AutoML: Projekt 1 - Tunowalność hiperparametrów

Aleksandra Idczak, Krzysztof Maślak, Jakub Winiarski

1 Wstęp

Celem projektu jest znalezienie nowych "defaultowych" wartości hiperparametrów dla trzech wybranych modeli uczenia maszynowego oraz analiza ich tunowalności. Wybraliśmy pięć zbiorów danych, na których sprawdzamy jak radzą sobie modele z rożnymi zestawami hiperparametrów. Do wyboru kolejnych zestawów hiperparametrów zastosowaliśmy dwie metody: Random Search oraz Bayes Search.

2 Tunowalność

Tunowalność (*ang. tunability*) algorytmu określa jak bardzo działanie algorytmu uczenia maszynowego może się poprawić dzięki odpowiedniemu doborowi wartości hiperparametrów. Aby ją zdefiniować zgodnie z [PPB] zdefiniujmy najpierw problem oraz wprowadźmy pomocnicze pojęcia.

Rozważmy zmienną docelową Y, wektor cech X oraz nieznany rozkład łączny P wektora (X, Y), z którego pobrano próbkę T o liczności n. Algorytm uczenia maszynowego konstruuje model predykcyjny $\hat{f}(X, \theta)$, zależny od hiperparametrów $\theta = (\theta_1, \ldots, \theta_k)$ z przestrzeni hiperparametrów $\Theta = \Theta_1 \times \cdots \times \Theta_k$. Błąd między Y a $\hat{f}(X, \theta)$ mierzymy pewną ustaloną funkcją straty $L(Y, \hat{f}(X, \theta))$. Interesuje nas ryzyko, a więc oczekiwana strata modelu na nowych danych: $R(\theta) = \mathbb{E}\big[L(Y, \hat{f}(X, \theta)) \mid P\big]$. Dla m różnych zbiorów danych $P_1, \ldots, P_m$ otrzymujemy: $R^{(j)}(\theta) = \mathbb{E}\big[L(Y, \hat{f}(X, \theta)) \mid P_j\big]$ dla $j = 1, \ldots, m$.

Definiujemy optymalną konfigurację hiperparametrów dla j-tego zbioru danych jako:

$$\theta^{(j)\star} = \arg\min_{\theta \in \Theta} R^{(j)}(\theta), \quad j = 1, \ldots, m,$$

oraz domyślną konfigurację hiperparametrów, jako konfigurację, która działa dobrze na wielu różnych zbiorach danych:

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

gdzie g jest pewną funkcją agregującą. Wykorzystując te dwie wartości możemy zdefiniować tunowalność algorytmu dla j-tego zbioru danych:

$$d^{(j)} = R^{(j)}(\theta^{\star}) - R^{(j)}(\theta^{(j)\star}).$$

Dla m zbiorów danych daje nam to rozkład poprawy wydajności algorytmu na m zbiorach danych i możemy rozumieć tunowalność jako wartość pewnej funkcji agregującej od tych wartości - przykładowo średniej lub mediany.

My w swoim eksperymencie jako R zastosowaliśmy miarę AUC, natomiast za g wzięliśmy średnią. Autorzy [PPB] do problemu minimalizacji występującego w definicji domyślnych hiperparametrów podeszli dwukrokowo. Najpierw przy użyciu modelu zastępczego estymowali funkcję $R^{(j)}(\theta)$, a później metodą Random Search znajdowali minimum jej przybliżonej postaci. My pominęliśmy pierwszy krok - optymalizowaliśmy funkcję $R^{(j)}(\theta)$ za pomocą Random Searcha bez pomocy modelu zastępczego.

3 Zbiory danych

Do przeprowadzenia eksperymentu wybraliśmy pięć zbiorów danych do klasyfikacji binarnej. Ich podstawowe własności opisaliśmy w tabeli. Dla wszystkich danych zastosowaliśmy jednakowy preprocessing - dla zmiennych kategorycznych OneHotEncoding, a numeryczne zostały przeskalowanie za pomocą StandardScaler. Zmienne celu zostały wszędzie zamienione na 1 i 0.

