Прогнозирование оттока клиентов

1. Цели и задачи проекта.

Выстраивание взаимоотношений с клиентами или customer relationship managmenet (CRM) является важным звеном в функционировании любого бизнеса. От того, насколько хорошо простроена работа с аудиторией зачастую зависит успех всей компании.

В проекте рассмотрена одна из наиболее актуальных задач из области CRM: прогнозирование оттока пользователей или churn prediction. Суть задачи заключается в заблаговременном нахождении сегмента пользователей, склонных через некоторый промежуток времени отказаться от использования некоторого продукта или услуги. Точное и своевременное нахождение таких пользователей позволяет эффективно управлять оттоком, например, выявлять причины оттока, помогать пользователям, попавшим в группу риска, решать их проблемы и задачи; проводить кампании по удержанию.

Эта задача актуальна для большинства организаций, оказывающих услуги в сегменте B2C и вдвойне актуальна в областях, где распространение услуги близко к отметке 100%. Хороший пример такой области – рынок мобильной связи, где насыщение уже фактически произошло, и как следствие постепенно снижается прирост клиентской базы. В такой ситуации задача удержания клиентов и выстраивания с ними взаимоотношений выходит на первый план.

В 2009 году проводилось соревнование KDD Cup: Customer relationship prediction. В рамках этого соревнования участникам предлагалось решить несколько задач из области Customer Relationship Management (CRM), одна из которых была оценка вероятности того, что клиент осуществит переход к конкуренту (churn prediction).

Данные для соревнования были предоставлены французской телекоммуникационной компанией Orange. В задаче речь идет о клиентских данных, поэтому данные были предварительно обфусцированы и анонимизированны. Мы будем работать с набором данных, состоящих из 40 тыс. объектов и включающий 230 переменных, из которых первые 190 переменных - числовые, и оставшиеся 40 переменные - категориальные.

Для того, чтобы находить пользователей, склонных к оттоку, строят прогнозные модели - модели, позволяющие прогнозировать вероятность того, что пользователь покинет сервис. В классической постановке строятся вероятностные модели бинарной классификации, где целевой класс представляют собой пользователи, покидающие сервис. Вероятность того, что пользователь принадлежит целевому классу и есть целевая величина - вероятность оттока. Соответственно, чем эта вероятность больше, тем больше шансов, что пользователь откажется от использования нашего сервиса.

Описательный анализ данных

Наблюдается дисбаланс по классам:

- отток: 7.44%
- не отток: 92.56%

Проведем описательный анализ и визуализацию так называемых "закрытых" данных - данных, которые предоставляются для анализа и построения моделей без описания. В рассматриваемых данных не известно, что содержательно представляют собой те или иные признаки, а известны лишь их типы: числовые, категориальные. Такие ситуации - не редкость при работе с «чувствительными» данными, например, в сфере банковской аналитики, HR-аналитики, сфере телекоммуникаций, страхования, здравоохранения, недвижимости или ритейла.

Рассчитаем корреляции переменных с целевой функцией и проанализируем полученные данные.

Числовые переменные

Целевая функция представляет собой бинарную переменную. Корреляции Пирсона и Спирмена, примененные к бинарным или категориальным признакам будут иметь мало смысла, поэтому мерой силы взаимосвязи вещественной переменной с целевой бинарной переменной может служить разница мат. ожиданий: К примеру, $X1 \in R$ и $X2 \in \{0, 1\}$ будут положительно коррелированы, если $E(X1 \mid X2 = 1) > E(X1 \mid X2 = 0)$.

Хоть эта величина не нормированная и может меняться в любом диапазоне, от $-\infty$ до $+\infty$, её гораздо легче интерпретировать, чем обычный коэффициент корреляции. Отсортируем сначала по убыванию корреляции, а затем по возрастанию разреженности, и выделим переменные с долей отсутствующих значений меньше 30%:

Vars	Nans	Corrs	Y=0	Y=1
Var153	0.1004	392543.5	33195	2789
Var38	0.1004	177037.7	33195	2789
Var113	0	162842.2	37024	2976
Var76	0.1004	113461.7	33195	2789
Var133	0.1004	104324.5	33195	2789
Var134	0.1004	31733.51	33195	2789
Var163	0.1004	29505.38	33195	2789
Var149	0.1454	28963.65	31609	2575
Var81	0.110875	17521.22	32806	2759
Var125	0.111	3876.42	32815	2745
Var13	0.111	399.17	32815	2745
Var140	0.111	360.58	32815	2745
Var6	0.110875	135.71	32806	2759
Var119	0.110875	45.42	32806	2759
Var74	0.111	42.03	32815	2745
Var28	0.100425	14.3	33194	2789
Var73	0	14.02	37024	2976
Var22	0.1004	7.03	33195	2789
Var21	0.110875	5.58	32806	2759
Var25	0.1004	3.4	33195	2789

Всего оказалось 42 полезных числовых переменных.

Наименее полезными, шумовыми оказались следующие числовые переменные:

- изначально полностью пустые (16 из 190): Var8 Var15 Var20 Var31 Var32 Var39 Var42 Var48 Var52, Var55 Var79 Var141 Var167 Var169 Var175 Var185
- а также очень разреженные числовые переменные, разреженность которых составляют от 92.52% до 99.65% (132 из 190), такое кол-во мы не будем выкладывать.

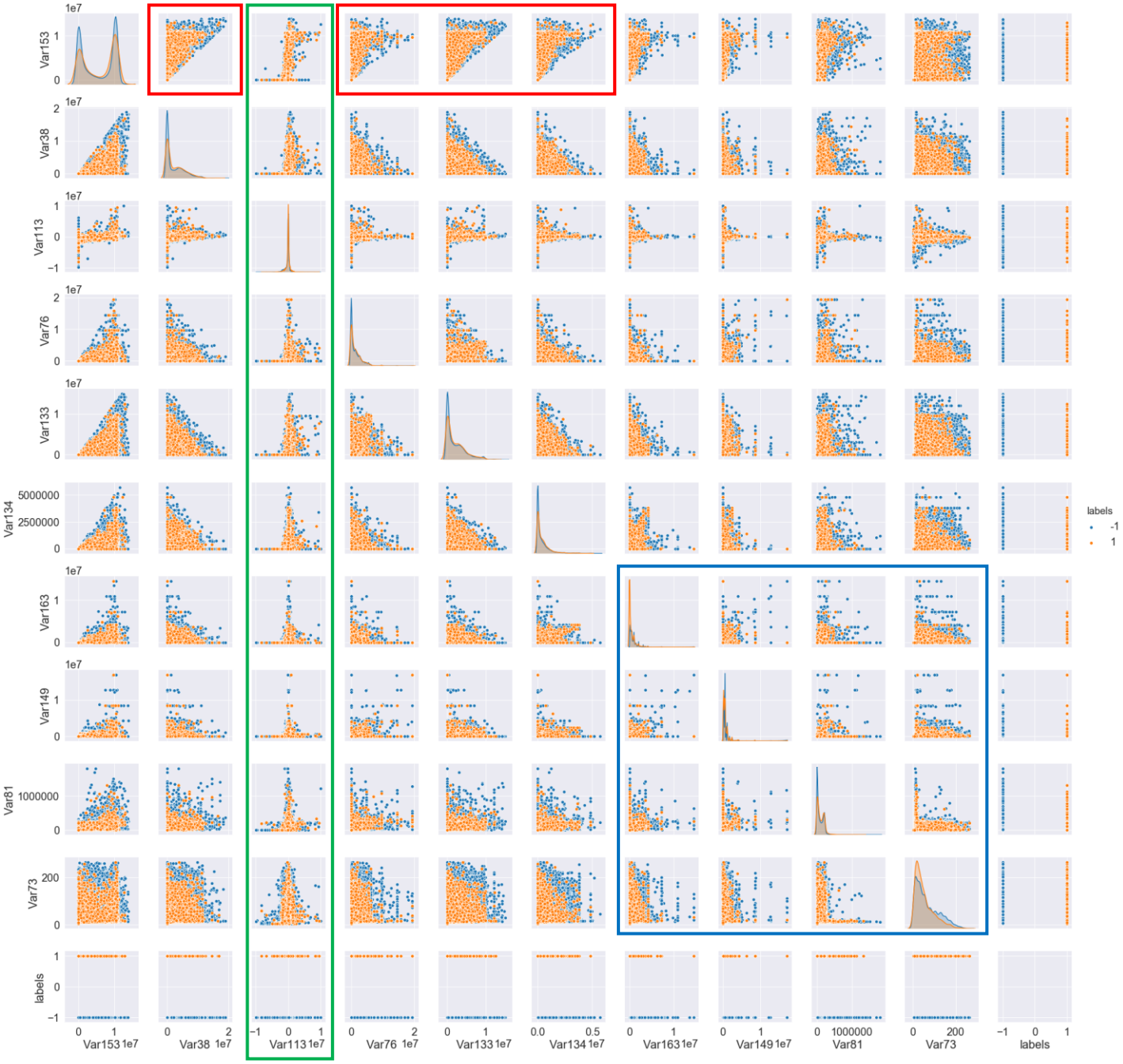
Рассмотрим топ 20 числовых переменных, наиболее сильно коррелирующих с целевой функцией. Для этих переменных построим:

- Распределения в разрезе классов
- Отобразим объекты в координатах пар признаков

В целом практически по большинству рассмотренных числовых переменных наблюдаются облака точек с нелинейными зависимостями. Из топа 20-ти числовых переменных:

Среди зависимостей первой десятки числовых переменных:

- В попарных координатах переменных Var153, Var38, Var133, Var76, Var134 между собой, видны выраженные взаимосвязи, в которых значения собираются выше или ниже диагонали на графике.
- В попарных координатах переменных Var163, Var149, Var81, Var73 с остальными переменными, видно, что некоторые значения собрались около определенных числовых категорий.
- Интересно посмотреть на переменную Var113, практически на всех попарных графиках, видно, что очень много значений собираются около нуля.

