

Names: Bob Skowron, Jason Walker

Keys: rskowron, jwalker

SVN: jwalker: https://svn.seas.wustl.edu/repositories/jwalker/cse427s_f17/

1.
 - a. The *InputFormat* to use is *KeyValueTextInputFormat*. The default split is tab so no extra settings necessary.
Code to add to driver: *job.setInputFormatClass(KeyValueTextInputFormat.class)*
To get the file path, we use the context to get the current input split and then read the path and name
FileSplit fileSplit = (FileSplit)context.getInputSplit();
String filename = fileSplit.getPath().getName();
 - b. See SVN
 - c. Mapper output for lines from Hamlet:
have hamlet@282
heaven hamlet@282
and hamlet@282
earth hamlet@282
there hamlet@133
are hamlet@133
more hamlet@133
things hamlet@133
in hamlet@133
heaven hamlet@133
and hamlet@133
earth hamlet@133

Names: Bob Skowron, Jason Walker

Keys: rskowron, jwalker

SVN: jwalker: https://svn.seas.wustl.edu/repositories/jwalker/cse427s_f17/

2. a. Cosine similarity: $\frac{r_x^T \cdot r_y}{\|r_x\| \cdot \|r_y\|}$

Normalizing (i.e. subtracting the relevant means) gives: $\frac{(r_x - \bar{r}_x)^T \cdot (r_y - \bar{r}_y)}{\|r_x - \bar{r}_x\| \cdot \|r_y - \bar{r}_y\|}$

This is equivalent to: $\frac{\sum_{s \in S_{xy}} (r_{xs} - \bar{r}_x) * (r_{ys} - \bar{r}_y)}{\sqrt{\sum_{s \in S_{xy}} (r_{xs} - \bar{r}_x)^2} * \sqrt{\sum_{s \in S_{xy}} (r_{ys} - \bar{r}_y)^2}}$

Which is the Pearson correlation.

- b.
 - An advantage of normalization is to remove bias. We can remove users who are overly critical (all low scores) or overly enthusiastic (all high rankings). Additionally, it improves the cosine similarity measure (now equivalent to the Pearson correlation).
 - Normalization requires that you compute and store the average rating for each user. This compute and storage of mean values must be performed each time the ratings change and requires pre-processing the entire set of ratings each time it changes.
- c. One disadvantage of the Jaccard similarity is that it does not take into account the value of the rating. Thus, even if two users rate the exact same items with opposite ratings it would have a high Jaccard similarity ranking. To overcome this problem, we could try to eliminate the apparent similarity between movies a user rates highly and those with low scores by rounding the ratings. For instance, we could consider ratings of 3, 4, and 5 as a “1” and consider ratings 1 and 2 as unrated.

Names: Bob Skowron, Jason Walker

Keys: rskowron, jwalker

SVN: jwalker: https://svn.seas.wustl.edu/repositories/jwalker/cse427s_f17/

3. a. The dual approach can be more efficient if the $\#users \gg \#items$. Also, it's easier to find similar items since items typically belong to a specific genre, ex. 60's rock versus 1700's baroque. It's more common for users to like items from different genres

b.

User	Movie1	Movie2	Movie3	Movie5
user1	1	3	2	
user2		2	3	5
user3	1	2		

Job 1 - Item Co-occurrence

Mapper Output: (user, (movie, rating))

(user1, (movie1, 1))

(user1, (movie2, 3))

(user1, (movie3, 2))

(user2, (movie2, 2))

(user2, (movie3, 3))

(user2, (movie5, 5))

(user3, (movie1, 1))

(user3, (movie2, 2))

Reducer Input: (user, list<movie, ratings>)

(user1, [(movie1,1), (movie2, 3), (movie3, 2)])

(user2, [(movie2, 2), (movie3, 3), (movie5, 5)])

(user3, [(movie1,1), (movie2, 3)])

Reducer Output: ((movie-id1, movie-id2), (ratings1, ratings2))

For movie-id1; For movie-id2 > movie-id1

((movie1,movie2), (1,3))

((movie1,movie3), (1,2))

((movie2,movie3), (2,3))

((movie2,movie5), (2,5))

((movie3,movie5), (3,5))

((movie1,movie2), (1,3))

Job 2 - Item Similarity

Mapper Output (identity)

((movie1,movie2), (1,3))

((movie1,movie3), (1,2))

((movie2,movie3), (2,3))

((movie2,movie5), (2,5))

((movie3,movie5), (3,5))

((movie1,movie2), (1,3))

Reducer Input:

((movie1,movie2), [(1,3),(1,2)])

((movie1,movie3), (1,2))

((movie2,movie3), [(3,2),(2,3)])

((movie2,movie5), (2,5))

((movie3,movie5), (3,5))

Reducer Output (similarity measure of movie pairs):

((movie1,movie2), 0.86)

((movie1,movie3), 0.39)

((movie2,movie3), 0.81)

((movie2,movie5), 0.49)

Names: Bob Skowron, Jason Walker

Keys: rskowron, jwalker

SVN: jwalker: https://svn.seas.wustl.edu/repositories/jwalker/cse427s_f17/

((movie3,movie5), 0.83)