1. Cel

Pierwszym celem projektu jest zbudowanie modelu, który umożliwi klasyfikację klubów piłkarskich na „dobre" lub „złe" na podstawie różnych atrybutów opisujących ich wyniki i charakterystyki. Klasyfikacja ta opiera się na wartości zmiennej GVB (Good vs Bad), gdzie wartość 1 oznacza „dobry" klub, a 0 oznacza „zły". Na potrzeby tego zadania wykorzystano zestaw danych zawierający różne cechy statystyczne niemalże wszystkich klubów piłkarskich występujących w najwyższej klasie rozgrywkowej swojego kraju. Celem jest więc stworzenie modelu, który na podstawie cech, które zostaną opisane w kolejnym rozdziale, będzie w stanie przewidzieć, czy dany klub powinien być sklasyfikowany jako „dobry" czy „zły". Taka klasyfikacja może pomóc w ocenie sportowej i komercyjnej wartości drużyn oraz wspierać decyzje menedżerskie i operacyjne dotyczące rozwoju klubu.

Drugim celem projektu jest ocena skuteczności różnych klasycznych modeli uczenia maszynowego w kontekście klasyfikacji klubów piłkarskich na „dobre" lub „złe". W tym celu wybrano cztery modele takie jak SVM, Drzewo decyzyjne, Las losowy i Klasyfikator głosujący. Analiza obejmuje porównanie wyników modeli przed i po optymalizacji ich parametrów, przy czym ocenie poddane są takie metryki jak dokładność, precyzja, czułość i wynik F1. Proces optymalizacji ma na celu dopasowanie parametrów modeli do specyfiki danych, co powinno skutkować poprawą jakości predykcji. Ostatecznym celem drugiej części analizy jest określenie, który z modeli najlepiej sprawdza się w praktyce i które parametry mają największy wpływ na ich efektywność.

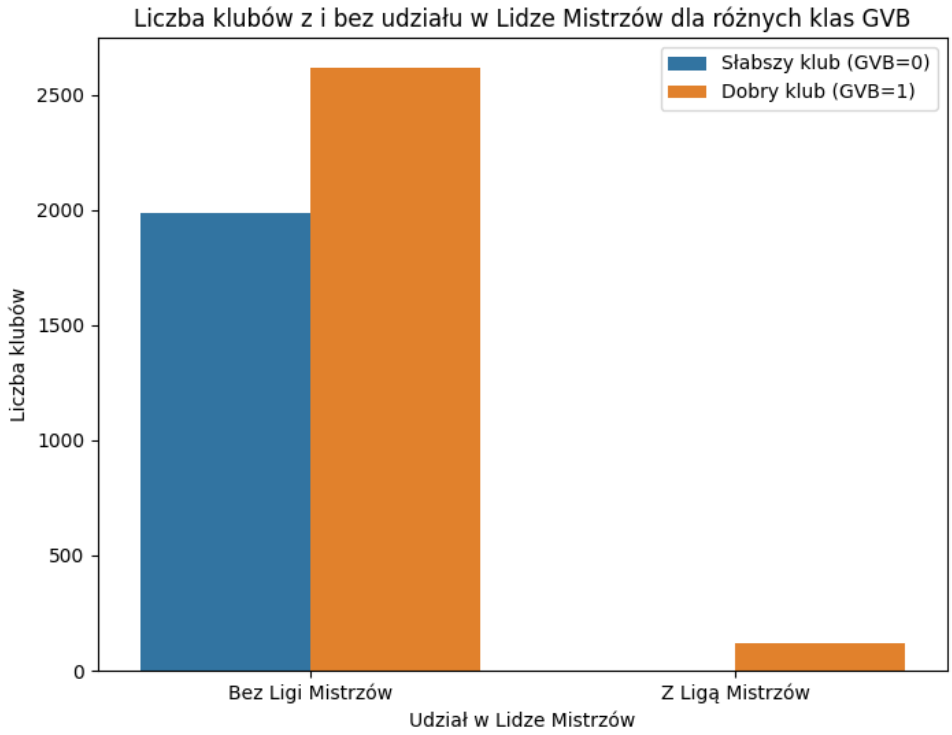
2. Materiały

Dane pochodzą ze strony kaggle.com i zawierają informacje na temat klubów piłkarskich. Każdy rekord odpowiada pojedynczemu klubowi, a kolumny opisują różne cechy klubu. Zbiór danych składa się z następujących kolumn:

- **Club**: Nazwa klubu piłkarskiego (identyfikator tekstowy).
- **IntReputation**: Międzynarodowa reputacja zawodników danego klubu.
- **Age**: Średni wiek zawodników klubu.
- **SkillMoves**: Średnia ocena umiejętności technicznych najlepszych 10 zawodników w klubie.

- **Crossing**: Średnia umiejętność dośrodkowań najlepszych 5 zawodników.
- **Finishing**: Średnia umiejętność wykańczania akcji najlepszych 5 zawodników.
- **HeadingAccuracy**: Średnia dokładność główek najlepszych 7 zawodników.
- **ShortPassing**: Średnia umiejętność podań krótkich.
- **Volleys**: Średnia ocena umiejętności tak zwanego woleja, odzwierciedlająca precyzję i technikę strzałów z powietrza.
- **Dribbling**: Średnia ocena dryblingu.
- **BallControl**: Średnia ocena kontroli nad piłką.
- **Acceleration**: Średnia przyspieszenia zawodników, kluczowa w kontratakach.
- **SprintSpeed**: Średnia maksymalna prędkość zawodników klubu.
- **Agility**: Średnia zwinność zawodników danego zespołu.
- **ShotPower**: Średnia siła strzału.
- **Jumping**: Ocena skoczności zawodników.
- **LongShots**: Średnia skuteczność strzałów z daleka.
- **Aggression**: Ocena agresywności zespołu co może wpływać na styl gry drużyny.
- **Interceptions**: Skuteczność przechwyceń piłki, wskazująca na efektywność obrony drużyny.
- **Positioning**: Średnia ocena umiejętności ustawiania się zawodników w trakcie gry.
- **Vision**: Średnia ocena wizji gry zawodników, istotna w planowaniu i kreowaniu akcji ofensywnych.
- **Penalties**: Średnia skuteczność wykonywania rzutów karnych.
- **Composure**: Średnia ocena opanowania zawodników.
- **StandingTackle**: Skuteczność w obronie pozycyjnej.
- **SlidingTackle**: Średnia ocena umiejętności wykonywania wślizgów.
- **ChampionsLeague**: Zmienna binarna (0 lub 1), wskazująca na udział klubu w Lidze Mistrzów w danym sezonie.
- **GVB**: Klasa docelowa (Good vs Bad) przyjmująca wartość 1 dla klubów dobrych oraz 0 dla klubów słabszych.

