COMPSCIX 415.2 Homework 5/Midterm

Sanatan Das

March 1, 2018

Contents

Code and Documents Git Repository

All the work can be found in the below Git repository location:

https://github.com/sanatanonline/compscix-415-2-assignments

Load packages (prerequisites to run the code in this document)

```
library(tidyverse)
library(nycflights13)
library(dplyr)
```

RStudio and R Markdown (3 points)

1. Use markdown headers in your document to clearly separate each midterm question and add a table of contents to your document.

Answer

The following is the code used for markdown header and to add a table of contents in the document.

```
title: "COMPSCIX 415.2 Homework 5/Midterm"
author: "Sanatan Das"
date: "March 2, 2018"
output:
  html_document:
    number_sections: yes
    toc: yes
    toc_depth: 2
  pdf_document:
    toc: yes
    toc_depth: '2'
```

The tidyverse packages (3 points)

1. Can you name which package is associated with each task below?

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

Answer:

The below are the packages associated with each task below.

- Plotting - Plotting is done mainly using *ggplot2* package which is a core member of *tidyverse* package.
- Data munging/wrangling - This mainly uses base R packages, *tibble* (which is a core member of *tidyverse* package) and *dplyr* package. We have used the datasets from *nycflights13* package.

- Reshaping (speading and gathering) data - Reshaping of data is done using the functions of base R, *tibble* (which is a core member of *tidyverse* package) and *tidyr* packages.
- Importing/exporting data - Import and export of data mainly uses the functions of *readR* package which is a core member of *tidyverse* package.

2. Now can you name two functions that you've used from each package that you listed above for these tasks?

- **Plotting -**
- **Data munging/wrangling -**
- **Reshaping data -**
- **Importing/exporting data (note that readRDS and saveRDS are base R functions) -**

Answer:

The following are functions used from the packages listed above for the tasks.

- Plotting - *geom_point()* and *geom_smooth()*
- Data munging/wrangling - *filter()* and *arrange()*
- Reshaping data - *spread()* and *gather()*
- Importing/exporting data (note that readRDS and saveRDS are base R functions) - *read_csv()* and *write_csv()*

R Basics (1.5 points)

1. Fix this code with the fewest number of changes possible so it works:

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

Answer

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

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

Explanation: '!' is not allowed in a variable name. If the code is executed, R gives the below error:

Error: unexpected '!' in "My_data.name____is.too00ooLong!"

2. Fix this code so it works:

```
my_string <- C('has', 'an', 'error', 'in', 'it)
```

Answer

There are two issues in the above code.

- my_string <- C(, in this code "C" is in uppercase whereas it should be lowercase. R is case sensitive.
- The last element is not enslosed by quote. So R can not parse it and throws parse error.

The correct code should be as below:

```
my_string <- c('has', 'an', 'error', 'in', 'it')
my_string
```

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

3. Look at the code below and comment on what happened to the values in the vector.

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

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

Answer

In R, a vector is a sequence of data elements of the same basic type. So it automatically converts the numbers to character String and displays enclosed in double quotes.

Data import/export (3 points)

1. Download the rail_trail.txt file from Canvas (in the Midterm Exam section here) and successfully import it into R. Prove that it was imported successfully by including your import code and taking a glimpse of the result.

Answer

```
# Read from rail_trail.txt file
rail_trail <- read_csv("C:/view/opt/apps/git/R/compscix-415-2-assignments/rail_trail.txt")
```

```
## Parsed with column specification:
## cols(
##   `hightemp|lowtemp|avgtemp|spring|summer|fall|cloudcover|precip|volume|weekday` = col_character()
## )
```

```
# glimpse rail_trail
glimpse(rail_trail)
```

```
## Observations: 90
## Variables: 1
## $ `hightemp|lowtemp|avgtemp|spring|summer|fall|cloudcover|precip|volume|weekday` <chr> ...
```

2. Export the file into an R-specific format and name it "rail_trail.rds". Make sure you define the path correctly so that you know where it gets saved. Then reload the file. Include your export and import code and take another glimpse.

Answer

```
# Read from rail_trail.txt file
rail_trail <- read_csv("C:/view/opt/apps/git/R/compscix-415-2-assignments/rail_trail.txt")
```

```
## Parsed with column specification:
## cols(
##   `hightemp|lowtemp|avgtemp|spring|summer|fall|cloudcover|precip|volume|weekday` = col_character()
## )
```

```
# glimpse rail_trail
glimpse(rail_trail)
```

```
## Observations: 90
## Variables: 1
## $ `hightemp|lowtemp|avgtemp|spring|summer|fall|cloudcover|precip|volume|weekday` <chr> ...
```

```
# Write to rail_trail.rds
saveRDS(rail_trail, "rail_trail.rds")
```

```
# load rail_trail.rds
rail_trail2 = readRDS("C:/view/opt/apps/git/R/compscix-415-2-assignments/rail_trail.rds")

# glimpse rail_trail
glimpse(rail_trail2)
```

```
## Observations: 90
## Variables: 1
## $ `hightemp|lowtemp|avgtemp|spring|summer|fall|cloudcover|precip|volume|weekday` <chr> ...
```

Visualization (6 points)

1. Critique this graphic: give only three examples of what is wrong with this graphic. Be concise.

Note: Please refer to the below link for the above mentioned graphic.

https://github.com/sanatanonline/compscix-415-2-assignments/blob/master/compscix4152_hw_5.pdf

Answer

This graphic has multiple issues which creates wrong impressions of the data visualization. The major three wrong representations are:

- This is not a standard statistical chart/plot representation (is it a bubble chart?). The numeric values definitely does not match the size of the images.
- What are those numbers represents? Are they percentage/total number of responders or what… not clear.
- What are those colors represent? No legend. Not clear.

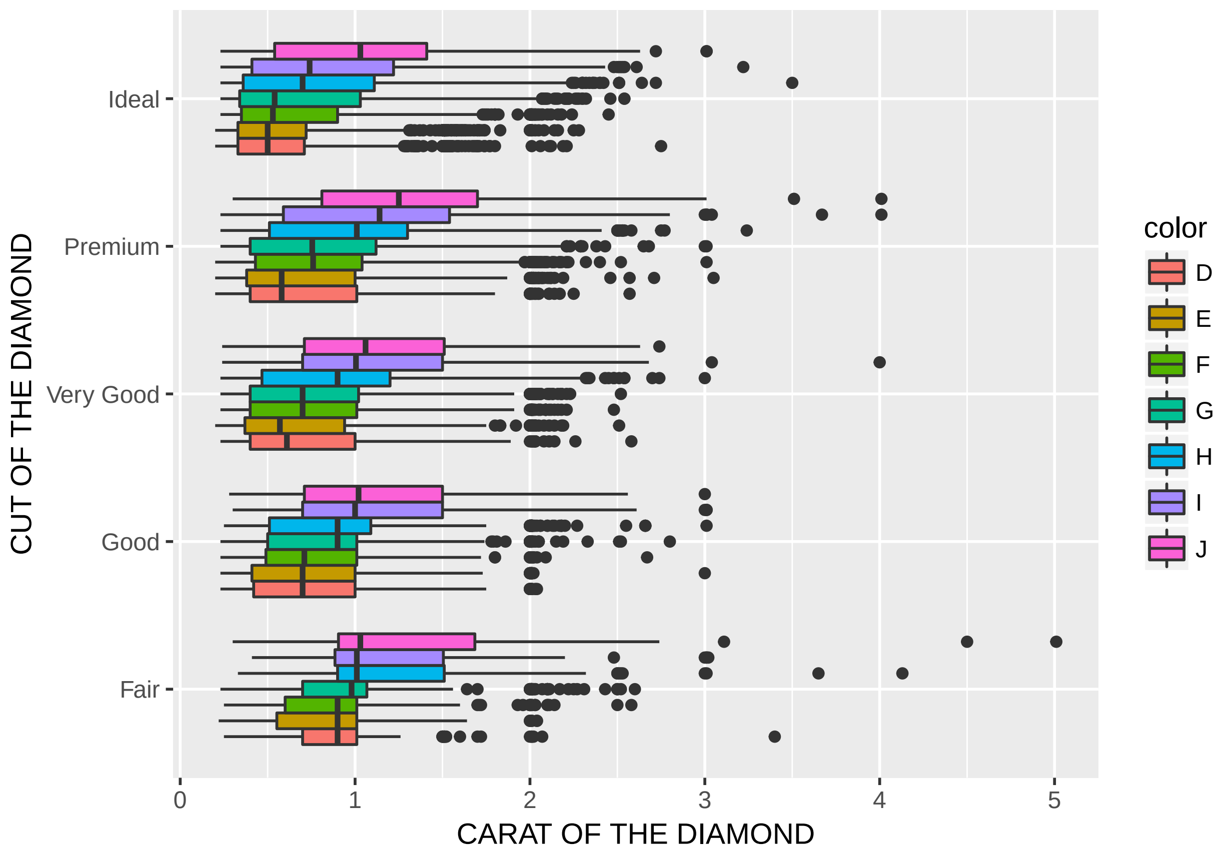
2. Reproduce this graphic using the diamonds data set.

Note: Please refer to the below link for the above mentioned graphic.

https://github.com/sanatanonline/compscix-415-2-assignments/blob/master/compscix4152_hw_5.pdf

Answer

```
ggplot(data = diamonds, aes(x = cut, y = carat, fill = color)) +
  geom_boxplot() +
  coord_flip() +
  labs(x="CUT OF THE DIAMOND", y="CARAT OF THE DIAMOND")
```



