**Báo cáo assignment 1**

**Họ và tên:** Dương Phan Bảo Linh

**Mã sinh viên:** B22DCCN485

**Part I:**

**Requirements**: BeautifulSoup4, lxml lib, pandas và requests module.

Các chỉ số em đã lấy được và có tên biến lần lượt như sau:

+ Player name: player

+ Nation: nationality

+ Team: team

+ Pos: position

+ Age: age

+ Born: birth\_year

+ Playing time

Matches Played: games

Starts: games\_starts

Minutes: minutes

+ Performance:

Goals: goals

non-Penalty Goals: goals\_pens

Ast: assists

Penalty Goals: pens\_made

CrdY: cards\_yellow, CrdR: cards\_red

+ Expected:

xG: xg

npxG: npxg

xAG: xg\_assist

+ Progression:

PrgC: progressive\_carries

PrgP: progressive\_passes

PrgR: progressive\_passes\_received

+ Per 90 mins:

Gls: goals\_per90

Ast: assists\_per90

G+A: goals\_assists\_per90

G-PK: goals\_pens\_per90

G+A-PK: goals\_assists\_pens\_per90

xG: xg\_per90

xAG: xg\_assist\_per90

xG+xAG: xg\_xg\_assist\_per90

npxG: npxg\_per90

npxG+xAG: npxg\_xg\_assist\_per90

+ Goalkeeping:

* Performance:

GA: gk\_goals\_against

GA90: gk\_goals\_against\_per90

SoTA: gk\_shots\_on\_target\_against

Saves: gk\_saves

Save%: gk\_save\_pct

W: gk\_wins

D: gk\_ties

L: gk\_losses

CS: gk\_clean\_sheets

CS%: gk\_clean\_sheets\_pct

* Penalty Kicks

PKatt: gk\_pens\_att

PKA: gk\_pens\_allowed

PKsv: gk\_pens\_saved

PKm: gk\_pens\_missed

Save%: gk\_pens\_save\_pct

+ Shooting:

* Standard

Gls: goals

Sh: shots

SoT: shots\_on\_target

SoT%: shots\_on\_target\_pct

Sh/90: shots\_per90

SoT/90: shots\_on\_target\_per90

G/Sh: goals\_per\_shot

G/SoT: goals\_per\_shot\_on\_target

Dist: average\_shot\_distance

FK: shots\_free\_kicks

PK: pens\_made

PKatt: pens\_att

* Expected

xG: xg

npxG/Sh: npxg\_per\_shot

G-xG: xg\_net

np:G-xG: npxg\_net

+ Passing

Cmp: passes\_completed

Att: passes

Cmp%: passes\_pct

TotDist: passes\_total\_distance

PrgDist: passes\_progressive\_distance

Cmp: passes\_completed\_short

Att: passes\_short

Cmp%: passes\_pct\_short

Cmp: passes\_completed\_medium

Att: passes\_medium

Cmp%: passes\_pct\_medium

Cmp: passes\_completed\_long

Att: passes\_long

Cmp%: passes\_pct\_long

xA: pass\_xa

A-xAG: xg\_assist\_net

KP: assisted\_shots

1/3: passes\_into\_final\_third

PPA: passes\_into\_penalty\_area

CrsPA: crosses\_into\_penalty\_area

+ Pass types

Live: passes\_live

Dead: passes\_dead

FK: passes\_free\_kicks

TB: through\_balls

Sw: passes\_switches

Crs: crosses

TI: throw\_ins

CK: corner\_kicks

In: corner\_kicks\_in

Out: corner\_kicks\_out

Str: corner\_kicks\_straight

Off: passes\_offsides

Blocks: passes\_blocked

+ Goal and Shot Creation:

SCA: sca

SCA90: sca\_per90

PassLive: sca\_passes\_live

PassDead: sca\_passes\_dead

TO: sca\_take\_ons

Sh: sca\_shots

Fld: sca\_fouled

Def: sca\_defense

GCA: gca

GCA90: gca\_per90

PassLive: gca\_passes\_live

PassDead: gca\_passes\_dead

TO: gca\_take\_ons

Sh: gca\_shots

Fld: gca\_fouled

Def: gca\_defense

+ Defensive Actions:

Tkl: tackles

TklW: tackles\_won

Def 3rd: tackles\_def\_3rd

Mid 3rd: tackles\_mid\_3rd

Att 3rd: tackles\_att\_3rd

Tkl: challenge\_tackles

Att: challenges

Tkl%: challenge\_tackles\_pct

Lost: challenges\_lost

Blocks: blocks

Sh: blocked\_shots

Pass: blocked\_passes

Int: interceptions

Tkl+Int: tackles\_interceptions

Clr: clearances

Err: errors

+ Possession

Touches: touches

Def Pen: touches\_def\_pen\_area

Def 3rd: touches\_def\_3rd

Mid 3rd: touches\_mid\_3rd

Att 3rd: touches\_att\_3rd

Att Pen: touches\_att\_pen\_area

Live: touches\_live\_ball

Att: take\_ons

Succ: take\_ons\_won

Succ%: take\_ons\_won\_pct

Tkld: take\_ons\_tackled

Tkld%: take\_ons\_tackled\_pct

Carries: carries

TotDist: carries\_distance

PrgDist: carries\_progressive\_distance

1/3: carries\_into\_final\_third

CPA: carries\_into\_penalty\_area

Mis: miscontrols

Dis: dispossessed

Rec: passes\_received

+ Playing time:

Mn/MP: minutes\_per\_game

Mn/Start: minutes\_per\_start

Compl: games\_complete

Subs: games\_subs

Mn/Sub: minutes\_per\_sub

unSub: unused\_subs

PPM: points\_per\_game

onG: on\_goals\_for

onGA: on\_goals\_against

On-Off: plus\_minus\_wowy

onxG: on\_xg\_for

onxGA: on\_xg\_against

Fls: fouls

Fld: fouled

Off: offsides

OG: own\_goals

Recov: ball\_recoveries

Won: aerials\_won

Lost: aerials\_lost

Won%: aerials\_won\_pct

Kết quả được lưu tại **./output/result.csv**

**Part II:**

+ Top 3 điểm cầu thủ cao nhất và thấp nhất ở mỗi chỉ số:

Kết quả được lưu tại ./output/top\_3\_best\_players.txt

./output/top\_3\_worst\_players.txt

+ Tìm trung vị của mỗi chỉ số:

Kết quả mean, median, std được lưu tại **./output/result2.csv**

Kết quả histogram của trung bình được lưu tại **./images/all (hoặc <team>)**

*Trong đó team là tên đội truyển*

* Xét đoạn code sau:

f = open('./output/best\_team\_in\_each\_category.txt', 'w', *encoding*='utf-8')

f.write('Best team in each category:\n\n'.upper())

freq = {}

for category in categories:

    max\_mean = 0

    best\_team = ''

    for team in sorted(teams):

        team\_players = list(filter(lambda *x*: *x*.stats['team'] == team, players))

        mean, \_, \_ = calculate\_mean\_median\_std(team\_players, category)

        if mean > max\_mean:

            max\_mean = mean

            best\_team = team

    f.write(f'Best team in {category: <35}: {best\_team: <20} : mean = {max\_mean: .4f}\n')

    if best\_team in freq:

        freq[best\_team] += 1

    else:

        freq[best\_team] = 1

f.close()

print(max(freq, *key*=freq.get))

Trong mỗi chỉ số chúng ta tìm tần suất xuất hiện đội tuyển nhiều nhất:

if best\_team in freq:

        freq[best\_team] += 1

    else:

        freq[best\_team] = 1

Và in ra kết quả:

print(max(freq, *key*=freq.get))

Kết quả trả về: **Manchester City**

**Dựa vào đây với mùa giải được xét, đội Manchester City là đội có phong độ tốt nhất!**

**Part 3:**

**Requirements**: BeautifulSoup4, pandas, seaborn, scikit-learn.

* Dữ liệu được đọc từ tệp CSV (result.csv) và các cột cần thiết được chọn lọc.
* Dữ liệu được chuẩn hóa bằng StandardScaler để đảm bảo rằng các đặc trưng có cùng đơn vị đo lường.

