Mini-AutoML dla Danych Tabelarycznych

Autorzy:

Hanna Szczerbińska

Helena Wałachowska

Paula Wołkowska

19 stycznia 2026

Spis treści

1 Wstęp

Celem projektu było stworzenie uproszczonego systemu AutoML, który umożliwiłby automatyczne wykonanie zadania klasyfikacji binarnej na dowolnym dostarczonym zbiorze danych. System miał skupiać się na skonstruowaniu i wykorzystaniu portfolio modeli.

2 Wybór modeli do portfolio

Proces wyboru modeli oraz ich hiperparametrów w celu konstrukcji portfolio przebiegał w dwóch fazach.

W pierwszej fazie dokonano wstępnej selekcji 100 modeli wraz z konfiguracjami ich hiperparametrów. Wyselekcjonowane modele pochodziły z trzech źródeł. Pierwsze 50 konfiguracji obejmowało sześć typów modeli, dla których konkretne wartości hiperparametrów zostały wylosowane z zakresów zaprezentowanych w Tabeli 1; ostatecznie do wstępnego portfolio z każdego typu modelu wybrano ustaloną liczbę wygenerowanych w ten sposób konfiguracji, również opisaną w Tabeli 1.

Typ modelu	**Zakres wartości hiperparametrów**	**Liczba wybranych modeli**
RandomForestClassifier	$n\_estimators \in \{50, 100, 200, 500\}$ $max\_depth \in \{10, 20, \text{None}\}$ (stałe: $min\_samples\_split$=2, $random\_state$=42, n_jobs=−1)	10
XGBClassifier	$n\_estimators \in \{100, 200, 300\}$ $max\_depth \in \{5, 7, 10\}$ $learning\_rate \in \{0.05, 0.1\}$ (stałe: $random\_state$=42, n_jobs=−1, $eval\_metric$="logloss")	10
LGBMClassifier	$n\_estimators \in \{100, 200\}$ $num\_leaves \in \{31, 63, 127\}$ $learning\_rate \in \{0.05, 0.1\}$ (stałe: $random\_state$=42, n_jobs=−1, $verbose$=−1)	10
CatBoostClassifier	$iterations \in \{100, 200\}$ $depth \in \{6, 8\}$ $learning\_rate \in \{0.05, 0.1\}$ (stałe: $random\_state$=42, $verbose$=False, $allow\_writing\_files$=False)	8
SVC	$C \in \{1.0, 10.0\}$ $kernel \in \{\text{"rbf"}, \text{"poly"}\}$ $\gamma \in \{\text{"scale"}, \text{"auto"}\}$ (stałe: $random\_state$=42, $probability$=True)	6
ExtraTreesClassifier	$n\_estimators \in \{50, 100, 200\}$ $max\_depth \in \{10, \text{None}\}$ (stałe: $random\_state$=42, n_jobs=−1)	6
	Suma	**50**

Tabela 1: Siatki hiperparametrów oraz liczba modeli wygenerowanych do portfoli w pierwszym etapie pierwszej fazy procesu.

W drugim etapie pierwszej fazy procesu generowania wstępnego portfolio utworzono 28 modeli wraz z odpowiadającymi im konfiguracjami hiperparametrów. Procedura doboru wartości hiperparametrów była oparta na algorytmie Random Search, obejmującym 15 iteracji, przy zastosowaniu trójkrotnej walidacji krzyżowej oraz metryki oceny jakości modeli w postaci balanced accuracy. Wybór hiperparametrów modeli przeprowadzono na podstawie dziesięciu zbiorów danych pochodzących z platformy OpenML o identyfikatorach: 31, 1464, 334, 50, 1504, 3, 1494, 1510, 1489 oraz 37. Procedura kalibracji modeli polegała na losowym wyborze czterech zbiorów danych dla każdego typu modelu przedstawionego w Tabeli 2 oraz na optymalizacji hiperparametrów dla każdego zbioru danych w oparciu o zakresy wartości zaprezentowane w Tabeli 2. W ten sposób dla każdego z siedmiu typów modeli uzyskano cztery konfiguracje hiperparametrów, które następnie zostały włączone do wstępnego portfolio.

W trzecim etapie pierwszej fazy procesu dokonano ręcznego wyboru 22 modeli wraz z odpowiadającymi im konfiguracjami hiperparametrów, na podstawie wyników zewnętrznych eksperymentów opublikowanych na platformie OpenML. W szczególności uwzględniono wielowarstwowe sieci neuronowe MLP o architekturach charakteryzujących się malejącą liczbą neuronów w kolejnych warstwach, klasyczne klasyfikatory liniowe i nieliniowe (SVM, regresję logistyczną oraz klasyfikator kNN), a także metody zespołowe

