**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.

**1  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

Ввод [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.**

Ввод [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 |
| **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().**

Ввод [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

**2  Предобработка данных**

**2.1  Удаление пропусков**

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

Ввод [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.**

Ввод [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()

**2.2  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().**

Ввод [6]:

data['days\_employed'] **=** data['days\_employed'].abs()

**Задание 7. Для каждого типа занятости выведите медианное значение трудового стажа days\_employed в днях.**

Ввод [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.**

Ввод [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.**

Ввод [9]:

data **=** data[(data['children'] **!=** **-**1) **&** (data['children'] **!=** 20)]

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

Ввод [10]:

data['children'].unique()

Out[10]:

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

**2.3  Удаление пропусков (продолжение)**

**Задание 11. Заполните пропуски в столбце days\_employed медианными значениями по каждого типа занятости income\_type.**

Ввод [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.**

Ввод [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

**2.4  Изменение типов данных**

**Задание 13. Замените вещественный тип данных в столбце total\_income на целочисленный с помощью метода astype().**

Ввод [13]:

data['total\_income'] **=** data['total\_income'].astype(int)

**2.5  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.**

Ввод [14]:

data['education'] **=** data['education'].str.lower()

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

Ввод [15]:

data.duplicated().sum()

Out[15]:

71

Ввод [16]:

data **=** data.drop\_duplicates()

**2.6  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().**

Ввод [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**

Ввод [18]:

data['total\_income\_category'] **=** data['total\_income'].apply(categorize\_income)

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

Ввод [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.**

Ввод [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'

Ввод [21]:

data['purpose\_category'] **=** data['purpose'].apply(categorize\_purpose)

**2.7  Шаг 3. Исследуйте данные и ответьте на вопросы**

**2.7.1  3.1 Есть ли зависимость между количеством детей и возвратом кредита в срок?**

Ввод [22]:

children\_debt**=**data[['children','debt']]

pivot\_table**=**children\_debt.pivot\_table(index**=**'children', values**=**'debt', aggfunc**=**['count', 'sum', 'mean'])

pivot\_table**=**pivot\_table.rename(columns**=**{'count':'Всего кредитополучателей', 'sum':'Всего должников','mean': 'Доля должников'})

display(pivot\_table)

|  | **Всего кредитополучателей** | **Всего должников** | **Доля должников** |
| --- | --- | --- | --- |
|  | **debt** | **debt** | **debt** |
| **children** |  |  |  |
| **0** | 14091 | 1063 | 0.075438 |
| **1** | 4808 | 444 | 0.092346 |
| **2** | 2052 | 194 | 0.094542 |
| **3** | 330 | 27 | 0.081818 |
| **4** | 41 | 4 | 0.097561 |
| **5** | 9 | 0 | 0.000000 |

**Вывод:** Заемщики без детей допускаю меньше просрочек по кредитным обязательствам. Так же очевидно, что данных по многодетным заемщикам недостаточно для корректным выводам по ним.

**2.7.2  3.2 Есть ли зависимость между семейным положением и возвратом кредита в срок?**

Ввод [23]:

family\_status\_debt**=**data[['family\_status','debt']]

pivot\_table**=**family\_status\_debt.pivot\_table(index**=**'family\_status', values**=**'debt', aggfunc**=**['count', 'sum', 'mean'])

pivot\_table**=**pivot\_table.rename(columns**=**{'count':'Всего кредитополучателей', 'sum':'Всего должников','mean': 'Доля должников'})

display(pivot\_table)

|  | **Всего кредитополучателей** | **Всего должников** | **Доля должников** |
| --- | --- | --- | --- |
|  | **debt** | **debt** | **debt** |
| **family\_status** |  |  |  |
| **Не женат / не замужем** | 2796 | 273 | 0.097639 |
| **в разводе** | 1189 | 84 | 0.070648 |
| **вдовец / вдова** | 951 | 63 | 0.066246 |
| **гражданский брак** | 4134 | 385 | 0.093130 |
| **женат / замужем** | 12261 | 927 | 0.075606 |

**Вывод:** Заемщики, не состоящие в официальном/не официальном браке, имеют больше просрочек.

**2.7.3  3.3 Есть ли зависимость между уровнем дохода и возвратом кредита в срок?**

Ввод [24]:

total\_income\_debt**=**data[['total\_income\_category','debt']]

pivot\_table**=**total\_income\_debt.pivot\_table(index**=**'total\_income\_category', values**=**'debt', aggfunc**=**['count', 'sum', 'mean'])

pivot\_table**=**pivot\_table.rename(columns**=**{'count':'Всего кредитополучателей', 'sum':'Всего должников','mean': 'Доля должников'})

display(pivot\_table)

|  | **Всего кредитополучателей** | **Всего должников** | **Доля должников** |
| --- | --- | --- | --- |
|  | **debt** | **debt** | **debt** |
| **total\_income\_category** |  |  |  |
| **A** | 25 | 2 | 0.080000 |
| **B** | 5014 | 354 | 0.070602 |
| **C** | 15921 | 1353 | 0.084982 |
| **D** | 349 | 21 | 0.060172 |
| **E** | 22 | 2 | 0.090909 |

**Вывод:** Оценивая платежную дисциплину по самым крупным категориям - B и С - можно сделать вывод о том, что заемщики с бОльшим доходом допускают меньшее количество просрочек.

**2.7.4  3.4 Как разные цели кредита влияют на его возврат в срок?**

Ввод [25]:

purpose\_category\_debt**=**data[['purpose\_category','debt']]

pivot\_table**=**purpose\_category\_debt.pivot\_table(index**=**'purpose\_category', values**=**'debt', aggfunc**=**['count', 'sum', 'mean'])

pivot\_table**=**pivot\_table.rename(columns**=**{'count':'Всего кредитополучателей', 'sum':'Всего должников','mean': 'Доля должников'})

display(pivot\_table)

|  | **Всего кредитополучателей** | **Всего должников** | **Доля должников** |
| --- | --- | --- | --- |
|  | **debt** | **debt** | **debt** |
| **purpose\_category** |  |  |  |
| **операции с автомобилем** | 4279 | 400 | 0.093480 |
| **операции с недвижимостью** | 10751 | 780 | 0.072551 |
| **получение образования** | 3988 | 369 | 0.092528 |
| **проведение свадьбы** | 2313 | 183 | 0.079118 |

**Вывод:** Реже всего просрочки по выплатам допускают по кредитам на цели приобретения недвижимости.

**2.7.5  3.5 Приведите возможные причины появления пропусков в исходных данных.**

*Ответ:* Часто данные теряются при копировании/пересылке, так же пропуски являются следствием невнимательности людей, которые данные вносили, либо пользователь могли не вносить свои данные намеренно.

**2.7.6  3.6 Объясните, почему заполнить пропуски медианным значением — лучшее решение для количественных переменных.**

*Ответ:* Медианные значения при анализе данных не искажают их так сильно как средние значения (среднее значение некорректно характеризует данные, когда некоторые значения сильно выделяются среди большинства).

**2.8  Шаг 4: общий вывод.**

В ходе работы на данными были: -получены и изучены данные; -определены и заполнены пропущенные и аномальные значения; -заменен вещественный тип данных на целочисленный; -выявлены и удалены дубликаты; -данные категоризированы; -даны ответы на вопросы "Есть ли зависимость между наличием детей и возвратом кредита в срок?", "Есть ли зависимость между семейным положением и возвратом кредита в срок?", "Есть ли зависимость между уровнем дохода и возвратом кредита в срок?", "Как разные цели кредита влияют на его возврат в срок?" -сделаны выводы на каждом этапе и общий вывод приведу ниже.

Отвечая на поставленный вопрос "Нужно разобраться, влияет ли семейное положение и количество детей клиента на факт погашения кредита в срок", могу сказать, что есть прямая зависимость между этими условиями:

-заемщики, не имеющие партнера, но имеющие 1-4 детей, являются самыми недобросовестными плательщиками; -заемщики с бОльшими доходами допускают меньшее количество просрочек.

Таким образом, ответственным заемщиком будет являться клиент с доходом от 50000р., состоящий в отношениях, не имеющий детей, нуждающийся в средствах на приобретение недвижимости.

Рекоммендую увеличить глубину проверки по скорингу заемшиков не находящихся в официальном браке/отношениях, имеющими 1-4 детей, имеющими целью крудитования покупку автомобиля или получение образования (как для наименее ответственной категории клиентов).