Введение в обработку естественного языка

Урок 4. Тематическое моделирование. EM-алгоритм

Задача тематическое моделирование

Продолжаем исследование датасета с твитами

Скачиваем датасет (источник - http://study.mokoron.com/ (http://study.mokoron.com/)):

- положительные - https://www.dropbox.com/s/fnpq3z4bcnoktiv/positive.csv?dl=0 (https://www.dropbox.com/s/fnpq3z4bcnoktiv/positive.csv?dl=0),
- отрицательные - https://www.dropbox.com/s/r6u59ljhhjdg6j0/negative.csv (https://www.dropbox.com/s/r6u59ljhhjdg6j0/negative.csv).

Или можно через ноутбук:

- !wget https://www.dropbox.com/s/fnpq3z4bcnoktiv/positive.csv (https://www.dropbox.com/s/fnpq3z4bcnoktiv/positive.csv)
- !wget https://www.dropbox.com/s/r6u59ljhhjdg6j0/negative.csv (https://www.dropbox.com/s/r6u59ljhhjdg6j0/negative.csv)

как альтернатива можно скачать данные из Роспотребнадзора https://zpp.rospotrebnadzor.ru/Forum/Appeals (https://zpp.rospotrebnadzor.ru/Forum
parse_rospotrebnadzor.ipynb устанавливаем количество скачанных страниц больше не 50-сят, хотябы 500 и для анализа берём только вопрось

Что надо сделать:

1. объединить в одну выборку (это только для твитов), для роспотребнадзора сформировать датасет из вопросов
2. провести исследование и выявить тематики о которых говорят в твитах (для твитов), а для роспотребнадзора так же выявить тематики о
3. сделать визуализацию кластеров тематик
4. проинтерпритировать получившиеся тематики

Сдайте задание до: 9 июня, 22:00 +05

Выполнил ***Соковнин ИЛ***

Задания:

Задача 1.

Скачивание и подготовка данных из Роспотребнадзора https://zpp.rospotrebnadzor.ru/Forum/Appeals (https://zpp.rospotrebnadzor.ru/Forum/Appea
parse_rospotrebnadzor_hw.ipynb.

Скачано 500 страниц. Для анализа взяты только вопросы.

В [1]:
```
import pandas as pd
```

В [2]:
```
columns = ['text']
question_df = pd.read_csv("./data/rpn_question.csv", names = columns)
question_df.head()
```

Out[2]:

	text
0	24 мая 2022 года получил в подарок сертификат ...
1	Добрый день! помогите пожалуйста разобраться в...
2	Добрый день.\r\nКупила в магазине техники элек...
3	Добрый день.\r\nКупила в магазине техники элек...
4	Добрый день!\r\nПрошу вас разъяснить следующий...

Задача 2.

провести исследование и выявить тематики о которых говорят в твитах (для твитов), а для роспотребнадзора так же выявить тематики о кото

Предобработка

В [ ]:
```
!pip install corus
```

B [3]:
```
import re
import numpy as np
from nltk.corpus import stopwords
from tqdm.notebook import tqdm
from multiprocessing import Pool
# from pymystem3 import Mystem
```

B [4]:
```
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\sil\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[4]:
```
True
```

B [5]:
```
# !pip install pymorphy2
```

B [6]:
```
import pymorphy2
morph = pymorphy2.MorphAnalyzer()
```

B [7]:
```
words_regex = re.compile('\w+')

def find_words(text, regex = words_regex):
    tokens =  regex.findall(text.lower())
    return [w for w in tokens if w.isalpha() and len(w) >= 3]


stopwords_list = stopwords.words('russian')

# mystem = Mystem()
# def lemmatize(words, lemmer = mystem, stopwords = stopwords_list):
#     lemmas = lemmer.lemmatize(' '.join(words))
#     return [w for w in lemmas if not w in stopwords
#             and w.isalpha()]

def lemmatize(words, lemmer = morph, stopwords = stopwords_list):
    lemmas = [lemmer.parse(w)[0].normal_form for w in words]
    return [w for w in lemmas if not w in stopwords
            and w.isalpha()]

def preprocess(text):
    return (lemmatize(find_words(text)))
```

B [8]:
```
question_df.text.iloc[1]
```

Out[8]:
```
'Добрый день! помогите пожалуйста разобраться в ситуации. в приложении Вайлдберриз я заказала игрушку для сына 16 мая это
т срок игрушку не доставили, более того, пишут каждый раз что товар задерживается и ставят новые сроки. что мне делать? м
```

B [9]:
```
print(preprocess(question_df.text.iloc[1]))
```

