PROG8430 – Data Analysis, Modeling and Algorithms

LECTURE 2 – INTRODUCTION TO ‘R’

Introduction to R

SOME MATERIAL BORROWED FROM MATTHEW KELLER, HUNG CHEN, R IN A NUTSHELL, THE R COOKBOOK AND DR. SMRUTI R. SARANGI AND MS. HAMEEDAH SULTAN

R project - Background

”R is a free software environment for statistical computing and graphics” (http://www.r-project.org)

R consists of a core and packages.

Packages contain functions that are not available in the core.

Versions of R exist of Windows, MacOS, Linux and various other variations on Unix.

R was originally written by Ross Ihaka and Robert Gentleman, at the University of Auckland and is an implementation of the S language, which was principally developed by John Chambers

Well, yes, but what *is* R?

R is “GNU S” — A language and environment for data manipulation, calculation and graphical display.

- R is similar to the award-winning S system, which was developed at Bell Laboratories by John Chambers et al.
- a suite of operators for calculations on arrays, in particular matrices,
- a large, coherent, integrated collection of intermediate tools for interactive data analysis,
- graphical facilities for data analysis and display either directly at the computer or on hardcopy
- a well developed programming language which includes conditionals, loops, user defined recursive functions and input and output facilities.

The Core of R is a language

The core of R is an interpreted computer language.

- Supports object-oriented and functional programming.
- It allows branching and looping as well as modular programming using functions.
- Most of the user-visible functions in R are written in R, calling upon a smaller set of internal primitives.
- It is possible for the user to interface to procedures written in C, C++ or FORTRAN languages for efficiency, and also to write additional primitives.

What does it do and what does it not do?

- data handling and storage: numeric, textual
- matrix algebra
- hash tables and regular expressions
- high-level data analytic and statistical functions
- classes (“OO”)
- graphics
- programming language: loops, branching, subroutines

- is not a database, but connects to DBMSs
- has no graphical user interfaces, but connects to Java, TclTk
- language interpreter can be very slow, but allows to call own C/C++ code
- no spreadsheet view of data, but connects to Excel/MsOffice
- no professional / commercial support

R and IDEs

There are many IDEs (Integrated Development Environments) available for R. Including:

- Emacs
- Eclipse/Architect
- Revolution-R
- Live-R
- RStudio

In this course we will be using RStudio

R Provides *LOTS of* Help

Once **R** is installed, there is a comprehensive built-in help system. At the program's command prompt you can use any of the following:

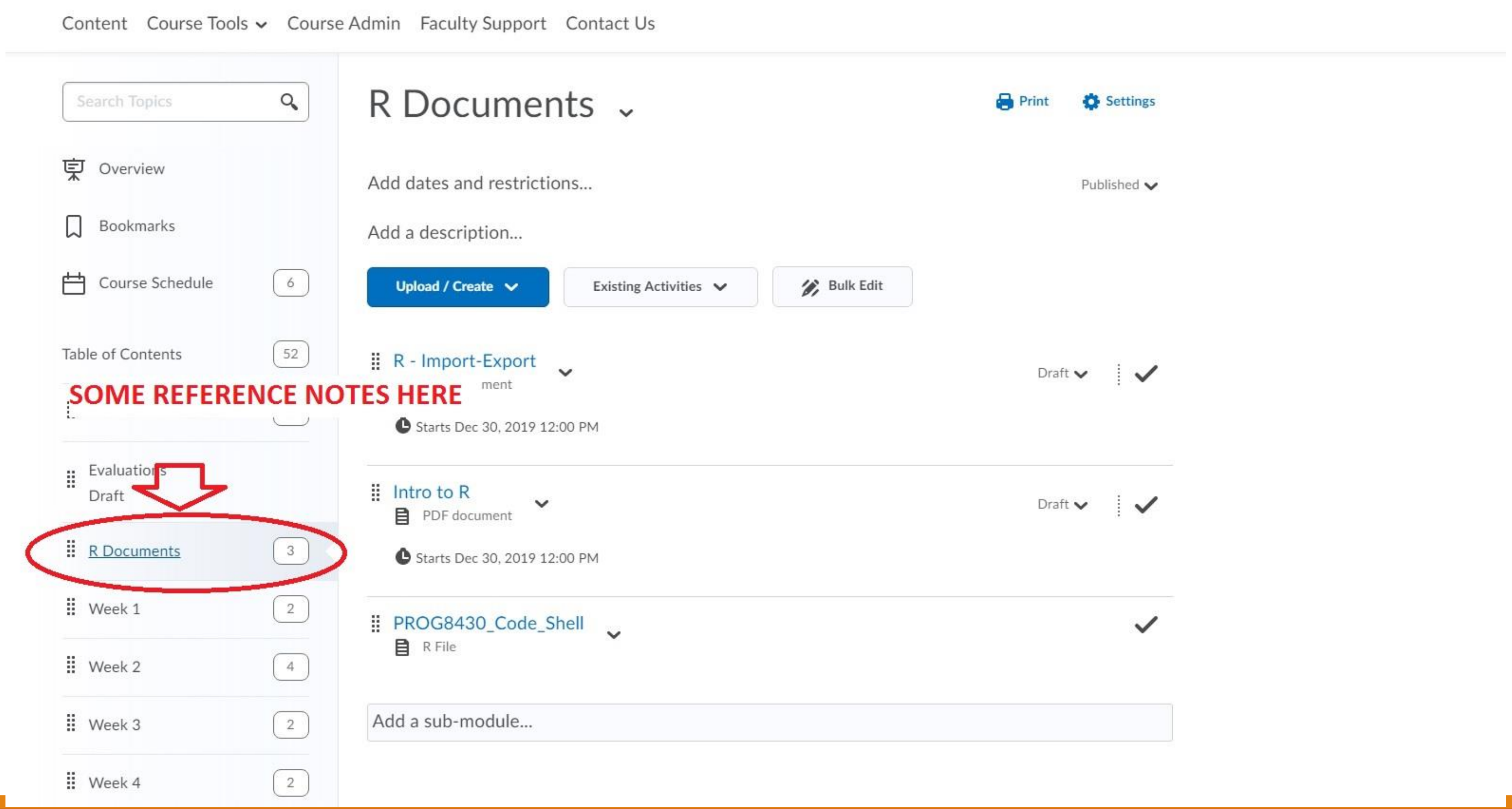
```
help.start()          # general help
help(max)             # help about function max
?max                  # same thing
apropos("max")   # list all function containing string max
example(max)          # show an example of function max

RSiteSearch("max")    # search for max in help manuals and archived mailing lists
```

