PROJET PYTHON

MESTRE PIERRE

2019

ESIEE
PARIS

PROJET PYTHON		
Semestre 1	**PIERRE MESTRE**	2019

Table des matières

I. INTODUCTION : 2

a. Présentation 2

b. Les Prérequis 2

II. STRUCTURATION DU PROGRAMME : 3

a. La problématique 3

b. Les librairies utilisés 3

c. Extraction des données du fichier CSV 3

d. Utilisation de l'API 4

e. Génération de la carte 4

f. Génération de l'histogramme : 5

III. LES RESULTATS 5

a. L'histogramme 5

b. La carte 6

IV. CONCLUSION : 6

PROJET PYTHON		
Semestre 1	**PIERRE MESTRE**	2019

I. INTODUCTION :

a. Présentation

L'objectif de ce premier projet en Python, est de réaliser un programme permettant de manipuler et traiter différentes données. Une fois traité, c'est données devront être représenté sur un graphique et une carte.

Je suis donc parti sur un projet qui va manipuler des données pour voir sur une carte quel sont les pays les plus denses en terme de population/Km².

b. Les Prérequis

Pour réaliser mon projet, je me suis aidé d'un data Frame que j'ai téléchargé sur le site Kaggle.com

Voici le lien pour le télécharger :

https://www.kaggle.com/sudalairajkumar/undata-country-profiles#country_profile_variables.csv

Dans le dossier de mon projet, ce fichier se nomme : **donnee.csv**

Ensuite, il a fallu que je trouve un fichier contenant toutes les coordonnées des pays pour pouvoir dessiner ma carte. Pour cela je suis allé sur le site ***https://geojson-maps.ash.ms/*** Ce site internet permet de générer un fichier .json personnalisé contenant toutes les coordonnée géographique de tous les pays que l'on souhaite. Pour ma part j'ai pris tous les pays du monde. Dans mon dossier de projet, ce fichier se nomme : **coordonne.geo.json**

Pour terminer, mon programma va utiliser plusieurs librairies. C'est pourquoi j'ai créé un script qui en l'exécutant, installe automatiquement toute les librairies manquantes pour pouvoir exécuter mon programme. Je vous invite donc à exécuter le fichier **SCRIPT INSTALLATION LIBRAIRIES.bat** qui se trouve dans mon dossier de projet avant toutes exécutions.

II. STRUCTURATION DU PROGRAMME :

a. La problématique

Dès le début de l'élaboration de mon programme, je me suis heurté à un gros problème. En effet, il fallait que je fasse le lien entre mon DataFrame (DF) et mon fichier GeoJson. Le souci étant que dans mon DataFrame, je n'avais que les noms complets des pays avec la densité correspondante, alors que dans mon GeoJson, je n'avais que le code du pays sur trois lettre (exemple pour France le code est FRA) et les coordonnés associés. Il a fallu donc que je trouve une solution, et pour cela je me suis aidé d'un API fourni par ***https://www.restcountries.eu***.

b. Les librairies utilisés

NOM LIBRAIRIES	UTILITÉ DE LA LIBRAIRIES
CSV	Permet de lire les manipuler les fichiers .csv
PANDAS	Permet de manipuler des données plus facilement
FOLIUM	Permet d'exposer sur une carte des données
JSON	Permet d'exploiter des fichiers .json
MATH	Permet d'appliquer des formules mathématiques (logarithme)
MATPLOTLIB.PYPLOT	Permet de générer des histogrammes

c. Extraction des données du fichier CSV

Pour extraire et manipuler les données, j'utilise une librairie qui se nomme pandas. Cette librairies va extraire les données de mon csv (la densité et le nom des pays) que je vais stocker afin de les manipuler plus tard.

```
df = pd.read_csv('donnee.csv', encoding = 'utf-8')
print("Chargement en cours veuillez patienter (environ 1 a 2 min)")
```

d. Utilisation de l'API

Pour rappel, l'objectif en utilisant l'API, est de récupérer le code iso sur trois lettres de chaque pays.

Pour cela, j'ai créé une fonction ***get_code_iso(name)*** qui prend en paramètre le nom complet du pays et nous retourne le code iso sur trois caractère du pays. Ce code est stocké dans une nouvelle colonne de données associé à chaque densité (comme dans un fichier excel).

```
def get_code_iso(name):
        url = f"https://restcountries.eu/rest/v2/name/{name}"
        # print(url)
        reponse = requests.get(url)
        if reponse.status_code != 200 :
                print("...")
                return ""
        dico = json.loads(reponse.text)
        return dico[0]['alpha3Code']
```

Je vais donc appliquer cette fonction ***get_code_iso(name)*** aux nom de chaque pays.

```
df['codeCountry'] = df['country'].apply(get_code_iso)
```

e. Génération de la carte

Pour générer la carte, j'utilise une librairie qui se nomme Folium. Cette librairie contient une fonction très ***choropleth*** qui permet de générer une carte automatiquement en renseignent quelques paramètres notamment :

- Les données avec une échelle de colloration
- Les coordonnées du geoJson

Tout cela mis dans la fonction choropleth permet de générer une carte qui est colorié avec une intensité de couleur proportionnel à la densité du pays.

```
state_geo = 'coordonne.geo.json'

m = folium.Map(location=[10, 10], zoom_start=3)
m.choropleth(
geo_data=state_geo,
name='choropleth',
data=df,
columns=['codeCountry', 'Population density (per km2, 2017)'],
key_on='feature.properties.iso_a3',
fill_color='YlGn',
fill_opacity=0.7,
line_opacity=0.2,
legend_name='Population density (per km2, 2017)'
)
folium.LayerControl().add_to(m)
```

f. Génération de l'histogramme :

En ce qui concerne l'histogramme, j'utilise **matplotlib.pyplot** qui permet facilement de générer des histogrammes, en lui indiquant des données.

Pour ma part j'ai choisi de générer 2 histogrammes un avec et un autre sans la fonction logarithme appliqué sur la densité. (C'est plus parlant).

Pour le premier histogramme voici le code :

```
df[['country', 'Population density (per km2, 2017)']].hist(bins=50, color='red')
plt.xlabel('densité (nombre de personne / Km²')
plt.ylabel('nombre de pays conscerné')
plt.title('HISTOGRAMME: densité de population / Km² par pays du monde')
```

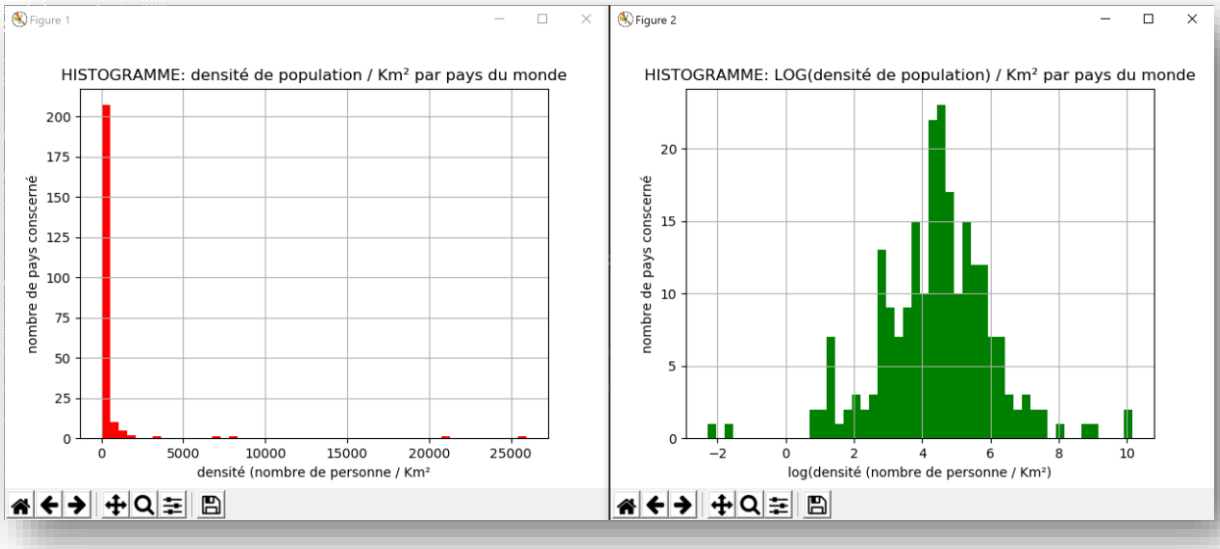
Et voici le code pour le second :

```
df['Population density (per km2, 2017)'] = np.log(df['Population density (per km2, 2017)'])
df[['country', 'Population density (per km2, 2017)']].hist(bins=50, color='green')
plt.xlabel('log(densité (nombre de personne / Km²)')
plt.ylabel('nombre de pays conscerné')
plt.title('HISTOGRAMME: LOG(densité de population) / Km² par pays du monde')
```

III. LES RESULTATS

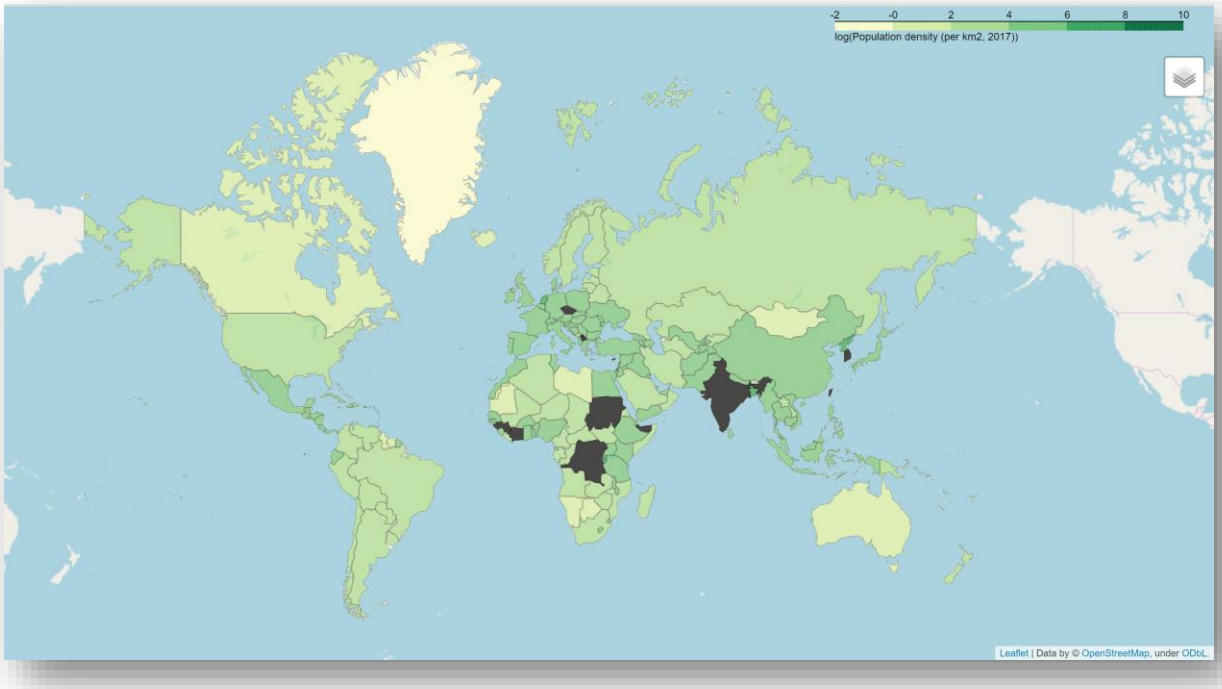
a. L'histogramme

Voici les histogrammes que doit donner le programme :



b. La carte

Voici la carte que doit générer le programme :



Les pays en noirs sont des pays où on n'a pas les données concernant la densité, car l'API n'arrive pas à récupérer le code du pays a trois caractères.

IV. CONCLUSION :

Développer un tel projet en python a été un réel défi car je n'avais aucune connaissance dans ce langage. Toutefois, je trouve que python est un langage très complet avec énormément de possibilité qu'il faut découvrir en pratiquant.

Je tiens toute fois à dire que j'ai partagé a pas mal de camarades de ma classe, le site que j'ai trouvé pour générer les données geo.json, et aussi ma manière de générer les cartes avec folium, ou je me suis aidé du site internet suivant :

https://python-graph-gallery.com/292-choropleth-map-with-folium/

Il peut donc y avoir des similitudes entre mon code et celui de certain groupe.