Foundation of Data Science - summary

Chapter1

- Cauchy-Schwarz Inequality: $|\mathbf{x} \cdot \mathbf{y}| \leq \|\mathbf{x}\| \|\mathbf{y}\|$
- An affine hyperplane in R^l is an affine subspace of dimension l-1
- A hyperplane P={**x**|**a**•**x**=b} is homogeneous if b=0,or equivalently, if 0∈P, that is, the hyperplane goes through the origin(b 是标量，这也是为啥 x 是 l-1 的超平面)

Algorithm PERCEPTRON

Input: Normalised training sequence S.

Objective: Compute weight vector $\mathbf{w}$ such that the hypothesis $\mathbf{x} \mapsto \text{sgn}(\mathbf{w} \cdot \mathbf{x})$ is consistent with S.

1. $\mathbf{w} \leftarrow \mathbf{0}$
2. repeat
3.   for all $(\mathbf{x}, y) \in S$ do
4.   if $\text{sgn}(\mathbf{w} \cdot \mathbf{x}) \neq y$ then
5. $\mathbf{w} \leftarrow \mathbf{w} + y\mathbf{x}$
6. until $\text{sgn}(\mathbf{w} \cdot \mathbf{x}) = y$ for all $(\mathbf{x}, y) \in S$

- Normalize 是根据数据里最长的那个，都除以它
- 算法中 w 加一位 bias，初始化都是 0
- Let S be a normalized sequence of examples such that there is a homogeneous linear separator consistent with S of margin γ. Then the perceptron algorithm applied to S finds a linear separator after at most $1/\gamma^2$ updates of **w**.

- Perceptron always finds separators if exists, but not optimal. SVM finds the one with maximum margin.
- 构造决策树

Input: Set $\mathcal{A}$ of features, set S of examples
Objective: Compute decision tree t.

1. if $S = \emptyset$ then
2. create leaf t with arbitrary value
   ▷ e.g., majority value for parent node
3. else if all examples in S have the same y-value then
4. create leaf t with that y-value
5. else
6. choose feature $A \in \mathcal{A}$ that discriminates best between examples in S
7. create new node t with feature A
8. partition examples in S according to their A-value into parts $S_1, \ldots, S_m$
9. recursively call algorithm on $\mathcal{A} \setminus \{A\}$ and the S_i and attach resulting trees t_i as children to t
10. return t

k-CNF: Boolean formula in conjunctive normal form with clauses (= disjunctions of literals) of at most k literals(k 约束的是小括号中的项)

k-DNF: similar

conjunction: 且

disjunction: 或

CNF->DNF: straightforward

DNF->CNF:A∨B=>¬(¬ A ∧¬ B)

- Computing a smallest decision tree for a given set of examples is NP-complete

Chapter2

- $$\mathrm{Var}(X) = \mathrm{E}\left[(X-\mu)^2\right].$$
- For any class H, VCdim(H) <=$\log_2$(|H|)
- If the concept class H has VC-dimension d, then for any combination function f, the class $\mathrm{COMB}_{f;k}$(H) has VC-dimension O(kd log(kd))
- PAC: a learning algorithm is probably approximately correct with respect to a probability distribution D on the instance space and a target function C*, if given ε,δ> 0, it draws a training sequence S from D and produces a hypothesis $H_{S,\varepsilon,\delta}$ such that:

$$\Pr_{S\sim\mathcal{D}}\left(\mathrm{err}_{\mathcal{D},C^*}(H_{S,\varepsilon,\delta}) < \varepsilon\right) \geq 1-\delta$$

- If not identically distributed and not independent, the following concentration inequalities cannot be applied
- Markov's Inequality

Let X be a ***nonnegative*** random variable, then for all a>0,

$$\Pr(X \geq a) \leq \frac{\mathrm{E}(X)}{a}$$

- Chebyshev's Inequality

Let X be a random variable. Then for all b > 0,

$$\Pr\left(|X - \mathrm{E}(X)| \geq b\right) \leq \frac{\mathrm{Var}(X)}{b^2}$$

- Let $X_1, \ldots, X_n$ be a ***pairwise*** independent sequence of random variables and $X := \sum_{1\ldots n} X_i$. Then,

$$\mathrm{Var}(X) = \sum_{i=1}^{n} \mathrm{Var}(X_i)$$

- Law of large number

Let X1,...,Xn be a pairwise independent sequence of random variables of variance $\mathrm{Var}(X_i) \leq \sigma^2$, and let $X := \sum_{1\ldots n} X_i$. Then for all c > 0(可以由 Chebyshev 和上面的引理证明),

$$\Pr\left(|X - \mathrm{E}(X)| \geq cn\right) \leq \frac{\sigma^2}{c^2 n}$$

weak law(其实没区别，只是左侧除以 n，注意 mu=E(Xi)):

$$\lim_{n\to\infty} \Pr\left(\left|\frac{\sum_{i=1}^{n} X_i}{n} - \mu\right| \geq \varepsilon\right) = 0$$

- Chernoff Bounds (Multiplicative version)

Let X1,..., Xn be a sequence of independent {0,1}-valued random variables. Let $X := \sum_{1\ldots n} X_i$ and $\mu := \mathrm{E}(X)$. Then for $0 \leq c \leq 1$:

$$\Pr(X \geq (1+c)\mu) \leq e^{-\frac{\mu c^2}{3}} \quad \Pr(X \leq (1-c)\mu) \leq e^{-\frac{\mu c^2}{2}}$$

Corollary(两式合一，上式更松，故都松):

$$\Pr\left(|X - \mu| \geq c\mu\right) \leq 2e^{-\frac{\mu c^2}{3}}$$

- Hoeffding Bounds

Let X1,..., Xn be a sequence of independent identically distributed {0,1}-valued random variables. Let $X := \sum_{1\ldots n} X_i$ and $\mu := \mathrm{E}(X)$. Then for $0 \leq d \leq 1$:

$$\Pr(X \geq \mu + dn) \leq e^{-2nd^2} \quad \Pr(X \leq \mu - dn) \leq e^{-2nd^2}$$

