4.2Exercises.R

Rahul Rajeev

2023-01-03

```
# Assignment 1: Test Scores
# Name: Rajeev, Rahul
# Date: 2023-01-02

# libraries and minimal theme
library(ggplot2)
library(plyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

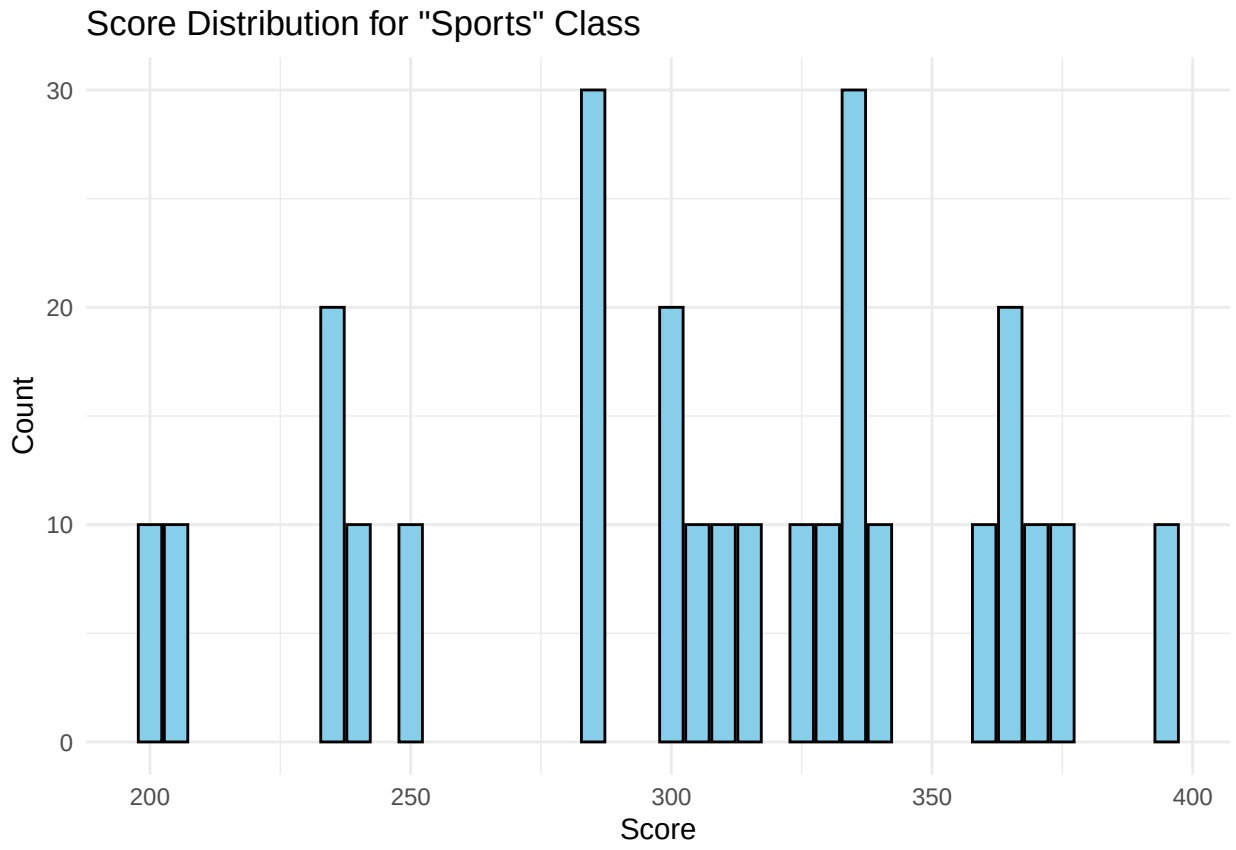
```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
theme_set(theme_minimal())

# loading scores dataset
setwd("C:/Users/rahul/Documents/Bellevue/DSC 520")

scores_df <- read.csv("data/scores.csv")

