1000", "4", "53", "Avg", "907", "BelowAvg", "33", "Above~
## $ GarageCond   <chr> "1000", "4", "53", "Avg", "910", "BelowAvg", "31", "Above~
## $ PavedDrive   <chr> "1000", "3", "0", "Y", "912", "N", "62", "P", "26"
## $ PoolQC       <chr> "1000", "3", "998", "Fa", "1", "Gd", "1", "Fa", "1"
## $ Fence        <chr> "1000", "5", "805", "MnPrv", "108", "GdPrv", "40", "MnWw"~
## $ MiscFeature  <chr> "1000", "3", "966", "Shed", "32", "Othr", "2", "Othr", "2"
## $ SaleType     <chr> "1000", "2", "0", "WD", "971", "other", "29", "other", "2~
```

```
factorSummaryFinal <- factorSummary %>%
    pivot_longer("MSZoning":"SaleType", names_to = "variable",
                 values_to = "value") %>%
    pivot_wider(names_from = Statistic, values_from = value) %>%
    mutate(missing=as.numeric(missing),
           n = as.numeric(n),
           unique = as.numeric(unique),
           `1st mode freq` = as.numeric(`1st mode freq`),
           `2nd mode freq` = as.numeric(`2nd mode freq`)) %>%
    mutate(missing_pct = 100*missing/n,
           unique_pct = 100*unique/n,
           freqRatio = (`1st mode freq`) / (`2nd mode freq`)) %>%
    select(variable, n, missing, missing_pct, unique, unique_pct, freqRatio,
           everything())
options(digits = 3)
options(scipen = 99)
factorSummaryFinal %>% kable()
```

variable	n	missing	missing_pct	unique	unique_pct	freqRatio	1st mode	1st mode freq	2nd mode	2nd mode freq	least common	least common freq
MSZoning	1000	0	0.0	4	0.4	5.32	RL	803	RM	151	RH	10
Alley	1000	938	93.8	3	0.3	1.82	Grvl	40	Pave	22	Pave	22
LotShape	1000	0	0.0	4	0.4	1.92	Reg	633	IR1	330	IR3	7
LandContour	1000	0	0.0	4	0.4	22.62	Lvl	905	Bnk	40	Low	26
LotConfig	1000	0	0.0	4	0.4	3.97	Inside	711	Corner	179	other	38
LandSlope	1000	0	0.0	3	0.3	19.71	Gtl	946	Mod	48	Sev	6
Neighborhood	1000	0	0.0	18	1.8	1.48	NAmes	167	CollgCr	113	Timber	20
Condition1	1000	0	0.0	6	0.6	17.08	Norm	871	Feedr	51	PosA	7
BldgType	1000	0	0.0	5	0.5	10.33	1Fam	837	TwnhsE	81	2fmCon	20
HouseStyle	1000	0	0.0	8	0.8	1.57	1Story	488	2Story	310	2.5Fin	5
RoofStyle	1000	0	0.0	3	0.3	4.32	Gable	795	Hip	184	other	21
Exterior1st	1000	0	0.0	8	0.8	1.87	VinylSd	328	HdBoard	175	CemntBd	36
Exterior2nd	1000	0	0.0	9	0.9	2.01	VinylSd	320	HdBoard	159	BrkFace	24
MasVnrType	1000	4	0.4	5	0.5	1.97	None	617	BrkFace	313	BrkCmn	8
ExterQual	1000	0	0.0	3	0.3	1.96	Avg	657	AboveAvg	336	BelowAvg	7
ExterCond	1000	0	0.0	3	0.3	8.54	Avg	880	AboveAvg	103	BelowAvg	17
Foundation	1000	0	0.0	4	0.4	1.12	CBlock	463	PConc	414	other	27
BsmtQual	1000	31	3.1	4	0.4	1.06	AboveAvg	488	Avg	459	BelowAvg	22
BsmtCond	1000	31	3.1	4	0.4	24.41	Avg	903	AboveAvg	37	BelowAvg	29
BsmtExposure	1000	32	3.2	5	0.5	4.77	No	668	Av	140	Mn	76
BsmtFinType1	1000	31	3.1	7	0.7	1.03	GLQ	273	Unf	265	LwQ	52
BsmtFinType2	1000	32	3.2	7	0.7	23.69	Unf	853	Rec	36	ALQ	11
Heating	1000	0	0.0	2	0.2	37.46	GasA	974	other	26	other	26
HeatingQC	1000	0	0.0	3	0.3	2.21	AboveAvg	664	Avg	300	BelowAvg	36
CentralAir	1000	0	0.0	2	0.2	14.62	Y	936	N	64	N	64
Electrical	1000	1	0.1	5	0.5	12.61	SBrkr	908	FuseA	72	FuseP	2
KitchenQual	1000	0	0.0	3	0.3	1.22	Avg	534	AboveAvg	439	BelowAvg	27
Functional	1000	0	0.0	6	0.6	35.54	Typ	924	Min2	26	Maj2	4
FireplaceQu	1000	466	46.6	4	0.4	1.04	AboveAvg	250	Avg	240	BelowAvg	44
GarageType	1000	53	5.3	7	0.7	2.15	Attchd	601	Detchd	280	2Types	3
GarageFinish	1000	53	5.3	4	0.4	1.49	Unf	434	RFn	291	Fin	222
GarageQual	1000	53	5.3	4	0.4	27.48	Avg	907	BelowAvg	33	AboveAvg	7
GarageCond	1000	53	5.3	4	0.4	29.36	Avg	910	BelowAvg	31	AboveAvg	6
PavedDrive	1000	0	0.0	3	0.3	14.71	Y	912	N	62	P	26
PoolQC	1000	998	99.8	3	0.3	1.00	Fa	1	Gd	1	Fa	1
Fence	1000	805	80.5	5	0.5	2.70	MnPrv	108	GdPrv	40	MnWw	8
MiscFeature	1000	966	96.6	3	0.3	16.00	Shed	32	Othr	2	Othr	2
SaleType	1000	0	0.0	2	0.2	33.48	WD	971	other	29	other	29

