UNIVERSITY COLLEGE OF ENGINEERING, ANNA UNIVERSITY- (BIT CAMPUS) TIRUCHIRAPPALLI

R - PROGRAMMING TUTORIAL

GUIDED BY

DR. D. SENTHIL KUMAR

(AP - CSE)

PRESENTED BY

M.NOORUNNISHA

(810017405007)

What is R?

- Programming language
- Ross Ihaka and Robert Gentleman in 1993
- Statistical data manipulation and analysis

What R is good at?

- Statistics for relatively advanced users: R has thousands of packages, designed, maintained, and widely used by statisticians.
- Statistical graphics: try doing some of our plots in Stata and you won't have much fun.
- Flexible code: R has a rather liberal syntax, and variables don't need to be declared as they would in (for example) C++,
- Vectorization: R is designed to make it very easy to write functions which are applied pointwise to every element of a vector.
- R is powerful: if a command doesn't exist already, you can code it yourself.

What R is not so good at?

- Numerical methods, such as solving partial deferential equations
- Analytical methods, such as algebraically integrating a function.
- Precision graphics, such as might be useful in psychology experiments.

General Properties

- Interface R with other languages (C, C++, Fortran) to provide fast implementations of subroutines
- R is open source and widely adopted by statisticians, biostatisticians, and geneticists.
- Contributing new packages to the central repository (CRAN) is easy
- R is portable, and works equally well on Windows, OS X and Linux.

Interfaces

- RStudio, *http://www.rstudio.org/*
- StatET, *http://www.walware.de/goto/statet/*
- ESS (Emacs Speaks Statistics), *http://ess.r-project.org/*
- R Commander: John Fox, “The R Commander: A Basic-Statistics Graphical
- Interface to R,” *Journal of Statistical Software* 14, no. 9 (2005):1–42.
- JGR (Java GUI for R), *http://cran.rproject.org/web/packages/JGR/index.html*

How to Install R Studio

- RStudio is an integrated development environment (IDE) for R. It includes a console, syntax-highlighting editor that supports direct code execution, as well as tools for plotting, history, debugging and workspace management.
- You need to follow the following three steps in the same order.
  - **Install R**
  - **Install R-Studio**
  - **Install R-Packages (If needed)**

1. Install R

For Windows :

Download the binary setup file for R from the following link.(R for Windows)

- Open the downloaded .exe file and Install R

For Mac :

Download the appropriate version of .pkg file form the following link. (R for Mac)

- Open the downloaded .pkg file and Install R

For Linux :

For complete R System installation in Linux, follow the instructions on the following link (Link)

- For Ubuntu with Apt-get installed, execute *sudo apt-get install r-base* in terminal.

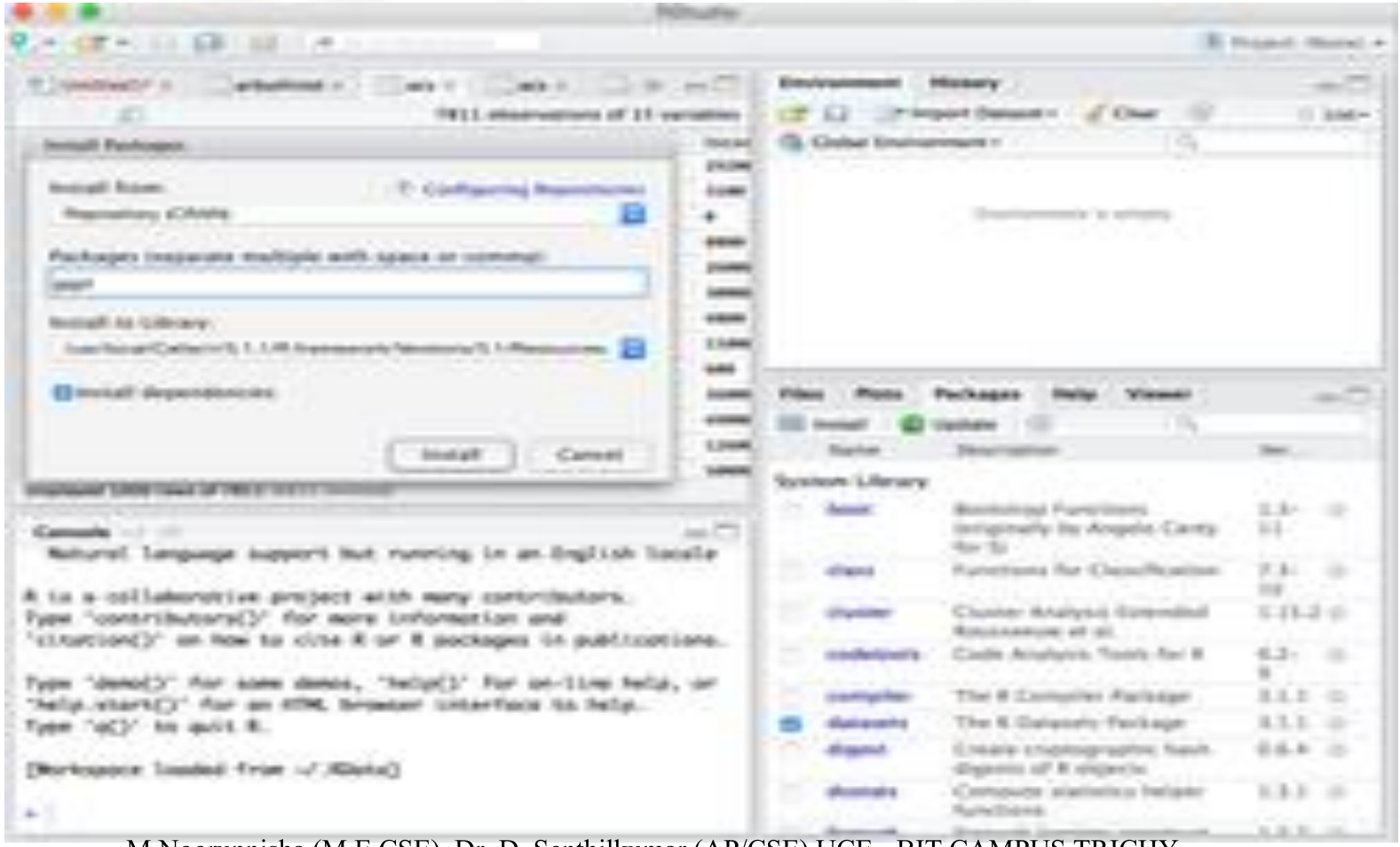
2. Install R Studio

- On the following link Download R Studio choose the appropriate installer file for your operating system, download it and then run it to install R-studio.

