DS311 - EDA Lab Assignment

William Lin, Norman Lo

2023-11-22

Explore the Ames Housing Dataset with R

The Ames Housing dataset contains information about home sales in Ames, Iowa between 2006 and 2010.

1. Load the Data

```
df <- read.csv("/Users/linw/Desktop/DS311-Technologies-in-Data-Analytic-FA23/Week_09_Exploratory_Data_A
df <- df[, -1] # Remove index column
```

The following code checks the dimensions of the data.

```
dim(df)
```

```
## [1] 1460   80
```

There are 1460 rows (homes) and 80 columns (variables) in the dataset.

Inspect the contents of the dataframe:

```
df[1:40, 1:8] # First 40 observations and first 8 variables
```

```
##    MSSubClass MSZoning LotFrontage LotArea Street Alley LotShape LandContour
## 1          60       RL          65    8450   Pave  <NA>      Reg         Lvl
## 2          20       RL          80    9600   Pave  <NA>      Reg         Lvl
## 3          60       RL          68   11250   Pave  <NA>      IR1         Lvl
## 4          70       RL          60    9550   Pave  <NA>      IR1         Lvl
## 5          60       RL          84   14260   Pave  <NA>      IR1         Lvl
## 6          50       RL          85   14115   Pave  <NA>      IR1         Lvl
## 7          20       RL          75   10084   Pave  <NA>      Reg         Lvl
## 8          60       RL          NA   10382   Pave  <NA>      IR1         Lvl
## 9          50       RM          51    6120   Pave  <NA>      Reg         Lvl
## 10        190       RL          50    7420   Pave  <NA>      Reg         Lvl
## 11         20       RL          70   11200   Pave  <NA>      Reg         Lvl
## 12         60       RL          85   11924   Pave  <NA>      IR1         Lvl
## 13         20       RL          NA   12968   Pave  <NA>      IR2         Lvl
## 14         20       RL          91   10652   Pave  <NA>      IR1         Lvl
## 15         20       RL          NA   10920   Pave  <NA>      IR1         Lvl
## 16         45       RM          51    6120   Pave  <NA>      Reg         Lvl
```

```
## 17         20       RL          NA   11241   Pave  <NA>      IR1         Lvl
## 18         90       RL          72   10791   Pave  <NA>      Reg         Lvl
## 19         20       RL          66   13695   Pave  <NA>      Reg         Lvl
## 20         20       RL          70    7560   Pave  <NA>      Reg         Lvl
## 21         60       RL         101   14215   Pave  <NA>      IR1         Lvl
## 22         45       RM          57    7449   Pave  Grvl      Reg         Bnk
## 23         20       RL          75    9742   Pave  <NA>      Reg         Lvl
## 24        120       RM          44    4224   Pave  <NA>      Reg         Lvl
## 25         20       RL          NA    8246   Pave  <NA>      IR1         Lvl
## 26         20       RL         110   14230   Pave  <NA>      Reg         Lvl
## 27         20       RL          60    7200   Pave  <NA>      Reg         Lvl
## 28         20       RL          98   11478   Pave  <NA>      Reg         Lvl
## 29         20       RL          47   16321   Pave  <NA>      IR1         Lvl
## 30         30       RM          60    6324   Pave  <NA>      IR1         Lvl
## 31         70  C (all)          50    8500   Pave  Pave      Reg         Lvl
## 32         20       RL          NA    8544   Pave  <NA>      IR1         Lvl
## 33         20       RL          85   11049   Pave  <NA>      Reg         Lvl
## 34         20       RL          70   10552   Pave  <NA>      IR1         Lvl
## 35        120       RL          60    7313   Pave  <NA>      Reg         Lvl
## 36         60       RL         108   13418   Pave  <NA>      Reg         Lvl
## 37         20       RL         112   10859   Pave  <NA>      Reg         Lvl
## 38         20       RL          74    8532   Pave  <NA>      Reg         Lvl
## 39         20       RL          68    7922   Pave  <NA>      Reg         Lvl
## 40         90       RL          65    6040   Pave  <NA>      Reg         Lvl
```

```
str(df)
```

```
## 'data.frame':    1460 obs. of  80 variables:
##  $ MSSubClass   : int  60 20 60 70 60 50 20 60 50 190 ...
##  $ MSZoning     : chr  "RL" "RL" "RL" "RL" ...
##  $ LotFrontage  : int  65 80 68 60 84 85 75 NA 51 50 ...
##  $ LotArea      : int  8450 9600 11250 9550 14260 14115 10084 10382 6120 7420 ...
##  $ Street       : chr  "Pave" "Pave" "Pave" "Pave" ...
##  $ Alley        : chr  NA NA NA NA ...
##  $ LotShape     : chr  "Reg" "Reg" "IR1" "IR1" ...
##  $ LandContour  : chr  "Lvl" "Lvl" "Lvl" "Lvl" ...
##  $ Utilities    : chr  "AllPub" "AllPub" "AllPub" "AllPub" ...
##  $ LotConfig    : chr  "Inside" "FR2" "Inside" "Corner" ...
##  $ LandSlope    : chr  "Gtl" "Gtl" "Gtl" "Gtl" ...
##  $ Neighborhood : chr  "CollgCr" "Veenker" "CollgCr" "Crawfor" ...
##  $ Condition1   : chr  "Norm" "Feedr" "Norm" "Norm" ...
##  $ Condition2   : chr  "Norm" "Norm" "Norm" "Norm" ...
##  $ BldgType     : chr  "1Fam" "1Fam" "1Fam" "1Fam" ...
##  $ HouseStyle   : chr  "2Story" "1Story" "2Story" "2Story" ...
##  $ OverallQual  : int  7 6 7 7 8 5 8 7 7 5 ...
##  $ OverallCond  : int  5 8 5 5 5 5 5 6 5 6 ...
##  $ YearBuilt    : int  2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 ...
##  $ YearRemodAdd : int  2003 1976 2002 1970 2000 1995 2005 1973 1950 1950 ...
##  $ RoofStyle    : chr  "Gable" "Gable" "Gable" "Gable" ...
##  $ RoofMatl     : chr  "CompShg" "CompShg" "CompShg" "CompShg" ...
##  $ Exterior1st  : chr  "VinylSd" "MetalSd" "VinylSd" "Wd Sdng" ...
##  $ Exterior2nd  : chr  "VinylSd" "MetalSd" "VinylSd" "Wd Shng" ...
##  $ MasVnrType   : chr  "BrkFace" "None" "BrkFace" "None" ...
##  $ MasVnrArea   : int  196 0 162 0 350 0 186 240 0 0 ...
```

