Объединение данных

Для объединения использовалось pd.merge_asof.

Для тренировочной выборки с загрузкой через dask.dataframe таблицы features.csv.

Для итогового файла predict_ansver — идет отбор по id в бд MYSQL и уже объединяется.

Выбор данных для обучения

Для обучения была отобраны данные по id абонента и сгруппированы. Получилось 831 тыс строк.

Из них лишь 0,07% были данные с 1 в целевой переменной или 60 тыс строк.

Для обучения итоговой модели я отобрал случайным образом 60 тыс строк с 0 в целевой переменной и собрал датасет из 120 тыс строк с равным кол-во 0 и 1.

Предобработка

Модель была испытана с заменой выбросов по 1.5 интерквартильным размахам, по 90% персентилю, по 95% персентилю, по 99% персентилю. Заменялись на моду, медиану, среднее.

В конечной модели замена выбросов не включена совсем.

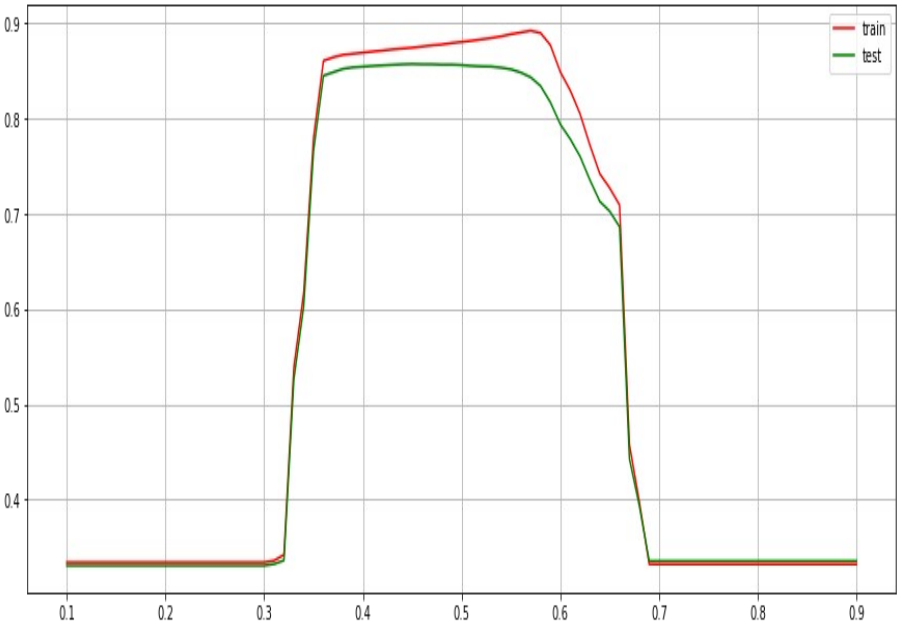
Так же было испытано применение понижение размерности PCA до 99% онформации данных, что сокращало кол-во колонок датафрейма на 60-90 колонок. Метрики данных испытаний так же были недалеко друг от друга, но при последнем испытании метрика без PCA была на 0.95% выше — поэтому оставил ее для конечного результата.

На тренировочной выборке практически при любых изменениях график f1 выглядел одинаково(максимум 0,867)

После применения PCA

Изменения в графике после применения понижения размерности до 99% или на 60-90 колонок

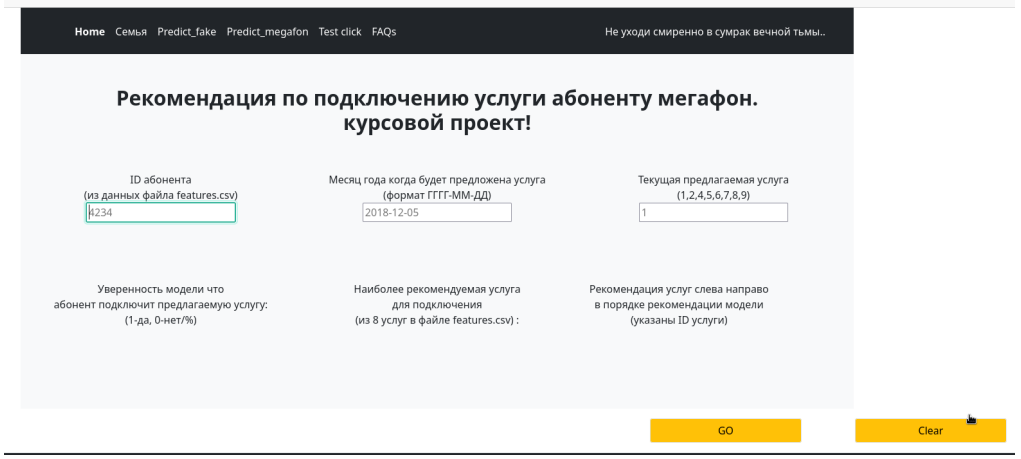
В конечной модели исключил PCA, т. к. метрика немного была лучше без нее на испытаниях всех данных из data_train.csv



Бд MYSQL

Для исключения проблем с оперативной памятью была создана БД MYSQL на облачном сервере линукс и туда заброшены все данные из файла features.csv. Там же реализована страница на сайте, где вводится id пользователя(который присутствует в файле features.csv), дата, услуга. В том числе там выдается ответ на запрошенную услугу, наиболее рекомендуемая и список в порядке убывания всех услуг. Просмотреть можно на сайте http://bainel.ru/predict_megafon.html.

- На сайте

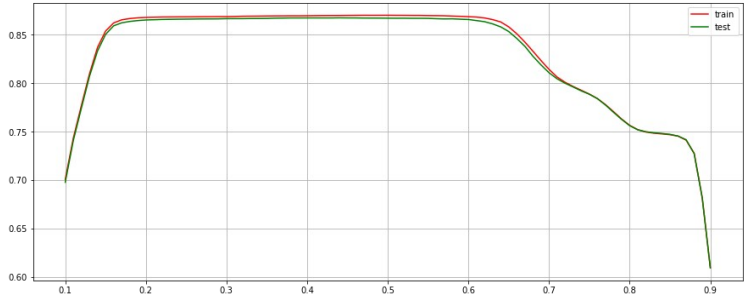


Выбор модели

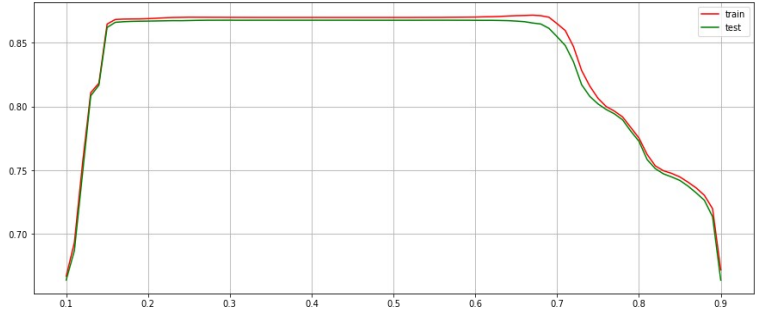
Были сделаны пробы на логистической регрессии и xgboost моделях.

Остановил свой выбор на xgboost, т. к. при одинаковых показателях все же алгоритм более совершенный и данные совершенно не ясные в плане что скрывается за данными цифрами. Поэтому не известны зависимости, не известно где и какие параметры можно преобразовывать в дамми переменные, а где нужно по возрастающей оставить категории. Считаю это большим минусом

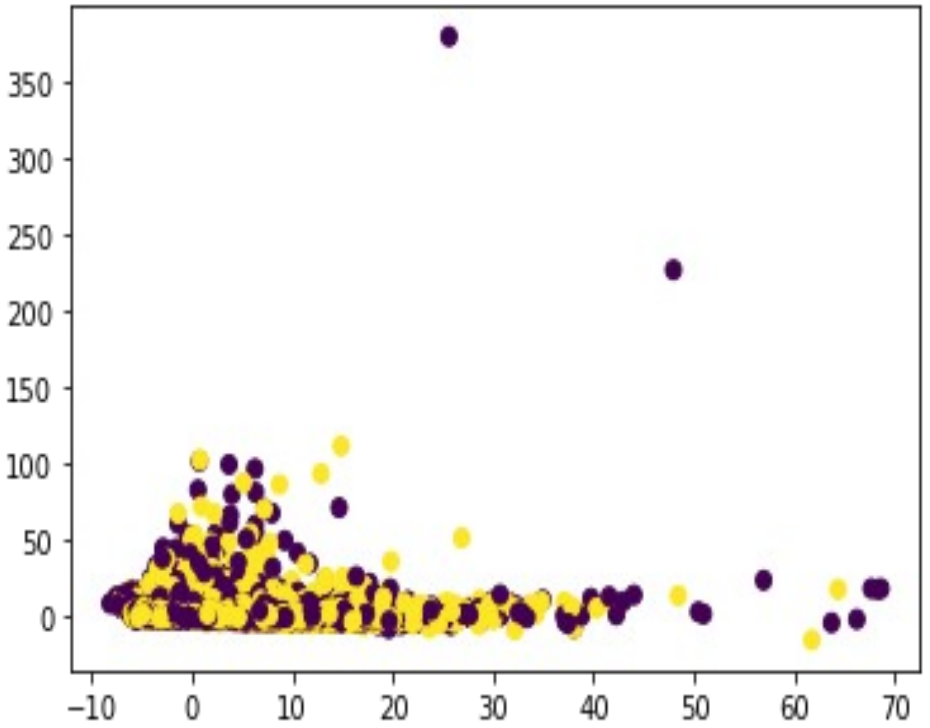
- На догистической регрессии



- На xgboost



Удаление данных



- В плане выбросов - они вроде бы есть
- В плане информации — не понятно с чем мы вообще работаем

Рекомендация услуг для предложения абоненту

Ввод [13]:
```
# что бы не прописывать постоянно - напишем функцию в отдельном файле

from predict_megafon import predict_megafon

out = predict_megafon(data)
out
```

```
модель загружена
```

Out[13]:

	id	1	2	3	4	5	6	7	8
0	2582523	4.0	6.0	2.0	1.0	9.0	5.0	8.0	7.0
1	1292549	6.0	4.0	2.0	1.0	9.0	5.0	8.0	7.0
2	4053116	6.0	4.0	2.0	1.0	9.0	5.0	8.0	7.0
3	4158361	4.0	6.0	2.0	1.0	9.0	5.0	8.0	7.0
4	3754468	6.0	4.0	2.0	1.0	9.0	5.0	8.0	7.0
...	...	...	...	...	...	...	...	...	...
9995	3761476	6.0	4.0	2.0	1.0	9.0	5.0	8.0	7.0
9996	32525	6.0	4.0	2.0	1.0	9.0	5.0	8.0	7.0
9997	68576	6.0	4.0	1.0	9.0	5.0	8.0	7.0	2.0
9998	2029031	6.0	4.0	2.0	1.0	9.0	5.0	8.0	7.0
9999	4274668	6.0	2.0	4.0	1.0	9.0	5.0	8.0	7.0

10000 rows × 9 columns

- Рекомендательная система построена на подставлении номера услуги в „vas_id“ и дальнейшим предиктом