tnarembekov0212@gmail.com
tg: FountainEww
+7-(700)-271-77-18 +7-(916)-964-25-83
Astana. Temirlan Narembekov
tnarembekov0212@gmail.com
tg: FountainEww
+7-(700)-271-77-18 +7-(916)-964-25-83
Astana.

1 Задание

В архиве во вложении данные по выдачам кредитов и платежам: плановым и фактическим. Данные актуальны на 08.12.2022. Проанализируйте характер поведения клиентов с точки зрения просрочки: какая динамика просрочки, наблюдается ли рост или снижение?

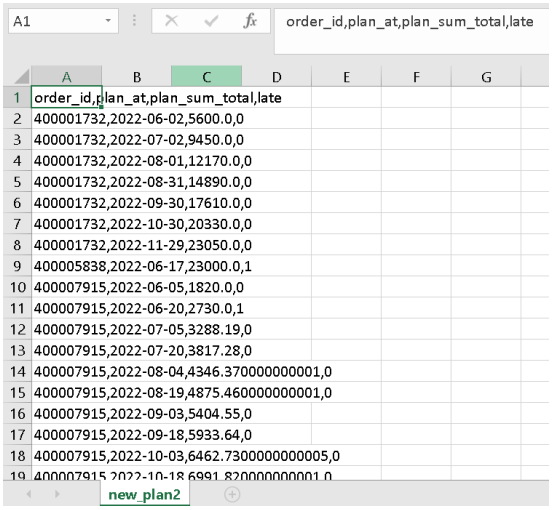
- order_id – номер заявки
- created_at – дата создания заявки
- put_at – дата выдачи
- closed_at – дата закрытия
- issued_sum – сумма выдачи
- plan_at – дата планового платежа
- plan_sum_total – сумма планового платежа (накопленным итогом)
- paid_at – дата фактического платежа
- paid_sum – сумма фактического платежа

1.Число(частота) просрочек

1.1 Обработка данных

Необходимо проанализировать просрочку, поэтому рассмотрим plan.csv.

Добавим столбец "late", состоящий из 0 и 1, где каждой записи order_id соответствует 1 (если есть просрочка) и 0 (если просрочки нет), после сохраним это в new_plan2.csv.



	order_id,plan_at,plan_sum_total,late
1	order_id,plan_at,plan_sum_total,late
2	400001732,2022-06-02,5600.0,0
3	400001732,2022-07-02,9450.0,0
4	400001732,2022-08-01,12170.0,0
5	400001732,2022-08-31,14890.0,0
6	400001732,2022-09-30,17610.0,0
7	400001732,2022-10-30,20330.0,0
8	400001732,2022-11-29,23050.0,0
9	400005838,2022-06-17,23000.0,1
10	400007915,2022-06-05,1820.0,0
11	400007915,2022-06-20,2730.0,1
12	400007915,2022-07-05,3288.19,0
13	400007915,2022-07-20,3817.28,0
14	400007915,2022-08-04,4346.370000000001,0
15	400007915,2022-08-19,4875.460000000001,0
16	400007915,2022-09-03,5404.55,0
17	400007915,2022-09-18,5933.64,0
18	400007915,2022-10-03,6462.7300000000005,0
19	400007915,2022-10-18,6991.820000000001,0

new_plan2

Для этого будем сверяться с payments.csv, чтобы сравнивать фактическую информацию оплат с запланированной.