Wszystkie powyższe cechy są liczbowe, poza kolumną Club (identyfikator tekstowy) i Champions League, której zleżność łatwo można było zwizualizować na wykresie [Rysunek 1]. Kolumna GVB jest zmienną docelową, a reszta stanowi predyktory, które mają być używane do przewidywania klasy klubu.



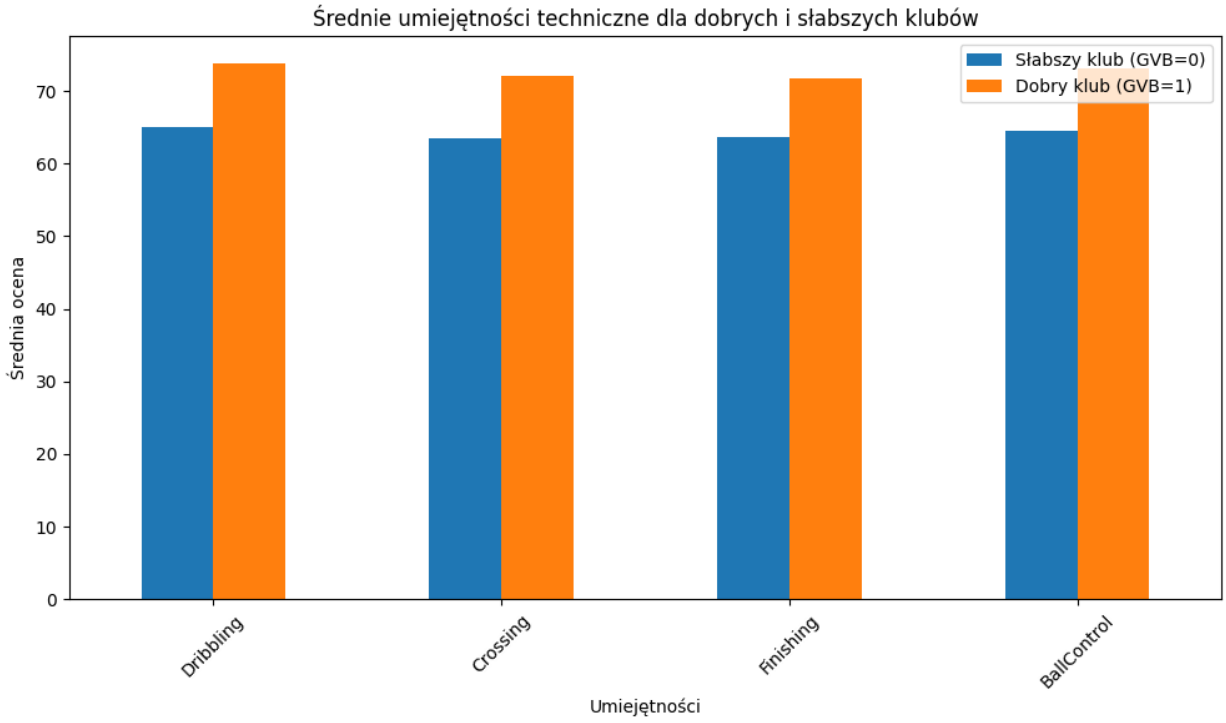
Rysunek 1

W celu zapewnienia kompletności zbioru danych, przeprowadzono analizę brakujących wartości w każdej kolumnie. Na etapie wstępnej analizy sprawdzono liczbę brakujących wartości w każdej kolumnie [Kod 1].

```
missing_values = df.isnull().sum()
if missing_values.sum() == 0:
    print("Brakujące wartości w zbiorze danych: Brak brakujących wartości")
else:
    print("Brakujące wartości:\n", missing_values)
```

Kod 1

W analizowanym zbiorze danych nie wykryto brakujących wartości, co oznacza, że wszystkie cechy są kompletne i gotowe do przetwarzania. Gdyby jednak brakowały dane, rozważone byłyby metody imputacji, takie jak wypełnienie średnią, medianą lub innymi metodami opartymi na dystrybucji wartości w danej kolumnie. Tak prezentuje się analiza umiejętności technicznych zespołów:



Rysunek 2

Przed modelowaniem istotne było sprawdzenie, czy zbiór danych jest zbalansowany pod względem klasy docelowej GVB [Kod 2]. W przeciwnym przypadku niezrównoważenie klas mogłoby prowadzić do błędnych wyników modeli klasyfikacyjnych. Rozkład klasy docelowej wykazał, że zbiór danych jest niezrównoważony, z przewagą jednej z klas. W celu wyrównania rozkładu zastosowano metodę nadpróbkowania SMOTE (Synthetic Minority Over-sampling Technique), która generuje syntetyczne przykłady klasy mniejszościowej, aby zapewnić równy rozkład między klasami.

```
class_balance = df['GVB'].value_counts()
print("Rozkład klasy docelowej GVB:\n", class_balance)
X = df.drop(columns=['Club', 'GVB'])
y = df['GVB']
smote = SMOTE(random_state=42)
X_resampled, y_resampled = smote.fit_resample(X, y)
new_class_balance = y_resampled.value_counts()
print("Rozkład klasy docelowej GVB po nadpróbkowaniu:\n", new_class_balance)
```

Kod 2

Dzięki zastosowaniu SMOTE udało się zbalansować zbiór danych, co zwiększa wiarygodność wyników uzyskanych z modeli klasyfikacyjnych. W celu zapewnienia porównywalności różnych cech, przeprowadzono standaryzację cech przy użyciu StandardScaler. Dzięki standaryzacji każda cecha ma średnią 0 i odchylenie standardowe 1, co pozwala algorytmom skutecznie wykorzystać różnorodne cechy bez ryzyka dominacji którejkolwiek z nich.

Po przeprowadzeniu nadpróbkowania oraz standaryzacji, uzyskany zbiór danych zapisano do pliku teams-stats-standard.csv, który zawiera wszystkie niezbędne cechy oraz klasy docelowe GVB. Nowym próbką wygenerowanym przez SMOTE przypisano unikalne identyfikatory. Dla celów modelowania danych podzielono zbiór na zestawy treningowy i testowy przy użyciu metody train_test_split. Dane wejściowe (X_scaled) oraz klasy (y) zostały rozdzielone w stosunku 70:30, co zapewnia odpowiednią ilość danych zarówno do trenowania, jak i do walidacji modeli.

3. Metody:

SVM

Dla modelu SVM (Support Vector Machine) skonstruowano wersję domyślną oraz zoptymalizowaną. SVM jest algorytmem klasyfikacyjnym, który poprzez odpowiednie ustawienie hiperparametrów stara się wyznaczyć optymalną hiperpowierzchnię rozdzielającą próbki danych należące do różnych klas. W modelu domyślnym użyto liniowego jądra, co pozwala na proste rozdzielenie danych, natomiast w modelu zoptymalizowanym zastosowano jądro RBF (z parametrami C=10 i gamma='auto'), które pozwala na lepsze dostosowanie się do bardziej złożonej struktury danych.

