



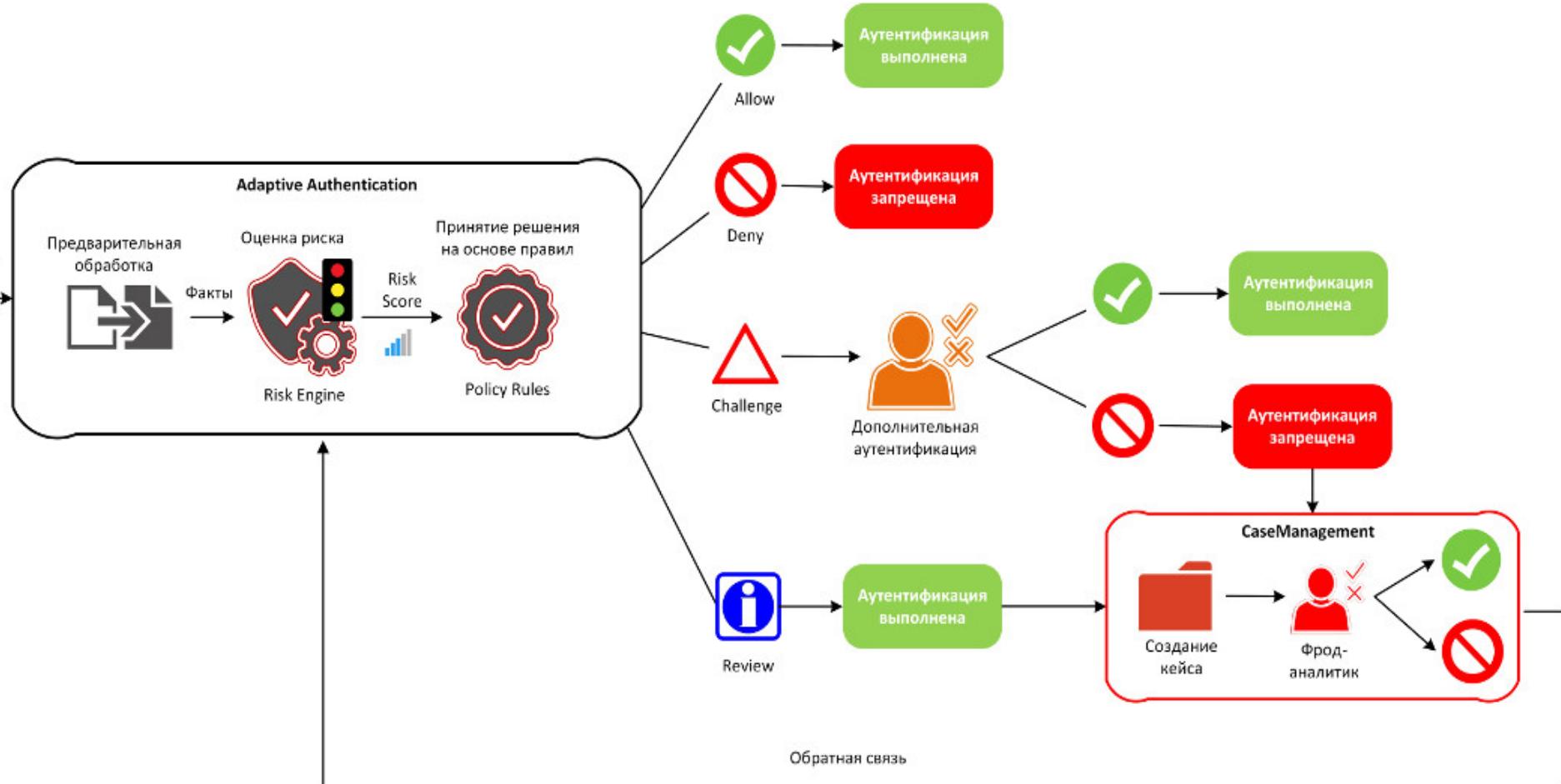
SCS

SBERBANK
CYBER
SECURITY



**Создание и валидация моделей для
предотвращения мошенничества в
банковской сфере**

Система ФМ от вендора – плюсы и минусы (1/3)





Система ФМ от вендора – плюсы и минусы (2/3)



Плюсы:

- Готовые коннекторы и интеграции с различного рода ИТ-решениями (СУБД, API, MQ, ...);
- Наличие всех подсистем настройки и разграничения доступов;
- Большое число доступных к использованию параметров из коробки;
- Готовые к встраиванию библиотеки в web и мобильные приложения;
- Гарантированная скорость работы при заданной нагрузке;
- Дополнительные источники негативной информации across the world;

Минусы:

- Наличие специфичных сервисов/бизнес-процессов заказчика все равно потребуют кастомизации;
- Ограничения платформы по настройке в существующих рамках (если вы не Сбербанк[☺]);
- Модель – черный ящик;
- Практически отсутствует возможность влиять на модель;

[Решение ... использовать дополнительно внешнюю модель собственной разработки](#)



Создание «Мозга» системы фрод-мониторинга (1/2)



Цель: создание модели, осуществляющей анализ каждой транзакции и выдающей «вероятность» мошенничества

Подготовительные этапы:

- **Понимание особенностей задачи и бизнес-процесса, где будет применяться модель.** Как минимум:
 - Онлайн/оффлайн модель;
 - Как будет выглядеть процесса работы с алертами системы и какие ресурсы на это будут выделяться;
 - Период латентности/вызревания фрода;
 - Внедрена ли уже какая-то модель;

- **Анализ доступных данных и выбор общего подхода к решению задачи:**
 - Какой объем транзакций будет подаваться на вход модели и оценочная доля фрода в них;
 - Есть ли обучающая выборка, за какой период. Ее чистота и полнота;
 - Какие данные/источники помимо самих транзакций доступны;



Создание «Мозга» системы фрод-мониторинга (2/2)



На примере задачи построение системы ФМ для физ. лиц в канале Сбербанк Онлайн:

- *Десятки миллионов транзакций и в них тысячные доли процента – мошенничество;*
- *Выделенный пул сотрудников, обрабатывающий фиксированное количество алертов в сутки;*
- *Обучающая выборка доступна за длительный период времени. Чистота обеспечивается отдельно разработанной моделью и группой экспертов, осуществляющих анализ обращений;*
- *Латентность фрода до 10-15 дней*
- *Помимо данных непосредственно текущей транзакции доступны исторические данные по транзакциям клиентов, а также данные клиентов как физических лиц;*
- *Онлайн-модель с жетским SLA (сотни мс)*

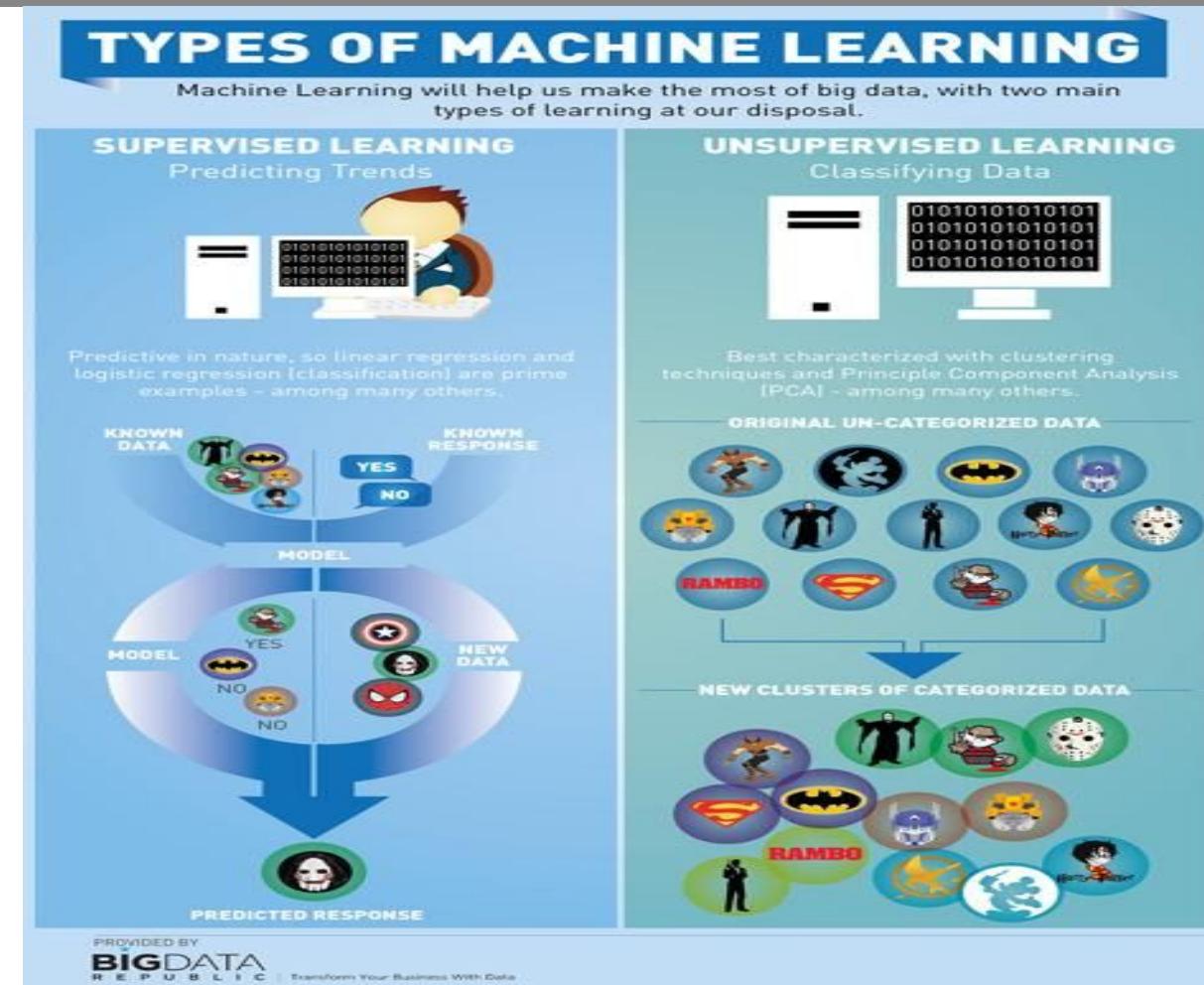
Возможные подходы к решению задачи



Обучение с учителем (supervised learning) – в данном случае мы решаем задачу бинарной классификации и строим модель, результат работы которой – «вероятность», что транзакция мошенническая.

