BIOSTATS 640 – Introduction to R
Fall 2024
https://people.umass.edu/biep640w/webpages/demonstrations.html

https://www.simplilearn.com/what-is-descriptive-statistics-article

03
Numerical Summarization &
One and Two Sample Inference
September 27, 2024

Right click to download R dataset
sepsis.Rdata

Welcome to Lesson 03!
In 2024, this introduction to using R Studio for one and two sample inference contains a few additional illustrations. These include: importing an Excel dataset, an introduction to factors in R (needed for analyses of categorical data), and some data visualizations. Have fun!

1. At a Glance

How to Work with Factors

Create a factor	`# from a character object`
	`names <- factor(c("Bob","Carol","Ted"))`
	
	`# from existing (numeric) vector`
	`v <- c(1,1,0,1)`
	`exposed <- factor(v,`
	`levels = c(0,1),`
	`labels = c("Not exposed", "Exposed"))`

Set levels
default is alphabetic
	`# User sets levels – nominal`
	`coffee <- c("small", "medium", "small", "large", "small", "large")`
	`coffeef <- factor(coffee,`
	`levels=c("small", "medium", "large"))`
	
	`# User sets levels - ordinal`
	`coffee <- c("small", "medium", "small", "large", "small", "large")`
	`coffeef <- factor(coffee,`
	`levels=c("small", "medium", "large")`
	`ordered=TRUE)`

One Sample Inference - Continuous

Numerical Summarization	`summary(outcome)` # Method 1
	
	`library(summarytools)` # Method 2
	`descr(df$outcome,`
	`  stats=c("n.valid", "mean", "sd", "med", "min", "max"),` # User chooses
	`  transpose=TRUE)`

Visualization	`library(ggplot2)`
	
	`# Small sample size:  Dot Plot`
	`ggplot(data=df) +`
	`  aes(x=1, y=outcome) +`
	`  geom_dotplot(binaxis='y', stackdir="center", na.rm=T)`
	
	`# Box and Whisker`
	`ggplot(data=df) +`
	`  aes(x=" ", y=outcome) +`
	`  geom_boxplot(na.rm=T) +`
	`  coord_flip( )` # (optional) for horizontal)
	
	`# Histogram – Optional Solution for binwidth, bw`
	`bw <- 2*IQR(outcome, na.rm=T)/length(outcome)^1/3`
	
	`# Histogram`
	`ggplot(data=df) +`
	`  aes(x=outcome) +`
	`  geom_histogram(binwidth=bw, na.rm=T)`

Confidence Interval Estimation	`# Confidence Interval for mean`
	`t.test(outcome ~ 1, data=df, conf.level=.90)$conf.int` # Default is conf.level=.95
	
	`# Confidence Interval for variance`
	`library(DescTools)`
	`VarTest(df$outcome, conf.level=.90)$conf.int` # Default is conf.level=.95

Hypothesis Testing	`# Wilcoxon Signed Rank Test`
	`Wilcox.test(x=df$outcome, mu=nullmedian, na.rm=T, correct=T,`
	`  alternative="two.sided")` # "two.sided", "greater", "less"
	
	`# One Sample t-test of mean`
	`t.test(outcome ~ 1, data=df, mu=nullmean)`
	
	`# One Sample t-test of variance`
	`library(DescTools)`
	`VarTest(df$outcome, sigma.squared=nullvariance)`

One Sample Inference – Discrete (Single Proportion)

Numerical Summarization

```
summary(outcome)                                                    # Method 1

library(summarytools)                                               # Method 2
freq(df$outcome)                                                    # Outcome must be factor
```

Visualization

```
library(ggplot2)

# Bar Chart with Counts Displayed (Outcome must be factor)
ggplot(data=df) +
       aes(x=outcome) +
       geom_bar(na.rm=T) +
       geom_text(stat='count',
                 aes(label=..count..),
                 size=fillin,                                        # Try size = 2.5
                 vjust=fillin)                                       # Try fillin = -1

# Bar Chart With Percents Displayed (Outcome must be factor)
ggplot(data=df) +
      aes(x=outcome) +
      aes(y=prop.table(stat(count))) +
      label = scales::percent(prop.table(stat(count))) +
      geom_bar(position="dodge", na.rm=T) +
      geom_text(stat='count',
                position = position_dodge(fillin),                   # Try position_dodge(.9)
                size=fillin ,                                        # Try size = 3
                vjust=fillin) +                                      # Try vjust = -.5
      scale_y_continuous(labels = scales::percent, limits=c(0,1))
```

Confidence Interval Estimation

```
# Confidence Interval for proportion – EXACT
binom.test(x=#events,n=ntrials,conf.level=.90)$conf.int       # Default is conf.level=.95

# Confidence Interval for proportion – NORMAL APPROXIMATION
prop.test(x=#events,n=ntrials,conf.level=.90)$conf.int        # Default is conf.level=.95
```

Hypothesis Testing

```
# Hypothesis Test for Binomial Proportion – EXACT
binom.test(x=#events,n=ntrials,p=nullp,
           alternative="less")                                # "two.sided", "greater", "less"

# Hypothesis Test for Binomial Proportion – NORMAL APPROXIMATION
prop.test(x=#events,n=ntrials,p=nullp,
           alternative="less")                                # "two.sided", "greater", "less"
```

