COMPSCI 514: Algorithms for Data Science

Cameron Musco

University of Massachusetts Amherst. Spring 2026.

Lecture 10

Logistics

- The midterm is this Thursday in class, 1-2:15pm.
- Closed book – no cheatsheets, calculators or other aids allowed.
- Study guide is posted on the course webpage (under the midterm row in the schedule tab).
- Past midterms are posted in Canvas. Note that some cover more material than we have seen – e.g., we will not cover the Johnson-Lindenstrauss lemma or low-distortion embeddings before Midterm 1.
- I will hold regular office hours today after class, and additional review office hours **Wednesday, 11am-12pm**.
- New material from today will not appear on the midterm.

Summary

Last Class:

- The similarity search problem.
- Locality sensitive hashing for fast similarity search.
- MinHash as a locality sensitive hash function for Jaccard similarity
- Balancing false positives and negatives with LSH signatures and repeated hash tables.

This Class:

- Finish up LSH – SimHash for cosine similarity.
- Midterm review.

Fast Similarity Search

Have a database of items – e.g., documents, images, audio clips, etc.

Define a similarity metric over these items – e.g., cosine similarity (dot product) if they are represented as vectors, or Jaccard similarity if represented as sets.

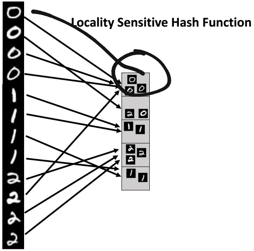
Want Fast Implementations For:

- **Near Neighbor Search:** Given a query item q, find if it has high similarity to any database item. $\Omega(n)$ time with a linear scan.
- **All-pairs Similarity Search:** Have n different query vectors and want to find all pairs with high similarity. $\Omega(n^2)$ time if we check all pairs explicitly.

Difficulty is that q almost never has an exact match in the database – if it did we could solve search in $O(1)$ time using a hash table.

LSH For Similarity Search

Key Idea: Design a locality sensitive hash function where the collision probability is higher when two inputs are more similar.



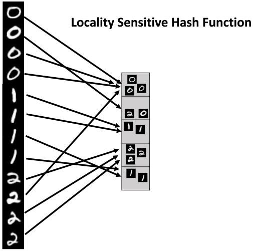
- Given item x, compute $\mathbf{h}(x)$. Only search for similar items in the $\mathbf{h}(x)$ bucket of the hash table.

LSH For Similarity Search

Key Idea: Design a locality sensitive hash function where the collision probability is higher when two inputs are more similar.

$MH(x)$

$[MH_1(x), MH_2(x) \ldots]$

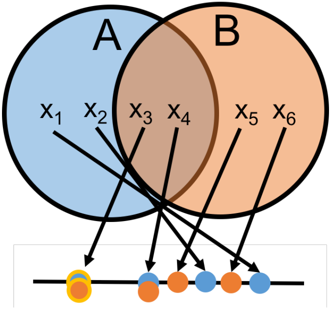


- Given item x, compute $\mathbf{h}(x)$. Only search for similar items in the $\mathbf{h}(x)$ bucket of the hash table.
- **Last Class:** Can boost success probability and reduce query time via repetition – using multiple hash tables and signatures of multiple hash values.
- Analyze the hit probability via the s-curve.

MinHashing

For the Jaccard similarity between sets, $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$, we used MinHashing to give a locality sensitive hash function satisfying:

$$\Pr[\underbrace{MH(A) = MH(B)}_{\text{collision}}] = J(A, B).$$



Generalizing Locality Sensitive Hashing

Repetition and s-curve tuning can be used for fast similarity search with any similarity metric, given a locality sensitive hash function for that metric.

Generalizing Locality Sensitive Hashing

Repetition and *s*-curve tuning can be used for fast similarity search with any similarity metric, given a locality sensitive hash function for that metric.

- LSH schemes exist for many similarity/distance measures: hamming distance, cosine similarity, etc.

Generalizing Locality Sensitive Hashing

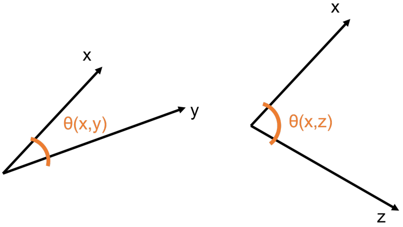
Repetition and s-curve tuning can be used for fast similarity search with any similarity metric, given a locality sensitive hash function for that metric.

- LSH schemes exist for many similarity/distance measures: hamming distance, cosine similarity, etc.

Generalizing Locality Sensitive Hashing

Repetition and *s*-curve tuning can be used for fast similarity search with any similarity metric, given a locality sensitive hash function for that metric.

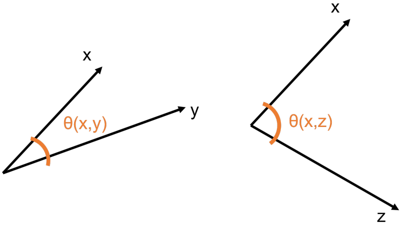
- LSH schemes exist for many similarity/distance measures: hamming distance, cosine similarity, etc.



Generalizing Locality Sensitive Hashing

Repetition and *s*-curve tuning can be used for fast similarity search with any similarity metric, given a locality sensitive hash function for that metric.

- LSH schemes exist for many similarity/distance measures: hamming distance, cosine similarity, etc.



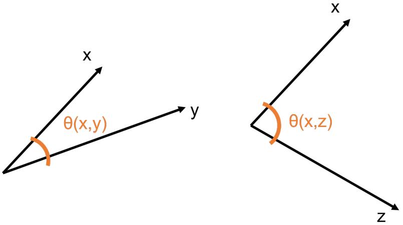
Cosine Similarity: $\cos(\theta(x,y))$

Generalizing Locality Sensitive Hashing

Repetition and *s*-curve tuning can be used for fast similarity search with any similarity metric, given a locality sensitive hash function for that metric.

- LSH schemes exist for many similarity/distance measures: hamming distance, cosine similarity, etc.

$$\frac{\langle x, y \rangle}{\|x\|_2 \cdot \|y\|_2}$$



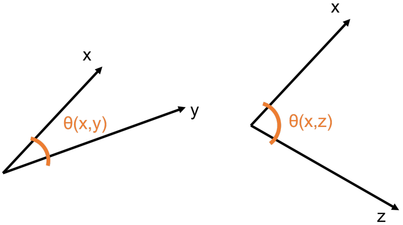
Cosine Similarity: $\cos(\theta(x, y))$ $[-1, 1]$

- $\cos(\theta(x, y)) = 1$ when $\theta(x, y) = 0°$ and $\cos(\theta(x, y)) = 0$ when $\theta(x, y) = 90°$, and $\cos(\theta(x, y)) = -1$ when $\theta(x, y) = 180°$

Generalizing Locality Sensitive Hashing

Repetition and s-curve tuning can be used for fast similarity search with any similarity metric, given a locality sensitive hash function for that metric.

- LSH schemes exist for many similarity/distance measures: hamming distance, cosine similarity, etc.



Cosine Similarity: $\cos(\theta(x,y)) = \frac{\langle x,y\rangle}{\|x\|_2 \cdot \|y\|_2}$.

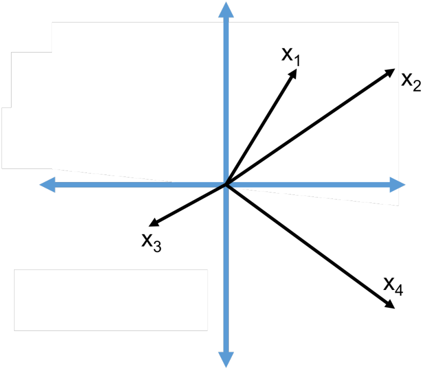
- $\cos(\theta(x,y)) = 1$ when $\theta(x,y) = 0°$ and $\cos(\theta(x,y)) = 0$ when $\theta(x,y) = 90°$, and $\cos(\theta(x,y)) = -1$ when $\theta(x,y) = 180°$

SimHash for Cosine Similarity

SimHash Algorithm: LSH for cosine similarity.

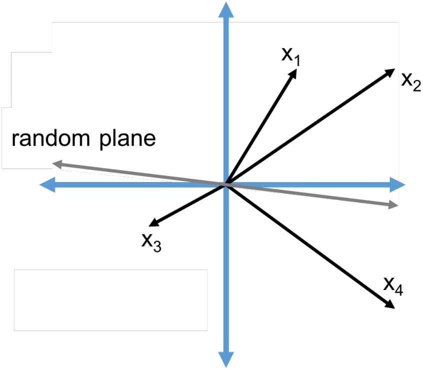
SimHash for Cosine Similarity

SimHash Algorithm: LSH for cosine similarity.



SimHash for Cosine Similarity

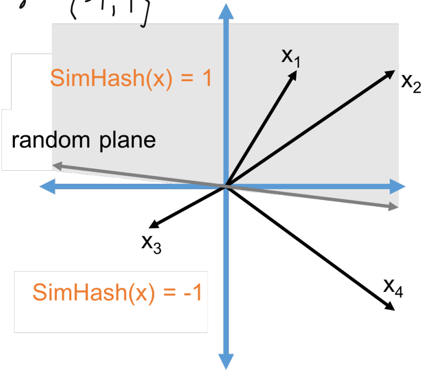
SimHash Algorithm: LSH for cosine similarity.



SimHash for Cosine Similarity

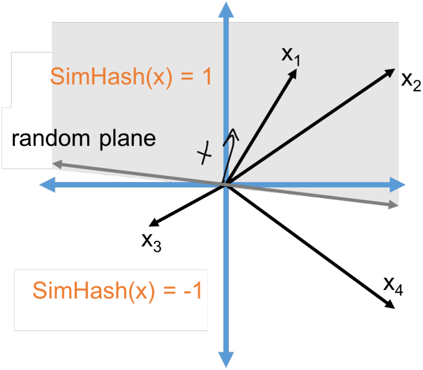
SimHash Algorithm: LSH for cosine similarity.

$\text{SimHash} : \mathbb{R}^d \to \{-1, 1\}$



SimHash for Cosine Similarity

SimHash Algorithm: LSH for cosine similarity.



normal vector of the plane

$SimHash(x) = \text{sign}(\langle x, t \rangle)$ for a random vector t.

SimHash for Cosine Similarity

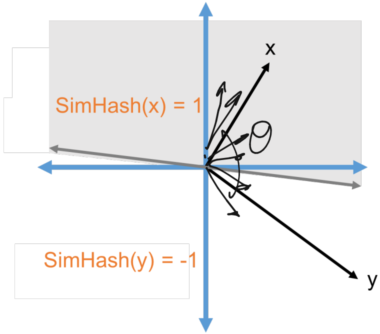
What is $\Pr\left[SimHash(x) = SimHash(y)\right]$?

SimHash for Cosine Similarity

What is $\Pr[SimHash(x) = SimHash(y)]$?

$SimHash(x) \neq SimHash(y)$ when the plane separates x from y.

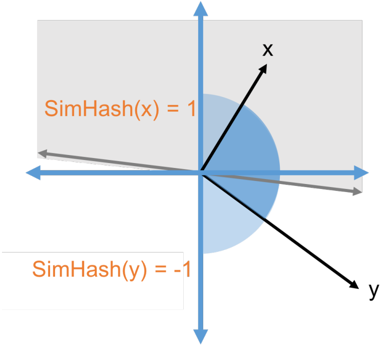
$$\frac{\theta}{180^\circ}$$



SimHash for Cosine Similarity

What is $\Pr[SimHash(x) = SimHash(y)]$?

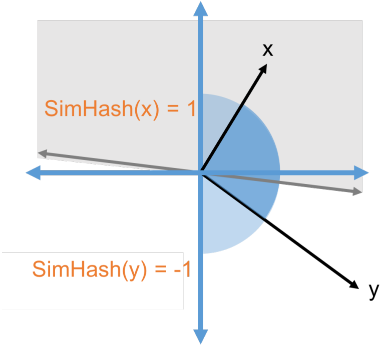
$SimHash(x) \neq SimHash(y)$ when the plane separates x from y.



SimHash for Cosine Similarity

What is $\Pr[SimHash(x) = SimHash(y)]$?

$SimHash(x) \neq SimHash(y)$ when the plane separates x from y.



- $\Pr[SimHash(x) \neq SimHash(y)] = \frac{\theta(x,y)}{\pi}$

SimHash for Cosine Similarity

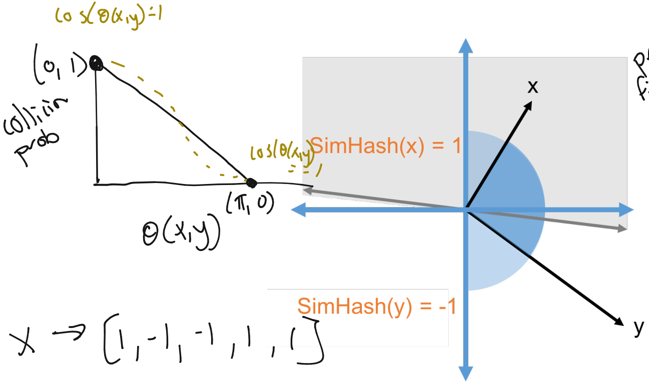
What is $\Pr[SimHash(x) = SimHash(y)]$?

t = # tables

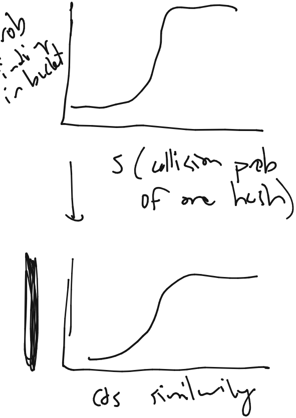
r: signature length

$1-(1-s^r)^t$

$SimHash(x) \neq SimHash(y)$ when the plane separates x from y.



$x \to [1, -1, -1, 1, 1]$



- $\Pr[SimHash(x) \neq SimHash(y)] = \frac{\theta(x,y)}{\pi}$
- $\Pr[SimHash(x) = SimHash(y)] = 1 - \frac{\theta(x,y)}{\pi} \approx \frac{\cos(\theta(x,y))+1}{2}$

Questions on MinHash and Locality Sensitive Hashing?

Midterm Review

Studying

Suggested Studying Approach:

- Review the study guide to get a sense of what you need to know, and then mostly focus on doing practice questions from the past midterms and the study guide.
- Review slides only as needed.
- If you get nervous during exams, do some practice exams under time constraints with no material in front of you.
- Note that many of the past exams were 2 hours long, but designed to be completed in 90 minutes. This exam will be a bit shorter.

Midterm Format

Rough Outline: (subject to changes)

- Question 1: 4-5 True/False questions. No justification needed.
- Question 2: 4-5 numerical answers, like quiz questions. No justification needed.
- Question 3: 4-5 part question on analyzing an algorithm. Similar in style to but easier than a homework question.
- Question 4: More challenging 4-5 part question on analyzing an algorithm – more similar to a homework question.
- Potentially some extra credit subquestions on Q3/Q4.

Questions

Content or Format Questions?

median trick

hash function → which is needed

Questions

Questions

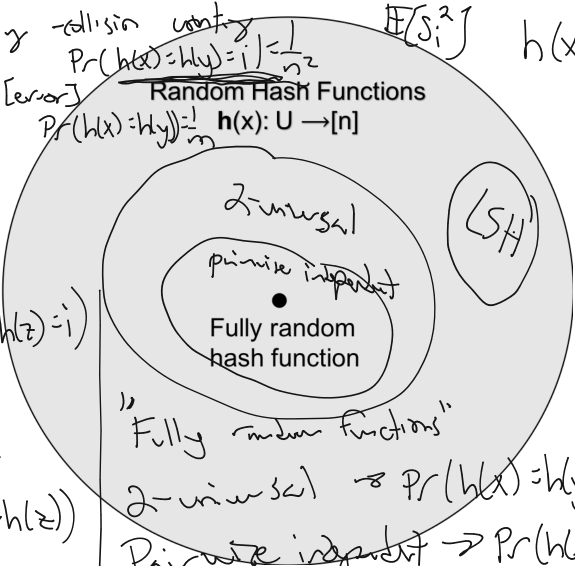
Random Hash Functions

2-level hashing - collision counting $\Pr(h(x)=h(y)=i)=\frac{1}{n^2}$ $\mathbb{E}[S_i^2]$ $h(x)=ax+b \bmod p$

Count-min $\mathbb{E}[\text{error}]$

Bloom filters $\Pr(h(x)=h(y))=\frac{1}{m}$

$$\left(1-\frac{1}{m}\right)^{nk}$$



$\Pr(h(x)=h(y)=h(z)=i)$
$=\frac{1}{n^3}$

for random

$\Pr(h(x)=h(y)=h(z))$
$=\frac{1}{n^2}$

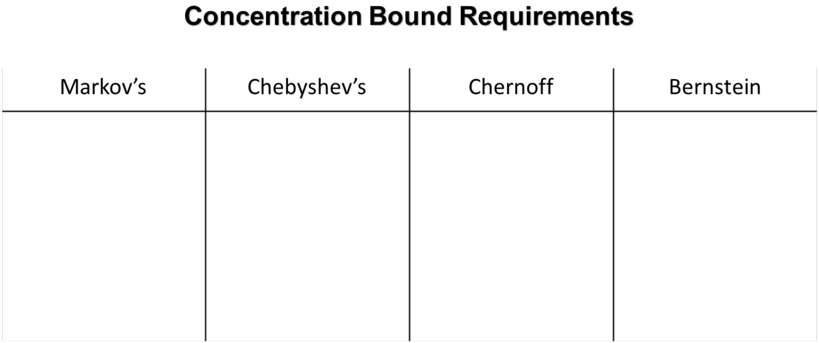
"Fully random functions"

2-universal → $\Pr(h(x)=h(y))\leq\frac{1}{n}$

Pairwise independent → $\Pr(h(x)=h(y)=i)=\frac{1}{n^2}$

LSH $\Pr(h(x)=h(y))$ = depends on similarity

Concentration Bounds



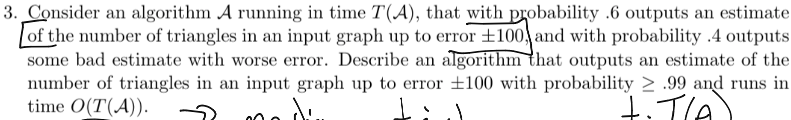
Concentration Bound Requirements

Markov's	Chebyshev's	Chernoff	Bernstein

Sampling For Mean Estimation

Say I have n numbers $x_1, \ldots, x_n$ all lying in $[-M, M]$ with mean $\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$. How can I estimate μ without reading all the numbers?

Median Trick

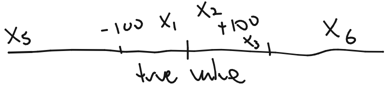


3. Consider an algorithm $\mathcal{A}$ running in time $T(\mathcal{A})$, that with probability .6 outputs an estimate of the number of triangles in an input graph up to error ±100, and with probability .4 outputs some bad estimate with worse error. Describe an algorithm that outputs an estimate of the number of triangles in an input graph up to error ±100 with probability ≥ .99 and runs in time $O(T(\mathcal{A}))$.

→ median trick $t \cdot T(A)$

- run algorithm t times, t estimates $X_1, X_2 \dots X_t$
- output median$(X_1, \dots X_t)$

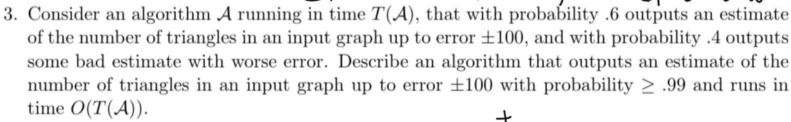
- X_i is "good" w.p. .6, X_i is "bad" otherwise
- what is prob median is "good" → median is good if ≥ .5t trials are good



The Chernoff bound states that for independent random variables $X_1, \dots, X_n$ taking values in $\{0,1\}$, letting $\mu = \mathbb{E}\left[\sum_{i=1}^n X_i\right]$, for any $\delta > 0$, $\Pr\left(\left|\sum_{i=1}^n X_i - \mu\right| > \delta\mu\right) \le 2\exp\left(-\frac{\delta^2\mu}{2+\delta}\right)$.

Median Trick

$I_i = 1$ if X_i is good $I_i = 0$ o.w.



3. Consider an algorithm $\mathcal{A}$ running in time $T(\mathcal{A})$, that with probability .6 outputs an estimate of the number of triangles in an input graph up to error ±100, and with probability .4 outputs some bad estimate with worse error. Describe an algorithm that outputs an estimate of the number of triangles in an input graph up to error ±100 with probability $\geq .99$ and runs in time $O(T(\mathcal{A}))$.

$$Pr(\text{median is good}) = Pr\left(\sum_{i=1}^{t} I_i \geq .5t\right) \geq .\,.99$$

$$\mathbb{E}\left[\sum_{i=1}^{t} I_i\right] = .6t$$

$\mu = .6t \quad \delta = 1/6$

$$Pr\left(\sum_{i=1}^{t} I_i < .5t\right) \leq Pr\left(\left|\sum I_i - \mathbb{E} I_i\right| \geq .1t\right)$$

$.1t = \delta \cdot \mu$

$.1t = \delta \cdot .6t \quad \delta = 1/6$

$$Pr\left(\left|\sum I_i - \mu\right| \geq \frac{1}{6} \cdot \mu\right) < 2\exp\left(\frac{-\delta^2 \mu}{2+\delta}\right)$$

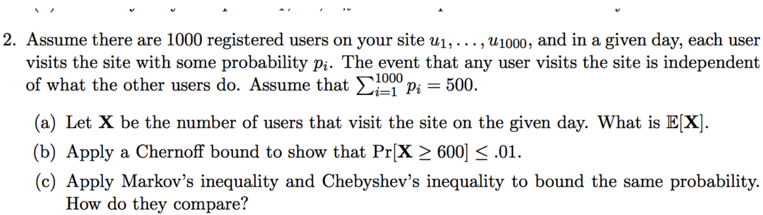
$$2\exp\left(\frac{-1/6^2 \cdot .6 \cdot t}{2+1/6}\right)$$

The Chernoff bound states that for independent random variables $X_1, \ldots, X_n$ taking values in $\{0, 1\}$, letting $\mu = \mathbb{E}\left[\sum_{i=1}^{n} X_i\right]$, for any $\delta > 0$,
$\Pr\left(\left|\sum_{i=1}^{n} X_i - \mu\right| > \delta\mu\right) \leq 2\exp\left(-\frac{\delta^2 \mu}{2+\delta}\right)$.

$2\exp(-ct) \leq .01$

$t = O(1)$

Example Problems



2. Assume there are 1000 registered users on your site $u_1, \ldots, u_{1000}$, and in a given day, each user visits the site with some probability p_i. The event that any user visits the site is independent of what the other users do. Assume that $\sum_{i=1}^{1000} p_i = 500$.

   (a) Let $\mathbf{X}$ be the number of users that visit the site on the given day. What is $\mathbb{E}[\mathbf{X}]$.

   (b) Apply a Chernoff bound to show that $\Pr[\mathbf{X} \geq 600] \leq .01$.

   (c) Apply Markov's inequality and Chebyshev's inequality to bound the same probability. How do they compare?

$$2\exp(-ct) \le \delta$$

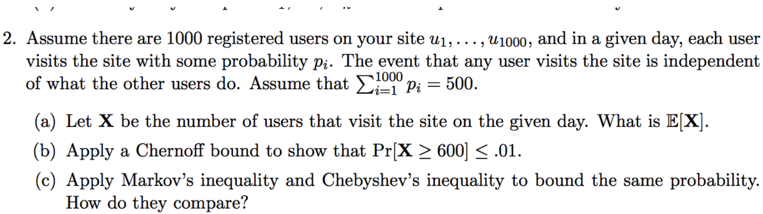
$$\exp(-ct) \le \delta/2$$

$$-ct \le \log(\delta/2)$$

$$t \ge \frac{-\log(\delta/2)}{c} = \frac{\log(2/\delta)}{c} = O(\log(1/\delta))$$

The Chernoff bound states that for independent random variables $X_1, \ldots, X_n$ taking values in $\{0,1\}$, letting $\mu = \mathbb{E}\left[\sum_{i=1}^n X_i\right]$, for any $\delta > 0$,
$\Pr\left(\left|\sum_{i=1}^n X_i - \mu\right| > \delta\mu\right) \le 2\exp\left(-\frac{\delta^2\mu}{2+\delta}\right)$.

Example Problems



2. Assume there are 1000 registered users on your site $u_1, \ldots, u_{1000}$, and in a given day, each user visits the site with some probability p_i. The event that any user visits the site is independent of what the other users do. Assume that $\sum_{i=1}^{1000} p_i = 500$.

   (a) Let $\mathbf{X}$ be the number of users that visit the site on the given day. What is $\mathbb{E}[\mathbf{X}]$.

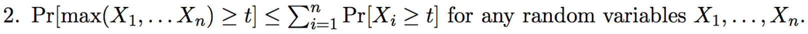
   (b) Apply a Chernoff bound to show that $\Pr[\mathbf{X} \geq 600] \leq .01$.

   (c) Apply Markov's inequality and Chebyshev's inequality to bound the same probability. How do they compare?

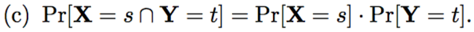
The Chernoff bound states that for independent random variables $X_1, \ldots, X_n$ taking values in $\{0,1\}$, letting $\mu = \mathbb{E}\left[\sum_{i=1}^{n} X_i\right]$, for any $\delta > 0$, $\Pr\left(\left|\sum_{i=1}^{n} X_i - \mu\right| > \delta\mu\right) \leq 2\exp\left(-\frac{\delta^2 \mu}{2+\delta}\right)$.

Example Problems

TRUE or FALSE:



2. $\Pr[\max(X_1, \ldots X_n) \geq t] \leq \sum_{i=1}^{n} \Pr[X_i \geq t]$ for any random variables $X_1, \ldots, X_n$.



(c) $\Pr[\mathbf{X} = s \cap \mathbf{Y} = t] = \Pr[\mathbf{X} = s] \cdot \Pr[\mathbf{Y} = t]$.