**ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ ГОРОДА МОСКВЫ**

**ГБОУ Школа № 1553 имени В.И. Вернадского**

**Техническая документация**

**Профиль «Информационные технологии»**

Классификатор по тематикам интернет ресурсов, в части реализации загрузки содержимого страниц из сети интернет и пользовательского интерфейса, использующего выбранные и обученные модели классификации.

Листов

|  |  |  |
| --- | --- | --- |
|  |  | **Команда**  Карпов Александр, 9 класс,  ГБОУ Школа №1553 им. В.И. Вернадского  Разумный Юрий, 9 класс,  ГБОУ Школа №1553 им. В.И. Вернадского |
|  |  |  |
|  |  | **Руководитель** |
|  |  | Тренихина Татьяна Аркадьевна  учитель информатики и ИКТ |
|  |  | ГБОУ Школа №1553 им. В.И. Вернадского |

Москва, 2023 г

**Содержание**

[Перечень определений, сокращений и обозначений](#_gjdgxs) **3**

[1 Введение](#_30j0zll) **4**

[1.1 Цели и задачи](#_1fob9te) 5

[2 Основная часть](#_3znysh7) **7**

[2.1 Анализ технических требований](#_2et92p0) 7

[2.1.1 Функциональные требования](#_tyjcwt) 7

[2.1.2 Нефункциональные требования](#_3dy6vkm) 7

[2.2 Проектирование и разработка](#_1t3h5sf) 8

[2.2.1 Обоснование выбора программно-технических средств, используемых при реализации](#_4d34og8) 8

[2.2.2 Структурная и функциональная схемы программного продукта](#_2s8eyo1) 10

[2.2.3 Блок-схема работы основного алгоритма](#_17dp8vu) 12

[2.3 Исходные коды](#_dsba6f5ai0e4) 15

[2.4 Описание испытаний](#_3rdcrjn) 16

[2.4.1 Испытание 1. Установка из файла](#_vc0r5o7p3xh9) 16

[2.4.2 Испытание 2. Выбор и ввод ссылки для дальнейшей классификации](#_swm6eyc13pkn) 18

[2.4.3 Испытание 3. Классификация, просмотр результатов](#_rhezh8vb3u9v) 21

[2.4.4 Испытание 4. Выгрузка результатов в таблицу](#_umddarynjf9y) 22

[2.4.5 Испытание 5. Обучение модели, метрики для 3-х моделей](#_m74art1vbccb) 23

[2.4.6 Испытание 6. Загрузка в программу файла с ссылками.](#_26in1rg) 26

[2.4.7 Испытание 7. Классификация введенного текста](#_nmgokz8weama) 30

[3 Список литературы](#_ygisbvc7omwf) 35

# **Перечень определений, сокращений и обозначений**

В тексте документа используются следующие определения:

| **Термин и определения** | **Определение, описание** |
| --- | --- |
| Лемматизация | Процесс приведения словоформы к лемме — её нормальной (словарной) форме |
| Векторизация | Процесс обработки естественного языка. В процессе используются языковые модели для сопоставления слов с пространством векторов |
| Precision | Точность |
| Recall | Полнота |
| F1-score | Среднее гармоническое precision и recall |
| Support | Количество выборок истинного ответа, лежащего в каждом классе целевых значений |
| Accuracy | Доля правильных ответов |
| Macro avg | Макроcредний показатель |
| Weighted avg | Взвешенное среднее |

В тексте документа используются следующие обозначения и сокращения:

| **Обозначение, сокращение** | **Определение** |
| --- | --- |
| БД | База данных |
| ИИ | Искусственный интеллект |
| ER | Entity-Relationship |
| TFIDF | Статистическая мера, используемая для оценки важности слова |

# **Введение**

В наше время быстро растет количество информации, получаемой человеком. Чтобы оптимизировать наше время и снизить нагрузку на мозг, людям проще классифицировать определенные данные, чтобы была возможность, например, не вчитываясь в новость, понять, с чем она связана.

В частности, при обращении к Интернет-ресурсам часто возникает необходимость сгруппировать просмотренные сайты по тематикам (классам), чтобы в дальнейшем быстро найти информации только определенной тематики. Возникает задача классификации Интернет-ресурсов по тематикам. Такие классификаторы ресурсов существуют, но у них много недостатков, например, низкая точность или же скудная функциональность.

В рамкаха Предпрофессиональной олимпиады по профилю “Информационные технологии” мы решили разработать программу для классификации Интернет-ресурсов по тематикам. В нашей работе реализуется классификация Интернет-ресурсы по следующим тематикам:

* Финансы (business);
* Развлечения (entertainment);
* Политика (politics);
* Медицина (medical);
* Графика (graphics);
* История (historical);
* Еда (food);
* Космос (space);
* Спорт (sport);
* Технологии (technologie).

В первой версии программы классификация будет осуществляться только для ресурсов на английском языке, так как при обучении использовался открытый набор данных с текстами сайтов на английском языке Dataset Text Document Classification (10 тематик) – (<https://www.kaggle.com/datasets/jensenbaxter/10dataset-text-document-classification/>). В дальнейшем программный продукт может быть адаптирован для ресурсов на русском языке, для этого нужно будет выполнить аналогичное обучение на новом наборе данных (в рамках выполнения работы такой открытый набор данных найти не удалось).

## **Цели и задачи**

Цель настоящей работы - реализовать классификацию ресурсов с применением машинного обучения и хранением обработанных и классифицированных данных в БД.

Для достижения поставленной цели необходимо решить следующие задачи:

* выбрать подходящий набор данных с текстами ресурсов по нескольким тематикам;
* обучить одну или несколько моделей машинного обучения на выбранном наборе данных;
* реализовать загрузку текстовой информации с Интернет-ресурса с заданным адресом;
* реализовать приложение, которое использует обученные модели для определения тематики текста, загруженного с Интернет-ресурса;
* реализовать сохранение классифицированных данных в БД.

# **Основная часть**

## **Анализ технических требований**

Разрабатываемый продукт должен удовлетворять приведенным ниже функциональным и нефункциональным требованиям.

### **Функциональные требования**

В программном продукте должны быть реализованы следующие функциональные возможности:

1. Внесение ссылки на сайт с целью его классификации.
2. Реализация базы данных для хранения классифицированных и обработанных данных интернет-ресурсов.
3. Возможность выгрузки данных сайта в формате CSV.
4. Возможность подгрузки файла с ссылками и данными для классификации и выгрузка файла с уже классифицированными данными оффлайн.
5. Возможность увидеть метрики при прогоне тестовой выборки, например: accuracy, precision, recall.
6. Наличие не менее 6 классов, по которым будет производиться классификация.

### **Нефункциональные требования**

Также к разрабатываемому продукту предъявляются следующие требования:

1. Требования к ОС: Windows 10, дистрибутив Linux, MacOS.
2. Интерфейс программы должен быть простым и интуитивным, не требующим дополнительного обучения.

## **Проектирование и разработка**

### **Обоснование выбора программно-технических средств, используемых при реализации**

В качестве сред разработки мы использовали PyCharm Community Edition (для разработки) и Google Collaboratory (для обучения моделей). PyCharm Community Edition – это бесплатная версия кроссплатформенной интегрированной среды разработки (IDE) для языка программирования Python под лицензией лицензией Apache License. Google Collaboratory – это бесплатная интерактивная облачная среда платформа для работы с блокнотами Jupyter на языке Python. Платформа предоставляет доступ к графическим процессорам GPU и TPU, увеличивающие возможности машинного обучения, позволяя сократить время обучения моделей.

В качестве языка программирования мы использовали язык программирования Python. Python – это высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нем программ. Также, на этом языке программирования написано большое количество библиотек, которые упрощают решение задач машинного обучения, анализа данных и работы с БД.

Для реализации графического пользовательского интерфейса мы использовали библиотеку PyQt. PyQt предоставляет интерфейс Python для Qt, одной из самых мощных и популярных кроссплатформенных библиотек графического интерфейса. PyQt был разработан RiverBank Computing Ltd.

В качестве базы данных для хранения данных нашего программного продукта мы выбрали SQLite. Это простая в использовании, бессерверная база данных, с удобной библиотекой для доступа с использованием Python.

NLTK – пакет библиотек и программ для символьной и статистической обработки естественного языка. С его помощью можно удалить стоп-слова, делить текст на предложения и проводить лемматизацию, а так же создать TF-IDF векторизатор. Так как использовавшийся для обучения моделей набор данных (датасет) содержал тексты на английском языке по десяти тематикам, то мы использовали лемматизатор для английского языка. Векторизатор TF-IDF – это метод представления текстов в виде векторов, которые учитывают меру оригинальности слова, вычисляя её путем сравнения количества появлений слова в тексте с количеством текстов, в которых оно появляется.

Для сохранения обученных моделей мы использовали модуль Python Pickle. Это – модуль, реализующий двоичные протоколы для сериализации и десериализации объектной структуры Python. При помощи него обученные модели были сохранены в файлы в Google Collaboratory и выгружены для последующего использования в приложении, с которым работает пользователь нашего программного продукта.

Для решения задачи машинного обучения мы использовали модели, входящие в sklearn. sklearn (scikit-learn) – это широко используемый пакет Python для машинного обучения. Он включает различные алгоритмы классификации, регрессии и кластеризации. В нашем программном продукте мы обучили и используем следующие модели классификаторов:

* наивный байесовский классификатор (GaussianNB);
* логистическая регрессия (LogisticRegression);
* метод опорных векторов (SVM).

### **Структурная и функциональная схемы программного продукта**

Ниже приведена структурная схема разработанного программного продукта (см. ниже Рисунок 1).

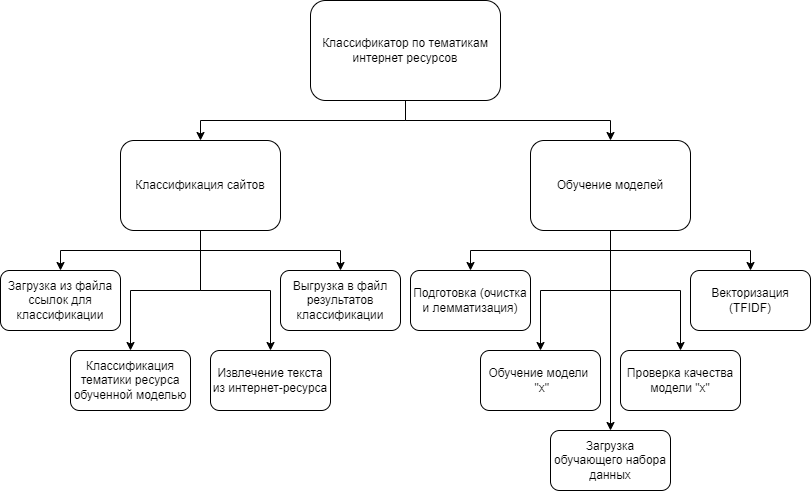


Рисунок 1. Структурная схема системы сбора данных

Ниже приведена функциональная схема программного продукта (см. ниже Рисунок 2).

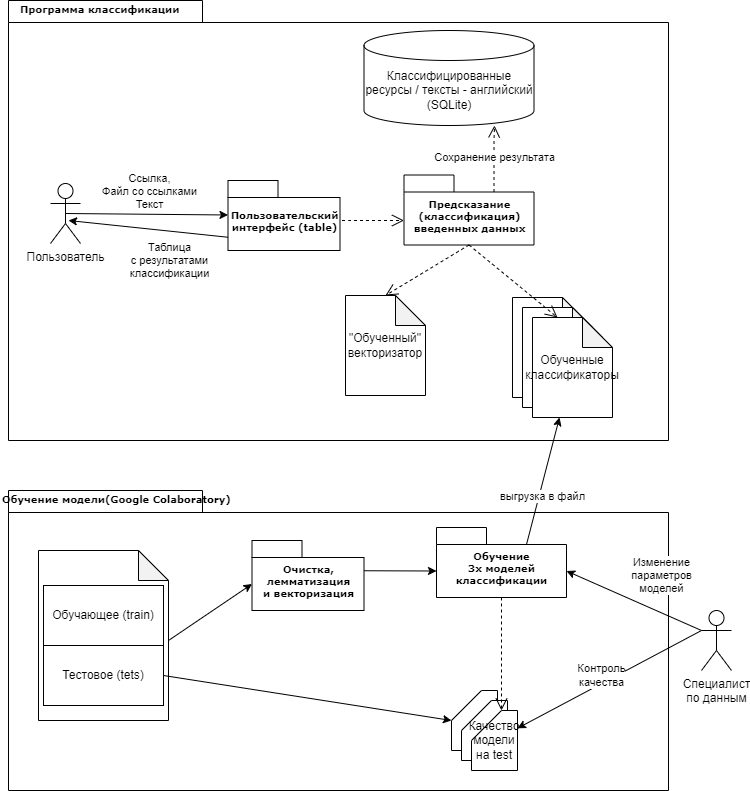


Рисунок 3. Функциональная схема

Данные хранятся в БД SQLite, ER-модель приведена ниже. Перечень классифицированных ресурсов и текстов хранится в таблице Set, а информация о названиях возможных тематики в таблице Categories.

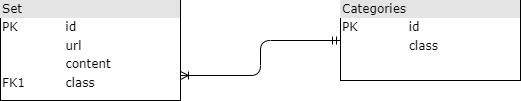


Рисунок 4. ER-модель

### **Блок-схема работы основного алгоритма**

Ниже приведено описание двух основных алгоритмов, реализованных в программном продукте:

* Обучения моделей классификации (см. ниже Рисунок 3).
* Классификация Интернет-ресурсов / текстов (см. ниже Рисунок 4).

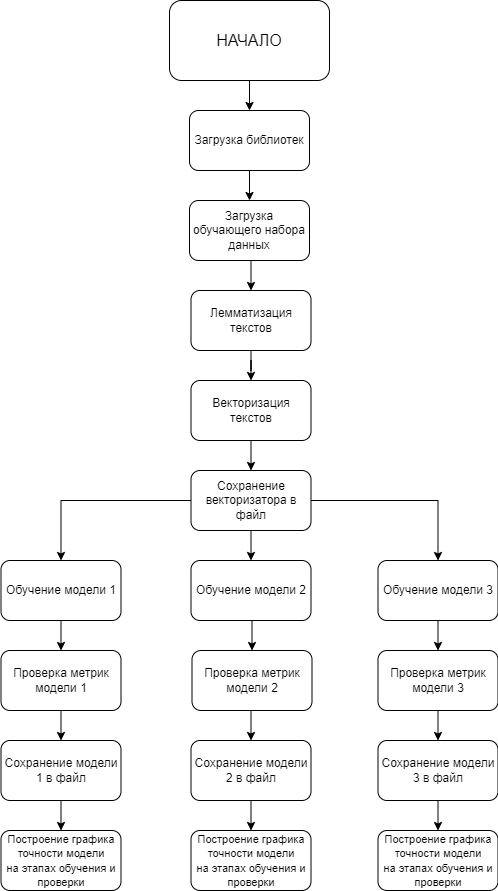


Рисунок 3. Блок-схема “Обучения моделей классификации”

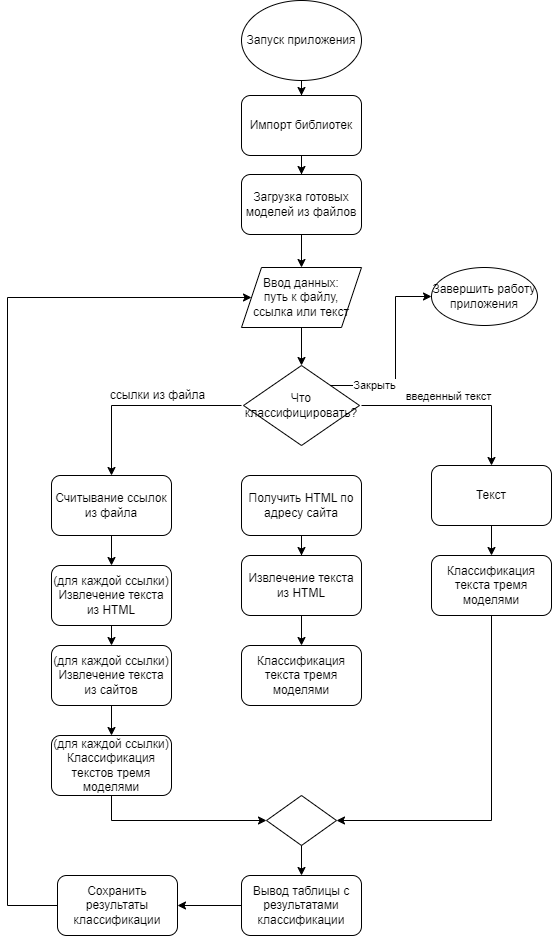


Рисунок 4. Блок-схема “Классификация Интернет-ресурсов / текстов”

## **Исходные коды**

<https://github.com/actggg/9-class-olimp/>

## **Описание испытаний**

### **Испытание 1. Установка из файла**

Предусловие: На комьютер скачаны файлы программы для классификации с <https://github.com/actggg/9-class-olimp>. На компьютере установлен Python 3.

|  | **Действие** | **Ожидаемый результат** | **Фактический результат** |
| --- | --- | --- | --- |
| 1. | Установить PyQt (pip install PyQt5) и проверить что она появилась в списке доступных (вызвать pip list) | Библиотека установлена. | PyQt5 успешно установился и отображается в списке установленных модулей. |
| 2. | Скачать файлы программы с <https://github.com/actggg/9-class-olimp> | На компьютере сохранились файлы \*.py (программа), \*.. в(база данных) и \*.pkl (модели). | В файловой системе доступны файлы программы: |

### **Испытание 2. Выбор и ввод ссылки для дальнейшей классификации**

Предусловие: На компьютере установлена программа для классификации (см. Испытание 1). Выбран сайт на английском по одной из допустимых тематик. Например, сайт на тему еда:

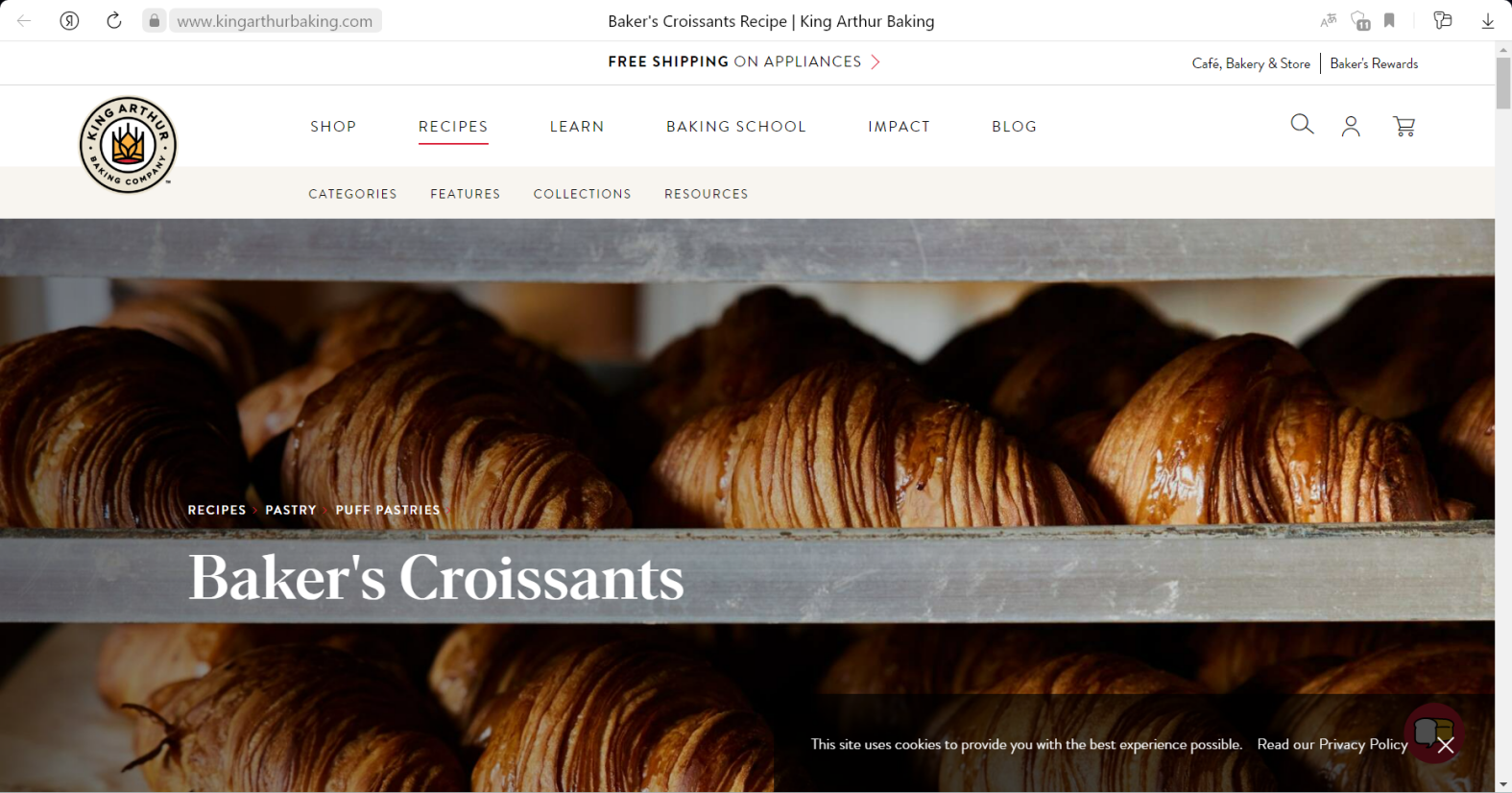


Рисунок 5.Сайт для классификации

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Действие** | **Ожидаемый результат** | **Фактический результат** |
| 1. | Выполнить файл “table.py” | Отображается главное окно программы | Отображается окно программы. |
| 2. | Ввести (или скопировать) ссылку в поле “Введите ссылку на сайт, который вы хотите классифицировать” | Введенная или скопированная ссылка отображается в поле ввода. |  |

### **Испытание 3. Классификация, просмотр результатов**

Предусловие: В запущенной ранее программе в поле “Введите ссылку на сайт, который вы хотите классифицировать” в введен адрес ссылки на интернет ресурс (см. Испытание 2).

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Действие** | **Ожидаемый результат** | **Фактический результат** |
| 1. | Нажать на находящуюся правее поля ввода кнопку “Классифировать” | Отображается таблица с результатами классификации. Если два и более метода вернули одну и ту же тематику, то в последнем столбце отображается название итогового класса. Иначе отображается прочерк. | Все методы определи тематику “Еда” (food). Итоговый класс также определяется как “Еда” (food). |

### **Испытание 4. Выгрузка результатов в таблицу**

Предусловие: Отображается форма с результатами классификации одной или нескольких ссылок (см. Испытание 3 и Испытание 6).

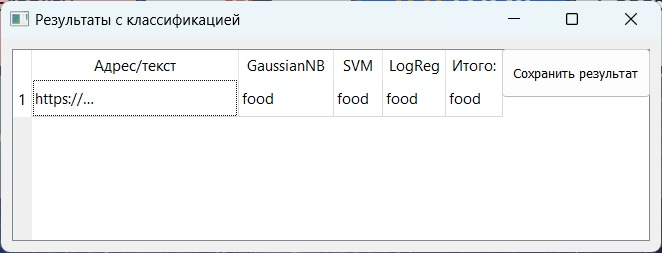


Рисунок 6. Результат классификации

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Действие** | **Ожидаемый результат** | **Фактический результат** |
| 1. | Нажать на кнопку “Сохранить результат”.  Открыть таблицу БД | В таблице БД отображается новая информация о классификации (тематика из поля “Итого”) |  |

### **Испытание 5. Обучение модели, метрики для 3-х моделей**

Предусловие: В Google Colaboratory открыт файл с блокнотом Jupyther, загруженным из <https://github.com/actggg/9-class-olimp>. Загружен файл с обучающим набором (database.scv).

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Действие** | **Ожидаемый результат** | **Фактический результат** |
| 1. | Пошагово выполнить ячейки блокнота: Загрузка набора данных из 1000 строк, на котором должны обучаться модели.  Подготовка данных для обучения моделей (очистка, разбиение на слова, векторизация, сохранение векторизатора в файл).  Разделение набора на обучающий и тестовый, обучение модели на обучающей части, оценка качества обучения на тестовой части, выведение метрики. | Отображается таблица с метриками качества на тестовом наборе данных для каждой модели. | Для наивного байесовского классификатора (GaussianNB):    Для метода опорных векторов (SVM):    Для логистической регрессии (LogReg): |

### **Испытание 6. Загрузка в программу файла с ссылками.**

Предусловие: На компьютере хранится файл с перечнем ссылок (links.tx1). Программа запущена, отображается стартовое окно программы (см. Испытание 2).

### 

Рисунок 6. Файл с ссылками

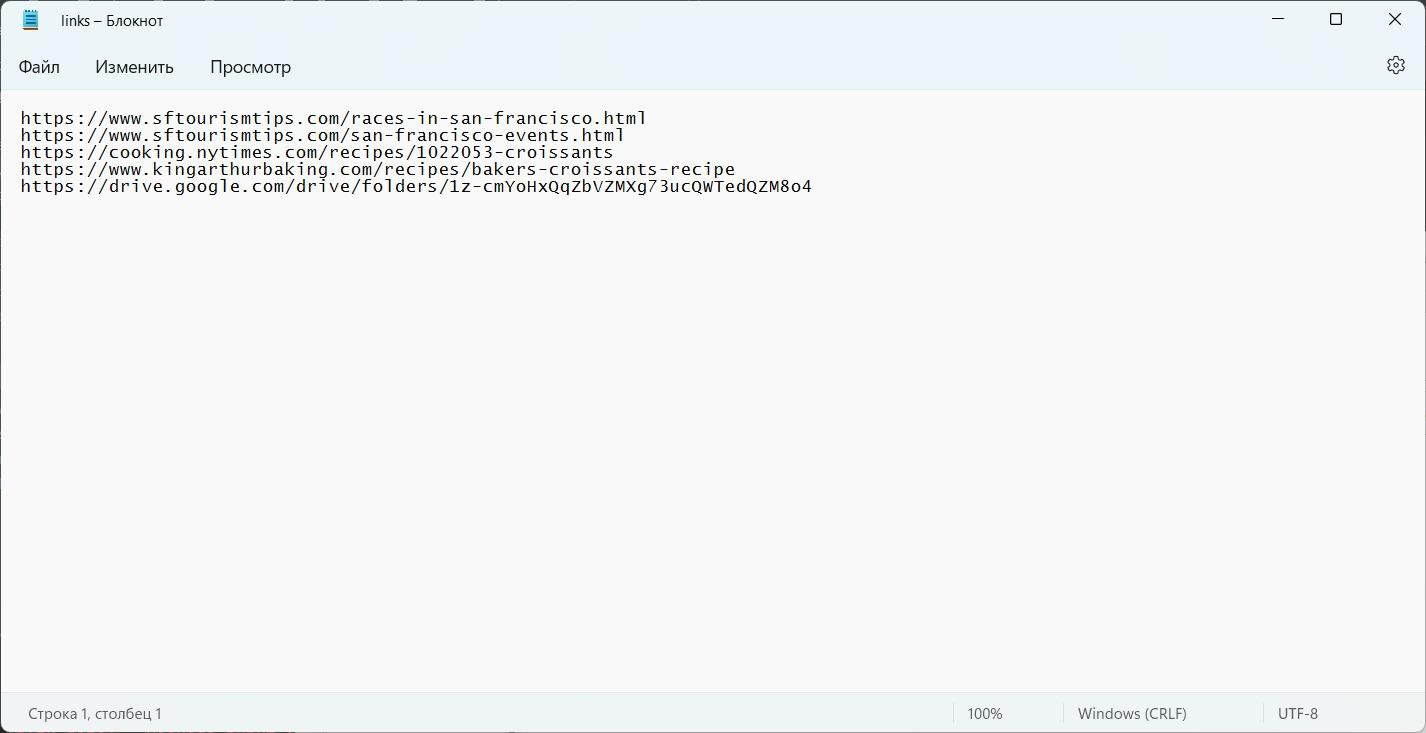


Рисунок 7. Содержимое файл со ссылкам (пример)

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Действие** | **Ожидаемый результат** | **Фактический результат** |
| 1. | Ввести адрес файла, в котором хранятся ссылки в поле “Введите путь до файла формата \*.txt. | В поле отображается путь к файлу. |  |
| 2. | Нажать кнопку “Классифицировать” рядом с адресом файла. | Отображается таблица с результатами классификации тремя методами и итоговый результат для каждой ссылки из файла. |  |

### **Испытание 7. Классификация введенного текста**

Предусловие: Программа запущена, отображается стартовое окно программы (см. Испытание 2).

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Действие** | **Ожидаемый результат** | **Фактический результат** |
| 1. | В поле “Введите текст для классификации” ввести текст на английском языке. | В поле отображается введенный текст. |  |
| 2. | Нажать кнопку “Классифицировать” рядом с адресом файла. | Отображается таблица с результатами классификации для введенного текста. |  |

# 

# **Список литературы**

* <https://habr.com/ru/post/538450/>
* <https://habr.com/ru/post/538458/>
* <https://izv.etu.ru/assets/files/izv-etu-4-2016-12-20.pdf>
* <https://github.com/Shaurov05/Website-Classification>
* <https://ana.cachopo.org/datasets-for-single-label-text-categorization>
* <https://archive.ics.uci.edu/ml/datasets.php>
* <https://www.kaggle.com/datasets/shawon10/url-classification-dataset-dmoz>
* <https://www.kaggle.com/code/swarnabha/pytorch-text-classification-torchtext-lstm/notebook>
* <https://www.kaggle.com/datasets/jensenbaxter/10dataset-text-document-classification>