# 1. determining variables
str(scores_df)
```

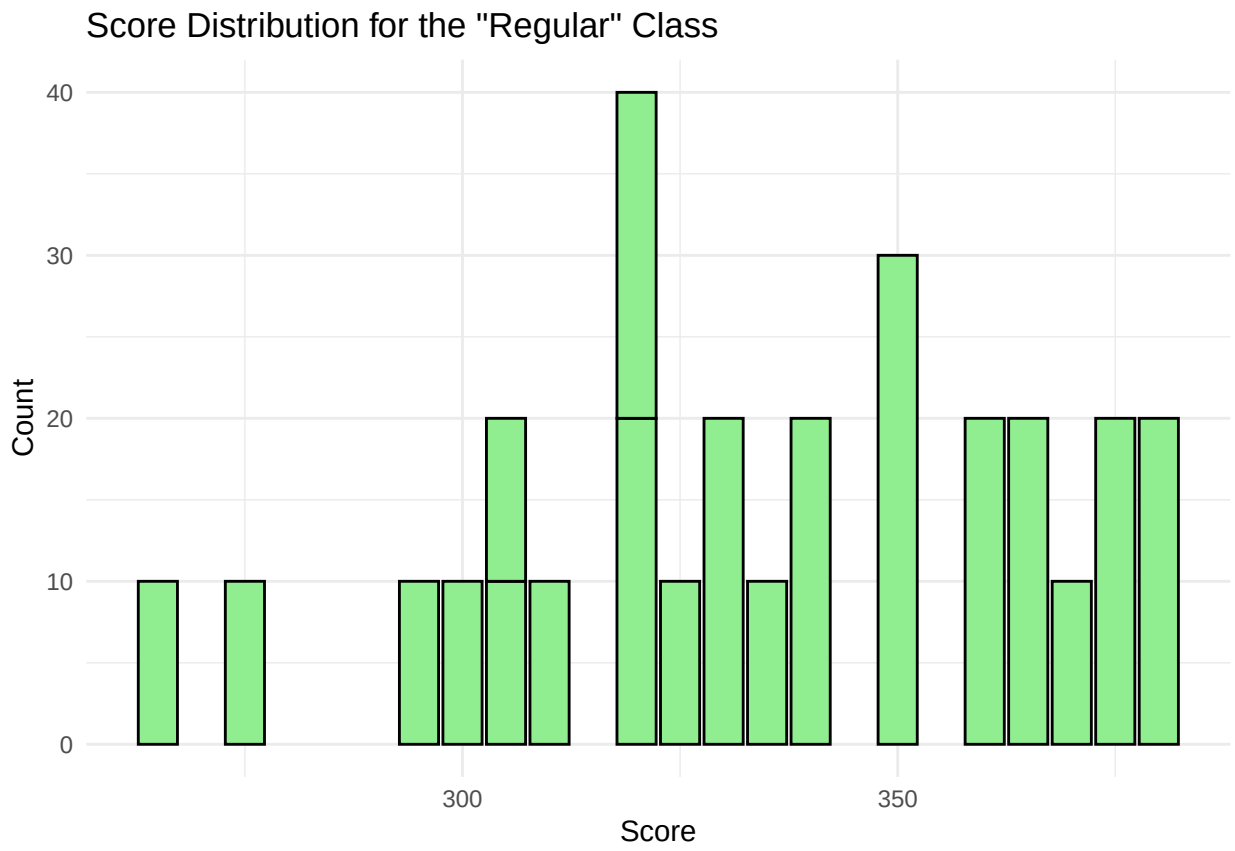
```
## 'data.frame':    38 obs. of  3 variables:
##  $ Count  : int  10 10 20 10 10 10 10 30 10 10 ...
##  $ Score  : int  200 205 235 240 250 265 275 285 295 300 ...
##  $ Section: chr  "Sports" "Sports" "Sports" "Sports" ...
```

```
# 2. There are three variables: count (numerical), score (numerical),
# and section (categorical, binary)
# Count and Score are both quantitative variables, and section is qualitative.
# The section specifies the grades in each course, the sports course vs.
# the non-sports or regular course. Both sections have different values of
```

```
# scores.