```
    # This was the same procedure as the preceding summary; however, I needed
    # to convert a few of the variables to numeric rather than characters!
```

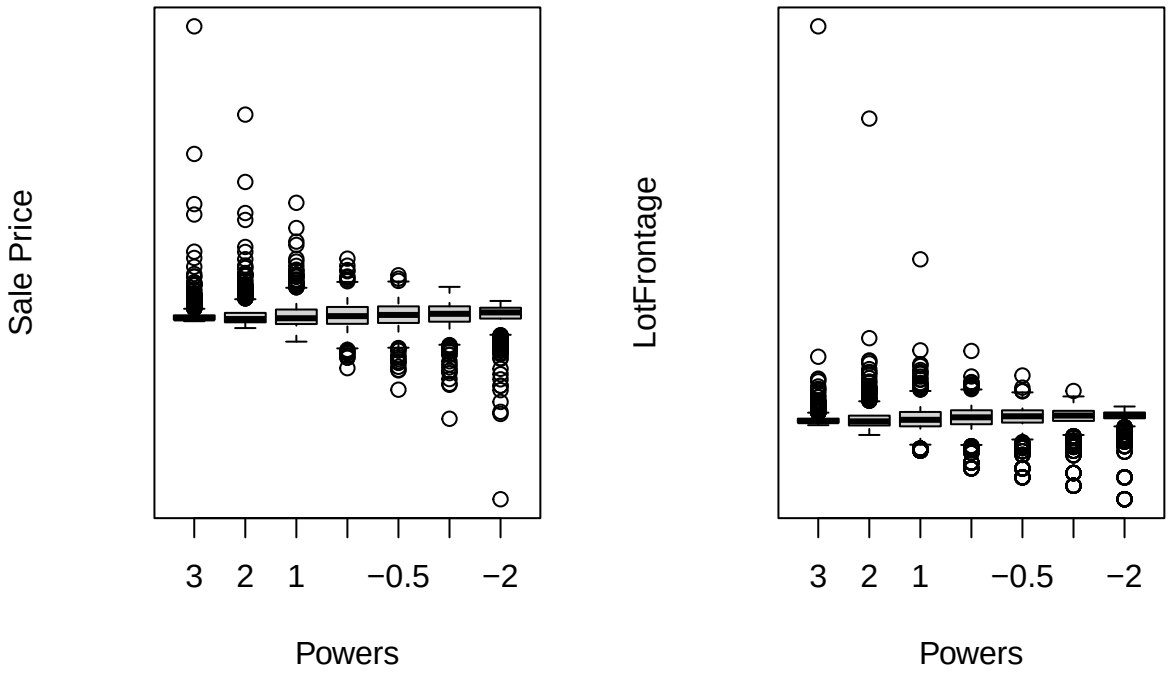
— Question 2: Transformations —

