Une non-réponse qui n'est pas un accident

Un taux de complétude global masque la seule chose qui compte. Décomposer la non-réponse par site et par semaine, nommer le mécanisme, et chiffrer ce que la suppression des lignes incomplètes fait à votre classement.

Pierre Robentz Cassion

Leçon 2 sur 8

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

Ce que couvre cette leçon

- 5,4 %, ce n'est pas un constat
- Décomposez par groupe avant toute chose
- Puis par période, à l'intérieur du groupe qui ressort
- Nommez le mécanisme, dans des mots que le rapport peut porter
- Chiffrez ce que leur suppression coûte
- Les trois réponses, et le moment où chacune est honnête
- Des lignes manquantes, pas seulement des valeurs
- La suite

5,4 %, ce n'est pas un constat

- Un taux de complétude ne signifie quelque chose qu'une fois qu'on sait où sont les trous — Une non-réponse dispersée...

Décomposez par groupe avant toute chose — En Python

```python
by_commune = (
    muac.assign(missing_age=muac["age_months"].isna())
    .groupby("commune")
    .agg(missing=("missing_age", "mean"), n=("missing_age", "size"))
    .sort_values("missing", ascending=False)
)
print((by_commune["missing"] * 100).round(1))
```

Décomposez par groupe avant toute chose — En R

```r
muac |>
  group_by(commune) |>
  summarise(missing = mean(is.na(age_months)), n = n()) |>
  arrange(desc(missing)) |>
  mutate(missing = round(100 * missing, 1))
```

Décomposez par groupe avant toute chose

Commune	Âge manquant	Lignes
Gros-Morne	17,7 %	361
Anse-Rouge	5,7 %	229
Saint-Michel	5,1 %	335
Dessalines	4,9 %	450
Ennery	3,0 %	263

Puis par période, à l'intérieur du groupe qui ressort — En Python

```
gros_morne = muac[muac["commune"] == "Gros-Morne"].copy()
gros_morne["week"] = gros_morne["screening_date"].dt.to_period("W").dt.start_time

print(
    gros_morne.groupby("week")["age_months"]
    .apply(lambda s: s.isna().mean())
    .sort_values(ascending=False)
    .head()
)
```

Puis par période, à l'intérieur du groupe qui ressort — En R

```r
muac |>
  filter(commune == "Gros-Morne") |>
  mutate(week = lubridate::floor_date(screening_date, "week", week_start = 1)) |>
  group_by(week) |>
  summarise(missing = mean(is.na(age_months)), n = n()) |>
  arrange(desc(missing)) |>
  head()
```

Nommez le mécanisme, dans des mots que le rapport peut porter

- **Manquant complètement au hasard.** Les trous ne dépendent de rien, pas même de la valeur absente. Supprimer ces...
- **Manquant au hasard.** Les trous dépendent de quelque chose d'*observé* — une commune, une semaine, un enquêteur. Les...
- **Manquant non au hasard.** Les trous dépendent de la valeur absente elle-même. Les enfants les plus malades sont ceux...

Nommez le mécanisme, dans des mots que le rapport peut porter

La catégorie n'est pas une subtilité statistique. C'est la différence entre « nous avons supprimé 226 lignes » et « nous avons supprimé 226 lignes, dont 54 d'une seule commune en une seule semaine, ce qui déplace le taux de cette commune de 1,1 point ».

Chiffrez ce que leur suppression coûte — En Python

```
def gam_rate(df):
    assessed = df["muac_mm"].notna() | df["oedema"].notna()
    cases = assessed & ((df["muac_mm"] < 125) | (df["oedema"] == True))
    return pd.Series({
        "assessed": int(assessed.sum()),
        "cases": int(cases.sum()),
        "rate": round(cases.sum() / assessed.sum(), 3),
    })


all_rows = muac.groupby("commune").apply(gam_rate)
complete = muac[muac["age_months"].notna()].groupby("commune").apply(gam_rate)

comparison = all_rows.join(complete, rsuffix="_complete")
comparison["shift"] = comparison["rate_complete"] - comparison["rate"]
print(comparison.sort_values("shift"))
```

Chiffrez ce que leur suppression coûte — En R (suite)

```r
gam_rate <- function(df) {
  df |>
    mutate(
      assessed = !is.na(muac_mm) | !is.na(oedema),
      case     = assessed & (muac_mm < 125 | oedema)
    ) |>
    summarise(
      assessed = sum(assessed, na.rm = TRUE),
      cases    = sum(case, na.rm = TRUE),
      rate     = round(cases / assessed, 3),
      .by      = commune
    )
}

comparison <- gam_rate(muac) |>
  left_join(gam_rate(filter(muac, !is.na(age_months))),
```

Chiffrez ce que leur suppression coûte — En R (suite)

```
          by = "commune", suffix = c("", "_complete")) |>
mutate(shift = rate_complete - rate) |>
arrange(shift)
```

Chiffrez ce que leur suppression coûte

Commune	Toutes lignes	Cas complets	Écart
Gros-Morne	14,5 %	13,4 %	-1,1 pt
Anse-Rouge	15,6 %	16,5 %	+0,9 pt
Dessalines	5,8 %	5,4 %	-0,4 pt
Saint-Michel	7,6 %	7,6 %	0,0 pt

Les trois réponses, et le moment où chacune est honnête

- Les supprimer, et déclarer la suppression — Légitime quand la non-réponse est dispersée et que vous l'avez montré
- Les garder, dans une catégorie à part — Souvent la bonne réponse pour une variable catégorielle, car « non renseigné »…
- Imputer — prudemment, et rarement — Pour une variable servant à *désagréger* plutôt qu'à calculer, remplir l'âge…

Les trois réponses, et le moment où chacune est honnête — En Python

```python
muac["age_imputed"] = muac["age_months"].isna()
muac["age_months_filled"] = muac.groupby("commune")["age_months"].transform(
    lambda s: s.fillna(s.median())
)
```

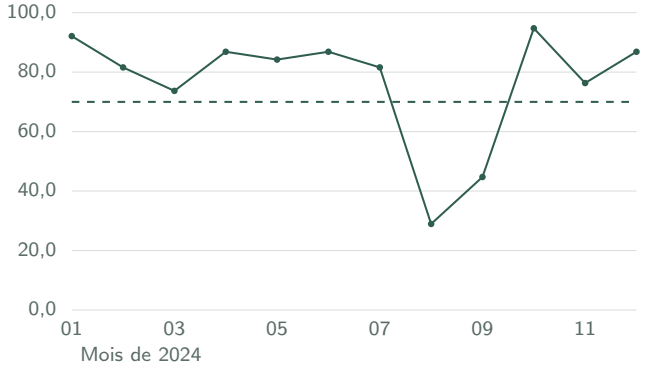
Les trois réponses, et le moment où chacune est honnête — En R

```r
muac <- muac |>
  mutate(age_imputed = is.na(age_months)) |>
  group_by(commune) |>
  mutate(age_months_filled = ifelse(is.na(age_months),
                                    median(age_months, na.rm = TRUE),
                                    age_months)) |>
  ungroup()
```

Les trois réponses, et le moment où chacune est honnête

- La colonne de marquage est l'essentiel — Une valeur imputée qu'on ne distingue plus d'une valeur mesurée a cessé d'être…

Des lignes manquantes, pas seulement des valeurs



Part des formations sanitaires ayant transmis un rapport mensuel au cours de 2024. L'effondrement d'août et septembre bouge de concert dans tout le district, ce qui en fait un événement de rapportage et non une défaillance de formation sanitaire.

Des lignes manquantes, pas seulement des valeurs — En Python

```
vax["reported"] = vax["report_submitted"] == True

reporting_rate = vax.groupby("period")["reported"].mean()
coverage = (
    vax[vax["reported"]]
    .groupby("period")
    .apply(lambda g: g["doses_administered"].sum() / g["target_population"].sum())
)
print(pd.DataFrame({"reporting_rate": reporting_rate, "coverage": coverage}))
```

Des lignes manquantes, pas seulement des valeurs — En R

```
vax |>
  summarise(
    reporting_rate = mean(report_submitted),
    coverage = sum(doses_administered[report_submitted]) /
               sum(target_population[report_submitted]),
    .by = period
  )
```

La suite

- La non-réponse est un trou là où quelque chose devrait être.

Lire la leçon complète, avec le code exécutable

https://data-analysis.cassion.dev/fr/cours/data-cleaning-and-validation/cleaning-02-missingness