Data Pre-Processing:Loading Data and Basic Visualisations

Hamid Abdulsalam

10 Otober 2019

Packages

We will be working with following Packages

```
library(tidyverse)
library(readxl)
library(DBI)
library(RMySQL)
library(haven)
```

Packages

```
## -- Attaching packages ---- tidyverse 1.2.1 --

## v ggplot2 3.2.0     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts ------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

The Data

We have a series of data files that we will be working with. They are located in course folder given

- *auto*: data on automobiles
- *personality*: data on Big Five personality traits for 434 persons
- *birth1.sas7bdat*: data on birth weight
- *fertz*: data on fertilizer
- *who_suicide_statistics*: data on suicide
- *potatoes*: impact of storage and cooking on potatoes' flavor
- *Employees*: data on employees of certain company

Importing data using Base R functions

Reading Comma delimited files (csv) using `read.table()`

```
who_suicide <- read.table(file.choose(),sep = ",",
header=TRUE)
head(who_suicide)
```

```
##   country year    sex         age suicides_no population
## 1 Albania 1985 female 15-24 years          NA     277900
## 2 Albania 1985 female 25-34 years          NA     246800
## 3 Albania 1985 female 35-54 years          NA     267500
## 4 Albania 1985 female  5-14 years          NA     298300
## 5 Albania 1985 female 55-74 years          NA     138700
## 6 Albania 1985 female   75+ years          NA      34200
```

`read.table()`

- Generally used to load data in different format into R
- *file*: Path to the file containing the data to be imported into R.
- *sep*: field separator character. `\t` is used for tab-delimited file. _ *header*: logical value. If TRUE, read.table() assumes that your file has a header row, so row 1 is the name of each column. If that's not the case, you can add the argument header = FALSE.
- *dec*: the character used in the file for decimal points.

Reading Comma Seperated File (txt) using `read.table()`

```
auto <- read.table(file.choose(),sep = ",", header=F)
head(auto[,4:9])
```

```
##    V4  V5   V6          V7  V8    V9
## 1 gas std  two convertible rwd front
## 2 gas std  two convertible rwd front
## 3 gas std  two   hatchback rwd front
## 4 gas std four       sedan fwd front
## 5 gas std four       sedan 4wd front
## 6 gas std  two       sedan fwd front
```

read.table() can be used to read Comma Seperated file (.txt) by specifying the argument sep = ","

Reading TAB delimited files (txt) using `read.table()`

```
potato <- read.table(file.choose(),sep = "\t", header=F)
head(auto[,4:9])
```

```
##   V1 V2 V3 V4 V5  V6  V7  V8
## 1  1  1  1  1  1 2.9 3.2 3.0
## 2  1  1  1  1  2 2.3 2.5 2.6
## 3  1  1  1  1  3 2.5 2.8 2.8
## 4  1  1  1  1  4 2.1 2.9 2.4
## 5  1  1  1  1  5 1.9 2.8 2.2
## 6  1  1  1  2  1 1.8 3.0 1.7
```

read.table() can also be used to TAB delimited files (txt) by specifying the argument sep = "\t"

Reading comma delimited files using `read.csv`

```
who_suicide<-read.csv(file.choose(), header=TRUE)
head(who_suicide)
```

```
##   country year    sex         age suicides_no population
## 1 Albania 1985 female 15-24 years          NA     277900
## 2 Albania 1985 female 25-34 years          NA     246800
## 3 Albania 1985 female 35-54 years          NA     267500
## 4 Albania 1985 female  5-14 years          NA     298300
## 5 Albania 1985 female 55-74 years          NA     138700
## 6 Albania 1985 female   75+ years          NA      34200
```

read.csv is used in reading comma delimited files into R.

Reading semicolon separated files using `read.csv2`

```
who_suicide_1<-read.csv2(file.choose(), header=TRUE)
head((who_suicide_1)
```

