Курсовая работа

Видеокурс от Megafon (GeekBrains)

Юдин Дмитрий

2023г.

Задача:

- Построить алгоритм, который для каждой пары пользователь-услуга определит вероятность подключения услуги.

Данные:

- 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.
- Отдельным набором данных будет являться нормализованный анонимизированный набор признаков, характеризующий профиль потребления абонента. Эти данные привязаны к определенному времени, поскольку профиль абонента может меняться с течением времени.
- target - целевая переменная, где 1 означает подключение услуги, 0 – абонент не подключил услугу соответственно.
- buy_time - время покупки, представлено в формате timestamp, для работы с этим столбцом понадобится функция datetime.fromtimestamp из модуля datetime.
- id - идентификатор абонента
- vas_id - подключаемая услуга

Анализ данных:

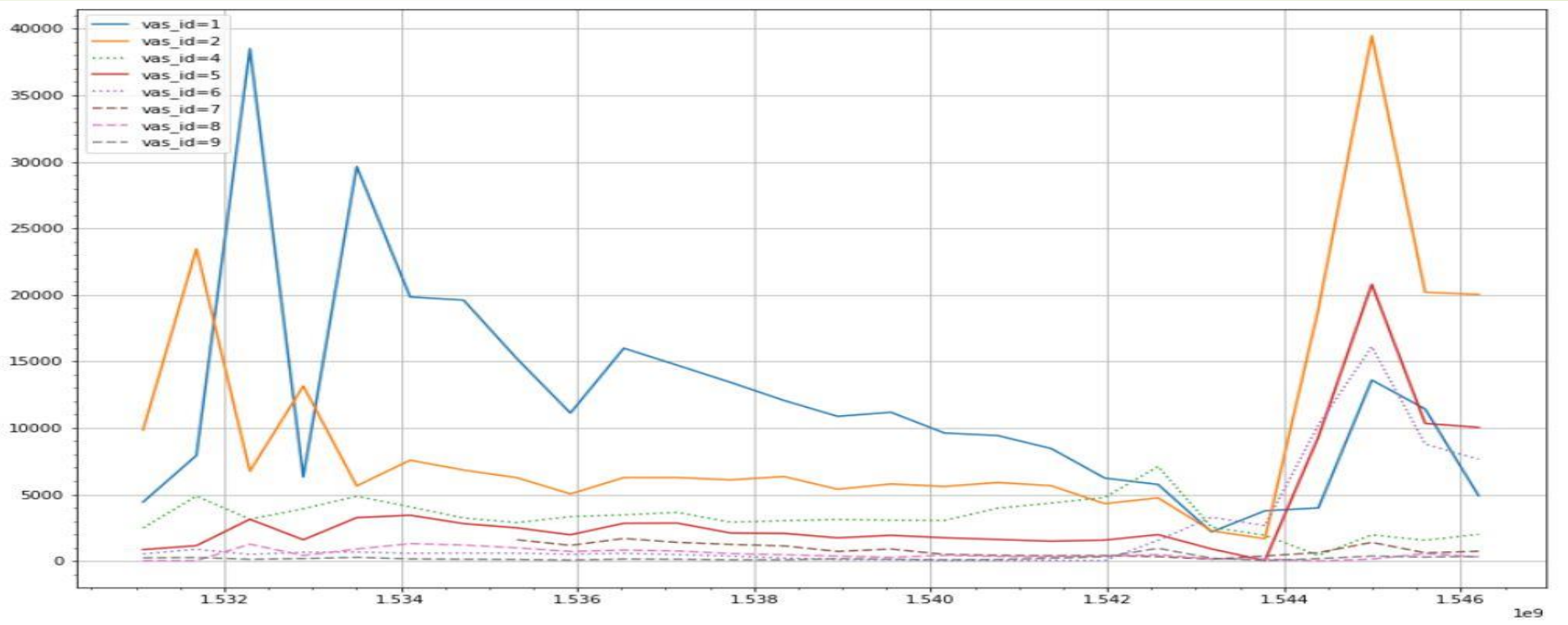
- Данные представлены во временном промежутке с 2018-07-09 по 2018-12-31.

Предлагаемые услуги (шт.):	Подключённые услуги (шт.):	Не подключённые услуги (шт.):
1.0 310175	1.0 5664	1.0 304511
2.0 249505	2.0 4797	2.0 244708
4.0 85756	4.0 21765	4.0 63991
5.0 94085	5.0 1692	5.0 92393
6.0 57878	6.0 24704	6.0 33174
7.0 15432	7.0 213	7.0 15219
8.0 13350	8.0 347	8.0 13003
9.0 5472	9.0 1004	9.0 4468

- Как видно из таблицы, самые предлагаемые услуги - 1 и 2, но они лидируют и по отказам подключений. А самые подключаемые - 4 и 6.

Анализ данных (продолжение):

- Графики зависимости количества предложений каждой услуги от времени:



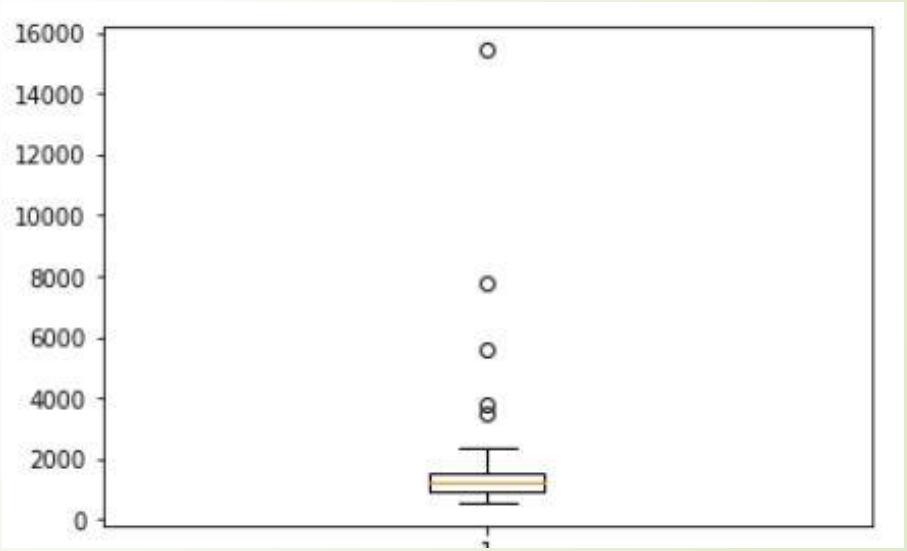
- Из графиков видно, что активнее всего услуги предлагают в начале временного ряда и в конце, что соответствует середине (2018-07-09) и концу (2018-12-31) 2018 года.

Анализ данных (продолжение):

- Из вычислений и BoxPlot видно аномально высокое общее количество подключаемых услуг в дату 2018-11-19 (buy_time - 1542574800). Даже если ошибок в данных нет, для обучения модели такие показатели можно считать выбросом.

```
df_train.loc[(df_train['target']==1, 'buy_time') ].value_counts()
```

```
1542574800    15418
1544994000     7802
1544389200     5616
1545598800     3786
1546203600     3495
1543179600     2348
1531688400     1553
1543784400     1503
1540760400     1484
1541365200     1435
1533502800     1360
1532293200     1320
1541970000     1283
1532898000     1230
1540155600     1092
1534107600     1053
1534712400      955
1539550800      932
1538946000      929
1535922000      900
1538341200      871
1537131600      864
1536526800      853
1537736400      797
1535317200      750
1531083600      557
Name: buy_time, dtype: int64
```



Выбор модели и результирующие тестовые метрики:

- Для создания модели использовался CatBoostClassifier, как одно из самых популярных решений для задач классификации.
- В пользу этого выбора также послужило умение CatBoost работать с дисбалансом данных в целевой переменной и автоматической обработкой типов признаков и отсутствующих значений.

TRAIN

	precision	recall	f1-score	support
0.0	0.96	0.99	0.97	516922
1.0	0.80	0.40	0.54	40285
accuracy			0.95	557207
macro avg	0.88	0.70	0.76	557207
weighted avg	0.94	0.95	0.94	557207

TEST

	precision	recall	f1-score	support
0.0	0.95	0.99	0.97	254545
1.0	0.64	0.30	0.41	19901
accuracy			0.94	274446
macro avg	0.79	0.64	0.69	274446
weighted avg	0.92	0.94	0.93	274446

CONFUSION MATRIX

col_0	0.0	1.0
target		
0.0	251140	3405
1.0	13933	5968