Homework 5/Midterm

Bryan Hee

March 3, 2018

Contents

The Tidyverse Packages **1**
Packages 1
Functions 2

R Basics **2**
Code Edit 1 2
Code Edit 2 2
Vector Storage 2

Data Import/Export **2**
Rail Trail Import 2
Rail Trail Export 3

Visualization **3**
Mrs. President Graph Critique 3
Reproduced Plot 4
More Useful Plot 4

Data Munging and Wrangling **5**
Table2 5
Price per Carat 5
Pricy and Small Diamonds 6

EDA - Exploratory Data Analysis **7**
Texas Housing Sales Dataset Time Period 7
Number of Texas Cities in Dataset 7
Highest Texas Housing Sales 8
Texas Housing Sales Efficiency 8
Missing Sales 9
Median Home Prices in Months w/ Sales > 500 10

Github Repository: https://github.com/Bhee555/compscix-415-2-assignments

The Tidyverse Packages

Packages

The following list represents tasks which can be accomplished using the tidyverse library and the corresponding package that is associated with each task:

a. Plotting - ggplot2()

b. Data munging/wrangling - dplyr()

c. Reshaping (spreading and gathering) data - tidyr()

d. Importing/exporting data - readr()

Functions

The following are 2 functions that we've used from each package listed above:

a. ggplot2() - geom_point(), coord_flip()
b. dplyr() - arrange(), mutate()
c. tidyr() - spread(), gather()
d. readr() - read_delim(), write_rds()

R Basics

Code Edit 1

The exclamation mark from the original object name was removed

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

Code Edit 2

The "it" was missing a closing parenthesis. The c defining the struct was capitalized.

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

Vector Storage

The entire vector is stored as characters rather than numerics (this differed from my expectation that only 3 and 4 would be stored as characters).

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

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

```
is.numeric(my_vector[1])
```

```
## [1] FALSE
```

Data Import/Export

Rail Trail Import

Import Rail Trail.txt:

```
holder <- 'C:/Users/BryanHee/OneDrive - stok LLC/Intro to Data Science/HW Assignments/Assignment 5/rail
rail_trail <- read_delim(file = holder, '|')
rm(holder)
glimpse(rail_trail)
```

```
## 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, ...
```

