HW1 - Intro to Machine Learning

Saurabh Madaan

09/15/2012

1 Part 1 - HW01pb1data.csv

1.1 Attributes

The dataset has 800 observations of 5 variables. The first 3 columns are *integers* whereas the 4th and 5th columns are *factors*.

In R, this can be seen using the following commands:

```
setwd("/Users/Saurabh/Documents/ML_UCSC/week1/HW")
data<-read.csv("HW01pb1data.csv",header=FALSE)
class(data)
str(data)
#800 obs. of 5 variables. V1-3 are int, V4-5 are Factors
```

1.2 Reason for Categorical Variables

Columns 4 and 5 mostly have integers in them, but when one looks in to the *levels*, it is quickly visible that they also have strings "thirty five" and "twenty five", respectively. Consequently, *R* treats them as factors.

Below are the R commands which provide more details:

```
l4<-levels(data[,4])
#has integers, and "thirty five"
which(data[,4]=="thirty five")
#[1] 405

l5<-levels(data[,5])
#has integers, and "twenty five"
which(data[,5]=="twenty five")
```

```
# [1] 531
```

1.3 Plots for numeric and categorical variables

1. Plot of the 1st column (numerical data)

```
plot(data[,1])
```

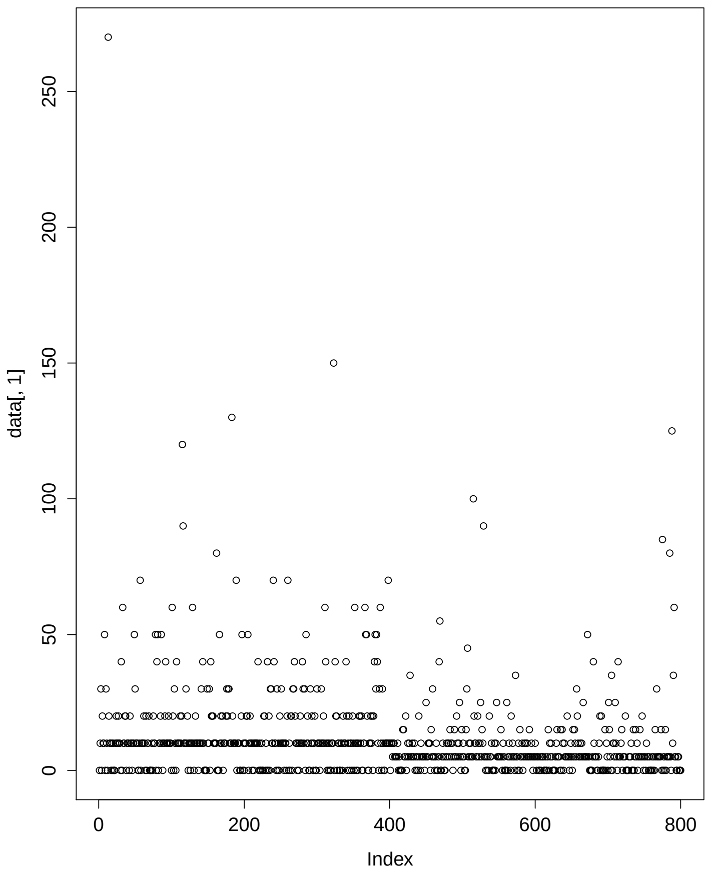

Figure 1: Plotting the numerical,1st column data

The y-axis of this plot are the values of the first column data, plotted against the index (row number) at which they occur in the dataset. This is a *scatter-plot* for a numerical data type