FAQ is on:

https://cran.r-project.org/bin/windows/base/rw-FAQ.html

NOTE – PROG8430 is not a course on R programming specifically. I will show you examples, but you will need to read the help files if you want a *deep* understanding.

Also, some links for reference are posted on eConestoga



Sample Datasets

R comes with a number of sample datasets that you can experiment with.

Type *data( )* to see the available datasets. The results will depend on which packages you have loaded.

Type *help(datasetname)* for details on a sample dataset.

R Packages

An R ‘package’ contains specialized functions and may also contain other R objects, for example data sets or documentation.

When you download R, already a number (around 30) of packages are downloaded as well.

To use a function in an R package, that package has to be attached to the system.

When you start R not all of the downloaded packages are attached, only seven packages are attached to the system by default.

Use the function search to see a list of packages that are currently attached to the system, this list is also called the search path.

search()

[1] ".GlobalEnv" "package:stats" "package:graphics"

[4] "package:grDevices" "package:datasets" "package:utils"

[7] "package:methods" "Autoloads" "package:base"

R Packages

To attach another package to the system you can use the menu or the library function.

The function library can also be used to list all the available libraries on your system with a short description. Run the function without any arguments

library()

Packages in library 'C:/PROGRA~1/R/R-25~1.0/library':

base	The R Base Package
Boot	Bootstrap R (S-Plus) Functions (Canty)
class	Functions for Classification
cluster	Cluster Analysis Extended Rousseeuw et al.

NOTE – There are literally thousands of packages. For most assignments I will specify which packages you can use. You are better to learn one package well than several poorly.

Quick Tutorial

Introducing the command line
R is Interactive

```
> 1+2+3
[1] 6
> 1+2*3
[1] 7
> (1+2)*3
[1] 9
```

- Automatically prints an object returned by an expression entered into the R console.
- Any number entered is interpreted as a vector (an ordered collection of numbers).
- The "[1]" means that the index of the first item displayed in the row is 1. In each of these cases, there is also only one element in the vector

- Some R commands may take a long time to run. You can cancel a command once it has begun by typing ctrl + c. Note that it may also take R a long time to cancel the command.

Data Types in R

Most Frequently Used Data Types in R And Some Common Examples

Vector	Data Frame
• Logical (e.g. TRUE or FALSE • Numeric (e.g. 2.7182818) • Character (E.g.”True”, ‘PROG8430’)	• A two-dimensional array or table. • Rectangular data form of variables and values. • Data type primarily used in analysis.

Numbers become vectors

```
> c(0, 1, 1, 2, 3, 5, 8)
[1] 0 1 1 2 3 5 8
> 1:50
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
[24] 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46
[47] 47 48 49 50
```

- Make vectors directly with the "Combine" function (or 'c')
- 1:50 yields all natural numbers between 1 and 50

Functions

Functions are the backbone of R

In R, the operations that do all of the work are called *functions.* Most functions are in the following form:

f(argument1, argument2, ...)

Where f is the name of the function, and argument1, argument2, . . . are the arguments to the function.

```
> exp(1)     #exponential from e

[1] 2.718282

> cos(3.141593)   #cosine of pi (NOTE – in radians)

[1] -1

> log2(1)

[1] 0
```

Multi-Argument Functions

Many functions require more than one argument. You can specify the arguments by name:

```
> log(x=64, base=4)
[1] 3
```

Or, if you give the arguments in the default order, you can omit the names:

```
> log(64,4)
[1] 3
```

Some functions are simply operators. For example, we used the addition operator ("+") above.

```
> 17 + 2
[1] 19
> 2 ^ 10
[1] 1024
> 3 == 4
[1] FALSE
```

Variables

More Variable Exercises

```
> b <- c(1,2,3,4,5,6,7,8,9,10,11,12)
> b
 [1]  1  2  3  4  5  6  7  8  9 10 11 12
> # let's fetch the 7th item in vector b
> b[7]
[1] 7
> # fetch items 1 through 6
> b[1:6]
[1] 1 2 3 4 5 6
> # fetch only members of b that are congruent to zero (mod 3)
> # (in non-math speak, members that are multiples of 3)
> b[b %% 3 == 0]
[1]  3  6  9 12
```

```
> # fetch items 1 through 6
> b[1:6]
[1] 1 2 3 4 5 6
> # fetch 1, 6, 11
> b[c(1,6,11)]
[1]  1  6 11
> b[c(8,4,9)]
[1] 8 4 9
> b %% 3 == 0
 [1] FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE
[12]  TRUE
> b[b %% 3 == 0]
[1]  3  6  9 12
```

Running Scripts

Creating, Storing and Running Scripts

Creating a Script

- You can create a draft of your code as you go by using an R script. An R script is just a plain text file that you save R code in. You can create an R script by going to File > New File > R script in the menu bar. R will then open a fresh script window.

Saving a Script

- You should write and edit all of your R code in a script before you run it in the console because creates a reproducible record of your work.
- To save a script, click the scripts pane, and then go to File > Save As in the menu bar.

Running a Script

- You can automatically execute a line of code in a script by clicking the Run button. R will run whichever section is highlighted.
- You can run the entire script by clicking the Source button. Don’t like clicking buttons? You can use Control + Return as a shortcut for the Run button.

