Porównanie użycia AutoML do klasyfikacji
PD 2 AutoML

Wojciech Kosiuk
Politechnika Warszawska

Michał Mazuryk
Politechnika Warszawska

Styczeń 2024

1 Abstrakt

Praca przedstawia podejście do klasyfikacji korzystając ze standardowej analizy i treningu modelu oraz używając frameworku AutoMLowego. Daje możliwość porównania podejść i wyników otrzymanych przez te dwa rozwiązania.

2 Zbiór danych

Zbiorem użytym w pracy był sztucznie wygenerowany zbiór *"artificial"*. Zawierał on dwie klasy, 1 i -1, równie liczne po 1000 obserwacji. Wyzwanie do treningu stanowiła liczna liczba zmiennych objaśniających (500), z których tylko część była rzeczywiście istotna. Zbiór nie zawierał braków danych.

3 Własny model

3.1 Opis podejścia

W celu opracowania naszego rozwiązania zdecydowaliśmy się na wykorzystanie modelu LightGBM, który obecnie uchodzi za jeden z najbardziej efektywnych modeli klasyfikacyjnych w dziedzinie uczenia maszynowego. Aby uzyskać z niego najlepsze wyniki połączyliśmy krok wyboru kolumn z optymalizacją hiperparametrów, a następnie wytrenowaliśmy końcowy model.

3.2 Wybór kolumn

Z 500 zmiennych obajśniających potrzebny był wybór odpowiednich kolumn do modelu końcowego. W tym celu został użyty algorytm RFE (ang. *Recursive Feature Extraction*).

Algorytm ten rekursywnie usuwa kolumny ze zbioru danych, w pierwszym etapie, wszystkie cechy są wykorzystane do zbudowania początkowego modelu, który następnie jest wykorzystany do oceny znaczenia poszczególnych cech. W

kolejnym kroku, najmniej istotna cecha (lub cechy, w zależności od konfiguracji metody) jest usuwana z zestawu danych.

Po jej usunięciu, proces jest powtarzany: model jest ponownie trenowany na nowym, zmniejszonym zestawie danych, a następnie oceniany pod kątem wydajności. Ta sekwencja trenowania modelu, oceny i eliminacji cech jest kontynuowana aż do osiągnięcia określonej liczby cech lub do momentu, gdy dalsze usuwanie cech zaczyna negatywnie wpływać na wydajność modelu.

W naszym przypadku użyliśmy go czterokrotnie, najpierw zmniejszając zbiór z 500 do 120 kolumn, a następnie do 45 i 30 zmiennych.

3.3 Wybór hiperparametrów

W trakcie działania całego algorytmu, występuje kilka etapów, podczas których dostrajane są hiperparametry do aktualnego zbioru danych.

Mianowicie, proces strojenia hiperparametrów odbywa się na ramce danych posiadających 500 kolumn (wszystkie), następnie na ramce ze 120 kolumnami, 45 i końcowo 30.

Do optymalizacji został użyty pakiet Optuna. Przeszukuje ona przestrzeń hiperparametrów (tabela 1), aby znaleźć najlepszą konfigurację dla aktualnego, mniejszego zestawu cech.

Ta procedura ma na celu zapewnienie, że każda iteracja RFE z sekcji 3.2 nie tylko eliminuje mniej istotne cechy, ale również dopasowuje model do pozostałych cech w celu maksymalizacji jego wydajności.

Tabela 1: Parametry dostrajane w Optunie dla modelu LightGBM

Parametr	Zakres
`lambda_l1`	`1e-8` do `10.0` (log)
`lambda_l2`	`1e-8` do `10.0` (log)
`num_leaves`	`2` do `256`
`feature_fraction`	`0.4` do `1.0`
`bagging_fraction`	`0.4` do `1.0`
`bagging_freq`	`1` do `7`
`min_child_samples`	`5` do `100`

Wyniki uzyskane z Optuny były rosnące przy zmniejszaniu liczby kolumn (tabela 2.)