Rail Trail Export

Export Rail Trail.rds:

```
write_rds(rail_trail, path = 'C:/Users/BryanHee/OneDrive - stok LLC/Intro to Data Science/HW Assignments
read_rds('C:/Users/BryanHee/OneDrive - stok LLC/Intro to Data Science/HW Assignments/Assignment 5/rail_
glimpse()
```

```
## 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

Mrs. President Graph Critique

The first critique of the graph provided is that on each of the age bins, the data is provided in percentage of respondents, however the overall data doesn't add up to 100%. I would assume that the remainder percentages of survey respondents replied with answers other than Yes/No, however, I believe it is the job of the data scientist to remove the need for any and all assumptions, such as this.

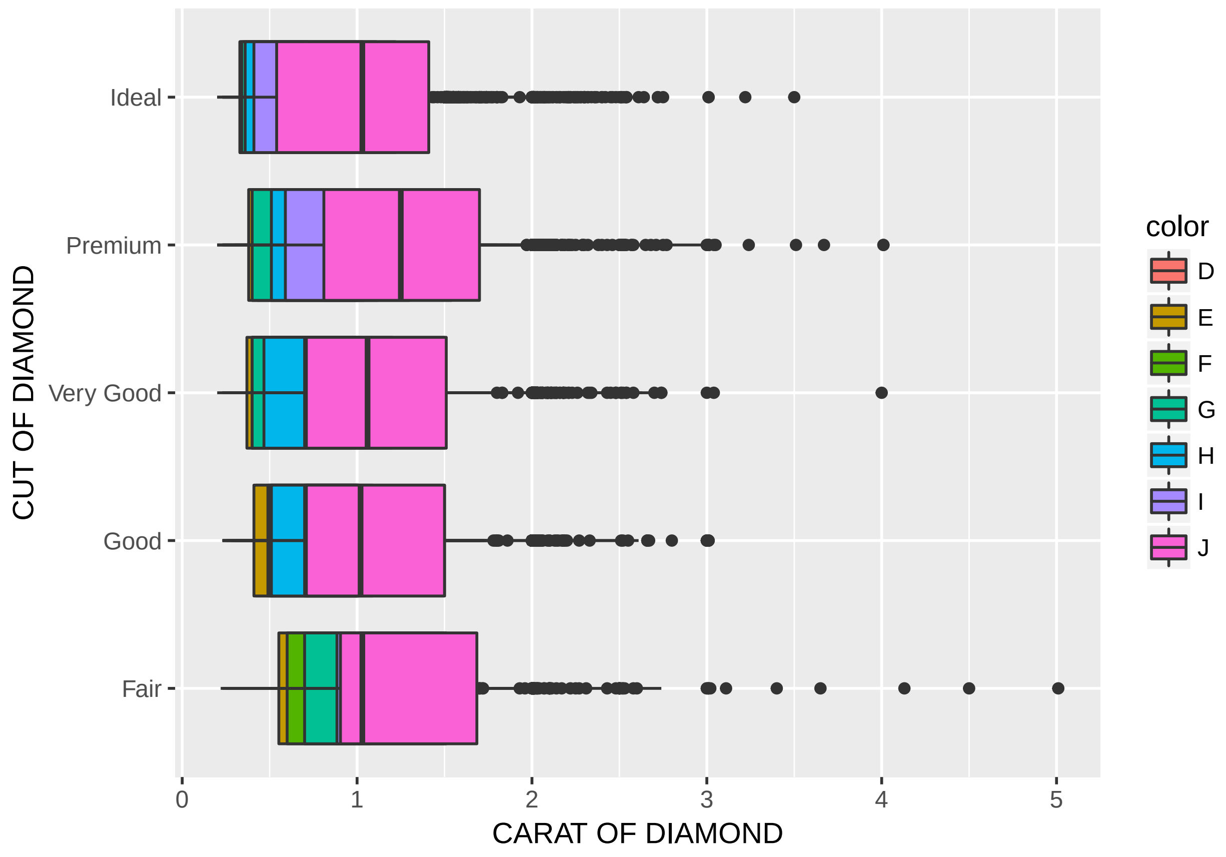
The second issue i have with the graphic is that there is no indication of the sample sizes for each of the respondent bins.

A third feature that i feel this graphic failed on, are the arbitrary color choices made for the men and women respondent bins. All of the age groups were black, yet genders get two colors? It took me a second to interpret the data because of this.

Reproduced Plot

Here is the reproduced plot from the diamonds dataset.

```
ggplot(diamonds,mapping = aes(cut, carat, fill = color)) +
  geom_boxplot(position = "identity") +
  labs(x = 'CUT OF DIAMOND', y = 'CARAT OF DIAMOND') +
  coord_flip()
```

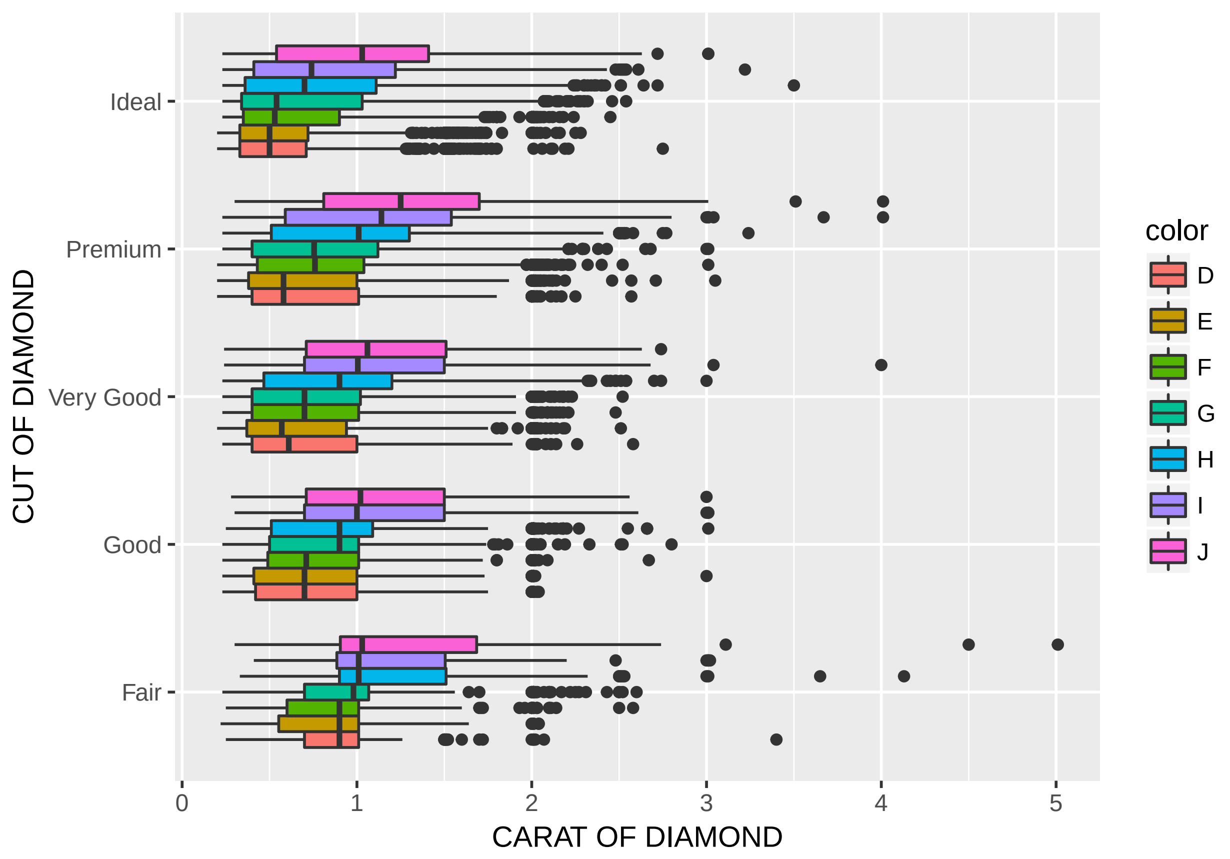


```
#out of all the questions, this took me the longest :), should have known position = "identity"
```

More Useful Plot

I think the following is a more useful plot because I can now visually see each color's carat size distribution while still being grouped by cut. By getting rid of position = "identity" the geom boxplot automatically separates each color.

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



Data Munging and Wrangling

Table2

No table2 is not currently tidy because the type column includes two variables.

```
spread(table2, key = 'type', value = '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
```

Price per Carat

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

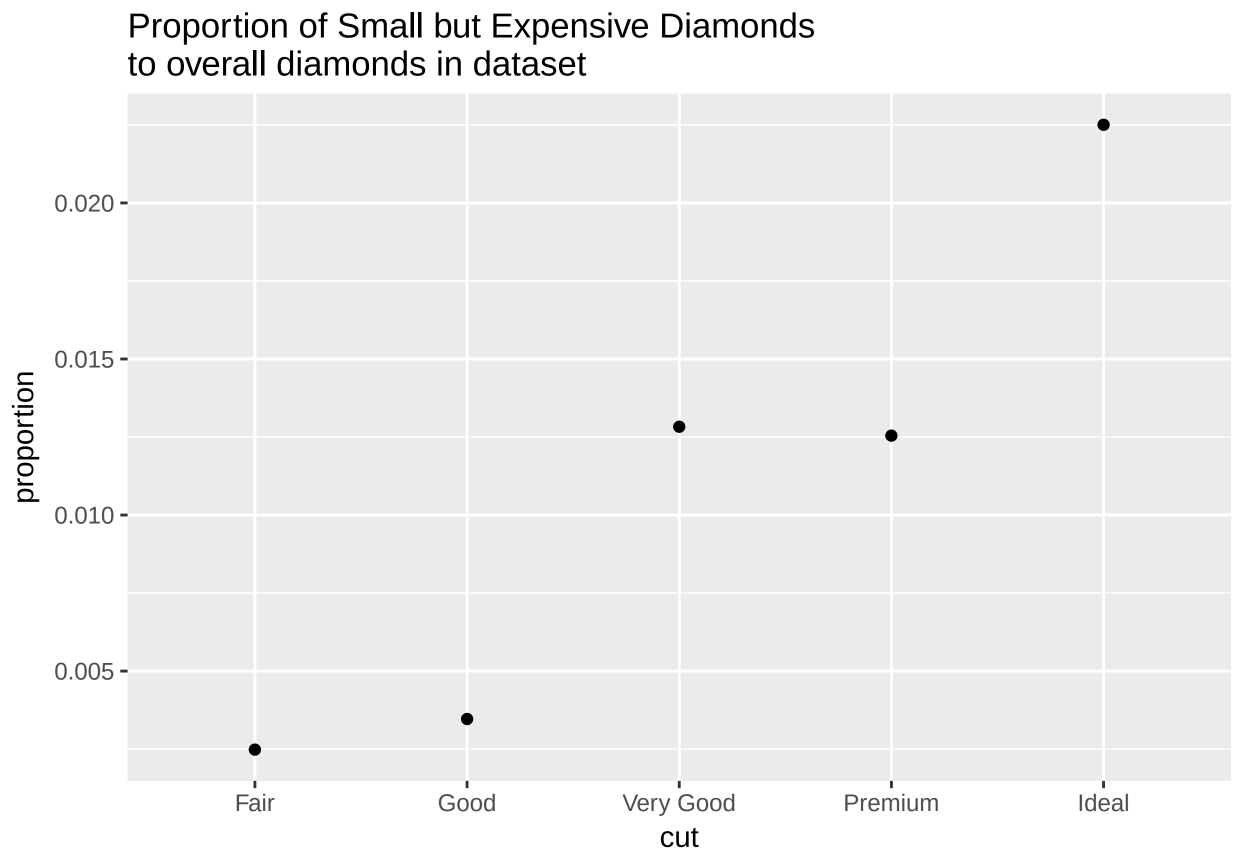
Pricy and Small Diamonds

The following data represents the number of diamonds, grouped by cut, that cost more than $10,000 and are smaller than 1.5 carats. For the most part, the distribution makes sense to me (see the plot below). This is because as the quality of cut goes up, you would also expect the price of the same size diamond to go up. However it is interesting to me that premium diamonds actually have a smaller proportion than Very Good diamonds do. I would be wary of the fair and good numbers since the overall sample sizes included in the diamonds dataset for these two cuts are much smaller than the other cuts.

```
pricy_small_diamonds <- diamonds %>%
  group_by(cut) %>%
  summarise(pricy_and_small = sum(price > 10000 & carat <1.5),
            total = n()) %>%
  mutate(proportion = pricy_and_small / total)
pricy_small_diamonds
```

```
## # A tibble: 5 x 4
##   cut       pricy_and_small total proportion
##   <ord>               <int> <int>      <dbl>
## 1 Fair                    4  1610    0.00248
## 2 Good                   17  4906    0.00347
## 3 Very Good             155 12082    0.0128
## 4 Premium               173 13791    0.0125
## 5 Ideal                 485 21551    0.0225
```

```
ggplot(pricy_small_diamonds, mapping = aes(x= cut, y = proportion)) +
  geom_point() +
  labs(title = "Proportion of Small but Expensive Diamonds \nto overall diamonds in dataset")
```



EDA - Exploratory Data Analysis

Texas Housing Sales Dataset Time Period

The data extends from January 2000 to July 2015

```
texas_housing <- txhousing %>%
arrange(date)
head(texas_housing, 1)
```

```
## # A tibble: 1 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
```

```
tail(texas_housing, 1)
```

```
## # A tibble: 1 x 9
##   city           year month sales   volume median listings inventory  date
##   <chr>         <int> <int> <dbl>    <dbl>  <dbl>    <dbl>     <dbl> <dbl>
## 1 Wichita Falls  2015     7   172 23850905 116700      811      6.50  2016
```

Number of Texas Cities in Dataset

There are 46 cities represented in the data set.

```r
texas_housing %>%
summarise(n_distinct(city))
```

```
## # A tibble: 1 x 1
##   `n_distinct(city)`
##                <int>
## 1                 46
```

Highest Texas Housing Sales

The highest sales, 8945, in the dataset occured in Houston in July 2015. This was the last month of sales data in the dataset.

```r
texas_housing %>%
  arrange(desc(sales)) %>%
  head(1)
```

```
## # A tibble: 1 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
```

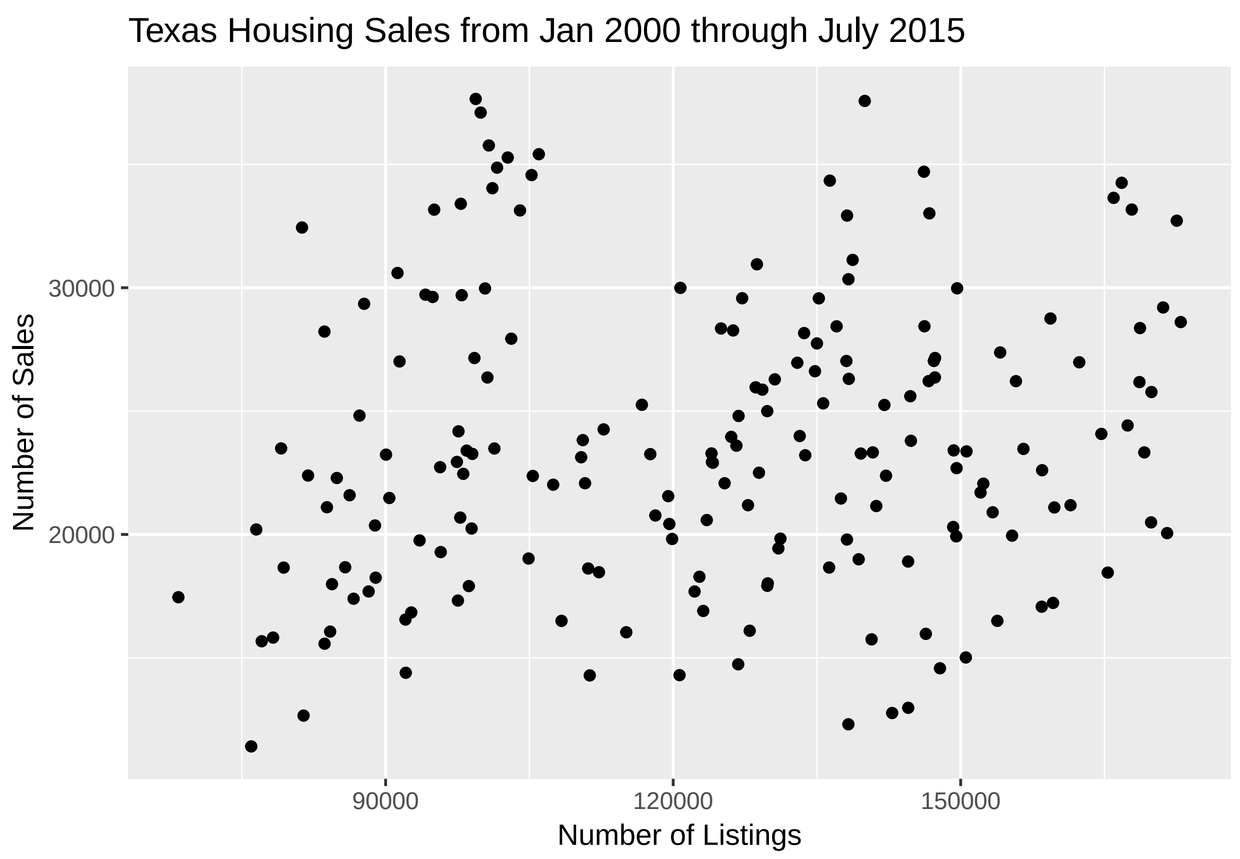
Texas Housing Sales Efficiency

I would expect the number of sales to go up as the number of listings go up. However, judging by the sales_efficiency metric defined below, as well as the scatterplot, that is not the case. There is no visible trend in the plot between the number of listings and the number of sales.

```r
texas_housing %>%
  group_by(date) %>%
  summarise(sales = sum(sales, na.rm = TRUE),
            listings = sum(listings, na.rm = TRUE)) %>%
  mutate(sales_efficiency = sales / listings) -> texas_housing_sales
texas_housing_sales
```

```
## # A tibble: 187 x 4
##     date sales listings sales_efficiency
##    <dbl> <dbl>    <dbl>            <dbl>
##  1  2000 11411    75978            0.150
##  2  2000 15674    77071            0.203
##  3  2000 20202    76505            0.264
##  4  2000 18658    79361            0.235
##  5  2000 22388    81906            0.273
##  6  2000 23488    79098            0.297
##  7  2000 21104    83877            0.252
##  8  2001 22289    84907            0.263
##  9  2001 17987    84406            0.213
## 10  2001 17396    86657            0.201
## # ... with 177 more rows
```

```r
  ggplot(texas_housing_sales, mapping = aes(x = listings, y = sales)) +
  geom_point() +
  labs(title = "Texas Housing Sales from Jan 2000 through July 2015", x = "Number of Listings", y = "Nu
```



Missing Sales

There are 568 rows of missing data for sales within the dataset. The “testing” tibble below lists the sum of missing sales entries for each city (sorted in descending order by number of missing entries) as well as the proportion of missing to overall sales entries by city.

```
texas_housing %>%
  group_by(city) %>%
  summarise(missing = sum(is.na(sales)),
            total = n()) %>%
  mutate(proportion = missing / total) -> missing_sales
missing_sales %>%
arrange(desc(missing))
```

```
## # A tibble: 46 x 4
##    city                missing total proportion
##    <chr>                 <int> <int>      <dbl>
##  1 South Padre Island      116   187     0.620
##  2 Kerrville               104   187     0.556
##  3 Midland                  75   187     0.401
##  4 Odessa                   72   187     0.385
##  5 San Marcos               46   187     0.246
##  6 Laredo                   36   187     0.193
##  7 Harlingen                25   187     0.134
##  8 Waco                     19   187     0.102
##  9 Texarkana                17   187     0.0909
## 10 Brazoria County         14   187     0.0749
## # ... with 36 more rows
```

```
summarise(missing_sales, sum(missing))
```

```
## # A tibble: 1 x 1
##   `sum(missing)`
##            <int>
## 1            568
```

Median Home Prices in Months w/ Sales > 500

The median house prices are different based on city. As the plot displays below some cities have drastically large ranges, while others have small ranges. Corpus Christi for instance, only has 5 months of data within the filtered dataset. Because of this, it makes more sense that range of data is the smallest for this city. Fort Bend, and Collin County have the largest ranges within their respective median home sale price data. I am most surprised that Collin County has the highest median house sale price among the dataset because i would have though higher density cities such as Houston or Austin would be higher.

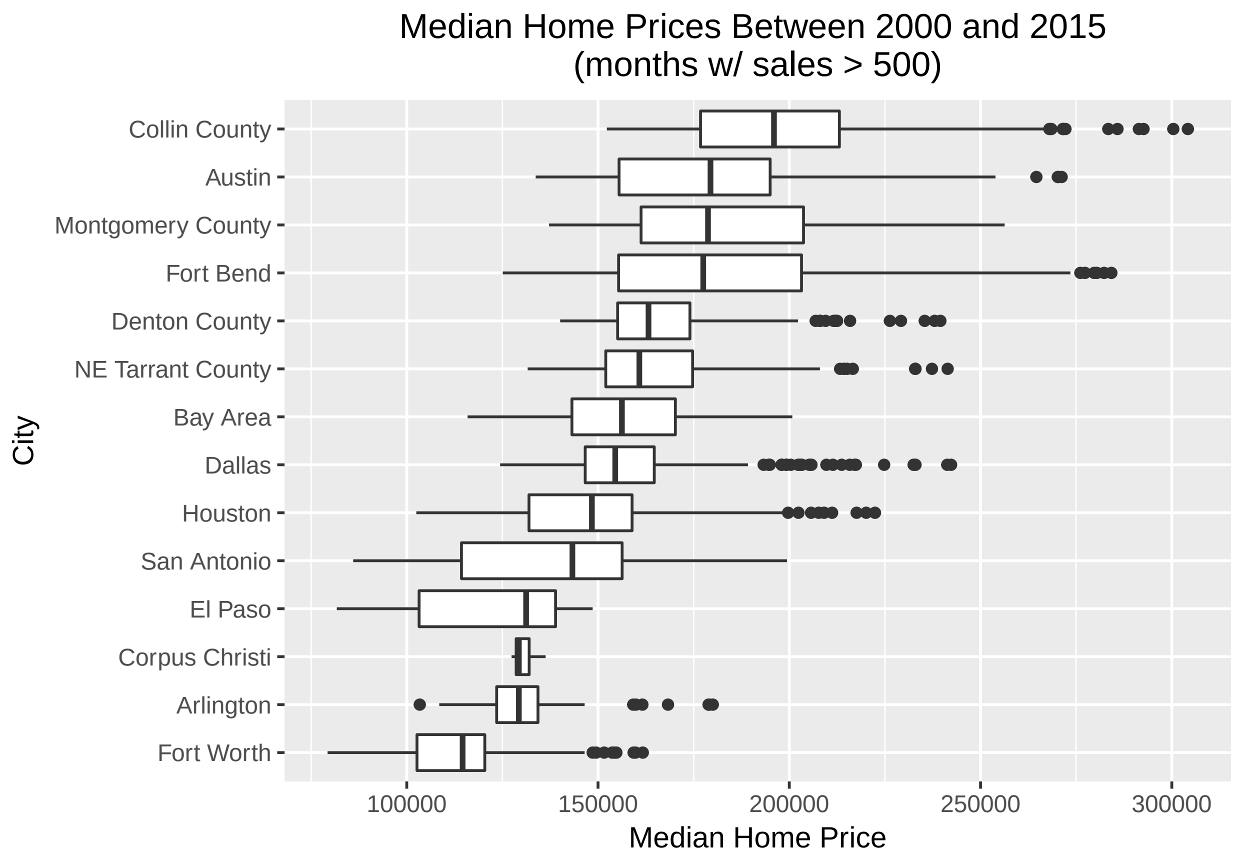
The cities that i would want to look into further are Collin County and Fort Worth because they have the highest and lowest median house prices in the filtered data. I'd also want to look into Corpus Christi because there are only 5 months in which there were more than 500 sales.

I would filter out cities / months with less than 500 sales because what we are trying to gauge is the average housing price for the city over a ~15 year period of data. Although median is a fairly robust metric for looking at housing prices, if there are only a few housing sales in any given city / month, there may be specific events that would skew the data. However, with more than 500 sales, it is safe to assume that the sample size is large enough to provide a fair representation of the housing market for that specific city / month.

```
texas_housing2 <- texas_housing
names(texas_housing2)[6]<-"med"
texas_housing2 %>%
  filter(sales > 500) %>%
  group_by(city) -> tex_hou_fil
summarise(tex_hou_fil, median = median(med), max = max(med), min = min(med), range = (max(med) - min(me
  arrange(desc(range))
```

```
## # A tibble: 14 x 6
##    city              median    max    min  range count
##    <chr>              <dbl>  <dbl>  <dbl>  <dbl> <int>
##  1 Fort Bend         177500 284200 125100 159100   176
##  2 Collin County     196000 304200 152300 151900   185
##  3 Austin            179400 271200 133700 137500   187
##  4 Houston           148400 222400 102500 119900   187
##  5 Montgomery County 178750 256300 137200 119100   108
##  6 Dallas            154500 242300 124400 117900   187
##  7 San Antonio       143300 199400  86000 113400   187
##  8 NE Tarrant County 160800 241400 131600 109800   158
##  9 Denton County     163200 239500 140100  99400   134
## 10 Bay Area          156250 200800 115900  84900    84
## 11 Fort Worth        114600 161700  79300  82400   172
## 12 Arlington         129300 180000 103400  76600    41
## 13 El Paso           131200 148600  81700  66900    72
## 14 Corpus Christi    129300 136300 127400   8900     5
```

```
ggplot(tex_hou_fil, mapping = aes(x = reorder(city, med, median), y = med)) +
geom_boxplot() +
coord_flip() +
labs(title = "Median Home Prices Between 2000 and 2015 \n(months w/ sales > 500)", x = "City", y = "Me
theme(plot.title = element_text(hjust = 0.5))
```


Median Home Prices Between 2000 and 2015
(months w/ sales > 500)
City
Collin County
Austin
Montgomery County
Fort Bend
Denton County
NE Tarrant County
Bay Area
Dallas
Houston
San Antonio
El Paso
Corpus Christi
Arlington
Fort Worth
Median Home Price