“COMPSCIX 415.2 Homework 5/Midterm”

Santosh Kanutala

7/12/2018

Contents

Github location . . . 2
RStudio and R Markdown . . . 2
    Question: 1 . . . 2
The tidyverse packages . . . 2
    Question: 1 . . . 2
    Question: 2 . . . 2
R Basics . . . 2
    Question: 1 . . . 2
    Question: 2 . . . 2
    Question: 3 . . . 3
Data import/export . . . 3
    Question: 1 . . . 3
    Question: 2 . . . 3
Visualization . . . 4
    Question: 1 . . . 4
    Question: 2 . . . 4
    Question: 3 . . . 5
Data munging and wrangling . . . 5
    Question: 1 . . . 5
    Question: 2 . . . 6
    Question: 3 . . . 6
EDA . . . 7
    Question: 1 . . . 7
    Question: 2 . . . 7
    Question: 3 . . . 8
    Question: 4 . . . 9
    Question: 5 . . . 10
    Question: 6 . . . 11
Git and Github . . . 15

Github location

My homework assignments can be found at https://github.com/santumagic/compscix-415-2assignments.git

RStudio and R Markdown

Question: 1

As part of this question, I have loaded the required packages and added instructions for table of contents etc in the YAML header.

```
# Load the required packages
library(tidyverse)
library(mdsr)
library(nycflights13)
```

The tidyverse packages

Question: 1

Plotting - **ggplot2**
Data munging/wrangling - **dplyr**
Reshaping (speading and gathering) data - **tidyr**
Importing/exporting data - **readr**

Question: 2

Plotting - **ggplot()** and **aes()**
Data munging/wrangling - **select()** and **filter()**
Reshaping (speading and gathering) data - **separate()** and **extract()**
Importing/exporting data - **read_csv()** and **read_delim()**

R Basics

Question: 1

```
My_data.name___is.too00ooLong <- c( 1 , 2 , 3 )
My_data.name___is.too00ooLong
```

```
## [1] 1 2 3
```

Answer: Just with one change (removal of '!'), the code works.

Question: 2

```
# this is a charactor vector
my_string <- c('has', 'an', 'error', 'in', 'it')
my_string
```

```
## [1] "has"   "an"    "error" "in"    "it"
```

Question: 3

```
my_vector <- c(1, 2, '3', '4', 5)
my_vector
```

```
## [1] "1" "2" "3" "4" "5"
```

Answer: This is a numeric vector and with or without the single or double quotes, vector takes values.

Data import/export

Question: 1

```
# Download and import the file rail_trail.txt
rail_trail.txt <- read.delim("/Users/skanutal/Documents/Santosh/Learning/Berkeley/rail_trail.txt", sep=
#glimpse the data from txt file
glimpse(rail_trail.txt)
```

```
## Observations: 90
## Variables: 10
## $ hightemp   <int> 83, 73, 74, 95, 44, 69, 66, 66, 80, 79, 78, 65, 41,...
## $ lowtemp    <int> 50, 49, 52, 61, 52, 54, 39, 38, 55, 45, 55, 48, 49,...
## $ avgtemp    <dbl> 66.5, 61.0, 63.0, 78.0, 48.0, 61.5, 52.5, 52.0, 67....
## $ spring     <int> 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, ...
## $ summer     <int> 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, ...
## $ fall       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, ...
## $ cloudcover <dbl> 7.6, 6.3, 7.5, 2.6, 10.0, 6.6, 2.4, 0.0, 3.8, 4.1, ...
## $ precip     <dbl> 0.00, 0.29, 0.32, 0.00, 0.14, 0.02, 0.00, 0.00, 0.0...
## $ volume     <int> 501, 419, 397, 385, 200, 375, 417, 629, 533, 547, 4...
## $ weekday    <int> 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, ...
```

Question: 2

```
# Export the .txt file as csv into a different location
rail_trail_csv <- write_delim(
  rail_trail.txt, delim = '|',path = "/Users/skanutal/Documents/Santosh/Learning/Berkeley/3. Intro to DS
  )
# Load the newly created csv file
rail_trail_csv_final <- read.csv(
  "/Users/skanutal/Documents/Santosh/Learning/Berkeley/3. Intro to DS/Assignments/rail_trail.csv", sep=
  )
# glimpse the data from the final csv file
glimpse(rail_trail_csv_final)
```

```
## Observations: 90
## Variables: 10
## $ hightemp   <int> 83, 73, 74, 95, 44, 69, 66, 66, 80, 79, 78, 65, 41,...
## $ lowtemp    <int> 50, 49, 52, 61, 52, 54, 39, 38, 55, 45, 55, 48, 49,...
## $ avgtemp    <dbl> 66.5, 61.0, 63.0, 78.0, 48.0, 61.5, 52.5, 52.0, 67....
## $ spring     <int> 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, ...
## $ summer     <int> 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, ...
## $ fall       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, ...
```

```
## $ cloudcover <dbl> 7.6, 6.3, 7.5, 2.6, 10.0, 6.6, 2.4, 0.0, 3.8, 4.1, ...
## $ precip     <dbl> 0.00, 0.29, 0.32, 0.00, 0.14, 0.02, 0.00, 0.00, 0.0...
## $ volume     <int> 501, 419, 397, 385, 200, 375, 417, 629, 533, 547, 4...
## $ weekday    <int> 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, ...
```

