1. **Цель проекта**

После недавнего A/B-теста, когда часть игроков могла приобрести премиумную броню по скидке, руководство хочет узнать, как это повлияло на ARPU (средняя прибыль на игрока), ARPPU (средняя прибыль на платящего игрока) и траты внутриигровой валюты.

Цель проекта - выяснить, стоит ли проводить акцию в дальнейшем. Если игроки, участвовавшие в акции, принесли больше денег, чем игроки, у которых акции не было, то стоит повторять акцию и при этом уже на всех игроках.

Для решения задачи необходимо выбрать среду для работы с данными, проанализировать суммарные и средние траты игроков, а также построить доверительные интервалы от средних значений с точностью 95%. Если доверительные интервалы пересекаются, то это означает, что результаты случайны и акция не принесла результатов.

**2. Анализ источников**

Для работы с данными был выбран язык Python, в частности его библиотека Pandas. Python идеально подходит если:

* Файлов много
* В файле более миллиона строк
* Файлы могут содержать мусор
* Нужны сложные статистические функции

В нашем случае мы имеем 5 файлов, в каждом более 8 миллионов строк, которые необходимо очистить от нечестных пользователей. Также нам необходимо построить доверительные интервалы, что является сложно статистической функцией, поэтому Python оптимальный вариант для анализа.

**3. Очистка данных**

**Формируем таблицу с суммарными тратами пользователей и внутриигровыми покупками:**

**import** pandas **as** pd  
  
money = pd.read\_csv(**'Money.csv'**)  
cash = pd.read\_csv(**'Cash.csv'**)  
df = pd.merge(money,cash,on=[**'user\_id'**,**'date'**])  
  
*### Таблица юзеров с суммами платежей и внутригировых трат ###*group\_pay = df.groupby(**'user\_id'**).sum(**'money'**)

print(group\_pay.head(10))

**Результат выполнения кода:**

money cash

user\_id

00036L-1T885I 3.97 7600

0004MK-MRBUKF 5.95 6150

0004MK-VW0O9L 4.96 6300

0005S4-UEC6BE 5.95 5000

0007I5-EWUD6O 6.94 6350

0009DN-TWTJIG 5.97 7250

000A88-I8NCEL 3.97 4500

000B68-JX82RN 5.95 7200

000B9M-CLSPQD 4.96 7750

000F5V-PDHPGB 6.95 5950

**Объединяем с таблицей уже известных читеров для их дальнейшего поиска, сортируем по убыванию суммы внутриигровой валюты:**

**import** pandas **as** pd  
**import** pay  
cheaters = pd.read\_csv(**'Cheaters.csv'**)  
df = pd.merge(pay.group\_pay,cheaters, on=**'user\_id'**)

*### Таблица сумм платежей и игровой валюты с проверкой на читерство ###*find\_cheaters = df.drop\_duplicates(keep=**'first'**).reset\_index(drop=**True**).sort\_values(**'cash'**,

ascending=**False**)

print(find\_cheaters.head(10))

**Результат выполнения кода:**

user\_id money cash cheaters

201872 6Q3KB0-DF2SIX 0.0 253150 1

134653 4HCVCA-V3M19Y 0.0 240950 0

22877 0R9XB0-4J1EVJ 0.0 238500 0

761380 PDOMHG-6CPEZ4 0.0 237600 1

343839 BGTT4M-1ZFLMA 0.0 235100 0

954307 VTOLQS-23AL7R 0.0 231750 1

730408 OCRQ1L-GZH0BP 0.0 230600 0

697560 N9HQS0-Y2L88S 0.0 226500 1

359716 BZRZ88-435FHY 0.0 225650 0

452362 F35CFN-CGKDDG 0.0 225350 0

print(find\_cheaters[690:700])

759257 PB8LKO-UJZRXE 0.00 68500 0

882249 TES7Y0-3LQ40U 0.00 64900 0

243464 83SEQM-9J397V 0.00 61950 0

548867 IALNLO-F5DDS6 0.00 55800 1

117261 3WOSJF-UC2DAO 0.00 50150 1

497666 GLGM0P-VMHL3W 0.00 46700 1

131247 4DBX3L-2LHKON 0.00 20700 0

882461 TF1GNR-J53YKF 15.94 13250 0

218951 7AJHSG-INMTKY 12.95 12700 0

1020870 Y1DZCO-IT6FQW 10.95 12500 0

Читерами считаются те, кто за все время проведения акции не сделал ни одной покупки, но при этом тратил внутриигровую валюту.

*### Читеры ###*cheaters = find\_cheaters.loc[(find\_cheaters[**'money'**]==0) & (find\_cheaters[**'cash'**]>0)]  
  
*### Честные игроки ###*players = find\_cheaters.loc[find\_cheaters[**'money'**]

> 0].drop(columns=[**'money'**,**'cash'**,**'cheaters'**])

**4. Использование статистических методов**

**Формируем таблицу с суммами трат групп тестирования по дням**

**import** pandas **as** pd  
**import** cheaters  
  
money = pd.read\_csv(**'Money.csv'**)  
abgroup = pd.read\_csv(**'ABgroup.csv'**)  
  
df = pd.merge(money,cheaters.players,on=**'user\_id'**)  
df = pd.merge(df,abgroup,on=**'user\_id'**)  
  
*### Таблица трат пользователей за каждый день с обозначенеим, к какой группе тестирования он относится ###*group\_pay = df.drop\_duplicates(keep=**'first'**).reset\_index(drop=**True**)  
group\_pay\_table = group\_pay.to\_csv(**'group\_pay.csv'**,sep=**';'**,index=**False**)  
  
*### Таблица с суммами трат по дням групп тестирования ###*df =group\_pay[**'money'**].groupby([group\_pay[**'date'**],group\_pay[**'group'**]]).sum()  
sum\_group\_pay = df.to\_csv(**'sum\_group\_pay.csv'**,sep=**';'**)

