Université Claude Bernard Lyon 1
M2 Intelligence Artificielle

TP ANS

Thomas Ranvier

Encadrant
Haytham Elghazel

11 décembre, 2019

1 Réduction de dimensions et Visualisation des données

1.1 Dataset ville.csv

La première étape est de normaliser les données, pour cela on utilise un standard scaler :

```
SS = preprocessing.StandardScaler()
SS.fit(X)
X_norm = SS.transform(X)
```

Une fois que la normalisation est faites nous pouvons réaliser une analyse en composantes principales :

```
pca = decomposition.PCA(n_components=.9)
pca.fit(X_norm)
X_pca = pca.transform(X)
```

On souhaite conserver un minimum de 90% de l'information initiale. Pour cela, on défini le paramètre *n_component* à .9, la PCA est ensuite réalisée automatiquement de manière à conserver 90% des informations. Nous pouvons ensuite afficher dans un graphe les deux principales composantes extraites :

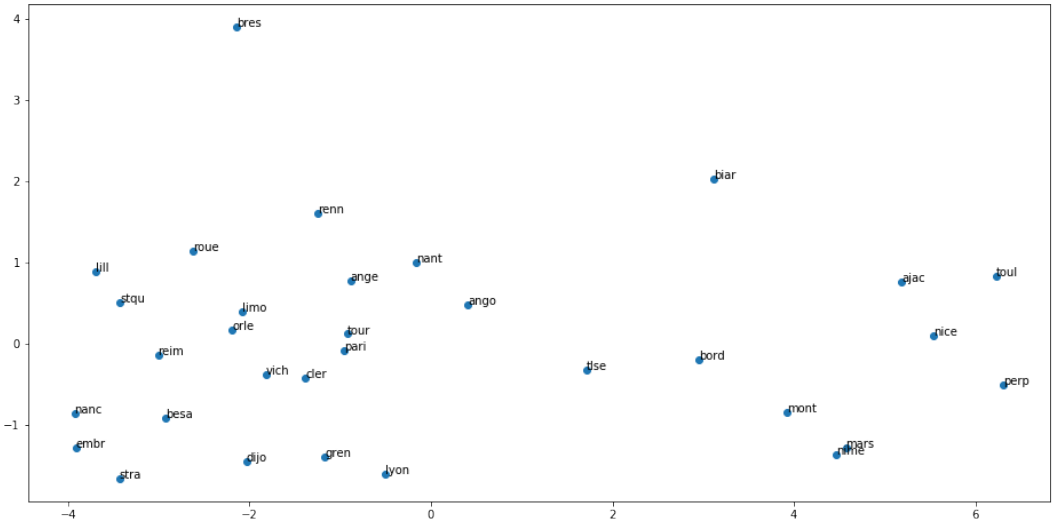

FIGURE 1 – Nuage de points des deux principales composantes extraites

A partir de ces informations nous pouvons maintenant en donner une interprétation. Pour aider cette interprétation j'ai représenté les poids de la PCA sous forme d'un tableau :

Column	x	y
janv	0.27151028270241	0.39933494407140435
fev	0.2884616402750767	0.2990718350740592
mars	0.3010810978605601	0.1294305108387102
avril	0.3035417432572752	-0.11530598159935421
mai	0.28353088840500135	-0.32314291387672084
juin	0.2784190723344509	-0.35846762578072205
juil	0.27290295393050706	-0.388796389848984
aout	0.2875777376958619	-0.3010133037803619
sept	0.3047202218103136	-0.11231622113872056
oct	0.30385479863119197	0.1224922600019204
nov	0.2924280754720525	0.2626946088194898
dec	0.27295490017747326	0.3869368750226594

FIGURE 2 – Poids extraits

Dans cette table on peut voir les poids des axes X et Y pour chaque colonne du fichier csv. En vert sont les poids positifs, en rouge les poids négatifs et en gris sont les poids trop bas pour avoir un fort impact. En effet, il n'est pas nécessaire de considérer les poids ayant une valeur absolue inférieur à $\frac{1}{\sqrt{12}}$ car ces derniers ont peu d'impact. Dans cette formule, 12 est le nombre de colonnes du fichier csv et donc, en l'occurrence, de mois dans l'année.