3. Install the packages (Optional)

- If your need to use R requires a particular package/library to be installed in R-studio. You can follow the instructions below to do so
  - Run R studio
  - Click on the Packages tab in the bottom-right section and then click on install. The following dialog box will appear
  - In the Install Packages dialog, write the package name you want to install under the Packages field and then click install. This will install the package you searched for or give you a list of matching package based on your package text.

R – STUDIO GUI



R – PROGRAMMING

INTRODUCTION AND BASICS

R – PROGRAMMING INTRODUCTION AND BASICS

- **What is R?**
- R Data Types & Operator
- R Matrix Tutorial: Create, Print, Add Column, Slice
- What is Factor in R? Categorical & Continuous
- R Data Frames: Create, Append, Select, Subset
- Lists in R: Create, Select [Example]
- If, Else, Elif Statement in R
- For Loop Syntax and Examples
- While Loop In R With Example
- Apply(), Sapply(), Tapply() in R With Examples
- Import Data into R: Read Csv, Excel, SPSS, STATA, SAS Files
- R Exporting Data to Csv, Excel, STATA, SAS and Text File
- R Select(), Filter(), Arrange():

What is R?

- Data analysis with R is done in a series of steps;
  - Programming
  - Transforming,
  - Discovering,
  - Modeling and communicate the results.

Cont..,

- Program: R is a clear and accessible programming tool
- Transform: R is made up of a collection of libraries designed specifically for data science
- Discover: Investigate the data, refine your hypothesis and analyze them
- Model: R provides a wide array of tools to capture the right model for your data
- Communicate: Integrate codes, graphs, and outputs to a report with R Markdown or build Shiny apps to share with the world

What is R used for?

- Statistical inference
- Data analysis
- Machine learning algorithm

R – PROGRAMMING INTRODUCTION AND BASICS

- What is R?
- **R Data Types & Operator**
- R Matrix Tutorial: Create, Print, Add Column, Slice
- What is Factor in R? Categorical & Continuous
- R Data Frames: Create, Append, Select, Subset
- Lists in R: Create, Select [Example]
- If, Else, Elif Statement in R
- For Loop Syntax and Examples
- While Loop In R With Example
- Apply(), Sapply(), Tapply() in R With Examples
- Import Data into R: Read Csv, Excel, SPSS, STATA, SAS Files
- R Exporting Data to Csv, Excel, STATA, SAS and Text File
- R Select(), Filter(), Arrange():

R DATA TYPES & OPERATOR

Data Types	R Code	Output
# Numeric	x <- 28 class(x)	## [1] "numeric"
# String	y <- "R is Fantastic" class(y)	## [1] "character"
# Boolean	z <- TRUE class(z)	## [1] "logical"
#Complex number	cn <- 2 + 3i Class(cn)	##[1] “complex"

Cont..,

- The attribute of an object becomes important whenmanipulating objects. All objects have two attributes, the mode and their length.
- The R function mode can be used to determine the mode of each object, while the function length will help to determine each object’s length.

Cont..,

MODE OF OBJECT	OUTPUT	LENGTH OF OBJECT	OUTPUT
mode(value)	[1] "numeric"	length(value)	[1] 1
mode(string)	[1] "character"	length(string)	
mode(2<4)	[1] "logical"	-	-
mode(cn)	[1] "complex"	length(cn)	[1] 1
mode(sin)	[1] "function"	-	-
names(value)	[1] NULL	-	-

Variables

- Variables store values and are an important component in programming, especially for a data scientist.

SYNTAX	RCODE	OUTPUT

# First way to declare a variable: use the `<-` 	
name_of_variable <- value
# Second way to declare a variable: use the `=` 	
name_of_variable = value
	# Print variable x
	x <- 42
	x
		## [1] 42

	y <- 10 y	## [1] 10
	# We call x and y and apply a subtraction x-y	## [1] 32

Vectors

- A vector is a one-dimensional array.

Vectors	Type – Rcode	OUTPUT
# Numerical	vec_num <- c(1, 10, 49) vec_num	## [1] 1 10 49
# Character	vec_chr <- c("a", "b", "c") vec_chr	## [1] "a" "b" "c"
# Boolean	vec_bool <- c(TRUE, FALSE, TRUE) vec_bool	##[1] TRUE FALSE TRUE

# Create the vectors	vect_1 <- c(1, 3, 5) 	
	vect_2 <- c(2, 4, 6)
	# Take the sum of A_vector and B_vector
	sum_vect <- vect_1 + vect_2
	# Print out total_vector
	sum_vect
		[1] 3 7 11

# Slice the first 5 rows of the vector	slice_vector <- c(1,2,3,4,5,6,7,8,9,10) slice_vector[1:5]	## [1] 1 2 3 4 5
# Faster way to create adjacent values	c(1:10)	## [1] 1 2 3 4 5 6 7 8 9 10

Operator Description

Operator Description	RCode	OUTPUT
Arithmetic **+ Addition** **- Subtraction** *** Multiplication** **/ Division** **^ or ** Exponentiation**	3 + 4 4-2 3*5 (5+5)/2 2^5	## [1] 7 ## [1] 2 ## [1] 15 ## [1] 5 ## [1] 32
Logical	logical_vector <- c(1:10) logical_vector>5 logical_vector[(logical_vector>5)]	## [1]FALSE FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE ## [1] 6 7 8 9 10

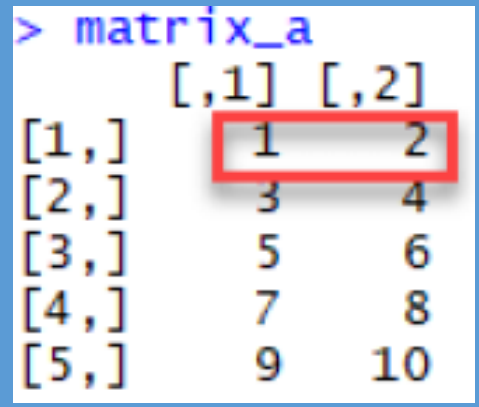
	# Print 5 and 6
	logical_vector <- c(1:10)
	logical_vector[(logical_vector>4) & (logical_vector<7)]
		## [1] 5 6

R – PROGRAMMING INTRODUCTION AND BASICS

- What is R?
- R Data Types & Operator
- **R Matrix Tutorial: Create, Print, Add Column, Slice**
- What is Factor in R? Categorical & Continuous
- R Data Frames: Create, Append, Select, Subset
- Lists in R: Create, Select [Example]
- If, Else, Elif Statement in R
- For Loop Syntax and Examples
- While Loop In R With Example
- Apply(), Sapply(), Tapply() in R With Examples
- Import Data into R: Read Csv, Excel, SPSS, STATA, SAS Files
- R Exporting Data to Csv, Excel, STATA, SAS and Text File
- R Select(), Filter(), Arrange():

What is a Matrix?

