Assignment: ASSIGNMENT 4

Name: Shekhar, Manish

Date: 2021-04-04

Read scores.csv

```
mydata <- read.csv("scores.csv")
head(mydata)
```

```
##   Count Score Section
## 1    10   200  Sports
## 2    10   205  Sports
## 3    20   235  Sports
## 4    10   240  Sports
## 5    10   250  Sports
## 6    10   265 Regular
```

```
mydata
```

```
##    Count Score Section
## 1     10   200  Sports
## 2     10   205  Sports
## 3     20   235  Sports
## 4     10   240  Sports
## 5     10   250  Sports
## 6     10   265 Regular
## 7     10   275 Regular
## 8     30   285  Sports
## 9     10   295 Regular
## 10    10   300 Regular
## 11    20   300  Sports
## 12    10   305  Sports
## 13    10   305 Regular
## 14    10   310 Regular
## 15    10   310  Sports
## 16    20   320 Regular
## 17    10   305 Regular
## 18    10   315  Sports
## 19    20   320 Regular
## 20    10   325 Regular
## 21    10   325  Sports
## 22    20   330 Regular
## 23    10   330  Sports
## 24    30   335  Sports
## 25    10   335 Regular
## 26    20   340 Regular
## 27    10   340  Sports
## 28    30   350 Regular
## 29    20   360 Regular
## 30    10   360  Sports
## 31    20   365 Regular
## 32    20   365  Sports
## 33    10   370  Sports
## 34    10   370 Regular
```

```
## 35    20   375 Regular
## 36    10   375  Sports
## 37    20   380 Regular
## 38    10   395  Sports
```

1. What are the observational units in this study?

```r
str(mydata)
```

```
## '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" ...
```

There are two observational units in this study - 1. Sectional score - Score obtained by students in the course (Sports section or Regular section) 2. Count of students - Count of students achieving above score

2. Identify the variables mentioned in the narrative paragraph and determine which are categorical and quantitative?

```r
str(mydata)
```

```
## '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" ...
```

```r
mydata$Section <- factor(mydata$Section,labels = c("Sports","Regular"))
summary(mydata)
```

```
##      Count           Score          Section
##  Min.   :10.00   Min.   :200.0   Sports :19
##  1st Qu.:10.00   1st Qu.:300.0   Regular:19
##  Median :10.00   Median :322.5
##  Mean   :14.47   Mean   :317.5
##  3rd Qu.:20.00   3rd Qu.:357.5
##  Max.   :30.00   Max.   :395.0
```

Count and Score are quantitative and Section is categorical Section can be changed to factor with two levels for better R interpretation

3. Create one variable to hold a subset of your data set that contains only the Regular Section and one variable for the Sports Section.

```r
mydata_Sports <- subset(mydata, mydata$Section == "Sports")
mydata_Regular <- subset(mydata, mydata$Section == "Regular")
head(mydata_Sports)
```

```
##    Count Score Section
## 6     10   265  Sports
## 7     10   275  Sports
## 9     10   295  Sports
## 10    10   300  Sports
## 13    10   305  Sports
## 14    10   310  Sports
```

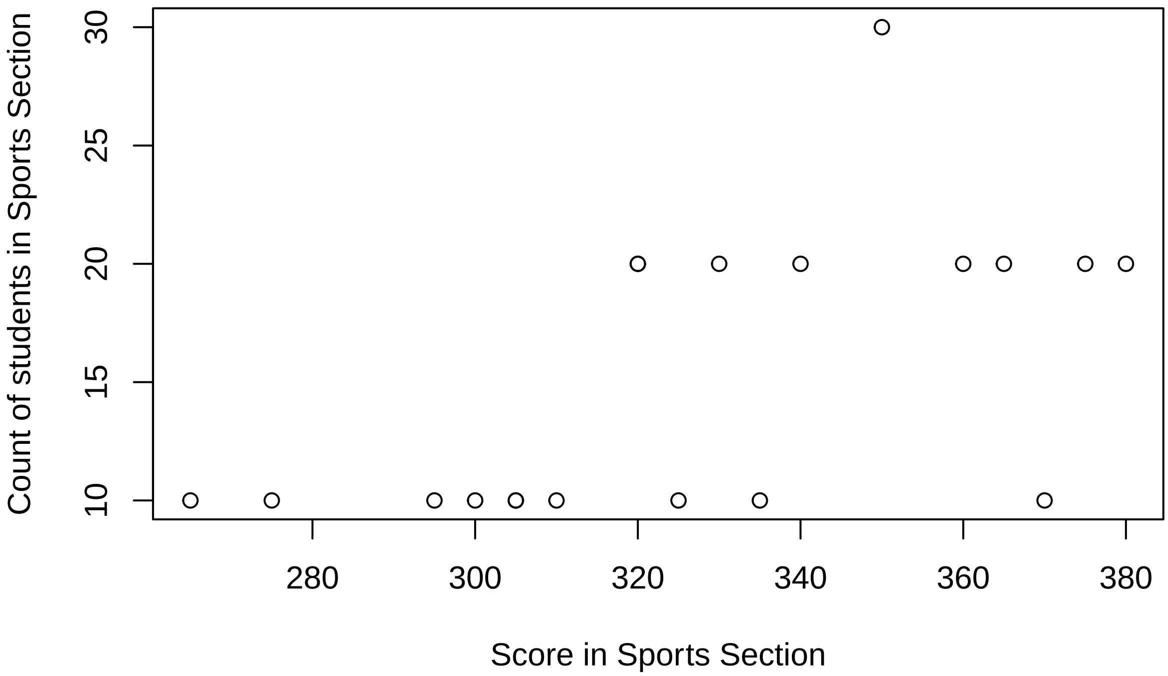
```r
head(mydata_Regular)
```

```
##   Count Score Section
## 1    10   200 Regular
```

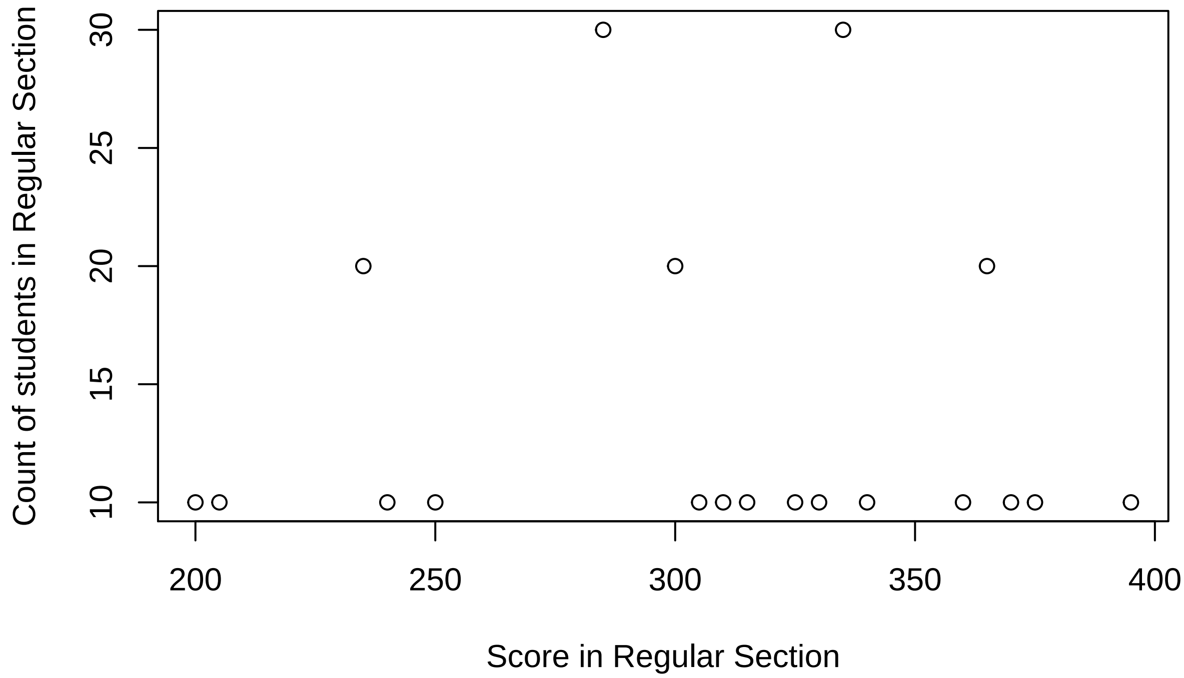
```
## 2     10   205 Regular
## 3     20   235 Regular
## 4     10   240 Regular
## 5     10   250 Regular
## 8     30   285 Regular
```

4. Use the Plot function to plot each Sections scores and the number of students achieving that score. Use additional Plot Arguments to label the graph and give each axis an appropriate label.

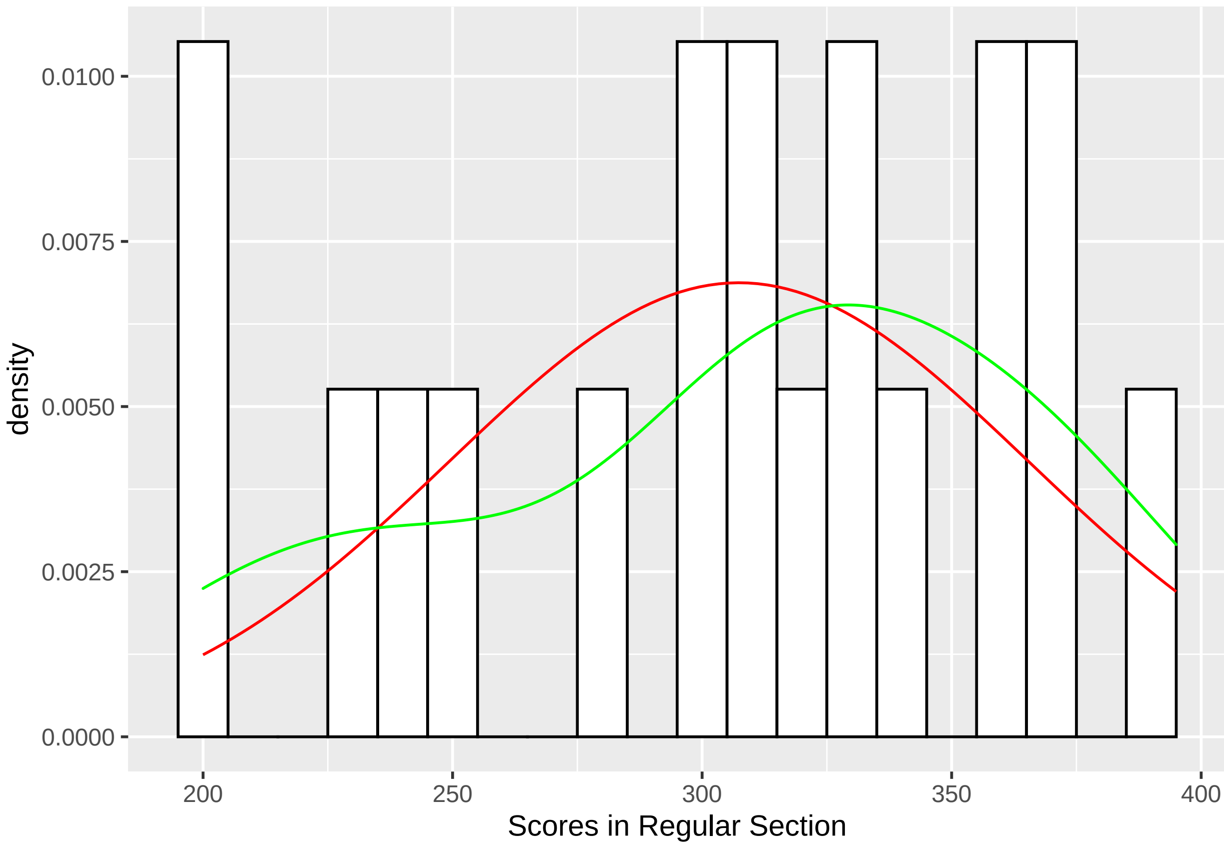
```
plot(mydata_Sports$Score, mydata_Sports$Count, type = "p", xlab = "Score in Sports Section", ylab = "Cou
```



