CLASSIFICATION AND STUDY OF SOCCER TEAM TRENDS

course name: Advanced topics in machine learning

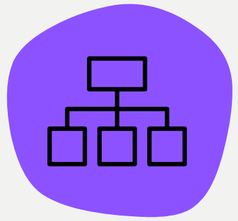
Lecturer's name: Dr. Chen Hajaj

SUBMISSION DATE: 28/03/2024

SERVES:
NOA FADIDA
ADI YAFRAH

Road map

Introduction



Data

Methodology

Analysis

conclusions

Introduction

As in any field of life, also in football machine learning and data mining models are useful for improving performance and learning the opponent.

It is easy to notice the changes that the football world is going through before our eyes in terms of the use of data.

In our project we will focus on two of the leading teams in the Premier League. We will classify and analyze the various parameters of the results in order to search and find what are the factors that influence the sequence of wins/losses of the team.

Data Base

Our dataset provides information about the team's games over 4 different seasons which are divided into 38 games per season.

Arsenal

```
[129]: as_sf_21_22
```

[129]:

	Date	Time	Round	Day	Venue	Result	GF	GA	Opponent	xG	xGA	Poss	Attendance	Captain	Formation	Referee	Match Report	Notes
0	2021-08-13	20:00	Matchweek 1	Fri	Away	L	0	2	Brentford	1.3	1.2	64	16479	Granit Xhaka	4-2-3-1	Michael Oliver	Match Report	NaN
1	2021-08-22	16:30	Matchweek 2	Sun	Home	L	0	2	Chelsea	0.3	3.1	35	58729	Granit Xhaka	4-2-3-1	Paul Tierney	Match Report	NaN
2	2021-08-28	12:30	Matchweek 3	Sat	Away	L	0	5	Manchester City	0.2	4.4	20	52276	Pierre-Emerick Aubameyang	5-4-1	Martin Atkinson	Match Report	NaN
3	2021-09-11	15:00	Matchweek 4	Sat	Home	W	1	0	Norwich City	2.7	0.7	51	58000	Pierre-Emerick Aubameyang	4-2-3-1	Michael Oliver	Match Report	NaN
4	2021-09-18	15:00	Matchweek 5	Sat	Away	W	1	0	Burnley	1.1	1.0	54	20000	Pierre-Emerick Aubameyang	4-1-4-1	Anthony Taylor	Match Report	NaN
5	2021-09-26	16:30	Matchweek 6	Sun	Home	W	3	1	Tottenham	1.1	1.0	46	59919	Pierre-Emerick Aubameyang	4-2-3-1	Craig Pawson	Match Report	NaN
6	2021-10-02	17:30	Matchweek 7	Sat	Away	D	0	0	Brighton	0.4	1.1	42	31266	Pierre-Emerick Aubameyang	4-2-3-1	Jonathan Moss	Match Report	NaN
7	2021-10-18	20:00	Matchweek 8	Mon	Home	D	2	2	Crystal Palace	1.7	0.7	54	59475	Pierre-Emerick Aubameyang	4-1-4-1	Mike Dean	Match Report	NaN
8	2021-10-22	20:00	Matchweek 9	Fri	Home	W	3	1	Aston Villa	2.7	1.4	53	59496	Pierre-Emerick Aubameyang	4-2-3-1	Craig Pawson	Match Report	NaN
9	2021-10-30	12:30	Matchweek 10	Sat	Away	W	2	0	Leicester City	0.7	1.3	36	32209	Pierre-Emerick Aubameyang	4-4-1-1	Michael Oliver	Match Report	NaN
10	2021-11-07	14:00	Matchweek 11	Sun	Home	W	1	0	Watford	1.6	0.5	60	59833	Pierre-Emerick Aubameyang	4-4-1-1	Kevin Friend	Match Report	NaN
11	2021-11-20	17:30	Matchweek 12	Sat	Away	L	0	4	Liverpool	0.4	4.3	38	53092	Pierre-Emerick Aubameyang	4-4-1-1	Michael Oliver	Match Report	NaN
12	2021-11-27	12:30	Matchweek 13	Sat	Home	W	2	0	Newcastle Utd	2.5	0.3	66	59886	Pierre-Emerick Aubameyang	4-4-1-1	Stuart Attwell	Match Report	NaN
13	2021-12-02	20:15	Matchweek 14	Thu	Away	L	2	3	Manchester Utd	1.4	1.9	56	73123	Pierre-Emerick Aubameyang	4-4-1-1	Martin Atkinson	Match Report	NaN
14	2021-12-06	20:00	Matchweek 15	Mon	Away	L	1	2	Everton	1.0	0.9	63	38906	Alexandre Lacazette	4-2-3-1	Mike Dean	Match Report	NaN
15	2021-12-11	15:00	Matchweek 16	Sat	Home	W	3	0	Southampton	1.8	0.6	61	59653	Alexandre Lacazette	4-4-1-1	Jarred Gillett	Match Report	NaN
16	2021-12-15	20:00	Matchweek 17	Wed	Home	W	2	0	West Ham	2.8	0.4	55	59777	Alexandre Lacazette	4-4-1-1	Anthony Taylor	Match Report	NaN
17	2021-12-18	17:30	Matchweek 18	Sat	Away	W	4	1	Leeds United	3.5	1.6	49	36166	Alexandre Lacazette	4-2-3-1	Andre Marriner	Match Report	NaN

