



Сентимент-анализ и аспектная классификация отзывов

Бизнес-сегмент: отели

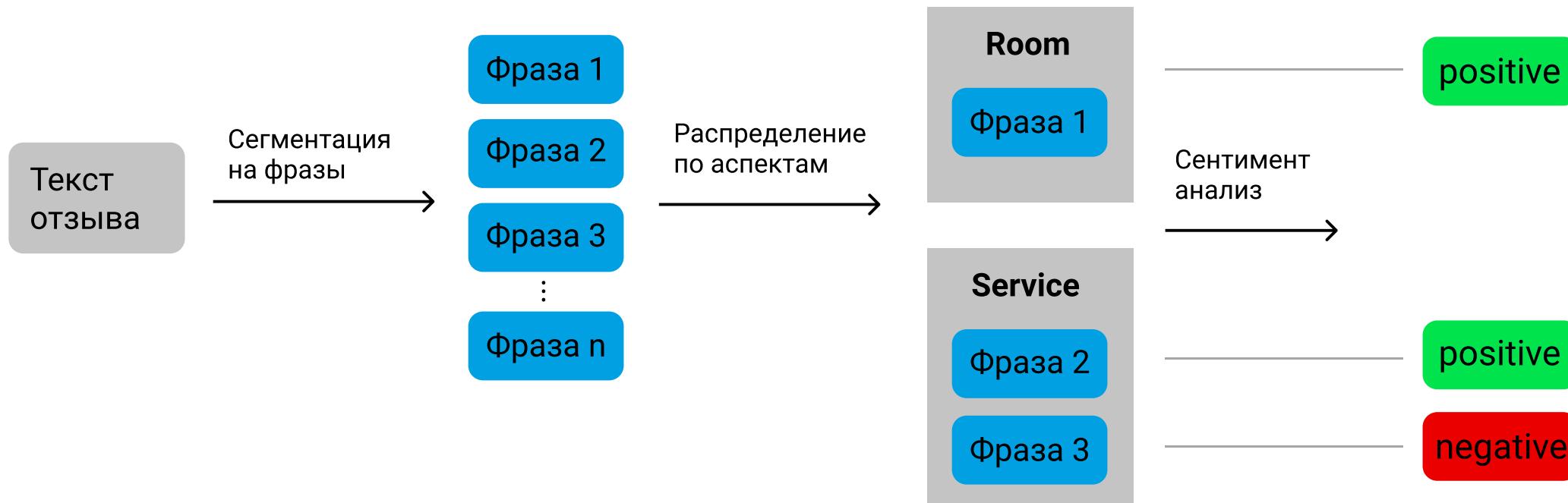
Зинкевич Олег

07 июня 2021

Цель и задачи:

Разработать систему анализа тональности и аспектной классификации отзывов.

1. Выявить критические для бизнеса аспекты, требующие улучшения (например, работа персонала, оборудование в комнате, оплата услуг и т.д.).
2. Направить входящее обращение в подходящий тематический сегмент для дальнейшей обработки (в случае использования чатбота, тикет системы).



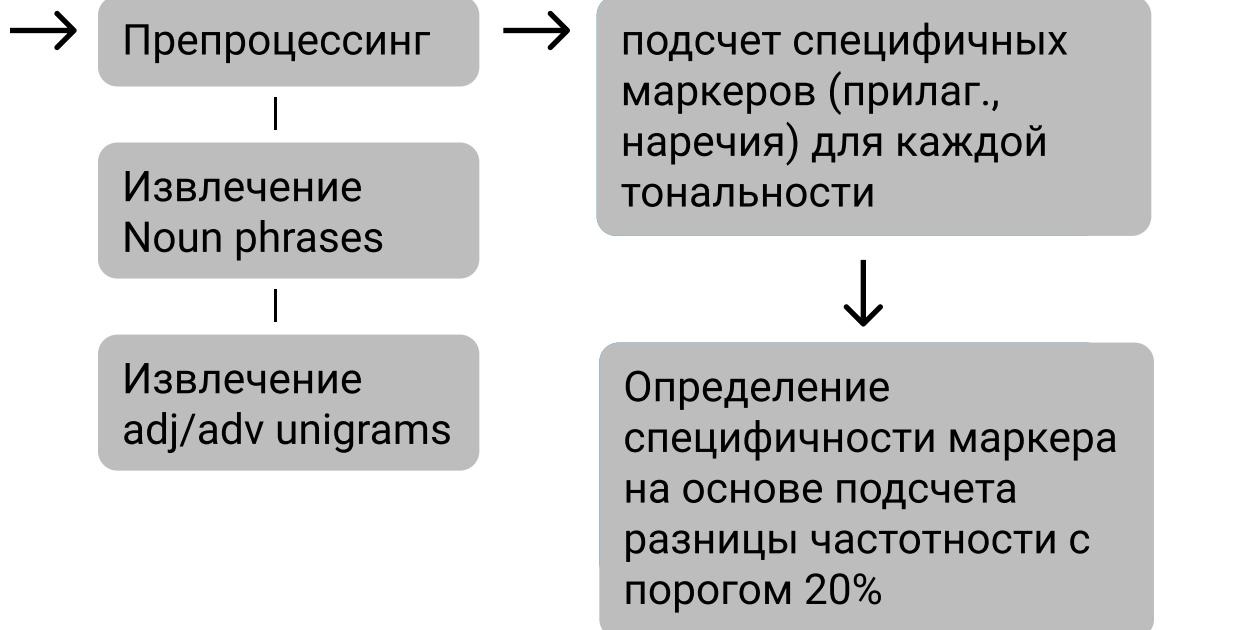
Архитектура проекта

database:
Postgresql

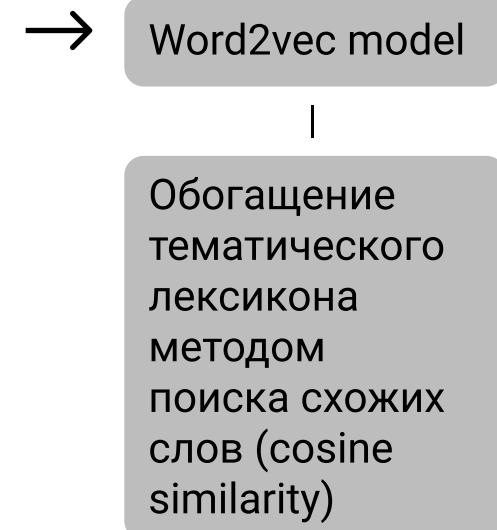
backend:
Django

frontend charts:
amcharts.js

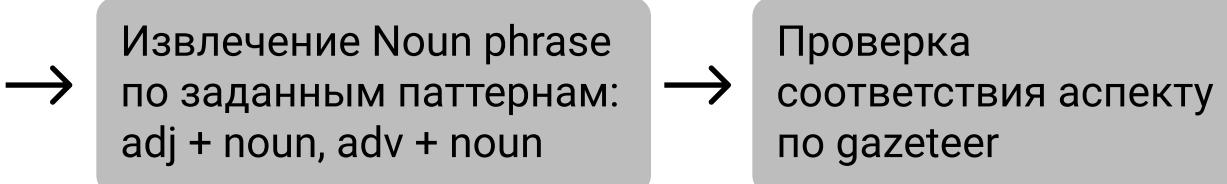
**Sentiment
lexicons
bootstrapping**



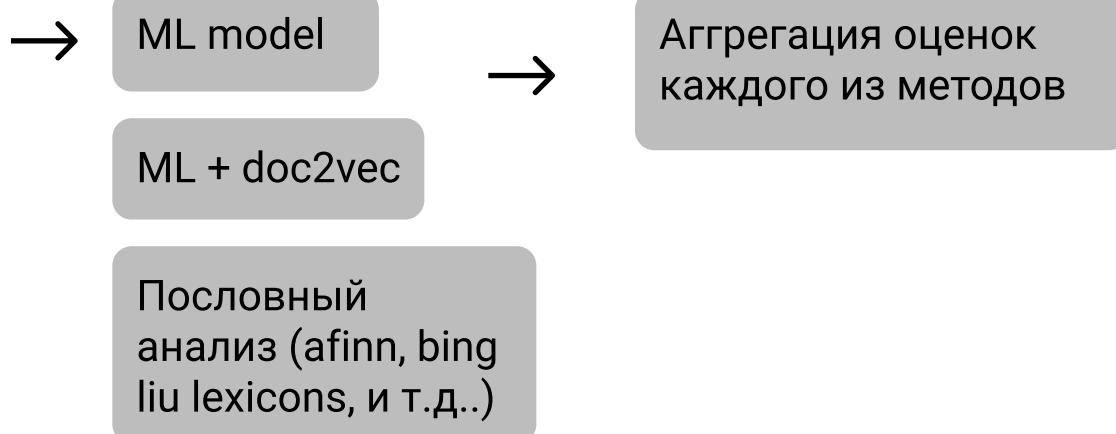
Bootstrap aspect/thematic lexicons



Определение тематического аспекта фразы



Sentiment classification



Рендеринг диаграмм + аналитика по каждому из аспектов

Используемые наборы данных

Основой для анализа и составления оценочного лексикона, а также тренировки ML моделей послужил:

Датасет (500 т. отзывов - tripadvisor, booking.com) с дальнейшей рандомизированной выборкой (40 000 отзывов), сбалансированной по количеству positive/negative категорий.

<https://www.kaggle.com/jiashenliu/515k-hotel-reviews-data-in-europe>

Дополнительные наборы данных (использовались для сравнения эффективности классификатора):

- **датасет** (4 т. отзывов - tripadvisor, booking.com) с тернарной классификацией категорий: negative/neutral/positive.
- **датасет** (“Sentiment 140 tweeter” - 1 млн. сообщений в tweeter) для сравнения эффективности классификации ML моделей для текстов, изобилующих сокращениями, сленгом, ошибками.

Сентимент лексиконы

AFINN (2011): <http://www2.imm.dtu.dk/pubdb/pubs/6010-full.html>

General Inquirer lexicon (Stone, 1968): http://www.wjh.harvard.edu/~inquirer/spreadsheet_guide.htm

Sentiment lexicon (Hu and Liu, 2004): <http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html>

MPQA subjectivity lexicon (Wilson et al., 2005): http://www.cs.pitt.edu/mpqa/subj_lexicon.html

SentiWordNet (Esuli and Sebastiani, 2006): <http://sentiwordnet.isti.cnr.it/>

Emotion lexicon (Mohammad and Turney, 2010): <http://www.purl.org/net/emolex>

Sentiment lexicon bootstrapping

Okay the hotel-motel may not be a - star appearance and the rooms are well they are rooms.. but for the money - I have a fridge, hot breakfast, free internet, nice pool and pool hang out area, quiet neighborhood day and night yet only blocks from stores, waterfront, and restaurants and a nice sized balcony and free milk and cookies at -pm. Not bad at all...