Обучение без учителя (unsupervised learning) – тут задача сводится к поиску аномалий (а точнее поиск «новизны»)

Методы, лежащие на границе 2х описанных – semi-supervised (PU-learning, graph-подходы), reinforcement learning (Q-learning)



Сразу необходимо выбрать метрику, по которой будет оцениваться качество полученной модели.

Влияющие факторы:

Сильная несбалансированность классов - традиционные метрики такие как accuracy (доля правильных ответов) или error rate неприменимы. Более релевантные метрики для такого случая

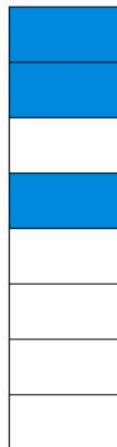
relevant elements

Example: average precision (AP)

- AP: we compute the precision at each relevant position and average them

$$AP = \frac{\sum_{k=1}^{|S|} P(k)}{|S|}$$

$$\frac{P@1 + P@2 + P@4}{3} = \frac{1/1 + 2/2 + 3/4}{3} = 0.92$$



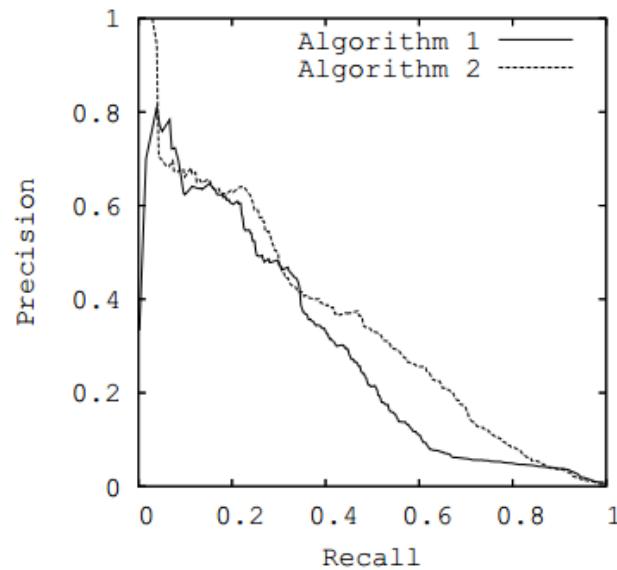
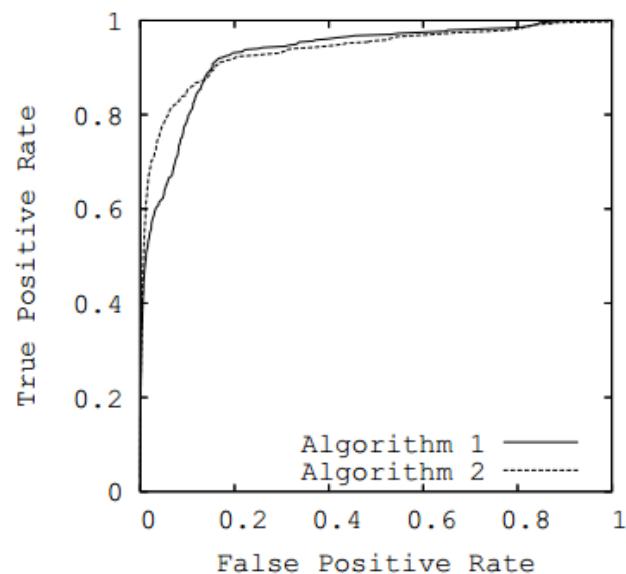
Выбор метрики оценки качества модели



Другие часто используемые метрики в задачах несбалансированных классов, которые не зависят от выбранного порога классификатора:

ROC-curve и ROC AUC

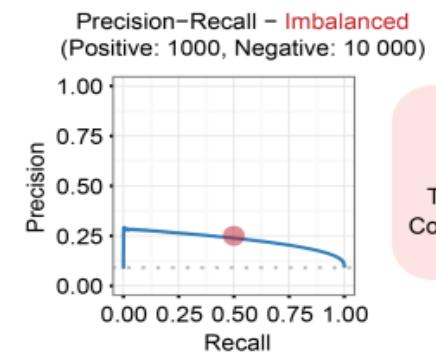
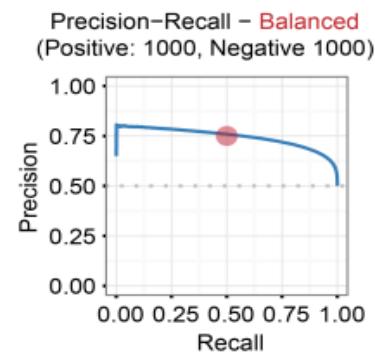
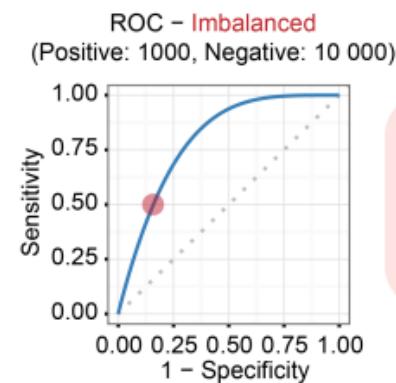
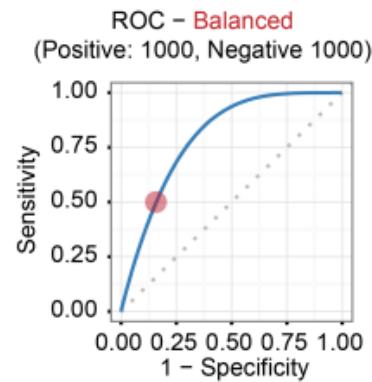
Precision-Recall (PR) curve и PR AUC



Выбор метрики оценки качества модели



Несмотря на то, что использование ROC AUC можно часто встретить в исследованиях проблем несбалансированных классов, использовать этот показатель нужно осмотрительно. PR-AUC в сравнении гораздо лучше читаем.

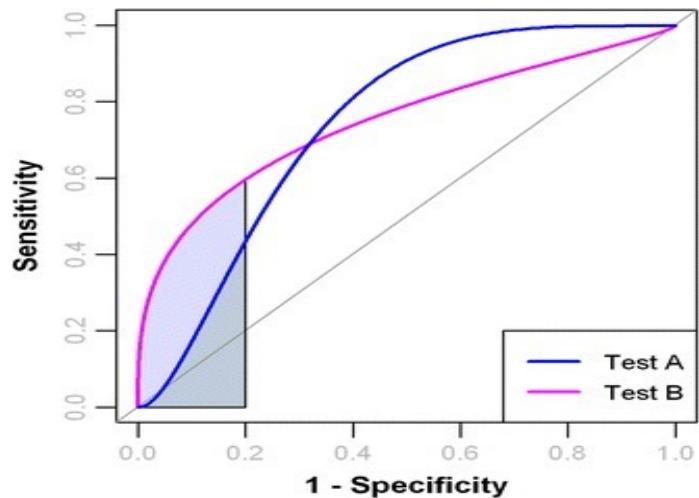


Выбор метрики оценки качества модели



Помимо сильной несбалансированности классов в задаче присутствует ограничение на количество алертов, которое может быть обработано. Описанные ранее метрики (AUC ROC/PR, Precision@K) – интегральные показатели, оценивающие модель на всем диапазоне порогов отсечек.

