Raport 2

Julita Kulesza, Agata Osmałek

16 stycznia 2024

1 Wprowadzenie

Celem projektu było zaproponowanie metody klasyfikacji, która pozwoli zbudować model o jak największej mocy predykcyjnej dla sztucznie wygenerowanego zbioru *artificial*. Zbiór ten zawierał łącznie 500 zmiennych, w czym tylko część z tych zmiennych była istotna dla klasyfikacji. Wśród danych nie było braków ani też obserwacji, które można uznać za potencjalne odstające. Z uwagi na fakt, że dane były wyłącznie numeryczne, uznałyśmy, że preprocessing danych jest zbędny. Aby jednak dostosować się do konwencji opisanej w zadaniu, w naszym rozwiązaniu etykiety -1 i 1 zostały przekształcone na odpowiednio 0 i 1.
Zbiór podzieliłyśmy w proporcji $8:2$, wykorzystując pierwszy ze zbiorów do trenowania obu typów modeli. Druga część zbioru została wykorzystana dwojako, tzn. do:

- obliczenia dokładności modelu AutoML,
- trenowania ręcznego meta-modelu.

2 Modele

2.1 Model z wykorzystaniem AutoML

AutoGluon to narzędzie do automatycznego dopasowywania modeli uczenia maszynowego. Oferuje on prosty interfejs do automatycznego dopasowywania modeli dla różnych problemów uczenia maszynowego, w tym klasyfikacji binarnej. W rozwiązaniu wykorzystałyśmy predyktor `TabularPredictor`, który służy do rozwiązywania problemów regresji lub klasyfikacji na danych tabelarycznych.
Model ten trenowałyśmy używając *balanced_ accuracy* jako docelowo optymalizowanej metryki, a pozostałe hiperparametry miały wartości domyślne.

2.2 Model ręczny

Pierwszym etapem w budowie modelu było zaadresowanie problemu nieistotnych zmiennych. W tym celu wykorzystałyśmy metodę selekcji zmiennych `Boruta`. Wybór tej metody nie był do końca przypadkowy, jako że jest to polska metoda selekcji. Niżej pokrótce opisujemy schemat jej działania.

- Dla każdej zmiennej w zbiorze danych tworzone są ich spermutowane kopie, które reprezentują losowy szum. Te zmienne są potem używane jako punkt odniesienia do porównania istotności zmiennych rzeczywistych.
- Następnie trenowany jest model uczenia maszynowego (my wybrałyśmy las losowy) na podstawie oryginalnych zmiennych i zmiennych pomocniczych.
- Istotność zmiennych oceniana jest na podstawie tego, jak bardzo przewyższają one istotność zmiennych pomocniczych.

- Zmienne są oceniane i oznaczane jako istotne lub nie na podstawie pewnego ustalonego progu istotności.

Celem ręcznego zbudowania klasyfikatora o dużej mocy predykcyjnej, zbadałyśmy zachowanie trzech popularnych modeli do klasyfikacji. Każdy z wymienionych dalej modeli trenowałyśmy zarówno na całym zbiorze danych jak i podzbiorze istotnych zmiennych, aby ocenić poprawę wynikającą z selekcji zmiennych.

2.2.1 Random Forest

Optymalny model, tak jak w przypadku pozostałych metod, został wybrany w toku pięciu powtórzeń walidacji 10-krotnej. W przeszukiwaniach po siatce parametrów

$$\begin{aligned}\text{param\_grid\_RF} = \{ & \\ & \text{'max\_depth'} : [3, 10], \\ & \text{'n\_estimators'} : [10, 100, 200], \\ & \text{'max\_features'} : [1, 4, 7] \\ \}, &\end{aligned}$$

jak i siatkach dla pozostałych modeli, wykorzystałyśmy metodę `GridSearchCV`. Bogatsze w doświadczenie z pierwszego projektu, zdecydowałyśmy się na mniejsze siatki.

2.2.2 XGBoost

Przeszukiwania pod kątem optymalnego klasyfikatora odbywały się na siatce

$$\begin{aligned}\text{param\_grid\_XGB} = \{ & \\ & \text{'max\_depth'} : \text{range}(2, 10, 3), \\ & \text{'n\_estimators'} : \text{range}(60, 200, 70), \\ & \text{'learning\_rate'} : [0.1, 0.01, 0.05] \\ \}. &\end{aligned}$$

2.2.3 LightGBM

Poniżej znaduje się siatka dopuszczanych parametrów dla naszego optymalnego klasyfikatora.

$$\begin{aligned}\text{param\_grid\_LGBM} = \{ & \\ & \text{'objective'} : [\text{'binary'}], \\ & \text{'boosting\_type'} : [\text{'gbdt'}], \\ & \text{'num\_leaves'} : [20, 30], \\ & \text{'learning\_rate'} : [0.05, 0.1, 0.2], \\ & \text{'n\_estimators'} : [50, 100, 200] \\ \} &\end{aligned}$$

W tym miejscu warto zaznaczyć, że `gbdt`, czyli `Gradient Boosting Decision Tree`, jest domyślnym parametrem klasyfikatora. Chciałyśmy jednak podkreślić co jest modelem bazowym w naszej implementacji.

