BỘ THÔNG TIN VÀ TRUYỀN THÔNG

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

BÁO CÁO BÀI TẬP LỚN

MÔN: lẬP TRÌNH VỚI PYTHON

Giảng viên: Kim Ngọc Bách

Sinh viên: Nguyễn Đức Anh

Mã sinh viên: B22DCCN027

Email: anh666666anh@gmail.com

Nhóm học phần: 11

1. Phần 1
   - File thực hiện thu thập dữ liệu phân tích cầu thủ tại giải bóng đá ngoại hạng Anh mùa 2023-2024 nằm tại: NguyenDucAnh_B22DCCN027/code/part1/fetchPlayerData.py
   - Thời gian file thực hiện trung bình: 45s
   - Trong fetchPlayerData.py
   - Hàm get_table thu thập dữ liệu từ web có tham số là url: str và trả về 1 Data Frame
   - Trong hàm get_table có chứa:

```
page = requests.get(url)
soup = BS(page.text, 'html.parser')

comments = soup.find_all(string=lambda text: isinstance(text, Comment))

comment_soup = BS(comments[28], 'html.parser')

table = comment_soup.find('table')
```

➔ Sử dụng 2 thư viện requests, beutifulsoup4 để quét dữ liệu
➔ Nội dung của 10 bảng nằm trong web được lưu dưới dạng comment của mã nguồn HTML và được đặt ở comment thứ 28

```
thead = table.find('thead')

rows = thead.find_all('tr')

prefixs = []
heads = rows[0].find_all('th')
for head in heads:
    if num := head.get('colspan'):
        content = head.text.replace(' ', '_')
        prefixs.append((int(num), content))

prefixs.append((1, ''))

columns = []
heads = rows[-1].find_all('th')

i = 0
for num, prefix in prefixs:
    for j in range(i, i + num):
        field = prefix + ('' if prefix == '' else '_') + heads[j].text.replace(' ', '')
        columns.append(field)
    i += num
```

➔ Vì phần tiêu đề chính bao gồm nhiều tiêu đề phụ và mỗi tiêu đề phụ có tên giống nhau ➔ nên cần dùng tiêu đề chính làm tiền tố để xác định rõ thông tin cột để tránh sự trùng lặp gây ra lỗi trong các thao tác sau này

```python
    df = pd.DataFrame(columns=columns[1:])

    # handling body
    tbody = table.find('tbody')

    rows = tbody.find_all('tr')
    i = 0

    for row in rows:
        datas = row.find_all('td')
        if len(datas) != 0:
            datas = [data.text for data in datas]
            df.loc[i] = datas
            i += 1

    return df
```

➔ Tạo 1 Data Frame để trả về và lấy nội dung của từng cầu thủ đưa vào data frame (kết thúc hàm get_table)

```python
tables = []
urls = [
    'https://fbref.com/en/comps/9/2023-2024/stats/2023-2024-Premier-League-Stats',
    'https://fbref.com/en/comps/9/2023-2024/keepers/2023-2024-Premier-League-Stats',
    'https://fbref.com/en/comps/9/2023-2024/shooting/2023-2024-Premier-League-Stats',
    'https://fbref.com/en/comps/9/2023-2024/passing/2023-2024-Premier-League-Stats',
    'https://fbref.com/en/comps/9/2023-2024/passing_types/2023-2024-Premier-League-Stats',
    'https://fbref.com/en/comps/9/2023-2024/gca/2023-2024-Premier-League-Stats',
    'https://fbref.com/en/comps/9/2023-2024/defense/2023-2024-Premier-League-Stats',
    'https://fbref.com/en/comps/9/2023-2024/possession/2023-2024-Premier-League-Stats',
    'https://fbref.com/en/comps/9/2023-2024/playingtime/2023-2024-Premier-League-Stats',
    'https://fbref.com/en/comps/9/2023-2024/misc/2023-2024-Premier-League-Stats'
]

for i, url in enumerate(urls):
    print(i)
    table = get_table(url)
    tables.append(table)
```

➔ Tạo 1 list table để lưu trữ danh sách data frame của từng bảng tương ứng với url trong list urls

