Aughdon Breslin

Professor Jie Shen

CS541 Artificial Intelligence

11/1/2022 - Lecture 7

Random Projection

We have a vector $v \in R^d$ where $d = 1M$, want to reduce the dimensions

- "Lets just keep the first 10" $\rightarrow d' = 10$. Maybe not the best
- Want to reduce from R^d to R^k:
  - Make matrix $M \in R^{k \times d}$
  - $v \rightarrow M * v \rightarrow$ k-dimensions

We want $||A' - B'|| \approx ||A - B||$ where $\epsilon \in (0, 1)$.

- $(1 - \epsilon) * ||A - B|| \leq ||A' - B'|| \leq (1 + \epsilon) * ||A - B||$

Making M

- $M \sim \mathcal{N}(0, 1) : k \times d \{v_1 \dots v_n\}$ where n is the sample size.

Algorithm

1. $M_{ij} \sim \mathcal{N}(0, 1)$.
2. $v_i \rightarrow M * v_i$ to go from $\mathbb{R}^d$ to $\mathbb{R}^k$ where $k \geq \frac{log(n)}{\epsilon^2}$.

Theoretical Guarantee of Random Projection Algorithm

- $\forall i, j \in \{1, \dots, n\}$,
  $(1 - \epsilon) * ||V_i - V_j|| \leq ||MV_i - MV_j|| \leq (1 + \epsilon) * ||V_i - V_j||$
- $V = (V_i - V_j) \in \mathbb{R}^d$
  $(1 - \epsilon) * ||V|| \leq ||MV|| \leq (1 + \epsilon) * ||V||$
  $(1 - \epsilon) * ||V||^2 \leq ||MV||^2 \leq (1 + \epsilon) * ||V||^2$
- $||MV||_2^2 = V^\top M^\top MV = V^\top (M^\top M) V$
  1. $E[V^\top M^\top MV] = V^\top E[M^\top M] V$ ("Easy to verify")
     Need to show $E[M^\top M] \approx I$.
     $M = [m_1 \dots m_d]_{k \times d}$ where m_i is a column vector.
     - $(M^\top M)_{d \times d} = \begin{bmatrix} m_1^\top \\ \vdots \\ m_d^\top \end{bmatrix} [m_1 \dots m_d]$

$$= \begin{bmatrix} m_1^\top m_1 & m_1^\top m_2 & \dots & m_1^\top m_d \\ m_2^\top m_1 & \ddots & & \\ \vdots & & & \\ m_d^\top m_1 & \dots & \dots & m_d^\top m_d \end{bmatrix}$$

- $E[m_i^\top m_j]$, where $1 \leq i, j, \leq d$
  $i \neq j \rightarrow E[a^\top b] = E[\sum_{i=1}^{k} a_i b_i] = \sum_{i=1}^{k} E[a_i b_i] = \sum_{i=1}^{k} E[a_i] * E[b_i] = 0$ TODO 1:20 in
  $i = j \rightarrow E[m_i^\top m_j] = k$
- $E[M^\top M] = k * I$
  $\tilde{M} = \frac{M}{\sqrt{k}} \sim \mathcal{N}(0, \frac{1}{k})$
- $v_i \rightarrow \tilde{M} * v_i$
  $E[||\tilde{M}V||^2] = ||V||^2$
- Can define $\tilde{M}$ as sparse $\{+1, -1, 0\}$:
  $$M_{ij} = \begin{cases} +1 : w.p. \frac{1}{6} \\ -1 : w.p. \frac{1}{6} \\ 0 : w.p. \frac{2}{3} \end{cases}$$

Principal Component Analysis (PCA)

Lets consider that we have some data matrix X.
A clean $X = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 4 & 6 & 8 & 10 \end{bmatrix}$. Easy to reduce: change axis to $y = 2x$.

A noisy $X = \begin{bmatrix} 1.001 & 2.001 & 3.01 & 3.999 & 4.999 \\ 2 & 4 & 6 & 8 & 10 \end{bmatrix}$.