```
original <- tibble(read.csv('housingData.csv'))
housing <- tibble(read.csv('housingData.csv')) # Load data into a tibble

########## (a) Identify Skewness and rectify ##########
```

```r
par(mfrow = c(1,2))
symbox(housing$SalePrice, data = housing, powers=c(3,2,1,0,-0.5,-1,-2),
       ylab = "Sale Price")
boxcox(housing$SalePrice)
```

```
##
## Results of Box-Cox Transformation
## ---------------------------------
##
## Objective Name:                  PPCC
##
## Data:                            housing$SalePrice
##
## Sample Size:                     1000
##
##  lambda  PPCC
##    -2.0 0.814
##    -1.5 0.897
##    -1.0 0.956
##    -0.5 0.989
##     0.0 0.996
##     0.5 0.979
##     1.0 0.936
##     1.5 0.864
##     2.0 0.766
```

```r
    # lambda = 0 (logarithmic transformation) reduces the skewness the most
    # for Sales Price

symbox(housing$LotFrontage, data = housing, powers=c(3,2,1,0,-0.5,-1,-2),
       ylab = "LotFrontage")
```




```
boxcox(housing$LotFrontage)
```

```
## Warning in is.not.finite.warning(x): There were 207 nonfinite values in x : 207
## NA's
```

```
## Warning in boxcox.default(housing$LotFrontage): 207 observations with
## NA/NaN/Inf in 'x' removed.
```

```
##
## Results of Box-Cox Transformation
## ---------------------------------
##
## Objective Name:                  PPCC
##
## Data:                            housing$LotFrontage
##
## Number NA/NaN/Inf's Removed:     207
##
## Sample Size:                     793
##
##  lambda  PPCC
##    -2.0 0.727
##    -1.5 0.797
##    -1.0 0.867
##    -0.5 0.927
```

```
##     0.0 0.967
##     0.5 0.978
##     1.0 0.946
##     1.5 0.861
##     2.0 0.723
```

```
    # lambda = 0.5 (square root) reduces the skewness the most for Lot Frontage

par(mfrow = c(2,2))
hist(housing$SalePrice, col = "red",
     main = "Before", xlab = "Sale Price")  # Visualize skewed variables
skewness(housing$SalePrice)                 # Skewness = 1.96 before
```

```
## [1] 1.96
```

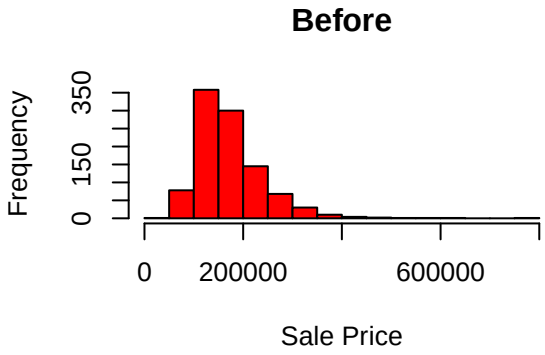
```
housing$SalePrice <- log(df$SalePrice + 1)
hist(housing$SalePrice, col = "blue",
     main = "After", xlab = "Sale Price")
skewness(housing$SalePrice)                 # Skewness = 0.146 after
```

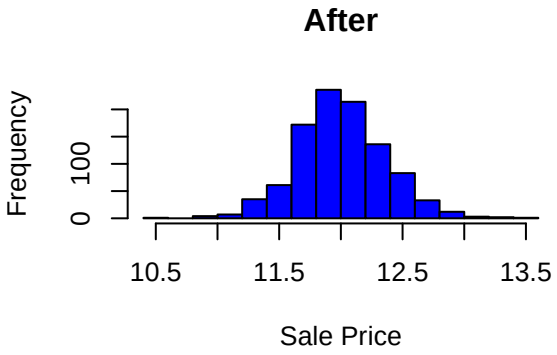
```
## [1] 0.146
```

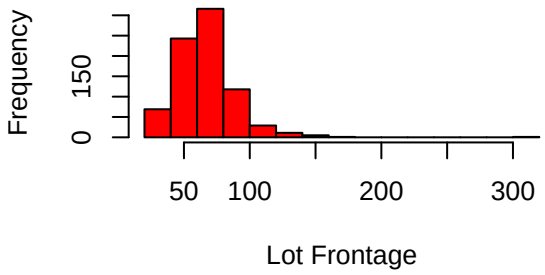
```
hist(housing$LotFrontage, col = "red", main = "", xlab = "Lot Frontage")
skewness(housing$LotFrontage, na.rm = TRUE) # Calculate skewness to verify
```

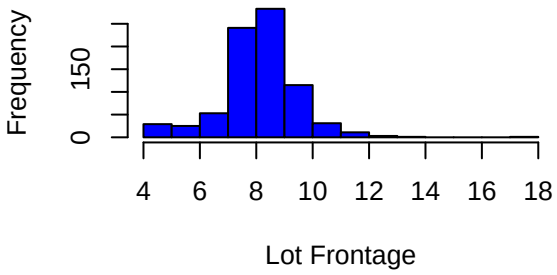
```
## [1] 1.91
```

