Поиск мошеннических операций в платежных транзакциях интернет-аукциона

Колеганов Николай

1. ПОСТАНОВКА ЗАДАЧИ

КАКАЯ ПРОБЛЕМА

В нашем мире постоянно растущей конкуренции на всех рынках качество и безопасность обслуживания, а значит и лояльность клиентов ставится на первое место среди приоритетов бизнеса.
В сфере электронных платежей для этого **требуется быстрый** и **качественный анализ транзакций на фрод**.

Необходимо с точностью не ниже 95% **классифицировать транзакции** на **легитимные** и **фрод**.
- При этом **классы** сильно **несбалансированные (2% фрод-транзакций)**.
- Бизнесу важно, чтобы модель **минимум блокировала легитимные транзакции** и **максимум** блокировала **фрод**.
- И **максимально быстро - это транзакции!**

Результат работы модели: **по вероятности класса «фрод»** к каждой транзакции нужно **применить одно из действий**:
- **PASS** – транзакция не является подозрительной, пропускаем
- **ALERT_AGENT** – о транзакции следует сообщить наблюдателям
- **LOCK_USER and ALERT_AGENT** – транзакцию следует заблокировать и сообщить наблюдателям для анализа
- **LOCK_USER** – транзакцию строго блокируем, она является мошеннической

КАК РЕШАЛ ЗАДАЧУ

Шаг 0 **Первичный анализ** задачи и **выдвижение гипотез** на основе EDA

Шаг 1 **Feature Engineering**, **балансировка классов** для обучения

Шаг 2 **Создание baseline модели** – логистической регрессии

Шаг 3 **Создание более сложных моделей**, анализ прироста качества, подбор гиперпараметров

Шаг 4 **Понижение размерности**

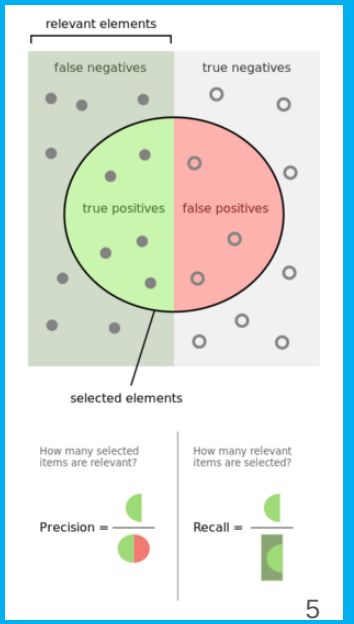
ЦЕЛЕВЫЕ МЕТРИКИ

- Метрика для **сравнения моделей и подбора гипер-параметров** (после балансировки классов) – **ROC–AUC**

- Метрики для **трансляции результатов бизнес-заказчикам – Precision** (доля срабатывания модели от всего потока транзакций) и **Recall** (доля пойманных мошеннических[1] транзакций от всех)

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$



1. Мошенническими транзакциями будем считать оценки классификатора выше 50%

2. АНАЛИЗ

КАКИЕ ЕСТЬ АНАЛОГИ

Аналог-1

Detecting Credit Card Fraud Using Machine Learning[1]

Недостатки:

- Отсутствует промышленное применение

Аналог-2

SAFE: A Neural Survival Analysis Model for Fraud Early Detection[2].

Недостаток:

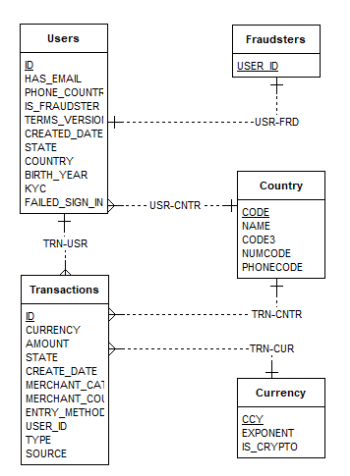
- Черный ящик для бизнеса

EDA

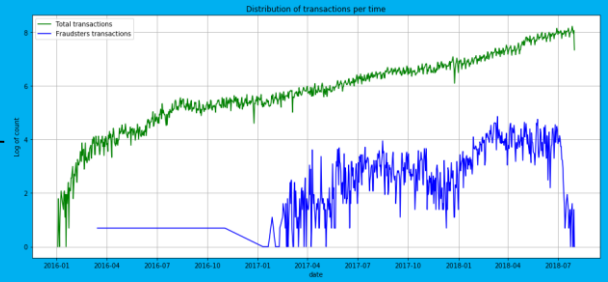
Исходные данные:

1. **Транзакции** пользователей (688 651, 11)
2. Справочник **пользователей** (9 944, 11)
3. Справочник стран (226, 5)
4. Справочник валют (184, 3)
5. Список злоумышленников (298, 1)

ERD



Log(count) всех и фрод транзакций



Созданы признаки:

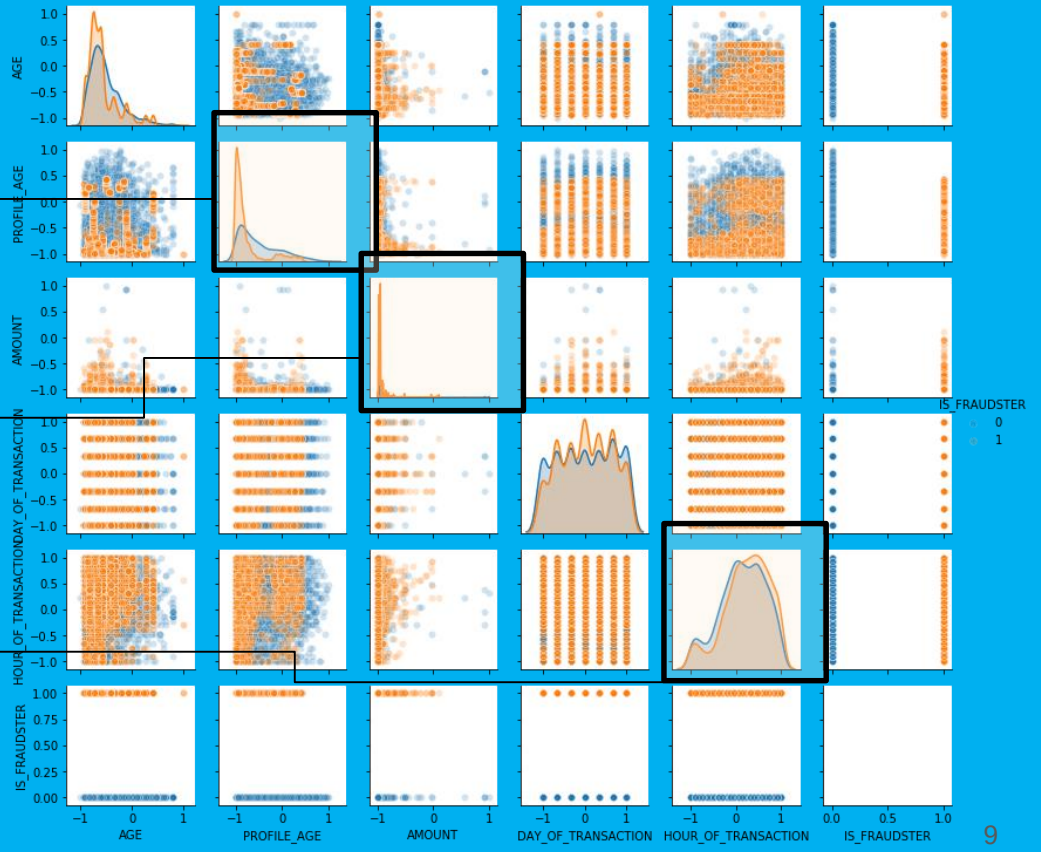
1. **PROFILE_AGE** – возраст профиля на момент транзакции
2. **HOMELAND** – признак проведения транзакции в стране из профиля пользователя
3. **ENTRY_METHOD_*** - OneHotEncoding категориального признака ENTRY_METHOD
4. **TYPE_*** - OneHotEncoding категориального признака TYPE
5. **HOUR_OF_TRANSACTION** – час из времени проведения транзакции
6. **HOMELAND_PHONE** – признак соответствия страны профиля пользователя стране указанного в профиле телефона

EDA

Средний возраст профиля фрод-транзакции отличается от среднего возраста обычного пользователя

Средний объем фрод-транзакций отличается от среднего объема транзакции обычного пользователя

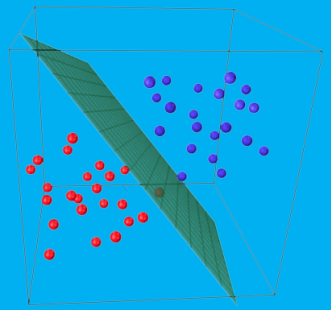
Средний час фрод-транзакции отличается от среднего часа транзакции обычного пользователя



АЛГОРИТМЫ И ТЕХНИКИ

– Логистическая регрессия

sklearn.linear_model.LogisticRegression



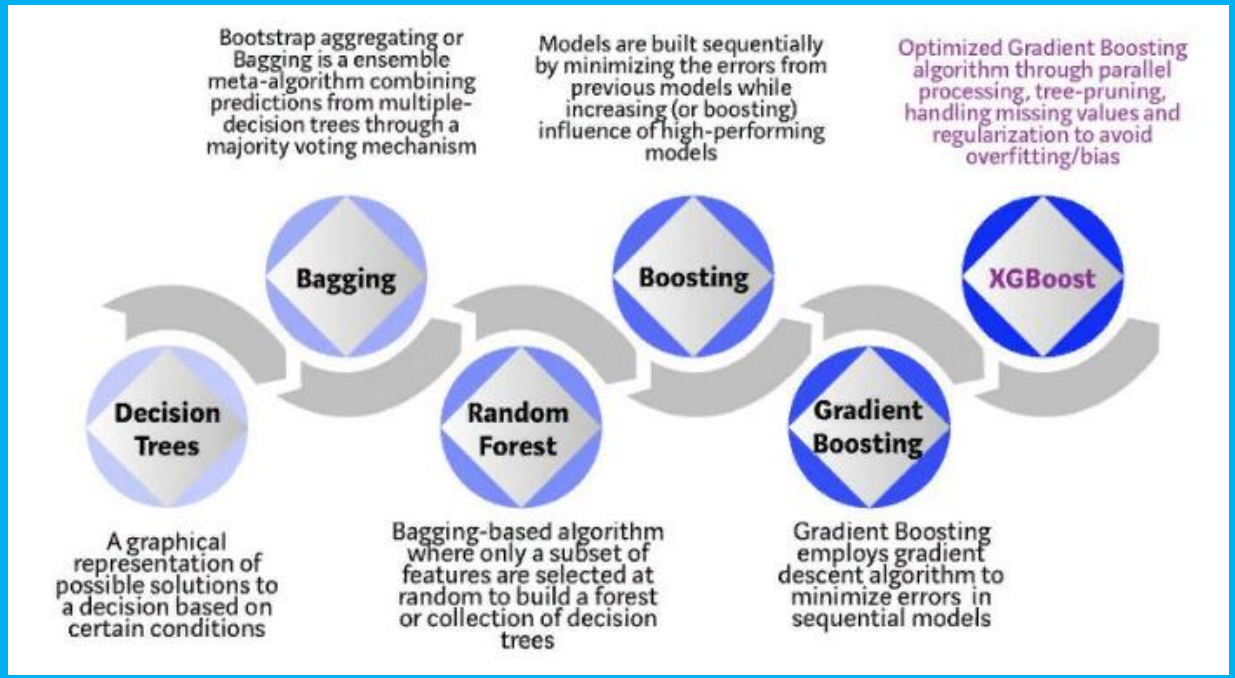
– Случайный лес

sklearn.ensemble.RandomForestClassifier

АЛГОРИТМЫ И ТЕХНИКИ

XGBoost

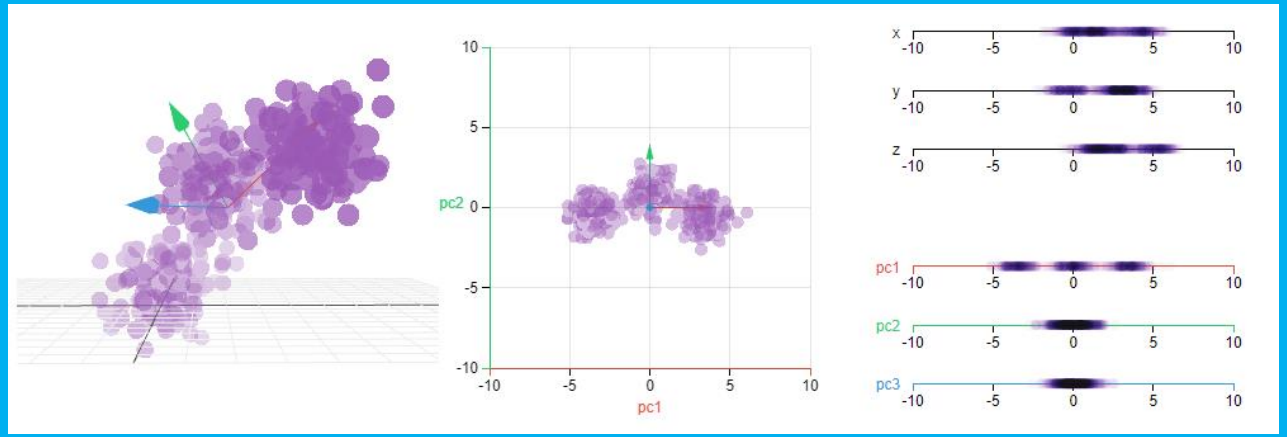
xgboost.XGBClassifier



АЛГОРИТМЫ И ТЕХНИКИ

Метод снижения размерности с помощью поиска главных компонент

sklearn.decomposition.PCA



3. МЕТОДИКА РЕШЕНИЯ

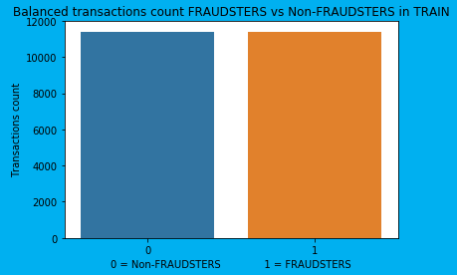
РЕАЛИЗАЦИЯ

1. Будем **обучаться на 70% транзакций до даты 2018-05-24**, на остальных 30% будем тестировать нашу модель
2. **Случайным образом отбираем легитимные транзакции** в обучающую выборку в кол-ве мошеннических транзакций
3. Строим **baseline модель – LogisticRegression**
4. Тестируем другие модели
5. Тестируем лучшую модель с понижением размерности

- В **тренировочном** датасете уникальных пользователей - **267 мошенников / 5873 легитимных**
- В **тестовом** датасете уникальных пользователей - **95 мошенников / 5153 легитимных**

ПРЕДОБРАБОТКА

- Сбалансированные классы необходимы для корректной оценки качества модели



	Логистическая регрессия
Несбалансированная выборка	ROC AUC = 0.66
Сбалансированная выборка	ROC AUC = 0.87

- Выбросы удалить нельзя, т.к. в них более 26% фрод-транзакций

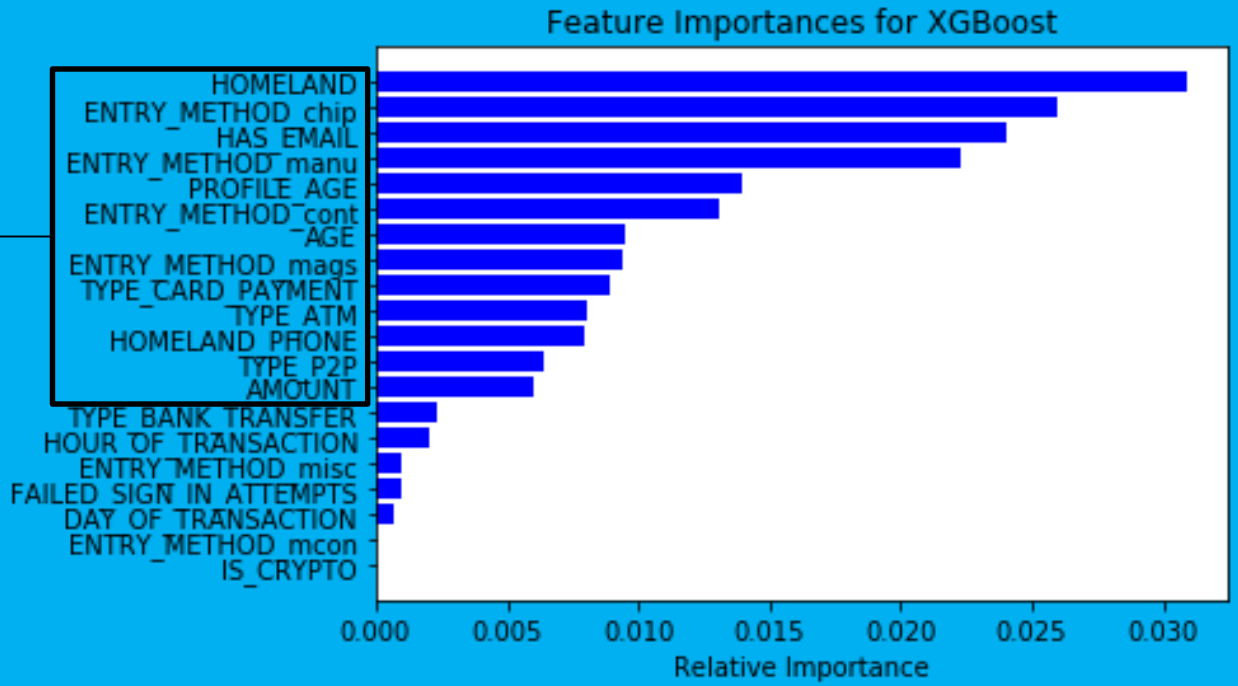
ИТОГОВАЯ МОДЕЛЬ

XGBoostClassifier на основе уменьшенного количества признаков с помощью **PCA** с подобранными гиперпараметрами на основе кросс-валидации Kfold кратности 3:

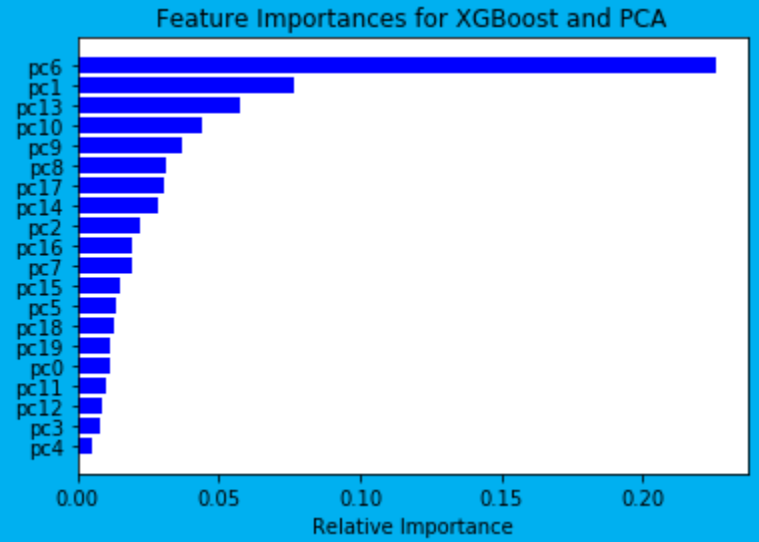
- Количество признаков снижено с 196 до 47;
- Максимальная глубина деревьев: 7
- Количество деревьев: 200

ВАЖНОСТЬ ПЕРЕМЕННЫХ

Общие с RF



PCA



Метрика	XGB	XGB + PCA
Precision	0.61	0.96
Recall	0.88	0.96
ROC-AUC	0.88	0.99

Расшифровка PC6

```
1  pcs.sort_values()
```

```
CURRENCY_EUR                -0.470389
TERMS_VERSION_2018-09-20    -0.259930
TERMS_VERSION_2018-05-25    -0.191110
STATE_COMPLETED             -0.166100
KYC_PASSED                  -0.135404
                                ...
KYC_PENDING                  0.119044
ENTRY_METHOD_manu            0.166564
HOMELAND                     0.193090
TERMS_VERSION_2018-03-20     0.437195
CURRENCY_GBP                 0.559896
Name: pc6, Length: 196, dtype: float64
```

ДЕЙСТВИЯ НАД ТРАНЗАКЦИЯМИ

1. Модель возвращает для каждой транзакции вероятность мошенничества
2. Вероятность можно разделить на группы действий требуемые с точки зрения бизнеса процесса

Вероятность	Действие	Вес
> 50%	ALERT	1
> 75%	LOCK and ALERT	2
> 90%	LOCK	3

По каждому пользователю выбираем наиболее весомое действие

```python
def check_alert(y_predicted):
    # Return the most important (heaviest by weight) flag as a result
    '''
    Rules:
        If percent is more than first level (50% for example) we need ALERT (weight=1) because it's suspicious transaction.
        If percent is more than second level (75% for example) we need LOCK and ALERT (weight=2) because it's very suspicious transaction and it's better to lock user and send alert signal to work with this user.
        If percent is more than max level (90% for example) we need LOCK (weight=3) because it's fraudster.
    '''

    # dictionary of alerts
    dict_of_alerts = {0: ['PASS'],
                      1: ['ALERT_AGENT'],
                      2: ['LOCK_USER', 'ALERT_AGENT'],
                      3: ['LOCK_USER']
                     }
    # for each prediction in y_prediction check the rules, get the max weight and apply dictionary to get the alert
    return dict_of_alerts[max([{       y >= .9: 3,
                                .75 <= y <  .9: 2,
                                .5  <= y < .75: 1,
                                       y <  .5: 0}[True] for y in y_predicted])]
```

4. РЕЗУЛЬТАТЫ

ОЦЕНКА И ВАЛИДАЦИЯ

Метрика	LR	RF	XGB
Precision	0.58	0.94	0.96
Recall	0.87	0.94	0.96
ROC-AUC	0.87	0.98	0.99

Валидация функции определения действия
Проверка на «хороших» пользователях из отложенной выборки – из вероятности отнесения к классу фрода рождается необходимое над транзакцией действие:

```
1  test_user_df['PATROL_SOLUTION'].value_counts()
```

```
[PASS]                          73
[LOCK_USER]                      6
[ALERT_AGENT]                    3
[LOCK_USER, ALERT_AGENT]         2
Name: PATROL_SOLUTION, dtype: int64
```

ВНЕДРЕНИЕ

Модель анализирует каждую транзакцию

```
X = df[df['TRN_ID']=='961f9451-2d7d-4c62-8593-bf44d15d38b0'].drop(parameters.id_features + [parameters.target_feature],axis=1)
y_pred = rf.predict_proba(X)[:,1][0]
print(f"Probability of class IS_FRAUDSTER=1 = {y_pred}")
```

```
Probability of class IS_FRAUDSTER=1 = 1.0
```

And model predicted this transaction is by fraudster!

И определяет необходимое действие

над пользователем по его транзакциям

```
def patrol(user_id, rf=None):
    if rf is None:
        # load our pretrained model
        from sklearn.externals import joblib
        #rf = joblib.load(parameters.model_pkl)
        rf = pickle.load(open(parameters.model_pkl,'rb'))
    # load data for user_id
    X = get_user_data(user_id = user_id, asserting=False)
    # get prediction
    if len(X) > 0:
        # if we have transactions for this user => act on them
        y_pred = rf.predict_proba(X)[:,1]
        return check_alert(y_pred)
    # if we have no transactions for this user => pass him
    return ['PASS']
```

```
patrol(user_id='fb23710b-609a-49bf-8a9a-be49c59ce6de')
```

```
['LOCK_USER']
```

```
# load our pretrained model
#model = joblib.load(parameters.model_pkl)
model = pickle.load(open(parameters.model_pkl,'rb'))

# check FRAUDSTER-users from users' dataset
d = {u: patrol(user_id=u, rf=model) for u in user_df[user_df['IS_FRAUDSTER']==True]['ID'].values}
print('Patrol-function actions on FRAUDSTERS:')
pd.DataFrame(list(d.items()), columns=['USER_ID','ACTION'])['ACTION'].value_counts()
```

```
Patrol-function actions on FRAUDSTERS:

[LOCK_USER]                294
[PASS]                       2
[LOCK_USER, ALERT_AGENT]     1
[ALERT_AGENT]                1
Name: ACTION, dtype: int64
```

5. ЗАКЛЮЧЕНИЕ

ВЫВОДЫ

1. Лучшая модель – **XGBoost**
2. Самые важные признаки:
   - HOMELAND – признак транзакции из страны регистрации профиля
   - ENTRY_METHOD – способ оплаты
   - HAS_EMAIL – указан ли email в профиле
   - PROFILE_AGE - Возраст профиля
   - AGE - возраст клиента
3. Модель классифицирует транзакции в условиях, приближенных к реальным, и готова к замерам производительности и расчетам необходимой производительности виртуальной машины для ее хостинга

КУДА ДАЛЬШЕ

- ✓ Требуется обогащение клиентского профиля для повышения качества модели
- ✓ Замер скорости обработки транзакций пользователей с учетом расчёта дополнительных параметров транзакций
- ✓ Если скорость удовлетворяет разрешенным пределам, то интеграция в production-систему

Спасибо за внимание!