AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

Wydział Zarządzania

Samodzielna Pracownia Zastosowań Matematyki w Ekonomii

Praca dyplomowa

Zastosowanie wybranych metod uczenia nadzorowanego w przewidywania wyników drużyn zespołowych

Autor: *Jakub Paweł Laszczyk*

Kierunek studiów: Informatyka i Ekonometria

Opiekun pracy: *dr hab. Łukasz Lach, prof. uczelni*

Kraków, 2023

Spis treści

Wstęp

W dzisiejszej erze cyfrowej transformacji, zastosowanie metod uczenia maszynowego stało się nieodłącznym elementem analizy i przetwarzania ogromnych zbiorów danych w różnych dziedzinach. Jednym z obszarów, w którym te techniki znalazły zastosowanie, jest przewidywanie wyników meczów piłki nożnej. Motywacja podjęcia tego tematu wiąże się z rosnącym zainteresowaniem analizą sportowych danych w celu uzyskania przewagi konkurencyjnej oraz lepszego zrozumienia czynników wpływających na rezultaty meczów.

Celem niniejszej pracy jest zbadanie skuteczności metod uczenia nadzorowanego w przewidywaniu wyników meczów piłki nożnej oraz zrozumienie, czy za pomocą tych technik możliwe jest osiągnięcie przewagi w stosunku do kursów bukmacherskich na podstawie ogólnodostępnych danych.

Pracę rozpoczęto od wprowadzenia do kontekstu poznawczego, a także omówienia kluczowych pojęć związanych z uczeniem maszynowym. Przedstawiono kolejno etapy procesu uczenia maszynowego oraz omówiono istotne miary służące do oceny skuteczności modeli. Następnie dokonano przeglądu istniejących badań dotyczących prognozowania wyników meczów piłki nożnej. Szczegółowo opisano zbiór danych użyty w analizie oraz przedstawiono trzy różne metody uczenia nadzorowanego, które zostały zastosowane i porównane: drzewa decyzyjne, lasy losowe oraz wzmocnienie gradientu. Metody te zostały wybrane ze względu na ich łatwą interpretowalność oraz zdolność do pracy z danymi o złożonej strukturze, charakterystycznej dla wyników sportowych. Przy tworzeniu badania zostały wykorzystane dane piłkarskie pochodzące z ogólnodostępnych zbiorów danych udostępnionych przez serwis Football-Data oraz Transfermarkt, wybór tych źródeł został podyktowany ich szerokim zakresem informacji oraz wiarygodnością prezentowanych danych. Kolejnym krokiem było przeprowadzenie wszechstronnej analizy danych, w której uwzględniono statystyki opisowe, badanie korelacji między zmiennymi oraz identyfikację obserwacji odstających. W finalnym etapie badania, dokonano prezentacji zaimplementowanych metod. Uzyskane rezultaty zostały zestawione ze sobą oraz porównanie z kursami bukmacherskimi.

1. Wymiar poznawczy

Początki użycia analizy statystycznej oraz uczenia maszynowego w sporcie sięgają 1858 roku kiedy to dziennikarz sportowy Henry Chadwick opracował metrykę zwaną „box score", polegała ona na zapisywaniu w formie tabeli wyników zawodników baseballa. Dzięki temu statystycy mogli mierzyć wydajność graczy oraz drużyn, w sposób ilościowy i wykorzystywać te dane do lepszego doboru zawodników. Jako krok milowy w kontekście popularyzacji analizy danych w sporcie można uznać wykorzystanie przez Billiego Beana oraz jego asystenta Paula Depodesta koncepcji oceny oraz doboru zawodników opartej na analizie statystyk meczowych. Metoda polegała na doborze zawodników którzy posiadają widoczne wady ale również jak pokazywały statystyki przynajmniej jedna cechę na najwyższym poziomie. Ustawiali oni zawodników w taki sposób aby wzajemnie się uzupełniali w celu minimalizowania ich wad oraz maksymalizacji szans na zdobycie punktów. Użycie metod statystycznych pozwoliło im osiągnąć spektakularny sukces jakim było doprowadzenie drużyny z 24 budżetem ligi na 5 miejsce w sezonie zasadniczym. Był to znak dla całego świata sportu, że użycie analizy danych może dać znaczącą przewagę w zawodach. Od tego czasu analiza statystyczna w sporcie obecna jest prawie wszędzie, kluby sportowe wykorzystują ją do lepszego rozpracowania przeciwników, wykrywania ryzyka kontuzji oraz identyfikacji nowych talentów.

W miarę postępu analizy sportowej, przemysł zakładów hazardowych skoncentrowanych na różnych dziedzinach sportu również znacząco się rozwinął. W 2022 roku globalny rynek zakładów sportowych został wyceniony na 81,03 miliarda dolarów, a prognozuje się, że do roku 2030 osiągnie wartość 167,50 miliardów dolarów.[1] W badaniu zostały wykorzystane kursy bukmachera bet365, jest to jedna z czołowych firm bukmacherskich na świecie, w samym ubiegłym roku (okres rozliczeniowy marzec 2021 – marzec 2022) wygenerowała ona przychody wynoszące 2,85 miliarda funtów[2]. Kursy bukmacherskie na dane wydarzenia obliczane są jako odwrotność prawdopodobieństwa wyliczonego za pomocą różnych technik analizy danych stosowanych przez firmy. Przy ustalaniu tych kursów, bukmacherzy opierają

[1] https://www.vantagemarketresearch.com/industry-report/sports-betting-market-1710 - data odczytu 10.08.2022r.;
[2] https://osgamers.com/frequently-asked-questions/how-much-money-does-bet-365-make - data odczytu 10.08.2022r.;

się na ogromnych bazach danych zawierających szeroki zakres statystyk związanych z wcześniejszymi wydarzeniami sportowymi. W przypadku piłki nożnej mogą być to takie czynniki jak bilans spotkań bezpośrednich, forma zespołu, kontuzje kluczowych zawodników oraz wiele innych wskaźników specyficznych dla danej drużyny. Ponieważ kurs bukmacherski jest odwrotnością prawdopodobieństwa, suma prawdopodobieństw wyliczanych z tych kursów dla danego wydarzenia sportowego powinna być równa jeden. Nie dzieje się tak z tego powodu, że do każdego kursu dodawana jest marża bukmacherska, zwykle wynosi ona około 10% i jest stosowana przez bukmacherów w celu osiągania długoterminowych zysków. W badaniach zostały wykorzystane kursy dziesiętne, są to najczęściej stosowane kursy na rynkach europejskich. Kursy te odzwierciedlają prawdopodobieństwo zdarzenia, przy czym niższy kurs (minimalny kurs to 1, co odpowiada 100% prawdopodobieństwu) oznacza większą szansę na wystąpienie danego zdarzenia. Natomiast wyższy kurs wskazuje na niskie prawdopodobieństwo według bukmachera.

1.1. Co to jest uczenie maszynowe?

Uczenie maszynowe jest szybko rozwijającą się gałęzią sztucznej inteligencji. Zajmuje się implementacją algorytmów, które działając na wzór ludzkiej inteligencji, samodzielnie analizują dane w celu odnajdywania wzorców i zależności. Aby lepiej zrozumieć, czym jest pojęcie „uczenie maszynowe", należy najpierw zrozumieć, czym jest umiejętność uczenia się u człowieka. Proces ten definiuje się jako zdolność do zdobywania wiedzy, umiejętności oraz doświadczenia, za pomocą przyswajania oraz wyciągania informacji z otoczenia. Nauka towarzyszy nam od początku istnienia. To dzięki niej stale rozwijamy się, odkrywamy nowe rzeczy oraz udoskonalamy stare.

Nils J. Nilsson[3] w swojej książce porównuje uczenie maszynowe do procesu poznawczego u zwierząt i ludzi, argumentując, że wiele metod opartych jest na modelach nauki biologicznej. Definiuje on uczenie maszynowe jako zmiany w systemach AI, które mogą być ulepszeniami działających już systemów lub tworzeniem nowych modeli. Autor podkreśla, że metody uczenia maszynowego są użyteczne w przypadkach, gdy zadania są trudne do precyzyjnego zdefiniowania lub gdy wiedza jest zbyt obszerna i zmienia się dynamicznie. Natomiast T.Mitchell[4] określił uczenie maszynowe

[3] N. J. Nilsson, *Introduction to machine learning,* Stanford University 1998r., s.10-11;
[4] T. Mitchell, *Machine Learning,* McGraw Hill 1997r., s.3;

jako *''badanie algorytmów komputerowych, które automatycznie poprawiają się dzięki doświadczeniu"*. Podobną definicje zaproponował również Peter Flach - „*Uczenie maszynowe to systematyczne badanie algorytmów i systemów, które poprawiają swoją wiedzę lub wydajność dzięki doświadczeniu*[5]". W obu tych definicjach można dostrzec, że według autorów istotą uczenia maszynowego jest poprawa działania algorytmów, uczących się poprzez doświadczenie, rozumiane jako dostarczanie nowych danych do modelu. Dzięki temu maszyny są w stanie prognozować i podejmować decyzje bez potrzeby jawnego programowania przez ludzi. Nils J.Nilsson zwrócił również uwagę na powód, dla którego maszyny muszą się uczyć. Jako podstawowe przyczyny podał ukryte zależności oraz korelacje, które mogą znajdować się w dużych zbiorach danych, a które jesteśmy w stanie zauważyć tylko za pomocą uczenia maszynowego. Z kolei maszyny, które uczą się wiedzy stopniowo, mogą sobie radzić lepiej z takimi zbiorami danych niż ludzie. Autor stwierdza również, że charakterystyka oraz dynamika zmian środowiska, w którym będzie pracował model, często nie są znane w momencie jego tworzenia. Dlatego model, który potrafi się uczyć, może być doskonalony na bieżąco.

1.2. Zastosowania uczenia maszynowego

Obecnie, każdego dnia, generowane są ogromne zasoby danych. Badania z 2013 roku mówią, że około 90% danych dostępnych na świecie zostało wytworzonych w latach 2011-2012[6]. Statystyka ta wskazuje na dużą dynamikę zjawiska produkcji informacji i pokazuje kierunek, w którym zmierza analiza danych. Dzięki dostępowi do rozległych baz danych, modele uczące się, stają się coraz skuteczniejsze i bardziej precyzyjne w rozwiązywaniu złożonych problemów. Mogą być wykorzystywane w różnorodnych dziedzinach życia: od przewidywania trendów konsumenckich w e-commerce, przez diagnozowanie chorób serca na podstawie danych medycznych, po optymalizację procesów produkcyjnych w przemyśle. W każdym z tych przykładów uczenie maszynowe pozwala na wykorzystanie dużej ilości danych i złożonych wzorców, które trudno byłoby odkryć za pomocą tradycyjnych metod analizy.

Powyższe wnioski sprawiają, że uczenie maszynowe jest jednym z kluczowych obszarów sztucznej inteligencji, który ma ogromny potencjał w transformacji rozmaitych

[5] P. A. Flach, *Machine Learning. The Art and Science of Algorithms that Make Sense of Data,* Cambridge University Press 2012r., s. 3;
[6] P. B. Brandtzæg, *Big Data, for Better or Worse: 90% of World's Data Generated Over Last Two Years,* Sintef 2013r.;

dziedzin życia. Jego nieustanny rozwój i zdolność do samodzielnego doskonalenia sprawiają, że jest to niezwykle ważne narzędzie w dzisiejszym świecie technologicznym.

1.3. Metody uczenia maszynowego

Podczas nauki, każdy z nas posiada swoje metody, które działają najlepiej w jego przypadku. Niektórzy robią notatki, inni powtarzają materiał lub rozwiązują zadania praktyczne, aby lepiej zrozumieć i zapamiętać wiedzę. Tak, jak różne techniki nauki są dostosowane do rozmaitych przedmiotów i sytuacji - tak samo, różne techniki uczenia maszynowego są dostosowane do odmiennych rodzajów danych i problemów. Najbardziej znanym rodzajem podziału algorytmów uczenia maszynowego, jest podział ze względu na informację trenującą. Informacja ta służy do odpowiedzi na pytanie „czy wyniki działania modelu są prawidłowe?". Ze względu na jej rodzaj oraz dostępność możemy wyróżnić trzy główne typy uczenia maszynowego:[7]

1) **Uczenie nadzorowane** (*ang. supervised learning*)

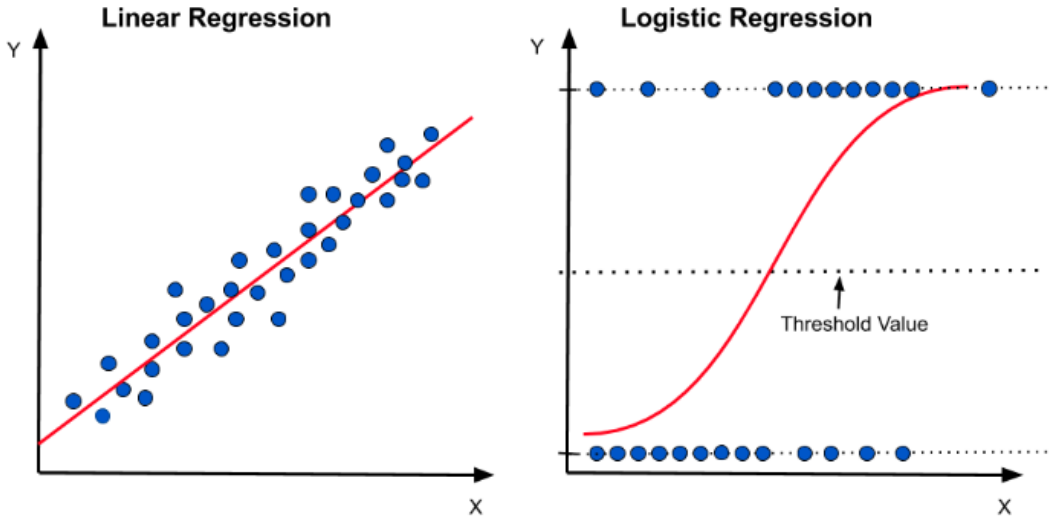
   Uczenie nadzorowane, nazywane również „z nauczycielem", jest rodzajem uczenia maszynowego, w którym system rozwija się na podstawie odpowiedzi dostarczonych przez programistę. Analogiczną stroną w procesie uczenia się przez ludzi, jest rola nauczyciela, który podaje poprawne odpowiedzi oraz pomaga dobrze zrozumieć nieznany temat. Maszyna w tym przypadku przyjmuje rolę ucznia, który na podstawie przykładów, stara się znaleźć wzorce i schematy poprawnego wykonania zadania. W tym rodzaju uczenia na samym początku do systemu przekazywane są pary danych wejściowych wraz z odpowiadającymi im informacjami wyjściowymi. Na podstawie tych par maszyna stara się znaleźć wzór funkcji najlepiej opisujący daną zależność, który następnie zostanie wykorzystany do prognozowania z wykorzystaniem jeszcze nieznanych danych. Najbardziej znanymi przykładami użycia uczenia nadzorowanego są: klasyfikacja obrazów, filtracja spamu, przewidywanie cen akcji czy diagnozowanie chorób. Jako najpopularniejsze problemy rozwiązywane za pomocą uczenia nadzorowanego, wyróżnia się regresję oraz klasyfikację.

   Regresja jest to metoda statystyczna, która umożliwia estymowanie nieznanej wartości ciągłej zmiennej, objaśnianej za pomocą zmiennych

[7] D. Banasiak, *Sztuczna inteligencja – Uczenie się maszyn – wykład 9,* Politechnika Wrocławska, s.5;

objaśniających. Celem regresji jest wyznaczenie wzoru funkcji, który najlepiej opisuje zależność między zmienną objaśnianą, a różnymi wartościami zmiennych niezależnych.[8] Najczęściej używanymi algorytmami regresji są regresja liniowa oraz regresja logistyczna.

Ilustracja 1. Wykres regresji liniowej i logistycznej.



Źródło: https://www.datacamp.com/tutorial/understanding-logistic-regression-python - data odczytu 13.07.2023r.

Wzór regresji w zapisie formalnym model przybiera postać:

$$Y = f(X, \beta) + \varepsilon \qquad (1)$$

Gdzie:

X – wektor zmiennych objaśniających

Y- zmienna objaśniana

β- wektor współczynników regresji

$f(X, \beta)$ - funkcja regresji o wartościach w liczbach rzeczywistych

ε - błąd losowy

[8] https://pl.wikipedia.org/wiki/Regresja_(statystyka) – data odczytu 22.07.2022r.;

Klasyfikacja jest rodzajem algorytmu, który za pomocą danych wejściowych i wyjściowych, szuka zależności i wzorców, a następnie przydziela obserwacje do odpowiednich klas. Typowym problemem klasyfikacyjnym jest diagnoza medyczna - na podstawie wyników badań oraz historii medycznej wystawiana jest diagnoza, czyli przypisanie do klasy zdrowej lub chorej. Definicja formalna: [9]

Dla danego zbioru danych trenujących $\{(x_1, y), \ldots, (x_n, y)\}$ znaleźć klasyfikator $h: X \rightarrow Y$, który przydziela obiektowi $x \in X$ klasę $y \in Y$

2) **Uczenie nienadzorowane** (*ang. Unsupervised learning*)

Uczenie nienadzorowane nazywane również „bez nauczyciela" jest typem uczenia maszynowego, w którym dostarczamy do modelu jedynie dane wejściowe bez korespondujących im danych wyjściowych. Jest to rodzaj uczenia najbardziej naśladujący proces nauki przez człowieka, ponieważ tak samo, jak ludzie na podstawie obserwacji otoczenia, intuicyjnie zauważa podobieństwa i wzorce. W uczeniu bez nauczyciela występują dwie główne techniki: [10]

a) Redukcja wymiarowości

Redukcja wymiarowości w uczeniu maszynowym polega na zmniejszeniu ilości cech z jednoczesnym zachowaniem jak największego poziomu informacji. Wymiarowość w tym kontekście rozumiana jest jako liczba cech. W uczeniu maszynowym często występuje zjawisko pogarszania się wydajności modelu wraz ze zwiększeniem wymiarowości. Spowodowane jest to wzrostem złożoności modelu wraz z liczbą cech. Redukcje wymiarowości używa się z kilku powodów. Najczęściej stosowanymi są: redukcja złożoności modelu, poprawa jakości predykcyjnej modelu oraz wizualizacja danych. Dwie główne techniki służące do redukcji wymiarowości to[11]selekcja cech oraz ekstrakcja cech.

Selekcja cech polega na wyborze ze zbioru danych cech, które maja największe znaczenie dla badanego problemu. Celem jest redukcja wymiarowości

[9] https://pl.wikipedia.org/wiki/Klasyfikacja_statystyczna - data odczytu 22.07.2023r.;
[10] https://datascience.eu/pl/uczenie-maszynowe/bezobslugowe-uczenie-sie-maszyn/ - data odczytu 22.07.2023r.;
[11] A. Uberoi, *Introduction to Dimensionality Reduction,* „GeeksforGeeks" 2023r.;

z zachowaniem najważniejszych cech. Do technik selekcyjnych należą: metoda filtrowania, zwijania oraz metoda wbudowane.

Ekstrakcja cech polega na tworzeniu nowych zmiennych poprzez łączenie lub przekształcanie już istniejących. To podejście stosowane jest, jeżeli chcemy zmniejszyć wymiarowość, a zachować całą pojemność informacyjną. Wyróżniamy kilka głównych techniki ekstrakcji cech: Analiza głównych składowych, liniowa analiza dyskryminacyjna, nieliniowa technika redukcji wymiarowości.[12]

b) Analiza skupień (klasteryzacja)

Jest to technika polegająca na grupowaniu zbioru danych w klastry. Celem tej metody jest pogrupowanie danych w taki sposób, żeby elementy w każdym klastrze były jak najbardziej podobne do siebie, przy jednoczesnym zachowaniu niskiego poziomu podobieństwa klaster między sobą. Grupowanie dokonywane jest za pomocą metryki prawdopodobieństwa, która wyraża podobieństwo lub odległość między danymi. Klasteryzacja używana jest najczęściej do eksploracji danych, wyszukiwania informacji oraz segmentacji obrazu. Do najczęściej stosowanych metod analizy skupień należą:

1) Metody hierarchiczne
2) Grupa metod k-średnich
3) Metody rozmytej analizy skupień [13]

3) **Uczenie przez wzmacnianie** (*ang. RL - reinforcement learning*)

Uczenie przez wzmacnianie różni się od uczenia nadzorowanego oraz uczenia bez nadzoru przede wszystkim brakiem zbioru testowego. W tym rodzaju uczenia występuje agent oraz środowisko, z którego pobierane są informacje. Agentem jest algorytm sztucznej inteligencji który pobiera informacje ze środowiska i na jej podstawie stara się generować nową wiedze lub udoskonalić już istniejąca w celu zmaksymalizowania nagrody.

[12] https://www.javatpoint.com/dimensionality-reduction-technique - data odczytu 26.07.2023r.;
[13] https://pl.wikipedia.org/wiki/Analiza_skupie%C5%84 – data odczytu 26.07.2023r.;

Jako najpopularniejsze techniki zdobywania wiedzy w uczeniu przez wzmacnianie określa się:

- uczenie się na pamięć
- uczenie się przez indukcję
- uczenie się przez dedukcję
- uczenie się przez analogię
- uczenie się przez instrukcję

1.4. Etapy uczenia maszynowego

Podstawą procesu uczenia jest dostarczanie maszynom danych, które służą do analizy i nauki, a na ich podstawie konstruowane są modele matematyczne. Przed dostarczeniem, dane powinny być starannie przygotowane w procesie obróbki wstępnej, który ma na celu wyeliminowanie zbędnych informacji i ułatwienie analizy. Proces ten obejmuje przede wszystkim oczyszczenie danych, usunięcie obserwacji odstających, standaryzacje, uzupełnienie braków, wykrywaniu anomalii oraz rozwiazywaniu problemów z integralnością danych[14]. Następnie dane dzielone są na zbiór treningowy oraz zbiór testowy, co pozwala na ocenę wydajności modelu na danych, które nie zostały użyte w procesie nauki. Kolejnym krokiem jest wybór algorytmu, który najczęściej jest definiowany przez rodzaj dostępnych danych oraz wyznaczony cel. W kolejnym etapie konieczne jest dobranie odpowiednich wartości hiperparametrów, które wpływają na działanie algorytmu, a ich wartości mogą się różnić w zależności od wybranej metody. Po wykonaniu tych czynności należy dobrać odpowiednią miarę sukcesu. Służy ona do oceny sprawności modelu dla dobranego problemu. W zależności od wybranej metody stosowane są różne miary oceny. Ponieważ praca dotyczy problemu klasyfikacyjnego, w dalszej części zostały opisane tylko miary stosowane do oceny algorytmów tego typu.[15]

Po dobraniu odpowiednich miar sukcesu należy przejść do trenowania. Na tym etapie dostarcza się dane do modelu, który samodzielnie buduje odpowiednie reguły. Kolejno następuje ewaluacja modelu na danych testowych, do której służy wybrana miara sukcesu. W zależności od uzyskanego wyniku, kończone są na tym etapie dalsze prace

[14] https://azure.microsoft.com/pl-pl/resources/cloud-computing-dictionary/what-is-machine-learning-platform - data odczytu 22.07.2023r.;
[15] A. Król-Nowak, K. Kotarba, *Podstawy uczenia maszynowego,* wydawnictwo AGH, Kraków 2022r., s.11-14;

lub przechodzi się do optymalizacji hiperparametrów. Polega ona na zmianie wartości hiperparametrów oraz ponownym wytrenowaniu modelu w celu poprawienia uzyskanych wyników. [16]

1.5. Miary jakości modelu

Podstawowymi miarami oceny błędu są:

Miara trafności klasyfikowania (dokładność) wynosi:[17]

$$N_{ov} = \frac{n_T}{n} \qquad (2)$$

gdzie:

n – liczba przykładów testowych,

n_T– liczba poprawnie sklasyfikowanych przykładów testowych,

n_F – liczba błędnie sklasyfikowanych przykładów testowych.

