Projekt MiniAutoML

Framework do klasyfikacji binarnej

Franciszek Filipek, Jan Skalski, Julia Girtler

Politechnika Warszawska
Wydział Matematyki i Nauk Informacyjnych

29.01.2026

Cel projektu

Celem projektu było utworzenie systemu **MiniAutoML** do **klasyfikacji binarnej** danych **tabelarycznych**, posiadającego:

- **portfolio modeli**, z którego wybierany miał być najlepszy model w procesie uzasadnionej selekcji,
- możliwość utworzenia **ensemblingu** z najlepszych modeli,
- implementację zapewniającą **powtarzalność** i kompatybilność z dowolnym tabelarycznym zbiorem danych do klasyfikacji binarnej.

Portfolio modeli

Portfolio modeli utworzono w oparciu o **screening** zbioru danych **MementoML** [1]. Zbiór zawierał wyniki miary **accuracy** dla:

- kilku wybranych algorytmów i różnych konfiguracji hiperparametrów,
- różnych zbiorów danych, na których były uczone algorytmy,
- **20** powtórzeń dla każdej konfiguracji.

W celu wyznaczenia najlepszych konfiguracji składających się na portfolio, utworzono ranking konfiguracji na podstawie wartości **score** opartej na mierze *accuracy*:

$$\textbf{score} = 0.6\,\textbf{mean} + 0.4\,\textbf{median} - \lambda\,\textbf{std} + \gamma\,log(\textbf{n\_datasets})$$

Ranking

	score	Accuracy			n_datasets	
model	**mean**	**mean**	**median**	**std**	**mean**	**n_configs**
ranger	0.932	0.895	0.895	0.102	35.088	1000
rfc	0.925	0.893	0.894	0.100	32.894	1000
xgboost	0.898	0.983	0.993	0.117	2.149	195
agtboost	0.890	0.950	0.945	0.092	1.715	473
gbm	0.889	0.863	0.871	0.114	33.754	1000
kknn	0.888	0.857	0.861	0.112	37.931	1000
svm	0.804	0.797	0.821	0.141	39.000	11
glmnet	0.745	0.752	0.690	0.143	38.034	1000
catboost	0.635	0.736	0.799	0.179	4.436	1000

Tabela: Uśredniony ranking po modelach na podstawie *mean_score*

Wybrane konfiguracje

Na podstawie tego rankingu w portfolio znalazło się **35** konfiguracji z najlepszym wynikiem **score** dla modeli:

- **xgboost** - **8** najlepszych konfiguracji,
- **gbm** - **8** najlepszych konfiguracji,
- **kknn** - **8** najlepszych konfiguracji,
- **glmnet** - **6** najlepszych konfiguracji,
- **rfc** - **5** konfiguracji inspirowanych artykułem [2].

W zbiorze MementoML najlepsze konfiguracje RandomForest posiadały zbyt dużą liczbę drzew do przeliczenia w rozsądnym czasie.

Klasa MiniAutoML

Przygotowana przez nas klasa przyjmuje na wejściu słownik modeli (wyrażonych jako konfiguracje hiperparametrów), ilość podziałów do kroswalidacji (lub "auto") oraz ziarno losowości. Wykonywane są następujące kroki:

- Inicjalizujemy pipeline który imputuje braki, zamienia zmienne kategoryczne na numeryczne (one-hot encoding) i skaluje zmienne.
- Po podaniu danych w metodzie **fit**, ewaluujemy wszystkie konfiguracje za pomocą kroswalidacji. Jeśli wybierzemy tryb "auto", liczba podziałów będzie zależeć od rozmiaru danych wejściowych:
  - Dla $n < 100 : k = 3$,
  - Dla $100 \leq n < 10000 : k = 5$,
  - Dla $n \geq 10000 : k = 10$.

  Na każdym foldzie wyliczamy wartość AUC, a potem uśredniamy po podziałach.

Klasa MiniAutoML

- Do każdego modelu dopisujemy parameter: **random_seed** aby zapewnić całkowitą replikowalność wyników
- Z każdej klasy modeli (np. rodzina regresji logistycznych) wybieramy jeden model który maksymalizuje AUC w obrębie tej rodziny
- Modele wybrane powyżej (ale nie więcej niż 5) wrzucamy do Ensembla głosującego (miękkie głosowanie) i dopasowujemy do danych
- Ensemble jest zbudowany na gotowej klasie Voting Classifier z pakietu scikit-learn
- W trakcie wywoływania metody **fit** printowane są czasy obliczania kolejnych kroswalidacji oraz czas dopasowania końcowego ensembla

Atrybuty

Po wywołaniu metody **fit** mamy dostęp do następujących atrybutów modelu:

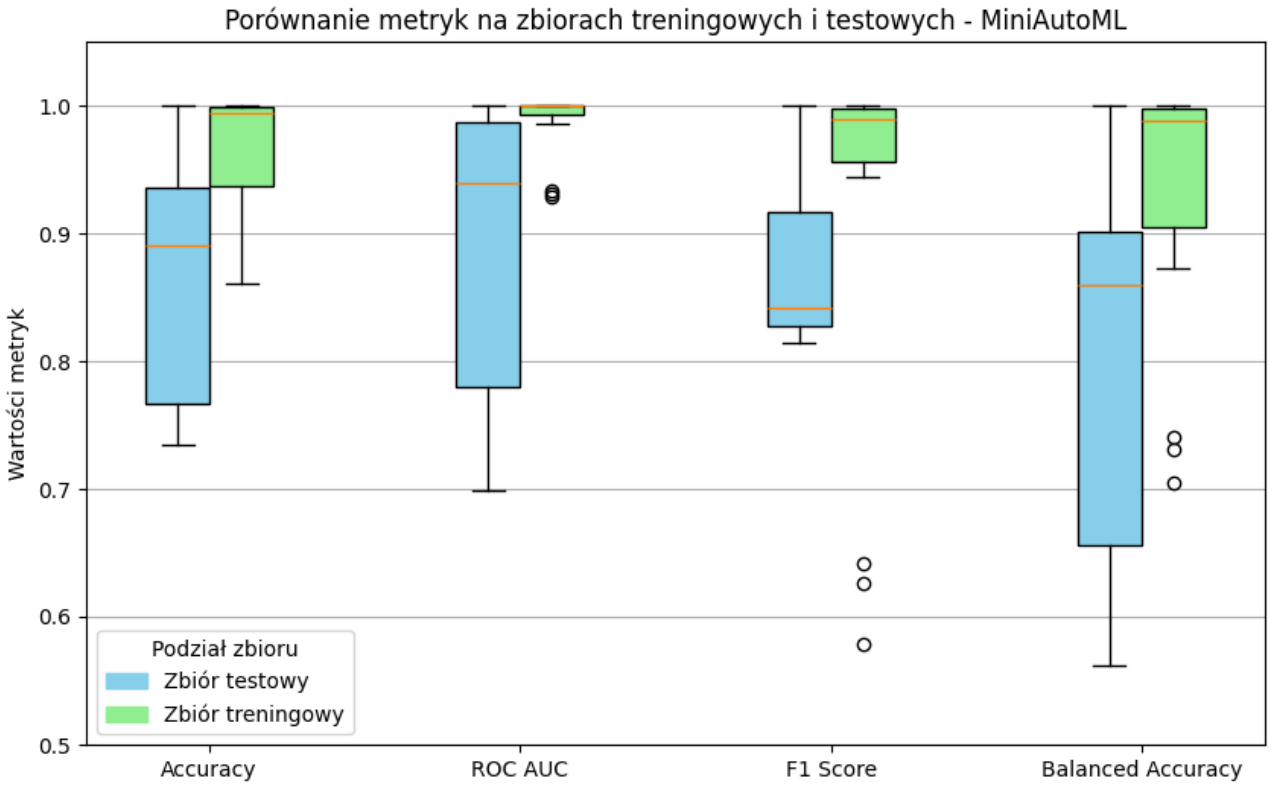
- **leaderboard** - ramka danych z wynikami kroswalidacji poszczególnych modeli uszeregowane od najlepszego
- **models_fitted** - lista dopasowanych modeli składowych w finalnym ensemblu
- **final_model** - końcowy model jako obiekt scikit-learn

Eksperymenty

Aby przetestować nasz framework wykonaliśmy testy na podanych zbiorach danych ze strony OpenM:

Zbiór danych	Wiersze	Kolumny	Proporcja klasy "1"
phoneme	5404	5	0.29
credit-g	1000	20	0.7
blood-transfusion	748	4	0.24
qsar-biodeg	1055	41	0.66
banknote-authentication	1372	4	0.44
socmob	1156	5	0.22

Eksperymenty



Framework radzi sobie bardzo dobrze na tych zbiorach, natomiast czasami ma tendencję do przeuczania się (być może wina dużych ensembli). Widać, że wariancja wyników na zbiorze testowym jest istotnie większa, co zgadza się z intuicją.

Bibliografia


Wojciech Kretowicz and Przemysław Biecek.
Mementoml: Performance of selected machine learning algorithm configurations on openml100 datasets, 2020.


Jan N. van Rijn and Frank Hutter.
Hyperparameter importance across datasets.
In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD 2018)*, pages 2367–2376. ACM, 2018.