Visualisation cartographique en Python sur Jupyter Notebook

Lors de ma mission au sein du Climate Lab de Groupama, l’enjeu était de représenter des zones inondables sur une carte pour pouvoir les visualiser et ensuite effectuer une prédiction quant au niveau de certaines crues.

L’outil Jupyter Notebook de la distribution Anaconda est très pratique pour pouvoir exécuter du code ligne par ligne et afficher des cartes à l’aide de différents modules (ou packages) dédiés.

J’en ai utilisé 3 lors de ma mission à savoir Kepler GL, Folium et Pysheds. Les paragraphes donnent des explications supplémentaires sur leur fonctionnement et leur pertinence dans le cadre d’un projet de data science.

Kepler GL

Ce package a été développé par l’entreprise Uber en 2019 pour pouvoir visualiser des données géographiques tels que le traffic urbain des taxis par exemple.

Le lien suivant <https://kepler.gl/> (page d’accueil de Kzpler GL) permet de prendre en main l’outil en important ses données au format .csv par exemple et de les visualiser. L’utilisation est très intuitive et de nombreux tutoriels existent sur Internet pour pouvoir prendre en main cet outil de la bonne manière.

Bien sûr il est possible d’installer le package correspondant et de l’utiliser directement dans Jupyter Notebook.

Cependant, il est aussi nécessaire d’installer d’autres packages auparavant et certains widgets pour afficher la carte correctement. Il suffit de rentrer les instructions suivantes dans jupyter notebook :

**!pip install ipywidgets**

*Si la commande "pip install" ne fonctionne pas penser à écrire "!pip install" pour installer le package voulu.* Voir le lien: <https://stackoverflow.com/questions/45767252/cant-use-pip-in-jupyter-notebook>

**!pip install keplergl**

**!pip install jupyter\_contrib\_nbextensions**

*Voir le lien* [*https://jupyter-contrib-nbextensions.readthedocs.io/en/latest/install.html*](https://jupyter-contrib-nbextensions.readthedocs.io/en/latest/install.html)

**!jupyter contrib nbextension install –user**

*Voir les liens :*

[*https://jupyter-contrib-nbextensions.readthedocs.io/en/latest/install.html*](https://jupyter-contrib-nbextensions.readthedocs.io/en/latest/install.html)

[*https://stackoverflow.com/questions/57352649/installing-jupyter-extension-failure-to-enable-nbextension*](https://stackoverflow.com/questions/57352649/installing-jupyter-extension-failure-to-enable-nbextension)

**!jupyter contrib nbextension install --py --sys-prefix keplergl**

Enfin, le lien suivant <https://towardsdatascience.com/kepler-gl-jupyter-notebooks-geospatial-data-visualization-with-ubers-opensource-kepler-gl-b1c2423d066f> permet de prendre en main le package kepler GL sous python avec l’outil Jupyter Notebook.

Néanmoins, bien que kepler GL soit très intuitif à utiliser, il met trop de temps à charger les données. C’est pourquoi nous avons opter pour Folium pour visualiser les cartes.

Folium

Ce package est très populaire en langage python bien qu’il soit un peu moins intuitif à utiliser que Kepler GL. Par ailleurs d’autre packages ont été utilisé pour pouvoir personnaliser l’affichage sur Folium (en particulier Geojson). Enfin, les librairies geopandas/geoseries et shapely ont été très utiles pour pouvoir manipuler des formes géométriques avant de les afficher grâce à Folium.

L’objectif va être d’utiliser ces librairies pour pouvoir agrandir/réduire certaines zones inondables selon le type de scénario projeté.

Ensuite on va pouvoir afficher ces zones inondables « projetées » sur une carte grâce à Folium.

Les liens ci-dessous permettent de bien comprendre comment utiliser les commandes associées aux packages/librairies évoquées ci-dessus :

LIENS POUR AFFICHER UNE CARTE À L'AIDE DE FOLIUM ET GEOJSON:

<https://nagasudhir.blogspot.com/2021/07/draw-borders-from-geojson-paths-in.html>

<https://www.youtube.com/watch?v=h16O4xt6yBU>

AUTRE LIEN UTILE, INTERACTIVE MAP WITH LAYERS SELECTION:

<https://kodu.ut.ee/~kmoch/geopython2019/L6/interactive-map-folium.html>

Parametres qu'on peut mettre dans la fonction "style\_function" :

<https://leafletjs.com/reference.html#path-option>

<https://stackoverflow.com/questions/55088688/how-do-you-add-geojsontooltip-to-folium-choropleth-class-in-folium>

# tooltip=folium.features.Tooltip(fields=['feature.properties.county\_name'])

<https://automating-gis-processes.github.io/CSC18/lessons/L4/point-in-polygon.html#reading-kml-files-in-geopandas>

Voir le site <https://towardsdatascience.com/how-to-step-up-your-folium-choropleth-map-skills-17cf6de7c6fe> pour des explications sur la "draggable legend" et bien d'autres choses avec du code HTML pour pouvoir personnaliser l’affichage de la carte sous Folium.

