DATA SCIENCE

BÁO CÁO ĐỒ ÁN

Giáo viên:
Trần Trung Kiên

Nhóm báo cáo:
Tiêu Thanh Sơn - 1412467
Võ Quốc Cường - 1412071

Đề tài: gợi ý phim

Nội dung

1. Phát biểu bài toán
2. Thu thập dữ liệu
   a. Quá trình thu thập
   b. Mô tả dữ liệu
   c. Giải quyết vấn đề giá trị thiếu, sai
3. Giải quyết bài toán
4. Tham khảo

1. Phát biểu bài toán

- Gợi ý phim cho người dùng sử dụng dữ liệu trên imdb.com
- Dữ liệu training là cặp dữ liệu:
  - Movies.
  - Ratings

1. Phát biểu bài toán:

- Ứng dụng của hệ thống gợi ý phim

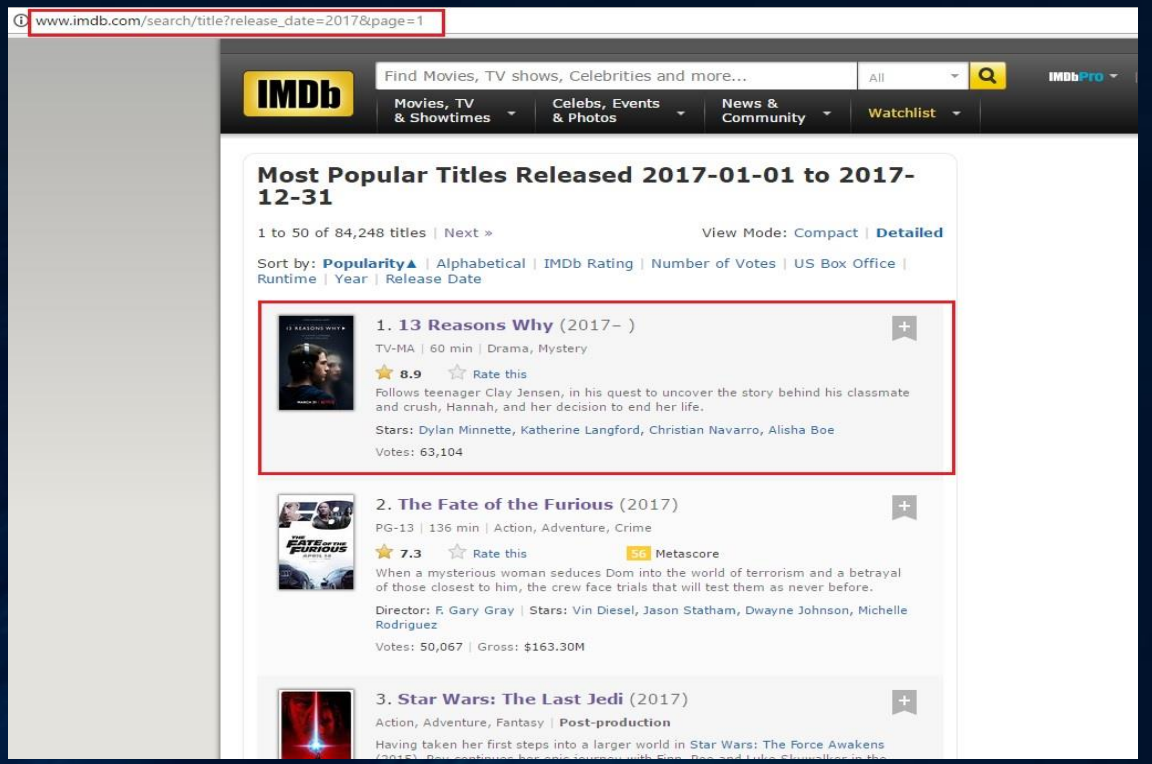
2. Thu thập dữ liệu

- Tập dữ liệu movies:
  - Dựa trên Advance Search của imdb.com dạng: http://www.imdb.com/search/title?release_date=[%year%]&page=[%page%]

