Исследование надёжности заёмщиков

Заказчик — кредитный отдел банка. Нужно разобраться, влияет ли семейное положение и количество детей клиента на факт погашения кредита в срок. Входные данные от банка — статистика о платёжеспособности клиентов.

Результаты исследования будут учтены при построении модели **кредитного скоринга** — специальной системы, которая оценивает способность потенциального заёмщика вернуть кредит банку.

Описание данных

children — количество детей в семье

days_employed — общий трудовой стаж в днях

dob_years — возраст клиента в годах

education — уровень образования клиента

education_id — идентификатор уровня образования

family_status — семейное положение

family_status_id — идентификатор семейного положения

gender — пол клиента

income_type — тип занятости

debt — имел ли задолженность по возврату кредитов

total_income — ежемесячный доход

purpose — цель получения кредита

Шаг 1. Просмотр данных

In [1]:
```
import pandas as pd
data= pd.read_csv('/datasets/data.csv') #открываю файл с данными
print(data.info()) # изучаю общую информацию по дата фрейму
print(data.shape)
data.tail(100)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21525 entries, 0 to 21524
Data columns (total 12 columns):
children            21525 non-null int64
days_employed       19351 non-null float64
dob_years           21525 non-null int64
education           21525 non-null object
education_id        21525 non-null int64
family_status       21525 non-null object
family_status_id    21525 non-null int64
gender              21525 non-null object
income_type         21525 non-null object
debt                21525 non-null int64
total_income        19351 non-null float64
purpose             21525 non-null object
dtypes: float64(2), int64(5), object(5)
memory usage: 2.0+ MB
None
(21525, 12)
```

Out[1]:

	children	days_employed	dob_years	education	education_id	family_status	family_status
21425	0	-106.691955	34	среднее	1	женат / замужем	
21426	0	NaN	49	среднее	1	женат / замужем	
21427	2	-1050.799939	42	среднее	1	женат / замужем	
21428	1	-255.663311	38	высшее	0	гражданский брак	
21429	0	-7030.056218	66	ВЫСШЕЕ	0	вдовец / вдова	
...	...	...	...	...	...	...	
21520	1	-4529.316663	43	среднее	1	гражданский брак	
21521	0	343937.404131	67	среднее	1	женат / замужем	
21522	1	-2113.346888	38	среднее	1	гражданский брак	
21523	3	-3112.481705	38	среднее	1	женат / замужем	
21524	2	-1984.507589	40	среднее	1	женат / замужем	

100 rows × 12 columns

Вывод

Много пропусков и нолей в столбцах 'days_employed','total_income' , регистр разный в строках, ошибки в данных, например, отрицательный стаж, отрицательное количество детей.

Шаг 2. Предобработка данных

Обработка пропусков

In [2]:
```
data['children']=data['children'].abs() #привела все отрицательные значения по
детям в положительное значение, так как возможно, данная ошибка вызвана челове
ческим фактором
data['days_employed']=data['days_employed'].abs() # тоже самое сделала со стаж
ем, так как отрицательными данные значения быть не могут. и вероятнее всего да
нные внесены с ошибкой
print("Количество пропусков:") #считаю все пропуски
print(data.isnull().sum())
#основные пропуски содержатся в столбцах "days_employed" и "total_income"
#почитаем медиану для данных столбцов, так как при разнице зп в 110 раз, медиа
на лучше характеризует

days_employed_median=data['days_employed'].median() # нашла медиану
total_income_median=data['total_income'].median()

data['days_employed'].fillna(days_employed_median, inplace=True) # подставила
 медианное значение в пропуски
data['total_income'].fillna(total_income_median, inplace=True)
print("Количество пропусков после проверки:")
print(data.isnull().sum()) #проверяем, остались ли пропуски
#пропусков нет
```

```
Количество пропусков:
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
Количество пропусков после проверки:
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
```

Вывод

Основные пропуски содержатся в столбцах "days_employed" и "total_income", по 2174 пропусков в каждом столбце. Пропуски заполнила медианой по столбцу, так как разница зп в 110 раз, медиана в данном случае больше подходит, чем среднее значение.