# 3. creating subsets for regular vs. sports sections
sports_df = scores_df[scores_df$Section == 'Sports',]
regular_df = scores_df[scores_df$Section == 'Regular',]

# 4. plot each data frame using ggplot and compare and contrast

# Sports class score distribution
ggplot(sports_df) + geom_bar(aes(x=Score, y=Count), stat="identity",
                             fill="skyblue", colour="black") +
  ggtitle('Score Distribution for "Sports" Class')
```




```
# Regular class score distribution
ggplot(regular_df) + geom_bar(aes(x=Score, y=Count), stat="identity",
                             fill="lightgreen", colour="black") +
  ggtitle('Score Distribution for the "Regular" Class')
```




```
# a. Comparing and contrasting the point distributions between the two section,
# looking at both tendency and consistency: Can you say that one section tended
# to score more points than the other? Justify and explain your answer.

# first let's calculate tendency and consistency as mean and standard deviation
# for both data sets

# sports tendency and consistency summary
sports_df %>%
  summarise(
    mean = mean(rep(Score,Count)),
    median = median(rep(Score,Count)),
    std = sd(rep(Score,Count)),
    min = min(rep(Score,Count)),
    max = max(rep(Score,Count)),
    students = sum(Count)
  )
```

```
##       mean median      std min max students
## 1 306.9231  312.5 52.63477 200 395      260
```

```
# regular tendency and consistency summary
regular_df %>%
  summarise(
    mean = mean(rep(Score,Count)),
    median = median(rep(Score,Count)),
    std = sd(rep(Score,Count)),
```

```
    min = min(rep(Score,Count)),
    max = max(rep(Score,Count)),
    students = sum(Count)
  )
```

```
##   mean median      std min max students
## 1  335    335 30.59389 265 380      290
```

```
# The class with regular applications tended to score higher than the class
# with sports applications, comparing 335 to about 310.
# The regular class also had a tighter spread of data with a standard deviation
# of around 31 while the sports class had a standard deviation of around 53.
# Students in the regular section tended to score higher than people in the
# sports section. Students in the sport section had the lowest scores, in the
# range of 200 - 265 in the sports section. The regular section however,
# had a smaller maximum score than the sports section. In addition,
# the median of the sports class was about 313 whereas the median of the regular
# section matched the mean of 335.

# b. Did every student in one section score more points than every student
# in the other section? If not, explain what a statistical tendency means
# in this context.

# No, some sports students scored more points than some regular students.
# The statistical tendency is defined by a student in the regular
# section section scored more than a student in the sports section.
# Since the data seems to overlap, it doesn't necessarily mean that students
# in one section scored more points than the other.

# c. What could be one additional variable that was not mentioned in the
# narrative that could be influencing the point distributions between the
# two sections?

# If students signed up for the sports section, but the professor used
# applications from sports the student is unfamiliar with, it might
# cause the student to score lower than just answering questions about
# regular applications. So knowledge of particular sports could be a variable.
# Although, since the section was advertised as a sports section, it could
# be dependent on the depth of the advertisement.

# Another variable that could be influencing the point distributions between
# the two sections could be the times at which the course was offered.
# Even though the class was advertised as a sports section, students who are
# following a particular academic plan may not have a choice in enrolling in
# one section or the other. This in turn could result in a lower range of
# scores.

