Automatyczne Uczenie Maszynowe

Tunability

Badanie tunowalności hiperparametrów algorytmów uczenia maszynowego

15.11.2024 r.

Natalia Choszczyk | Filip Langiewicz

Spis treści:

1. Wstęp

W ramach projektu “Tunability” zbadaliśmy tunowalność hiperparametrów trzech modeli uczenia maszynowego wykorzystanych w naszym przypadku do klasyfikacji binarnej. Wzięliśmy pod uwagę modele: DecisionTreeClassifier, KNeighborsClassifier z pakietu scikit-learn oraz XGBoost z pakietu xgboost. Do badania wykorzystaliśmy 4 różnorodne zbiory danych. Porównane zostały również dwie metody losowania punktów: Random Search (RandomizedSearchCV z pakietu scikit-learn) oraz Bayes Search [2] (BayesSearchCV z pakietu scikit-optimize). Wzorowaliśmy się na artykule: ***“Tunability: Importance of Hyperparameters of Machine Learning Algorithms”*** [1].

2. Zbiory Danych

Do wykonania eksperymentu użyliśmy czterech zbiorów danych opisanych poniżej.

Credit Approval (credit)

Rozmiar zbioru: **690 wierszy**, **16 kolumn**.
Zbiór dotyczący kart kredytowych. Wszystkie dane zostały zakodowane w celu zachowania poufności. Zmienne są zarówno numeryczne, jak i kategoryczne. Zbiór zawiera brakujące wartości, które uzupełniliśmy średnią w przypadku danych numerycznych i wartością najczęstszą w przypadku danych kategorycznych.

Go To College (college)

Rozmiar zbioru: **1000 wierszy**, **11 kolumn**.
Syntetyczny zbiór. W tym przypadku przewidujemy, czy uczniowie pójdą na studia. Zbiór zawiera głównie zmienne kategoryczne, ale również numeryczne oraz binarne. Nie ma braków danych.

Pima Indians Diabetes Database (diabetes)

Rozmiar zbioru: **768 wierszy**, **9 kolumn**.
Dane dotyczą indiańskiego plemienia Pima oraz ich zachorowań na cukrzycę. Na podstawie danych medycznych przewidujemy, czy dana osoba choruje na cukrzycę. Wszystkie dane są numeryczne. Zbiór zawiera braki danych w postaci wartości 0 w zmiennych, gdzie nie jest to możliwe (np. ciśnienie krwi).

Palmer Penguins (penguins)

Rozmiar zbioru: **274 wiersze**, **7 kolumn**.
Dane dotyczą obserwacji dwóch gatunków pingwinów. Jedna zmienna jest kategoryczna, a reszta to zmienne numeryczne. Nie występują braki danych.

Podsumowanie zbiorów danych

Zbiór danych	Liczba obserwacji	Liczba kolumn
credit	690	16
college	1000	11
diabetes	768	9
penguins	274	7

Tabela 1: Podsumowanie zbiorów danych wykorzystanych w eksperymencie

Analiza oraz preprocessing danych

Przed wykonaniem eksperymentu przeprowadziliśmy eksploracyjną analizę danych każdego zbioru. Przyjrzeliśmy się zmiennym, ich rozkładom oraz zależnościom między nimi. Następnie wykonaliśmy preprocessing, w którym zmapowaliśmy zmienną celu w każdym zbiorze na 0 i 1 oraz przygotowaliśmy pipeline służący do uzupełniania braków danych, skalowania oraz one-hot encodowania zmiennych.

3. Modele

Podczas eksperymentu badamy tunowalność trzech modeli uczenia maszynowego:

DecisionTreeClassifier (scikit-learn)

Badane hiperparametry:

- **criterion:** funkcja mierząca jakość podziału węzłów drzewa
- **max_depth:** maksymalna głębokość drzewa
- **max_leaf_nodes:** maksymalna liczba liści

KNeighborsClassifier (scikit-learn)

Badane hiperparametry:

- **n_neighbors:** liczba sąsiadów uwzględnianych przy klasyfikacji nowego punktu
- **weights:** sposób ważenia głosów sąsiadów przy klasyfikacji
- **metric:** miara odległości używana do określenia "bliskości" sąsiadów

XGBoost (xgboost)

Badane hiperparametry:

- **booster:** typ boostera używanego do tworzenia modelu
- **learning_rate:** współczynnik uczenia
- **max_depth:** maksymalna głębokość drzewa decyzyjnego
- **lambda:** parametr regularyzacji L2

Zakresy badanych hiperparametrów

Do każdego zbioru danych wykorzystaliśmy tę samą siatkę parametrów zarówno przy losowaniu punktów metodą Random Search, jak i metodą Bayes Search. Wybrane przez nas zakresy wartości pochodzą z artykułu [1], dokumentacji modeli lub dostępnych w sieci źródeł wiedzy i są przedstawione w Tabeli 2.

