1 LSH
1.1 Min Hashing
$h_\pi(C) = \min_i \pi(i)$ first row in randomly shuffled column that contains a one.
$P(h(C_1) = h(C_2)) = sim_J(C_1, C_2)$ Jaccard similarity
Shuffling implemented with hash function $\pi(i) = ai + b \mod n$

```
for each column c do
    for each row r do
        if c has 1 in row r then
            for each hash function h_i do
                M(i,c) ← min{h_i(r), M(i,c)}
```

1.2 Hashing Signature Matrix
M partitioned into b bands of r rows. Probability C1,C2 collide on at least one band: $1 - (1 - s^r)^b$
Family of hash functions F is $(d_1 1, d_2, p_1, p_2)$ sensitive if

$$\forall x, y \in S : d(x, y) \leq d_1 \implies P[h(x) = h(y)] \geq p_1$$
$$\forall x, y \in S : d(x, y) \geq d_2 \implies P[h(x) = h(y)] \leq p_2$$

r-way and: For $h = [h_1, \cdot, h_r]$ $h(x) = h(y) \leftrightarrow h_i(x) = h_i(y)$ $\forall i$
F' is (d_1, d_2, p_1^r, p_2^r) sensitive
b-way or: For $h = [h_1, \cdot, h_b]$ $h(x) = h(y) \leftrightarrow h_i(x) = h_i(y)$ for some i
F' is $(d_1, d_2, 1 - (1 - p_1)^b, 1 - (1 - p_2)^b)$ sensitive

1.3 Other Distance Functions
Cosine distance: Family of hash functions for uniformly random vector u
$h_u(x) = \text{sign}(u^T x)$ $\quad P(h_u(x) = h_u(y)) = 1 - \frac{\theta_{x,y}}{\pi}$
Euclidean distance: Family functions for random line divided into a buckets. If distance $d = ||x - y||_2 >> a$, $h(x) = h(y)$ with low probability. F forms a $(\frac{a}{2}, 2a, \frac{1}{2}, \frac{1}{3})$ sensitive family.

2 Supervised Learning
2.1 Canonical hyperplanes
$x' = \bar{x} + \frac{w}{||w||}\gamma$
$w^T x' + b = w^T \bar{x} + b + \frac{w^T w}{||w||}\gamma = 1$
$\gamma = 1/||w||$

2.2 Formulations

$$\min_{w,b} w^T w \quad s.t.\ y_i(w^T x_i + b) \geq 1$$
$$\min_{w,b} w^T w + C \sum_i \xi_i \quad \text{s.t. } y_i(w^T x_i + b) \geq 1 - \xi_i$$
$$\min_{w,b} w^T w + C \sum_i \max\{0, 1 - y_i(w^T x_i + b)\}$$
$$\min_{w,b} \lambda w^T w + \sum_i \max\{0, 1 - y_i(w^T x_i + b)\}$$
$$\min_{w,b} \sum_i \max\{0, 1 - y_i(w^T x_i + b)\} \quad \text{s.t. } ||w|| < 1/\lambda$$

2.3 Online Convex Programming
Regret: $R_T = \sum_{t=1}^T l_t - \min_w \sum_{t=1}^T f_t(w)$
OCP Regret: $R_T \leq \frac{||S||^2\sqrt{T}}{2} + (\sqrt{T} - 1/2)||\nabla f||^2$ for $\eta_t = 1/\sqrt{t}$

If new point violates margin $y_t(w_t x_t + b) < 1$
Update $w_{t+1} = w_t - \eta_t \nabla f_t(w_t)$
Project $\min\{w, \frac{w}{||w||\lambda}\}$

2.4 Modifications
- **SGD:** training samples picked at random.
- **PEGASOS:** uses minibatch of random samples, loss function with $\lambda/T w^T w$ term. Strongly convex loss function for better convergence.
- **PSGD:** randomly partition data to k machines which run SGD independently. After T iterations take weighted sum of obtained weights.
  $w_T = \frac{1}{k}\sum_{i=1}^k w_i$. Parallelization helps if $k = O(1/\lambda)$
- **L1-ball projection:** $Proj_S(w) = \arg\min_{||w'||_1 \leq c} ||w' - w||_2$ using $w_i = \text{sign}(w_i)\max\{w_i - \beta, 0\}$
- **multi-class:** $l(W, (x, y)) = \max_{r \in k \setminus y}[1 - (Wx)_y + (Wx)_r]_+$

2.5 Feature selection
L1 regularization: replace $||w||_2$ with $||w||_1$ for sparse solutions.
modified projection: $\bar{w}_i = \text{sign}(w_i)\max\{|w_i| - \beta, 0\}$ where β is computed in linear time.

