Міністерство освіти і науки України
Національний технічний університет України «КПІ» імені Ігоря Сікорського
Кафедра обчислювальної техніки ФІОТ

ЗВІТ
з лабораторної роботи №5
з навчальної дисципліни «Вступ до технології Data Science»

Тема:

РЕАЛІЗАЦІЯ МЕТОДІВ МАШИННОГО НАВЧАННЯ (MACHINE LEARNING (ML))

Виконав:
Студент 3 курсу кафедри ІПІ ФІОТ,
Навчальної групи ІП-11
Лошак В.І.

Перевірив:
Професор кафедри ОТ ФІОТ
Писарчук О.О.

Київ 2023

I. Мета роботи:
Виявити дослідити та узагальнити особливості аналізу даних з використанням методів та технологій машинного навчання (Machine Learning (ML))

II. Завдання:
Завдання III рівня складності 9 балів: реалізувати на вибір ТРИ з п'яти сформованих груп технічних вимог.

Група технічних вимог_1:
Реалізувати кластеризацію вхідних даних, отриманих Вами у ході виконання Дз_1, модельних та (або) реальних – на власний вибір. Методи Machine Learning з переліку: k-means (k-середніх); Support Vector Machine (машина опорних векторів); k-nearest neighbors (найближчих сусідів); ієрархічна кластеризація – для кластеризації обраних даних обрати самостійно. Провести аналіз та пояснення отриманих результатів, сформувати висновки.

Група технічних вимог_2:
Реалізувати кластеризацію за кольоровою ознакою об'єктів на самостійно обраному цифровому зображенні. Методи Machine Learning з переліку: k-means (k-середніх); Support Vector Machine (машина опорних векторів); k-nearest neighbors (найближчих сусідів) – для кластеризації обраного зображення обрати самостійно.
За необхідності провести покращення якості зображення: зміна кольору; підвищення контрасту; фільтрація, тощо. Етапи покращення якості та кластерізації повинні забезпечувати виділення геометричних або кольорових ознак обраного на цифровому зображенні об'єкту для його подальшої ідентифікації. Провести аналіз отриманих результатів, сформувати висновки.

Група технічних вимог_3:
Підрахувати кількість об'єктів на обраному цифровому зображенні. Об'єкти, що підлягають обрахунку обрати самостійно. Зміст етапів попередньої обробки зображень (корекція кольору, фільтрація, векторизація, кластеризація) має буди результатом R&D процесів, що конкретизується обраним зображенням і об'єктами для підрахунку. Провести аналіз отриманих результатів, сформувати висновки.

III. Результати виконання лабораторної роботи.

1. ***Реалізувати кластеризацію вхідних даних***

В лабораторній здійснюється обробка даних з електричного мікроскопу наведених за посиланням: https://www.kaggle.com/competitions/data-science-bowl-2018/data
Функції для зчитування Id та власне самих зображень.

```python
def image_ids_in(root_dir, ignore=[]):
    ids = []
    for id in os.listdir(root_dir):
        if id in ignore:
            print('Skipping ID:', id)
        else:
            ids.append(id)
    return ids
```

```python
def read_image(image_id, space="rgb"):
    image_file = STAGE1_TRAIN_IMAGE_PATTERN.format(image_id, image_id)
    image = skimage.io.imread(image_file)
    # Drop alpha which is not used
    image = image[:, :, :3]
    #hsv is hue saturation value format of the image
    if space == "hsv":
        image = skimage.color.rgb2hsv(image)
    return image
```

Рис. 1 – Функції для читання вхідних даних в заданому форматі(rgb, hsv)

Для успішного виконання поставленого завдання необхідно кластеризувати дані за найактивнішим кольором що присутній у зображенні. Для кластеризації обрано алгоритм K-means що є простим і добре пристосований до задач такого типу. Щоб знайти оптимальні параметри цього алгоритму використовуємо метод Ліктя. Вибрано декілька зразків даних і знайдено оптимальний параметр K для них використовуючи вище наведений алгоритм.

