In [100…

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib.image as mpimg
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

Features description

---

date : the date of the game

time : the time of the game

comp : the competition of the game

round : the round of the game

day : the day of the week of the game

venue : the venue of the game

result : the result of the game

gf : the goals for the home team

ga : the goals for the away team

opponent: the opponent of the home team

xg : the expected goals for the home team

xga : the expected goals for the away team

poss : the possession of the home team

captain : the captain of the home team

formation : the formation of the home team

referee : the referee of the game

sh : the shots of the home team

sot : the shots on target of the home team

dist : the average distance of the shots of the home team

fk : the free kicks of the home team

pk : the penalty kicks of the home team

pka : the penalty kicks attempted of the home team

season : the season year of the match

team: the home team

Preprocessing and Data Cleaning

In [100…

```
df = pd.read_csv('D:\\projects\\datasets\\matches.csv',encoding='utf-8')
pd.set_option('display.max_columns',len(df.columns))
df_c = df.copy()
df_c
```

Out[1007]:

	Unnamed: 0	date	time	comp	round	day	venue	result	gf	ga	opponent	xg	xga	poss	attendance	captain	for
0	0	2020-09-21	20:15 (21:15)	Premier League	Matchweek 2	Mon	Away	W	3	1	Wolves	1.9	0.6	65	NaN	Fernandinho	
1	2	2020-09-27	16:30 (17:30)	Premier League	Matchweek 3	Sun	Home	L	2	5	Leicester City	0.9	2.9	72	NaN	Fernandinho	
2	4	2020-10-03	17:30 (18:30)	Premier League	Matchweek 4	Sat	Away	D	1	1	Leeds United	1.2	2.4	49	NaN	Kevin De Bruyne	
3	5	2020-10-17	17:30 (18:30)	Premier League	Matchweek 5	Sat	Home	W	1	0	Arsenal	1.3	0.9	58	NaN	Raheem Sterling	
4	7	2020-10-24	12:30 (13:30)	Premier League	Matchweek 6	Sat	Away	D	1	1	West Ham	1.0	0.3	69	NaN	Raheem Sterling	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
4783	87	2020-07-07	18:00 (19:00)	Premier League	Matchweek 34	Tue	Away	L	1	2	Watford	1.2	1.2	56	NaN	Alexander Tettey	
4784	88	2020-07-11	12:30 (13:30)	Premier League	Matchweek 35	Sat	Home	L	0	4	West Ham	0.6	3.5	53	NaN	Alexander Tettey	
4785	89	2020-07-14	20:15 (21:15)	Premier League	Matchweek 36	Tue	Away	L	0	1	Chelsea	0.1	2.5	33	NaN	Alexander Tettey	
4786	90	2020-07-18	17:30 (18:30)	Premier League	Matchweek 37	Sat	Home	L	0	2	Burnley	0.3	1.8	42	NaN	Alexander Tettey	
4787	91	2020-07-26	16:00 (17:00)	Premier League	Matchweek 38	Sun	Away	L	0	5	Manchester City	1.0	3.2	27	NaN	Christoph Zimmermann	

4788 rows × 28 columns

In [100...
```
df_c.rename(columns=lambda x: x.strip().lower().replace(' ', '_'), inplace=True)
```

In [100...
```
df_c.shape
```

Out[1009]: (4788, 28)

In [101...
```
df_c.describe()
```

Out[1010]:

	unnamed:_0	gf	ga	xg	xga	poss	attendance	notes	sh	sot	
count	4788.000000	4788.000000	4788.000000	4788.000000	4788.000000	4788.000000	3155.000000	0.0	4788.000000	4788.000000	4786.0
mean	63.044069	1.447995	1.405388	1.396512	1.364745	50.432957	38397.586688	NaN	12.619256	4.261278	17.3
std	42.865191	1.312635	1.286927	0.828847	0.814947	12.810958	17595.849137	NaN	5.548444	2.459963	3.0
min	0.000000	0.000000	0.000000	0.000000	0.000000	18.000000	2000.000000	NaN	0.000000	0.000000	5.3
25%	28.000000	0.000000	0.000000	0.800000	0.700000	41.000000	25513.500000	NaN	9.000000	2.000000	15.4
50%	62.000000	1.000000	1.000000	1.300000	1.200000	51.000000	36347.000000	NaN	12.000000	4.000000	17.2
75%	87.000000	2.000000	2.000000	1.900000	1.800000	60.000000	53235.500000	NaN	16.000000	6.000000	19.1
max	182.000000	9.000000	9.000000	7.000000	7.000000	82.000000	75546.000000	NaN	36.000000	15.000000	39.9

In [101…

```
df_c.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4788 entries, 0 to 4787
Data columns (total 28 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   unnamed:_0    4788 non-null   int64  
 1   date          4788 non-null   object 
 2   time          4788 non-null   object 
 3   comp          4788 non-null   object 
 4   round         4788 non-null   object 
 5   day           4788 non-null   object 
 6   venue         4788 non-null   object 
 7   result        4788 non-null   object 
 8   gf            4788 non-null   int64  
 9   ga            4788 non-null   int64  
 10  opponent      4788 non-null   object 
 11  xg            4788 non-null   float64
 12  xga           4788 non-null   float64
 13  poss          4788 non-null   int64  
 14  attendance    3155 non-null   float64
 15  captain       4788 non-null   object 
 16  formation     4788 non-null   object 
 17  referee       4788 non-null   object 
 18  match_report  4788 non-null   object 
 19  notes         0 non-null      float64
 20  sh            4788 non-null   int64  
 21  sot           4788 non-null   int64  
 22  dist          4786 non-null   float64
 23  fk            4788 non-null   int64  
 24  pk            4788 non-null   int64  
 25  pkatt         4788 non-null   int64  
 26  season        4788 non-null   int64  
 27  team          4788 non-null   object 
dtypes: float64(5), int64(10), object(13)
memory usage: 1.0+ MB
```

In [101…

```
df_c.isna().sum().to_frame()
```

Out[1012]:

	0
unnamed:_0	0
date	0
time	0
comp	0
round	0
day	0
venue	0
result	0
gf	0
ga	0
opponent	0
xg	0
xga	0
poss	0
attendance	1633
captain	0
formation	0
referee	0
match_report	0
notes	4788
sh	0
sot	0
dist	2
fk	0
pk	0

	0
pkatt	0
season	0
team	0

In [101…

```
df_c = df_c.drop(['notes', 'comp', 'time','match_report',df_c.columns[0]], axis=1) ## df_c.drop(columns=['notes'],inplace=True)
df_c
```

Out[1013]:

	date	round	day	venue	result	gf	ga	opponent	xg	xga	poss	attendance	captain	formation	referee	sh	sot
0	2020-09-21	Matchweek 2	Mon	Away	W	3	1	Wolves	1.9	0.6	65	NaN	Fernandinho	4-2-3-1	Andre Marriner	13	8
1	2020-09-27	Matchweek 3	Sun	Home	L	2	5	Leicester City	0.9	2.9	72	NaN	Fernandinho	4-2-3-1	Michael Oliver	16	5
2	2020-10-03	Matchweek 4	Sat	Away	D	1	1	Leeds United	1.2	2.4	49	NaN	Kevin De Bruyne	4-3-3	Mike Dean	23	1
3	2020-10-17	Matchweek 5	Sat	Home	W	1	0	Arsenal	1.3	0.9	58	NaN	Raheem Sterling	3-1-4-2	Chris Kavanagh	13	5
4	2020-10-24	Matchweek 6	Sat	Away	D	1	1	West Ham	1.0	0.3	69	NaN	Raheem Sterling	4-3-3	Anthony Taylor	14	7
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4783	2020-07-07	Matchweek 34	Tue	Away	L	1	2	Watford	1.2	1.2	56	NaN	Alexander Tettey	4-2-3-1	Anthony Taylor	12	3
4784	2020-07-11	Matchweek 35	Sat	Home	L	0	4	West Ham	0.6	3.5	53	NaN	Alexander Tettey	4-2-3-1	Kevin Friend	11	2
4785	2020-07-14	Matchweek 36	Tue	Away	L	0	1	Chelsea	0.1	2.5	33	NaN	Alexander Tettey	4-1-4-1	Jonathan Moss	2	0
4786	2020-07-18	Matchweek 37	Sat	Home	L	0	2	Burnley	0.3	1.8	42	NaN	Alexander Tettey	4-2-3-1	Kevin Friend	6	2
4787	2020-07-26	Matchweek 38	Sun	Away	L	0	5	Manchester City	1.0	3.2	27	NaN	Christoph Zimmermann	4-2-3-1	Craig Pawson	5	4

4788 rows × 23 columns

In [101…

```
df_c.isna().sum()
```

Out[1014]:

```
date          0
round         0
day           0
venue         0
result        0
gf            0
ga            0
opponent      0
xg            0
xga           0
poss          0
attendance    1633
captain       0
formation     0
referee       0
sh            0
sot           0
dist          2
fk            0
pk            0
pkatt         0
season        0
team          0
dtype: int64
```