1. Препроцессинг

удаление #hashtags, usernames, urls, html тегов, двойных пробелов, спец. символов.

Автокоррекция опечаток: **calcualte > calculate**.

Нормализация сокращений: **Mr > mister**

POS теггинг, токенизация, лемматизация.

2. Извлечение Noun phrases

для каждого сегмента (positive/negative)
aspect word (noun, gerund) + adjective
aspect word (noun, gerund) + adverb

3. Извлечение только adj/adv

для каждого сегмента (positive/negative)

См. скрипты в папке:

https://github.com/olegzinkevich/hse_computational_linguistics/tree/master/final_project_brand_aspect/bootstrap_lexicon

Sentiment lexicon bootstrapping

1. Извлечение и подсчет специфичных маркеров (прилаг., наречия) для каждого сегмента отзывов (positive, negative)

word	neg_adj_count	pos_adj_count
clean	2929.0	3962.0
strong	115.0	66.0
average	494.0	196.0
overnight	95.0	108.0
fussy	18.0	18.0

а) Определение специфичности маркера на основе подсчета разницы частотности с **порогом 20%**.

```
def percent_difference_calc(x, word_type):
    if x['neg_{word_type}_count'] > x['pos_{word_type}_count']:
        increase = x['neg_{word_type}_count'] - x['pos_{word_type}_count']
        result = (increase / x['neg_{word_type}_count']) * 100
        # threshold 20%
        if result < 20:
            non_specific_words.append(x['word'])
        else:
            negative_specific_words.append(x['word'])
    return result
```

б) Определение специфичности маркера с помощью метрики **PMI**.

```
noun_cnt = all_nouns_counter.get(splitted[1], 0)
# nouns_count.append(noun_cnt)

prob_collocation = phrase_cnt / corpus_tkns_count
prob_identifier = identifier_cnt / corpus_tkns_count
prob_noun = noun_cnt / corpus_tkns_count

try:
    pmi = np.log2( prob_collocation / (prob_identifier * prob_noun) )
except:
    pmi = 0.00
pmi_metric.append(pmi)
```

