AUTOMATYCZNE UCZENIE MASZYNOWE

HW2 — sprawozdanie

Zuzanna Glinka
Jakub Kasprzak

Spis treści

1 Wstęp

Poniższy dokument stanowi sprawozdanie z pracy domowej nr 2 z przedmiotu Automatyczne Uczenie Maszynowe. Jej celem było skonstruowanie i wytrenowanie modeli uczenia maszynowego do zadania klasyfikacji binarnej na dostarczonym zbiorze. Metryką mierzącą dokładność modelu jest balanced accuracy.

Należało przygotować model w dwóch wariantach:

- ręczny, samodzielnie dobierając typ modelu i dopasowując hiperparametry;
- wykorzystując wybrane frameworki służące do automatycznego uczenia maszynowego.

2 Przygotowanie danych

Zbiór danych posiada 500 zmiennych, wszystkie numeryczne o wartościach x przedziału [0, 1000]. Zbiór treningowy składa się z 2000 obserwacji, z których połowa należy do klasy 1, a połowa do klasy -1. Zbiór testowy zawiera 600 obserwacji.

W celu redukcji wymiarowości zastosowaliśmy dwie metody: RFE (rekurencyjna eliminacja wsteczna) oraz RFECV (rekurencyjna eliminacja wsteczna z kroswalidacją). RFECV jest metodą, która samodzielnie jest w stanie dobrać istotne zmienne. Żeby jednak skrócić czas jej wykonywania, najpierw zastosowaliśmy RFE, które wybrało połowę kolumn niosących najwięcej informacji, a dopiero później RFECV, które ograniczyło liczbę zmiennych do 17. Tak wyznaczony zbiór podzieliliśmy na próbkę treningową i walidacyjną w stosunku 70:30. Do skalowania danych użyliśmy StandardScaler z biblioteki sklearn[Pedregosa et al., 2011], z której pochodziły również obie metody selekcji zmiennych. Dane nie miały braków i nie wymagały żadnych dodatkowych działań.

3 Wyniki eksperymentu

3.1 Modele przygotowane ręcznie

Dla ręcznej metody klasyfikacji testowaliśmy modele takie jak: ExtraTreesClassifier, RandomForestClassifier,GradientBoostingClassifier, SVC z biblioteki sklearn[Pedregosa et al., 2011] oraz Catboost[Dorogush et al., 2018]. W celu reprodukowlności wyników, dla wszystkich modeli stosowaliśmy random_state = 1111. Wybrane przez nas ostatecznie modele mają następujące parametry:

1. ExtraTreesClassifier
   - n_estimators = 80
   - min_samples_split = 2
   - citerion = "gini"
2. RandomForestClassifier
   - n_estimators = 70
   - citerion = "gini"

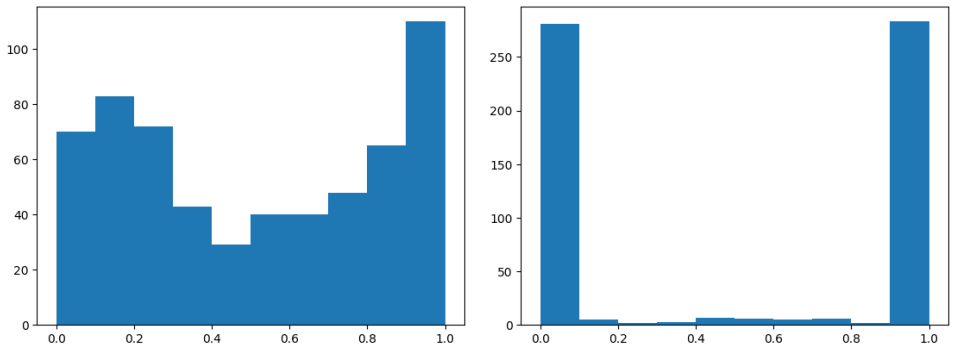
3. GradientBoostingClassifier
    - n_estimators = 100
    - learning_rate = 1
    - max_depth = 15
    - min_samples_split = 3
    - min_samples_leaf = 2
4. Support Vector Machine
    - wykorzystywaliśmy domyślne parametry
5. Catboost Classifier
    - iterations = 250
    - learning_rate = 0.2
    - depth = 10