In [101…

```
print(df_c.duplicated().sum(),"\n" )
print(df_c.nunique().to_frame())
```

0

```
              0
date        610
round        38
day           7
venue         2
result        3
gf           10
ga           10
opponent     26
xg           51
xga          51
poss         65
attendance  1344
captain      178
formation     22
referee       35
sh           35
sot          16
dist        193
fk            5
pk            4
pkatt         4
season        5
team         26
```

there are

26 teams

5 seasons

35 referee

38 rounds

In [101…

```
print(df_c['attendance'].mean())
print(df_c['attendance'].median())
print(df_c['attendance'].mode()[0])

df_c['attendance'].fillna(df_c['attendance'].mode()[0],inplace=True)
df_c.sample(5)
```

38397.58668779715
36347.0
2000.0

Out[1016]:

	date	round	day	venue	result	gf	ga	opponent	xg	xga	poss	attendance	captain	formation	referee	sh	sot	d
2510	2021-08-28	Matchweek 3	Sat	Home	W	5	0	Arsenal	4.4	0.2	80	52276.0	İlkay Gündoğan	4-3-3	Martin Atkinson	25	10	14
839	2019-08-31	Matchweek 4	Sat	Away	D	1	1	Southampton	1.5	0.9	59	30499.0	Ashley Young	4-2-3-1	Mike Dean	21	8	1
2848	2022-05-15	Matchweek 37	Sun	Away	D	1	1	Leeds United	2.2	1.6	50	36638.0	Lewis Dunk	3-4-3	Mike Dean	15	6	14
4302	2019-10-21	Matchweek 9	Mon	Away	L	0	1	Sheffield Utd	1.0	0.9	67	30775.0	Granit Xhaka	4-2-3-1	Mike Dean	9	3	18
1279	2024-02-24	Matchweek 26	Sat	Home	L	1	2	Fulham	1.7	1.2	57	73487.0	Bruno Fernandes	4-2-3-1	Michael Oliver	21	8	1

In [101…

```
df_c.dropna(inplace=True)
df_c.isna().sum().sum()
```

Out[1017]: 0

In [101…

```
conditions = [
    df_c['venue'] == "Home" ,
    df_c['venue']== "Away"
]
values = [1,2]
df_c['venue'] = np.select(conditions,values,0)
df_c.sample(5)
```

Out[1018]:

	date	round	day	venue	result	gf	ga	opponent	xg	xga	poss	attendance	captain	formation	referee	sh	sot	dist
4427	2019-12-28	Matchweek 20	Sat	1	D	1	1	Crystal Palace	1.4	0.3	59	31108.0	Pierre Højbjerg	4-4-2	Andy Madley	14	5	15.8
1266	2023-11-26	Matchweek 13	Sun	2	W	3	0	Everton	2.2	2.4	49	39257.0	Bruno Fernandes	4-2-3-1	John Brooks	8	3	17.4
2714	2021-12-27	Matchweek 19	Mon	2	D	1	1	Newcastle Utd	1.6	1.3	68	52178.0	Harry Maguire	4-2-2-2	Craig Pawson	13	3	17.9
317	2020-12-20	Matchweek 14	Sun	2	L	2	6	Manchester Utd	1.8	3.6	59	2000.0	Liam Cooper	4-1-4-1	Anthony Taylor	17	4	13.9
1469	2024-02-24	Matchweek 26	Sat	2	W	2	1	Manchester Utd	1.2	1.7	43	73487.0	Harrison Reed	4-2-3-1	Michael Oliver	17	5	17.1

In [101…

```
# Define conditions for each possible result
conditions = [
    df_c['result'] == 'W',
    df_c['result'] == 'D',
    df_c['result'] == 'L'
]

# Define corresponding values for each condition
values = [1, 0, -1]

df_c['result'] = np.select(conditions, values,0)
df_c.sample(5)
```

Out[1019]:

	date	round	day	venue	result	gf	ga	opponent	xg	xga	poss	attendance	captain	formation	referee	sh	sot	d
4787	2020-07-26	Matchweek 38	Sun	2	-1	0	5	Manchester City	1.0	3.2	27	2000.0	Christoph Zimmermann	4-2-3-1	Craig Pawson	5	4	1
1532	2023-11-26	Matchweek 13	Sun	1	-1	0	3	Manchester Utd	2.4	2.2	51	39257.0	James Tarkowski	4-4-1-1	John Brooks	24	6	1
1573	2023-12-09	Matchweek 16	Sat	2	-1	0	1	Sheffield Utd	0.6	0.8	59	28509.0	Christian Nørgaard	4-3-3	Stuart Attwell	10	4	1
2206	2022-08-20	Matchweek 3	Sat	2	-1	0	1	Tottenham	0.7	1.7	50	61298.0	Rúben Neves	3-5-2	Simon Hooper	20	3	2
1453	2023-10-29	Matchweek 10	Sun	2	0	1	1	Brighton	0.7	1.4	29	31550.0	Tim Ream	4-2-3-1	Michael Salisbury	10	5	1

In [102…

```
df_c['day'] = df_c['day'].str.strip().str.lower()
df_c['day'].sample(5)
```

Out[1020]:

```
35     fri
2620   thu
4691   sat
1375   sun
335    mon
Name: day, dtype: object
```

In [102…

```
conditions = [
   df_c['day']=='sat' ,
   df_c['day']=='sun',
   df_c['day']=='mon',
   df_c['day']=='tue',
   df_c['day']=='wed',
   df_c['day']=='thu' ,
   df_c['day']=='fri'
]

values = [1,2,3,4,5,6,7]

df_c['day'] = np.select(conditions, values,0)
df_c.sample(5)
```

Out[1021]:

	date	round	day	venue	result	gf	ga	opponent	xg	xga	poss	attendance	captain	formation	referee	sh	sot	dist
1102	2023-08-12	Matchweek 1	1	2	-1	1	5	Newcastle Utd	1.8	3.3	48	52207.0	John McGinn	4-2-3-1	Andy Madley	16	6	14.5
2972	2021-10-24	Matchweek 9	2	1	-1	1	2	Leicester City	1.0	1.5	54	16814.0	Pontus Jansson	3-5-2	Simon Hooper	15	5	16.3
1897	2023-05-18	Matchweek 25	6	1	1	4	1	Brighton	3.6	0.5	35	52122.0	Kieran Trippier	4-3-3	Robert Jones	22	8	13.3
4460	2019-12-04	Matchweek 15	5	2	-1	2	5	Liverpool	1.5	2.0	42	53094.0	Gylfi Sigurðsson	5-4-1	Mike Dean	12	4	15.5
4372	2019-08-25	Matchweek 3	2	2	0	1	1	Wolves	1.6	1.5	35	30522.0	Ben Mee	4-4-2	Craig Pawson	13	4	16.3

In [102...

```
x = df_c['result'].value_counts()
x
```

Out[1022]:

```
result
 1    1895
-1    1820
 0    1071
Name: count, dtype: int64
```

In [102...

```
df_c['date'] = pd.to_datetime(df_c['date'], format='%Y-%m-%d')
df_c['date'].sample(5)
```

Out[1023]:

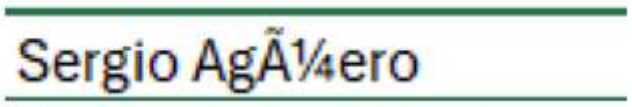
```
2381   2023-03-05
4547   2020-02-08
4490   2019-09-29
4189   2019-10-25
2938   2021-11-27
Name: date, dtype: datetime64[ns]
```

In [102...

```
print(df_c.loc[29:35,'captain'],"\n")  ## transform the latin words into eng

# Load the image
img = mpimg.imread('D:\\projects\\sergio.PNG')  # Use the correct path to your image

# Display the image
plt.figure(figsize=(4, 2))
plt.imshow(img)
```

```
plt.axis('off')  # Hide the axis
plt.show()
```

```
29     Sergio Agüero
30       Fernandinho
31       Fernandinho
32     İlkay Gündoğan
33       Fernandinho
34    Raheem Sterling
35    Raheem Sterling
Name: captain, dtype: object
```



In [102…
```
df_c.loc[0:5,]
```

Out[1025]:

	date	round	day	venue	result	gf	ga	opponent	xg	xga	poss	attendance	captain	formation	referee	sh	sot	dist
0	2020-09-21	Matchweek 2	3	2	1	3	1	Wolves	1.9	0.6	65	2000.0	Fernandinho	4-2-3-1	Andre Marriner	13	8	21.1
1	2020-09-27	Matchweek 3	2	1	-1	2	5	Leicester City	0.9	2.9	72	2000.0	Fernandinho	4-2-3-1	Michael Oliver	16	5	19.8
2	2020-10-03	Matchweek 4	1	2	0	1	1	Leeds United	1.2	2.4	49	2000.0	Kevin De Bruyne	4-3-3	Mike Dean	23	1	18.2
3	2020-10-17	Matchweek 5	1	1	1	1	0	Arsenal	1.3	0.9	58	2000.0	Raheem Sterling	3-1-4-2	Chris Kavanagh	13	5	17.7
4	2020-10-24	Matchweek 6	1	2	0	1	1	West Ham	1.0	0.3	69	2000.0	Raheem Sterling	4-3-3	Anthony Taylor	14	7	20.9
5	2020-10-31	Matchweek 7	1	2	1	1	0	Sheffield Utd	1.6	0.5	65	2000.0	Kevin De Bruyne	4-3-3	Michael Oliver	16	8	18.5

