ISA 401: Business Intelligence & Data Visualization

12: Data Correction and Imputation

Fadel M. Megahed, PhD

Raymond E. Glos Professor in Business
Farmer School of Business
Miami University

@FadelMegahed
fmegahed
fmegahed@miamioh.edu
Automated Scheduler for Office Hours

Fall 2025

Quick Refresher from Last Class

- ✅ Converting tidy data to technically correct data
- ✅ Examining the consistency of data

Learning Objectives for Today's Class

- Correct errors (with transformation rules, deductive correction and deterministic imputation)
- Describe different scenarios for missing data
- Apply some basic imputation techniques for missing data (e.g., mean/median/mode, kNN, etc)

Error Correction
(i.e., Correcting Inconsistencies in the Data)

Correction

> Correction methods aim to fix inconsistent observations by **altering invalid values in a record based on information from valid values**. Depending on the method this is either a **single-step procedure** or a **two-step procedure** where first, an error localization method is used to empty certain fields, followed by an imputation step. --- De Jonge and Van Der Loo (2013)

Simple Transformation Rules

In practice, data cleaning procedures involve a lot of **ad-hoc transformations**. This may lead to long scripts where one selects parts of the data, changes some variables, selects another part, changes some more variables, etc.

- When such scripts are neatly written and commented, they can almost be **treated as a log of the actions** performed by the analyst.
- However, as scripts get longer it is **better to store the transformation rules separately and log which rule is executed on what record**.

One approach to **do** this in R is through the deducorrect package.

type	package	function()	description
rules	deducorrect	correctionRules()	Rules for determinstic correction
correction	deducorrect	correctWithRules()	Apply simple replacement rules to a data.frame.

Simple Transformation Rules: Example

```
people = tibble::tibble(
  name = c('Tom Cruise', 'Serena Williams
  height = c(172, 1.75, 69.3, 154, 6.25),
  unit = c('cm', 'm', 'inch', 'cm', 'ft')
)

u <- deducorrect::correctionRules(
  expression(
    ## 1-------
    if(unit == "cm") height <- height/100
    ## 2-------
    if(unit == "inch") height <- height/3
    ## 3-------
    if(unit == "ft") height <- height/3.2
    ## 4-------
    unit <- "m"
  ))
```

Note that the rules **can also be stored in a txt file**.

```
people_corrected = deducorrect::correctWi
  rules = u, dat = people)

people_corrected$corrected
```

```
## # A tibble: 5 × 3
##   name             height unit
##   <chr>             <dbl> <chr>
## 1 Tom Cruise         1.72 m
## 2 Serena Williams    1.75 m
## 3 Taylor Swift       1.76 m
## 4 Ariana Grande      1.54 m
## 5 The Rock           1.91 m
```

```
people_corrected$corrections
```

```
##   row variable  old      new
## 1   1   height  172     1.72     if (u
## 2   1     unit   cm        m
```

Deductive Correction

When the data you are analyzing is generated by people rather than machines or measurement devices, certain typical human-generated errors are likely to occur. Examples include: (a) **typing errors** in numbers, (b) **rounding errors** in numbers, and (c) **sign errors**.

These errors can be accounted for through the deducorrect package.

type	package	function()	description
rule def	editrules	editmatrix()	A set of comparison operators representing a linear system of (in)equations. The function editmatrix generates an editmatrix from a character vector, or an expression vector.
typos	deducorrect	correctTypos()	Tries to detect and repair records that violate linear equality constraints by correcting simple typos
rounding	deducorrect	correctRounding()	Tries to detect and repair records that violate linear (in)equality constraints by correcting possible rounding errors
sign	deducorrect	correctSigns()	Correct sign errors and value interchanges in data records.

Deductive Correction: Correct Typos

```
e <- editrules::editmatrix("x + y == z") # defining the rules
d <- data.frame(x = 123, y = 132, z = 246) # creating the data frame
cor <- deducorrect::correctTypos(e, d) # attempting to correct Typos
cor$corrected # new values for the data.frame
```

```
##     x   y   z
## 1 123 123 246
```

```
cor$corrections # the log of what has changed
```

```
##   row variable old new
## 1   1        y 132 123
```

Deductive Correction: Correct Rounding Errors

```
e <- editrules::editmatrix("x + y == z") # defining the rules
d <- data.frame(x = 101, y = 100, z = 200) # creating the data frame
cor <- deducorrect::correctRounding(e, d) # attempting to correct Rounding
cor$corrected # new values for the data.frame (may not be unique)
```

```
##     x   y   z
## 1 100 100 200
```

```
cor$corrections # the log of what has changed
```

```
##   row variable old new
## 1   1        x 101 100
```

