Коллаборативная фильтрация на MapReduce

Выполнил: Курцев Дмитрий, группа 317

Решение состоит из 7 Map-reduce стадий. Изначально была сделана предобработка двух файлов movies.csv и ratings.csv, чтобы избавиться от ненужных записей.

1 стадия

Map. На вход первому мапперу приходит файл с рейтингами. Он разбивает каждую строчку на ключ-значение, где key - ID пользователя, value - {ID фильма, рейтинг}. Таким образом, сложность данной стадии $O(1)$.
Reduce. На вход получает все фильмы, которые посмотрел каждый из пользователей. Выходом являются всевозможные пары фильмов (i, j)(ключи) и пары разностей рейтингов этих фильмов и среднего рейтинга пользователя (числитель функции sim) $(r_{u,i} - \overline{r_u}, r_{u,j} - \overline{r_u})$(значения), для каждого пользователя. Таким образом необходимая память $O(\alpha I)$, а время $O((\alpha I)^2)$ отдельно для каждого пользователя. Всего памяти $O(\alpha IU)$, времени $O((\alpha I)^2 U)$.

2 стадия

Map. Ничего не делает, прокидывает дальше строку. $O(1)$.
Reduce. Происходит вычисление $sim(i, j)$. По памяти и времени требует информации по всем пользователям, посмотревших фильм i и фильм j. Таким образом, отдельно для каждой пары сложность по памяти и времени $O(\alpha U)$. Всего $O(\alpha U I^2)$.

3 стадия

Map. На вход поступает выход редьюсера со второго этапа и файл с рейтингами. Выход прошлого редьюсера помечается флагом *'S'* (sim). Он разбивается на ключ-значение, где key - ID одного из фильмов, value - {ID второго фильма, sim}. И так дважды для обоих айтемов. Файл с рейтингами помечается флагом *'R'* (rating). Печатает ключ - ID фильма, значение - {ID пользователя, рейтинг}. Сложность $O(1)$.
Reduce. Для каждого фильма i берёт всех пользователей, которые посмотрели этот фильм с их рейтингами $r_{u,i}$. Печатает ключ - {ID пользователя u, ID всех остальных фильмов j}, значение $\{r_{u,i},\ \ sim(i,j)\}$. Сложность по памяти $O(\alpha U + I)$, по времени $O(\alpha UI)$.

4 стадия

Map. Ничего не делает, прокидывает дальше строку. $O(1)$.
Reduce Вычисляет $\hat{r}_{u,i}$. Для этого он хранит все рейтинги всех фильмов j, которые посмотрел пользователь u и $sim(i,j)$. Печатает ключ - {ID пользователя, ID фильма}, значение - рейтинг. Таким образом, получим, что сложность по памяти и времени $O(\alpha I)$ для отдельной пары (u,i).

5 стадия

Map. На вход поступает выход редьюсера с четвёртого этапа и файл с рейтингами. Выход прошлого редьюсера печатается далее. Работа с файлом с рейтингами аналогична первому мапперу. Печатает ключ - ID пользователя, значение - {ID фильма, рейтинг}. Сложность $O(1)$.
Reduce. Оставляет топ-100 фильмов с самым высоким предсказанным рейтингом, которые пользователь ещё не посмотрел. Для этого происходит сортировка по всем фильмам. Так получим по па-

мяти сложность $O((1-\alpha)I)$, по времени $O(((1-\alpha)I)\log[(1-\alpha)I])$ для каждого пользователя.

6 стадия

Map. На вход поступает выход редьюсера с пятого этапа и файл с названиями фильмов. Выход прошлого редьюсера печатается далее с флагом *'R'* (rating). Файл с названиями помечается, как *'N'* (name). Печатает ключ - ID фильма, значение - имя фильма. Сложность $O(1)$.
Reduce. Каждому ID фильма сопоставляет его имя. Печатает ключ - ID пользователя, значение - {имя фильма, предсказанный рейтинг}. Сложность по памяти и времени $O(U)$ для каждого фильма.

7 стадия

Map. Ничего не делает, прокидывает дальше строку. $O(1)$.
Reduce. Печатает итоговый ответ. Для каждого пользователя 100 фильмов, поэтому сложность $O(1)$.