Исследование надежности заемщиков

Во второй части проекта вы выполните шаги 3 и 4. Их вручную проверит ревьюер. Чтобы вам не пришлось писать код заново для шагов 1 и 2, мы добавили авторские решения в ячейки с кодом.

Откройте таблицу и изучите общую информацию о данных

Задание 1. Импортируйте библиотеку pandas. Считайте данные из csv-файла в датафрейм и сохраните в переменную `data`. Путь к файлу:

`/datasets/data.csv`

In [1]:
```
import pandas as pd

try:
    data = pd.read_csv('/datasets/data.csv')
except:
    data = pd.read_csv('https://code.s3.yandex.net/datasets/data.csv')
```

Задание 2. Выведите первые 20 строчек датафрейма `data` на экран.

In [2]:
```
data.head(20)
```

Out[2]:

	children	days_employed	dob_years	education	education_id	family_status	family_status_id	gender	income_type	debt	total_income	purpose
0	1	-8437.673028	42	высшее	0	женат / замужем	0	F	сотрудник	0	253875.639453	покупка жилья
1	1	-4024.803754	36	среднее	1	женат / замужем	0	F	сотрудник	0	112080.014102	приобретение автомобиля
2	0	-5623.422610	33	Среднее	1	женат / замужем	0	M	сотрудник	0	145885.952297	покупка жилья
3	3	-4124.747207	32	среднее	1	женат / замужем	0	M	сотрудник	0	267628.550329	дополнительное образование
4	0	340266.072047	53	среднее	1	гражданский брак	1	F	пенсионер	0	158616.077870	сыграть свадьбу
5	0	-926.185831	27	высшее	0	гражданский брак	1	M	компаньон	0	255763.565419	покупка жилья
6	0	-2879.202052	43	высшее	0	женат / замужем	0	F	компаньон	0	240525.971920	операции с жильем

	children	days_employed	dob_years	education	education_id	family_status	family_status_id	gender	income_type	debt	total_income	purpose
7	0	-152.779569	50	СРЕДНЕЕ	1	женат / замужем	0	M	сотрудник	0	135823.934197	образование
8	2	-6929.865299	35	ВЫСШЕЕ	0	гражданский брак	1	F	сотрудник	0	95856.832424	на проведение свадьбы
9	0	-2188.756445	41	среднее	1	женат / замужем	0	M	сотрудник	0	144425.938277	покупка жилья для семьи
10	2	-4171.483647	36	высшее	0	женат / замужем	0	M	компаньон	0	113943.491460	покупка недвижимости
11	0	-792.701887	40	среднее	1	женат / замужем	0	F	сотрудник	0	77069.234271	покупка коммерческой недвижимости
12	0	NaN	65	среднее	1	гражданский брак	1	M	пенсионер	0	NaN	сыграть свадьбу
13	0	-1846.641941	54	неоконченное высшее	2	женат / замужем	0	F	сотрудник	0	130458.228857	приобретение автомобиля
14	0	-1844.956182	56	высшее	0	гражданский брак	1	F	компаньон	1	165127.911772	покупка жилой недвижимости
15	1	-972.364419	26	среднее	1	женат / замужем	0	F	сотрудник	0	116820.904450	строительство собственной недвижимости
16	0	-1719.934226	35	среднее	1	женат / замужем	0	F	сотрудник	0	289202.704229	недвижимость
17	0	-2369.999720	33	высшее	0	гражданский брак	1	M	сотрудник	0	90410.586745	строительство недвижимости
18	0	400281.136913	53	среднее	1	вдовец / вдова	2	F	пенсионер	0	56823.777243	на покупку подержанного автомобиля
19	0	-10038.818549	48	СРЕДНЕЕ	1	в разводе	3	F	сотрудник	0	242831.107982	на покупку своего автомобиля

Задание 3. Выведите основную информацию о датафрейме с помощью метода `info()`.