Algorytm	Hiperparametr	Typ zmiennej	Liczba wartości	Zakres
Decision Tree Classifier	criterion	categorical	3	['gini', 'entropy', 'log_loss']
	max_depth	int	29	[1, 29]
	max_leaf_nodes	int	9	[10, 90]
KNeighbors Classifier	n_neighbors	int	30	[1, 30]
	metric	categorical	4	['euclidean', 'manhattan', 'chebyshev', 'minkowski']
	weights	categorical	2	['uniform', 'distance']
XGBoost	booster	categorical	3	['gbtree', 'gblinear', 'dart']
	learning_rate	numeric	-	[0.0001, 0.3]
	max_depth	int	8	[3, 10]
	lambda	numeric	-	$[2^{-10}, 2^{10}]$

Tabela 2: Wybrane przez nas zakresy hiperparametrów

4. Eksperyment

Wykonaliśmy optymalizację hiperparametrów dla każdego z trzech modeli i czterech zbiorów danych oraz dla dwóch metod samplingu: **Random Search** oraz **Bayes Search**. W ten sposób uzyskaliśmy 24 przypadki (dla poszczególnych modeli, danych oraz metod samplingu), za pomocą których badamy tunowalność modeli. Za miarę jakości wybraliśmy **f1-score**, a w każdym przypadku wykonaliśmy po **200 iteracji**.

5. Wyniki

5.1 Wyniki f1 - dopasowania modelu

Random Search

Pierwszą zastosowaną metodą losowania punktów był Random Search. Najlepsze wyniki f1, jakie uzyskały modele na naszych zbiorach danych po tej optymalizacji hiperparametrów są przedstawione w Tabeli 3.

zbiór danych \ model	DecisionTree	KNeighbors	XGBoost
credit	0.86	0.85	0.86
college	0.86	0.85	0.90
diabetes	0.63	0.66	0.65
penguins	1.00	1.00	0.99

Tabela 3: Wyniki f1 dla metody Random Search

Bayes Search

Następnie zastosowaliśmy metodę losowania punktów opartą na algorytmie Bayes Search. Wyniki dopasowania modelu ukazane są w Tabeli 4.

zbiór danych \ model	DecisionTree	KNeighbors	XGBoost
credit	0.86	0.84	0.86
college	0.86	0.85	0.90
diabetes	0.65	0.66	0.66
penguins	1.00	1.00	0.99

Tabela 4: Wyniki f1 dla metody Bayes Search

5.2 Uzyskane wyniki tunowalności algorytmów

Tunowalność algorytmów dla poszczególnych zbiorów danych mierzona za pomocą definicji zawartej w [1] przedstawiona jest w Tabeli 5.

Algorytm	Zbiór danych	Tunowalność dla Random Search	Tunowalność dla Bayes Search
DecisionTree	credit	0.0254	0.0254
	college	0.0073	0.0091
	diabetes	0.0000	0.0140
	penguins	0.0000	0.0034
KNeighbors	credit	0.0064	0.0031
	college	0.0262	0.0262
	diabetes	0.0000	0.0000
	penguins	0.0069	0.0069
XGBoost	credit	0.0134	0.0134
	college	0.0000	0.0074
	diabetes	0.0000	0.0115
	penguins	0.0000	0.0101