```
                                            # Skewness = 1.91 before

housing$LotFrontage <- sqrt(housing$LotFrontage)
hist(housing$LotFrontage, col = "blue", main = "",
     xlab = "Lot Frontage")
```










```
skewness(housing$LotFrontage, na.rm = TRUE) # Skewness reduced to 0.246
```

```
## [1] 0.246
```

```
########## (b) Imputation of missing values ##########

  # i - Mean value imputation
housing$LotFrontage[is.na(housing$LotFrontage)] <- mean(housing$LotFrontage,
                                                        na.rm = TRUE)

      # Imputes missing values with a mean calculated excluding NA values

  # ii - Regression with error

dftwo <- tibble(read.csv('housingData.csv')) %>%
  select(where(is.numeric))

      # Categorical variables are not pre-processed, therefore LotFrontage
      # will be regressed on solely numerical variables

linear_model <- lm(LotFrontage ~ ., data = dftwo, na.action = na.omit)
missingvals <- dftwo[is.na(dftwo$LotFrontage),]
predicted <- predict(linear_model, newdata = missingvals)
resids <- residuals(linear_model)
imputation <- predicted + resids
```

```
## Warning in predicted + resids: longer object length is not a multiple of
## shorter object length
```

```
dftwo$LotFrontage[is.na(dftwo$LotFrontage)] <- imputation
```

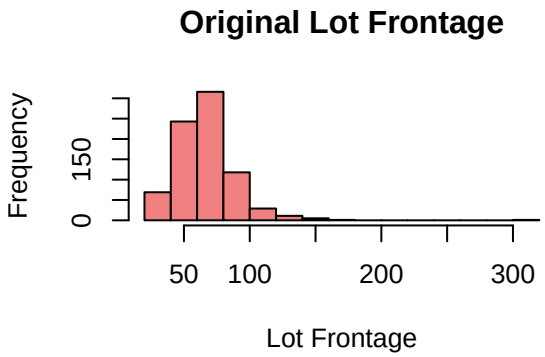
```
## Warning in dftwo$LotFrontage[is.na(dftwo$LotFrontage)] <- imputation: number of
## items to replace is not a multiple of replacement length
```

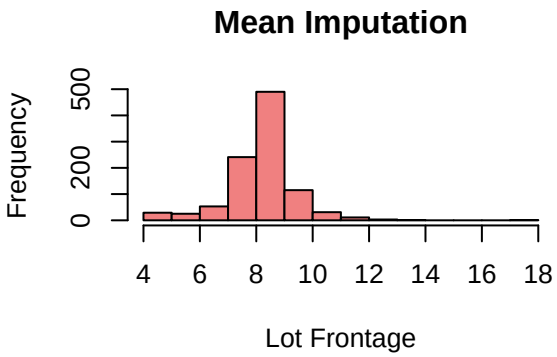
```
      # LotFrontage was regressed on all numeric variables, but to ensure
      # the ability for the residuals not to contain NA values, I set
      # na.action to na.omit rather than na.exclude. I then extracted the
      # predicted values, but only for the missing data within LotFrontage
      # to ensure non-missing data wasn't imputed. I used the residuals
      # to estimate error and added them to the predicted values, and then
      # imputed all missing values.

