DSCI353-353m-453: Class 10a-p Inferential Stats Review, T-test

Profs: R. H. French, L. S. Bruckman, P. Leu, K. Davis, S. Cirlos

TAs: W. Oltjen, K. Hernandez, M. Li, M. Li, D. Colvin

01 November, 2022

Contents

10.1.2.1 Class Readings, Assignments, Syllabus Topics

10.1.2.1.1 Reading, Lab Exercises, SemProjects

- Readings:
  - For today: OIS 7.1-4
  - For next class:
- Laboratory Exercises:
  - LE 5: Is due Thursday November 10th
  - LE 6: Will be posted Thursday November 10th
- Office Hours: (Class Canvas Calendar for Zoom Link)
  - Wednesday @ 4:00 PM to 5:00 PM, Will Oltjen
  - Saturday @ 3:00 PM to 4:00 PM, Kristen Hernandez
  - **Office Hours are on Zoom, and recorded**

- Semester Projects
  - DSCI 451 Students Biweekly Update #5 Due Friday Nov. 4th
  - DSCI 451 Students
    - Next **Report Out #3 is Due November 25th**
  - All DSCI 351/351M/451 Students:
    - **Peer Grading of Report Out #2 is Due November 8th**
  - Exams
    - Final: Monday December 19, 2022, 12:00PM - 3:00PM, Nord 356 or remote

10.1.2.1.2 Textbooks

Introduction to R and Data Science

- For R, Coding, Inferential Statistics
  - Peng: R Programming for Data Science
  - Peng: Exploratory Data Analysis with R

Textbooks for this class

- OIS = Diez, Barr, Çetinkaya-Runde: Open Intro Stat v4
- R4DS = Wickham, Grolemund: R for Data Science

Textbooks for DSCI353/353M/453, And in your Repo now

- ISLR2 = James, Witten, Hastie, Tibshirani: Intro to Statistical Learning with R
- ESL = Trevor Hastie, Tibshirani, Friedman: Elements of Statistical Learning
- DLwR = Chollet, Allaire: Deep Learning with R

10.1.2.1.3 Tidyverse Cheatsheets, Functions and Reading Your Code

Look at the Tidyverse Cheatsheet

- **Tidyverse For Beginners Cheatsheet**
  - In the Git/20s-dsci353-353m-453-prof/3-readings/3-CheatSheets/ folder
- **Data Wrangling with dplyr and tidyr Cheatsheet**

Tidyverse Functions & Conventions

```
- The pipe operator `%>%`
- Use `dplyr::filter()` to subset data row-wise.
- Use `dplyr::arrange()`  to sort the observations in a data frame
- Use `dplyr::mutate()` to update or create new columns of a data frame
- Use `dplyr::summarize()` to turn many observations into a single data point
- Use `dplyr::arrange()` to change the ordering of the rows of a data frame
- Use `dplyr::select()` to choose variables from a tibble,
  - keeps only variables you mention
- Use `dplyr::rename()` keeps all the variables and renames variables
  - rename(iris, petal_length = Petal.Length)
- These can be combined using `dplyr::group_by()`
  - which lets you perform operations "by group".
- The `%in%` matches conditions provided by a vector using the c() function
- The **forcats** package has tidyverse functions
  - for factors (categorical variables)
- The **readr** package has tidyverse functions
  - to read_..., melt_... col_..., parse_... data and objects
```

Reading Your Code: Whenever you see

- The assignment operator `<-`, think **"gets"**
- The pipe operator, `%>%`, think **"then"**

