ГУАП

КАФЕДРА № 43

ОТЧЁТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| д-р техн. наук, доцент |  |  |  | С. И. Колесникова |
| должность, уч. Степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ОТЧЁТ О ЛАБОРАТОРНОЙ РАБОТЕ №2 | | | | | |
| НАИВНЫЙ БАЙЕСОВСКИЙ КЛАССИФИКАТОР | | | | | |
| по дисциплине: СИСТЕМНЫЙ АНАЛИЗ | | | | | |
|  | | | | | |
| РАБОТУ ВЫПОЛНИЛ | | | | | |
| СТУДЕНТ ГР. | 4330М |  |  |  | А.А. Кинько |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2023

**Цель работы**

Целью работы является освоение наивного байевского классификатора

**Текст задания**

Согласно варианту №8:

1. Выбрать данные для распознавания спама (<https://github.com/stedy/Machine-Learning-with-R-datasets>)
2. Создать фразу для распознавания класса
3. Создать правило на основе НБК для классификации

**Ход решения**

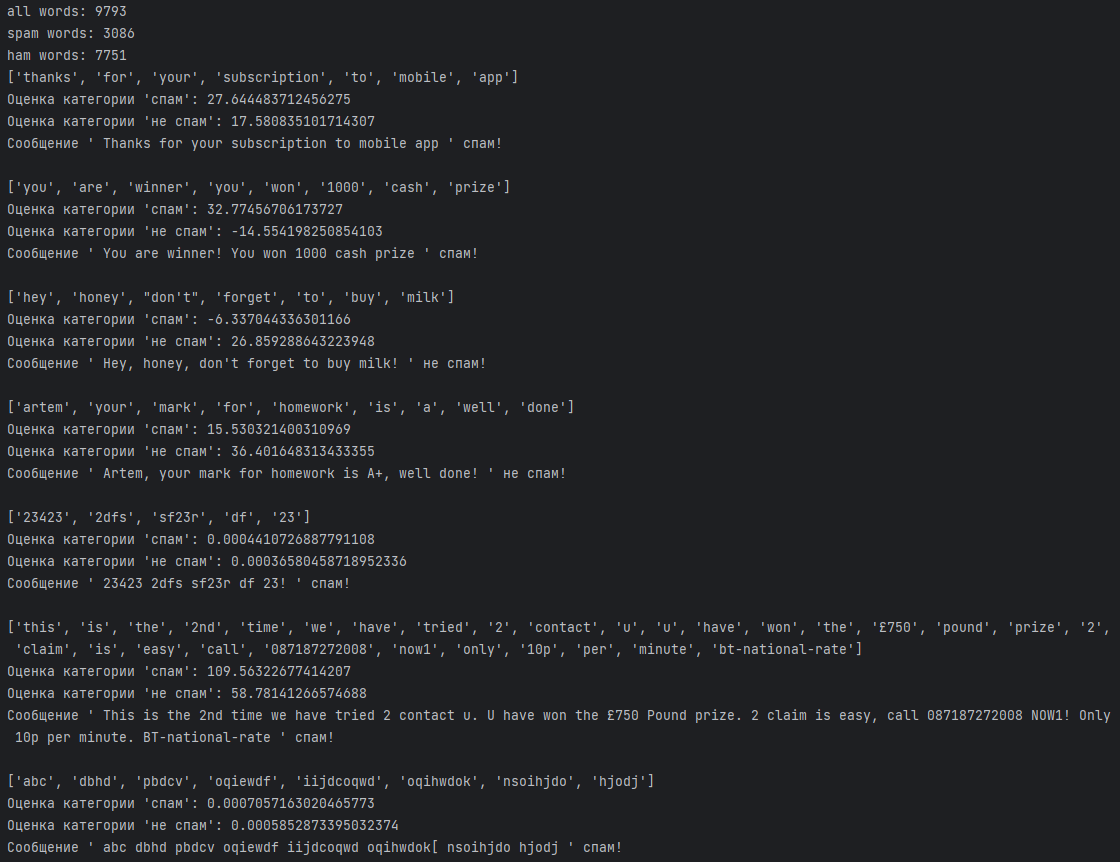
Для классификации фразы составим базу знаний. Фразы со спамом будем помечать меткой «spam», все остальные – «ham». Далее пройдемся по каждой фразе, приведем ее к нижнему регистру, уберем все символы, которые не являются латинскими буквами (предполагается, что весь датасет состоит из английских сообщений), а также разделим строку на слова. Далее для каждого слова подсчитаем количество вхождений во фразы с меткой спама и не спама.

Составив фразу на проверку, рассчитаем наиболее вероятный класс для нее:

– коэффициент, выбираемый эмпирически (примем равным 0.5); – -е слово в фразе; – проверяемый класс («спам» или «не спам»), – количество вхождений -го слова в класс; – количество вхождений всех слов в класс; – количество всех уникальных слов в базе знаний.

Так как вероятность может быть слишком малой, воспользуемся модернизацией формул, где используется логарифмирование:

Реализовав данный алгоритм с помощью языка Python (листинг представлен в Приложении А), получим несколько результатов проверки.



**Выводы**

В ходе выполнения лабораторной работы были получены знания и навыки реализации и применения наивного байесовского классификатора на примере решения задачи классификации сообщений на «спам» и «не спам». На относительно малом количестве обучающих фраз были получены результаты: фразы, которые человек воспримет как рекламные, относились к спаму; случайный набор символов относился к спаму, однако разница между оценками очень мала, потому можно судить, что такие фразы классификатор не может распознать и результат зависит только от пропорции количества фраз «спам» и «не спам».

**Приложение А**

Листинг программы (Python 3.6):

# Вариант №8  
import io  
import math  
import string  
  
spam\_table = dict()  
  
  
def create\_spam\_table(path):  
 with io.open(path, encoding='utf-8') as file:  
 for line in file:  
 line = line.split(',', 1)  
 spam\_class = line[0]  
 message = [word.lower().strip(string.punctuation) for word in line[1].rsplit()]  
 print(message)  
 for word in message:  
 if word == '':  
 continue  
 else:  
 if word not in spam\_table.keys():  
 if spam\_class == '"ham"':  
 spam\_table.update({word: [0, 1]})  
 else:  
 spam\_table.update({word: [1, 0]})  
 else:  
 spam\_num, ham\_num = spam\_table.get(word)[0], spam\_table.get(word)[1]  
 if spam\_class == '"ham"':  
 spam\_table.update({word: [spam\_num, ham\_num + 1]})  
 else:  
 spam\_table.update({word: [spam\_num + 1, ham\_num]})  
 print(spam\_table)  
 all\_words = 0  
 spam\_words = 0  
 ham\_words = 0  
 for key in spam\_table.keys():  
 all\_words += 1  
 if spam\_table.get(key)[0] > 0:  
 spam\_words += 1  
 if spam\_table.get(key)[1] > 0:  
 ham\_words += 1  
 print("all words:", all\_words)  
 print("spam words:", spam\_words)  
 print("ham words:", ham\_words)  
 return all\_words, spam\_words, ham\_words  
  
  
def check\_spam(message, all\_words, spam\_words, ham\_words, alpha):  
 words = message.split(' ')  
 words = [word.lower().strip(string.punctuation) for word in words]  
 probabilities\_spam = []  
 probabilities\_ham = []  
 print(words)  
  
 for word in words:  
 if word in spam\_table.keys():  
 probabilities\_spam.append(math.log(spam\_table.get(word)[0] + alpha / (alpha \* all\_words + spam\_words)))  
 probabilities\_ham.append(math.log(spam\_table.get(word)[1] + alpha / (alpha \* all\_words + ham\_words)))  
 else:  
 probabilities\_spam.append(alpha / (alpha \* all\_words + spam\_words))  
 probabilities\_ham.append(alpha / (alpha \* all\_words + ham\_words))  
  
 print("Оценка категории 'спам':", sum(probabilities\_spam))  
 print("Оценка категории 'не спам':", sum(probabilities\_ham))  
 if sum(probabilities\_spam) > sum(probabilities\_ham):  
 print("Сообщение '", message, "' спам!\n")  
 else:  
 print("Сообщение '", message, "' не спам!\n")  
  
  
all\_words, spam\_words, ham\_words = create\_spam\_table("spam-data.csv")  
# check\_spam("Магазине гора яблок. Купи семь килограмм шоколадку", all\_words, spam\_words, ham\_words, 1)  
check\_spam("Thanks for your subscription to mobile app", all\_words, spam\_words, ham\_words, 2)  
check\_spam("You are winner! You won 1000 cash prize", all\_words, spam\_words, ham\_words, 2)  
check\_spam("Hey, honey, don't forget to buy milk!", all\_words, spam\_words, ham\_words, 2)  
check\_spam("Artem, your mark for homework is A+, well done!", all\_words, spam\_words, ham\_words, 2)  
check\_spam("23423 2dfs sf23r df 23!", all\_words, spam\_words, ham\_words, 2)  
# "spam","This is the 2nd time we have tried 2 contact u. U have won the £750 Pound prize. 2 claim is easy, call 087187272008 NOW1! Only 10p per minute. BT-national-rate."  
check\_spam("This is the 2nd time we have tried 2 contact u. U have won the £750 Pound prize. 2 claim is easy, call 087187272008 NOW1! Only 10p per minute. BT-national-rate", all\_words, spam\_words, ham\_words, 2)  
check\_spam("abc dbhd pbdcv oqiewdf iijdcoqwd oqihwdok[ nsoihjdo hjodj", all\_words, spam\_words, ham\_words, 2)

Файл «spam-data.csv» содержит строки формата:

"ham","Its a part of checking IQ"  
"spam","500 New Mobiles from 2004, MUST GO!”