Modele ocenialiśmy korzystając z metryk f1 oraz balanced accuracy. Otrzymane przez nie wyniki znajdują się w tabeli 1. Jak można w niej zobaczyć, najlepsze wartości metryk otrzymał model ExtraTreesClassifier uzyskując odpowiednio 0.897 i 0.898 dla f1 i balanced accuracy. Ewaluacja modeli była przeprowadzana na wydzielonym wcześniej zbiorze walidacyjnym. Mimo że dla większości modeli dobraliśmy pewien zestaw najlepszych hiperparametrów, ich dopasowywanie nie poprawiało jakości modeli w znaczący sposób.

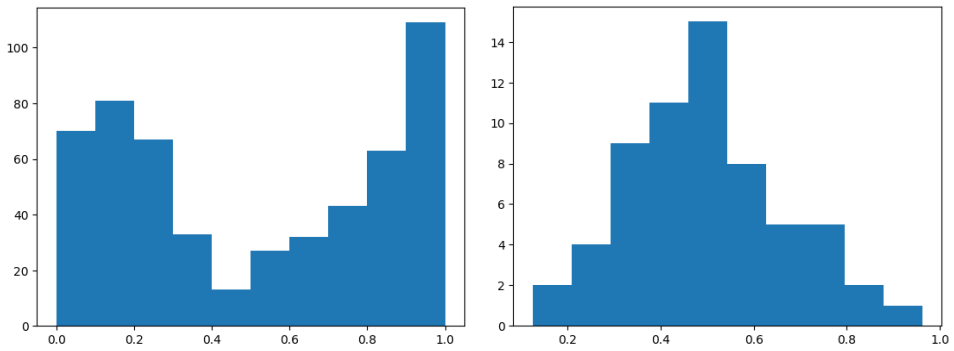
	Balanced accuracy	F1 score
Extra Trees Classifier	0.898	0.897
SVC	0.831	0.835
Gradient Boosting Classifier	0.890	0.888
Random Forest Classifier	0.886	0.885
Cat Boost Classifier	0.896	0.893

Tabela 1: Wyniki testowanych modeli

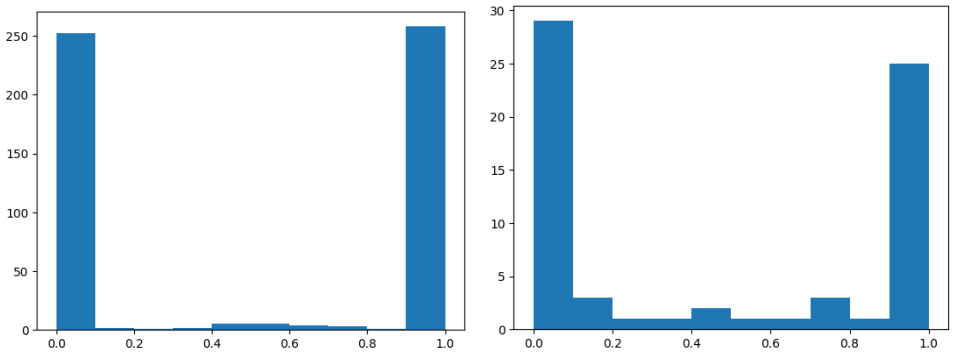
Porównywaliśmy również poprawność przewidywania klas z prawdopodobieństwem przypisania do danej klasy. Sprawdzając rozkład prawdopodobieństw zauważylismy, że rozkład dla etykiet przewidzianych za pomocą ExtraTreesClassifier jest dość równomierny, podczas gdy dla modelu GradientBoostingClassifier rozkład prawdopodobieństw skupia się wokół wartości bliskich 0 i 1 (rys. 1). Postanowiliśmy sprawdzić rozkłady tych prawdopodobieństw w przypadkach dobrego i złego przewidzenia przynależności do klasy. Okazało się, że model Gradient Boosting jest zawsze bardzo pewny swoich decyzji, niezależnie od tego, czy są słuszne, czy nie, natomiast model Extra Trees podejmuje złe decyzje głównie wtedy, kiedy prawdopodobieństwo przynależenia do założonej klasy jest bliskie $\frac{1}{2}$. W związku z tym drugi z tych modeli wydaje się być dużo bardziej wiarygodny, szczególnie gdybyśmy chcieli wykorzystać go produkcyjnie.



Rysunek 1: Rozkłady prawdopodobieństw dla klas przewidzianych odpowiednio przez ExtraTreesClassifier, GradientBoostingClassifier



Rysunek 2: Rozkłady prawdopodobieństw odpowiednio poprawnie i niepoprawnie przewidzianych klas dla modelu ExtraTreesClassifier



Rysunek 3: Rozkłady prawdopodobieństw odpowiednio poprawnie i niepoprawnie przewidzianych klas dla modelu GradientBoostingClassifier

3.2 Modele z wykorzystaniem frameworków AutoMLowych

Do rozwiązania z wykorzystaniem metod AutoML wykorzystaliśmy pakiety Autogluon[Erickson et al., 2020] i dwie wersje pakietu AutoSklearn[Feurer et al., 2020]. Z powyższych rozwiązań najlepszy wynik osiągnął AutoGluon (tabela 2), który na naszym zbiorze walidacyjnym uzyskał wynik odpowiednio 0.897 i 0.898 dla miar balanced accuracy i f1. Najlepszym modelem wewnątrz Autogluona okazał się WeigthedEnsemble_L2, czyli ensemble wielu modeli na ostatniej war-

stwie multi-layer stackingu.

	Balanced accuracy	F1 score
Autogluon	0.897	0.898
Autosklearn2	0.884	0.883
Autosklearn	0.888	0.891

Tabela 2: Wyniki testowanych frameworków AutoMLowych na zbiorze walidacyjnym

4 Wyniki na 5% zbioru testowego

Weryfikowaliśmy wyniki wybranych przez nas najlepszych modeli z użyciem udostępnionej aplikacji, dzięki której mogliśmy sprawdzić otrzymywane przez nie wyniki na 5% próbki testowej. Oba z wybranych przez nas modeli uzyskały w tym teście balanced accuracy na poziomie 0.9667, co oznacza, że zadziałały błędnie tylko dla jednej z trzydziestu obserwacji obecnych w próbce.

5 Wnioski

Jak widzimy w powyższych tabelach, wyniki dla modelu przygotowanego ręcznie i modelu z wykorzystaniem frameworka AutoMLowego są bardzo podobne. Dla tych danych ręczne przygotowanie zwraca niewiele większą najlepszą wartość miary zbalansowanej dokładności niż najlepsza miara dla rozwiązania AutoML. Jak widać, przy odpowiednim wstępnym przygotowaniu danych, zarówno framework AutoML, jak i ręczne dobieranie modelu i hiperparametrów mogą zwrócić podobne wyniki. Jest to jednak jedynie sztucznie wygenerowany zbiór, nie można zatem uogólniać wniosków wyciągniętych na podstawie przeprowadzonego na nim eksperymentu na rzeczywiste problemy. Pokazuje to jednak, że frameworki AutoML przynajmniej mają potencjał i, szczególnie dla osób, które nie są głębiej zapoznane z tematyką uczenia maszynowego, mogą być bardzo użytecznym narzędziem, dzięki któremu stworzą modele wystarczające na swoje potrzeby samodzielnie i mniejszym nakładem pracy.

Literatura

[Dorogush et al., 2018] Dorogush, A. V., Ershov, V., and Gulin, A. (2018). Catboost: gradient boosting with categorical features support.

[Erickson et al., 2020] Erickson, N., Mueller, J., Shirkov, A., Zhang, H., Larroy, P., Li, M., and Smola, A. (2020). Autogluon-tabular: Robust and accurate automl for structured data.

[Feurer et al., 2020] Feurer, M., Eggensperger, K., Falkner, S., Lindauer, M., and Hutter, F. (2020). Auto-sklearn 2.0: Hands-free automl via meta-learning. *arXiv:2007.04074 [cs.LG]*.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J.,

Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.