DATA608_module1_Exploratory Data Analysis

David Simbandumwe

```
knitr::opts_chunk$set(warning = FALSE, message = FALSE)
rm(list=ls())
```

Principles of Data Visualization and Introduction to ggplot2

I have provided you with data about the 5,000 fastest growing companies in the US, as compiled by Inc. magazine. lets read this in:

```
inc <- read.csv("https://raw.githubusercontent.com/charleyferrari/CUNY_DATA_608/maste
```

And lets preview this data:

```
head(inc)
```

```
##   Rank                         Name Growth_Rate   Revenue
## 1    1                         Fuhu      421.48 1.179e+08
## 2    2        FederalConference.com      248.31 4.960e+07
## 3    3                The HCI Group      245.45 2.550e+07
## 4    4                      Bridger      233.08 1.900e+09
## 5    5                       DataXu      213.37 8.700e+07
## 6    6 MileStone Community Builders      179.38 4.570e+07
##                       Industry Employees         City State
## 1 Consumer Products & Services       104   El Segundo    CA
## 2         Government Services         51     Dumfries    VA
## 3                       Health       132 Jacksonville    FL
## 4                       Energy        50      Addison    TX
## 5      Advertising & Marketing       220       Boston    MA
## 6                  Real Estate        63       Austin    TX
```

```
summary(inc)
```

```
##       Rank          Name            Growth_Rate         Revenue
##  Min.   :   1   Length:5001        Min.   :  0.340   Min.   :2.000e+06
##  1st Qu.:1252   Class :character   1st Qu.:  0.770   1st Qu.:5.100e+06
##  Median :2502   Mode  :character   Median :  1.420   Median :1.090e+07
##  Mean   :2502                      Mean   :  4.612   Mean   :4.822e+07
##  3rd Qu.:3751                      3rd Qu.:  3.290   3rd Qu.:2.860e+07
##  Max.   :5000                      Max.   :421.480   Max.   :1.010e+10
##
##    Industry           Employees           City              State
##  Length:5001        Min.   :    1.0   Length:5001        Length:5001
##  Class :character   1st Qu.:   25.0   Class :character   Class :character
##  Mode  :character   Median :   53.0   Mode  :character   Mode  :character
##                     Mean   :  232.7
##                     3rd Qu.:  132.0
##                     Max.   :66803.0
##                     NA's   :12
```

Think a bit on what these summaries mean. Use the space below to add some more relevant non-visual exploratory information you think helps you understand this data:

The skim() function outlines the structure of the dataset and key stats for the numeric and char variables. This is an interesting function because it deals with numeric and categorical data automatically.

```
library(skimr)

skim(inc)
```

Data summary

Name	inc
Number of rows	5001
Number of columns	8
_______________________	
Column type frequency:	
character	4
numeric	4
________________________	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
Name	0	1	2	51	0	5001	0
Industry	0	1	5	28	0	25	0
City	0	1	4	22	0	1519	0
State	0	1	2	2	0	52	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Rank	0	1	2501.64	1443.51	1.0e+00	1.252e+03	2.502e+03	3.751e+03	5.0000e+03	▇▇▇▇▇
Growth_Rate	0	1	4.61	14.12	3.4e-01	7.700e-01	1.420e+00	3.290e+00	4.2148e+02	▇▁▁▁▁
Revenue	0	1	48222535.49	240542281.14	2.0e+06	5.100e+06	1.090e+07	2.860e+07	1.0100e+10	▇▁▁▁▁
Employees	12	1	232.72	1353.13	1.0e+00	2.500e+01	5.300e+01	1.320e+02	6.6803e+04	▇▁▁▁▁