```
['добрый', 'день', 'помочь', 'пожалуйста', 'разобраться', 'ситуация', 'приложение', 'вайлдберриз', 'заказать', 'игрушка',
оять', 'май', 'однако', 'срок', 'игрушка', 'доставить', 'писать', 'каждый', 'товар', 'задерживаться', 'ставить', 'новый',
```

B [10]:
```
preprocessed_text = list(tqdm(map(preprocess, question_df['text']), total=len(question_df)))
```

```
100%                                              2495/2495 [02:44<00:00, 15.13it/s]
```

B [11]:
```
question_df['text'] = preprocessed_text
question_df.sample(3)
```

Out[11]:

	text
1885	[здравствуйте, оплатить, товар, алиэкспресс, с...
1060	[компания, представиться, юридический, организ...
1149	[гражданин, вакцинировать, пфайзер, молдавия, ...

Модель LDA

Первая модель, которую мы рассмотрим, LDA - латентное размещение Дирихле. Воспользуемся реализацией из библиотеки gensim.

B [21]:
```
!pip install -U gensim
```

...

В [12]:
```
from gensim.models import *
from gensim import corpora
```

Модель использует векторное представление документов, например, мешок слов (bow), поэтому спервa создадим словарь:

В [13]:
```
dictionary = corpora.Dictionary(question_df['text'])

dictionary.filter_extremes(no_below = 10, no_above = 0.9, keep_n=None) # игнорируем слова, которые встречаются реже 10 ра
dictionary.save('lenta.dict')
```

Векторизуем документы:

В [14]:
```
corpus = [dictionary.doc2bow(text) for text in question_df['text']]
corpora.MmCorpus.serialize('lenta.model', corpus)
```

Теперь можем обучать модель:

В [15]:
```
%time lda = ldamodel.LdaModel(corpus, id2word=dictionary, num_topics=20, chunksize=50, update_every=1, passes=2)
```