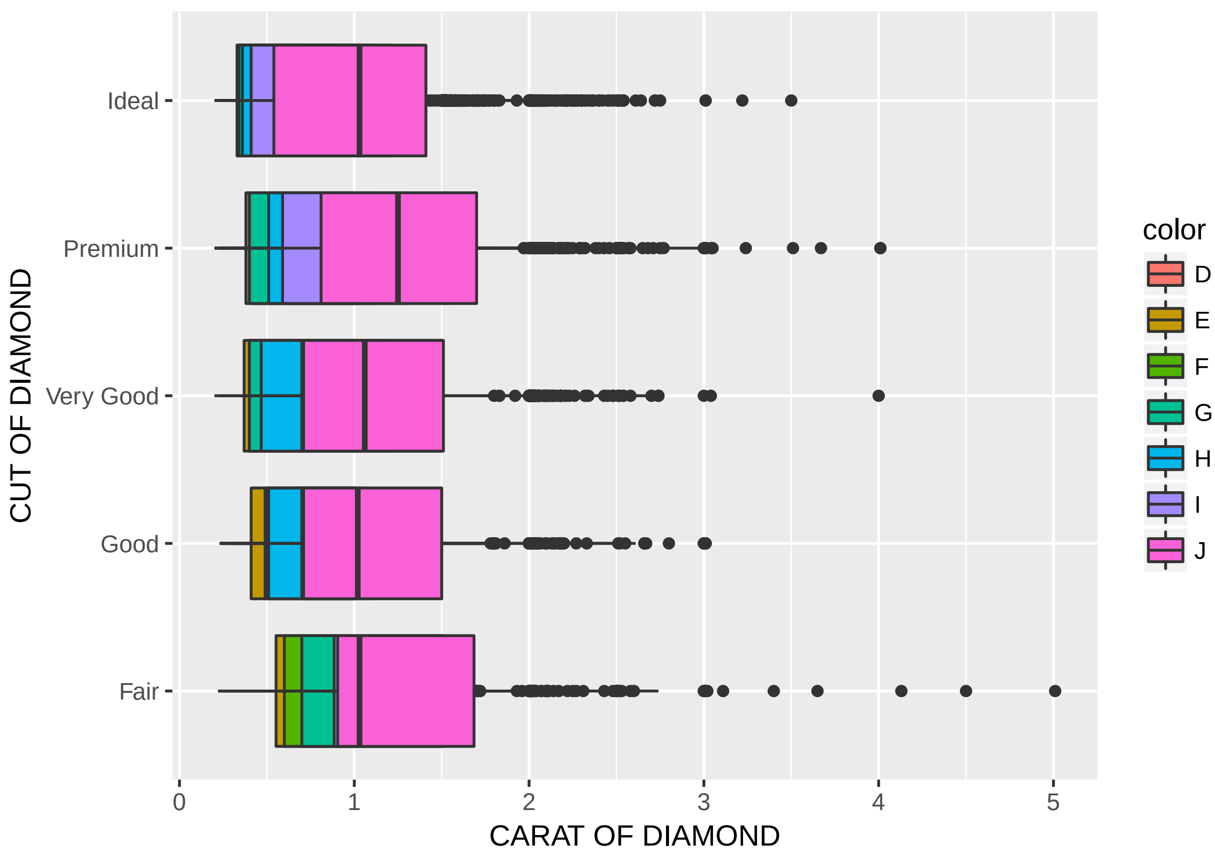
Visualization

Question: 1

Answer:
1. Both the categories age group and gender are plotted on same axis, which is confusing at a first glanse.
2. These are two seperate charts, but they look like one. The first chart is a chart with three ranges (<45, 45 to 64, and >64), the second chart is a men vs women chart. This simple difference is not easily visible with how it is layed out currently.
3. With the way the data is currently layed out it is not clear that yes/no data points are proportions and the title should visually be represented.

