CSE427s Homework 7

Adrien Xie, Guangda Ouyang

TOTAL POINTS

73 / 100

QUESTION 1

Computing Word Co-Occurrence 50 pts

1.1 Implementing Co-Occurence 20 / 20

✓ + **6 pts** **Correct $$\small\textsf{WordCo.java}$$**

✓ + **2 pts** **Student comments in $$\small\textsf{WordCo.java}$$**

✓ + **10 pts** **Correct $$\small\textsf{WordCoMapper.java}$$**

✓ + **2 pts** **Student comments in $$\small\textsf{WordCoMapper.java}$$**

+ **7 pts** Partially correct WordCoMapper.java
- **1.5 pts** Pages not selected
+ **0 pts** No WUSTL key or missing files
+ **0 pts** No credit
+ **0 pts** Graded by ET

1.2 Checking the Output 5 / 5

✓ + **1 pts** **Correct number of word pairs: $$\small298,917$$**

✓ + **1 pts** **Correct number of pairs $$\small\text{the,lovers}$$: $$\small3$$**

✓ + **1 pts** **Correct number of pairs $$\small\text{loved,you}$$: $$\small15$$**

✓ + **1 pts** **Correct number of pairs $$\small\text{you,loved}$$: $$\small6$$**

✓ + **1 pts** **Correct number of pairs $$\small\text{verona,julia}$$: $$\small2$$**

- **0.5 pts** No pages selected
+ **0 pts** No credit

1.3 Communication Costs 5 / 5

✓ + **5 pts** **Correct communication cost: $$\small 1,012,561$$**

- **0.5 pts** Pages not selected
+ **0 pts** No credit

1.4 The Drawbacks 10 / 10

+ **5 pts** Partially Correct
- **1 pts** Pages not selected
+ **0 pts** No credit

✓ + **10 pts** **Correct drawback: it fails to model the complete pattern of linguistic semantics, ie. the context window might be too small compared to tri-gram or n-gram.**

1.5 Stripes 10 / 10

✓ + **5 pts** **Correct communication cost: $$k$$**

+ **3 pts** Partially correct communication cost

✓ + **5 pts** **Correct explanation of effect on memory usage**

+ **3 pts** Partially correct explanation of effect on memory usage
+ **0 pts** No Credit
- **1 pts** Page not selected

QUESTION 2

Collaborative Filtering: Similarity Measures 20 pts

2.1 Pearson Correlation 5 / 10

+ **10 pts** totally correct with proof and provide math equation

✓ + **5 pts** **not efficient proof with math equation**

+ **2 pts** partially correct
+ **0 pts** no credit
+ **0 pts** Grade by YQ

2.2 Quality vs. Feasibility 5 / 5

✓ + **2.5 pts** **benefit Correct**

✓ + **2.5 pts** **Disadvantage Correct**

+ **0 pts** No credict
+ **0 pts** grade by YQ

2.3 Jaccard Similarity 5 / 5

✓ + **0 pts** **Graded by GD**

✓ + **5 pts** **Correct**

+ **4 pts** Mostly correct

+ **2.5 pts** Partially correct

QUESTION 3

Collaborative Filtering in MapReduce 30 pts

3.1 Benefits 5 / 5

✓ + **2.5 pts** **Correct benefit (1/2). Benefits of the dual approach include: being more efficient to compute if there are significantly more users than items, reducing the bias of user comparison since classifications for items tend to be more well-defined leading to more accurate predictions, finding similar items is easier than finding similar people**

+ **1.5 pts** Partially correct benefit (1/2)

+ **0 pts** Incorrect benefit (1/2)

✓ + **2.5 pts** **Correct benefit (2/2)**

+ **1.5 pts** Partially correct benefit (2/2)

+ **0 pts** Incorrect benefit (2/2)

