

Final_project

Edgar Rosales,

2023-12-11

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

Part 1:data importing and pre-processing

house_data, which originated from house_sales.csv and imported through the read.csv function, provides a dataset that consists of 21,613 objects and 21 different variables. These variables consist of numerics (int & num) and character strings.

```
house_data <- read.csv("house_sales.csv", na.strings = "")
str(house_data)
```

```
## 'data.frame': 21613 obs. of 21 variables:
## $ id : num 7.13e+09 6.41e+09 5.63e+09 2.49e+09 1.95e+09 ...
## $ date : chr "20141013T000000" "20141209T000000" "20150225T000000" "20141209T000000" ...
## $ price : num 221900 538000 180000 604000 510000 ...
## $ bedrooms : num 3 3 2 4 3 4 3 3 3 3 ...
## $ bathrooms : num 1 2.25 1 3 2 4.5 2.25 1.5 1 2.5 ...
## $ sqft_living : num 1180 2570 770 1960 1680 ...
## $ sqft_lot : num 5650 7242 10000 5000 8080 ...
## $ floors : num 1 2 1 1 1 1 2 1 1 2 ...
## $ waterfront : int 0 0 0 0 0 0 0 0 0 0 ...
## $ view : int 0 0 0 0 0 0 0 0 0 0 ...
## $ condition : int 3 3 3 5 3 3 3 3 3 3 ...
## $ grade : int 7 7 6 7 8 11 7 7 7 7 ...
## $ sqft_above : int 1180 2170 770 1050 1680 3890 1715 1060 1050 1890 ...
## $ sqft_basement: int 0 400 0 910 0 1530 0 0 730 0 ...
## $ yr_built : int 1955 1951 1933 1965 1987 2001 1995 1963 1960 2003 ...
## $ yr_renovated : int 0 1991 0 0 0 0 0 0 0 0 ...
## $ zipcode : int 98178 98125 98028 98136 98074 98053 98003 98198 98146 98038 ...
## $ lat : num 47.5 47.7 47.7 47.5 47.6 ...
## $ long : num -122 -122 -122 -122 -122 ...
## $ sqft_living15: int 1340 1690 2720 1360 1800 4760 2238 1650 1780 2390 ...
## $ sqft_lot15 : int 5650 7639 8062 5000 7503 101930 6819 9711 8113 7570 ...
```

taking care of missing data

```
colSums(is.na(house_data))
```

```
##           id           date           price           bedrooms           bathrooms
##           0             0             0           1134           1068
## sqft_living sqft_lot       floors waterfront           view
##           1110          1044             0             0             0
## condition      grade sqft_above sqft_basement yr_built
##           0             0             0             0             0
## yr_renovated  zipcode      lat           long sqft_living15
##           0             0             0             0             0
## sqft_lot15
##           0
```

```
house_data_clean <- na.omit(house_data)
```

```
colSums(is.na(house_data_clean))
```

```
##           id           date           price           bedrooms           bathrooms
##           0             0             0             0             0
## sqft_living sqft_lot       floors waterfront           view
##           0             0             0             0             0
## condition      grade sqft_above sqft_basement yr_built
##           0             0             0             0             0
## yr_renovated  zipcode      lat           long sqft_living15
##           0             0             0             0             0
## sqft_lot15
##           0
```

checking for any duplicated rows

```
#Checking for duplicated rows
duplicates <- house_data_clean[duplicated(house_data_clean), ]
```

```
#Display the duplicated rows
print(duplicates)
```

```
## [1] id           date           price           bedrooms           bathrooms
## [6] sqft_living sqft_lot       floors waterfront           view
## [11] condition      grade sqft_above sqft_basement yr_built
## [16] yr_renovated  zipcode      lat           long sqft_living15
## [21] sqft_lot15
## <0 rows> (or 0-length row.names)
```

```
#convert 'date' to Date format
house_data_clean$date <- as.Date(house_data_clean$date, format="%Y%m%dT%H%M%S")
str(house_data_clean)
```

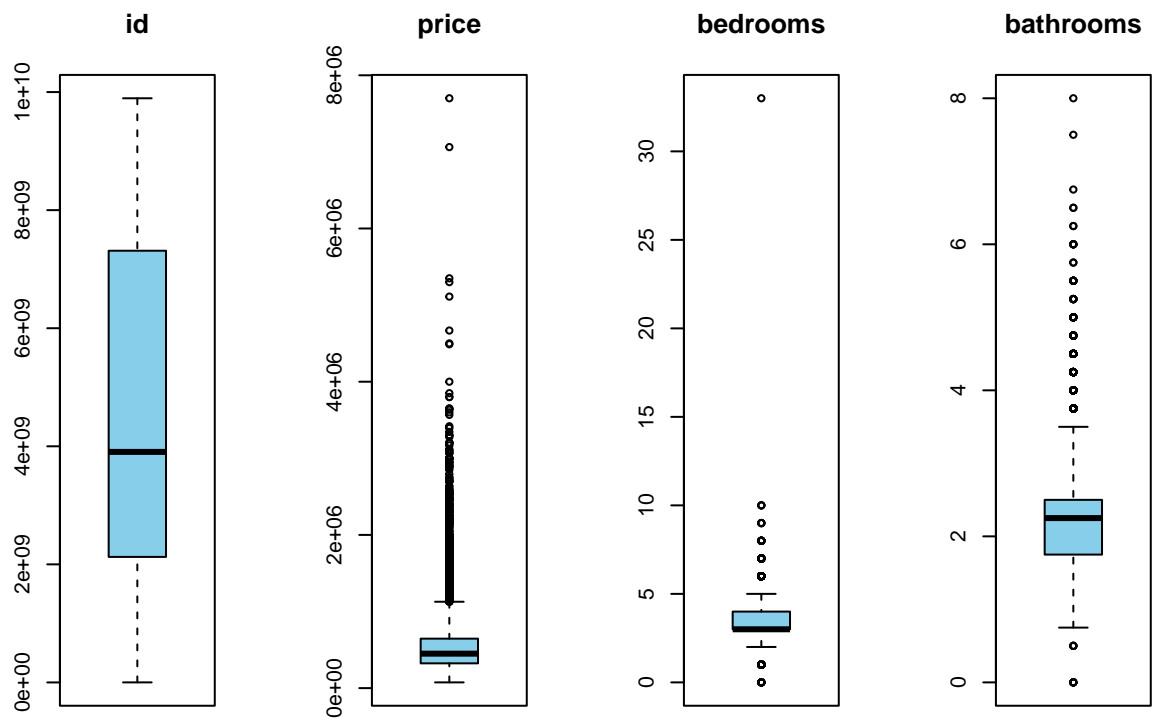
```
## 'data.frame':   17618 obs. of  21 variables:
## $ id           : num  7.13e+09 6.41e+09 5.63e+09 2.49e+09 1.95e+09 ...
## $ date          : Date, format: "2014-10-13" "2014-12-09" ...
## $ price          : num  221900 538000 180000 604000 510000 ...
```

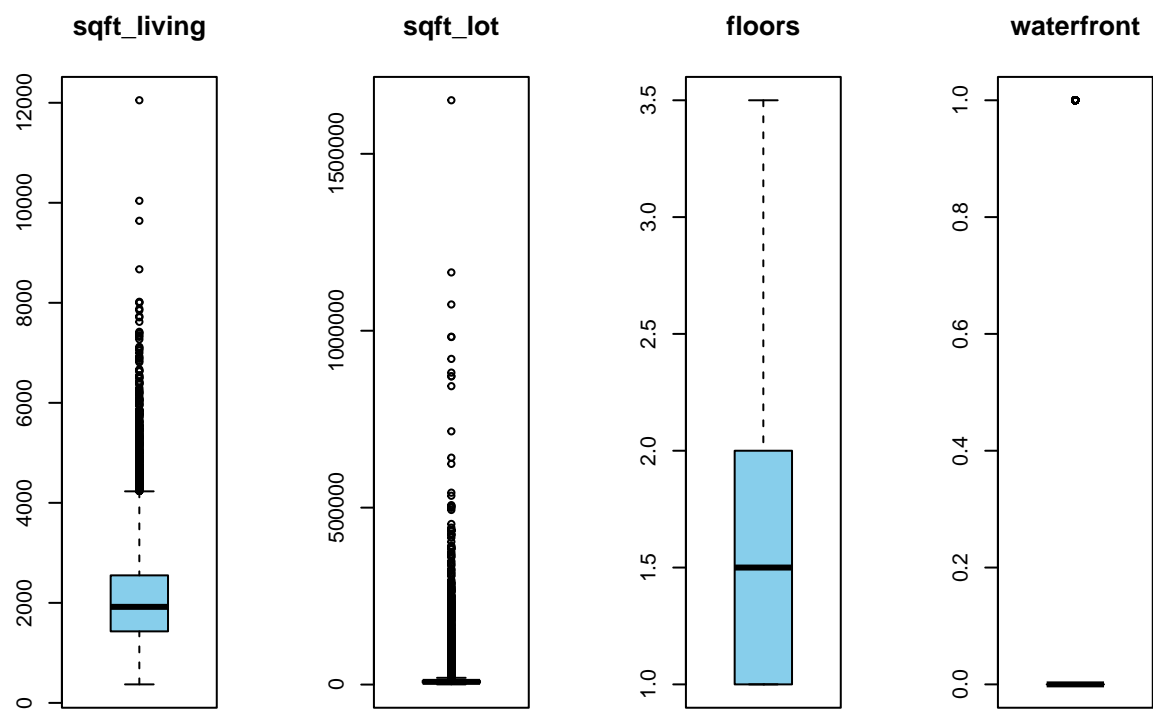
```
## $ bedrooms      : num  3 3 2 4 3 4 3 3 3 3 ...
## $ bathrooms     : num  1 2.25 1 3 2 4.5 2.25 1.5 1 2.5 ...
## $ sqft_living   : num  1180 2570 770 1960 1680 ...
## $ sqft_lot      : num  5650 7242 10000 5000 8080 ...
## $ floors        : num  1 2 1 1 1 1 2 1 1 2 ...
## $ waterfront    : int   0 0 0 0 0 0 0 0 0 0 ...
## $ view          : int   0 0 0 0 0 0 0 0 0 0 ...
## $ condition     : int   3 3 3 5 3 3 3 3 3 3 ...
## $ grade         : int   7 7 6 7 8 11 7 7 7 7 ...
## $ sqft_above    : int  1180 2170 770 1050 1680 3890 1715 1060 1050 1890 ...
## $ sqft_basement: int   0 400 0 910 0 1530 0 0 730 0 ...
## $ yr_built      : int  1955 1951 1933 1965 1987 2001 1995 1963 1960 2003 ...
## $ yr_renovated  : int   0 1991 0 0 0 0 0 0 0 0 ...
## $ zipcode       : int  98178 98125 98028 98136 98074 98053 98003 98198 98146 98038 ...
## $ lat           : num  47.5 47.7 47.7 47.5 47.6 ...
## $ long          : num -122 -122 -122 -122 -122 ...
## $ sqft_living15: int  1340 1690 2720 1360 1800 4760 2238 1650 1780 2390 ...
## $ sqft_lot15    : int  5650 7639 8062 5000 7503 101930 6819 9711 8113 7570 ...
## - attr(*, "na.action")= 'omit' Named int [1:3995] 11 13 19 24 25 27 32 33 38 39 ...
## ..- attr(*, "names")= chr [1:3995] "11" "13" "19" "24" ...
```

