Projekt I - Zaawansowane Metody Uczenia Maszynowego - Klasyfikacja binarna

Wojciech Celej

1. Użyte narzędzia

Do realizacji projektu użyto języka *Python* wraz z pakietami:

- *sklearn* - funkcje do preprocessingu danych, metryk, implementacja Random Forest, CV wraz z GridSearchem
- *category_encoders* - paczka zawierająca funkcję do enkodowania kolumn kategorycznych (testowano *BinaryEncoder*, *WOEEncodder*, *TargetEncoder*)
- *shap* - próba wyjaśnienia modelu
- *xgboost*, *LightGBM*, *CatBoost* - implementacja modeli boostingowych
- *matplotlib*, *seaborn* - wizualizacja wyników

2. Opis przetwarzania danych

a) Zabiegi wstępne

Usunięcie kolumn pustych, usunięcie kolumn kategorycznych powtarzających się, usunięcie kolumn z kategorycznych z liczbą kategorii większą niż 100. Dalsze kroki bazowały na technikach opisanych w *sklearn* link

b) Wypełnienie pustych wartości numerycznych

Brakujące wartości wypełniono medianami, dodatkowo dla każdej zmiennej numerycznej w której wystąpiły braki danych utworzono kolumnę kategoryczną informującą, czy w dla danej obserwacji wystąpił brak danych

c) Ekstrakcja cech, selekcja cech

Do ekstrakcji cech ze zmiennych numerycznych użyto algorytmów PCA i RBFSampler z pakietu *sklearn*, jednak nie odnotowano zauważalnej poprawy wyników klasyfikacji. Również ekstrakcja cech nie poprawiła ostatecznych wyników, jednakże użycie wszystkich zmiennych w znaczący sposób wypływało na szybkość treningu modelu. Stąd do poszukiwania optymalnych hiperparametrów użyto 100 najistotniejszych zmiennych (pomiaru istotności dokonano na podstawie oceny ważności zmiennych dokonanego przez *xgboost* z domyślnymi parametrami oraz metodę *TreeExplainer* z pakietu *shap*), pomysł z link

d) Zakodowanie zmiennych kategorycznych

Pierwsze podejście do tematu na podstawie artykułu: link.
Kodowanie zmiennych kategorycznych jest wymagane przez modele *xgboost* i *Random Forest*. Modele *LightGBM* i *CatBoost* radzą sobie ze zmiennymi katgorycznymi surowymi. Do zakodowania zmiennych użyto metody Weight of Evidence (WOE). Metoda ta jest prosta i skuteczna, dodatkowo nie tworzy dodatkowych kolumn jak *Binary Encoding* czy *One hot encoding*. Jej opis zaczerpnąłem z link

Przeuczenie kontrolowane jest przez dodanie szumu gaussowskiego. Testowałem również *Target Encoding* zgodnie z link1 i link2, jednak na zbiorze testowym dał on gorsze rezultaty.

e) Przeskalowanie zmiennych metodą odporną na outliery

W tym celu użyto funkcji *RobustScaler* z pakietu *sklearn*, który centruje zmienną w medianie i skaluje używając IQR.

3. Wybór modeli i hiperparametrów

W zasadzie pewniakiem do wyboru był *xgboost*. Udało mi się jednak znaleźć nowsze metody boostingowe. Ich zalety i wzajemne porównanie opisano w artykule link.
Do powyższych metod dobrano *Random Forest*, ponieważ jest to metoda popularna, a zarazem prosta i skuteczna. Wszystkie modele z powyższych umożliwiają ustawienie wag dla klas, skuteczne w przypadku problemu niezbalansowanych klas.

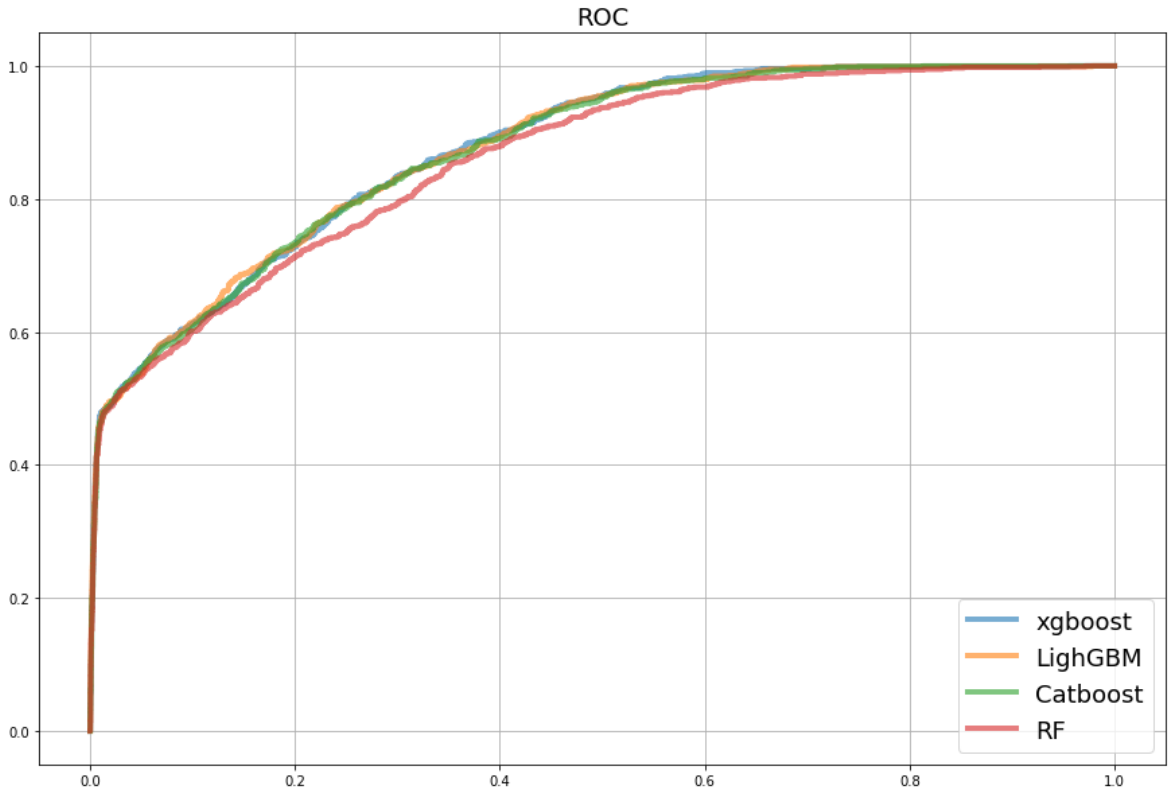
Dobór hiperparametrów przeprowadzono za pomocą *GridSearchCV* i *RandomizedSearchCV*. Jako metryki użyłem wartości AUC klasyfikatora. Metody boostingowe uczono pamiętając o wczesnym zatrzymaniu uczenia zgodnie z link

Ustalone parametry:

- *xgboost* - dokonano wyboru na podstawie artykułów link1 oraz link2
  ```
  xgb_params = { 'learning_rate': 0.01, 'n_estimators': 5000, 'subsample': 0.9,
  'max_depth': 3, 'colsample_bytree': 0.4, 'gamma': 0.5, 'reg_alpha': 0.05
  'scale_pos_weight': 12.7 }
  ```
- *LightGBM* - na podstawie dokumentacji link
  ```
  lgb_params = { 'bagging_fraction': 0.8, 'feature_fraction': 0.8, 'learning_rate':
  0.01, 'max_bin': 100, 'max_depth': -1, 'n_estimators': 5000, 'num_leaves': 20,
  'scale_pos_weight': 2 }
  ```
- *CatBoost* - parametry domyślne (`scale_pos_weight` takie jak w xgboost), surowe zmienne kategoryczne
- *Random Forest* - na podstawie link
  ```
  rf_params = { 'n_estimators': 1400, 'min_samples_split': 10, 'min_samples_leaf': 4,
  'max_features': 'sqrt', 'max_depth': 40, 'bootstrap': False, 'class_weight':
  "balanced" }
  ```

4. Wyniki

Klasyfikatory uczono na 2/3 zbioru treningowego, resztę stanowił zbiór testowy. Ocenę jakości klasyfikatorów ewaluowano na tym samym zbiorze testowym.



Policzone współczynniki jakości klasyfikacji

Model	**Dokładność**	**Czułość**	**Precyzja**	**Precyzja10%**	**AUC**
xgboost	0.799470	0.722508	0.228247	0.417865	0.874485
LightGBM	0.951136	0.467626	0.781787	0.418622	0.874592
Catboost	0.801591	0.727646	0.231221	0.414837	0.872444
RF	0.950530	0.431655	0.807692	0.409538	0.859959

Różnice między *xgboost* a *LightGBM* nie są znaczące, ale model *LightGBM* był mniej stabilny ze względu na parametr `scale_pos_weight`. Ostatecznego wyboru pomiędzy metodami uzyskano stosując 10-krotną CV. Otrzymane wyniki po uśrednieniu to:

Model	**Precyzja10%**	**AUC**
xgboost	m: 0.398504, std: 0.017017	m: 0.862823, std: 0.009737
LightGBM	m: 0.401247, std: 0.019556	m: 0.864362, std: 0.009091

Stąd przesłane wyniki pochodzą z modelu *LightGBM*.