Trong file chứa list columns là thông tin các cột mà đề bài yêu cầu

```
for i, table in enumerate(tables):
    new_table = table.filter(items=columns[i][1], axis=1)
    tables[i] = new_table

common_columns = ['Player', 'Nation', 'Pos', 'Squad', 'Age']

results = tables[0]

for i in range(1, len(tables)):
    results = results.merge(tables[i], how = 'outer', on = common_columns)
```

➔ thực hiện thao tác join để gộp 10 bảng lại với nhau dựa trên cột chung của 10 bảng được liệt kê trong danh sách common_columns

```
df = results.copy()
df = df.convert_dtypes()
df.info()
df['Playing_Time_Min'] = df['Playing_Time_Min'].str.replace(',', '').astype('Int64')

for column in df.columns[4:]:
    df[column] = pd.to_numeric(df[column], errors='coerce')


df = df[df['Playing_Time_Min'] > 90]
df['firstname'] = df['Player'].str.split().str[0].astype('string')

df_sorted = df.sort_values(by=['firstname', 'Age'], ascending=[True, False])

df_sorted.drop('firstname', axis=1, inplace=True)
df_sorted.to_csv('results.csv', index=False)
```

➔ chuyển đổi thành các loại dữ liệu thích hợp để so sánh và sử dụng sau này

➔ tìm kiếm và sắp xếp theo các yêu cầu của đề bài và lưu file

2. Phần 2

- Các file nằm trong NguyenDucAnh_B22DCCN027/code/part2
- File top3_and_result2.py thực hiện việc tìm top 3 cầu thủ có chỉ số cao nhất và thấp nhất ở mỗi chỉ số + tạo ra file result2.csv

```python
import pandas as pd
import numpy as np

table = pd.read_csv('../part1/results.csv')


Run Cell | Run Above | Debug Cell
# %%
fields = ['']
row = ['all']

for column in table.columns[4:]:
    table.sort_values(by=column, ascending=False, inplace=True)

    datas = table.dropna(axis=0, subset=[column])

    print(datas.head(3).get(['Player', 'Nation', 'Pos', 'Squad', column]))
    print(datas.tail(3)[::-1].get(['Player', 'Nation', 'Pos', 'Squad', column]))

    fields.append(f"Median of {column}")
    fields.append(f"Mean of {column}")
    fields.append(f"Std of {column}")

    arr = np.array(table[column].dropna())

    row.append(np.median(arr))
    row.append(np.mean(arr))
    row.append(np.std(arr))
```

➔ thực hiện việc xắp xếp để tìm cầu thủ 1 cách thuận tiện và sau đó in ra màn hình. Nhưng vì có tận 172 * 6 dòng nên em đã tạo ra file top_3_player.txt cùng nằm trong thư mục để in ra chi tiết về các cầu thủ

➔ tạo danh sách row để lưu thông tin trung bình, trung vị, độ lệch tiêu chuẩn của mỗi cột

```python
df = pd.DataFrame(columns=fields)

Run Cell | Run Above | Debug Cell
# %%
df.loc[len(df)] = row

teams = table['Squad'].unique()


Run Cell | Run Above | Debug Cell
# %%
rows = [[f"{team}"] for team in teams]

for i, team in enumerate(teams):
    stats = table[table['Squad'] == team].drop(columns=['Player', 'Nation', 'Pos', 'Squad'])
    medians = stats.median().to_list()
    means = stats.mean().to_list()
    stds = stats.std().to_list()

    for j in range(len(medians)):
        rows[i].append(medians[j])
        rows[i].append(means[j])
        rows[i].append(stds[j])


Run Cell | Run Above | Debug Cell
# %%
for row in rows:
    df.loc[len(df)] = row

Run Cell | Run Above | Debug Cell
# %%
df.to_csv('results2.csv')
```

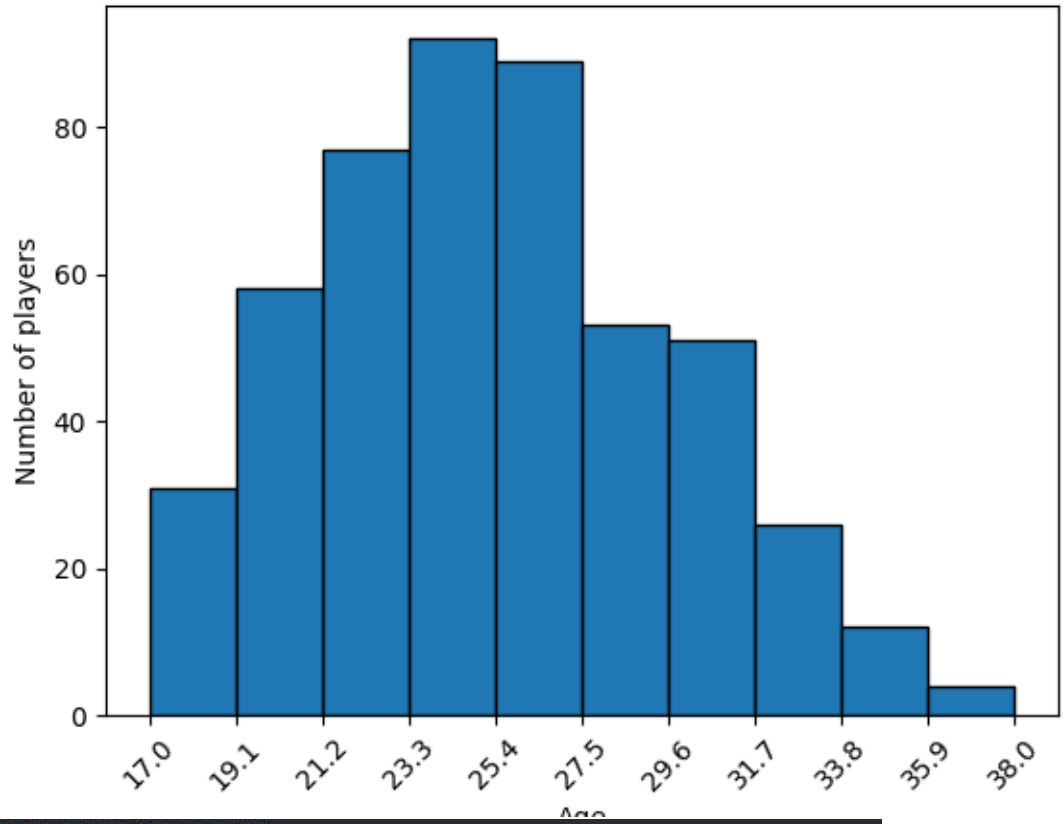
➔ tạo data frame để lưu thông tin trung bình, trung vị, độ lệch chuẩn của tất cả cầu thủ, mỗi đội và lưu file result2.csv

➔ tìm tất cả các đội lưu trong list teams và duyệt qua mỗi đội loại trừ đi 4 cột không phải là số ở đầu để thực hiện việc tính toán ➔ lưu file result2.csv

➔cột thứ 2 của result2.csv là tên của các đội + all(đại diện cho toàn bộ cầu thủ)

- File histogram_for_players.py phân bố chỉ số trong toàn giải và mỗi đội vì có 172 chỉ số nên cần 172 đồ thị nên ảnh được lưu ở "NguyenDucAnh_B22DCCN027/images/all_players" và tên của mỗi bức ảnh là chỉ số cần xem xét

- Ví dụ: về biểu đồ histogram về chỉ số tuổi và toàn bộ cầu thủ




```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

table = pd.read_csv('../part1/results.csv')


for column in table.columns[4:]:
    data = table[column].dropna(axis=0)

    _, bins_edges, __ = plt.hist(data, edgecolor='black')
    plt.ylabel('Number of players')
    plt.xlabel(column.replace('_', ' '))
    plt.xticks(bins_edges, rotation=45)
    column = column.replace("/", "_divide_")
    plt.savefig(f"../../images/all_players/{column}.png")
    plt.clf()
```

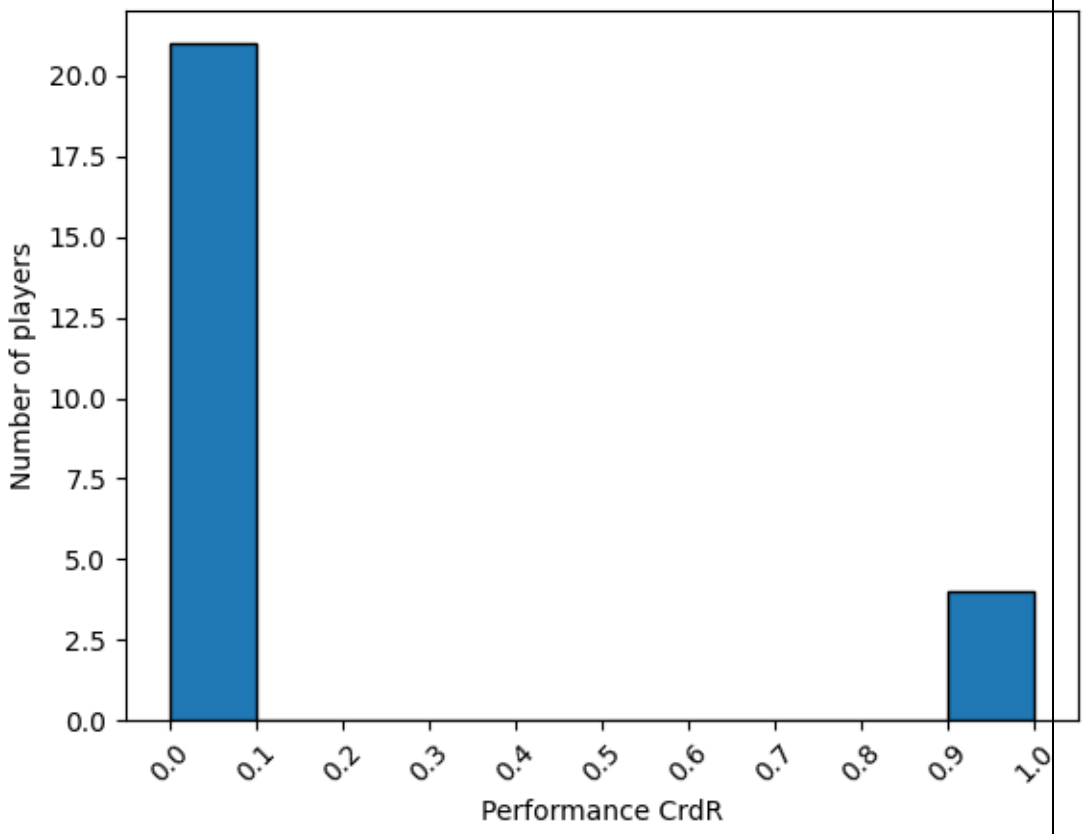
- ➔ lưu file đến NguyenDucAnh_B22DCCN027/images/all_players và xóa ảnh để đảm bảo ảnh hiển thị đúng kết quả,
  ➔ thực hiện quay giá trị của trục x để xem giá trị rõ hơn

➔ tên file ảnh là tên cột cần so sánh

- File histogram_for_teams.py thực hiện tạo ra biểu đồ mỗi chỉ số của mỗi đội vì có 20 đội và mỗi đội cần 172 biểu đồ cho mỗi chỉ số nên kết quả được lưu vào thư mục "NguyenDucAnh_B22DCCN027/images/teams" với dạng

  [tên đội] [chỉ số].png(ảnh chứa thông tin về biểu đồ của [tên đội] với chỉ số so sánh là [chỉ số]

  - Ví dụ: về số lượng cầu thủ nhận thẻ đỏ của wolves



- File best_team.py Tìm đội bóng có chỉ số điểm số cao nhất ở mỗi chỉsố và tìm kiếm đội nào có phong độ tốt nhất giải ngoại Hạng Anh mùa 2023-2024

```python
import pandas as pd

df = pd.read_csv('../part1/results.csv')


columns_to_sum = df.columns[5:].tolist()
df[columns_to_sum] = df[columns_to_sum].apply(pd.to_numeric, errors='coerce')


team_stats = df.groupby('Squad')[columns_to_sum].sum()


highest_team_stats = team_stats.idxmax()
highest_team_stats.to_csv('best_squad.csv')
print(highest_team_stats.value_counts())
```

➔ thực hiện tìm kiếm đội tốt nhất ở mỗi chỉ số và lưu vào file best_squad.csv với cột 2 là đội có chỉ số cao nhất ở mỗi chỉ số