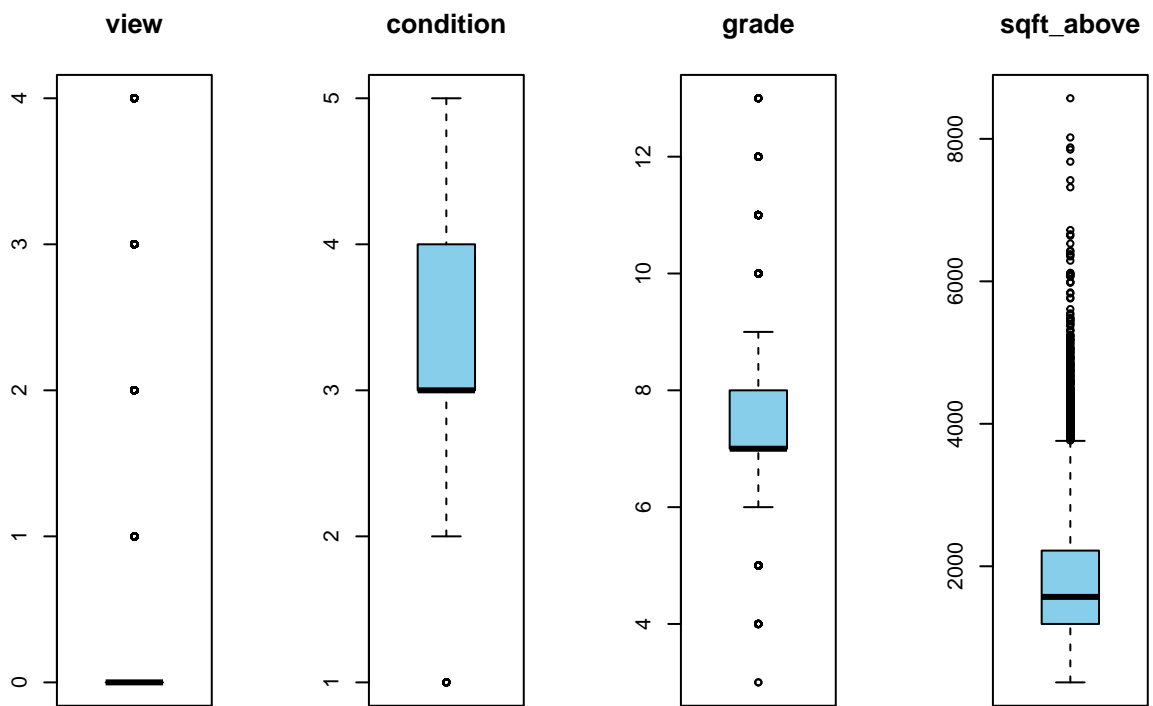
handling outliers

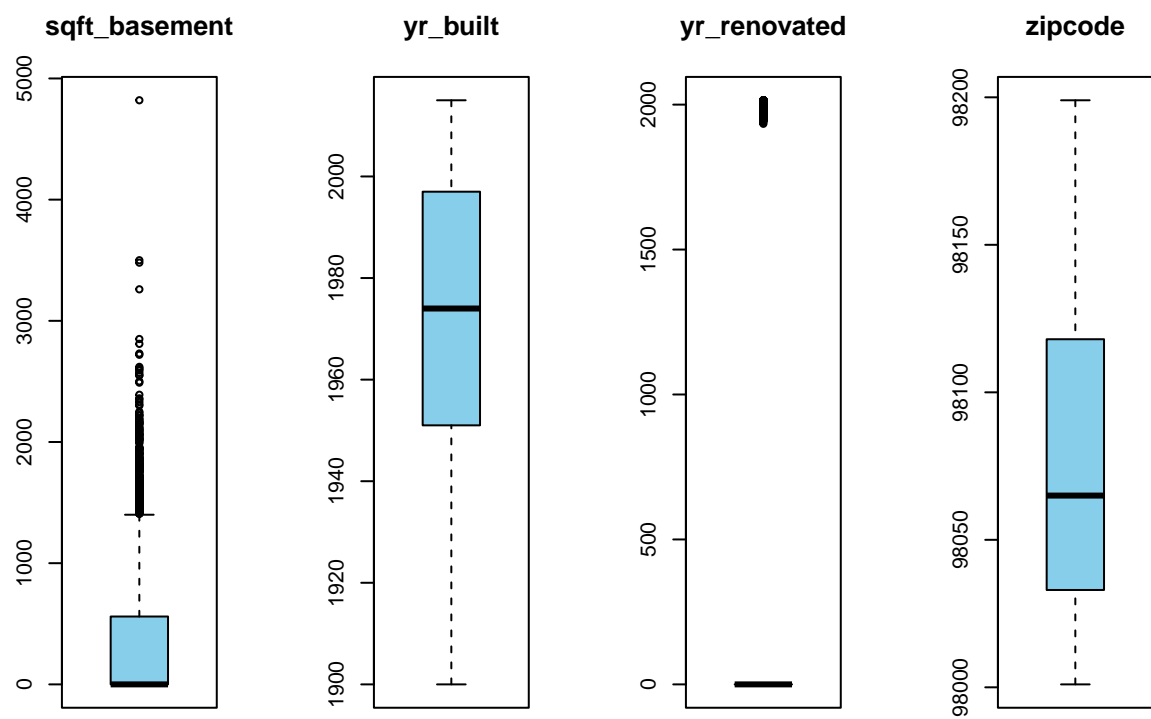
Here we are showing an example of what our data looks like before we handle outliers

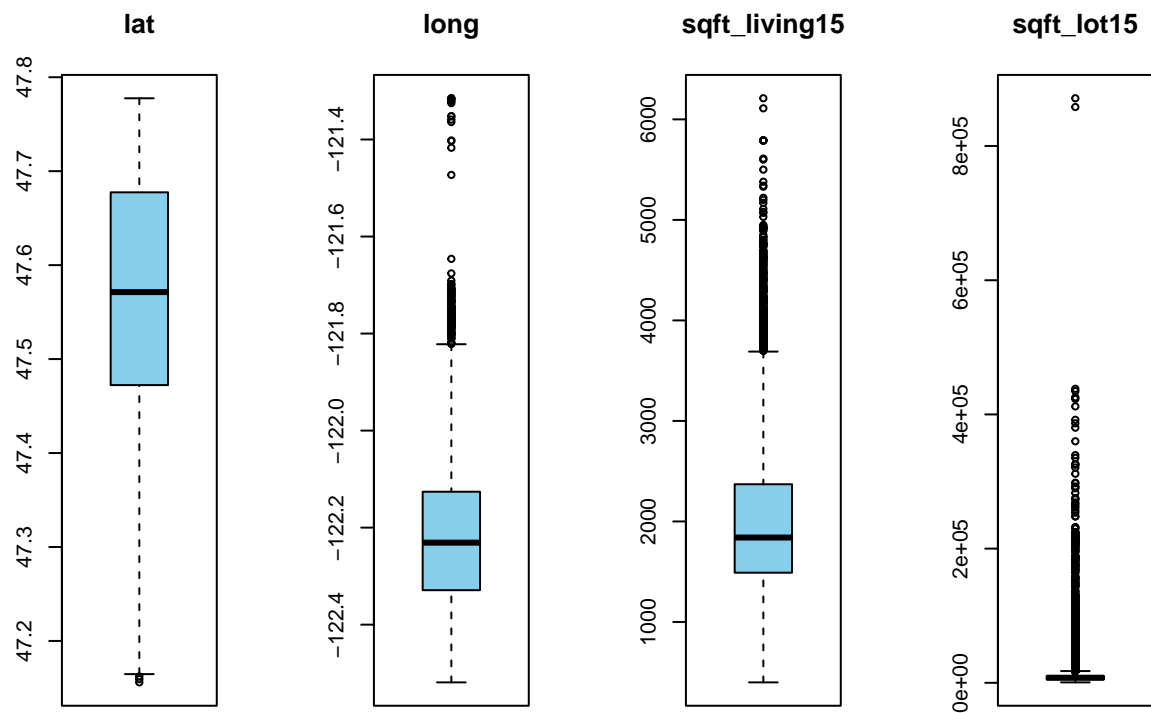
```
#boxplot for every numeric variable
par(mfrow = c(1, 4))
for (i in 1:ncol(house_data_clean)) {
  if (is.numeric(house_data_clean[, i])) {
    boxplot(house_data_clean[, i], main = names(house_data_clean)[i], col = "skyblue", border = "black",
  }
}
```











