CS246: Mining Massive Datasets

Assignment number: 1

Submission time: _____ **and date:** _____

Fill in and include this cover sheet with each of your assignments. It is an honor code violation to write down the wrong time. Assignments are due at 9:30 am, either handed in at the beginning of class or left in the submission box on the 1^{st} floor of the Gates building, near the east entrance. Failure to include the coversheet with you assignment will be penalized by 2 points.

Each student will have a total of *two* free late periods. *One late period expires at the start of each class.* (Assignments are due on Thursdays, which means the first late period expires on the following Tuesday at 9:30am.) Once these late periods are exhausted, any assignments turned in late will be penalized 50% per late period. However, no assignment will be accepted more than *one* late period after its due date. (If an assignment is due to Thursday then we will not accept it after the following Thursday.)

Your name: Blaž Sovdat
Email: blaz.sovdat@gmail.com **SUNet ID:** _____

Collaborators: _____

I acknowledge and accept the Honor Code.

(Signed) _____

(For CS246 staff only)

Late days: 0 1

Section	Score
1	
2	
3	
4	
Total	

Comments:

1 MapReduce

Algorithm. Assuming the input format from HW1 text, the algorithm works as follows.

- Mapper. Given U<TAB>$F_1, \ldots, F_n$, it generates all pairs $(F_i, (U, F_j))$ for $i \neq j$, meaning that F_i and F_j have mutual friend U. Friendless people and people with a single friend require special handling — in such cases the mapper simply passes (U, U) or (F_i, U), just so reducers get all IDs. (See source code for details.)

- Reducer. Given $(F, [(U_1, F_1), \ldots, (U_m, F_m)])$, reducer removes all F_i's that are F's friends — it achieves this by checking whether $F_i = U_j$ for some $i \neq j$ using hash table — and then counts the number of mutual friends F and F_i have in common. It then sorts the resulting list of pairs $(F_i, c(F_i))$, where $c(F_i)$ is the number of friends F and F_i have in common, in decreasing order by $c(F_i)$ and outputs (i.e., recommends) the first ten people from the sorted list: $(F, [F'_1, \ldots, F'_{10}])$. In case of a tie, it sorts by ID's increasingly. (Note that friendless people and people with a single friend require special handling; see source code for details.)

Recommendations. The list of selected recommendations follows:

- 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 Association rules

(a) We consider each measure in turn.

- *Confidence.* The problem with confidence ignoring $\Pr[B]$ is that even if $\text{conf}(A \to B)$ is high, A may have no influence on B, i.e., the fraction of baskets that contain both A and B is roughly the same as the fraction of baskets that contain B. (One solution to this is *interest* of a rule, defined as $\text{conf}(A \to B) - S(B)$.)
- *Lift.* Lift divides confidence of a rule $A \to B$ with $S(B)$, so if A has no influence on B, we get $\text{lift}(A \to B) = 1$. If A has negative influence on B, then lift is less than one; otherwise, if A has positive influence on B, lift is greater than one.
- *Conviction.* If A has no influence on B, then $\text{conv}(A \to B) = 1$. However, if A does influence B, then conviction will be either greater than one (positive influence) or less than one (negative influence).

(b) We consider each measure in turn.

- *Confidence.* Clearly $\text{conf}(A \to B) \neq \text{conf}(B \to A)$, i.e., confidence is not symmetrical. To see this, suppose $\text{Support}(A \cup B) = \text{Support}(A) = 1$ and $\text{Support}(B) = 6$. Then $\text{conf}(A \to B) = 1/6 \neq 1 = \text{conf}(B \to A)$. (For example of such a rule, see [RLU13, example 6.1, page 200], where $\text{conf}(\{cat\} \to kitten) \neq \text{conf}(\{kitten\} \to cat)$.) More intuitively, the probability of seeing B given A is, in general, not the same as the probability of seeing A given B.
- *Lift.* Intuitively, lift is symmetrical because it measures how much more A and B appear together than they would if they were statistically independent. More formally,

$$\begin{aligned}\text{lift}(A \to B) &= \frac{\text{conf}(A \to B)}{S(B)} \\ &= \frac{N \,\text{Support}(A \cup B)}{\text{Support}(A)\,\text{Support}(B)} \\ &= \text{lift}(B \to A)\end{aligned}$$

  implying that lift is symmetrical.
- *Conviction.* Conviction is not symmetrical because it compares "how much A appears without B", which is in general not the same as "how much B appears without A". To get counterexample, let us take the toy data [RLU13, example 6.1, page 200] and consider $\text{conv}(\{cat\} \to dog) = 3/20$ and $\text{conv}(\{dog\} \to cat) = 7/20$.