Замена типа данных

In [3]:
```
data['days_employed']=data['days_employed'].astype('int')
data['total_income']=data['total_income'].astype('int')
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21525 entries, 0 to 21524
Data columns (total 12 columns):
children            21525 non-null int64
days_employed       21525 non-null int64
dob_years           21525 non-null int64
education           21525 non-null object
education_id        21525 non-null int64
family_status       21525 non-null object
family_status_id    21525 non-null int64
gender              21525 non-null object
income_type         21525 non-null object
debt                21525 non-null int64
total_income        21525 non-null int64
purpose             21525 non-null object
dtypes: int64(7), object(5)
memory usage: 2.0+ MB
```

Вывод

Для перевода из дробного числа в целочисленное я использовала метод astype(). Исходя из общей информации о таблице, мы видим, что два столбца имеют вещественный тип данных, который и нужно было заменить на целочисленный. Также можно использовать метод to_numeric().

Обработка дубликатов

In [4]:
```
data['education']=data['education'].str.lower() #приводим все строки таблицы к нижнему регистру, чтобы избежать выявить и убрать все дубликаты
data['family_status']=data['family_status'].str.lower()
data['gender']=data['gender'].str.lower()
data['income_type']=data['income_type'].str.lower()
data['purpose']=data['purpose'].str.lower()
print("Количество дубликатов до удаления", data.duplicated().sum())
data = data.drop_duplicates().reset_index(drop=True) # возможно, появление дубликатов связано с ошибкой выгрузки
print("Количество дубликатов после проверки", data.duplicated().sum()) #проверяю есть ли дубликаты
```

```
Количество дубликатов до удаления 71
Количество дубликатов после проверки 0
```

Вывод

Дубликаты вызывают смещение наших финальных оценок. Именно поэтому их и надо удалять. Воспользовавшись приведением данных таблицы к нижнему регистру, было выявлено около 20 дубликатов. избавилась от них методом drop_duplicates() и переустановила индексы.

Лемматизация

In [5]:
```
from pymystem3 import Mystem #импортирую библиотеку pymystem3 для лемматизации
m = Mystem()
lemmas=[]
for row in data['purpose']: #циклом прохожусь по каждому слову в ячейках столбца 'purpose'
    word=''.join(m.lemmatize(row))
    lemmas.append(word)

from collections import Counter
print(Counter(lemmas))  # включаю счетчик уникальных лемм, вывожу список уникальных целей кредита с их количеством
data['purpose_lemmas']=pd.DataFrame(lemmas, columns=['purpose_lemmas'])  #делаю дата фрейм из списка лемм, добавляю столбец 'purpose_lemmas' в главную таблицу, куда попадают все леммы столбца 'purpose'
```

```
Counter({'автомобиль\n': 972, 'свадьба\n': 791, 'на проведение свадьба\n': 768, 'сыграть свадьба\n': 765, 'операция с недвижимость\n': 675, 'покупка коммерческий недвижимость\n': 661, 'операция с жилье\n': 652, 'покупка жилье для сдача\n': 651, 'операция с коммерческий недвижимость\n': 650, 'покупка жилье \n': 646, 'жилье\n': 646, 'покупка жилье для семья\n': 638, 'строительство собственный недвижимость\n': 635, 'недвижимость\n': 633, 'операция со свой недвижимость\n': 627, 'строительство жилой недвижимость\n': 624, 'покупка недвижимость\n': 621, 'покупка свой жилье\n': 620, 'строительство недвижимость\n': 619, 'ремонт жилье\n': 607, 'покупка жилой недвижимость\n': 606, 'на покупка свой автомобиль\n': 505, 'заниматься высокий образование\n': 496, 'сделка с подержанный автомобиль\n': 486, 'на покупка подержать автомобиль\n': 478, 'свой автомобиль\n': 478, 'на покупка автомобиль\n': 471, 'приобретение автомобиль\n': 461, 'дополнительный образование\n': 460, 'сделка с автомобиль\n': 455, 'высокий образование\n': 452, 'образование\n': 447, 'получение дополнительный образование\n': 446, 'получение образование\n': 442, 'профильный образование\n': 436, 'получение высокий образование\n': 426, 'заниматься образование\n': 408})
```

Вывод

Проанализировав основные леммы целей кредита, можно выделить 5 основых категорий целей кредита: 'автокредит', 'кредит на коммерческую недвижимость', 'ипотека', 'кредит на свадьбу', 'кредит на образование'.

