**HW1**

Прудников Евгений

Код программы: <https://github.com/EvgeniyPrudnikov/Magistracy/blob/master/IR/main.ipynb>

[**Нормализация текста** 2](#_Toc478941036)

[**Построение инвертированного индекса** 2](#_Toc478941037)

[**Алгоритм поиска** 3](#_Toc478941038)

[**Результаты** 3](#_Toc478941039)

# **Нормализация текста**

Для нормализации текстов документов было использовано, сначала, регулярное выражение «\W+» которое чистило текст от всех символов кроме букв чисел и знака нижнего подчеркивания (\_), затем убирались стоп слова с помощью библиотеки nlkt и модуля stopwords для английского языка, далее проводился стэмминг с помощью все того же nlkt, для стемминга был выбран стеммер портера.

Такая же нормализация была применена к текстам запросов.

# **Построение инвертированного индекса**

Был построен простой инвертированный индекс, который имеет вид:

{

‘\_\_metadata\_\_’: {

‘num\_of\_docs’: …,

‘index\_len’: …,

‘avgdl’: ...,

‘avg\_docs\_len’: …,

‘max\_docs\_len’: …

},

‘term1’: [(doc\_id, term\_frequency, doc\_lenght), (…)],

‘term2’: [(doc\_id, term\_frequency, doc\_lenght), (…)],

…

}

В поле metadata содержатся общие характеристики индекса, такие как длина индекса (количество термов) (index\_len), количество документов (num\_of\_docs), средняя длина документа (avgdl), средняя длина списка словопозиций (avg\_docs\_len), максимальная длина списка словопозиций (max\_docs\_len).

Для каждого терма индекс хранит список словопозиций, который включает идентификатор документа, частота этого терма в этом документе, длина документа (кол-во термов)

Статистики индексов:

Индекс по документам:

{

'avg\_docs\_len': 18.416861329369294,

'avgdl': 61.946428571428569,

'index\_len': 4709,

'max\_docs\_len': 730,

'num\_of\_docs': 1400

}

Индекс по заголовкам:

{

'avg\_docs\_len': 8.5413819286256647,

'avgdl': 8.0350000000000001,

'index\_len': 1317,

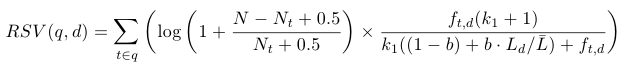
'max\_docs\_len': 365,

'num\_of\_docs': 1400

}

# **Алгоритм поиска**

Был реализован алгоритм поиска, основанный на BM25. Для каждого найденного документа для каждого терма из запроса подсчитывалась RSV по формуле:



С параметрами *k1* = 1.2, *b* = 0.75 (так же я пробовал другие комбинации параметров, однако, на результаты поиска это почему-то не повлияло)

Далее, результаты сортировались по убыванию RSV и top-10 записывался в файл.

# **Результаты**

Были получены следующие результаты (eval.py):

search by docs:

mean precision: 0.22024691358

mean recall: 0.306024492383

mean F-measure: 0.256145210109

MAP@10: 0.21955973307

search by titles:

mean precision: 0.173333333333

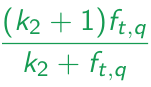
mean recall: 0.246569763465

mean F-measure: 0.203564866877

MAP@10: 0.170676459926

Видно, что поиск по заголовкам документов показывает чуть худшие результаты по сравнению с поиском по полному документу.

Так же я пробовал добавлять к формуле RSV множитель



И подбирать параметр *k2,* однако, это никак не улучшилорезультаты, а при значениях параметра 10, 100, 1000 даже значительно ухудшило.