Analiza tunowalności hiperparametrów wybranych algorytmów uczenia maszynowego w kontekście różnych metod optymalizacji

Mikołaj Ozimek, Julia Dudzińska, Weronika Gozdera

18 listopada 2025

1 Wstęp

Głównym celem projektu jest analiza tunowalności hiperparametrów wybranych algorytmów uczenia maszynowego w kontekście różnych metod optymalizacji (losowania punktów). Algorytmy uczenia maszynowego poddane przez nas analizie to Random Forest, XGBoost, Decision Tree i KNN. Porównaliśmy wyniki uzyskane podczas optymalizacji wykorzystującej punkty losowane przy użyciu metod Grid Search, Random Search i Bayesian TPE. Do optymalizacji została wykorzystana biblioteka Optuna. Proces uczenia przebiegał zbiorach danych z platformy OpenML przedstawionych w Tabeli 1.

Nazwa	**Liczba obserwacji**	**Liczba zm. objaśniających**
Adult [1]	48842	14
Diagnostic [3]	569	30
Bank Marketing [2]	45211	16
Mushroom [4]	8124	22
Titanic [5]	1309	13

Tabela 1: Wykorzystane zbiory danych

Rozważane podczas optymalizacji zakresy hiperparametrów zostały określone na podstawie wyników otrzymanych w [7] i przedstawione w Tabeli 2. Zakresy zostały dostosowane do oczekiwanego typu danych (np. modyfikacja z liczb niecałkowitych na całkowite).

Model	**Hiperparametr**	**Typ danych**	**Zakres / Wybór**
Decision Tree (rpart)	ccp_alpha	float	[0.0, 0.008]
	max_depth	int	[12, 27]
	min_samples_leaf	int	[4, 42]
	min_samples_split	int	[5, 49]
Random Forest (ranger)	n_estimators	int	[206, 1740]
	bootstrap	categorical	{True, False}
	max_features	float	[0.323, 0.974]
	min_samples_split	float	[0.007, 0.513]
KNN	n_neighbors	int	[10, 30]
XGBoost	n_estimators	int	[921, 4551]
	eta	float	[0.002, 0.355]
	subsample	float	[0.545, 0.958]
	max_depth	int	[6, 14]
	colsample_bytree	float	[0.419, 0.864]