При указанном ограничении нам же интересна эффективность модели на небольшом интервале, для этого подходят модификации рассмотренных ранее метрик:



$$p@K = \frac{\sum_{k=1}^K r^{true}(\pi^{-1}(k))}{K} = \frac{\text{релевантных элементов}}{K}.$$

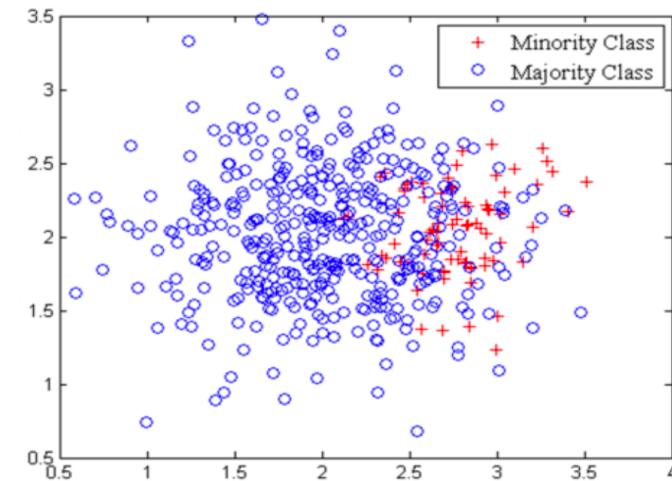
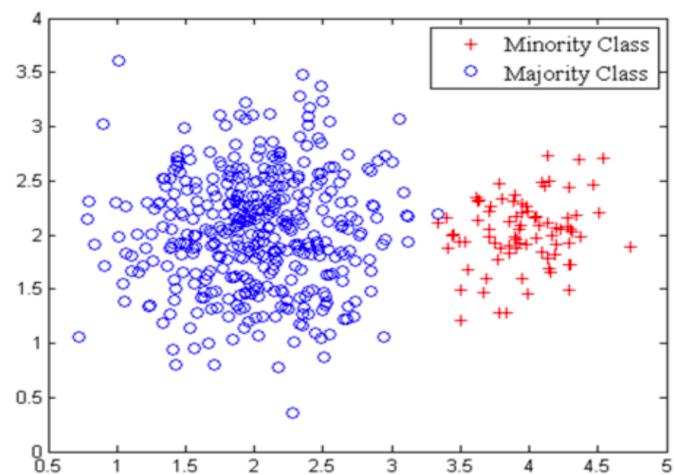
Учет особенностей задачи при разработке модели – *class imbalance and overlapping*



Сильная несбалансированность и перекрытие классов (imbalance and overlapping)

Из постановки задачи видно, что доля минорного класса не превышает тысячных долей процента. Т.е. на 1 однушко мошенническую транзакцию приходятся сотни тысяч легитимных операций. Это говорит о сильной несбалансированности классов.

Помимо этого злоумышленники стараются, чтобы мошеннические транзакции как можно больше походили на транзакции самих клиентов. В результате мошеннические транзакции в пространстве признаков перемешиваются с легитимными транзакциями (*small disjunct, border lines, noisy examples*)





Учет особенностей задачи при разработке модели – *class imbalance and overlapping*



Сильная несбалансированность и перекрытие классов (imbalance and overlapping)

В результате - большинство классификаторов out-of-box, обученных в лоб на таких данных (imbalance and overlapping) обычно показывают плохие результаты, поэтому требуется использование корректирующих методов.

Можно выделить 2 группы таких методов:

- I. *Работа на уровне данных - данные преобразуются на этапе препроцессинга так, чтобы в результате получить более сбалансированный и очищенный набор.*
- II. *Работа на уровне алгоритмов - модификация/расширение существующих и разработка новых алгоритмов с учетом несбалансированного соотношений классов транзакций и минорного класса, использование ансамблей:*

Учет особенностей задачи при разработке модели – *class imbalance and overlapping*

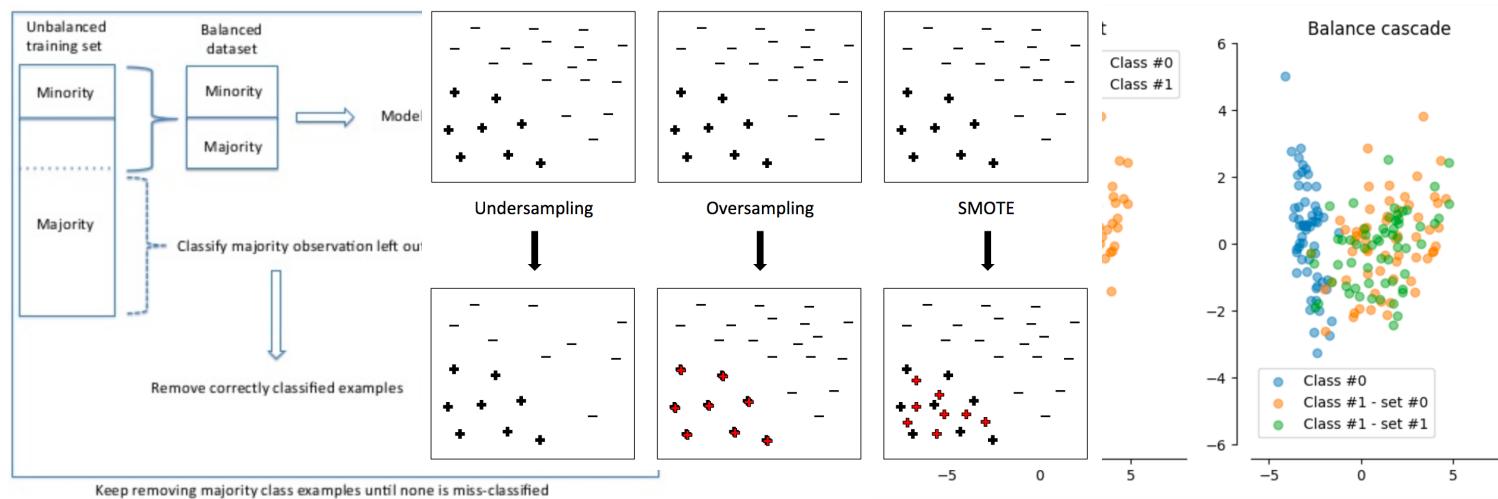


Работа на уровне данных

Методы на уровне данных можно разделить на следующие группы:

- сэмплирование (*under- и oversampling*), в результате которого происходит или прореживание основного класса, или же дублирование/искусственная генерация (*SMOTE*) примеров минорного класса

Но таким базовым методам присущи определенные недостатки, поэтому лучше использовать более продвинутые техники - *Borderline-SMOTE*, *ADASYN*, *EasyEnsemble*, *BalanceCascade*





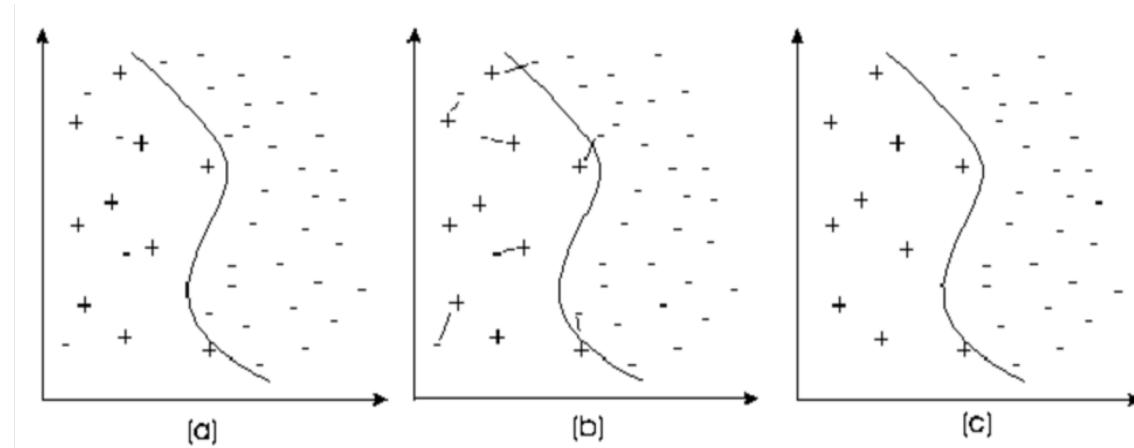
Учет особенностей задачи при разработке модели – *class imbalance and overlapping*



Работа на уровне данных

Методы, основанные на вычислении расстояний (*distance-based*)

В них обычно происходит прореживание основного класса, но с учетом расстояний до границ классов и/или удаление шумовых/граничных примеров каждого из классов. Примеры таких алгоритмов — Tomek link, One Sided Selection, Neighborhood Cleaning Rule.



Работа на уровне данных

Зачастую применяются сразу подходы из обоих групп, например, SMOTE + Tomek.

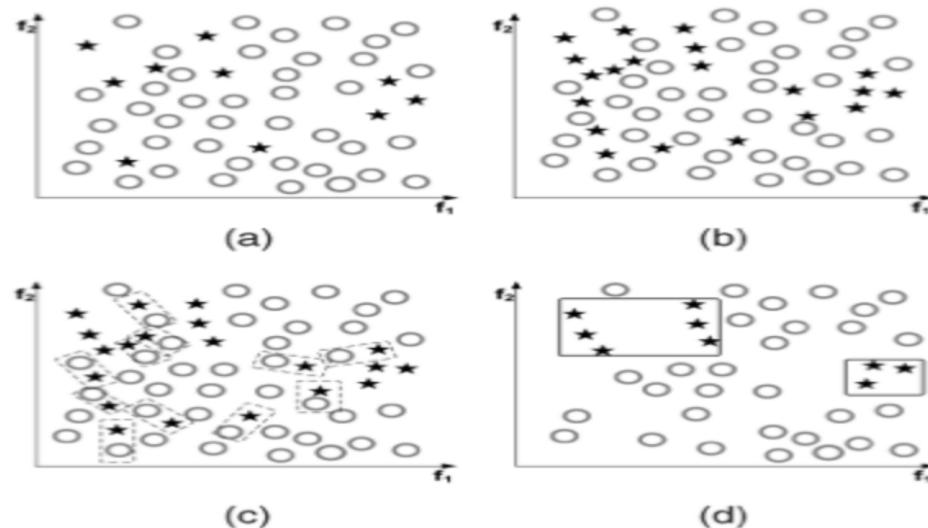


Fig. 5. (a) Original data set distribution. (b) Post-SMOTE data set. (c) The identified Tomek Links. (d) The data set after removing Tomek links.