```
##   country year    sex         age suicides_no population
## 1 Albania 1985 female 15-24 years          NA     277900
## 2 Albania 1985 female 25-34 years          NA     246800
## 3 Albania 1985 female 35-54 years          NA     267500
## 4 Albania 1985 female  5-14 years          NA     298300
## 5 Albania 1985 female 55-74 years          NA     138700
## 6 Albania 1985 female   75+ years          NA      34200
```

The read.csv2 is often used to read semicolon separated files into R. Setting *sep* argument is optional when using *read.csv2* as seen in the case of *read.csv*

Reading “Tab-separated value” files (“.txt”) using read.delim()

```
potato <- read.delim(file.choose(),sep = "\t", header=F)
head(auto[,4:9])
```

```
##   V1 V2 V3 V4 V5  V6  V7  V8
## 1  1  1  1  1  1 2.9 3.2 3.0
## 2  1  1  1  1  2 2.3 2.5 2.6
## 3  1  1  1  1  3 2.5 2.8 2.8
## 4  1  1  1  1  4 2.1 2.9 2.4
## 5  1  1  1  1  5 1.9 2.8 2.2
## 6  1  1  1  2  1 1.8 3.0 1.7
```

read.delim() is used for reading TAB delimited files (txt) by specifying the argument sep = ‘"”

Using `readr` Package

- In terms of speed, readr is ~10x faster than base `read.table()` functions (read.csv,read.csv2).
- By default , strings are untouched and common date/time formats are automatically passed.

readr

- `read_csv()`: comma delimited files
- `read_csv2()`: semicolon separated files
- `read_tsv()`: tab delimited files
- `read_delim()`: files with any delimiter

Import Excel files

We can always use the `readxl` package to get data out of Excel and into R. The `readxl` package supports both .xls format and the modern xml-based .xlsx format.

To import excel sheet into R, we use the function `read_excel()` and specify the sheet number in the arguments.

Import Excel files using `read_excel()` in `readxl` Package

```
sht1 <- read_excel(file.choose(), sheet = 1)
sht2 <- read_excel(file.choose(), sheet = 2)
str(sht1)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    20 obs. of  8 va
##  $ EmployeeID: num  120 121 123 124 125 126 127 128 129 130 .
##  $ Last_Name : chr  "Collins" "Kenobi" "Bouchard" "White" ...
##  $ First_Name: chr  "Barnabas" "Obi-wan" "Angelique" "Cassand
##  $ Gender    : chr  "M" NA "F" "F" ...
##  $ HireDate  : POSIXct, format: "2015-08-01" "2011-07-13" ...
##  $ JobTitle  : chr  "CEO" "Accountant II" "Analyst I" NA ...
##  $ Salary    : num  500000 91000 76 358000 65000 85000 94000
##  $ Department: chr  "Administration" "Jedi" "Research" "Admin
```

Importing data from databases

DBI

To import data from a database you first have to create a connection to it. We the package DBI to connect to SQL server through R aand RMYSQL to perform SQL queries within R. The Function `dbConnect()` creates a connection between your R session and a SQL database. The first argument is mapping the data between between R and the database. For hosted SQL DB , we need to specify the following arguments in dbConnect(): `dbname`, `host`, `port`, `user` and `password`.

Establish a connection

To extract data from database in a remote server, we need to first establish the connection to the server in R.

```
host <- "courses.csrrinzqubik.us-east-1.rds.amazonaws.com"
connect <- dbConnect(RMySQL::MySQL(), dbname = "tweater",
host = host, port = 3306, user = "student", ‘ password =
“datacamp”)“
```

List the database tables

Once we are connected to the database. We can use `dbListTables()` to see what tables the database contains:

```
tables <- dbListTables(connect) tables
```

```
## [1] "comments" "tweats"   "users"
```

Import data from tables

We can use the `dbReadTable()` function to import data from the database tables.

```
users <- dbReadTable(connect, "users")
users
```

```
##   id      name     login
## 1  1 elisabeth  elismith
## 2  2      mike     mikey
## 3  3      thea   teatime
## 4  4    thomas tomatotom
## 5  5    oliver olivander
## 6  6      kate  katebenn
## 7  7    anjali    lianja
```

Importing data from the web

We can also import csv file hosted on remote server directly into R. You can use `read_csv` to directly import csv files from the web.