preprocessing.py

C: > Users > user > Desktop > TEST-CHALLENGES > DEVIM > preprocessing.py > check_late

```python
# Преобразуем строки в datetime
payments["paid_at"] = pd.to_datetime(payments["paid_at"]).dt.date
plan["plan_at"] = pd.to_datetime(plan["plan_at"]).dt.date
# Функция для вычисления `late`
def check_late(row, payments):
    order_id = row["order_id"]
    plan_at = row["plan_at"]
    plan_sum_total = row["plan_sum_total"]
    # Все платежи по заказу
    order_payments = payments[payments["order_id"] == order_id].copy()
    # Добавляем столбец "учтен" (сначала все платежи не учтены)
    order_payments["used"] = False
    # Найти все платежи **до текущей даты plan_at**, но не только в этом периоде
    relevant_payments = order_payments[order_payments["paid_at"] <= plan_at]
    # Если платежей вообще нет, значит 100% просрочка
    if relevant_payments.empty:
        return 1
    # Считаем баланс с учетом ранних платежей
    total_paid = 0
    for index, payment in relevant_payments.iterrows():
        if total_paid >= plan_sum_total:
            break  # Если уже оплатили нужную сумму — выходим
        total_paid += payment["paid_sum"]
        order_payments.at[index, "used"] = True  # Отмечаем платеж как учтенный
    # Проверяем, хватило ли денег
    if total_paid >= plan_sum_total:
        return 0  # Нет просрочки
    # Если денег не хватило, проверяем, есть ли платежи **после plan_at** (опоздавшие)
    late_payments = order_payments[~order_payments["used"] & (order_payments["paid_at"] > plan_at)]
    if not late_payments.empty:
        return 1  # Просрочка, так как часть денег пришла поздно
    return 1  # Просрочка (недостаточно денег)
# Рассчитываем `late` для первых 10 строк в `plan`
plan_subset = plan.head(10).copy()  # Берем первые 10 строк
plan["late"] = plan.apply(lambda row: check_late(row, payments), axis=1)
plan.to_csv("C:\\Users\\user\\Desktop\\TEST-CHALLENGES\\DEVIM\\task_2_data\\new_plan2.csv", index=False)
```

Далее сгруппируем данные по plan_at и будем складывать данные в "late"и, таким образом, посчитаем количество просрочек за каждый день в плане.

1.2 Анализ по дням

Теперь мы готовы рассмотреть график просрочек на временной шкале за каждый день.

1)ТРЕНД Здесь мы наблюдаем явный Тренд: до 2022-09-11 идет постепенный рост числа просрочек, а после число так же уходит на спад, но при этом фиксируется на примерно одном и том же малом числе с 2023-03-09 по 2023-06-11.

Проверим статистическую гипотезу Вальда-Вольфовица об отсутствии тренда и случайности:
Введем гипотезу H_0 и альтернативу H_1:

H_0 : Тренд отсутствует - т.е.случайность ряда просрочек,
H_1 : Тренд присутствует - неслучайность.

```python
# Преобразуем в бинарную форму относительно медианы
median = np.median(daily_overdues)
binary_sequence = np.where(daily_overdues > median, 1, 0)
# Запускаем тест Вальда-Вольфовица
z_stat, p_value = runs.runstest_1samp(binary_sequence, correction=True)
```

```
Статистика Z: -14.806094980127613
P-value: 1.3379040845584634e-49
Гипотеза о случайности отвергается: имеется тренд.
```

Как и ожидалось, существует тренд.
в конце периода(по сравнению с началом) количество просрочек значительно упало
Но это если не учитывать, что в конце периода было намного меньше платежей чем в начале.

1.3 Анализ количества платежей

Давайте посмотрим на то как менялось число транзакций изо дня в день.

Как и было замечено, количество платежей за отдельный день не оставалось прежним, оно менялось, причем на графике видно, что похожим образом как и число просрочек.
Имеет смысл теперь рассматривать долю просрочек среди числа всех платежей за конкретный день.

1.4 Анализ долей просрочек

Видим, что доля в конце выросла по сравнению с началом. И вообще кажется, что динамика растет.
Узнаем наверняка определив монотонный тренд, т.е. рост или снижение.

Удобно сделать это тестом Манна-Кендалла:

