Zadanie 6 - raport

Michał Sobczak

15 June 2025

W moim rozwiązaniu skorzystałem z frameworku **LightGBM**, a dokładnie z funkcji **LGBMRegressor**. Jest to stosunkowo nowa biblioteka, która dobrze odnajduje się podczas trenowania modeli. Podaję link do oficjalnego github, skąd głównie brałem informacje. Framework ten działa stosunkowo szybko i nie zużywa dużo pamięci, dlatego byłem w stanie trenować moje modele na własnym komputerze. Doczytałem również, że **lightGBM** dobrze sobie radzi w przypadku gdzie mamy dużo danych - co ma szczególne znaczenie w naszym projekcie, ponieważ pracujemy na dużej liczbie próbek.

Na samym początku próbowałem użyć techniki **PCA**, by zredukować liczbę danych, lecz negatywnie wpływało to na mój wynik, stąd w dalszej części zrezygnowałem z tego pomysłu.

Teraz po krótce skupimy się na użytych parametrach i ich wartościach.Wykonałem odpowiednią siatkę parametrów (grid search) i na jej podstawie wybrałem najlepsze wartości:

- **n_estimators** - liczba drzew trenowanych przez model. Intuicyjnie warto ustawić ten parametr na dużą wartość - lecz zbyt duża liczba grozi przeuczeniem - wybrałem 2000.
- **subsample** - określa, jaki procent próbek treningowych jest wykorzystywane do trenowania każdego drzewa. Zauważmy, że w przypadu gdy ustawimy ten parametr na wartość równą 1, to możemy mieć do czynienia z przeuczeniem, bo model będzie się uczył cały czas na tych samych danych, więc ustawienie tego na 0.8 dodaje pewnej losowości.
- **learning_rate** - współczynnik uczenia. Domyślna wartość wynosi 0.1, dlatego testowałem kilka zbliżonych wartości. Najoptymalniejsza : 0.05
- **num_leaves** - maksymalna liczba liści w pojedynczym drzewie decyzyjnym. Domyślna wartość wynosi 31 - trenowałem zbliżone wartości, w moim przypadku najoptymalniejszą wartością okazało się 35.
- **max_depth** - maksymalna głębokość drzewa. Pomaga ograniczyć złożonośc modelu. Najlepsza wartość okazała się 15.