In [3]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21525 entries, 0 to 21524
Data columns (total 12 columns):
```

```
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   children          21525 non-null  int64
 1   days_employed     19351 non-null  float64
 2   dob_years         21525 non-null  int64
 3   education         21525 non-null  object
 4   education_id      21525 non-null  int64
 5   family_status     21525 non-null  object
 6   family_status_id  21525 non-null  int64
 7   gender            21525 non-null  object
 8   income_type       21525 non-null  object
 9   debt              21525 non-null  int64
 10  total_income      19351 non-null  float64
 11  purpose           21525 non-null  object
dtypes: float64(2), int64(5), object(5)
memory usage: 2.0+ MB
```

Предобработка данных

Удаление пропусков

Задание 4. Выведите количество пропущенных значений для каждого столбца. Используйте комбинацию двух методов.

In [4]:
```
data.isna().sum()
```

Out[4]:
```
children               0
days_employed       2174
dob_years              0
education              0
education_id           0
family_status          0
family_status_id       0
gender                 0
income_type            0
debt                   0
total_income        2174
purpose                0
dtype: int64
```

Задание 5. В двух столбцах есть пропущенные значения. Один из них — `days_employed`. Пропуски в этом столбце вы обработаете на следующем этапе. Другой столбец с пропущенными значениями — `total_income` — хранит данные о доходах. На сумму дохода сильнее всего влияет тип занятости, поэтому заполнить пропуски в этом столбце нужно медианным значением по каждому типу из столбца `income_type`. Например, у человека с типом занятости `сотрудник` пропуск в столбце `total_income` должен быть заполнен медианным доходом среди всех записей с тем же типом.

In [5]:
```
for t in data['income_type'].unique():
    data.loc[(data['income_type'] == t) & (data['total_income'].isna()), 'total_income'] = \
```

