Diagnosing and Treating Data Issues
Week 6

PH 700A, Spring 2025

Table of contents

1 Week 6 Data Diagnostics and Treatment

1.1 Session Overview

- Continuous Data Treatment
- Categorical Data Treatment
- String Data and Regular Expressions
- Evaluating Missing Data

1.2 Packages

- `outliers`
- `finalfit`
- `VIM`

1.3 Questions on Data Origins

- Were the participants randomly or non-randomly selected?
- Can you assume that participants are representative of their source?
- Were the measurements collected without bias?
- **Do you have any reason to believe the data is not a representation of reality?**

2 Continuous Data Analysis

2.1 Outlier Detection with `outliers`

The package `outliers` contains commands for statistical tests to identify outliers.

Each command will only assess for a single *most extreme* data point.

Manual removal of an observation should therefore be iterative.

Command	Use Case
`cochran.test()`	Variance test
`dixon.test()`	One value vs. normal distribution
`grubbs.test()`	Two values in one tail of the dist.

Omit outliers with caution – these tests are data-driven.

Outlier removal should be thoughtful, intentional, and performed only if the participants are unique compared to the whole sample.

2.2 Assumptions

"LINE" Mnemonic:

L - *Linearity* of Association

I - *Independence* of Residuals (Error)

N - *Normality* of Residuals (Error)

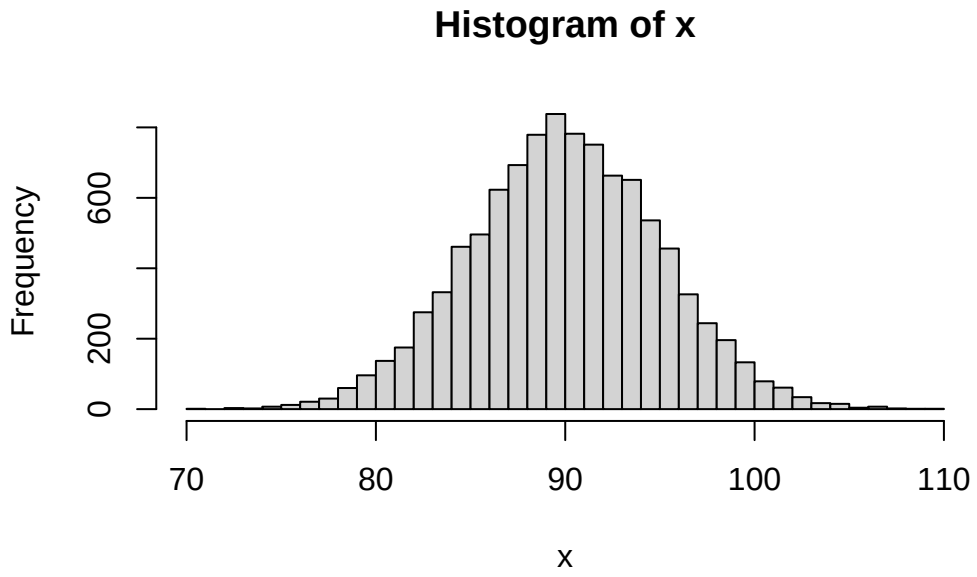
E - *Equality* of Variances

Also:

Orthogonality of the Predictors

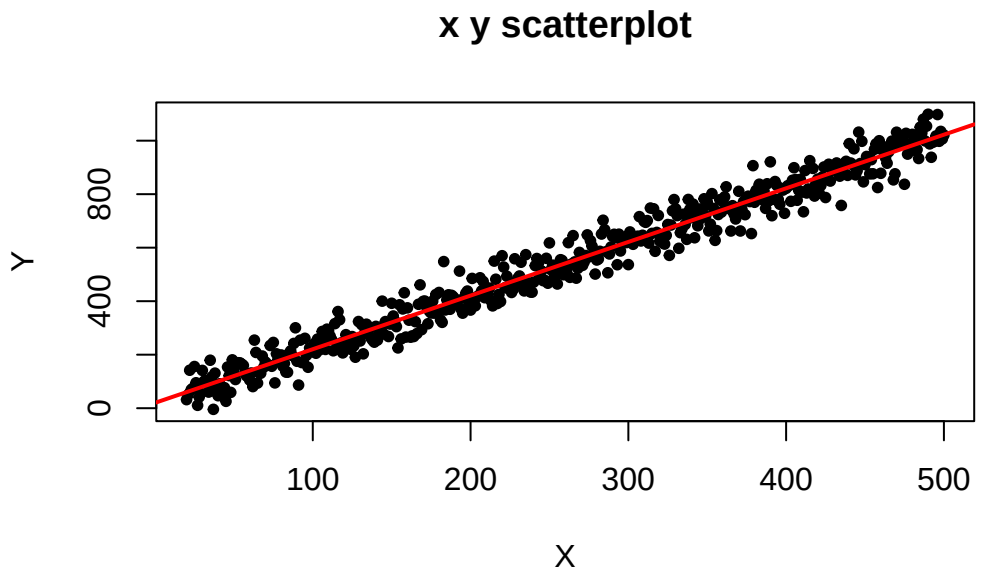
2.3 Univariate Distribution

An example of a normal distribution of one continuous variable.

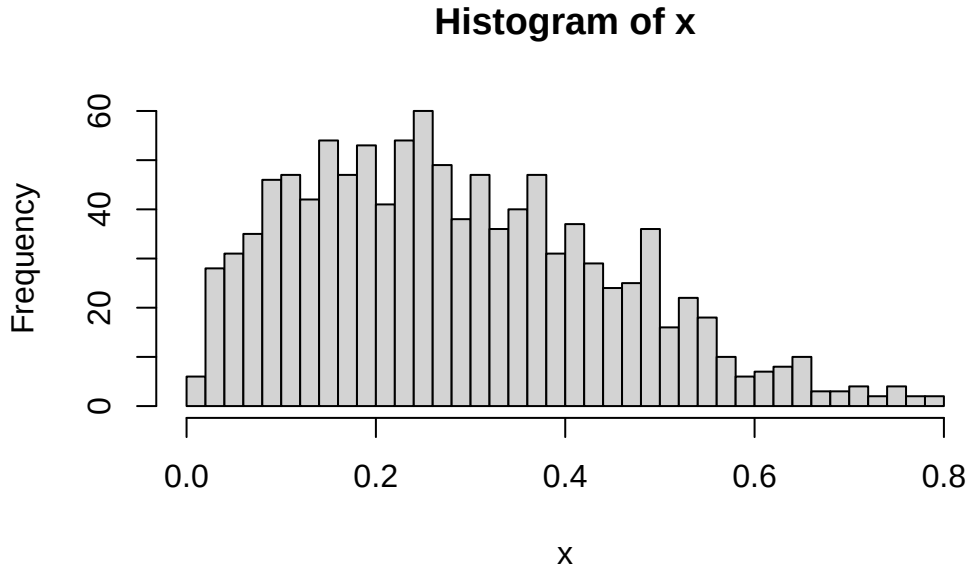


2.4 Bivariate Relationship

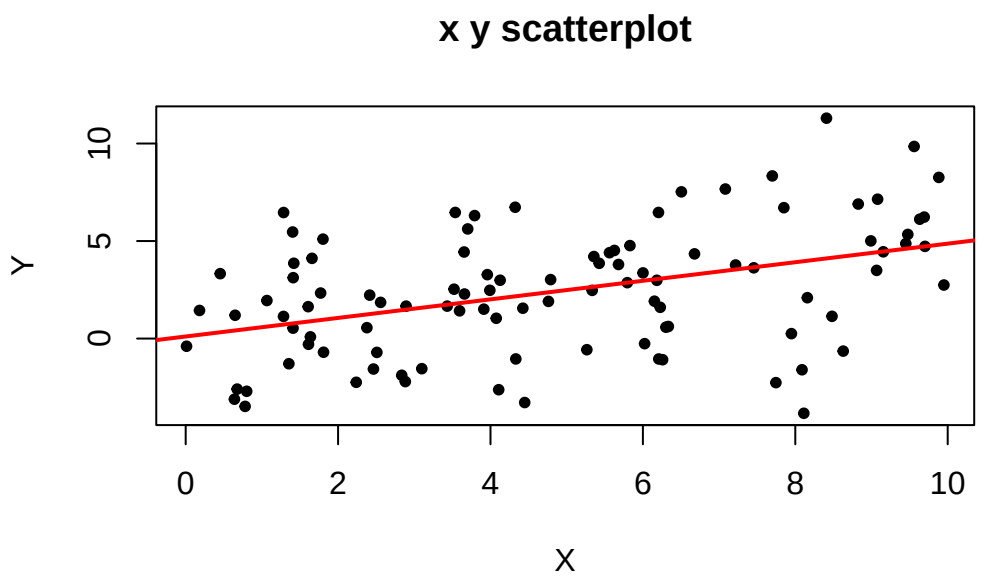
An example of a linear relationship between two continuous variables.