**Результат выполнения кода:**

date group

10.07.2021 control 393870.25

test 410341.94

11.07.2021 control 392476.58

test 410937.94

12.07.2021 control 390787.80

test 409077.72

13.07.2021 control 382743.77

test 400947.80

14.07.2021 control 364514.38

test 383022.82

15.07.2021 control 335543.71

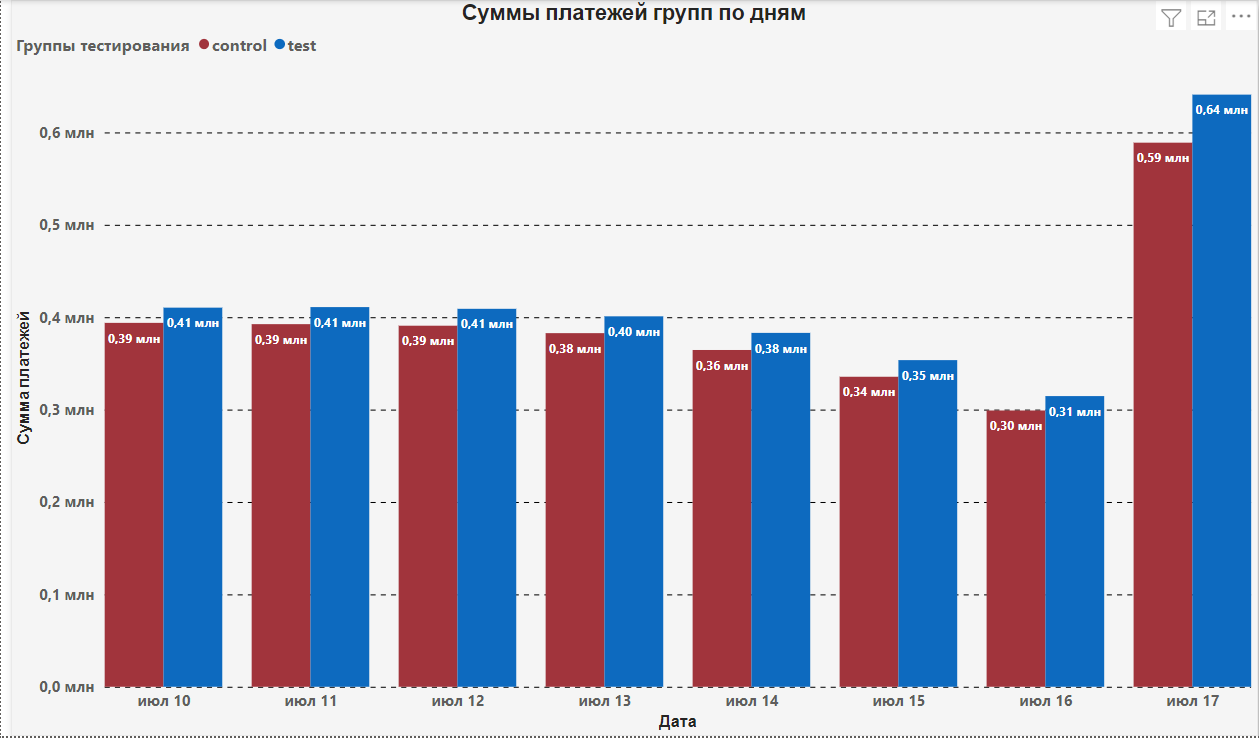
test 353538.54

16.07.2021 control 299028.24

test 314611.74

17.07.2021 control 588901.06

test 640910.68

Строим график в Power BI

Как мы видим, акция существенно не повлияла на платежи игроков. Игроки тестовой группы тратили больше в день в среднем на 0.2 млн. Большой всплеск наблюдается в последний день проведения акции.

Ищем **ARPU** (средняя прибыль на игрока) по группам тестирования

Для начала надо посчитать количество игроков в каждой группе:

**import** pandas **as** pd  
**import** cheaters **as** ch  
  
df = pd.read\_csv(**'ABgroup.csv'**)  
abgroup = df.drop\_duplicates(keep=**'first'**).reset\_index(drop=**True**)  
*### Фильтруем читеров ###*abgroup = pd.merge(abgroup,ch.players, on=**'user\_id'**)  
  
*### Кол-во игроков в каждой тестируемой группе ###*count\_us\_gr = abgroup[**'group'**].value\_counts()

**Результат выполнения кода:**

control 539840

test 539236

ARPU:

**import** pandas **as** pd  
**import** ABgroup **as** ab  
arpu = pd.read\_csv(**'sum\_group\_pay.csv'**,sep=**';'**)  
  
*### ARPU ###*cnt\_index = arpu[arpu[**'group'**] ==**'control'**].index  
arpu.loc[cnt\_index,**'ARPU'**] = arpu[**'money'**]/ab.count\_us\_gr[0]  
  
test\_index = arpu[arpu[**'group'**]==**'test'**].index  
arpu.loc[test\_index,**'ARPU'**] = arpu[**'money'**]/ab.count\_us\_gr[1]  
  
arpu = arpu.to\_csv(**'ARPU.csv'**,sep=**';'**,index=**False**)

**Результат выполнения кода:**

date group money ARPU

10.07.2021 control 393870.25 0.729606

10.07.2021 test 410341.94 0.760969

11.07.2021 control 392476.58 0.727024

11.07.2021 test 410937.94 0.762074

12.07.2021 control 390787.80 0.723896

12.07.2021 test 409077.72 0.758625

13.07.2021 control 382743.77 0.708995

13.07.2021 test 400947.80 0.743548

14.07.2021 control 364514.38 0.675227

14.07.2021 test 383022.82 0.710306

15.07.2021 control 335543.71 0.621561

15.07.2021 test 353538.54 0.655629

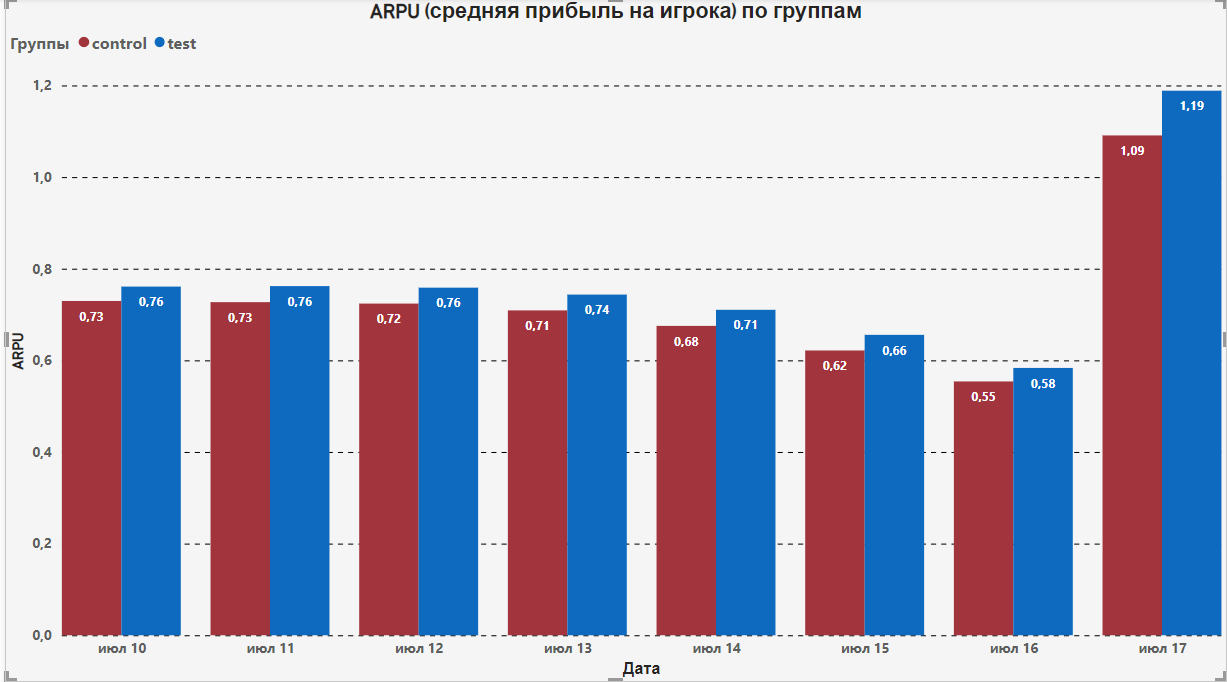
16.07.2021 control 299028.24 0.553920

16.07.2021 test 314611.74 0.583440

17.07.2021 control 588901.06 1.090881

17.07.2021 test 640910.68 1.188553

Строим график в Power BI



У тестируемой группы ARPU (средняя прибыль на игрока) немного больше, чем у контрольной группы.

**Ищем ARPU для игровых платформ:**

Для начала посчитаем количество игроков на каждой игровой платформе и найдем, сколько было платежей за каждый день от каждой игровой платформы.

**import** cheaters **as** ch  
df = pd.read\_csv(**'Platforms.csv'**)  
money = pd.read\_csv(**'Money.csv'**)  
  
df = df.drop\_duplicates(keep=**'first'**).reset\_index(drop=**True**)  
platforms = pd.merge(df,ch.players, on=**'user\_id'**)  
  
*### Кол-во игроков на каждой платформе ###*count\_players = platforms[**'platform'**].value\_counts()

**Результат выполнения кода:**

XBox 360478

PC 359465

PS4 359133

pay\_gr = pd.merge(money,platforms,on=**'user\_id'**)  
*### Таблица суммарных трат пользователей по игровым платформам ###*sum\_gr = pay\_gr[**'money'**].groupby([pay\_gr[**'date'**],pay\_gr[**'platform'**]]).sum()  
sum\_gr = sum\_gr.to\_csv(**'sum\_platforms.csv'**,sep=**';'**)

**Результат выполнения кода:**

date platform

10.07.2021 PC 269623.25

PS4 263354.52

XBox 271234.42

11.07.2021 PC 268809.62

PS4 263435.68

XBox 271169.22

12.07.2021 PC 268174.86

PS4 261821.72

XBox 269868.94

13.07.2021 PC 262318.91

PS4 256353.93

XBox 265018.73

14.07.2021 PC 248811.86

PS4 245229.58

XBox 253495.76

15.07.2021 PC 229229.25

PS4 226058.54

XBox 233794.46

16.07.2021 PC 203309.06

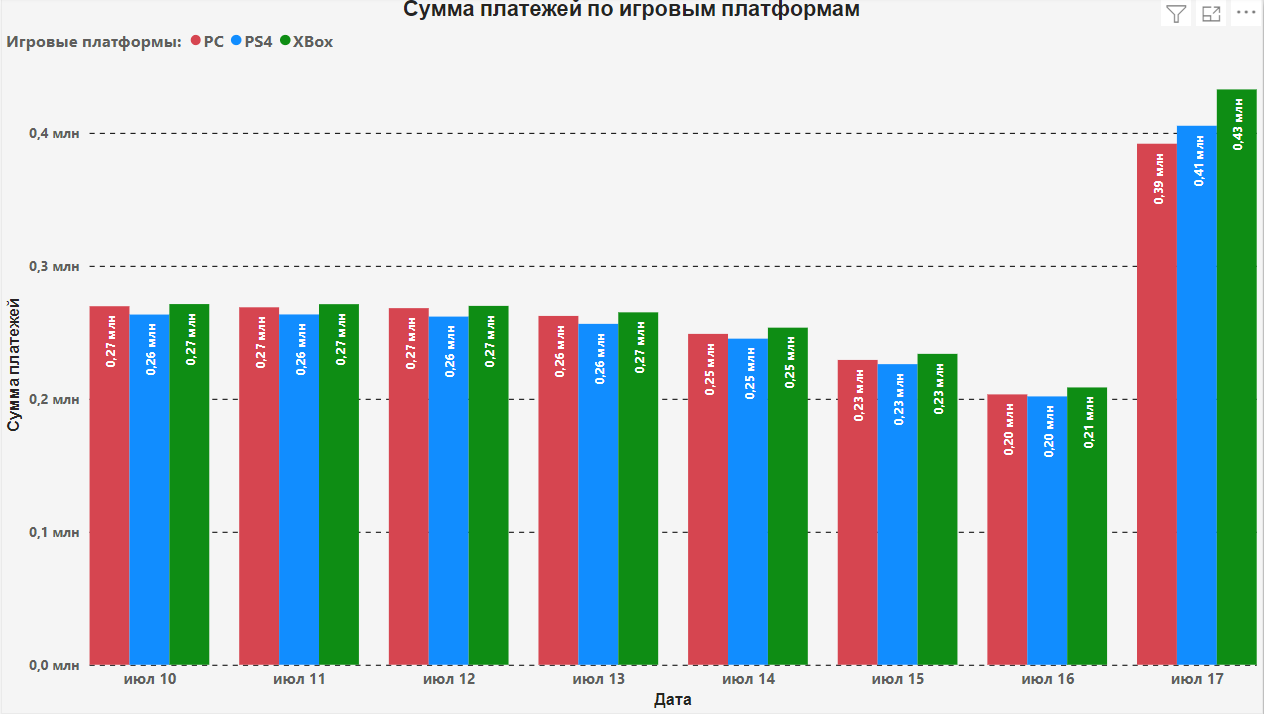
PS4 201790.43

XBox 208540.49

17.07.2021 PC 391826.13

PS4 405326.68

XBox 432658.93



Существенных различий между игровыми платформами нет, пользователи Xbox тратят чуть больше остальных, пользователи PS4 самые экономные.

**ARPU:**

**import** pandas **as** pd  
**import** platforms **as** pl  
df = pd.read\_csv(**'sum\_platforms.csv'**,sep=**';'**)  
  
xbox\_index = df[df[**'platform'**] ==**'XBox'**].index  
pc\_index = df[df[**'platform'**]==**'PC'**].index  
ps4\_index = df[df[**'platform'**]==**'PS4'**].index  
  
df.loc[xbox\_index,**'ARPU'**] = df[**'money'**]/pl.count\_players[0]  
df.loc[pc\_index,**'ARPU'**] = df[**'money'**]/pl.count\_players[1]  
df.loc[ps4\_index ,**'ARPU'**] = df[**'money'**]/pl.count\_players[2]  
  
df = df.to\_csv(**'arpu\_platforms,csv'**,sep=**';'**)

**Результат выполнения кода:**

date platform money ARPU

10.07.2021 PC 269623.25 0.750068

10.07.2021 PS4 263354.52 0.733306

10.07.2021 XBox 271234.42 0.752430

11.07.2021 PC 268809.62 0.747805

11.07.2021 PS4 263435.68 0.733532

11.07.2021 XBox 271169.22 0.752249

12.07.2021 PC 268174.86 0.746039

12.07.2021 PS4 261821.72 0.729038

12.07.2021 XBox 269868.94 0.748642

13.07.2021 PC 262318.91 0.729748

13.07.2021 PS4 256353.93 0.713813

13.07.2021 XBox 265018.73 0.735187

14.07.2021 PC 248811.86 0.692173

14.07.2021 PS4 245229.58 0.682838

14.07.2021 XBox 253495.76 0.703221

15.07.2021 PC 229229.25 0.637696

15.07.2021 PS4 226058.54 0.629456

15.07.2021 XBox 233794.46 0.648568

16.07.2021 PC 203309.06 0.565588

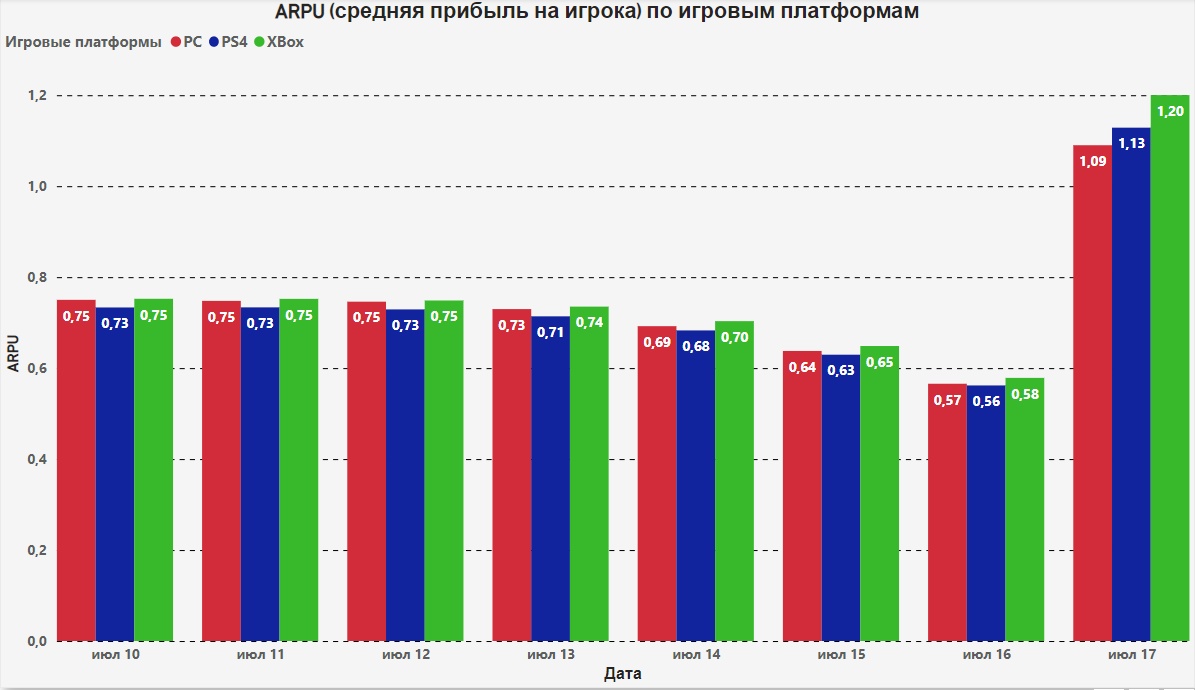
16.07.2021 PS4 201790.43 0.561882

16.07.2021 XBox 208540.49 0.578511

17.07.2021 PC 391826.13 1.090026

17.07.2021 PS4 405326.68 1.128626

17.07.2021 XBox 432658.93 1.200237



**ARPPU**

**import** pandas **as** pd

*### Загружаем таблицу с тратами каждого пользователя за каждый день ###*df = pd.read\_csv(**'group\_pay.csv'**,sep=**';'**)

*### Таблица с суммами трат по дням для групп тестирования ###*

sum\_gr\_pay = pd.read\_csv(**'sum\_group\_pay.csv'**,sep=**';'**)  
  
*### Оставляем строки с теми, кто делал платежи ###*df = df[df[**'money'**]>0]  
  
*### Кол-во заплативших пользователей каждый день ###*count\_pay = df[**'money'**].groupby([df[**'date'**],df[**'group'**]]).count()  
print(count\_pay)

**Результат выполнения кода:**

date group

10.07.2021 control 324875

test 335806

11.07.2021 control 324042

test 336006

12.07.2021 control 322620

test 334928

13.07.2021 control 317323

test 329120

14.07.2021 control 305262

test 317418

15.07.2021 control 285429

test 297646

16.07.2021 control 261176

test 271526

17.07.2021 control 321094

test 329332

count\_list = count\_pay.to\_list()

*### ARPPU по группам тестирования ###*sum\_gr\_pay[**'ARPPU'**] = sum\_gr\_pay[**'money'**]/count\_list  
arppu = sum\_gr\_pay.to\_csv(**'ARPPU.csv'**,sep=**';'**,index=**False**)

**Результат выполнения кода:**

date group money ARPPU

10.07.2021 control 393870.25 1.212375

10.07.2021 test 410341.94 1.221961

11.07.2021 control 392476.58 1.211190

11.07.2021 test 410937.94 1.223008

12.07.2021 control 390787.80 1.211294

12.07.2021 test 409077.72 1.221390

13.07.2021 control 382743.77 1.206165

13.07.2021 test 400947.80 1.218242

14.07.2021 control 364514.38 1.194103

14.07.2021 test 383022.82 1.206683

15.07.2021 control 335543.71 1.175577

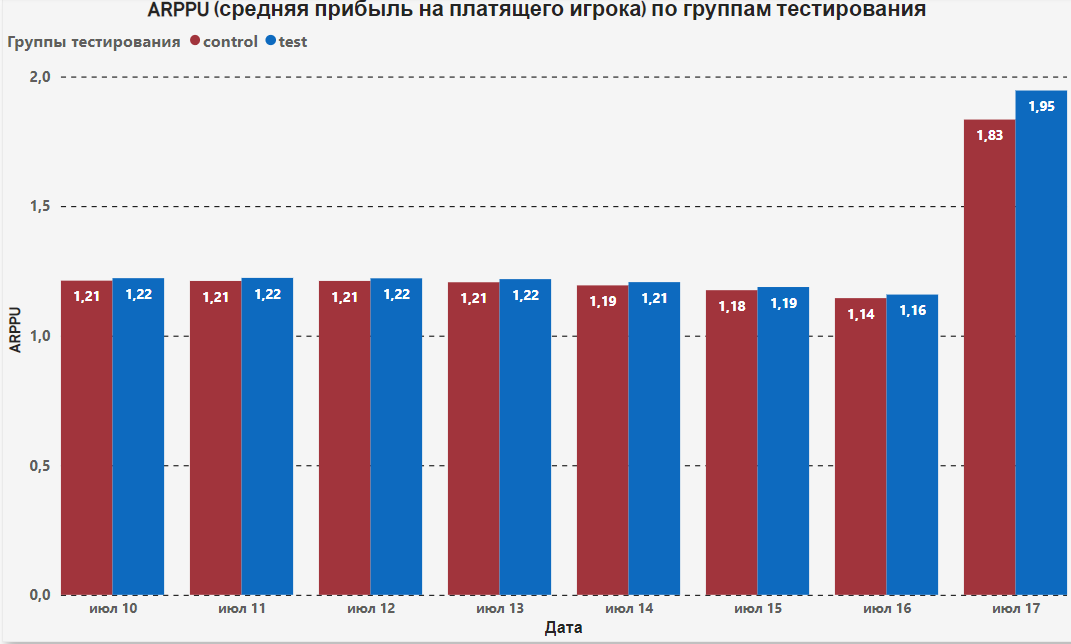
15.07.2021 test 353538.54 1.187782

16.07.2021 control 299028.24 1.144930

16.07.2021 test 314611.74 1.158680

17.07.2021 control 588901.06 1.834046

17.07.2021 test 640910.68 1.946093



ARPPU (средняя прибыль на платящего игрока) практически одинакова, самое большое отличие в последний день проведения акции.

ARPPU по игровым платформам

**import** pandas **as** pd  
**import** platforms **as** pl  
*### Загружаем суммарные траты по игровым платформам ###*  
sum\_platform = pd.read\_csv(**'sum\_platforms.csv'**,sep=**';'**)

*### Траты каждого пользователя по дням с указанием его игровой платформы###*  
df= pl.pay\_gr

*### Оставляем строки с теми, кто делал платежи ###*df = df[df[**'money'**]>0]  
  
*### Кол-во заплативших пользовталей ###*platform\_pay = df[**'money'**].groupby([df[**'date'**],df[**'platform'**]]).count()  
platform\_list = platform\_pay.to\_list()  
  
*### ARPPU ###*sum\_platform[**'ARPPU'**] = sum\_platform[**'money'**]/platform\_list  
ARPPU = sum\_platform.to\_csv(**'ARPPU\_platform.csv'**,sep=**';'**,index=**False**)  
**Результат выполнения кода:**

date platform money ARPPU

10.07.2021 PC 269623.25 1.219049

10.07.2021 PS4 263354.52 1.212788

10.07.2021 XBox 271234.42 1.219810

11.07.2021 PC 268809.62 1.218329

11.07.2021 PS4 263435.68 1.214930

11.07.2021 XBox 271169.22 1.218311

12.07.2021 PC 268174.86 1.218345

12.07.2021 PS4 261821.72 1.211420

12.07.2021 XBox 269868.94 1.219438

13.07.2021 PC 262318.91 1.213265

13.07.2021 PS4 256353.93 1.208607

13.07.2021 XBox 265018.73 1.214974

14.07.2021 PC 248811.86 1.199590

14.07.2021 PS4 245229.58 1.197165

14.07.2021 XBox 253495.76 1.204690

15.07.2021 PC 229229.25 1.181746

15.07.2021 PS4 226058.54 1.177106

15.07.2021 XBox 233794.46 1.186449

16.07.2021 PC 203309.06 1.150628

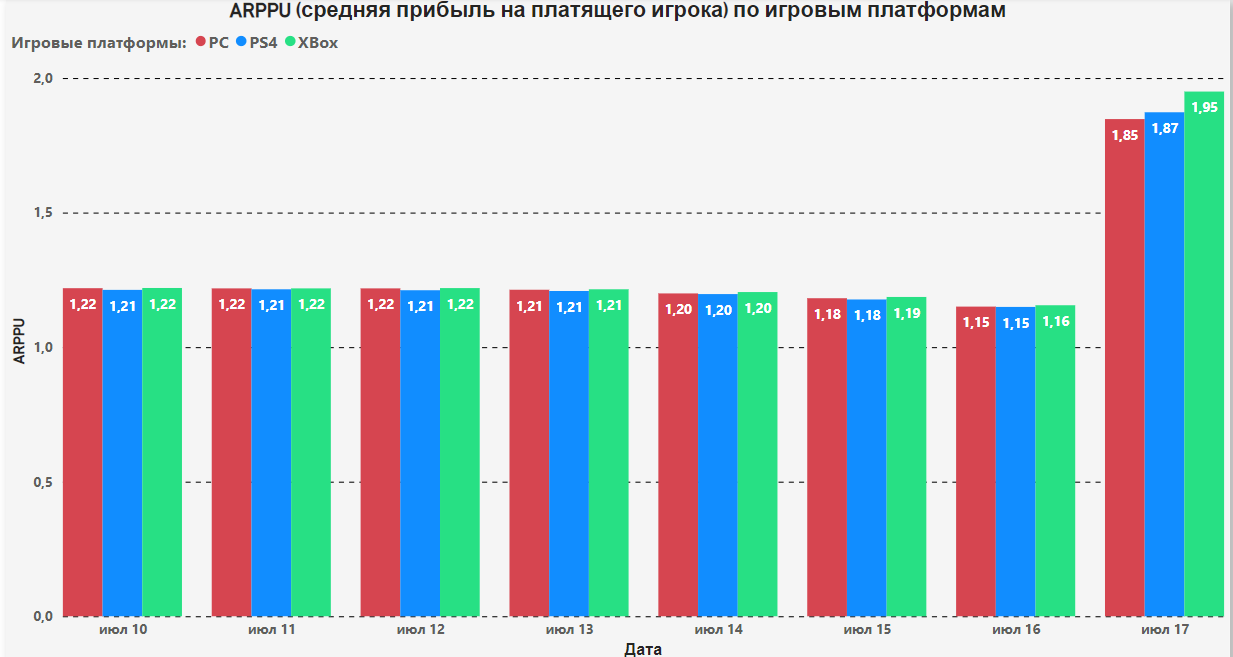
16.07.2021 PS4 201790.43 1.149430

16.07.2021 XBox 208540.49 1.155663

17.07.2021 PC 391826.13 1.847478

17.07.2021 PS4 405326.68 1.872767

17.07.2021 XBox 432658.93 1.949731



Все игровые платформы имеют примерно равные показатели по ARPPU.

**Построение доверительных интервалов**

**Доверительный интервал для контрольной группы от значений ARPU:**

**import** pandas **as** pd  
**import** matplotlib.pyplot **as** plt  
**from** scipy **import** stats  
**import** statsmodels.api **as** sm  
**import** numpy **as** np  
arpu = pd.read\_csv(**'ARPU.csv'**,sep=**';'**)  
arpu\_cnt = arpu[arpu[**'group'**]==**'control'**]  
  
*### Метрики ###***def** val(df=arpu\_cnt, column=**'ARPU'**):  
 print(df[column].describe())  
 print(**'--------Дисперсия--------'**)  
 print(df[column].var())  
val()  
print()  
*### Доверительный интервал ###***def** my\_norm\_confidence(df=arpu\_cnt, column = arpu\_cnt[**'ARPU'**], alpha = 0.95):  
 interval = stats.norm.interval(alpha, loc=arpu\_cnt[**'ARPU'**].mean(), scale=arpu\_cnt[**'ARPU'**].std())  
 **return** interval  
confidence = my\_norm\_confidence()  
print(**'Доверительный интервал:'**,confidence)  
print()  
*### Аномальное событие ###*arpu\_cnt[**'anomaly'**] = np.where(arpu\_cnt.ARPU > confidence[1], 1,0)  
print(**'Аномальное событие:'**,**'\n'**,arpu\_cnt[arpu\_cnt[**'anomaly'**]==1])  
  
*### Построение графиков ###*arpu\_cnt[**'ARPU'**].hist()  
plt.axhline(y=confidence[1], color=**"r"**, linestyle=**"-"**)  
plt.axhline(y=confidence[0], color=**"y"**, linestyle=**"-"**)  
plt.title(**"Доверительный интервал контрольной группы"**)  
plt.show()

**Результат выполнения кода:**

count 8.000000

mean 0.728889

std 0.158779

min 0.553920

25% 0.661810

50% 0.716445

75% 0.727669

max 1.090881

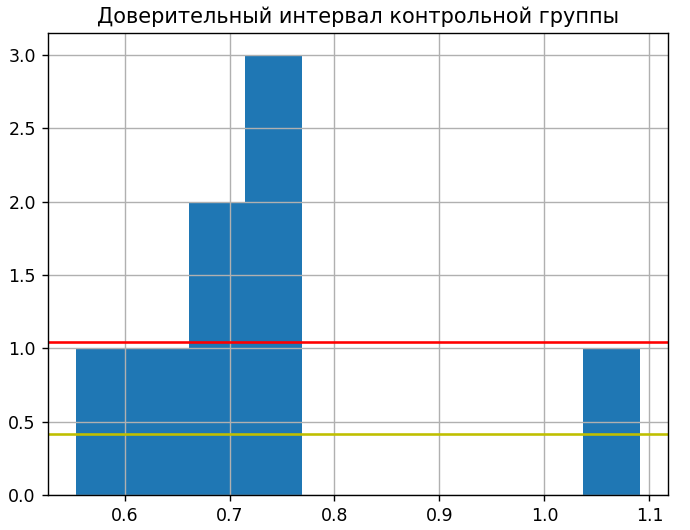
--------Дисперсия--------

0.02521081257091376

Доверительный интервал: (0.4176872245100419, 1.0400899826254055)

Аномальное событие:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Group | Money | ARPU | anomaly |
| 17.07.2021 | control | 588901.06 | 1.090881 | 1 |



**Доверительный интервал для тестируемой группы:**

**import** pandas **as** pd  
**import** matplotlib.pyplot **as** plt  
**from** scipy **import** stats  
**import** statsmodels.api **as** sm  
**import** numpy **as** np  
arpu = pd.read\_csv(**'ARPU.csv'**,sep=**';'**)  
arpu\_test = arpu[arpu[**'group'**]==**'test'**]  
  
*### Метрики ###***def** val(df=arpu\_test, column=**'ARPU'**):  
 print(df[column].describe())  
 print(**'--------Дисперсия--------'**)  
 print(df[column].var())  
val()  
print()  
*### Доверительный интервал ###***def** my\_norm\_confidence(df=arpu\_test, column = arpu\_test[**'ARPU'**], alpha = 0.95):  
 interval = stats.norm.interval(alpha, loc=arpu\_test[**'ARPU'**].mean(), scale=arpu\_test[**'ARPU'**].std())  
 **return** interval  
confidence = my\_norm\_confidence()  
print(**'Доверительный интервал:'**,confidence)  
print()  
*### Аномальное событие ###*arpu\_test[**'anomaly'**] = np.where(arpu\_test.ARPU > confidence[1], 1,0)  
print(**'Аномальное событие:'**,**'\n'**,arpu\_test[arpu\_test[**'anomaly'**]==1])  
  
*### Построение графиков ###*arpu\_test[**'ARPU'**].hist()  
plt.axhline(y=confidence[1], color=**"r"**, linestyle=**"-"**)  
plt.axhline(y=confidence[0], color=**"y"**, linestyle=**"-"**)  
plt.title(**"Доверительный интервал тестируемой группы"**)  
plt.show()

**Результат выполнения кода:**

count 8.000000

mean 0.770393

std 0.180320

min 0.583440

25% 0.696637

50% 0.751086

75% 0.761245

max 1.188553

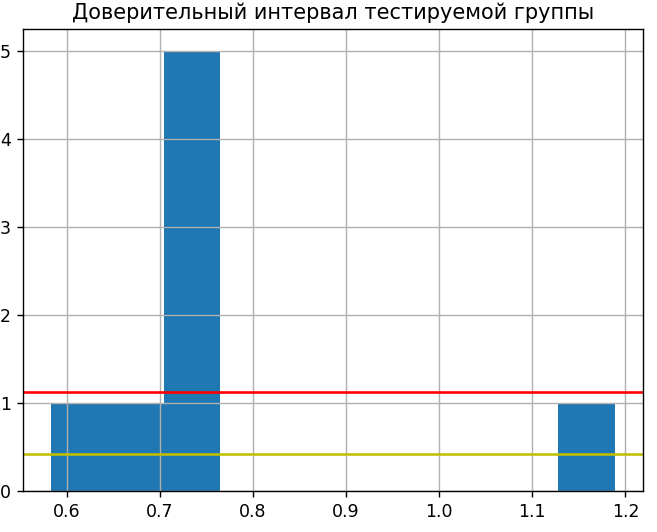
--------Дисперсия--------

0.032515412462320516

Доверительный интервал: (0.41697171053034443, 1.123814318221445)

Аномальное событие:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Group | Money | ARPU | anomaly |
| 17.07.2021 | test | 640910.68 | 1.188553 | 1 |



Доверительные интервалы (0.4176872245100419, 1.0400899826254055) и (0.41697171053034443, 1.123814318221445) пересекаются, это означает, что результаты случайны и акция не принесла результатов.

**Внутриигровые траты по группам теста:**

**import** pandas **as** pd  
**import** cheaters **as** ch  
  
df = pd.read\_csv(**'Cash.csv'**)  
abgroup = pd.read\_csv(**'ABgroup.csv'**)  
*### Очищаем данные по внутриигровым тратам от читеров ###*cash = pd.merge(df,ch.players, on=**'user\_id'**)  
*### Объединяем данные по внутриигровым тратам и данные по группам тестирования ###*df = pd.merge(cash,abgroup, on = **'user\_id'**)  
*### Сумма внутриигровых трат ###*cash\_gr = df[**'cash'**].groupby([df[**'group'**],df[**'date'**]]).sum()  
cash\_gr = cash\_gr.to\_csv(**'cash\_sum\_group.csv'**)

**Результат выполнения кода:**

group date

control 10.07.2021 3314070000

11.07.2021 3310449600

12.07.2021 3290581600

13.07.2021 3210397600

14.07.2021 3011168400

15.07.2021 2650527600

16.07.2021 2155123600

17.07.2021 4112335200

test 10.07.2021 3519510000

11.07.2021 3521359600

12.07.2021 3500207600

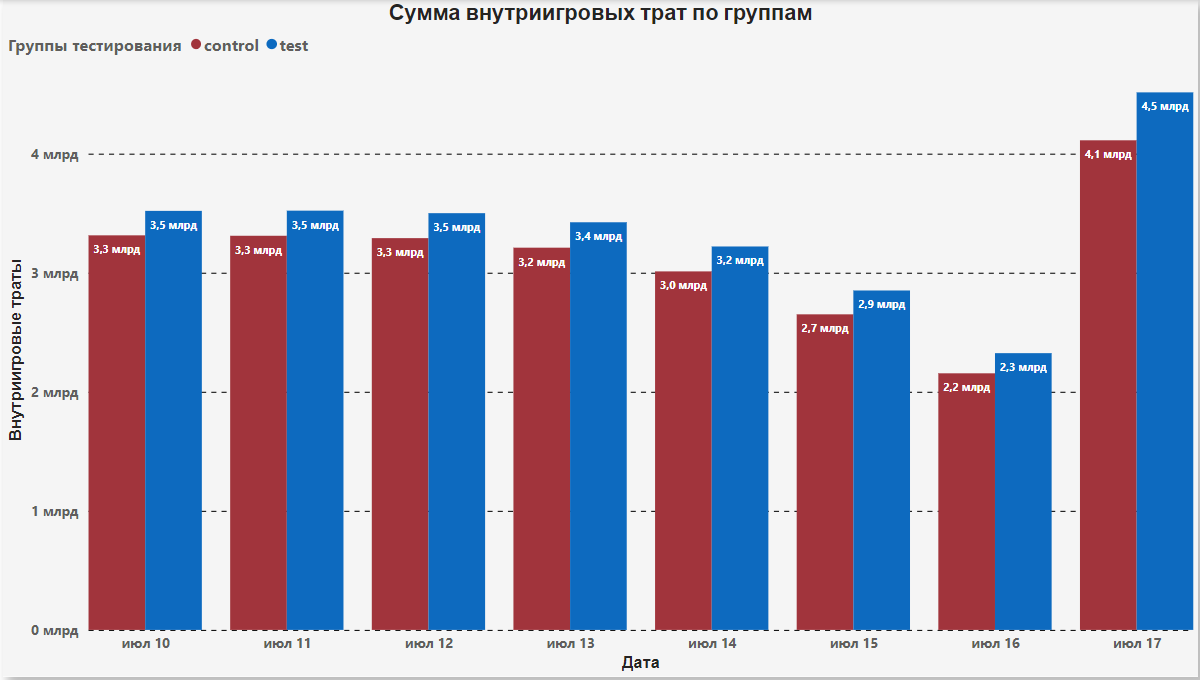
13.07.2021 3423822000

14.07.2021 3220481600

15.07.2021 2850800000

16.07.2021 2324441600

17.07.2021 4514888000



Тестовая группа незначительно больше тратила внутриигровую валют

**Внутриигровые траты по игровым платформам:**

**import** pandas **as** pd  
**import** cheaters **as** ch  
  
df = pd.read\_csv(**'Cash.csv'**)  
platform = pd.read\_csv(**'Platforms.csv'**)  
*### Очищаем данные по внутриигровым тратам от читеров ###*cash = pd.merge(df,ch.players, on=**'user\_id'**)  
*### Объединяем данные по внутриигровым тратам и данные по группам тестирования ###*df = pd.merge(cash,platform, on = **'user\_id'**)  
*### Сумма внутриигровых трат ###*cash\_gr = df[**'cash'**].groupby([df[**'platform'**],df[**'date'**]]).sum()  
  
cash\_gr = cash\_gr.to\_csv(**'cash\_sum\_platforms.csv'**)

**Результат выполнения кода:**

platform date

PC 10.07.2021 2298280800

11.07.2021 2296422800

12.07.2021 2284878000

13.07.2021 2227665200

14.07.2021 2081964400

15.07.2021 1827645200

16.07.2021 1473451200

17.07.2021 2721346000

PS4 10.07.2021 2228067600

11.07.2021 2226638800

12.07.2021 2212231600

13.07.2021 2161579600

14.07.2021 2030007600

15.07.2021 1798367200

16.07.2021 1465643200

17.07.2021 2847443600

XBox 10.07.2021 2307231600

11.07.2021 2308747600

12.07.2021 2293679600

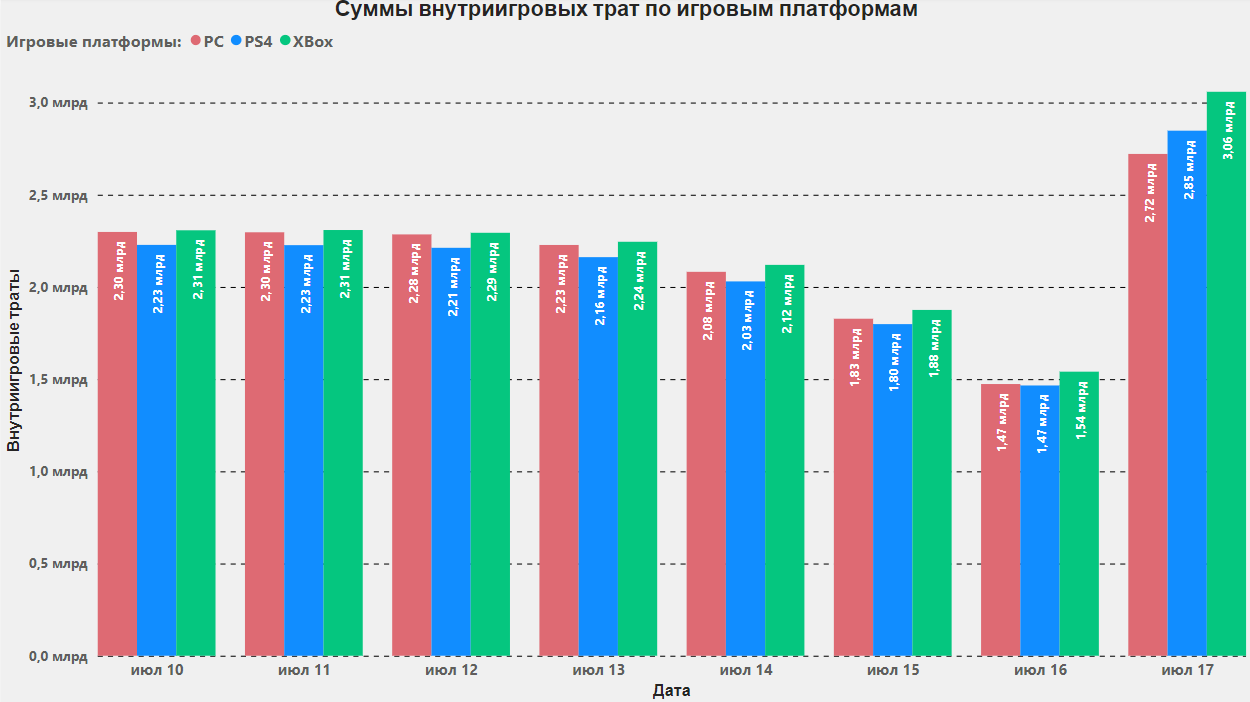
13.07.2021 2244974800

14.07.2021 2119678000

15.07.2021 1875315200

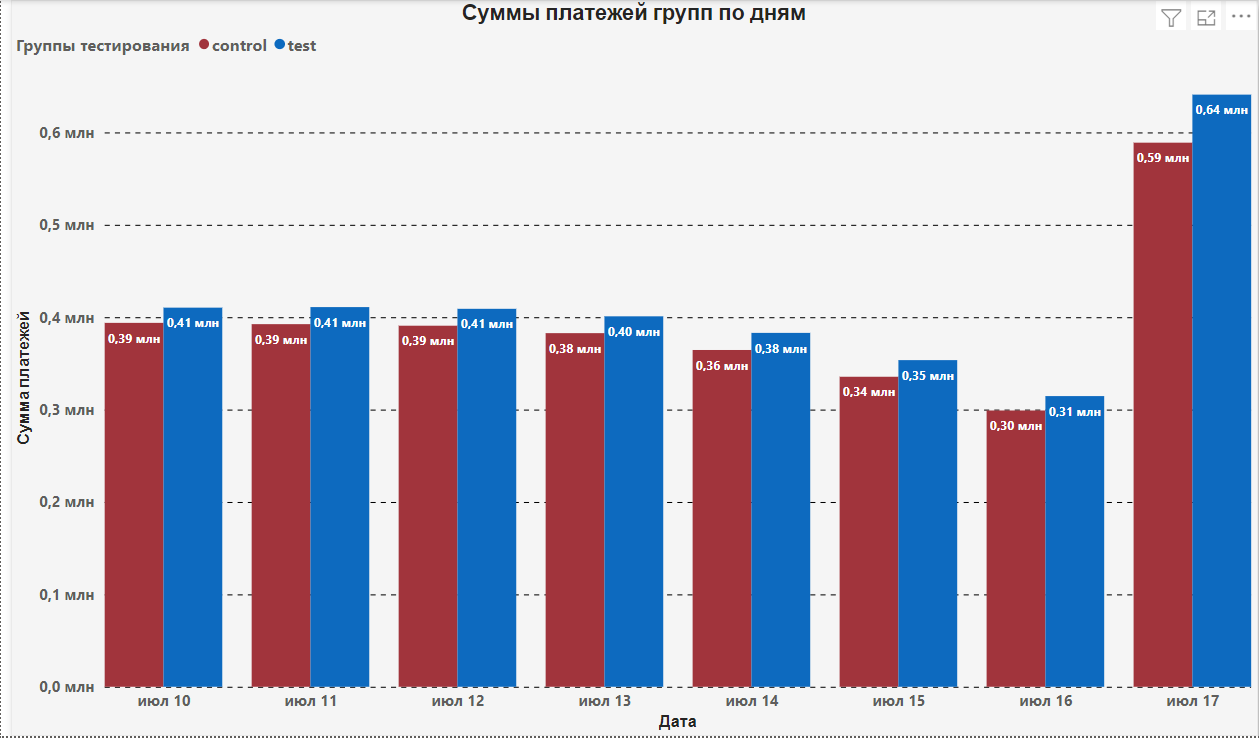
16.07.2021 1540470800

17.07.2021 3058433600



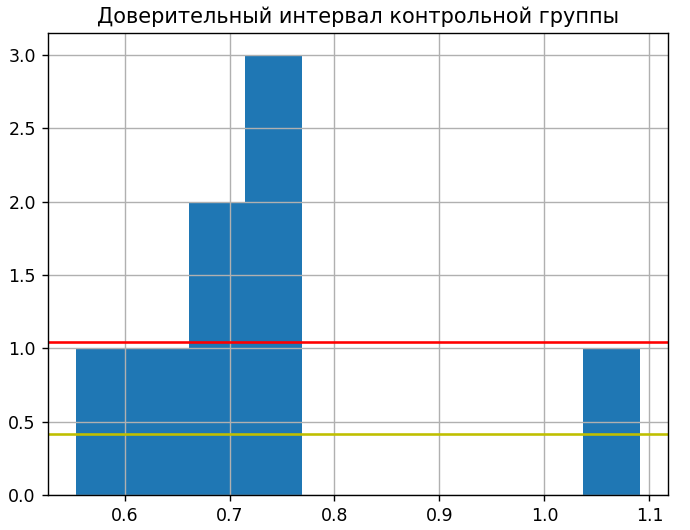
Больше всего внутриигровой валюты тратится на Xbox, а меньше всего на Ps4, но разница не существенная. Явный спад был в предпоследний день акции, а в последний день произошел всплеск трат.

**Выводы по проекту**

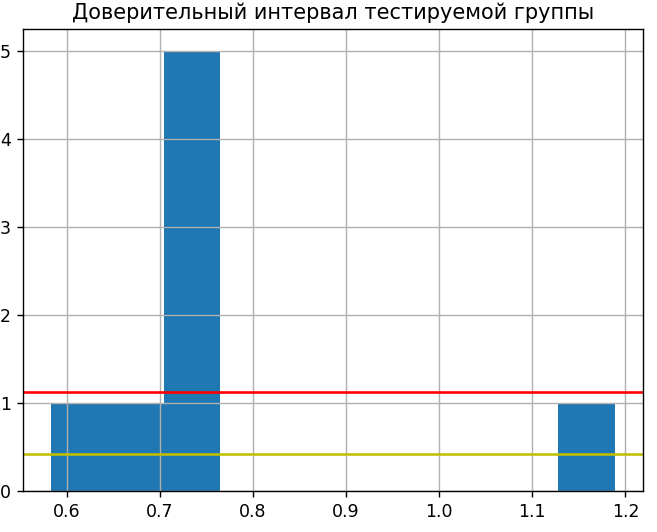


Смотря на данный график, мы видим, что акция не особо заинтересовала игроков. Средний прирост по платежам в день составляет 0.02 млн, или 5%.

1. Сравнивая доверительные интервалы контрольной группы (0.4176872245100419, 1.0400899826254055)



и тестируемой группы : (0.41697171053034443, 1.123814318221445)



Мы видим, что доверительные интервалы пересекаются, это означает, что результаты случайны и акция не принесла результатов.

Исходя из пунктов выше, можно сделать вывод, что акцию не стоит проводить повторно.