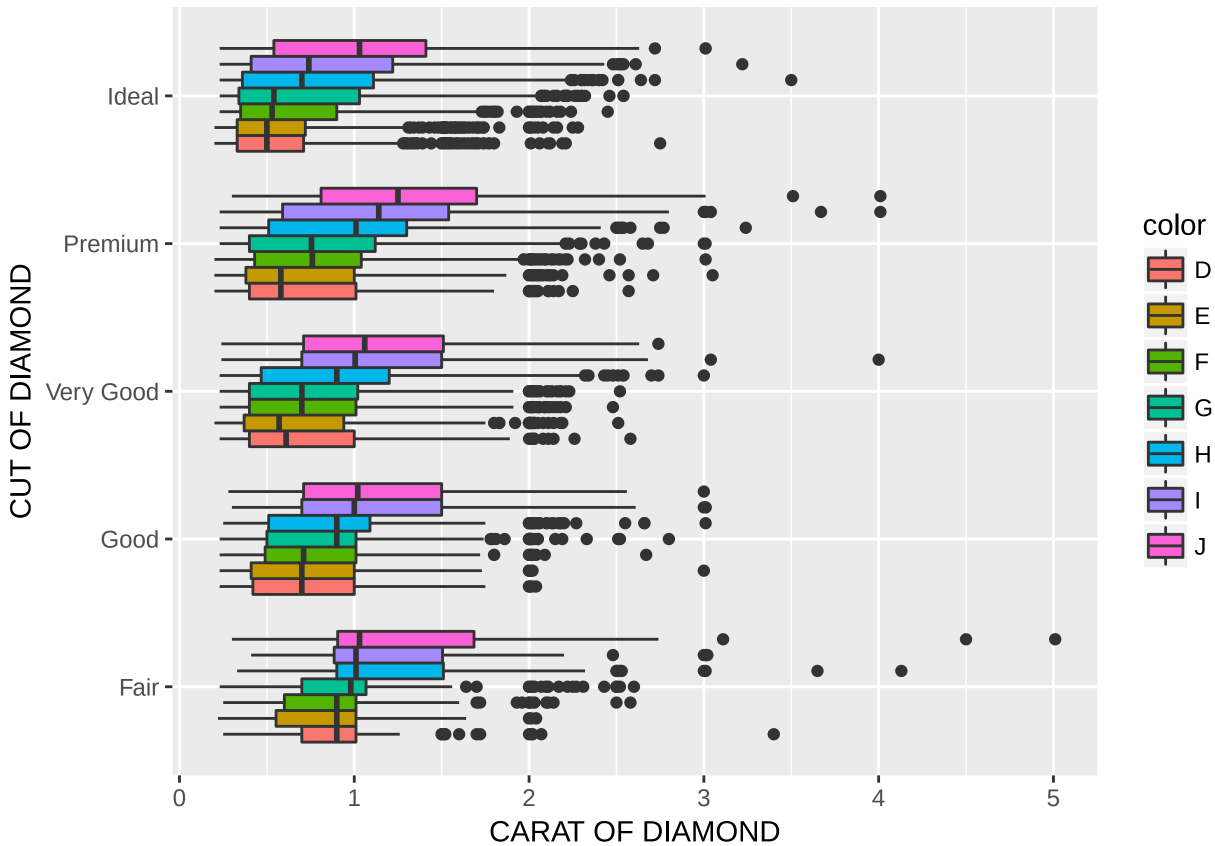
Question: 2

```r
# Reproduce the given graph
ggplot(data = diamonds, mapping = aes(x = cut, y = carat, fill = color)) +
  geom_boxplot (position = "identity") +
  coord_flip() +
  labs(x = "CUT OF DIAMOND", y = "CARAT OF DIAMOND")
```



Question: 3

```
# Enhancing the graph by changing the position to "dodge"
ggplot(data = diamonds, mapping = aes(x = cut, y = carat, fill = color)) +
  geom_boxplot (position = "dodge") +
  coord_flip() +
  labs(x = "CUT OF DIAMOND", y = "CARAT OF DIAMOND")
```



Explanation: By using position = "dodge", we can compare the individual values side by side.

Data munging and wrangling

Question: 1

```
# Finding the dataset tidy or not
table2
```

```
## # A tibble: 12 x 4
##    country      year type            count
##    <chr>       <int> <chr>           <int>
##  1 Afghanistan  1999 cases             745
##  2 Afghanistan  1999 population   19987071
##  3 Afghanistan  2000 cases            2666
##  4 Afghanistan  2000 population   20595360
##  5 Brazil       1999 cases           37737
##  6 Brazil       1999 population  172006362
```

```
##   7 Brazil       2000 cases           80488
##   8 Brazil       2000 population  174504898
##   9 China        1999 cases          212258
##  10 China        1999 population 1272915272
##  11 China        2000 cases          213766
##  12 China        2000 population 1280428583
```

```r
# It is not a tidy data, so below code makes it a tidy dataset
table2_tidy <- spread(table2, type, count)
# Display table2 in tidy way
table2_tidy
```

```
## # A tibble: 6 x 4
##   country      year  cases population
##   <chr>       <int>  <int>      <int>
## 1 Afghanistan  1999    745   19987071
## 2 Afghanistan  2000   2666   20595360
## 3 Brazil       1999  37737  172006362
## 4 Brazil       2000  80488  174504898
## 5 China        1999 212258 1272915272
## 6 China        2000 213766 1280428583
```

Answer: To make this data tidy, there needs to be one observation per row, which we can achieve with a "spread".

Question: 2

```r
# modify the diamonds dataset by adding an additional column
enhanced_diamonds <- diamonds %>% mutate(price_per_carat = price / carat)
```

Question: 3

```r
# finding the number of diamonds with price > 10000 and carat <1.5
diamond_target <- diamonds %>%
mutate (target_segment = (price > 10000 & carat < 1.5)) %>%
group_by(cut)
# finding the proportion
diamond_target %>%
  summarise(target_propotion = (sum(target_segment)/length(target_segment))*100,
target_count = sum(target_segment))
```

```
## # A tibble: 5 x 3
##   cut       target_propotion target_count
##   <ord>                <dbl>        <int>
## 1 Fair                 0.248            4
## 2 Good                 0.347           17
## 3 Very Good            1.28           155
## 4 Premium              1.25           173
## 5 Ideal                2.25           485
```

Answer:
As seen in the above dataset there are 485 ideal diamonds, and they comprise 2.25% of all ideal diamonds. This makes sense, since as the diamond is more ideal, small diamonds are more expensive. Similarly, most

fair diamonds won't have the same price as any of the others. It is interesting that very-good and premium diamonds are the same. Which implies that we are missing some other parameter, likely clarity, colour or some such variable.