Optymalizacja hiperparametrów w wersji zoptymalizowanej odbyła się przy użyciu wyszukiwania siatki (Grid Search) i walidacji krzyżowej, co umożliwiło znalezienie najlepszych ustawień dla parametrów C i gamma. Parametr C kontroluje margines decyzyjny, pozwalając modelowi lepiej zróżnicować kategorie w przypadku klasy "Good" i "Bad", a gamma wpływa na zakres działania jądra RBF, pomagając modelowi lepiej uchwycić wzorce w danych.

Drzewo decyzyjne

W projekcie zastosowano dwa modele drzewa decyzyjnego – jeden z domyślnymi ustawieniami, a drugi zoptymalizowany przy użyciu wyszukiwania siatki parametrów (Grid Search). Drzewo decyzyjne jest klasyfikacyjnym algorytmem uczenia maszynowego, który poprzez budowanie hierarchii decyzji na podstawie cech wejściowych grupuje dane do określonych klas. Modele te

zostały stworzone w celu przewidywania zmiennej docelowej GVB, która informuje, czy klub piłkarski jest "dobry" (1), czy "zły" (0).

Las losowy

Dla modelu lasu losowego (random forest) zbudowano wersję domyślną oraz zoptymalizowaną z dostosowanymi ustawieniami hiperparametrów. Random Forest to algorytm klasyfikacyjny oparty na zespole drzew decyzyjnych, który klasyfikuje dane na podstawie uśrednionych wyników z wielu drzew.

Ustawienia zoptymalizowanego modelu obejmują zwiększenie liczby drzew do 10,000 (n_estimators=10000), co stabilizuje uśredniony wynik oraz zmniejsza wariancję predykcji. Głębokość drzew została ustawiona na 50 (max_depth=50), co umożliwia modelowi lepsze uchwycenie złożoności danych. Parametr min_samples_split=2 pozwala na bardziej szczegółowe podziały w drzewach, a min_samples_leaf=1 zwiększa elastyczność modelu, umożliwiając każdemu liściowi większą precyzję. Dodatkowo, ustawienie max_features='sqrt' ogranicza liczbę używanych cech w każdym podziale do pierwiastka z całkowitej liczby cech, co sprzyja różnorodności drzew i minimalizuje ryzyko nadmiernego dopasowania modelu do danych treningowych. Natomiast parametr bootstrap=True pozwala na losowe próbkowanie z powtórzeniami, co zwiększa stabilność modelu, uśredniając wyniki i ograniczając wpływ pojedynczych, nietypowych obserwacji na wynik końcowy.

Klasyfikator głosujący

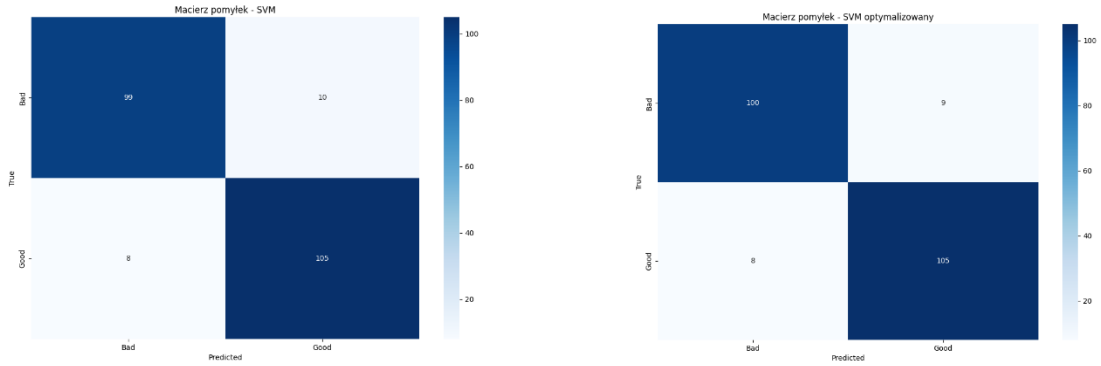
Klasyfikator z głosowaniem większościowym (Voting Classifier) został przygotowany w dwóch wariantach: wersji bazowej z domyślnymi ustawieniami modeli oraz wersji zoptymalizowanej, w której dostosowano hiperparametry dla poszczególnych modeli składowych. Voting Classifier łączy prognozy z różnych algorytmów, co umożliwia uzyskanie stabilniejszych i bardziej precyzyjnych wyników.

W zoptymalizowanej wersji klasyfikatora zastosowano GridSearchCV i RandomizedSearchCV do doboru najlepszych hiperparametrów, co pozwoliło na znaczącą poprawę dokładności predykcji. W skład klasyfikatora weszły różne modele, w tym Random Forest, Gradient Boosting, SVM oraz K-Nearest Neighbors (kNN). Dzięki wykorzystaniu mechanizmu głosowania miękkiego (voting='soft') i nadaniu większych wag modelom Random Forest i Gradient Boosting, Voting Classifier lepiej uśrednia prognozy, co przekłada się na większą dokładność i stabilność modelu w radzeniu sobie ze złożonymi danymi.

4. Wyniki i dyskusja

SVM

Analiza wyników modeli SVM z ustawieniami domyślnymi i zoptymalizowanymi opierała się na wskaźnikach oceny klasyfikacji oraz na wizualizacji wyników przy użyciu wykresów. Oba modele osiągnęły wysoką dokładność na testowym zbiorze danych. Poniżej przedstawiono szczegółowe wyniki [Rysunek 3]:

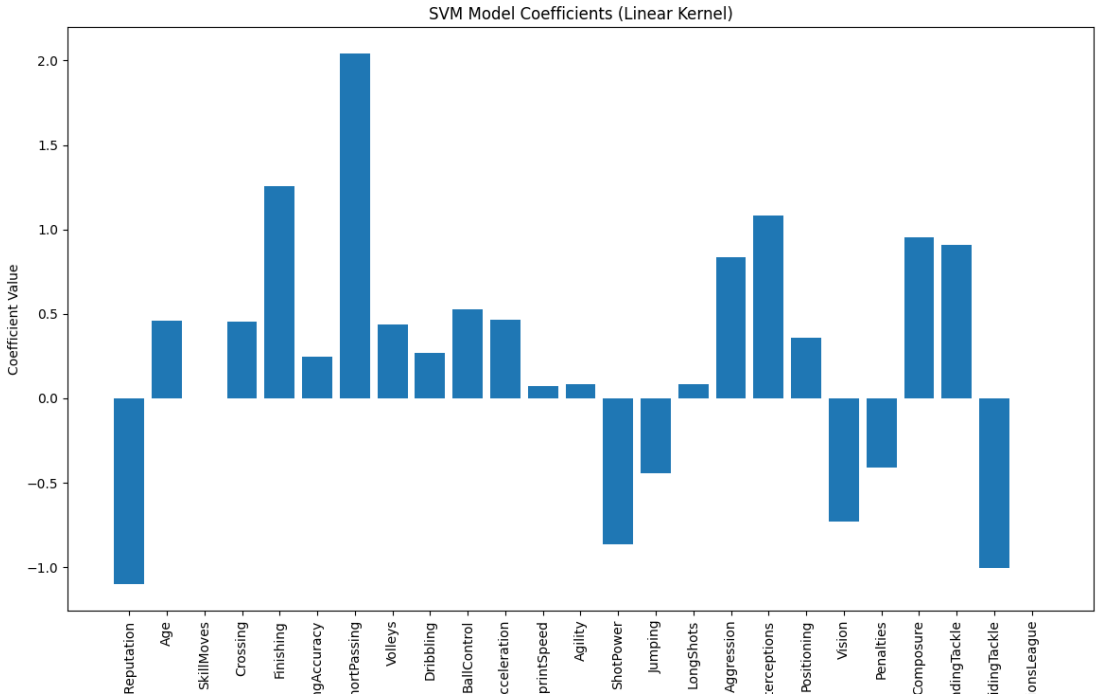


Rysunek 3

W wersji zoptymalizowanej model uzyskał minimalnie wyższe wartości w precyzji oraz czułości w przypadku klasy "Good", co pokazuje, że zmiana jądra z liniowego na RBF przyczyniła się do nieznacznej poprawy skuteczności. Wskaźnik F1-score również wzrósł dla klasy "Good".

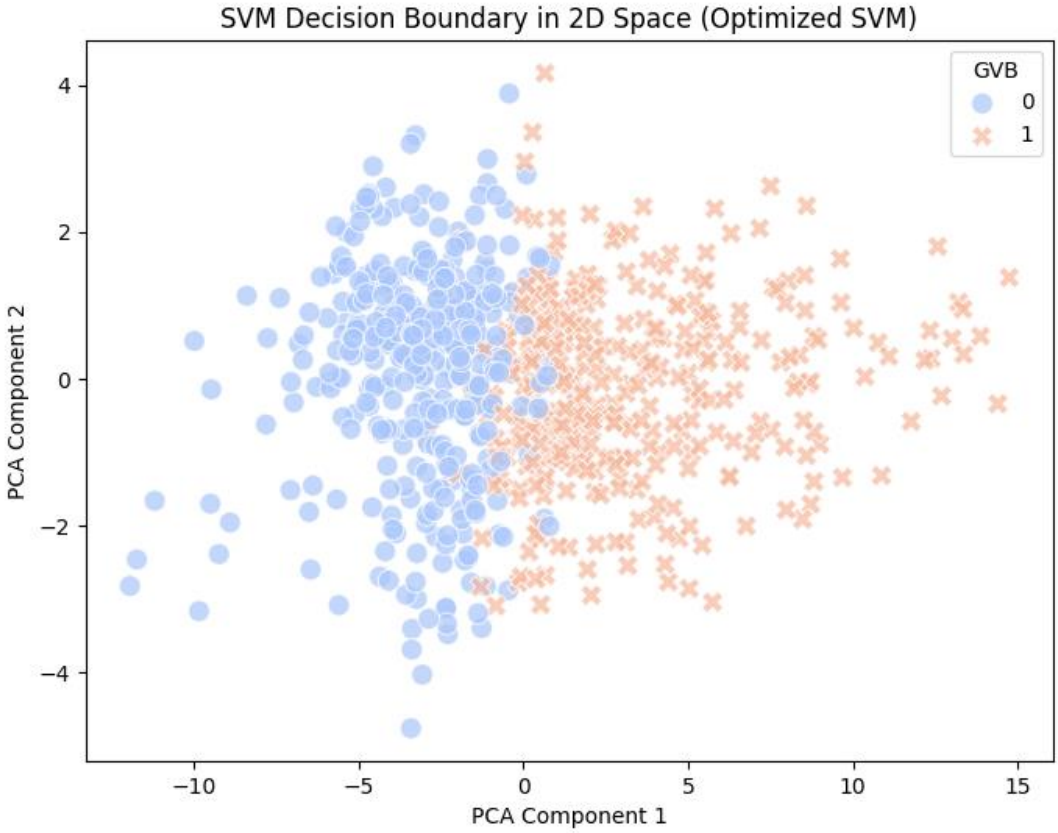
Rysunek 4 Macierz par atrybutów

Na [Rysunek 4] przedstawiono wykres zależności między wybranymi cechami względem zmiennej docelowej GVB. Kolory oznaczają klasy zmiennej GVB: kluby oznaczone jako „0” (kolor niebieski) oraz „1” (kolor pomarańczowy). Na przekątnych wykresu znajdują się histogramy rozkładu wartości danej cechy dla obu klas, co pozwala zobaczyć różnice między nimi. Ten wykres pomaga ocenić, które cechy mogą mieć znaczący wpływ na klasyfikację i zrozumieć, jak różne cechy są ze sobą powiązane.



Rysunek 5 Współczynniki modelu SVM

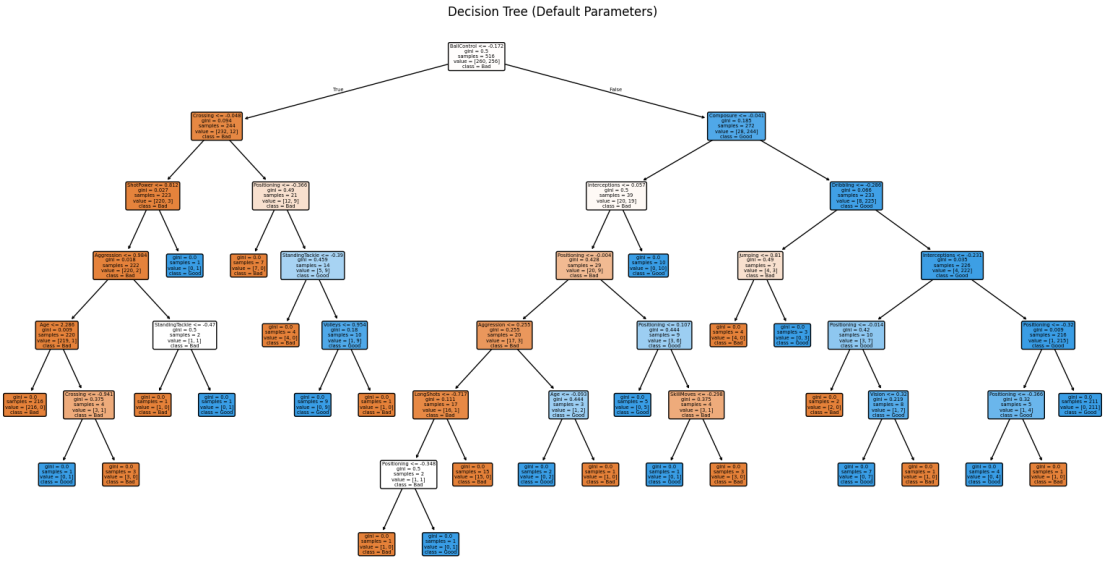
Wykres [Rysunek 5] przedstawia współczynniki przypisane różnym cechom przez model SVM z jądrem liniowym. Wartości dodatnie oznaczają, że wzrost danej cechy zwiększa prawdopodobieństwo przypisania klasy „1” (dobry klub), a wartości ujemne oznaczają, że wzrost cechy zwiększa prawdopodobieństwo przypisania klasy „0” (słaby klub). Wykres ten pozwala zrozumieć, które cechy mają największy wpływ na klasyfikację oraz w jaki sposób wpływają one na wynik końcowy modelu. Na przykład cechy takie jak „Passing” i „Finishing” mają najwyższe dodatnie współczynniki, co sugeruje, że mają znaczący wpływ na przypisanie klasy „1”.



