Recipe Project Recommender Systems

Kevin Li

Problem: Build and Test a Recipe Recommender System

Goal: High precision → of the 30,000 recommendations, earn the highest fraction of recipes actually downloaded by the 3,000 test users

Algorithms: Popular (baseline), Collaborative Filtering (User-Based and Item-Based), SVD, and Content-Based Filtering / LSA.

Given:

Data set with 200 recipe names and the target variable to determine whether it will be used in testing predictions (target = 1)

Data set with 10,000 user IDs and the test variable to determine whether that user will be used as a member of the test set (test =1)

Data set connectings users to recipes by the count of downloads for a certain recipe ID by that user

Recipes

- 150 recipes as the query set
- 50 recipes as the target set

Users

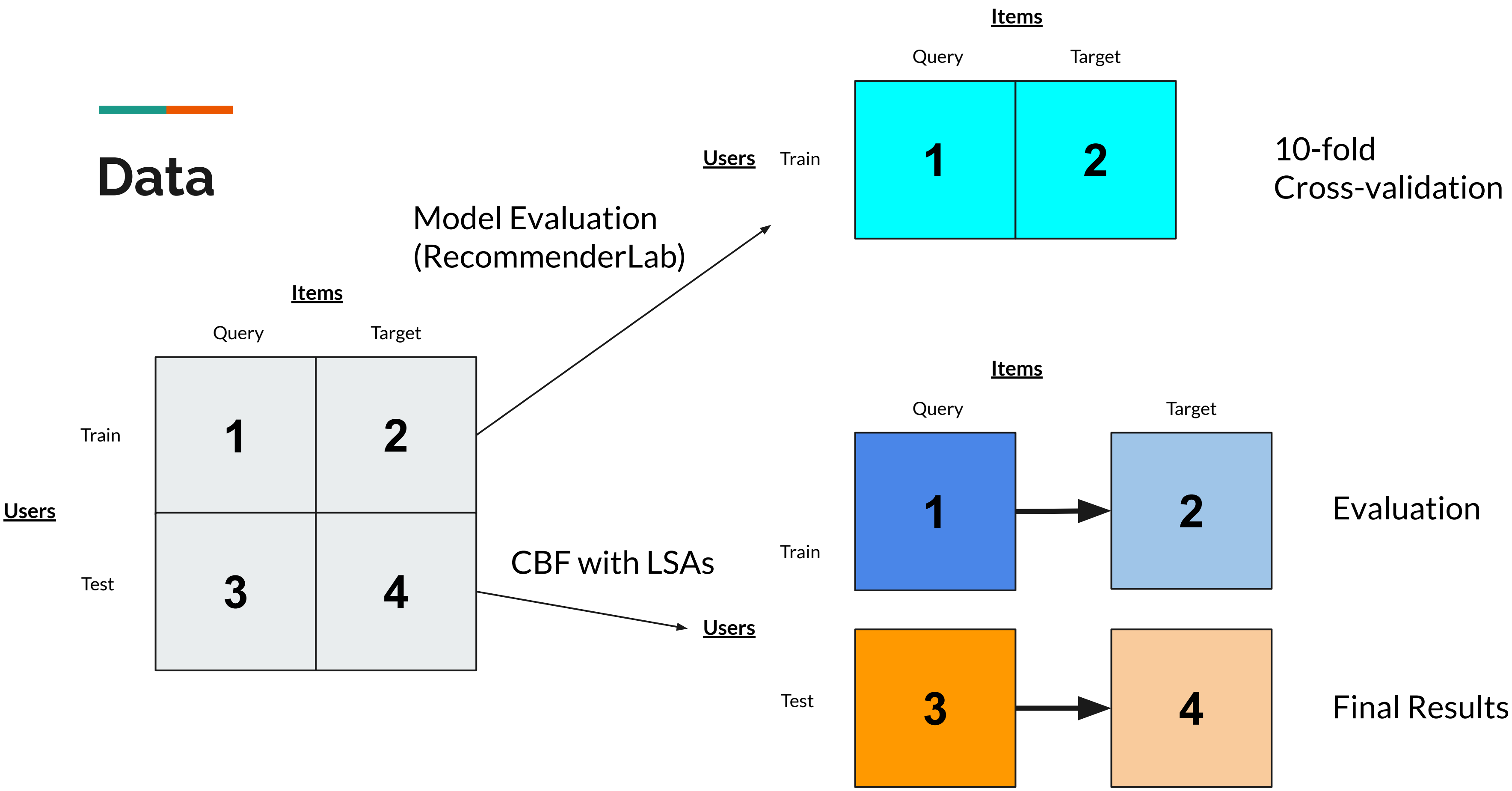
- 7,000 users as the training set
- 3,000 users as the test set

Questions: Which methods worked best and what we did to make the methods work well?