  # iii - Predictive Mean Matching

dfthree <- tibble(read.csv('housingData.csv'))

imputed <- mice(dfthree, method = 'pmm', m = 5, maxit = 10,
                seed = 500)
```

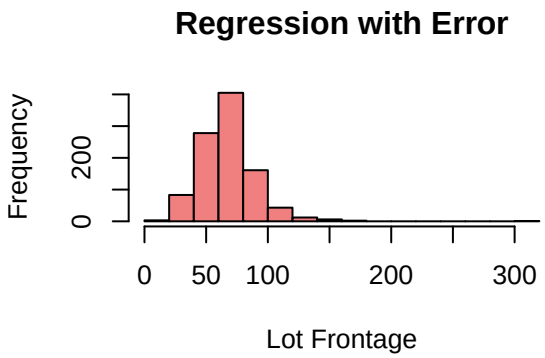
```
##
##  iter imp variable
##   1   1  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   1   2  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   1   3  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   1   4  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   1   5  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   2   1  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   2   2  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   2   3  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   2   4  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   2   5  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   3   1  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   3   2  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   3   3  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   3   4  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   3   5  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   4   1  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   4   2  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   4   3  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   4   4  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   4   5  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   5   1  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   5   2  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   5   3  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   5   4  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   5   5  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   6   1  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   6   2  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   6   3  LotFrontage*  MasVnrArea*  GarageYrBlt*
```

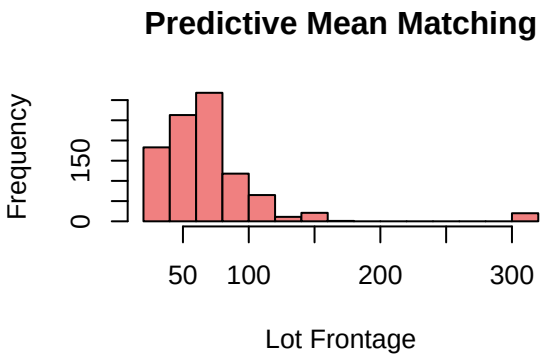
```
##   6   4  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   6   5  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   7   1  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   7   2  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   7   3  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   7   4  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   7   5  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   8   1  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   8   2  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   8   3  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   8   4  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   8   5  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   9   1  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   9   2  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   9   3  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   9   4  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   9   5  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   10   1  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   10   2  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   10   3  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   10   4  LotFrontage*  MasVnrArea*  GarageYrBlt*
##   10   5  LotFrontage*  MasVnrArea*  GarageYrBlt*

## Warning: Number of logged events: 338
```

```r
completedData <- complete(imputed, 3)

dfthree$LotFrontage[is.na(dfthree$LotFrontage)] <- completedData$LotFrontage[
                                                    is.na(dfthree$LotFrontage)]

        # This code was taken mostly from the mice site. We performed
        # predictive mean matching, created 5 different imputed datasets
        #