Corollary:

$$\Pr\left(|X - \mu| \geq dn\right) \leq 2e^{-2nd^2}$$

- Uniform convergence

Let H be a finite hypothesis class, $\varepsilon, \delta > 0$, and

$$m \geq \frac{1}{2\varepsilon^2}\left(\ln|\mathcal{H}| + \ln\left(\frac{2}{\delta}\right)\right)$$

Then for all probability distributions D on X and all target functions C*,

$$\Pr_{\substack{S\sim\mathcal{D}\\ |S|=m}}\left(\forall H \in \mathcal{H} : |\mathrm{err}_S(H) - \mathrm{err}_{\mathcal{D},C^*}(H)| < \varepsilon\right) \geq 1 - \delta.$$

(proved by Hoeffding bounds)

- Simple consistency algorithm

For a class $\mathcal{H}$ of hypotheses, we let $\text{CONS}_{\mathcal{H}}$ be the simple learning algorithm that, given $\varepsilon, \delta > 0$,

- draws a training sequence S of length $m := \lceil \frac{1}{\varepsilon}(\ln|\mathcal{H}| + \ln(1/\delta)) \rceil$
- returns a hypothesis $H \in \mathcal{H}$ consistent with S or $\perp$, if there is no such hypothesis.

Corollary:

For every probability distribution D and every target function C*∈H, the algorithm $Cons_H$ is probably approximately correct with respect to D and C*

- Almost consistency algorithm (this use uniform convergence)

For a class $\mathcal{H}$ of hypotheses, we let $\text{ACONS}_{\mathcal{H}}$ be the learning algorithm that, given $\varepsilon, \delta > 0$,

- draws a training sequence S of length $m := \lceil \frac{2}{\varepsilon^2}(\ln|\mathcal{H}| + \ln(2/\delta)) \rceil$,
- returns a hypothesis $H \in \mathcal{H}$ with $\text{err}_S(H) \leq \varepsilon/2$ or $\perp$, if there is no such hypothesis.

Corollary:

The same, also PAC

- Simple consistency algorithm for VC Dimension

Let $\mathcal{H}$ be a hypothesis class such that $d := \text{VC}(\mathcal{H}) < \infty$. We let $\text{VCCONS}_{\mathcal{H}}$ be the simple learning algorithm that, given $\varepsilon, \delta > 0$,

- draws a training sequence S of length $m := \lceil \frac{c}{\varepsilon}\left(d \log\left(\frac{1}{\varepsilon}\right) + \log\left(\frac{1}{\delta}\right)\right) \rceil$ for the constant c of the theorem,
- returns a hypothesis $H \in \mathcal{H}$ consistent with S or $\perp$, if there is no such hypothesis.

Corollary:

The same, also PAC

- Occam's razor

Fix any description language, and consider a training sample S drawn from distribution D. With probability at least 1-δ, any rule h consistent with S that can be described in this language using fewer than(<) b bits will have errD(h) <=ε or |S| = 1/ε [b ln(2) + ln(1/δ)]. Equivalently, with probability at least 1-δ , all rules that can be described in fewer than b bits will have errD(h) <= b ln(2)+ln(1/δ) / |S|.

- Occam's razor application: decision tree

A tree described using O(k $\log_2$ d)bits, b is number of features, log base is 2, k is number of nodes

|S| >= 1/ε [ck $\log_2$ d + ln(1/δ)] for constant c

- Shatter: Given a set S of examples and a concept class H, we say that S is shattered by H if for every A⊆S there exists some h ∈ H that labels all examples in A as positive and all examples in S \ A as negative

- VC-dimension VC(H) of H is the size of the largest set shattered by H, or ∞ if arbitrarily large sets are shattered by H(看的是存在性，而不是全部都)

- Common VC dimensions:

1. $X=R^2$, H the calss of all axis-parallel rectangles. VC=4
2. $X=R^l$, H the class of all halfspaces in X. VC=l+1
3. X=R, H the class of all finite subsets of X. VC=∞
4. X=Σ* for alphabet Σ of size>=2, H :={Lw|w∈Σ*}, 就是所有前缀，VC=2

(prove by >=, then <=)

- Union of VCs

$$\mathrm{VCdim}(H \cup H') \leq \mathrm{VCdim}(H) + \mathrm{VCdim}(H') + 1$$

- Sample size bound

Let H be a finite hypothesis class. Let ε,δ> 0 and

$$m \geq \frac{1}{\varepsilon}\left(\ln|\mathcal{H}| + \ln\left(\frac{1}{\delta}\right)\right)$$

Then for all probability distributions D on X and all target functions C*,

$$\Pr_{\substack{S\sim\mathcal{D}\\|S|=m}}\left(\forall H \in \mathcal{H} : \left(\mathrm{err}_S(H) = 0 \implies \mathrm{err}_{\mathcal{D},C^*}(H) < \varepsilon\right)\right) \geq 1-\delta$$

- VC-dimension sample size bound

There is a constant c such that following holds. Let H be a hypothesis class of VC dimension d < ∞.

