POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

INF8008 - Prétraitement de données

Hiver 2025

Travail Pratique 3

Transformation, distribution et statistiques descriptives

Durée	4 heures
Session	Hiver 2025
Lieu de réalisation	L-3714 - En ligne
Date de Remise	Vendredi 28 février 2025 à 23h59
Taille de l'équipe	**2 personnes**
Pondération	**10%**
Directives particulières	**1.** Tout retard dans la remise du compte-rendu entraîne automatiquement une pénalité comme discuté dans le plan de cours. **2.** Aucun retard de plus de 24 heures ne sera admis, la note de zéro sur dix (0/10) sera attribuée aux étudiants concernés. **3.** Aucun compte-rendu ne sera corrigé, s'il est soumis par une équipe dont la taille est différente **de deux (2) étudiants** sans l'approbation préalable du chargé de laboratoire. La note de zéro sur dix (0/10) sera attribuée aux étudiants concernés. **4.** Soumission du compte rendu par **Moodle** uniquement (https://moodle.polymtl.ca). **5.** Aucune soumission "hors **Moodle**" ne sera corrigée. La note de zéro sur dix (0/10) sera attribuée aux étudiants concernés.

Table des matières

1. Travail à remettre

Vous devez remettre sur Moodle un fichier compressé .zip contenant :

1) Le code : Un Jupyter notebook en Python ou Google Colab qui contient le code tel implanté avec les librairies minimales demandées pour ce TP (Python, Pandas, Matplotlib). Le code doit être exécutable sans erreur et accompagné des commentaires appropriés dans le notebook de manière. Tous vos résultats doivent être reproductibles avec le code dans le notebook. Attention, en aucun cas votre code ne doit avoir été copié de d'ailleurs.

2) Un fichier pdf dont le nom est formé des numéros de matricules des membres de l'équipe, séparé par un trait de soulignement (_). Celui-ci doit représenter votre notebook complètement exécuté sous format pdf (par exemple, obtenu via latex ou imprimé en pdf avec le navigateur). Assurez-vous que le PDF est entièrement lisible. Tutoriel youtube

ATTENTION : Assurez-vous que votre zip ne dépasse pas la taille limite acceptée sur Moodle.

2. Évaluation

Rubriques	**Points**
Implémentation correcte et efficace du code Réponses aux questions de réflexion ou d'analyse	9
Qualité du code (noms significatifs, structure, performance, gestion d'exception, etc.) et commentaires	1
Total de points	10

3. Environnement et outils nécessaires

Google Colab ou Notebook Jupyter.

4. Introduction

Le TP3 porte principalement sur l'échantillonnage et l'imputation, mais nous aborderons tout de même l'agrégation, la tabulation, le remodelage, le pivotement et les statistiques descriptives. Nous survolons l'utilisation de fonctions de base de Pandas et de l'analyse de données numériques.

Dans ce travail, vous aurez à utiliser les données des deux fichiers .csv suivants :

- **patients.csv :** Matrice de données sur les patients
- **appointments.csv :** Matrice de données de 100 000 rendez-vous faits par 943 patients et portant sur 1682 docteurs

Ces données ont été générées synthétiquement par un grand modèle de langage pour approximer les interactions réelles dans le domaine des soins de santé tout en garantissant la confidentialité des données.

Les librairies python qui seront à utiliser pour ce TP sont les suivantes : matplotlib, pandas et numpy.

5. Objectifs du laboratoire

Cette séance de laboratoire a pour but de permettre à l'étudiant(e) de :

- Se familiariser avec les librairies matplotlib, pandas et numpy
- Se familiariser avec l'échantillonnage et l'imputation
- Se familiariser avec l'agrégation, la tabulation, le remodelage et le pivotement
- Se familiariser avec la visualisation de données descriptives et numériques

6. Exercices

Complétez le fichier notebook « TP3.ipynb » fourni afin de répondre aux questions suivantes qui y sont demandées. Vous devrez soumettre vos réponses dans le notebook. Veuillez lire attentivement les consignes concernant les livrables ainsi que le code d'honneur.

6.1 Prédiction de votes par factorisation de matrice

Mise en contexte :
Un principe fondamental des systèmes de recommandations est de prédire les rendez-vous des patients aux docteurs qui n'ont pas encore de rendez-vous et sont présumés des candidats à recommander. On recommande alors les docteurs dont les rendez-vous estimés sont les plus élevés. Une méthode très facile à implémenter consiste à factoriser une matrice de rendez-vous dans un espace à dimensions réduites puis à effectuer le produit des matrices factorisées. Ce produit constitue la prédiction des rendez-vous sur la base de réduction de dimensions.

Le fichier appointments.csv contient 100 000 rendez-vous de 943 patients pour 1682 docteurs. Il faut, dans un premier temps, créer cette matrice et ensuite la factoriser en un produit de matrices dont le rang est plus petit. Mais la matrice originale contient un grand nombre de valeurs manquantes et il est nécessaire d'effectuer une imputation avant la factorisation par des méthodes analytiques, notamment la méthode de décomposition en valeurs singulières que nous utiliserons (SVD).

De plus, il faut aussi effectuer une validation croisée afin de déterminer le bon nombre dimensions pour la factorisation ainsi que s'il est préférable de faire une imputation de valeurs aléatoires, de la moyenne des lignes, ou de la moyenne des lignes et des colonnes.

A)

Mettez les données de "df_appointments " sous la forme d'une matrice de rendez-vous patients × docteurs. **(1.5 points)**

Pour ce faire :

1. Retirez la colonne "date" du DataFrame "df_appointments".
2. Triez les données par "patient.id" et "docteur.id".

3. Regroupez les données par "patient.id" et faites l'agrégation des "niveau.recommandation" pour chaque "doctor.id".
4. Convertissez en matrice.

Affichez les dimensions de la matrice, celle-ci devrait être de "shape" (943, 1682).

B)

Pour factoriser une matrice, elle ne doit contenir aucune valeur manquante. Imputez (remplacez) les valeurs manquantes (valeurs non observées) dans la matrice créée précédemment selon les trois méthodes suivantes. **(2.5 points)**

1. Remplacer les "nan" par des valeurs aléatoires d'une distribution normale (0,1).

2. Remplacer les "nan" par la moyenne des lignes.

3. Remplacer les "nan" par la moyenne des lignes et colonnes.

Affichez la matrice obtenue pour chaque méthode.

C)

Mise en contexte :

Il faut maintenant effectuer une validation croisée à 5 replis en échantillonnant aléatoirement 5 ensembles de rendez-vous parmi les 100 000 rendez-vous observés qui serviront tour à tour d'ensemble de test, les autres étant utilisées pour l'entraînement (estimation des matrices de factorisation).

Pour ce faire, utilisez la fonction validation_croisee_replis() qui effectue une validation croisée par replis de la factorisation SVD et retourne l'erreur quadratique moyenne. Elle utilise une matrice, echantillonReplis, dont chaque ligne contient l'indice des observations de test pour chaque « replis » et qui est créée plus loin.

Utilisez la variable valeurs_observees (obtenue à l'aide de la matrice créer dans la partie "A" ci-dessus) et la fonction random.choice() de numpy.

Affichez les dimensions de l'échantillon, vous devriez avoir (5, 19389). **(1 point)**

D)

Pour chacune des 3 méthodes d'imputation utilisées précédemment, affichez le résultat de la validation croisée. **(0.5 point)**

Les résultats qui seront affichés pour chaque méthode devraient être similaire à ceux-ci :

1. Méthode 1 : array([3.07936543 2.97096366 3.00239858 2.98364498 3.07552508])

2. Méthode 2 : array([1.35600187 1.36579327 1.37056779 1.35480193 1.3561739])

3. 3. Méthode 3 : array([1.34846446 1.36657222 1.3627661 1.3456932 1.34794998])

Affichez les résultats de la validation croisée pour chaque méthode (Méthode 1, Méthode 2, Méthode 3) sur un seul graphique en boîte à moustaches (boxplot).

E)

Explorer le nombre de dimensions qui est optimal par visualisation de la performance prédictive en fonction du nombre de dimensions. Prenez la moyenne par ligne et colonne comme imputation et cet ensemble de dimensions à explorer : (2, 4, 8, 10, 15, 20, 40). N'utilisez qu'un échantillon de données. **(1 point)**

Pour ce faire :

1. Calculez l'erreur quadratique pour chaque dimension

2. Affichez dans un graphique l'erreur quadratique selon le nombre de dimensions.

Note : Le graphique doit utiliser des marqueurs de type 'o' et un style de ligne '-'. N'oubliez pas de nommer les axes et de donner un titre.

6.2 Probabilités bayésiennes

Mise en contexte :

La seconde approche de prédiction est basée sur les probabilités conditionnelles avec des ratios de vraisemblance. Ces calculs nécessitent des tableaux de contingences entre deux variables, la variable à prédire et le facteur de prédiction. Nous voudrons prédire si un individu est susceptible de recommender un docteur en fonction de trois variables, son sexe, son âge et sa condition de santé. Il nous faut donc créer les tableaux suivants.

F)

Créez un tableau des données de votes et des patients.

Pour ce faire :

1. Renommez la colonne "patients.id" du DataFrame appointments par "id ".

2. Faites un left-join de nouveau DataFrame créé avec le DataFrame des patients.

3. Enlevez les colonnes "date" et " zip".

4. Ajoutez une nouvelle colonne "recommande" qui est à True si le "niveau.recommandation" est supérieur à 4.

Affichez le nouveau tableau. **(1 point)**

G)

Créez un nouveau tableau "recommande_gender" qui présente la proportion de patients qui recommande ou qui ne recommande pas un docteur.

Pour ce faire :

1. À partir du DataFrame "appointments", regroupez les données par "docteur.id" et par "gender".
2. Appliquez la fonction "recommande_tbl" sur la colonne "recommande" et réinitialisez l'index.
3. Créez un "pivot table" dont l'index est "docteur.id", la colonne est " gender ", les valeurs sont "recommande" et "recommandePas", puis utilisez la sommation comme fonction d'agrégation.
4. Créez une colonne "LS" à l'aide de la fonction "ratio_vraisemblance" qui contient le résultat du ratio de vraisemblance. Voici les paramètres à donner à cette fonction :
    a. EH: aime_gender[('recommande', 'F')]
    b. EnH: aime_gender[('recommandePas', 'F')]
    c. nEH: aime_gender[('recommande', 'M')]
    d. nEnH: aime_gender[('recommandePas', 'M')]

Note : La valeur LS de votre 1ère ligne devrait être environ 1.147059.

Affichez le nouveau DataFrame. **(1 point)**

H)

Affichez dans un graphique à barres la valeur LS pour les dix premiers "docteurs.id" en utilisant le tableau "recommande_gender". **(0.5 point)**

Note : N'oubliez pas de nommer vos axes et votre graphique.

Bon travail!