projet

April 3, 2023

1 Machine Learning and Data Mining Project

1.1 Data retrieval

We retrieve all Pokémons data that we need

[ ]:
```
import kaggle
import os
import shutil

# Si data existe, supprimer le dossier
if os.path.exists('./data'):
    shutil.rmtree('./data')

# Créer le dossier data
os.mkdir('./data')

kaggle.api.authenticate()

kaggle.api.dataset_download_files('kvpratama/pokemon-images-dataset', path='./
 ↪data/', unzip=True)
kaggle.api.dataset_download_files('vishalsubbiah/pokemon-images-and-types',␣
 ↪path='./data/', unzip=True)

# Supprimer le dossier ./data/pokemon_jpg et ./data/images
shutil.rmtree('./data/pokemon_jpg')
shutil.rmtree('./data/images')

# Move ./data/pokemon/pokemon dans ./data/pokemon_images
shutil.move('./data/pokemon/pokemon/', './data/pokemon_images')

# Supprimer le dossier ./data/pokemon
shutil.rmtree('./data/pokemon')

# Remove every file that does not have only number
for file in os.listdir('./data/pokemon_images'):
    if not file.split('.')[0].isdigit():
        os.remove('./data/pokemon_images/' + file)
```

```
for file in os.listdir('./data/pokemon_images/'):
    # Remove every file that has a name over 151
    if int(file.split('.')[0]) > 151:
        os.remove('./data/pokemon_images/' + file)

# Ajouter les id des pokémons dans le fichier data/pokemon.csv
"""
Name,Type1,Type2
bulbasaur,Grass,Poison
ivysaur,Grass,Poison
venusaur,Grass,Poison
charmander,Fire

=>
Id,Name,Type1,Type2
1,bulbasaur,Grass,Poison
2,ivysaur,Grass,Poison
3,venusaur,Grass,Poison
4,charmander,Fire
"""

file = open('./data/pokemon.csv', 'r')
lines = file.readlines()
file.close()

file = open('./data/pokemon.csv', 'w')
file.write('Id,Name,Type1,Type2\n')
for i in range(1, len(lines)):
    file.write(str(i) + ',' + lines[i])
file.close()
```

```
Warning: Your Kaggle API key is readable by other users on this system! To fix
this, you can run 'chmod 600 /home/batiste/.kaggle/kaggle.json'
Warning: Your Kaggle API key is readable by other users on this system! To fix
this, you can run 'chmod 600 /home/batiste/.kaggle/kaggle.json'
```

[ ]:
```
# Création des méta-données de chaque pokémon
"""
{
    "path": filename,
    "size" : imgfile.size,
    "format" : extension,
    "orientation" : orientation
}
"""
```

```python
# Loop through every image (png file) and gather path, size, format, orientation, and write everything to a json file
import json
import os
import shutil
from PIL import Image

# Create a folder for the json files
if os.path.exists('./data/json_metadata_files'):
    shutil.rmtree('./data/json_metadata_files')
os.makedirs('./data/json_metadata_files', exist_ok=True)

# Create a list of all the files in the directory
png_files = os.listdir('./data/pokemon_images')

# sort the list in numerical order (file name : 100.png => 100)
png_files.sort(key=lambda f: int(''.join(filter(str.isdigit, f))))
print(png_files)

for file in png_files :
    imgfile = Image.open('./data/pokemon_images/' + file)

    # Get the size :
    width, height = imgfile.size

    # Get the extension :
    extension = file.split('.')[-1]

    # Get the orientation :
    orientation = 'landscape' if width > height else 'portrait'

    # Create a dictionary with the data
    data = {
        "path": file,
        "size" : imgfile.size,
        "format" : extension,
        "orientation" : orientation
    }

    print(f"Image : {file} | Size : {imgfile.size} | Format : {extension} | Orientation : {orientation}")

    # Write the data to a json file
    json_file = open('./data/json_metadata_files/' + file.split('.')[0] + '.json', 'w')
    json.dump(data, json_file)
    json_file.close()
```

```
['1.png', '2.png', '3.png', '4.png', '5.png', '6.png', '7.png', '8.png',
'9.png', '10.png', '11.png', '12.png', '13.png', '14.png', '15.png', '16.png',
'17.png', '18.png', '19.png', '20.png', '21.png', '22.png', '23.png', '24.png',
'25.png', '26.png', '27.png', '28.png', '29.png', '30.png', '31.png', '32.png',
'33.png', '34.png', '35.png', '36.png', '37.png', '38.png', '39.png', '40.png',
'41.png', '42.png', '43.png', '44.png', '45.png', '46.png', '47.png', '48.png',
'49.png', '50.png', '51.png', '52.png', '53.png', '54.png', '55.png', '56.png',
'57.png', '58.png', '59.png', '60.png', '61.png', '62.png', '63.png', '64.png',
'65.png', '66.png', '67.png', '68.png', '69.png', '70.png', '71.png', '72.png',
'73.png', '74.png', '75.png', '76.png', '77.png', '78.png', '79.png', '80.png',
'81.png', '82.png', '83.png', '84.png', '85.png', '86.png', '87.png', '88.png',
'89.png', '90.png', '91.png', '92.png', '93.png', '94.png', '95.png', '96.png',
'97.png', '98.png', '99.png', '100.png', '101.png', '102.png', '103.png',
'104.png', '105.png', '106.png', '107.png', '108.png', '109.png', '110.png',
'111.png', '112.png', '113.png', '114.png', '115.png', '116.png', '117.png',
'118.png', '119.png', '120.png', '121.png', '122.png', '123.png', '124.png',
'125.png', '126.png', '127.png', '128.png', '129.png', '130.png', '131.png',
'132.png', '133.png', '134.png', '135.png', '136.png', '137.png', '138.png',
'139.png', '140.png', '141.png', '142.png', '143.png', '144.png', '145.png',
'146.png', '147.png', '148.png', '149.png', '150.png', '151.png']
Image : 1.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 2.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 3.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 4.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 5.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 6.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 7.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 8.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 9.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 10.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 11.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 12.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 13.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 14.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 15.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 16.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 17.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 18.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 19.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 20.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 21.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 22.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 23.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 24.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 25.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 26.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 27.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 28.png | Size : (256, 256) | Format : png | Orientation : portrait
```