```
house <-
read.csv("https://factual.ng/training/house.csv",
header = T)
str(house)
```

```
## 'data.frame':    781 obs. of  8 variables:
##  $ MLS.       : int  132842 134364 135141 135712 136282 1
##  $ Location   : Factor w/ 34 levels "Arroyo Grande",..: 1
##  $ Price      : int  795000 399000 545000 909000 109900 3
##  $ Bedrooms   : int  3 4 4 4 3 3 4 3 4 3 ...
##  $ Bathrooms  : int  3 3 3 4 1 3 2 2 3 2 ...
##  $ SQFT       : int  2371 2818 3032 3540 1249 1800 1603 1
##  $ Price.SQFT: num  335 142 180 257 88 ...
```

Importing data from other statistical software

To import data from other statistical software such as Stata, SPSS, Sas. We use the package called haven.

- *SAS*: `read_sas()`
- *STATA*: `read_dta()`
- *SPSS*: `read_sav()`

read_sas()to read SAS data file

we use the read_sas() function in haven package to read sas files into R

```
birth <- read_sas(file.choose())
head(birth)
```

```
## # A tibble: 6 x 10
##   Weight Black Married   Boy MomAge MomSmoke CigsPerDay MomWt
##    <dbl> <dbl>   <dbl> <dbl>  <dbl>    <dbl>       <dbl>     <
## 1   4111     0       1     1     -3        0           0
## 2   3997     0       1     0      1        0           0
## 3   3572     0       1     1      0        0           0
## 4   1956     0       1     1     -1        0           0
## 5   3515     0       1     1     -6        0           0
## 6   3757     0       1     0      3        0           0
## # ... with 1 more variable: MomEdLevel <dbl>
```

`read_dta()` to read Stata data file

we use the *read_dta()* function in haven package to Stata files into R

```
alcohol <- read_dta(file.choose())
head(alcohol)
```

```
## # A tibble: 6 x 4
##   adults  kids income consume
##    <dbl> <dbl>  <dbl>   <dbl>
## 1      2     2    758       1
## 2      2     3   1785       1
## 3      3     0   1200       1
## 4      1     0    545       1
## 5      4     1    547       1
## 6      2     2   1264       1
```

read_sav() to read SPSS data file into R \small

we use the *read_sav()* function in haven package to Stata files into R

