Experimental Protocols

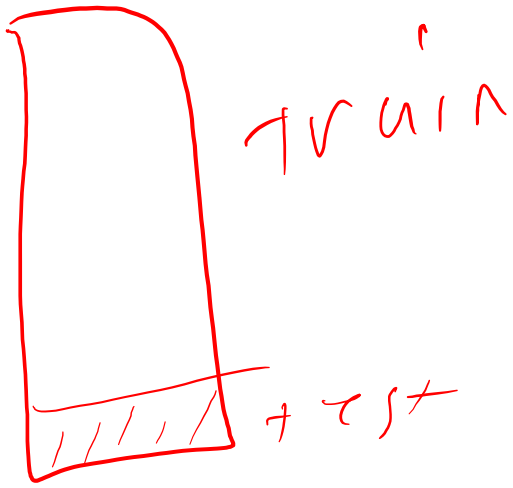
Introduction

- We’ve seen several metrics, basics of cross-validation
- We now turn to the details of cross-validation

Goals of Offline Evaluation

- To *estimate* the recommender's quality
  - High-throughput evaluation
  - Answer important research questions
- Often cannot answer if recommender really works
  - User-based evaluation needed
  - Link to business metrics is weak

Machine Learning



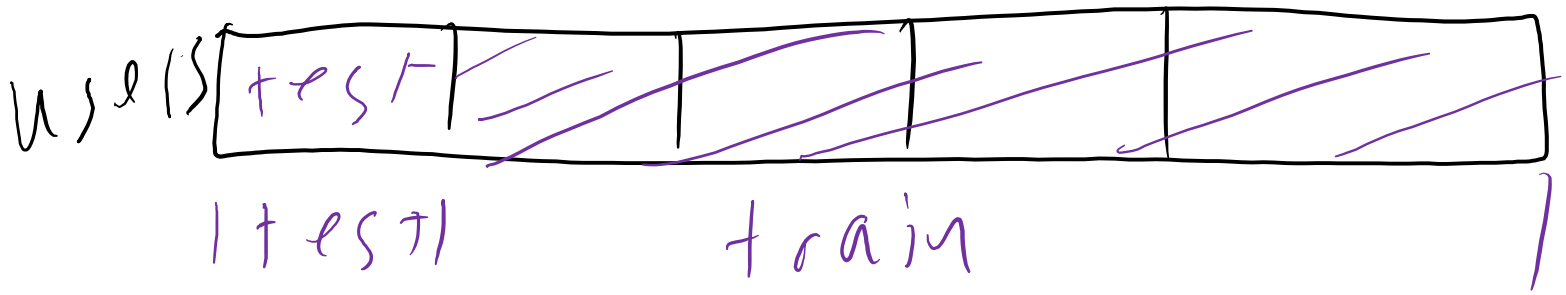
- Hidden data
  - Hold out some data
  - Try to predict/classify it
- Cross-validation
  - Split data into partitions, hold out each in turn
  - Average results
  - Allows us to test over all items/ratings/users
  - Multiple test sets improve reliability
- Measure score or classification accuracy

Basic Structure

- Partition data set into *k* partitions
- For *i* = 1 to *k*
  - train on all sets other than *I*
  - test on set *I*
- What *k* to use?
  - Large values → more training data
  - Small values → more efficient
  - 5 and 10 are common

Splitting data

- Split ratings
- Split users
  - Allows more control for measuring expected user experience
  - Only some ratings from each test user put in test set
    - Remainder kept in train set w/ other users' ratings
- Split items
  - Rarely, if ever, done


users
test
|test|
train
|train|test|
train
Probe
Test
ratings for u

Splitting users

- Split user ratings randomly
  - Very common
  - Use to compare with existing results
- Split user ratings by time
  - More accurate simulation of user experience
  - Results often ‘worse’
- Best, but expensive: only train on ratings before time of test rating

Using log data

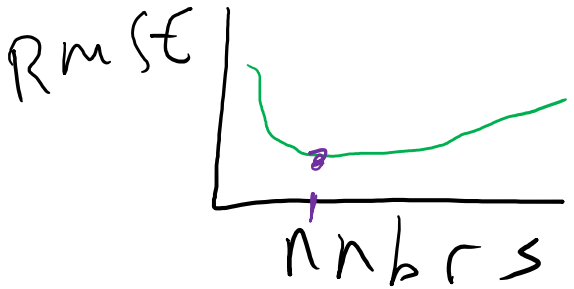
- Often unary (clicked, purchased), nothing known about absent items
- Basic structure is the same
- More in the next lecture

Application: Parameter Sweep

Goal: find best parameter values (e.g. neighborhood size)

Method:

- Cross-fold data set
- Try many settings, test with test sets
- Pick the best on 1 or more metrics



Overfitting

Overfitting is when the recommender learns its training data too well, hurting general performance

When sweeping for parameters:

- Set aside a *validation* set of ratings
- Cross-fold on remaining ratings to pick optimal parameter values
- Report results of best configuration (from cross-fold) on validation set

Good Practice

- Split users into *k* partitions (5 is common)
- Split user ratings by time
  - Use random to compare with previous results
- Include user query ratings in train data
- Document your protocol carefully
  - So you can run it again
  - So others can compare

Variations

- Generate *k* samples of users instead of partitions
  - Can be more efficient
- Only hold out 1 item per user
  - Very targeted recommendation testing
- Stratified sampling

Document your protocol

Experimental Protocols