Tabela 2: Zastosowane zakresy hiperparametrów

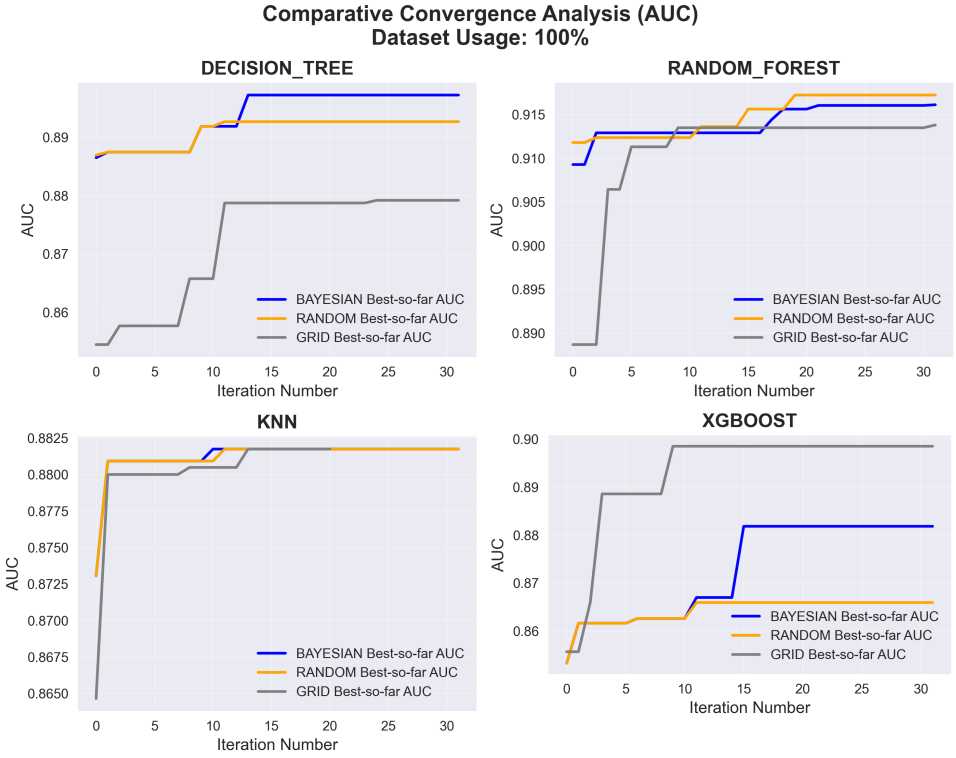
2 Optymalizacja hiperparametrów

W procesie optymalizacji dla każdej z metod przeprowadzono 32 próby. Liczba ta została podyktowana przede wszystkim możliwościami sprzętowymi, jako że liczba wywołań została dodatkowo zwielokrotniona przez rozważanie różnych procentowych części zbiorów danych. Wybór rozważanych hiperparametrów wykonywany był raz, przed optymalizacją na wszystkich zbiorach danych, aby zapewnić rozważanie tych samych wartości dla wszystkich zbiorów. W przypadku optymalizacji wykorzystującej Grid Search, przestrzeń poszukiwań była dyskretyzowana tak, aby liczba punktów do testowania była zbliżona do liczby iteracji dla pozostałych

metod (podział wartości dla każdego parametru opierał się na pierwiastku stopnia wymiaru przestrzeni hiperparametrów z liczby prób). W każdej iteracji oceniano modele na podstawie 5-krotnej walidacji krzyżowej, rozważając zarówno AUC jak i accuracy (jednak opierając ostateczne wyniki na podstawie AUC – odnosząc się w dalszej części do *tunowalności*, mamy na myśli *tunowalność AUC*). Cały proces optymalizacji przebiegał w zdefiniowanym zakresie wykorzystania danych: 10%, 25%, 50%, 75%, 100%. Aż do Sekcji 2.3 rozważany jest pełen zakres (100%) zbiorów danych.

2.1 Stabilizacja optymalizacji

Rysunek 1 przedstawia zmianę najlepszego (względem AUC) otrzymanego dotychczas wyniku (średniego dla wszystkich zbiorów danych) wraz z kolejnymi iteracjami optymalizacji. Poza wykresem dla metody Random Forest (prawy górny róg), wszystkie wyniki ustabilizowały się od około 15 iteracji. Dla Random Forest stabilizacja następuje dopiero około 20-25 iteracji, z wyjątkiem metody Grid Search, która poprawia swój wynik w ostatniej iteracji (niestety z wyżej wspomnianych powodów nie byliśmy w stanie zbadać większej liczby iteracji), jest to jednak niewielka zmiana. Bazując na otrzymanych wynikach, najmniejsza liczba iteracji pozwalająca uzyskać stabilne wyniki optymalizacji to 25.

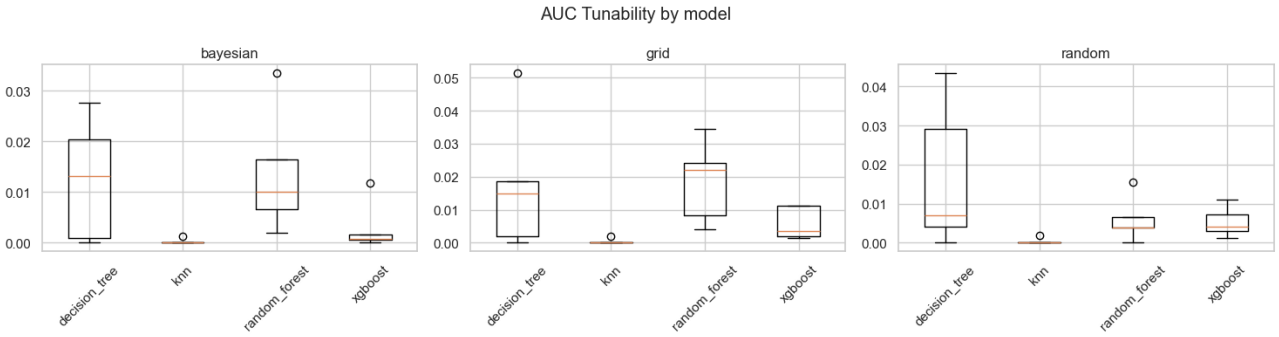


Rysunek 1: Najlepszy (względem AUC) dotychczasowy wynik na przestrzeni iteracji procesu optymalizacji, dla poszczególnych modeli i metod samplingu

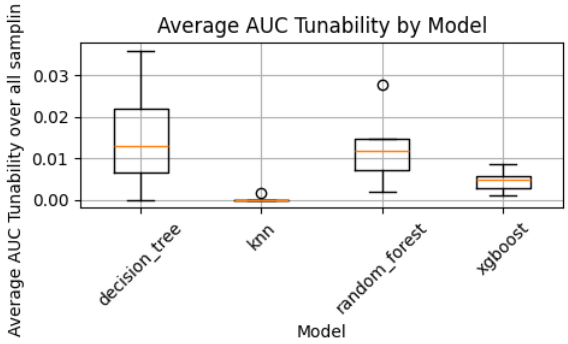
2.2 Tunowalność poszczególnych algorytmów

Tunowalność poszczególnych algorytmów z podziałem na różne metody samplingu została przedstawiona na Rysunku 2, natomiast ogólna tunowalność (na podstawie średnich wyników metod samplingu) na Rysunku 3. Zarówno dane na Rysunku 2 jak i w Tabeli 3b wskazują na to, że najbardziej tunowalne okazały się algorytmy Decision Tree oraz Random Forest, natomiast najmniej – KNN. Ciężko jednak wyciągnąć jednoznaczne wnioski z otrzymanych wyników, jako

że do dyspozycji mamy tylko 5 wartości (tyle, ile zbiorów danych) – wskazują na to również duże odchylenia standardowe przedstawione w Tabel 3b.



Rysunek 2: Tunowalność AUC dla poszczególnych metod samplingu



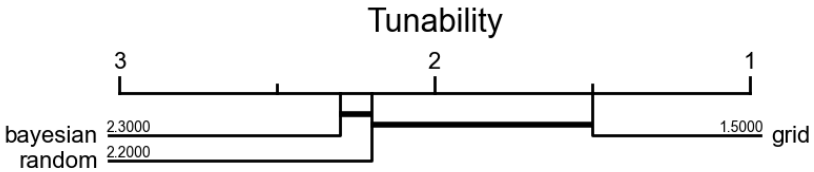
(a) Wyniki w formie wykresu

Model	Mean	Std
decision_tree	0.015494	0.014039
knn	0.000328	0.000734
random_forest	0.012784	0.009724
xgboost	0.004713	0.002854

(b) Wyniiki w formie tabeli

Rysunek 3: Średnia tunowalność AUC dla metod samplingu (każda wartość to średnia z wartości otrzymanych dla 3 metod)