```
pers <- read_sav("data/personality.sav")
head(pers)
```

```
## # A tibble: 6 x 4
##   Neurotic Extroversion Agreeableness Conscientiousness
##      <dbl>        <dbl>          <dbl>             <dbl>
## 1       39           38             31                12
## 2        6           38             27                12
## 3       17           39             32                13
## 4       28           35             39                13
## 5       26           35             46                14
## 6       17           37             28                15
```

Basic Visualisations

Scatter Plot in R

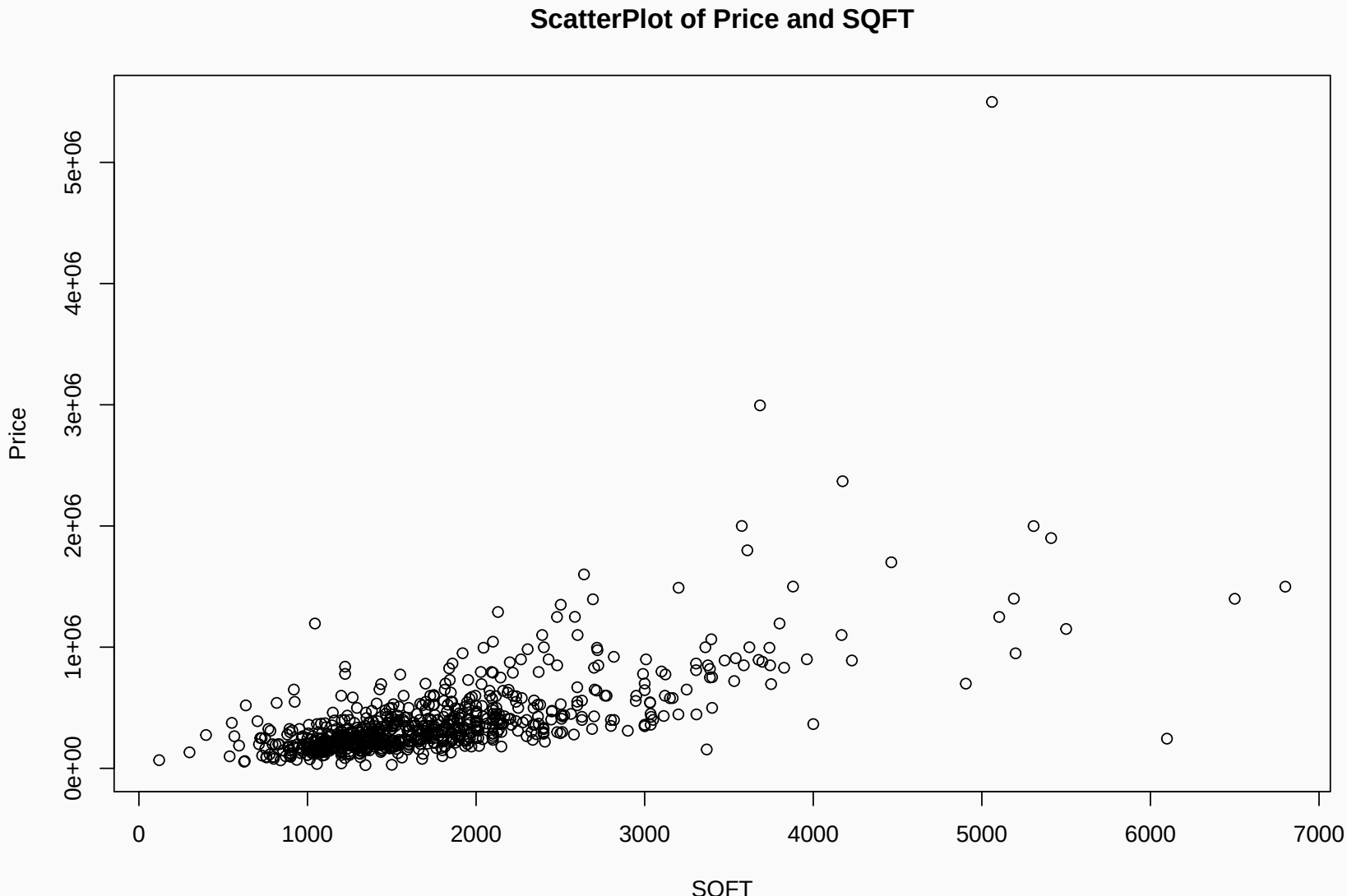
Scatter Plot is used to see the relationship between two continuous variables. Let's load in the data on housing price in California.

```
hs<-read.csv(file.choose(),header=T)
str(hs)
```

```
## 'data.frame':    781 obs. of  8 variables:
##  $ MLS.       : int  132842 134364 135141 135712 136282 1
##  $ Location   : Factor w/ 34 levels "Arroyo Grande",..: 1
##  $ Price      : int  795000 399000 545000 909000 109900 3
##  $ Bedrooms   : int  3 4 4 4 3 3 4 3 4 3 ...
##  $ Bathrooms  : int  3 3 3 4 1 3 2 2 3 2 ...
##  $ SQFT       : int  2371 2818 3032 3540 1249 1800 1603 1
##  $ Price.SQFT: num  335 142 180 257 88 ...
##  $ Status     : Factor w/ 3 levels "Foreclosure",..: 3 3
```

Relationship between the Price of the House and SQFT