One Sample PAIRED - Continuous

How to Reshape

```
*  From WIDE to LONG
library(tidyverse)

longdf <- wide_old %>%
            pivot_longer(
               cols= c(salary2000,salary2010,salary2020),       # repeated measures WIDE vars
               names_to="year",                                 # NEW long variable for time
               names_prefix = "salary",                         # DROP the prefix "salary"
               values_to="salary")                              # NEW long variable outcome


*  From LONG to WIDE
library(tidyverse)
widedf <- long_old %>%
          pivot_wider(
            names_from=c("visit"),            # NEW wide var name (the start of it)
            names_prefix="sbp_visit",         # ADD this prefix to new wide var name
            values_from="sbp")                # Get values of new wide var from this old var
```

Numerical Summarization

```
* WIDE:  Paired variables (e.g., pre and post) in WIDE format
myvars <- c("prevar", "postvar")
descr(df[myvars],
      stats=c("n.valid", "mean", "sd", "med", "min", "max"),        # User chooses
      transpose=TRUE)

* LONG:  Paired variables (e.g., pre and post) are in LONG FORMAT
library(summarytools)
with(df,
   stby(data = outcomevar,
        INDICES = timevar,                                          # timevar must be factor
        FUN = descr, stats = c("mean", "sd", "min", "med","max"),   # User chooses
        transpose=TRUE))
```

Visualization

```
library(tidyverse)
library(ggplot2)

# Plot individual profiles:  Data MUST be in LONG format
longdf %>% arrange(idvar)                          # Possibly not necessary, but sort by idvar
ggplot(data=longdf) +
  aes(x=timevar, y=outcomevar, group=idvar) +
  geom_line()+
  geom_point()
```

Confidence Interval Estimation

```
# Confidence Interval for mean
t.test(outcome ~ 1, data=df, conf.level=.90)$conf.int         # Tip. Outcome = post - pre

# Confidence Interval for variance
library(DescTools)
VarTest(df$outcome, conf.level=.90)$conf.int                   # Default is conf.level=.95
```

Hypothesis Testing

```
# Wilcoxon Signed Rank Test
Wilcox.test(x=df$outcome, mu=nullmedian, na.rm=T, correct=T,
     alternative="two.sided")                                  # "two.sided", "greater", "less"

# One Sample t-test of mean
t.test(outcome ~ 1, data=df, mu=nullmean)

# One Sample t-test of variance
library(DescTools)
VarTest(df$outcome, sigma.squared=nullvariance)
```

Two Independent Samples Inference - Continuous

Numerical Summarization

```
* Data are in LONG format
by(df[ , c("outcomevar")],                                  # summarize only outcomevar
   df$groupvar,                                             # grouping variable
   summary)                                                 # use function summary in {base}

library(summarytools)
with(df,
   stby(data = outcomevar,
        INDICES = groupvar,                                      # groupvar must be factor
        FUN = descr, stats = c("mean", "sd", "min", "med","max"),   # User chooses
        transpose=TRUE))
```

Visualization

```
library(ggplot2)

# Small sample size:  Side-by-Side Dot Plot
ggplot(data=df) +
       aes(x=groupvar, y=outcome) +                          # groupvar must be factor
       geom_dotplot(binaxis='y', stackdir="center", na.rm=T)

# Side-by-Side Box and Whisker
ggplot(data=df) +
       aes(x=" ", y=outcome, fill=groupvar) +                # groupvar must be factor
       geom_boxplot(na.rm=T) +
       coord_flip( )                                         # (optional) for horizontal)

# Histogram – Optional Solution for binwidth, bw
bw <- 2*IQR(outcome, na.rm=T)/length(outcome)^1/3

# Side-by-Side Histogram (cb choice here)
ggplot(data=df) +
       aes(x=outcome) +
       geom_histogram(binwidth=bw, na.rm=T) +
       facet_grid(groupvar ~ .)                              # 2 panels by groupvar in column
```

Confidence Interval Estimation

```
* LONG:  data are in LONG format
# Confidence Interval for mean difference (group1 – group2)
t.test(outcome ~ groupvar, data=df, conf.level=.90)$conf.int
```

Hypothesis Testing

```
# Wilcoxon Rank Sum Test of Equality of Medians
Wilcox.test(outcome ~ groupvar, data=df, na.rm=T, correct=T,
     alternative="two.sided")                               # "two.sided", "greater", "less"

# Two Sample Test of Equality of Variances
var.test(outcome ~ groupvar, data=df,
      alternative = "two.sided")                            # "two.sided", "greater", "less"

# Two Sample Test of Equality of Means – UNEQUAL variances
t.test(outcome ~ groupvar, data=df,
    alternative="two.sided")                               # "two.sided", "greater", "less"

# Two Sample Test of Equality of Means – EQUAL variances
t.test(outcome ~ groupvar, data=df, var.equal=TRUE,
    alternative="two.sided")                               # "two.sided", "greater", "less"
```

Two Sample Inference – Discrete (Two Independent Proportions)

Numerical Summarization	`table(df$discrete1,discrete2, useNA="always")` # Method 1 `library(summarytools)` # Method 2 `with(df,` `  ctable(rowvar, colvar,` # vars must be factor `  prop="n"),` # User chooses "n", "r", "c" `  totals=TRUE)` # use this if you want totals

Visualization	`library(ggplot2)`
	
	`# Bar Chart of discrete outcome by discrete predictor`
	`ggplot(data=df) +`
	`  aes(x=predictorf) +` # x = predictor var, must be factor
	`  aes(fill=outcomef) +` # fill = outcome var, must be factor
	`  geom_bar(position="dodge")` # User chooses "dodge" or "stack"
	
	`# Bar Chart of discrete outcome by discrete predictor – DISPLAY PERCENTS`
	`ggplot(data=df) +`
	`  aes(x=predictorf) +` # x = predictor var, must be factor
	`  aes(fill=outcomef) +` # fill = outcome var, must be factor
	