2.5 Violation of Assumptions



2.6 Non-standard Bivariate Relationship



2.7 Mathematical Transformations

Continuous variables can be transformed to adhere to assumptions.

Transformation	Code
Base 10 Logarithm	`log10(var)`
Base 10 Antilog	`10^(var)`
Natural Logarithm	`log(var)`
Natural Exponent	`exp(var)`
Square Exponent	`(var)^2`
Square Root	`sqrt(var)`
Inverse	`1/(var)`
Mean Centering	`var - mean(var)`

2.8 Example Code Using Mutate

```
df <- df %>%
      mutate(
          varCentered = var - mean(var),
          varLn = log(var),
          varSqrt = sqrt(var),
          varInv = 1/var,
```

```
        varSq = var^2
    )
```

If assumptions cannot be met with mathematical transformation, consider *categorization* or *quantile generation.*

2.9 String Data Processing

Character data must be converted to a numeric form for analysis.

- Few variations in strings can be manually categorized or converted to factor directly

```
str(df$arrival_transport)
```

```
 chr [1:222] "WALK IN" "AMBULANCE" "AMBULANCE" "AMBULANCE" "AMBULANCE" ...
```

```
table(df$arrival_transport)
```

```
AMBULANCE     OTHER   UNKNOWN   WALK IN
      133         1        13        75
```

```
df$arrival_transport <- as.factor(df$arrival_transport)
str(df$arrival_transport)
```

```
 Factor w/ 4 levels "AMBULANCE","OTHER",..: 4 1 1 1 1 3 3 4 4 4 ...
```

2.10 Unstandardized Strings

Strings that vary highly due to capitalization, symbols, spelling, or detail requires special consideration.

```
tab1 <- table(df.triage$chiefcomplaint)
print(tab1)
```

```
                      ? AORTIC DISECTION
                                      1
                             ?INFECTION
                                      1
                               Abd pain
                                      5
                               ABD PAIN
                                      1
          Abd pain, Abdominal distention
```

1	
Abd pain, Back pain	
2	
Abd pain, Diarrhea, Vomiting	
1	
Abd pain, Dysuria	
1	
Abd pain, LEAKING ASCITES	
1	
Abd pain, N/V	
2	
Abd pain, n/v/d	
2	
Abd pain, Right sided abdominal pain	
1	
Abd pain, Transfer	
1	
Abdominal distention	
1	
Abdominal distention, Abd pain, LETHAGIC	
1	
ABDOMINAL MASS, FEVER	
1	
Abnormal CT, LUQ abd pain	
1	
Abnormal labs	
2	
Abnormal labs, ETOH, Hypotension	
1	
Abnormal labs, Hyperglycemia	
1	
Abnormal labs, Weakness	
1	
Allergic reaction	
1	
Altered mental status	
4	
Anemia	
1	
Anemia, Neutropenia, Transfer	
1	
Assault	
1	
B Leg swelling	
1	
Back pain	
1	
Back pain, Decreased PO intake, R Ear pain	

1	
BRBPR	
3	
CAR VS POLE	
1	
Cardiac arrest, Transfer	
1	
Chest pain	
5	
Chest pain, Dizziness	
1	
Chest pain, Jaw pain, L Arm pain	
1	
Chest pain, N/V	
1	
Chest pain, Nausea	
1	
Chest pain, NSTEMI, Transfer	
1	
Chest pain, Transfer	
3	
Clogged foley	
1	
Coffee ground emesis	
1	
Confusion	
1	
Confusion, s/p Fall	
1	
Cough, Dyspnea	
3	
Dehydration, Fatigue	
1	
Dehydration, Nausea, Rash	
1	
Depression, Anxiety	
1	
Diarrhea, Hypotension	
1	
DIARRHEA,FEVER	
1	
DISLODGED ABD TUBE	
1	
Dizziness	
2	
Dizziness, Diplopia	
1	
Dizziness, Weakness	

	1
DKA, Transfer	1
DVT, Transfer	1
Dyspnea	4
Dyspnea on exertion	1
Dyspnea, ABNORMAL LAB VALUES	1
Dyspnea, Foot swelling	1
Dyspnea, Hypoxia	1
Dyspnea, Productive cough	1
Dyspnea, Transfer	3
ELEVATED INR	1
Epigastric pain	2
ETOH, Hallucinations	1
ETOH, Hypoglycemia	1
ETOH, SI	1
EYE EVAL	1
Eye pain	1
FACIAL DROOP	1
Facial numbness	1
Fatigue, s/p Fall	1
FEVER	1
Fever, Neutropenia	1
Foot laceration	1
HEAD BLEED	3
Head injury, Neck pain, s/p Fall	

	1
Head injury, s/p Fall	1
Headache, SAH, Transfer	1
Hematemesis	1
Hematuria	1
Hematuria, Dysuria	1
Hemodialysis	1
HYPERGLYCEMIA	1
Hyperglycemia, Overdose	1
Hypertension	1
HYPERTENSION	1
Hypoglycemia	1
Hypotension	3
HYPOTHERMIA	1
ICH	1
ILI	1
Insomnia	1
L Arm numbness, L Arm swelling	1
L Arm pain, L Arm swelling	1
L Eye pain	1
L Foot pain	1
L Foot pain, Wound eval	1
L Foot swelling, L Foot pain	1
L Knee pain	1
L Leg DVT	

	1
L Leg pain, s/p Fall	1
L Leg pain, Weakness	1
L Weakness, Unable to ambulate	1
LARYNGITIS/SOB WITH TALKING	1
LEFT EYE SWELLING, REDNESS	1
LEFT HAND PAINS	1
LEFT HIP FX	1
Leg pain, Wound eval	1
Leg weakness, Abnormal CT	1
Lethargy	1
LETHARGY/SOB	1
Lower back pain	1
Lower back pain, s/p Fall	1
LOWER EXTREMITY PAIN	1
MVC/INTUBATED TRAUMA	1
N/V	1
N/V, Hyperglycemia	1
N/V, Tachycardia	1
N/V, Tinnitus	1
n/v/d, Abd pain	1
Neck pain, Med refill	1
Palpitations	1
PE	1
PICC EVAL	

	1
PICC LINE INFECTION	1
Presyncope, Weakness	1
Psych eval	1
Psychiatric hold	2
Psychiatric hold, Altered mental status	1
R Foot pain	1
R FOOT PAIN	1
R FOOT ULCER/CELLULITIS	1
R LEG CELLULITIS	1
R Leg pain, R Leg swelling	2
R Leg pain, Weakness	1
R RIB PAIN	1
Rectal pain	1
RESP ARREST	1
RIGHT EAR PAIN/DRAINAGE	1
RIGHT FOOT INFECTION	1
Right sided abdominal pain	1
RLQ abdominal pain	1
S/P ARREST	1
s/p cardiac arrest	1
S/P FALL	1
s/p Fall, L Shoulder pain	1
s/p Fall, R Wrist pain, R Wrist injury	1
s/p Fall, SDH	

	1
s/p Fall, Transfer	1
SDH/SAH	1
SEIZURE	2
SHORTNESS OF BREATH	2
SI	3
SOB	1
SOB/ABNL LABS	1
Suprapubic pain	1
SW	1
Syncope	1
Syncope, Transfer	1
T-SPINE FX DISPLACEMENT	1
Tachycardia	1
Tachycardia, Hypoxia	1
Toe pain	1
Transfer	1
Transfer, Abnormal EKG	1
Transfer, CVA	1
Transfer, Dyspnea	1
Transfer, MVC	1
Transfer, PE	1
Transfer, s/p Fall	1
UNKNOWN-CC	2
Urinary retention	

```
                                     1
Urinary retention, Lower abdominal pain
                                     1
                            VOMITING BLOOD
                                     1
                                  Weakness
                                     1
                   Weakness, Abnormal labs
                                     1
Weakness, Atrial fibrillation, Transfer
                                     1
                      Weakness, GI bleed
                                     1
                    Weakness, Hypotension
                                     1
                                Wound eval
                                     2
                     Wound eval, Transfer
                                     1
```

2.11 Uniform Processing

Additional factors will need to be created to consolidate common values.

```
                        VOMITING BLOOD                                  Weakness
                                     1                                         1
                Weakness, Abnormal labs Weakness, Atrial fibrillation, Transfer
                                     1                                         1
                     Weakness, GI bleed                   Weakness, Hypotension
                                     1                                         1
                              Wound eval                    Wound eval, Transfer
                                     2                                         1
```

Patterns will vary highly.

Some professional inference must be applied when generating these factors.

```
                                  Var1 Freq
1                  Abnormal labs, Weakness    1
2                      Dizziness, Weakness    1
3                    L Leg pain, Weakness     1
4           L Weakness, Unable to ambulate    1
5                Leg weakness, Abnormal CT    1
6                     Presyncope, Weakness    1
7                     R Leg pain, Weakness    1
```

```
8                                  Weakness    1
9                  Weakness, Abnormal labs    1
10 Weakness, Atrial fibrillation, Transfer    1
11                      Weakness, GI bleed    1
12                    Weakness, Hypotension    1
```

2.12 Regular Expressions

"Regex" functions evaluate *character patterns*. These patterns are validated between what is provided by you and what is found in the data, and then another function can be applied based on the result.

Using regex functions requires some knowledge of special characters with unique meanings.

Symbol	Definition
[]	singular character of a set of characters or a range
A-Z	any single capital letter between "A" and "Z"
a-z	any single lowercase letter between "a" and "z"
0-9	any single number between 0 and 9
?	denotes an optional pattern that does not need to be matched
+	repeating; a preceding character must match *at least* once
*	optional or repeatable; the preceding character can occur any number of times
^	leading anchor; a character that follows must *start* the entire string
$	trailing anchor; the character that precedes must be at the *end* of the string
.	any character of any type. essentially a *wildcard* slot
|	alternation; essentially an "or" symbol allowing any pattern in a list
{}	quantifier; a character can occur a specific number of times

2.13 Regex Examples

Regex Entry	Matches	Does Not Match
^I8	"I8", "I8 & 163"	"the I8"
colou?r	"color", "colour"	"coloor"
I[0-2]	"I0", "I1", "I2"	"I3", "i2"
Rich+	"Rich", "Richard", "Richhhhhh"	"Rick"
[5]$	"12345", "555", "asdf5"	"123456", "56", "asdf5"
PH 700.	"PH 700A", "PH 700a", "PH 700x"	"PH 700", "PH 70"

2.14 Functions

Base:

`grep()/grepl()`

Tidyverse:

The `stringr` package contains many functions for working with strings. See the *Cheatsheet* on String Manipulation with stringr for full command list and examples. Most useful ones include:

`str_detect()`

`str_sub()`

`str_replace()`

2.15 `grep/grepl` Differences

`grep` returns only the addresses based on detection of the pattern.

`grepl` returns a logical `TRUE` or `FALSE` based on detection of the pattern over the entire column.

```
grep("[Ww]eakness", df.triage$chiefcomplaint)
```

```
 [1]  32  33  34  50  71  77  85 135 136 159 210 214
```

```
grepl("[Ww]eakness", df.triage$chiefcomplaint)
```

```
  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE FALSE FALSE
 [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [49] FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE
 [73] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [85]  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [97] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[109] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[121] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[133] FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[145] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[157] FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[169] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[181] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[193] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[205] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE
[217] FALSE FALSE FALSE FALSE FALSE FALSE
```

2.16 `grepl` To Make Variables

```
df.triage <- df.triage %>%
  mutate(weakness = if_else(
                            grepl("[Ww]eakness", chiefcomplaint),
                            1,
                            0)
         )

table(df.triage$weakness)
```

```
  0   1
210  12
```

2.17 `stringr` To Make Variables

```
library(tidyverse)

table(str_detect(df.triage$chiefcomplaint, "[Ww]eakness"))
```

```
FALSE  TRUE
  210    12
```

```
df.triage <- df.triage %>%
  mutate(weakness = if_else(
                            str_detect(chiefcomplaint, "[Ww]eakness"),
                            1,
                            0)
         )
table(df.triage$weakness)
```

```
  0   1
210  12
```

2.18 Modifying Parts of Strings

Using `str_replace()`, we can modify and standardize existing character columns.

```
str_replace(column, "string to find", "string replacement")
```

Variants of `str_replace` include:

- `str_replace_all()` for multiple changes over different string patterns. Requires a `c()` list of patterns and changes
- `str_replace_na()` to change actual missing `NA` to a string "NA". Good for generating a unique category of "missing" or "unknown"