  # iv - Visualize transformations

par(mfrow = c(2,2))
hist(original$LotFrontage, col = "lightcoral", main = "Original Lot Frontage",
     xlab = "Lot Frontage")
hist(housing$LotFrontage, col = "lightcoral", main = "Mean Imputation",
     xlab = "Lot Frontage")
hist(dftwo$LotFrontage, col = "lightcoral", main = "Regression with Error",
     xlab = "Lot Frontage")
hist(dfthree$LotFrontage, col = "lightcoral", main = "Predictive Mean Matching",
     xlab = "Lot Frontage")
```







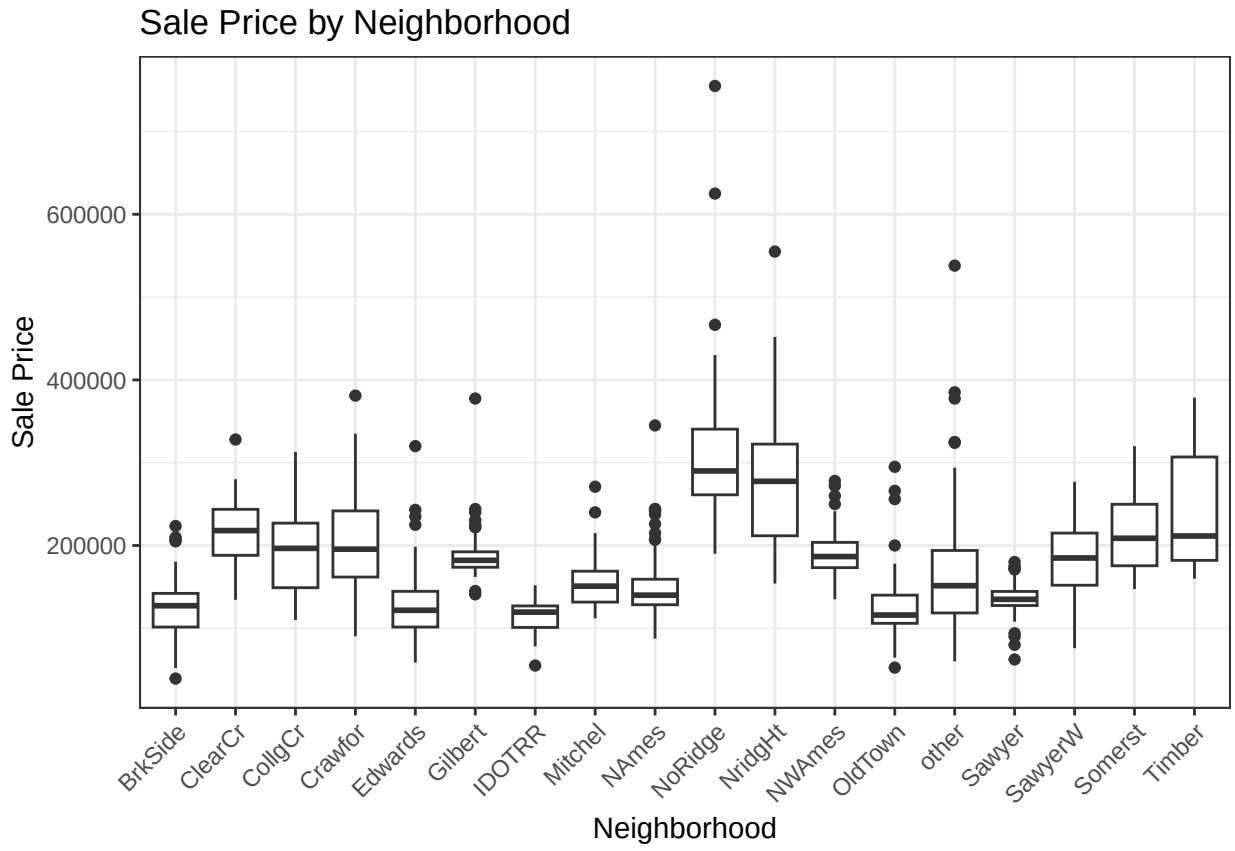



```
      # Interestingly, regression with error visually preserved the original
      # distribution the best. PMM may have been influenced by outliers.
      # However, it did also did well preserving the shape of the original
      # distribution. Mean imputation moved the distribution to the right, and
      # may have also been affected by outliers, as it was just a
      # mean calculation. Although, if we increase the maximum iterations
      # of PMM, we expect that the means will essentially converge to values
      # very close to the original values. maxit = 50 basically imitates
      # the original distribution.

########## (c) Categorical variable wrangling ##########

dfc <- tibble(read.csv('housingData.csv'))

dfc <- dfc %>%
  mutate(Exterior1st = fct_lump_n(Exterior1st, n = 4, other_level = "Other"))

levels(dfc$Exterior1st)
```

```
## [1] "HdBoard" "MetalSd" "VinylSd" "Wd Sdng" "Other"
```

```
      # Collapse all factor levels into 5 categories: HdBoard, MetalSd,
      # VinylSd, WdSdng, and Other to reduce dimensionality in the case
      # that we create binary dummy variables in pre-processing.
```

```
########## (d) More fun with factors ##########

    # i - Compute average sale price by neighborhood

avg_sale_byneighborhood <- original %>%
  group_by(Neighborhood) %>%
  summarize(avg_sale = mean(SalePrice, na.rm = TRUE))

    # ii - Parallel boxplots of Sale Price by Neighborhood

ggplot(data = original, aes(Neighborhood, SalePrice)) +
  geom_boxplot() +
  labs(title = "Sale Price by Neighborhood",
       x = "Neighborhood",
       y = "Sale Price") +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
    # iii - Re-order the factor levels
original <- original %>%
  mutate(Neighborhood = fct_reorder(Neighborhood,
                                    SalePrice,
                                    .fun = median,
                                    .desc = TRUE))
```

```r
    # iv - Parallel boxplots of reordered factor levels
ggplot(data = original, aes(Neighborhood, SalePrice)) +
  geom_boxplot() +
  labs(title = "Sale Price by Neighborhood (ordered by median)",
       x = "Neighborhood",
       y = "Sale Price") +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```