4.2 data transformation

2022-07-04

1. Use the apply function on a variable in your dataset

```
apply(X= housing, MARGIN=2, FUN = max)
```

```
##                Sale Date               Sale Price              sale_reason
##            "2016-12-16"               "4400000"                     "19"
##         sale_instrument             sale_warning                 sitetype
##                    "27"                       NA                     "R4"
##               addr_full                    zip5                  ctyname
##       "9985 185TH CT NE"                 "98074"                       NA
##              postalctyn                      lon                      lat
##               "REDMOND"             "-122.1643"              "47.73255"
##           building_grade square_feet_total_living                bedrooms
##                    "13"                 "13540"                     "11"
##          bath_full_count         bath_half_count         bath_3qtr_count
##                    "23"                     "8"                     "8"
##              year_built          year_renovated          current_zoning
##                  "2016"                 "2016"                  "URPSO"
##               sq_ft_lot               prop_type            present_use
##               "1631322"                    "R"                   "300"
```

2. Use the aggregate function on a variable in your dataset

```
# Continuous variable first, then categorical variable
aggregate(sq_ft_lot ~ ctyname, housing, mean)
```

```
##     ctyname sq_ft_lot
## 1   REDMOND  9000.644
## 2 SAMMAMISH 25475.455
```

3. Use the plyr function on a variable in your dataset – more specifically, I want to see you split some data, perform a modification to the data, and then bring it back together

```
sq_foot_average <- ddply(housing, summarise, .variables = c("ctyname"), sq_average <- mean(sq_ft_lot,na
sq_foot_average
```

```
##     ctyname sq_average <- mean(sq_ft_lot, na.rm = T)
## 1   REDMOND                                 9000.644
## 2 SAMMAMISH                                25475.455
## 3      <NA>                                36820.635
```

```
living_sq_foot_average <- ddply(housing, summarise, .variables = c("ctyname"), living_sq_average <- mean
living_sq_foot_average
```

```
##     ctyname living_sq_average <- mean(square_feet_total_living, na.rm = T)
## 1   REDMOND                                                       2461.493
## 2 SAMMAMISH                                                       3788.182
## 3      <NA>                                                       2612.213
```

```
combined_sq_foot <- bind_cols(sq_foot_average,living_sq_foot_average)
```

```
## New names:
## * `ctyname` -> `ctyname...1`
## * `ctyname` -> `ctyname...3`
```