For PROG8430 – All Your R Scripts Should follow the pattern given in PROG8430_Code_Shell.R

```
#########################################################
### PROG8430                                 ##
#########################################################
# Code Shell for Course                      ##
# When submitting assignments, the title should  ##
# be here                                    ##
#########################################################
# Written by David Marsh
# ID: 8643279
#
#########################################################
### Basic Set Up                             ##
#########################################################
```

Import/Export

Reading Data from Files

Reads in a data frame from a file

Steps:

- Store the data frame in a file
- Read it in
  - > df <- read.table (“<filename>”)

Access the data frame

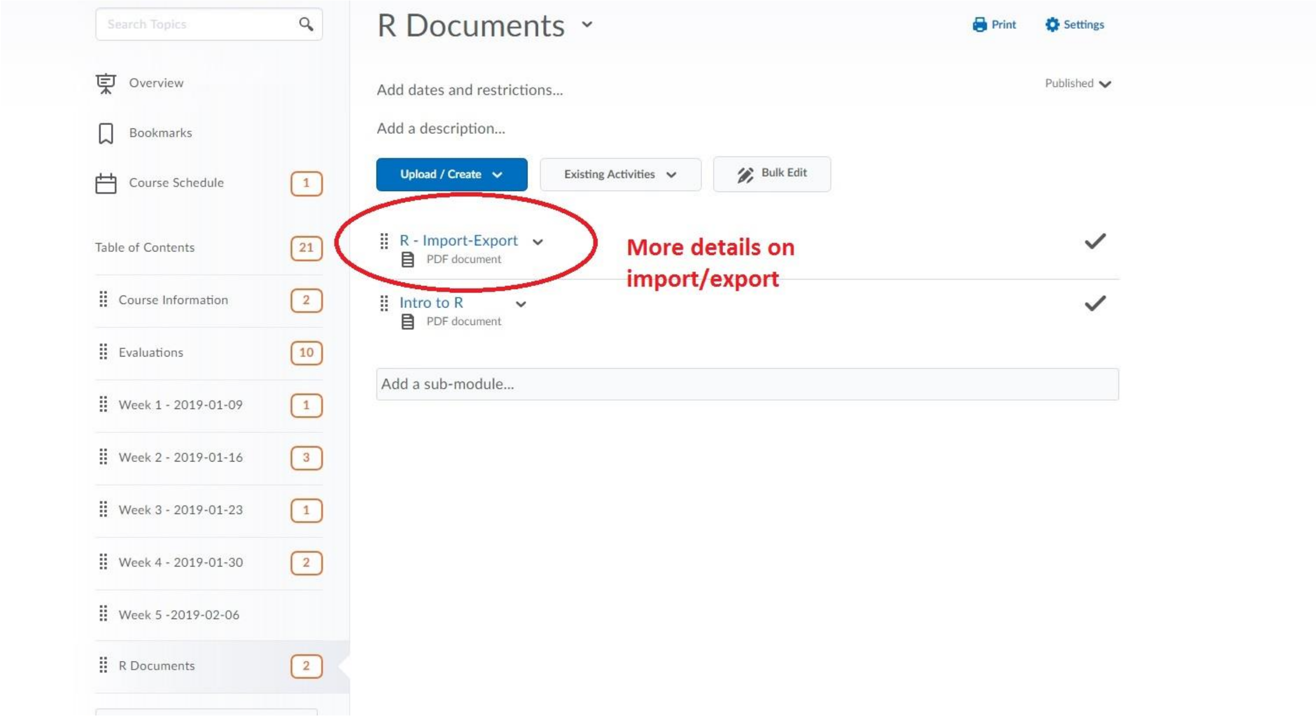
Review the file

There are many ways to import/export data in R

We will be focussing on basic, generic functions, but there are many more available.

The R-Project web page, as always, is a source of information.

The course website also contains an overview file on Import/Export functionality.



Importing datasets in to R using ‘read’ function

```
# Read "comma separated value" files (".csv")
# Record of Car Sales
Cars <- read.csv("PROG8430_Car_Sale.csv", header = TRUE, sep = ",")
names(Cars) <- c("Dlr","Model","Sold")
str(Cars)
```

- All of the examples that follow are from PROG8430_Demo_Read_Summarize.R in eConestoga
- ‘read.*’ is part of the core and basic packages installed with ‘R’.
- Note the “sep=“ command identifying separators.
- Names() can rename columns or show names (often they are awkward names
- Str() gives structure of columns
- NOTE – File address can also be a web address.

‘Read.*’ is flexible and can handle different file formats

```
# Read "tab delimited" files (".txt")
# Dealership Details
Dlr <- read.delim("PROG8430_Car_Dlr.txt", header = TRUE, sep = "\t")
names(Dlr) <- c("Dlr","Emp","Year","Bldg","Mgrs")
str(Dlr)
Dlr <- Dlr[c(1:3,5)]   # keep column 1-3 and 5
str(Dlr)
Dlr <- Dlr[-c(4)]      # drop column 4
str(Dlr)
```

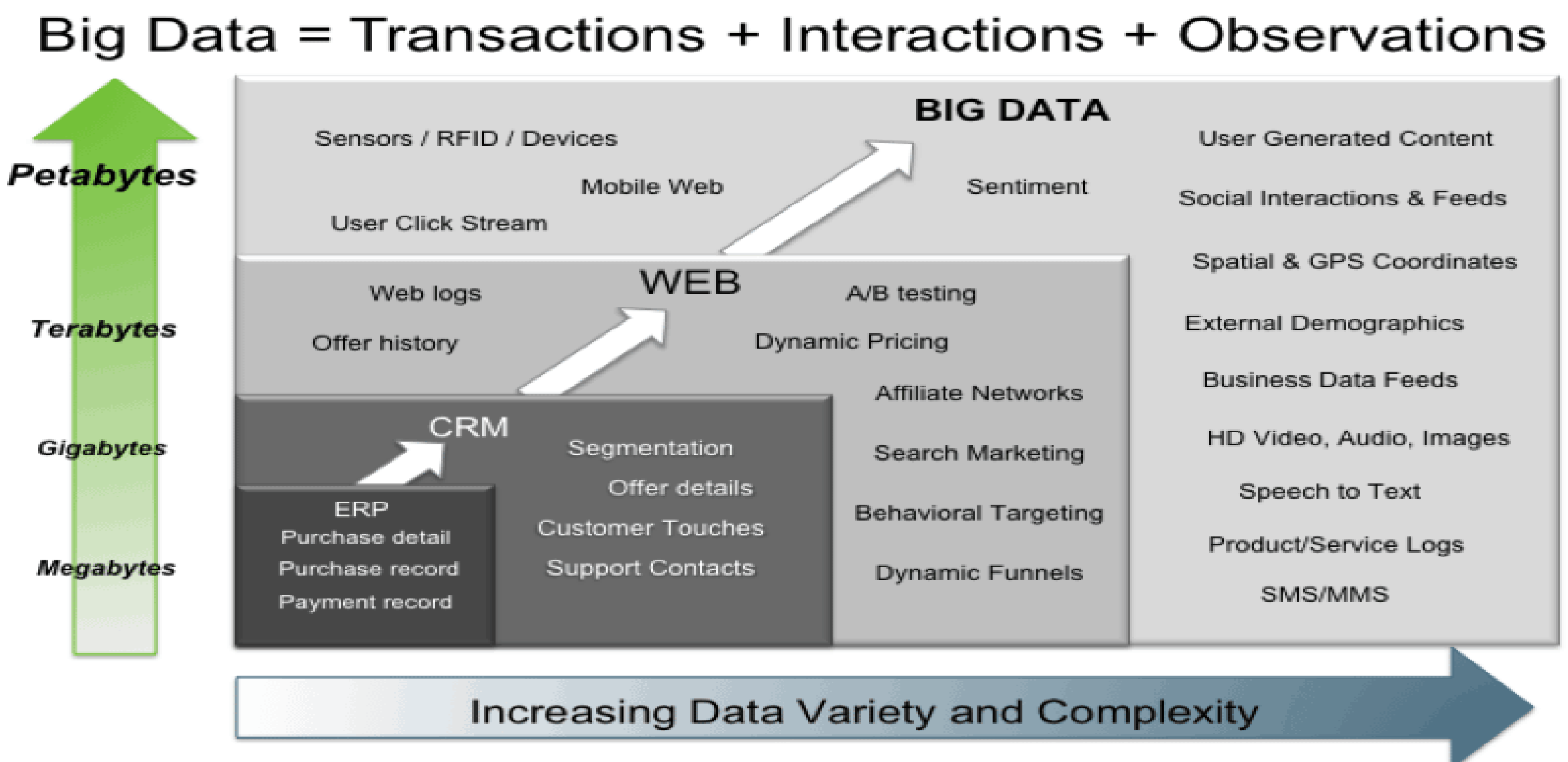
- colnames() can rename individual columns
- [c(1,2,)] identified columns and can keep or drop columns
- Can also keep or drop individually

Name	Description
Dlr	Dealer Number
Emp	Number of Employees
Year	Year the building was built
Bldg	Height of the building
Mgr	Name of the Manager

Data Structures for Data Analysis

YES, WE NEED STRUCTURE.

Big Data has a Variety of data sources, timelines and types.



Source: *Contents of above graphic created in partnership with Teradata, Inc.*

BUT! Most data analysis techniques require structured data

Non-Rectangular Data Structures

Type	Key Features	Key Uses
Time Series	• Repeated, successive measures of the same variable	• Statistical forecasting methods • Typical data stream from IoT
Spatial Data Structures	• *Object* representation – focus is an object (e.g. Waterloo campus) and it's spatial co-ordinates • *Field* representation – focuses on a small unit of space and *one* metric (e.g. pixel brightness)	• Mapping analysis • Location analysis

Graph/ Network Data Structures	• Represents physical, social and abstract relationships
	    • E.g. graph of a social network shows connections between people and the network
	    • E.g. Distribution hubs showing roads (minimum spanning trees)
		• Network optimization (transportation, logistics, sports team formations).
		• Recommender systems

Rectangular Data Structures

The most common and typical frame of reference for data analysis

A 2 dimensional matrix of records (*rows*) and features (*variables*). Essentially a spreadsheet or database table.

Data from unstructured sources will need to be extracted, processed and manipulated to form this structure

Data from relational databases must be extracted and put in to a single table for most data analysis

Rectangular Data Glossary (because it combines CS and Stats!)

Term	**Definition**	**Synonyms**
Data Frame	• Rectangular data (often indexed) which is the basic structure of data science, statistical and machine learning models.	
Feature	• A column is often called a feature	• Attribute, input, predictor, variable, predictors
Outcome	• Many projects involve predicting or prescribing an outcome (yes/no; estimated response, etc.)	• Dependant variable, response, target, output,
Records	• A row in the table	• Case, example, instance, observation, pattern, sample

Types of Data

Summary of Variables Types

Categorical Variables

- Categorical Variables
- Dichotomous Variables
- Ordinal Variables

Numerical Variables

- Discrete
- Continuous
  - Interval Variables
  - Ratio Variables

Summary of Categorical Feature Types

Type	Definition	Examples	Synonyms
Categorical	• Can only take on a specific set or values or categories but have no intrinsic order	• Province of residence • Hair colour	• Factors, nominal, enumerated
Dichotomous	• Only two possible values	• Enrolled in school or not. • Own an IPhone or not.	• Binary, logical, indicator, Boolean
Ordinal	• Has a specific ordering	• Likert scale ("disagree", "no opinion", "agree" • Moh's 'hardness' scale	• Ordered factor

Adapted from Bruce and Bruce, 2017, Wiley Academic

Summary of Numerical Feature Types

Type	**Definition**	**Examples**	**Synonyms**
Discrete	• Can only take on integer values	• Counts	• Integer, counts
Continuous	• Can take on any value in a domain		• Float, numeric
Interval	• Measured on a continuum (often scaled in a linear fashion)	• Temperature	
Ratio	• Like interval, but 0 represents “no value”	• Height, mass, distance	

Summary of Numerical Feature Types

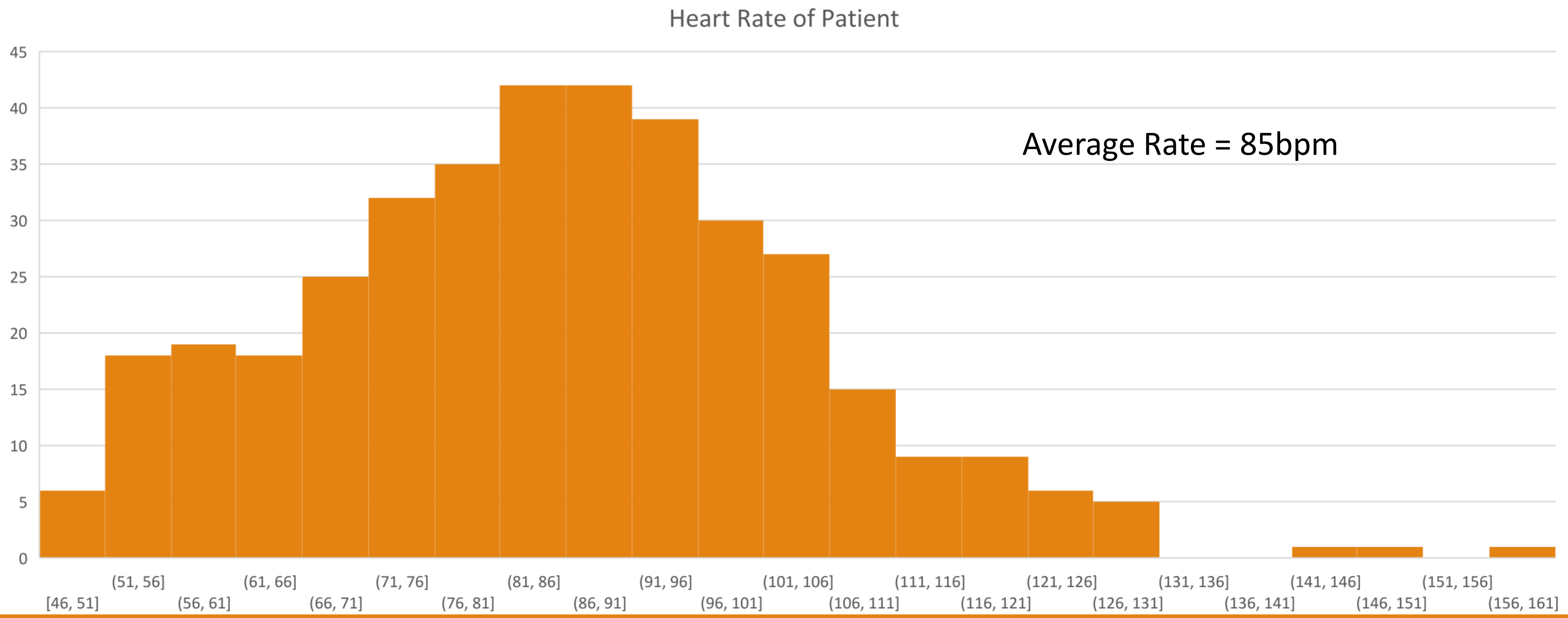
Type	Definition	Examples	Synonyms
Discrete	• Can only take on integer values	• Counts	• Integer, counts
Continuous	• Can take on any value in a domain		• Float, numeric
Interval	• Measured on a continuum (often scaled in a linear fashion)	• Temperature	
Ratio	• Like interval, but 0 represents “no value”	• Height, mass, distance	

Summarizing Data

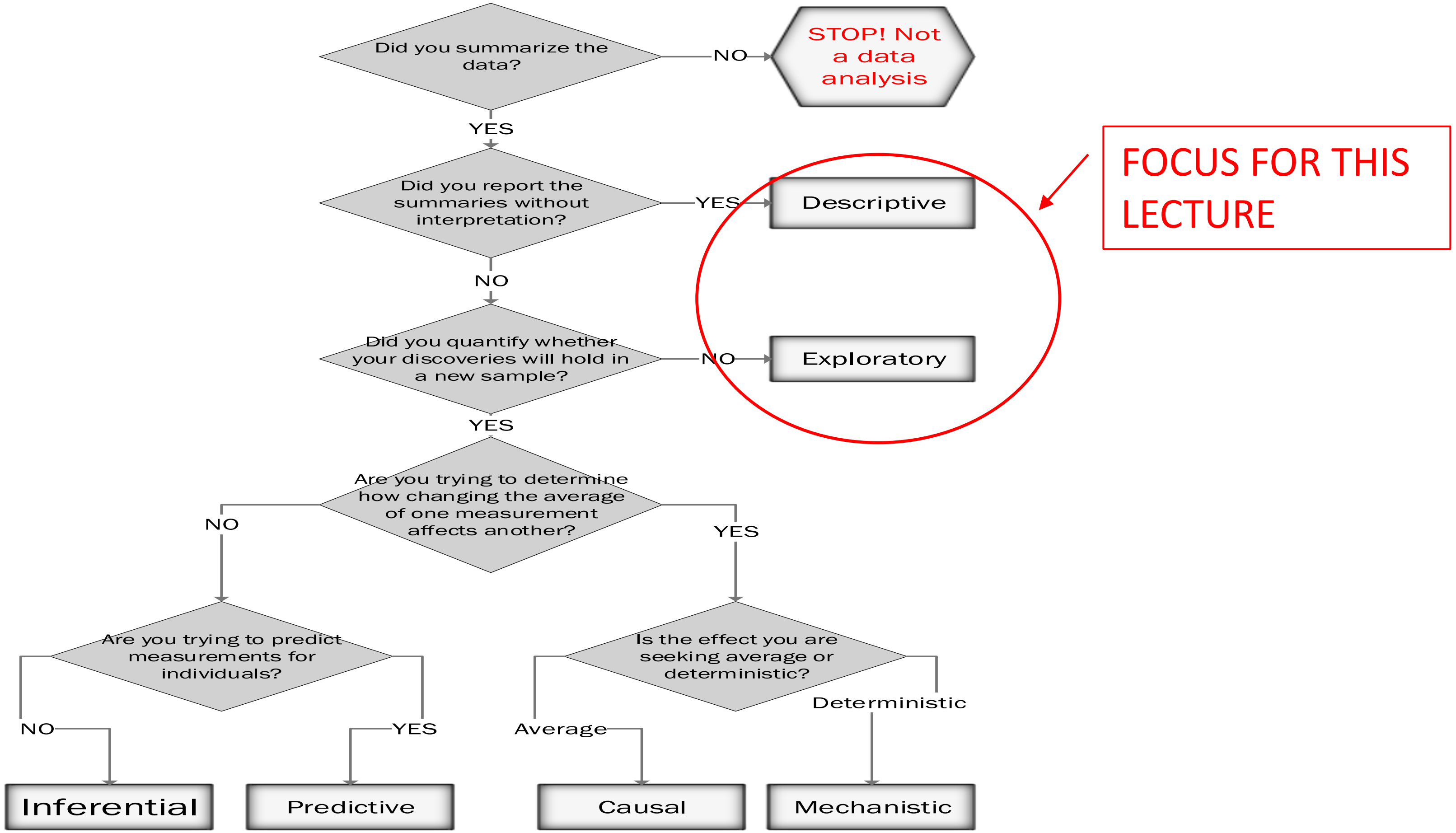
Describe the heart rate of this patient. Is it normal? Too high? Too low?

78	112	101	84	78	99	74	65	122	73	106	52	83	65	93	85	67	54	62	96
90	85	150	74	100	66	100	87	111	88	93	86	92	74	105	114	84	96	72	68
116	99	87	101	80	100	87	117	51	77	112	78	97	52	81	72	90	97	84	85
128	104	53	82	73	81	90	55	72	80	96	75	102	83	90	71	87	67	103	60
74	59	80	73	72	59	82	69	108	51	94	96	82	92	65	89	110	81	85	67
82	72	60	59	122	95	80	73	102	61	93	75	95	81	98	54	93	86	78	96
119	46	63	84	78	84	85	105	89	67	88	96	83	98	85	81	100	86	92	97
76	66	57	93	80	57	59	94	77	108	80	101	119	51	79	64	65	157	86	88
89	126	98	81	85	83	77	89	74	103	87	87	79	68	82	69	91	107	83	80
95	94	49	124	91	93	129	60	123	107	73	86	100	85	58	77	79	95	84	86
109	91	99	121	72	66	57	52	59	107	77	87	95	111	106	104	91	49	56	64
64	63	59	78	67	101	53	112	118	98	77	82	93	109	98	55	95	83	105	59
103	100	90	129	50	87	63	51	66	62	93	76	116	97	82	98	114	55	90	83
85	89	78	82	102	81	78	91	71	86	116	123	92	90	105	107	84	73	75	103
104	54	87	78	87	69	99	68	124	82	94	78	109	49	57	72	93	65	95	116
91	73	105	114	102	96	90	87	79	68	89	68	88	105	68	63	85	94	95	69
101	91	87	71	93	77	87	72	60	67	69	78	114	65	80	104	80	86	130	107
75	110	70	106	54	67	60	71	103	80	47	66	88	96	66	102	65	94	56	74
97	53	92	71	89	98	70	90	104	103	75	74	91	93	143	83	76	52	95	84

Describe the heart rate of this patient. Is it normal? Too high? Too low?



Everything from here on out in the course is about summarizing data.



Modified from Jeffery T. Leek, Roger D. Peng; Science 20 Mar 2015: Vol. 347, Issue 6228, pp. 1314-1315, DOI: 10.1126/science.aaa6146s

Descriptive Analysis a.k.a. 90% of Analysis

A descriptive data analysis seeks to summarize and organize the data so they can be easily understood. This is presented without further interpretation or commentary.

E.g. Canadian Census. The Census collects data on all the people in Canada at a specific time.

Statistics Canada conducts the Census of Population in order to develop a statistical portrait of Canada and Canadians on one specific day. The census is designed to provide information about people and housing units in Canada by their demographic, social and economic characteristics.

(http://www23.statcan.gc.ca/imdb/p2SV.pl?Function=getSurvey&SDDS=3901)

Why bother with Descriptive Analysis?

1. Descriptive Analysis could be an end in itself (E.g. Summarization, Control Charts, Census, Survey of customer attitudes, current sentiment analysis). By summarizing data, descriptive analysis simplifies and hastens understanding of features.

2. But, necessary first step in any analytical procedure
   1. Get to know the data
   2. Double check on accuracy, reliability, etc.

3. Asks Questions!
   1. Does the range make sense?
   2. Are there negative values when there should be none?
   3. Does the summary make sense or seem reasonable?
   4. Have I not thought of certain questions

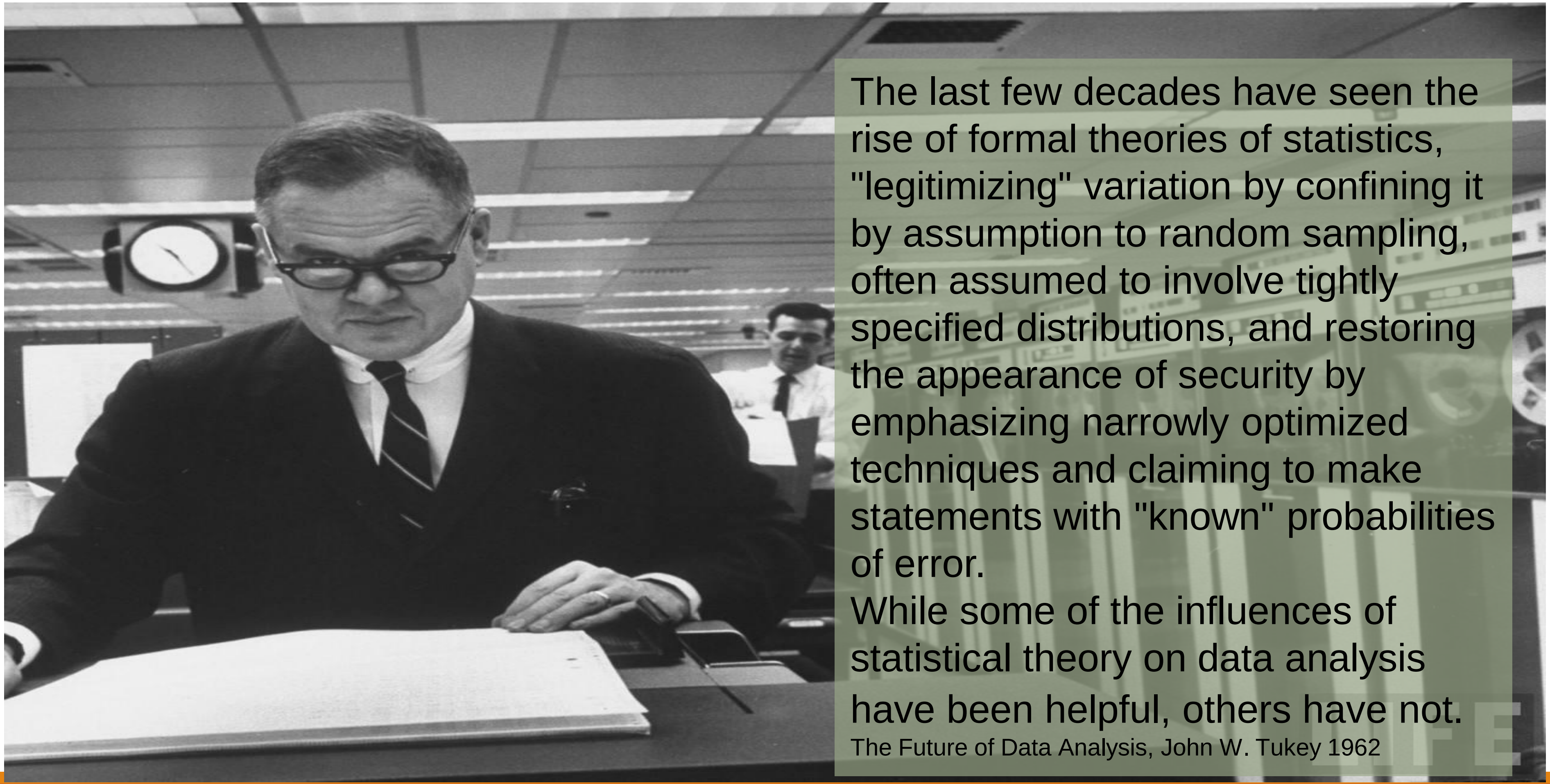
Exploratory Data Analysis (EDA) is relatively new – by extension, so is this use of Descriptive Data Analysis

John Tukey proposed a science of data analysis in 1962 in a paper *The Future of Data Analysis*

He wrote a foundational text on the subject which is *still* relevant: *Exploratory Data Analysis, 1977.*

He combined statistics, engineering and computer science (and coined the terms *software* and *bit* (short for *binary digit*)).

Data Analysis and Statistics are NOT the same thing



The last few decades have seen the rise of formal theories of statistics, "legitimizing" variation by confining it by assumption to random sampling, often assumed to involve tightly specified distributions, and restoring the appearance of security by emphasizing narrowly optimized techniques and claiming to make statements with "known" probabilities of error.

While some of the influences of statistical theory on data analysis have been helpful, others have not.

The Future of Data Analysis, John W. Tukey 1962

Data Analysis and Statistics are NOT the same thing

Exposure, the effective laying open of the data to display the unanticipated, is to us a major portion of data analysis. Formal statistics has given almost no guidance to exposure; indeed, it is not clear how the informality and flexibility appropriate to the exploratory character of exposure can be fitted into any of the structures of formal statistics so far proposed.

The Future of Data Analysis, John W. Tukey 1962

Types of Descriptive Statistics

Summarize Data

- Measures of Location
- Measures of Dispersion

Organize Data

- Tables
- Graphs

Summarize Data

MEASURES OF LOCATION

Summarizing Data

Measures of Location

- Mean
- Median
- Mode

Measures of Dispersion

- Range
- Quartiles
- Deviation
- Variance
- Standard Deviation

Measures of Location (Central Tendency) MEAN

- The Mean is a measure of *central tendency*
  - What most people mean by “average”
  - Sum of a set of numbers divided by the number of numbers in the set

$$\mu = \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Summarizing Data

Measures of Location

- Mean
- Median
- Mode

Measures of Dispersion

- Range
- Quartiles
- Deviation
- Variance
- Standard Deviation

Measures of Location (Central Tendency) MEDIAN & MODE

Median is the middle value when a set of numbers are arranged in ascending order. That is, it divides a set precisely in to two equal halves: 50% above the median, 50% below the median. Also known as the 50^{th} percentile.

Mode is the most frequently occurring number in a set.
Is it possible that there are no modes, one mode or multiple modes in a set of data,

Summary of Central Tendency Measures

Type	Definition	Synonyms
Mean	Sum of all values divided by number of values.	Average
Median	The value such that one half of the data lies above and below	50% percentile
Mode	The most frequently occurring number.	