	`  geom_bar(aes( y=..count../tapply(..count.., ..x.. ,sum)[..x..]), # required`
	`  position="dodge") +` # must be side-by-side (not stacked)
	
	`  geom_text(aes( y=..count../tapply(..count.., ..x.. ,sum)[..x..],` # required
	`  label=scales::percent(..count../tapply(..count.., ..x.. ,sum)[..x..]) ), # required`
	`  stat="count",` # required
	`  position=position_dodge(1.0),` # for centering
	`  vjust=-0.5,`
	`  size=3) +`
	
	`  scale_y_continuous(limits=c(0,100)) +` # Good to force scale to full
	`  scale_y_continuous(labels = scales::percent)` # Dispay as percents 0% to 100%

Hypothesis Testing	`# Fisher Exact Test of Equality of Proportions (NULL:  Odds Ratio = 1)`
	`fisher.test(df$rowvar,df$colvar)`
	
	`# Chi Square Test of Equality of Proportions – WITH continuity correction (default)`
	`chisq.test(df$rowvar,df$colvar)`
	
	`# Chi Square Test of Equality of Proportions – WITHOUT continuity correction`
	`chisq.test(df$rowvar,df$colvar, correct=FALSE)`

2. Introduction to the Ibuprofen Sepsis Study (`sepsis.Rdata`)

Source:
Bernard GR, Wheeler AP, and Russell JA et. al (1997) The effects of ibuprofen on the physiology and survival of patients with sepsis. *New England Journal of Medicine, Vol. 336, No. 13, 912-918.*

Sepsis is a potentially life threatening condition that occurs when the body's immune system produces an extreme response to an infection. It can cause tissue damage, organ failure and death. The Ibuprofen Sepsis Study Group conducted a randomized controlled trial comparing ibuprofen versus placebo for the treatment of sepsis.

n=455
# variables = 23

Data dictionary/Codebook

Variable	Label	Type	Codings	Missing
id	Patient ID	numeric	range: 1 - 455	0 (0%)
treat	Treatment	numeric, labelled	0 = Placebo 1 = Ibuprofen	0 (0%)
race	Race	numeric, labelled	0 = White 1 = AfricanA 2 = Other	0 (0%)
apache	Baseline Apache Score	numeric	range: 0 - 41	1 (0.2%)
o2del	Oxygen Delivery at Baseline (ml/min/m^2)	numeric	range: 316.9 - 947.2	287 (63.1%)
fate	Mortality Status at 30 Days	numeric, labelled	0 = Alive 1 = Died	0 (0%)
followup	Follow-up (hours)	numeric	range: 1 - 720	0 (0%)
temp0	Baseline Temperature (degrees F)	numeric	range: 91.6 - 107	0 (0%)
temp1	Temperature after 2 hours (degrees F)	numeric	range: 90.7 - 106.7	35 (7.7%)
temp2	Temperature after 4 hours (degrees F)	numeric	range: 93.6 - 107.9	53 (11.6%)
temp3	Temperature after 8 hours (degrees F)	numeric	range: 92.3 - 104.7	37 (8.1%)
temp4	Temperature after 12 hours (degrees F)	numeric	range: 90.3 - 104.5	34 (7.5%)
temp5	Temperature after 16 hours (degrees F)	numeric	range: 91.6 - 104.4	33 (7.3%)
temp6	Temperature after 20 hours (degrees F)	numeric	range: 88.9 - 104.5	23 (5.1%)
temp7	Temperature after 24 hours (degrees F)	numeric	range: 88.7 - 104.2	42 (9.2%)
temp8	Temperature after 28 hours (degrees F)	numeric	range: 93.4 - 103.6	48 (10.5%)

Data dictionary/Codebook - *continued*

Variable	Label	Type	Codings	Missing
temp9	Temperature after 32 hours (degrees F)	numeric	range: 93.2 - 104.4	54 (11.9%)
temp10	Temperature after 36 hours (degrees F)	numeric	range: 92.3 - 104	56 (12.3%)
temp11	Temperature after 40 hours (degrees F)	numeric	range: 92.1 - 103.4	53 (11.6%)
temp12	Temperature after 44 hours (degrees F)	numeric	range: 91.2 - 103.3	49 (10.8%)
temp13	Temperature after 72 hours (degrees F)	numeric	range: 93.7 - 104.9	52 (11.4%)
temp14	Temperature after 96 hours (degrees F)	numeric	range: 92.2 - 103.3	139 (30.5%)
temp15	Temperature after 120 hours (degrees F)	numeric	range: 95 - 105.1	73 (16.0%)

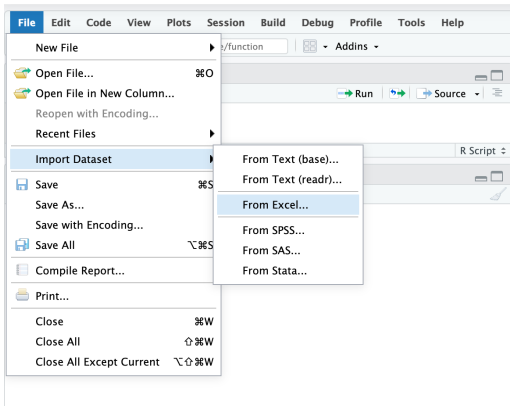
3. How to Import an Excel Dataset into R Studio

Preliminaries (Important):