EDA

Question: 1

```r
# Select year and month from the dataset with default sorting order
txhousing %>% select(year,month)
```

```
## # A tibble: 8,602 x 2
##     year month
##    <int> <int>
##  1  2000     1
##  2  2000     2
##  3  2000     3
##  4  2000     4
##  5  2000     5
##  6  2000     6
##  7  2000     7
##  8  2000     8
##  9  2000     9
## 10  2000    10
## # ... with 8,592 more rows
```

```r
#Select year and month from the dataset and finding the maximum year and month
txhousing %>% select(year,month) %>% arrange(desc(year), desc(month))
```

```
## # A tibble: 8,602 x 2
##     year month
##    <int> <int>
##  1  2015     7
##  2  2015     7
##  3  2015     7
##  4  2015     7
##  5  2015     7
##  6  2015     7
##  7  2015     7
##  8  2015     7
##  9  2015     7
## 10  2015     7
## # ... with 8,592 more rows
```

Answer:
The data is from Jan 2000 to July 2015

Question: 2

```r
# total number of cities in the dataset
total_cities <- txhousing %>% select(city) %>% unique()
count(total_cities)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1    46
```

Answer:
There are 46 unique cities in the dataset.

Question: 3

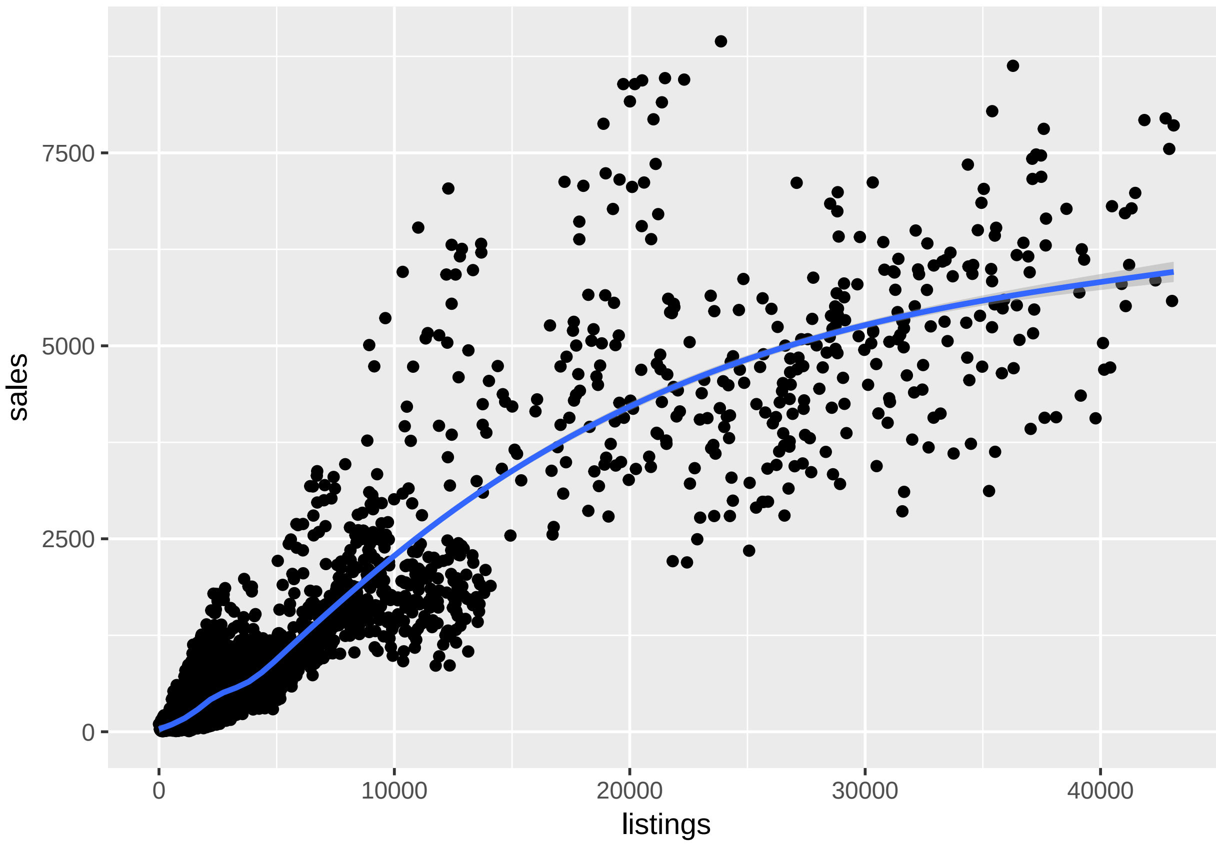
```r
# arrange the volumes in descending order and find the top city
txhousing %>% arrange(desc(volume))
```

```
## # A tibble: 8,602 x 9
##    city     year month sales     volume median listings inventory  date
##    <chr>   <int> <int> <dbl>      <dbl>  <dbl>    <dbl>     <dbl> <dbl>
##  1 Houston  2015     7  8945 2568156780 217600    23875       3.4 2016.
##  2 Houston  2015     6  8449 2490238594 222400    22311       3.2 2015.
##  3 Houston  2014     6  8391 2342443127 211200    19725       2.9 2014.
##  4 Houston  2014     7  8391 2278932511 199700    20214       3   2014.
##  5 Houston  2014     8  8167 2195184825 202400    20007       2.9 2015.
##  6 Houston  2013     7  8468 2168720825 187800    21497       3.3 2014.
##  7 Houston  2014     5  7877 2154791886 199300    18883       2.8 2014.
##  8 Houston  2013     5  8439 2121508529 186100    20526       3.3 2013.
##  9 Houston  2015     5  7357 2097957518 220100    21101       3.1 2015.
## 10 Houston  2013     8  8155 2083377894 186700    21366       3.3 2014.
## # ... with 8,592 more rows
```

