Для составления рекомендаций пользователям, используются два модуля, реализованных на языке Python.

Первый модуль анализирует историю заказа пользователя и рекомендует книги из аналогичных историй рубрик из книг, подходящих по возрастному ограничению пользователя

Второй модуль обрабатывает аннотации книг, выделяя из них ключевые слова и уточняет по ним список рекомендованных пользователю книг.

Модуль recommend.py

Данный модуль содержит функцию для составления списка рекомендации для пользователей библиотек.

Зависимости

Для работы необходимы библиотеки для python:

* Random – генерация случайных значений
* psycopg2 – подключение к базе данных postgres
* pandas – работа с массивами данных
* numpy – работы с многомерными массивами
* os – работа с файловой системой
* csv – работа с .csv файлами

Установка библиотек:

pip install -r requirements\_r.txt

Составление рекомендации

**Вызов функции:**



**Входные данные:**

user\_list: Список id пользователей, кому необходимо дать рекомендацию

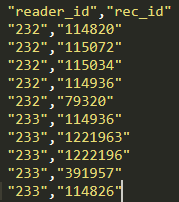
amount\_of\_users\_blocks: необязательный параметр, по умолчанию = 1, нужен в случае пустого user\_list , число блоков, на которые будут поделены все пользователи

block\_len: необязательный параметр, по умолчанию = 1000, нужен в случае пустого user\_list, число пользователей, которые будут содержаться в блоке

**Выходные данные:**

Файл **export.csv** (id пользователя, id рекомендованной книги)

**Пример файла export.csv:**



Подробнее об алгоритме:

Если подается пустой массив, то для него должны подаваться аргументы **amount\_of\_users\_blocks** и **block\_len**. Из таблиц циркуляции(circulaton\_\*.csv) известно, что интересующих записей с пользователями примерно 177 тысяч, поэтому можно взять amount\_of\_users\_blocks = 180 ,block\_len = 1000.

Если подается непустой массив, то **amount\_of\_users\_blocks** и **block\_len** указывать не надо.

Каждый логический блок функции будет начинаться с хэштега (#number )

**# 1. Подключение к базе данных**

Создается подключение, которое используется библиотеками **psycopg2** и **pandas**  для исполнения PostgreSQL команд в коде функции.

**#2. Алгоритм**

**# 2.1. Заполнение локального списка пользователей**

Для каждого блока пользователей, если **user\_list**  не пуст, то заполняется локальный массив с id пользователей **users\_id** из **user\_list** , в противном случае **users\_id** получается выборкой id из таблицы **circulations** записей**,** где есть хотя бы 1 взятая книга, это определяется по статусу **'На руках'**.

**# 2.2. Создание промежуточного файла и словаря рубрика-список книг по рубрике**

При помощибиблиотеки **pandas** создается дополнительный файл(если его нет) **rubrics\_booklist.csv** для хранения данных : рубрика - список книг по рубрике.

Потом файл переносится в локальную переменную **rubrics**.

**# 2.3. Создание списка рекомендаций для каждого пользователя**

Создается локальная переменная **recomend\_list**  для хранения списка рекомендаций.

Из таблицы **circulations** получаются id книг, которые брал пользователь, ими заполняется **user\_books\_id.**

Из объединения таблиц **rubrics** и **main\_catalog** получаются рубрики взятых книг, ими заполняется **user\_rubrics.**

Из таблицы **main\_catalog** получаются возрастные ограничения книг, которые брал пользователь, ими заполняется **user\_ager, user\_ager** фильтруется на предмет **None.**

Для каждой книги проверяется наличие рубрик и возрастных огрничений, если они указаны, то в соответствии им составляются рекомендации, заполняется **recomend\_list.**

Если у пользователя есть хотя бы одна рубрика, и в списке рекомендаций книг меньше 5, то он дополняется случайными книгами у которых есть соответствие по рубрикам с рубриками пользователя. Для избежание бесконечного цикла вводится ограничивающая переменная **i**.

Перед выводом массив **recomend\_list** перемешивается и отсекается только 5 элементов.

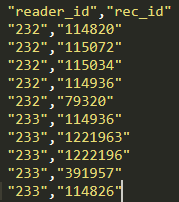
**# 3. Создание файла export.csv**

Список рекомендаций преобразуется в удобный для дальнейшей работы вид:

id пользователя, id рекомендованной книги

Он заносится в файл **export.csv,** если он существует, то просто дополняется записями , если его нет, то создается и первой строкой добавляется названия столбцов.

Пример:



Модуль parse.py

Данный модуль содержит функции для выделения ключевых слов из аннотаций кинг и уточнения списка рекомендованных книг для читателя

Зависимости

Для работы необходимы библиотеки для python:

* psycopg2 – подключение к базе данных postgres
* pymystem3 – обертка на python для морфологического анализатора Yandex Mystem 3.1
* spacy – библиотека для NLP задач с загруженной моделью ru\_core\_news\_sm

Установка библиотек:

* pip install psycopg2
* pip install pymystem3
* pip install spacy
* python -m spacy download ru\_core\_news\_sm

Для запуска скрипта необходимо указать в аргументах запуска:

* базу
* имя пользователя
* хост базы
* порт
* пароль

Выделение ключевых слов из аннотаций

Изначальные настройки для работы:

delim = " AAAAA " # Разделитель между аннотациями

batch\_size = 3000 # Количество аннотаций, обрабатываемых за раз

nlp = spacy.load("ru\_core\_news\_sm") # Загрузка модели для NLP

nlp.max\_length = 3000000 # Увеличения количества символов, которые может за раз обработать модель

m = Mystem() # инициализация морфологического анализатора

Для запуска необходимо вызвать функцию **parse\_until\_nothing**. Функция вызывает **parse\_batch**, пока она не 5 раз подряд не False, обозначающий, что нет новых аннотаций для обработки

Подробнее об алгоритме:

Функция **parse\_batch**:

Аргументы отсутствуют

Возвращает:

bool

* **True**, если обработка была выполнена успешно
* **False**, если не было обработано ни одной строки

Запускает запрос на выдачу **batch\_size** аннотаций, которые еще не были разбиты на ключевые слова

Если таких нет – возвращает **False**

Иначе собирает все аннотации в одну строку, которую отправляет на выделение из нее ключевых слов.

Полученные ключевые слова отправляются в функцию **pass\_to\_lemmatize**

Возвращается **True**

Функция **pass\_to\_lemmatize:**

Аргументы:

* **words** – str. Строка, содержащая id книг и их аннотации

Запускается морфологический анализатор для лемматизации полученного текста.

Его результат отдается в **prepare\_to\_output**

Функция **prepare\_to\_output:**

Аргументы:

* rows – list. Список строк, содержащих в себе id книги и ее ключевые слова

Разбивает полученную строку по кортежам (id книги, ключевое слово) и выводит их в таблицу

Подбор рекомендаций пользователю:

Для уточнения рекомендаций для пользователя запускается функция **get\_recommend\_usr**

Подробнее об алгоритме:

Функция **get\_recommend\_usr:**

Аргументы:

* **user\_id** – int. id читателя
* **recommend\_ids** – list. Список id книг, рекомендуемых пользователю

Собирает id книг, прочитанные пользователем и отправляет все 3 параметра в **get\_user\_rec\_by\_words**

Функция **get\_user\_rec\_by\_words**

Аргументы:

* **user\_id** – int. id читателя
* **read\_ids** – list. Список id прочитанных пользоватлем книг
* **recommend\_ids** – list. Список id книг, рекомендуемых пользователю

Возвращает:

* В 1й файл id пользователи и до 5 рекомендованных книг
* В 2й файл До 5 строк, сожержащих id читателя, название, автор, год издания, аннотация, id книги.
* Каждое поле заключено в кавычки и разделено через **;**

Получает уточненные рекомендации из функции **get\_same\_books\_from\_given** и записывает рекомендации и информацию о них в файлы

Функция **get\_same\_books\_from\_given**

Аргументы:

* **read\_ids** – list. Список id прочитанных пользоватлем книг
* **recommend\_ids** – list. Список id книг, рекомендуемых пользователю

Возвращает:

* list. Список, длиной до 5 книг, наиболее подходящих пользователю

Проверяет длину списка рекомендованных книг. Если он не превышает 5, то такой список в уточнении не нуждается.

Запрашиваются ключевые слова из прочитанных книг и рекомендованных книг функциями

**get\_words** и **get\_dict\_words\_by\_ids.**

Если для них не находится ключевых слов – обрезается список рекомендаций до 5.

Далее ведется подсчет количества совпадающих ключевых слов и выдаются 5 с наибольшим количеством совпадающих слов.

Функция **get\_words**

**Аргументы:**

* **read\_ids** – list**.** Список id прочитанных пользователем книг

Возвращает:

set

* Множество из ключевых слов, относящихся к прочитанным книгам
* Если таковых нет – None

Функция **get\_dict\_words\_by\_ids**

Аргументы:

* **recommend\_ids** – list.Список id книг, рекомендуемых пользователю

Возвращает:

dict

* Словарь в формате {id : Множество из ключевых слов, относящихся к книге}
* Если таковых нет - None