Яндекс Музыка

Сравнение Москвы и Петербурга окружено мифами. Например:

- Москва — мегаполис, подчинённый жёсткому ритму рабочей недели;
- Петербург — культурная столица, со своими вкусами.

На данных Яндекс Музыки вы сравните поведение пользователей двух столиц.

Цель исследования — проверьте три гипотезы:

1. Активность пользователей зависит от дня недели. Причём в Москве и Петербурге это проявляется по-разному.
2. В понедельник утром в Москве преобладают одни жанры, а в Петербурге — другие. Так же и вечером пятницы преобладают разные жанры — в зависимости от города.
3. Москва и Петербург предпочитают разные жанры музыки. В Москве чаще слушают поп-музыку, в Петербурге — русский рэп.

Ход исследования

Данные о поведении пользователей вы получите из файла `yandex_music_project.csv` . О качестве данных ничего не известно. Поэтому перед проверкой гипотез понадобится обзор данных.

Вы проверите данные на ошибки и оцените их влияние на исследование. Затем, на этапе предобработки вы поищете возможность исправить самые критичные ошибки данных.

Таким образом, исследование пройдёт в три этапа:

1. Обзор данных.
2. Предобработка данных.
3. Проверка гипотез.

Обзор данных

Составьте первое представление о данных Яндекс Музыки.

Задание 1

In [2]:
```
import pandas as pd
# импорт библиотеки pandas
```

Задание 2

In [3]:
```
df = pd.read_csv('/datasets/yandex_music_project.csv')
# чтение файла с данными и сохранение в df
```

Задание 3

In [4]:
```
print(df.head(10))
# получение первых 10 строк таблицы df
```

```
     userID                        Track            artist   genre  \
0  FFB692EC            Kamigata To Boots  The Mass Missile    rock
1  55204538  Delayed Because of Accident  Andreas Rönnberg    rock
2    20EC38            Funiculì funiculà      Mario Lanza     pop
3  A3DD03C9        Dragons in the Sunset        Fire + Ice    folk
4  E2DC1FAE                   Soul People        Space Echo   dance
5  842029A1                     Преданная         IMPERVTOR  rusrap
6  4CB90AA5                          True     Roman Messer   dance
7  F03E1C1F             Feeling This Way   Polina Griffith   dance
8  8FA1D3BE     И вновь продолжается бой               NaN  ruspop
9  E772D5C0                     Pessimist               NaN   dance

            City        time        Day
0  Saint-Petersburg  20:28:33  Wednesday
1            Moscow  14:07:09     Friday
2  Saint-Petersburg  20:58:07  Wednesday
3  Saint-Petersburg  08:37:09     Monday
4            Moscow  08:34:34     Monday
5  Saint-Petersburg  13:09:41     Friday
6            Moscow  13:00:07  Wednesday
7            Moscow  20:47:49  Wednesday
8            Moscow  09:17:40     Friday
9  Saint-Petersburg  21:20:49  Wednesday
```

Задание 4

In [5]:
```
df.info()
# получение общей информации о данных в таблице df
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 65079 entries, 0 to 65078
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0     userID  65079 non-null  object
 1   Track     63848 non-null  object
 2   artist    57876 non-null  object
 3   genre     63881 non-null  object
```

```
 4     City     65079 non-null  object
 5    time      65079 non-null  object
 6    Day       65079 non-null  object
dtypes: object(7)
memory usage: 3.5+ MB
```

Количество значений в столбцах различается. Значит, в данных есть пропущенные значения.

Задание 5

In [6]:
```
# 1. Заголовки столбцов указаны неодинаково: присутствуют лишние пробелы перед именем и разный регистр:
# заголовки - userID,Track,City,Day.
# 2. Предположительно, в данных есть пустые ячейки - столбцы Track,aretist, genre.
# Напишите ваш ответ здесь комментарием. Не удаляйте символ #. Не меняйте тип этой ячейки на Markdown.
```

Выводы

В каждой строке таблицы — данные о прослушанном треке. Часть колонок описывает саму композицию: название, исполнителя и жанр. Остальные данные рассказывают о пользователе: из какого он города, когда он слушал музыку.

Предварительно можно утверждать, что данных достаточно для проверки гипотез. Но встречаются пропуски в данных, а в названиях колонок — расхождения с хорошим стилем.

Чтобы двигаться дальше, нужно устранить проблемы в данных.