The initial data before cleaning contains:

- 2 different tables of data for each season and each group
- A total of 8 tables per group
- 38 rows per table
- 39 columns
- In addition, there are position tables of the opposing teams for each season that we have also attached to the tables

Data Base

We have numeric, date and binary values in the array. But most of our variables are categorical.

The data set is built from the details of the game, which contain columns such as: date, time, home, result, type of vehicle, manufacturer, referee and more..

- 'Time'
- 'Round'
- 'Day'
- 'Venue'
- 'Result'
- 'GF'
- 'GA'
- 'xG'
- 'xGA'
- 'Poss'
- 'Captain'
- 'Formation'
- 'Referee'
- 'location'
- 'CrdY'
- 'CrdR'
- '2CrdY'
- 'Fls'
- 'Fld'
- 'Off'
- Crs'
- 'Int',
- TklW'
- 'PKwon'
- 'PKcon',
- 'OG'
- 'Recov'
- 'Won'
- 'Lost'
- 'month'
- 'year'

Methodology

1. Cleaning and consolidating data
2. Treatment of categorical variables
3. Normalization
4. Identification of relevant features for analysis using PCA
5. Data classification using different algorithms and comparing them:

- Algorithm based on k-mean distance
- Hierarchical Clustering - Hierarchical algorithm

Cleaning and consolidating data

After importing the data, we arranged the location table that we will merge later: we removed irrelevant columns, changed column names and specific group names that did not correspond exactly to the main data tables to which we will merge this table

הורדת עמודות לא רלוונטיות והשארת המיקום בלבד

[37]:
```
loc_19_20=loc_19_20.drop(columns=['Pl','W','D','L','F','A','GD','Pts','Last 6'])
loc_20_21=loc_20_21.drop(columns=['Pl','W','D','L','F','A','GD','Pts','Last 6'])
loc_21_22=loc_21_22.drop(columns=['Pl','W','D','L','F','A','GD','Pts','Last 6'])
loc_22_23=loc_22_23.drop(columns=['Pl','W','D','L','F','A','GD','Pts','Last 6'])
```

[38]:
```
loc_19_20.rename(columns={'#': 'location'}, inplace=True)
loc_20_21.rename(columns={'#': 'location'}, inplace=True)
loc_21_22.rename(columns={'#': 'location'}, inplace=True)
loc_22_23.rename(columns={'#': 'location'}, inplace=True)
loc_19_20.rename(columns={'Team': 'Opponent'}, inplace=True)
loc_20_21.rename(columns={'Team': 'Opponent'}, inplace=True)
loc_21_22.rename(columns={'Team': 'Opponent'}, inplace=True)
loc_22_23.rename(columns={'Team': 'Opponent'}, inplace=True)
```

[39]:
```
loc_19_20.loc[5,'Opponent'] = 'Tottenham'
loc_19_20.loc[14,'Opponent'] = 'Brighton'
loc_19_20.loc[12,'Opponent'] = 'Newcastle Utd'
loc_19_20.loc[2,'Opponent'] = 'Manchester Utd'
loc_19_20.loc[15,'Opponent'] = 'West Ham'
loc_19_20.loc[6,'Opponent'] = 'Wolves'
loc_19_20.loc[8,'Opponent'] = 'Sheffield Utd'
```

Cleaning and consolidating data

After that we created a function to which we sent each time 2 different tables but of the same team and the same season and combined into one table.

