Automatyczne Uczenie Maszynowe, Praca domowa nr 1

Aleks Kapich, Michał Binda, Maciej Borkowski

15.11.2024

Spis treści

1 Wstęp

Projekt miał na celu zbadanie tunowalności [1] trzech algorytmów uczenia maszynowego:

- Decision Tree
- K Nearest Neighbours
- Random Forest

W ramach analizy wykorzystane zostały cztery zbiory danych dostępne na platformie OpenML o następujących ID:

- **37 diabetes** - zbiór dotyczy objawów cukrzycy u pacjentów, zmienna celu to obecność objawów
- **1462 banknote-authentication** - zbiór dotyczący autentyczności banknotów, zmienna celu opisuje, czy banknot jest falsyfikatem
- **1464 blood-transfusion-service-center** - zmienna celu tego zbioru determinuje, czy badany był dawcą krwi
- **1489 phoneme** - celem tego zestawu danych jest rozróżnienie dźwięków nosowych i ustnych

Każdy ze zbiorów danych dotyczył problemów klasyfikacji binarnej.

2 Schemat wykonania

Każdy ze zbiorów danych został poddany preprocessingowi wykonanemu z użyciem *pipeline* - braki w zmiennych numerycznych były zastępowane średnią, zmienne zaś normalizowane. Dla zmiennych kategorycznych jako uzupełnienie braków posłużyła moda oraz zastosowany został One Hot Encoding.

Następnie, pipeline'y zostały wykorzystane w procesie poszukiwania optymalnych hiperparametrów modeli uczenia maszynowego. Optymalizacja miała miejsce z użyciem metod *RandomizedSearchCV* oraz *BayesSearchCV* odpowiednio z bibliotek *sklearn* oraz *skopt*.

Dla każdego zbioru danych oraz algorytmu uczenia maszynowego wyżej wymienione metody sprawdzały 50 konfiguracji hiperparametrów. Wybrana przez nas miara jakości modeli to ROC AUC przy pięciokrotnej kroswalidacji. Przy liczeniu tunowalności jako baseline wybrane zostały modele z biblioteki *sklearn* o domyślnych wartościach hiperparametrów. Średnia tunowalność poszczególnych algorytmów dla danego zbioru danych wyznaczona została jako różnica pomiędzy średnią wartością ROC AUC dla 50 modeli przetestowanych w procesie optymalizacji parametrów oraz wartością ROC AUC dla modelu stanowiącego baseline.

Przestrzeń hiperparametrów wybrana została na podstawie parametrów użytych w artykule [1], dokumentacji biblioteki sklearn oraz własnego doświadczenia. Tabela 1 przedstawia siatkę użytą w eksperymencie.

3 Wyniki i wnioski

3.1 Decision Tree

Modele wykorzystujące hiperparametry otrzymywane metodą optymalizacji bayesowskiej otrzymywały średnio lepsze wyniki niż modele otrzymane przy *Random Search*. W istocie, przy metodzie *Random Search* mieliśmy często do czynienia z modelami o widocznie niższym ROC AUC. Rysunki 1 przedstawiają wykresy violin plot wartości ROC AUC dla różnych konfiguracji hiperparametrów z podziałem na zbiory danych.

Wykresy na rysunku 2 przedstawiają liczbę iteracji każdej metody wymaganej do uzyskania stabilnych wyników optymalizacji. Optymalizacja Bayesowska dla każdego z czterech zbiorów danych okazała się być lepsza od optymalizacji metodą *RandomSearch*. Zestawy hiperparametrów otrzymane tą metodą gwarantowały wyższe wyniki ROC AUC modeli, ponadto optymalizacja stabilizowała się szybciej. Liczba iteracji potrzebna do stabilizacji różniła się pomiędzy zbiorami danych. Dla zbiorów 37 i 1462 wyniki ustabilizowały się przed upływem 10 iteracji, by po upływie kolejnych ok. 30 iteracji ulec ostatniej poprawie. W wypadku zbiorów 1464 oraz 1489 potrzebne było zaledwie 10-20 iteracji do otrzymania całkowitej stabilizcji.

Rysunek 3 przedstawia wykres średniej tunowalności na wszystkich zbiorach danych. Znaczącą poprawę widzimy dla zbioru 37 i porównując ją z innymi algorytmami jest najbardziej istotna. Użycie metody Bayes Search pozwala na lepszy tuning modelu drzewa decyzyjnego. Z wyjątkiem zbioru danych 1462, optymalizacja hiperparametrów przynosiła zauważalną poprawę jakości modelu.