Let ε,δ> 0 and

$$m \geq \frac{c}{\varepsilon}\left(d\log\left(\frac{1}{\varepsilon}\right) + \log\left(\frac{2}{\delta}\right)\right)$$

Then for all probability distributions D on X and all target functions C*,

$$\Pr_{\substack{S\sim\mathcal{D}\\|S|=m}}\left(\forall H \in \mathcal{H} : \left(\mathrm{err}_S(H) = 0 \implies \mathrm{err}_{\mathcal{D},C^*}(H) < \varepsilon\right) > 1-\delta\right.$$

- Growth function sample size bound

Let H be a hypothesis class. Then for all 0<ε,δ<=1 and all

$$m \geq \frac{2}{\varepsilon}\left(\log(g_{\mathcal{H}}(2m)) + \log\left(\frac{2}{\delta}\right)\right)$$

Then for all probability distributions D on X and all target functions C*,

$$\Pr_{\substack{S\sim\mathcal{D}\\|S|=m}}\left(\forall H \in \mathcal{H} : \left(\mathrm{err}_S(H) = 0 \implies \mathrm{err}_{\mathcal{D},C^*}(H) < \varepsilon\right) > 1-\delta\right.$$

- Growth function of H is the function g_H : N -> N defined by

$$g_{\mathcal{H}}(n) := \max\left\{|\mathcal{H}[Y]| \;\middle|\; Y \subseteq \mathbb{X} \text{ with } |Y| = n\right\}$$

- $g_H(n)=2^n$ ⇔ n<=VC(H)
- Sauer-Shelah Lemma

Let d := VC(H). Then for all n ∈ N,

$$g_{\mathcal{H}}(n) \leq \sum_{i=0}^{d}\binom{n}{i} \leq \left(\frac{en}{d}\right)^d$$

(This function can be used when n<=d)

- The concept class *C* is PAC-learnable with respect to the hypothesis class *H* if there is a learning algorithm that,

1. given ε,δ, draws a sequence S of at most m(ε,δ) training examples;
2. outputs a hypothesis H ∈ *H*;
3. is probably approximately correct with respect to every distribution D on X and every target function C* ∈ *C*.

If *H* = *C*, we just say that *C* is PAC-learnable.

- A concept class *C* is PAC-learnable if and only if VC(*C*) <∞

Chapter3

- 证明时可以使用：ΣXi <= f => Xi <= f (也就是个体小于等于整体)
- $1+x<=e^x$,涉及到 ln 就想想这个
- 就我的理解，所有本章算法都是 multiplicative weights update 算法。
- Weighted majority algorithm

For some constant $0 < \alpha \leq 1/2$.

- $w_i^{(1)} := 1$ for all $i \in [n]$.

  Intuition: Initially, we give the same weight to each expert's advice.

- For $t \geq 1$, $d^{(t)} := \begin{cases} 1 & \text{if } \sum_{\substack{i\in[n] \\ a_i^{(t)}=1}} w_i^{(t)} \geq \sum_{\substack{i\in[n] \\ a_i^{(t)}=0}} w_i^{(t)}, \\ 0 & \text{otherwise.} \end{cases}$

  Intuition: Buy, if the weighted majority of the experts recommends it.

- For $t \geq 1$ and $i \in [n]$, $w_i^{(t+1)} := \begin{cases} w_i^{(t)} & \text{if } a_i^{(t)} = p^{(t)}, \\ (1-\alpha)w_i^{(t)} & \text{otherwise} \end{cases}$

  Intuition: Decrease weights of experts with wrong prediction by a factor $(1-\alpha)$.

- Analysis of above algorithm

For every $t \geq 1$ and every $i \in [n]$,

$$\ell^{(t)} \leq \frac{2\ln n}{\alpha} + 2(1+\alpha)\ell_i^{(t)}$$

Intuition: The inequality holds, in particular, for i being the best expert. Thus in the long run, our algorithm guarantees our losses to be at most a bit more than twice the losses of the best expert.

- Multiplicative weight update algorithm

For some constant $0 < \alpha < 1$.

- $w_i^{(1)} := 1$ for all $i \in I$.
- For $t \geq 1$ and $i \in I$,

$$w_i^{(t+1)} := (1-\alpha)^{L_{ij^{(t)}}} w_i^{(t)}$$

每回合按照如下式子随机选取 expert

$$\Pr_{\mathcal{D}^{(t)}}(\{i\}) := p_i^{(t)} := \frac{w_i^{(t)}}{\sum_{i'\in I} w_{i'}^{(t)}}$$

(这之所以是之前的扩展，因为 Lij 是[0,1]，可以变动的)

- Analysis for above

For every $t \geq 1$ and every $i \in I$,

$$\sum_{s=1}^{t} L^{(s)} \leq \frac{\ln n}{\alpha} + (1|+\alpha)\sum_{s\leq t} L_{ij^{(s)}}$$

可以看出 the randomized strategy of the multiplicative weight update algorithm beats the deterministic strategy of the weighted majority algorithm by almost a factor 2

- Weak learning

Let $0 \le \gamma < 1/2$. A learning algorithm is weak learning algorithm with error parameter γ (short: γ-weak learner) for $\mathcal{C}$ if, given $\delta > 0$, the algorithm draws a sequence S of $m = m(\delta)$ examples and computes a hypothesis H such that for all probability distributions $\mathcal{D}$ on $\mathbb{X}$ and all $C^* \in \mathcal{C}$,

$$\Pr_{S \sim \mathcal{D}} \left(\mathrm{err}_{\mathcal{D},C^*}(H) < \gamma \right) \ge 1 - \delta.$$

- Strong learning (PAC-learning)

A learning algorithm is a PAC-learning algorithm or strong learning algorithm (short: strong learner) for $\mathcal{C}$ if, given $\varepsilon, \delta > 0$, the algorithm draws a sequence S of $m = m(\varepsilon, \delta)$ examples and computes a hypothesis H such that for all probability distributions $\mathcal{D}$ on $\mathbb{X}$ and all $C^* \in \mathcal{C}$,

$$\Pr_{S \sim \mathcal{D}} \left(\mathrm{err}_{\mathcal{D},C^*}(H) < \varepsilon \right) \ge 1 - \delta.$$

- AdaBoost (means adaptive updates of the distribution, put less weight on those correct)

