Fayez DEBBABI
fayez.debbabi@telecom-bretagne.eu

Souleymane DIEYE
souleymane.dieye@telecom-bretagne.eu

Etude et implémentation de l'algorithme des k-means

INF 413

Formation ingénieur généraliste

Année scolaire 2016 - 2017

SOMMAIRE

1. INTRODUCTION

Dans l'analyse de données plusieurs algorithmes d'auto-apprentissage existent qui permettent de résoudre le problème de classification des données de manière à les diviser en sousgroupe et essayer de les regroupement en termes de ressemblance (qui possèdent des caractéristiques communes) qui peuvent être un critère de proximité si l'on définisse par exemple une norme à exécuter en tant que fonction de mesure, on cite quelques exemples des méthodes de regroupement de données : Le K-means algorithme et les méthodes de regroupement hiérarchique.

Dans les parties qui suivent de TP on va étudier en détails l'algorithme de k-means et on va l'implémenter en python.

2. ALGORITHME K-MEANS

2.1 BUT ET HISTOIRE

Les données avant ont été représentés d'une manière indépendante c'est-à-dire sous différentes formes et l'idée donc était d'essayer de regrouper ces données sous formes des groupes ou des classes qui les classifient en termes de ressemblance on cherche alors une segmentation de manière plus précise, une optimisation d'un critère qu'on fixe dès le début pour aboutir à une homogénéité relative au critère de ressemblance qu'on cherche à analyser et bien sûr tout on garantissant que l'algorithme soit un algorithme non supervisé qui permet de trouver un modèle de ressemblance en fonction des données entrées.

2.2 EXPLICATION DE L'ALGORITHME K-MEANS

L'algorithme de k-means permet à partir de n données (x1, x2,…xn) entrées de dimension d chacun, de k groupes de centres (c1,c2,…,ck) où chaque donnée va être un membre d'une classe (un groupe) qui caractérise que tous les points appartenant à ce groupe ont des points communs et des caractéristiques de ressemblance en minimisant par exemple la distance (dans notre cas c'est la distance euclidienne) entre un point et le centre le plus proche de chaque partition.

$$d(x, y) = \|y - x\|$$

Ce qui revient à trouver :

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2$$

Description :

- Les étapes de k-means sont :
- Initialisation des groupes en choisissant k sommets comme centres de k groupes.
- Affectation des sommets aux groupes les plus proches en termes de distance (Euclidienne pour notre cas) : (re)affecter chaque individu s au groupe Gi de centre ci tel que distance (s, ci) est minimal
- Mettre à jour les représentants des groupes ci : recalculer ci de chaque groupe (en calculant le barycentre pour notre cas)
- Aller à l'étape 2 selon le critère d'arrêt (exemple : nombre maximal d'itérations ou le fait que aucun individu a changé de son groupe après une mise à jour)

Analyse :
Il y a un nombre fini de partitions possibles à K classes. De plus, chaque étape de l'algorithme fait strictement diminuer la fonction de coût, positive, et fait découvrir une meilleure partition. Cela permet d'affirmer que l'algorithme converge toujours en temps fini, c'est-à-dire se termine.

La solution finale n'est pas toujours optimale. De plus le temps de calcul peut être exponentiel en fonction du nombre de points, même dans le plan (en deux dimensions). Dans la pratique, il est possible d'imposer une limite (un critère d'arrêt) sur le nombre d'itérations ou un critère sur l'amélioration entre itérations.

2.3 COMPLEXITE

Complexité La complexité de cet algorithme est de l'ordre de O(K*N*i*s), où i est le nombre d'itération de l'algorithme et s est la complexité du calcul de la distance.

2.4 VARIANTES :

Il existe plusieurs variantes de l'algorithme selon le résultat voulu :

- Choix des centres : Il est préférable dans la plupart des cas de choisir les centres aléatoirement au sein des données lorsqu'on n'a pas une grande connaissance de ces dernières. Pour le calcul des centres, il existe 2 approches :
  1- Soit on calcul des centres fictifs au sein du groupe.

2- Soit on calcul des centres fictifs et on attribue le centre au point le plus proche à ce centre fictif. C'est très important de spécifier son choix de de centres car différents choix peuvent donner différents résultats de clustering !