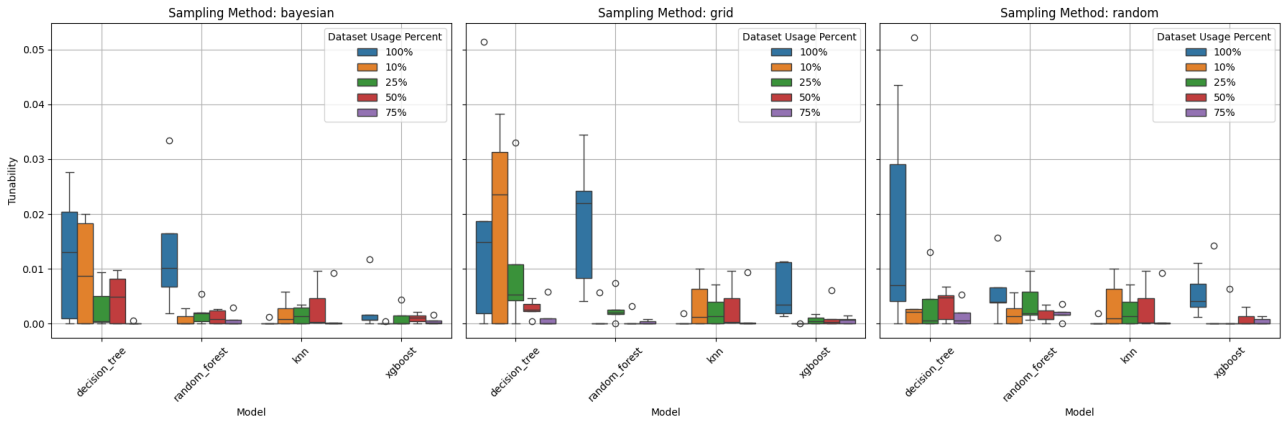
W celu analizy wpływu techniki losowania punktów na tunowalność algorytmów zastosowaliśmy diagram różnic krytycznych [6] (ang. *Critical Difference Diagram*), przedstawiony na Rysunku 4. Możemy zauważyć, że średnie rangi badanych technik różnią się, jednak różnice te nie są statystycznie istotne, na co wskazują poziome linie łączące pary metod. Wybór metody losowania nie wydaje się mieć więc znaczącego wpływu na tunowalność, co sugeruje, że nie wystąpił *bias sampling*. Ponownie należy wziąć jednak pod uwagę ograniczoną ilość danych oraz niewielką liczbę iteracji optymalizacji – można np. oczekiwać, że przy zwiększonej liczbie iteracji różnica pomiędzy techniką losową a beyesowską stałaby się bardziej znacząca (jako, że pierwsze 10 iteracji algorytmu TPE opiera się na zastosowaniu techniki losowej).



Rysunek 4: Diagram różnic krytycznych porównujący techniki losowania parametrów pod względem tunowalności

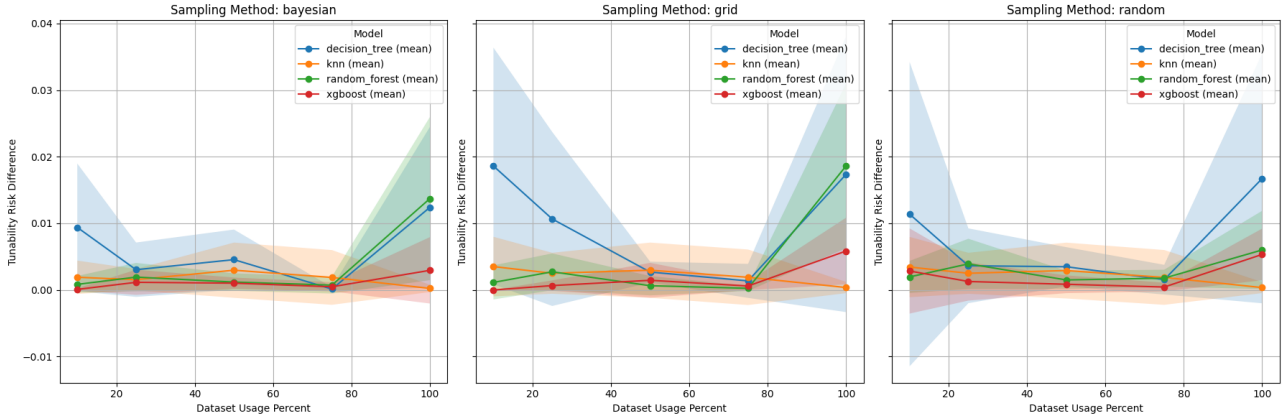
2.3 Wpływ wielkości zbioru danych na tunowalność

Aby przeprowadzić analizę wpływu rozmiaru zbioru danych na wyniki, rozważyliśmy podzbiory o rozmiarze 10%, 25%, 50%, 75%, 100%, przeprowadzając analogiczny proces optymalizacji dla każdego rozmiaru podzbioru. Otrzymana tunowalność została przedstawiona na Rysunku 5. Ciężko nam zinterpretować otrzymane wyniki – jedynie dla Decision Tree można zauważyć, że 100% zbioru danych osiąga zawsze najwyższą lub drugą najwyższą tunowalność. Nieregularne wyniki są prawdopodobnie związane z małą próbką danych – podobnie, jak podczas ogólnej analizy tunowalności, każdy wykres pudełkowy jest tworzony na podstawie tylko 5 wartości.



Rysunek 5: Tunowalność dla różnych wielkości zbiorów danych – wykresy pudełkowe z podziałem na metody samplingu

Aby sprawdzić, czy występuje monotoniczność względem procentowej wielkości zbioru danych, przedstawiliśmy wyniki również na wykresach liniowych, widocznych na Rysunku 6. Jak widać, relacja nie jest monotoniczna dla żadnej metody samplingu – wydaje się, że tunowalność wzrasta dla mniejszych i większych wielkości, a dla środkowych pozostaje najmniejsza. Jedynie dla algorytmu KNN można zaobserwować powtarzającą się zależność – tunowalność maleje wraz ze wzrostem rozmiaru zbioru danych, dla 100% przyjmując najmiejszą, bliską zeru wartość. Jest to najprawdopodobniej spowodowane tym, że dla KNN jest optymalizowany tylko jeden parametr, więć przestrzeń poszukiwań jest znacznie mniejsza, niż w przypadku innych algorytmów. Być może pozostałe algorytmy pokazałyby analogiczną monotoniczność, gdyby maksymalna liczba iteracji byłaby znacznie większa, niż 32.



Rysunek 6: Tunowalność dla różnych wielkości zbiorów danych – wykresy liniowe (średnia i odchylenie standardowe) z podziałem na metody samplingu

Literatura

[1] Adult. https://www.openml.org/d/1590.

[2] Bank marketing. https://www.openml.org/d/1461.

[3] Breast cancer wisconsin (diagnostic). https://www.openml.org/d/1510.

[4] Mushroom. https://www.openml.org/d/24.

[5] Titanic. https://www.openml.org/d/40945.

[6] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, 2019.

[7] Philipp Probst, Bernd Bischl, and Anne-Laure Boulesteix. Tunability: Importance of hyperparameters of machine learning algorithms, 2018.

A Proces optymalizacji hiperparametrów - metody bayesowskie - algorytm TPE

Poniżej przedstawione zostały optymalne hiperparametry dla każdego algorytmu oraz zestawu danych.

Random forest:

	n_estimators	bootstrap	max_features	min_samples_split	AUC_d	AUC_o
Adult	1087	False	0.576	0.011	0.5	0.918
Diagnostic	809	True	0.475	0.059	0.5	0.993
Bank Marketing	951	False	0.501	0.010	0.5	0.837
Mushroom	977	True	0.586	0.028	0.5	1.0
Titanic	1470	False	0.830	0.194	0.5	0.992

Tabela 3: Optymalne hiperparametry oraz AUC.

Decision Tree:

	ccp_alpha	max_depth	min_samples_leaf	min_samples_split	AUC_d	AUC_o
Adult	4.602e-05	13	30	34	0.850	0.909
Diagnostic	1.461e-04	18	24	18	0.967	0.978
Bank Marketing	1.384e-04	16	42	16	0.672	0.898
Mushroom	1.528e-05	18	4	33	0.986	0.999
Titanic	1.318e-04	13	16	48	0.965	0.992

Tabela 4: Optymalne hiperparametry oraz AUC.

XGBoost:

	n_estimators	eta	subsample	max_depth	colsample_bytree	AUC_d	AUC_o
Adult	2492	0.0054	0.8418	7	0.4242	0.919	0.933
Diagnostic	1718	0.1640	0.6335	9	0.6705	0.995	0.996
Bank Marketing	1963	0.0040	0.9006	9	0.6551	0.920	0.935
Mushroom	4036	0.0032	0.7862	8	0.5165	1.0	1.0
Titanic	1461	0.0021	0.7028	8	0.5565	0.992	0.994

Tabela 5: Optymalne hiperparametry oraz AUC.

KNN:

	n_neighbors	AUC_d	AUC_o
Adult	30	0.863	0.887
Diagnostic	18	0.987	0.993
Bank Marketing	25	0.765	0.790
Mushroom	10	1.0	1.0
Titanic	29	0.942	0.945

Tabela 6: Optymalne hiperparametry oraz AUC.