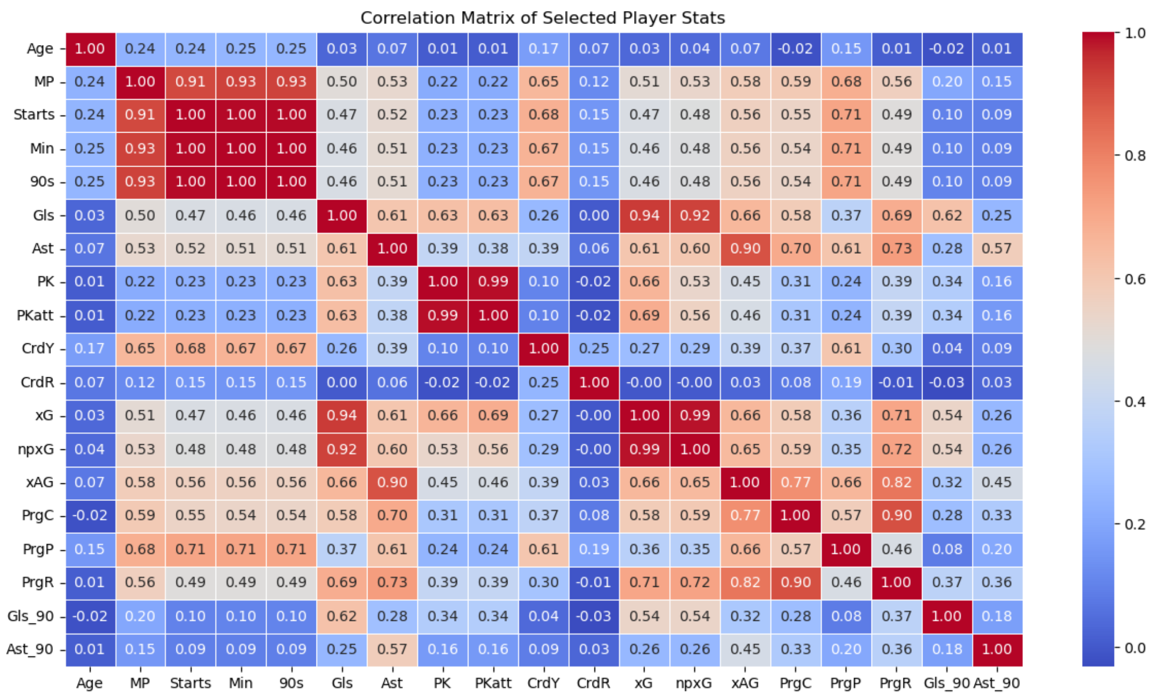
In [545…
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
```

In [479…
```
df = pd.read_csv('datasetpremier.csv')
```

In [ ]:
```
Heatmap ดู Correlation ของค่าต่าง ๆ
```

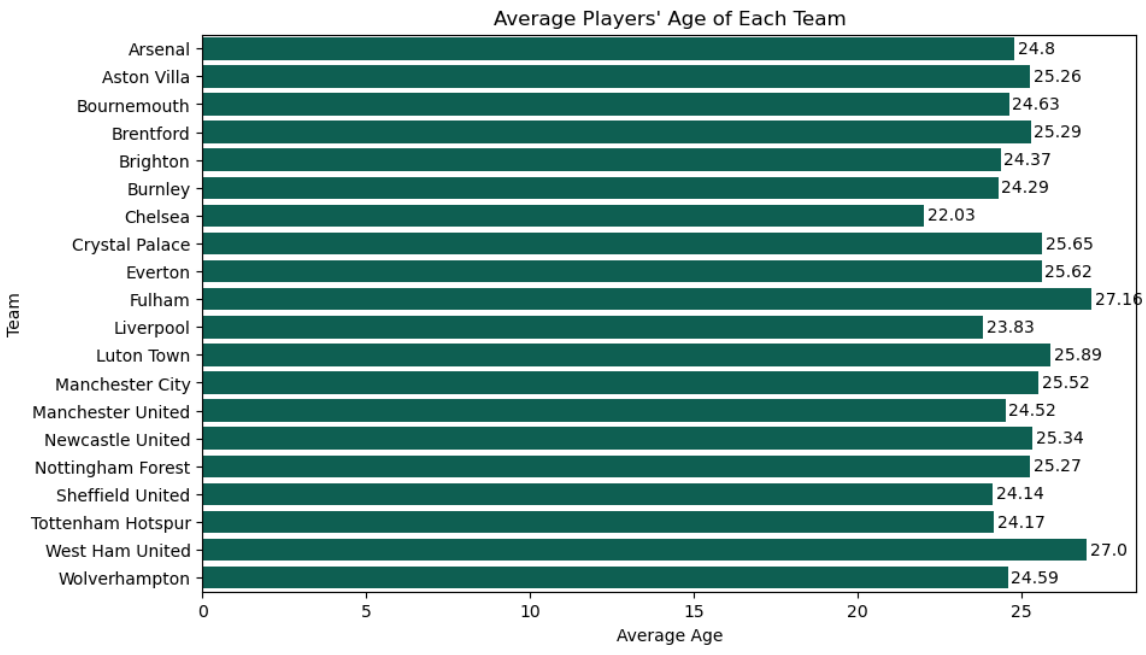
In [481…
```
selected_columns = ['Age', 'MP', 'Starts', 'Min', '90s', 'Gls', 'As
                    'PK', 'PKatt', 'CrdY', 'CrdR', 'xG', 'npxG', 'x
                    'PrgC', 'PrgP', 'PrgR', 'Gls_90', 'Ast_90']

