Travail 1

IFT3700  
Introduction à la science des données

TRAVAIL PRÉSENTÉ À   
Alain Tapp

13 Novembre 2022  
Par : Hugo Carrier 20197563

# Méthodologie Global

## Préparation des données

Parce que nous utiliser des métriques « 'precomputed' », nous devons faire attention à la manière que nous traitons les données. Premièrement, les similarités et dissimilarités calculées sont sous forme de matricielle et nous avons besoin de les séparer pour les prochaines étapes. Lorsque nous les séparons, nous utilisons la méthode train\_test\_split de python. L’option « shuffle » doit être mis à faut car sinon, il sera difficile de trouver les colonnes liées aux données pour l’apprentissage du modèle avec les données pour tester les données. La raison que ce problème existe est que nous voulons utiliser des données différentes pour l’apprentissage et les tests afin d’assurer une meilleure qualité pour le rapport. Si quelqu’un voulait répété notre procédure, il doit s’assurer de mélanger les données avant de faire les matrice de dissimilarités.

## K plus porche voisin

Pour cette approche, nous avons utilisé plusieurs valeurs de k pour tester notre modèle. Celle-ci nommé hyperparamètre doit être fixer par nous. Nous les avons fait varier entre 2 et 7. Parce que le but était d’étudier la performance des algorithmes, nous n’avons pas tester une deuxième fois avec une valeur de k fixer après une première vague de tests. Cependant, le faire aurait pu augmenter la valeur du rapport, mais nous croyons que la question sera répondue avec seulement un vague d’apprentissage et de test, puis un comparaison des résultats.

## K Médoïdes et partition binaire

Nous avons simplement suivi la méthodologie vue en classes en s’assurant que les données d’apprentissage soit différente de ceux du test.

## Isomap et PCoA

Isomap et PCoA sont deux algorithmes de réduction de dimension. Pour augmenter leur valeur de tests, nous avons ensuite appliquer un k plus proche voisin afin de les regrouper. D’autres méthodes auraient pu être appliquer afin de les regrouper une les données passées dans l’algorithme. Étant donné que les résultats sont passés dans deux modèle, nous avons du séparer les données en 3 groupes. Une partie pour l’apprentissage du premier modèle, une partie pour l’apprentissage du deuxième modèle et finalement le reste pour tester les modèles.

# Adult

## Les données

Pour obtenir les données de l’ensemble adulte, nous avons télécharger celle-ci sur <https://archive.ics.uci.edu/ml/datasets/adult>. Selon la description, le but de ces données est de terminer si une personne fait plus de 50 000 $ par années. L’ensemble de données adultes fournis différente statistique sur des gens. Nous avons 14 colonnes d’information afin de déterminer la tranche de salaire. Donc 15 colonnes au totales. Certaines colonnes sont descriptives et d’autres contiennent des valeurs numériques continues. Les données contiennent quelques trous, mais ces décrite comme assez propre selon la source.

## Notions de similarité proposée.

### Explication de nos choix

Nous avons approché le problème avec le désir de considéré le plus d’information possible sans en inférer. C’est-à-dire que nous voulions inclure l’information des 14 colonnes mais ne pas devoir boucher les trous s’il en a. Après quelque recherche, nous somme tomber sur la méthode Jaccard que nous avons trouvé intéressante. Or, afin de la rendre utilisable avec nos données et en même temps, proposé une version originale de cette similarité, nous avons dû modifier l’approche.

### Explication de la méthodologie

La méthode de Jaccard peut être facilement utiliser lorsque les données sont binaires ou sont des ensembles. Les nôtres ne le sont pas, alors nous devions penser à une idée intéressante pour résoudre ce problème. Initialement, nous avions pensé transformer les colonnes descriptives en binaire. Par exemple, la colonne sexe serait devenu deux colonnes : est homme et est femme. L’étape suivante aurait été de simplement comparer les colonnes descriptives et de divisé la somme des valeur identique par le nombre de colonnes. Après réflexion et quelques essaies, on s’est rendu que nous avions rendu le problème plus que complexe que nécessaire et que les colonnes non modifiées contenant beaucoup de valeurs descriptives fausseraient nos résultats pas leur transformation en de nombreuses sous-colonnes. Au lieu de cela, nous implémenté notre première originalité à l’approche. Nous avons simplement comparé les colonnes si elles sont identiques ou non et nous avons divisé par le nombre total de colonnes.

Cette façon considère que chacune des colonnes apportent la même valeur à notre similarité. Cependant, nous n’avons pas encore traiter les colonnes avec des nombres continues. Afin de les considérer, nous avons pensé à une autre idée originale pour résoudre le problème. Nous avons simplement considéré la distance absolue et rapporter cette valeur en pourcentage.