Idea:

- We want to find a low-rank M.
- We want $M \approx X$.

Minimize:

- $min_M ||M - X||_F$ s.t. $rank(M) \leq r \leftarrow$ non-convex.

In 1990's, smart guys invented,

Singular Value Decomposition

Treat it as a black-box algorithm:
$X \rightarrow \boxed{SVD} \rightarrow (U, S, V)$.

SVD Guarantees:

1. $X = USV^\top$ where $U \in \mathbb{R}^{d \times d}, S \in \mathbb{R}^{d \times n}, V \in \mathbb{R}^{n \times n}$.
2. $U^\top U = I, V^\top V = I$.
3. S is diagonal.

Back to PCA

- Keep first d columns in U to get $U' \in \mathbb{R}^{d \times r}$,
  first $r \times r$ in S to get $S' \in \mathbb{R}^{r \times r}$, and
  first r rows in $V^\top$ to get $V'^\top \in \mathbb{R}^{r \times n}$.
- Multiply $U'S'V'^\top$ to get $M^* \in \mathbb{R}^{d \times n}$ TODO 1:20

11/8/2022 - Lecture 8

PCA

- Observe $\mathbf{X} \in \mathbb{R}^{d \times n}$. $X = L + N$ where $rank(L) \leq r, r \leq d, r \leq n$.
- We want to the low rank matrix M as $min_M ||M - X||_F$ such that $rank(M) \leq r$.
  - We know the global optimum of this problem is given by SVD.
- From above, use SVD
  - $X \rightarrow \boxed{SVD} \rightarrow (U, S, V)$.
  - $U = (U_1 \dots U_d)$ where each U_i is a column vector.
  - $S = \begin{pmatrix} S_1 & & & | & \\ & \ddots & & | & \mathbf{0} \\ & & S_d & | & \end{pmatrix}$.
  - $V^\top = \begin{pmatrix} V_1^\top \\ \vdots \\ V_r^\top \end{pmatrix}$ where each $V_i^\top$ is a row vector.
  - $X = \sum_{i=1}^{d} S_i U_i V_i^\top$.
    - If $S_1 = 1M$ and the rest of $S_{2 \dots d} \approx 10^{-12}$, then X can be approximated as: $X = 10^6 * U_i V_i^\top + 10^{-12} \sum_{i=2}^{d} U_i V_i^\top$.
    - X can be approximated by a rank 1 matrix!
- If we want to approximate M to r dimensions

Recommendation Systems

Rating Systems

- Real Valued (1 star to 5 star)
- Binary Feedback (liked/disliked)

"Real Valued" Data

- A matrix $Z \in \mathbb{R}^{n \times p}$ can be made from n users and p items with values in each cell for their rating.
- This matrix will be quite sparse as most users will not have a review for most items
- Ω is a list of coordinates of the known values in Z (only holds their position, their value is only stored within Z).

Traditional Collaborative Filtering (CF) for Real Valued Data:

- $min||(Z - X)_\Omega||_F^2$, s.t. $rank(X) \leq r$ where r is the desired rank of the model.

Example:

- $Z = \begin{bmatrix} & 1 & \\ 2 & & \\ & & 3 \end{bmatrix}$, $\Omega = [(1,2),(2,1),(3,3)]$, counting from 1.
- We can fill this matrix in (for now arbitrarily), like:
  $X = \begin{bmatrix} 1000 & 1 & -50 \\ 2 & 69 & 14 \\ 15 & 3.2 & 3 \end{bmatrix}$.
- Our goal is for X to become approximately equivalent to Z to guess how likely user n is to like item p.

Furthermore, $X = UV$ where $U_{n \times r}$, $V_{r \times p}$, which allows us to expand our objective function to CF: $min_{U,V}||(Z - UV^\top)_\Omega||_F^2 + \lambda(||U||_F^2 + ||V||_f^2)$, where λ is a hyperparameter.

Limitations of Real-Valued Systems

- The maximum score is bounded.

