

## Анализ ключевых слов в Google Colab

*Пайдершев Алексей Константинович*

*Приамурский государственный университет имени Шолом-Алейхема*

*Студент*

### Аннотация

Цель исследования — выявить и оценить ключевые слова в текстовых данных с использованием инструментов и методов анализа. Исследование проводилось в облачной среде Google Colab с применением языка Python, что обеспечило удобство и гибкость при выполнении экспериментов. Исследование может применяться для анализа научных статей или документов, позволяя быстро выявлять ключевые термины и тематики. Оно также полезно для автоматизации отчетов, например, в новостных агрегаторах или аналитических платформах.

**Ключевые слова:** Google Colab, Анализ ключевых слов, scikit-learn, gensim.

## Keyword Analysis in Google Colab

*Paidershev Aleksei Konstantinovich*

*Sholom-Aleichem Priamurskiy State University*

*Student*

### Abstract

The purpose of the study is to identify and evaluate keywords in text data using analysis tools and methods. The research was conducted in the Google Colab cloud environment using Python, which provided convenience and flexibility when performing experiments. The research can be used to analyze scientific articles or documents, allowing you to quickly identify key terms and topics. It is also useful for automating reports, for example, in news aggregators or analytical platforms.

**Key words:** Google Colab, Keyword analysis, scikit-learn, gensim.

## 1 Введение

### 1.1 Актуальность

Актуальность исследования «Анализ ключевых слов в Google Colab» обусловлена растущим объемом текстовых данных в цифровую эпоху и необходимостью их эффективной обработки. Во многих областях, таких как маркетинг, наука, образование и информационные технологии, возникает потребность в автоматическом выявлении ключевых слов для анализа текстов, классификации документов, улучшения поисковых систем и оптимизации контента.

## 1.2 Обзор исследований

А. В. Дробчик в данной статье предпринял попытку определения тематики медиатекстов, размещенных на сайтах ведущих университетов, методом автоматического извлечения ключевых слов с использованием библиотеки Sklearn в среде Google Colab [1], о том, что современные средства компьютерной обработки текста позволили автоматизировать многие рутинные задачи лексикографов и терминологов написала М. И. Ладушина [2], Д. Д. Пронин в статье обосновывает и описывает применение статистического метода оценки эмоциональной тональности текста литературного произведения как инновационного актуального инструментария компьютерной лингвистики [3], анализ естественного языка с помощью методов машинного обучения и компьютерного зрения провел Э. Б. Гармаев [4].

## 1.3 Цель исследования

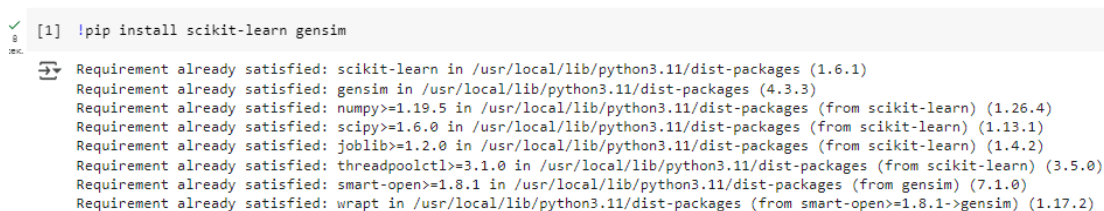
Цель исследования — выявить и оценить ключевые слова в текстовых данных с использованием инструментов и методов анализа.

## 2 Материалы и методы

В данном исследовании используется среда Google Colab и язык программирования Python

## 3 Результаты

Устанавливаем две библиотеки для анализа текста (рис.1): scikit-learn для работы с алгоритмами машинного обучения и gensim для тематического моделирования и обработки естественного языка (NLP).



```
[1] !pip install scikit-learn gensim

Requirement already satisfied: scikit-learn in /usr/local/lib/python3.11/dist-packages (1.6.1)
Requirement already satisfied: gensim in /usr/local/lib/python3.11/dist-packages (4.3.3)
Requirement already satisfied: numpy>=1.19.5 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (1.26.4)
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (1.13.1)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (3.5.0)
Requirement already satisfied: smart-open>=1.8.1 in /usr/local/lib/python3.11/dist-packages (from gensim) (7.1.0)
Requirement already satisfied: wrapt in /usr/local/lib/python3.11/dist-packages (from smart-open>=1.8.1->gensim) (1.17.2)
```

Рисунок 1. Установка библиотек

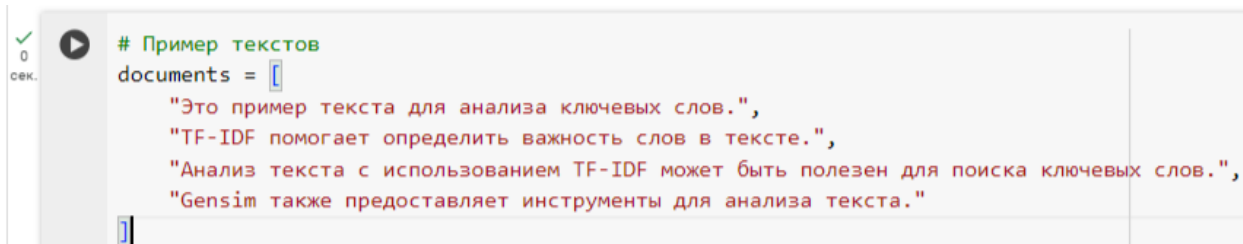
Импортируем класс TfidfVectorizer из библиотеки scikit-learn (рис.2), который используется для преобразования текста в числовую матрицу TF-IDF. Это позволяет оценить важность слов в документах относительно всего корпуса текстов.



```
[2] from sklearn.feature_extraction.text import TfidfVectorizer
```

Рисунок 2. Импорт класса

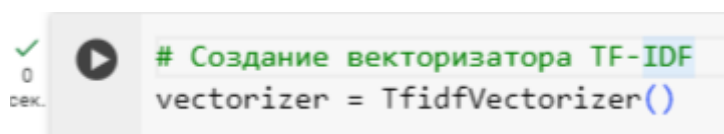
Создадим список `documents`, содержащий примеры текстов, которые будут использоваться для анализа ключевых слов с помощью методов, таких как TF-IDF (рис.3).



```
# Пример текстов
documents = [
    "Это пример текста для анализа ключевых слов.",
    "TF-IDF помогает определить важность слов в тексте.",
    "Анализ текста с использованием TF-IDF может быть полезен для поиска ключевых слов.",
    "Gensim также предоставляет инструменты для анализа текста."
```

Рисунок 3. Создание списка

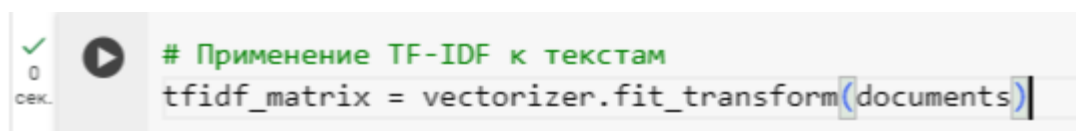
Создадим объект `TfidfVectorizer`, который будет использоваться для преобразования текстов в числовую матрицу TF-IDF, оценивающую важность слов в документах (рис.4).



```
# Создание векторизатора TF-IDF
vectorizer = TfidfVectorizer()
```

Рисунок 4. Создание объекта

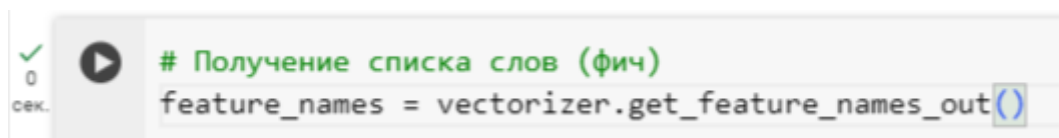
Применим TF-IDF к текстам, преобразуя их в числовую матрицу `tfidf_matrix`, где каждая строка соответствует документу, а каждый столбец — уникальному слову, с значениями, отражающими важность слов в документах (рис.5).



```
# Применение TF-IDF к текстам
tfidf_matrix = vectorizer.fit_transform(documents)
```

Рисунок 5. Применение TF-IDF к текстам

Получим список уникальных слов (фич) из текстов, которые были использованы для создания TF-IDF матрицы, и сохраняем их в переменной `feature_names` (рис.6).



```
# Получение списка слов (фич)
feature_names = vectorizer.get_feature_names_out()
```

Рисунок 6. Получим список уникальных слов

Импортируем библиотеку `pandas`, которая будет использоваться для удобного преобразования и анализа данных, например, для работы с TF-IDF матрицей в виде таблицы (рис.7).



Рисунок 7. Преобразование матрицы TF-IDF в массив и вывод результатов

Преобразуем TF-IDF матрицу в массив `tfidf_array`, а затем создадим таблицу `tfidf_df` с помощью `pandas`, где строки — документы, столбцы — слова, а значения — веса TF-IDF (рис.8).



Рисунок 8. Преобразование матрицы в массив

Выведем таблицу `tfidf_df`, где показаны значения TF-IDF для каждого слова в каждом документе, позволяя визуальнo оценить важность слов (рис.9).

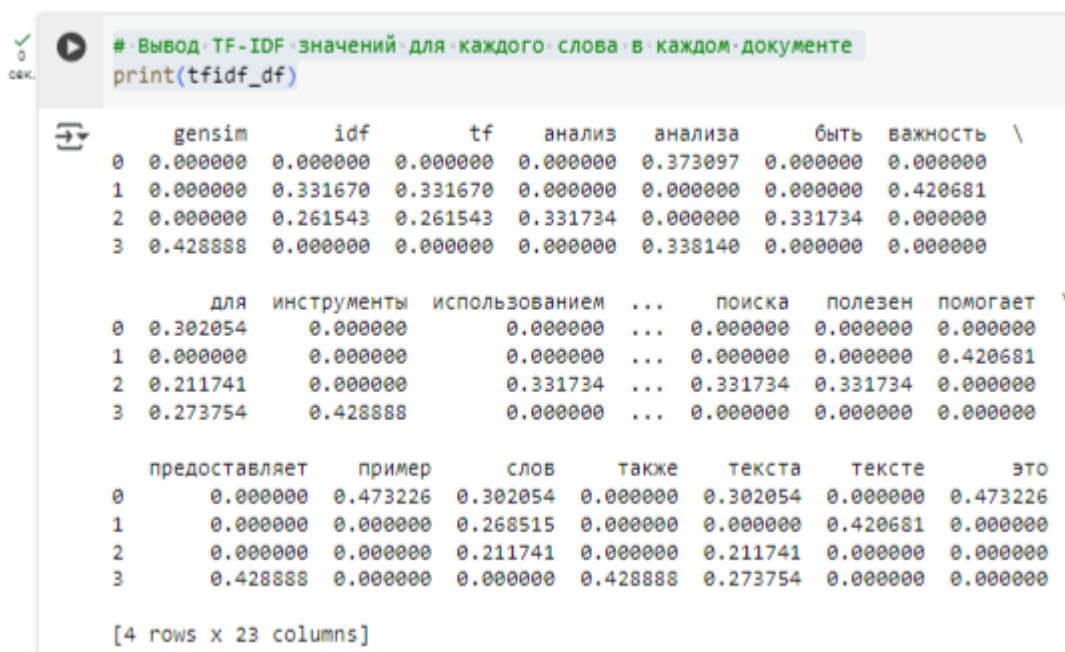


Рисунок 9. Таблица

#### 4 Вывод

В заключение можно сказать, что каждый документ имеет свои ключевые слова, которые отражают его основную тематику. Слова с высокими значениями TF-IDF, такие как «пример», «gensim», «анализ» и «TF-IDF», являются наиболее значимыми для соответствующих документов. Нулевые значения TF-IDF указывают на отсутствие слова в документе или его низкую важность. Этот анализ помогает понять, какие слова являются ключевыми для каждого документа, что может быть полезно для задач классификации, поиска или извлечения информации.

**Библиографический список**

1. Дробчик А. В. Определение тематической доминанты в медиатекстах методом автоматического извлечения ключевых слов (на материале веб-сайтов российских вузов) // Медиатизация практик современного общества: проблемы и перспективы. Владимир: Редакционная коллегия, 2023. С. 237.
2. Ладушина М. И. Как язык Python помогает лексикографам // Journal of applied linguistics and lexicography. 2022. Т. 4. №. 2. С. 107-121.
3. Пронина Е.В., Пронин Д.Д. Метод анализа эмоциональной тональности текста как цифровой инструмент изучения текста произведения литературы (на примере романа М.Ю. Лермонтова «Герой нашего времени») // Современный ученый. №6, 2022. С. 8.
4. Гармаев Э.Б. Библиотеки python для работы с текстом естественного языка на русском // Актуальные проблемы современной когнитивной науки. Сборник статей Всероссийской научно-практической конференции, Магнитогорск, 2019. С. 19.