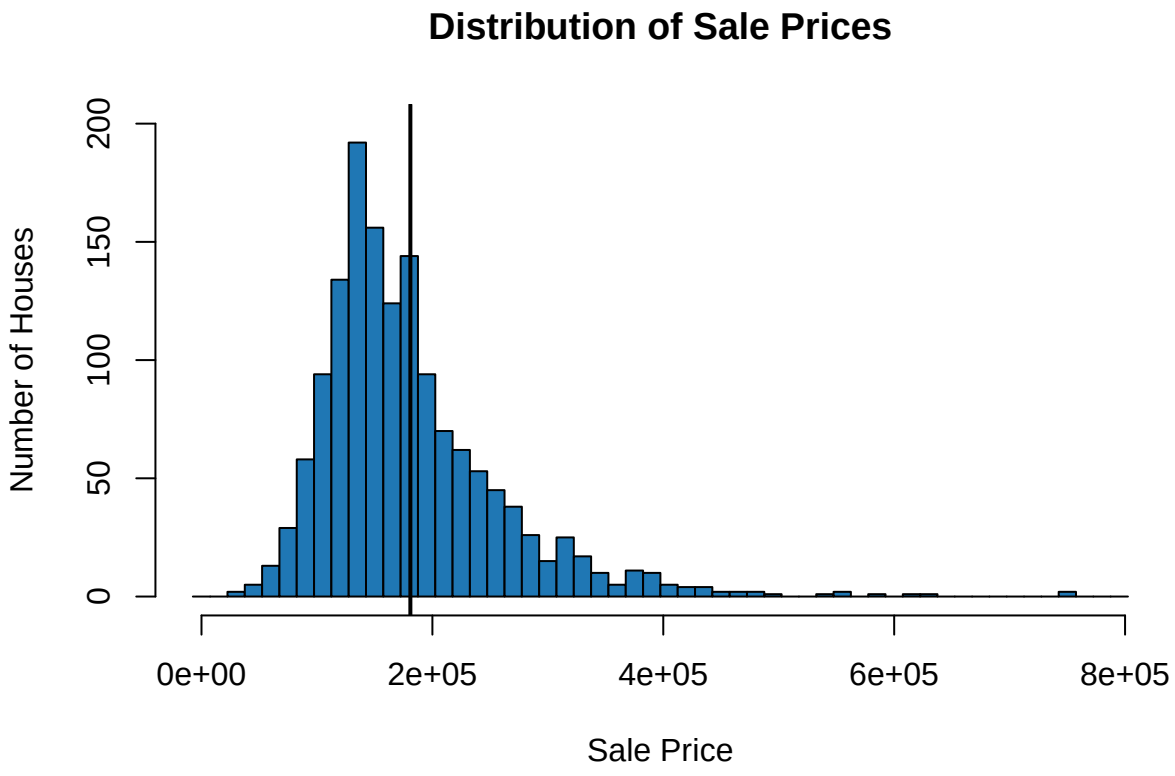
```
##  $ ExterQual    : chr  "Gd" "TA" "Gd" "TA" ...
##  $ ExterCond    : chr  "TA" "TA" "TA" "TA" ...
##  $ Foundation   : chr  "PConc" "CBlock" "PConc" "BrkTil" ...
##  $ BsmtQual     : chr  "Gd" "Gd" "Gd" "TA" ...
##  $ BsmtCond     : chr  "TA" "TA" "TA" "Gd" ...
##  $ BsmtExposure : chr  "No" "Gd" "Mn" "No" ...
##  $ BsmtFinType1 : chr  "GLQ" "ALQ" "GLQ" "ALQ" ...
##  $ BsmtFinSF1   : int  706 978 486 216 655 732 1369 859 0 851 ...
##  $ BsmtFinType2 : chr  "Unf" "Unf" "Unf" "Unf" ...
##  $ BsmtFinSF2   : int  0 0 0 0 0 0 0 32 0 0 ...
##  $ BsmtUnfSF    : int  150 284 434 540 490 64 317 216 952 140 ...
##  $ TotalBsmtSF  : int  856 1262 920 756 1145 796 1686 1107 952 991 ...
##  $ Heating      : chr  "GasA" "GasA" "GasA" "GasA" ...
##  $ HeatingQC    : chr  "Ex" "Ex" "Ex" "Gd" ...
##  $ CentralAir   : chr  "Y" "Y" "Y" "Y" ...
##  $ Electrical   : chr  "SBrkr" "SBrkr" "SBrkr" "SBrkr" ...
##  $ X1stFlrSF    : int  856 1262 920 961 1145 796 1694 1107 1022 1077 ...
##  $ X2ndFlrSF    : int  854 0 866 756 1053 566 0 983 752 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  1710 1262 1786 1717 2198 1362 1694 2090 1774 1077 ...
##  $ BsmtFullBath : int  1 0 1 1 1 1 1 1 0 1 ...
##  $ BsmtHalfBath : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  2 2 2 1 2 1 2 2 2 1 ...
##  $ HalfBath     : int  1 0 1 0 1 1 0 1 0 0 ...
##  $ BedroomAbvGr : int  3 3 3 3 4 1 3 3 2 2 ...
##  $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 2 2 ...
##  $ KitchenQual  : chr  "Gd" "TA" "Gd" "Gd" ...
##  $ TotRmsAbvGrd : int  8 6 6 7 9 5 7 7 8 5 ...
##  $ Functional   : chr  "Typ" "Typ" "Typ" "Typ" ...
##  $ Fireplaces   : int  0 1 1 1 1 0 1 2 2 2 ...
##  $ FireplaceQu  : chr  NA "TA" "TA" "Gd" ...
##  $ GarageType   : chr  "Attchd" "Attchd" "Attchd" "Detchd" ...
##  $ GarageYrBlt  : int  2003 1976 2001 1998 2000 1993 2004 1973 1931 1939 ...
##  $ GarageFinish : chr  "RFn" "RFn" "RFn" "Unf" ...
##  $ GarageCars   : int  2 2 2 3 3 2 2 2 2 1 ...
##  $ GarageArea   : int  548 460 608 642 836 480 636 484 468 205 ...
##  $ GarageQual   : chr  "TA" "TA" "TA" "TA" ...
##  $ GarageCond   : chr  "TA" "TA" "TA" "TA" ...
##  $ PavedDrive   : chr  "Y" "Y" "Y" "Y" ...
##  $ WoodDeckSF   : int  0 298 0 0 192 40 255 235 90 0 ...
##  $ OpenPorchSF  : int  61 0 42 35 84 30 57 204 0 4 ...
##  $ EnclosedPorch: int  0 0 0 272 0 0 0 228 205 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 320 0 0 0 0 ...
##  $ ScreenPorch  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolQC       : chr  NA NA NA NA ...
##  $ Fence        : chr  NA NA NA NA ...
##  $ MiscFeature  : chr  NA NA NA NA ...
##  $ MiscVal      : int  0 0 0 0 0 700 0 350 0 0 ...
##  $ MoSold       : int  2 5 9 2 12 10 8 11 4 1 ...
##  $ YrSold       : int  2008 2007 2008 2006 2008 2009 2007 2009 2008 2008 ...
##  $ SaleType     : chr  "WD" "WD" "WD" "WD" ...
##  $ SaleCondition: chr  "Normal" "Normal" "Normal" "Abnorml" ...
##  $ SalePrice    : int  208500 181500 223500 140000 250000 143000 307000 200000 129900 118000 ...
```

2. Explore Data Distributions

Sale Price

Histogram for SalePrice:

```
hist(df$SalePrice, xlab = "Sale Price", col = ("#1f77b4"),
     ylab = "Number of Houses", main = "Distribution of Sale Prices",
     breaks = seq(-7500, 802500, by = 15000), ylim = c(0, 200))
abline(v = mean(df$SalePrice), col = 'black', lwd = 2)
```



Summary statistics:

```
y = df$SalePrice

text <- function(y) {
  noquote(c(paste("Median:", median(y)), paste("Mean:", mean(y)),
                paste("Standard Deviation:", sd(y))))
}

summary(y); cat(text(y), sep="\n")
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   34900  129975  163000  180921  214000  755000
```

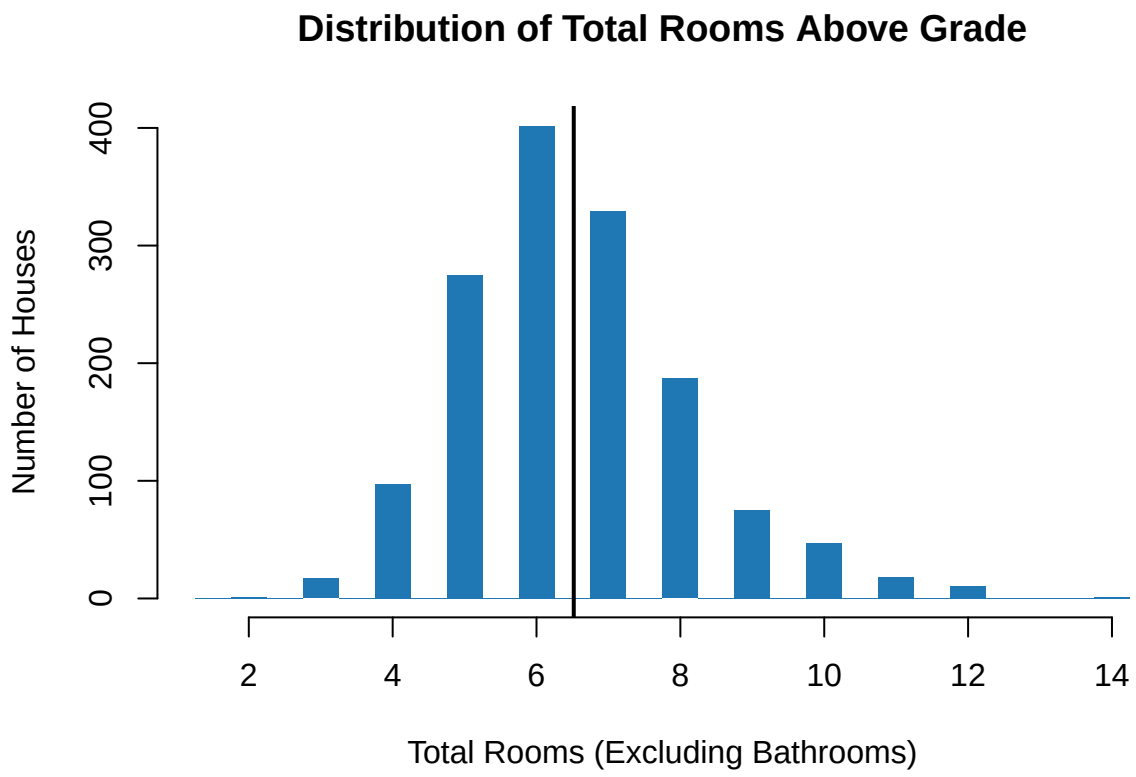
```
## Median: 163000
## Mean: 180921.195890411
## Standard Deviation: 79442.5028828866
```

The distribution of sale prices looks like a log normal distribution. Most houses in this sample are clustered around the median value of $63,000, but the higher-end homes are pulling the mean up to over $180,000.

Total Rooms Above Grade

Histogram for TotRmsAbvGrd:

```
hist(df$TotRmsAbvGrd, xlab = "Total Rooms (Excluding Bathrooms)",
     ylab = "Number of Houses", breaks = seq(1.25, 14.25, by = 0.5),
     main = "Distribution of Total Rooms Above Grade",
     col = ("#1f77b4"), border=par('bg'))
abline(v = mean(df$TotRmsAbvGrd), col = 'black', lwd = 2)
```



Summary statistics:

```
y = df$TotRmsAbvGrd
summary(y); cat(text(y), sep="\n")
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   5.000   6.000   6.518   7.000  14.000
```

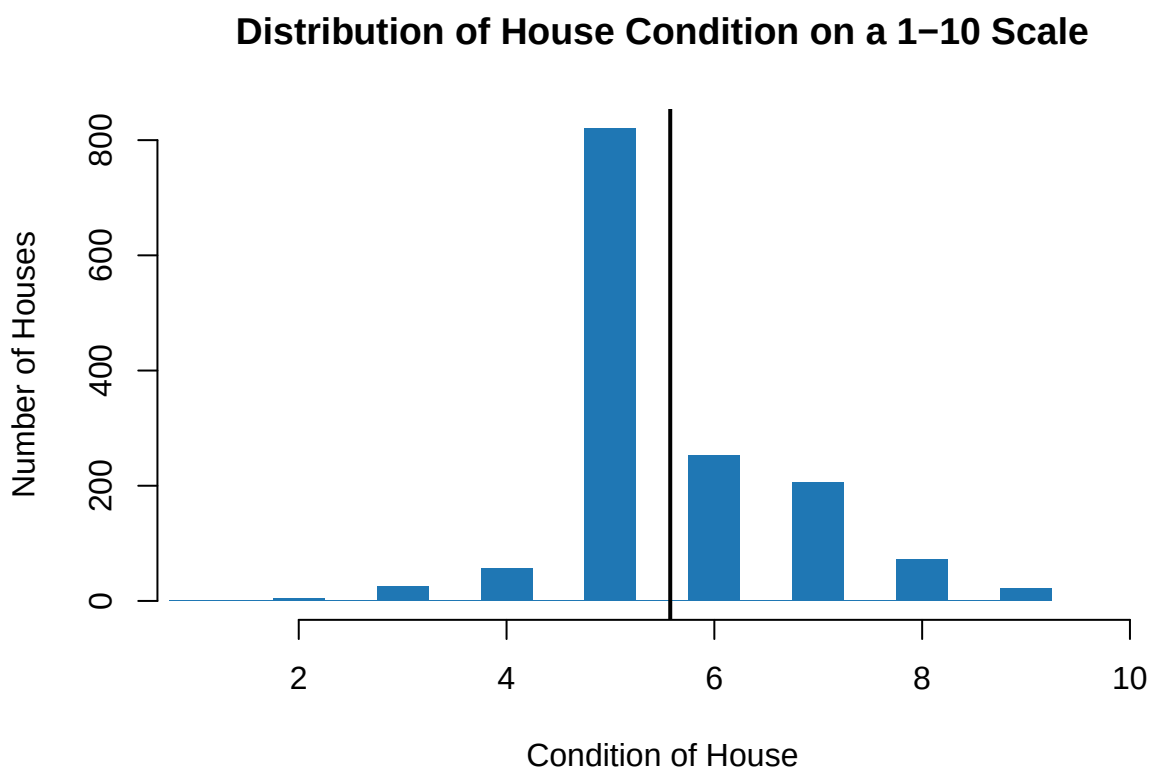
```
## Median: 6
## Mean: 6.51780821917808
## Standard Deviation: 1.62539329058405
```

The number of rooms in houses is approximately normally distributed, with a mean and median around 6 rooms. There are some houses with twice as many rooms as the average, but overall the distribution is less skewed than the sale price distribution.

Overall Condition

Histogram for OverallCond:

```
hist(df$OverallCond, xlab = "Condition of House",
     ylab = "Number of Houses", breaks = seq(0.75, 9.25, by = 0.5),
     main = "Distribution of House Condition on a 1-10 Scale",
     xlim = range(1, 10),
     col = ("#1f77b4"), border=par('bg'))
abline(v = mean(df$OverallCond), col = 'black', lwd = 2)
```



Summary statistics:

```
y = df$OverallCond
summary(y); cat(text(y), sep="\n")
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   5.000   5.000   5.575   6.000   9.000
```

```
## Median: 5
## Mean: 5.57534246575342
## Standard Deviation: 1.11279933671273
```

Most homes have a condition of 5. It seems like we should treat this as a categorical rather than numeric variable, since the difference between conditions is so abrupt.

3. Explore Differences between Subsets

One useful way to explore a categorical variable is to create subsets of the full dataset based on that categorical variable, then plot their distributions based on some other variable. Since this dataset is traditionally used for predicting the sale price of a house, let's use SalePrice as that other variable.

Create three variables, each of which represents a record-wise subset of df (meaning, it has the same columns as df, but only some of the rows).

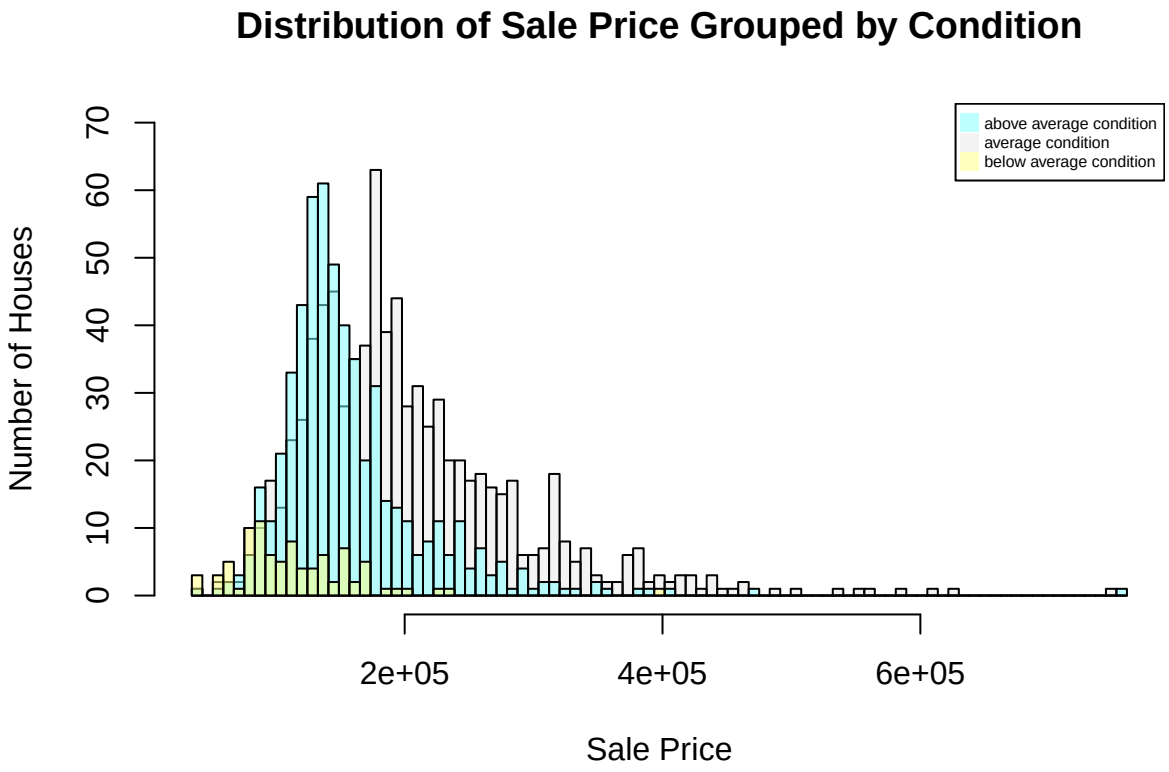
- below_average_condition: home sales where the overall condition was less than 5
- average_condition: home sales where the overall condition was exactly 5
- above_average_condition: home sales where the overall condition was greater than 5

```
below_average_condition = df[df$OverallCond < 5, ]
average_condition = df[df$OverallCond == 5, ]
above_average_condition = df[df$OverallCond > 5, ]
```