```
Wall time: 2.64 s
```

Посмотрим на получившиеся темы:

В [16]:
```
lda.show_topics(num_topics=10, num_words=10, formatted=True)
```

Out[16]:
```
[(5,
  '0.170*"делать" + 0.102*"упаковка" + 0.068*"начало" + 0.049*"весь" + 0.043*"понимать" + 0.042*"оказаться" + 0.042*"домо
 (0,
  '0.046*"торговый" + 0.042*"сертификат" + 0.041*"выдать" + 0.031*"кредит" + 0.028*"московский" + 0.027*"дать" + 0.026*"к
*"отдел"'),
 (14,
  '0.032*"день" + 0.028*"сказать" + 0.027*"телефон" + 0.026*"это" + 0.023*"мочь" + 0.020*"номер" + 0.018*"всё" + 0.014*"ц
 (9,
  '0.089*"компания" + 0.064*"медицинский" + 0.055*"ооо" + 0.047*"отказываться" + 0.046*"петербург" + 0.045*"санкт" + 0.04
0.030*"переписка"'),
 (2,
  '0.134*"товар" + 0.131*"магазин" + 0.040*"продавец" + 0.028*"купить" + 0.027*"покупка" + 0.024*"доставка" + 0.024*"верн
зврат"'),
 (4,
  '0.113*"заказ" + 0.086*"возврат" + 0.068*"средство" + 0.066*"деньга" + 0.046*"денежный" + 0.046*"день" + 0.038*"вернуть
ь"'),
 (3,
  '0.073*"договор" + 0.040*"выдача" + 0.040*"ковид" + 0.039*"пункт" + 0.031*"итог" + 0.028*"год" + 0.025*"предоставление"
е"'),
 (18,
  '0.073*"цена" + 0.049*"наличие" + 0.035*"правило" + 0.032*"продажа" + 0.031*"вакцина" + 0.031*"работник" + 0.028*"стать
луатация"'),
 (17,
  '0.167*"код" + 0.091*"заявка" + 0.081*"помещение" + 0.040*"яндекс" + 0.039*"книга" + 0.029*"маркет" + 0.029*"изделие" +
 (1,
  '0.372*"карта" + 0.238*"сумма" + 0.114*"ваш" + 0.051*"сутки" + 0.042*"перевести" + 0.036*"внести" + 0.016*"деньга" + 0.
о"')]
```

Задача 3.

сделать визуализацию кластеров тематик

На полученные темы можно посмотреть, изобразив их на плоскости с помощью библиотеки pyLDAvis. Чтобы спроецировать темы на плоскост

В [32]:
```
!pip install pyLDAvis
```

...

В [17]:
```
import pyLDAvis
```

В [20]:
```
# import pyLDAvis.gensim as gensimvis
import pyLDAvis.gensim_models as gensimvis
```

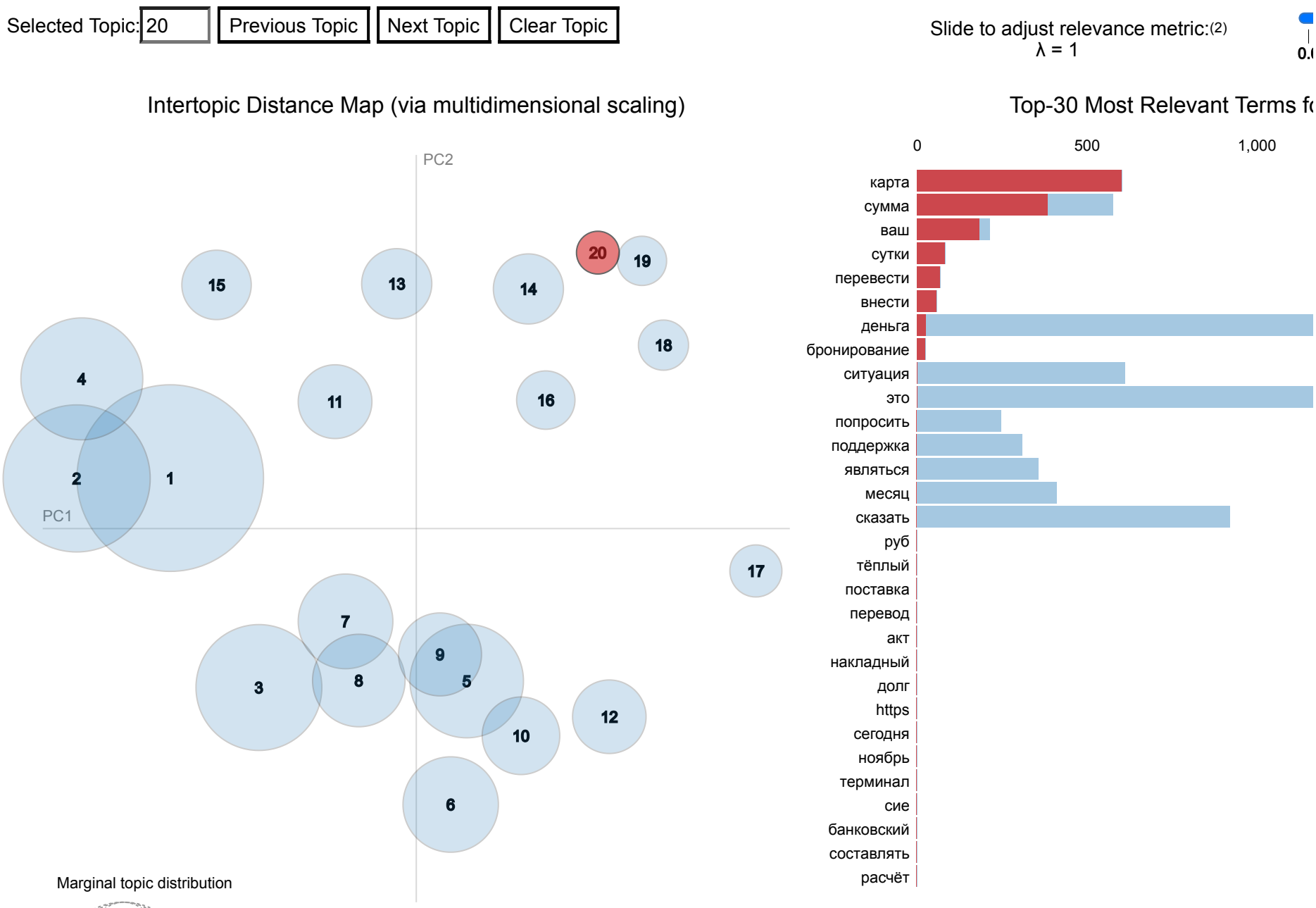
```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel\ipkernel.py:287: DeprecationWarning: `should_run_async` will not cal
se pass the result to `transformed_cell` argument and any exception that happen during thetransform in `preprocessing_exc
  and should_run_async(code)
```

B [21]:
```
%time vis_data = gensimvis.prepare(lda, corpus, dictionary)
pyLDAvis.display(vis_data)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel\ipkernel.py:287: DeprecationWarning: `should_run_async` will not cal
se pass the result to `transformed_cell` argument and any exception that happen during thetransform in `preprocessing_exc
  and should_run_async(code)
C:\ProgramData\Anaconda3\lib\site-packages\pyLDAvis\_prepare.py:246: FutureWarning: In a future version of pandas all arg
bels' will be keyword-only.
  default_term_info = default_term_info.sort_values(

Wall time: 5.24 s
```

Out[21]:



Посмотрим на метрики качества получившейся модели: перплексию и среднюю когерентность:

B [22]:
```
print(lda.log_perplexity(corpus))
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel\ipkernel.py:287: DeprecationWarning: `should_run_async` will not cal
se pass the result to `transformed_cell` argument and any exception that happen during thetransform in `preprocessing_exc
  and should_run_async(code)

-11.691879024670575
```

B [23]:
```
print('Персплексия: ', np.exp(lda.log_perplexity(corpus)))
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel\ipkernel.py:287: DeprecationWarning: `should_run_async` will not cal
se pass the result to `transformed_cell` argument and any exception that happen during thetransform in `preprocessing_exc
  and should_run_async(code)

Персплексия:  8.361376773967142e-06
```

In [25]:
```
coherence_model_lda = CoherenceModel(model=lda, texts=question_df['text'], dictionary=dictionary, coherence='c_v')
coherence_lda = coherence_model_lda.get_coherence()
print('Средняя когерентность: ', coherence_lda)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel\ipkernel.py:287: DeprecationWarning: `should_run_async` will not cal
se pass the result to `transformed_cell` argument and any exception that happen during thetransform in `preprocessing_exc
  and should_run_async(code)

Средняя когерентность:  0.3475513418552212
```

Теперь можно подобрать оптимальное количество тем, опираясь на значение метрик:

In [26]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel\ipkernel.py:287: DeprecationWarning: `should_run_async` will not cal
se pass the result to `transformed_cell` argument and any exception that happen during thetransform in `preprocessing_exc
  and should_run_async(code)
```

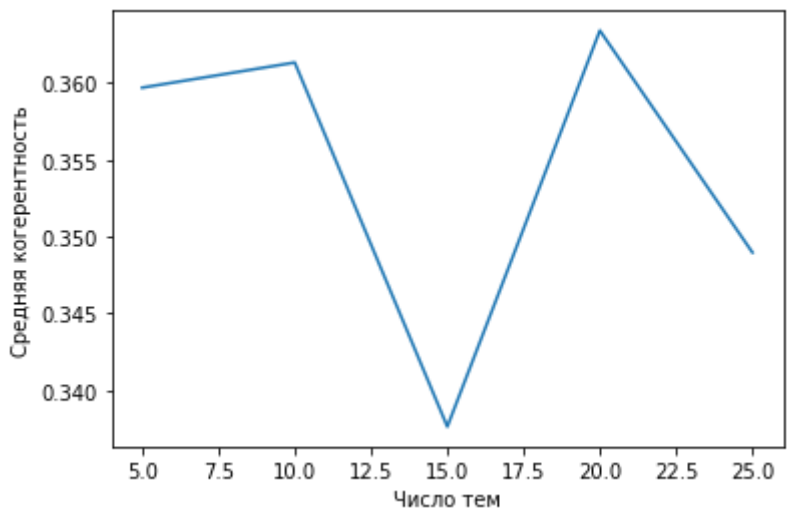
In [28]:
```
topics_list = [5, 10, 15, 20, 25]
coherences = []

for num in tqdm(topics_list):
    lda = ldamodel.LdaModel(corpus, id2word=dictionary, num_topics=num, chunksize=50, update_every=1, passes=2)
    coherences.append(CoherenceModel(model=lda, texts=question_df['text'], dictionary=dictionary, coherence='c_v').get_co

plt.plot(topics_list, coherences)
plt.xlabel("Число тем")
plt.ylabel("Средняя когерентность")
plt.show()
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel\ipkernel.py:287: DeprecationWarning: `should_run_async` will not cal
se pass the result to `transformed_cell` argument and any exception that happen during thetransform in `preprocessing_exc
  and should_run_async(code)

100%                                                5/5 [01:32<00:00, 18.59s/it]
```

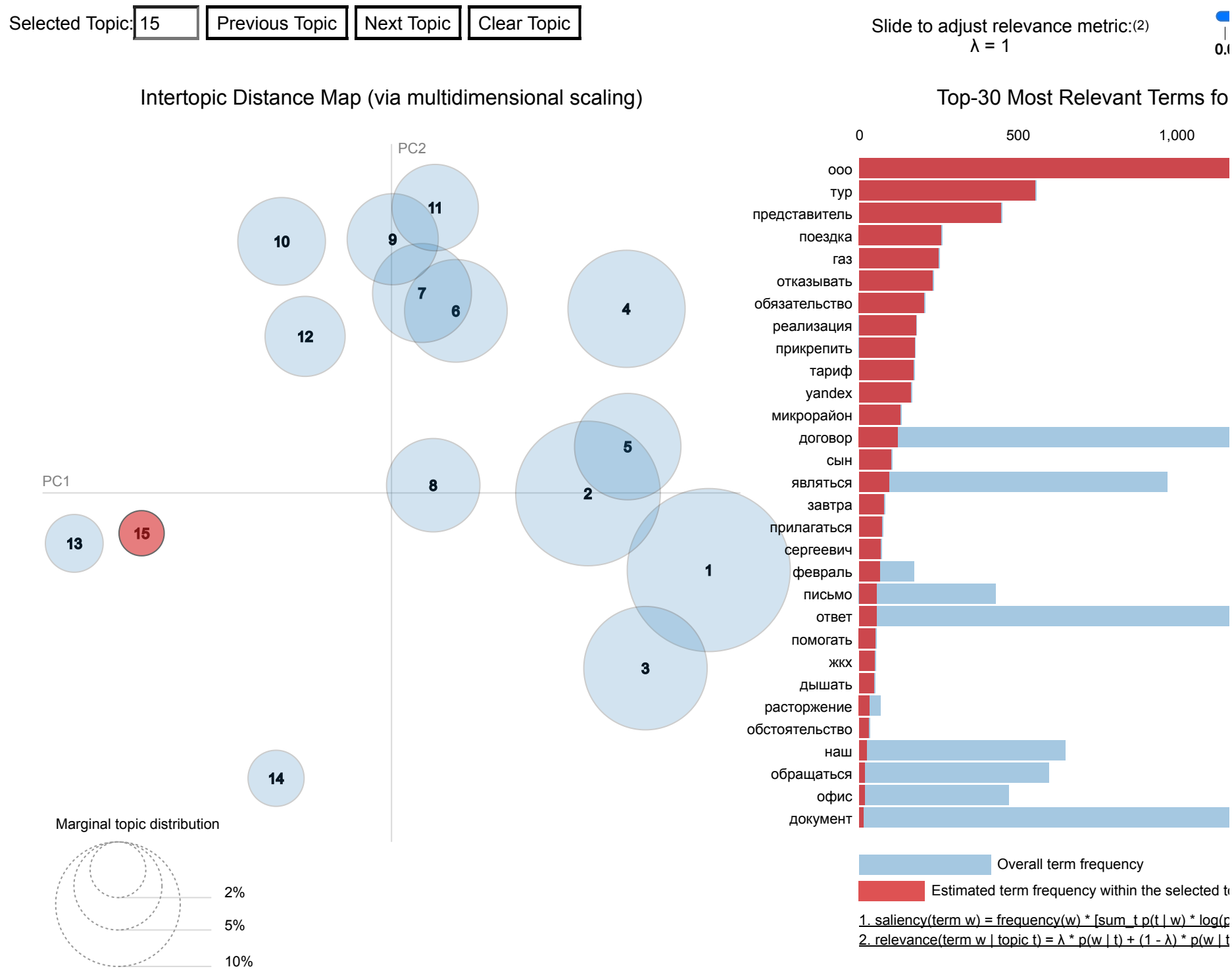


In [ ]:
```
```

B [29]:
```
lda_15 = ldamodel.LdaModel(corpus, id2word=dictionary, num_topics=15, chunksize=50, update_every=1, passes=2)
vis_data = gensimvis.prepare(lda_15, corpus, dictionary)
pyLDAvis.display(vis_data)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel\ipkernel.py:287: DeprecationWarning: `should_run_async` will not cal
se pass the result to `transformed_cell` argument and any exception that happen during thetransform in `preprocessing_exc
  and should_run_async(code)
C:\ProgramData\Anaconda3\lib\site-packages\pyLDAvis\_prepare.py:246: FutureWarning: In a future version of pandas all arg
bels' will be keyword-only.
  default_term_info = default_term_info.sort_values(
```

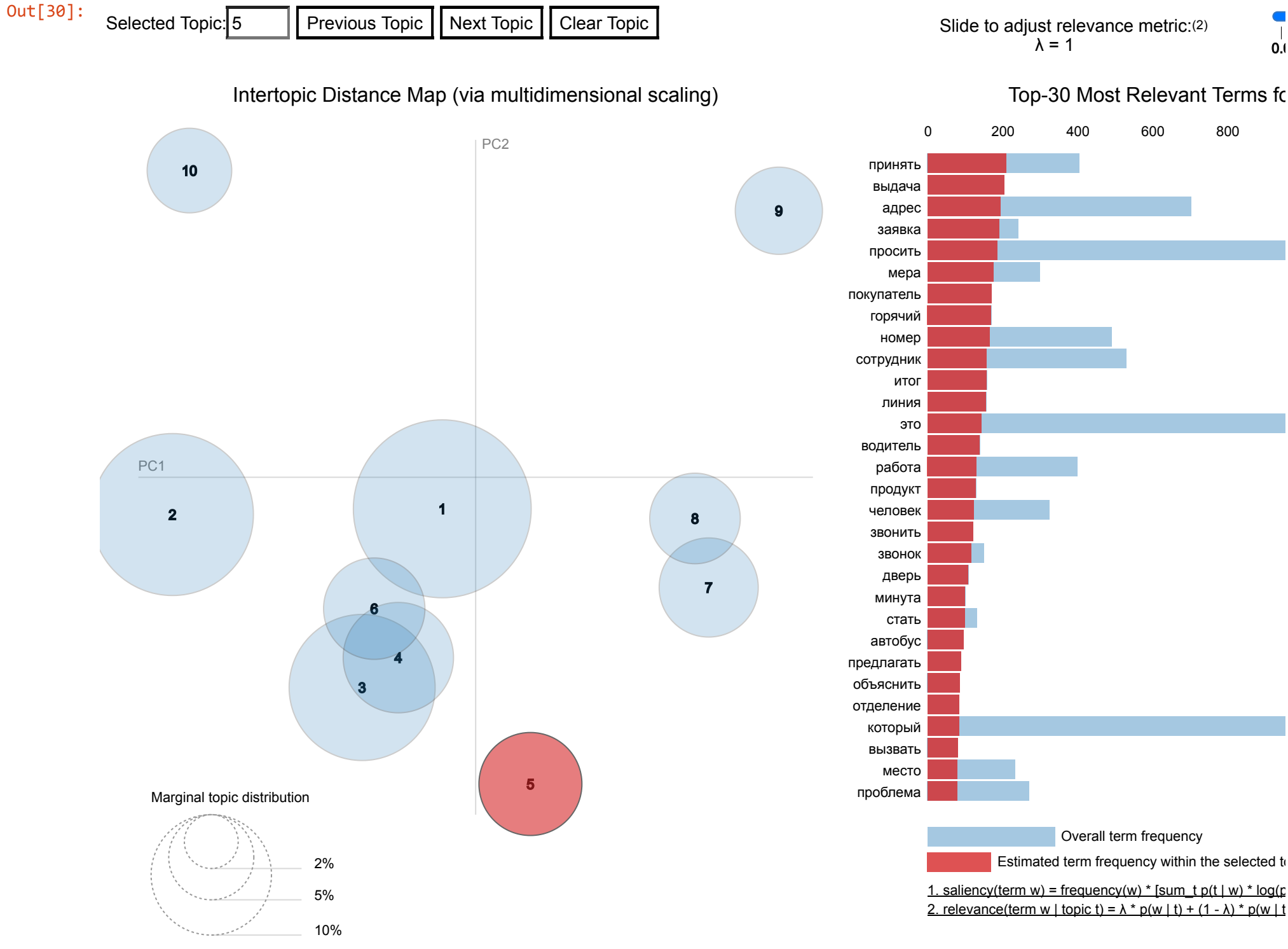
Out[29]:



B [30]:
```
lda_15 = ldamodel.LdaModel(corpus, id2word=dictionary, num_topics=10, chunksize=50, update_every=1, passes=2)
vis_data = gensimvis.prepare(lda_15, corpus, dictionary)
pyLDAvis.display(vis_data)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel\ipkernel.py:287: DeprecationWarning: `should_run_async` will not cal
se pass the result to `transformed_cell` argument and any exception that happen during thetransform in `preprocessing_exc
  and should_run_async(code)
C:\ProgramData\Anaconda3\lib\site-packages\pyLDAvis\_prepare.py:246: FutureWarning: In a future version of pandas all arg
bels' will be keyword-only.
  default_term_info = default_term_info.sort_values(
```

Out[30]:




```
Видно, что для качественной тематизации неоходима дополнительна предобработка тексто.<br>
В частности включить в стоп лист, слова общего значения, которые не определяют тематики.<br>
Такие как: это, дата, год, также, ооо, дать и т. д.
```

Задача 4.

проинтерпритировать получившиеся тематики

Проинтерпритируем несколько тематик

Тематика 1. Смысл достаточно чёткий. Можно назвать тематику "Правовая защита потребителя"

Тематика_1 = [право, договор, код, документ, рубль, дать, потребитель, закон, также, который, просить, информация, нарушение, связь, дейс
предоставить, год, требовать, дата, защита, ковид, сумма, ооо, медицинский, направить, обратиться]

Тематика 2. Смысл достаточно чёткий. Можно назвать тематику "Проблемы связанные с получением заказа".

Тематика_2 = [товар, заказ, деньга, сайт, интернет, вернуть, день, магазин, возврат, доставка, продавец, оплатить, получить, руб, оплата, отме
поддержка, указать, банк, заказать, сообщение, приобрести, пункт, ozon, отправить, кабинет]

Тематика 3 - достаточно сумбурная. Похоже на общий вопрос о помощи без чёткой формулировки проблемы.
Общий смысл - "Куда обратиться и что делать для решения проблемы".

Тематика 3 = [это, сказать, всё, день, мочь, ещё, ноябрь, делать, свой, написать, прийти, ситуация, дать, сегодня, который, время, купить, хотя
позвонить, решить, сделать, карта, добрый, здравствуйте, ответить, работать, упаковка]

Тематика 4. Смысл чёткий. Можно назвать тематику "Гарантийный ремонт и сервисное обслуживание".

Тематика 4 = [магазин, товар, цена, центр, купить, проверка, адрес, продажа, покупка, ремонт, продавец, данный, чек, просить, сервисный, об
здравствуйте, это, срок, кредит, обнаружить, продавать, качество, вход, наличие, жалоба]

Тематика 5. Смысл чёткий. Можно назвать тематику "Сервисное обслуживание клиентов".

Тематика 5 = [принять, выдача, адрес, заявка, просить, мера, покупатель, горячий, номер, сотрудник, итог, линия, это, водитель, работа, прод
стать, автобус, предлагать, объяснить, отделение, который, вызвать, место, проблема]

In [ ]: