Коллаборативная фильтрация на MapReduce. Описание решения

Пацация Александр, 317 группа

23 мая 2022 г.

1 Описание задания

В задании требуется реализовать один из подходов в организации рекомендательных систем – коллаборативную фильтрацию в парадигме MapReduce. Выборка представляет из себя набора $(u, i, r_{ui}$, где $u-$ пользователь, $i-$ фильм, $r_{ui}-$ рейтинг.

Для каждой пары i, j определена мера близости:

$$sim(i,j) = \frac{\sum\limits_{u \in U_{ij}} (r_{ui} - \overline{r_u})(r_{uj} - \overline{r_u})}{\sqrt{\sum\limits_{u \in U_{ij}} (r_{ui} - \overline{r_u})^2} \sqrt{\sum\limits_{u \in U_{ij}} (r_{uj} - \overline{r_u})^2}} \tag{1}$$

где $U_{ij}-$ множество пользователей оценивших фильмы i и j, $\overline{r_u}-$ средний рейтинг пользователя u.

Имея значения $sim(i,j)$, рейтинги для неизвестных фильмов определяются по следующей формуле:

$$\hat{r}_{ui} = \frac{\sum\limits_{j \in I_u} sim(i,j) r_{uj}}{\sum\limits_{j \in I_u} sim(i,j)} \tag{2}$$

где I_u- множество фильмов, которые оценил пользователь u. В задании предполагается работа с small версией датасета MovieLens. Датасет состоит из файлов:

1. ratings.csv, в каждой строке которого записаны идентификатор пользователя, идентификатор фильма и временная метка (не понадобится)

2. movies.csv, в которой каждому идентификатору фильма ставится в соответствие полное название.

2 Шаг 1

2.1 маппер

На первом шаге маппер получает на вход тройки (u, i, r) и выдает $(u, (i, r))$ с ключом u. Сложность по числу операций соответствует, количеству строк в файле ratings.csv – $O(\alpha \frac{UI}{M})$, где $U-$ количество пользователей, $I-$ количество фильмов, $\alpha-$ доля известных оценок, $M-$ количество мапперов. Сложность по памяти – $O(1)$, так как маппер ничего оне сохраняет.

2.2 редьюсер

Редьюсер, получая на вход $(u, (i, r))$, просто агрегирует все значения (i, r) по ключу u, то есть а выходе мы получаем $(u, (i_1, .., i_n), (r_{ui_1}, .., r_{ui_n}))$. Обозначим $items = (i_1, .., i_n)$, $ratings = (r_{ui_1}, r_{ui_n})$ для сокращения записи. Сложность по числу операций – $O(\alpha\frac{UI}{R}))$, где $R-$ количество редьюсеров. Для каждого пользователя редьюсер сохраняет все фильмы, которые он смотрел, поэтому сложность по памяти – $O(\beta\frac{I}{R})$, где $\beta-$ средня доля оцененных фильмов для всех пользователей.

3 Шаг 2

3.1 маппер

На втором шаге, маппер, получая на вход выход редьюсера предыдущего шага, центрирует значения $ratings = (r_{ui_1}, .., r_{ui_n}) \to (r_{ui_1} - \hat{r}_u, .., r_{ui_n} - \hat{r}_u) = ratings_c$ и для каждого i выдает $(i, r_{ui_1} - \hat{r}_u, items, ratungs_c)$ с ключом i. Сложность по числу операций опять же – $O(\alpha\frac{UI}{M})$. На одной итерации мапера хранится $O(\beta\frac{I}{M})$ значений, что и соответствует сложности по памяти.

3.2 редьюсер

Редьюсер, собрав по ключу i все значения, для каждого j собирает множество U_{ij} и подсчитывает значение $sim(i, j)$ по формуле (1) и выдает $(i, (j_1, ..j_n), (sim(i, j_1), .., sim(i, j_n)))$ по ключу i. Сложность по числу операций – $O((\alpha I)^2\frac{U}{R})$, так как такова сложность выхода маппера. Пусть $\gamma-$ средняя доля пользователей смотревших какой-либо фильм, тогда сложность по памяти составляет – $O(\gamma\frac{UI}{R})$

4 Шаг 3

4.1 маппер

Маппер получает на вход выход предыдущего шага и снова данные из файла ratings.csv. Выходы предыдущего шага преобразуются в $(i, j, sim(i, j))$ с ключом i, данные из файла ratings.csv преобразуются в (i, u, r_{ui}). Редьюсеры предыдущего шага выводят $O(\zeta I^2)$, где $\zeta-$ количество ненулевых $sim(i, j)$, поэтому сложность по времени $O(\frac{(\alpha I)^2+\alpha UI)}{M}$. Сложность по памяти $O(I)$.

4.2 редьюсер

Редьюсер на данном шаге собирает все необходимое для предсказания рейтингов по формуле (2). Собрав все выход маппера по ключу i, редьюсер выдает $((u, i), (sim(i, j), (r_{uj})))$ для всех j. Сложность по времени $O(\frac{(\zeta I)^2+\alpha UI)}{R}$. В памяти редьюсер хранит все что касается идентификатора i, поэтому сложность по памяти – $O(\frac{\gamma U+\beta I}{R})$

5 Шаг 4

5.1 маппер

Маппер, помимо выхода предыдущего шага, получает на вход снова ratings.csv. Выходы предыдущего шага остаются без изменений, строки ratings.csv преобразуются в строки с ключом (u, i). Сложность по времени – $O(\alpha\frac{I^3}{M})$, сложность по памяти $O(1)$.

5.2 редьюсер

Редьюсер данного шага считает оценки $\hat{r}_{ui}$ параллельно отсеивая пары (u, i), для которых уже известны оценки. Для этого в процессе агрегации значений по ключу (u, i), если редьюсеру попадается строка из файла ratings.csv, пара (u, i) признается ненужной и все поступающие значения по данному ключу игнорируются. Для остальных ключей подсчитывается $\hat{r}_{ui}$ по формуле (2) и выдается $(u, \hat{r}_{ui}, i)$. Сложность по времени $O(\alpha\frac{I^3+UI}{R})$, сложность по памяти $O(\beta\frac{I}{R})$.

6 Шаг 5

6.1 маппер

Маппер преобразуют тройки $(u, \hat{r}_{ui}, i)$ в $(u, \hat{r}_{ui}, title_i)$, где $title_i$– название фильма с идентификатором i. Для этого маппер считывает файл movies.csv (так как он имеет небольшой размер), и ставит каждому поступающему значению i его название. Выход редьюсеров предыдущего шага имеет сложность $O(UI)$, поэтому сложность по времени маппера – $O(\frac{UI}{M})$

6.2 редьюсер

На выходе редьюсера мы должны получить для каждого пользователя топ 100 фильмов с наивысшей предсказанной оценкой. Для этого, перед тем как подать данные редьюсеру, проведем сортировку для всех элементов $(u, \hat{r}_{ui}, title_i)$. Для $u, \hat{r}_{ui}$ сортировка численная (для $\hat{r}_{ui}$ сортировка по убыванию), а для $title_i$– лексикографическая. Далее, редьюсер для каждого пользователя u отбирает и выводит первые 100 фильмов. Сложность по времени $O(\frac{UI}{R})$. Сложность по памяти константная – $O(1)$, так как редьюсеры не хранят больше 100 значений фильмов. Сложность выхода составляет $O(U)$, снова из-за того, что для каждого пользователя мы выводим только 100 фильмов с наивысшей оценкой.

Общее время работы MapReduce задачи $\approx$ 40 минут.