Предобработка данных

Переименование столбцов

Задание 6

In [7]:
```
print(df.columns)
# перечень названий столбцов таблицы df
```

```
Index(['  userID', 'Track', 'artist', 'genre', '  City  ', 'time', 'Day'], dtype='object')
```

Задание 7

In [8]:
```
df=df.rename(columns={'  userID':'user_id','Track':'track','  City  ':'city', 'Day':'day'})
# переименование столбцов
```

Задание 8

In [9]:
```
print(df.columns)
# проверка результатов - перечень названий столбцов
```

```
Index(['user_id', 'track', 'artist', 'genre', 'city', 'time', 'day'], dtype='object')
```

Обработка пропущенных значений

Задание 9

In [10]:
```
print(df.isna().sum())
#подсчёт пропусков
```

```
user_id       0
track      1231
artist     7203
genre      1198
city          0
time          0
day           0
dtype: int64
```

Задание 10

In [11]:
```
df=df.fillna('unknown')
#замена пропущенных значений на 'unknown'
```

Задание 11

In [12]:
```
print(df.isna().sum())
# проверка на отсутствие пропусков
```

```
user_id    0
track      0
artist     0
genre      0
city       0
time       0
day        0
dtype: int64
```

Обработка дубликатов

Задание 12

In [13]:
```
df.duplicated().sum()
# подсчёт явных дубликатов
```

Out[13]: 3826

Задание 13

In [14]:
```
df=df.drop_duplicates().reset_index(drop=True)
#удаление явных дубликатов, создание новых индексов и удаление старых
```

Задание 14

In [15]:
```
df.duplicated().sum()
# проверка на отсутствие явных дубликатов
```

Out[15]: 0

Задание 15

In [16]:
```
print(sorted(df['genre'].unique()))
# просмотр уникальных отсортированных названий жанров
```

```
['acid', 'acoustic', 'action', 'adult', 'africa', 'afrikaans', 'alternative', 'alternativepunk', 'ambient', 'americana', 'animated', 'anime',
'arabesk', 'arabic', 'arena', 'argentinetango', 'art', 'audiobook', 'author', 'avantgarde', 'axé', 'baile', 'balkan', 'beats', 'bigroom', 'bl
ack', 'bluegrass', 'blues', 'bollywood', 'bossa', 'brazilian', 'breakbeat', 'breaks', 'broadway', 'cantautori', 'cantopop', 'canzone', 'carib
bean', 'caucasian', 'celtic', 'chamber', 'chanson', 'children', 'chill', 'chinese', 'choral', 'christian', 'christmas', 'classical', 'classic
metal', 'club', 'colombian', 'comedy', 'conjazz', 'contemporary', 'country', 'cuban', 'dance', 'dancehall', 'dancepop', 'dark', 'death', 'dee
p', 'deutschrock', 'deutschspr', 'dirty', 'disco', 'dnb', 'documentary', 'downbeat', 'downtempo', 'drum', 'dub', 'dubstep', 'eastern', 'eas
y', 'electronic', 'electropop', 'emo', 'entehno', 'epicmetal', 'estrada', 'ethnic', 'eurofolk', 'european', 'experimental', 'extrememetal',
'fado', 'fairytail', 'film', 'fitness', 'flamenco', 'folk', 'folklore', 'folkmetal', 'folkrock', 'folktronica', 'forró', 'frankreich', 'franz
ösisch', 'french', 'funk', 'future', 'gangsta', 'garage', 'german', 'ghazal', 'gitarre', 'glitch', 'gospel', 'gothic', 'grime', 'grunge', 'gy
psy', 'handsup', "hard'n'heavy", 'hardcore', 'hardstyle', 'hardtechno', 'hip', 'hip-hop', 'hiphop', 'historisch', 'holiday', 'hop', 'horror',
'house', 'hymn', 'idm', 'independent', 'indian', 'indie', 'indipop', 'industrial', 'inspirational', 'instrumental', 'international', 'irish',
'jam', 'japanese', 'jazz', 'jewish', 'jpop', 'jungle', 'k-pop', 'karadeniz', 'karaoke', 'kayokyoku', 'korean', 'laiko', 'latin', 'latino', 'l
eftfield', 'local', 'lounge', 'loungeelectronic', 'lovers', 'malaysian', 'mandopop', 'marschmusik', 'meditative', 'mediterranean', 'melodic',
'metal', 'metalcore', 'mexican', 'middle', 'minimal', 'miscellaneous', 'modern', 'mood', 'mpb', 'muslim', 'native', 'neoklassik', 'neue', 'ne
w', 'newage', 'newwave', 'nu', 'nujazz', 'numetal', 'oceania', 'old', 'opera', 'orchestral', 'other', 'piano', 'podcasts', 'pop', 'popdance',
'popelectronic', 'popeurodance', 'poprussian', 'post', 'posthardcore', 'postrock', 'power', 'progmetal', 'progressive', 'psychedelic', 'punja
bi', 'punk', 'quebecois', 'ragga', 'ram', 'rancheras', 'rap', 'rave', 'reggae', 'reggaeton', 'regional', 'relax', 'religious', 'retro', 'rhyt
hm', 'rnb', 'rnr', 'rock', 'rockabilly', 'rockalternative', 'rockindie', 'rockother', 'romance', 'roots', 'ruspop', 'rusrap', 'rusrock', 'rus
sian', 'salsa', 'samba', 'scenic', 'schlager', 'self', 'sertanejo', 'shanson', 'shoegazing', 'showtunes', 'singer', 'ska', 'skarock', 'slow',
'smooth', 'soft', 'soul', 'soulful', 'sound', 'soundtrack', 'southern', 'specialty', 'speech', 'spiritual', 'sport', 'stonerrock', 'surf', 's
wing', 'synthpop', 'synthrock', 'sängerportrait', 'tango', 'tanzorchester', 'taraftar', 'tatar', 'tech', 'techno', 'teen', 'thrash', 'top',
'traditional', 'tradjazz', 'trance', 'tribal', 'trip', 'triphop', 'tropical', 'türk', 'türkçe', 'ukrrock', 'unknown', 'urban', 'uzbek', 'vari
été', 'vi', 'videogame', 'vocal', 'western', 'world', 'worldbeat', 'ïîï', 'электроника']
```

