Analiza tunowalności algorytmów uczenia maszynowego

Szymon Gut, Maciej Orsłowski

November 2023

1 Abstrakt

W tym dokumencie, opisane są eksperymenty służące do wyznaczenia tunowalności hiperparametrów oraz modeli uczenia maszynowego zgodnie z definicjami określonymi w artykule naukowym: link. W kolejnych sekcjach zostaną opisane zbiory danych na których zostały przeprowadzone eksperymenty, algorytmy uczenia maszynowego, które badaliśmy, wybrane metody optymalizacji hiperparametrów modelu oraz wizualizacje i wnioski wyciagnięte na podstawie przeprowadzonych doświadczeń.

2 Wykorzystywane zbiory danych

Do przeprowadzenia eksperymentów w celu wyznaczenia tunowalności modeli i hiperparametrów zdecydowaliśmy się użyć czterech zbiorów danych, dla zadania klasyfikacji:

- iris - klasyfikacja wieloklasowa (3 klasy)
- diabetes - klasyfikacja binarna
- wine - klasyfikacja wieloklasowa (3 klasy)
- breast cancer - klasyfikacja binarna

Zbiór diabetes domyślnie został stworzony dla zadania regresji, jednakże na cel naszego projektu, zamienilismy zmienną odpowiedzi o wartościach z dystrybucji ciągłej, na wartości dyskretne. Zrobione to zostało tak, że wartości poniżej mediany zamienione zostały na 0, zaś wartości większe lub równe na 1.

3 Metody samplingu i analizowane algorytmy

Wykorzystywane przez nas metody samplingu, dla których następnie została wyznaczona tunowalność to Random Search (metoda opierająca się na wyborze punktów z rozkładu jednostajnego) oraz Bayes Search (metoda opierająca się na technice Bayesowskiej).

Tunowalności wyznaczyliśmy dla 3 różnych modeli (oraz ich parametrów), używanych w zagadnieniach klasyfikacji: KNN, XGBoost oraz Random Forest.

Przestrzenie hiperparametrów zostały dobrane zgodnie ze wspomnianym artykułem. Zaprezentowane są one w tabeli 1.

Algorytm	Hiperparametr	Typ	Dolna Granica	Górna Granica
kknn	k	integer	1	30
XGBoost	nrounds	integer	1	5000
XGBoost	eta	float	2^{-10}	1
XGBoost	subsample	float	0.1	1
XGBoost	booster	string	-	-
XGBoost	max_depth	integer	1	16
XGBoost	min_child_weight	float	1	2^7
XGBoost	colsample_bytree	float	0	1
XGBoost	alpha	float	2^{-10}	2^{10}
RandomForest	n_estimators	integer	1	2000
RandomForest	max_depth	integer	3	p
RandomForest	max_samples	float	0.1	1
RandomForest	max_features	float	0	1
RandomForest	min_samples_leaf	float	0.1	0.5

Table 1: Parametry algorytmów wraz z ich ograniczeniami.

4 Random Search

4.1 KNN

Z racji na mały zakres hiperparametru k, dla tego algorytmu, metoda random search została wywołana na 20 iteracji, z 5-krotną crosswalidacją.

Na wykresie 1 została przedstawiona historia tuningu, gdzie możemy zobaczyć wizualizację średni score modelu (w tym przypadku accuracy uśrednione przy crosswalidacji) na zbiorze testowym oraz treningowym dla każdego wybranego datasetu.

W celu zbadania stabilości został sporządzony wykres 2 przedstawiający testowy score najlepszego modelu, jaki uzyskaliśmy po odpaleniu optymalizacji hiperparametrów na określoną liczbę iteracji. W tym przypadku algorytm przy niewielkiej liczbie iteracji był w stanie ustabilizować się.

W przypadku tego algorytmu dostępny był tylko jeden parametr - `n_neighbors`. Optymalizacja tego hiperparametru nie wniosła jednak znaczącej poprawy - na połowie zbiorów danych można spodziewać się poprawy o około 1.8% (wykres 3).