We removed duplicate or irrelevant columns, split the date into separate columns, removed redundant characters and the last row of each table which was a summary specific to each table and was not relevant.

איחוד טבלאות לפי קבוצה כולל הוספת מיקום

```
[43]: def data_prepare (team_general, team_stat, location):
          # הוספת מיקום בטבלה של קבוצה יריבה
          team_main = pd.merge(team_general, location, on='Opponent', how='left')

          # סידור עמודות משנה לשמות של עמודה רעלה בטבלת STAT
          new_columns = [col[1] for col in team_stat.columns]
          team_stat.columns = new_columns
          #print(team_stat.columns)

          # איחוד טבלאות לטבלה אחת
          team_main = pd.merge(team_main, team_stat, on='Round', how='outer')

          # הסרת העמודות הכפולות
          team_main = team_main.drop(columns=['Date_y', 'Time_y', 'Day_y', 'Venue_y',

          # הורדת תווים מיותרים שנוצרו מכפילות עמודות
          team_main.columns = [col.replace('_x', '') for col in team_main.columns]
          team_main['Round'] = team_main['Round'].str.replace('Matchweek ', '')

          # הוספת עמודה רק עם החודש של המחשק מתוך התאריך
          #team_main['Month'] = team_main['Date'].astype(int).astype(str)
          # Assuming 'df' is your DataFrame
          #df['Month'] = df['Date'].str.split('-', expand=True)[1]
          team_main['Date'] = pd.to_datetime(team_main['Date'])
          team_main['month'] = team_main['Date'].dt.month
          team_main['year'] = team_main['Date'].dt.year
```

```
]: groups_arsenal
```

]:

	Time	Round	Day	Venue	Result	GF	GA	xG	xGA	Poss	...	Int	TklW	PKwon	P
0	14:00	1	Sun	Away	W	1.0	0.0	1.1	0.4	62.0	...	6	6	0	
1	12:30	2	Sat	Home	W	2.0	1.0	0.8	1.5	67.0	...	9	7	0	
2	17:30	3	Sat	Away	L	1.0	3.0	1.0	2.5	48.0	...	7	7	0	
3	16:30	4	Sun	Home	D	2.0	2.0	2.4	2.0	55.0	...	11	12	0	
4	16:30	5	Sun	Away	D	2.0	2.0	0.8	2.7	48.0	...	11	10	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
147	20:00	34	Tue	Home	W	3.0	1.0	1.7	0.8	55.0	...	7	4	0	
148	16:30	35	Sun	Away	W	2.0	0.0	1.3	1.3	45.0	...	8	18	0	
149	16:30	36	Sun	Home	L	0.0	3.0	0.9	1.7	41.0	...	7	6	0	
150	17:30	37	Sat	Away	L	0.0	1.0	0.6	0.6	81.0	...	6	3	0	
151	16:30	38	Sun	Home	W	5.0	0.0	2.8	0.5	51.0	...	2	14	0	

152 rows × 31 columns

categorical variables

ENCODING for all the categorical variables we have

¶ טיפול בערכים מספריים והפיכתם לבינאריים

```
9]: def replace_values(df, column_name):
        df[column_name] = df[column_name].replace({'Home': 1, 'Away': 0})
        return df
```

	Round	Venue	GF	GA	xG	xGA	Poss	location	CrdY	CrdR	...	Formation_3-4-3	Formation_4-1-2-1-2◆	Formation_4-1-4-1	Formation_4-2-3-1	Formation_4-3-1-2	Formation_4-3-2-1	Formation_4-3-3	Formation_4-4-1-1	Formation_4-4-2	Formation_5-4-1
0	1	0	1.0	0.0	1.1	0.4	62.0	13.0	3	0	...	0	0	0	1	0	0	0	0	0	0
1	2	1	2.0	1.0	0.8	1.5	67.0	10.0	2	0	...	0	0	0	1	0	0	0	0	0	0
2	3	0	1.0	3.0	1.0	2.5	48.0	1.0	1	0	...	0	0	0	0	1	0	0	0	0	0
3	4	1	2.0	2.0	2.4	2.0	55.0	6.0	3	0	...	0	0	0	0	0	0	1	0	0	0
4	5	0	2.0	2.0	0.8	2.7	48.0	19.0	3	0	...	0	1	0	0	0	0	0	0	0	0

5 rows × 89 columns

1 0 1 0
0 1 0 1
0 1 0 1

Normalization

Each column was on a different scale, we normalized the data set

	Round	Venue	GF	GA	xG	xGA	Poss	location	CrdY	CrdR	...	Formation_4-1-2-1-2◆	Formation_4-1-4-1	Formation_4-2-3-1	Formation_4-3-1-2	Formation_4-3-2-1	Formation_4-3-3	Formation_4-4-1-1	Formation_4-4-2	Formation_5-4-1	Cluster
0	0.000000	0.0	0.2	0.0	0.243902	0.090909	0.688525	0.631579	0.428571	0.0	...	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0
1	0.027027	1.0	0.4	0.2	0.170732	0.340909	0.770492	0.473684	0.285714	0.0	...	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0
2	0.054054	0.0	0.2	0.6	0.219512	0.568182	0.459016	0.000000	0.142857	0.0	...	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	2
3	0.081081	1.0	0.4	0.4	0.560976	0.454545	0.573770	0.263158	0.428571	0.0	...	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	2
4	0.108108	0.0	0.4	0.4	0.170732	0.613636	0.459016	0.947368	0.428571	0.0	...	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2

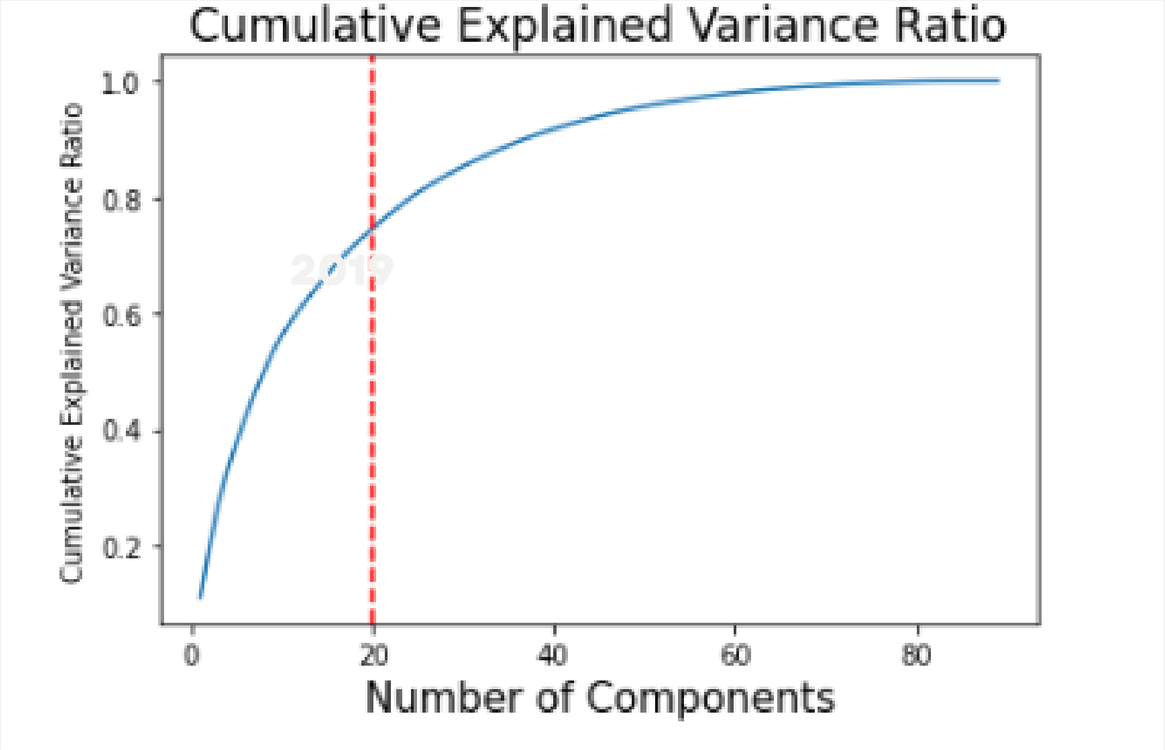
5 rows × 90 columns

PCA

Identifying relevant features for analysis with the help of PCA

The data set we selected contains a large number of columns. We will select the most informative columns. We will use PCA for this.

To decide how many dimensions we want to reduce the data to, we calculate the cumulative explained variance ratio



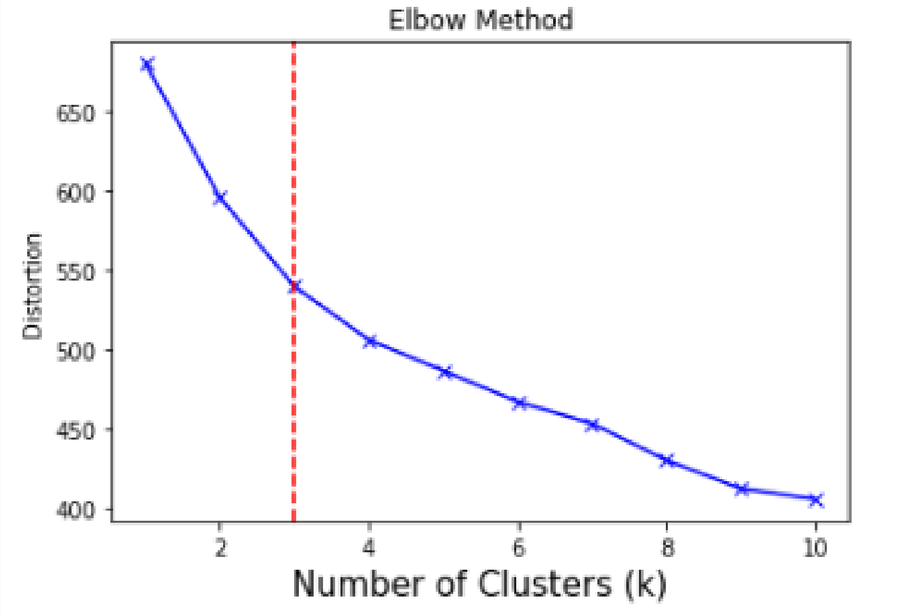
According to the graph, you can see that the "elbow point" is when we choose 20 dimensions

We chose to reduce the data from 74 columns to 20 columns

K-MEAN

We performed this algorithm on the Arsenal team

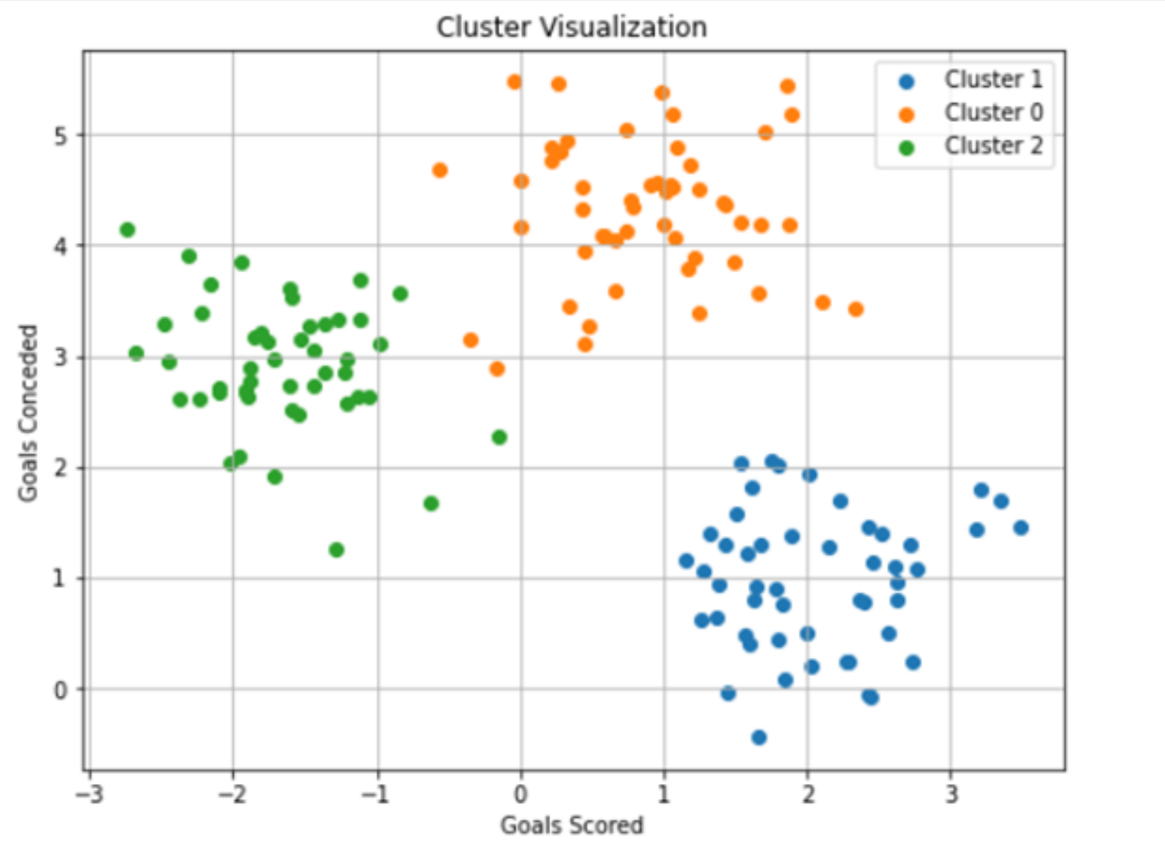
We created a graph to decide how many groups to classify the data into.
We chose to classify into 3 groups. It can be seen that at this stage the slope is significantly smaller




```
[66]: print(normal_groups_arsenal['Cluster'].value_counts())
      2    59
      0    54
      1    37
      Name: Cluster, dtype: int64
```