Answer:
From the above dataset, Houston, in July/2015 had sales volume of $ 2.568 B.

Question: 4

```
# plotting the relation between listings and sales
ggplot(data = txhousing,mapping = aes(x=listings, y = sales)) +
geom_point() +
geom_smooth()
```



Answer:
From the above plot, we can assume that the sales are incresing along with the number of listings.

Question: 5

```r
# finding the cities with valid sales
valid_cities <- txhousing %>%
mutate(valid_sales = !is.na(sales)) %>%
group_by(city)
valid_cities # show valid cities
```

```
## # A tibble: 8,602 x 10
## # Groups:   city [46]
##    city      year month sales   volume median listings inventory  date
##    <chr>    <int> <int> <dbl>    <dbl>  <dbl>    <dbl>     <dbl> <dbl>
##  1 Abilene  2000     1    72  5380000  71400      701       6.3 2000
##  2 Abilene  2000     2    98  6505000  58700      746       6.6 2000.
##  3 Abilene  2000     3   130  9285000  58100      784       6.8 2000.
##  4 Abilene  2000     4    98  9730000  68600      785       6.9 2000.
##  5 Abilene  2000     5   141 10590000  67300      794       6.8 2000.
##  6 Abilene  2000     6   156 13910000  66900      780       6.6 2000.
##  7 Abilene  2000     7   152 12635000  73500      742       6.2 2000.
##  8 Abilene  2000     8   131 10710000  75000      765       6.4 2001.
##  9 Abilene  2000     9   104  7615000  64500      771       6.5 2001.
## 10 Abilene  2000    10   101  7040000  59300      764       6.6 2001.
## # ... with 8,592 more rows, and 1 more variable: valid_sales <lgl>
```

```r
# finding the proportions
proportions_cities <- valid_cities %>%
summarize(proportion = round(1 - sum(valid_sales)/length(valid_sales),4)) %>%
arrange(desc(proportion))
proportions_cities # city proportions
```

```
## # A tibble: 46 x 2
##    city                 proportion
##    <chr>                     <dbl>
##  1 South Padre Island       0.620
##  2 Kerrville                0.556
##  3 Midland                  0.401
##  4 Odessa                   0.385
##  5 San Marcos               0.246
##  6 Laredo                   0.192
##  7 Harlingen                0.134
##  8 Waco                     0.102
##  9 Texarkana                0.0909
## 10 Brazoria County          0.0749
## # ... with 36 more rows
```

