Movie Recommender System

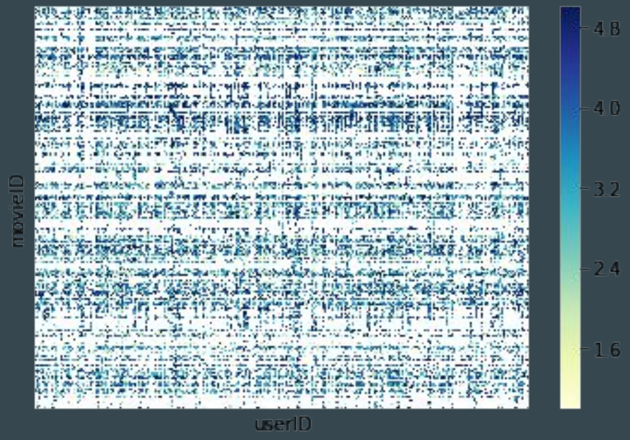
MovieLens Dataset

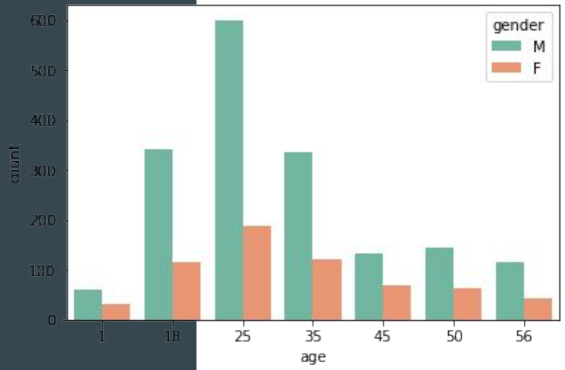
Yuyang Tian
Yuting Xu
Xiaoyun Wang
Don Park

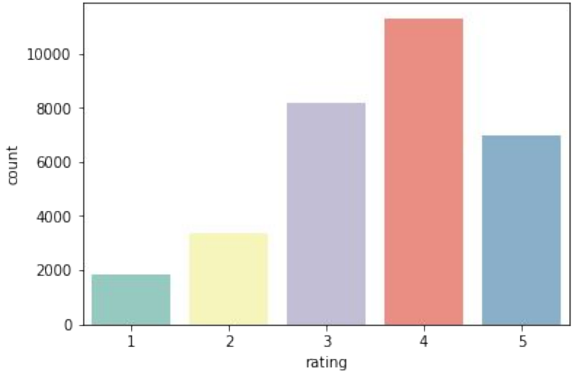
Initial Data Analysis

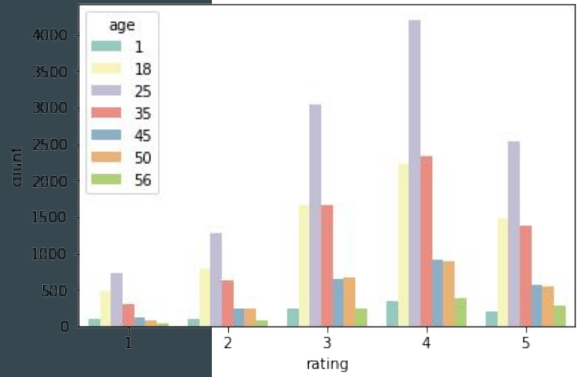
Dataset

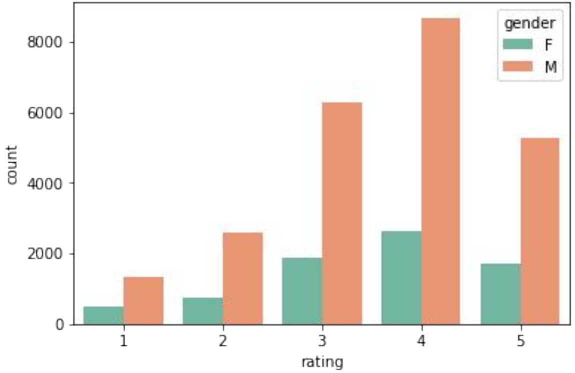
- 31620 ratings
- 1465 movies
- 2353 users
- Density: 0.917%











Collaborative Filtering

Reveal User v.s. Movie Interaction	Add Biases	Optimization
Latent Factor Model (SVD)	Baseline Predictor	Stochastic Gradient Descent

- Find the latent factors
    - Romance? Horror?
- Weigh the relationship between users and factors & movies and factors
- Derive unknown ratings by factors

Separate user and movie behavior

- User Bias
    - Rating scale of a user
    - Behavior of a user
- Movie Bias
    - Popularity of a movie
    - Selection bias

- Stochastic: Fast Convergence in practice

- Gradient Descent: Find optimal components to reconstruct a full matrix with the minimal loss against the original sparse matrix

Matrix Factorization

$$\hat{r}_{ui} = \mathbf{q}_i \mathbf{p}_u = \sum_f q_{if}^T p_{fu}$$

1. "Matrix factorization characterizes both movies and users by vector of factors inferred from item rating patterns." [Koren]
2. How to fill the blank entries?

	user1	user2	user3	user4
movie1	1		4	
movie2		5		
movie3				5
movie4	4		2	
movie5				
movie6		3	4.78	
movie7				3
movie8	2			
movie9		3		
movie10			4	

10 rows ✕ 4 columns

≈

	factor1	factor2	factor3
movie1	q11	q12	q13
movie2	q21	q22	q23
movie3	q31	q32	q33
movie4	q41	q42	q43
movie5	q51	q52	q53
movie6	-0.5	0.6	1.5
movie7	q71	q72	q73
movie8	q81	q82	q83
movie9	q91	q92	q93
movie10	q10 1	q10 2	q10 3

Transpose Q = 10rows ✕ 3columns

✕

(-0.5)*(-2)+0.6*0.3+1.5*2.4=4.78(estimated)

	user1	user2	user3	user4
factor1	p11	p12	-2	p14
factor2	p21	p22	0.3	p24
factor3	p31	p32	2.4	p34

P = 3 rows ✕ 4columns

Objective Function

$$\min_{q_u,p_i} \sum_{u,i} (r_{ui} - \mathbf{q}_i^T \mathbf{p}_u)^2 + \lambda(||\mathbf{q}_i||^2 + ||\mathbf{p}_u||^2)$$

Cost function: min(SSE + Regularization)
SSE: how well the mode is fit
Regularization: avoid overfit

Our goal is to find the Q and P

The idea: minimum of the objective function

Overall: we used known rating to get P,Q then predict rating.

.

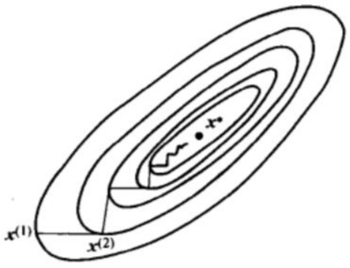
$$r_{xi} = \underbrace{\mu}_{\text{Overall mean rating}} + \underbrace{b_x}_{\text{Bias for user } x} + \underbrace{b_i}_{\text{Bias for movie } i} + \underbrace{q_i \cdot p_x}_{\text{User-Movie interaction}}$$

Stochastic gradient descent

- **Gradient descent optimization**

To minimize the objective function f(x):

—find the direction in which decreases the fastest: negative gradient
—propose a new point $x' = x - \gamma \nabla_x f(x)$, where γ is the learning rate
—converges when every element of the gradient is zero (or, in practice, very close to zero).



- **Stochastic gradient descent**

—Minimize an objective function that has the form of a sum:

$$Q(\omega) = \frac{1}{n}\sum_{i=1}^{n} Q_i(\omega)$$

$$\omega = \omega - \gamma \nabla Q(\omega) = \omega - \gamma \frac{1}{n}\sum_{i=1}^{n} \nabla Q_i(\omega)$$

Problem:
large training sets are more computationally expensive.

—Approximated by a gradient at a single sample

$$\omega = \omega - \gamma \nabla Q_i(\omega)$$

Model Process

- Randomly select a sample r_{ui} from training set
- Predict $\hat{r}_{ui}$ by $\hat{r}_{ui} = q_i^T p_u$
- Compute the associated prediction error $e_{ui} \overset{def}{=} r_{ui} - q_i^T p_u$
- Update the parameters by a learning rate in the opposite direction of the gradient

$$q_i \leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i)$$
$$p_u \leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u)$$

- Repeat step 1-4, until find the optimal p_u, q_i
- Generate the prediction rating matrix

$$R_{pred} = Q^T \times P = [q_1, q_2, \dots, q_I]^T \times [p_1, p_2, \dots, p_U]$$

SVD Only

Configuration

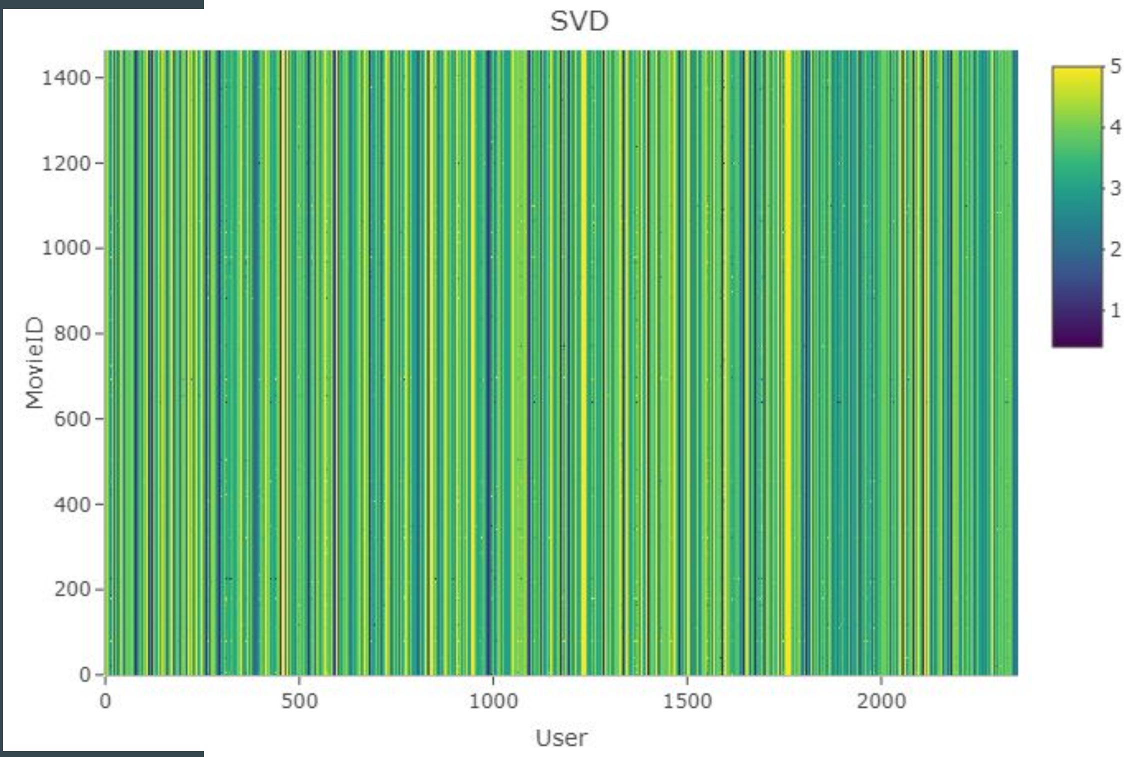
- # of latent factors: 100
- Fill unrated data with 0 (mean)
- Normalized then trained
- No calibration run (default configuration)

Result

RMSE = 1.429

Good start, but can do better.

Biased based on mean rating



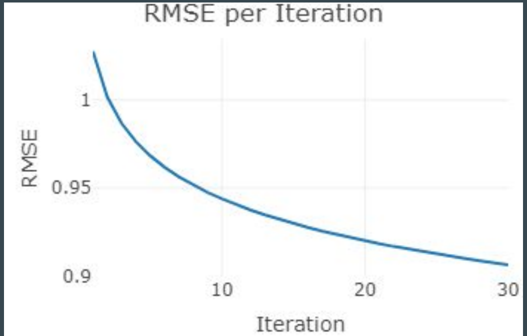
SVD with SGD

Configuration

- # of latent factors: 100
- # of epochs: 30
- Learning Rate: 0.005
- Regularization Term: 0.02
- Fill unrated data with 0 (mean)
- Normalized then trained

Result

RMSE = **0.927**



Summary

Summary

- Matrix Factorization can be used to predict user ratings
- SVD, an extension of Matrix Factorization, is a good model to predict user ratings
- SVD+SGD has improved performance over just SVD