(1) Make sure that you have downloaded from the course website the dataset arthritis.xlsx.
(2) Strongly encouraged: *(Source: marinstats lectures)* Importing Excel Data into R (video, 8:12)

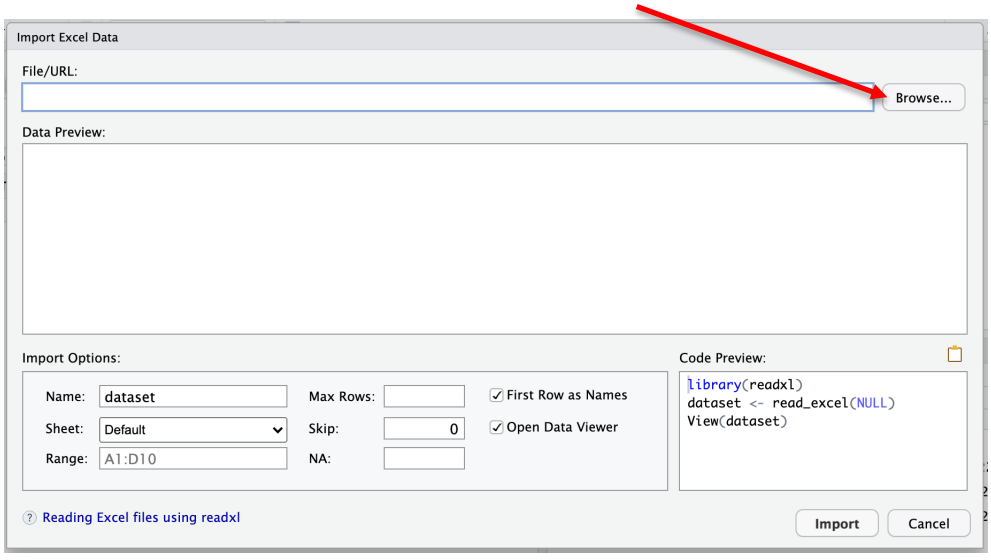
How to Import Excel Data Using R Studio Menus

Step 1: At upper left; FILE > IMPORT DATASET > FROM EXCEL

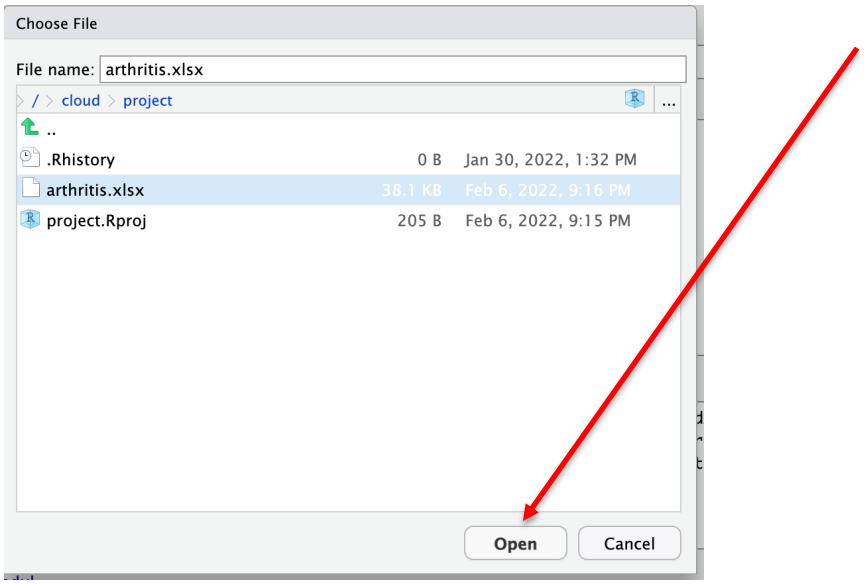


Note: R may return a message saying that you need to install readxl. Click YES. Then wait until you get a prompt.

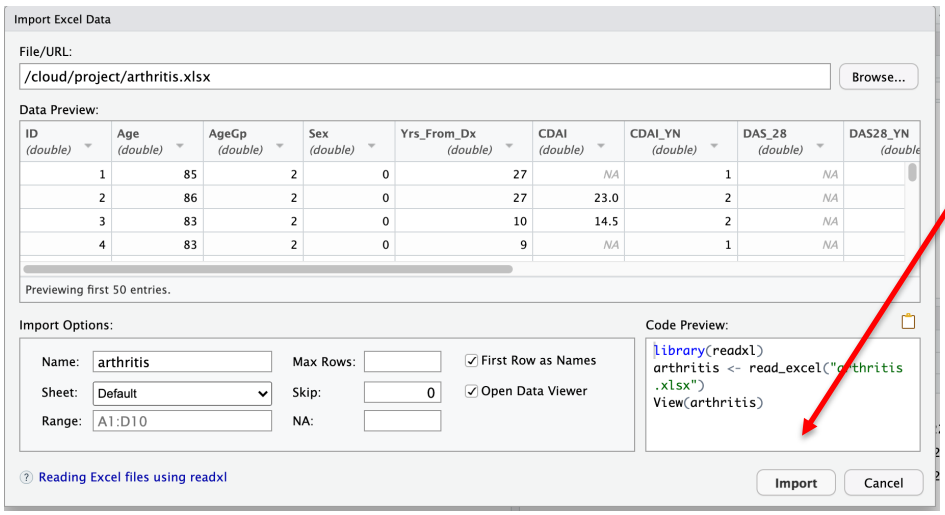
Step 2: At upper right, click on the icon BROWSE.