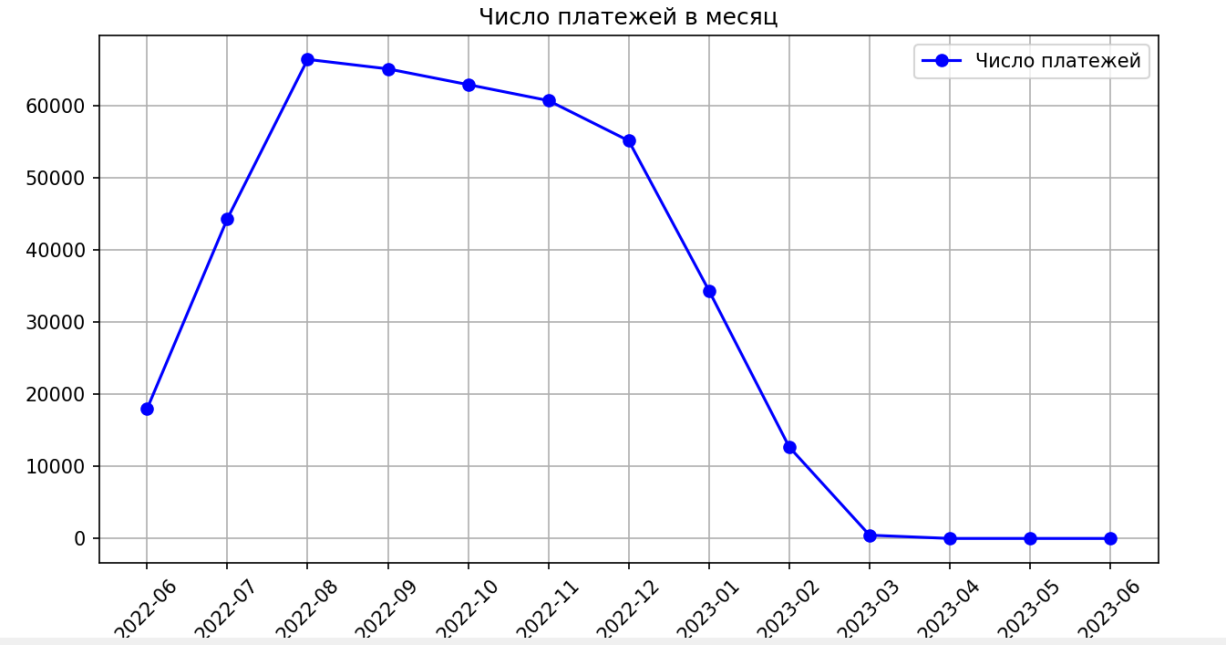
```
31    result = original_test(daily_stats["overdue_ratio"])
32    print(result)

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

Mann_Kendall_Test(trend='decreasing', h=True, p=0.020822131216128303, z=-2.3111941887103558,
.0, var_s=3230428.3333333335, slope=-0.0002325155593624674, intercept=0.3558425670069317)
```

p-значение и trend=’decreasing’ указывают на то, что динамика снижается, хотя по графику может показаться обратное.

Дело в том, что несмотря на рост в конце, убывание с 50 по 200 день(а так же с 260 по 270) перевешивает это, отсюда и можно сделать Вывод: Происходит общее снижение динамики количества просроченных платежей.

Имеет смысл перейти от ежедневных к ежемесячным показателям.

1.5 Анализ по месяцам




```
Число просрочек в месяц: Mann_Kendall_Test(trend='decreasing', h=True, p=0.0006196284746113889, z=-3
7307692307692307, s=-57.0, var_s=267.6666666666667, slope=-2261.103174603175, intercept=23467.619047
доля просрочек в месяц: Mann_Kendall_Test(trend='no trend', h=False, p=0.21922547902520906, z=1.2285
7692307692, s=21.0, var_s=265.0, slope=0.042493483581458186, intercept=0.1519356502353888)
```

ВЫВОД: Ежемесячные Доли просрочек не имеют статистически значимый тренд, но это не означает стабильность динамики:
1)с июня 2022 по ноябрь 2022 доля количества просрочек снижается, в то время как само число просрочек стремительно растет и затем так же стремительно убывает(а число платежей точно так же быстро растет, но в разы медленней убывает).
2)С ноября 2022 заметен уверенный рост в долях, в то время как число просрочек и платежей заметно уменьшается.
3)В целом, если не учитывать динамику числа всех платежей, то можно наблюдать снижение числа просрочек в месяц.
А если принимать во внимание рост и последующий спад числа платежей, то динамика такова, что в конечном итоге доля количества просроченных платежей растет за этот период в общей картине.