- Choix des K : On peut choisir le K manuellement, dans ce cas c'est l'utilisateur qui entre le nombre de clusters dans le programme. Sinon, on peut utiliser des algorithmes pour trouver le meilleur compris entre le K, complexité et répartition des données. Par exemple on cite l'algorithme :Elbow method , Gap statistic …

  Dans notre algorithme on a laissé la liberté à l'utilisateur de choisir le K.

- Calcul de la distance : On peut essayer plusieurs distances dépendant de notre cas, par exemple si on a un graphe pondéré comme set de Data, on peut utiliser la distance Gausienne entre 2 point définie par exp(-(poids)²/var) où le poids est celui entre les 2 points et la variance est définie par l'utilisateur à titre d'exemple. Le plus fréquent c'est la distance euclidienne, mais on peut l'améliorer en utilisant la distance euclidienne standardisée par un vecteur (cas des données Iris par exemple).

- Condition d'arrêt : C'est la condition pour arrêter l'algorithme qui peut être soit un nombre maximale d'itération soit une condition de convergence (une variation maximale des centres après 2 itérations). Le mieux reste une convergence absolue (variation nulle) mais qui peut s'avérer couteuse en temps et en ressources.

2.5 AVANTAGES ET INCONVENIENTS :

Avantage:

1)Si le nombre des données entrés est très élevé le K-means permet un calcul plus rapide en termes du temps en comparant avec l'algorithme hiérarchique, à condition que le nombre des groupes reste relativement faible.

2) L'algorithme K-means permet de produire des groupes beaucoup plus denses (plus proches) que le clusturing hiérarchique avec l'indépendance des partitions entre elles.

Inconvénients et limitations

1) Difficile de prédire la bonne valeur de k.
2) Vu qu'on initialise avec des partitions différentes, le résultat peut varier selon les partitions déjà choisies.
3) Le k-means n'est pas bien adapté pour des clusters de différente taille et de différentes densités.

3. IMPLEMENTATION DE L'ALGORITHME

3.1 STRUCTURE :

Le paradigme de programmation a été de choisir une programmation objet pour les clusters et le calcul. La structure de données est assez simple :

- ► Pour les points : Listes.
- ► Pour les Clusters : Listes d'objets clusters.

Cela nous permet d'avoir une meilleure maniabilité et facilité au niveau du calcul, affectation et gestion de nos données.

Le programme est en python et divisé en 3 scripts :

- ► Main.py : c'est le script principal de notre programme et qui contient toutes les fonctions relatives au : choix du K, calcul des centres, affectation des points et vérification de la convergence.
- ► lecture_ecriture.py :C'est le script pour lire des fichiers, écrire des fichiers et générer des données aléatoires.
- ► test.py : C'est le script conçu pour les simulations. Il réalise les différentes simulations de notre projet (notamment celle en 2D)

Le code est aussi conçu d'une façon qui permet sa réutilisabilité : L'utilisateur peut par exemple ajouter une distance dans Main.py dans la classe calcul , et il n'aura qu'à rajouter son entrée sur la partie « input » dans le même script pour la lancer.