2. Thu thập dữ liệu

a. Thu thập

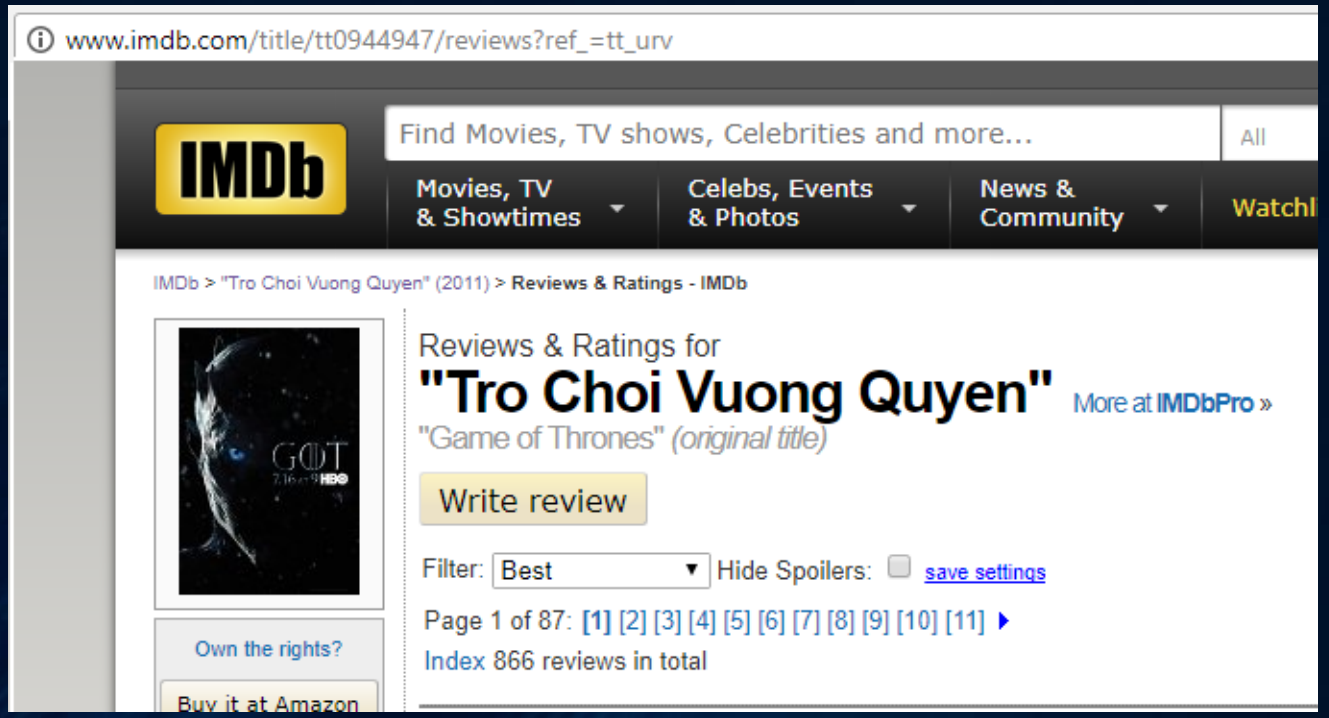
- Tập dữ liệu movies:



2. Thu thập dữ liệu

a. Thu thập

- Tập dữ liệu ratings: vào phần rating của movie để thu thập



2. Thu thập dữ liệu

b. Mô tả dữ liệu

- Dữ liệu movies:

movies.csv

```
,movie_id,title,genres
0,1,Toy Story (1995),Animation|Children's|Comedy
1,2,Jumanji (1995),Adventure|Children's|Fantasy
2,3,Grumpier Old Men (1995),Comedy|Romance
3,4,Waiting to Exhale (1995),Comedy|Drama
4,5,Father of the Bride Part II (1995),Comedy
5,6,Heat (1995),Action|Crime|Thriller
6,7,Sabrina (1995),Comedy|Romance
7,8,Tom and Huck (1995),Adventure|Children's
8,9,Sudden Death (1995),Action
9,10,GoldenEye (1995),Action|Adventure|Thriller
```