Image : 29.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 30.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 31.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 32.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 33.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 34.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 35.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 36.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 37.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 38.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 39.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 40.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 41.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 42.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 43.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 44.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 45.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 46.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 47.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 48.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 49.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 50.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 51.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 52.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 53.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 54.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 55.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 56.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 57.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 58.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 59.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 60.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 61.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 62.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 63.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 64.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 65.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 66.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 67.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 68.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 69.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 70.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 71.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 72.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 73.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 74.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 75.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 76.png | Size : (256, 256) | Format : png | Orientation : portrait

Image : 77.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 78.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 79.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 80.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 81.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 82.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 83.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 84.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 85.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 86.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 87.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 88.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 89.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 90.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 91.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 92.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 93.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 94.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 95.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 96.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 97.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 98.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 99.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 100.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 101.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 102.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 103.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 104.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 105.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 106.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 107.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 108.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 109.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 110.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 111.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 112.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 113.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 114.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 115.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 116.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 117.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 118.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 119.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 120.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 121.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 122.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 123.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 124.png | Size : (256, 256) | Format : png | Orientation : portrait

```
Image : 125.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 126.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 127.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 128.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 129.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 130.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 131.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 132.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 133.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 134.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 135.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 136.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 137.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 138.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 139.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 140.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 141.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 142.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 143.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 144.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 145.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 146.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 147.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 148.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 149.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 150.png | Size : (256, 256) | Format : png | Orientation : portrait
Image : 151.png | Size : (256, 256) | Format : png | Orientation : portrait
```