Задание 16

In [17]:
```
df['genre']=df['genre'].replace('hip', 'hiphop')
df['genre']=df['genre'].replace('hop', 'hiphop')
df['genre']=df['genre'].replace('hip-hop', 'hiphop')

# устранение неявных дубликатов
```

Задание 17

In [18]:
```
print(sorted(df['genre'].unique()))
# проверка на отсутствие неявных дубликатов
```

```
['acid', 'acoustic', 'action', 'adult', 'africa', 'afrikaans', 'alternative', 'alternativepunk', 'ambient', 'americana', 'animated', 'anime',
'arabesk', 'arabic', 'arena', 'argentinetango', 'art', 'audiobook', 'author', 'avantgarde', 'axé', 'baile', 'balkan', 'beats', 'bigroom', 'bl
ack', 'bluegrass', 'blues', 'bollywood', 'bossa', 'brazilian', 'breakbeat', 'breaks', 'broadway', 'cantautori', 'cantopop', 'canzone', 'carib
bean', 'caucasian', 'celtic', 'chamber', 'chanson', 'children', 'chill', 'chinese', 'choral', 'christian', 'christmas', 'classical', 'classic
metal', 'club', 'colombian', 'comedy', 'conjazz', 'contemporary', 'country', 'cuban', 'dance', 'dancehall', 'dancepop', 'dark', 'death', 'dee
p', 'deutschrock', 'deutschspr', 'dirty', 'disco', 'dnb', 'documentary', 'downbeat', 'downtempo', 'drum', 'dub', 'dubstep', 'eastern', 'eas
y', 'electronic', 'electropop', 'emo', 'entehno', 'epicmetal', 'estrada', 'ethnic', 'eurofolk', 'european', 'experimental', 'extrememetal',
'fado', 'fairytail', 'film', 'fitness', 'flamenco', 'folk', 'folklore', 'folkmetal', 'folkrock', 'folktronica', 'forró', 'frankreich', 'franz
ösisch', 'french', 'funk', 'future', 'gangsta', 'garage', 'german', 'ghazal', 'gitarre', 'glitch', 'gospel', 'gothic', 'grime', 'grunge', 'gy
psy', 'handsup', "hard'n'heavy", 'hardcore', 'hardstyle', 'hardtechno', 'hiphop', 'historisch', 'holiday', 'horror', 'house', 'hymn', 'idm',
'independent', 'indian', 'indie', 'indipop', 'industrial', 'inspirational', 'instrumental', 'international', 'irish', 'jam', 'japanese', 'jaz
z', 'jewish', 'jpop', 'jungle', 'k-pop', 'karadeniz', 'karaoke', 'kayokyoku', 'korean', 'laiko', 'latin', 'latino', 'leftfield', 'local', 'lo
unge', 'loungeelectronic', 'lovers', 'malaysian', 'mandopop', 'marschmusik', 'meditative', 'mediterranean', 'melodic', 'metal', 'metalcore',
'mexican', 'middle', 'minimal', 'miscellaneous', 'modern', 'mood', 'mpb', 'muslim', 'native', 'neoklassik', 'neue', 'new', 'newage', 'newwav
e', 'nu', 'nujazz', 'numetal', 'oceania', 'old', 'opera', 'orchestral', 'other', 'piano', 'podcasts', 'pop', 'popdance', 'popelectronic', 'po
peurodance', 'poprussian', 'post', 'posthardcore', 'postrock', 'power', 'progmetal', 'progressive', 'psychedelic', 'punjabi', 'punk', 'quebec
ois', 'ragga', 'ram', 'rancheras', 'rap', 'rave', 'reggae', 'reggaeton', 'regional', 'relax', 'religious', 'retro', 'rhythm', 'rnb', 'rnr',
'rock', 'rockabilly', 'rockalternative', 'rockindie', 'rockother', 'romance', 'roots', 'ruspop', 'rusrap', 'rusrock', 'russian', 'salsa', 'sa
mba', 'scenic', 'schlager', 'self', 'sertanejo', 'shanson', 'shoegazing', 'showtunes', 'singer', 'ska', 'skarock', 'slow', 'smooth', 'soft',
'soul', 'soulful', 'sound', 'soundtrack', 'southern', 'specialty', 'speech', 'spiritual', 'sport', 'stonerrock', 'surf', 'swing', 'synthpop',
'synthrock', 'sängerportrait', 'tango', 'tanzorchester', 'taraftar', 'tatar', 'tech', 'techno', 'teen', 'thrash', 'top', 'traditional', 'trad
jazz', 'trance', 'tribal', 'trip', 'triphop', 'tropical', 'türk', 'türkçe', 'ukrrock', 'unknown', 'urban', 'uzbek', 'variété', 'vi', 'videoga
me', 'vocal', 'western', 'world', 'worldbeat', 'ïîï', 'электроника']
```