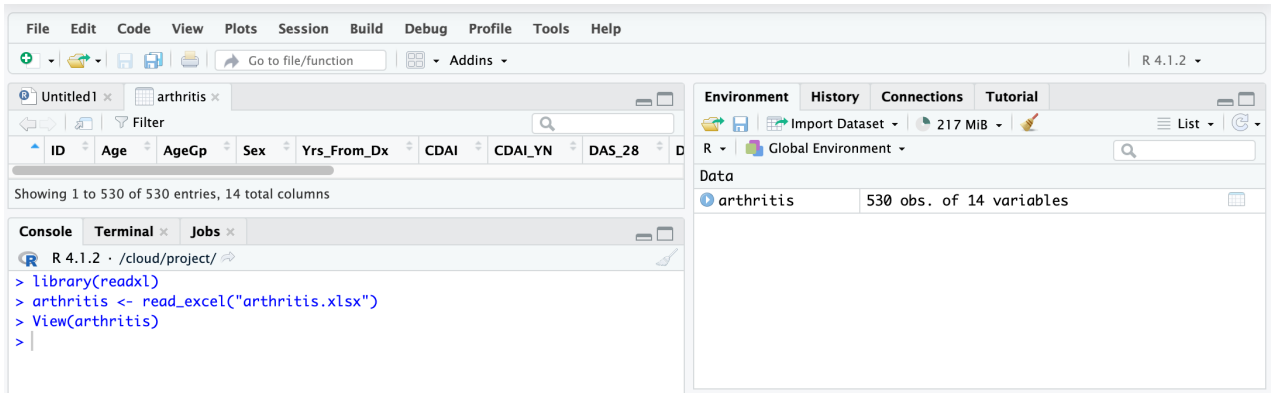
Step 3: Navigate to choose arthritis.xlsx. At lower right, click OPEN



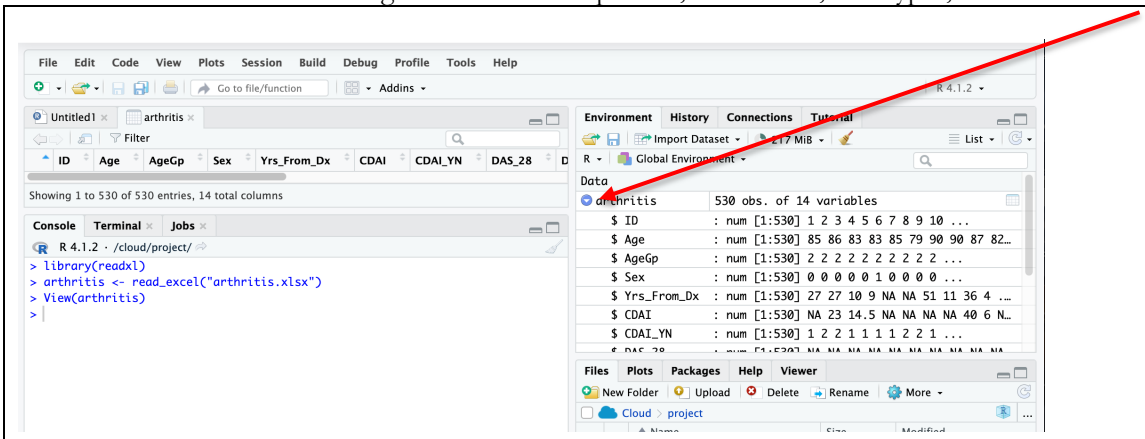
Step 4: Take your time here in making your selections. All set? At lower right, click IMPORT



You should now see the following



Step 5: At right, in the Environment tab, click on the down arrow next to the dataset name arthritis.
You should then see the following information: sample size, # variables, data types, etc.



4. Categorical Variables in R: Introduction to Factors

Exploration and analysis of categorical data in R involves working with R factor objects. In statistical/biostatistical parlance, we talk about categorical variables. A categorical variable might be:

- Qualitative/nominal: e.g., state of residence (MA, NY, CT, etc)
- Qualitative/ordinal: e.g., level of pain ("low", "medium", "high")
- Quantitative discrete count: e.g., number of visits to the dentist (0, 1, 2, etc.)

Character and factor vectors are NOT the same!

Character Vector	Factor Vector
• R stores as string	• R stores as whole numbers 1, 2, 3, etc Why: R stores factor levels internally ("under the hood") as numbers. This is necessary in order for the factor vector to be used in data analysis. *Makes sense, right?*
• Can contain as many "values" as you like.	• Can contain ONLY pre-defined levels

• Print will display elements in quotes
 	
Example:
`> names <- c("Bob", "Carol", "Ted")`
`> names`
`[1] "Bob" "Carol" "Ted"`
	• Print will display elements WITHOUT quotes; Note: R will also show you the levels
	
	Example:
	`> season <-`
	`factor(c("summer","fall","fall","winter","winter",`
	`"spring"),levels=c("winter","spring","summer","fall"))`
	
	`> season`
	`[1] summer fall fall winter winter spring`
	`Levels: winter spring summer fall`

• CANNOT be used in data analysis • summary( ) will produce NO OUTPUT	• Can be used in data analysis • summary( ) will produce frequency table

How to Work with Factors

Create a factor

Use the functions factor( ) and c( )

```
# from scratch
coffee <-  factor(  c("small", "medium", "large)  )

# from existing (numeric) vector
v <- c(1,1, 0, 1)
exposed <- factor(v,
                              levels = c(0,1),
                              labels = c("Not exposed", "Exposed) )

# convert character to factor
v <-  c("Boston", "Atlanta", "Seattle")
city <- factor(v)
```