Dane	wiersze	zmienne	braki danych	zm. kategoryczne	procent klasy 1	źródło
Adult	48842	14	nie	7	23.93%	OpenML
Phishing	11055	30	nie	30	55.69%	OpenML
Phoneme	5403	5	nie	0	29.35%	OpenML
Iranian Churn	3150	13	nie	5	15.72%	UCI
Shoppers	12330	17	nie	7	15.47%	UCI

Tabela 1: Kolumny "zmienne" i "zm. kategoryczne" odnoszą się do ilości zmiennych objaśniających - nie uwzględniamy tam zmiennej celu. Kolumna "procent klasy 1" informuje jaki odsetek wartości zmiennej celu należy do klasy "1".

4 Modele i hiperparametry

Modele, które zdecydowaliśmy się zbadać to Random Forest, XGBoost i regresja logistyczna z karą elastic net. Skorzystaliśmy z ich implementacji w pakiecie scikit-learn. Dla każdego z nich wybraliśmy hiperparametry, dla których potem policzyliśmy nowe wartości domyślne. Dla każdego modelu stworzyliśmy przestrzeń hiperparametrów, z której następnie będą wybierane/losowane kolejne wartości hiperparametrów.

Autorzy w [PPB] definiują dobrą przestrzeń hiperparametrów jako taką, która z dużym prawdopodobieństwem zawiera w sobie optymalne hiperparametry dla większości zbiorów danych. Podają później tabelkę z zakresami hiperparametrów, z których oni korzystali w swoim eksperymencie. Ze względu na mniejszą moc obliczeniową, a także brak odpowiedników niektorych hiperparametrów w modelach ze scikit-learn, zdecydowaliśmy się trochę zawęzić przestrzeń poszukiwań. Poniżej znajdują się tabelki zawierające wybrane przez nas zakresy.

Hiperparametr	Zakres	Znaczenie hiperparametru
n_estimators	1 – 2000	liczba drzew
max_samples	0.5 – 1	frakcja próbek użytych w każdym drzewie
max_features	0.2 – 0.8	liczba zmiennych losowanych przy podziale
min_samples_leaf	0 - 0.1	minimum procent próbek w liściu
criterion	gini / entropy	kryterium podziału

Tabela 2: Zakresy i typy hiperparametrów Random Forest używanych w eksperymencie.

Hiperparametr	Zakres	Znaczenie hiperparametru
n_estimators	1 – 3000	liczba drzew/iteracji w boosting
learning_rate	0 – 0.3	współczynnik uczenia, wpływ pojedynczego drzewa
subsample	0.5 – 1	frakcja próbek użytych w każdym drzewie
max_depth	2 – 8	maksymalna głębokość pojedynczego drzewa
max_features	0.2 – 0.8	frakcja cech losowanych przy każdym podziale drzewa

Tabela 3: Zakresy i znaczenie wybranych hiperparametrów Gradient Boosting w eksperymencie.

Hiperparametr	Zakres	Znaczenie hiperparametru
C	0 - 2	odwrotność siły regularizacji
l1_ratio	0 – 1	udział regularyzacji L1 w karze

Tabela 4: Zakresy i znaczenie hiperparametrów Logistic Regression z karą Elastic Net.

5 Eksperyment

5.1 Opis

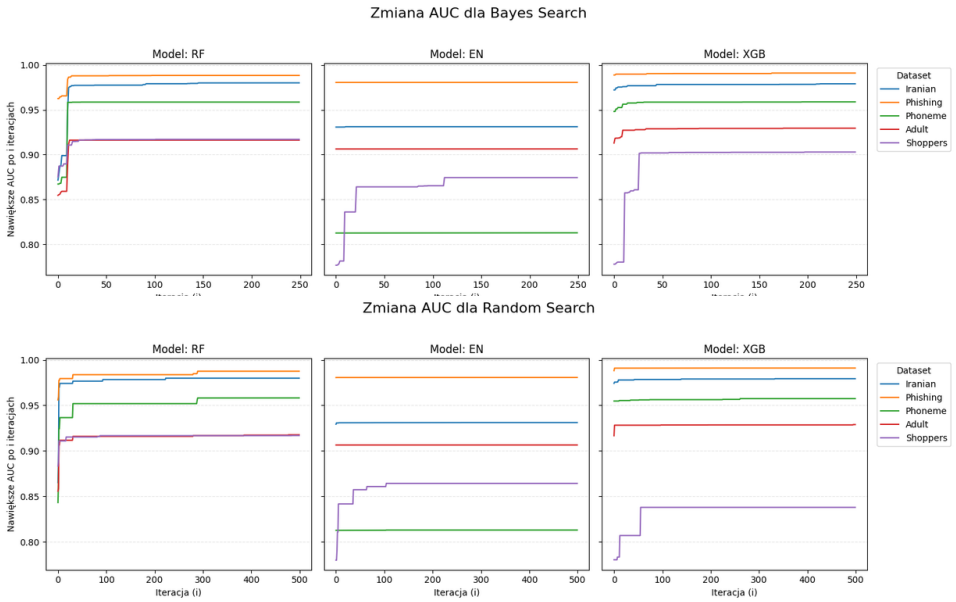
Obie metody optymalizacji hiperparametrów (*Bayes Search* i *Random Search)* przetestowaliśmy na wszystkich zbiorach danych. Wykonaliśmy trzykrotną kroswalidację i policzyliśmy średnie AUC z trzech zbiorów.