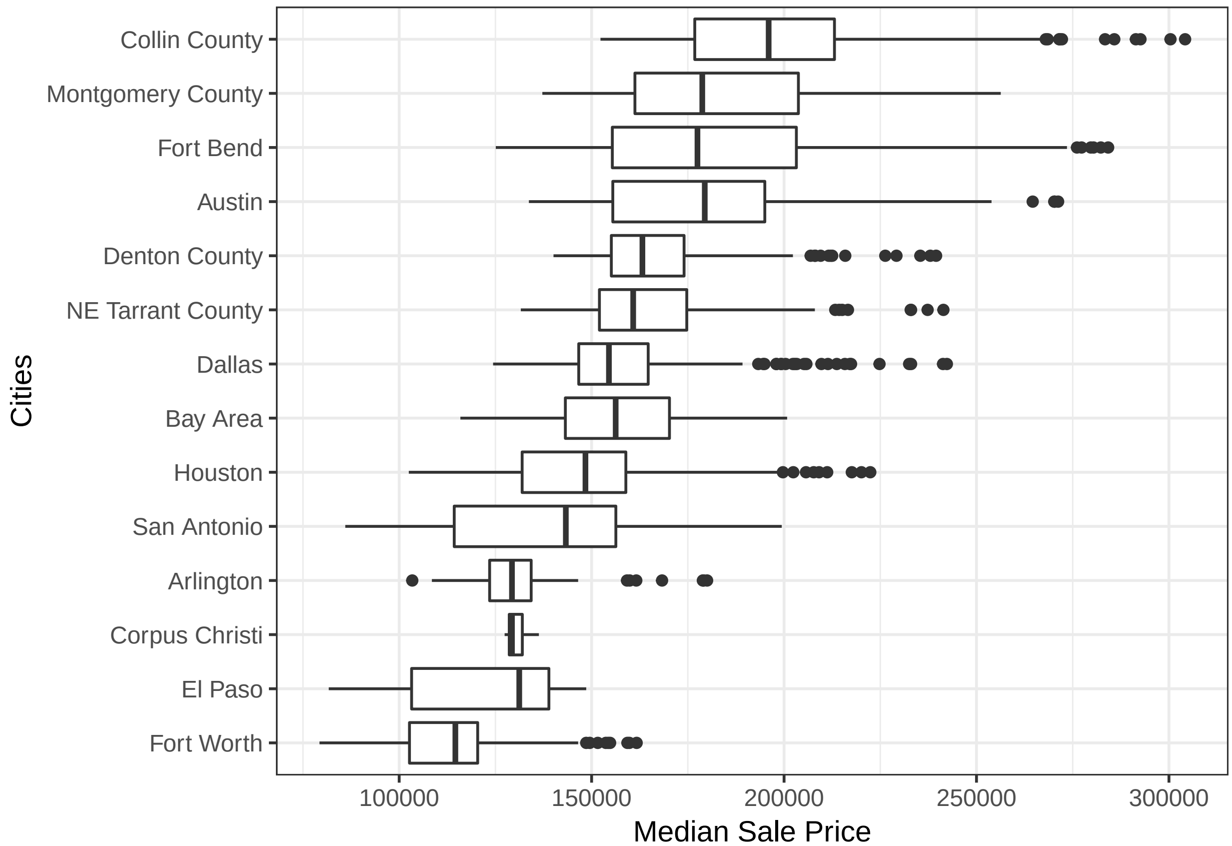
Question: 6

Are the distributions of the median sales price (column name median), when grouped by city, different? The same? Show your work.

```
# cities above 500 summarise by volume
txhousing %>% group_by(sales > 500)
```

```
## # A tibble: 8,602 x 10
## # Groups:   sales > 500 [3]
##    city     year month sales   volume median listings inventory  date
##    <chr>   <int> <int> <dbl>    <dbl>  <dbl>    <dbl>     <dbl> <dbl>
##  1 Abilene  2000     1    72  5380000  71400      701       6.3 2000
##  2 Abilene  2000     2    98  6505000  58700      746       6.6 2000.
##  3 Abilene  2000     3   130  9285000  58100      784       6.8 2000.
##  4 Abilene  2000     4    98  9730000  68600      785       6.9 2000.
##  5 Abilene  2000     5   141 10590000  67300      794       6.8 2000.
##  6 Abilene  2000     6   156 13910000  66900      780       6.6 2000.
##  7 Abilene  2000     7   152 12635000  73500      742       6.2 2000.
##  8 Abilene  2000     8   131 10710000  75000      765       6.4 2001.
##  9 Abilene  2000     9   104  7615000  64500      771       6.5 2001.
## 10 Abilene  2000    10   101  7040000  59300      764       6.6 2001.
## # ... with 8,592 more rows, and 1 more variable: `sales > 500` <lgl>
```

```
# cities above 500 distributions
cities_above500 <- txhousing %>% filter(sales > 500)
ggplot(data = cities_above500, mapping = aes(x = reorder(city,median,mean), y = median)) +
  geom_boxplot() +
  labs(x = 'Cities', y = 'Median Sale Price') +
  coord_flip() +
  theme_bw()
```