Deductive Correction: Correct Sign Errors

```
e <- editrules::editmatrix("x + y == z") # defining the rules
d <- data.frame(x = -100, y = 100, z = 200) # creating the data frame
cor <- deducorrect::correctSigns(e, d) # attempting to correct the sign
cor$corrected # new values for the data.frame
```

```
##     x   y   z
## 1 100 100 200
```

```
cor$corrections # the log of what has changed
```

```
##   row variable  old new
## 1   1        x -100 100
```

Non-Graded Activity

03:00

Activity | Your Solution

> *Over the next 3 minutes, please examine the following:*

- Which of the aforementioned `correct...()` functions can be used to correct the following situation?
  - `correctWithRules()`
  - `correctTypos()`
  - `correctSigns()`
  - `correctRounding()`

```
d <- data.frame(x = 173, y = 200, z = 379) # creating the data frame
```

Missing Data Handling (Complete Cases, Col. Deletion, & Imputation)

Types of Missing Data: MCAR

Missing data are typically grouped into three categories:

- **Missing completely at random (MCAR):** When data are MCAR, the fact that the data are missing is independent of the observed and unobserved data. For example,
  - If we were to perform annual health checks for all FSB students, some of you may have missing lab values because a batch of lab samples was processed improperly.
  - Here, the missing data **reduce the analyzable population of the study and consequently, the statistical power, but do not introduce bias:** when data are MCAR, the **remaining data can be considered a simple random sample of the full data set** of interest.
  - Hence, the `NA` data can be simply be handled by removing such observations. In R, the `na.omit()` can be used without the need for introducing any packages. We refer to this `NA` handling approach as **complete case analysis**.

Source: Mack C, Su Z, Westreich D. Managing Missing Data in Patient Registries: Addendum to Registries for Evaluating Patient Outcomes: A User's Guide, Third Edition [Internet]. Rockville (MD): Agency for Healthcare Research and Quality (US); 2018 Feb. Types of Missing Data. Available from: https://www.ncbi.nlm.nih.gov/books/NBK493614/

Types of Missing Data: MAR

- **Missing at random (MAR):** When data are MAR, the fact that the **data are missing is systematically related to the observed but not the unobserved data**. For example,
  - A dataset examining depression may encounter data that are MAR if male participants are less likely to complete a survey about depression severity than female participants.
  - That is, if the probability of survey completion is related to their `sex` (which is fully observed) but **not the severity of their depression**, then the data may be regarded as MAR.
  - Approaches to Handle `NAs` for MAR Data:
    - Complete case analyses of MAR data may or may not result in bias.
    - Impute (i.e., guess) the missing values based on a proper accounting of the known factors (e.g., their sex)

Source: Mack C, Su Z, Westreich D. Managing Missing Data in Patient Registries: Addendum to Registries for Evaluating Patient Outcomes: A User's Guide, Third Edition [Internet]. Rockville (MD): Agency for Healthcare Research and Quality (US); 2018 Feb. Types of Missing Data. Available from: https://www.ncbi.nlm.nih.gov/books/NBK493614/

Types of Missing Data: MNAR

- **Missing not at random (MNAR):** When data are MNAR, the fact that the data are missing is **systematically related to the unobserved data**, that is, the missingness is related to events or factors which are not measured by the researcher. To extend the previous example,
  - the depression registry may encounter data that are MNAR if participants with severe depression are more likely to refuse to complete the survey about depression severity.
  - Approaches to Handle `NAs` for MAR Data:
    - Complete case analyses of MAR data may or may not result in bias.
    - Imputation approaches **may not be suitable since the missingness pattern is due to the impact of unobserved auxiliary variables**.