Day:Date	Foundation	Practicum	Reading	Due
w01a:Tu:8/30/22	ODS Tool Chain	R, Rstudio, Git		
w01b:Th:9/1/22	Setup ODS Tool Chain	Bash, Git, Slack, Agile	PRP4-33	LE1
w02a:Tu:9/6/22	Bash-Git-Knuth-Lit.Prog.	RIntroR	PRP35-64	
w02b:Th:9/8/22	What is Data Science	OIS:Intro2R	OIS1,2	
w02Pr:Fr:9/9/22			PRP65-93	**451 Update1**
w03a:Tu:9/13/22	Data Intro	Data Analytic Style	PRP94-116	LE2 **LE1 Due**
w03b:Th:9/15/22	Rand. Var. Normal Dist.	Git, Rmds, Loops	OIS4	
w04a:Tu:9/20/22	Tidy Check Explore	Tidy GapMinder	EDA1-31	
w04b:Th:9/22/22	Inference, DSCI Process	Other Distrib. 7 ways	R4DS1-3	LE3 **LE2 Due**
w04Pr:Fr:9/23/22			EDA32-58	**451 Update2**
w05a:Tu:9/27/22	OIS4 Rand. Var.	EDA of PET Degr.	OIS5	
w05b:Th:9/29/22	OIS5 Found. of Infer.	Multivar Corr. Plot	R4DS4-6	
w05Pr:Fr:9/30/22				**451 RepOut1**
w06a:Tu:10/4/22	Pred., Algorithm, Model		R4DS7-8	
w06b:Th:10/6/22	Summ. Stats & Vis.	Anscombe's Quartets	R4DS9-16	LE4 **LE3 Due**
w06Pr:Fr:10/7/22				**451 Update3**
w07a:Tu:10/11/22	Midterm Rev. Tidy Data	Correl Plots Summ Stats	OIS6.1-2	**PeerRv1 Due**
w07b:Th:10/13/22	HypoTest, Infer. Recap	Penguin EDA, Sampling		
w08a:Tu:10/18/22	**MIDTERM**	**EXAM**		
w08b:Th:10/20/22	Programming & Coding	Code Packaging		**LE4 Due**
w08Pr:Fr:10/21/22				**451 Update4**
Tu:10/24,25	**CWRU**	**FALL BREAK**	R4DS17-21	
w09b:Th:10/27/22	Cat. Inf. 1 & 2 propor.	Indep. Test,2-way tables	OIS6.3-4	LE5
w09Pr:Fr:10/28/22				**451 RepOut2**
w10a:Tu:11/1/22	Goodness of Fit, χ^2 test	t-tests 1&2 means	OIS7.1-4	
w10b:Th:11/3/22	Num. Infer, Cont. Tables	Stat. Power		
w10Pr:Fr:11/4/22				**451 Update5**
w11a:Tu:11/8/22	Sample & Effect Size	Stat. Power GGmap	OIS8	**PeerRv2 Due**
w11b:Th:11/10/22	Inf. 4 Regr, Test & Train	Curse of Dimen.	ISLR1,2.1,2	LE6 **LE5 Due**
w12a:Tu:11/15/22	Lin. Regr. Part 1	Residuals	OIS9	
w12b:Th:11/17/22	Lin. Regr. Part 2	Regr. Diagnostics		
w12Pr:Fr:11/18/22				**451 Update6**
w13a:Tu:11/22/22	Mult. Lin. Regr.	Var. & Mod. Selec.,	ISLR3.1	LE7 **LE6 due**
w13b:Th:11/24/22	Log. Regr.	GIS Trends	ISLR3.2	
w13Pr:Fr:11/25/22				**451 RepOut3**
w14a:Tu:11/23/22	Classificat., Sup. Lrning	Caret, Broom 4 modeling	ISLR4.1-3	
Th,Fr:11/24,25	**THANKSGIVING**	**Vacation**		
w15a:Tu:11/29/22		Clustering		**PeerRv3 Due**
w15b:Th:12/1/22	Big Data Analytics	Dist. Comp., Hadoop		
w15SPr:Fr:12/2/22		Read Article by	Mirletz,2015	
w16a:Tu:12/6/22	Final Exam Review			
w15b:Th:12/8/22				**LE7 due**
Friday 12/12	**SemProj**	**Final Report**		**SemProj4 due**
Monday 12/19	**FINAL EXAM**	**12:00-3:00pm**	Nord 356	or remote

Figure 1: DSCI351-351M-451 Syllabus

10.1.2.1.4 Syllabus

10.1.3 Inferential Statistics Review

10.1.3.1 Descriptive Statistics

Here, we'll look at measures of

- central tendency,
- variability,
- and distribution shape
  - for continuous variables.

For illustrative purposes, we'll use several of the variables

- from the Motor Trend Car Road Tests (mtcars) dataset
- Our focus will be on
  - miles per gallon (mpg),
  - horsepower (hp),
  - and weight (wt):

First, we'll look at descriptive statistics for all 32 cars.

Then we'll examine descriptive statistics

- by transmission type (am)
- engine cylinder configuration (vs).

The former is coded

- 0 = automatic,
- 1 = manual,

and the later is coded

- 0 = V-shape and
- 1 = straight.