here we are showing the changes in boxblots after adjusting for outliers

```
#specify the constant multiplier for outlier detection
k <- 1.5

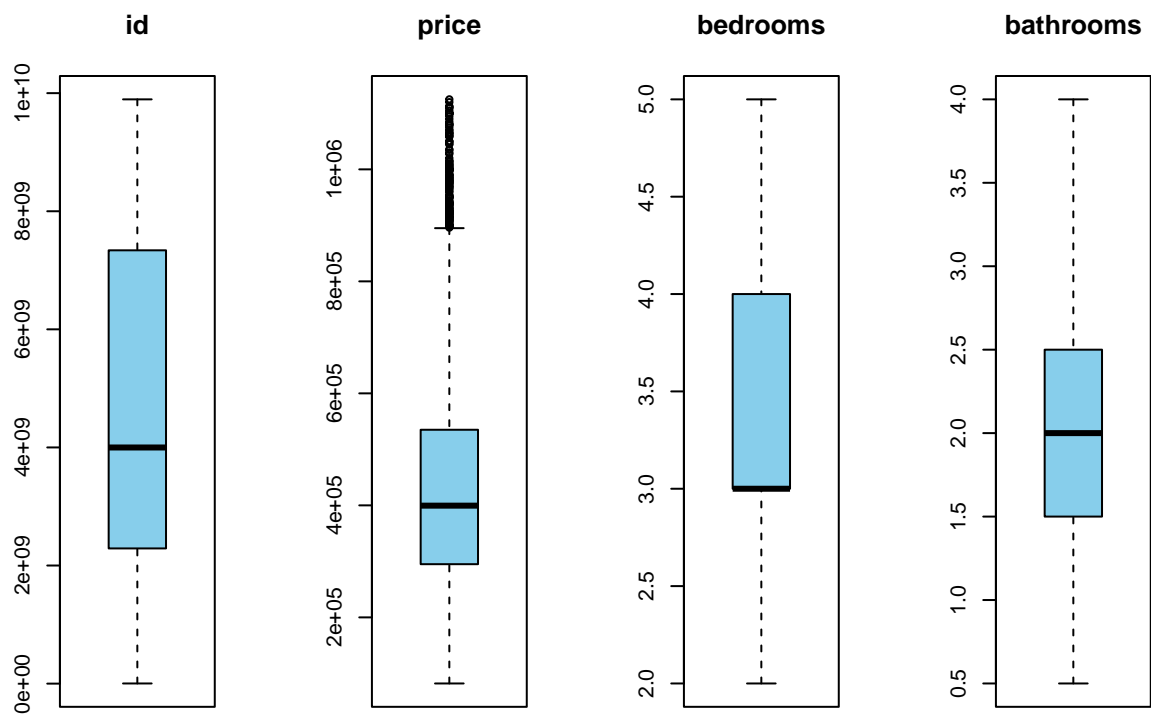
#loop through each numeric variable
for (col in names(house_data_clean)) {
  if (is.numeric(house_data_clean[[col]])) {
    # Calculate Q1, Q3, and IQR for the current variable
    Q1 <- quantile(house_data_clean[[col]], 0.25)
    Q3 <- quantile(house_data_clean[[col]], 0.75)
    IQR <- Q3 - Q1

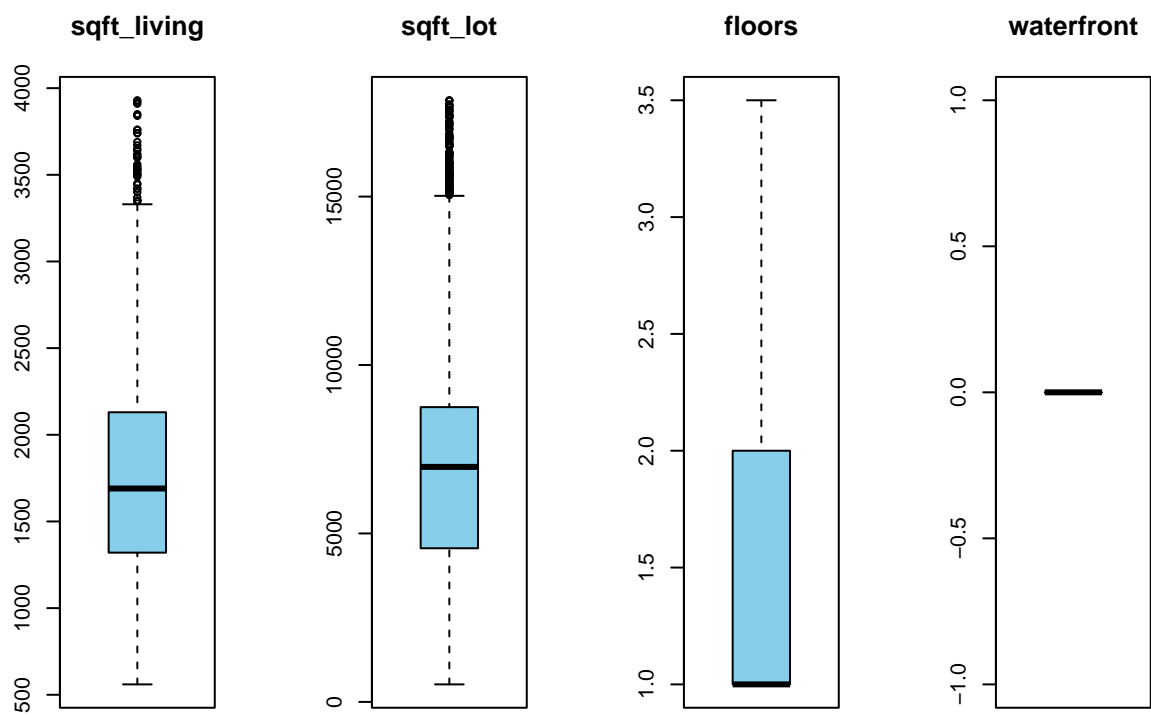
    #outliers for the current variable
    outliers <- house_data_clean[[col]] < Q1 - k * IQR | house_data_clean[[col]] > Q3 + k * IQR

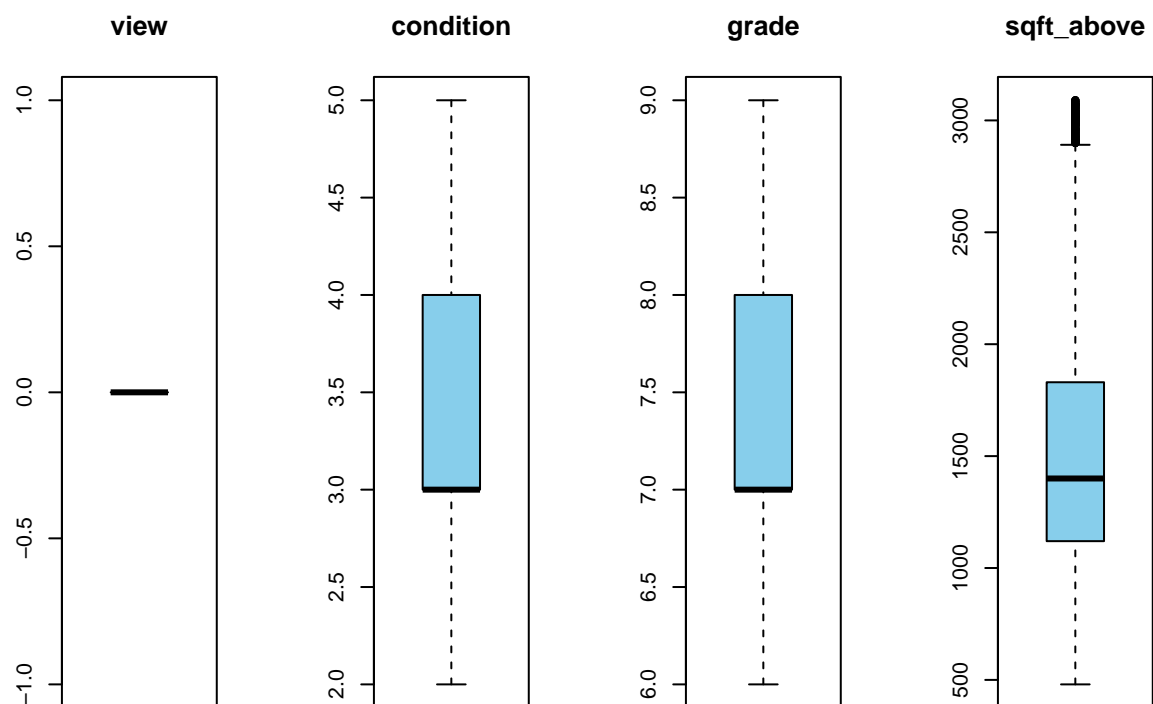
    #Remove outliers from the dataset
    house_data_clean <- house_data_clean[!outliers, ]
  }
}
```

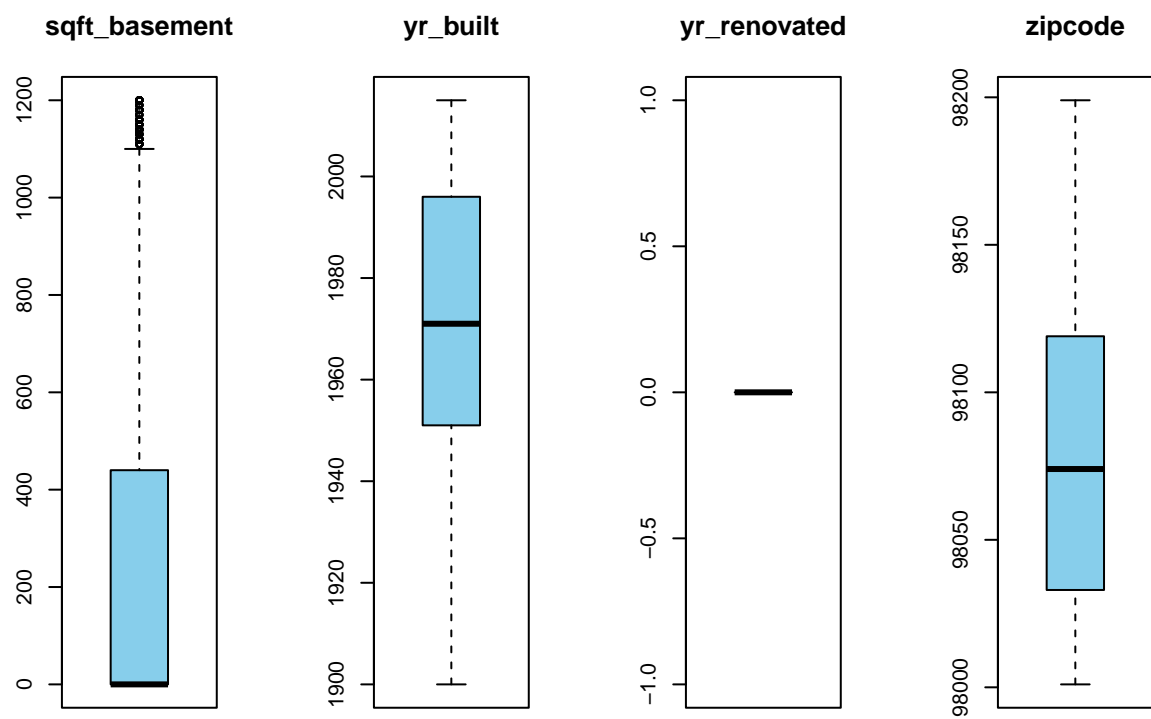
```
par(mfrow = c(1, 4))
for (i in 1:ncol(house_data_clean)) {
  if (is.numeric(house_data_clean[, i])) {
    boxplot(house_data_clean[, i], main = names(house_data_clean)[i], col = "skyblue", border = "black",
  }
}
```

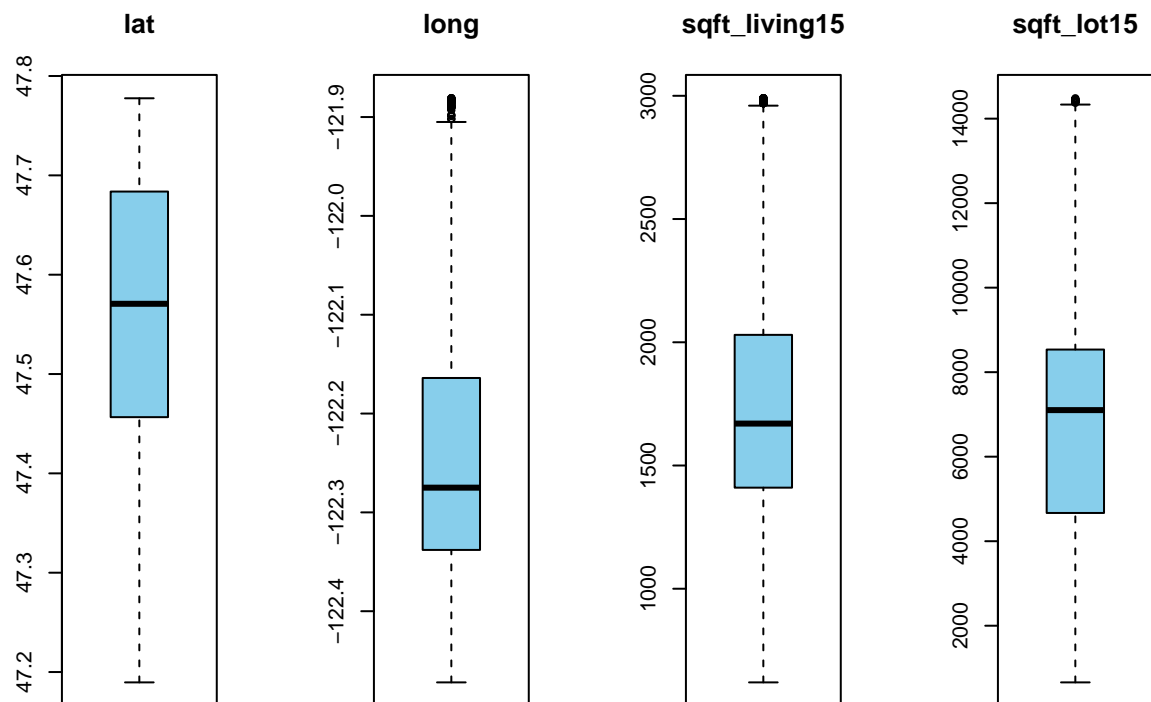

}











Here we use the aggregate function to group taverage house value per zipcode

```
zip_prices <- aggregate(house_data$price, by = list(zipcode = house_data$zipcode), FUN = mean)
colnames(zip_prices) <- c("zipcode", "mean_price")
```

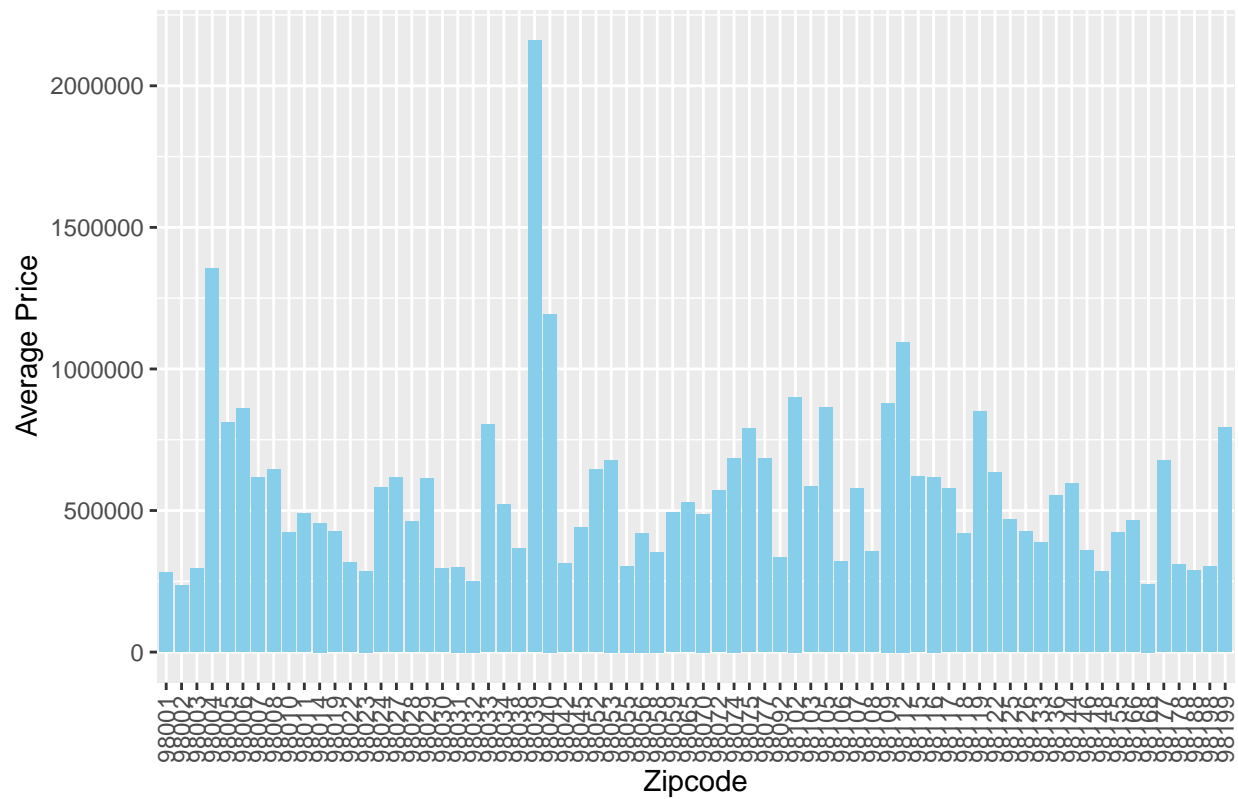
```
head(zip_prices)
```

```
##  zipcode mean_price
## 1  98001  280804.7
## 2  98002  234284.0
## 3  98003  294111.3
## 4  98004 1355927.1
## 5  98005  810164.9
## 6  98006  859684.8
```

```
library(ggplot2)
```

```
ggplot(zip_prices, aes(x = factor(zipcode), y = mean_price)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(title = "Average House Price by Zipcode", x = "Zipcode", y = "Average Price") +
  theme(axis.text.x = element_text(angle = 90, vjust = .5, hjust=2))
```

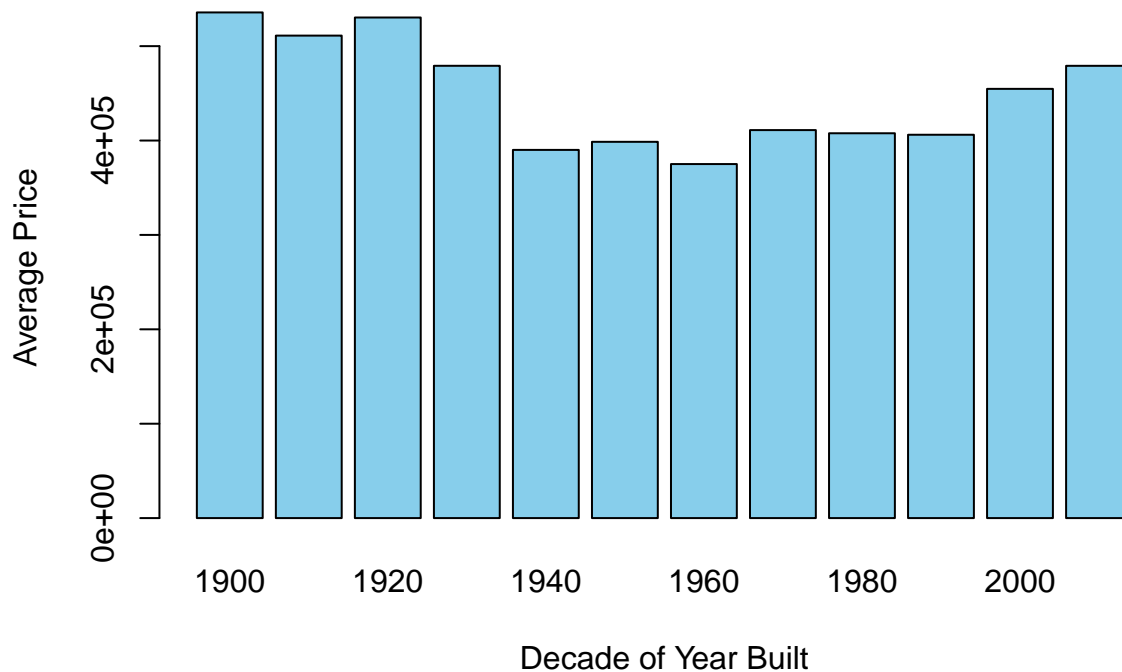
Average House Price by Zipcode



```
house_data_clean$decade_built <- 10 * (house_data_clean$yr_built %/% 10) # Create a decade category
avg_price_by_decade <- aggregate(house_data_clean$price, by = list(decade_built = house_data_clean$decade_built), FUN = mean)
colnames(avg_price_by_decade) <- c("decade_built", "avg_price")

barplot(avg_price_by_decade$avg_price, names.arg = avg_price_by_decade$decade_built, col = "skyblue", m
```

Average House Prices by Decade of Year Built



```
#aggregate average house prices by bedrooms and bathrooms
avg_price_bed_bath <- aggregate(house_data_clean$price, by = list(bedrooms = house_data_clean$bedrooms,
colnames(avg_price_bed_bath) <- c("bedrooms", "bathrooms", "avg_price")

#new column with labels
avg_price_bed_bath$label <- paste(avg_price_bed_bath$bedrooms, "BR, ",
                                avg_price_bed_bath$bathrooms, "BA")

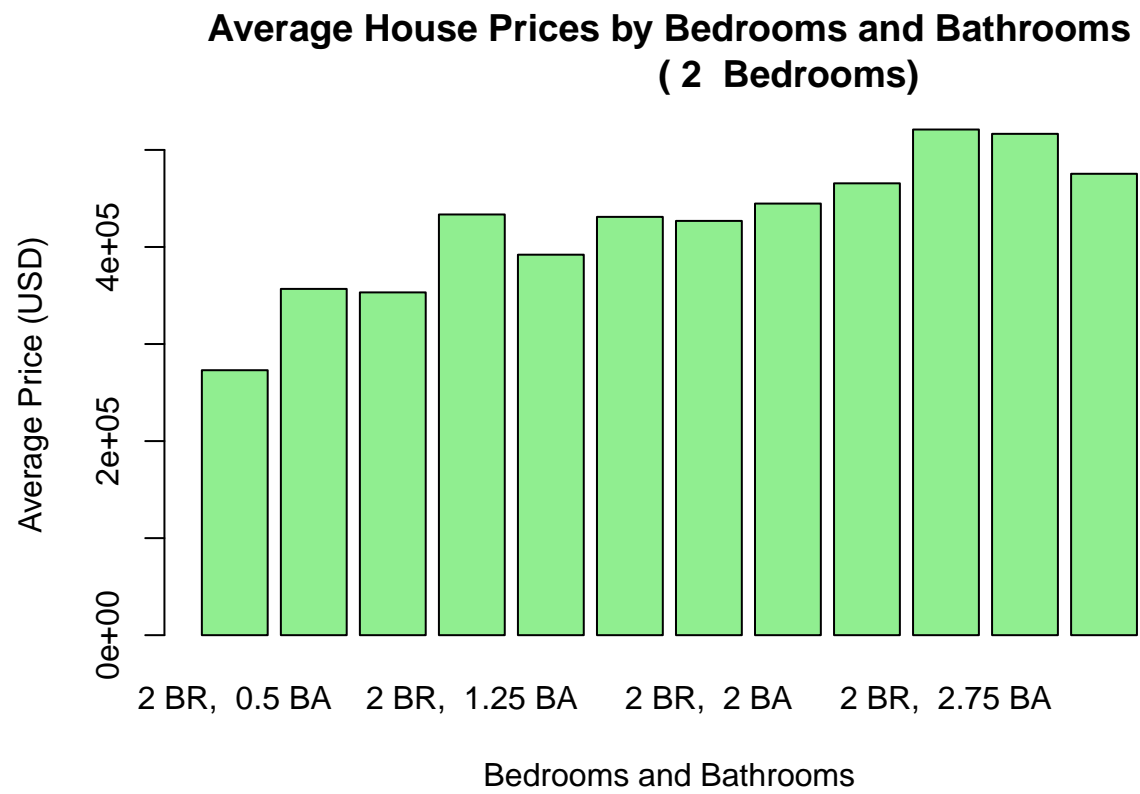
#unique values of 'bedrooms'
unique_bedrooms <- unique(avg_price_bed_bath$bedrooms)

#separate bar charts for each unique number of bedrooms
for (bedroom_value in unique_bedrooms) {
  #subset data for the current number of bedrooms
  subset_data <- subset(avg_price_bed_bath, bedrooms == bedroom_value)

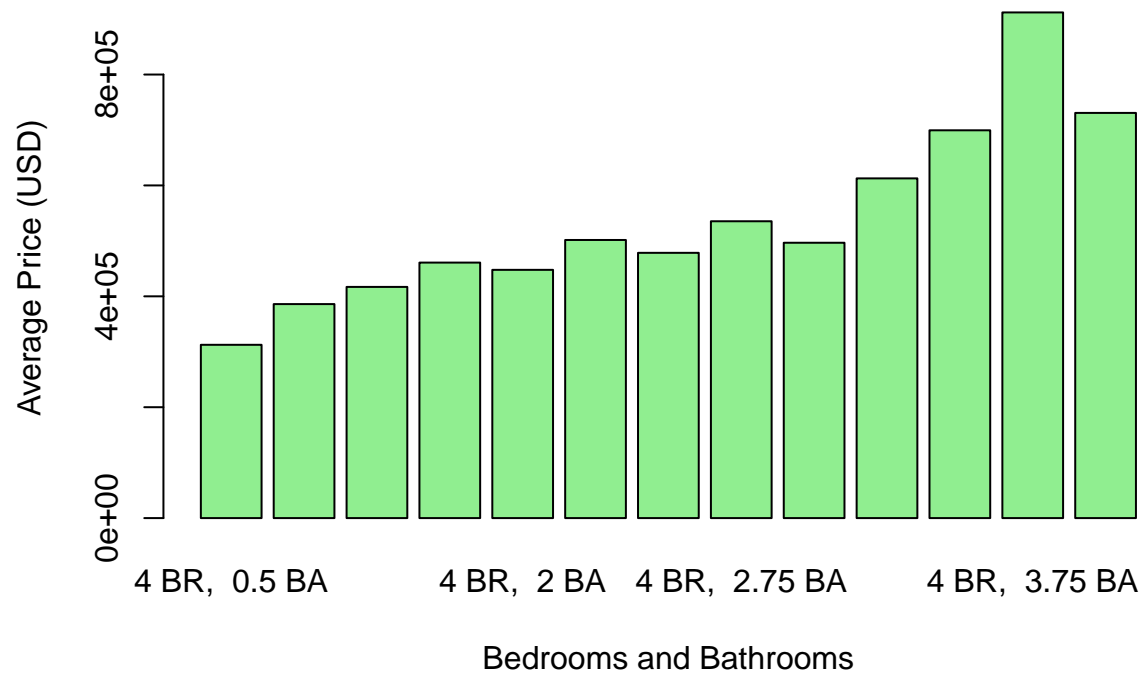
  #new column with labels
  subset_data$label <- paste(subset_data$bedrooms, "BR, ",
                            subset_data$bathrooms, "BA")

  #bar chart
  barplot(subset_data$avg_price, beside = TRUE, names.arg =
    subset_data$label, col = "lightgreen",
    main = paste("Average House Prices by Bedrooms and Bathrooms
                  (", bedroom_value, " Bedrooms)"),
    xlab = "Bedrooms and Bathrooms", ylab = "Average Price (USD)")
}
```

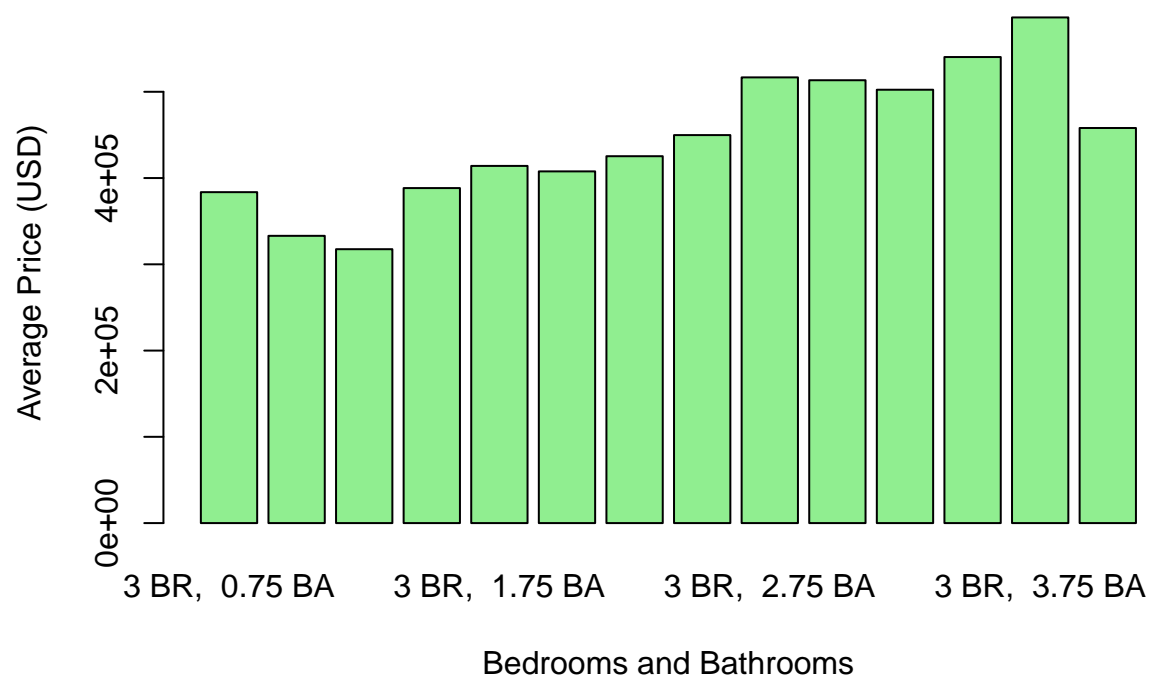
}



**Average House Prices by Bedrooms and Bathrooms
(4 Bedrooms)**



**Average House Prices by Bedrooms and Bathrooms
(3 Bedrooms)**



Average House Prices by Bedrooms and Bathrooms (5 Bedrooms)



feature construction

```
house_data_clean$age <- 2023 - house_data_clean$yr_built
```

```
house_data_clean$bed_bath_ratio <- house_data_clean$bedrooms / house_data_clean$bathrooms
```

```
house_data_clean$total_sqft <- house_data_clean$sqft_living + house_data_clean$sqft_lot
```

```
house_data_clean$month_sold <- as.numeric(substr(house_data_clean$date, 5, 6))
```

```
house_data_clean$price_per_sqft <- house_data_clean$price / house_data_clean$sqft_living
```

Normalizing

```
#function to perform Min-Max scaling
min_max_scaling <- function(x) {
  return((x - min(x)) / (max(x) - min(x)))
}
```

```
#columns to normalize
columns_to_normalize <- c("age", "bed_bath_ratio", "total_sqft", "month_sold", "price_per_sqft")
```

```
#Min-Max scaling to selected columns
```

```
house_data_clean[columns_to_normalize] <- lapply(house_data_clean[columns_to_normalize], min_max_scaling)
```

```
#summary statistics of normalized features
summary(house_data_clean[columns_to_normalize])
```

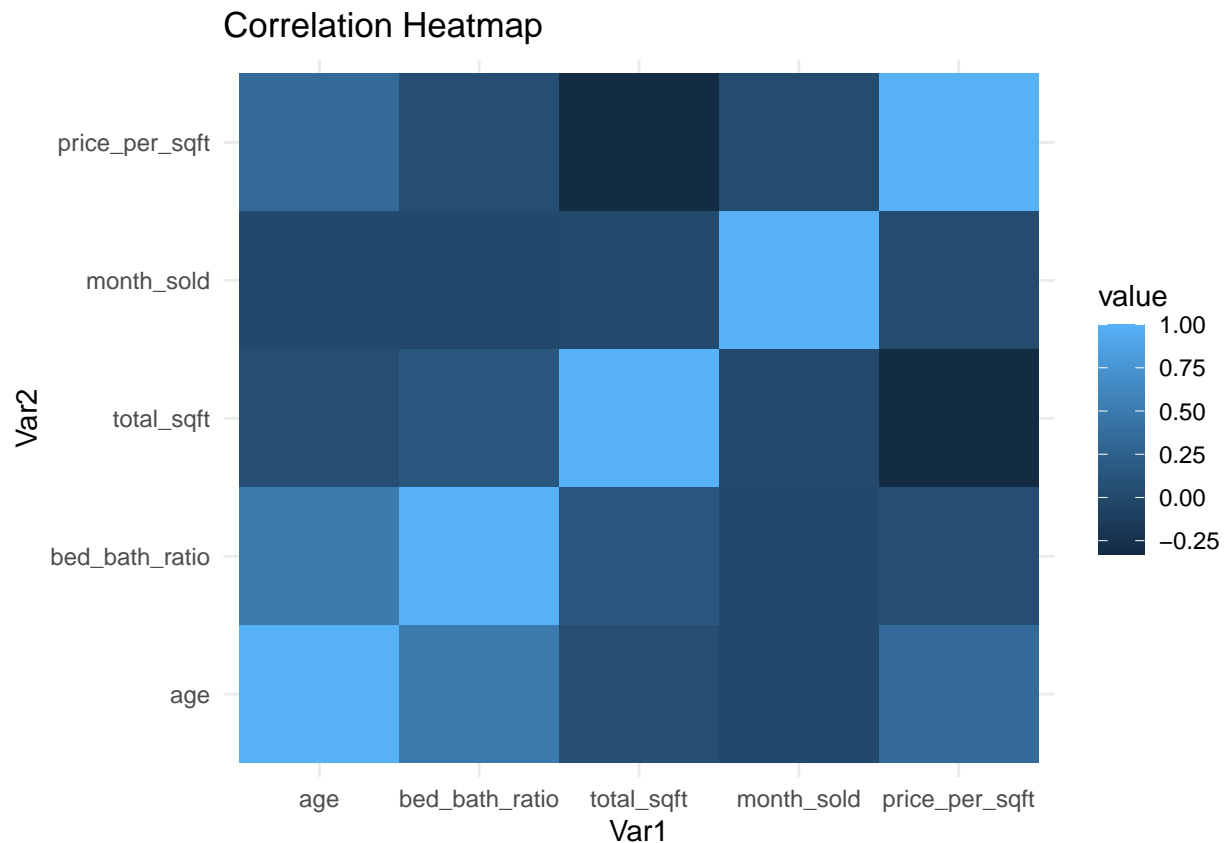
```
##      age      bed_bath_ratio      total_sqft      month_sold
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.0000   Min.   :0.0000
## 1st Qu.:0.1652   1st Qu.:0.09722   1st Qu.:0.2501   1st Qu.:1.0000
## Median :0.3826   Median :0.14881   Median :0.3740   Median :1.0000
## Mean   :0.3892   Mean   :0.16406   Mean   :0.3706   Mean   :0.7777
## 3rd Qu.:0.5565   3rd Qu.:0.18750   3rd Qu.:0.4773   3rd Qu.:1.0000
## Max.   :1.0000   Max.   :1.00000   Max.   :1.0000   Max.   :1.0000
## price_per_sqft
##  Min.   :0.0000
## 1st Qu.:0.1261
## Median :0.2208
## Mean   :0.2404
## 3rd Qu.:0.3223
## Max.   :1.0000
```

```
#calculate correlation matrix
correlation_matrix <- cor(house_data_clean[columns_to_normalize])
```

```
#create a heatmap of the correlation matrix
library(ggplot2)
library(reshape2)
```

```
#melt the correlation matrix for visualization
correlation_melted <- melt(correlation_matrix)
```

```
#heatmap
ggplot(data = correlation_melted, aes(x = Var1, y = Var2, fill = value)) +
  geom_tile() +
  theme_minimal() +
  labs(title = "Correlation Heatmap")
```



Part2

Identify categorical, ordinal and numerical variables

```
#display the structure of the dataframe
str(house_data_clean)
```

```
## 'data.frame':    11263 obs. of  27 variables:
## $ id             : num  7.13e+09 5.63e+09 2.49e+09 1.95e+09 1.32e+09 ...
## $ date           : Date, format: "2014-10-13" "2015-02-25" ...
## $ price          : num  221900 180000 604000 510000 257500 ...
## $ bedrooms       : num  3 2 4 3 3 3 3 3 2 3 ...
## $ bathrooms      : num  1 1 3 2 2.25 1.5 1 2.5 1 1.75 ...
## $ sqft_living     : num  1180 770 1960 1680 1715 ...
## $ sqft_lot        : num  5650 10000 5000 8080 6819 ...
## $ floors          : num  1 1 1 1 2 1 1 2 1 1 ...
## $ waterfront      : int   0 0 0 0 0 0 0 0 0 0 ...
## $ view            : int   0 0 0 0 0 0 0 0 0 0 ...
## $ condition       : int   3 3 5 3 3 3 3 3 4 4 ...
## $ grade           : int   7 6 7 8 7 7 7 7 7 7 ...
## $ sqft_above      : int  1180 770 1050 1680 1715 1060 1050 1890 860 1370 ...
## $ sqft_basement   : int    0 0 910 0 0 0 730 0 300 0 ...
## $ yr_built        : int  1955 1933 1965 1987 1995 1963 1960 2003 1942 1977 ...
## $ yr_renovated    : int    0 0 0 0 0 0 0 0 0 0 ...
## $ zipcode         : int  98178 98028 98136 98074 98003 98198 98146 98038 98115 98074 ...
## $ lat             : num   47.5 47.7 47.5 47.6 47.3 ...
```

```
## $ long          : num  -122 -122 -122 -122 -122 ...
## $ sqft_living15 : int   1340 2720 1360 1800 2238 1650 1780 2390 1330 1370 ...
## $ sqft_lot15    : int   5650 8062 5000 7503 6819 9711 8113 7570 6000 10208 ...
## $ decade_built : num   1950 1930 1960 1980 1990 1960 1960 2000 1940 1970 ...
## $ age           : num    0.522 0.713 0.435 0.243 0.174 ...
## $ bed_bath_ratio: num    0.3229 0.1875 0.0972 0.1198 0.0972 ...
## $ total_sqft    : num    0.28 0.488 0.287 0.434 0.37 ...
## $ month_sold    : num    0 1 0 1 1 1 1 1 1 0 ...
## $ price_per_sqft: num    0.1426 0.2075 0.3131 0.3066 0.0888 ...
## - attr(*, "na.action")= 'omit' Named int [1:3995] 11 13 19 24 25 27 32 33 38 39 ...
## ..- attr(*, "names")= chr [1:3995] "11" "13" "19" "24" ...
```

```
categorical_vars <- c("waterfront", "view", "zipcode")
```

```
# Checking the unique values in each categorical variable
sapply(house_data_clean[categorical_vars], function(x) unique(x))
```

```
## $waterfront
## [1] 0
##
## $view
## [1] 0
##
## $zipcode
## [1] 98178 98028 98136 98074 98003 98198 98146 98038 98115 98107 98019 98103
## [13] 98133 98092 98002 98112 98052 98027 98117 98058 98056 98119 98007 98148
## [25] 98105 98042 98166 98122 98144 98001 98034 98125 98008 98116 98118 98059
## [37] 98023 98102 98168 98029 98006 98109 98177 98030 98126 98040 98155 98053
## [49] 98031 98024 98108 98106 98032 98072 98033 98055 98011 98005 98075 98188
## [61] 98004 98010 98199 98022 98065 98077 98014 98070 98039
```

```
categorical_vars
```

```
## [1] "waterfront" "view"          "zipcode"
```

Identifying N

```
ordinal_vars <- c("condition", "grade")
```

```
#checking the unique values in each ordinal variable
sapply(house_data_clean[ordinal_vars], function(x) unique(x))
```

```
##      condition grade
## [1,]         3     7
## [2,]         5     6
## [3,]         4     8
## [4,]         2     9
```

```
numerical_vars <- sapply(house_data_clean, is.numeric)
numerical_vars
```

```
##          id          date          price          bedrooms          bathrooms
##          TRUE          FALSE          TRUE          TRUE          TRUE
##    sqft_living    sqft_lot    floors    waterfront    view
##          TRUE          TRUE          TRUE          TRUE          TRUE
##    condition    grade    sqft_above    sqft_basement    yr_built
##          TRUE          TRUE          TRUE          TRUE          TRUE
##    yr_renovated    zipcode    lat    long    sqft_living15
##          TRUE          TRUE          TRUE          TRUE          TRUE
##    sqft_lot15    decade_built    age    bed_bath_ratio    total_sqft
##          TRUE          TRUE          TRUE          TRUE          TRUE
##    month_sold    price_per_sqft
##          TRUE          TRUE
```

Measures of centrality and distribution

```
numerical_vars <- sapply(house_data_clean, is.numeric)
numerical_vars <- names(numerical_vars[numerical_vars])

#summary statistics for numerical variables
summary_stats <- summary(house_data_clean[numerical_vars])

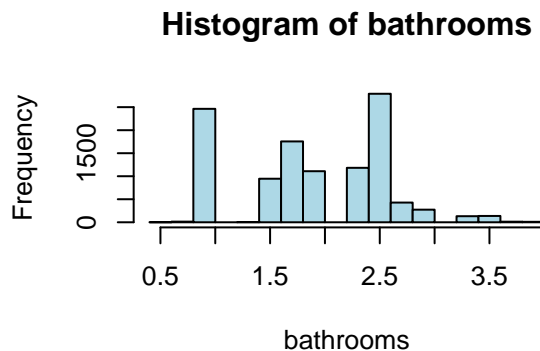
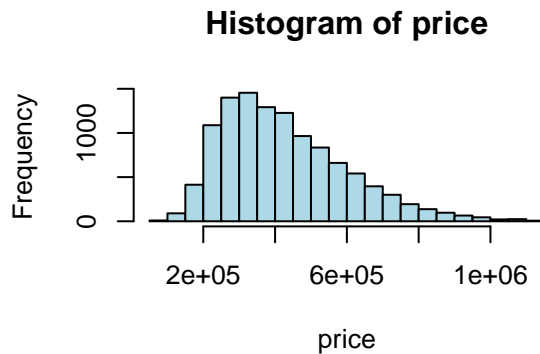
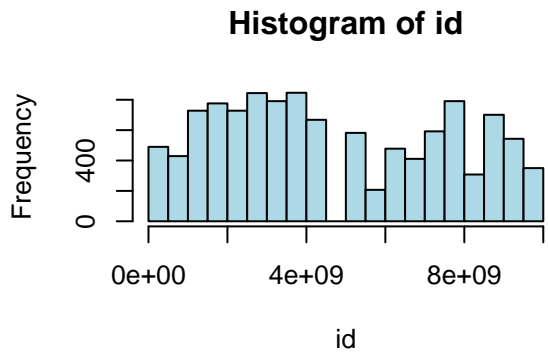
print(summary_stats)
```

```
##          id          price          bedrooms          bathrooms
## Min.    :2.800e+06    Min.    : 82000    Min.    :2.000    Min.    :0.500
## 1st Qu.:2.290e+09    1st Qu.: 295000    1st Qu.:3.000    1st Qu.:1.500
## Median :3.999e+09    Median : 399500    Median :3.000    Median :2.000
## Mean   :4.684e+09    Mean   : 428819    Mean   :3.221    Mean   :1.938
## 3rd Qu.:7.338e+09    3rd Qu.: 535000    3rd Qu.:4.000    3rd Qu.:2.500
## Max.   :9.895e+09    Max.   :1125000    Max.   :5.000    Max.   :4.000
##    sqft_living    sqft_lot    floors    waterfront    view
## Min.    : 560    Min.    : 520    Min.    :1.000    Min.    :0    Min.    :0
## 1st Qu.:1320    1st Qu.: 4560    1st Qu.:1.000    1st Qu.:0    1st Qu.:0
## Median :1690    Median : 6976    Median :1.000    Median :0    Median :0
## Mean   :1746    Mean   : 6801    Mean   :1.455    Mean   :0    Mean   :0
## 3rd Qu.:2130    3rd Qu.: 8750    3rd Qu.:2.000    3rd Qu.:0    3rd Qu.:0
## Max.   :3930    Max.   :17859    Max.   :3.500    Max.   :0    Max.   :0
##    condition    grade    sqft_above    sqft_basement    yr_built
## Min.    :2.000    Min.    :6.000    Min.    : 480    Min.    : 0.0    Min.    :1900
## 1st Qu.:3.000    1st Qu.:7.000    1st Qu.:1120    1st Qu.: 0.0    1st Qu.:1951
## Median :3.000    Median :7.000    Median :1400    Median : 0.0    Median :1971
## Mean   :3.433    Mean   :7.328    Mean   :1524    Mean   :221.6    Mean   :1970
## 3rd Qu.:4.000    3rd Qu.:8.000    3rd Qu.:1830    3rd Qu.:440.0    3rd Qu.:1996
## Max.   :5.000    Max.   :9.000    Max.   :3090    Max.   :1200.0    Max.   :2015
##    yr_renovated    zipcode    lat    long    sqft_living15
## Min.    :0    Min.    :98001    Min.    :47.19    Min.    : -122.5    Min.    : 620
## 1st Qu.:0    1st Qu.:98033    1st Qu.:47.46    1st Qu.: -122.3    1st Qu.:1410
## Median :0    Median :98074    Median :47.57    Median : -122.3    Median :1670
## Mean   :0    Mean   :98082    Mean   :47.56    Mean   : -122.2    Mean   :1741
## 3rd Qu.:0    3rd Qu.:98119    3rd Qu.:47.68    3rd Qu.: -122.2    3rd Qu.:2030
## Max.   :0    Max.   :98199    Max.   :47.78    Max.   : -121.9    Max.   :2990
##    sqft_lot15    decade_built    age    bed_bath_ratio
## Min.    : 659    Min.    :1900    Min.    :0.0000    Min.    :0.00000
```

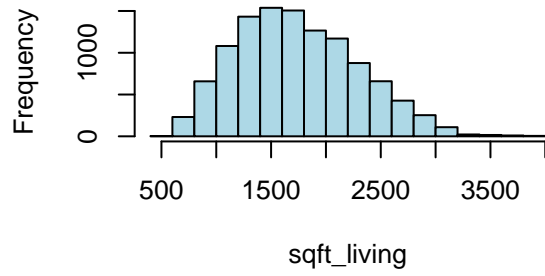
```
## 1st Qu.: 4668    1st Qu.:1950    1st Qu.:0.1652    1st Qu.:0.09722
## Median : 7102    Median :1970    Median :0.3826    Median :0.14881
## Mean   : 6698    Mean   :1966    Mean   :0.3892    Mean   :0.16406
## 3rd Qu.: 8535    3rd Qu.:1990    3rd Qu.:0.5565    3rd Qu.:0.18750
## Max.   :14482    Max.   :2010    Max.   :1.0000    Max.   :1.00000
## total_sqft      month_sold      price_per_sqft
## Min.   :0.0000    Min.   :0.0000    Min.   :0.0000
## 1st Qu.:0.2501    1st Qu.:1.0000    1st Qu.:0.1261
## Median :0.3740    Median :1.0000    Median :0.2208
## Mean   :0.3706    Mean   :0.7777    Mean   :0.2404
## 3rd Qu.:0.4773    3rd Qu.:1.0000    3rd Qu.:0.3223
## Max.   :1.0000    Max.   :1.0000    Max.   :1.0000
```

```
#histograms for numerical variables
par(mfrow = c(2, 2)) # Set up a 2x2 grid for subplots

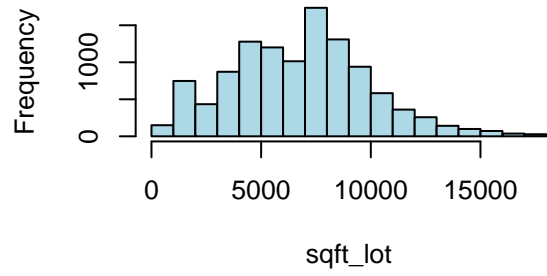
for (var in numerical_vars) {
  hist(house_data_clean[[var]], main = paste("Histogram of", var),
       xlab = var, col = "lightblue", border = "black")
}
```



