Естественный отбор: проранжируй комментарии с помощью ML

Команда «doriam»
Мышкина Анастасия

Улучшение качества комментариев под постами

Задача

Ранжирование комментариев к постам по их качеству с целью:

- Повышения общего восприятие поста в целом
- Экономии времени (пользователь может сразу сконцентрироваться на лучших комментариях)
- Улучшение качества дискуссии (ранжирование позволяет выводить в топ комментарии с глубокими мыслями)

Необходимо создать механизм сортировки комментариев к постам по их популярности на основе методов машинного обучения

Создание модели

У нас есть 2 jsonl файла: train/test, в каждом из которых содержится текст поста и комментариев под ним , В test есть значения `score` (позиция ранжирования). Их нужно предсказать для test

Этапы:

1. Провести EDA
2. Обработка текста
3. Векторизация текста
4. Проведение экспериментов для выбора лучшей модели
5. Обучение модели
6. Предсказание значения `score` в тестовой выборке

Анализ результатов

После обучения модели мы сможем ответить на вопросы:

- Какие комментарии чаще становятся популярными?
- Какие комментарии чаще становятся неинтересными?
- Что можно сделать с этой информацией

Нужно подумать над механизмом взаимодействия с комментаторами, чтобы мотивировать их писать интересные комментарии

Данные содержат тренировочную (ranking_train.jsonl) и тестовую (ranking_test.jsonl) выборки.
Каждый JSON имеет поле text с текстом поста и массив `comments` из 5 элементов. Каждый элемент массива это словарь с ключами `text` с текстом комментария и `score` с позицией ранжирования (0 соответствует самому популярному комментарию, 4 самому непопулярному). В тестовом файле в `score` стоит null. Его нужно предсказать.

train

Содержит:
- 88107 постов
- 440535 комментариев

test

Содержит:
- 14004 постов
- 70020 комментариев

Сначала мы привели полученные данные к таблице со столбцами:
- `Текст поста`
- `Текст комментария`
- `score`

Предобработка текста
Включает в себя:
- Удаление знаков препинания
- Удаление лишних пробелов
- Приведение к нижнему регистру
- Удаления \n \t (символы переноса строки и табуляция)
- Удаление ссылок
- Удаление стоп слов

После этого мы сделали стемминг для приведения слов к основе

	post_text	com_text	score
0	many summer combinator fundees decided continu...	going back school identical giving founders go...	0
1	many summer combinator fundees decided continu...	invariably dont see success set fall back orig...	1
2	many summer combinator fundees decided continu...	school way connected going real world entered ...	2
3	many summer combinator fundees decided continu...	guess really depends hungry much believe produ...	3
4	many summer combinator fundees decided continu...	know pollground decided go back school getting...	4

Рис. 1. Таблица после предобработки для тренировочной выборки

Ознакомиться с кодом можно по ссылке:
https://github.com/nastymauz/Cup_IT_VK/blob/main/preprocess.ipynb

Добавим **новые признаки** в данные:

- Количество общих слов в посте и в комментарии (`common_words`)
- Отношение числа слов в посте к общей длине поста (`ratio_post`)
- Длина поста (`len_of_post`)
- Длина комментария (`len_of_coms`)
- Общая длина поста и комментария (`len_of_text`)

Для векторизации текста сначала мы используем метод **TF-IDF** для определения наиболее важных терминов в тексте, а затем выполним сингулярное разложение матрицы (**SVD**) термин-текст для сжатия данных и выделения наиболее значимых факторов или тем. Это позволяет улучшить качество модели и уменьшить время обучения.

	score	common_words	len_of_post	len_of_comm	len_of_text	ratio_post	post_text_svd	com_text_svd
0	0.0	6	11	96	107	0.545455	0.017271	0.118448
1	1.0	3	11	34	45	0.272727	0.017271	0.059355
2	2.0	2	11	44	55	0.181818	0.017271	0.078515
3	3.0	1	11	31	42	0.090909	0.017271	0.071116
4	4.0	5	11	9	20	0.454545	0.017271	0.046326

Рис. 2. train после добавления новых признаков и векторизации

	score	common_words	len_of_post	len_of_comm	len_of_text	ratio_post	post_text_svd	com_text_svd
0		1	8	16	24	0.125	0.003099	0.034419
1		0	8	10	18	0.000	0.003099	0.018142
2		1	8	20	28	0.125	0.003099	0.029268
3		2	8	25	33	0.250	0.003099	0.054525
4		0	8	4	12	0.000	0.003099	0.013465

Рис. 3. test после добавления новых признаков и векторизации

Ознакомиться с кодом можно по ссылке:
https://github.com/nastymauz/Cup_IT_VK/blob/main/preprocess.ipynb (добавление новых признаков)
https://github.com/nastymauz/Cup_IT_VK/blob/main/vector.ipynb (векторизация)

В ходе решения задачи выбора модели для ранжирования комментариев были рассмотрены модели:

- Логистическая регрессия
- Решающее дерево
- Градиентный бустинг
- Дерево решений
- CatBoost

Целевая метрика: NDCG (Normalized Discounted Cumulative Gain), она хороша тем, что позволяет оценивать качество персонализированных рекомендаций.
В отличие от других метрик, таких как точность (precision) или полнота (recall), NDCG учитывает не только правильность порядка рекомендаций, но и их релевантность и порядок в списке рекомендаций.

Модель	*LogisticRegression*	*DecisionTree*	*GradientBoosting*	*RandomForest*	*CatBoost*
NDCG	*0.6289*	*0.6026*	*0.6313*	*0.6117*	*0.6303*

	precision	recall	f1-score
0.0	0.34	0.55	0.42
1.0	0.23	0.16	0.19
2.0	0.22	0.13	0.16
3.0	0.23	0.12	0.16
4.0	0.29	0.46	0.35

Рис. 4. метрики precision, recall, f1-score для градиентного бустинга. Он неплохо предсказывает комментарии со `score` 0 и 4

Лучше всего NDCG метрика оказалась у градиентного бустинга. Кроме этого, стандартные метрики у него тоже лучше всех (рис. 4). Будем использовать его. Он позволяет снизить ошибку модели на каждой итерации, комбинируя решения базовых моделей. *(значения метрик precision, recall, f1-score всех рассмотренных моделей можно найти в ноутбуке)*
Будем обучать на тренировочной части, разделив датасет в соотношении 85:15

Ознакомиться с кодом можно по ссылке:
https://github.com/nastymauz/Cup_IT_VK/blob/main/model_building.ipynb
https://github.com/nastymauz/Cup_IT_VK/blob/main/to_jsonl.ipynb

Проанализировав комментарии с высокой и низкой оценкой можно сделать вывод:

Интересные комментарии пользователям обычно содержат:
- Вопросы по теме поста, на которые отвечают другие комментаторы и ветка ответов становится интересной
- Грамотное высказывание своей точки зрения,
- Дополнительную информацию к теме поста, которая может быть полезна или интересна для других читателей

Неинтересные же комментарии обычно содержат:
- Несвязанную с темой поста информацию
- Однословные, неисчерпывающие комментарии
- Слишком длинный текст
- Ненормативную лексику
- Спам

Исходя из этих инсайтов можно сделать памятку для начинающих комментаторов, чтобы они знали в теории, какие комментарии могут оказаться неудачными.

\tWikidata, the free knowledge base that anyone can edit	I'm not big into the Wiki world, but it's also struck me as odd how different pages refer to the same facts and yet are totally disparate. If one page updates, does somebody manually have to go update the other page? Does a bot do it?This looks like a great response to that. I just hope they've made it easy to interface with.

Рис. 4. Пример удачного комментария (вопрос по теме поста)

Ask HN: Freelancer? Seeking freelancer? (February 2012)	SEEKING WORK (telecommute) - Chennai, India.I'm an M.Sc. in computer science, and a member of ACM and the GNOME Foundation.I'm good with C, POSIX programming environment (esp. on Linux), the GNU toolchain and other development tools like automake, valgrind, etc.I like computer graphics, so much that I've been a GIMP developer for nearly 10 years now. I also write code for GEGL and Raster.A sample graphics article:\nhttps://banu.com/blog/6/flower-disk-sampling-for-the-thin-le...I also have experience with network programming (bsd-sockets) and the POSIX programming environment in general. Some articles which made it to Hacker News frontpage:https://banu.com/blog/2/how-to-use-epoll-a-complete-example-...\nhttps://banu.com/blog/7/drawing-circles/I have been creating

Рис. 5. Пример неудачного комментария (спам и слишком длинный текст)

Наша модель может помочь комментаторам оценивать насколько их комментарий будет интересен для других пользователей в момент его написания. Это может повысить качество дисскуссий под постами и отсечь глупы, неинтересные комментарии.

Для этого нужно мотивировать пользователей переписывать комментарии, которые были определены нашим сервисом как неудачные.

Мы предлагаем идею:

Создание системы рейтинга для комментаторов. Поощрять пользователей за полезные и интересные комментарии. ВКонтакте часто мотивирует пользователей участвовать в новых сервисах посредством мини-приложений, в которых можно получать статусы, которые будут видны в профиле, и в которых будет топ друзей комментаторов, разные задания (например написать 5 комментариев с высоким рейтингом), розыгрыши и тд.

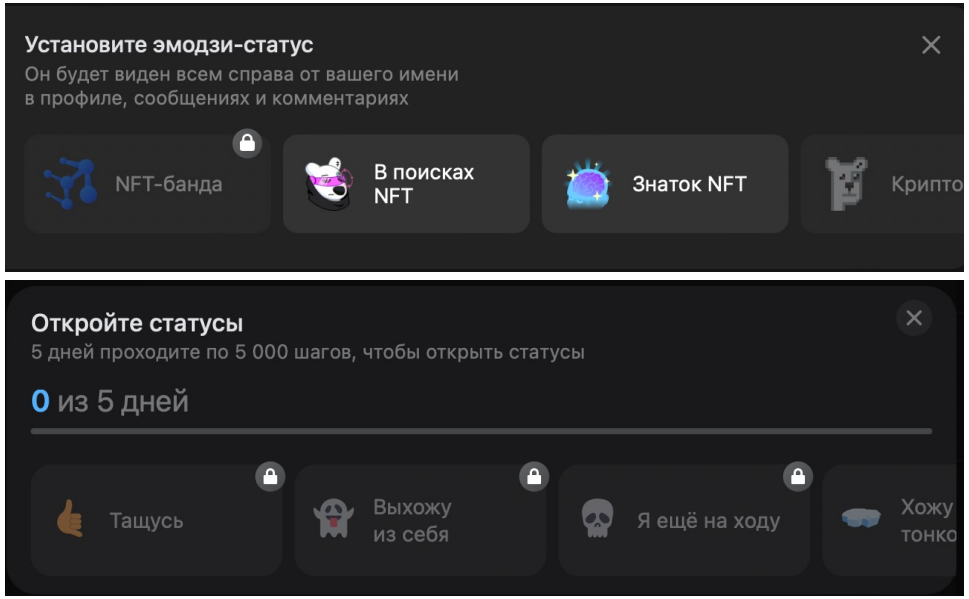

Рис. 6. Пример статусов в мини-приложениях, которые разблокируются при выполнении заданий

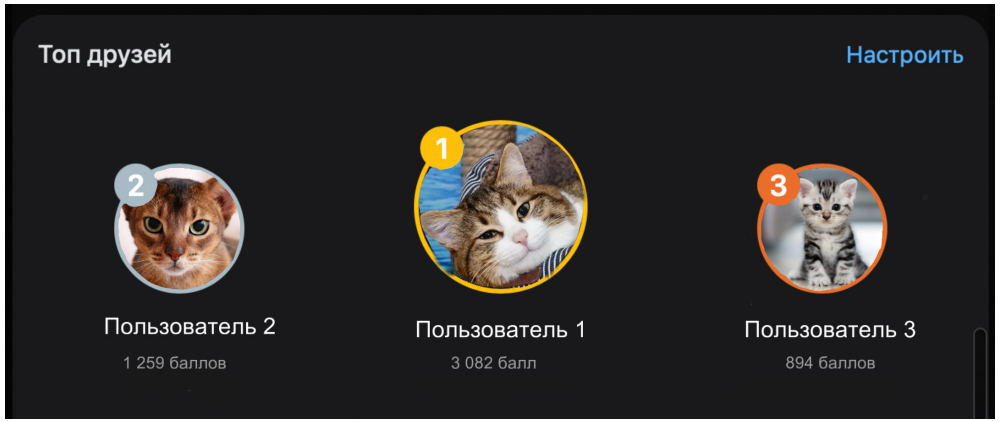

Рис. 7. Топ друзей комментаторов в мини-приложении

Мышкина Анастасия

Бизнес-информатика,
РЭУ им. Г. В. Плеханова

manaastya.0111@gmail.com