Voir ce lien à propos des polygones qui ne sont pas valides:

<https://stackoverflow.com/questions/20833344/fix-invalid-polygon-in-shapely>

Voir ce lien pour transformer des polygones non valides en des polygones valides grâce à l'instruction make\_valid():

<https://shapely.readthedocs.io/en/latest/manual.html#validation.make_valid>

Voir ce lien pour extraire les polygones d'un multipolygone grâce à l'instruction list(multipolygon.geoms):

<https://stackoverflow.com/questions/38930192/how-to-extract-polygons-from-multipolygons-in-shapely>

Voir ce lien pour vérifier le type de polygone ('Polygon' or 'MultiPolygon') grâce à l'instruction .geom\_type:

<https://stackoverflow.com/questions/39142876/check-if-a-polygon-is-a-multipolygon-in-shapely>

<https://shapely.readthedocs.io/en/stable/manual.html#general-attributes-and-methods>

Vérifier si un polygone est valide grâce à l'instruction .is\_valid:

<https://stackoverflow.com/questions/20833344/fix-invalid-polygon-in-shapely>

<https://shapely.readthedocs.io/en/latest/reference/shapely.is_valid.html?highlight=is_valid>

<https://shapely.readthedocs.io/en/latest/manual.html#unary-predicates>

Transformer un polygone non valide en un multipolygone valide grâce à l'instruction .make\_valid:

<https://shapely.readthedocs.io/en/latest/manual.html#validation.make_valid>

En fait, la fonction .scale de geopandas/geoseries peu s'appliquer à l'ensemble d'une liste entière de type GeoSeries

Voir ce lien pour utiliser la fonction .scale de geopandas pour pouvoir agrandir ou réduire un polygone.

<https://geopandas.org/en/stable/docs/reference/api/geopandas.GeoSeries.scale.html>

<https://stackoverflow.com/questions/49558464/shrink-polygon-using-corner-coordinates>

Autres liens intéressant pour agrandir/rétrécir un polygon : <https://stackoverflow.com/questions/41449084/shapelys-scale-function-in-geopandas-returns-points-of-similar-magnitude>

<https://stackoverflow.com/questions/20230481/shrink-polygon-to-a-specific-area-by-offsetting>

Lien de la librairie Geopandas/GeoSeries : <https://geopandas.org/en/stable/docs/reference/geoseries.html>

Lien de la librairie Shapely : <https://shapely.readthedocs.io/en/stable/manual.html>

Ces 2 derniers liens contiennent toutes les méthodes principalement utilisées pour pouvoir travailler avec des formes géométriques. Il peut être intéressant de se baser sur ces liens pour rechercher d’autres méthodes/fonctions.

Autres liens divers pour pouvoir manipuler un pandas data frame :

<https://stackoverflow.com/questions/50788508/how-can-i-replicate-rows-of-a-pandas-dataframe>

<https://gis.stackexchange.com/questions/162659/joining-concat-list-of-similar-dataframes-in-geopandas>

Avec tous ces liens on comprend que l’affichage avec Folium est plus compliqué qu’avec Kepler GL. Cependant, on peut mieux utiliser et comprendre toutes ces notions liées à la cartographie et la manipulation de formes géométriques. On peut alors réduire ou agrandir certaines zonez inondables en fonction de certains scénarios projetés. De plus, le chargement des données ainsi que l’affichage de la carte finale au format .html sont moins lourds et fastidieux par rapport à Kepler GL.

Enfin une dernière tentative de modélisation a été effectuée à l’aide du package Pysheds disponible en python.

Pysheds

Ce package permet d’importer une image au format .tif qui correspond à un modèle de terrain.

Ce modèle de terrain contient toute la topographie du terrain en question à modéliser (reliefs, crevasses, collines, montagnes, etc.).

Grâce à ces informations, on sait quelles sont les zones où les gouttes d’eau (pluie) vont s’accumuler. Ce qui permettra d’identifier les cours d’eau du terrain.

Puis, en ajustant certains paramètres, on peut définir la largeur de l’inondation autour du cours d’eau selon le type de scénario voulu.

Enfin, on va pouvoir extraire ces zones inondables sous forme d’image fixe pour ensuite les afficher sur une carte grâce à Folium.

Lien pour prendre en main Pysheds :

<https://github.com/mdbartos/pysheds>

<https://mattbartos.com/pysheds/>

<https://mattbartos.com/pysheds/hand.html>

lien pour afficher une image fixe sur Folium :

<https://stackoverflow.com/questions/49582831/adding-jpg-images-to-folium-popup>