

## General Remarks

$d(x, y) = \ x - y\ $	$\text{dist}(x, y) = 1 - \text{sim}(x, y)$
$l_2$ -euclidean distance	$\sqrt{\sum_{i=1}^d (x_i - y_i)^2}$
$l_1$ -manhattan distance	$\sum_{i=1}^d  x_i - y_i $
$l_p$ -distance	$(\sum_{i=1}^d  x_i - y_i ^p)^{1/p}$
$l_\infty$ -distance	$\max_i  x_i - y_i $
Mahalanobis norm	$\ w\ _G^2 = \ Gw\ _2^2$
Cosine-Similarity	$\cos \frac{x^T y}{\ x\ _2 \ y\ _2}$
Jaccard-Distance	$1 - \text{sim}(A, B) = 1 - \frac{ A \cap B }{ A \cup B }$

## Function Properties

### Concave function

$$f(a + s) - f(a) \geq f(b + s) - f(b) \quad \forall a \leq b, s > 0$$

**Convex functions** A function  $f : S \rightarrow \mathbb{R}$ ,  $S \subseteq \mathbb{R}^d$ , is called convex if  $\forall x, x' \in S, \lambda \in [0, 1]$  it holds that  $\lambda f(x) + (1 - \lambda)f(x') \geq f(\lambda x + (1 - \lambda)x')$

### H-strongly convex

$$f(x') \geq f(x) + \nabla f(x)^T (x' - x) + \frac{H}{2} \|x' - x\|^2, H > 0$$

1-D:  $f$  is H-sc  $\Leftarrow f''(x) \geq H, \forall x$ .

d-D:  $f$  is H-sc  $\Leftarrow \lambda_{\min}(\nabla^2 f(x)) \geq H, \forall x$ .

### Subgradients

Given a convex not necessarily differentiable function  $f$ , a subgradient  $\mathbf{g}_x \in \nabla f(x)$  is the slope of a linear lower bound of  $f$ , tight at  $x$ , that is  $\forall x' \in S : f(x') \geq f(x) + \mathbf{g}_x^T (x' - x)$

## Locality Sensitive Hashing

### Near-duplicate detection

$$\{(x, y) \in X \times X : x \neq y, d(x, y) \leq \epsilon\}$$

**( $r, \epsilon$ )-neighbour search** Find all points with distance  $\leq r$  and no points with distance  $> (1 + \epsilon)r$  from query  $q$ . Pick  $(r, (1 + \epsilon) \cdot r, p, q)$ -sensitive family and boost.

### Min-hashing

$h(C) = h_\pi(C) = \min_{i:C(i)=1} \pi(i)$   
 $\pi(i) = h_{a,b}(i) = ((a \cdot i + b) \bmod p) \bmod N$ ,  
 $p$  prime (fixed)  $> N$ ,  $N$  number of documents  
 $(d_1, d_2, 1 - d_1, 1 - d_2)$ -sensitive with Jaccard sim.

- 1: **for** each column  $c$  **do**
- 2:     **for** each row  $r$  **do**
- 3:         **if**  $c$  has 1 in row  $r$  **then**
- 4:             **for** each hash fn  $h_i$  **do**
- 5:                  $M_{i,c} \leftarrow \min\{h_i(r), M_{i,c}\}$

**Band-hashing** Signature matrix into  $b$  bands of  $r$  hash fns, per-column into  $b$  hash tables. If any h-table has a collision, report candidate pair.  $s^r$  prob of col on band  $j$ :  $P(\text{col in } \geq 1 \text{ band}) = 1 - (1 - s^r)^b$

**( $d_1, d_2, p_1, p_2$ )-sensitivity** Assume  $d_1 < d_2, p_1 > p_2$ .  
 $\forall x, y \in S : d(x, y) \leq d_1 \Rightarrow Pr[h(x) = h(y)] \geq p_1$   
 $\forall x, y \in S : d(x, y) \geq d_2 \Rightarrow Pr[h(x) = h(y)] \leq p_2$