Категоризация данных

In [6]:
```
def purpose_group (row): # функция для категоризации целей кредита
    try:
        if "автомобиль" in row:
            return ('автокредит')
        elif "коммерческий недвижимость" in row:
            return ('кредит на коммерческую недвижимость')
        elif "недвижимость" in row:
            return ('ипотека')
        elif "жилье" in row:
            return ('ипотека')
        elif "свадьба" in row:
            return ('кредит на свадьбу')
        elif "образование" in row:
            return ('кредит на образование')
        return 'другое'
    except:
        print('проверь код, есть какая-то ошибка')
data['purpose_group']=data['purpose_lemmas'].apply(purpose_group) # применяю функцию по каждому слову в ячейках столбца 'purpose_lemmas' и получаю новый столбец с целью кредита
```

Вывод

С помощью функции purpose_group, примененной к стобцу 'purpose_lemmas' и получаю новый столбец с категоризацией по цели кредита

Шаг 3.

- Есть ли зависимость между наличием детей и возвратом кредита в срок?

In [7]:
```
def with_children (i):
    try:
        if i>=3:
            return ('многодетная семья')
        elif i>0:
            return ('есть дети')
        return 'нет детей'
    except:
        print('проверь код, есть какая-то ошибка')
data['children_id']=data['children'].apply(with_children)
data_pivot_children = data.pivot_table(index=['children_id'], columns='debt',
values='total_income', aggfunc='count')

data_pivot_children['ratio, %']=data_pivot_children[1]/data_pivot_children[0]*
100 #добавляю столбец 'ratio, %', который характеризует % невыплативших кредит
по категориям семьи, в которых "есть дети","нет детей" или "многодетная семья"
data_pivot_children.sort_values('ratio, %', ascending=False)
print(data_pivot_children)
x=data_pivot_children.loc["есть дети", 1]
print(x)
y=data_pivot_children.loc["многодетная семья", "ratio, %"]
print(y)
#df.loc[row_indexer, column_indexer].
```

```
debt                0     1   ratio, %
children_id
есть дети        6268   639  10.194639
многодетная семья   417    39   9.352518
нет детей         13028  1063   8.159349
639
9.352517985611511
```

Вывод

После анализа отношения количества невыплаченных кредитов в срок к количеству людей, взявших кредит, можно сделать вывод о том, что:

самое большое количество кредитуемых не имеет детей (65,7%) при этом 8,15 % из них не возвращают кредит в срок, и в семьях с детьми чаще не возвращают кредиты в срок.

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

In [8]:
```
data_pivot_family = data.pivot_table(index=['family_status'], columns='debt', values='total_income', aggfunc='count') #строю сводную таблицу

data_pivot_family['ratio, %']=data_pivot_family[1]/data_pivot_family[0]*100 #добавляю столбец 'ratio, %', который характеризует % невыплативших кредит во время в соответствующей категории по семейному статусу
print(data_pivot_family.sort_values('ratio, %', ascending=False))

#не очень информативные данные по категоризации 'family_status', объединим всех одиночек в одну группу, а женатых и гражданский брак в отдельную сводную
#для этого функцией пробежимся по столбцу 'family_status_id'. Гражданский брак "1" и женатые "0", остальные одинокие "2","3","4"
def family_or_solitary (i):
    try:
        if i==1:
            return ('в браке')
        elif i==0:
            return ('в браке')
        return 'одинокие'
    except:
        print('проверь код, есть какая-то ошибка')
data['family_or_solitary']=data['family_status_id'].apply(family_or_solitary)

data_pivot_family_id = data.pivot_table(index=['family_or_solitary'], columns='debt', values='total_income', aggfunc='count') #строю сводную таблицу

data_pivot_family_id['ratio, %']=data_pivot_family_id[1]/data_pivot_family_id[0]*100 #добавляю столбец 'ratio, %', который характеризует % невыплативших кредит во время в соответствующей категории по семейному статусу
print(data_pivot_family_id.sort_values('ratio, %', ascending=False))
```

```
debt                       0    1   ratio, %
family_status
не женат / не замужем   2536  274  10.804416
гражданский брак        3763  388  10.310922
женат / замужем        11408  931   8.160940
в разводе               1110   85   7.657658
вдовец / вдова           896   63   7.031250
debt                     0     1  ratio, %
family_or_solitary
одинокие              4542   422  9.291061
в браке              15171  1319  8.694219
```

Вывод

Можно сделать вывод о том, что зависимости от между семейным положением и возвратом кредита в срок нет. Обе категории примерно одинаково часто не возвращают кредит в срок.