2 Data categorization

2.1 Discrimation criterias

- Type1/Type2
- Couleur

Tag : #couleur, #type

[ ]:
```python
import os
import numpy as np
from sklearn.cluster import KMeans, MiniBatchKMeans
from PIL import Image

colors_dict = {}

# Fonction pour obtenir les couleurs dominantes d'une image
def get_dominant_colors(image_path, k, image_processing_size=None):
    # Charger l'image et la convertir en tableau numpy
    image = Image.open(image_path)
```

```
    # Redimensionner l'image pour accélérer le traitement
    if image_processing_size is not None:
        image = image.resize(image_processing_size, Image.Resampling.LANCZOS)

    # Convertir l'image en tableau numpy
    image = np.array(image)

    # Transformer le tableau numpy en un tableau 2D
    w, h, d = tuple(image.shape)
    image_array = np.reshape(image, (w * h, d))

    # Appliquer l'algorithme K-Means
    # kmeans = KMeans(n_clusters=k, random_state=0, n_init=10).fit(image_array)
    # Use mini-batch k-means instead
    kmeans = MiniBatchKMeans(n_clusters=k, random_state=0, batch_size=256,␣
↪n_init=10).fit(image_array)

    # Obtenir les couleurs dominantes
    dominant_colors = []
    for center in kmeans.cluster_centers_:
        # All files are png, so we have to ignore very low alpha values
        if center[3] < 10:
            continue
        dominant_colors.append(center)

    try :
        colors = [
            [
                dominant_colors[0][0],
                dominant_colors[0][1],
                dominant_colors[0][2]
            ],
            [
                dominant_colors[1][0],
                dominant_colors[1][1],
                dominant_colors[1][2]
            ],
            [
                dominant_colors[2][0],
                dominant_colors[2][1],
                dominant_colors[2][2],
            ]
        ]
    except :
        colors = [
            [
```

```
                dominant_colors[0][0],
                dominant_colors[0][1],
                dominant_colors[0][2]
            ],
            [
                dominant_colors[1][0],
                dominant_colors[1][1],
                dominant_colors[1][2]
            ]
        ]

    return colors


# Parcourir tous les fichiers d'images dans le dossier
images_folder = "./data/pokemon_images"
cpt = 0

png_files = os.listdir(images_folder)
# sort the list in numerical order (file name : 100.png => 100)
png_files.sort(key=lambda f: int(''.join(filter(str.isdigit, f))))

for filename in png_files:
    image_path = os.path.join(images_folder, filename)

    # Obtenir les couleurs 3 dominantes de l'image
    colors = get_dominant_colors(image_path, 5, (100, 100))

    # Print the progress
    cpt+=1
    print(str(cpt) + "/" + str(len(os.listdir(images_folder))))
    print(image_path)

    # Add the colors to the colors_dict dictionary
    colors_dict[filename] = colors
```

```
1/151
./data/pokemon_images/1.png
2/151
./data/pokemon_images/2.png
3/151
./data/pokemon_images/3.png
4/151
./data/pokemon_images/4.png
5/151
./data/pokemon_images/5.png
6/151
```

```
./data/pokemon_images/6.png
7/151
./data/pokemon_images/7.png
8/151
./data/pokemon_images/8.png
9/151
./data/pokemon_images/9.png
10/151
./data/pokemon_images/10.png
11/151
./data/pokemon_images/11.png
12/151
./data/pokemon_images/12.png
13/151
./data/pokemon_images/13.png
14/151
./data/pokemon_images/14.png
15/151
./data/pokemon_images/15.png
16/151
./data/pokemon_images/16.png
17/151
./data/pokemon_images/17.png
18/151
./data/pokemon_images/18.png
19/151
./data/pokemon_images/19.png
20/151
./data/pokemon_images/20.png
21/151
./data/pokemon_images/21.png
22/151
./data/pokemon_images/22.png
23/151
./data/pokemon_images/23.png
24/151
./data/pokemon_images/24.png
25/151
./data/pokemon_images/25.png
26/151
./data/pokemon_images/26.png
27/151
./data/pokemon_images/27.png
28/151
./data/pokemon_images/28.png
29/151
./data/pokemon_images/29.png
30/151
```

```
./data/pokemon_images/30.png
31/151
./data/pokemon_images/31.png
32/151
./data/pokemon_images/32.png
33/151
./data/pokemon_images/33.png
34/151
./data/pokemon_images/34.png
35/151
./data/pokemon_images/35.png
36/151
./data/pokemon_images/36.png
37/151
./data/pokemon_images/37.png
38/151
./data/pokemon_images/38.png
39/151
./data/pokemon_images/39.png
40/151
./data/pokemon_images/40.png
41/151
./data/pokemon_images/41.png
42/151
./data/pokemon_images/42.png
43/151
./data/pokemon_images/43.png
44/151
./data/pokemon_images/44.png
45/151
./data/pokemon_images/45.png
46/151
./data/pokemon_images/46.png
47/151
./data/pokemon_images/47.png
48/151
./data/pokemon_images/48.png
49/151
./data/pokemon_images/49.png
50/151
./data/pokemon_images/50.png
51/151
./data/pokemon_images/51.png
52/151
./data/pokemon_images/52.png
53/151
./data/pokemon_images/53.png
54/151
```

```
./data/pokemon_images/54.png
55/151
./data/pokemon_images/55.png
56/151
./data/pokemon_images/56.png
57/151
./data/pokemon_images/57.png
58/151
./data/pokemon_images/58.png
59/151
./data/pokemon_images/59.png
60/151
./data/pokemon_images/60.png
61/151
./data/pokemon_images/61.png
62/151
./data/pokemon_images/62.png
63/151
./data/pokemon_images/63.png
64/151
./data/pokemon_images/64.png
65/151
./data/pokemon_images/65.png
66/151
./data/pokemon_images/66.png
67/151
./data/pokemon_images/67.png
68/151
./data/pokemon_images/68.png
69/151
./data/pokemon_images/69.png
70/151
./data/pokemon_images/70.png
71/151
./data/pokemon_images/71.png
72/151
./data/pokemon_images/72.png
73/151
./data/pokemon_images/73.png
74/151
./data/pokemon_images/74.png
75/151
./data/pokemon_images/75.png
76/151
./data/pokemon_images/76.png
77/151
./data/pokemon_images/77.png
78/151
```

```
./data/pokemon_images/78.png
79/151
./data/pokemon_images/79.png
80/151
./data/pokemon_images/80.png
81/151
./data/pokemon_images/81.png
82/151
./data/pokemon_images/82.png
83/151
./data/pokemon_images/83.png
84/151
./data/pokemon_images/84.png
85/151
./data/pokemon_images/85.png
86/151
./data/pokemon_images/86.png
87/151
./data/pokemon_images/87.png
88/151
./data/pokemon_images/88.png
89/151
./data/pokemon_images/89.png
90/151
./data/pokemon_images/90.png
91/151
./data/pokemon_images/91.png
92/151
./data/pokemon_images/92.png
93/151
./data/pokemon_images/93.png
94/151
./data/pokemon_images/94.png
95/151
./data/pokemon_images/95.png
96/151
./data/pokemon_images/96.png
97/151
./data/pokemon_images/97.png
98/151
./data/pokemon_images/98.png
99/151
./data/pokemon_images/99.png
100/151
./data/pokemon_images/100.png
101/151
./data/pokemon_images/101.png
102/151
```

```
./data/pokemon_images/102.png
103/151
./data/pokemon_images/103.png
104/151
./data/pokemon_images/104.png
105/151
./data/pokemon_images/105.png
106/151
./data/pokemon_images/106.png
107/151
./data/pokemon_images/107.png
108/151
./data/pokemon_images/108.png
109/151
./data/pokemon_images/109.png
110/151
./data/pokemon_images/110.png
111/151
./data/pokemon_images/111.png
112/151
./data/pokemon_images/112.png
113/151
./data/pokemon_images/113.png
114/151
./data/pokemon_images/114.png
115/151
./data/pokemon_images/115.png
116/151
./data/pokemon_images/116.png
117/151
./data/pokemon_images/117.png
118/151
./data/pokemon_images/118.png
119/151
./data/pokemon_images/119.png
120/151
./data/pokemon_images/120.png
121/151
./data/pokemon_images/121.png
122/151
./data/pokemon_images/122.png
123/151
./data/pokemon_images/123.png
124/151
./data/pokemon_images/124.png
125/151
./data/pokemon_images/125.png
126/151
```

```
./data/pokemon_images/126.png
127/151
./data/pokemon_images/127.png
128/151
./data/pokemon_images/128.png
129/151
./data/pokemon_images/129.png
130/151
./data/pokemon_images/130.png
131/151
./data/pokemon_images/131.png
132/151
./data/pokemon_images/132.png
133/151
./data/pokemon_images/133.png
134/151
./data/pokemon_images/134.png
135/151
./data/pokemon_images/135.png
136/151
./data/pokemon_images/136.png
137/151
./data/pokemon_images/137.png
138/151
./data/pokemon_images/138.png
139/151
./data/pokemon_images/139.png
140/151
./data/pokemon_images/140.png
141/151
./data/pokemon_images/141.png
142/151
./data/pokemon_images/142.png
143/151
./data/pokemon_images/143.png
144/151
./data/pokemon_images/144.png
145/151
./data/pokemon_images/145.png
146/151
./data/pokemon_images/146.png
147/151
./data/pokemon_images/147.png
148/151
./data/pokemon_images/148.png
149/151
./data/pokemon_images/149.png
150/151
```

```
./data/pokemon_images/150.png
151/151
./data/pokemon_images/151.png
```

[ ]:
```
# Convert it to json format
j = str(colors_dict).replace("'", '"')

# j to json :
json_file = open('./data/colors.json', 'w')
json_file.write(j)
json_file.close()
```

2.2 Rassemblement des données dans un même fichier

[ ]:
```
# Fichier csv : data/pokemon.csv
"""
Id,Name,Type1,Type2
1,bulbasaur,Grass,Poison
2,ivysaur,Grass,Poison
"""

# Fichiers json des méta-données : data/json_metadata_files
"""
{
    "path": "1.png",
    "size" : [256, 256],
    "format" : "png",
    "orientation" : "portrait"
}
"""

# Fichier JSON des couleurs dominantes : data/colors.json
"""
{
    "1.png": [
        [117, 165, 142],
        [5, 10, 7],
        [160, 207, 189]
    ],
}
"""

import os
import json
import webcolors
from scipy.spatial import KDTree
```

```
def find_closest_color(requested_color, color_list):
    """Find the closest color to a given RGB value"""
    # Create a KDTree from the list of colors
    tree = KDTree(color_list)
    # Query the tree for the nearest color
    dist, index = tree.query(requested_color)
    return color_list[index]

def get_metadata(id, file_list):
    file = open('./data/json_metadata_files/' + file_list[int(id)-1], 'r')
    metadata = file.read()
    file.close()

    metadata = json.loads(metadata)

    return metadata

def get_colors(id, colors_dict):

    color_list = []
    for color in webcolors.CSS3_HEX_TO_NAMES.keys():
        color_list.append(webcolors.hex_to_rgb(color))

    colors = colors_dict[f"{id}.png"]
    color_names = []
    for color in colors:
        closest_color = find_closest_color(color, color_list)
        color_name = webcolors.rgb_to_name(closest_color)
        color_names.append(color_name)
    return color_names


# Création d'un dict data pour le fichier data.json
data = {}

# Récupération des données du fichier data/pokemon.csv
file = open('./data/pokemon.csv', 'r')
lines_pokemon_csv = file.readlines()
file.close()

lines_pokemon_csv = lines_pokemon_csv[1:]
lines_pokemon_csv = [line.replace("\n", "").split(',') for line in␣
 ↪lines_pokemon_csv]

# Récupération des données du fichier data/json_metadata_files
list_metadata_files = os.listdir('./data/json_metadata_files')
# sort the list in numerical order (file name : 100.png => 100)
```

```
list_metadata_files.sort(key=lambda f: int(''.join(filter(str.isdigit, f))))

# Récupération des données du fichier data/colors.json
file = open('./data/colors.json', 'r')
colors_json = file.read()
file.close()

colors_json = json.loads(colors_json)

# Remplissage du dict data
for line in lines_pokemon_csv :
    if int(line[0]) > 151 :
        break
    print(line)
    # check if there is more than one type
    if len(line) > 3:
        pokemon = {
            "id": line[0],
            "name": line[1],
            "type1": line[2],
            "type2": line[3],
            "metadata": get_metadata(line[0], list_metadata_files),
            "colors": get_colors(line[0], colors_json)
        }
    else :
        pokemon = {
            "id": line[0],
            "name": line[1],
            "type1": line[2],
            "metadata": get_metadata(line[0], list_metadata_files),
            "colors": get_colors(line[0], colors_json)
        }

    data[line[0]] = pokemon

# Convert it to json format
j = str(data).replace("'", '"')

# j to json :
json_file = open('./data/data.json', 'w')
json_file.write(j)
json_file.close()
```

```
['1', 'bulbasaur', 'Grass', 'Poison']
['2', 'ivysaur', 'Grass', 'Poison']
['3', 'venusaur', 'Grass', 'Poison']
['4', 'charmander', 'Fire']
```

```
['5', 'charmeleon', 'Fire']
['6', 'charizard', 'Fire', 'Flying']
['7', 'squirtle', 'Water']
['8', 'wartortle', 'Water']
['9', 'blastoise', 'Water']
['10', 'caterpie', 'Bug']
['11', 'metapod', 'Bug']
['12', 'butterfree', 'Bug', 'Flying']
['13', 'weedle', 'Bug', 'Poison']
['14', 'kakuna', 'Bug', 'Poison']
['15', 'beedrill', 'Bug', 'Poison']
['16', 'pidgey', 'Normal', 'Flying']
['17', 'pidgeotto', 'Normal', 'Flying']
['18', 'pidgeot', 'Normal', 'Flying']
['19', 'rattata', 'Normal']
['20', 'raticate', 'Normal']
['21', 'spearow', 'Normal', 'Flying']
['22', 'fearow', 'Normal', 'Flying']
['23', 'ekans', 'Poison']
['24', 'arbok', 'Poison']
['25', 'pikachu', 'Electric']
['26', 'raichu', 'Electric']
['27', 'sandshrew', 'Ground']
['28', 'sandslash', 'Ground']
['29', 'nidoran-f', 'Poison']
['30', 'nidorina', 'Poison']
['31', 'nidoqueen', 'Poison', 'Ground']
['32', 'nidoran-m', 'Poison']
['33', 'nidorino', 'Poison']
['34', 'nidoking', 'Poison', 'Ground']
['35', 'clefairy', 'Fairy']
['36', 'clefable', 'Fairy']
['37', 'vulpix', 'Fire']
['38', 'ninetales', 'Fire']
['39', 'jigglypuff', 'Normal', 'Fairy']
['40', 'wigglytuff', 'Normal', 'Fairy']
['41', 'zubat', 'Poison', 'Flying']
['42', 'golbat', 'Poison', 'Flying']
['43', 'oddish', 'Grass', 'Poison']
['44', 'gloom', 'Grass', 'Poison']
['45', 'vileplume', 'Grass', 'Poison']
['46', 'paras', 'Bug', 'Grass']
['47', 'parasect', 'Bug', 'Grass']
['48', 'venonat', 'Bug', 'Poison']
['49', 'venomoth', 'Bug', 'Poison']
['50', 'diglett', 'Ground']
['51', 'dugtrio', 'Ground']
['52', 'meowth', 'Normal']
```

```
['53', 'persian', 'Normal']
['54', 'psyduck', 'Water']
['55', 'golduck', 'Water']
['56', 'mankey', 'Fighting']
['57', 'primeape', 'Fighting']
['58', 'growlithe', 'Fire']
['59', 'arcanine', 'Fire']
['60', 'poliwag', 'Water']
['61', 'poliwhirl', 'Water']
['62', 'poliwrath', 'Water', 'Fighting']
['63', 'abra', 'Psychic']
['64', 'kadabra', 'Psychic']
['65', 'alakazam', 'Psychic']
['66', 'machop', 'Fighting']
['67', 'machoke', 'Fighting']
['68', 'machamp', 'Fighting']
['69', 'bellsprout', 'Grass', 'Poison']
['70', 'weepinbell', 'Grass', 'Poison']
['71', 'victreebel', 'Grass', 'Poison']
['72', 'tentacool', 'Water', 'Poison']
['73', 'tentacruel', 'Water', 'Poison']
['74', 'geodude', 'Rock', 'Ground']
['75', 'graveler', 'Rock', 'Ground']
['76', 'golem', 'Rock', 'Ground']
['77', 'ponyta', 'Fire']
['78', 'rapidash', 'Fire']
['79', 'slowpoke', 'Water', 'Psychic']
['80', 'slowbro', 'Water', 'Psychic']
['81', 'magnemite', 'Electric', 'Steel']
['82', 'magneton', 'Electric', 'Steel']
['83', 'farfetchd', 'Normal', 'Flying']
['84', 'doduo', 'Normal', 'Flying']
['85', 'dodrio', 'Normal', 'Flying']
['86', 'seel', 'Water']
['87', 'dewgong', 'Water', 'Ice']
['88', 'grimer', 'Poison']
['89', 'muk', 'Poison']
['90', 'shellder', 'Water']
['91', 'cloyster', 'Water', 'Ice']
['92', 'gastly', 'Ghost', 'Poison']
['93', 'haunter', 'Ghost', 'Poison']
['94', 'gengar', 'Ghost', 'Poison']
['95', 'onix', 'Rock', 'Ground']
['96', 'drowzee', 'Psychic']
['97', 'hypno', 'Psychic']
['98', 'krabby', 'Water']
['99', 'kingler', 'Water']
['100', 'voltorb', 'Electric']
```

```
['101', 'electrode', 'Electric']
['102', 'exeggcute', 'Grass', 'Psychic']
['103', 'exeggutor', 'Grass', 'Psychic']
['104', 'cubone', 'Ground']
['105', 'marowak', 'Ground']
['106', 'hitmonlee', 'Fighting']
['107', 'hitmonchan', 'Fighting']
['108', 'lickitung', 'Normal']
['109', 'koffing', 'Poison']
['110', 'weezing', 'Poison']
['111', 'rhyhorn', 'Ground', 'Rock']
['112', 'rhydon', 'Ground', 'Rock']
['113', 'chansey', 'Normal']
['114', 'tangela', 'Grass']
['115', 'kangaskhan', 'Normal']
['116', 'horsea', 'Water']
['117', 'seadra', 'Water']
['118', 'goldeen', 'Water']
['119', 'seaking', 'Water']
['120', 'staryu', 'Water']
['121', 'starmie', 'Water', 'Psychic']
['122', 'mr-mime', 'Psychic', 'Fairy']
['123', 'scyther', 'Bug', 'Flying']
['124', 'jynx', 'Ice', 'Psychic']
['125', 'electabuzz', 'Electric']
['126', 'magmar', 'Fire']
['127', 'pinsir', 'Bug']
['128', 'tauros', 'Normal']
['129', 'magikarp', 'Water']
['130', 'gyarados', 'Water', 'Flying']
['131', 'lapras', 'Water', 'Ice']
['132', 'ditto', 'Normal']
['133', 'eevee', 'Normal']
['134', 'vaporeon', 'Water']
['135', 'jolteon', 'Electric']
['136', 'flareon', 'Fire']
['137', 'porygon', 'Normal']
['138', 'omanyte', 'Rock', 'Water']
['139', 'omastar', 'Rock', 'Water']
['140', 'kabuto', 'Rock', 'Water']
['141', 'kabutops', 'Rock', 'Water']
['142', 'aerodactyl', 'Rock', 'Flying']
['143', 'snorlax', 'Normal']
['144', 'articuno', 'Ice', 'Flying']
['145', 'zapdos', 'Electric', 'Flying']
['146', 'moltres', 'Fire', 'Flying']
['147', 'dratini', 'Dragon']
['148', 'dragonair', 'Dragon']
```

```
['149', 'dragonite', 'Dragon', 'Flying']
['150', 'mewtwo', 'Psychic']
['151', 'mew', 'Psychic']
```

