AutoML - Praca Domowa 1

Adrianna Wieczorek i Joanna Witczak

November 21, 2023

Contents

1 Wstep

W ramach projektu zbadałyśmy tunowalność algorytmów wykorzystywanych w problemie klasyfikacji binarnej.
Eksperyment przeprowadziłyśmy dla poniższych metod:

- RandomForest
- SVM
- GradientBoosting.

W ramach eksperymentu porównałyśmy dwa sposoby wyboru najlepszych hiperparametrów z siatki. Pierwsza metoda opiera sie o losowanie punktów z rozkładu jednostajnego – RandomizedSearchCV, druga metoda bazuje na optymalizacji bayesowskiej. Do oceny jakości modeli wykorzystałyśmy dokładność.

2 Opis Eksperymentu

2.1 Losowanie z siatki za pomoca losowania z rozkładu jednostajnego

W przeprowadzonym przez nas eksperymencie, badałyśmy jakość dopasowania algorytmu na 4 zbiorach danych: "credit-g", "monks-problems-1", "monks-problems-2", "pc1". Wykorzystane przez nas zbiory dostepne sa w openML.

W celu poprawienia jakości danych, zbudowałyśmy pipeline. W ramach niego przeprowadzone sa nastepujace czynności:

- Imputacja braków danych, w przypadku zmiennych ciagłych uzupełniane sa one przez wartość średnia, a dla zmiennych kategorycznych jest to najcześciej wystepujaca wartość.
- Skalowanie przeprowadzone dla zmiennych ciagłych. Jest ono szczególnie istotne w przypadku metody SVM.
- One-Hot-Encoding.

W kolejnym kroku przeprowadziłyśmy optymalizacje hiperparametrów dla każdego algorytmu za pomoca RandomizedSearchCV. Jest to metoda opierajaca sie na losowaniu punktów z rozkładu jednostajnego. Dzieki niej nie musimy badać wszystkich możliwych kombinacji hiperparametrów. W ramach naszego eksperymentu rozważyłyśmy przeszukiwanie siatki w 50 krokach. Okazało sie to być wystarczajace, gdyż stabilizacja wyników nastepowała stosunkowo szybko - około 20 iteracji.

Każda z wybranych siatek (dla danego algorytmu) wykorzystałyśmy do dopasowania modelu dla każdego zbioru i wyznaczenia średnio najlepszej siatki hiperparametrów. Porównujac wyniki dokładności uzyskane za jej pomoca dla każdego zbioru danych z "najlepsza" siatka wybrana dla tego zbioru, otrzymałyśmy różnice rzedu drugiego miejsca po przecinku. Zdarzyły sie przypadki gdy średnio najlepsza siatka zwróciła wynik lepszy, niż ta dobrana w sposób spersonalizowany.

Dla lasu losowego rozważałyśmy nastepujace hiperparametry dla metody Random Search:

- n_estimators, czyli liczba drzew decyzyjnych w lesie losowym,
- min_samples_split, czyli parametr informujacy o minimalnej wymaganej liczbie obserwacji w danym weźle drzewa decyzyjnego, aby nastapił podział wezła,
- warm_start, czyli parametr odpowiadajacy za to, czy w kolejnej iteracji używamy rozwiazania z poprzedniego dopasowania I do tego dokładamy kolejne drzewa (wartość True), czy dopasowujemy zupełnie nowy las (wartość False),
- max_depth, czyli maksymalna głebokość drzewa (najdłuższa ścieżka miedzy wezłem korzenia a wezłem liścia),
- min_samples_leaf, czyli minimalna liczba próbek, które powinny być obecne w weźle liścia.

Table 1: Siatka hiperparametrów dla Random Forest

Parametr	**Wartości**
model__n_estimators	[20, 50, 100, 200, 500, 700, 1000, 1500, 2000]
model__min_samples_split	[1e-5, 1e-4, 0.0001, 0.001, 0.01, 0.1]
model__warm_start	[True, False]
model__max_depth	[10, 30, 50, 70, 100, 150, 200, 500, 1000]
model__min_samples_leaf	[2, 3, 5, 10, 15, 20]

Dla Gradient Boosting rozważałyśmy nastepujace hiperparametry dla metody Random Search:

- n_estimators, czyli liczba drzew decyzyjnych w modelu,
- subsample, czyli frakcja obserwacji, która ma zostać wykorzystana do dopasowania poszczególnych uczacych sie modeli,
- loss, czyli funkcja straty do optymalizacji,
- max_depth, czyli maksymalna głebokość drzewa (najdłuższa ścieżka miedzy wezłem korzenia a wezłem liścia),
- min_samples_leaf, czyli minimalna liczba próbek, które powinny być obecne w weźle liścia.

Table 2: Siatka hiperparametrów dla Gradient Boosting

Parametr	**Wartości**
model__n_estimators	[10, 50, 100, 150, 500, 1000, 1500, 2000, 2500]
model__subsample	[1e-5, 0.005, 0.01, 0.05, 0.1, 0.5, 0.7, 0.9]
model__max_depth	[15, 50, 100, 150, 500, 1000, 1500, 2000, 2500]
model__min_samples_leaf	[1, 3, 5, 10, 15, 20]
model__loss	['log_loss', 'exponential']

Dla SVM rozważałyśmy nastepujace hiperparametry dla metody Random Search:

- C, czyli hiperparametr regularyzacji, gdzie siła regularyzacji jest odwrotnie proporcjonalna do C,
- kernel, czyli hiperparametr określajacy typ jadra, jakie ma zostać użyte w algorytmie,
- gamma, czyli współczynnik jadra dla jader typu rbf", poly" i sigmoid",
- tol, czyli tolerancja dla kryterium zatrzymania.

Table 3: Siatka hiperparametrów dla SVM

Parametr	**Wartości**
model__C	[0.001, 0.005, 0.01, 1, 10, 20, 50, 100, 200, 500, 1000]
model__kernel	['rbf', 'sigmoid', 'poly']
model__gamma	[0.01, 0.1, 0.5, 1, 5, 20, 50, 100]
model__tol	[1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 0.1]

2.2 Optymalizacja Bayesowska

Tak jak w przypadku zrandomizowanego wyboru hiperparametrów pracowałyśmy na danych oczyszczonych i wystandaryzowanych. Dzieki sposobowi działania tego algorytmu mogłyśmy przeszukiwać przestrzeń ciagła, a nie dyskretna. Optymalizacja bayesowska to technika, która wykorzystuje modele probabilistyczne do tunowania hiperparametrów modeli. Na poczatku tworzony jest model zastepczy, który przybliża funkcje celu. Nastepnie na podstawie tego przybliżenia wybierane sa punkty w przestrzeni hiperparametrów do ewaluacji. Po każdej ewaluacji model jest aktualizowany, a proces iteracyjnie powtarza sie, dażac do znalezienia optymalnej konfiguracji hiperparametrów minimalizujacej funkcje celu.

Rozważałyśmy te same hiperparametry oraz te same modele co w poprzednim kroku.

Jako punkt odniesienia przyjełyśmy średnio najlepsza siatke parametrów otrzymana metoda RandomizedSearchCV.

2.3 Testy

Przeprowadziłyśmy test Kołmogorowa Smirnowa do porównania rozkładów dokładności otrzymanych w kolejnych iteracjach (maksimum do danej iteracji) dla optymalizacji bayesowskiej i RandomizedSearchCV. W ich wyniku otrzymałyśmy, że na podstawie dostepnych danych możemy stwierdzić, że rozkłady dokładności na poziomie istotności 0.05 sa istotnie różne.

W zwiazku z tym zdecydowałyśmy, sie sprawdzić, czy wyniki otrzymywane dla metody optymalizacji bayesowskiej sa istotnie lepsze od wyników otrzymywanych w sposób losowy.

Przeprowadziłyśmy test Wilcoxona, w wyniku którego otrzymałyśmy, że w 9 na 12 przypadków przyjmujemy hipoteze alternatywna (na poziomie istotności 0.05), mówiaca o tym, że dokładność otrzymywana metoda optymalizacji bayesowskiej jest lepsza niż, ta za pomoca losowania z rozkładu jednostajnego.

Warto podkreślić, że testy wykonywałyśmy oddzielnie dla każdej metody i każdego zbioru danych.

