New York University Tandon School of Engineering
Computer Science and Engineering

Midterm Exam, Solutions

1. Always, sometimes, never. (12pts – 3pts each)

Indicate whether each of the following statements is **always** true, **sometimes** true, or **never** true. Provide a short justification or example to explain your choice.

(a) For random variables X and Y, $\mathbb{E}[10X - 3Y] = 10\mathbb{E}[X] - 3\mathbb{E}[Y]$.

ALWAYS SOMETIMES NEVER

Follows from linearity of expectation, which holds for any r.v.'s.

(b) If random variables X, Y have the same variance σ^2, the average $Z = \frac{1}{2}(X + Y)$ has $\text{Var}[Z] \geq \frac{1}{2}\sigma^2$.

ALWAYS **SOMETIMES** NEVER

True when $X = Y$ for example, in which case $\text{Var}[Z] = \sigma^2$. Not true when $X = -Y$, in which case $\text{Var}[Z] = 0$.

(c) For a value $z > 0$ and random variable Y that always takes positive integer values $1, 2, 3, \ldots$, we have $\Pr[Y \geq z] \leq \frac{\mathbb{E}[Y]}{z}$.

ALWAYS SOMETIMES NEVER

This is the statement of Markov's inequality. On the exam I forgot to specify $z > 0$, so points were also awarded to SOMETIMES answers.

(d) Increasing the number of tables in a locality sensitive hashing scheme increases the expected number of false negatives.

ALWAYS SOMETIMES **NEVER**

Increasing number of tables strictly decreases false negatives because it strictly increases the chance of checking every point in the database.

(e) Let $X_1, \ldots, X_n$ be random variables (not necessarily independent). $\Pr[\min_j X_j \leq z] \leq \sum_{j=1}^{n} \Pr[X_j \leq z]$.

ALWAYS SOMETIMES NEVER

This is true by union bound. $\Pr[\min_j X_j \leq z] = \Pr[X_1 \leq z]$ and $\ldots$ and $\Pr[X_n \leq z]$.

2. Mark-and-Recapture, The Missing Analysis (15pts)

Recall the mark-and-recapture estimator introduced in Lecture 1. We collect m items $d_1, \ldots, d_m$, where each d_i is selected uniformly at random from the set $\{1, \ldots, n\}$, where n is unknown. To estimate n we return

$$\tilde{n} = \frac{m(m-1)}{2D} \qquad \text{where} \qquad D = \sum_{i=1}^{m} \sum_{j<i} \mathbb{1}[d_i == d_j].$$

In class I claimed that if $m = O(\sqrt{n}/\epsilon)$, the with high probability,

$$(1 - \epsilon)n \leq \tilde{n} \leq (1 + \epsilon)n.$$

However, I never proved that fact. You will fill in the analysis here.

(a) (3pts) Show that $\mathbb{E}[D] = \frac{m(m-1)}{2n}$. We already proved this in class, so I'm asking you to reprove it.

By linearity of expectation,

$$\mathbb{E}[D] = \sum_{i=1}^{m} \sum_{j<i} \mathbb{E}[\mathbb{1}[d_i == d_j]] = \sum_{i=1}^{m} \sum_{j<i} \frac{1}{n}.$$

There are $m(m-1)/2$ terms in the sum, which proves the bound.

(b) (4pts) Show that $\text{Var}[D] \leq \frac{m(m-1)}{2n}$.

In the sum above, the random variables $\mathbb{1}[d_i == d_j]$ are *pairwise* independent. So we can apply linearity of variance:

$$\text{Var}[D] = \sum_{i=1}^{m}\sum_{j<i}\text{Var}[\mathbb{1}[d_i == d_j]] = \sum_{i=1}^{m}\sum_{j<i}\left(\frac{1}{n} - \frac{1}{n^2}\right) \leq \sum_{i=1}^{m}\sum_{j<i}\frac{1}{n} = \frac{m(m-1)}{2n}.$$

(c) (5pts) Prove that if $m = c\sqrt{n}/\epsilon$ for a large enough constant c, then

$$\Pr[|D - \mathbb{E}[D]| \geq \epsilon\mathbb{E}[D]] \leq 1/100.$$

Will will apply Chebyshev's inequality. From (b) we have that the standard deviation of D is upper bounded by $\sigma = \sqrt{\frac{m(m-1)}{2n}} \leq \frac{m}{\sqrt{2n}}$, which is $\leq \frac{c/\sqrt{2}}{\epsilon}$ when $m = c\sqrt{n}/\epsilon$. So by Chebyshev's Inequality we have that:

$$\Pr[|D - \mathbb{E}[D]| \geq \frac{10c/\sqrt{2}}{\epsilon}] \leq 1/100.$$

At the same time, we have that $\mathbb{E}[D] = \frac{m(m-1)}{2n} \geq \frac{m^2}{4n} = \frac{c^2}{4\epsilon^2}$. Combing with the bound above, we have

$$\Pr[|D - \mathbb{E}[D]| \geq \epsilon\frac{40}{c\sqrt{2}}\mathbb{E}[D]] \leq 1/100.$$

Setting $c \geq 40/\sqrt{2}$ gives the bound.