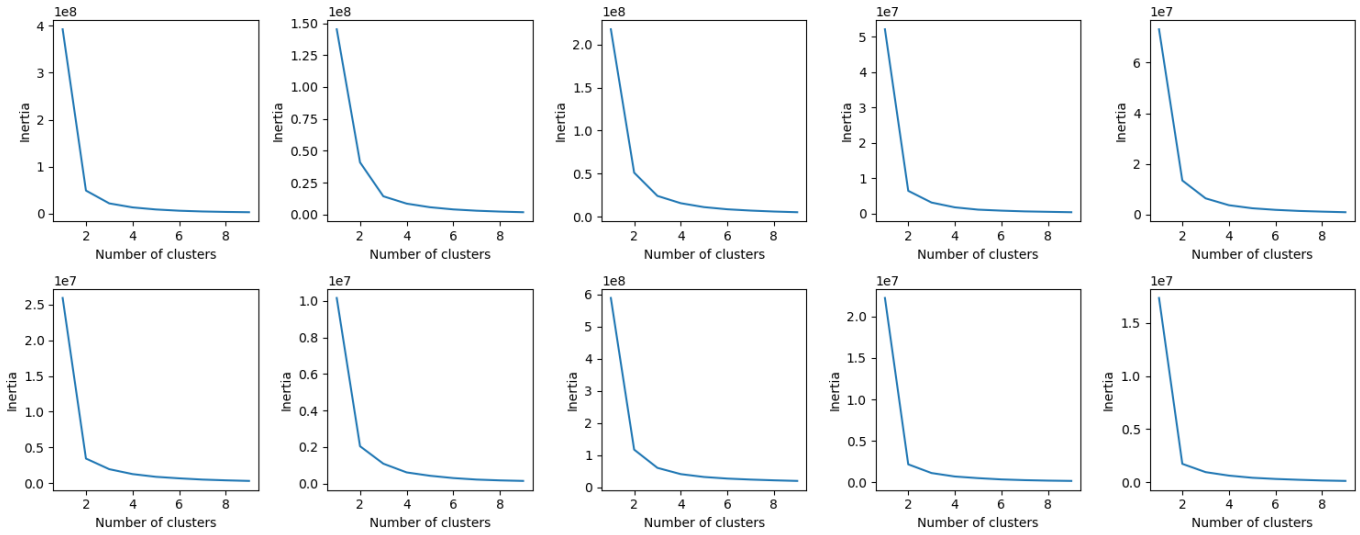

Рис. 2— Оптимальні значення параметра кластеризації.

```python
def get_domimant_colors(img, top_colors=2):
    img_l = img.reshape((img.shape[0] * img.shape[1], img.shape[2]))
    clt = KMeans(n_clusters = top_colors)
    clt.fit(img_l)
    # grab the number of different clusters and create a histogram
    # based on the number of pixels assigned to each cluster
    numLabels = np.arange(0, len(np.unique(clt.labels_)) + 1)
    (hist, _) = np.histogram(clt.labels_, bins = numLabels)
    # normalize the histogram, such that it sums to one
    hist = hist.astype("float")
    hist /= hist.sum()
    return clt.cluster_centers_, hist
```

Рис. 3— Функція що використовує оптимальний параметр кластеризаці за замовчуванням. Використовується для екстракції домінантного кольору зображення. Повертає гістограму топ кольорів у зображенні .

Для отримання залишкових даних про всі зображення у вигляді зображення таких як ширина та висота використовуємо:

```python
def get_images_details(image_ids):
    details = []
    for image_id in image_ids:
        image_hsv = read_image(image_id, space="hsv")
        height, width, l = image_hsv.shape
        dominant_colors_hsv, dominant_rates_hsv = get_domimant_colors(image_hsv, top_colors=1)
        dominant_colors_hsv = dominant_colors_hsv.reshape(1, dominant_colors_hsv.shape[0] * dominant_colors_hsv.sh
        info = [image_id, width, height, dominant_colors_hsv.squeeze()]
        details.append(info)
    return details
```

Рис. 4— Функція для завантаження даних в програму(все крім самого зображення)

```python
def plot_elbow_kmeans(data, k_max):
    scores = []
    for k in range(1, k_max):
        kmeans = KMeans(n_clusters=k)
        kmeans.fit(data)
        scores.append(kmeans.inertia_)


    plt.plot(range(1, k_max), scores)
    plt.xlabel('Number of clusters')
    plt.ylabel('Inertia')
```
[159]

Рис. 5— Функція для побудови графіку «ліктя».

Щоб визначити який параметр кластеризації використовувати для безпосередньо кластеризації зображень в датасеті по групах побудовано графік.

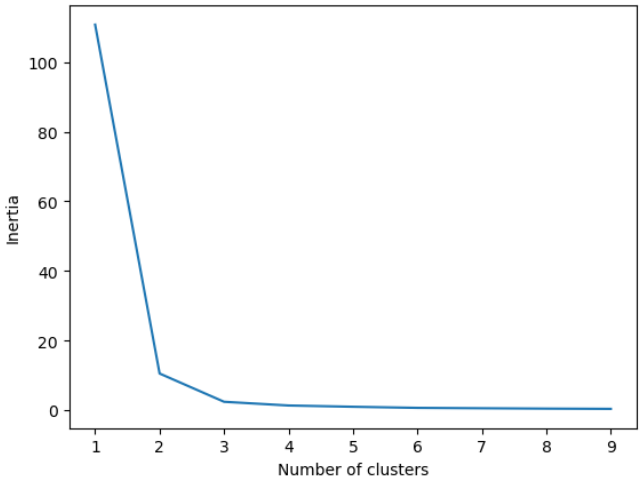

Рис. 6— Доцільно використовувати параметр 3 щоб отримати три групи зображень

```python
kmeans = KMeans(n_clusters=3).fit(X)
clusters = kmeans.predict(X)
train_df[HSV_CLUSTER] = clusters
train_df
```
[31] Python

```
c:\Users\vikto\miniconda3\envs\venv1\Lib\site-packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known
to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by
setting the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
```