The stat.desc() function computes additional descriptive statistics about the series in a data frame. The function provides metrics on the number of na / null records and includes statistical measures such as std.dev, var etc.

```
library(pastecs)
library(dplyr)

stat_df <- stat.desc(Filter(is.numeric, inc))
stat_df <- format(stat_df, scientific = F, digits = 2, drop0trailing = TRUE)
stat_df
```

```
##                    Rank Growth_Rate             Revenue Employees
## nbr.val            5001        5001                5001      4989
## nbr.null              0           0                   0         0
## nbr.na                0           0                   0        12
## min                   1        0.34             2000000         1
## max                5000      421.48         10100000000     66803
## range              4999      421.14         10098000000     66802
## sum            12510706    23063.74        241160900000   1161030
## median             2502        1.42            10900000        53
## mean            2501.64        4.61            48222535     232.7
## SE.mean           20.41         0.2             3401441      19.2
## CI.mean.0.95      40.02        0.39             6668317      37.6
## var          2083710.06      199.48 57860589014049984 1830955.2
## std.dev         1443.51       14.12           240542281    1353.1
## coef.var           0.58        3.06                   5       5.8
```

The ad.test() function tests for normal distribution for each variable in the dataset. With the resulting p-value of less than 0.05 we reject the null hypothesis for all variables. The variables are not normally distributed at a confidence level 0.95.

```
library(nortest)
ad_t <- ad.test(inc$Growth_Rate)
print(paste0(ad_t$method , ' variable ' , ad_t$data.name, ' normal distributed ', ad_
```

```
## [1] "Anderson-Darling normality test variable inc$Growth_Rate normal distributed F
```

```
ad_t <- ad.test(inc$Revenue)
print(paste0(ad_t$method , ' variable ' , ad_t$data.name, ' normal distributed ', ad_
```

```
## [1] "Anderson-Darling normality test variable inc$Revenue normal distributed FALSE
```

```
ad_t <- ad.test(inc$Employees)
print(paste0(ad_t$method , ' variable ' , ad_t$data.name, ' normal distributed ', ad_
```

```
## [1] "Anderson-Darling normality test variable inc$Employees normal distributed FAL
```

The rcorr() function creates a correlation matrix for numeric variables. It highlights the relationship between variable pairs in the dataset

```
library(Hmisc)

inc_num <- inc %>% dplyr::select(where(is.numeric))
rcorr(as.matrix(inc_num))
```

```
##              Rank Growth_Rate Revenue Employees
## Rank         1.00       -0.40    0.08      0.05
## Growth_Rate -0.40        1.00    0.01     -0.02
## Revenue      0.08        0.01    1.00      0.28
## Employees    0.05       -0.02    0.28      1.00
##
## n
##             Rank Growth_Rate Revenue Employees
## Rank        5001        5001    5001      4989
## Growth_Rate 5001        5001    5001      4989
## Revenue     5001        5001    5001      4989
## Employees   4989        4989    4989      4989
##
## P
##             Rank   Growth_Rate Revenue Employees
## Rank               0.0000      0.0000  0.0001
## Growth_Rate 0.0000             0.6558  0.2070
## Revenue     0.0000 0.6558              0.0000
## Employees   0.0001 0.2070      0.0000
```

And finaly the average for variables for Revenue and Employees was computed using summarise in the dplyr package.

```
inc %>% tidyr::drop_na(Revenue, Employees) %>% group_by(State) %>%
    select(City, State, Revenue, Employees) %>%
    summarise(avg_rev=mean(Revenue),avg_emp=mean(Employees)) %>%
    arrange(desc(avg_rev))
```

```
## # A tibble: 52 × 3
##    State    avg_rev avg_emp
##    <chr>      <dbl>   <dbl>
##  1 ID    231523529.    342.
##  2 AK    171500000    1264
##  3 IA    123142857.    405.
##  4 IL    122201471.    380.
##  5 HI     99485714.     88.7
##  6 WI     92615584.    202.
##  7 DC     78019048.    220.
##  8 OH     68745161.    204.
##  9 NC     68537037.    272.
## 10 MI     61950794.    293.
## # … with 42 more rows
```

```
inc %>% group_by(City) %>%
    select(City, State, Revenue, Employees) %>%
    summarise(avg_rev=mean(Revenue),avg_emp=mean(Employees)) %>%
    arrange(desc(avg_rev))
```

```
## # A tibble: 1,519 × 3
##    City                avg_rev avg_emp
##    <chr>                 <dbl>   <dbl>
##  1 Vernon Hills    5053050000    3407
##  2 Beloit          4700000000    6549
##  3 Mt. Sterling    4500000000    3919
##  4 West Des Moines 2800000000    4589
##  5 Tarrytown       1902300000    1572
##  6 Ponte Vedra     1400000000    5347
##  7 Twinsburg       1352650000     330
##  8 Corte Madera    1200000000    2900
##  9 Huntersville    1172266667.   1146.
## 10 Flint           1100000000     761
## # … with 1,509 more rows
```

// Question 1

Create a graph that shows the distribution of companies in the dataset by State (ie how many are in each state). There are a lot of States, so consider which axis you should use. This visualization is ultimately going to be consumed on a 'portrait' oriented screen (ie taller than wide), which should further guide your layout choices.

```
inc %>% group_by(State) %>% summarise(num = n()) %>%
    ggplot(aes(x=reorder(State,num), y=num, fill=State)) +
    coord_flip() +
    theme_light() +
    geom_bar(stat = 'identity', width=0.3 , show.legend = FALSE) +
    labs(
        x = 'State',
        y = 'Number of Companies',
        title = 'Distribution of Companies (by State)'
    )
```

// Quesiton 2

Lets dig in on the state with the 3rd most companies in the data set. Imagine you work for the state and are interested in how many people are employed by companies in different industries. Create a plot that shows the average and/or median employment by industry for companies in this state (only use cases with full data, use R's `complete.cases()` function.) In addition to this, your graph should show how variable the ranges are, and you should deal with outliers.

The state with the 3rd most companies is NY. The boxplot depicts the distribution of employees from each industry. The variable ranges for the data and captures the median and mean value of the number of Employees.

```
st3 <- inc %>% group_by(State) %>% summarise(num = n()) %>%
              arrange(desc(num))

state_st <- st3$State[3]
print(paste0('State: ', state_st))
```

```
## [1] "State: NY"
```

```
print(paste0('Original Record Count: ', st3$num[3]))
```

```
## [1] "Original Record Count: 311"
```

