[HW2] - metody klasyfikacji

Dominik Kędzierski 313488

Styczeń 2024

1 Wstęp

Celem zadania było zaproponowanie metody klasyfikacji, która pozwoli zbudować model o jak największej mocy predykcyjnej. Do dyspozycji był sztucznie wygenerowanym zbiór danych artificial, w którym zostały ukryte istotne zmienne. Należało dokonać klasyfikacji do dwóch klas. Dokładność modelu należało zmierzyć za pomocą miary zrównoważonej dokładności balanced accuracy.
Modele należało przygotować w dwóch wariantach:

- wykorzystując poznane frameworki AutoMLowe
- ręcznie, czyli wybrać rodzaj modelu, hiperparametry

2 Opis danych oraz preprocessing

Dane treningowe zawierały 2000 obserwacji, każda obserwacja składała się z 500 kolumn zmiennych numerycznych. Ponieważ, ilość kolumn w stosunku do liczby obserwacji była duża, zdecydowałem się zmniejszyć wymiar danych za pomocą kilku kolejnych sposobów:

- Redukcja kolumn o dużej korelacji
- Wybór zmiennych za pomocą random forest
- Reduckja wymiarowości za pomocą PCA

Pierwszym etapem była redukcja ilości kolumn za pomocą korelacji. W tym celu dla wszystkich kolumn została obliczona macierz korelacji. Następnie z pary kolumn o dużej korelacji (większej niż 0.7), została usuwana jedna z nich. W ten sposób udało się zmniejszyć liczbę kolumn do 489. Kolejnym krokiem był wybór kolumn przy pomocy

Random forest z parametrem określającym ilość drzew ustawionym 100. Następnie zostały wybrane te kolumny które zostały zaklasyfikowane jako ważne dla tego modelu. Po tym kroku ilość zmiennych zmalała do 187.

Ostatnim krokiem była redukcja wymiarowości przy użyciu PCA. Po wytrenowaniu oraz transformacji danych, ostateczna ilość zmiennych wyniosła 50.

W celu stworzenia zbioru walidacyjnego do testowania modeli, 0.33 z danych treningowych została wyodrębniona jako dane walidacyjne za pomocą których sprawdzane była jakoś modeli.

3 Modele AutoML

W celu budowy modeli wykorzystałem popularny framework AutoML - FLAML Na każdym z nich modele zostały wytrenowane dla danych z każdego etapu w trakcie redukcji wymiaru zmiennych.

Wszystkie modele zostały wytrenowane z ustawionym parametrem *ensemble=True*, dzięki czemu każdy model był ostatecznie modelem łączonym. Drugim argumentem jaki został ustawiony dla każdego modelu był limit czasowy wynoszący 10 minut.

Wyniki uzyskane przez wytrenowane modele są następujące:

Model	balanced accuracy zbioru treningowego	balanced accuracy zbioru walidacyjnego
Model bez preprocessingu danych	0.7873315868263473	0.5168601234205114
Model po redukcji za pomocą korelacji	0.7574360208155119	0.5152439024390244
Model po redukcji dodatkowo za pomocą Random forest	0.6985047761619618	0.5545474581251837
Model po redukcji dodatkowo za pomocą PCA	0.99030510407755928	0.7000624449015574

Tabela 1: Wartości metryki dla poszczególnych modeli oraz zbiorów

Jak widać, model którego dane miały najmniejszą ilość kolumn, tzn były po redukcji kolumn za pomocą korelacji, wyborze istotnych zmiennych za pomocą Random forest oraz redukcji wymiarów przy pomocy PCA uzyskał najlepszą wartość zarówno na zbiorze testowym jak i walidacyjnym. Różnice między tymi zbiorami są jednak bardzo duże co świadczy o przetrenowaniu modelu, jednak ciężko było temu zapobiec.

4 Modele ręczne

Do wytrenowania modeli ręcznych użyłem sieci neuronowej stworzonej przy pomocy frameworku Tensorflow oraz algorytmu XGBoost. Na podstawie wniosków wyciągniętych z trenowania modeli przy użyciu AutoML do treningu obu algorytmów użyłem tylko zbiorów danych po wszystkich przekształceniach zmniejszających ich wymiar.

4.1 Sieć neuronowa

W trakcie przeprowadzania eksperymentów, żadna architektura sieci neuronowej nie uzyskała dobrych wyników. Sieci bardzo szybko przetrenowywały się osiągając accuracy na poziomie zbliżonym do 1 na zbiorze treningowym, przy czym wartość metryki dla zbioru walidacyjnego osiągała maksymalnie wartość w okolicach 0.6.

4.2 XGBoost

W celu znalezienia optymalnych hiperparametrów algorytmu posłużyłem się metodą random search. Algorytm działał na 100 iteracjach przy 5 krotnej kroswalidacja w każdej iteracji. Siatka parametrów użytych w trakcie poszukiwania przedstawiona jest w poniższej tabeli.

Parametr	Wartości
max_depth	3, 4, 5, 6, 7, 8, 9
learning_rate	0.1, 0.01, 0.05, 0.001
n_estimators	100, 150, 200, 250, 300, 350
min_child_weight	1, 2, 3

Tabela 2: Siatka parametrów użytych w random search

W przeciwieństwie do sieci neuronowej algorytm XGBoost osiągnął dosyć dobry wynik. Wynosił on odpowiednio:

- Zbiór treningowy - 1.0
- Zbiór walidacyjny - 0.702982662356744

5 Wnioski i podsumowanie

Na podstawie wyników wytrenowanych modeli, do predykcji danych testowych został użyty model AutoML wytrenowany na podstawie danych po wszystkich przekształceniach oraz model XGBoost wytrenowany ręcznie również na podstawie danych po wszystkich przekształceniach.

Niemniej w obu modelach widać dosyć duże przetrenowania. Modele te bardzo dobrze radzą sobie z danymi treningowymi, jednak na danych walidacyjnych osiągają już dużo niższe wyniki. Może to wynikać zarówno z niedostatecznej redukcji wymiarów, wybrania tylko kolumn które były ważne, jak również może być to spowodowane niewystarczającą licznością próbki treningowej.