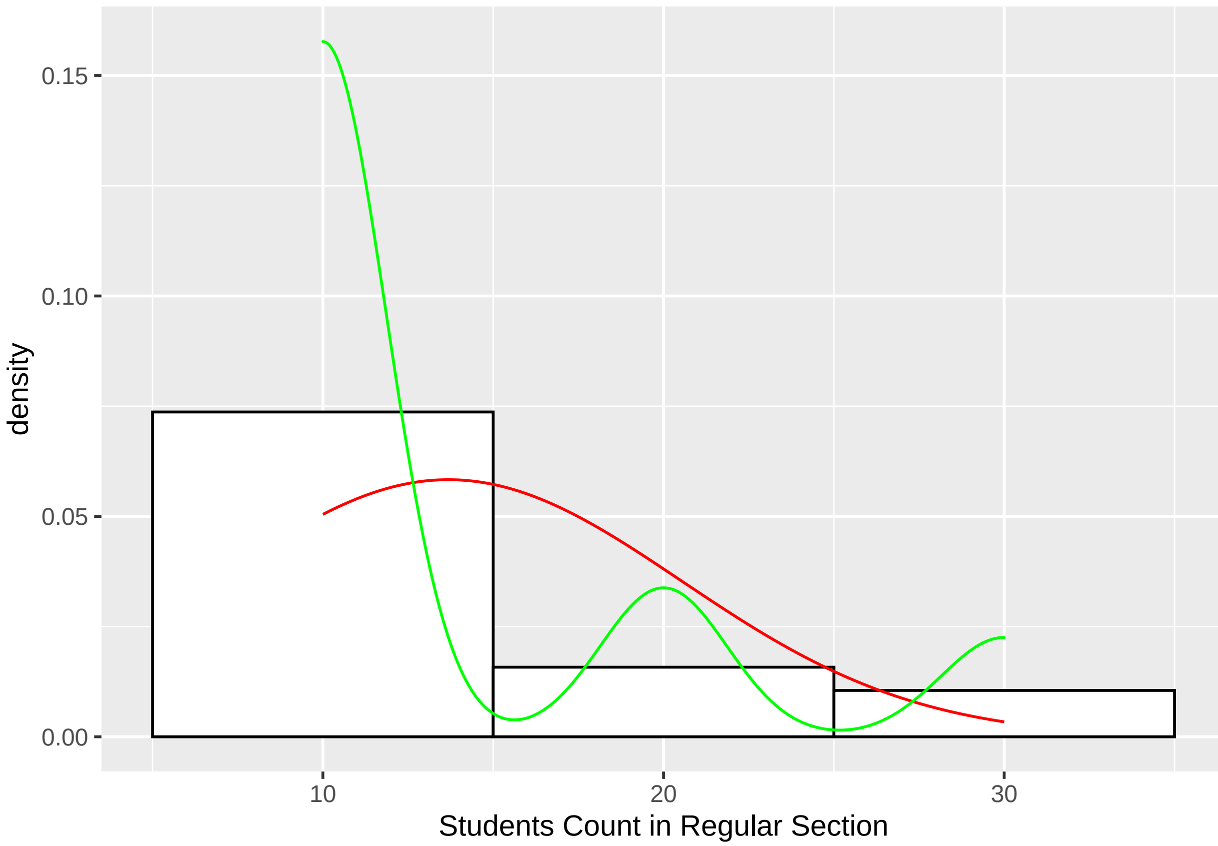

```
plot(mydata_Regular$Score, mydata_Regular$Count, type = "p", xlab = "Score in Regular Section", ylab =

# install.packages("ggplot2")
library(ggplot2)
```



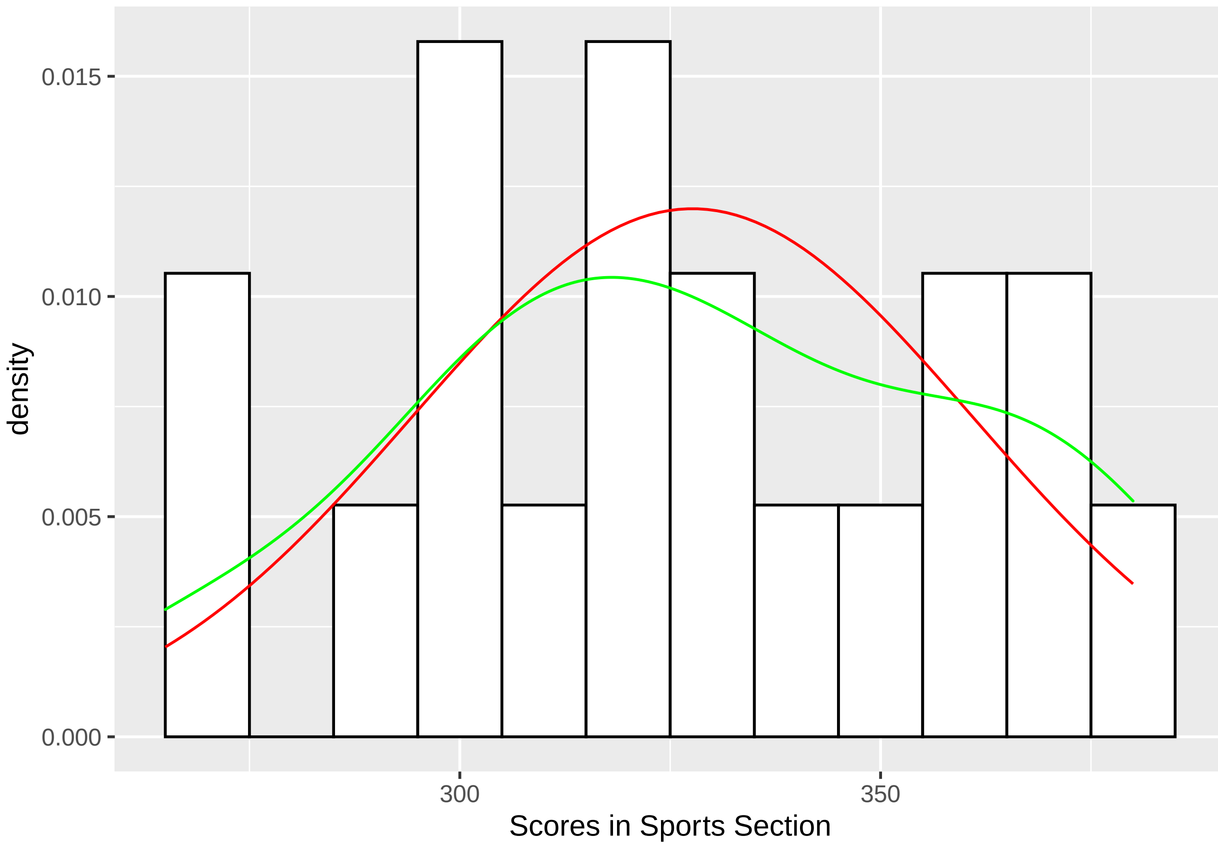

```r
# using ggplot to plot histograms for each numerical variable and their corresponding normal curves
# Plot histogram of Score from Regular section
ggplot(mydata_Regular, aes(x=Score)) +
    geom_histogram(binwidth = 10,
                   color = "Black",
                   fill = "White",
                   aes(y=..density..)) +
    xlab("Scores in Regular Section") +
    stat_function(fun = dnorm,
                  color = "Red",
                  args = list(mean = mean(mydata_Regular$Score, na.rm = TRUE),
                                         sd = sd(mydata_Regular$Score, na.rm = TRUE
                                                 ))) +
    geom_density(color = "Green")
```




```r
# Plot histogram of Student counts from Regular section
ggplot(mydata_Regular, aes(x=Count)) +
    geom_histogram(binwidth = 10,
                   color = "Black",
                   fill = "White",
                   aes(y=..density..)) +
    xlab("Students Count in Regular Section") +
    stat_function(fun = dnorm,
                  color = "Red",
                  args = list(mean = mean(mydata_Regular$Count, na.rm = TRUE),
                                         sd = sd(mydata_Regular$Count, na.rm = TRUE
                                                 ))) +
    geom_density(color = "Green")
```




```r
# Plot histogram of Score from Sports section
ggplot(mydata_Sports, aes(x=Score)) +
    geom_histogram(binwidth = 10,
                   color = "Black",
                   fill = "White",
                   aes(y=..density..)) +
    xlab("Scores in Sports Section") +
    stat_function(fun = dnorm,
                  color = "Red",
                  args = list(mean = mean(mydata_Sports$Score, na.rm = TRUE),
                                         sd = sd(mydata_Sports$Score, na.rm = TRUE
                                                 ))) +
    geom_density(color = "Green")
```




```r
# Plot histogram of Student counts from Regular section
ggplot(mydata_Sports, aes(x=Count)) +
    geom_histogram(binwidth = 10,
                   color = "Black",
                   fill = "White",
                   aes(y=..density..)) +
    xlab("Student Count in Sports Section") +
    stat_function(fun = dnorm,
                  color = "Red",
                  args = list(mean = mean(mydata_Sports$Count, na.rm = TRUE),
                                         sd = sd(mydata_Regular$Count, na.rm = TRUE
                                                 ))) +
    geom_density(color = "Green")
```

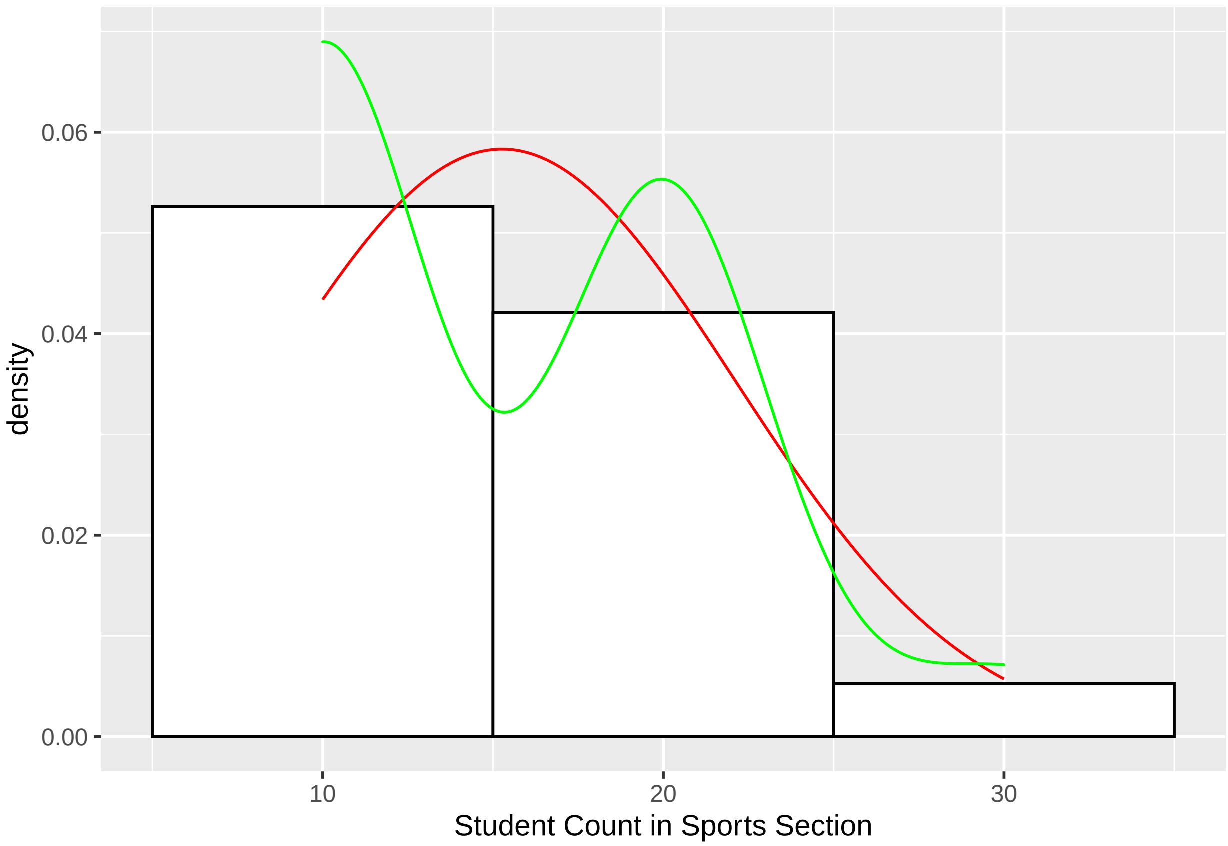


Chart 3 - Scores in Regular section appears to be negatively skewed a bit and has pretty much the same kurtosis as it's normal representation. Density is lower (0.065) at mean and around it and thus lesser observations should be near mean in comparison to Sports dataset (lesser area under curve in comparison to sports sample).

Chart 5 - Scores in Sports section appears to little platykurtic than it's normal representation. Density is higher (0.01) at mean and around it and thus more observations should be near mean (area under curve near mean should be more than same area in Regular chart - Chart 3) (higher area under curve in comparison to Regular sample).

4. a. Once you have produced your Plots answer the following questions: 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.