We did the classification and you can see the division of the data into groups and how many rows there are in each of the groups

graphs and conclusions k-mean



Cluster 0: This cluster represents teams or players that have scored fewer goals (even negative) but have also conceded fewer goals.

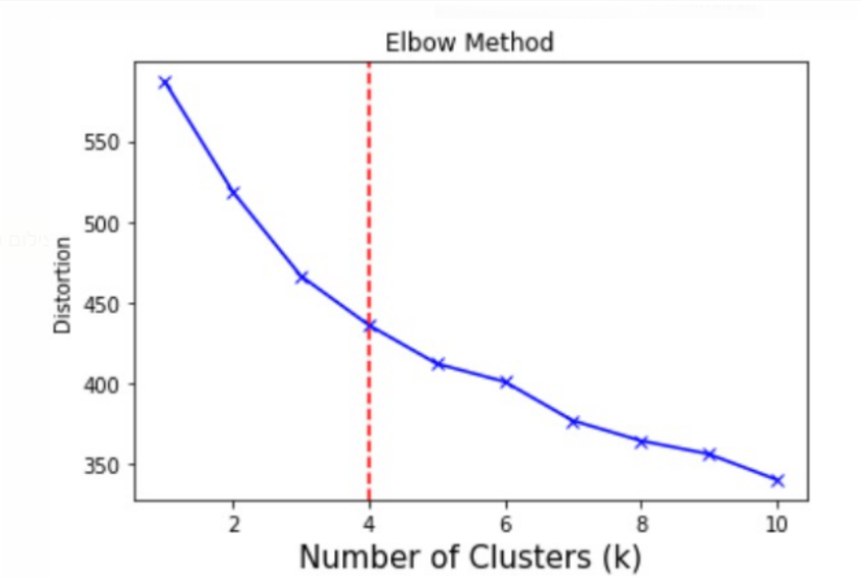
Cluster 1 -This cluster represents teams or players that have both scored and conceded a high number of goals.

Cluster 2 -This cluster represents teams or players with positive goal-scoring but have also conceded more than one goal

