Hệ gợi ý

Nguyễn Tuấn Đạt
Đặng Quang Trung

Ngày 9 tháng 1 năm 2017

Nội dung

Mô tả dữ liệu

Giới thiệu bộ dữ liệu

- Bộ dữ liệu: ml-20m (size: 190MB).
- Download: `http://grouplens.org/datasets/movielens/`.
- Bộ dữ liệu mô tả đánh giá 1 - 5 sao phim từ MovieLens.
- Bộ dữ liệu chứa:
  - 20000263 rating và 465564 tag của 27278 bộ phim.
  - Dữ liệu đươc tạo bởi 138493 users.
- Các users được thu thập ngẫu nhiên. Tất cả users đã đánh giá ít nhất 20 bộ phim.
- Các file dữ liệu chứa trong 6 tập, genome-scores.csv, genome-tags.csv, links.csv, movies.csv, ratings.csv và tags.csv.
  - Các file đã sử dụng: movies.csv và ratings.csv.

Mô tả dữ liệu

- Cấu trúc file dữ liệu ratings:
  - Tất cả đánh giá của người dùng điều chứa trong file ratings.csv
  - Mỗi dòng của file sau dòng header có định dạng (userId,movieId,rating,timestamp).
  - Các rating thực hiện trên thang điểm 5 sao, với gia số (0.5 sao - 5.0 sao).
- Cấu trúc file dữ liệu Movies:
  - Thông tin của các movies chứa trong file moives.csv.
  - Mỗi dòng sau dòng header có định dạng (movieId,title,genres).
  - Genres là danh sách thể loại được lựa chọn: Action, Adventure, Animation , Children's, Comedy, Crime, Documentary, Drama, Fantasy, Film-Noir, Horror, Musical, Mystery, Romance, Sci-Fi, Thriller, War, Western, (no genres listed)

Xử lý dữ liệu

- Sử dụng Java để xử lý dữ liệu ban đầu(chuẩn hóa dữ liệu). Các file sử dụng gồm có: ratings.csv, movies.csv.
- File ratings.csv
  - Đưa ra file biểu diễn theo định dạng (userId,movieId,rating).
  - File biểu diễn mà trận rating của các user.
- File movies.csv
  - Đưa ra file dưới dạng ma trận.
  - Hàng biểu diễn cho movie và cột biểu diễn cho danh sách các đặc tính.

$$(i,j) = \begin{cases} 1 & \text{nếu movie i có đặc tính j} \\ 0 & \text{nếu movie i ko đặc tính j} \end{cases}$$

Collaborative Filtering

Ý tưởng :

- Bước 1: Xét người dùng cần gợi ý phim x. Ta tìm tập N người dùng có tập đánh giá phim tương đồng với người dùng x.
- Bước 2: Ước lượng đánh giá của người dùng x với những phim mà anh ấy chưa xem bằng cách dựa vào tập N của x. Sau đó, ta đưa ra t phim có ước lượng cao nhất để gợi ý xem cho người dùng x.

Tìm kiếm tập người dùng tương đồng

Xét một phim i nào đó: Tính độ tương đồng giữa người dùng x và người dùng y bằng độ đo cosin:

$$sim(x, y) = cos(\overrightarrow{r_x}, \overrightarrow{r_y}) = \frac{\overrightarrow{r_x} \bullet \overrightarrow{r_y}}{||\overrightarrow{r_x}||||\overrightarrow{r_y}||}$$

Với mỗi người dùng ta sẽ chọn ra k người dùng gần với x nhất có đánh giá cho phim i.

Ước lượng với những phim chưa đánh giá

Xét người dùng x và bộ phim i $r_x i$ sẽ được ước lượng bằng công thức:

$$r_{xi} = \frac{\sum_{j \in N(x)} S_{x,j} * r_{ji}}{\sum_{j \in N(x)} S_{x,j}}$$

với : S_{xj} là độ tương đồng của người dùng x và người dùng j.
Ước lượng với trung bình trọng số:

$$r_{xi} = b_{xi} + \frac{\sum_{j \in N(i;x)} S_{i,j} \bullet (r_{xj} - b_{xj}}{\sum_{j \in N(i;x)} S_{ij}}$$

Với

$$b_{xi} = \mu + b_x + b_i$$

- μ là trung bình độ lệch
- b_x trung bình độ lệch theo người dùng x
- b_i trung bình độ lệch theo phim i

Latent Factor Model

Ý tưởng :

Sử dụng SVD để giảm số chiều của dữ liệu.

$$R = P * Q^T$$

với Q(item,factor) và P(user,factor);
Ước lượng được tính bằng công thức

$$r_{xi} = q_i * p_x$$

Hàm đánh giá

$$min_{P,Q} \sum_{i,j \in R} = (R_{ij} - q_j * p_i)^2$$

Để mô hình đúng hơn với S dữ liệu đã bị mất ta thêm vào hàm đánh giá các tham số để mong muốn có kết quả tốt hơn trong việc ước lượng các đánh giá.

$$min_{P,Q} \sum_{i,j \in R} = (R_{ij} - q_j * p_i)^2 + [\lambda_1 \sum_i ||p_i||^2 + \lambda_1 \sum_j ||q_j||^2]$$

Ta sẽ sử dụng SGD để tối thiểu hàm đánh giá: Ta thu được:
Với mỗi r_{xi} :

- $\varepsilon_{xi} = 2(r_{xi} - q_i * p_x)$
- $q_i = q_i + \mu_1(\varepsilon_{xi} * p_x - \lambda_2 q_i)$
- $p_x = p_x + \mu_1(\varepsilon_{xi} * q_i - \lambda_2 p_x)$

với $\mu_{1,2}$ là tốc độ học

Content-based

Ý tưởng: Giới thiệu các movie đến user x với các movie có đặc tính gần với các movie mà user x đã đánh giá cao trước đó.

- Mỗi người dùng x sẽ có một vector người dùng.

Movie

user		0	1	2	3	4	5
	x	3.5	4.0	5.0	0	2.0	4.5

Hình: vector người dùng

- Chọn ngưỡng đánh giá rating của người dùng α

$$(x, i) = \begin{cases} 1 & \text{nếu rating(x,i)} \geq \alpha \\ -1 & \text{nếu rating(x,i)} \leq \alpha \end{cases}$$

Content-based

- $\alpha = 4.0$ Ta có vector người dùng

Movie

		0	1	2	3	4	5
user	x	-1	1	1	-1	-1	1

Hình: vector người dùng sau khi đánh giá

- Tìm đặc tính của user ta thực hiện:
  - $V_{feature} = V_{user} * M_{movie}$.
    Trong đó: $V_{feature}$ vector đặc tính người dùng, V_{user} vector người dùng, M_{movie} ma trận movie và các đặc tính.

Content-based

- Chuẩn hóa lại $V_{feature}$

$$V_{feature}(i) = \begin{cases} 1 & \text{nếu } V_{feature}(i) > 0 \\ 0 & \text{nếu } V_{feature}(i) < 0 \end{cases}$$

- Sử dụng độ đo Cosin để tình khoảng cách giữa vector đặc tính của người dùng và movie
- Gợi ý những movie gần với vector đặc tính của người dùng.

RMSE

Sử dụng tiêu chuẩn ước lượng:

Root Mean Square Error (RMSE): $\sqrt{\frac{1}{|R|}\sum_{(i,x)\in R}(\hat{r_{xi}} - r_{xi})^2}$

Ước lượng trên chỉ áp dụng được với Collaborative Filtering và Latent Factor Model, Content-base không thể đánh giá theo phương pháp này.

Thực hiện

- Sử dụng Java để tạo ma trận dữ liệu.
- Sử dụng nén thưa(thư viện Java, Matlab) để nhét toàn bộ tập dữ liệu vào bộ nhớ.
- Tách lấy 1000 rating làm tập test phần còn lại làm tập train.

Collaborative Filtering

- Chọn k=5,10 phần tử có độ tương đồng gần nhất.
- Kết hợp với trung bình trọng số để đoán các rating trong tập test.

Latent Factor Model

- Chọn số factor=30,50,100 (30).
- Lựa chọn các tham số $\mu \approx 0.0002(0.0001)$.
- Chọn $\lambda \approx 0.1(0.02)$.
- Sử dụng Stochastic Gradient Descent.

Kết quả

Dự đoán trên 1000 tập rating ngẫu nhiên

- Collaborative Filtering(k=5) : 1.13
- Latent Factor Model(Matlab SVD: 1.4406)
- Latent Factor Model($\lambda_1 = \lambda_2 = 0.002, SGD$) : 3.5(cũ) ?? (update 1.1361)

Cảm ơn thầy và các bạn đã lắng nghe

Tài liệu tham khảo

- Slide Datamining Stanford