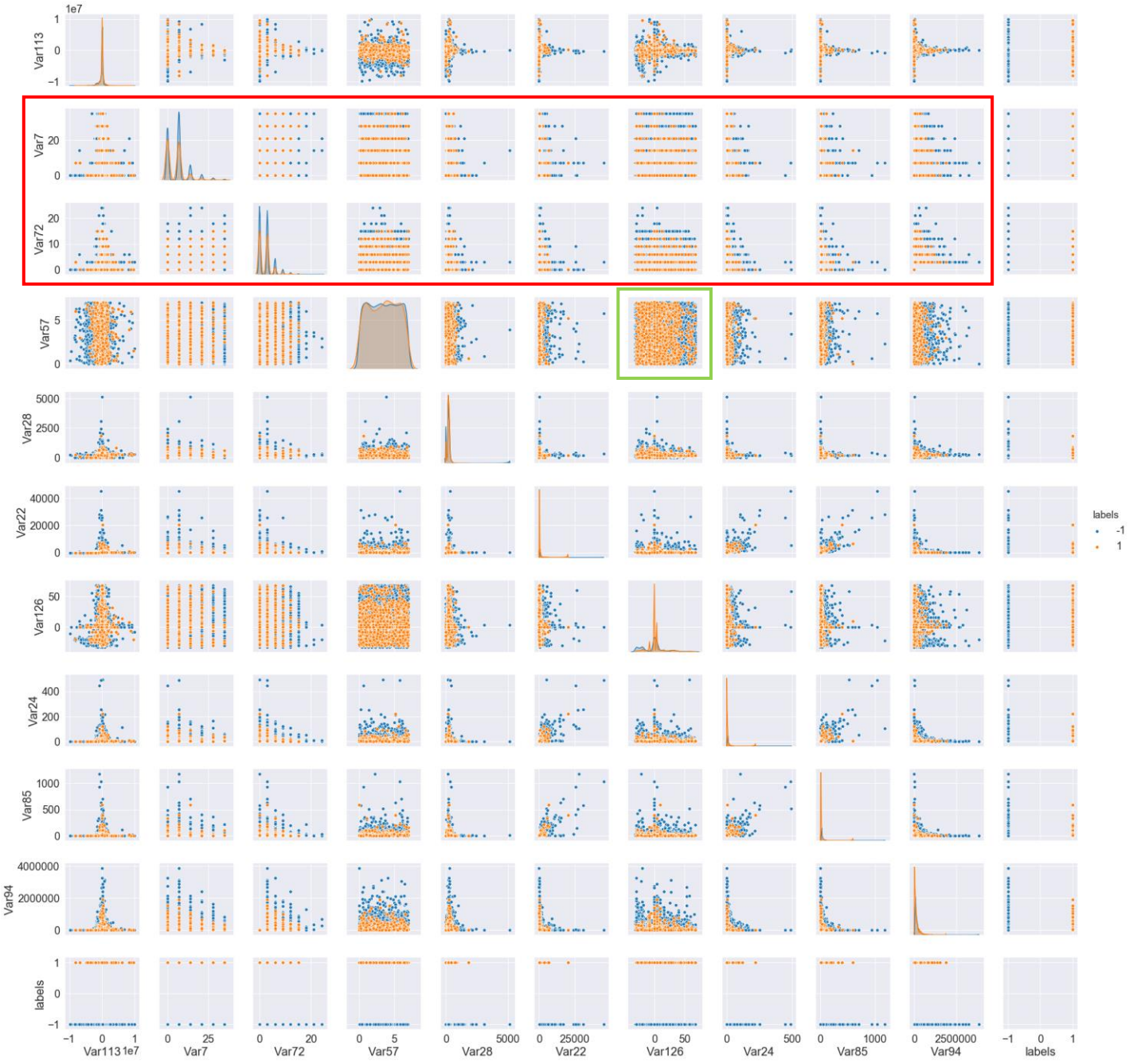


Среди зависимостей второй десятки числовых переменных:

- видны слабо выраженные зависимости, в которых значения собираются выше или ниже линии под углом: Var6, Var119 - Var22, Var21, Var25
- видна практически полная линейная зависимость: Var21 - Var22
- и большая корреляция: Var21, Var22 - Var25

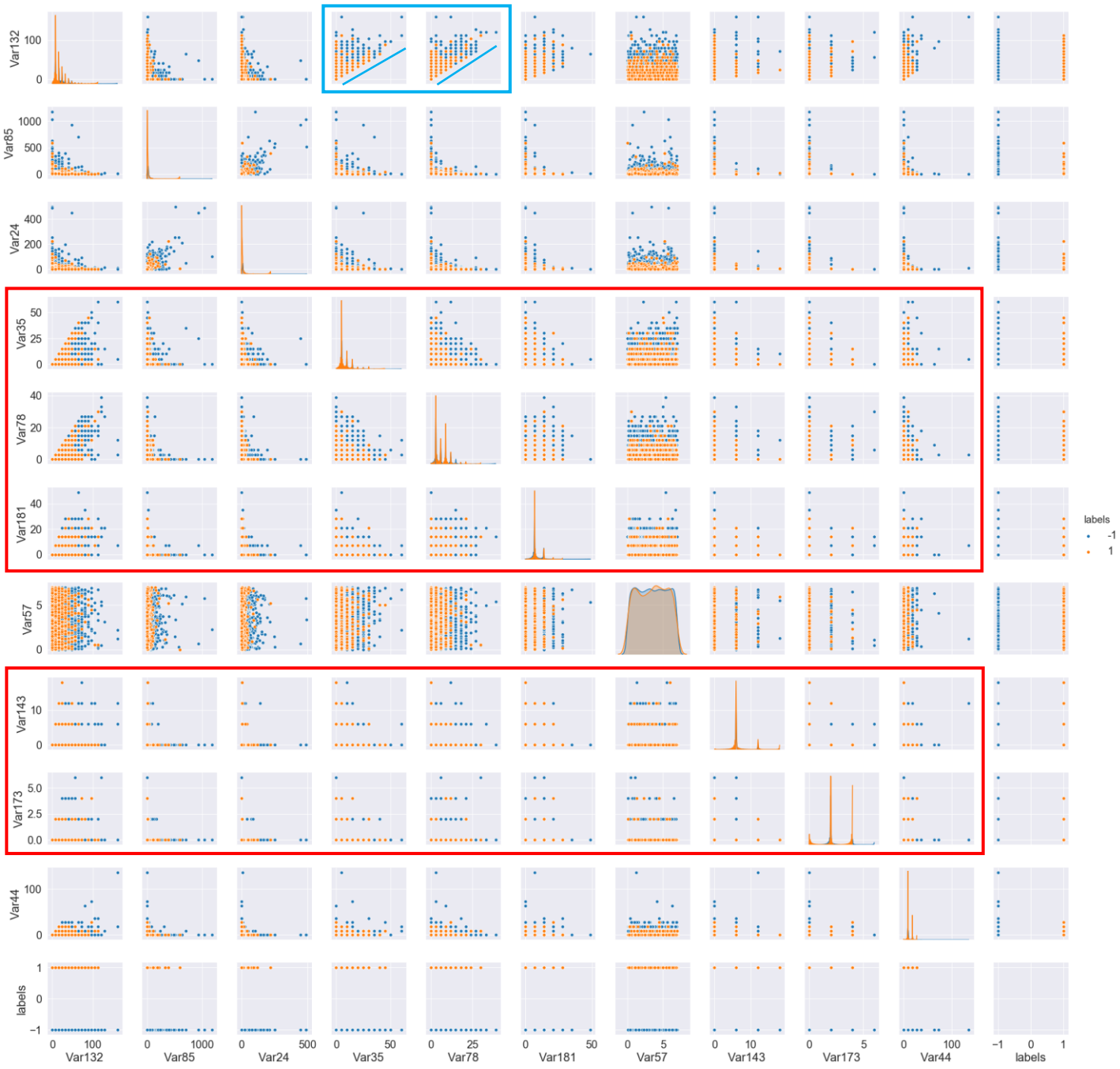
Среди 10 случайно выбранных числовых переменных:

- по взаимосвязи со всеми 10 переменными видно, что Var7, Var72 имеют определенные кратные числовые категории.
- интересно посмотреть на взаимосвязь: Var57-Var126 - объекты занимают определенную прямоугольную область.



Среди 10 наименее коррелирующих числовых переменных с целевой:

- по взаимосвязи со всеми 10 переменными видно, что Var35, Var78, Var143, Var173, Var181 имеют определенные кратные числовые категории.
- интересно посмотреть на взаимосвязь: Var35, Var78 - Var132 видны зависимости, в которых значения собираются выше или ниже линии под углом.



По остальным рассмотренным числовым признакам сильных взаимосвязей на графиках не наблюдается.

Категориальные переменные

Для подсчета меры взаимосвязи категориальных переменных с целевой бинарной используем коэффициент V Крамера и предположим, какие переменные окажут наибольшее влияние (вклад) в модель:

Vars	Nans	Corrs	Y=0	Y=1	Categories
Var218	0.014	0.101756	36619	2821	3
Var206	0.110875	0.085054	32806	2759	22
Var228	0	0.075575	37024	2976	30
Var205	0.038675	0.072558	35606	2847	4
Var229	0.569425	0.062485	16266	957	5
Var207	0	0.058276	37024	2976	14
Var225	0.523375	0.056515	17923	1142	4
Var227	0	0.055619	37024	2976	7
Var221	0	0.049119	37024	2976	7
Var210	0	0.04854	37024	2976	6
Var226	0	0.048516	37024	2976	23
Var211	0	0.031932	37024	2976	2
Var195	0	0.031671	37024	2976	23
Var219	0.1049	0.025773	33158	2646	23
Var194	0.74525	0.019348	9512	678	4
Var201	0.74525	0.018134	9512	678	3
Var203	0.003075	0.014221	36906	2971	6
Var196	0	0.012235	37024	2976	4
Var208	0.003075	0.011882	36906	2971	3
Var223	0.1049	0.011714	33158	2646	5

Всего оказалось 18 категориальных переменных, с долей отсутствующих значений <90% и, возможно, содержащие не больше 30-ти категорий.

Наименее полезными, шумовыми оказались следующие категориальные переменные:

- изначально полностью пустые (2 из 40): Var209 Var230.
- а также очень разреженные категориальные переменные, пустые ячейки которых составляют от 74.53% до 98.59% (4 из 40): Var194, Var201, Var213, Var191, Var224, Var215.