Rysunek 6 Granica decyzyjna zoptymalizowanego modelu SVM

Na wykresie [Rysunek 6] przedstawiono granicę decyzyjną zoptymalizowanego modelu SVM w przestrzeni dwóch głównych składowych uzyskanych z analizy PCA (Principal Component Analysis). Obserwacje są oznaczone kolorami zgodnie z ich przynależnością do klasy GVB: „0” (niebieskie kółka) oraz „1” (pomarańczowe krzyżyki). Granica decyzyjna oddziela obszary przypisane do klas „0” i „1” i pokazuje, jak model SVM rozdziela dane w tej uproszczonej przestrzeni. Wykres ten umożliwia wizualną ocenę skuteczności klasyfikacji i pokazuje, jak dobrze model radzi sobie z rozdzielaniem klas w danych.

Drzewo decyzyjne

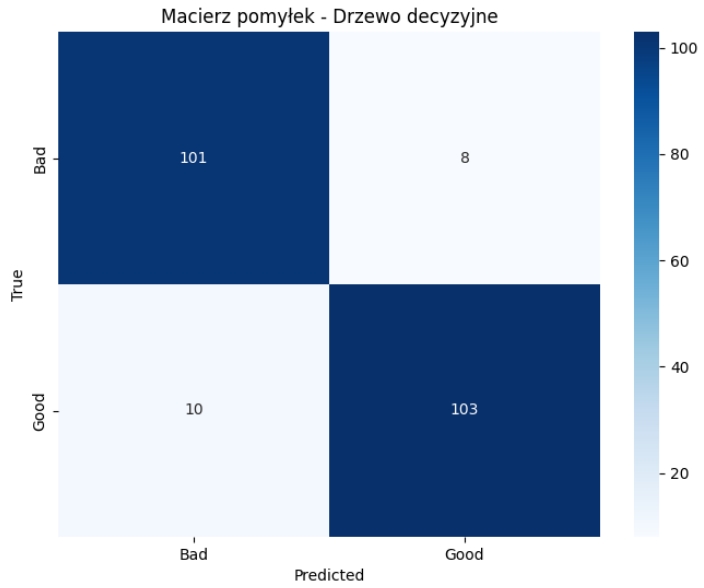


Rysunek 7 Drzewo decyzyjne

Powyższy diagram [Rysunek 7] przedstawia drzewo decyzyjne dla założonego problemu. Każdy węzeł drzewa reprezentuje decyzję opartą na jednej z cech, a każda gałąź prowadzi do kolejnych decyzji, aż do liści, które reprezentują ostateczną klasyfikację.

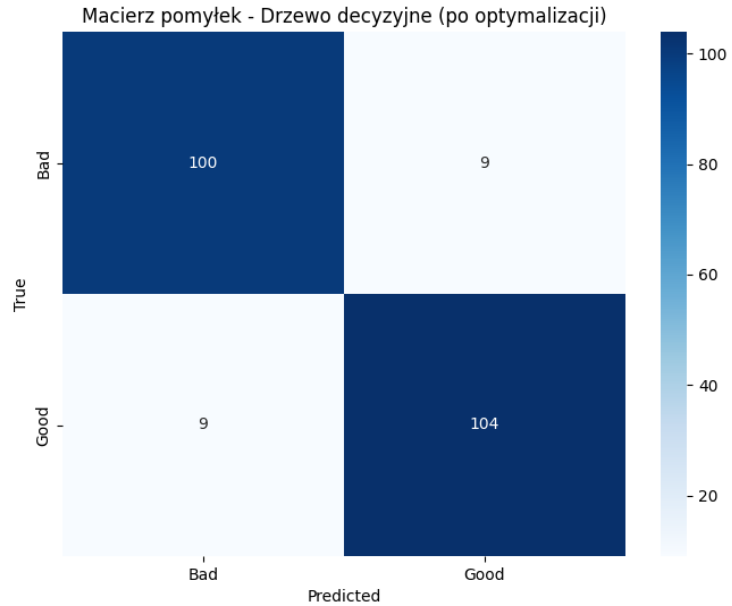
Wyniki klasyfikacji dla drzewa decyzyjnego, zarówno z ustawieniami domyślnymi, jak i po optymalizacji, zostały przedstawione za pomocą macierzy pomyłek. Każda macierz pokazuje liczbę prawidłowych i błędnych klasyfikacji dla klas "Bad" i "Good".

1. **Macierz pomyłek dla drzewa decyzyjnego z ustawieniami domyślnymi** [Rysunek 8]: w tej macierzy widzimy, że model z ustawieniami domyślnymi przewiduje klasy z dużą dokładnością. Klasa "Bad" została poprawnie zaklasyfikowana 101 razy, a klasa "Good" – 103 razy, przy niskiej liczbie błędnych klasyfikacji (8 dla "Bad" i 10 dla "Good").



Rysunek 8

2. **Macierz pomyłek dla drzewa decyzyjnego po optymalizacji** [Rysunek 9]: zoptymalizowany model wykazuje niemal identyczne wyniki jak model z ustawieniami domyślnymi, co wskazuje, że dodatkowe dostrajanie parametrów nie przyniosło znaczącej poprawy. Liczba błędnych klasyfikacji pozostała na podobnym poziomie (9 błędów dla "Bad" i 9 dla "Good").

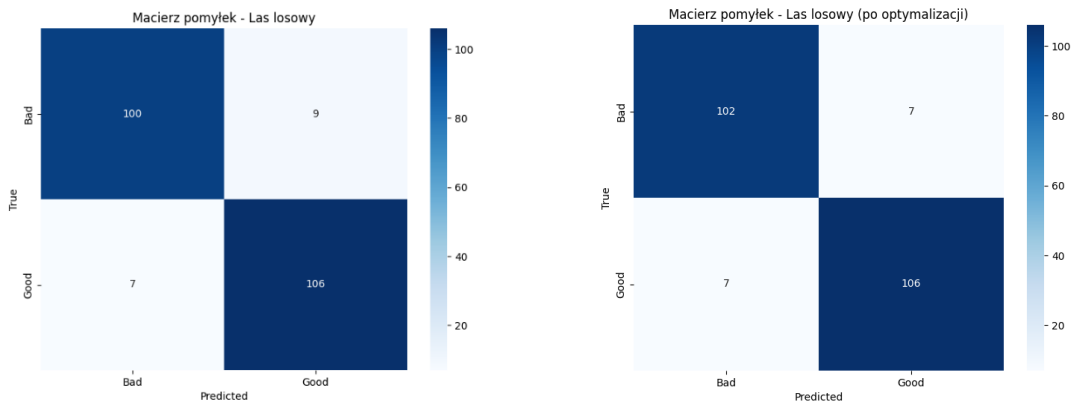


Rysunek 9

Wyniki uzyskane zarówno dla modelu drzewa decyzyjnego z ustawieniami domyślnymi, jak i zoptymalizowanego, są niemalże identyczne, z dokładnością wynoszącą 91.89% w obu przypadkach. Co więcej, raporty klasyfikacji dla obu modeli wskazują na bardzo podobne wartości precyzji, recall oraz F1-score, zarówno dla klasy "Bad" (0), jak i "Good" (1). Przyczyny identycznych wyników można upatrywać w tym, że struktura drzewa decyzyjnego, nawet przy domyślnych ustawieniach, była wystarczająca do skutecznego rozdzielenia obu klas. Zoptymalizowanie parametrów, takich jak głębokość drzewa czy minimalna liczba próbek w liściu, nie miało wpływu na wyniki, ponieważ drzewo decyzyjne bez tych zmian również świetnie radziło sobie z zadaniem. Dane były na tyle proste, że dodatkowa optymalizacja nie przyniosła znaczących korzyści.

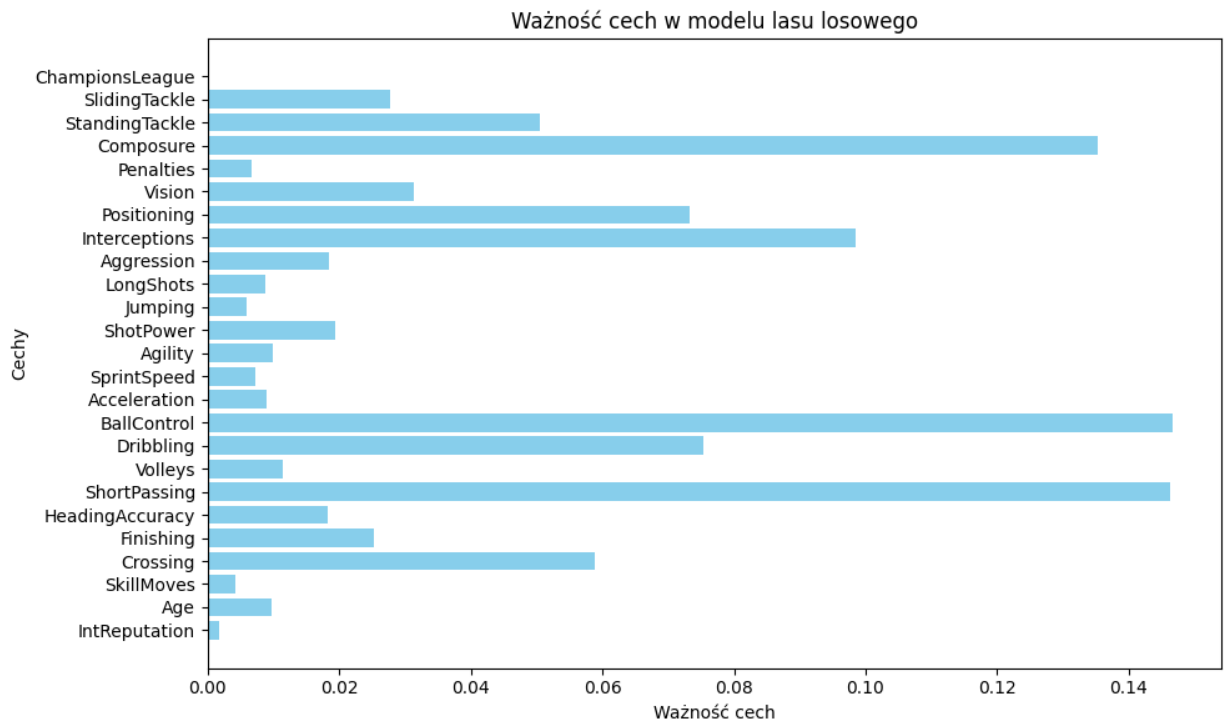
Las losowy

Analiza wyników dla modelu lasu losowego wykazała wysoką dokładność zarówno w wersji domyślnej (92.79%), jak i zoptymalizowanej (93.69%). Oba modele efektywnie klasyfikowały dane, osiągając precyzję i czułość na bardzo wysokim poziomie. Poniżej przedstawiono macierze pomyłek obu modeli [Rysunek 10].



Rysunek 10

Wyniki zaoptymalizowanego modelu wzbogacono o wykres ważności cech [Rysunek 11], który przedstawia cechy najbardziej wpływające na decyzję modelu. Jasno można odczytać, że najistotniejszymi cechami przy klasyfikacji jest kontrola and piłką oraz krótkie podania. To niewątpliwie niezwykle ważne aspekty piłki nożnej.



Rysunek 11

Klasyfikator głosujący

W celu oceny efektywności klasyfikatora głosującego (Voting Classifier) przeprowadzono eksperymenty z modelami o ustawieniach domyślnych oraz zoptymalizowanych. Wyniki analizowano, korzystając z metryk ewaluacyjnych, takich jak dokładność, precyzja, czułość (recall) oraz wynik F1, co pozwoliło na obiektywne porównanie obu wersji klasyfikatora.

Poniżej przedstawiono szczegółowe zmiany wprowadzone w drugim, tak zwanym zoptymalizowanym modelu:

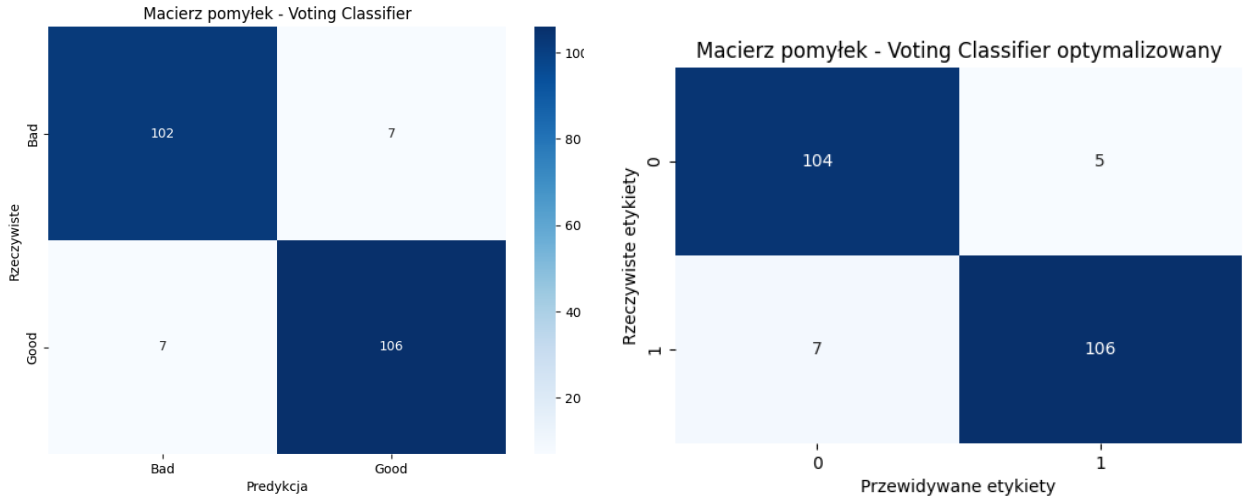
1. **Random Forest (Las losowy)**:
    - **n_estimators**: Zwiększono liczbę drzew z domyślnej (100) do maksymalnie 500, co pozwala modelowi lepiej uśredniać wyniki, zwiększając dokładność i stabilność klasyfikacji.
    - **max_depth**: Przetestowano i ustawiono maksymalną głębokość drzew na 15, dzięki czemu model zyskał możliwość dokładniejszego uchwycenia złożoności danych, ale z kontrolą głębokości, co redukuje ryzyko nadmiernego dopasowania.

- **min_samples_split** oraz **min_samples_leaf**: Zoptymalizowano minimalną liczbę próbek wymaganych do podziału (2, 5, 10) oraz liczbę próbek na liść (1, 2, 4), co zapewniło bardziej zrównoważone drzewo o większej ogólności.

2. **Gradient Boosting**:
    - **n_estimators**: Zwiększono liczbę estymatorów do 300, aby poprawić stabilność modelu.
    - **learning_rate**: Przetestowano mniejsze wartości współczynnika uczenia (0.01, 0.05, 0.1), co zwiększyło precyzję, umożliwiając modelowi lepsze dopasowanie do danych bez nadmiernego skoku na każdym kroku.
    - **max_depth**: Ograniczenie maksymalnej głębokości do 7 pozwoliło modelowi uchwycić kluczowe wzorce bez nadmiernego dopasowania.
3. **Support Vector Machine (SVM)**:
    - **C** oraz **kernel**: Przetestowano wartości C (0.1, 1, 10) i różne funkcje jądra (linear, rbf) w celu lepszego dopasowania marginesów do danych. Umożliwiło to modelowi bardziej elastyczne wyznaczanie granic decyzyjnych.
    - **probability=True**: Włączono obliczanie prawdopodobieństw, co jest kluczowe w soft voting.
4. **K-Nearest Neighbors (kNN)**:
    - **n_neighbors** oraz **weights**: Zoptymalizowano liczbę sąsiadów (3, 5, 7, 9) oraz sposób ich ważenia (uniform, distance). Dopasowanie tej liczby do danych pozwoliło na lepsze odwzorowanie lokalnych wzorców, co poprawiło precyzję klasyfikacji.
5. **Voting Classifier z głosowaniem miękkim wagami**:
    - Wersja zoptymalizowana korzysta z soft voting, gdzie każdy model dostarcza prawdopodobieństwa dla klasy zamiast samej klasyfikacji. Dodatkowo przypisano wagi dla najlepszych modeli (2 dla Random Forest i Gradient Boosting, 1 dla SVM i kNN, oraz 0.5 dla Logistic Regression), co zwiększyło ich wpływ na końcowy wynik.

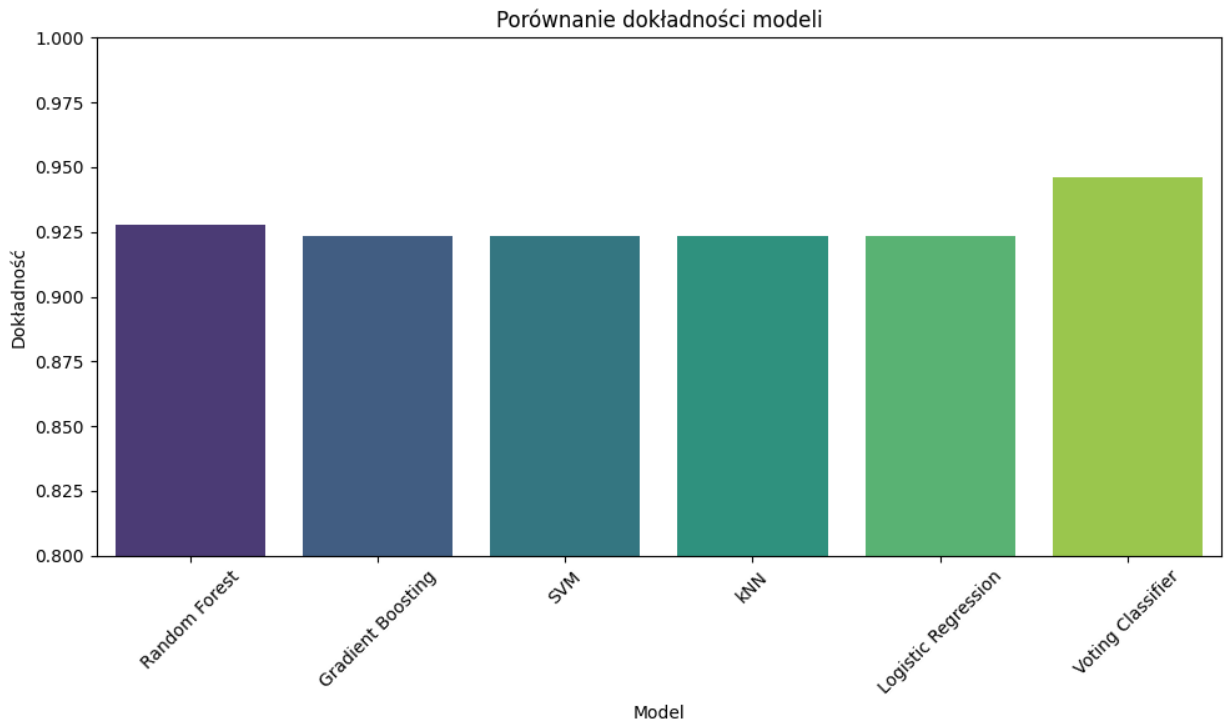
W przypadku klasyfikatora domyślnego uzyskano dokładność na poziomie 93.7%. Zarówno dla klasy 0, jak i klasy 1, wskaźniki precyzji, czułości i F1-score wynosiły 0.94, co wskazuje na równą jakość klasyfikacji dla obu klas. Średnie wartości dla obu klas (macro avg oraz weighted avg) również wynosiły 0.94, co świadczy o stabilności modelu.