2. Plot of the 4th column (categorical data)

```
plot(data[,4])
```

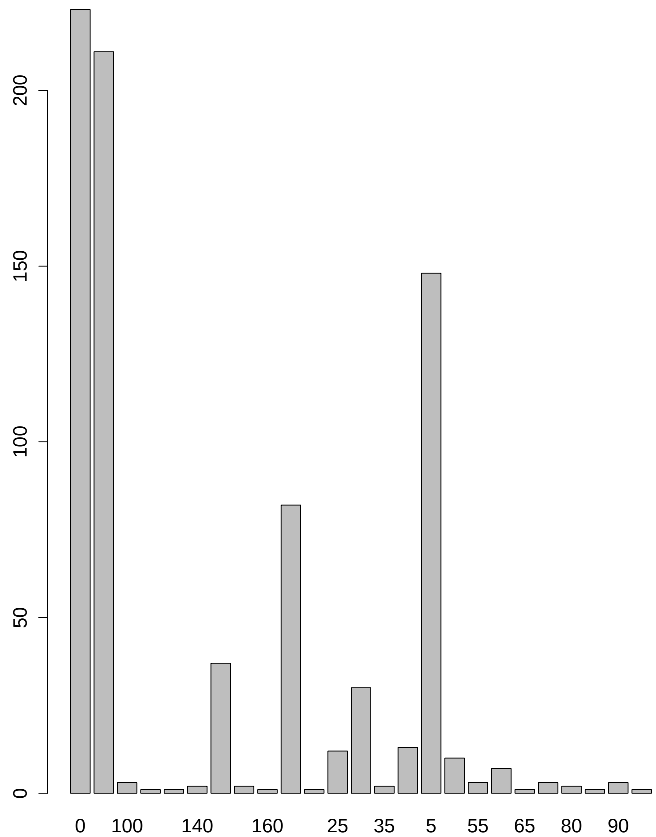

Figure 2: Plotting the categorical, 4th column data

This graph is a distribution: it is a *histogram* with the x-axis showing the values in the 4th column of our dataset, and the y-axis depicts the frequency with which values occur in a specified range. This graph is default since the 4th column data happens to be a categorical variable in our dataset.

2 Part 2 - HW01pb2data.csv

2.1 Extract a random sample of 10k observations

I used the *sample* command in *R* to obtain a smaller sample composed of 10,000 random records from our dataset.

Below are the R commands used to read and structure the data:

```
setwd("/Users/smadaan/Documents/ML_UCSC/week1/HW")
data<-read.csv("HW01pb2data.csv",header=FALSE)
str(data)
# 'data.frame': 2000000 obs. of 1 variable:

nrow(data)
#[1] 2000000

# selecting a sample of 10,000 random records
ss<-seq(1,nrow(data))
rand.ind<-sample(ss,10000,replace=F) #set of random indices for subset
small_data<-data[,1][rand.ind]
length(small_data)
#[1] 10000
```

2.2 Descriptive Stats on Sample

Below are the commands used to compute the mean, max, variance and 1st quartile from the data.

The resulting values are included as comments in the below code (mean =9.41002, max=16.93748, var=4.004991).

```
#mean, max and other descriptive stats
mean(small_data)
#[1] 9.41002
max(small_data)
#[1] 16.93748
var(small_data)
#[1] 4.004991
quantile(small_data,0.25)
#8.079612
```

2.3 Descriptive Stats for the Entire data

Below is the *R* code to compute descriptive stats for the entire dataset.

```
mean(data)
#9.451468
max(data)
# [1] 18.96657
var(data)
# 4.001822
quantile(data[,1],0.25)
# 8.10388
```

The population values for these parameters are very close to those that we found for the smaller sample. The table below shows the comparison.

Statistic	Small Sample	Entire-Dataset
Mean	9.41002	9.451468
Max	16.93748	18.96657
Variance	4.004991	4.001822
1st Quartile	8.079612	8.10388

Table 1: Comparison of statistics between the sample and entire-population datasets

3 Analysis of Ocean-view and Desert Home Prices

3.1 Box Plots for Pricing Data

R code for box plots:

```
boxplot(data.desert[,1], data.ocean[,1],col=c("red","blue"),
          main="House Box Plots: Comparison of House Prices",
          names=c("Desert View","Ocean View"),ylab="Prices (in thousand 
              dollars)")
```

The BoxPlot shows the distribution of prices between the two kinds of houses. Desert View houses have a much lower price than Ocean-view houses. The edges of the box indicate the 1st and last quartiles– the median value (bold line) is much closer to the 25th percentile for desert view houses, indicating that the median is lower than the mean for desert houses. Finally, although the desert view houses are less expensive on average, the most expensive desert homes have a higher price (over 2.5 million dollars) than the most expensive ocean-view houses (less than 2.5 million dollars).

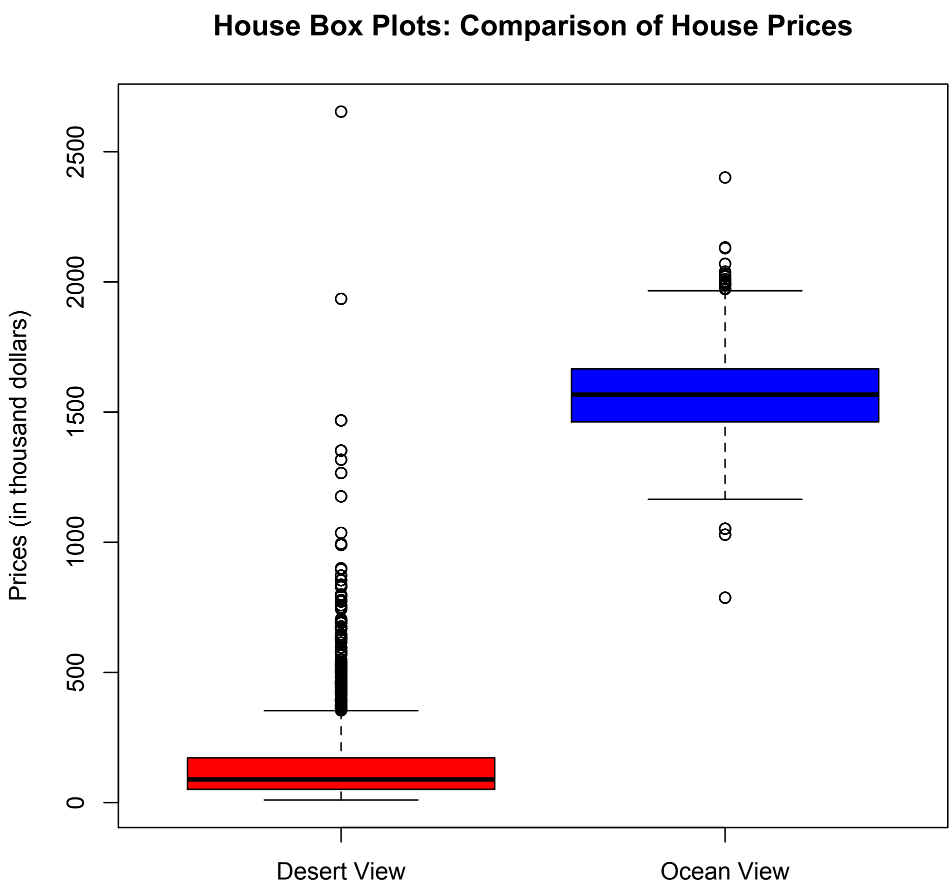

Figure 3: Comparison of prices between desert and ocean-view and desert houses

3.2 Frequency Histogram for Ocean-view Houses

R code for histogram:

```
hist(data.ocean[,1], breaks=seq(from=0,to=3000,by=500),
        xlab="Price (in thousand dollars)", main="Price Distribution of Ocean-view Houses")
```

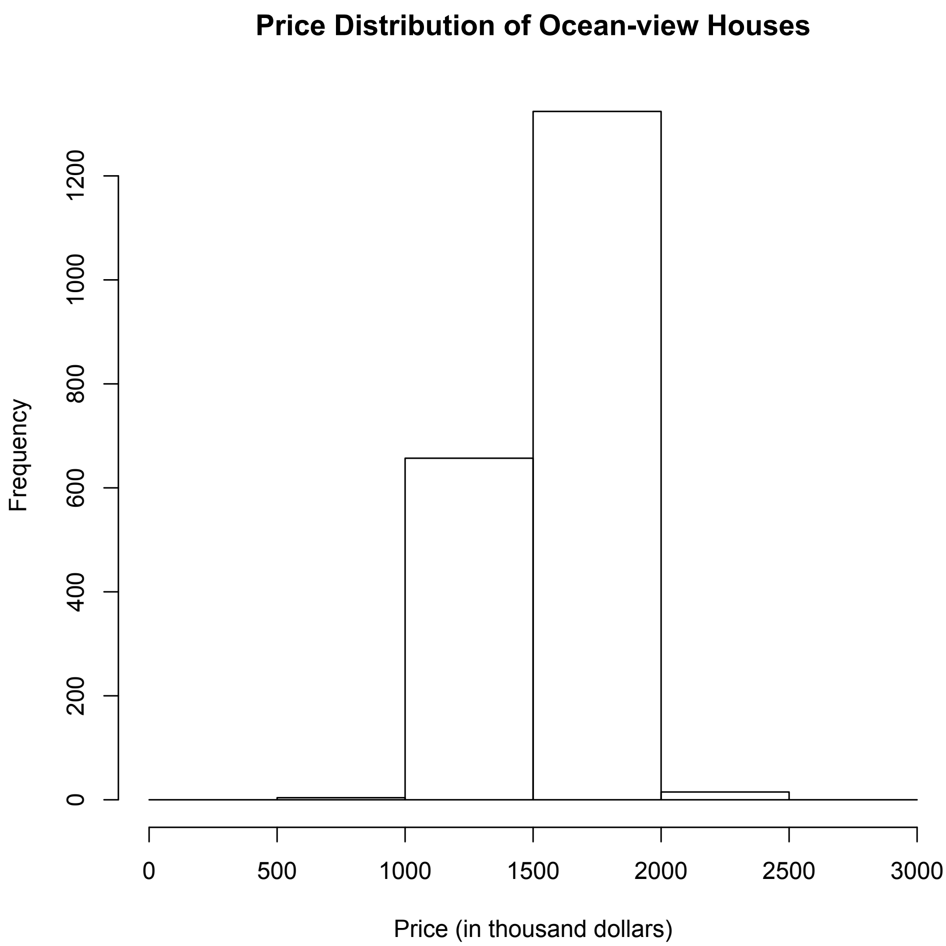

Figure 4: Histogram for prices of the ocean-view houses (0 to 3 million dollars)

3.3 ECDF for House Prices

R code for ecdf plots:

```
hist(data.ocean[,1], breaks=seq(from=0,to=3000,by=500),
          xlab="Price (in thousand dollars)", main="Price Distribution of Ocean-view Houses")
```

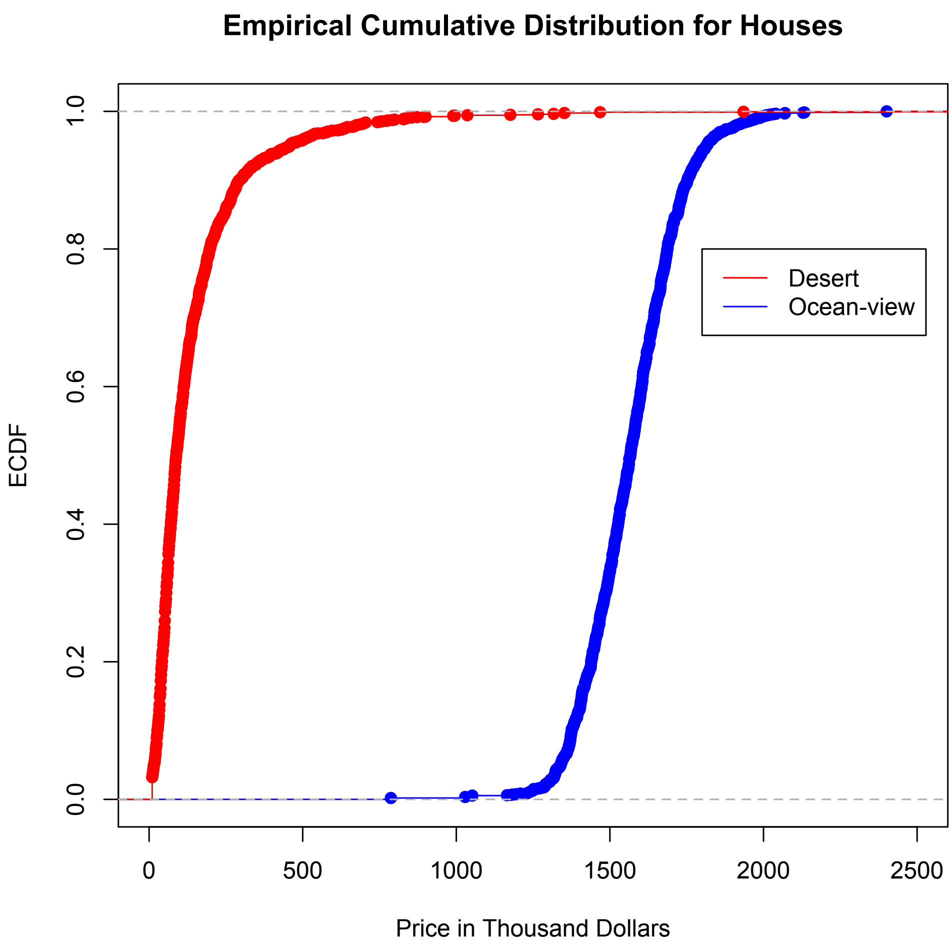

Figure 5: ECDF for Desert and Ocean-view Houses

4 Analyzing Data for Orange Trees

4.1 Age vs. Circumference relation

We first explore the data to identify the 3 variables: Tree, age, circumference. Using the *summary* command on the dataset outputs useful information, including the (min,max) parameters for age and circumference, and the different levels for 'Tree' (1,2,3,4,5).

x-axis (age) ranges from 118-152, and y-axis (circumference) ranges from 30-214.

Below is the *R* code for the same:

```
#-- explore the data
head(orange)
levels(orange[,1])
str(orange)
summary(orange)

# 3 variables: Tree(factor), age (numeric), circumference (numeric)
# range for age: 118-1582
# range for circumference: 30-214

#-- plot age vs circumference
plot(orange$age, orange$circumference, xlab="Age of Tree (days since 1968/12/31)",
        ylab="Trunk Circumference (mm)", pch=20,main="Circumference vs. Age for Trees",
        col=orange$Tree)
legend('bottomright', legend = levels(factor(orange$Tree)),
                text.col=seq_along(levels(orange$Tree)), title="Tree Type")
```

The resulting plot shows the variation of tree circumference with age, for the different groups/levels of trees.

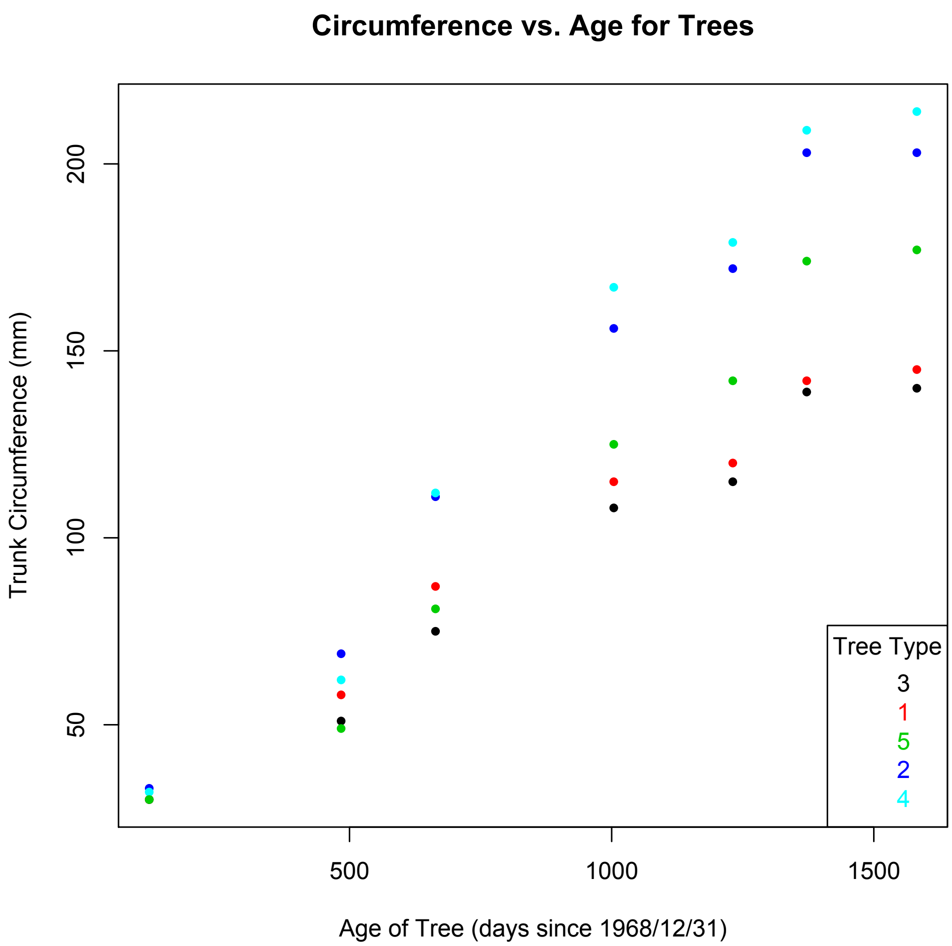

Figure 6: Age vs. Circumference, for the 5 groups of Orange Trees

4.2 Correlation between age and circumference for 1st Tree type

The two variables (age, circumference) are strongly correlated, with a correlation value of 0.9854675.

R code:

```
orange.1<-orange[which(orange$Tree==1),]
cor(orange.1$age,orange.1$circumference)
# [1] 0.9854675
```

4.3 Covariance and Correlation for Each Tree-type

Approach: Create an empty data frame with the different tree groups as the first column. Then, leverage the *by* command to calculate the parameters for each group. This saves having to individually calculate and *merge* results.

Below is the *R* code with results:

```
names(orange)
t.levels<-sort(levels(orange$Tree))

stats<-data.frame(matrix(nrow=length(t.levels),ncol=0))
stats$Tree<-t.levels
stats$COVARIANCE<-as.matrix(by(orange, orange$Tree,
                              function(x){cov(x$age,x$circumference)}))
stats$CORRELATION<-as.matrix(by(orange, orange$Tree,
                              function(x){cor(x$age,x$circumference)}))
stats
'
> stats
  Tree COVARIANCE CORRELATION
1    1   22239.83   0.9881766
2    2   22340.07   0.9854675
3    3   30442.81   0.9877376
4    4   34290.45   0.9873624
5    5   37062.62   0.9844610
'
```

4.4 Effect of Adding 10 to each Circumference value

Correlation and covariance remain *unchanged*.

Below is the *R* code with results:

```
new.stats<-data.frame(matrix(nrow=length(t.levels),ncol=0))
new.stats$Tree<-t.levels
new.stats$COVARIANCE<-as.matrix(by(orange, orange$Tree,
                              function(x){cov(x$age,x$circumference+10)}))
new.stats$CORRELATION<-as.matrix(by(orange, orange$Tree,
                              function(x){cor(x$age,x$circumference+10)}))
new.stats
# no change in covariance or correlation
'
> new.stats
  Tree COVARIANCE CORRELATION
1    1   22239.83   0.9881766
2    2   22340.07   0.9854675
3    3   30442.81   0.9877376
4    4   34290.45   0.9873624
5    5   37062.62   0.9844610
'
```

4.5 Effect of Doubling Circumference

Covariance *doubles* and correlation remains *unchanged*.

Below is the *R* code with results:

```
new2.stats<-data.frame(matrix(nrow=length(t.levels),ncol=0))
new2.stats$Tree<-t.levels
new2.stats$COVARIANCE<-as.matrix(by(orange, orange$Tree,
                              function(x){cov(x$age,x$circumference*2)}))
new2.stats$CORRELATION<-as.matrix(by(orange, orange$Tree,
                              function(x){cor(x$age,x$circumference*2)}))
new2.stats
'
> new2.stats
  Tree COVARIANCE CORRELATION
1    1   44479.67   0.9881766
2    2   44680.14   0.9854675
3    3   60885.62   0.9877376
4    4   68580.90   0.9873624
5    5   74125.24   0.9844610
'
# covariance doubles, correlation remains same
```

4.6 Effect of Multiplying Circumference by -2

Covariance becomes negative (and double in magnitude compared to the original dataset), and correlation becomes *negative* of its original value. Results:

New covariance = original covariance * (-2)

New correlation = -original correlation

Below is the *R* code with results:

```
new3.stats<-data.frame(matrix(nrow=length(t.levels),ncol=0))
new3.stats$Tree<-t.levels
new3.stats$COVARIANCE<-as.matrix(by(orange, orange$Tree,
                              function(x){cov(x$age,x$circumference*(-2))}))
new3.stats$CORRELATION<-as.matrix(by(orange, orange$Tree,
                              function(x){cor(x$age,x$circumference*(-2))}))
new3.stats
'
> new3.stats
  Tree COVARIANCE CORRELATION
1    1  -44479.67  -0.9881766
2    2  -44680.14  -0.9854675
3    3  -60885.62  -0.9877376
4    4  -68580.90  -0.9873624
5    5  -74125.24  -0.9844610
'
# new.corr = orig.corr*-2, correlation becomes -ve
```

5 Desert Homes: Revisited

5.1 Calculating the Median Price

Approach: *R* does not have a native *median* function, so we define a custom *median* function to calculate the median home price. This results in:

Below is the *R* code with results:

```

median = function (data.vector)
{
        median=NA;
        l = length(data.vector);
        sv<-sort(data.vector);
```

```
        median=ifelse(is.integer(l/2),sv[l/2],sv[l/2+1]);
        return(median);
}


median(data.desert[,1])
#89
mean(data.desert[,1])
#144.0348
```

Median value: 89

Mean value: 144.0348

(note: the values are in units of thousand dollars).

Median is *smaller* than the mean.

5.2 Characteristics of the Distribution

We found earlier that the median is lower than the mean. This generally indicates that the distribution is positively, or *right-skewed*.

This can be further confirmed by plotting a histogram to get a sense of the distribution, as shown below.

```
hist(data.desert[,1], main="Distribution of Desert Home Prices",
        xlab="Home Price", ylab="Number of Homes")
# also, median < mean, positive/right skewed
```

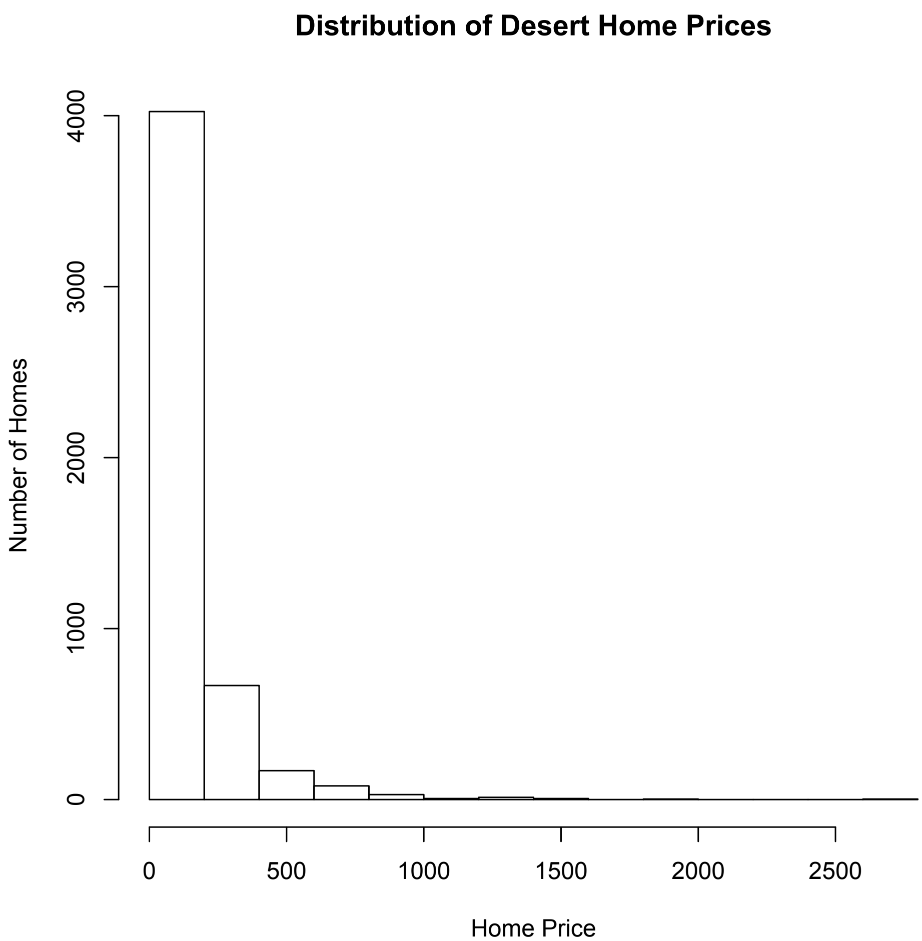

Figure 7: Histogram showing desert home prices

5.3 Effect of Higher Home Prices on Median Value

If we increase house price by 10 (thousand dollars) each, the median value *increases* by the same amount.

Specifically, the new median (99k) is higher than the old median (89k) by 10 (thousand dollars). Below is the *R* code with results:

```
median(data.desert[,1]+10)
#99
#median also increases by 10k
```

5.4 Effect of Doubling Home Prices on Median Value

If home prices double, the median value also *doubles*.

Specifically, the new median (178k) is twice the old median (89k). Below is the *R* code with results:

```
median(data.desert[,1]*2)
#178
#median also doubles
```

6 Closing Notes and References

I read material from *The Art of R Programming* by Norman Matloff for an introduction to *R*, and googled online references for the use of the *by* command.

Tools used include the *R* package for Mac, *TeXShop* for writing Latex files. I did *not* collaborate with another class member on this assignment.

Included in my submission is this *pdf* file, and all the *R* files where my code is programmed.