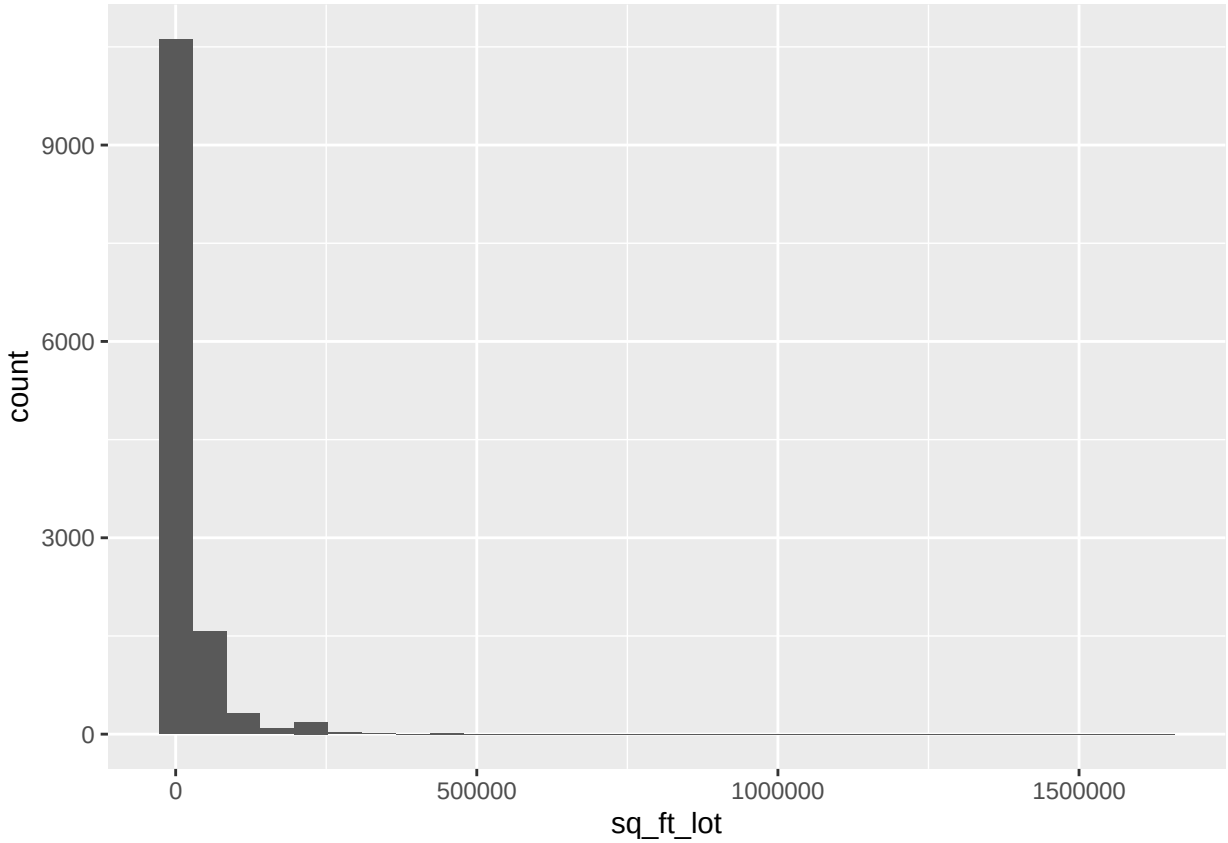
```
combined_sq_foot
```

```
##   ctyname...1 sq_average <- mean(sq_ft_lot, na.rm = T) ctyname...3
## 1     REDMOND                                 9000.644     REDMOND
## 2   SAMMAMISH                                25475.455   SAMMAMISH
## 3        <NA>                                36820.635        <NA>
##   living_sq_average <- mean(square_feet_total_living, na.rm = T)
## 1                                                       2461.493
## 2                                                       3788.182
## 3                                                       2612.213
```

4. Check distributions of the data sqare foot is positive skewed. Most of the data resides at the beginning of the distribution

```
ggplot(housing, aes(x = sq_ft_lot) ) + geom_histogram()
```

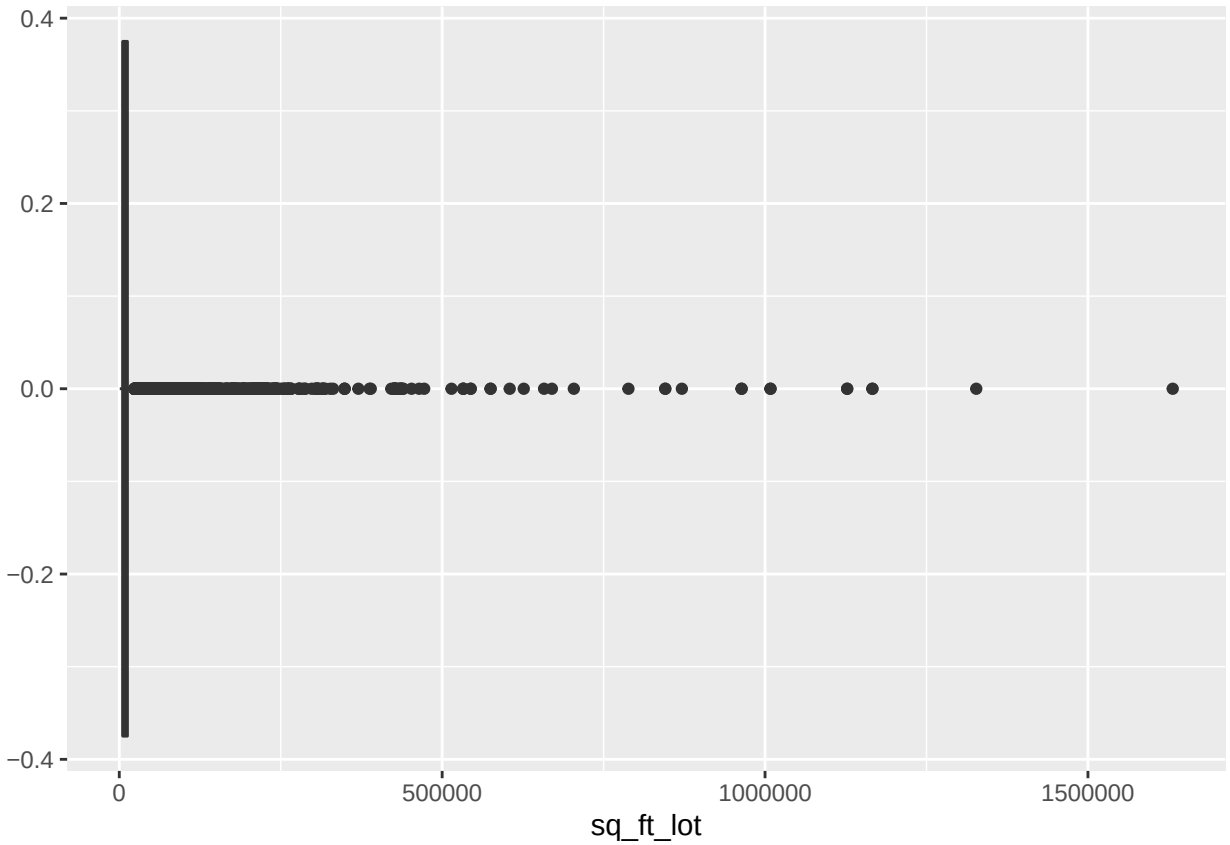
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



5. Identify if there are any outliers

Yes outliers do exist in this date for sq_ft_lot. Some variables exceed 100,000 sqare feet!

```
ggplot(housing, aes(x = sq_ft_lot) ) + geom_boxplot()
```



6. Create at least 2 new variables

```
names(housing)[2] <- 'Sales_Price'
names(housing)[1] <- 'Sales_Date'
housing$Column_1 <- "Test_1"
housing$Column_2 <- "Test_2"
average_sales_price <- mean(housing$Sales_Price)
housing_column_names <- colnames(housing)

housing
```

```
## # A tibble: 12,865 x 26
##    Sales_Date          Sales_Price sale_reason sale_instrument sale_warning
##    <dttm>                    <dbl>       <dbl>           <dbl> <chr>
##  1 2006-01-03 00:00:00      698000           1               3 <NA>
##  2 2006-01-03 00:00:00      649990           1               3 <NA>
##  3 2006-01-03 00:00:00      572500           1               3 <NA>
##  4 2006-01-03 00:00:00      420000           1               3 <NA>
##  5 2006-01-03 00:00:00      369900           1               3 15
##  6 2006-01-03 00:00:00      184667           1              15 18 51
##  7 2006-01-04 00:00:00     1050000           1               3 <NA>
##  8 2006-01-04 00:00:00      875000           1               3 <NA>
##  9 2006-01-04 00:00:00      660000           1               3 <NA>
## 10 2006-01-04 00:00:00      650000           1               3 <NA>
## # ... with 12,855 more rows, and 21 more variables: sitetype <chr>,
```

```
## #   addr_full <chr>, zip5 <dbl>, ctyname <chr>, postalctyn <chr>, lon <dbl>,
## #   lat <dbl>, building_grade <dbl>, square_feet_total_living <dbl>,
## #   bedrooms <dbl>, bath_full_count <dbl>, bath_half_count <dbl>,
## #   bath_3qtr_count <dbl>, year_built <dbl>, year_renovated <dbl>,
## #   current_zoning <chr>, sq_ft_lot <dbl>, prop_type <chr>, present_use <dbl>,
## #   Column_1 <chr>, Column_2 <chr>
```