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

In [9]:
```
def income_id (i): # данную функцию применю к столбцу "total_income", чтобы ка
тегоризировать клиентов банка по доходу
    try:
        if i>1500000:   # разбиваю по категориям на основе статистических данн
ых по зп россиян 2017-2018 гг.
            return ('богатые')
        elif i>1000000:
            return ('состоятельные')
        elif i>500000:
            return ('верхний средний класс')
        elif i>250000:
            return ('средний класс')
        elif i>150000:
            return ('предсредний класс')
        elif i>100000:
            return ('нижний средний класс')
        elif i>36000:
            return ('выше бедности')
        return 'бедные'
    except:
        print('проверь код, есть какая-то ошибка')
data['income_id']=data['total_income'].apply(income_id) # применяю к стобцу "t
otal_income" функцию, чтобы получить данные для категоризации
#print(data[['total_income','income_id']].head(20)) #проверяю, работает ли фун
кция
data_pivot_income = data.pivot_table(index=['income_id'], columns='debt', valu
es='total_income', aggfunc='count') #строю сводную таблицу

data_pivot_income['ratio, %']=data_pivot_income[1]/data_pivot_income[0]*100 #д
обавляю столбец 'ratio, %', который характеризует % невыплативших кредит во вр
емя в соответствующей категории по доходу
data_pivot_income.sort_values('ratio, %', ascending=False) #сортирую по значен
иям
```

Out[9]:

debt	0	1	ratio, %
income_id			
богатые	6	1	16.666667
нижний средний класс	7146	661	9.249930
предсредний класс	5840	532	9.109589
выше бедности	4038	350	8.667657
средний класс	2410	180	7.468880
верхний средний класс	185	12	6.486486
состоятельные	17	1	5.882353
бедные	71	4	5.633803

Вывод

Можно разбить столбец по квартилям данных, которые можно получить методом describe(). Строить выводы по 6 клиентам и говорить, что людям с доходом 1,5 млн опасно давать кредиты, неверно. Я проводила категоризацию на основании статистики по средней ЗП за 2017-2018 год из интернета. Малоимущие А1, беднейшие А2 в нашем списке отсутствуют, так как минимальная зп по нашим данным 20667 рублей. Категория А3 Бедные от 18 000 руб, Категория Выше бедности от 36 000 руб на человека, средний класс по данной статистике разделили на 3 группы: Категория А5 Нижний средний класс от 100 000 на человека, А6 Предсредний класс с зп более 150 000 руб, А7 Средний класс от 250 000 руб, А8 Верхний средний класс от 500 000 руб. Состоятельные А9 от 1 000 000 руб, а А10 богатые люди, зарабатывающие свыше 1 500 000 руб в месяц. После категоризации по доходу и построения сводной таблицы можно сделать вывод о том, что уровень дохода не влияет на возвратность в срок кредита, так как богатые люди возращают кредит в срок реже, чем бедные люди, при этом состоятельные люди возвращают кредит в срок чаще, чем нижний средний класс.

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

In [10]:
```
data_pivot_purpose = data.pivot_table(index='purpose_group',  columns='debt', values='total_income', aggfunc='count')

data_pivot_purpose['ratio, %']=data_pivot_purpose[1]/data_pivot_purpose[0]*100
#добавляю столбец 'ratio, %', который характеризует % невыплативших кредит в соответствии с категорией цель кредита
data_pivot_purpose.sort_values('ratio, %', ascending=False)
data_pivot_purpose
```

Out[10]:

debt	0	1	ratio, %
purpose_group			
автокредит	3903	403	10.325391
ипотека	8817	683	7.746399
кредит на коммерческую недвижимость	1212	99	8.168317
кредит на образование	3643	370	10.156464
кредит на свадьбу	2138	186	8.699719

Вывод

Проанализировав данные, можно сделать вывод о том, что есть небольшая зависимость возврата кредита в срок от цели кредита. Реже всего невыплата кредита происходит по ипотеке, так как люди чаще всего более ответственно подходят к данному виду кредитования, так как оно долгосрочное. А чаще всего невыплата кредита происходит в категории "автокредит", так как часто владельцы недооценивают стоимость обслуживания автомобиля.

Шаг 4. Общий вывод

В данном проекте главной задачей было проанализировать статистику о платёжеспособности клиентов и сделать выводы, влияет ли семейное положение и количество детей, а также цель кредитования клиента на факт погашения кредита в срок. по результатам анализа можно сделать вывод о том, что: людьми с детьми на 2% чаще, чем люди без детей, выплачивают кредит не в срок. одинокие люди почти на 1% чаще не выплачивают кредит в срок, чем люди в браке(в гражданском в том числе). есть небольшая зависимость выплаты кредита в срок от целей кредита: чаще не выплачивается автокредит, а кредит по ипотеке, наоборот, возвращают в срок чаще.