# Filter data to include only the selected columns

data\_filtered = data[attributes].dropna()

# Standardize the data

scaler = StandardScaler()

data\_scaled = scaler.fit\_transform(data\_filtered)

* Hàm find\_optimal\_clusters sử dụng phương pháp "Elbow" để xác định số lượng cụm tối ưu cho thuật toán K-Means. Phương pháp này tính toán tổng bình phương sai số (SSE) cho các số lượng cụm khác nhau và vẽ đồ thị để tìm "elbow" của đồ thị.

def find\_optimal\_clusters(*data*, *max\_k*):

    iters = range(1, *max\_k*+1)

    sse = []

    for k in iters:

        kmeans = KMeans(*n\_clusters*=k, *random\_state*=42)

        kmeans.fit(*data*)

        sse.append(kmeans.inertia\_)

    plt.figure(*figsize*=(8, 6))

    plt.plot(iters, sse, *marker*='o')

    plt.xlabel('Cluster Centers')

    plt.ylabel('SSE')

    plt.title('Elbow Method For Optimal k')

    plt.savefig('./images/elbow.png')

    plt.show()

* Sau khi xác định số lượng cụm tối ưu, thuật toán K-Means được áp dụng để phân cụm dữ liệu.
* Kết quả phân cụm được thêm vào dữ liệu gốc.
* Dữ liệu được giảm chiều xuống 2D bằng phương pháp PCA (Phân tích thành phần chính).

pca = PCA(*n\_components*=2)

data\_pca = pca.fit\_transform(data\_scaled)

data['pca1'] = data\_pca[:, 0]

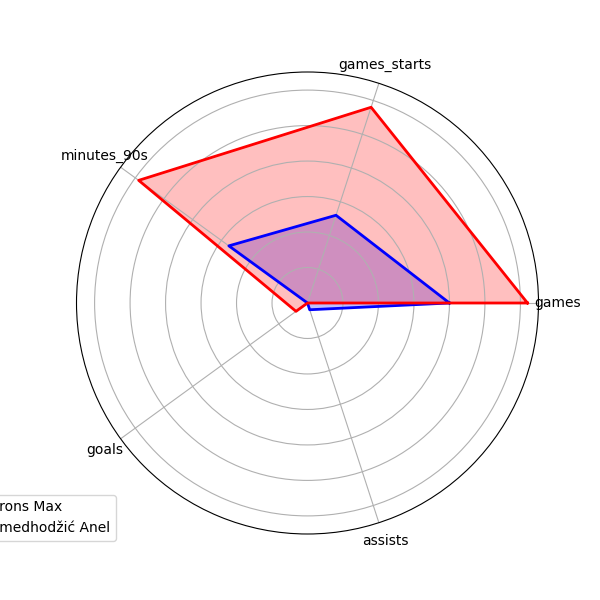
data['pca2'] = data\_pca[:, 1]

* Các cụm được trực quan hóa trên mặt phẳng 2D bằng biểu đồ phân tán.
* Hàm radar\_chart tạo biểu đồ radar để so sánh các đặc trưng của hai cầu thủ cụ thể.
* Biểu đồ này hiển thị các đặc trưng của hai cầu thủ trên cùng một biểu đồ để dễ dàng so sánh.

Ví dụ CLI:

**python -u ".\code\src\part\_three.py" --p1 "Aarons Max" --p2 "Ahmedhodžić Anel" --Attribute games,games\_starts,minutes\_90s,goals,assists**

Ta được kết quả:



**Part 4:**

Bao gồm các bước:

Scrap data từ: **https://www.footballtransfers.com/us/leagues-cups/national/uk/premier-league/2023-2024**

Kết quả được lưu tại: **./code/output/result4.csv**