3 Wyniki

3.1 Model z wykorzystaniem AutoML

Dopasowanie modelu z parametrami domyślnymi trwało łącznie około trzech minut. Wynik, uzyskany przy pomocy opcji `evaluate` na zbiorze testowym, wskazywał na dość wysoką dokładność modelu, tj.

$$balanced\_accuracy = 0.809882.$$

Biorąc pod uwagę wyniki na sprawdzarce, jest to wynik co najmniej pesymistyczny.

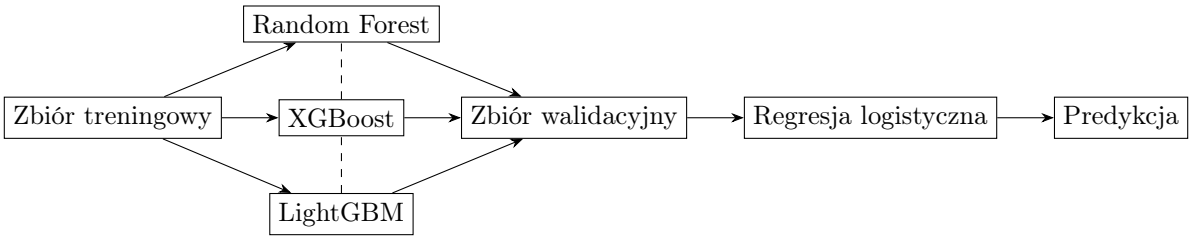
3.2 Model ręczny

Dokładność klasyfikacji, na potrzeby naszej oceny, liczyłyśmy metodą walidacji krzyżowej na zbiorze treningowym z wykorzystaniem najlepszego estymatora dla każdej z metod. Zdajemy sobie sprawę, że wynik taki jest mniej wiarygodny niż wynik na niezależnym zbiorze, jednak służyło to przede wszystkim wstępnej ocenie czy dany model powinien w ogóle wchodzić do finalnego stos-modelu (ang. *stack model*). Wyniki prezentowane w tabeli pokazują jak bardzo poprawiła się zdolność predykcyjna każdego z modeli po zastosowaniu selekcji zmiennych, w szczególności dla lasów losowych.

Tabela 1: Dokładność predykcyjna w modelach przed i po selekcji zmiennych metodą `Boruta`

Model	**Przed**	**Po**
Random Forest	0.637	0.871
XGBoost	0.817	0.866
LightGBM	0.812	0.866

Aby dalej zwiększyć moc predykcyjną modelu, stworzyłyśmy stos-model, a dokładniej model regresji logistycznej, bazujący na trzech wyżej wymienionych modelach. Poniżej prezentujemy jego uproszczony schemat.



Model ten w próbach na sprawdzarce dawał wyniki lepsze, niż jakikolwiek z modeli samodzielnie, osiągając dokładność rzędu 0.9333.

4 Wnioski

Znaczna liczba zmiennych w wyjściowym zbiorze nie tylko stwarzała problemy obliczeniowe, objawiające się przede wszystkim długim czasem trenowania każdego z modeli, ale też finalnie pogarszała ich zdolność predykcyjną. Utwierdziło nas to w przekonaniu o słuszności wyboru `Boruty` jako metody selekcji.

Wykorzystanie okrojonych siatek parametrów wejściowych do modeli mimo wszystko wiązało się z wielogodzinnymi przeszukiwaniami w celu znalezienia optymalnego estymatora.

Co było dla nas zaskakujące, selektor `Boruta` po 10 iteracjach wyróżnił zaledwie 17 zmiennych jako istotne dla budowy modelu. Albo więc zmienne istotne były rzadkie, albo metoda ta „wyłapuje" zmienne jedynie o największej mocy predykcyjnej, ignorując te, dla których moc predykcyjna jest średnia bądź mała.

Regresja logistyczna, jako samodzielny model z domyślnymi parametrami, okazała się mieć moc predykcyjną porównywalną do monety, dlatego pomijamy ją w naszym rozwiązaniu. Niemniej, pozwoliła ona stworzyć stos-model o lepszej mocy predykcyjnej niż którykolwiek z modeli wchodzących w jego skład.