```
plot(SQFT,Price, xlab="SQFT",ylab="Price",
main="ScatterPlot of Price and SQFT")
```

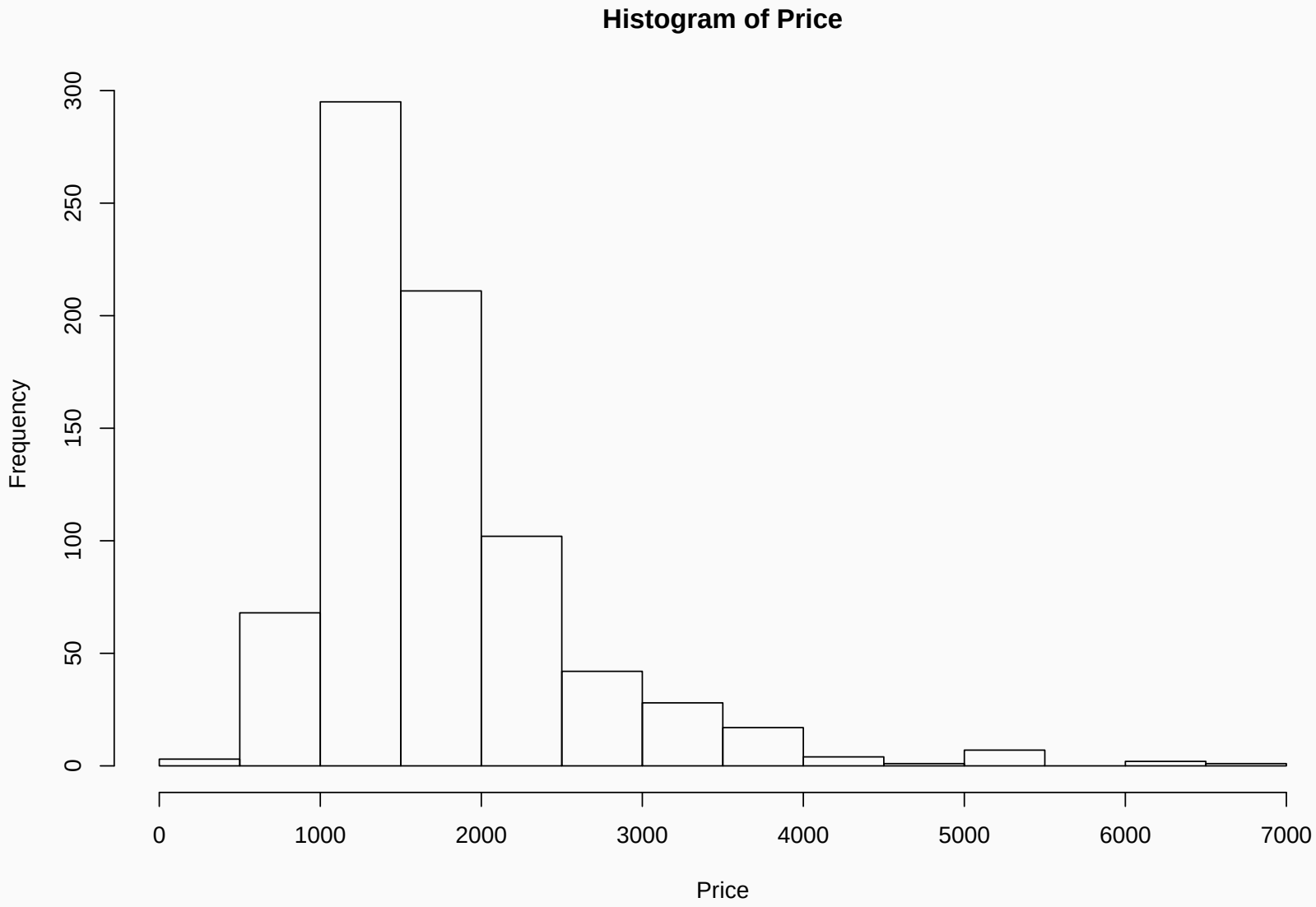


Histogram

Histogram is used to plot continuous variable. It breaks the data into bins and shows frequency distribution of these bins. We can always change the bin size and see the effect it has on visualization. It's often used to see the distribution of a variable.

