DSA5101 Exercises

Zhao Peiduo

AY2025/26 Sem1

Disclaimer: Answers could be based on ChatGPT results which may be inaccurate / incorrect. I would like to thank Angel for reporting a significant number of errors in the original version of this document.

Lecture 1: Frequent Itemsets and Association Rules

Question 1

Given the following transaction dataset:

TID	**Items**
1	{Bread, Coke, Milk}
2	{Beer, Bread}
3	{Beer, Coke, Diaper, Milk}
4	{Beer, Bread, Diaper, Milk}
5	{Coke, Diaper, Milk}

1. Generate the support of all single items, and all **pairs that contain Bread**.
2. Compute the confidence and interest of the rule {Milk, Bread} → Coke.
3. Given a support threshold $s = 3$, determine the frequent pairs using the Apriori algorithm.
4. Generate the association rules from the frequent pairs.

Relevant Concepts

Support: Number of transactions containing the itemset I.

$$\text{support}(I) = |\{\text{transactions containing } I\}|$$

Frequent Itemset: An itemset is frequent if $\text{support}(I) \geq s$.
Confidence: For rule $I \to j$,

$$\text{conf}(I \to j) = \frac{\text{support}(I \cup \{j\})}{\text{support}(I)}$$

Interest:

$$\text{Interest}(I \to j) = |\text{conf}(I \to j) - Pr[j]|$$

Apriori Algorithm (Key Idea): If an itemset is frequent, then **all of its subsets must also be frequent**. Thus, we build larger itemsets level-by-level and prune any itemsets containing infrequent subsets.

Answer

(1) Support of Single Items and Pairs Containing Bread
Single items:

$$\text{Bread} = 3,\ \text{Coke} = 3,\ \text{Milk} = 4,\ \text{Beer} = 3,\ \text{Diaper} = 3$$

Pairs **containing Bread**:

$$\text{(Bread, Milk)} = 2, \quad \text{(Bread, Coke)} = 1, \quad \text{(Bread, Beer)} = 2, \quad \text{(Bread, Diaper)} = 1$$

(2) Confidence and Interest of {Milk, Bread} → Coke

$$\text{support}(\{\text{Milk, Bread}\}) = 2, \quad \text{support}(\{\text{Milk, Bread, Coke}\}) = 1$$

$$\text{confidence} = \frac{1}{2} = 0.5$$

$$Pr[\text{Coke}] = \frac{3}{5} = 0.6$$

$$\text{Interest} = |0.5 - 0.6| = 0.1$$

(3) Apriori Algorithm (Support Threshold $s = 3$)
Pass 1: Find frequent single items

$$L_1 = \{\text{Milk} = 4,\ \text{Bread} = 3,\ \text{Beer} = 3,\ \text{Coke} = 3,\ \text{Diaper} = 3\}$$

(All are frequent since each ≥ 3)
Generate candidate pairs C_2 from L_1:

Candidate Pair	Support Count
(Bread, Coke)	1
(Bread, Milk)	2
(Bread, Beer)	2
(Bread, Diaper)	1
(Coke, Milk)	3
(Coke, Beer)	1
(Coke, Diaper)	2
(Milk, Beer)	2
(Milk, Diaper)	3
(Beer, Diaper)	2

Compute support of each pair (from transactions):

$$(\text{Coke, Milk}) = 3, \quad (\text{Milk, Diaper}) = 3$$

All other pairs have support < 3 and are pruned. Therefore the frequent pairs are:

$$L_2 = \{\{\text{Coke, Milk}\},\ \{\text{Milk, Diaper}\}\}$$

(4) Association Rules
From {Coke, Milk}:

$$\text{Coke} \to \text{Milk (1.0)}, \qquad \text{Milk} \to \text{Coke (0.75)}$$

From {Milk, Diaper}:

$$\text{Milk} \to \text{Diaper (0.75)}, \qquad \text{Diaper} \to \text{Milk (1.0)}$$

Question 2

Consider the same transaction dataset and support threshold $s = 3$. Use the precomputed pair counts below to illustrate **PCY**, the **multistage refinement**, and the **multihash** refinement. Assume we index items as Bread = 1, Coke = 2, Milk = 3, Beer = 4, Diaper = 5.

Pair	Support Count
(Bread, Coke)	1
(Bread, Milk)	2
(Bread, Beer)	2
(Bread, Diaper)	1
(Coke, Milk)	3
(Coke, Beer)	1
(Coke, Diaper)	2
(Milk, Beer)	2
(Milk, Diaper)	3
(Beer, Diaper)	2

1. Apply **PCY** with $B = 4$ buckets and hash $h(i, j) = (i + j) \bmod 4$. Show bucket counts, the first bitmap, and the resulting candidate pairs.
2. Apply the **multistage** refinement with a second independent hash $g(i, j) = (2i + 3j) \bmod 4$. Show the second bucket counts/bitmap and the final candidate set before counting.
3. Apply the **multihash** refinement using two hash functions in Pass 1 and explain the pruning effect.

Relevant Concepts

PCY (Park–Chen–Yu): During Pass 1 (when counting singletons), also hash every *occurrence* of a pair $\{i, j\}$ into a bucket and count bucket totals. Buckets whose totals are $\geq s$ are marked *frequent* in a bitmap. In Pass 2, a pair $\{i, j\}$ is a *candidate* only if:

- Both i and j are frequent singletons, and
- The pair hashes to a bucket marked 1 in the bitmap.

Multistage Refinement: Add a second hash and bitmap. Rehash only pairs that pass PCY. In Pass 3, count only pairs that hash to *frequent* buckets under *both* bitmaps.
Multihash Refinement: Instead of deferring the second hash to Pass 2/3, apply multiple hash functions *during Pass 1*. A pair remains a candidate only if it hashes to a *frequent* bucket in *both* hash tables.

Answer

(1) PCY with $B = 4$, $h(i, j) = (i + j) \bmod 4$ where i and j are indices. Therefore the mappings are:

- Bucket 0: (Bread, Milk), (Milk, Diaper)
- Bucket 1: (Coke, Milk), (Bread, Beer), (Beer, Diaper)
- Bucket 2: (Bread, Diaper), (Coke, Beer)
- Bucket 3: (Bread, Coke), (Coke, Diaper), (Milk Beer)

And we can get the count table below:

Bucket	0	1	2	3
Count	5	7	2	5

Given support threshold of 3, bucket 2 is infrequent and can be pruned. We can eliminate (Bread, Diaper) and (Coke, Beer).

$$\text{Bitmap}_1 = [1, 1, 0, 1]$$

PCY candidate pairs:

(Bread, Coke), (Bread, Milk), (Bread, Beer), (Coke, Milk), (Coke, Diaper), (Milk, Beer), (Milk, Diaper), (Beer, Diaper)

(2) Multistage: second hash $g(i,j) = (2i + 3j) \bmod 4$
The possible bucket assignments under the second hash function are:

- Bucket 0: (Bread, Coke)
- Bucket 1: (Coke, Milk), (Milk, Diaper)
- Bucket 2: (Bread, Beer), (Milk, Beer)
- Bucket 3: (Bread, Milk), (Coke, Diaper), (Beer, Diaper)

Bucket	0	1	2	3
Count under g	1	6	4	6

$$\text{Bitmap}_2 = [0, 1, 1, 1]$$

Therefore bucket 0 is pruned and (Bread, Coke) is removed.
Final multistage candidate set:

{(Bread, Milk), (Bread, Beer), (Coke, Milk), (Coke, Diaper), (Milk, Beer), (Milk, Diaper), (Beer, Diaper)}

(3) Multihash Refinement
Use two hash functions in Pass 1:

$$h_1(i,j) = (i + j) \bmod 4, \quad h_2(i,j) = (2i + 3j) \bmod 4.$$

A pair becomes a candidate only if it hashes to *frequent buckets in both bitmaps*:

$$\text{Keep } (i,j) \text{ only if } \text{Bitmap}_1[h_1(i,j)] = 1 \text{ and } \text{Bitmap}_2[h_2(i,j)] = 1.$$

Essentially, all non-frequent pairs are pruned in one pass: (Bread, Diaper) and (Coke, Beer) are pruned from bitmap1 and (Bread, Coke) and (Coke, Beer) are pruned from bitmap2. (Coke, Beer) is not frequent for either hash function.
Comparison with Apriori: All methods return the same frequent pairs, but PCY, multistage, and multihash *reduce the number of candidate pairs stored and counted.*

Question3

For the same transaction dataset and minimum support threshold of $s = 3$:

TID	**Items**
1	{Bread, Coke, Milk}
2	{Beer, Bread}
3	{Beer, Coke, Diaper, Milk}
4	{Beer, Bread, Diaper, Milk}
5	{Coke, Diaper, Milk}

We want to find frequent itemsets using Random Sampling, SON Algorithm and Toivonen's Algorithm. Suppose we sample 1,3,5 for random sampling and Toivonen's Algorithm, and split the transactions into two datasets: {1, 2, 3} and {4, 5}, trace the algorithm.

Relevant Concepts

Random Sampling: Run Apriori on a randomly selected sample. May miss real frequent itemsets (false negatives) because some may not appear frequently enough in the sample.
SON Algorithm: Break data into chunks, find locally frequent itemsets, union them, then verify globally. No false negatives, but intermediate false positives are removed in verification.
Toivonen's Algorithm: Sample with a slightly lowered threshold and track a **negative border**. If any negative-border itemset is actually frequent, repeat sampling. Final output contains no false positives or false negatives.