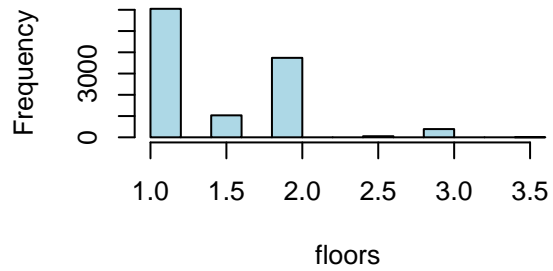
Histogram of sqft_living



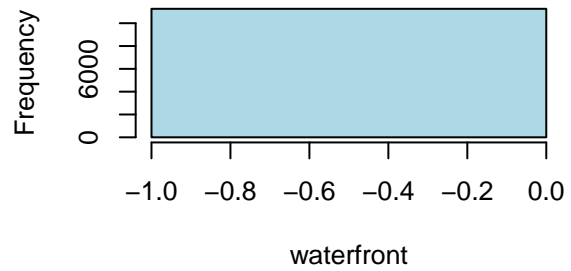
Histogram of sqft_lot



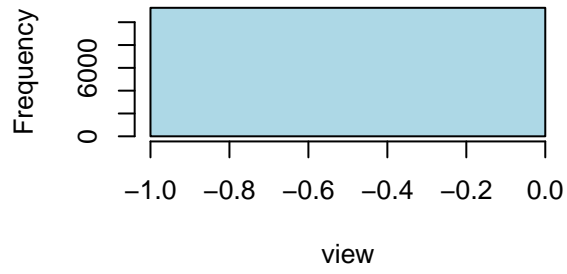
Histogram of floors



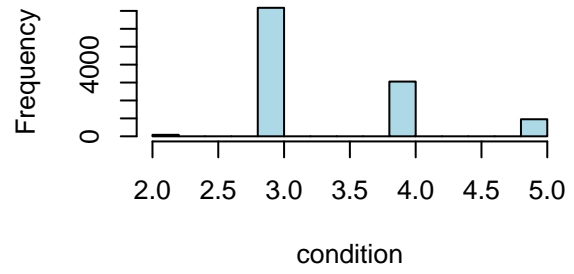
Histogram of waterfront



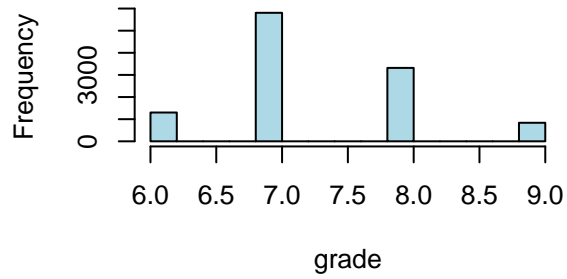
Histogram of view



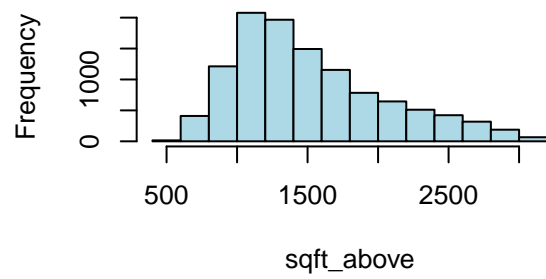
Histogram of condition



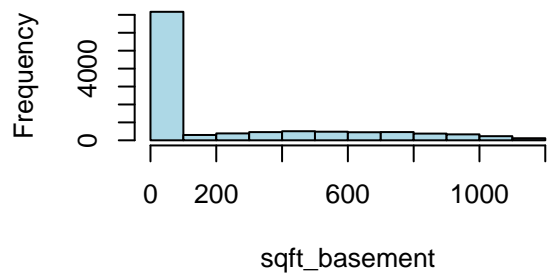
Histogram of grade



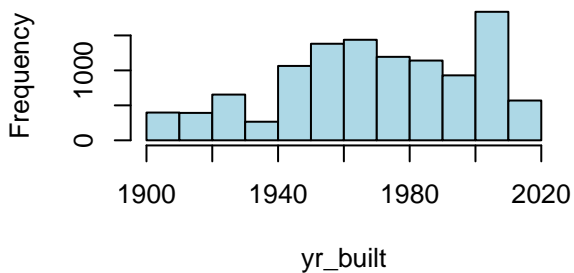
Histogram of sqft_above



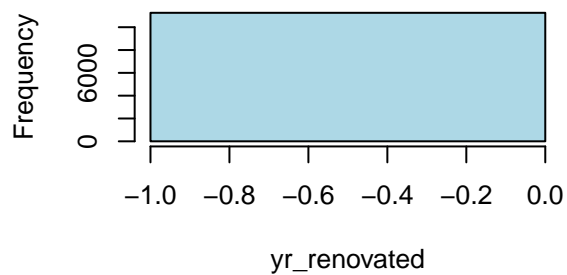
Histogram of sqft_basement



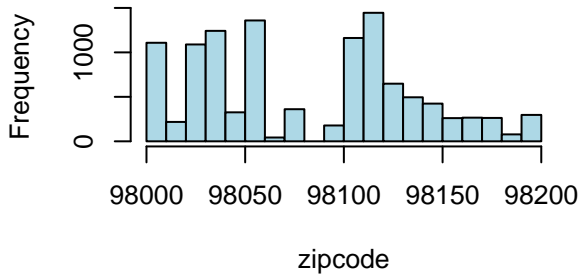
Histogram of yr_built



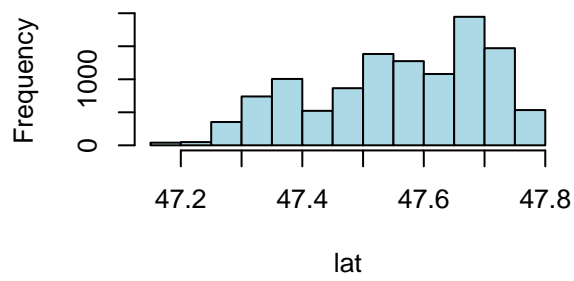
Histogram of yr_renovated



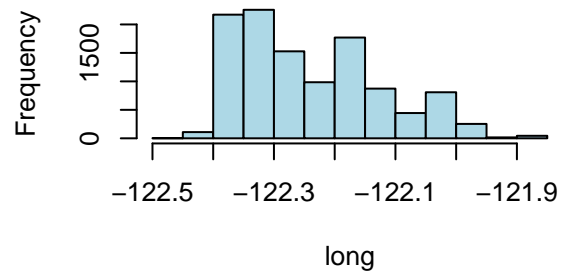
Histogram of zipcode



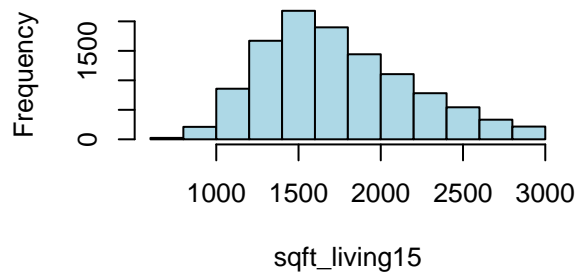
Histogram of lat



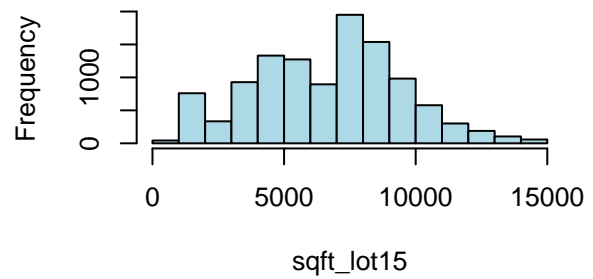
Histogram of long

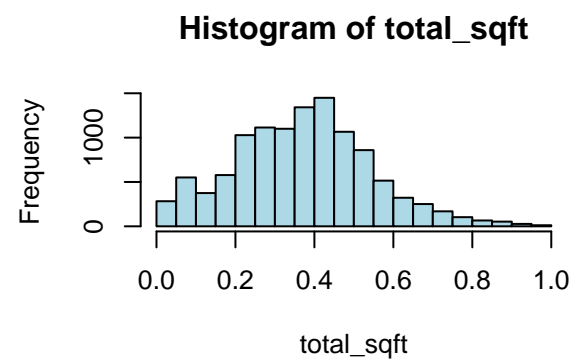
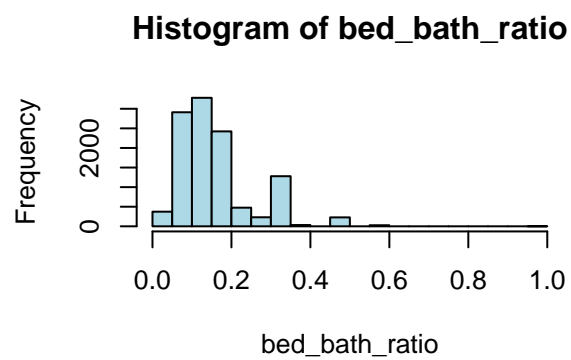
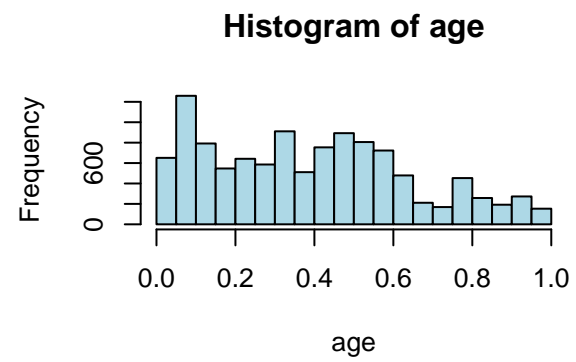
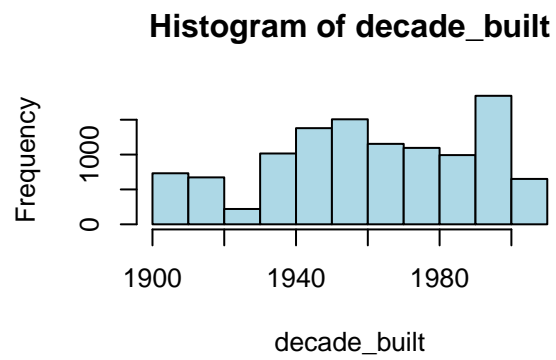


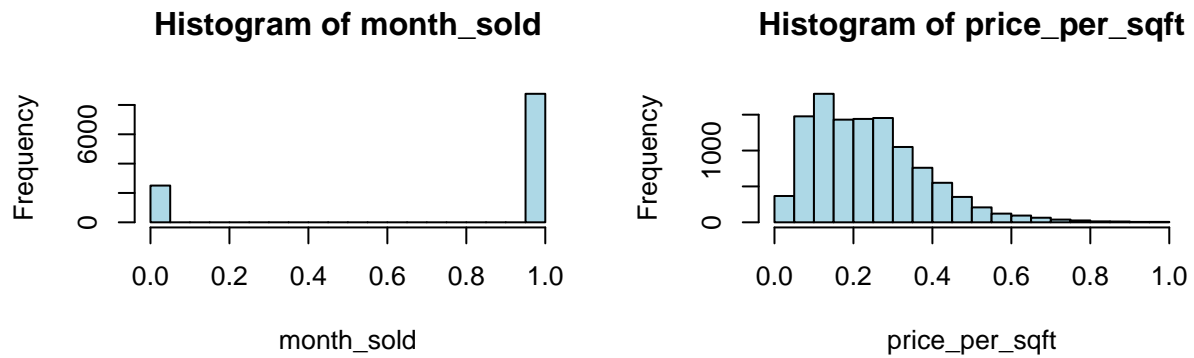
Histogram of sqft_living15



Histogram of sqft_lot15







```
library(caret)

nearZeroVarCols <- nearZeroVar(house_data_clean, saveMetrics = TRUE)

#Extract the names of near-zero variance columns
nearZeroVarNames <- names(house_data_clean)[nearZeroVarCols$nzv]

#Exclude near-zero variance variables from the dataset
filtered_data <- house_data_clean[, !names(house_data_clean) %in% nearZeroVarNames]

correlation_matrix_filtered <- cor(filtered_data[, sapply
                                   (filtered_data, is.numeric)])

#Visualize the correlation matrix using a heatmap
library(corrplot)

## corrplot 0.92 loaded

corrplot(correlation_matrix_filtered, method = "color")
```



```
#correlations with the dependent variable
cor_with_dependent <- cor(filtered_data$price, filtered_data[, supply
                                                                    (filtered_data, is.numeric)])
```

```
# Print correlations with the dependent variable
print(cor_with_dependent)
```

```
##           id price  bedrooms bathrooms sqft_living  sqft_lot   floors
## [1,] 0.04447945    1 0.2029717 0.3140286 0.4748575 -0.1148228 0.2124237
##      condition   grade sqft_above   yr_built   zipcode     lat
## [1,] 0.1089844 0.4956905 0.3563547 -0.06116387 0.02368126 0.5065564
##           long sqft_living15 sqft_lot15 decade_built    age
## [1,] -0.007718589    0.3928909 -0.1262258 -0.05935805 0.06116387
##      bed_bath_ratio total_sqft month_sold price_per_sqft
## [1,]   -0.1882257 -0.0306033 0.02300534    0.6038503
```

```
#Check for multicollinearity using VIF (Variance Inflation Factor)
library(car)
```

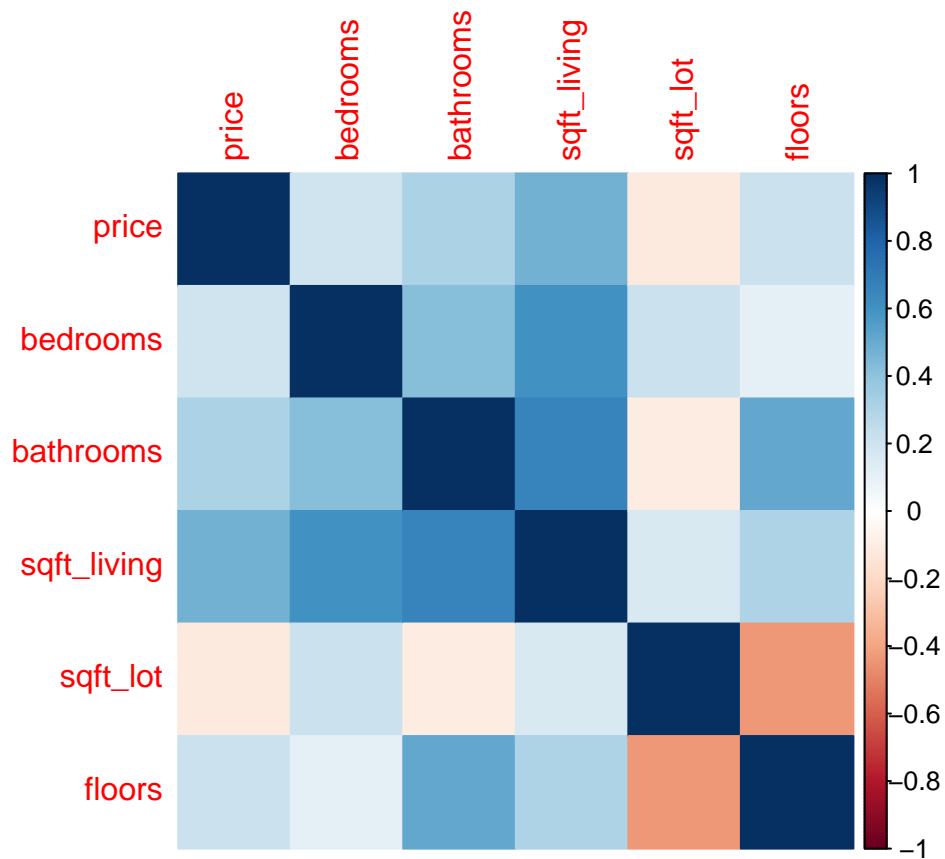
```
## Loading required package: carData
```

```
vif_model <- lm(price ~ floors + condition + grade + lat + long +
                sqft_living15 + sqft_lot15 + decade_built + bed_bath_ratio
                + total_sqft + price_per_sqft, data = filtered_data)
```

```
vif_values <- vif(vif_model)
print(vif_values)
```

```
##      floors      condition      grade      lat      long
##      2.052377      1.231731      2.111340      1.477706      1.443766
## sqft_living15 sqft_lot15 decade_built bed_bath_ratio total_sqft
##      1.826544      4.597014      2.593980      1.464198      4.701616
## price_per_sqft
##      1.818936
```

```
#Correlation heatmap
library(corrplot)
correlation_matrix <- cor(filtered_data[, c("price", "bedrooms", "bathrooms",
                                             "sqft_living", "sqft_lot", "floors")])
corrplot(correlation_matrix, method = "color")
```



For our goal of predicting house prices, we'll use a supervised learning approach. In this setup:

Dependent Variable (Predicting): price Independent Variables (Factors Influencing Price): bedrooms, bathrooms, sqft_living, sqft_lot, floors, condition, grade, lat, long, sqft_living15, sqft_lot15, decade_built, age, bed_bath_ratio, total_sqft, price_per_sqft These variables are used to train a model, and the model then predicts house prices based on new input data. This process is the essence of supervised learning.

```
# Split the data into training and testing sets
set.seed(123)
```



```

sample_index <- sample(1:nrow(filtered_data), 0.7 * nrow(filtered_data))
train_data <- filtered_data[sample_index, ]
test_data <- filtered_data[-sample_index, ]

# Build the model on the training set
final_model <- lm(price ~ bedrooms + bathrooms + floors + condition + grade
                  + lat + long + sqft_living15 + sqft_lot15 + decade_built
                  + bed_bath_ratio + total_sqft + price_per_sqft,
                  data = train_data)

# Make predictions on the test set
predictions <- predict(final_model, newdata = test_data)

# Evaluate model performance
# Mean Squared Error (MSE), Mean Absolute Error (MAE), R-squared
mse <- mean((test_data$price - predictions)^2)
mae <- mean(abs(test_data$price - predictions))
rsquared <- 1 - (sum((test_data$price - predictions)^2) /
               sum((test_data$price - mean(test_data$price))^2))

cat("Mean Squared Error (MSE):", mse, "\n")

```

```
## Mean Squared Error (MSE): 6235040050
```

```
cat("Mean Absolute Error (MAE):", mae, "\n")
```

```
## Mean Absolute Error (MAE): 59109.51
```

```
cat("R-squared:", rsquared, "\n")
```

```
## R-squared: 0.7967922
```