Курсовая работа по видеокурсу от Мегафон

Определение вероятности подключения услуг

Наталья Недельки на

Условия и предположения

- Основные условия
  - Построить модель предсказания вероятности подключения услуги vas_id пользователю id около времени buy_time,
    - с учетом анонимизированного время-зависимого профиля пользователя features.csv
  - Модель в формате pickle
  - На входе data_test.csv, на выходе answers_test.csv с предсказаниями и вероятностями
  - Уделить внимание внедрению модели в production
    - Использование luigi, контейнеризация для учета зависимостей от пакетов.
- Предположения
  - Модель должна работать с произвольными наборами, но предсказания делаются по признакам из профиля пользователя около времени предсказания,
    - поэтому огромный файл features.csv надо использовать и привязывать во время предсказания.
    - Непонятно, как предсказуемо обрезать файл, с учетом общего случая.

Модель и обучение

- В качестве модели был выбран CatBoostClassifier из catboost от Yandex.
  - Хороший предыдущий опыт работы с этим пакетом
  - Наилучшие результаты часто достигались именно на нем, с разумным временем обучения
  - Надежность работы (низкая вероятность отказов во время обучения и т.д.)
- Сравнения с другими моделями не проводилось (попробую позже)
- Инжиниринг признаков:
  - Выделение дополнительных категориальных признаков с ограниченным числом значений (напр. <20) не приводило к улучшению метрик, но существенно удлиняло обучение
  - Остановилась на единственном cat-признаке vas_id с явным преобразованием OneHotEncoding
- Оптимизация параметров:
  - Пробовала оценивать важность признаков как из CatBoostClassifier, так и SelectFromModel(LogisticsRegression).
    - Заметное число признаков имело 0-ю или очень низкую важность
    - Но использование обоих вариантов исключения признаков не улучшали статистику результата, при этом практически не влияя на время обучения.
    - Поэтому для простоты остановилась на полном наборе
- Валидация
  - В силу временной зависимости набора для валидации модели использовались последние 10% тренировочного набора. Кросс-валидация не использовалась
  - Финальная модель обучалась на всем тренировочном наборе
- Весь код обучения - в utils.py и notebook.ipynb

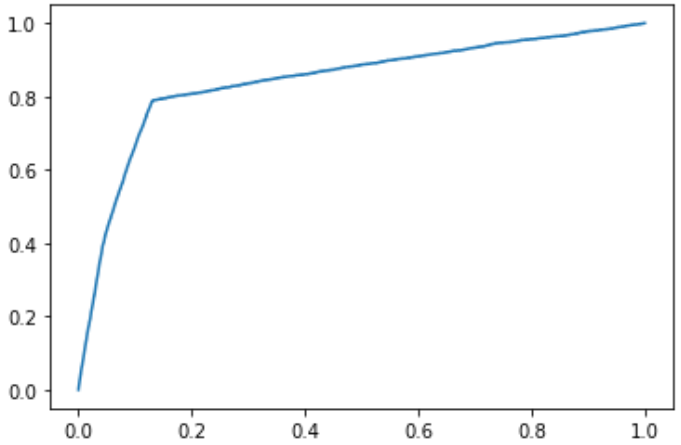
Результаты модели на валидации

Основные метрики на валидации

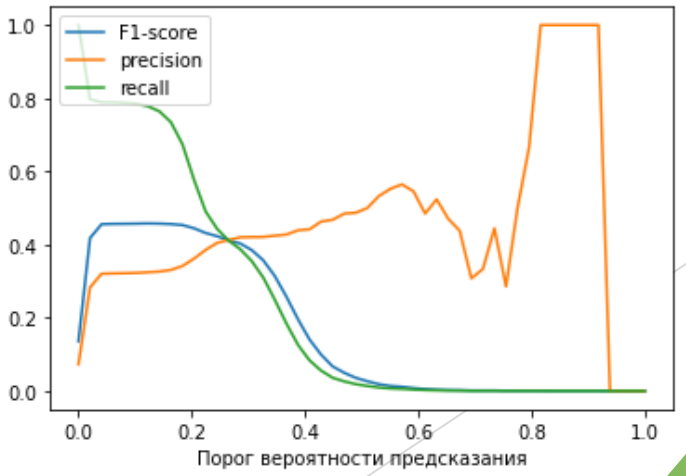
```
y_pred = y_proba > 0.12244897959183673

F1-score: 0.4578827408952039

F1-score-macro: 0.69058119173769

ROC-AUC-score: 0.8431906123005778

              precision    recall  f1-score   support

         0.0       0.98      0.87      0.92     77100
         1.0       0.32      0.78      0.46      6066

    accuracy                           0.87     83166
   macro avg       0.65      0.83      0.69     83166
weighted avg       0.93      0.87      0.89     83166
```

- Порог предсказания получился довольно низким, иначе recall (на 1) был очень плохим.
  - В финальной модели задан порог 0.2 по умолчанию, переопределяется параметром threshold
- ROC-кривая выглядит неплохо, ROC-AUC-score 0.84

ROC-кривая



Precision, recall, f1-score в зависимости от порога



Инжиниринг модели и внедрение

- Пайплайн модели реализован с помощью sklearn.pipeline и т.п.
  - См. make_feature_pipeline в src/utils.py
- Загрузка и привязка огромной таблицы features.csv выполнялась с пакетом dask.dataframe
  - Выполнялась в 2 этапа:
    - Составление "индекса" с id и feature_buy_time (load_features, LTaskIndexFeatures), достаточного для поиска ближайшего времени через merge_asof
    - Реальный поиск через pandas.merge_asof и мерж через dask.dataframe.merge (add_features, LTaskAddFeatures)
  - Устойчиво работала в контейнере на VM Ubuntu с 4GB памяти.
- Успешно задействован пакет luigi, очень удобен из-за автоматического отслеживания зависимостей и промежуточных артефактов.
  - Задачи хорошо параметризованы, параметры документированы (--help или README.md), что должно облегчить внедрение.
  - См. Src/luigi_pipeline.py
- Независимость от пакетов обеспечена через docker.
  - Зависимости в requirements.txt
  - Подробности см. в README.md
  - Входной и выходной файл сохраняются по умолчанию в директории data, для удобства отображения в хост-систему.
- Запуск модели
  - Run.sh [параметры согласно --help или README.md]
- Модель тестировалась на Windows 10 с 16GB и Ubuntu VM (VirtualBox) с 4GB.

Интерпретации предсказаний и формирование предложений

- Можно воспользоваться следующей моделью издержек:
  - Недополученная выгода от упущенных клиентов (false negatives)
  - Затраты на донесение отвергнутых предложений (false positives)
- Минимизируем miss_cost * fn + promo_cost * fp
- Поскольку представляется, что miss_cost >> promo_cost, то кажется разумным минимизировать fn, то есть улучшать recall (чувствительность, покрытие).
  - В этом плане выбор низкого порога вероятности для улучшения recall представляется верным.
  - При этом надо не допускать слишком низкой метрики precision, поскольку promo_cost все же материален.