```
    data.loc[(data['income_type'] == t), 'total_income'].median()
```

Обработка аномальных значений

Задание 6. В данных могут встречаться артефакты (аномалии) — значения, которые не отражают действительность и появились по какой-то ошибке. таким артефактом будет отрицательное количество дней трудового стажа в столбце `days_employed`. Для реальных данных это нормально. Обработайте значения в этом столбце: замените все отрицательные значения положительными с помощью метода `abs()`.

In [6]:
```
data['days_employed'] = data['days_employed'].abs()
```

Задание 7. Для каждого типа занятости выведите медианное значение трудового стажа `days_employed` в днях.

In [7]:
```
data.groupby('income_type')['days_employed'].agg('median')
```

Out[7]:
```
income_type
безработный        366413.652744
в декрете            3296.759962
госслужащий          2689.368353
компаньон            1547.382223
пенсионер          365213.306266
предприниматель       520.848083
сотрудник            1574.202821
студент               578.751554
Name: days_employed, dtype: float64
```

У двух типов (безработные и пенсионеры) получатся аномально большие значения. Исправить такие значения сложно, поэтому оставьте их как есть.

Задание 8. Выведите перечень уникальных значений столбца `children`.

In [8]:
```
data['children'].unique()
```

Out[8]:
```
array([ 1,  0,  3,  2, -1,  4, 20,  5])
```

Задание 9. В столбце `children` есть два аномальных значения. Удалите строки, в которых встречаются такие аномальные значения из датафрейма `data`.

In [9]:
```
data = data[(data['children'] != -1) & (data['children'] != 20)]
```

Задание 10. Ещё раз выведите перечень уникальных значений столбца `children`, чтобы убедиться, что артефакты удалены.

In [10]:
```
data['children'].unique()
```

Out[10]:
```
array([1, 0, 3, 2, 4, 5])
```

Удаление пропусков (продолжение)

Задание 11. Заполните пропуски в столбце `days_employed` медианными значениями по каждого типа занятости `income_type`.

In [11]:
```
for t in data['income_type'].unique():
    data.loc[(data['income_type'] == t) & (data['days_employed'].isna()), 'days_employed'] = \
    data.loc[(data['income_type'] == t), 'days_employed'].median()
```

Задание 12. Убедитесь, что все пропуски заполнены. Проверьте себя и ещё раз выведите количество пропущенных значений для каждого столбца с помощью двух методов.

In [12]:
```
data.isna().sum()
```

Out[12]:
```
children            0
days_employed       0
dob_years           0
education           0
education_id        0
family_status       0
family_status_id    0
gender              0
income_type         0
debt                0
total_income        0
purpose             0
dtype: int64
```

Изменение типов данных

Задание 13. Замените вещественный тип данных в столбце `total_income` на целочисленный с помощью метода `astype()`.

In [13]:
```
data['total_income'] = data['total_income'].astype(int)
```

Обработка дубликатов

Задание 14. Обработайте неявные дубликаты в столбце `education`. В этом столбце есть одни и те же значения, но записанные по-разному: с использованием заглавных и строчных букв. Приведите их к нижнему регистру.

In [14]:
```
data['education'] = data['education'].str.lower()
```

Задание 15. Выведите на экран количество строк-дубликатов в данных. Если такие строки присутствуют, удалите их.

In [15]:
```
data.duplicated().sum()
```

Out[15]: 71

In [16]:
```
data = data.drop_duplicates()
```

Категоризация данных

Задание 16. На основании диапазонов, указанных ниже, создайте в датафрейме `data` столбец `total_income_category` с категориями:

- 0–30000 — `'E'`;
- 30001–50000 — `'D'`;
- 50001–200000 — `'C'`;
- 200001–1000000 — `'B'`;
- 1000001 и выше — `'A'`.

Например, кредитополучателю с доходом 25000 нужно назначить категорию `'E'`, а клиенту, получающему 235000, — `'B'`. Используйте собственную функцию с именем `categorize_income()` и метод `apply()`.

In [17]:
```
def categorize_income(income):
    try:
        if 0 <= income <= 30000:
            return 'E'
        elif 30001 <= income <= 50000:
            return 'D'
        elif 50001 <= income <= 200000:
            return 'C'
        elif 200001 <= income <= 1000000:
            return 'B'
        elif income >= 1000001:
            return 'A'
    except:
        pass
```

In [18]:
```
data['total_income_category'] = data['total_income'].apply(categorize_income)
```

Задание 17. Выведите на экран перечень уникальных целей взятия кредита из столбца `purpose`.

In [19]:
```
data['purpose'].unique()
```

Out[19]:
```
array(['покупка жилья', 'приобретение автомобиля',
       'дополнительное образование', 'сыграть свадьбу',
       'операции с жильем', 'образование', 'на проведение свадьбы',
       'покупка жилья для семьи', 'покупка недвижимости',
       'покупка коммерческой недвижимости', 'покупка жилой недвижимости',
       'строительство собственной недвижимости', 'недвижимость',
       'строительство недвижимости', 'на покупку подержанного автомобиля',
       'на покупку своего автомобиля',
       'операции с коммерческой недвижимостью',
       'строительство жилой недвижимости', 'жилье',
       'операции со своей недвижимостью', 'автомобили',
       'заняться образованием', 'сделка с подержанным автомобилем',
       'получение образования', 'автомобиль', 'свадьба',
       'получение дополнительного образования', 'покупка своего жилья',
       'операции с недвижимостью', 'получение высшего образования',
       'свой автомобиль', 'сделка с автомобилем',
       'профильное образование', 'высшее образование',
       'покупка жилья для сдачи', 'на покупку автомобиля', 'ремонт жилью',
       'заняться высшим образованием'], dtype=object)
```

Задание 18. Создайте функцию, которая на основании данных из столбца `purpose` сформирует новый столбец `purpose_category`, в который войдут следующие категории:

- `'операции с автомобилем'`,
- `'операции с недвижимостью'`,
- `'проведение свадьбы'`,
- `'получение образования'`.

Например, если в столбце `purpose` находится подстрока `'на покупку автомобиля'`, то в столбце `purpose_category` должна появиться строка `'операции с автомобилем'`.