1. Idea
   a) Repeatedly run the weak learner on subsets of the initial training set.
   b) These subsets are randomly drawn from different probability distributions.
   c) The distributions are adapted in each round using multiplicative weight updates.
2. Weak learner W
   a) We only run W with examples drawn according to probability distributions D on X
   b) Call a hypothesis good if it has true error less than γ. W generates a good hypothesis with probability at least $1-\delta$
   c) As we know D and the correct labels for samples from X, we can check if a hypothesis is good.
   d) We run W on D until it returns a good hypothesis, if bad, re-run
3. MWU algorithm

$$L_{ij} := \begin{cases} 1 & \text{if } j(x_i) = c_i \\ 0 & \text{otherwise.} \end{cases}$$

   Entries for loss matrix are posistive

   Update parameter $\alpha=1/2-\gamma$ (注意，这就和 weak learner W 的真值误差产生了联系, W 越差，学的步伐越小)
4. Boosting algorithm
   - We consider a run of the MWU algorithm where $j^{(s)}$ is a hypothesis obtained by running W on $\mathcal{D}^{(s)}$ until it returns a good hypothesis.
   - We run the algorithm for $t = \frac{2}{\alpha^2} \ln \frac{1}{\varepsilon}$ rounds.
   - The final hypothesis H that we return is defined by

$$H(x) := \begin{cases} 1 & \text{if } \left|\{s \le t \mid j^{(s)}(x) = 1\}\right| \ge t/2, \\ 0 & \text{otherwise.} \end{cases}$$

Thus $\mathrm{err}_s(H) < \varepsilon$

This is for S, so setting ε can force H to be correct on all examples

H may from a class H* which is still simple

可以看出，训练多少轮和 W 的真值误差，以及想要达到的真值误差都有关系

If we want H to classify all examples correctly, we need to take ε≈1/n, and thus run the MWU algorithm for O(log n) rounds

This majority function has VC dimension bound by O(log n * VC(H)), so with large n, we can show hypothesis from H* has small error

- Bandit learning

Observe only payoff of machine we pick, not the others; setting is adversarial, the adversary knows our strategy

Maximal single-action reward:

$$q_{\max}^{(t)} := \max_{a \in [n]} \sum_{s=1}^{t} q_a^{(s)}$$

Reward:

$$q(\mathbf{a}) := \sum_{s=1}^{t} q_{a^{(s)}}^{(s)}$$

Weak regret:

$$r(\mathbf{a}) := q_{\max}^{(t)} - q(\mathbf{a})$$

Algorithm EXP3

Parameter: γ, where $0 < \gamma \leq 1$.

Initialisation: $w_a^{(1)} := 1$ for all $a \in [n]$

1. for $s = 1, 2, \ldots, t$ do
2. $\mathcal{D}^s$ probability distribution defined by

$$\Pr_{\mathcal{D}^{(s)}}(\{a\}) := p_a^{(s)} := (1-\gamma)\frac{w_a^{(s)}}{\sum_{a'=1}^{n} w_{a'}^{(s)}} + \frac{\gamma}{n}$$

3. action $a^{(s)}$ drawn randomly from $\mathcal{D}^{(s)}$
4. reward $q^{(s)} \leftarrow q_{a^{(s)}}^{(s)}$
5. weights are updated as follows:

$$w_a^{(s+1)} \leftarrow \begin{cases} w_a^{(s)} \cdot \exp\left(\frac{\gamma q^{(s)}}{n p_a^{(s)}}\right) & \text{if } a = a^{(s)}, \\ w_a^{(s)} & \text{otherwise} \end{cases}$$

Exp3 stands for exponential-weight algorithm for exploration and exploitation

Parameter γ determines the tradeoff between exploration and expectation: the closer γ gets to 1, the more weight we put on exploration

Expect regret for Exp3:

$$r(Exp3) \leq (e-1) \cdot \gamma \cdot q_{\max}^{(t)} + \frac{1}{\gamma} \cdot n \cdot \ln n$$

Strong regret: for best possible sequence of actions

Chapter4

- the random projection theorem the probability of the length of the projection of a single vector differing significantly from its expected value is exponentially small in k

SVD 就是确定一个最长投影 v，再确定一个最长 v，一直到满秩了。不过实际计算时，是矩阵的特征值求解。

- Universal: A family H of hash functions from U to T is universal if for all distinct x, x'∈ U,

$$\Pr_{h\in\mathcal{H}}\left(h(x)=h(x')\right)\leq\frac{1}{|\mathbb{T}|}$$

- k-universal: if for all distinct x1,...xk ∈U,

$$\Pr_{h\in\mathcal{H}}\left(h(x_1)=h(x_2)=\ldots=h(x_k)\right)\leq\frac{1}{|\mathbb{T}|^{k-1}}$$

- Strongly k-universal if for all distinct x1,...,xk ∈U and all y1,...yk ∈T,

$$\Pr_{h\in\mathcal{H}}\left(h(x_1)=y_1\wedge\ldots\wedge h(x_k)=y_k\right)=\frac{1}{|\mathbb{T}|^{k}}$$

- 高维球 concentration near equator

Let $\ell\geq 3$ *and* $c\geq 1$. *Then*

$$\operatorname{vol}\left(\left\{\mathbf{x}\in B^\ell \;\middle|\; |x_1|>\frac{c}{\sqrt{\ell-1}}\right\}\right)\leq\frac{2}{c}e^{-c^2/2}$$

- Spherical Gaussian distribution(各向 variance 都是 theta^2)

$$p(\mathbf{x})=\frac{1}{(2\pi)^{\ell/2}\sigma^\ell}\exp\left(-\frac{\|\mathbf{x}-\boldsymbol{\mu}\|^2}{2\sigma^2}\right)$$

- Gaussian Annulus Theorem

Let $b\leq\sqrt{\ell}$, *and let* $\mathbf{x}\in\mathbb{R}^\ell$ *be drawn from an* ℓ*-dimensional spherical Gaussian distribution with mean* $\mu=0$ *and variance* $\sigma^2=1$. *Then*

$$\Pr\left(\sqrt{\ell}-b<\|\mathbf{x}\|<\sqrt{\ell}+b\right)\geq 1-3e^{-cb^2},$$

for a constant $c>0$ *not depending on* ℓ *and* b.

