22.7.2021

Курсовой проект

Модель определения вероятности подключения услуги

Максим Завражин

1. **Задачи:**

Определение вероятности подключения новой определенной услуги для каждой test-пары пользователь-услуга

Обзор датасета и загрузка данных:

- **data_train.csv** В качестве исходных данных доступна информация об отклике абонентов на предложение подключения одной из услуг. Каждому пользователю может быть сделано несколько предложений в разное время, каждое из которых он может или принять, или отклонить. Обучение проходит на 831 653 размеченных записей.

- **data_test.csv** Тестовый набор данных для итогового обучения модели. Содержит 71 231 записей.

- **features.csv** Отдельный нормализованный анонимизированный набор признаков, характеризующий профиль потребления абонента (весом 21 GB). Эти данные привязаны к определенному времени, поскольку профиль абонента может меняться с течением времени. Содержит 4 512 528 записей. Даты совпадают как с трейном, так и с тестом

Описание данных:

- **id** - идентификатор абонента
- **vas_id** - подключаемая услуга
- **buy_time** - время покупки
- **target** - целевая переменная, где 1 означает подключение услуги, 0 - абонент не подключил услугу.

Данные train и test разбиты по периодам – на train доступно 4 месяцев, а на test отложен последующий месяц.

---

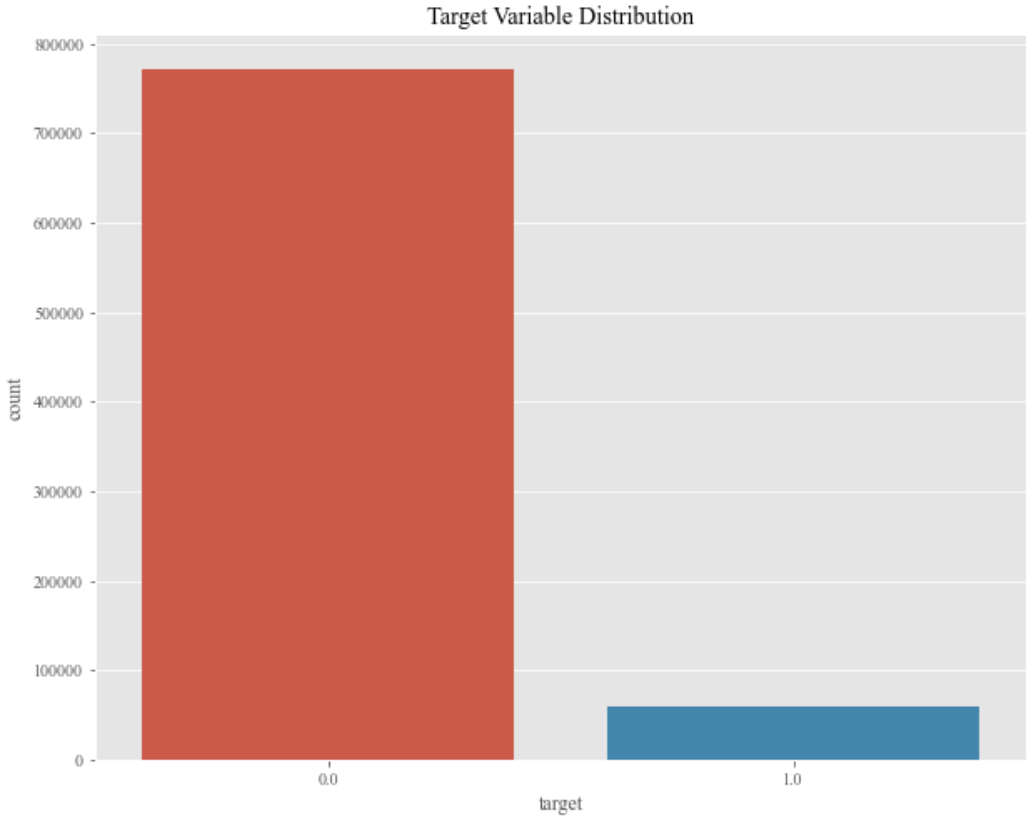
2. **Ход выполнения**

Выбор модели на начальном этапе не имел смысла т.к. на дефолтных настройках половина моделей не видела класс **1** как таковой.

Метрика **ROC-AUC** не подходит для дисбаланса классов!!!

Поэтому задачей №1 была борьба с этим дисбалансом. Выбор моделей производился на сбалансированном методом **OVERSAMPLING** датасете

Метод **SMOTE** намеренно не использовался, т.к. у test и у train сдвиг по времени и генерация новых данных на основе трена может ухудшить итоговые предсказания, хотя на валидации выглядеть лучше.



Результаты тестирования моделей (итоговый результат после выбора порога вероятности):

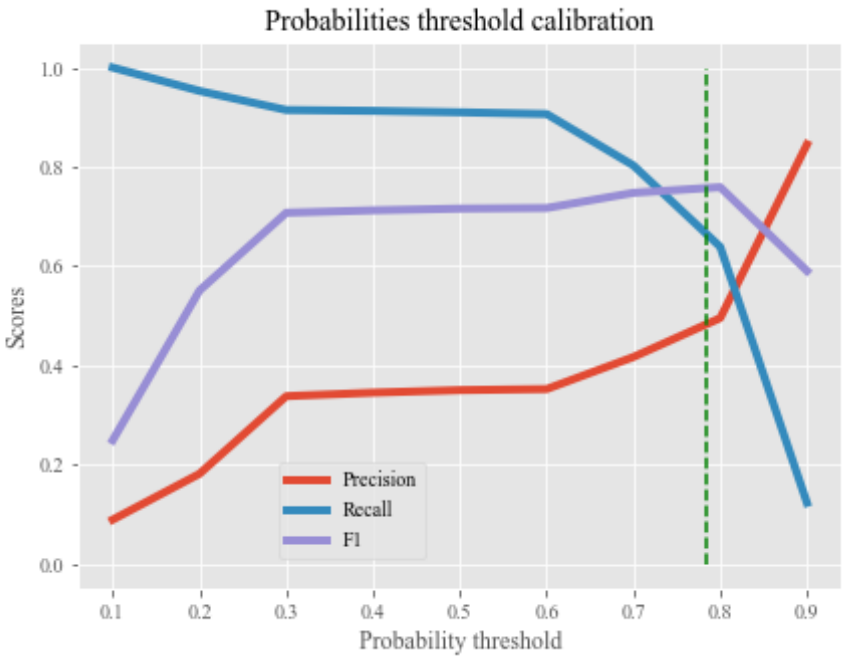
№	*Модель*	*f1-score (macro)*
1	LogisticRegression	0.62
2	RandomForestClassifier	0.748
3	CatBoostClassifier	0.755
4	LightGBMClassifier	0.755
5	***XGBoostClassifier***	***0.759***

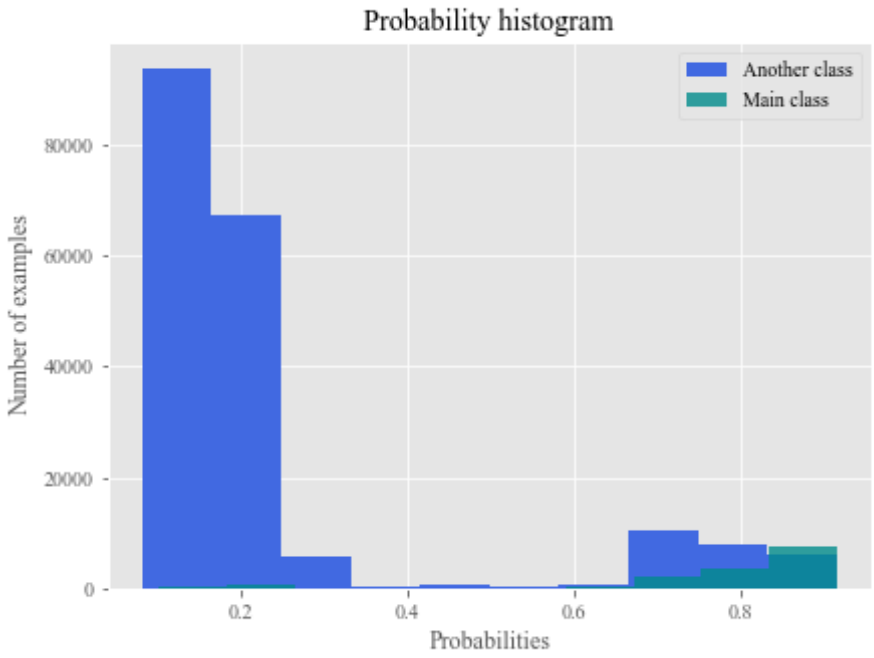
Применение **OVERSAMPLING** в конечном счете дало ровно такой же результат, как и установка параметра модели xgboost **scale_pos_weight=DISBALANCE**. Выбор, естественно, пал на второй вариант, т.к. он не увеличивает датасет.

Итоговые параметры финальной модели

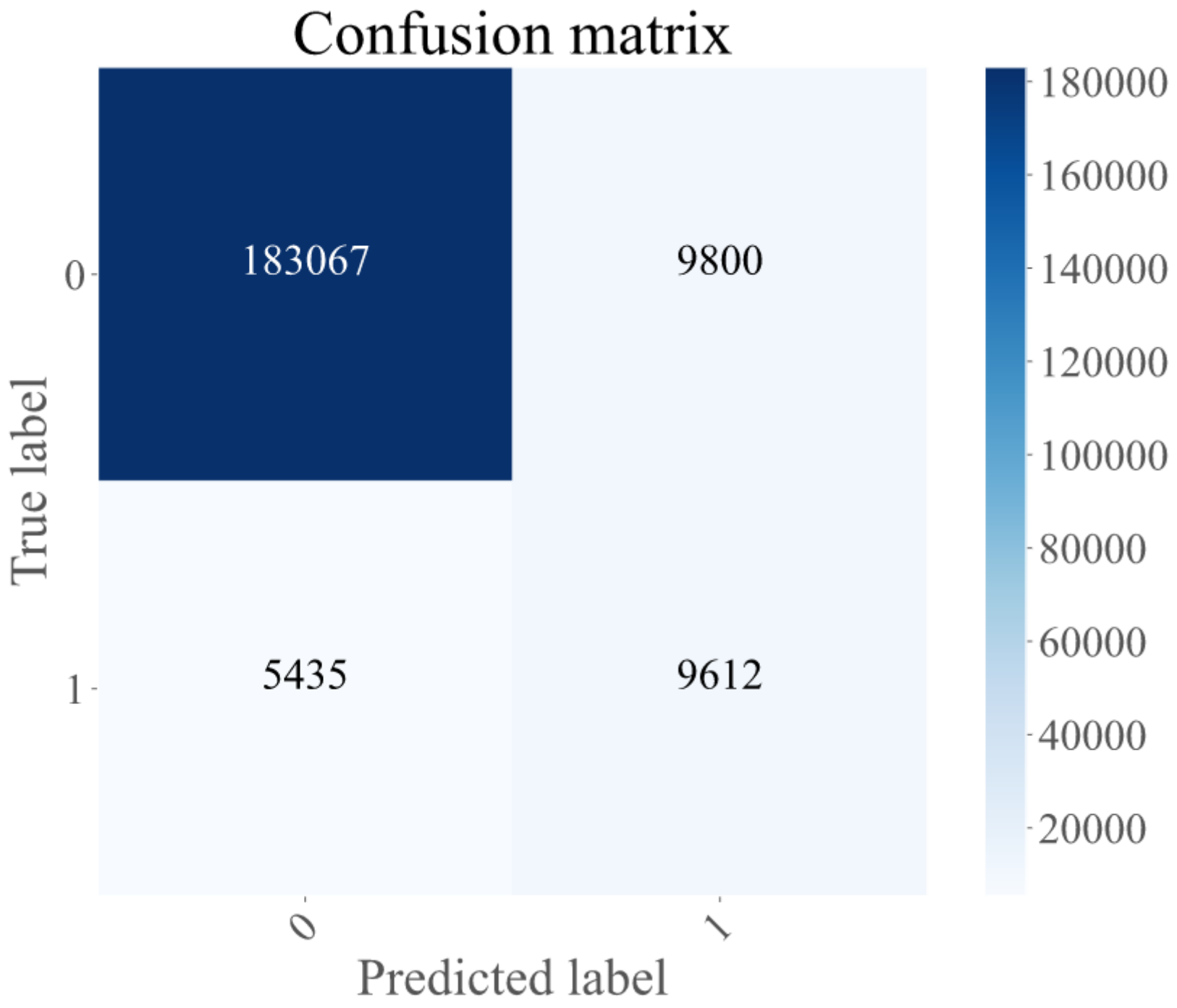
```
xgb_params = {
    "random_state": 13,
    "max_depth": 7,
    "n_estimators": 450,
    "learning_rate": 0.004,
    "reg_lambda": 0.85,
    "reg_alpha": 0.8,
    "eval_metric": "logloss",
    "scale_pos_weight": DISBALANCE,
```

3. Результаты выбора порога вероятности и итоговая матрица ошибок:





f1	precision	recall	probability
0.759	0.4952	0.6388	0.8
0.7476	0.4167	0.8027	0.7
0.717	0.3521	0.9064	0.6
0.7157	0.35	0.9099	0.5
0.7123	0.3449	0.9127	0.4
0.7074	0.3378	0.9145	0.3
0.5902	0.8453	0.1227	0.9
0.5507	0.1814	0.9529	0.2
0.249	0.0889	1.0	0.1


Confusion matrix
183067
9800
5435
9612
0
1
True label
0
1
Predicted label
180000
160000
140000
120000
100000
80000
60000
40000
20000

!!! Выбор метрики f1_macro (при постановке задачи) кажется странным. Т.к. при сильном дисбалансе ценным является минорный класс (собственно, и по логике задачи так же) и надо бы максимизировать предсказания на нем.!!!

Методы понижения размерности (PCA и пр.) не использовались, т.к. в реальных задачах необходима интерпретируемость результатов.

В результате работы сохраняется pipeline (Preprocessing+XGBoostClassifier) и модель xgboost в формате pickle.

Модель помещается в директорию **model.**

(Файл пайплайна занимает 4Гб памяти, но это удобнее, чем при появлении новой порции тестовых данных каждый раз стартовать Preprocessing с распаковкой 20Гб файла features.csv).

Для запуска predictions (согласно заданию, файл **data_test.csv** в корне, остальные – в директории **data**) используется файл **LUIDGI_pred_run.py**.

С учетом того, что в предобработке датасета используется кодирование категориальных данных средним (target) наличие файла data_train.csv тоже требуется.

Персональные предложения, с учетом большого количества данных об абонентах предлагается делать UPLIFT-моделированием (кто-то подключит и сам, а кого-то стоит простимулировать)