Examine properties

```
# show datatype
class(v)
typeof(v)
attributes(v)

# show names of factor levels
levels(coffee)

# show number of factor levels
levels(coffee)
```

Set levels

IMPORTANT: by default R stores levels ALPHABETICALLY

```
# set levels explictly - nominal
coffee <- factor(coffee, levels = c("small", "medium", "large") )

# set levels explictly - ordinal
coffee <- factor(coffee, levels = c("small", "medium", "large"), ordered=TRUE)
coffee <- as.ordered(coffee)
```

How to Work with Factors - *continued*

Set levels - *continued*

```
#  add a level
city <-  c("Boston", "Atlanta", "Seattle")
levels(city) <- c(levels(city), "Philadelphia")

#  drop UNUSED levels
droplevels(city)

#  change the names of the levels entirely using relevel( ) in package {plyr}
relevel(coffee, c("small"="Tall", "medium" = "Grande", "large" = "Venti"))
In this example, the old names are "small","medium", "Large" and the new names are "Tall", "Grande" and "Venti"
```

Set reference level Tip - This is useful in regression

```
# change reference level (default is level 1)
relevel(coffee, ref="medium")
```

Statistics

```
# the following will NOT work if factor is unordered (same for other statistical functions)
min(coffee)

# the following will work if factor is ordered
min(coffee)
```

5. One Sample Inference

Dataset (right click to download):
sepsis.Rdata

Packages used:
{DescTools}, {stargazer}, {summarytools} {tidyverse}

Tip for Hypothesis Testing

Alternative Hypothesis	R Code option
Two sided	, alternative="two.sided"
Right tail	, alternative="greater"
Left tail	, alternative="less"

Tip for Confidence Intervals

If you want ...	R Code option
95% CI	Nothing you need to do ... this is default
90% CI	, conf.level = .90
... and so on	, conf.level = .FILLIN

Load R dataset to session

Step 1: If you have not already done so, right click to download sepsis.Rdata from Canvas or the public course website.

Step 2: R Studio/Posit in the Cloud Users Only) Upload sepsis.Rdata

Step 3: Put sepsis.Rdata into your working directory

Step 4: load(file="sepsis.Rdata")

5.1. One Sample – Continuous Outcome
Normal Distribution Model

At a Glance

Numerical Summarization	`summary(outcome)` # Method 1
	
	`library(summarytools)` # Method 2
	`descr(df$outcome,`
	`  stats=c("n.valid", "mean", "sd", "med", "min", "max"),` # User chooses
	`  transpose=TRUE)`

Confidence Interval Estimation	`# Confidence Interval for mean`
	`t.test(outcome ~ 1, data=df, conf.level=.90)$conf.int` # Default is conf.level=.95
	
	`# Confidence Interval for variance`
	`library(DescTools)`
	`VarTest(df$outcome, conf.level=.90)$conf.int` # Default is conf.level=.95

Hypothesis Testing	`# One Sample t-test of mean`
	`t.test(outcome ~ 1, data=df, mu=nullmean)`
	
	`# One Sample t-test of variance`
	`library(DescTools)`
	`VarTest(df$outcome, sigma.squared=nullvariance)`

Examples.

```
# Z Test of mean:  Population variance/standard deviation are KNOWN

library(DescTools)
ZTest(sepsis$o2del,
        mu=1000,                                    # null hypothesis mean
        sd_pop=409,                                 # known population standard deviation sigma
        alternative="greater")                      # alternative: true mean > null mean
```

```
         One Sample z-test

data:  sepsis$o2del
z = 0.75478, Std. Dev. Population = 409, p-value = 0.2252        # Null mu=1000 v mu > 1000 is NOT rejected
alternative hypothesis: true mean is greater than 1000
95 percent confidence interval:
 971.9137      Inf
sample estimates:
mean of x
 1023.817
```

```
# T-test of mean:  Population variance/standard deviation NOT known

t.test(o2del~1,                                     # model formulation
       data=sepsis,                                 # data to use
       mu=1200,                                     # null hypothesis mean
       alternative="two.sided",                     # alternative: true mean ≠ null mean
       conf.level=.90,                              # show 90% CI
       na.rm=TRUE)                                  # omit NA's (missing values)
```

```
         One Sample t-test

data:  o2del
t = -5.5773, df = 167, p-value = 0.00000009658              # 2 sided p << .0001.  Reject null (mu=1200)
alternative hypothesis: true mean is not equal to 1200
90 percent confidence interval:
  971.5676 1076.0665                                        # 90% CI does NOT contain null mu=1200
sample estimates:
mean of x
 1023.817
```

```
# Test of Variance
library(DescTools)                                  # You could use var.test( ) in {base}. I like this

VarTest(sepsis$o2del,
      sigma.squared=1600)                           # Null hypothesis variance (not SD!)
```

```
         One Sample Chi-Square test on variance

data:  sepsis$o2del
X-squared = 17498, df = 167, p-value < 0.00000000000000022      # 2 sided p << .0001 Reject null (sigma2 = 1600)
alternative hypothesis: true variance is not equal to 1600
95 percent confidence interval:
 136784.9 210324.6                                              # 95% CI does NOT contain null sigma2 = 1600
sample estimates:
variance of x
     167643.2
```

5.2. One Sample – Discrete Outcome
Binomial Distribution Model

At a Glance

Numerical Summarization	`summary(outcome)` # Method 1 `library(summarytools)` # Method 2 `freq(df$outcome)` # Outcome must be factor

