

Приложения Apple Store и Google Play.

Исследование: Получили ли приложения Apple Store лучшие отзывы, чем приложения Google Play?

Этапы проекта

1. Загрузка данных

- * Загрузка двух наборов данных
- * Выберите столбцы, с которыми необходимо работать
- * Сгруппируйте данные на этой основе предыдущего пункта

2. Предобработка данных

- * Проверьте типы данных и исправьте их
- * Добавьте столбец platform в оба датарейма и Apple и Google
- * Измените имена столбцов для подготовки датафреймов к соединению
- * Объедините эти наборы данных
- * Удалите значения NaN
- * Отфильтровать только те приложения, которые были проверены хотя бы один раз
- * Проведите визуализацию данных (по столбцу) platform просмотрев данные также аналитически)

3. Формулировка гипотезы (Получили ли приложения Apple Store лучшие отзывы, чем приложения Google Play?)

- * Получить распределение данных
- * Проверка теста

Выводы

Импорт необходимых модулей и настройка рабочей области

Ввод [539]:

```
import numpy as np
import pandas as pd
import scipy
import seaborn as sns
import matplotlib.pyplot as plt
```

Загрузка данных и общая информация

APPLE

Ввод [4]:

```
ap = pd.read_csv('AppleStore.csv')
ap.info()
display(ap.head())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7197 entries, 0 to 7196
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   id               7197 non-null    int64  
 1   track_name       7197 non-null    object  
 2   size_bytes       7197 non-null    int64  
 3   currency         7197 non-null    object  
 4   price            7197 non-null    float64 
 5   rating_count_tot 7197 non-null    int64  
 6   rating_count_ver 7197 non-null    int64  
 7   user_rating      7197 non-null    float64 
 8   user_rating_ver 7197 non-null    float64 
 9   ver              7197 non-null    object  
 10  cont_rating      7197 non-null    object  
 11  prime_genre      7197 non-null    object  
 12  sup_devices.num  7197 non-null    int64  
 13  ipadSc_urls.num 7197 non-null    int64  
 14  lang.num         7197 non-null    int64  
 15  vpp_lic          7197 non-null    int64  
dtypes: float64(3), int64(8), object(5)
memory usage: 899.8+ KB
```

	id	track_name	size_bytes	currency	price	rating_count_tot	rating_count_ver	user
0	284882215	Facebook	389879808	USD	0.0	2974676		212
1	389801252	Instagram	113954816	USD	0.0	2161558		1289
2	529479190	Clash of Clans	116476928	USD	0.0	2130805		579
3	420009108	Temple Run	65921024	USD	0.0	1724546		3842
4	284035177	Pandora - Music & Radio	130242560	USD	0.0	1126879		3594

Ввод [8]:

```
print('Количество строк =', ap.shape[0])
print('Количество столбцов =', ap.shape[1])
```

Количество строк = 7197
Количество столбцов = 16

Датасет Apple содержит 7197 строк и 16 столбцов, пропущенные значения отсутствуют. Типы данных - float64(3), int64(8), object(5)

GOOGLE

Ввод [5]:

```
go = pd.read_csv('googleplaystore.csv')
go.info()
display(go.head())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10841 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   App              10841 non-null   object  
 1   Category         10841 non-null   object  
 2   Rating           9367 non-null   float64 
 3   Reviews          10841 non-null   object  
 4   Size              10841 non-null   object  
 5   Installs         10841 non-null   object  
 6   Type              10840 non-null   object  
 7   Price             10841 non-null   object  
 8   Content Rating  10840 non-null   object  
 9   Genres            10841 non-null   object  
 10  Last Updated    10841 non-null   object  
 11  Current Ver     10833 non-null   object  
 12  Android Ver     10838 non-null   object  
dtypes: float64(1), object(12)
memory usage: 1.1+ MB
```

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND DESIGN	4.1	159	19M	10,000+	Free	0	Everyone
1	Coloring book moana	ART_AND DESIGN	3.9	967	14M	500,000+	Free	0	Everyone
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	Everyone
3	Sketch - Draw & Paint	ART_AND DESIGN	4.5	215644	25M	50,000,000+	Free	0	Teen
4	Pixel Draw - Number Art Coloring Book	ART_AND DESIGN	4.3	967	2.8M	100,000+	Free	0	Everyone

Ввод [7]:

```
print('Количество строк =', go.shape[0])
print('Количество столбцов =', go.shape[1])
```

Количество строк = 10841
 Количество столбцов = 13

Датафрэйм Google содержит 10841 строк и 13 столбцов из которых в 5 столбцах есть пропущенные значения(NaN). Типы данных - строковый объект(12) и с плавающей точкой float64 в столбце 'Rating'

Выбор и переименование столбцов APPLE

Для целей данного исследования стоит отобрать только те столбцы, которые связывают приложения и их рейтинги. Создадим фрэйм apple, где переименуем столбцы.

Ввод [18]:

```
apple = ap.loc[:, ['track_name', 'price', 'rating_count_tot', 'user_rating', 'cont_rating',
apple.head()
```

Out[18]:

	track_name	price	rating_count_tot	user_rating	cont_rating	prime_genre
0	Facebook	0.0	2974676	3.5	4+	Social Networking
1	Instagram	0.0	2161558	4.5	12+	Photo & Video
2	Clash of Clans	0.0	2130805	4.5	9+	Games
3	Temple Run	0.0	1724546	4.5	9+	Games
4	Pandora - Music & Radio	0.0	1126879	4.0	12+	Music

Ввод [154]:

```
apple.set_axis(['app_name', 'price', 'total_reviews', 'rating', 'content', 'genre', 'platform'],
```

Добавим столбец 'platform' в датафрэйм apple

Ввод [20]:

```
apple['platform'] = 'apple'
```

Ввод [155]:

```
apple.columns
```

Out[155]:

```
Index(['app_name', 'price', 'total_reviews', 'rating', 'content', 'genre',
       'platform'],
      dtype='object')
```

Ввод [156]:

```
apple.head()
```

Out[156]:

	app_name	price	total_reviews	rating	content	genre	platform
0	Facebook	0.0	2974676	3.5	4+	Social Networking	apple
1	Instagram	0.0	2161558	4.5	12+	Photo & Video	apple
2	Clash of Clans	0.0	2130805	4.5	9+	Games	apple
3	Temple Run	0.0	1724546	4.5	9+	Games	apple
4	Pandora - Music & Radio	0.0	1126879	4.0	12+	Music	apple

Выбор и переименование столбцов Google

Ввод [306]:

```
google = go.loc[:, ['App', 'Price', 'Reviews', 'Rating', 'Content Rating', 'Category']]  
google.head()
```

Out[306]:

	App	Price	Reviews	Rating	Content Rating	Category
0	Photo Editor & Candy Camera & Grid & ScrapBook	0	159	4.1	Everyone	ART_AND DESIGN
1	Coloring book moana	0	967	3.9	Everyone	ART_AND DESIGN
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	0	87510	4.7	Everyone	ART_AND DESIGN
3	Sketch - Draw & Paint	0	215644	4.5	Teen	ART_AND DESIGN
4	Pixel Draw - Number Art Coloring Book	0	967	4.3	Everyone	ART_AND DESIGN

Добавим столбец 'platform' в датафрэйм google

Ввод [308]:

```
google['platform'] = 'google'
```

Ввод [309]:

```
google.set_axis(['app_name', 'price', 'total_reviews', 'rating', 'content', 'genre', 'platform'])
```

Ввод [310]:

```
google.columns
```

Out[310]:

```
Index(['app_name', 'price', 'total_reviews', 'rating', 'content', 'genre',
       'platform'],
      dtype='object')
```

Ввод [311]:

```
google.head()
```

Out[311]:

	app_name	price	total_reviews	rating	content	genre	platform
0	Photo Editor & Candy Camera & Grid & ScrapBook	0	159	4.1	Everyone	ART_AND DESIGN	google
1	Coloring book moana	0	967	3.9	Everyone	ART_AND DESIGN	google
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	0	87510	4.7	Everyone	ART_AND DESIGN	google
3	Sketch - Draw & Paint	0	215644	4.5	Teen	ART_AND DESIGN	google
4	Pixel Draw - Number Art Coloring Book	0	967	4.3	Everyone	ART_AND DESIGN	google

ПРЕДОБРАБОТКА ДАННЫХ

Проверьте типы данных и исправьте их

PRICE

Ввод [331]:

```
google.price.unique()
```

Out[331]:

```
array(['0', '$4.99', '$3.99', '$6.99', '$1.49', '$2.99', '$7.99', '$5.99',
       '$3.49', '$1.99', '$9.99', '$7.49', '$0.99', '$9.00', '$5.49',
       '$10.00', '$24.99', '$11.99', '$79.99', '$16.99', '$14.99',
       '$1.00', '$29.99', '$12.99', '$2.49', '$10.99', '$1.50', '$19.99',
       '$15.99', '$33.99', '$74.99', '$39.99', '$3.95', '$4.49', '$1.70',
       '$8.99', '$2.00', '$3.88', '$25.99', '$399.99', '$17.99',
       '$400.00', '$3.02', '$1.76', '$4.84', '$4.77', '$1.61', '$2.50',
       '$1.59', '$6.49', '$1.29', '$5.00', '$13.99', '$299.99', '$379.99',
       '$37.99', '$18.99', '$389.99', '$19.90', '$8.49', '$1.75',
       '$14.00', '$4.85', '$46.99', '$109.99', '$154.99', '$3.08',
       '$2.59', '$4.80', '$1.96', '$19.40', '$3.90', '$4.59', '$15.46',
       '$3.04', '$4.29', '$2.60', '$3.28', '$4.60', '$28.99', '$2.95',
       '$2.90', '$1.97', '$200.00', '$89.99', '$2.56', '$30.99', '$3.61',
       '$394.99', '$1.26', 'Everyone', '$1.20', '$1.04'], dtype=object)
```

Значения в столбце 'price_g' имеют строковый тип вида: знак доллар и следом цена, за исключением значений '0' и 'Everyone'. Удалим наблюдение с ценой 'Everyone' и для того чтобы при помощи срезов преобразовать тип данных во float, добавим к ячейкам '0' знак \$.

Ввод [340]:

```
google[google['price'] == 'Everyone']
```

Out[340]:

app_name	price	total_reviews	rating	content	genre	platform
----------	-------	---------------	--------	---------	-------	----------

Ввод [334]:

```
google.loc[google['price'] == '0', 'price'] = '$0'
```

Ввод [335]:

```
google['price'] = google['price'].apply(lambda x: x[1:])
```

Ввод [336]:

```
google['price'] = google['price'].astype('float')
google['total_reviews'] = google['total_reviews'].astype('int')
```

Ввод [337]:

```
google.dtypes
```

Out[337]:

```
app_name        object
price          float64
total_reviews   int32
rating         float64
content        object
genre          object
platform       object
dtype: object
```

Ввод [171]:

```
apple.dtypes
```

Out[171]:

```
app_name        object
price          float64
total_reviews   int64
rating         float64
content        object
genre          object
platform       object
dtype: object
```

Мы преобразовали тип данных столбца 'price' во float, а тип данных столбца 'total_reviews' в int

Объедините эти наборы данных

Ввод [312]:

```
df = pd.concat([apple, google], ignore_index=True)
display(df)
```

	app_name	price	total_reviews	rating	content	genre	platform
0	Facebook	0.0	2974676	3.5	4+	Social Networking	apple
1	Instagram	0.0	2161558	4.5	12+	Photo & Video	apple
2	Clash of Clans	0.0	2130805	4.5	9+	Games	apple
3	Temple Run	0.0	1724546	4.5	9+	Games	apple
4	Pandora - Music & Radio	0.0	1126879	4.0	12+	Music	apple
...
18033	Sya9a Maroc - FR	0	38	4.5	Everyone	FAMILY	google
18034	Fr. Mike Schmitz Audio Teachings	0	4	5.0	Everyone	FAMILY	google
18035	Parkinson Exercices FR	0	3	NaN	Everyone	MEDICAL	google
18036	The SCP Foundation DB fr nn5n	0	114	4.5	Mature 17+	BOOKS_AND_REFERENCE	google
18037	iHoroscope - 2018 Daily Horoscope & Astrology	0	398307	4.5	Everyone	LIFESTYLE	google

18038 rows × 7 columns

Ввод [499]:

```
#удаляем дубликаты
df = df.drop_duplicates().reset_index(drop=True)
print(df.duplicated().sum())
```

0

После объединения фреймов apple и google и удаления дубликатов получили результирующий датафрейм df с 17546 строками и 7 столбцами. Типы данных - float64(2), int64(1), object(4)

Удалите значения NaN

Ввод [315]:

df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17553 entries, 0 to 17552
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   app_name    17553 non-null   object  
 1   price       17553 non-null   object  
 2   total_reviews 17553 non-null   object  
 3   rating      16088 non-null   float64 
 4   content     17552 non-null   object  
 5   genre        17553 non-null   object  
 6   platform    17553 non-null   object  
dtypes: float64(1), object(6)
memory usage: 960.1+ KB
```

Ввод [316]:

print(f'Доля пропущенных значений в столбце "rating" составляет: {round((df.rating.isnull())

Доля пропущенных значений в столбце "rating" составляет: 8.3%, удалим их.

Ввод [317]:

df.dropna(subset=['rating'], inplace=True)

Ввод [318]:

print(f'После удаления пропущенных значений датафрэйм df содержит {df.shape[0]} строк и {df

После удаления пропущенных значений датафрэйм df содержит 16088 строк и 7 столбцов

Ввод []:

df.dropna(subset=['total_reviews'], inplace=True)

Ввод [503]:

df['total_reviews'] = df['total_reviews'].astype(int)

Отфильтровать только те приложения, которые были проверены хотя бы один раз

Ввод [341]:

df[df['total_reviews'] == 0]['total_reviews'].count()

Out[341]:

0

Количество приложений, отмеченных хотя бы раз составляет 929 штук. Исключим их из анализа.

Ввод [342]:

```
df = df[df['total_reviews'] != 0].reset_index(drop=True)
df
```

Out[342]:

	app_name	price	total_reviews	rating	content	genre	platform
0	Facebook	0.0	2974676	3.5	4+	Social Networking	apple
1	Instagram	0.0	2161558	4.5	12+	Photo & Video	apple
2	Clash of Clans	0.0	2130805	4.5	9+	Games	apple
3	Temple Run	0.0	1724546	4.5	9+	Games	apple
4	Pandora - Music & Radio	0.0	1126879	4.0	12+	Music	apple
...
15154	FR Calculator	0	7	4.0	0+	FAMILY	google
15155	Sya9a Maroc - FR	0	38	4.5	0+	FAMILY	google
15156	Fr. Mike Schmitz Audio Teachings	0	4	5.0	0+	FAMILY	google
15157	The SCP Foundation DB fr nn5n	0	114	4.5	17+	BOOKS_AND_REFERENCE	google
15158	iHoroscope - 2018 Daily Horoscope & Astrology	0	398307	4.5	0+	LIFESTYLE	google

15159 rows × 7 columns

CONTENT

Приведем к единообразию столбец 'content'

Ввод [343]:

```
df.content.unique()
```

Out[343]:

```
array(['4+', '12+', '9+', '17+', '0+', '10+', '18+', nan], dtype=object)
```

Ввод [323]:

```
df.loc[df['content'] == 'Everyone', 'content'] = '0+'
df.loc[df['content'] == 'Teen', 'content'] = '12+'
df.loc[df['content'] == 'Everyone 10+', 'content'] = '10+'
df.loc[df['content'] == 'Mature 17+', 'content'] = '17+'
df.loc[df['content'] == 'Adults only 18+', 'content'] = '18+'
```

Ввод [344]:

```
df.content.unique()
```

Out[344]:

```
array(['4+', '12+', '9+', '17+', '0+', '10+', '18+', nan], dtype=object)
```

Для того, чтобы восполнить пропущенное значение в ячейке 'Unrated', найдем наиболее часто встречаемый контент жанра Tools

Ввод [325]:

```
df[df['content'] == 'Unrated']
```

Out[325]:

	app_name	price	total_reviews	rating	content	genre	platform
13146	DC Universe Online Map	0	1186	4.1	Unrated	TOOLS	google

Ввод [327]:

```
df.query('genre == "TOOLS"')['content'].value_counts()
```

Out[327]:

```
0+      726
12+      5
17+      1
Unrated   1
Name: content, dtype: int64
```

Ввод [328]:

```
df.loc[df['content'] == 'Unrated', 'content'] = '0+'
```

Ввод [348]:

```
df[df.content.isnull()]
```

Out[348]:

	app_name	price	total_reviews	rating	content	genre	platform
14911	Life Made Wi-Fi Touchscreen Photo Frame	Everyone	3.0M	19.0	NaN	1.9	google

Ввод [349]:

```
df = df.dropna(subset=['content'])
```

Ввод [350]:

```
df.content.unique()
```

Out[350]:

```
array(['4+', '12+', '9+', '17+', '0+', '10+', '18+'], dtype=object)
```

Ввод [504]:

```
display(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 15158 entries, 0 to 15157
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype  
---  --  
 0   app_name         15158 non-null   object 
 1   price            15158 non-null   object 
 2   total_reviews    15158 non-null   int32  
 3   rating           15158 non-null   float64
 4   content          15158 non-null   object 
 5   genre             15158 non-null   object 
 6   platform          15158 non-null   object 
 7   test              15158 non-null   object 
dtypes: float64(1), int32(1), object(6)
memory usage: 1006.6+ KB
```

None

Привели к единообразию различные виды контента и избавились от последних пропущенных значений.

GENRE

Исследуем составные названия жанров данных с платформы google

Ввод [356]:

```
df.query('platform=="apple"').genre.unique()
```

Out[356]:

```
array(['Social Networking', 'Photo & Video', 'Games', 'Music',
       'Reference', 'Health & Fitness', 'Weather', 'Utilities', 'Travel',
       'Shopping', 'News', 'Navigation', 'Lifestyle', 'Entertainment',
       'Food & Drink', 'Sports', 'Book', 'Finance', 'Education',
       'Productivity', 'Business', 'Catalogs', 'Medical'], dtype=object)
```

Ввод [357]:

```
len(df.query('platform=="apple"').genre.unique())
```

Out[357]:

23

Ввод [373]:

```
df.query('platform=="google"').genre.unique()
```

Out[373]:

```
array(['ART_AND DESIGN', 'AUTO_AND VEHICLES', 'BEAUTY',
       'BOOKS_AND REFERENCE', 'BUSINESS', 'COMICS', 'COMMUNICATION',
       'DATING', 'EDUCATION', 'ENTERTAINMENT', 'EVENTS', 'FINANCE',
       'FOOD_AND DRINK', 'HEALTH_AND FITNESS', 'HOUSE_AND HOME',
       'LIBRARIES_AND DEMO', 'LIFESTYLE', 'GAME', 'FAMILY', 'MEDICAL',
       'SOCIAL', 'SHOPPING', 'PHOTOGRAPHY', 'SPORTS', 'TRAVEL_AND LOCAL',
       'TOOLS', 'PERSONALIZATION', 'PRODUCTIVITY', 'PARENTING', 'WEATHER',
       'VIDEO_PLAYERS', 'NEWS_AND MAGAZINES', 'MAPS_AND NAVIGATION'],
      dtype=object)
```

Ввод [358]:

```
len(df.query('platform=="google"').genre.unique())
```

Out[358]:

33

Попробуем привести жанры к единому образию. Изменим названия жанров apple (т.к. их меньше), чтобы они были в одном стиле с жанрами google.

Ввод [389]:

```
df.loc[df['genre'] == 'Book', 'genre'] = 'BOOKS_AND_REFERENCE'
df.loc[df['genre'] == 'Business', 'genre'] = 'BUSINESS'
df.loc[df['genre'] == 'Catalogs', 'genre'] = 'CATALOGS'
df.loc[df['genre'] == 'Education', 'genre'] = 'EDUCATION'
df.loc[df['genre'] == 'Entertainment', 'genre'] = 'ENTERTAINMENT'
df.loc[df['genre'] == 'Finance', 'genre'] = 'FINANCE'
df.loc[df['genre'] == 'Food & Drink', 'genre'] = 'FOOD_AND_DRINK'
df.loc[df['genre'] == 'Games', 'genre'] = 'GAME'
df.loc[df['genre'] == 'Health & Fitness', 'genre'] = 'HEALTH_AND_FITNESS'
df.loc[df['genre'] == 'Lifestyle', 'genre'] = 'LIFESTYLE'
df.loc[df['genre'] == 'Medical', 'genre'] = 'MEDICAL'
df.loc[df['genre'] == 'Music', 'genre'] = 'ENTERTAINMENT'
df.loc[df['genre'] == 'Navigation', 'genre'] = 'MAPS_AND_NAVIGATION'
df.loc[df['genre'] == 'News', 'genre'] = 'NEWS_AND_MAGAZINES'
df.loc[df['genre'] == 'Photo & Video', 'genre'] = 'PHOTOGRAPHY'
df.loc[df['genre'] == 'Productivity', 'genre'] = 'PRODUCTIVITY'
df.loc[df['genre'] == 'Reference', 'genre'] = 'BOOKS_AND_REFERENCE'
df.loc[df['genre'] == 'Shopping', 'genre'] = 'SHOPPING'
df.loc[df['genre'] == 'Social Networking', 'genre'] = 'SOCIAL'
df.loc[df['genre'] == 'Sports', 'genre'] = 'SPORTS'
df.loc[df['genre'] == 'Travel', 'genre'] = 'TRAVEL_AND_LOCAL'
df.loc[df['genre'] == 'Utilities', 'genre'] = 'TOOLS'
df.loc[df['genre'] == 'Weather', 'genre'] = 'WEATHER'
```

Изменим регистр в названии жанров, сделаем заглавной только первую букву:

Ввод [405]:

```
genre = list(df['genre'])
new_g = [str(i).title() for i in genre]
len(new_g)
for i in genre:
    i = str(i).title()
```

Ввод [406]:

```
df.genre.unique()
```

Out[406]:

```
array(['Social', 'Photography', 'Game', 'Entertainment',
       'Books_And_Reference', 'Health_And_Fitness', 'Weather', 'Tools',
       'Travel_And_Local', 'Shopping', 'News_And_Magazines',
       'Maps_And_Navigation', 'Lifestyle', 'Food_And_Drink', 'Sports',
       'Finance', 'Education', 'Productivity', 'Business', 'Catalogs',
       'Medical', 'Art_And_Design', 'Auto_And_Vehicles', 'Beauty',
       'Comics', 'Communication', 'Dating', 'Events', 'House_And_Home',
       'Libraries_And_Demo', 'Family', 'Personalization', 'Parenting',
       'Video_Players'], dtype=object)
```

Привели жанры к одному стилю, а также переименовали. Теперь их можно сравнивать.

RATING

Ввод [444]:

```
df.groupby('platform')['rating'].value_counts().unstack()
```

Out[444]:

rating	1.0	1.2	1.4	1.5	1.6	1.7	1.8	1.9	2.0	2.1	...	4.1	4.2	4.3
platform														
apple	44.0	NaN	NaN	56.0	NaN	NaN	NaN	NaN	106.0	NaN	...	NaN	NaN	NaN
google	16.0	1.0	3.0	3.0	4.0	8.0	8.0	12.0	12.0	8.0	...	656.0	887.0	1016.0

2 rows × 15 columns

Ввод [446]:

```
df.query('platform == "apple"]').rating.unique()
```

Out[446]:

```
array([3.5, 4.5, 4., 3., 5., 2.5, 2., 1.5, 1.])
```

Ввод [450]:

```
df.query('platform == "google"]').rating.unique()
```

Out[450]:

```
array([4.1, 3.9, 4.7, 4.5, 4.3, 4.4, 3.8, 4.2, 4.6, 3.2, 4., 4.8, 4.9,
       3.6, 3.7, 3.3, 3.4, 3.5, 3.1, 5., 2.6, 3., 1.9, 2.5, 2.8, 2.7,
       1., 2.9, 2.3, 2.2, 1.7, 2., 1.8, 2.4, 1.6, 2.1, 1.4, 1.5, 1.2])
```

Интуитивно кажется, что для сравнения данных необходимо привести деления шкал рейтингов к одному стилю. Но видимо, задачи данного исследования этого не предусматривают. Оставим все как есть.

TOTAL_REVIEWS

Ввод [489]:

```
df.total_reviews.info()
```

```
<class 'pandas.core.series.Series'>
Int64Index: 15158 entries, 0 to 15158
Series name: total_reviews
Non-Null Count Dtype
-----
15158 non-null object
dtypes: object(1)
memory usage: 236.8+ KB
```

Тип данных столбца с кол-вом голосов имеет строковый тип. Изменим его на int

Ввод [493]:

```
df.loc['total_reviews'] = df['total_reviews'].astype('int', errors='ignore')
```

C:\Users\Лунтик\AppData\Local\Temp\ipykernel_20344\1267319619.py:1: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

```
df.loc['total_reviews'] = df['total_reviews'].astype('int', errors='ignore')
```

ИССЛЕДОВАТЕЛЬСКИЙ АНАЛИЗ ДАННЫХ

Проведите визуализацию данных (по столбцу platform), просмотрев данные также аналитически

Ввод [506]:

```
df.query('platform=="apple"').describe()
```

Out[506]:

	total_reviews	rating
count	6.268000e+03	6268.000000
mean	1.480381e+04	4.049697
std	8.098468e+04	0.726943
min	1.000000e+00	1.000000
25%	7.800000e+01	4.000000
50%	5.125000e+02	4.500000
75%	3.963500e+03	4.500000
max	2.974676e+06	5.000000

Ввод [507]:

```
df.query('platform=="google"').describe()
```

Out[507]:

	total_reviews	rating
count	8.890000e+03	8890.000000
mean	4.728599e+05	4.187885
std	2.905373e+06	0.522435
min	1.000000e+00	1.000000
25%	1.640000e+02	4.000000
50%	4.708000e+03	4.300000
75%	7.119725e+04	4.500000
max	7.815831e+07	5.000000

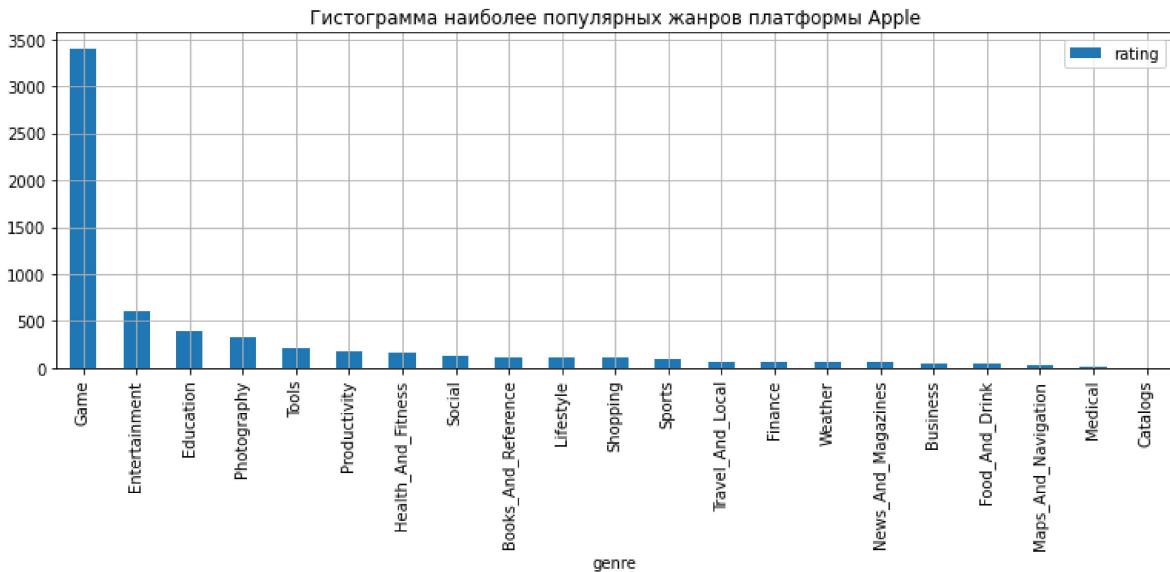
Определим наиболее популярные ЖАНРЫ приложений:

Ввод [409]:

```
(df
    .query('platform=="apple"')
    .pivot_table(index='genre', values='rating', aggfunc='count').sort_values(by='rating',
    .plot(grid=True, kind='bar', figsize=(13, 4))
)
plt.title('Гистограмма наиболее популярных жанров платформы Apple')
```

Out[409]:

Text(0.5, 1.0, 'Гистограмма наиболее популярных жанров платформы Apple')



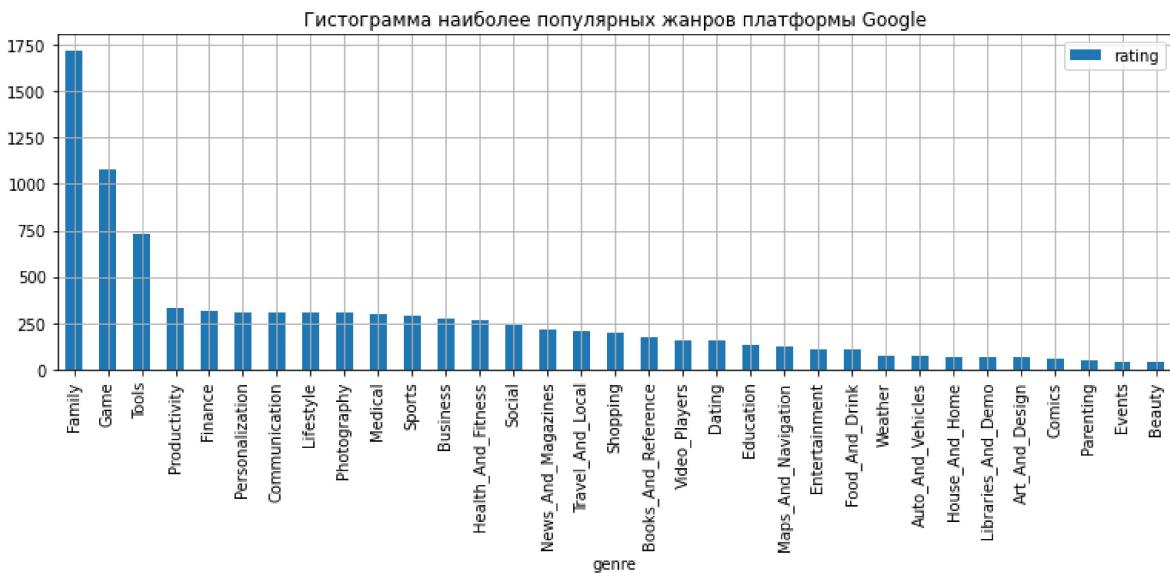
В apple наиболее популярны приложения жанра Game, их оценивают в почти в 7 раз чаще, чем ближайший жанр Развлечения.

Ввод [464]:

```
(df
    .query('platform=="google"')
    .pivot_table(index='genre', values='rating', aggfunc='count').sort_values(by='rating',
    .plot(grid=True, kind='bar', figsize=(13, 4))
)
plt.title('Гистограмма наиболее популярных жанров платформы Google')
```

Out[464]:

Text(0.5, 1.0, 'Гистограмма наиболее популярных жанров платформы Google')



В google наиболее популярны приложения жанра Family, а Game находится на втором месте с кол-вом голосов чуть больше 1 тыс и на третьем - Tools. Остальные жанры оценивали примерно одинаково и не так часто (голосов меньше 300).

Определим жанры приложений с наибольшим средним рейтингом:

Ввод [467]:

```
df.pivot_table(index=['platform', 'genre'], values='rating', aggfunc=np.mean).sort_values(by
```

Out[467]:

platform	genre	rating
	Events	4.435556
	Education	4.375969
	Art_And_Design	4.358065
	Books_And_Reference	4.347458
	Personalization	4.333871
	Parenting	4.300000
google	Game	4.281285
	Beauty	4.278571
	Health_And_Fitness	4.261450
	Social	4.254918
	Shopping	4.252239
	Weather	4.244000
	Sports	4.225175
apple	Books_And_Reference	4.224576
google	Productivity	4.201796
apple	Catalogs	4.200000
google	Family	4.191153
	Auto_And_Vehicles	4.190411
apple	Health_And_Fitness	4.188679
	Game	4.185735

Рассмотрим подробней жанр Games

Ввод [466]:

```
df.loc[df.genre.str.contains('Game')].groupby(['platform', 'genre'])['rating'].agg(['std',
```

Out[466]:

platform	genre	std	mean	var	median	count
apple	Game	0.593298	4.185735	0.352003	4.5	3400
google	Game	0.367117	4.281285	0.134775	4.3	1074

Приложения жанра Game платформы apple оценивают в 3,5 раза чаще, чем google. Средний рейтинг игр у google выше, при этом медианна среднего рейтинга больше у apple.

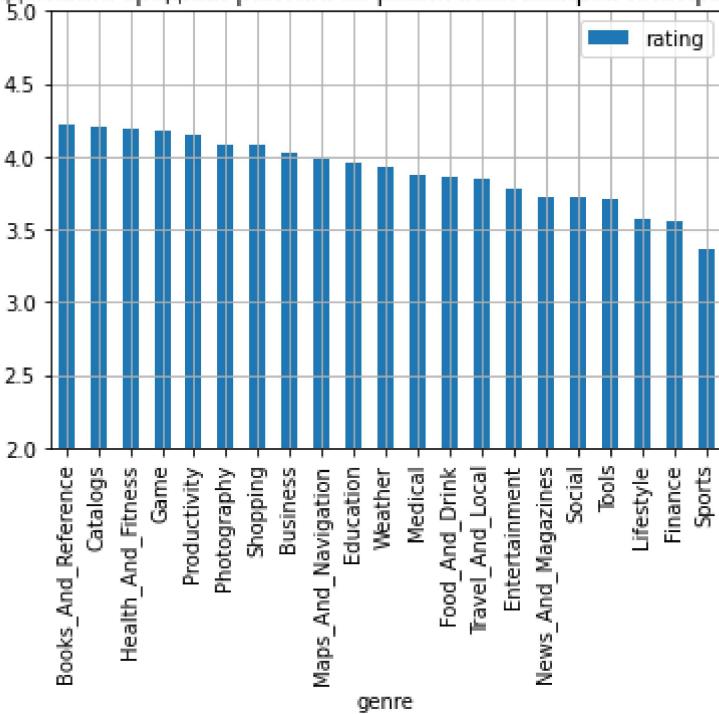
Ввод [475]:

```
(df
    .query('platform=="apple"')
    .pivot_table(index='genre', values='rating', aggfunc=np.mean).sort_values(by='rating',
    .plot(grid=True, kind='bar'))
)
plt.title('Распределение средних рейтингов различных жанров платформы Apple')
plt.ylim(2,5)
```

Out[475]:

(2.0, 5.0)

Распределение средних рейтингов различных жанров платформы Apple



Ввод [477]:

```
(df
    .query('platform=="google"')
    .pivot_table(index='genre', values='rating', aggfunc=np.mean).sort_values(by='rating',
    .plot(grid=True, kind='bar', figsize=(10, 4))
)
plt.title('Распределение средних рейтингов различных жанров платформы Google')
plt.ylim(2,5)
```

Out[477]:

(2.0, 5.0)



В среднем рейтинг жанров приложений google выше, чем у apple.

Получить распределение данных

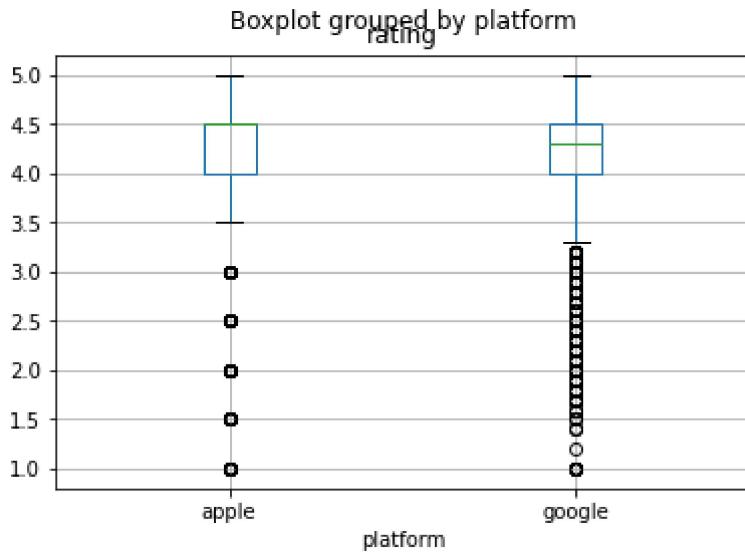
RATING

Ввод [532]:

```
df.boxplot(column='rating', by='platform')
```

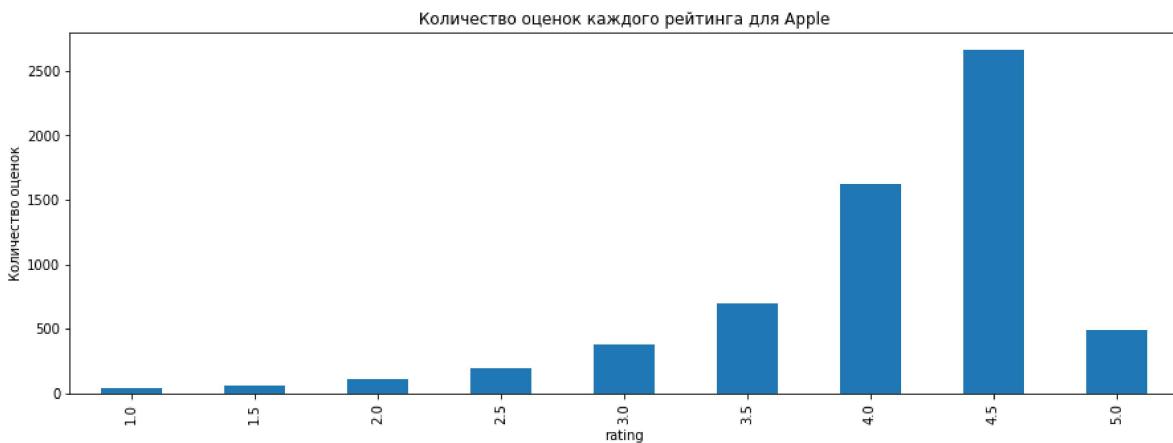
Out[532]:

```
<AxesSubplot:title={'center':'rating'}, xlabel='platform'>
```



Ввод [508]:

```
fig, ax = plt.subplots(figsize=(15,5))
plt.xlabel("Рейтинг")
plt.ylabel("Количество оценок")
plt.title("Количество оценок каждого рейтинга для Apple")
df.query('platform=="apple"').groupby(['rating']).count()['total_reviews'].plot(ax=ax, kind='bar')
plt.show()
```

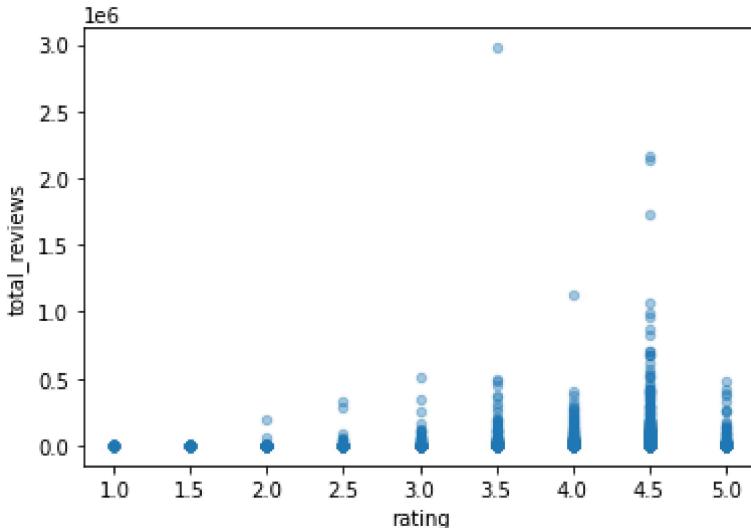


Ввод [526]:

```
df.query('platform=="apple"').plot(kind='scatter', x='rating', y='total_reviews', alpha=0.4)
```

Out[526]:

```
<AxesSubplot:xlabel='rating', ylabel='total_reviews'>
```



Распределение рейтинга apple не похоже на нормальное.

Ввод [519]:

```
fig, ax = plt.subplots(figsize=(15,5))
plt.xlabel("Рейтинг")
plt.ylabel("Количество оценок")
plt.title("Количество оценок каждого рейтинга для Google")
df.query('platform=="google"').groupby(['rating']).count()['total_reviews'].plot(ax=ax, kind='bar')
plt.show()
```



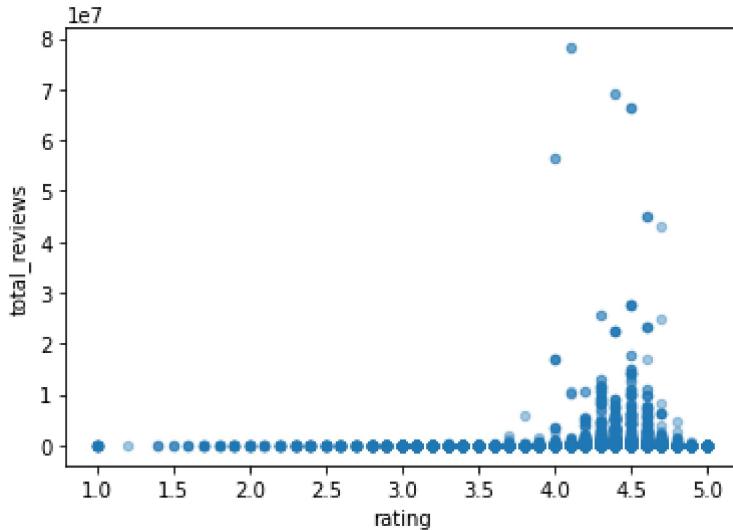
Распределение в рейтинге google выглядит нормальным примерно с 3.7 до 4.7 баллов. При значении рейтинга 4.8 происходит резкое снижение кол-ва оценок, значений 4.9 еще меньше. Однако кол-во максимально возможных баллов 5.0 следом возрастает.

Ввод [524]:

```
df.query('platform=="google"').plot(kind='scatter', x='rating', y='total_reviews', alpha=0.
```

Out[524]:

```
<AxesSubplot:xlabel='rating', ylabel='total_reviews'>
```



Ввод [527]:

```
len(set(df.query('(platform=="apple") & (genre=="Game")')['app_name']) & set(df.query('(pla
```

Out[527]:

```
107
```

3. Проверка гипотез

Н0: наблюдаемая разница в среднем рейтинге приложений Apple Store и Google Play обусловлена случайностью (а не платформой);

Н1: наблюдаемая разница в средних рейтингах пользователей Apple и Google обусловлена не случайностью (а фактически обусловлена платформой).

Пороговое значение alpha (уровень статистической значимости) выбираем равным 5%, т.е вероятность отвергнуть нулевую гипотезу когда она верна = 5/100 = 1/20.

Ввод [541]:

```
# Create a subset of the column 'Rating' by the different platforms.
# Call the subsets 'apple' and 'google'
apple_rating = df[df['platform'] == 'apple']['rating']
google_rating = df[df['platform'] == 'google']['rating']
```

Исследуем распределение рейтинга приложений google на нормальность

Ввод [544]:

```
value, p = scipy.stats.normaltest(google_rating)
print(value, p)
if p >= 0.01:
    print('Не вероятно, что признак распределен нормально.')
else:
    print('Значения рейтинга распределены НЕ нормально.')
```

3425.6074044257534 0.0

Значения рейтинга распределены НЕ нормально.

Ввод [543]:

```
if np.var(apple_rating)==np.var(google_rating):
    print('True')
else:
    print('False')
```

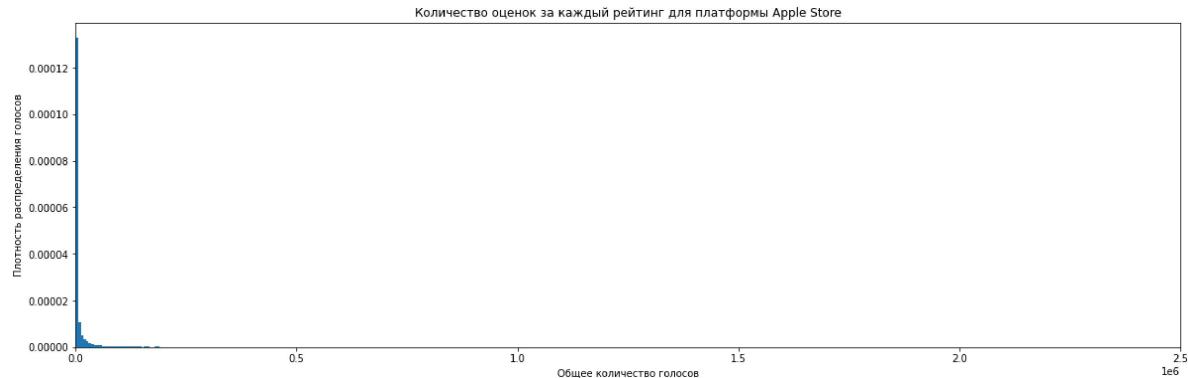
False

Дисперсии рейтинга платформ **НЕ** равны. Визуально глядя на гистограммы, приведенные выше, мы также можем опровергнуть предположение о нормальности распределения данных рейтинга платформ.

Рассмотрим распределения по платформам общего количества голосов за приложения

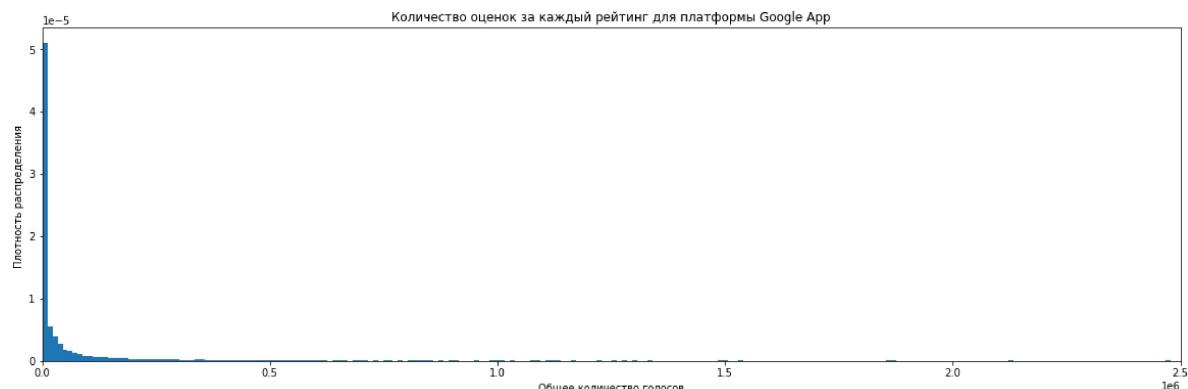
Ввод [603]:

```
fig, ax = plt.subplots(figsize=(20,6))
plt.xlabel("Общее количество голосов")
plt.ylabel("Плотность распределения голосов")
plt.title("Количество оценок за каждый рейтинг для платформы Apple Store")
plt.hist(df.query('platform=="apple"')['total_reviews'], bins=500, density=True)
plt.xlim(0,2500000)
plt.show()
```



Ввод [602]:

```
fig, ax = plt.subplots(figsize=(20,6))
plt.xlabel("Общее количество голосов")
plt.ylabel("Плотность распределения ")
plt.title("Количество оценок за каждый рейтинг для платформы Google App")
plt.hist(df.query('platform=="google"')['total_reviews'], bins=7000, density=True)
plt.xlim(0,2500000)
plt.show()
```



Проверка теста

Поскольку распределение данных не является нормальным, применим здесь непараметрический тест. Это просто метка для статистических тестов, используемых, когда данные обычно не распределены. Эти тесты чрезвычайно мощны из-за того, как мало предположений нам нужно сделать.

Ввод [606]:

```
# Создадим колонку `Permutation1` и поместим туда результат перемешивания значений рейтинга
df['Permutation1'] = np.random.permutation(df['rating'])

# Вызовем метод describe() для наших перемешанных данных и сгруппируем их по платформам.
df['Permutation1'].groupby(by=df['platform']).describe()
```

Out[606]:

	count	mean	std	min	25%	50%	75%	max
platform								
apple	6268.0	4.135195	0.615996	1.0	4.0	4.3	4.5	5.0
google	8890.0	4.127604	0.621174	1.0	4.0	4.3	4.5	5.0

Ввод [607]:

```
# Давайте сравним эти данные с предыдущими аналитическими выводами:
df.groupby(by='platform')['rating'].describe()
```

Out[607]:

	count	mean	std	min	25%	50%	75%	max
platform								
apple	6268.0	4.049697	0.726943	1.0	4.0	4.5	4.5	5.0
google	8890.0	4.187885	0.522435	1.0	4.0	4.3	4.5	5.0

Ввод [608]:

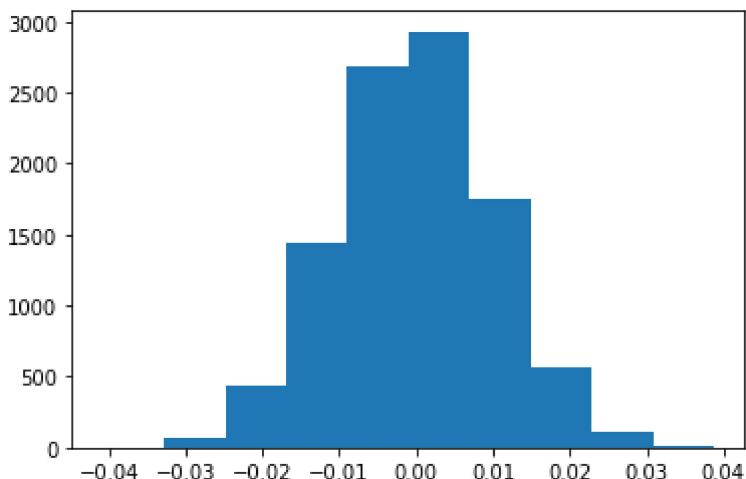
```
# The difference in the means for Permutation1 (0.001103) now looks hugely different to our
# It's sure starting to look like our observed difference is significant, and that the Null
# But to be sure, let's create 10,000 permutations, calculate the mean ratings for Google and
# Let's create a vector with the differences - that will be the distribution of the Null.

# First, make a list called difference.
difference = []

# Now make a for loop that does the following 10,000 times:
# 1. makes a permutation of the 'Rating' as you did above
# 2. calculates the difference in the mean rating for apple and the mean rating for google.
for i in range(10000):
    permutation = np.random.permutation(df['rating'])
    difference.append(np.mean(permutation[df['platform']=='apple']) - np.mean(permutation[df['platform']=='google']))
```

Ввод [609]:

```
# Make a variable called 'histo', and assign to it the result of plotting a histogram of the
histo = plt.hist(difference)
```



Ввод [611]:

```
# Now make a variable called obs_difference, and assign it the result of the mean of our 'a
obs_difference = df['rating'].groupby(df['platform']=='apple').mean()-df['rating'].groupby(
# Make this difference absolute with the built-in abs() function.
obs_difference =abs(obs_difference)

# Print out this value; it should be 0.1420605474512291.
print(obs_difference)
```

```
platform
False    0.138188
True     0.138188
Name: rating, dtype: float64
```

Выходы

Каков наш вывод?

Ввод [617]:

```
'''
What do we know?

Recall: The p-value of our observed data is just the proportion of the data given the null
As a result, we're going to count how many of the differences in our difference list are at
If less than or equal to 5% of them are, then we will reject the Null.
'''

# positiveExtremes = []
# negativeExtremes = []
# for i in range(len(difference)):
#     if (difference[i] >= obs_difference):
#         positiveExtremes.append(difference[i])
#     elif (difference[i] <= -obs_difference):
#         negativeExtremes.append(difference[i])

# print(len(positiveExtremes))
# print(len(negativeExtremes))
len(difference)
```

Out[617]:

10000

Каково наше решение?¶

Получается, что нулевые различия как минимум столь же экстремальны, как и наблюдаемые нами различия!

Таким образом, р-значение наших наблюдаемых данных равно 0.

Не имеет значения, какой уровень значимости мы выберем; наши наблюдаемые данные статистически значимы, и мы отвергаем H_0 .

Мы пришли к выводу, что платформа действительно влияет на рейтинги. В частности, мы должны посоветовать нашему клиенту интегрировать только Google Play в интерфейс своей операционной системы.

Type *Markdown* and \LaTeX : α^2