2.3 Profils des utilisateurs

[ ]:
```python
import random
import json
import os

# Création des profil d'utilisateurs, élements favoris: type, couleurs.
class User:
    def __init__(self, id, name, images_list):
        self.id = id
        self.name = name
        self.images_list = images_list
        self.favorite_types = []
        self.favorite_colors = []
        self.liked_images = []
        self.recommended_images = []

    def add_favorite_type(self, type):
        self.favorite_types.append(type)

    def add_favorite_color(self, color):
        self.favorite_colors.append(color)

    def get_favorite_types(self):
        return self.favorite_types

    def get_favorite_colors(self):
        return self.favorite_colors

    def get_images_list(self):
        return self.images_list

    def get_liked_images(self):
        return self.liked_images

    def print_user(self):
        print(f"User {self.id} : {self.name}")
        print(f"Favorite types : {self.favorite_types}")
        print(f"Favorite colors : {self.favorite_colors}")
        print(f"Images list : {self.images_list}")
        print(f"Liked images : {self.liked_images}")

    def get_data(self):
```

```
        return {
            "id": self.id,
            "name": self.name,
            "favorite_types": self.favorite_types,
            "favorite_colors": self.favorite_colors,
            "images_list": self.images_list,
            "liked_images": self.liked_images,
            "recommended_images": self.recommended_images
        }


# All types :
types = ["Normal", "Fire", "Water", "Electric", "Grass", "Ice", "Fighting",␣
 ↪"Poison", "Ground", "Flying", "Psychic", "Bug", "Rock", "Ghost", "Dragon",␣
 ↪"Dark", "Steel", "Fairy"]

NOMBRE_UTILISATEURS = 5
NOMBRE_IMAGES = 20

# Création des utilisateurs
users = []
for i in range(NOMBRE_UTILISATEURS):
    # Création d'une liste d'images aléatoire
    images_list = []
    for j in range(NOMBRE_IMAGES):
        images_list.append(random.randint(1, 151))
    users.append(User(i, f"user{i}", images_list))
```

[ ]:
```
"""
One piece of the data.json file
{
    "1": {
        "id": "1",
        "name": "bulbasaur",
        "type1": "Grass",
        "type2": "Poison",
        "metadata": {
            "path": "1.png",
            "size": [
                256,
                256
            ],
            "format": "png",
            "orientation": "portrait"
        },
        "colors": [
            "dimgray",
```

```
            "cadetblue",
            "silver"
        ]
    },
}
"""

# Like random des images pour chaque utilisateur

import random
import webcolors
from scipy.spatial import KDTree

# On vide les données des utilisateurs
for user in users:
    user.favorite_types = []
    user.favorite_colors = []
    user.liked_images = []


for user in users:
    for image in user.get_images_list():
        user.liked_images.append((image, True if random.randint(0, 1) == 1 else␣
 ↪False))

# Ajout de types et de couleurs favoris pour chaque utilisateur en fonction des␣
 ↪images likées, et ajout du nombre de fois que le type ou la couleur est liké
for user in users:
    for image_like_relation in user.get_liked_images():
        # Si l'image est likée
        if image_like_relation[1]:
            # Récupération des données de l'image
            file = open('./data/data.json', 'r')
            data = file.read()
            file.close()

            data = json.loads(data)

            image_data = data[str(image_like_relation[0])]

            # Ajout des types favoris
            if "type2" in image_data:
                user.add_favorite_type(image_data["type1"])
                user.add_favorite_type(image_data["type2"])
            else:
                user.add_favorite_type(image_data["type1"])
```

```
            # Ajout des couleurs favoris
            for color in image_data["colors"]:
                user.add_favorite_color(color)


# Compter le nombre d'occurence de chaque type et de chaque couleur
for user in users:
    # Types
    user.favorite_types = {i: user.favorite_types.count(i) for i in user.
 ↪favorite_types}
    # Couleurs
    user.favorite_colors = {i: user.favorite_colors.count(i) for i in user.
 ↪favorite_colors}

# Trie dans l'ordre décroissant
for user in users:
    # Types
    user.favorite_types = dict(sorted(user.favorite_types.items(), key=lambda␣
 ↪item: item[1], reverse=True))
    # Couleurs
    user.favorite_colors = dict(sorted(user.favorite_colors.items(), key=lambda␣
 ↪item: item[1], reverse=True))
```

[ ]:
```
for user in users:
    print("====================================")
    user.print_user()
```

```
====================================
User 0 : user0
Favorite types : {'Poison': 3, 'Psychic': 2, 'Water': 2, 'Bug': 1, 'Ground': 1,
'Fire': 1, 'Flying': 1, 'Fairy': 1, 'Rock': 1, 'Fighting': 1, 'Normal': 1,
'Ghost': 1}
Favorite colors : {'dimgray': 5, 'darkslategray': 4, 'gray': 2, 'lavender': 2,
'slategray': 2, 'silver': 2, 'rosybrown': 2, 'darkgray': 2, 'olivedrab': 1,
'darkolivegreen': 1, 'darkkhaki': 1, 'pink': 1, 'lightslategray': 1,
'burlywood': 1, 'peru': 1, 'gainsboro': 1, 'black': 1, 'darksalmon': 1,
'lightsteelblue': 1, 'mistyrose': 1}
Images list : [11, 151, 34, 100, 60, 75, 22, 19, 31, 138, 146, 122, 140, 126,
67, 66, 113, 140, 109, 94]
Liked images : [(11, True), (151, True), (34, False), (100, False), (60, True),
(75, False), (22, False), (19, False), (31, True), (138, False), (146, True),
(122, True), (140, True), (126, False), (67, False), (66, True), (113, True),
(140, False), (109, True), (94, True)]
====================================
User 1 : user1
Favorite types : {'Normal': 4, 'Bug': 3, 'Flying': 3, 'Grass': 1, 'Poison': 1,
'Fighting': 1}
```

```
Favorite colors : {'darkslategray': 5, 'indianred': 3, 'lightgray': 2,
'darkolivegreen': 2, 'wheat': 2, 'darkseagreen': 1, 'gainsboro': 1,
'lightslategray': 1, 'peru': 1, 'whitesmoke': 1, 'rosybrown': 1, 'black': 1,
'antiquewhite': 1, 'darkgray': 1, 'gray': 1}
Images list : [46, 123, 127, 4, 18, 100, 143, 138, 66, 62, 134, 8, 13, 30, 95,
52, 72, 106, 18, 81]
Liked images : [(46, True), (123, True), (127, False), (4, False), (18, True),
(100, False), (143, True), (138, False), (66, False), (62, False), (134, False),
(8, False), (13, True), (30, False), (95, False), (52, True), (72, False), (106,
True), (18, True), (81, False)]
====================================
User 2 : user2
Favorite types : {'Psychic': 3, 'Flying': 2, 'Grass': 2, 'Poison': 2,
'Fighting': 1, 'Normal': 1, 'Ground': 1, 'Rock': 1, 'Electric': 1, 'Fairy': 1}
Favorite colors : {'darkslategray': 7, 'darkgray': 3, 'gray': 3, 'dimgray': 3,
'sandybrown': 2, 'darkolivegreen': 2, 'darkseagreen': 2, 'khaki': 2,
'rosybrown': 2, 'lightgray': 1, 'tan': 1, 'whitesmoke': 1, 'gainsboro': 1}
Images list : [106, 97, 84, 29, 96, 51, 142, 69, 70, 27, 67, 30, 127, 92, 104,
142, 101, 45, 85, 122]
Liked images : [(106, True), (97, True), (84, True), (29, False), (96, True),
(51, True), (142, False), (69, True), (70, True), (27, False), (67, False), (30,
False), (127, False), (92, False), (104, False), (142, True), (101, True), (45,
False), (85, False), (122, True)]
====================================
User 3 : user3
Favorite types : {'Flying': 3, 'Ice': 2, 'Fire': 2, 'Fighting': 1, 'Water': 1,
'Normal': 1, 'Grass': 1, 'Poison': 1, 'Ground': 1}
Favorite colors : {'darkslategray': 4, 'dimgray': 3, 'lightsteelblue': 2,
'peru': 2, 'burlywood': 2, 'darkgray': 1, 'lightslategray': 1, 'silver': 1,
'darkkhaki': 1, 'gray': 1, 'darkcyan': 1, 'darkseagreen': 1, 'khaki': 1,
'slategray': 1, 'gainsboro': 1, 'rosybrown': 1}
Images list : [145, 66, 91, 107, 98, 114, 70, 56, 128, 131, 123, 6, 69, 72, 22,
71, 144, 49, 146, 105]
Liked images : [(145, False), (66, True), (91, True), (107, False), (98, False),
(114, False), (70, False), (56, False), (128, True), (131, False), (123, False),
(6, True), (69, True), (72, False), (22, False), (71, False), (144, True), (49,
False), (146, True), (105, True)]
====================================
User 4 : user4
Favorite types : {'Normal': 2, 'Water': 1, 'Poison': 1, 'Psychic': 1, 'Fairy':
1, 'Ground': 1}
Favorite colors : {'darkslategray': 3, 'rosybrown': 3, 'dimgray': 2,
'lightslategray': 1, 'skyblue': 1, 'gainsboro': 1, 'gray': 1, 'silver': 1,
'pink': 1, 'lightgray': 1}
Images list : [57, 73, 94, 122, 13, 115, 45, 126, 22, 143, 115, 14, 2, 145, 121,
9, 108, 104, 150, 45]
Liked images : [(57, False), (73, True), (94, False), (122, True), (13, False),
(115, False), (45, False), (126, False), (22, False), (143, False), (115, True),
```

```
(14, False), (2, False), (145, False), (121, False), (9, False), (108, True),
(104, True), (150, False), (45, False)]
```