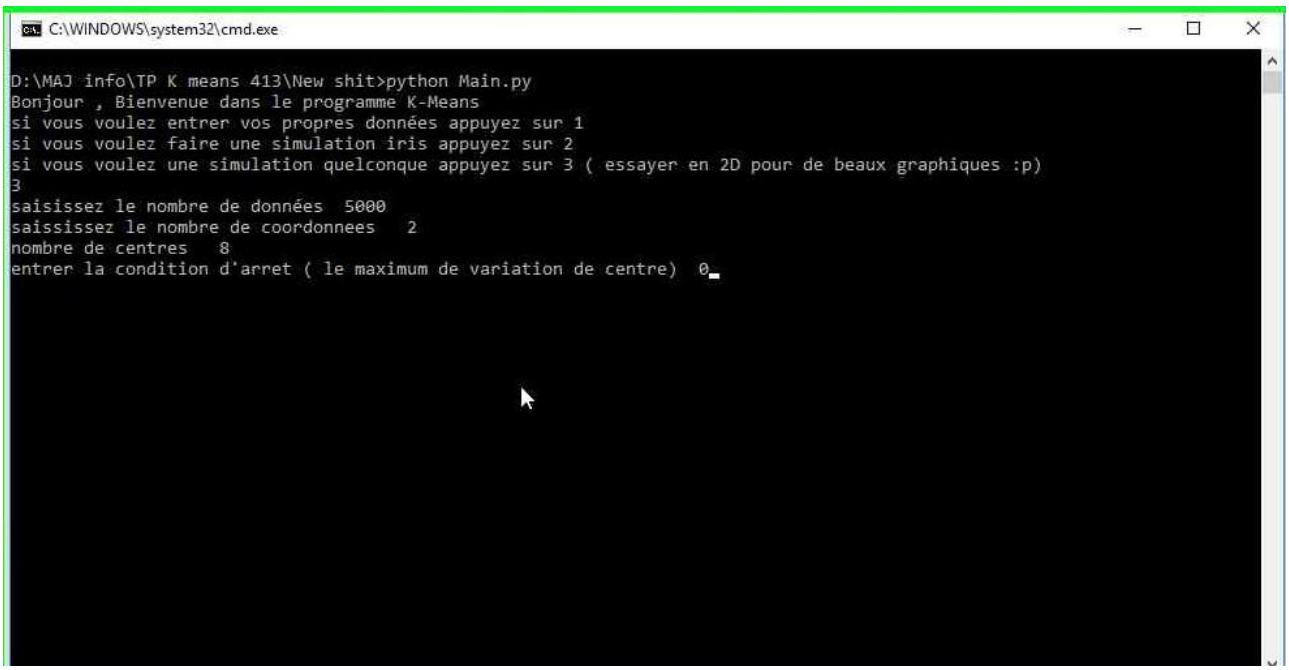
4. TESTS DE L'ALGORITHME :

4.1 SIMULATIONS :

On a commencé nos tests sur des simulations basiques. Cela se fait sur des données aléatoires, avec un nombre K qu'on choisit nous-même. Pour la condition d'arrêt, on a opté pour une convergence totale (variation nulle) et on aussi opté pour une représentation 2D et ce pour s'assurer visuellement de la convergence de l'algorithme et la qualité de son clustering. Ainsi, on a pu suivre en temps réel l'exécution de notre implémentation et corriger les éventuelles erreurs survenues.

La simulation se fait comme ça :

On lance tout d'abord le fichier run.bat et on choisit une simulation quelconque :



Après on appuie sur Entrée et on suit l'exécution en temps réel :