	image_id	width	height	hsv_dominant	hsv_cluster
0	00071198d059ba7f5914a526d124d28e6d010c92466da2...	256	256	[0.0, 0.0, 0.02408716538373055]	0
1	003cee89357d9fe13516167fd67b609a164651b2193458...	256	256	[0.0, 0.0, 0.035367060642616806]	0
2	00ae65c1c6631ae6f2be1a449902976e6eb8483bf6b074...	320	256	[0.6969922721241343, 0.2417388976327306, 0.760...	1
3	0121d6759c5adb290c8e828fc882f37dfaf3663ec885c6...	320	256	[0.6439642758207026, 0.21017601724122487, 0.80...	1
4	01d44a26f6680c42ba94c9bc6339228579a95d0e2695b1...	320	256	[0.7435220601544179, 0.2324891863323699, 0.786...	1
...	...	...	...	...	...
665	fec226e45f49ab81ab71e0eaa1248ba09b56a328338dce...	256	256	[0.0, 0.0, 0.06294316310508558]	0
666	feffce59a1a3eb0a6a05992bb7423c39c7d52865846da3...	256	256	[0.0, 0.0, 0.0762871237362146]	0
667	ff3407842ada5bc18be79ae453e5bdaa1b68afc842fc22...	696	520	[0.0, 0.0, 0.06788620819681142]	0
668	ff3e512b5fb860e5855d0c05b6cf5a6bcc7792e4be1f0b...	256	256	[0.0, 0.0, 0.015257891486672656]	0
669	ff599c7301daa1f783924ac8cbe3ce7b42878f15a39c2d...	360	360	[0.0, 0.0, 0.0475662672476387]	0

Рис. 7— Датафрейм з інформацією про зображення

Для кожного кластеру виведено нну кількість зображень що належать цьому кластеру на екран.

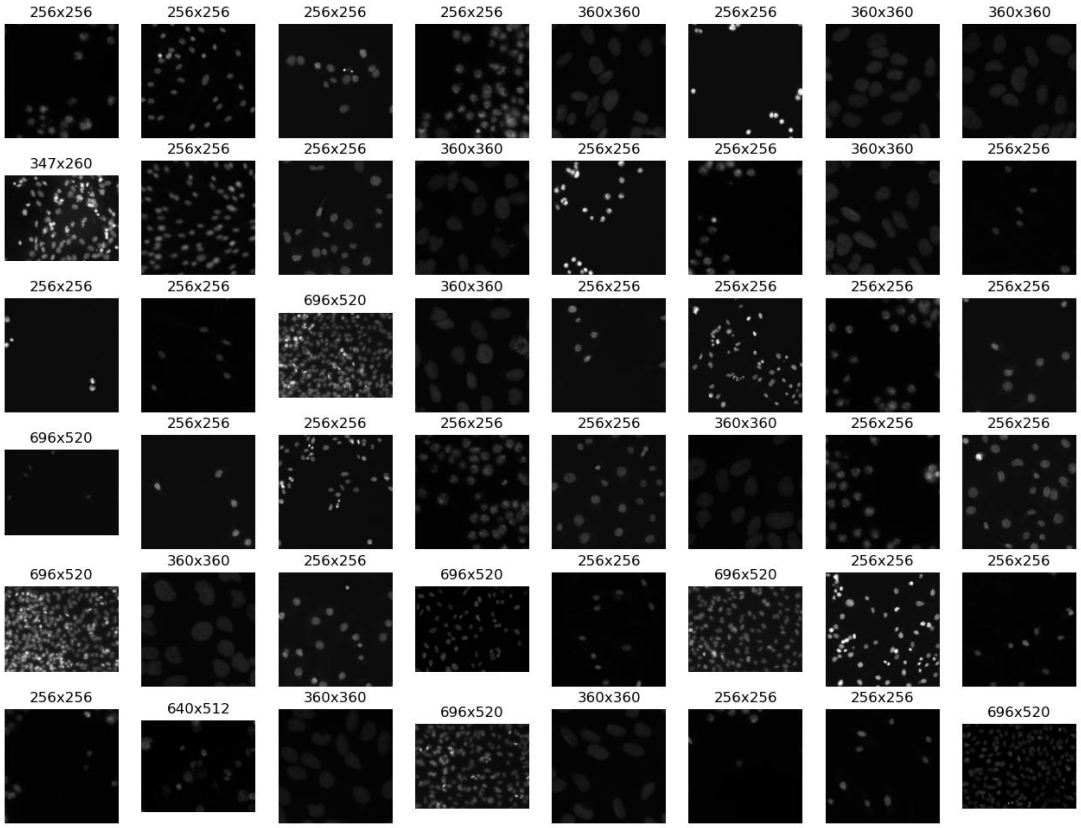

Рис. 8—Кластер 1.

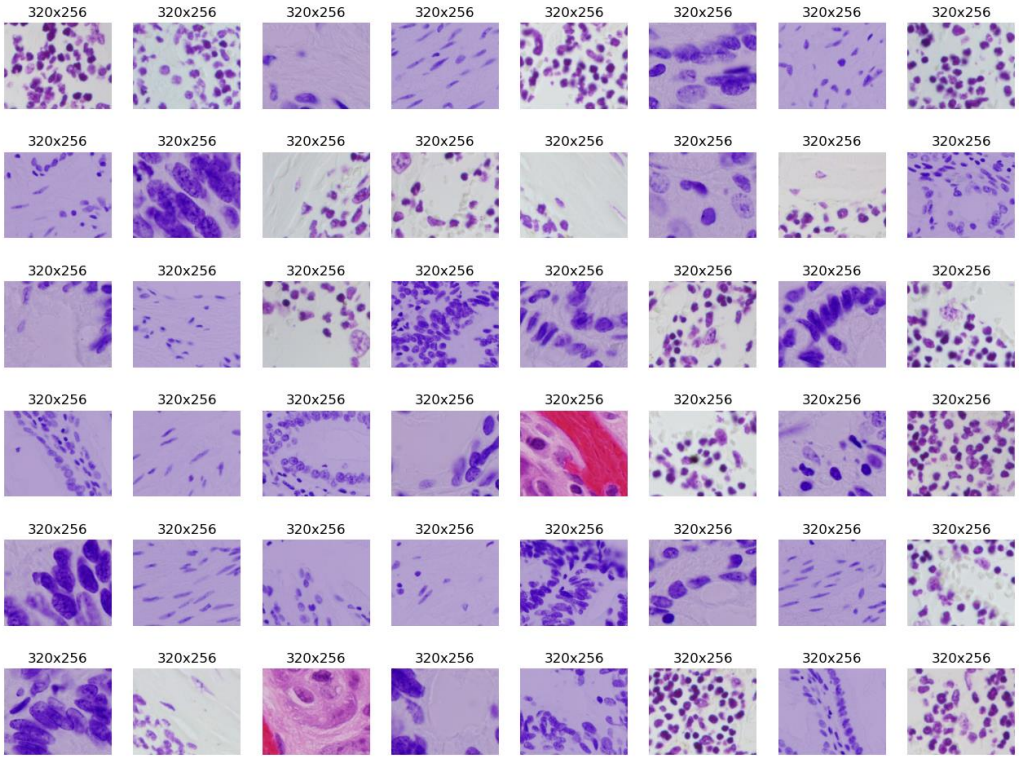

Рис. 9—Кластер 2.

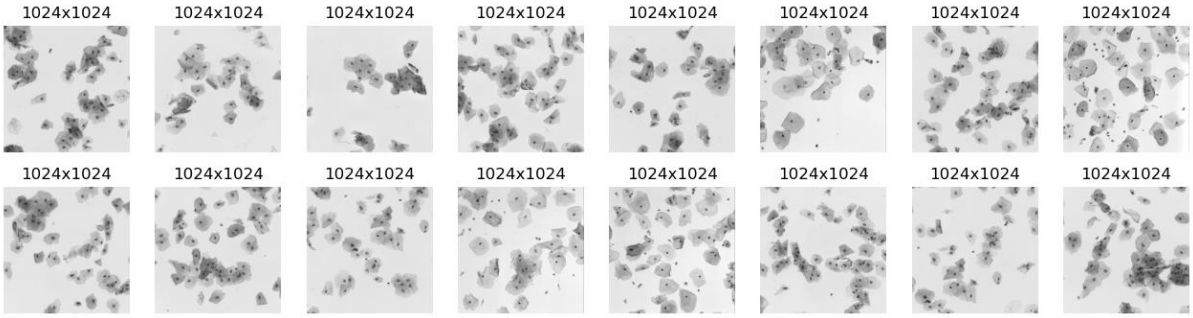

Рис. 10—Кластер 3

Отже як видно з результату кластеризація пройшла успішно. Присутній дизбаланс в класах але загалом зображення розподілені за домінуючим кольором зображення.

2. ***Реалізувати кластеризацію за кольоровою ознакою об'єктів на самостійно обраному цифровому зображенні.***

Для аналізу використано перше зображення з другого кластеру в датасеті.

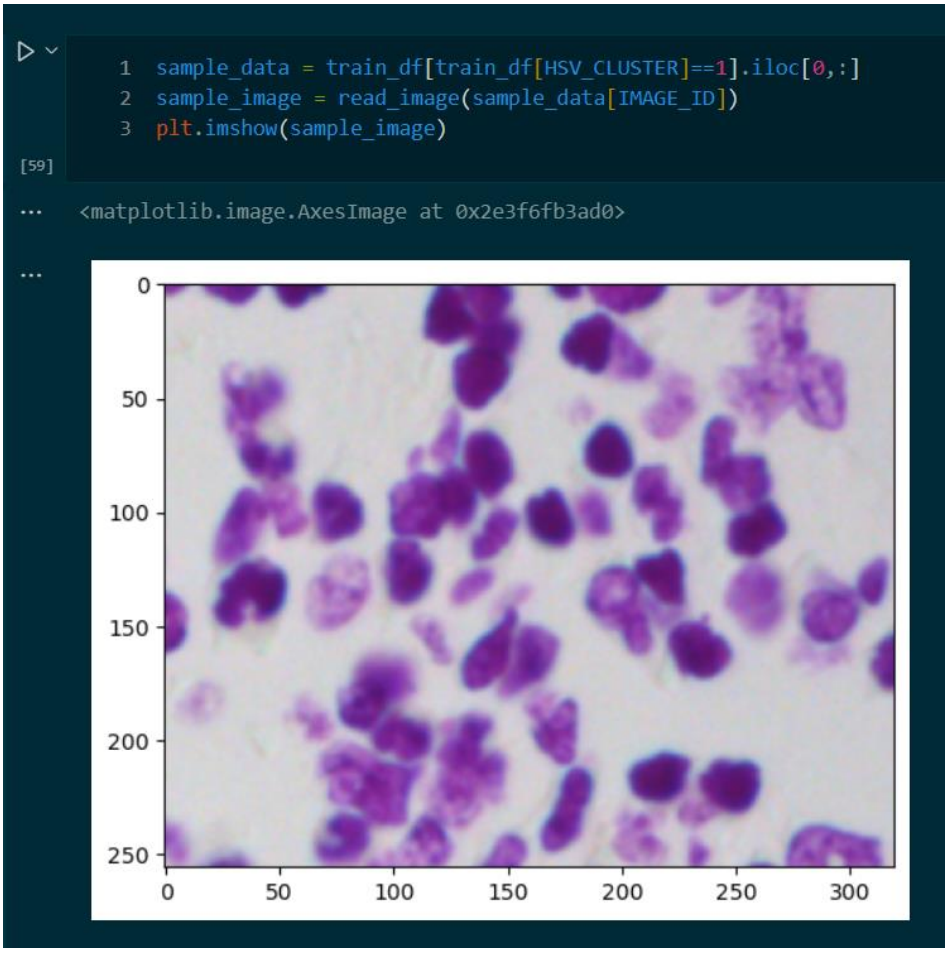

Рис. 11— Зображення для кластеризації кольорів.

Щоб краще розуміти що саме відбувається зображено всі пікселі присутні в зображенні на 3D графіку шкалами якого є відповідно R G та B показники зображення

```python
def plot_pixels(image_reshaped, title, colors=None, N=10000):
    if colors is None:
        colors = image_reshaped

    # choose a random subset
    rng = np.random.RandomState(0)
    i = rng.permutation(image_reshaped.shape[0])[:N]
    colors = colors[i]
    R, G, B = image_reshaped[i].T

    fig = plt.figure(figsize=(10, 8))
    ax = fig.add_subplot(111, projection='3d')
    ax.scatter(R, G, B, c=colors, marker='.')
    ax.set(xlabel='Red', ylabel='Green', zlabel='Blue')
    ax.set_title(title)
```

Рис. 12— Функція для побудови графіку пікселів за кольорами.

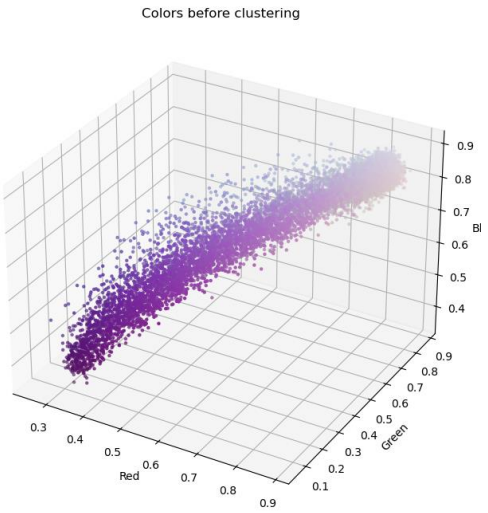

Рис. 13— перед застосуванням кластеризації пікселі зображення мають вигляд.

Для того щоб обрати оптимальний параметр кластеризації знову використано лікоть.

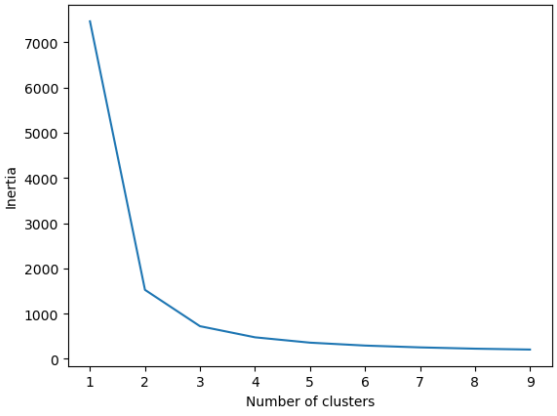

Рис. 14— оптимальний параметр для кластеризації 3

```python
clusters_amount = 3
kmeans = KMeans(clusters_amount).fit(sample_image_reshaped)
new_colors = kmeans.cluster_centers_[kmeans.predict(sample_
plot_pixels(sample_image_reshaped, colors=new_colors,
            title="Reduced color space")
```

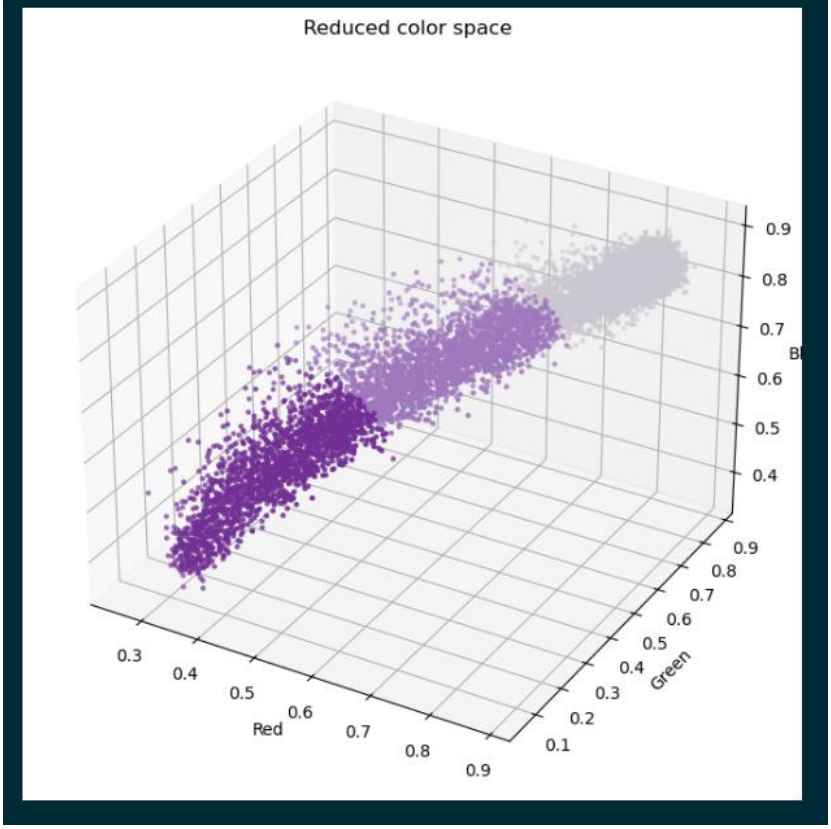

Рис. 15 —Пікселі зображення забарвлені кольором свого кластера.

```python
sample_image_color_clust = new_colors.reshape(sample_image.shape)

fig, ax = plt.subplots(1, 2, figsize=(16, 6),subplot_kw=dict(xticks=[], yticks=[]))
fig.subplots_adjust(wspace=0.05)
ax[0].imshow(sample_image)
ax[0].set_title('Original Image', size=16)
ax[1].imshow(sample_image_color_clust)
ax[1].set_title('Clustered image', size=16)
```

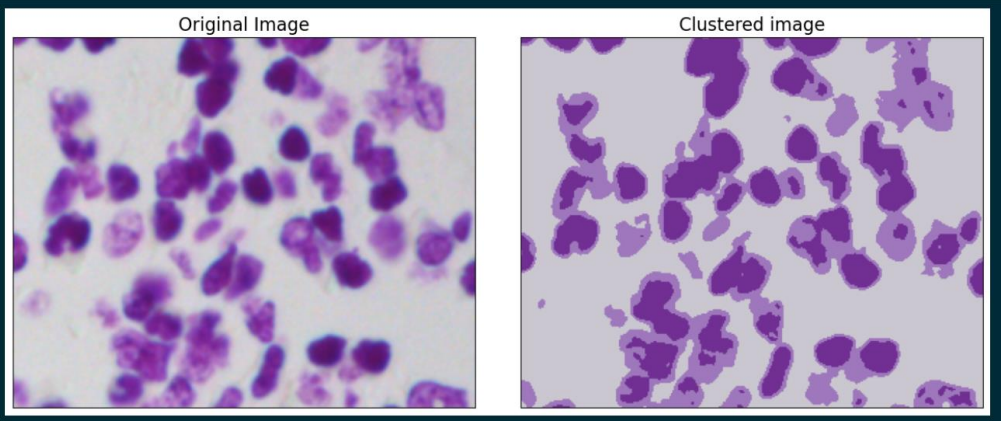

Рис. 16 — Реконструйоване зображення в зжатому просторі кольорів.

Залежно від кількості кластерів зображення буде використовувати різну кількість базових кольорів. В ході цієї лабораторної роботи використано алгоритмом підбору параметрів K-means для зменшення втрат під час стиснення зображення.

3. ***Підрахувати кількість об'єктів на обраному цифровому зображенні. Об'єкти, що підлягають обрахунку обрати самостійно.***

Об'єктом підрахунку є клітини що відображені на індивідуальному зображення. Для обробки використано те саме зображення що і в попередньому завданні. Для підрахунку обрано використовувати алгоритм K-means наступним чином. До dataframe використаного в попередньому завданні для кластеризації за кольорами додано також координати кожного пікселя в зображенні. Після цього локалізовано всі пікселі зображення що не належать до заднього плану в окремий датафрейм. На цьому датафреймі що містить лише пікселі клітин проведено пошук за відстанню. Таким чином знайдено кластери для пікселів клітин базуючись на їх близькості. Після підбору оптимального параметра кластеризації цей параметр за значенням дорівнює кількості клітин що видно на фото.

Рис. 17 — три базові кольори кластеризованого зображення.

```
pixels_df = pd.DataFrame(new_colors)
pixels_df.columns = ['R', 'G', 'B']
pixels_df['Position'] = [[i//sample_data

pixels_df
```

[153]

	R	G	B	Position
0	0.446170	0.186919	0.58271	[0, 0]
1	0.446170	0.186919	0.58271	[0, 1]
2	0.446170	0.186919	0.58271	[0, 2]
3	0.446170	0.186919	0.58271	[0, 3]
4	0.446170	0.186919	0.58271	[0, 4]
...	...	...	...	...
81915	0.622792	0.469767	0.74089	[255, 315]
81916	0.797796	0.781329	0.81663	[255, 316]
81917	0.797796	0.781329	0.81663	[255, 317]
81918	0.797796	0.781329	0.81663	[255, 318]
81919	0.797796	0.781329	0.81663	[255, 319]

Рис. 18 — dataframe з доданою інформацією про положення пікселя.

	R	G	B	Position
0	0.446170	0.186919	0.58271	[0, 0]
1	0.446170	0.186919	0.58271	[0, 1]
2	0.446170	0.186919	0.58271	[0, 2]
3	0.446170	0.186919	0.58271	[0, 3]
4	0.446170	0.186919	0.58271	[0, 4]
...	...	...	...	...
81911	0.622792	0.469767	0.74089	[255, 311]
81912	0.622792	0.469767	0.74089	[255, 312]
81913	0.622792	0.469767	0.74089	[255, 313]
81914	0.622792	0.469767	0.74089	[255, 314]
81915	0.622792	0.469767	0.74089	[255, 315]

32470 rows × 4 columns

Рис. 19 — Пікселі що є частиною клітини.

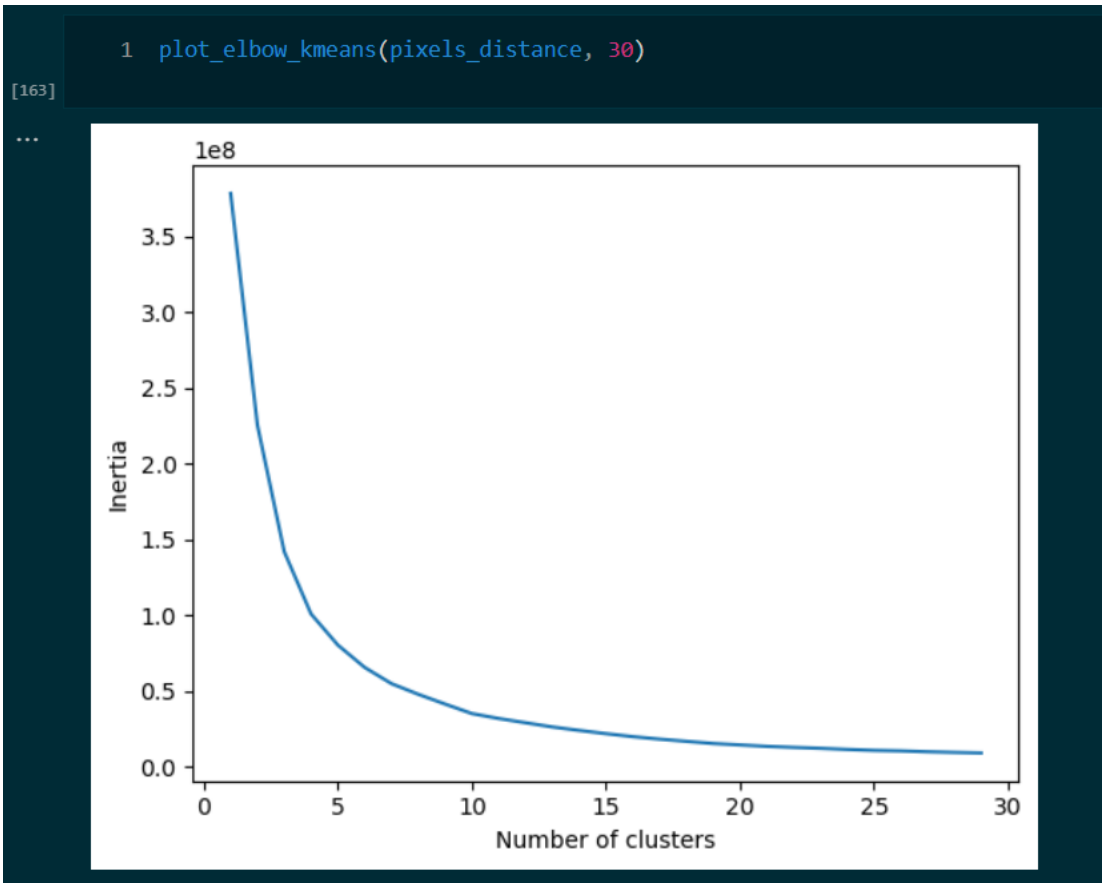

Рис. 20 — Пошук оптимального параметра кластеризації з використанням ліктя. Оптимальним є значення параметра 30

```python
merged_df = pixels_df.merge(non_background_pixels_df[['distance_cluster']], left_index=True, right_index=True, how='left')

# Find the maximum cluster number in non_background_pixels_df
max_cluster = non_background_pixels_df['distance_cluster'].max()

# Replace NaN values with a new cluster number
merged_df['distance_cluster'].fillna(max_cluster + 1, inplace=True)
merged_df
```

	R	G	B	Position	distance_cluster
0	0.446170	0.186919	0.58271	[0, 0]	36.0
1	0.446170	0.186919	0.58271	[0, 1]	36.0
2	0.446170	0.186919	0.58271	[0, 2]	36.0
3	0.446170	0.186919	0.58271	[0, 3]	36.0
4	0.446170	0.186919	0.58271	[0, 4]	36.0
...	...	...	...	...	...
81915	0.622792	0.469767	0.74089	[255, 315]	12.0
81916	0.797796	0.781329	0.81663	[255, 316]	45.0
81917	0.797796	0.781329	0.81663	[255, 317]	45.0
81918	0.797796	0.781329	0.81663	[255, 318]	45.0
81919	0.797796	0.781329	0.81663	[255, 319]	45.0

Рис. 20 — Комбінація dataframe без та з кластеризацією за відстанню в один. Присвоєння пікселям фона окремого кластеру.

```python
def random_color():
    return [np.random.randint(0, 256) for _ in range(3)]

# Map each unique distance_cluster value to a random color
color_map = {cluster: random_color() for cluster in merged_df['distance_cluster'].unique()}
merged_df['color'] = merged_df['distance_cluster'].map(color_map)

sample_image_dist_clust = np.array(merged_df['color'].to_list()).reshape(sample_image.shape)

fig, ax = plt.subplots(1, 2, figsize=(16, 6),subplot_kw=dict(xticks=[], yticks=[]))
fig.subplots_adjust(wspace=0.05)
ax[0].imshow(sample_image)
ax[0].set_title('Original Image', size=16)
ax[1].imshow(sample_image_dist_clust)
ax[1].set_title('Distance Clustered image', size=16);
```

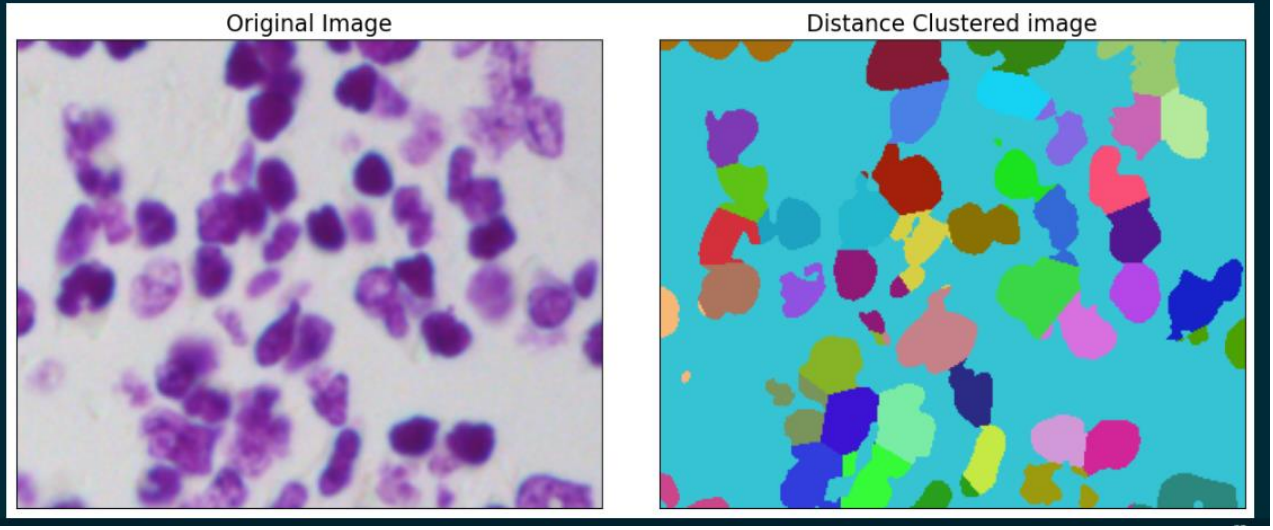

Рис. 21 — Візуалізація кластерів за відстанню у вигляді зображення з рандомних кольорів для кожного окремо взятого кластера

IV. Висновки.

Результати лабораторної роботи показують потенціал і багатофункціональність застовування базового алгоритму K-means для досить складних процесів обробки зображень. Як для кластиризації зображень, так і для виділення features конкретно взятого зображення. Також його можна використовувати для побудови алгоритмів компресії, для стилізації зображень і в безлічі інших галузей.

Виконав: студент ФІОТ Лошак В.І. ІП-11