STA130H1F

Class #3

Prof. Nathan Taback

2018-24-09

Intact internships

bit.ly / intact HQ

Welcome back to STA130 😄

Today's class

- Statistical data

Welcome back to STA130 😄

Today's class

- Statistical data

- Tidy data

Welcome back to STA130 😄

Today's class

- Statistical data

- Tidy data

- Data wrangling

Welcome back to STA130 😄

Today's class

- Statistical data

- Tidy data

- Data wrangling

- Boxplots

Statistical data

What is statistical data?

- Statistical data is obtained by observing (random) variables.

- A random variable can be given a precise mathematical definition that we will cover later in the course.

- In this class we will discuss examples.

Observing a few variables on STA130 students

- What is your height?
- How many years have been at UofT?
- What is your sex (male or female)?

Collecting this data will generate three variables: `height`, `years`, and `eye_colour`.

Enter variables on STA130 students

```
height <- c()
years <- c()
eye_colour <- c()
```

Put the variables into an R data frame.

NB: `data_frame` is the `tidyverse` version of base R `data.frame`.

```
sta130_dat <- data_frame(height, years, eye_colour)
```

R   into data frame

We could have entred this in a spreadsheet program like MS Excel, saved it as a CSV file, then imported the file into R.

Tidy data

There are three interrelated rules which make a dataset tidy:

1. Each variable must have its own column.

2. Each observation must have its own row.

3. Each value must have its own cell.

Variable : height, years, eye-Colour.
(measurements)

Observation : a person   6 people
                                  n=6

Cell :

The variable for a particular observation.

obs	height	years	eye-Colour
1	///////		
2			
3			
4			
5			
6			

Suppose that a first year class of 250 students has the following distribution of eye colour.

Colour	N
Blue	105
Hazel	55
Green	75
Other	15

We can create a tidy data set with a categorical variable `eye_col`.

Suppose that a first year class of 250 students has the following distribution of eye colour.

Colour	N
Blue	105
Hazel	55
Green	75
Other	15

We can create a tidy data set with a categorical variable eye_col.

[handwritten: Create a variable where each value is blue. n=105]

```r
library(tidyverse)
blue_eye <- rep("Blue", 105)
hazel_eye <- rep("Hazel", 55)
green_eye <- rep("Green", 75)
other_eye <- rep("Other", 15)
eye_col = c(blue_eye, hazel_eye,
            green_eye, other_eye)
eye_data <- data_frame(stnum = 1:250, eye_col)
glimpse(eye_data)
```

[handwritten: Variable]

[handwritten: Concatenated all the eye colours for Students.]

[handwritten: 1,2,...,250]

```
## Observations: 250
## Variables: 2
## $ stnum   <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,...
## $ eye_col <chr> "Blue", "Blue", "Blue", "Blue", "Blue", "Blue", "Blue"...
```

[handwritten: = # of rows]

[handwritten: Variable]

Tidy data

Which data set is 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
```

```
## # A tibble: 6 x 3
##    country     year rate
## * <chr>      <int> <chr>
## 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
```

$f(x_iy) = z$.

three variables: year, cases, population

(A) observation: Country in a particular year.

$f(x_iy) = Prop = \dfrac{Cases}{population}$

(B) 2+3

Not tidy

Tidy data

> "For a given dataset, it is usually easy to figure out what are observations and what are variables, but it is surprisingly difficult to precisely define variables and observations in general." (Wickham, 2014)

A general rule of thumb:

- It is easier to describe functional relationships between variables (e.g., z is a linear combination of x and y, density is the ratio of weight to volume) than between rows.

- It is easier to make comparisons between groups of observations (e.g., average of group a vs. average of group b) than between groups of columns.

(Wickham, 2014)

Data Wrangling

Data wrangling

- The `ggplot` library implements a **grammer of graphics**.

- Similarily the `dplyr` library presents a grammer for data wrangling.

The Economic Guide to Picking a Major

FiveThirtyEight

Politics   Sports   Science & Health   **Economics**   Culture

SEP. 12, 2014 AT 7:37 AM

The Economic Guide To Picking A College Major

By **Ben Casselman**
Filed under **Higher Education**
Get the data on **GitHub**

Students walk across the campus of UCLA in Los Angeles. KEVORK DJANSEZIAN / GETTY IMAGES

> "...A college degree is no guarantee of economic success. But through their choice of major, they can take at least some steps toward boosting their odds."

The Economic Guide to Picking a Major

- The data used in the article is from the American Community Survey 2010-2012 Public Use Microdata Series.

- We can use the `fivethirtyeight` library in R.

Data behind the article

```
library(fivethirtyeight) # load the library
glimpse(college_recent_grads)
```

name of the data set. (handwritten annotation)

```
## Observations: 173
## Variables: 21
## $ rank                         <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11,...
## $ major_code                   <int> 2419, 2416, 2415, 2417, 2405, 2418...
## $ major                        <chr> "Petroleum Engineering", "Mining A...
## $ major_category               <chr> "Engineering", "Engineering", "Eng...
## $ total                        <int> 2339, 756, 856, 1258, 32260, 2573,...
## $ sample_size                  <int> 36, 7, 3, 16, 289, 17, 51, 10, 102...
## $ men                          <int> 2057, 679, 725, 1123, 21239, 2200,...
## $ women                        <int> 282, 77, 131, 135, 11021, 373, 166...
## $ sharewomen                   <dbl> 0.1205643, 0.1018519, 0.1530374, 0...
## $ employed                     <int> 1976, 640, 648, 758, 25694, 1857, ...
## $ employed_fulltime            <int> 1849, 556, 558, 1069, 23170, 2038,...
## $ employed_parttime            <int> 270, 170, 133, 150, 5180, 264, 296...
## $ employed_fulltime_yearround  <int> 1207, 388, 340, 692, 16697, 1449, ...
## $ unemployed                   <int> 37, 85, 16, 40, 1672, 400, 308, 33...
## $ unemployment_rate            <dbl> 0.018380527, 0.117241379, 0.024096...
## $ p25th                        <dbl> 95000, 55000, 50000, 43000, 50000,...
## $ median                       <dbl> 110000, 75000, 73000, 70000, 65000...
## $ p75th                        <dbl> 125000, 90000, 105000, 80000, 7500...
## $ college_jobs                 <int> 1534, 350, 456, 529, 18314, 1142, ...
```

Select variables/columns using select()

To retrieve a data frame with only major, number of male and female graduates we use the `select()` function in the `dplyr` library.

name of data frame

```
select(college_recent_grads,major, men,women)
```

list of variables that want to select from data frame.

```
## # A tibble: 173 x 3
##    major                                    men women
##    <chr>                                  <int> <int>
##  1 Petroleum Engineering                   2057   282
##  2 Mining And Mineral Engineering           679    77
##  3 Metallurgical Engineering                725   131
##  4 Naval Architecture And Marine Engineering 1123  135
##  5 Chemical Engineering                   21239 11021
##  6 Nuclear Engineering                     2200   373
##  7 Actuarial Science                       2110  1667
##  8 Astronomy And Astrophysics               832   960
##  9 Mechanical Engineering                 80320 10907
## 10 Electrical Engineering                 65511 16016
## # ... with 163 more rows
```

Select observations/rows using filter()

[handwritten: NB: In R = means assignment x = c(1,2) x <- c(1,2)]

[handwritten: Stuck with this syntax]

If we want to retrieve only those observations (rows) that pertain to engineering majors then we need to specify that the value of the `major` variable is Electrical Engineering.

[handwritten: data frame.]

```r
# == is a test for equality and is different than =.
EE <- filter(college_recent_grads,
             major == "Electrical Engineering")
glimpse(EE)
```

```
## Observations: 1
## Variables: 21
## $ rank              <int> 10
## $ major_code        <int> 2408
## $ major             <chr> "Electrical Engineering"
## $ major_category    <chr> "Engineering"
## $ total             <int> 81527
## $ sample_size       <int> 631
## $ men               <int> 65511
## $ women             <int> 16016
## $ sharewomen        <dbl> 0.1964503
## $ employed          <int> 61928
## $ employed_fulltime <int> 55450
```

Combine `select()` and `filter()`

- We can drill down to get certain pieces of information using `filter()` and `select()` together.

- The `median` variable is median salary.

```
select(filter(college_recent_grads, median <= 25000 ),
       major, men, women)
```

(1) Which students, and (2) variables are in this data frame?

🖥 Respond at **PollEv.com/nathantaback**
📱 Text **NATHANTABACK** to **37607** once to join, then **A, B, C, or D**

(1) 50% of the students in the original data set that earn 25,000; (2) three variables: major, men, women	**A**
(1) All students in the original data set in a major where the median salary is at least 25,000; (2) all variables in the data set.	**B**
(1) 50% of the students in the original data set that earn at most 25,000; (2) three variables: major, men, women	**C**
(1) All students in the original data set in a major where the median salary is at most 25,000; (2) three variables: major, men, women	**D**

The pipe operator %>% *(circled)*

In the code:

```
select(filter(college_recent_grads, median >= 60000),
       major,men,women)
```

filter is nested inside select.

The pipe operator allows is an alternative to nesting and yields easier to read code.

The same expression can be written with the pipe operator

```
college_recent_grads %>%
  filter(median >= 60000) %>%
  select(major, men, women)
```

— Piped into

— piped into

Create new variables from existing variables using `mutate()`

What percentage of graduates from each major where the median earnings is at least $60,000 are men ?

[handwritten: original data]
[handwritten: only select rows median ≥ 60000]

```
college_recent_grads %>%
  filter(median >= 60000) %>%
  select(major, men, women) %>%
  mutate(total = men + women,
         pct_male = round((men / total)*100, 2))
```

[handwritten: Select three columns.]
[handwritten: creates two new variables.]

Compare to nested code:

```
mutate(select(filter(college_recent_grads,median >= 60000),
              major, men, women),
       total = men + women,
       pct_male = round((men / total)*100, 2))
```

Create new variables from existing variables using mutate()

format table for my slides — ignore for now

```
knitr::kable(college_recent_grads %>%
    filter(median >= 60000) %>%
    select(major, men, women) %>%
    mutate(total = men + women,
           pct_male = round((men / total)*100, 2)),
    format = "html")
```

Selects three columns.

creates two new columns.

major	men	women	total	pct_male
Petroleum Engineering	2057	282	2339	87.94
Mining And Mineral Engineering	679	77	756	89.81
Metallurgical Engineering	725	131	856	84.70
Naval Architecture And Marine Engineering	1123	135	1258	89.27
Chemical Engineering	21239	11021	32260	65.84
Nuclear Engineering	2200	373	2573	85.50

1 2 3 4 5 6

$(\%>\%)$ pipe operator only works if tidyverse library is loaded.

Create new variables from existing variables using `mutate()` and `ifelse()`

- Suppose that we would like to create a categorical variable to identify majors with between 45% and 55% women (ie., approximately equal numbers of males and females).

Create new variables from existing variables using `mutate()` and `ifelse()`

- Suppose that we would like to create a categorical variable to identify majors with between 45% and 55% women (ie., approximately equal numbers of males and females).

- We can use `ifelse()` in a `mutate()` statement.

The format of an `ifelse()` statement in R is:

```
ifelse(test, yes, no)
```

Create new variables from existing variables using `mutate()` and `ifelse()`

- Suppose that we would like to create a categorical variable to identify majors with between 45% and 55% women (ie., approximately equal numbers of males and females).

- We can use `ifelse()` in a `mutate()` statement.

The format of an `ifelse()` statement in R is:

`ifelse(test, yes, no)`

```
people <- c("Jamie", "Lei", "Francois", "Fanny")
ifelse(people == "Lei" | people == "Fanny", "Female", "Male")
```

```
## [1] "Male"   "Female" "Male"   "Female"
```

(handwritten annotations:) return if the condition is true. return if cond is false. test statement. or

```
college_recent_grads %>%
  select(major, men, women) %>%
  mutate(total = men + women,
         pct_female = round((women / total)*100, 2),
         sex.equal = ifelse(pct_female >= 45 & pct_female <= 55,
                            "Yes","No")) %>%
  select(major,sex.equal)
```

```
## # A tibble: 173 x 2
##    major                                  sex.equal
##    <chr>                                  <chr>
##  1 Petroleum Engineering                  No
##  2 Mining And Mineral Engineering         No
##  3 Metallurgical Engineering              No
##  4 Naval Architecture And Marine Engineering No
##  5 Chemical Engineering                   No
##  6 Nuclear Engineering                    No
##  7 Actuarial Science                      No
##  8 Astronomy And Astrophysics             Yes
##  9 Mechanical Engineering                 No
## 10 Electrical Engineering                 No
## # ... with 163 more rows
```

Rename variables using `rename()`

- It's considered bad practice in R to use periods in variable names.

- We can use `rename()` to change the name of `sex.equal` to `sex_equal`.

[handwritten: Saves data into my_college_dat]

```r
my_college_dat <- college_recent_grads %>%
  select(major, men, women, median) %>%
  mutate(total = men + women,
         pct_female = round((women / total)*100, 2),
         sex.equal = ifelse(pct_female >= 45 &
                              pct_female <= 55, "Yes","No")) %>%
  select(major,sex.equal, median)

my_college_dat <- my_college_dat %>%
  rename(sex_equal = sex.equal, salary_median = median)
glimpse(my_college_dat)
```

```
## Observations: 173
## Variables: 3
## $ major         <chr> "Petroleum Engineering", "Mining And Mineral Eng...
## $ sex_equal     <chr> "No", "No", "No", "No", "No", "No", "No", "Yes",...
## $ salary_median <dbl> 110000, 75000, 73000, 70000, 65000, 65000, 62000...
```

Sort a data frame using arrange()

```
my_college_dat %>%
    select(major, salary_median) %>%
    arrange(desc(salary_median))
```

```
## # A tibble: 173 x 2
##    major                                    salary_median
##    <chr>                                             <dbl>
##  1 Petroleum Engineering                            110000
##  2 Mining And Mineral Engineering                    75000
##  3 Metallurgical Engineering                         73000
##  4 Naval Architecture And Marine Engineering         70000
##  5 Chemical Engineering                              65000
##  6 Nuclear Engineering                               65000
##  7 Actuarial Science                                 62000
##  8 Astronomy And Astrophysics                        62000
##  9 Mechanical Engineering                            60000
## 10 Electrical Engineering                            60000
## # ... with 163 more rows
```

Summarize a data frame using `summarize()`

The average number of female grads and the total number of majors in the data set.

```
college_recent_grads %>%
   select(major, men, women) %>%
   summarise(femgrad_mean = mean(women, na.rm = T), N = n())
```

```
## # A tibble: 1 x 2
##    femgrad_mean      N
##           <dbl> <int>
## 1       22647.    173
```
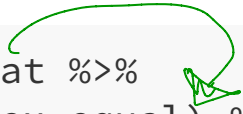
F

mean number of women.

number of rows in the dataframe

Summarize groups in a data frame using `summarize()` and `group_by()`

The median salary in majors with 45%-55% female students.

```
my_college_dat %>%
   group_by(sex_equal) %>%
   summarise(median(salary_median))
```

```
## # A tibble: 3 x 2
##   sex_equal `median(salary_median)`
##   <chr>                       <dbl>
## 1 No                          36000
## 2 Yes                         37400
## 3 <NA>                        53000
```
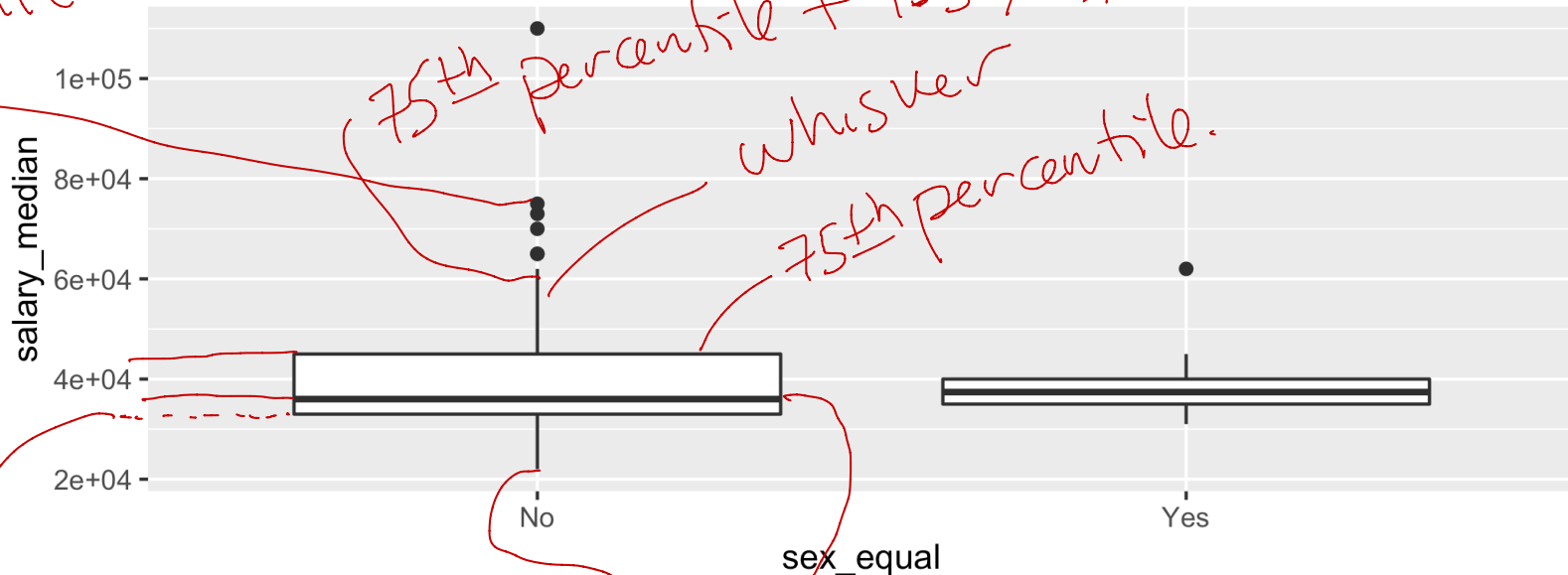
Boxplots to compare distribution of salary in males versus females

Box and Whiskers plot

data frame is piped into ggplot.

```
my_college_dat %>% filter(is.na(sex_equal) == FALSE) %>%
  ggplot(aes(x = sex_equal, y = salary_median)) + geom_boxplot()
```

outliers.

75th percentile + 1.5 x IQR

whisker

75th percentile.

25th percentile

50th percentile / Median

25th percentile — 1.5 x IQR.

Anatomy of a Boxplot

A boxplot summarizes the distribution of a quantitative variable using five statistics while plotting unusual observations (*outliers*).

Anatomy of a Boxplot

A boxplot summarizes the distribution of a quantitative variable using five statistics while plotting unusual observations (*outliers*).

The five statistics are:

- $Q_1 = 25^{th}$ percentile (first quartile) : the value that 25% of obs. fall below
- Median = 50th percentile
- $Q_3 = 75^{th}$ percentile (third quartile)
- lower whisker = $Q_1 - 1.5 \times IQR$
- upper whisker = $Q_3 + 1.5 \times IQR$

NB: $IQR = Q_3 - Q_1$ is called the inter-quartile range.

IQR is a measure of how spread out the distribution of the variable is.

Anatomy of a Boxplot

An **outlier** in is defined as any value of the quantitative variable that is either:

less than $Q_1 - 1.5 \times IQR$ or greater than $Q_3 + 1.5 \times IQR$.

Anatomy of a Boxplot

An **outlier** in is defined as any value of the quantitative variable that is either:

less than $Q_1 - 1.5 \times IQR$ or greater than $Q_3 + 1.5 \times IQR$.
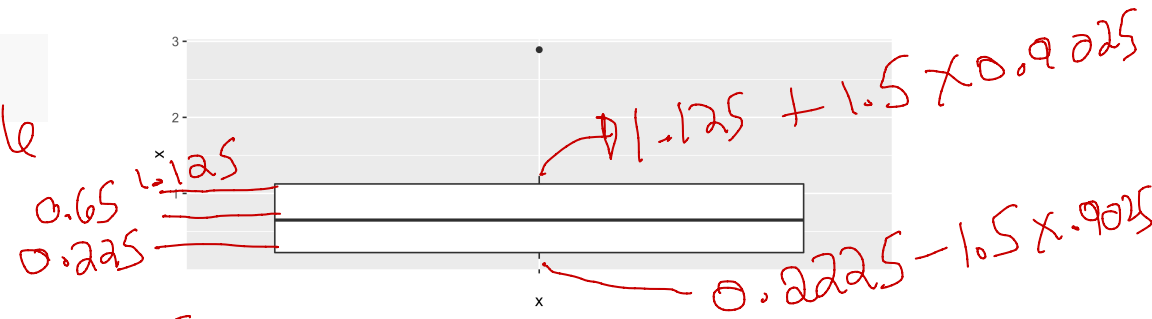
The whiskers of the boxplot capture data outside the box, but not more than $1.5 \times IQR$.

```
x
```

[1] 0.14 0.15 0.15 0.44 0.54 0.76

```
data_frame(x) %>%
  ggplot(aes(x = "", y = x)) +
  geom_boxplot()
```

data is cutoff didn't print.

```
quantile(x, 0.25)
```

25th percentile
Q1

25%
0.2225

0.65 1.125
0.225

$1.125 + 1.5 \times 0.9025$
$0.2225 - 1.5 \times .9025$

0.2225 is the value such that 25% of the observations in x fall below.

```
quantile(x, 0.50)
```

Median.

50%
0.65

and 75% of the values of x are above 0.2225

```
quantile(x, 0.75)
```

Q3 - 75th percentile.

75%
1.125

```
quantile(x, 0.75) - quantile(x,
```

quantile(x, 0.25)

75%
0.9025

IQR = 1.125 - 0.2225.
= 0.9025.

- Slides on joining (merging) data frames and the Trump tweets example have been removed. This topic will be covered on Oct. 1.

- The values of the variable x on the previous slide got cut off. A version of the previous slide with all the information is posted in HTML and PDF.