Findings

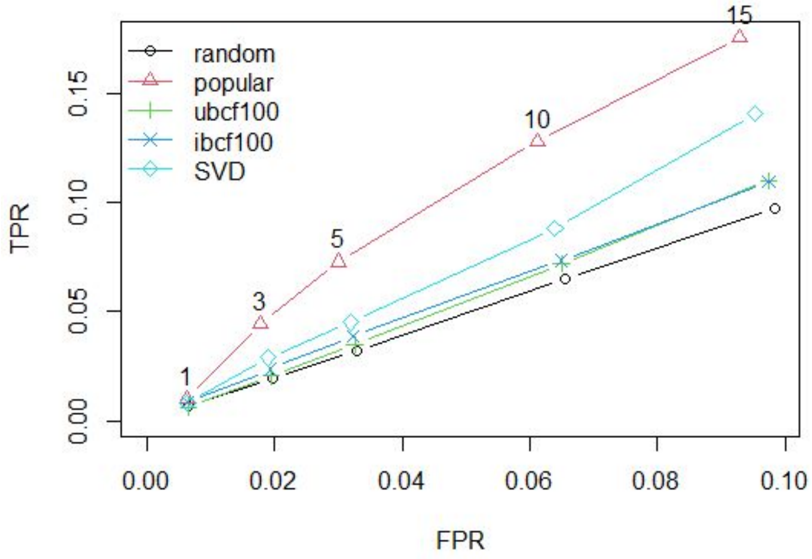
1) Latent semantic analysis / content-based filtering (0.291)
2) Item-item collaborative filtering when using a binarized matrix and GoodRating (0.147)
   a) Without the binarized matrix, precision was 0.073
3) Popular (0.141)
4) Singular Value Decomposition (0.0880)
5) User-user collaborative filtering (0.065)

Data

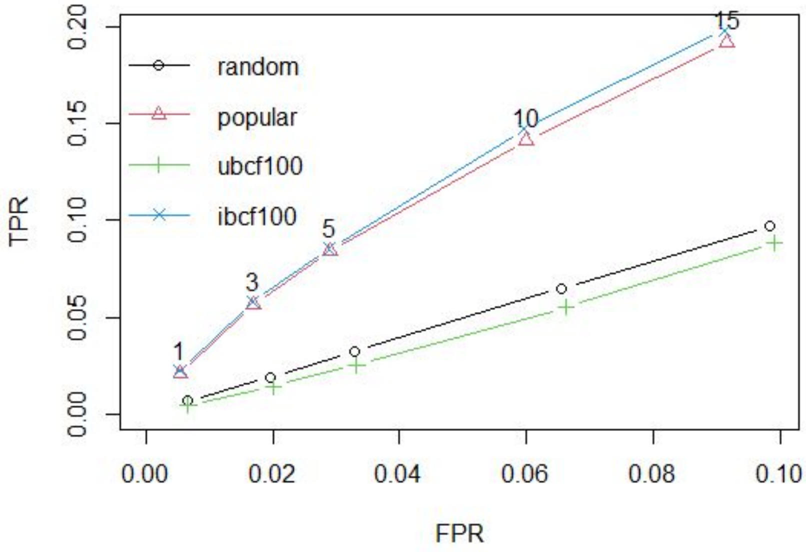

Model Evaluation
(RecommenderLab)
Items
Query
Target
Train
Users
Test
1
2
3
4
CBF with LSAs
Items
Query
Target
Users
Train
1
2
10-fold
Cross-validation
Items
Query
Target
Train
Test
1
2
3
4
Users
Evaluation
Final Results

Model Evaluation Using RecommenderLab

- **Preprocessing:** recipe id mapping table
- **Create the Rating Matrix:**
  - Real rating matrix
  - Binary rating matrix (doesn't support SVD)
- **Types of algorithms**
  - Popular items (POPULAR) – **Benchmark**
  - User-based collaborative filtering (UBCF)
  - Item-based collaborative filtering (IBCF)
  - SVD with column-mean imputation (SVD)
  - Funk SVD (SVDF)
- **Evaluation metrics**
  - ROC curve: TPR vs FPR
  - Precision-Recall curve: Precision vs Recall
- **Evaluate the models & Visualize the results**



Using Real Rating Matrix
with good rating set to 1

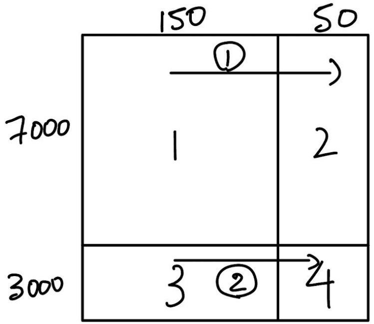


Using Binary Real Rating Matrix

Content Based Filtering - LSA

Purpose

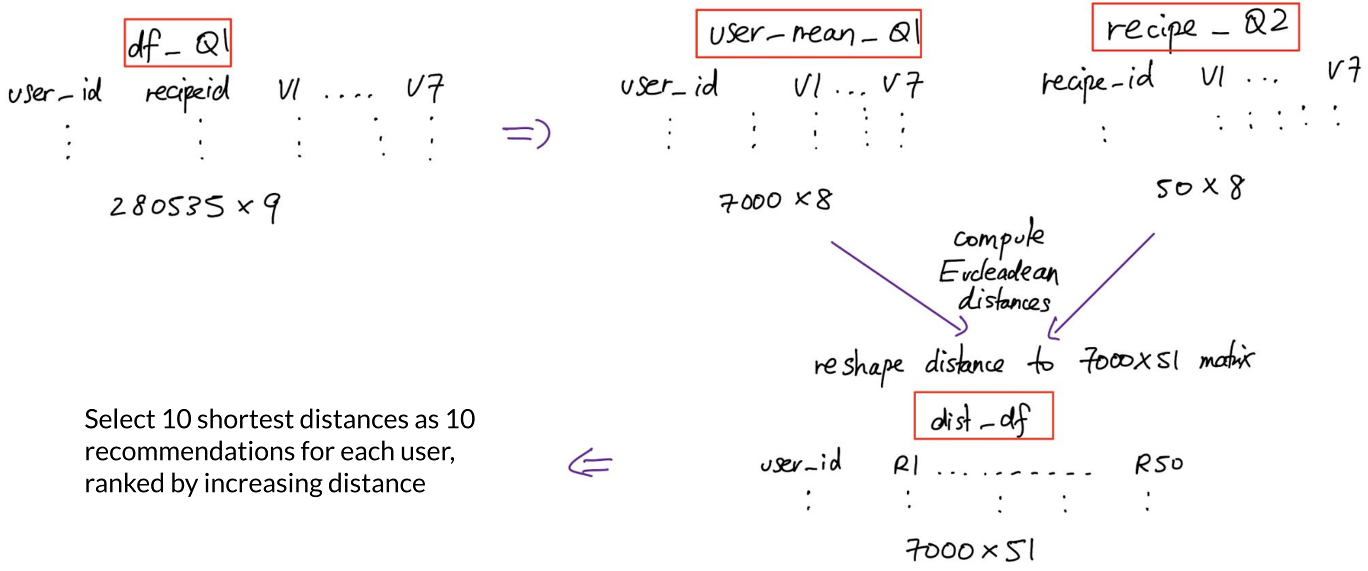
Utilizing pre-processing from Homework 5 (lemmatizing, stop words, special replacements) to create user profile for each user based on recipes they have previously downloaded, allowing us to recommend recipes that are similar to their previous behaviors.



General Methodology

1. Use Quadrant 1 to predict Quadrant 2 recommendations
2. Calculate Precision@10
3. Use Quadrant 3 to predict Quadrant 4 recommendations

LSA Flowchart



Testing Set

	id	recipeid
1	1	80
2	1	95
3	1	125
4	1	146
5	1	156
6	1	163
7	1	179
8	1	199
9	1	7
10	1	12
11	1	18
12	1	19
13	1	39
14	2	80
15	2	86
16	2	95
17	2	116
18	2	122
19	2	126
20	2	143
21	2	146
22	2	152
23	2	156
24	2	168
25	2	173
26	2	179
27	2	6
28	2	10
29	2	11
30	2	12

Recommendations

	user_id	recipe_id
1	1	143
2	1	149
3	1	173
4	1	85
5	1	4
6	1	47
7	1	146
8	1	6
9	1	58
10	1	154
11	2	20
12	2	152
13	2	10
14	2	116
15	2	122
16	2	32
17	2	149
18	2	55
19	2	143
20	2	58
21	5	80
22	5	122
23	5	2
24	5	32
25	5	194
26	5	6
27	5	11
28	5	146
29	5	10
30	5	49

→

Precision@10

	user_id	prop_correct
1	1	0.1
2	2	0.7
3	5	0.5
4	6	0.4
5	7	0.3
6	8	0.2
7	10	0.3
8	11	0.2
9	12	0.4
10	13	0.4
11	14	0.6
12	16	0.3
13	17	0.2
14	19	0.3
15	22	0.3
16	23	0.3
17	24	0.2
18	25	0.1
19	26	0.2
20	27	0.6
21	29	0.4
22	30	0.2
23	31	0.5
24	34	0.3
25	37	0.0
26	40	0.3
27	41	0.1

→

Precision@10 value:

29.1%

Creating Quadrant 4 Predictions

user_id	Rec_1	Rec_2	Rec_3	Rec_4	Rec_5	Rec_6	Rec_7	Rec_8	Rec_9	Rec_10
3	122	41	60	164	29	51	174	152	86	194
4	143	29	32	116	55	18	85	49	149	20
9	55	49	30	48	154	18	149	20	11	95
15	109	39	6	58	164	154	86	47	199	7
18	122	41	60	164	51	174	29	152	194	86
20	19	109	69	10	48	47	12	179	2	149
21	168	41	194	60	174	11	164	51	58	29
28	12	193	48	112	10	69	7	18	15	51
32	80	30	55	48	18	154	149	122	20	194
33	12	193	48	10	112	69	18	7	15	179
35	18	55	30	80	149	48	7	143	15	154
36	39	29	47	193	69	49	95	173	122	194
38	194	168	174	60	55	11	42	49	80	51
39	12	193	18	48	112	7	15	10	32	55
42	95	143	32	49	18	164	15	149	146	194
44	29	69	193	163	194	49	19	149	47	10
53	168	164	122	60	15	194	32	143	41	18
54	6	163	2	49	179	199	194	164	143	30
55	173	39	58	154	6	47	122	20	95	49
60	112	173	86	125	7	6	193	85	154	11
61	164	125	7	95	85	168	58	199	173	112
62	29	95	58	164	6	146	49	15	168	122
64	29	193	69	194	163	19	49	47	48	10
67	168	41	126	11	194	80	174	60	4	42
72	29	193	69	163	47	194	39	49	19	10
73	194	55	168	174	60	11	80	49	42	58
74	80	55	49	48	11	194	42	4	30	149
75	109	39	6	58	164	154	7	86	47	199
76	29	18	95	32	49	15	164	149	7	194

Dim: 3000 x 11

Precision@10 value:

28.7%

Models	Parameters	TPR	FPR	Precision	Recall
Recommenderlab-evaluationScheme					
Popular		0.1412	0.0600	0.1412	0.1412
UBCF	nn = 10 nn = 20 nn = 50 nn = 100	0.0609 0.0596 0.0624 0.0655	0.0615 0.0616 0.0612 0.0607	0.1217 0.1191 0.1247 0.1309	0.0609 0.0596 0.0624 0.0655
IBCF	k = 10 k = 50 k = 100	**0.1415** **0.1437** **0.1474**	0.0600 0.0599 0.0596	0.1415 0.1437 0.1474	0.1415 0.1437 0.1474
SVD	None	0.0880	0.0639	0.0880	0.0880
SVDF	None	0.0879	0.0647	0.0879	0.0879
Manually calculated					
CBF with LSAs		**0.291**		**0.291**	**0.291**

LSA CBF

Pros

- Clustering: Isolate the core concepts that differentiate recipes, and use that to drive business goals
- May work well for recipes because users already have a set idea of what they are looking for

Cons

- Wouldn’t work as well for more diverse / varying documents
- Doesn’t prioritize serendipity / novelty
- Less impact from other users, creates a less social experience

Thanks for listening!

Any questions?