Typ modelu	Hiperparametr	Rozkład / zbiór wartości
SVM	C	uniform$(0.01, 100)$
	kernel	{rbf, linear, poly}
	γ	{scale, auto} $\cup$ uniform$(0.001, 1)$
kNN	*n_neighbors*	randint$(3, 20)$
	weights	{uniform, distance}
	metric	{euclidean, manhattan, minkowski}
	p	randint$(1, 3)$
Naive Bayes	*var_smoothing*	uniform$(10^{-12}, 10^{-6})$
Gradient Boosting	*n_estimators*	randint$(50, 300)$
	max_depth	randint$(3, 10)$
	learning_rate	uniform$(0.01, 0.3)$
	min_samples_split	randint$(2, 20)$
	min_samples_leaf	randint$(1, 10)$
CatBoost	*n_estimators*	randint$(50, 300)$
	max_depth	randint$(3, 10)$
	learning_rate	uniform$(0.01, 0.3)$
	min_data_in_leaf	randint$(1, 20)$
LightGBM	*n_estimators*	randint$(50, 300)$
	max_depth	randint$(3, 15)$
	learning_rate	uniform$(0.01, 0.3)$
	min_child_samples	randint$(5, 50)$
	subsample	uniform$(0.6, 1.0)$
AdaBoost	*n_estimators*	randint$(50, 300)$
	learning_rate	uniform$(0.1, 1.5)$

Tabela 2: Siatki hiperparametrów modeli wygenerowanych do portfolio w drugim etapie pierwszej fazy procesu.

(Random Forest, AdaBoost oraz XGBoost) w kilku wariantach parametrów. Zastosowanie ręcznie skonfigurowanych modeli pozwoliło na zwiększenie różnorodności portfolio oraz uzupełnienie go o konfiguracje sprawdzone w niezależnych badaniach empirycznych.

W drugiej fazie procesu konstrukcji portfolio, spośród wstępnie wybranych 100 modeli wyselekcjonowano 50 najlepszych konfiguracji. Selekcja została przeprowadzona na podstawie ewaluacji wszystkich 100 modeli na 15 zbiorach danych (5 małych, 5 średnich oraz 5 dużych), losowo wybranych z platformy OpenML. Każdy model generował predykcje dla wszystkich 15 zbiorów danych, a jakość predykcji oceniano przy użyciu metryki balanced accuracy. Następnie dla każdego zbioru danych utworzono ranking modeli na podstawie wartości metryki ewaluacyjnej. Dla każdego modelu obliczono średnią pozycję w rankingach uzyskanych dla wszystkich zbiorów danych. Do ostatecznego portfolio 50 najlepszych modeli zakwalifikowano konfiguracje o najwyższej średniej pozycji rankingowej. Wybrane modele wraz z odpowiadającymi im wartościami hiperparametrów zapisano w kolejności od najlepszego według średniego rankingu w pliku `models.json`.

3 Metoda selekcji modeli z portfolio dla nowego zbioru danych

Kluczowym etapem realizacji projektu była implementcja systemu Mini-AutoML z podstawowymi metodami `init, fit, predict, predict_proba`.

Metoda `fit` realizuje proces selekcji modelu (lub zespołu modeli) oraz ich końcowego dopasowania do danych treningowych. Procedura ta przebiega w kilku następujących etapach.

Na początku dane są wstępnie przetwarzne i inicjalizowany jest licznik czasu, który służy do kontrolowania budżetu czasowego całej procedury dopasowania.

W zależności od liczby obserwacji w zbiorze treningowym wybierana jest liczba foldów walidacji krzyżowej: pięć dla mniejszych zbiorów danych ($<$10000) oraz trzy dla większych. Walidacja przeprowadzana jest w sposób stratyfikowany, co zapewnia zachowanie proporcji klas w poszczególnych podziałach.

Pierwszym właściwym etapem selekcji modeli jest szybki screening. W tym kroku każdy model z portfolio trenowany jest na części danych treningowych (80%), a następnie oceniany na wydzielonym zbiorze walidacyjnym (20%) przy użyciu miary ROC-AUC obliczanej na podstawie ciągłych wyników predykcji (prawdopodobieństw lub wartości funkcji decyzyjnej). Etap ten ma na celu odrzucenie konfiguracji o najsłabszej jakości przy minimalnym koszcie obliczeniowym. Spośród wszystkich modeli do kolejnego etapu

przekazywana jest jedynie ustalona liczba najlepszych konfiguracji (domyślnie 15).

W drugim etapie przeprowadzana jest pełna ewaluacja wyselekcjonowanych modeli z wykorzystaniem walidacji krzyżowej. Dla każdego modelu trenowanego na kolejnych foldach zapisywane są predykcje walidacyjne typu out-of-fold, które następnie służą do obliczenia średniej wartości miary ROC-AUC oraz do dalszej kalibracji progu decyzyjnego. Modele oceniane są do momentu wyczerpania budżetu czasowego.

Jeżeli ensembling nie jest aktywny, wybierany jest pojedynczy model o najwyższej średniej wartości ROC-AUC. Model ten jest następnie trenowany ponownie na pełnym zbiorze treningowym. Na podstawie predykcji out-of-fold wyznaczany jest optymalny próg decyzyjny maksymalizujący wartość miary balanced accuracy, który zastępuje domyślny próg 0.5.