4.2 XGBoost

Z racji na bardzo dużą przestrzeń hiperparametrów, dla tego algorytmu, metoda random search została wywołana na 150 iteracji, z 5-krotną crosswalidacją.

Historia tuningu przedstawiona została na wykresie 4, zaś stabilność na wykresie 5. Algorytm bardzo szybko uzyskał stabilność jednak co jest widoczne to fakt, że wybrane parametry nadal nie dawały zadowalających wyników.

Tunowalność parametrów jest zerowa co jest bezpośrednim skutkiem tego, że

model nie był w stanie skutecznie się wytrenować gdy do szukania parametrów użyty został Random Search (wykres 6).

4.3 Random Forest

Podobnie jak w przypadku XGBoost, metoda random search również została wywołana na 150 iteracji z 5-krotną crosswalidacją.

Historia oraz stabilność tuningu przedstawione zostały odpowiednio na wykresach 7 oraz 8. W przypadku tego algorytmu po 30 iteracjach Random Search osiągał stabilność.

Tunowalność parametru `n_estimators` jest średnio ujemna co może znaczyć, że wartość tego parametru w najlepszych odnalezionych zestawach parametru często nie była optymalna (wykres 9). Przyczyną tego mogła być zbyt rozległa przestrzeń poszukiwań dla tego parametru. Optymalizacja pozostałych parametrów daje potencjał poprawy wyników modelu o średnio 1-2% dla 25% datasetów.

4.4 Tunowalność modeli

Tunowalność modeli została przedstawiona na wykresie 10. Została ona wyliczona zgodnie z definicjami znajdującymi się w wspomnianym artykule naukowym, które zostały dodatkowo wspomniane poniżej. Oznaczenia:

- θ - hiperparametry modelu
- m - liczba zbiorów w danych w naszym przypadku równa 4
- $\hat{f}(X, \theta)$ - model predykcyjny
- L(Y, $\hat{f}(X, \theta)$ - loss modelu
- P_j - zbiór danych nr. j

Risk algorytmu ze względu na zbiór danych:

$$R^{(j)}(\theta) = \mathbb{E}[L(Y, \hat{f}(X, \theta))|P_j], \quad j = 1, \ldots, m.$$

Najlepszy zestaw hiperparametrów dla zbioru danych:

$$\theta^{(j)*} := \arg\min_{\theta \in \Theta} R^{(j)}(\theta)$$

Globalny najlepszy zestaw hiperparametrów dla zbioru danych:

$$\theta^* := \arg\min_{\theta \in \Theta} g(R^{(1)}(\theta), \ldots, R^{(m)}(\theta)).$$

Tunowalność algorytmu ze względu na zbiór danych może być obliczona następująco:

$$d^{(j)} := R^{(j)}(\theta^*) - R^{(j)}(\theta^{(j)*}), \quad j = 1, \ldots, m$$

Górne i dolne wąsy na boxplocie (wykres 10) oznaczają odpowiednio kwantyl 0.9 oraz 0.1. Kwantyl 0.9 oznacza jak dużą poprawę można oczekiwać na co najmniej 10% zbioru danych.

Patrząc na wyniki pierwsze co rzuca się w oczy to przewaga Random Forest nad pozostalymi dwoma modelami. Dla modelu XGBoost wyszła tunowalność modelu równa zero, gdyż parametry globalnie najlepsze okazywały się również najlepsze lokalnie. Analiza ta nie jest dokładna i uzyskane wyniki, na pewno nie są mocno informatywne, w celu lepszej analizy dobrze by było przetestować to na conajmniej 10 zbiorach danych, gdyż wyniki zebrane na podstawie tylko 4 nie muszą być do końca informatywne, co zresztą widać na podstawie XGBoost.