NOTE: Сэмплирование искривляет апостериорную вероятность, возвращаемую моделью (при подготовке обучающих выборок изменяется соотношение классов по сравнению с реальным распределением в данных).



Учет особенностей задачи при разработке модели – *class imbalance and overlapping*

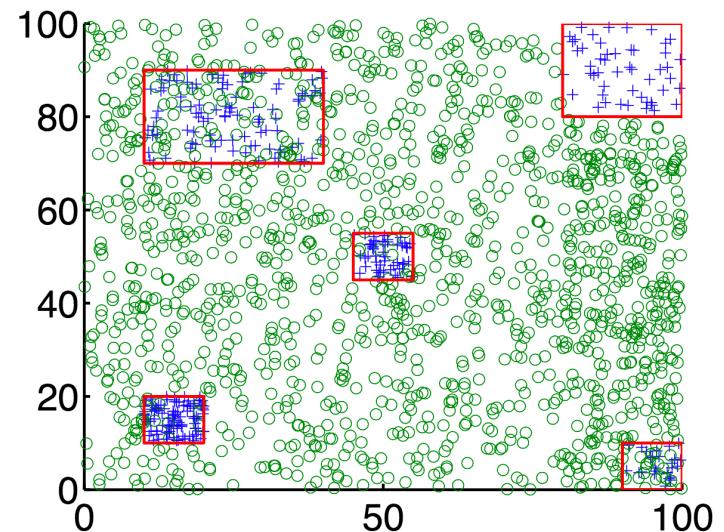


Работа на уровне алгоритмов

- *Imbalance learning* – основная цель алгоритма - повышение точности в детектировании минорного класса за счет использования специальных функций потерь или логики работы.

Примеры алгоритмов: *HDDT, BoxDrawing, BalanceCascade, SMOTEBoost*;

$$d_H(f_+, f_-) = \sqrt{\sum_{j=1}^p \left(\sqrt{\frac{|f_{+j}|}{|f_+|}} - \sqrt{\frac{|f_{-j}|}{|f_-|}} \right)^2}$$





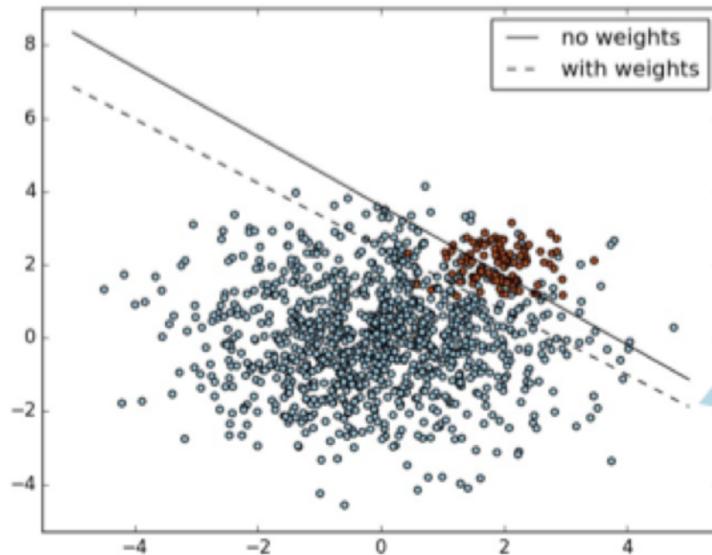
Учет особенностей задачи при разработке модели – *class imbalance and overlapping*



Работа на уровне алгоритмов

- *Cost-sensitive learning* – цель в минимизации общей стоимости ошибок классификации за счет присвоения разной стоимости ошибкам первого и второго рода.

*Примеры: RandomForest, xgboost и многие другие. На самом деле в подавляющем большинстве алгоритмов присутствует такой параметр как *class_weight*, который и отвечает за соотношение стоимостей ошибок*



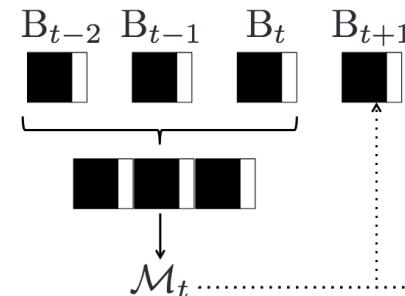
■ Учет особенностей задачи при разработке модели –
нестационарность процесса и латентность фрода

Нестационарность процесса (concept drift):

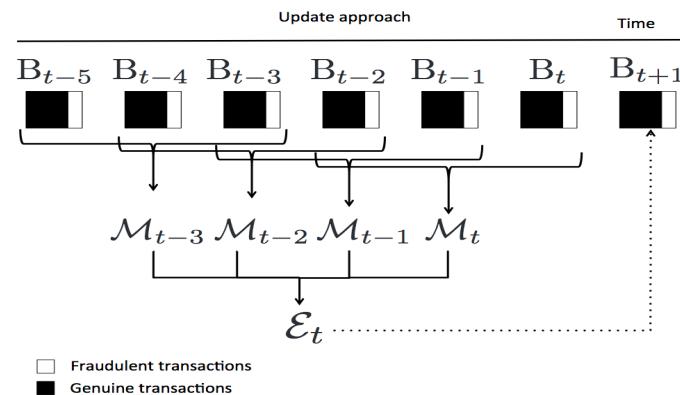
Модель будет работать с нестационарным во времени процессом, поэтому применять статические модели нельзя

Должен быть реализован процесс регулярного обновления («дообучения») модели:

- Самый простой – скользящее окно, когда модель обучается на предыдущих t -интервалах и предсказывает $t+1$ -интервал



- Взвешенный ансамбль обновляемых моделей

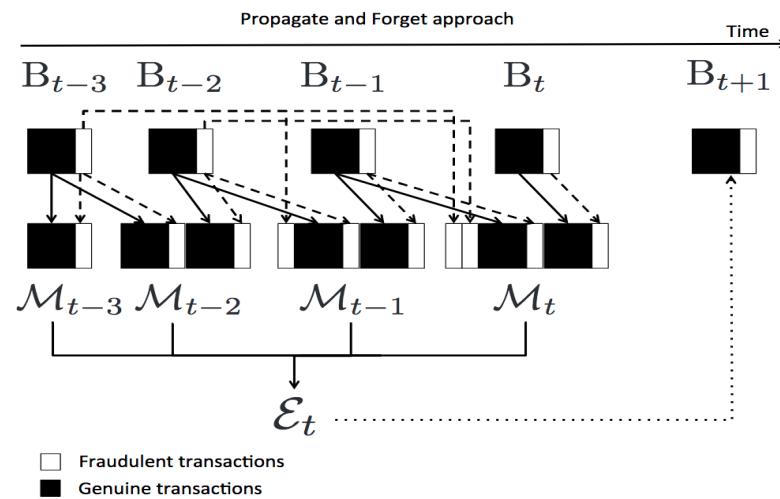


□ Fraudulent transactions
■ Genuine transactions

Учет особенностей задачи при разработке модели – нестационарность процесса и латентность фродо

Нестационарность процесса (concept drift):

➤ Ансамбль «забывающих» моделей



В результате возникают дополнительные гиперпараметры модели, которые нужно учесть в процессе оптимизации:

- частоты обновления (размер чанка);
- число чанков (глубины исторических данных) для обучения моделей;
- число моделей в ансамбле;

NOTE: базовые модели должны обучаться с учетом принципов отраженных на слайдах *imbalance and overlapping*



Учет особенностей задачи при разработке модели – нестационарность процесса и латентность фрода

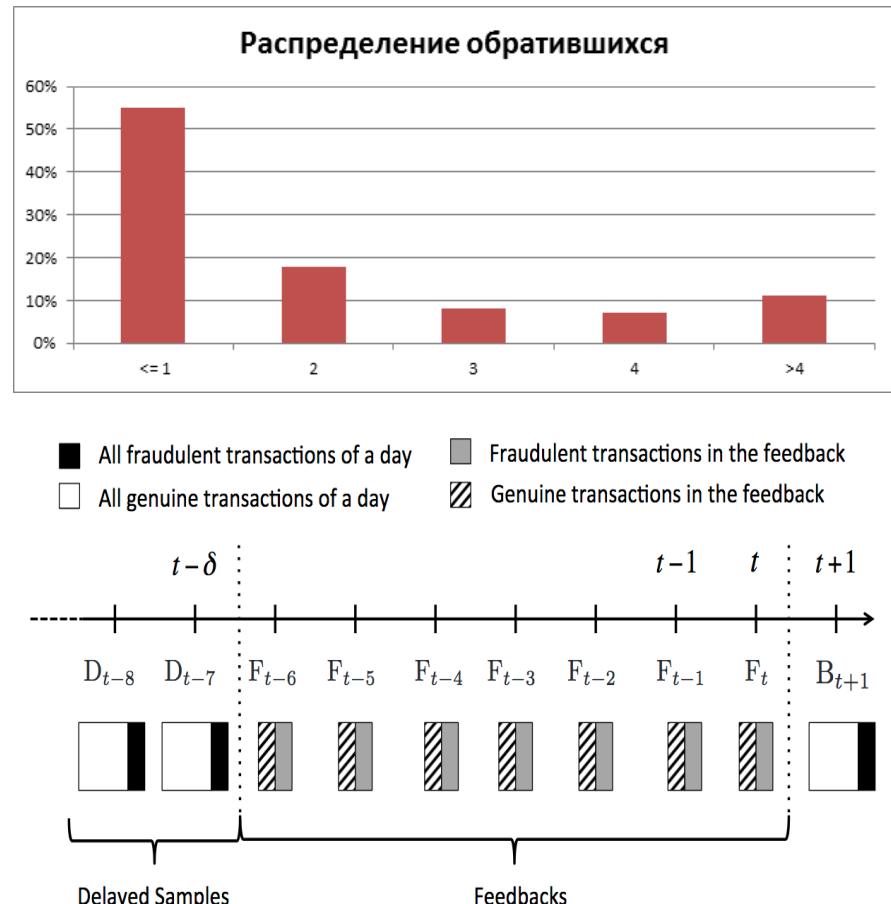


Период латентности фрода:

Но при работе модели в промышленном режиме у нас возникает еще нюанс – информация о фроде поступает с задержкой

Поэтому для получения более эффективных моделей, а также несмещенных оценок их эффективность в процесс обучения и валидации моделей необходимо:

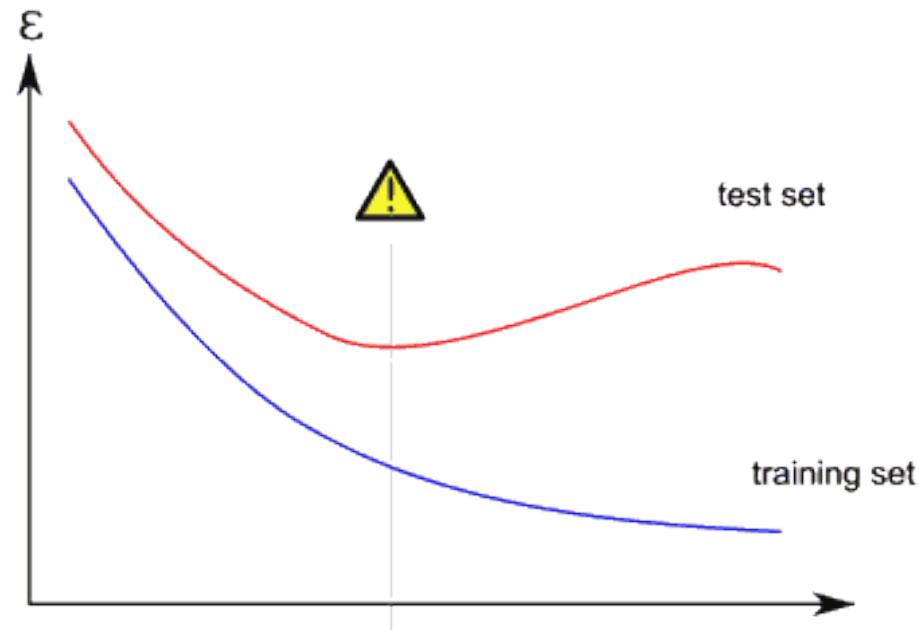
- Или не учитывать данные о пропущенном фроде, а учитывать только фрода из разборов алертов (менее предпочтительно)
- Или реализовать генератор «вскрытия» фрода на исторических данных с распределением вероятностей, соответствующих латентностям поступающего фрода (более предпочтительно)





Валидация моделей:

- *Оптимизация гиперпараметров;*
- *Сравнение нескольких моделей и отбор лучших;*
- *Получение несмещенной оценки эффективности модели (overfitting detection);*

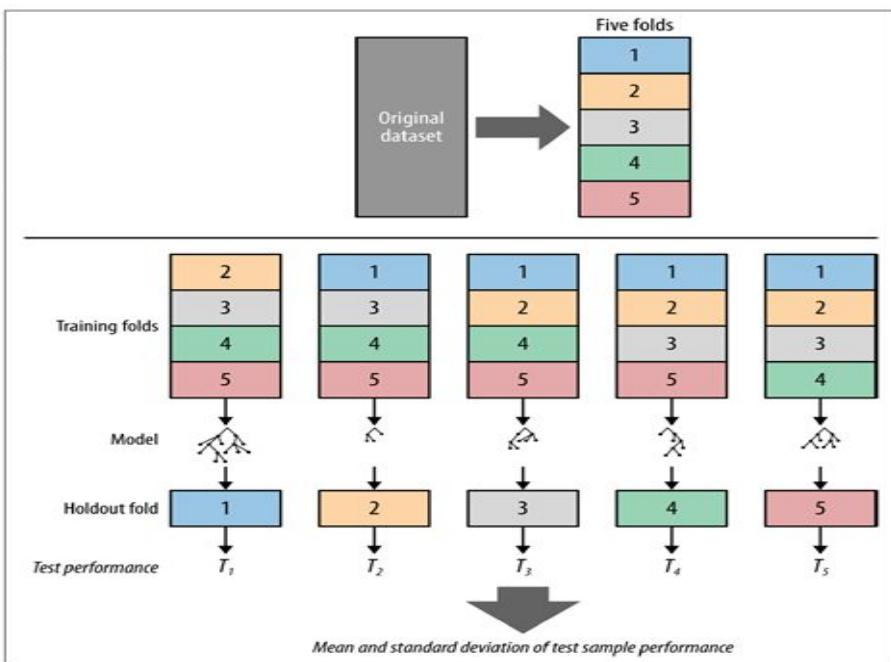




Нюансы валидации моделей

Валидация применяемые подходы:

- **Hold out set – не путать с test set**
 - **K-fold Cross-validation (а лучше stratified k-fold) – gold standard при тюнинге гиперпараметров и сравнении моделей**
 - **Leave p-out (и как частный случай leave one-out)**

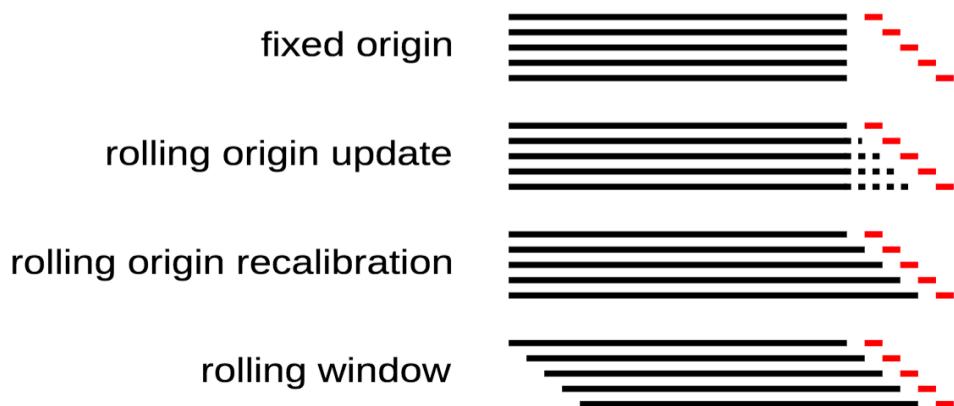




Однако и при кросс-валидации хватает подводных камней, о которых нужно помнить:

- Все еще можно переобучить модель – особенно при наличии выбросов/шумов в данных и выборе соответствующей метрики оценки (RMSE, например)
- В K-fold (еще называется *out of sample*) предполагается, что нет взаимосвязей между наблюдениями (они независимы) ... но в случае работы с временными рядами или наличию фич, зависящих от времени эта предпосылка не выполняется.

Необходимо использовать подходы *out-of-time cross-validation*



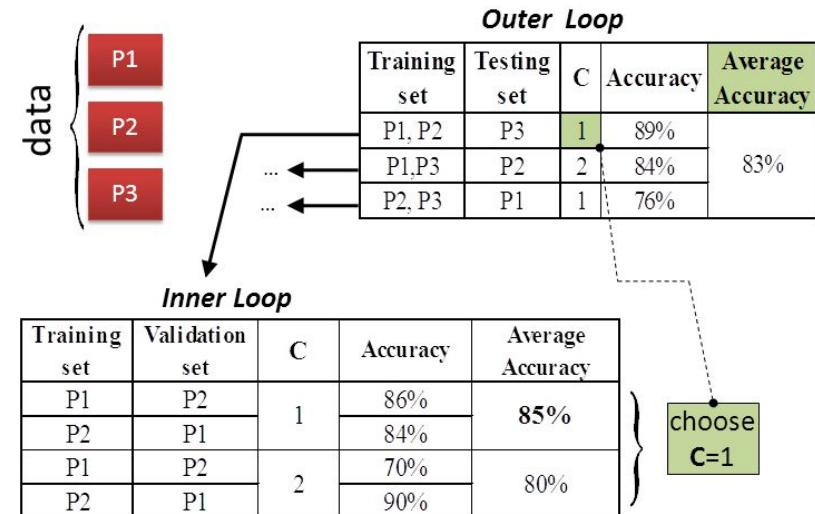
- Необходимо применять *cross-validation* в «правильных» местах, иначе могут появляться *feature leakage/contamination*.

Наиболее частые кейсы, в результате которых могут возникнуть такие эффекты:

- Временные ряды (см. пункт выше) – используйте модифицированные подходы, учитывающие временную природу данных
- Модификации данных (например, *scaling*) до начала кросс-валидации → выполняйте внутри фолдов;
- Тюнинг гиперпараметров/отбор фич - используйте вложенную (*nested*) *cross-validation* или дополнительный *test set*

Example of nested cross-validation

Consider that we use 3-fold cross-validation and we want to optimize parameter C that takes values “1” and “2”.