Dla każdego zbioru danych wybraliśmy iterację z najwyższą wartością AUC i zapisaliśmy odpowiadające jej wartości hiperparametrów. Ponadto dla optymalizacji *Random Search* dla każdej iteracji policzyliśmy średnią wartość AUC dla

wszystkich zbiorów danych i wybraliśmy iterację z najwyższym średnim AUC. Hiperparametry z tej iteracji to nowe wartości domyślne (default) hiperparametrów, posłuzyly nam potem do obliczenia tunowalności algorytmów.

5.2 Ustalenie liczby iteracji

Eksperyment rozpoczęliśmy od ustalenia liczby iteracji metod samplingu hiperparametrów. Musieliśmy też wziąć pod uwagę nasze ograniczenia sprzętowe. Zdecydowaliśmy się ustalić liczbę iteracji Random Search na 500. Niestety dla optymalizacji Bayes Search oraz zbioru *Adults Dataset* obliczenia trwały zbyt długo, aby eksperyment mający 500 iteracji można było policzyć w rozsądnym czasie. Z tego względu ustaliliśmy liczbę iteracji na 250, dzięki czemu eksperyment związany z Bayes Search trwał 30h.



Rysunek 1: Stabilność optymalizacji za pomocą metody Bayes Search

Analiza stabilności Random Search i Bayes Search pokazuje, że Bayes Search zwykle szybciej (pamiętając, że na powyższych wykresach mamy inne skale osi OX) i bardziej płynnie zbliża się do optimum, podczas gdy Random Search charakteryzuje się bardziej skokowym przebiegiem i większą zmiennością. W sposób szczególny uwidacznia się to w przypadku modelu Random Forest. W przypadku Elastic Net oba podejścia zachowują się stabilnie na większości zbiorów, a wyraźna poprawa pojawia się jedynie na zbiorze Shoppers, gdzie obie metody uzyskują z porównywalną szybkością satysfakcjonujące wyniki. Dla XGBoost obie metody osiągają zbliżony poziom stabilizacji i podobnie szybko dochodzą do optymalnych wartości. Patrząc na oba wykresy, w większości przypadków stosunkowo szybko udawało się znaleźć optymalne hiperparametry. Bardzo często wystarczało do tego już 50 iteracji.

6 Wyniki

6.1 Optymalne hiperparametry

W tabeli 5 przedstawiliśmy wartości optymalnych (default) hiperparametrów wybranych zgodnie z metodyką opisaną w sekcji 5.1 oraz najlepsze hiperparametry dla każdego ze zbiorów danych.

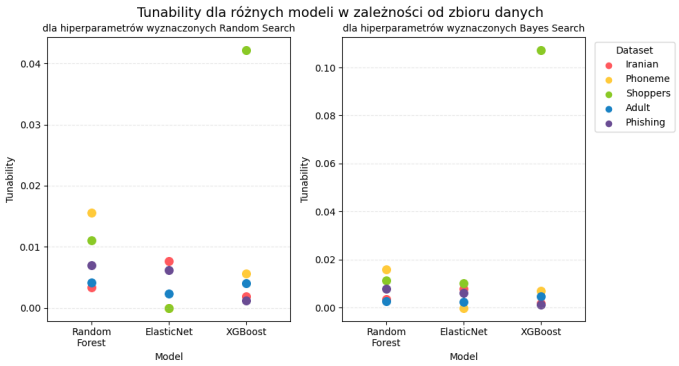
6.2 Tunowalność

Tunowalość modeli policzyliśmy odejmując od wartości AUC dla modelu z defaultowymi hiperparametrami wartość AUC dla najlepszego modelu na danym zbiorze danych. Rysunek 2 przedstawia tunowalność modeli dla wszystkich zbiorów danych.

