Финальная работа по курсу «Аналитик данных с

нуля 2.0»

Описание кейса:

Проведение A/B-теста.

Представьте, что вы работаете в компании, которая разрабатывает игры. Ваш основной хит - бесплатный командный онлайн-шутер. В игре есть внутриигровая валюта, которую вы можете выигрывать, побеждая в матчах, а можете покупать за настоящие деньги. На днях в игре прошёл A/B-тест - некоторые игроки могли приобрести премиумную броню по скидке. Ваше руководство хочет узнать, как это повлияло на ARPU (средняя прибыль на игрока), ARPPU (средняя прибыль на платящего игрока) и траты внутриигровой валюты.

Цель данного проекта заключается в том, чтобы выяснить, насколько проведение акции повлияло на финансовые показатели игры и стоит ли внедрять подобные акции в будущем.

Для успешного решения этой задачи буду использовать следующие шаги:

**Сбор и подготовка данных:**

* Загрузить данные из всех указанных таблиц (Money, Cheaters, Platforms, Cash, ABgroup).
* Объединить данные в единую структуру для удобного анализа.
* Провести предварительную очистку данных, исключая записи, связанные с читерами.

**Анализ метрик:**

* Рассчитать ARPU и ARPPU для тестовой и контрольной групп.
* Изучить динамику расходов внутриигровой валюты в обеих группах.
* Построить доверительные интервалы для средних значений ARPU и ARPPU.

**Визуализация данных:**

* Создать графики и сводную таблицу для наглядного представления изменений в метриках.

**Принятие решения:**

* Сравнить результаты тестовой и контрольной групп.
* Определить, была ли акция успешной и стоит ли ее продолжать.

1. **Сбор и подготовка данных.**

Так как данные представляют собой csv файлы, я буду использовать Visual Studio Code. Код загрузки данных выглядит так:

cheaters\_df = pd.read\_csv('Cheaters.csv')

platforms\_df = pd.read\_csv('Platforms.csv')

abgroup\_df = pd.read\_csv('ABgroup.csv')

money\_df = pd.read\_csv('Money.csv')

cash\_df = pd.read\_csv('Cash.csv')

Далее убедимся, что в датафреймах нет повторяющихся строк. Особенно это касается cheaters\_df, platform\_df и abgroup\_df так как данные должны быть уникальны. Для этого буду использовать метод drop\_duplicates из библиотеки Pandas.

cheaters\_unique = cheaters\_df.drop\_duplicates(subset=['user\_id'])

platforms\_unique = platforms\_df.drop\_duplicates(subset=['user\_id'])

abgroup\_unique = abgroup\_df.drop\_duplicates(subset=['user\_id'])

У нас есть список выявленных читеров, которых мы исключим из анализа. Однако могут быть и другие, незамеченные читеры. Чтобы их обнаружить, сначала проанализируем траты валюты и платежи уже известных читеров и сравним их с обычными игроками.

Аккаунты с читерами обычно тратят значительно больше валюты, чем обычные игроки, при этом не покупая её. Это и будет основой для поиска скрытых читеров.

При сравнении данных траты валют и совершенных платежей были замечены еще 344 подозрительных игрока, которые буду считать выявленными читерами.

1. **Анализ метрик**

Для расчета ARPU и ARPPU необходимо определить прибыль для каждой группы и разделить ее на количество пользователей в этой группе. ARPPU рассчитывается аналогично, но учитываются только пользователи, которые совершали платежи за валюту.

Доверительные интервалы можно найти с помощью библиотеки scipy. Для этого используются столбцы money и cash, которые содержат сумму платежей пользователя за период теста. Доверительная вероятность составляет 95%. Для расчета доверительных интервалов требуются среднее значение (µ) и стандартное отклонение (σ), а также значение доверительной вероятности. Среднее и стандартное отклонение можно найти с помощью методов Pandas: df[column].mean() и df[column].std().

Разделим игроков на группы и сразу добавим их платежи и платформу для этого выполним следующий код:

test\_group = players[players['group'] == 'test']\

.merge(money\_df, on = ['user\_id'])

control\_group = players[players['group'] == 'control']\

.merge(money\_df, on = ['user\_id'])

Создадим функцию для анализа данных:

def researh(df, column, color = 'blue', text = 'график распределения'):

    print("Базовые метрики")

    print(df[column].describe())

    print('Медиана - ', df[column].median())

    print("------------------------------------")

    print("Самые популярные значения метрики, топ 5")

    print(df[column].value\_counts().nlargest(5))

    print("------------------------------------")

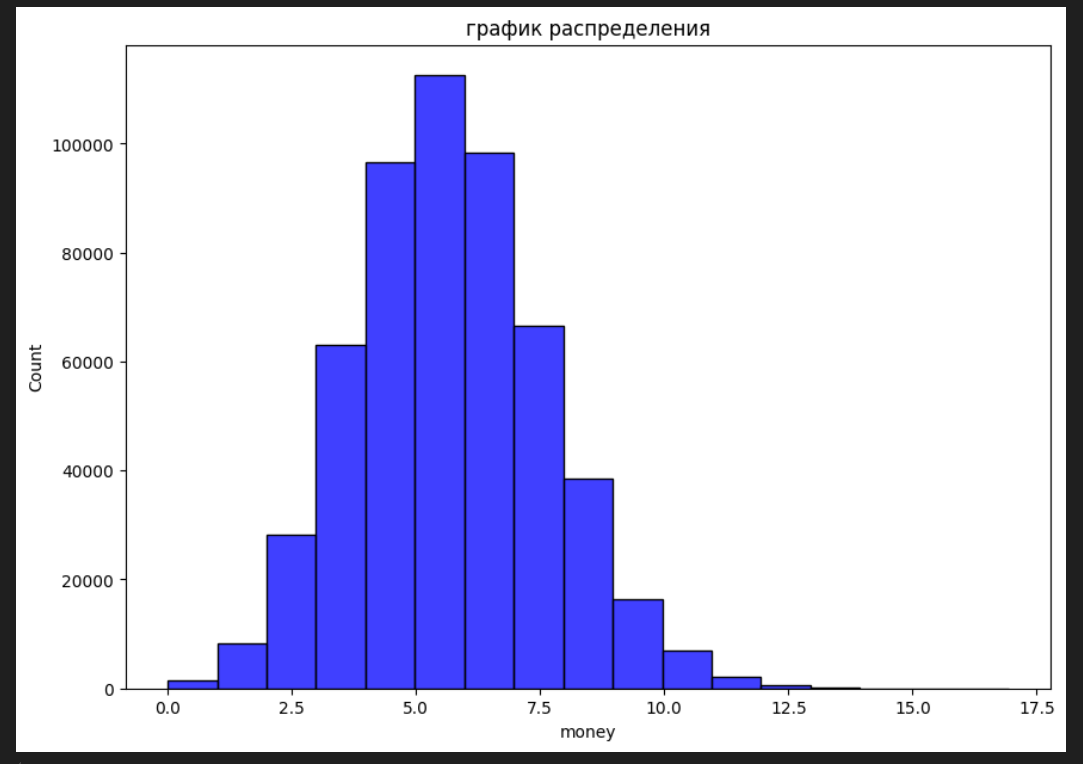
    print("Эксцесс ", kurtosis(df[column]))

    print("Ассиметрия ", skew(df[column]))

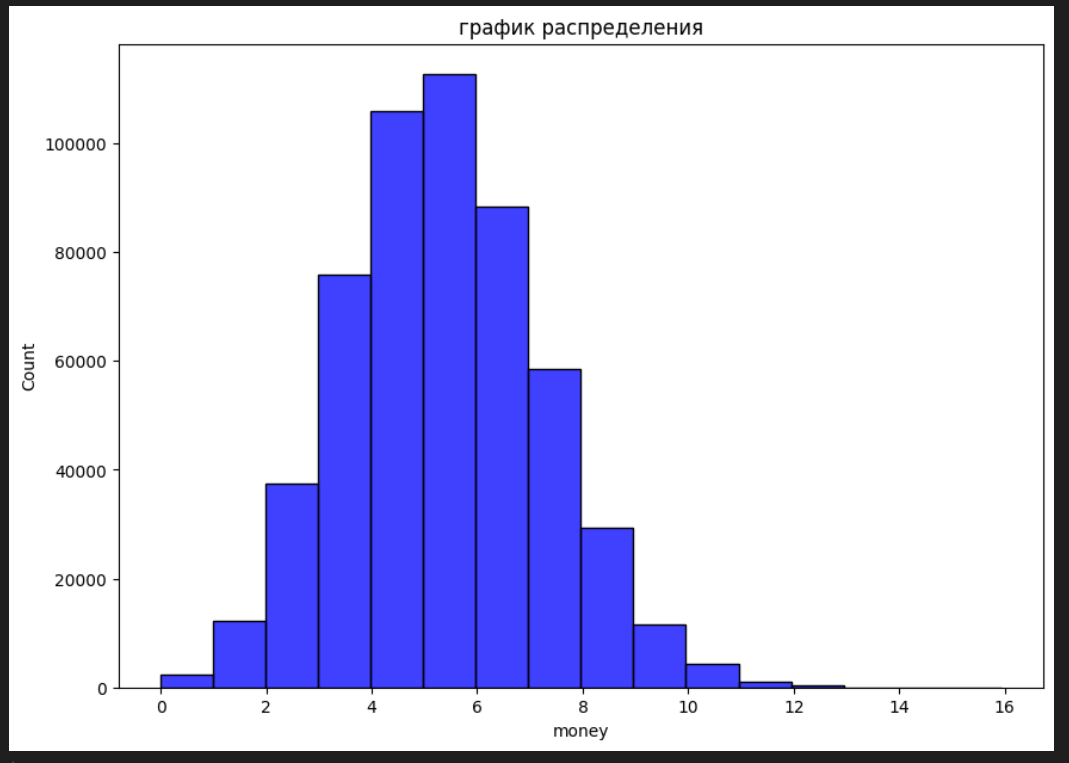
    plot = sb.histplot(df[column], color = color, bins=15,  binwidth= 1);

    plot.set\_title(text)

    plt.show()



Тестовая группа



Контрольная группа

Далее найдем суммы платежей для каждой группы:

print('Сумма платежей тестовой группы: ', test\_group\_sum := test\_group.money.sum())

print('Сумма платежей контрольной группы: ', control\_group\_sum := control\_group.money.sum())

Рассчитаем ARPU и ARPPU для каждой группы:

ARPU

ARPU\_test = test\_group\_sum / test\_group.drop\_duplicates('user\_id').count()[0]

ARPU\_control = control\_group\_sum / control\_group.drop\_duplicates('user\_id').count()[0]

print('ARPU тестовой группы - ', ARPU\_test, '\n', 'ARPU контрольной группы - ', ARPU\_control)

ARPU тестовой группы - 6.162275602990467

ARPU контрольной группы - 5.829478250563439

Найдем их доверительные интервалы возьмем доверительную вероятность 95%

def confidence(df, column, alpha = 0.95):

    interval = stats.norm.interval(alpha, loc=df[column].mean(), scale=df[column].std())

    return interval

test\_confidence = confidence(test\_group.groupby('user\_id').agg('sum'), 'money')

print('Доверительный интервал тестовой группы', test\_confidence)

control\_confidence = confidence(control\_group.groupby('user\_id').agg('sum'), 'money')

print('Доверительный интервал контрольной группы', control\_confidence)

Доверительный интервал тестовой группы (2.4170771425285396), (9.907474063452419)

Доверительный интервал контрольной группы (2.1572457079500182), (9.501710793176855)

Несмотря на то, что разница между группами существует, она невелика, причем доверительные интервалы практически полностью накладываются друг на друга. Это позволяет предположить, что полученный эффект скорее всего обусловлен случайностью.

print('ARPU тестовой группы больше контрольной на', round((ARPU\_test - ARPU\_control) / ARPU\_control \* 100, 1),'%')

ARPU тестовой группы больше контрольной на 5.7 %

ARPPU

Для этого сначала отделим платящих игроков:

df = test\_group.groupby('user\_id').agg('sum').reset\_index()

test\_group\_paying = df[df['money'] > 0]

df = control\_group.groupby('user\_id').agg('sum').reset\_index()

control\_group\_paying = df[df['money'] > 0]

ARPPU\_test = test\_group\_sum / test\_group\_paying.drop\_duplicates('user\_id').count()[0]

ARPPU\_control = control\_group\_sum / control\_group\_paying.drop\_duplicates('user\_id').count()[0]

print('ARPPU тестовой группы - ', ARPPU\_test, '\n', 'ARPPU контрольной группы - ', ARPPU\_control)

ARPPU тестовой группы - 6.163144115007149

ARPPU контрольной группы - 5.831108828541795

test\_paying\_confidence = confidence(test\_group\_paying.groupby('user\_id').agg('sum'), 'money')

print('Доверительный интервал тестовой группы', test\_paying\_confidence)

control\_paying\_confidence = confidence(control\_group\_paying.groupby('user\_id').agg('sum'), 'money')

print('Доверительный интервал контрольной группы', control\_paying\_confidence)

Доверительный интервал тестовой группы (2.4204277288673626), (9.905860501146952)

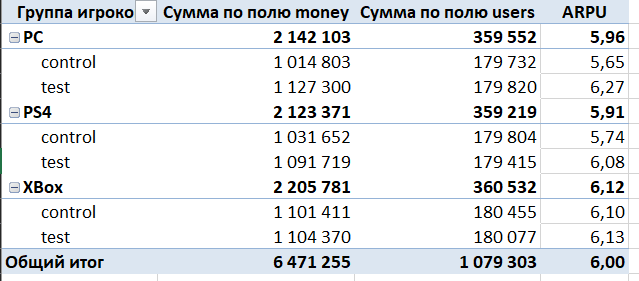
Доверительный интервал контрольной группы (2.163338538240738), (9.498879118842845)

print('ARPPU тестовой группы больше контрольной на', round((ARPPU\_test - ARPPU\_control) / ARPPU\_control \* 100, 1),'%')

ARPPU тестовой группы больше контрольной на 5.7 %

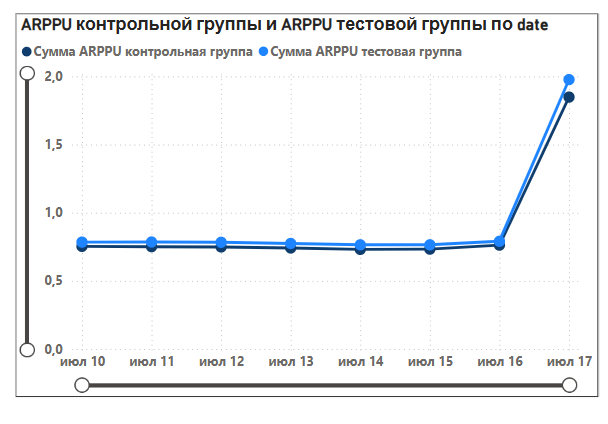
Доверительные интервалы перекрываются по обоим показателям, что свидетельствует о вероятной случайности полученных результатов. Возможно, акция оказывает влияние, однако оно незначительное и нестабильное.

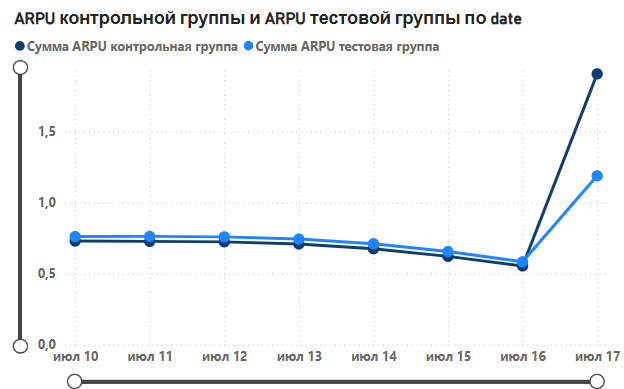
1. **Визуализация данных**



Анализ показал, что акция наиболее успешна для игроков на ПК и PlayStation 4. В то же время, для пользователей Xbox эффект от акции практически не наблюдается.

Графики из PowerBI:





Источники:

<https://pandas.pydata.org/pandas-docs/stable/reference/general_functions.html>

https://pandas.pydata.org/docs/reference

https://seaborn.pydata.org/

<https://ppt-online.org/566565>

<https://numpy.org/doc/stable/>

https://docs.yandex.ru/docs/view?tm=1633098627&tld=ru&lang=ru&name=Prakticheskaya-statistika-dlya-specialistov-Data-Science\_RuLit\_Me\_613729.pdf&text=%D0%BA%D0%BD%D0%B8%D0%B3%D0%B0%20%D0%BF%D0%BE%20%D1%81%D1%82%D0%B0%D1%82%D0%B8%D1%81%D1%82%D0%B8%D0%BA%D0%B8%20%D0%B4%D0%BB%D1%8F%20data%20science&url=https%3A%2F%2Fwww.rulit.me%2Fdata%2Fprograms%2Fresources%2Fpdf%2FPrakticheskaya-statistika-dlya-specialistov-Data-Science\_RuLit\_Me\_613729.pdf&lr=194&mime=pdf&l10n=ru&sign=47506b392ef67b14856852b3c1e8024a&keyno=0