5 Bayes Search

5.1 KNN

Metoda Bayes Search została wywołana na 20 iteracji, z 5-krotną crosswalidacją. Przy wywoływaniu tej metody bardzo często był otrzymywany warning, informujacy nas o tym, że metoda wybrała po raz kolejny ten sam parametr, więc liczba iteracji na którą została ustawiona funkcja, mogłaby być minimalnie mniejsza.

Historia tuningu przedstawiona została na wykresie 11, natomiast stabilność na wykresie 12. Podobnie jak w przypadku Random Search, tutaj również niemal od razu algorytm dawał stabilne wyniki.

Wykres 13 przedstawia się bardzo podobnie jak w przypadku Random Search. Również tutaj optymalizacja jedynego parametru nie przyniosła dużej poprawy jakości modelu.

5.2 XGBoost

Metoda Bayes Search została wywołana na 150 iteracji, z 5-krotną crosswalidacją.

Historia tuningu przedstawiona została na wykresie 14, zaś stabilność na wykresie 15. Przy wywołaniu algorytmu Bayes Search na **40 iteracji** uzyskujemy stabilny wynik na każdym z badanych zbiorów.

Tunowalność większości parametrów jest tym razem niezerowa (wykres 16). Wartość 0 dla zmiennej `booster` można tłumaczyć małym obszarem poszukiwań dla niej. Optymalizacja zmiennych `max_depth` oraz `alpha` okazała się jednak nie mieć żadnego wpływu na jakość otrzymywanych modeli. W przypadku pozostałych parametrów wyniki oznaczają potencjał do poprawy wyników modelu rzędu 2-4% na 25% zbiorach danych.

5.3 RandomForest

Tak samo jak w XGBoost, metoda Bayes Search w przypadku Random Forest również została wywołana na 150 iteracji z 5-krotną crosswalidacją.

Historia tuningu dla modelu Random Forest przedstawiona została na wykresie 17, zaś jego stabilność na wykresie 18. Tutaj niemal od razu algorytm zaczął osiągać stabilne wyniki.

Tunowalność większości parametrów jest średnio zerowa, nie licząc `max_depth` dla którego jest ujemna (wykres 19). Wiązać się to może z tym że metoda Bayes Search dla każdego zbioru danych sprawdza inne zestawy parametrów, co oznacza, że znaleziony średnio najlepszy zestaw niekoniecznie dobrze się sprawdza na każdym z datasetów.

5.4 Tunowalność modeli

Tunowalność modeli została przedstawiona na wykresie 20. Podczas, gdy tunowalność XGBoost przy optymalizacji Random Search była zerowa, tak tutaj możemy zauważyć, że XGBoost jest zdecydowanie najlepszy. Możemy zauważyć, że dla tego modelu na conajmniej 10% zbioru danych możemy się wzrostu accuracy modelu po tuningu, aż o 12%.

6 Wpływ metody losowania na tunowalność modeli oraz hiperparametrów

W przypadku KNN tunowalność była niemal identyczna zarówno w przypadku Random Search jak i Bayes Search. Wynika to jednak najprawdopodobniej z niewielkiej liczby hiperparametrów (tylko jeden).

Największy wpływ metody losowania na tunowalność widać w przypadku XGBoost, który nie był w stanie wcale się nauczyć gdy parametry były losowane losowo, natomiast w przypadku Bayes Search tunowalność algorytmu jak i parametrów była najwyższa spośród wszystkich modeli.

Random Forest jednak dawał nieco lepszą tunowalność w przypadku Random Search, zarówno gdy porównuje się tunowalność parametrów jak i samego algorytmu.

7 Appendix: Wizualizacje

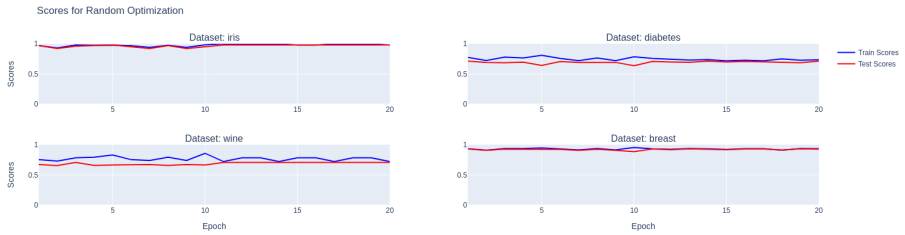

Figure 1: Historia tuningu KNN dla Random Search

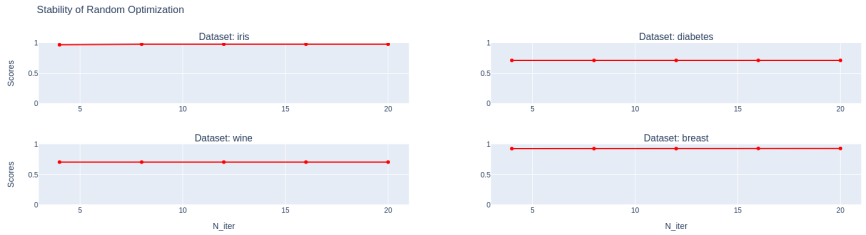

Figure 2: Stabilność Random Search przy szukania optymalnego zestawu hiperparametrów dla KNN

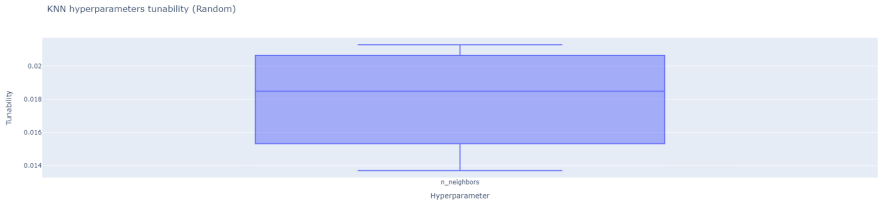

Figure 3: Tunowalność parametrów KNN dla metody optymalizacji Random Search

Figure 4: Historia tuningu XGBoost dla Random Search

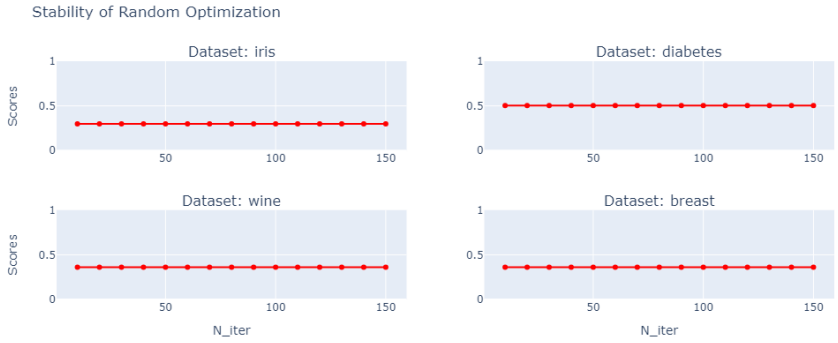

Figure 5: Stabilność Random Search przy szukania optymalnego zestawu hiperparametrów dla XGBoost