#==============================================================================
# Assignment: Experimenting with Plyr on the Housing Dataset
# Name: Rajeev, Rahul
# Date: 2023-01-03

# importing libraries
library('readxl')
```

```
library(pastecs)
```

```
##
## Attaching package: 'pastecs'
```

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

```
library(scales)

# reading in xlsx file, then transferring the format into a data frame
housing_xl <- read_excel("data/week-7-housing.xlsx")
housing_df <- data.frame(housing_xl)

# just checking whether the data frame loaded properly
head(housing_df)
```

```
##    Sale.Date Sale.Price sale_reason sale_instrument sale_warning sitetype
## 1 2006-01-03     698000           1               3         <NA>       R1
## 2 2006-01-03     649990           1               3         <NA>       R1
## 3 2006-01-03     572500           1               3         <NA>       R1
## 4 2006-01-03     420000           1               3         <NA>       R1
## 5 2006-01-03     369900           1               3           15       R1
## 6 2006-01-03     184667           1              15        18 51       R1
##            addr_full  zip5 ctyname postalctyn       lon      lat building_grade
## 1  17021 NE 113TH CT 98052 REDMOND    REDMOND -122.1124 47.70139              9
## 2  11927 178TH PL NE 98052 REDMOND    REDMOND -122.1022 47.70731              9
## 3 13315 174TH AVE NE 98052    <NA>    REDMOND -122.1085 47.71986              8
## 4  3303 178TH AVE NE 98052 REDMOND    REDMOND -122.1037 47.63914              8
## 5  16126 NE 108TH CT 98052 REDMOND    REDMOND -122.1242 47.69748              7
## 6   8101 229TH DR NE 98053    <NA>    REDMOND -122.0341 47.67545              7
##   square_feet_total_living bedrooms bath_full_count bath_half_count
## 1                     2810        4               2               1
## 2                     2880        4               2               0
## 3                     2770        4               1               1
## 4                     1620        3               1               0
## 5                     1440        3               1               0
## 6                     4160        4               2               1
##   bath_3qtr_count year_built year_renovated current_zoning sq_ft_lot prop_type
## 1               0       2003              0             R4      6635         R
## 2               1       2006              0             R4      5570         R
## 3               1       1987              0             R6      8444         R
## 4               1       1968              0             R4      9600         R
## 5               1       1980              0             R6      7526         R
## 6               1       2005              0          URPSO      7280         R
##   present_use
## 1           2
## 2           2
## 3           2
## 4           2
## 5           2
## 6           2
```

```r
# 1. Use the apply function on a variable in your dataset
# I used the apply function to find the min and max of sale price

