Rapport des premières analyses

José Manuel Rodríguez Caballero

March 17, 2025

Contents

1 Introduction

Dans ce rapport, nous présentons les premières analyses requises dans le cadre du projet du cours *STT-7335 Méthodes d'analyse de données*. Nous suivons le plan demandé, qui inclut:

1. **Séparation** du jeu de données en deux sous-ensembles (entraînement et test).
2. **Ajustement** de plusieurs modèles (k-nn, régression linéaire, etc.) sur les observations complètes.
3. **Imputation** des valeurs manquantes selon deux méthodes différentes: moyenne/mode et un modèle multivarié (ici, `mice`).
4. **Ré-ajustement** des mêmes modèles sur les données imputées.
5. **Évaluation** sur le jeu de données test et comparaison des performances (MSE) avec un modèle nul.

Le jeu de données retenu est un ensemble de vers poétiques (*poet, sex, period, date_ of_ birth, date_ of_ death*, etc.) et des scores d'émotions (*anger, anticipation, disgust, ...*). Notre variable cible est `anticipation`. Les scripts `R` ci-joints permettent de reproduire l'intégralité des analyses et de vérifier la *reproductibilité* des résultats.

2 Séparation du jeu de données

Nous avons d'abord séparé le jeu complet (3443 observations) en deux parties:

- **Entraînement (train)**: 75 % des observations, utilisé pour ajuster les modèles.
- **Test**: 25 % restant, afin d'évaluer la performance prédictive.

Le code `R` (`prepare_data`) effectue cette division de manière aléatoire avec une graine fixe. On obtient deux data frames: `train_df` et `test_df`.

3 Gestion des données manquantes

Le rapport requiert d'abord une étape sans imputation (on retire les lignes incomplètes), puis deux approches d'imputation distinctes:

3.1 Retrait des lignes incomplètes

Nous avons défini un vecteur `vars_needed` contenant les variables essentielles au modèle (`anticipation, joy, trust, heterosexual, ...`). Ensuite, nous avons filtré (`filter(complete.cases(...))`) pour conserver uniquement les observations ne présentant aucune valeur manquante. Les modèles ajustés sur ces données seront utilisés comme référence.

3.2 Imputation par la moyenne/le mode

Pour chaque colonne:

- Si elle est numérique, on remplace les valeurs manquantes par la *moyenne* observée chez les non-manquants.
- Si elle est factorielle ou logique, on utilise la *modalité la plus fréquente* (mode).

Nous avons ainsi imputé toutes les colonnes ayant des données manquantes dans `train_df`, puis appliqué la même logique au test (`test_df`), en utilisant les valeurs calculées sur **le train**.

3.3 Imputation par un modèle multivarié (`mice`)

Pour exploiter les *corrélations* entre variables, nous utilisons la librairie `mice`. Celle-ci applique, par défaut, des méthodes d'imputation adaptées au type de chaque variable (p. ex. `logreg` pour les variables binaires, `pmm` pour les variables numériques, `polyreg` pour les facteurs, etc.). Nous avons laissé `mice` choisir automatiquement en précisant un maximum de 5 itérations (`maxit=5`). Nous avons ensuite imputé les valeurs manquantes dans `train_df`, puis refait la même chose pour `test_df`.

4 Ajustement des modèles

Conformément au plan, nous avons ajusté **quatre** modèles distincts pour chacune des versions du jeu de données (sans imputation, imputation moyenne/mode, imputation `mice`). Notons qu'avec `train_knn` (voir ci-dessous), nous n'appliquons pas la réduction de dimension par ACP, alors qu'avec `train_knn_pca`, nous utilisons explicitement l'option `pca` dans l'argument `preProcess`. Cela illustre la différence entre une approche k-nn simple et une approche k-nn appliquée après analyse en composantes principales.

1. **k-nn** (*k nearest neighbors*)

   Modèle mathématique : Pour un nouveau point $\mathbf{x}$, la prédiction du *k-nn* est donnée par :

   $$\hat{y}_{\text{k-nn}}(\mathbf{x}) \;=\; \frac{1}{k} \sum_{i \in \mathcal{N}_k(\mathbf{x})} y_i,$$

   où $\mathcal{N}_k(\mathbf{x})$ représente l'ensemble des indices des k plus proches voisins de $\mathbf{x}$ dans le jeu d'entraînement, et y_i sont les valeurs-cibles correspondantes.

   Formule en R (exemple avec le package `caret`) :

   ```
   train_knn <- train(
   anticipation ~ joy + surprise + trust + joy:trust + heterosexual,
   data       = train_df,
   method     = "kknn",
   trControl  = trainControl(method = "cv", number = 5),
   preProcess = c("center","scale"),
   tuneGrid   = expand.grid(
   kmax     = c(1,3,5,7),
   distance = 2,
   kernel   = "optimal"
   )
   )
   ```

   Remarque : Ici, `preProcess` ne contient pas `pca`, donc aucune réduction de dimension par ACP n'est effectuée.