The following code will produce a plot of the distributions of sale price for each of these subsets:

```
a = below_average_condition$SalePrice
b = average_condition$SalePrice
c = above_average_condition$SalePrice

# Create colors with reduced opacity (alpha) to see overlap
c1 = rgb(255,255,127, max = 255, alpha = 128) # Yellow
c2 = rgb(216,216,216, max = 255, alpha = 77)  # Grey
c3 = rgb(127,255,255, max = 255, alpha = 128) # Cyan

# Create custom bins so all are on the same scale
breaks_by = as.integer(median(df$SalePrice) / 20)
breaks_seq = seq(min(df$SalePrice), max(df$SalePrice) + breaks_by,
                 by = breaks_by)

hgA = hist(a, breaks = breaks_seq, plot = FALSE)
hgB = hist(b, breaks = breaks_seq, plot = FALSE)
hgC = hist(c, breaks = breaks_seq, plot = FALSE)

# Plot three histograms
plot(hgB, col = c2, ylim = c(0, 70),
     xlab = "Sale Price", ylab = "Number of Houses",
     main = "Distribution of Sale Price Grouped by Condition")
plot(hgC, col = c3, add = TRUE)
plot(hgA, col = c1, add = TRUE)
legend("topright",
```

```
       legend = c("above average condition",
                  "average condition",
                  "below average condition"),
       col = c(c3, c2, c1), cex = 0.5,
       pch = 15, pt.cex = 1.3)
```



First, we note again that the majority of the houses have average condition, then about 1/3 have above average condition, then less than 10% have below average condition.

As we might expect, the average condition therefore contains houses across a broader spectrum of the sale price range than either the below-average or above-average houses.

Another unsurprising finding is that below-average condition houses have a price distribution that is much lower than average or above-average condition houses.

But what might be surprising is that above-average condition houses do not seem to have higher average sale prices than average condition houses. In fact, above-average condition houses seem more clustered around a particular price range, especially the $100,000 to $200,000 range, whereas average condition houses are more frequent above $200,000. We might want to investigate further to understand what kinds of houses are rated as above-average condition, since this goes against a standard assumption that better condition would mean higher cost.

4. Explore Correlations

To understand more about what features of these homes lead to higher sale prices, let's look at some correlations. We'll return to using the full df, rather than the subsets.

In the cell below, print out both the name of the column and the Pearson correlation for the column that is most positively correlated with SalePrice (other than SalePrice, which is perfectly correlated with itself).

We'll only check the correlations with some kind of numeric data type.

```
# Create a df that contains only numeric columns
# and doesn't include SalePrice
numeric_df = df[, unlist(lapply(df, is.numeric))]
numeric_df = numeric_df[, !names(numeric_df) %in% "SalePrice"]

# Create list of correlations
correlations = cor(numeric_df, df$SalePrice)

# Find the largest value, and the column name at that index
max_corr_value = max(correlations, na.rm = TRUE)
i = which(correlations == max_corr_value)
max_corr_column = rownames(correlations)[[i]]

noquote(c(paste("Most Positively Correlated Column:",
                max_corr_column),
          paste("Most Positively Correlated Value:",
                max_corr_value)))
```

```
## [1] Most Positively Correlated Column: OverallQual
## [2] Most Positively Correlated Value: 0.790981600583805
```

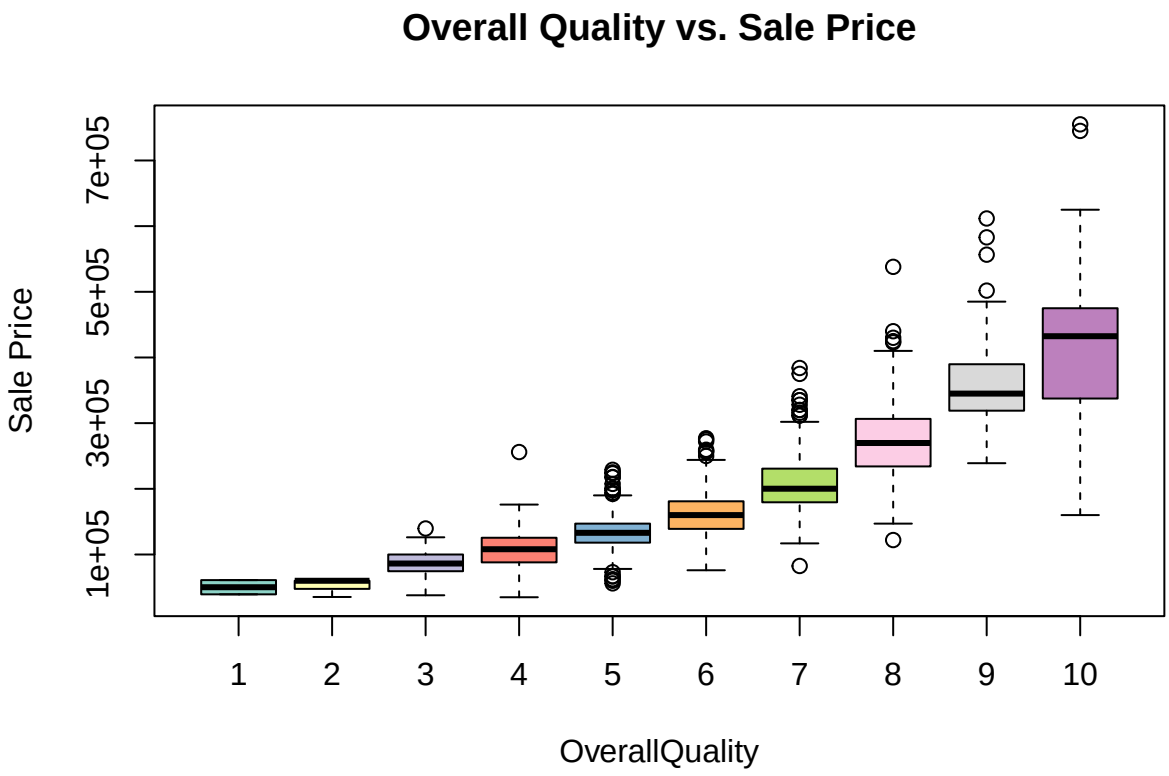
Now, find the most negatively correlated column:

```
min_corr_value = min(correlations, na.rm = TRUE)
j = which(correlations == min_corr_value)
min_corr_column = rownames(correlations)[[j]]

noquote(c(paste("Most Negatively Correlated Column:",
                min_corr_column),
          paste("Most Negative Correlated Value:",
                min_corr_value)))
```

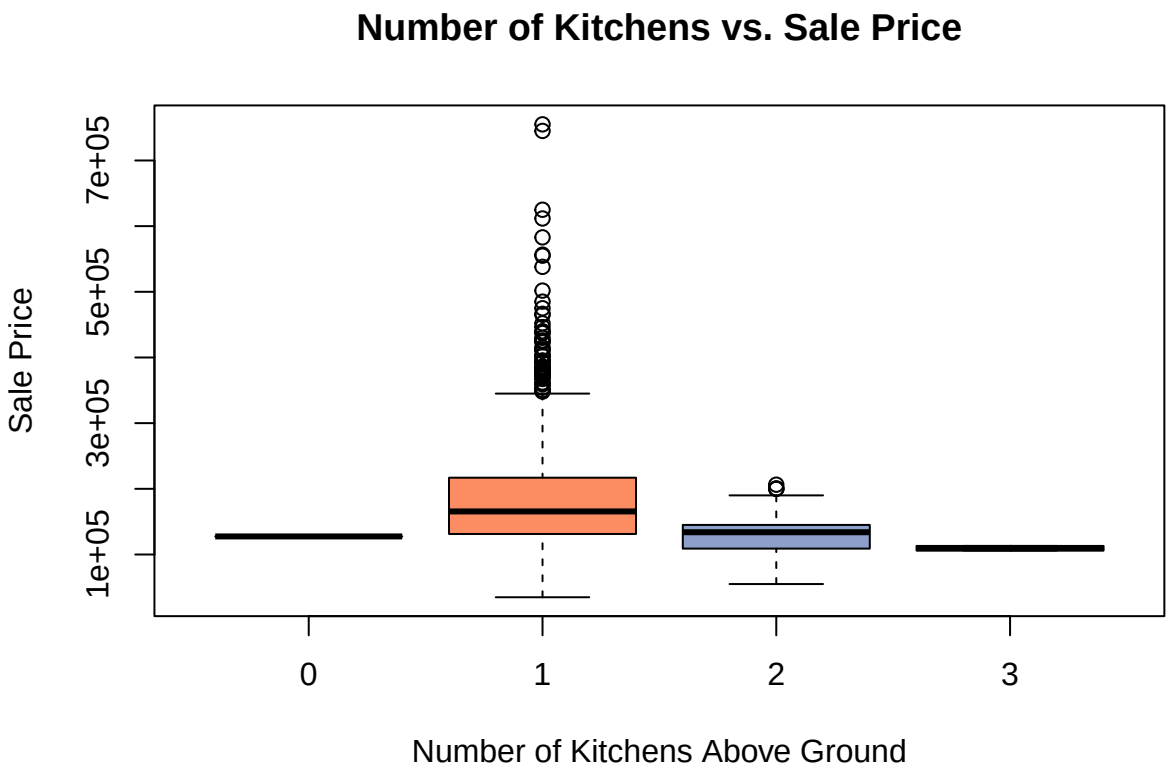
```
## [1] Most Negatively Correlated Column: KitchenAbvGr
## [2] Most Negative Correlated Value: -0.135907370842141
```

Boxplots of the relevant columns:

```
library(RColorBrewer)
colors = brewer.pal(n = length(unique(df[, max_corr_column])),
                    name = "Set3")
colors2 = brewer.pal(n = length(unique(df[, min_corr_column])),
                    name = "Set2")

boxplot(df$SalePrice ~ df[, max_corr_column],
        xlab = "OverallQuality", ylab = "Sale Price",
        main = "Overall Quality vs. Sale Price",
        col = colors)
```




```
boxplot(df$SalePrice ~ df[, min_corr_column],
        xlab = "Number of Kitchens Above Ground",
        ylab = "Sale Price", col = colors2,
        main = "Number of Kitchens vs. Sale Price")
```



The column with the highest correlation is overall quality. According to the data description:

OverallQual: Rates the overall material and finish of the house

```
10   Very Excellent
9    Excellent
8    Very Good
7    Good
6    Above Average
5    Average
4    Below Average
3    Fair
2    Poor
1    Very Poor
```

It is somewhat difficult to understand how this is different from OverallCond, which has similar values.

There is a clear positive relationship between overall quality and sale price, although it looks like potentially an exponential relationship rather than a linear one. For example, the minimum "non-outlier" (Q1 - 1.5*IQR) home with quality 10 (Very Excellent) sells for about the same price as the median home with quality 6 (Above Average).

The column with the most negative correlation is the number of kitchens above ground. According to the data description:

KitchenAbvGr: Kitchens above grade

From the plot, it is clear that almost all houses have 1 or 2 kitchens above grade, although there are some with 0 or 3.

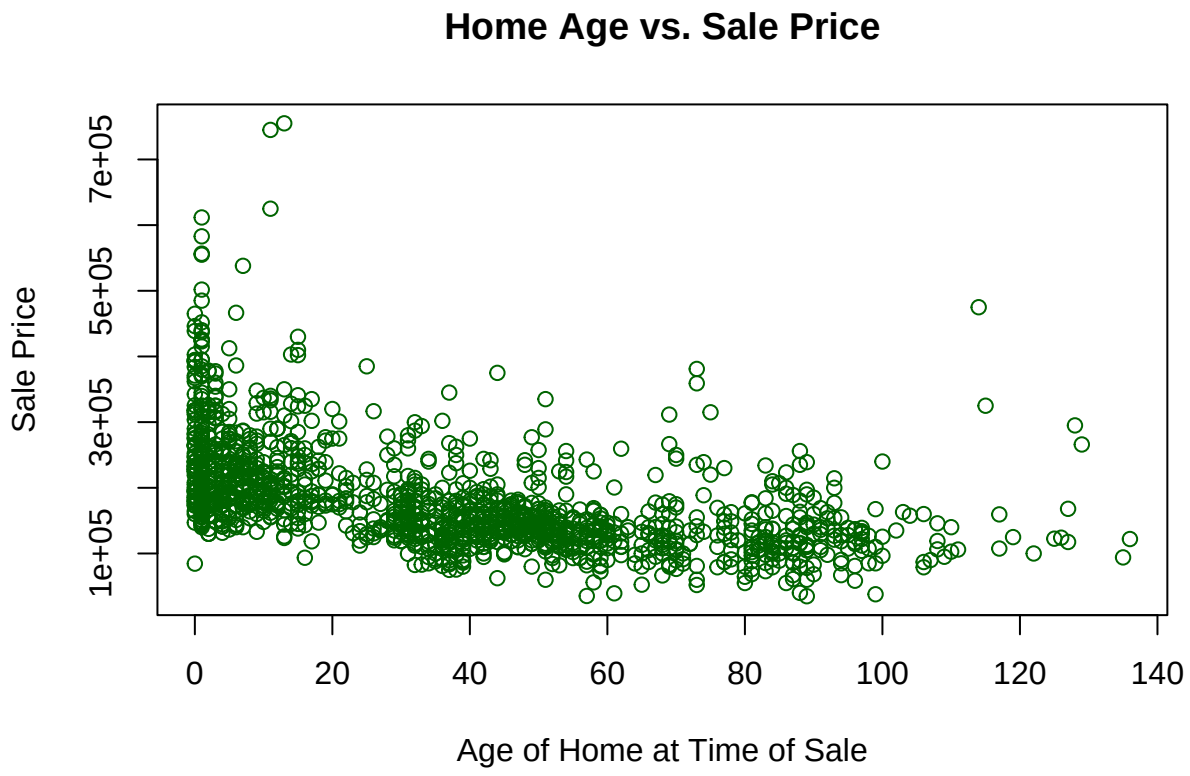
Somewhat similar to the earlier OverallCond discussion, it seems that more kitchens are associated with lower price, which is somewhat counterintuitive. Essentially all of the houses with 2 kitchens sold for less than $200,000, whereas homes with 1 kitchen sometimes sold for much more.

One thing we might want to investigate is what kinds of homes have two kitchens. Are they also homes with low quality, possibly student housing at Iowa State University?

5. Engineer and Explore a New Feature

Create a new feature Age, which represents the difference between the year sold and the year built, and plot the relationship between the age and sale price.

```
# Make a new column, Age
df$Age = df$YrSold - df$YearBuilt

# Plot Age vs. SalePrice
plot(df$Age, df$SalePrice, xlab = "Age of Home at Time of Sale",
     ylab = "Sale Price", col = "darkgreen",
     main = "Home Age vs. Sale Price")
```



In general, newer houses appear to be more valuable, with value increasing as homes age. Interestingly the variance seems to increase once the home age goes over 100 years, with several above-average sale prices and fewer home sales in general.

We are also seeing potential housing booms and busts over the past decades, indicated by e.g. relatively few 20-year-old houses compared to 25-year-old houses being sold. We might find something interesting if we investigate this further.