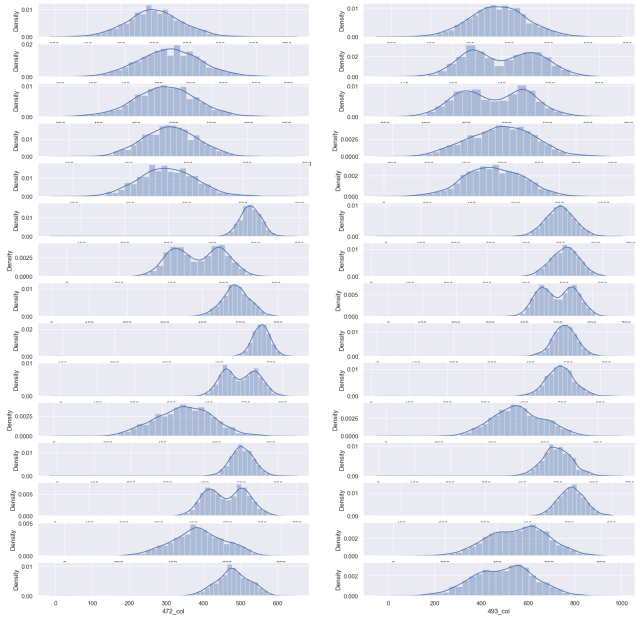
Tabela 2: Wyniki Balanced Accuracy z Optuny dla różnej liczby kolumn przy użyciu kroswalidacji

Liczba Kolumn	Wyniki Balanced Accuracy (CV = 5)	Średnia
500	[0.811, 0.830, 0.822, 0.838, 0.815]	0.823
120	[0.840, 0.838, 0.830, 0.863, 0.827]	0.839
45	[0.867, 0.833, 0.855, 0.855, 0.839]	0.850
30	[0.835, 0.850, 0.850, 0.869, 0.852]	0.851

4 Trening

Ostatecznie wybrane kolumny z procesu zostały następnie przeanalizowane aby użyć je do treningu. Utworzono macierz korelacji oraz sprawdzono wykres gęstości 1 by ocenić potencjalne transformacje. Uznaliśmy, że dodatkowe przekształcenia zbioru nie są potrzebne.

Uruchomiono ponownie wybór hiperparametrów z większą liczbą prób i na jego podstawie wytrenowano model końcowy, bazując na całym zbiorze treningowym.



Rysunek 1: Wykres gęstości ostatecznych 30 kolumn.

5 Automatyczny model

Jako automatyczny model wybrany został najlepszy uzyskany z treningu za pomocą pakietu Autogluon. Użyto metryki balanced accuracy oraz uruchomiono proces dla dwóch zestawów kolumn, na wszystkich 500 oraz na 30 najlepszych kolumnach wybranych we własnym modelu. Ich wyniki na zbiorze treningowym przedstawiają tabele 3 i 4.

6 Wyniki

Sprawdzając predykcje 3 uzyskanych modeli dla 5% zbioru testowego, dla własnego modelu LightGBM uzyskaliśmy 0.90, model z Autogluon dla 30 kolumn

Tabela 3: Wyniki dla AutoGluon na 30 najlepszych kolumnach z własnego modelu.

Model	Score_val	Pred_time_val	Fit_time
WeightedEnsemble_L2	0.8600	0.128745	8.885844
CatBoost	0.8500	0.000000	7.185618
RandomForestEntr	0.8325	0.112245	0.987415
RandomForestGini	0.8225	0.106959	1.123426
KNeighborsDist	0.8200	0.020181	0.016736
KNeighborsUnif	0.8200	0.306705	0.199934
ExtraTreesEntr	0.8150	0.063782	0.622011
ExtraTreesGini	0.8100	0.079726	0.694910
NeuralNetTorch	0.7700	0.007037	9.949798
XGBoost	0.7700	0.008978	0.137797
NeuralNetFastAI	0.6500	0.048695	3.353565

Tabela 4: Wyniki dla AutoGluon na wszystkich kolumnach

Model	Score_val	Pred_Time_Val	Fit_time
WeightedEnsemble_L2	0.8450	0.129028	29.804363
CatBoost	0.8275	0.028120	21.063402
LightGBM	0.8225	0.007975	3.461825
XGBoost	0.8225	0.033614	10.817066
LightGBMLarge	0.8025	0.078226	34.440094
LightGBMXT	0.7700	0.004280	2.274262
RandomForestEntr	0.7175	0.134500	3.542768
KNeighborsDist	0.7150	0.011391	0.083511
KNeighborsUnif	0.7150	0.015673	0.088278
RandomForestGini	0.6800	0.086032	1.911875
ExtraTreesGini	0.6800	0.180001	1.270312
ExtraTreesEntr	0.6500	0.122755	1.645213
NeuralNetFastAI	0.5900	0.016273	4.450800
NeuralNetTorch	0.5800	0.026409	7.251000

0.93, a dla wersji z 500 kolumnami 0.88.

Korelacje predykcji na zbiorze testowym między uzyskanymi wynikami to:

- model Autogluon 30, a LightGBM - 0.945
- model Autogluon 30, a 500 - 0.916
- model Autogluon 500, a LightGBM - 0.886

Jako wybrane modele uznajemy model z Autogluon dla wersji z 30 kolumnami oraz utworzony standardowo LightGBM.