- Reduction mapping

In the following, we let $k,\ell\in\mathbb{R}$, where $k\leq\ell$.
We draw vectors $\mathbf{u}_1,\ldots,\mathbf{u}_k\in\mathbb{R}^\ell$ independently from the ℓ-dimensional spherical Gaussian distribution with mean $\mathbf{0}$ and variance σ^2 in each direction and let

$$U:=\frac{1}{\sqrt{k}}\begin{pmatrix}\mathbf{u}_1\\ \vdots\\ \mathbf{u}_k\end{pmatrix}\in\mathbb{R}^{k\times\ell}.$$

- Random projection theorem(就是上面降维 mapping 不错，和原来接近)

For all $\mathbf{x} \in \mathbb{R}^{\ell}$ *and all* $\varepsilon > 0$,

$$\Pr\Big(\big|\|U\mathbf{x}\| - \|\mathbf{x}\|\big| > \varepsilon\|\mathbf{x}\|\Big) \leq 3e^{-c\varepsilon^2 k},$$

where the probability is over the choice of the vectors $\mathbf{u}_1, \ldots, \mathbf{u}_k$ *used to construct the matrix* U *and* c *is the constant from the Gaussian Annulus Theorem.*

- Johnson-Lindenstrauss Lemma（保持相对距离）

Let $0 < \varepsilon < 1$ *and* $k, \ell, n \in \mathbb{N}$ *such that* $k \geq \frac{3}{c\varepsilon^2} \ln n$, *where* c *is the constant from the Gaussian Annulus Theorem.*
Then for every set $X \subseteq \mathbb{R}^{\ell}$ *of size* $|X| = n$,

$$\Pr\Big(\forall \mathbf{x}, \mathbf{y} \in X : (1-\varepsilon)\|\mathbf{x}-\mathbf{y}\| \leq \|U\mathbf{x} - U\mathbf{y}\| \leq (1+\varepsilon)\|\mathbf{x}-\mathbf{y}\|\Big) \geq 1 - \frac{3}{2n}$$

- Principle component analysis, we extract features that are combinations of original features and try to minimize the squared Euclidean distance between original data and their projections
- SVD，怎么一个一个找

The first singular value of A is

$$\sigma_1(A) := \max\big\{\|A\mathbf{v}\| \;\big|\; \mathbf{v} \in \mathbb{R}^{\ell} \text{ width } \|\mathbf{v}\| = 1\big\}.$$

A first right-singular vector of A is a vector $\mathbf{v} \in \mathbb{R}^{\ell}$ with $\|\mathbf{v}\| = 1$ and $\|A\mathbf{v}\| = \sigma_1(A)$.

之后类似，不过都得和之前的垂直

- Singular value 个数等于秩
- SVD，写成向量形式

Let $A \in \mathbb{R}^{n\times\ell}$ *with singular values* $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r$, *corresponding right-singular vectors* $\mathbf{v}_1, \ldots, \mathbf{v}_r$ *and left-singular vectors* $\mathbf{u}_1, \ldots, \mathbf{u}_r$.
Then

$$A = \sum_{j=1}^{r} \sigma_j \mathbf{u}_j \mathbf{v}_j^T$$

- Relations between singular values and eigenvalues

If A is symmetric, then its singular values are the absolute values of the nonzero eigenvalues.

- Best-fit subspaces(当初贪心算法，所以前 k 都是最好的 k 个)

V_k *is a best-fit* k*-dimensional subspace for* A, *that is, a* k*-dimensional subspace* $V \subseteq \mathbb{R}^{\ell}$ *that minimises the sum of the squared distances of the vectors* $\mathbf{a}_i$ *to* V.

- Rank-k approximation of a matrix

$A \in \mathbb{R}^{n\times\ell}$ with singular values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r$, corresponding right-singular vectors $\mathbf{v}_1, \ldots, \mathbf{v}_r$ and left-singular vectors $\mathbf{u}_1, \ldots, \mathbf{u}_r$.

For $1 \leq k \leq r$, let

$$A_k := \sum_{j=1}^{k} \sigma_j \mathbf{u}_j \mathbf{v}_j^T.$$

- Frobenius norm

$$\|A\|_F = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{\ell} a_{ij}^2}.$$

- 刚才的 norm 的性质

Let $A \in \mathbb{R}^{n \times \ell}$ with singular values $\sigma_1 \geq \ldots \geq \sigma_r > 0$. Then

$$\|A\|_F = \sqrt{\sum_{i=1}^{r} \sigma_i^2}.$$

- 2-norm(就是第一 singular value)

$$\|A\|_2 = \sup_{\substack{\mathrm{v} \in \mathbb{R}^\ell \\ \|\mathrm{v}\|=1}} \|A\mathbf{v}\|.$$

Chapter5

- Variance= $E[x^2]-(E[x])^2$

$$= \frac{1}{N} \sum_{u \in \mathbb{U}} (f_u - \mathrm{E}(f_u))^2$$

- Simple sampling algorithm

```
Algorithm SimpleSample
Input:   Stream a_1, ..., a_n                         ▷ Assume n ≥ 1
 1. i ← 0
 2. while not end of stream do
 3.     i ← i + 1
 4.     sample ← a_i with probability 1/i
                          ▷ otherwise sample keeps its current value
 5. return sample
```

- Reservoir sampling

Algorithm ReservoirSample

```
Input: Stream a_1, ..., a_n, k ≤ n
for i = 1, ..., k do
    sample[i] ← a_i
                                        ▷ variable i has value k now
while not end of stream do
    i ← i + 1
    replace ← { true   with probability k/i,
              { false  otherwise
    if replace then
        choose j uniformly at random from [k]
        sample[j] ← a_i
return sample
```

- Universal hashing

A family $\mathcal{H}$ of hash functions from $\mathbb{U}$ to $\mathbb{T}$ is universal if for all distinct $x, x' \in \mathbb{U}$,

$$\Pr_{h\in\mathcal{H}}\left(h(x) = h(x')\right) \le \frac{1}{|\mathbb{T}|}.$$

- 构造 universal hashing

Let $M \le N$, and let $p \ge N$ be a prime. Suppose that $\mathbb{U} = \{0, \ldots, N-1\}$ and $\mathbb{T} = \{0, \ldots, M-1\}$. For $a, b \in \mathbb{N}$, define $h_{a,b} : \mathbb{U} \to \mathbb{T}$ by

$$h_{a,b}(x) := ((ax + b) \bmod p) \bmod M$$

Then the family $\mathcal{H} := \{h_{a,b} \mid a, b \in \{0, \ldots, p-1\}, a \ne 0\}$ is universal

- Collision

Let $\mathcal{H}$ be a universal family of hash functions from $\mathbb{U}$ to $\{0, \ldots, 2^k - 1\}$ Then for every $\delta > 0$ and every set $S \subseteq \mathbb{U}$ of cardinality $|S| \le n$,

$$\mathrm{E}_{h\in\mathcal{H}}\left(\mathrm{coll}(h, S)\right) = \frac{n(n-1)}{2^{k+1}}$$

and $\Pr_{h\in\mathcal{H}}\left(\mathrm{coll}(h, S) \ge \frac{n^2}{\delta 2^{k+1}}\right) \le \delta.$

- k-universal

Let $k \ge 2$, and let $\mathcal{H}$ be a family of hash functions from $\mathbb{U}$ to $\mathbb{T}$.

(1) $\mathcal{H}$ is k-universal if for all distinct $x_1, \ldots, x_k \in \mathbb{U}$,

$$\Pr_{h\in\mathcal{H}}\left(h(x_1) = h(x_2) = \ldots = h(x_k)\right) \le \frac{1}{|\mathbb{T}|^{k-1}}$$

(2) $\mathcal{H}$ is strongly k-universal if or all distinct $x_1, \ldots, x_k \in \mathbb{U}$ and all $y_1, \ldots, y_k \in \mathbb{T}$,

$$\Pr_{h\in\mathcal{H}}\left(h(x_1) = y_1 \wedge \ldots \wedge h(x_k) = y_k\right) = \frac{1}{|\mathbb{T}|^k}$$

- Strong k-universal => k-universal; 2-universal ⇔ universal
- Another characteristic

Let $2 \leq k \leq |\mathbb{U}|$, and let $\mathcal{H}$ be a family of hash functions from $\mathbb{U}$ to $\mathbb{T}$. Then $\mathcal{H}$ is strongly k-universal if and only if it has the following two properties.

k-Independence: *For all distinct $x_1, \ldots, x_k \in \mathbb{U}$ and all $y_1, \ldots, y_k \in \mathbb{T}$,*

$$\Pr_{h \in \mathcal{H}}\left(\bigwedge_{i=1}^{k} h(x_i) = y_i\right) = \prod_{i=1}^{k} \Pr_{h \in \mathcal{H}}\left(h(x_i) = y_i\right).$$

That is, the indicator random variables for the events $h(x_i) = y_i$ are independent.

Uniformity: *For all $x \in \mathbb{U}$ and $y \in \mathbb{T}$,*

$$\Pr_{h \in \mathcal{H}}(h(x) = y) = \frac{1}{|\mathbb{T}|}$$

- 构造 strongly k-universal(到比自己大)

- We choose a prime power $q \geq N$ and let $\mathbb{F}_q$ denote the field with q elements (unique up to isomorphism).
- We fix an arbitrary embedding $g_1 : \mathbb{U} \to \mathbb{F}_q$ and an arbitrary bijection $g_2 : \mathbb{F}_q \to \{0, \ldots, q-1\}$.
- For $\mathbf{a} = (a_0, \ldots, a_{k-1}) \in \mathbb{F}_q^k$, let $p_{\mathbf{a}} : \mathbb{F}_q \to \mathbb{F}_q$ be the polynomial function

$$p_{\mathbf{a}}(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{k-1} x^{k-1},$$

and let $f_{\mathbf{a}} : \mathbb{U} \to \{0, \ldots, q-1\}$ be the function $g_2 \circ p_{\mathbf{a}} \circ g_1$.

- 构造到比自己小很多

It remains to construct a strongly k-universal family mapping $\mathbb{U}$ to $\mathbb{T} := \{0, \ldots, M-1\}$ for an $M \ll N$.

- We choose a prime power $q \geq N$ and define the mappings $f_{\mathbf{a}} : \mathbb{U} \to \{0, \ldots, q-1\}$ for $\mathbf{a} \in \mathbb{F}_q^k$ as on the previous slide.
- We define functions $h_{\mathbf{a}} : \mathbb{U} \to \{0, 1 \ldots, M-1\}$ by

$$h_{\mathbf{a}}(x) := f_{\mathbf{a}}(x) \mod M.$$

- We let $\mathcal{H}_{q,M}^k := \{h_{\mathbf{a}} \mid \mathbf{a} \in \mathbb{F}_q^k\}$.

还得 M devides q，才是 strongly k-universal(否则是近似于)

- Zcount

（前提是数据流是 uniformly 选取的，不现实，所以才有了其他的方法，才要 hash）

Algorithm ZCount

```
z ← 0
while not end of stream do
    a ← next stream element
    if zeros(a) > z then
        z ← zeros(a)
return 2^(z+1/2)        ▷ z maximum number of zeros of stream elements
```

- Flajolet-Martin Algorithm

Let H be a strongly 2-universal family of hash functions from U to [M], where M is the first power of 2 greater than or equal to N.

Algorithm FMCount

h drawn uniformly at random from $\mathcal{H}$
$z \leftarrow 0$
while not end of stream **do**
  $a \leftarrow$ next stream element
  if zeros($h(a)$) $> z$ **then**
    $z \leftarrow$ zeros($h(a)$)
return $2^{z+1/2}$

- MCount

就是 run FMCount 2k-1 个，返回中位数

- pth frequency moment

$$F_p(\mathbf{a}) := \sum_{u \in \mathbb{U}} (f_u(\mathbf{a}))^p$$

- AMS-Estimator

Algorithm AMS-Estimator

$i = 0$
while not end of stream **do**
  $i \leftarrow i + 1$
  with probability $1/i$ **do**
    $a \leftarrow a_i$
    $r \leftarrow 0$
  if $a_i = a$ **then**
    $r \leftarrow r + 1$
return $i(r^k - (r-1)^k)$

(E(return) = Fk，这个方法接近真实的 p 不到 1/2，所以没法 boost)

- Tug-of-War

Algorithm Tug-of-War

draw h uniformly at random from $\mathcal{H}$
$x \leftarrow 0$
while not end of stream **do**
  $a \leftarrow$ next element from stream
  $x \leftarrow x + h(a)$
return x^2

(更好，不过要求 strongly 4-universal，且映射到{-1,1}，且针对 F2)

$(E(B)=F_2$ and $Var(B)<=2F_2^2)$

- Averaging the Tug-of-War estimator

Algorithm AVG-TOW(k)

draw $h_1, \ldots, h_k$ independently from $\mathcal{H}$
for $i = 1, \ldots, k$ do
  $x_i \leftarrow 0$
while not end of stream do
  $a \leftarrow$ next element from stream
  for $i = 1, \ldots, k$ do
    $x_i \leftarrow x_i + h_i(a)$
return $\frac{1}{k} \sum_{i=1}^{k} x_i^2$

Simple sketch

Algorithm SIMPLE SKETCH(k)

draw h from $\mathcal{H}$.
for $i = 1, \ldots, k$ do
  $S[i] := 0$
while not end of stream do
  $(a, c) \leftarrow$ next update
  $S[h(a)] \leftarrow S[h(a)] + c$
return S

Count min sketch

Algorithm COUNT MIN SKETCH(k, ell)

draw $h_1, \ldots, h_\ell$ independently from $\mathcal{H}$.
for $i = 1, \ldots, k$ do
  for $j = 1, \ldots, \ell$ do
    $S[i, j] \leftarrow 0$
while not end of stream do
  $(a, c) \leftarrow$ next update
  for $j = 1, \ldots, \ell$ do
    $S[h_j(a), j] \leftarrow S[h_j(a), j] + c$
return S

$d_u^* := \min_{j \in [\ell]} S[h_j(u), j]$ 最后用的是最小值当作 d*

- CM heavy hitter

Algorithm CM HEAVY HITTERS(k, ℓ, τ)

- compute a CM sketch S with parameters k, ℓ
- maintain $\|\mathbf{d}\|_1$ during the computation
- during the computation, maintain a set H of elements $u \in \mathbb{U}$ whose estimated value $d_u^* := \min_j S[h_j(u), j]$ is at least $\tau\|\mathbf{d}\|_1$
- after each update (or whenever H gets too large), remove those elements u whose value d_u^* has dropped below $\tau\|\mathbf{d}\|_1$ from H
- return all $u \in H$ with $d_u^* \geq \tau\|\mathbf{d}\|_1$

注意到，少写一个步骤，就是把当前这个添加到 H 中

Chapter6

- Map worker failure: completed 也 reset to idle; Reduce worker: completed are not reset

Matrix multiplication

1.

First Map-Reduce Round

MAP function: On input $(A, (i, j, v))$, emit $(j, (A, i, v))$.
On input $(B, (j, k, w))$, emit $(j, (B, k, w))$.

REDUCE function: On input $(j, values)$, emit $((i, k), vw)$ for all $(A, i, v), (B, k, w) \in values$.

Second Map-Reduce Round

MAP function: The identity function: on input $((i, k), x)$, emit $((i, k), x)$.

REDUCE function: On input $((i, k), values)$, compute the sum x^* of all $x \in values$ and emit $(C, (i, k, x^*))$.

2.

MAP function: On input $(A, (i, j, v))$, emit all key-value pairs $((i, k), (A, j, v))$ for $k \in [n]$.

On input $(B, (j, k, w))$, emit all key-value pairs $((i, k), (B, j, w))$ for $i \in [\ell]$.

REDUCE function: On input $((i, k), values)$, compute the sum x of all vw for $(A, j, v), (B, j, w) \in values$ and emit $(C, (i, k, x))$

- Communication cost: sum of input size to all tasks
  1. 2plm + 2qmn + 2pqlmn
  2. plm + qmn + (p + q)lmn:
- Replication rate: Number of key-value pairs produced by all ***map*** tasks divided by the input size
- maximum load: Maximum input length for single ***reducer*** or reduce task
  1. pl+qn; pqm
  2. pm+qm
- 单轮，但是减少 communication 版本

Choose parameter s = number of stripes.
Let $h : [n] \to [s]$ be the mapping that assigns each row/column index to its stripe. For example, $h(i) = \lceil is/n \rceil$.

MAP function: On input $(A, (i, j, v))$, emit all key-value pairs $((h(i), u), (A, i, j, v))$ for $u \in [s]$.

On input $(B, (j, k, w))$, emit all key-value pairs $((t, h(j)), (B, j, k, w))$ for $t \in [s]$.

REDUCE function: On input $((t, u), values)$, for all $i \in h^{-1}(t)$ and all $k \in h^{-1}(u)$, compute the sum

$$c_{ik} := \sum_{(A,i,j,v),(B,j,k,w) \in values} vw$$

and emit $(C, (i, k, c_{ik}))$.

Multiway joins-Hypercube algorithm

MAP function: On input $(R_i, (a_1, \ldots, a_{k_i}))$, emit all pairs

$$\big((p_1, \ldots, p_k), (R_i, (a_1, \ldots, a_{k_i}))\big)$$

such that

- $p_j \in [s_j]$ for all $j \in [k]$,
- $p_j = h_j(a_{j'})$ for all $j \in [k], j' \in [k_i]$ such that $A_{ij'} = A_j$.

REDUCE function: On input $(\overline{p}, \mathit{values})$, compute

$$\mathcal{Q}(\overline{p}) := \mathcal{R}_1(\overline{p}) \bowtie \ldots \bowtie \mathcal{R}_m(\overline{p}),$$

where

$$\mathcal{R}_i(\overline{p}) := \{t \mid (R_i, t) \in \mathit{values}\},$$

and emit all pairs (Q, t) for $t \in \mathcal{Q}(\overline{p})$.

- Replication rate:

$$\frac{\sum_{i=1}^{m}\left(n_i \cdot \prod_{j \in [k] \setminus \mathrm{Idx}(i)} s_j\right)}{\sum_{i=1}^{m} n_i}$$

Chapter7

- Different “connect”

Connected is usually associated with undirected graphs (two way edges): there is a path between every two nodes.

Strongly connected is usually associated with directed graphs (one way edges): there is a route between every two nodes.

Complete graphs are undirected graphs where there is an edge between every pair of nodes

- Strongly connected => average probability is stationary
- e_{n+1}={0,0,...,0,1} (n 个 0)
- markov chain is connected if Graph is strongly connected
- $p_t=p_0Q^t$
- Aperiodic if the greatest common divisor of the length of all cycles in GQ is 1. A Markov chain is ergodic if it is connected and aperiodic.
- Connected a->π, ergodic p->π
- 任意构造 ergodic

Let $Q \in \mathbb{R}^n$ be the transition matrix of a connected Markov chain. Then for every α with $0 < \alpha < 1$,

$$\alpha Q + (1 - \alpha) I,$$

where I is the $(n \times n)$ identity matrix, is the transition matrix of an ergodic Markov chain with the same stationary distribution.

- MCMC-Metropolis-Hastings Algorithm

$$q_{uv} := \begin{cases} \frac{1}{d} & \text{if } uv \in E(G) \text{ and } p(v) \geq p(u) \\ \frac{1}{d} \cdot \frac{p(v)}{p(u)} & \text{if } uv \in E(G) \text{ and } p(v) < p(u) \\ 1 - \sum_{v' \in N(u)} q_{uv'} & \text{if } u = v \\ 0 & \text{otherwise} \end{cases}$$

```
b ← { 1  with probability deg(u)/d,
    { 0  otherwise
if b = 1 then
    choose a neighbour v' ∈ N(u) in G uniformly at random
    if p(v') ≥ p(u) then
        v ← v'
    else
        v ← { v' with probability p(v')/p(u)
            { u  with probability 1 − p(v')/p(u)
else
    v ← u
return v
```

- Page rank

$$q_{ij} = \begin{cases} \frac{1}{d_i^+} & \text{if } (i, j) \in E(G_{\text{web}}) \\ 0 & \text{otherwise.} \end{cases}$$

Initialize n webpages with w=(w1,w2…wn)

w <-wQ

wj=∑wi/di+ (i,j)∈E(Gweb)

- 这个就成了，但是网络不连接或者 ergodic （对于一个 state，下面二者选一）

$$q_{ij}^* := \begin{cases} (1 - r) q_{ij} + \frac{r}{n} & \text{if } d^+(i) > 0, \\ \frac{1}{n} & \text{if } d^+(i) = 0. \end{cases}$$

Exercises

- Exercise3

Fibonacci:

$$\frac{(\frac{1+\sqrt{5}}{2})^n-(\frac{1-\sqrt{5}}{2})^n}{\sqrt{5}}$$

- Exercise5

100 维球内生成点

每个点，gaussian100 次，生成的向量除以长度 √ ($x_1^{2,}x_2$^2...x_{100}^2)，现在在面上了

再选长度，uniformly distribute from 0~1，开 100 次方根。两者一乘，就得了。

下面这个是球面的：

We can draw a unit vector $\mathbf{x} \in \mathbb{R}^\ell$ uniformly at random by drawing $\mathbf{y} \in \mathbb{R}^\ell$ from an ℓ-dimensional spherical Gaussian with mean $\mathbf{0}$ and variance 1 and letting

$$\mathbf{x} := \frac{\mathbf{y}}{\|\mathbf{y}\|}.$$

- VC of triangles in 2-d is 7.
- 与坐标轴平行的直角三角形，左下角是直角。Vc 为 4
- $(1-x)^m<=e^{-mx}$
-

Questions

1. Chapter2, P27, yellow remark
2. Chapter2, log and ln (find out their base)
   Ln 应该是自然对数， log 很可能是 2
3. Chapter2, P29, yellow remark
4. Chapter3, P11, Taylor expansion, ln(1 - x) >= -x - x^2 (0<=x<=1/2)
5. Chapter3, P23, yellow remark
6. Chapter7, markov chain and markov chain monte carlo difference?
7. Tug of war 助教的代码有地方有问题。事实上，双射是怎么实现的。
8. 有空的话看看 tug of war 里助教说的 median of means 怎么实现的

考试卷子第三题，b,c,d

Exercise sheet 3, ex3, 斐波那契

Exercise sheet 7, ex1，c，应该是逆矩阵，不是转置。看看答案。最后要不要求倒数

ε,δ

αβχδεφγηιφκλμνοπθρστυϖωξψζ

θωερτψυιοπασδφγηφκλζξχϖβνμ