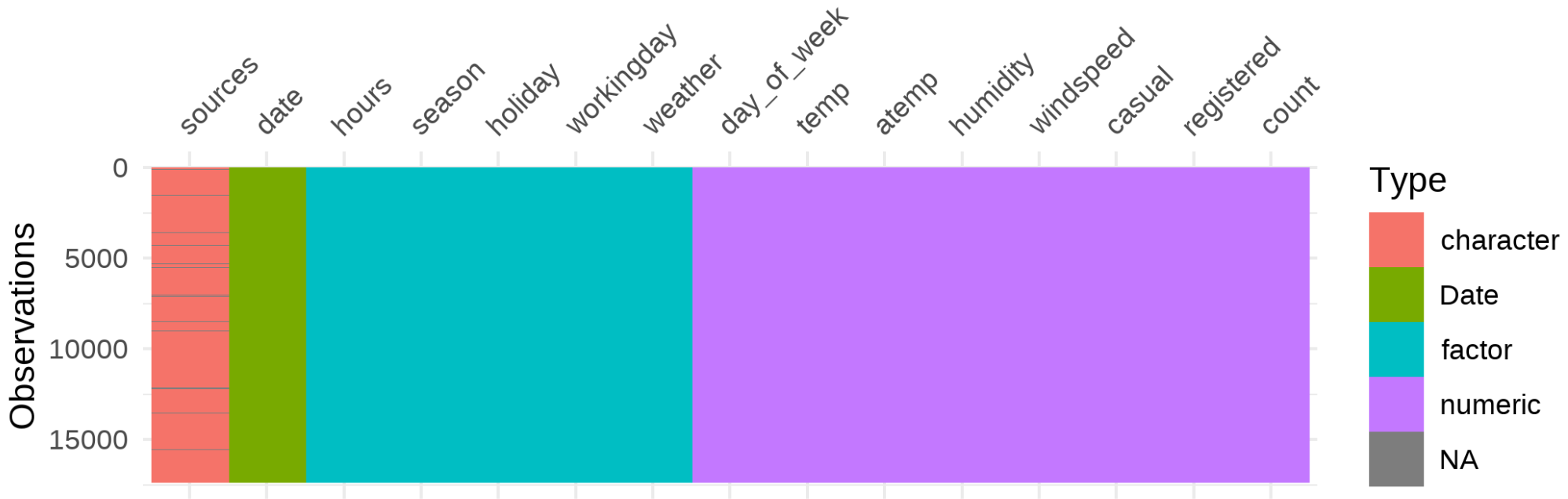
Source: Mack C, Su Z, Westreich D. Managing Missing Data in Patient Registries: Addendum to Registries for Evaluating Patient Outcomes: A User's Guide, Third Edition [Internet]. Rockville (MD): Agency for Healthcare Research and Quality (US); 2018 Feb. Types of Missing Data. Available from: https://www.ncbi.nlm.nih.gov/books/NBK493614/

The Bike Sharing Dataset

```
bike_tbl = readr::read_csv('../../data/bike_sharing_data.csv') |>
  # making their column names tidy
  janitor::clean_names() |>
  # creating date, hours, and day of the week variables
  dplyr::mutate(datetime = lubridate::mdy_hm(datetime), date = lubridate::as_date(datetime),
         hours = lubridate::hour(datetime), day_of_week =  lubridate::wday(date)) |>
  # moving them to the front
  dplyr::relocate(date, day_of_week, hours) |>
  select(-c(datetime)) |>
  # converting int columns to factors
  dplyr::mutate(dplyr::across(.cols = c(hours, season, holiday, workingday, weather),
                        .fn = as.character)) |>
  dplyr::mutate(dplyr::across(.cols = c(hours, season, holiday, workingday, weather),
                        .fn = as.factor))
```

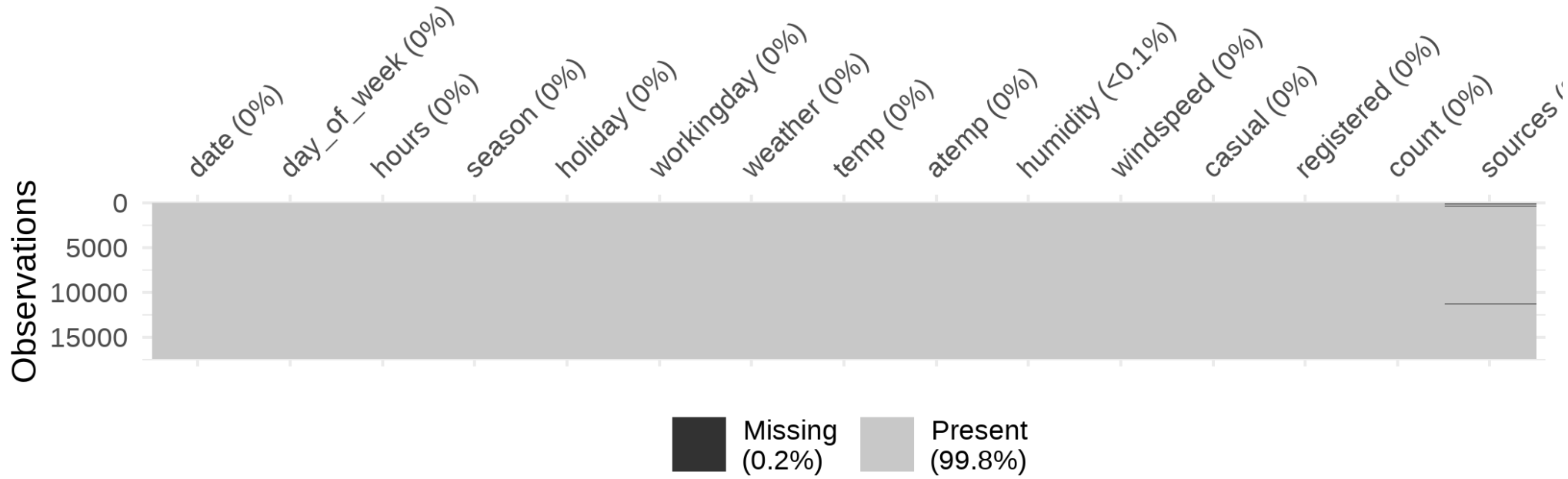
Bike Sharing Data: Viz Missing [1]

```
visdat::vis_dat(bike_tbl) # function from visdat
```



