Le profil que l'on exécute avant d'y toucher

Un export neuf est une pièce à conviction tant que vous n'y touchez pas. Le lire d'abord comme du texte brut, inventorier chaque colonne, prouver ou réfuter la clé, et enregistrer le résultat comme trace de ce qui est arrivé.

Pierre Robentz Cassion

Leçon 1 sur 8

Nettoyage et validation des données — Avant de modifier quoi que ce soit

Ce que couvre cette leçon

- Le fichier est une pièce à conviction tant que vous ne l'éditez pas
- Lisez-le d'abord comme du texte
- Ce à quoi un profil doit répondre
- La forme, face à une attente que vous écrivez
- L'inventaire des colonnes
- Les sentinelles ne sont pas encore des valeurs manquantes
- La clé en est-elle une ?
- Les bornes, avant de croire un résumé
- Enregistrez le profil à côté des données
- Comparer cet export au précédent
- La suite

Le fichier est une pièce à conviction tant que vous ne l'éditez pas

- Dans six mois, quelqu'un vous demandera si le chiffre d'août a toujours été ainsi.

Lisez-le d'abord comme du texte

- Lisez une première fois avec tout en chaîne de caractères — Cette lecture sert à regarder, pas à calculer

Lisez-le d'abord comme du texte — En Python

```
import pandas as pd

PATH = "muac-screening-artibonite-2024.v1.csv"
raw = pd.read_csv(PATH, dtype="string", keep_default_na=False)

print(raw.shape)
print(raw.dtypes)
```

Lisez-le d'abord comme du texte — En R

```
library(readr)
library(dplyr)

PATH <- "muac-screening-artibonite-2024.v1.csv"
raw <- read_csv(PATH, col_types = cols(.default = col_character()))

dim(raw)
```

Ce à quoi un profil doit répondre

- **Combien est arrivé ?** Lignes et colonnes, face à ce que vous attendiez.
- **Qu'est réellement chaque colonne ?** Ses valeurs brutes, pas le type deviné.
- **Où sont les trous ?** Par colonne et par site — jamais un chiffre global.
- **La clé en est-elle une ?** Non pas « la colonne existe-t-elle » mais « est-elle unique ».
- **Qu'est-ce qui sort des bornes ?** Face aux limites du secteur, pas à celles des données.
- **Quels codes circulent ?** Chaque valeur distincte de chaque colonne catégorielle.

La forme, face à une attente que vous écrivez — En Python

```
EXPECTED_COLUMNS = [
    "child_id", "commune", "screening_date", "age_months",
    "sex", "muac_mm", "oedema", "outcome",
]

print(f"{len(raw)} rows, {raw.shape[1]} columns")
print("missing columns:", set(EXPECTED_COLUMNS) - set(raw.columns))
print("unexpected columns:", set(raw.columns) - set(EXPECTED_COLUMNS))
```

La forme, face à une attente que vous écrivez — En R

```r
EXPECTED_COLUMNS <- c(
  "child_id", "commune", "screening_date", "age_months",
  "sex", "muac_mm", "oedema", "outcome"
)

cat(nrow(raw), "rows,", ncol(raw), "columns\n")
setdiff(EXPECTED_COLUMNS, names(raw))
setdiff(names(raw), EXPECTED_COLUMNS)
```

L'inventaire des colonnes — En Python

```
def inventory(df):
    rows = []
    for column in df.columns:
        values = df[column]
        counts = values.value_counts(dropna=False)
        rows.append({
            "column": column,
            "blank": int((values == "").sum()),
            "distinct": int(values.nunique(dropna=False)),
            "top": counts.index[0] if len(counts) else None,
            "top_n": int(counts.iloc[0]) if len(counts) else 0,
        })
    return pd.DataFrame(rows)


print(inventory(raw).to_string(index=False))
```

L'inventaire des colonnes — En R

```r
inventory <- function(df) {
  purrr::map_dfr(names(df), function(column) {
    values <- df[[column]]
    counts <- sort(table(values, useNA = "ifany"), decreasing = TRUE)
    tibble::tibble(
      column   = column,
      blank    = sum(values == "" | is.na(values)),
      distinct = dplyr::n_distinct(values),
      top      = names(counts)[1],
      top_n    = as.integer(counts[1])
    )
  })
}

print(inventory(raw), n = Inf)
```

L'inventaire des colonnes

Colonne	Vides	Distinctes	Remarque
age_months	226	55	5,4 % de vides — la leçon 2 demande où
oedema	44	4	quatre valeurs distinctes dans une colonne booléenne

L'inventaire des colonnes

- Quatre valeurs distinctes dans une colonne booléenne — voilà le constat

Les sentinelles ne sont pas encore des valeurs manquantes — En Python

```
print(raw["muac_mm"].value_counts().head())
print("rows coded -99:", int((raw["muac_mm"] == "-99").sum()))
```