Confidence Interval Estimation	`# Confidence Interval for proportion – EXACT`
	`binom.test(x=#events,n=ntrials,conf.level=.90)$conf.int` # Default is conf.level=.95
	
	`# Confidence Interval for proportion – NORMAL APPROXIMATION`
	`prop.test(x=#events,n=ntrials,conf.level=.90)$conf.int` # Default is conf.level=.95

Hypothesis Testing	`# Hypothesis Test for Binomial Proportion – EXACT`
	`binom.test(x=#events,n=ntrials,p=nullp,`
	`alternative="less")` # "two.sided", "greater", "less"
	
	`# Hypothesis Test for Binomial Proportion – NORMAL APPROXIMATION`
	`prop.test(x=#events,n=ntrials,p=nullp,`
	`alternative="less")` # "two.sided", "greater", "less"

Examples.

```
# Binomial Proportion:  Exact Inference

library(tidyverse)


# For small to moderate sample size – For illustration I will obtain a small sample size = 25
temp <- sepsis %>%
       sample_n(25, na.rm=TRUE)

xevents <- sum(temp$treat, na.rm=TRUE)            # sum of 0/1 events gives x = xevents = # successes
ntrials <- sum(!is.na(temp$treat))                # sum of !is.na gives n = ntrials = # trials


binom.test(x=xevents,n=ntrials,p=.5)                          # Hypothesis Test (Null: p = .50)
```

```
         Exact binomial test

data:  xevents and ntrials
number of successes = 14, number of trials = 25, p-value = 0.69
alternative hypothesis: true probability of success is not equal to 0.5     # p=.69 do NOT reject null proportion =.50
95 percent confidence interval:
 0.3492816 0.7559763
sample estimates:
probability of success
                  0.56
```

```
# Binomial Proportion:  Normal Approximation

library(tidyverse)

xevents <- sum(sepsis$treat, na.rm=TRUE)            # sum of 0/1 events gives x = xevents = # successes
ntrials <- sum(!is.na(sepsis$treat))                # sum of !is.na gives n = ntrials = # trials


prop.test(x=xevents,n=ntrials,p=.5, correct=FALSE)    # Hypothesis Test (Null: p = .50)
```

```
          1-sample proportions test without continuity correction

data:  xevents out of ntrials, null probability 0.5
X-squared = 0.10769, df = 1, p-value = 0.7428                # pvalue = .74 do NOT reject null proportion =.50
alternative hypothesis: true p is not equal to 0.5
95 percent confidence interval:
 0.4466279 0.5381163                                         # 95% CI contains the null proportion = .50
sample estimates:
        p
0.4923077
```

5.3. One Sample PAIRED – Continuous Outcome Normal Distribution Model

Preliminary – Is your paired data wide or long?

What is wide data?

Answer: For each studyid, the pre and post data are in the SAME row (horizontal)
e.g., pre = sbp1 and post = sbp2

```
  studyid sbp1 sbp2
1       1  120  115
2       2  140  138
```

What is long data?

Answer: For each studyid, the pre and post data are each in their OWN/SEPARATE rows (vertical)
In long data, you have a variable that tells you occasion (pre v post)
and another variable that is the outcome

```
  studyid    visit sbp
1       1      pre 120
2       1     post 115
3       2      pre 140
4       2     post 138
```

At a Glance

Numerical Summarization	`* WIDE: Paired variables (e.g., pre and post) in WIDE format`
	`myvars <- c("prevar", "postvar")`
	`descr(df[myvars],`
	`  stats=c("n.valid", "mean", "sd", "med", "min", "max"),  # User chooses`
	`  transpose=TRUE)`
	
	`* LONG: Paired variables (e.g., pre and post) are in LONG FORMAT`
	`library(summarytools)`
	`with(df,`
	`  stby(data = outcomevar,`
	`  INDICES = timevar,  # timevar must be factor`
	`  FUN = descr, stats = c("mean", "sd", "min", "med","max"),  # User chooses`
	`  transpose=TRUE))`

Confidence Interval Estimation	`# Confidence Interval for mean`
	`t.test(outcome ~ 1, data=df, conf.level=.90)$conf.int  # Tip. Outcome = post - pre`
	
	`# Confidence Interval for variance`
	`library(DescTools)`
	`VarTest(df$outcome, conf.level=.90)$conf.int  # Default is conf.level=.95`

Hypothesis Testing	`# One Sample t-test of mean`
	`t.test(outcome ~ 1, data=df, mu=nullmean)`
	
	`# One Sample t-test of variance`
	`library(DescTools)`
	`VarTest(df$outcome, sigma.squared=nullvariance)`

Examples.

```
# Paired Data Student t-Test:  WIDE

t.test(sepsis$temp0,sepsis$temp7, paired=TRUE,                  # data in WIDE
       var.equal=FALSE,
       na.rm=TRUE)
```

```
          Paired t-test

data:  sepsis$temp0 and sepsis$temp7
t = 13.144, df = 412, p-value < 0.00000000000000022          # p << .0001.  Null of equality pre/post is rejected
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 1.093632 1.478282                                           # 95% CI does NOT contain Null difference of 0
sample estimates:
mean of the differences
              1.285957
```

```
# Paired Data Student t-Test: LONG

library(tidyverse)

# paired t LONG requires sorted by id then by occasion nested in id
longdf <- longdf %>%
          arrange(id, hour)

# Now do paired t - LONG
t.test(temp ~ hour, data=longdf, paired=TRUE)
```

```
          Paired t-test

data:  temp by hour
t = 13.144, df = 412, p-value < 0.00000000000000022          # p << .0001.  Null of equality pre/post is rejected
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 1.093632 1.478282                                           # 95% CI does NOT contain Null difference of 0
sample estimates:
mean of the differences
              1.285957
```

6. Two Sample Inference

Dataset (right click to download):
sepsis.Rdata

Packages used:
{DescTools}, {stargazer}, {summarytools} {tidyverse}

Tip for Hypothesis Testing

Alternative Hypothesis	R Code
Two sided	, alternative="two.sided"
Right tail	, alternative="greater"
Left tail	, alternative="less"

Tip for Confidence Intervals

If you want ...	R Code
95% CI	Nothing you need to do ... this is default
90% CI	, conf.level = .90
... and so on	, conf.level = .FILLIN

Load R dataset to session

Step 1: If you have not already done so, right click to download sepsis.Rdata from course website.

