# Data Analysis Practical Test E15: Data Preparation

TAESEUNG HAHN

https://github.com/tshahn/DataAnalPrac

## COMMON: Load Package

```
pkgs = c("tidyverse", "caret", "stringr", "lubridate")
for (pkg in pkgs) if (!pkg %in% installed.packages()[,1]) install.packages(pkg)
invisible(lapply(pkgs, library, character.only=TRUE))
```

```
── Attaching packages ─────────────────────── tidyverse 1.3.0 ──
✓ ggplot2  3.3.0      ✓ purrr   0.3.4
✓ tibble   3.0.1      ✓ dplyr   0.8.5
✓ tidyr    1.0.2      ✓ stringr 1.4.0
✓ readr    1.3.1      ✓ forcats 0.5.0
── Conflicts ──────────────────────────── tidyverse_conflicts() ──
✗ dplyr::filter()     masks stats::filter()
✗ dplyr::lag()        masks stats::lag()
```

## Question No. 1

제공 데이터 파일: E15Q1_data_raw.csv

- 1-24번 컬럼: Analog Data
- 25번 컬럼: 제품코드 (Binary)
- 26번 컬럼: 불량코드 (Integer with range 1 to 7)

### 1. Download Data

```
colname_url = "http://archive.ics.uci.edu/ml/machine-learning-databases/00198/Faults27x7_var"
data_url = "http://archive.ics.uci.edu/ml/machine-learning-databases/00198/Faults.NNA"

colname_path = "./data/colname_org"
data_path = "./data/data_org"
```

```
download.file(url=colname_url, destfile=colname_path, method="wget")
download.file(url=data_url, destfile=data_path, method="wget")
```

### 2. Load Original Data

```
cname = read_delim(colname_path, delim="\n", col_names=FALSE) %>% pull
data_org = read_delim(data_path, delim="\t", col_names=cname)
```

```
Parsed with column specification:
cols(
  X1 = ←[31mcol_character()←[39m
)

Parsed with column specification:
cols(
  .default = col_double()
)

See spec(...) for full column specifications.
```

### 3. MERGE FAULT COLUMNS INTO ONE

```
fault_cols = data_org %>% select(28:34)
type_cols = data_org %>% select(TypeOfSteel_A300, TypeOfSteel_A400)
analog_cols = data_org %>% select(-c(28:34), -TypeOfSteel_A300, -TypeOfSteel_A400)
```

```
head(analog_cols)
```

| X_Minimum | X_Maximum | Y_Minimum | Y_Maximum | Pixels_Areas | X_Perimeter | Y_Perimeter | Sum_of_Luminosity | Minimum_of_Luminosity | |
|-----------|-----------|-----------|-----------|--------------|-------------|-------------|-------------------|------------------------|---|
| <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | |
| 42 | 50 | 270900 | 270944 | 267 | 17 | 44 | 24220 | 76 | 1 |
| 645 | 651 | 2538079 | 2538108 | 108 | 10 | 30 | 11397 | 84 | 1 |

| X_Minimum | X_Maximum | Y_Minimum | Y_Maximum | Pixels_Areas | X_Perimeter | Y_Perimeter | Sum_of_Luminosity | Minimum_of_Luminosity | |
|-----------|-----------|-----------|-----------|--------------|-------------|-------------|-------------------|-----------------------|---|
| <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | |
| 829 | 835 | 1553913 | 1553931 | 71 | 8 | 19 | 7972 | 99 | 1 |
| 853 | 860 | 369370 | 369415 | 176 | 13 | 45 | 18996 | 99 | 1 |
| 1289 | 1306 | 498078 | 498335 | 2409 | 60 | 260 | 246930 | 37 | 1 |
| 430 | 441 | 100250 | 100337 | 630 | 20 | 87 | 62357 | 64 | 1 |

### WAY 1 (if you are familiar with long/wide form transformation)

```
fault = fault_cols %>%
    pivot_longer(cols=1:7, names_to="Fault", values_to="YN") %>%
    filter(YN==1) %>%
    pull(Fault) %>% as.factor

type = type_cols %>%
    pivot_longer(cols=1:2, names_to="Type", values_to="YN") %>%
    filter(YN==1) %>%
    pull(Type) %>% as.factor
```

### WAY 2 (if you are familiar with linear algebra)

```
fault = as.matrix(fault_cols) %*% as.matrix(seq_along(fault_cols))
type = as.matrix(type_cols) %*% as.matrix(seq_along(type_cols))
```

### WAY 3 (if you are familiar with column/row-wise operation)

```
fault = as.factor(apply(fault_cols, 1, which.max))
type = as.factor(apply(type_cols, 1, which.max))
```

### WAY 4: Gather (substitute for pivot_longer())

### 4. Bind and Export Data

```
data_raw = cbind(analog_cols, SteelType=type, Fault=fault)
```

```
write_delim(data_raw, path="./data/E15Q1_data_raw.csv", delim=",")
```

# Question No. 2

- 공장의 전력 사용량에 대한 데이터 (총 3개 파일)
- 각 데이터 파일에 대한 설명:
    1. E15Q21_usage.csv
    - 900초마다 기록된 900초 단위 전력 총 사용량
    - 1번 컬럼: Datetime (UnixTimestamp)
    - 2번 컬럼: Usage

    2. E15Q22_weather.csv

    - 일자별 평균 기온
    - 1번 컬럼: Date (YYYY-MM-DD)
    - 2번 컬럼: Daily Average Temperature

    3. E15Q23_usage_history.tsv

    - 1분에 2번씩 기록된 각 용도별 전력 누적사용량
    - 1번 컬럼: Time (HH:MM)
    - 2번 컬럼: Weather Class (A/B/C/D)
    - 3-7번 컬럼: 각 용도(A/B/C/D/E)별 전력 누적 사용량

### 1. Generate Time Standard

```
hh = str_pad(00:23, 2, pad='0')
mm = str_pad(00:59, 2, pad='0')
ss = c("00", "30")
time_day = expand.grid(hh=hh, mm=mm, ss=ss) %>%
    arrange(hh, mm, ss) %>%
    mutate(time = paste(hh, mm, ss, sep=":")) %>%
    select(time)
```

```
n_day = 70
time = data.frame(replicate(n_day, time_day)) %>%
```

```
    pivot_longer(cols=seq(n_day)) %>%
    transmute(day = rep(seq(n_day), NROW(.)/n_day),
              time = value) %>%
    arrange(day, time) %>%
    mutate(timestamp_idx = rep(1:6720, each=30),
           timestamp = 1504224000 + (timestamp_idx-1)*900,
           dt = as.Date(as.POSIXct(timestamp, origin="1970-01-01"))) %>%
    arrange(day, timestamp_idx, time)
```

```
head(time)
```

A tibble: 6 × 5

| day | time | timestamp_idx | timestamp | dt |
|-----|------|---------------|-----------|-----|
| <int> | <fct> | <int> | <dbl> | <date> |
| 1 | 00:00:00 | 1 | 1504224000 | 2017-09-01 |
| 1 | 00:00:30 | 1 | 1504224000 | 2017-09-01 |
| 1 | 00:01:00 | 1 | 1504224000 | 2017-09-01 |
| 1 | 00:01:30 | 1 | 1504224000 | 2017-09-01 |
| 1 | 00:02:00 | 1 | 1504224000 | 2017-09-01 |
| 1 | 00:02:30 | 1 | 1504224000 | 2017-09-01 |

## 2. Generate Weather Class

```
wclass = sample(c('A', 'B' ,'C', 'D'), NROW(time), c(0.2, 0.3, 0.4, 0.1), replace=TRUE)
```

## 3. Generate Usage by Time and Purpose

```
set.seed(23)
usage_init = replicate(5, abs(rnorm(NROW(time), 0, 3))) %>%
    as.data.frame(make.names=FALSE) %>%
    transmute_all(~ifelse(. > 0.5, ., 0))
colnames(usage_init) = LETTERS[1:5]
timely_usage_init = cbind(time, wclass, usage_init)
```

## 4. ADJUSTMENT 1: Adjust Usage by Day of the Week

```
set.seed(53)
timely_usage_adj1 = timely_usage_init %>%
    mutate_at(vars(C, E), list(~ifelse(day%%7 == 2, .+1+abs(rnorm(1, 0, 0.1)), .))) %>%  # Saturday Usage Ad
    mutate_at(vars(B, E), list(~ifelse(day%%7 == 3, .+2+abs(rnorm(1, 0, 0.1)), .)))       # Sunday Usage Adju
```

```
timely_usage_adj1 %>% head
```

A data.frame: 6 × 11

| | day | time | timestamp_idx | timestamp | dt | wclass | A | B | C | D | E |
|---|-----|------|---------------|-----------|-----|--------|---|---|---|---|---|
| | <int> | <fct> | <int> | <dbl> | <date> | <fct> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> |
| 1 | 1 | 00:00:00 | 1 | 1504224000 | 2017-09-01 | C | 0.579637 | 3.878765 | 3.3663863 | 2.580289 | 0.7968933 |
| 2 | 1 | 00:00:30 | 1 | 1504224000 | 2017-09-01 | C | 1.304046 | 5.810091 | 0.8243956 | 0.000000 | 0.0000000 |
| 3 | 1 | 00:01:00 | 1 | 1504224000 | 2017-09-01 | C | 2.739801 | 0.000000 | 4.1396554 | 1.272027 | 0.0000000 |
| 4 | 1 | 00:01:30 | 1 | 1504224000 | 2017-09-01 | B | 5.380164 | 1.589156 | 0.9473668 | 3.444810 | 4.3132982 |
| 5 | 1 | 00:02:00 | 1 | 1504224000 | 2017-09-01 | B | 2.989815 | 3.045425 | 2.4339320 | 1.229422 | 6.3342472 |
| 6 | 1 | 00:02:30 | 1 | 1504224000 | 2017-09-01 | B | 3.322471 | 1.747114 | 2.4407186 | 0.000000 | 0.0000000 |