1.1.1 Axe X

Pour cet axe on peut voir que les poids sont tous positifs et que les valeurs sont très similaires. Nous pouvons donc en déduire que les villes les plus hautes sur cet axe sont des villes dans lesquelles les températures sont globalement les plus élevées. Inversement, les villes les plus basses sur l'axe sont les villes avec les températures les plus basses sur l'ensemble de l'année.
On peut confirmer cette interprétation du fait que les villes du Sud sont hautes sur l'axe et les villes du Nord sont basses sur cet axe.

1.1.2 Axe Y

Pour cet axe on peut voir que les mois de mars, avril, septembre et octobre présentent des poids trop bas pour avoir un fort impact.
On voit que les mois d'été présentent des poids négatifs et que les mois d'hiver présentent des poids positifs. On peut en déduire que les villes les plus basses sur cet axe sont des villes présentant des températures plus hautes que la moyenne en hiver. Inversement, les villes les plus haute sont celles présentant des températures élevées en été.
On peut maintenant comprendre que les mois de mars, avril, septembre et octobre ont peut d'impact sur cet axe puisque ce sont des mois d'automne et de printemps et donc de transition entre hiver et été.

1.2 Dataset crimes.csv

On exécute les mêmes étapes que précédemment pour obtenir le graphe suivant :

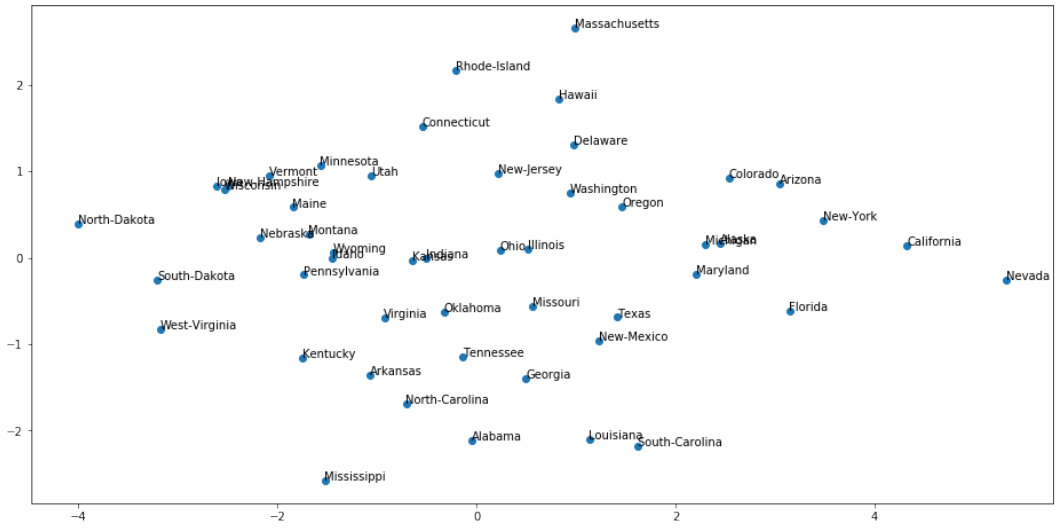

FIGURE 3 – Nuage de points des deux principales composantes extraites

On visualise à nouveau les poids extraits sous forme de table :

Column	x	y
Meutre	0.3002791578316997	-0.6291744430112162
Rapt	0.43175936096941936	-0.16943511837816946
Vol	0.39687549301416375	0.04224697634114735
Attaque	0.39665170032111596	-0.34352815149910343
Viol	0.4401572099736322	0.20334058734057459
Larcin	0.3573595318423982	0.4023191210395476
Auto_Theft	0.29517680934298646	0.5024209325110738

FIGURE 4 – Poids extraits

Les codes couleurs sont les mêmes que précédemment.

1.2.1 Axe X

On voit que dans cet axe les poids sont tous positifs et ont tous un impact assez fort.
Globalement, les états les plus hautes sur cet axe seront donc celles présentant le plus de crime et les plus basses celles présentant le moins de crimes.
Cela signifie que le nombre de crimes au Nevada est très élevé, alors qu'il est très bas au North-Dakota.

1.2.2 Axe Y

Pour cet axe, on peut voir que les états les plus hauts présentent en proportion plus de crime de type larcin, vol de voiture et viols. Les états les plus bas présentent en proportion plus de crimes violents tels que des meurtres, enlèvements et attaques.
Le Mississippi est l'état dans lequel les crimes sont les plus violents et le Massachusetts l'état dans lequel les crimes sont les moins violents.

1.3 Dataset 50_startups.csv

On exécute les mêmes étapes que précédemment pour obtenir le graphe suivant :

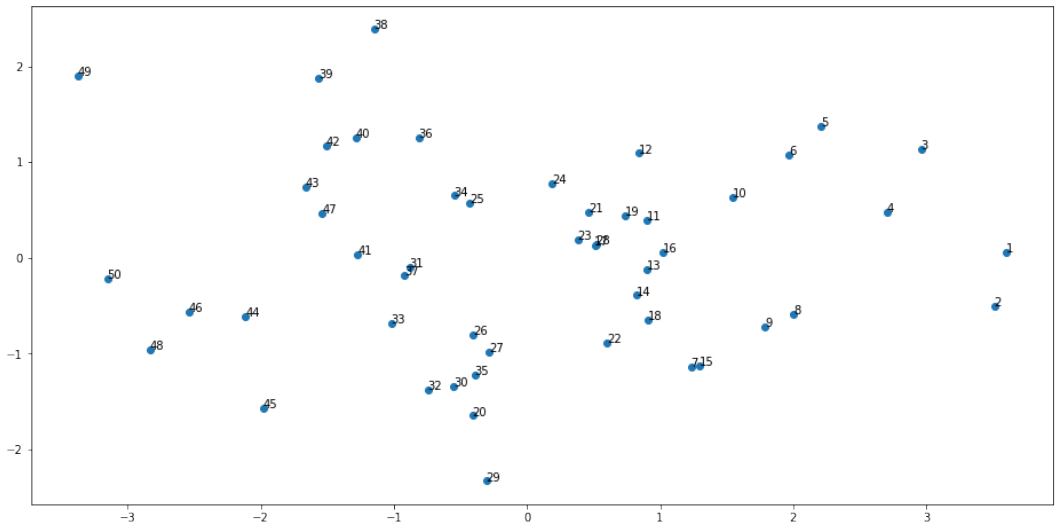

FIGURE 5 – Nuage de points des deux principales composantes extraites

On visualise à nouveau les poids extraits sous forme de table :

Column	x	y
Depenses R&D	0.5934785451377884	-0.04048087265631398
Depenses Administration	0.14737886162577352	-0.950513138115713
Depenses Marketing Spend	0.5206469366239095	0.30797097610937324
Benefice	0.5958099151500958	0.006320687671744419

FIGURE 6 – Poids extraits

Les codes couleurs sont les mêmes que précédemment.

1.3.1 Axe X

Encore une fois cet axe n'a que des poids positifs ayant un fort impact, les startups hautes sur l'axe ont donc de fortes dépenses et bénéfices.
On peut donc considérer que les startups en meilleure 'santé' sont celles vers la droite, car elles ont d'importantes dépenses et bénéfices, alors que celles sur dans le négatif on forcément un bénéfice négatif.

1.3.2 Axe Y

Sur cet axe les startups hautes ont des dépenses marketing élevées et les basses ont des dépenses administratives élevées.

2 Clustering

2.1 Utilisation de KMeans

La première méthode utilisée est le KMeans, avec trois clusters :

```
k_means = cluster.KMeans(n_clusters=3)
k_means.fit(X_pca)
clustering = k_means.labels_
```