➔ kết quả khi chạy file này có kết quả như sau

```
Manchester City     39
Liverpool           26
Tottenham           11
Bournemouth         10
Sheffield Utd        9
Chelsea              8
Brighton             8
Newcastle Utd        7
Nott'ham Forest      7
Arsenal              7
Everton              6
Crystal Palace       6
West Ham             5
Brentford            4
Luton Town           4
Manchester Utd       3
Wolves               3
Burnley              2
Aston Villa          1
Fulham               1
Name: count, dtype: int64
```

➔ qua đó có thể thấy MC dẫn đầu với 39 chỉ số cao nhất nên có thể kết luận Manchester City là đội tốt nhất trong giải ngoại hạng anh này

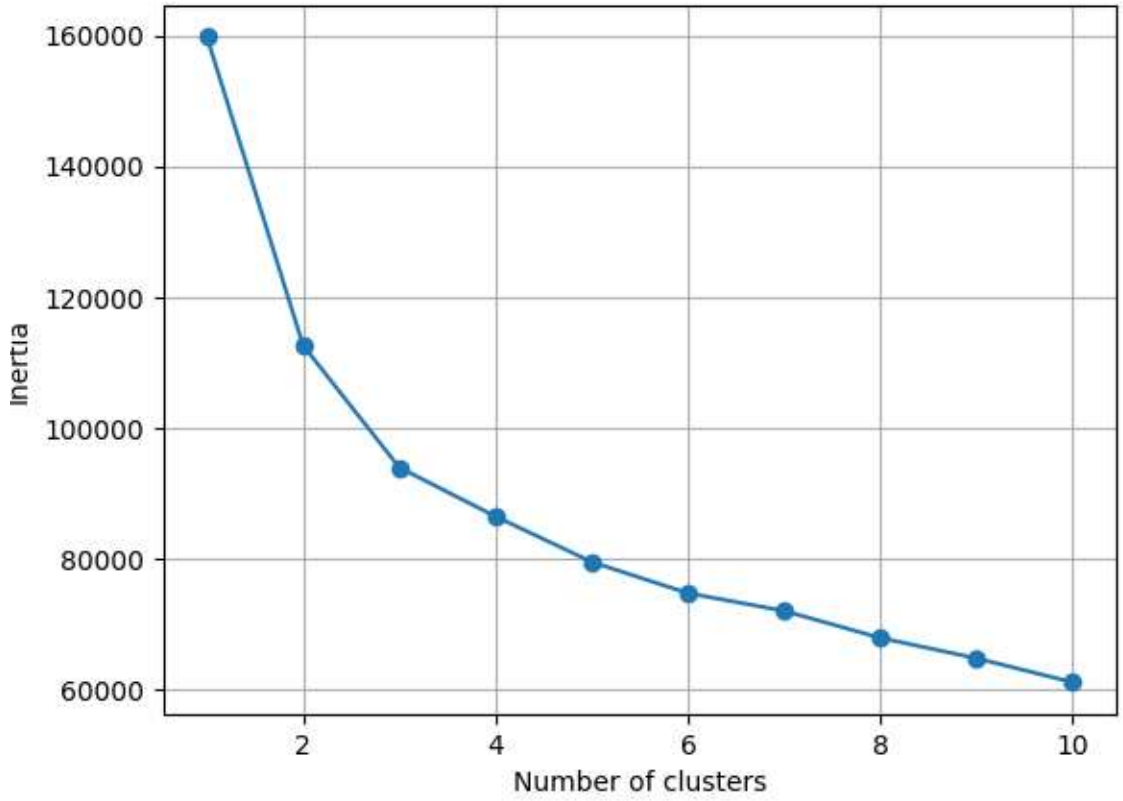
3. Phần 3
   - Các file nằm trong NguyenDucAnh_B22DCCN027/code/part3
   - File get_optimise_k_means.py thực hiện tìm kiếm k tối ưu nhất để dùng cho việc phân cụm trong thuật toán k-means, em đã áp dụng phương pháp elbow để tìm kiếm k tối ưu. Trong file chứa các mã sau

```python
import pandas as pd
import numpy as np

df = pd.read_csv('../part1/results.csv')
data = df[df.columns[4:]].copy()
data.dropna(axis='columns', inplace=True)

Run Cell | Run Above
# %% [markdown]
# 1. scale the data to standardize values

Run Cell | Run Above | Debug Cell
# %%
data = ((data - data.min()) / (data.max() - data.min())) * 9 + 1
```

