УДК 004

Фильтр спама на языке программирования Python в google colab

Романов Даниил Алексеевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
Целью данной статьи является, создание программы классифицирующей спам-сообщения. Программа написана на языке программирования python, с использованием модулей pandas, string, nltk для обработки текста. Результатом исследования станет готовая программа с подробным описанием ее реализации.
Ключевые слова: google colab, Python, фильтр спама, pandas, string, nltk

Spam filter in Python programming language in google colab

Romanov Daniil Alekseevich
Sholom-Aleichem Priamursky State University
Student

Abstract
The purpose of this article is to create a program that classifies spam messages. The program is written in the python programming language, using the pandas, string, nltk modules for text processing. The result of the study will be a finished program with a detailed description of its implementation.
Keywords: google colab, python, spam filter, pandas, string, nltk

1 Введение

1.1 Актуальность

Актуальность проблемы спама сегодня ни у кого не вызывает сомнений. Достаточно лишь привести цифру, что более половины всех электронных писем, поступающих в корпоративные сети, являются спамом в том или ином виде. Потери от спама, которые несут корпоративные пользователи, исчисляются десятками тысяч долларов за счет потери рабочего времени и использовании сетевых ресурсов. Справиться с похожими задачами и помогает фильтр спама.

1.2 Обзор исследований

В своей работе А.С. Катасёв, Д.В. Катасёва решали задачи нейросетевой технологии для классификации электронных почтовых сообщений. Предлагаемая ими технология применяется для классификации сообщения на категории ‘спам’/’не спам’ [1]. Мезенцева Е.М. предложила схему повышения качества фильтрации спама в сообщениях интерактивных

разделов сайтов на основе совмещения работы классификаторов Байеса и Фишера [2]. И. А. Пономаренко рассматривает различные алгоритмы систем автоматического распознавания спама с целью выявления наиболее эффективных [3]. Ильичева З.С. описывает методы обнаружения спама и принципы их работы, применяя их в практических задачах [4].

1.3 Цель исследования

Цель исследования – создать программу на языке программирования Python фильтрующую спам.

2 Материалы и методы

Для создания фильтра спама используются модули pandas, string и nltk. Используется среда разработки google colab. В работе используются данные с платформы stepik[5].

3 Результаты и обсуждения

Перед началом работы требуется войти в среду разработки google colab (рис.1), а так же иметь google диск.

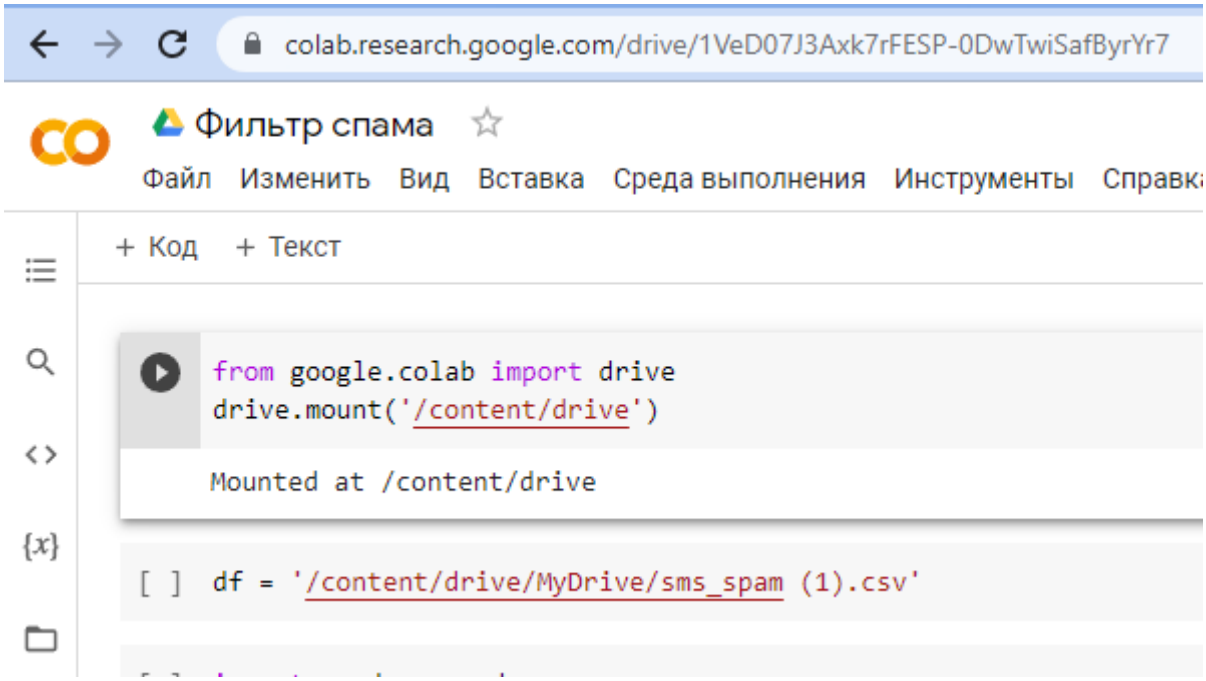

Рисунок 1 – Среда разработки google colab

Скачиваем файл и загружаем его на google диск: https://drive.google.com/file/d/1OcB9tROngJt4Wl0y8HdxLc6pq6fBjOp/edit [5]. Он содержит в себе таблицу со спам-сообщениями и нормальными. На основе этой таблицы программа научиться классифицировать сообщения. Подключаем google диск к google colabe. Для этого нажимаем на иконку папки с правой стороны и выбираем значок ‘подключить диск’ (рис.2).

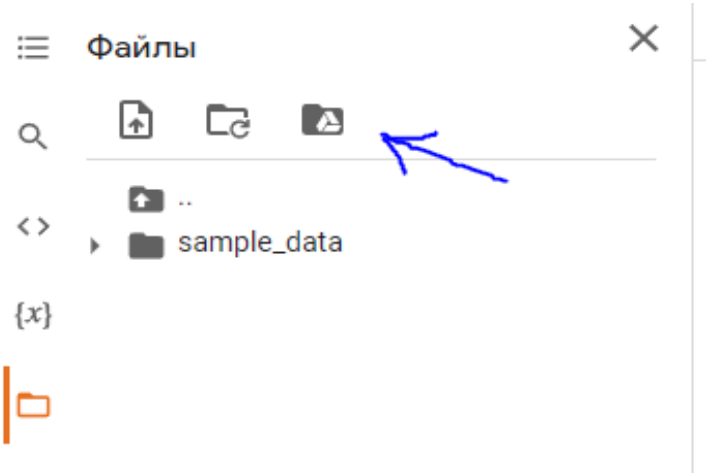

Рисунок 2 – Подключение google диска к google colab

Создаём переменную и загружаем в неё таблицу (рис.3). Скопируйте путь к таблице и вставьте в кавычки.

```
[ ]  df = '/content/drive/MyDrive/sms_spam (1).csv'
```

Рисунок 3 – Загружаем таблицу в переменную

Загружаем библиотеку pandas. После чего в новую переменную загружаем таблицу, попутно именуя столбцы как label и sms. Просматриваем таблицу убедившись, что всё в порядке (рис.4).

```
[ ]  import pandas as pd

     data = pd.read_csv(df, names = ['label', 'sms'])
     data.head()
```

	label	sms
0	type	text
1	ham	Hope you are having a good week. Just checking in
2	ham	K..give back my thanks.
3	ham	Am also doing in cbe only. But have to pay.
4	spam	complimentary 4 STAR Ibiza Holiday or £10,000 ...

Рисунок 4 – Просмотр таблицы

После того как набор данных загружен, нужно адаптировать его для функции ‘прогнозирования’, используя модули string и nltk. Эта функция будет определять с какой вероятностью введённое пользователем сообщение является спамом.

Выполняем следующие преобразования для каждого из сообщений:

- Заглавные буквы: преобразуем все заглавные буквы в строчные буквы.
- Пунктуация: удалим все знаки препинания.
- Стоп-слова: удалим все часто используемые слова, такие как «Я, или, она, сделала, ты, чтобы».
- Токенизация: мы токенизируем содержимое SMS, в результате чего будет составлен список слов для каждого сообщения.

Загружаем стоп-слова и знаки препинания и смотрим на их содержание (рис. 5).

```
[ ]  #загрузка знаков препинания
     import string
     import nltk
     nltk.download('stopwords')
     nltk.download('punkt')

     stopwords = nltk.corpus.stopwords.words('english')
     punctuation = string.punctuation

     print(stopwords[:5])
     print(punctuation)
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
['i', 'me', 'my', 'myself', 'we']
!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
```

Рисунок 5 – Загрузка стоп-слов и знаков препинания

Обрабатываем таблицу с помощью функции, удаляя стоп-слова, знаки препинания и разбивая предложение на слова. В результате чего получится список токенов без знаков препинания, стоп-слов или заглавных букв (рис.6).

```
[ ]  #предварительная обработка содержимого sms-сообщений
     def pre_process(sms):
        lowercase = "".join([char.lower() for char in sms if char not in punctuation])
        tokenize = nltk.tokenize.word_tokenize(lowercase)
        remove_stopwords = [word for word in tokenize if word not in stopwords]
        return tokenize

     data['processed'] = data['sms'].apply(lambda x: pre_process(x))
     data.head()
```

	label	sms	processed
0	type	text	[text]
1	ham	Hope you are having a good week. Just checking in	[hope, you, are, having, a, good, week, just, ...
2	ham	K..give back my thanks.	[kgive, back, my, thanks]
3	ham	Am also doing in cbe only. But have to pay.	[am, also, doing, in, cbe, only, but, have, to...
4	spam	complimentary 4 STAR Ibiza Holiday or £10,000 ...	[complimentary, 4, star, ibiza, holiday, or, £...

Рисунок 6 – Предварительная обработка sms-сообщений

После того, как каждое SMS разбилось на отдельные слова, можно приступить к созданию двух разных списков: в одном будут находиться слова из сапам-сообщений, в другом из нормальных (рис.7). На результат будет влиять то, как часто конкретное слово встречается в списке спам-сообщений или нормальных сообщений.

```
[ ]  #категоризация слов, связанных с spam/ham
     def categorie_words():
       spam_words = []
       ham_words = []

       #слова, связанные с spam
       for sms in data['processed'][data['label'] == 'spam']:
         for word in sms:
           spam_words.append(word)

       #слова, связанные с ham
       for sms in data['processed'][data['label'] == 'ham']:
         for word in sms:
           ham_words.append(word)

       return spam_words, ham_words

     spam_words, ham_words = categorie_words()

     print(spam_words[:5])
     print(ham_words[:5])
```

```
['complimentary', '4', 'star', 'ibiza', 'holiday']
['hope', 'you', 'are', 'having', 'a']
```

Рисунок 7 – Разделение слов на два списка и проверка его содержания

Теперь можно перейти к функции прогнозирования, которая будет определять с какой вероятностью вводимое сообщение является спамом (рис.8).

```
[ ]  #он просматривает все слова из пользовательского ввода и подсчитывает их вхождения как в ham_words, так и в spam_words
     def predict(sms):
        spam_counter = 0
        ham_counter = 0

        for word in sms:
           spam_counter += spam_words.count(word)
           ham_counter += ham_words.count(word)

        print('***RESULTS***')
        if ham_counter > spam_counter:
          #повышение точности
          accuracy = (ham_counter / (ham_counter + spam_counter)) * 100
          print('message is not spam, with {}% accuracy'.format(accuracy))
        elif spam_counter > ham_counter:
          accuracy = (spam_counter / (ham_counter + spam_counter)) * 100
          print('message is spam, with {}% accuracy'.format(accuracy))
        else:
          print('message could be spam, with 50% accuracy')
```

Рисунок 8 – Функция прогнозирования

Создаём блок кода, позволяющий пользователю ввести сообщение (рис. 9).

```
[ ]  #сбор данных пользователя
     user_input = input('Please type a spam or ham message to check if our function predicts accurately\n')

     Please type a spam or ham message to check if our function predicts accurately
     Hello my techer
```

Рисунок 9 – переменная отвечающая за пользовательский ввод

Нужно будет собрать строку слов от пользователя, предварительно обработать ее, а затем, наконец, передать их в качестве входных данных в функцию прогнозирования (рис.10).

```
[ ]  processed_input = pre_process(user_input)
     predict(processed_input)

     ***RESULTS***
     message is not spam, with 98.3790523690773% accuracy
```

Рисунок 10 – Обработка пользовательского ввода в функции прогнозирования

Выводы

В данной работе была создана и описана программа фильтрующая спам, а также описаны основные методы и принципы работы алгоритмов, фильтрующих спам. Код программы можно посмотреть по ссылке

https://colab.research.google.com/drive/1VeD07J3Axk7rFESP-0DwTwiSafByrYr7?usp=sharing

Библиографический список

1. Катасёв А. С., Катасёва Д. В. Разработка нейросетевой системы классификации электронных почтовых сообщений //Вестник Казанского государственного энергетического университета. 2015. №. 1 (25). С.68-78
2. Мезенцева Е. М. Исследование и разработка статистических алгоритмов фильтрации сообщений в интерактивных ресурсах инфо-коммуникационных сетей : дис. Поволжская государственная академия телекоммуникаций и информатики, 2013.
3. Пономаренко И. А. Разработка системы автоматического распознавания спама //Информационные технологии. 2019. С. 175-175.
4. Ильичева З.С. Исследование методов обнаружения спама и принципов работы компьютерных спам-фильтров // В сборнике: Молодой исследователь: вызовы и перспективы. Сборник статей по материалам LXVIII международной научно-практической конференции. 2018. С. 278-282
5. Курс “Анализ данных это просто” URL: https://stepik.org/lesson/414060/step/1?auth=login&unit=403565