**r-way AND**  $h(x) = h(y) \iff \forall i : h_i(x) = h_i(y)$

$(d_1, d_2, p_1^r, p_2^r)$  - big  $r$ , more FN

**b-way OR**  $h(x) = h(y) \iff \exists i : h_i(x) = h_i(y)$

$(d_1, d_2, 1 - (1 - p_1)^b, 1 - (1 - p_2)^b)$  - big  $b$ , more FP

**AND-OR cascade**  $(d_1, d_2, 1 - (1 - p_1^r)^b, 1 - (1 - p_2^r)^b)$

**OR-AND cascade**

$(d_1, d_2, (1 - (1 - p_1)^b)^r, (1 - (1 - p_2)^b)^r)$

## Hash Functions

**Euclidean distance**  $h_{w,b}(x) = \lfloor (\frac{w^T x - b}{a}) \rfloor$  where

$$w \leftarrow \frac{w}{\|w\|_2}, w \sim \mathcal{N}(0, I), w_i \sim \mathcal{N}(0, 1),$$

$b \sim \text{Unif}([0, a])$ , yields  $(a/2, 2a, 1/2, 1/3)$ -sensitive

**Cosine distance**  $\mathcal{H} = \{h(v) = \text{sgn}(w^T v)\}$  where

$$w \sim \text{Unif}\{x \in \mathbb{R}^n : \|x\|_2 = 1\}$$

$$Pr(h_u(x) = h_v(y)) = 1 - \Theta_{x,y}/\pi$$

$(\Theta_1, \Theta_2, 1 - \Theta_1/\pi, 1 - \Theta_2/\pi)$ -sensitive

## Support Vector Machines

**SVM** SVM = Max margin linear classifier

$$\min_{w, \xi} \sum_{i=1}^n \frac{1}{2} w^T w + C \sum_{i=1}^n \xi_i \text{ s. t. } y_i w^T x_i \geq 1 - \xi_i \quad \forall i$$

Support vectors (SV) are all data points on the margin and data points with non-zero slack

**Regularized hinge loss formulation**  $C = 1/\lambda$

$$\min_w \lambda w^T w + C \sum_i \max(0, 1 - y_i w^T x_i)$$

**Norm-constrained hinge loss minimization**

$$\min_w \sum_i \max(0, 1 - y_i w^T x_i) \text{ s.t. } \|w\|_2 \leq \frac{1}{\sqrt{\lambda}}$$

**Strongly convex formulation**

$$\min_w \frac{1}{T} \sum_{t=1}^T \left( \frac{\lambda}{2} \|w\|_2^2 + \max(0, 1 - y_t w^T x_t) \right)$$

$$\text{s.t. } \|w\|_2 \leq \frac{1}{\sqrt{\lambda}}$$

$$\text{Proj}_S^{(2)}(w) = w \cdot \min \left( 1, \frac{1/\sqrt{\lambda}}{\|w\|} \right)$$

$$\text{Proj}_S^{(\infty)}(w) = \max \left( (1/\sqrt{\lambda}, \dots, 1/\sqrt{\lambda})^T, w \right)$$

**Small  $C$ , Big  $\lambda$ :** Greater margin, more misclassification

## Kernels

### Dual SVM Formulation

$$\max_{\alpha} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

$$\text{s.t. } 0 \leq \alpha_i \leq C$$

$$\Rightarrow \text{optimal } w : w^* = \sum_i \alpha_i^* y_i x_i = \sum_{i \in \text{SV}} \alpha_i^* y_i x_i$$

**Kernel trick** Substitute inner product  $x_i^T x_j$  in dual formulation and in classification function with  $k(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ , where  $\phi(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^{>d}$

**Kernel functions** A kernel is function  $k : X \times X \rightarrow \mathbb{R}$ :

1. Symmetry:  $\forall x, x' \in X : k(x, x') = k(x', x)$
2. PSD:  $\forall n \in \mathbb{N}$ , any set  $S = \{x_1, \dots, x_n\} \subseteq X$ , the Gram matrix is PSD.

## Random Features (Inverse Kernel Trick)

**Shift-invariant kernel**  $k(x, y) = k'(x - y)$ . Then the kernel has Fourier transform, such that:

$$k(x - y) = \int_{\mathbb{R}^d} p(w) \cdot e^{i w^T (x - y)} dw$$

where  $p(w)$  is the Fourier transformation, i.e. we map  $k(s)$  to another function  $p(w)$ .

**Random fourier features (prerequisites)** Interpret kernel as expectation  $k(x - y) =$

$$\int_{\mathbb{R}^d} p(w) \cdot \underbrace{e^{i w^T (x - y)}}_{g(w)} dw = \mathbb{E}_{w,b} [z_{w,b}(x) z_{w,b}(y)]$$

$$\text{where } z_{w,b}(x) = \sqrt{2} \cos(w^T x + b),$$

$$b \sim U([0, 2\pi]), w \sim p(w)$$

**Random fourier features (kernel approximation)**

1.  $w_i \sim p, b_i \sim U([0, 2\pi])$  for  $i = 1, \dots, m$ ; iid
2.  $z(x) = [z_{w_1, b_1}(x), \dots, z_{w_m, b_m}(x)] / \sqrt{m}$
3.  $z(x)^T z(y) = \frac{1}{m} \sum_{i=1}^m z_{w_i, b_i}(x) \cdot z_{w_i, b_i}(y)$
4. If  $m \rightarrow \infty$ , then (almost surely)  
 $z(x)^T z(y) \rightarrow \mathbb{E}_{w,b} [z_{w,b}(x) \cdot z_{w,b}(y)] = k(x - y)$

## Online Convex Programming

**Regret**  $R_T = (\sum_{t=1}^T f_t(w_t)) - \min_{w \in S} \sum_{t=1}^T f_t(w)$

**No-regret**  $\lim_{T \rightarrow \infty} \frac{R_T}{T} \rightarrow 0$

**Online convex programming (OCP)** If  $y_t w_t^T x_t < 1 : w_{t+1} = \text{Proj}_S(w_t - \eta_t \nabla f_t(w_t))$

$$\text{Proj}_S(w) = \arg \min_{w' \in S} \|w' - w\|_2 = \min \left( w, \frac{w}{\lambda \|w\|_2} \right)$$

**Regret for OCP**

$$\frac{R_T}{T} \leq \frac{1}{\sqrt{T}} [\|w_0 - w^*\|_2^2 + \|\nabla f\|_2^2]$$

where  $\|\nabla f\|_2^2 = \sup_{w \in S, t \in \{1, \dots, T\}} \|\nabla f(w)\|_2^2$

**Parallel stochastic gradient descent** 1. Split data into  $k$  subsets,  $k$  = number of machines, want  $k = O(1/\lambda)$

2. Each machine produces  $w_i$  on its subset

3. After  $T$  iterations, compute  $w = \frac{1}{k} \sum_{i=1}^k w_i$

**PEGASOS** Online SVM: H-sc. lossfn. Minibatch + reg

## Active Learning

**Uncertainty sampling** Repeat until all labels inferred:

1. Assign uncertainty score  $U_t(x)$  to each unlabeled data point:  
 $U_t(x) = U(x|_{x_{1:t-1}, y_{1:t-1}})$
2. Greedily pick the most uncertain point and request label  $x_t = \arg \max_x U_t(x)$  and retrain classifier

For SVM:  $U_t(x) = \frac{1}{|w_t^T x|}$

Cost to pick  $m$  labels:  $m \cdot n \cdot d + m \cdot C(m)$

$n$  = number of data points,  $d$  = dimensions,

$C(m)$  = cost to train classifier

**Hashing a hyperplane query** Draw  $u, v \sim \mathcal{N}(0, I)$ .

Then resulting two-bit hash is:

$$h_{u,v}(a, b) = \left[ \text{sign}(u^T a), \text{sign}(v^T b) \right]$$

Define the hash family:

$$h_{\mathcal{H}}(z) = \begin{cases} h_{u,v}(z, z) & \text{if } z \text{ is a database point vector} \\ h_{u,v}(z, -z) & \text{if } z \text{ is a query hyperplane vector} \end{cases}$$

**Version space** Set of all classifiers consistent with the data:  $V(D) = \{w : \forall (x, y) \in D : \text{sign}(w^T x) = y\}$

**Relevant version space**  $\hat{V}(D; U)$  describes all possible labelings  $h$  of all unlabeled data  $U$  that are still possible under some model  $w$ , or,

$$\hat{V}(D; U) = \{h : U \rightarrow \{+1, -1\} : \exists w \in V(D) \forall x \in U : \text{sign}(w^T x) = h(x)\}$$

**Generalized Binary Search (GBS)**

- 1: Start with  $D = \emptyset$
- 2: **while**  $|\hat{V}(D; U)| > 1$  **do**
- 3:   **for each** unlabeled example  $x$  in  $U$  **do**
- 4:      $v^+(x) = |\hat{V}(D \cup \{(x, +)\}; U)|$
- 5:      $v^-(x) = |\hat{V}(D \cup \{(x, -)\}; U)|$
- 6:     ▷ number of labelings left if  $x$  is  $-/+$
- 7:   Pick  $x^* = \arg \min_x \max(v^-(x), v^+(x))$
- 8:    $D = D \cup \{x^*\}$

**Decision rules** for GBS SVM,  $m \sim \text{margin} \sim \frac{1}{\|w\|}$

**Max-min margin**  $\max_x \min(m^+(x), m^-(x))$

**Ratio margin**  $\max_x \min\left(\frac{m^+(x)}{m^-(x)}, \frac{m^-(x)}{m^+(x)}\right)$

## Clustering

### K-Means

**Cost Function**  $L(\mu) = L(\mu_1, \dots, \mu_k) =$

$$\sum_{i=1}^N \underbrace{\min_{j \in \{1, \dots, k\}} \|x_i - \mu_j\|_2^2}_{d(\mu, x_i)}$$

**Objective**  $\mu^* = \arg \min_{\mu} L(\mu)$

**Algorithm** Until convergence:

- 1: Assign each point  $x_i$  to closest center

$$z_i \leftarrow \arg \min_{j \in \{1, \dots, l\}} \|x_i - \mu_j^{(t-1)}\|_2^2$$

- 2: Update center as mean of assigned data points

$$\mu_j^{(t)} \leftarrow \frac{1}{n_j} \sum_{i: z_i = j} x_i$$

**Online k-means algorithm**

$$\frac{\text{dd}(\mu, x_t)}{\mathbf{d}\mu_j} = \begin{cases} 0 & \text{if } j \notin \arg \min_i \|\mu_i - x_t\|^2 \\ 2(u_j - x_t) & \text{else} \end{cases}$$

1. Initialize centers randomly
2. For  $t = 1 : N$ 
  - Find  $c = \arg \min \|\mu_j - x_t\|_2$
  - $\mu_c = \mu_c + \eta_t(x_t - \mu_c)$
3. For convergence:  $\sum_t \eta_t = \infty$  and  $\sum_t \eta_t^2 < \infty$ , e.g.  $\eta_t = \frac{c}{t}$ .

## Coresets

**Idea** Replace many points by one weighted point

$$L_k(u; C) = \sum_{(w, x) \in C} w \cdot \min_j \|u_j - x\|_2^2$$

$(k, \epsilon)$ -coreset  $C$  is called a  $(k, \epsilon)$ -coreset for  $D$ , if for all  $\mu$ :  $(1 - \epsilon)L_k(\mu; D) \leq L_k(\mu; C) \leq (1 + \epsilon)L_k(\mu; D)$

**Operations Merge:** union of two  $(k, \epsilon)$ -cses is a  $(k, \epsilon)$ -cset

**Compress:**  $(k, \delta)$ -cset of a  $(k, \epsilon)$ -cset of  $D$  is a  $(k, \epsilon + \delta + \epsilon\delta)$ -cset of  $D$

**Construction  $D^2$ -sampling:** iteratively build  $B = \emptyset$  by

$$\text{sampling} \propto \frac{d(x, B)^2}{\sum_{x' \in X} d(x', B)^2}$$

**Importance sampling:**

$$\propto \frac{\alpha d(x, B)^2}{c_{\Phi}} + \frac{2\alpha \sum_{x' \in B_x} d(x', B)^2}{|B_x| c_{\Phi}} + \frac{4|X|}{|B_x|} \text{ where}$$

$$c_{\Phi} = \frac{1}{|X|} \sum_{x \in X} d(x, B)^2$$

$B_x$  = pts in  $X$  that belong to same cluster as  $x$

## Bandits

**Regret**  $R_T = \sum_{t=1}^T (\mu^* - \mu_{i_t})$ ,  $i_t$  chosen arm at time  $t$

**$\epsilon$ -greedy** The algorithm goes as follows:

1. Set  $\epsilon_t = \mathcal{O}(\frac{1}{t})$
2. With probability  $\epsilon_t$ : explore by picking uniformly at random
3. With probability  $1 - \epsilon_t$ : exploit by picking arm with highest empirical mean

Regret:  $R_T = \mathcal{O}(k \log(T))$

## UCB1 & LinUCB

**Hoeffding's inequality**

$$\Pr(|\mu - \frac{1}{m} \sum_{t=1}^m X_t| \geq b) \leq 2 \cdot \exp(-2b^2 m)$$

**Confidence bound** Want: Hoeffding  $\leq \delta$

$$\Rightarrow b = \sqrt{\frac{1}{2m} \ln \frac{2}{\delta}}$$

**UCB/Mean update**  $UCB(i) = \hat{\mu}_i + \sqrt{\frac{2 \ln t}{\eta_i}}$

$$j = \arg \max_i UCB(i)$$

$$\hat{\mu}_j = \hat{\mu}_j + \frac{1}{\eta_j} (y_t - \hat{\mu}_j)$$

**Contextual bandits** Reward is now  $y_t = f(x_t, z_t) + \epsilon_t$  with  $z_t$  user features. For us:  $f(x_i, z_t) = w_{x_i}^T z_t$

**LinUCB (disjoint)** Ridge regression on  $z_t$ :

$$\hat{w}_i = (D_i^T D_i + I)^{-1} D_i^T y_i.$$

$$|\hat{w}_i^T z_t - w_i^T z_t| \leq \alpha \sqrt{z_t^T (D_i^T D_i)^{-1} z_t} \text{ with probability } 1 - \delta \text{ if } \alpha = 1 + \sqrt{\log(2/\delta)/2}$$

$$R_T/T = \mathcal{O}(d \cdot d' \cdot \text{poly log } T / \sqrt{T})$$

**Hybrid model** Reward is now

$$y_t = w_{x_t}^T z_t + \beta^T \phi(x_t, z_t) + \epsilon_t$$

**Rejection sampling** First obtain data log through pure exploration, and then reiterate:

1. Get event  $(x_t^{(1)}, \dots, x_t^{(k)}, z_t, a_t, y_t)$  from log
2. Use algorithm that is testing to pick  $a'_t$ :
  - If  $a'_t = a_t \Rightarrow$  Feed back reward  $y_t$
  - Else ignore log line
3. Stop when T rewards have been fed back

## Submodular Functions

**Submodularity** A function  $F : 2^V \mapsto \mathbb{R}$  is called submodular iff for all  $A \subseteq B, s \notin B$ :

$$F(A \cup \{s\}) - F(A) \geq F(B \cup \{s\}) - F(B)$$

**Monotonic** if  $S \subseteq T \Rightarrow F(S) \leq F(T)$

**Union-intersection def.** Same preconditions as above:

$$F(A) - F(A \cap B) \geq F(A \cup B) - F(B)$$

**Closure properties**  $F, F_i$  submodular on  $V$ ;  $S, W \subseteq V$

**Linear Combinations**

$$F'(S) = \sum_i \lambda_i F_i(S), \lambda_i \geq 0$$

**Restriction**  $F'(S) = F(S \cap W)$

**Conditioning**  $F'(S) = F(S \cup W)$

**Reflection**  $F'(S) = F(V \setminus S)$

**Monotonic truncation**  $F$  also monotonic,

$$F'(S) = \min(c, F(S)), c \in \mathbb{R}$$

**Min/Max** For  $F_{1,2}(A)$ ,  $\max\{F_1(A), F_2(A)\}$  or

$\min\{F_1(A), F_2(A)\}$  **not** submodular in general.

**Concavity**  $F(A) = g(|A|)$  where  $g : \mathbb{N} \mapsto \mathbb{R}$ , then  $F$  submodular iff  $g$  concave.

**Lazy Greedy** Optimizing submodular **monotonic** functions:  $\Delta(s|A_i) \geq \Delta(s|A_{i+1})$ .

- 1:  $A_0 = \emptyset$ . Keep ordered list of marginal benefits  $\Delta_s$  from previous iterations  $j < i$ , for all  $s \in S$ , set of articles
- 2: **for**  $i = 1 \dots k$  **do**
- 3:    $s$  is the article for which  $\Delta_s$  is at the top
- 4:   Update:  $\Delta_s = F(A_{i-1} \cup \{s\}) - F(A_{i-1})$
- 5:   **if**  $\Delta_s$  is not top element anymore **then**
- 6:     re-sort the list, goto 3
- 7:    $A_i = A_{i-1} \cup \{s\}$