Automatyczne uczenie maszynowe - Praca domowa 1

Laura Korona
Piotr Nieciecki

Listopad 2023

1 Wstęp

Celem pracy domowej jest analiza tunowalności hiperparametrów wybranych algorytmów na różnych zbiorach danych. Do tunowania wykorzystano dwie techniki losowania punktów - random search i optymalizację bayesowską.

2 Wybrane zbiory danych i algorytmy

Wybrano 3 algorytmy klasyfikacji binarnej do zbadania:

- Random Forest
- Gradient Boosting
- K Nearest Neighbors.

Wykorzystano implementacje algorytmów z biblioteki `SciKitLearn` oraz `scikit-optimize`. Natomiast ze strony `openml.org` wybrano następujące 4 zbiory danych:

- scene (ID 312)
- qsar-biodeg (ID 1494)
- pc1 (ID 1068)
- hill-valley (ID 1479).

Powyższe zbiory danych mają ten sam rząd wielkości - każdy z nich ma kilka tysięcy rekordów i kilkaset cech.

3 Opis eksperymentu

Główna część eksperymentu polegała na ustaleniu jakości wytrenowanych modeli dla konfiguracji hiperparametrów ustalanych na podstawie badanych metod optymalizacji. Za miarę jakości poszczególnych modeli obraliśmy średnią dokładność (**accuracy**) modeli po pięciu kroswalidacjach. W każdym przypadku dla danego algorytmu i zbioru danych sprawdzono po 30 konfiguracji hiperparametrów.

3.1 Zakres hiperparametrów dla poszczególnych modeli

Wykorzystane metody optymalizacji wymagają jawnego zdefiniowania zakresów optymalizowanych parametrów. Na podstawie dokumentacji wykorzystanej biblioteki - `scikit-learn` ustaliliśmy takie wartości, które działają dla większości badanych przez społeczność zbiorów danych.

3.1.1 Random forest

Dla lasu losowego przyjęto następujace zakresy dla tunowanych hiperparamentrów:

- `n_estimators` - przedział $[50, 150]$ (domyślna wartość dla tego hiperparametru w bibliotece `scikitlearn` wynosi 100, więc badane są wartości do niej zbliżone (domyślne wartości hiperparametrów często dają dobre rezultaty))
- `criterion` - badane są wszystkie dostępne rodzaje, czyli `gini`, `entropy`, `log_loss`
- `min_samples_split` - przedział $[2, 50]$ (domyślna wartość dla tego hiperparametru w bibliotece `scikitlearn` wynosi 2, więc warto tę wartość rozważyć, lecz sprawdzamy również trochę większych wartości, by uniknąć zbyt głębokich drzew)

3.1.2 Gradient boosting

Rozważane zakresy tunowanych hiperparametrów znajdują się poniżej:

- `loss` - badane są obydwa dostępne rodzaje tego parametru, czyli `log_loss` i `exponential`
- `n_estimators` - badany jest przedział $[100, 500]$, gdyż domyślna wartość w `scikitlearn` wynosi 100 (przyjęto zatem, że warto ją rozważyć), lecz podana jest również w dokumentacji tej biblioteki informacja, że im większa wartość tego hiperparametru, tym prawdopodobnie uzyskuje się lepsze wyniki
- `min_samples_split` - tak samo, jak dla lasu losowego, wybrano przedział $[2, 50]$, również na podstawie sugerowanej wartości w `scikitlearn`

3.1.3 K najbliższych sąsiadów

Poniżej znajdują się przyjęte zakresy dla tunowanych hiperparametrów algorytmu:

- `n_neighbors` - domyślna wartość dla tego hiperparametru wynosi 5, więc badany jest przedział zbliżony do tej wartości: $[2, 20]$
- `weights` - rozważane są obydwa rodzaje wag dostępne w `scikitlearn`, czyli `uniform` i `distance`
- `leaf_size` - sugerowana wartość w dokumentacji wynosi 30, badane są więc zbliżone do niej wartości: $[20, 40]$

4 Wnioski

4.1 Najlepsze domyślne konfiguracje

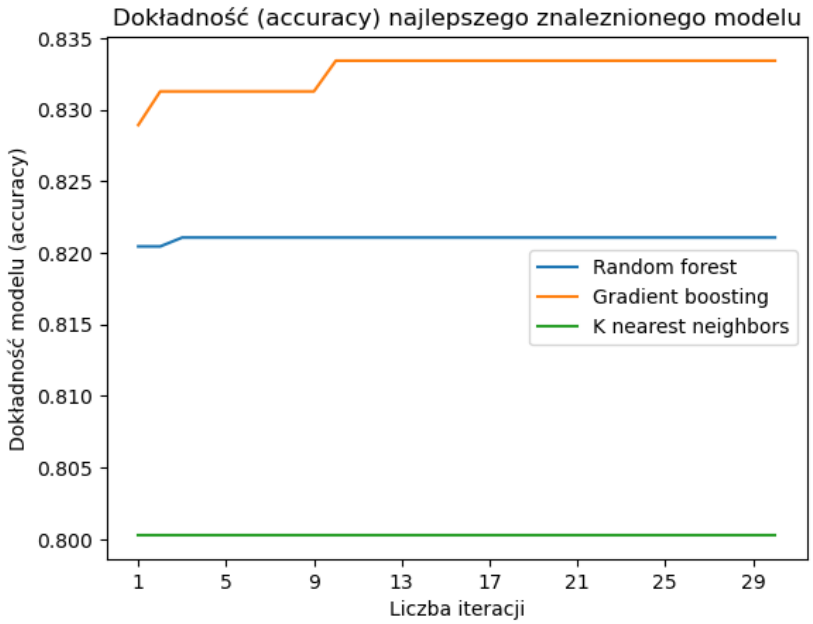
Policzyliśmy średnią dla każdej z konfiguracji po badanych zbiorach danych. Za najlepszą domyślą konfigurację dla danego algorytmu wybraliśmy tę dla której ta

wartość była największa. Z powodu, że optymalizacja bayesowska przy każdej iteracji po zbiorach lub algorytmach przechodzi przez inne konfiguracje hiperparamerów, to dla niej nie wyznaczyliśmy takiego parametru.
Random Search - najlepsza konfiguracja:

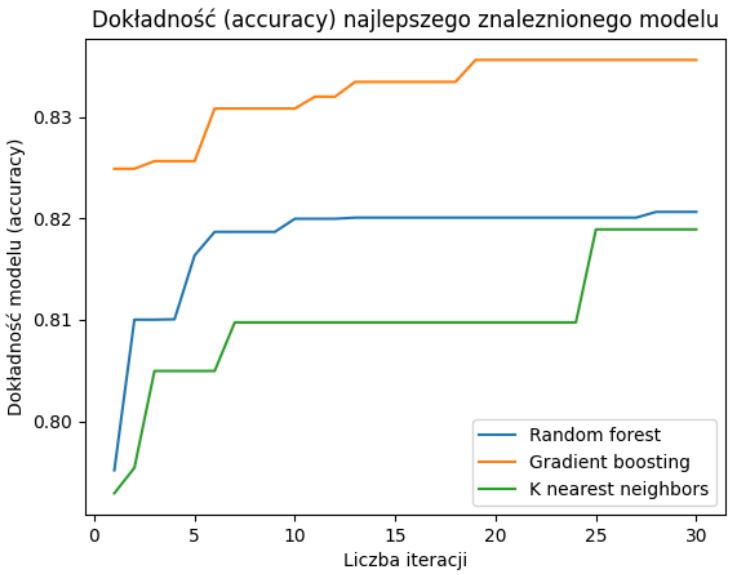
- Random Forest: `n_estimators`: 102, `max_depth`: 24, `criterion`: 'log_loss'
- Gradient Boosting: `n_estimators`: 451, `min_samples_leaf`: 30, `loss`: 'exponential'
- K Nearest Neighbors: `weights`: 'uniform', `n_neighbors`: 6, `leaf_size`: 20

4.2 Ile iteracji każdej metody potrzebujemy, żeby uzyskać stabilne wyniki optymalizacji?

Zbadano, ile iteracji potrzeba, by wyniki stabilizacji przeprowadzanych optymalizacji ustabilizowały się. Poniższe wykresy przedstawiają otrzymane wyniki.



Rysunek 1: Dokładność przy najlepszych znalezionych podczas Random Search hiperparametrach w zależności od liczby iteracji



Rysunek 2: Dokładność przy najlepszych znalezionych podczas optymalizacji bayesowskiej w zależności od liczby iteracji

Można zauważyc, że podczas Random Search, Random Forest stabilizuje się przy około 4 iteracjach, Gradient Boosting - przy mniej więcej 11, zaś dla K Nearest Neighbors najlepszy zestaw hiperparametrów wylosował się akurat w pierwszej iteracji.

Natomiast podczas optymalizacji bayesowskiej Random Forest stabilizuje się przy mniej więcej 10 iteracjach, Gradient Boosting - przy 19, a K Nearest Neighbors przy 25.

4.3 Tunowalność poszczególnych algorytmów

4.3.1 Random search

Dla algorytmu random search tunowalność została wyznaczona w następujący sposób. Jest to różnica przyjętej jakości modelu dla modelu operatego na parametrach, które najlepiej się sprawiły w średniej dla wszystkich zbiorów (przedstawione w rozdziale 4.1) od najlepszej konfiguracji dla danego zbioru.

Uzyskane tak wyniki zostały zmieszczone w tabelce 1. Warto zwrócić uwagę, że w kilku przypadkach konfiguracja najlepsza dla wszystkich zbiorów okazała się również najlepszą dla poszczególnych.

Tabela 1: Wyniki tunowaności dla algorytmu random search

	scene	qsar-biodeg	pc1	hill-valley
Random forest	-0.000834	-0,008530	-0,008108	-0.000840
Gradient boosting	-0.000415	-0.015165	-0.001797	0.0
KNN	0.0	0.0	-0.001801	-0.013172

4.3.2 Optymalizacja bayesowska

Z powodu tego, że dla optymalizacji bayesowkiej każde uruchomienie dla różnych algorytmów i zbiorów danych przechodzi po różnych konfiguracjach utrudnione było wyznaczanie najlepszych domyślnych parametrów. Z tego powodu w tym przypadku porównaliśmy wyniki do domyślnych hiperparametrów wyznaczonych przy pomocy random search'a. Wyniki zamieszczone są w tabeli 2. Warto zwrócić uwagę, że zdarzyło się tak, że wynik dla najlepszej średniej konfiguracji okazał się lepszy od zoptymalizowanej bayesowsko.

Tabela 2: Wyniki tunowaności dla optymalizacji bayesowskiej

	scene	qsar-biodeg	pc1	hill-valley
Random forest	0.00124	-0.007582	-0.003607	-0.009896
Gradient boosting	-0.000415	-0.015165	-0.003595	-0.001632
KNN	-0.020771	-0.023696	-0.009013	-0.050355

4.4 Czy występuję *bias sampling?*

Porównując dane z tabel 1 i 2 można zobaczyć, dane w nich różnią się od siebie, ale nie można wyznaczyć żadnego dominującego trendu, który wskazywałby na *bias samplingu*.