Задание 4.

Для проведения рубрицирования необходимо использовать полученную в Задании 3 векторную модель. На первом этапе необходимо рассчитать вектор каждого текста, для этого необходимо произвести следующие операции с каждым из текстов:

– текст разбивается на массив слов;

– для каждого слова в массиве необходимо получить соответствующий ему нормализованный вектор из модели;

– для слов, вектор которых неизвестен, значение вектора считать равным 0;

– найти среднее арифметическое полученных векторов;

Таким образом каждому тексту выборки будет соответствовать числовое значение вектора. Затем необходимо произвести рубрицирование:

– отсортировать значения векторов;

– используя отсортированные значения необходимо разделить тексты на 5-10 категорий, основываясь на близости значений векторов (числовые характеристики, соответствующие темам текстов должны иметь минимальное отклонение между собой относительно всего датасета);

– определить среднее арифметическое числового значения для каждой категории (это число будет числовым значением категории);

– получить набор ближайших по вектору слов из созданной модели (эти слова будут условно-ключевыми);

Данный индекс должен быть представлен в виде таблицы, названной «автоматическое рубрицирование», содержащей следующие поля: *имя файла (статьи), заголовок, числовое значение категории, ключевые слова.*

Для проведения автоматического рубрицирования необходимо написать программу, реализующую алгоритм из задания.

Для того, чтобы загрузить готовую модель, используйте конструкцию «w2v_model = Word2Vec.load("путь_к_файлу.model")».

Для проверки наличия слова в модели – «w2v_model.wv.has_index_for(word)».

Для получения нормализованного вектора по слову – «w2v_model.wv.get_vector(word).sum()».

Перейдем к алгоритму из задания:

– текст разбивается на массив слов;

Используйте line.split()

– для каждого слова в массиве необходимо получить соответствующий ему нормализованный вектор из модели;

– для слов, вектор которых неизвестен, значение вектора считать равным 0;

Для этих пунктов необходимо проверить наличие слова в модели, затем получить его вектор, просуммировать и увеличить счетчик слов. Альтернативно, можно вычитать из значения длины массива, полученного в первом пункте, единицу, каждый раз, когда слово отсутствует в представлении.

– найти среднее арифметическое полученных векторов;

```
result = 0
words = 0
with open(f"resources/текст_{j}.txt", 'r') as file:
    for line in file:
        line = re.sub(patterns, ' ', line)
        for word in line.split():
            if w2v_model.wv.has_index_for(word):
                result += w2v_model.wv.get_vector(word).sum()
            words += 1
```

Таким образом каждому тексту выборки будет соответствовать числовое значение вектора. Затем необходимо произвести рубрицирование:
– отсортировать значения векторов;
– используя отсортированные значения необходимо разделить тексты на 5-10 категорий, основываясь на близости значений векторов (числовые характеристики, соответствующие темам текстов должны иметь минимальное отклонение между собой относительно всего датасета);
– определить среднее арифметическое числового значения для каждой категории (это число будет числовым значением категории);
– получить набор ближайших по вектору слов из созданной модели (эти слова будут условно-ключевыми);

Для получения блиайших слов используйте «model.most_similar(positive=[your_word_vector], topn=1))», где второй параметр обозначает число получаемых слов.