Answer

(1) Random Sampling Example
Choose a random sample of size 3 (i.e., $p = 3/5 = 0.6$). Suppose we sample TIDs: 1, 3, 5.

$$\begin{array}{c|l} 1 & \{\text{Bread, Coke, Milk}\} \\ 3 & \{\text{Beer, Coke, Diaper, Milk}\} \\ 5 & \{\text{Coke, Diaper, Milk}\} \end{array}$$

Scaled support threshold $= ps = 0.6 \cdot 3 = 1.8 \approx 2$.
In sample:

$$\text{Milk} = 3,\ \text{Coke} = 3,\ \text{Diaper} = 2,\ \text{Beer} = 1,\ \text{Bread} = 1 \Rightarrow \mathit{SampleFrequent} : \{\text{Milk, Coke, Diaper}\}.$$

False Negative: Bread and Beer are actually frequent globally but rejected here.
(2) SON Algorithm Example
Split data into two chunks:

$$\text{Chunk 1: TIDs } 1, 2, 3 \qquad \text{Chunk 2: TIDs } 4, 5$$

Scaled threshold in each chunk $= ps$.
Chunk 1 has 3 transactions, full dataset has 5 $\rightarrow p = 3/5 = 0.6$.

$$ps = 0.6 \cdot 3 = 1.8 \approx 2.$$

Chunk 1 frequent: {Milk, Coke, Bread, Beer}
Chunk 2 has 2 transactons.

$$ps = 0.6 \cdot 2 = 1.2 \approx 1.$$

Chunk 2 frequent: {Milk, Coke, Bread, Beer, Diaper}
Second pass (global count) confirms all 5 are frequent. Therefore, no false negatives. False positives removed in pass 2.
(3) Toivonen's Algorithm Example
Sample same subset as in Random Sampling (TIDs 1,3,5). Use reduced threshold (e.g., $0.8ps = 0.8 \cdot 2 = 1.6 \approx 2$). So same frequent sample set:

$$\text{Frequent in sample: } \{\text{Milk, Coke, Diaper}\}$$

Negative Border: Itemsets not frequent in sample but all subsets frequent: {Bread}, {Beer}
Check these in full dataset: Both have support 3, so they are actually frequent. This triggers a resample and rerun. On the second sample we will recover all five frequent items by luck or by taking a larger sample. Therefore, there will be no false positives and no false negatives.

Final Comparison

Method	**Passes Needed**	**False Negatives?**	**False Positives?**
Random Sampling	1 or 2	**Yes**	*No* (if verified)
SON Algorithm	2	*No*	*Yes* (pruned in pass 2)
Toivonen's Algorithm	2 typically	*No*	*No*

Lecture 2: Finding Similar items: Locality Sensitivity Hashing

Question 1

We consider three short documents:

$$D_1 = \text{"ABABA"}, \qquad D_2 = \text{"ABCA"}, \qquad D_3 = \text{"BABA"}$$

Relevant Concepts

2-shingles: all length-2 character substrings (set semantics).
Min-hash $h_\pi(C)$: for permutation π of row indices, $h_\pi(C)$ is the *position in* π of the first row (scanning top-to-bottom in the permuted order) where column C has a 1.
Jaccard similarity: $\text{sim}(A, B) = \dfrac{|A \cap B|}{|A \cup B|}$.
Signature similarity (sig/sig): fraction of rows in which two signature columns agree.
LSH banding (AND-OR vs. OR-AND): With bands of size r, *AND-OR* declares a candidate if two columns are identical within *any* band (AND inside band, OR across bands). *OR-AND* requires matching in *all* bands (OR inside band, AND across bands).

Answer

(1) 2-shingles and input matrix.

$$D_1 : \{AB, BA\}, \quad D_2 : \{AB, BC, CA\}, \quad D_3 : \{AB, BA\}$$
$$S = \{AB, BA, BC, CA\} \ \ (1\text{: } AB,\ 2\text{: } BA,\ 3\text{: } BC,\ 4\text{: } CA)$$

Shingle	D_1	D_2	D_3
AB (1)	1	1	1
BA (2)	1	0	1
BC (3)	0	1	0
CA (4)	0	1	0

(2) Min-hash with vertical permutation columns (one table).
Three permutations (as columns):

$$\pi_1 = \begin{bmatrix}1\\2\\3\\4\end{bmatrix}, \quad \pi_2 = \begin{bmatrix}2\\3\\1\\4\end{bmatrix}, \quad \pi_3 = \begin{bmatrix}4\\3\\2\\1\end{bmatrix}$$

π_1	π_2	π_3	D_1	D_2	D_3
1	2	4	1	1	1
2	3	3	1	0	1
3	1	2	0	1	0
4	4	1	0	1	0

Scanning each permutation top-to-bottom and recording the *position in the permutation* of the first 1 in each column gives the signature matrix:

	D_1	D_2	D_3
π_1	1	1	1
π_2	2	1	2
π_3	3	1	3

(3) Similarities.
Column/column (Jaccard):

$$\text{sim}(D_1, D_2) = \tfrac{1}{4}, \quad \text{sim}(D_1, D_3) = 1, \quad \text{sim}(D_2, D_3) = \tfrac{1}{4}.$$

Signature (sig/sig): match fractions across the three signature rows

$$\text{sim}_{\text{sig}}(D_1, D_2) = \tfrac{1}{3}, \quad \text{sim}_{\text{sig}}(D_1, D_3) = 1, \quad \text{sim}_{\text{sig}}(D_2, D_3) = \tfrac{1}{3}.$$

Question 2

We apply Locality-Sensitive Hashing (LSH) on Min-Hash signatures. Assume we use $b = 6$ bands and $r = 5$ rows per band (so $m = 30$ hash functions in total).

1. If a pair of documents D_1 and D_2 have Jaccard similarity $s = 0.7$, compute the probability that they hash to the same bucket in **at least one band**.
2. We construct a $(4, 3)$ AND-OR hash family from a hash family H where $\Pr[h(x) = h(y)] = s$. For $s = 0.85$, compute the probability that two items collide.
3. How many hash functions are required in total to construct the following composed hash family: First a $(4, 3)$ AND-OR construction, followed by a $(2, 5)$ OR-AND construction?

Relevant Concepts

LSH Banding Probability:

$$\text{P(match in one band)} = s^r$$

$$\text{P(no match in any band)} = (1 - s^r)^b$$

$$\text{P(candidate pair)} = 1 - (1 - s^r)^b$$

AND-OR Construction:

$$\text{AND over } r \text{ hash functions: } s^r$$

$$\text{OR over } b \text{ groups: } 1 - (1 - s^r)^b$$

Number of Hash Functions:

$$(r \cdot b) \text{ per AND-OR or OR-AND block.}$$

Answer

(1)

$$1 - (1 - 0.7^5)^6 = 1 - (1 - 0.16807)^6 = 1 - (0.83193)^6 \approx 1 - 0.334 = 0.666$$

(2)

$$s^4 = (0.85)^4 = 0.522$$

$$1 - (1 - 0.522)^3 = 1 - (0.478)^3 = 1 - 0.109 = 0.891$$

(3)

$$(4 \cdot 3) \cdot (2 \cdot 5) = 12 \cdot 10 = 120 \text{ hash functions}$$

Lecture 3: Clustering

Question

You are given the following tiny 2D data set of five points:

$$A(0, 0),\ B(1, 0),\ C(0, 2),\ D(5, 0),\ E(5, 1).$$

(a) **Hierarchical clustering (HC).** Run agglomerative HC with *single*, *complete*, and *average* linkage. For each linkage:

- show the sequence of merges (with merge distances), and
- report the 2-cluster partition obtained by cutting the dendrogram into two clusters.

(b) **K-means ($k = 2$).** Initialize centroids at $\mu_1^{(0)} = (0, 0)$ and $\mu_2^{(0)} = (5, 1)$. Perform one full assignment–update iteration (continue if assignments change). Report final centroids, cluster memberships and the total SSE.

(c) **BFR (one pass then an update).** Process the points in two chunks.

- **Chunk 1:** $\{(0, 0), (0, 1), (1, 0), (5, 5), (5, 6)\}$.
- **Chunk 2:** $\{(0, 2), (5, 4.5), (100, 100)\}$.

Use BFR summaries with per-dimension statistics $(N, \mathrm{SUM}, \mathrm{SUMSQ})$. Assign a new point to a Discard Set (DS) cluster if its squared Mahalanobis distance $MD^2 \le 16$ (use an $\varepsilon = 0.01$ variance in a dimension if the variance would be 0). After processing Chunk 2, list the updated DS summaries for both clusters and identify any points in the RS (retained set).

(d) **CURE (on the final two clusters).** For each cluster found in (b), choose $c = 2$ representative points (farthest apart within the cluster) and shrink them toward the cluster centroid with $\alpha = 0.5$. Give the coordinates of the shrunken representatives and the minimum inter-cluster distance between the two CURE representative sets.