In [102…
```
df_c.sort_values(by='date',ascending=True,inplace=True)
df_c
```

Out[1026]:

	date	round	day	venue	result	gf	ga	opponent	xg	xga	poss	attendance	captain	formation	referee	sh	sot	c
4750	2019-08-09	Matchweek 1	7	2	-1	1	4	Liverpool	0.9	1.8	43	53333.0	Grant Hanley	4-2-3-1	Michael Oliver	12	5	1
760	2019-08-09	Matchweek 1	7	1	1	4	1	Norwich City	1.8	0.9	57	53333.0	Jordan Henderson	4-3-3	Michael Oliver	15	7	1
4028	2019-08-09	Matchweek 1	7	1	1	4	1	Norwich City	1.8	0.9	57	53333.0	Jordan Henderson	4-3-3	Michael Oliver	15	7	1
4332	2019-08-10	Matchweek 1	1	2	0	1	1	Bournemouth	1.3	1.3	47	10714.0	Oliver Norwood	3-5-2	Kevin Friend	8	3	1
4598	2019-08-10	Matchweek 1	1	1	-1	0	5	Manchester City	1.1	3.2	43	59870.0	Aaron Cresswell	4-2-3-1	Mike Dean	5	3	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1253	2024-05-19	Matchweek 38	2	2	1	4	2	Brentford	3.4	1.1	46	17124.0	Dan Burn	4-3-3	Simon Hooper	12	7	1
1101	2024-05-19	Matchweek 38	2	1	1	2	0	Wolves	4.5	0.5	67	60059.0	Virgil van Dijk	4-3-3	Chris Kavanagh	36	13	1
1519	2024-05-19	Matchweek 38	2	2	-1	0	2	Liverpool	0.5	4.5	33	60059.0	Max Kilman	3-5-1-1	Chris Kavanagh	4	2	1
1671	2024-05-19	Matchweek 38	2	1	-1	2	4	Fulham	2.0	1.1	41	12027.0	Carlton Morris	3-4-3	Matt Donohue	14	5	1
1329	2024-05-19	Matchweek 38	2	2	-1	1	3	Manchester City	0.4	1.9	29	55097.0	Kurt Zouma	3-4-3	John Brooks	3	2	1

4786 rows × 23 columns

In [102…

```
from unidecode import unidecode

# Clean up the names
df_c['captain'] = df_c['captain'].apply(unidecode)

print("\nDataFrame after unidecode:")
df_c
```

DataFrame after unidecode:

Out[1027]:

	date	round	day	venue	result	gf	ga	opponent	xg	xga	poss	attendance	captain	formation	referee	sh	sot	c
4750	2019-08-09	Matchweek 1	7	2	-1	1	4	Liverpool	0.9	1.8	43	53333.0	Grant Hanley	4-2-3-1	Michael Oliver	12	5	1
760	2019-08-09	Matchweek 1	7	1	1	4	1	Norwich City	1.8	0.9	57	53333.0	Jordan Henderson	4-3-3	Michael Oliver	15	7	1
4028	2019-08-09	Matchweek 1	7	1	1	4	1	Norwich City	1.8	0.9	57	53333.0	Jordan Henderson	4-3-3	Michael Oliver	15	7	1
4332	2019-08-10	Matchweek 1	1	2	0	1	1	Bournemouth	1.3	1.3	47	10714.0	Oliver Norwood	3-5-2	Kevin Friend	8	3	1
4598	2019-08-10	Matchweek 1	1	1	-1	0	5	Manchester City	1.1	3.2	43	59870.0	Aaron Cresswell	4-2-3-1	Mike Dean	5	3	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1253	2024-05-19	Matchweek 38	2	2	1	4	2	Brentford	3.4	1.1	46	17124.0	Dan Burn	4-3-3	Simon Hooper	12	7	1
1101	2024-05-19	Matchweek 38	2	1	1	2	0	Wolves	4.5	0.5	67	60059.0	Virgil van Dijk	4-3-3	Chris Kavanagh	36	13	1
1519	2024-05-19	Matchweek 38	2	2	-1	0	2	Liverpool	0.5	4.5	33	60059.0	Max Kilman	3-5-1-1	Chris Kavanagh	4	2	1
1671	2024-05-19	Matchweek 38	2	1	-1	2	4	Fulham	2.0	1.1	41	12027.0	Carlton Morris	3-4-3	Matt Donohue	14	5	1
1329	2024-05-19	Matchweek 38	2	2	-1	1	3	Manchester City	0.4	1.9	29	55097.0	Kurt Zouma	3-4-3	John Brooks	3	2	1

4786 rows × 23 columns

In [102...]

```
df_c['week'] = df_c['round'].str.replace(r'\D+', '', regex=True)
df_c.pop('round')
df_c
```

Out[1028]:

	date	day	venue	result	gf	ga	opponent	xg	xga	poss	attendance	captain	formation	referee	sh	sot	dist	fk	pk
4750	2019-08-09	7	2	-1	1	4	Liverpool	0.9	1.8	43	53333.0	Grant Hanley	4-2-3-1	Michael Oliver	12	5	17.3	1	0
760	2019-08-09	7	1	1	4	1	Norwich City	1.8	0.9	57	53333.0	Jordan Henderson	4-3-3	Michael Oliver	15	7	17.1	1	0
4028	2019-08-09	7	1	1	4	1	Norwich City	1.8	0.9	57	53333.0	Jordan Henderson	4-3-3	Michael Oliver	15	7	17.1	1	0
4332	2019-08-10	1	2	0	1	1	Bournemouth	1.3	1.3	47	10714.0	Oliver Norwood	3-5-2	Kevin Friend	8	3	19.0	0	0
4598	2019-08-10	1	1	-1	0	5	Manchester City	1.1	3.2	43	59870.0	Aaron Cresswell	4-2-3-1	Mike Dean	5	3	12.7	1	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1253	2024-05-19	2	2	1	4	2	Brentford	3.4	1.1	46	17124.0	Dan Burn	4-3-3	Simon Hooper	12	7	15.4	2	0
1101	2024-05-19	2	1	1	2	0	Wolves	4.5	0.5	67	60059.0	Virgil van Dijk	4-3-3	Chris Kavanagh	36	13	14.8	0	0
1519	2024-05-19	2	2	-1	0	2	Liverpool	0.5	4.5	33	60059.0	Max Kilman	3-5-1-1	Chris Kavanagh	4	2	19.0	1	0
1671	2024-05-19	2	1	-1	2	4	Fulham	2.0	1.1	41	12027.0	Carlton Morris	3-4-3	Matt Donohue	14	5	19.3	2	1
1329	2024-05-19	2	2	-1	1	3	Manchester City	0.4	1.9	29	55097.0	Kurt Zouma	3-4-3	John Brooks	3	2	10.3	0	0

4786 rows × 23 columns

In [102…

```
df_c['season'] = df_c['date'].dt.year
del df_c['date']
df_c
```

Out[1029]:

	day	venue	result	gf	ga	opponent	xg	xga	poss	attendance	captain	formation	referee	sh	sot	dist	fk	pk	pkatt
4750	7	2	-1	1	4	Liverpool	0.9	1.8	43	53333.0	Grant Hanley	4-2-3-1	Michael Oliver	12	5	17.3	1	0	0
760	7	1	1	4	1	Norwich City	1.8	0.9	57	53333.0	Jordan Henderson	4-3-3	Michael Oliver	15	7	17.1	1	0	0
4028	7	1	1	4	1	Norwich City	1.8	0.9	57	53333.0	Jordan Henderson	4-3-3	Michael Oliver	15	7	17.1	1	0	0
4332	1	2	0	1	1	Bournemouth	1.3	1.3	47	10714.0	Oliver Norwood	3-5-2	Kevin Friend	8	3	19.0	0	0	0
4598	1	1	-1	0	5	Manchester City	1.1	3.2	43	59870.0	Aaron Cresswell	4-2-3-1	Mike Dean	5	3	12.7	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1253	2	2	1	4	2	Brentford	3.4	1.1	46	17124.0	Dan Burn	4-3-3	Simon Hooper	12	7	15.4	2	0	0
1101	2	1	1	2	0	Wolves	4.5	0.5	67	60059.0	Virgil van Dijk	4-3-3	Chris Kavanagh	36	13	14.8	0	0	0
1519	2	2	-1	0	2	Liverpool	0.5	4.5	33	60059.0	Max Kilman	3-5-1-1	Chris Kavanagh	4	2	19.0	1	0	0
1671	2	1	-1	2	4	Fulham	2.0	1.1	41	12027.0	Carlton Morris	3-4-3	Matt Donohue	14	5	19.3	2	1	1
1329	2	2	-1	1	3	Manchester City	0.4	1.9	29	55097.0	Kurt Zouma	3-4-3	John Brooks	3	2	10.3	0	0	0

