Recommendation Systems

Big Data Lectures – Chapter 7

Lexin Li

Division of Biostatistics
University of California, Berkeley

Outline

- list of topics:
  - basics:
    - the Netflix Challenge
    - terminology
    - long tail property
    - content-based *vs* collaborative filtering
    - challenges
    - evaluation
  - some key techniques:
    - content-based recommendation
    - neighborhood-based methods
    - latent factor models
    - matrix completion
  - additional topics:
    - Google's Adwords problem
    - bandits and dynamic treatment regime

Recommendation Systems: Basics

Netflix challenge

- the Netflix challenge:
  - Netflix, the world's largest internet-based movie rental company
  - October 2, 2006, publicly released a set of data, and offered a Grand Prize of $1m
  - goal: produce a system for recommending movies to users based on predicting how much someone is going to like any particular movie
  - data: ratings (from 1 star to 5 stars) that users have assigned to movies they have seen
    - training set: $\sim$100 million ratings, 480,000 users, 18,000 movies (user-movie rating matrix is $\sim 99\%$ **sparse**)
    - quiz set: $\sim$1.5 million ratings but withheld
    - test set: $\sim$1.5 million ratings but withheld
  - evaluation metric: root mean squared error (RMSE)
  - overall average: 1.0528 for quiz, 1.0540 for test
  - *Cinematch*: 0.9514 for quiz, 0.9525 for test ($\sim 9.5\%$ improvement)
  - grand prize: RMSE 0.8572 (10%), or better, on the test set
  - progress prize: 50k best with at least 1% better than previous year
  - July 26, 2009, the grand prize was won

Introduction

- **recommendation systems:**
  - predict **user** response to **item**
  - item: news article, product/service ads, movie, treatment, drug
- examples:
  - offer online newspaper readers with news articles, based on prediction of reader interests
  - offer customers of an online retailer with suggestions (products / ads) about what they might like to buy, based on their past history of purchases and/or product searches
  - offer patients with (additional) laymen-friendly information helping to better comprehend their health status, based on patients' health records (e.g., as laboratory results, treatment plans, medical reports)
- long-tail property:
  - physical institutions provide only the most popular items, while online institutions provide the entire range of items
  - a good story: *Into Thin Air* and *Touching the Void* on Amazon

Introduction

- (broad) categories of existing solutions:
  - **content-based system**: measures similarity by looking for common features of the items or users; e.g., if a Netflix user has watched many cowboy movies, then recommend a movie classified in the database as having the cowboy genre
  - **collaborative filtering**: measure similarity of users by their item preferences; measure similarity of items by the users who like them — **user-item interaction**
  - **hybrid system**: uses both types of information
- challenges:
  - user-item interactions are extremely **sparse**
  - **cold start** problem:
    - for content-based system, user has to dedicate an amount of effort using the system, so to construct their user profile, before the system can start providing any recommendation — fail for new user
    - for collaborative filtering, it would fail to consider items which no one has rated previously — fail for new item

Introduction

- evaluation:
  - accuracy of the predicted scores:
    - root mean squared error (RMSE)
    - mean absolute error
  - accuracy of the recommended list:
    - precision and recall:

$$\begin{aligned} \text{precision}(L) &= \frac{1}{N_{\text{user}}} \sum \frac{|L(u) \cap T(u)|}{|L(u)|} \\ \text{recall}(L) &= \frac{1}{N_{\text{user}}} \sum \frac{|L(u) \cap T(u)|}{|T(u)|} \end{aligned}$$

    - average precision for a given user u:

$$\frac{\sum_{k=1,\dots,K} \text{precision(k)}}{\text{number of items clicked in } M \text{ recommended items}}$$

where precision(k) is the precision at cut-off k, i.e., the ratio of number of clicked items up to the position k over the number k

Some Key Techniques

Content-based recommendation

- **construction** of feature profile:
  - **item profile**: movie features (genre, release date, cast); news article features (topic, word frequencies); image tags
  - **user profile**: browsing history; demographical information
- recommend items for **a given user**:
  - recommend similar items based on item profiles
  - build a decision / classification rule given item features as covariates
  - would fail for new users

Neighborhood-based recommendation

- key idea:
  - recommend **similar** items, or items of **similar** users
  - simple, efficient, stable

- item-based recommendation:

$$\hat{r}_{ui} = \bar{r}_i + \frac{\sum_{j \in \mathcal{N}_u(i)} w_{ij}(r_{uj} - \bar{r}_j)}{\sum_{j \in \mathcal{N}_u(i)} w_{ij}}$$

  - for user u, the rating for item i is the weighted average of the same user's ratings on **similar** items $j \in \mathcal{N}_u(i)$
  - $\bar{r}_i$ is the average rating of all users have given to item i; **mean-centering normalization**

Neighborhood-based recommendation

- user-based recommendation:

$$\hat{r}_{ui} = \bar{r}_u + \frac{\sum_{v \in \mathcal{N}_i(u)} w_{uv}(r_{vi} - \bar{r}_v)}{\sum_{v \in \mathcal{N}_i(u)} w_{uv}}$$

  - for item i, the rating by user u is the weighted average of the ratings from **similar** users $v \in \mathcal{N}_i(u)$
  - $\bar{r}_u$ is the average rating of all items that user u has given; **mean-centering normalization**

- key components:
  - **similarity measure / weight**: Pearson correlation or other measures; based on **user and item profiles**
  - **neighborhood selection**: to address **data sparsity**, can cluster users and/or items into small groups with strong similarity first

Latent factor models

- **latent factor model / matrix factorization**:
  - key idea: map both users and items to a joint latent factor space of dimensionality k, such that user-item interactions are modeled as inner products in that space
  - each item i is associated with a vector $q_i \in \mathbb{R}^k$, and each user u is associated with a vector $p_u \in \mathbb{R}^k$
  - $q_i^\mathsf{T} p_u$ captures the interaction between user u and item i, i.e., the overall interest of the user in characteristics of the item
  - model only the **observed** ratings; avoid overfitting via **regularization**

- model and estimation:
  - model: for a known link function g,

$$g(E(r_{ui})) = b_0 + b_i + b_u + q_i^\mathsf{T} p_u$$

  - estimation: take $g =$ identity as an example, minimize

$$\sum_{observed} \left(r_{ui} - b_0 - b_i - b_u - q_i^\mathsf{T} p_u\right)^2 + \lambda(b_i^2 + b_u^2 + \|q_i\|^2 + \|p_u\|^2)$$

Latent factor models

- latent factor models **incorporating user/item features**:
  - key idea: model item/user latent factors based on item/user features
  - use a Bayesian framework for computation

- model setup:

$$\begin{aligned}
r_{ui} &\sim \mathrm{Normal}(\mu_{ui}, \sigma^2) \\
r_{ui} &\sim \mathrm{Bernoulli}(\mu_{ui}) \\
g(\mu_{ui}) &= b_0 + b_i + b_u + q_i^{\mathsf{T}} p_u \\
b_i &= \alpha^{\mathsf{T}} z_i + \epsilon_{ib}, \quad \epsilon_{ib} \sim \mathrm{Normal}(0, \sigma_i) \\
b_u &= \beta^{\mathsf{T}} x_u + \epsilon_{ub}, \quad \epsilon_{ub} \sim \mathrm{Normal}(0, \sigma_u) \\
q_i &= \Phi z_i + \epsilon_{iq}, \quad \epsilon_{iq} \sim \mathrm{MVN}(0, \Sigma_i) \\
p_u &= \Psi x_u + \epsilon_{up}, \quad \epsilon_{up} \sim \mathrm{MVN}(0, \Sigma_u)
\end{aligned}$$

Matrix completion

- matrix completion / factorization with **rank regularization**:
  - key idea: assume the incomplete rating matrix is of a **low rank** structure
  - solution: directly regularize the rank of the rating matrix
  - regularization via matrix nuclear norm, which is a suitable convex relaxation of the rank constraint
- model and estimation: minimize

$$\frac{1}{2} \sum_{observed} (r_{ui} - R_{ui})^2 + \lambda \|\mathbf{R}\|_*$$

  - $\mathbf{R}$ is the estimated rating matrix
  - $\|\mathbf{R}\|_* = \sum_j \sigma_j$ is the nuclear norm and σ_j's are the singular values of $\mathbf{R}$
  - the summation is only over the observed entries

Additional Topics

Adwords problem

- story time...
  - Adwords problem: a fundamental problem of **search advertising**
  - history: around year 2000, a company called Overture (later bought by Yahoo!) introduced a new (revolutionary) kind of search — advertisers bid on keywords (words in a search query), and when a user searched for that keyword, the links to all the advertisers who bid on that keyword are displayed in the order highest-bid-first; if the advertiser's link was clicked on, they paid what they had bid
  - Google Adwords system: years later, Google adapted the idea in a system called **Adwords**
- refinements and constraints:
  - show only a limited number of ads with each query at a certain order
  - users specify a total budget (in a month)
  - does not simply order ads by the amount of the bid, but by the amount they **expect** to receive for display of the ad — the value of an ad was taken to be the product of the bid and the **click-through rate**
  - **on-line** decision making

Adwords problem

- Adwords problem:
  - input:
    - a set of bids by advertisers for search queries
    - a click-through rate for each advertiser-query pair
    - a budget for each advertiser (for a month, or other time length)
    - a limit on the number of ads to be displayed with each search query
  - output: respond to each search query with a set of advertisers s.t.
    - the size of the set is no larger than the limit on the number of ads per query
    - each advertiser has bid on the search query
    - each advertiser has enough budget left to pay for the ad if it is clicked upon
- greedy algorithm, balance algorithm, implementation

Bandits and dynamic treatment regimes

- **multi-armed bandits**:
  - a gambler at a row of slot machines decides which machines to play, how many times to play each machine and in which order to play them
  - when played, each machine provides a random reward from a distribution specific to that machine
  - objective: maximize the sum of rewards through a sequence of plays
  - **contextual bandits**: contexts such as user's geographical information; health information, provide additional useful information

- **personalized treatment**:
  - recommendation of treatments, medicine, ...
  - sequentially finds treatment decisions over the course of a patient's disease based both on both his/her baseline and evolving characteristics, and his/her responses to previous treatments, with the goal of yielding the most favorable outcome on average

Bandits and dynamic treatment regimes

- **personalized web services**:
  - recommendation of news articles, ads, services, products, ...
  - sequentially selects articles to serve users, based both on contextual information about the users and articles, and on user-click feedback, to maximize total user clicks, with the goal of maximizing total user clicks in the long run
  - when a presented article is clicked, a payoff of 1 is incurred; otherwise, the payoff is 0
  - with this definition of payoff, the expected payoff of an article is precisely its **click-through rate (CTR)**, and choosing an article with maximum CTR is equivalent to maximizing the expected number of clicks from users, which in turn is the same as maximizing the total expected payoff in the bandit formulation

Bandits and dynamic treatment regimes

- basic concepts:
  - **action (arm)**: A; a set of treatments (often binary), news articles, ads
  - **reward**: $R \in [0, 1]$; reward for the action a is observed, but the rewards of other actions are not
  - **context**: $\boldsymbol{X}$; a vector of covariates of patients / users
  - **policy**: a function mapping the past observations and the contextual information to a distribution over the actions
- tasks:
  - **policy evaluation**: to estimate the expected total reward (the mean response) of a given policy
  - **policy optimization**: to obtain a policy that (approximately) maximizes expected total rewards
- policy:
  - **stationary policy**: the action depends on the current, observed context alone
  - **nonstationary policy**: the action depends on both the current, observed context, and the observed history of context-action-reward

Policy evaluation

- stationary policy evaluation:
  - direct method: estimate the reward function from data and use this estimate in place of actual reward to evaluate the policy

$$\frac{1}{n}\sum_{k=1}^{n}\sum_{a\in\mathcal{A}}\nu(a|\boldsymbol{x}_k)\hat{R}(\boldsymbol{x}_k, a)$$

  - inverse probability weighted estimator: use importance weighting to correct for the incorrect proportions of actions in the history

$$\frac{1}{n}\sum_{k=1}^{n}\frac{\nu(a_k|\boldsymbol{x}_k)}{\hat{\mu}_k(a_k|\boldsymbol{x}_k)}\cdot R_k$$

  - doubly robust estimator: combine the above two
- nonstationary policy evaluation: Dudik et al. (2014)

Policy optimization

- one trial:
  - action/treatment: $A = 0/1$; reward/observed outcome R; context/baseline covariates $\boldsymbol{X}$; potential outcomes: $R^*(0), R^*(1)$
  - for any policy / treatment regime, the potential outcome is: $R^*(g) = g(\boldsymbol{X})R^*(1) + \{1 - g(\boldsymbol{X})\}R^*(0)$
  - optimal treatment regime: $g^{opt} = \arg\max_{a \in \mathcal{A}} E\{R^*(g)\}$
  - under some assumptions, one can write $E\{R^*(g)\}$ as:

$$E\{R^*(g)\} = E_{\boldsymbol{X}}\left[E(R|A=1,\boldsymbol{X})g(\boldsymbol{X}) + E(R|A=0,\boldsymbol{X})\{1 - g(\boldsymbol{X})\}\right]$$

- Q-learning:
  - estimate $E(R|A=1,\boldsymbol{X})$ and $E(R|A=0,\boldsymbol{X})$, and compare the two estimates
- A-learning:
  - estimate $E(R|A=1,\boldsymbol{X}) - E(R|A=0,\boldsymbol{X})$, and compare to zero

Policy optimization

- multiple trials:
  - **dynamic treatment regimes (DTR)**:
    - a set of rules that specify what treatment to choose, given current characteristics, past treatment history and outcomes
  - multi-arm bandits:
    - choose a sequence of actions $A_t = a_t$, $t = 1, 2, \ldots, T$, to minimize its T-trial regret with respect to the optimal arm-selection strategy

$$E\left[\sum_{t=1}^{T} R_{t,a_t^*}\right] - E\left[\sum_{t=1}^{T} R_{t,a_t}\right]$$

    - balance: the **exploitation** of actions that did well in the past and the **exploration** of actions that might give higher payoffs in the future
  - ϵ-greedy algorithm:
    - in each trial t, first estimates the average payoff of each arm a
    - with probability $1 - \epsilon$ chooses the greedy arm, i.e., the arm with highest payoff estimate; with probability ϵ chooses a random arm
    - by decaying ϵ properly, $R_A(T)/T$ converges to 0 with probability 1
  - **targeted maximum likelihood estimation (TMLE)**

Additional readings

- Agarwal, D., and Chen, B.C. (2009). Regression-based latent factor models. *KDD'09*
- Agarwal, D., Zhang, L., and Mazumder, R. (2011). Modeling item-item similarities for personalized recommendations on Yahoo! front page. *Annals of Applied Statistics*, **5**, 1839-1875
- Dudik, M., Erhan, D., Langford, J., and Li, L. (2014). Doubly robust policy evaluation and optimization. *Statistical Sciences*, **29**, 485-511
- Feuerverger, A., He, Y., and Khatri, S. (2012). Statistical significance of the Netflix challenge. *Statistical Sciences*, **27**, 202-231
- Rajaraman, A., Leskovec, J., and Ullman, J.D. (2012). *Mining of Massive Datasets*. Chapters 8, 9
- Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (Eds.) (2011). *Recommender Systems Handbook*. Springer
- Schulte, P.J.; Tsiatis, A.A.; Laber, E.B.; Davidian, M. (2014). **Q**- and **A**-Learning Methods for Estimating Optimal Dynamic Treatment Regimes. Statistical Science, **29**, 640-661