(c) We consider each measure in turn.

- *Confidence.* First note that $0 \leq \text{conf}(A \to B) \leq 1$. Suppose B appears in all transactions in which A apperas, meaning that rule holds 100% of the time. This means $\text{Support}(A) = \text{Support}(A \cup B)$, implying $\text{conf}(A \to B) = 1$. So confidence is desirable.

- *Lift.* Suppose $A \to B$ always holds. Then $\text{lift}(A \to B) = 1/S(B) = N/\text{Support}(B)$, since we have $\text{conf}(A \to B) = 1$. Suppose now $A' \to B'$ is another (different) rule that always holds and suppose that $\text{Support}(B) \neq \text{Support}(B')$. It follows $\text{lift}(A \to B) \neq \text{lift}(A' \to B')$, meaning that lift is not desirable. Let us again take [RLU13, example 6.1, page 200] and consider $\text{lift}(\{cat\} \to dog) = 1/(7/8) = 8/7$, while $\text{lift}(\{bites\} \to cat) = 1/(6/8) = 8/6$; note that both rules always hold in the data.
- *Conviction.* Suppose $A \to B$ holds all the time. Then $\text{conf}(A \to B) = 1$, giving us $\text{conv}(A \to B) = (1 - S(B))/0 := +\infty$, i.e., we *define* such expressions to have value $+\infty$, which we treat as greater than any real number. This means conv is desirable.

(d) Top 5 rules with corresponding confidence scores in decreasing order of confidence score for itemsets of size 2.

(1) $\{\text{DAI93865}\} \to \{\text{FRO40251}\}$, with confidence 1.0

(2) $\{\text{GRO85051}\} \to \{\text{FRO40251}\}$, with confidence 0.999176276771005

(3) $\{\text{GRO38636}\} \to \{\text{FRO40251}\}$, with confidence 0.9906542056074766

(4) $\{\text{ELE12951}\} \to \{\text{FRO40251}\}$, with confidence 0.9905660377358491

(5) $\{\text{DAI88079}\} \to \{\text{FRO40251}\}$, with confidence 0.9867256637168141

(e) Top 5 rules with corresponding confidence scores in decreasing order of confidence score for itemsets of size 3. (See HW1 text for how break ties.)

(1) $\{\text{DAI23334}, \text{ELE92920}\} \to \{\text{DAI62779}\}$, with confidence 1.0

(2) $\{\text{DAI31081}, \text{GRO85051}\} \to \{\text{FRO40251}\}$, with confidence 1.0

(3) $\{\text{DAI55911}, \text{GRO85051}\} \to \{\text{FRO40251}\}$, with confidence 1.0

(4) $\{\text{DAI62779}, \text{DAI88079}\} \to \{\text{FRO40251}\}$, with confidence 1.0

(5) $\{\text{DAI75645}, \text{GRO85051}\} \to \{\text{FRO40251}\}$, with confidence 1.0

3 Locality-Sensitive Hashing

We define LSH scheme as a family $\mathcal{F}$ of hash functions such that for any two objects x and y we have

$$\Pr_{h \in \mathcal{F}}[h(x) = h(y)] = \text{sim}(x, y),$$

where sim is a similarity function that maps pairs of objects to real numbers from the unit interval.

