1

Code: 1_1.py

Description of the pipeline:

For each line, say [u, a, b, c], we emit

((u, a), -inf), ((u, b), -inf), ((u, c), -inf), ((a, b), 1), ((a, c), 1), ((b, c), 1)

That is, for a pair that can be friends of friends, the score is 1, where the score will be accumulated to find the number of mutual friends. If a pair is known to be friends, the score is -inf, such that the accumulated score of this pair will be -inf. We can use accumulated scores to filter out existing friends and make recommendations.

Specific recommendations

Format: (user id, (recommended ids))

```
(924, (439, 2409, 6995, 11860, 15416, 43748, 45881))
(8941, (8943, 8944, 8940))
(8942, (8939, 8940, 8943, 8944))
(9019, (9022, 317, 9023))
(9020, (9021, 9016, 9017, 9022, 317, 9023))
(9021, (9020, 9016, 9017, 9022, 317, 9023))
(9022, (9019, 9020, 9021, 317, 9016, 9017, 9023))
(9990, (13134, 13478, 13877, 34299, 34485, 34642, 37941))
(9992, (9987, 9989, 35667, 9991))
(9993, (9991, 13134, 13478, 13877, 34299, 34485, 34642, 37941))
```

2

Note:

$A \rightarrow B$ is an association rule: if a basket contains item set A, it also contains item set B. A rule's correctness is subject to measurement.

Pr(A) is the probability that a basket contains item set A.

2 (a)

conf(A->B) = Pr(B|A) = Pr(AB) / Pr(A)

If Pr(B) is near 1, so is conf(A->B), but this doesn't guarantee a strong rule.

List and conviction don't suffer from this, because they explicitly include Pr(B).

2 (b)

Conf is NOT symmetric. Counterexample: If $A \supset B$, $conf(A \rightarrow B) = 1$. $conf(B \rightarrow A) < 1$

Lift is symmetric. Proof: $lift(A \rightarrow B) = \frac{conf(A \rightarrow B)}{S(B)} = \frac{supp(A \cup B)N}{supp(A)supp(B)} = lift(B \rightarrow A)$

Conv is NOT symmetric. Counterexample: If $A \supset B$, $conv(A \rightarrow B) = \infty$. $conv(B \rightarrow A)$ is bounded.

2 (c)

Conf is desirable. Its range is $[0,1]$. If $A \rightarrow B$ always holds, $conf(A \rightarrow B) = \Pr(B|A) = 1$

Lift is NOT desirable. Its range is $[0, \infty)$. When $A = B$, it can be arbitrarily large, or 1.

Conv is desirable. Its range is $(0, \infty)$. (Approaching 0 when B is almost universal but $\Pr(AB) = 0$.) If $A \rightarrow B$ always holds, conf is 1, so conv is inf, as long as $S(B) \neq 1$.

2 (d)

Format: (A -> B, confidence)

[(u'DAI93865 -> FRO40251', 1.0),

```
 (u'GRO85051 -> FRO40251', 0.999176276771005),
 (u'GRO38636 -> FRO40251', 0.9906542056074766),
 (u'ELE12951 -> FRO40251', 0.9905660377358491),
 (u'DAI88079 -> FRO40251', 0.9867256637168141)]
```

2 (e)

Format: ((A, B) -> C, confidence)

```
[(u'(DAI23334, ELE92920) -> DAI62779', 1.0),
 (u'(DAI31081, GRO85051) -> FRO40251', 1.0),
 (u'(DAI55911, GRO85051) -> FRO40251', 1.0),
 (u'(DAI62779, DAI88079) -> FRO40251', 1.0),
 (u'(DAI75645, GRO85051) -> FRO40251', 1.0)]
```

Code for (d, e): 1_2_de.py

Comment: More triplets have confidence 1!

3 (a)

Pr(don't know) = Pr(no 1s chosen) =

$$=\frac{\binom{n-m}{k}}{\binom{n}{k}}=\frac{\frac{(n-m)!}{(n-m-k)!k!}}{\frac{n!}{(n-k)!k!}}=\frac{(n-m)!}{n!}\frac{(n-k)!}{(n-k-m)!}=\frac{(n-k)!}{n!}\frac{(n-m)!}{(n-k-m)!}=\frac{(n-k)}{n}\frac{(n-k-1)}{(n-1)}\cdots\frac{(n-k-m+1)}{(n-m+1)}\leq\left(\frac{n-k}{n}\right)^{m}$$

assuming $n > m, k$ and $n - m > k$.

The last inequality holds because each term on the LHS $\leq \frac{n-k}{n}$, and there are m terms.

3 (b)

Pr(don't know) $\leq \left(\frac{n-k}{n}\right)^{m} = \left(1-\frac{k}{n}\right)^{\frac{n}{k}\frac{km}{n}} \sim e^{-\frac{km}{n}} \leq e^{-10}$ so $k \geq 10\frac{n}{m}$

The approximation depends on n >> k, which only a small fraction of rows is chosen.

3 (c)

Row index	S1	S2
1	0	0
2	1	1
3	0	1
4	1	0

Jaccard similarity = 1 / 3

Pr(S1 and S2 have the same minhash value) = 2 / 4 = 1 / 2

Explanation: The (0, 0) above (1, 1) can amplify the latter. The same is not true for randomly permuted indices, because (0, 0) is equally likely to appear above other rows.

4 (a)

Markov's inequality: If X is a nonnegative random variable and a > 0, we have $\Pr(X \geq a) \leq \frac{E(X)}{a}$.

Therefore, $\Pr\left[\sum |T \cap W_j| \geq 3L\right] \leq \frac{E\left[\sum |T \cap W_j|\right]}{3L}$.

For each data point in T,

$$\Pr\left(x \in W_j | x \in T\right) = \Pr\left(x \in W_j | d_{xz} > c\lambda\right)$$

$$= \Pr\left(h_{j1}(x) = h_{j1}(z), \dots, h_{jk}(x) = h_{jk}(z) | d_{xz} > c\lambda\right)$$

$$\leq p_2^k \quad \text{(because each h is } (\lambda, c\lambda, p_1, p_2)\text{-sensitive)}$$

$$= p_2^{\log_{\frac{1}{p_2}} n} = p_2^{-\log_{p_2} n} = n^{-\log_{p_2} p_2} = n^{-1}$$

Therefore, for any data point, $\Pr(x \in T \cap W_j) \leq n^{-1}$, so $E(|T \cap W_j|) \leq 1$

So, $\Pr\left[\sum |T \cap W_j| \geq 3L\right] \leq \frac{E\left[\sum |T \cap W_j|\right]}{3L} \leq \frac{1}{3}$.

4 (b)

Because each h is $(\lambda, c\lambda, p_1, p_2)$-sensitive, we have

$$\Pr[\forall j, g_j(x^*) \neq g_j(z)] < (1 - p_1^k)^L = \left(1 - p_1^{-\log_{p_2} n}\right)^{n^{\frac{\log p_1}{\log p_2}}} = \left(1 - n^{-\log_{p_2} p_1}\right)^{n^{\log_{p_2} p_1}} < \frac{1}{e}.$$

4 (c)

Pr(reported point is not (c,λ)-ANN)

= Pr(no ANN points is in $\bigcup W_j$ or some ANN points are in $\bigcup W_j$ but none is chosen)

≤Pr(no ANN points is in $\bigcup W_j$) + Pr(some ANN points are in $\bigcup W_j$ but none is chosen)

First term = Pr(no ANN points is in $\bigcup W_j$)

≤Pr(x* is not in $\bigcup W_j$)   (because this event is a super set of the above)

$< \frac{1}{e}$ (by (b))

Second term = Pr(some ANN points are in $\bigcup W_j$ but none is chosen)

≤Pr(at least 3L non-ANN points are in $\bigcup W_j$)   (this event is a super set of the above)

$=\Pr(|T \cap \cup W_j| \geq 3L) = \Pr(|\cup T \cap W_j| \geq 3L)$

$\leq \Pr(\Sigma|T \cap W_j| \geq 3L)$ (upper bound here because $\Sigma|T \cap W_j| \geq |\cup T \cap W_j|$)

$\leq \frac{1}{3}$ (by (a))

Note in this problem, we cannot simply use $\Pr(|\cup T \cap W_j| \leq \Sigma|T \cap W_j| < 3L) > \frac{2}{3}$, because $|\cup T \cap W_j| < 3L$ does not guarantee that there are more ANN points in $\cup W_j$ to choose from.

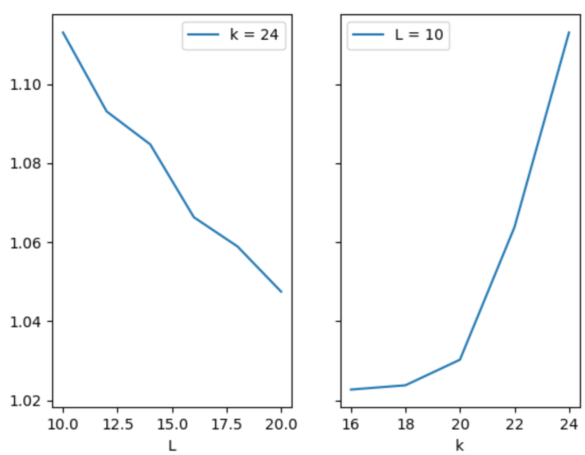
Combining the two terms, we have

Pr(reported point is (c,λ)-ANN) = 1 - Pr(reported point is not (c,λ)-ANN) $> \frac{2}{3} - \frac{1}{e}$.

4 (d)

Code: 1_4.py

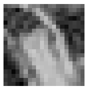
Method	**Search time**
Linear	0.423524
LSH (k=24, L=10) without hashing time	0.115407



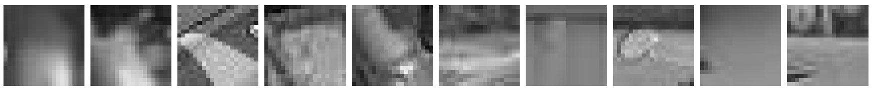
Left: Error decays with increasing L. Reason: $L \uparrow \Rightarrow p_1 \uparrow, p_2 \downarrow \Rightarrow$more sensitive hash functions

Right: Error increases with increasing k. Reason: $k \uparrow \Rightarrow p_2 \uparrow \Rightarrow$less sensitive hash functions

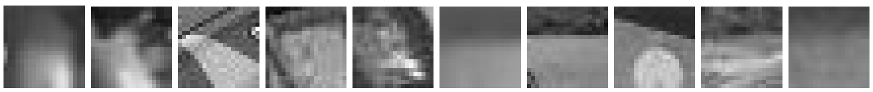
Image 99 (The problem was given with MATLAB in mind, so the 100th image is actually image[99].)



Linear search results



LSH results



The two methods appear comparable in terms of search quality.