2. Thu thập dữ liệu

b. Mô tả dữ liệu

- Dữ liệu ratings:

ratings.csv

```
,user_id,movie_id,rating
0,1,1177,5
1,1,656,3
2,1,903,3
3,1,3340,4
4,1,2287,5
```

3. Tiền xử lý

- Dữ liệu movies:
  - Đối với dữ liệu rời rạc ta có thể chuyển thành dạng one hot (coi mỗi giá trị rời rạc là một cột mang giá trị nhị phân).

3. Tiền xử lý

- Dữ liệu ratings:
  - Chuyển đổi movie id và user id về dạng số thự tự để tiện trong quá trình học

4. Máy học

a. Mô hình content-based:

- Content-based là gì?
  - Là mô hình gợi ý đánh giá đặc tính của *items* để gợi ý.

4. Máy học

a. Mô hình content-based:

- Các bước xây dựng thuật toán
  - U, M lần lượt là số lượng user và movie.
  - Ma trận rating.

	u_0	u_1	u_2	u_3	u_4	u_5	u_6
i_0	5	5	2	0	1	?	?
i_1	3	?	?	0	?	?	?
i_2	?	4	1	?	?	1	2
i_3	2	2	3	4	4	?	4
i_4	2	0	4	?	?	?	5

4. Máy học

a. Mô hình content-based:

- Các bước xây dựng thuật toán
  - Feature vector của movie m1:

    x1 = [0.11, 0.89]
  - Xác định vector w cho mỗi user.
  - Dự đoán rating:

$$y_{i,j} = (w_j)^T \times x_i \quad (1)$$

	u1	u2	u3	u4	u5	u6	f1	f2
m1	?	4	5	2	1	?	0.11	0.89
m2	1	?	1	4	3	5	0.62	0.38
m3	4	5	2	?	?	4	0.21	0.79
m4	5	4	?	?	1	1	0.15	0.85
m5	?	3	4	5	3	1	0.2	0.8
	w_1	w_2	w_3	w_4	w_5	w_6		

4. Máy học

a. Mô hình content-based:

- Các bước xây dựng thuật toán
  - Đưa về bài toán hồi quy hoặc phân lớp.
  - Sử dụng mô hình Ridge regression.

4. Máy học

a. Mô hình content-based:

❑ Các bước xây dựng thuật toán

- Công thức dự đoán rating:

$$y_{i,j} = (w_j)^T \times x_i \quad (1)$$

- Hàm chi phí:

$$L_j = \frac{1}{2S_j} \sum_{i:r_{i,j}=1} ((w_j)^T \times x_i - y_{i,j})^2 + \frac{\lambda}{2S_j} \|w_j\|_2^2 \quad (2)$$

- S_j là số lượng movie đã được đánh giá bởi user j.
- $r_{i,j} = 1$ khi user j đã rated cho movie i.

4. Máy học

a. Mô hình content-based:

❑ Các bước xây dựng thuật toán

- Vì hàm này chỉ phụ thuộc vào các movie đã rated. Nên ta có thể rút gọn:

$$L_j = \frac{1}{2S_j} \left\| (w_j)^T \times \hat{X}_j - \hat{y}_j \right\|_2^2 + \frac{\lambda}{2S_j} \left\| w_j \right\|_2^2 \quad (3)$$

- $\hat{X}_j$ là ma trận con của ma trận feature X, chứa các vector feature của những bộ phim đã rated bởi user j.

4. Máy học

a. Mô hình content-based:

❑ Kết quả:

- Ảnh hưởng của việc chuẩn hóa ma trận movie feature đến quá trình học.

- Nhận xét: chuẩn hóa ma trận movie cho kết quả học tốt hơn.

In [17]:
```
# TH1: không chuẩn hóa movie feature, learning_rate=0.1
W_1 = train_ridge_regression(train_ratings, X_nn, n_users)
Y_pred_1 = X_nn.dot(W_1)
# tính độ lỗi
train_err_1 = RMSE(train_ratings, Y_pred_1)
vali_err_1 = RMSE(vali_ratings, Y_pred_1)
print 'RMSE for train: ', train_err_1
print 'RMSE for validation:  ', vali_err_1
```

```
RMSE for train:  0.908151624472
RMSE for validation:   1.07518261218
```

In [18]:
```
# TH2: chuẩn hóa movie feature, learning_rate=0.1
W_2 = train_ridge_regression(train_ratings, X_n, n_users)
Y_pred_2 = X_n.dot(W_2)
# tính độ lỗi
train_err_2 = RMSE(train_ratings, Y_pred_2)
vali_err_2 = RMSE(vali_ratings, Y_pred_2)
print 'RMSE for train: ', train_err_2
print 'RMSE for validation:  ', vali_err_2
```

```
RMSE for train:  0.908769544536
RMSE for validation:   1.05801430628
```

4. Máy học

a. Mô hình content-based:

❑ Kết quả:

- Ảnh hưởng của learning_rate đến quá trình học.

In [20]:
```
# learning_rate=0.1
W_4 = train_ridge_regression(train_ratings, X_n, n_users, learning_rate=0.1)
Y_pred_4 = X_n.dot(W_4)
# tính độ lỗi
train_err_4 = RMSE(train_ratings, Y_pred_4)
vali_err_4 = RMSE(vali_ratings, Y_pred_4)
print 'RMSE for train: ', train_err_4
print 'RMSE for validation:  ', vali_err_4
```

```
RMSE for train:  0.908769544536
RMSE for validation:   1.05801430628
```

In [19]:
```
# learning_rate=0.5
W_3 = train_ridge_regression(train_ratings, X_n, n_users, learning_rate=0.5)
Y_pred_3 = X_n.dot(W_3)
# tính độ lỗi
train_err_3 = RMSE(train_ratings, Y_pred_3)
vali_err_3 = RMSE(vali_ratings, Y_pred_3)
print 'RMSE for train: ', train_err_3
print 'RMSE for validation:  ', vali_err_3
```

```
RMSE for train:  0.919323398701
RMSE for validation:   1.0346479408
```

4. Máy học

a. Mô hình content-based:

❑ Kết quả:

- Ảnh hưởng của learning_rate đến quá trình học.

```
In [21]: # learning_rate=0.01
         W_5 = train_ridge_regression(train_ratings, X_n, n_users, learning_rate=0.05)
         Y_pred_5 = X_n.dot(W_5)
         # tính độ lỗi
         train_err_5 = RMSE(train_ratings, Y_pred_5)
         vali_err_5 = RMSE(vali_ratings, Y_pred_5)
         print 'RMSE for train: ', train_err_5
         print 'RMSE for validation:  ', vali_err_5

         RMSE for train:  0.907087641243
         RMSE for validation:   1.06810553779
```

- Nhận xét: khi giảm *learning_rate* thì càng bị *overfitting*. Ở thử nghiệm trên, *learning_rate*=0.5 cho kết quả tốt nhất trên *validation.*

4. Máy học

a. Mô hình content-based:

❑ Kết quả:

- Sử dụng kết quả dự đoán của tham số tốt nhất (chuẩn hóa dữ liệu movie và learning_rate=0.5) chạy trên tập test:

```
In [22]: test_err = RMSE(test_ratings, Y_pred_3)
         print 'RMSE for test: ', test_err

         RMSE for test:  1.03009245862
```

4. Máy học

a. Mô hình content-based:

- Các bước xây dựng thuật toán
  - Kiểm tra lỗi: sử dụng Root Mean Squared Error (RMSE).

