BURDIN Kevin 11507706

FIRMIN-PIGNOT Jeff 11409595

GARDINAZZI Yuri 12111344

Rapport - Analyse de médias sociaux géolocalisés

TP Réalisé avec le langage Python, sur un notebook Jupyter.

- Contenu de l'archive :

L'archive contient un fichier notebook Jupyter, une carte de l'agglomération lyonnaise (pour notre visualisation), ainsi que ce présent rapport. Notre notebook peut lire des datasets en format csv. Nous avons utilisé le dataset suivant :

https://perso.liris.cnrs.fr/marc.plantevit/ENS/DMTP/flickr_data.csv

- Structure générale de notre code
  - Les importations sont toutes regroupées dans une cellule Jupyter.
  - Notre code est découpé selon les différentes étapes du TP.
- Prétraitement des données :
  - Suppression des doublons
  - Restriction de la localisation des tuples à la région lyonnaise, l'objet de notre étude, en se basant sur la latitude et la longitude
  - Initialement, nous nous sommes basés sur la présence du tag « lyon » afin de définir notre espace des datas. Cependant, certaines données localisées à Lyon n'avaient pas le tag lyon, et à l'inverse, certaines données, plus excentrées le possédaient. Nous avons donc fini par filtrer par localisation.
- Visualisation des données :
  - Fond de carte exporté à partir du site : https://www.openstreetmap.org/export
  - Placement des points en fonction de leur localisation.
- Clustering :
  - Les points d'intérêt correspondent à des zones géographiques où une quantité dense de photos se situe. On va donc utiliser un algorithme de clustering qui se base sur la densité : dbscan.
  - Les paramètres du clustering ont été donnés en tâtonnant, en cherchant un clustering qui serait pertinent.
  - Nous avons également normalisé les données avant de les utiliser lors du clustering.
  - Il en ressort un ensemble de labels qui correspond au n° de cluster de chaque donnée.

- Visualisation des données après clustering
    - Le module cm de matplotlib nous propose de construire un ensemble de couleurs différentes, que nous avons utilisé pour assigner une couleur pour chaque cluster.
    - Les localisations ont été regroupées par cluster dans des dictionnaires.
    - Les points sont placés par cluster (et donc par couleur), ainsi que leur légende associée, sur le même fond de carte qu'utilisé précédemment.
- Découverte de sous-groupes
    - Utilisation de la bibliothèque pysubgroup
    - Itérations dans une boucle, en excluant le cluster « -1 », cluster non pertinent de dbscan.
    - Caractérisation des clusters selon les sous-groupes obtenus :
    - Cluster 0 : lat < 45.76, long < 4.82. Un sous-groupe a une data d'upload en 2014, et un autre une date de la prise de la photo dans [2014,2015[.
    - Cluster 1 : long dans [4.83, 4.84 [, lat dans [45.76,45.77[principalement. On remarque un sous-groupe avec une data d'upload en 2014.
    - Cluster 2 : long >= 4.86, lat dans les 45.76 principalement. On remarque des sous-groupes avec des tags nuls.
    - Cluster 3 : long < 4.82 et lat < 45.76 principalement, avec des sous-groupes avec des dates de prise de la photo et d'upload en 2015.
    - Cluster 4 : long dans [4.84, 4.86[et lat >= 45.78 principalement. On remarque des sous-groupes avec des dates d'upload et de prise de la photo en 2014.
    - Cluster 5 : long >= 4.86 et lat >= 45.78 principalement. On remarque des sous-groupes avec une date d'upload en octobre.

- Quel(s) traitement(s) supplémentaires doit-on faire pour détecter des événements géolocalisés sur ces mêmes données ?
    - En général, un événement se déroule dans un espace restreint. Une bonne façon de détecter les événements serait de se baser sur les photos qui sont prises le même jour, à des coordonnées longitude/latitude relativement proches (donc appartenants à un cluster), et de regarder les tags et/ou les titres de ces dernières. On pourrait se baser sur les sous-groupes des clusters de l'exercice d'avant afin de les caractériser, en regardant les données relatives à ces sous-groupes, afin de voir si ces derniers sont reliés à un événement.