Банки — Сегментация пользователей по потреблению

Цель: сегментировать пользователей на основе потребляемых продуктов банка, для составления более детализированных маркетинговых кампаний, нацеленых на продажу продуктов банка определенным сегментам.

Презентация - https://drive.google.com/file/d/1b1jOYjmieJHCDebNRbvqITIhIPEMosxp/view?usp=sharing

Дашборд - https://public.tableau.com/authoring/1TableauPublic/Sheet1#1

Изучим данные и подготовим их к анализу

импортируем необходимые бибилотеки

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from scipy import stats as st
```

чтение файла bank_scrooge.csv с данными и сохранение его в переменную

```
path = "https://drive.google.com/uc?export=download&id=1-U61mhTz_N1ARjy2XSAZ7IlQqGjeqP0F"
df = pd.read_csv(path)
```

Подготовим данные для анализа(убрать дубликаты, пропуски, сделать из качественных колличественные перменные)

рассмотрим типы данных

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype
```

```
---  ------          --------------  -----
 0   USERID          10000 non-null  int64
 1   score           10000 non-null  float64
 2   city            10000 non-null  object
 3   gender          10000 non-null  object
 4   age             9974 non-null   float64
 5   equity          10000 non-null  int64
 6   balance         7705 non-null   float64
 7   products        10000 non-null  int64
 8   credit_card     10000 non-null  int64
 9   last_activity   10000 non-null  int64
 10  EST_SALARY      10000 non-null  float64
 11  churn           10000 non-null  int64
dtypes: float64(4), int64(6), object(2)
memory usage: 937.6+ KB
None
```

приведем название колонок USERID и EST_SALARY к нижнему регистру

```
df = df.rename(columns={"EST_SALARY": "est_salary", "USERID":
"user_id"})
```

проверим данные на наличие дубликатов

```
print('всего дубликатов:', df.duplicated().sum())
print('доля дубликатов:', df.duplicated().sum() /
df.duplicated().count())

df = df.drop_duplicates().reset_index(drop=True)
```

```
всего дубликатов: 0
доля дубликатов: 0.0
```

создадим 3 колонки с городами в бинарном фомате

используем небольшую функцию

```
def city_num_1(elem):
    if elem == 'Ярославль':
        return 1
    else:
        return 0

def city_num_2(elem):
    if elem == 'Ростов':
        return 1
    else:
        return 0
```

```
def city_num_3(elem):
    if elem == 'Рыбинск':
        return 1
    else:
        return 0

df['city_yaroslav'] = df['city'].agg(city_num_1)
df['city_rostov'] = df['city'].agg(city_num_2)
df['city_ribinsk'] = df['city'].agg(city_num_3)
```

создадим 2 колонки с полами в бинарном фомате

```
def gender_num_1(elem):
    if elem == 'М':
        return 1
    else:
        return 0

def gender_num_2(elem):
    if elem == 'Ж':
        return 1
    else:
        return 0

df['male'] = df['gender'].agg(gender_num_1)
df['female'] = df['gender'].agg(gender_num_2)
```

Проверим данные по деньгам на счете, как они отображаются

проверка на дубликаты, пропущенных значений возраста слишком мало, их можно спокойно удалить. Рассмотрим пропущенные значения баланса подоробнее

```
print(df.isna().sum())
```

```
user_id             0
score               0
city                0
gender              0
age                26
equity              0
balance          2295
products            0
credit_card         0
last_activity       0
est_salary          0
```

```
churn                0
city_yaroslav        0
city_rostov          0
city_ribinsk         0
male                 0
female               0
dtype: int64
```

баланс ноль всего у двух аккаунтов из 10000.

```
print(len(df.query('balance == 0')))
```

```
2
```

для простоты заполним все пропущеннные значения нулями, посмотрим результат.

создадим датафрейм example со всеми пропущенными значениями + 2 изначальных нуля(они ничего не меняют)

```
df['balance'] = df['balance'].fillna(0)
```

```
df = df.dropna()
```

```
example = (df.query('balance == 0'))
```

```
example['products'].count()
```

```
2281
```

```
example['products'].sum()
```

```
2655
```

```
example['credit_card'].sum()
```

```
1860
```

лишь 421 клиент из 2281 не имеют кредиток, скорее всего баланс не отображается из-за отсутвия дебетовой карты. В датафрейме example всего на 400 продуктов больше чем людей.

пометим этих людей в новой колонке no_debet, возможно вернемся к ним позже когда будем составлять сегменты.

```
example = example.query('credit_card == 1')['user_id']
```

```
df['no_debet'] = df.query('credit_card == 1 and user_id in @example')
['credit_card']
```

Выделим ряд сегментов на основе потребления

попытаемся выделить сегменты, они отличаются по количеству потребляемых продуктов.

```
df['products'].unique()
```

```
array([2, 3, 1, 4, 5, 0])
```

всего существует 5 варинтов потребления (будем игнорировать 0 продуктов).

```
df.groupby('products')['user_id'].count()
```

```
products
0       1
1    3323
2    5119
3    1038
4     474
5      19
Name: user_id, dtype: int64
```

Создадим первый сегмент

```
segment_one = df.query('products == 1')
```

мы уже распределили 34% клиентов "начинающих" с одним продуктом

нам остается распределить пользователей с 2 и более продуктами. Пик колличества клиентов находится на двух продуктах. Выделим их в отдельный сегмент "средняков"

```
segment_two = df.query('products == 2')
```

оставшихся клиентов ~1500 ~15% распределим в третий сегмент "продвинутый"

```
segment_three = df.query('products > 2')
```

Сравним среднии и медианы сегментов

```
for elem in ['score', 'age', 'equity', 'balance', 'products',
'credit_card', 'last_activity', 'est_salary', 'churn']:
    print('element', elem)
    print('mean // median')
    print(segment_one[elem].mean(), '//', segment_one[elem].median())
    print(segment_two[elem].mean(), '//' , segment_two[elem].median())
    print(segment_three[elem].mean(), '//',
```

```
segment_three[elem].median())
    print('///////////////')

element score
mean // median
854.2952151670178 // 883.0
844.5585075210003 // 844.0
850.2155453951666 // 848.0
///////////////
element age
mean // median
42.19620824556124 // 40.0
42.76890017581559 // 40.0
43.76355323318093 // 41.0
///////////////
element equity
mean // median
1.3298224495937405 // 0.0
3.1506153545614377 // 3.0
3.7165251469627694 // 4.0
///////////////
element balance
mean // median
255398.0162293109 // 0.0
762175.8541336198 // 476496.99
1055120.1768060091 // 698914.43
///////////////
element products
mean // median
1.0 // 1.0
2.0 // 2.0
3.334421946440235 // 3.0
///////////////
element credit_card
mean // median
0.8636773999398134 // 1.0
0.6257081461222895 // 1.0
0.4644023514043109 // 0.0
///////////////
element last_activity
mean // median
0.5275353596148059 // 1.0
0.5155303770267631 // 1.0
0.5375571521881124 // 1.0
///////////////
element est_salary
mean // median
180605.23381582907 // 142163.0
130400.3787458488 // 109508.21
134768.02451338994 // 112401.7
```

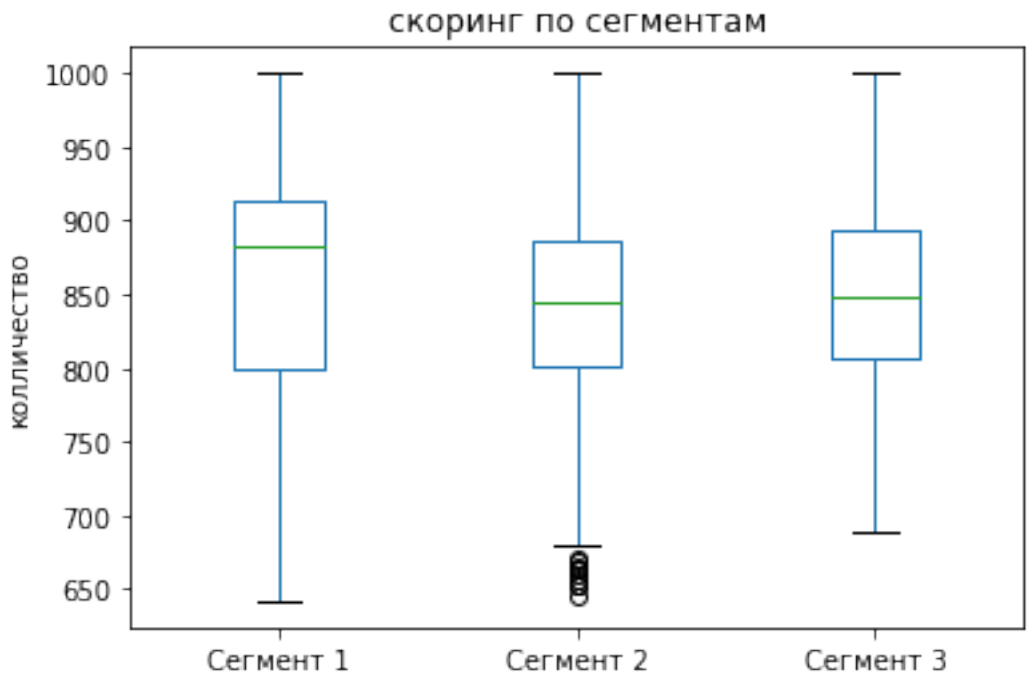
```
//////////////
element churn
mean // median
0.07071922961179657 // 0.0
0.1910529400273491 // 0.0
0.395166557805356 // 0.0
//////////////
```

