HW1

Shuyuan Wang, sw3449

Sep 15, 2019

Part 1: Loading, Cleaning the Exploring Data in `R`

i. Load the data into a dataframe called `housing`.

```
housing <- read.csv('NYChousing.csv',header=TRUE)
```

ii. How many rows and columns does the dataframe have?

```
dim(housing)
```

```
## [1] 2506   22
```

`dim()` function returns the number of rows and columns respectively. Therfore, the dataframe has 2506 rows and 22 columns.

iii. Run the appropriate function to display the variable names of the dataframe.

```
colnames(housing)
```

```
##  [1] "UID"                          "PropertyName"
##  [3] "Lon"                          "Lat"
##  [5] "AgencyID"                     "Name"
##  [7] "Value"                        "Address"
##  [9] "Violations2010"               "REACNumber"
## [11] "Borough"                      "CD"
## [13] "CityCouncilDistrict"          "CensusTract"
## [15] "BuildingCount"                "UnitCount"
## [17] "YearBuilt"                    "Owner"
## [19] "Rental.Coop"                  "OwnerProfitStatus"
## [21] "AffordabilityRestrictions"    "StartAffordabilityRestrictions"
```

iv. Run this command, and explain, in words, what this does:

```
apply(is.na(housing),2,sum)
```

```
##                            UID                   PropertyName
##                              0                              0
##                            Lon                            Lat
##                             15                             15
##                       AgencyID                           Name
##                              0                              0
##                          Value                        Address
##                             52                              0
##                 Violations2010                     REACNumber
##                              0                           1873
##                        Borough                             CD
```

```
##                              0                               0
##            CityCouncilDistrict                     CensusTract
##                             10                               0
##                  BuildingCount                       UnitCount
##                              0                               0
##                      YearBuilt                           Owner
##                              0                               0
##                    Rental.Coop               OwnerProfitStatus
##                              0                               0
##       AffordabilityRestrictions StartAffordabilityRestrictions
##                              0                               5
```

`apply()` takes arguments of an array, margin (1 or 2, 1 indicating rows, 2 indicating colums), and a function which would be applied to the arrays over the margin, respectively. `is.na(housing)` returns a boolean matrix with `TRUE` standing for missing value, `False` standing for non-missing value. Therefore, the command above returns the `sum` of all the `TRUE` values across the rows for each column.

v. Remove the rows of the dataset for which the variable `Value` is NA.

```r
housing <- housing[!is.na(housing$Value),]
```

(After removal, the dataset is still called `housing`.)

vi. How many rows did you remove with the previous call? Does this agree with your result from (iv)?

```r
2506-dim(housing)[1]
```

```
## [1] 52
```

I removed 52 rows with the previous call.

The result from (iv) shows there are 52 rows with `NA` value in variable `Value`. The number of removal agrees with the result from (iv).

vii. Calculate the third quartile of the property values, i.e., the third quartile Q3 is the 75th percentile. Use the `quantile()` function to complete this task.

```r
Q3 <- quantile(housing$Value,probs=0.75)
Q3
```

```
##     75%
## 2684851
```

The third quantile of property values is `2684851`.

viii. Create a new variable in the dataset called `HighValue` that is equal to "High" if the property's value is greater than Q3 and is equal to "NotHigh" if the property's value is less than or equal to Q3.

```r
housing$HighValue <- ifelse(housing$Value>Q3,'High','NotHigh')
```

ix. Display a contingency table that shows the proprtions of `HighValue` split by `Borough`. Note that the `table()` function is the easiest way to tackle this problem but the `table()` function gives raw counts.

```
table(housing$HighValue,housing$Borough)/nrow(housing)
```

```
##
##               Bronx    Brooklyn   Manhattan      Queens Staten Island
##   High    0.055827221 0.053789731 0.114506927 0.019559902   0.006519967
##   NotHigh 0.214751426 0.284026080 0.232273839 0.011817441   0.006927465
```

x. What is the proportion of properties whose values are in the upper quartile and are located in The Bronx? Solve this question in two ways: (1) by using the table from (ix), and (2) by using logical/relational commands and using the function `mean()`.