## 5. Generate Average Daily Temperature and the Temperature of a 30-second Unit.

```
set.seed(311)

weather = time %>%
    select(dt) %>%
    unique %>%
    mutate(avg_temp = rnorm(NROW(.), 23, 5))

timely_usage_adj1_tmpr = timely_usage_adj1 %>%
    left_join(weather, by="dt") %>%
    mutate(temp = avg_temp + rnorm(NROW(.), 0, 1))
```

```
head(timely_usage_adj1_tmpr)
```

A data.frame: 6 × 13

| | day | time | timestamp_idx | timestamp | dt | wclass | A | B | C | D | E | avg_temp | temp |
|---|-----|------|---------------|-----------|-----|--------|---|---|---|---|---|----------|------|
| | <int> | <fct> | <int> | <dbl> | <date> | <fct> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> |

| | day | time | timestamp_idx | timestamp | dt | wclass | A | B | C | D | E | avg_temp | temp |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | <int> | <fct> | <int> | <dbl> | <date> | <fct> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> | <dbl> |
| 1 | 1 | 00:00:00 | 1 | 1504224000 | 2017-09-01 | C | 0.579637 | 3.878765 | 3.3663863 | 2.580289 | 0.7968933 | 26.74755 | 25.79651 |
| 2 | 1 | 00:00:30 | 1 | 1504224000 | 2017-09-01 | C | 1.304046 | 5.810091 | 0.8243956 | 0.000000 | 0.0000000 | 26.74755 | 27.37505 |
| 3 | 1 | 00:01:00 | 1 | 1504224000 | 2017-09-01 | C | 2.739801 | 0.000000 | 4.1396554 | 1.272027 | 0.0000000 | 26.74755 | 26.19446 |
| 4 | 1 | 00:01:30 | 1 | 1504224000 | 2017-09-01 | B | 5.380164 | 1.589156 | 0.9473668 | 3.444810 | 4.3132982 | 26.74755 | 27.07863 |
| 5 | 1 | 00:02:00 | 1 | 1504224000 | 2017-09-01 | B | 2.989815 | 3.045425 | 2.4339320 | 1.229422 | 6.3342472 | 26.74755 | 27.14825 |
| 6 | 1 | 00:02:30 | 1 | 1504224000 | 2017-09-01 | B | 3.322471 | 1.747114 | 2.4407186 | 0.000000 | 0.0000000 | 26.74755 | 27.01240 |

## 6. ADJUSTMENT 2: Adjust Usage by Temperature and Purpose

```
set.seed(145)
timely_usage_adj2 = timely_usage_adj1_tmpr %>%
    mutate(A = abs(0.1*temp - A))    # A is dependent on temperature
```

## 7. Create Join Key for 15-minute unit

```
key15 = timely_usage_adj2 %>%
    group_by(timestamp_idx) %>%
    summarise(usage15 = sum(A, B, C, D, E))
```

---

## 8. Create Final Tibbles for the Test

```
timely_usage = timely_usage_adj2 %>% left_join(key15, by="timestamp_idx")
```

```
usage_history = timely_usage_adj2 %>%
    mutate_at(vars(LETTERS[1:5]), ~cumsum(.)) %>%
    mutate(time=substr(time, 1, 5)) %>%
    select(time, wclass, LETTERS[1:5])
```

```
usage = timely_usage_adj2 %>%
    group_by(day, timestamp_idx, timestamp) %>%
    summarise(amount = sum(A, B, C, D, E)) %>%
    as_tibble %>%
    select(timestamp, amount)
```

---

## 9. Check Adjustment

```
timely_usage %>%
    mutate(wday = weekdays(dt)) %>%
    group_by(wday) %>% summarise(wdaysum = sum(A, B, C, D, E))
```

A tibble: 7 × 2

| wday | wdaysum |
|---|---|
| <chr> | <dbl> |
| Friday | 316274.2 |
| Monday | 314674.6 |
| Saturday | 378722.4 |
| Sunday | 435757.8 |
| Thursday | 315223.6 |
| Tuesday | 313612.4 |
| Wednesday | 314866.1 |

```
timely_usage %>%
    group_by(dt, avg_temp) %>%
    summarise_at(vars(LETTERS[1:5]), ~(sum(.))) %>%
    pivot_longer(cols=LETTERS[1:5]) %>%
    group_by(name) %>%
    summarise(c = cor(avg_temp, value))
```

A tibble: 5 × 2

| name | c |
|---|---|
| <chr> | <dbl> |

| name | c |
|------|------|
| <chr> | <dbl> |
| A | 0.730820745 |
| B | 0.080761375 |
| C | 0.178267511 |
| D | -0.009414051 |
| E | 0.170544094 |

## 10. Export Data to Files

```
write_delim(usage, path="./data/E15Q21_usage.csv", delim=",")
write_delim(weather, path="./data/E15Q22_weather.csv", delim=",")
write_delim(usage_history, path="./data/E15Q23_usage_history.tsv", delim="\t")
```