Binary Feedback

- On Amazon reviews, other users mostly only interact with the review by clicking "useful" or "not useful."
- Benefits:
  - Eases the process of data acquisition
  - Saves on storage
- Drawbacks:
  - Limited amount of information
  - How can we tell if a user likes one product more than another?
  - Destroys low-rank structure.
    - Lets say the threshold for a good product is 3:
      $X = \begin{bmatrix} 2 & 4 \\ 1 & 2 \end{bmatrix}$ v.s. $Z = \begin{bmatrix} -1 & +1 \\ -1 & -1 \end{bmatrix}$.
    - Rank of matrix in real-valued data X is 1, but rank of binary data Z is 2.
- We want to predict the true (real-valued) preference of the user based on the binary observation.

CF for binary data (Natural Approach):

- $min_X||(Z - sign(X))_\Omega||_F^2$, such that $rank(X) \leq r$.
- The sign of X is based on some threshold, like 3 from above.

Why Natural Approach Fails:

- Sign function absorbs magnitude:
  - $Z = \begin{bmatrix} -1 & 1 \\ -1 & -1 \end{bmatrix}, X = \begin{bmatrix} -1 & 3 \\ -2 & -4 \end{bmatrix}, Z - sign(X) = \begin{bmatrix} -1 & 5 \\ -5 & -1 \end{bmatrix}$

Probabilistic Model

- For $(i,j) \in \Omega$, $Z_{i,j} = \begin{cases} +1, w.p.\ f(X_{i,j}), \\ -1, w.p.\ 1 - f(X_{i,j}). \end{cases}$

- $f(X_{i,j}) = \frac{1}{1+e^{-X_{i,j}}}$ is the logistic function.

Maximum Likelihood Estimator (James-Stein)

We have a label as follows: $y = \begin{cases} +1, w.p. f_w(x) \\ -1, w.p. 1 - f_w(x) \end{cases}$.

- If $y_{1...n} = +1$, then $f_w(x) = 1$ for $x_{1..n}$,
- If its more mixed,
  1. $P(y_1 = +1, y_2 = -1, \dots y_n = +1) = \prod_{i=1}^n P(y_i = \hat{y}_i)$
     $= \prod_{i=1}^n (f_w(x_i) * 1\{\hat{y}_i = +1\} + (1 - f_w(x_i)) * 1\{\hat{y}_i = -1\})$.
     Let $h_w(x_y, \hat{y}_i) = (f_w(x_i) * 1\{\hat{y}_i = +1\} + (1 - f_w(x_i)) * 1\{\hat{y}_i = -1\})$.
     - $1\{\}$ here is the indicator function: $1 := \begin{cases} 1, x \in A, \\ 0, x \notin A. \end{cases}$
  2. $max_w(log(\prod_{i=1}^n h_w(x_i, \hat{y}_i))) = \sum_{i=1}^n log(h_w(x_i, \hat{y}_i))$
     $\iff min_w \sum_{i=1}^n log(\frac{1}{h_w(x_i, \hat{y}_i)})$
     - $min_w \sum_{i=1}^n log(\frac{1}{h_w(x_i, \hat{y}_i)}) = min_w \sum_{i=1}^n log(1 + e^{-X_{i,j}})$.
- Negative Log-likelihood Function:
  - similar to logistic regression
- Maximum Likelihood Estimation:
  - $minL(X; Z, \Omega)$, s.t. $rank(X) \leq r$.

11/15/2022 - Lecture 9

Mean Estimation

Estimating Average Height

- Assume $D = \mathcal{N}(60, 1)$, $E[D] = 60$, $Var[D] = 1$.
- Estimator $\hat{\mu} = \frac{1}{n} \sum_{i=1}^n x_i$.
  - $E[\hat{\mu}] = E[\frac{1}{n} \sum_{i=1}^n x_i] = \frac{1}{n} \sum_{i=1}^n E[x_i] = \mu$.
  - $Var[\hat{\mu}] = Var[\frac{1}{n} \sum_{i=1}^n x_i] = \frac{1}{n^2} \sum_{i=1}^n Var[x_i] = \frac{1}{n}$.

When Data is Noisy

- 1-dimensional: $||\hat{\mu} - \mu||_2^2 \geq \Omega(\epsilon)$ if corrupt ϵ-fraction, where $0 < \epsilon < \frac{1}{2}$ is the portion of the data that an adversary could corrupt.
  - For $D_1 = \mathcal{N}(\mu, 1)$ and $D_2 = \mathcal{N}(\mu_2, 1)$, construct corrupted data Q_1 and Q_2 s.t. $||\mu_1 - \mu_2||_2^2 \geq \Omega(\epsilon)$, and $D_\epsilon = (1 - \epsilon)D_1 + (\epsilon)Q_1 = (1 - \epsilon)D_2 + (\epsilon)Q_2$.
  - Let ϕ_1 be the pdf of D_1 and ϕ_2 be the pdf of D_2. Let μ_1, μ_2 be s.t. the total variance distance between D_1, D_2 is:

$$\frac{1}{2} \int |\phi_1 - \phi_2| dx = \frac{\epsilon}{1 - \epsilon} \implies ||\mu_1 - \mu_2|| \geq \frac{2\epsilon}{1 - \epsilon}.$$

- $Q_1 = \frac{1-\epsilon}{\epsilon}(\phi_2 - \phi_1) * \mathbf{1}_{\phi_2 \geq \phi_1}$ and $Q_2 = \frac{1-\epsilon}{\epsilon}(\phi_1 - \phi_2) * \mathbf{1}_{\phi_1 \geq \phi_2}$.
- $(1-\epsilon)\phi_1 + \epsilon * \frac{1-\epsilon}{\epsilon}(\phi_2 - \phi_1) * \mathbf{1}_{\phi_2 \geq \phi_1}$
  $= (1-\epsilon)\phi_1 + (1-\epsilon)(\phi_2 - \phi_1) * \mathbf{1}_{\phi_2 \geq \phi_1}$
  $= \begin{cases} (1-\epsilon)\phi_2 & \phi_1 \leq \phi_2 \\ (1-\epsilon)\phi_1 & \phi_1 > \phi_2 \end{cases}$
- Similarly for Q_2,
  $(1-\epsilon)\phi_1 + \epsilon * \frac{1-\epsilon}{\epsilon}(\phi_2 - \phi_1) * \mathbf{1}_{\phi_2 \geq \phi_1}$
  $= \begin{cases} (1-\epsilon)\phi_1 & \phi_1 \geq \phi_2 \\ (1-\epsilon)\phi_2 & \phi_1 < \phi_2 \end{cases}$.
- We cannot do better than the precision that is decided by ϵ.

- What about higher dimensions?

Natural Approaches

- Learn each coordinate separately
  - We want $\hat{\mu}$ s.t. $||\hat{\mu}_i - \mu_i|| \geq \Omega(\epsilon)$.
  - In n-dim: $||\hat{\mu} - \mu||_2^2 \geq \sum_{i=1}^n$ TODO 30 min
  - Not a good estimator
- Maximum Likelihood Estimator/Negative Log Likeliood
  - $min(NLL(F, x_1, \ldots, x_m)) = -\sum_{i=1}^n log(F(x_i))$ for sample $S = \{x_i\}_{i=1}^m$.
    - We can assume F is Gaussian. $F(x_i) = \frac{1}{\sqrt{2\pi}} e^{-\frac{||x_i - \mu||_2^2}{2}}$.
  - $min(NLL) = min_\mu(-\sum_{i=1}^n log(\frac{1}{\sqrt{2\pi}} e^{-\frac{||x_i-\mu||_2^2}{2}}))$
    $= min_\mu(-Nlog(\frac{1}{\sqrt{2\pi}}) + \sum_{i=1}^n \frac{||x_i-\mu||_2^2}{2})$
    $\implies min_\mu \frac{1}{2} \sum_{i=1}^m ||x_i - \mu||_2^2$, $\hat{\mu} =$ empirical mean $= \frac{1}{m}\sum_{i=1}^m x_i$.
  - Not computationally efficient

Efficient Algorithm - Convex Programming

Output

- Vector $\hat{w} = (w_1, w_2, \ldots, w_m)$ such that $\hat{\mu} = \frac{1}{m}\sum_{i=1}^m w_i x_i$ is close to μ by solving a convex program.