- **0.5 pts** Pages not selected

+ **0 pts** No credit

✓ + **0 pts** **Graded by CS**

3.2 Implementing your Program 3 / 25

+ **5 pts** Correct first mapper output: $$\small\textsf{(user-id, (movie-id, rating))}$$

+ **3 pts** Partially correct first mapper output

+ **5 pts** Correct first reducer input: $$\small\textsf{(user-id, [(movie-id, rating), (movie-id, rating), \ldots])}$$

+ **3 pts** Partially correct first reducer input

+ **5 pts** Correct first reducer output: $$\small\textsf{(movie-id, movie-id), (} \mathbf{r_1, r_2, r_1 r_2, r_1^2, r_2^2}\textsf{)}$$

+ **3 pts** Partially correct first reducer output

+ **5 pts** Correct second reducer input: $$\small\textsf{((movie-id, movie-id), [(}\mathbf{r_1, r_2, r_1 r_2, r_1^2, r_2^2}\textsf{), \ldots])}$$

+ **3 pts** Partially correct second reducer input

+ **5 pts** Correct second reducer output: $$\small\textsf{((movie-id, movie-id), cosine\_similarity)}$$

✓ + **3 pts** **Partially correct second reducer output**

- **2.5 pts** Pages not selected

+ **0 pts** No credit

✓ + **0 pts** **graded by CS**

You do not show any of the inputs or outputs for the MR jobs.

Problem 1

(b) By using “wc -l part-r-00000” and “grep pattern part-r-00000” commands, we have:

- 298917 pairs.
- the,lovers 3
- loved,you 15
- you,loved 6
- verona,julia 2

(c)

communication cost = map input records + reduce input records

= 173126 + 839435 = 1012561

Overview
Counters
Configuration
Map tasks
Reduce tasks

Tools

File System Counters	FILE: Number of bytes read	0	13452222	13452222
	FILE: Number of bytes written	14025780	13595552	27621332
	FILE: Number of large read operations	0	0	0
	FILE: Number of read operations	0	0	0
	FILE: Number of write operations	0	0	0
	HDFS: Number of bytes read	5284754	0	5284754
	HDFS: Number of bytes written	0	4124037	4124037
	HDFS: Number of large read operations	0	0	0
	HDFS: Number of read operations	12	3	15
	HDFS: Number of write operations	0	2	2
	Name	Map	Reduce	Total
Job Counters	Data-local map tasks	0	0	5
	Killed map tasks	0	0	1
	Launched map tasks	0	0	5
	Launched reduce tasks	0	0	1
	Total megabyte-milliseconds taken by all map tasks	0	0	230532096
	Total megabyte-milliseconds taken by all reduce tasks	0	0	36761600
	Total time spent by all map tasks (ms)	0	0	225129
	Total time spent by all maps in occupied slots (ms)	0	0	225129
	Total time spent by all reduce tasks (ms)	0	0	35900
	Total time spent by all reduces in occupied slots (ms)	0	0	35900
	Total vcore-milliseconds taken by all map tasks	0	0	225129
	Total vcore-milliseconds taken by all reduce tasks	0	0	35900
	Name	Map	Reduce	Total
Map-Reduce Framework	Combine input records	0	0	0
	Combine output records	0	0	0
	CPU time spent (ms)	8960	3270	12230
	Failed Shuffles	0	0	0
	GC time elapsed (ms)	2594	411	3005
	Input split bytes	523	0	523
	Map input records	173126	0	173126
	Map output bytes	11773346	0	11773346
	Map output materialized bytes	13452240	0	13452240
	Map output records	839435	0	839435
	Merged Map outputs	0	4	4
	Physical memory (bytes) snapshot	830238720	150646784	980885504
	Reduce input groups	0	298917	298917
	Reduce input records	0	839435	839435
	Reduce output records	0	298917	298917
	Reduce shuffle bytes	0	13452240	13452240
	Shuffled Maps	0	4	4
	Spilled Records	839435	839435	1678870
	Total committed heap usage (bytes)	662454272	60751872	723206144
	Virtual memory (bytes) snapshot	6017675264	1511354368	7529029632
	Name	Map	Reduce	Total

1.1 Implementing Co-Occurence 20 / 20

- ✓ **+ 6 pts** **Correct $$\small\textsf{WordCo.java}$$**
- ✓ **+ 2 pts** **Student comments in $$\small\textsf{WordCo.java}$$**
- ✓ **+ 10 pts** **Correct $$\small\textsf{WordCoMapper.java}$$**
- ✓ **+ 2 pts** **Student comments in $$\small\textsf{WordCoMapper.java}$$**
- **+ 7 pts** Partially correct WordCoMapper.java
- **- 1.5 pts** Pages not selected
- **+ 0 pts** No WUSTL key or missing files
- **+ 0 pts** No credit
- **+ 0 pts** Graded by ET

Problem 1

(b) By using "wc -l part-r-00000" and "grep pattern part-r-00000" commands, we have:

- 298917 pairs.
- the,lovers 3
- loved,you 15
- you,loved 6
- verona,julia 2

(c)

communication cost = map input records + reduce input records

= 173126 + 839435 = 1012561

Overview
Counters
Configuration
Map tasks
Reduce tasks
Tools

	Name	Map	Reduce	Total
File System Counters	FILE: Number of bytes read	0	13452222	13452222
	FILE: Number of bytes written	14025780	13595552	27621332
	FILE: Number of large read operations	0	0	0
	FILE: Number of read operations	0	0	0
	FILE: Number of write operations	0	0	0
	HDFS: Number of bytes read	5284754	0	5284754
	HDFS: Number of bytes written	0	4124037	4124037
	HDFS: Number of large read operations	0	0	0
	HDFS: Number of read operations	12	3	15
	HDFS: Number of write operations	0	2	2
	Name	Map	Reduce	Total
Job Counters	Data-local map tasks	0	0	5
	Killed map tasks	0	0	1
	Launched map tasks	0	0	5
	Launched reduce tasks	0	0	1
	Total megabyte-milliseconds taken by all map tasks	0	0	230532096
	Total megabyte-milliseconds taken by all reduce tasks	0	0	36761600
	Total time spent by all map tasks (ms)	0	0	225129
	Total time spent by all maps in occupied slots (ms)	0	0	225129
	Total time spent by all reduce tasks (ms)	0	0	35900
	Total time spent by all reduces in occupied slots (ms)	0	0	35900
	Total vcore-milliseconds taken by all map tasks	0	0	225129
	Total vcore-milliseconds taken by all reduce tasks	0	0	35900
	Name	Map	Reduce	Total
Map-Reduce Framework	Combine input records	0	0	0
	Combine output records	0	0	0
	CPU time spent (ms)	8960	3270	12230
	Failed Shuffles	0	0	0
	GC time elapsed (ms)	2594	411	3005
	Input split bytes	523	0	523
	Map input records	173126	0	173126
	Map output bytes	11773346	0	11773346
	Map output materialized bytes	13452240	0	13452240
	Map output records	839435	0	839435
	Merged Map outputs	0	4	4
	Physical memory (bytes) snapshot	830238720	150646784	980885504
	Reduce input groups	0	298917	298917
	Reduce input records	0	839435	839435
	Reduce output records	0	298917	298917
	Reduce shuffle bytes	0	13452240	13452240
	Shuffled Maps	0	4	4
	Spilled Records	839435	839435	1678870
	Total committed heap usage (bytes)	662454272	60751872	723206144
	Virtual memory (bytes) snapshot	6017675264	1511354368	7529029632
	Name	Map	Reduce	Total

1.2 Checking the Output 5 / 5

✓ + 1 pts Correct number of word pairs: $$\small298,917$$

✓ + 1 pts Correct number of pairs $$\small\text{the,lovers}$$: $$\small3$$

✓ + 1 pts Correct number of pairs $$\small\text{loved,you}$$: $$\small15$$

✓ + 1 pts Correct number of pairs $$\small\text{you,loved}$$: $$\small6$$

✓ + 1 pts Correct number of pairs $$\small\text{verona,julia}$$: $$\small2$$

- 0.5 pts No pages selected

+ 0 pts No credit

Problem 1

(b) By using “wc -l part-r-00000” and “grep pattern part-r-00000” commands, we have:

- 298917 pairs.
- the,lovers 3
- loved,you 15
- you,loved 6
- verona,julia 2

(c)

communication cost = map input records + reduce input records

= 173126 + 839435 = 1012561

Overview
Counters
Configuration
Map tasks
Reduce tasks

Tools

Group	Name	Map	Reduce	Total
File System Counters	FILE: Number of bytes read	0	13452222	13452222
	FILE: Number of bytes written	14025780	13595552	27621332
	FILE: Number of large read operations	0	0	0
	FILE: Number of read operations	0	0	0
	FILE: Number of write operations	0	0	0
	HDFS: Number of bytes read	5284754	0	5284754
	HDFS: Number of bytes written	0	4124037	4124037
	HDFS: Number of large read operations	0	0	0
	HDFS: Number of read operations	12	3	15
	HDFS: Number of write operations	0	2	2
	Name	Map	Reduce	Total
Job Counters	Data-local map tasks	0	0	5
	Killed map tasks	0	0	1
	Launched map tasks	0	0	5
	Launched reduce tasks	0	0	1
	Total megabyte-milliseconds taken by all map tasks	0	0	230532096
	Total megabyte-milliseconds taken by all reduce tasks	0	0	36761600
	Total time spent by all map tasks (ms)	0	0	225129
	Total time spent by all maps in occupied slots (ms)	0	0	225129
	Total time spent by all reduce tasks (ms)	0	0	35900
	Total time spent by all reduces in occupied slots (ms)	0	0	35900
	Total vcore-milliseconds taken by all map tasks	0	0	225129
	Total vcore-milliseconds taken by all reduce tasks	0	0	35900
	Name	Map	Reduce	Total
Map-Reduce Framework	Combine input records	0	0	0
	Combine output records	0	0	0
	CPU time spent (ms)	8960	3270	12230
	Failed Shuffles	0	0	0
	GC time elapsed (ms)	2594	411	3005
	Input split bytes	523	0	523
	Map input records	173126	0	173126
	Map output bytes	11773346	0	11773346
	Map output materialized bytes	13452240	0	13452240
	Map output records	839435	0	839435
	Merged Map outputs	0	4	4
	Physical memory (bytes) snapshot	830238720	150646784	980885504
	Reduce input groups	0	298917	298917
	Reduce input records	0	839435	839435
	Reduce output records	0	298917	298917
	Reduce shuffle bytes	0	13452240	13452240
	Shuffled Maps	0	4	4
	Spilled Records	839435	839435	1678870
	Total committed heap usage (bytes)	662454272	60751872	723206144
	Virtual memory (bytes) snapshot	6017675264	1511354368	7529029632
	Name	Map	Reduce	Total

1.3 Communication Costs 5 / 5

✓ + **5 pts** **Correct communication cost: $$\small 1,012,561$$**

- **0.5 pts** Pages not selected

+ **0 pts** No credit

(d)

Example 1: Paris is the capital of France.

Example 2: Ronaldo joins Juventus.

In English, most useful information of a sentence are usually presented by the subject and the object, while they are usually separated (by verb and other things).

We can easily eye-ball that the meaningful co-occurrence we want to learn are (Paris, capital), (Paris, France) in the first example, and (Ronaldo, Juventus) in the second examples. Merely counting the co-occurrence of words directly next to each other would miss both of this pairs, meanwhile, collecting some meaningless pairs such as (Paris, is) and (is, the).

(e)

When k is the size of the vocabulary, the communication costs are $\frac{k^2-k}{2}$ when using pairs and k – 1 when using stripes. The memories needed in reduce() fuction are 2 ids + 1 count * m for pairs and 1 id + (k – 1 ids and counts) * m when using stripes. (m is the total number of users)

We can see that the communication cost for pairs are significantly higher than stripes' as k gets larger since it's a quadratic verses linear function. On the other hand, pairs requires much less memory, especially when m gets big because of the difference between the coefficients before m.

1.4 The Drawbacks 10 / 10

+ **5 pts** Partially Correct
- **1 pts** Pages not selected
+ **0 pts** No credit

✓ + **10 pts** **Correct drawback: it fails to model the complete pattern of linguistic semantics, ie. the context window might be too small compared to tri-gram or n-gram.**

(d)

Example 1: Paris is the capital of France.

Example 2: Ronaldo joins Juventus.

In English, most useful information of a sentence are usually presented by the subject and the object, while they are usually separated (by verb and other things).

We can easily eye-ball that the meaningful co-occurrence we want to learn are (Paris, capital), (Paris, France) in the first example, and (Ronaldo, Juventus) in the second examples. Merely counting the co-occurrence of words directly next to each other would miss both of this pairs, meanwhile, collecting some meaningless pairs such as (Paris, is) and (is, the).

(e)

When k is the size of the vocabulary, the communication costs are $\frac{k^2-k}{2}$ when using pairs and k – 1 when using stripes. The memories needed in reduce() fuction are 2 ids + 1 count * m for pairs and 1 id + (k – 1 ids and counts) * m when using stripes. (m is the total number of users)

We can see that the communication cost for pairs are significantly higher than stripes' as k gets larger since it's a quadratic verses linear function. On the other hand, pairs requires much less memory, especially when m gets big because of the difference between the coefficients before m.

1.5 Stripes 10 / 10

- ✓ **+ 5 pts** **Correct communication cost: $$k$$**
- **+ 3 pts** Partially correct communication cost
- ✓ **+ 5 pts** **Correct explanation of effect on memory usage**
- **+ 3 pts** Partially correct explanation of effect on memory usage
- **+ 0 pts** No Credit
- **- 1 pts** Page not selected

Problem 2

(a)

Cosine similarity = C(x,y) = cos(r_x, r_y) = $\frac{\sum_{s \in S_{xy}} r_{xs} r_{ys}}{\sqrt{\sum_{s \in S_{xy}} r_{xs}^2}\sqrt{\sum_{s \in S_{xy}} r_{ys}^2}}$

If we normalize vectors by subtracting the vector means, i.e., substituting r_{xs} with $(r_{xs} - \overline{r_x})$ and substituting r_{ys} with $(r_{ys} - \overline{r_y})$, we get the Pearson correlation:

Pearson correlation = P(x,y) = $\frac{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})(r_{ys} - \overline{r_y})}{\sqrt{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})^2}\sqrt{\sum_{s \in S_{xy}} (r_{ys} - \overline{r_y})^2}}$

(b)

- Pearson correlation removes bias because of normalization (subtracting means).
- Computing Pearson correlation has more complex processes (computing the mean and subtract it) so the implementation is harder; also, we have to store all the vectors in order to compute the mean, requiring more memory.

(c)

- Easy to implement.
- Only tells similarity of two users' watch lists. Does not tell anything about the ratings.
- Applying a threshold. Divide the data into high ratings (>3) and low ratings (<3). We maybe want to ignore ratings lower than 3.

2.1 Pearson Correlation **5 / 10**

+ **10 pts** totally correct with proof and provide math equation
✓ + **5 pts** **not efficient proof with math equation**
+ **2 pts** partially correct
+ **0 pts** no credit
+ **0 pts** Grade by YQ

Problem 2

(a)

Cosine similarity = C(x,y) = cos(r_x, r_y) = $\frac{\sum_{s\in S_{xy}} r_{xs} r_{ys}}{\sqrt{\sum_{s\in S_{xy}} r_{xs}^2}\sqrt{\sum_{s\in S_{xy}} r_{ys}^2}}$

If we normalize vectors by subtracting the vector means, i.e., substituting r_{xs} with $(r_{xs} - \overline{r_x})$ and substituting r_{ys} with $(r_{ys} - \overline{r_y})$, we get the Pearson correlation:

Pearson correlation = P(x,y) = $\frac{\sum_{s\in S_{xy}} (r_{xs} - \overline{r_x})(r_{ys} - \overline{r_y})}{\sqrt{\sum_{s\in S_{xy}} (r_{xs} - \overline{r_x})^2}\sqrt{\sum_{s\in S_{xy}} (r_{ys} - \overline{r_y})^2}}$

(b)

- Pearson correlation removes bias because of normalization (subtracting means).
- Computing Pearson correlation has more complex processes (computing the mean and subtract it) so the implementation is harder; also, we have to store all the vectors in order to compute the mean, requiring more memory.

(c)

- Easy to implement.
- Only tells similarity of two users' watch lists. Does not tell anything about the ratings.
- Applying a threshold. Divide the data into high ratings (>3) and low ratings (<3). We maybe want to ignore ratings lower than 3.

2.2 Quality vs. Feasibility 5 / 5

- ✓ **+ 2.5 pts** **benefit Correct**
- ✓ **+ 2.5 pts** **Disadvantage Correct**
- **+ 0 pts** No credict
- **+ 0 pts** grade by YQ

Problem 2

(a)

Cosine similarity = C(x,y) = cos(r_x, r_y) = $\frac{\sum_{s \in S_{xy}} r_{xs} r_{ys}}{\sqrt{\sum_{s \in S_{xy}} r_{xs}^2}\sqrt{\sum_{s \in S_{xy}} r_{ys}^2}}$

If we normalize vectors by subtracting the vector means, i.e., substituting r_{xs} with $(r_{xs} - \overline{r_x})$ and substituting r_{ys} with $(r_{ys} - \overline{r_y})$, we get the Pearson correlation:

Pearson correlation = P(x,y) = $\frac{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})(r_{ys} - \overline{r_y})}{\sqrt{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})^2}\sqrt{\sum_{s \in S_{xy}} (r_{ys} - \overline{r_y})^2}}$

(b)

- Pearson correlation removes bias because of normalization (subtracting means).
- Computing Pearson correlation has more complex processes (computing the mean and subtract it) so the implementation is harder; also, we have to store all the vectors in order to compute the mean, requiring more memory.

(c)

- Easy to implement.
- Only tells similarity of two users' watch lists. Does not tell anything about the ratings.
- Applying a threshold. Divide the data into high ratings (>3) and low ratings (<3). We maybe want to ignore ratings lower than 3.

2.3 Jaccard Similarity 5 / 5

✓ + **0 pts** **Graded by GD**

✓ + **5 pts** **Correct**

+ **4 pts** Mostly correct

+ **2.5 pts** Partially correct

Problem 3

(a)

- When having more users than items, it saves computation since computing similarities between users would be expensive.
- Does not have to rebuild everything when users' list changes.

(b)

	User1	User2	User3
Movie1	1		1
Movie2	3	2	2
Movie3	2	3	
Movie4			
Movie5		5	

cos(movie1, movie2) = $\frac{1*3+1*2}{\sqrt{1^2+1^2}\sqrt{3^2+2^2}}$ = $5/\sqrt{2*13}$ = 0.98

cos(movie1, movie3) = $\frac{1*2}{\sqrt{1^2}\sqrt{2^2}}$ = $2/\sqrt{4}$ = 1

cos(movie1, movie4) = 0

cos(movie1, movie5) = 0

cos(movie2, movie3) = $\frac{3*2+2*3}{\sqrt{3^2+2^2}\sqrt{2^2+3^2}}$ = $12/\sqrt{13*13}$ = 0.923

cos(movie2, movie4) = 0

cos(movie2, movie5) = $\frac{2*5}{\sqrt{2^2}\sqrt{5^2}}$ = $10/\sqrt{4*25}$ = 1

cos(movie3, movie4) = 0

cos(movie3, movie5) = $\frac{3*5}{\sqrt{3^2}\sqrt{5^2}}$ = 1

cos(movie4, movie5) = 0

3.1 Benefits 5 / 5

✓ + **2.5 pts** **Correct benefit (1/2). Benefits of the dual approach include: being more efficient to compute if there are significantly more users than items, reducing the bias of user comparison since classifications for items tend to be more well-defined leading to more accurate predictions, finding similar items is easier than finding similar people**

+ **1.5 pts** Partially correct benefit (1/2)
+ **0 pts** Incorrect benefit (1/2)

✓ + **2.5 pts** **Correct benefit (2/2)**

+ **1.5 pts** Partially correct benefit (2/2)
+ **0 pts** Incorrect benefit (2/2)
- **0.5 pts** Pages not selected
+ **0 pts** No credit

✓ + **0 pts** **Graded by CS**

Problem 3

(a)

- When having more users than items, it saves computation since computing similarities between users would be expensive.
- Does not have to rebuild everything when users' list changes.

(b)

	User1	User2	User3
Movie1	1		1
Movie2	3	2	2
Movie3	2	3	
Movie4			
Movie5		5	

cos(movie1, movie2) = $\frac{1*3+1*2}{\sqrt{1^2+1^2}\sqrt{3^2+2^2}}$ = $5/\sqrt{2*13}$ = 0.98

cos(movie1, movie3) = $\frac{1*2}{\sqrt{1^2}\sqrt{2^2}}$ = $2/\sqrt{4}$ = 1

cos(movie1, movie4) = 0

cos(movie1, movie5) = 0

cos(movie2, movie3) = $\frac{3*2+2*3}{\sqrt{3^2+2^2}\sqrt{2^2+3^2}}$ = $12/\sqrt{13*13}$ = 0.923

cos(movie2, movie4) = 0

cos(movie2, movie5) = $\frac{2*5}{\sqrt{2^2}\sqrt{5^2}}$ = $10/\sqrt{4*25}$ = 1

cos(movie3, movie4) = 0

cos(movie3, movie5) = $\frac{3*5}{\sqrt{3^2}\sqrt{5^2}}$ = 1

cos(movie4, movie5) = 0

3.2 Implementing your Program 3 / 25

+ **5 pts** Correct first mapper output: $$\small\textsf{(user-id, (movie-id, rating))}$$
+ **3 pts** Partially correct first mapper output
+ **5 pts** Correct first reducer input: $$\small\textsf{(user-id, [(movie-id, rating), (movie-id, rating), \ldots])}$$
+ **3 pts** Partially correct first reducer input
+ **5 pts** Correct first reducer output: $$\small\textsf{(movie-id, movie-id), (} \mathbf{r_1, r_2, r_1 r_2, r_1^2, r_2^2}\textsf{)}$$
+ **3 pts** Partially correct first reducer output
+ **5 pts** Correct second reducer input: $$\small\textsf{((movie-id, movie-id), [(}\mathbf{r_1, r_2, r_1 r_2, r_1^2, r_2^2}\textsf{), \ldots])}$$
+ **3 pts** Partially correct second reducer input
+ **5 pts** Correct second reducer output: $$\small\textsf{((movie-id, movie-id), cosine\_similarity)}$$

✓ + **3 pts** **Partially correct second reducer output**

- **2.5 pts** Pages not selected

+ **0 pts** No credit

✓ + **0 pts** **graded by CS**

You do not show any of the inputs or outputs for the MR jobs.