Выводы

Предобработка обнаружила три проблемы в данных:

- нарушения в стиле заголовков,
- пропущенные значения,
- дубликаты — явные и неявные.

Вы исправили заголовки, чтобы упростить работу с таблицей. Без дубликатов исследование станет более точным.

Пропущенные значения вы заменили на `'unknown'`. Ещё предстоит увидеть, не повредят ли исследованию пропуски в колонке `genre`.

Теперь можно перейти к проверке гипотез.

Проверка гипотез

Сравнение поведения пользователей двух столиц

Первая гипотеза утверждает, что пользователи по-разному слушают музыку в Москве и Санкт-Петербурге. Проверим это предположение по данным о трёх днях недели — понедельнике, среде и пятнице. Для этого:

- Разделим пользователей Москвы и Санкт-Петербурга.
- Сравним, сколько треков послушала каждая группа пользователей в понедельник, среду и пятницу.

Задание 18

In [19]:
```
count_city=df.groupby('city')['city'].count()
print(count_city)
#подсчёт прослушиваний в каждом городе
```

```
city
Moscow              42741
Saint-Petersburg    18512
Name: city, dtype: int64
```

Задание 19

In [20]:
```
count_day=df.groupby('day')['day'].count()
count_day

#подсчёт прослушиваний в каждый из трёх дней
```

Out[20]:
```
day
Friday       21840
Monday       21354
Wednesday    18059
Name: day, dtype: int64
```

Задание 20

In [21]:
```
def number_tracks(day, city):
    track_list = df[df['day']== day]
    track_list = track_list[track_list['city']== city]
    track_list_count = track_list['user_id'].count()
    return track_list_count
```

```
#не понимаю!
#выберите только строки track_list со значением переменной city в столбце city
#track_list_count = #вызовите метод подсчета строк для track_list и выберите столбец user_id
#return  #верните значение track_list_count из функции print(track_list)
```

Задание 21

In [30]:
```
number_tracksMonM=number_tracks('Monday', 'Moscow')
number_tracksMonM
# количество прослушиваний в Москве по понедельникам
```

Out[30]: 15740

In [31]:
```
number_tracksMonS=number_tracks('Monday', 'Saint-Petersburg')
number_tracksMonS
# количество прослушиваний в Санкт-Петербурге по понедельникам
```

Out[31]: 5614

In [32]:
```
number_tracksWedM=number_tracks('Wednesday', 'Moscow')
number_tracksWedM
# количество прослушиваний в Москве по средам
```