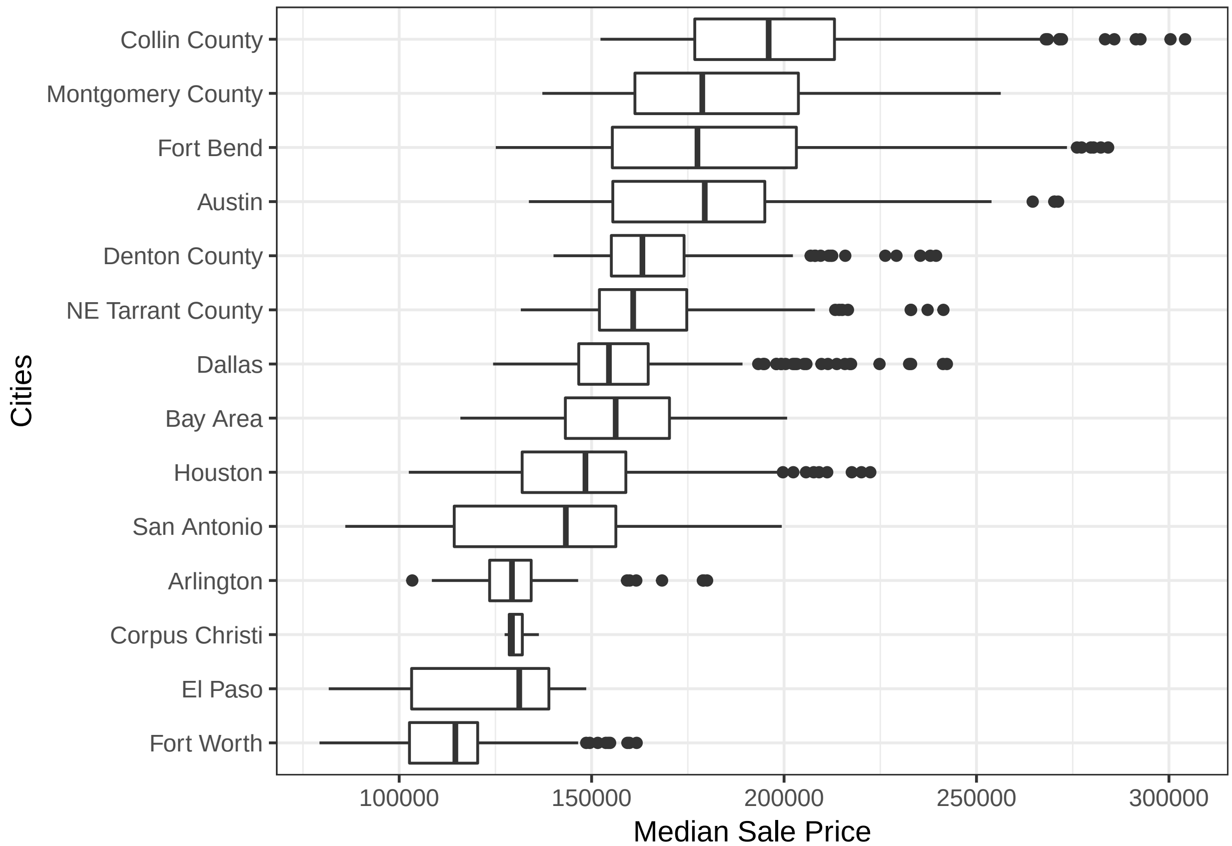


Answer: From the graph above interestingly we can observe that the median price is higher compared to other cities.

- San Antonio
- El Paso
- Austin

Any cities that stand out that you'd want to investigate further?

```
cities_above500_outliers <- txhousing %>% filter(sales > 500)
ggplot(data = cities_above500, mapping = aes(x = reorder(city,median,mean), y = median)) +
  geom_boxplot() +
  labs(x = 'Cities', y = 'Median Sale Price') +
  coord_flip() +
  theme_bw()
```

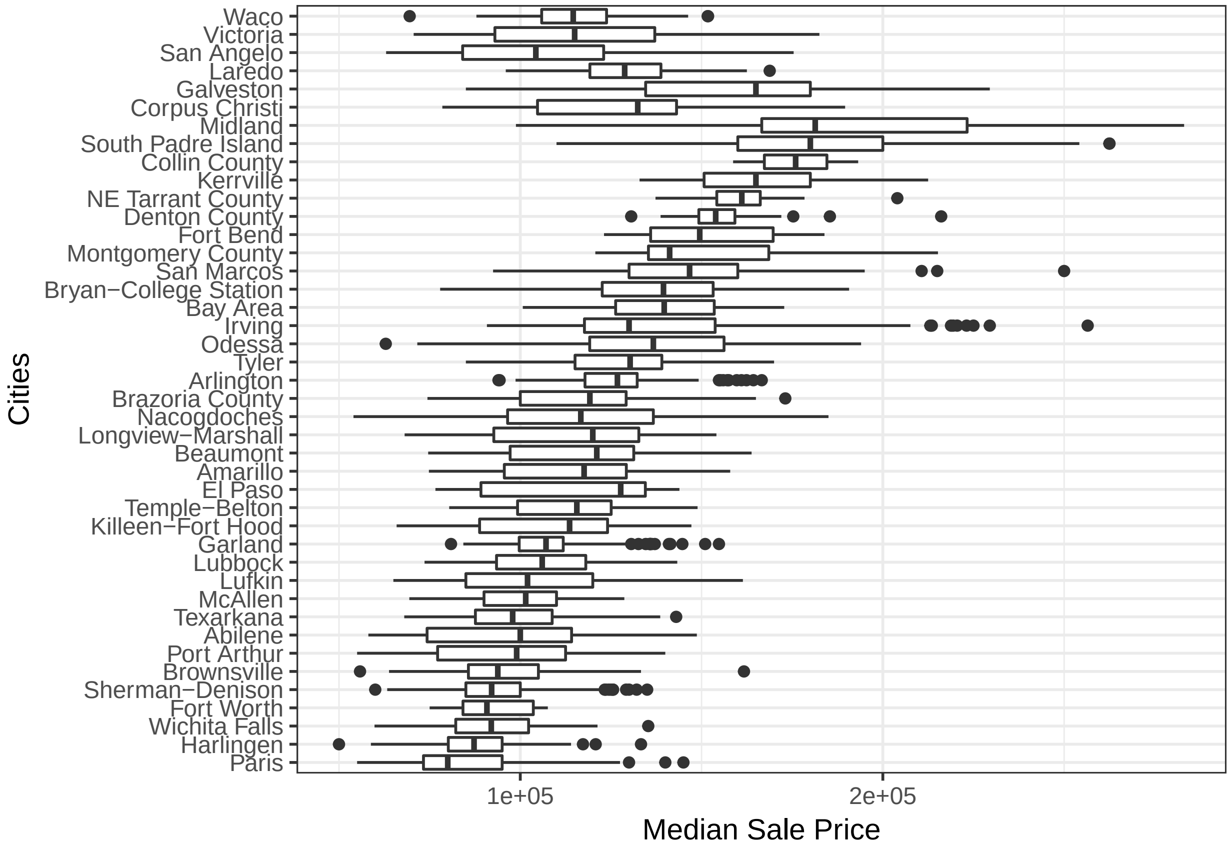


Answer: By the above plaot we can conclude that the following cities with lots of outliers needs to be investigated.