Efficient Robust Mean Estimation - Filter

1. Compute empirical mean and covariance μ_T, Σ_T where T is the corrupted dataset.
   - $\mu_T = \frac{1}{m}\sum_{i=1}^m x_i$
   - $\Sigma_T = \frac{1}{m}\sum_{i=1}^m (x_i - \mu_T)(x_i - \mu_T)^\top$
2. Compute largest eigenvalue λ^* of $\Sigma_T - I$, and eigenvector v^*.
3. If λ^* is small, return μ_T.
4. Otherwise, find $t > C_1$ such that
   - $P_{X \in T}[|v^* \cdot (X - \mu_T)| > t] > C_2 e^{-\frac{t^2}{2}} + \frac{C_3 \epsilon}{t^2 log(nlog(\frac{n}{\epsilon\tau}))}$ where τ is the probability that the algorithm fails; $0 < \epsilon < \frac{1}{2}$.

5. Remove X such that $|v^* \cdot (X - \mu_T)| > t$, then cycle back to step 1.

List-decodable Mean Estimation

- α is the fraction of clean data samples

Output

- Need to relax our output to instead output a list of mean estimations such that at least one mean should be close to the true data cluster while the others may be of the adversarial fake data clusters. $E[D] \in \{\mu_1, \ldots, \mu_m\}$.
- Our target is that the number of output means will be upper bounded as $m = O(\frac{1}{\alpha})$.

Algorithm: Multi-filtering

- Do a clustering of the dataset into a tree of subsets T_i's, where the number of subsets will be less than $\frac{1}{\alpha}$.
- Iterate through each node
    1. Create a leaf node, an estimate $\hat{\mu}_i$.
    2. Create child nodes, subsets T_i's.
        - One node, cleaner set
        - Two nodes, overlapping subsets
    3. Delete if it can't be α-good.
        - T_i is α-good if α-fraction of T_i are clean.
- No more filtering, then return al $\hat{\mu}_i$'s.

Guarantee

- $min_{i \in \{\mu_1, \ldots \mu_m\}} ||\hat{\mu}_i - \mu||_2 \in O(\frac{1}{\sqrt{\alpha}})$.

11/22/2022 - Lecture 10

Mean Estimation

Goal

- Given a dataset D, we want to estimate the mean $E[D] = \mu$.

Markov's

- $P(X > t * \mu) < \frac{1}{t}$

Chebyshev's

- Given $Var(D)$, $P(|x - \mu| > t) \leq \frac{Var(X)}{t^2}$.
- $X = \frac{1}{n} \sum_{i=1}^{n} x_i$.
- $Var[X] = \frac{1}{n} Var[x] = \frac{\alpha}{n}$.
  $P(|x - \mu| > t) \leq \frac{\frac{\alpha}{n}}{t^2}$.
- w.p. $1 - \frac{\alpha}{n\epsilon^2}$, $|x - \mu| \leq \epsilon$.

- Error Rate $\rightarrow \frac{\epsilon}{d}$ where d is dimensions. Best possible error rate is just ϵ.

Mean Estimation against an Adversary

Situation

- We have the knowledge of the clean data's distribution. Say an adversary can then corrupt 10% of the dataset and place these points wherever they please. They can either (a) follow the normal distribution and thus not be distinguishable, but then the true mean would remain unaffected, or (b) focus their effort by placing all of the points as far from the true mean as possible, but there are algorithms to counter this.