4. Máy học

b. Mô hình Matrix Factorization

- Đầu tiên cần hiểu: Collaborative filtering là gì?
  - Là hệ thống gợi ý items dựa trên sự tương quan (similarity) giữa các users và/hoặc items.
  - Về cơ bản, hệ thống lọc cộng tác khắc phục được những hạn chế của content-based và thường cho hiệu quả tốt hơn.
  - Gồm 2 phương pháp: Neighborhood-based Collaborative Filtering (NBCF) và Matrix Factorization Collaborative Filtering.

4. Máy học

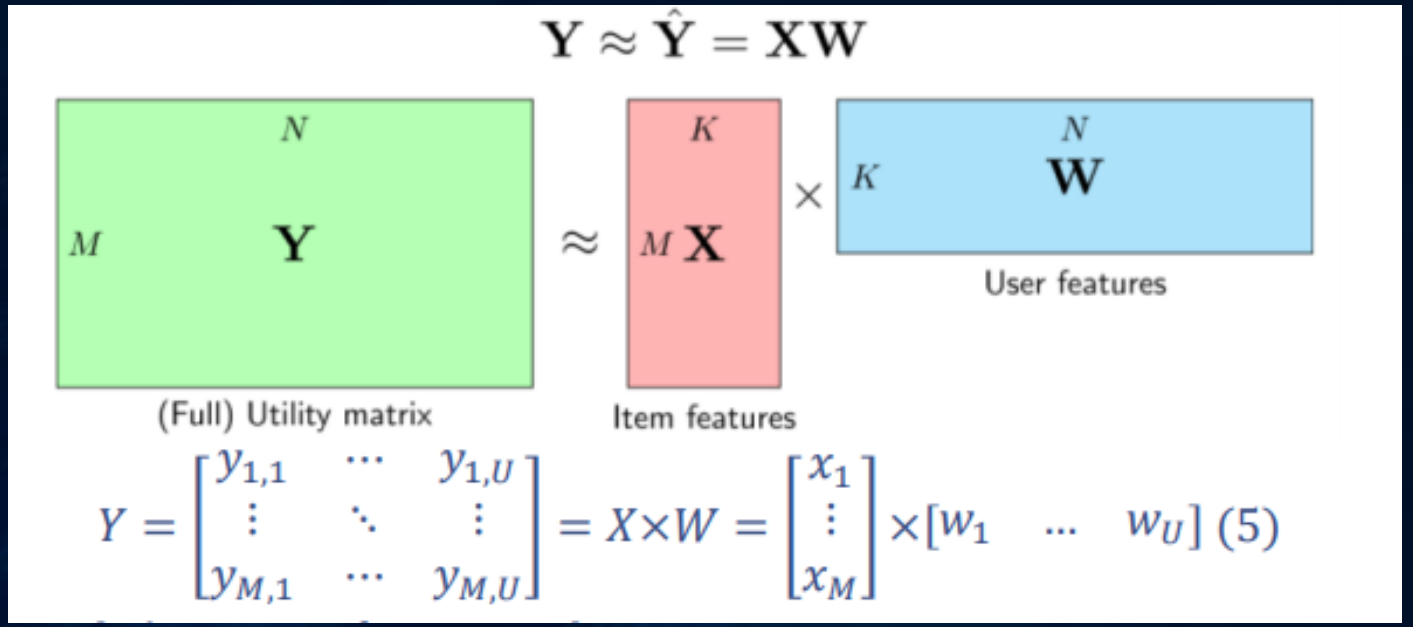
b. Mô hình Matrix Factorization

- Các bước xây dựng thuật toán
  - Tương tự như content-based, ma trận rating Y được xấp xỉ: Y = X x W

4. Máy học

b. Mô hình Matrix Factorization

- Khác biệt so với content-based?
- Huấn luyện đồng thời X và W dựa vào dữ liệu ratings.



4. Máy học

b. Mô hình Matrix Factorization

- X là ma trận movie feature (M x K).
- W là ma trận user feature (K x U)
- K là gì? Là số thuộc tính ẩn mô tả sự kết nối giữa user và item.

4. Máy học

b. Mô hình Matrix Factorization

- Chuẩn hóa ma trận rating, có 2 cách: theo user hoặc theo item

	u_0	u_1	u_2	u_3	u_4	u_5	u_6
i_0	5	5	2	0	1	?	?
i_1	4	?	?	0	?	2	?
i_2	?	4	1	?	?	1	1
i_3	2	2	3	4	4	?	4
i_4	2	0	4	?	?	?	5
$\bar{u}_j$	3.25	2.75	2.5	1.33	2.5	1.5	3.33

	u_0	u_1	u_2	u_3	u_4	u_5	u_6
i_0	1.75	2.25	-0.5	-1.33	-1.5	0	0
i_1	0.75	0	0	-1.33	0	0.5	0
i_2	0	1.25	-1.5	0	0	-0.5	-2.33
i_3	-1.25	-0.75	0.5	2.67	1.5	0	0.67
i_4	-1.25	-2.75	1.5	0	0	0	1.67

	u_0	u_1	u_2	u_3	u_4	u_5	u_6	
i_0	5	5	2	0	1	?	?	2.6
i_1	4	?	?	0	?	2	?	2
i_2	?	4	1	?	?	1	1	1.75
i_3	2	2	3	4	4	?	4	3.17
i_4	2	0	4	?	?	?	5	2.75

	u_0	u_1	u_2	u_3	u_4	u_5	u_6
i_0	2.4	2.4	-.6	-2.6	-1.6	0	0
i_1	2	0	0	-2	0	0	0
i_2	0	2.25	-0.75	0	0	-0.75	-0.75
i_3	-1.17	-1.17	-0.17	0.83	0.83	0	0.83
i_4	-0.75	-2.75	1.25	0	0	0	2.25

4. Máy học

b. Mô hình Matrix Factorization

- Xây dựng hàm mất mát:

$$L(X,W) = \frac{1}{2S}\sum_{u=1}^{U}\sum_{m:r_{m,u}=1}\left(x_m \times w_u - y_{m,n}\right)^2 + \frac{\lambda}{2}(\|X\|_F^2 + \|W\|_F^2) \quad (7)$$

- Trong đó: S là tổng số ratings.
- $r_{m,u} = 1$ khi user u đã rated cho movie m.

4. Máy học

b. Mô hình Matrix Factorization

- Sử dụng Gradient Descent để tối ưu X và W
- Để tối ưu W ta cố định X và ngược lại.
- Khi cố đinh X:

$$L(W) = \frac{1}{2S}\sum_{u=1}^{U}\sum_{m:r_{m,u}=1}\left(x_m \times w_u - y_{m,n}\right)^2 + \frac{\lambda}{2}\|W\|_F^2 (8)$$

- Xét user u:

$$L(w_u) = \frac{1}{2S}\sum_{m:r_{m,u}=1}\left(x_m \times w_u - y_{m,n}\right)^2 + \frac{\lambda}{2}\|w_u\|_2^2 (9)$$

4. Máy học

b. Mô hình Matrix Factorization

- Rút gọn:

$$L(w_u) = \frac{1}{2S} \left\| \hat{X}_u \times w_u - \hat{y}_u \right\|_2^2 + \frac{\lambda}{2} \|w_u\|_2^2 \quad (10)$$

- Với $\hat{X}_u$ là sub matrix của X: là ma trận được tạo bởi các hàng của X ứng với các items đã được rated bởi user u và $\hat{y}_u$ là các ratings tương ứng.

4. Máy học

b. Mô hình Matrix Factorization

- Đạo hàm:

$$\frac{\partial L(w_u)}{\partial w_u} = \frac{1}{s}\hat{X}_u{}^T\left(\hat{X}_u \times w_u - \hat{y}_u\right) + \lambda w_u \quad (11)$$

- Cập nhật:

$$w_u -= \eta\left(\frac{1}{s}\hat{X}_u{}^T\left(\hat{X}_u \times w_u - \hat{y}_u\right) + \lambda w_u \quad (12)\right)$$

4. Máy học

b. Mô hình Matrix Factorization

- Tương tự với ma trận X:
- Cập nhật x_m:

$$x_m -= \eta(\frac{1}{s}(x_m \times \widehat{W}_m - \hat{y}^m) \times \widehat{W}_m{}^T + \lambda x_m \quad (17)$$

- $\widehat{W}_m$ là ma trận được tạo bằng các cột của W ứng với các users đã đánh giá item đó và $\hat{y}^m$ là vector ratings tương ứng.

4. Máy học

b. Mô hình Matrix Factorization

- Một số vấn đề tối ưu?
  - Hạn chế vòng lặp.
  - Overfitting.
  - Lựa chọn các tham số learning_rate, lamda phù hợp.

4. Máy học

b. Mô hình Matrix Factorization

- Kiểm tra lỗi: sử dụng Root Mean Squared Error.

4. Máy học

b. Mô hình Matrix Factorization

❑ Kết quả:

- Ảnh hưởng của chuẩn hóa ma trận rating theo user và movie
  - Cố định các giá trị *K = 10, max_patience=None, learning_rate=0.5, lamda=0.1, max_epoch=100.*

```
# user-based
# n_users và n_movies đã được tính từ lúc trước khi chia dữ liệu
X_u, W_u, avg_rating_u = train(train_ratings, vali_ratings, n_users, n_movies, user_based=1)
```

```
Info of returned: epoch  99  train RMSE:  1.02617194562 validation RMSE: 1.03934541922
```

```
# movie-based
X_m, W_m, avg_rating_m = train(train_ratings, vali_ratings, n_users, n_movies, user_based=0)
```

```
Info of returned: epoch  99  train RMSE:  0.973792050828 validation RMSE: 0.9821349886
```

  - Nhận xét: chuẩn hóa dữ liệu *theo movie-based* cho kết quả tốt hơn. Ở các thử nghiệm tiếp theo, ta sẽ mặc định sử dụng chuẩn hóa *movie-based.*

4. Máy học

b. Mô hình Matrix Factorization

❑ Kết quả:

- Ảnh hưởng của K đến quá trình học
  - Cố định các giá trị *max_patience=None, learning_rate=0.5, lamda=0.1, max_epoch=100.* Lần lượt thử *K = 5, 30, 50.*

```
# K = 5
X_1, W_1, avg_rating_1 = train(train_ratings, vali_ratings, n_users, n_movies, K = 5)
```

```
Info of returned: epoch  99  train RMSE:  0.974129980177 validation RMSE: 0.979579511178
```

```
# K = 30
X_2, W_2, avg_rating_2 = train(train_ratings, vali_ratings, n_users, n_movies, K = 30)
```

```
Info of returned: epoch  99  train RMSE:  0.974129860668 validation RMSE: 0.979579912528
```

```
# K = 50
X_3, W_3, avg_rating_3 = train(train_ratings, vali_ratings, n_users, n_movies, K = 50)
```

```
Info of returned: epoch  99  train RMSE:  0.974130272562 validation RMSE: 0.979579868461
```

- Nhận xét: K lớn có thể gây *overfitting*. Giá trị *K=30* cho độ lỗi nhỏ nhất trên *validation*. Ta sẽ dùng giá trị này để huấn luyện và kiểm tra trên *test*.

4. Máy học

b. Mô hình Matrix Factorization

❑ Kết quả:

- Ảnh hưởng của learning rate
  - Cố định các giá trị *max_patience=None, lamda=0.1, max_epoch=100.* Lần lượt thử *learning_rate = 0.5, 0.1, 0.01.*

```
# learning_rate=0.5
X_4, W_4, avg_rating_4 = train(train_ratings, vali_ratings, n_users, n_movies, learning_rate=0.5)
```

```
Info of returned: epoch  99  train RMSE:  0.974130296693 validation RMSE: 0.979579868751
```

```
# learning_rate=0.1
X_5, W_5, avg_rating_5 = train(train_ratings, vali_ratings, n_users, n_movies, learning_rate=0.1)
```

```
Info of returned: epoch  99  train RMSE:  1.06025183522 validation RMSE: 1.06561020565
```

```
# learning_rate=0.01
X_6, W_6, avg_rating_6 = train(train_ratings, vali_ratings, n_users, n_movies, learning_rate=0.01)
```

```
Info of returned: epoch  99  train RMSE:  1.51415658309 validation RMSE: 1.51879106153
```

  - Nhận xét: learning càng giảm thì độ lỗi trên train và validation càng tăng. Giá trị learning_rate=0.5 cho kết quả tốt nhất, ta sẽ sử dụng giá trị này để huấn luyện và kiểm tra trên test.

4. Máy học

b. Mô hình Matrix Factorization

❑ Kết quả:

- Ảnh hưởng của weight decay đến quá trình học
  - Cố định các giá trị *K = 10, max_patience=None, learning_rate=0.5, max_epoch=100.* Lần lượt thử *lamda=0.0, 0.1, 0.5.*

```
# lamda=0.0
X_7, W_7, avg_rating_7 = train(train_ratings, vali_ratings, n_users, n_movies, lamda=0.0)
```

```
Info of returned: epoch  99  train RMSE:  3.14772845478 validation RMSE: 3.1572188132
```

```
# lamda=0.1
X_8, W_8, avg_rating_8 = train(train_ratings, vali_ratings, n_users, n_movies, lamda=0.1)
```

```
Info of returned: epoch  99  train RMSE:  0.974130194555 validation RMSE: 0.979580354061
```

```
# lamda=0.5
X_9, W_9, avg_rating_9 = train(train_ratings, vali_ratings, n_users, n_movies, lamda=0.5)
```

```
Info of returned: epoch  99  train RMSE:  0.97413010318 validation RMSE: 0.979579876099
```

  - Nhận xét: khi sử dụng *lamda=0.1*, kết quả tốt hơn nhiều so với khi *lamda=0*.

4. Máy học

b. Mô hình Matrix Factorization

- Kết quả:
  - Kết hợp stop early, huấn luyện và kiểm tra trên test:
    - Huấn luyện với *K = 30, max_patience=50, learning_rate=0.5, lamda=0.1, max_epoch=200.*
    - *Kết quả:*

```
X, W, avg_rating = train(train_ratings, vali_ratings, n_users, n_movies, K = 30,
                         max_patience=50, learning_rate=0.5, lamda=0.1, max_epoch=200)
```

```
Info of returned: epoch  77  train RMSE:  0.962130172264 validation RMSE: 0.967280623471
```

```
test_err = RMSE(test_ratings, X, W, avg_rating)
print "Test RMSE: ", test_err
```

```
Test RMSE:  0.966832473621
```

5. Tham khảo:

https://gallery.cortanaintelligence.com/Experiment/Recommender-Movie-recommendation-3

https://www.slideshare.net/xamat/recommender-systems-machine-learning-summer-school-2014-cmu

https://hal.archives-ouvertes.fr/inria-00580523/document

https://pdfs.semanticscholar.org/7e98/a98bbc25ab2e4e425d802b3e48257984435e.pdf

http://herbrete.vvv.enseirb-matmeca.fr/IR/CF_Recsys_Survey.pdf

THANK FOR LISTENING