apply(housing_df['Sale.Price'], MARGIN=2, FUN=min)
```

```
## Sale.Price
##        698
```

```r
apply(housing_df['Sale.Price'], MARGIN=2, FUN=max)
```

```
## Sale.Price
##    4400000
```

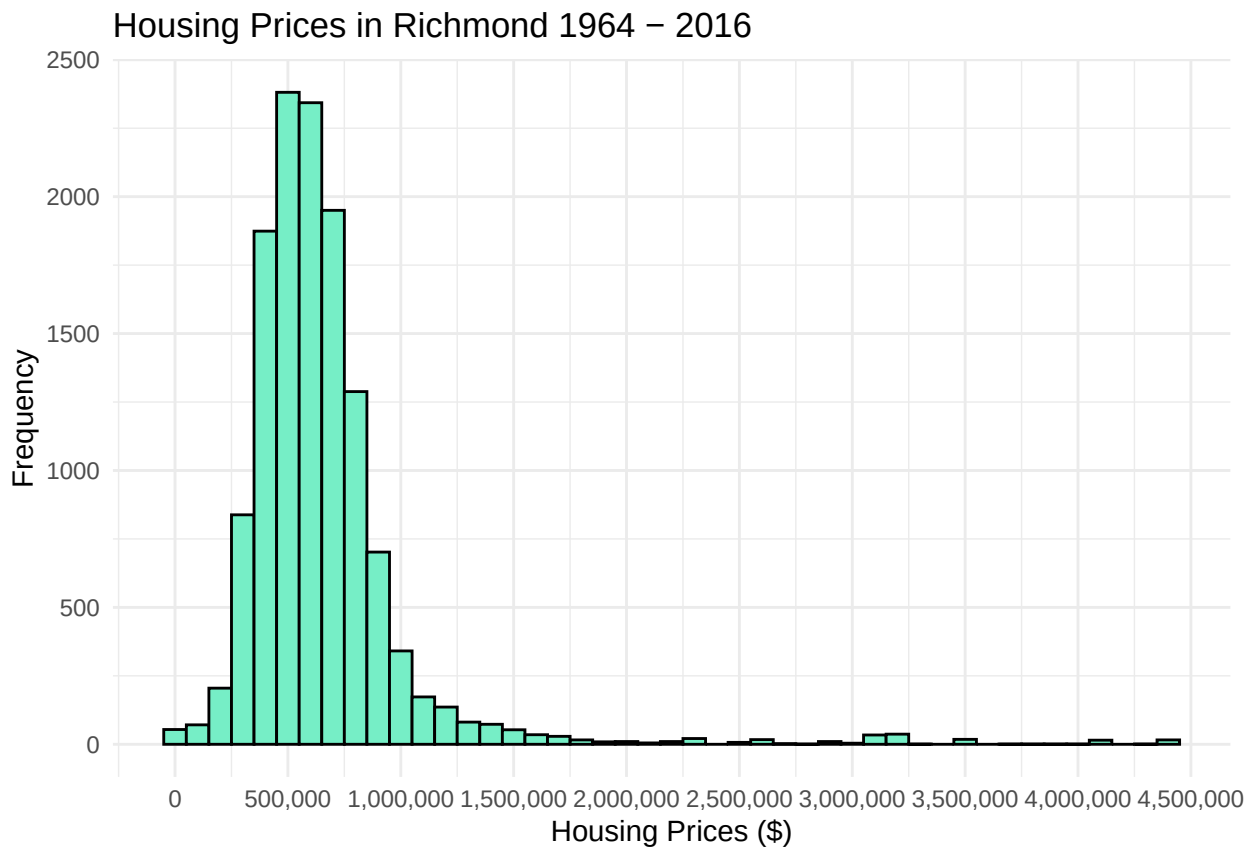
```r
# Apparently the minimum house price in this dataset was $698 dollars, which
# I believe is an error in the dataset, and can be corrected when working with
# the data. Apparently this particular house was actually sold for $2.25M
# in 2018. The maximum house price in this data set is $4.4M.

# 2. Use the aggregate function on a variable in your dataset
aggregate(housing_df$Sale.Price, by = list(housing_df$zip5),
          FUN = mean)
```