Bike Sharing Data: Viz Missing [2]

```
visdat::vis_miss(bike_tbl) # function from visdat
```



Bike Sharing Data: Missing Relationships [1]

```
bike_tbl_grouped = bike_tbl |>
  dplyr::group_by(day_of_week) |> # group_by from dplyr #>>
  dplyr::summarise(nmissing = is.na(sources) |> sum()) # summarise from dplyr #>>

bike_tbl_grouped |> DT::datatable(
  fillContainer = FALSE, options = list(pageLength = 4), height = 600)
```

Show 4 entries Search:

	day_of_week	nmissing
1	1	77
2	2	88
3	3	75
4	4	79

Showing 1 to 4 of 7 entries Previous 1 2 Next

Bike Sharing Data: Missing Relationships [2]

```
bike_tbl_grouped = bike_tbl |>
  dplyr::group_by(day_of_week, hours) |> # group_by from dplyr #>>
  dplyr::summarise(nmissing = is.na(sources) |> sum()) # summarise from dplyr #>>

bike_tbl_grouped |> DT::datatable(
  fillContainer = FALSE, options = list(pageLength = 4), height = 600)
```

Show 4 entries Search:

	day_of_week	hours	nmissing
1	1	0	4
2	1	1	3
3	1	10	3
4	1	11	1

Showing 1 to 4 of 168 entries

Previous 1 2 3 4 5 … 42 Next

Some Useful Imputation Functions

type	package	function()	description
identify NA cells	base	is.na()	returns logicals T/F for each NA cell
identify NA in each col	base & purrr	purrr::map_df(.x = df, .f= is.na) |> colSums()	returns total NA per column
rule-based deductions	deducorrect	deduImpute()	Based on observed values and edit rules, impute as many variables deductively as possible.
replace NAs	tidyr	replace_na(list(x = 0, y = 'unknown'))	Replace/Impute NA with pre-configured values (e.g., mean, median, VIM::maxCat(), etc.)
across column imputations	VIM	kNN()	k-nearest neighbours for imputation
across column imputations	VIM	rangerImpute()	use of random forests for imputation
multiple imputation	mice	mice()	multivariate imputation by chained equations

A Simple Demo Using a Modified Iris Dataset

Please refer to our in class demo for more details.

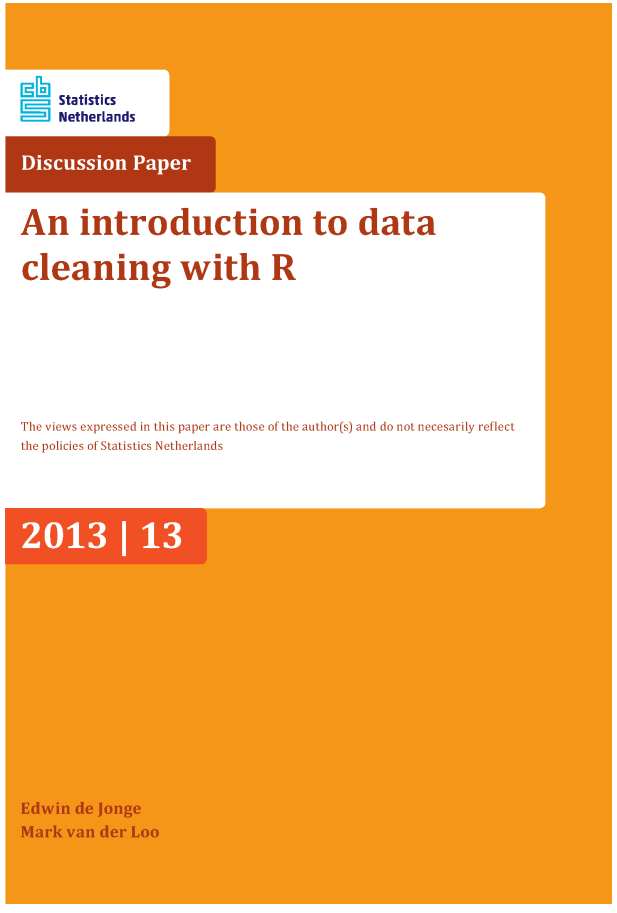
Recap

Summary of Main Points

By now, you should be able to do the following:

- Correct errors (with transformation rules, deductive correction and deterministic imputation)
- Describe different scenarios for missing data
- Apply some basic imputation techniques for missing data (e.g., mean/median/mode, kNN, etc)

Supplementary Reading



- From technically correct to consistent data