[ ]:
```python
import os
import random
import json
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from PIL import Image
from sklearn.cluster import MiniBatchKMeans
from scipy.spatial.distance import cdist

NOMBRE_RECOMMANDATIONS = 3

def load_data(json_file):
    with open(json_file, "r") as file:
        data = json.load(file)
    return data

data = load_data("data/data.json")

def extract_features(data):
    features = []
    for id, pokemon in data.items():
        img = Image.open(os.path.join("data/pokemon_images",␣
 ↪pokemon["metadata"]["path"]))
        img = img.resize((64, 64))
        img_array = np.asarray(img)
        img_array = img_array.flatten() / 255.0
        features.append(img_array)
    return np.array(features)

features = extract_features(data)

n_clusters = 10
minibatch_kmeans = MiniBatchKMeans(n_clusters=n_clusters, random_state=42,␣
 ↪n_init=10)
minibatch_kmeans.fit(features)

def find_closest_cluster(user, data, minibatch_kmeans):
    user_preferences = []

    for id, pokemon in data.items():
        type1 = pokemon.get("type1", "")
        type2 = pokemon.get("type2", "")
```

```
        if type1 in user.get_favorite_types() or type2 in user.
 ↪get_favorite_types():
            for color in pokemon["colors"]:
                if color in user.get_favorite_colors():
                    user_preferences.append((id, pokemon))
                    break

    user_features = extract_features({id: pokemon for id, pokemon in␣
 ↪user_preferences})
    cluster_assignments = minibatch_kmeans.predict(user_features)

    closest_cluster = np.argmax(np.bincount(cluster_assignments))
    return closest_cluster

def recommend_images(data, minibatch_kmeans, closest_cluster, user_features,␣
 ↪max_recommendations, liked_images_ids):
    distances = []

    for id, pokemon in data.items():
        img_features = extract_features({id: pokemon}).reshape(1, -1)
        cluster_assignment = minibatch_kmeans.predict(img_features)

        if cluster_assignment == closest_cluster and id not in liked_images_ids:
            distance = np.linalg.norm(user_features - img_features)
            distances.append((id, distance))

    # Triez les images en fonction de leur distance par rapport aux préférences␣
 ↪de l'utilisateur
    sorted_distances = sorted(distances, key=lambda x: x[1])

    # Prenez les max_recommendations images les plus proches
    closest_images_ids = [x[0] for x in sorted_distances[:max_recommendations]]

    return closest_images_ids



recommendations = {}

for user in users:
    closest_cluster = find_closest_cluster(user, data, minibatch_kmeans)
    user_features = extract_features({str(id): data[str(id)] for id, liked in␣
 ↪user.get_liked_images() if liked})
    liked_images_ids = [str(id) for id, liked in user.get_liked_images() if␣
 ↪liked]
```

```
    recommendations[user.name] = recommend_images(data, minibatch_kmeans,␣
 ↪closest_cluster, user_features, NOMBRE_RECOMMANDATIONS, liked_images_ids)

print(recommendations)


for user_name, recommended_pokemons in recommendations.items():
    print(f"{user_name} a {len(recommended_pokemons)} Pokémon recommandés :␣
 ↪{recommended_pokemons}")

# Ajout des recommandations dans les données des utilisateurs
for user in users:
    user.recommended_images = recommendations[user.name]
```

```
{'user0': ['3', '44', '34'], 'user1': ['93', '3', '130'], 'user2': ['44', '3',
'93'], 'user3': ['124', '2', '111'], 'user4': ['3', '44', '83']}
user0 a 3 Pokémon recommandés : ['3', '44', '34']
user1 a 3 Pokémon recommandés : ['93', '3', '130']
user2 a 3 Pokémon recommandés : ['44', '3', '93']
user3 a 3 Pokémon recommandés : ['124', '2', '111']
user4 a 3 Pokémon recommandés : ['3', '44', '83']
```

3 Sauvegarde des données

[ ]:
```
import json
import os
import shutil

# Ecriture des données de tous les utilisateurs dans un fichier "data/users.
 ↪json"
with open("data/users.json", "w") as file:
    json.dump([user.__dict__ for user in users], file)
```

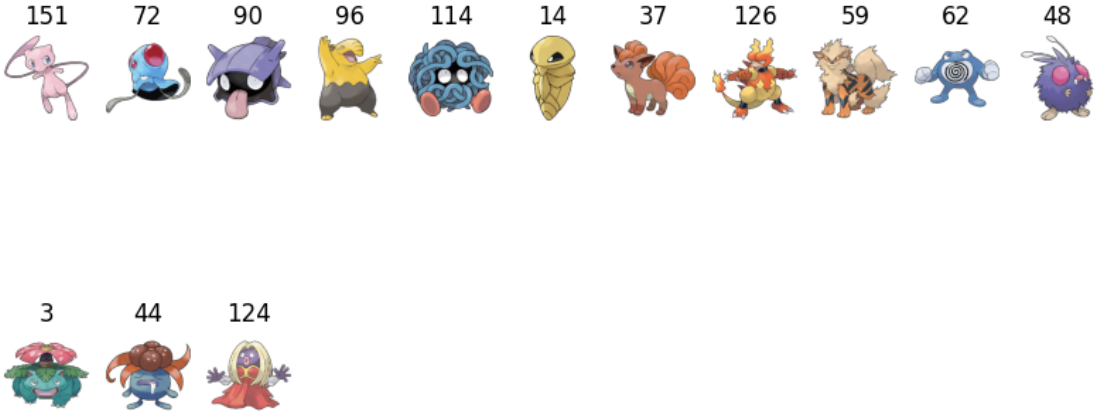
3.1 Affichage des données

[ ]:
```
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec

def plot_images(user_name, liked_images, recommended_images):
    fig = plt.figure(figsize=(10, 5))
    gs = gridspec.GridSpec(2, len(liked_images), figure=fig)

    # Affichez les images aimées
    plt.subplot(gs[0, :])
    for i, image_id in enumerate(liked_images):
```

```
        img = Image.open(os.path.join("data/pokemon_images",␣
↪data[image_id]["metadata"]["path"]))
        plt.subplot(gs[0, i])
        plt.imshow(img)
        plt.axis("off")
        plt.title(f"{image_id}")

    # Affichez les images recommandées
    plt.subplot(gs[1, :])
    for i, image_id in enumerate(recommended_images):
        img = Image.open(os.path.join("data/pokemon_images",␣
↪data[image_id]["metadata"]["path"]))
        plt.subplot(gs[1, i])
        plt.imshow(img)
        plt.axis("off")
        plt.title(f"{image_id}")

    plt.suptitle(f"{user_name}'s Liked and Recommended Images")
    plt.show()

# Créez un dictionnaire pour stocker les utilisateurs avec leur nom comme clé
users_dict = {user.name: user for user in users}

# Affichez les images aimées et recommandées pour chaque utilisateur
for user_name, recommended_images in recommendations.items():
    user = users_dict[user_name]
    liked_images = [str(id) for id, liked in user.get_liked_images() if liked]
    plot_images(user_name, liked_images, recommended_images)
```

```
/tmp/ipykernel_482/2429952938.py:12: MatplotlibDeprecationWarning: Auto-removal
of overlapping axes is deprecated since 3.6 and will be removed two minor
releases later; explicitly call ax.remove() as needed.
  plt.subplot(gs[0, i])
/tmp/ipykernel_482/2429952938.py:21: MatplotlibDeprecationWarning: Auto-removal
of overlapping axes is deprecated since 3.6 and will be removed two minor
releases later; explicitly call ax.remove() as needed.
  plt.subplot(gs[1, i])
```

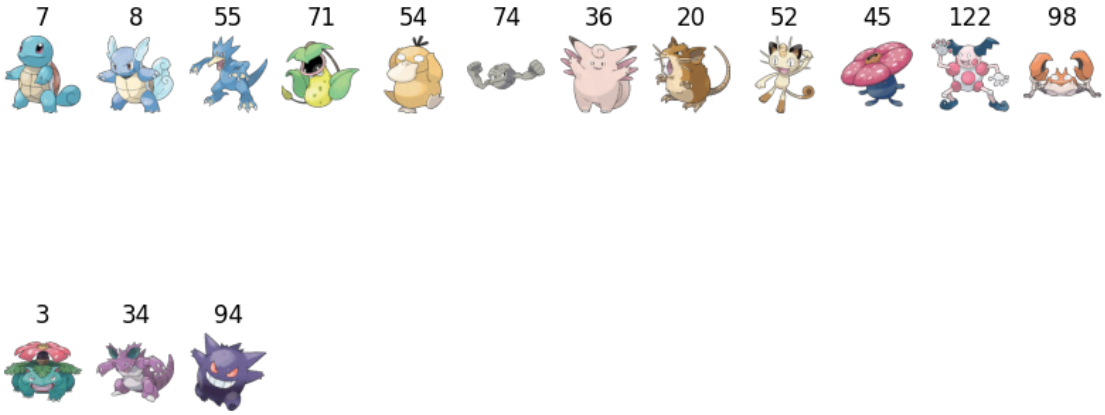
user0's Liked and Recommended Images



user1's Liked and Recommended Images

user2's Liked and Recommended Images

user3's Liked and Recommended Images



user4's Liked and Recommended Images