4786 rows × 22 columns

In [103...
```
df_c.rename(columns={'poss' : 'poss_home'},inplace=True)
```

In [103...
```
df_c['poss_away'] = 100-df_c['poss_home']
df_c
```

Out[1031]:

	day	venue	result	gf	ga	opponent	xg	xga	poss_home	attendance	captain	formation	referee	sh	sot	dist	fk	pk
4750	7	2	-1	1	4	Liverpool	0.9	1.8	43	53333.0	Grant Hanley	4-2-3-1	Michael Oliver	12	5	17.3	1	0
760	7	1	1	4	1	Norwich City	1.8	0.9	57	53333.0	Jordan Henderson	4-3-3	Michael Oliver	15	7	17.1	1	0
4028	7	1	1	4	1	Norwich City	1.8	0.9	57	53333.0	Jordan Henderson	4-3-3	Michael Oliver	15	7	17.1	1	0
4332	1	2	0	1	1	Bournemouth	1.3	1.3	47	10714.0	Oliver Norwood	3-5-2	Kevin Friend	8	3	19.0	0	0
4598	1	1	-1	0	5	Manchester City	1.1	3.2	43	59870.0	Aaron Cresswell	4-2-3-1	Mike Dean	5	3	12.7	1	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1253	2	2	1	4	2	Brentford	3.4	1.1	46	17124.0	Dan Burn	4-3-3	Simon Hooper	12	7	15.4	2	0
1101	2	1	1	2	0	Wolves	4.5	0.5	67	60059.0	Virgil van Dijk	4-3-3	Chris Kavanagh	36	13	14.8	0	0
1519	2	2	-1	0	2	Liverpool	0.5	4.5	33	60059.0	Max Kilman	3-5-1-1	Chris Kavanagh	4	2	19.0	1	0
1671	2	1	-1	2	4	Fulham	2.0	1.1	41	12027.0	Carlton Morris	3-4-3	Matt Donohue	14	5	19.3	2	1
1329	2	2	-1	1	3	Manchester City	0.4	1.9	29	55097.0	Kurt Zouma	3-4-3	John Brooks	3	2	10.3	0	0

4786 rows × 23 columns

In [103…

```
df_c.to_csv('D:\\projects\\datasets\\clean_data_matches.csv', index=False)
```

Exploratory Data Analysis (EDA)

In [103…

```
data = pd.read_csv('D:\\projects\\datasets\\clean_data_matches.csv',encoding='utf-8')
data
```

Out[1033]:

	day	venue	result	gf	ga	opponent	xg	xga	poss_home	attendance	captain	formation	referee	sh	sot	dist	fk	pk
0	7	2	-1	1	4	Liverpool	0.9	1.8	43	53333.0	Grant Hanley	4-2-3-1	Michael Oliver	12	5	17.3	1	0
1	7	1	1	4	1	Norwich City	1.8	0.9	57	53333.0	Jordan Henderson	4-3-3	Michael Oliver	15	7	17.1	1	0
2	7	1	1	4	1	Norwich City	1.8	0.9	57	53333.0	Jordan Henderson	4-3-3	Michael Oliver	15	7	17.1	1	0
3	1	2	0	1	1	Bournemouth	1.3	1.3	47	10714.0	Oliver Norwood	3-5-2	Kevin Friend	8	3	19.0	0	0
4	1	1	-1	0	5	Manchester City	1.1	3.2	43	59870.0	Aaron Cresswell	4-2-3-1	Mike Dean	5	3	12.7	1	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4781	2	2	1	4	2	Brentford	3.4	1.1	46	17124.0	Dan Burn	4-3-3	Simon Hooper	12	7	15.4	2	0
4782	2	1	1	2	0	Wolves	4.5	0.5	67	60059.0	Virgil van Dijk	4-3-3	Chris Kavanagh	36	13	14.8	0	0
4783	2	2	-1	0	2	Liverpool	0.5	4.5	33	60059.0	Max Kilman	3-5-1-1	Chris Kavanagh	4	2	19.0	1	0
4784	2	1	-1	2	4	Fulham	2.0	1.1	41	12027.0	Carlton Morris	3-4-3	Matt Donohue	14	5	19.3	2	1
4785	2	2	-1	1	3	Manchester City	0.4	1.9	29	55097.0	Kurt Zouma	3-4-3	John Brooks	3	2	10.3	0	0

4786 rows × 23 columns

Features description

---

day : the day of the week of the game (1 - 2 - 3 - 4 - 5 - 6 - 7) = (sat - sun - mon - tue - wed - thu - fri)

venue : the venue of the game (home = 1 - away = 2)

result : the result of the game (win = 1 - lose = -1 - draw = 0)

gf : the goals for the home team *****

ga : the goals for the away team

opponent: the opponent of the home team

xg : the expected goals for the home team

xga : the expected goals for the away team

poss_home : the possession of the home team ----------

captain : the captain of the home team

formation : the formation of the home team

referee : the referee of the game

sh : the shots of the home team ----------

sot : the shots on target of the home team ----------

dist : the average distance of the shots of the home team ----------

fk : the free kicks of the home team ----------

pk : the penalty kicks of the home team ----------

pka : the penalty kicks attempted of the home team

season : the season year of the match (2019 - 2020 - 2021 - 2022 - 2023 - 2024)

team: the home team

poss_away : the possession of the home team

In [103…

```
teams = data.groupby('team')[['gf','poss_home','sh','sot','dist','fk','pk']].mean()
teams.sort_values(by='team',ascending=True,inplace=True)
teams.reset_index(inplace=True)
teams
```

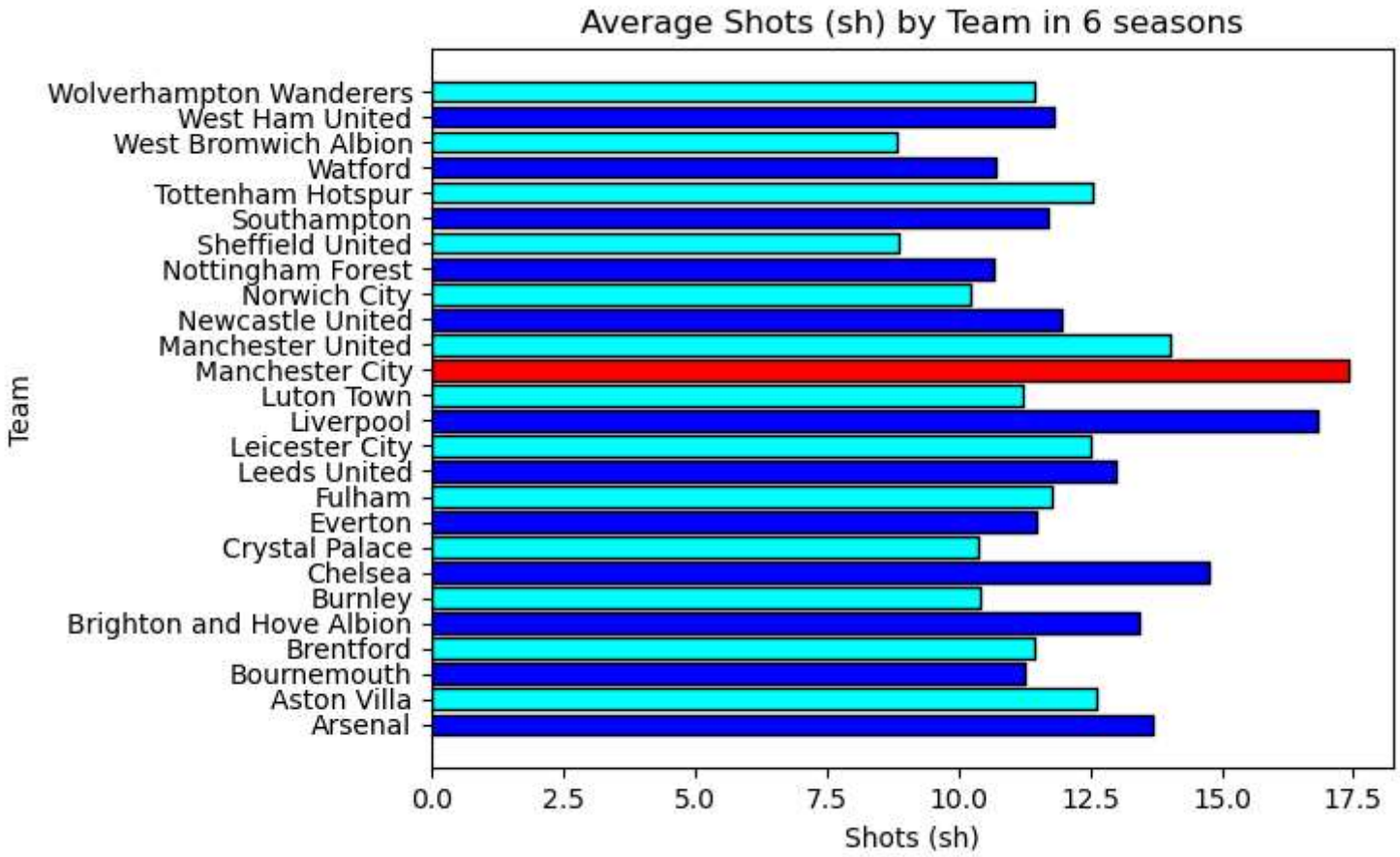
Out[1034]:

	team	gf	poss_home	sh	sot	dist	fk	pk
0	Arsenal	1.780702	55.171053	13.710526	4.451754	16.943421	0.473684	0.144737
1	Aston Villa	1.447368	48.315789	12.631579	4.337719	17.425439	0.464912	0.092105
2	Bournemouth	1.149123	42.973684	11.245614	3.666667	16.477193	0.464912	0.052632
3	Brentford	1.421053	44.587719	11.456140	3.921053	16.102632	0.368421	0.140351
4	Brighton and Hove Albion	1.263158	54.802632	13.421053	4.232456	17.242982	0.346491	0.122807
5	Burnley	0.968421	42.721053	10.426316	3.305263	17.205789	0.394737	0.068421
6	Chelsea	1.672932	60.236842	14.770677	5.078947	17.050376	0.556391	0.195489
7	Crystal Palace	1.141593	44.371681	10.362832	3.557522	17.143805	0.473451	0.079646
8	Everton	1.118421	44.254386	11.482456	3.815789	17.092544	0.412281	0.087719
9	Fulham	1.078947	49.671053	11.782895	3.697368	17.848684	0.361842	0.085526
10	Leeds United	1.407895	53.473684	12.993421	4.401316	17.326974	0.328947	0.092105
11	Leicester City	1.679825	53.732456	12.530702	4.438596	18.091667	0.473684	0.153509
12	Liverpool	2.109023	62.090226	16.838346	5.860902	16.905639	0.458647	0.139098
13	Luton Town	1.368421	42.368421	11.236842	3.526316	16.157895	0.289474	0.131579
14	Manchester City	2.477444	65.338346	17.394737	6.120301	16.802256	0.593985	0.176692
15	Manchester United	1.691729	54.289474	14.007519	5.116541	18.201880	0.609023	0.195489
16	Newcastle United	1.434211	43.574561	11.978070	4.008772	17.789912	0.451754	0.127193
17	Norwich City	0.644737	46.078947	10.223684	3.026316	18.485526	0.447368	0.065789
18	Nottingham Forest	1.144737	39.092105	10.684211	3.342105	17.175000	0.210526	0.052632
19	Sheffield United	0.750000	40.572368	8.881579	2.697368	17.123026	0.144737	0.078947
20	Southampton	1.178947	49.010526	11.689474	4.094737	18.198421	0.568421	0.094737
21	Tottenham Hotspur	1.770677	52.936090	12.541353	4.586466	17.837594	0.575188	0.105263
22	Watford	0.921053	41.828947	10.697368	3.092105	17.876316	0.552632	0.105263
23	West Bromwich Albion	0.921053	37.947368	8.815789	2.789474	18.497368	0.421053	0.105263
24	West Ham United	1.469298	43.614035	11.820175	3.916667	16.371930	0.394737	0.096491

	team	gf	poss_home	sh	sot	dist	fk	pk
25	Wolverhampton Wanderers	1.061404	49.228070	11.447368	3.706140	17.750000	0.403509	0.087719

In [103…

```
tm = teams['team']
shoots = teams['sh']
max_shoots = shoots.argmax()
colors = ['blue','cyan']*len(tm)
colors[max_shoots]='red'
plt.barh(teams['team'], teams['sh'], color=colors, edgecolor='black')

plt.xlabel('Shots (sh)')
plt.ylabel('Team')
plt.title('Average Shots (sh) by Team in 6 seasons')
plt.show()
```

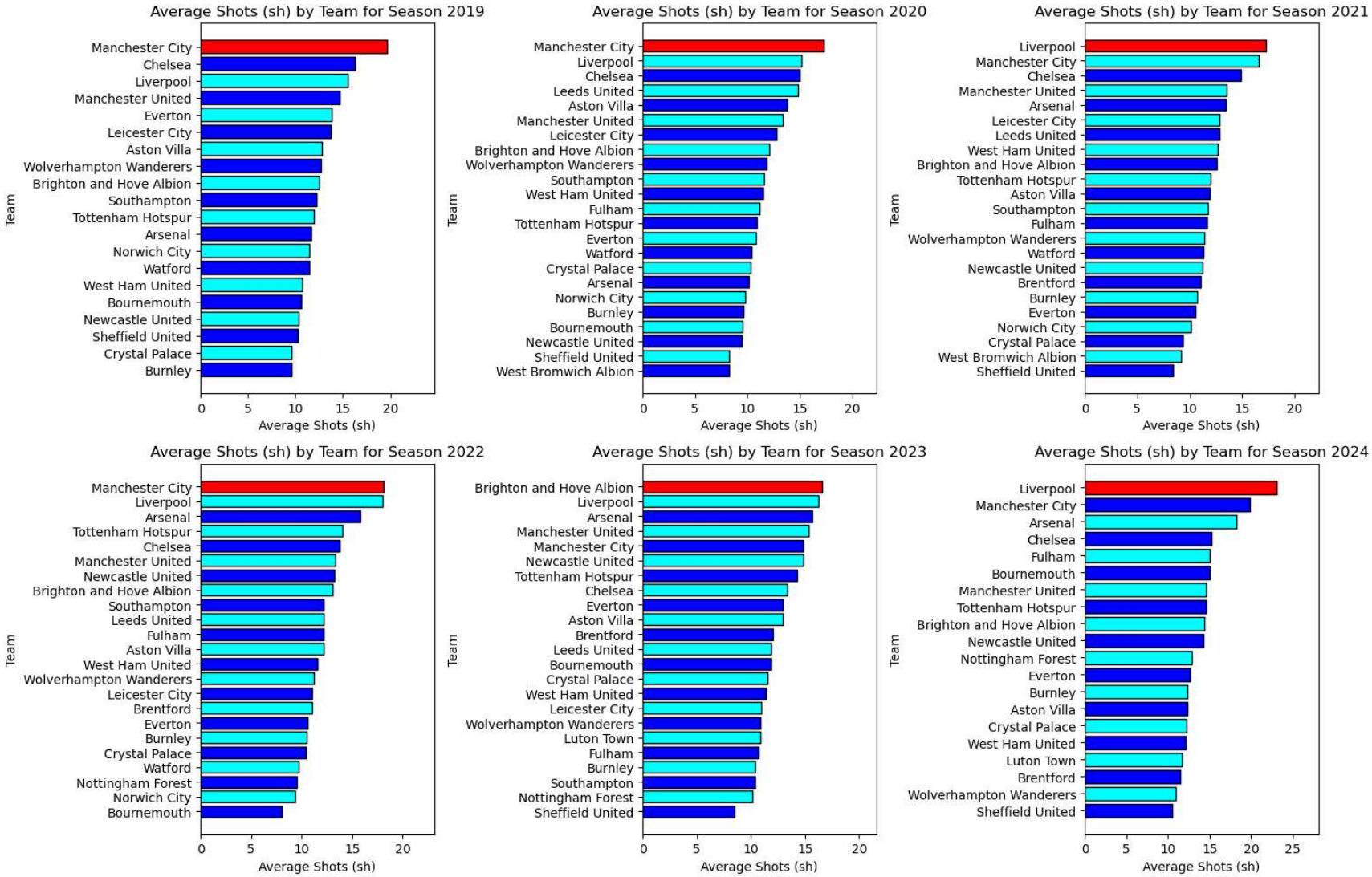


manchester city has the biggest Average shots in the premier league

west bromwich albion and sheffield united have the lowest shots rate

In [103…

```python
if 'season' not in data.columns:
    raise ValueError("The 'season' column is missing from the dataset.")

# Get unique seasons
seasons = data['season'].unique()

fig, axs = plt.subplots(nrows=2, ncols=3, figsize=(15, 10))
axs = axs.flatten()  # Flatten the 2D array of axes to 1D for easy indexing

# Loop through each season and create a horizontal bar plot
for i, season in enumerate(seasons):
    # Group by 'team' for the specific season and calculate the mean of 'sh'
    teams_season = data[data['season'] == season].groupby('team')['sh'].mean().sort_values(ascending=True)

    # Get teams and their average shots
    tm = teams_season.index
    shoots = teams_season.values

    # Find the index of the maximum shots
    max_shoots_index = shoots.argmax()

    # Create a colors list for the bars
    colors = ['blue', 'cyan'] * (len(tm) // 2) + ['blue'] * (len(tm) % 2)

    # Set the color of the team with the maximum shots to red
    colors[max_shoots_index] = 'red'

    # Create the horizontal bar plot
    axs[i].barh(tm, shoots, color=colors, edgecolor='black')

    axs[i].set_xlabel('Average Shots (sh)')
    axs[i].set_ylabel('Team')
    axs[i].set_title(f'Average Shots (sh) by Team for Season {season}')
    axs[i].set_xlim(0, shoots.max() + 5)  # Set x-limit for better visibility

# Adjust layout for better spacing
plt.tight_layout()
plt.show()
```

