



**POLITECHNIKA KRAKOWSKA im. T. Kościuszki**  
Wydział Mechaniczny  
**Katedra Informatyki Stosowanej**



Kierunek studiów: Informatyka stosowana  
Specjalność: -

STUDIA STACJONARNE

# **PRACA DYPLOMOWA**

INŻYNIERSKA

**Ireneusz Kasprzak**

Opracowanie w Pythonie modelu uczenia maszynowego oceniającego  
jakość potencjalnych transferów piłkarskich

Developing a machine learning model in Python to evaluate the quality  
of potential football transfers

Promotor:  
mgr inż. **Adam Piwowarczyk**

Kraków, rok akad. 2024/2025





\*) – niepotrzebne skreślić

## OŚWIADCZENIE O SAMODZIELNYM WYKONANIU PRACY DYPLOMOWEJ

Oświadczam, że przedkładana przeze mnie praca dyplomowa inżynierska została napisana przeze mnie samodzielnie. Jednocześnie oświadczam, że ww. praca:

- 1) nie narusza praw autorskich w rozumieniu ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (Dz.U. z 2021 r. poz. 1062) oraz dóbr osobistych chronionych prawem cywilnym, a także nie zawiera danych i informacji, które uzyskałem/am\* w sposób niedozwolony,
- 2) nie była wcześniej podstawą żadnej innej procedury związanej z nadawaniem tytułów zawodowych, stopni lub tytułów naukowych.

**Jednocześnie wyrażam zgodę na:**

- 1) poddanie mojej pracy kontroli za pomocą systemu Antyplagiat oraz na umieszczenie tekstu pracy w bazie danych uczelni, w celu ochrony go przed nieuprawnionym wykorzystaniem. Oświadczam, że zostałem/am\* poinformowany/a\* i wyrażam zgodę, by system Antyplagiat porównywał tekst mojej pracy z tekstem innych prac znajdujących się w bazie danych uczelni, z tekstami dostępnymi w zasobach światowego Internetu oraz z bazą porównawczą systemu Antyplagiat,
- 2) to, aby moja praca pozostała w bazie danych uczelni przez okres wynikający z przepisów prawa. Oświadczam, że zostałem poinformowany i wyrażam zgodę, że tekst mojej pracy stanie się elementem porównawczej bazy danych uczelni, która będzie wykorzystywana, w tym także udostępniana innym podmiotom, na zasadach określonych przez uczelnię, w celu dokonywania kontroli antyplagiatowej prac dyplomowych/doktorskich, a także innych tekstów, które powstaną w przyszłości.

.....  
podpis

- 1) Wyrażam zgodę na udostępnianie mojej pracy dyplomowej w Akademickim Systemie Archiwizacji Prac na PK do celów naukowo-badawczych z poszanowaniem przepisów ustawy o prawie autorskim i prawach pokrewnych (Dz.U. z 2021 r. poz. 1062)..

TAK/NIE\*

.....  
podpis

*Jednocześnie przyjmuję do wiadomości, że w przypadku stwierdzenia popełnienia przeze mnie czynu polegającego na przypisaniu sobie autorstwa istotnego fragmentu lub innych elementów cudzej pracy, lub ustalenia naukowego, Rektor PK stwierdzi nieważność postępowania w sprawie nadania mi tytułu zawodowego (art. 77 ust. 5 ustawy z dnia 18 lipca 2018 r. Prawo o szkolnictwie wyższym i nauce, (Dz.U. z 2021 r., poz. 478, z późn. zm.)).*

.....  
podpis



# SPIS TREŚCI

<b>PRACA DYPLOMOWA</b>	<b>1</b>
<b>SPIS TREŚCI</b>	<b>6</b>
<b>1. CEL I ZAKRES PRACY</b>	<b>8</b>
<b>2. WSTĘP</b>	<b>9</b>
<b>3. OPIS DANYCH I NARZĘDZI</b>	<b>11</b>
3.1. Opis danych	11
3.2. Statystyki zawodników	11
3.3. Statystyki zespołów	16
3.4. Obliczanie metryk	18
3.5. Opis narzędzi	21
<b>4. PROJEKT I IMPLEMENTACJA APLIKACJI</b>	<b>32</b>
4.1. Wyszukiwanie zawodników	34
4.2. Statystyki	37
4.3. Styl gry drużyny	41
4.4. Przewidywanie statystyk	45
4.5. Ocena dopasowania do zespołu	47
<b>5. TRENOWANIE MODELI</b>	<b>51</b>
<b>6. WYNIKI I OGRANICZENIA PROJEKTU</b>	<b>54</b>
6.1. Wyniki modeli predykcyjnych	54
6.2. Porównanie predykcji do rzeczywistych statystyk	55
6.3. Ograniczenia projektu	59
<b>7. WNIOSKI</b>	<b>60</b>
<b>8. LITERATURA</b>	<b>62</b>
<b>9. SUMMARY</b>	<b>63</b>



## 1. Cel i zakres pracy

Celem niniejszej pracy inżynierskiej jest opracowanie i implementacja modelu uczenia maszynowego w języku Python, który wspiera ocenę jakości potencjalnych transferów piłkarskich. Projekt ma wspierać decyzje związane z transferami poprzez analizę danych o zawodnikach, ich statystyk oraz stylu gry drużyn. Wprowadzenie takiego narzędzia może wpłynąć na proces decyzyjny w zarządzaniu sportowym, pozwalając na bardziej świadome i trafne decyzje związane z transferami zawodników.

Praca składa się z dwóch głównych części: teoretycznej i praktycznej. W części teoretycznej zostaną omówione następujące zagadnienia:

- Narzędzia i biblioteki wykorzystane w projekcie.
- Podstawy uczenia maszynowego.
- Charakterystyka wykorzystanych danych piłkarskich i ich rola w pracy.
- Zastosowanie wizualizacji danych w kontekście analizy sportowej.

W części praktycznej zostanie zaprezentowana aplikacja podzielona na pięć funkcjonalnych zakładerek:

- Wyszukiwanie: Umożliwia przeszukiwanie bazy danych piłkarzy według wybranych kryteriów, takich jak: narodowość, pozycja, liga, sezon czy minimalne wartości kluczowych statystyk.
- Statystyki: Umożliwia wybór konkretnego zawodnika i sezonu, prezentując kluczowe statystyki gracza oraz obliczone na ich podstawie metryki. Dodatkowo generowany jest wykres radarowy, który porównuje wybranego zawodnika do przeciętnego gracza na tej samej pozycji w tej samej lidze i sezonie.
- Styl gry zespołu: Pozwala na analizę stylu gry drużyny w wybranym sezonie. Poprzez interaktywny wykres suwakowy użytkownik może ocenić takie aspekty jak bezpośredniość gry, odległość oddawania strzałów czy posiadanie piłki. Dane te są porównywane do innych zespołów z top 5 lig europejskich w wybranym sezonie.
- Przewidywanie statystyk: W tej zakładce można wykorzystać zaimplementowane modele predykcyjne, które na podstawie historycznych danych będą przewidywały przyszłe statystyki zawodników. Statystyki zostaną zaprezentowane w formie wykresu radarowego, co ułatwi ich wizualną analizę.
- Ocena dopasowania do zespołu: Umożliwia ocenę potencjalnych transferów. Po wybraniu zawodnika oraz drużyny docelowej aplikacja przeanalizuje styl gry dotychczasowej drużyny w jakiej grał dany piłkarz oraz styl gry drużyny docelowej. Na tej podstawie zostanie wystawiona ocena dopasowania zawodnika do zespołu, a także przewidziane jego statystyki w nowym klubie. Statystyki zostaną wyświetlone na wykresie radarowym.



## 2. Wstęp

Piłka nożna, jako jedna z najpopularniejszych dyscyplin sportowych na świecie generuje ogromne zainteresowanie zarówno ze strony kibiców, jak i osób związanych zawodowo z tym sportem. Każdy sezon przynosi nowe, mniej lub bardziej zaskakujące transfery zawodników, które mogą zmieniać układ sił w rozgrywkach ligowych oraz pucharowych. Decyzje dotyczące transferów opierają się jednak na wielu czynnikach, takich jak potencjał zawodnika, styl gry zespołu czy strategia klubu. W dobie rosnącej dostępności zaawansowanych danych piłkarskich oraz rozwoju technologii uczenia maszynowego pojawia się możliwość optymalizacji procesu zakupu piłkarzy poprzez wykorzystywanie metod analitycznych. Współczesna analiza piłkarska opiera się nie tylko na tradycyjnym skautingu i doświadczeniu, ale również na liczbach, modelach predykcyjnych i wizualizacji danych.

Zastosowanie analizy danych w sporcie nie jest nowością. W Stanach Zjednoczonych w latach 70. rozpoczęto wykorzystywanie zaawansowanych statystyk w baseballu. W książce „Moneyball” [1] opisano strategię Oakland Athletics i ich menedżera Billy’ego Beane’a, którzy dzięki analityce statystycznej stworzyli konkurencyjną drużynę mimo ograniczonego budżetu. Podobne podejście stosuje się w NBA, gdzie analiza danych jest wykorzystywana od dekad. Przykłady takie jak Los Angeles Dodgers w baseballu czy Golden State Warriors w NBA pokazują, że inwestowanie w dane i technologie przynosi wymierne efekty na boisku.

W piłce nożnej rozwój analityki postępuje z opóźnieniem w porównaniu do amerykańskich dyscyplin, ale w ostatnich latach stał się nieodzownym elementem zarządzania w klubach takich jak Liverpool, Brentford czy Midtjylland. Liverpool korzysta z zaawansowanych modeli statystycznych, aby optymalizować decyzje transferowe i taktyczne. Kluby z niższymi budżetami, jak Brentford, także wykorzystują statystyki, aby pozyskiwać niedocenianych zawodników o dużym potencjale rozwoju. FC Midtjylland, z kolei wykorzystało analizę danych do zrozumienia gry w stałych fragmentach, co pozwoliło mu odnieść sukces na krajowej scenie piłkarskiej. W artykule The Guardian [2] piłkarz duńskiego zespołu mówi: „Analizowaliśmy tysiące rzutów wolnych i różnych, przekształcając je w zbiór strategii (playbook) stałych fragmentów gry złożony z 20-25 schematów. Inne kluby też to robią, ale nasze dane pokazują, że 49% wszystkich naszych bramek w zeszłym sezonie padło właśnie po stałych fragmentach.” Ta wypowiedź pokazuje, że inwestowanie w analizę danych przynosi wymierne korzyści, przekładając się na poprawę wyników sportowych.

Ostatnio głośno jest także o City Football Group, międzynarodowej organizacji łączącej kluby piłkarskie z całego świata, które wspólnie korzystają z zaawansowanych narzędzi analitycznych. Dzięki tej współpracy możliwe jest również, aby większe kluby w tej grupie wykorzystywały mniej prestiżowe drużyny do ogrywania młodych piłkarzy o wysokim potencjale, którzy jeszcze nie są w stanie konkurować na najwyższym poziomie. Przykładem może być Savinho, który został wypożyczony do Girony

z francuskiego Troyes, by zdobywać doświadczenie. Kiedy okazał się wartościowym zawodnikiem w hiszpańskiej LaLiga, po sezonie trafił do największego klubu w organizacji, czyli Manchesteru City. Taki model pozwala na efektywne zarządzanie talentami, przy jednoczesnym minimalizowaniu ryzyka inwestycyjnego.

Jednym z najbardziej wymagających obszarów analizy sportowej jest właśnie ocena transferów, która wymaga uwzględnienia zarówno danych historycznych zawodników, jak i charakterystyki drużyn, do której mają trafić. Współcześnie fundamentem sukcesu wielu gigantów piłkarskich są dobrze rozwinięte działy analityczne, jednak wciąż brakuje powszechnie dostępnych narzędzi, które wspierałyby decyzje transferowe i taktyczne, szczególnie dla mniejszych klubów z niskim budżetem. Niniejsza praca inżynierska koncentruje się na połączeniu zaawansowanych metod analizy danych z intuicyjnym interfejsem użytkownika. Kluczowym elementem projektu jest zgromadzenie odpowiednich danych oraz opracowanie narzędzia, które umożliwi użytkownikowi eksplorację statystyk piłkarskich, analizę osiągnięć zawodników i tworzenie wizualizacji wspierających ocenę transferów. Aplikacja ma wyróżniać się połączeniem różnych funkcjonalności, które umożliwią uzyskanie pełnego obrazu gry zawodnika oraz zespołu, a także zrozumienie ryzyka i korzyści wynikających z potencjalnego zakupu.

Projekt ten odpowiada na aktualne potrzeby rynku, jednocześnie przyczyniając się do rozwoju nowoczesnych metod zarządzania w sporcie i podnoszenia jakości podejmowanych decyzji.

### 3. Opis danych i narzędzi

Rozdział ten przedstawia szczegółowe informacje na temat danych oraz narzędzi wykorzystanych w projekcie. Opisano źródła danych statystycznych, proces ich przetwarzania i integracji, a także narzędzia programistyczne użyte do analizy i implementacji aplikacji.

#### 3.1. Opis danych

Dane wykorzystane do projektu pochodzą z różnych stron gromadzących statystyczne dane piłkarskie takich jak Football Reference (fbref.com), czy WhoScored (whoscored.com). Dane statystyczne zawodników zostały pobrane z platformy Kaggle (pliki z danymi z fbref [3] [4]), a dane dotyczące zespołów zostały zebrane ręcznie. Dane zawierają statystyki top 5 lig europejskich. Po oczyszczeniu, połączeniu danych z dwóch sezonów oraz obliczeniu dodatkowych kolumn plik ze statystykami zawodników (merged\_players.csv) zawiera 2874 wiersze oraz 122 kolumny. Dane odnoszą się do sezonów 2021/2022 oraz 2022/2023. Natomiast plik zawierający dane o zespołach zawiera ostatecznie 292 wiersze, 50 kolumn i zawiera dane o sezonach 2021/2022, 2022/2023 oraz 2023/2024. Poniżej znajduje się objaśnienie danych wykorzystanych do budowy aplikacji.

#### 3.2. Statystyki zawodników

W tej części pracy przedstawione zostały wszystkie dane dotyczące piłkarzy, które zostały wykorzystane do budowy aplikacji. Pozwalają one na szczegółową analizę gry zawodników i zawierają zarówno podstawowe informacje dotyczące zawodników, jak i szczegółowe dane związane z ich występami na boisku.

Poniższa tabela zawiera nazwy i objaśnienie kolumn wykorzystywanych w projekcie. Każda z tych danych ma na celu dostarczenie pełniejszego obrazu na temat zawodników i ich stylu gry.

*Tabela 3.1. – dane o zawodnikach*

Nazwa kolumny	Opis
name	imię i nazwisko zawodnika
Nation	narodowość
Pos	pozycja
Squad	drużyna
Comp	liga
Age	wiek
foot	noga wiodąca

height_in_cm	wzrost (w cm)
Born	rok urodzenia
MP	rozegrane mecze
Starts	mecze w wyjściowym składzie
Min	rozegrane minuty
90s	rozegrane 90 min
Goals	gole/mecz
Shots	strzały/mecz
SoT	strzały na bramkę / mecz
SoT%	procent strzałów na bramkę
G/Sh	gole/strzały
G/SoT	gole/strzały na bramkę
ShoDist	odległość oddawania strzału
ShoFK	strzały z rzutów wolnych / mecz
ShoPK	strzelone rzuty karne / mecz
PKatt	wykonane rzuty karne / mecz
PasTotCmp	celne podania / mecz
PasTotAtt	próby podań / mecz
PasTotCmp%	celność podań
PasTotDist	całkowita odległość podań / mecz
PasTotPrgDist	progresywna odległość podań / mecz
PasShoCmp	celne krótkie podania / mecz
PasShooAtt	próby krótkich podań / mecz
PasShoCmp%	celność krótkich podań
PasMedCmp	celne średnie podania / mecz
PasMedAtt	próby średnich podań / mecz
PasMedCmp%	celność średnich podań
PasLonCmp	celne długie podania / mecz
PasLonAtt	próby długich podań / mecz
PasLonCmp%	celność długich podań
Assists	asysty /mecz
PasAss	podania prowadzące do strzału / mecz
Pas3rd	podania w ostatnią tercję boiska / mecz
PPA	podania w pole karne / mecz
CrsPA	dośrodkowania w pole karne / mecz
PasProg	progresywne podania / mecz
PasAtt	próby podań / mecz
PasLive	podania z gry / mecz
PasDead	podania ze stojącej piłki / mecz

PasFK	podania z rzutów wolnych / mecz
TB	podania między obrońcami na wolną przestrzeń / mecz
Sw	zmiany stron (podania na ponad 36.6m na szerokość boiska) / mecz
PasCrs	dośrodkowania / mecz
CK	rzuty różne / mecz
CkIn	rzuty różne dochodzące do bramki / mecz
CkOut	rzuty różne odchodzące od bramki / mecz
CkStr	proste rzuty różne / mecz
PasOff	spalone po podaniach / mecz
PasBlocks	zablokowane podania / mecz
SCA	akcje prowadzące do strzału / mecz
ScaPassLive	celne podania z gry prowadzące do strzału / mecz
ScaPassDead	podania ze stojącej piłki prowadzące do strzału / mecz
ScaDrib	dryblingi prowadzące do strzału / mecz
ScaSh	strzały prowadzące do innego strzału / mecz
ScaFld	faule prowadzące do strzału / mecz
ScaDef	akcje defensywne prowadzące do strzału / mecz
GCA	akcje prowadzące do gola / mecz
GcaPassLive	celne podania z gry prowadzące do gola / mecz
GcaPassDead	celne podania ze stojącej piłki prowadzące do gola / mecz
GcaDrib	dryblingi prowadzące do gola / mecz
GcaSh	strzały prowadzące do gola / mecz
GcaFld	faule prowadzące do gola / mecz
GcaDef	akcje defensywne prowadzące do gola / mecz
Tkl	próby odbiorów / mecz
TklWon	udane odbiory / mecz
TklDef3rd	odbiory w defensywnej tercji / mecz
TklMid3rd	odbiory w środkowej tercji / mecz
TklAtt3rd	odbiory w ofensywnej tercji / mecz