Łączny błąd klasyfikowania (ang. overall error rate) jest określony wzorem:

$$E_{ov} = 1 - N_{ov} = \frac{n_F}{n} \qquad (3)$$

gdzie:

n – liczba przykładów testowych,

n_F – liczba błędnie sklasyfikowanych przykładów testowych.

Oprócz tych miar należy również utworzyć tzw. Macierz pomyłek. Utworzona jest ona poprzez zestawienie klasy prognozowanej i klasy faktycznie zaobserwowanej w wyniku czego powstają 4 przypadki:

[16] A. Król-Nowak, K. Kotarba, *Podstawy uczenia maszynowego,* wydawnictwo AGH, Kraków 2022r., s.11-13;
[17] A. Król-Nowak, K. Kotarba, *Podstawy uczenia maszynowego,* wydawnictwo AGH, Kraków 2022r., s.20-21;

True-Positive (TP - prawdziwie pozytywne) to przypadki, gdy model przewiduje pozytywną klasę, a w rzeczywistości została ona zaobserwowana jako pozytywna.

True-Negative (TN – prawdziwie negatywna) to przypadki, gdy model przewiduje negatywną klasę, a w rzeczywistości została ona zaobserwowana jako negatywna.

False-Positive (FP – fałszywie pozytywna) to przypadki, gdy model przewiduje pozytywną klasę, a w rzeczywistości została ona zaobserwowana jako negatywna.

False-Negative (FN – fałszywie negatywna) to przypadki, gdy model przewiduje negatywną klasę, a w rzeczywistości została ona zaobserwowana jako pozytywna.[18]

1- Klasa pozytywna,

0 - Klasa negatywna,

Tabela 1. Macierz pomyłek

Wartość przewidywana	Wartość rzeczywista	
	1	0
1	n_{TP}	n_{FP}
0	n_{FN}	n_{TN}

Źródło: Opracowanie własne przy wykorzystaniu macierzy pomyłek A. Król-Nowak, K. Kotarba, *Podstawy uczenia maszynowego,* wydawnictwo AGH, Kraków 2022, s.21;

Aleksandra Król-Nowak oraz Katarzyna Kotarba[19] sugerują aby w macierzy pomyłek zwrócić szczególna uwagę na przypadki fałszywie pozytywne które odpowiadają błędom pierwszego rodzaju oraz przypadki fałszywie negatywne które odpowiadają błędom drugiego rodzaju. Z tego powodu, że nie można kontrolować obu błędów w tym samym czasie należy wybrać który przypadek jest ważniejszy w kontekście podejmowanego problemu i na nim oprzeć swoje obliczenia.

[18] M. Gromada, *Confusion matrix, Macierz błędu, tablica / macierz pomyłek – czyli ocena jakości klasyfikacji (część 1),* mathspace,pl 2015r.;
[19] A. Król-Nowak, K. Kotarba, *Podstawy uczenia maszynowego,* wydawnictwo AGH, Kraków 2022r., s.21;

Na podstawie macierzy pomyłek zostały utworzone kolejne cztery miary jakości klasyfikacji:

Czułość (ang. *sensitivity, recall*) określana również jako odsetek prawdziwie pozytywnych (ang. *true positive rate*, TPR) – mierzy skuteczność modelu w przewidywaniu rzeczywiście pozytywnych przypadków spośród wszystkich zaobserwowanych pozytywnych przypadków

$$TPR = \frac{TP}{P} = \frac{TP}{TP+FN} \qquad (4)$$

Swoistość (ang. *specificity*, SPC) określana również jako odsetek prawdziwie negatywnych (ang. *true negative rate*, TNR) - mierzy skuteczność modelu w przewidywaniu rzeczywiście negatywnych przypadków spośród wszystkich zaobserwowanych negatywnych przypadków

$$TNR = \frac{TN}{N} = \frac{TN}{FP+TN} \qquad (5)$$

Precyzja (ang. *precision*, lub *positive predictive value*, PPV – wartość predykcyjna dodatnia) mierzy ona skuteczność modelu w identyfikowaniu prawdziwych pozytywnych przypadków spośród wszystkich przewidywanych jako pozytywne

$$PPV = \frac{TP}{TP+FP} \qquad (6)$$

Dokładność (ang. *accuracy*, ACC) - mierzy ona skuteczność modelu w identyfikowaniu prawdziwych negatywnych wyników spośród wszystkich przewidywanych jako negatywne.[20],[21]

$$ACC = \frac{TP+TN}{P+N} \qquad (7)$$

Krzywa ROC (ang. Receiver operating characteristic) - służy do oceny dokładności predykcji. Krzywą ROC tworzy wykres zależności miedzy czułością, a 1-swoistość. Idealny model reprezentowany byłby przez krzywa która idzie wzdłuż lewej krawędzi wykresu, natomiast w model losowym krzywa ROC byłaby równa przekątnej.

[20] Tablica pomyłek – Wikipedia, wolna encyklopedia – data odczytu 23.07.2023r.;
[21] M. Gromada, *Zasięg (TPR – czułość / TNR – specyficzność) i precyzja (PPV / NPV) – czyli ocena jakości klasyfikacji (część 2),* mathspace,pl 2015r.;

AUC (ang. Area under curve) – to obszar pod krzywa ROC, służy do oceny poprawności klasyfikatora. Przyjmuje wartości z zakresu [0,1], gdzie 0.5 oznacza model losowy, a 1 model idealny.

2. Wymiar metodologiczny

W niniejszym rozdziale omówione zostaną zastosowane w badaniu metody uczenia oraz istniejące już prace naukowe w obszarze przewidywania rezultatów sportowych za pomocą technik uczenia maszynowego. Przedstawiony zostanie również zbiór danych, który posłuży jako podstawa do konstrukcji modeli.

2.1. Opis użytych metod

2.1.1. Drzewa decyzyjne

Drzewa decyzyjne są algorytmem uczenia nadzorowanego wykorzystywanym do graficznego wspierania procesu podejmowania decyzji. Metoda podziału w drzewach ma charakter hierarchiczny, a podziały dokonywane są w taki sposób aby wynikiem były jak najbardziej jednorodne klasy. Drzewo decyzyjne składa się z korzenia, węzłów, gałęzi oraz liści. Korzenie są podstawą istnienia drzewa, znajdują się u góry i zawierają cały zbiór treningowy, w węzłach dokonywany jest podział zbioru w oparciu o wybrane kryterium podziału, następnie na podstawie tego kryterium wybierana jest odpowiednia gałąź łącząca zbiór danych podzielony w danym węźle decyzyjnym z kolejnymi zbiorami. Nowe zbiory stają się kolejnymi węzłami decyzyjnymi w procesie analizy lub klasyfikacji. Liście są końcowymi węzłami bez potomków, na nich kończy się etap uczenia, nie wychodzą z nich kolejne gałęzie. Klasyfikacja danej obserwacji polega na poruszaniu się od początkowego węzła drzewa decyzyjnego (korzenia) do końcowego węzła (liścia) i przypisaniu obserwacji do określonej klasy, która jest zdefiniowana w tym liściu.[22] [23]

Kryterium podziału

Kryterium podziału służy do znalezienia optymalnego podziału zbioru w węźle, w taki sposób aby zwiększyć jednorodność klas, i poprawić zdolność do predykcji. Kryterium podziału określane tez testem związane jest z atrybutami naszego modelu oraz przyjmowanymi przez nie wartościami. Metody które możemy wykorzystać do podziału w drzewach klasyfikacyjnych to:

---

[22] A. Król-Nowak, K. Kotarba, *Podstawy uczenia maszynowego,* wydawnictwo AGH, Kraków 2022r., s.109;

[23] https://mfiles.pl/pl/index.php/Drzewo_decyzyjne - data odczytu 25.07.2023r.;

a) Współczynnik Giniego

Kryterium Giniego opiera swoje działanie na wartości indeksu Giniego, czyli mierze koncentracji zmiennej losowej. Przyjmuje wartości z przedziału [0,1], gdzie 0 oznacza przynależność wszystkich obiektów do jednej klasy. Współczynnik Giniego obliczany dla drzewa binarnego może mieć następująca postać:

$$gini(S_j) = 1 - \sum_{i=1}^{p} \left(\frac{m_i^{(j)}}{|S_j|}\right)^2 \qquad (8)$$

„gdzie:

$gini(S_j)$ – współczynnik Giniego dla j-tego zbioru który jest wyznaczony przez wybrane kryterium podziału.

$|S_j|$ – moc zbioru S_j , j = 1, 2,

$m_i^{(j)}$– liczba obiektów i-tej klasy w zbiorze S_j.

Na podstawie współczynników Giniego obliczonych dla każdego zbioru wyznaczonego przez badany podział, można obliczyć współczynnik podziału Giniego. j = 1, 2, otrzymujemy:

$$g = \frac{|S_1|}{|S_1|+|S_2|} gini(S_1) + \frac{|S_2|}{|S_1|+|S_2|} gini(S_2) = \frac{m_1^{(1)}+m_2^{(1)}}{m_1+m_2} gini(S_1) + \frac{m_1^{(2)}+m_2^{(2)}}{m_1+m_2} gini(S_2) \qquad (10)$$

gdzie:

g – współczynnik podziału Giniego,

$|S_j|$ – moc zbioru , S_j, j= 1, 2,

$m_1^{(j)}$, $m_2^{(j)}$– liczba obiektów odpowiednio klasy nr 1 i 2 w zbiorze S_j,

m_1, m_2– liczba wszystkich obiektów odpowiednio klasy nr 1 i 2."[24]

Schemat wyboru optymalnego kryterium podziału:

1. Obliczamy współczynnik $gini(S_j)$ dla każdego ze zbiorów powstających w wyniku podziału.
2. Obliczmy współczynnik podziału g
3. Powtarzamy kroki 1-2 dla różnych kryteriów podziału względem zmiennych zależnych
4. Wybieramy kryterium które posiada najmniejszą wartość współczynnika g

b) Entropia i zysk informacyjny

Entropia służy do pomiaru stopnia niejednorodności w zestawie danych. Na podstawie entropii każdego atrybutu możemy oszacować ich zysk informacyjny. Zysk informacyjny mierzy spodziewane zmniejszenie entropii poprzez wykorzystanie danej cechy. Ta wartość wskazuje, które zmienne najlepiej dzielą zbiór danych na jak najbardziej jednorodne podzbiory. W związku z tym, wybierając kryterium, powinniśmy dążyć do maksymalizacji zysku informacyjnego.[25]

Schemat wyboru optymalnego kryterium podziału opisany przez Aleksandrę Król-Nowak[26]:

$d_1, \dots, d_p$ – kolejne wartości atrybutu decyzyjnego

$S_j, j = 1, \dots, r$ – zbiory powstałe w wyniku podziału względem badanego kryterium

$P(d_i)$ - prawdopodobieństwo wystąpienia wartości d_i atrybutu decyzyjnego w zbiorze danych

1. Obliczenie wartości entropii dla atrybutu decyzyjnego:

[24] A. Król-Nowak, K. Kotarba, *Podstawy uczenia maszynowego,* wydawnictwo AGH, Kraków 2022r., s.112
[25] https://analizadanychwpilce.com/2018/10/15/wykorzystanie-modelu-drzewa-decyzyjnego-do-analizy-wynikow-spotkania/ - data odczytu 25.07.2023r.;
[26] A. Król-Nowak, K. Kotarba, *Podstawy uczenia maszynowego,* wydawnictwo AGH, Kraków 2022r., s.113;

$$H = -\sum_{i=1}^{p} P(d_i) * log_2\big(P(d_i)\big) \qquad (10)$$

2. Obliczenie współczynnika entropii dla atrybutu decyzyjnego w zbiorze S_j:

$$H_j = -\sum_{i=1}^{p} P\big(d_i|S_j\big) * log_2\left(P\big(d_i|S_j\big)\right) \qquad (11)$$

3. Obliczenie entropii podziału zbioru ze względu na wybrane kryterium:

$$E = \sum_{j=1}^{r}\left(\frac{|S_j|}{\sum_{j=1}^{r}|S_j|} * H_j\right) \qquad (12)$$

4. Obliczenie zysku informacyjnego

$$Gain = H - E \qquad (13)$$

Kryterium stopu

Kryterium stopu odpowiada za zatrzymanie konstrukcji modelu, inaczej mówiąc decyduje o tym kiedy węzeł staje się liściem. Dzieje się tak następujących przypadkach:

1. Aktualny zbiór obiektów jest pusty.
2. Wszystkie obiekty w węźle są tej samej klasy.
3. Brak atrybutu który jest w stanie kontynuować dzielenie przykładów.[27]

Schemat budowy drzewa decyzyjnego[28]:

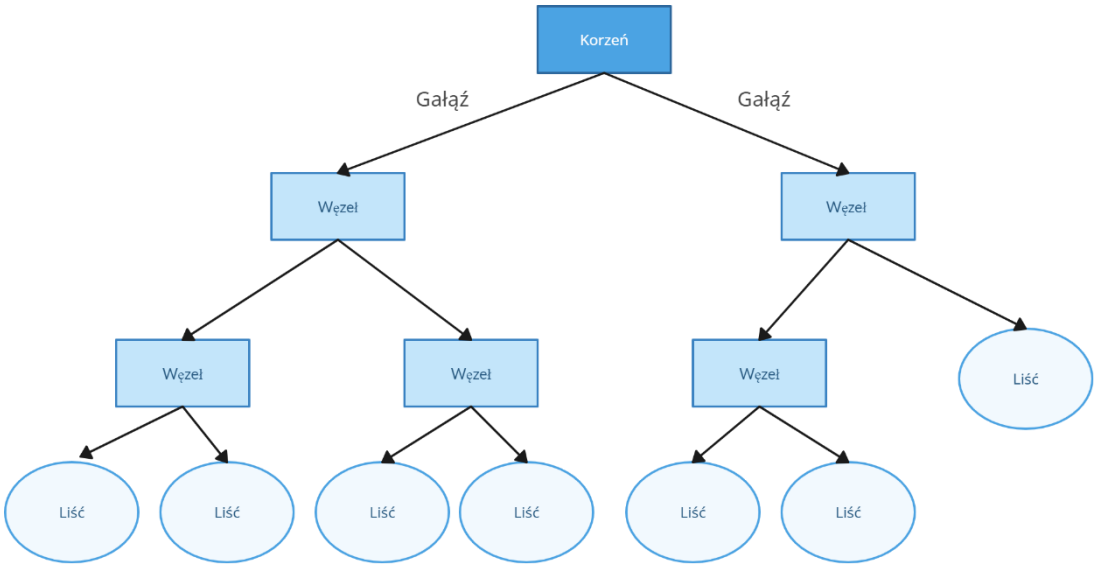
1. Ustalenie warunku stopu
2. Wybranie korzenia
3. Wybór kryterium podziału
4. Podział węzła decyzyjnego

[27] N.H. Son, *Drzewa decyzyjne*;
[28] A. Król-Nowak, K. Kotarba, *Podstawy uczenia maszynowego,* wydawnictwo AGH, Kraków 2022r., s.113;

5. Dla każdego powstałego węzła sprawdzane jest kryterium stopu:
   a) Jeżeli warunek stopu nie jest spełniony powrót do punktu 3
   b) Jeżeli warunek stopu jest spełniony to węzeł staje się liściem
6. Wykonywanie kroków 3-5 do czasu, aż każdy węzeł stanie się liściem
7. Likwidacja w procesie przycinania drzewa fragmentów o małym znaczeniu predykcyjnym.
8. Zastosowanie drzewa do predykcji na nieznanych danych.

Ze względu na skłonność algorytmów drzew decyzyjnych do przeuczenia się, czyli brakiem zdolności modelu do uogólnienia, co wpływa na niska jakość predykcji. Aby uniknąć nadmiernego dopasowania w trakcie tworzenia drzewa, ważne jest, aby przestrzegać zasady dążenia do konstruowania drzewa o jak najmniejszej złożoności. Popularną metodą zwalczająca zjawisko przeuczenia jest przycinanie drzewa. Najpopularniejsza metoda jest przycinanie wsteczne, polega ono na generowaniu dużego, dobrze dopasowanego do danych drzewa, a następnie usuwanie od dołu najmniej efektywnych gałęzi. Przycinanie drzewa oprócz zwalczania zjawiska przeuczenia, poprawia interpretowalność modelu dla użytkownika oraz zmniejsza złożoność obliczeniową co prowadzi do mniejszego czasu trenowania.

Ilustracja 2. Schemat budowy drzewa decyzyjnego.



Źródło: Opracowanie własne.

Zagrożenia podczas stosowania algorytmów drzew decyzyjnych:

1. Algorytmy łatwo ulegają przeuczeniu, zbyt dobrze dopasowują się do danych treningowych przez co istnieje później problem słabej generalizacji danych co wpływa na jakość predykcji na zbiorze testowym.
2. Brak uwzględnienia zmiennych o dużej sile predykcyjnej w przypadku jej silnej korelacji z innymi zmiennymi.
3. Małe uwzględnienie zmiennych o niskiej sile predykcyjnej.
4. Brak spójności wyników, jeżeli do budowania drzewa wykorzystywane są różne zbiory treningowe i testowe lub wprowadzone zostały nawet niewielkie zmiany w danych, może prowadzić to do powstawania różnych drzew klasyfikacyjnych.

2.1.2. Podejście wielomodelowe

Zważywszy na opisane wyżej problemy przy zastosowaniu drzew decyzyjnych, często stosuje się procedury agregacyjne, takie jak boosting, bagging (agregacja bootstrapowa) oraz lasy losowe. Te metody opierają się na stworzeniu grupy klasyfikatorów, mających na celu zmniejszenia wariancji modelu bez zwiększania obciążenia. Innymi słowy, tworzone jest wiele modeli, następnie ich wyniki są łączone w jeden, w celu uzyskania lepszej precyzji i bardziej stabilnych rezultatów. W pracy zostały użyte metody lasów losowych oraz wzmacniania gradientu. Metoda lasów losowych jest bezpośrednio oparta na metodzie baggingu dlatego również zostanie ona opisana. Eugeniusz Gatnar[29] zdefiniował agregacje modeli jako „*Agregacja modeli polega na wyodrębnieniu V prób uczących, według których budowane są modele w postaci drzew* $D_1(x), \dots, D_v(x)$*, łączonych następnie w jeden model zagregowany* $D^*(x)$*”*. Metoda baggingu zwana również agregacja bootstrapową jest jedną z pierwszych metod agregacji modelów. Polega ona na zbudowaniu M modeli bazowych na podstawie prób uczących $U_1, \dots, U_m$ losowanych ze zwracaniem ze zbioru uczącego[30],

[29] E. Gatnar, *Agregacja modeli dyskryminacyjnych*, Prace Naukowe Akademii Ekonomicznej we Wrocławiu. Taksonomia , Wrocław 2002r., s.219;
[30] P. Marcin, *Podejście wielomodelowe w regresji danych symbolicznych interwałowych*, Prace Naukowe Uniwersytetu Ekonomicznego we Wrocławiu, s.211-220;

każda z prób uczących zawiera n obiektów przy czym waga każdego wylosowanego obiektu jest jednakowa i wynosi:

$$w_i = \frac{1}{n} \qquad (14)$$

Innymi słowy podstawą baggingu jest wielokrotne losowanie ze zwracaniem podzbiorów danych, w wyniku czego powstają nowe zbiory uczące na podstawie których budowane są modele bazowe. Część danych która nie zostaje wylosowana tworzy zbiór walidacyjny (ang. Out-of-bag), który może być stosowany jako dodatkowy zbiór testowy. Predykcja modelu zagregowanego oparta jest na głosowaniu większościowym. W klasyfikacji klasa obiektu przypisywana jest na podstawie klasy dominującej spośród modeli bazowych.

Algorytm baggingu można zobrazować w następujących krokach:

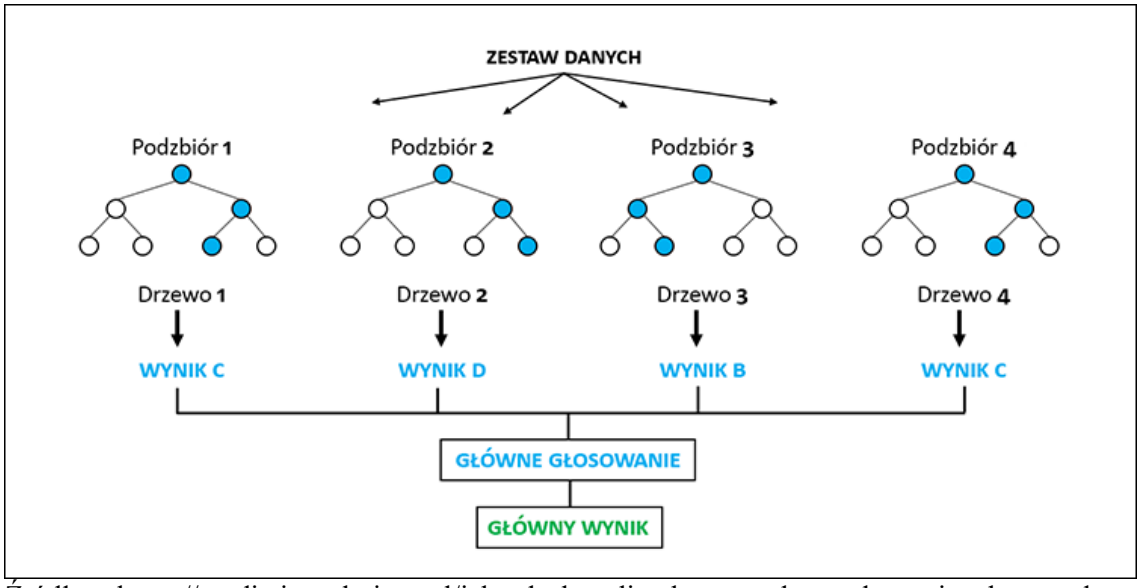
1. Ustalenie liczby modeli bazowych.
2. Dla każdego modelu bazowego wylosowanie próby bootstrapowej, a następnie budowa modelu na jej podstawie.
3. Agregacja modeli i wybór klasy obiektu na podstawie głosowania większościowego.

Podczas działania algorytmu baggingu do tworzenie każdego modelu bazowego wykorzystywane są te same cechy, w przypadku kiedy występują cechy silnie skorelowane ze zmienną objaśnianą może to prowadzić do wysokiej korelacji miedzy modelami bazowymi co skutkuje obniżoną efektywnością całego procesu. Jednym ze sposobów na radzenie sobie z tym problemem jest zastosowanie algorytmu lasów losowych. Działanie tego algorytmu oparte jest na tej samej zasadzie co baggingu z tą różnicą, że przy budowie drzew w każdym węźle wybierany jest tylko pewien losowy podzbiór cech który jest znacznie mniejszy od oryginalnego zestawu. Z tego podzbioru wybierana jest cecha o najmniejszym wskaźniku błędu i to na jej podstawie tworzony jest podział. To sprawia, że algorytm drzew losowych dobrze radzi sobie z wielowymiarowymi zestawami danych, a drzewa są bardziej różnorodne i mniej skorelowane. Na koniec tak samo jak w algorytmie baggingu, klasa obiektu przypisywana jest za pomocą głosowania większościowego.

Algorytm lasów losowych można przedstawić w następujących krokach:

1. Ustalenie liczby modeli bazowych oraz liczby zmiennych.
2. Dla każdego modelu bazowego wylosowanie próby bootstrapowej,
   a następnie budowa drzewa na jej podstawie, przy czym w każdym węźle losowana bez zwracania jest ustalona z góry liczba cech z których wybierana jest najlepsza i na jej podstawie ustalany jest podział.
3. Agregacja modeli i wybór klasy obiektu na podstawie głosowania większościowego.

Ilustracja 3. Schemat budowy algorytmów lasów losowych.



Źródło: https://predictivesolutions.pl/jak-udoskonalic-algorytm-drzew-decyzyjnych - odczyt 20.07.2023r.

Kolejnym algorytmem który używa agregacji jest wzmacnianie gradientu(ang. Gradient boosting). W przeciwieństwie do algorytmów baggingu oraz lasów losowych gdzie tworzone drzewa nie są od siebie zależne, w tym wypadku drzewa tworzone są sekwencyjnie. Każde kolejne utworzone drzewo działa w oparciu o poprzednie. Idea wzmacniania gradientu jest minimalizowane funkcji straty która opisuje różnice pomiędzy wartością prognozowaną, a wartością rzeczywistą. W tym celu tworzony jest duży model złożony z wielu cech, które pojedynczo wykazują mniejszą zdolności predykcyjną. W przypadku wzmacniania gradientu drzew decyzyjnych początkowo trenowany jest prosty model drzewa decyzyjnego, a następnie w każdej kolejnej iteracji

za pomocą gradientu trenowany jest model który ma na celu minimalizacje błędów resztkowych poprzedniego. W teorii każdy kolejny model w algorytmach boostingu powinien minimalizować błędy popełnione przez poprzednie modele, może to jednak prowadzić do nadmiernego dopasowania modelu, w którym model staje się zbyt skomplikowany i traci zdolność do uogólniania. Dlatego najważniejszymi parametrami przy stosowaniu algorytmów opartych na boostingu są złożoność modelu oraz liczba modeli bazowych, odpowiednie dobranie tych parametrów jest kluczowe dla uzyskania równowagi między dopasowaniem modelu do danych treningowych, a umiejętnością dopasowania się do nieznanych obserwacji.

Schemat tworzenia algorytmu wzmocnienia gradientu:

1. Obliczanie średniej wartości zmiennej wynikowej oraz resztek dla każdej próby.
2. Dla każdej iteracji:
   a) Dopasowywanie drzewa regresji do pseudoreszt.
   b) Tworzymy predykcje skorygowaną o pseudoreszty przemnożone przez współczynnik uczenia.
   c) Obliczamy nowe resztki
3. Powtarzamy krok 2 określoną liczbę razy.
4. Tworzymy ostateczną predykcje z użyciem wszystkich modeli składowych.

2.2. Przedstawienie wcześniejszych opracowań dotyczących przewidywania wyników zawodów sportowych z użyciem uczenia maszynowego

Na przestrzeni ostatnich lat, metody uczenia maszynowego stają się coraz bardziej popularne w prognozowaniu wyników sportowych. Rozwój ten można przypisać rosnącemu zainteresowaniu sportem i jego analizą. W szczególności, metody uczenia nadzorowanego, takie jak regresja liniowa, drzewa decyzyjne czy sieci neuronowe stanowią skuteczne narzędzie do prognozowania wyników meczów w piłce nożnej i innych dyscyplinach sportowych.

W artykule „Neural network quarterbacking” Michael C. Purucker [31]przeprowadził jedno z pierwszych badań dotyczących prognozowania wyników w National Football League (NFL) za pomocą modelu sieci neuronowej (ANN). Do badań wykorzystano dane z pierwszych ośmiu rund zawodów i pięciu parametrów: zdobytych jardów, zdobytych jardów poprzez przemieszczanie, liczby strat, czasu posiadania i kursów bukmacherskich. Do rozróżniania wysoko i nisko skutecznych drużyn użyto nieuporządkowanych metod opartych na klastrowaniu. Następnie wykorzystano ANN z algorytmem uczenia wstecznego. Osiągnięta dokładność wyniosła 60,7%, podczas gdy efektywność prognoz ekspertów z tej dziedziny wyniosła 72%. Stwierdzono, że algorytm wstecznej propagacji jest najskuteczniejszym podejściem. Jako ograniczenie tego badania, można wskazać niską liczbę użytych cech.

W 2003 roku, J. Kahn [32]osiągnął większą dokładność. Wynik ten był jednocześnie minimalnie wyższy od rezultatu ekspertów NFL, którzy przeprowadzali prognozy tych samych meczów. Badacz zebrał dane dotyczące 208 meczów w sezonie 2003/2004. Cechy, jakich użył to: różnica w ogólnej liczbie jardów, różnica w liczbie zdobytych jardów poprzez przemieszczanie, różnica w liczbie strat, wskaźnik drużyny gości i wskaźnik drużyny gospodarzy. Badanie przeprowadzono w formie problemu klasyfikacyjnego. Pierwsze 192 mecze służyły jako zestaw danych treningowych, a pozostałe kolejki (z 14 i 15 tygodnia) zostały użyte jako zestaw testowy. Podczas testowania wyznaczono 10-3-2 na optymalna strukturę sieci. W ten sposób osiągnięto dokładność wynoszącą 75% we wszystkich meczach w tygodniach 14 i 15. Wynik został porównany z prognozami ośmiu komentatorów sportowych ze stacji ESPN. W meczach z 14 tygodnia, eksperci z dziedziny prognozowali poprawnie średnio 63% meczów, a w meczach z 15 tygodnia osiągnęli skuteczność wynoszącą 87%.

Z kolei, Dragan Miljković zastosował metodę naiwnego klasyfikatora bayesowskiego, oraz algorytm wektorów nośnych do prognozowania wyników meczów koszykówki. Do trenowania modelu użył danych dotyczących statystyk meczowych zawodników oraz formy drużyny. W konsekwencji osiągnął dokładność wynoszącą 67%, która może być uznana za satysfakcjonujący wynik w porównaniu z 68,3% osiągniętym przez ekspertów.[33]

---

[31] M. C. Purucker, *Neural network quarterbacking vol. 15*, IEEE Potentials 1996r., s. 9-15;
[32] A, Uberoi, Introduction to Dimensionality Reduction, ECE539 2003r.;
[33] D. Miljković, L. Gajić, A. Kovačević, Z. Konjović*, The Use of Data Mining for Basketball Matches Outcomes Prediction*, Faculty of Tehnical Sciences, Novi Sad, Republic of Serbia 2010r.;

Analogicznie Alan McCabe oraz Jarrod Trevathan [34] próbowali przewidywać wyniki w czterech różnych sportach: NRL (National Rugby League), AFL (Australian Rules Football), Super Rugby (Rugby Union) i English Premier League Football (EPL). Do tego celu wykorzystali dane z roku 2002 dotyczących formy zawodników, rezultatów poszczególnych drużyn z ostatnich meczów oraz mocy drużyny określanej poprzez ranking. Użyli wielo-warstwowego neuronu, wytrenowanego za pomocą algorytmów BP oraz metod gradientu sprzężonego. Do badania użyto identycznych funkcji we wszystkich sportach. Średnia dokładność algorytmu ANN dla rozgrywek NRL wyniosła 63,2% dla AFL 65,1%, a Super Rugby 67,5%. Wyniki te były zbliżone do przewidywań ekspertów znajdujących się na poziomie 60-65%. Podobnie, dla rozgrywek piłkarskich osiągnięta została skuteczność 54,6%, podczas gdy eksperci z tej dziedziny osiągają w szacunkach wyniki na poziomie 50-55%.

W przypadku prognoz meczów piłkarskich istnieją dwie główne strategie. Pierwsza z nich traktuje ten problem jako zadanie klasyfikacji, w której wynik meczu jest prognozowany jako jedna z trzech kategorii: wygrana, remis lub przegrana. Druga strategia polega na traktowaniu tego problemu jako zadania regresji, w którym przewiduje się liczbę zdobytych goli przez obie drużyny, a następnie, na podstawie tych przewidywań, określa się ostateczny rezultat meczu. W artykule" *Predicting The Dutch Football Competition Using Public Data*" skupiono się na wykorzystaniu publicznie dostępnych danych z 13 sezonów holenderskiej ligi piłkarskiej. W celu doboru odpowiedniego zestawu danych zastosowano techniki redukcji wymiarowości. W kolejnym kroku zastosowano techniki klasyfikacyjne. Największą dokładność na danych publicznych (54,702%) osiągnięto poprzez kombinacje PCA z 15% wariancja oraz klasyfikatorem naiwnym Bayesa. Autor zauważył, że model osiąga niewystarczająco dokładne wyniki w wykrywaniu remisów, co może być spowodowane ich niską liczbą w zbiorze danych (tylko 22% wszystkich rezultatów). Ciekawym aspektem tego badania jest również porównanie modeli zbudowanych z danych ogólnie dostępnych, do modelu opartego na kursach bukmacherów. Klasyfikator naiwny Bayesa który osiągał najadekwatniejsze efekty, budowany na kursach bukmacherskich uzyskał najmniejszą dokładność. Porównując wszystkie wykorzystane modele, najwyższą precyzję wyników osiągnął model hybrydowy łączący dane dotyczące statystyk meczowych oraz kursów

[34] A. McCabe, J. Trevathan, *Artificial intelligence in sports prediction*, Information Technology: New Generations, ITNG 2008r.;

ustalanych przez bukmacherów. Problem niedoszacowania remisów występował również u Bena Ulmera i Matthew Fernandeza[35]. Badacze dostrzegli, że model zbudowany z wektorów nośnych, który najdokładniej przewidywał poprawne rezultaty, w przypadku meczów skończonych remisem, osiągał tylko 1% poprawnych predykcji (poprawnie wytypował 3 z 294 meczów zakończonych wynikiem remisowym). Również Menno Heijboer [36] zwrócił uwagę na problem niedoszacowania remisów, który występował we wszystkich stosowanych przez niego algorytmach. W celu uniknięcia tego efektu zastosowano dwie techniki: SMOTE[37] i ClusterCentroids. Technika SMOTE (Synthetic Minority Over-sampling Technique) polega na tworzeniu syntetycznych punktów danych na podstawie wyników sąsiednich punktów danych, dzięki czemu nie są one dokładnymi replikami oryginalnych danych. To zmniejsza ryzyko nadmiernego dopasowania (overfittingu) i jednocześnie ułatwia algorytmom dopasowanie granicy decyzyjnej. Druga metoda, ClusterCentroids (CC), to podejście do undersamplingu, w którym nowy zestaw danych jest generowany na podstawie centroidów - klastrów uzyskanych za pomocą algorytmu K-średnich. Jedną z zalet metody CC jest to, że dotyczy ona tylko klasy większościowej, pozostawiając klasy mniejszościowe nienaruszone. Najdokładniejsze wyniki w kontekście remisów osiągnęły algorytmy drzewa losowego oraz gradient boostingu w połączeniu z metodą CC. Jednak po analizie wyników Hejiboer stwierdził, że skoncentrowanie się na przewidywaniu remisów nie wpływa znacząco na ogólną wydajność modelu. Jako najbardziej wartościowe cechy w przewidywaniu wyników ocenił siłę drużyny, siłę defensywy oraz ranking ELO. Siła defensywy została również wyznaczona jako najważniejsza cecha w badaniach prowadzonych przez Prasetio, D. i Harlili, D [38]którzy do predykcji korzystali, między innymi, z oceny zawodników z FIFA, a ich algorytm regresji logistycznej osiągnął dokładność 69.5%.

João Gomes, Filipe Portela oraz Manuel Filipe Santos do utworzenia modelu również użyli publicznych danych z podobną charakterystyką. Tym razem jednak dane służyły do przewidzenia wyników meczów angielskiej Premier League. Porównano

---

[35] B. Ulmer, M.P. Fernández, *Predicting Soccer Match Results in the English Premier League*, Stanford University 2014r.;
[36] M. Heijboer, *Predicting football match outcomes using machine learning algorithms,* Tilburg University 2022r.;
[37] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, *SMOTE: Synthetic Minority Over-sampling Technique,* AI Access Foundation and Morgan Kaufmann Publishers 2002r.;

[38] D. Prasetio, D. Harlili, *Predicting football match results with logistic regression*, International Conference On Advanced Informatics: Concepts, Theory And Application ICAICTA 2016r., s. 1-5;

metody NB (naive bayes), decision tree oraz svm. Zauważono, że svm osiągnęła największą dokładność wynoszącą 50,8%, co było gorszym wynikiem w porównaniu do ligi holenderskiej. Autor przetestował swój model na danych z 7 kolejek Premier League. W porównaniu do kursów bukmacherów udało im się osiągnąć zadowalającą marżę 20%.

Ponadto Huang K Y i Chang W L [39] stworzyli model przewidujący wyniki mistrzostw świata w 2006 roku bazujący na statystykach obu drużyn z poprzedniej fazy rozgrywek. Dane zostały utworzone w formie relacji statystyk obu drużyn, a jako algorytm został wybrany percepton wielowarstwowy z regułą uczenia wstecznego z propagacją błędu. Architekturę sieci neuronowej wyznaczono na 8 wejść, 11 węzłów ukrytych i 1 wyjście (8-11-1) i osiągnięto znaczną dokładność 76,9% przy wykluczeniu remisów. W przeciwieństwie do większości badań, które oparte są na statystykach meczowych lub formie, Johannes Stübinger, Benedikt Mangold i Julian Knoll[40] skupili się na utworzeniu modelu opartego na charakterystyce zawodników. Wybrali do tego takie dane jak wzrost, waga, wiek oraz statystyki wyrażające wartość piłkarską zawodników. Do przewidywania wyników posłużyli się metodą regresji oraz algorytmami drzew losowych, boostingu, regresji liniowej oraz maszyn wektorów wsparcia. Ze wszystkich badanych algorytmów najlepsze wyniki osiągnął algorytm lasów losowych.

Kolejnym artykułem, w którym autorzy sprawdzili metody regresji w kontekście przewidywania meczów jest „*Predicting Final Result of Football Match Using Poisson Regression Model*”[41]. W badaniu, do przewidzenia rezultatów meczów została wykorzystana metoda regresji Poissona. Autorzy przewidzieli średnią liczbę goli zdobytych przez każdą drużynę zakładając, że liczba goli zdobytych przez drużyny piłkarskie podlega jednowymiarowemu rozkładowi Poissona. Model został utworzony z czterech cech: średniej goli zdobytych w meczach, przewadze własnego boiska, sile ofensywnej gospodarza oraz sile defensywnej przeciwnika. Predykcja została wykonana na 100 meczach sezonu 2017-2018 Premier League, a osiągnięta dokładność wynosiła 61%.

---

[39] K. Y. Huang, W. L. Chang, *A neural network method for prediction of 2006 world cup football game,* International Joint Conference on Neural Networks (IJCNN) 2010r., s. 1-8;

[40] J. Stübinger, B, Mangold, J. Knoll, *Machine Learning in Football Betting: Prediction of Match Results Based on Player Characteristics,* University of Erlangen-Nürnberg 2019r.;

[41] H. R. Azhari, Y. Widyaningsih, D. Lestari, *Predicting Final Result of Football Match Using Poisson Regression Model,* Universitas Indonesia, Kampus Baru UI 2018r.;

Przywołane artykuły opisywały metody i rezultaty przewidywania wyników meczów przed ich rozpoczęciem. Ciekawe badania na temat przewidywania rezultatów w trakcie trwania meczu przeprowadzili Pieter Robberechts, Jan Van Haaren i Jesse Davis. W ich badaniach ponownie pojawił się problem poprawnego przewidywania remisów. Ze sprawdzanych algorytmów jedynie klasyfikator bayesowski posiadał dobrze skalibrowane prawdopodobieństwa wygranej, remisu i przegranej. W wyniku działania modelu zostało zaobserwowane, że zwycięstwa w większej ilości pojedynków ma negatywny wpływ na prawdopodobieństwo zwycięstw. Natomiast wyższy ranking Elo, uprzednio zdobyte gole, żółte kartki dla przeciwnika oraz udane podania progresywne mają pozytywny wpływ na współczynnik zdobywania bramek.

2.3. Opis zbioru danych

Zbiór danych zawiera statystyki ze wszystkich meczów ostatnich 10 sezonów Premier League. Dane dotyczące meczów pochodzą ze strony England Football Results Betting Odds | Premiership Results & Betting Odds (football-data.co.uk), a wartości składów został pobrana ze strony transfermarkt. Na danych z sezonów 2013/2014-2021/2022 została przeprowadzona analiza oraz zbudowany model, natomiast na sezonie 2022/2023 została wykonana predykcja. Ze względu na różnicę dostępności danych (pierwotnie dane obejmowały zdarzenia z każdego meczu, podczas gdy przy wykonywaniu predykcji dla nowych spotkań takie dane nie byłyby dostępne) zmienne zostały przekształcone tak, aby dotyczyły wyłącznie wcześniejszych wydarzeń. Zmienne zostały podzielone na kilka kategorii. Każda z nich odnosi się do kluczowych aspektów analizy związanej z meczami piłki nożnej. Poniżej został przedstawiony opis każdej kategorii:

Forma

Zmienne opisujące formę drużyn odnoszą się do średniej zdobytych punktów w ostatnich 5 lub 3 meczach. Zmienna HT_AvgPointsCurrentS oraz AT_AvgPointsCurrentS wyrażają pozycje w tabeli w obecnym sezonie wyrażoną za pomocą średniej zdobytych punktów w dotychczasowych meczach trwającego sezonu.

Tabela 2. Zmienne dotyczące formy zespołu.

Nazwa zmiennej	Opis	Typ zmiennej
HT_pointsLast3g	średnia punktów zdobytych przez drużynę gospodarzy w 3 ostatnich meczach	Numeryczna (od 0 do 3)
AT_pointsLast3g	średnia punktów zdobytych przez drużynę gości w 3 ostatnich meczach	Numeryczna (od 0 do 3)
HT_poinstLast5g	średnia punktów zdobytych przez drużynę gospodarzy w 5 ostatnich meczach	Numeryczna (od 0 do 3)
AT_pointsLast5g	średnia punktów zdobytych przez drużynę gości w 5 ostatnich meczach	Numeryczna (od 0 do 3)
HT_AvgPointsCurrentS	średnia punktów drużyny gospodarzy w obecnym sezonie	Numeryczna (od 0 do 3)
AT_AvgPointsCurrentS	średnia punktowo drużyny gości w obecnym sezonie	Numeryczna (od 0 do 3)

Źródło: Opracowanie własne.

Historyczne rezultaty

Aby wyrazić siłę drużyny w kontekście historycznym i dostarczyć do modelu informacji na temat wyników poszczególnych zespołów w poprzednich latach, zostały utworzone następujące zmienne:

Tabela 3. Zmienne dotyczące historycznych rezultatów.

Nazwa zmiennej	Opis	Typ zmiennej
HT_AvgPLastS	średnia punktów z ostatniego sezonu dla drużyny gospodarzy	Numeryczna (od 0 do 3)
AT_AvgPLastS	średnia punktów z ostatniego sezonu dla drużyny gości	Numeryczna (od 0 do 3)
HT_AvgPLast2S	średnia punktów z dwóch ostatnich sezonów dla drużyny gospodarzy	Numeryczna (od 0 do 3)
AT_AvgPLast2S	średnia punktów z dwóch ostatnich sezonów dla drużyny gości	Numeryczna (od 0 do 3)

Źródło: Opracowanie własne.

Przewaga własnego boiska

Na temat przewagi własnego boiska zostało przeprowadzone wiele badań. A. M. Nevill, Sue M. Newell oraz Sally Gale[42] potwierdzili istnienie przewagi własnego boiska w 8 ligach angielskich. W badanym przez nich sezonie 60% meczów kończyło się przewagą gospodarza. Autorzy zauważyli, że w ligach, w których występuje wzmożona frekwencja kibiców, przewaga własnego boiska odgrywa większe znaczenie. Anna Waters [43] w swoich badaniach spróbowała wyjaśnić przyczynę tego zjawiska. Jako główne powody podała stan psychiczny, różnicę w nastawieniu oraz lepszą jakość snu. Dla zobrazowania wspomnianego efektu zostały porównane ze sobą 3 tabele przedstawiające rezultat końcowy sezonu 2021/2022 Premier League.

[42]A. M. Nevill, S. M. Newell, S. Gale, *Factors associated with home advantage in English and Scottish soccer matches,* „Journal of Sports Sciences" 1996r., 14:2, s. 181-186;

[43] A. Waters, G. Lovell, *An Examination of the Homefield Advantage in a Professional English Soccer Team from a Psychological Standpoint*, „Football Studies" vol. 5 no. 1 2002r.;

Ilustracja 4. Tabele sezonu 2021/2022 Premier League z podziałem na mecze domowe i wyjazdowe.

Team	P	W	D	L	GF	GA	GD	Pts
1 Manchester City	19	17	1	1	60	17	+43	52
2 Manchester United	19	15	3	1	36	10	+26	48
3 Arsenal	19	14	3	2	53	25	+28	45
4 Liverpool	19	13	5	1	46	17	+29	44
5 Newcastle	19	11	6	2	36	14	+22	39
6 Aston Villa	19	12	2	5	33	21	+12	38
7 Brentford	19	10	7	2	35	18	+17	37
8 Tottenham	19	12	1	6	37	25	+12	37
9 Brighton	19	10	4	5	37	21	+16	34
10 Nottingham Forest	19	8	6	5	27	24	+3	30
11 Wolves	19	9	3	7	19	20	-1	30
12 Fulham	19	8	5	6	31	29	+2	29
13 West Ham	19	8	4	7	26	24	+2	28
14 Crystal Palace	19	7	7	5	21	23	-2	28
15 Chelsea	19	6	7	6	20	19	+1	25
16 Bournemouth	19	6	4	9	20	28	-8	22
17 Leeds	19	5	7	7	26	37	-11	22
18 Everton	19	6	3	10	16	27	-11	21
19 Leicester	19	5	4	10	23	27	-4	19
20 Southampton	19	2	5	12	19	37	-18	11

Team	P	W	D	L	GF	GA	GD	Pts
1 Arsenal	19	12	3	4	35	18	+17	39
2 Manchester City	19	11	4	4	34	16	+18	37
3 Newcastle	19	8	8	3	32	19	+13	32
4 Brighton	19	8	4	7	35	32	+3	28
5 Manchester United	19	8	3	8	22	33	-11	27
6 Fulham	19	7	2	10	24	24	0	23
7 Liverpool	19	6	5	8	29	30	-1	23
8 Tottenham	19	6	5	8	33	38	-5	23
9 Aston Villa	19	6	5	8	18	25	-7	23
10 Brentford	19	5	7	7	23	28	-5	22
11 Chelsea	19	5	4	10	18	28	-10	19
12 Crystal Palace	19	4	5	10	19	26	-7	17
13 Bournemouth	19	5	2	12	17	43	-26	17
14 Everton	19	2	9	8	18	30	-12	15
15 Leicester	19	4	3	12	28	41	-13	15
16 Southampton	19	4	2	13	17	36	-19	14
17 West Ham	19	3	3	13	16	31	-15	12
18 Wolves	19	2	5	12	12	38	-26	11
19 Leeds	19	2	3	14	22	41	-19	9
20 Nottingham Forest	19	1	5	13	11	44	-33	8

Źródło:https://www.whoscored.com/Regions/252/Tournaments/2/Seasons/8618/Stages/19793/TeamStatistics/England-Premier-League-2021-2022 - data odczytu: 21.07.2023r.

Z analizy powyższych tabel wynika, że wszystkie kluby z wyjątkiem Southampton miały lepszą średnią punktów w meczach granych „u siebie" niż na wyjeździe. Warty uwagi jest fakt, że niektóre kluby osiągały zdecydowanie lepsze wyniki w meczach domowych niż wyjazdowych. Na przykład klub Nottingham Forest w tabeli meczów „u siebie" zajął 10 miejsce, a w tabeli wyjazdowej 20, zdobywając tylko 8 punktów w 19 meczach. Drużyna Wolves zdobyła tyle samo punktów w meczach domowych i tylko 2 punkty więcej na wyjeździe. Aby dostarczyć informacji na temat przewagi własnego boiska zostały wygenerowane następujące dane:

Tabela 4. Zmienne opisujące przewagę własnego boiska.

Nazwa zmiennej	Opis	Typ zmiennej
HT_pointsHLast3g	średnia punktów zdobytych przez drużynę gospodarzy w 3 ostatnich meczach domowych	Numeryczna (od 0 do 3)
AT_pointsALast3g	średnia punktów zdobytych przez drużynę gości w 3 ostatnich meczach wyjazdowych	Numeryczna (od 0 do 3)

HT_AvgGoalsHLastS	średnia strzelanych goli przez drużynę gospodarzy w jej meczach domowych w poprzednim sezonie	Numeryczna (od 0,7368 do 3,3158)
AT_AvgGoalsALastS	średnia strzelanych goli przez drużynę gości w jej meczach wyjazdowym w poprzednim sezonie	Numeryczna (od 0,5263 do 2,5263)

Źródło: Opracowanie własne.

Siły ofensywy i defensywy

W przywołanym wcześniej tekście Menno Heijboer uznał siły ofensywy i defensywy za zmienne mające największy wpływ na wynik meczu. Siła ofensywna została ujęta jako średnia strzałów, średnia celnych strzałów oraz średnia strzelanych goli, natomiast siła defensywy jako średnia straconych goli.

Tabela 5. Zmienne opisujące siłę ofensywy i defensywy.

Nazwa zmiennej	Opis	Typ zmiennej
HT_ShotsLast5g	średnia strzałów wykonanych przez drużynę gospodarzy w 5 ostatnich meczach	Numeryczna (od 5,2 do 24,6)
AT_ShotsLast5g	średnia strzałów wykonanych przez drużynę gości w 5 ostatnich meczach	Numeryczna (od 4,4 do 25)
HT_ShotsOTLast5g	średnia strzałów celnych wykonanych przez drużynę gospodarzy w 5 ostatnich meczach	Numeryczna (od 0,8 do 10,4)
AT_ShotsOTLast5g	średnia strzałów celnych wykonanych przez drużynę	Numeryczna (od 1 do 10,8)

	gości w 5 ostatnich meczach	
HT_Goals_Last5g	Średnia strzelonych goli w 5 ostatnich meczach przez drużynę gospodarzy	Numeryczna (od 0 do 4,8)
AT_Goals_Last5g	Średnia strzelonych goli w 5 ostatnich meczach przez drużynę gości	Numeryczna (od 0 do 4,4)
HT_GoalsConLast5	Średni a straconych goli w 5 ostatnich meczach przez drużynę gospodarzy	Numeryczna (od 0 do 3,6)
AT_GoalsConLast5	Średnia straconych goli w 5 ostatnich meczach przez drużynę gości	Numeryczna (od 0 do 4)

Źródło: Opracowanie własne.

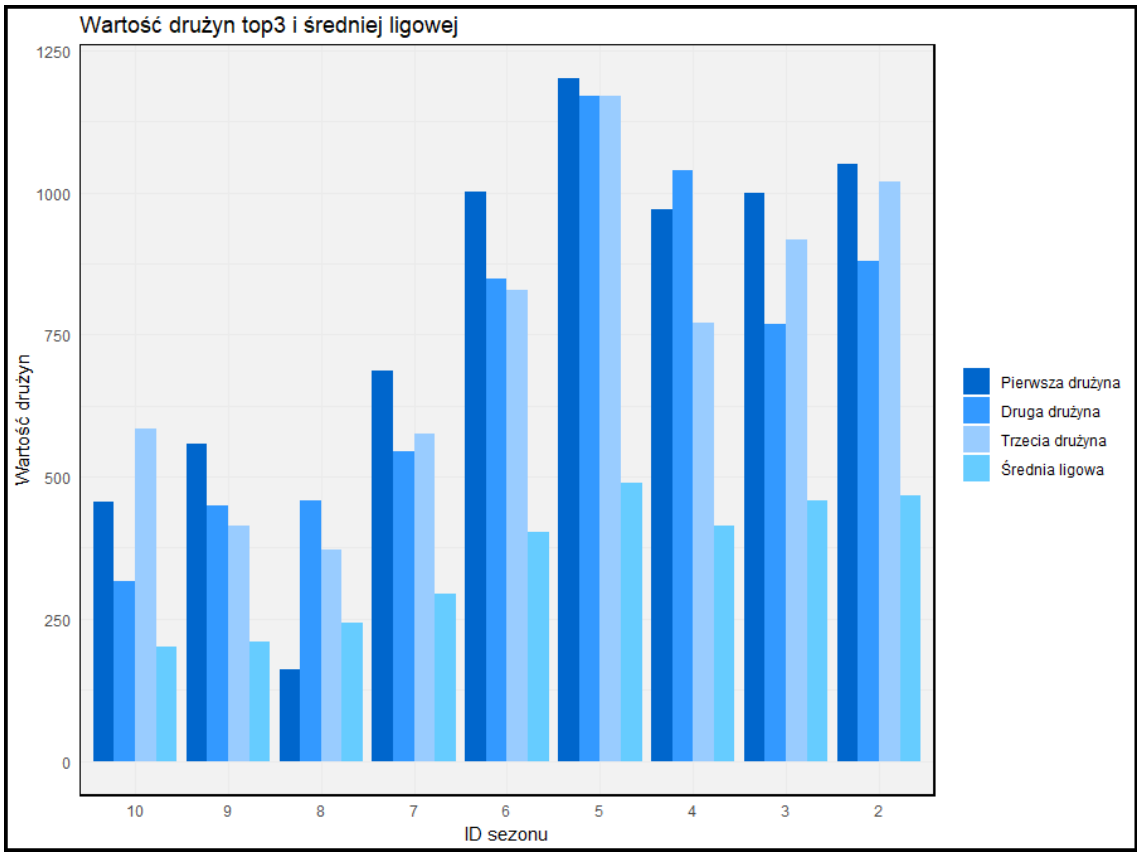
Wartość składu

Każde zawody sportowe charakteryzują się występowaniem faworytów oraz tzw. underdogów, czyli drużyn którym przypisywane są mniejsze szanse na wygraną. W trakcie analizy obecnego badania zostało zauważone, że w przeważającej większości sezonów czołówka ligowa miała znacznie wyższą wartość drużyny od średniej. Dodatkowo można zaobserwować wzrost tej zależności w ostatnich latach. Na przestrzeni badanych 9 sezonów tylko raz drużyna kończąca sezon na miejscach w czołówce „top 3" posiadała niższą wartość składu od średniej. Co więcej tylko w sezonie 2015/2016 (w którym drużyna Leicester sensacyjnie wygrała ligę z jedenastym budżetem) w drużynach, które zakończyły sezon na pierwszych trzech miejscach, znajdowały się grupy spoza top 5 największych budżetów w lidze. We wszystkich pozostałych sezonach można zaobserwować zależność, w której drużyny kończące ligę na miejscach top 3 plasowały się również w czołówce 5 najwyżej wartościowych składów. Z tej analizy można wyciągnąć wnioski, że drużyny z wyższą wartością składu posiadają większą szansę na korzystny rezultat. Na wykresie poniżej zostały przedstawione wartości drużyn kończących sezon na miejscach top3 oraz średnia wartość

drużyn w danym sezonie. Z wykresu można odczytać, przyspieszenie wzrostu średniej wartości drużyn w lidze od sezonu 2016/2017. Jako powód podaje się rekordowy kontrakt na sprzedaż praw do nadawania meczów Angielskiej Premier League. Z tego powodu zmienne zostały poddane skalowaniu min-maks dla każdego sezonu z osobna według podanego wzoru:

$$x_{new} = \frac{x_{old} - x_{min}}{x_{max} - x_{min}} \qquad (15)$$

Ilustracja 5. Wykres wartości drużyn.



Źródło: opracowanie własne

Tabela 6. Zmienne opisujące wartość składu.

Nazwa zmiennej	Opis	Typ zmiennej
HomeTeam_Value	wartość drużyny gospodarzy na danego początku sezonu.	Numeryczna (od 0 do 1)
AwayTeam_Value	wartość drużyny gości na początku danego sezonu.	Numeryczna (od 0 do 1)

Źródło: Opracowanie własne.

Bilans w spotkaniach bezpośrednich

Z zaobserwowanych przeze mnie zależności wynika, że niektóre drużyny preferują grę z niektórymi przeciwnikami. Warto dodać, że nie zostało to statystycznie potwierdzone - jest to jedynie moja obserwacja, którą zdecydowałem się sprawdzić.

Tabela 7. Zmienne opisujące wyniki w bezpośrednich starciach.

Nazwa zmiennej	Opis	Typ zmiennej
h2h_diff	różnica w punktach w meczach bezpośrednich między drużynami.	Numeryczna (od -3 do 3)

Źródło: Opracowanie własne.

Kursy bukmacherów

Kursy bukmachera bet365 na mecze ligowe, zostały wykorzystane do oceny jakości modelu.

Tabela 8. Zmienne opisujące kursy bukmachera.

Nazwa zmiennej	Opis	Typ zmiennej
B3651	kurs bukmachera bet365 na wygrana drużyny gospodarzy.	Numeryczna (od 1,06 do 23)
B365x2	kurs bukmachera bet365 na remis lub drużynę gości.	Numeryczna (od 1,005 do 11,333)

Źródło: Opracowanie własne.

Zmienna wynikowa

Zmienna opisująca rezultat meczu.

Tabela 9. Zmienna opisująca wynik spotkania.

Nazwa zmiennej	Opis	Typ zmiennej
HomeWin	Zmienna binarna opisująca zwycięstwo drużyny gospodarzy.	Kategoryczna (1 - gospodarze wygrali mecz 0 – brak wygranej drużyny gospodarzy)

Źródło: Opracowanie własne.

Rozkład zmiennej HomeWin prezentuje się następująco:

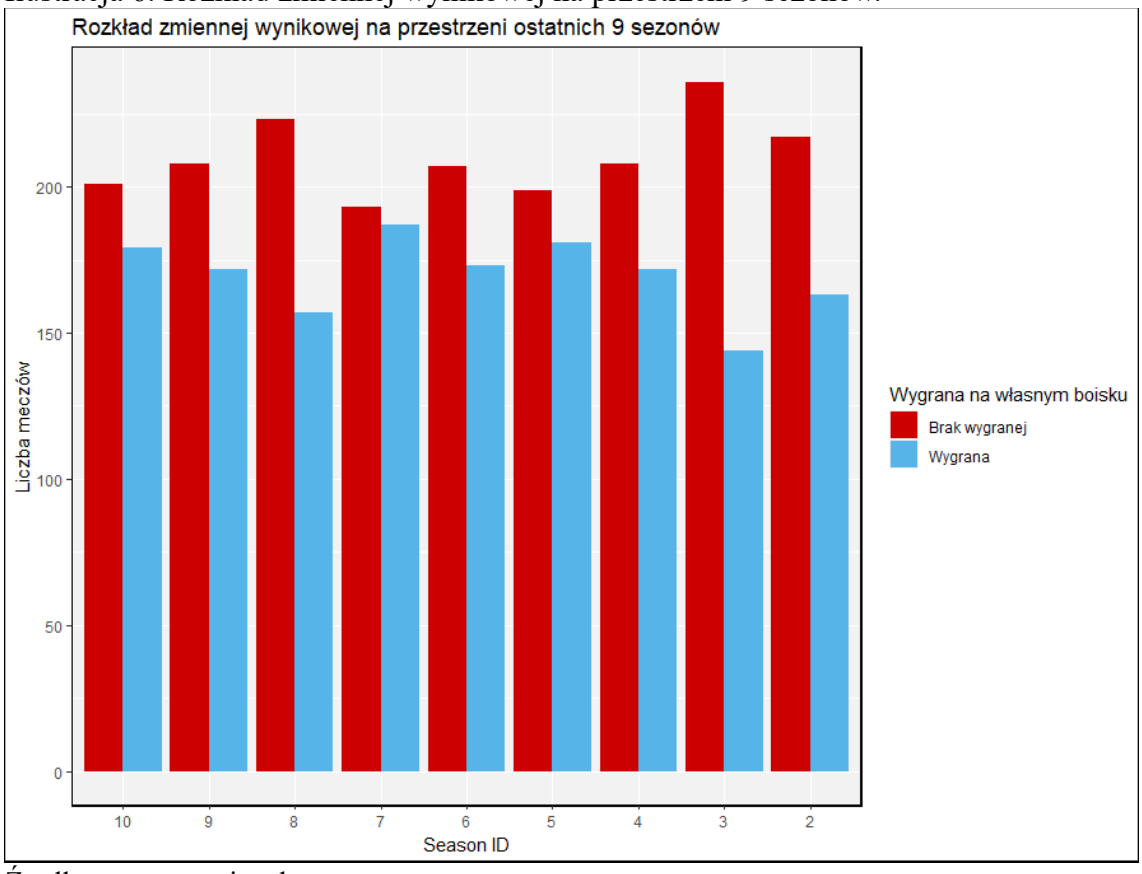
Tabela 10. Rozkład zmiennej wynikowej.

0 – Wygrana gospodarza	1 – remis lub wygrana gości
1892	1528

Źródło: Opracowanie własne.

55,32% meczów kończyło się wygraną gospodarzy, a 44,68% remisem lub wygraną drużyny przyjezdnej. Dla zobrazowania rozkładu zmiennej w całym zbiorze danych z podziałem na sezony został wygenerowany wykres słupkowy:

Ilustracja 6. Rozkład zmiennej wynikowej na przestrzeni 9 sezonów.



Źrodło: opracowanie własne.

Rozkład zmiennej wynikowej prezentuje się podobnie w badanych sezonach. Największe różnice zostały zaobserwowane w sezonie 2021/2022 oraz sezonie 2015/2016 kiedy to gospodarze wygrali odpowiednio 37,9% oraz 41,31% meczów.

Pozostałe zmienne

Tabela 11. Pozostałe zmienne użyte w pracy.

Nazwa zmiennej	Opis	Typ zmiennej
SeasonID	Zmienna opisująca przynależność do odpowiedniego sezonu, 1 oznacza najnowszy sezon(2022/2023), 10	Numeryczna (od 1 do 10)

	oznacza najstarszy badany sezon(2013/2014).	
HomeTeam	Opisuje nazwę drużyny gospodarzy	Zmienna tekstowa
AwayTeam	Opisuje nazwę drużyny gości	Zmienna tekstowa

Źródło: Opracowanie własne.

3. Prezentacja wyników badania empirycznego

3.1. Braki w danych

Większość braków w danych wystąpiła z powodu charakterystyki zmiennych opartych na kilku poprzednich kolejkach. Mecze z tych kolejek zostały usunięte. Pozostałe braki danych występowały w przypadku drużyn beniaminków, tzn. drużyn, które w danym sezonie awansowały do Premier League. Efekt ten wynikał z obecności zmiennych, które informowały o wydajności drużyny w sezonach poprzednich. Usuniecie tych obserwacji prowadziłoby do całkowitej utraty informacji na temat tych drużyn. Ze względu na różnice między Premier League, a drugim poziomem rozgrywkowym w Anglii oraz historię efektywności drużyn po awansie, uznane zostało za błędne wybranie statystyk opisujących grę drużyny w lidze niżej. Jako najlepsze podejście wybrano uzupełnienie brakujących wartości średnią z odpowiadających statystyk drużyn beniaminków z poprzedzającego sezonu. Pozostałe braki danych dotyczyły zmiennej opisującej liczbę zdobytych punktów w meczach bezpośrednich. Ponownie ten problem występował w przypadku drużyn beniaminków. W tej sytuacji jednak zdecydowano się zastąpić brakujące dane liczbą 1, co sugeruje wynik remisowy w ostatnim meczu rozgrywanym między tymi drużynami.

3.2. Analiza obserwacji odstających

Za obserwacje odstające uznawane są te, które znacząco różnią się od pozostałych elementów próby. Mogą być one większe lub mniejsze od pozostałych wyników. Efekt ten prowadzi do innego związku między tymi obserwacjami, a zmienną zależną, niż w pozostałych przypadkach. Występowanie obserwacji odstających może być spowodowane błędnymi obliczeniami, nieprawidłowym wprowadzeniem danych, wpływem rzadkich zjawisk lub innych anomalii występujących w zbiorze danych. Niosą one za sobą szereg problemów takich jak: zmiana średniej z próby, zmiana wariancji oraz innych statystyk opisowych. W przypadku uczenia maszynowego obserwacje odstające mogą wprowadzać niepotrzebny szum do bazy danych, doprowadzić model do błędnych predykcji lub do przeuczenia. Mimo, że nie ma jednej dobrej metody wykrywania obserwacji odstających, w literaturze zostało opisane wiele sposobów, które pomagają radzić sobie z tym zjawiskiem. W dalszej części zostały przeanalizowane metody IQR oraz z-score.

3.2.1. Metoda IQR

Wykrywanie obserwacji odstających za pomocą zakresu międzykwartylowego (różnica między trzecim i pierwszym kwartylem) jest jedną z najprostszych i najczęściej stosowanych metod. Obserwacja uznawana jest za odstającą, jeżeli znajduje się poniżej wartości dolnej granicy lub powyżej wartości górnej. Wzory potrzebne do zastosowania tej metody:

Rozstęp miedzykwartylowy: $IQR = Q_3 - Q_1$

Górna granica: $Q_3 + 1{,}5 * IQR$

Dolna granica: $Q_1 - 1{,}5 * IQR$

Tabela 12. Wyniki przeprowadzonej analizy metodą IQR.

Liczba_Obserwacji	Dolna_Granica	Górna_Granica	Nazwa_Zmiennej
0	-1	4,333333	HT_pointsHLast3g
0	-1,66667	3,666667	AT_pointsALast3g
0	-1,33333	4	HT_pointsLast3g
0	-0,5	3,5	AT_pointsLast3g
0	-0,7	3,3	HT_poinstLast5g
0	-1	3,8	AT_pointsLast5g
0	-4,5	4,3	h2h_diff
40	4,4	20,4	HT_ShotsLast5g
43	4,6	20,6	AT_ShotsLast5g
33	0,425	7,825	HT_ShotsOTLast5g
46	0,7	7,9	AT_ShotsOTLast5g
22	-0,7	3,3	HT_Goals_Last5g
29	-0,7	3,3	AT_Goals_Last5g
23	-0,2	3	HT_GoalsConLast5
14	-0,2	3	AT_GoalsConLast5
0	-0,11184	2,888158	HT_AvgPLast2S
0	-0,09211	2,855263	AT_AvgPLast2S
0	0,048246	2,75	HT_AvgPLastS
0	0,048246	2,75	AT_AvgPLastS
143	0,210526	2,736842	HT_AvgGoalsHLastS

0	-0,13158	2,605263	AT_AvgGoalsALastS
0	-0,67742	1,340136	HomeTeam_Value
0	-0,6269	1,25786	AwayTeam_Value
13	-0,1	2,833333	HT_AvgPointsCurrentS
12	-0,125	2,875	AT_AvgPointsCurrentS

Źródło: Opracowanie własne.

3.2.2. Metoda Z-score

Ta metoda mierzy, w jakiej odległości od średniej znajduje się wartość w jednostkach odchylenia standardowego. Dla każdej obserwacji obliczana jest statystyka Z:

$$Z_i = \frac{x_i - \bar{x}}{S_x} \qquad (16)$$

Gdzie:

x_i - wartość i – tej obserwacji.

$\bar{x}$ - średnia z próby.

S_x - odchylenie standardowe z próby.

Najczęściej przyjmowana wartość według której obserwacja jest uznawana za odstająca to 3 odchylenia standardowe, taka wartość została też przyjęta w moich obliczeniach.[44]

Tabela 13. Wyniki analizy przeprowadzonej metodą z-score.

Liczba_Obserwacji	Dolna_Granica	Gorna_Granica	Nazwa_Zmiennej
0	-0,90668	4,066846	HT_pointsHLast3g
0	-1,2956	3,660967	AT_pointsALast3g
0	-1,14665	3,838847	HT_pointsLast3g
0	-1,0965	3,938657	AT_pointsLast3g
0	-0,73307	3,454096	HT_poinstLast5g
0	-0,66911	3,480953	AT_pointsLast5g
0	-5,06366	4,896154	h2h_diff

[44] P. Denderski, *Metody identyfikacji obserwacji odstających*, Instytut Nauk Ekonomicznych Polskiej Akademii Nauk University of Leicester, luty 2019r., s33;

12	3,287127	21,7346	HT_ShotsLast5g
13	3,332357	22,16848	AT_ShotsLast5g
14	0,123843	8,380196	HT_ShotsOTLast5g
14	0,127261	8,561457	AT_ShotsOTLast5g
22	-0,6457	3,32397	HT_Goals_Last5g
29	-0,62556	3,367068	AT_Goals_Last5g
23	-0,45767	3,199031	HT_GoalsConLast5
14	-0,45894	3,132062	AT_GoalsConLast5
0	0,153629	2,644183	HT_AvgPLast2S
0	0,155299	2,640915	AT_AvgPLast2S
0	0,126292	2,70023	HT_AvgPLastS
0	0,127696	2,697141	AT_AvgPLastS
32	0,008869	3,069712	HT_AvgGoalsHLastS
16	-0,07071	2,524551	AT_AvgGoalsALastS
0	-0,61602	1,267889	HomeTeam_Value
0	-0,61519	1,265777	AwayTeam_Value
1	-0,19595	2,936893	HT_AvgPointsCurrentS
4	-0,17905	2,946498	AT_AvgPointsCurrentS

Źródło: Opracowanie własne.

W przypadku obu metod obserwacje odstające zostały usunięte z wyjątkiem zmiennych HT_AvgGoalshLastS oraz AT_AvgGoalsALastS. Z uwagi na to, że obie zmienne opisują średnią liczbę goli zdobywanych w poprzedzającym sezonie, pozbycie się tych zmiennych skutkowałoby usunięciem wszystkich meczów drużyn, które w poprzedzającym sezonie ligowym zdobyły najwięcej goli. W obawie o potencjalne straty informacyjne w wyniku usunięcia tych obserwacji, postanowiono zlogarytmować wyżej wspomniane zmienne. Poprzez logarytmowanie można zmniejszyć efekt obserwacji odstających, jednocześnie zachowując zawarte w nich informacje. Oba zestawy danych utworzone w wyniku usunięcia obserwacji zostały ze sobą porównane w kontekście precyzji działania modelów. Zaobserwowano, że model oparty na danych, w których dokonano redukcji przy użyciu metody Z-score, przyniósł lepsze wyniki. Może to być spowodowane mniejszą ilością usuniętych obserwacji w porównaniu do drugiego

zestawu. W związku z tym, dalsza analiza zostanie oparta na tym zestawie danych zawierającym 2746 obserwacji.

3.3. Analiza podstawowych statystyk opisowych

Tabela 14. Podstawowe statystyki opisowe użytych zmiennych.

Nazwa zmiennej	Średnia	Odchylenie standardowe	Median	Min	Max	Rozstęp	Skośność	Kurtoza	Błąd standardowy
HT_pointsHLast3g	1,570527	0,819208	1,333333	0	3	3	0,052096	-0,70703	0,015633
AT_pointsALast3g	1,173464	0,816134	1	0	3	3	0,383789	-0,60376	0,015574
HT_pointsLast3g	1,333697	0,814536	1,333333	0	3	3	0,203288	-0,707	0,015544
AT_pointsLast3g	1,408109	0,828402	1,333333	0	3	3	0,160417	-0,76625	0,015809
HT_poinstLast5g	1,349017	0,679178	1,4	0	3	3	0,219253	-0,43822	0,012961
AT_pointsLast5g	1,39461	0,676367	1,4	0	3	3	0,147655	-0,57971	0,012907
h2h_diff	-0,08563	1,661626	0	-3	3	6	0,000428	-0,55827	0,031709
HT_ShotsLast5g	12,40517	2,948088	12	5,2	21,6	16,4	0,465505	0,01389	0,056259
AT_ShotsLast5g	12,64406	3,012967	12,4	4,4	22	17,6	0,4162	-0,10506	0,057497
HT_ShotsOTLast5g	4,201238	1,313219	4	0,8	8,2	7,4	0,427227	-0,15242	0,02506
AT_ShotsOTLast5g	4,292353	1,338289	4,2	1	8,2	7,2	0,469646	-0,13572	0,025539
HT_Goals_Last5g	1,314567	0,624525	1,2	0	3,2	3,2	0,523997	-0,08747	0,011918
AT_Goals_Last5g	1,343554	0,622613	1,2	0	3,2	3,2	0,470549	-0,0843	0,011881
HT_GoalsConLast5	1,363001	0,583489	1,4	0	3	3	0,300348	-0,30728	0,011135
AT_GoalsConLast5	1,33496	0,580918	1,2	0	3	3	0,35626	-0,19404	0,011086
HT_AvgPLast2S	1,39152	0,409665	1,223684	0,947368	2,605263	1,657895	0,851578	-0,35494	0,007818
AT_AvgPLast2S	1,391498	0,41046	1,223684	0,947368	2,605263	1,657895	0,856032	-0,34569	0,007833
HT_AvgPLastS	1,403838	0,423505	1,236842	0,894737	2,631579	1,736842	0,902132	-0,12059	0,008082
AT_AvgPLastS	1,403879	0,42373	1,236842	0,894737	2,631579	1,736842	0,906046	-0,11338	0,008086
HT_AvgGoalsHLastS	0,377209	0,304395	0,351398	-0,30538	1,198696	1,504077	0,433989	-0,2263	0,005809
AT_AvgGoalsALastS	0,137268	0,33952	0,100083	-0,64185	0,926762	1,568616	0,148344	-0,63308	0,006479
HomeTeam_Value	0,319496	0,309496	0,177001	0	1	1	0,918295	-0,53896	0,005906
AwayTeam_Value	0,319108	0,309802	0,177001	0	1	1	0,929302	-0,51777	0,005912
HT_AvgPointsCurrentS	1,357755	0,509778	1,285714	0	2,923077	2,923077	0,387062	-0,25362	0,009728
AT_AvgPointsCurrentS	1,372675	0,508543	1,291667	0	2,925926	2,925926	0,403073	-0,22565	0,009705

Źródło: Opracowanie własne.

Niektóre zmienne zostaną opisane wspólnie z racji, że ich wartości są bardzo zbliżone, a ich charakterystyka odnosi się do tych samych statystyk aczkolwiek z podziałem na drużynę gospodarzy i drużynę przyjezdną. Do opisu zostaną użyte statystyki pierwszej z wymienionych zmiennych.

HT_pointsHLast3g – średnio drużyny w 3 ostatnich meczach domowych zdobywały 1,57pkt, przy odchyleniu od średniej wynoszącym 0,819208. Połowa danych znajduje się poniżej wartości 1,333. Skośność bliska 0 sugeruje równy rozkład zmiennych wokół średniej.

AT_pointsALast3g – drużyny w 3 ostatnich meczach wyjazdowych średnio zdobywają 1,173464 pkt. Odchylenie danych od średniej wynosi +/- 0,816134. Dodatnia skośność sugeruje, że więcej wyników znajduję się poniżej wartości średniej. Wartość najmniejsza wynosząca 0 oraz największa równa 3 prawdopodobnie odnosi się do początkowych kolejek sezonu.

HT_pointsLast3g oraz AT_pointsLast3g – drużyny zdobywały średnio 1,333697 punktu w trzech ostatnich meczach. Mediana zbliżona jest do tej średniej wartości, a skośność jest prawie zerowa. Wskazuje to na to, że dane mają symetryczny rozkład, co oznacza, że większość drużyn osiągała wyniki zbliżone do średniej, a rozkład punktów jest równomierny i nie ma wyraźnych skupisk ani odchyleń w danych.

HT_poinstLast5g i AT_pointsLast5g – średnia zdobytych punktów przez drużyny w 5 ostatnich meczach wynosi 1,349017 i jest zbliżona do wartości średniej zdobytych punktów w 3 ostatnich meczach. Połowa danych znajduje się powyżej wartości 1,4, dane są symetrycznie rozłożone wokół średniej.

h2h_diff – średnia jest zbliżona do mediany, natomiast odchylenie standardowe równe 1,661626 mówi o dużym rozproszeniu danych.

HT_ShotsLast5g i AT_ShotsLast5g – przeciętnie drużyny oddawały ponad 12 strzałów na bramkę rywala w okresie 5 meczów. Najmniejsza wartość wyniosła 5,2, a największa 21,6. Więcej wyników znajdowało się poniżej wartości średniej.

HT_ShotsOTLast5g oraz AT_ShotsOTLast5g – najmniejsza wartość wyniosła 0,8, świadczy to o tym, że istnieje drużyna która w okresie 5 meczów oddawała mniej niż jeden celny strzał na mecz. Maksymalna wartość wyniosła 8,2, a średnio drużyny strzelały celnie 4,201238 razy na mecz.

HT_AvgGoalsHLastS – Wartość mediany jest zbliżona do wartości średniej. Występuje małe odchylenie wartości od średniej.

AT_AvgGoalsALastS – Niższe wartości średniej oraz mediany niż w przypadku zmiennej HT_AvgGoalsHLastS sugerują, że drużyny zdobywały średnio mniej goli na w meczach wyjazdowych niż w przypadku spotkań rozgrywanych na własnym boisku.

HT_Goals_Last5g i AT_Goals_Last5g – Średnio drużyny zdobywały trochę ponad jeden gol na mecz. Mediana wynosiła 1,2, co sugeruje, że większość drużyn osiągała wyniki zbliżone do tej wartości. Skośność ma wartość 0,523997, więc więcej wyników znajdowało się poniżej średniej.

HT_GoalsConLast5 i AT_GoalsConLast5 – drużyny przeciętnie traciły 1,363001 goli na mecz, co naturalne jest wartością zbliżona do liczby goli strzelanych przez zespoły. Odchylenie standardowe wyniosło 0,624525.

HT_AvgPLast2S i AT_AvgPLast2S – W ciągu dwóch sezonów, drużyny osiągały średnio 1,39152 punktu, przy odchyleniu standardowym wynoszącym 0,409665. Najlepsza drużyna zdobyła średnio 2,605263 punktu. Skośność, która wynosiła 0,851578, wskazuje na to, że rozkład punktów ma tendencję do rozciągania się w stronę wyższych wartości.

HT_AvgPLastS i AT_AvgPLastS – Statystyki tych zmiennych bardzo zbliżone są do statystyk opisujących średnią punktów z dwóch ostatnich sezonów. To sugeruje, że obie zmienne opisują prawdopodobnie podobne zjawiska lub tendencje. Istnieje możliwość, że mają one podobny wpływ na analizowany aspekt, takim przypadku, aby uniknąć nadmiernego powielania informacji, może być rozważone usunięcie jednej z tych zmiennych.

HomeTeam_value i AwayTeam_value – zmienne zostały standaryzowane metodą min-maks. Skośność 0,918295 wskazuje na to, że rozkład danych ma silną asymetrię w kierunku wyższych wartości. Większość obserwacji ma wartości poniżej średniej, a niewielka liczba obserwacji osiąga bardzo wysokie wartości.

3.4 Analiza korelacji

W celu odpowiedniego doboru zmiennych została przeprowadzona analiza korelacji Pearsona. Mierzy ona siłę zależności liniowej między zmiennymi. Jej ograniczeniem jest podatność na obserwacje odstające. W literaturze nie ma jednego ścisłego podziału poziomu korelacji uznawanego za prawidłowy. Klasyfikacja zależy od postawionego problemu oraz charakterystyki danych. W tym przypadku zastosowano poniższy sposób podziału:

Tabela 15. Interpretacja korelacji.

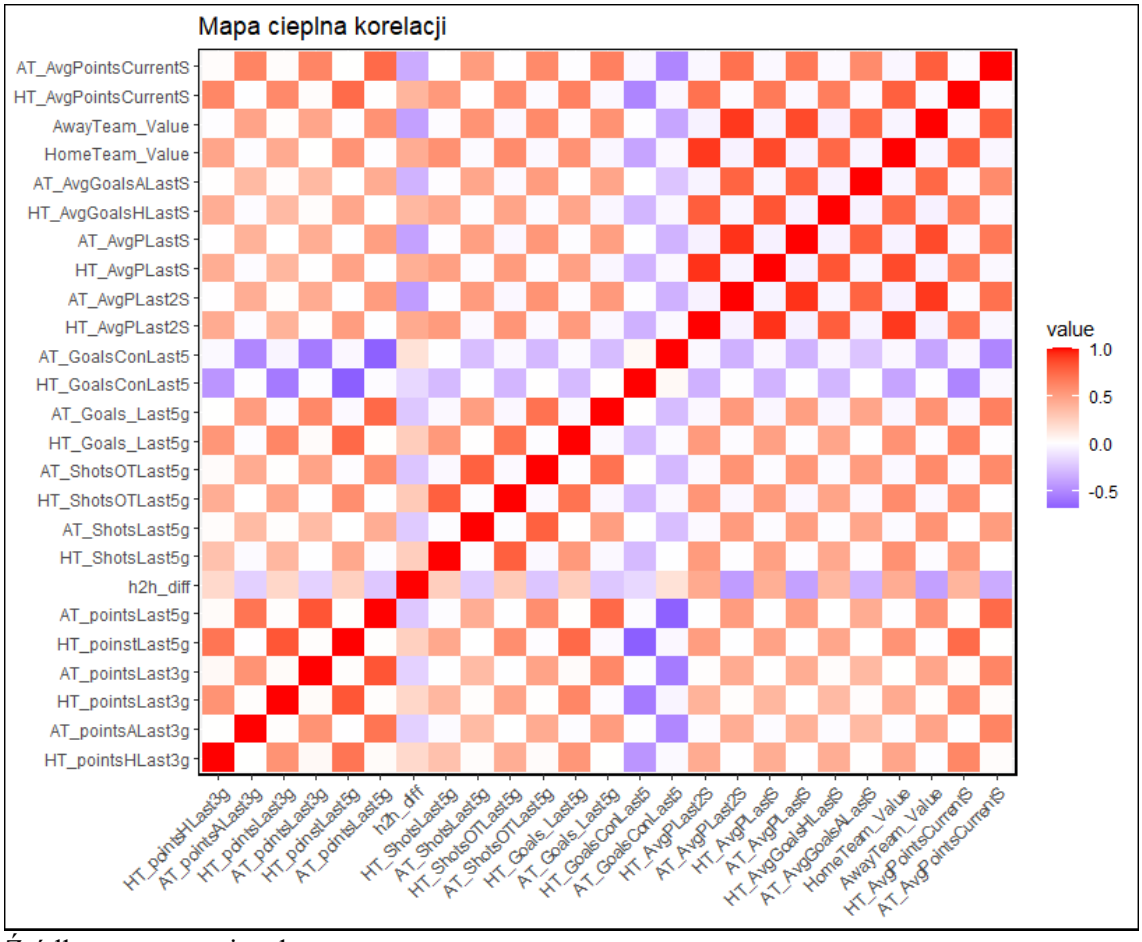
Rozmiar korelacji	Interpretacja
0,90 do 1,00 (-0,90 do -1,00)	Bardzo silna pozytywna (negatywna) korelacja
0,70 do 0,90 (-0,70 do -0,90)	Silna pozytywna (negatywna) korelacja
0,50 do 0,70 (-0,50 do -0,70)	Umiarkowana pozytywna (negatywna) korelacja
0,30 do 0,50 (-0,30 do -0,50)	Niska pozytywna (negatywna) korelacja
0,00 do 0,30 (0,00 do -0,30)	Mała lub żadna korelacja

Źródło: D. E. Hinkle, W. Wiersma, S. G. Jurs, *Applied Statistics for the Behavioral Sciences,* Houghton Mifflin, Tom 663, s109.[45]

Wyniki zostały przedstawione przy użyciu mapy cieplnej, w której wyższe wartości korelacji są reprezentowane przez kolory o cieplejszych odcieniach. Silna pozytywna korelacja ma intensywny czerwony kolor, natomiast silna negatywna korelacja jest koloru fioletowego. W przypadkach, kiedy kolor traci barwę występuje słaba korelacja lub jej brak.

[45]W. Wiersma, S. G. Jurs, *Applied Statistics for the Behavioral Sciences,* Houghton Mifflin, tom 663, Londyn 2003r., s109;

Ilustracja 7. Mapa cieplna korelacji.



Źródło: opracowanie własne.

Na mapie dobrze widoczne są skupienia zmiennych wysoko skorelowanych. Jeden z takich obszarów widoczny jest w lewym dolnym rogu. Zmienne w tym fragmencie mapy dotyczą formy drużyny w ostatnich meczach, zdobytych goli oraz punktów. Większa liczba zdobytych goli zazwyczaj prowadzi do wyższej średniej zdobytych punktów. Z tej racji można oczekiwać wysokiej korelacji między tymi zmiennymi. Kolejnym obszarem, w którym obserwujemy silną pozytywną korelację między kilkoma zmiennymi, jest prawy górny róg. Znajdują się tam zmienne opisujące siłę drużyny, takie jak wartość drużyny, liczba zdobytych punktów w poprzednich sezonach oraz średnia liczba punktów zdobytych w obecnym sezonie. Widzimy silną korelację między wartością drużyny oraz liczbą zdobytych punktów w poprzednim sezonie. Jest to spójne z analizą przeprowadzoną przy okazji opisu zmiennej wartości składu. Możemy również dostrzec, że zmienne wykazują znaczącą korelację w kategoriach związanych z drużyną

grającą u siebie oraz drużyną gości. Poniżej zostały przedstawione wszystkie pary których korelacja wynosi więcej niż 0,7.

Tabela 16. Pary zmiennych z korelacją wynoszącą powyżej 0,7.

Zmienna 1	Zmienna 2	Korelacja
HT_poinstLast5g	HT_pointsLast3g	0,819793
AT_pointsLast5g	AT_pointsLast3g	0,823328
HT_Goals_Last5g	HT_poinstLast5g	0,742126
HT_AvgPointsCurrentS	HT_poinstLast5g	0,735597
AT_Goals_Last5g	AT_pointsLast5g	0,744286
AT_AvgPointsCurrentS	AT_pointsLast5g	0,738733
HT_ShotsOTLast5g	HT_ShotsLast5g	0,779908
AT_ShotsOTLast5g	AT_ShotsLast5g	0,778182
HT_Goals_Last5g	HT_ShotsOTLast5g	0,703178
AT_Goals_Last5g	AT_ShotsOTLast5g	0,705234
HT_AvgPLastS	HT_AvgPLast2S	0,931322
HT_AvgGoalsHLastS	HT_AvgPLast2S	0,787951
HomeTeam_Value	HT_AvgPLast2S	0,915052
HT_AvgPointsCurrentS	HT_AvgPLast2S	0,707621
AT_AvgPLastS	AT_AvgPLast2S	0,929697
AT_AvgGoalsALastS	AT_AvgPLast2S	0,766475
AwayTeam_Value	AT_AvgPLast2S	0,915225
AT_AvgPointsCurrentS	AT_AvgPLast2S	0,710383
HT_AvgGoalsHLastS	HT_AvgPLastS	0,824391
HomeTeam_Value	HT_AvgPLastS	0,862683
AT_AvgGoalsALastS	AT_AvgPLastS	0,787689
AwayTeam_Value	AT_AvgPLastS	0,861707
HomeTeam_Value	HT_AvgGoalsHLastS	0,746098
AwayTeam_Value	AT_AvgGoalsALastS	0,749849
HT_AvgPointsCurrentS	HomeTeam_Value	0,782218
AT_AvgPointsCurrentS	AwayTeam_Value	0,787619

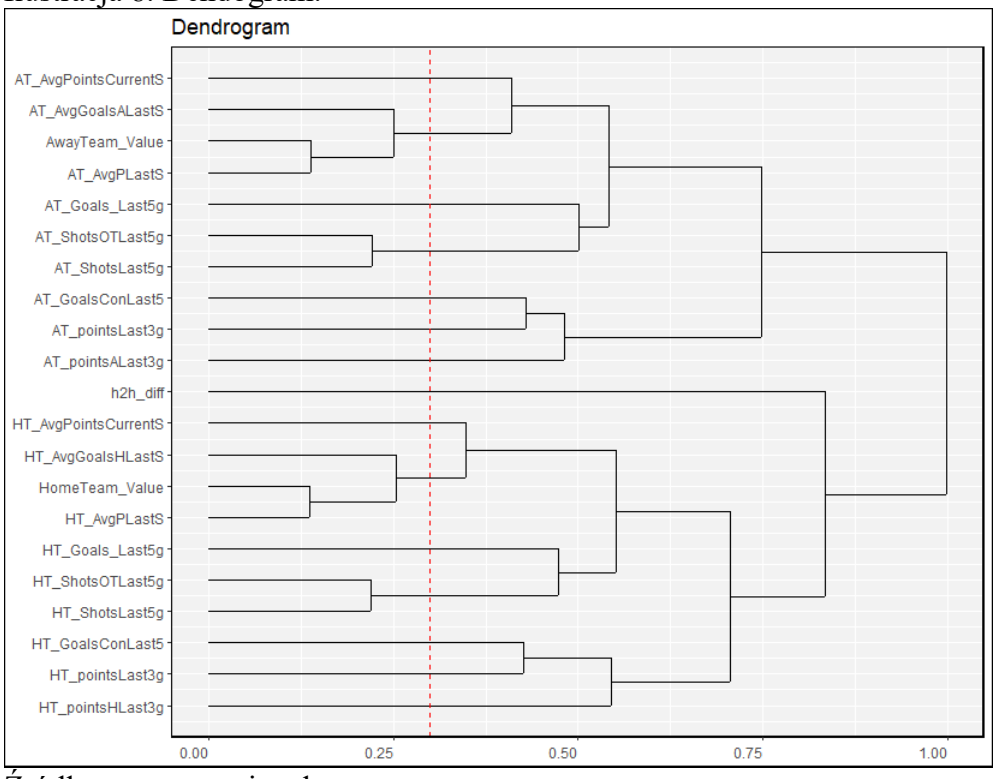
Źródło: Opracowanie własne.

