

# Corrigé — Examen du 13/11/2025

Pré-traitement des données — Pr. YOUNES DAOUI  
2ème année Génie Informatique et Digitalisation

## Q1. Encodage de `ville_client` (150 modalités) — 2 pts

---

**Réponse :** On propose un **Target Encoding** (ou Frequency Encoding).

**Justification :**

- La variable `ville_client` est **nominale** (pas d'ordre entre les villes)
- Elle contient **150 modalités** → le One-Hot Encoding créerait 150 nouvelles colonnes, ce qui :
  - augmente énormément la dimensionnalité
  - rend le modèle lent et sujet au surapprentissage (*curse of dimensionality*)
- Le **Label Encoding** est inadapté car il crée une fausse relation d'ordre entre les villes
- Le **Target Encoding** remplace chaque ville par la moyenne de la variable cible pour cette ville → une seule colonne, informative, sans explosion dimensionnelle
- Alternative : Le **Frequency Encoding** remplace chaque ville par sa fréquence d'apparition dans le dataset

**Remarque :**

Règle générale : peu de modalités → One-Hot | beaucoup de modalités → Target Encoding / Frequency Encoding

## Q2. Outliers et Scaling — 4 pts

---

### 2.1 Deux techniques de détection des outliers (2 pts)

#### Technique 1 — Méthode IQR (Interquartile Range) :

- **Q1** (1er quartile) : valeur en dessous de laquelle se trouvent 25% des données
- **Q3** (3ème quartile) : valeur en dessous de laquelle se trouvent 75% des données
- **IQR** =  $Q3 - Q1$  : étendue des 50% centraux des données
- Bornes :  $[Q1 - 1.5 \times IQR ; Q3 + 1.5 \times IQR]$
- Toute observation en dehors de ces bornes est considérée comme un outlier

#### Technique 2 — Z-score :

- $\mu$  (mu) : la moyenne de la variable
- $\sigma$  (sigma) : l'écart-type de la variable
- $Z = (X - \mu) / \sigma$  : mesure à combien d'écarts-types une observation se situe par rapport à la moyenne
- Si  $|Z| > 3 \rightarrow$  l'observation est considérée comme un outlier

#### Remarque :

Autres techniques acceptables : boxplot (visuel), méthode de Tukey, Isolation Forest, DBSCAN, LOF.

### 2.2 Scaling avec outliers légitimes (2 pts)

Réponse : On utilise le **RobustScaler**.

$$X_{\text{scaled}} = (X - \text{médiane}) / IQR \quad \text{où } IQR = Q3 - Q1$$

#### Justification :

- Les outliers sont **légitimes** (valeurs correctes mais extrêmes)  $\rightarrow$  on ne veut pas les supprimer
- La **Normalisation Min-Max** est très sensible aux outliers (le min et le max sont tirés par les extrêmes)
- La **Standardisation Z-score** utilise la moyenne et l'écart-type, qui sont aussi sensibles aux outliers
- Le **RobustScaler** utilise la **médiane** et l'**IQR**, qui sont des mesures **robustes** et résistantes aux valeurs extrêmes
- Les outliers légitimes n'auront pas d'impact disproportionné sur le scaling

### Q3. Valeurs manquantes — 4 pts

---

A — **revenu\_annuel** (manquant à 35%, enquête sociale)

Type : **MNAR** (Missing Not At Random)

- Les personnes à **revenu élevé refusent** de déclarer → la probabilité de valeur manquante dépend de la **valeur elle-même** (le revenu)
- C'est la signature classique du MNAR

Méthode de gestion :

1. **Créer une variable indicatrice** **revenu\_manquant** (0/1) — car l'absence de réponse est elle-même une information (elle indique un revenu élevé)
2. **Imputer avec un modèle prédictif** (KNN, régression, Random Forest) en utilisant les autres variables du dataset
3. Ne **pas** utiliser la moyenne ou médiane → elles seraient biaisées vers le bas puisque les hauts revenus sont manquants
4. La distribution étant **asymétrique**, la médiane serait préférée à la moyenne même pour une imputation simple

B — **coupon\_code\_saisonnier** (manquant à 92%, e-commerce)

Type : **MNAR** (ou éventuellement MAR)

- 92% de valeurs manquantes car la majorité des clients **n'ont jamais utilisé de coupon** → la valeur manquante signifie en réalité "pas de coupon utilisé"
- La donnée manquante est liée à sa propre nature

Méthode de gestion :

1. **Recodage de la variable** : créer une variable binaire **a\_utilise\_coupon** (0 = non / 1 = oui)
  - Les 92% de manquants deviennent 0 (pas de coupon)
  - Les 8% restants deviennent 1 (coupon utilisé)
2. C'est plus pertinent que de supprimer ou d'imputer, car le "manquant" a un **sens métier clair** : le client n'a pas de coupon
3. La suppression de la variable serait une perte d'information, car l'utilisation de coupons peut être un bon prédicteur

**A retenir :**

Quand un taux de manquant est très élevé (92%) et que le manquant a un sens métier, la meilleure approche est souvent de recoder la variable plutôt que d'imputer.

## Q4. Interprétation des résultats PCA — 4 pts

### Résultats donnés :

- PC1 = 60%, PC2 = 22%, PC3 = 9%
- 12 capteurs, plusieurs hautement corrélés

### Interprétation :

#### 1. Variance cumulée :

Composantes	Variance cumulée
PC1	60%
PC1 + PC2	82%
PC1 + PC2 + PC3	91%

**2. PC1 (60%)** capture la majorité de l'information. Les 12 capteurs sont fortement redondants — une seule composante résume 60% de la variabilité. Cette composante représente probablement l'état global de fonctionnement de la machine (normal vs anormal).

**3. PC2 (22%)** capture un 2ème axe de variation, possiblement lié à un sous-groupe de capteurs (ex : capteurs thermiques vs capteurs mécaniques).

**4. PC3 (9%)** ajoute un apport marginal. Au-delà de PC3, les composantes n'apportent presque plus d'information utile.

#### 5. Réduction de la dimensionnalité :

- On passe de **12 variables** à **2 composantes** (PC1 + PC2) tout en conservant **82%** de l'information
- Ou à **3 composantes** pour **91%** de l'information
- C'est une réduction massive (12 → 2 ou 3) grâce à la forte corrélation entre les capteurs

#### 6. Utilité pour la détection d'anomalies :

- En projetant les données dans l'espace réduit (PC1, PC2), les ingénieurs peuvent visualiser les données en 2D
- Les points éloignés du cluster principal dans cet espace réduit correspondent aux anomalies (pannes, dysfonctionnements)
- Les composantes résiduelles (PC4 à PC12) avec très peu de variance peuvent aussi servir à détecter les anomalies

## Q5. Feature Engineering — 3 features de ventes — 3 pts

Variables disponibles : `date_vente` , `quantite_vendue` , `prix_unitaire` , `revenu_total` , `id_produit` , `id_magasin`

#	Feature	Formule / Description	Utilité
1	<code>mois_vente / jour_semaine</code>	Extraire le mois et le jour de la semaine depuis <code>date_vente</code>	Capture la saisonnalité (ventes plus élevées le weekend, en décembre, etc.)
2	<code>prix_moyen_par_produit</code>	Moyenne de <code>prix_unitaire</code> groupé par <code>id_produit</code>	Capture le positionnement prix du produit, permet de comparer le prix actuel au prix moyen
3	<code>ventes_totales_par_magasin</code>	Somme de <code>quantite_vendue</code> groupé par <code>id_magasin</code>	Capture la performance du magasin — un magasin très fréquenté vendra plus

Autres features acceptables :

`remise` (si `prix < prix_moyen`), `est_weekend` , `chiffre_affaires_moyen_produit` , `trimestre`

## Q6. Pipeline de traitement de données textuelles — 4 pts

Les étapes principales :

Étape	Rôle
1. Collecte	Récupérer les données textuelles brutes (fichiers, API, base de données)
2. Nettoyage	Supprimer le bruit : caractères spéciaux, chiffres, HTML, ponctuation, URLs. Mettre en minuscules pour uniformiser.
3. Tokenisation	Découper le texte en unités élémentaires (tokens). Ex : "Le chat mange" → ["le", "chat", "mange"]
4. Stop words	Retirer les mots très fréquents mais peu informatifs (le, la, de, est, un...). Ils n'apportent pas de valeur discriminante pour l'analyse.
5. Stemming / Lemmatisation	<b>Stemming</b> : réduire les mots à leur racine ("mangeons" → "mang"). Rapide mais imprécis. <b>Lemmatisation</b> : réduire à la forme canonique ("mangeons" → "manger"). Plus précis mais plus lent.
6. Vectorisation	Transformer le texte en vecteurs numériques. Méthodes : <b>Bag of Words</b> (comptage), <b>TF-IDF</b> (fréquence pondérée), <b>Word2Vec</b> (représentation dense).

Schéma du pipeline :

Texte brut → Nettoyage → Tokenisation → Stop words → Stemming/Lemmatisation → Vectorisation → Modèle ML

A retenir sur TF-IDF :

- TF (Term Frequency) = fréquence du mot dans le document
- IDF (Inverse Document Frequency) =  $\log(N / \text{nombre de documents contenant le mot})$
- TF-IDF =  $TF \times IDF$  → un mot fréquent dans un document mais rare dans le corpus global reçoit un poids élevé