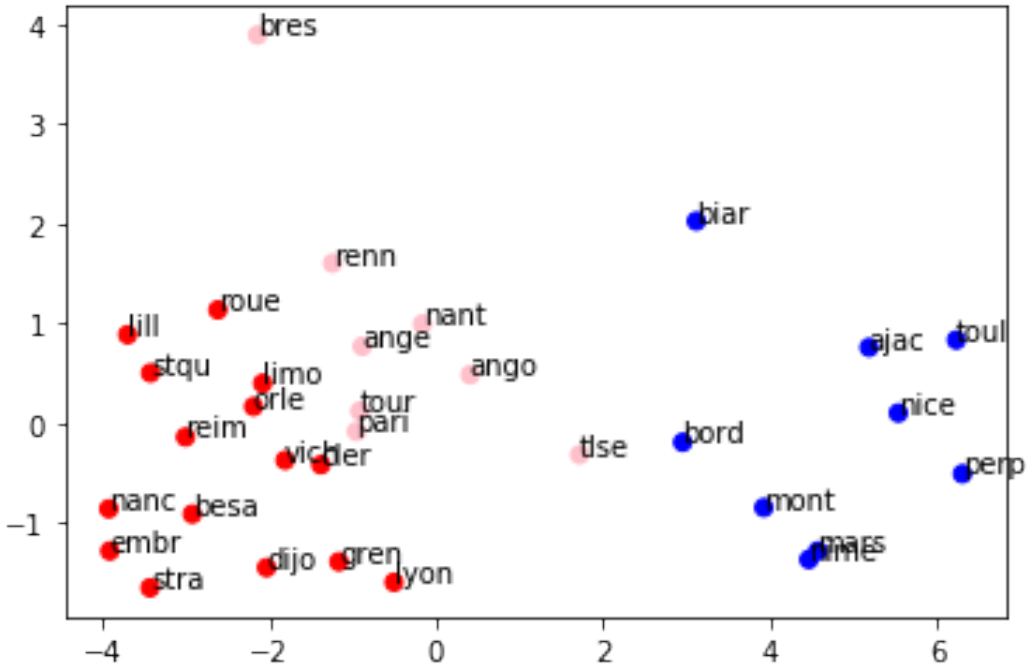

FIGURE 7 – Trois clusters avec KMeans

2.2 Utilisation de l'AgglomerativeClustering

La seconde méthode de clustering utilisée est l'AgglomerativeClustering, elle permet de tester différentes méthodes d'agrégation.
La méthode d'agrégation par défaut est la méthode 'ward' :

```
agglo = cluster.AgglomerativeClustering(n_clusters=3, linkage='ward')
agglo.fit(X_pca)
clustering = agglo.labels_
```

Voici le graphe obtenu :

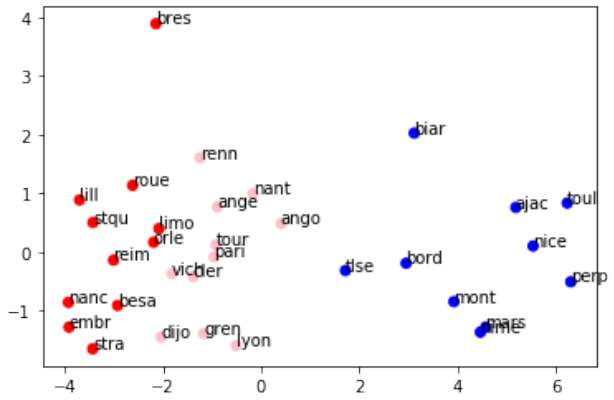

FIGURE 8 – Trois clusters avec AgglomerativeClustering, méthode ward

La seconde méthode d'agrégation testée est la méthode 'average' :

```
agglo = cluster.AgglomerativeClustering(n_clusters=3, linkage='average')
agglo.fit(X_pca)
clustering = agglo.labels_
```

Voici le graphe obtenu :

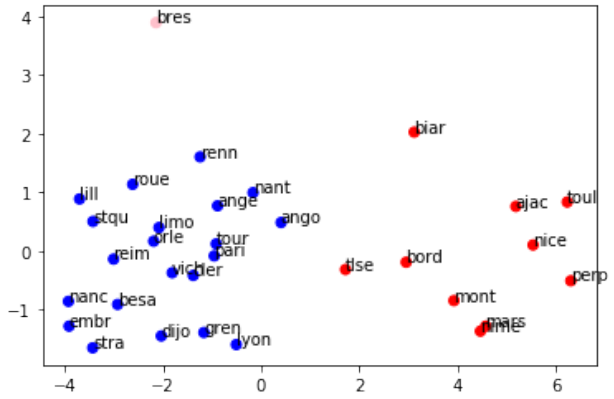

FIGURE 9 – Trois clusters avec AgglomerativeClustering, méthode average

2.3 Indice Silhouette

Avec l'indice silhouette on s'aperçoit que le clustering avec l'indice maximal est le clustering avec 2 clusters, avec un score d'environ 0.63. Ce qui donne un clustering entre les villes du Nord et celle du Sud.