2. **k-nn + PCA**

   Modèle mathématique : On commence par réduire la dimension des variables explicatives via l'Analyse en Composantes Principales (ACP). Soit

   $$\mathbf{z} \;=\; W^{\top}\big(\mathbf{x} - \overline{\mathbf{x}}\big),$$

   où $\mathbf{z}$ est le vecteur des composantes principales, W est la matrice des vecteurs propres, et $\overline{\mathbf{x}}$ la moyenne des prédicteurs. Ensuite, on applique le *k-nn* à $\mathbf{z}$. La prédiction pour un nouveau point $\mathbf{x}$ devient :

   $$\hat{y}_{\text{k-nn+PCA}}(\mathbf{x}) \;=\; \frac{1}{k} \sum_{i \in \mathcal{N}_k(\mathbf{z})} y_i,$$

où $\mathcal{N}_k(\mathbf{z})$ désigne l'ensemble des k plus proches voisins dans l'espace des composantes principales.

Formule en R (exemple avec `caret` et prétraitement PCA) :

```
train_knn_pca <- train(
anticipation ~ joy + surprise + trust + joy:trust + heterosexual,
data       = train_df,
method     = "kknn",
trControl  = trainControl(method = "cv", number = 5),
preProcess = c("center","scale","pca"),
tuneGrid   = expand.grid(
kmax     = c(1,3,5,7),
distance = 2,
kernel   = "optimal"
)
)
```

3. **Régression linéaire simple**

Modèle mathématique : On modélise la variable réponse y en fonction de p prédicteurs $x_1, \ldots, x_p$ par :

$$y_i \;=\; \beta_0 \;+\; \beta_1\, x_{i1} \;+\; \cdots + \;\beta_p\, x_{ip} \;+\; \varepsilon_i,$$

où ε_i est un terme d'erreur aléatoire centré.

Formule en R (avec `lm`) :

```
lm_model <- lm(
anticipation ~ joy + surprise + trust + joy:trust + heterosexual,
data = train_df
)
```

4. **Régression linéaire + PCA**

Modèle mathématique : On projette d'abord les prédicteurs $\mathbf{x}$ sur leurs composantes principales :

$$\mathbf{z} \;=\; W^{\top}\big(\mathbf{x} - \overline{\mathbf{x}}\big).$$

Si l'on conserve M composantes, alors on ajuste :

$$y_i \;=\; \alpha_0 \;+\; \sum_{m=1}^{M} \alpha_m\, z_{im} \;+\; \varepsilon_i.$$

Les α_m sont estimés par moindres carrés sur l'espace des composantes principales.

Formule en R (exemple avec `caret` et prétraitement PCA) :

```
train_lm_pca <- train(
anticipation ~ joy + surprise + trust + joy:trust + heterosexual,
data       = train_df,
method     = "lm",
trControl  = trainControl(method = "cv", number = 5),
preProcess = c("center","scale","pca")
# Possibilité de tuner pcaComp si désiré
)
```

Nous avons également inclus un **modèle nul** (qui prédit la moyenne de `anticipation` sur l'échantillon d'entraînement) comme point de comparaison.

5 Évaluation sur le jeu de test

Pour chaque configuration (sans imputation, imputation moyenne/mode, imputation `mice`), nous prédisons la variable `anticipation` sur `test_df` et calculons le *Mean Squared Error* (MSE). Les résultats sont rassemblés dans un tableau comparatif (voir Table 1), permettant de juger quelle approche est la plus efficace. Nous regardons en particulier:

- **MSE du modèle nul**: repère de base.
- **k-nn et k-nn + PCA**: le meilleur k est sélectionné par validation croisée.
- **Régression linéaire et régression linéaire + PCA**.

5.1 Observations mal prédites

Pour mieux comprendre les situations dans lesquelles notre modèle k-NN (après imputation `mice`) échoue, nous avons identifié les observations de l'ensemble de test associées à de fortes erreurs absolues de prédiction (c'est-à-dire une différence importante entre la valeur prédite et la valeur réelle de `anticipation`). Quelques constats et pistes d'interprétation :

- **Poèmes très courts ou structure singulière :** Certains poèmes contiennent peu de mots (`n_words` faible) ou présentent des émotions extrêmes (`anger` = 1, `fear` = 2, etc.). Ces configurations atypiques compliquent la tâche du k-NN, puisqu'il manque de "voisins similaires" dans l'échantillon d'apprentissage pour bien estimer `anticipation`.
- **Cumul ou absence d'émotions marquées :** On remarque que plusieurs de ces observations mal prédites cumulent des indicateurs émotionnels (`anger`, `disgust`, `negative`, etc.) ou, à l'inverse, présentent très peu de traits affectifs. Cela peut dérouter le modèle si peu de poèmes partagent la même configuration.
- **Poètes suicidaires ou au style atypique :** Plusieurs auteurs (p. ex. *John Davidson*, *Hart Crane*, *Sara Teasdale*, *Anne Sexton*) sont regroupés sous la variable `suicidal` = `TRUE`. Le registre émotionnel de leurs œuvres peut être particulier et difficile à "faire correspondre" au sein de l'échantillon d'entraînement si ces auteurs y sont peu représentés.

- **Grande différence entre `anticipation` observée et prédite :** Certains poèmes se voient attribuer une valeur réelle d'`anticipation` (par ex. 2) tandis que la prédiction k-NN reste proche de 0 (ou vice versa). Le modèle n'arrive alors pas à capturer les indices textuels et métadonnées qui signaleraient une forte `anticipation`.
- **Hypothèse d'outliers ou de contextes rares :** Enfin, plusieurs poèmes à fortes erreurs partagent des profils uniques ou très singuliers (combinaison de variables extrêmes, style propre à l'auteur, etc.). Une analyse plus poussée (cluster, étude qualitative des textes) permettrait de préciser si ces cas doivent être traités comme "atypiques".

Ces observations mal prédites soulignent les limites du k-NN dans des cas peu fréquents ou très spécifiques. Mieux représenter le style poétique, affiner l'ensemble des prédicteurs ou recourir à des méthodes plus robustes aux outliers pourraient constituer des pistes d'amélioration pour de futurs travaux.

6 Résultats et interprétation

Le tableau ci-dessous illustre un exemple de comparaison des MSE sur le jeu de test, selon la méthode d'imputation et le modèle utilisé. Cet exemple est directement extrait d'une exécution du script `R` fourni.

Stratégie	**Null MSE**	**k-nn**	**LM**	**k-nn + PCA**	**LM + PCA**
Removed Missing	0.159	0.109	0.0905	0.105	0.0914
Mean/Mode	0.151	0.101	0.0808	0.0960	0.0819
MICE	0.151	0.0991	0.0808	0.0965	0.0819

Table 1: Exemple de comparaison des MSE sur l'échantillon de test.

On remarque que la **régression linéaire** (avec ou sans PCA) parvient en général à obtenir des MSE plus faibles que k-nn, et qu'avec l'imputation par `mice`, on obtient de légers gains. Le **modèle nul**, quant à lui, affiche des MSE plus élevés, ce qui témoigne de l'intérêt d'ajouter des prédicteurs.

7 Conclusion et perspectives

À la lumière des différentes expériences menées, plusieurs enseignements peuvent être dégagés :

- **Imputation des données manquantes :** L'utilisation de méthodes d'imputation (moyenne/mode ou via `mice`) s'avère préférable à la simple suppression des observations incomplètes, car elle permet de conserver davantage d'information et, ce faisant, d'améliorer la qualité prédictive. La méthode `mice`, en particulier, peut tirer parti des corrélations entre variables pour une meilleure estimation.
- **Comparaison des modèles :** Parmi les quatre modèles testés (k-nn, k-nn + PCA, régression linéaire simple et régression linéaire + PCA), la régression linéaire se distingue généralement par des MSE plus faibles, surtout lorsqu'elle est associée à l'ACP. Le modèle k-nn reste compétitif, mais semble légèrement pénalisé dans le cas de variables peu informatives ou partiellement corrélées.

- **Effet de la PCA :** L'introduction d'une phase de réduction de dimension (ACP) peut aider dans certains cas à stabiliser les estimations et à diminuer le risque de surapprentissage, surtout si le nombre de prédicteurs est élevé. Cependant, dans la présente étude, la réduction de dimension n'apporte qu'un léger gain pour la régression linéaire ; pour le k-nn, l'intérêt de la PCA peut varier selon le choix de k et la répartition des variables.
- **Modèle nul :** Le modèle nul (prédiction de la moyenne d'`anticipation`) rappelle qu'il est utile de comparer toute approche supervisée à une référence basique. Ici, ce modèle simple est surpassé par toutes les approches qui intègrent des prédicteurs, preuve que les variables explicatives utilisées sont pertinentes.
- **Observations extrêmes ou rares :** Une partie non négligeable des erreurs les plus élevées concerne des poèmes courts ou fortement chargés en émotions négatives/positives, rédigés par des poètes suicidaires ou peu représentés dans le corpus. De futures analyses pourraient se concentrer sur ces profils atypiques (ex. détection d'outliers, augmentation de données, ou raffinement du modèle) pour mieux expliquer et prédire leur valeur d'`anticipation`.

En somme, ces premières analyses montrent que l'ajout de prédicteurs et l'imputation des valeurs manquantes contribuent à améliorer les performances. Les résultats soulignent aussi l'intérêt d'explorer la nature des poèmes "mal prédits" pour renforcer la robustesse du modèle. De prochaines étapes pourraient inclure des méthodes plus avancées (réseaux de neurones, modèles d'ensemble, ou exploitation du texte brut) et une évaluation plus fine de la sensibilité aux outliers et à la variance des prédictions.