Les sentinelles ne sont pas encore des valeurs manquantes — En R

```
raw |> count(muac_mm, sort = TRUE) |> head()
sum(raw$muac_mm == "-99")
```

La clé en est-elle une ? — En Python

```
duplicated_ids = raw["child_id"].duplicated(keep=False)
print("rows sharing a child_id:", int(duplicated_ids.sum()))
print("distinct ids involved:", raw.loc[duplicated_ids, "child_id"].nunique())
```

La clé en est-elle une ? — En R

```
raw |>
  group_by(child_id) |>
  filter(n() > 1) |>
  ungroup() |>
  summarise(rows = n(), ids = n_distinct(child_id))
```

Les bornes, avant de croire un résumé — En Python

```
muac = pd.to_numeric(raw["muac_mm"], errors="coerce")
muac = muac.where(muac != -99)

print(muac.describe())
print(muac.nsmallest(10).tolist())
print(muac.nlargest(10).tolist())
```

Les bornes, avant de croire un résumé — En R

```r
muac <- suppressWarnings(as.numeric(raw$muac_mm))
muac[muac == -99] <- NA

summary(muac)
head(sort(muac), 10)
head(sort(muac, decreasing = TRUE), 10)
```

Les bornes, avant de croire un résumé

- Les sept premières sont des centimètres jamais convertis — un PB de 13,4 cm
- Regardez toujours les dix plus petites et les dix plus grandes valeurs de toute colonne de mesure — Cela coûte une…

Enregistrez le profil à côté des données — En Python (suite)

```
from pathlib import Path
import json

profile = {
    "file": PATH,
    "rows": len(raw),
    "columns": list(raw.columns),
    "blank_by_column": {c: int((raw[c] == "").sum()) for c in raw.columns},
    "distinct_by_column": {c: int(raw[c].nunique()) for c in raw.columns},
    "duplicate_key_rows": int(raw["child_id"].duplicated(keep=False).sum()),
    "categorical_values": {
        c: sorted(raw[c].unique().tolist())
        for c in ["commune", "sex", "oedema", "outcome"]
    },
}
```

Enregistrez le profil à côté des données — En Python (suite)

```
Path("outputs/profiles").mkdir(parents=True, exist_ok=True)
Path("outputs/profiles/muac-2024-q4.json").write_text(json.dumps(profile, indent=2))
```

Enregistrez le profil à côté des données — En R (suite)

```r
profile <- list(
  file    = PATH,
  rows    = nrow(raw),
  columns = names(raw),
  blank_by_column    = sapply(raw, function(x) sum(x == "" | is.na(x))),
  distinct_by_column = sapply(raw, dplyr::n_distinct),
  duplicate_key_rows = sum(duplicated(raw$child_id) | duplicated(raw$child_id, fromLast =
    TRUE)),
  categorical_values = lapply(
    raw[c("commune", "sex", "oedema", "outcome")],
    function(x) sort(unique(x))
  )
)

dir.create(here::here("outputs", "profiles"), recursive = TRUE, showWarnings = FALSE)
jsonlite::write_json(
  profile,
```

Enregistrez le profil à côté des données — En R (suite)

```
  here::here("outputs", "profiles", "muac-2024-q4.json"),
  pretty = TRUE, auto_unbox = TRUE
)
```

Comparer cet export au précédent — En Python

```
previous = json.loads(Path("outputs/profiles/muac-2024-q3.json").read_text())

for column, values in profile["categorical_values"].items():
    was, now = set(previous["categorical_values"][column]), set(values)
    if was != now:
        print(f"{column}: new {sorted(now - was)}, gone {sorted(was - now)}")

growth = (profile["rows"] - previous["rows"]) / previous["rows"]
print(f"row count moved {growth:.1%}")
```

Comparer cet export au précédent — En R

```r
previous <- jsonlite::read_json(
  here::here("outputs", "profiles", "muac-2024-q3.json"),
  simplifyVector = TRUE
)

for (column in names(profile$categorical_values)) {
  was <- previous$categorical_values[[column]]
  now <- profile$categorical_values[[column]]
  if (!setequal(was, now)) {
    cat(column, ": new", setdiff(now, was), "| gone", setdiff(was, now), "\n")
  }
}

sprintf("row count moved %.1f%%", 100 * (profile$rows - previous$rows) / previous$rows)
```

Comparer cet export au précédent

Un profil que vous n'avez pas enregistré est un profil que vous n'avez pas exécuté. Toute sa valeur tient à la possibilité de le rouvrir plus tard.

La suite

- Vous savez désormais que ce registre a perdu 5,4 % de ses âges.

Lire la leçon complète, avec le code exécutable

https://data-analysis.cassion.dev/fr/cours/data-cleaning-and-validation/cleaning-01-profile-on-arrival