Intuitive Approach

- Lets say we have points $x_1 \dots x_n$, where some are corrupted and some are not.
- We will assign weights $q_1 \dots q_n$ to each of these points, where $0 \leq q_i \leq 1$.
- Goal:
  - If x_i is clean, then $q_i = 1$.
  - If x_i is corrupted, then $q_i = 0$.
- $\implies \frac{1}{(1-\eta)n} \sum_{i=1}^{n} q_i x_i$ where η is the fraction of data that is corrupted

Constraints

1. $0 \leq q_i \leq 1$
2. $\frac{1}{(1-\eta)n} \sum_{i=1}^{n} (x_i - \hat{\mu})(x_i - \hat{\mu})^\top \leq C$ where C is given as long as we know the clean data distribution.
   - $\hat{\mu} = \frac{1}{(1-\eta)n} \sum_{i=1}^{n} q_i x_i$
   - LHS = $Cov(D)$ where $D = \mathcal{N}(\mu, I)$
   - Lets say $q_i = 1$ but x_i is corrupted, then the loss will be large
   - As long as our variance is within a certain range, we'll be able to get correct estimations.
3. $\sum_{i=1}^{n} q_i \geq (1-\eta)n$.
   - The clean x_i should have large q_i. If only 20% is corrupted then the weights should imply at least 80% is clean.
   - Either the corrupt data will behave like the clean data, or they will be removed.

Gaussian data will always have $\eta < \frac{1}{2}$.

- If 99% of reviews are fake, there are several ways to display the reviews
  - 4.9 stars (110 people)
  - Warning: Received 1 one-star review
  - ...
- If the true mean follows a different gaussian curve, while fake data creates different gaussian curves, we can only hope that one mean is close to the true mean.

List-decodable Mean Estimation

Goal:

- $min_{1 \leq i \leq k} ||\hat{\mu}_i - \mu|| \leq \epsilon$.

Example: What is the capital of Pennsylvania?

- 90%: Philadelphia (incorrect)
- 10%: Harrisburg (correct)

To mitigate:

- Pair their answer with a confidence rating
- Pair their answer with what percent of people that'd agree with them

Active Learning -- One of his favorite problems

Setup

- Let X fall within a circle around the origin. w^* is the true direction, and its orthogonal line is the division point between positive and negative labels.
- The grand truth model is $y = sign(w^* * x)$.
  - If $w^* x > 0$, $y = 1$. If $w^* x < 0$, $y = -1$.

Goal

- We want to find points that will make our approximated model w_1 disagree with the grand truth model w^*.
- The regions between the orthogonal of w_0 and w^* where the two disagree are called the **informative regions**.

Strategy

- Draw points from the informative regions and only label them. This can save on labeling costs.
- This should allow $w_1, w_2, w_3, \ldots$ to converge on w^*.

Results

- Number of labels per iteration = d.
- Initial Angle $\in O(n)$
- Number of iterations = $log\frac{1}{\epsilon}$
- Total number of labels = $d * log\frac{1}{\epsilon}$.

Theoretical Bound

- $\Omega(d * log\frac{1}{\epsilon}) \rightarrow$ this algorithm is near optimal!
- Number of labels you need to collect: $d\frac{1}{\epsilon}$.

In worst case, w_0 is pointing completely opposite from w^*.

- This means $||w_0 - w^*|| \leq 2$.

- $\implies V_0 = \{w : ||w - w_0|| < 2\}$ where $w^* \in V_0$ and V_0 is the trust region for w^*.
  - Basically we can guarantee w^* is in the trust region V_0.
- $\implies ||w_1 - w^*|| \leq 1 \implies V_0 = \{w : ||w - w_0|| < 1\} \implies \dots$

Margin-based Active Learning 2007

11/29/22 - Lecture 11

Sparse Linear Regression

12/6/22 - Lecture 12

Binary Classification via Majority Vote

$$\begin{matrix} x_1 & w_1 & \to y_1 \\ \vdots & \vdots & \vdots \quad \to majority\ vote \to \hat{y} \\ x_k & w_k & \to y_k \end{matrix}$$

Goal:

- Q: What is the min k to make sure $\hat{y}$ is correct w.p. η?
- Each worker has $P(y_i = y^*) > 0.5$.

Result:

- $k = \frac{1}{(1-P)^2} log\frac{1}{\delta}$ using Hoeffding's where P is the probability that the worker guesses correctly.
- $\delta = 0.01$, $log\frac{1}{\delta} = 4$

Problem:

- $\begin{matrix} x_1 & w_1 & \to y_1 & = y^* & w.p.\ 0.99 \\ \vdots & \vdots & \vdots \\ x_k & w_k & \to y_k & = y^* & w.p.\ 0.99 \end{matrix}$
- $(0.99)^k$ trends to 0

Claim:

- If the goal is to make sure $S = \{x_1, \dots, x_n\}$ are correctly labeled, then:
  $\hat{k} \to \hat{k}' = \frac{1}{P^2} log(\frac{n}{\delta})$.
- Basically, the number of workers $\hat{k}$ would have to scale with the size of the dataset n.

Union Bound:

Say there are n events $E_1, \ldots, E_n$.

- $\delta_i = P(!E_i) = \frac{\delta}{n}$
- $P(E_1 \cap E_2 \cap \ldots \cap E_n) \geq 1 - \sum_{i=1}^{n} \delta_i \equiv$
  $P(E_1 \cap E_2 \cap \ldots \cap E_n) \geq 1 - \frac{P(!E_1 \cup \ldots \cup !E_n)}{\sum_{i=1}^{n} P(!E_i)}$.

Overhead:

$Overhead = \frac{\#\ labels\ from\ crowd}{\#\ labels\ from\ PAC}$.

Standard PAC:

- Probably Approximately Correct model using k experts
- Each expert outputs a correct label y_i^* for its input x_i
  - These are standard PAC learners.

Crowdsourcing:

- No expert, so we cannot have the lower bound $\sqrt{n}$.
- The number of labels needed is a constant n.

Theorem

- We introduce three models h_1, h_2, h_3.
- Dataset $D = X * Y$.
  - $D_1 = D \rightarrow h_1$
    - Original data distribution (tells us nothing)
  - $D_2 = \frac{1}{2} D_I + \frac{1}{2} D_C \rightarrow h_2$
    - $\begin{cases} D_I : \{x | h_1(x) \neq y^*\} \\ D_C : \{x | h_1(x) = y^*\} \end{cases}$ (D_I means incorrect, D_C means correct)
    - Creates a mixture model (maximum confusion).
      - For $(x, y^*) \in D_2$, $P(h_1(x) = y^*) = \frac{1}{2}$.
  - $D_3 = \{x | h_1(x) \neq h_2(x)\} \rightarrow h_3$
    - D_3 is the disagreement region between D_1 and D_2.

Standard PAC Learners

- $(x_1, y_1^*), \ldots, (x_n, y_n^*) \rightarrow \blacksquare \rightarrow h : err(h) \leq \epsilon$ w.p. $1 - \delta$
  - $err(h) = P(h(x) \neq y^*)$ as long as
    - $n_\epsilon = \frac{1}{\epsilon}(d + log\frac{1}{\delta})$
  - If num labels = $\frac{3}{\delta}$ ... $n_{\sqrt{\epsilon}} = (\frac{1}{\sqrt{\epsilon}}(d + \log\frac{1}{\delta}))$
- Applying the black box to each dataset yields h_1, h_2, and h_3.
- With h_1, h_2, h_3, $\forall x$ output, $y = f(x) = majority(h_1(x),\ h_2(x),\ h_3(x))$.
- $err_D(f) \leq \epsilon^2 \leftarrow$ strong learner

- $\begin{cases} err_{D_1}(h_1) \leq \epsilon \\ err_{D_2}(h_2) \leq \epsilon \leftarrow \text{weak learners} \\ err_{D_3}(h_3) \leq \epsilon \end{cases}$

Crowdsourcing Overhead

Overhead = $\frac{\#\ labels\ from\ crowd}{\frac{1}{\epsilon}}$

$= \frac{\frac{3}{\sqrt{\epsilon}}}{\frac{1}{\epsilon}} \rightarrow 0$

$= 3\sqrt{\epsilon}\log\frac{1}{\epsilon} \rightarrow_{\epsilon \rightarrow 0} 0.$

This means there exists a crowd sourcing algorithm that can correctly label all labels.