By simply looking at the two scatterplots, it seems Sports section has more number of students scoring higher marks say >= 300 in comparison to Regular section.

We can also calculate total score of all students in each section, only when they scored > 317.5 (which is mean of the score in whole data set/population) and compare them.

```
# Total score in Regular section considering scores > population score mean
total_score_Regular_grtr_mean <- sum(ifelse(mydata_Regular$Score>=317.5,mydata_Regular$Count*mydata_Regu
mydata_Sports
```

```
##    Count Score Section
## 6     10   265  Sports
## 7     10   275  Sports
## 9     10   295  Sports
## 10    10   300  Sports
```

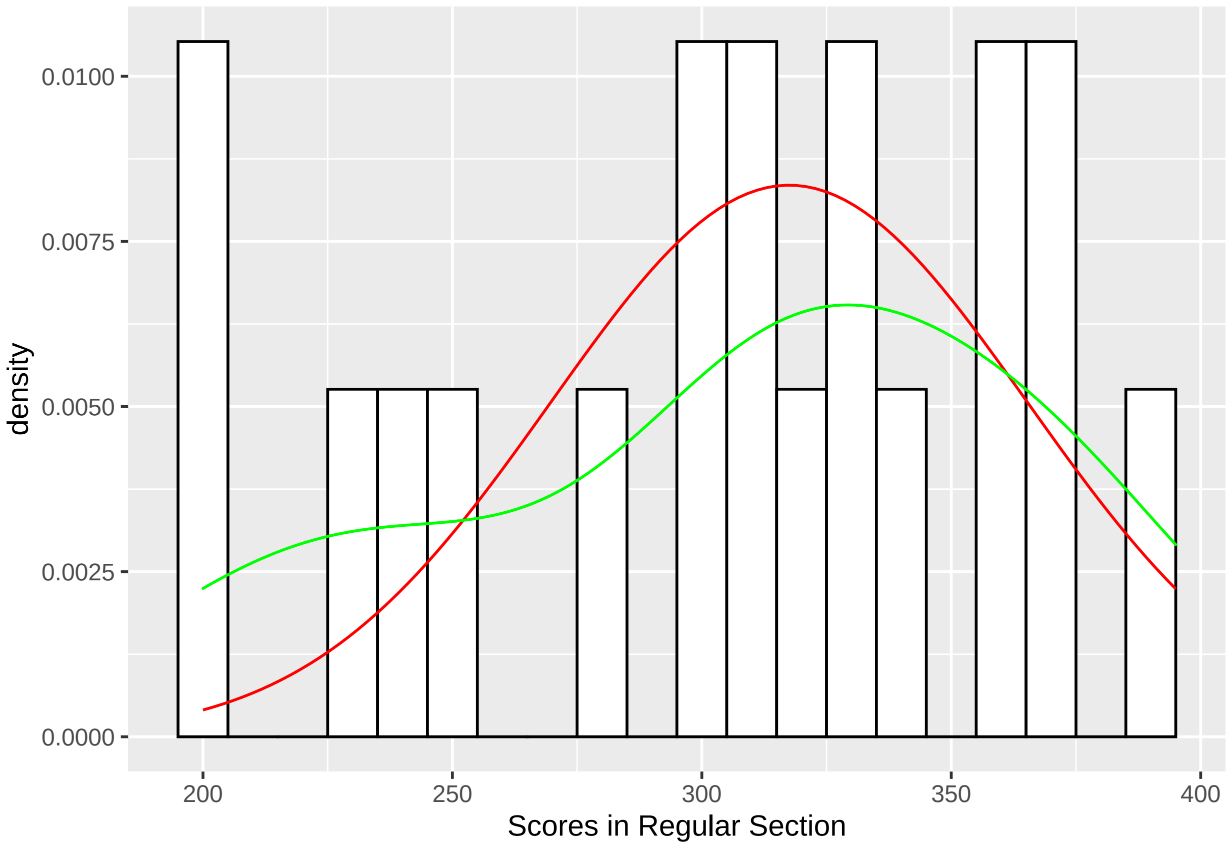
```
## 13    10   305  Sports
## 14    10   310  Sports
## 16    20   320  Sports
## 17    10   305  Sports
## 19    20   320  Sports
## 20    10   325  Sports
## 22    20   330  Sports
## 25    10   335  Sports
## 26    20   340  Sports
## 28    30   350  Sports
## 29    20   360  Sports
## 31    20   365  Sports
## 34    10   370  Sports
## 35    20   375  Sports
## 37    20   380  Sports
```

```
# Total score in Sports section considering scores > population score mean
total_score_Sports_grtr_mean <- sum(ifelse(mydata_Sports$Score>=317.5,mydata_Sports$Count*mydata_Sports$
# In which section do we see higher scores on an average
if(mean(total_score_Regular_grtr_mean) > mean(total_score_Sports_grtr_mean)){print("Students in Regular
```

```
## [1] "Students in Sports section are achieving more higher scores in general"
```

Also, lets look at the distribution of score in each section and compare it with score in overall course (population). This can be done by comparing density charts of individual section with normal density curve formed using mean and standard deviation of population (overall course).

```
# histogram and density chart of Regular sample (variable score) vs normal density chart of population
ggplot(mydata_Regular, aes(x=Score)) +
    geom_histogram(binwidth = 10,
                   color = "Black",
                   fill = "White",
                   aes(y=..density..)) +
    xlab("Scores in Regular Section") +
    stat_function(fun = dnorm,
                  color = "Red",
                  args = list(mean = mean(mydata$Score, na.rm = TRUE),
                                         sd = sd(mydata$Score, na.rm = TRUE
                                                ))) +
    geom_density(color = "Green")
```




```
# histogram and density chart of Sports sample (variable score) vs normal density chart of population (
ggplot(mydata_Sports, aes(x=Score)) +
    geom_histogram(binwidth = 10,
                   color = "Black",
                   fill = "White",
                   aes(y=..density..)) +
    xlab("Scores in Sports Section") +
    stat_function(fun = dnorm,
                  color = "Red",
                  args = list(mean = mean(mydata$Score, na.rm = TRUE),
                                         sd = sd(mydata$Score, na.rm = TRUE
                                                 ))) +
    geom_density(color = "Green")
```

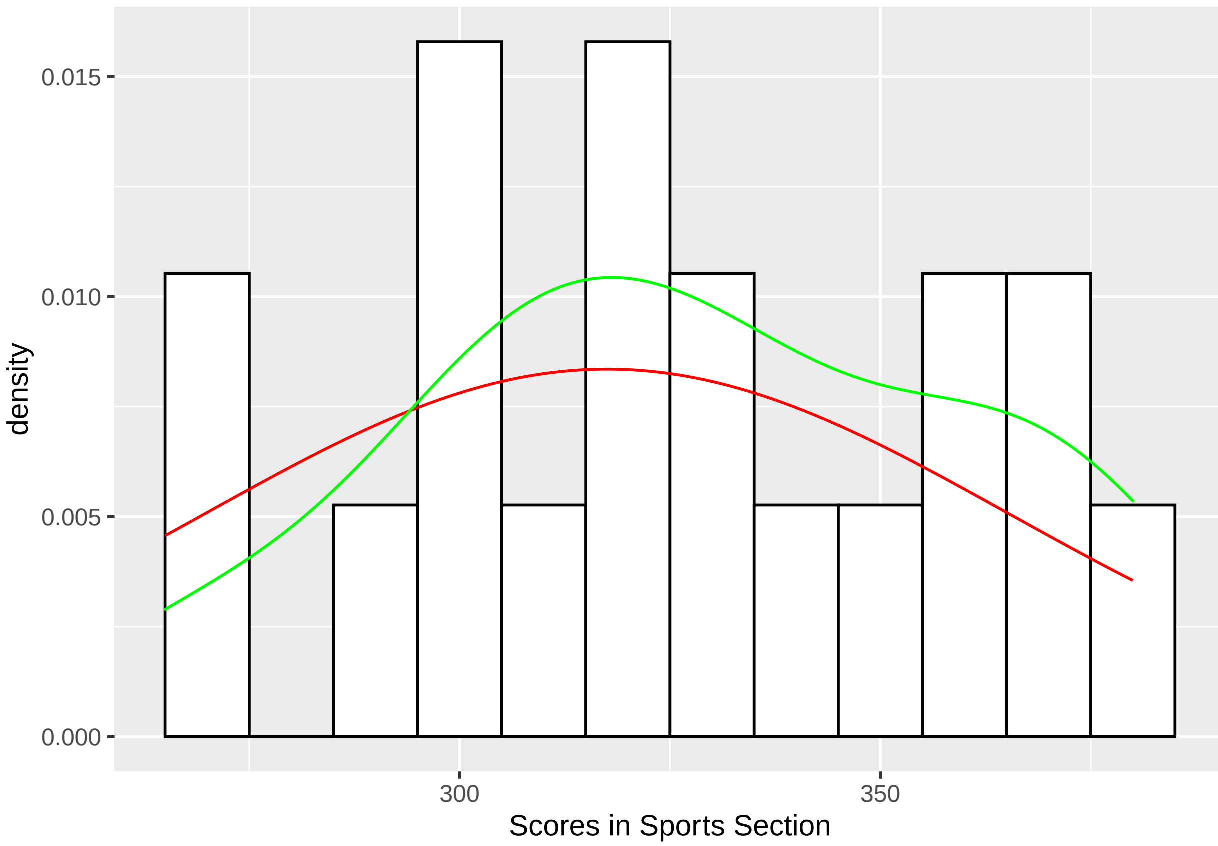


Chart 1 As we can see in the first plot above, kurtosis of scores in Regular Section (green) is lower than the kurtosis of the population (red) and in most cases it stays under the population curve which implies that most scores are lesser than that of population. Also, sample (Regular section - green curve) has negative skew (fatter tail on left side of the scale -> lower score).

Chart 2 As we can see in the second plot above, both mean and kurtosis seems higher in Sports Section (green) than that of the population (red). Thus, there are more observations near mean in Sports sample than in population data.

Density is higher near mean in Sports sample i.e. 0.01 in comparison to density of mean in Regular sample i.e. 0.006. Thus, there are more observations near mean in Sports sample.

Also, sample (Sports section - green curve) has positive skew (fatter tail on right side of the scale -> higher score).

Let's also look this numerically.

```
library(pastecs)
round(stat.desc(mydata[,1:2], basic = FALSE, norm = TRUE), digits = 3)
```

```
##                 Count    Score
## median         10.000  322.500
## mean           14.474  317.500
## SE.mean         1.046    7.750
## CI.mean.0.95    2.120   15.702
## var            41.607 2282.095
## std.dev         6.450   47.771
## coef.var        0.446    0.150
## skewness        1.072   -0.687
## skew.2SE        1.401   -0.897
```

```
## kurtosis     -0.048   -0.103
## kurt.2SE     -0.032   -0.068
## normtest.W    0.681    0.947
## normtest.p    0.000    0.072
```

```r
round(stat.desc(mydata_Regular[,1:2], basic = FALSE, norm = TRUE), digits = 3)
```

```
##               Count    Score
## median       10.000  315.000
## mean         13.684  307.368
## SE.mean       1.569   13.313
## CI.mean.0.95  3.297   27.970
## var          46.784 3367.690
## std.dev       6.840   58.032
## coef.var      0.500    0.189
## skewness      1.438   -0.419
## skew.2SE      1.373   -0.400
## kurtosis      0.585   -1.061
## kurt.2SE      0.288   -0.523
## normtest.W    0.593    0.945
## normtest.p    0.000    0.318
```

```r
round(stat.desc(mydata_Sports[,1:2], basic = FALSE, norm = TRUE), digits = 3)
```

```
##               Count    Score
## median       10.000  325.000
## mean         15.263  327.632
## SE.mean       1.404    7.632
## CI.mean.0.95  2.949   16.033
## var          37.427 1106.579
## std.dev       6.118   33.265
## coef.var      0.401    0.102
## skewness      0.596   -0.073
## skew.2SE      0.569   -0.070
## kurtosis     -0.788   -1.087
## kurt.2SE     -0.389   -0.536
## normtest.W    0.733    0.970
## normtest.p    0.000    0.767
```

Population mean (Score) = 317 Population standard deviation (Score) = 47

Sample Regular mean (Score) = 307 Sample Regular standard deviation (Score) = 58

Sample Sports mean (Score) = 328 Sample Sports standard deviation (Score) = 33

As, Sports sample's mean is higher than population mean it is more probable that a new student will end up scoring high if placed in this Section. Standard deviation is also lower than population which further supports above statement as data points are packed tighter.

Also, if we look at mean of student count. Sports section also seems to have more students (mean Sports > mean Population). So, students are preferring to join Sports section.