W związku z dużymi korelacjami z innymi zmiennymi zostały usunięte zmienne HT_AvgPLast2S, AT_AvgPLast2S, AT_pointsLast5g, HT_pointsLast5g. Pozostałe zmienne z silnymi korelacjami będą brane pod uwagę w dalszej analizie, możliwe, że będzie trzeba je usunąć na późniejszym etapie.

3.5. Analiza skupień

Aby dobrać odpowiedni zestaw zmiennych do modelu została wykonana hierarchiczna analiza skupień. Celem tej analizy jest dobranie zmiennych w grupy charakteryzujące się silną korelacją między zmiennymi oraz niewielką korelacją między grupami. Do wykonania analizy skupień został wybrany algorytm DIANA[46]. Jest to algorytm oparty na analizie dzielącej. Rozpoczyna się od utworzeniu jednego dużego klastra. W następnych krokach wybierany jest najbardziej niejednorodny klaster i dzielony jest on rekurencyjnie na dwa. Działanie algorytmu kończy się w momencie, w którym wszystkie punkty danych należą do odpowiednich klastrów lub kiedy zostanie spełnione odpowiednie kryterium stopu. Wyniki działania algorytmu zostały przedstawione na dendrogramie. Próg odcięcia odpowiada korelacji równej 0.7.

Ilustracja 8. Dendogram.



Źródło: opracowanie własne.

[46] https://stat.ethz.ch/R-manual/R-devel/library/cluster/html/diana.html - data odczytu 02.08.2023r.;

W wyniku przeprowadzonej analizy skupień dane zostały podzielone na 15 klastrów:

Tabela 17. Podział zmiennych na klastry.

Zmienna	Klaster
HT_pointsHLast3g	1
AT_pointsALast3g	2
HT_pointsLast3g	3
AT_pointsLast3g	4
h2h_diff	5
HT_ShotsLast5g	6
HT_ShotsOTLast5g	6
AT_ShotsLast5g	7
AT_ShotsOTLast5g	7
HT_Goals_Last5g	8
AT_Goals_Last5g	9
HT_GoalsConLast5	10
AT_GoalsConLast5	11
HT_AvgPLastS	12
HT_AvgGoalsHLastS	12
HomeTeam_Value	12
AT_AvgPLastS	13
AT_AvgGoalsALastS	13
AwayTeam_Value	13
HT_AvgPointsCurrentS	14
AT_AvgPointsCurrentS	15

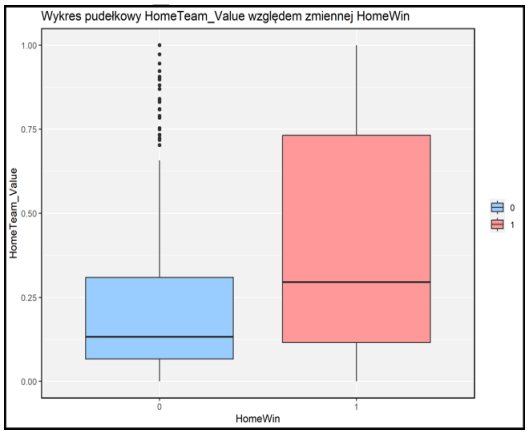
Źródło: Opracowanie własne.

3.6 Wybór wyjściowego zbioru danych

W celu dokładniejszego zrozumienia względnej zdolności predykcyjnej poszczególnych zmiennych, przeprowadzono analizę za pomocą wykresów pudełkowych. Wykresy te przedstawiają charakterystyki rozkładu wartości

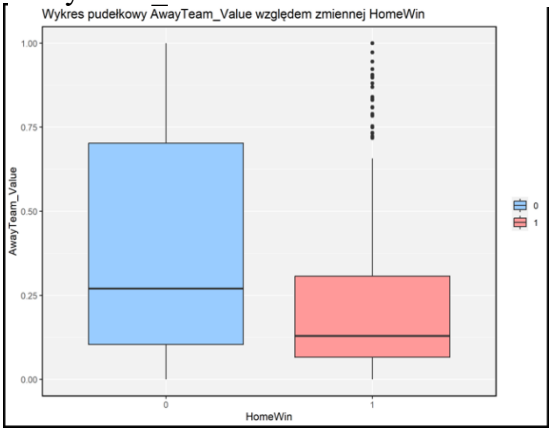
poszczególnych zmiennych w kontekście różnych wartości zmiennej objaśnianej, tj. "HomeWin". Podczas ich analizy kierowano się uwzględnieniem kilku kluczowych aspektów. Po pierwsze, zbadano, czy zmienne wykazywały zbliżone wartości mediany dla obu kategorii zmiennej objaśnianej (czyli "HomeWin"). To pomagało określić, czy istnieje jasna różnica w tendencji zmiennych między dwiema grupami. Ponadto, szczególną uwagę zwrócono na zakres międzykwartylowy, który wskazuje na dystrybucję danych między pierwszym a trzecim kwartylem. Analiza tego zakresu pozwoliła ocenić, w jakich przedziałach wartości występuje większa zmienność między wartościami zmiennych dla różnych wartości "HomeWin". Działanie to pomogło w identyfikacji zmiennych, które potencjalnie mogą mieć istotny wpływ na zdolność predykcyjną w odniesieniu do zmiennej objaśnianej "HomeWin". Zmienne które zostały uznane za najlepsze w kontekście predykcji to: HomeTeam_Value, AwayTeam_Value, HT_AvgGoalsHLastS, HT_AvgPointsCurrentS, AT_pointsALast3g poniżej znajdują się ich wykresy:

Ilustracja 10. Wykres pudełkowy zmiennej HomeTeam_Value.



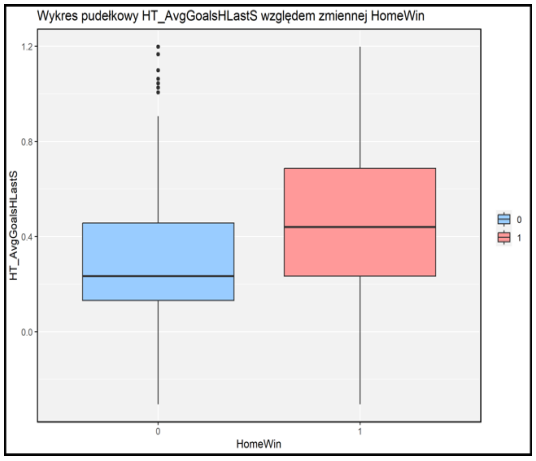
Źródło: opracowanie własne.

Ilustracja 9. Wykres pudełkowy zmiennej AwayTeam_value.



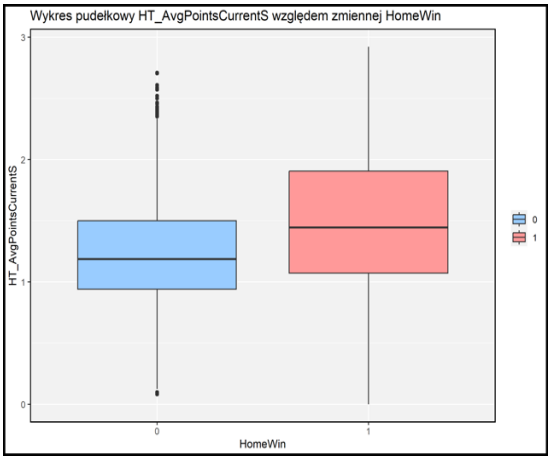
Źródło: opracowanie własne.

Ilustracja 12. Wykres pudełkowy zmiennej HT_AvgGoalsHlastS.



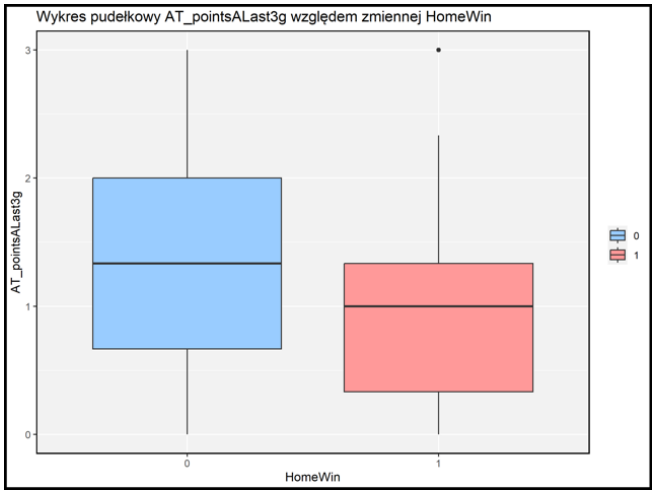
Źródło: opracowanie własne.

Ilustracja 11. Wykres pudełkowy zmiennej HT_AvgPointsCurrentS.



Źródło: opracowanie własne.

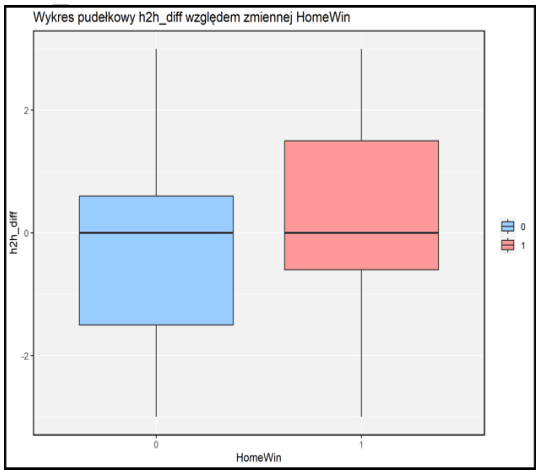
Ilustracja 13. Wykres pudełkowy zmiennej AT_pointsALast3g.



Źródło: opracowanie własne.

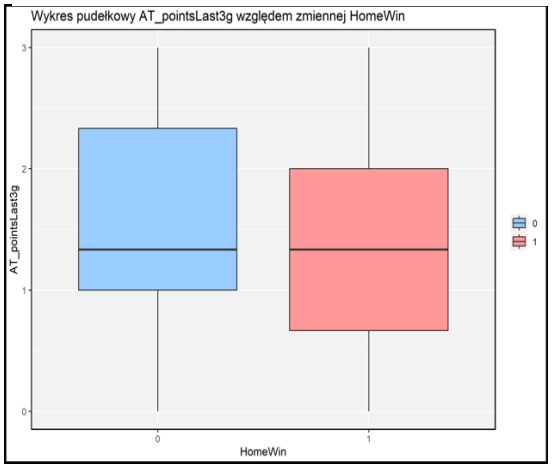
W wyniku przeprowadzonej analizy, dostrzeżono zmienne, które mogą charakteryzować się ograniczoną zdolnością predykcyjną. Są to zmienne: HT_pointsLast3g, AT_pointsLast3g oraz h2h_diff.

Ilustracja 15. Wykres pudełkowy zmiennej h2h_diff.



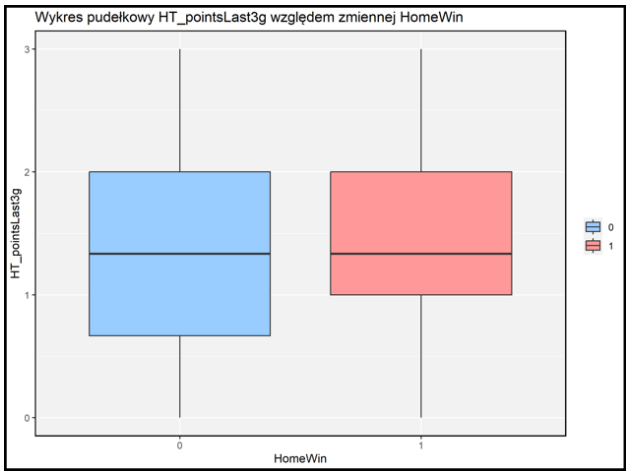
Źródło: opracowanie własne.

Ilustracja 14. Wykres pudełkowy zmiennej AT_pointsLast3g.



Źródło: opracowanie własne.

Ilustracja 16. Wykres pudełkowy zmiennej HT_pointslast3g.



Źródło: opracowanie własne.

Zmienne HT_pointsLast3g oraz AT_pointsLast3g zostaną wykluczone z dalszej analizy i nie będą uwzględniane w procesie modelowania. Natomiast, ze względu na obecność znaczącej liczby brakujących danych w zmiennej h2h_diff, zostanie ona poddana dalszym obserwacjom. W celu uzupełnienia tych brakujących danych, zostaną wdrożone różnorodne metody, co pozwoli na dokładniejsze zrozumienie charakterystyki tej zmiennej oraz jej potencjalnego wpływu na analizowany proces.

W wyniku powyższej analizy został wybrany podstawowy zestaw danych wejściowych: HT_pointsHLast3g,

- AT_pointsALast3g,
- h2h_diff,
- HT_ShotsOTLast5g,
- AT_ShotsOTLast5g,
- HT_Goals_Last5g,
- AT_Goals_Last5g,
- HT_GoalsConLast5,
- AT_GoalsConLast5,
- HomeTeam_Value,
- AwayTeam_Value,
- HT_AvgPointsCurrentS,
- AT_AvgPointsCurrentS,

3.7. Prezentacja kodu źródłowego

W celu utworzenia kodu został napisany kod w języku programowania R, w środowisku RStudio.

3.7.1 Drzewa decyzyjne

Do utworzenia modelu drzewa klasyfikacyjnego oprócz wbudowanych bibliotek środowiska Rstudio zostały użyte dodatkowe biblioteki:

Rpart() –tworzenie, trenowanie oraz przycinanie modelu,

Mlr()- tworzenie modelu i strojenie parametrów,

Caret() – ocena modelu, generowanie macierzy pomyłek,

Rpart.plot() – generowanie wykresów drzew,

pROC()- do obliczenia krzywej ROC i wygenerowania wykresu,

W pierwszej kolejności dane zostały wczytane i podzielone na zbiór treningowy oraz zbiór testowy w proporcjach 70% (1929 obserwacji) do 30%(817 obserwacji). Dokonano tego z użyciem funkcji sample() która w sposób losowy przypisuje indeksy do danych

w odpowiednich proporcjach. Aby wyniki były powtarzalne zostało ustawione ziarno losowości.

Ilustracja 17. Kod źródłowy dla podziału na zbiór testowy i treningowy.

```
set.seed(123)
ind <- sample(2, nrow(dane_tree1), replace = TRUE, prob = c(0.7, 0.3))
train_set <- dane_tree1[ind==1,]
test_set <- dane_tree1[ind==2,]
```

Źródło: opracowanie własne.

Następnie utworzono podstawowe drzewo decyzyjne do którego został wykorzystany wyjściowy zbiór danych oraz domyślne wartości parametrów.

Ilustracja 18. Kod źródłowy dla utworzenia podstawowego drzewa.

```
tree1 = rpart(HomeWin ~ .,
              data=train_set,
              method = 'class')
```

Źródło: opracowanie własne.

Tabela 18. Domyślne wartości parametrów drzewa decyzyjnego.

Parametr	Wartość
Minsplit	20
Minbucket	-
Cp	0,01
Maxcompete	4
Maxsurrogate	5
Usesurrogate	2
Surrogatestyle	0
Maxdepth	30
Xval	10
Kryterium podziału	Gini

Źródło: Opracowanie własne.

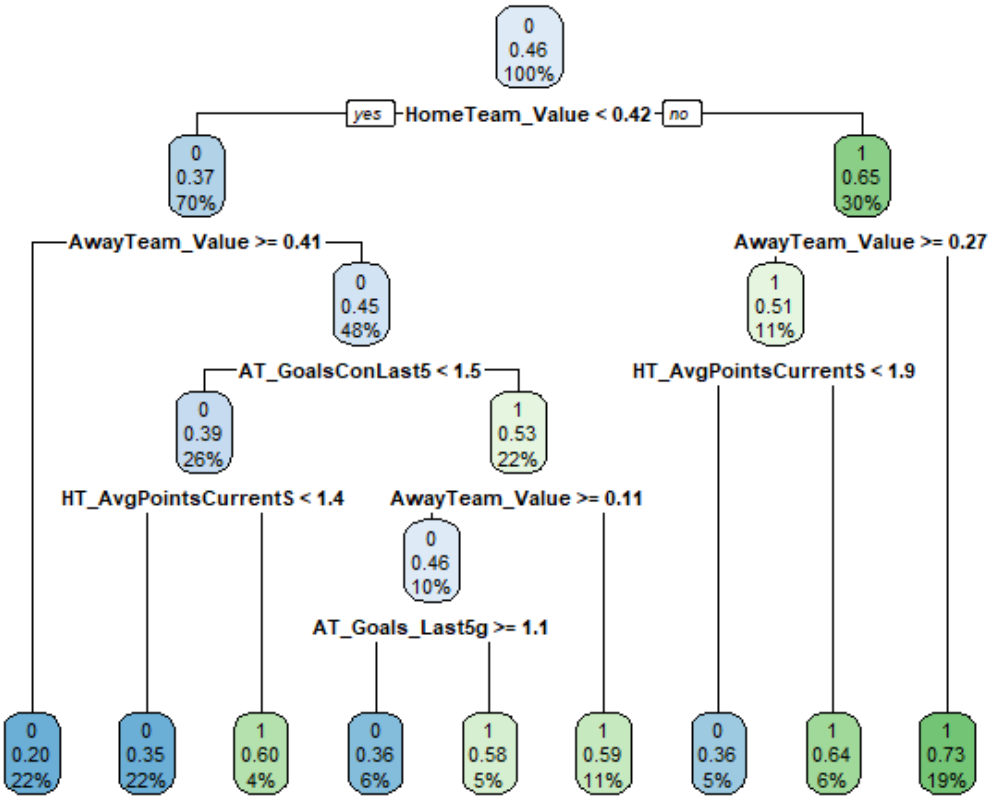
Wygenerowano również wykres przedstawiający schemat powstałego drzewa oraz sprawdzono jego zdolność do predykcji na zbiorze testowym.

Ilustracja 19. Kod źródłowy dla predykcji dla modelu 1.

```
rpart.plot(tree1)

predicted <- predict(tree1, train_set, type = 'class')
confusionMatrix(predicted,train_set$HomeWin)

predicted1 <- predict(tree1, test_set, type = 'class')
confusionMatrix(predicted1,test_set$HomeWin)
```

Źródło: opracowanie własne.

Ilustracja 20. Graficzne przedstawienie modelu 1.



Źródło: opracowanie własne.

Aby sprawdzić czy już na tym etapie może występować zjawisko przeuczenia, porównano otrzymana dokładność z dokładnością predykcji na zbiorze treningowym która wyniosła 68.48%. Różnica tylko 4 punktów procentowych nie sugeruje nadmiernego przeuczenia modelu. Następnym etapem było dokładne dopasowanie

odpowiednich parametrów modelu. Zwrócono szczególną uwagę na następujące parametry:

Maxdepth – Wartość „maxdepth” reguluje maksymalną głębokość drzewa, innymi słowy określa ile drzewo może osiągnąć poziomów od korzenia zanim zostanie zatrzymany jego wzrost. Wybranie dużej wartości maxdepth prowadzi do wygenerowania rozrośniętego drzewa, które dobrze oddaje zależności w danych treningowych jednak może to prowadzić do osłabionej zdolności drzewa w generalizowaniu nowych danych. W przypadku wybrania za małej wartości istnieje ryzyko niedouczenia modelu.

Minsplit – Parametr minsplit definiuje minimalną liczbę obserwacji, która musi znajdować się w węźle, aby mógł zostać dokonany podział na kolejne węzły. Gdy liczba obserwacji w danym węźle nie przekracza wartości minsplit, znaczy to, że podział nie zostanie wykonany, a ten węzeł stanie się liściem.

Cp – To współczynnik złożoności drzewa, odpowiada on za kontrolowanie rozmiaru drzewa i przycinanie zbędnych gałęzi. Im wyższa wartość cp, tym bardziej restrykcyjne są kryteria przycinania. W początkowej fazie budowy drzewa, dąży się do utworzenia modelu o większej złożoności, które uwzględni jak najwięcej zależności występujących w danych treningowych, dlatego początkowo wartość cp została ustawiona na 0. Na etapie przycinania drzewa zostanie wybrana optymalna wartość tego parametru.

Wszystkie pozostałe parametry zostały zachowane w ich domyślnych wartościach. W celu dostrojenia parametrów modelu został utworzony obiekt zadania klasyfikacyjnego, określona została metoda walidacji krzyżowej, miara którą będzie się kierował model w celu weryfikacji parametrów oraz siatka parametrów które zostaną sprawdzone. W wyniku działania siatki parametrów zostały sprawdzone wszystkie kombinacje parametrów maxdepth (1:10), cp =0, minsplit(1:30), przy 10 krotnej walidacji krzyżowej.

Ilustracja 21. Kod źródłowy dla procesu strojenia hiperparametrów.

```
getParamSet("classif.rpart")
tree2 <- makeClassifTask(
  data=train_set,
  target="HomeWin"
)

control_grid = makeTuneControlGrid()
resample = makeResampleDesc("CV",iter = 5,predict = "both")
measure = acc

param_grid <- makeParamSet(
  makeDiscreteParam("maxdepth", values=1:10),
  makeDiscreteParam("cp", values = 0),
  makeDiscreteParam("minsplit", values=1:30),
  makeDiscreteParam('xval',value =10)
)
```

Źródło: opracowanie własne.

W wyniku działania powyższego kodu został wyznaczony najlepszy zestaw parametrów, dla którego wykonano predykcje na zbiorze testowym.

Ilustracja 22. Kod źródłowy dla predykcji i wyboru parametrów.

```
best_params = setHyperPars(
  makeLearner("classif.rpart", predict.type = "prob"),
  par.vals = dt_tuneparam_multi$x
)

best_model_multi <- mlr::train(best_params, tree2)
best_tree_model <- best_model_multi$learner.model
rpart.plot(best_tree_model)
predicted2 <- predict(best_tree_model, newdata = test_set, type = "class")
confusionMatrix(predicted2, test_set$HomeWin)
```

Źródło: opracowanie własne.

Na tym etapie również dodawane były poszczególne zmienne, które początkowo nie były wykorzystane. W przypadku, gdy dodanie danej zmiennej poprawiało dokładność, była ona uwzględniana w modelu. Natomiast jeśli dodanie zmiennej nie miało wpływu na jakość lub osłabiało jego zdolność prognostyczną, taka zmienna nie była włączana do modelu. Kombinacją zmiennych która cechowała się największą dokładnością, był podstawowy zestaw zmiennych powiększony o zmienną HT_AvgGoalsHLastS. Po sprawdzeniu wag dla poszczególnych zmiennych w modelu, zdecydowano się również na usunięcie zmiennych HT_pointsHlast3g, AT_pointsAlast3g oraz

HT_ShotsOTLast5g ponieważ nie wnosiły one nic do modelu. W wyniku strojenia parametrów zostały wyznaczone następujące wartości parametrów:

Maxdepth = 5,

Minsplit = 27,

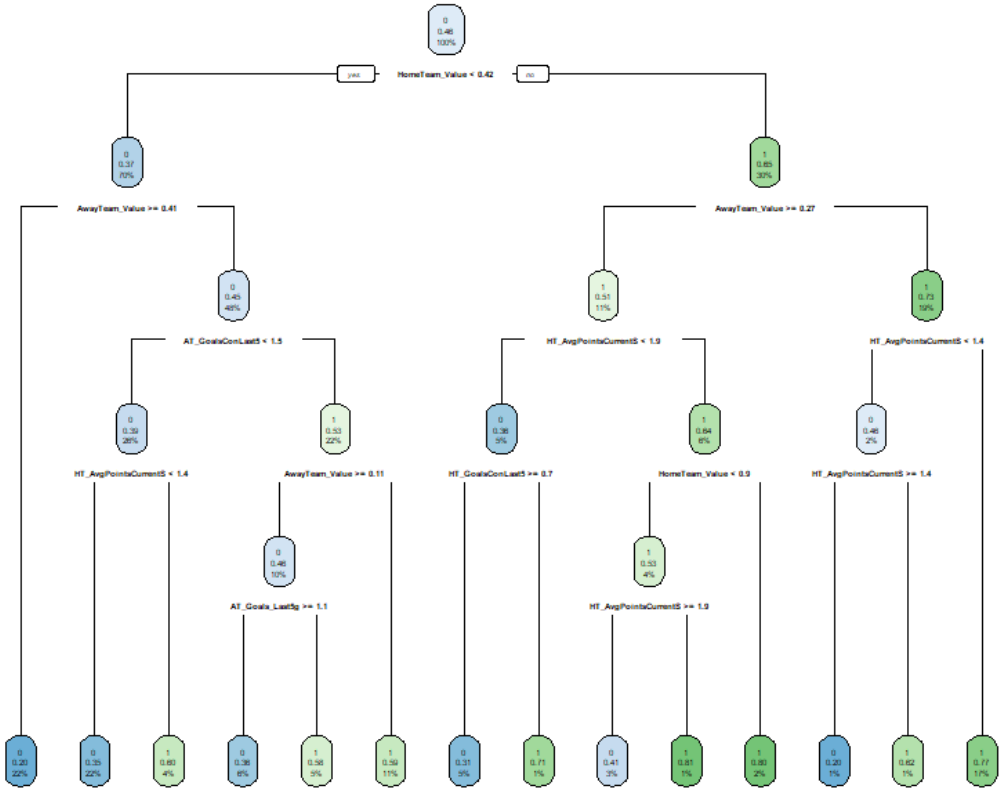
Zostało również sprawdzone, jakie zmienne miały największy wpływ na proces podejmowania decyzji. Zmienna, która okazała się najbardziej znacząca to wartość drużyny przyjezdnej. Kolejne odpowiadały za formę drużyn w obecnym sezonie oraz wartość drużyny gospodarzy. Oprócz wcześniej wymienionych zmiennych najmniejszą wagę w modelu osiągnęły zmienne HT_GoalsConLast5 oraz HT_Goals_Last5g.

Tabela 19. Wagi parametrów w modelu 3.

Zmienna	Waga
AwayTeam_Value	119,697
AT_AvgPointsCurrentS	89,4049
HT_AvgPointsCurrentS	88,79269
HomeTeam_Value	77,02568
HT_AvgGoalsHLastS	74,88642
AT_Goals_Last5g	37,76713
AT_GoalsConLast5	36,45533
AT_ShotsOTLast5g	30,03777
h2h_diff	13,07669
HT_GoalsConLast5	7,285428
HT_Goals_Last5g	3,288901

Źródło: Opracowanie własne.

Ilustracja 23. Graficzne przedstawienie modelu 2.



Źródło: opracowanie własne.

Następnie został zastosowany proces przycięcia drzewa z wykorzystaniem funkcji prune() oraz optymalnego poziomu cp (0,005113636) wyznaczonego na podstawie minimalizowania błędu walidacji krzyżowej.

Ilustracja 24. Kod źródłowy dla procesu przycinania drzewa.

```
best_tree_model$cptable
best_cp <- best_tree_model$cptable[which.min(best_tree_model$cptable[,"xerror"]),"CP"]
plotcp(best_tree_model)
pruned_tree <- prune(best_tree_model, cp = best_cp)
print(pruned_tree)
rpart.plot(pruned_tree)
predicted3 <- predict(pruned_tree, newdata = test_set, type = "class")
confusionMatrix(predicted3, test_set$HomeWin)
```

Źródło: opracowanie własne.

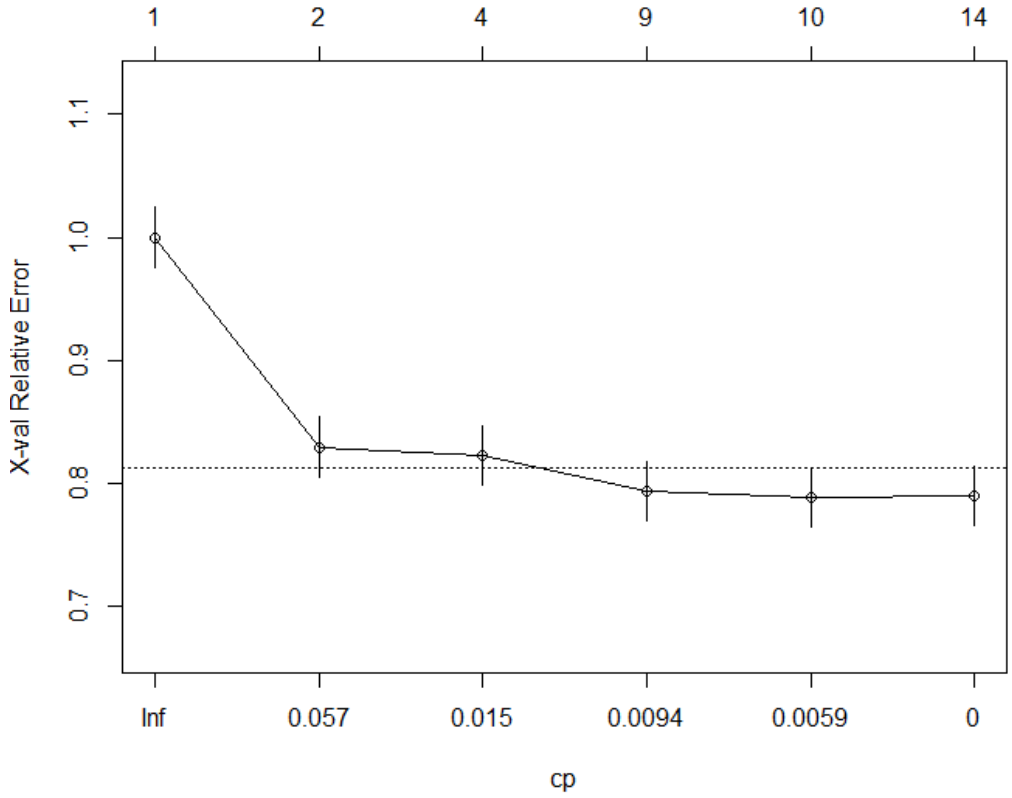
Tabela 20. Tabela wartości cp dla modelu 2.

	cp	nsplit	rel error	xerror	xstd
1	0,195455	0	1	1	0,024859
2	0,016477	1	0,804545	0,829545	0,024206
3	0,013068	3	0,771591	0,826136	0,024186
4	0,006818	8	0,690909	0,796591	0,024005
5	0,005114	9	0,684091	0,781818	0,023907
6	0	13	0,663636	0,782955	0,023915

Źródło: Opracowanie własne.

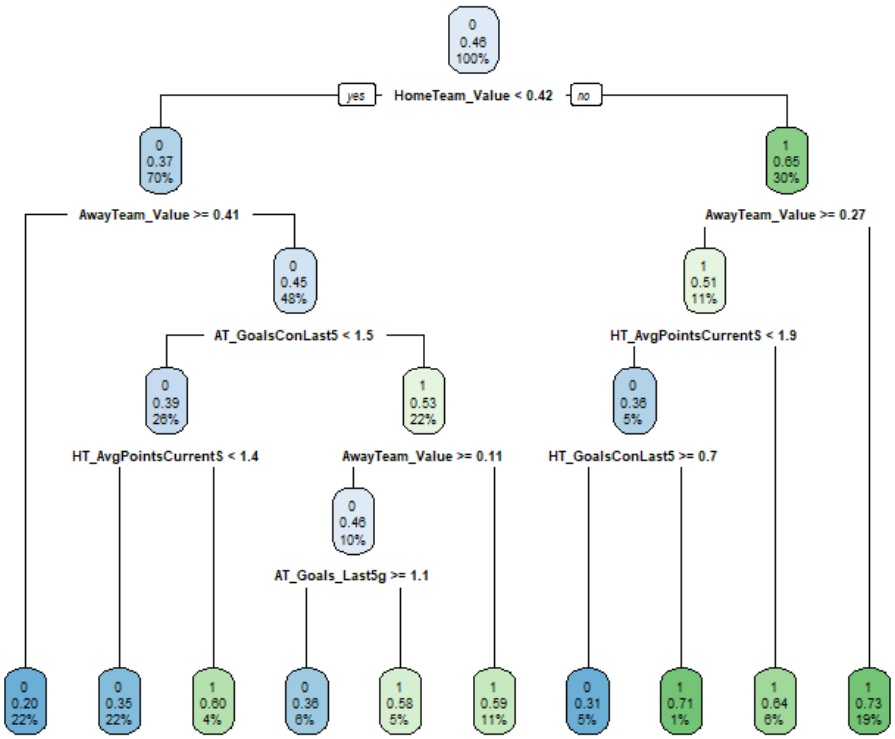
W celu lepszego zrozumienia zależności błędu klasyfikacji krzyżowej względem cp i rozmiaru drzewa został wygenerowany wykres.

Ilustracja 25. Wykres zależności pomiędzy cp, a rozmiarem drzewa i błędem walidacji krzyżowej.



Źródło: opracowanie własne.

Ilustracja 26. Graficzne przedstawienie modelu 3.



Źródło: opracowanie własne.

Proces budowy modelu kończy weryfikacja jakości przyciętego drzewa.

3.7.2 Lasy losowe

Kolejną zastosowaną metodą był algorytm lasów losowych. Przy tworzeniu modelu zostały wykorzystane następujące funkcje:

Ranger() – budowanie, trenowanie modelu oraz strojenie parametrów

Caret()- ocena jakości modelu, generowanie macierzy pomyłek

pROC()- generowanie krzywej ROC

ggplot2()- tworzenie wykresów

Podczas procesu trenowania modelu, dane zostały podzielone zgodnie z wcześniejszym podejściem, a następnie stworzono podstawowy model, który miał domyślne wartości parametrów.

Ilustracja 27. Kod źródłowy dla utworzenia oraz predykcji modelu.

```
model <- ranger(HomeWin ~ .,
                data = train_set)
p1 <- predict(model, train_set)
confusionMatrix(p1$predictions, train_set$HomeWin)
p2 <- predict(model, test_set)
confusionMatrix(p2$predictions, test_set$HomeWin)
```

Źródło: opracowanie własne.

Ten model osiągnął 100% dokładności na zbiorze treningowym oraz 65.85% dokładności na zbiorze testowym. Tak wysoka dokładność na zbiorze treningowym sugeruje, że model jest nadmiernie dopasowany do tych danych, dlatego należy dobrać odpowiednie wartości parametrów. Przy budowie lasów losowych istnieją 3 najważniejsze parametry na które należy zwrócić uwagę:

Num.trees – Określa liczbę drzew, które znajdują się w lesie. Z reguły większa liczba drzew oferuje lepsze wyniki, jednak wytrenowanie takiego modelu zużywa większą moc obliczeniową.

Mtry – Algorytm lasu losowego w każdym węźle losuje bez zwracania określoną przez mtry liczbę zmiennych. Na podstawie najlepszej z nich dokonywany jest podział. Domyślnie mtry przyjmuje wartość równą pierwiastkowi kwadratowemu z liczby zmiennych. Duża wartość mtry wprowadza mniejszą losowość do modelu, ponieważ częściej zdarza się, że w wylosowanych zmiennych znajdują się takie o dużej wartości predykcyjnej. Powoduje to jednak, że większość drzew w lesie jest do siebie podobna. Natomiast mała wartość może powodować bardzo słabą moc predykcyjną niektórych drzew

Max.depth – Wpływa na maksymalną głębokość każdego drzewa w lesie. Wartość domyślna ustawiona jest na 0 i oznacza maksymalny możliwy rozrost drzew, na jaki pozwala zbiór danych. Jednak model z niską wartością max.depth jest bardziej podatny na przeuczenie. Wytrenowanie takiego modelu zabiera również większą ilość czasu.

Jako pierwszy z parametrów została sprawdzona liczba drzew w lesie. Użyto do tego funkcji ranger oraz pętli sprawdzającej jak zachowuje się błąd prognozowania w zależności od różnych wartości num.trees. Sprawdzone zostały wartości liczby drzew z zakresu od 50 do 1000. Wartość mtry ustawiono jako 4, a kryterium podziału stanowił

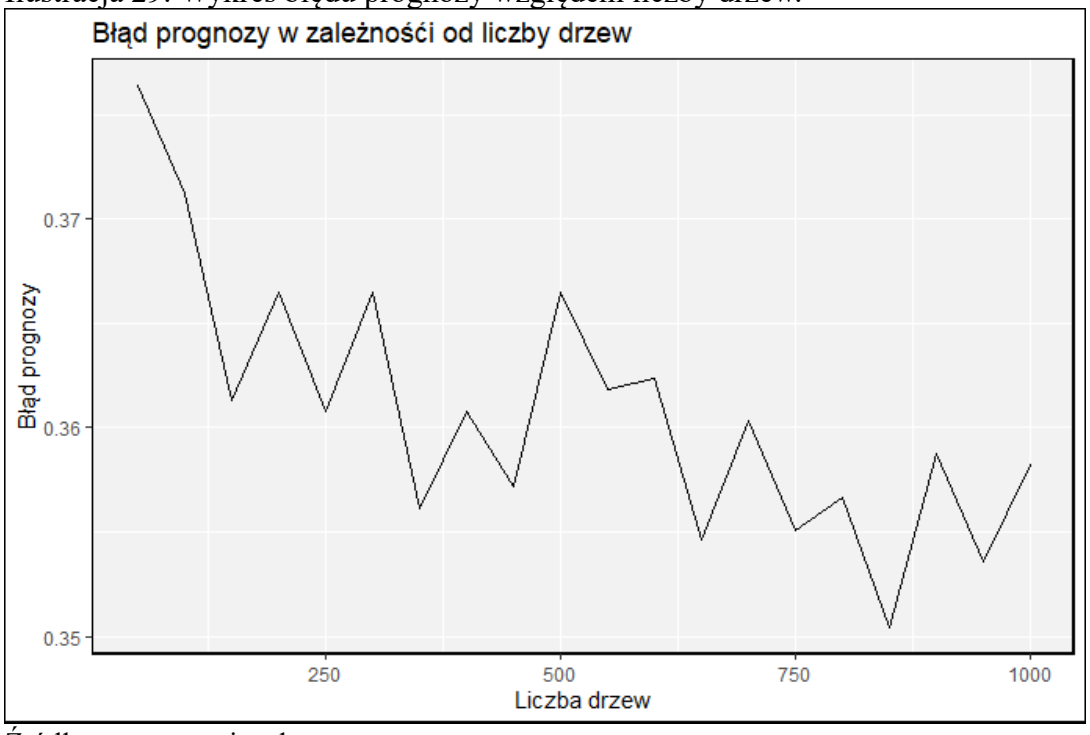
indeks Giniego. Następnie został wygenerowany wykres przedstawiający wyniki działania pętli.

Ilustracja 28. Kod źródłowy dla doboru liczby drzew.

```
num.trees <- seq(50, 1000, by = 50)
oob.error <- vector("numeric", length(num.trees))

for (i in seq_along(num.trees)) {
  model <- ranger(HomeWin ~ .,
                  data = train_set,
                  num.trees = num.trees[i],
                  mtry = 4,
                  importance = 'impurity')
  oob.error[i] <- model$prediction.error
}

df <- data.frame(NumTrees = num.trees, OOBE = oob.error)
ggplot(df, aes(x = NumTrees, y = OOBE)) +
  geom_line() +
  labs(title = "OOB Error vs Liczba drzew", x = "Liczba drzew", y = "OOB Error")+
  theme(panel.border = element_rect(color = 'black',fill = NA,size = 1),
        panel.background = element_rect(fill='gray95'),
        plot.background = element_rect(color = 'black', size = 1),
        axis.title.x = element_blank(),
        axis.title.y = element_blank())

min.error.index <- which.min(oob.error)
best.num.trees <- num.trees[min.error.index]
```

Źródło: opracowanie własne.

Ilustracja 29. Wykres błędu prognozy względem liczby drzew.



Źródło: opracowanie własne.

Kolejnymi parametrami które należało dopasować były mtry oraz max.depth. W tym celu została utworzona siatka zmiennych zawierające wszystkie kombinacje mtry z zakresu od 1 do 10 oraz max.depth z zakresu od 1 do 15. Sprawdzenie wartości parametrów przebiegało analogicznie jak w przypadku liczby drzew. Wartość num.trees została ustawiona na najlepszą wartość wyznaczoną w poprzednim kroku.

Ilustracja 30. Kod źródłowy dla utworzenia i zastosowania siatki parametrów.

```
tune_grid <- expand.grid(max.depth = seq(1, 15, by = 1),
                         mtry.size = seq(1, 10, by = 1))

for(i in 1:nrow(tune_grid)){
  max_depth_value <- tune_grid$max.depth[i]
  mtry_value <- tune_grid$mtry.size[i]

  model <- ranger(HomeWin ~ .,
                  data = train_set,
                  num.trees = best.num.trees,
                  mtry = mtry_value,
                  max.depth = max_depth_value,
                  importance = 'impurity')
  oob.error[i] <- model$prediction.error
}

min.error.index <- which.min(oob.error)
best_params <- tune_grid[min.error.index,]
```

Źródło: opracowanie własne.

Ostatnim etapem było zbudowanie lasu losowego za pomocą wybranych wcześniej parametrów, oraz ocena modelu.

Ilustracja 31. Kod źródłowy dla zastosowania modelu lasów losowych.

```
model <- ranger(HomeWin ~ .,
                data = train_set,
                num.trees = best.num.trees,
                mtry = best_params$mtry.size,
                max.depth = best_params$max.depth,
                importance = 'impurity')

print(model)
attributes(model)
p3 <- predict(model, train_set)
confusionMatrix(p3$predictions, train_set$HomeWin)
p4 <- predict(model, test_set)
confusionMatrix(p4$predictions, test_set$HomeWin)
```

Źródło: opracowanie własne.

Ponownie, rozpoczęto od sprawdzenia modelu na podstawowym zbiorze danych. Następnie dodawano kolejne zmienne i weryfikowano jakość prognozy. Model z największą dokładnością zawierał podstawowy zestaw zmiennych powiększony

o HT_AvgGoalsHLastS oraz AT_ShotsLast5g. Wartości dobrane w procesie strojenie zostały przedstawione w poniższej tabeli.

Tabela 21. Wartości parametrów dla modelu 4.

Parametr	Wartość
Num.trees	850
Mtry	4
Max.depth	7

Źródło: Opracowanie własne.

Tabela 22. Wagi zmiennych w modelu 4.

Zmienna	Waga
HT_AvgPointsCurrentS	45,73125
HomeTeam_Value	45,57244
AwayTeam_Value	45,02436
AT_AvgPointsCurrentS	37,0539
HT_AvgGoalsHLastS	32,03984
AT_ShotsLast5g	26,01121
HT_ShotsOTLast5g	22,16368
h2h_diff	20,6538
AT_GoalsConLast5	19,06619
AT_ShotsOTLast5g	19,03663
AT_Goals_Last5g	14,22887
HT_GoalsConLast5	13,3657
HT_Goals_Last5g	13,24854
HT_pointsHLast3g	9,746709
AT_pointsALast3g	9,224401

Źródło: Opracowanie własne.

Możemy zauważyć, że wagi zmiennych są do siebie bardziej zbliżone niż w przypadku algorytmu drzewa decyzyjnego. Największe znaczenie dla modelu miały trzy zmienne: HT_AvgPointsCurrentS, HomeTeam_Value, AwayTeam_Value.

3.7.3. Ekstremalne wzmacnianie gradientu

Ostatnim użytym algorytmem jest algorytm XGBoost (ang. Extreme Gradient boosting). Jest to zmodyfikowany algorytm wzmacniania gradientu, który został opracowany przez Tianqi Chena, i od tego czasu wygrywał liczne nagrody w konkursach organizowanych przez platformę Kaggle. W algorytm XGBoost w porównaniu do tradycyjnych metod wzmacniania gradientu wprowadzono miedzy innymi składnik regularyzacji, który odpowiada za kontrole złożoności modelu oraz redukcje wariancji stosując system kar nakładany na model za zbyt duża liczbę obserwacji w segmencie.

W celu utworzenia modelu zostały użyte następujące funkcje:

Xboost() – budowanie i trenowanie modelu,

Mlr() – strojenie hiperparametrów modelu,

Caret() – weryfikacja jakości modelu,

Opis parametrów użytych w modelu XGBoost:

Nrounds – określa liczbę rund uczenia, w każdej rundzie tworzony jest nowy model bazowy.

Eta – jest to współczynnik odpowiedzialny za korekty wagi w kolejnych modelach. Niska wartość oznacza mniejsze korekty co może być pomocne w przypadku problemu przeuczenia jednak wydłuża to proces uczenia modelu.