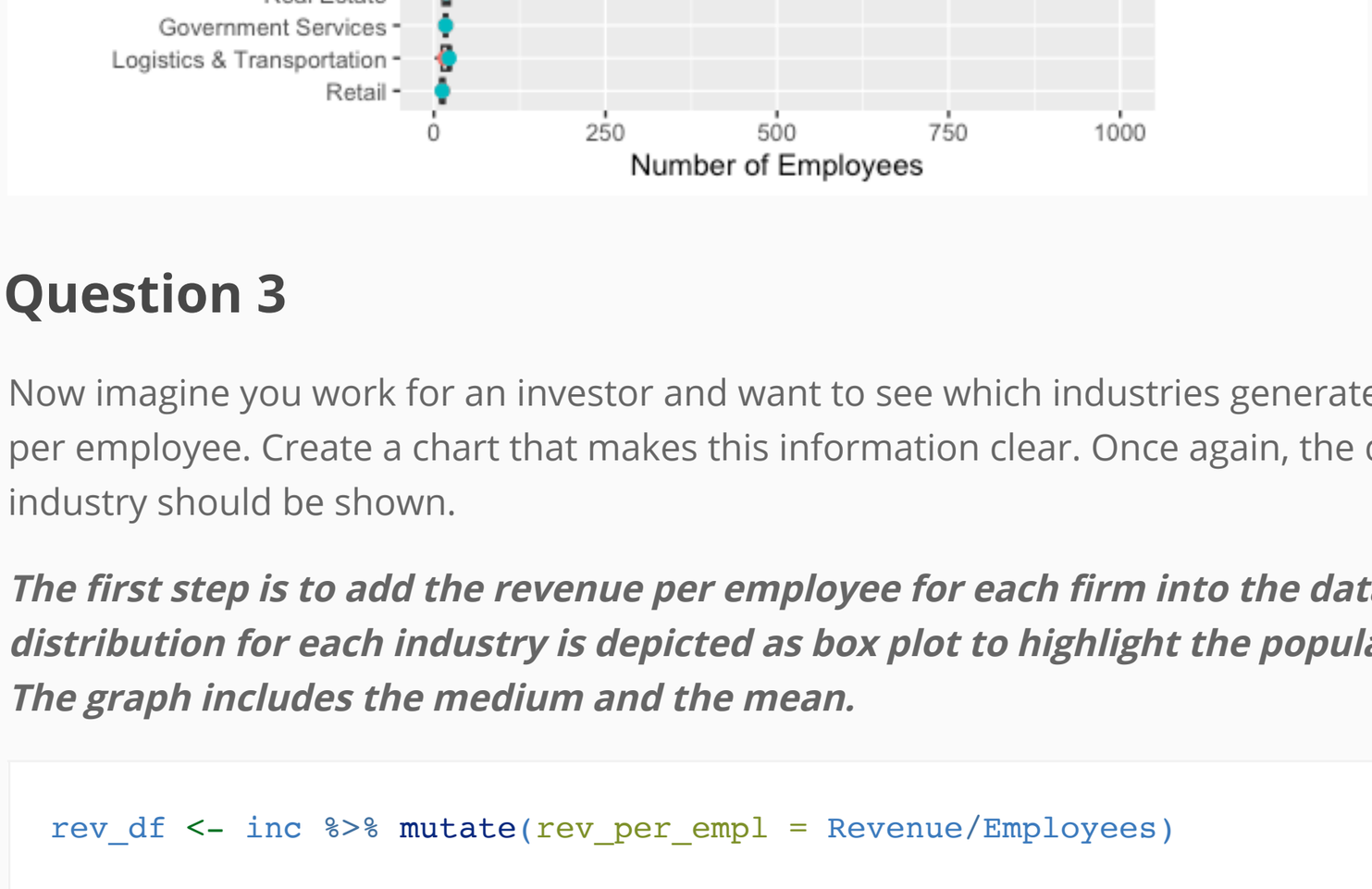
```
ny_df <- inc  %>% filter(State == state_st, complete.cases(.)) %>% arrange(Industry) '
print(paste0('Filter Complete Record: ', nrow(ny_df)))
```

```
## [1] "Filter Complete Record: 311"
```

```
ny_df <- ny_df %>% group_by(Industry) %>% filter(!(abs(Employees - median(Employees))
print(paste0('Filter Outliers: ', nrow(ny_df)))
```

```
## [1] "Filter Outliers: 262"
```

```
ny_df %>% ggplot(aes(x=reorder(Industry,Employees), y=Employees)) +
    coord_flip() +
    geom_boxplot(show.legend = FALSE, outlier.colour = NA) +
    stat_summary(fun="mean", size=2, geom = "point", aes(color="Mean"))+
    stat_summary(fun="median", size=2, geom = "point", aes(color="Median"))+
    labs(
        x = 'Industry',
        y = 'Number of Employees',
        title = 'NY Distribution of Employees (by Industry)'
    )
```



// Question 3

Now imagine you work for an investor and want to see which industries generate the most revenue per employee. Create a chart that makes this information clear. Once again, the distribution per industry should be shown.

The first step is to add the revenue per employee for each firm into the dataset. The distribution for each industry is depicted as box plot to highlight the population distribution. The graph includes the medium and the mean.

```
rev_df <- inc %>% mutate(rev_per_empl = Revenue/Employees)


print(paste0('Original Record Count: ', nrow(rev_df)))
```

```
## [1] "Original Record Count: 5001"
```

```
rev_df <- rev_df  %>% filter(complete.cases(.)) %>% arrange(Industry) %>% select(Name
print(paste0('Filter Complete Record: ', nrow(rev_df)))
```

```
## [1] "Filter Complete Record: 4989"
```

```
rev_df <- rev_df %>% group_by(Industry) %>% filter(!(abs(rev_per_empl - median(rev_pe
print(paste0('Filter Outliers: ', nrow(rev_df)))
```

```
## [1] "Filter Outliers: 4312"
```

```
rev_df %>% ggplot(aes(x=reorder(Industry,rev_per_empl), y=rev_per_empl)) +
    coord_flip() +
    geom_boxplot(show.legend = FALSE, outlier.colour = NA) +
    stat_summary(fun="mean", size=2, geom = "point", aes(color="Mean"))+
    stat_summary(fun="median", size=2, geom = "point", aes(color="Median"))+
    scale_y_continuous(labels = scales::comma) +
    labs(
        x = 'Industry',
        y = 'Revenue per Employees',
        title = 'Distribution of Revenue Per Employee (by Industry)'
    )
```