corr_matrix = df[selected_columns].corr()
plt.figure(figsize=(15, 8))
sns.heatmap(corr_matrix, cmap="coolwarm", annot=True, fmt=".2f", li
plt.title("Correlation Matrix of Selected Player Stats")
plt.show()
```



	Age	MP	Starts	Min	90s	Gls	Ast	PK	PKatt	CrdY	CrdR	xG	npxG	xAG	PrgC	PrgP	PrgR	Gls_90	Ast_90
Age	1.00	0.24	0.24	0.25	0.25	0.03	0.07	0.01	0.01	0.17	0.07	0.03	0.04	0.07	-0.02	0.15	0.01	-0.02	0.01
MP	0.24	1.00	0.91	0.93	0.93	0.50	0.53	0.22	0.22	0.65	0.12	0.51	0.53	0.58	0.59	0.68	0.56	0.20	0.15
Starts	0.24	0.91	1.00	1.00	1.00	0.47	0.52	0.23	0.23	0.68	0.15	0.47	0.48	0.56	0.55	0.71	0.49	0.10	0.09
Min	0.25	0.93	1.00	1.00	1.00	0.46	0.51	0.23	0.23	0.67	0.15	0.46	0.48	0.56	0.54	0.71	0.49	0.10	0.09
90s	0.25	0.93	1.00	1.00	1.00	0.46	0.51	0.23	0.23	0.67	0.15	0.46	0.48	0.56	0.54	0.71	0.49	0.10	0.09
Gls	0.03	0.50	0.47	0.46	0.46	1.00	0.61	0.63	0.63	0.26	0.00	0.94	0.92	0.66	0.58	0.37	0.69	0.62	0.25
Ast	0.07	0.53	0.52	0.51	0.51	0.61	1.00	0.39	0.38	0.39	0.06	0.61	0.60	0.90	0.70	0.61	0.73	0.28	0.57
PK	0.01	0.22	0.23	0.23	0.23	0.63	0.39	1.00	0.99	0.10	-0.02	0.66	0.53	0.45	0.31	0.24	0.39	0.34	0.16
PKatt	0.01	0.22	0.23	0.23	0.23	0.63	0.38	0.99	1.00	0.10	-0.02	0.69	0.56	0.46	0.31	0.24	0.39	0.34	0.16
CrdY	0.17	0.65	0.68	0.67	0.67	0.26	0.39	0.10	0.10	1.00	0.25	0.27	0.29	0.39	0.37	0.61	0.30	0.04	0.09
CrdR	0.07	0.12	0.15	0.15	0.15	0.00	0.06	-0.02	-0.02	0.25	1.00	-0.00	-0.00	0.03	0.08	0.19	-0.01	-0.03	0.03
xG	0.03	0.51	0.47	0.46	0.46	0.94	0.61	0.66	0.69	0.27	-0.00	1.00	0.99	0.66	0.58	0.36	0.71	0.54	0.26
npxG	0.04	0.53	0.48	0.48	0.48	0.92	0.60	0.53	0.56	0.29	-0.00	0.99	1.00	0.65	0.59	0.35	0.72	0.54	0.26
xAG	0.07	0.58	0.56	0.56	0.56	0.66	0.90	0.45	0.46	0.39	0.03	0.66	0.65	1.00	0.77	0.66	0.82	0.32	0.45
PrgC	-0.02	0.59	0.55	0.54	0.54	0.58	0.70	0.31	0.31	0.37	0.08	0.58	0.59	0.77	1.00	0.57	0.90	0.28	0.33
PrgP	0.15	0.68	0.71	0.71	0.71	0.37	0.61	0.24	0.24	0.61	0.19	0.36	0.35	0.66	0.57	1.00	0.46	0.08	0.20
PrgR	0.01	0.56	0.49	0.49	0.49	0.69	0.73	0.39	0.39	0.30	-0.01	0.71	0.72	0.82	0.90	0.46	1.00	0.37	0.36
Gls_90	-0.02	0.20	0.10	0.10	0.10	0.62	0.28	0.34	0.34	0.04	-0.03	0.54	0.54	0.32	0.28	0.08	0.37	1.00	0.18
Ast_90	0.01	0.15	0.09	0.09	0.09	0.25	0.57	0.16	0.16	0.09	0.03	0.26	0.26	0.45	0.33	0.20	0.36	0.18	1.00

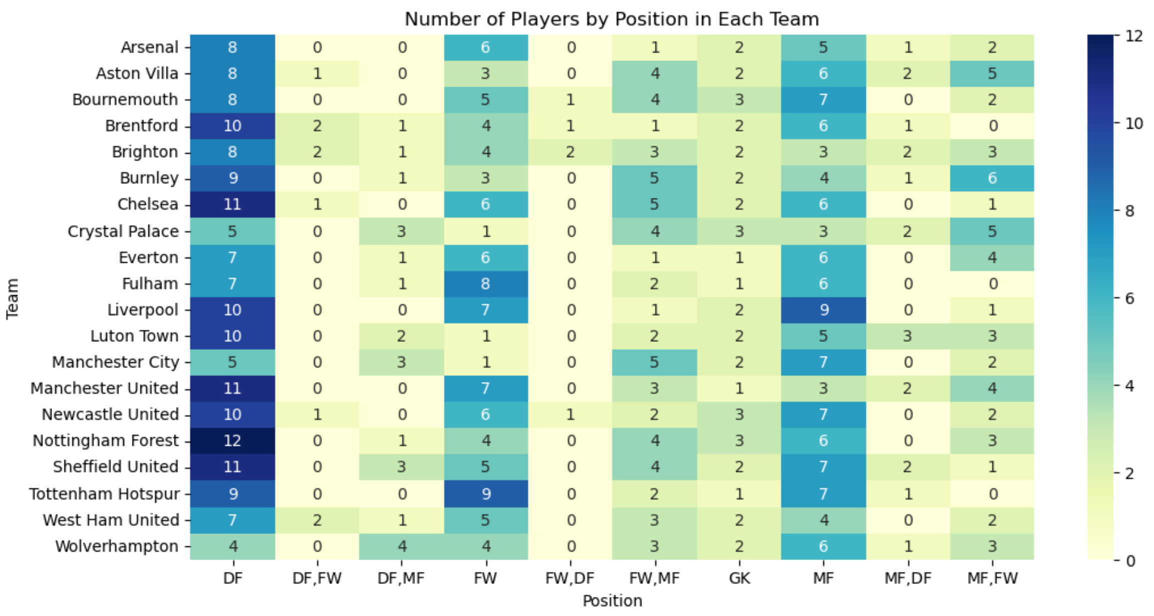
In [485…
```
average_age_by_team = df.groupby('Team')['Age'].mean().round(2).res
average_age_by_team.rename(columns={'Age': 'Average Age'}, inplace=

print(average_age_by_team)

plt.figure(figsize=(10, 6))
ax = sns.barplot(x="Average Age", y="Team", data=average_age_by_tea

for i, v in enumerate(average_age_by_team['Average Age']):
    ax.text(v + 0.1, i, str(v), color='black', va='center')

plt.title("Average Players' Age of Each Team")
plt.xlabel("Average Age")
plt.ylabel("Team")
plt.xticks(rotation=0)
plt.show()
```

```
                  Team  Average Age
0              Arsenal        24.80
1          Aston Villa        25.26
2          Bournemouth        24.63
3            Brentford        25.29
4             Brighton        24.37
5              Burnley        24.29
6              Chelsea        22.03
7       Crystal Palace        25.65
8              Everton        25.62
9               Fulham        27.16
10           Liverpool        23.83
11          Luton Town        25.89
12     Manchester City        25.52
13   Manchester United        24.52
14    Newcastle United        25.34
15   Nottingham Forest        25.27
16    Sheffield United        24.14
17   Tottenham Hotspur        24.17
18     West Ham United        27.00
19       Wolverhampton        24.59
```



In [487…

```
position_count = df.groupby(['Team', 'Pos']).size().unstack(fill_va

plt.figure(figsize=(12, 6))
sns.heatmap(position_count, annot=True, cmap="YlGnBu", fmt='d')
plt.title("Number of Players by Position in Each Team")
plt.xlabel("Position")
plt.ylabel("Team")
plt.show()
```

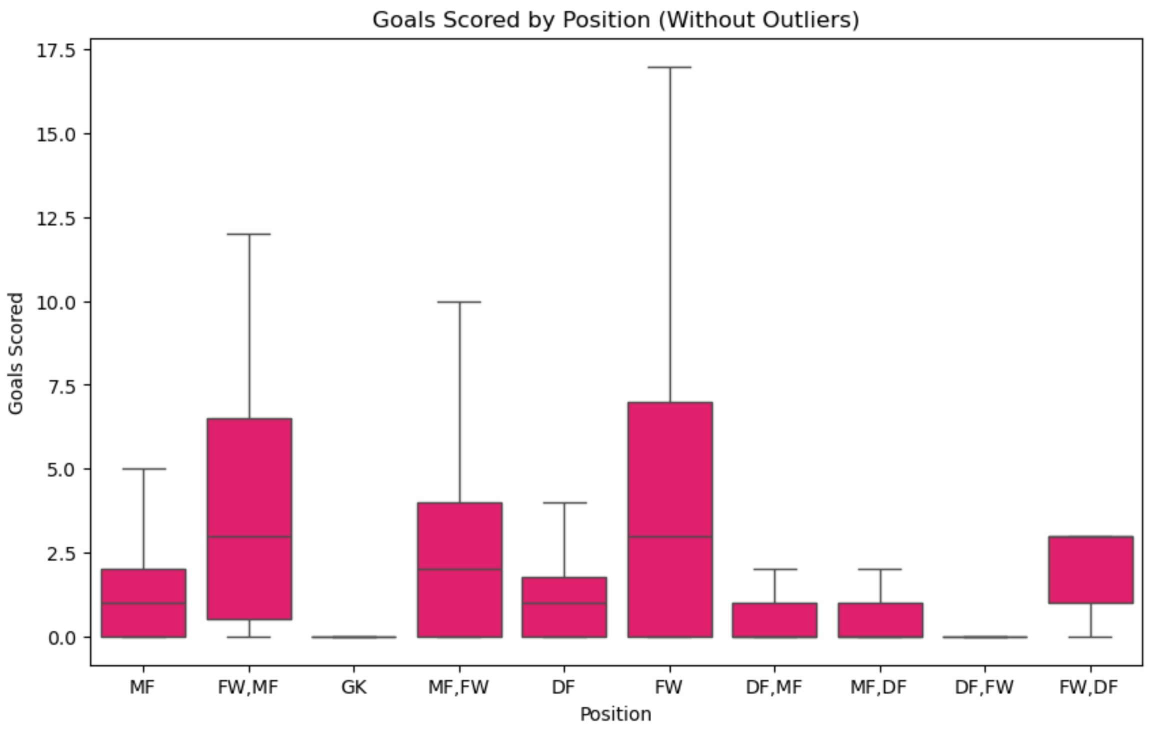


In [ ]:

Boxplot หาค่าผิดปกติ
Boxplot: เปรียบเทียบจำนวนประตู (Gls) ตามตำแหน่ง

In [489…

```python
plt.figure(figsize=(10, 6))
sns.boxplot(x=df["Pos"], y=df["Gls"], color="#FF0065", showfliers=F
plt.title("Goals Scored by Position (Without Outliers)")
plt.xlabel("Position")
plt.ylabel("Goals Scored")
plt.show()
```



In [491…

```python
df['Player'].unique()
```

Out[491…

```
array(['Rodri', 'Phil Foden', 'Ederson', 'Julián Álvarez', 'Kyle W
alker',
       'Bernardo Silva', 'Erling Haaland', 'Rúben Dias', 'Manuel A
kanji',
       'Joško Gvardiol', 'Nathan Aké', 'Jeremy Doku', 'Mateo Kovač
```

```
ić',
       'Kevin De Bruyne', 'John Stones', 'Jack Grealish', 'Rico Le
wis',
       'Matheus Nunes', 'Stefan Ortega', 'Oscar Bobb', 'Kalvin Phi
llips',
       'Sergio Gómez', 'Aymeric Laporte', 'Cole Palmer', 'James Mc
atee',
       'Virgil van Dijk', 'Luis Díaz', 'Alexis Mac Allister',
       'Mohamed Salah', 'Alisson', 'Trent Alexander-Arnold',
       'Dominik Szoboszlai', 'Darwin Núñez', 'Wataru Endo',
       'Andrew Robertson', 'Joe Gomez', 'Cody Gakpo', 'Ibrahima Ko
naté',
       'Curtis Jones', 'Diogo Jota', 'Jarell Quansah', 'Ryan Grave
nberch',
       'Harvey Elliott', 'Caoimhín Kelleher', 'Conor Bradley',
       'Joël Matip', 'Kostas Tsimikas', 'Bobby Clark', 'Stefan Baj
cetic',
       'Ben Doak', 'Jayden Danns', 'Owen Beck', 'Thiago Alcántar
a',
       'James McConnell', 'Kaide Gordon', 'William Saliba', 'Decla
n Rice',
       'Martin Ødegaard', 'Ben White', 'Bukayo Saka', 'Gabriel Mag
alhães',
       'David Raya', 'Kai Havertz', 'Gabriel Martinelli',
       'Oleksandr Zinchenko', 'Leandro Trossard', 'Gabriel Jesus',
       'Jakub Kiwior', 'Takehiro Tomiyasu', 'Eddie Nketiah', 'Jorg
inho',
       'Thomas Partey', 'Aaron Ramsdale', 'Emile Smith Rowe',
       'Fabio Vieira', 'Reiss Nelson', 'Jurriën Timber', 'Cédric S
oares',
       'Mohamed Elneny', 'Ethan Nwaneri', 'Conor Gallagher',
       'Moisés Caicedo', 'Nicolas Jackson', 'Axel Disasi', 'Thiago
Silva',
       'Enzo Fernández', 'Đorđe Petrović', 'Raheem Sterling',
       'Levi Colwill', 'Marc Cucurella', 'Malo Gusto', 'Mykhailo M
udryk',
       'Robert Sánchez', 'Benoît Badiashile', 'Noni Madueke',
       'Trevoh Chalobah', 'Ben Chilwell', 'Armando Broja', 'Reece
James',
       'Lesley Ugochukwu', 'Christopher Nkunku', 'Carney Chukwueme
ka',
       'Alfie Gilchrist', 'Ian Maatsen', 'Cesare Casadei', 'Roméo
Lavia',
       'Deivid Washington', 'Mason Burstow', 'Joshua Acheampong',
       'Alex Matos', 'Jimi Tauriainen', 'Bruno Guimarães', 'Fabian
Schär',
       'Anthony Gordon', 'Dan Burn', 'Sean Longstaff', 'Alexander
Isak',
       'Kieran Trippier', 'Miguel Almirón', 'Martin Dúbravka',
       'Sven Botman', 'Nick Pope', 'Joelinton', 'Lewis Miley',
       'Jacob Murphy', 'Jamaal Lascelles', 'Valentino Livramento',
       'Elliot Anderson', 'Callum Wilson', 'Emil Krafth', 'Lewis H
all',
       'Harvey Barnes', 'Sandro Tonali', 'Joe Willock', 'Loris Kar
ius',
       'Matt Targett', 'Matt Ritchie', 'Ben Parkinson', 'Paul Dumm
```

```
ett',
       'Alex Murphy', 'Joe White', 'Amadou Diallo', 'Michael Ndiwe
ni',
       'Guglielmo Vicario', 'Pedro Porro', 'Son Heung-min',
       'Cristian Romero', 'Dejan Kulusevski', 'Destiny Udogie',
       'Micky van de Ven', 'Pape Matar Sarr', 'James Maddison',
       'Yves Bissouma', 'Brennan Johnson', 'Richarlison',
       'Rodrigo Bentancur', 'Ben Davies', 'Emerson', 'Timo Werne
r',
       'Pierre Højbjerg', 'Oliver Skipp', 'Giovani Lo Celso',
       'Radu Drăgușin', 'Bryan Gil', 'Manor Solomon', 'Eric Dier',
       'Ivan Perišić', 'Davinson Sánchez', 'Alejo Véliz', 'Dane Sc
arlett',
       'Mikey Moore', 'Jamie Donley', 'André Onana', 'Diogo Dalo
t',
       'Bruno Fernandes', 'Alejandro Garnacho', 'Marcus Rashford',
       'Rasmus Højlund', 'Casemiro', 'Kobbie Mainoo', 'Aaron Wan-B
issaka',
       'Scott McTominay', 'Harry Maguire', 'Raphaël Varane',
       'Jonny Evans', 'Antony', 'Victor Lindelöf', 'Christian Erik
sen',
       'Luke Shaw', 'Sofyan Amrabat', 'Lisandro Martínez', 'Mason
Mount',
       'Anthony Martial', 'Sergio Reguilón', 'Amad Diallo',
       'Willy Kambwala', 'Facundo Pellistri', 'Hannibal Mejbri',
       'Omari Forson', 'Jadon Sancho', 'Ethan Wheatley',
       'Donny van de Beek', 'Daniel Gore', 'Ollie Watkins', 'Ezri
Konsa',
       'John McGinn', 'Douglas Luiz', 'Emiliano Martínez', 'Pau To
rres',
       'Lucas Digne', 'Moussa Diaby', 'Matty Cash', 'Leon Bailey',
       'Diego Carlos', 'Boubacar Kamara', 'Youri Tielemans',
       'Clément Lenglet', 'Álex Moreno', 'Nicolò Zaniolo', 'Jacob
Ramsey',
       'Morgan Rogers', 'Robin Olsen', 'Jhon Durán', 'Tim Iroegbun
am',
       'Calum Chambers', 'Leander Dendoncker', 'Tyrone Mings',
       'Omari Kellyman', 'Philippe Coutinho', 'Bertrand Traoré',
       'Cameron Archer', 'Kaine Kesler-Hayden', 'Finley Munroe',
       'Jaden Philogene Bidace', 'Vladimír Coufal', 'Emerson Palmi
eri',
       'Jarrod Bowen', 'James Ward-Prowse', 'Tomáš Souček', 'Kurt
Zouma',
       'Alphonse Areola', 'Lucas Paquetá', 'Edson Álvarez',
       'Mohammed Kudus', 'Nayef Aguerd', 'Michail Antonio',
       'Konstantinos Mavropanos', 'Łukasz Fabiański', 'Angelo Ogbo
nna',
       'Saïd Benrahma', 'Ben Johnson', 'Aaron Cresswell', 'Pablo F
ornals',
       'Danny Ings', 'Maxwel Cornet', 'Divin Mubama', 'George Eart
hy',
       'Thilo Kehrer', 'Kaelan Casey', 'Joachim Andersen',
       'Tyrick Mitchell', 'Jordan Ayew', 'Jefferson Lerma',
       'Jean-Philippe Mateta', 'Eberechi Eze', 'Chris Richards',
       'Marc Guéhi', 'Joel Ward', 'Will Hughes', 'Sam Johnstone',
       'Dean Henderson', 'Odsonne Édouard', 'Jeffrey Schlupp',
       'Daniel Muñoz', 'Adam Wharton', 'Nathaniel Clyne', 'Michael
Olise',
       'Cheick Doucouré', 'Jaïro Riedewald', 'Matheus França',
       'Naouirou Ahamada', 'David Ozoh', 'Jesurun Rak Sakyi',
       'James Tomkins', 'Remi Matthews', 'Bernd Leno', 'Antonee Ro
binson',
       'Andreas Pereira', 'João Palhinha', 'Timothy Castagne',
       'Calvin Bassey', 'Alex Iwobi', 'Willian', 'Tosin Adarabioy
o',
       'Rodrigo Muniz', 'Raúl Jiménez', 'Tim Ream', 'Bobby Reid',
       'Harry Wilson', 'Issa Diop', 'Harrison Reed', 'Tom Cairne
y',
       'Saša Lukić', 'Kenny Tete', 'Carlos Vinícius', 'Adama Traor
é',
       'Luke Harris', 'Fodé Ballo-Touré', 'Aleksandar Mitrović',
       'Jordan Pickford', 'James Tarkowski', 'Jarrad Branthwaite',
       'James Garner', 'Dwight McNeil', 'Abdoulaye Doucouré',
       'Vitaliy Mykolenko', 'Ashley Young', 'Dominic Calvert-Lewi
n',
       'Jack Harrison', 'Idrissa Gana Gueye', 'Amadou Onana',
       'Ben Godfrey', 'Nathan Patterson', 'Beto', 'Séamus Colema
n',
       'Arnaut Danjuma', 'Michael Keane', 'André Gomes', 'Lewis Do
bbin',
       'Youssef Chermiti', 'Neal Maupay', 'Thomas Cannon',
       'Tyler Onyango', 'Lewis Warrington', 'Pascal Groß', 'Lewis
Dunk',
       'Jan Paul van Hecke', 'Simon Adingra', 'Billy Gilmour',
       'Bart Verbruggen', 'Danny Welbeck', 'João Pedro', 'Igor',
       'Joël Veltman', 'Jason Steele', 'Facundo Buonanotte',
       'Kaoru Mitoma', 'Evan Ferguson', 'Carlos Baleba',
       'Pervis Estupiñán', 'Adam Webster', 'Adam Lallana', 'James
Milner',
       'Tariq Lamptey', 'Jack Hinshelwood', 'Solly March', 'Jakub
Moder',
       'Mahmoud Dahoud', 'Julio Enciso', 'Ansu Fati', 'Valentín Ba
rco',
       'Odeluga Offiah', "Mark O'Mahony", 'Benicio Boaitey',
       'Illia Zabarnyi', 'Dominic Solanke', 'Ryan Christie', 'Net
o',
       'Lewis Cook', 'Marcos Senesi', 'Justin Kluivert', 'Adam Smi
th',
       'Marcus Tavernier', 'Antoine Semenyo', 'Milos Kerkez',
       'Lloyd Kelly', 'Philip Billing', 'Max Aarons', 'Dango Ouatt
ara',
       'Alex Scott', 'Luis Sinisterra', 'Chris Mepham', 'Mark Trav
ers',
       'Joe Rothwell', 'Enes Ünal', 'David Brooks', 'Ionuț Radu',
       'Jaidon Anthony', 'Tyler Adams', 'James Hill', 'Kieffer Moo
re',
       'Hamed Junior Traorè', 'Romain Faivre', 'Dominic Sadi',
       'Max Kilman', 'Nélson Semedo', 'José Sá', 'Mario Lemina',
       'João Gomes', 'Toti Gomes', 'Matheus Cunha', 'Rayan Aït-Nou
ri',
       'Craig Dawson', 'Hwang Hee-chan', 'Pablo Sarabia', 'Pedro N
eto',
       'Tommy Doyle', 'Jean-Ricner Bellegarde', 'Matt Doherty',
       'Santiago Bueno', 'Boubacar Traoré', 'Hugo Bueno',
       'Daniel Bentley', 'Fábio Silva', 'Leon Chiwome', 'Nathan Fr
aser',
       'Sasa Kalajdzic', 'Tawanda Chirewa', 'Jonny Castro',
       'Enso Gonzalez', 'Mark Flekken', 'Vitaly Janelt',
       'Christian Nørgaard', 'Nathan Collins', 'Yoane Wissa',
       'Ethan Pinnock', 'Mathias Jensen', 'Mads Roerslev', 'Bryan
Mbeumo',
       'Kristoffer Ajer', 'Ivan Toney', 'Keane Lewis-Potter', 'Ben
Mee',
       'Mathias Jørgensen', 'Frank Onyeka', 'Aaron Hickey',
       'Mikkel Damsgaard', 'Saman Ghoddos', 'Yehor Yarmoliuk',
       'Rico Henry', 'Kevin Schade', 'Shandon Baptiste',
       'Thomas Strakosha', 'Michael Olakigbe', 'Josh Dasilva',
       'Myles Peart-Harris', 'Morgan Gibbs-White', 'Murillo',
       'Anthony Elanga', 'Ryan Yates', 'Callum Hudson-Odoi', 'Chri
s Wood',
       'Danilo', 'Ola Aina', 'Orel Mangala', 'Nicolás Domínguez',
       'Neco Williams', 'Willy Boly', 'Matt Turner', 'Matz Sels',
       'Moussa Niakhate', 'Harry Toffolo', 'Ibrahim Sangaré',
       'Taiwo Awoniyi', 'Serge Aurier', 'Gonzalo Montiel',
       'Andrew Omobamidele', 'Divock Origi', 'Nuno Tavares',
       'Odisseas Vlachodimos', 'Joe Worrall', 'Scott McKenna', 'Fe
lipe',
       'Gio Reyna', 'Cheikhou Kouyaté', 'Rodrigo Ribeiro',
       'Andrey Santos', 'Brandon Aguilera', 'Thomas Kaminski',
       'Alfie Doughty', 'Carlton Morris', 'Ross Barkley', 'Teden M
engi',
       'Gabriel Osho', 'Issa Kaboré', "Amari'i Bell", 'Chiedozie O
gbene',
       'Tahith Chong', 'Reece Burke', 'Elijah Adebayo',
       'Albert Sambi Lokonga', 'Jordan Clark', 'Andros Townsend',
       'Tom Lockyer', 'Marvelous Nakamba', 'Jacob Brown',
       'Pelly Ruddock Mpanzu', 'Daiki Hashioka', 'Ryan John Gile
s',
       'Fred Onyedinma', 'Mads Juel Andersen', 'Cauley Woodrow',
       'Luke Berry', 'Joseph Johnson', 'James Shea', 'Zack Nelso
n',
       'Sander Berge', "Dara O'Shea", 'James Trafford', 'Zeki Amdo
uni',
       'Vitinho', 'Charlie Taylor', 'Wilson Odobert', 'Josh Brownh
ill',
       'Josh Cullen', 'Lyle Foster', 'Jacob Bruun Larsen',
       'Maxime Estève', 'Lorenz Assignon', 'Jordan Beyer',
       'Luca Koleosho', 'Jóhann Berg Guðmundsson', 'Ameen Al-Dakhi
l',
       'Arijanet Muric', 'David Datro Fofana', 'Connor Roberts',
       'Jay Rodriguez', 'Hannes Delcroix', 'Aaron Ramsey',
       'Hjalmar Ekdal', 'Mike Trésor', 'Anass Zaroury', 'Benson Ma
nuel',
       'Nathan Redmond', 'Jack Cork', 'Michael Obafemi',
       'Han-Noah Massengo', 'Gustavo Hamer', 'Jayden Bogle',
       'Jack Robinson', 'Vinicius Souza', 'Anel Ahmedhodžić',
       'Wes Foderingham', 'Auston Trusty', 'Oliver Norwood',
       'Oliver McBurnie', 'Ben Osborn', 'Ben Brereton', 'Andre Bro
```

```
oks',
       'George Baldock', 'Oliver Arblaster', 'Luke Thomas',
       'William Osula', 'Ivo Grbić', 'Mason Holgate', 'Yasser Laro
uci',
       'John Egan', 'Max Lowe', 'Anis Ben Slimane', 'Bénie Adama T
raore',
       'Rhian Brewster', 'Chris Basham', 'Tom Davies',
       'Rhys Norrington-Davies', 'John Fleck', 'Sam Curtis',
       'Daniel Jebbison', 'Antwoine Hackford', 'Sydie Peck', 'Ryan
One'],
      dtype=object)
```

In [ ]:

```
การแสดงผลการทำประตู (Goals) กับ Expected Goals (xG) (Scatter Plot)
การแสดงผลการทำประตู (Gls) และค่าสถิติที่คาดการณ์ (Expected Goals, xG)
```

In [ ]:

In [493…

```
df['Player'] = df['Player'].str.strip().str.lower()
df[df['Player'].str.contains('virgil van dijk')]
```

Out[493…

	Player	Nation	Pos	Age	MP	Starts	Min	90s	Gls	Ast	...	Ast_9
25	virgil van dijk	nl NED	DF	32.0	36	36	3177.0	35.3	2.0	2.0	...	0.0

1 rows × 34 columns

In [495…

```
df['Nation'].unique()
```

Out[495…

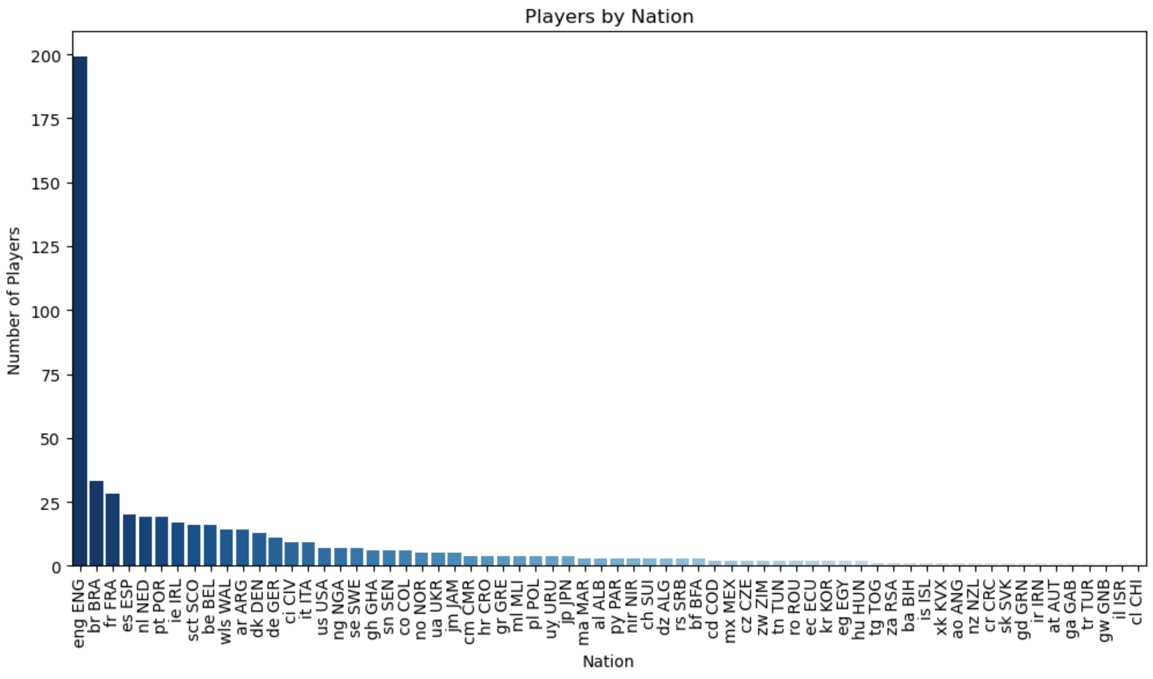
```
array(['es ESP', 'eng ENG', 'br BRA', 'ar ARG', 'pt POR', 'no NO
R',
       'ch SUI', 'hr CRO', 'nl NED', 'be BEL', 'de GER', 'co COL',
       'eg EGY', 'hu HUN', 'uy URU', 'jp JPN', 'sct SCO', 'fr FR
A',
       'ie IRL', 'nir NIR', 'cm CMR', 'gr GRE', 'wls WAL', 'ua UK
R',
       'pl POL', 'it ITA', 'gh GHA', 'ec ECU', 'sn SEN', 'rs SRB',
       'al ALB', 'se SWE', 'py PAR', 'sk SVK', 'kr KOR', 'ml MLI',
       'dk DEN', 'ro ROU', 'il ISR', 'ma MAR', 'ci CIV', 'tn TUN',
       'jm JAM', 'bf BFA', 'cz CZE', 'mx MEX', 'dz ALG', 'us USA',
       'ng NGA', 'gw GNB', 'tr TUR', 'ga GAB', 'at AUT', 'zw ZIM',
       'cd COD', 'ir IRN', 'gd GRN', 'nz NZL', 'cr CRC', 'za RSA',
       'tg TOG', 'is ISL', 'xk KVX', 'ao ANG', 'ba BIH', 'cl CH
I'],
      dtype=object)
```

In [497…

```
Nation_counts = df["Nation"].value_counts()
print("Number of players from each Nation:")
print(Nation_counts)

plt.figure(figsize=(12,6))
sns.barplot(x=Nation_counts.index, y=Nation_counts.values,palette="
plt.xticks(rotation=90)
plt.xlabel("Nation")
plt.ylabel("Number of Players")
plt.title("Players by Nation")
plt.show()
```

```
Number of players from each Nation:
Nation
eng ENG    199
br BRA      33
fr FRA      28
es ESP      20
nl NED      19
          ...
ga GAB       1
tr TUR       1
gw GNB       1
il ISR       1
cl CHI       1
Name: count, Length: 66, dtype: int64
```

```
/var/folders/fg/bz8nn0xj1tj4z45g40xf_hf00000gn/T/ipykernel_79919/409
5907148.py:6: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set `legend
=False` for the same effect.

  sns.barplot(x=Nation_counts.index, y=Nation_counts.values,palette
="Blues_r")
```



In [499…
```
Nation_counts.index
```

Out[499…

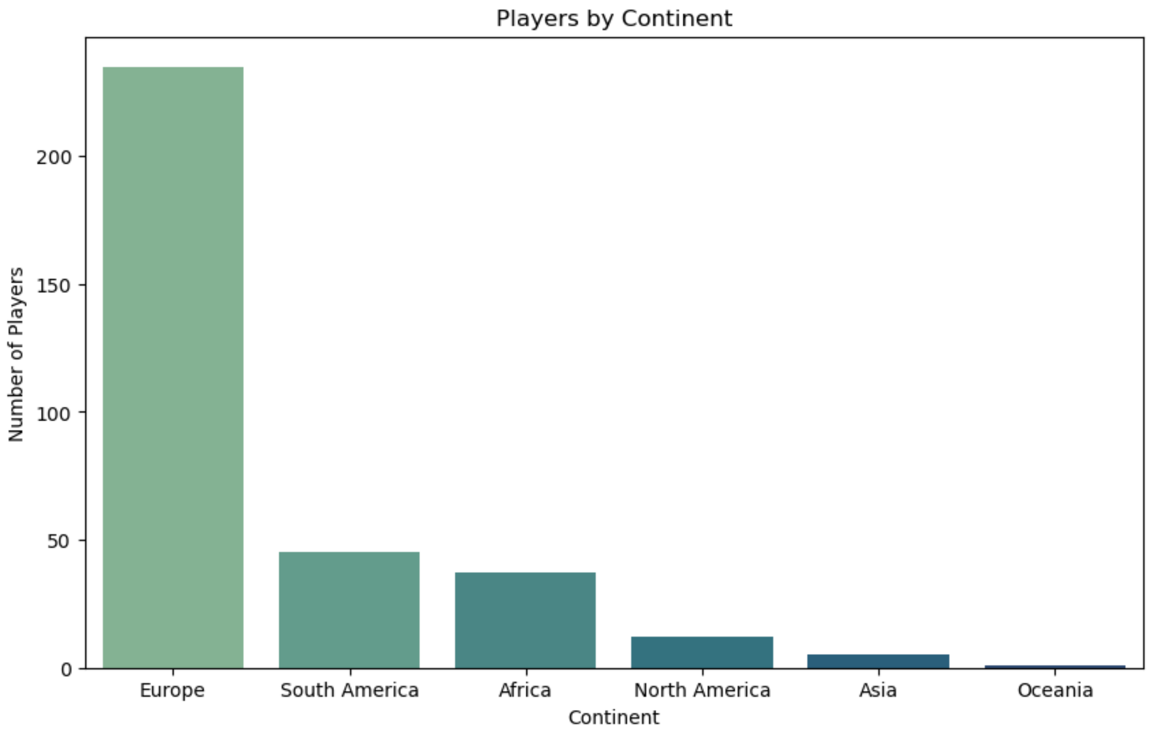
```
Index(['eng ENG', 'br BRA', 'fr FRA', 'es ESP', 'nl NED', 'pt PO
R', 'ie IRL',
       'sct SCO', 'be BEL', 'wls WAL', 'ar ARG', 'dk DEN', 'de GE
R', 'ci CIV',
       'it ITA', 'us USA', 'ng NGA', 'se SWE', 'gh GHA', 'sn SEN',
'co COL',
       'no NOR', 'ua UKR', 'jm JAM', 'cm CMR', 'hr CRO', 'gr GRE',
'ml MLI',
       'pl POL', 'uy URU', 'jp JPN', 'ma MAR', 'al ALB', 'py PAR',
'nir NIR',
       'ch SUI', 'dz ALG', 'rs SRB', 'bf BFA', 'cd COD', 'mx MEX',
'cz CZE',
       'zw ZIM', 'tn TUN', 'ro ROU', 'ec ECU', 'kr KOR', 'eg EGY',
'hu HUN',
       'tg TOG', 'za RSA', 'ba BIH', 'is ISL', 'xk KVX', 'ao ANG',
'nz NZL',
       'cr CRC', 'sk SVK', 'gd GRN', 'ir IRN', 'at AUT', 'ga GAB',
'tr TUR',
       'gw GNB', 'il ISR', 'cl CHI'],
      dtype='object', name='Nation')
```

In [501…

```python
import matplotlib.pyplot as plt
import seaborn as sns

continent_map = {
    # Europe
    "ENG": "Europe", "FRA": "Europe", "ESP": "Europe", "NED": "Euro
    "IRL": "Europe", "SCO": "Europe", "BEL": "Europe", "WAL": "Euro
    "GER": "Europe", "ITA": "Europe", "SWE": "Europe", "UKR": "Euro
    "GRE": "Europe", "ALB": "Europe", "POL": "Europe", "SVK": "Euro
    "SUI": "Europe", "SRB": "Europe", "ROU": "Europe", "CZE": "Euro
    "KVX": "Europe", "BIH": "Europe", "AUT": "Europe", "TUR": "Euro

    # South America
    "BRA": "South America", "ARG": "South America", "COL": "South A
    "URU": "South America", "ECU": "South America", "PAR": "South A
    "CHI": "South America",

    # North America
    "MEX": "North America", "USA": "North America", "CRC": "North A
    "JAM": "North America", "GRN": "North America",

    # Africa
    "GHA": "Africa", "SEN": "Africa", "CIV": "Africa", "CMR": "Afri
    "BFA": "Africa", "ALG": "Africa", "NGA": "Africa", "GNB": "Afri
    "ZIM": "Africa", "COD": "Africa", "RSA": "Africa", "TOG": "Afri
    "EGY": "Africa", "MAR": "Africa", "TUN": "Africa", "DZA": "Afri

    # Asia
    "JPN": "Asia", "KOR": "Asia", "IRN": "Asia", "ISR": "Asia",

    # Oceania
    "NZL": "Oceania"
}

def extract_country_code(nation_str):
    return nation_str.split()[-1]
df["Country_Code"] = df["Nation"].apply(extract_country_code)

df["Continent"] = df["Country_Code"].map(continent_map)
df["Continent"].value_counts()
print("Number of players by Continent:")
print(continent_counts)
plt.figure(figsize=(10, 6))
sns.barplot(x=continent_counts.index, y=continent_counts.values,pal
plt.xlabel("Continent")
plt.ylabel("Number of Players")
plt.title("Players by Continent")
plt.show()
```

```
Number of players by Continent:
Continent
Europe           235
South America     45
Africa            37
North America     12
Asia               5
Oceania            1
Name: count, dtype: int64
```

```
/var/folders/fg/bz8nn0xj1tj4z45g40xf_hf00000gn/T/ipykernel_79919/218
4331400.py:49: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set `legend
=False` for the same effect.

  sns.barplot(x=continent_counts.index, y=continent_counts.values,pa
lette="crest")
```

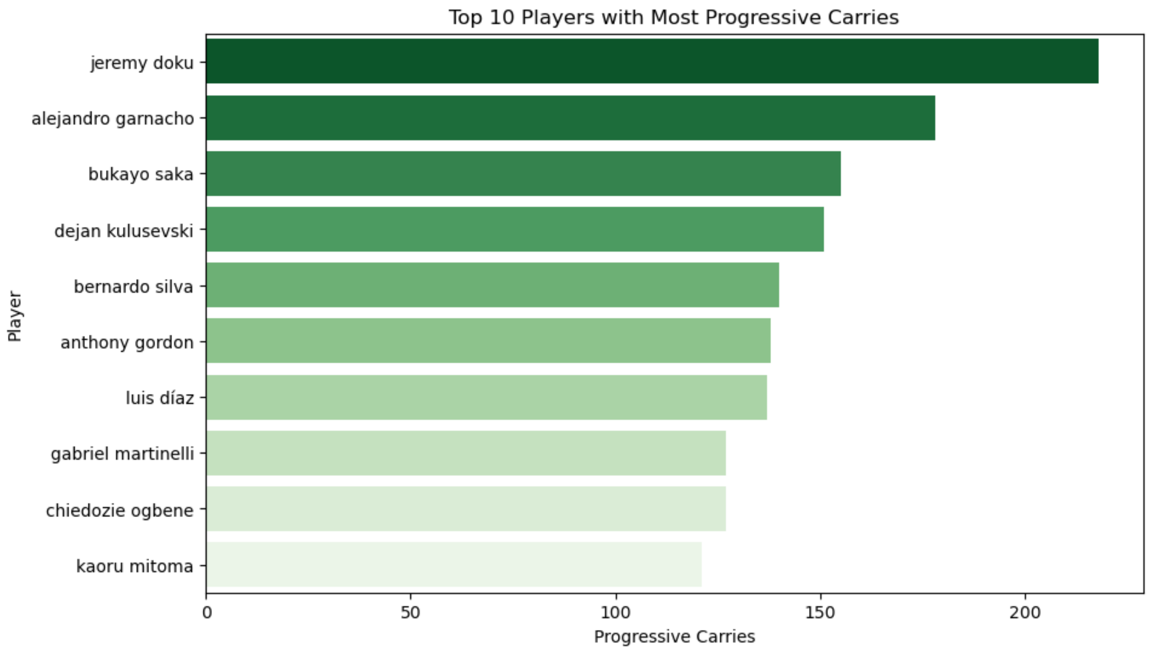


In [505…

```
top_10_progressive_carries = df[['Player', 'Team', 'PrgC']].sort_va
```

```
plt.figure(figsize=(10, 6))
sns.barplot(x='PrgC', y='Player', data=top_10_progressive_carries,
plt.title('Top 10 Players with Most Progressive Carries')
plt.xlabel('Progressive Carries')
plt.ylabel('Player')
plt.show()
```

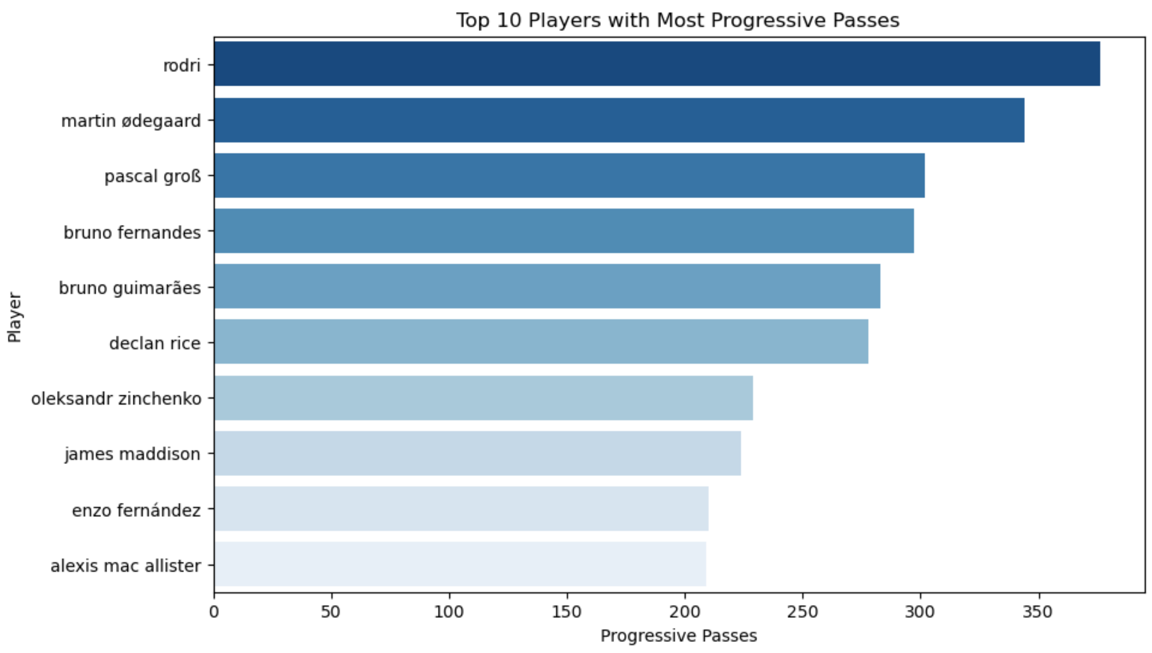
```
/var/folders/fg/bz8nn0xj1tj4z45g40xf_hf00000gn/T/ipykernel_79919/169
1219761.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `y` variable to `hue` and set `legend
=False` for the same effect.

  sns.barplot(x='PrgC', y='Player', data=top_10_progressive_carries,
palette='Greens_r')
```



In [507…

```
top_10_progressive_passers = df[['Player', 'Team', 'PrgP']].sort_va

plt.figure(figsize=(10, 6))
sns.barplot(x='PrgP', y='Player', data=top_10_progressive_passers,
plt.title('Top 10 Players with Most Progressive Passes')
plt.xlabel('Progressive Passes')
plt.ylabel('Player')
plt.show()
```

```
/var/folders/fg/bz8nn0xj1tj4z45g40xf_hf00000gn/T/ipykernel_79919/218
053844.py:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `y` variable to `hue` and set `legend
=False` for the same effect.

  sns.barplot(x='PrgP', y='Player', data=top_10_progressive_passers,
palette='Blues_r')
```



In [509…

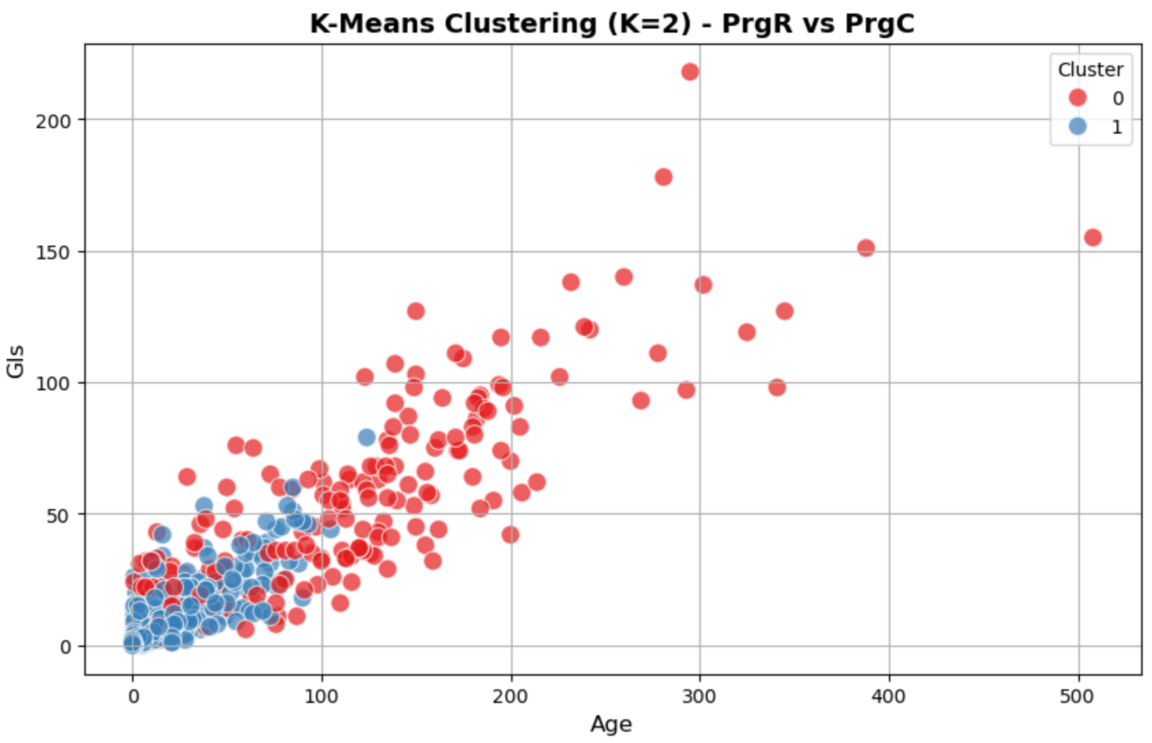
```
df.columns
```

Out[509…

```
Index(['Player', 'Nation', 'Pos', 'Age', 'MP', 'Starts', 'Min', '9
0s', 'Gls',
       'Ast', 'G+A', 'G-PK', 'PK', 'PKatt', 'CrdY', 'CrdR', 'xG',
'npxG',
       'xAG', 'npxG+xAG', 'PrgC', 'PrgP', 'PrgR', 'Gls_90', 'Ast_9
0', 'G+A_90',
       'G-PK_90', 'G+A-PK_90', 'xG_90', 'xAG_90', 'xG+xAG_90', 'np
xG_90',
       'npxG+xAG_90', 'Team', 'Country_Code', 'Continent'],
      dtype='object')
```

In [571…

```
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
import seaborn as sns

features = ['Age', 'MP', 'Starts', 'Min', '90s', 'Gls', 'Ast',
                    'PK', 'PKatt', 'CrdY', 'CrdR', 'xG', 'npxG', 'x
                    'PrgC', 'PrgP', 'PrgR', 'Gls_90', 'Ast_90']
df_cluster = df.dropna(subset=features)

scaler = StandardScaler()
df_scaled = scaler.fit_transform(df_cluster[features])

kmeans = KMeans(n_clusters=2, random_state=0, n_init=10)
df_cluster['Cluster'] = kmeans.fit_predict(df_scaled)

plt.figure(figsize=(10, 6))
sns.scatterplot(x=df_cluster["PrgR"], y=df_cluster["PrgC"], hue=df_
                palette="Set1", alpha=0.7, s=100)

plt.xlabel("Age", fontsize=12)
plt.ylabel("Gls", fontsize=12)
plt.title("K-Means Clustering (K=2) - PrgR vs PrgC", fontsize=14, f
plt.legend(title="Cluster")
plt.grid(True)
plt.show()
```



In [ ]:

```
1.	Cluster 0 (สีแดง):
•		ผู้เล่นที่มีจำนวน Progressive Runs (PrgR) และ Progressive
•		อาจเป็นกลุ่มผู้เล่นที่มีสไตล์ชอบครองบอลและพาบอลไปข้างหน้าเป็นหลัก
•		มักเป็นนักเตะในตำแหน่ง ปีก (Winger) หรือ กองกลางตัวรุก (Atta
2.	Cluster 1 (สีฟ้า):
•		ผู้เล่นที่มีจำนวน Progressive Runs (PrgR) และ Progressive
•		อาจเป็นผู้เล่นแนวรับที่เน้นการจ่ายบอลสั้น ๆ หรือนักเตะที่มีบทบาทไม่สูง
•		มักเป็นนักเตะในตำแหน่ง กองหลัง (Defender) หรือ กองกลางตัวรับ
```

In [511…

```
df_cluster.head()
```

Out[511…

	Player	Nation	Pos	Age	MP	Starts	Min	90s	Gls	Ast	...	G
0	Rodri	es ESP	MF	27.0	34	34	2931.0	32.6	8.0	9.0	...	
1	Phil Foden	eng ENG	FW,MF	23.0	35	33	2857.0	31.7	19.0	8.0	...	
2	Ederson	br BRA	GK	29.0	33	33	2785.0	30.9	0.0	0.0	...	
3	Julián Álvarez	ar ARG	MF,FW	23.0	36	31	2647.0	29.4	11.0	8.0	...	
4	Kyle Walker	eng ENG	DF	33.0	32	30	2767.0	30.7	0.0	4.0	...	

5 rows × 35 columns

In [535…

```
cluster_summary = df_cluster.groupby('Cluster')[features].mean()
```

```
cluster_summary['count'] = df_cluster['Cluster'].value_counts()
cluster_summary
```

Out[535…

	Age	MP	Starts	Min	90s	G
Cluster						
0	25.954751	29.203620	24.063348	2122.529412	23.582353	2.2624
1	24.066225	10.115894	4.894040	476.625828	5.296026	0.3576
2	25.298246	32.894737	27.438596	2418.228070	26.864912	10.3333

In [564…
```
df_cluster['PrgR'].max()
```

Out[564…
```
508.0
```

In [ ]:
```
```

In [560…
```
df_cluster[(df_cluster['PrgC']>=200)&(df_cluster['Age']<=23)]
```

Out[560…

	Player	Nation	Pos	Age	MP	Starts	Min	90s	Gls	Ast	…	G+ PK_
11	jeremy doku	be BEL	FW,MF	21.0	29	18	1595.0	17.7	3.0	8.0	…	0

1 rows × 37 columns

In [566…
```
df_cluster[(df_cluster['PrgR']>=400)&(df_cluster['Age']<=23)]
```

Out[566…

	Player	Nation	Pos	Age	MP	Starts	Min	90s	Gls	Ast	…	G+ PK_
59	bukayo saka	eng ENG	FW	21.0	35	35	2919.0	32.4	16.0	9.0	…	0.

1 rows × 37 columns

In [517…
```
df.head()
```

Out[517…

	Player	Nation	Pos	Age	MP	Starts	Min	90s	Gls	Ast	...	Pl
0	rodri	es ESP	MF	27.0	34	34	2931.0	32.6	8.0	9.0	...	
1	phil foden	eng ENG	FW,MF	23.0	35	33	2857.0	31.7	19.0	8.0	...	
2	ederson	br BRA	GK	29.0	33	33	2785.0	30.9	0.0	0.0	...	
3	julián álvarez	ar ARG	MF,FW	23.0	36	31	2647.0	29.4	11.0	8.0	...	
4	kyle walker	eng ENG	DF	33.0	32	30	2767.0	30.7	0.0	4.0	...	

5 rows × 36 columns

In [521…

```
# แยกตำแหน่งออกเป็นหลายคอลัมน์และใส่ค่า 1/0
position_dummies = df['Pos'].str.get_dummies(sep='/')

# รวมเข้ากับ DataFrame เดิม
df_cluster = pd.concat([df_cluster, position_dummies], axis=1)
```

In [523…

```
df_cluster.head(5)
df_cluster['Min'].max
```

Out[523…

```
<bound method Series.max of 0      2931.0
1      2857.0
2      2785.0
3      2647.0
4      2767.0
        ...
575      28.0
576      21.0
577      13.0
578      10.0
579       8.0
Name: Min, Length: 580, dtype: float64>
```

In [573…

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report,
import matplotlib.pyplot as plt

df['Transfer_Chance'] = ((df['G+A_90'] > 0.5) & (df['Min'] < 1000))

features = df[['Age', 'Min', 'G+A_90', 'xG_90', 'xAG_90']]
target = df['Transfer_Chance']

X_train, X_test, y_train, y_test = train_test_split(features, targe

model = RandomForestClassifier(n_estimators=100, random_state=42)
```

```
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))

mae = mean_absolute_error(y_test, y_pred)
print(f"Mean Absolute Error (MAE): {mae:.4f}")

df['Transfer_Probability'] = model.predict_proba(features)[:, 1]

top_5_transfers = df.sort_values(by='Transfer_Probability', ascendi

print(top_5_transfers[['Player', 'Age', 'Min', 'G+A_90', 'xG_90', '
```

```
Accuracy: 1.0
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       168
           1       1.00      1.00      1.00         6

    accuracy                           1.00       174
   macro avg       1.00      1.00      1.00       174
weighted avg       1.00      1.00      1.00       174

Mean Absolute Error (MAE): 0.0000
               Player   Age    Min  G+A_90  xG_90  xAG_90  \
307      adama traoré  27.0  377.0    1.19   0.36    0.17
447       kevin schade  21.0  333.0    0.81   0.32    0.19
223        jhon durán  19.0  475.0    0.95   0.38    0.06
388         enes ünal  26.0  328.0    1.10   0.80    0.28
448  shandon baptiste  25.0  229.0    0.79   0.35    0.08

     Transfer_Probability
307                  0.99
447                  0.98
223                  0.98
388                  0.97
448                  0.96
```

In [587…

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split, cross_val_sco
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score, make_scor
import matplotlib.pyplot as plt
import seaborn as sns

features = ['Age', 'MP', 'xG', 'npxG', 'PrgP', 'PrgC']
X = df[features]
y = df['Gls']

player_names = df['Player']

X_train, X_test, y_train, y_test, player_train, player_test = train
    X, y, player_names, test_size=0.2, random_state=42)
```

```
model = LinearRegression()
model.fit(X_train, y_train)

kf = KFold(n_splits=5, shuffle=True, random_state=42)
mse_scorer = make_scorer(mean_squared_error)
cv_scores = cross_val_score(model, X, y, cv=kf, scoring=mse_scorer)

print(f'Cross Validation Scores (MSE): {cv_scores}')
print(f'Average MSE: {np.mean(cv_scores):.2f}')
print(f'Root Mean Squared Error (RMSE): {np.sqrt(np.mean(cv_scores)

y_pred = model.predict(X_test)

print(f'Mean Squared Error (MSE): {mean_squared_error(y_test, y_pre
print(f'Root Mean Squared Error (RMSE): {np.sqrt(mean_squared_error
print(f'R-squared (R²): {r2_score(y_test, y_pred):.2f}')

comparison = pd.DataFrame({
    'Player': player_test,
    'Actual': y_test,
    'Predicted': y_pred
}).reset_index(drop=True)

comparison['Error'] = comparison['Actual'] - comparison['Predicted'
print(comparison[['Player', 'Actual', 'Predicted', 'Error']].head(1
```

```
Cross Validation Scores (MSE): [1.35445256 1.57103467 2.31105668 1.3
3356356 1.78656411]
Average MSE: 1.67
Root Mean Squared Error (RMSE): 1.29
Mean Squared Error (MSE): 1.35
Root Mean Squared Error (RMSE): 1.16
R-squared (R²): 0.93
             Player  Actual  Predicted     Error
0        sam curtis     0.0  -0.072453  0.072453
1   nathaniel clyne     0.0   0.106243 -0.106243
2        ian maatsen     0.0   0.694573 -0.694573
3   marcus rashford     7.0   7.287379 -0.287379
4        ryan yates     1.0   2.560460 -1.560460
5        bernd leno     0.0  -0.374227  0.374227
6     luca koleosho     1.0   2.094824 -1.094824
7  mikkel damsgaard     0.0   0.914575 -0.914575
8       ivan perišić     0.0   0.042549 -0.042549
9  łukasz fabiański     0.0  -0.192322  0.192322
```

In [601…

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split, cross_val_sco
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report,
import matplotlib.pyplot as plt
import seaborn as sns

df['Overperformer'] = (df['Gls'] > df['xG']).astype(int)
features = ['Age', 'MP', 'npxG', 'PrgP', 'PrgC', 'Gls_90', 'Ast_90'
```

```
X = df[features]
y = df['Overperformer']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size

model = LogisticRegression(max_iter=1000, random_state=42)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
print(f'Accuracy: {accuracy_score(y_test, y_pred):.2%}')
print(classification_report(y_test, y_pred))

df['Predicted_Overperformer'] = model.predict(X)
overperformers = df[df['Predicted_Overperformer'] == 1]

print("Top 10 Overperformers:")
print(overperformers[['Player', 'Gls', 'xG']].sort_values(by='Gls',
```

```
Accuracy: 87.07%
              precision    recall  f1-score   support

           0       0.87      0.97      0.92        86
           1       0.86      0.60      0.71        30

    accuracy                           0.87       116
   macro avg       0.87      0.78      0.81       116
weighted avg       0.87      0.87      0.86       116

Top 10 Overperformers:
                     Player   Gls    xG
83              cole palmer  22.0  18.2
117         alexander isak  21.0  20.3
1               phil foden  19.0  10.3
28          mohamed salah  18.0  21.2
146          son heung-min  17.0  12.0
265  jean-philippe mateta  16.0  10.9
59              bukayo saka  16.0  15.5
458              chris wood  14.0  11.9
62              kai havertz  13.0  12.3
404          matheus cunha  12.0   9.5
```

```
In [ ]:
```