Step 2: R Studio/Posit in the Cloud Users Only) Upload `sepsis.Rdata`

Step 3: Put `sepsis.Rdata` into your working directory

Step 4: `load(file="sepsis.Rdata")`

6.1. Two Independent Samples – Continuous Outcome Normal Distribution Model

At a Glance

Numerical Summarization

```
* LONG:  data are in LONG format
by(df[ , c("outcomevar")],                                  # summarize only outcomevar
   df$groupvar,                                             # grouping variable
   summary)                                                 # use function summary in {base}

library(summarytools)
with(df,
   stby(data = outcomevar,
        INDICES = groupvar,                                          # groupvar must be factor
        FUN = descr, stats = c("mean", "sd", "min", "med","max"),    # User chooses
        transpose=TRUE))
```

Confidence Interval Estimation

```
* LONG:  data are in LONG format
# Confidence Interval for mean difference (group1 – group2)
t.test(outcome ~ groupvar, data=df, conf.level=.90)$conf.int
```

Hypothesis Testing

```
# Two Sample Test of Equality of Variances
var.test(outcome ~ groupvar, data=df,
      alternative = "two.sided")                            # "two.sided", "greater", "less"

# Two Sample Test of Equality of Means – UNEQUAL variances
t.test(outcome ~ groupvar, data=df,
    alternative="two.sided")                                # "two.sided", "greater", "less"

# Two Sample Test of Equality of Means – EQUAL variances
t.test(outcome ~ groupvar, data=df, var.equal=TRUE,
    alternative="two.sided")                                # "two.sided", "greater", "less"
```

Examples.

```
# Test of Equality of Variances

# REQUIRED:  group variable must be factor
sepsis$fatef <- factor(sepsis$fate,
                    levels=c(0,1),
                    labels=c("Alive", "Dead"))

var.test(o2del ~ fatef, data=sepsis)                    # Preliminary: test of vars

# Test of Equality of Means
t.test(o2del ~ fatef, data=sepsis,                      # t-test assuming equal var (provides CI, too)
        var.equal=TRUE)
```

```
        F test to compare two variances

data:  o2del by fatef
F = 0.91965, num df = 100, denom df = 66, p-value = 0.6975          # okay to assume equal variances
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.5846373 1.4175632
sample estimates:
ratio of variances
         0.9196542
```

```
        Two Sample t-test

data:  o2del by fatef
t = 2.5796, df = 166, p-value = 0.01076                              # reject Null of equal means
alternative hypothesis: true difference in means between group Alive and group Dead is not equal to 0
95 percent confidence interval:
  38.40254 288.94124
sample estimates:
mean in group Alive  mean in group Dead
          1089.0910            925.4191
```

6.2. Two Independent Samples – Discrete Outcome
Binomial Distribution Model

At a Glance

Numerical Summarization	`table(df$discrete1,df$discrete2, useNA="always")` # Method 1 `library(summarytools)` # Method 2 `with(df,` `  ctable(rowvar, colvar,` # vars must be factor `  prop="n"),` # User chooses "n", "r", "c" `  totals=TRUE)` # use this if you want totals

Hypothesis Testing	`# Fisher Exact Test of Equality of Proportions (NULL: Odds Ratio = 1)`
	`fisher.test(df$rowvar,df$colvar)`
	
	`# Chi Square Test of Equality of Proportions – WITH continuity correction (default)`
	`chisq.test(df$rowvar,df$colvar)`
	
	`# Chi Square Test of Equality of Proportions – WITHOUT continuity correction`
	`chisq.test(df$rowvar,df$colvar, correct=FALSE)`

Example.

```
mytable <- table(sepsis$treat,sepsis$fate)                  # Use table( ) to create table
dimnames(mytable) <- list(
    Treatment=c("Untreated","Treated"),
    Fate=c("Alive","Dead"))

mytable
chisq.test(mytable,correct=FALSE)                           # large n, no correction needed
```

```
          Fate
 Treatment  Alive Dead
  Untreated   139   92
  Treated     140   84

         Pearson's Chi-squared test

data:  mytable
X-squared = 0.25959, df = 1, p-value = 0.6104                # p-value = .61 Do NOT reject null of independence
```

7. Additional Resources

__1. *(Source: BIOSTATS 540 Fall 2022)*
One and Two Sample Inference: Choose a Test Chart (pdf, 3 pp)

__2. *(Source: DataAnalytics.org.uk)*
Basic Statistics (html)

__3. *(Source: UCLA Statistical Methods and Data Analytics)*
Choosing the Correct Statistical Test in SAS, Stata, SPSS and R (html)

__4. **Excellent!**
(Source: https://lindeloev.github.io/tests-as-linear/linear_tests_cheat_sheet.pdf)
Common Statistical Tests are Linear Models (html)