Pour y arriver, nous avons trouvé le maximum et le minimum de chacune des colonnes pour calculer la distance maximale. Par la suite nous calculons la distance absolue entre nos points puis nous divisons par la distance des extremums de cette colonne. Ce résultat donne une dissimilarité. Si deux chiffres sont identiques, la valeur obtenue sera 0. Pour trouver une similarité, il suffit simplement de faire 1 moins la valeur obtenue. Ainsi nous avons une valeur qui se retrouve entre 0 et 1, ce qui nous permet de considérer l’information de chacune des colonnes à part égale.

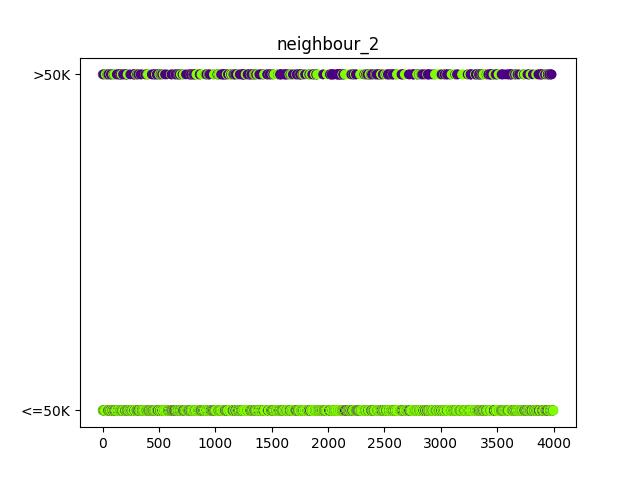
Au final, le résultat total sera entre 0 et 1 dû à la division du nombre de colonnes. Si deux valeurs sont identiques, nous obtiendrons 1 comme similarité, plus elles seront différentes, plus leur similarité tendra vers 0. Avec la similarité et la dissimilarité calculées, nous avons par la suite pue poursuivre l’analyse en l’utilisant avec les diverses méthodologies demander dans le devoir.

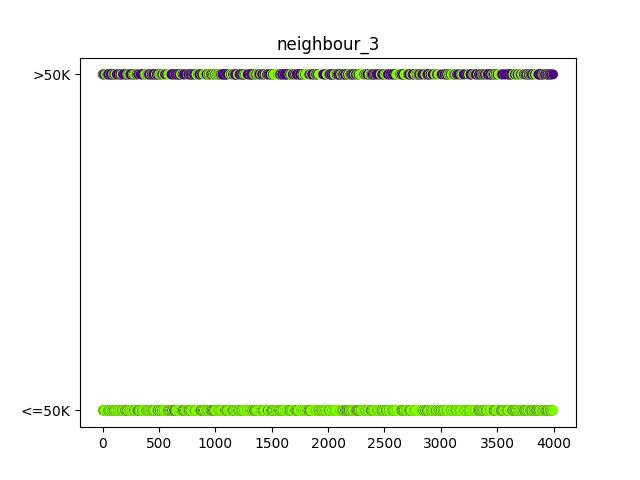
## Présentation des résultats

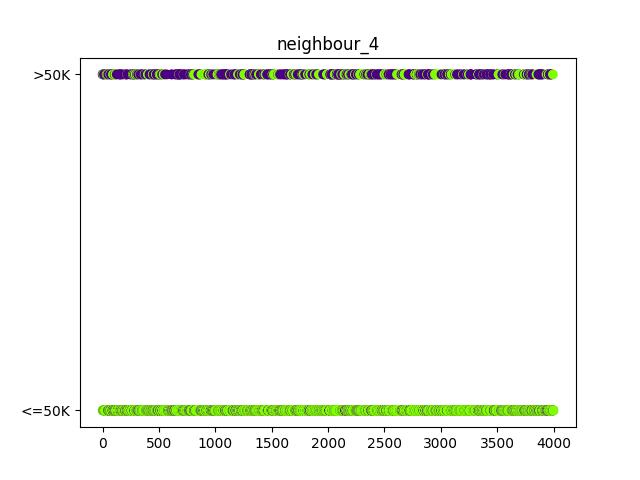
Pour tous les résultats qui suivent, vert signifie que la vraie valeur est"<=50K" et mauve est “>50K".

