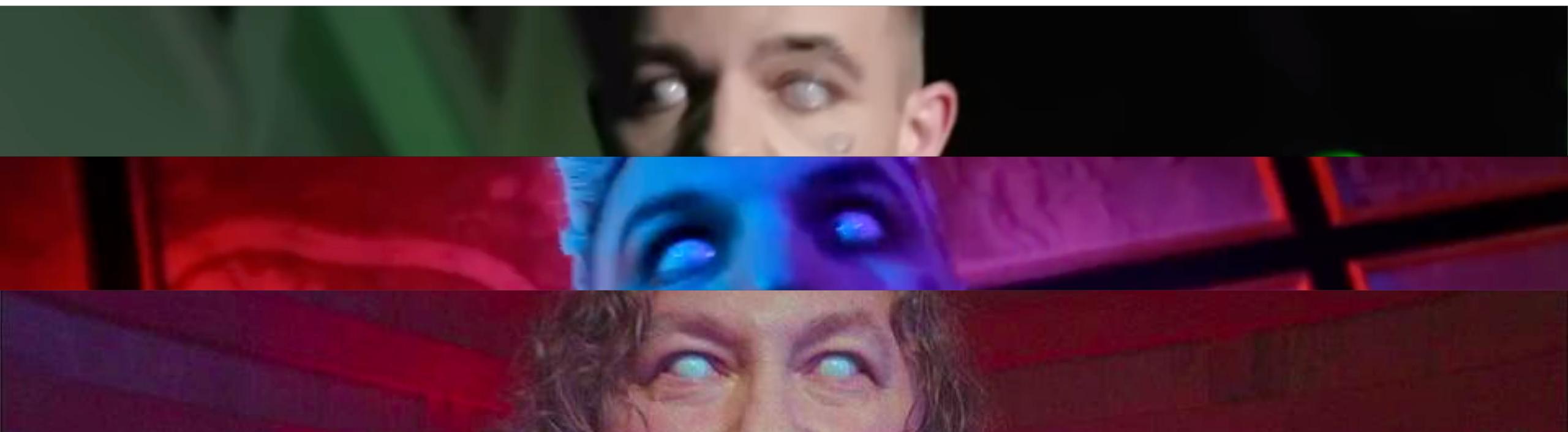


"Кис-кис, вдыхай меня через кес" или о чем говорят любители рэпа

Python для тематического моделирования
комментариев Вконтакте



От А! до Я

Aalto University
School of Science

Дмитрий Сергеев
Data Scientist

Telegram: @dmitryserg
Mail: Sergeyev.D.A@Yandex.ru
Github: <https://github.com/DmitrySerg>



Филипп Ульянкин
Data Scientist

Telegram: @Ppilif
Mail: Filfonul@gmail.com
Github: <https://github.com/FUlyankin>

План

- Парсинг VK и YouTube
- Предобработка текстов и LDA
- Нестандартное использование тематических векторов в домашних условиях

Цель

- Выяснить, различаются ли темы для общения у слушателей разных музыкальных жанров
- Понять, можно ли с этими темами сделать что-то ещё

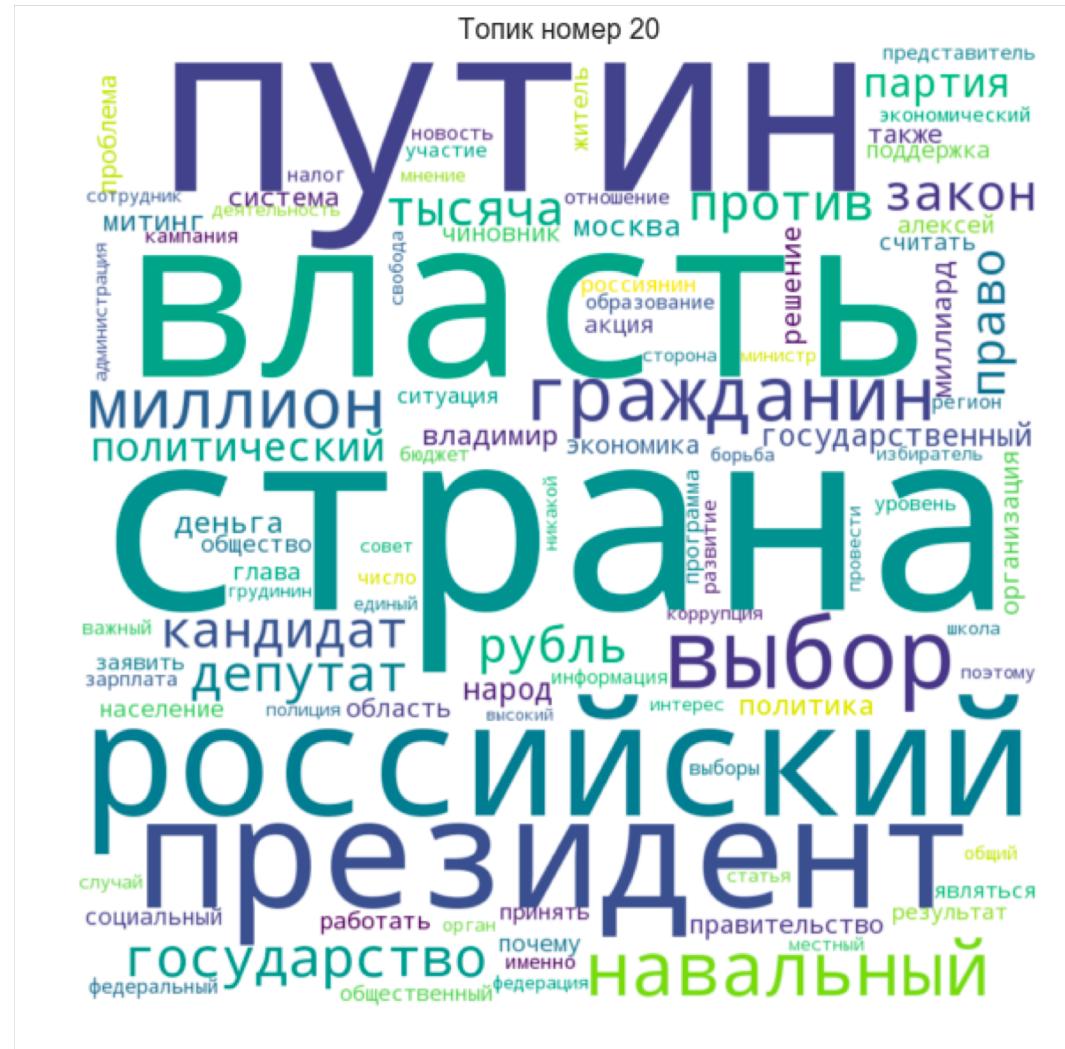
Зачем?