Celem eksperymentu zdefiniowanym w projekcie była "analiza i opis tunowalności poszczególnych algorytmów". Zdecydowaliśmy się podzielić analizę na dwa wykresy - dla różnych metod samplingu. Po uwzględnieniu skali okazuje się, że wartości tunowalności dla poszczególnych modeli są średnio minimalnie większe dla metody *Bayes Search* niż dla *Random Search*.

Model	hiperparametr	default	Adult		Phishing		Phoneme		Iranian		Shoppers	
			Random	Bayes	Random	Bayes	Random	Bayes	Random	Bayes	Random	Bayes
RandomForest	n_estimators	349	871	41	1282	1628	1424	1478	775	1414	113	2000
	min_samples_leaf	0.002	1.8e-4	3e-4	1.1e-4	1e-12	1.2e-4	1e-12	0.001	7e-4	0.012	0.011
	max_features	0.472	0.283	0.395	0.337	0.2	0.454	0.2	0.376	0.283	0.512	0.8
	max_samples	0.765	0.832	0.639	0.954	0.678	0.911	1	0.948	0.964	0.687	0.5
	criterion	gini	gini	gini	gini	entropy	entropy	gini	entropy	entropy	entropy	entropy
XGBoost	n_estimators	604	2693	2151	1669	936	1200	2977	1970	277	1244	2630
	learning_rate	0.067	0.009	0.056	0.022	0.019	0.076	0.076	0.009	0.081	0.007	1e-12
	max_depth	6	5	3	4	6	7	8	6	6	2	2
	max_features	0.301	0.734	0.2	0.207	0.2	0.356	0.2	0.286	0.247	0.226	0.8
	subsample	0.674	0.987	1	0.754	0.635	0.748	0.645	0.601	0.808	0.735	0.653
ElasticNet	l1_ratio	0.161	0.015	1e-12	0.986	1	0.161	1e-12	0.15	0.008	0.161	0.204
	C	0.01	0.273	0.342	1.949	2	0.01	0.034	1.975	1.992	0.01	4.1e-4

Tabela 5: Optymalne wartości hiperparametrów oraz wartości *default* wyznaczone podczas eksperymentu. Wartości są zaokrąglone do 3 miejsca po przecinku.



Rysunek 2: Porównanie tunowalności dla Random Search vs Bayes Search

Są to oczekiwane wyniki z dwóch względów. Po pierwsze, wartości hiperparametrów powinny dawać lepsze średnie wyniki AUC dla *Bayes Search*, ponieważ jest on bardziej wyrafinowaną metodą optymalizacyjną, a nie polega na losowym wybieraniu hiperparametrów.

Różnica jest jednak niewielka - wartości AUC dla hiperparametrów domyślnych są niewiele niższe od wartości AUC dla hiperparametrów optymalnych. Można wysnuć wniosek, że algorytmy mają małą tunowalność. Jednak może to być spowodowane małą liczbą m zbiorów danych - domyślne hiperparametry ustalano na podstawie 5 zbiorów danych. Prawdopodobnie, gdybyśmy ustalali domyślne hiperparametry przy użyciu kroswalidacji na 4 zbiorach danych i wyznaczali tunowalność na "testowym" zbiorze 5, to wartości tunowalności by wzrosły, co miało miejsce także w [PPB].

Wyjątkiem jest zbiór danych Shoppers dla algorytmu XGBoost. Stawiamy hipotezę iż jest to związane z wysokim niezbalansowaniem klas w zbiorze danych co okazało się być problematyczne dla algorytmu XGBoost przy "defaultowym" doborze hiperparametrów.

W związku z powyższym, zjawisko bias samplingu wystąpiło w nieznacznym stopniu.

Dodatkowe materiały

Szczegółowy kod źródłowy znajduje się w repozytorium github [GIT].

Literatura

[PPB] P. Probst, A.-L. Boulesteix, B. Bischl, *Tunability: Importance of Hyperparameters of Machine Learning Algorithms*, Journal of Machine Learning Research, 20(53), 2019.

[GIT] Repozytorium github projektu: `https://github.com/kubawini/AutoML-Project-1/settings`