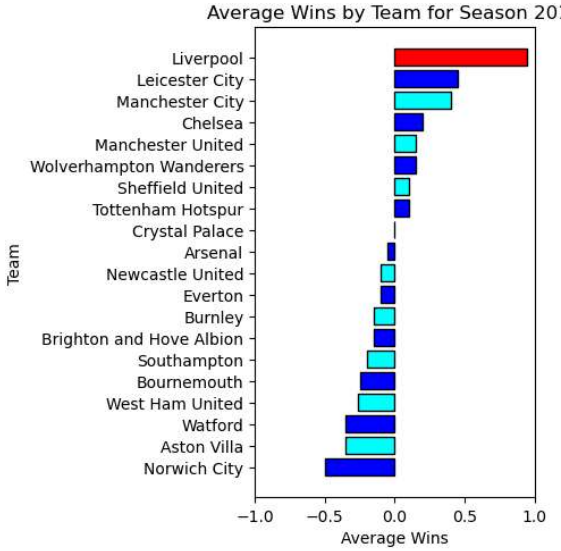


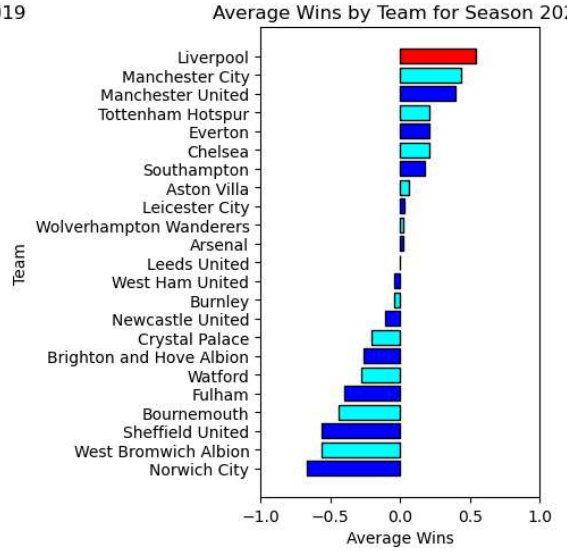
manchester city and liverpool have much shoots on the goal during the match

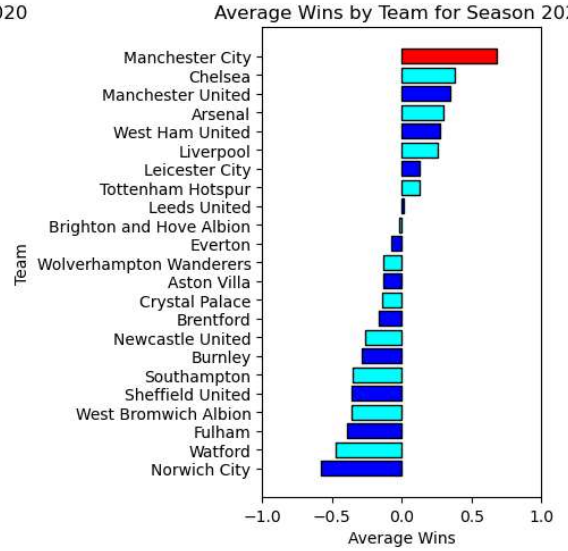
In [103…

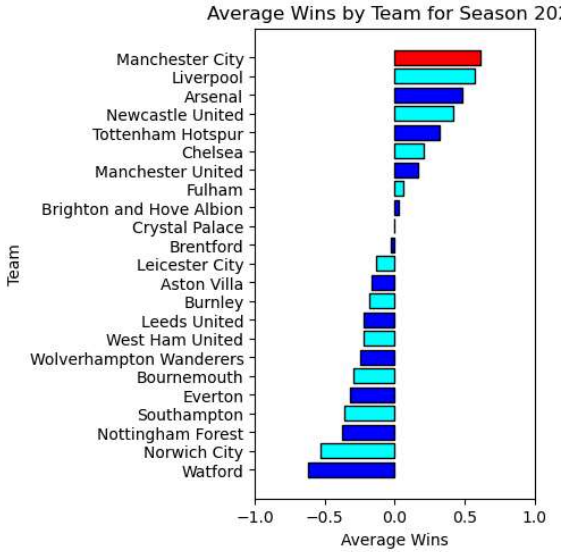
```
# Ensure the 'season' column is in the dataset
if 'season' not in data.columns:
    raise ValueError("The 'season' column is missing from the dataset.")
```

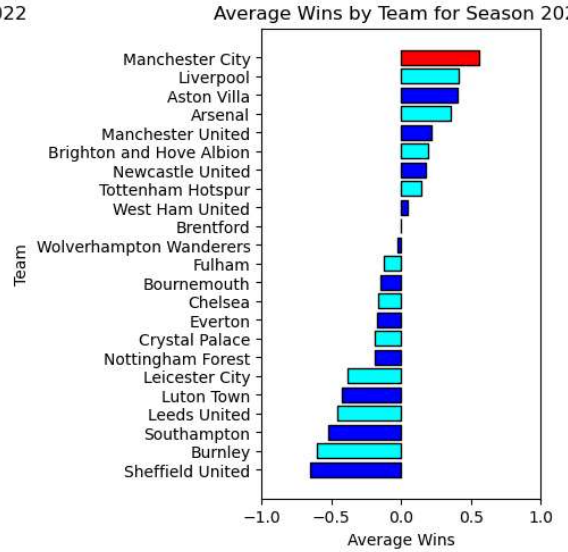
```python
# Get unique seasons
seasons = data['season'].unique()


fig, axs = plt.subplots(nrows=2, ncols=3, figsize=(15, 10))
axs = axs.flatten()  # Flatten the 2D array of axes to 1D for easy indexing

# Loop through each season and create a horizontal bar plot
for i, season in enumerate(seasons):
    # Group by 'team' for the specific season and calculate the mean of 'result'
    teams_season = data[data['season'] == season].groupby('team')['result'].mean().sort_values(ascending=True)

    # Get teams and their average wins
    tm = teams_season.index
    wins = teams_season.values

    # Find the index of the maximum wins
    max_wins_index = wins.argmax()

    # Create a colors list for the bars
    colors = ['blue', 'cyan'] * (len(tm) // 2) + ['blue'] * (len(tm) % 2)

    # Set the color of the team with the maximum wins to red
    colors[max_wins_index] = 'red'

    # Create the horizontal bar plot
    axs[i].barh(tm, wins, color=colors, edgecolor='black')

    axs[i].set_xlabel('Average Wins')
    axs[i].set_ylabel('Team')
    axs[i].set_title(f'Average Wins by Team for Season {season}')
    axs[i].set_xlim(-1, 1)  # Set x-limit for better visibility, as wins can be -1, 0, or 1

# Adjust layout for better spacing
plt.tight_layout()
plt.show()
```

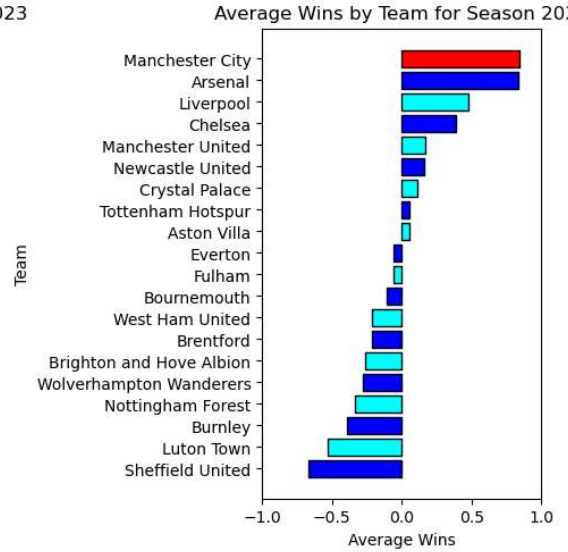












Manchester City boasts a formidable team, backed by an impressive four-year winning record

On the other hand, Norwich City has struggled, consistently ranking as one of the weakest teams in the Premier League.