3 Active Learning
Pick most uncertain points for labeling $x^* = \arg\min_{x_i \in U} |w^T x_i$

3.1 Hashing a Hyperplane
$h_{u,v} = [h_u(a), h_v(b)] = [\text{sign}(u^T a), \text{sign}(v^T b)]$ $\quad u, v\ \mathcal{N}(0, 1)$

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

$$P(h(w) = h(x)) = P(h_u(w) = h_u(x))P(h_v(-w) = h_v(x))$$
$$= \frac{1}{4} - \frac{1}{\pi^2}(\theta - \frac{\pi}{2})^2$$

3.2 Generalised binary search (GBS)
$D = \{(x_1, y_1) \cdots (x_n, y_n)\}$ $\quad \mathcal{V}(D) = \{w : \forall (x, y) \in D\ \text{sign}(w^t x) = y\}$
unlabeled pool: $U = \{x_1', \cdots, x_n'\}$
relevant version space: $\hat{\mathcal{V}}(D, U) = \{h : U \to \{+1, -1\} : \exists w \in \mathcal{V}(D) \forall x \in U\ \text{sign}(w^T x) = h(y)\}$

```
start with D = {}
while |V̂(D,U)| > 1 do
    for each unlabeled example x in a compute do
        |V̂(D ∪ {x+}, U)| = w+
        |V̂(D ∪ {x−}, U)| = w−
    pick example where |w− − w+| is smallest
    request label and add to D
```

3.3 Version space reduction
max-min margin: $\max |w_- - w_+|$ **ratio margin:** $\max\{\frac{w_-}{w_+}, \frac{w_+}{w_-}\}$

3.4 Tricks
- only pick from a random subsample of points
- only use 'fancy' criteria for first 10 examples then switch to uncertainty sampling
- occasionally pick points uniformly at random

4 Unsupervised Learning
4.1 Online K-means
$L(\mu) = \sum_{i=1}^N \min_j ||u_j - x_i||_2^2$

$$\frac{\partial f_i(\mu)}{\partial \mu_j} = \begin{cases} 0 & \text{if} j \notin \arg\min ||\mu_j - x_i||^2 \\ 2(\mu_j - x_i) & \text{otherwise} \end{cases}$$

```
initialize centers randomly
for t = 1 : N do
    c ← arg min_j ||μ_j − x_t||^2
    μ_c ← μ_c + η_t(x_t − μ_c)
```

Converges to local optimum given $\sum_t \eta_t = \infty$ and $\sum_t \eta_t^2 < \infty$

4.2 Constructing Coresets

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

$$C = \{(x_{R1}, \frac{1}{nq(R1)}), \cdots, (x_{Rn}, \frac{1}{nq(Rn)})\}$$

$$L(\mu, C) = \sum_{i=1}^{n} \frac{1}{nq(R_i)} \min_j ||x_{Ri} - \mu_j||^2$$

$$\mathrm{E}[L(\mu, C)] = \sum_{l=1}^{N} \sum_{i=1}^{n} \frac{q(l)}{nq(l)} \min_j ||x_{Rl} - \mu_j||^2 = L(\mu, D)$$

$B \leftarrow \emptyset, D \leftarrow D'$
while $D' \neq \emptyset$ **do**
  $S \leftarrow$ uniformly sample $10dk \log(\frac{1}{epsilon})$points from D'
  Remove $\frac{|D'|}{2}$ points nearest to S from D'
  $B \leftarrow B \cup S$
Partition D into Voronoi cells D_b centered at $b \in B$
$q(x) \propto \frac{5}{|D_b|} + \frac{dist(x,B)^2}{\sum_{x'} dist(x',B)^2}$ $\gamma(x) = \frac{1}{|C|q(x)}$

$C \leftarrow$ sample $10dk \log^2 n \log(\frac{1}{\delta})/\epsilon^2$ from D via q

Can find a coreset in $\mathcal{O}(k^3/\epsilon^{d+1})$ and weak coreset in $\mathcal{O}(\text{poly(k, d}/\epsilon))$

merge: union of two (k, ϵ) coresets is a (k, ϵ) coreset
compress: (k, δ) coreset of a (k, ϵ) coreset is a $(k, \epsilon + \delta + \epsilon\delta)$

5 Recommender Systems

regret: $\mu^* n - \mu_j \sum_{j=1}^{K} \mathrm{E}[n_j(n)]$ after n steps where μ^* is expectation of optimal machine payoff and μ_j expectation of machine j.

