Анастасия Костяницына
БКЛ-151

ОТЧЕТ № 2
Определение языка

Так как в code2lang всего 432 языка, в данном задании определение языка осуществлялось лишь некоторых из них:

'be', 'bg', 'ky',
'lez' , 'mhr', ‘mn',
'mk', 'mo', 'ru',
'sty', 'uk', 'en',
'kk', 'uk', 'fr'

Сначала был реализован алгоритм определения языка, опирающийся на частотный список слов для языков.

Первый метод: частотные слова

Для формирования частотного списка слов для языков были использованы 100 тестов из википедии для каждого языка (так как возникали некоторые ошибки при скачивании текстов, реальное кол-во текстов может варьироваться). Каждый такой текст был разбит на слова и избавлен от знаков препинания. Далее для получившихся словоформ была определена их частотность в корпусе (для каждого языка отдельно).

Есть два словаря:
word_langs - ключ = слово, а значение = языки, в которых оно встретилось, и частотность этого слова в языке
lang_word - ключ = язык, значение = частотные списки

Предсказание языка происходит следующим образом:

После очистки текстов от знаков и деления их на слова, программа находит общие слова между частотными списками языков и заданного текста, зачем для каждого такого слова определяется частотность их в конкретном языке. Значения для каждого совпавшего слова суммируются. Язык с наибольшей суммой побеждает.

Результаты работы программы:

	precision	recall	f1-score	support
be	0.99	0.92	0.95	96
bg	0.66	1.00	0.79	98
en	0.79	1.00	0.88	93
fr	0.96	1.00	0.98	93
kk	0.98	0.94	0.96	100
ky	1.00	0.90	0.95	100
lez	0.99	0.86	0.92	100
mhr	0.94	0.98	0.96	96
mk	0.90	1.00	0.95	100
mn	1.00	0.94	0.97	100
ru	0.85	0.56	0.68	95
uk	1.00	0.77	0.87	96
avg / total	0.92	0.91	0.91	1167

Второй метод: частотные символьные n-граммы

По аналогии с предыдущим пунктом создаем списки n-граммы (n=3) для всех языков выборки. Сортируем по частотности и выбираем только 300 первых из них.

Словарь:

all_ngram: ключ - язык, значение - топ-300 ngram.

Предсказывание языка:

Входной текст делится на слова, убираются знаки препинания, определяется список 300 самых частотных ngram, затем сравниваются два вектора: вектор из словаря и вектор ngram для входного текста. Находим расстояние между ними. Язык с наименьшим расстоянием побеждает.

Результаты работы:

	precision	recall	f1-score	support
be	1.00	0.99	0.99	96
bg	0.98	0.99	0.98	98
en	0.92	1.00	0.96	93
fr	1.00	1.00	1.00	93
kk	0.99	0.99	0.99	100
ky	0.99	0.94	0.96	100
lez	0.98	0.95	0.96	100
mhr	0.98	0.97	0.97	96
mk	0.98	0.99	0.99	100
mn	1.00	0.99	0.99	100
ru	0.98	0.99	0.98	95
uk	1.00	1.00	1.00	96
avg / total	0.98	0.98	0.98	1167

Ошибки при работе алгоритмов

Первый вариант:

```
[[ 88   2   3   1   0   0   0   0   1   0   1   0]
 [  0  98   0   0   0   0   0   0   0   0   0   0]
 [  0   0  93   0   0   0   0   0   0   0   0   0]
 [  0   0   0  93   0   0   0   0   0   0   0   0]
 [  1   1   1   1  94   0   0   0   1   0   1   0]
 [  0   2   2   0   1  90   1   2   1   0   1   0]
 [  0   0   7   1   0   0  86   3   0   0   3   0]
 [  0   0   2   0   0   0   0  94   0   0   0   0]
 [  0   0   0   0   0   0   0   0 100   0   0   0]
 [  0   0   4   0   1   0   0   1   0  94   0   0]
 [  0  33   4   1   0   0   0   0   4   0  53   0]
 [  0  13   2   0   0   0   0   0   4   0   3  74]]
```

Второй вариант:

```
[[94  0  0  0  1  0  0  0  0  0  0  0]
 [ 0 99  1  0  0  0  0  0  0  0  0  0]
 [ 0  0 97  0  1  0  0  0  0  0  0  0]
 [ 1  2  1 94  0  0  0  0  0  1  1  0]
 [ 0  0  0  0 93  0  0  0  0  0  0  0]
 [ 0  0  0  0  1 99  0  0  0  0  0  0]
 [ 0  0  0  0  0  0 93  0  0  0  0  0]
 [ 0  0  0  0  0  0  0 96  0  0  0  0]
 [ 1  0  0  1  3  0  0  0 95  0  0  0]
 [ 0  0  0  0  0  0  0  0  0 99  1  0]
 [ 0  0  0  0  1  0  0  0  2  0 93  0]
 [ 0  0  0  0  1  0  0  0  0  0  0 95]]
```

Как видно по результатам, второй алгоритм работает лучше.