Нюансы валидации моделей (1/3)



1. VALIDATION STEP

The validation step is where you optimize the parameters for each algorithm you want to use. The two most common approaches are k-fold cross validation and a validation set — which approach you use will depend on your requirements and constraints!

VALIDATION SET

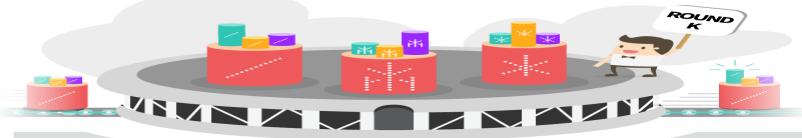
A validation set reduces the amount of data you can use, but it is simple and cheap



V S

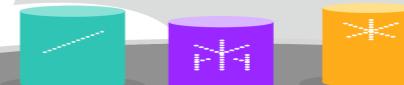
K-FOLD CROSS VALIDATION

K-folds will provide better results, but it is more expensive and time-consuming



2. TEST STEP

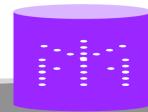
The test step is where you take the best version of each algorithm and apply it to your test set -- a set of data that has not been used in either the training or validating of the models. Your challenge here is to decide which metric to use for evaluation.



1. Linear	0.83
2. Tree	0.78
3. Other	0.71

3. YOUR BEST MODEL

Based on the metrics you chose, you will be able to evaluate one algorithm against another and see which performed best on your test set. Now you're ready to deploy the model on brand new data!



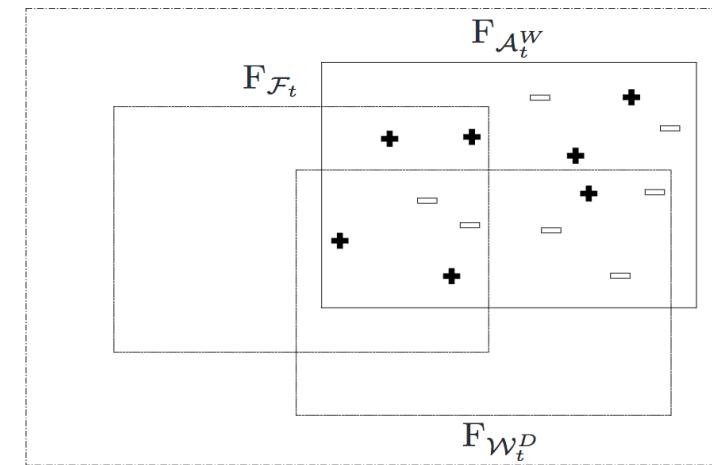
Нюансы сравнения разрабатываемых моделей с уже внедренным решением

Наличие уже работающей модели порождает дополнительные трудности при оценки эффективности новой модели

Идеальный вариант – возможность проведения А/В тестирования, когда определенный % транзакций направляется на новую модель, а результаты отрабатываются в том же бизнес-процессе

Но обычно такой вариант требует сложную ИТ-инфраструктуры и затратен по ресурсам. Альтернативой может выступать сравнение результатов работы моделей на пересечении их сработок

A/B Testing





Summary по разработке модели



Подводя итог всему описанному ранее разрабатываемая модель:

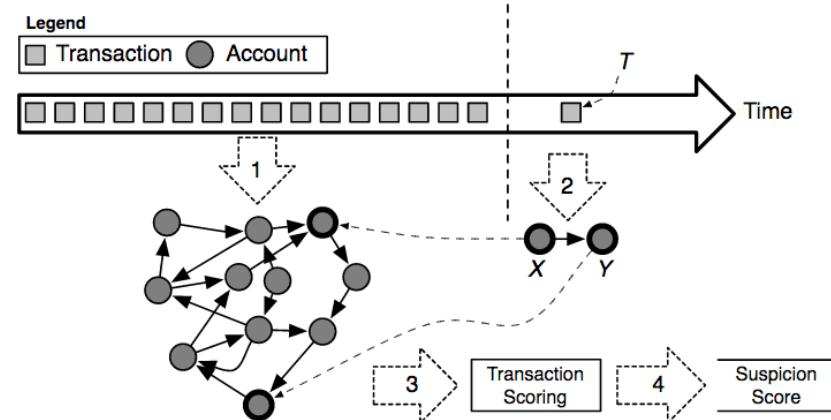
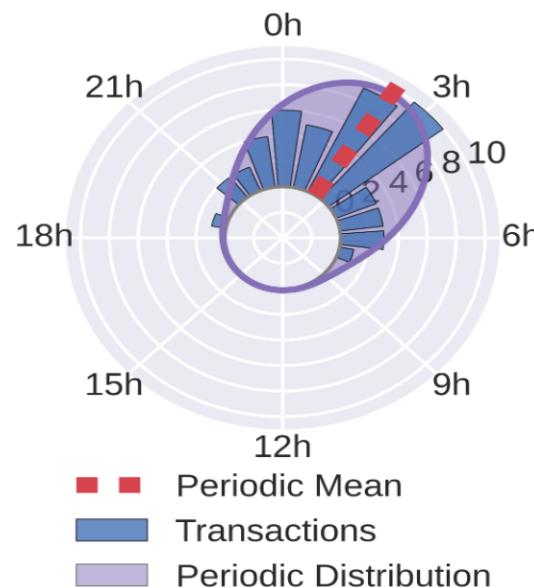
- *Будет оцениваться по partial Precision-Recall AUC в диапазоне соответствующем общему количеству сработок в ~1 000 кейсов сутки;*
- *Должна быть адаптивной;*
- *Базовые алгоритмы модели должны включать корректирующие методы (для imbalance and overlapping данных);*
- *Процедура кросс-валидации модели должна учитывать распределение данных во времени и корректно выполнять тюнинг параметров и отбор моделей с учетом кросс-валидации;*
- *Должна учитывать латентность появления разметки в процессе валидации и дообучения (для упрощения задачи – исключаем);*
- *Признаки, используемые моделью должны хорошо описывать поведение клиента*



- Одних только данных в самой транзакции (сумма, ip-адрес, fingerprint и пр.) недостаточно для построения эффективной модели. Необходимо к имеющимся создать признаки, описывающие (профилирующие) поведение клиента. В литературе, посвященной борьбе с фрэдом часто это называют *RFM - Recency, Frequency, Monetary Value*.
- Обычно это достигается за счет использования широкого набора различного рода агрегаций, математических функций: перцентили, средние и отклонения, скользящие окна и многое другое.
- Примеры возможных признаков:
 - среднее расходов клиента в разрезе типов операций со скользящим недельным окном за последние три месяца, его среднеквадратичное отклонение;
 - среднее/перцентили расходов клиента в разрезе типов операций с дневным скользящим окном за последний месяц;
 - число предыдущих транзакций по данному мерчанту всего/за последние 30 дней;
 - сумма транзакций за последние 24 часа;
 - наличие жалоб на поставщика услуг за последний месяц и т. д.



- Хорошими кандидатами на включение являются:
- признаки учитывающие периодичность поведения клиента. Например, распределение входов/операций в систему по 24 часам (преобразования Фурье, von Mises distribution и взвеси распределений);
 - признаки построенные на графах переводов клиентов (наличие связей, силы связей, характеристики вершины, окружение вершины и пр.);
 - кластеризация и сравнения поведения клиента с характеристиками его кластера;



Feature Engineering – categorical features



- Самые популярные методы 1-hot encoding и numeric-encoding. Но последний применим только для нелинейных моделей и оба для случаев с low cardinality.

Categorical Feature	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10
Louise =>	1	0	0	0	0	0	0	0	0	0
Gabriel =>	0	1	0	0	0	0	0	0	0	0
Emma =>	0	0	1	0	0	0	0	0	0	0
Adam =>	0	0	0	1	0	0	0	0	0	0
Alice =>	0	0	0	0	1	0	0	0	0	0
Raphael =>	0	0	0	0	0	1	0	0	0	0
Chloe =>	0	0	0	0	0	0	1	0	0	0
Louis =>	0	0	0	0	0	0	0	1	0	0
Jeanne =>	0	0	0	0	0	0	0	0	1	0
Arthur =>	0	0	0	0	0	0	0	0	0	1

Categorical Feature	Numeric
Louise =>	1
Gabriel =>	2
Emma =>	3
Adam =>	4
Alice =>	5
Raphael =>	6
Chloe =>	7
Louis =>	8
Jeanne =>	9
Arthur =>	10

- Развитие label-encoding → binary encoding и hashing trick

Categorical Feature	=	x1	x2	x4	x8
Louise =>	1	1	0	0	0
Gabriel =>	2	0	1	0	0
Emma =>	3	1	1	0	0
Adam =>	4	0	0	1	0
Alice =>	5	1	0	1	0
Raphael =>	6	0	1	1	0
Chloe =>	7	1	1	1	0
Louis =>	8	0	0	0	1
Jeanne =>	9	1	0	0	1
Arthur =>	10	0	1	0	1

➤ High cardinality:

COUNTING – хорошо улавливается нелинейными моделями

Supervised ratio - AVERAGED + VARIANCE (но аккуратно с overfitting)

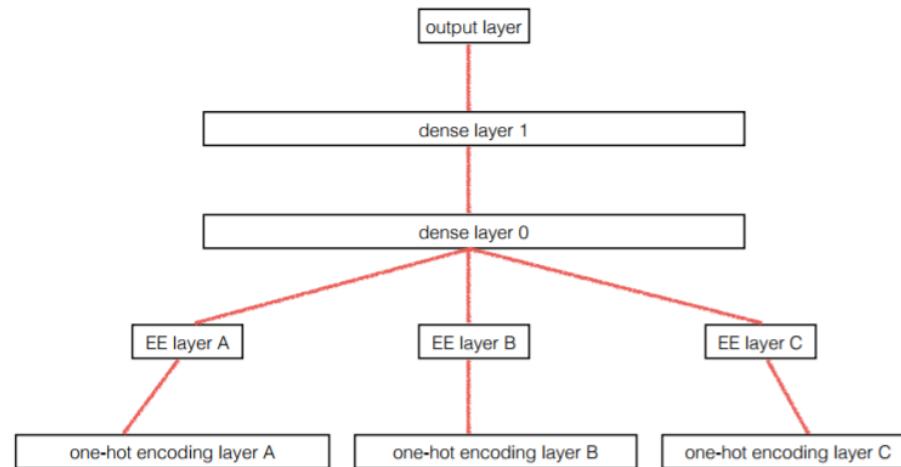
$$SR_i = \frac{P_i}{N_i + P_i}$$

Weight of evidence

$$WoE = \left[\ln \left(\frac{\text{Relative Frequency of Goods}}{\text{Relative Frequency of Bads}} \right) \right] * 100$$

Cat2Vec

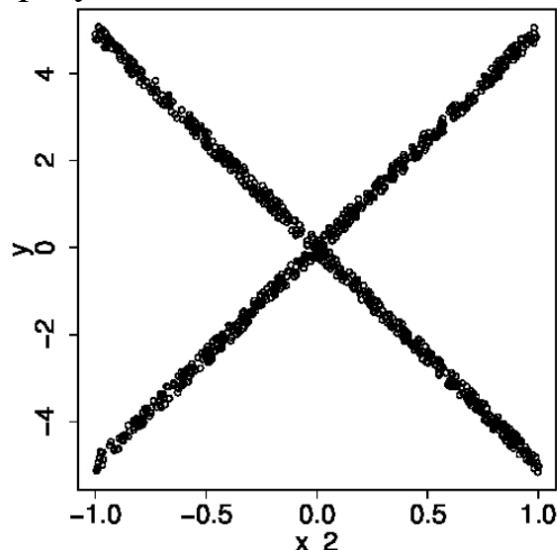
Embedding



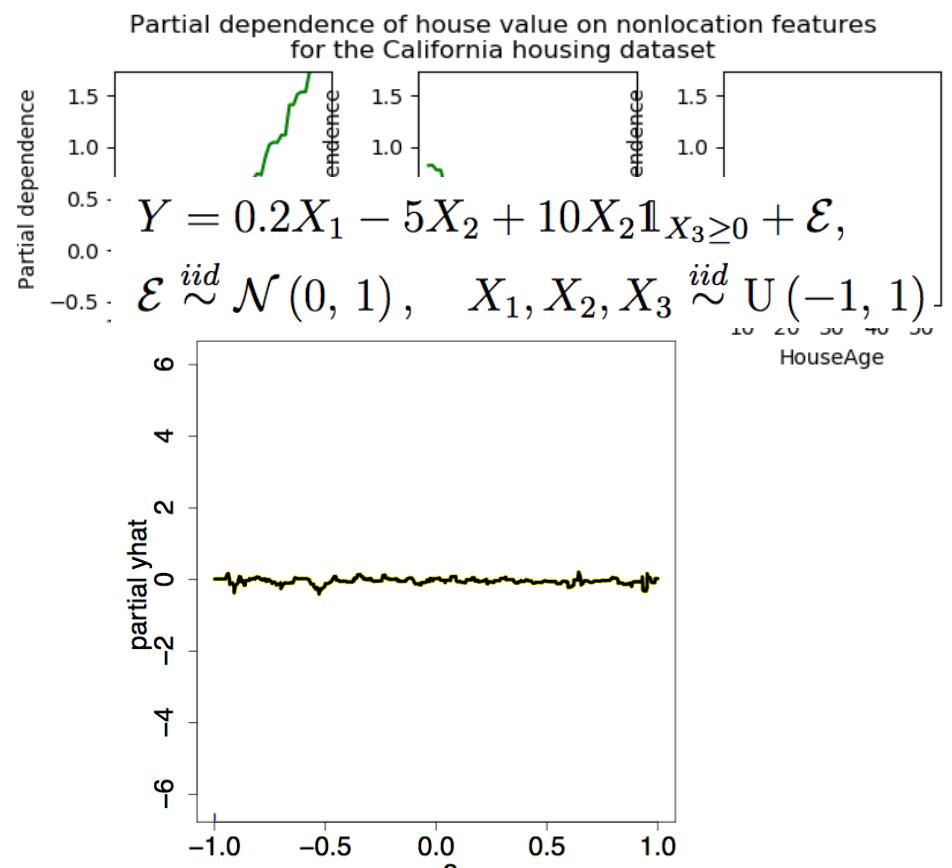
Partial Dependence plot

$$\hat{f}_S = \frac{1}{N} \sum_{i=1}^N \hat{f}(\mathbf{x}_S, \mathbf{x}_{Ci})$$

Ограничение – только для если этот набор фичей слабо зависят от оставшихся фичей. В противном случае – возможен неправильный результат



(a) Scatterplot of Y versus X_2



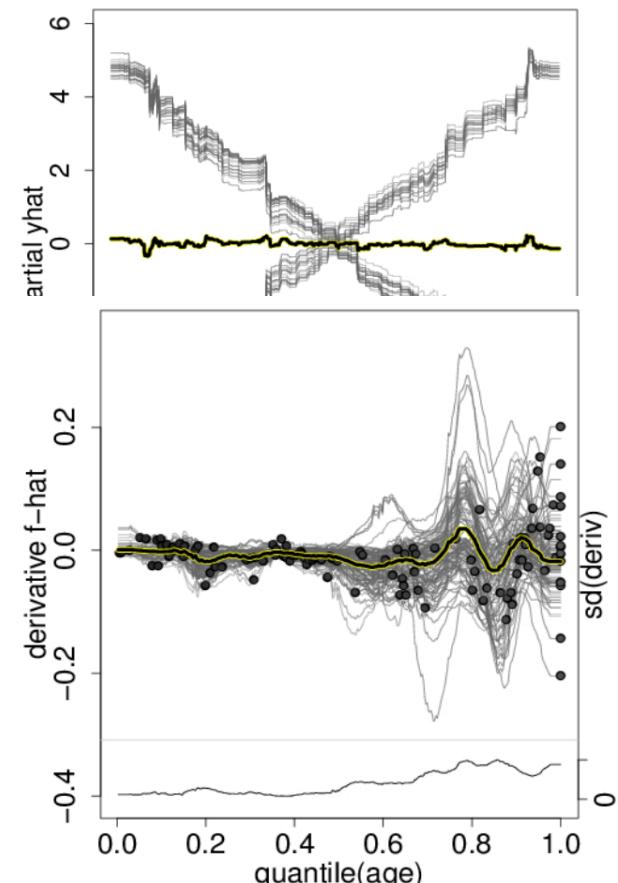
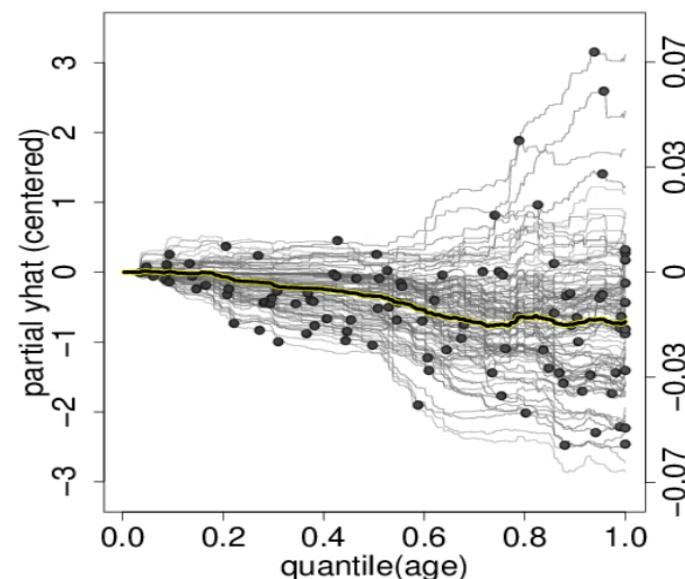
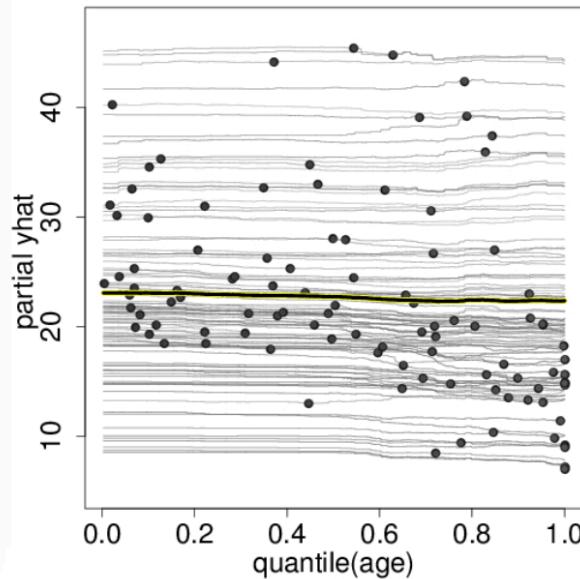
(b) PDP

Продвинутые техники валидации - интерпретируемость модели и результатов ее предсказаний



Individual Conditional Expectation (ICE) – PDP, но без агрегации

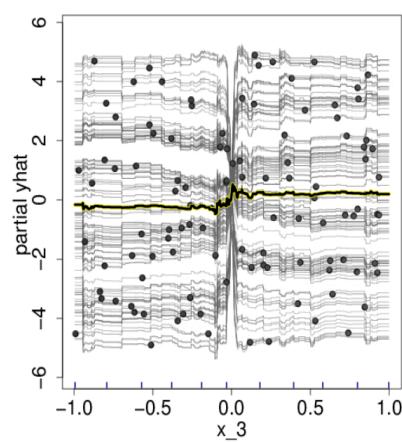
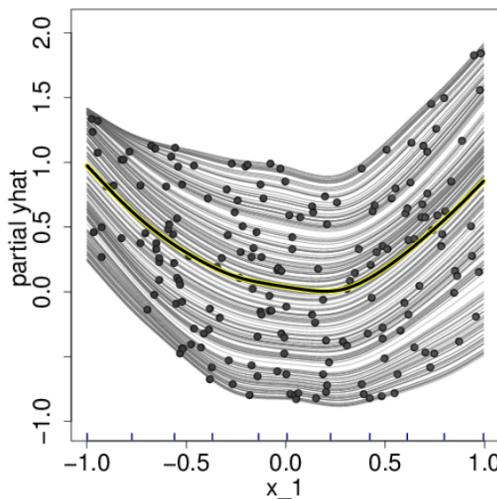
Centred ICE plot $\hat{f}_{\text{cent}}^{(i)} = \hat{f}^{(i)} - \mathbf{1}\hat{f}(x^*, \mathbf{x}_{Ci})$, и Derivative ICE



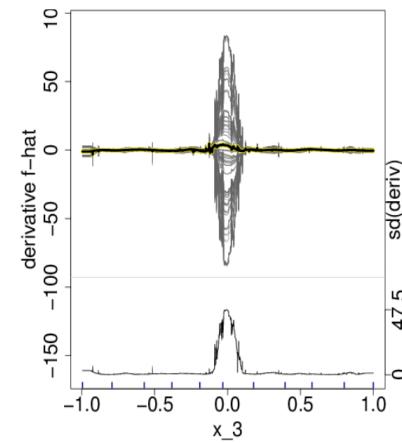
- Интерпретируемость black-box моделей – понимаем как предсказание меняется при изменении тех или иных параметров и насколько они зависят от оставшихся признаков (валидация моделей);
- Определение для отдельно взятого случае влияние на предсказание изменений по каждому признаку;
- Определение как модель поведет себя в регионах, где обучающие данные пока отсутствуют;

Как «читать» ICE plots:

- Если все линии лежат поверх друг друга (совпадает с PDP), то это означает что все прочие признак никак не влияют на зависимость $f(Xs)$;
- Если все линии имеют одинаковую форму, но разный уровень, то f – аддитивная от Xs и Xc
- Если же мы видим разнородность в определенных регионах форм линий относительно друг друга, то в этом есть взаимное влияние признаков на предсказание



(a) ICE (10 curves)



(b) d-ICE

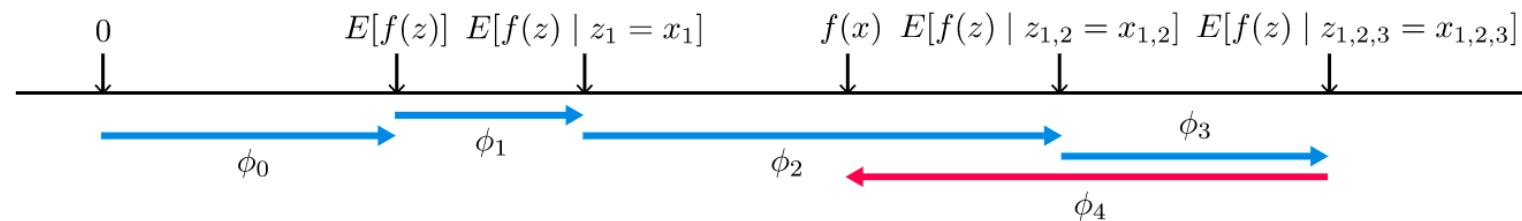


Продвинутые техники валидации - интерпретируемость модели и результатов ее предсказаний



SHAP (SHapley Additive eXplanations) values

$$g(z') = \phi_0 + \sum_{i=1}^M \phi_i z'_i, \quad \phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|!(M - |z'| - 1)!}{M!} [f_x(z') - f_x(z' \setminus i)] \quad f_x(z') = f(h_x(z')) = E[f(z) \mid z_S],$$



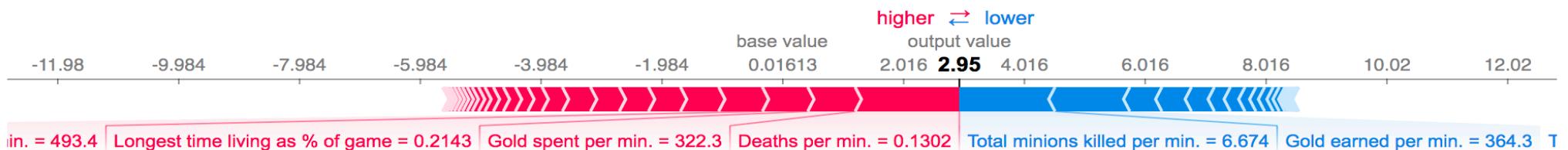
НО - они native встроены в xgboost и lightGBM! И есть ядро для оценки произвольных моделей – <https://github.com/slundberg/shap>

Продвинутые техники валидации - интерпретируемость модели и результатов ее предсказаний

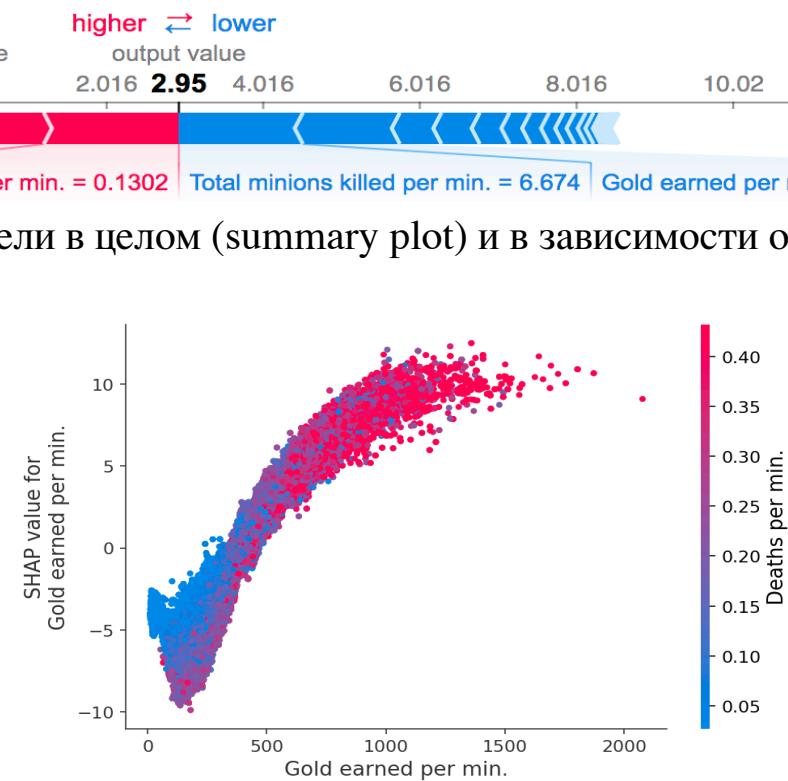
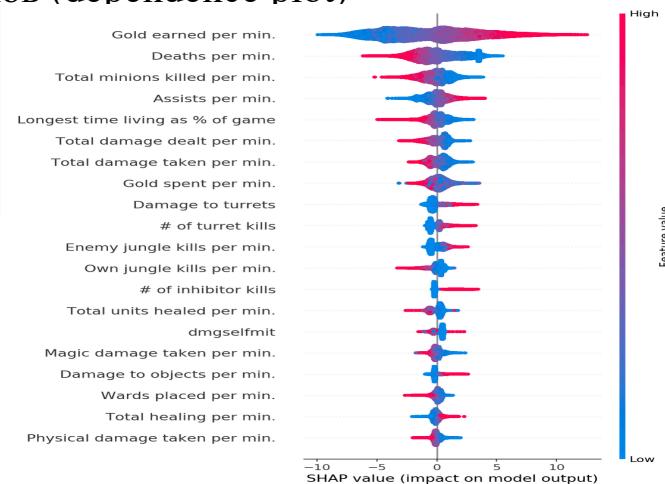


Что позволяют сделать SHAP values:

- Получение понимания почему по конкретному примеру какие признаки оказались наиболее значимыми в принятии решения



- Оценить как значение признака влияет на предсказание модели в целом (summary plot) и в зависимости от других признаков (dependence plot)



СПАСИБО
ЗА ВНИМАНИЕ



SCS

SBERBANK
CYBER
SECURITY