TklDri	odbioru dryblującemu przeciwnikowi / mecz
TklDriAtt	próby odbiorów dryblującego przeciwnika / mecz
TklDri%	skuteczność odbiorów dryblującego przeciwnika / mecz
TklDriPast	minięcie przez drybling / mecz
Blocks	zablokowane piłki / mecz
BlkSh	zablokowane strzały / mecz
BlkPass	zablokowane podania / mecz
Int	przechwyty / mecz
Tkl+Int	odbioru + przechwyty / mecz
Clr	wybicia piłki / mecz
Err	błędy prowadzące do strzału przeciwnika / mecz
Touches	kontakty z piłką / mecz
TouDefPen	kontakty we własnym polu karnym / mecz
TouDef3rd	kontakty w defensywnej tercji / mecz
TouMid3rd	kontakty w środkowej tercji / mecz
TouAtt3rd	kontakty w ofensywnej tercji / mecz
TouAttPen	kontakty w polu karnym przeciwnika / mecz
TouLive	kontakty z piłką w grze / mecz
DriSucc	udane dryblingi / mecz
DriAtt	próby dryblingów / mecz
Carries	prowadzenia piłki / mecz
CarTotDist	całkowity dystans prowadzenia piłki / mecz
CarPrgDist	progresywny dystans prowadzenia piłki / mecz
CarProg	progresywne prowadzenie piłki / mecz
Car3rd	prowadzenie piłki w ostatnią tercję / mecz
CPA	prowadzenie piłki w pole karne / mecz
CarMis	straty piłki przy prowadzeniu / mecz
CarDis	straty piłki po odbiorze przeciwnika / mecz
Rec	odebrane podania / mecz
RecProg	progresywne odebrane podania / mecz

CrdY	żółte kartki / mecz
CrdR	czerwone kartki / mecz
2CrdY	drugie żółte kartki / mecz
FIs	faule popełnione / mecz
FId	faule wywalczone / mecz
Off	spalone / mecz
Crs	dośrodkowania / mecz
TkIW	odbory z przejęciem piłki / mecz
PKwon	wywalczone rzuty karne / mecz
PKcon	sprokurowane rzuty karne / mecz
OG	gole samobójcze / mecz
Recov	liczba odzyskanych bezpieczeństwa piłek / mecz
AerWon	wygrane pojedynki powietrzne / mecz
AerLost	przegrane pojedynki powietrzne / mecz
Deep_Progression_M	metryka głębszej progresji
Fouls_Won_M	metryka fauli
Finishing_Efficiency_M	metryka jakości wykończenia
Chance_Creation_M	metryka kreowania okazji
Goal_Creation_M	metryka kreowania goli
Aerial_Duels_M	metryka pojedynków powietrznych
Defensive_Actions_M	metryka akcji defensywnych
Aggressiveness_M	metryka agresywności
Dribbles_M	metryka dryblingu
Pressure_M	metryka pressingu
Goals_M	metryka goli
Assists_M	metryka asyst
image_url	link do zdjęcia zawodnika
season	sezon

W niniejszym podrozdziale przedstawiono kluczowe statystyki zawodników, które stanowią fundament analizy stylu gry zawodnika oraz oceny potencjału w kontekście transferów. Wśród najistotniejszych danych znajdują się pozycja zawodnika, która jest kluczowa przy wyszukiwaniu graczy na daną pozycję oraz przy obliczaniu średnich metryk dla danych pozycji. Równie ważne są metryki oparte na statystykach zakończonych na "\_M", które zostały obliczone przy użyciu wskaźników takich jak podania progresywne, próby odbiorów czy udane odbory. Te metryki są podstawą do trenowania modeli predykcyjnych, które przewidują statystyki wybranego zawodnika.

Dzięki tym statystykom możliwe jest dokładne określenie efektywności zawodników w różnych aspektach gry, a także dokonanie bardziej precyzyjnych porównań między piłkarzami.

### 3.3. Statystyki zespołów

Poniżej zaprezentowano dane dotyczące drużyn piłkarskich, które zostały wykorzystane w kontekście aplikacji. Statystyki te obejmują zarówno ogólne informacje o drużynach, takie jak liga, trener, formacja oraz szczegółowe statystyki boiskowe, np. odległość oddawania strzałów, liczba krótkich podań na mecz.

Poniższa tabela zawiera nazwy i objaśnienia kolumn wykorzystywanych w projekcie do analizy zespołów. Każda z tych danych ma na celu dostarczenie pełniejszego obrazu na temat taktyki drużyny, jej stylu gry oraz porównania z innymi zespołami.

*Tabela 3.2. – dane o zespołach*

Nazwa kolumny	Opis
Team	drużyna
League	liga
Season	sezon
Coach	trener
Formation	formacja
Place	miejsce
Goals	gole
Shots pg	strzały / mecz
xG	współczynnik oczekiwanych goli
Non-OG Goals	gole (bez samobójczych)
xG Diff	różnica między strzelonymi golami, a xG
Shots	strzały
xG/Shots	xG na strzały
Yellow cards	żółte kartki
Red cards	czerwone kartki
Possession%	procent posiadania piłki
Pass%	procent celnych podań
AerialsWon	wygrane pojedynki powietrzne / mecz
Tackles pg	odbiory / mecz
Interceptions pg	przechwyty / mecz
Fouls pg	faule / mecz
Offside pg	spalone / mecz
ShOutOfBox	strzały spoza pola karnego / mecz



ShSixYardBox	strzały z pola bramkowego / mecz
ShPenaltyArea	strzały z pola karnego / mecz
Dribbles pg	dryblingi / mecz
Open Play	gole z gry
CounterAttack	gole z kontrataku
Set Piece	gole ze stałych fragmentów gry
Penalty	gole z rzutów karnych
Long Balls pg	długie podania / mecz
Short Passes pg	krótkie podania / mecz
Left Side %	ataki lewą stroną
Middle %	ataki środkiem
Right side %	ataki prawą stroną
Own Third %	posiadanie piłki na własnej tercji boiska
Middle Third %	posiadanie piłki w środkowej tercji boiska
Opposition Third %	posiadanie piłki na przeciwnej tercji boiska
Directness	metryka bezpośredniości ataku
Shot Distance	metryka dystansu strzału
Dribbles	metryka dryblingu
Fouls	metryka fauli
Counter Attack	metryka kontrataków
Chance Creation	metryka kreowania szans
Finishing	metryka wykończenia akcji
Long Balls	metryka długich podań
Agressiveness	metryka agresywności
Aerial Play	metryka gry powietrznej
Possession	metryka posiadania piłki
Control	metryka kontroli spotkania
Set Pieces	metryka stałych fragmentów gry

Na tym etapie zaprezentowane zostały kluczowe dane, które dostarczają podstawowej wiedzy o stylu gry zespołów. Wśród najważniejszych informacji znajdują się podstawowe dane takie jak liga, trener oraz główna formacja. Ponadto, statystyki dotyczące posiadania piłki w różnych tercjach boiska pozwalają zrozumieć, jak drużyna dominuje nad rywalami i kontroluje przebieg meczu. Na podstawie tych danych obliczona została metryka kontrolowania gry, która odzwierciedla jej zdolność do przejęcia inicjatywy na boisku. Ważną rolę odgrywają również pozostałe metryki, jak np. bezpośredniość ataku czy agresywność, które były obliczane na podstawie innych kolumn z danymi, takich jak liczba strzałów, goli, odbiorów piłki oraz żółtych i czerwonych kartek.

Metryki były niezbędne do wykonania wykresów, które wizualizują styl gry drużyny. Dzięki tym szczegółowym statystykom możliwe jest wyciąganie wniosków o mocnych stronach drużyny oraz obszarach do poprawy.

### 3.4. Obliczanie metryk

Obliczanie metryk, które odpowiednio opisują styl gry danego zawodnika i zespołu były kluczowe, aby przystąpić do trenowania modeli. Metryki zostały obliczone wykorzystując szczegółowe statystyki z plików z danymi. Po obliczeniu zostały one przeskalowane do zakresu od 0 do 10 i zaokrąglone do dwóch cyfr po przecinku.

#### 3.4.1. Obliczenie metryk zawodników

W tej części pracy przedstawione zostaną metody obliczania kluczowych metryk, które pozwalają na szczegółową ocenę wydajności zawodników. Obliczone metryki, takie jak głęboka progresja, skuteczność wykończenia czy kreowanie okazji pozwalają na kompleksową ocenę wpływu zawodnika na przebieg meczu w różnych aspektach gry. Każda z tych metryk jest obliczana na podstawie odpowiednich statystyk. W poniższym opisie szczegółowo przedstawione zostaną wzory oraz sposób obliczeń dla każdej z tych metryk.

Głęboka progresja to suma statystyk progresywnych prowadzeń piłki i progresywnych podań, pomnożona razy 500.

$$Deep\ Progression = (Car\ Prog + Pas\ Prog) \cdot 500$$

Wywalczone faule to statystyka fauli pomnożona razy 500.

$$Fouls\ Won = Fld \cdot 500$$

Skuteczność wykończenia to iloraz statystyki goli przez statystykę strzałów, pomnożony razy 500.

$$Finishing\ Efficiency = \frac{Goals}{Shots} \cdot 500$$

Kreowanie okazji to statystyka akcji prowadzących do strzału, pomnożona razy 500.

$$Chance\ Creation = SCA \cdot 500$$

Kreowanie goli to statystyka akcji prowadzących do bramki, pomnożona razy 500.

$$Goal\ Creation = GCA \cdot 500$$

Gole to statystyka goli pomnożona razy 500.

$$Goals = Goals \cdot 500$$

Asysty to statystyka asyst pomnożona razy 500.

$$Assists = Assists \cdot 500$$

Pojedynki powietrzne to iloraz wygranych pojedynków powietrznych przez sumę wygranych oraz przegranych pojedynków powietrznych, pomnożony razy 500.

$$Aerial\ Duels = \frac{AerWon}{AerWon + AerLost} \cdot 500$$

Akcje defensywne to suma statystyki liczby prób odbiorów, wygranych odbiorów pomnożonych razy 3, przechwytów pomnożonych razy 3, zablokowanych piłek pomnożonych razy 2, odzyskanych bezpiecznych piłek pomnożonych razy 3 i wybić piłki pomnożonych razy 2, pomnożona razy 50.

$$Defensive\ Actions = (Tkl + TklWon \cdot 3 + Int \cdot 3 + Blocks \cdot 2 + Recov \cdot 3 + Clr \cdot 2) \cdot 50$$

Agresywność to iloraz sumy statystyk żółtych i czerwonych kartek przez liczbę prób odbiorów pomnożony razy 500.

$$Aggressiveness = \frac{CrdY + CrdR}{Tkl} \cdot 500$$

Dryblingi to iloraz statystyki udanych dryblingów pomnożonych razy 5 przez liczbę prób dryblingów, pomnożony razy 100.

$$Dribbles = \frac{DriSucc \cdot 5}{DriAtt} \cdot 100$$

Pressing to suma statystyk odbiorów w defensywnej tercji boiska, odbiorów w środkowej tercji boiska pomnożonych razy 3 oraz odbiorów w ofensywnej tercji boiska pomnożonych razy 5, pomnożona razy 150.

$$Pressure = (TklDef3rd + TkldMid3rd \cdot 3 + TklAtt3rd \cdot 5) \cdot 150$$

### 3.4.2. Obliczenie metryk drużyny

W tej sekcji przedstawiono metody obliczania kluczowych metryk, które pozwalają na ocenę stylu gry drużyn. Metryki takie jak bezpośredniość, odległość strzału, faule, agresywność i kontrola gry są obliczane na podstawie statystyk meczowych i stanowią istotny element analizy taktyki zespołów. Dzięki tym obliczeniom możliwe jest lepsze

zrozumienie strategii drużyny, jej podejścia do gry, a także wykonanie wizualizacji stylu gry na późniejszym etapie projektu.

Bezpośredniość to iloraz statystyki strzałów na mecz przez sumę długich piłek i krótkich podań na mecz pomnożonych razy 2, pomnożony razy 500.

$$Directness = \frac{Shots\ pg}{Long\ Balls\ pg + Short\ Passes\ pg \cdot 2} \cdot 500$$

Odległość strzału to iloraz statystyki strzałów spoza pola karnego przez strzały na mecz, pomnożony razy 500.

$$Shot\ Distance = \frac{ShOutOfBox}{Shots\ pg} \cdot 500$$

Faule to iloraz statystyki fauli na mecz przez sumę odbiorów i przechwytyłów na mecz, pomnożony razy 500.

$$Fouls = \frac{Fouls\ pg}{Tackles\ pg + Interceptions\ pg} \cdot 500$$

Kontrataki to iloraz statystyki kontrataków przez liczbę goli, pomnożony razy 500.

$$Counter\ Attack = \frac{CounterAttack}{Goals} \cdot 500$$

Stwarzanie szans to statystyka współczynnika oczekiwanych goli.

$$Chance\ Creation = xG$$

Wykończenie to statystyka różnicy goli i oczekiwanych goli pomnożona razy 5.

$$Finishing = xG\ Diff \cdot 5$$

Długie piłki to iloraz statystyki długich piłek na mecz przez sumę długich piłek i krótkich podań na mecz, pomnożony razy 500.

$$Long\ Balls = \frac{Long\ Balls\ pg}{Long\ Balls\ pg + Short\ Passes\ pg} \cdot 500$$

Agresywność to iloraz sumy żółtych kartek i czerwonych kartek pomnożonych razy 3, przez faule na mecz.

$$Aggressiveness = \frac{(Yellow\ cards + Red\ cards \cdot 3)}{Fouls\ pg}$$

Gra powietrzna to statystyka wygranych pojedynków powietrznych razy 5.

$$Aerial\ Play = AerialsWon \cdot 5$$

Posiadanie to statystyka posiadania piłki.

$$Possession = Possession\%$$

Kontrola to iloraz statystyki posiadania piłki na przeciwnej tercji boiska przez posiadanie piłki na własnej tercji boiska, pomnożona razy 50.

$$Control = \frac{Opposition\ Third\ \%}{Own\ Third\ \%} \cdot 50$$

Stałe fragmenty gry to iloraz statystyki goli ze stałych fragmentów gry przez liczbę bramek, pomnożony razy 500.

$$Set\ Pieces = \frac{Set\ Piece}{Goals} \cdot 500$$

### 3.5. Opis narzędzi

W niniejszym rozdziale szczegółowo przedstawiono narzędzia i biblioteki, które zostały wykorzystane w ramach realizacji projektu. W pracy wykorzystano język programowania Python oraz różnorodne biblioteki, które wspierają analizę danych, budowę modeli uczenia maszynowego oraz tworzenie interfejsu użytkownika. Wybór technologii był podyktowany ich funkcjonalnością, wszechstronnością oraz łatwością integracji z pozostałymi bibliotekami. Poniżej zaprezentowano narzędzia niezbędne do realizacji projektu wraz z najważniejszymi cechami oraz konkretnymi zastosowaniami w kontekście pracy.

#### 3.5.1. Python

Dokładne szczegóły na temat Pythona można znaleźć w oficjalnej dokumentacji tego języka programowania [5], która stanowi podstawę tego rozdziału. Python to dynamicznie typowany język programowania, który zdobył popularność dzięki swojej prostocie i czytelności. Został stworzony przez Guido van Rossum w 1991 roku i od tego czasu rozwija się jako projekt open-source. Python wyróżnia się składnią zbliżoną do języka naturalnego, co czyni go łatwym w nauce i przyjaznym zarówno dla początkujących, jak i zaawansowanych programistów.

## Cechy Pythona

### 1. Czytelność i prostota

Python został zaprojektowany z naciskiem na czytelność kodu. Dzięki temu pozwala na szybkie pisanie i rozumienie skryptów, co jest kluczowe w projektach akademickich i komercyjnych.

### 2. Wszechstronność

Python znajduje zastosowanie w różnych dziedzinach:

- Analiza danych i uczenie maszynowe (biblioteki takie jak Pandas, NumPy, Scikit-learn).
- Tworzenie aplikacji webowych (frameworki takie jak Django, Flask).
- Automatyzacja zadań.
- Rozwój grafiki i gier komputerowych.

### 3. Dynamiczne typowanie

Python umożliwia programowanie bez konieczności deklarowania typów zmiennych, co pozwala na szybsze tworzenie kodu.

### 4. Rozbudowany ekosystem bibliotek

Python posiada bogatą bazę bibliotek i modułów, które rozszerzają jego możliwości – od analizy danych (Pandas, NumPy), przez wizualizacje (Matplotlib, Seaborn), po uczenie maszynowe (TensorFlow, PyTorch).

## Zastosowania Pythona

W niniejszej pracy Python pełnił kluczową rolę jako język umożliwiający:

- Przetwarzanie i analizę danych za pomocą bibliotek takich jak Pandas i NumPy.
- Tworzenie wizualizacji danych z wykorzystaniem bibliotek Matplotlib i Seaborn.
- Skalowanie danych i przygotowanie ich do modeli za pomocą narzędzi z biblioteki Scikit-learn.
- Implementację modeli uczenia maszynowego (RandomForestRegressor) oraz ocenę ich wyników.
- Budowę interfejsu użytkownika za pomocą Streamlit, umożliwiającego prezentację wyników w sposób dynamiczny.
- Wbudowany moduł Unicodedata umożliwił przekształcenie znaków złożonych na ich proste odpowiedniki. Na przykład, litera "é" została rozbита na "e", aby ułatwić użytkownikowi proces wyszukiwania zawodnika.

Python to wszechstronny język programowania, który dzięki swojej elastyczności i szerokim możliwościom pozwolił na efektywną realizację zadań w mojej pracy inżynierskiej.

## 3.5.2. Biblioteki do analizy i przetwarzania danych

### 1. Pandas

Wszystkie informacje zawarte w tym podrozdziale zostały zaczerpnięte z dokumentacji biblioteki [6]. Pandas to jedna z najpopularniejszych bibliotek Pythona, zaprojektowana do przetwarzania i analizy danych. Nazwa biblioteki pochodzi od „Panel Data” i podkreśla jej przeznaczenie do pracy z dużymi zbiorami danych tabelarycznych.

### Cechy biblioteki Pandas

#### 1. Struktury danych: Series i DataFrame

- Series: Jednowymiarowa struktura przypominająca listę lub kolumnę z bazy danych.
- DataFrame: Dwuwymiarowa struktura danych, która jest tablicą złożoną z wierszy i kolumn. Pozwala na przechowywanie danych w formacie podobnym do tabel w arkuszach kalkulacyjnych.

#### 2. Łatwe manipulowanie danymi

- Pandas oferuje funkcje umożliwiające filtrowanie, sortowanie, grupowanie oraz przekształcanie danych.
- Obsługuje operacje takie jak wstawianie, usuwanie i edytowanie kolumn oraz wierszy.

#### 3. Obsługa różnych formatów danych

Pandas pozwala na łatwe ładowanie i zapisywanie danych w popularnych formatach, takich jak .csv, .xlsx (Excel), .json i inne.

#### 4. Obsługa brakujących wartości

Biblioteka zawiera zaawansowane narzędzia do wykrywania, wypełniania oraz usuwania brakujących danych.

#### 5. Integracja z innymi bibliotekami

Pandas doskonale współpracuje z innymi bibliotekami Pythona, takimi jak NumPy, Matplotlib czy Scikit-learn.

### Zastosowania Pandas

W mojej pracy inżynierskiej biblioteka Pandas była wykorzystywana na takich etapach realizacji projektu jak:

- Wczytywanie danych: Dane były ładowane z plików .csv za pomocą funkcji takich jak `pd.read_csv()`.
- Przetwarzanie danych: Manipulowanie strukturą danych, modyfikując kolumny, łącząc zbiory danych oraz przekształcając dane do odpowiedniego formatu.
- Analiza danych: Korzystanie z funkcji takich jak `groupby` oraz `describe` do agregacji i wyciągania podstawowych statystyk z danych.

- Przygotowanie danych do uczenia maszynowego: Pandas pomogło w czyszczeniu i przygotowania danych wejściowych do trenowania modeli.

Biblioteka Pandas to kluczowe narzędzie w analizie danych, oferujące szeroki zakres funkcjonalności do manipulacji danymi w sposób intuicyjny i wydajny. Znacznie ułatwiła proces przetwarzania danych.

## 2. NumPy

Podrozdział o bibliotece NumPy bazuje na materiałach zawartych w dokumentacji [7]. NumPy (Numerical Python) to biblioteka Pythona zaprojektowana do obliczeń numerycznych i pracy z tablicami wielowymiarowymi. Jest podstawą wielu innych bibliotek związanych z analizą danych i uczeniem maszynowym, takich jak Pandas, Matplotlib czy Scikit-learn. Dzięki swojej wydajności i funkcjonalności stała się kluczowym narzędziem w środowisku analizy danych.

### Cechy biblioteki NumPy

#### 1. Tablice wielowymiarowe (ndarray)

Głównym obiektem NumPy jest ndarray, który reprezentuje tablicę wielowymiarową. W odróżnieniu od standardowych list Pythona, tablice NumPy są bardziej wydajne pod względem pamięci oraz pozwalają na szybkie obliczenia matematyczne.

#### 2. Szybkie obliczenia matematyczne

NumPy wykorzystuje implementację w języku C, co sprawia, że operacje matematyczne na dużych tablicach są bardzo wydajne w porównaniu do natywnych struktur Pythona.

#### 3. Obsługa funkcji matematycznych i statystycznych

Biblioteka NumPy zawiera wiele funkcji matematycznych, takich jak obliczenia macierzowe, statystyki (np. średnia, mediana, odchylenie standardowe), funkcje trygonometryczne oraz algebrę liniową.

#### 4. Generowanie liczb losowych

NumPy zawiera moduł `numpy.random`, który pozwala na generowanie liczb losowych z różnych rozkładów probabilistycznych, co jest przydatne w symulacjach i modelowaniu.

#### 5. Integracja z innymi bibliotekami

NumPy jest szeroko wykorzystywane jako podstawa dla innych bibliotek analitycznych, takich jak Pandas czy Scikit-learn.



## Zastosowania NumPy

W mojej pracy inżynierskiej NumPy pełniło kluczową rolę w realizacji następujących zadań:

- Przetwarzanie danych numerycznych: NumPy wykorzystano do szybkich obliczeń na dużych zbiorach danych, takich jak normalizacja i standaryzacja.
- Statystyki i analiza danych: Dzięki wbudowanym funkcjom matematycznym NumPy umożliwiło obliczenie podstawowych miar statystycznych (np. średnia, mediana).

NumPy to niezwykle wydajne i wszechstronne narzędzie do pracy z danymi numerycznymi. Dzięki swojej funkcjonalności pozwoliło znacząco przyspieszyć proces obliczeń w mojej pracy inżynierskiej i zachować czytelność kodu.

## 3. Matplotlib

Podrozdział poświęcony bibliotece Matplotlib opiera się na informacjach zawartych w oficjalnej dokumentacji [8]. Ta biblioteka służy do wizualizacji danych i jest elastycznym narzędziem, które pozwala generować różnorodne wykresy. Pakiet Matplotlib jest szeroko stosowany w nauce, inżynierii, finansach i analizie danych.

### Cechy biblioteki Matplotlib

#### 1. Wielorakość typów wykresów

Matplotlib umożliwia tworzenie wielu różnych typów wykresów, takich jak wykresy liniowe, słupkowe, wykresy 3D, histogramy i wiele innych.

#### 2. Elastyczność i dostosowywanie

Matplotlib pozwala na pełną kontrolę nad wyglądem wykresów: kolorami, rozmiarami, stylami linii, etykietami osi, tytułami oraz legendami.

#### 3. Interaktywność

Biblioteka wspiera interaktywne środowiska, takie jak Jupyter Notebook, co pozwala na dynamiczne eksplorowanie danych.

#### 4. Zintegrowana z innymi bibliotekami

Matplotlib doskonale współpracuje z bibliotekami NumPy i Pandas, co ułatwia bezpośrednie tworzenie wykresów z danych zapisanych w tych strukturach.

#### 5. Wsparcie dla różnych formatów plików wyjściowych

Wykresy mogą być eksportowane do różnych formatów, takich jak .png, .pdf, czy .jpg, co jest przydatne w przygotowywaniu raportów czy publikacji.

### Podstawowe moduły Matplotlib

- matplotlib.pyplot: Najczęściej używany moduł, który dostarcza funkcje do tworzenia wykresów w stylu podobnym do MATLAB-a.

- `mpl_toolkits.mplot3d`: Moduł umożliwiający tworzenie wykresów 3D.
- `matplotlib.animation`: Moduł służący do tworzenia animacji w oparciu o dane.

## **Zastosowanie Matplotlib**

W opisywanym projekcie biblioteka Matplotlib była używana głównie do:

- Wizualizacji wyników analiz danych: Tworzenie wykresów radarowych i suwaków do zobrazowania stylu gry zawodników oraz drużyn.
- Analizy danych eksploracyjnej (EDA): Histogramy i wykresy punktowe pozwalały zrozumieć rozkład danych i ich relacje.

Matplotlib to wszechstronna biblioteka do wizualizacji danych w Pythonie. Dzięki swojej elastyczności pozwoliła skutecznie przedstawiać wyniki analiz i wspierać proces interpretacji danych.

## **4. Seaborn**

Podrozdział o Seaborn bazuje na materiałach zawartych w dokumentacji [9]. To zaawansowana biblioteka do wizualizacji danych oparta na Matplotlib, zaprojektowana z myślą o łatwiejszym tworzeniu estetycznych wykresów. Biblioteka Seaborn integruje się również z Pandas, co ułatwia pracę z danymi zapisanymi w formacie tabelarycznym.

### **Cechy biblioteki Seaborn**

#### **1. Estetyczne i czytelne wizualizacje**

Domyślne style i kolory Seaborn są zaprojektowane tak, aby wykresy były bardziej przejrzyste i atrakcyjne wizualnie.

#### **2. Wbudowane funkcje statystyczne**

Seaborn oferuje funkcje do wizualizacji zintegrowane z analizą statystyczną, takie jak:

- Dodawanie linii regresji do wykresów
- Wizualizacja rozkładów danych
- Tworzenie macierzy korelacji

#### **3. Zaawansowane typy wykresów**

Seaborn zapewnia zaawansowane typy wykresów, takie jak heatmaps.

#### **4. Obsługa wielowymiarowych danych**

Seaborn ułatwia analizę relacji między zmiennymi, oferując funkcje takie jak `pairplot` (wykresy par zmiennych) czy `facetgrid` (siatki wykresów).

### **Podstawowe funkcje i moduły**

- `sns.scatterplot`: Do tworzenia wykresów punktowych.
- `sns.lineplot`: Do rysowania wykresów liniowych.

- `sns.barplot`: Do wykresów słupkowych.
- `sns.heatmap`: Do wizualizacji danych w formie macierzy (np. macierzy korelacji).
- `sns.pairplot`: Tworzenie wykresów dla każdej pary zmiennych w zbiorze danych.

### Zastosowanie Seaborn

W mojej pracy Seaborn był używany głównie do wizualizacji rozkładów danych, analizy korelacji między zmiennymi za pomocą heatmap.

Biblioteka Seaborn to istotne narzędzie do wizualizacji danych, które łączy estetykę z funkcjonalnością. Pozwoliło ono skutecznie przedstawiać dane w sposób zrozumiały i atrakcyjny wizualnie.

## 3.5.3. Biblioteki do uczenia maszynowego

### 1. Scikit-learn

Dokładne szczegóły na temat pakietu Scikit-learn można znaleźć w oficjalnej dokumentacji [10], która stanowi podstawę tego podrozdziału. Scikit-learn to jedna z najpopularniejszych bibliotek w Pythonie, służąca do uczenia maszynowego. Biblioteka ta oferuje szeroki zestaw narzędzi do modelowania danych, takich jak klasyfikacja, regresja, klasteryzacja, czy metody walidacji. Jest zbudowana na podstawie bibliotek NumPy, SciPy i Matplotlib, co zapewnia jej efektywność oraz łatwą integrację z innymi narzędziami Pythona.

### Cechy biblioteki Scikit-learn

#### 1. Szeroka gama algorytmów uczenia maszynowego

Scikit-learn obsługuje różne metody, w tym algorytmy klasyfikacji, regresji, czy klasteryzacji.

#### 2. Łatwość użycia

Scikit-learn oferuje spójny interfejs API, dzięki czemu modele są łatwe do trenowania, oceniania i optymalizacji.

#### 3. Przydatne narzędzia wspierające

Biblioteka pozwala na przekształcanie danych poprzez skalowanie, normalizację, czy redukcję wymiarowości, a także zapewnia funkcje do oceny skuteczności modeli, np.  $R^2$ , MSE. Ważną kwestią jest też możliwość automatyczne dzielenie danych na zbiory treningowe i testowe za pomocą funkcji `train_test_split`.

#### 4. Wydajność

Biblioteka jest zoptymalizowana do pracy z dużymi zbiorami danych dzięki wykorzystaniu NumPy i SciPy.

## Podstawowe moduły i funkcje

- **Preprocessing (Przetwarzanie danych)**  
Moduł `sklearn.preprocessing` umożliwia skalowanie i normalizację danych, co jest niezbędne do poprawnego działania wielu algorytmów. Przykład: `StandardScaler`, `MinMaxScaler`.
- **Podział danych**  
Funkcja `train_test_split` z modułu `sklearn.model_selection` dzieli dane na zbiory treningowe i testowe.
- **Regresja i klasyfikacja**  
Moduły, takie jak `sklearn.ensemble` (np. `Random Forest`), czy `sklearn.linear_model` (np. regresja liniowa), oferują gotowe do użycia modele.
- **Walidacja modeli**  
Funkcje, takie jak `cross_val_score` czy `GridSearchCV`, wspierają dobór parametrów modeli i ocenę ich wydajności.
- **Metryki oceny**  
Moduł `sklearn.metrics` pozwala na ocenę wyników modeli, np. `mean_squared_error`, `r2_score`.

## Zastosowanie Scikit-learn

Scikit-learn był kluczowym narzędziem do budowy i oceny modeli uczenia maszynowego podczas realizacji projektu. Oto konkretne zastosowania:

1. **Podział danych**  
Użyłem funkcji `train_test_split` do podziału danych na zbiór treningowy i testowy.
2. **Skalowanie danych**  
`StandardScaler` pozwoliło na znormalizowanie danych wejściowych, aby poprawić wydajność modeli.
3. **Trenowanie modeli**  
Do przewidywania metryk zawodników wykorzystano algorytm lasów losowych (`Random Forest`), który osiągnął obiecujące rezultaty.
4. **Ocena modeli**  
Do oceny skuteczności trenowanych modeli użyto funkcji takich jak `mean_squared_error` i `r2_score`.

Scikit-learn odegrał kluczową rolę w mojej pracy, umożliwiając szybkie budowanie i ocenę modeli uczenia maszynowego. Dzięki swojej prostocie i szerokim możliwościom był idealnym wyborem do realizacji zadań związanych z predykcją.

## **2. Joblib**

Podrozdział opisujący Joblib oparty jest na oficjalnej dokumentacji tego pakietu [11]. Joblib to biblioteka w Pythonie zaprojektowana z myślą o efektywnym wykonywaniu zadań związanych z równoległym przetwarzaniem danych. Jest szczególnie przydatna w uczeniu maszynowym, gdzie modele mogą być duże i wymagają przechowywania lub ponownego użycia bez konieczności ponownego trenowania. Joblib wyróżnia się szybkością oraz możliwością pracy z dużymi danymi, co czyni ją lepszym wyborem w wielu przypadkach niż standardowa biblioteka pickle.

### **Cechy biblioteki Joblib**

#### **1. Efektywny zapis i wczytywanie modeli**

Joblib umożliwia zapisywanie i odczytywanie dużych obiektów Pythona, w tym modeli uczenia maszynowego. Proces zapisu jest zoptymalizowany pod kątem pracy z dużymi macierzami NumPy, co czyni Joblib idealnym narzędziem do przechowywania modeli.

#### **2. Równoległe przetwarzanie**

Dzięki modułowi Parallel, Joblib wspiera równoległe wykonywanie kodu, co przyspiesza czasochłonne operacje. Pozwala na równoległe przetwarzanie zadań na wielu rdzeniach CPU.

#### **3. Wsparcie dla dużych obiektów**

Joblib efektywnie zarządza pamięcią, dzieląc duże dane na bloki podczas zapisu, co pozwala uniknąć problemów związanych z przepełnieniem pamięci.

#### **4. Kompatybilność**

Biblioteka jest kompatybilna z popularnymi narzędziami ekosystemu Pythona, takimi jak NumPy, Pandas, czy Scikit-learn, co sprawia, że jest często używana w projektach uczenia maszynowego.

### **Zastosowanie Joblib w mojej pracy**

Joblib został użyty w projekcie do zapisu i odczytu modeli. Po zakończeniu trenowania modeli zostały one zapisane do pliku za pomocą funkcji `joblib.dump()`. Umożliwiło to załadowanie modeli bez konieczności ponownego trenowania, co oszczędzało czas i zasoby obliczeniowe.

Joblib jest niezastąpionym narzędziem w projektach związanych z uczeniem maszynowym, umożliwiając szybkie i efektywne zarządzanie obiektami oraz równoległe przetwarzanie. Jest pomocne przy zapisywaniu i ponownym wykorzystywaniu modeli na różnych etapach projektu.

### 3.5.4. Streamlit

Podrozdział opisujący framework Streamlit opiera się na informacjach zawartych w oficjalnej dokumentacji tego narzędzia [12]. Streamlit to nowoczesna i łatwa w użyciu biblioteka do tworzenia interaktywnych aplikacji webowych w Pythonie. Dzięki niej możliwe jest szybkie tworzenie aplikacji analitycznych, wizualizacyjnych, interaktywnych dashboardów oraz aplikacji do przetwarzania danych. Streamlit eliminuje konieczność znajomości skomplikowanych frameworków webowych, takich jak Flask czy Django, pozwalając na tworzenie aplikacji zaledwie w kilku liniach kodu. Streamlit dobrze współpracuje z popularnymi bibliotekami analitycznymi wykorzystanymi w tym projekcie, takimi jak NumPy, Pandas, czy Matplotlib.

#### Cechy biblioteki Streamlit

1. **Szybkie tworzenie aplikacji webowych**

Streamlit umożliwia szybkie tworzenie aplikacji z wykorzystaniem prostych skryptów w Pythonie. Tworzenie aplikacji interaktywnych nie wymaga zaawansowanej wiedzy na temat frameworków webowych.

2. **Interaktywność**

Streamlit automatycznie generuje komponenty, takie jak suwaki, formularze, przyciski, wykresy czy tabele, umożliwiające użytkownikowi interakcję z aplikacją w czasie rzeczywistym.

3. **Integracja z narzędziami do wizualizacji**

Streamlit wspiera szeroką gamę narzędzi do tworzenia wykresów i wizualizacji, takich jak Matplotlib, Plotly, Altair, Bokeh i inne. Dzięki temu użytkownicy mogą łatwo prezentować dane w formie wizualnej.

4. **Łatwa składnia**

Streamlit oferuje prostą składnię. Programista może korzystać z podstawowych elementów Pythona, takich jak funkcje, pętle czy zmienne, by budować dynamiczne aplikacje.

5. **Obsługa danych**

Streamlit współpracuje z bibliotekami analitycznymi, takimi jak Pandas i NumPy. Pozwala to na łatwe przetwarzanie danych w czasie rzeczywistym i dynamiczne aktualizowanie wyników na podstawie danych wejściowych.

6. **Cache'owanie**

Streamlit posiada pamięć podręczną, umożliwiając szybsze działanie aplikacji, zwłaszcza przy ponownym wywołaniu tych samych danych.

## **7. Obsługa multimediów**

Streamlit pozwala na łatwe dodawanie obrazów, plików audio, wideo oraz innych multimediów, co pozwala na wzbogacenie aplikacji o dane wizualne.

## **Zastosowanie Streamlit w mojej pracy**

W projekcie biblioteka Streamlit została wykorzystana jako narzędzie do stworzenia interaktywnej aplikacji webowej umożliwiającej użytkownikowi analizę i wizualizację danych związanych z potencjalnymi transferami piłkarskimi. Streamlit pozwolił na szybkie i łatwe zbudowanie intuicyjnego interfejsu użytkownika.

### **Główne zalety Streamlit w kontekście projektu to:**

#### **1. Łatwość w tworzeniu interaktywnych komponentów**

Streamlit umożliwia łatwe dodawanie elementów interaktywnych, takich jak pola tekstowe, suwaki, przyciski i wykresy, co sprawia, że użytkownicy mogą wchodzić w interakcje z aplikacją w czasie rzeczywistym.

#### **2. Szybka integracja z danymi**

Streamlit pozwolił szybko zintegrować aplikację z danymi zapisanymi w plikach .csv oraz zbudować dynamiczne wykresy i tabele, które w czasie rzeczywistym odzwierciedlają aktualny stan danych. W połączeniu z funkcjami Pandas, takie jak ładowanie i filtrowanie danych, aplikacja jest w stanie na bieżąco aktualizować informacje.

#### **3. Zoptymalizowana wydajność**

Dzięki wbudowanemu mechanizmowi cache'owania, Streamlit pozwolił na optymalizację działania aplikacji, co pozwoliło na szybsze ładowanie danych oraz szybsze renderowanie wykresów, nawet przy dużych zestawach danych.

#### **4. Szerokie możliwości wizualizacji**

Streamlit wspiera integrację z bibliotekami wizualizacyjnymi, takimi jak Matplotlib, co pozwoliło na tworzenie atrakcyjnych i interaktywnych wykresów, w tym wykresów radarowych, które służą do porównywania wyników zawodników z przeciwnymi wartościami w danej lidze.

Streamlit to narzędzie, które umożliwia szybkie tworzenie interaktywnych aplikacji webowych w Pythonie, zwłaszcza tych związanych z analizą i wizualizacją danych. Dzięki Streamlit udało się stworzyć przejrzystą i dynamiczną aplikację do analizy danych piłkarskich, której użytkownicy mogą łatwo przeglądać statystyki zawodników oraz porównywać ich wyniki.

## 4. Projekt i implementacja aplikacji

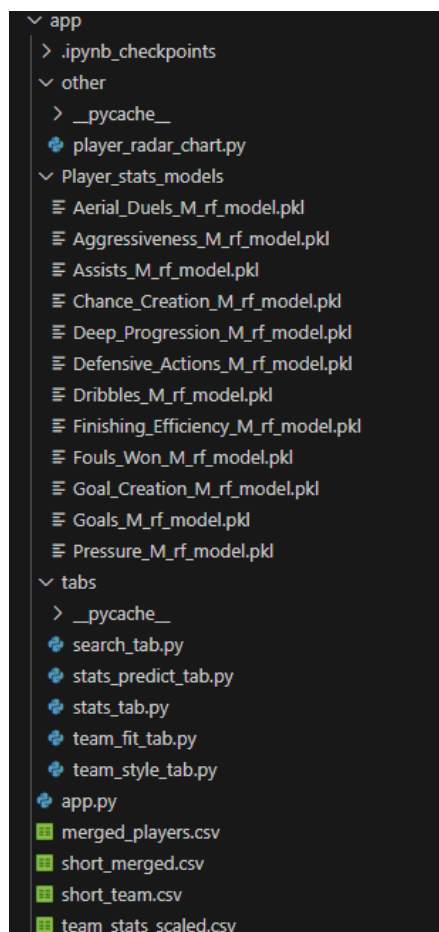
Celem niniejszego rozdziału jest przedstawienie procesu projektowania i implementacji aplikacji wspierającej ocenę potencjalnych transferów piłkarskich. Aplikacja ta została zbudowana w języku Python z wykorzystaniem frameworka Streamlit, który umożliwia tworzenie interaktywnych interfejsów użytkownika dla analiz danych. W projekcie uwzględniono zarówno wymagania funkcjonalne, jak i estetyczne, co pozwoliło stworzyć intuicyjne narzędzie skierowane do użytkowników związanych z branżą piłkarską.

Aplikacja została zaprojektowana w sposób modułowy, co pozwala na łatwą rozbudowę i utrzymanie kodu. Główny plik `app.py` (rys. 4.1.) pełni rolę centralnego punktu aplikacji, zarządzając konfiguracją strony, a także renderowaniem poszczególnych zakładek, które zostały zdefiniowane w oddzielnych plikach w folderze `tabs`. Każda z zakładek jest odpowiedzialna za inną część funkcjonalności aplikacji, np. analiza statystyk piłkarzy, przewidywanie statystyk czy ocena dopasowania do zespołu. Z kolei struktura aplikacji (rys. 4.2.) pokazuje logiczny podział projektu, w którym foldery takie jak `other`, `Player_stats_models` i `tabs` zawierają poszczególne moduły. Każdy z nich odpowiada za różne aspekty aplikacji, jak wykresy, modele uczenia maszynowego, interfejs każdej z zakładek. Taka budowa umożliwia efektywne zarządzanie kodem, a także jego łatwą modyfikację i skalowanie w przyszłości.

```
1 import streamlit as st
2 from tabs.search_tab import render_search_tab
3 from tabs.stats_tab import render_stats_tab
4 from tabs.team_style_tab import render_teams_style_tab
5 from tabs.stats_predict_tab import render_stats_predict_tab
6 from tabs.team_fit_tab import render_team_fit_tab
7
8 st.set_page_config(page_title="Ocena potencjalnych transferów piłkarskich", page_icon="⚽")
9
10 # Tytuł
11 st.title("Ocena potencjalnych transferów piłkarskich ⚽")
12
13 # Zakładki
14 tabs = st.tabs(["Wyszukiwanie", "Statystyki", "Styl gry zespołu", "Przewidywanie statystyk", "Ocena dopasowania do zespołu" ])
15
16 with tabs[0]:
17     render_search_tab()
18
19 with tabs[1]:
20     render_stats_tab()
21
22 with tabs[2]:
23     render_teams_style_tab()
24
25 with tabs[3]:
26     render_stats_predict_tab()
27
28 with tabs[4]:
29     render_team_fit_tab()
30
```

Rys 4.1. - Kod pliku `app.py`





Rys 4.2. - Struktura aplikacji

Aplikacja została podzielona na pięć głównych zakładek, odpowiadających różnym funkcjonalnościom: "Wyszukiwanie zawodników", "Statystyki", "Styl gry zespołu", "Przewidywanie statystyk" oraz "Ocena dopasowania do zespołu". Każda z nich oferuje unikalne możliwości, takie jak przeszukiwanie bazy danych zawodników, analiza ich kluczowych statystyk, wizualizacja stylu gry zespołów czy ocena potencjalnych transferów na podstawie dostępnych danych.

Zakładka "Wyszukiwanie zawodników" jest odpowiedzialna za przeglądanie bazy danych z zawodnikami według różnych kryteriów, takich jak pozycja, liga czy narodowość. Możliwe jest również filtrowanie wyników na podstawie zaawansowanych statystyk, takich jak liczba minut w sezonie, celność podań czy odbiory na mecz. Wyniki wyszukiwania są wyświetlane w formie tabeli, a same wyniki można eksportować do pliku .csv.

W zakładce "Statystyki" użytkownik ma możliwość przeglądania statystyk wybranego zawodnika. Zakładka wyświetla dane dotyczące piłkarza, takie jak pozycja, wiek, narodowość oraz szczegółowe statystyki boiskowe. Dodatkowo zakładka zawiera metryki zawodnika przedstawione na wykresie radarowym, które można łatwo porównać

do przeciętnego piłkarza grającego na tej samej pozycji, w tej samej lidze i sezonie co wybrany zawodnik.

Za analizę stylu gry zespołu odpowiada trzecia zakładka. Pozwala ona na wyświetlenie statystyk oraz metryk dotyczących konkretnej drużyny dla wybranego sezonu, a także informacji takich jak strony ataku i posiadanie piłki w poszczególnych tercjach boiska.

Zakładka "Przewidywanie statystyk" wykorzystuje modele predykcyjne i pozwala użytkownikowi na wybranie zawodnika oraz wyświetlenie przewidywanych metryk na sezon 2024-2025. Statystyki wyświetlają się zarówno w formie tekstowej jak i na wykresie radarowym.

Korzystając z zakładki "Ocena dopasowania do zespołu" użytkownik ma możliwość sprawdzenia dopasowania wybranego zawodnika do drużyny. Aplikacja porównuje kluczowe metryki zespołu w jakim dotychczas występował zawodnik oraz zespołu docelowego i wystawia ocenę dopasowania w skali od 1 do 6. Na podstawie wystawionej oceny oraz przy wykorzystaniu tych samych modeli predykcyjnych co w drugiej zakładce, aplikacja przedstawia możliwe statystyki zawodnika w nowej drużynie.

Dzięki zastosowaniu Streamlit możliwe było stworzenie aplikacji, która łączy zaawansowane analizy danych z prostotą obsługi. W tym rozdziale szczególną uwagę poświęcono opisowi interfejsu aplikacji, którego elementy zostały zilustrowane za pomocą zrzutów ekranu.

## 4.1. Wyszukiwanie zawodników

Domyślnie aplikacja wyświetla zakładkę „Wyszukiwanie” (rys. 4.3.) z funkcjonalnościami umożliwiającymi filtrowanie i przeszukiwanie bazy danych piłkarzy na podstawie różnych kryteriów. Zakładka zawiera paski rozwijalne umożliwiające wybór sezonu (rys. 4.4.), pozycji zawodnika, wiodącej nogi, a także opcję filtrowania po lidze (rys. 4.5.). Dodatkowo dostępne jest pole tekstowe do wyszukiwania zawodnika po nazwisku. Centralną część ekranu zajmuje graficzna reprezentacja boiska, na której wyszczególniana jest pozycja zawodnika, jeśli została wybrana (rys. 4.6.).

## Ocena potencjalnych transferów piłkarskich

[Wyszukiwanie](#)
[Statystyki](#)
[Styl gry zespołu](#)
[Przewidywanie statystyk](#)
[Ocena dopasowania do zespołu](#)

### Wyszukiwanie zawodników

Wybierz sezon:

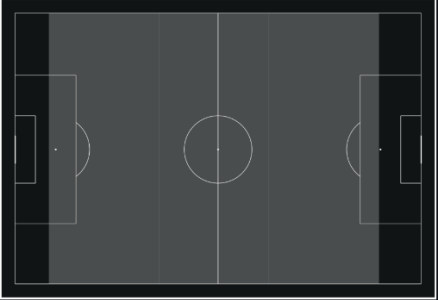
Dowolny

Szukaj zawodnika po nazwisku:

☐ Filtruj według ligi

Wybierz pozycję:

Dowolna



Wybierz wiodącą nogę:

Dowolna

Wybierz narodowość:

Choose an option

### Kluczowe statystyki

Wyświetl kluczowe statystyki

Rys. 4.3. - Zakładka "Wyszukiwanie zawodników"

Wybierz sezon:

Dowolny

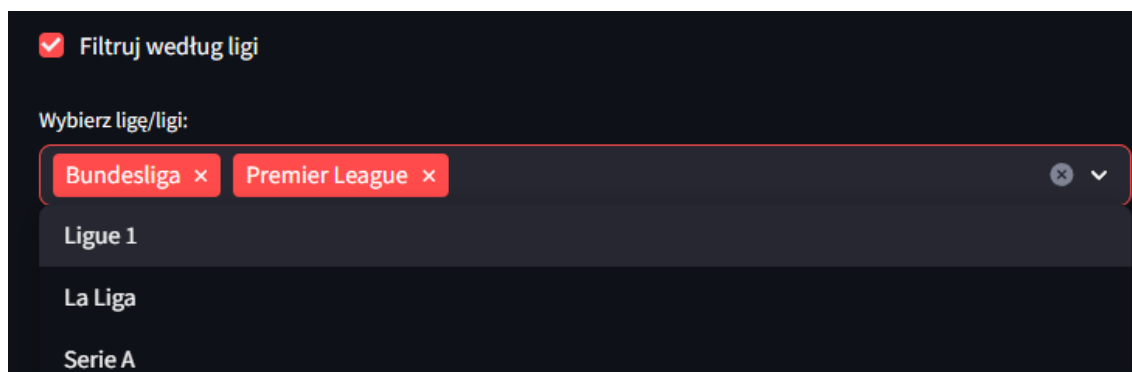
Dowolny

2021-2022

2022-2023

☐ Filtruj według ligi

Rys.4.4. - Wybór sezonu



Rys. 4.5. - Możliwość filtrowania po lidze



Rys. 4.6. - Wyświetlenie wybranej pozycji na boisku

W dolnej części strony umieszczono sekcję z kluczowymi statystykami, które użytkownik może wykorzystać do filtrowania zawodników (rys. 4.7.). Na końcu znajduje się przycisk "Szukaj zawodników", który uruchamia wyszukiwanie na podstawie wybranych kryteriów. Wyniki wyszukiwania wyświetlane są w formie tabeli (rys. 4.8.) prezentującej wszystkie dane dotyczące zawodnika i jego statystyki. Aplikacja daje możliwość pobrania tabeli z wynikami w formacie .csv.

## Kluczowe statystyki

Wyświetl kluczowe statystyki

☒ Liczba minut w sezonie

Liczba minut w sezonie (min wartość)

1200 Press Enter to apply

☐ Liczba goli

☐ Liczba strzałów

☐ Procent strzałów celnych

☐ Gole na strzał

☐ Śr. odległość strzałów

Rys. 4.7. - Rozwinięcie statystyk do filtrowania

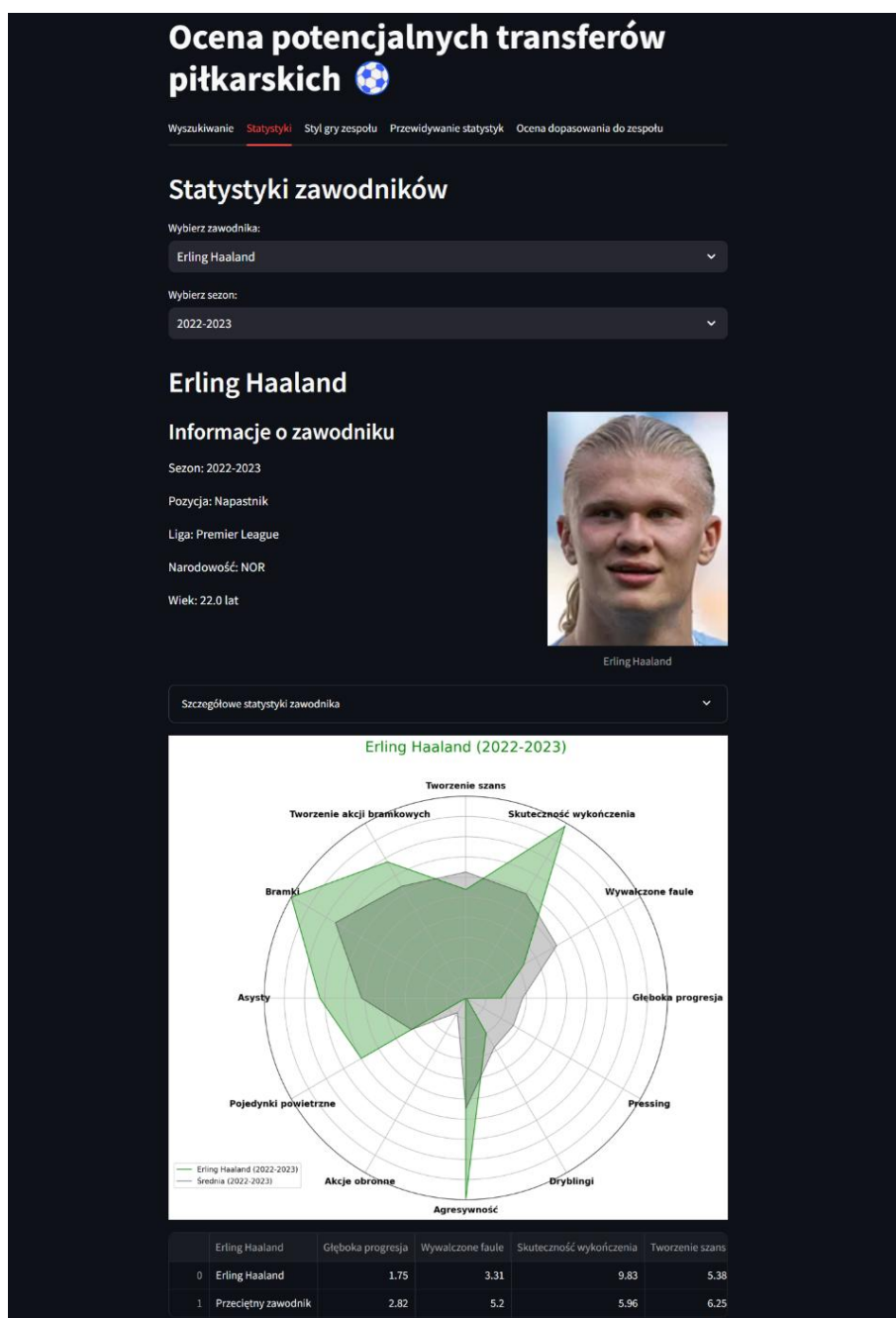
Znalezieni zawodnicy:

	Nazwisko	Sezon	Liga	Pozycja	Wiodąca noga	Narodowość	Drużyna
1	Lionel Messi	2021-2022	Ligue 1	FWMF	lewa	ARG	Paris S-G
2	Marco Reus	2021-2022	Bundesliga	MFFW	prawa	GER	Dortmund
3	Thomas Muller	2021-2022	Bundesliga	MF	prawa	GER	Bayern Munich
4	Christopher Nkunku	2021-2022	Bundesliga	FWMF	prawa	FRA	RB Leipzig
5	Dominik Szoboszlai	2021-2022	Bundesliga	MFFW	prawa	HUN	RB Leipzig
6	Florian Wirtz	2021-2022	Bundesliga	MFFW	prawa	GER	Leverkusen

Rys. 4.8. - Przykładowy wynik wyszukiwania

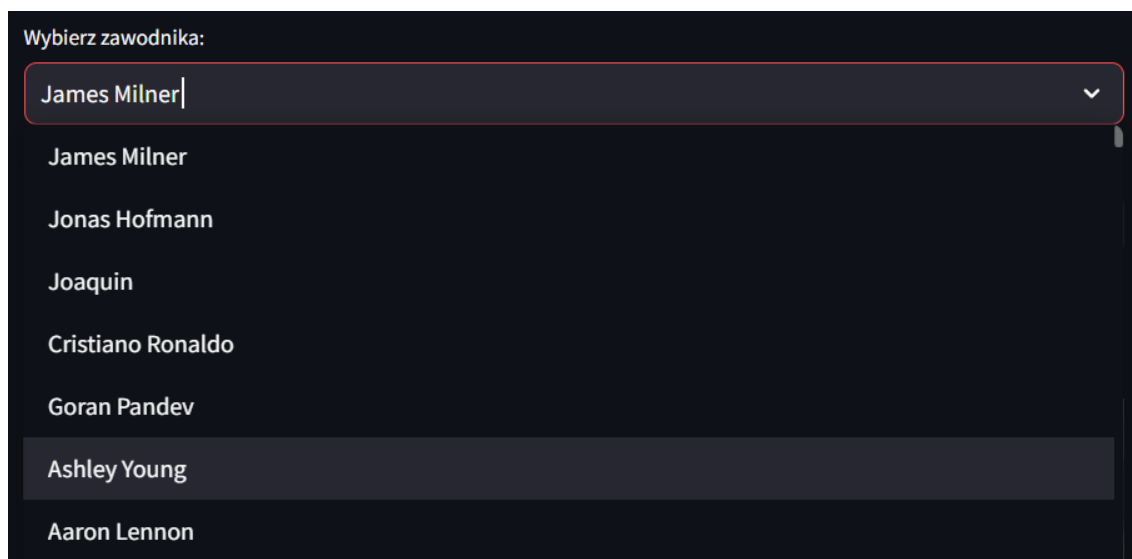
## 4.2. Statystyki

Podstrona "Statystyki" (rys. 4.9.) przedstawia analizę danych dotyczących wybranego zawodnika. Interfejs tej sekcji stawia na przejrzystość i prostotę.



Rys. 4.9. - Zakładka "Statystyki"

Pierwszym krokiem w procesie interakcji z zakładką jest wybór zawodnika z rozwijanej listy, zawiera unikalne nazwiska zawodników dostępne w załadowanym zbiorze danych (rys. 4.10.). Aplikacja pozwala na rozwinięcie listy i wpisanie nazwiska na klawiaturze, aby ułatwić wybór zawodnika. Po dokonaniu wyboru aplikacja wyświetla opcję wyboru sezonu, jeśli wybrany zawodnik ma dostępne dwa sezony w bazie danych.



Rys. 4.10. - Wybór zawodnika z paska rozwijalnego

Po wyborze zawodnika aplikacja wyświetla podstawowe informacje o piłkarzu (rys. 4.11.). Dane te obejmują szczegóły dotyczące pozycji na boisku, ligi, narodowości, wieku zawodnika. Informacje te są prezentowane w lewej kolumnie ekranu. Natomiast w prawej kolumnie wyświetlane jest zdjęcie zawodnika.



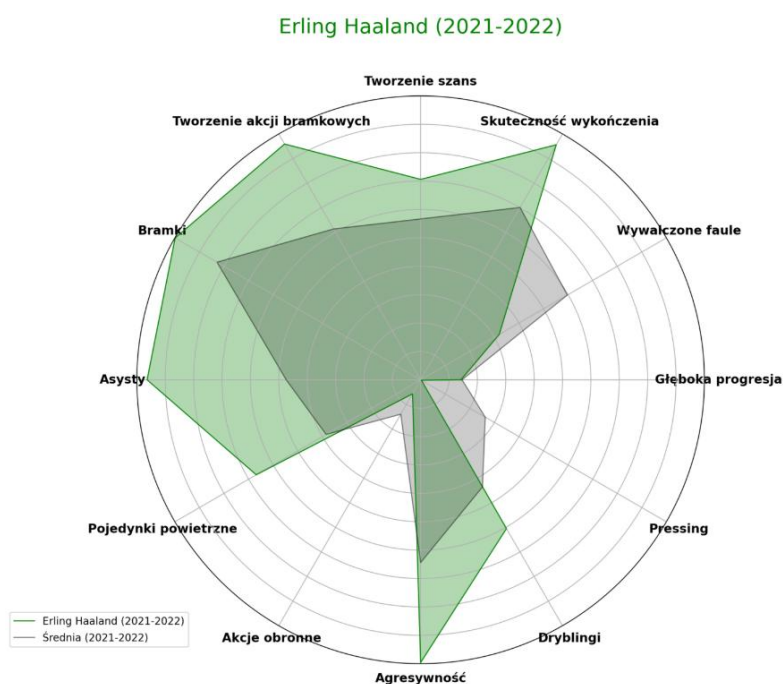
Rys. 4.11. - Informacje o zawodniku

Centralnym elementem zakładki są statystyki, które są dostępne po rozwinięciu sekcji "Szczegółowe statystyki zawodnika" (rys. 4.12.). Dane te obejmują bardziej szczegółowe statystyki, takie jak liczba strzałów na mecz, średnia odległość strzału czy procent celnych podań.



Rys. 4.12. - Szczegółowe statystyki zawodnika

Aby ułatwić analizę i porównanie wyników zawodnika, aplikacja generuje wizualizację w postaci wykresu radarowego (rys. 4.13.). Na wykresie prezentowane są wartości metryk dla wybranego zawodnika oraz wartości tych metryk dla przeciętnego zawodnika z tej samej ligi w tym samym sezonie. Obliczenie wskaźników dla przeciętnego zawodnika obejmowało filtrację odpowiednich piłkarzy i wykorzystanie funkcji `.mean()` (rys. 4.14.). Wykres ten jest szczególnie przydatny do oceny, w jakich obszarach zawodnik przewyższa ligowych rywali, a w jakich pozostaje poniżej średniej.



Rys. 4.13. - Wykres radarowy



```
# Filtracja danych dla średniego zawodnika w tej samej lidze, pozycji, sezonie i obliczenie wartości metryk (mean)
league_data = data[(data['Comp'] == player_data.iloc[0]['Comp']) &
                  (data['Pos'] == player_data.iloc[0]['Pos']) &
                  (data['season'] == season)]
league_avg_data = league_data[metrics].mean()
```

Rys. 4.14. - Obliczanie metryk dla przeciętnego zawodnika

Dodatkowo aplikacja generuje tabelę podsumowującą (rys. 4.15.), która zawiera wartości wybranych metryk zarówno dla analizowanego zawodnika, jak i dla przeciętnego gracza w lidze. Tabela pozwala na szybkie porównanie wyników, a także eksport danych do pliku .csv.

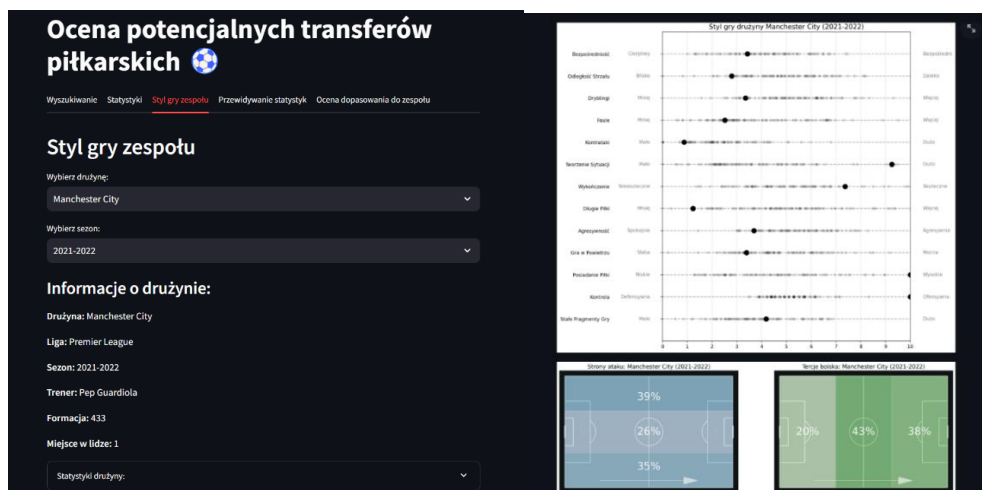
	Erling Haaland	Głęboka progresja	Wywalczone faule	Skuteczność wykończenia	Tworzenie szans
0	Erling Haaland	1.42	3.2	9.56	7.06
1	Przeciętny zawodnik	1.46	5.97	7.01	5.67

Rys. 4.15. - Tabela z wynikami

Podsumowując, zakładka "Statystyki" stanowi kompleksowe narzędzie do analizy piłkarskiej, integrujące dane tabelaryczne, wizualizacje i tekstowe opisy. Dzięki czytelnemu interfejsowi użytkownik może w łatwy sposób uzyskać pełen obraz statystyk zawodnika oraz porównać je z średnią ligową.

### 4.3. Styl gry drużyny

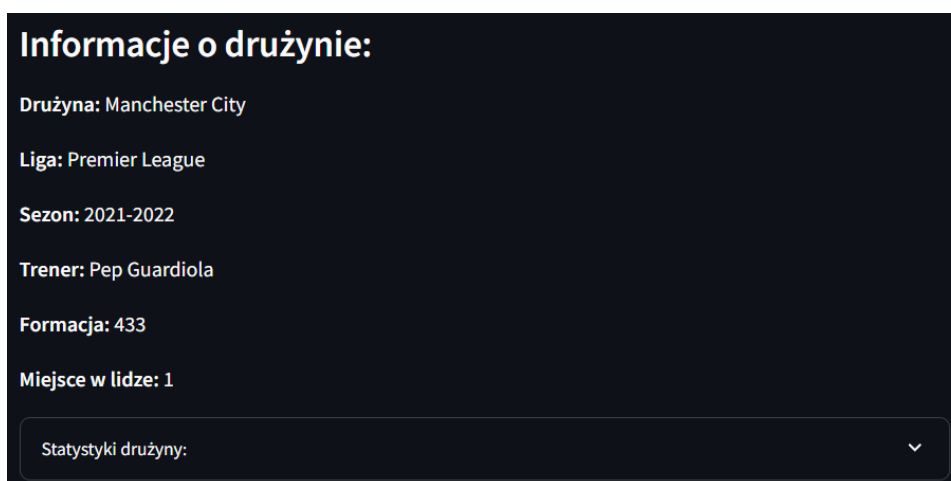
Karta "Styl gry zespołu" (rys. 4.16.) umożliwia użytkownikowi analizę taktyki i charakterystyki wybranej drużyny piłkarskiej. Interfejs stawia na przejrzysty sposób dostarczania zarówno ogólnych informacji o zespole, jak i wizualizacji jego stylu gry.



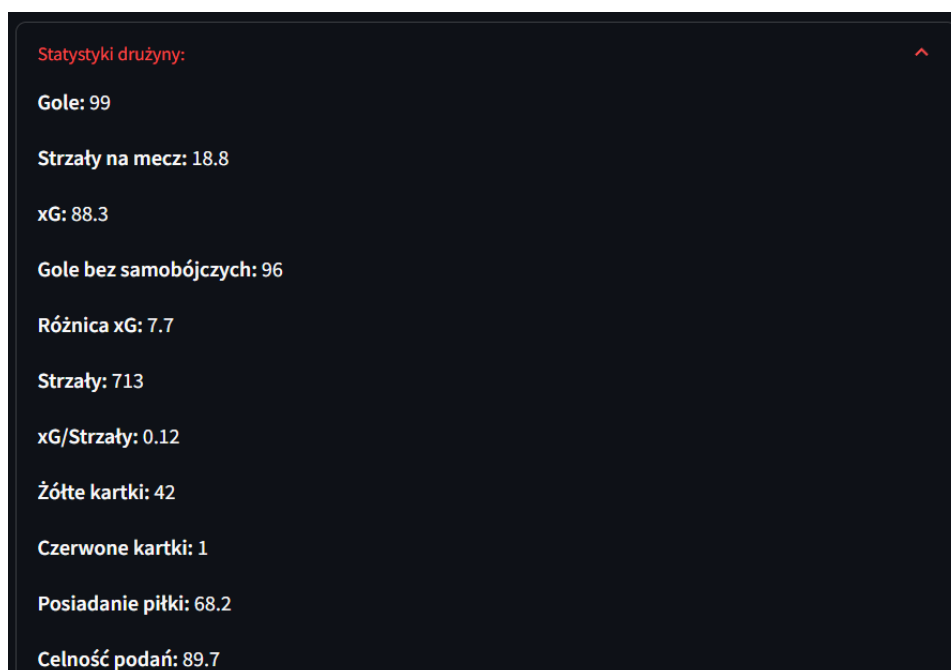
Rys. 4.16. - Zakładka "Styl gry zespołu"

Użytkownik rozpoczyna interakcję od wyboru drużyny z listy dostępnych zespołów. Po wybraniu drużyny aplikacja automatycznie dostosowuje listę dostępnych sezonów dla wybranego zespołu, pozwalając na szczegółowe analizowanie danych historycznych.

W centralnej części interfejsu wyświetlane są podstawowe informacje o drużynie (rys. 4.17.), takie jak jej nazwa, liga, trener, dominująca formacja, a także miejsce w lidze na koniec wybranego sezonu. Użytkownik może rozwinąć szczegółowe dane (rys. 4.18.), aby zapoznać się z kluczowymi wskaźnikami, takimi jak liczba oddanych strzałów na mecz, posiadanie piłki, celność podań czy średnia liczba fauli na mecz.

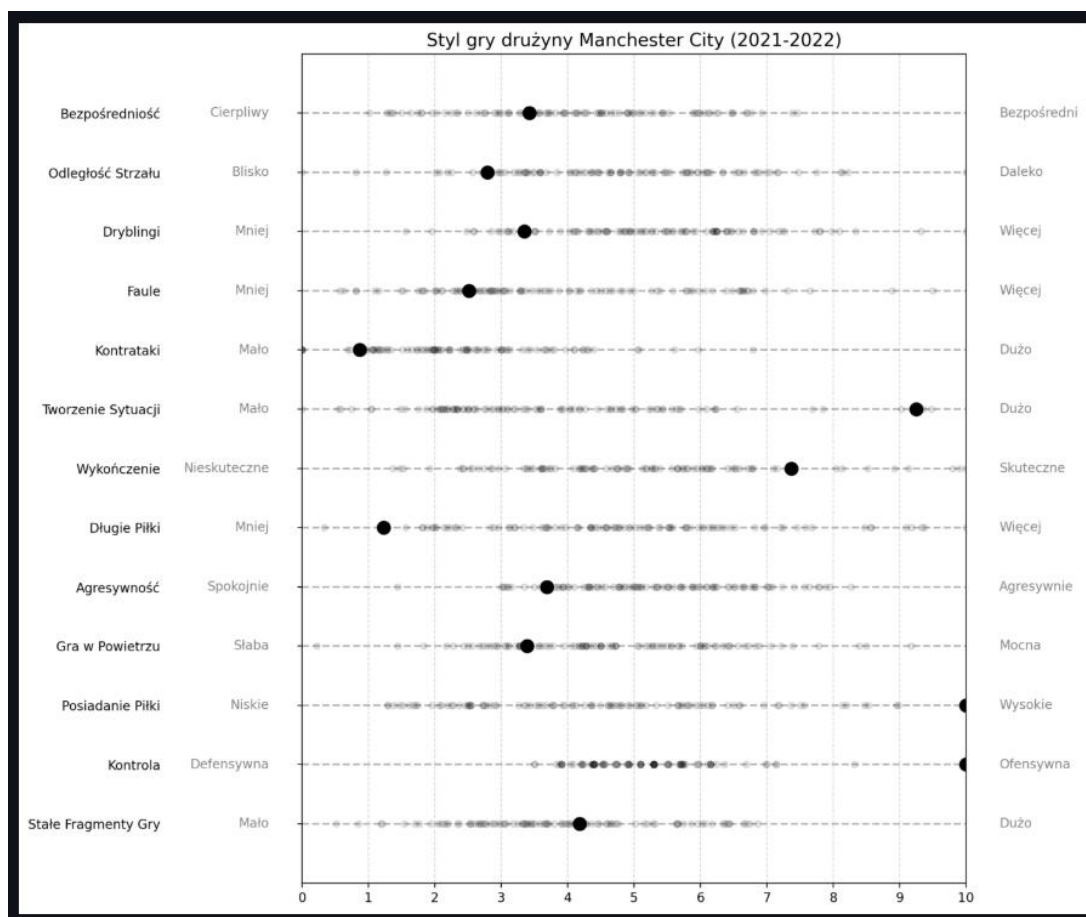


Rys. 4.17. - Informacje o drużynie



Rys. 4.18. - Szczegółowe statystyki drużyny

Jednym z głównych elementów zakładki jest wizualizacja stylu gry zespołu na wykresie suwakowym (rys. 4.19.). Inspiracją do takiego przedstawienia danych była publikacja na blogu Medium [13]. Funkcja `plot_team_sliders` (rys. 4.20.) filtruje dane dla wybranej drużyny i sezonu, a następnie rysuje wykres z punktami reprezentującymi wartości metryk stylu gry drużyn. Użytkownik może zobaczyć, jak drużyna wypada w takich aspektach jak bezpośredniość gry, tworzenie sytuacji, agresywność czy gra w powietrzu, w porównaniu do innych zespołów z tej samej ligi i sezonu. Każdy z suwaków przedstawia nie tylko dane analizowanej drużyny, ale także rozkład wyników pozostałych zespołów (w postaci mniejszych, półprzezroczystych kropek na wykresie), co pozwala na łatwe porównanie metryk.



Rys. 4.19. - Wykres metryk zespołu

```

# Funkcja do rysowania suwaków stylu gry zespołu
def plot_team_sliders(team_name, season, team_stats, style_cols):

    season_data = team_stats[team_stats['Season'] == season] # Filtrowanie danych dla wybranego sezonu

    team_data = season_data[season_data['Team'] == team_name].iloc[0] # Pobieranie danych dla wybranego klubu

    fig, ax = plt.subplots(figsize=(12, 10))
    plt.subplots_adjust(left=0.3, right=0.9, top=0.95, bottom=0.05)

    y_positions = np.linspace(0, len(style_cols) - 1, len(style_cols))[::-1]

    # Rysowanie suwaków
    for y, col in zip(y_positions, style_cols):
        ax.plot([0, 10], [y, y], 'k--', alpha=0.3)

        # Rysowanie kropek dla innych drużyn z tego samego sezonu
        for _, row in season_data.iterrows():
            if row['Team'] != team_name:
                ax.plot(row[col], y, 'ko', markersize=5, alpha=0.1)

        # Rysowanie kropki dla wybranego klubu
        ax.plot(team_data[col], y, 'ko', markersize=10, label=team_name)

        # Dodanie opisów suwaków
        if col in slider_labels:
            ax.text(-0.5, y, slider_labels[col][0], va='center', ha='right', fontsize=10, color='gray')
            ax.text(10.5, y, slider_labels[col][1], va='center', ha='left', fontsize=10, color='gray')

    # Ustawienia osi
    ax.set_xlim(0, 10)
    ax.set_ylim(-1, len(style_cols))
    ax.set_yticks(y_positions)

    yticks = ax.get_yticklabels()
    for tick in yticks:
        tick.set_x(-0.2)

    translated_style_cols = [style_column_mapping.get(col, col) for col in style_cols]
    ax.set_yticklabels(translated_style_cols)

    ax.set_xticks(range(0, 11, 1))
    ax.set_title(f'Styl gry drużyny {team_name} ({season})', fontsize=14)

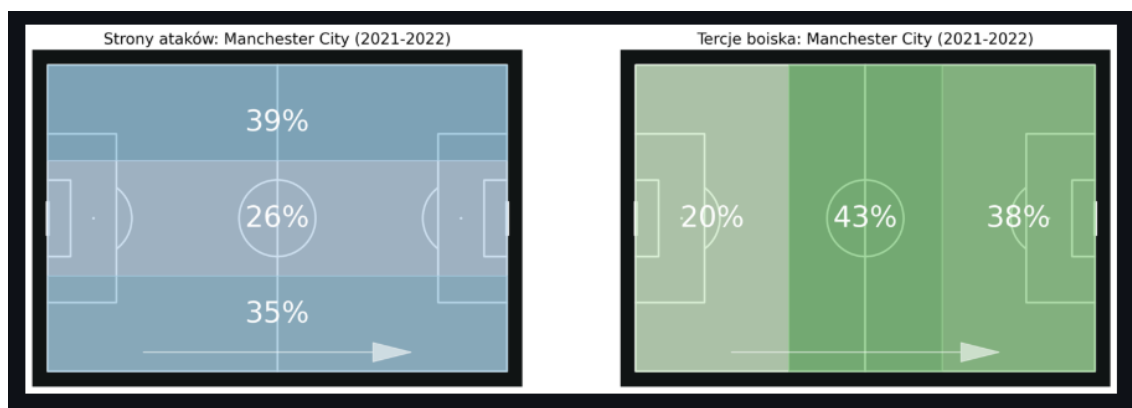
    plt.grid(axis='x', linestyle='--', alpha=0.5)

    # Zwrócenie wykresu
    return fig

```

Rys. 4.20. - Kod funkcji do rysowania wykresu suwakowego

Wizualizacja na dwóch schematach boisk jest kolejnym ważnym komponentem zakładki (rys. 4.21.). Boisko po lewej stronie obrazuje rozkład ataków zespołu z podziałem na lewą, środkową i prawą stronę boiska, używając kolorów wskazujących intensywność ataków w danych sektorach. Drugie boisko prezentuje, w jakich tercjach boiska drużyna była najbardziej aktywna, podkreślając preferencje zespołu w prowadzeniu gry. Strzałki w dolnej części boiska wskazują kierunek ataku.



Rys. 4.21. - Grafiki prezentujące strony ataków i posiadanie piłki w poszczególnych tercjach boiska

Zakładka "Styl gry zespołu" stanowi zintegrowane narzędzie analityczne, które łączy w sobie dane statystyczne, dynamiczne wykresy i graficzne przedstawienie taktyki. Umożliwia użytkownikowi kompleksowy wgląd w charakterystykę gry zespołu w wybranym sezonie.

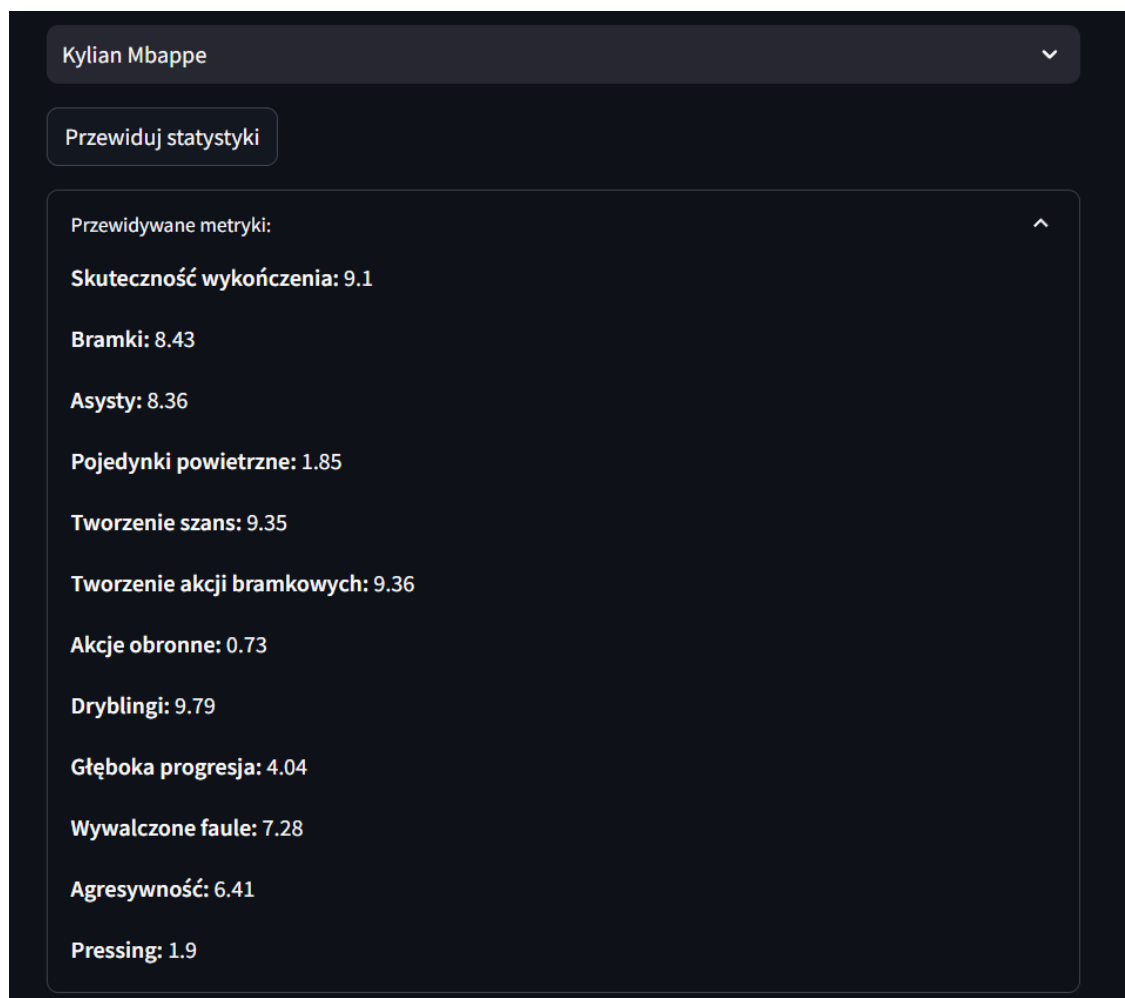
#### 4.4. Przewidywanie statystyk

Zakładka "Przewidywanie statystyk" (rys. 4.22.) umożliwia użytkownikowi dokonanie predykcji statystyk wybranego zawodnika na sezon 2024-2025 korzystając z danych historycznych oraz modeli predykcyjnych.

Rys. 4.22. - Zakładka "Przewidywanie statystyk"

Użytkownik rozpoczyna analizę od wyboru zawodnika z rozwijanej listy. Lista zawiera nazwiska wszystkich dostępnych piłkarzy. Po dokonaniu wyboru użytkownik ma możliwość uruchomienia procesu predykcji poprzez kliknięcie przycisku.

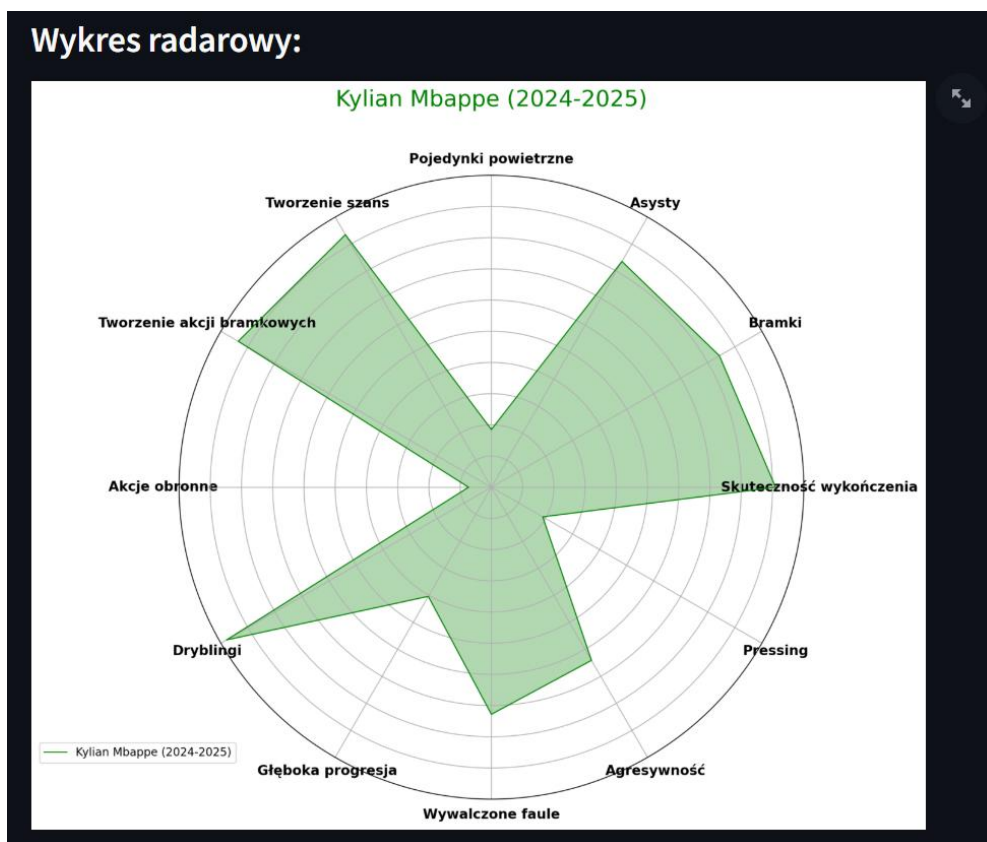
Po przeprowadzeniu predykcji za pomocą modeli aplikacja przedstawia wyniki w formie rozwijanego menu (rys. 4.23.). Aplikacja wyświetla takie metryki jak skuteczność wykończenia, dryblingi czy wywalczone faule.



Rys. 4.23. - Przewidywane metryki

Jednym z najważniejszych elementów zakładki jest radarowy wykres wizualizujący przewidywane statystyki dla wybranego zawodnika (rys. 4.24.). Ten wykres pozwala szybko ocenić, w których aspektach zawodnik będzie kluczowy w nadchodzącym sezonie.

### Wykres radarowy:



Rys. 4.24. - Wykres radarowy

## 4.5. Ocena dopasowania do zespołu

Sekcja "Ocena dopasowania do drużyny" (rys. 4.25.) pozwala na ocenę, jak dobrze dany zawodnik pasuje do wybranej drużyny na podstawie porównania stylu gry obu zespołów. Interfejs tej zakładki jest zaprojektowany tak, aby użytkownik mógł szybko uzyskać ocenę dopasowania, szczegółowe komentarze oraz przewidywane statystyki zawodnika w kontekście drużyny.

Rys. 4.25. - Zakładka "Ocena dopasowania do zespołu"

Użytkownik wybiera z rozwijalnych list nazwisko piłkarza oraz drużynę, dla której chciałby ocenić dopasowanie zawodnika. Po naciśnięciu przycisku „Sprawdź” aplikacja przeprowadza obliczenia i prezentuje wynik oceny dopasowania.

Ocena dopasowania opiera się na obliczeniu odległości euklidesowej między metrykami drużyny aktualnego zawodnika, a metrykami wybranej drużyny. Kluczowe metryki, które najlepiej obrazują styl gry zespołu, takie jak Bezpośredniość, Posiadanie piłki, czy Kontrola są porównywane, a wynik tej odległości jest następnie normalizowany na skalę 1-6. Dodatkowo, jeśli formacje obu drużyn są zgodne, wynik dopasowania jest zwiększany o 0,5 punktu. Na podstawie tej oceny generowany jest komentarz, który opisuje, jak łatwa będzie adaptacja zawodnika do nowego zespołu.

Po obliczeniu wyniku oceny dopasowania, aplikacja generuje kolorową ikonę z oceną wraz z komentarzem (rys. 4.26.), co pozwala na szybkie rozpoznanie wyniku. Kolor ikony uzależniony jest od wartości (zielony dla oceny 6, czerwony dla 1). Jeżeli formacja drużyny zawodnika zgadza się z formacją drużyny docelowej aplikacja zwraca odpowiednią informację.

Rys. 4.26. - Ocena dopasowania



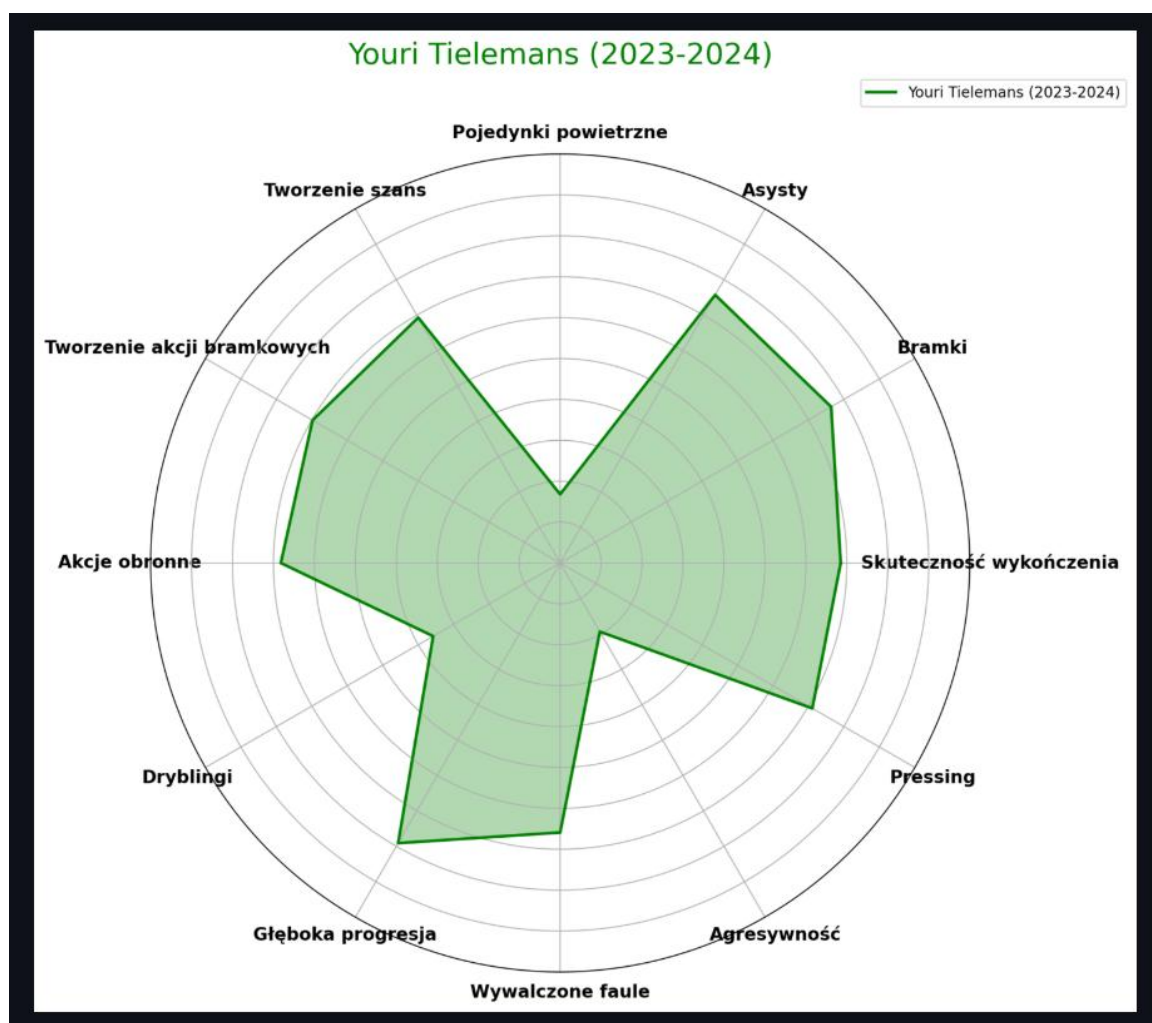
Kolejnym elementem jest skalowanie przewidywanych statystyk zawodnika w zależności od oceny dopasowania. Jeśli dopasowanie jest wysokie, statystyki nie są zmieniane, ale jeśli wynik oceny jest niski, wartości statystyk są zmniejszane o określony procent, co ma odzwierciedlać trudniejszy proces adaptacji w drużynie o zupełnie innym stylu gry.

Poniżej aplikacja pozwala na sprawdzenie wartości metryk zawodnika, takich jak tworzenie akcji bramkowych, dryblingi, czy głęboka progresja. Te dane pomagają użytkownikowi lepiej zrozumieć potencjał zawodnika w kontekście transferu do nowego zespołu. Statystyki te są przedstawione w rozwijanym elemencie (rys. 4.27.).



Rys. 4.27. - Przewidywane statystyki

W dolnej części zakładki aplikacja generuje wykres radarowy na podstawie przeliczonych metryk (rys. 4.5.4.).



Rys. 4.28. - Wykres radarowy

## 5. Trenowanie modeli

Na początku procesu trenowania modeli predykcyjnych wczytywane są dane z pliku .csv przy wykorzystaniu biblioteki Pandas, zawierającego statystyki piłkarzy. W kolejnych krokach, dane są filtrowane, aby wybrać tylko te kolumny, które będą używane do trenowania. Z danych zostają wybrane takie cechy jak liczba bramek, asyst, statystyki defensywne oraz obliczone metryki piłkarzy, jak np. głęboka progresja, wywalczone faule czy dryblingi. Dane te zostają zapisane do zmiennej df\_selected.

Następnie przypisywane są odpowiednie zestawy cech (features) oraz zmiennej docelowej (target), której wartość chcemy przewidywać. Dla każdej metryki, jak np. Finishing\_Efficiency\_M, Goals\_M, czy Assists\_M, przygotowywane są odpowiednie zestawy cech (rys. 5.1.). Wybrano cechy, które mają istotny wpływ na wynik konkretnej statystyki i zostały określone na podstawie analizy danych.

```
# Filtrowanie wybranych kolumn
selected_columns = ['Goals', 'Assists', 'Deep_Progression_M', 'Fouls_Won_M', 'Finishing_Efficiency_M',
                   'Chance_Creation_M', 'Goal_Creation_M', 'Aerial_Duels_M', 'Defensive_Actions_M',
                   'Aggressiveness_M', 'Dribbles_M', 'Pressure_M', 'Goals_M', 'Assists_M',
                   'CarProg', 'PasProg', 'Fld', 'Shots', 'SCA', 'GCA', 'AerWon', 'AerLost',
                   'Tkl', 'TklWon', 'Int', 'Blocks', 'Recov', 'Clr', 'CrdY', 'CrdR',
                   'DriSucc', 'DriAtt', 'TklDef3rd', 'TklMid3rd', 'TklAtt3rd', 'Age']

df_selected = df[selected_columns]

# Definicja cech dla modelu dotyczącego skuteczności wykończenia
features_finishing = ['Goals', 'Assists', 'Deep_Progression_M', 'Fouls_Won_M', 'Chance_Creation_M',
                     'Goal_Creation_M', 'Aerial_Duels_M', 'Defensive_Actions_M', 'Aggressiveness_M',
                     'Dribbles_M', 'Pressure_M', 'CarProg', 'PasProg', 'Fld', 'Shots', 'SCA', 'GCA', 'Age']

# Definicja cech dla modelu dotyczącego goli
features_goals = ['Goals', 'Assists', 'Deep_Progression_M', 'Fouls_Won_M', 'Chance_Creation_M',
                  'Goal_Creation_M', 'Aerial_Duels_M', 'Defensive_Actions_M', 'Aggressiveness_M', 'Dribbles_M',
                  'Pressure_M', 'CarProg', 'PasProg', 'Fld', 'Shots', 'SCA', 'GCA', 'Age']

# Definicja cech dla modelu dotyczącego asyst
features_assists = ['Assists', 'Goals', 'Deep_Progression_M', 'Fouls_Won_M', 'Chance_Creation_M',
                   'Goal_Creation_M', 'Aerial_Duels_M', 'Defensive_Actions_M', 'Aggressiveness_M', 'Dribbles_M',
                   'Pressure_M', 'CarProg', 'PasProg', 'Fld', 'Shots', 'SCA', 'GCA', 'Age']

# Definicja cech dla modelu dotyczącego pojedynków powietrznych
features_aerial_duels = selected_columns

# Definicja cech dla pozostałych modeli
selected_columns = ['Goals', 'Assists', 'Deep_Progression_M', 'Fouls_Won_M', 'Finishing_Efficiency_M',
                   'Chance_Creation_M', 'Goal_Creation_M', 'Aerial_Duels_M', 'Defensive_Actions_M',
                   'Aggressiveness_M', 'Dribbles_M', 'Pressure_M', 'Goals_M', 'Assists_M',
                   'CarProg', 'PasProg', 'Fld', 'Shots', 'SCA', 'GCA', 'AerWon', 'AerLost',
                   'Tkl', 'TklWon', 'Int', 'Blocks', 'Recov', 'Clr', 'CrdY', 'CrdR',
                   'DriSucc', 'DriAtt', 'TklDef3rd', 'TklMid3rd', 'TklAtt3rd', 'Age']
```

Rys. 5.1. - Dane wykorzystane do trenowania poszczególnych modeli predykcyjnych

Po dokonaniu wyboru cech, dane są dzielone na zbiór treningowy i testowy za pomocą funkcji train\_test\_split z biblioteki sklearn.model\_selection. Standardowo 80% danych jest wykorzystywane do trenowania modelu, a 20% do jego testowania. Taki

podział zapewnia, że model będzie oceniany na wcześniej niewidzianych danych, co pozwala na jego ocenę.

Zanim model zostanie wytrenowany, przeprowadzana jest standardyzacja cech, aby przekształcić dane wejściowe do jednolitej skali. W tym celu wykorzystywana jest klasa `StandardScaler`, co pozwala na standaryzację danych, czyli ustawienie ich średniej na 0 i odchylenia standardowego na 1. Jest to kluczowe, ponieważ modele uczenia maszynowego, szczególnie te oparte na drzewach decyzyjnych, mogą działać lepiej, gdy cechy mają podobne skale.

Model jest trenowany z wykorzystaniem algorytmu Random Forest. Składa się on z wielu drzew decyzyjnych, a jego działanie polega na agregacji wyników z tych drzew, co pozwala uzyskać stabilniejsze i bardziej dokładne predykcje. Parametry modelu, takie jak liczba drzew (`n_estimators`), maksymalna głębokość drzewa (`max_depth`) oraz inne (rys. 5.2.), zostały dostosowane tak, aby model osiągnął jak najlepsze wyniki.

```
# Model Random Forest z parametrami
rf = RandomForestRegressor(n_estimators=n_estimators,
                           max_depth=max_depth,
                           min_samples_split=min_samples_split,
                           min_samples_leaf=min_samples_leaf,
                           random_state=42)

# Parametry dla poszczególnych zmiennych docelowych

# Parametry dla Finishing_Efficiency_M
train_random_forest('Finishing_Efficiency_M', features_finishing, n_estimators=50, max_depth=4, min_samples_split=2, min_samples_leaf=2)

# Parametry dla Goals_M
train_random_forest('Goals_M', features_goals, n_estimators=25, max_depth=2, min_samples_split=2, min_samples_leaf=4)

# Parametry dla Assists_M
train_random_forest('Assists_M', features_assists, n_estimators=25, max_depth=2, min_samples_split=2, min_samples_leaf=4)

# Parametry dla Aerial_Duels_M
train_random_forest('Aerial_Duels_M', features_aerial_duels, n_estimators=30, max_depth=3, min_samples_split=2, min_samples_leaf=4)

# Parametry dla poszczególnych metryk
metrics_rf = {
    'Chance_Creation_M': {'n_estimators': 150, 'max_depth': 8, 'min_samples_split': 2, 'min_samples_leaf': 4, 'max_features': 'sqrt', 'bootstrap': True},
    'Goal_Creation_M': {'n_estimators': 150, 'max_depth': 12, 'min_samples_split': 3, 'min_samples_leaf': 5, 'max_features': 'sqrt', 'bootstrap': True},
    'Defensive_Actions_M': {'n_estimators': 200, 'max_depth': 12, 'min_samples_split': 2, 'min_samples_leaf': 3, 'max_features': 'sqrt', 'bootstrap': True},
    'Dribbles_M': {'n_estimators': 120, 'max_depth': 10, 'min_samples_split': 4, 'min_samples_leaf': 2, 'max_features': 'log2', 'bootstrap': False},
    'Deep_Progression_M': {'n_estimators': 200, 'max_depth': 12, 'min_samples_split': 4, 'min_samples_leaf': 4, 'max_features': 'sqrt', 'bootstrap': True},
    'Fouls_Won_M': {'n_estimators': 150, 'max_depth': 12, 'min_samples_split': 2, 'min_samples_leaf': 5, 'max_features': 'sqrt', 'bootstrap': True},
    'Aggressiveness_M': {'n_estimators': 200, 'max_depth': 16, 'min_samples_split': 6, 'min_samples_leaf': 6, 'max_features': 'sqrt', 'bootstrap': True},
    'Pressure_M': {'n_estimators': 150, 'max_depth': 12, 'min_samples_split': 2, 'min_samples_leaf': 3, 'max_features': 'sqrt', 'bootstrap': True},
}
```

Rys. 5.2. - Dobór parametrów modeli

Po wytrenowaniu modelu na zbiorze treningowym, następuje jego ocena na zbiorze testowym (rys. 5.3.). W tym celu model dokonuje predykcji na nieznanymi wcześniej danych testowych. Jako miary oceny jakości modelu wykorzystano Mean Squared Error (MSE) oraz R-squared ( $R^2$ ). MSE mierzy średnią kwadratową różnicę między rzeczywistymi, a przewidywanymi wartościami, podczas gdy  $R^2$  informuje o tym, jak dobrze model dopasowuje się do danych – im bliższa 1 wartość  $R^2$ , tym model jest bardziej trafny.

```
MSE dla Finishing_Efficiency_M: 0.7505117606503686
R^2 dla Finishing_Efficiency_M: 0.9390726742103497
Model Finishing_Efficiency_M zapisany jako Player_stats_models/Finishing_Efficiency_M_rf_model.pkl
```

Rys. 5.3. - Ocena modelu

Po ocenie jakości modelu, przyszedł czas na jego zapisanie (rys. 5.4.). Wytrenowany model jest zapisywany do pliku z wykorzystaniem biblioteki joblib. Dzięki temu model może być wykorzystany w praktycznych zastosowaniach bez konieczności każdorazowego uczenia go na nowo, co pozwala zaoszczędzić zasoby obliczeniowe oraz przyspieszyć proces dokonywania predykcji.

```
# Zapisanie modelu
model_filename_rf = f"Player_stats_models/{metric}_rf_model.pkl"
joblib.dump(model_rf, model_filename_rf)
print(f"Model {metric} zapisany jako {model_filename_rf}")
```

Rys. 5.4. - Zapisanie modeli

Parametry, takie jak liczba drzew czy ich głębokość, są dobierane zgodnie z metryką, którą chcemy przewidywać. W zależności od modelu są one modyfikowane, aby osiągnąć jak najlepsze wyniki.

Podsumowując, proces trenowania modelu wymaga odpowiedniego przygotowania danych, doboru cech, podziału na zbiory treningowe i testowe, przetwarzania danych, trenowania modelu, oceny jakości oraz jego zapisania.

## 6. Wyniki i ograniczenia projektu

W ramach projektu opracowano modele predykcyjne oparte na statystykach piłkarzy. Celem było stworzenie narzędzia umożliwiającego użytkownikowi przewidywanie różnych wskaźników. Poniżej przedstawiono wyniki oraz omówiono ograniczenia projektu.

### 6.1. Wyniki modeli predykcyjnych

Metryki jakości modeli (MSE,  $R^2$ ) zostały przedstawione w poniższej tabeli.

tabela 6.1. – wartości metryk jakości modeli

Model	MSE	$R^2$
Skuteczność wykończenia	0.319	0.964
Gole	0.376	0.970
Asysty	0.425	0.967
Pojedynki powietrzne	0.095	0.989
Tworzenie szans	0.319	0.964
Tworzenie akcji bramkowych	0.515	0.950
Akcje defensywne	0.233	0.972
Dryblingi	0.145	0.984
Głęboka progresja	0.579	0.933
Wywalczone faule	0.548	0.932
Agresywność	0.739	0.920
Pressing	0.315	0.960

Przeprowadzone trenowanie modelu Random Forest przyniosło bardzo dobre rezultaty, co zostało potwierdzone wartościami współczynnika determinacji  $R^2$  bliskimi 1 oraz niskimi wartościami średniego błędu kwadratowego (MSE).

Najwyższą jakość predykcji osiągnięto dla zmiennych związanych z pojedynkami powietrznymi oraz dryblingiem, co świadczy o precyzyjnym uchwyceniu kluczowych zależności. Modele dla wskaźników ofensywnych, takich jak metryka bramek (Goals\_M,  $R^2 = 0.9697$ ) oraz asyst (Assists\_M,  $R^2 = 0.9670$ ), również osiągnęły wysoką skuteczność.

Dla pozostałych zmiennych odnotowano nieznacznie niższe wyniki i wskazują na dobrą jakość predykcji. Najniższą wartość  $R^2$  równą 0.920 osiągnął model dotyczący predykcji metryki agresywności, co nadal jest bardzo obiecującym wynikiem.

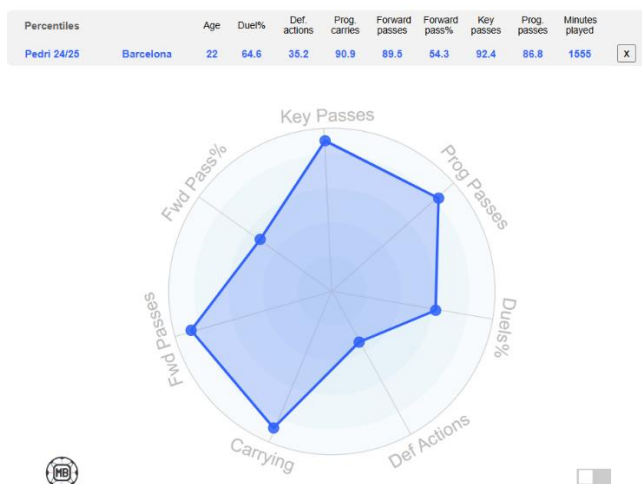
Przedstawione wyniki potwierdzają, że model Random Forest potrafi wspierać ocenę potencjalnych transferów zawodników i w poprawny sposób przewidywać metryki zawodnika w poszczególnych obszarach gry. Precyzyjne przewidywanie kluczowych

wskaźników dostarcza wartościowych informacji na temat rzeczywistego potencjału piłkarzy.

Podsumowując, modele Random Forest skutecznie odwzorowują analizowane zależności i mogą być zastosowane w praktycznych analizach, pod warunkiem uwzględnienia dostosowania danych wejściowych.

## 6.2. Porównanie predykcji do rzeczywistych statystyk

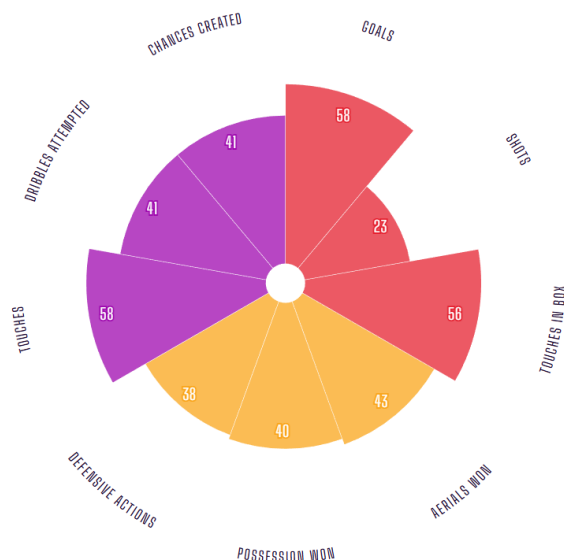
W analizie sportowej istnieją zaawansowane narzędzia opracowane przez chociażby takie firmy takie jak DataMB czy Opta, które pozwalają na generowanie wykresów radarowych (rys. 6.1., rys. 6.2.). Takie wykresy były inspiracją do wizualnego przedstawienia statystyk zawodników w projekcie. Wykresy te są przydatne w prezentowaniu i porównywaniu różnych aspektów piłkarzy. Dzięki nim możliwe jest dokładne przedstawienie danych statystycznych w czytelnej i intuicyjnej formie. W momencie pisania pracy rozgrywki ligowe są na półmetku, co pozwala na wstępne porównanie statystyk przewidywanych w projekcie do aktualnych statystyk piłkarzy. Wykresy radarowe dostarczane przez Opta, są najbardziej zbliżone do tych zaimplementowanych w aplikacji. Metryki takie jak Chances Created i Tworzenie szans, Defensive Actions i Akcje obronne, Goals i Bramki, Dribbles Attempted i Dryblingi, Aerial Won i Pojedynki powietrzne są do siebie zbliżone, co pozwala na zweryfikowanie przynajmniej niektórych predykcji.



Rys. 6.1. - Wykres radarowy od DataMB

## YOURI TIELEMANS

Midfielder template. Percentile rank vs midfielders with at least 500 minutes played.



Rys. 6.2. - Wykres radarowy od Opta

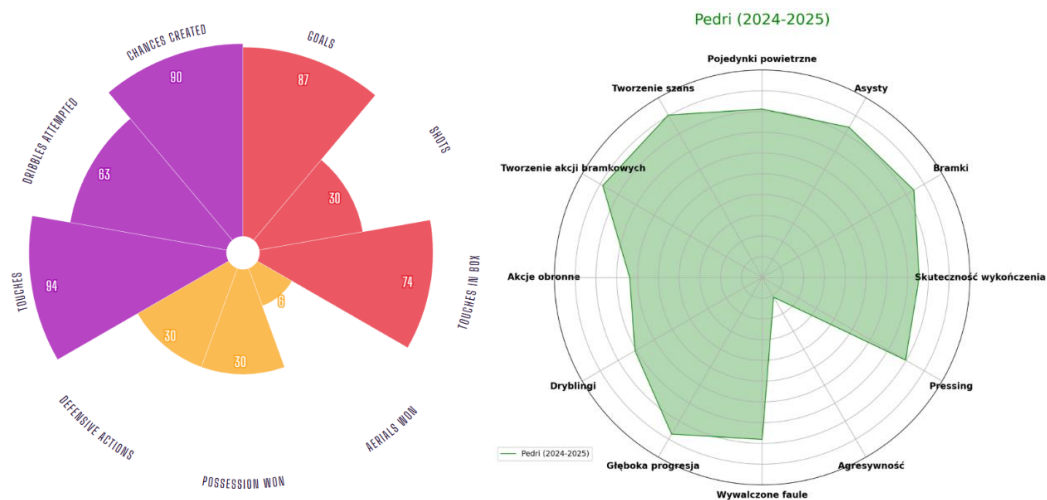
## Pedri

tabela 6.2. – porównanie wartości predykcji do wskaźników od OPTA dla Pedriego

Metryka	OPTA	Aplikacja
Tworzenie szans	90	9.04
Bramki	87	8.43
Dryblingi	63	7.07
Akcje obronne	30	6.38
Pojedynki powietrzne	6	8.12

W przypadku takiego zawodnika jak Pedri aplikacja bardzo dobrze radzi sobie z predykcją metryk odpowiedzialnych za tworzenie szans, bramki oraz dryblingi (rys. 6.3.). W przypadku takich statystyk jak akcje defensywne czy zwłaszcza pojedynki powietrzne predykcja jest wyraźnie niedokładna. Może to wynikać ze zmiany trenera w zespole, w którym gra Pedri, która nie jest uwzględniana w projekcie, a co za tym idzie innych założeń taktycznych dla piłkarza.





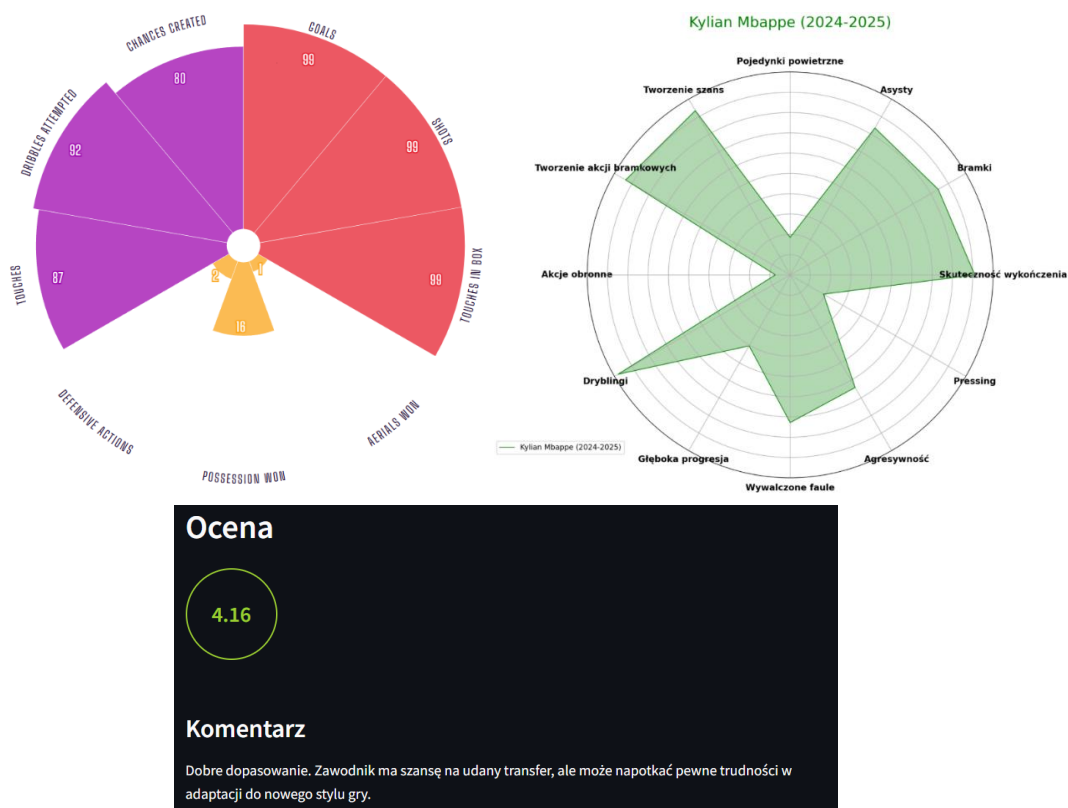
Rys. 6.3. - Wykresy radarowe dla Pedriego (aktualne wyniki oraz predykcja aplikacji)

## Kylian Mbappe

tabela 6.3. – porównanie wartości predykcji do wskaźników od OPTA dla Kyliana Mbappe

Metryka	OPTA	Aplikacja
Tworzenie szans	80	9.35
Bramki	99	8.43
Dryblingi	92	9.79
Akcje obronne	2	0.73
Pojedynki powietrzne	1	1.85

Oceniając przewidywanie statystyk dla Kyliana Mbappe można zaobserwować bardzo dokładne predykcje wybranych metryk (rys. 6.4.). Pomimo, że zawodnik zmienił klub, aplikacja trafnie oceniła jego statystyki. Sama ocena dopasowania do Realu Madryt, do którego trafił zawodnik wynosi ponad 4 (dobre dopasowanie), co również sugeruje, że zawodnik może stosunkowo szybko wkomponować się w nowy zespół.



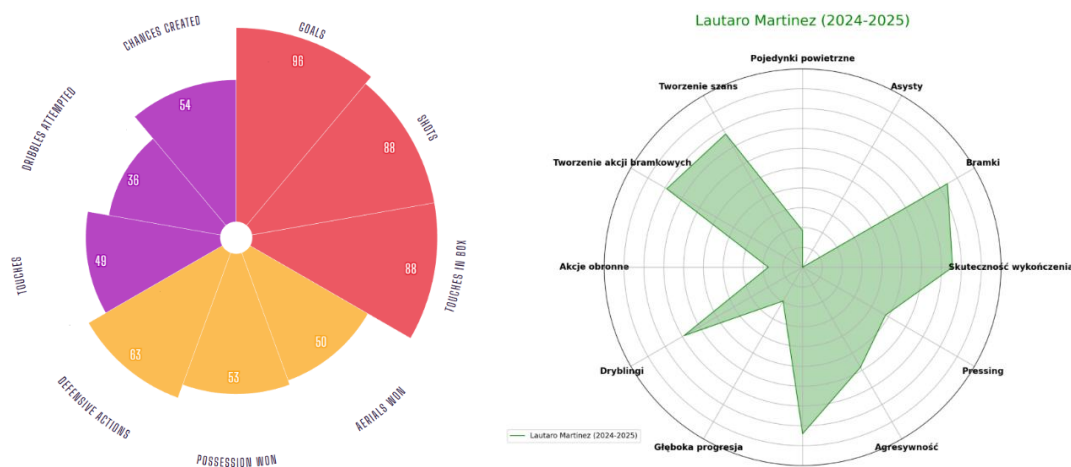
Rys. 6.4. - Wykresy radarowe dla Kyliana Mbappe (aktualne wyniki oraz predykcja aplikacji) i ocena dopasowania do zespołu

## Lautaro Martinez

tabela 6.4. – porównanie wartości predykcji do wskaźników od OPTA dla Lautaro Martineza

Metryka	OPTA	Aplikacja
Tworzenie szans	54	7.76
Bramki	96	8.43
Dryblingi	36	6.91
Akcje obronne	63	1.73
Pojedynki powietrzne	50	1.85

Aplikacja wykazuje niższą dokładność predykcji dla Lautaro Martineza (6.5.). Zauważalne są duże rozbieżności w takich metrykach jak akcje obronne, pojedynki powietrzne i dryblingi. Różnica przy wartości akcji obronnych może wynikać z odmiennej interpretacji danych i braku podziału na akcje obronne oraz pressing, tak jak w projekcie. Przykład tego zawodnika ilustruje, że model nie zawsze przewiduje dokładnie wyniki i wskazuje na znaczący potencjał do poprawy narzędzia.



Rys. 6.5. - Wykresy radarowe dla Lautaro Martineza (aktualne wyniki oraz predykcja aplikacji)

### 6.3. Ograniczenia projektu

Modele predykcyjne opierają się na dostępnych danych historycznych. Brak pełnych i reprezentatywnych danych może wpływać na jakość predykcji. Dynamika składów drużyn, forma zawodników czy kontuzje nie są uwzględniane, co prowadzi do nieoczekiwanych rezultatów.

Zmiany trenera oraz taktyki drużyny, mające znaczący wpływ na występy zawodników, nie są uwzględniane w modelach predykcyjnych. Takie czynniki mogą znacząco wpływać na precyzyjność modeli.

Różnice w interpretacji danych i obliczaniu metryk statystycznych mogą wpływać na dokładność porównań i predykcji. Różne źródła danych mogą inaczej definiować wskaźniki, takie jak akcje obronne, pojedynki powietrzne, czy pressing, co prowadzi do niejednorodności w analizach. Tego rodzaju rozbieżności utrudniają precyzyjne porównywanie wyników i może to wymagać dalszej standaryzacji modeli predykcyjnych.

Aplikacja docelowo miała oceniać transfery w skali 1-6 na podstawie statystyk zawodnika oraz drużyny, w której grał w poprzednim sezonie, statystyk drużyny docelowej, ceny i rodzaju transakcji (wypożyczenie/transfer definitywny). Ze względu na ograniczoną ilość danych o transferach historycznych (około 300 wierszy), model zwracał taką samą ocenę dla każdego transferu, niezależnie od kwoty i rodzaju transakcji, mimo że wyniki oceny modelu były bardzo wysokie.

## 7. Wnioski

Praca inżynierska związana z opracowaniem modeli przewidujących statystyki piłkarzy dostarczyła obiecujących rezultatów, ale także wskazała obszary wymagające dalszej poprawy.

Modele oparte na algorytmie Random Forest wykazały wysoką skuteczność w przewidywaniu wielu kluczowych wskaźników. Szczególnie dobre wyniki uzyskano dla zmiennych związanych z bramkami oraz tworzeniem szans, co wskazuje na zdolność modelu do precyzyjnego odwzorowania zależności w danych.

Pomimo wysokiej jakości predykcji dla większości zmiennych, niektóre metryki, takie jak akcje defensywne czy pojedynki powietrzne, wykazały nieoczekiwane rezultaty predykcji. Analiza wyników wskazuje, że różnice te mogą wynikać z czynników zewnętrznych, takich jak zmiana trenera, taktyki drużynowej lub odmienne interpretacje danych statystycznych.

Modele predykcyjne silnie opierają się na dostępnych danych historycznych. Brak pełnych, reprezentatywnych danych lub różnice w definicjach wskaźników statystycznych mogą prowadzić do spadku dokładności predykcji. Wykorzystanie bardziej rozbudowanej bazy danych, obejmującej statystyki z kilku lub nawet kilkunastu sezonów, mogłoby znacząco zwiększyć dokładność i uniwersalność przewidywań.

Wykorzystanie wykresów radarowych w wizualizacji statystyk zawodników było krokiem w stronę nowoczesnej analizy sportowej. Inspiracja rozwiązaniami firm, takich jak Opta i DataMB, umożliwiła stworzenie intuicyjnego narzędzia do szybkiej oceny zawodników. W przyszłości warto rozważyć wzbogacenie funkcjonalności o bardziej interaktywne wykresy i możliwość porównywania kilku konkretnych piłkarzy na jednym wykresie.

Modele Random Forest udowodniły swoją użyteczność w analizie sportowej, jednak nie uwzględniają dynamicznych zmian, takich jak forma zawodników, kontuzje oraz zmiany taktyki zespołów. Włączenie tych czynników mogłoby znacząco poprawić precyzję przewidywań.

W odniesieniu do celów pracy, zaprojektowany model oraz aplikacja skutecznie wspierają ocenę jakości potencjalnych transferów piłkarskich, dostarczając rzetelnych danych i wizualizacji pomocnych w podejmowaniu decyzji. Aplikacja umożliwia analizę stylu gry drużyny, ocenę dopasowania zawodnika do zespołu oraz przewidywanie przyszłych statystyk, co stanowi praktyczne wsparcie dla menedżerów i analityków sportowych.

Na dalszym etapie rozwoju projektu warto skoncentrować się na kilku kluczowych obszarach, takich jak poszerzenie bazy danych o dodatkowe sezony oraz szczegółowe informacje o transferach, wykorzystaniu bardziej zaawansowanych algorytmów, takich jak sieci neuronowe. Efektywna byłaby również integrację aplikacji z zewnętrznymi systemami analizy sportowej, zapewniając w ten sposób dostęp do aktualnych statystyk.

Przedstawione wyniki wskazują na duży potencjał projektu, zarówno w analizie statystyk piłkarzy, jak i w ocenie transferów. W potencjalnym rozwoju projektu warto założyć poszerzenie bazy danych o dodatkowe sezony oraz szczegółowe informacje o transferach, uwzględnienie dynamicznych czynników, takich jak taktyka zespołu czy forma zawodnika, rozbudowanie funkcjonalności aplikacji, np. o możliwość integracji z innymi systemami analizy sportowej od dużych firm, które od kilku lat zbierają szczegółowe dane statystyczne.

Podsumowując, projekt udowodnił skuteczność wykorzystania algorytmów uczenia maszynowego w analizie sportowej, jednocześnie wskazując na kluczowe obszary wymagające dalszych badań i optymalizacji. Rozwój narzędzia pozwoli na jego bardziej wszechstronne zastosowanie w profesjonalnych analizach piłkarskich.

## 8. Literatura

- [1] - "Moneyball: The Art of Winning an Unfair Game" - Micheal Lewis
- [2] - "What we do isn't rocket science": how Midtjylland started football's data revolution (<https://www.theguardian.com/football/2020/oct/25/what-we-do-isnt-rocket-science-how-fc-midtjylland-started-footballs-data-revolution>, dostęp: 14.01.2025)
- [3] - 2021-2022 Football Player Stats (<https://www.kaggle.com/datasets/vivovinco/20212022-football-player-stats>, dostęp: 06.10.2024)
- [4] - 2022-2023 Football Player Stats (<https://www.kaggle.com/datasets/vivovinco/20222023-football-player-stats>, dostęp: 20.11.2024)
- [5] - Oficjalna dokumentacja języka Python (<https://docs.python.org/3/>, dostęp: 03.01.2025)
- [6] - Oficjalna dokumentacja biblioteki Pandas (<https://pandas.pydata.org/docs/>, dostęp: 03.01.2025)
- [7] - Oficjalna dokumentacja biblioteki NumPy (<https://numpy.org/doc/>, dostęp: 03.01.2025)
- [8] - Oficjalna dokumentacja biblioteki Matplotlib (<https://matplotlib.org/stable/contents.html>, dostęp: 03.01.2025)
- [9] - Oficjalna dokumentacja biblioteki Seaborn (<https://seaborn.pydata.org/>, dostęp: 03.01.2025)
- [10] - Oficjalna dokumentacja biblioteki Scikit-learn (<https://scikit-learn.org/>, dostęp: 03.01.2025)
- [11] - Oficjalna dokumentacja biblioteki Joblib (<https://joblib.readthedocs.io/>, dostęp: 03.01.2025)
- [12] - Oficjalna dokumentacja Streamlit (<https://streamlit.io/>, dostęp: 03.01.2025)
- [13] - Exploring Playing Styles of Top 5 League Teams (<https://medium.com/@alf.19x/exploring-playing-styles-of-top-5-league-teams-c68906a332d2>, dostęp: 29.12.2024)

## 9. Summary

This engineering thesis aims to develop a machine learning model in Python to assist in evaluating football transfer quality. The tool analyzes data on players, statistics, and team styles to support transfer decision-making. The thesis is divided into two parts: theoretical and practical.

The theoretical section introduces the tools and libraries used in the project, machine learning basics, football data features, and the role of data visualization in sports. The practical part focuses on an application consisting of five sections: Search, Statistics, Team Style, Statistics Prediction, and Team Fit.

The "Search" tab allows users to filter players by criteria such as nationality, position, league, and key statistics. The "Statistics" tab enables users to view a player's key stats and compare them with averages for their position and league. The "Team Style" tab analyzes a team's playing style for a selected season and compares it to other teams. The "Statistics Prediction" tab uses models to forecast a player's future statistics, and the "Team Fit" tab assesses how well a player's style fits a team and predicts their stats in the new team.

Football data analysis is increasingly popular, with clubs using advanced technologies to assess transfers. This project combines data analysis methods with an intuitive user interface to evaluate transfers using historical data from sources like Football Reference, Transfermarkt, and WhoScored. After processing and merging the data, a dataset was created for analysis.

The application uses Python and libraries like Pandas, NumPy, Matplotlib, Seaborn, Scikit-learn, Joblib, and Streamlit to create an interactive tool. Predictive models, based on Random Forest, show high accuracy in predicting stats like goals and assists. Results confirmed the application's effectiveness in transfer decision-making.

In conclusion, the project demonstrates how machine learning and data analysis can improve football transfer decisions, enhancing team management.