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

```r
# Total number of observation in population
str(mydata)
```

```
## '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: Factor w/ 2 levels "Sports","Regular": 2 2 2 2 2 1 1 2 1 1 ...
```

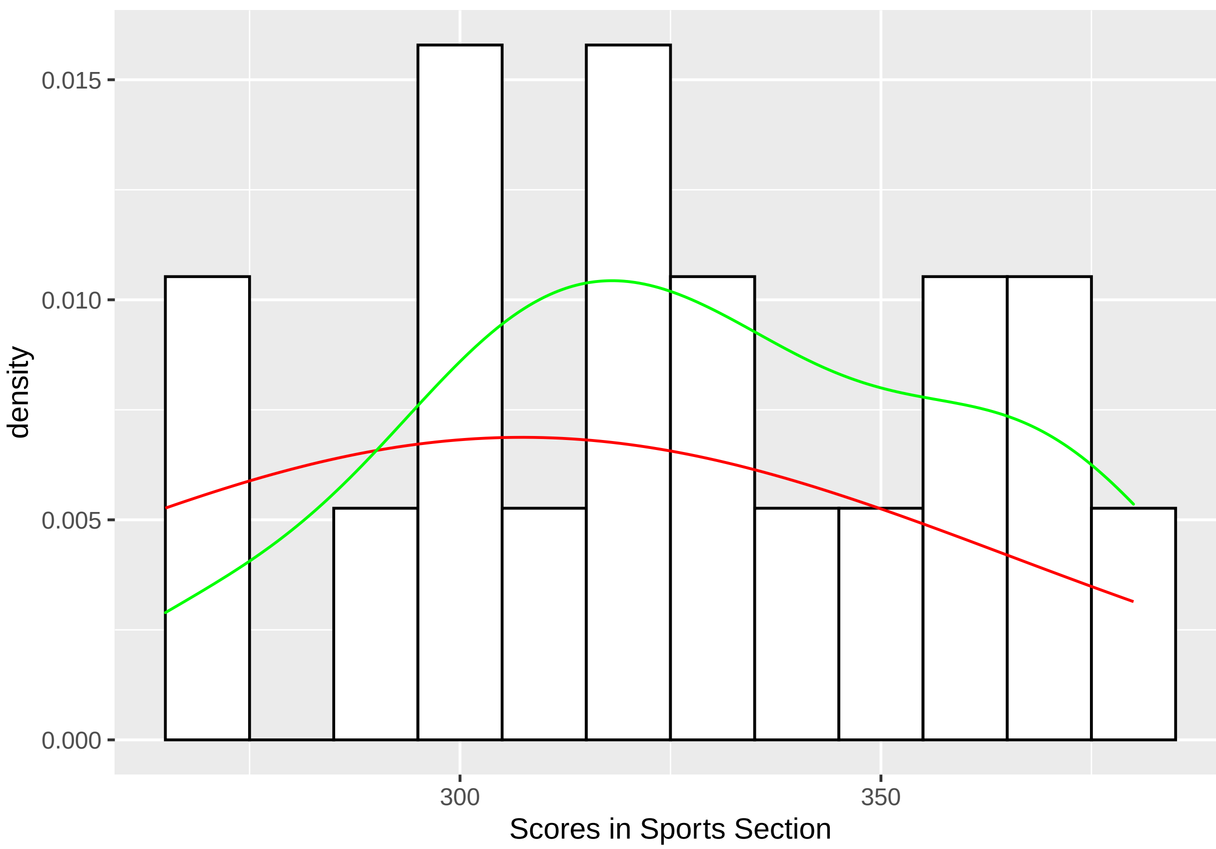
```
# Total number of students in Regular section
sum(mydata_Regular[,1])
```

```
## [1] 260
```

```
# Total number of students in Sports section
sum(mydata_Sports[,1])
```

```
## [1] 290
```

```
# Not every student in Sports section scored more than every student in Regular but majority of them di
ggplot(mydata_Sports, aes(x=Score)) +
    geom_histogram(binwidth = 10,
                   color = "Black",
                   fill = "White",
                   aes(y=..density..)) +
    xlab("Scores in Sports Section") +
    stat_function(fun = dnorm,
                  color = "Red",
                  args = list(mean = mean(mydata_Regular$Score, na.rm = TRUE),
                                         sd = sd(mydata_Regular$Score, na.rm = TRUE
                                                )))) +
    geom_density(color = "Green")
```



As we can see that two density plots intersect near left tail and some part of green chart (Sports) is below red chart (Regular) which means there are some observations or students in Regular section who scored more than few in Sports section (area under left side of the red curve will be little higher than that of green curve, but as % 0f observation towards tail are smaller this subset will be pretty small). But majority of students in Sports seems to have scored more than students in Regular section. Mean of scores in Sports section is greater than mean of scores in Regular. Also, kurtosis & density is higher in Sports curve (green) than in Regular curve (red), which suggests more students in Sports are concentrated near mean which is already higher than that of Regular section.

4. 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?

I think "age" or "gender" could be another variable that was not mentioned in the narrative that could be influencing the point distributions between the two sections.

2. a. Read housing data and use apply function on a variable in the dataset

```
library(readxl)
housing_data <- read_xlsx("week-6-housing.xlsx",col_names = TRUE,trim_ws = TRUE)
head(housing_data)
```

```
## # A tibble: 6 x 24
##   `Sale Date`         `Sale 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
## # ... with 19 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>
```

```
str(housing_data)
```

```
## tibble[,24] [12,865 x 24] (S3: tbl_df/tbl/data.frame)
##  $ Sale Date               : POSIXct[1:12865], format: "2006-01-03" "2006-01-03" ...
##  $ Sale Price              : num [1:12865] 698000 649990 572500 420000 369900 ...
##  $ sale_reason             : num [1:12865] 1 1 1 1 1 1 1 1 1 1 ...
##  $ sale_instrument         : num [1:12865] 3 3 3 3 3 15 3 3 3 3 ...
##  $ sale_warning            : chr [1:12865] NA NA NA NA ...
##  $ sitetype                : chr [1:12865] "R1" "R1" "R1" "R1" ...
##  $ addr_full               : chr [1:12865] "17021 NE 113TH CT" "11927 178TH PL NE" "13315 174TH AVE N
##  $ zip5                    : num [1:12865] 98052 98052 98052 98052 98052 ...
##  $ ctyname                 : chr [1:12865] "REDMOND" "REDMOND" NA "REDMOND" ...
##  $ postalctyn              : chr [1:12865] "REDMOND" "REDMOND" "REDMOND" "REDMOND" ...
##  $ lon                     : num [1:12865] -122 -122 -122 -122 -122 ...
##  $ lat                     : num [1:12865] 47.7 47.7 47.7 47.6 47.7 ...
##  $ building_grade          : num [1:12865] 9 9 8 8 7 7 10 10 9 8 ...
##  $ square_feet_total_living: num [1:12865] 2810 2880 2770 1620 1440 4160 3960 3720 4160 2760 ...
##  $ bedrooms                : num [1:12865] 4 4 4 3 3 4 5 4 4 4 ...
##  $ bath_full_count         : num [1:12865] 2 2 1 1 1 2 3 2 2 1 ...
##  $ bath_half_count         : num [1:12865] 1 0 1 0 0 1 0 1 1 0 ...
##  $ bath_3qtr_count         : num [1:12865] 0 1 1 1 1 1 1 0 1 1 ...
##  $ year_built              : num [1:12865] 2003 2006 1987 1968 1980 ...
```

```
##  $ year_renovated          : num [1:12865] 0 0 0 0 0 0 0 0 0 0 ...
##  $ current_zoning          : chr [1:12865] "R4" "R4" "R6" "R4" ...
##  $ sq_ft_lot               : num [1:12865] 6635 5570 8444 9600 7526 ...
##  $ prop_type               : chr [1:12865] "R" "R" "R" "R" ...
##  $ present_use             : num [1:12865] 2 2 2 2 2 2 2 2 2 2 ...
```

```r
# Rename columns with spaces to have underscore
colnames(housing_data)[1] <- "Sale_Date"
colnames(housing_data)[2] <- "Sale_Price"
str(housing_data)
```

```
## tibble[,24] [12,865 x 24] (S3: tbl_df/tbl/data.frame)
##  $ Sale_Date               : POSIXct[1:12865], format: "2006-01-03" "2006-01-03" ...
##  $ Sale_Price              : num [1:12865] 698000 649990 572500 420000 369900 ...
##  $ sale_reason             : num [1:12865] 1 1 1 1 1 1 1 1 1 1 ...
##  $ sale_instrument         : num [1:12865] 3 3 3 3 3 15 3 3 3 3 ...
##  $ sale_warning            : chr [1:12865] NA NA NA NA ...
##  $ sitetype                : chr [1:12865] "R1" "R1" "R1" "R1" ...
##  $ addr_full               : chr [1:12865] "17021 NE 113TH CT" "11927 178TH PL NE" "13315 174TH AVE N
##  $ zip5                    : num [1:12865] 98052 98052 98052 98052 98052 ...
##  $ ctyname                 : chr [1:12865] "REDMOND" "REDMOND" NA "REDMOND" ...
##  $ postalctyn              : chr [1:12865] "REDMOND" "REDMOND" "REDMOND" "REDMOND" ...
##  $ lon                     : num [1:12865] -122 -122 -122 -122 -122 ...
##  $ lat                     : num [1:12865] 47.7 47.7 47.7 47.6 47.7 ...
##  $ building_grade          : num [1:12865] 9 9 8 8 7 7 10 10 9 8 ...
##  $ square_feet_total_living: num [1:12865] 2810 2880 2770 1620 1440 4160 3960 3720 4160 2760 ...
##  $ bedrooms                : num [1:12865] 4 4 4 3 3 4 5 4 4 4 ...
##  $ bath_full_count         : num [1:12865] 2 2 1 1 1 2 3 2 2 1 ...
##  $ bath_half_count         : num [1:12865] 1 0 1 0 0 1 0 1 1 0 ...
##  $ bath_3qtr_count         : num [1:12865] 0 1 1 1 1 1 1 0 1 1 ...
##  $ year_built              : num [1:12865] 2003 2006 1987 1968 1980 ...
##  $ year_renovated          : num [1:12865] 0 0 0 0 0 0 0 0 0 0 ...
##  $ current_zoning          : chr [1:12865] "R4" "R4" "R6" "R4" ...
##  $ sq_ft_lot               : num [1:12865] 6635 5570 8444 9600 7526 ...
##  $ prop_type               : chr [1:12865] "R" "R" "R" "R" ...
##  $ present_use             : num [1:12865] 2 2 2 2 2 2 2 2 2 2 ...
```

```r
# get the mean sale price using apply function
mean_sale_price <- apply(housing_data[,2], MARGIN = 2, FUN = mean)
mean_sale_price
```

```
## Sale_Price
##   660737.7
```

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

```r
# use aggregate function to get mean sale price of houses by year built
aggregate(housing_data$Sale_Price, by = list(housing_data$year_built), FUN = mean)
```

```
##     Group.1        x
## 1      1900 394499.7
## 2      1903 430000.0
## 3      1905 620000.0
## 4      1906 550000.0
## 5      1909   1070.0
## 6      1910 150000.0
## 7      1912 619666.7
```

```
## 8       1913  457500.0
## 9       1914  835000.0
## 10      1915  228150.0
## 11      1916  350000.0
## 12      1918 1033833.3
## 13      1919  476800.0
## 14      1920  509083.3
## 15      1922  424587.5
## 16      1923  300000.0
## 17      1924  649500.0
## 18      1925  387250.0
## 19      1926  318333.3
## 20      1927 1173750.0
## 21      1928  520000.0
## 22      1929 1242500.0
## 23      1930  402191.7
## 24      1931  168828.5
## 25      1932  588146.2
## 26      1933  440500.0
## 27      1934  750000.0
## 28      1935 1616333.3
## 29      1936  485182.3
## 30      1937  846594.3
## 31      1938 1675500.0
## 32      1939  520000.0
## 33      1940  681411.1
## 34      1941  348517.2
## 35      1942  343561.0
## 36      1943  501200.0
## 37      1944  335626.5
## 38      1945  354330.9
## 39      1946  626875.0
## 40      1947  390378.7
## 41      1948  713522.6
## 42      1949  485525.4
## 43      1950  360315.0
## 44      1951  583972.0
## 45      1952  786191.7
## 46      1953  463553.7
## 47      1954  657591.3
## 48      1955  563706.3
## 49      1956  625561.5
## 50      1957  511411.5
## 51      1958  428233.8
## 52      1959  468616.6
## 53      1960  451005.4
## 54      1961  581580.0
## 55      1962  515826.5
## 56      1963  508518.7
## 57      1964  566355.5
## 58      1965  484418.3
## 59      1966  478482.7
## 60      1967  497566.3
## 61      1968  446930.1
```

```
## 62      1969  444439.2
## 63      1970  419788.3
## 64      1971  442688.5
## 65      1972  552177.1
## 66      1973  556947.5
## 67      1974  591669.8
## 68      1975  535944.1
## 69      1976  502248.9
## 70      1977  494102.5
## 71      1978  512763.1
## 72      1979  545454.4
## 73      1980  546471.3
## 74      1981  539075.9
## 75      1982  586006.0
## 76      1983  527091.5
## 77      1984  561059.2
## 78      1985  599990.3
## 79      1986  583642.8
## 80      1987  662669.3
## 81      1988  774747.3
## 82      1989  762350.0
## 83      1990  837696.4
## 84      1991  807708.3
## 85      1992  630408.5
## 86      1993  700939.1
## 87      1994  752529.6
## 88      1995  694532.9
## 89      1996  689408.3
## 90      1997  738764.9
## 91      1998  791991.1
## 92      1999 1016032.6
## 93      2000  829172.7
## 94      2001  695094.1
## 95      2002  599826.2
## 96      2003  645323.4
## 97      2004  632882.3
## 98      2005  647728.2
## 99      2006  692548.0
## 100     2007  664465.2
## 101     2008  866785.5
## 102     2009  756906.6
## 103     2010  649072.9
## 104     2011  677745.2
## 105     2012  922800.5
## 106     2013  912130.4
## 107     2014  825761.6
## 108     2015  888559.7
## 109     2016  893875.0
```