(d) (3pts) Conclude that if $m = O(\sqrt{n}/\epsilon)$, then with probability 99/100,

$$(1-\epsilon)n \leq \tilde{n} \leq (1+\epsilon)n.$$

Hint: You may assume that $\epsilon \leq 1/2$, and use the fact that in that case $\frac{1}{1-\epsilon} \leq 1 + 2\epsilon$ and $\frac{1}{1+\epsilon} \geq 1 - \epsilon$.

We have from above that that if $m = O(\sqrt{n}/\epsilon)$ then:

$$(1-\epsilon/2)\mathbb{E}[D] \leq D \leq (1+\epsilon/2)\mathbb{E}[D].$$

Inverting all sides we have:

$$\begin{aligned}
\frac{1}{1+\epsilon/2}\frac{1}{\mathbb{E}[D]} \leq \frac{1}{D} \leq \frac{1}{1-\epsilon/2}\frac{1}{\mathbb{E}[D]} \\
(1-\epsilon)\frac{1}{\mathbb{E}[D]} \leq \frac{1}{D} \leq (1+\epsilon)\frac{1}{\mathbb{E}[D]} \\
(1-\epsilon)\frac{m(m-1)}{2\mathbb{E}[D]} \leq \frac{m(m-1)}{2D} \leq (1+\epsilon)\frac{m(m-1)}{2\mathbb{E}[D]} \\
(1-\epsilon)n \leq \tilde{n} \leq (1+\epsilon)n.
\end{aligned}$$

3. Locality sensitive hash for hamming similarity. (10pts)

The giant internet company Popflix operates a movie streaming service that offers a total of d movies. They keep track of the movies viewed by each subscriber to the service by storing a length d binary vector with a 1 in all entries corresponding to movies that subscriber has watched. All other entries are set to 0. Popflix would like to efficiently find other subscribers with similar viewing habits using an LSH scheme. For two length d binary vectors $\mathbf{q}, \mathbf{y} \in \{0,1\}^d$, they model viewing similarity by the hamming similarity:

$$s(\mathbf{x}, \mathbf{y}) = 1 - \frac{\|\mathbf{q} - \mathbf{y}\|_0}{d}.$$

Above $\|\mathbf{q} - \mathbf{y}\|_0$ is the hamming distance, $\|\mathbf{q} - \mathbf{y}\|_0 = \sum_{i=1}^{d}|q_i - y_i|$. q_i and y_i denote the i^{th} entries of $\mathbf{q}$ and $\mathbf{y}$, respectively. For example, the hamming similarity between the follow two vectors is 2/3.

$$\begin{aligned}
\mathbf{q} &= [1, 0, 0, 1, 1, 0] \\
\mathbf{y} &= [1, 0, 1, 1, 0, 0]
\end{aligned}$$

(a) (6pts) To implement their LSH scheme, Popflix first needs a locality sensitive hash function $h : \{0,1\}^d \to \{1,\ldots,n\}$ that maps user vectors into a table of size n. We construct h as follows: Let j be a uniform random integer in $\{1,\ldots,d\}$ and define the function $c : \{0,1\}^d \to \{0,1\}$ as $c(\mathbf{x}) = x_j$. Additionally, let g be a uniform random hash function from $\{0,1\} \to \{1,\ldots,n\}$. Finally, let:

$$h(\mathbf{x}) = g(c(\mathbf{x})).$$

Prove that h is a locality sensitive hash function for hamming similarity. **Hint:** Write down an expression for $\Pr[h(\mathbf{q}) == h(\mathbf{y})]$.

Note that for binary vectors $\mathbf{q}, \mathbf{y}$, $\|\mathbf{q}-\mathbf{y}\|_0$ is exactly the number of entries where the vectors *differ*. $d - \|\mathbf{q}-\mathbf{y}\|_0$ is the number of entries where the vectors are the same. So we have, for any binary $\mathbf{q}, \mathbf{y}$

$$Pr[h(\mathbf{q}) == h(\mathbf{y})] = \Pr[q_g = y_g] = \frac{d - \|\mathbf{q}-\mathbf{y}\|_1}{d} = 1 - \frac{\|\mathbf{q}-\mathbf{y}\|_1}{d}.$$

So our collision probability is *exactly proportional* to the hamming similarity. It increases when similarity increases, and decreases when it decreases. So h is a locality sensitive hash function.

(b) (4pts) What are **two reasons** Popflix might **not want to use** locality sensitive hashing for similarity search, and might instead perform a linear scan.

- LSH requires preprocessing time to hash all database elements into multiple tables. This is slower than a linear scan. So you only save time with LSH if you have many queries.
- LSH requires more space.
- LSH always fails with some probability – it can never be 100% reliable.

4. Regularization (15pts)

Regularization is a popular technique in machine learning. It is often used to improve final test error, but can also help speed-up optimization methods like gradient descent by improving the condition number of the function being regularized.