(a) Let sim be a similarity function and let $d(x, y) := 1 - \text{sim}(x, y)$. Suppose there exist x, y, z such that $d(x, y) + d(y, z) < d(x, z)$, i.e., d violates triangle inequality. First note

that $\Pr_{h\in\mathcal{F}}[h(x) \neq h(y)] = d(x,y)$. We thus have

$$\begin{aligned}\Pr_{h\in\mathcal{F}}[h(x) \neq h(y) \ \vee \ h(y) \neq h(z)] &\leq \Pr_{h\in\mathcal{F}}[h(x) \neq h(y)] + \Pr_{h\in\mathcal{F}}[h(y) \neq h(z)] \\ &< \Pr_{h\in\mathcal{F}}[h(x) \neq h(z)] \leq 1,\end{aligned}$$

which gives us

$$\Pr_{h\in\mathcal{F}}[h(x) = h(y) \wedge h(y) = h(z)] > 1 - \Pr_{h\in\mathcal{F}}[h(x) \neq h(z)] = \Pr_{h\in\mathcal{F}}[h(x) = h(z)].$$

But this is absurd, because '$h(x) = h(z)$' is subevent of '$h(x) = h(y) \wedge h(y) = h(z)$', i.e., the latter implies the former, meaning that the former is at least as likely as the latter. This contradiction establishes our claim.

(b) Now we are going to show that $\mathrm{sim}_{\mathrm{Over}}(A,B) = |A \cap B|/\min(|A|,|B|)$ does not have a LSH scheme. Let $A := \{a_1, a_2, a_3\}$, $C := \{c_1, c_2, c_3\}$ and $B := \{a_1, a_2, c_1, c_2\}$. Then $|A\cap B| = |B\cap C| = 2$ and $|A\cap C| = 0$, while $|A| = |B| = |C| = 3$, so $(1-\mathrm{sim}_{\mathrm{Over}}(A,B))+(1-\mathrm{sim}_{\mathrm{Over}}(B,C)) = 2/3$ and $1-\mathrm{sim}_{\mathrm{Over}}(A,C)) = 1-0 = 1$, violating triangle inequality. By (a) there is no LSH scheme for this measure.

(c) Now we do the same for $\mathrm{sim}_{\mathrm{Dice}}(A,B) = 2|A \cap B|/(|A| + |B|)$. Same as before, let $A := \{a_1, a_2, a_3\}$, $C := \{c_1, c_2, c_3\}$ and $B := \{a_1, a_2, c_1, c_2\}$, so that $1 - \mathrm{sim}_{\mathrm{Dice}}(A,B) = 1-\mathrm{sim}_{\mathrm{Dice}}(B,C) = 3/7$ and $1-\mathrm{sim}_{\mathrm{Dice}}(A,C) = 1$. Since $6/7 < 1$, the function $1-\mathrm{sim}_{\mathrm{Dice}}$ violates triangle inequality and by (a) there can be no LSH scheme for it.

4 LSH for Approximate Near Neighbour Search

In what follows we use notation from HW1 text. We set $[L] := \{1, 2, \ldots, L\}$ for convinience.

(a) Let $W_j := \{x \in \mathcal{A} \mid g_j(x) = g_j(z)\}$ for $1 \leq j \leq L$ and let $T := \{x \in \mathcal{A} \mid d(x,z) > c\lambda\}$. The goal is to prove

$$\Pr\left[\sum_{j=1}^{L} |T \cap W_j| \geq 3L\right] \leq \frac{1}{3},$$

which means $1/3$ is an upper bound on the probability that hash functions give "many" false positives. By Markov's inequality we have

$$\Pr\left[\sum_{j=1}^{L} |T \cap W_j| \geq 3L\right] \leq \frac{1}{3L}\,\mathbb{E}\left[\sum_{j=1}^{L} |T \cap W_j|\right] = \frac{1}{3L}\sum_{j=1}^{L} \mathbb{E}[|T \cap W_j|].$$

(Note that we view $|T\cap W_j|$ as a random variable and that $|T\cap W_j| \geq 0$ and $3L > 0$, thus satisfying conditions for Markov's inequality.) It now suffices to show $\mathbb{E}[|T \cap W_j|] \leq 1$

for all $1 \leq j \leq L$. For each $x_i \in \mathcal{A}$ define $X_i = 1$ if $x_i \in W_j \cap T$ and $X_i = 0$ otherwise, and let $X = X_1 + \ldots + X_n$. We thus have

$$\mathbb{E}[|T \cap W_j|] = \mathbb{E}[X] = \mathbb{E}\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} \mathbb{E}[X_i] \leq np_2^k \leq 1,$$

thus establishing the claim. To see that $np_2^k \leq 1$, suppose $np_2^k > 1$. Then $n > 1/p_2^k$, implying $\log n > k \log(1/p_2) = \log(1/p_2) \frac{\log n}{\log(1/p_2)} = \log n$, which is absurd. (Here we used $k = \log_{1/p_2}(n) = \log n / \log(1/p_2)$.)