- A matrix is a 2-dimensional array that has m number of rows and n number of columns.
- **SYNTAX:** matrix(data, nrow, ncol, byrow = FALSE)

# Construct a matrix with 5 rows that contain the numbers 1 up to 10 and byrow = TRUE	**matrix_a <-matrix(1:10, byrow = TRUE, nrow = 5)** **matrix_a**	
# Print dimension of the matrix	dim(matrix_a)	## [1] 5 2

Slice a Matrix

- We can select elements one or many elements from a matrix by using the square brackets [ ].
- For example:
  - matrix_c[1,2] selects the element at the first row and second column.
  - matrix_c[1:3,2:3] results in a matrix with the data on the rows 1, 2, 3 and columns 2, 3,
  - matrix_c[,1] selects all elements of the first column.
  - matrix_c[1,] selects all elements of the first row.

R – PROGRAMMING INTRODUCTION AND BASICS

- What is R?
- R Data Types & Operator
- R Matrix Tutorial: Create, Print, Add Column, Slice
- **What is Factor in R? Categorical & Continuous**
- R Data Frames: Create, Append, Select, Subset
- Lists in R: Create, Select [Example]
- If, Else, Elif Statement in R
- For Loop Syntax and Examples
- While Loop In R With Example
- Apply(), Sapply(), Tapply() in R With Examples
- Import Data into R: Read Csv, Excel, SPSS, STATA, SAS Files
- R Exporting Data to Csv, Excel, STATA, SAS and Text File
- R Select(), Filter(), Arrange():

WHAT IS FACTOR IN R? CATEGORICAL & CONTINUOUS

- **What is Factor in R?**
  - Factors are variables in R which take on a limited number of different values; such variables are often referred to as categorical variables.
  - Two types of variables: categorical and continuous.
  - Syntax

```
factor(x = character(), levels, labels = levels, ordered = is.ordered(x))
Arguments:
- x: A vector of data. Need to be a string or integer, not decimal.
- Levels: A vector of possible values taken by x. This argument is optional. The default value
 is the unique list of items of the vector x.
- Labels: Add a label to the x data. For example, 1 can take the label `male` while 0, the labe
l `female`.
- ordered: Determine if the levels should be ordered.
```

RCODE:

```
# Create gender vector
• gender_vector <- c("Male", "Female", "Female", "Male", "Male")
• class(gender_vector)
# Convert gender_vector to a factor
• factor_gender_vector <-factor(gender_vector)
• class(factor_gender_vector)
```

OUTPUT:

```
## [1] "character"
## [1] "factor"
```

Nominal categorical variable

- A categorical variable has several values but the order does not matter. For instance, male or female categorical variable do not have ordering.

RCODE:

Create a color vector

- color_vector <- c('blue', 'red', 'green', 'white', 'black', 'yellow')

Convert the vector to factor

- factor_color <- factor(color_vector)

- factor_color

Cont..,

OUTPUT:

[1] blue red green white black yellow

Levels: black blue green red white yellow

Ordinal categorical variable

- Ordinal categorical variables do have a natural ordering. We can specify the order, from the lowest to the highest with order = TRUE and highest to lowest with order = FALSE. We can use summary to count the values for each factor.

Cont..,

Continuous variables

- Continuous class variables are the default value in R. They are stored as numeric or integer.
- **RCODE:**
  - dataset <- mtcars
  - class(dataset)

OUTPUT:

[1] "numeric"

R – PROGRAMMING INTRODUCTION AND BASICS

- What is R?
- R Data Types & Operator
- R Matrix Tutorial: Create, Print, Add Column, Slice
- What is Factor in R? Categorical & Continuous
- **R Data Frames: Create, Append, Select, Subset**
- Lists in R: Create, Select [Example]
- If, Else, Elif Statement in R
- For Loop Syntax and Examples
- While Loop In R With Example
- Apply(), Sapply(), Tapply() in R With Examples
- Import Data into R: Read Csv, Excel, SPSS, STATA, SAS Files
- R Exporting Data to Csv, Excel, STATA, SAS and Text File
- R Select(), Filter(), Arrange():

R DATA FRAMES: CREATE, APPEND, SELECT, SUBSET

What is a Data Frame?

- A data frame is a list of vectors which are of equal length. A matrix contains only one type of data, while a data frame accepts different data types (numeric, character, factor, etc.).
-

Create a data frame

```
# Create a, b, c, d variables
a <- c(10,20,30,40)
b <- c('book', 'pen',
'textbook', 'pencil_case')
df <- data.frame(a,b,c,d)
df
```

```
## a b
## 1 1 book
## 2 2 pen
## 3 3 textbook
## 4 4 pencil_case
# Name the data frame
names(df) <- c('ID', 'items', 'store',
'price')
df
```

Cont..,

Slice Data Frame

- It is possible to SLICE values of a Data Frame. We select the rows and columns to return into bracket precede by the name of the data frame.

RCODE:

- ## Select Rows 1 to 3 and columns 3 to 4
- df[1:3, 3:4]

OUTPUT:

- ## store price
- ## 1 TRUE 2.5
- ## 2 FALSE 8.0
- ## 3 TRUE 10.0

Cont..,

Append a Column to Data Frame

- You can also append a column to a Data Frame. You need to use the symbol $ to append a new Variable.

RCODE:

Create a new vector

- quantity <- c(10, 35, 40, 5)

Add `quantity` to the `df` data frame

- df$quantity <- quantity
- df

OUTPUT:

ID items store price quantity

1 10 book TRUE 2.5 10

2 20 pen FALSE 8.0 35

3 30 textbook TRUE 10.0 40

4 40 pencil_case FALSE 7.0 5

Cont..,

Select a column of a data frame

- Sometimes, we need to store a column of a data frame for future use or perform operation on a column. We can use the $ sign to select the column from a data frame.

RCODE:

```
# Select the column ID
```

- df$ID

OUTPUT:

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

Cont..,

Subset a data frame

- In the previous section, we selected an entire column without condition. It is possible to subset based on whether or not a certain condition was true.
- We use the subset() function.

SYNTAX:

```
subset(x, condition)
arguments:
- x: data frame used to perform the subset
- condition: define the conditional statement
```

Cont..,

RCODE:

Select price above 5

- subset(df, subset = price > 5)

OUTPUT:

- ID items store price
- 2 20 pen FALSE 8
- 3 30 textbook TRUE 10
- 4 40 pencil_case FALSE 7

R – PROGRAMMING INTRODUCTION AND BASICS

- What is R?
- R Data Types & Operator
- R Matrix Tutorial: Create, Print, Add Column, Slice
- What is Factor in R? Categorical & Continuous
- R Data Frames: Create, Append, Select, Subset
- **Lists in R: Create, Select [Example]**
- If, Else, Elif Statement in R
- For Loop Syntax and Examples
- While Loop In R With Example
- Apply(), Sapply(), Tapply() in R With Examples
- Import Data into R: Read Csv, Excel, SPSS, STATA, SAS Files
- R Exporting Data to Csv, Excel, STATA, SAS and Text File
- R Select(), Filter(), Arrange():

LISTS IN R: CREATE, SELECT [EXAMPLE]

What is a List?

