



Инструмент для анализа клиентских отзывов

Проект в рамках всероссийской научно-технологической программы по решению проектных задач в области искусственного интеллекта и смежных дисциплин «Сириус.ИИ»

АО «Тинькофф Банк»

Март, 2024

Проектная команда «SpectrUM» :

71 регион



Кончаков Павел
10 класс
Капитан

71 регион



Шабает Андрей
10 класс

71 регион



Григорьев Илья
10 класс



71 регион



Ведешкин Тимофей
10 класс

Актуальность проекта:

1. Конкурентоспособность и адаптация:

Разработка инструмента для анализа клиентских отзывов позволит банку Тинькофф повысить свою конкурентоспособность за счет адаптации своих продуктов и услуг под потребности клиентов. Это особенно важно в контексте острой конкуренции на рынке банковских услуг.

2. Улучшение обслуживания и удовлетворенности клиентов:

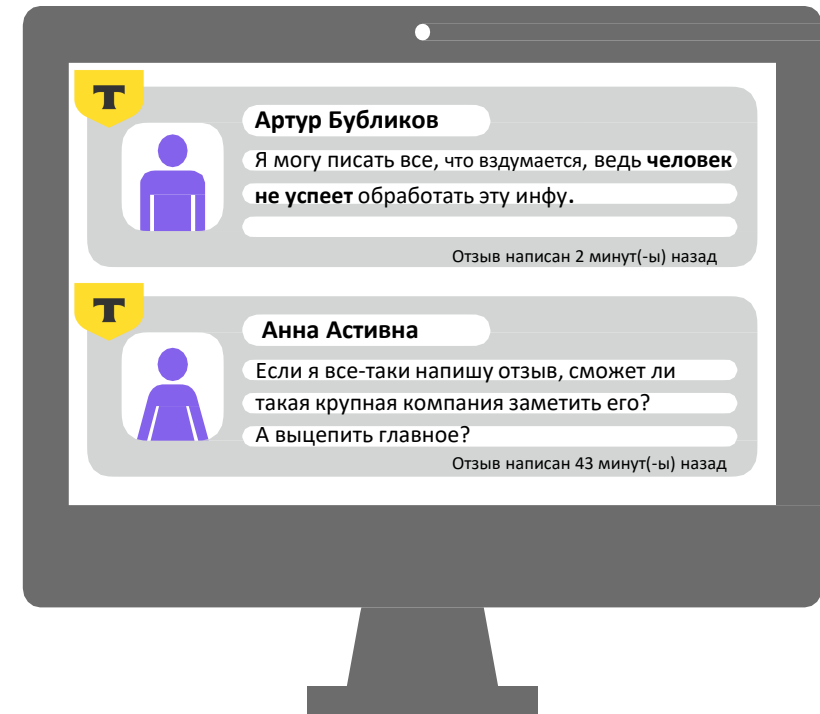
Путем анализа отзывов пользователей банк сможет выявить слабые места в обслуживании и недостатки продуктов, что позволит ему принимать меры для их улучшения. Это в свою очередь повысит уровень удовлетворенности клиентов и способствует повышению лояльности.

Проблематика проекта:

Возможное отсутствие оперативной реакции на изменения рыночной среды:

Без эффективного инструмента анализа клиентских отзывов банк может быть лишен возможности оперативно реагировать на изменения в требованиях рынка и предпочтениях клиентов, что затрудняет его конкурентоспособность и способность к инновациям. Отсутствие достаточного времени для выявления главной темы отзывов.

Недостаточная адаптация продуктов под потребности клиентов:
Отсутствие систематического анализа клиентских отзывов может приводить к тому, что банк не полностью учитывает потребности и ожидания своих клиентов при разработке и улучшении продуктов и услуг. Это может привести к разочарованию клиентов и потере лояльности.



Идея проекта:



Создать инструмент, который сможет обрабатывать информационные ресурсы и собирать отзывы.

Отзывы будут проходить анализ нейронной сетью, помимо тега 'хороший отзыв' или 'плохой отзыв',

Мы хотим научить нейронную сеть понимать отзыв и резюмировать его, выделяя самое главное (что не хватает пользователю).

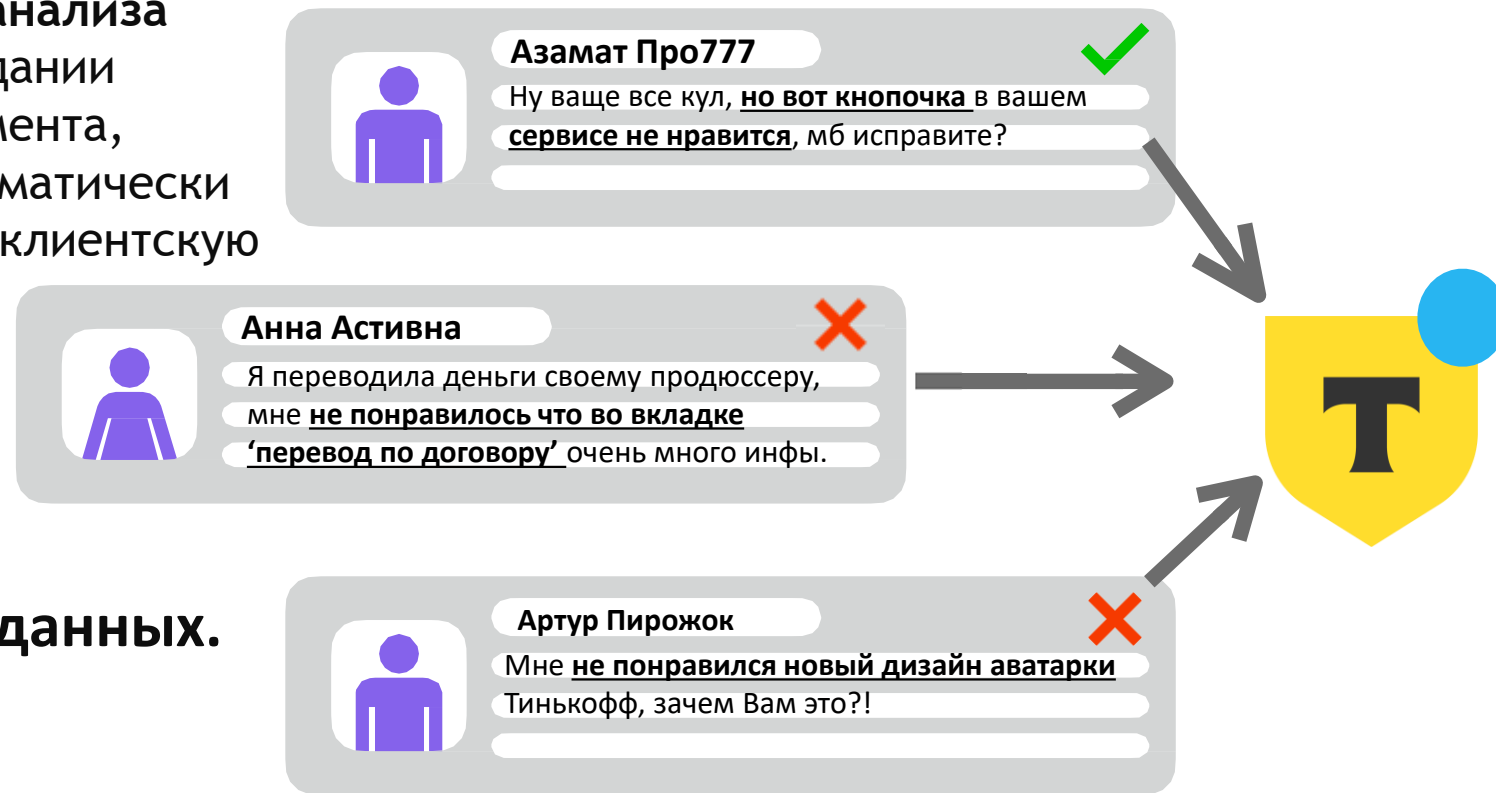


Цель проекта:

Главная цель проекта "Инструмент для анализа клиентских отзывов" заключается в создании эффективного и инновационного инструмента, который позволит банку Тинькофф систематически собирать, анализировать и использовать клиентскую обратную связь

Задачи проекта:

1. Разработка алгоритма сбора данных.
2. Чистка данных от лишнего.
3. Извлечение признаков.
4. Анализ и интерпретация отзывов



Анализ области:

- Где клиенты могут оставлять отзывы на продукты ‘Тинькофф’?
 1. Социальные сети и сайты
 2. Анкеты и опросы
 3. Рейтинг на картах, в магазинах приложений (Apple Store, Google Play, RuStore и так далее)
- Как можно их собирать?
 1. Автоматизированные системы парсинга.
 2. За счёт размещения опросов на интернет-ресурсах

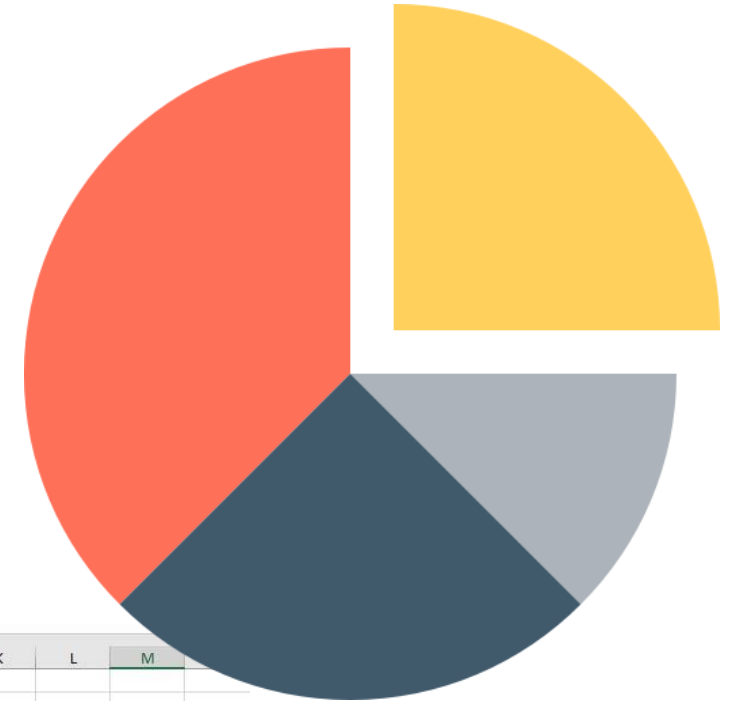


Анализ области:

- Как можно их обрабатывать?

Анализ сообщений локальной нейронной сетью:

- (1) С целью выявления положительных и отрицательных отзывов.
- (2) С целью выявления основной проблемы отзыва.



(1)



(2)



	A	B	C	D	E	F	G	H	I	J	K	L	M
1													
2													
3													
4													
5													
6													
7													
8													
9													
10													
11													
12													

*скриншоты принадлежат команде Spectrum, сделаны во время этапов работы с проектом от АО «Тинькофф Банк»

Анализ области:



Сравнение решений разных компаний, сопоставление с нашей идеей:

Компания:	Решение:	
 Ростелеком	Операторы отвечают на звонки вручную. Ввели речевую аналитику для улучшения клиентского опыта.	✗
 Steam	Не хотят видеть и принимать критику на платформе. Имеют очень хитрую систему обхода отрицательных отзывов.	✗
 Riot Games	Решают проблему отрицательных отзывов путём “хейта” других платформ (Steam, Epic games). Тем самым, переманивают игроков на свою платформу.	✗
 MeDsci <small>сеть клиник</small>	Анализируют отзывы клиентов, но делают это вручную. Не стремятся к созданию автоматизированных инструментов	✗
 Наше решение	С помощью систем парсинга собираем отзывы с интернет-ресурсов, используем ИИ для анализа семантики отзыва, берем главное и решаем проблему!	✓

Первоначальный вариант решения:



1. Использование локальной, готовой, обученной на большой базе данных модели.
2. Вариативная система парсинга (сбор данных с разных интернет-ресурсов).
3. Написанный сценарий(код), умеющий отсылать запросы и интегрировать с ИИ-моделью.

Плюсы:

- Использование локальной нейронной сети позволит нам разворачивать ее прямо в коде Python, отсутствует необходимость использовать ВПН, клиенты браузеров для выгрузки ответов.
- Полностью автоматизированный инструмент, не требующий внедрения человека.



Какие модели лучше использовать?



Протестировав GPT 3.5, мы пришли к выводу, что использовать данную модель нецелесообразно (хоть и имели api_key для использования), так как:

Использование таких сервисов стоит денег, есть вероятность, что мы можем быть заблокированы за частые запросы. Немало важную деталь играет Использование ВПН для работы с сервисом.

Прибегнув к бесплатным моделям, которые мы можем использовать, выделили **Mistral 7B Instruct v0.2**, которую и использовали в дальнейшем. **Zephyr beta 7b** - нейросеть, которая очень быстро и круто работает с отзывами, пробовали на личном примере

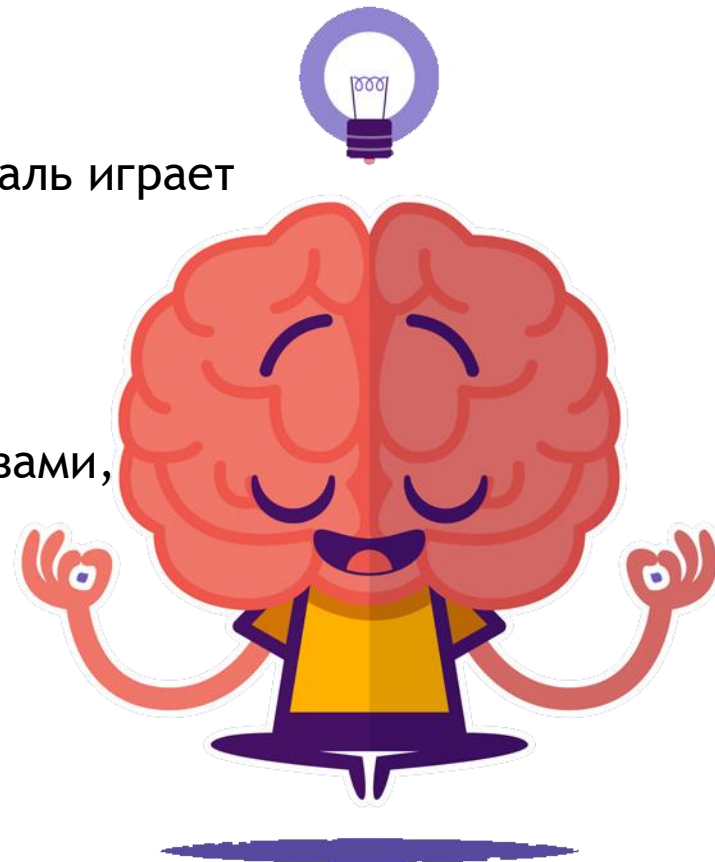
Предполагаемый вариант для нашего проекта:

Сайга-Мистраль

(третья русская нейросеть после YaGPT и GigaChat, публично доступная по API)

Интегрировать с ней хотим напрямую, в LMStudio не смогли найти бесплатную версию.

Слайд по пунктам №3 и №4



Наше решение на первый этап:



Идея, по нашему мнению, должна быть подкреплена практической частью. Мы готовы ее представить! Мы разработали инструмент, который работает на основе ИИ.

- 1 - Наш код (1) посылает запросы с **отзывами**, которые **получены путем парсинга**.
- 2 – Локальная нейронная сеть (2) обрабатывает их и **вычленяет главное**

(1) Код на языке Python

```
usage
def analyze_problems(review):
    messages = [
        {"role": "system", "content": "Вы помощник в определении проблемы, с которой столкнул"}
    ]

    try:
        completion = openai.ChatCompletion.create(
            model="local-model", messages=messages, temperature=0,

        )

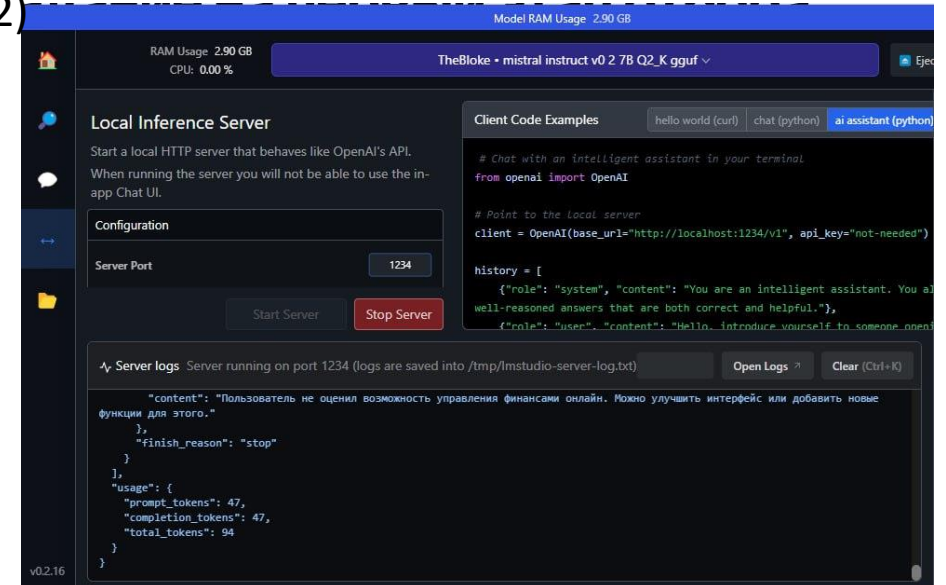
        response = completion.choices[0].message.content.strip()

        sheet.cell(row=sheet.max_row + 1, column=1, value=response)

        wb.save('output.xlsx')
    except Exception as e:
        print(f"Ошибка: {e}")
        return np.nan

df['problems'] = df['review'].apply(analyze_problems)
```

Наш изначальный выбор: Mistral 7B Instruct v0.2 (2)



Планы на будущее: Переход на сайгу-мистраль и взаимодействие с ней напрямую.

Планы на будущее: Определение наполненности отзывов.
Чистка ненужного.

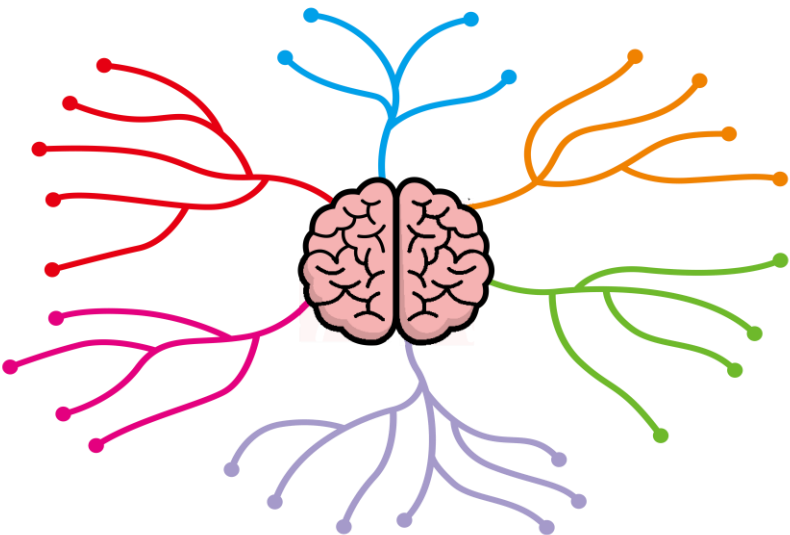


В рамках 1-ого этапа нам удалось:

Обработанные данные формировать в таблицу, которая автоматически обновляется (3) на нашем сайте, который мы хотим разработать для удобной и синхронной работы сотрудников компании.

*Сотрудники Тинькофф смогут удобно работать с табличными данными и заниматься решением задач, исходя из проблем пользователя

Будущее проекта:
Работа над логикой сайта и дизайном



Добро пожаловать на Tinkoff-Sirius III!

[Зарегистрироваться](#) [Войти](#) [Таблица](#)

(3) Это содержимое вашей главной страницы.
© 2024 Tinkoff page

Excel Table

Unnamed: 0
Пользователь не оценил дизайн приложения. Можно исправить это обновлением дизайна.
Пользователь ничего не поправил в приложении.
Пользователь не оценил точность определения нового стандарта банковских услуг (тинькофф). Можно улучшить описание или предоставить более конкретные примеры.
Низкие процентные ставки поражают пользователя. (Можно предложить увеличить процентные ставки.)
Проблема с технической поддержкой и замедленные ответы.
Пользователь не определил проблемы с приложением. Вывод: Отзыв положитель.
Кредитная система работает неудовлетворительно.
Пользователь не оценил простоту интерфейса. Можно улучшить его функциональность.
Отсутствие физических разделов в приложении.
Пользователь не оценил возможность управления финансами онлайн. Можно улучшить интерфейс или добавить новые функции для этого.

[Вернуться на главную](#)

Работа со вторым этапом:

План реализация решения:

-1 шаг(Поиск источников, с которых мы будем брать информацию):

- Выбрали такие источники, как banki.ru и sravni.ru.

Описание:

Мы хотели рассмотреть парсинг данных с соц.сетей, на основании статистики выявили, что там не так много полезных для нас отзывов, эту статистику можно найти в интернете.

Как мы парсили?

- Для первого источника (banki.ru) использовали bs4 и urllib
- Для второго источника (sravni.ru) использовали selenium и webdriver_manager, взаимодействовали через ChromeDriver

Подробнее о парсерах:

Для первого источника (banki.ru) использовали bs4 и urllib
Обычный парсер через get запросы по ссылке.

Была сложность с response.

Извлекает данные:

Имя пользователя, дата написания,
Рейтинг-оценку отзыва, сам отзыв.

Пример:

'name': 'Дмитрий', 'date': '22 марта',
'rating': 5, 'review_text': 'Впарили
кредитку, попользовался — денег
банку не принёс, да вот я гад такой.'

```
import requests
from bs4 import BeautifulSoup
import json
import unicodedata

def clean_html(html_content):
    cleaned_content = ''.join(char for char in html_content if unicodedata.category(char)[0] != 'C')
    return cleaned_content

def extract_review_body(url):
    try:
        response = requests.get(url)
        if response.status_code == 200:
            encodings = ['utf-8', 'ISO-8859-1', 'windows-1252'] #Хз почему-то ругался на энкодинги, сразу ставим много
            for encoding in encodings:
                try:
                    html_content = response.content.decode(encoding).replace('\x00', '')
                    cleaned_content = clean_html(html_content)
                    soup = BeautifulSoup(cleaned_content, 'html.parser')
                    script_tags = soup.find_all('script', type='application/ld+json')
                    for script_tag in script_tags:
                        script_content = script_tag.contents[0].strip()
                        script_data = json.loads(script_content)
                        if script_data.get('@type', '') == 'Review':
                            review_info = {
                                'review_name': script_data.get('name', ''),
                                'author_name': script_data.get('author', {}).get('name', ''),
                                'rating_value': script_data.get('reviewRating', {}).get('ratingValue', ''),
                                'review_body': script_data.get('author', {}).get('reviewBody', '')
                            }
                            return review_info
                    return "Review body not found."
                except UnicodeDecodeError as e:
                    print(f"Failed to decode using {encoding}: {e}")
                    continue
            print("Failed to decode the HTML content using any of the tried encodings.")
            return None
        else:
            print(f"Failed to fetch the webpage at {url}. Status code: {response.status_code}")
            return None
    except Exception as e:
        print(f"An error occurred while fetching the HTML content: {e}")
        return None

#тестик
url = "https://www.banki.ru/services/responses/bank/response/11397996/"

review_body = extract_review_body(url)
print("Review Body:")
print(review_body)
```

-2 шаг (Чистим от лишнего и структурируем собранные данные для последующего анализа)

- Парсер banki.ru потребовал дополнительную чистку от HTML-символов для чистки.
- Парсер sravni.ru устроен так, что после парсинга не требуется дополнительная чистка.

```
def text_cleaner(self, text):
    if pd.isna(text):
        return ''

    cleaned_text = re.sub(r'<[^>]+>|&lt;|&gt;', '', text)
    cleaned_text = cleaned_text.replace('&lt;p&gt;', '')
    cleaned_text = cleaned_text.lower()
    cleaned_text = self.remove_stopwords(cleaned_text)
    cleaned_text = self.remove_emojis(cleaned_text)
    cleaned_text = cleaned_text.replace('&quot;', '').replace('&amp;', '').replace('&lt;', '').replace('&gt;', '')
    return cleaned_text
```

Merging and Cleaning

```
[20]: merged_raw = pd.concat([df_reviews, df_reviews_selenium], ignore_index=True)
merged_raw
```

	review_name	author_name	rating_value	review_body	source	date
0	Благодарность за качество обслуживания	user-895516823510	5	<p>Хочу выразить благодарность сотрудник...	banki.ru	NaN
1	Обманули ? Да и ладно, мы же банк	user-358217358629	1	<p>Как у вас стало все хуже и хуже, обма...	banki.ru	NaN
2	Оспаривание операции	Didi1989i	5	<p>Заказала через приложение «Магнит» до...	banki.ru	NaN
3	Закрытие счета	user-253616412975	5	<p>Хочу поблагодарить всю команду Тинько...	banki.ru	NaN
4	Сервис на тарифе Премиум	Дмитрий	5	Отличный сервис на тарифе Премиум.\nСегодня 28...	sravni	28 марта
5	Тиньков перепутал меня с покойником и заблокир...	МС	5	Я давно клиент Тинькофф банка, добросовестно по...	sravni	26 марта
6	Не слушают клиентов от слова «совсем»	Елена	5	Было бы 0 звезд поставила бы 0! Ответительный...	sravni	26 марта
7	Быстро, четко и не выходя из дома)	Максим	5	Для начала дали кредитную карту 57, мне понрав...	sravni	26 марта
8	Хорошие сотрудники и банк	Цыден	5	Остались только хорошие впечатления от посещен...	sravni	25 марта
9	Профессиональное обслуживание	Пользователь	5	Профессиональное обслуживание. Карту доставил ...	sravni	24 марта
10	Отзыв персональному менеджеру Тинькоффбанка	Marina	5	Выражаю свою благодарность и признательность п...	sravni	22 марта
11	Нарушение прав человека — невозможно закрыть к...	Дмитрий	5	Впарили кредитку, попользовался — денег банку ...	sravni	22 марта
12	Сейчас тинькофф прайвек — Лучшее предложение н...	Илья	5	Пользуюсь сервисом Private уже около года. До ...	sravni	20 марта
13	Благодарность	Владимир	5	Пользуюсь сервисом Private. Обслуживание удобн...	sravni	20 марта

-3 шаг(Извлечение признаков)

1. Провели предварительную обработку текстов: удаление стоп-слов, лемматизацию / стемминг, удаление пунктуации.

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

```
[21]: df_reviews
```

[21]:	review_name	author_name	rating_value	review_body	source	review_body_cleaned	review_body_lemmatized	review_body_stemmed
0	Отличный менеджер по продажам	user-823317540379	5	<p>Я "теплый клиент", но Макси...	banki.ru	ря теплый клиент, Максим - единственный менедж...	ря теплый клиент , максим - единственны...	пя тепл клиент, макс - единствен менеджер, кот...
0	Общение с поддержкой	user-477917532850	5	<p>Обратилась в поддержку 25.03.2024 в 0...	banki.ru	рОбратилась поддержку 25.03.2024 09.30 общалас...	робращаться поддержка 25.03 . 2024 09.30...	побрат поддержк 25.03.2024 09.30 обща александ...

```
[22]: import pandas as pd  
import re  
import nltk
```

-3 шаг(Извлечение признаков)

1. Преобразовали собранные тексты отзывов в векторное представление с использованием методов NLP, таких как TF-IDF.

Этот выбор обычно хорош для задач кластеризации текста, но может быть полезно также рассмотреть использование других методов, таких как Word2Vec или Doc2Vec, особенно если у будут очень большие объемы данных и мы захотим учитывать семантическое сходство между словами или предложениями.

```
! usage
def analyze_reviews(file_path):
    df = load_and_preprocess_data(file_path)
    num_reviews = len(df)
    vectorizer = TfidfVectorizer()
    X = vectorizer.fit_transform(df['processed_review'])
    kmeans = KMeans(n_clusters=10, n_init=10)
    df['cluster'] = kmeans.fit_predict(X)

    # Генерация облака слов для каждого кластера
    for i in range(10):
        cluster_text = " ".join(df[df['cluster'] == i]['processed_review'].values)
        generate_word_cloud(cluster_text, i)

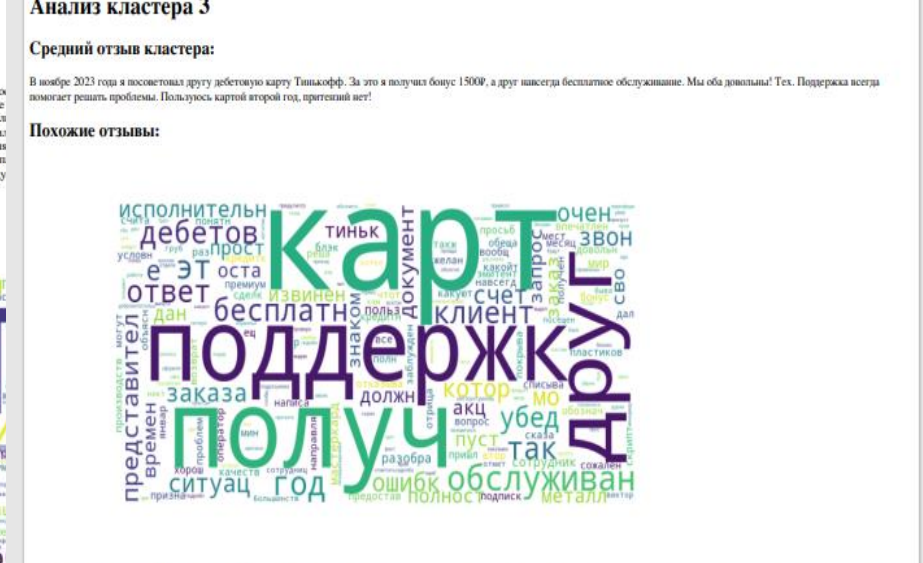
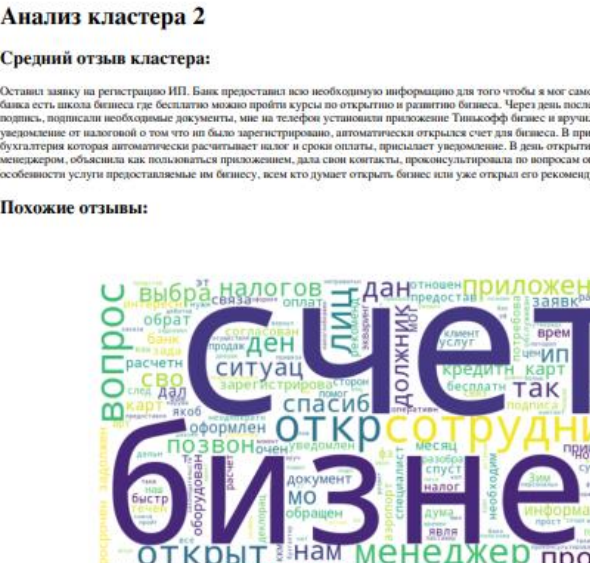
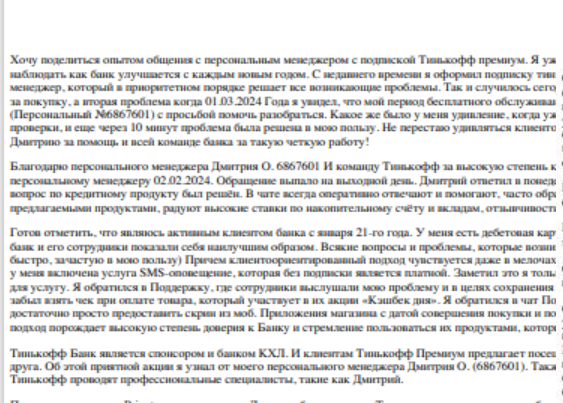
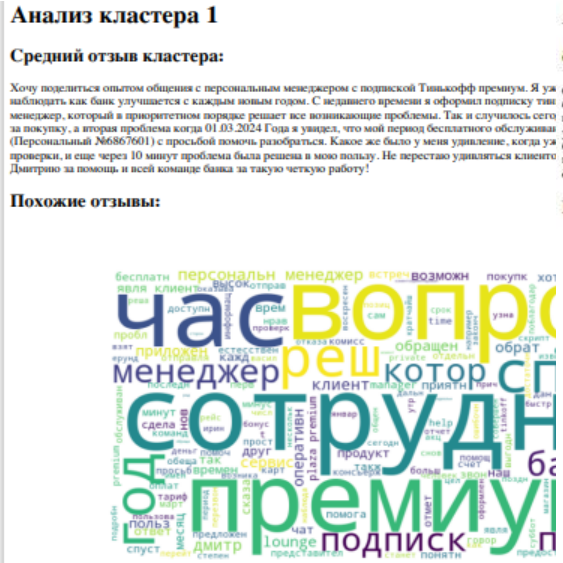
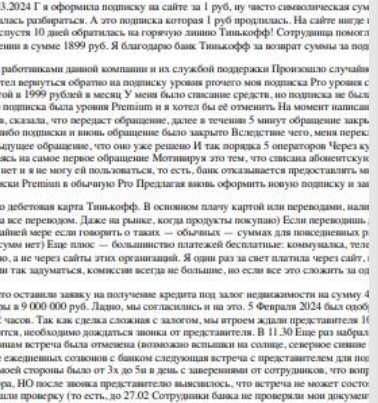
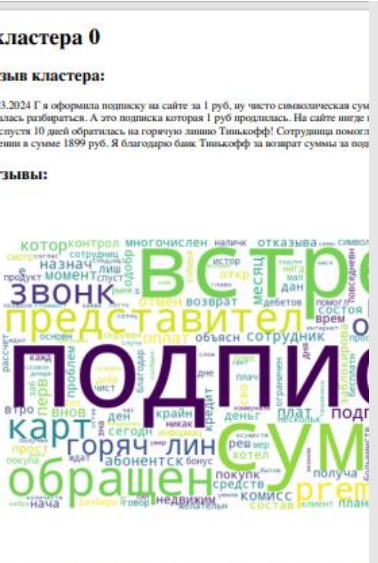
    # Поиск и добавление по 5 наиболее схожих отзывов в каждый кластер
    for i in range(10):
        try:
            cluster_center_idx = cosine_similarity(np.asarray(X[df['cluster'] == i].toarray()), np.asarray(X[df['cluster'] == i].mean(axis=0)))
            cluster_center_review = df[df['cluster'] == i].iloc[cluster_center_idx]['review_text']
            similar_reviews_indices = cosine_similarity(np.asarray(X[df['cluster'] == i].toarray()), np.asarray(X[df['cluster'] == i].mean(axis=0)))
            similar_reviews = df[df['cluster'] == i].iloc[similar_reviews_indices[1:]]['review_text'].values
            create_html_page(i, cluster_center_review, similar_reviews, num_reviews)
        except Exception as e:
            print(f"An error occurred while processing cluster {i}: {e}")
```

Использование KMeans для кластеризации векторов TF-IDF может дать хорошие результаты

-4 шаг (Анализ и интерпретация отзывов)

Слайд по пунктам №5 и №6

Использовали полученные признаки для выявления общих тем и тенденций в собранных отзывах. Сделали векторизацию на 10 кластеров. Анализа настроений нет - там надо применять что-то посерьезней , хотя бы модель типа bert, появляются сложности с русским языком. Такой анализ в HTML-формате генерирует наша программа(только не 4, а 10 кластеров):



Вывод:

Наше решение было успешно загружено на GitHub, вы можете опробовать его и посмотреть.

Будем ждать встречи 1 апреля в 13:00!