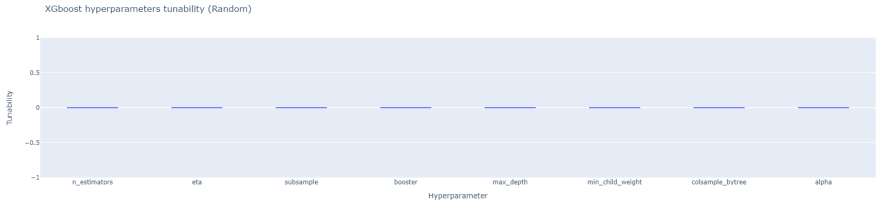

Figure 6: Tunowalność parametrów XGBoost dla metody optymalizacji Random Search

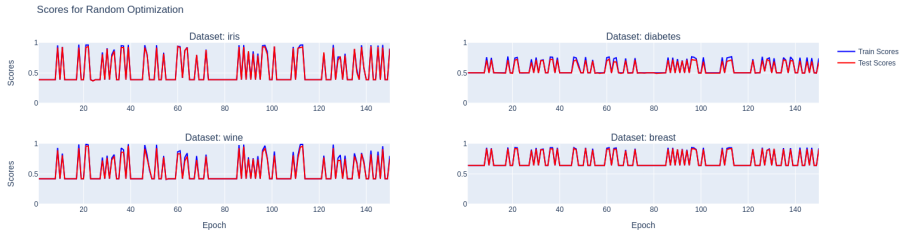

Figure 7: Historia tuningu Random Forest dla Random Search

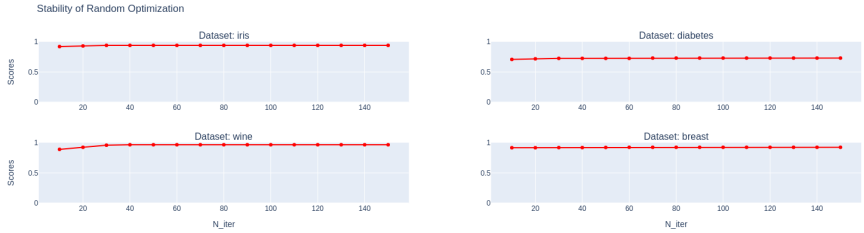

Figure 8: Stabilność Random Search przy szukania optymalnego zestawu hiperparametrów dla Random Forest

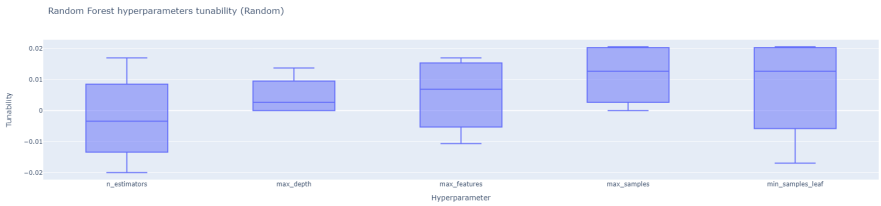

Figure 9: Tunowalność parametrów Random Forest dla metody optymalizacji Random Search