---

detecting outliers

In [103…

```
new = pd.DataFrame()
for column in data.select_dtypes(include=['float64', 'int64']):
    if(column=='pk' or column=='pkatt'):
        new[column] = data[column]
        continue
    shoots = data[column]
    q1 = shoots.quantile(0.25)
    q3 = shoots.quantile(0.75)
    iqr = q3 - q1

    lower_bound = q1 - 1.5 * iqr
    upper_bound = q3 + 1.5 * iqr

    clean = shoots[(shoots >= lower_bound) & (shoots <= upper_bound)]

    new[column] = clean

new
```

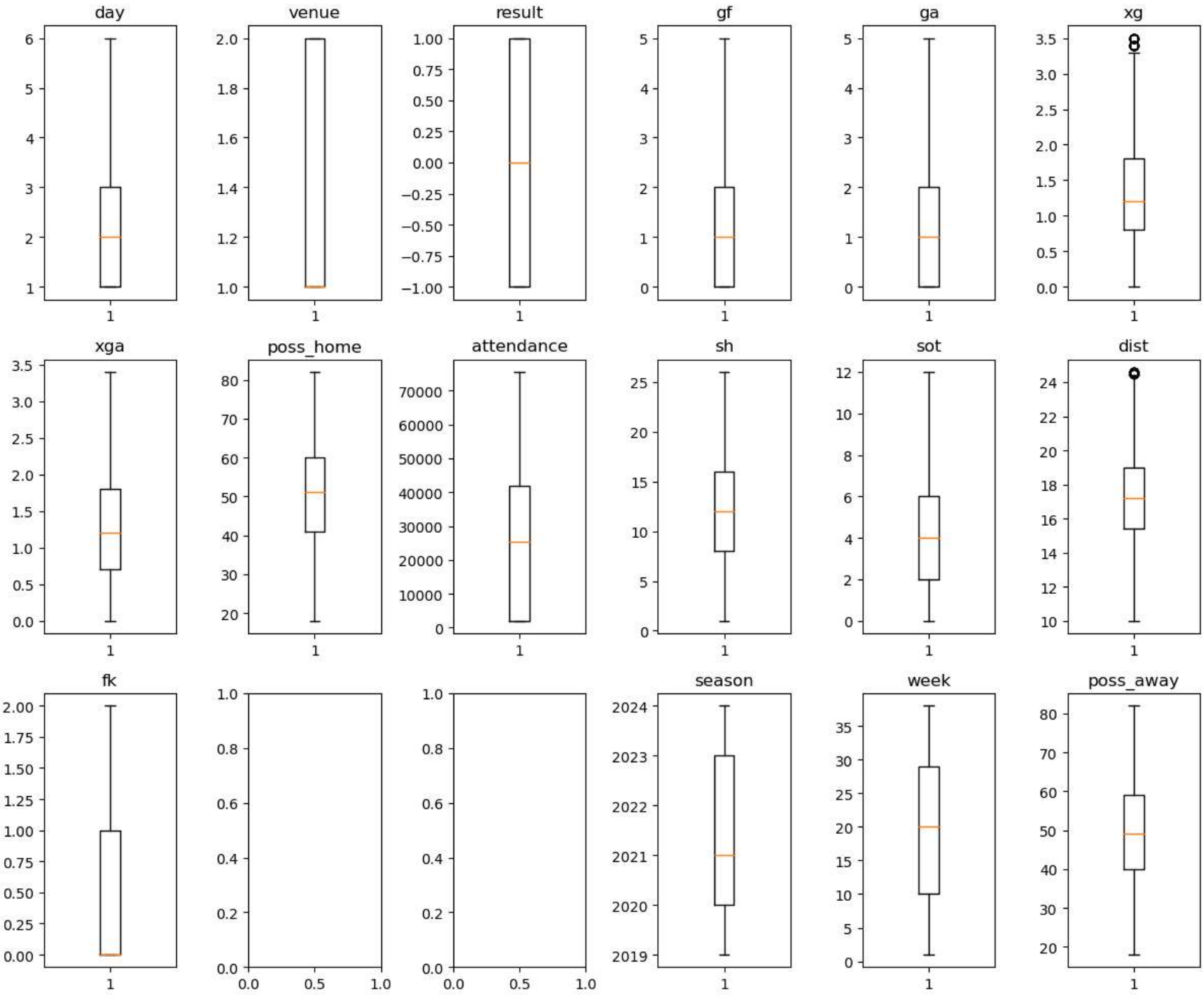
Out[1038]:

	day	venue	result	gf	ga	xg	xga	poss_home	attendance	sh	sot	dist	fk	pk	pkatt	season	week	poss_away
3	1	2	0	1.0	1.0	1.3	1.3	47	10714.0	8.0	3.0	19.0	0.0	0	0	2019	1	53
4	1	1	-1	0.0	5.0	1.1	3.2	43	59870.0	5.0	3.0	12.7	1.0	0	0	2019	1	57
5	1	1	1	3.0	1.0	2.4	0.7	70	60407.0	NaN	6.0	18.6	2.0	0	0	2019	1	30
6	1	2	-1	0.0	3.0	1.2	0.9	53	19784.0	11.0	3.0	18.7	1.0	0	0	2019	1	47
7	1	2	0	0.0	0.0	1.1	0.9	64	25151.0	10.0	2.0	17.1	0.0	0	0	2019	1	36
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4781	2	2	1	4.0	2.0	3.4	1.1	46	17124.0	12.0	7.0	15.4	2.0	0	0	2024	38	54
4782	2	1	1	2.0	0.0	NaN	0.5	67	60059.0	NaN	NaN	14.8	0.0	0	0	2024	38	33
4783	2	2	-1	0.0	2.0	0.5	NaN	33	60059.0	4.0	2.0	19.0	1.0	0	0	2024	38	67
4784	2	1	-1	2.0	4.0	2.0	1.1	41	12027.0	14.0	5.0	19.3	2.0	1	1	2024	38	59
4785	2	2	-1	1.0	3.0	0.4	1.9	29	55097.0	3.0	2.0	10.3	0.0	0	0	2024	38	71

4627 rows × 18 columns

In [103…

```
fig, axs = plt.subplots(nrows=3, ncols=6, figsize=(12, 10))
axs = axs.flatten()


for idx, column in enumerate(new.columns):
    if(column=='pk' or column=='pkatt'):
        continue

    axs[idx].boxplot(new[column].dropna())
    axs[idx].set_title(column)

plt.tight_layout()
plt.show()
```


day
venue
result
gf
ga
xg
xga
poss_home
attendance
sh
sot
dist
fk
season
week
poss_away

In [104…

```
print(new.isna().sum().to_frame(),"\n\n")

new.dropna(inplace=True)
print(new.isna().sum().to_frame())
```

```
            0
day         0
venue       0
result      0
gf         33
ga         31
xg         74
xga        81
poss_home   0
attendance  0
sh         67
sot        15
dist       96
fk         42
pk          0
pkatt       0
season      0
week        0
poss_away   0
```

```
            0
day         0
venue       0
result      0
gf          0
ga          0
xg          0
xga         0
poss_home   0
attendance  0
sh          0
sot         0
dist        0
fk          0
pk          0
pkatt       0
season      0
week        0
poss_away   0
```

In [104…

```
new
```

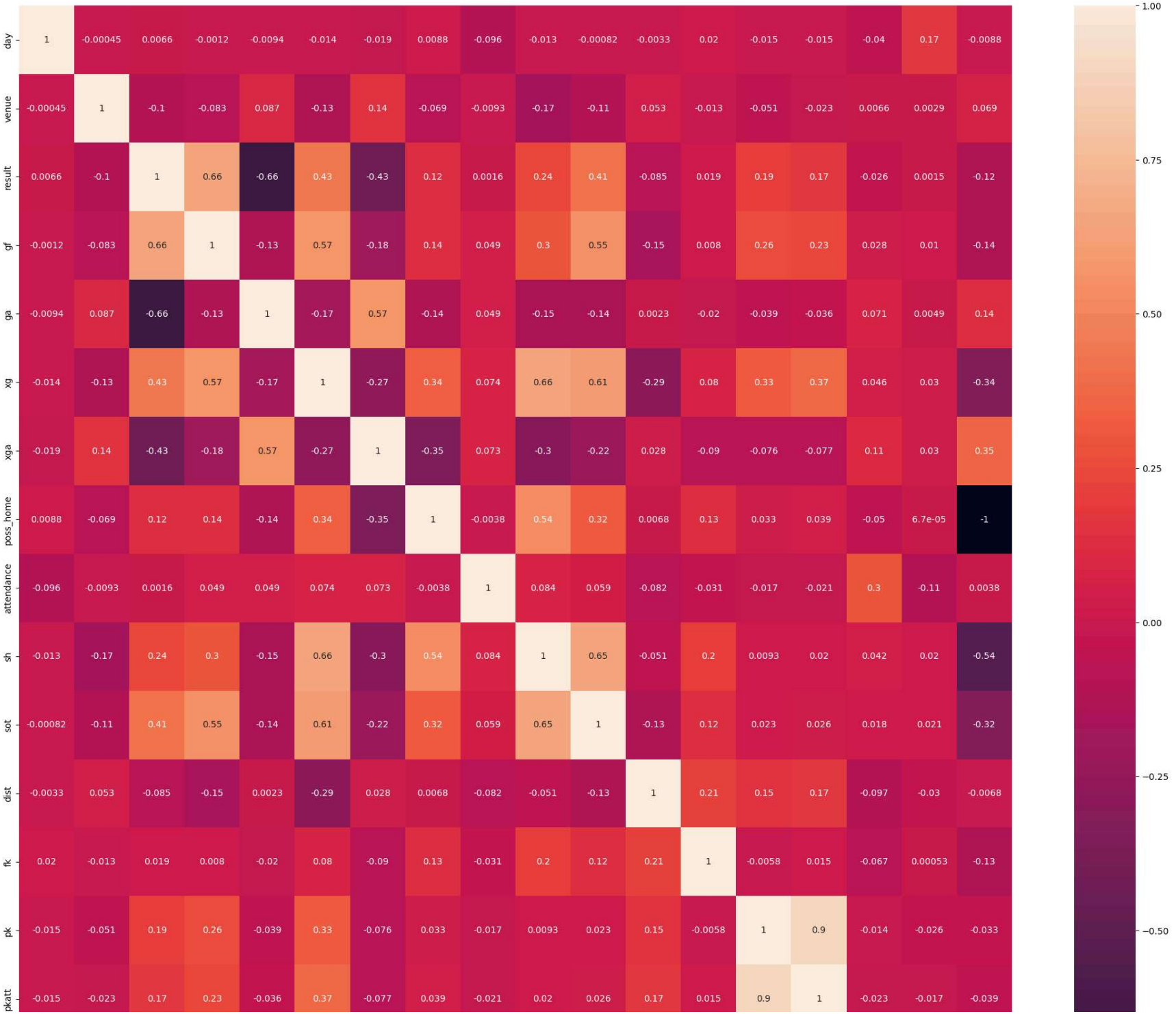
Out[1041]:

	day	venue	result	gf	ga	xg	xga	poss_home	attendance	sh	sot	dist	fk	pk	pkatt	season	week	poss_away
3	1	2	0	1.0	1.0	1.3	1.3	47	10714.0	8.0	3.0	19.0	0.0	0	0	2019	1	53
4	1	1	-1	0.0	5.0	1.1	3.2	43	59870.0	5.0	3.0	12.7	1.0	0	0	2019	1	57
6	1	2	-1	0.0	3.0	1.2	0.9	53	19784.0	11.0	3.0	18.7	1.0	0	0	2019	1	47
7	1	2	0	0.0	0.0	1.1	0.9	64	25151.0	10.0	2.0	17.1	0.0	0	0	2019	1	36
8	1	1	1	3.0	0.0	0.9	1.2	47	19784.0	10.0	4.0	15.0	0.0	0	0	2019	1	53
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4779	2	2	-1	0.0	5.0	0.9	2.5	46	25191.0	8.0	2.0	13.8	0.0	0	0	2024	38	54
4780	2	1	-1	1.0	2.0	1.2	1.7	72	21109.0	20.0	3.0	22.4	1.0	0	0	2024	38	28
4781	2	2	1	4.0	2.0	3.4	1.1	46	17124.0	12.0	7.0	15.4	2.0	0	0	2024	38	54
4784	2	1	-1	2.0	4.0	2.0	1.1	41	12027.0	14.0	5.0	19.3	2.0	1	1	2024	38	59
4785	2	2	-1	1.0	3.0	0.4	1.9	29	55097.0	3.0	2.0	10.3	0.0	0	0	2024	38	71

4248 rows × 18 columns

In [104…

```
corr = new.corr()
plt.figure(figsize=(25,25))
sns.heatmap(corr,annot=True)
plt.show()
```


day 1 -0.00045 0.0066 -0.0012 -0.0094 -0.014 -0.019 0.0088 -0.096 -0.013 -0.00082 -0.0033 0.02 -0.015 -0.015 -0.04 0.17 -0.0088
venue -0.00045 1 -0.1 -0.083 0.087 -0.13 0.14 -0.069 -0.0093 -0.17 -0.11 0.053 -0.013 -0.051 -0.023 0.0066 0.0029 0.069
result 0.0066 -0.1 1 0.66 -0.66 0.43 -0.43 0.12 0.0016 0.24 0.41 -0.085 0.019 0.19 0.17 -0.026 0.0015 -0.12
gf -0.0012 -0.083 0.66 1 -0.13 0.57 -0.18 0.14 0.049 0.3 0.55 -0.15 0.008 0.26 0.23 0.028 0.01 -0.14
ga -0.0094 0.087 -0.66 -0.13 1 -0.17 0.57 -0.14 0.049 -0.15 -0.14 0.0023 -0.02 -0.039 -0.036 0.071 0.0049 0.14
xg -0.014 -0.13 0.43 0.57 -0.17 1 -0.27 0.34 0.074 0.66 0.61 -0.29 0.08 0.33 0.37 0.046 0.03 -0.34
xga -0.019 0.14 -0.43 -0.18 0.57 -0.27 1 -0.35 0.073 -0.3 -0.22 0.028 -0.09 -0.076 -0.077 0.11 0.03 0.35
poss_home 0.0088 -0.069 0.12 0.14 -0.14 0.34 -0.35 1 -0.0038 0.54 0.32 0.0068 0.13 0.033 0.039 -0.05 6.7e-05 -1
attendance -0.096 -0.0093 0.0016 0.049 0.049 0.074 0.073 -0.0038 1 0.084 0.059 -0.082 -0.031 -0.017 -0.021 0.3 -0.11 0.0038
sh -0.013 -0.17 0.24 0.3 -0.15 0.66 -0.3 0.54 0.084 1 0.65 -0.051 0.2 0.0093 0.02 0.042 0.02 -0.54
sot -0.00082 -0.11 0.41 0.55 -0.14 0.61 -0.22 0.32 0.059 0.65 1 -0.13 0.12 0.023 0.026 0.018 0.021 -0.32
dist -0.0033 0.053 -0.085 -0.15 0.0023 -0.29 0.028 0.0068 -0.082 -0.051 -0.13 1 0.21 0.15 0.17 -0.097 -0.03 -0.0068
fk 0.02 -0.013 0.019 0.008 -0.02 0.08 -0.09 0.13 -0.031 0.2 0.12 0.21 1 -0.0058 0.015 -0.067 0.00053 -0.13
pk -0.015 -0.051 0.19 0.26 -0.039 0.33 -0.076 0.033 -0.017 0.0093 0.023 0.15 -0.0058 1 0.9 -0.014 -0.026 -0.033
pkatt -0.015 -0.023 0.17 0.23 -0.036 0.37 -0.077 0.039 -0.021 0.02 0.026 0.17 0.015 0.9 1 -0.023 -0.017 -0.039
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50

In [104…

```
x = new[['gf', 'ga', 'xg', 'xga', 'poss_home', 'attendance', 'sh', 'dist', 'fk', 'pk', 'pkatt']]
y = new['result']


x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)

model = LinearRegression()
model.fit(x_train, y_train)

y_pred = model.predict(x_test)

mse = mean_squared_error(y_test, y_pred)
R = model.score(x_train,y_train)

n = len(y_test)
p = x_test.shape[1]
adjusted_r_squared = 1 - (1 - R) * (n - 1) / (n - p - 1)


print(f"Slope (Coefficient): {model.coef_[0]}")
print(f"Intercept: {model.intercept_}")
print(f"Mean Squared Error: {mse}")
print(f"R^2: {R}")
print(f"adjusted R squared : {adjusted_r_squared}")
```

```
Slope (Coefficient): 0.4219254301592154
Intercept: 0.09285844741956883
Mean Squared Error: 0.18393711154486617
R^2: 0.7746360937471435
adjusted R squared : 0.7716778563142301
```

Slope (Coefficient: 0.422): This coefficient suggests that for every unit increase in the predictor variable (the independent variable(s) used in the model), the expected change in the dependent variable (the outcome you are predicting) is approximately 0.422. This indicates a positive relationship, meaning that as the predictor variable increases, the outcome variable also tends to increase.

Intercept (0.093): The intercept represents the expected value of the dependent variable when all independent variables are equal to zero. While the practical interpretation of the intercept can vary depending on the context of the data, it indicates that even without any contributions from the predictor variables, there is a baseline value for the outcome.

Mean Squared Error (MSE: 0.184): The MSE indicates the average squared difference between the observed actual outcomes and the predictions made by the model. A lower MSE value suggests that the model's predictions are relatively close to the actual outcomes. In this case, an MSE of 0.184 suggests that while there are some discrepancies between predicted and actual values, the model is generally performing well.

R-squared (R^2: 0.775): The R-squared value indicates that approximately 77.5% of the variability in the dependent variable can be explained by the independent variable(s) included in the model. This is a strong indication of the model's explanatory power, suggesting that the chosen predictor variables are significantly contributing to the prediction of the outcome.

Conclusion: Overall, the model appears to have a good fit, as indicated by the R-squared value, while the slope shows a positive correlation between the predictor and the outcome variable. However, it is essential to consider the context of the data and possibly evaluate the model further through additional diagnostics to ensure its robustness and reliability.

In [ ]: