АБ-тест для классификатора фрода

Данные

Имеются данные про 35,000 продавцов, из которых примерно 8% помечены как мошенники.

Исключим из рассмотрения продавцов с неполными/противоречивыми данными:

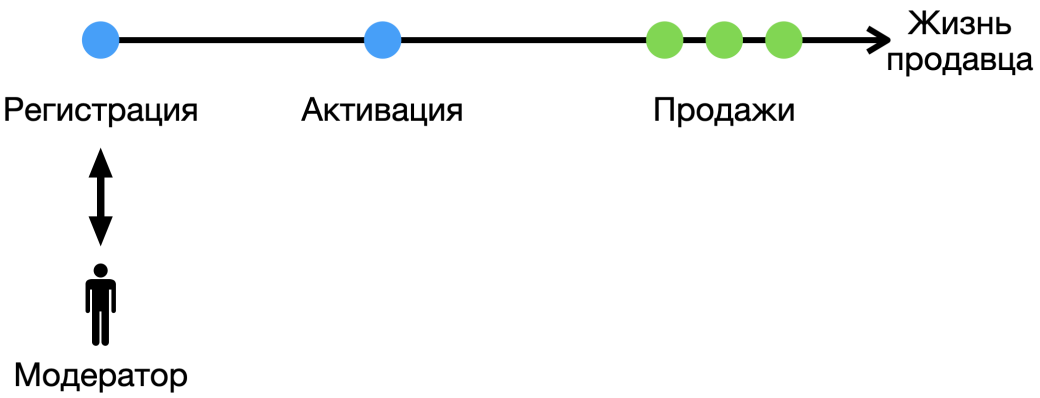
1. без даты регистрации – 175 записей;
2. регистрация была после активации - 19 записей;
3. без индикатора мошенничества - 700 записей.

В результате получим 34,108 записей.

```
import pandas as pd

df = pd.read_csv('dataset.csv', sep=';')
df['registration_date'] = pd.to_datetime(
    df['registration_date'], format="%d.%m.%Y"
)
df['activation_date'] = pd.to_datetime(
    df['activation_date'], format="%d.%m.%Y"
)

df = df.query('registration_date > "1970-01-01"')
df = df.query('activation_date.isna() or (registration_date <= activation_date)')
df = df.dropna(subset=['ind_frod'])

df.shape
# (34108, 5)
```

В почищенном датасете все регистрации совершены в 2023 году между 2 января и 31 декабря. В неделю совершалось примерно 600-700 регистраций и обнаруживалось 50-60 мошенников.

Картина мира



Жизнь каждого продавца на маркетплейсе начинается с регистрации. После регистрации должна произойти активация. Только после активации станут возможны первые продажи.

Будем считать, что существует процедура проверки, согласно которой продавцу присваивается индикатор мошенничества `ind_frod`. Присвоение статуса мошенника может произойти на стадии регистрации, активации или после очередной продажи. До внедрения ML-модели на этапе регистрации проверка осуществляется живым модератором и/или автоматическими проверками.

В предложенных данных 2% продавцов не имеют индикатора мошенничества. Можно предположить, например, что это продавцы, которые не показались подозрительными на этапах регистрации и активации, но почему-то так и не продали ни одного товара, из-за чего их нельзя уверенно считать добросовестными продавцами. Будем считать, что вынесение вердикта о мошенничестве происходит для подавляющего большинства продавцов достаточно быстро, например, за первые 14 дней. Назовем это время временем проверки T_{check}.

В любой системе обнаружения мошенников возможны ложноположительные срабатывания, то есть ситуации, когда добросовестный продавец ошибочно называется мошенником. В этом случае продавец будет оспаривать решение модерации. Пересмотр его статуса займет определенное время у команды модерации. Будем считать, что время пересмотра спорных случаев также входит в T_{check}.

Дизайн АБ-теста

Гипотеза

Классификатор мошенников, разработанный ML-командой, способен находить больше мошенников на этапе регистрации, чем имеющаяся система модерации.

Что делаем в продукте

В контрольной группе на этапе регистрации будет работать текущая система модерации, в тестовой – предлагаемый ML-классификатор.

На каких продавцах проводим тест

Рассматриваем всех продавцов, которые зарегистрировались в период эксперимента.

Для рассчетов будем брать индикатор мошенничества на момент, когда прошло время T_{check} после окончания эксперимента.