Out[32]: 11056

In [33]:
```
number_tracksWedS=number_tracks('Wednesday', 'Saint-Petersburg')
number_tracksWedS
# количество прослушиваний в Санкт-Петербурге по средам
```

Out[33]: 7003

In [34]:
```
number_tracksFriM=number_tracks('Friday', 'Moscow')
number_tracksFriM
```

Out[34]: 15945

In [35]:
```
number_tracksFriS=number_tracks('Friday', 'Saint-Petersburg')
number_tracksFriS
# количество прослушиваний в Санкт-Петербурге по пятницам
```

Out[35]: 5895

Задание 22

In [36]:
```
data=[['Moscow',number_tracksMonM,number_tracksWedM,number_tracksFriM],
      ['Saint-Petersburg',number_tracksMonS,number_tracksWedS,number_tracksFriS]]
columns=['City','Monday','Wednesday','Friday']
info=pd.DataFrame(data=data, columns=columns)
print(info)

# создание таблицы с результатами
# вывод таблицы на экран
```

```
              City  Monday  Wednesday  Friday
0           Moscow   15740      11056   15945
1  Saint-Petersburg    5614       7003    5895
```

Выводы

Данные показывают разницу поведения пользователей:

- В Москве пик прослушиваний приходится на понедельник и пятницу, а в среду заметен спад.
- В Петербурге, наоборот, больше слушают музыку по средам. Активность в понедельник и пятницу здесь почти в равной мере уступает среде.

Значит, данные говорят в пользу первой гипотезы.

Музыка в начале и в конце недели

Согласно второй гипотезе, утром в понедельник в Москве преобладают одни жанры, а в Петербурге — другие. Так же и вечером пятницы преобладают разные жанры — в зависимости от города.

Задание 23

In [37]:
```
moscow_general=df[df['city']=='Moscow']
#получение таблицы moscow_general из тех строк таблицы df, для которых значение в столбце 'city' равно 'Moscow'
```

In [38]:
```
spb_general=df[df['city']=='Saint-Petersburg']
#получение таблицы spb_general из тех строк таблицы df, для которых значение в столбце 'city' равно 'Saint-Petersburg'
```

Задание 24

In [30]:
```
def genre_weekday(df, day, time1, time2):

    genre_df=df[df['day'] == day]
    genre_df=genre_df[genre_df['time'] < time2]
    genre_df=genre_df[genre_df['time'] > time1]
    genre_df_grouped = genre_df.groupby('genre')['genre'].count()
    genre_df_sorted = genre_df_grouped.sort_values(ascending=False)
    return genre_df_sorted.head(10)

    #последовательная фильтрация
    #оставляем в genre_df только те строки df, у которых день равен day
    #оставляем в genre_df только те строки genre_df, у которых время меньше time2
    #оставляем в genre_df только те строки genre_df, у которых время больше time1
    #сгруппируем отфильтрованный датафрейм по столбцу с названиями жанров, возьмём столбец genre и посчитаем кол-во строк для каждого жанра м
    #отсортируем результат по убыванию (чтобы в начале Series оказались самые популярные жанры)
    #вернём Series с 10 самыми популярными жанрами в указанный отрезок времени заданного дня
```

Задание 25

In [31]:
```
genre_weekday(moscow_general,'Monday','07:00','11:00')

#вызов функции для утра понедельника в Москве (вместо df — таблица moscow_general)
```

Out[31]:
```
genre
pop            781
dance          549
electronic     480
rock           474
hiphop         286
ruspop         186
world          181
rusrap         175
alternative    164
unknown        161
Name: genre, dtype: int64
```

In [32]:
```
genre_weekday(spb_general,'Monday','07:00','11:00')
# вызов функции для утра понедельника в Петербурге (вместо df — таблица spb_general)
```

Out[32]:
```
genre
pop            218
dance          182
rock           162
```

```
electronic     147
hiphop          80
ruspop          64
alternative     58
rusrap          55
jazz            44
classical       40
Name: genre, dtype: int64
```

In [33]:

```
genre_weekday(moscow_general,'Friday','17:00','23:00')
# вызов функции для вечера пятницы в Москве
```

Out[33]:

```
genre
pop            713
rock           517
dance          495
electronic     482
hiphop         273
world          208
ruspop         170
alternative    163
classical      163
rusrap         142
Name: genre, dtype: int64
```

In [34]:

```
genre_weekday(spb_general,'Friday','17:00','23:00')
#вызов функции для вечера пятницы в Петербурге
```

Out[34]:

```
genre
pop            256
electronic     216
rock           216
dance          210
hiphop          97
alternative     63
jazz            61
classical       60
rusrap          59
world           54
Name: genre, dtype: int64
```

Выводы

Если сравнить топ-10 жанров в понедельник утром, можно сделать такие выводы:

1. В Москве и Петербурге слушают похожую музыку. Единственное различие — в московский рейтинг вошёл жанр “world”, а в петербургский — джаз и классика.

