AutoML - Praca Domowa 2

Damian Sarna

16 stycznia 2024

1 Wstęp

Celem eksperymentu jest porównanie jakości predykcji algorytmów uczenia maszynowego budowanych w sposób ręczny z algorytmami tworzonymi za pomocą metod automatycznych.

2 Opis eksperymentu

Dostępny zbiór treningowy został podzielony na dwa zbiory: właściwy zbiór treningowy (90%), na którym dla każdego algorytmu zostanie wykonana kalibracja hiperparametrów z kroswalidacja 5-krotną oraz zbiór walidacyjny służący do obiektywnego porównania wyników (pozostałe 10%). Miarą porównania jakości algorymów jest balanced accuracy = $(\frac{TP}{P} + \frac{TN}{N})$.

Dla celów pierwszej części eksperymentu zostały wybrane cztery różne algorytmy uczenia maszynowego rozwiązujące problem klasyfikacji binarnej: XGBoost, LightGBM, RandomForest oraz KNeighborsClassifier. Ostateczny klasyfikator „ręczny" został stworzony metodą stackingu wymienionych czterech klasyfikatorów.

W drugiej części eksperymentu, skoncentrowanej na rozwiązaniu problemu klasyfikacji w sposób automatyczny, wykorzystano (lub przynajmniej próbowano, o czym później) kilka dostępnych frameworków: auto-sklearn, TabPFN, MLJAR i AutoGluon. Jako ostateczny klasyfikator w tej części wybrany został ten osiągający najlepszy wynik na zbiorze walidacyjnym.

3 Wyniki

3.1 Modele przygotowane ręcznie

3.1.1 Przygotowanie danych

Zbiór danych wykorzystany do treningu charakteryzuje się dużą liczbą kolumn (500) w porównaniu do liczby wierszy (1800), a ponieważ istnieje podejrzenie, że duża część z nich może być nieistotna, może to wprowadzać dodatkową wariancję i pogarszać dokładności osiągane przez poszczególne klasyfikatory. W celu ograniczenia liczby kolumn wykorzystano wartości feature importance dużego modelu RandomForest złożonego z 10000 drzew, dopasowanego na danych treningowych. Na podstawie analizy rozkładu wartości feature importance został wybrany ekspercko próg odcięcia równy 0.004, który został przekroczony przez 19 zmiennych. Pozostałe 481 zmiennych o niższych istotnościach zostało usuniętych.

3.1.2 Modelowanie

Dla każdego z 4 algorytmów dokonano najpierw kalibracji najważniejszych hiperparametrów:

- KNeighbors: n_neighbors
- RandomForest: n_estimators, max_depth
- XGBoost: n_estimators, max_depth
- LightGBM: n_estimators, max_depth, extra_trees

Klasyfikatory o optymalnych hiperparametrach osiągały następujące wyniki na zbiorze walidacyjnym:

Algorytm	Balanced accuracy
KNeighborsClassifier	0.9155
RandomForest	0.9058
XGBoost	0.9110
LightGBM	0.9006

Tabela 1: Wyniki algorytmów budowanych ręcznie

Do budowy ostatecznego klasyfikatora StackingClassifier użyto metody RandomForest z 250 drzewami. Miara balanced accuracy dla tego klasyfikatora na zbiorze walidacyjnym wyniosła 0.925.

3.2 Modele przygotowane automatyczne

3.2.1 Przygotowanie danych

Metody automatyczne nie wymagają żadnych kroków związanych z przygotowaniem danych, ponieważ etap preprocessingu również jest w nich automatyzowany. Do dalszych analiz wykorzystano więc pełny zbiór zawierający 500 kolumn.

3.2.2 Modelowanie

Próba skorzystania z modelu TabPFN generowała ostrzeżenie, że jego implementacja na chwilę obecną nie nadaje się do trenowania modeli na zbiorach wiekszych niż 1000 obserwacji. Ostrzeżenie to można było zignorować, co też uczyniono, jednakże wyniki zwrócone przez algorytm były bardzo słabe (dokładność na poziomie 0.67), toteż biorąc pod uwagę wspomniane ograniczenie, zrezygnowano z tego podejścia. Podczas instalacji biblioteki auto-sklearn natomiast wystąpił błąd, którego nie udało się naprawić („Getting requirements to build wheel did not run successfully."), więc i ta próba zakończyła się niepowodzeniem. Ostatecznie etap modelowania udało się przeprowadzić tylko dla frameworków AutoGluon oraz MLJAR. Preset wykorzystany w AutoGluonie to „good_quality". Wyniki prezentują się następująco:

Algorytm	Balanced accuracy
AutoGluon	0.8898
MLJAR	0.8950

Tabela 2: Wyniki algorytmów budowanych automatycznie

4 Podsumowanie

Kluczowym etapem pozwalającym na zbudowanie modeli dobrej jakości okazało się odpowiednie przygotowanie danych. Modele zbudowane na pełnym zbiorze (500 kolumn) cechowały się niską dokładnością wg miary balanced accuracy (ok. 0.7-0.8 w zależności od konkretnej metody) i dopiero usunięcie zbędnych kolumn istotnie poprawiło jakość ich wskazań. Prosta metoda heurystyczna bazująca na feature importance okazała się w tym przypadku wystarczająca. Metody automatyczne poradziły sobie z tym problemem równie dobrze, a zatem potwierdza się, że spełniają one swoje zadanie, ułatwiając pracę użytkownikowi, który musiałby poświęcić dodatkowy czas na samodzielną identyfikację problemu nadmiaru predyktorów oraz usunięcie zbędnych kolumn, a przed tym również na wybór konkretnej metody realizującej to zadanie.

Modele zbudowane w sposób ręczny ostatecznie osiągnęły nieco wyższą dokładność na zbiorze walidacyjnym (poszczególne algorytmy ręczne ok. 0.90 i ponad 0.92 po stackingu w porównaniu do ok. 0.90 dla modeli automatycznych), natomiast ze względu na dość mały rozmiar próbki walidacyjnej (200 obserwacji) trudno jednoznacznie stwierdzić, czy ta różnica jest istotna. Należy zaznaczyć, że AutoGluon nie korzystał z najdokładniejszego presetu, a MLJAR nie miał wbudowanej miary balanced accuracy, co również mogło mieć negatywny wpływ na wyniki. Z pewnością istotną przewagą modeli budowanych ręcznie jest lepsza interpretowalność oraz krótszy czas wykonywania (ok. 5 minut dla całej metody ręcznej vs. ok. 2 godzin dla metody automatycznej); z drugiej strony łączny czas pracy jest prawdopodobnie zbliżony, ponieważ wszystkie pozostałe kroki związane z pisaniem kodu algorytm wykonuje sam. Rezultaty eksperymentu sugerują, że na dzień dzisiejszy doświadczony użytkownik posiadający wyczucie w modelowaniu powinien uzyskać przynajmniej tak dobre wyniki, jak maszyna.