(a) (5pts) Let $f(\mathbf{x})$ be a differentiable function mapping a length d vector $\mathbf{x}$ to a scalar value. Let g be the function with added Euclidean regularization:

$$g(\mathbf{x}) = f(\mathbf{x}) + \lambda\|\mathbf{x}\|_2^2$$

Above $\lambda > 0$ is a non-negative constant that controls the amount of regularization. Write down an expression for the gradient of g, $\nabla g(\mathbf{x})$, in terms of $\nabla f(\mathbf{x})$, λ, and $\mathbf{x}$.

We applying linearity of the gradient. $\nabla g(\mathbf{x}) = \nabla f(\mathbf{x}) + \lambda\nabla\left[\|\mathbf{x}\|_2^2\right]$. We have that $\|\mathbf{x}\|_2^2 = \sum_{i=1}^n x_i^2$, so the partial derivative with respect to i is $2x_i$. So $\nabla\left[\|\mathbf{x}\|_2^2\right] = 2\mathbf{x}$ and overall we have:

$$\nabla g(\mathbf{x}) = \nabla f(\mathbf{x}) + 2\lambda\mathbf{x}.$$

(b) (5pts) Prove that $h(\mathbf{x}) = \lambda\|\mathbf{x}\|_2^2$ is an α_1-strongly convex, β_1-smooth function with $\alpha_1 = \beta_1 = 2\lambda$. We use the first order definition of smoothness and strong convexity:

$$\frac{\alpha}{2}\|\mathbf{x}-\mathbf{y}\|_2^2 \leq h(\mathbf{y}) - h(\mathbf{x}) - \nabla h(\mathbf{x})^T(\mathbf{y}-\mathbf{x}) \leq \frac{\beta}{2}\|\mathbf{x}-\mathbf{y}\|_2^2.$$

Consider the middle term.

$$\begin{aligned} h(\mathbf{y}) - h(\mathbf{x}) - \nabla h(\mathbf{x})^T(\mathbf{y}-\mathbf{x}) &= \lambda\cdot\left(\|\mathbf{y}\|_2^2 - \|\mathbf{x}\|_2^2 - 2\mathbf{x}^T\mathbf{y} + 2\|\mathbf{x}\|_2^2\right) \\ &= \lambda\cdot\left(\|\mathbf{y}\|_2^2 + \|\mathbf{x}\|_2^2 - 2\mathbf{x}^T\mathbf{y}\right) = \lambda\|\mathbf{x}-\mathbf{y}\|_2^2. \end{aligned}$$

So we can talk $\alpha = \beta = 2\lambda$ to make the above inequality true.

(c) (5pts) Assume that f is α_2-strongly convex and β_2-smooth where $\beta_2 > \alpha_2$. Using part (b), prove that the regularized function $g(\mathbf{x})$ has strictly better condition number than $f(\mathbf{x})$.

Hint: As a first step, show that when you add an α_1-strongly convex, β_1-smooth function to an α_2-strongly convex, β_2-smooth function, then the resulting function is $(\alpha_1 + \alpha_2)$-strongly-convex and $(\beta_1 + \beta_2)$-smooth.

The first step from the hint follows directly from the first order definition above. If we have both

$$\frac{\alpha_g}{2}\|\mathbf{x}-\mathbf{y}\|_2^2 \leq g(\mathbf{y}) - g(\mathbf{x}) - \nabla g(\mathbf{x})^T(\mathbf{y}-\mathbf{x}) \leq \frac{\beta_g}{2}\|\mathbf{x}-\mathbf{y}\|_2^2$$
$$\frac{\alpha_h}{2}\|\mathbf{x}-\mathbf{y}\|_2^2 \leq h(\mathbf{y}) - h(\mathbf{x}) - \nabla h(\mathbf{x})^T(\mathbf{y}-\mathbf{x}) \leq \frac{\beta_h}{2}\|\mathbf{x}-\mathbf{y}\|_2^2,$$

then summing the inequalities gives

$$\frac{\alpha_g + \alpha_h}{2}\|\mathbf{x}-\mathbf{y}\|_2^2 \leq f(\mathbf{y}) - f(\mathbf{x}) - \nabla f(\mathbf{x})^T(\mathbf{y}-\mathbf{x}) \leq \frac{\beta_g + \beta_h}{2}\|\mathbf{x}-\mathbf{y}\|_2^2,$$

where $f = g + h$.

So we have that the regularized function has condition number:

$$\frac{\beta_2 + 2\lambda}{\alpha_2 + 2\lambda}.$$

We claim that this is always *smaller* than $\frac{\beta_2}{\alpha_2}$, which means that the regularized function is *better* conditioned than f. To see why this is the case observe:

$$\frac{\beta_2 + 2\lambda}{\alpha_2 + 2\lambda} \leq \frac{\beta_2 + 2\frac{\beta_2}{\alpha_2}\lambda}{\alpha_2 + 2\lambda} = \frac{\beta_2}{\alpha_2}.$$

The inqeuality follows from the fact that $\lambda \geq 0$ and $\beta_2 \geq \alpha_2$.