Из рассмотренных категориальных переменных, наблюдаются совпадения по количеству категорий:

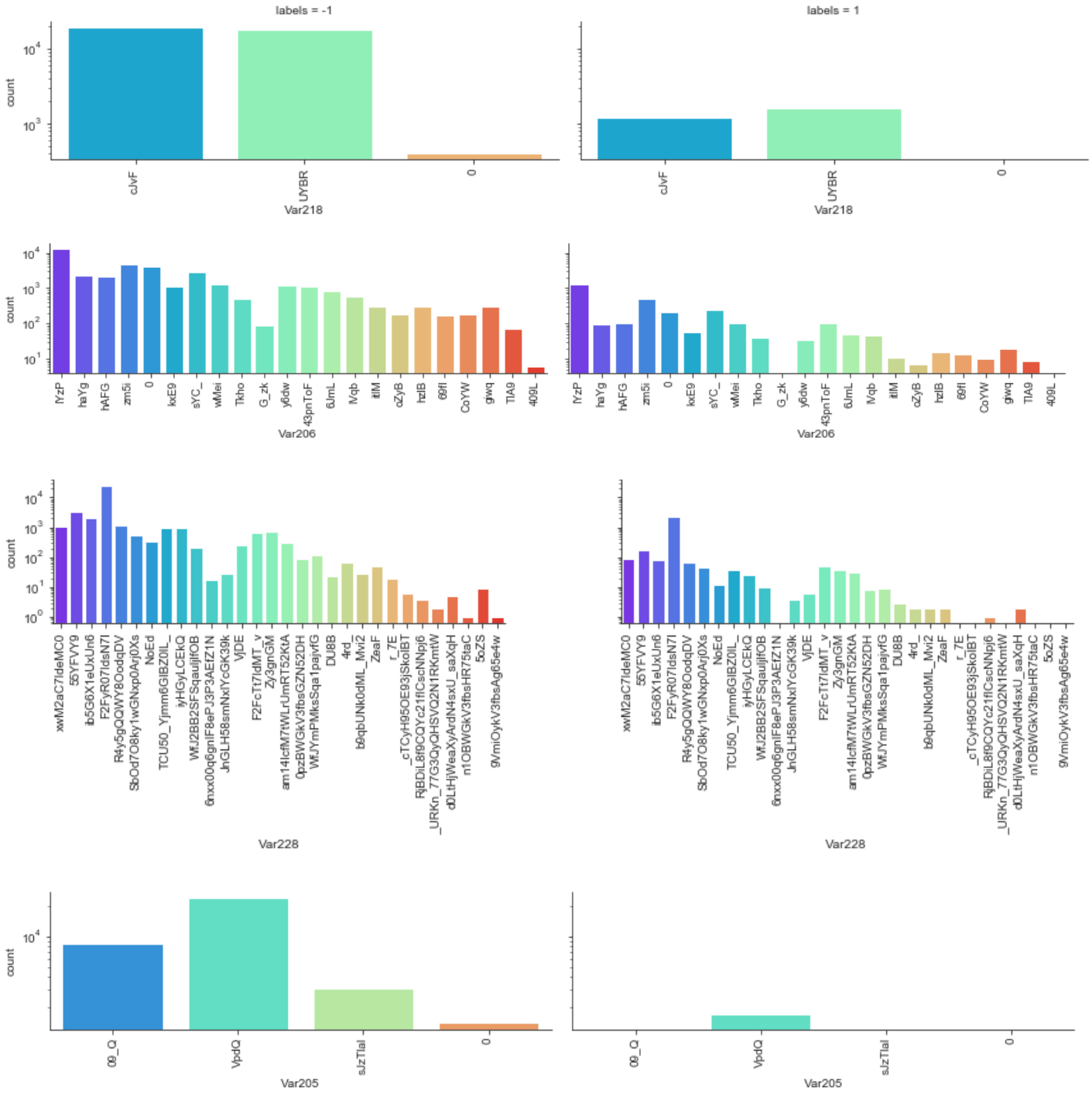
Vars	Nans	Corrs	Y=0	Y=1	Categories
Var226	0	0.048516	37024	2976	23
Var195	0	0.031671	37024	2976	23
Var219	0.1049	0.025773	33158	2646	23
Var214	0.50915	0.525991	18452	1182	13298
Var200	0.50915	0.525991	18452	1182	13298
Var220	0	0.292593	37024	2976	3891
Var198	0	0.292593	37024	2976	3891
Var222	0	0.292593	37024	2976	3891

- Var195, Var226, Var219 - в каждой переменной 23 категории.
- Var198, Var220, Var222 - в каждой переменной 3891 категории и одинаковая корреляция.
- Var200, Var214 - в каждой переменной 13298 категории, одинаковая корреляция и разреженность.

Что может говорить о том, что названия категорий могут относиться к одному семейству, или по одним и тем же названиям категории показаны в разных разрезах. Можно удалить, к примеру: Var220, Var222, Var214.

Построим и сравним гистограммы значений категориальных переменных для разных классов.

Можно было бы сразу отобразить гистограммы категориальных переменных для разных классов, однако некоторые переменные содержат слишком много категорий для отображения (есть переменные, содержащие в себе около 12-13 тыс. категорий, и они имеют относительно высокие коэффициенты Крамера). Отобразим в порядке убывания коэффициента Крамера только первые пять переменных, содержащие не больше 30-ти категорий:



В целом из-за того, что выборка не сбалансирована по классам, есть существенная разница по частотам категорий при разных классах в рассмотренных категориальных переменных.

Окажется ли дальнейший отбор признаков полезным для построения модели?

Так как шумовых признаков достаточно (из 190 числовых признаков и 40 категориальных полезными могут оказаться только 42 и 18 соответственно) в целом отбор признаков необходим для улучшения качества и скорости модели.

По числовым признакам отбор с помощью модели может оказаться полезным:

- Хоть разница мат. ожиданий и показала определенные корреляции, модель может сделать отбор признаков по корреляции с целевой переменной лучше.
- L1 регуляризация может убрать лишние переменные, имеющие высокие корреляциями с другими, для остальных нелинейных зависимостей помогут случайные леса и градиентный бустинг.

По категориальным признакам отбор признаков может оказаться полезным:

- необходимо понять (если делать undersampling), на сколько может быть выгодно кодирование категориальных признаков с большим кол-вом категорий, поскольку кол-во объектов оттока составляет всего 2976. Они могут пригодиться только, если делать отбор с помощью модели или понижать размерность:

Vars	Nans	Corrs	Y=0	Y=1	Categories
Var216	0	0.22894	37024	2976	1819
Var198	0	0.292593	37024	2976	3891
Var220	0	0.292593	37024	2976	3891
Var222	0	0.292593	37024	2976	3891
Var199	0.0001	0.407831	37021	2975	4401
Var202	0.000025	0.363433	37023	2976	5543
Var217	0.014	0.529863	36619	2821	12471
Var200	0.50915	0.525991	18452	1182	13298
Var214	0.50915	0.525991	18452	1182	13298

2. Методика измерения качества и критерий успеха.

Так как выборка несбалансированная: отток=1 (7.44%), неотток=-1 (92.56%), accuracy будет нерепрезентативной метрикой однозначно. До того, как выбрать порог бинаризации, нужно оценить качество оценок принадлежности к классу 1 через площадь под кривой AUC–PRC. PR-кривая строится в осях precision и recall и, в отличие от ROC-кривой, изменяется при изменении баланса классов, что проще интерпретировать в случае нашей несбалансированной выборки, поэтому AUC–PRC и будет нашей ключевой метрикой качества.

Задача чем-то похожа на задачи медицинской диагностики, где в приоритете recall, а затем precision. То есть, в задаче определения, склонен клиент уйти или нет (0 - неотток, 1 - отток), нужно будет стараться избегать ошибок false negative, требуя высокий recall, гораздо хуже пропустить наличие склонности к оттоку, нежели определить в начале, что клиент склонен к оттоку, и при дальнейшей диагностике выявить ошибку. Однако необходимо будет посмотреть на графики recall и precision, и в зависимости от экономических соображений, подбирать порог бинаризации.

3. Техническое описание решения.

1) Подберем оптимальную стратегию проведения кросс-валидации:

Выберем на сколько фолдов мы будем делить выборку:

- много фолдов - низкое смещение и большой разброс;
- мало фолдов - большое смещение и низкий разброс;
- чем больше выборка, тем меньше нужно фолдов. В данном случае нужно выбрать меньше фолдов, поскольку данных итак много. Однако многое зависит от алгоритма, если обучение не слишком трудоемкое, то можно выбрать и побольше фолдов для снижения смещения. Выберем 5 фолдов (4 на тест и 1 на контроль).

Выберем тип разбиения данных:

- необходимо **стратифицировать** выборку: а) выборка не сбалансирована 7.44% оттока и 92.56% не оттока, поэтому использование стратификации может сделать модели, очень чувствительные к дисбалансу, более стабильными; б) в объектах нет регрессии, так чтобы при их перестановке модель переобучилась.
- Применить **shuffle внутри** самой стратегии, поскольку объекты изначально могут быть расположены в определенном порядке
- не применять **ShuffleSplit**: тестовые выборки на итерациях kfold могут покрыть всю выборку, а на shufflesplit они могут пересекаться, и, таким образом, shufflesplit может не учесть важные участки выборки и особые объекты, которых достаточно, поскольку кол-во числовых и категориальных переменных полностью не пустых составляет 174 и 38 соответственно.
- **leave one out** лучше подходит для маленьких выборок, а данном случае мы имеем дело с относительно большой выборкой (40 тыс. объектов). Хоть тестовая выборка в данном случае состоит их одного объекта на каждой итерации, данных у нас предостаточно, и данная стратегия будет слишком накладной по времени.

Учитывая вышесказанное, выбор оптимальной стратегии проведения кросс-валидации на тестовой выборке падает на **StratifiedKFold с 4-мя фолдами и с shuffle внутри**.

2) Удалим следующие переменные из датасета:

- полностью пустые числовые (16 из 190): 'Var8' 'Var15' 'Var20' 'Var31' 'Var32' 'Var39' 'Var42' 'Var48' 'Var52', 'Var55' 'Var79' 'Var141' 'Var167' 'Var169' 'Var175' 'Var185'.
- полностью пустые категориальные: 'Var209', 'Var230'.
- дублирующиеся категориальные: 'Var220', 'Var222', 'Var214'.

3) Построим для начала несколько простых моделей и найдем baseline-решение:

1) Пре-процессинг:
   1. Заполнение пропущенных значений числовых переменных: mean.
   2. Масштабирование числовых переменных: StandardScaler.
   3. Заполнение пропущенных значений категориальных переменных: constant (добавление еще одной категории 'NA').
   4. Обработка категориальных признаков OneHotEncoder.
   5. Понижение размерности преобразованных категориальных признаков: TruncatedSVD. Поскольку общее кол-во категорий доходит до 42433, TruncatedSVD может очень хорошо подойти для снижения размерности разреженных матриц, к примеру, снизим кол-во компонент до первоначальных 35-и.

2) Подбор классификатора со сбалансированными весами классов: Logistic_Regression, Random_Forest, xgboost. Лучшим по качеству оказался xgboost:

	prc_auc	roc_auc
Logistic_Regression	0.133334	0.658076
Random_Forest	0.098907	0.582692
xgboost	0.186459	0.719646

3) Для получения финального Pipeline применим несколько типов стратегий и найдем лучшее решение:

1) Пре-процессинг:
   1. Заполнение пропущенных значений числовых переменных: constant, mean, median.
   2. Масштабирование числовых переменных: StandardScaler
   3. Обработка категориальных признаков: OneHotEncoder, OrdinalEncoder, HashingEncoder, BinaryEncoder.
2) Undersampling: RandomUnderSampler, TomekLinks, NeighbourhoodCleaningRule, EditedNearestNeighbours.
3) Важность признаков: SelectFromModel (ExtraTreesClassifier, LinearSVC, LogisticRegression), RFE (LinearSVC).
4) Подбор оптимальных параметров модели (параметров препроцессинга, undersampling и классификатора).

Подробный подбор параметров Pipeline:

1) Обработка пропущенных значений

Сравним эти стратегии между собой с помощью оценки качества моделей кросс-валидации, построенных на датасетах с использованием различных стратегий. В данном случае будем рассматривать только числовые переменные, поскольку в категориальных переменных, обработка пропущенных значений будет выглядеть как просто добавление еще одной категории:
Рассмотрим обработку пропущенных значений путем замены нулем, средней и медианой:

num__imputer__strategy	prc_auc	roc_auc
constant	0.191749	0.723661
mean	0.19221	0.726136
median	0.187328	0.717882

Обработка пропущенных значений не сильно сказывается на результатах модели. Наиболее оптимальным с точки зрения качества оказалась обработка пропущенных значений путем замены на среднее.

2) Обработка категориальных признаков

Количество категорий в данном датасете доходит до 13 тыс., что означает, что нам не обойтись без кодировщиков категориальных переменных:

- **pandas.get_dummies**
- **sklearn.feature_extraction.DictVectorizer**
- **sklearn.preprocessing.OneHotEncoder** - в данном случае отлично подходит для подбора параметров при проведении кросс-валидации в трансформере pipeline. FeatureUnion, однако из-за присвоения каждой категории отдельной колонки, а таких колонок может быть 13 тыс., серьезно загрузит память. get_dummies, DictVectorizer и OneHotEncoder выполняют практически одну и ту же функцию.

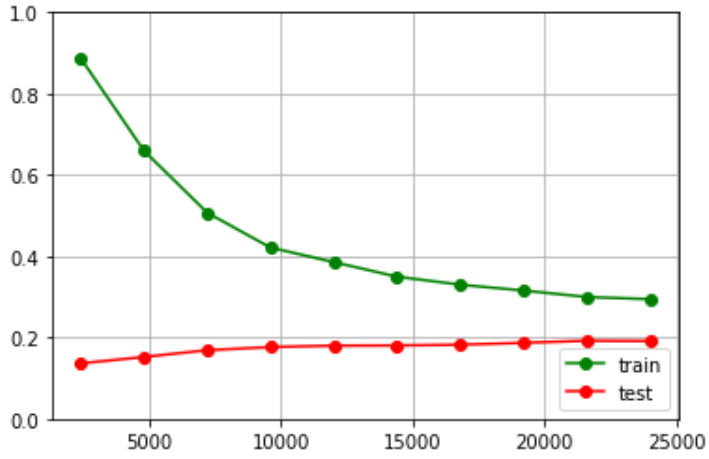
- **category_encoders.binary.BinaryEncoder** - гибрид one-hot и hashing кодировщиков, в сравнении с OneHotEncoder, создает меньше колонок, сохраняя информацию и является более эффективным по использованию памяти, отлично справляется с большим кол-вом категорий.
- **category_encoders.hashing.HashingEncoder** - похож на one-hot encoding, однако как и BinaryEncoder создает меньшее кол-во колонок, но иногда с некоторой потерей данных. Также хорошо справляется с большим кол-вом категорий.
- **category_encoders.ordinal.OrdinalEncoder** - преобразует каждое строковое значение в целое число. Первое уникальное значение в вашем столбце становится 1, второе становится 2, третье становится 3 и так далее. OrdinalEncoder использует один столбец целых чисел для представления классов, из-за чего можно не применять понижение размерности.

category_encoders	prc_auc	roc_auc
OneHotEncoder	0.19221	0.726136
OrdinalEncoder	0.208867	0.734563
HashingEncoder	0.195196	0.726873
BinaryEncoder	0.208453	0.736686

Четыре выше рассмотренных метода обработки категориальных признаков, включая изначальный OneHotEncoding не сильно разнятся по результатам. Самым оптимальным по качеству модели оказался OrdinalEncoder, как и самым удобным для дальнейшего отбора признаков, поскольку он оставляет исходное кол-во колонок и не применяет хеширование.

3) Undersampling

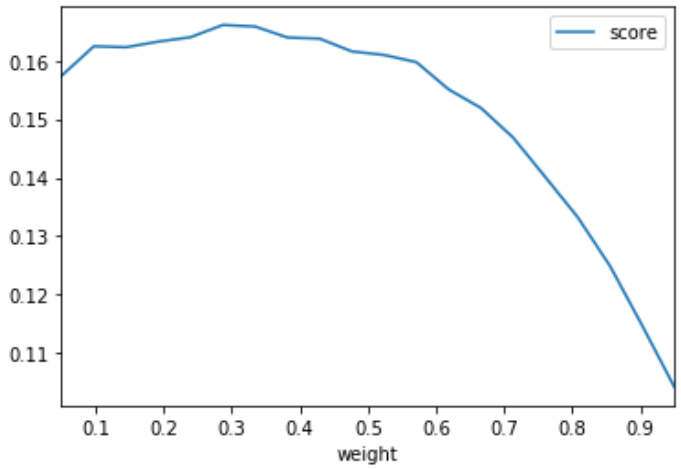
Давайте оценим, как много объектов действительно нужно для построения качественной модели. Для обучения доступна достаточно большая выборка и может так оказаться, что, начиная с некоторого момента, рост размера обучающей выборки перестает влиять на качество модели. Посмотрим, так ли это в этом случае: построим кривые обучения, обучая модель на выборках разного размера начиная с небольшого количество объектов в обучающей выборке и постепенно наращивая её размер с некоторым шагом.



Кривая обучения продолжает уменьшаться при росте количества объектов, однако тестовая кривая значительно растет до 10000 и затем слабо продолжает расти до 21599 объекта. Можно сделать вывод, что для построения качественной модели достаточно 10 тыс. объектов обучающей выборки.

Часто несбалансированные по классам выборки приводят к различным проблемам при обучении моделей. Давайте попробуем по-разному обработать выборку, поиграть с распределением объектов по классам и сделать выводы о том, как соотношение классов влияет на качество модели.

1) Зададим веса объектам так, чтобы **соотношение классов** с учетом весов объектов изменилось. При xgbclassifier кривая оценок является горизонтальной линией, поэтому для наглядности построим кривую для альтернативного классификатора - randomforestclassifier изменяя в нем class_weight:



При изменении весов классов результаты классификации модели изменяются, лучший результат достигается при весе класса 1 в 0.7131, оптимальным вариантом будет вес класса в 0.5, поскольку при увеличении веса класса 1 от 0.7131 к 1, результаты не увеличиваются.

2) Применим к выборке технологию **undersampling**: для этого нужно убрать из обучения некоторое количество объектов большего класса таким образом, чтобы соотношение классов изменилось. Попробуем не менее трёх различных вариантов undersampling (варианты могут отличаться как по количество отфильтрованных объектов, так и по принципу выборка объектов для отсеивания из выборки).

undersampling	prc_auc	roc_auc
RandomUnderSampler	0.176577	0.719175
TomekLinks	0.189859	0.726386
NeighbourhoodCleaningRule	0.19221	0.726136
EditedNearestNeighbours	0.188334	0.724279

Качество модели, как и результаты классификации при разных стратегиях undersampling меняются не сильно, последние три стратегии оказались близкими по методу удаления объектов. RandomUnderSampler оказалась самой быстрой, однако она оставляет равное кол-во объектов неоттока и оттока, т.е. меньше чем 5000. TomekLinks оказалась самой долгой. Наиболее оптимальной с точки зрения качества оказалась стратегия NeighbourhoodCleaningRule.

4) Отбор признаков

Посмотрим все ли признаки оказались полезными для построения моделей. Проведем процедуру отбора признаков, попробуем разные варианты отбора. В SelectFromModel рассмотрим три варианта отбора признаков: ExtraTreesClassifier, LinearSVC с L1 и LogisticRegression с L1, а в RFE рассмотрим только LinearSVC с L1.

feature_selection	prc_auc	roc_auc
SelectFromModel		
ExtraTreesClassifier(n_estimators = 10)	0.190625	0.72564
LinearSVC(penalty="l1")	0.193742	0.726784
LogisticRegression(penalty = 'l1')	0.191718	0.726947
RFE (LinearSVC(penalty="l1"))		
n_features_to_select = 60	0.1899	0.724395
n_features_to_select = 100	0.19343	0.727048

Среди моделей, используемых в функции SelectFromModel, самой оптимальной оказалась LinearSVC(penalty="l1") с небольшим отрывом, однако ExtraTreesClassifier убирает больше признаков. Среди двух рассмотренных методик feature_selection, RFE хоть и отбрасывает большее кол-во признаков, является очень долгой, с другой стороны ExtraTreesClassifier оказалась самой оптимальной.

5) Подбор оптимальных параметров модели

Подбор параметров препроцессинга, лучших параметров undersampling и feature_selection:
Параметры по предыдущим результатам (наилучшие из предыдущих подборов):
1) Пре-процессинг:
   1. Заполнение пропущенных значений числовых переменных: mean.
   2. Масштабирование числовых переменных: StandardScaler
   3. Обработка категориальных признаков: OrdinalEncoder.
2) Undersampling: NeighbourhoodCleaningRule.
3) Важность признаков: SelectFromModel(ExtraTreesClassifier(n_estimators = 10))
4) Оптимальные параметры модели классификатора: XGBClassifier (learning_rate= 0.08, max_depth= 4, n_estimators= 100, class_weight='balanced')

Подбор параметров как пре-процессинга, так и самого классификатора немного улучшил качество финальной модели:

	prc_auc	roc_auc
xgboost_baseline	0.186459	0.719646
xgboost_final	0.20612	0.731107

Для дальнейшего улучшения модели можно:

- подобрать все параметры (в parameters_grid_xgb), как препроцессинга, так и самого классификатора на GridSearchCV, хоть это и займет очень много времени, при этом использовать больше вариантов undersampling-а, обработки категориальных признаков, моделей для отбора признаков;
- попробовать подбирать разные пороги отбора признаков в параметрах функций feature_selection_.
- изначально удалить все признаки, пропуски на которых составляют >75%, к примеру.
- использовать методы понижения размерности: PCA, MDS, t-SNE, а затем классификатор knn при меньших количествах объектов.

6) Важность признаков

Рассмотрим, какие признаки внесли наибольший вклад в модель, а какие наименьший?
Определим важность **числовых** переменных:
Сначала определим какие признаки остались после feature_selection, а затем определим, какие признаки после feature_selection оставил сам классификатор в feature_importances:

После применения feature_selection от **174** числовых признаков, осталось **39**:

Var6	Var35	Var78	Var119	Var144
Var7	Var38	Var81	Var123	Var149
Var13	Var57	Var83	Var125	Var153
Var21	Var65	Var85	Var126	Var160
Var22	Var72	Var94	Var132	Var163
Var24	Var73	Var109	Var133	Var181
Var25	Var74	Var112	Var134	Var189
Var28	Var76	Var113	Var140	

С одной стороны, по описательному анализу в начале, разреженность лишь 42 из 190 числовых переменных ниже 90%. С другой стороны, в данный момент, то, что из 174 числовых переменных модель в feature_selection оставила только 39, говорит о том, что она могла удалить разреженные переменные, пропуски которых заменены на среднюю, и в результате имеющие низкую дисперсию.

Наиболее важные числовые признаки в модели:

Vars	clf_importance_mean
Var132	0.045196
Var76	0.035448
Var74	0.028882
Var126	0.023006
Var144	0.019525
Var22	0.018859
Var113	0.018224
Var140	0.01708
Var13	0.017038
Var123	0.016731

Наименее важные числовые признаки в модели:

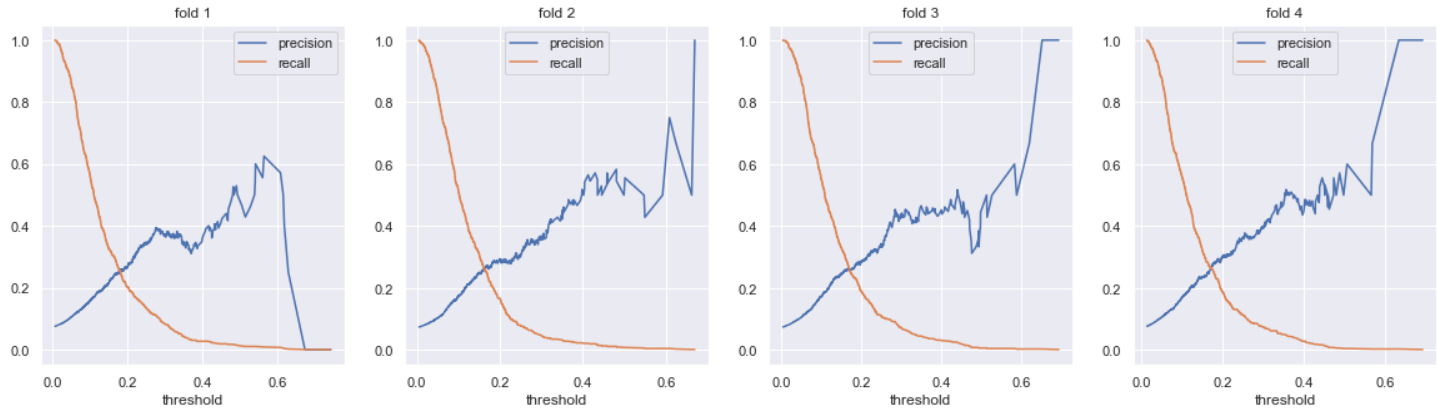
Vars	clf_importance_mean
Var81	0.012059
Var160	0.011823
Var28	0.011417
Var109	0.011066
Var65	0.009947
Var149	0.009664
Var35	0.009485
Var24	0.009048
Var133	0.008358
Var57	0.007104

По результатам feature_importances_ одна из двух ниже рассмотренных числовых переменных оказалась наиболее важной, а другая - наоборот, в добавок по наблюдениям в описательном анализе, в попарных координатах числовых переменных, интересно было посмотреть на их связь:

- Var57-Var126: объекты занимали определенную прямоугольную область.
- Var35-Var132: значения собираются выше или ниже линии под углом.
- Var133-Var76: значения собираются выше или ниже диагонали на графике.

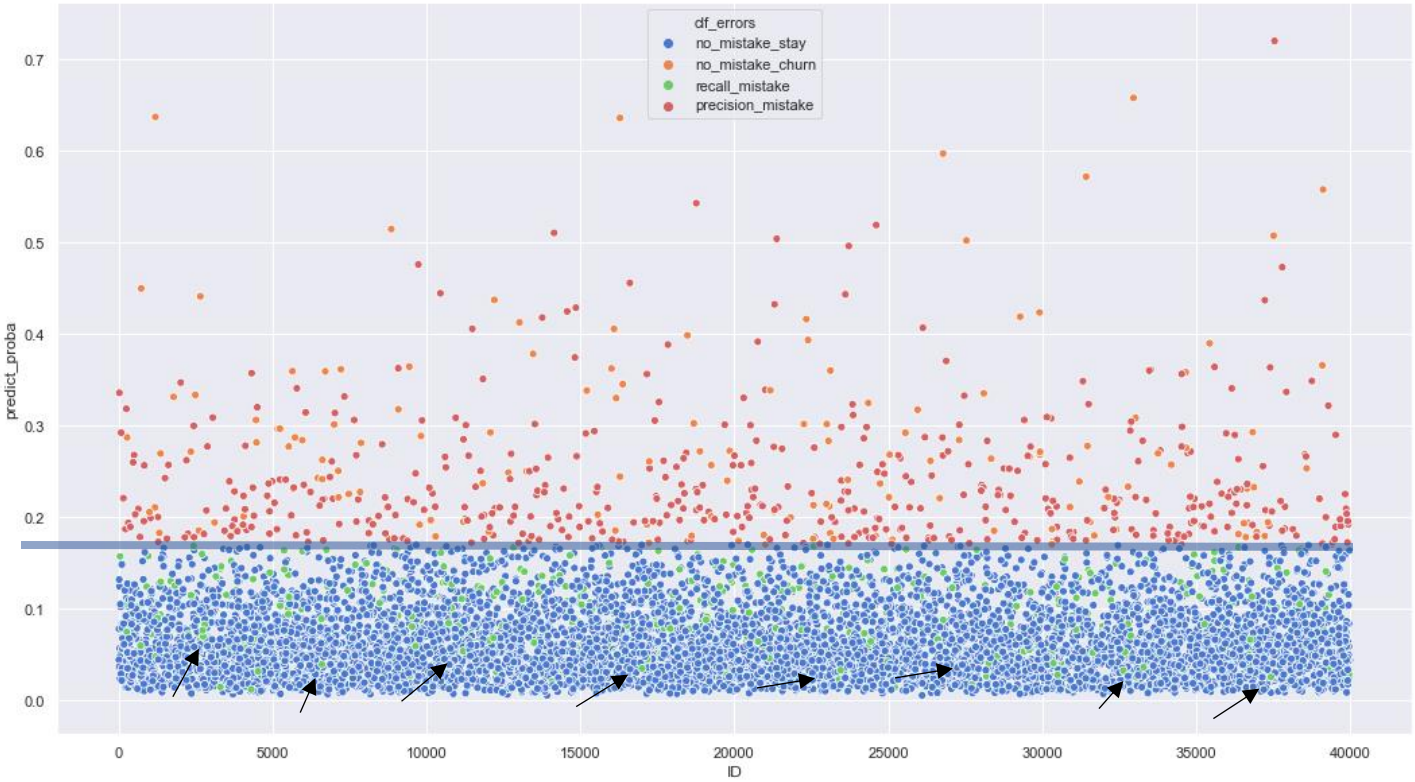
7) Анализ объектов, на которых достигается наибольшая ошибка классификации.

Посмотрим на каких объектах достигается наибольшая ошибка классификации и есть ли межу этими объектами что-то общее и видны ли какие-либо закономерности. Предположим, почему наибольшая ошибка достигается именно на этих объектах. В данном случае "наибольшую" ошибку можно понимать, как отнесение объекта с чужому классу с большой долей уверенности (с высокой вероятностью).



Как видно, кривая recall, в отличие от кривой precision на всех фолдах, очень резкая, поэтому при подобранной максимальной prc_auc, придется подбирать порог бинаризации ниже стандартного 0.5.

Рассмотрим на контрольной выборке как ведут себя оптимальные по PRC-AUC recall и precision: выберем, к примеру, порог, при котором recall и precision примерно равны: T = 0.17



Ошибки recall уходят далеко в глубь, реально склонных к оттоку тяжело различить, так как площадь под PRC не слишком то и большая ~0.21, поэтому придется регулировать порог бинаризации в зависимости от экономических соображений.