```
##   Group.1        x
## 1   98052 649375.4
## 2   98053 672623.7
## 3   98059 645000.0
## 4   98074 951543.8
```

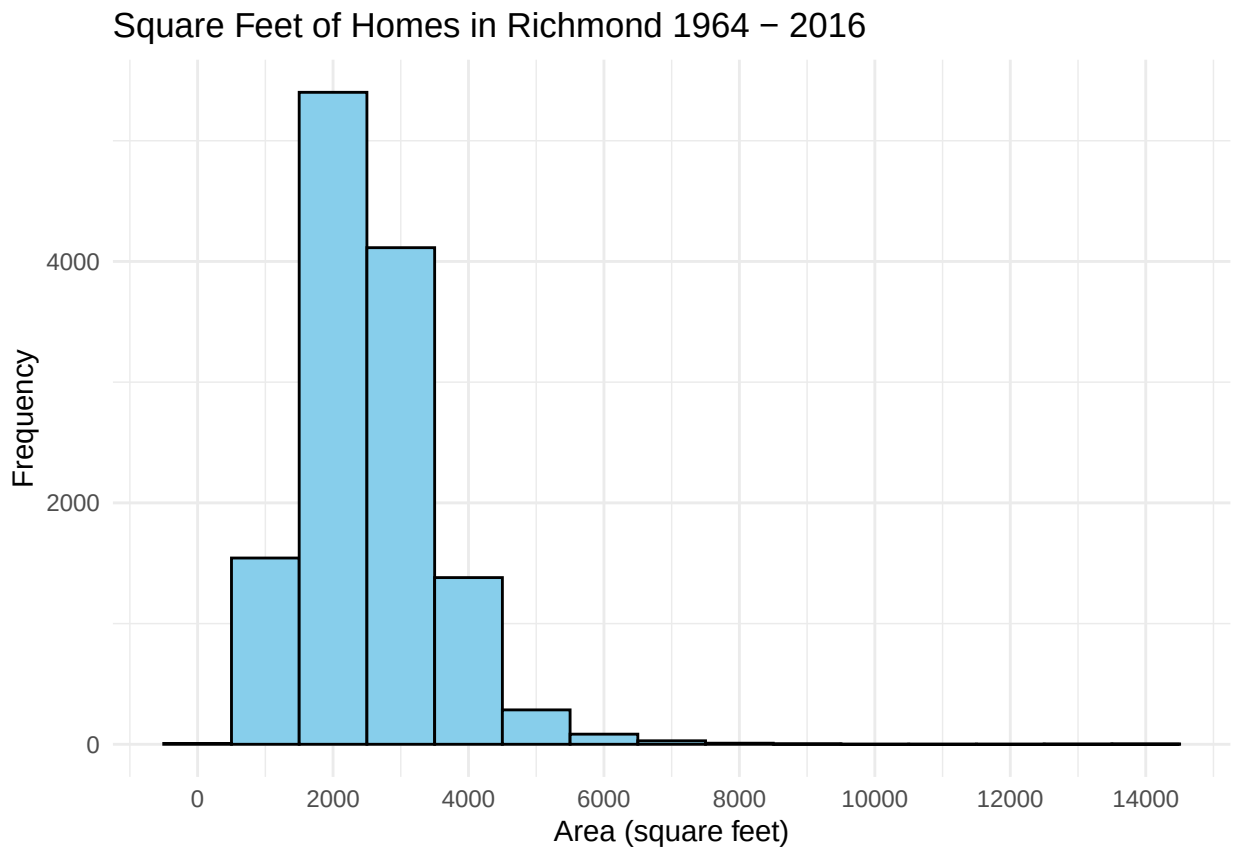
```r
# I used the aggregate function to find the mean sale price by zip code.

# 3. Use the plyr function on a variable in your dataset
ddply(housing_df, .(zip5), summarize,
      mean=round(mean(square_feet_total_living), 2),
      sd = round(sd(square_feet_total_living), 2))
```

```
##    zip5    mean      sd
## 1 98052 2498.84  878.65
## 2 98053 2580.32 1113.27
## 3 98059 4360.00      NA
## 4 98074 3681.51 1266.11
```

```r
# I used ddply to find the average square feet  and the standard deviation
# for houses in the data set, organized by zip code.

# 4. Check distributions of the data

# I will look at distributions of sale price and total square feet.
ggplot(housing_df, aes(x=Sale.Price)) +
  geom_histogram(fill="aquamarine2", colour="black",binwidth=100000, bins=10) +
  ggtitle('Housing Prices in Richmond 1964 - 2016') + ylab("Frequency") +
  xlab('Housing Prices ($)') +
  scale_x_continuous(breaks = scales::pretty_breaks(n = 10),
                     labels = label_comma())
```




```
# distribution of total square feet
ggplot(housing_df, aes(x=square_feet_total_living)) +
  geom_histogram(fill="skyblue", colour="black",binwidth=1000, bins=10) +
  ggtitle('Square Feet of Homes in Richmond 1964 - 2016') + ylab("Frequency") +
  xlab('Area (square feet)')+
  scale_x_continuous(breaks = scales::pretty_breaks(n = 10))
```




```
# Both distributions appear to be skew left and not normal.
# Housing prices peak around $600K to $700K.
# The square feet distribution peaks at about 2000-3000 square feet.
# The data is heavy tailed, sharp peaks and tails at each end.