3.2 K Nearest Neighbours

W przypadku algorytmu KNN modele wykorzystujące hiperparametry otrzymywane metodą optymalizacji bayesowskiej uzyskały zbliżone wartości ROC AUC, jak modele otrzymane przy użyciu random search. Wykresy violin plot przedstawia rysunek 4.

Jak widzimy na rysunku 5, optymalizacja bayesowska w porównaniu do optymalizacji metodą Random Search wypadła bardzo podobnie dla zbiorów 1462 oraz 1489, a stabilizacja została osiągnięta po niewielkiej liczbie iteracji. Istotną różnicę widać jedynie dla pierwszych iteracji. Dla zbioru 1464 lepsza okazała się być optymalizacja metodą Random Search, a stabilizacja została osiągnięta już po

ok. 15 iteracjach. W przypadku zbioru 37 optymalizacja bayesowska szybciej osiągnęła stabilność, a ostateczny wynik jest zbliżony do osiągniętego metodą Random Search.

Tunowalność dla tego algorytmu jest mało znacząca, w szczególności dla zbiorów danych 1462 i 1489, gdzie praktycznie nie widać poprawy względem modelu baseline. Dla pozostałych dwóch zbiorów danych średnie wzrosty ROC AUC oscyluja dookoła wartości 0.03. Ponownie lepiej wypada metoda Bayes Search i staje się to regułą w naszych eksperymentach.

3.3 Random Forest

Modele wykorzystujące hiperparametry otrzymywane metodą optymalizacji bayesowskiej otrzymywały średnio porównywalne wyniki 7 niż modele otrzymane przy *Random Search*, przy czym ich odchylenie jest widocznie mniejsze.

W przypadku modelu lasu losowego widoczna jest szybsza stabilność optymalizacji metodą Bayes Search 8. Dla zbiorów 1462 i 1464 ostatecznie wyniki uzyskane obiema metodami są do siebie zbliżone, natomiast istotne różnice widać dla zbiorów 37, gdzie przeważa Random Search, oraz dla zbioru 1489, gdzie wygrywa Bayes Search.

Tunowalność w przypadku tego algorytmu jest bardzo mało znacząca, dla zbiorów 1462 i 1489 średnia tunowalność okazała się być mniejsza niż 0, tj. algorytmy średnio wypadały gorzej niż baseline. Dla pozostałych dwóch zbiorów notowana była średnio minimalna poprawa ROC AUC.

4 Podsumowanie

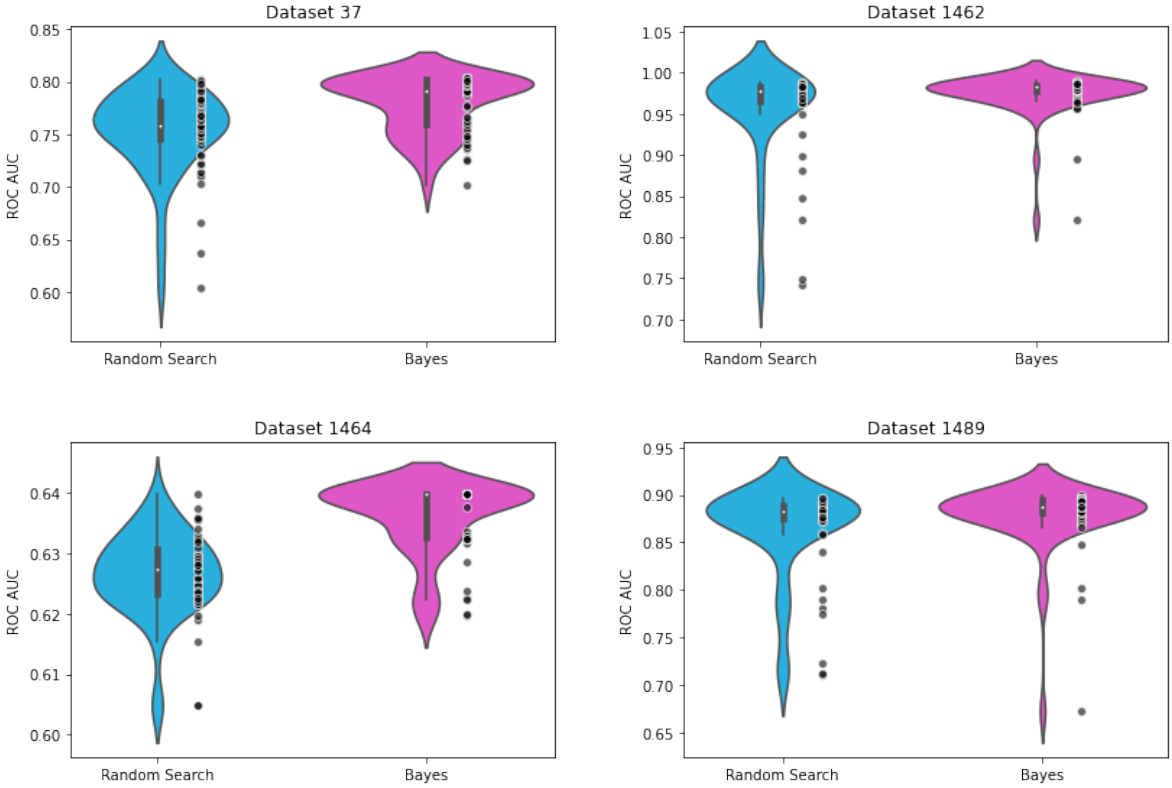
- **Decision Tree:**
  - Modele optymalizowane metodą Bayes Search osiągnęły lepsze wyniki ROC AUC w porównaniu z Random Search.
  - Stabilność wyników optymalizacji została szybciej osiągnięta przy Bayes Search, szczególnie dla mniejszych zbiorów danych.
  - Poprawa wyników względem domyślnych hiperparametrów była najbardziej istotna spośród wszystkich algorytmów.
- **K Nearest Neighbours:**
  - Wyniki ROC AUC uzyskane obiema metodami optymalizacji były porównywalne, choć w niektórych przypadkach Bayes Search wykazał przewagę na wczesnym etapie iteracji.
  - Tunowalność KNN była niższa w porównaniu do Decision Tree.
- **Random Forest:**
  - Modele zoptymalizowane metodą Bayes Search wykazały mniejsze odchylenia wyników niż Random Search, przy porównywalnym poziomie ROC AUC.
  - Stabilność optymalizacji szybciej osiągnięto metodą Bayes Search, zwłaszcza dla większych zbiorów danych.
  - Niska tunowalność i najsłabsze wyniki w porównaniu do reszty algorytmów.

Z przeprowadzonych eksperymentów wynika, że algorytmy Decision Tree oraz KNN osiągają lepsze wyniki, gdy dla każdego zbioru korzystamy z odpowiednio dobranych hiperparametrów. Algorytmy te są bardziej tunowalne niż Random Forest. Największą poprawę możemy zaobserwować dla algorytmu Decision Tree. Bayes Search okazał się w większości przypadków bardziej efektywny w osiąganiu wyższej jakości modeli i stabilności wyników. Tunowalność okazała się być również w dużej mierze uzależniona od zbioru danych.

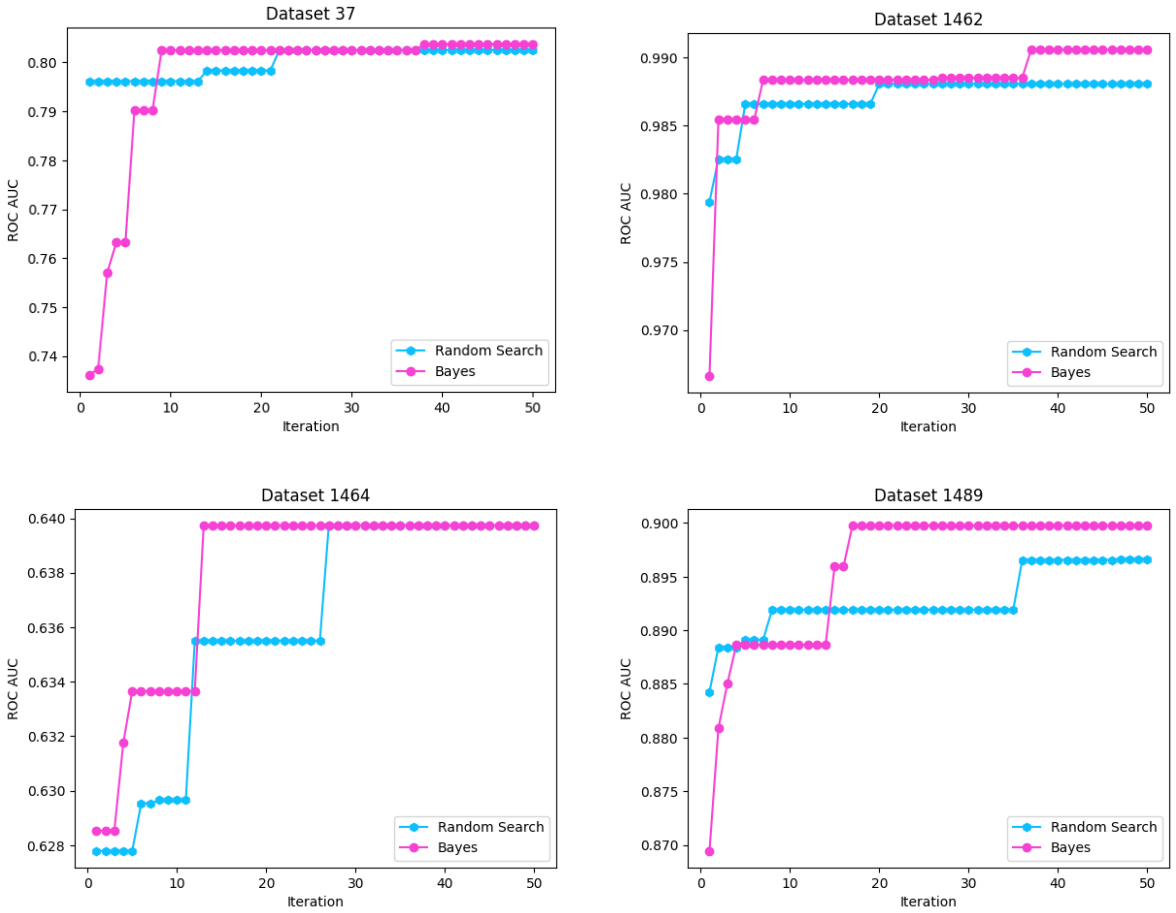
5 Załączniki

algorytm	**hiperparametr**	**zakres wartości**
Decision Tree	max_depth	[1, 30]
	min_samples_split	[2, 60]
	min_samples_leaf	[1, 60]
	max_features	{ None, log2, sqrt }
	criterion	{ gini, entropy }
K-Nearest Neighbors	n_neighbors	[1, 30]
	p	{ 1, 2 }
	weights	{ uniform, distance }
	algorithm	{ auto, ball_tree, kd_tree, brute }
Random Forest	n_estimators	{50, 100, ..., 450, 500}
	max_depth	[5, 30]
	min_samples_split	[2, 60]
	min_samples_leaf	[1, 60]
	max_features	{ None, log2, sqrt }
	criterion	{ gini, entropy }

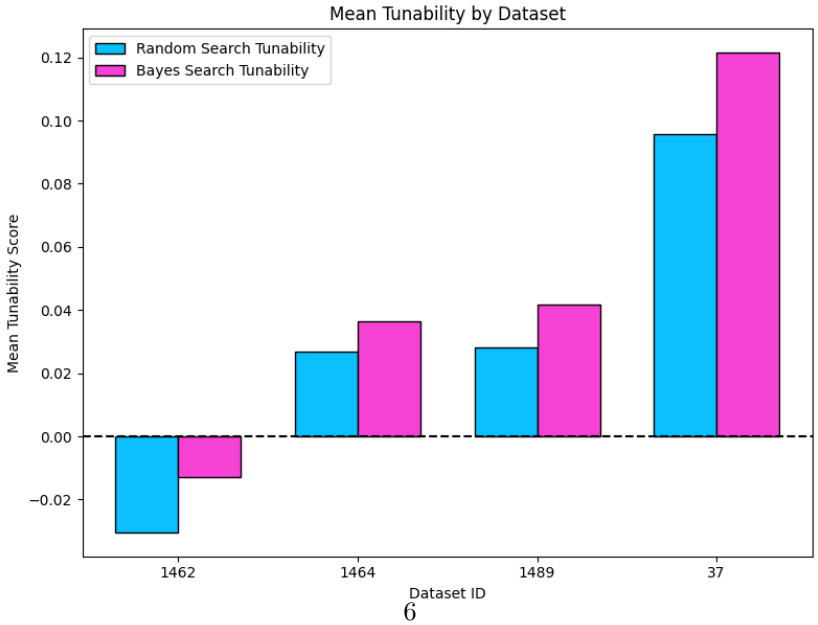
Tabela 1: Siatka hiperparametrów dla wybranych algorytmów.