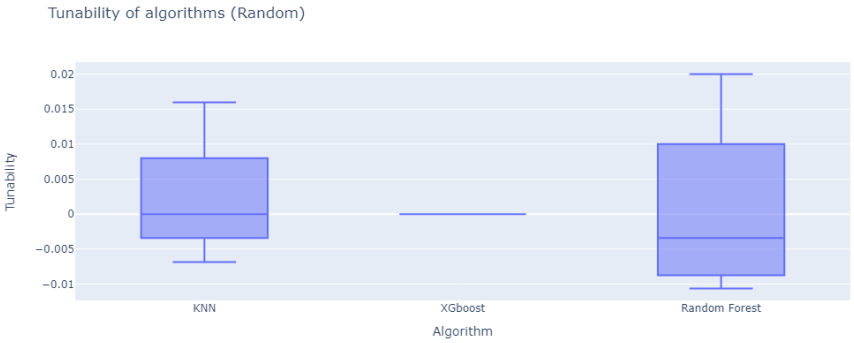

Figure 10: Tunowalność modeli dla metody optymalizacji Random Search

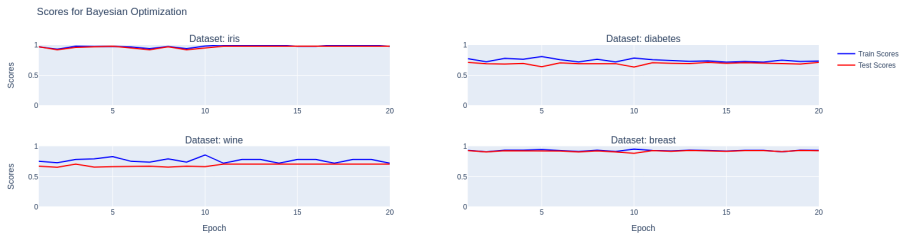

Figure 11: Historia tuningu KNN dla Bayes Search

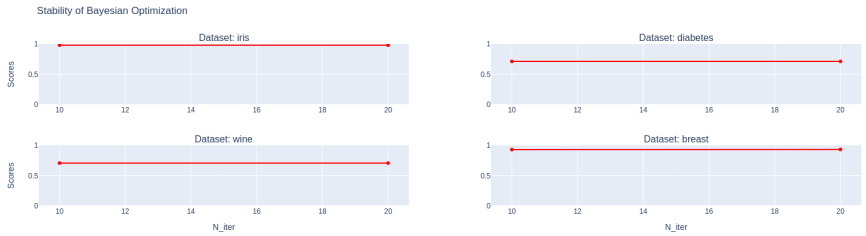

Figure 12: Stabilność Bayes Search przy szukania optymalnego zestawu hiperparametrów dla KNN

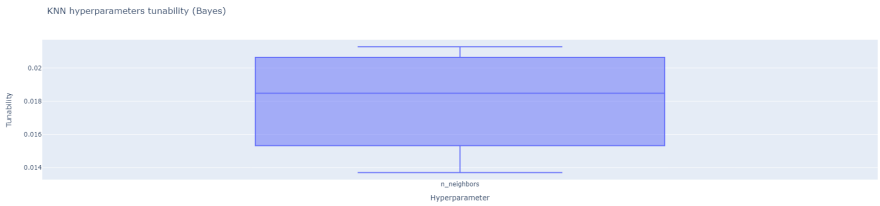

Figure 13: Tunowalność parametrów KNN dla metody optymalizacji Bayes Search

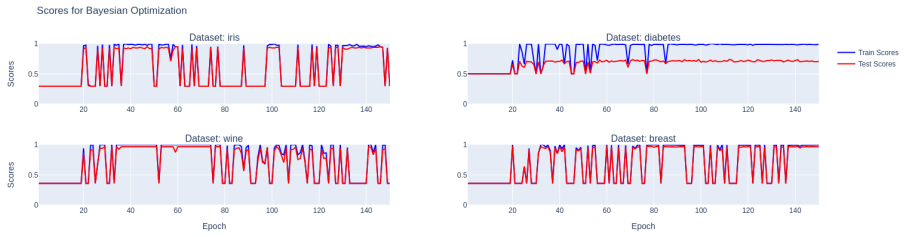

Figure 14: Historia tuningu XGBoost dla Bayes Search

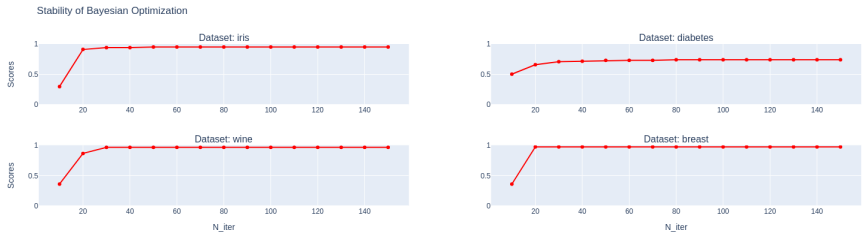

Figure 15: Stabilność Bayes Search przy szukania optymalnego zestawu hiperparametrów dla XGBoost

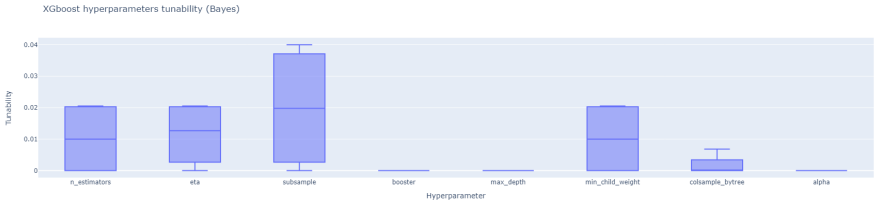

Figure 16: Tunowalność parametrów XGBoost dla metody optymalizacji Bayes Search

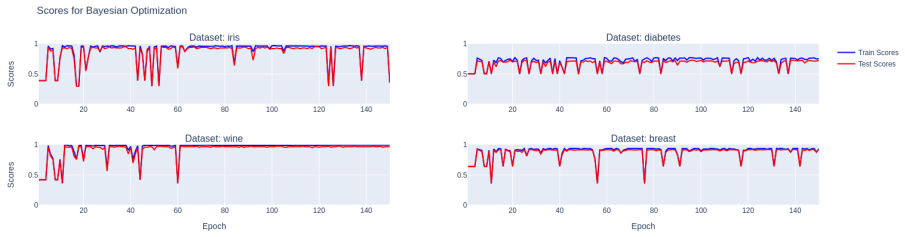

Figure 17: Historia tuningu Random Forest dla Bayes Search

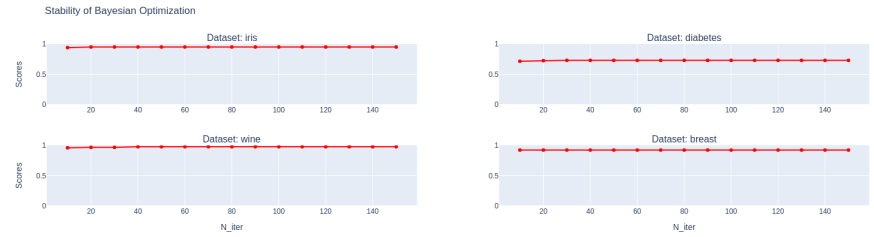

Figure 18: Stabilność Bayes Search przy szukania optymalnego zestawu hiperparametrów dla Random Forest

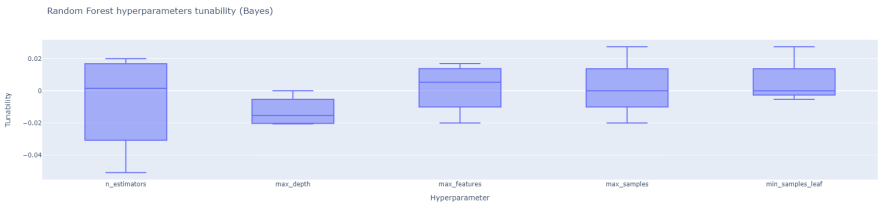

Figure 19: Tunowalność parametrów Random Forest dla metody optymalizacji Bayes Search

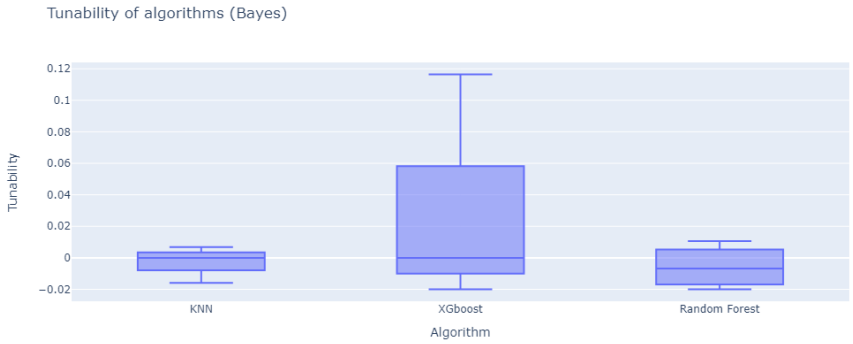

Figure 20: Tunowalność modeli dla metody optymalizacji Bayes Search