создадим отдельный столбец с сегментами

```
def segment(elem):
    if elem == 1:
        return 1
    elif elem == 2:
        return 2
    else:
        return 3
df['segment'] = df['products'].agg(segment)
```

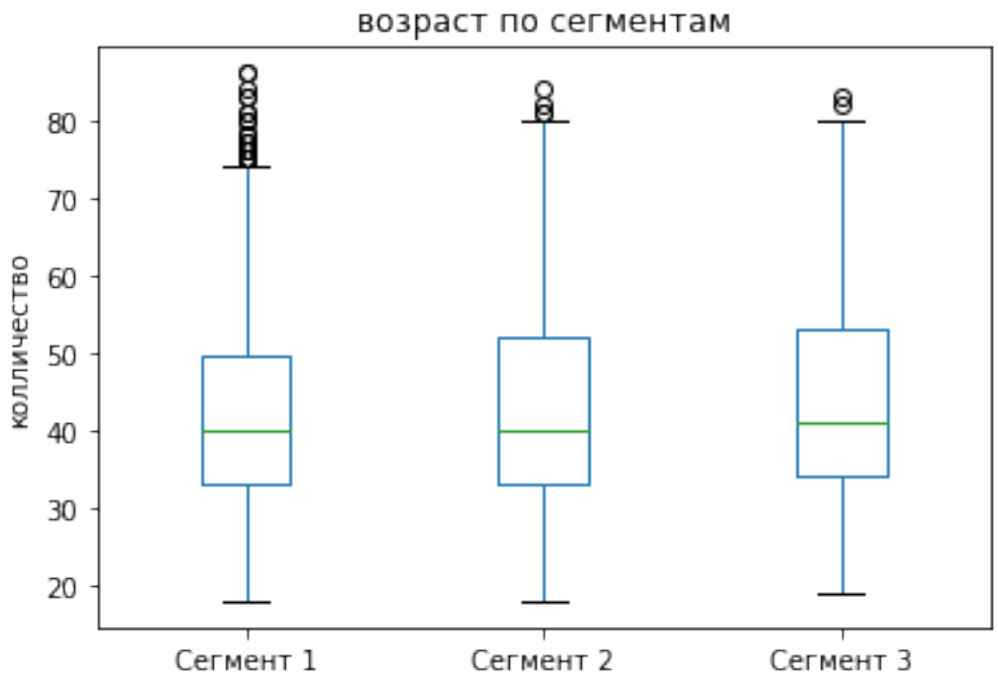
Между сегментами практически нет разницы в скоринге

```
data = pd.DataFrame({"Сегмент 1": df.query('products == 1')['score'],
"Сегмент 2": df.query('products == 2')['score'], "Сегмент
3":df.query('products > 2')['score']})

ax = data[['Сегмент 1', 'Сегмент 2', 'Сегмент 3']].plot(kind='box',
title='скоринг по сегментам')
plt.ylabel('колличество')

plt.show()
```



Между сегментами практически нет разницы в возрасте

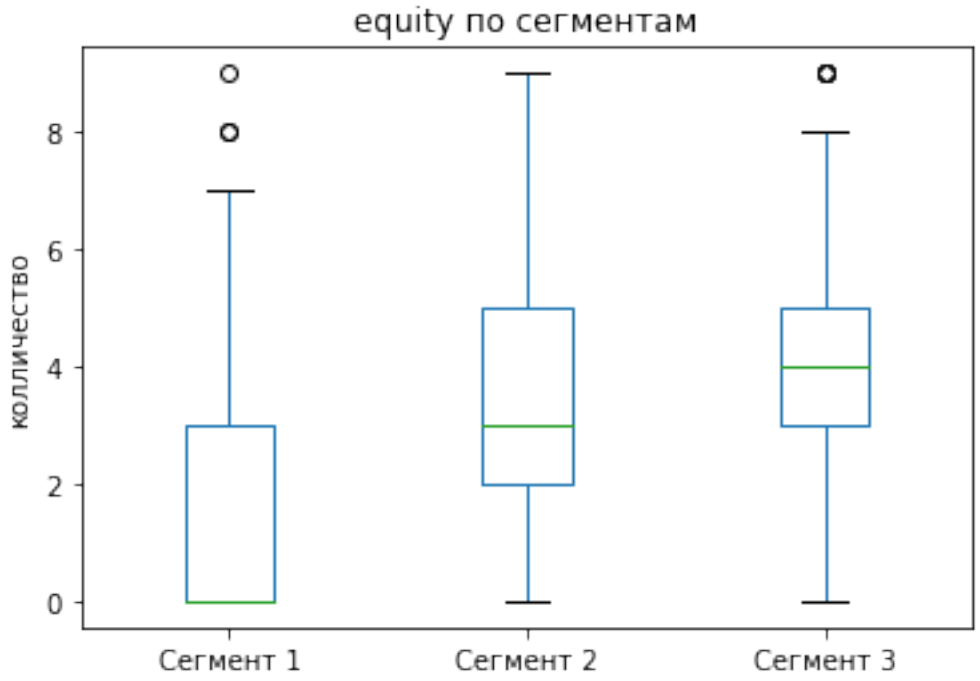
```
data = pd.DataFrame({"Сегмент 1": df.query('products == 1')['age'],
"Сегмент 2": df.query('products == 2')['age'], "Сегмент
3":df.query('products > 2')['age']})

ax = data[['Сегмент 1', 'Сегмент 2', 'Сегмент 3']].plot(kind='box',
title='возраст по сегментам')
plt.ylabel('колличество')

plt.show()
```



Между сегментами есть значительная разница в equity

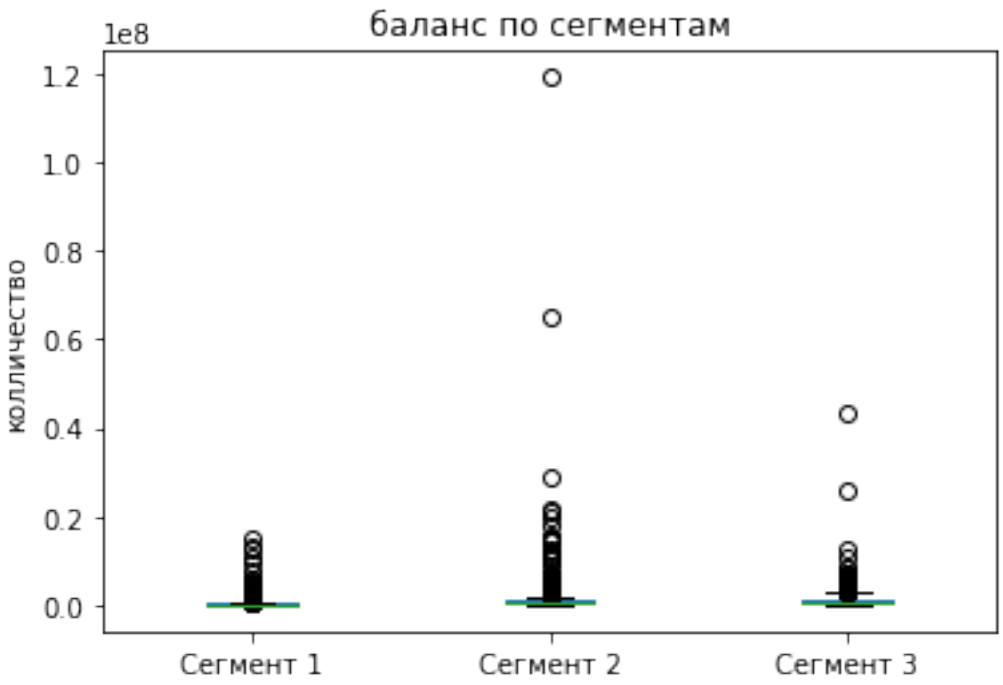
меньше всего показатели в первом, больше всего в третьем

```
data = pd.DataFrame({"Сегмент 1": df.query('products == 1')['equity'],
"Сегмент 2": df.query('products == 2')['equity'], "Сегмент
3":df.query('products > 2')['equity']})

ax = data[['Сегмент 1', 'Сегмент 2', 'Сегмент 3']].plot(kind='box',
title='equity по сегментам')
plt.ylabel('колличество')

plt.show()
```



баланс выше всего во втором сегменте

```
data = pd.DataFrame({"Сегмент 1": df.query('products == 1')
['balance'], "Сегмент 2": df.query('products == 2')['balance'],
"Сегмент 3":df.query('products > 2')['balance']})

ax = data[['Сегмент 1', 'Сегмент 2', 'Сегмент 3']].plot(kind='box',
title='баланс по сегментам')
plt.ylabel('колличество')

plt.show()
```

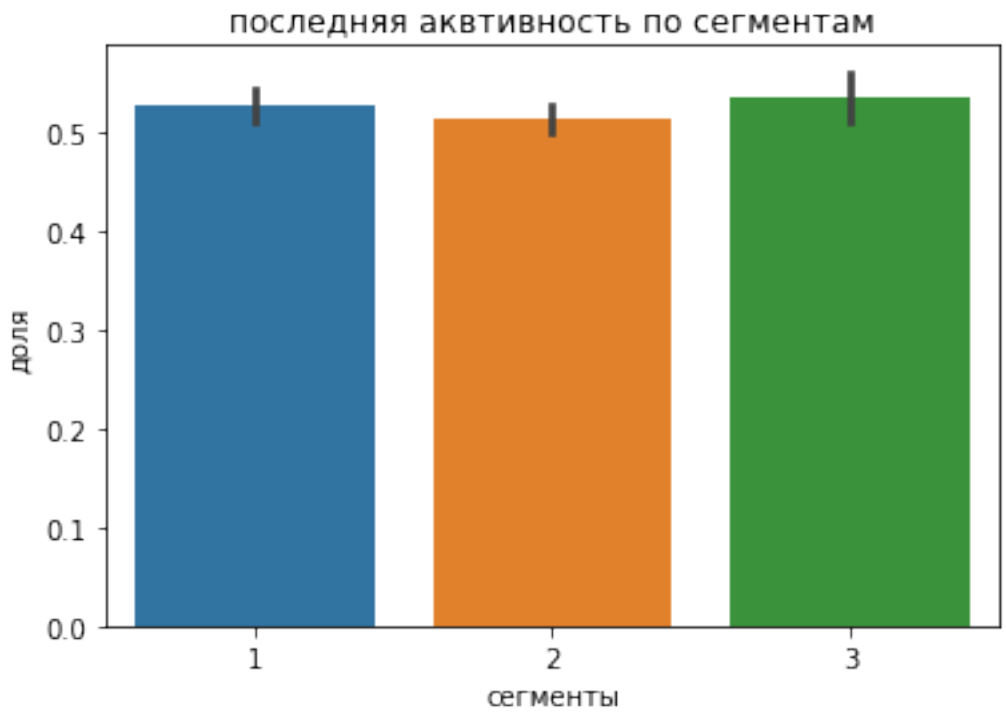


Визуализируем среднее кол-во кредитных карт, их доля уменьшается к 2-3 сегментам

```
sns.barplot (x=df['segment'], y=df['credit_card'], data=df)
plt.title('кредитные карты по сегментам')
plt.xlabel('сегменты')
plt.ylabel('доля')

Text(0, 0.5, 'доля')
```

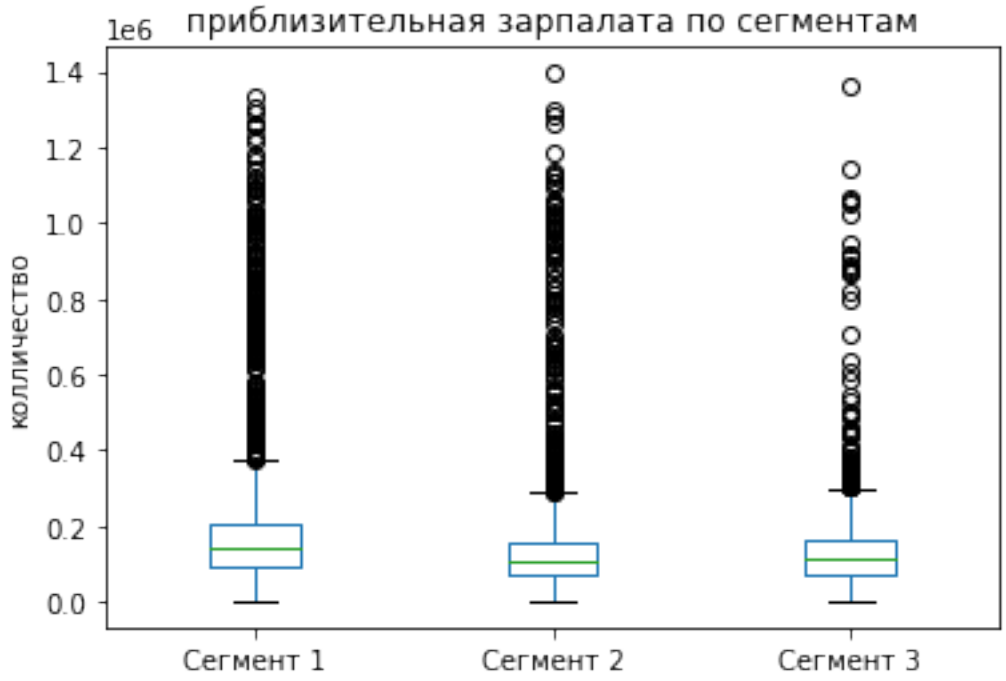
Между сегментами практически нет разницы в последней активности

```
sns.barplot (x=df['segment'], y=df['last_activity'], data=df)
plt.title('последняя аквтивность по сегментам')
plt.xlabel('сегменты')
plt.ylabel('доля')

Text(0, 0.5, 'доля')
```



В сегменте один "приблизительная зарплата" значительно выше, почему не ясно

```
data = pd.DataFrame({"Сегмент 1": df.query('products == 1')
['est_salary'], "Сегмент 2": df.query('products == 2')['est_salary'],
"Сегмент 3":df.query('products > 2')['est_salary']})

ax = data[['Сегмент 1', 'Сегмент 2', 'Сегмент 3']].plot(kind='box',
title='приблизительная зарпалата по сегментам')
plt.ylabel('колличество')

plt.show()
```



В первом сегменте много клиентов без баланса, с одним продуктом, большинство с кредитными картами.

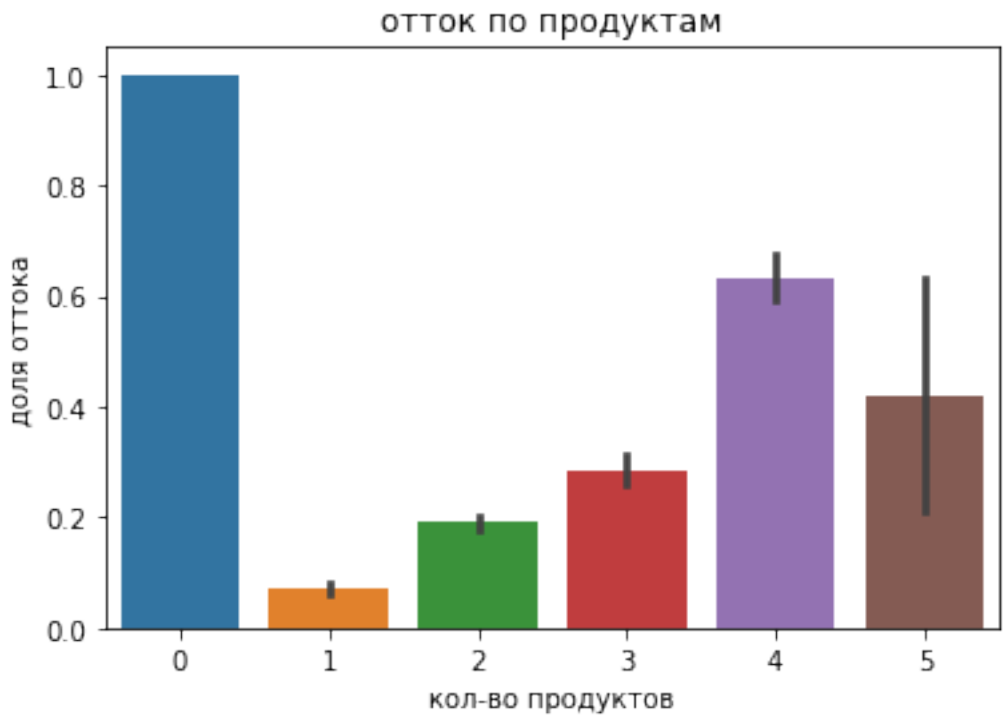
Во втором сегменте много клиентов с балансом, лишь у половины есть кредитные карты, больше оценка недвижемости, повышается отток

Во третьем сегменте баланс еще выше, меньше чем у половины есть кредитные карты, самая высокая оценка недвижемости, высокий отток

Визуалцизация оттока

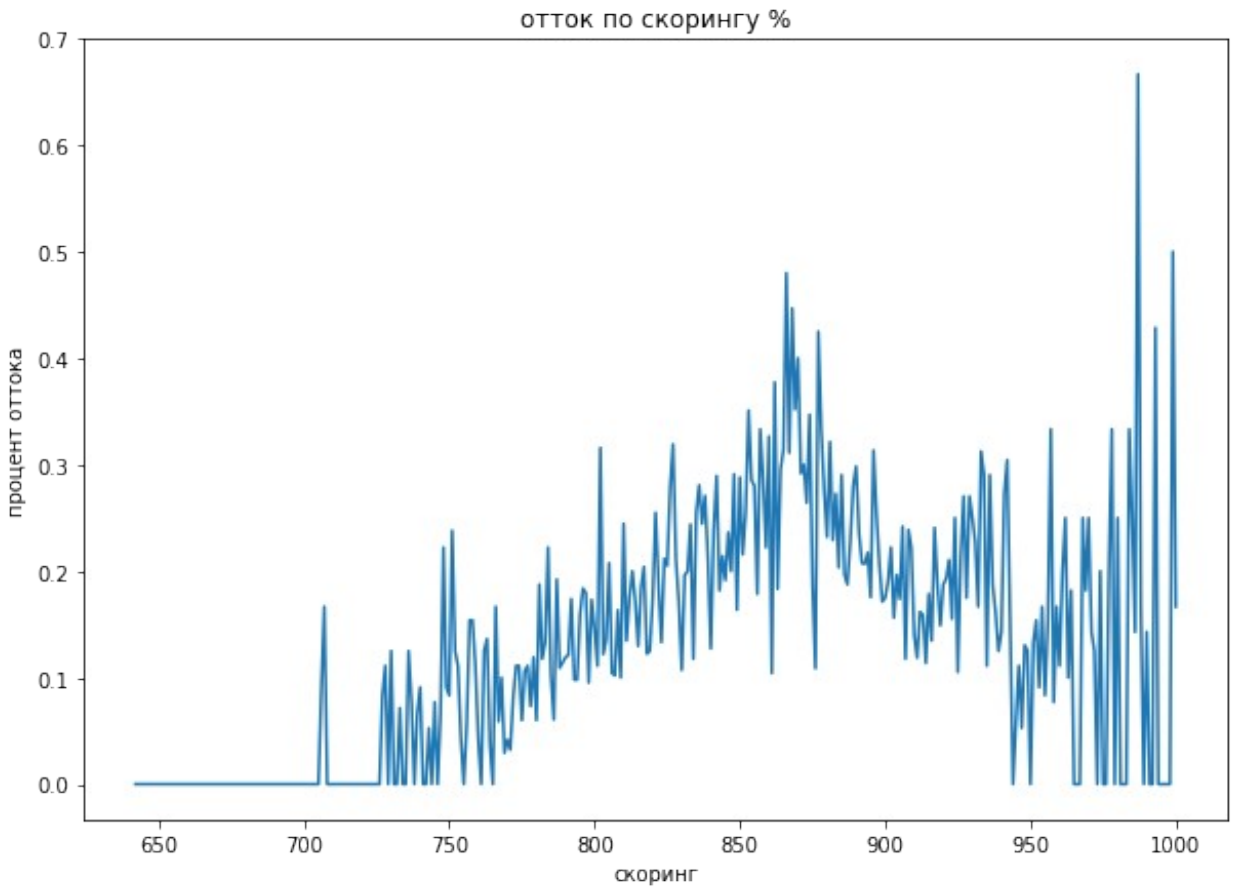
отток растет до 4 продуктов, однако людей с пятью и больше продуктами слишком мало чтобы делать какие-то выводы

```
sns.barplot (x=df['products'], y=df['churn'], data=data)
plt.title('отток по продуктам')
plt.xlabel('кол-во продуктов')
plt.ylabel('доля оттока')

Text(0, 0.5, 'доля оттока')
```



отток выше у клиентов с рейтингом от 850 до 900, и от 950

```
fig, ax = plt.subplots(figsize =(10, 7))
ax.plot(df.groupby('score')['churn'].mean())

plt.title('отток по скорингу %')
plt.xlabel('скоринг')
plt.ylabel('процент оттока')

plt.show()
```



отток растет у клентов от 20 до 30, и от 45 до 55

создадим визуализацию оттока по возрасту

```
x = list(set(df['age'].tolist()))
y = []
for elem in x:
    y.append(round(df.query('age == @elem')['churn'].mean()*100))


first_age = []
second_age = []
third_age = []
fourth_age = []
i = 0
for elem in y:
    if i < 17:
        first_age.append(elem)
    elif i < 34:
        second_age.append(elem)
    elif i < 51:
        third_age.append(elem)
    else:
```

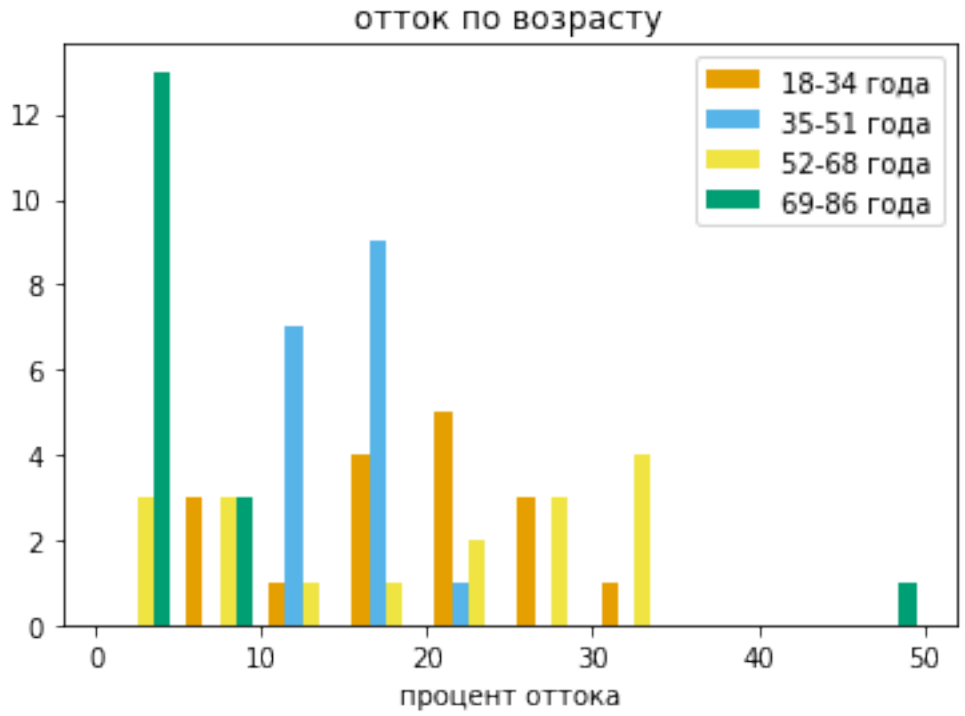
```
        fourth_age.append(elem)
    i += 1
print(len(first_age), len(second_age), len(third_age), len(fourth_age)
)
```

```
17 17 17 17
```

сделаем 4 равных сегмента по возрасту

```
colors = ['#E69F00', '#56B4E9', '#F0E442', '#009E73']
names = ['18-34 года', '35-51 года', '52-68 года',
         '69-86 года']

plt.hist([first_age, second_age, third_age, fourth_age], bins = 10,
         color = colors, label=names)

plt.legend()
plt.xlabel('процент оттока')
plt.title('отток по возрасту')
```

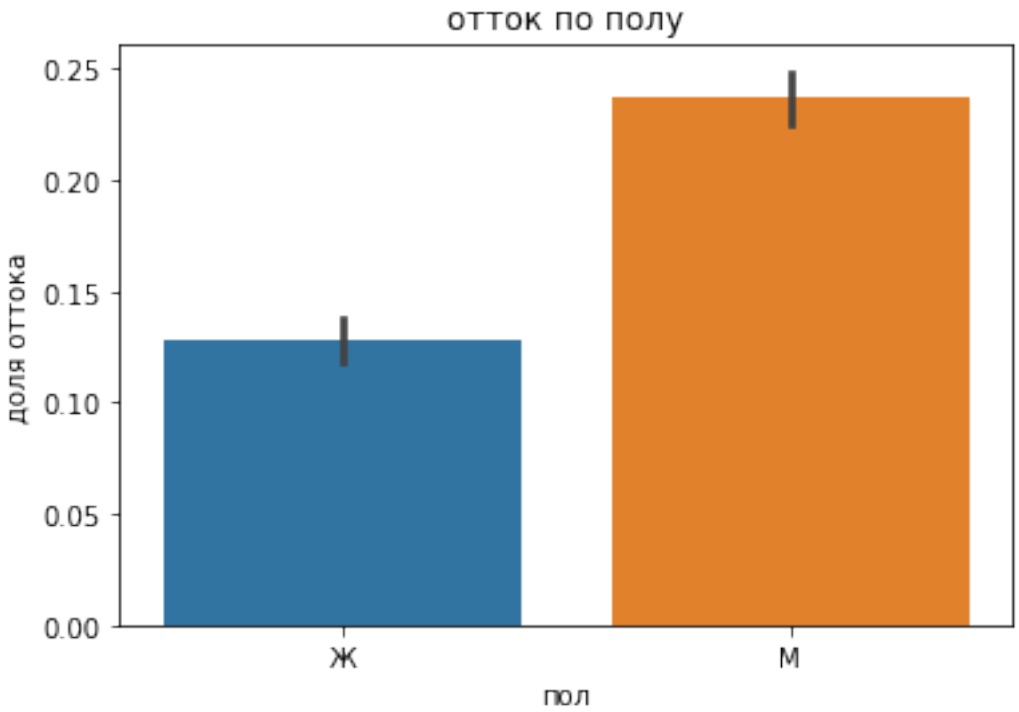
```
Text(0.5, 1.0, 'отток по возрасту')
```



отток выше у мужчин

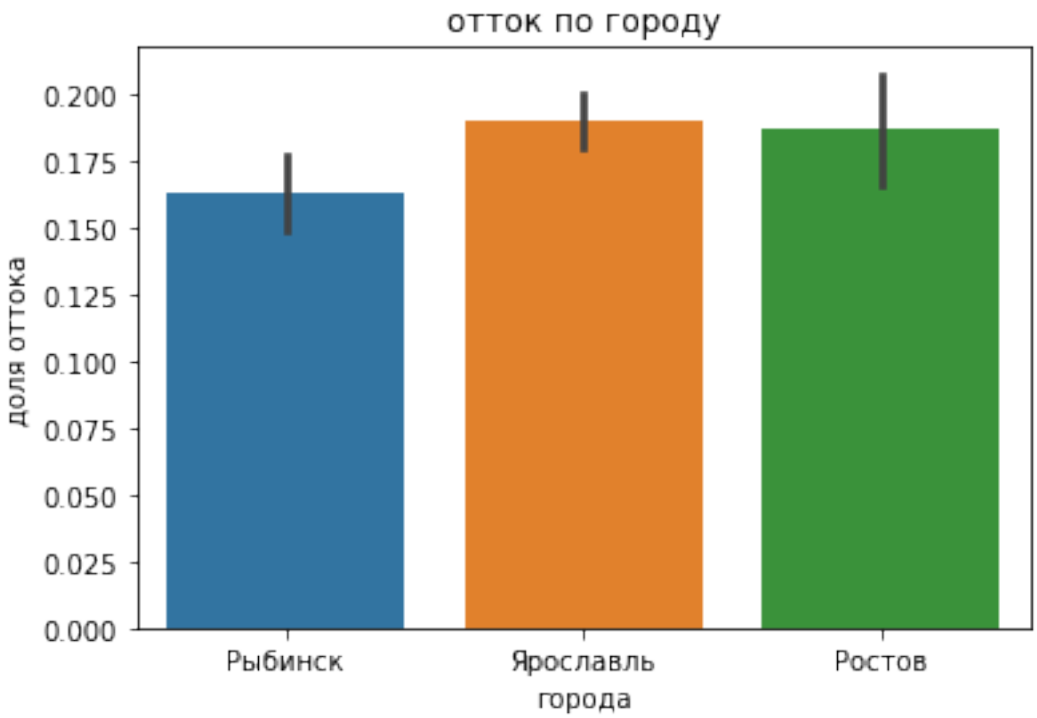
```
sns.barplot (x=df['gender'], y=df['churn'], data=data)
plt.title('отток по полу')
```

```
plt.xlabel('пол')
plt.ylabel('доля оттока')

Text(0, 0.5, 'доля оттока')
```



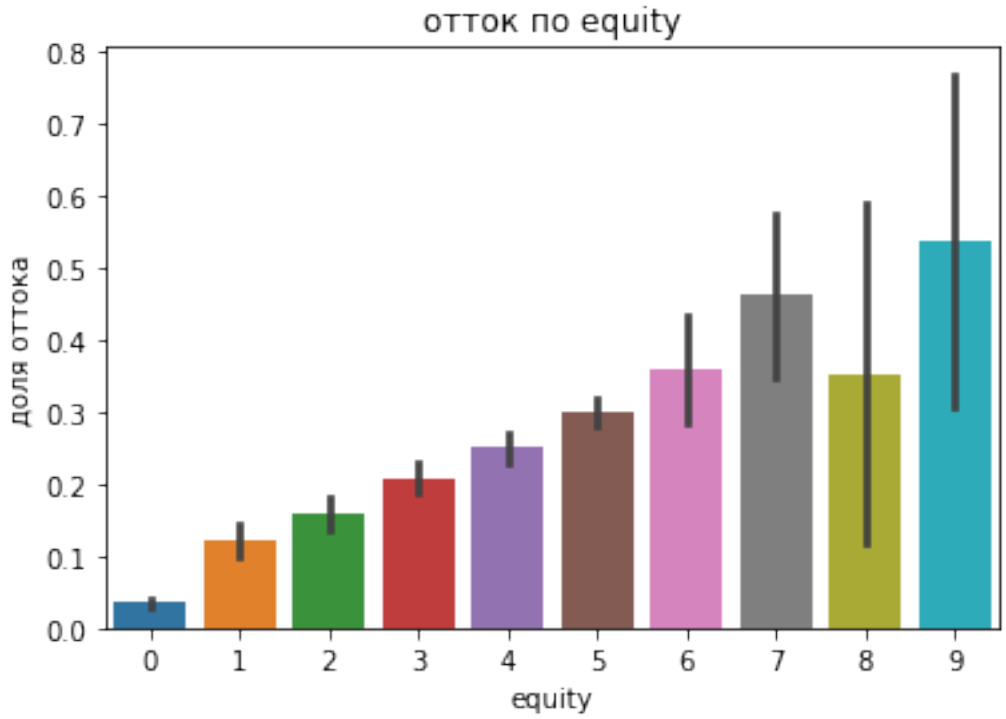
отток чуть меньше в Рыбинске

```
sns.barplot (x=df['city'], y=df['churn'], data=data)
plt.title('отток по городу')
plt.xlabel('города')
plt.ylabel('доля оттока')

Text(0, 0.5, 'доля оттока')
```



чем больше equity, тем больше вероятность ухода

```
sns.barplot (x=df['equity'], y=df['churn'], data=data)
plt.title('отток по equity')
plt.xlabel('equity')
plt.ylabel('доля оттока')

Text(0, 0.5, 'доля оттока')
```



Рассмотрим корреляции

```
bank_stat_multi = df.pivot_table(index='user_id', values=['score',
'age', 'equity', 'balance', 'products', 'credit_card',
'last_activity', 'est_salary', 'churn', 'city_yaroslav',
'city_rostov', 'city_ribinsk', 'male', 'female'])

bank_stat_multi.corr().head(20)
```

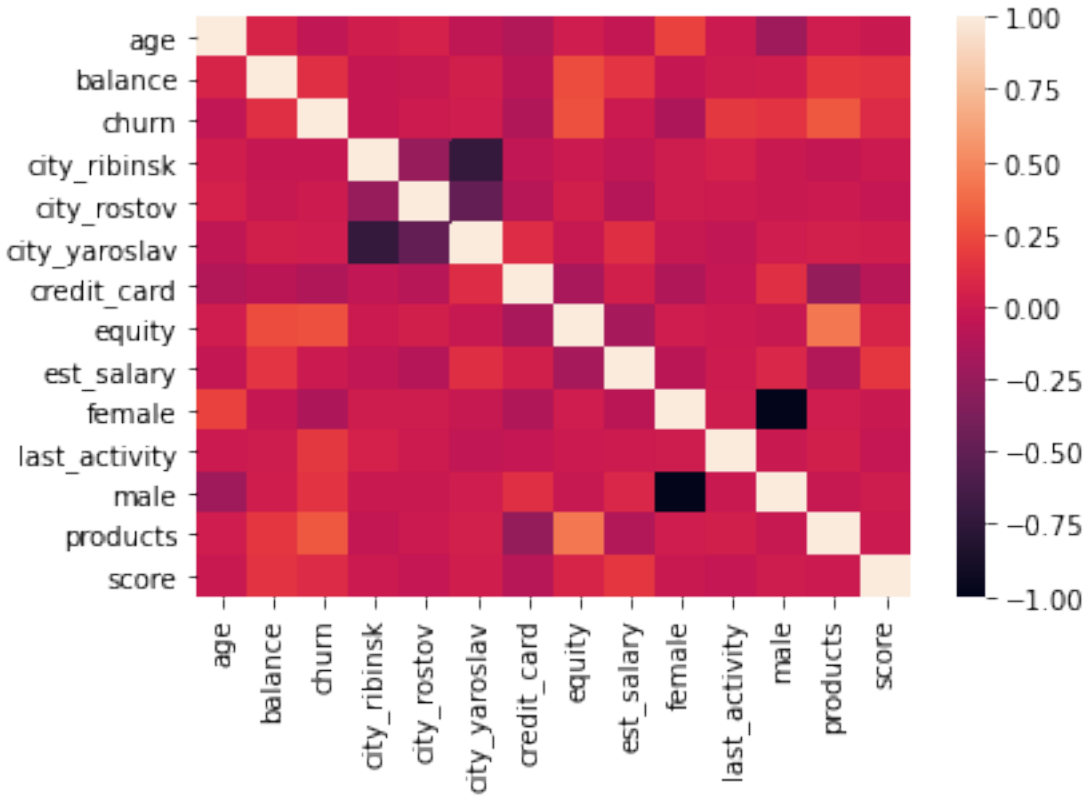
	age	balance	churn	city_ribinsk	city_rostov
age	1.000000	0.066956	-0.047006	0.028101	0.046890
balance	0.066956	1.000000	0.127544	-0.025815	-0.019650
churn	-0.047006	0.127544	1.000000	-0.030977	0.006785
city_ribinsk	0.028101	-0.025815	-0.030977	1.000000	-0.247541
city_rostov	0.046890	-0.019650	0.006785	-0.247541	1.000000
city_yaroslav	-0.058705	0.037232	0.023032	-0.723188	-0.490138
credit_card	-0.123334	-0.082898	-0.129284	-0.050403	-0.098674
equity	0.021403	0.250676	0.266245	-0.003810	0.035891

```
est_salary    -0.039718  0.155487 -0.001167     -0.054151    -0.106329

female         0.211735 -0.031006 -0.141060      0.011694     0.014758

last_activity -0.007800  0.015610  0.166059      0.048806     0.007654

male          -0.211735  0.031006  0.141060     -0.011694    -0.014758

products       0.018690  0.157342  0.297584     -0.043436    -0.006026

score         -0.010854  0.142902  0.103394     -0.000146    -0.037411


                city_yaroslav  credit_card    equity  est_salary
female  \
age                -0.058705    -0.123334  0.021403   -0.039718
0.211735
balance             0.037232    -0.082898  0.250676    0.155487 -
0.031006
churn               0.023032    -0.129284  0.266245   -0.001167 -
0.141060
city_ribinsk       -0.723188    -0.050403 -0.003810   -0.054151
0.011694
city_rostov        -0.490138    -0.098674  0.035891   -0.106329
0.014758
city_yaroslav       1.000000     0.115684 -0.022156    0.124512 -
0.021040
credit_card         0.115684     1.000000 -0.166711    0.036979 -
0.128114
equity             -0.022156    -0.166711  1.000000   -0.174250
0.021710
est_salary          0.124512     0.036979 -0.174250    1.000000 -
0.081349
female             -0.021040    -0.128114  0.021710   -0.081349
1.000000
last_activity      -0.049364    -0.033139 -0.003320    0.002230
0.012985
male                0.021040     0.128114 -0.021710    0.081349 -
1.000000
products            0.043373    -0.253316  0.431688   -0.117908
0.022755
score               0.026799    -0.094921  0.063018    0.162975 -
0.010052

                last_activity      male  products     score
age                -0.007800 -0.211735  0.018690 -0.010854
balance             0.015610  0.031006  0.157342  0.142902
churn               0.166059  0.141060  0.297584  0.103394
city_ribinsk        0.048806 -0.011694 -0.043436 -0.000146
city_rostov         0.007654 -0.014758 -0.006026 -0.037411
```

```
city_yaroslav      -0.049364  0.021040  0.043373  0.026799
credit_card        -0.033139  0.128114 -0.253316 -0.094921
equity             -0.003320 -0.021710  0.431688  0.063018
est_salary          0.002230  0.081349 -0.117908  0.162975
female              0.012985 -1.000000  0.022755 -0.010052
last_activity       1.000000 -0.012985  0.036527 -0.031491
male               -0.012985  1.000000 -0.022755  0.010052
products            0.036527 -0.022755  1.000000 -0.004460
score              -0.031491  0.010052 -0.004460  1.000000
```

```
sns.heatmap(bank_stat_multi.corr())
```

```
<AxesSubplot:>
```



Для анализа корреляций используем шкалу Чеддока.

equity churn - существует слабая корреляция между высокой оценкой недвижемости и уходом клиентов

balance equity - чем выше балланс тем больше оценка недвижемости, переменные связаны слабой корреляцией

balance score - скоринг и баланс также связаны слабой корреляцией

product churn - чем больше продуктов у клиента, тем больше вероятность ухода(слабая корреляция)

male churn + age - с возрастом мужчин становится меньше и они больше подвержены уходу(слабая корреляция)

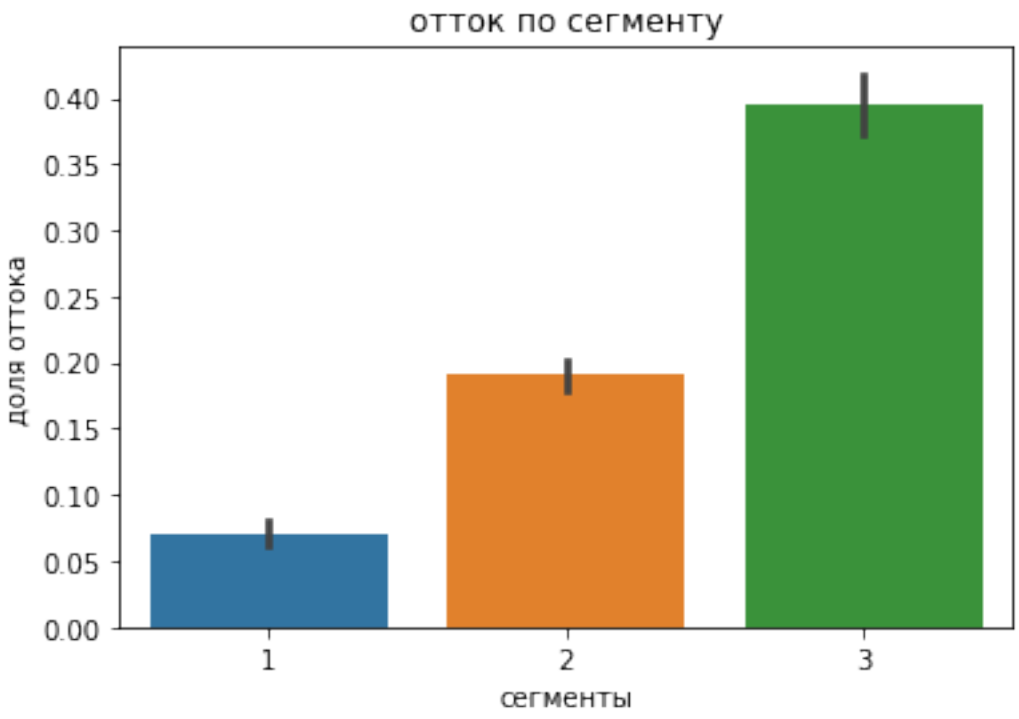
Отдельно рассмотрмим общие черты уходящих пользователей

будем считать, что определенный процент людей всегда уходит, нас будут интересовать лишь ушедшие из 2 и 3 сегментов (где процент ушедших слишком высок)

рассмотрим ['churn'] клиентов по сегментам. Нас полносью устраивают показатели первого сегмента, второй сегмент незначительно отличается от желаемого результата. Третий сегмент требует вмешательства, почти каждый второй клиент с тремя и более заказами перестает пользоваться услугами банка.

```
sns.barplot (x=df['segment'], y=df['churn'], data=df)
plt.title('отток по сегменту')
plt.xlabel('сегменты')
plt.ylabel('доля оттока')
```

```
Text(0, 0.5, 'доля оттока')
```




```
segments = [segment_one, segment_two, segment_three]
```

```
for elem in [segment_one, segment_two, segment_three]:
    print(elem['churn'].sum() / elem['user_id'].count() * 100)
```

```
7.071922961179657
19.10529400273491
39.5166557805356
```

отдельно рассмотрим черты уходящих пользователей 2 и 3 сегментов, на всякий случай сравним кол-во мужчин и женщин.

Две основные корреляции equity и пол(male)

```
df.query('products == 2').groupby('gender')['user_id'].count()
```

```
gender
Ж    2676
M    2443
Name: user_id, dtype: int64
```

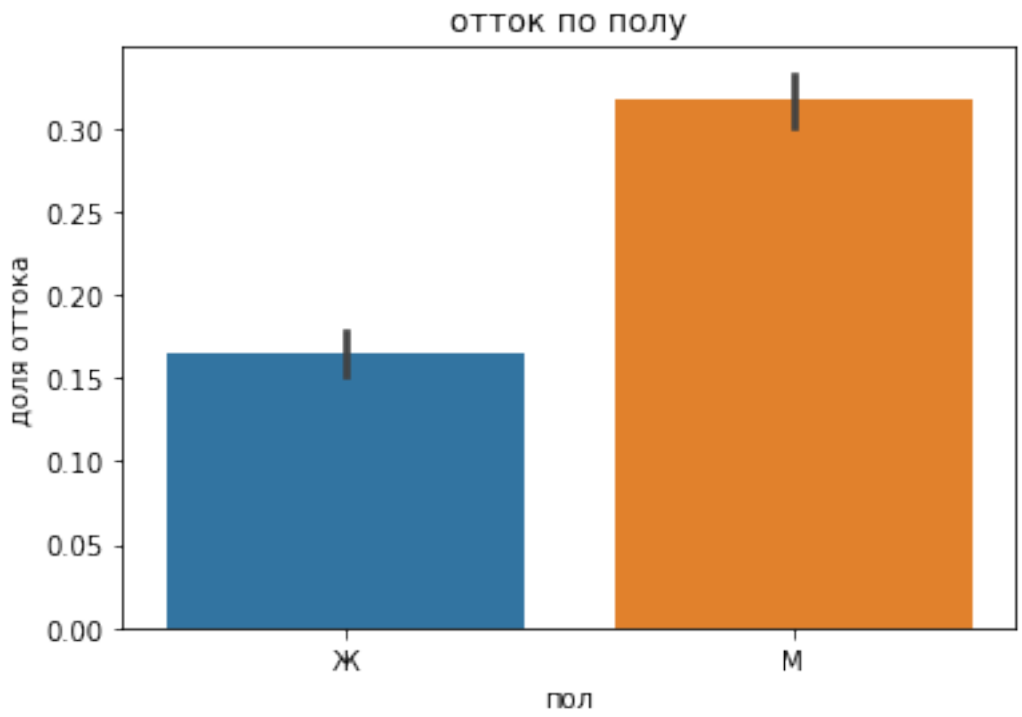
```
df.query('products > 2').groupby('gender')['user_id'].count()
```

```
gender
Ж    779
M    752
Name: user_id, dtype: int64
```

узнаем долю ушедших по полу второго и третьего сегментов

```
sns.barplot (x=df.query('products>1')['gender'], y=df['churn'],
data=df)
plt.title('отток по полу')
plt.xlabel('пол')
plt.ylabel('доля оттока')
```

```
Text(0, 0.5, 'доля оттока')
```




```
df.query('products == 2 and churn == 1').groupby('gender')
['user_id'].count()/df.query('products == 2').groupby('gender')
['user_id'].count()*100

gender
Ж    12.780269
М    26.033565
Name: user_id, dtype: float64

df.query('products > 2 and churn == 1').groupby('gender')
['user_id'].count()/df.query('products > 2').groupby('gender')
['user_id'].count()*100

gender
Ж    29.396662
М    50.000000
Name: user_id, dtype: float64
```

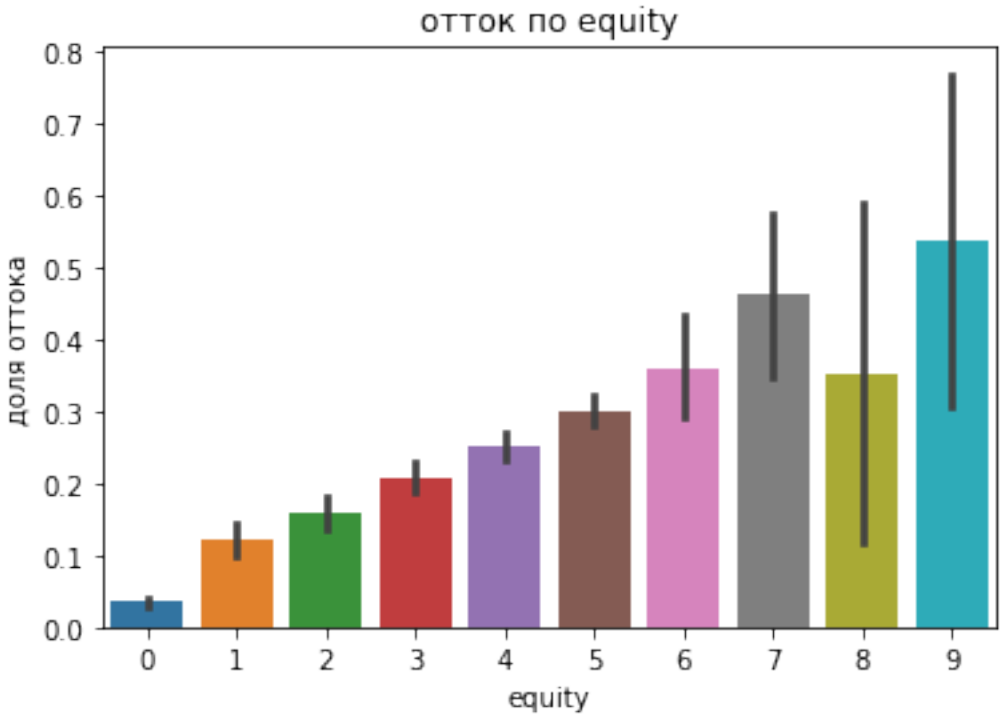
доля уходящих мужчин значительно выше

большая часть показателей в пределах нормы, в том числе возраст, это мы узнали по корреляции

кол-во уходящик увеличивается к средним значениям equity

```
sns.barplot (x=df['equity'], y=df['churn'], data=df)
plt.title('отток по equity')
```

```
plt.xlabel('equity')
plt.ylabel('доля оттока')

Text(0, 0.5, 'доля оттока')
```




```
df.query('products > 2 and churn == 1').groupby('equity')
['user_id'].count()/df.query('products > 2').groupby('equity')
['user_id'].count()*100

equity
0    22.857143
1    25.581395
2    29.801325
3    35.294118
4    39.460784
5    47.587719
6    51.851852
7    57.692308
8    75.000000
9    50.000000
Name: user_id, dtype: float64
```

Проверка гипотез

Метод проверки гипотез

В проверках наших гипотез мы будем использовать T-тест чтобы проверить, отличается ли среднее одной переменной от среднего другой переменной.

Чтобы его провести данные должны отвечать следующим критериям:

1)случайная выборка

2)в каждой выборке нормальное распредление и они больше 30 (Благодаря ЦПТ, если размер выборки составляет хотя бы несколько десятков значений, выборочные средние, которые можно получить из одной и той же генеральной совокупности, будут распределены нормально вокруг истинного среднего этой совокупности)

3)они не зависят друг от друга

4)дисперсии должны быть равны(При этом если выборки достаточно велики (30 и больше значений) и равны по размеру между собой, то такой подход оправдан: симуляции, проведённые учёными, показывают, что даже если дисперсии на самом деле не равны, то тест редко ошибается.)

В нашем случае выборки разного размера, на всякий случай укажем equal_var = False

Условием для применения U-критерия Манна-Уитни является отсутствие в сравниваемых группах совпадающих значений признака (все числа – разные) или очень малое число таких совпадений.

Таким образом, если известно, что выборки имеют нормальное распределение(большие выборки) и равные дисперсии(не обязательно, просто указываем параметр), то t-тест будет более подходящим методом, чем Манн-Уитни тест.

Проверка гипотез

проверить гипотезу 1: "различия возраста между теми клиентами, которые пользуются двумя продуктами банка, и теми, которые пользуются одним."

нулевая гипотеза "возраст клиентов с одним продуктом и двумя продуктами одинаковый"

альтернативная гипотеза "возраст клиентов с одним продуктом и двумя продуктами разный"

Выберем стандартный порог статистической значимости 0.05, нет нужды выбирать меньший.

```
df['age']
```

```
0       25.0
1       37.0
2       30.0
3       51.0
4       34.0
        ...
9995    27.0
9996    46.0
9997    24.0
9998    68.0
9999    58.0
Name: age, Length: 9974, dtype: float64
```

```
alpha = 0.05
optimal_value= df.query('products == 1')['age'].mean()

results = st.ttest_ind(df.query('products == 2')['age'],
df.query('products == 1')['age'], equal_var = False)

print('p-значение:', results.pvalue)

if results.pvalue < alpha:
    print("Отвергаем нулевую гипотезу")
else:
    print("Не получилось отвергнуть нулевую гипотезу")
```

```
p-значение: 0.03299080704232691
Отвергаем нулевую гипотезу
```

проверить гипотезу 2: "Количество потребляемых продуктов банка зависит от переменной equity"

Создадим три категории клиентов: низкий equity(0-2), средний equity(3-5), высокий equity(6-9).

Сделаем три проверки:

нулевая гипотеза 1 "клиенты с низким equity имеют столько же продуктов как и клиенты со средним"

альтернативная гипотеза 1 "клиенты с низким equity имеют меньше продуктов чем клиенты со средним"

нулевая гипотеза 2 "клиенты со средним equity имеют столько же продуктов как и клиенты с высоким"

альтернативная гипотеза 2 "клиенты со средним equity имеют меньше продуктов чем клиенты с высоким"

нулевая гипотеза 3 "клиенты с низким equity имеют столько же продуктов как и клиенты с высоким"

альтернативная гипотеза 3 "клиенты с низким equity имеют меньше продуктов чем клиенты с высоким"

Выберем стандартный порог статистической значимости 0.05 для всех проверок

первая проверка

```
alpha = 0.05

results = st.ttest_ind(df.query('equity > 2 and equity < 6')
['products'], df.query('equity < 3')['products'], equal_var = False)

print('p-значение:', results.pvalue)

if results.pvalue < alpha:
    print("Отвергаем нулевую гипотезу")
else:
    print("Не получилось отвергнуть нулевую гипотезу")
```

```
p-значение: 0.0
Отвергаем нулевую гипотезу
```

вторая проверка

```
alpha = 0.05

results = st.ttest_ind(df.query('equity > 5')['products'],
df.query('equity > 2 and equity < 6')['products'], equal_var = False)

print('p-значение:', results.pvalue)

if results.pvalue < alpha:
    print("Отвергаем нулевую гипотезу")
else:
    print("Не получилось отвергнуть нулевую гипотезу")
```

```
p-значение: 0.0004424057769224473
Отвергаем нулевую гипотезу
```

третья проверка

```
alpha = 0.05

results = st.ttest_ind(df.query('equity > 5')['products'],
df.query('equity < 3')['products'], equal_var = False)

print('p-значение:', results.pvalue)

if results.pvalue < alpha:
    print("Отвергаем нулевую гипотезу")
```

```
else:
    print("Не получилось отвергнуть нулевую гипотезу")
```

```
p-значение: 2.9674218301093537e-32
Отвергаем нулевую гипотезу
```

Действительно кол-во продуктов банка отличается в зависимости от equity

Рекомендации отделу маркетинга

Разделим рекомендации на два раздела, по сегментам / по уходящим клиентам

Мы выделили три сегмента, описывающих всех клиентов банка, основа разделения - кол-во потребляемых продуктов.

Рекомендации основаны на мнении, что лучший клиент имеет несколько продуктов банка(дебет, кредитка, кредит), регулярно ими пользуется и не уходит из банка.

Первый сегмент - клиенты ограниченно пользующиеся услугами банка, в основном у них нет дебетовых карт, лишь кредитные, их equity мало или не подтверждено, у них нет кредитов, однако они практически не отказываются от услуг банка. В этом сегменте 33% клиентов.

Второй сегмент - у этих клиентов уже минимум 2 продукта банка, в 60% случаев один из них кредитка, это костяк всей деятельности банка(51% клентов).

Третий сегмент - практически идеальные клиенты, пользующиеся 3 и более продуктами банка, единственная проблема огромный отток, члены этого сегмента постоянно отказываются от услуг банка. Их всего 16% от общего числа.

Любая рекламная деятельность в своем итоге должна иметь целью создание из клиента первого сегмента, клиента третьего сегмента не подверженного оттоку. Таким образом основой будущих рекламных компаний должны являться клиенты первого сегмента без дебетовых карт, по итогу они должны их завести, тоже самое можно сказать про небольшую часть клиентов без кредиток из перового сегмента.

В то же время клиенты из второго сегмента должны получать предложения о создании кредиток и получении кредитов.

Сегмент один соотношение кредитка/нет (84/16)

Сегмент два соотношение кредитка/нет (60/40)

Клиентов из третьего сегмента следует скорее игнорировать в обширных кампаниях, или предлагать им отсутствующие у них продукты. На них будет направлено второе предложение.

Вторая рекомендация - лояльность

Вышеупомянутые компании будут намного более эффективны в условиях низкого оттока.

Оттоки сегментов 1, 2, 3: 7%, 19%, 39%

На данный момент нас устроит отток в первом сегменте, однако стоит поработать над оставшимися.

Возможная программа лояльности может связать все необходимые для идеального клиента продукты (дебет, кредитка, кредит), благодаря программе можно не только продвигать необходимые продукты, но и способствовать их постоянному использованию (читай низкому оттоку).

Возможность накапливать бонусы, кешбек, льготные предложения помогут продвигать пользователей по сегментам, в то же время снизят отток клиентов.

Подобные меры будут эффективны во втором сегменте, однако не значительно поменяют ситуацию в третьем. Чтобы выявить реальные причины оттока в третьем нужно новое исследование.

Краткий вывод

Существует три сегмента(один продукт, два продутка, больше двух). Необходимо сконцентрировать внимание на рекламе дебетовых карт клиентам первого сегмента, второй сегмент должен получать предложения о создании кредиток и кредитах, третий сегмент менее важен.

Продвигать необходимые продукты может помочь программа лояльности, направленная на борьбу с оттоком во втором и особенно третьем сегменте.