Projekt 2

Automatyczne uczenie maszynowe

Marta Balcerzak Michał Dębski Maciej Koczorowski

Wprowadzenie

Celem projektu jest stworzenie uproszczonego systemu AutoML - MiniAutoML, który umożliwi automatyczne wykonanie zadania klasyfikacji binarnej na dowolnym dostarczonym zbiorze danych. System będzie posiadał portfolio zawierające 39 konfiguracji modeli, z których następnie wybierze najlepszy model dla dostarczonych danych treningowych. Portfolio zostanie stworzone w oparciu o wyniki kroswalidacji przeprowadzonej na różnych zbiorach danych, a następnie zapisane w pliku JSON.

1 Wybór modeli do portfolio

W celu zbudowania portfolio przygotowujemy wstępną selekcję kandydatów. Do ich wyznaczenia skorzystamy z 5-krotnej kroswalidacji na zbiorach pochodzących z platformy OpenML: adult, mushroom, phoneme, wine i electricity o numerach id, odpowiednio, 1590, 24, 1489, 187 oraz 151, dla wybranych modeli przedstawionych w poniższej tabeli. Aby uniknąć dominacji jednej rodziny algorytmów, wprowadzamy kontrolę różnorodności, nakładając ograniczenie liczby modeli z poszczególnych rodzin: 8 modeli RandomForest, 10 modeli XGBoost, 10 modeli CatBoost, 6 modeli Extra Trees oraz 5 modeli kNN. Takie liczności poszczególnych rodzin ustaliliśmy, mając na uwadze, że modele drzewiaste są bardziej uniwersalne. Do zbudowania portfolio próbowaliśmy także wykorzystać pakiet AutoGluon jako mechanizm automatycznego doboru modeli, stosując tryb zero-shot, oparty na konfiguracjach domyślnych, z dwuminutowym ograniczeniem czasowym, maksymalizujący metrykę zrównoważonej dokładności. Jednak w trakcie implementacji natknęliśmy się na problem związany z obsługą typów przez niego zwracanych. Zdecydowaliśmy się zatem zrezygnować z tego pomysłu i zrealizować podejście alternatywne.

Algorytm	Hiperparametr	Granica dolna	Granica górna
XGBoost	n_estimators	50	501
	learning_rate	0.01	0.51
	max_depth	2	10
	subsample	0.5	1
	colsample_bytree	0.5	1
	gamma	0	5
CatBoost	iterations	50	501
	learning_rate	0.01	0.51
	depth	2	10

	l2_leaf_reg	1	11
	border_count	32	255
kNN	n_neighbors	1	51
	leaf_size	10	60
	p	1, 2	
	weights	"uniform", "distance"	
Random Forest Extra Trees	n_estimators	50	501
	max_depth	3	25
	min_samples_leaf	1	11
	min_samples_split	2	11
	max_features	"sqrt", "log2", 0.2, 0.5, 0.8	

Tabela 1: Użyte zakresy hiperparametrów dla poszczególnych algorytmów

2 Selekcja modeli dla nowego zbioru danych

Aby wybrać najlepszy model na podstawie dostarczonych danych treningowych, w naszym systemie MiniAutoML będziemy wczytywać plik JSON z przygotowanym wcześniej portfolio modeli oraz plik z meta-danymi, w którym każdy wiersz odpowiada jednemu wcześniej rozwiązanemu zadaniu klasyfikacyjnemu. Dla każdego takiego zadania zapisane są w nim meta-cechy, takie jak liczba zmiennych objaśniających, liczba obserwacji, stopień niezrównoważenia klas oraz liczba cech o silnej skośności dodatniej (większej od 1) i ujemnej (mniejszej od -1), a także listy algorytmów, które znalazły się w najlepszej dziesiątce dla danego zadania. W celu znalezienia takiego meta-zbioru przeprowadziliśmy 3-krotną kroswalidację dla modeli z naszego portfolio na zbiorach pochodzących z platformy OpenML: diabetes, pizza, climate, spambase, pc1, credit, jm1, ada_prior, phoneme i ozone o numerach id, odpowiednio, 37, 1444, 1467, 44, 1068, 29, 1053, 1037, 1489 oraz 1487, maksymalizując zrównoważoną dokładność. Po przeprowadzeniu kroswalidacji dla każdego modelu otrzymaliśmy wektor wyników tej metryki z poszczególnych foldów walidacyjnych, następnie dla każdego z wektorów obliczyliśmy wartość funkcji, zdefiniowanej następująco:

$$f(x) = \bar{x} - \theta \cdot s(x),$$

gdzie $\bar{x}$ oznacza średnią z wektora, a $s(x)$ odchylenie standardowe. Funkcja ta uwzględnia zarówno średnią skuteczność modelu, jak i jego stabilność pomiędzy kolejnymi foldami walidacji. Parametr θ określa, jak silnie karana jest niestabilność modelu. Dzięki temu preferowane są modele, które nie tylko osiągają wysoką skuteczność, ale również zachowują ją w sposób powtarzalny na różnych podzbiorach danych. Jako najlepsze 10 modeli dla każdego zadania uznaliśmy te, które dawały najlepszy wynik tej funkcji.

2.1 Działanie systemu

System MiniAutoML najpierw przewiduje, na podstawie meta-zbioru, za pomocą modelu k najbliższych sąsiadów, które z modeli mają największą szansę osiągnąć wysoką jakość dla nowego problemu, analizując jego meta-cechy, i wybiera 5 modeli

o najwyższym przewidywanym prawdopodobieństwie skuteczności. Podejście to znacząco redukuje liczbę modeli podlegających kosztownej walidacji i pozwala skupić się na algorytmach najlepiej dopasowanych do charakteru danych.

Przy dalszych badaniach zajmujemy się już tylko pięcioma wybranymi wcześniej modelami. Do oceny jakości każdego z nich wykorzystujemy 3-krotną kroswalidację, a jako miarę tej jakości przyjmujemy wcześniej wspomnianą metrykę.

Zarówno podczas budowy meta-zbioru, jak i przy badaniu pięciu najlepszych modeli dla nowego zadania, poddajemy nasze dane wstępnemu przetwarzaniu. System automatycznie dzieli dane wejściowe na numeryczne i jakościowe, a następnie uzupełnia braki danych kategorycznych najczęściej występującą wartością oraz medianą w przypadku danych numerycznych, która nie jest wrażliwa na obserwacje odstające. W kolejnym kroku skaluje zmienne numeryczne w celu uzyskania jednostkowej wariancji, a zmienne kategoryczne przekształca za pomocą kodowania one-hot, tzn. zmiany każdej kategorii na oddzielną cechę binarną. Ponadto zmienne o silnie skośnym rozkładzie przekształca przy użyciu logarytmu.

Po przeprowadzeniu obliczeń wybieramy trzy modele, które uzyskały najlepszy wynik rozważanej metryki, i na ich podstawie proponujemy dwa komitety klasyfikatorów: voting oraz stacking, dla których również obliczamy wartość tej metryki przy użyciu 3-krotnej kroswalidacji. Wyniki tych komitetów dopisujemy do listy rozważanych modeli i z niej wybieramy najlepszy model lub komitet, który następnie jest trenowany ponownie na całym zbiorze danych treningowych. Dalej, tak utworzony model jest wykorzystywany do generowania predykcji metodami predict oraz predict_proba, które zapewniają możliwość uzyskania, odpowiednio, etykiet klas oraz prawdopodobieństw przynależności do klas dla nowych danych.

3 Wnioski

Po przetestowaniu naszego systemu na różnych zbiorach danych możemy zauważyć, że wśród najlepszych modeli najczęściej pojawiają się RandomForest, CatBoost oraz XGBoost, a także złożone z nich komitety, osiągają one wartości zrównoważonej dokładności powyżej 0.9. Warty uwagi jest również fakt, że podczas budowy meta-zbioru planowaliśmy poszerzenie zbioru rozważanych meta-cech oraz poszukiwanie większej liczby najlepszych modeli. Niestety, okazało się, że tego typu operacje są bardzo kosztowne obliczeniowo oraz czasowo, co pokazuje, jak warte uwagi są pakiety, takie jak AutoGluon, które umożliwiają efektywne automatyczne dobieranie modeli przy znacznie niższym nakładzie pracy i zasobów.