Ссылка на скрипт:

<https://colab.research.google.com/drive/13o8p7tz9UpF9slavVwueqaooaW8Inyga?usp=sharing>

Маркеры, специфичные для positive sentiment

adjectives: https://drive.google.com/file/d/1RMyxSi2ILGGmjiQoqmROXITM9a-v_FB5/view?usp=sharing

adverbs: https://drive.google.com/file/d/1Do6ayy12UAk_K40V024dNOREdIPePT6V/view?usp=sharing

Маркеры, специфичные для negative sentiment

adjectives: https://drive.google.com/file/d/1qH3ulJ_EDhKlmikk6cjslePZkcTyrIYi/view?usp=sharing

adverbs: https://drive.google.com/file/d/190IZzZxFHS7cvH_GSwvPDNDInaU4iD2/view?usp=sharing

Набор stop words

1. Неспецифичные слова с порогом < 20% послужили основой создания словаря stop слов.

Существующая проблема: стандартные списки стоп слов - sklearn, nltk, terrier и т.д. не подходят для задачи сентимент-классификации, т.к. содержат слова отрицания, слова с эмотивной оценкой (better, worse и т.д.). Пример: разница f1_score ML модели: **0.81** (стандартный список стоп слов) vs. **0.85** (созданный для данного бизнес домена).

<https://drive.google.com/file/d/1Cbmxbu5DOacQBAL2Xm7V6laNsFg8KZ50/view?usp=sharing>

Bootstrap aspect/thematic lexicons

- word2vec

Bath:	bathtub, towel, shampoo...	Room:	suite, hall, apartment...	Payment:	receipt order money...
					• • •

Тренировка word2vec модели на основе 250 000 отзывов.

```
1 model = gensim.models.Word2Vec (documents, size=150, window=10, min_count=2, workers=10)
2 model.train(documents, total_examples=len(documents), epochs=10)
```

Алгоритм набора тематических слов:

start_word (пример: “payment”) > поиск слов (cosine vector similarity) (“transaction”, “visa”...) > поиск схожих векторов (слов) для каждого найденного до заданного лимита, например, 1000 слов..

```
[('transaction', 0.7008122801780701), ('visa', 0.686706006526947), ('receipt', 0.6855411529541016), ('documentation', 0.65851061),
```

Ссылка на скрипт:

https://colab.research.google.com/drive/1PvRLHWaSz-WvhEO_6KS-hpx7p9IXZuDz?usp=sharing

Формирование уникальных тематических списков из полученных массивов
(посредством анализа пересечения, разницы и объединения множеств).

https://drive.google.com/file/d/1IMI5jnrMEqHnb9AOQvjkSrs_3pEKDSA-/view?usp=sharing

Определение аспекта фразы

Извлечение Noun phrase по заданным паттернам: adj + noun, adv + noun.

```
grammar = "NP: {(<JJ>+<NN>) | (<JJ>+<NNS>) | (<JJR>+<NN>) " \
           "| (<JJR>+<NNS>) | (<JJS>+<NN>) | (<JJS>+<NNS>) | " \
           "| (<RB>+<NN>) | (<RB>+<NNS>) | (<JJ>+<VBG>) | " \
           "| (<JJR>+<VBG>) | (<JJS>+<VBG>)}"

chunk_parser = nltk.RegexpParser(grammar)
tree = chunk_parser.parse(pos_tokens)
```

```
nouns = []
nouns_chunked = []
for subtree in tree.subtrees(filter = lambda t: t.label() == 'NP'):
    ls = []
    for item in subtree.leaves():
        (word, tag) = item
        if tag == 'NN' or tag == 'NNS':
            nouns.append(word)
            lemmatized_noun = lemmatizer.lemmatize(item[0]).lower()
            ls.append(lemmatized_noun)
    ls = ' '.join(ls)
    nouns_chunked.append(ls)
```

Алгоритм определения категории для фразы на основе найденного аспекта и газеттера

- Если существительное (аспект) имеет определитель/маркер, алгоритм назначает тематический аспект для данного существительного (на основе тематического лексикона/газеттера).
- Если аспектных слов несколько, назначается аспект, имеющий макс. кол-во вхождений тематических слов.
- Если не найдены маркированные аспекты, алгоритм назначает категорию по найденному сущ. или отглагольному сущ. (например, staying).
- Если и в этом случае не найдена категория, назначается категория 'general'.

Ссылка на скрипты:

Препроцессинг данных:

https://github.com/olegzinkevich/hse_computational_linguistics/blob/master/final_project_brand_aspect/data_process/utils.py

Классификация тематического аспекта:

https://github.com/olegzinkevich/hse_computational_linguistics/blob/master/final_project_brand_aspect/bootstrap_lexicon/classify_review_sentence_func.py

Sentiment classification

Используемые подходы:

1.
ML модель

max f1_score:
SVC: **0.869**

2.
ML + doc2vec

max f1_score:
doc2vec + logistic
regression: **0.865**

3.
**Пословный анализ на
основе эмотивных
лексиконов**

Afinn
Bing Liu lexicon
MPQA
Vader
Textblob
SentiWordNet

max f1_score:
Textblob: **0.631**
Afinn: **0.663**
Vader: **0.498**

4.
**Агрегация оценок
каждого из методов и
выявление среднего
показателя (-1, 0, 1).**

Метод классификации на основе лексикона неэффективен, не учитывает специфику бизнес домена, отрицание за границей фразы, сарказм и т.д. См:
https://colab.research.google.com/drive/1CB_dEhR_Fiy-JgDeKs-HLghx6JL0Pwuu?usp=sharing

ML модель классификации

- Препроцессинг (лемматизация, токенизация, удаление #hashtags, username и т.д.).
- Исключение из feature space непризнаковых, неспецифичных слов.
- Кросс-валидация, hyperparameter tuning.

Наиболее эффективна модель SVC (Support Vector Classification):

- TF-IDF: max_features 10 000, порог частотности 20%, используется собственный словарь stop words.

```
0.8697125643278524
precision    recall   f1-score   support
0            0.88      0.86      0.87      3973
1            0.86      0.88      0.87      3994
accuracy          0.87      0.87      0.87      7967
macro avg       0.87      0.87      0.87      7967
weighted avg     0.87      0.87      0.87      7967
[[3422  551]
 [ 487 3507]]
```

Ссылка на скрипт:

<https://colab.research.google.com/drive/1Z4F1A9idzBF5GZLDNqOyb2NbI8I2tVS?usp=sharing>

Сравнение ML моделей

Макс. показатель: SVC, tf-idf: 0.869

CountVectorizer (bag of words):

CountVectorizer, ngram = (1,1)

```
[('log_reg', 0.8466056634104118), ('naive_bayes', 0.8387535550884135), ('linear_svc', 0.8346111042413751), ('extra_trees', 0.8313342401384939), ('grad_boost', 0.8184740942252999), ('random_forest', 0.8165574378632373), ('sgd', 0.7849635217014962), ('decision_trees', 0.6938914306912328)]
```

CountVectorizer, ngram = (2,2)

```
[('naive_bayes', 0.856683566217386), ('log_reg', 0.8471621120316557), ('sgd', 0.8383207617163349), ('extra_trees', 0.8275627550389514), ('linear_svc', 0.817361196982812), ('random_forest', 0.8045628786942005), ('grad_boost', 0.7790898973661431), ('decision_trees', 0.6922839124520836)]
```

CountVectorizer, ngram=(3, 3)

```
[('naive_bayes', 0.79108445653518), ('log_reg', 0.7713614442933103), ('linear_svc', 0.7691974774329171), ('sgd', 0.7663534067021145), ('extra_trees', 0.755595400024731), ('random_forest', 0.7491034994435514), ('grad_boost', 0.7023618152590577), ('decision_trees', 0.6575986150612094)]
```

Тренировка модели на неспецифичном датасете (сообщения в tweeter)

CountVectorizer (bag of words):

```
[('linear_svc', 0.7026652452025586), ('log_reg', 0.6992537313432836), ('sgd', 0.6955223880597015), ('decision_trees', 0.6486140724946695)]
```

Doc2vec

- Препроцессинг (лемматизация, токенизация, удаление #hashtags, username и т.д.).
- Исключение из feature space непризнаковых, неспецифичных слов.

Параметры: distributed bag of words (PV-DBOW), vector_size=300, iter=20, min_count=2

Logistic reg:|

0.8650103519668737

	precision	recall	f1-score	support
0	0.87	0.85	0.86	1176
1	0.86	0.88	0.87	1239
accuracy			0.87	2415
macro avg	0.87	0.86	0.86	2415
weighted avg	0.87	0.87	0.86	2415

[[1000 176]
[150 1089]]

Ссылка на скрипт:

<https://colab.research.google.com/drive/1JCm1otUQVLZXeymM8ea9SW9jPlyeH14t?usp=sharing>

Ссылки на модели

Doc2vec:

<https://drive.google.com/file/d/1e8rN1UMa81Md6b2ZhcaGidOaEEnhmtVs/view?usp=sharing>

Logistic regression model trained on Doc2vec:

<https://drive.google.com/file/d/1-7nNSBo2iP8Fr00-mZPc8SogCT6ZuciZ/view?usp=sharing>

Word2vec:

https://drive.google.com/drive/folders/1WIkmpz_AGi2Tvg2tslGiWVp7Lu-4NWKK?usp=sharing

SVC tf-idf vectorizer:

https://drive.google.com/file/d/1U0FLH8u5f8ncN7W415d015rM_-H81ZMn/view?usp=sharing

SVC tf-idf model:

https://drive.google.com/file/d/1--0wVsGmAue0zYgJEXsU4x_b9uhme-Fy/view?usp=sharing

Выводы и дальнейшая стратегия

Метод классификации, основанный на пословном анализе, не эффективен, т.к. не учитывает :

- **инверторы** (лексику, трансформирующую позитив в негатив и наоборот);
- **отрицание** всей фразы;
- **интенсификаторы** (“очень”, “совсем” в контексте сарказма, иронии;
- **сравнения**: “good climate (not moist like in other places)”.
- **контекстуальные фразы**, характерные для бизнес домена, но не включающие слова с эмотивной оценкой (например: “в ванной была вода на полу”, “по комнате были разбросаны вещи”).
- **контекст с выражением желания, намерения** (“I was looking for a good hotel to order”), в которых слова с положительными коннотациями не являются маркерами эмоциональной реакции.

Метод, основанный на тренировке ML модели:

- требует релевантности признакового пространства и лексикона, характерного для бизнес-домена. Модель, работающая для домена “отели” будет неэффективна для домена “рестораны”, “авиалиний”.
- метод doc2vec более эффективен для анализа текстов, аналогичных по размеру тренируемым.

Список использованной литературы

Указаны основные источники:

Bing Liu, Guang Qiu, Jiajun Bu, Chun Chen. Opinion Word Expansion and Target Extraction through Double Propagation // Computational Linguistics. - March 2011, - Vol. 37, - Issue 1.

Bing Liu. Sentiment Analysis and Subjectivity [Electronic resource]. - URL:
<https://www.cs.uic.edu/~liub/FBS/NLP-handbook-sentiment-analysis.pdf> (date of treatment: 10.04.2021)

Bing Liu. Sentiment Analysis: Mining Opinions, Sentiments, and Emotions // - 2015, - Cambridge University Press.

Chris Manning, Hinrich Schütze. Foundations of Statistical Natural Language Processing // - 1999, - June 18, - The MIT Press.

Dan Jurafsky, James H. Martin. Speech and Language Processing [Electronic resource]. - URL: <https://web.stanford.edu/~jurafsky/slp3> (date of treatment: 01.03.2021)

Ellen Riloff, Rosie Jones. Learning Dictionaries for Information Extraction by Multi-Level Bootstrapping // Proceedings of the Sixteenth National Conference on Artificial Intelligence. - 1999, - July 18-22.

Jun Feng, Cheng Gong, Xiaodong Li, Raymond Y. K. Automatic Approach of Sentiment Lexicon Generation for Mobile Shopping Reviews. // Wireless Communications and Mobile Computing. - 2018. - Vol 1, - P. 1-13.

Peter D. Turney. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews // Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), - 2002, - Philadelphia, - P. 417 -424.

Qian Liu, Zhiqiang Gao, Bing Liu and Yuanlin Zhang. Automated Rule Selection for Aspect Extraction in Opinion Mining. // Proceedings of International Joint Conference on Artificial Intelligence. - 2015, - July 25-31.