(1) By using the table from (ix), the proportion is `0.05582722`.

(2) By using the function `mean()`, the proportion is `0.05582722`.

```
mean(housing$HighValue=='High' & housing$Borough=='Bronx')
```

```
## [1] 0.05582722
```

xi. Given a randomly selected property is in The Bronx, what is the probability that its value is in the upper quartile? Solve this question in two ways: (1) by using the table from (ix), and (2) by using logical/relational/filtering commands and using the function `mean()`.

```
0.055827221/(0.055827221+0.214751426)
```

```
## [1] 0.2063253
```

(1) By using the table from (ix), the conditional probability is `0.2063253`.

```
mean(housing$HighValue=='High' & housing$Borough=='Bronx')/mean(housing$Borough=='Bronx')
```

```
## [1] 0.2063253
```

(2) By using the function `mean()`, the conditional probability is `0.2063253`.

xii. Create a new variable in the dataset called `logValue` that is equal to the logarithm of the property's `Value`. What are the minimum, median, mean, and maximum values of `logValue`?

```
housing$logValue <- log(housing$Value)
summary(housing$logValue)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    8.41   12.49   13.75   13.68   14.80   20.47
```

The minimum, median, mean and maximum are 8.41, 13.75, 13.68, 20.47, respectively.

xiii. Create a new variable in the dataset called `logUnits` that is equal to the logarithm of the number of units in the property. The number of units in each piece of property is stored in the variable `UnitCount`.

```
housing$logUnits <- log(housing$UnitCount)
```

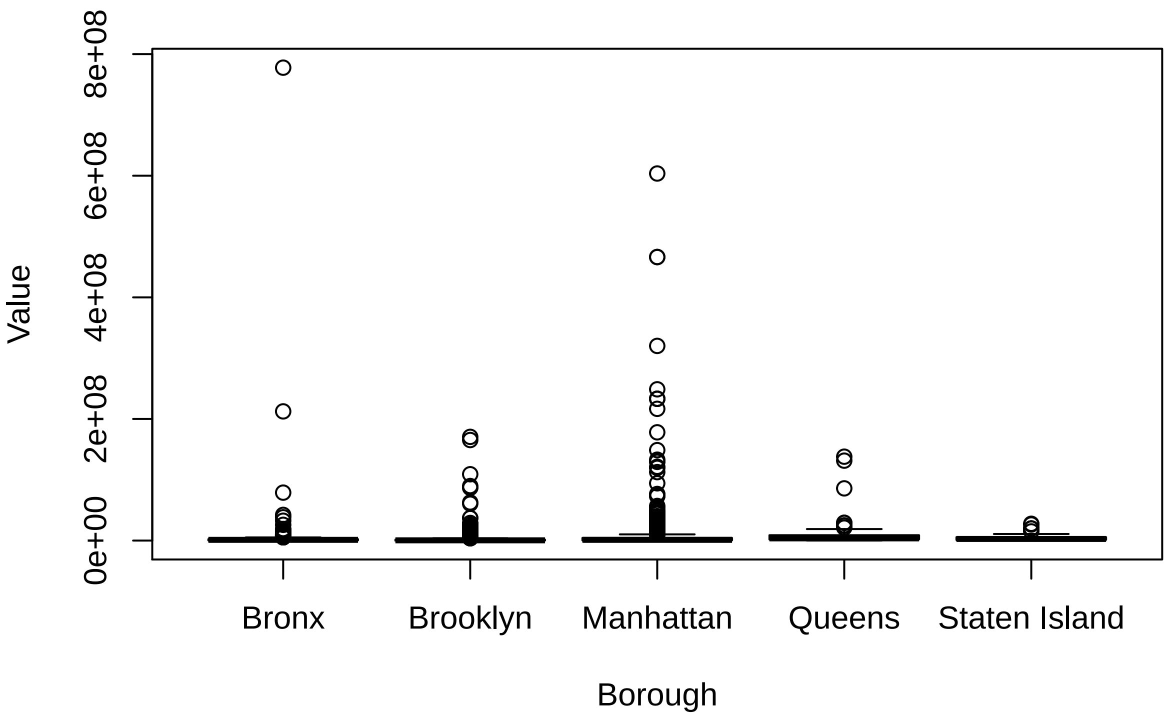
xiv. Finally create a new variable in the dataset called `after1950` which equals `TRUE` if the property was built in or after 1950 and `FALSE` otherwise. You'll want to use the `YearBuilt` variable here. This can be done in a single line of code.

```
housing$after1950 <- housing$YearBuilt>=1950
```

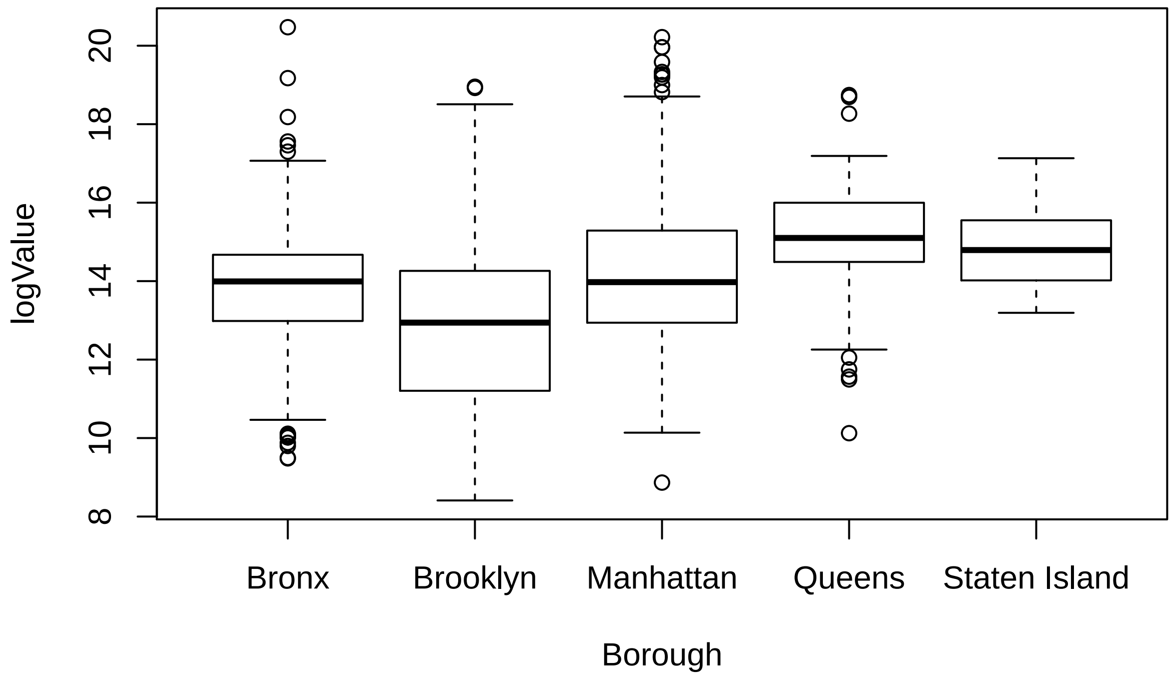
Part 2: EDA

2i. Create a multiple boxplot (side-by-side boxplots) comparing property value across the five boroughs. Create a multiple boxplot (side-by-side boxplots) comparing property `logValue` across the five boroughs. Make sure to label the plots appropriately.

```
boxplot(Value~Borough,data=housing)
```

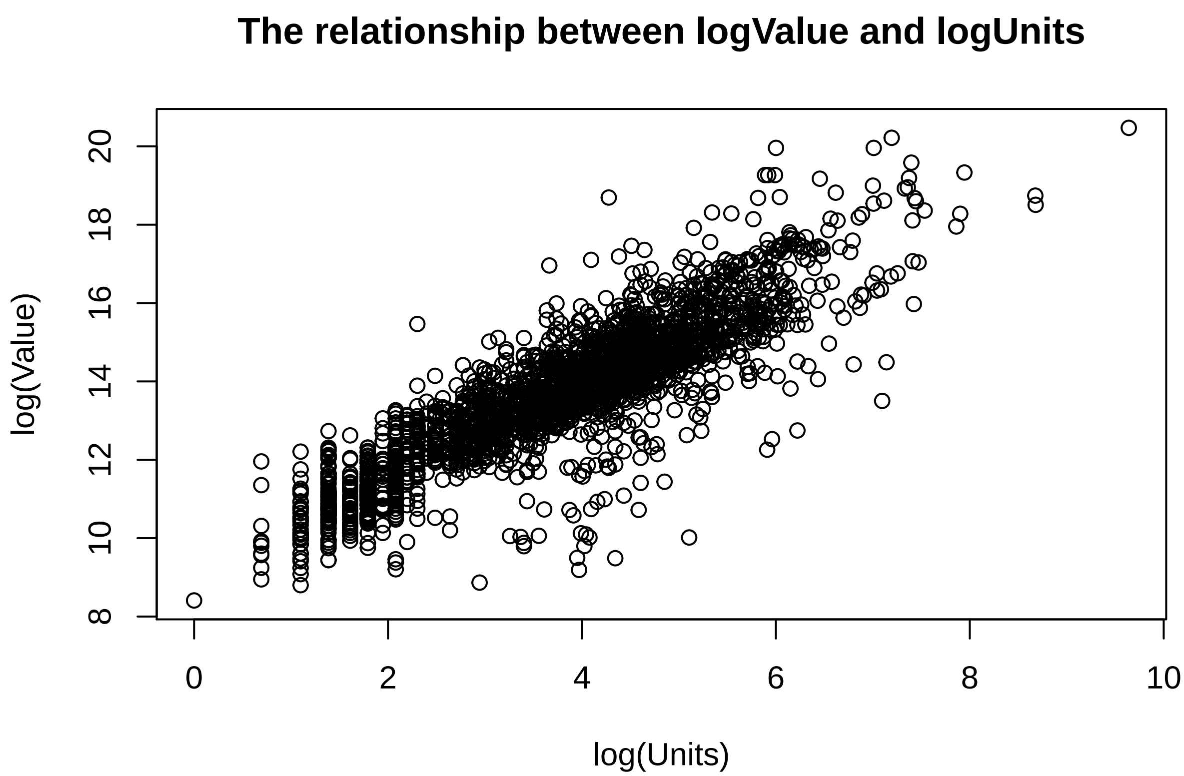



```
boxplot(logValue~Borough,data=housing)
```



2ii. Plot property `logValue` against property `logUnits`. Name the x and y labels of the plot appropriately. `logValue` should be on the y-axis.

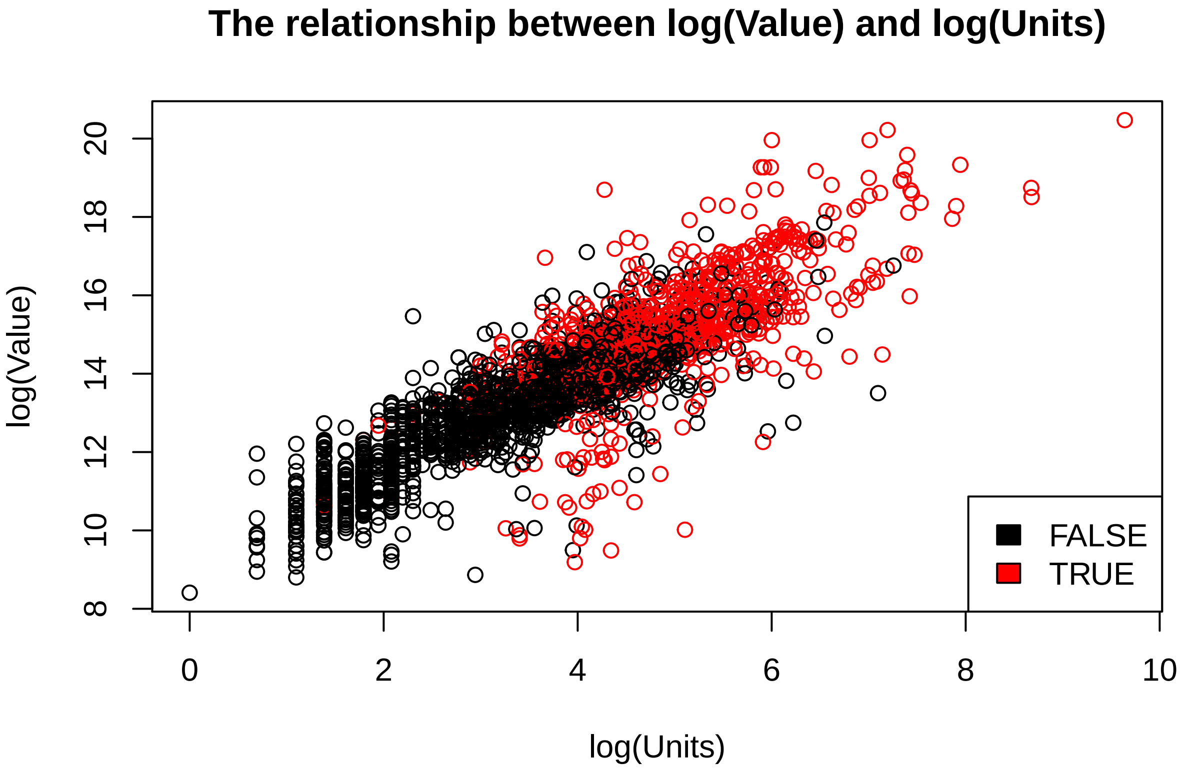
```
plot(x=housing$logUnits, y=housing$logValue, xlab='log(Units)', ylab='log(Value)',
     main='The relationship between logValue and logUnits')
```



2iii. Make the same plot as above,but now include the argument `col = factor(housing$after1950)`. Describe this plot and the covariation between the two variables. What does the coloring in the plot tell us?

Hint: `legend("bottomright", legend = levels(factor(housing$after1950)), fill = unique(factor(housing$afte`

```
plot(x=housing$logUnits, y=housing$logValue, xlab='log(Units)', ylab='log(Value)',
     main='The relationship between log(Value) and log(Units)',
     col=factor(housing$after1950))
legend("bottomright", legend = levels(factor(housing$after1950)),
            fill = unique(factor(housing$after1950)))
```



There is a linear and increasing relationship between `logUnits` and `logValue`, i.e. as `logUnits` gets larger, `logValue` gets larger as well. The covariation should be close to 1. The coloring represents differenct groups. The red one indicates the house built after 1950 and the black one indicates the house built before 1950. It can be shown that houses built after 1950 generally have more units and be more expensive than those built before 1950. No matter when does a house is built, the relationship between `logValue` and `logUnits` is increasing.

2iv. The `cor()` function calculates the correlation coefficient between two variables. What is the correlation between property `logValue` and property `logUnits` in (i) the whole data, (ii) just Manhattan (iii) just Brooklyn (iv) for properties built after 1950 (v) for properties built before 1950?

(i) Whole data:

```
cor(x=housing$logValue,y=housing$logUnits)
```

```
## [1] 0.8727348
```

(ii) Just Manhattan:

```
cor(x=housing$logValue[housing$Borough=='Manhattan'],y=housing$logUnits[housing$Borough=='Manhattan'])
```

```
## [1] 0.8830348
```

(iii) Just Brooklyn:

```
cor(x=housing$logValue[housing$Borough=='Brooklyn'],y=housing$logUnits[housing$Borough=='Brooklyn'])
```

```
## [1] 0.9102601
```

(iv) For properties built after 1950:

```
cor(x=housing$logValue[housing$after1950],y=housing$logUnits[housing$after1950])
```

```
## [1] 0.721735
```

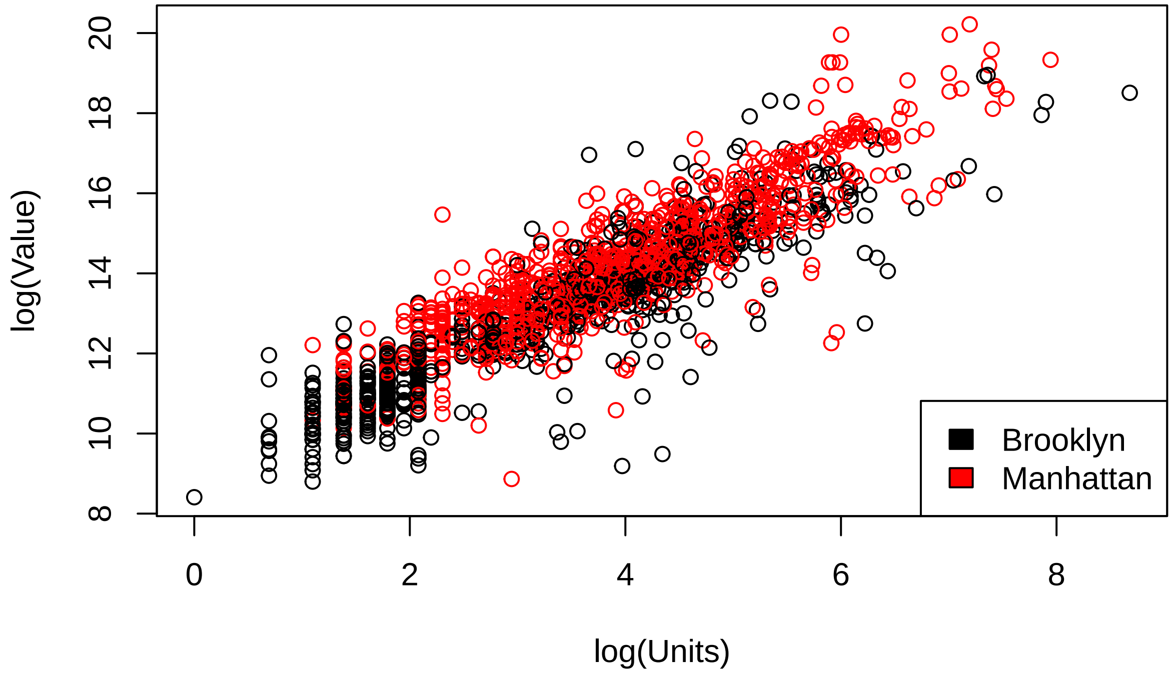
(v) For properties built before 1950:

```
cor(x=housing$logValue[!housing$after1950],y=housing$logUnits[!housing$after1950])
```

```
## [1] 0.8643297
```

2v. Make a single plot showing property `logValue` against property `logUnits` for Manhattan and Brooklyn. When creating this plot, clearly distinguish the two boroughs.

```
sub <- housing[housing$Borough=='Manhattan' | housing$Borough=='Brooklyn',]
plot(y=sub$logValue, x=sub$logUnits, xlab='log(Units)', ylab='log(Value)',
     col=factor(sub$Borough))
legend("bottomright", legend = levels(factor(sub$Borough)),
       fill = unique(factor(sub$Borough)))
```



2vi. Consider the following block of code. Give a single line of R code which gives the same final answer as the block of code. There are a few ways to do this.

The given code is to calculate the median of the value for the houses in Manhattan.

```
median(housing$Value[housing$Borough=='Manhattan'])
```

```
## [1] 1172362
```

This line of code gives the same result.

2vii. For five boroughs, what are the median property values? (Use `Value` here, not `logValue`.)

```
tapply(housing$Value,housing$Borough,median)
```

```
##         Bronx      Brooklyn     Manhattan        Queens Staten Island
##       1192950        417610       1172362       3611700       2654100
```