Используйте собственную функцию с именем `categorize_purpose()` и метод `apply()`. Изучите данные в столбце `purpose` и определите, какие подстроки помогут вам правильно определить категорию.

In [20]:
```
def categorize_purpose(row):
    try:
        if 'автом' in row:
            return 'операции с автомобилем'
        elif 'жил' in row or 'недвиж' in row:
            return 'операции с недвижимостью'
        elif 'свад' in row:
            return 'проведение свадьбы'
        elif 'образов' in row:
            return 'получение образования'
    except:
        return 'нет категории'
```

In [21]:
```
data['purpose_category'] = data['purpose'].apply(categorize_purpose)
```

Шаг 3. Исследуйте данные и ответьте на вопросы

3.1 Есть ли зависимость между количеством детей и возвратом кредита в срок?

Для того, чтобы ответить на вопрос возьмём данные из колонки 'debt' (она показывает была ли у клиента задолженность по кредитам) и сгруппируем их по колонке 'children' (которая содержит данные о количестве детей). После чего посчитаем средние значения по группам.

In [22]:
```
data.groupby('children')['debt'].mean()
```

Out[22]:
```
children
0    0.075438
1    0.092346
2    0.094542
3    0.081818
4    0.097561
5    0.000000
Name: debt, dtype: float64
```

Мы видим зависимость, но она не является линейной.

Посчитаем количество клиентов по группам, чтобы понимать, насколько эти группы представлены и можем ли мы считать показатели значимыми.

In [23]:
```
data['children'].value_counts()
```

Out[23]:
```
0    14091
1     4808
2     2052
3      330
4       41
5        9
Name: children, dtype: int64
```

?Мы наблюдаем? заметный скачок в вероятности наличия задолженности при появления первого ребёнка и затем плавный рост при увеличении количества детей. Однако среди клиентов с тремя детьми мы можем наблюдать резкое падение. Всего в таблице приведены данные о более чем 20000 клиентов и, среди них, имеют 3 детей 330 человек. Учитывая это, не стоит списывать такой результат на случайность, это может иметь причину.

Группы клиентов с 4 и 5 детьми ещё меньше представлены в нашей выборке, поэтому основывать на них наши выводы будет не слишком обоснованно.

4 ребёнка: хотя статистика по этой группе подходит под тенденцию роста доли должников, окажись в ней другие настолько же случайные люди, она могла бы показать обратный рост или стагнацию.

5 детей: в данной группе всего 9 строк, поэтому то, что у таких клиентов не оказалось задолженностей, не стоит брать во внимание, так как в целом по выборке вероятность, что у клиента была задолженность меньше 10%.

In [24]:
```
data.groupby(['children','family_status'])['debt'].mean()
data.groupby('family_status')['debt'].mean()
data.groupby('children')['family_status'].value_counts()
data.groupby('children')['income_type'].value_counts()
```

Out[24]:
```
children  income_type
0         сотрудник          6574
          пенсионер          3510
          компаньон          3137
          госслужащий         866
          предприниматель       2
          безработный           1
          студент               1
1         сотрудник          2880
          компаньон          1298
          госслужащий         354
          пенсионер           275
          безработный           1
2         сотрудник          1315
          компаньон           529
          госслужащий         187
          пенсионер            20
          в декрете             1
3         сотрудник           209
          компаньон            79
          госслужащий          36
          пенсионер             6
4         сотрудник            31
          госслужащий           7
          компаньон             2
          пенсионер             1
5         сотрудник             6
          компаньон             2
          госслужащий           1
Name: income_type, dtype: int64
```

Здесь я провёл несколько дополнительных действий, чтобы проверить, не связаны ли аномалии с нетипичными показателями в других столбцах

Вывод: количество детей не имеет прямой зависимости с возвратом кредита в срок, но само наличие детей снижает такую вероятность. Дополнительно можно отметить, что среди клиентов с тремя детьми процент имевших задолженности значительно ниже, чем у клиентов с другим количеством детей.

3.2 Есть ли зависимость между семейным положением и возвратом кредита в срок?

Посчитаем средние значения для каждой группы

In [25]:
```
data.groupby('family_status')['debt'].mean()
```

Out[25]:
```
family_status
Не женат / не замужем    0.097639
в разводе                0.070648
вдовец / вдова           0.066246
гражданский брак         0.093130
женат / замужем          0.075606
Name: debt, dtype: float64
```

Наименее надежны одинокие и состоящие в гражданском брак

Проверим все ли группы достаточно представлены в выборке (спойлер: да, все)

In [26]:
```
data['family_status'].value_counts()
```

Out[26]:
```
женат / замужем          12261
гражданский брак          4134
Не женат / не замужем     2796
в разводе                 1189
вдовец / вдова             951
Name: family_status, dtype: int64
```

Может возраст оказывает существенное влияние на надёжность клиентов?

Посмотрим, каков средний возраст клиентов, сгруппированных по семейному положению.

In [33]:
```
data.groupby('family_status')['dob_years'].mean()
```

Out[33]:
```
family_status
Не женат / не замужем    38.368026
в разводе                45.561817
вдовец / вдова           56.501577
гражданский брак         42.067731
женат / замужем          43.558519
Name: dob_years, dtype: float64
```

Самой "молодой" оказалась группа одиноких. Гражданский брак и женатые идут почти вровень вторыми и третьими. Затем с небольшим отрывом те, кто в разводе.

Получается, что семейное положение действительно влияет, и списать всё на возраст не получится. Но давайте всё же посмотрим, что там с возрастом.

Посчитаем среднюю вероятность оказаться в списке должников, сгруппированную по возрасту.

In [38]:
```
data.groupby(['dob_years']).agg(
    perc_of_debtors=('debt', 'mean'),
```

```
    n_of_debtors=('debt', 'count')
)
```

Out[38]:

dob_years	perc_of_debtors	n_of_debtors
0	0.080000	100
19	0.071429	14
20	0.078431	51
21	0.127273	110
22	0.136612	183
23	0.080000	250
24	0.091255	263
25	0.120787	356
26	0.115764	406
27	0.100000	490
28	0.113772	501
29	0.119926	542
30	0.101313	533
31	0.136937	555
32	0.100990	505
33	0.095321	577
34	0.105882	595
35	0.076672	613
36	0.077899	552
37	0.096226	530
38	0.107744	594
39	0.077058	571
40	0.078203	601
41	0.081531	601

dob_years	perc_of_debtors	n_of_debtors
42	0.071066	591
43	0.086444	509
44	0.075786	541
45	0.079108	493
46	0.081545	466
47	0.077568	477
48	0.061798	534
49	0.061386	505
50	0.090551	508
51	0.054054	444
52	0.057971	483
53	0.076586	457
54	0.065539	473
55	0.052154	441
56	0.066946	478
57	0.066225	453
58	0.052863	454
59	0.068182	440
60	0.058981	373
61	0.028409	352
62	0.051873	347
63	0.044776	268
64	0.046512	258
65	0.072539	193
66	0.043956	182
67	0.053892	167

dob_years	perc_of_debtors	n_of_debtors
68	0.080808	99
69	0.048193	83
70	0.046154	65
71	0.017857	56
72	0.060606	33
73	0.000000	8
74	0.000000	6
75	0.000000	1

Вероятность для клиентов младше 35 почти повсеместно выше 10%. От 35 и старше наблюдается заметное снижение 7,5% и ниже для большей части выборки.

Тут лучше сделать 2 группы и посчитать среднее

Теперь проверим, насколько много людей в этих возрастных группах и убедимся, что результат подходит для нашего исследования.

In [29]:
```
data['dob_years'].value_counts().sort_index()
```

Out[29]:
```
0     100
19     14
20     51
21    110
22    183
23    250
24    263
25    356
26    406
27    490
28    501
29    542
30    533
31    555
32    505
33    577
34    595
35    613
36    552
37    530
38    594
39    571
40    601
41    601
```

```
42    591
43    509
44    541
45    493
46    466
47    477
48    534
49    505
50    508
51    444
52    483
53    457
54    473
55    441
56    478
57    453
58    454
59    440
60    373
61    352
62    347
63    268
64    258
65    193
66    182
67    167
68     99
69     83
70     65
71     56
72     33
73      8
74      6
75      1
Name: dob_years, dtype: int64
```

Получается, что семейное положение действительно влияет на вероятность возврата кредит в срок. Те, кто вступает в официальный брак, оказываются более надёжными даже после развода или гибели супруга.

При этом мы выяснили, что возраст также является важным фактором.

Вывод: да, семейное положение влияет. Женатые, разведённые и вдовцы возвращают кредит в срок чаще, чем одинокие и состоящие в гражданском браке.

3.3 Есть ли зависимость между уровнем дохода и возвратом кредита в срок?

Посчитаем среднее для клиентов по ранее созданным группам и отсортируем их по вероятности возврата кредита в срок.

In [40]:
```
data.groupby('total_income_category')['debt'].mean().sort_values()
```

Out[40]:
```
total_income_category
D    0.060172
B    0.070602
A    0.080000
C    0.084982
E    0.090909
Name: debt, dtype: float64
```

Линейной зависимости нет, но показатели различаются.

In [31]:
```
data.groupby('total_income_category')['income_type'].value_counts()
```

Out[31]:
```
total_income_category  income_type
A                      компаньон            15
                       сотрудник            10
B                      сотрудник          2360
                       компаньон          1733
                       пенсионер           552
                       госслужащий         366
                       предприниматель       2
                       безработный           1
C                      сотрудник          8504
                       компаньон          3273
                       пенсионер          3081
                       госслужащий        1060
                       безработный           1
                       в декрете             1
                       студент               1
D                      пенсионер           164
                       сотрудник           136
                       компаньон            25
                       госслужащий          24
E                      пенсионер            15
                       сотрудник             5
                       госслужащий           1
                       компаньон             1
Name: income_type, dtype: int64
```

Вывод: зависимость есть, но она не линейная. Лучше всего в возвратом кредита в срок справляются клиенты из категории 'E' (до 30 т.р.)

3.4 Как разные цели кредита влияют на его возврат в срок?

Посчитаем среднее для клиентов по цели кредита.

In [41]:
```
data.groupby('purpose_category')['debt'].mean().sort_values()
```

Out[41]:
```
purpose_category
операции с недвижимостью    0.072551
проведение свадьбы          0.079118
получение образования       0.092528
```

```
операции с автомобилем      0.093480
Name: debt, dtype: float64
```

Вывод: лучше всего возвращают в срок кредиты связанные с *недвижимостью*, затем *свадьбы* и в конце *автомобили* и *образование*

3.5 Приведите возможные причины появления пропусков в исходных данных.

Ответ: пропуски встречаются в столбцах с данными об уровне дохода и трудовом стаже. При этом количество пропусков одинаковое и приходится на одни и те же строки. Возможно это связано с тем, что информация об этих клиентах взята из источника, где эти данные не были актуальны. Также может быть, что данные защищены законом и банк не может требовать их указать.

3.6 Объясните, почему заполнить пропуски медианным значением — лучшее решение для количественных переменных.

Ответ: потому что данный метод показывает наиболее типичные результаты, на которые не влияют возможные выбросы

Шаг 4: общий вывод.

Коротко о том, что сделали и какие самые ключевые итоги

In [ ]: