**Аналитическая записка**

**Цель**: провести предобработку и анализ комментариев для классификации их по признакам.

**Методология**:

1.Формирование структуры данных

2.Предобработка данных (удаление лишних символов, пробелов, приведение к единому регистру, токенизация, удаление стоп-слов, лемматизация)

3.Кластеризация

4.Классификация

5.Анализ результатов и выводы

**Формирование структуры набора данных**

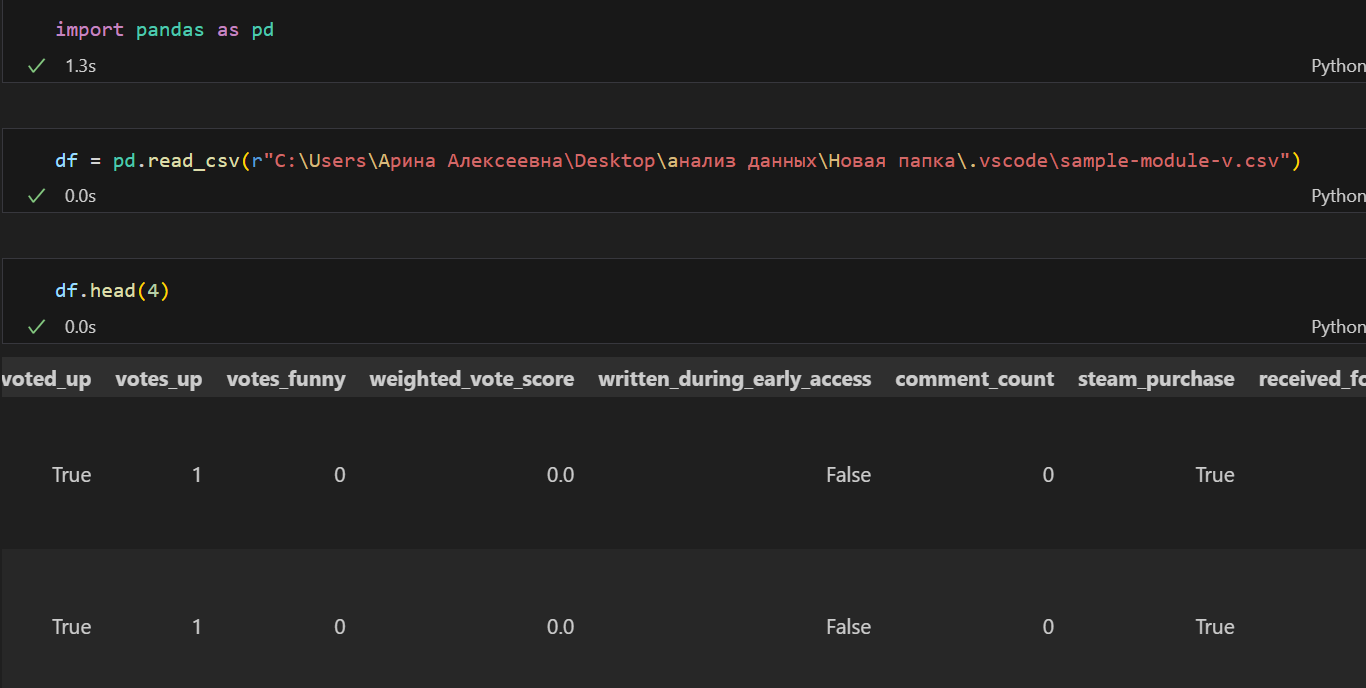
****

Рисунок 1-Загрузка и просмотр данных

Произвожу загрузку необходимых пакетов, парсинг csv-файла с помощью библиотеки пандас, изучение первых 4-х строк таблицы (Рис.1).

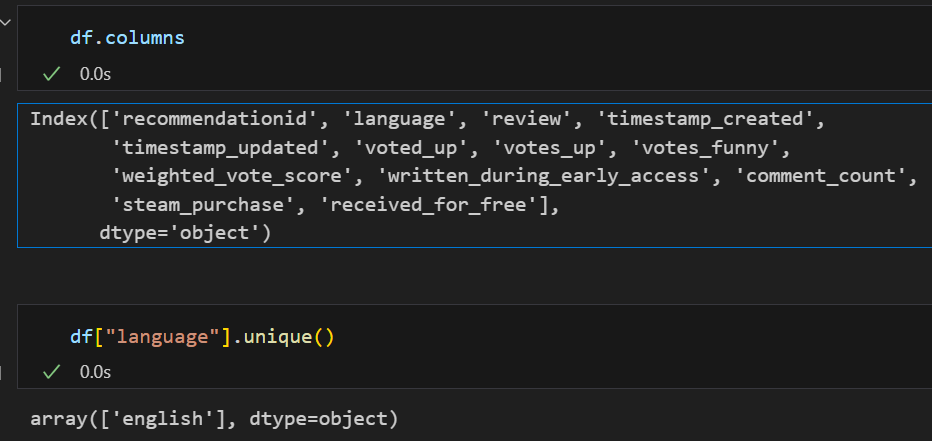


Рисунок 2-Анализ имеющихся данных

Для наглядности вывожу список столбцов, содержащихся в датафрейме, чтобы сформировать нужную структуру данных для достижения цели анализа. Просматриваю количество уникальных значений столбца language, чтобы понять нужна ли фильтрация по нему. Видно, что комментарии только на английском языке, значит фильтрация не нужна (Рис 2).



Рисунок 3-Формирование структуры данных

Формирую датафрейм с нужными мне столбцами (Рис.3).

Описание атрибутов:

Recommendationid -идентификатор комментария (нужен для идентификации)

Review – комментарий (ключевой атрибут для анализа)

voted\_up – атрибут для определения позитивности комментария (может косвенно помочь с классификацией полезности, так как зачастую положительные комментарии полезны)

votes\_up – атрибут для определения полезности

votes\_funny – атрибут для определения забавности

Остальные атрибуты отброшены, так как для данной задачи классификации на «полезный», «бесполезный», «забавный» не влияют.

**Предобработка данных**

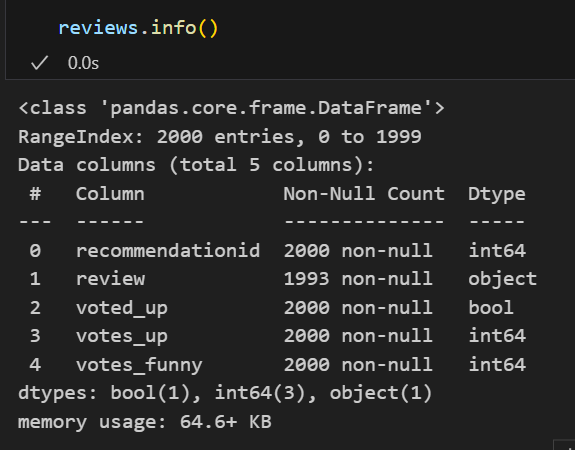
****

Рисунок 4-Просмотр общей информации о данных

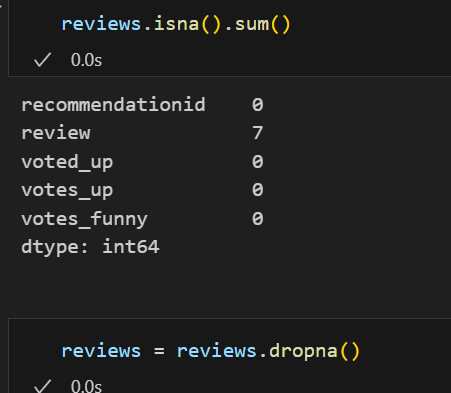


Рисунок 5-Проверка пропусков

Проверяю есть ли пропуски в данных, ведь они влияют на качество дальнейшего анализа. Выявлено 7, но так как в таблице 2000 строк (Рис.4), то их удаление никак не повлияет на результаты анализа (Рис.5).

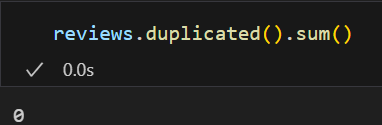


Рисунок 6-Проверка на дубликаты

Проверяю на наличие дубликатов, их нет, значит можно преступить к работе с текстовыми данными.

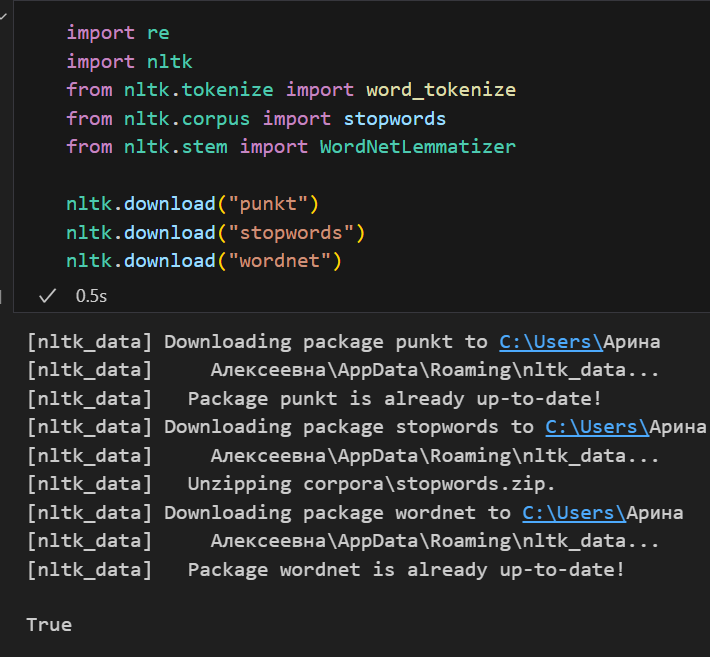


Рисунок 7-Загрузка пакетов для предобработки

Загружая необходимые для предобработки пакеты и проверяю все ли успешно прошло(Рис.7).

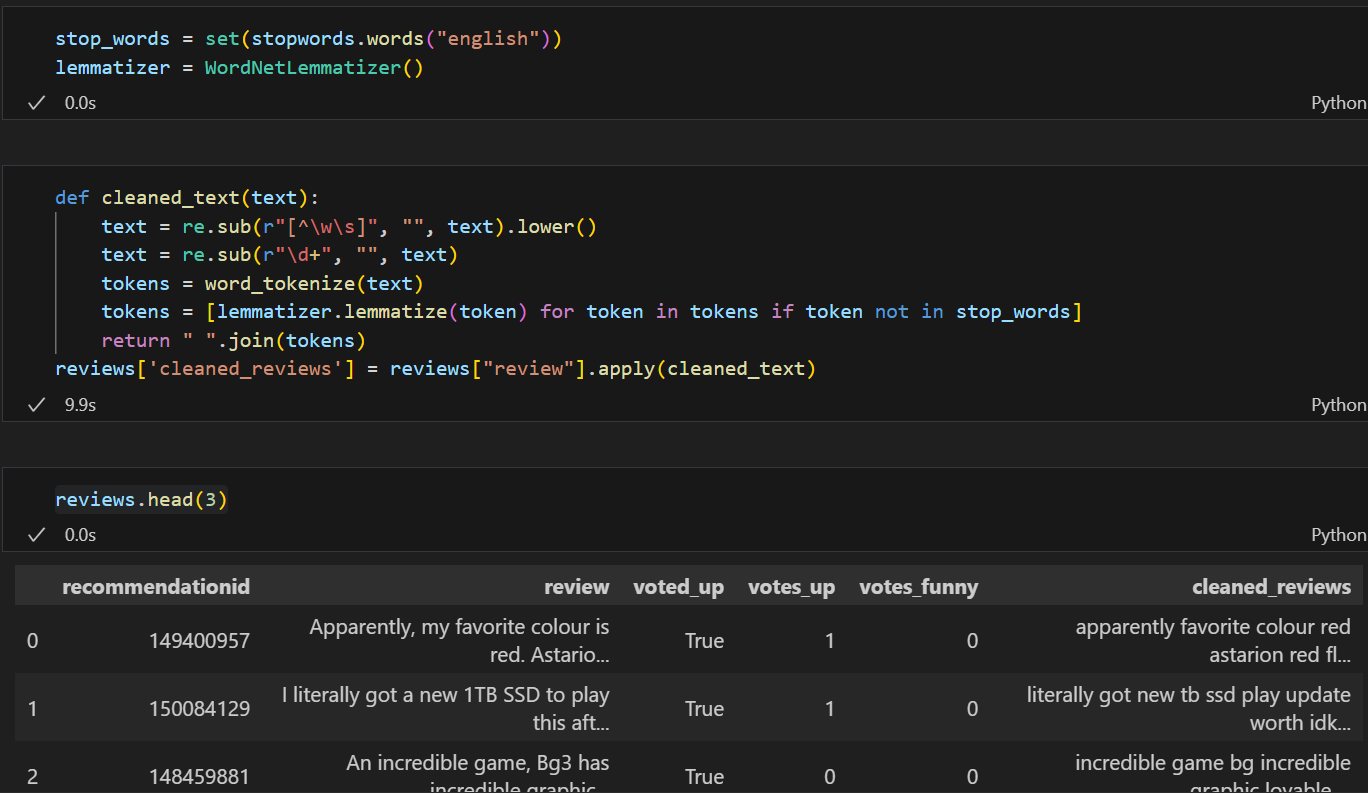


Рисунок 8-Инициализация лемматизатора и стоп-слов

С помощью пакетов для предобработки, провожу очищение данных (см. рис.7). А именно:

Сначала инициализирую лемматизатор (приведение к начальной словоформе) и набор стоп-слов (например предлоги\союзы) (Рис.8);

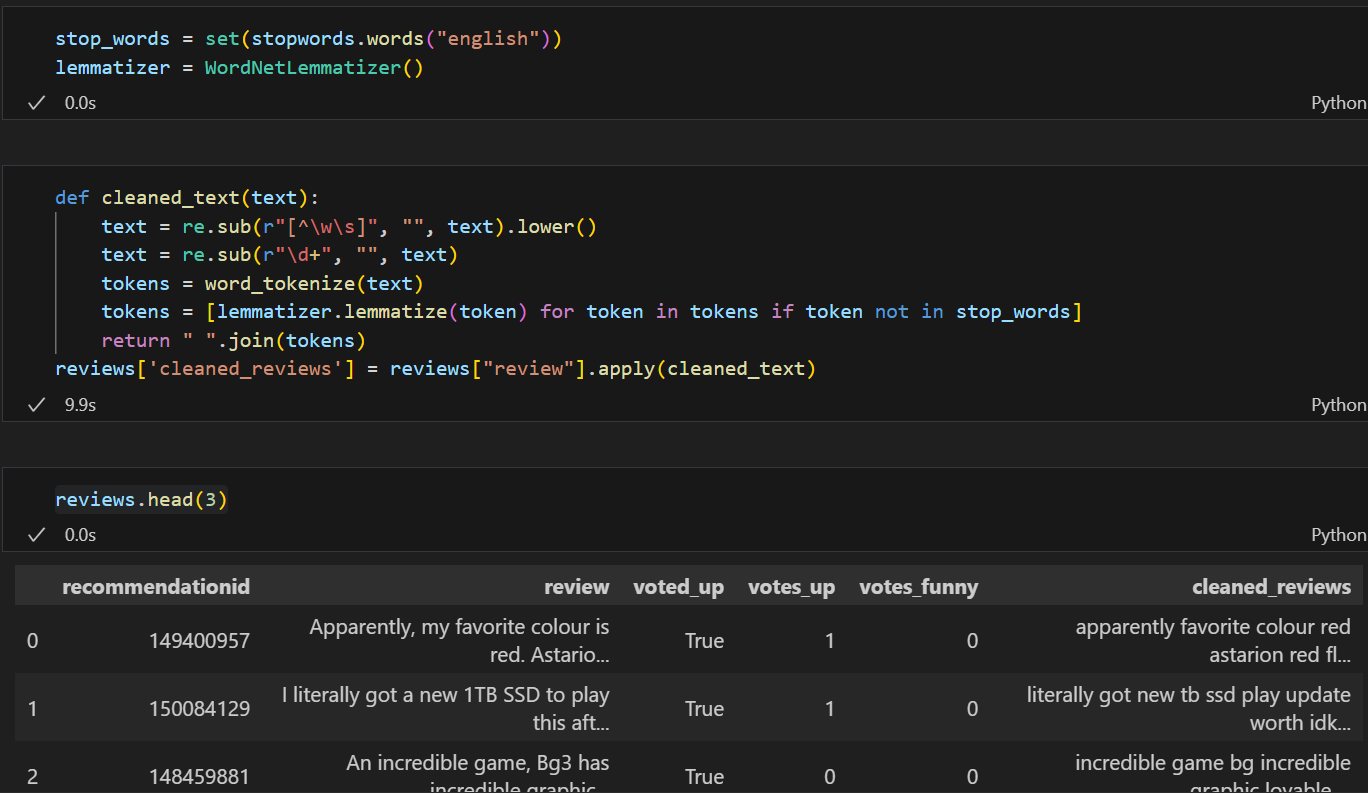


Рисунок 9-Функция предобработки данных

Создаю функцию предобработки, в которой с помощью использования регулярных выражений удаляю лишние символы (знаки препинания и цифры), привожу данные к нижнему регистру, разбиваю их на токены, удаляю от стоп-слов и лемматизирую (Рис.9).

Функция возвращает очищенные данные, приведенные к нормальной словоформе.

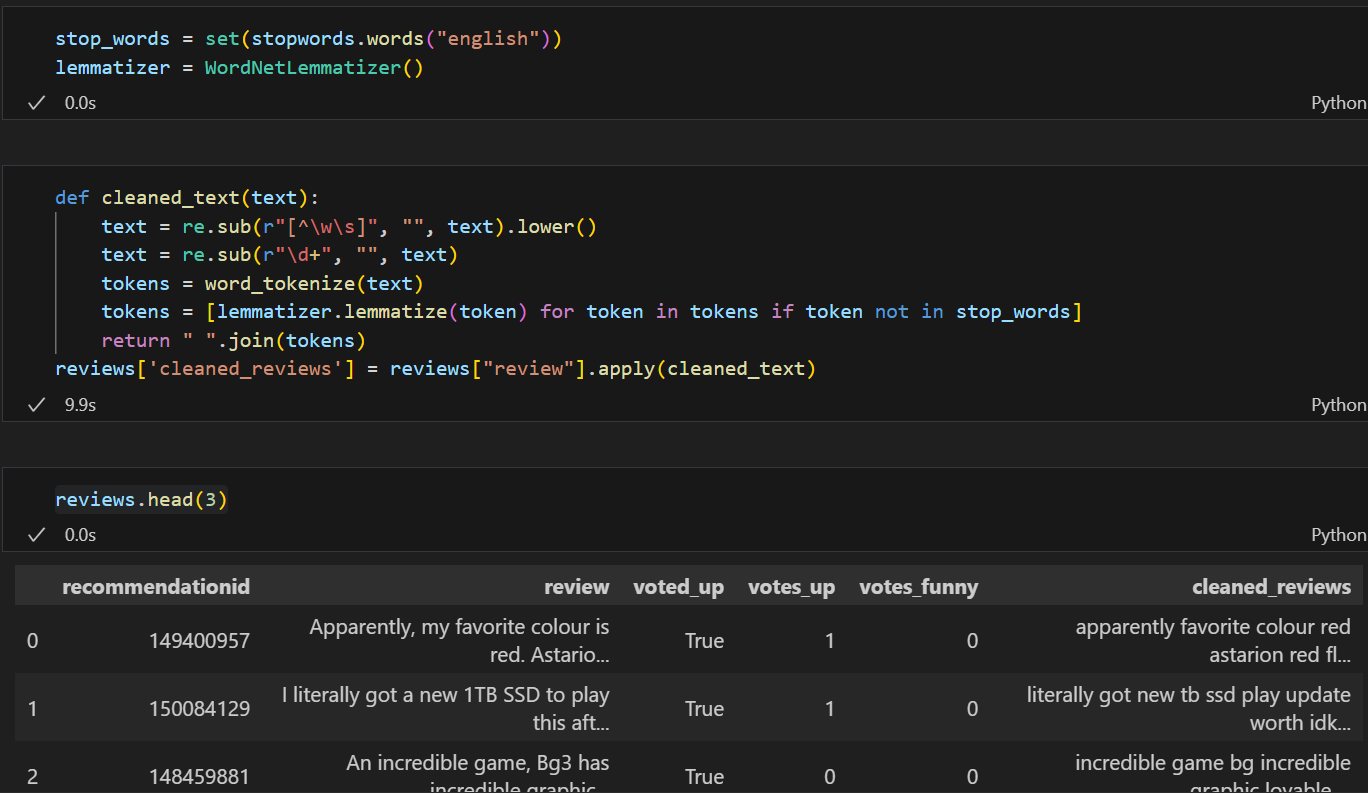


Рисунок 10-Применение предобработки

Этот этап (предобработки ) важен, так как во время него значительно сокращается объем, качество данных, что повлияет на более качественный результат в дальнейшем и сократит время на выполнение кластеризации и классификации.

Добавляю в датафрейм новый столбец с постобработанными данными, с помощью метода apply применяю функцию к данным(Рис.10).

Проверяю, добавился ли столбец(Рис 11).

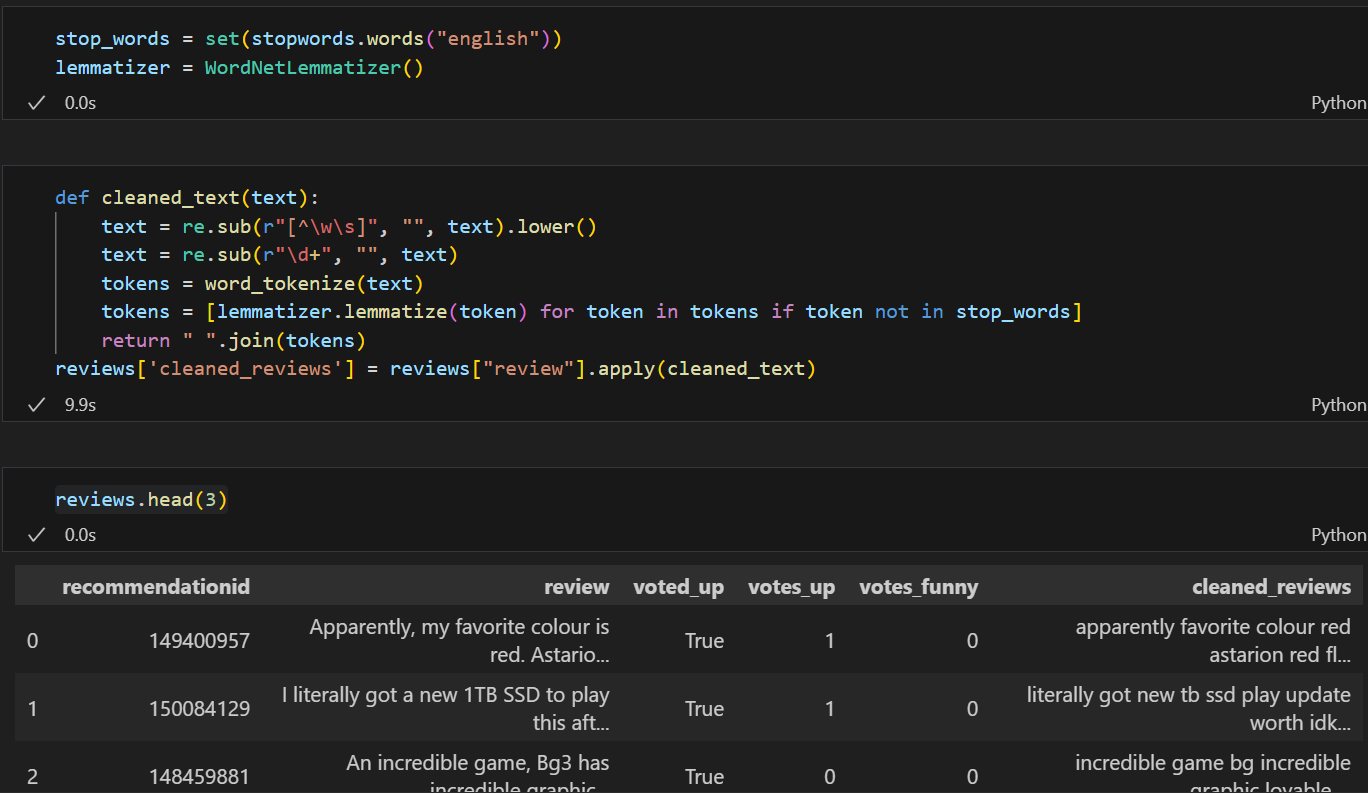


Рисунок 11-Проверка

Также нужно проверить постобработанный столбец на наличие шумов или пустых значений. Вручную просмотрела данные, видно, что есть значения равные одному слову, одной букве или пустые вообще.

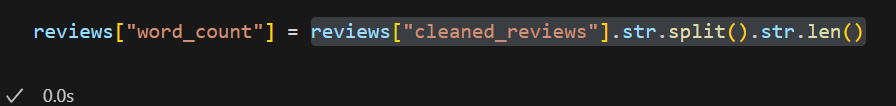


Рисунок 12-Добавление столбца с количеством слов

Оцениваю их количество, добавив столбец с количеством слов в очищенном отзыве (Рис.12).



Рисунок 13-Строки с одним и менее слов

Смотрю сколько тех, что состоят из одного слова и менее -их 294 (Рис.13).

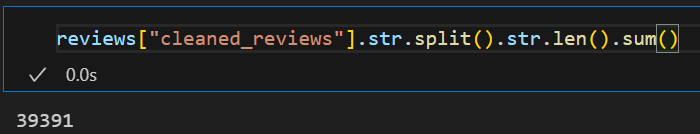


Рисунок 14-Количество слов во всех текстах

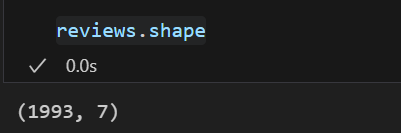
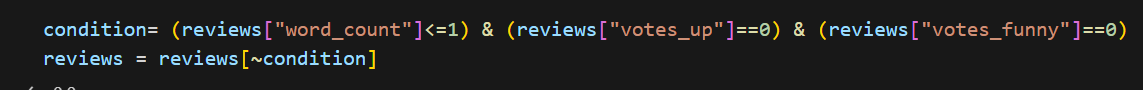


Рисунок 15-Размер датафрейма

Чтобы понять, можно ли их удалять без вреда для дальнейшего анализа, смотрю количество слов и строк во всех очищенных текстах (Рис.14-15).



Можно удалить те строки, где одновременно комментарий не веселый, бесполезный и содержит 1<= слов, так как они будут шумом и бесполезны при анализе. Прописываю условие и с помощью булевой маски удаляю строки. (Рис.16)

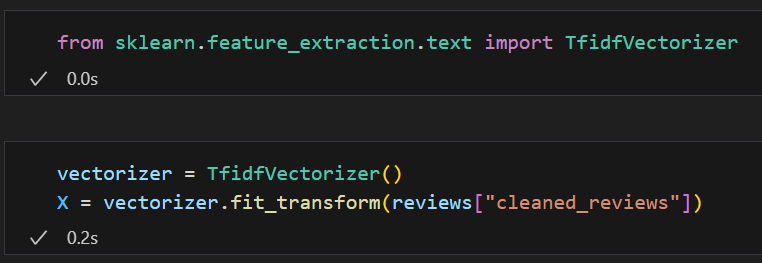


Рисунок 16-Векторизация текста

Для того, чтобы преступить к обучению модели кластеризации, нужно провести векторизацию, так как машина не поймет текст, она работает с числами (Рис.16).

Импортирую необходимые пакеты библиотеки scikit-learn. Выбираю алгоритм векторизации данных – TF-IDF. Этот метод определяют частоту слов в конкретном тексте и их редкость в общем наборе текстов, чтобы выявить важность. Выбрала этот метод, так как он подходит для небольших объемов данных (кол-во слов) и структуры (размер отзывов).

В результате получится матрица, где значения строк-комментарии, значения столбцов-вес слова

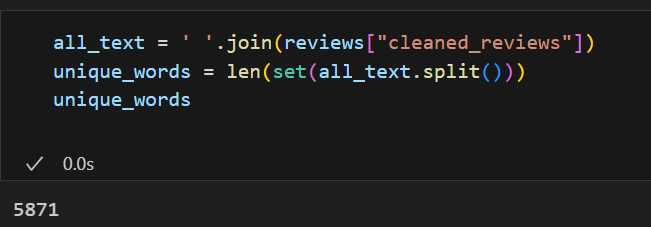


Рисунок 17-Количество уникальных слов во всем тексте

Создаю объект векторизатора (см.Рис.16), где в параметрах указываю количество уникальных слов = 3000, чтобы уменьшить объем данных.

Указала 3000, так как общее количество слов по столбцу очищенных отзывов = 39175, а строк 1767. Количество уникальных (Рис.17). Оптимальное количество будет 50-80% от уникальных.

**Кластеризация**

После векторизации можно приступать к кластеризации данных.

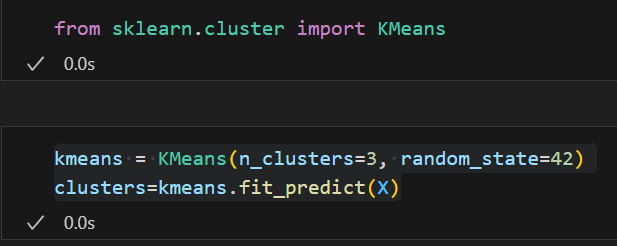


Рисунок 18-Кластеризация

Выбрала модель кластеризации без учителя K-means, так как предобработаны данные, векторизованы с помощью TF-IDF подхода и можно взять фиксированное количество кластеров = 3 (полезный, бесполезный, забавный) по заданию. Импортирую необходимые пакеты для кластеризации, формирую объект модели и обучаю модель на векторизованных раннее данных (см.Рис.16) с помощью библиотеки scikit-learn (Рис.18). Добавляю столбец кластеров в датафрейм

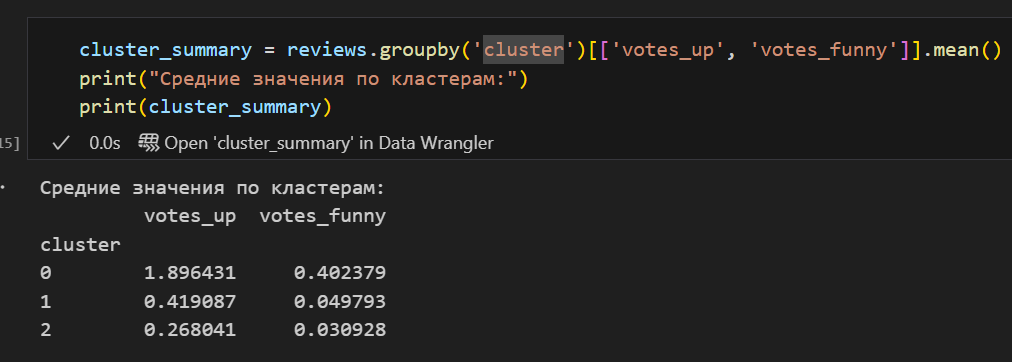


Рисунок 19-Статистические характеристики кластеров

Для распределения кластеров на метки полезный\бесполезный\забавный вывожу статистические характеристики кластеров в соответствии со столбцами votes\_funny и votes up (Рис.19).

Вывод: 0-высокий votes\_up, значит полезный; 1-высокий votes\_funny, значит забавный;2-низкие метки обе, значит бесполезный

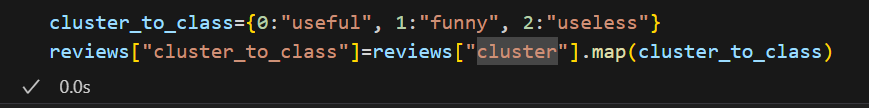


Рисунок 20-Добавление столбца

Добавляю столбец с метками кластеров в соответствии с числовыми (Рис.20).

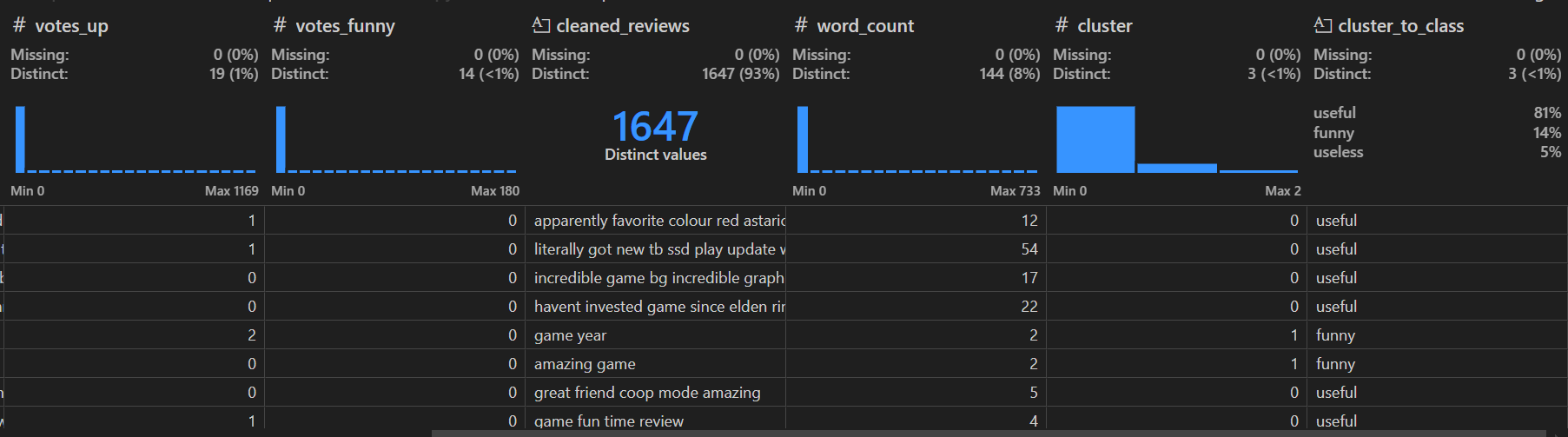


Рисунок -Проверка таблицы после кластеризации

Проверяю успешно ли всё прошло (Рис.21).

**Классификация**

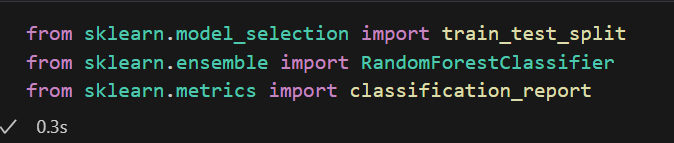
****

Рисунок -Импорт пакетов для классификации

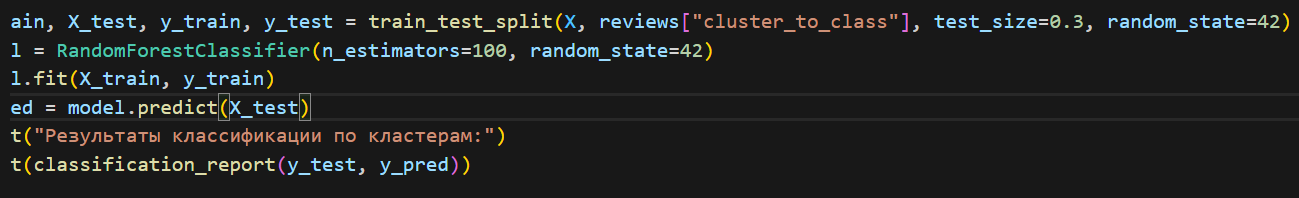


Рисунок -Классификация

После кластеризации классифицирую отзывы по следующим классам: полезный, бесполезный, забавный. Настраиваю модель на классификацию текста по перечисленным выше классам.

Выбрала модель классификации Random Forest с учителем, так как хорошо работает с разреженными данными (которые получаются в результате TF-IDF -векторизации), устойчив к шуму, так как после кластеризации метки могут быть не идеальными и содержать смешанные отзывы, а Random Forest снижает влияние ошибок.

Разделяю данные на тестовую и обучающую выборки, чтобы не переобучать модель, а тренировать на одной части, а проверять способности на другой.

Обучаю модель с помощью метода fit, применяю метод predict, который использует обученную модель для предсказания классов на тестовой выборке. Вывожу метрики классификации (Рис.23).



Рисунок 24-Результаты классификации

В результате сравниваются истинные метки с предсказанными. Видно, что precision (доля правильных предсказаний для классов) близок к 1 или равен 1 в случае useless, значит предсказания funny, useful, useless верны. Recall (доля найденных примеров классов) для funny найдено 84% useful 99% useless 81 % реальных. F1-score – баланс между долей правильных предсказаний и найденных примеров классов, показывает насколько хорошо модель предсказывает класс. Наилучшим образом модель предсказала useful, так как показатель = 98%. Параметр Support – количество примеров в текстах (funny 69; useful 436; useless 26). Accuracy 96% предсказаний верны.

**Анализ результатов и выводы**

Анализ результатов:

Точность модели составляет 96%

Результаты распределения составляют 81% полезные; 14% забавные; 5 % бесполезные.

Вывод:

Большинство комментариев полезные, что говорит о высоком качестве обратной связи.