Метрики

В качестве целевой метрики будем использовать полноту (recall):

$$

$$\text{recall} = \frac{N_{\text{detected\_fraud}}}{N_{\text{fraud}}}$$

$N_{\text{detected\_fraud}}$ – число мошенников, обнаруженных на этапе регистрации.

N_{fraud} – число всех мошенников.

В качестве контрольной метрики используем FPR (false positive rate):

$$FPR = \frac{FP}{N_{\text{normal}}}$$

FP – число добросовестных продавцов, ошибочно принятых за мошенников.

N_{normal} – число всех добросовестных продавцов.

FPR не должен быть сильно большим, чтобы не перегружать команду модерации пересмотром случаев несправедливо заблокированных продавцов.

Пусть команда модерации может обработать в неделю X спорных случаев, а пиковая нагрузка может быть N регистраций в неделю. Тогда мы можем оценить максимальный допустимый FPR:

$$FPR_{max} = \frac{X}{0.92N}$$

Ожидаемый эффект и размер выборки

Используя датасет, на котором обучалась ML-модель, мы можем получить оценку полноты теста recall_T(по кросс-валидации или на отложенной выборке).

Исходя из наших данных, мы также можем оценить полноту текущей системы модерации. Для этого посмотрим долю продавцов с `ind_frod == 1`, которые не дошли до активации.

```
df.query('ind_frod==1')['activation_date'].isna().mean()
# 0.4309608540925267
```

Оценка полноты контроля получилась $\text{recall}_C = 0.43$.

Зададим вероятности ошибок первого и второго рода: $\alpha = 0.05$ и $\beta = 0.2$.

Чем больше продлится эксперимент и чем больше будет recall_T, тем меньше будет ошибка второго рода. Рассмотрим несколько вариантов, используя синтетические AA- и АБ-тесты.

Будем отбирать данные за указанное количество дней, начиная с 1 июля (потому что в июле в России нет праздников):

```
def get_experiment_data(df: pd.DataFrame, start: datetime,
                        days: int) -> pd.DataFrame:
    end = start + timedelta(days=days - 1)
    df = df[df['registration_date'].between(start, end)]
    df = df.query('ind_frod == 1').copy()
    return df
```

На отобранных данных проведем тесты:

```
import numpy as np
import pandas as pd
from scipy import stats
```

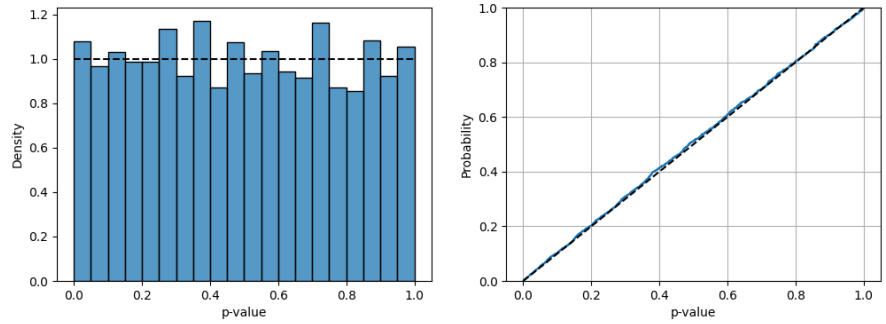
```
def make_synthetic_aa_test(df: pd.DataFrame, recall_c: float) -> np.ndarray:
    arr = []
    ind_frod = df['ind_frod'].values
    for _ in range(10_000):
        split = np.random.binomial(n=1, p=0.5, size=len(df))
        control_frauds = int(ind_frod @ (1 - split))
        test_frauds = int(ind_frod @ split)
        control = np.where(np.random.rand(control_frauds) < recall_c, 1, 0)
        test = np.where(np.random.rand(test_frauds) < recall_c, 1, 0)
        arr.append(stats.ttest_ind(control, test).pvalue)
    return np.array(arr)


def make_synthetic_ab_test(df: pd.DataFrame, recall_c: float,
                           recall_t: float) -> np.ndarray:
    arr = []
    ind_frod = df['ind_frod'].values
    for _ in range(10_000):
        split = np.random.binomial(n=1, p=0.5, size=len(df))
        control_frauds = int(ind_frod @ (1 - split))
        test_frauds = int(ind_frod @ split)
        control = np.where(np.random.rand(control_frauds) < recall_c, 1, 0)
        test = np.where(np.random.rand(test_frauds) < recall_t, 1, 0)
        arr.append(stats.ttest_ind(control, test).pvalue)
    return np.array(arr)
```