2. c. 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

```r
library(plyr)
# using ddply() split data by number of bedrooms and find mean Sale Price
ddply(housing_data, .(bedrooms), function(x) mean(x$Sale_Price))
```

```
##    bedrooms        V1
## 1         0  844059.5
## 2         1  722814.1
## 3         2  544946.4
## 4         3  564958.6
## 5         4  735910.0
## 6         5  836974.0
## 7         6  767494.3
## 8         7 1307281.7
## 9         8 1122500.0
## 10        9  581500.0
## 11       10  450000.0
## 12       11 1825000.0
```

2. d. Check distributions of the data We can check distributions of data by simply running stats.desc() on the data

```
str(housing_data)
```

```
## tibble[,24] [12,865 x 24] (S3: tbl_df/tbl/data.frame)
##  $ Sale_Date               : POSIXct[1:12865], format: "2006-01-03" "2006-01-03" ...
##  $ Sale_Price              : num [1:12865] 698000 649990 572500 420000 369900 ...
##  $ sale_reason             : num [1:12865] 1 1 1 1 1 1 1 1 1 1 ...
##  $ sale_instrument         : num [1:12865] 3 3 3 3 3 15 3 3 3 3 ...
##  $ sale_warning            : chr [1:12865] NA NA NA NA ...
##  $ sitetype                : chr [1:12865] "R1" "R1" "R1" "R1" ...
##  $ addr_full               : chr [1:12865] "17021 NE 113TH CT" "11927 178TH PL NE" "13315 174TH AVE N
##  $ zip5                    : num [1:12865] 98052 98052 98052 98052 98052 ...
##  $ ctyname                 : chr [1:12865] "REDMOND" "REDMOND" NA "REDMOND" ...
##  $ postalctyn              : chr [1:12865] "REDMOND" "REDMOND" "REDMOND" "REDMOND" ...
##  $ lon                     : num [1:12865] -122 -122 -122 -122 -122 ...
##  $ lat                     : num [1:12865] 47.7 47.7 47.7 47.6 47.7 ...
##  $ building_grade          : num [1:12865] 9 9 8 8 7 7 10 10 9 8 ...
##  $ square_feet_total_living: num [1:12865] 2810 2880 2770 1620 1440 4160 3960 3720 4160 2760 ...
##  $ bedrooms                : num [1:12865] 4 4 4 3 3 4 5 4 4 4 ...
##  $ bath_full_count         : num [1:12865] 2 2 1 1 1 2 3 2 2 1 ...
##  $ bath_half_count         : num [1:12865] 1 0 1 0 0 1 0 1 1 0 ...
##  $ bath_3qtr_count         : num [1:12865] 0 1 1 1 1 1 1 0 1 1 ...
##  $ year_built              : num [1:12865] 2003 2006 1987 1968 1980 ...
##  $ year_renovated          : num [1:12865] 0 0 0 0 0 0 0 0 0 0 ...
##  $ current_zoning          : chr [1:12865] "R4" "R4" "R6" "R4" ...
##  $ sq_ft_lot               : num [1:12865] 6635 5570 8444 9600 7526 ...
##  $ prop_type               : chr [1:12865] "R" "R" "R" "R" ...
##  $ present_use             : num [1:12865] 2 2 2 2 2 2 2 2 2 2 ...
```

```
summary(housing_data)
```

```
##    Sale_Date                     Sale_Price       sale_reason
##  Min.   :2006-01-03 00:00:00   Min.   :    698   Min.   : 0.00
##  1st Qu.:2008-07-07 00:00:00   1st Qu.: 460000   1st Qu.: 1.00
##  Median :2011-11-17 00:00:00   Median : 593000   Median : 1.00
##  Mean   :2011-07-28 15:07:32   Mean   : 660738   Mean   : 1.55
##  3rd Qu.:2014-06-05 00:00:00   3rd Qu.: 750000   3rd Qu.: 1.00
##  Max.   :2016-12-16 00:00:00   Max.   :4400000   Max.   :19.00
##  sale_instrument  sale_warning         sitetype          addr_full
##  Min.   : 0.000   Length:12865       Length:12865       Length:12865
```

```
##  1st Qu.: 3.000   Class :character   Class :character   Class :character
##  Median : 3.000   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 3.678
##  3rd Qu.: 3.000
##  Max.   :27.000
##       zip5         ctyname           postalctyn             lon
##  Min.   :98052   Length:12865       Length:12865       Min.   :-122.2
##  1st Qu.:98052   Class :character   Class :character   1st Qu.:-122.1
##  Median :98052   Mode  :character   Mode  :character   Median :-122.1
##  Mean   :98053                                         Mean   :-122.1
##  3rd Qu.:98053                                         3rd Qu.:-122.0
##  Max.   :98074                                         Max.   :-121.9
##       lat        building_grade  square_feet_total_living    bedrooms
##  Min.   :47.46   Min.   : 2.00   Min.   :  240            Min.   : 0.000
##  1st Qu.:47.67   1st Qu.: 8.00   1st Qu.: 1820            1st Qu.: 3.000
##  Median :47.69   Median : 8.00   Median : 2420            Median : 4.000
##  Mean   :47.68   Mean   : 8.24   Mean   : 2540            Mean   : 3.479
##  3rd Qu.:47.70   3rd Qu.: 9.00   3rd Qu.: 3110            3rd Qu.: 4.000
##  Max.   :47.73   Max.   :13.00   Max.   :13540            Max.   :11.000
##  bath_full_count  bath_half_count  bath_3qtr_count   year_built
##  Min.   : 0.000   Min.   :0.0000   Min.   :0.000   Min.   :1900
##  1st Qu.: 1.000   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:1979
##  Median : 2.000   Median :1.0000   Median :0.000   Median :1998
##  Mean   : 1.798   Mean   :0.6134   Mean   :0.494   Mean   :1993
##  3rd Qu.: 2.000   3rd Qu.:1.0000   3rd Qu.:1.000   3rd Qu.:2007
##  Max.   :23.000   Max.   :8.0000   Max.   :8.000   Max.   :2016
##  year_renovated    current_zoning       sq_ft_lot         prop_type
##  Min.   :   0.00   Length:12865       Min.   :    785   Length:12865
##  1st Qu.:   0.00   Class :character   1st Qu.:   5355   Class :character
##  Median :   0.00   Mode  :character   Median :   7965   Mode  :character
##  Mean   :  26.24                      Mean   :  22229
##  3rd Qu.:   0.00                      3rd Qu.:  12632
##  Max.   :2016.00                      Max.   :1631322
##   present_use
##  Min.   :  0.000
##  1st Qu.:  2.000
##  Median :  2.000
##  Mean   :  6.598
##  3rd Qu.:  2.000
##  Max.   :300.000
```

```
# By looking at summary output we know that data contains -
# 1. housing sales from 2006-01-03 till 2016-12-16
# 2. mean sale price is 660738
# 3. house size varies from 250 to 13,540 sq ft, with mean being 2540 sq ft
# 4. houses built in year 1900 to 2016. So we do have quite old houses
# 5. lot sq ft range
head(housing_data)
```

```
## # A tibble: 6 x 24
##   Sale_Date           Sale_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
## # ... with 19 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>
```

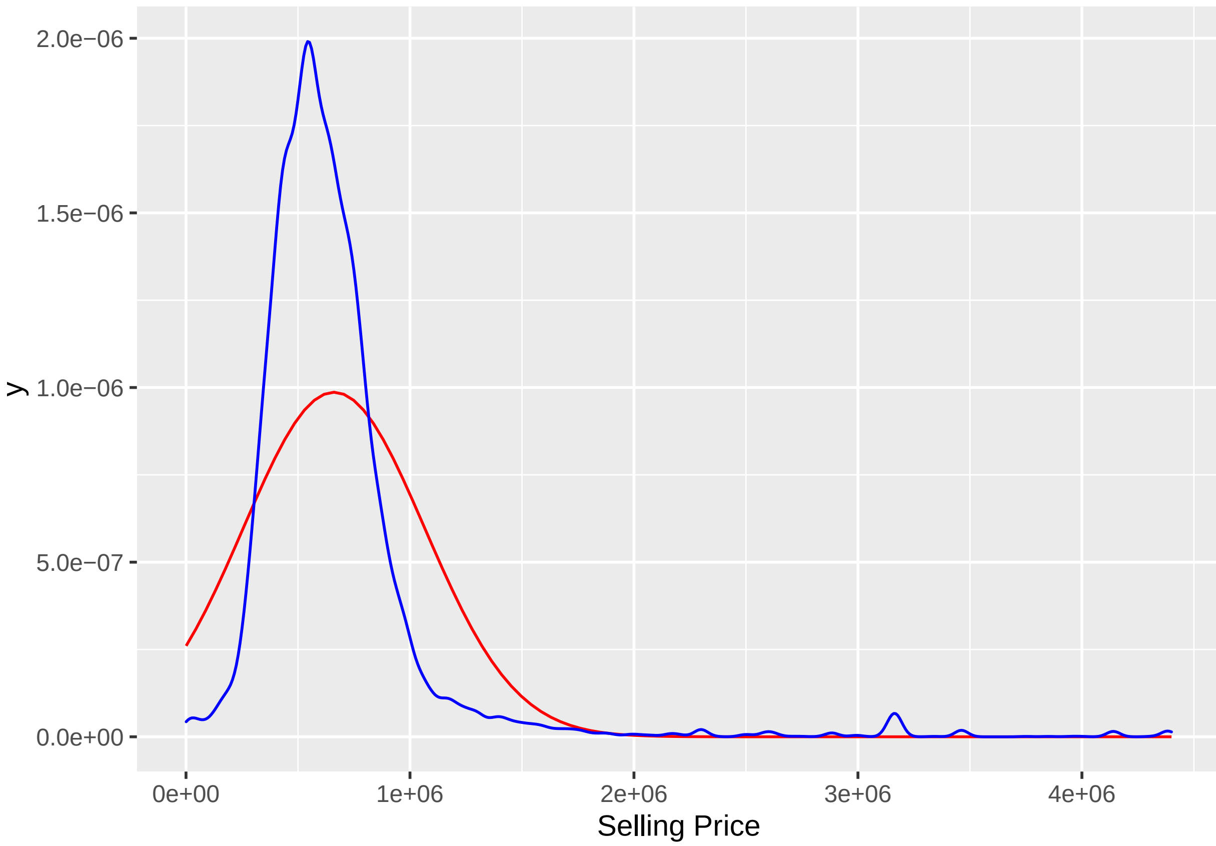
```r
# 6. check unique values of city in the data
unique(housing_data$ctyname)
```

```
## [1] "REDMOND"   NA          "SAMMAMISH"
```

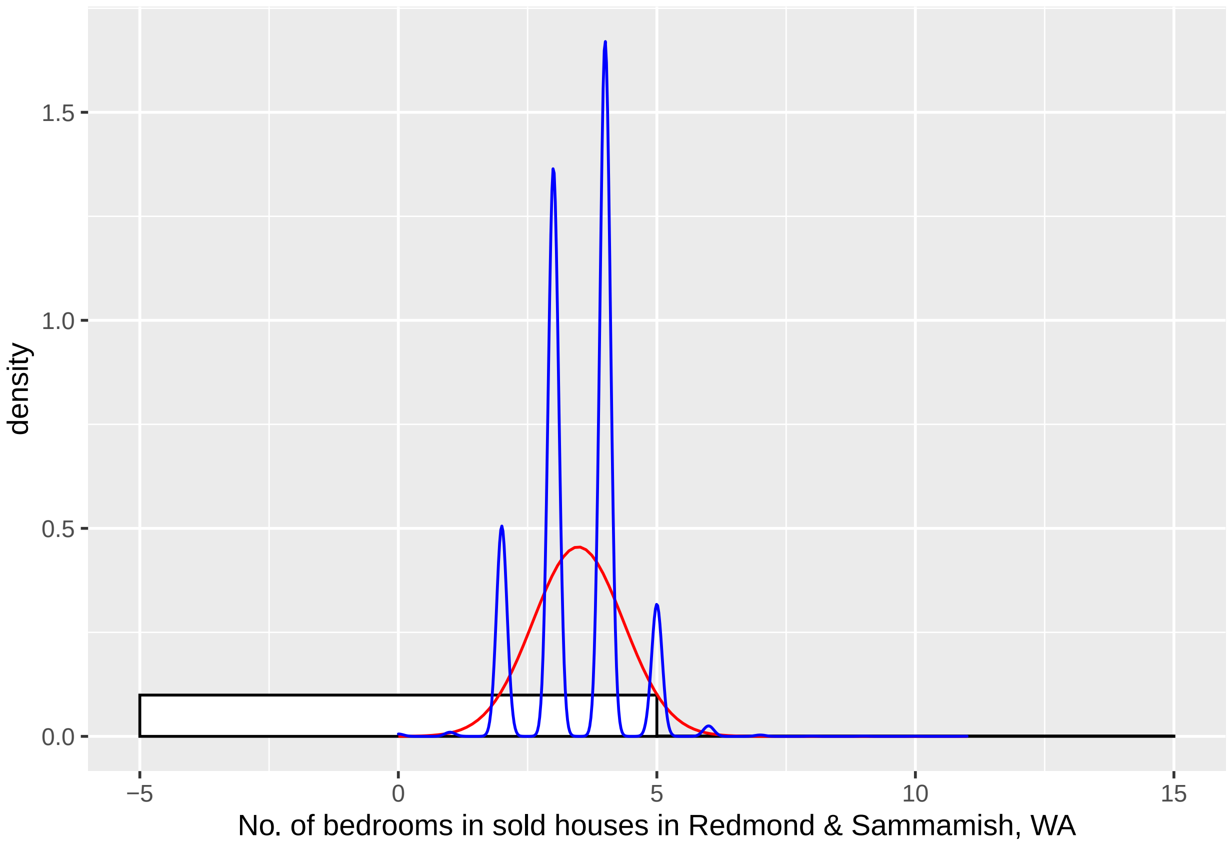
```r
# we can see that we only have data for Redmond and Sammamish, WA
# 7. Average sell price
mean_sale_price <- apply(housing_data[,2], MARGIN = 2, FUN = mean)
mean_sale_price
```

```
## Sale_Price
##   660737.7
```

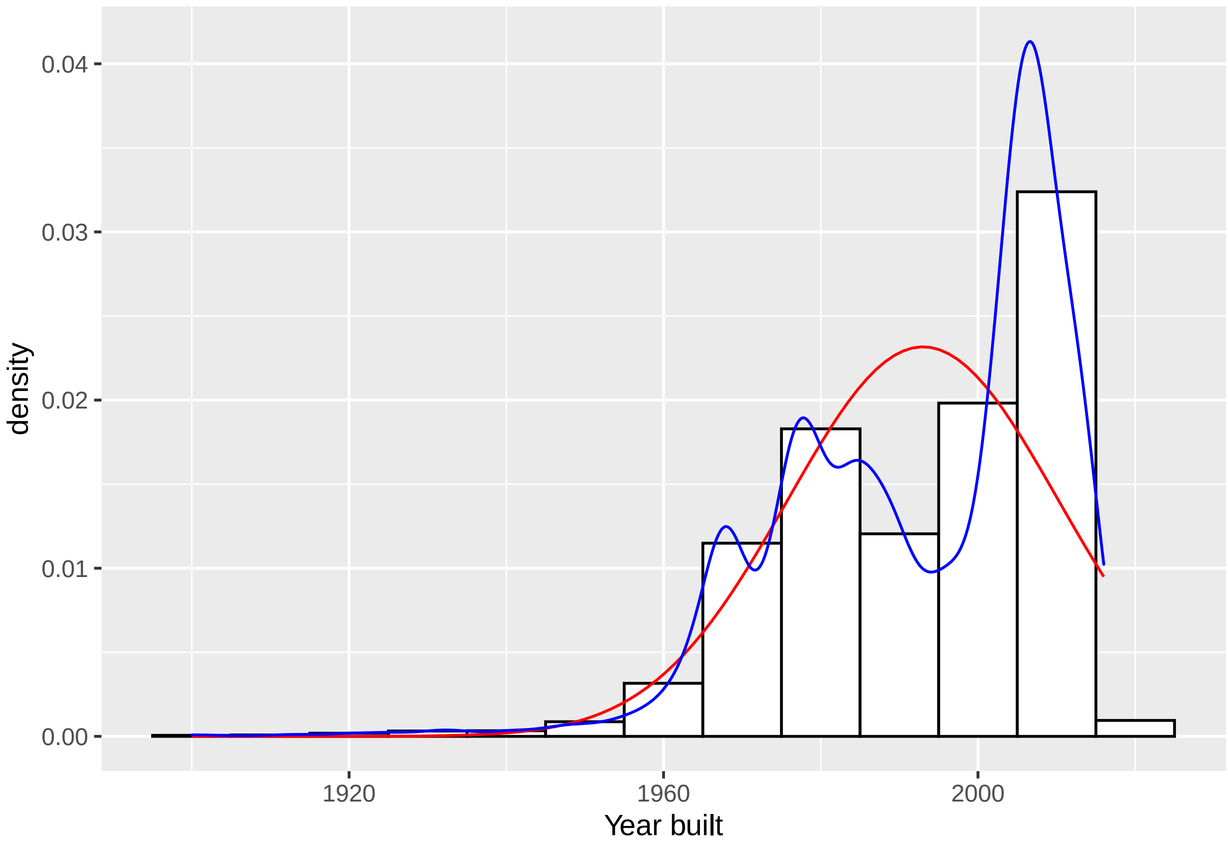
```r
# Average sale price is 660737.7

# we can also analyze data distribution by plotting density curves

# plotting density curve of selling price and comparing it with normal curve plotted with it's own mean
library(ggplot2)
ggplot(housing_data, aes(x=Sale_Price)) +
    xlab("Selling Price") +
    stat_function(color = "Red", data = housing_data, fun = dnorm, args = list(mean = mean(housing_data$
    geom_density(color = "Blue")
```



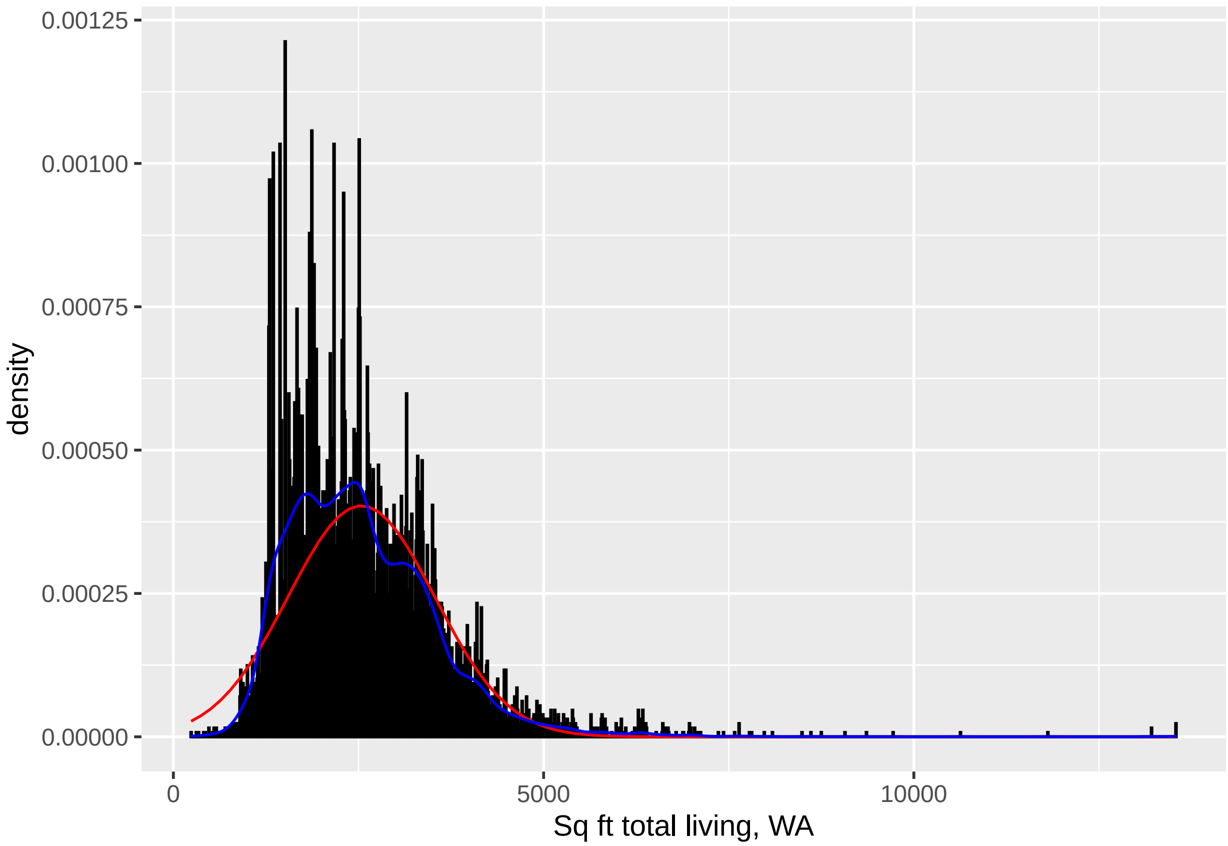

```
# Density chart is showing pretty steep kurtosis and thus there seems to be quite a few observations ne

# plotting density curve of number of bedrooms and comparing it with normal curve plotted with it's own
library(ggplot2)
ggplot(housing_data, aes(x=bedrooms)) +
    geom_histogram(binwidth = 10,
                   color = "Black",
                   fill = "White",
                   aes(y=..density..)) +
    xlab("No. of bedrooms in sold houses in Redmond & Sammamish, WA") +
    stat_function(color = "Red", data = housing_data, fun = dnorm, args = list(mean = mean(housing_data$
    geom_density(color = "Blue")
```



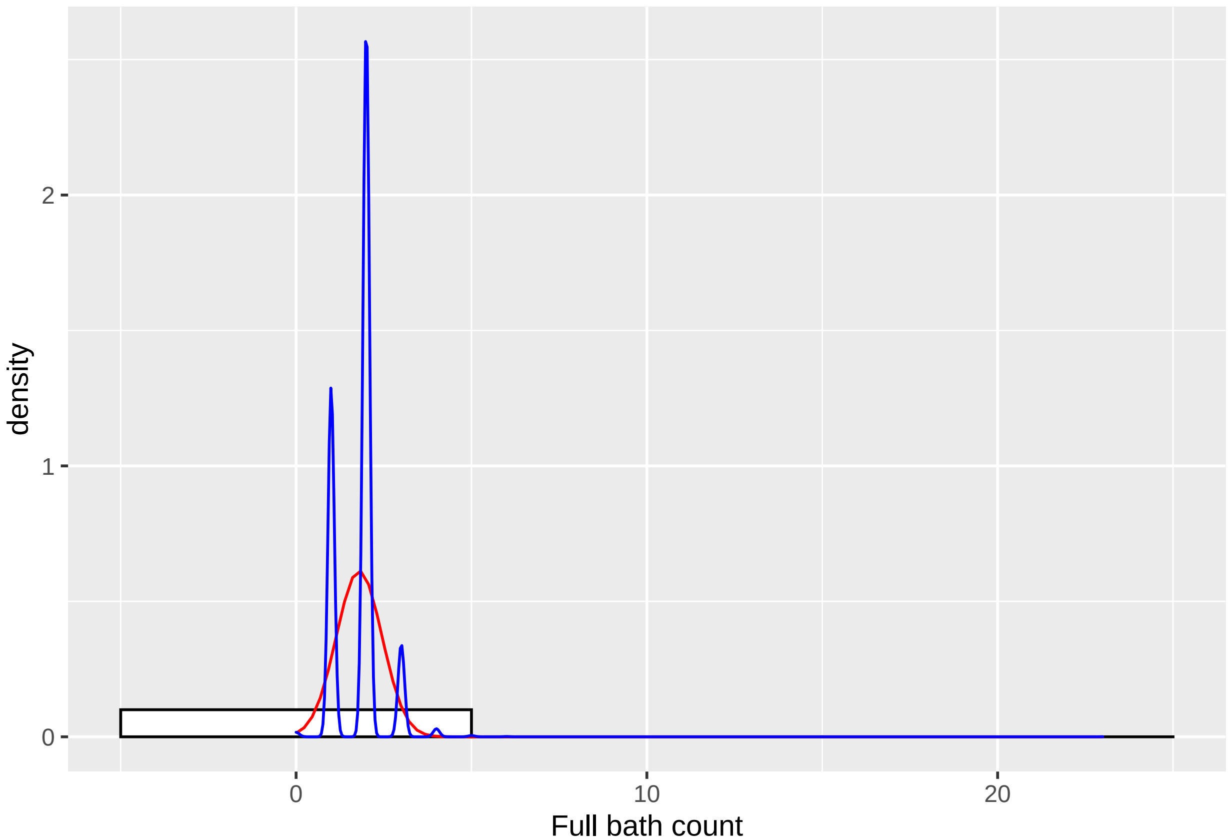

```
# Data seems multi modal which makes me think this more as a categorical data.

# plotting density curve of year_built and comparing it with normal curve plotted with it's own mean an
ggplot(housing_data, aes(x=year_built)) +
    geom_histogram(binwidth = 10,
                   color = "Black",
                   fill = "White",
                   aes(y=..density..)) +
    xlab("Year built") +
    stat_function(color = "Red", data = housing_data, fun = dnorm, args = list(mean = mean(housing_data$
    geom_density(color = "Blue")
```




```
# Chart is negatively skewed with steep kurtosis (leptokurtic) on the right of the scale. We can see th


# plotting density curve of square_feet_total_living and comparing it with normal curve plotted with it
ggplot(housing_data, aes(x=square_feet_total_living)) +
    geom_histogram(binwidth = 10,
                   color = "Black",
                   fill = "White",
                   aes(y=..density..)) +
    xlab("Sq ft total living, WA") +
    stat_function(color = "Red", data = housing_data, fun = dnorm, args = list(mean = mean(housing_data$
    geom_density(color = "Blue")
```




```
# While square feet of total living appears nearly normal it's a bit positively skewed. There are few h


# plotting density curve of bath_full_count and comparing it with normal curve plotted with it's own me
ggplot(housing_data, aes(x=bath_full_count)) +
    geom_histogram(binwidth = 10,
                   color = "Black",
                   fill = "White",
                   aes(y=..density..)) +
    xlab("Full bath count") +
    stat_function(color = "red",data = housing_data, fun = dnorm, args = list(mean = mean(housing_data$b
    geom_density(color = "Blue")
```




```
# Clearly it appears to be multi modal and thus seems to be categorical data.
```

2. e. Identify if there are any outliers

```
str(housing_data)
```

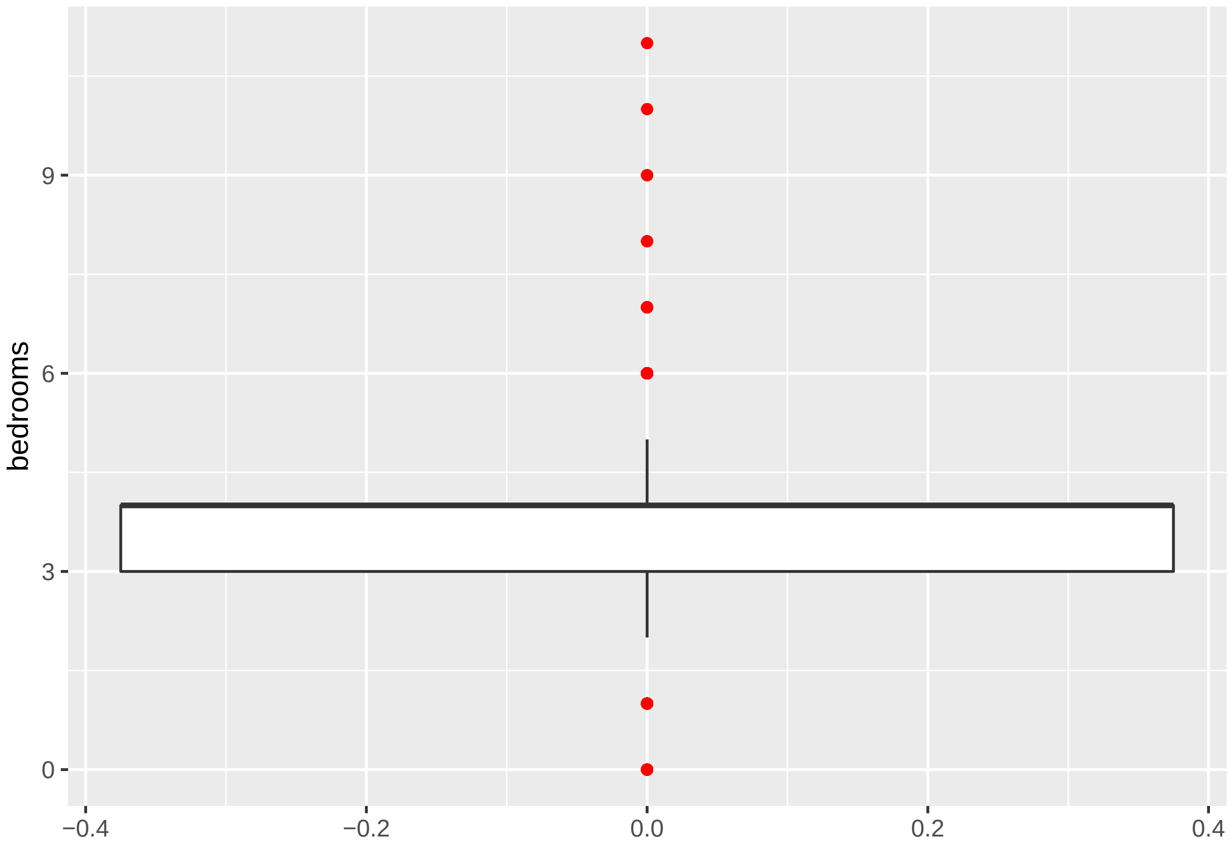
```
## tibble[,24] [12,865 x 24] (S3: tbl_df/tbl/data.frame)
##  $ Sale_Date               : POSIXct[1:12865], format: "2006-01-03" "2006-01-03" ...
##  $ Sale_Price              : num [1:12865] 698000 649990 572500 420000 369900 ...
##  $ sale_reason             : num [1:12865] 1 1 1 1 1 1 1 1 1 1 ...
##  $ sale_instrument         : num [1:12865] 3 3 3 3 3 15 3 3 3 3 ...
##  $ sale_warning            : chr [1:12865] NA NA NA NA ...
##  $ sitetype                : chr [1:12865] "R1" "R1" "R1" "R1" ...
##  $ addr_full               : chr [1:12865] "17021 NE 113TH CT" "11927 178TH PL NE" "13315 174TH AVE I
##  $ zip5                    : num [1:12865] 98052 98052 98052 98052 98052 ...
##  $ ctyname                 : chr [1:12865] "REDMOND" "REDMOND" NA "REDMOND" ...
##  $ postalctyn              : chr [1:12865] "REDMOND" "REDMOND" "REDMOND" "REDMOND" ...
##  $ lon                     : num [1:12865] -122 -122 -122 -122 -122 ...
##  $ lat                     : num [1:12865] 47.7 47.7 47.7 47.6 47.7 ...
##  $ building_grade          : num [1:12865] 9 9 8 8 7 7 10 10 9 8 ...
##  $ square_feet_total_living: num [1:12865] 2810 2880 2770 1620 1440 4160 3960 3720 4160 2760 ...
##  $ bedrooms                : num [1:12865] 4 4 4 3 3 4 5 4 4 4 ...
##  $ bath_full_count         : num [1:12865] 2 2 1 1 1 2 3 2 2 1 ...
##  $ bath_half_count         : num [1:12865] 1 0 1 0 0 1 0 1 1 0 ...
##  $ bath_3qtr_count         : num [1:12865] 0 1 1 1 1 1 1 0 1 1 ...
##  $ year_built              : num [1:12865] 2003 2006 1987 1968 1980 ...
##  $ year_renovated          : num [1:12865] 0 0 0 0 0 0 0 0 0 0 ...
##  $ current_zoning          : chr [1:12865] "R4" "R4" "R6" "R4" ...
```

```
##  $ sq_ft_lot                : num [1:12865] 6635 5570 8444 9600 7526 ...
##  $ prop_type                : chr [1:12865] "R" "R" "R" "R" ...
##  $ present_use              : num [1:12865] 2 2 2 2 2 2 2 2 2 2 ...
```

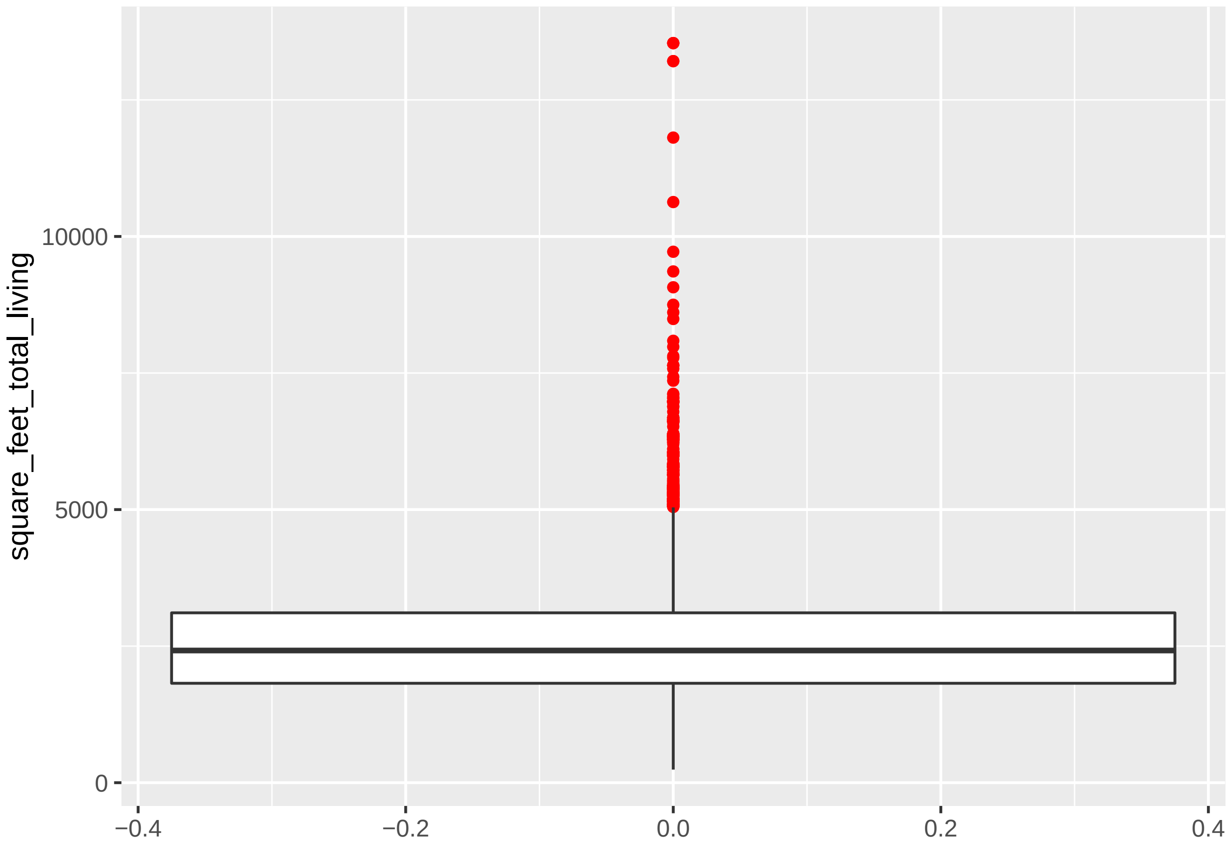
```
# Identifying stats just by looking at some basic descriptive stats
round(stat.desc(housing_data[,c("square_feet_total_living","bedrooms","bath_full_count","year_renovated
```

```
##              square_feet_total_living  bedrooms bath_full_count year_renovated
## nbr.val                     12865.000 12865.000       12865.000      12865.000
## nbr.null                        0.000    19.000          51.000      12696.000
## nbr.na                          0.000     0.000           0.000          0.000
## min                           240.000     0.000           0.000          0.000
## max                         13540.000    11.000          23.000       2016.000
## range                       13300.000    11.000          23.000       2016.000
## sum                      32670747.000 44753.000       23137.000     337633.000
## median                       2420.000     4.000           2.000          0.000
## mean                         2539.506     3.479           1.798         26.244
## SE.mean                         8.727     0.008           0.006          2.006
## CI.mean.0.95                   17.106     0.015           0.011          3.931
## var                        979738.805     0.768           0.424      51748.325
## std.dev                       989.818     0.876           0.651        227.483
## coef.var                        0.390     0.252           0.362          8.668
##                year_built    sq_ft_lot
## nbr.val         12865.000 1.286500e+04
## nbr.null            0.000 0.000000e+00
## nbr.na              0.000 0.000000e+00
## min              1900.000 7.850000e+02
## max              2016.000 1.631322e+06
## range             116.000 1.630537e+06
## sum          25639979.000 2.859705e+08
## median           1998.000 7.965000e+03
## mean             1993.003 2.222857e+04
## SE.mean             0.152 5.019510e+02
## CI.mean.0.95        0.298 9.838990e+02
## var               296.534 3.241400e+09
## std.dev            17.220 5.693329e+04
## coef.var            0.009 2.561000e+00
```

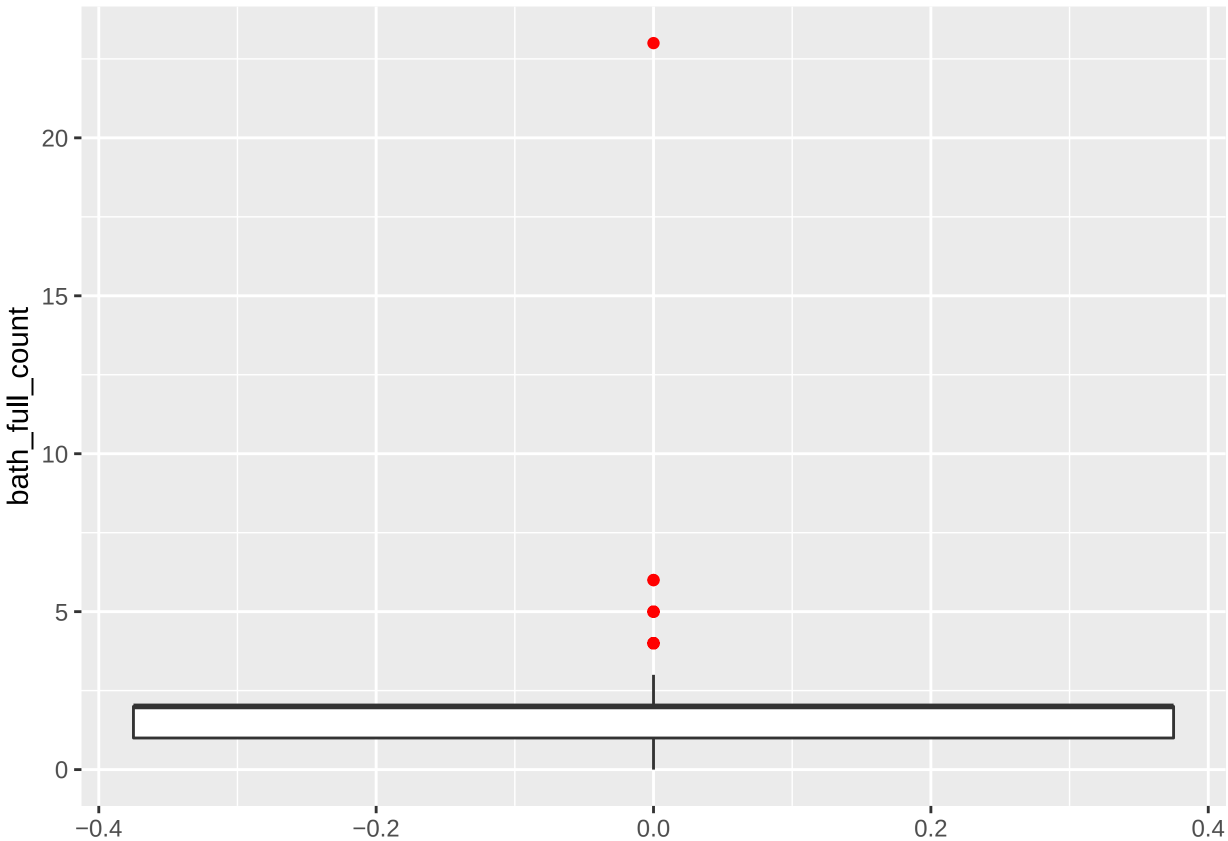
```
# Observations are as below  -
# 1. Bedrooms - We seem to have houses with minimum of 0 bedrooms and maximum of 11 bedrooms. Both appe
# 2. square_feet_total_living - We have house with minimum living sq feet as 240 while maximum being 13
# 3. bath_full_count - We have house with 0 bathroom and 23 bathrooms, while on an average houses seems
# 4. year_built - We have house built in 1900 and recent house that is built is 2016. Average houses ar

# Let's also plot box plots for above variables to see outliers.
ggplot(housing_data, aes(y = bedrooms)) +
  geom_boxplot(outlier.colour = "Red")
```



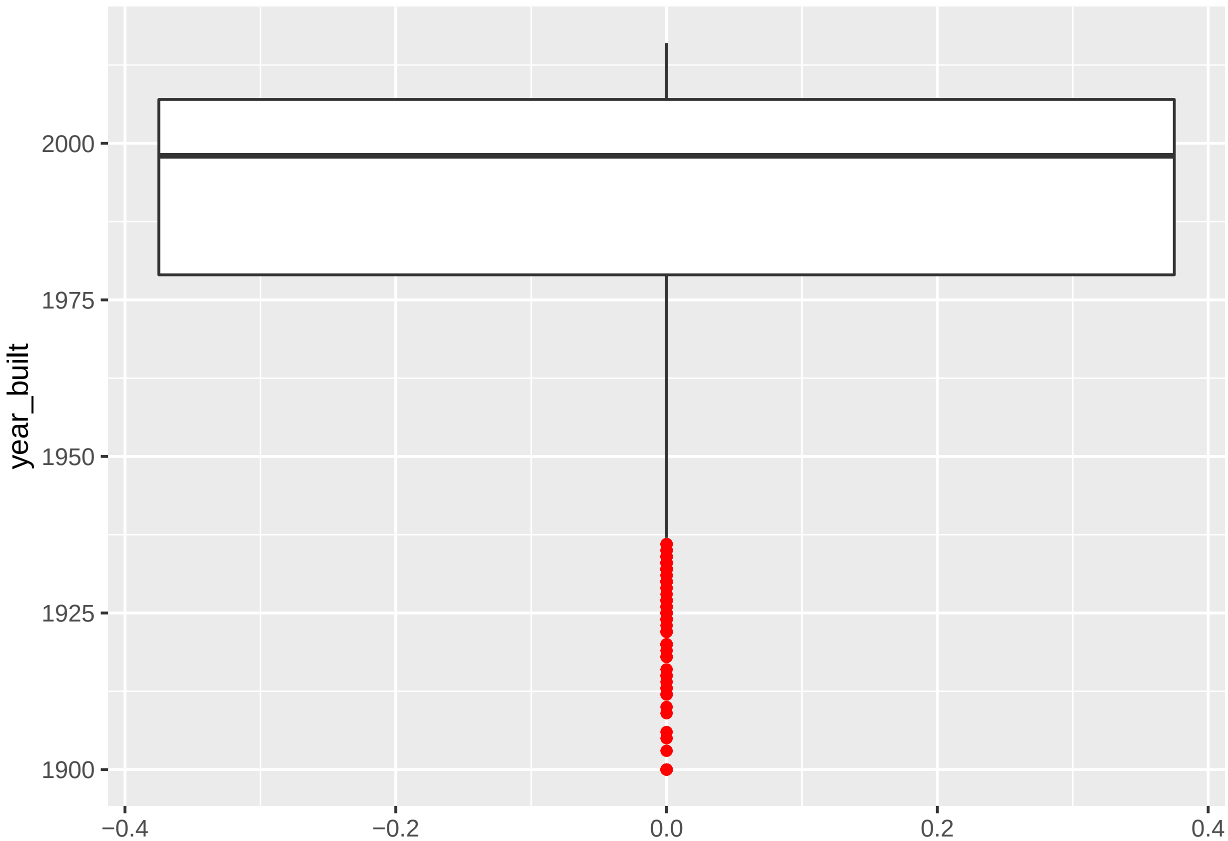

```
# It seems all houses with number of bedrooms >= 6 and <= 1 are marked as outliers in red

ggplot(housing_data, aes(y = square_feet_total_living)) +
  geom_boxplot(outlier.colour = "Red")
```




```
# It seems all houses with living sq ft >= 5000 are marked as outliers with mean or avg sq ft living be

ggplot(housing_data, aes(y = bath_full_count)) +
  geom_boxplot(outlier.colour = "Red")
```




```
# It seems all houses with number of full baths >= 3 (approx) are marked as outliers with mean at 1.8

ggplot(housing_data, aes(y = year_built)) +
  geom_boxplot(outlier.colour = "Red")
```




```
# Any house built before 1938 (approx) seems to be marked as an outlier in red
```

2. f. create at least two new variables

```
# deriving year of sale of the house
housing_data["year_of_sale"] <- substr(housing_data$Sale_Date,1,4)
# derive renovated flag
housing_data["is_renovated"] <- ifelse(housing_data$year_renovated != 0, 1, 0)
str(housing_data)
```

```
## tibble[,26] [12,865 x 26] (S3: tbl_df/tbl/data.frame)
##  $ Sale_Date               : POSIXct[1:12865], format: "2006-01-03" "2006-01-03" ...
##  $ Sale_Price              : num [1:12865] 698000 649990 572500 420000 369900 ...
##  $ sale_reason             : num [1:12865] 1 1 1 1 1 1 1 1 1 1 ...
##  $ sale_instrument         : num [1:12865] 3 3 3 3 3 15 3 3 3 3 ...
##  $ sale_warning            : chr [1:12865] NA NA NA NA ...
##  $ sitetype                : chr [1:12865] "R1" "R1" "R1" "R1" ...
##  $ addr_full               : chr [1:12865] "17021 NE 113TH CT" "11927 178TH PL NE" "13315 174TH AVE 
##  $ zip5                    : num [1:12865] 98052 98052 98052 98052 98052 ...
##  $ ctyname                 : chr [1:12865] "REDMOND" "REDMOND" NA "REDMOND" ...
##  $ postalctyn              : chr [1:12865] "REDMOND" "REDMOND" "REDMOND" "REDMOND" ...
##  $ lon                     : num [1:12865] -122 -122 -122 -122 -122 ...
##  $ lat                     : num [1:12865] 47.7 47.7 47.7 47.6 47.7 ...
##  $ building_grade          : num [1:12865] 9 9 8 8 7 7 10 10 9 8 ...
##  $ square_feet_total_living: num [1:12865] 2810 2880 2770 1620 1440 4160 3960 3720 4160 2760 ...
##  $ bedrooms                : num [1:12865] 4 4 4 3 3 4 5 4 4 4 ...
##  $ bath_full_count         : num [1:12865] 2 2 1 1 1 2 3 2 2 1 ...
##  $ bath_half_count         : num [1:12865] 1 0 1 0 0 1 0 1 1 0 ...
```

```
##  $ bath_3qtr_count         : num [1:12865] 0 1 1 1 1 1 1 0 1 1 ...
##  $ year_built              : num [1:12865] 2003 2006 1987 1968 1980 ...
##  $ year_renovated          : num [1:12865] 0 0 0 0 0 0 0 0 0 0 ...
##  $ current_zoning          : chr [1:12865] "R4" "R4" "R6" "R4" ...
##  $ sq_ft_lot               : num [1:12865] 6635 5570 8444 9600 7526 ...
##  $ prop_type               : chr [1:12865] "R" "R" "R" "R" ...
##  $ present_use             : num [1:12865] 2 2 2 2 2 2 2 2 2 2 ...
##  $ year_of_sale            : chr [1:12865] "2006" "2006" "2006" "2006" ...
##  $ is_renovated            : num [1:12865] 0 0 0 0 0 0 0 0 0 0 ...
```