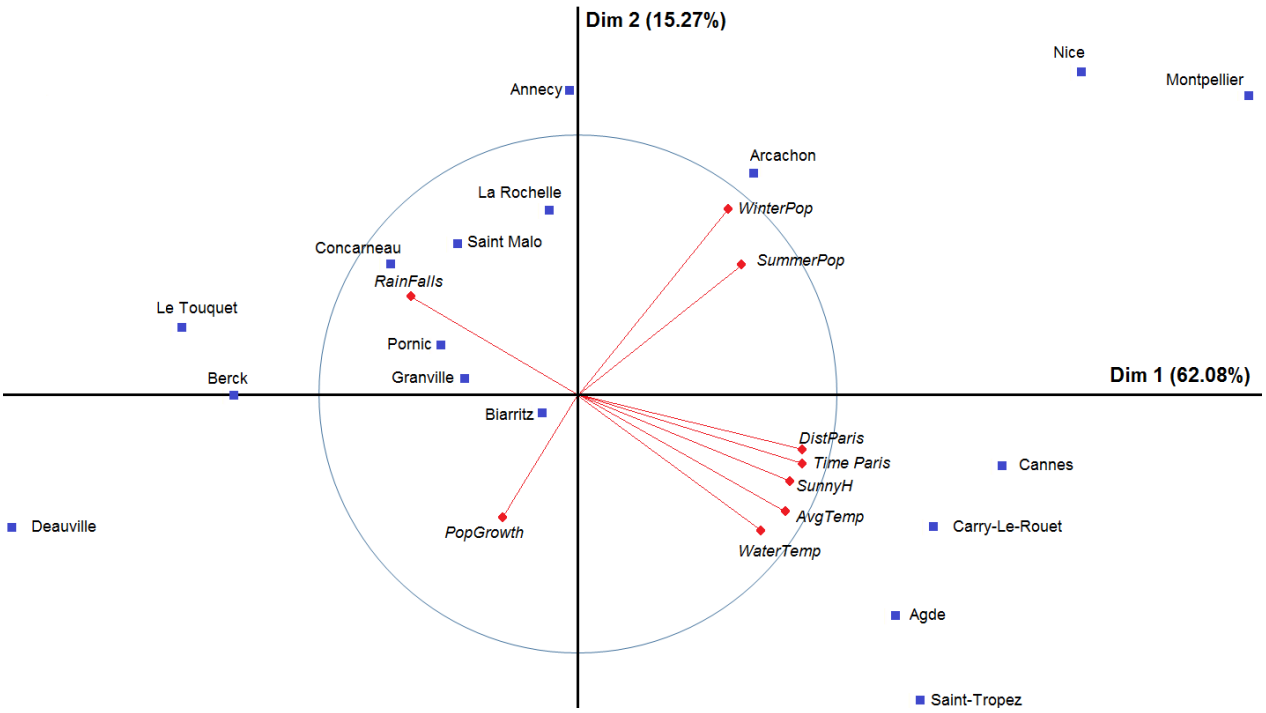
Analyse de données - TP 4 & 5

ISEP – 14 et 21 Mars 2022

Instructions : Préparez un rapport incluant le code source et vos résultats, et déposez-le sur Moodle. Pas plus de 2 personnes par groupe. N'oubliez pas de mettre les 2 noms sur le rapport, ou de faire 2 rendus.

A Interprétation d'un cercle de corrélation

Dans cet exercice, nous vous proposons de visualiser une projection des caractéristiques importantes de plusieurs stations balnéaires françaises. Le résultat est présenté ci-après.



— WinterPop : la population hivernale (sans les touristes)
— SummerPop : la population estivale (avec les touristes)
— PopGrowth : l'augmentation de la population (en pourcentage) en comparant décembre et juillet.
— RainFalls : les précipitations moyennes pendant la période estivale (en mm)
— DistParis : la distance depuis Paris (en km)
— TimeParis : le temps de trajet depuis Paris (en h)
— SunnyH : le nombre moyen d'heures d'ensoleillement sur le moins de juillet (en h)

— AvgTemp : la température moyenne en juillet (en degrés Celsius)
— WaterTemp : la température moyenne de l'eau en juillet pour la baignade (en degrés Celsius)

Répondez aux questions suivantes :

1. Que représentent les pourcentages sur les axes des abscisses et des ordonnées ? Commentez.
2. En vous basant uniquement sur la Figure, répondez aux questions suivantes en justifiant vos réponses :
   A. Quelle est la ville où l'eau est la plus chaude en été ?
   B. Quelle est la ville dont la population augmente le plus entre l'hiver et l'été ?
   C. Vrai ou Faux : les attributs "DistParis" et "TimeParis" sont redondants.
   D. Dans les données originales, c'est la ville d'Annecy qui a le plus de millimètres de pluie en juillet. La figure confirme-t-il cette information ? Si oui justifiez. Si non, expliquez comment ça peut être possible.
   E. Vrai ou Faux : ce sont les villes ayant une faible population hivernale qui ont la plus forte augmentation de population en été.
   F. Vrai ou Faux : l'ensoleillement augmente quand on s'éloigne de Paris.
   G. Vrai ou Faux : la population diminue à Annecy pendant la période estivale.
   H. Quelles sont les 2 villes les plus similaires ?

B ACP sur les données Iris

Les Iris de Fisher sont un jeu de données très connu décrivant 150 fleurs de type "Iris" à partir des caractéristiques de leurs pétales et de leurs sépales. Ces données contiennent 3 classes pour 3 espèces de fleurs (50 de chaque) : Iris Virginica, Iris Versicolor et Iris Setosa. L'une des trois classes est linéairement bien séparable, les 2 autres non. L'objectif de cet exercice est d'étudier la visualisation de ces données à l'aide de l'ACP.

1. Récupérez les données iris à avec la bibliothèque panda. Vous pouvez aussi utiliser directement les données Iris de panda.
2. Extrayez la liste des attributs à partir de la fonction *tolist()* de l'attribut *columns* de votre jeu de données.
3. Retirez et stockez à part la classe de données de votre liste d'attribut.
4. Normalisez les données afin qu'elles soient centrées et réduites. Pour ce faire, vous pouvez utiliser le package *StandardScaler* de sklearn, puis la fonction du même nom.

   ```
   #Normalize data
   from sklearn.preprocessing import StandardScaler
   scaler = StandardScaler()
   data = scaler.fit_transform(data[liste_attributs])
   ```

5. En suivant les étapes suivantes, utilisez l'ACP pour projeter et visualiser vos données en 2 dimensions :

— Initialisez l'algorithme de l'ACP, puis projetez votre jeu de données dans le nouvel espace de l'ACP (voir code ci-dessous).
— Récupérez et affichez le pourcentage de variance expliquée des 2 premières composantes dans les attributs de votre objet *pca*.
— Affichez les 2 premières composantes sous la forme d'un nuage de point pour visualiser les résultats. Vous utiliserez les classes des iris que vous aviez stockées à part pour définir la couleur de vos points (paramètre *label* de la fonction *scatter*).
— Commentez.

```
#4 PCA code
pca = PCA(n_components=4)
#we initialiaze PCA to compute 4 components
data_proj = pca.fit_transform(data[liste_attributs])
#we project the data on the 4 new components (but we will display only 2)
```

6. Écrivez une fonction permettant de tracer un cercle de correlation comme celui vu en cours (vous pouvez vous aider d'internet, mais citez vos sources et commentez votre code). A partir de la figure obtenue, que pouvez-vous dire sur la correlation entre les différents attributs de ces données ?

C Données golub : ACP vs MDS vs LLE vs Isomap vs t-SNE

On va maintenant s'intéresser aux données génétiques du dataset "golub". Ces données contiennent les niveaux d'expressions de 7129 gènes prises sur 72 échantillons. Chaque échantillon est associé à une variante de la leucémie, AML(25) et ALL(47). Notre objectif est de visualiser les 72 échantillons sur un plan en 2D.

C.1 Partie 1

1. Chargez le jeu de données *Golub_data* en faisant très attention aux paramètres d'ouverture. Que remarquez vous par rapport au descriptif ? Faites les transformations si nécessaire, puis normalisez les données.
2. Ouvrez maintenant les labels dans le fichier *Golub_class2* en faisant aussi attention aux paramètres.
3. En vous inspirant de l'exercice précédent, réalisez une ACP sur les données *Golub* et visualisez le résultat. Projetez les labels sur votre ACP. Commentez. Remarque : Les données golub ayant de nombreux attributs, il est recommandé de ne calculer que les 1ère composantes principales.
4. Réalisez une MDS sur ce jeu de données en utilisant la fonction **MDS** du package du même nom. Qu'observez-vous par rapport aux résultats obtenus avec l'ACP ?
   — Afin de pouvoir comparer avec vos camarades, nous vous recommandons de fixer le paramètre *random_state* de l'algorithme MDS à 0.
5. Concluez sur l'efficacité de ces deux méthodes pour ces données.

C.2 Partie 2

1. Analysez maintenant la fonction **Isomap** du package du même nom. Appliquez cette fonction sur les données *Golub* en faisant varier le nombre de voisins à 4, 8, 10, 13, 16 et 20. Remarque : la fonction *subplot* peut être pratique pour comparer les résultats pour les différentes valeurs de voisinage.
2. En utilisant le package *LocallyLinearEmbedding* de sklearn, appliquez la fonction LLE sur les données *Golub* en faisant varier le nombre de voisins à 3, 5, 8, 10, 12 et 15. Analysez les résultats et déterminez le(s) meilleur(s) paramètre(s) de voisinage.
3. En utilisant le package *TSNE* de sklearn, appliquez la fonction du même nom sur les données *Golub*. En comparant les résultats avec ceux de l'algorithme LLE, essayez de trouver une bonne valeur pour le paramètre de perplexité.
4. Concluez sur ces différentes méthodes.

D Données Digits : ACP vs t-SNE

Le fichier "digits.csv" [1] contient un célèbre jeu de données de la famille des MNIST, et qui décrit des chiffres dessinés par 43 personnes au format bitmap 32x32. Les bitmaps 32x32 sont divisés en blocs de 4x4 pixels dans lesquels on compte le nombre de pixel contenant de l'encre (pour dessiner le chiffre). On obtient ainsi une matrice de taille 8x8 donc chaque élément est un entier entre 1 et 16. C'est cette matrice qui est codée sur les 64 premières colonnes du fichier, suivie du nombre dessiné (entre 0 et 9) dans la 65ème colonne.

1. Chargez le fichier *digits.csv* en faisant attention aux paramètres d'ouverture. Vérifiez le format des données, et stockez la 65ème colonne avec la classe dans une variable à part. Remarque : Comme pour les iris, ces données peuvent aussi être importées directement depuis sklearn.
2. En vous inspirant de l'exercice précédent, réalisez une ACP sur ces données en utilisant la classe (le chiffre dessiné) comme couleur. Les classes sont-t-elles bien séparées ?
3. En utilisant le package *TSNE* de sklearn, appliquez maintenant la fonction TSNE sur vos données. Il y a deux paramètres importants : les données et la perplexité. Rappelez la définition de la perplexité pour l'algorithme t-SNE, puis testez votre algorithme pour les valeurs 10, 50 et 100. Visualisez ensuite les résultats et comparez avec l'ACP. Commentez. Remarque : t-SNE peut mettre du temps à s'exécuter, ceci est normal.

E Données Alon

Ces données connectées par Alon et al. contiennent 2000 gènes testé pour 62 patients atteint d'un cancer du colon : 40 diagnostiqués positifs et 22 patients sains.

1. `https://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits`

— Chargez les données *Alon*, décrivez-les et ré-appliquez les approches vues dans l'exercice précédent. Commentez.