Зачем?



Зачем?

Кто выбирает президентов? - исследуем избирателей Вконтакте (OpenDataDay 2018)



<https://www.youtube.com/watch?v=l653eXrSV2o>

<https://github.com/DmitrySerg/OpenData/tree/master/RussianElections2018>

Подход

■ Найти жанры

X Весь список

🎵 Под настроение

⚽ Для занятий

Поп

Инди

Рок

Метал

Альтернатива

Электроника

Танцевальная

Рэп и хип-хоп

R&B

Джаз

Блюз

Регги

Ска

Панк

Классика

Эстрада

Шансон

Кантри

Авторская песня

Лёгкая музыка

🎬 Саундтреки

Музыка мира

Детская

Подход

- Найти топовых исполнителей по жанрам

The screenshot shows a dark-themed interface for a music streaming platform. At the top, the word 'Шансон' is displayed in large white letters. Below it is a navigation bar with tabs: 'ОБЗОР', 'ТРЕКИ', 'АЛЬБОМЫ', 'ИСПОЛНИТЕЛИ' (which is highlighted with a yellow underline), and 'КОНЦЕРТЫ'. To the right of the tabs are two buttons: a yellow one labeled 'Radio жанра' with a radio icon, and a white one with a heart icon. The main content area is titled 'Популярные исполнители' and features four circular profile pictures of artists: Mikhail Krug, Irina Krug, Sergey Trofimov, and Butyrka. Each artist's name and genre ('шансон') are written below their respective portraits.

Шансон

ОБЗОР ТРЕКИ АЛЬБОМЫ ИСПОЛНИТЕЛИ КОНЦЕРТЫ

Radio жанра

Популярные исполнители

Михаил Круг
шансон

Ирина Круг
шансон

Сергей Трофимов
шансон

Бутырка
шансон

Подход

Итого выбрали **11** жанров:

Эстрада, Попса, Рок, Метал, Классика, Рэп,
Шансон, Джаз, Панк, Танцевальная, Инди

И **134** исполнителя

Подход

- Найти соответствующие группы в ВК

ГРУППА "БУТЫРКА" • Русский Шансон
Поклонники творчества группы "БУТЫРКА", ПОДПИСЫВАЙТЕСЬ 

 **ГРУППА "БУТЫРКА" • Русский Шансон** запись закреплена
30 апр в 17:49

▼

Нас 95.000!
Уважаемые подписчики, спасибо, что Вы с нами!
ЛЕГЕНДАРНЫЕ ХИТЫ "ЗОЛОТОГО" СОСТАВА ГРУППЫ "БУТЫРКА" ДЛЯ ВАС!



Подход

- И плейлисты на YouTube

Бутырка – тема ▶

Главная Плейлисты О канале

Лучшие композиции – Бутырка

 Запахло весной Smotrakoff 4 685 749 просмотров • 8 лет назад	 Бутырка - Сорвутся голуби Znahar2012 10 144 697 просмотров • 7 лет назад	 Бутырка - А для вас я никто iljucha86 7 414 993 просмотра • 7 лет назад	 Бутырка - За Ростовскую братву Mindaugas Zukauskas 7 458 054 просмотра • 6 лет назад
--	---	---	--

