Data Generation via Latent Factor Simulation for Fairness-aware Re-ranking

The 7th FAccTRec Workshop on Responsible Recommendation at RecSys 2024

Elena Štefancová, Cassidy All, Joshua Paup,
Martin Homola, Nicholas Mattei, Robin Burke

COMENIUS UNIVERSITY BRATISLAVA
University of Colorado Boulder
Tulane University

Fairness-Aware Recommendation

- **approaches**
  - pre-processing
  - model-based
  - post-processing or re-ranking
- **challenges**
  - limited repertoire of data sets
  - limited distribution of protected items / groups
- **solution**
  - **synthetic data**

Synthetic Data Generation

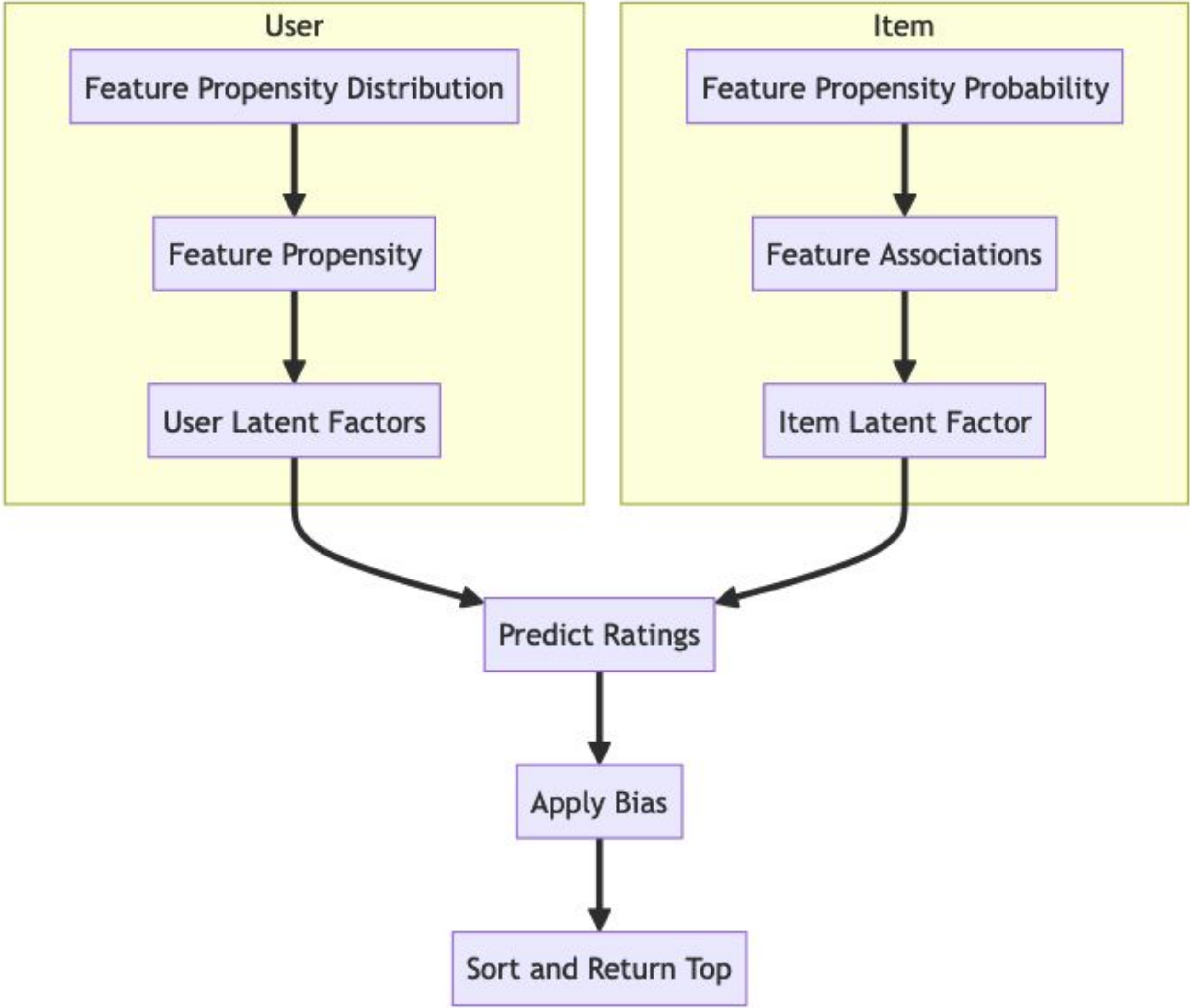
- heuristic-based
  - leverage strong assumptions on the dataset distributions and properties
  - e.g DataGenCARS (user profiles designed manually)
- clustering-based
  - profiles mined from historical data
  - quality depends on the assumptions made about user behavior
  - e.g. Monti et Al. 2019, Pasinato et Al. 2013
- learning from a data source
  - matrix factorization, variational auto-encoders
  - unrealistic models, low dimensionality, sparsity, lack of item/user metadata
- data obfuscation
  - maintains user privacy by modifying existing data but retains real user data

Simulated ratings are not needed

- to study re-ranking
  - what we need to simulate is the output of a recommender system
    - for each user, top k items with associated ratings
  - we don’t need to create rating profiles, build a recommender and produce predictions
- to study fairness
  - we need to simulate how unfairness impacts recommender system behavior
    - protected item features
      - that may (through bias) influence predicted ratings
    - minority users
      - that have different preferences from the majority

Latent Factor Simulation (LAFS)

- a probabilistic approach to handling protected features
- inspired by the matrix factorization technique for collaborative filtering



LAFS: Generating users

- **Core ideas**
  - produce a user latent factor matrix as if matrix factorization had been applied (k factors)
  - assume that protected features are capture by a subset of latent factors
- **Experimenter specification**
  - experimenter specifies a (normal) feature propensity distribution
    - For each latent factor, what is the mean and variance across the user base
    - Experiment may want to have multiple type of users with different characteristics
- **Generate "users"**
  - draw from this distribution to create user profiles with propensities for each of the k factors
- **Generate factors**
  - treat the propensity as a mean value and draw a sample from a normal distribution centered at this point
  - the reason to do this is so that user factors are not deterministically tied to the propensities

LAFS: Generating items

- **Similar to users**
  - item factor matrix
  - difference: the propensity vector is binary
    - the item has the feature or it doesn't
  - this enables us to categorize items as having protected values of sensitive features or not
- **Experimenter specification**
  - vector of probabilities (instead of mean and variance)
- **Generate items**
  - vector of binary-valued features for each
- **Generate item factors**
  - as with users, treat the propensity as a mean value and draw a sample from a normal distribution centered at this point

LAFS: Putting them together

- **Create Predictions**
    - for each user, predict random *m* items and compute their ratings multiplying user factor and item factor
- **Inject Bias** (against protected items)
    - applying a randomly generated penalty to the computed rating for its each sensitive feature
    - the penalty is drawn from an experimenter-specified vector of bias distributions
- **Create Recommendations**
    - for each user, sort items by rating and select the top *n* as recommendations
        - list focuses on highest predicted ratings as a recommender would
- **Normalization**
    - an experimenter-specified min-max transformation is applied
        - so the data looks more like RecSys output

Challenges and Future Work

- **evaluating synthetic datasets** for recommender systems
  - lack of predictive measures connecting dataset properties with performance
- **compare synthetic and real-world** output and latent factors
  - explore a variety of visualizations, metrics and other techniques
- **incorporate new features** for more realistic simulations
  - item popularity, feature correlation

github.com/that-recsys-lab/lafs

LAFS: Formal specification

n_i	The number of items
n_u	The number of users
k	The number of latent factors
s	The number of sensitive factors ($s < k$)
σ_f	Standard deviation for factor generation
$d_u = [(\mu_{u1}, \sigma_{u1}), (\mu_{u2}, \sigma_{u2})..., (\mu_{uk}, \sigma_{uk})]$	Feature propensity distributions for users
$\Pi_u = [\pi_{u1}, \pi_{u2}, ..., \pi_{uf}]$	Feature propensities for users
$\Pi_U = [\Pi_u \forall u]$	The matrix of all user-feature associations
U	$< n_u \times n_f >$ matrix of user latent factors
$d_i = [d_{i1}, d_{i2}, ..., (d_{if}]$	Feature propensity probabilities for items
$\Pi_i = [\pi_{i1}, \pi_{i2}, ..., \pi_{if}]$	Feature associations for item i
$\Pi_I = [\Pi_i \forall i]$	The matrix of all item-feature associations
V	$< n_i \times n_f >$ matrix of item latent factors
l'	The size of the initial recommendation list
l	The size of the output recommendation list
$B = [(\mu_1, \sigma_1), (\mu_2, \sigma_2), ...(\mu_f, \sigma_s)]$	Bias generators for sensitive features