When it comes to calculating descriptive statistics,

- R has an embarrassment of riches.

Let's start with functions that are included in the base R.

Then we'll look at extensions that are available

- through the use of user-contributed packages.

In the base installation, you can use the summary() function

- to obtain descriptive statistics.

```
myvars <- c("mpg", "hp", "wt")
summary(mtcars[myvars])
```

10.1.3.1.1 Descriptive statistics with `summary()`

```
##       mpg              hp              wt
##  Min.   :10.40   Min.   : 52.0   Min.   :1.513
##  1st Qu.:15.43   1st Qu.: 96.5   1st Qu.:2.581
##  Median :19.20   Median :123.0   Median :3.325
##  Mean   :20.09   Mean   :146.7   Mean   :3.217
##  3rd Qu.:22.80   3rd Qu.:180.0   3rd Qu.:3.610
```

```
##  Max.   :33.90   Max.   :335.0   Max.   :5.424
```

10.1.3.1.2 Descriptive statistics via `sapply()`

I don't tend to use the "apply" functions that much anymore.

```
mystats <- function(x, na.omit = FALSE) {
  if (na.omit)
    x <- x[!is.na(x)]
  m <- mean(x)
  n <- length(x)
  s <- sd(x)
  skew <- sum((x - m) ^ 3 / s ^ 3) / n
  kurt <- sum((x - m) ^ 4 / s ^ 4) / n - 3
  return(c(
    n = n,
    mean = m,
    stdev = s,
    skew = skew,
    kurtosis = kurt
  ))
}

myvars <- c("mpg", "hp", "wt")
sapply(mtcars[myvars], mystats)
```

```
##                mpg          hp          wt
## n        32.000000  32.0000000 32.00000000
## mean     20.090625 146.6875000  3.21725000
## stdev     6.026948  68.5628685  0.97845744
## skew      0.610655   0.7260237  0.42314646
## kurtosis -0.372766  -0.1355511 -0.02271075
```

```
library(Hmisc)
```

10.1.3.1.3 Descriptive statistics via `describe()` in the `Hmisc` package

```
## Loading required package: lattice
```

```
## Loading required package: survival
```

```
## Loading required package: Formula
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
myvars <- c("mpg", "hp", "wt")
describe(mtcars[myvars])
```

```
## mtcars[myvars]
##
##  3  Variables      32  Observations
```

```
## ------------------------------------------------------------------------------
## mpg
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       32        0       25    0.999    20.09    6.796    12.00    14.34
##      .25      .50      .75      .90      .95
##    15.43    19.20    22.80    30.09    31.30
##
## lowest : 10.4 13.3 14.3 14.7 15.0, highest: 26.0 27.3 30.4 32.4 33.9
## ------------------------------------------------------------------------------
## hp
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       32        0       22    0.997    146.7    77.04    63.65    66.00
##      .25      .50      .75      .90      .95
##    96.50   123.00   180.00   243.50   253.55
##
## lowest :  52  62  65  66  91, highest: 215 230 245 264 335
## ------------------------------------------------------------------------------
## wt
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       32        0       29    0.999    3.217    1.089    1.736    1.956
##      .25      .50      .75      .90      .95
##    2.581    3.325    3.610    4.048    5.293
##
## lowest : 1.513 1.615 1.835 1.935 2.140, highest: 3.845 4.070 5.250 5.345 5.424
## ------------------------------------------------------------------------------
```

```
library(pastecs)
myvars <- c("mpg", "hp", "wt")
stat.desc(mtcars[myvars])
```

10.1.3.1.4 Descriptive statistics via `stat.desc()` in the `pastecs` package

```
##                      mpg           hp          wt
## nbr.val       32.0000000   32.0000000  32.0000000
## nbr.null       0.0000000    0.0000000   0.0000000
## nbr.na         0.0000000    0.0000000   0.0000000
## min           10.4000000   52.0000000   1.5130000
## max           33.9000000  335.0000000   5.4240000
## range         23.5000000  283.0000000   3.9110000
## sum          642.9000000 4694.0000000 102.9520000
## median        19.2000000  123.0000000   3.3250000
## mean          20.0906250  146.6875000   3.2172500
## SE.mean        1.0654240   12.1203173   0.1729685
## CI.mean.0.95   2.1729465   24.7195501   0.3527715
## var           36.3241028 4700.8669355   0.9573790
## std.dev        6.0269481   68.5628685   0.9784574
## coef.var       0.2999881    0.4674077   0.3041285
```

```
library(psych)
```

10.1.3.1.5 Descriptive statistics via `describe()` in the `psych` package

```
##
```

```
## Attaching package: 'psych'

## The following object is masked from 'package:Hmisc':
##
##     describe

## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
myvars <- c("mpg", "hp", "wt")
describe(mtcars[myvars])
```

```
##     vars  n   mean    sd median trimmed   mad   min    max  range skew kurtosis
## mpg    1 32  20.09  6.03  19.20   19.70  5.41 10.40  33.90  23.50 0.61    -0.37
## hp     2 32 146.69 68.56 123.00  141.19 77.10 52.00 335.00 283.00 0.73    -0.14
## wt     3 32   3.22  0.98   3.33    3.15  0.77  1.51   5.42   3.91 0.42    -0.02
##        se
## mpg  1.07
## hp  12.12
## wt   0.17
```

10.1.3.1.6 Descriptive statistics by group using `by()`

When comparing groups of individuals or observations,

- the focus is usually on the descriptive statistics of each group,
- rather than the total sample.

Group statistics can be generated using base R's `by()` function.

- The format is `by(data, INDICES, FUN)`
  - where data is a data frame or matrix,
  - `INDICES` is a factor or list of factors that defines the groups,
- and `FUN` is an arbitrary function that operates on
  - all the columns of a data frame.

```
dstats <- function(x)
  sapply(x, mystats)
myvars <- c("mpg", "hp", "wt")
by(mtcars[myvars], mtcars$am, dstats)
```

```
## mtcars$am: 0
##                  mpg           hp         wt
## n        19.00000000  19.00000000 19.0000000
## mean     17.14736842 160.26315789  3.7688947
## stdev     3.83396639  53.90819573  0.7774001
## skew      0.01395038  -0.01422519  0.9759294
## kurtosis -0.80317826  -1.20969733  0.1415676
## ------------------------------------------------------------
## mtcars$am: 1
##                  mpg          hp         wt
## n        13.00000000  13.0000000 13.0000000
## mean     24.39230769 126.8461538  2.4110000
## stdev     6.16650381  84.0623243  0.6169816
## skew      0.05256118   1.3598859  0.2103128
## kurtosis -1.45535200   0.5634635 -1.1737358
```

10.1.3.1.7 Descriptive statistics for groups defined by multiple variables

In the next example, summary statistics are generated

- for two by variables (am and vs)
- and the results for each group are printed with custom labels.

Additionally, missing values

- are omitted before calculating statistics.

```
dstats <- function(x)
  sapply(x, mystats, na.omit = TRUE)
myvars <- c("mpg", "hp", "wt")
by(mtcars[myvars],
   list(Transmission = mtcars$am,
        Engine = mtcars$vs),
   FUN = dstats)
```

```
## Transmission: 0
## Engine: 0
##                  mpg          hp         wt
## n        12.0000000  12.0000000 12.0000000
## mean     15.0500000 194.1666667  4.1040833
## stdev     2.7743959  33.3598379  0.7683069
## skew     -0.2843325   0.2785849  0.8542070
## kurtosis -0.9635443  -1.4385375 -1.1433587
## ------------------------------------------------------------
## Transmission: 1
## Engine: 0
##                  mpg          hp          wt
## n         6.0000000   6.0000000  6.00000000
## mean     19.7500000 180.8333333  2.85750000
## stdev     4.0088652  98.8158219  0.48672117
## skew      0.2050011   0.4842372  0.01270294
## kurtosis -1.5266040  -1.7270981 -1.40961807
## ------------------------------------------------------------
## Transmission: 0
## Engine: 1
##                  mpg          hp         wt
## n         7.0000000   7.0000000  7.0000000
## mean     20.7428571 102.1428571  3.1942857
## stdev     2.4710707  20.9318622  0.3477598
## skew      0.1014749  -0.7248459 -1.1532766
## kurtosis -1.7480372  -0.7805708 -0.1170979
## ------------------------------------------------------------
## Transmission: 1
## Engine: 1
##                  mpg         hp         wt
## n         7.0000000  7.0000000  7.0000000
## mean     28.3714286 80.5714286  2.0282857
## stdev     4.7577005 24.1444068  0.4400840
## skew     -0.3474537  0.2609545  0.4009511
## kurtosis -1.7290639 -1.9077611 -1.3677833
```

10.1.3.2 Summarizing data interactively with dplyr

So far, we've focused on methods

- that generate a comprehensive set of descriptive statistics
- for a given data frame.