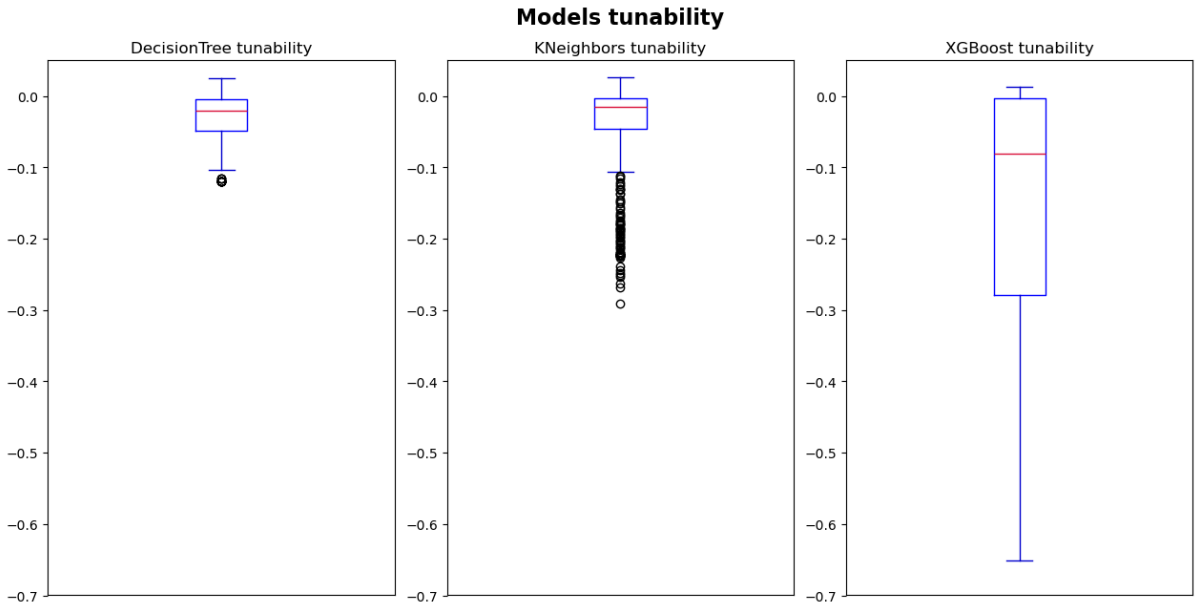
Tabela 5: Tunowalność modeli na poszczególnych zbiorach

Stąd możemy wyliczyć tunowalność modeli, które braliśmy pod uwagę. Wyniki te przedstawia Tabela 6.

metoda samplingu \ algorytm	DecisionTree	KNeighbors	XGBoost
Random Search	0.0082	0.0099	**0.0033**
Bayes Search	**0.0130**	0.0091	0.0106

Tabela 6: Wielkość tunowalności algorytmów w zależności od metody losowania punktów

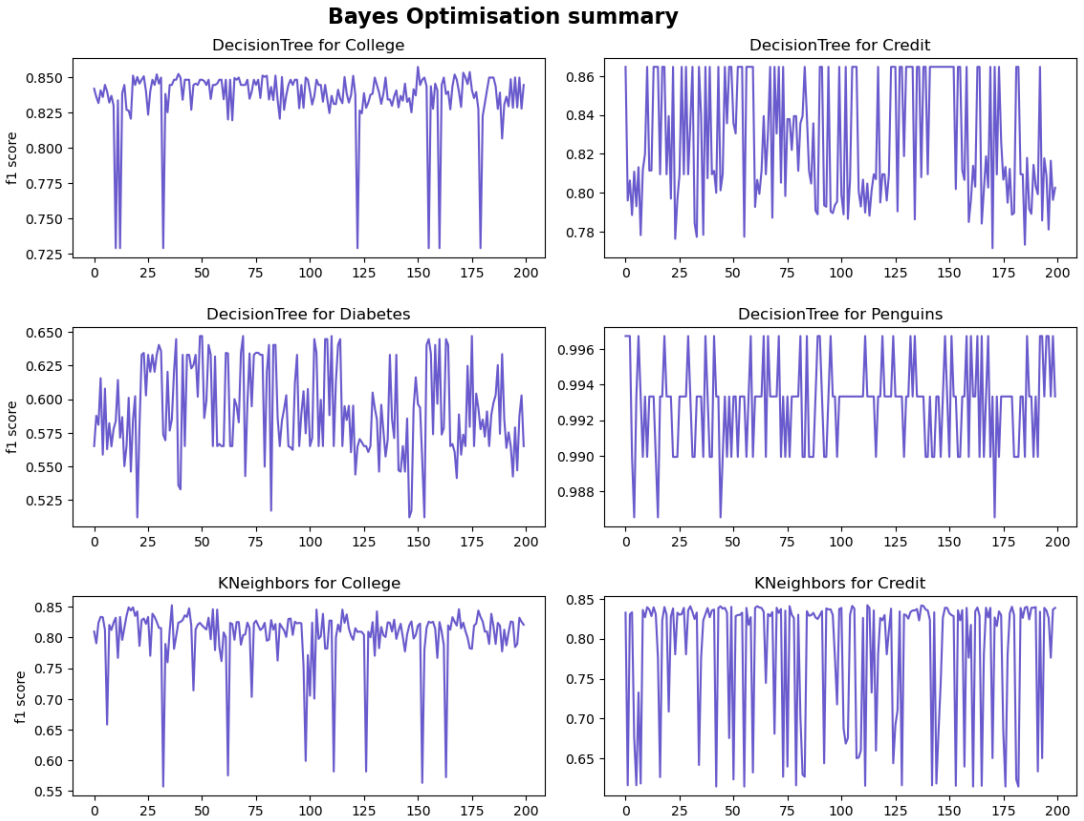
Na Wykresie 1 możemy porównać jak rozkłada się tunowalność w poszczególnych algorytmach uczenia maszynowego bez podziału na zbiory danych.

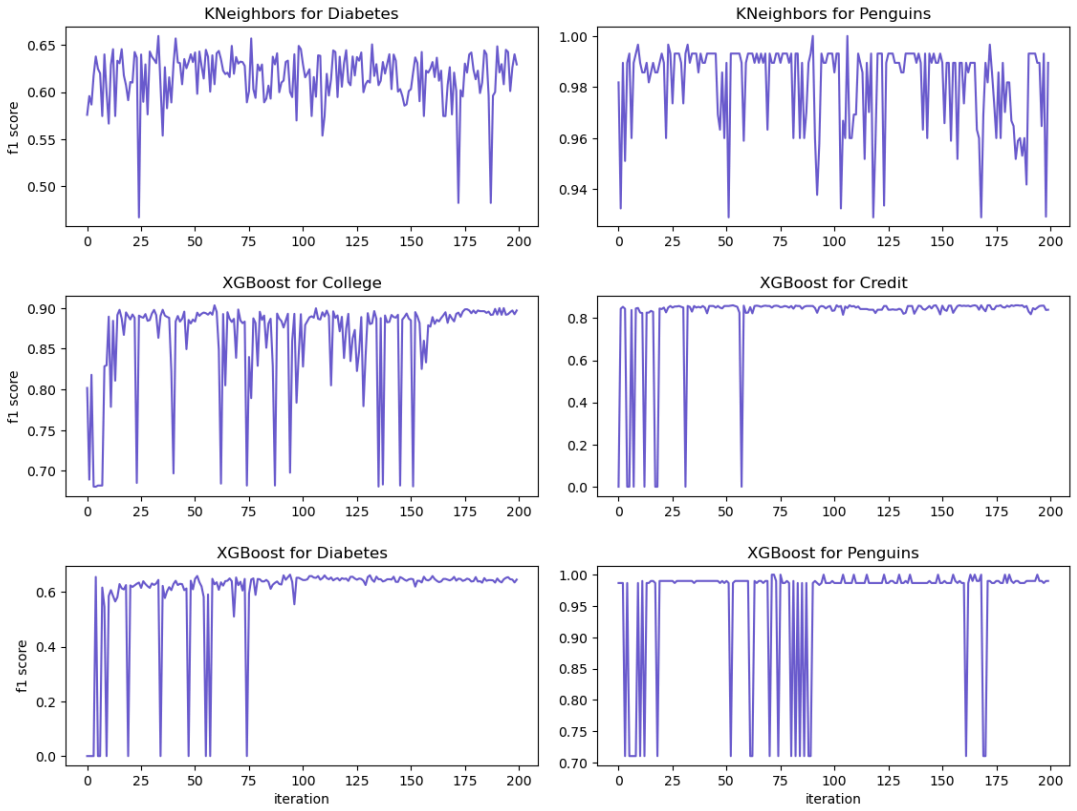


Wykres 1: Porównanie tunowalności algorytmów

5.3 Liczba iteracji potrzebnych do uzyskania stabilnych wyników

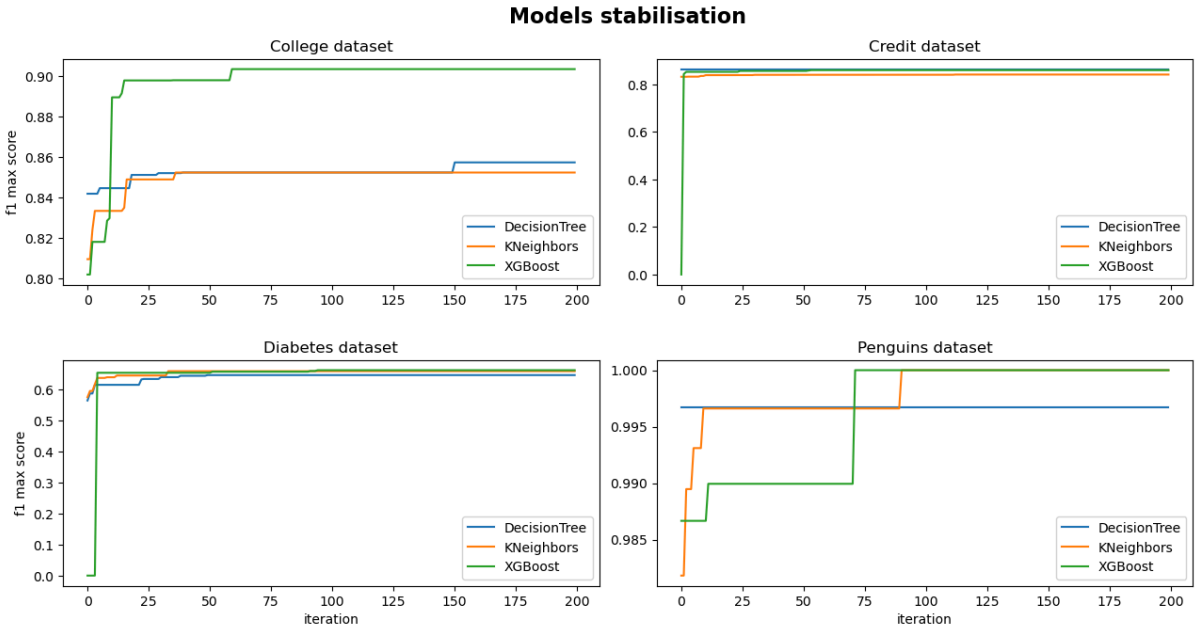
Na Wykresie 2 możemy zobaczyć, jak zmieniała się wartość dopasowania modelu f1 w zależności od liczby iteracji dla metody Bayes Search.





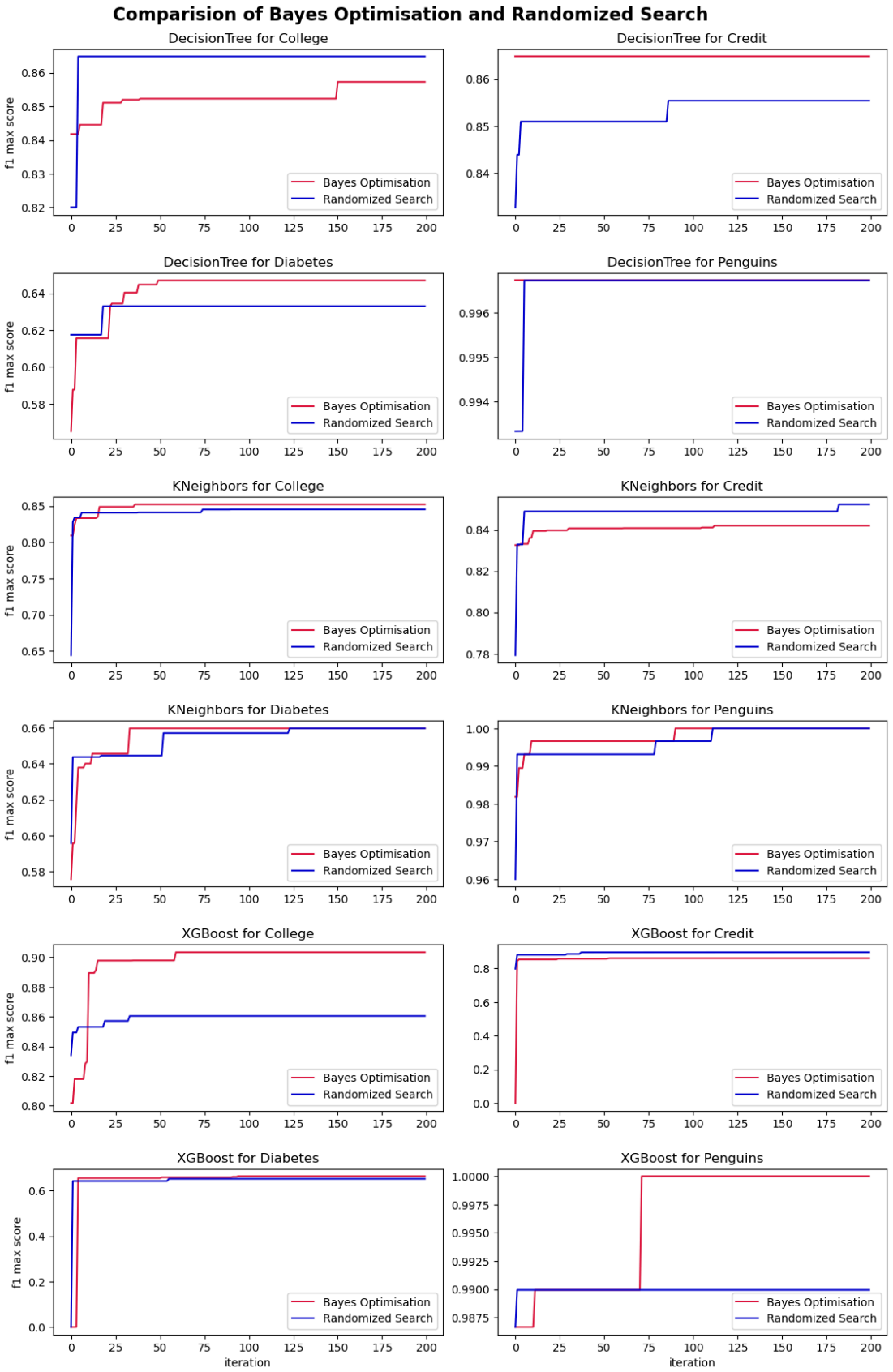
Wykres 2: Zbieżność metody Bayes Search

Wykres 3 przedstawia porównanie stabilizacji poszczególnych modeli - ile iteracji jest potrzebnych by uzyskać stabilne wyniki optymalizacji dla metody Bayes Search.



Wykres 3: Porównanie stabilności modeli dla metody Bayes Search

Na Wykresie 4 widzimy porównanie zbieżności zastosowanych przez nas metod losowania punktów.



Wykres 4: Porównanie liczby iteracji potrzebnych do uzyskania stabilnych wyników optymalizacji

6. Wnioski

Wyniki naszego eksperymentu potwierdzają, że zarówno wybór modelu, jak i metoda optymalizacji hiperparametrów mają istotny wpływ na skuteczność klasyfikacji oraz tunowalność algorytmu.

W analizowanych przypadkach:

1. **Porównanie modeli**: XGBoost osiągnął najwyższe wyniki f1 na większości zbiorów danych, szczególnie dobrze radząc sobie z bardziej złożonymi zbiorami, takimi jak „college" i „credit". Model ten wykazał się również wyższą tunowalnością w przypadku metody Bayes Search.

2. **Tunowalność algorytmów**: Modele różnią się tunowalnością zależnie od zbioru danych i metody samplingu. W przypadku Bayes Search, DecisionTree i XGBoost wykazały wyższą tunowalność niż KNeighbors, co sugeruje, że są bardziej podatne na optymalizację poprzez dostosowanie hiperparametrów.

3. **Porównanie metod optymalizacji**: Bayes Search zapewnił wyższą stabilność wyników i lepszą konwergencję dla większości algorytmów, co sugeruje jego przewagę nad Random Search, zwłaszcza przy ograniczonej liczbie iteracji. Wyniki jednoznacznie wskazują, że użycie różnych metod losowania punktów może mieć istotny wpływ na badaną tunowalność modelu. Zwłaszcza dla modeli DecisionTree i XGBoost uzyskana wartość tunowalności różni się znacząco.

4. **Stabilizacja wyników**: Stosując metodę samplingu Bayes Search w większości przypadków uzyskaliśmy lepsze wyniki w mniejszej liczbie iteracji.

Ogólnie rzecz biorąc, nasze badanie wykazało, że optymalizacja hiperparametrów z użyciem Bayes Search pozwala osiągnąć lepszą tunowalność i wyższą skuteczność modeli, szczególnie w przypadku bardziej złożonych algorytmów jak XGBoost.

7. Bibliografia

[1] Philipp Probst, Anne-Laure Boulesteix, Bernd Bischl. Tunability: Importance of Hyperparameters of Machine Learning Algorithms.
[2] scikit-optimize.github.io/stable/auto_examples/sklearn-gridsearchcv-replacement.html