Début

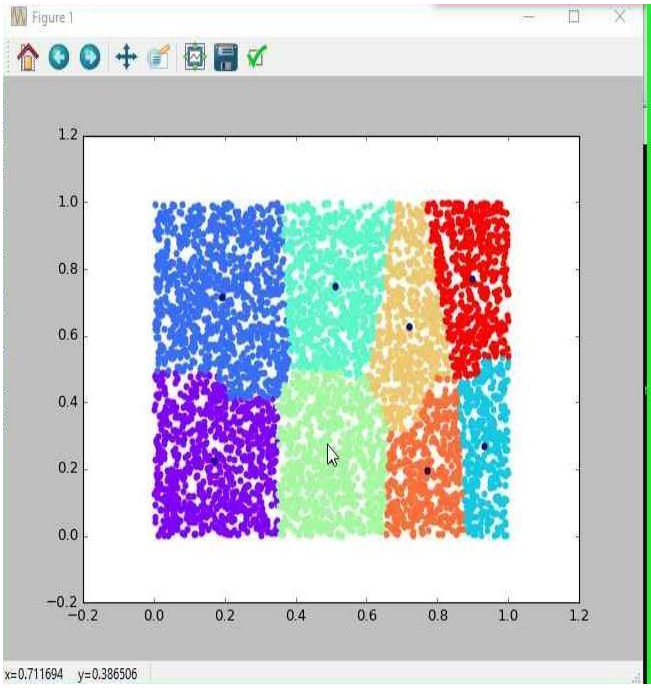


Figure 1
x=0.711694 y=0.386506

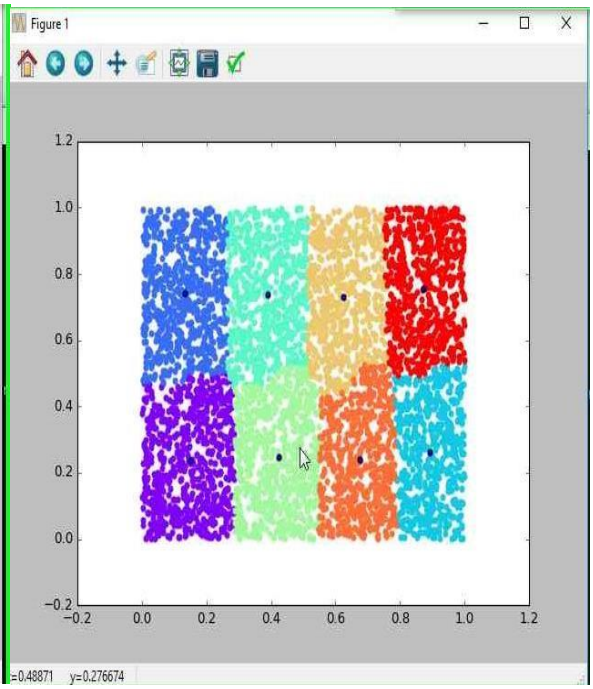


Figure 1
=0.48871 y=0.276674

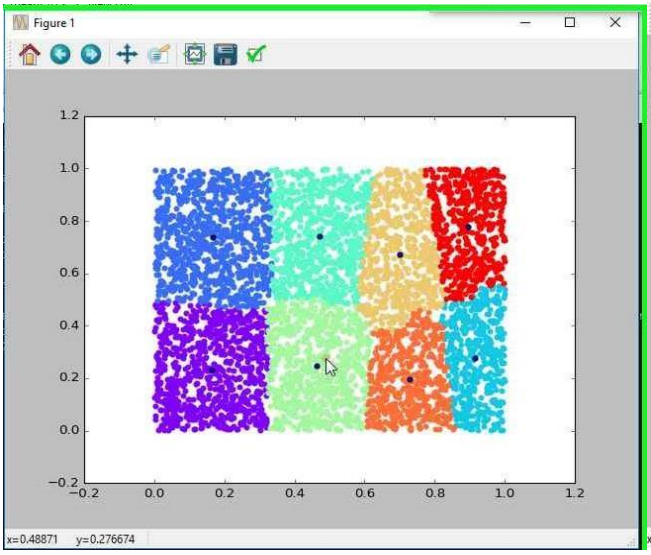


Figure 1
x=0.48871 y=0.276674