2. В Москве пропущенных значений оказалось так много, что значение `'unknown'` заняло десятое место среди самых популярных жанров. Значит, пропущенные значения занимают существенную долю в данных и угрожают достоверности исследования.

Вечер пятницы не меняет эту картину. Некоторые жанры поднимаются немного выше, другие спускаются, но в целом топ-10 остаётся тем же самым.

Таким образом, вторая гипотеза подтвердилась лишь частично:

- Пользователи слушают похожую музыку в начале недели и в конце.
- Разница между Москвой и Петербургом не слишком выражена. В Москве чаще слушают русскую популярную музыку, в Петербурге — джаз.

Однако пропуски в данных ставят под сомнение этот результат. В Москве их так много, что рейтинг топ-10 мог бы выглядеть иначе, если бы не утерянные данные о жанрах.

Жанровые предпочтения в Москве и Петербурге

Гипотеза: Петербург — столица рэпа, музыку этого жанра там слушают чаще, чем в Москве. А Москва — город контрастов, в котором, тем не менее, преобладает поп-музыка.

Задание 26

In [38]:
```
moscow_genres=moscow_general.groupby('genre')['genre'].count().sort_values(ascending=False)


#одной строкой: группировка таблицы moscow_general по столбцу 'genre', выбор столбца `genre`, подсчёт числа значений 'genre' методом count(),
#сортировка получившегося Series в порядке убывания и сохранение обратно в moscow_genres
```

Задание 27

In [39]:
```
print(moscow_genres.head(10))
# просмотр первых 10 строк moscow_genres
```

```
genre
pop            5892
dance          4435
rock           3965
electronic     3786
hiphop         2096
classical      1616
world          1432
alternative    1379
ruspop         1372
rusrap         1161
Name: genre, dtype: int64
```

Задание 28

In [42]:
```
spb_genres=spb_general.groupby('genre')['genre'].count().sort_values(ascending=False)
# одной строкой: группировка таблицы spb_general по столбцу 'genre', выбор столбца `genre`, подсчёт числа значений 'genre' методом count(), с
# сортировка получившегося Series в порядке убывания и сохранение обратно в spb_genres
```

Задание 29

In [41]:
```
print(spb_genres.head(10))
# просмотр первых 10 строк spb_genres
```

```
genre
pop            2431
dance          1932
rock           1879
electronic     1736
hiphop          960
alternative     649
classical       646
rusrap          564
ruspop          538
world           515
Name: genre, dtype: int64
```

Выводы

Гипотеза частично подтвердилась:

- Поп-музыка — самый популярный жанр в Москве, как и предполагала гипотеза. Более того, в топ-10 жанров встречается близкий жанр — русская популярная музыка.
- Вопреки ожиданиям, рэп одинаково популярен в Москве и Петербурге.

Итоги исследования

Вы проверили три гипотезы и установили:

1. День недели по-разному влияет на активность пользователей в Москве и Петербурге.

Первая гипотеза полностью подтвердилась.

1. Музыкальные предпочтения не сильно меняются в течение недели — будь то Москва или Петербург. Небольшие различия заметны в начале недели, по понедельникам:

- в Москве слушают музыку жанра “world”,

- в Петербурге — джаз и классику.

Таким образом, вторая гипотеза подтвердилась лишь отчасти. Этот результат мог оказаться иным, если бы не пропуски в данных.

1. Во вкусах пользователей Москвы и Петербурга больше общего, чем различий. Вопреки ожиданиям, предпочтения жанров в Петербурге напоминают московские.

Третья гипотеза не подтвердилась. Если различия в предпочтениях и существуют, на основной массе пользователей они незаметны.

На практике исследования содержат проверки статистических гипотез. Из части данных одного сервиса невозможно сделать какие-то выводы о всех пользователях сервиса без методов статистики. Проверки статистических гипотез покажут, насколько они достоверны, исходя из имеющихся данных. С методами проверок гипотез вы ещё познакомитесь в следующих темах.