Model	Dane	p-wartość (K-S)	p-wartość (Wilcoxon)
RandomForest	dane 1	0	0
RandomForest	dane 2	0.00004	0.0005
RandomForest	dane 3	0	0
RandomForest	dane 4	0	1.0
SVM	dane 1	0	0
SVM	dane 2	0	0.00005
SVM	dane 3	0	0.00382
SVM	dane 4	0	1
GradientBoosting	dane 1	0	0
GradientBoosting	dane 2	0.00192	0.0008
GradientBoosting	dane 3	0	0
GradientBoosting	dane 4	0	1

Table 4: Wyniki testów Wilcoxona i K-S

3 Wyniki

3.1 Stabilizacja RandomForest

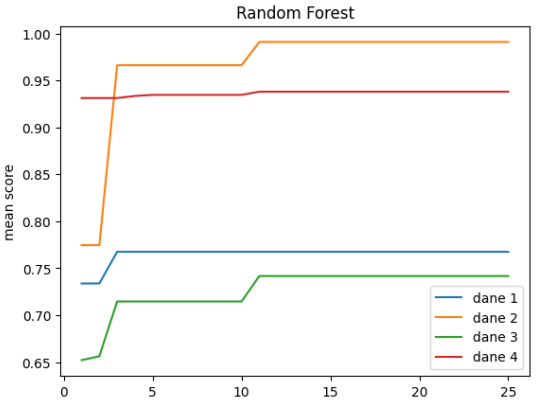

Figure 1: Stabilizacja RandomForest dla Random search.

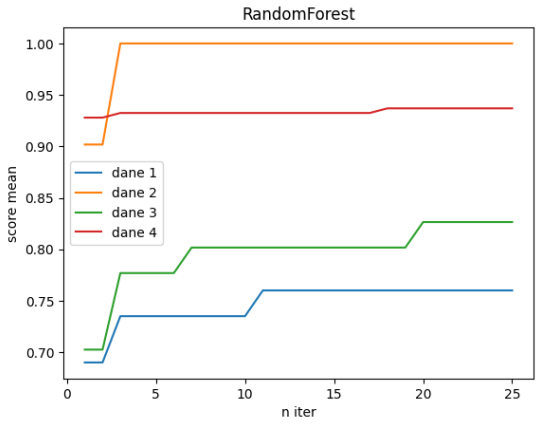

Figure 2: Stabilizacja RandomForest dla optymalizacji bayesowskiej.

3.2 Stabilizacja GradientBoosting

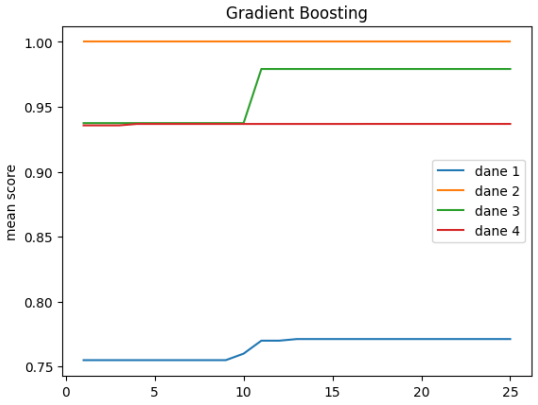

Figure 3: Stabilizacja GradientBoosting dla Random search.

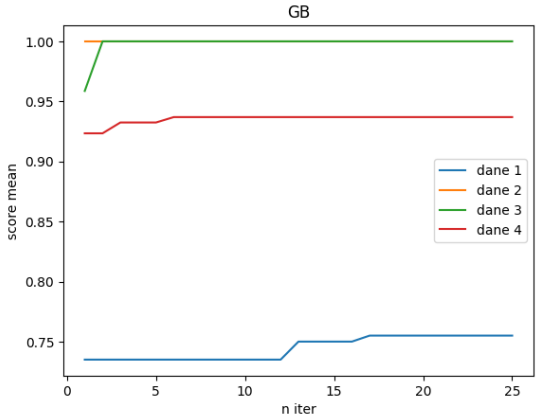

Figure 4: Stabilizacja GradientBoosting dla optymalizacji bayesowskiej.

3.3 Stabilizacja SVM

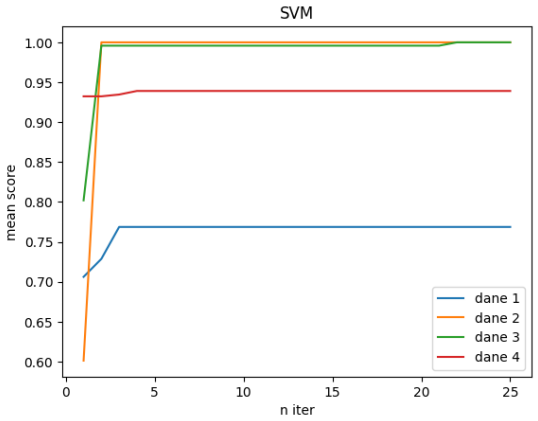

Figure 5: Stabilizacja SVM dla Random search.

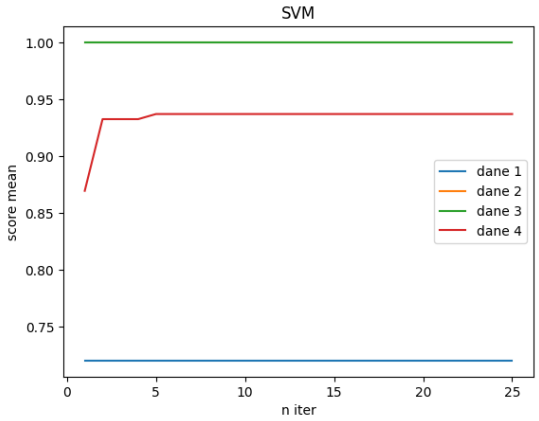

Figure 6: Stabilizacja SVM dla optymalizacji bayesowskiej.

3.4 Tunowalność

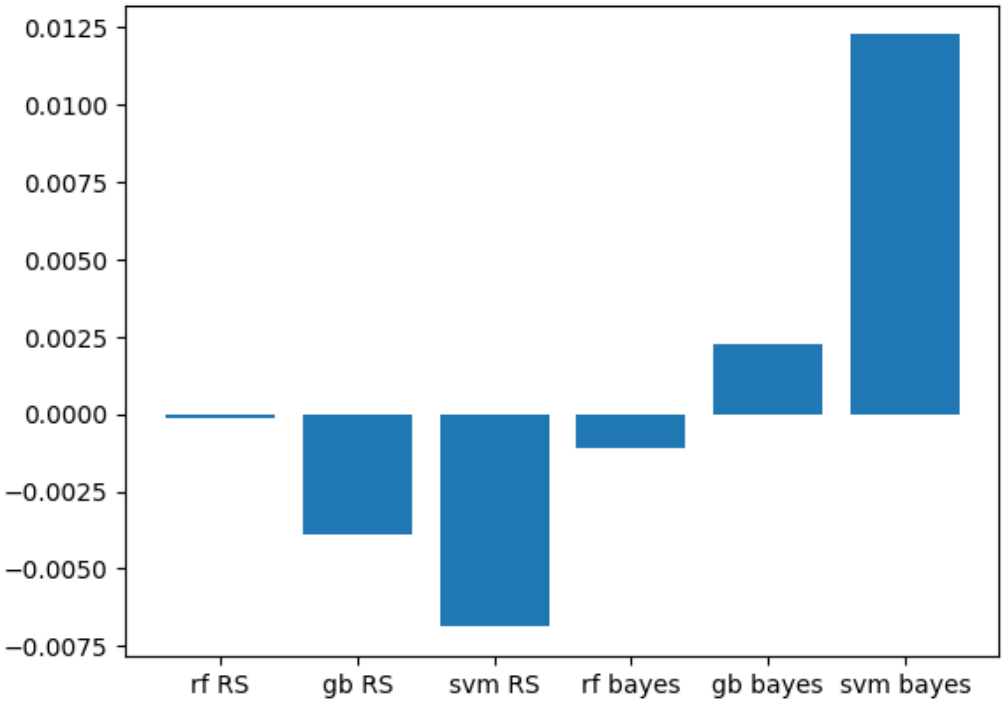

Figure 7: Tunowalność algorytmów.

4 Wnioski

Za pomoca optymalizacji bayesowskiej jesteśmy w stanie otrzymać nieznacznie szybsza zbieżność do optymalnego rozwiazania oraz jesteśmy w stanie otrzymać średnio lepsze wyniki w tej samej liczbie kroków niż za pomoca losowania siatek z rozkładu jednostajnego.

Badajac tunowalność algorytmów zaobserwowałyśmy, że w 3 na 6 przypadków tunowalność przyjeła wartość ujemna. Sugeruje to, że dokładność otrzymana za pomoca średnio najlepszej siatki była znaczaco gorsza od wyników otrzymanych za pomoca najlepszej siatki.