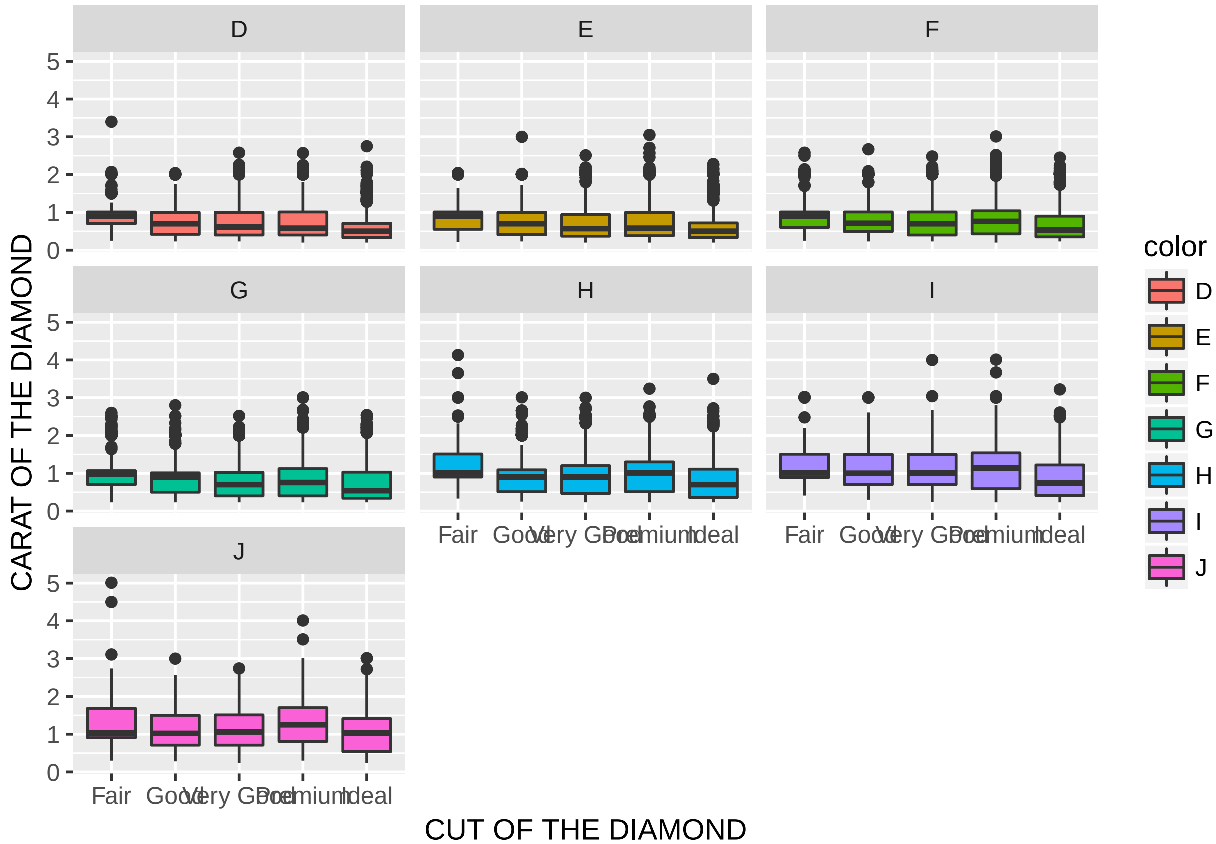
I think the box plot will be like above. But I am not sure how to get them overlapped.

3. The previous graphic is not very useful. We can make it much more useful by changing one thing about it. Make the change and plot it again.

Answer

The graphic is not very useful because the box plots are overlapped. We can make it useful by making the below change:

```
ggplot(data = diamonds, aes(x = cut, y = carat, fill = color)) +
  geom_boxplot() +
  facet_wrap(~color) +
  labs(x="CUT OF THE DIAMOND", y="CARAT OF THE DIAMOND")
```



Data munging and wrangling (6 points)

1. Is this data "tidy"? If yes, leave it alone and go to the next problem. If no, make it tidy. Note: this data set is called table2 and is available in the tidyverse package. It should be ready for you to use after you've loaded the tidyverse package.

```
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
```

Answer

This is not a tidy data. This datast intermingles the values of population and cases in the same columns. As a result, we would need to untangle the values whenever we want to work with each variable separately.

The *key* column contains only keys (and not just because the column is labelled *key*). Conveniently, the *value* column contains the values associated with those keys.

We can use the *spread()* function to tidy this layout. So the tidy form of the dataset would be like below:

```
spread(table2, type, count)
```

```
## # 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
```

2. Create a new column in the diamonds data set called price__per__carat that shows the price of each diamond per carat (hint: divide). Only show me the code, not the output.

We can do this using the code below:

```
mutate(diamonds, price_per_carat <- price / carat)
```

3. For each cut of diamond in the diamonds data set, how many diamonds, and what proportion, have a price > 10000 and a carat < 1.5? There are several ways to get to an answer, but your solution must use the data wrangling verbs from the tidyverse in order to get credit.

- **Do the results make sense? Why?**
- **Do we need to be wary of any of these numbers? Why?**

Answer

```
filter(diamonds, price > 10000, carat < 1.5)
```

```
## # A tibble: 834 x 10
##    carat cut       color clarity depth table price     x     y     z
##    <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  1.03 Ideal     E     VVS2     60.6  59.0 10003  6.50  6.53  3.95
##  2  1.23 Very Good G     VVS2     60.6  55.0 10004  6.93  7.02  4.23
##  3  1.25 Ideal     F     VS2      61.6  55.0 10006  6.93  6.96  4.28
##  4  1.21 Very Good F     VS1      62.3  58.0 10009  6.76  6.85  4.24
##  5  1.01 Fair      D     SI2      64.6  58.0 10011  6.25  6.20  4.02
##  6  1.05 Ideal     F     VVS2     60.5  55.0 10011  6.67  6.58  4.01
##  7  1.35 Premium   G     VS1      62.1  59.0 10012  7.06  7.02  4.37
##  8  1.13 Ideal     F     VS1      60.9  57.0 10016  6.73  6.76  4.11
##  9  1.21 Premium   F     VS1      62.6  59.0 10018  6.81  6.76  4.25
## 10  1.01 Very Good F     VVS1     62.9  57.0 10019  6.35  6.41  4.01
## # ... with 824 more rows
```

```
diamonds2 <- diamonds %>%
  group_by(cut) %>%
  summarise(prop = sum(price > 10000, carat < 1.5)/ n()) %>%
  arrange(cut)

print(tbl_df(diamonds2))
```

```
## # A tibble: 5 x 2
##   cut        prop
##   <ord>     <dbl>
## 1 Fair      0.889
## 2 Good      0.946
## 3 Very Good 0.979
## 4 Premium   0.969
## 5 Ideal     1.01
```

This table shows almost the whole proportion of each cut have price more than 10000 when carat < 1.5? Confusing.

EDA (6 points)

Take a look at the txhousing data set that is included with the ggplot2 package and answer these questions:

1. During what time period is this data from?

2. How many cities are represented?

3. Which city, month and year had the highest number of sales?

4. What kind of relationship do you think exists between the number of listings and the number of sales? Check your assumption and show your work.

5. What proportion of sales is missing for each city?

6. Looking at only the cities and months with greater than 500 sales:

- **Are the distributions of the median sales price (column name median), when grouped by city, different? The same? Show your work.**
- **Any cities that stand out that you'd want to investigate further?**
- **Why might we want to filter out all cities and months with sales less than 500?**

Answer

To do the EDA on *txhousing* data, first we take a quick look at the dataset. We can use ?txhousing for help to understand the variabls.

```
txhousing
```

```
## # 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 Abilene  2000     1  72.0  5380000  71400      701      6.30  2000
##  2 Abilene  2000     2  98.0  6505000  58700      746      6.60  2000
##  3 Abilene  2000     3 130    9285000  58100      784      6.80  2000
##  4 Abilene  2000     4  98.0  9730000  68600      785      6.90  2000
##  5 Abilene  2000     5 141   10590000  67300      794      6.80  2000
##  6 Abilene  2000     6 156   13910000  66900      780      6.60  2000
##  7 Abilene  2000     7 152   12635000  73500      742      6.20  2000
##  8 Abilene  2000     8 131   10710000  75000      765      6.40  2001
##  9 Abilene  2000     9 104    7615000  64500      771      6.50  2001
## 10 Abilene  2000    10 101    7040000  59300      764      6.60  2001
## # ... with 8,592 more rows
```

From the above result, we see its a dataset of 9 variables with 8602 observations. Now we will do the analysis to answer the above questions.

1. During what time period is this data from?

```
arrange(txhousing, year, month)
```

```
## # 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 Abilene        2000     1   72.0   5.38e6  71400      701      6.30  2000
##  2 Amarillo       2000     1  102     8.86e6  80000      972      5.30  2000
##  3 Arlington      2000     1  241     2.62e7  94000     1417      3.70  2000
##  4 Austin         2000     1 1025     1.73e8 133700     3084      2.00  2000
##  5 Bay Area       2000     1  244     2.93e7 100700     1766      4.30  2000
##  6 Beaumont       2000     1   97.0   1.01e7  82100      876      6.10  2000
##  7 Brazoria Co~   2000     1   55.0   5.24e6  74400      512      5.90  2000
##  8 Brownsville    2000     1   NA    NA          NA      400      9.10  2000
##  9 Bryan-Colle~   2000     1   61.0   5.61e6  77900      498      4.20  2000
## 10 Collin Coun~   2000     1  464     9.48e7 158700     2844      4.00  2000
## # ... with 8,592 more rows
```

```
arrange(txhousing, desc(year), desc(month))
```

```
## # 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 Abilene        2015     7   268   4.58e7 148700      986      5.00  2016
##  2 Amarillo       2015     7   354   6.23e7 149700     1247      4.50  2016
##  3 Arlington      2015     7   605   1.25e8 178900      752      1.70  2016
##  4 Austin         2015     7  3466   1.15e9 264600     7913      3.00  2016
##  5 Bay Area       2015     7   849   1.97e8 200800     2144      3.20  2016
##  6 Beaumont       2015     7   318   5.29e7 139300     1561      6.40  2016
##  7 Brazoria Co~   2015     7    NA  NA          NA       NA     NA     2016
##  8 Brownsville    2015     7    NA  NA          NA       NA     NA     2016
##  9 Bryan-Colle~   2015     7   414   9.04e7 190700      894      3.30  2016
## 10 Collin Coun~   2015     7  1861   6.14e8 292600     2809      2.10  2016
## # ... with 8,592 more rows
```

From the above results, we see that the data is collected monthly from Jan 2000 to July 2015

2. How many cities are represented?

```
count(unique(txhousing[,1]))
```

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

There are 46 cities represented in *txhousing* dataset.

3. Which city, month and year had the highest number of sales?

In this dataset *sales* variable represents the **number of sales.** So we arrange the dataset in descending order by sales.

```
arrange(txhousing, desc(sales))
```

```
## # A tibble: 8,602 x 9
```

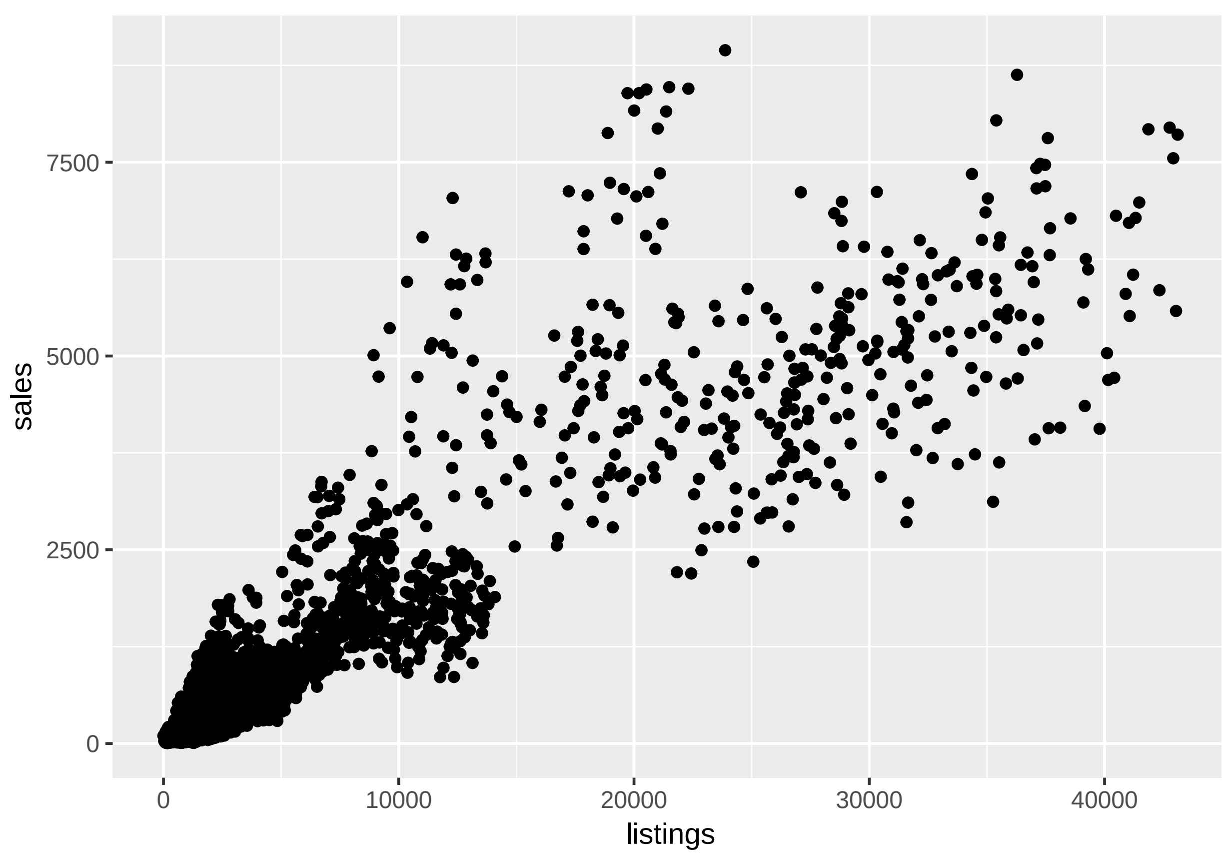
```
##    city     year month sales     volume median listings inventory  date
##    <chr>   <int> <int> <dbl>      <dbl>  <dbl>    <dbl>     <dbl> <dbl>
##  1 Houston  2015     7  8945 2568156780 217600    23875      3.40  2016
##  2 Houston  2006     6  8628 1795898108 155200    36281      5.60  2006
##  3 Houston  2013     7  8468 2168720825 187800    21497      3.30  2014
##  4 Houston  2015     6  8449 2490238594 222400    22311      3.20  2015
##  5 Houston  2013     5  8439 2121508529 186100    20526      3.30  2013
##  6 Houston  2014     6  8391 2342443127 211200    19725      2.90  2014
##  7 Houston  2014     7  8391 2278932511 199700    20214      3.00  2014
##  8 Houston  2014     8  8167 2195184825 202400    20007      2.90  2015
##  9 Houston  2013     8  8155 2083377894 186700    21366      3.30  2014
## 10 Houston  2006     5  8040 1602621368 151200    35398      5.50  2006
## # ... with 8,592 more rows
```

In the above result we see, Houston had the highest number of sales 8945 in July (month 7), 2015. This city had maximum volume (total value of sales) too.

4. What kind of relationship do you think exists between the number of listings and the number of sales? Check your assumption and show your work.

```r
ggplot(data = txhousing) +
  geom_point(mapping = aes(x = listings, y = sales))
```

```
## Warning: Removed 1426 rows containing missing values (geom_point).
```

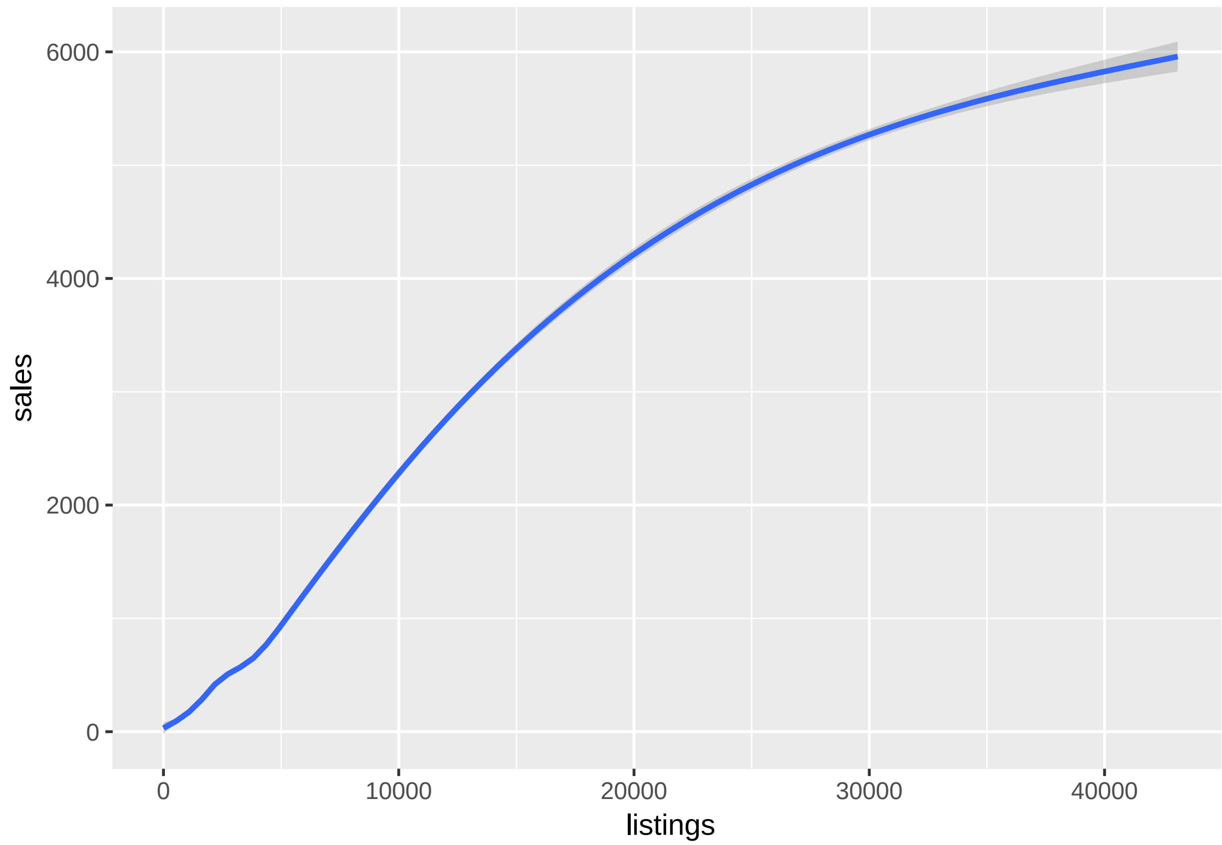


In the above scatter plot we see that there is a relationship between the number of listings and the number of sales. When there is more number of listings, number of sales increases. The below plot confirms the trend.

```r
ggplot(data = txhousing) +
  geom_smooth(mapping = aes(x = listings, y = sales))
```

```
## `geom_smooth()` using method = 'gam'

## Warning: Removed 1426 rows containing non-finite values (stat_smooth).
```



5. What proportion of sales is missing for each city?

We can find out the proportion of sales is missing for each city using following code.

```r
missing_sales_prop_per_city <- txhousing %>%
  group_by(city) %>%
  summarise(prop = sum(is.na(sales))/ n()) %>%
  arrange(city)

# we have to print 46 rows for 46 cities
print(tbl_df(missing_sales_prop_per_city), n=46)
```

```
## # A tibble: 46 x 2
##    city                        prop
##    <chr>                      <dbl>
##  1 Abilene                  0
##  2 Amarillo                 0
##  3 Arlington                0
##  4 Austin                   0
##  5 Bay Area                 0
##  6 Beaumont                 0
```

```
##  7 Brazoria County       0.0749
##  8 Brownsville           0.0107
##  9 Bryan-College Station 0
## 10 Collin County         0
## 11 Corpus Christi        0.00535
## 12 Dallas                0
## 13 Denton County         0
## 14 El Paso               0
## 15 Fort Bend             0
## 16 Fort Worth            0
## 17 Galveston             0.00535
## 18 Garland               0
## 19 Harlingen             0.134
## 20 Houston               0
## 21 Irving                0
## 22 Kerrville             0.556
## 23 Killeen-Fort Hood     0.00535
## 24 Laredo                0.193
## 25 Longview-Marshall     0.0642
## 26 Lubbock               0.00535
## 27 Lufkin                0
## 28 McAllen               0.0107
## 29 Midland               0.401
## 30 Montgomery County     0
## 31 Nacogdoches           0.0588
## 32 NE Tarrant County     0
## 33 Odessa                0.385
## 34 Paris                 0
## 35 Port Arthur           0.0107
## 36 San Angelo            0
## 37 San Antonio           0
## 38 San Marcos            0.246
## 39 Sherman-Denison       0
## 40 South Padre Island    0.620
## 41 Temple-Belton         0.0588
## 42 Texarkana             0.0909
## 43 Tyler                 0
## 44 Victoria              0
## 45 Waco                  0.102
## 46 Wichita Falls         0
```

6. Looking at only the cities and months with greater than 500 sales:

- **Are the distributions of the median sales price (column name median), when grouped by city, different? The same? Show your work.**
- **Any cities that stand out that you'd want to investigate further?**
- **Why might we want to filter out all cities and months with sales less than 500?**

First filter out the months with less than or equal to 500 as we are going to look at only the cities and months with greater than 500 sales.

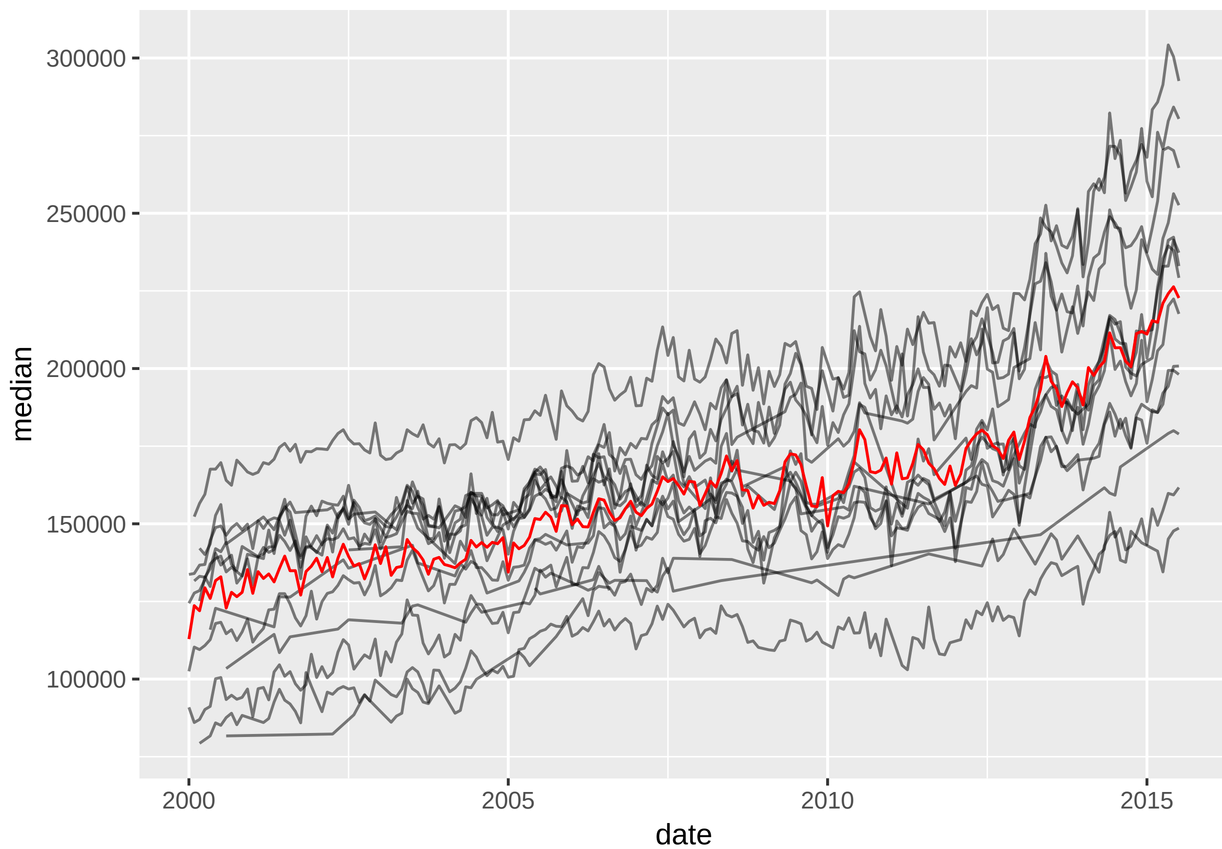
```
newtxhousing <- filter(txhousing, sales > 500)

newtxhousing
```

```
## # A tibble: 1,883 x 9
##    city        year month sales   volume median listings inventory  date
##    <chr>      <int> <int> <dbl>    <dbl>  <dbl>    <dbl>     <dbl> <dbl>
##  1 Arlington  2000     8   507 60875199 103400     1417      3.50  2001
##  2 Arlington  2001     5   536 69878959 114400     1592      3.70  2001
##  3 Arlington  2001     6   534 67744182 108500     1627      3.80  2001
##  4 Arlington  2001     8   505 65080743 113600     1616      3.70  2002
##  5 Arlington  2002     5   503 67240236 116100     1741      3.90  2002
##  6 Arlington  2002     7   509 66954143 119100     1925      4.40  2002
##  7 Arlington  2003     5   502 67131982 118000     2544      5.90  2003
##  8 Arlington  2003     7   524 73194692 123500     2799      6.50  2004
##  9 Arlington  2003     8   531 72397143 123900     2801      6.40  2004
## 10 Arlington  2004     5   527 72401436 118300     2922      6.50  2004
## # ... with 1,873 more rows
```

And we see there are such 1883 records.

```r
ggplot(newtxhousing, aes(date, median)) +
  geom_line(aes(group = city), alpha = 1/2) +
  geom_line(stat = "summary", fun.y = "mean", colour = "red")
```



Looking at the above plot, we can see the some of the cities (I guess big cities like Houston) has more sales than other cities.

```r
#OutVals = boxplot(newtxhousing$sales)$out
#which(newtxhousing %in% OutVals)

OutVals <- tibble(boxplot(newtxhousing$median, plot=FALSE)$out)
```

```r
OutVals[1]
```

```
## # A tibble: 56 x 1
##    `boxplot(newtxhousing$median, plot = FALSE)$out`
##                                               <dbl>
##  1                                           248900
##  2                                           246900
##  3                                           245700
##  4                                           245300
##  5                                           253900
##  6                                           270300
##  7                                           271200
##  8                                           270200
##  9                                           264600
## 10                                           252600
## # ... with 46 more rows
```

```r
outliers_cities <- subset(newtxhousing, newtxhousing$median %in% OutVals)

outliers_cities
```

```
## # A tibble: 0 x 9
## # ... with 9 variables: city <chr>, year <int>, month <int>, sales <dbl>,
## #   volume <dbl>, median <dbl>, listings <dbl>, inventory <dbl>,
## #   date <dbl>
```

```r
filter(newtxhousing, newtxhousing$median %in% OutVals[1])
```

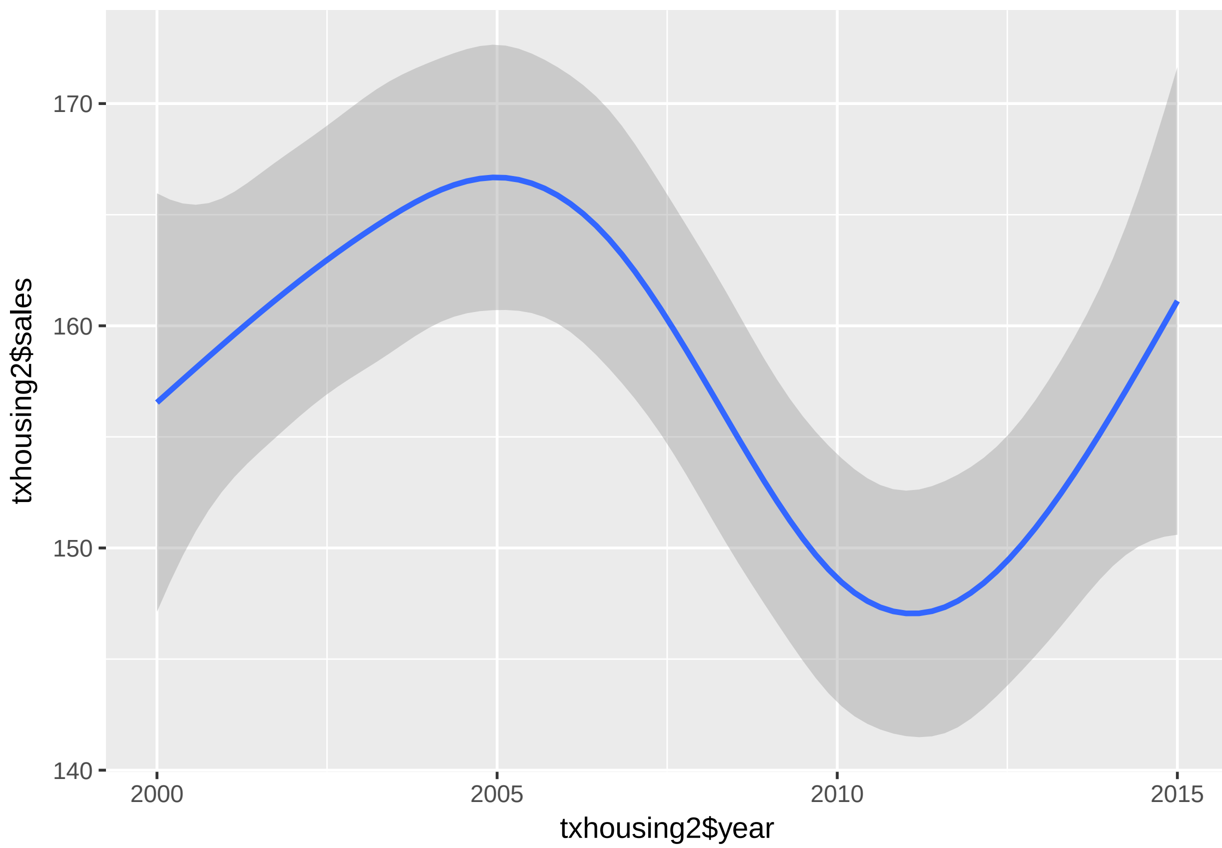
```
## # A tibble: 0 x 9
## # ... with 9 variables: city <chr>, year <int>, month <int>, sales <dbl>,
## #   volume <dbl>, median <dbl>, listings <dbl>, inventory <dbl>,
## #   date <dbl>
```

First lets see the records having sales less than 500.

```r
txhousing2 <- filter(txhousing, sales < 500)

ggplot(data = txhousing2) +
  geom_smooth(mapping = aes(x = txhousing2$year, y = txhousing2$sales))
```

```
## `geom_smooth()` using method = 'gam'
```



There is no much change for the records having sales less than 500.

Git and Github (1.5 points)

To demonstrate your use of git and Github, at the top of your document put a hyperlink to your Github repository.

Answer

All the work is pushed to Github repository. The repository URL is below.

https://github.com/sanatanonline/compscix-415-2-assignments

End of Homework 5/Midterm