True Predicted	-1	1	All
-1	6947	451	7398
1	458	144	602
All	7405	595	8000

Приоритетным параметром для рассмотрения является recall, поэтому будем понимать "наибольшую" ошибку как отнесение объекта к чужому классу с наименьшей вероятностью (к примеру, ниже 2%):

ID	predict_proba	predict_label	label	clf_errors
29190	0.011146	-1	1	recall_mistake
4309	0.011875	-1	1	recall_mistake
25211	0.014499	-1	1	recall_mistake
3308	0.014686	-1	1	recall_mistake
34104	0.016056	-1	1	recall_mistake
27730	0.017725	-1	1	recall_mistake
16471	0.017851	-1	1	recall_mistake
4135	0.01907	-1	1	recall_mistake

Для просмотра объектов, на которых достигается данный вид ошибок, выберем наиболее важные признаки по модели:

по числовым:

	Var6	Var7	Var13	Var21	Var22	Var24	Var25	Var28	Var35	Var38	Var57	Var65	Var72	Var73	Var74	Var76	Var78	Var81	Var83	Var85
count	7136	7142	7142	7136	7229	6862	7229	7228	7229	7230	**8000**	**7142**	**4433**	**8000**	**7142**	**7230**	**7229**	**7140**	7229	7229
mean	1317	7	1224	236	291	5	98	224	1	2580000	**4**	**15**	**4**	**67**	**98**	**1490000**	**0**	**105000**	20	9
std	2431	6	2410	503	626	10	191	94	3	3040000	**2**	**10**	**2**	**54**	**277**	**1850000**	**2**	**114000**	71	20
min	0	0	0	0	0	0	0	0	0	0	**0**	**9**	**3**	**4**	**0**	**0**	**0**	**0**	0	0
25%	518	0	0	112	135	0	16	167	0	6860	**2**	**9**	**3**	**24**	**0**	**90900**	**0**	**16500**	0	0
50%	854	7	228	144	180	2	48	220	0	1220000	**4**	**9**	**3**	**52**	**7**	**898000**	**0**	**73000**	10	4
75%	1442	7	1623	232	285	6	120	264	0	4570000	**5**	**18**	**6**	**104**	**91**	**2310000**	**0**	**184000**	25	10
max	82264	35	51536	20652	25815	448	5864	2434	35	18100000	**7**	**126**	**18**	**260**	**13783**	**19400000**	**27**	**1700000**	2535	932

ID	Var6	Var7	Var13	Var21	Var22	Var24	Var25	Var28	Var35	Var38	Var57	Var65	Var72	Var73	Var74	Var76	Var78	Var81	Var83	Var85
29190	1253	21	1600	312	390	20	168	167	0	0	6	27	9	104	301	315496	0	24576	0	22
4309	1246	7	4712	180	225	4	128	177	0	3762	2	9	NaN	164	175	1531200	0	20318	0	10
25211	2219	14	336	320	400	0	112	98	30	5183412	6	18	6	64	7	1000688	3	42799	30	6
3308	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	6	NaN	NaN	6	NaN	NaN	NaN	NaN	NaN	NaN
34104	1750	7	504	220	275	2	56	234	0	5033550	5	9	3	76	182	1148304	0	17442	25	4
27730	1204	7	0	136	170	0	8	322	0	1268328	4	9	3	34	56	6220800	0	130697	0	0
16471	770	0	0	64	80	4	8	265	0	8450580	1	9	3	18	0	638056	0	55356	10	4
4135	4879	35	1684	568	710	32	72	187	0	76404	5	45	6	228	70	1818888	9	24744	75	104

По данным объектам на числовых признаках тяжело что-либо предположить, можно сравнивать средние и 75% персентили, к примеру.

по категориальным:

	Var191	Var192	Var193	Var194	Var195	Var196	Var197	Var198	Var199	Var200	...	Var225	Var226	Var227	Var228	Var229
count	137	7942	8000	1997	8000	8000	7973	8000	8000	3925	...	3818	8000	8000	8000	3442
unique	1	287	38	3	16	4	182	1848	1549	3548	...	3	23	7	27	4
top	r__I	oUPBcmzkzH	RO12	SEuy	taul	1K8T	0Xwj	fhk21Ss	r83_sZi	Uw6SDm8	...	ELof	FSa2	RAYp	F2FyR07IdsN7I	am7c
freq	137	67	5727	1960	7671	7926	745	736	144	10	...	1821	1304	5555	5190	1872

ID	Var191	Var192	Var193	Var194	Var195	Var196	Var197	Var198	Var199	Var200	...	Var225	Var226	Var227	Var228	Var229
29190	NaN	gE8Wq6edh4	RO12	NaN	taul	1K8T	PGNs	iJzviRg	3cO76t0	nFLuZLx	...	ELof	453m	ZI9m	R4y5gQQWY8OodqDV	am7c
4309	NaN	8I1r4RXXnK	2Knk1KF	NaN	taul	1K8T	uErj	fhk21Ss	n1zVHpT8NN	yP01_Zt	...	ELof	Xa3G	ZI9m	TCU50_Yjmm6GIBZ0lL_	am7c
25211	NaN	xOXr4RXktW	RO12	NaN	taul	1K8T	0Y9G	gmq88LT	76MiCML	6Lq_UvT	...	ELof	7FJQ	RAYp	F2FyR07IdsN7I	NaN
3308	NaN	vAsTmBfHUn	RO12	NaN	taul	1K8T	TyGl	65yJUS7	BZ1DVSQNvCeg4	NaN	...	NaN	Qu4f	RAYp	F2FyR07IdsN7I	NaN
34104	NaN	kMJt0GoOh3	RO12	SEuy	taul	1K8T	487l	1nw8HPr	Cy0RhGn	bXJihTs	...	kG3k	Qu4f	RAYp	F2FyR07IdsN7I	am7c
27730	NaN	1qmr4RQxul	RO12	NaN	taul	1K8T	EJC9	34dq7r9	JtWlK1RmE1	NaN	...	NaN	FSa2	RAYp	F2FyR07IdsN7I	NaN
16471	NaN	DQlrEyXXnK	RO12	NaN	taul	1K8T	TyGl	LG0JHBC	JSmlJZ_	NaN	...	NaN	Aoh3	RAYp	F2FyR07IdsN7I	NaN
4135	NaN	zcRZptzip9	TW8dXluYzKpktZjuY8kSPBaZa	SEuy	CiJsoa4TQ0rGHlMp	1K8T	487l	fhk21Ss	kqNaUgs	m4emXBZ	...	kG3k	Qu4f	ZI9m	VjDE	NaN

Можно предположить, что наибольшая ошибка recall достигается на этих объектах из-за наличия пустых ячеек в следующих категориальных переменных:

Var191, Var194, Var201, Var203, Var213, Var215, Var224

4. Ожидаемый экономический эффект от использования модели.

Построим простую экономическую модель для оценки эффекта от внедрения полученного решения на практике.

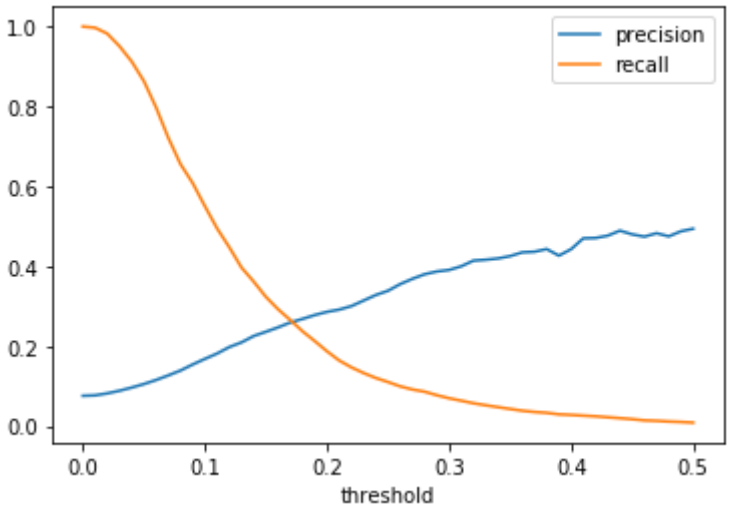
Например, введем следующие параметры:

- сколько денег в среднем приносит один пользователь в месяц: **тарифный план 50 евро**.
- сколько денег в среднем мы будем вкладывать в удержание одного пользователя: **0-20 евро**.
- с какой вероятностью пользователь примет наше предложение: **0-38%**.
- сколько пользователей (например, топ 1% или топ 25% согласно ранжированию по нашей модели) будет участвовать в кампании: **будем искать оптимальный от 0 до 100%**.

Рассчитаем экономический эффект в зависимости от порога бинаризации и от уровня вложений в удержание абонента:

1.Рассмотрим true positive, false positive и false negative при разных порогах бинаризации от 0 до 0.5, на верхних графиках кривая recall находится около нуля уже к 0.5, поэтому дальше порога 0.5 нет смыла рассматривать.

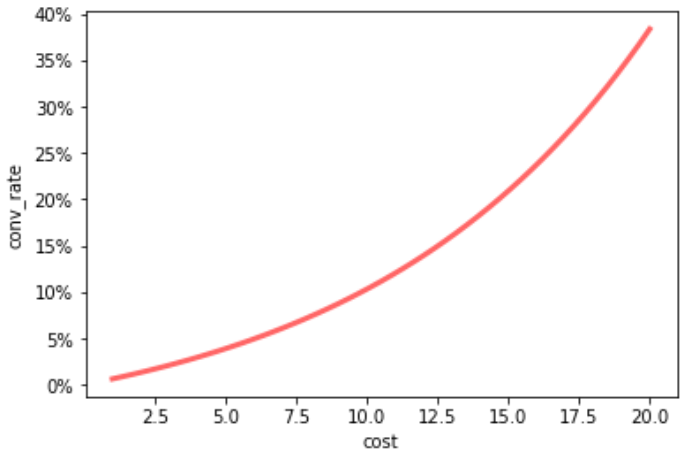
Рассчитаем средние precision и recall по 4-м фолдам тестовой выборки для получения показателей для оптимизации:



2. Создадим таблицу показателей (доли от выборки) при разных порогах бинаризации. Разделим на кол-во пользователей в контрольной выборке (8000) для получения долей. Показатели по долям true positive, false positive и false negative возьмем из контрольной выборки, будем экстраполировать их на весь датасет:

threshold	0	0.01	0.02	0.03	0.04	0.05	...	0.5
precision	0.075366	0.076485	0.081386	0.088004	0.095937	0.104748	...	0.493737
recall	1	0.99706	0.982359	0.950866	0.912643	0.86354	...	0.008151
true_churn	0.074375	0.074375	0.073375	0.0705	0.0675	0.064625	...	0.001125
precision_mistake	0.925625	0.917125	0.837375	0.738375	0.634875	0.53825	...	0.000625
predicted_churn	1	0.9915	0.91075	0.808875	0.702375	0.602875	...	0.00175
recall_mistake	0	0	0.001	0.003875	0.006875	0.00975	...	0.07325

3. Предположим, что конверсия помимо многих факторов экспоненциально зависит от кол-ва денег (скидки, разные пакеты бесплатных услуг и т.д.), вложенных для удержания абонентов:



конверсия не слишком высокая 0-40% и кол-во денег 0-20 евро, т.е. доходит до 40% от примерной средней месячной абонентской платы (необходимо учесть, что эти данные 2009 г.):

conv_rate = np.exp(0.1*cost)*0.06-0.06

4. Теперь создадим итоговую таблицу экономического эффекта при разных показателях порога бинаризации и уровня вложений, применив формулу:

Экономический эффект = (true_positive x конверсия x абон_плата x 12мес) / ((спрогнозированный_отток + false_negative) x издержки)

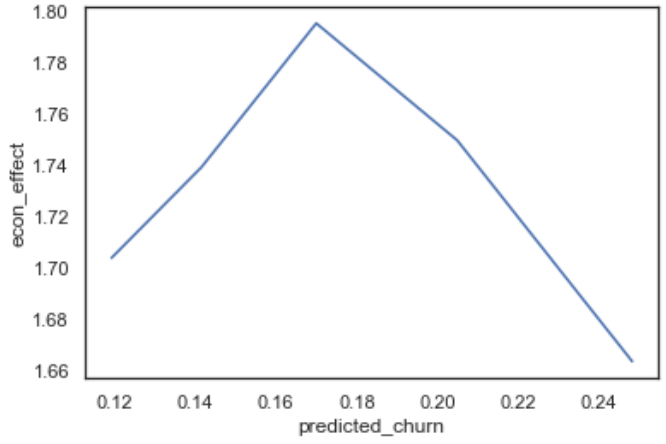
Экономическим эффектом будет не net profit, а отдача на вложенный капитал с минимальным оттоком, поэтому будем искать максимальное соотношение при оптимальном пороге. В формуле (абон_плата x 12мес) - сколько денег может принести удержанный пользователь (удержание или потеря длятся по 12 месяцев после проведения кампании), без дисконтирования (будем считать, что на тот момент процентные ставки находились около нуля).

Рассмотрим срез итоговой таблицы (ближайшие точки к максимуму):

cost	conv_rate	...	0.1	0.11	0.12	0.13	0.14	...
...	...	...	...	...	...	...	...	...
18.48	0.320827	...	1.50655	1.584762	1.626223	1.575306	1.543178	...
18.86	0.335577	...	1.544063	1.624222	1.666716	1.614531	1.581603	...
19.24	0.350898	...	1.582671	1.664834	1.708391	1.654901	1.62115	...
19.62	0.366812	...	1.622407	1.706634	1.751284	1.696451	1.661853	...
20	0.383343	...	1.663309	1.749659	1.795434	1.739219	1.703748	...

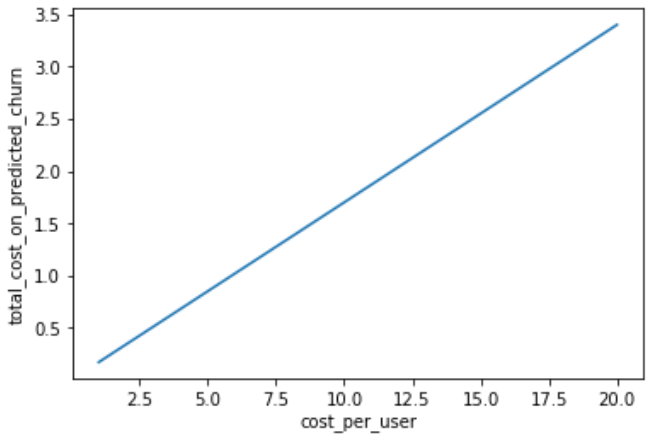
По верхней таблице найдем макс значение: 1.7954. При данной модели оптимальный размер топа при увеличении издержек не меняется.

Рассмотрим, как меняется экономическом эффект при разном оптимальном размере топа:



threshold	0.10	0.11	0.12	0.13	0.14
precision	0.17	0.18	0.20	0.21	0.23
recall	0.55	0.49	0.45	0.40	0.36
true_churn	0.04	0.04	0.03	0.03	0.03
precision_mistake	0.21	0.17	0.14	0.11	0.09
predicted_churn	0.25	0.20	0.17	0.14	0.12
recall_mistake	0.03	0.04	0.04	0.05	0.05

Таким образом, согласно модели, для проведения кампании по удержанию и достижению максимального экономического эффекта необходимо выбрать топ 17% пользователей склонных к оттоку, вероятности оттока которых выше порога бинаризации 0.12.



Уровень издержек на одного пользователя компании необходимо выбрать самостоятельно согласно ее бюджету, применимого к этим топ 17% пользователей (здесь любой уровень издержек экономически оправдан, будь cost=1 eur или все 20 eur):

Всегда ли применение модели экономически оправдано? Приведем пример набора значений параметров, при которых применение модели перестает быть оправданным.

Рассмотрим **net profit** от проведения кампании по удержанию:

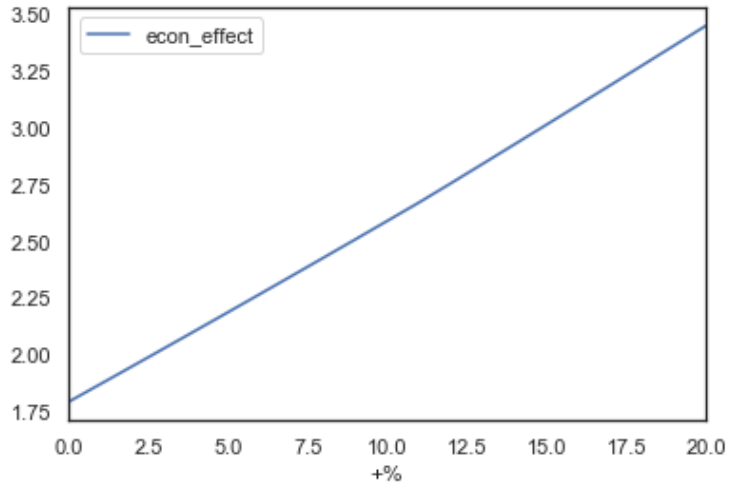
net profit = (true_positive x конверсия x абон_плата x 12мес) - (спрогнозированный_отток x издержки)