- A list is a great tool to store many kinds of object in the order expected.

Vector with numeric from 1 up to 5

- vect <- 1:5

A 2x 5 matrix

- mat <- matrix(1:9, ncol = 5)
- dim(mat)

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

## [[2]]
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    3    5    7    9
## [2,]    2    4    6    8    1
```

R – PROGRAMMING INTRODUCTION AND BASICS

- What is R?
- R Data Types & Operator
- R Matrix Tutorial: Create, Print, Add Column, Slice
- What is Factor in R? Categorical & Continuous
- R Data Frames: Create, Append, Select, Subset
- Lists in R: Create, Select [Example]
- **If, Else, Elif Statement in R**
- For Loop Syntax and Examples
- While Loop In R With Example
- Apply(), Sapply(), Tapply() in R With Examples
- Import Data into R: Read Csv, Excel, SPSS, STATA, SAS Files
- R Exporting Data to Csv, Excel, STATA, SAS and Text File
- R Select(), Filter(), Arrange():

IF, ELSE, ELIF STATEMENT IN R

The if, else, ELIF statement

- An if-else statement is a great tool for the developer trying to return an output based on a condition.
- **SYNTAX:**

```
if (condition1) {
expr1
} else it (condition2) {
expr2
} else if (condition3) {
expr3
} else {
expr4
}
```

Cont..,

RCODE:

```
category <- 'A'
price <- 10
if (category =='A'){
cat('A vat rate of 8% is applied.','The total price is',price *1.08)
} else if (category =='B'){
cat('A vat rate of 10% is applied.','The total price is',price *1.10)
} else {
cat('A vat rate of 20% is applied.','The total price is',price *1.20)
}
```

OUTPUT:

A vat rate of 8% is applied. The total price is 10.8

R – PROGRAMMING INTRODUCTION AND BASICS

- What is R?
- R Data Types & Operator
- R Matrix Tutorial: Create, Print, Add Column, Slice
- What is Factor in R? Categorical & Continuous
- R Data Frames: Create, Append, Select, Subset
- Lists in R: Create, Select [Example]
- If, Else, Elif Statement in R
- **For Loop Syntax and Examples**
- While Loop In R With Example
- Apply(), Sapply(), Tapply() in R With Examples
- Import Data into R: Read Csv, Excel, SPSS, STATA, SAS Files
- R Exporting Data to Csv, Excel, STATA, SAS and Text File
- R Select(), Filter(), Arrange():

For Loop Syntax and Examples

```
# Create fruit vector
        fruit <- c('Apple', 'Orange', 'Passion fruit', 'Banana')
# Create the for statement
    for ( i in fruit){
    print(i)
    }
```

OUTPUT:

```
## [1] "Apple"
## [1] "Orange"
## [1] "Passion fruit"
## [1] "Banana"
```

For Loop over a matrix

- A matrix has 2-dimension, rows and columns. To iterate over a matrix, we have to define two for loop, namely one for the rows and another for the column.
- **Syntax**

```
# Create a matrix
mat <- matrix(data = seq(10, 20, by=1), nrow = 6, ncol =2)
# Create the loop with r and c to iterate over the matrix
for (r in 1:nrow(mat))
for (c in 1:ncol(mat))
print(paste("Row", r, "and column",c, "have values of", mat[r,c]))
```

Cont..,

OUTPUT:

```
## [1] "Row 1 and column 1 have values of 10"
## [1] "Row 1 and column 2 have values of 16"
## [1] "Row 2 and column 1 have values of 11"
## [1] "Row 2 and column 2 have values of 17"
## [1] "Row 3 and column 1 have values of 12"
## [1] "Row 3 and column 2 have values of 18"
## [1] "Row 4 and column 1 have values of 13"
## [1] "Row 4 and column 2 have values of 19"
## [1] "Row 5 and column 1 have values of 14"
## [1] "Row 5 and column 2 have values of 20"
## [1] "Row 6 and column 1 have values of 15"
## [1] "Row 6 and column 2 have values of 10"
```

R – PROGRAMMING INTRODUCTION AND BASICS

- What is R?
- R Data Types & Operator
- R Matrix Tutorial: Create, Print, Add Column, Slice
- What is Factor in R? Categorical & Continuous
- R Data Frames: Create, Append, Select, Subset
- Lists in R: Create, Select [Example]
- If, Else, Elif Statement in R
- For Loop Syntax and Examples
- While Loop In R With Example
- Apply(), Sapply(), Tapply() in R With Examples
- Import Data into R: Read Csv, Excel, SPSS, STATA, SAS Files
- R Exporting Data to Csv, Excel, STATA, SAS and Text File
- R Select(), Filter(), Arrange():

WHILE LOOP IN R WITH EXAMPLE

- A loop is a statement that keeps running until a condition is satisfied. The syntax for a while loop is the following:

```
 #Create a variable with value 1
begin <- 1
#Create the loop
while (begin <= 10){
#See which we are
cat('This is loop number',begin)
#add 1 to the variable begin after each loop
begin <- begin+1
print(begin)
}
```

OUTPUT:

```
## This is loop number 1[1] 2
## This is loop number 2[1] 3
## This is loop number 3[1] 4
## This is loop number 4[1] 5
## This is loop number 5[1] 6
## This is loop number 6[1] 7
## This is loop number 7[1] 8
## This is loop number 8[1] 9
## This is loop number 9[1] 10
## This is loop number 10[1] 11
```

R – PROGRAMMING INTRODUCTION AND BASICS

- What is R?
- R Data Types & Operator
- R Matrix Tutorial: Create, Print, Add Column, Slice
- What is Factor in R? Categorical & Continuous
- R Data Frames: Create, Append, Select, Subset
- Lists in R: Create, Select [Example]
- If, Else, Elif Statement in R
- For Loop Syntax and Examples
- While Loop In R With Example
- **Apply(), Sapply(), Tapply() in R With Examples**
- Import Data into R: Read Csv, Excel, SPSS, STATA, SAS Files
- R Exporting Data to Csv, Excel, STATA, SAS and Text File
- R Select(), Filter(), Arrange():

APPLY(), SAPPLY(), TAPPLY() IN R WITH EXAMPLES

Function	Arguments	Objective	Input	Output
apply	apply(x, MARGIN, FUN)	Apply a function to the rows or columns or both	Data frame or matrix	vector, list, array
lapply	lapply(X, FUN)	Apply a function to all the elements of the input	List, vector or data frame	list
sapply	sappy(X FUN)	Apply a function to all the elements of the input	List, vector or data frame	vector or matrix

R – PROGRAMMING INTRODUCTION AND BASICS

- What is R?
- R Data Types & Operator
- R Matrix Tutorial: Create, Print, Add Column, Slice
- What is Factor in R? Categorical & Continuous
- R Data Frames: Create, Append, Select, Subset
- Lists in R: Create, Select [Example]
- If, Else, Elif Statement in R
- For Loop Syntax and Examples
- While Loop In R With Example
- Apply(), Sapply(), Tapply() in R With Examples
- **Import Data into R: Read Csv, Excel, SPSS, STATA, SAS Files**
- R Exporting Data to Csv, Excel, STATA, SAS and Text File
- R Select(), Filter(), Arrange():

IMPORT DATA INTO R: READ CSV, EXCEL, SPSS, STATA, SAS FILES

Library	Objective	Function	Default Arguments
utils	Read CSV file	read.csv()	file, header =,TRUE, sep = ","
readxl	Read EXCEL file	read_excel()	path, range = NULL, col_names = TRUE
haven	Read SAS file	read_sas()	path
haven	Read STATA file	read_stata()	path
haven	Read SPSS fille	read_sav()	path

R – PROGRAMMING INTRODUCTION AND BASICS

- What is R?
- R Data Types & Operator
- R Matrix Tutorial: Create, Print, Add Column, Slice
- What is Factor in R? Categorical & Continuous
- R Data Frames: Create, Append, Select, Subset
- Lists in R: Create, Select [Example]
- If, Else, Elif Statement in R
- For Loop Syntax and Examples
- While Loop In R With Example
- Apply(), Sapply(), Tapply() in R With Examples
- Import Data into R: Read Csv, Excel, SPSS, STATA, SAS Files
- **R Exporting Data to Csv, Excel, STATA, SAS and Text File**
- R Select(), Filter(), Arrange():

Export to Hard drive

- To begin with, you can save the data directly into the working directory. The following code
- prints the path of your working directory:

RCODE:

- directory <-getwd()
- directory

OUTPUT:

[1] "/Users/15_Export_to_do"

Export CSV

SYNTAX:

```
write.csv(df, path)
arguments
-df: Dataset to save. Need to be the same name of the data frame in the environment.
-path: A string. Set the destination path. Path + filename + extension i.e. "/Users/USERNAME/Do
wnloads/mydata.csv" or the filename + extension if the folder is the same as the working direct
ory
```

RCODE:

```
write.csv(df, "table_car.csv")
```

Export to Excel file

- Export data to Excel is trivial for Windows users and trickier for Mac OS user. Both users will use the library xlsx to create an Excel file. The slight difference comes from the installation of the library. Indeed, the library xlsx uses Java to create the file. Java needs to be installed if not present in your machine.
- **Windows users**
  - If you are a Windows user, you can install the library directly with conda:
  - conda install -c r r-xlsx
  - library(xlsx)
  - write.xlsx(df, "table_car.xlsx")
  - library(haven)
  - write_sav(df, "table_car.sav" ## spss file
  - write_sas(df, "table_car.sas7bdat")
  - write_dta(df, "table_car.dta") ## STATA File
  - save(df, file ='table_car.RData')

R – PROGRAMMING INTRODUCTION AND BASICS

- What is R?
- R Data Types & Operator
- R Matrix Tutorial: Create, Print, Add Column, Slice
- What is Factor in R? Categorical & Continuous
- R Data Frames: Create, Append, Select, Subset
- Lists in R: Create, Select [Example]
- If, Else, Elif Statement in R
- For Loop Syntax and Examples
- While Loop In R With Example
- Apply(), Sapply(), Tapply() in R With Examples
- Import Data into R: Read Csv, Excel, SPSS, STATA, SAS Files
- R Exporting Data to Csv, Excel, STATA, SAS and Text File
- **R Select(), Filter(), Arrange()**

R SELECT(), FILTER(), ARRANGE()

Verb	Objective	Code	Explanation
glimpse	check the structure of a df	glimpse(df)	Identical to str()
select()	Select/exclude the variables	select(df, A, B,C) select(df, A:C) select(df, -C)	Select the variables A, B and C Select all variables from A to C Exclude C
arrange()	Sort the dataset with one or many variables	arrange(A) arrange(desc (A), B)	Descending sort of variable A and ascending sort of B

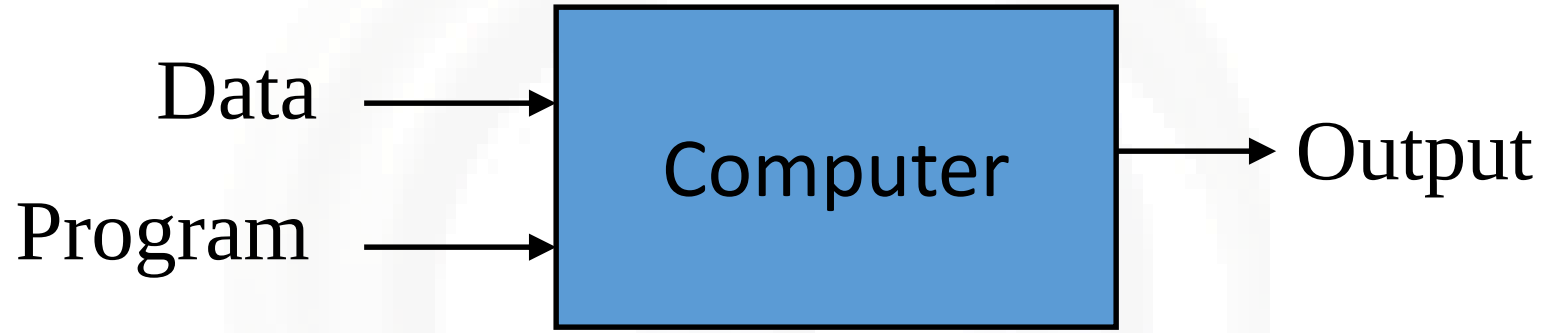
Data Mining

Machine Learning

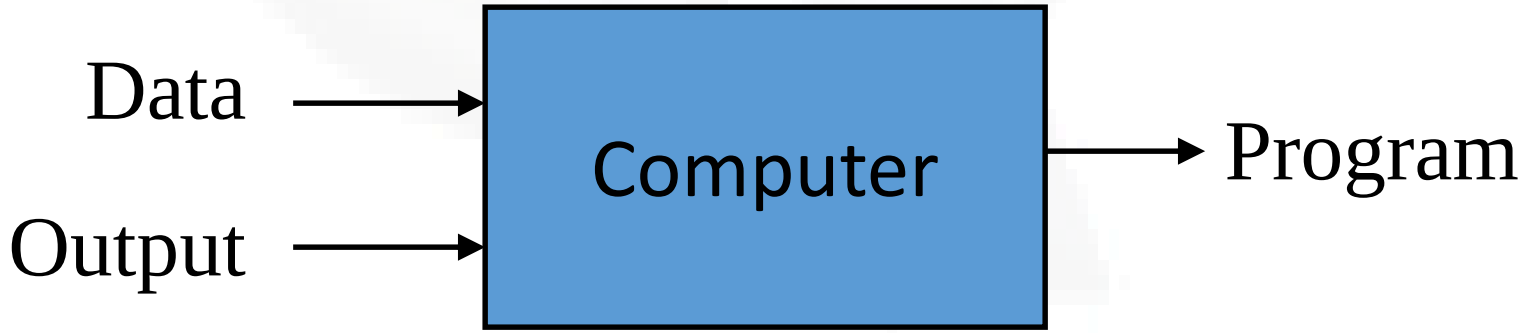
What Is Machine Learning?

- Automating automation
- Getting computers to program themselves
- Writing software is the bottleneck
- Let the data do the work instead!

Traditional Programming



Machine Learning



Magic?

No, more like gardening

- **Seeds** = Algorithms
- **Nutrients** = Data
- **Gardener** = You
- **Plants** = Programs

Sample Applications

- Web search
- Computational biology
- Finance
- E-commerce
- Space exploration
- Robotics
- Information extraction
- Social networks
- Debugging
- [Your favorite area]

Machine Learning

- Tens of thousands of machine learning algorithms
- Hundreds new every year
- Every machine learning algorithm has three components:
  - **Representation**
  - **Evaluation**
  - **Optimization**

Representation

- Decision trees
- Sets of rules / Logic programs
- Instances
- Graphical models (Bayes/Markov nets)
- Neural networks
- Support vector machines
- Model ensembles
- Etc.

Evaluation

- Accuracy
- Precision and recall
- Squared error
- Likelihood
- Posterior probability
- Cost / Utility
- Margin
- Entropy
- K-L divergence
- Etc.

Optimization

- Combinatorial optimization
  - E.g.: Greedy search
- Convex optimization
  - E.g.: Gradient descent
- Constrained optimization
  - E.g.: Linear programming

Types of Learning

- **Supervised (inductive) learning**
  - Training data includes desired outputs
- **Unsupervised learning**
  - Training data does not include desired outputs
- **Semi-supervised learning**
  - Training data includes a few desired outputs
- **Reinforcement learning**
  - Rewards from sequence of actions

Inductive Learning

- **Given** examples of a function *(X, F(X))*
- **Predict** function *F(X)* for new examples *X*
  - Discrete *F(X)*: Classification
  - Continuous *F(X)*: Regression
  - *F(X)* = Probability(*X*): Probability estimation

What We'll Cover

- **Supervised learning**
  - Decision tree induction
  - Rule induction
  - Instance-based learning
  - Bayesian learning
  - Neural networks
  - Support vector machines
  - Model ensembles
  - Learning theory
- **Unsupervised learning**
  - Clustering
  - Dimensionality reduction

ML in Practice

- Understanding domain, prior knowledge, and goals
- Data integration, selection, cleaning, pre-processing, etc.
- Learning models
- Interpreting results
- Consolidating and deploying discovered knowledge
- Loop

Statistical Analysis using - R

Phases of testing

- Select an appropriate statistical test
  - Compare means of groups?
  - Compare variances of groups?
  - Compare the distributions
  - Model the relationship between two variables?
- Calculate the test statistic and p-value
  - These are automated by the computer
- Draw conclusions
  - This is not automated by the computer!

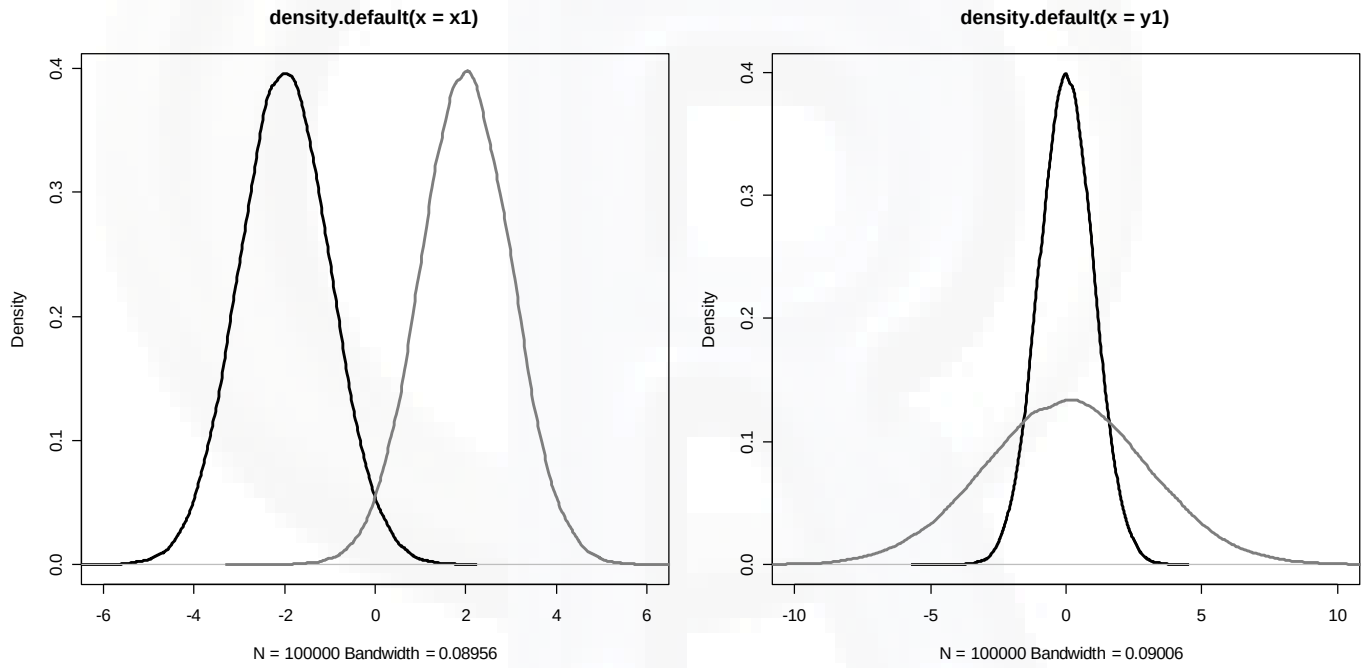
How to select the test?

- There are two types of test
  - Parameteric
    - Assume that the variable is normally distributed
  - Non-parametric
    - Does not assume that the variable is normally distributed
    - But, can make other restricting assumptions!
- Only parametric ones are used on this course

Cont..,

- What kind of hypothesis you want to test?
  - Is the prime interest in the difference in means?
    - Are men taller than women
  - Can the difference in variance be of interest?
    - Is the height of men more variable than the height of women?
  - Do you want to predict the variable with another variable
    - Can a persons height be predicted from shoesize?

Mean and variance, an example



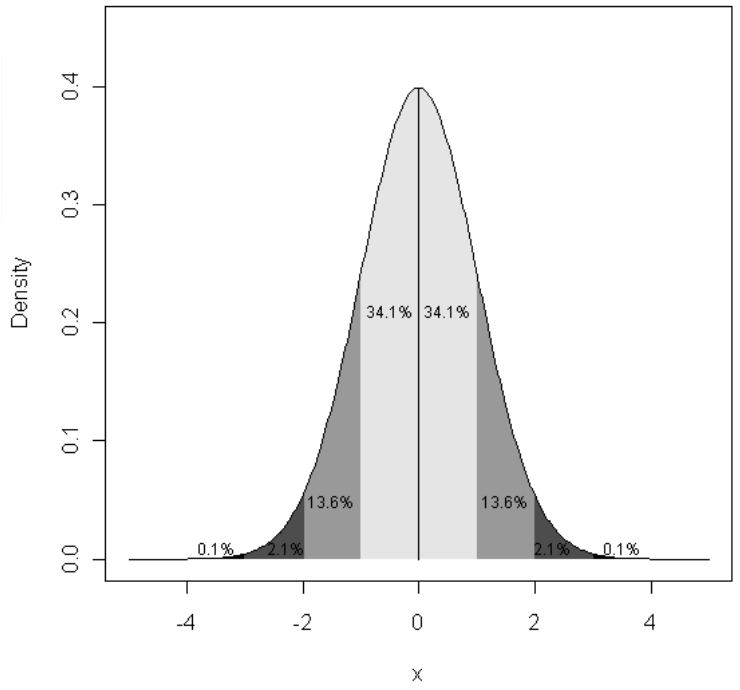
Different tests

- Compare the means of groups
  - **One Sample Test**
    - **One Sample T-Test**
  - Two Sample Test
    - Two Sample T-Test
    - Paired T-Test
    - ANOVA
- Compare the variability of groups
  - F-test
- Compare the distribution of categorical variables
  - Chi Square
- Predict a variable with another variable
  - Correlation
  - (Linear) Regression

One-sample t-test I/XI

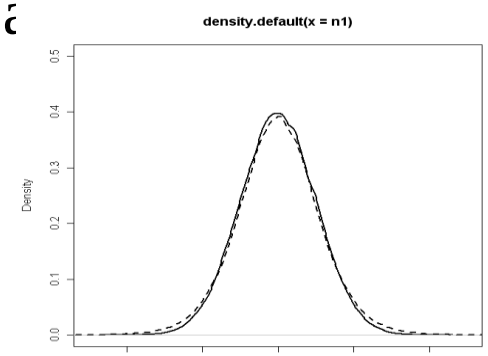
- Comparison of the mean of the data against some known value of group mean.
  - Is the mean of height of the sampled students different from the population mean (we known the population mean to be 167 cm)?
- This simple test will act a primer to all other tests, since deep down they have the same principles:
  - Calculate a test statistic (here, T)
  - Calculate the degrees of freedom
  - Compare the test statistic to a distribution (here, T)
  - Get the p-value

Normal distribution I/III



One-sample t-test II/XI

- The idea behind the t-test is the following
  - Transform the variable of interest to follow a t-distribution.
    - T-distribution is very similar to a normal distribution, but with a small degrees of freedom it's tails are fatter.
    - Degrees of freedom is the parameter that defines the shape of the t-distribution.
  - Compare the calculated t statistic to standard t distribution with the certain degrees of freedom.
  - If the test statistic falls in the area where less than 5% of the values in the standard distribution a nificant with p-value of 0.05.



One-sample t-test III/XI

- What are degrees of freedom?
  - Assume we know three values (1,2,3) and the mean of the values (2).
  - To calculate the degrees of freedom, we have to think how many of those values can we erase, and still be able to say what it was. Note that we have the mean to help as here.
  - In this case, one can erase one of the values, and still be able to say what it was.
    - If we erase number 1, we have to values (2,3) left. Since we know the mean (2), we can say with confidence that the one value that was removed was 1.
    - The same goes for all other values as well.
  - Since one value could be erased, we say that the degrees of freedom is 2 (equal to the number of observations left).

One-sample t-test IV/XI

- So, how do we get the test statistic then?
  - Say we have five observations of height (160, 170, 172, 174, 181)
  - The mean height of population is 167
  - We first calculate a mean of the observations, that's 171.4
  - Then we calculate the standard deviation, that's 7.6
  - Last, we plug these into the formula:

$$T = \frac{M - \mu}{\frac{s}{\sqrt{n}}},$$

- Using the numbers we just calculated that becomes:
  - T = (171.4 – 167) / (7.6 / 2.24) = 1.30
- Last, the value of T is compared to a table of critical values, where we can see, that T = 1.30 with df = 5-1 = 4 is not statistically significant
  - We don't use a table here, but R (see the next slide)

One-sample t-test V/XI -R

- > height<-c(160, 170, 172, 174, 180)
- > t.test(height, mu=167)

One Sample t-test

- data: s
  - t = 1.2941, df = 4, p-value = 0.2653
  - alternative hypothesis: true mean is not equal to 167
  - 95 percent confidence interval:
  - 161.9601 180.8399
  - sample estimates:
  - mean of x
  - 171.4

One-sample t-test VI/XI

- What is that p-value anyway?
  - P-value is a risk of saying that there is a difference between the groups means when there actually isn't.
  - So, if there is a difference in heights, the p-value should be small, and there is not any difference, then it should be high.
  - Traditionally p-values were coded with three stars:
    - 0.05 *
    - 0.01 **
    - 0.001 ***
  - Nowadays it's more customary to report the p-value as such.
- How to interpret the p-value?
  - If the p-value is less than 0.05 then the test usually said to be statistically significant.
    - This cut-off is made from the top of one's head, but it is often used, purely on traditional basis.

One-sample t-test VII/XI

- What happens, if the difference remains at the same level, but we add more observations?
  - With 10 observations:
    - t = 3.6565, df = 9, p-value = 0.005264
  - With 20 observations:
    - t = 2.474, df = 19, p-value = 0.02296
  - With 100 observations:
    - t = 6.6407, df = 99, p-value = 1.696e-09

One-sample t-test VIII/XI

- Pay attention to the degrees of freedom!
- One Sample t-test
  - data: s
  - t = 1.2941, df = 4, p-value = 0.2653
  - alternative hypothesis: true mean is not equal to 167
  - 95 percent confidence interval:
  - 161.9601 180.8399
  - sample estimates:
  - mean of x
  - 171.4
- Here we had 5 observations, so the degrees of freedom should be 4, as they are.
  - If they weren't, then something went wrong, and you should check your procedures.

One-sample t-test IX/XI

- What about the confidence interval?
- One Sample t-test
  - data: s
  - t = 1.2941, df = 4, p-value = 0.2653
  - alternative hypothesis: true mean is not equal to 167
  - 95 percent confidence interval:
  - 161.9601 180.8399
  - sample estimates:
  - mean of x
  - 171.4
- Confidence intervals gives a range of values. The true mean estimated from the sample is in this range with 95% probability.
  - If you sample the same population again 100 times, the true mean should fall into this range about 95% of the time.

One-sample t-test X/XI

- How do you calculate a confidence interval?
  - We use normal distribution (or t-distribution) for calculations.
  - Using the normal distribution, 95% of the values are in the range of +/- 1.96 standard deviations from the mean.
  - Since we want the estimate of the mean to be in this range, we use that
  - 1.96 for calculations.

Cont..,

- First calculate a standard error (standard deviation of the estimated mean)
  - SE = SD / sqrt(n) = 7.6 / 3.4 = 2.235
- The positive confidence interval is then
  - mean + 1.96*SE = 171.4 + 1.96 * 2.235 = 177.61
- And the negative confidence interval is
  - mean - 1.96*SE = 171.4 - 1.96 * 2.235 = 167.01
- These values are not equal to the ones given in the t-test output from R. The reason is that these were calculated in a slightly different way (using normal distribution instead of t distribution)

One-sample t-test XI/XI

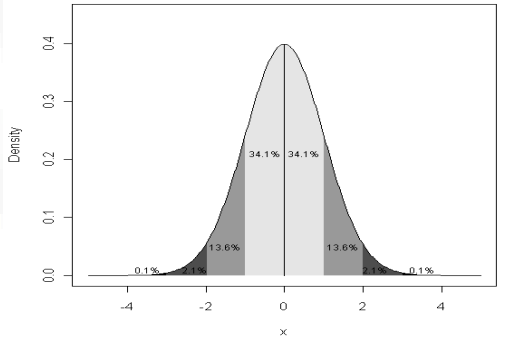
- Calculating the correct confidence interval by hand in R using the t distribution
  - First check the correct quantile from the t-distribution
    - Two-tailed test, so should 0.975
    - qt(0.975, df=4) # 2.776445
  - The calculate the standard error
    - sd(height) / sqrt(5) # 3.4

Cont..,

- Calculate the positive confidence interval
  - 171.4 + 2.776445 * 3.4 # 180.8399
- Calculate the negative confidence interval
- 171.4 - 2.776445 * 3.4 # 161.9601
- Now these are the same values as output by t.test( ) in R.
- Note that the confidence intervals calculated on the basis of t- distribution are slightly wider than those based on the normal distribution.
  - That’s how it should be.

Normal or t distribution

- Two-tailed test:
  - Both ends taken into account (5% of the values are in both ends)
  - In the two-tailed test, the cut-off point for quantile from the distribution is 0.975
- One-tailed test:
  - Only one end taken into account
  - The quantile is 0.95.



Different tests

- Compare the means of groups
  - One Sample Test
    - One Sample T-Test
  - **Two Sample Test**
    - **Two Sample T-Test**
    - Paired T-Test
    - ANOVA
- Compare the variability of groups
  - F-test
- Compare the distribution of categorical variables
  - Chi Square
- Predict a variable with another variable
  - Correlation
  - (Linear) Regression

Two-sample t-test

- Two-sample t-test compares means of two groups.
- The idea is the same as in the one-sample t-test.
  - First we calculate the difference in group means.
  - Then we divide it by the standard error.
    - There are different ways to estimate the standard error depending on whether the variances in the groups can be assumed to be equal or unequal.
  - Thus, we get the test statistic, and we conclude testing as with one-sample t-test.

Two-sample t-test in R

- > x<-rnorm(10, mean=0, sd=1)
- > y<-rnorm(10, mean=3, sd=1)
- > t.test(x, y)
- Welch Two Sample t-test
- data: x and y
- t = -10.7387, df = 17.753, p-value = 3.416e-09
- alternative hypothesis: true difference in means is not equal to 0
- 95 percent confidence interval:
- -4.217709 -2.836288
- sample estimates:
- mean of x mean of y
- -0.3181124 3.2088861

Cont..,

- > t.test(x, y, var.equal=T)
- Two Sample t-test
- data: x and y
- t = -10.7387, df = 18, p-value = 2.95e-09
- alternative hypothesis: true difference in means is not equal to 0
- 95 percent confidence interval:
- -4.217021 -2.836976
- sample estimates:
- mean of x mean of y
- -0.3181124 3.2088861

Note on degreed of freedom

- Note that in the two-sample test assuming equal variances, the degrees of freedom are calculated as a sum of
  - Number of observation in group A -1
  - Number of observation in group B -1
- So the df should always two less than the number of observations in the whole data set.

Different tests

- Compare the means of groups
  - One Sample Test
    - One Sample T-Test
  - **Two Sample Test**
    - Two Sample T-Test
    - **Paired T-Test**
    - ANOVA
- Compare the variability of groups
  - F-test
- Compare the distribution of categorical variables
  - Chi Square
- Predict a variable with another variable
  - Correlation

Paired t-test

- Paired t-test is applied in situations where there is a paired setting.
  - The samples were measured before and after some treatment.
- The demo data Hygrometer contains paired data
  - There are two observations per every hygrometer.
  - Each one of them was read before and after a longer rainy period.
  - Note that after preprocessing done on the first day, the data are now in two different columns is R. The order of the hygrometers is exactly the same in both columns, otherwise the pairing would be meaningless.

Cont..,

- Paired t-test equal running a one-sample t-test on the differences between the two observations.
  - Subtract the observation for hygrometer 1 on day 1 from the observation for hygrometer 1 on day 2.
  - Do this for all hygrometer, and run a one-sample t-test on these differences.

Paired t-test in R

- > x<-rnorm(10, mean=10, sd=1)
- > y<-x+rnorm(10, mean=0, sd=1)
- > t.test(x, y, paired=T)

Paired t-test

- data: x and y
- t = 0.5283, df = 9, p-value = 0.6101
- alternative hypothesis: true difference in means is not equal to 0
- 95 percent confidence interval:
- -0.5609109 0.9026993

Cont..,

- sample estimates:
- mean of the differences
- 0.1708942

Running the paired t-test

- > dif<-x-y
- > t.test(dif, mu=0)
- One Sample t-test
- data: dif
- t = 0.5283, df = 9, p-value = 0.6101
- alternative hypothesis: true mean is not equal to 0
- 95 percent confidence interval:
- -0.5609109 0.9026993
- sample estimates:
- mean of x
- 0.1708942


Thank
you