```
hist(hs$Price, xlab="Price", main="Histogram of
Price")
```

Histogram od SQFT

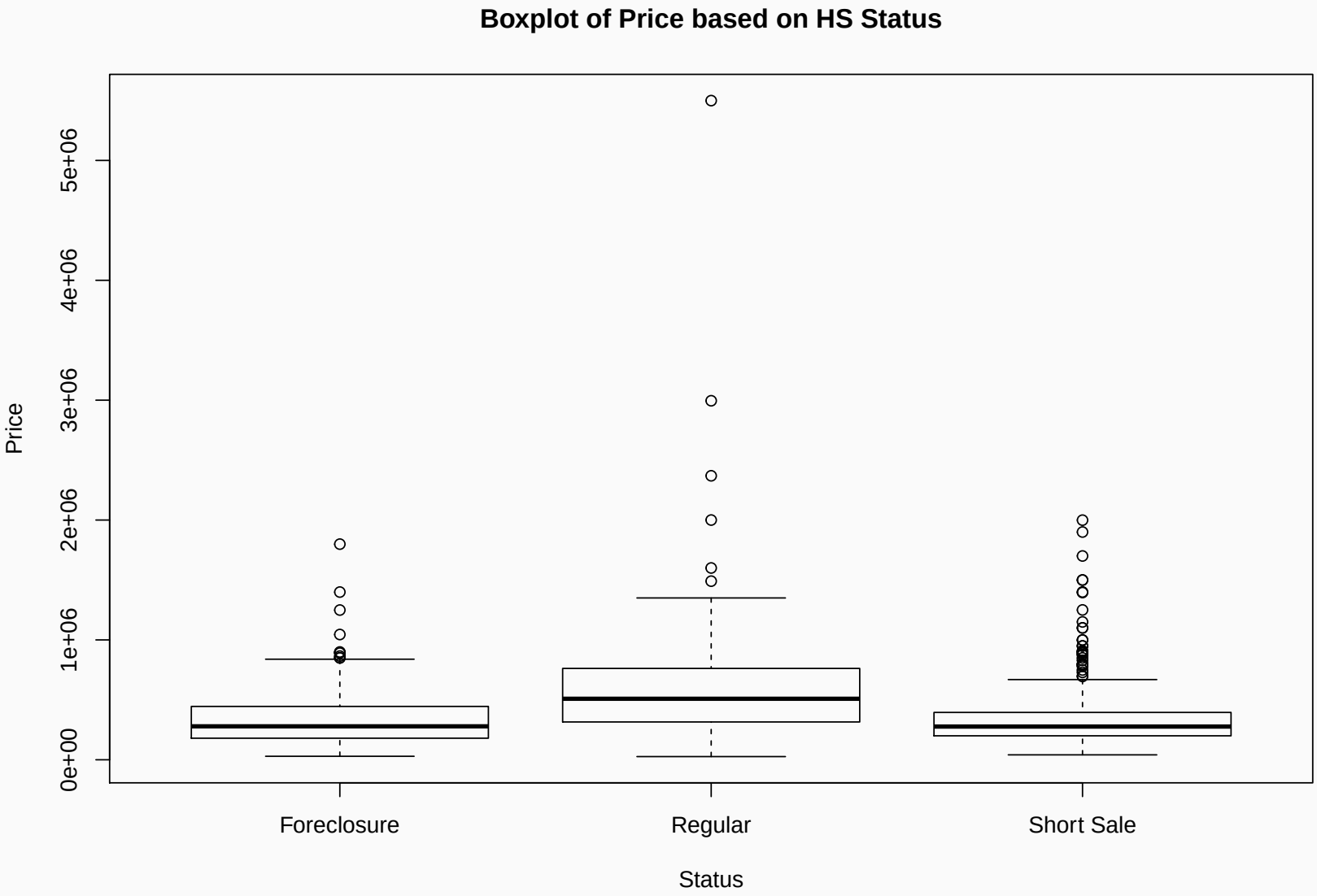


Boxplots

Box Plots are used to plot a combination of categorical and continuous variables. This plot is useful for visualizing the spread of the data and detect outliers.

```
boxplot(Price~Status, data = hs, main="Boxplot of
Price based on HS Status")
```

Boxplot of Price based on the House Status



The End of Session 2

Any Questions ?