cost	conv_rate	0	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.1	0.11	0.12	0.13	0.14	0.15	0.16	0.17	0.18	0.19	0.2	0.21	0.22	...	0.50
1.00	0.01	-0.7	-0.7	-0.6	-0.5	-0.4	-0.4	-0.3	-0.2	-0.2	-0.1	-0.1	-0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
1.38	0.01	-1.0	-1.0	-0.9	-0.7	-0.6	-0.5	-0.4	-0.3	-0.2	-0.2	-0.1	-0.1	-0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
1.76	0.01	-1.2	-1.2	-1.1	-0.9	-0.8	-0.6	-0.5	-0.4	-0.3	-0.2	-0.2	-0.1	-0.1	-0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
2.14	0.01	-1.5	-1.5	-1.3	-1.1	-0.9	-0.7	-0.6	-0.4	-0.3	-0.2	-0.2	-0.1	-0.1	-0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
2.52	0.02	-1.8	-1.7	-1.5	-1.3	-1.1	-0.9	-0.7	-0.5	-0.4	-0.3	-0.2	-0.1	-0.1	-0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
2.90	0.02	-2.0	-2.0	-1.8	-1.5	-1.2	-1.0	-0.8	-0.6	-0.4	-0.3	-0.2	-0.1	-0.1	-0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
3.28	0.02	-2.2	-2.2	-2.0	-1.7	-1.4	-1.1	-0.8	-0.6	-0.5	-0.3	-0.2	-0.2	-0.1	-0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
3.66	0.03	-2.5	-2.4	-2.2	-1.8	-1.5	-1.2	-0.9	-0.7	-0.5	-0.4	-0.3	-0.2	-0.1	-0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	...	0.0
4.04	0.03	-2.7	-2.7	-2.4	-2.0	-1.6	-1.3	-1.0	-0.7	-0.5	-0.4	-0.3	-0.2	-0.1	-0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.1	0.1	...	0.0
4.42	0.03	-2.9	-2.9	-2.6	-2.2	-1.8	-1.4	-1.1	-0.8	-0.6	-0.4	-0.3	-0.2	-0.1	-0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.1	0.1	...	0.0
4.80	0.04	-3.2	-3.1	-2.7	-2.3	-1.9	-1.5	-1.1	-0.8	-0.6	-0.4	-0.3	-0.2	-0.1	-0.1	0.0	0.0	0.0	0.0	0.0	0.1	0.1	0.1	0.1	...	0.0
5.18	0.04	-3.4	-3.3	-2.9	-2.5	-2.0	-1.5	-1.2	-0.8	-0.6	-0.4	-0.3	-0.2	-0.1	0.0	0.0	0.0	0.0	0.0	0.1	0.1	0.1	0.1	0.1	...	0.0
5.56	0.05	-3.6	-3.5	-3.1	-2.6	-2.1	-1.6	-1.2	-0.9	-0.6	-0.4	-0.3	-0.2	-0.1	0.0	0.0	0.0	0.0	0.1	0.1	0.1	0.1	0.1	0.1	...	0.0
5.94	0.05	-3.8	-3.7	-3.3	-2.7	-2.2	-1.7	-1.3	-0.9	-0.6	-0.4	-0.3	-0.1	0.0	0.0	0.0	0.0	0.1	0.1	0.1	0.1	0.1	0.1	0.1	...	0.0
6.32	0.05	-4.0	-3.9	-3.4	-2.9	-2.3	-1.8	-1.3	-0.9	-0.6	-0.4	-0.3	-0.1	0.0	0.0	0.0	0.0	0.1	0.1	0.1	0.1	0.1	0.1	0.1	...	0.0
6.70	0.06	-4.1	-4.1	-3.6	-3.0	-2.4	-1.8	-1.4	-0.9	-0.6	-0.4	-0.3	-0.1	0.0	0.0	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.2	...	0.0
7.08	0.06	-4.3	-4.3	-3.7	-3.1	-2.5	-1.9	-1.4	-0.9	-0.6	-0.4	-0.2	-0.1	0.0	0.0	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.2	0.2	...	0.0
7.46	0.07	-4.5	-4.4	-3.9	-3.2	-2.5	-1.9	-1.4	-0.9	-0.6	-0.4	-0.2	-0.1	0.0	0.1	0.1	0.1	0.1	0.2	0.2	0.2	0.2	0.2	0.2	...	0.0
7.84	0.07	-4.7	-4.6	-4.0	-3.3	-2.6	-2.0	-1.5	-0.9	-0.6	-0.4	-0.2	0.0	0.1	0.1	0.1	0.1	0.2	0.2	0.2	0.2	0.2	0.2	0.2	...	0.0
8.22	0.08	-4.8	-4.7	-4.1	-3.4	-2.7	-2.0	-1.5	-0.9	-0.6	-0.4	-0.2	0.0	0.1	0.1	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	...	0.0
8.60	0.08	-5.0	-4.9	-4.2	-3.5	-2.7	-2.0	-1.5	-0.9	-0.6	-0.3	-0.1	0.0	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.3	0.3	...	0.0
8.98	0.09	-5.1	-5.0	-4.3	-3.6	-2.8	-2.0	-1.5	-0.9	-0.5	-0.3	-0.1	0.1	0.2	0.2	0.2	0.2	0.3	0.3	0.3	0.3	0.3	0.3	0.3	...	0.0
9.36	0.09	-5.2	-5.1	-4.4	-3.6	-2.8	-2.0	-1.5	-0.9	-0.5	-0.2	0.0	0.1	0.2	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	...	0.0
9.74	0.10	-5.3	-5.2	-4.5	-3.7	-2.8	-2.0	-1.5	-0.8	-0.5	-0.2	0.0	0.2	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	...	0.0
10.12	0.11	-5.4	-5.3	-4.6	-3.7	-2.9	-2.0	-1.4	-0.8	-0.4	-0.1	0.1	0.3	0.4	0.4	0.4	0.3	0.4	0.4	0.4	0.4	0.4	0.4	0.4	...	0.1
10.50	0.11	-5.5	-5.4	-4.7	-3.8	-2.9	-2.0	-1.4	-0.7	-0.4	-0.1	0.1	0.3	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	...	0.1
10.88	0.12	-5.6	-5.5	-4.7	-3.8	-2.9	-2.0	-1.4	-0.7	-0.3	0.0	0.2	0.4	0.5	0.5	0.5	0.4	0.5	0.5	0.4	0.4	0.4	0.4	0.4	...	0.1
11.26	0.13	-5.7	-5.6	-4.8	-3.8	-2.8	-1.9	-1.3	-0.6	-0.2	0.1	0.3	0.5	0.6	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	...	0.1
11.64	0.13	-5.7	-5.6	-4.8	-3.8	-2.8	-1.9	-1.2	-0.5	-0.1	0.1	0.3	0.5	0.6	0.6	0.6	0.5	0.6	0.6	0.5	0.5	0.5	0.5	0.5	...	0.1
12.02	0.14	-5.8	-5.7	-4.8	-3.8	-2.8	-1.8	-1.2	-0.5	0.0	0.2	0.4	0.6	0.7	0.7	0.7	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	...	0.1
12.40	0.15	-5.8	-5.7	-4.8	-3.8	-2.7	-1.8	-1.1	-0.4	0.1	0.3	0.5	0.7	0.8	0.8	0.7	0.7	0.7	0.7	0.6	0.6	0.6	0.6	0.6	...	0.1
12.78	0.16	-5.8	-5.7	-4.8	-3.8	-2.7	-1.7	-1.0	-0.3	0.2	0.4	0.6	0.8	0.9	0.8	0.8	0.7	0.7	0.7	0.7	0.7	0.6	0.7	0.6	...	0.1
13.16	0.16	-5.9	-5.7	-4.8	-3.7	-2.6	-1.6	-0.9	-0.1	0.3	0.5	0.7	0.9	1.0	0.9	0.9	0.8	0.8	0.8	0.7	0.7	0.7	0.7	0.7	...	0.1
13.54	0.17	-5.8	-5.7	-4.7	-3.7	-2.5	-1.5	-0.8	0.0	0.4	0.7	0.9	1.0	1.1	1.0	1.0	0.9	0.9	0.8	0.8	0.8	0.7	0.8	0.7	...	0.1
13.92	0.18	-5.8	-5.7	-4.7	-3.6	-2.4	-1.4	-0.7	0.1	0.5	0.8	1.0	1.2	1.2	1.1	1.1	1.0	1.0	0.9	0.9	0.8	0.8	0.8	0.8	...	0.1
14.30	0.19	-5.8	-5.7	-4.6	-3.5	-2.3	-1.2	-0.5	0.3	0.7	0.9	1.1	1.3	1.3	1.2	1.2	1.0	1.0	1.0	0.9	0.9	0.9	0.9	0.8	...	0.1
14.68	0.20	-5.7	-5.6	-4.5	-3.4	-2.2	-1.1	-0.4	0.4	0.8	1.1	1.3	1.4	1.5	1.3	1.3	1.1	1.1	1.1	1.0	1.0	0.9	0.9	0.9	...	0.1
15.06	0.21	-5.7	-5.5	-4.4	-3.3	-2.1	-0.9	-0.2	0.6	1.0	1.3	1.4	1.6	1.6	1.5	1.4	1.2	1.2	1.1	1.1	1.0	1.0	1.0	1.0	...	0.1
15.44	0.22	-5.6	-5.4	-4.3	-3.1	-1.9	-0.7	0.0	0.8	1.2	1.4	1.6	1.7	1.8	1.6	1.5	1.3	1.3	1.2	1.2	1.1	1.1	1.1	1.0	...	0.1
15.82	0.23	-5.5	-5.3	-4.2	-3.0	-1.7	-0.5	0.1	1.0	1.4	1.6	1.7	1.9	1.9	1.7	1.6	1.4	1.4	1.3	1.2	1.2	1.1	1.1	1.1	...	0.1
16.20	0.24	-5.3	-5.2	-4.0	-2.8	-1.5	-0.3	0.4	1.2	1.6	1.8	1.9	2.1	2.1	1.8	1.7	1.5	1.5	1.4	1.3	1.3	1.2	1.2	1.2	...	0.1
16.58	0.26	-5.2	-5.1	-3.9	-2.6	-1.3	-0.1	0.6	1.4	1.8	2.0	2.1	2.2	2.2	2.0	1.8	1.6	1.6	1.5	1.4	1.3	1.3	1.3	1.2	...	0.1
16.96	0.27	-5.0	-4.9	-3.7	-2.4	-1.1	0.1	0.8	1.7	2.0	2.2	2.3	2.4	2.4	2.1	2.0	1.8	1.7	1.6	1.5	1.4	1.4	1.4	1.3	...	0.2
17.34	0.28	-4.9	-4.7	-3.5	-2.2	-0.8	0.4	1.1	1.9	2.3	2.5	2.5	2.6	2.6	2.3	2.1	1.9	1.8	1.7	1.6	1.5	1.5	1.4	1.4	...	0.2
17.72	0.29	-4.6	-4.5	-3.2	-1.9	-0.6	0.7	1.3	2.2	2.6	2.7	2.8	2.9	2.8	2.5	2.3	2.0	2.0	1.8	1.7	1.6	1.5	1.5	1.5	...	0.2
18.10	0.31	-4.4	-4.3	-3.0	-1.7	-0.3	1.0	1.6	2.5	2.8	3.0	3.0	3.1	3.0	2.7	2.4	2.2	2.1	1.9	1.8	1.7	1.6	1.6	1.6	...	0.2
18.48	0.32	-4.2	-4.0	-2.7	-1.4	0.0	1.3	1.9	2.8	3.1	3.2	3.3	3.3	3.2	2.8	2.6	2.3	2.2	2.1	1.9	1.8	1.7	1.7	1.6	...	0.2
18.86	0.34	-3.9	-3.7	-2.4	-1.1	0.3	1.6	2.3	3.1	3.4	3.5	3.5	3.6	3.4	3.0	2.8	2.5	2.4	2.2	2.1	1.9	1.8	1.8	1.7	...	0.2
19.24	0.35	-3.6	-3.4	-2.1	-0.7	0.7	2.0	2.6	3.4	3.8	3.8	3.8	3.8	3.7	3.2	3.0	2.6	2.5	2.3	2.2	2.0	1.9	1.9	1.8	...	0.2
19.62	0.37	-3.3	-3.1	-1.7	-0.4	1.1	2.4	3.0	3.8	4.1	4.1	4.1	4.1	3.9	3.5	3.2	2.8	2.7	2.5	2.3	2.2	2.1	2.0	1.9	...	0.2
20.00	0.38	-2.9	-2.7	-1.3	0.0	1.5	2.8	3.4	4.2	4.5	4.5	4.4	4.4	4.2	3.7	3.4	3.0	2.9	2.6	2.5	2.3	2.2	2.1	2.0	...	0.2

Ярким примером, при котором применение модели перестает быть оправданным, является выбор параметров в области выше диагонали верхней таблицы, т.е. при пороге бинаризации от 0 до 0.19 и всей шкале издержек. Здесь также видно, что при около 100%-м recall (T: 0-0.02), когда выбираем практически всех пользователей, чтобы не упустить никого, применение модели не оправдано при любых издержках.

Также видно, что максимальная прибыль с кампании по удержанию пользователей достигается при максимальных издержках на пороге 0.09, однако макс. отдача на вложенные средства и минимальный отток по пользователям из таблиц сверху достигается на пороге 0.12, поэтому выбрали именно последний порог.

Оценим изменение экономического эффекта от проведения кампании по удержанию пользователей при увеличении качества модели на 1%, 3%.



Процент увеличения качества модели будем считать за **процент снижения кол-ва false positive и false negative**, при том, что без каких-либо манипуляций не меняются: 1) кол-во всего предсказанного оттока, 2) при том же оптимальном пороге - издержки и net profit.
Как видно, при увеличении качества модели на 1% и на 3% экономический эффект не сильно возрастает.

Является ли экономически оправданным вложение средств в улучшение качества модели? На сколько нужно улучшить модель, чтобы это качественно сказалось на экономическом эффекте от удержания?

При одном оптимальном пороге бинаризации чистая прибыль от кампании все та же, а при линейном увеличении качества экономический эффект тоже растет линейно. Экономический эффект может увеличиться в два или в три раза при тех же издержках благодаря более качественной модели, поэтому вложение средств в улучшение качества модели является экономически оправданным. Компании нужно решить, какой процент увеличения качества модели является для нее существенным и достижимым, к примеру, чтобы в полтора раза увеличить экономический эффект, необходимо увеличить качество модели на 11%.