Имеем следующие результаты для оценки вероятности ошибки второго рода:

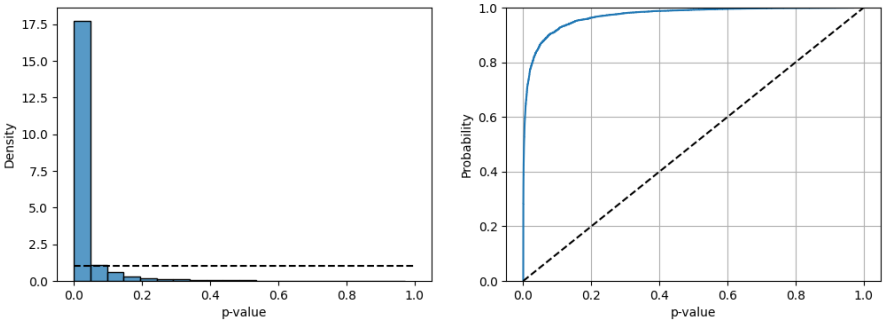
days \ recall_T	0.6	0.65	0.70	0.75	0.80
7	0.7	0.54	0.36	0.2	0.09
14	0.53	0.31	0.14	0.04	0.01
21	0.39	0.17	0.05	0.01	0.0

Эта таблица помогает оценивать необходимую длительность эксперимента, если мы имеем заданное recall_T и хотим контролировать ошибки 1-го и 2-го рода на заданных уровнях. Например, пусть мы имеем ML-модель с $\mathrm{recall}_T = 0.7$. Тогда нам придется взять 2 недели данных.

Вот как выглядит empirical CDF для p-value синтетических тестов, если длительность эксперимента 2 недели, а $\mathrm{recall}_T = 0.7$.



Синтетический AA-тест, P(type I error) = 0.052



Синтетический АБ-тест, P(type II error) = 0.140

Комментарий: для проведения эксперимента выбирается длительность кратная 7 дням, чтобы учесть возможную сезонность в поведении продавцов.

План действий в зависимости от результатов эксперимента

Если recall_T стат. значимо больше recall_C и $\text{FPR} \leq \text{FPR}_{max}$, начинаем использовать ML-алгоритм. Иначе используем старую систему модерации.

Вопросы

Как определить мошенника?

1. На этапе регистрации - ложные / украденные / противоречивые данные.
2. На этапе публикации каталога товаров - ложная информация о продукте.
3. На этапе продаж:
   - увод на свой сайт;
   - попытки узнать контакты клиента;
   - несоответсвующий описанию товар.

Какие ещё могут быть схемы мошенничества?

Мошенничество с картами (детали зависят от правил маркетплейса и платежных систем):

1. Создается продавец.
2. Создаются подставные покупатели.
3. С помощью украденных данных о картах (номер, CVV) совершаются покупки.

4. Продавец пытается вывести деньги до того, как настоящие владельцы карт увидят списание и запросят возврат.

Мошенничество с возвратом средств (детали зависят от правил маркетплейса и платежных систем):

1. Продавец покупает через подставного покупателя товар.
2. Покупатель оформляет возврат после того, как деньги поступают продавцу.
3. И продавец, и покупатель могут остаться с деньгами, а расходы лягут на площадку.

Какие фичи могут помочь клиентам избежать неприятностей с мошенниками?

1. Если продавец запрашивает данные или просит уйти по внешней ссылке, выводить пользователю предупреждение.
    - Возможно, стоит вовсе исключать из сообщений ссылки и телефоны.
2. Для популярных продуктов проверять ключевые характеристики и в случае сильного расхождения инициировать модерацию.
3. Контроль качества отзывов:
    - много отзывов от одного человека;
    - дубликаты;
    - отзывы от непокупателей;
    - бот сети.
4. Для товара показывать среднюю цену по площадке.
    - Позволяет увидеть, что у данного продавца цена аномально завышена или занижена.

Через какую механику мошенник узнает контакты покупателя?

1. Соцсети
2. Профиль самого маркетплейса
3. Спрашивает в чате маркетплейса
4. Гугл/Яндекс поиск по ФИО или по аватарке

Что можем сделать, чтобы усложнить жизнь мошенникам?

1. Не давать продавцам доступ к контактам покупателей.
2. Сделать интеграцию с сервисами, которые на этапе регистрации будут проверять информацию о юр. лицах. (Например https://focus.kontur.ru).
3. Предупреждать клиента об опасности перехода по внешним ссылкам, об опасности передавать данные.