🔏 Justifications Prêtes pour l'Exam

Les phrases exactes à écrire pour avoir tous les points

Pré-traitement des données — Pr. YOUNES DAOUI

📌 Pourquoi ce document ?

En exam, dire "j'utilise le Target Encoding" = **la moitié des points**. Dire **pourquoi** = **tous les points**. Ce document contient les **justifications techniques complètes** à écrire pour chaque type de question. Chaque encadré vert est une réponse prête à recopier.

1. 🏷️ ENCODAGE

Si on te dit : "Variable nominale avec PEU de modalités" (< 10)

Réponse : One-Hot Encoding.

Justification : La variable est nominale — il n'existe aucun ordre naturel entre les modalités. Le Label Encoding attribuerait des valeurs numériques (0, 1, 2...) qui créeraient une **fausse relation d'ordre** que le modèle interpréterait comme une hiérarchie, ce qui biaiserait les résultats. Le One-Hot Encoding crée une colonne binaire par modalité, éliminant tout biais ordinal. Avec un nombre réduit de modalités, l'augmentation de dimensionnalité reste acceptable.

Si on te dit : "Variable nominale avec BEAUCOUP de modalités" (> 10)

Réponse : Target Encoding (ou Frequency Encoding).

Justification : Le One-Hot Encoding créerait N nouvelles colonnes, provoquant une **explosion de la dimensionnalité** (curse of dimensionality), ralentissant l'entraînement et augmentant le surapprentissage. Le Label Encoding est inadapté car la variable est nominale et introduirait une fausse relation d'ordre. Le **Target Encoding** remplace chaque modalité par la moyenne de la variable cible pour cette catégorie → **une seule colonne**, informative, sans explosion dimensionnelle.

Si on te dit : "Variable ordinale" (ex: Bac, Licence, Master)

Réponse : Label Encoding ordonné.

Justification : La variable possède un **ordre naturel** (Bac < Licence < Master < Doctorat). Le Label Encoding respecte cette hiérarchie en attribuant des entiers croissants. Le One-Hot Encoding ferait perdre cette information d'ordre car chaque

modalité serait indépendante. Le Label Encoding permet au modèle de capturer la relation ordinale.

2. 🔍 OUTLIERS

Si on te dit : "Donnez deux techniques de détection"

Technique 1 — IQR : On calcule **Q1** (1er quartile, 25% des données en dessous), **Q3** (3ème quartile, 75%), et **IQR** = Q3 - Q1 (étendue des 50% centraux). Les bornes sont [Q1 - 1.5×IQR ; Q3 + 1.5×IQR]. Toute observation hors de cet intervalle est un outlier. Méthode **robuste** car les quartiles ne sont pas influencés par les extrêmes.

Technique 2 — Z-score : On calcule Z = (X - μ) / σ, où μ = moyenne et σ = écart-type. Z mesure à combien d'**écarts-types** une observation est de la moyenne. Si |Z| > 3 → outlier. Suppose une distribution approximativement normale.

Si on te dit : "Outliers légitimes, quel scaling ?"

Réponse : RobustScaler. **Formule :** X_{scaled} = (X − médiane) / IQR

Justification : Les outliers sont **légitimes** (correctes mais extrêmes) → on ne les supprime pas. La Normalisation Min-Max est très sensible (min et max tirés par les extrêmes). La Standardisation Z-score utilise moyenne et écart-type, aussi **influencés par les outliers**. Le RobustScaler utilise la **médiane** et l'**IQR**, mesures statistiques **robustes** et résistantes aux valeurs extrêmes.

Si on te dit : "Stratégies de traitement"

1. Suppression : Adaptée quand les outliers sont des erreurs de saisie et que le dataset est assez grand.
2. Capping (Winsorizing) : Remplacer par la borne (Q3+1.5×IQR). Conserve l'observation tout en limitant l'impact.
3. Remplacement par la médiane : Plus robuste que la moyenne car non influencée

par les extrêmes.

4. **Transformation log :** Réduit l'écart entre extrêmes et valeurs centrales.

3. ❓ VALEURS MANQUANTES

Si on te dit : "Totalement aléatoire / bug capteur"

Type : MCAR. La probabilité qu'une valeur soit manquante ne dépend ni d'elle-même, ni d'aucune autre variable. Le mécanisme est totalement aléatoire.

Méthode : Imputation par la moyenne (si symétrique) ou médiane (si asymétrique). Si taux faible (< 5%), la suppression est aussi acceptable car elle n'introduit pas de biais en contexte MCAR.

Si on te dit : "Dépend d'une AUTRE variable" (ex: les vieux répondent moins)

Type : MAR. La probabilité de manquer dépend d'une autre variable observée (ex: l'âge), mais pas de la valeur manquante elle-même.

Méthode : Imputation conditionnelle — imputer par le mode/médiane par groupe (ex: par tranche d'âge). L'imputation multiple ou un modèle prédictif (KNN) sont aussi adaptés car ils exploitent les relations entre variables.

Si on te dit : "Les riches cachent leur revenu"

Type : MNAR. La probabilité de manquer dépend de la valeur elle-même. C'est le plus problématique.

Méthode : (1) Variable indicatrice `revenu_manquant` (0/1) — l'absence est elle-même prédictive. (2) Imputer via modèle prédictif (KNN, RF). Les méthodes simples sont biaisées car les valeurs manquantes ne sont pas représentatives.

Si on te dit : "92% manquant / jamais utilisé de coupon"

Méthode : Recodage en binaire (0/1).

Un taux de 92% est trop élevé pour imputer. Le manquant a un **sens métier clair** : le client n'a jamais utilisé de coupon. On crée `a_utilise_coupon` (0=non, 1=oui). Les 92% manquants → 0, les 8% restants → 1. Cela **conserve l'information** tout en éliminant le problème.

4. 📏 SCALING

Si on te dit : "Différence entre Normalisation et Standardisation"

Normalisation (Min-Max) : $X_{norm} = (X - X_{min})/(X_{max} - X_{min})$. Valeurs dans **[0, 1]**. Très sensible aux outliers car min et max sont directement tirés par les extrêmes.

Standardisation (Z-score) : $X_{std} = (X - \mu)/\sigma$. Centre en **0**, écart-type **1**. Ne borne pas les valeurs. Moins sensible que Min-Max mais la moyenne et σ restent influencés par les outliers. Adaptée aux distributions normales.

Si on te dit : "Quel scaling pour KNN / SVM / K-Means"

Ces algorithmes sont basés sur des **calculs de distance** (euclidienne). Sans mise à l'échelle, une variable avec grande amplitude (Revenu 0–500 000) **dominerait complètement** par rapport à une variable petite (Âge 0–80). Le scaling est **obligatoire** pour que chaque variable contribue de manière équitable. On choisit le Z-score en général, ou le RobustScaler si outliers.

Si on te dit : "Scaling pour arbre de décision / Random Forest"

Pas nécessaire. Les arbres fonctionnent en divisant selon des **seuils sur chaque variable individuellement** — ils ne calculent pas de distances. L'échelle absolue n'affecte pas la capacité de l'arbre à trouver le meilleur seuil.

5. 📊 ACP

Si on te dit : "Interprétez les résultats" (PC1=60%, PC2=22%...)

Structure en 4 points :

(1) Variance cumulée : PC1=60%, PC1+PC2=82%, PC1+PC2+PC3=91%. Avec 2 composantes, on conserve 82% de l'information, supérieur au seuil de 80%.

(2) PC1 : Capture 60% à elle seule → forte redondance entre les variables originales. Résume l'axe principal de variation.

(3) Réduction : On passe de N variables à 2-3 composantes tout en conservant l'essentiel. Réduction possible grâce à la forte corrélation.

(4) Perte : Si on ne retient que PC1+PC2 : perte = 100%−82% = 18%.

Si on te dit : "Avantages et limites"

Avantages : Réduction dimensionnelle / Élimination multi-colinéarité (composantes orthogonales) / Visualisation 2D-3D / Réduit le surapprentissage.

Limites : Perte d'interprétabilité (chaque PC = combinaison linéaire) / Ne capture que les relations linéaires / Sensible aux outliers et à l'échelle → il faut standardiser avant.

6. 🔧 FEATURE ENGINEERING

Si on te dit : "Proposez des features et justifiez"

Feature	Justification à écrire
prix_par_m2	Normalise le prix par la surface → permet de comparer des biens de tailles différentes. Indicateur standard du marché.
ancienneté (date2−date1)	Capture l'effet du temps. Un bâtiment ancien = loyer différent d'un neuf.
mois / jour_semaine	Capture les effets saisonniers et cycliques (soldes, fêtes, weekend).
est_weekend (0/1)	Capture une rupture de comportement entre semaine et weekend.
moyenne_par_groupe	Encode le comportement agrégé d'une catégorie (performance du magasin, prix du produit).
log(X)	Réduit l'asymétrie et compresse les extrêmes → relation plus linéaire.

7. 📝 PIPELINE NLP

Si on te dit : "Présentez les étapes et le rôle"

Étape	Justification à écrire
Nettoyage	Supprimer le bruit (ponctuation, HTML, URLs) et convertir en minuscules pour uniformiser. Sans cela, "Chat" et "chat" seraient deux mots différents.
Tokenisation	Découper le texte en unités élémentaires (tokens). Passage du texte continu à une liste de mots exploitable par un modèle.
Stop words	Retirer les mots très fréquents mais non informatifs (le, la, de...). Ils n'apportent aucune capacité de discrimination. Réduit le bruit et la dimensionnalité.
Stemming / Lemma	Réduire les mots à leur forme de base pour regrouper les variantes ("mangeons", "mangé" → même concept). Stemming = rapide mais imprécis. Lemmatisation = précise mais lente.
TF-IDF	Transformer en vecteurs numériques. TF = fréquence dans le doc. IDF = rareté dans le corpus. Un mot fréquent dans un document mais rare globalement reçoit un poids élevé → capture son importance discriminante.