➔Sau khi đọc dữ liệu từ file em đã loại trừ 4 cột đầu không phải là số để so sánh và thực hiện việc chuẩn hóa dữ liệu về 1 phạm vi nhất định. Pham vi mà em dùng ở đây là [1, 9] để chuẩn hóa dữ liệu

```python
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
from IPython.display import clear_output



Run Cell | Run Above
# %% [markdown]
# ### Using The elbow method to identify optimis

Run Cell | Run Above | Debug Cell
# %%
from sklearn.cluster import KMeans

pca = PCA(n_components=2)
data_2d = pca.fit_transform(data)

def optimise_k_means(data_2d, max_k):
    means = []
    inertias = []

    for k in range(1, max_k + 1):
        kmeans = KMeans(n_clusters=k)
        kmeans.fit(data)

        means.append(k)
        inertias.append(kmeans.inertia_)

    plt.plot(means, inertias, 'o-')
    plt.xlabel("Number of clusters")
    plt.ylabel("Inertia")
    plt.grid(True)
    plt.savefig("../../images/optimise k.png")
    plt.show()

optimise_k_means(data_2d, 10)
```

➔ sau khi chuẩn hóa dữ liệu, thực hiện giảm chiều dữ liệu với thuật toán PCA, và dùng thư viện sklearn dùng model Kmean cộng với mathplotlib.pyplot để vẽ biểu đồ elbow để xác định k, phạm vi k mà em chọn ở đây là [1, 10] kết quả sẽ được hiển thị ra màn hình và được lưu trong folder images với tên optimise k.png có hình ảnh như sau



⇨ Qua đó có thể thấy được điểm gãy tại 3 và từ đó có thể xác định được k = 3 là tối ưu nhất cho dữ liệu các cầu thủ này

- File kmean_and_pca.py thực hiện 2 yêu cầu của đề bài

  1.Sử dụng thuật toán K-means để phân loại các cầu thủ thành các nhóm có chỉ số giống nhau.

  2. Sử dụng thuật toán PCA, giảm số chiều dữ liệu xuống 2 chiều, vẽ hình phân cụm các điểm dữ liệu trên mặt 2D

  - Sau khi thực thi file thì nó sẽ hiện lên biểu đồ phân cụm của file và 1 bức ảnh tự động lưu ở folder images có tên là kmean pca.png
  - Chi tiết thực hiện của file

```python
import pandas as pd  # type: ignore
import numpy as np  # type: ignore

df = pd.read_csv('../part1/results.csv')
data = df[df.columns[4:]].copy()
data.dropna(axis='columns', inplace=True)

Run Cell | Run Above
# %% [markdown]
# 1. scale the data to standardize values

Run Cell | Run Above | Debug Cell
# %%
data = ((data - data.min()) / (data.max() - data.min())) * 9 + 1
```

➔ đọc dữ liệu từ results.csv và loại bỏ 4 cột đầu không phải là số và sau đó chuẩn hóa dữ liệu về phạm vi [1, 9]

```python
def random_centroids(data, k):
    centroids = []

    for _ in range(k):
        centroid = data.apply(lambda x: float(x.sample().iloc[0]))
        centroids.append(centroid)

    return pd.concat(centroids, axis=1)
```

➔ hàm tạo ra k tâm ngẫu nhiên của dữ liệu data với data và k là tham số

➔ ví dụ về 3 tâm ngẫu nhiên có kết quả trả về Data frame như sau

	0	1	2
Age	3.571429	4.857143	3.142857
Playing_Time_MP	7.500000	9.250000	4.750000
Playing_Time_Starts	2.184211	1.710526	1.000000
Playing_Time_Min	1.216346	8.553185	5.610877
Performance_Ast	1.000000	1.000000	2.384615
...	...	...	...
Performance_Crs	1.211765	1.158824	1.211765
Performance_OG	5.500000	1.000000	1.000000
Performance_Recov	1.797468	2.291139	2.670886
Aerial_Duels_Won	2.478571	1.707143	2.157143
Aerial_Duels_Lost	1.798817	2.331361	3.236686

```python
# 3. get labels for each data point

Run Cell | Run Above | Debug Cell
# %%
def get_labels(data, centroids):
    distances = centroids.apply(lambda x: np.sqrt(((data - x) ** 2).sum(axis=1)))
    return distances.idxmin(axis=1)
```

- ➔ hàm lấy dữ liệu và vị trí của tâm trong biểu đồ để tính toán xử lý khoảng cách và sau đó phân loại điểm dữ liệu đó sẽ thuộc về cụm nào
  - ⇨ Ví dụ về cụm của dữ liệu của các centroids đã đề cập ở bên trên

```
0      0
1      0
2      1
3      0
4      0
      ..
488    1
489    1
490    1
491    1
492    0
Length: 493, dtype: int64
```

⇨

⇨ Với cột đầu tiên là số thứ tự dòng mà dataframe tự xác định trong data frame và cột thứ 2 biểu diễn dòng nào thuộc cụm nào

4. create new centroids based on mean values of each cluster

```python
def new_centroids(data, labels, k):
    centroids = data.groupby(labels).apply(lambda x: np.exp(np.log(x).mean())).T


    return centroids
```

- ⇨ Hàm tạo centroids mới dựa vào sự phân cụm của các điểm dữ liệu sau khi gọi hàm get_labels

```python
def plot_clusters(data, labels, centroids, iteration):
    pca = PCA(n_components=2)
    data_2d = pca.fit_transform(data)
    centroids_2d = pca.transform(centroids.T)
    time.sleep(1)
    clear_output(wait=True)
    plt.title(f'Iteration {iteration}')
    plt.scatter(x=data_2d[:,0], y=data_2d[:,1], c=labels)
    plt.scatter(x=centroids_2d[:,0], y=centroids_2d[:,1])
    plt.savefig("../../images/kmeans pca.png")
    plt.show()
```

- ⇨ Hàm vẽ biểu đồ, hiển thị và lưu vào images dưới file tên “kmeans pca.png”

```
max_iterations = 100
k = 3

centroids = random_centroids(data, k)
old_centroids = pd.DataFrame()
iteration = 1

while iteration < max_iterations and not centroids.equals(old_centroids):
    old_centroids = centroids

    labels = get_labels(data, centroids)
    centroids = new_centroids(data, labels, k)
    plot_clusters(data, labels, centroids, iteration)
    iteration += 1
```

- ➔ vòng lặp thực thi cho đến khi các cụm không có sự thạy đổi
- ➔ số lần lắp tối đa mà em chọn là 100
- ⇨ Vì trước khi chuyển sang file .py em đã sử dụng jupyter + thư viện Ipython.display.clear_ouput để hiện thị ảnh phân cụm sau mỗi lần lặp cho đến khi cụm không có sự thay đổi, do thư viện Ipython hạn chế trên .py nên em đã quay màn hình về sự thay đổi của cụm tại vị trí NguyenDucAnh_B22DCCN027/images/chang_of_centriods.mp4

- File radarChartPlog.py thực hiện vẽ biểu đồ radar chart so sánh các thuộc tính giữa 2 cầu thủ được nhận từ đối số dòng lệnh, chi tiết triển khai bao gồm

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import argparse


# example: python radarChartPlot.py  --p1 "Aaron Cresswell" --p2 "Aaron Ramsdale" --Attribute ""

# initialize parser
parser = argparse.ArgumentParser()

parser.add_argument("--p1", required=True, type=str, help="Name of player 1")
parser.add_argument("--p2", required=True, type=str, help="Name of player 2")
parser.add_argument("--Attribute", required=True, type=str, help="Attributes that i want to use in radar plot")

# Example: python3 radarChartPlot.py  --p1 "Aaron Cresswell" --p2 "Aaron Ramsdale" --Attribute "Nation,Pos,Squad"
agrs = parser.parse_args()
player1 = agrs.p1
player2 = agrs.p2
attrs = agrs.Attribute.split(",")
```

- ⇨ Dùng thư viện arparse đẻ lấy đối số dòng lệnh

```
df = pd.read_csv("../part1/results.csv")


# transform attributes in attrs into a consistent scale
# new scale
new_max = 10
new_min = 0
new_range = new_max - new_min

# do a linear transformation on each variable to change value to "new_range"
for attr in attrs:
    max_val = df[attr].max()
    min_val = df[attr].min()
    val_range = max_val - min_val

    df[attr] = df[attr].apply(lambda x: (((x - min_val) * new_range) / val_range) + new_min)


df = df[df['Player'].isin([player1, player2])]
df = df[['Player'] + attrs]
```

- ➔ thực hiện chuẩn hóa dữ liệu để so sánh giữa 2 cầu thủ
  - ⇨ Tìm kiếm tên cầu thủ và các thuộc tính cần so sánh

```
import plotly.graph_objects as go

fig = go.Figure()

player1_values = df.loc[0].tolist()
# print(player1_values)

fig.add_trace(go.Scatterpolar(
    r = player1_values[1:],
    theta = attrs,
    fill = 'toself',
    name = player1_values[0]
))


player2_values = df.loc[1].tolist()
fig.add_trace(go.Scatterpolar(
    r = player2_values[1:],
    theta = attrs,
    fill = 'toself',
    name = player2_values[0]
))

fig.update_layout(
    polar = dict(
        radialaxis=dict(
        visible=True,
        range=[0, new_max]
    )),
    showlegend=False
)

fig.show()
```

- ⇨ ở đây em thay vì sử dụng matplotlib e đã sử dụng plotply để vẽ radar do tính mạnh mẽ của nó
- ⇨ kết quả của python3 radarChartPlot.py --p1 "Aaron Cresswell" --p2 "Aaron Ramsdale" --Attribute "Playing_Time_MP,Playing_Time_Starts,Playing_Time_Min"

4. Phần 4
   - File NguyenDucAnh_B22DCCN027/code/part4/fetch_data.py thực hiện quét dữ liệu từ web
   - Thời gian thực hiện trong khoảng thời gian 4 → 8 phút
   - Do trang chứa thông tin là dynamic web nên em sử dụng thư viện selenium để thực hiện quét nội dung
   - Chi tiết file như sau

```python
from selenium import webdriver
import pandas as pd
from bs4 import BeautifulSoup
import time


# instantiate options for Chrome
options = webdriver.ChromeOptions()

# run the browser in headless mode
options.add_argument("--headless=new")

# instantiate Chrome Webdriver
driver = webdriver.Chrome(options=options)


# create data frame to save datas
idx = 0
df = pd.DataFrame(columns=['Player', 'Nation', 'From', 'To', 'Price'])


URL = 'https://www.footballtransfers.com/us/transfers/confirmed/2023-2024/uk-premier-league/'
LIMIT = 18
pages = []
```

- ➔ thêm"—headless=new" để tránh mở trình duyệt khi đang quét dữ liệu
  ➔ các cột được lấy từ web bao gồm player, nation, from, to, price
  ➔ có bảng chứa trong 18 url

```python
pages = []

for num in range(1, LIMIT + 1):
    driver.get(URL + str(num))

    page = driver.page_source
    pages.append(page)
    # print(num)
    time.sleep(1)
```

- ➔ pages chứa thông tin nội dung html mà quét được từ webpage của mỗi url

```python
for i, page in enumerate(pages):
    soup = BeautifulSoup(page, 'html.parser')

    table = soup.find('tbody', id='player-table-body')
    rows = table.find_all('tr')

    for row in rows:
        # tag td
        nation = row.find('figure', class_='small-icon-image').find('img')
        date = row.find('td', class_="td-date d-none d-lg-table-cell").text
        # print(nation.attrs)
        player = row.find('td', class_='td-player').find('span').text
        divs = row.find('td', class_='td-transfer').find_all('div', class_='transfer-club__name')

        price = row.find('td', class_='text-right td-price td-price--no-tag').find('span').text
        try:

            nation = nation.attrs['alt']
            fromm = divs[0].text
            too = divs[-1].text

            df.loc[idx] = [player, nation, fromm, too, (0.0 if price == 'Free' else float(price[1:len(price)-1]))]
            idx += 1
        except (AttributeError, IndexError):
            pass

df.to_csv('data.csv', index=False)
```

- ⇨ Sử dụng thư viện beutifulsoup4 để phân tích cấu trúc file html để lấy nội dung
- ⇨ Lưu file trong NguyenDucAnh_B22DCCN027/code/part4/data.csv