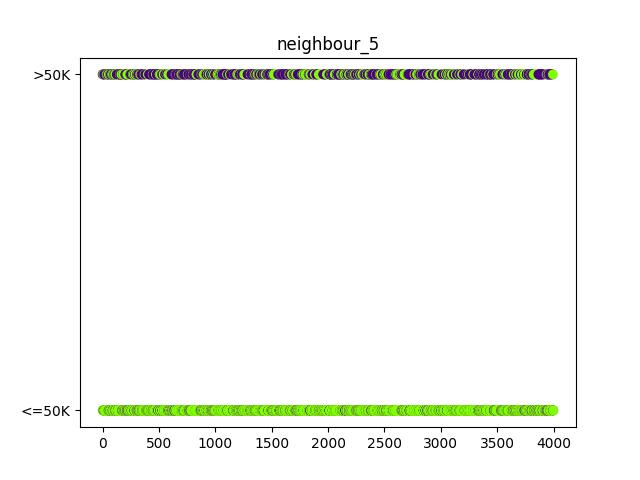
### K- voisin :

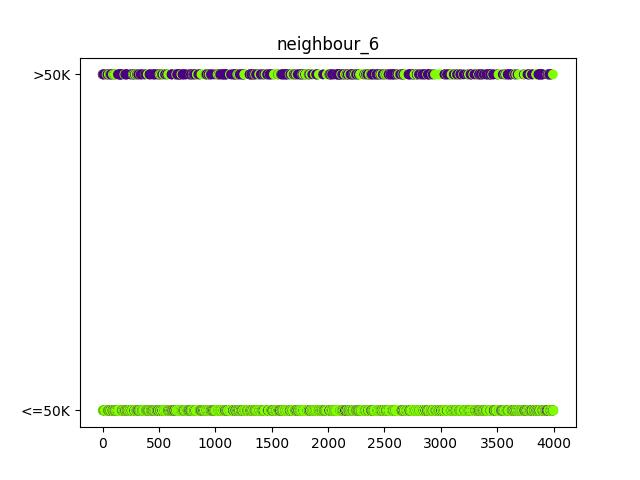
Voici la présentation de différent résultat obtenu.

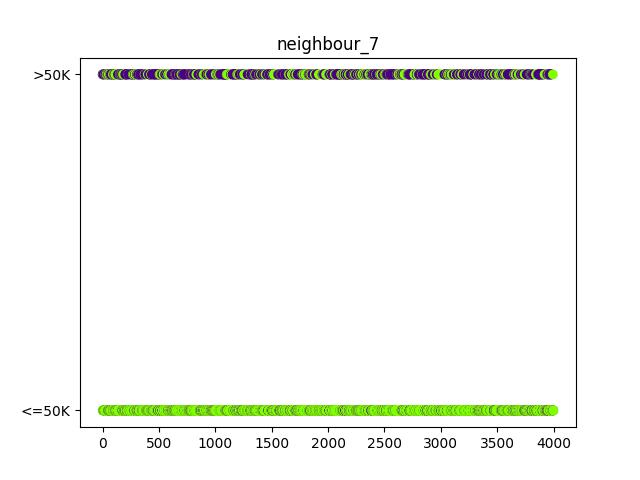






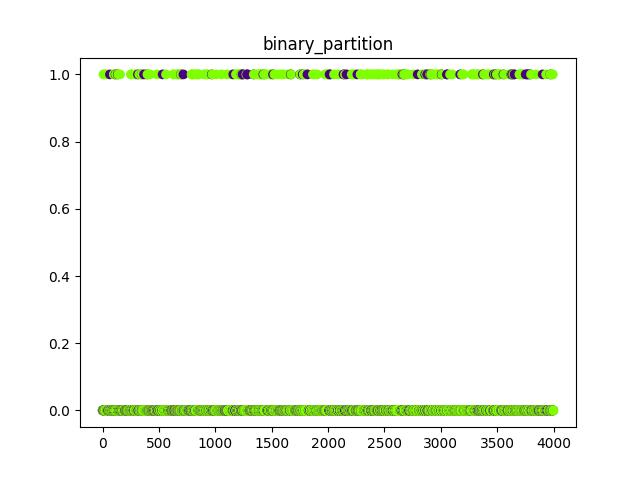


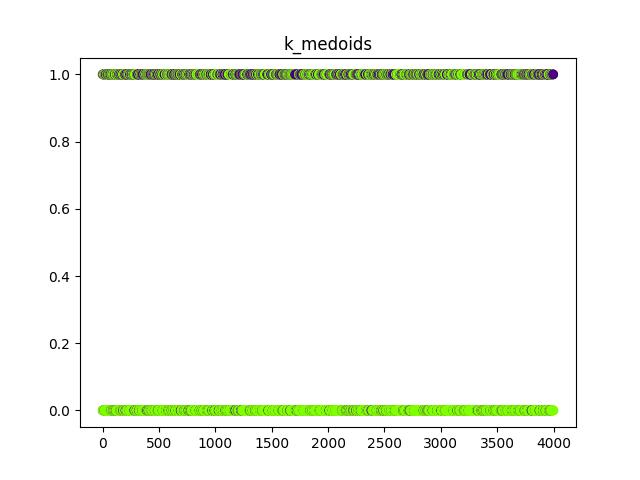




Visuellement nous pouvions penser que nous avons quand même bien réussi à séparer les données. Cependant, lorsque nous faisons l’analyse avec les chiffres, on se rend compte que notre modèle n’apporte pas beaucoup de valeur (vois annexe 4.1) . Nous avons sortie la v-mesure et le nombre d’erreurs. Selon la v-mesure. Le modèle avec k = 7 est le meilleur pour une valeur de la v-mesure d’environs 0.17. Nous avons un total de 768 erreurs sur 4000 valeurs.

### Partition binaire et k médoïdes



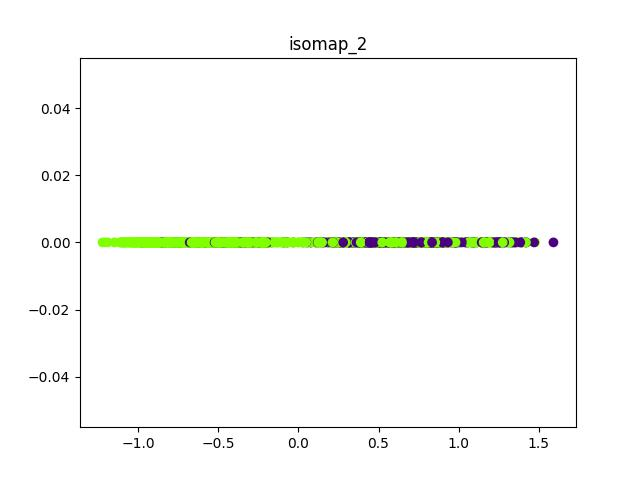


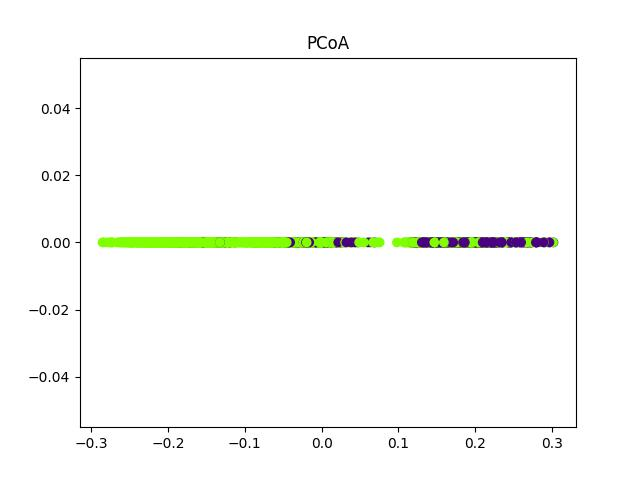
Visuellement, les résultats des deux modèles ne semblent pas être concluant. Comparativement à k voisin, ils ont l’air d’avoir moins bien réussis. La ligne du haut, représentant probablement ceux qui ont plus de 50 000 et elle est parsemer de couleur vert. Rappelons-nous que la couleur verte signifie moins de 50 000.

Lorsqu’on compare avec la v-mesure, nous obtenons environs 0.0001 pour la partition binaire et environs 0.13 pour k médoïdes. L’approche de partition binaire ne semble pas du tout concluante pour ce type de similitude. K médoïdes semble apporter une légère amélioration lorsqu’on évalue sa v-mesure. En revanche, sa quantité d’erreurs totales est de 2828 sur 4000. Ainsi, les k plus proches voisins simple être une bien meilleure méthodologie pour nos données.

### Isomap et PCoA

Isomap et PCoA sont deux approches qui permettent de visualiser les similitudes grâce à une réduction de dimensionnalité.





Visuellement, on remarque quand même une séparation de couleur. PCoA semble avoir moins de couleur mélanger mais cette analyse est assez subjective.

Lorsque nous avons mis ces réductions de dimensionnalité dans un k plus proche voisin, nous n’avons pas obtenus de bon résultat. Les meilleures valeurs de la v-mesure tournent autour de 0.026 (voir annexe 4.1). Ainsi on peut conclure que notre similitude ne sépare pas tout à fait assez bien les données dans leur groupe réspectif.

## Discussion de l’approche

Notre approche est loin d’être parfait. Nous le voyons par les résultats un peu désastreux. Il y a plusieurs problèmes. Premièrement, en python, notre méthodologie prend du temps à être calculer. Nous avons utilisé un ensemble de 5000 données, 1000 pour l’apprentissage et 4000 pour tester le modèle. Il nous prit 30 minutes pour faire la matrice de dissimilarités. Elle pourrait probablement être optimiser dans des langages plus performants, mais pour python ceci est un gros problème. Un autre problème c’est qu’elle traite chacun des colonnes égales. On peut voir ça comme une qualité car ça apporte une simplicité mais c’est aussi un défaut. Il y a probablement certaines colonnes qui apporte plus d’information que d’autre et notre modèle ignore cette possibilité. Il serait aussi probablement possible d’augmenter la qualité si on augmentait la complexité des ressemblances. Dans notre cas, les valeurs descriptives sont égales ou non. Or, certaines valeurs descriptives ont des ressemblances entre elles. Par exemple le niveau l’éducation ont un ordre. La différence entre une maitrise et un bac n’est pas la même qu’entre une maitrise et une quatrième année,

La plupart de ces problèmes sont causé par un choix que nous avons fait. Nous avons désiré un modèle simple et facilement réplicable. Notre méthodologie peut être facilement répliqué et même utiliser sur d’autres types de données. En science de données, c’est un gros avantage car il en devient facile de vérifier les conclusions que nous avons apporté.

Un autre avantage c’est que nous n’avons pas nécessairement besoin de corrigé les données. Le modèle fonctionne correctement lorsque certaine information est manquante. Les informations manquantes sont vues comme des différence mais étant donné que le modèle traite les colonnes à part égal, les résultats sont moins influencé par ces erreurs.

# Annexe

## Sortie pour adults

neighbour\_2 analyse :

v-mesure:0.0987973747543009

Total errors: 866 on 4000

>50K: 183 on 489

<=50K: 683 on 3511

neighbour\_3 analyse :

v-mesure:0.13839457860947285

Total errors: 845 on 4000

>50K: 359 on 862

<=50K: 486 on 3138

neighbour\_4 analyse :

v-mesure:0.1303952922641252

Total errors: 814 on 4000

>50K: 191 on 557

<=50K: 623 on 3443

neighbour\_5 analyse :

v-mesure:0.16277328520633846

Total errors: 787 on 4000

>50K: 307 on 816

<=50K: 480 on 3184

neighbour\_6 analyse :

v-mesure:0.1455506087067676

Total errors: 789 on 4000

>50K: 188 on 576

<=50K: 601 on 3424

neighbour\_7 analyse :

v-mesure:0.16978712431962814

Total errors: 768 on 4000

>50K: 279 on 779

<=50K: 489 on 3221

k\_medoids analyse :

v-mesure:0.12946153649550987

Total errors: 2828 on 4000

>50K: 2068 on 2297

<=50K: 760 on 1703

binary\_partition analyse :

v-mesure:0.00014491851532472935

Total errors: 2881 on 4000

>50K: 2826 on 3760

<=50K: 55 on 240

PCoA\_neighbour\_2 analyse :

v-mesure:0.026849965017929283

Total errors: 803 on 3200

>50K: 178 on 323

<=50K: 625 on 2877

PCoA\_neighbour\_3 analyse :

v-mesure:0.026849965017929283

Total errors: 803 on 3200

>50K: 178 on 323

<=50K: 625 on 2877

PCoA\_neighbour\_4 analyse :

v-mesure:0.026849965017929283

Total errors: 803 on 3200

>50K: 178 on 323

<=50K: 625 on 2877

PCoA\_neighbour\_5 analyse :

v-mesure:0.026849965017929283

Total errors: 803 on 3200

>50K: 178 on 323

<=50K: 625 on 2877

isomap\_neighbour\_2 analyse :

v-mesure:0.022119295941346956

Total errors: 821 on 3200

>50K: 195 on 339

<=50K: 626 on 2861

isomap\_neighbour\_3 analyse :

v-mesure:0.022119295941346956

Total errors: 821 on 3200

>50K: 195 on 339

<=50K: 626 on 2861

isomap\_neighbour\_4 analyse :

v-mesure:0.022119295941346956

Total errors: 821 on 3200

>50K: 195 on 339

<=50K: 626 on 2861

isomap\_neighbour\_5 analyse :

v-mesure:0.022119295941346956

Total errors: 821 on 3200

>50K: 195 on 339

<=50K: 626 on 2861