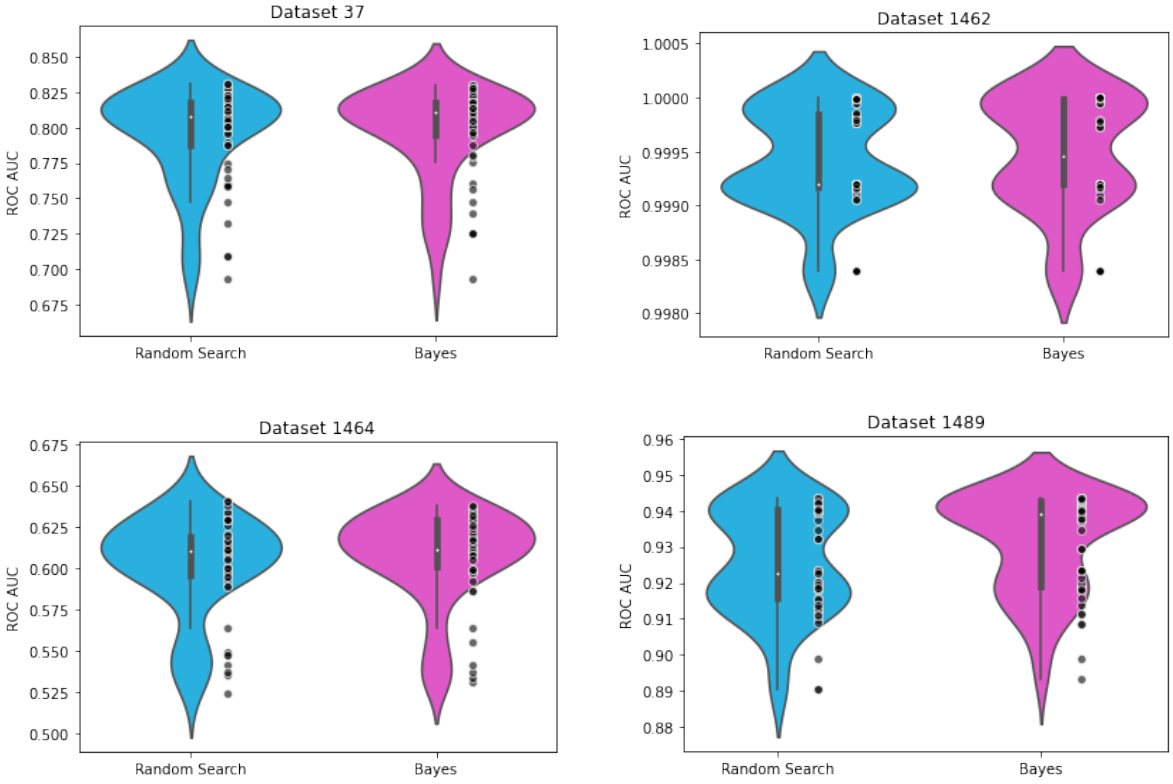
Rysunek 1: Wartości ROC AUC dla różnych konfiguracji hiperparametrów z podziałem na zbiory danych dla algorytmu Decision Tree



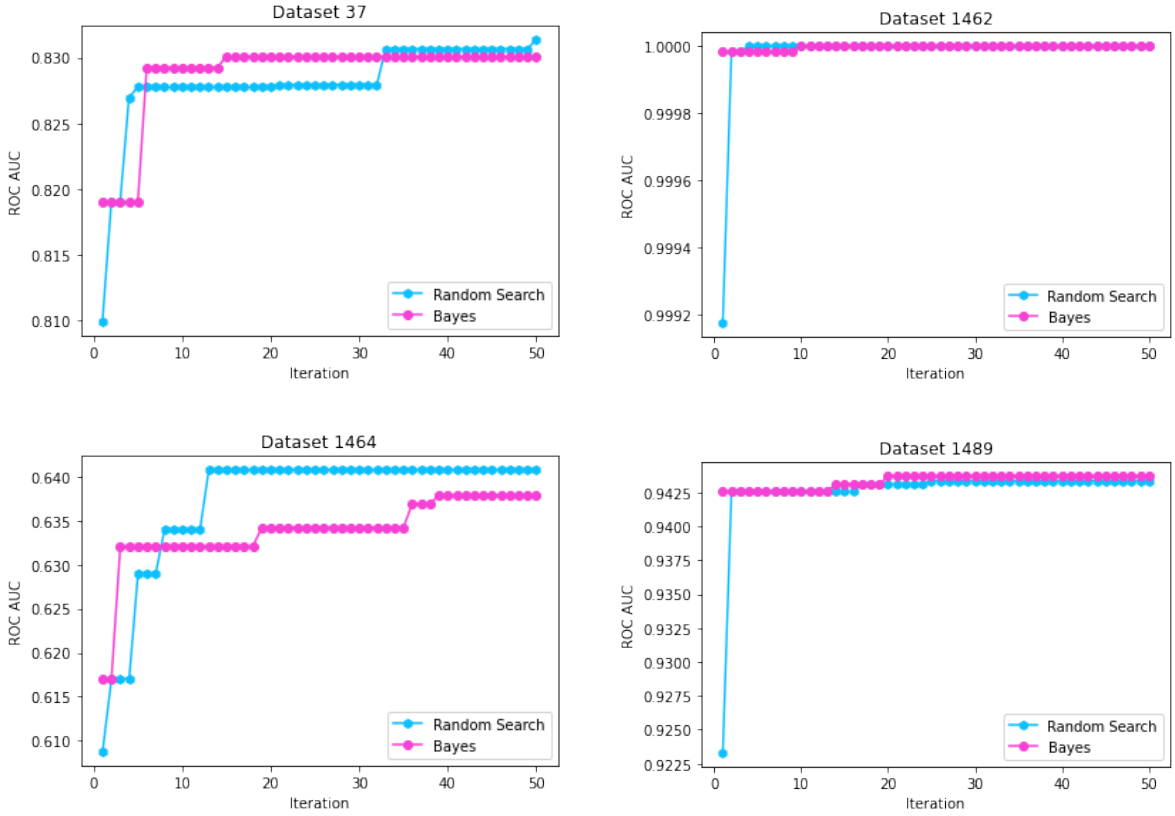
Rysunek 2: Liczba iteracji każdej metody wymagana do uzyskania stabilnych wyników optymalizacji dla algorytmu Decision Tree



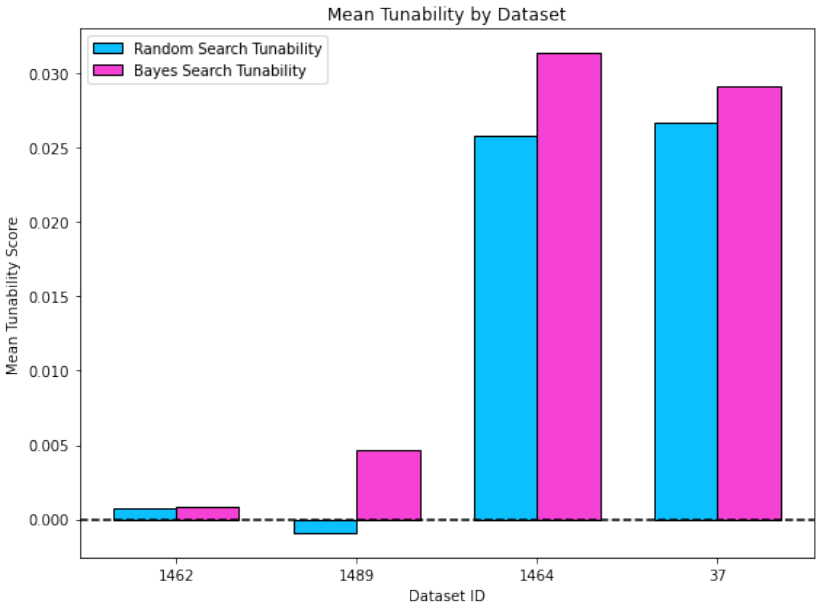
Rysunek 3: Wykres średniej tunowalności dla poszczególnych zbiorów danych dla algorytmu Decision Tree



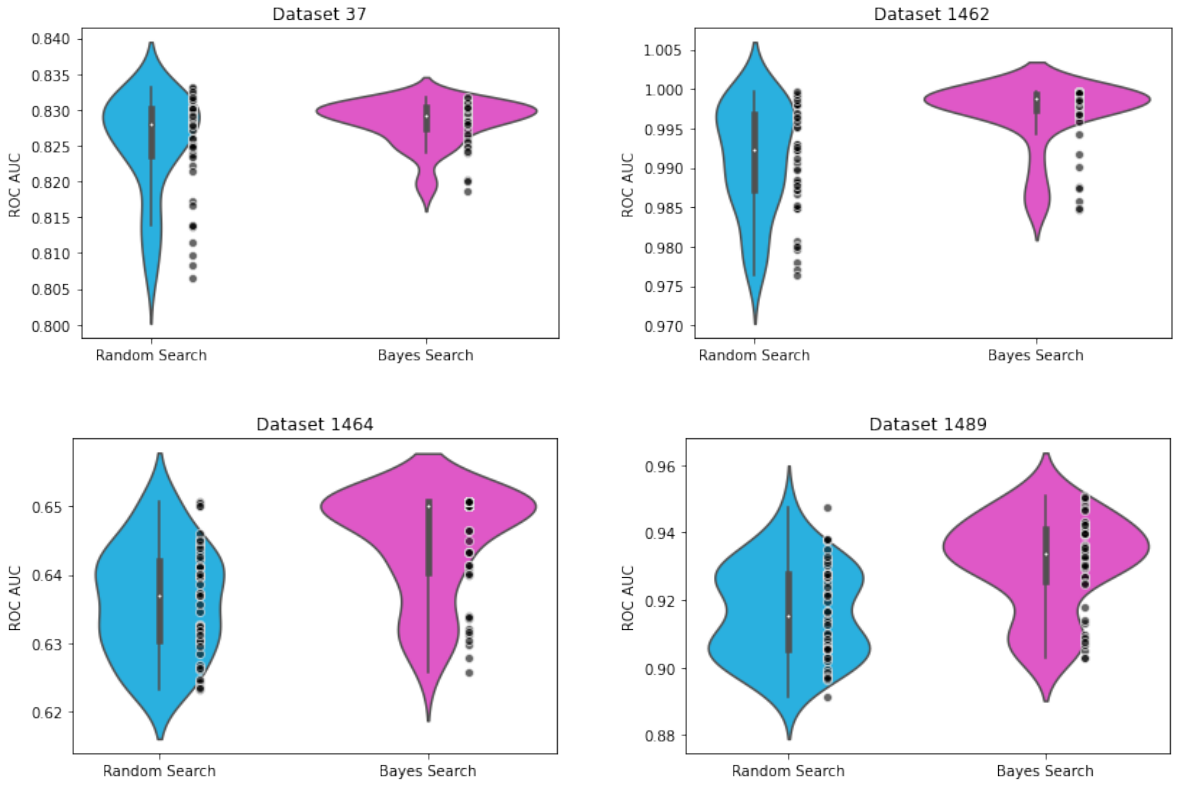
Rysunek 4: Wartości ROC AUC dla różnych konfiguracji hiperparametrów z podziałem na zbiory danych dla algorytmu KNN



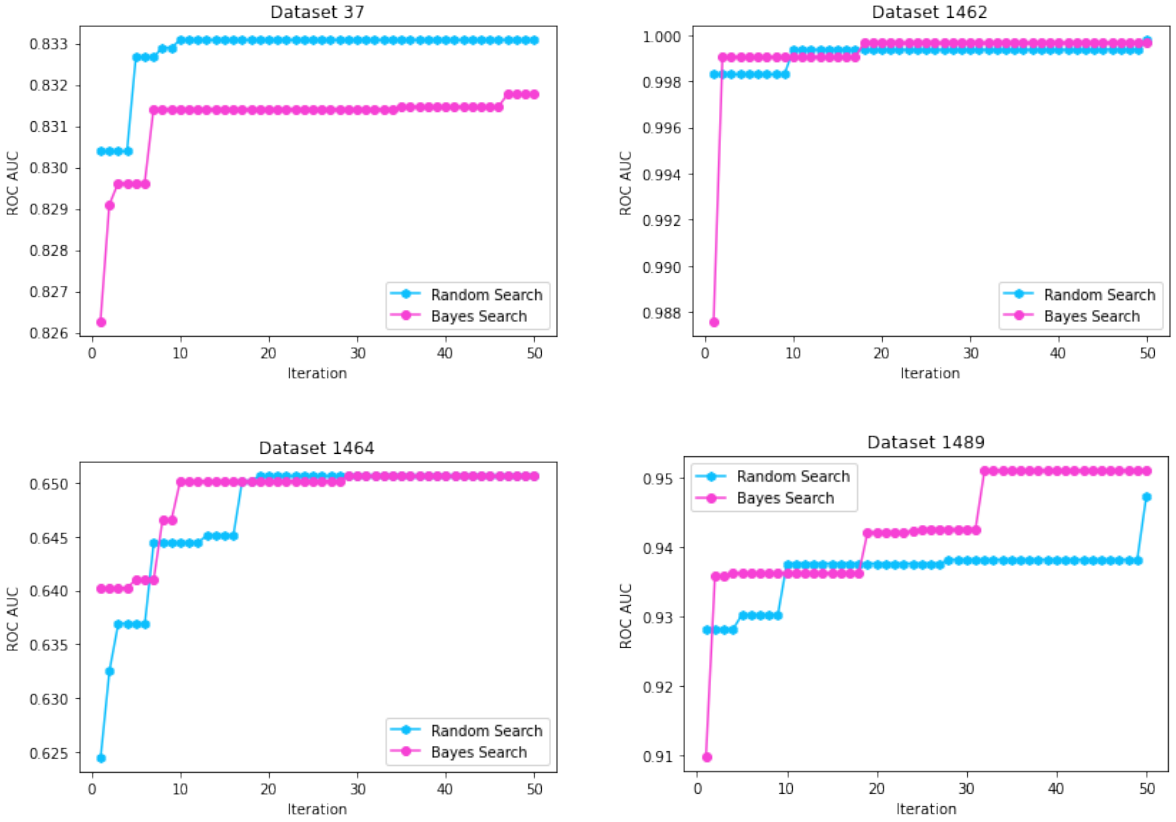
Rysunek 5: Liczba iteracji każdej metody wymagana do uzyskania stabilnych wyników optymalizacji dla algorytmu KNN



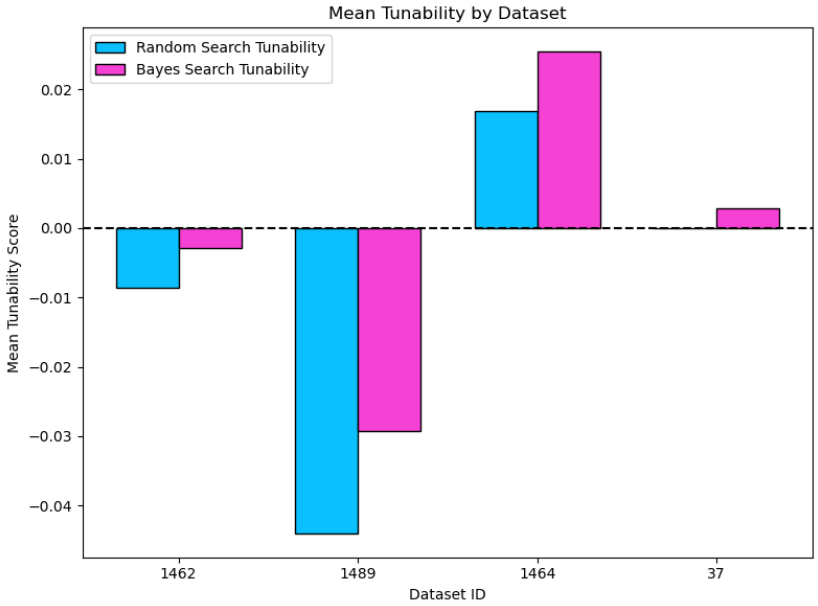
Rysunek 6: Wykres średniej tunowalności dla poszczególnych zbiorów danych dla algorytmu KNN



Rysunek 7: Wartości ROC AUC dla różnych konfiguracji hiperparametrów z podziałem na zbiory danych dla algorytmu Random Forest



Rysunek 8: Liczba iteracji każdej metody wymagana do uzyskania stabilnych wyników optymalizacji dla algorytmu Random Forest



Rysunek 9: Wykres średniej tunowalności dla poszczególnych zbiorów danych dla algorytmu Random Forest

Bibliografia

[1] B. Bischl P. Probst A. Boulesteix. "Tunability: Importance of Hyperparameters of Machine Learning Algorithms". W: *Journal of Machine Learning Research* 20 (2019). URL: `https://jmlr.org/papers/volume20/18-444/18-444.pdf`.