Max.depth – Odpowiada za głębokość każdego drzewa.

Subsample – Kontroluje liczbę obserwacji jakie są przekazywane do każdego drzewa. Mniejsza wartość oznacza większą losowość w doborze obserwacji.

Colsample_bytree – określa jaka liczba zmiennych będzie przekazywana do każdego drzewa, tak samo jak w przypadku subsample mniejsza wartość od 1 oznacza większą losowość.

Min.child.weight – jeśli suma wag „dzieci" jest mniejsza niż ten parametr to węzeł nie jest tworzony.

Na początku dane zostały podzielone na zbiór testowy i uczący analogicznie jak w poprzednich przykładach. Oba zbiory zostały przekształcone do formatu xgb.DMatrix oraz wyodrębnione zostały etykiety danych.

Ilustracja 32. Kod źródłowy dla podziału na zbiór treningowy i testowy.

```
set.seed(123)
ind <- sample(2, nrow(danexboost), replace = TRUE, prob = c(0.7, 0.3))
train_set <- danexboost[ind==1,]
test_set <- danexboost[ind==2,]

dtrain <- xgb.DMatrix(data = as.matrix(train_set[, -length(test_set)]), label = train_set[,length(test_set)])
dtest <- xgb.DMatrix(data = as.matrix(test_set[, -length(test_set)]), label = test_set[,length(test_set)])
```

Źródło: opracowanie własne.

Następnie określone zostały parametry funkcji celu i metryki ewaluacji, oraz utworzona została lista obiektów zawierająca zbiór treningowy oraz testowy.

Ilustracja 33. Kod źródłowy dla określenia parametrów.

```
xgb_params <- list("objective" = "binary:logistic",
                   "eval_metric" = "logloss"
                   )
watchlist <- list(train = dtrain, test = dtest)
```

Źródło: opracowanie własne.

Kolejnym krokiem było utworzenie modelu z podstawowymi wartościami parametrów i na jego podstawie dobranie odpowiedniej wartości liczby rund uczenia oraz eta. Dobór tej wartości był oparty na minimalizacji logarytmicznego błędu logistycznego. W poniższej tabeli znajdują się wartości parametrów, z którymi został rozpoczęty trening modelu. Aby uzyskać w modelu powtarzalność wyników, zostało ustawione ziarno losowości.

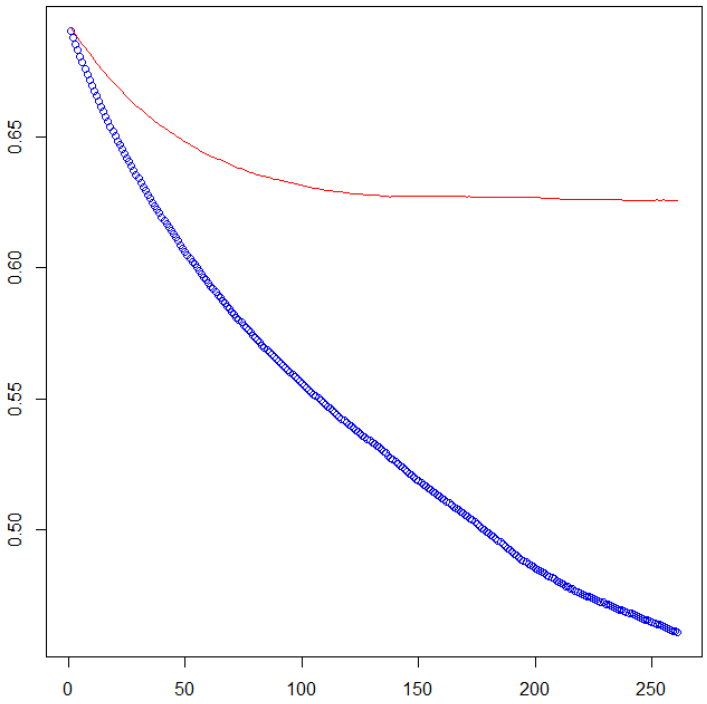
Tabela 23. Początkowe wartości parametrów.

Parametr	Wartość
Max.depth	6
Subsample	0,8
Colsample_bytree	1
Min.child.weight	1
Nrounds	500
Eta	0,1

Źródło: Opracowanie własne.

W celu lepszego zrozumienia zależności miedzy wartościami eta i nround został wygenerowany wykres przedstawiający jak zmienia się logarytmiczny błąd logistyczny w zbiorze testowym(na czerwono) oraz treningowym(na niebiesko).

Ilustracja 34. Wykres rozkładu błędu na zbiorze treningowym i testowym.



Źródło: opracowanie własne.

W celu wyznaczenia optymalnej wartości nround dla ustalonego poziomu eta za pomocą funkcji xgb.cv została przeprowadzona walidacja krzyżowa.

Ilustracja 35. Kod źródłowy dla doboru wartości nround.

```
params <- list(booster = "gbtree", objective = "binary:logistic", eta=0.01,
               gamma=0, max_depth=6, min_child_weight=1, subsample=0.8, colsample_bytree=1)

xgbcv <- xgb.cv( params = params, data = xgtrain, nrounds = 500, nfold = 5,
                 showsd = T, stratified = T, print_every_n = 10, early_stop_round = 20, maximize = F)


xgbcv$evaluation_log[xgbcv$evaluation_log$test_logloss_mean==min(xgbcv$evaluation_log$test_logloss_mean),]
```

Źródło: opracowanie własne.

Następnie została przeprowadzona weryfikacja podstawowego modelu na danych testowych oraz treningowych.

Ilustracja 36. Kod źródłowy dla weryfikacji modelu.

```
preds <- predict(xg_model, xgtest)
preds <- ifelse(preds > 0.5, 1, 0)
preds <- as.factor(preds)
x <- as.factor(test_set[,length(test_set)])

table(pred = preds, true = test_set[,length(test_set)])
confusionMatrix(preds,x)

preds <- predict(xg_model, xgtrain)
preds <- ifelse(preds > 0.5, 1, 0)
preds <- as.factor(preds)
x <- as.factor(train_set[,length(train_set)])
table(pred = preds, true = train_set[,length(train_set)])
confusionMatrix(preds,x)
```

Źródło: opracowanie własne.

W kolejnym etapie przeprowadzono optymalizacje hiperparametrów z wykorzystaniem funkcji mlr(). W tym celu utworzone zostały zadania klasyfikacji na podstawie danych treningowych i testowych oraz obiekt uczący wykorzystujący algorytmy klasyfikacji XGBoost. W obrębie obiektu uczącego zdefiniowano również wcześniej ustalone wartości nround oraz eta. Ponadto, skonfigurowano siatkę parametrów, która służyła do identyfikacji optymalnych wartości hiperparametrów. Siatka ta zawierała różne kombinacje parametrów, które miały zostać przebadane pod kątem ich wpływu na jakość modelu. W celu oceny i porównania wyników modelu została zastosowana walidacja krzyżowa.

Ilustracja 37. Kod źródłowy dla doboru hiperparametrów.

```
test_set$HomeWin <- as.factor(test_set$HomeWin)
train_set$HomeWin <- as.factor(train_set$HomeWin)
traintask <- makeClassifTask (data = train_set,target = "HomeWin")
testtask <- makeClassifTask (data = test_set,target = "HomeWin")

lrn <- makeLearner("classif.xgboost",predict.type = "response")
lrn$par.vals <- list( objective="binary:logistic", eval_metric="error", nrounds=271, eta=0.01,set.seed(123))


params <- makeParamSet(makeIntegerParam("max_depth",lower = 3L,upper = 10L),
                       makeNumericParam("min_child_weight",lower = 1L,upper = 10L),
                       makeNumericParam("subsample",lower = 0.5,upper = 1),
                       makeNumericParam("colsample_bytree",lower = 0.5,upper = 1))


resample <- makeResampleDesc("CV",stratify = T,iters=5L)
ctrl <- makeTuneControlRandom(maxit = 10L)
tune_params <- tuneParams(learner = lrn, task = traintask,
                     resampling = resample, measures = acc,
                     par.set = params ,control = ctrl,show.info = T)

tune_params$y
```

Źródło: opracowanie własne.

Ostatnim krokiem było przypisanie wyznaczonych wartości parametrów do modelu, a następnie obniżanie wartości eta wraz z jednoczesnym podnoszeniem wartości nrounds, oraz sprawdzenie różnych zestawów danych. Poniżej została zamieszczona tabela z rozkładem wag zmiennych w modelu.

Tabela 24. Wagi zmiennych w modelu 5.

Zmienna	Waga
AwayTeam_Value	0,18372917
HT_AvgPointsCurrentS	0,16975184
HomeTeam_Value	0,14523383
AT_AvgPointsCurrentS	0,10601171
HT_AvgGoalsHLastS	0,09426241
HT_ShotsOTLast5g	0,07103434
AT_GoalsConLast5	0,05761435
h2h_diff	0,04167142
AT_ShotsOTLast5g	0,03987216
HT_Goals_Last5g	0,02431244
AT_Goals_Last5g	0,02328586
HT_pointsHLast3g	0,01859008
HT_GoalsConLast5	0,01494303
AT_pointsALast3g	0,00968736

Źródło: Opracowanie własne.

Ponownie najlepsze rezultaty osiągnięto z podstawowym zestawem danych powiększonym o zmienna HT_AvgGoalsHLastS.

4. Przedstawienie wyników

Model 1 – model z zastosowaniem algorytmu drzewa decyzyjnego utworzony z podstawowego zbioru zmiennych i domyślnych wartości parametrów.

Model 2 – model 1 z dobranymi wartościami hiperparametrów.

Model 3 – model 2 po zastosowaniu procesu przycięcia drzewa.

Model 4 – model z zastosowaniem algorytmu lasów losowych z parametrami zaprezentowanymi w tabeli 22.

Modele 5,6,7 zostały utworzone na podstawie algorytmu XGBoost, ich parametry zostały zaprezentowane w poniższej tabeli.

Tabela 25. Hiperparametry modeli XGBoost.

	Model 5	Model 6	Model 7
Booster	gbtree	gbtree	gbtree
Nrounds	547	134	152
Eta	0,01	0,04	0,03
Max depth	4	3	3
Subsample	0,842	0,704	0,704
Colsample by tree	0,764	0,992	0,992
Min child weight	1	4,98	4,98

Źródło: Opracowanie własne.

Tabela 26. Macierz pomyłek modelu 1.

Wartość przewidywana	Wartość rzeczywista	
	1	0
1	232	145
0	142	298

Źródło: Opracowanie własne.

Tabela 27. Macierz pomyłek modelu 2.

Wartość przewidywana	Wartość rzeczywista	
	1	0
1	216	131
0	158	312

Źródło: Opracowanie własne.

Tabela 28. Macierz pomyłek modelu 3.

Wartość przewidywana	Wartość rzeczywista	
	1	0
1	235	147
0	139	296

Źródło: Opracowanie własne.

Tabela 29. Macierz pomyłek model 4.

Wartość przewidywana	Wartość rzeczywista	
	1	0
1	181	79
0	193	364

Źródło: Opracowanie własne.

Tabela 30. Macierz pomyłek model 5.

Wartość przewidywana	Wartość rzeczywista	
	1	0
1	205	103
0	169	340

Źródło: Opracowanie własne.

Tabela 31. Macierz pomyłek model 6.

Wartość przewidywana	Wartość rzeczywista	
	1	0
1	207	106
0	167	337

Źródło: Opracowanie własne.

Tabela 32. Macierz pomyłek model 7.

Wartość przewidywana	Wartość rzeczywista	
	1	0
1	211	103
0	163	340

Źródło: Opracowanie własne.

Tabela 33. Porównanie miar wyliczonych z macierzy pomyłek.

Miara	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7
Dokładność	64,87%	64,63%	64,99%	66,71%	66,71%	66,59%	67,44%
Precyzja	61,54%	62,24%	61,52%	69,61%	66,56%	66,13%	67,2%
Czułość	67,27%	70,43%	66,82%	82,17%	76,75%	76,07%	76,75%
Swoistość	62,03%	57,75%	62,83%	48,40%	54,81%	55,35%	56,42%
Łączny błąd klasyfikowania	35,13%	35,37%	35,01%	33,29%	33,29%	33,41%	32,56%
AUC	69,64%	67,44%	69,59%	69,59%	72,1%	71,49%	71,54%

Źródło: Opracowanie własne.

Tabela 34. Rentowność modelów na podstawie kursów bukmacherskich.

	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7
Zysk/strata	-3051,834	-2966,27	-2813,834	-520,227	2034,916	1764,577	-542,6445
% rentowności	-9,54%	-9,27%	-8,79%	-1,63%	6,36%	5,51%	-1,70%

Źródło: Opracowanie własne.

4.1 Omówienie wyników

Oprócz klasycznych miar oceny jakości modelu, zdolność do predykcji nowych danych każdego modelu została sprawdzona na 320 meczach sezonu 2022/2023 angielskiej Premier League. Z uwagi na braki danych dotyczące pierwszych kolejek do predykcji nie zostały wykorzystane dane dotyczące pierwszych sześciu kolejek. Zysk modelu reprezentuje sumę uzyskaną w przypadku obstawiania każdego meczu za kwotę 100 jednostek. W przypadku poprawnej predykcji zysk z pojedynczego meczu obliczana jest według formuły

$$zysk = kurs * stawka - stawka \qquad (17)$$

W przypadku błędnej predykcji notowana jest strata równa obstawionej kwocie. Procent rentowności obliczany jest jako suma zysku podzielona przez całą zainwestowana kwotę. W przypadku badanej dyscypliny najlepsi eksperci osiągają rentowność na poziomie 5%-15%. Zgodnie z przewidywaniami najmniej złożone algorytmy drzew decyzyjnych osiągnęły najsłabszy wynik. Co ciekawe regulacja parametrów w tym przypadku poprawiła jedynie minimalnie dokładność oraz rentowność. Algorytm lasów losowych osiągnął taką samą skuteczność jak 5 model utworzony z wykorzystaniem algorytmu XGBoost, jednak znacząco różniła się rentowność obu modeli z korzyścią po stronie modelu 5 który wypracował zysk na poziomie 6.36% co było jednocześnie najlepszym wynikiem pośród wszystkich zastosowanych modeli. Porównując macierze pomyłek wszystkich modeli można dostrzec różnice w tendencji typowania wyników, modele oparte na algorytmie drzew losowych przewidywały zwycięstwo gospodarzy średnio

w 45% przypadków, natomiast algorytm lasów losowych oraz XGBoost miały tendencje do faworyzowania braku wygranej gospodarzy.

Model 4 przewidywał ten rezultat w 68.18% przypadków w rezultacie czego z powodzeniem udało mu się prognozować największy procent wszystkich meczów zakończonych takim rezultatem(82.17%) jednak nie przełożyło się to na wypracowanie większego zysku. Model 5 natomiast przewidywał brak zwycięstwa gospodarzy w 63% przypadków z czego 66.56% było poprawnych. Ponieważ modele 4-7 osiągnęły dużo lepsze wyniki w kontekście zysku można założyć, że dobrą drogą jest przykładanie większej wagi do poprawnego przewidywania meczów zakończonych wygraną drużyny przyjezdnej lub remisem. Najwyższą skuteczność wśród wszystkich opracowanych modeli osiągnął model numer 7. Niemniej jednak, warto zaznaczyć, że mimo wysokiej skuteczności, ten model wykazał ujemną rentowność. To może sugerować, że model poprawnie przewidział wyniki większej liczby meczów które miały wyraźnego faworyta co jest związane z mniejszym kursem bukmacherskim, ten sam problem prawdopodobnie dotyczył algorytmu drzew decyzyjnych. Model 3 poprawnie wytypował największy procent wygranej gospodarzy jednak, z analizy wyglądu drzewa oraz wagi zmiennych wynika, że brał on pod uwagę w dużo większym stopniu niż inne modele zmienne charakteryzujące faworytów spotkań takie jak wartość drużyny oraz średnia punktów zdobywanych w obecnym sezonie.

4.2. Podsumowanie i kroki na przyszłość

Przewidzenie wyników meczów piłki nożnej stanowi wyzwanie z uwagi na dużą liczbę niezależnych czynników, które wpływają na ostateczny rezultat. Otrzymane wyniki dają jednak podstawę do twierdzenia, że modele oparte na drzewach losowych oraz ogólnodostępnych danych mogą stanowić pomoc w poprawnej predykcji meczów piłki nożnej. Najbardziej obiecujące rezultaty uzyskano przy zastosowaniu techniki XGBoost, gdzie najlepszy model osiągnął dokładność na poziomie 66,71%. Warto zaznaczyć, że ten model przyniósł zysk na poziomie 6,36%. To zadowalający wynik, biorąc pod uwagę, że typowa stopa zwrotu ekspertów w tym obszarze oscyluje w przedziale 5% - 15%.

Jednak z racji na to, że nie wszystkie modele wykonane ta metodą przyniosły zysk, warto byłoby w przyszłości udoskonalić model o większą liczbę zmiennych takich

jak kontuzje kluczowych zawodników, bardziej szczegółowe statystyki meczowe oraz opinie ekspertów.

Literatura

A, Uberoi, Introduction to Dimensionality Reduction, ECE539 2003r.;

A. Król-Nowak, K. Kotarba, *Podstawy uczenia maszynowego,* wydawnictwo AGH, Kraków 2022r.;

A. M. Nevill, S. M. Newell, S. Gale, *Factors associated with home advantage in English and Scottish soccer matches,* „Journal of Sports Sciences" 1996r.;

A. McCabe, J. Trevathan, *Artificial intelligence in sports prediction*, Information Technology: New Generations, ITNG 2008r.;

A. Uberoi, Introduction to Dimensionality Reduction, „GeeksforGeeks" 2023r.;

A. Waters, G. Lovell, *An Examination of the Homefield Advantage in a Professional English Soccer Team from a Psychological Standpoint*, „Football Studies", vol. 5 no. 1 2002r.;

B. Ulmer, M.P. Fernández, *Predicting Soccer Match Results in the English Premier League*, Stanford University 2014r.;

D. Banasiak, *Sztuczna inteligencja – Uczenie się maszyn – wykład 9,* Politechnika Wrocławska;

D. E. Hinkle, W. Wiersma, *Applied Statistics for the Behavioral Sciences,* Houghton Mifflin, tom 663, Londyn 2003r.;

D. Miljković, L. Gajić, A. Kovačević, Z. Konjović*, The Use of Data Mining for Basketball Matches Outcomes Prediction*, Faculty of Tehnical Sciences, Novi Sad, Republic of Serbia 2010r.;

D. Prasetio, D. Harlili, *Predicting football match results with logistic regression*, International Conference On Advanced Informatics: Concepts, Theory And Application ICAICTA 2016r.;

E. Gatnar, *Agregacja modeli dyskryminacyjnych*, Prace Naukowe Akademii Ekonomicznej we Wrocławiu. Taksonomia , Wrocław 2002r.;

H. R. Azhari, Y. Widyaningsih, D. Lestari, *Predicting Final Result of Football Match Using Poisson Regression Model,* Universitas Indonesia, Kampus Baru UI 2018r.;

J. Stübinger, B, Mangold, J. Knoll, *Machine Learning in Football Betting: Prediction of Match Results Based on Player Characteristics,* University of Erlangen-Nürnberg 2019r.;

K. Y. Huang, W. L. Chang, *A neural network method for prediction of 2006 world cup football game,* International Joint Conference on Neural Networks (IJCNN) 2010r.;
M. C. Purucker, *Neural network quarterbacking vol. 15*, IEEE Potentials 1996r.;
M. Gromada, *Confusion matrix, Macierz błędu, tablica / macierz pomyłek – czyli ocena jakości klasyfikacji (część 1),* mathspace,pl 2015r.;

M. Gromada, *Zasięg (TPR – czułość / TNR – specyficzność) i precyzja (PPV / NPV) – czyli ocena jakości klasyfikacji (część 2),* mathspace,pl 2015r.;

M. Heijboer, *Predicting football match outcomes using machine learning algorithms,* Tilburg University 2022r.;
N. J. Nilsson, *Introduction to machine learning,* Stanford University 1998r.;
N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, *SMOTE: Synthetic Minority Over-sampling Technique,* AI Access Foundation and Morgan Kaufmann Publishers 2002r.;
P. A. Flach, *Machine Learning. The Art and Science of Algorithms that Make Sense of Data,* Cambridge University Press 2012r.;

P. B. Brandtzæg, *Big Data, for Better or Worse: 90% of World's Data Generated Over Last Two Years,* Sintef 2013r.;
P. Marcin, *Podejście wielomodelowe w regresji danych symbolicznych interwałowych*, Prace Naukowe Uniwersytetu Ekonomicznego we Wrocławiu;
P.Denderski, *Metody identyfikacji obserwacji odstających*, Instytut Nauk Ekonomicznych Polskiej Akademii Nauk University of Leicester, luty 2019r.;

T. Mitchell, *Machine Learning,* McGraw Hill, 1997r.;

Wykaz innych materiałów

https://www.vantagemarketresearch.com/industry-report/sports-betting-market-1710 - data odczytu 10.08.2022r.;

https://osgamers.com/frequently-asked-questions/how-much-money-does-bet-365-make - data odczytu 10.08.2022r.;

https://pl.wikipedia.org/wiki/Regresja_(statystyka) – data odczytu 22.07.2022r.;

https://pl.wikipedia.org/wiki/Klasyfikacja_statystyczna - data odczytu 22.07.2023r.;

https://datascience.eu/pl/uczenie-maszynowe/bezobslugowe-uczenie-sie-maszyn/ - data odczytu 22.07.2023r.;

https://www.javatpoint.com/dimensionality-reduction-technique - data odczytu 26.07.2023r.;

https://pl.wikipedia.org/wiki/Analiza_skupie%C5%84 – data odczytu 26.07.2023r.;

https://pl.wikipedia.org/wiki/Tablica_pomyłek – data odczytu 23.07.2023r.;

https://azure.microsoft.com/pl-pl/resources/cloud-computing-dictionary/what-is-machine-learning-platform - data odczytu 22.07.2023r.;
https://mfiles.pl/pl/index.php/Drzewo_decyzyjne - data odczytu 25.07.2023r.;

https://analizadanychwpilce.com/2018/10/15/wykorzystanie-modelu-drzewa-decyzyjnego-do-analizy-wynikow-spotkania/ - data odczytu 25.07.2023r.;

https://www.mimuw.edu.pl/~son/datamining/DM2008/W06.pdf - data odczytu 23.07.2023r.;

https://stat.ethz.ch/R-manual/R-devel/library/cluster/html/diana.html - data odczytu 02.08.2023r.;

Spis tabel

Spis ilustracji

Spis równań