Po przeprowadzeniu optymalizacji klasyfikatora wynik dokładności wzrósł do 94.6%, a wynik F1 dla obu klas wyniósł 0.95. Dzięki temu zoptymalizowany model charakteryzuje się większą spójnością i dokładnością, co przekłada się na lepsze wyniki we wszystkich wskaźnikach [Rysunek 12]. Precyzja dla klasy 1 wzrosła do 0.95, a czułość dla klasy 0 również osiągnęła 0.95, co podkreśla bardziej zrównoważoną i dokładną klasyfikację obu klas.



Rysunek 12

Optymalizacja poprawiła wyniki klasyfikacji, ale jednocześnie zwiększyła czas obliczeń do 33 sekund. Z tego względu należy rozważyć kompromis między wyższą dokładnością, a czasem przetwarzania, szczególnie w przypadku dużych zbiorów danych lub systemów wymagających szybkiej predykcji.



Rysunek 13

Wykres [Rysunek 13] pokazuje, że Voting Classifier osiągnął najwyższą dokładność w porównaniu do innych modeli. Voting Classifier łączył wyniki kilku innych modeli, co pozwoliło mu na uzyskanie lepszej dokładności niż każdy z nich oddzielnie. Pozostałe modele, takie jak Random Forest, Gradient Boosting, SVM, kNN oraz Logistic Regression, uzyskały podobne wyniki dokładności, mieszczące się w przedziale od 0.90 do 0.93.

5. Podsumowanie:

Projekt uwzględniał cztery modele uczenia maszynowego, analizując zarówno ich wersje bazowe, jak i zoptymalizowane:

1. **SVM (Support Vector Machine)**: Model bazowy z liniowym jądrem osiągnął dokładność 91.89%. Zoptymalizowany model z jądrem RBF poprawił wynik do 92.34%, co pokazało, że lepiej dopasowane jądro może uchwycić złożoność danych. Optymalizacja parametrów C i gamma metodą Grid Search umożliwiła modelowi lepsze wyznaczanie granicy decyzyjnej, co poprawiło precyzję i czułość klasyfikacji.

2. **Decision Tree (Drzewo decyzyjne)**: Wersja bazowa drzewa decyzyjnego oraz zoptymalizowana wykazały podobne wyniki, z dokładnością na poziomie 91.89%. Dodatkowe dostrojenie parametrów, takich jak głębokość drzewa czy minimalna liczba próbek w węźle, nie wpłynęło istotnie na poprawę klasyfikacji. Wynik ten wskazuje, że struktura danych była wystarczająco prosta, aby model bazowy skutecznie klasyfikował kluby bez potrzeby dalszej optymalizacji.

3. **Random Forest (Las losowy)**: Las losowy okazał się jednym z najefektywniejszych modeli, osiągając dokładność 93.69% w wersji zoptymalizowanej. Zwiększenie liczby drzew do 10,000 oraz dostosowanie parametrów, takich jak głębokość drzew i minimalna liczba próbek, poprawiło stabilność predykcji i skuteczność klasyfikacji.

4. **Voting Classifier (Klasyfikator głosujący)**: Zastosowanie klasyfikatora głosującego, który łączy wyniki kilku modeli (m.in. Random Forest, Gradient Boosting, SVM, kNN) z wagami przypisanymi modelom o najwyższej skuteczności, przyniosło najwyższą dokładność (94.6%). Wersja zoptymalizowana korzystała z głosowania miękkiego, co oznaczało, że klasyfikacja opierała się na prawdopodobieństwach przypisania danej klasy przez każdy model. Dodatkowo, przypisanie większych wag Random Forest i Gradient Boosting umożliwiło bardziej precyzyjną klasyfikację.

Optymalizacja modeli zwiększyła ich skuteczność w różnym stopniu, pozwalając na lepsze dopasowanie do danych i poprawę klasyfikacji. Wyniki pokazały, że różne modele reagują odmiennie na zmiany hiperparametrów, co miało istotny wpływ na ich końcową jakość predykcji.

- **Najlepszy model**: Voting Classifier osiągnął najwyższą dokładność oraz równowagę między precyzją i czułością, co wskazuje na jego stabilność i efektywność.
- **Najgorszy model**: Drzewo decyzyjne, pomimo stosunkowo wysokiej dokładności, nie wykazało znaczących korzyści z optymalizacji parametrów.

Projekt wykazał, że klasyczne modele uczenia maszynowego, odpowiednio dostosowane i zoptymalizowane, mogą skutecznie klasyfikować kluby piłkarskie na podstawie ich cech. Najlepszy wynik osiągnął Voting Classifier, co sugeruje, że modele zespołowe, łączące zalety poszczególnych metod klasyfikacyjnych, mogą być szczególnie efektywne w tego typu zadaniach. Wyniki projektu pokazują także znaczenie optymalizacji parametrów i przygotowania danych, takich jak standaryzacja oraz balansowanie klas, które mają kluczowy wpływ na jakość i stabilność predykcji. Projekt wykazał, że klasyczne modele klasyfikacyjne mogą stanowić bardzo skuteczne narzędzie wspierające decyzje świata piłki nożnej. Dzięki nim można szybko ocenić wartość sportową i potencjał drużyn piłkarskich, co jest przydatne zarówno w kontekście rozwoju klubu, jak i jego strategii komercyjnych.

6. Bibliografia

Dane: https://www.kaggle.com/datasets/ernestonlm/clubs-and-national-teams-stats-from-fifa-16-to-23

https://www.kdnuggets.com/2022/02/random-forest-decision-tree-key-differences.html

https://www.gov.pl/web/popcwsparcie/podzial-modeli-uczenia-maszynowego-wraz-z-przykladami-zastosowania

youtube: Support Vector Machines Part 1**,** Voting Classifier(Hard Voting and Soft Voting Classifier)