Hierarchical Clustering

We performed this algorithm on the Liverpool team

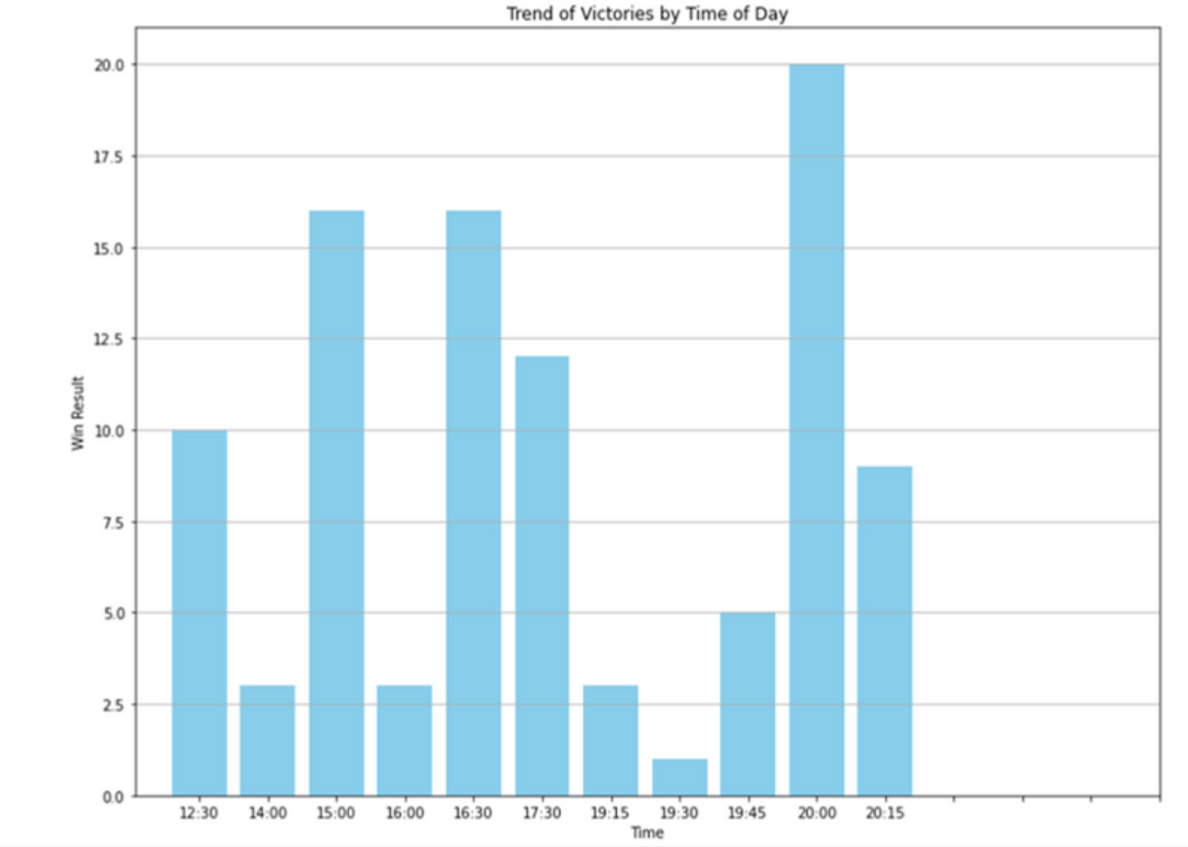
We chose to classify into 4 groups. It can be seen that at this stage the slope is significantly smaller




```
]: from sklearn.cluster import AgglomerativeClustering
   # Perform divisive hierarchical clustering
   clustering = AgglomerativeClustering(n_clusters=None, affinity='euclidean', linka
   clusters = clustering.fit_predict(normal_groups_liverpool)
   print(clusters)
   (פלטא
```

```
[0 1 1 1 1 2 1 1 2 2 1 2 1 1 1 1 1 0 2 0 1 2 0 0 1 1 0 3 1 2 0 0 2 0 1 0 0
 2 1 2 0 2 3 1 1 2 2 3 2 2 0 1 2 0 0 2 0 0 2 0 2 3 3 2 0 2 0 1 1 3 3 0 0 2
 0 2 1 1 3 2 1 3 2 1 2 3 2 1 1 0 1 1 0 2 0 2 2 2 0 2 1 0 1 1 0 1 2 0 2 1 3
 0 0 2 3 3 0 1 0 3 3 2 2 0 3 3 2 1 0 0 3 3 3 0 1 3 2 3 3 3 2 0 1 0 2 0 1 0
 3 2]
```

graphs and conclusions Hierarchical Clustering



The highest number of wins games in Liverpool occurs around 20:00, with another significant peak at 14:00 and 15:00, The lowest number of victories is recorded at 19:30

Conclusion and Future Work

Projects of this type are significant for soccer analysis and the success of sports teams. By playing soccer teams, analysts and coaches can gain valuable insights into different playing styles, team performance levels, and strategic patterns. This information can be used to optimize team strategies, identify areas for improvement and make data-driven decisions to improve overall team performance.

THANK YOU