However, in interactive, exploratory data analyses,

- our goal is to answer targeted questions.

In this case, we'll want to obtain

- a limited number of statistics
- on specific groups of observations.

The dplyr package, provides us with tools

- to quickly and flexibly accomplish this.

The summarize(), and summarize_all() functions

- can be used to calculate any statistic,
- and the group_by() function
  - can be used to specify the groups
  - on which to calculate those statistics.

As a demonstration, let's ask and answer a set of questions

- using the Salaries data frame in the carData package.

The dataset contains 2008-2009 9-month salaries in US dollars (salary)

- for 397 faculty members at a university in the United States.

The data were collected as part of ongoing efforts

- to monitor salary differences
- between male and female faculty.

```
library(dplyr)
library(carData)

Salaries %>%
  summarize(med = median(salary),
            min = min(salary),
            max = max(salary))
```

```
##      med   min    max
## 1 107300 57800 231545
```

The Salaries dataset is passed to the summarize() function,

- which calculates the median, minimum and maximum value for salary
- and returns the result as a one row tibble (data frame).

The median 9-month salary is $107,300 and

- at least one person was making more than $230,000.
- I clearly need to ask for a raise.

What is the

- faculty count,
- median salary,
- and salary range
  - by sex and rank?

```
Salaries %>%
  group_by(rank, sex) %>%
  summarize(
    n = length(salary),
    med = median(salary),
    min = min(salary),
    max = max(salary)
  )
```

```
## `summarise()` has grouped output by 'rank'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 6 x 6
## # Groups:   rank [3]
##   rank      sex        n     med   min    max
##   <fct>     <fct>  <int>   <dbl> <int>  <int>
## 1 AsstProf  Female    11  77000  63100  97032
## 2 AsstProf  Male      56  80182  63900  95079
## 3 AssocProf Female    10  90556. 62884 109650
## 4 AssocProf Male      54  95626. 70000 126431
## 5 Prof      Female    18 120258. 90450 161101
## 6 Prof      Male     248 123996  57800 231545
```

When categorical variables are specified in a by_group() statement,

- the summarize() function generates a row of statistics
- for each combination of their levels.

Women have a lower median salary

- than men within each faculty rank.

In addition,

- there are a very large number of male full professors
- at this university.

What is the mean years of service and years since Ph.D.

- for faculty
- by sex and rank?

```
Salaries %>%
  group_by(rank, sex) %>%
  select(yrs.service, yrs.since.phd) %>%
  summarize_all(mean)
```

```
## Adding missing grouping variables: `rank`, `sex`
```

```
## # A tibble: 6 x 4
## # Groups:   rank [3]
##   rank      sex    yrs.service yrs.since.phd
##   <fct>     <fct>        <dbl>         <dbl>
## 1 AsstProf  Female        2.55          5.64
## 2 AsstProf  Male          2.34          5
## 3 AssocProf Female       11.5          15.5
## 4 AssocProf Male         12.0          15.4
## 5 Prof      Female       17.1          23.7
## 6 Prof      Male         23.2          28.6
```

The `summarize_all()` function

- calculates a summary statistic for each non-grouping variable
  - (yrs.service and yrs.since.phd here).

If you want more than one statistic for each variable,

- provide them in a list.
- For example, `summarize_all(list(mean = mean, std = sd))`
  - would calculate the mean and standard deviation for each variable.

Men and women have comparable experience histories

- at the Assistant and Associate Professor levels.

However, female Full Professors

- have fewer years of experience
- than their male counterparts.

One advantage of the dplyr approach is that

- results are returned as tibbles (data frames).

This allows you to

- analyze these summary results further,
- lot them,
- and reformat them for printing.

This is taking advantage of the object oriented nature of R.

It also provides an easy mechanism for aggregating data.

In general, data analysts have their own preferences

- for which descriptive statistics to display
- and how they like to see them formatted.

This is probably why there are many variations available.

Choose the one that works best for you, or create your own!

10.1.4 Links

Robert I. Kabacoff, R in Action, 3rd Edition, Manning Publications 2020