Relevant Concepts

Agglomerative hierarchical clustering. Start with singletons; iteratively merge the closest pair of clusters. Single-link: min pairwise distance; Complete-link: max pairwise distance; Average-link: average of all cross-pair distances.
K-means. Iterate (i) assign points to nearest centroid; (ii) update each centroid to the mean of its assigned points. **SSE** $= \sum_i \|x_i - \mu_{c(i)}\|^2$.
BFR summaries and assignment. For a DS cluster in d dimensions keep $(N, \mathrm{SUM}_j, \mathrm{SUMSQ}_j)_{j=1}^d$ to compute the centroid $\mu_j = \mathrm{SUM}_j/N$ and variance $\sigma_j^2 = \mathrm{SUMSQ}_j/N - \mu_j^2$. For a point x, $MD^2 = \sum_{j=1}^d \frac{(x_j-\mu_j)^2}{\sigma_j^2}$; assign if MD^2 is below a threshold; otherwise it enters RS.
CURE. Pick c well-scattered representatives per cluster, shrink each toward the centroid:

$$r' = \mu + \alpha\,(r - \mu), \quad 0 < \alpha < 1.$$

Cluster distance is the minimum distance between shrunken representative pairs.

Answer

Pairwise distances (Euclidean).

	A	B	C	D	E
A	0	1	2	5	$\sqrt{26}$
B	1	0	$\sqrt{5}$	4	$\sqrt{17}$
C	2	$\sqrt{5}$	0	$\sqrt{29}$	$\sqrt{26}$
D	5	4	$\sqrt{29}$	0	1
E	$\sqrt{26}$	$\sqrt{17}$	$\sqrt{26}$	1	0

(a) HC with three linkages.
Single-link merges:

$$\{A\}+\{B\}\ (1),\quad \{D\}+\{E\}\ (1),\quad \{AB\}+\{C\}\ (2),\quad \{ABC\}+\{DE\}\ (4).$$

Cut to 2 clusters $\Rightarrow \{A, B, C\}, \{D, E\}$.
Complete-link merges:

$$\{A\}+\{B\}\ (1),\quad \{D\}+\{E\}\ (1),\quad \{AB\}+\{C\}\ (\min\{\sqrt{29}, \sqrt{5}\} = \sqrt{5}),\quad \{ABC\}+\{DE\}\ (\sqrt{29}).$$

Cut to 2 clusters $\Rightarrow \{A, B, C\}, \{D, E\}$.
Average-link merges:

$$\{A\}+\{B\}\ (1),\quad \{D\}+\{E\}\ (1),\quad \{AB\}+\{C\}\ (\tfrac{2+2.236}{2} = 2.118),\quad \{ABC\}+\{DE\}\ (\tfrac{69.885}{10} \approx 6.9885).$$

Cut to 2 clusters $\Rightarrow \{A, B, C\}, \{D, E\}$.
(b) K-means ($k = 2$).
Iteration 1 assignments from $\mu_1^{(0)} = (0, 0)$, $\mu_2^{(0)} = (5, 1)$:

$$\mathcal{C}_1 = \{A, B, C\}, \quad \mathcal{C}_2 = \{D, E\}.$$

Updated centroids:

$$\mu_1 = (\tfrac{0+1+0}{3}, \tfrac{0+0+2}{3}) = (0.333, 0.667), \quad \mu_2 = (\tfrac{5+5}{2}, \tfrac{0+1}{2}) = (5, 0.5).$$

Reassignment with these centroids is unchanged $\Rightarrow$ converged.

$$\text{SSE}_{\mathcal{C}_1} = 0.555 + 0.888 + 1.888 = 3.333, \quad \text{SSE}_{\mathcal{C}_2} = 0.25 + 0.25 = 0.5, \quad \text{total SSE} = 3.833.$$

(c) BFR (two chunks, $MD^2 \le 16$ to assign).
Chunk 1 $\{(0,0),(0,1),(1,0),(5,5),(5,6)\}$ produces two DS clusters:
DS_1 from $\{(0,0),(0,1),(1,0)\}$:

$$N = 3,\ \text{SUM} = (1,1),\ \text{SUMSQ} = (1,1),\ \mu_1 = (0.333, 0.333),\ \sigma_1^2 = (0.222, 0.222).$$

DS_2 from $\{(5,5),(5,6)\}$(use $\varepsilon = 0.01$ for x):

$$N = 2,\ \text{SUM} = (10,11),\ \text{SUMSQ} = (50,61),\ \mu_2 = (5, 5.5),\ \sigma_2^2 \approx (0.00, 0.25)$$

Chunk 2 $\{(0,2),(5,4.5),(100,100)\}$:
For $x = (0,2)$,

$$MD^2_{\text{to DS}_1} = \frac{(0-0.333)^2}{0.222} + \frac{(2-0.333)^2}{0.222} \approx 13.03 \le 16 \Rightarrow \text{assign to DS}_1.$$

For $x = (5,4.5)$,

$$MD^2_{\text{to DS}_2} = \frac{(5-5)^2}{0.01} + \frac{(4.5-5.5)^2}{0.25} = 0 + 4 = 4 \le 16 \Rightarrow \text{assign to DS}_2.$$

For $x = (100,100)$, both MD^2 values are enormous $\Rightarrow$ **RS**.
Updated DS summaries:
DS_1 ($N = 4$) with points $\{(0,0),(0,1),(1,0),(0,2)\}$:

$$N = 4,\ \text{SUM} = (1,3),\ \text{SUMSQ} = (1,5),\ \mu_1' = (0.25, 0.75),\ \sigma_1'^2 = (0.1875, 0.6875).$$

DS_2 ($N = 3$) with points $\{(5,5),(5,6),(5,4.5)\}$(use $\varepsilon = 0.01$ for x):

$$N = 3,\ \text{SUM} = (15,15.5),\ \text{SUMSQ} = (75,81.25),\ \mu_2' = (5, 5.1667),\ \sigma_2'^2 \approx (0.00, 0.3889).$$

$\text{RS} = \{(100,100)\}$ (no CS formed since RS has a singleton only).
(d) CURE ($c = 2$, $\alpha = 0.5$ on the k-means clusters from (b)).
Clusters from (b):

- $\mathcal{C}_1 = \{A, B, C\}$ with centroid $\mu_1 = (0.333, 0.667)$;
- $\mathcal{C}_2 = \{D, E\}$ with centroid $\mu_2 = (5, 0.5)$.

Representatives (farthest apart):

- For $\mathcal{C}_1$: choose $B(1,0)$ and $C(0,2)$ (distance ≈ 2.236).
- For $\mathcal{C}_2$: choose $D(5,0)$ and $E(5,1)$.

Shrink by $\alpha = 0.5$ via $r' = \mu + \alpha(r - \mu)$:
$\mathcal{C}_1$:

$$B' \approx (0.333 + 0.5 \cdot (0.667),\ 0.667 + 0.5 \cdot (-0.667)) \approx (0.667,\ 0.333)$$
$$C' \approx (0.333 + 0.5 \cdot (-0.333),\ 0.667 + 0.5 \cdot (1.333)) \approx (0.167,\ 1.333).$$

$\mathcal{C}_2$:

$$D' = (5,\ 0.5 + 0.5 \cdot (-0.5)) = (5, 0.25), \quad E' = (5,\ 0.5 + 0.5 \cdot (0.5)) = (5, 0.75).$$

Minimum inter-cluster distance between shrunken reps:

$$\min\{\ \|B' - D'\|, \|B' - E'\|, \|C' - D'\|, \|C' - E'\|\ \} \approx 4.334.$$

(The closest pair is B' to D'.)

Lecture 4: Dimensionality Reduction

Question 1

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 3 & 4 \end{bmatrix}.$$

1. Perform **power iteration** for 3 iterations starting from $x^{(0)} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$. Estimate the top eigenvalue λ_1 and top eigenvector u_1. After that, use determinant to compute the eigenvalues analytically to verify your answer.
2. **Query with SVD:** For a query vector $q = \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix}$, project to the top $k = 1$ concept space using V_1.
3. **Dimensionality reduction:** Form the rank-1 approximation $A_{(1)}$ from the top 1 singular values. Compute the reconstruction loss $\|A - A_{(1)}\|_F$.

Relevant Concepts

SVD of SPD matrices: If A is symmetric positive definite, its SVD is $A = U\Sigma U^\top$ and $\sigma_i = \lambda_i$ (eigenvalues).
Power iteration:

$$x^{(t+1)} = \frac{Ax^{(t)}}{\|Ax^{(t)}\|_2}, \qquad \lambda^{(t)} = \frac{(x^{(t)})^\top Ax^{(t)}}{(x^{(t)})^\top x^{(t)}}$$

Concept-space projection: For rank-k, coordinates $= q^\top V_k$.
Reconstruction loss:

$$\|A - A_{(k)}\|_F = \sqrt{\sum_{i>k} \sigma_i^2}.$$

Answer

(1) Power Iteration

Start:

$$x^{(0)} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}.$$

$$Ax^{(0)} = \begin{bmatrix} 3 \\ 6 \\ 8 \end{bmatrix}, \qquad x^{(1)} = \frac{1}{\sqrt{3^2 + 6^2 + 8^2}} \begin{bmatrix} 3 \\ 6 \\ 8 \end{bmatrix} = \begin{bmatrix} 0.28735 \\ 0.57470 \\ 0.76626 \end{bmatrix}.$$

Second iteration:

$$Ax^{(1)} = \begin{bmatrix} 1.62831 \\ 3.73553 \\ 5.07649 \end{bmatrix}, \quad x^{(2)} = \begin{bmatrix} 0.25014 \\ 0.57384 \\ 0.77983 \end{bmatrix}.$$

Third iteration:

$$Ax^{(2)} = \begin{bmatrix} 1.60381 \\ 3.73731 \\ 5.09098 \end{bmatrix}, \quad x^{(3)} = \begin{bmatrix} 0.24614 \\ 0.57356 \\ 0.78131 \end{bmatrix},$$

Since the eigenvectors obtained are normalized:

$$\lambda^{(3)} = x^{(3)\top} Ax^{(3)} \approx 6.5160.$$

Top eigenvector:

$$u_1 \approx \begin{bmatrix} 0.24614 \\ 0.57356 \\ 0.78131 \end{bmatrix}.$$

The characteristic polynomial of

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 3 & 4 \end{bmatrix}$$

is

$$\det(A - \lambda I) = -\lambda^3 + 7\lambda^2 - 3\lambda - 1 = 0$$

Thus the eigenvalues are

$$\lambda_1 = 6.5160, \qquad \lambda_2 = 0.70243, \qquad \lambda_3 = -0.21848.$$

(2) Query with $k = 1$
Since A is SPD, $V = U$. Let $V_1 = [u_1]$.
We compute

$$q^\top V_1 = [u_{11}] \approx [0.49228].$$

(3) Reconstruction Loss
The best rank-1 approximation keeps the top two singular values, so

$$A_{(2)} = 6.5160\, u_1 u_1^\top.$$

Using the previously computed eigenvectors

$$u_1 \approx \begin{bmatrix} 0.24614 \\ 0.57356 \\ 0.78131 \end{bmatrix}$$

we obtain the explicit rank-1 approximation:

$$A_{(1)} \approx \begin{bmatrix} 0.39477 & 0.91990 & 1.25310 \\ 0.91990 & 2.14358 & 2.92000 \\ 1.25310 & 2.92000 & 3.97766 \end{bmatrix}$$

The difference matrix is

$$A - A_{(2)} \approx \begin{bmatrix} 1 - 0.39477 & 1 - 0.91990 & 1 - 1.25310 \\ 1 - 0.91990 & 2 - 2.14358 & 3 - 2.92000 \\ 1 - 1.25310 & 3 - 2.92000 & 4 - 3.97766 \end{bmatrix} = \begin{bmatrix} 0.60523 & 0.08010 & -0.25310 \\ 0.08010 & -0.14358 & 0.079997 \\ -0.25310 & 0.079997 & 0.022338 \end{bmatrix}.$$

Its Frobenius norm is

$$\|A - A_{(2)}\|_F = \sqrt{(0.60523)^2 + (-0.14358)^2 + (0.022338)^2 + 2(0.08010)^2 + 2(-0.25310)^2 + 2(0.07997)^2} = 0.735296$$

(Side note for comparison) The reconstruction error based solely on discarded singular values is

$$\|A - A_{(1)}\|_F = \sqrt{\sigma_2^2 + \sigma_3^2} = \sqrt{0.70243^2 + 0.21848^2} = 0.735623.$$

Question 2

Let

$$A = \begin{bmatrix} 1 & 2 & 0 & 0 \\ 2 & 3 & 1 & 0 \\ 0 & 1 & 3 & 2 \\ 0 & 0 & 2 & 3 \end{bmatrix}.$$

1. Perform **one-pass CUR** using a predefined sample: columns $\{1, 3\}$ and rows $\{2, 4\}$. Form C, R, intersection W, compute $U = W^+$, and $A_{\text{CUR}} = CUR$.
2. Report the Frobenius error $\|A - A_{\text{CUR}}\|_F$.

Relevant Concepts

CUR decomposition: Sample columns C and rows R of A; let W be their intersection. Set $U = W^+$ (pseudoinverse) and approximate $A \approx CUR$. **Pseudoinverse (invertible square W):** If W is square and nonsingular, $W^+ = W^{-1}$.

Answer

(1) Fixed samples and CUR factors.

$$C = A[:, \{1,3\}] = \begin{bmatrix} 1 & 0 \\ 2 & 1 \\ 0 & 3 \\ 0 & 2 \end{bmatrix}, \quad R = A[\{2,4\}, :] = \begin{bmatrix} 2 & 3 & 1 & 0 \\ 0 & 0 & 2 & 3 \end{bmatrix},$$

$$W = \text{intersection} = A[\{2,4\},\{1,3\}] = \begin{bmatrix} 2 & 1 \\ 0 & 2 \end{bmatrix}, \quad U = W^{-1} = \begin{bmatrix} \frac{1}{2} & -\frac{1}{4} \\ 0 & \frac{1}{2} \end{bmatrix}.$$

Then

$$A_{\text{CUR}} = CUR = \begin{bmatrix} 1 & \frac{3}{2} & 0 & -\frac{3}{4} \\ 2 & 3 & 1 & 0 \\ 0 & 0 & 3 & \frac{9}{2} \\ 0 & 0 & 2 & 3 \end{bmatrix}.$$

(2) CUR error.

$$\|A - A_{\text{CUR}}\|_F = \sqrt{0.5^2 + 0.75^2 + (-2.5)^2 + 1^2} \approx 2.8395.$$

Lecture 5: Recommender System

Question 1

We consider the following 3×3 user–item rating matrix ("?" means missing):

$$R = \begin{array}{c|ccc} & M_1 & M_2 & M_3 \\ \hline U_1 & 4 & 5 & ? \\ U_2 & 2 & ? & 4 \\ U_3 & ? & 4 & 3 \end{array}$$

1. Perform **user-based collaborative filtering** using cosine similarity to predict M3's rating.
2. Perform **item-based collaborative filtering** using cosine similarity to predict M3's rating.
3. Fit a simple **latent factor model** with $k = 1$ factor and show one step of SGD on entry $r_{1,1} = 4$ (from the matrix). Initially $p = [1,1,1], q = [1,1,1]$, learning rate $\eta = 0.1$
4. Given the below ground truth and user-based predictions, evaluate **RMSE**:

$$r_{1,3} = 5, r_{2,2} = 3.5, r_{3,1} = 2,$$

$$\hat{r}_{1,3} = 4.75, \hat{r}_{2,2} = 3, \hat{r}_{3,1} = 3.75,$$

Relevant Concepts

Row-mean centering: Unknown values treated as 0 after centering. If r_{ui} is missing, use $\tilde{r}_{ui} = 0$.

$$\tilde{r}_{ui} = r_{ui} - \bar{r}_u, \qquad \bar{r}_u = \frac{\text{mean of known ratings in row } u}{\# \text{ known}}$$

Cosine similarity:

$$\text{sim}(a,b) = \frac{a^\top b}{\|a\|\|b\|}$$

User-based CF prediction:

$$\hat{r}_{u,i} = \bar{r}_u + \frac{\sum_v \text{sim}(u,v)\, \tilde{r}_{v,i}}{\sum_v |\text{sim}(u,v)|}$$

Item-based CF prediction:

$$\hat{r}_{u,i} = \bar{r}_i + \frac{\sum_j \text{sim}(i,j)\, \tilde{r}_{u,j}}{\sum_j |\text{sim}(i,j)|}$$

Latent factor model (k=1), where e is the error:

$$\hat{r}_{ui} = p_u q_i, \qquad p_u \leftarrow p_u + \eta(e_{ui} q_i), \quad q_i \leftarrow q_i + \eta(e_{ui} p_u)$$

RMSE:

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum (r_{ui} - \hat{r}_{ui})^2}$$

Answer

(1) Row-mean centered ratings

User means:

$$\bar{r}_1 = 4.5, \qquad \bar{r}_2 = 3, \qquad \bar{r}_3 = 3.5.$$

Centered ratings (missing entries become 0):

$$\tilde{R} = \begin{array}{c|ccc} & M_1 & M_2 & M_3 \\ \hline U_1 & -0.5 & 0.5 & 0 \\ U_2 & -1 & 0 & 1 \\ U_3 & 0 & 0.5 & -0.5 \end{array}$$

Cosine similarities

$$\text{sim}(U_1, U_2) = \frac{(-0.5)(-1) + (0.5)(0) + 0 \cdot 1}{\sqrt{(-0.5)^2 + 0.5^2 + 0^2}\,\sqrt{(-1)^2 + 0^2 + 1^2}} = \frac{0.5}{(\sqrt{0.5})(\sqrt{2})} = 0.5$$

$$\text{sim}(U_1, U_3) = \frac{(-0.5)(0) + (0.5)(0.5) + 0(-0.5)}{(\sqrt{0.5})(\sqrt{0.5})} = \frac{0.25}{0.5} = 0.5$$

$$\text{sim}(U_2, U_3) = \frac{(-1)(0) + (0)(0.5) + (1)(-0.5)}{(\sqrt{2})\,(\sqrt{0.5})} = \frac{-0.5}{1} = -0.5$$

Predicted rating for U_1 on M_3 (user-based)

$$\hat{r}_{1,3} = 4.5 + \frac{0.5(1) + 0.5(-0.5)}{0.5 + 0.5} = 4.75$$

(2) Item-based CF

Compute item means:

$$\bar{r}_{M_1} = 3, \qquad \bar{r}_{M_2} = 4.5, \qquad \bar{r}_{M_3} = 3.5.$$

Centered by columns (missing→0):

$$\tilde{R}^{(items)} = \begin{array}{c|ccc} & M_1 & M_2 & M_3 \\ \hline U_1 & 1 & 0.5 & 0 \\ U_2 & -1 & 0 & 0.5 \\ U_3 & 0 & -0.5 & -0.5 \end{array}$$

Compute similarity between M1 and M3:

$$\text{sim}(M_1, M_3) = \frac{(1)(0) + (-1)(0.5) + (0)(-0.5)}{\sqrt{1^2 + (-1)^2 + 0^2}\,\sqrt{0^2 + (0.5)^2 + (-0.5)^2}} = \frac{-0.5}{(\sqrt{2})(\sqrt{0.5})} = -0.5$$

Compute similarity between M2 and M3:

$$\text{sim}(M_2, M_3) = \frac{(0.5)(0) + (0)(0.5) + (-0.5)(-0.5)}{\sqrt{0.5^2 + 0^2 + (-0.5)^2}\sqrt{0^2 + (0.5)^2 + (-0.5)^2}} = \frac{0.25}{(\sqrt{0.5})(\sqrt{0.5})} = 0.5$$

Prediction:

$$\hat{r}_{1,3} = 3.5 + \frac{-0.5(1) + 0.5(0.5)}{0.5 + 0.5} = 3.25$$

(3) Latent factor model ($k = 1$)

Observed entry example: (U_1, M_1) with $r_{11} = 4$

$$\hat{r}_{11} = 1 \cdot 1 = 1$$
$$e_{11} = 4 - 1 = 3$$

Update:

$$p_1 \leftarrow 1 + 0.1(3)(1) = 1.3$$
$$q_1 \leftarrow 1 + 0.1(3)(1) = 1.3$$

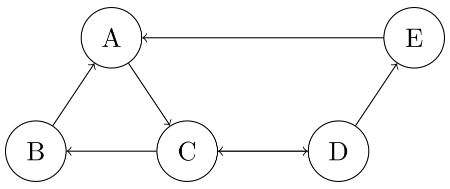
(4) RMSE

$$\begin{aligned}\text{RMSE} &= \sqrt{\frac{(5 - 4.75)^2 + (3.5 - 3)^2 + (2 - 3.75)^2}{3}} \\ &= \sqrt{\frac{0.0625 + 0.25 + 3.0625}{3}} \\ &\approx 1.06066\end{aligned}$$

Lecture 6: Link Analysis

Question 1: PageRank and Topic-Specific PageRank

Consider the directed graph below, with nodes ordered as A, B, C, D, E:



(a) Construct the **transition matrix** M (column-stochastic).

(b) With $\beta = 1$ (no teleport), and initial vector

$$r^{(0)} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix}^\top,$$

compute the PageRank vector after **one** and **two** iterations.

(c) Let the **topic set** be $\{A, B\}$ and teleport probability $1 - \beta = 0.1$. Compute the topic-specific PageRank after one iteration and state *which nodes increase.*

Relevant Concepts

PageRank iteration:

$$r^{(t+1)} = \beta M r^{(t)} + (1 - \beta)v,$$

where v is the teleport vector.

Topic-specific teleport vector:

$$v_i = \begin{cases} \frac{1}{|\mathcal{T}|}, & i \in \mathcal{T}, \\ 0, & \text{otherwise.} \end{cases}$$

Column-stochastic M: Each column j assigns $1/\deg^+(j)$ to its outgoing edges.

Answer

(a) Transition matrix

Out-degrees:

$$\deg^+(A) = 1,\ \deg^+(B) = 1,\ \deg^+(C) = 2,\ \deg^+(D) = 2,\ \deg^+(E) = 1.$$

Thus

$$M = \begin{array}{c|ccccc} & A & B & C & D & E \\ \hline A & 0 & 1 & 0 & 0 & 1 \\ B & 0 & 0 & 1/2 & 0 & 0 \\ C & 1 & 0 & 0 & 1/2 & 0 \\ D & 0 & 0 & 1/2 & 0 & 0 \\ E & 0 & 0 & 0 & 1/2 & 0 \end{array}.$$

(b) PageRank iterations ($\beta = 1$)

Initial:

$$r^{(0)} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}.$$

One iteration:

$$r^{(1)} = Mr^{(0)} = \begin{bmatrix} 2 \\ 0.5 \\ 1.5 \\ 0.5 \\ 0.5 \end{bmatrix}.$$

Two iterations:

$$r^{(2)} = Mr^{(1)} = \begin{bmatrix} 1 \\ 0.75 \\ 2.25 \\ 0.25 \\ 0.25 \end{bmatrix}.$$

(c) Topic-specific PageRank

Topic set $\mathcal{T} = \{A, B\}$, teleport probability $1 - \beta = 0.1$:

$$v = \begin{bmatrix} 1/2 \\ 1/2 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

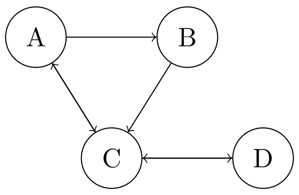
Topic-specific PageRank after one iteration:

$$r^{(1)}_{\text{topic}} = 0.9 \begin{bmatrix} 2 \\ 0.5 \\ 1.5 \\ 0.5 \\ 0.5 \end{bmatrix} + 0.1 \begin{bmatrix} 1/2 \\ 1/2 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1.85 \\ 0.5 \\ 1.35 \\ 0.45 \\ 0.45 \end{bmatrix}.$$

Nodes whose score increases: A, B.

Question 2: HITS Algorithm

Consider the directed graph:



(a) Write the **adjacency matrix** L.

(b) Run **one iteration** of HITS starting with all hub scores $h = 1$. Compute new authority and hub scores.

(c) Normalize both vectors.

Relevant Concepts

HITS updates:

$$a = L^\top h, \qquad h = La.$$

Normalization:

$$a \leftarrow \frac{a}{\|a\|_2}, \qquad h \leftarrow \frac{h}{\|h\|_2}.$$

Answer

(a) Adjacency matrix

$$L = \begin{array}{c|cccc} & A & B & C & D \\ \hline A & 0 & 1 & 1 & 0 \\ B & 0 & 0 & 1 & 0 \\ C & 1 & 0 & 0 & 1 \\ D & 0 & 0 & 1 & 0 \end{array}$$

(b) One HITS iteration

Initial:

$$h^{(0)} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Authorities:

$$a^{(1)} = L^\top h^{(0)} = \begin{bmatrix} 1 \\ 1 \\ 3 \\ 1 \end{bmatrix}$$

Hubs:

$$h^{(1)} = La^{(1)} = \begin{bmatrix} 4 \\ 3 \\ 2 \\ 3 \end{bmatrix}$$

(c) Normalization

$$\|a^{(1)}\|_2 = \sqrt{1^2 + 1^2 + 3^2 + 1^2} = \sqrt{12}.$$

$$a^{(1)}_{\text{norm}} = \frac{1}{\sqrt{12}} \begin{bmatrix} 1 \\ 1 \\ 3 \\ 1 \end{bmatrix}.$$

$$\|h^{(1)}\|_2 = \sqrt{4^2 + 3^2 + 2^2 + 3^2} = \sqrt{38}.$$

$$h^{(1)}_{\text{norm}} = \frac{1}{\sqrt{38}} \begin{bmatrix} 4 \\ 3 \\ 2 \\ 3 \end{bmatrix}.$$

Question 3: Web Spam Detection (TrustRank / BadRank)

A small web graph has pages $P = \{A, B, C, D, E\}$. Let trusted seed set be $\{A\}$ and spam seed set be $\{E\}$. The transition matrix is M is the same as in Question 1:

$$M = \begin{array}{c|ccccc} & A & B & C & D & E \\ \hline A & 0 & 1 & 0 & 0 & 1 \\ B & 0 & 0 & 1/2 & 0 & 0 \\ C & 1 & 0 & 0 & 1/2 & 0 \\ D & 0 & 0 & 1/2 & 0 & 0 \\ E & 0 & 0 & 0 & 1/2 & 0 \end{array}.$$

Let $\beta = 0.85$ and initial vectors $t^{(0)} = b^{(0)} = [1, 0, 0, 0, 0]^\top$.

(a) Compute one iteration of **TrustRank** using teleport vector concentrated on A.

(b) Compute one iteration of **BadRank** using teleport vector concentrated on E.

(c) Identify which nodes are likely trustworthy vs spammy.

Relevant Concepts

TrustRank iteration:

$$t^{(k+1)} = \beta M t^{(k)} + (1 - \beta) e_A.$$

BadRank iteration:

$$b^{(k+1)} = \beta M b^{(k)} + (1 - \beta) e_E.$$

Answer

(a) TrustRank

$$t^{(1)} = 0.85 M t^{(0)} + 0.15 e_A = 0.85 \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} + 0.15 \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.15 \\ 0 \\ 0.85 \\ 0 \\ 0 \end{bmatrix}.$$

(b) BadRank

$$b^{(1)} = 0.85 M b^{(0)} + 0.15 e_E = 0.85 \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + 0.15 \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.85 \\ 0 \\ 0 \\ 0 \\ 0.15 \end{bmatrix}.$$

(c) Interpretation

- High TrustRank $\Rightarrow$ likely **trusted**: C, A
- High BadRank $\Rightarrow$ likely **spam**: A, E

Combining:

- C is trusted but not spammy $\Rightarrow$ trustworthy
- E is clearly spam
- A receives mixed signals (trusted but also pointed by spam)

Lecture 7: Graphs

Question 1: Personalized PageRank with Sweep Cut

Consider the directed graph below (same as standard PageRank examples but smaller for hand-tracing): Let the adjacency matrix be column-stochastic in node order (A, B, C, D):

$$M = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}.$$

We run **Personalized PageRank (PPR)** with damping $\alpha = 0.85$, teleport set

$$s = e_A = [1, 0, 0, 0]^\top.$$

1. Perform two iterations of PPR starting from
$$p^{(0)} = \frac{1}{4}[1, 1, 1, 1]^\top.$$
2. Treating the underlying graph as undirected for conductance computation and ignoring self-loops, use the PPR scores to compute a **sweep cut** over ordering (A, B, C, D) sorted by decreasing $p^{(2)}$. Report the conductance of each prefix set and identify the best community.

Relevant Concepts

PPR iteration:

$$p^{(t+1)} = \alpha M p^{(t)} + (1 - \alpha)s.$$

Sweep cut:

- Sort nodes by PPR score (in decreasing order), giving an ordering
$$v_1, v_2, \ldots, v_n.$$
- Form prefix sets
$$S_k = \{v_1, v_2, \ldots, v_k\}, \quad k = 1, 2, \ldots, n - 1.$$
- For each S_k, compute the conductance (undefined for the whole graph):
$$\phi(S_k) = \frac{\text{cut}(S_k, \bar{S}_k)}{\min(\text{vol}(S_k), \text{vol}(\bar{S}_k))}.$$
- **Best community:** choose the prefix set S_k with the *smallest* conductance $\phi(S_k)$ (excluding the empty set and the full set).

Answer

First iteration:

$$Mp^{(0)} = \begin{bmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.50 \end{bmatrix}, \quad p^{(1)} = 0.85Mp^{(0)} + 0.15s = \begin{bmatrix} 0.3625 \\ 0.2125 \\ 0.2125 \\ 0.425 \end{bmatrix}$$

Second iteration:

$$Mp^{(1)} = \begin{bmatrix} 0.2125 \\ 0.3625 \\ 0.2125 \\ 0.6375 \end{bmatrix}, \quad p^{(2)} = 0.85Mp^{(1)} + 0.15s = \begin{bmatrix} 0.3306 \\ 0.3081 \\ 0.1806 \\ 0.5419 \end{bmatrix}.$$

Sorted PPR scores (descending):

$$p_D > p_A > p_B > p_C \quad \Longrightarrow \quad (D,\ A,\ B,\ C).$$

(2) Sweep Cut from PPR Ordering
Edges:

$$A-B, \quad B-C, \quad C-A, \quad C-D.$$

Degrees:

$$\deg(A) = 2, \quad \deg(B) = 2, \quad \deg(C) = 3, \quad \deg(D) = 1 \quad \text{(ignore self loop)}$$

so total volume $\text{vol}(V) = 8$.
Using ordering (D, A, B, C), the prefix sets are:

$$S_1 = \{D\}, \quad S_2 = \{D, A\}, \quad S_3 = \{D, A, B\}.$$

Set $S_1 = \{D\}$. Only edge leaving S_1 is $C-D$:

$$\text{cut}(S_1, \bar{S}_1) = 1, \quad \text{vol}(S_1) = \deg(D) = 1, \quad \text{vol}(\bar{S}_1) = 7.$$

$$\phi(S_1) = \frac{1}{\min(1, 7)} = 1.$$

Set $S_2 = \{D, A\}$. Edges leaving:

$$A-B, \quad C-A \quad C-D \quad \Rightarrow \quad \text{cut}(S_2, \bar{S}_2) = 3.$$

Volumes:

$$\text{vol}(S_2) = \deg(D) + \deg(A) = 1 + 2 = 3, \quad \text{vol}(\bar{S}_2) = 8 - 3 = 5.$$

$$\phi(S_2) = \frac{3}{\min(3, 5)} = \frac{3}{3} = 1.$$

Set $S_3 = \{D, A, B\}$. Edges leaving:

$$A-C, \quad B-C \quad C-D \quad \Rightarrow \quad \text{cut}(S_3, \bar{S}_3) = 3.$$

Volumes:

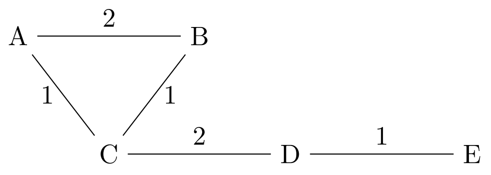
$$\text{vol}(S_3) = \deg(D) + \deg(A) + \deg(B) = 1 + 2 + 2 = 5, \quad \text{vol}(\bar{S}_3) = 3.$$

$$\phi(S_3) = \frac{3}{\min(5, 3)} = \frac{3}{3} = 1.$$

So the three partitions are equivalently good as they have the same conductance.

Question 2: Louvain Community Detection

Consider the undirected weighted graph:



Initial communities = each node alone.

1. For node C, compute ΔQ when moving C into communities of A, B, and D.
2. Determine the best move for C.
3. After updating C, repeat for node B and determine its best move.

Relevant Concepts

Modularity gain for moving node i into community C:

$$\Delta Q = \frac{k_{i,C}}{m} - \frac{k_i \cdot \text{tot}_C}{2m^2},$$

where $k_{i,C}$ = sum of edge weights from i to community C, k_i = degree of i, tot_C = total degree of C, m = total edge weight of graph.

Answer

Compute total weight:

$$m = 2 + 1 + 1 + 2 + 1 = 7.$$

Degrees:

$$k_A = 3,\ k_B = 3,\ k_C = 4,\ k_D = 3,\ k_E = 1.$$

For node C:

$$k_{C,A} = 1, \quad k_{C,B} = 1, \quad k_{C,D} = 2.$$

Compute ΔQ:

$$\Delta Q_{C\to A} = \Delta Q_{C\to B} = \frac{1}{7} - \frac{4\cdot 3}{2\cdot 49} = 0.1429 - 0.1224 = 0.0204.$$

$$\Delta Q_{C\to D} = \frac{2}{7} - \frac{4\cdot 3}{2\cdot 49} = 0.2857 - 0.1224 = 0.1633.$$

Best move: $C \to D$.

Now update node B (with C already merged into D). Current communities: $\{A\}$, $\{B\}$, $\{C, D\}$, $\{E\}$. Total degrees of communities:

$$\text{tot}_A = 3, \quad \text{tot}_B = 3, \quad \text{tot}_{C,D} = 4 + 3 = 7, \quad \text{tot}_E = 1.$$

Connections from B:

$$k_{B,A} = 2, \qquad k_{B,\{C,D\}} = 1, \qquad k_{B,E} = 0.$$

Compute gains:

$$\Delta Q_{B\to A} = \frac{2}{7} - \frac{3\cdot 3}{98} = 0.2857 - 0.0918 = 0.1939.$$

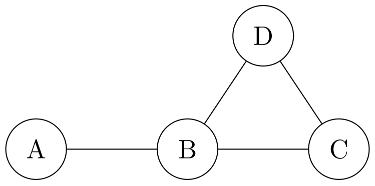
$$\Delta Q_{B\to\{C,D\}} = \frac{1}{7} - \frac{3\cdot 7}{98} = 0.1429 - 0.2143 = -0.0714.$$

$$\Delta Q_{B\to E} = 0 - \frac{3\cdot 1}{98} = -0.0306.$$

Best move: $B \to A$.

Final communities after updating C and then B: $\{A,B\}$, $\{C,D\}$, $\{E\}$.

Question 3: Graph Embedding via Random Walks



We run random-walk-based Skip-Gram for the above graph. Let the random walk be:

$$A \to B \to C \to D.$$

1. Using window size $w = 1$, list all (center,context) training pairs.
2. If the embedding dimension is $d = 2$ and initial embeddings are

   $$v_A = [1,0],\ v_B = [0,1],\ v_C = [-1,0],\ v_D = [0,-1],$$

   compute the SGD gradient update for the pair (B,C) using negative sampling with 1 negative sample A and learning rate $\eta = 0.1$.

Relevant Concepts

Skip-Gram objective with negative sampling:

$$\ell = \log\sigma(v_c^\top v_{ctx}) + \log\sigma(-v_c^\top v_{neg}).$$

Gradient Update Function, where σ is the sigmoid activation function:

$$\nabla_{v_c} = (1-\sigma(v_c^\top v_{ctx}))v_{ctx} - \sigma(v_c^\top v_{neg})v_{neg}, \qquad v \leftarrow v + \eta\nabla_v$$

Answer

(1) Training pairs for $w = 1$

From the walk $A \to B \to C \to D$, the Skip-Gram pairs are:

$$(A,B),(B,A),(B,C),(C,B),(C,D),(D,C).$$

(2) Gradient update for pair (B,C)

$$v_B = [0,1], \qquad v_C = [-1,0], \qquad v_A = [1,0].$$

Dot products

$$v_B^\top v_C = 0, \qquad v_B^\top v_A = 0, \qquad \sigma(0) = 0.5.$$

Gradient for the center vector v_B

$$\nabla_{v_B} = (1-0.5)v_C \ - \ 0.5v_A = 0.5[-1,0] - 0.5[1,0] = [-1,0].$$

$$v_B' = v_B + 0.1\nabla_{v_B} = [0,1] + 0.1[-1,0] = [-0.1,\ 1].$$

Gradient for the context vector v_C

$$\nabla_{v_C} = (1-0.5)v_B = 0.5[0,1] = [0,0.5],$$

$$v_C' = v_C + 0.1\nabla_{v_C} = [-1,0] + 0.1[0,0.5] = [-1,\ 0.05].$$

Final updated embeddings

$$v'_B = [-0.1,\ 1], \qquad v'_C = [-1,\ 0.05].$$

Lecture 8: Streams

Question 1: Reservoir Sampling

We run **Reservoir Sampling (reservoir size $k = 2$)** on the following stream of 7 items:

$$S = [A, B, C, D, E, F, G].$$

(a) Calculate the probability that item G ends up in the reservoir.

(b) Suppose the random choices are as follows, show the final reservoir.

Item	Random outcome
C	replace position 1
D	skip
E	replace position 2
F	skip
G	replace position 1

Relevant Concepts

- Reservoir sampling keeps each element with probability k/i at position i.
- All items must end with equal probability k/n.

Answer

(a) Probability that G is included. At time $i = 7$, probability of selection:

$$\Pr(G \text{ selected}) = \frac{2}{7}.$$

(b) Run with given random choices.

i	Incoming	Reservoir
1	A	$[A, -]$
2	B	$[A, B]$
3	C	$[C, B]$
4	D	skip $\Rightarrow [C, B]$
5	E	$[C, E]$
6	F	skip $\Rightarrow [C, E]$
7	G	$[G, E]$

Final reservoir: $[\mathrm{G}, \mathrm{E}]$

Question 2: DGIM — Bucketized Stream

A stream of bits ends with the following 16 bits (most recent on the right). Window size is $N = 16$:

$$S = 1\,0\,1\,1\,0\,0\,1\,0\,1\,0\,0\,1\,1\,1\,0\,1.$$

(a) Construct the DGIM bucket structure.

(b) Give the upper bound of error in estimating the number of 1s.

(c) Using the DGIM buckets, estimate the number of 1s in the final 16 bits.

Relevant Concepts

- DGIM stores buckets of size with powers of 2. When a 1 arrives at time t: create a new size-1 bucket with end time t. If there are more than two buckets of the same size, merge the two oldest of that size into one bucket of double the size, whose timestamp is the newer of those two.
- Error bound = half the size of the largest bucket.

Answer

(a) Bucket structure
The 1s in the window occur at positions, where 1 is the leftmost position:

$$16,\ 14,\ 13,\ 12,\ 9,\ 7,\ 4,\ 3,\ 1.$$

After applying DGIM merging rules, the resulting buckets (right to left) are:

$$1\{16\},\ 2\{13,14\},\ 2\{9,12\},\ 4\{1,3,4,7\}$$

(b) Error bound:

$$\text{Error} \le \frac{4}{2} = 2.$$

(c) Estimate #(1s).
dd all buckets except count only half of oldest:

$$1+2+2+\frac{4}{2} = 1+2+2+2 = 7.$$

Question 3: Bloom Filters

We insert elements $\{A, B, C\}$ into a Bloom filter of size $m = 10$ bits, using two hash functions:

$$h_1(x) = (|x| \mod 10), \quad h_2(x) = (3|x| + 1 \mod 10).$$

Let:

$$|A| = 1, \quad |B| = 2, \quad |C| = 3.$$

(a) Show the bit array after inserting all elements.

(b) Check whether D with $|D| = 4$ is reported as “possibly in set”.

(c) Compute the false-positive probability formula.

Relevant Concepts

- Bloom filter sets bits at positions $h_1(x)$ and $h_2(x)$.
- False positive rate for k hashes, n elements.

$$\left(1 - e^{-kn/m}\right)^k$$

Answer

(a) Insert A, B, C.

$$A : h_1 = 1,\ h_2 = 4$$
$$B : h_1 = 2,\ h_2 = 7$$
$$C : h_1 = 3,\ h_2 = 0$$

Final bit array:

$$[1, 1, 1, 1, 1, 0, 0, 1, 0, 0].$$

(b) Query D ($|D| = 4$):

$$h_1(D) = 4, \quad h_2(D) = 3.$$

Both positions 3 and 4 are 1.
Thus:

$$D \text{ is reported as “possibly in set”}.$$

(c) False positive probability:

$$(1 - e^{-2\cdot 3/10})^2 = (1 - e^{-0.6})^2 \approx 0.203571$$

Question 4: Flajolet–Martin (FM) Algorithm

Consider the stream:

$$S = [5, 1, 2, 8, 4, 16].$$

The hash function is:

$$h(x) = x \text{ written in binary}.$$

(a) List the binary values and number of trailing zeros.

(b) Compute the FM estimate.

Relevant Concepts

- FM tracks longest suffix of zeros in hashed values.
- Estimate: 2^R, where R = max trailing zeros.

Answer

x	$h(x)$	Trailing zeros
5	101	0
1	1	0
2	10	1
8	1000	3
4	100	2
16	10000	4

Largest number of trailing zeros: $R = 4$. Therefore FM estimate: $2^4 = 16$.

Question 5: AMS Algorithm for 2nd Moment

The stream is:

$$S = [a, b, a, a, c, b, a].$$

(a) Compute the exact 2nd moment.

(b) Suppose AMS picks item b at $i = 2$ randomly, continue the run and compute the AMS estimate.

(c) Repeat for $i = 3$ (item a) and average the two estimates.

Relevant Concepts

For frequencies f_x, the 2nd moment is:

$$F_2 = \sum_x f_x^2.$$

AMS estimate (for sample position i):

$$X = n(2c - 1),$$

where c = number of occurrences of the sampled element from position i onward (inclusive of i, min 1).

Answer

(a) True frequencies:

$$f_a = 4, \quad f_b = 2, \quad f_c = 1.$$
$$F_2 = 4^2 + 2^2 + 1^2 = 21.$$

(b) Pick $i = 2$ (b).
Occurrences of b from position 2 onward: $c = 2$, stream length $n = 7$.

$$X_1 = 7(2 \cdot 2 - 1) = 7(3) = 21.$$

(c) Pick $i = 3$ (a).
Occurrences of a from position 3 onward: $c = 3$.

$$X_2 = 7(2 \cdot 3 - 1) = 7(5) = 35.$$

Average AMS estimate:

$$\frac{21 + 35}{2} = 28 \text{ vs true } 21.$$

Lecture 9: Web Advertising

Question 1: Online Bipartite Matching (Greedy vs BALANCE)

We have two advertisers, A and B. Both have budget \$2. The bids are:

	x	y	z
A	1	1	0
B	1	0	1

For each incoming query sequence below, determine whether BALANCE is **guaranteed** to be optimal. Justify in one sentence.

1. Will BALANCE be optimal on the sequence $xyxz$?
2. Will BALANCE be optimal on the sequence $xxyz$?
3. Will BALANCE be optimal on the sequence $yzzx$?
4. Will BALANCE be optimal on the sequence $zxxy$?

Relevant Concepts

- **Greedy** assigns each query to any advertiser with the highest bid who still has budget.
- **BALANCE** always assigns to the advertiser with the *largest remaining budget* (break ties arbitrarily).
- BALANCE is known to be a 1/2-approximation and sometimes strictly better than Greedy.
- BALANCE is optimal when advertisers never compete for the same query or when the high-demand advertiser always loses budget first under any optimal solution.

Answer

1. **No.** OPTIMAL allocates 2 queries to A and 2 to B. On the other hand, suppose the tie breaker allocates x to B when A and B have the same budget, then BALANCE allocates 2 queries to A and 1 query to B, the last query z is not allocated.
2. **No.** Suppose the tie breaker allocates x to A when A and B have the same budget, BALANCE spends A's budget early, preventing assigning a later profitable z to B.
3. **Yes.** No conflict on y, and BALANCE correctly splits z, z to B and x to A.
4. **No.** OPTIMAL allocates 2 queries to A and 2 to B. On the other hand, suppose the tie breaker allocates x to A when A and B have the same budget, then BALANCE allocates 1 query to B and 2 queries to A, the last query y is not allocated.

Question 2: Multi-Armed Bandits — Greedy and ε-Greedy

You have two arms with true mean rewards:

$$\mu_1 = 0.6, \qquad \mu_2 = 0.4.$$

Suppose the first two pulls (one per arm) return:

$$R_1 = 1, \qquad R_2 = 0.$$

1. If we use the **pure Greedy** algorithm, which arm will be selected on the next pull?
2. Using ε-Greedy with $\varepsilon = 0.2$, what is the probability that arm 1 is selected on the next pull?

Relevant Concepts

- **Greedy**: always selects arm with the largest empirical mean.
- **ε-Greedy**:
$$P(\text{exploit}) = 1 - \varepsilon, \quad P(\text{explore}) = \varepsilon.$$
  During exploration, choose uniformly over all arms.

Answer

1. Arm 1 since it has the higher empirical mean $\hat{\mu}_1 = 1 > \hat{\mu}_2 = 0$
2. $$P(\text{choose arm 1}) = (1 - \varepsilon) \cdot 1 + \varepsilon \cdot \frac{1}{2} = 0.8 \cdot 1 + 0.2 \cdot 0.5 = 0.9.$$

Question 3: UCB1 Algorithm

Assume two arms. At time $t = 5$, each arm has been pulled the following number of times:

$$n_1 = 3, \qquad n_2 = 2.$$

Empirical means:

$$\hat{\mu}_1 = 0.5, \qquad \hat{\mu}_2 = 0.4.$$

Using the UCB1 index:

$$\text{UCB}_i = \hat{\mu}_i + \sqrt{\frac{2 \ln t}{n_i}},$$

determine which arm will be chosen at time $t = 6$.

Relevant Concepts

- UCB1 trades off exploration and exploitation using a confidence-bound term.
- The arm with the higher UCB index is selected.

Answer

$$\text{UCB}_1 = 0.5 + \sqrt{\frac{2 \ln 5}{3}} \approx 0.5 + 1.0358 = 1.5358.$$

$$\text{UCB}_2 = 0.4 + \sqrt{\frac{2 \ln 5}{2}} \approx 0.4 + 1.2686 = 1.6686.$$

So we choose arm 2.

Lecture 10: Submodular Functions

Question 1: Probabilistic Set Cover

Suppose we have a universe of elements

$$U = \{e_1, e_2, e_3, e_4\},$$

and three sets that cover elements with certain probabilities:

$$\begin{aligned} S_1 &= \{e_1, e_2\}, \quad \Pr(\text{each covered}) = 0.8 \\ S_2 &= \{e_2, e_3\}, \quad \Pr(\text{each covered}) = 0.5 \\ S_3 &= \{e_3, e_4\}, \quad \Pr(\text{each covered}) = 0.7 \end{aligned}$$

Each set has cost 1. You want to choose a collection of sets so that the expected coverage of all 4 elements is at least 0.9.

(a) Compute the expected coverage of each element if we pick all three sets.

(b) Using the probabilistic set cover greedy strategy (pick the set that increases total expected coverage the most per unit cost), determine the order in which sets are selected.

(c) After selecting two sets greedily, what is the expected coverage of all elements? Do we need to continue to select more sets?

Relevant Concepts

Probabilistic coverage. If an element appears in several chosen sets, each covering it independently with probabilities $p_1, p_2, \ldots$, then:

$$\Pr(\text{element covered}) = 1 - \prod_i (1 - p_i).$$

Greedy probabilistic set cover. At each step choose the set that provides the largest *marginal gain*:

$$\Delta(S) = \sum_{e \in S} \left[(1 - \prod_{i \in C_e} (1 - p_i))_{\text{new}} - (1 - \prod_{i \in C_e} (1 - p_i))_{\text{old}} \right],$$

per unit cost.

Answer

(a) Expected coverage picking all sets.

$$\begin{aligned} \Pr(e_1 \text{ covered}) &= 1 - (1 - 0.8) = 0.8 \\ \Pr(e_2 \text{ covered}) &= 1 - (1 - 0.8)(1 - 0.5) = 1 - (0.2)(0.5) = 0.9 \\ \Pr(e_3 \text{ covered}) &= 1 - (1 - 0.5)(1 - 0.7) = 1 - (0.5)(0.3) = 0.85 \\ \Pr(e_4 \text{ covered}) &= 1 - (1 - 0.7) = 0.7 \end{aligned}$$

(b) Greedy order.
Initial expected coverage = 0 for all elements.
Marginal gains:

- S_1: contributes $0.8 + 0.8 = 1.6$
- S_2: contributes $0.5 + 0.5 = 1.0$
- S_3: contributes $0.7 + 0.7 = 1.4$

So the first pick is S_1.
Recompute gains after selecting S_1:

- For e_2, already covered with 0.8, adding S_2 increases:

$$\Delta_{S_2}(e_2) = 0.9 - 0.8 = 0.1, \qquad \Delta_{S_2}(e_3) = 0.5.$$

  Total $= 0.6$.

- For S_3:

$$\Delta_{S_3}(e_3) = 0.7, \qquad \Delta_{S_3}(e_4) = 0.7.$$

  Total $= 1.4$.

  Thus the greedy order is:

$$S_1 \to S_3 \to S_2.$$

(c) Expected coverage after selecting S_1 and S_3.

$$\Pr(e_1) = 0.8, \qquad \Pr(e_2) = 0.8,$$
$$\Pr(e_3) = 1 - (1 - 0.5)(1 - 0.7) = 0.85, \qquad \Pr(e_4) = 0.7.$$

None of the elements exceed the required 0.9 threshold, so we need to select more sets.

Question 2: Simple Greedy vs Lazy Greedy for Submodular Maximization

Consider the monotone submodular function

$$f(S) = \sqrt{w(S)},$$

where $w(S) = \sum_{e \in S} w(e)$ is a modular weight function over the ground set

$$V = \{a, b, c, d, e\}.$$

Suppose the element weights are

$$w(a) = 10000, \quad w(b) = 4096, \quad w(c) = 16, \quad w(d) = 4, \quad w(e) = 1.$$

We wish to select $k = 3$ elements.
The marginal gain of adding an element t to a set S is

$$\Delta_f(t \mid S) = \sqrt{w(S) + w(t)} - \sqrt{w(S)}.$$

1. **Simple greedy:** At each step, recompute $\Delta_f(t \mid S)$ for all remaining elements and pick the one with the largest marginal gain.
   (a) List the order of selected elements.
   (b) Compute the total number of marginal-gain evaluations performed by simple greedy.
2. **Lazy greedy:** Lazy greedy begins by evaluating $\Delta_f(t \mid \emptyset)$ for all $t \in V$ and inserting these values into a max-heap. Thereafter it recomputes a marginal gain only when the heap top is stale.
   (a) Describe the sequence of marginal-gain recomputations.
   (b) Compute the total number of marginal-gain evaluations performed by lazy greedy.
3. Explain why lazy greedy performs *strictly fewer* marginal-gain evaluations than simple greedy for this instance, even though the selected set is identical.

Relevant Concepts

Simple Greedy. At each step recompute Δ_f for *all* remaining elements and pick the best.
Lazy Greedy. Maintains a max-heap of (possibly outdated) marginal gains; recompute only when the top candidate may no longer be optimal. For submodular functions, marginal gains only decrease, so cached values serve as upper bounds.

Answer

(a) Simple greedy.
Round 1: $S = \emptyset$.

$$\Delta(a) = 100, \qquad \Delta(b) = 64, \qquad \Delta(c) = 4, \qquad \Delta(d) = 2, \qquad \Delta(e) = 1.$$

Pick a.
Round 2: $S = \{a\}$.

$$\Delta(b \mid S) \approx 18.72, \qquad \Delta(c \mid S) \approx 0.0800, \qquad \Delta(d \mid S) \approx 0.0200, \qquad \Delta(e \mid S) \approx 0.005.$$

Pick b.
Round 3: $S = \{a, b\}$.

$$\Delta(c \mid S) \approx 0.0674, \qquad \Delta(d \mid S) \approx 0.0168, \qquad \Delta(e \mid S) \approx 0.0042.$$

Pick c.
Selected order:

$$a \to b \to c.$$

Simple-greedy evaluations:

$$5 \text{ (round 1)} + 4 \text{ (round 2)} + 3 \text{ (round 3)} = 12.$$

(b) Lazy greedy.
Initialization, 5 evaluations: compute $\Delta(t \mid \emptyset)$:

$$100,\ 64,\ 4,\ 2,\ 1.$$

Heap top: $(100, a)$. Accept a without recomputation.
Step 2 (second element), 1 evaluation.
Top of heap is $(64, b)$ (stale). Recompute:

$$\Delta(b \mid \{a\}) \approx 18.72.$$

Since this new value still exceeds the stale upper bounds for c, d, e, accept b.
Step 3 (third element), 3 evaluations.
Heap contains stale values $(4, c), (2, d), (1, e)$.
Lazy greedy proceeds as follows:

1. Pop $(4, c)$ and recompute:
$$\Delta(c \mid \{a, b\}) \approx 0.067.$$
Since $0.057 < 2$, push back $(0.067, c)$.
2. Pop $(2, d)$ and recompute:
$$\Delta(d \mid \{a, b\}) \approx 0.017.$$
Since $0.017 < 1$, push back $(0.017, d)$.
3. Pop $(1, e)$ and recompute:
$$\Delta(e \mid \{a, b\}) \approx 0.004.$$
Push back $(0.003, e)$.
4. Now the heap contains updated values:
$$(0.067, c),\ (0.017, d),\ (0.004, e).$$
Pop $(0.067, c)$ and accept c.

Lazy-greedy evaluations:

$$5 \text{ (initial)} + 1 \text{ (recompute } b) + 3 \text{ (recompute } c, d, e) = 9.$$

(c) Comparison.
Simple greedy performs 12 evaluations while lazy greedy performs 9 evaluations. Lazy greedy saves work because stale marginal gains provide upper bounds: although c, d, e must all be recomputed in Round 3, lazy greedy never recomputes elements unnecessarily in the second round. Therefore, while both algorithms select the same set $\{a, b, c\}$, lazy greedy performs fewer marginal-gain computations.