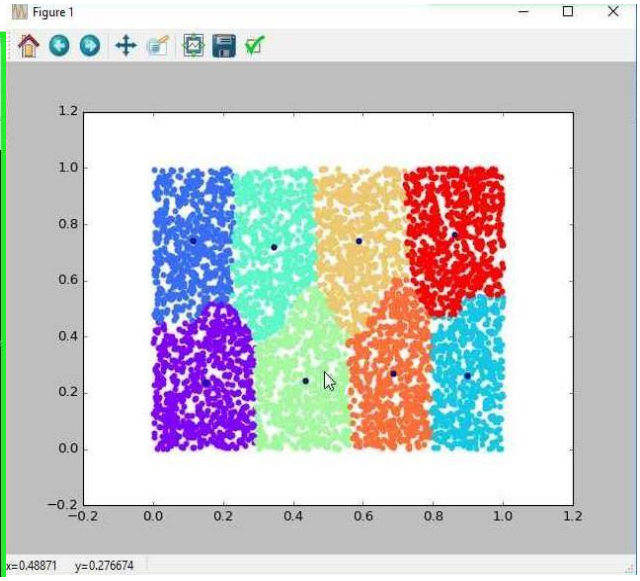
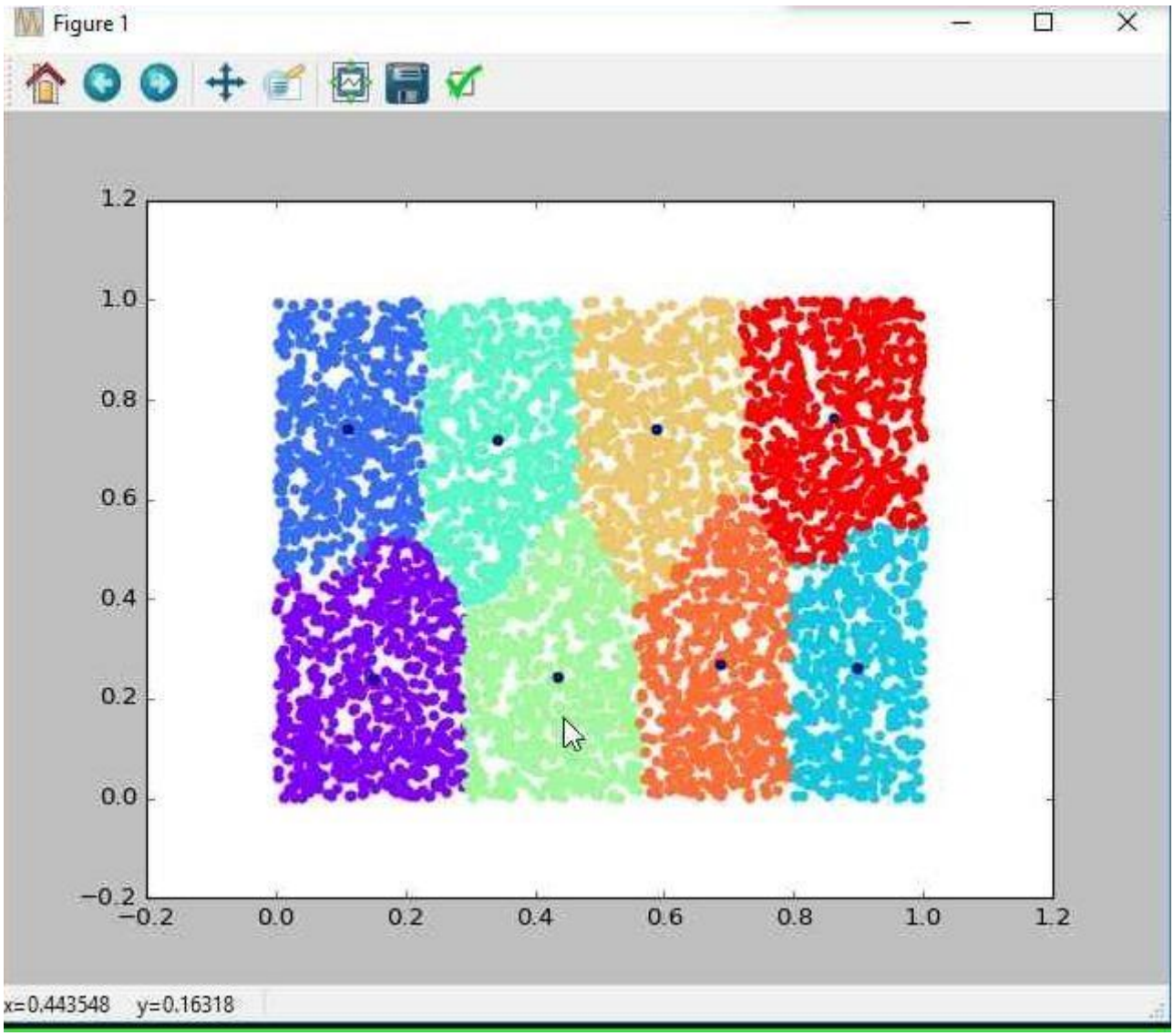


Figure 1
x=0.48871 y=0.276674

FIN



On voit donc bel et bien une bonne convergence et un bon clustering sur nos données aléatoires.