2.19 `str_replace` Example

```
tab1 %>% filter(grepl("[Ss]/[Pp]", Var1)) %>% print()
```

```
                                    Var1 Freq
1                     Confusion, s/p Fall    1
2                       Fatigue, s/p Fall    1
3        Head injury, Neck pain, s/p Fall    1
4                   Head injury, s/p Fall    1
5                    L Leg pain, s/p Fall    1
6               Lower back pain, s/p Fall    1
7                               S/P ARREST    1
8                     s/p cardiac arrest    1
9                                 S/P FALL    1
10              s/p Fall, L Shoulder pain    1
11 s/p Fall, R Wrist pain, R Wrist injury    1
12                          s/p Fall, SDH    1
13                    s/p Fall, Transfer    1
14                    Transfer, s/p Fall    1
```

"Status post", often written as "S/P", is medical shorthand that refers to a state after an intervention.

The "S/P" varies in capitalization and location in each row that it's found.

2.20 Standardizing S/P

```
df.triage <- df.triage %>%
  mutate(chiefcomplaint = str_replace_all(chiefcomplaint, c("s/p" = "status post", "S/P" = "status p

df.triage %>% select(chiefcomplaint) %>% filter(grepl("status post", chiefcomplaint)) %>% table()
```

```
chiefcomplaint
                   Confusion, status post Fall
                                             1
                     Fatigue, status post Fall
                                             1
      Head injury, Neck pain, status post Fall
                                             1
                Head injury, status post Fall
                                             1
                 L Leg pain, status post Fall
                                             1
             Lower back pain, status post Fall
                                             1
                             status post ARREST
                                             1
                   status post cardiac arrest
                                             1
                               status post FALL
                                             1
             status post Fall, L Shoulder pain
                                             1
status post Fall, R Wrist pain, R Wrist injury
                                             1
                          status post Fall, SDH
                                             1
                   status post Fall, Transfer
                                             1
                   Transfer, status post Fall
                                             1
```

2.21 Missing Values

True missing values should occur randomly throughout the data.

Missing values that exist with a pattern are biased.

Proper coding of missing values include:

- `NA`
- `NaN`

Improper coding of missing values include:

- `-1`
- `""`
- `.`
- `" "`

2.22 Pattern Identification by Observation

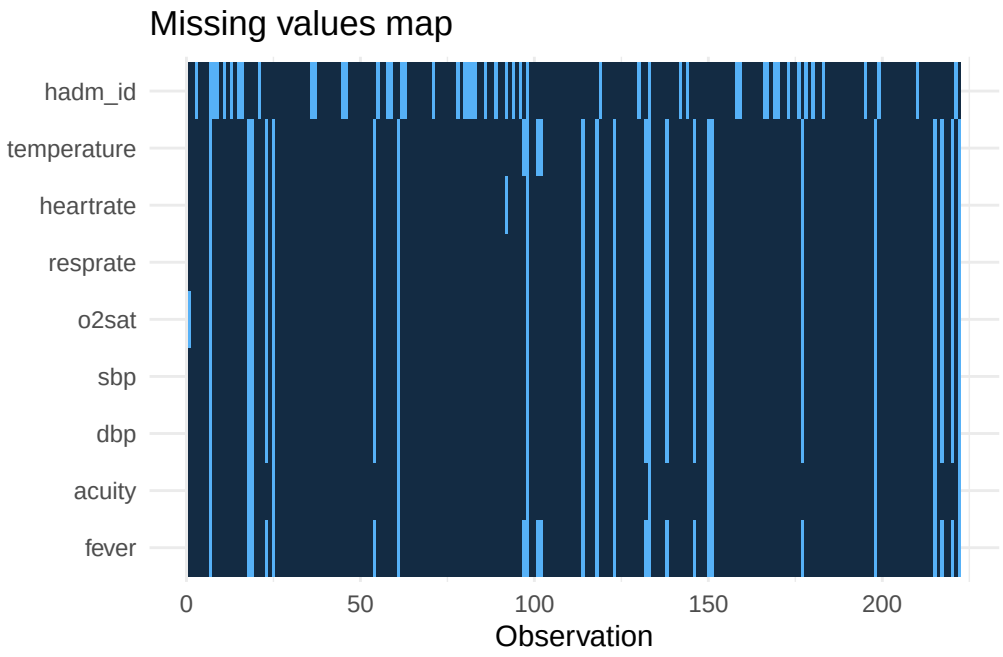
Using `library(finalfit)`, we can look for patterns among missing values.

Visual evaluation is performed using the `missing_plot()` command.

```
library(finalfit)

df %>%
  select(all_of(variables_to_check)) %>%
  missing_plot()
```

2.23 Missing Plot Example

```
library(finalfit)

df <- left_join(df, df.triage, by = "stay_id")
  df$subject_id.y <- NULL
  df <- df %>% rename(subject_id = subject_id.x)

df %>% select(hadm_id, temperature, heartrate, resprate, o2sat, sbp, dbp, acuity, fever) %>% missing
```



2.24 Missing Values Evaluation with Aggregation Plots

The VIM package can also be used to evaluate missing data patterns visually.

VIM stands for _V_isualization and _I_mputation of _M_issing values

The command is `aggr()` for an aggregation plot.

```
library(VIM)

aggr(df, numbers = TRUE, prop = c(TRUE,FALSE))
```

The `aggr` function calculates and displays the amount of missing values by variable and variable combinations that appear to be missing simultaneously.

- `numbers = TRUE` shows the frequency of the sample (right side) based on missing value combinations

Imputation can be used as a way to treat missing values.

2.25 Aggregation Plot Example

```
library(VIM)
```

```
Loading required package: colorspace
```

```
Loading required package: grid
```

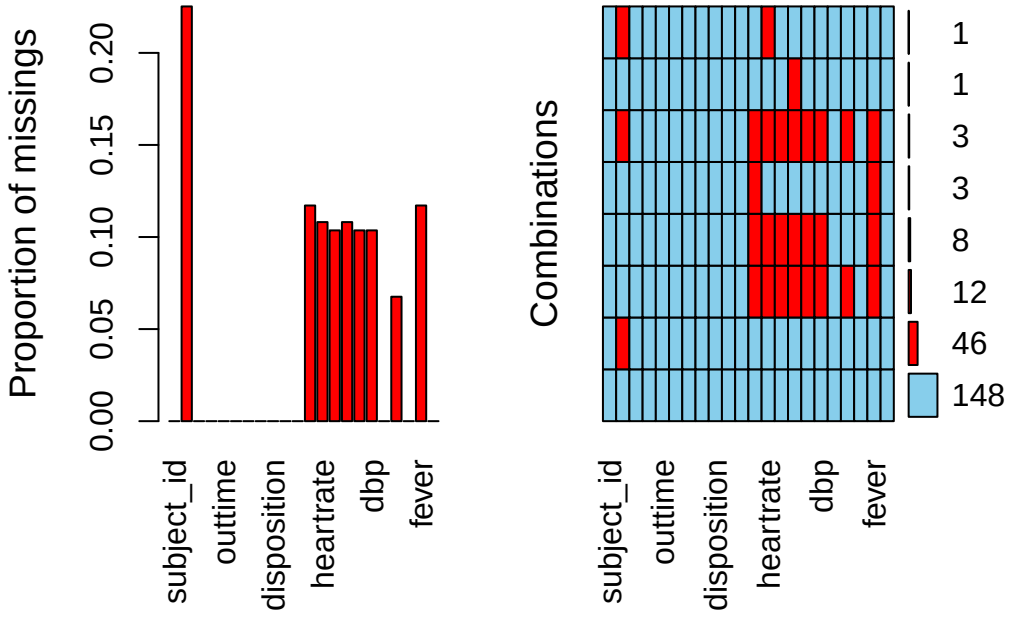
```
VIM is ready to use.
```

```
Suggestions and bug-reports can be submitted at: https://github.com/statistikat/VIM/issues
```

```

Attaching package: 'VIM'
```

```
The following object is masked from 'package:datasets':

    sleep
```

```
aggr(df, numbers = TRUE, prop = c(TRUE,FALSE))
```



2.26 Treatment of Missing Values

Analyses will omit missing values by default.

Missing value presence may denote a bias in measurement or sampling.

First treatment option is to examine presence vs. absence of values as an independent variable.

```
df <- df %>%
  mutate(binaryPvA = if_else(!is.na(var), 1, 0))
```

Second option is to create an imputation method. This is only advised if you know that present data can serve as a good proxy for the missing data.

VIM contains imputation functions like *k-Nearest Neighbors* (`kNN()`) and *Hot Deck Imputation* (`hotdeck()`) to replace missing values with imputed (i.e. simulated) values. This is a very advanced concept and should only be used not be used without significant knowledge of your data.