Crowdsourcing Algorithm

Phase 1: Get h_1.

- $err_{D_1}(h_1) \leq \sqrt{\epsilon}$
- $x_1 \ldots x_{n_{\sqrt{\epsilon}}}$ where $\hat{k} = log(n_{\sqrt{\epsilon}}) = \frac{1}{\sqrt{\epsilon}} log(\frac{1}{\epsilon})$

Phase 2: Get h_2 given h_1.

- $D_2 = \frac{1}{2}D_I + \frac{1}{2}D_C$.
  - $D_I = h_1(x) \neq y^*$.
  - $D_C = h_1(x) = y^*$.
- Given an $x \sim D_1$, get $h_1(x)$.
  - We need to devise an algorithm to place x into either S_I or S_C.
  - Key Idea: Lets say $\sqrt{\epsilon} = 0.1$. For 100 inputs, 90% of them should be correct.
    - Confidence of correctness = 90%.
  - If $w_1 = h_1$, we can guess h_1 is correct. To increase our confidence, increase the number of workers w compared against h_1 and take the majority vote.
    - By guessing h_1 is correct, we can put x into S_C.
  - In order to put x into S_I, we need more power.
    - Let the number of iterations be $t = 1, \ldots, T = \log(\frac{1}{\epsilon})$ where t is odd.
      - T is not very large.
    - Hire worker w_t and take the majority vote $maj(w_1, w_2, \ldots, w_t)$.
      - If all of them ($\frac{T}{2}$) disagree with $h_1(x)$ ($w_i \neq h_1(x)$) then we can guess h_1 is incorrect.
      - If at least one agrees with $h_1(x)$, then we guess that h_1 is correct.

Phase 3: Get h_3.

- $err_{D_1}(h_1) \leq \sqrt{\epsilon}$
- $x_1 \ldots x_{n_{\sqrt{\epsilon}}}$ where $\hat{k} = \frac{1}{\sqrt{\epsilon}} log(\frac{1}{\epsilon})$.

Efficient PAC Learning from the Crowd - Aswathi et. al. 2017

Situation:

- Lets say 80% of the workers are adversarial incorrect workers,
  and 20% of the workers are grand truth correct workers
- We hire 100 workers. We observe 90 "+"s and 10 "-"s.
  - We know that "+" is correct because at most 80 adversarial workers could incorrectly label "+", which means at least 1 correct worker must have labeled it (which means all 20 have correctly labeled it +).
- If we observe 70 "+"s and 30 "-"s, the grand truth is unknown.
  - The 20 correct workers could have labeled + or -, and the incorrect workers could have labeled the rest.
  - So, we must query it to get the correct label, and then we can remove all workers on the wrong label since we know they're adversarial. If the grand truth turns out to be +, then we can remove the 30 workers that labeled it -.

Final Exam Review

5 questions

1 tests knowledge on recommendations FOR SURE

- more general, once you observe some data matrix and missing values, you can apply same approach to solve the problem given linear dependence on rows or columns (theres some group structure/similarities)
- whats the meaning of each row, or of each column

1 tests knowledge on noise/robustness

- mean estimation, linear regression, svm

1 general question

- which problem you like most, talk about understanding of that problem
- whats your understanding of active learning

1 dimensional reduction problem

- PCA - singular value decomposition, why can we do this?
- random projection
  - what is the tradeoff between RP and PCA? when is one better and why? in what aspect, how can we pick between these two approaches?

1 active learning

- higher level, classical examples of how active learning helps to reduce labeling cost
- one example: look at classification on real line, do binary search. reduces cost from n to log n
- one example where active learning does NOT work

"take it easy", "not so technical"