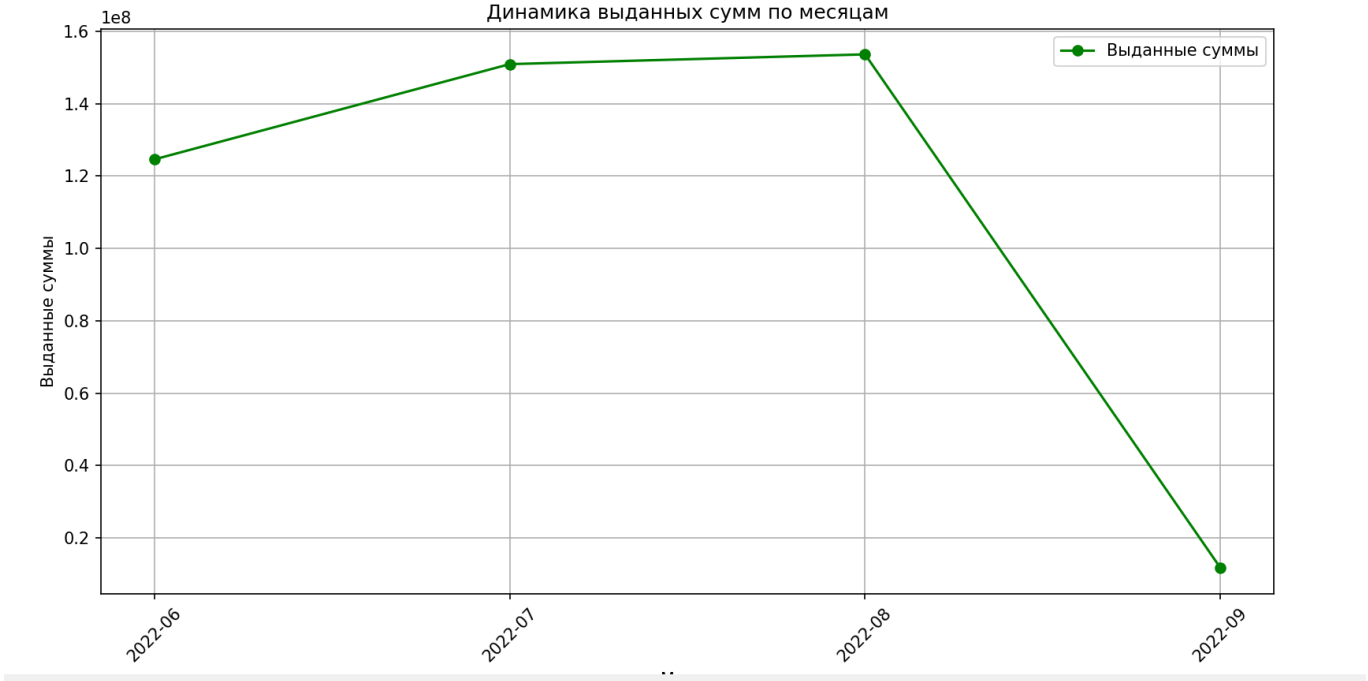
2 Суммы просрочек

Теперь рассмотрим динамику под другим углом.
Снова будем работать с ежемесячными данными и сделаем таблицу в которой будет отображено общая сумма задолженностей, а так же погашенная и просроченная суммы за каждый месяц.

	month	plan_sum_total	paid_sum	late	late_ratio
0	2022-06	7.234713e+07	6.370377e+07	8.643363e+06	0.119471
1	2022-07	2.385414e+08	1.467863e+08	9.175504e+07	0.384650
2	2022-08	3.761703e+08	1.739525e+08	2.022178e+08	0.537570
3	2022-09	4.724597e+08	1.141835e+08	3.582762e+08	0.758321
4	2022-10	5.646301e+08	3.601413e+07	5.286159e+08	0.936216
5	2022-11	6.456994e+08	2.085753e+07	6.248419e+08	0.967698
6	2022-12	6.566420e+08	3.703001e+06	6.529390e+08	0.994361
7	2023-01	4.379735e+08	NaN	NaN	NaN
8	2023-02	1.699278e+08	NaN	NaN	NaN
9	2023-03	6.171690e+06	NaN	NaN	NaN
10	2023-04	5.951928e+04	NaN	NaN	NaN
11	2023-05	6.585682e+04	NaN	NaN	NaN
12	2023-06	3.349837e+04	NaN	NaN	NaN

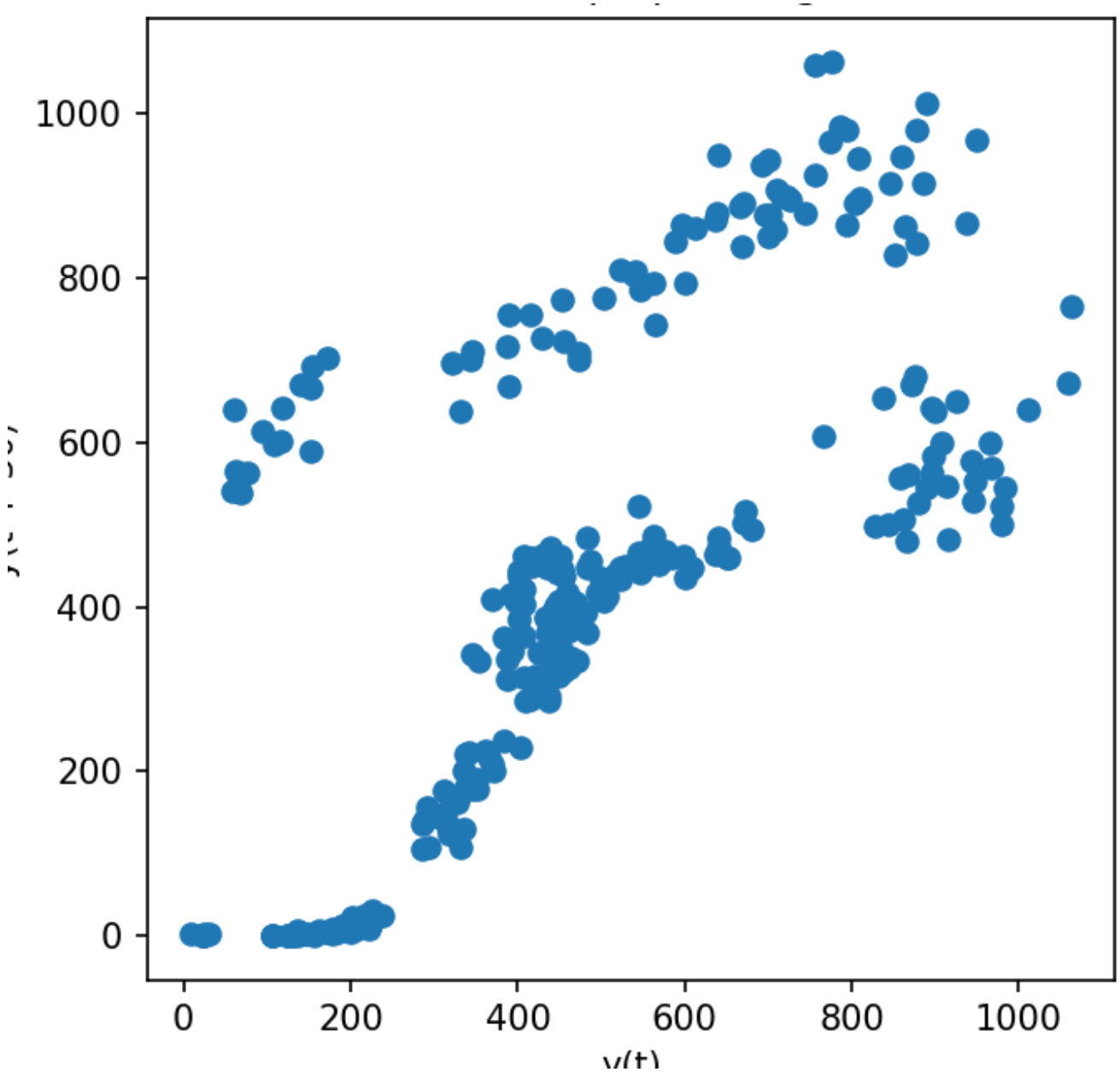
Заметим, что с января 2023 отсутствуют данные о платежах.

Видим рост в обоих показателях.
Интересно посмотреть на то, как меняется issued_sum для полной картины. Наблюдаем рост выдачи до августа 2022 и резкий спад в



сентябре 2022(начиная с сентября 2022 отсутствуют данные по новым выдачам - можем считать, что выдача прекратилась т.к. данные актуальны на декабрь 2022.) Но несмотря на это обе динамики продолжают расти даже после сентября 2022.
ВЫВОД: Динамика суммы просрочек растёт.

Вернемся к ежедневному количеству просрочек периода.
Мы отмечали отсутствие тренда. А как вообще зависимы эти данные?(и зависимы ли вообще)
Построим лаговый график с лагом 30.



Заметна сильная корреляция - точки разбиваются на группы, это указавает на некоторую зависимость и неоднородность распределения данных.