Подход

- А дальше - модельки

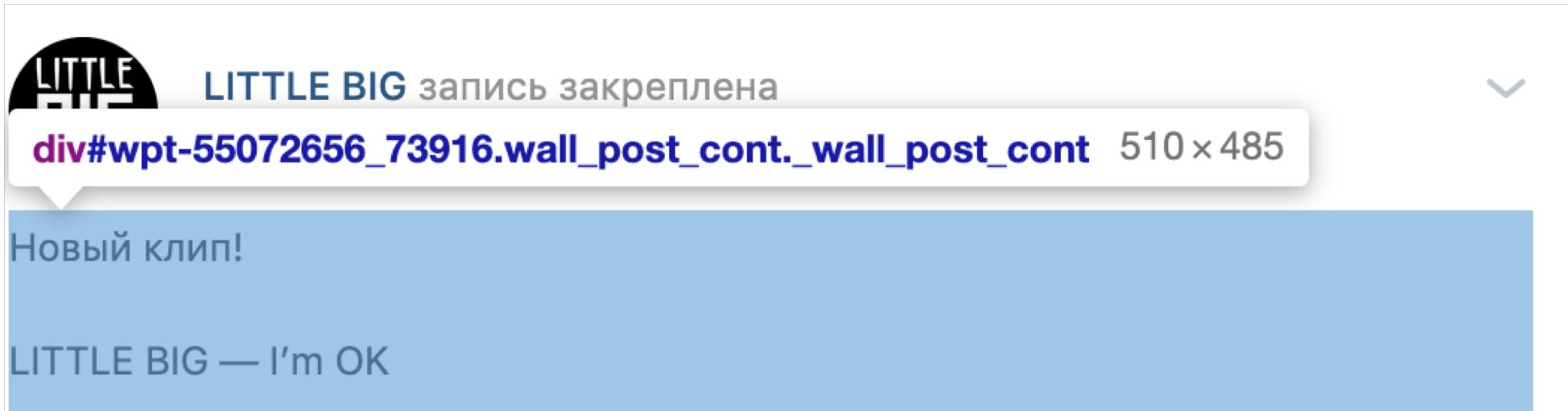
Парсинг

VK.com - отличный API:

<https://vk.com/dev/manuals>

[Ссылка на код с парсером](#)

Парсинг



```
target_groups =  
    'little_big': '55072656', # 510 145  
    'sidxram': '75762231', #170 104  
    'нейромонах_феофан': '30192213', # 136 804  
    'denderty': '18362978', # 10 219  
    'oligarkh': '54550120', # 13 488  
}
```

Парсинг

```
def vkDownload(method, parameters, token=token, version=version):
    """
        Возвращает результат запроса по методу

        method: string
            метод из документации, который хотим использовать

        parameters: string
            параметры используемого метода

        token: string
            токен OAuth доступа

        version: string
            версия API
    """

    # составляем ссылку
    url = 'https://api.vk.com/method/{method}?{parameters}&access_token={token}&v={version}'
    url = url.format(method=method, parameters=parameters, token=token, version=version)
    # запрашиваем ссылку и переводим в json (словарь)
    response = requests.get(url).json()

    return response
```

Парсинг

Метод execute - парсим в 25 раз быстрее

```
# делаем из всех постов батчи по 25 штук, чтобы было быстрее
posts_batch = makeBatch(posts)

for batch in tqdm_notebook(posts_batch):
    time.sleep(0.4)

# готовим запрос для комментов из батча, который выяснит число комментов под постом
begin = 'https://api.vk.com/method/execute?code=return['
end = '];&access_token=' + token + '&v=5.78'
middle = ''

# для каждого поста из батча используем метод wall.getComments
for bt in batch:
    middle += 'API.wall.getComments({ "owner_id": "' + str(group_id) + '", "post_id": "' + str(bt) + '"}),'

# делаем запрос для текущих 25 комментов
requests.get(begin + middle[:-1] + end)
```

Парсинг

YouTube.com - отвратительный API:

<https://console.developers.google.com/apis/api/youtube.googleapis.com/>

[Ссылка на код с парсером](#)

Парсинг

```
base_search_url = 'https://www.googleapis.com/youtube/v3/search?'
url = base_search_url+'key={}&channelId={}&part=snippet,id&order=date&maxResults=25'.format(
    api_key,
    "UCs6h0i7chZZX3fWD4IOh1kQ"
)
req = requests.get(url)
```

Парсинг

В результате собрали:

- ~ **6** миллионов комментариев из Вконтакте
- ~ **4** миллиона из YouTube

Предобработка

В основном - gensim



Предобработка

- Соединили финальные корпусы текста в один

Предобработка

- Соединили финальные корпусы текста в один
- Bag-of-words

Предобработка

- Соединили финальные корпусы текста в один
- Bag-of-words
- Выкинули слишком короткие комментарии
(оптимальный результат на 40+ слов)

Предобработка

- Соединили финальные корпусы текста в один
- Bag-of-words
- Выкинули слишком короткие комментарии
(оптимальный результат на 40+ слов)
- Bigrams + фильтрация по частоте

Предобработка

```
# Build the bigram models
bigram = models.Phrases(texts, min_count=3, threshold=5) # higher threshold fewer phrases.

# Faster way to get a sentence clubbed as a bigram
bigram_mod = models.phrases.Phraser(bigram)

def make_bigrams(texts):
    return [bigram_mod[doc] for doc in texts]

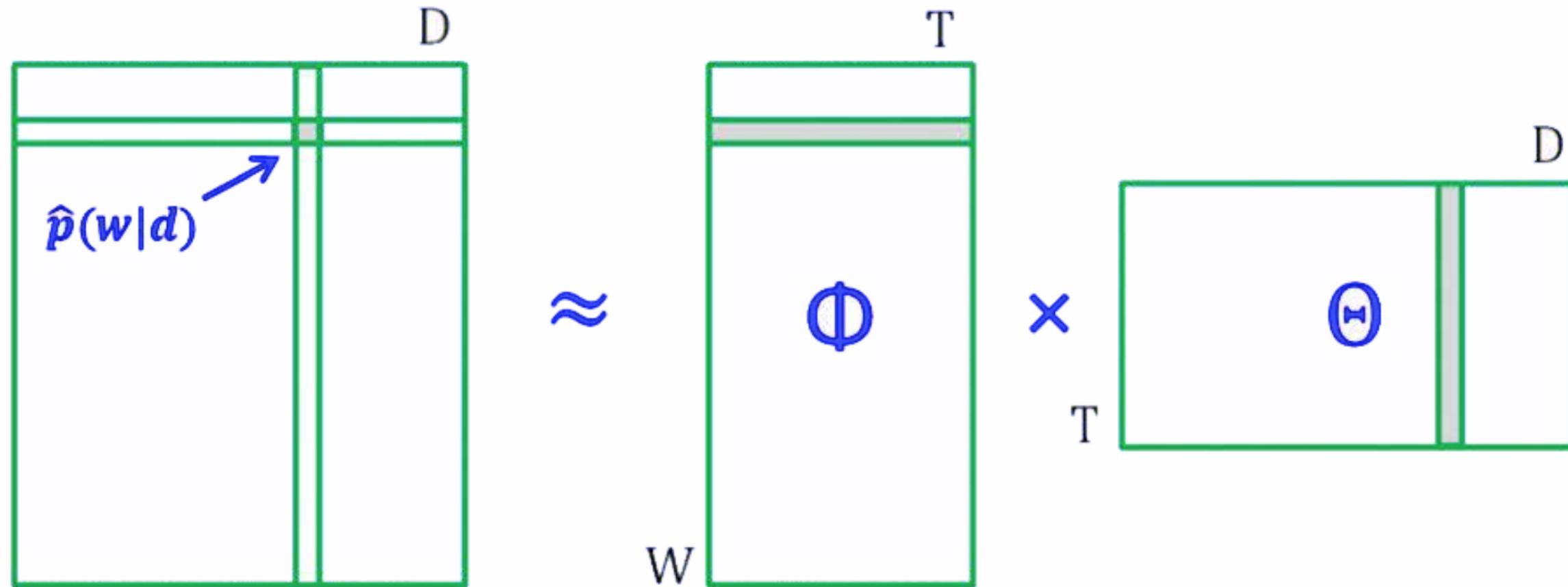
texts = make_bigrams(texts)
dictionary = corpora.Dictionary(texts)                      # составляем словарь из терминов
dictionary.filter_extremes(no_below=3, no_above=0.4, keep_n=3*10**6)
corpus = [dictionary.doc2bow(text) for text in texts]    # составляем корпус документов
```

[Ссылка на код с предобработкой](#)

Модель

- Latent Dirichlet allocation (LDA)

Модель



Модель

- Latent Dirichlet allocation (LDA)
- Главный гиперпараметр для нас - число тем
- Остановились на **30** темах

[Ссылка на код с моделью](#)

Модель

- Важно: при сохранении модели нужно отдельно сохранять атрибут **expElogbeta**
- При подгрузке модели отдельно его подгружать и присваивать:

```
with open('lda_models/ldamodel_30_bigrams_long_comments', 'rb') as f:  
    ldamodel_30 = pickle.load(f)  
  
with open('lda_models/corpus', 'rb') as f:  
    corpus = pickle.load(f)  
  
expElogbeta = np.load("lda_models/ldamodel_30_bigrams_long_comments.expElogbeta.npy")  
ldamodel_30.expElogbeta = expElogbeta
```

Результаты

- Из 30 тем 27 удалось проинтерпретировать
- В основном, темы связаны с песнями, альбомами, благодарностями за концерт и т.д.
- Но есть и исключения

[Ссылка на код с визуализацией](#)

Результаты

Визуализация - WordCloud

```
def plotWordCloud(topic_number):
    """
        Строит визуализацию слов на основе текстов топиков
    """
    # получаем частоты и слова топика

    text = dict(lda_30_topics[topic_number][1])

    # строим облако слов
    wordcloud = WordCloud(background_color="white", max_words=100, width=900, height=900, collocations=False)
    wordcloud = wordcloud.generate_from_frequencies(text)
    plt.figure(figsize=(15, 10))
    plt.title("Топик номер {}".format(topic_number))
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis("off");
```

Топик номер 1

A word cloud visualization representing political discourse in Russian media. The words are arranged in a grid, with larger words indicating higher frequency or importance. The colors of the words vary, suggesting different semantic clusters or sources.

The main themes visible in the word cloud include:

- Ukraine**: A large blue word at the top left, associated with " другой_страна" (another country), " описание" (description), " восемь" (eight), " золото" (gold), and " вместе" (together).
- Russia**: A large green word in the center, associated with " политика" (politics), " против" (against), " пропаганда" (propaganda), and " запретить" (to ban).
- Army**: A large yellow word on the left, associated with " автомат" (automatic), " оборот" (turnover), " пустить" (to let), " государство" (state), and " партия" (party).
- Crime**: A large purple word in the center, associated with " народный" (national), " бла" (bla), " житель" (resident), " позор" (shame), and " польша" (Poland).
- World War II**: A large red word in the center, associated with " народ" (people), " мир" (world), " союз" (union), " позиция" (position), and " продажный" (mercenary).
- Soviet Union**: A large orange word at the bottom left, associated with " ССР" (SSR), " СССР" (SSSR), " журналист" (journalist), " тачка" (car), " военный" (military), and " youtube" (YouTube).
- USA**: A large pink word at the bottom right, associated with " запад" (West), " запад" (West), " США" (USA), " европа" (Europe), " представитель" (representative), and " система" (system).

Other notable words include " режим" (regime), " гагарин" (Gagarin), " пара_год" (pair year), " частность" (particularity), " выступить" (to appear), " лечь" (to lie down), " время" (time), " вне_время" (beyond time), " миллион" (million), " нытьё" (grief), " погибший" (dead), " наш_страна" (our country), " бла_бла" (bla-bla), " конь" (horse), " год" (year), " победа" (victory), " считаться" (to be considered), " олигарх" (oligarch), " ум_сходить" (to go crazy), " влиять" (to influence), and " забыть" (to forget).

Топик номер 27

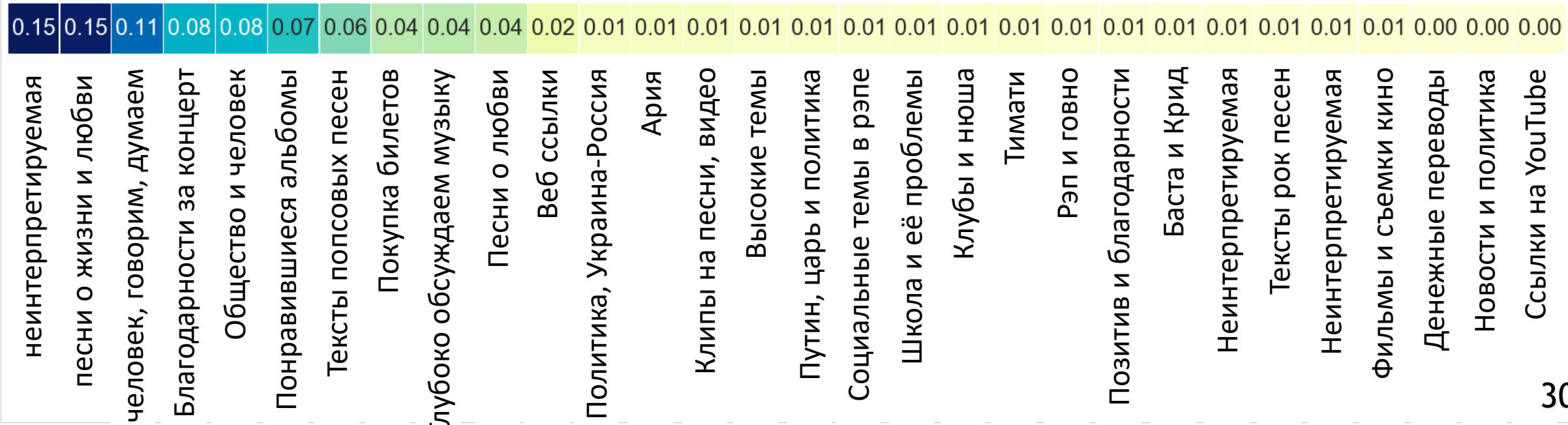
Топик номер 6

Топик номер 19

Результаты

Посчитав средние тематические вектора по подписчикам можно получить тематические профили по нужным группировкам

Тематический профиль для shanson:



Тематический профиль для тимати:

Результаты

А имея средние вектора
можно считать расстояния
между ними!

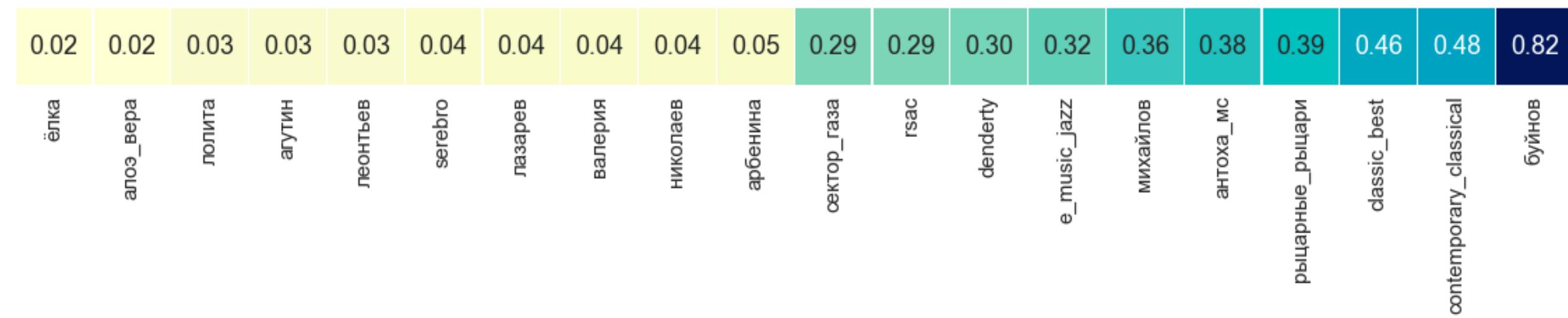
В данном случае
косинусное

	Estrada	Popsa	Rock	Metal	Classic	Rap	Shanson	Jazz	Pank	Dance	Indi
Estrada	0.00	0.05	0.01	0.06	0.12	0.04	0.09	0.12	0.03	0.05	0.02
Popsa	0.05	0.00	0.06	0.12	0.20	0.04	0.08	0.20	0.08	0.05	0.04
Rock	0.01	0.06	0.00	0.06	0.09	0.04	0.08	0.12	0.01	0.04	0.01
Metal	0.06	0.12	0.06	0.00	0.15	0.09	0.16	0.09	0.07	0.07	0.06
Classic	0.12	0.20	0.09	0.15	0.00	0.14	0.23	0.04	0.09	0.14	0.11
Rap	0.04	0.04	0.04	0.09	0.14	0.00	0.11	0.15	0.04	0.01	0.02
Shanson	0.09	0.08	0.08	0.16	0.23	0.11	0.00	0.23	0.14	0.13	0.08
Jazz	0.12	0.20	0.12	0.09	0.04	0.15	0.23	0.00	0.12	0.14	0.12
Pank	0.03	0.08	0.01	0.07	0.09	0.04	0.14	0.12	0.00	0.03	0.02
Dance	0.05	0.05	0.04	0.07	0.14	0.01	0.13	0.14	0.03	0.00	0.02
Indi	0.02	0.04	0.01	0.06	0.11	0.02	0.08	0.12	0.02	0.02	0.00

Результаты

А также можно искать похожих (и непохожих)
друг на друг исполнителей!

Киркоров: топ-10 похожих и непохожих



Лсп: топ-10 похожих и непохожих

0.01	0.02	0.02	0.02	0.02	0.03	0.03	0.03	0.03	0.04	0.04	0.29	0.30	0.31	0.31	0.32	0.32	0.40	0.41	0.43	0.77
лэджей	Максим	охххумирон	gazgolder	noizemс	барских	little_big	пошлая_молли	майами	гш	антоха_ms	басков	хадн_дадн	ludvigvonbeethoven	шуфутинский	русскийшансон	contemporary_classical	рыцарные_рыцари	classic_best	буйнов	

Король_и_шут: топ-10 похожих и непохожих

0.01	0.02	0.02	0.02	0.02	0.02	0.03	0.03	0.03	0.03	0.22	0.23	0.23	0.24	0.24	0.25	0.28	0.32	0.35	0.70	
лпс	бн_2	руки_вверх	агутин	зэмфира	ротару	мумий_тролль	звери	трофимов	электрофорез	jazz_джаз	miyagi&эндштиль	peterchaikovsky	inclassic	classic_best	classicstories	казускома	contemporary_classical	рыцарные_рыцари	classic_best	буйнов

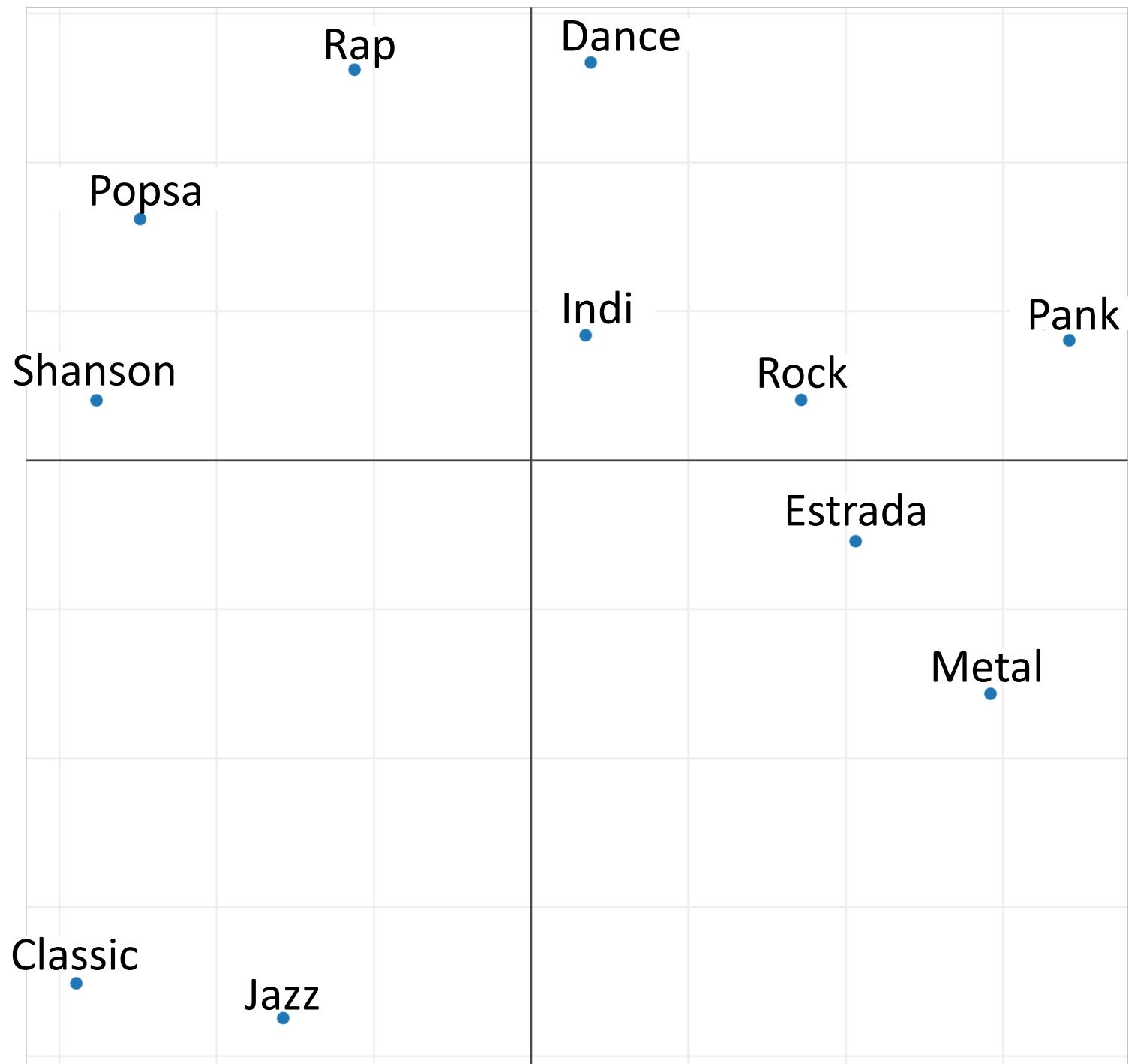
Результаты

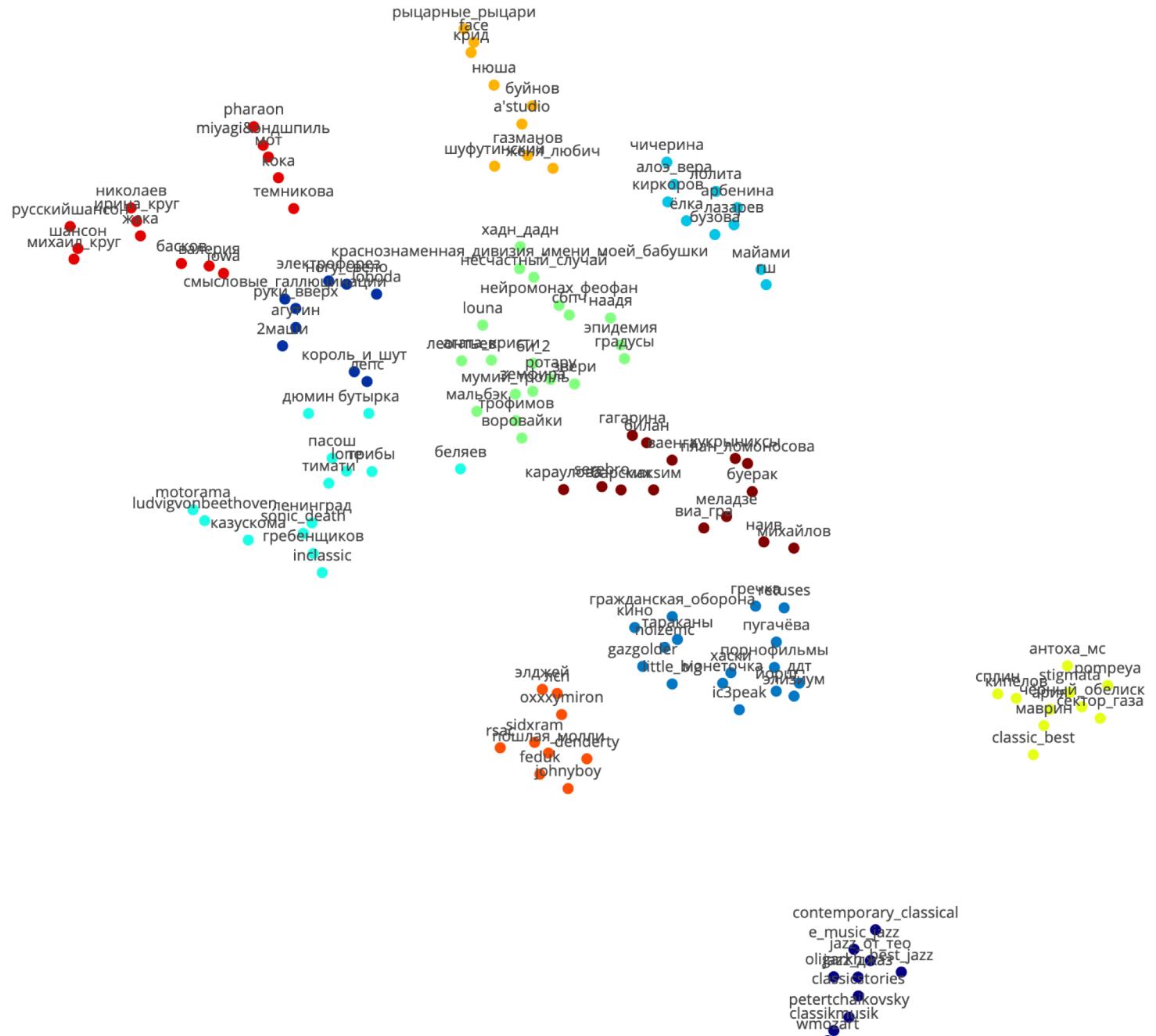
Наконец, можно посмотреть в многомерное пространство на плоскости!

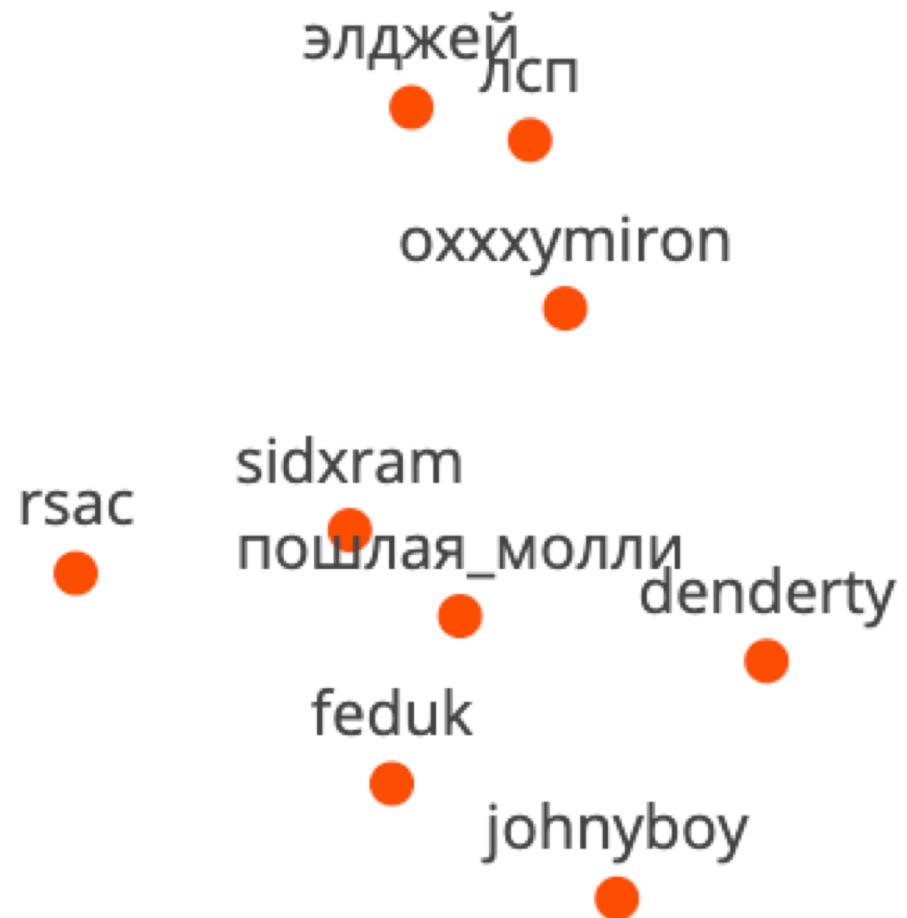
Multicore-TSNE - как TSNE, только Multicore

```
from MulticoreTSNE import MulticoreTSNE
```

```
tsne = MulticoreTSNE()  
tsne_features = tsne.fit_transform(music_style_themes_clusters)
```













Выводы

- Слушатели разных жанров действительно обсуждают разное

Выводы

- Слушатели разных жанров действительно обсуждают разное
- Настолько разное, что по темам можно заново выделять кластеры музыкальных жанров

Выводы

- Слушатели разных жанров действительно обсуждают разное
- Настолько разное, что по темам можно заново выделять кластеры музыкальных жанров
- И это круто!

От А! до Я

Aalto University
School of Science

Дмитрий Сергеев
Data Scientist

Telegram: @dmitryserg
Mail: Sergeyev.D.A@Yandex.ru
Github: <https://github.com/DmitrySerg>



Филипп Ульянкин
Data Scientist

Telegram: @Ppilif
Mail: Filfonul@gmail.com
Github: <https://github.com/FUlyankin>