(b) Let $x^\star \in \mathcal{A}$ be a point such that $d(x^\star, z) \leq \lambda$. The goal here is to get a lower bound on the probability that at least one hash function maps $x^\star$ and z to the same bucket. Because $\mathcal{G} = \mathcal{H}^k$ and $\mathcal{H}$ is $(\lambda, c\lambda, p_1, p_2)$-sensitive for d, we have $\Pr[g_j(x^\star) = g_j(z)] \geq p_1^k$ for $d(x^\star, z) \leq \lambda$. Furthermore, we have

$$\begin{aligned}\Pr\left[\forall j \in [L] : g_j(x^\star) \neq g_j(z)\right] &= \prod_{j=1}^{L} \Pr[g_j(x^\star) \neq g_j(z)] \\ &< (1 - p_1^k)^L \leq (1 - p_1^k)^{1/p_1^k} \leq 1/e.\end{aligned}$$

Note that $(1-p)^{1/p} \leq 1 - p \leq 1/e$ for $p \in (0, 1)$ follows from the more general inequality $1 - x \leq e^{-x}$ for all $x \in \mathbb{R}$. To see that $L \geq 1/p_1^k$, suppose $L < 1/p_1^k$, implying $\log L < -k \log p_1$, which gives us $\frac{\log(1/p_1)}{\log(1/p_2)} \log n < \frac{\log(1/p_1)}{\log(1/p_2)} \log n$, which is absurd. (Another way to see $f(p) := (1-p)^{1/p} \leq 1/e$ is to note that $\lim_{p \to 0} f(p) = 1/e$ and that f is monotonically decreasing on $(0, 1)$, because $f'(p) < 0$ for all $p \in (0, 1)$.)

(c) We know there exists at least one $x \in \mathcal{A}$ such that $d(x, z) \leq c\lambda$, where $z \in \mathcal{A}$ is the query point. We now establish a lower bound on the probability that the procedure finds (c, λ)-ANN if there is only one such point. Noe that if $g_j(x) = g_j(z)$ for at least one $j \in [L]$ and if $\sum_{j=1}^{L} |W_j \cap T| < 3L$, then the procedure always finds x; we will bound this probability from below. (Otherwise the procedure might or might not find x.) Let $x^\star \in \mathcal{A}$ be a point returned by the procedure. We then have

$$\begin{aligned}\Pr[d(x^\star, z) \leq c\lambda] &\geq \Pr\left[\exists j \in [L] : g_j(x) = g_j(z) \wedge \sum_{j=1}^{L} |W_j \cap T| < 3L\right] \\ &= 1 - \Pr\left[\forall j \in [L] : g_j(x) \neq g_j(z) \vee \sum_{j=1}^{L} |W_j \cap T| \geq 3L\right] \\ &\geq 1 - \left(\Pr[\forall j \in [L] : g_j(x) \neq g_j(z)] + \Pr\left[\sum_{j=1}^{L} |W_j \cap T| \geq 3L\right]\right) \\ &\geq 1 - (1/3 + 1/e) > 1 - 2/e > 1/4,\end{aligned}$$