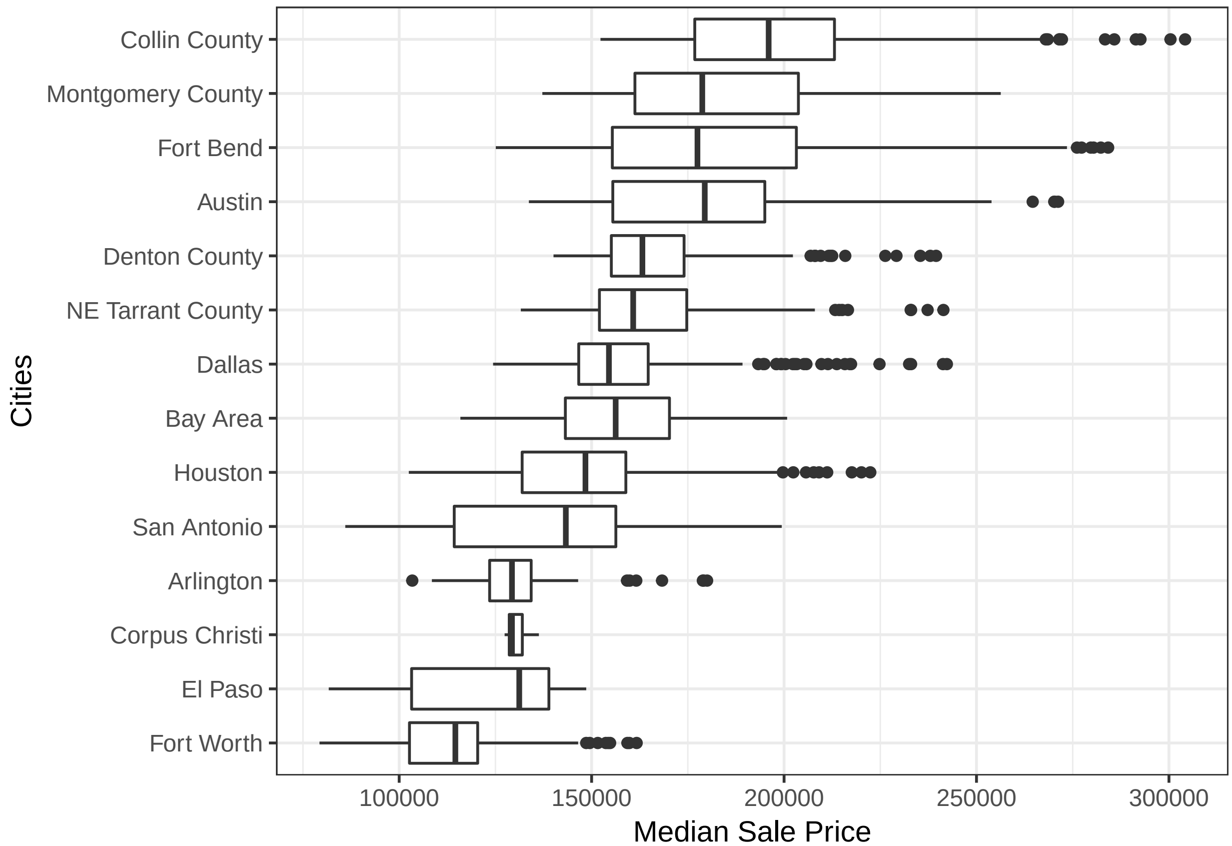
- Denton County
- Dallas
- Houston

Why might we want to filter out all cities and months with sales less than 500?

```r
# Let's take a look at cities below 500 sales
small_cities <- txhousing %>% filter (sales < 500)
ggplot(data = small_cities, mapping = aes(x = reorder(city,median,mean), y = median)) +
  geom_boxplot() +
  labs(x = 'Cities', y = 'Median Sale Price') +
  coord_flip() +
  theme_bw()
```



```r
# Let's take a look at cities abvove 500 sales
large_cities <- txhousing %>% filter (sales > 500)
ggplot(data = large_cities, mapping = aes(x = reorder(city,median,mean), y = median)) +
  geom_boxplot() +
  labs(x = 'Cities', y = 'Median Sale Price') +
  coord_flip() +
  theme_bw()
```



Answer: By looking at the above two box plot graph, it is clearly observed that the small cities with sales < 500 are high in number and they are just adding noise to the dataset.

Git and Github

Answer: Git hub location is added in the front page of this document