4.2 DONNEES IRIS :

Présentation :

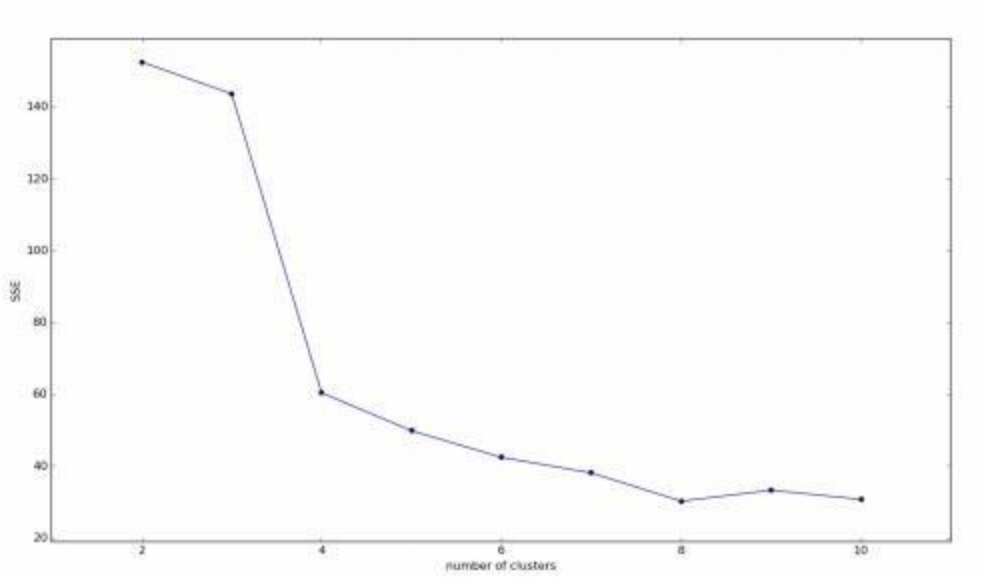
Les données IRIS sont des données constituées de 150 points répartis en 3 groupes. Les 50 premiers sont répartis linéairement et le reste non. Ces données sont très utilisées pour vérifier le bon clustering des algorithmes de type K-means et constituent donc un bon critère de validité de ces derniers.

Distance :

La distance dans les données IRIS est cruciale. En effet par exécution naïve avec une distance usuelle comme l'euclidienne on aboutit à une bonne répartition pour le premier groupe (50 points) mais une mauvaise dans le reste (69 points contre 31) donc une erreur de **19 !**
On peut améliorer cette dernière en passant à la distance de Minkowski d'un grand ordre (puissance de 8) ou directement la distance de Tchebychev. Cela nous donnera donc 50 points pour le premier groupe et un meilleur clustering pour le reste (59 contre 41) et donc amélioration de **10 !**
Néanmoins cela reste assez loin de l'équipartition. Et après renseignement sur ces données il se voit qu'on peut améliorer notre clustering on utilisant une distance Euclidienne Standardisée par le vecteur **v=[0.7826,-0.4194,0.9490,0.9565] .** Ce dernier était fourni dans la documentation des données. Et cela nous a permis d'arriver à 50 points pour le premier groupe, 54 pour le deuxième et 46 pour le troisième et donc seulement une erreur de **4 !**

Choix du K :

Le choix du K était naturellement de 3 comme précisé par la documentation, mais on pouvait par exemple utiliser **l'Elbow method** pour avoir le meilleur K.



On trouve alors un saut de 2 à 4 , donc un clustering optimale en K=3 comme spécifié à la documentation.

*Le code de ce dernier à été récupéré sur :
https://datasciencelab.wordpress.com/2013/12/27/finding-the-k-in-k-means-clustering/

5. CONCLUSION :

Ce TP a été pour nous une introduction au Data analysis et ses algorithmes. C'était une expérience intéressante du point de vue d'apprentissage formel (nouveau domaine, codage en objet) mais aussi d'apprentissage parallèle (Travailler sur des articles de recherches, implémenter de nouveaux algorithmes).

Finalement, on peut dire que ce TP a été une chance pour nous d'approcher un nouveau domaine mais aussi une formation à l'autoformation.

6. BIBLIOGRAPHIE

https://fr.wikipedia.org/wiki/K-moyennes
https://www.quora.com/How-can-we-choose-a-good-K-for-K-means-clustering
https://en.wikipedia.org/wiki/Determining_the_number_of_clusters_in_a_data_set
http://www.galvanize.com/blog/introduction-k-means-cluster-analysis/
https://en.wikipedia.org/wiki/Cluster_analysis
http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/
https://datasciencelab.wordpress.com/2013/12/27/finding-the-k-in-k-means-clustering/