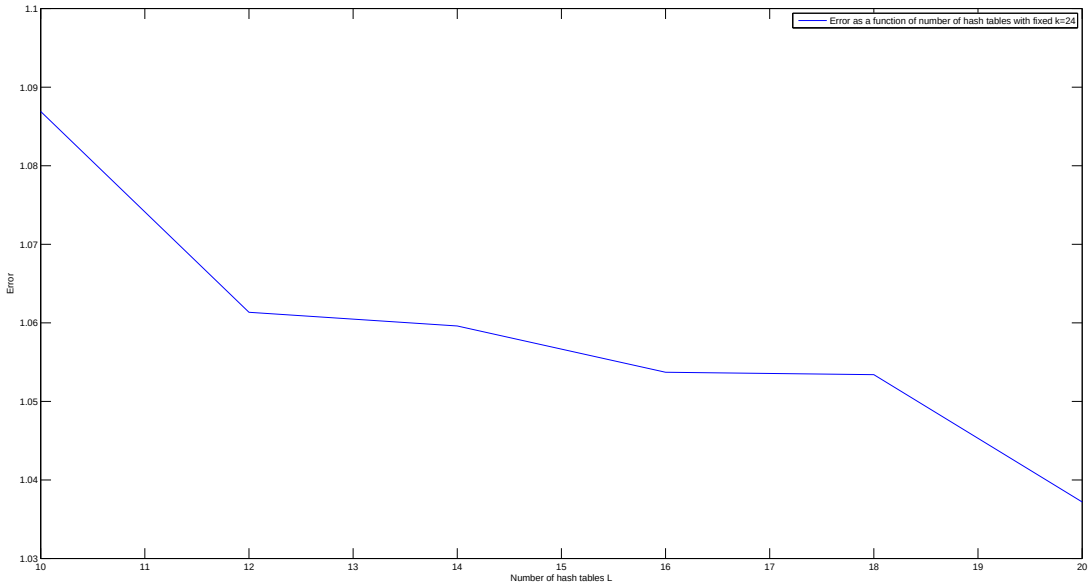

Figure 1: Error plot for $L = 10, 12, \ldots, 20$ with fixed $k = 24$.

meaning that the point the procedure finds is an (c, λ)-ANN with probability greater than $1/4$. (Note that we can amplify this probability by (independently) running the procedure $r > 0$ times and taking the closest $x^\star$ to get $\Pr[d(x^\star, z) \leq c\lambda] > 1 - (3/4)^r$.)

(d) Answers follow.

- Average search time for LSH is 0.0104 seconds; for linear search it is 0.1021 seconds. LSH is roughly ten times faster. See `hwq1i1.m`.
- Figure 1 plots error for various L's with fixed k. We see that error decreases as the number of hashtables increases — the more hashtables we have the more likely we are to map two similar points to the same bucket with at least one hashtable, thus making them a candidate pair. Figure 2 plots error for various k's with fixed L. Here, error increases as k increases — the greater the number of keys the greater the number of possible buckets in a hashatable, meaning we are more likely to hash similar points to different buckets.[1] (See `err_l.m` and `err_k.m` for code that generated the plots.)
- Figure 3 shows NNs found by linear search, while figure 4 shows NNs found by LSH with $L = 10$ and $k = 24$. The query image is shown in figure 5. Although there is barely any visual resemblance between NNs and the query image, some (in this case four) of the NNs found by linear search were also found by LSH — a promising

[1] What we mean is that the hashtable with greater number of possible buckets is more likely — or at least as likely — to produce a false negative than the hashtable with lesser number of possible buckets.

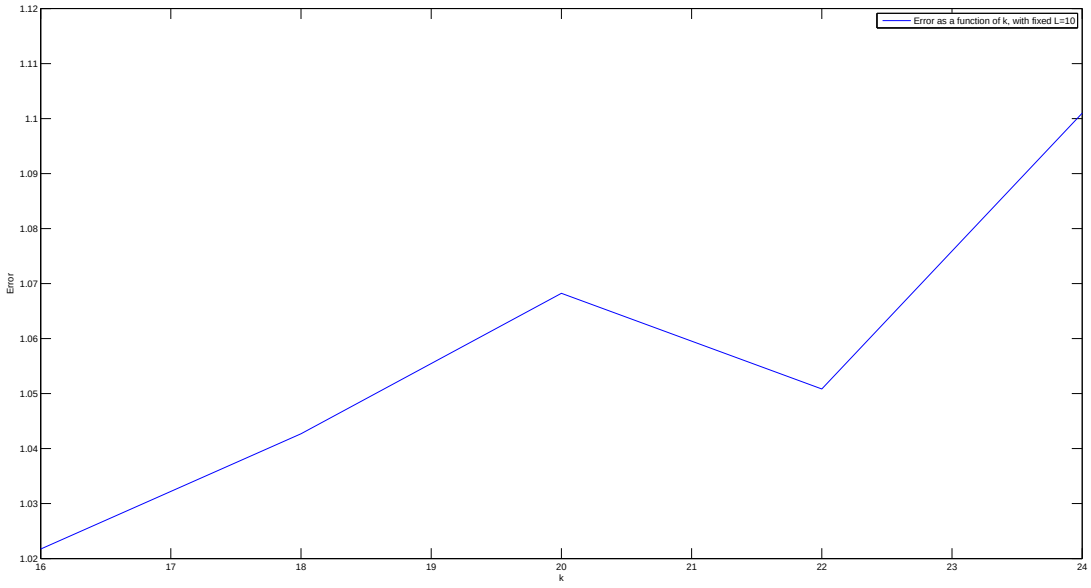

Figure 2: Error plot for $k = 14, 16, \ldots, 24$ with fixed $L = 10$.

result. See `plot10nn.m`. (**Note:** Images in figures 3 and 4 are sorted increasingly by distance from query point, from left to right, top to bottom, i.e., the top leftmost image is the closest one, while the rightmost bottom one is farthest from the query point.)

References

[RLU13] Anand Rajaraman, Jure Leskovec, and Jeffrey D. Ullman. *Mining of Massive Datasets*. Second edition, 2013. Unpublished.

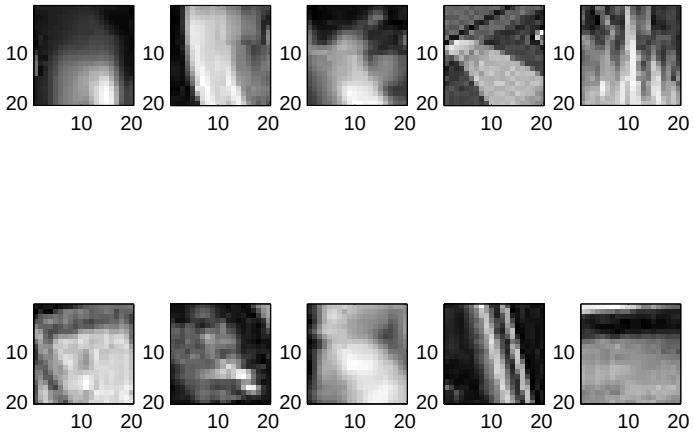

Figure 3: Top 10 NNs returned by linear search.

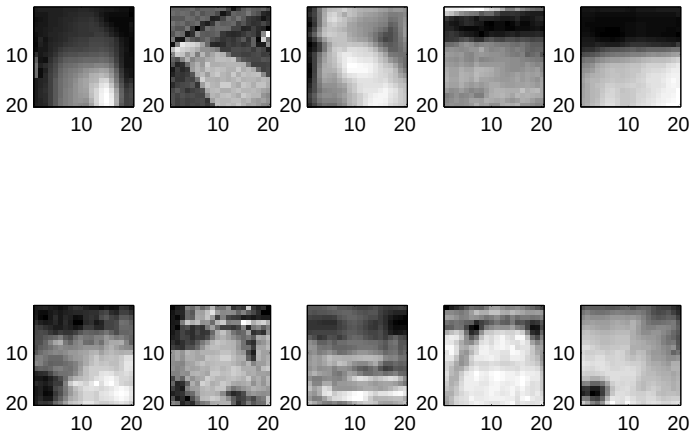

Figure 4: Top 10 NNs returned by LSH.

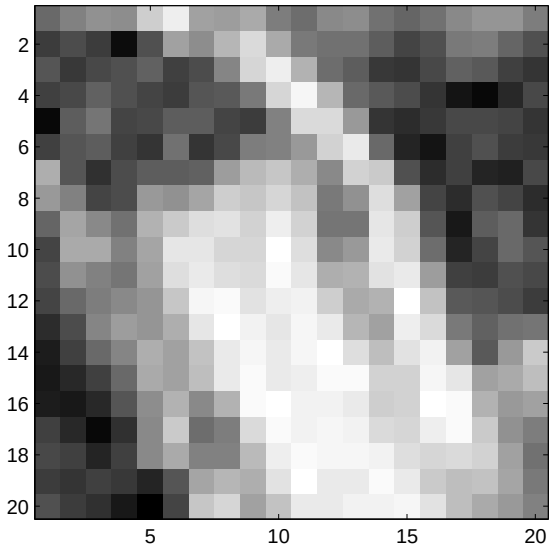

Figure 5: Query image.