4 Esembling

W przypadku aktywnego ensemblingu metoda fit wybiera zbiór modeli o najwyższej jakości predykcji. Do ensemble może wejść maksymalnie pięć modeli, przy czym wprowadzono dodatkowe ograniczenie różnorodności polegające na tym, że z każdej rodziny algorytmów (np. modele drzewiaste, boostingowe, SVM) mogą zostać wybrane co najwyżej dwa modele.

Podczas walidacji krzyżowej każdy kandydat do ensemble oceniany jest analogicznie jak w przypadku selekcji pojedynczego modelu, a następnie porównywany z najgorszym modelem ze swojej rodziny aktualnie zawartym w ensemble (lub, jeśli ograniczenie dotyczące maksymalnej liczby modeli z danej rodziny nie zostało jeszcze osiągnięte, z najgorszym modelem zawartym w ensemble niezależnie od jego rodziny). Jeśli nowy model osiąga wyższą średnią wartość ROC-AUC, zastępuje on ten, z którym był porównywany.

Po zakończeniu selekcji wszystkie modele wchodzące w skład ensemble są trenowane ponownie na pełnym zbiorze treningowym. Predykcje out-of-fold poszczególnych modeli są następnie uśredniane, a na ich podstawie wyznaczany jest wspólny próg decyzyjny maksymalizujący wartość miary balanced accuracy. W fazie predykcji ensemble stosuje miękkie głosowanie (soft voting), polegające na uśrednianiu prawdopodobieństw generowanych przez poszczególne modele, a następnie porównaniu uzyskanej wartości z nauczonym progiem decyzyjnym.

5 Wnioski

Jakość predykcji możliwych do osiągnięcia przez stworzony system Mini-AutoML została oceniona na podstawie pięciu zbiorów testowych pochodzących z platformy OpenML oraz zbioru testowego X zaproponowanego w ramach zajęć. Wartości miary balanced accuracy uzyskane przy użyciu skonstruowanego portfolio modeli, a także składy ensemble wybrane dla poszczególnych zbiorów danych, przedstawiono w Tabelach 3.

Eksperymenty przeprowadzono przy ograniczeniu czasowym wynoszącym 20 minut, z ustalonym ziarnem losowości systemu AutoML równym 42 oraz przy pozostałych parametrach pozostawionych na wartościach domyślnych. Jakość predykcji uzyskanych na zbiorach danych pochodzących z platformy OpenML okazała się zadowalająca, osiągając wartości miary balanced accuracy bliskie 100% dla trzech z nich. Uzyskane wyniki są zgodne z rezultatami raportowanymi w literaturze oraz w niezależnych eksperymentach dostępnych dla tych zbiorów danych.

W przypadku zbioru testowego X uzyskana wartość miary balanced accuracy była znacząco niższa i wyniosła około 55%. Przypuszczamy, że niski wynik może wynikać z ograniczonej informatywności cech oraz specyfiki konstrukcji tego zbioru danych. Hipotezę tę potwierdza fakt, że niezależna próba predykcji przeprowadzona z wykorzystaniem systemu AutoGluon również nie pozwoliła na uzyskanie wyższej wartości miary ewaluacyjnej.

Zbiór danych	**Tryb**	**Wybrany model / skład ensemble**	**Balanced Accuracy**	**Zakres prawdopodobieństw**	**Czas dopasowania [s]**
Zbiór testowy X	Ensemble	ExtraTrees (2), RandomForest (2), SVC (1)	0.5586	[0.2952, 0.7586]	128.51
	Bez ensemble	ExtraTreesClassifier	0.5489	[0.2500, 0.8050]	137.03
credit-g (OpenML)	Ensemble	RandomForest (3), CatBoost (2)	0.7405	[0.2052, 0.9777]	330.54
	Bez ensemble	CatBoostClassifier	0.7333	[0.1355, 0.9780]	477.29
blood-transfusion-service-center (OpenML)	Ensemble	CatBoost (4), MLP (1)	0.6949	[0.0440, 0.7367]	32.96
	Bez ensemble	MLPClassifier	0.7027	[0.0107, 0.6830]	40.27
wdbc (OpenML)	Ensemble	SVC (1), ExtraTrees (1), CatBoost (1), XGBoost (2)	0.9453	[0.0004, 0.9996]	189.87
	Bez ensemble	SVC	0.9688	[0.0001, 1.0000]	106.56
monks-problems-2 (OpenML)	Ensemble	MLP (1), GradientBoosting (1), LightGBM (3)	0.9832	[0.0000, 0.9899]	108.09
	Bez ensemble	MLPClassifier	1.0000	[0.0000, 0.9977]	161.80
PhishingWebsites (OpenML)	Ensemble	GradientBoosting (1), LightGBM (4)	0.9722	[0.0000, 1.0000]	281.24
	Bez ensemble	LGBMClassifier	0.9733	[0.0000, 1.0000]	256.99

Tabela 3: Porównanie wyników systemu Mini-AutoML z włączonym ensemblingiem oraz bez ensemblingu na zbiorach testowych (OpenML oraz zbiór X).