Adapted from Bruce and Bruce, 2017, Wiley Academic

Summarizing Data

Measures of Location

- Mean
- Median
- Mode

Measures of Dispersion

- Range
- Quartiles
- Deviation
- Variance
- Standard Deviation

Measures of Dispersion (a.k.a. How spread out the data is)

A measure of dispersion, is used to describe the variability (how spread out or tightly clustered) in a sample or population. It is usually used in conjunction with a measure of central tendency, such as the mean or median, to provide an overall description of a set of data.

Example

Data set 1: [0,25,50,75,100]

Data set 2: [48,49,50,51,52]

Both have a mean of 50, but data set 1 clearly has greater dispersion than data set 2.

Measures of Dispersion
RANGE

The Range is one measure of dispersion

The range is the difference between the maximum and minimum values in a set

Example

Data set 1: [1,25,50,75,100]; R: 100-1 = 99

Data set 2: [48,49,50,51,52]; R: 52-48 = 4

The range ignores how data are distributed and only takes the extreme scores into account

RANGE = (Xlargest – Xsmallest)

Measures of Dispersion: Variance and Standard Deviation

Variance:

The variance of a set of numbers is the average of the square of the deviations from the mean.

$$Var(x) = \sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Standard Deviation:

The standard deviation of a set of numbers is the positive square root of the variance.

$$\sigma = \sqrt{\sigma^2} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Organizing Data

TABLES AND GRAPHS

Types of Descriptive Statistics

Summarize Data

- Measures of Location
- Measures of Dispersion

Display Data

- Plots
- Graphs

Visualizing Data

Displaying Data

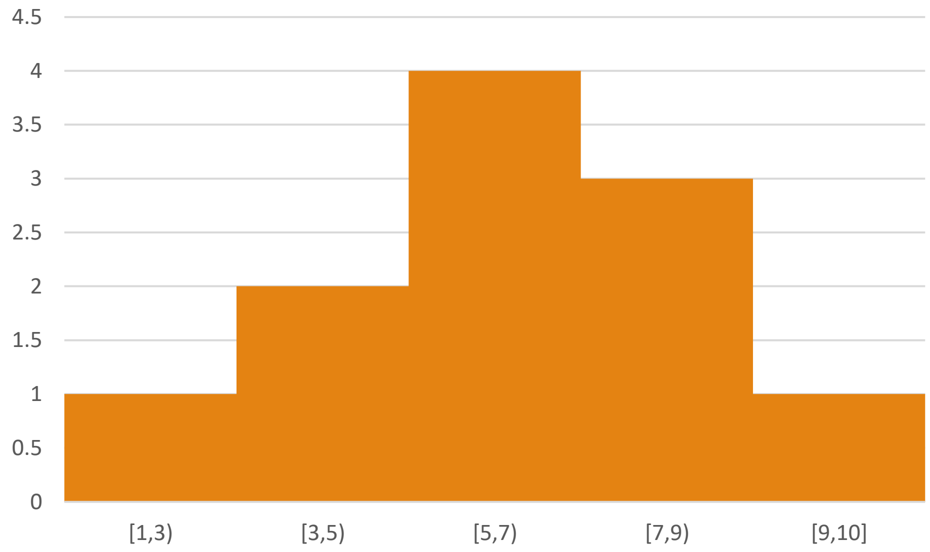
Numeric Data

- Histogram
- Box-Whisker Plots
- Venn Diagrams
- Time Series Plots
- Scatter Plots

Histogram

A histogram (or relative frequency histogram) is a display that includes contiguous rectangles. The axis of measurement is divided in to equal intervals with rectangles constructed on top with heights proportional to the percentage of data falling in to the interval.

Provides the shape, centre, and dispersion of the dataset.

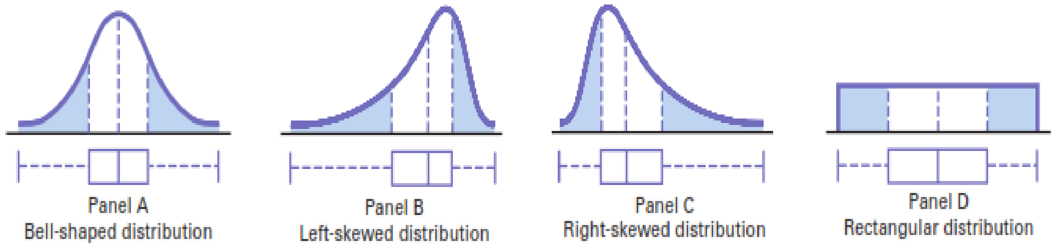


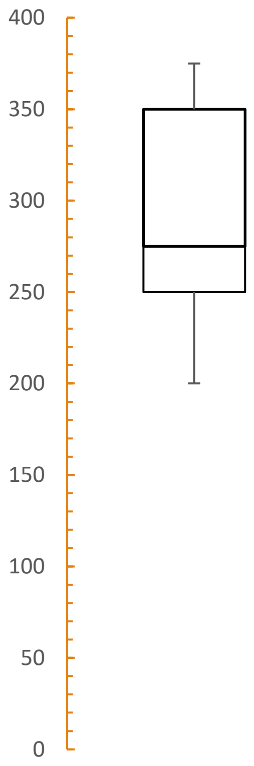
Box-Whisker Plots

A Box-Whisker plot consists of five numbers: median, 3rd quartile, 1st quartile, minimum and maximum.

The interquartile range (Q3-Q1) is described by the box and therefore it contains ~ 50% of the data.

The "whiskers" extend out to the minimum and the maximum.





Scatter Plots

Scatter plots are useful for comparing paired data sets (i.e. data with two data points each. For example, height and weight of patients).

The positions of the points in the plot represent the values of the data points plotted on the cartesian plane.

They are very useful for comparing the relationships between data sets (or variables).