# 5. Identify if there are any outliers
# For the housing prices distributions, the outliers appear to be houses priced
# higher than about $2.5M or higher. The points are spread out near the right
# end of the distribution. There are also houses in the $0 to $500k range
# that could be either errors in entry or very cheap houses. As discussed above,
# the minimum in the housing prices was $698 which is too low for a house.

# For the square feet distribution, there aren't any outstanding big outliers
# from the distributions, but there are a few houses that are within some
# questionable ranges in the bins. Performing a quick apply won't hurt:

apply(housing_df['square_feet_total_living'], MARGIN=2, FUN=min)
```

```
## square_feet_total_living
##                      240
```

```
apply(housing_df['square_feet_total_living'], MARGIN=2, FUN=max)
```

```
## square_feet_total_living
##                    13540
```

```
# We find that the minimum square feet in living is 240, and the maximum is
# 13450 square feet which are quite shocking. To live in 240 square feet could
```

```
# also be an error in the system, but 13450 could be a possibility of a very
# wealthy person.

# 6. Create at least 2 new variables

# created a price/sq foot living, price/sq foot lot, and total bath count
new_housing_df <- housing_df %>%
  mutate(price_per_sqft_living = Sale.Price/square_feet_total_living,
         price_per_sqft_lot = Sale.Price/sq_ft_lot,
         total_bath_count = bath_full_count + bath_half_count + bath_3qtr_count)

head(new_housing_df)
```

```
##    Sale.Date Sale.Price sale_reason sale_instrument sale_warning sitetype
## 1 2006-01-03     698000           1               3         <NA>       R1
## 2 2006-01-03     649990           1               3         <NA>       R1
## 3 2006-01-03     572500           1               3         <NA>       R1
## 4 2006-01-03     420000           1               3         <NA>       R1
## 5 2006-01-03     369900           1               3           15       R1
## 6 2006-01-03     184667           1              15        18 51       R1
##            addr_full  zip5 ctyname postalctyn       lon      lat building_grade
## 1  17021 NE 113TH CT 98052 REDMOND    REDMOND -122.1124 47.70139              9
## 2  11927 178TH PL NE 98052 REDMOND    REDMOND -122.1022 47.70731              9
## 3 13315 174TH AVE NE 98052    <NA>    REDMOND -122.1085 47.71986              8
## 4  3303 178TH AVE NE 98052 REDMOND    REDMOND -122.1037 47.63914              8
## 5  16126 NE 108TH CT 98052 REDMOND    REDMOND -122.1242 47.69748              7
## 6   8101 229TH DR NE 98053    <NA>    REDMOND -122.0341 47.67545              7
##   square_feet_total_living bedrooms bath_full_count bath_half_count
## 1                     2810        4               2               1
## 2                     2880        4               2               0
## 3                     2770        4               1               1
## 4                     1620        3               1               0
## 5                     1440        3               1               0
## 6                     4160        4               2               1
##   bath_3qtr_count year_built year_renovated current_zoning sq_ft_lot prop_type
## 1               0       2003              0             R4      6635         R
## 2               1       2006              0             R4      5570         R
## 3               1       1987              0             R6      8444         R
## 4               1       1968              0             R4      9600         R
## 5               1       1980              0             R6      7526         R
## 6               1       2005              0          URPSO      7280         R
##   present_use price_per_sqft_living price_per_sqft_lot total_bath_count
## 1           2             248.39858          105.19970               3
## 2           2             225.69097          116.69479               3
## 3           2             206.67870           67.79962               3
## 4           2             259.25926           43.75000               2
## 5           2             256.87500           49.14961               2
## 6           2              44.39111           25.36635               4
```

```
# Not really sure if the total_bath_count would be entirely important, but
# it seemed like another variable I could try creating.
```