5.1 Epsilon Greedy

set ϵ_t $1/t$ and explore with $P = \epsilon_t$, exploit otherwise
$R_T = \mathcal{O}(k \log T)$

5.2 Confidence Bounds

Let X_1, X_m be iid RV taking values in [0,1]
$\mu = \mathrm{E}[X]$ $\hat{\mu_m} = \frac{1}{m}\sum_{l=1}^{m} X_l$
then $P(|\mu - \hat{\mu}_m| \geq b) \leq 2\exp(-2b^2 m)$ with $b = \sqrt{\frac{1}{2m}\log\frac{2}{\delta}}$

5.3 UCB1

Optimum in face of uncertainty

set $\hat{\mu}_1, \cdots, \hat{\mu}_k = 0, n_1, \cdots, n_k = 0$
for $t = 1 : T$ **do**
  $\text{UCB}(i) = \hat{\mu}_i + \sqrt{\frac{2\log t}{n_i}}$
  pick $j = \arg\max_i \text{UCB}(i)$ and observe y_t
  $n_j \leftarrow n_j + 1$ $\hat{\mu}_j \leftarrow \hat{\mu}_j + \frac{y_t - \hat{\mu}_j}{n_j}$

6 LinUCB

$y_t = w_i^T z_t + \epsilon_t$ $\arg\min_w \sum_{t=1}^{n}(y_t - w_i^T z_i) + ||w||_2^2$
closed form solution: $\hat{w}_i = (D_i^T D_i + I)^{-1} D_i^T y_i$ (Ridge Regression)

$$D_i \begin{bmatrix} - & z_1 & - \\ & \vdots & \\ - & z_n & - \end{bmatrix} \qquad y_i = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

$$|\hat{w}_i^T z_t - w_i^T z^T| \leq \alpha\sqrt{z_t^T (D_i^T D_i + I)^{-1} z_t}$$

with probability at least $1-\delta$ as long as $\alpha = 1 + \sqrt{\log(2/\delta)/2}$
$R_T/T = \mathcal{O}(dd'\text{poly}(\log(T/\sqrt{T})))$

6.1 Hybrid

$y_t = w_i^T z_t + \beta^T \phi(x_i, z_t) + \epsilon_t$

6.2 Rejection Sampling

for $t = 1 : T$ **do**
  Get next event from log $(\{x_t^{1:k}\}, z_t, a_t, y_t)$
  Use bandit algorithm to select a_t'
  if $a_t' = a_t$ **then**
    feed back reward y_t to the algorithm
  else
    ignore log line

7 Appendix

7.1 Distance Functions

$d(s, t)$ is a distance function if
a) $d(s, t) \geq 0$ **b)** $d(s, s) = 0$ **c)** $d(t, s) = d(s, t)$ **d)** $d(s, r) \leq d(s, t) + d(t, r)$

l_p distance:	$d_p(x, x') = (\sum_{i=1}^{D} \|x - x'\|^p)^{1/p}$
l_∞ distance:	$d_\infty(x, x') = \max_i \|x - x'\|$
cosine distance:	$d(x, x') = \frac{arc\cos(x^T x')}{\|\|x\|\|_2 \|\|x'\|\|_2}$
edit distance:	how many inserts/deletes to transform one string into another
Jaccard distance:	$d(x, x') = 1 - \frac{x \cap x'}{x \cup x'}$

7.2 Convex Programming

convex set:
S is convex if $\forall x, x' \in S, \lambda \in [0, 1]$ $\lambda x + (1-\lambda)x' \in S$
convex function:
$f : \mathbb{R}^d \rightarrow \mathbb{R}$ is convex if $\forall x, x' \in S, \lambda \in [0, 1]$
$f(\lambda x + (1-\lambda)x') \leq \lambda f(x) + (1-\lambda)f(x')$
subgradient:
for $f(x)$ convex, $g(x)$ is a subgradient of f at x_0 iff
$\forall x$ $f(x) \geq f(x_0) + g^T(x - x_0)$

7.3 Submodularity

Set function F on V is submodular if $\forall A \subseteq B$ and $s \notin B$ and $a_1, a_2 \notin A$

$$F(A \cup \{s\}) - F(A) \geq F(B \cup \{s\}) - F(B) \quad \text{Diminishing}$$
$$F(S) + F(T) \geq F(S \cup T) + F(S \cap T)$$
$$F(A \cup \{a_1\}) + F(A \cup \{a_2\}) \geq F(A \cup \{a_1, a_2\}) + f(A)$$

Closedness under

Non-negative linear combinations: for $F_1, \cdots, F_m$ submodular functions and $\lambda_1, \cdots, \lambda_m \geq 0$

$$F(A \cup \{s\}) - F(A) = \left(\sum_i \lambda_i F_i(A \cup \{s\})\right) - \left(\sum_i \lambda_i F_i(A)\right)$$
$$= \sum_i \lambda_i(F_i(A \cup \{s\} - F_i(A)) \geq \sum_i \lambda_i(F_i(B$$
$$\geq (F(B \cup \{s\} - F(B))$$

Restrictions: for F submodular over V and $S, W \subseteq V$

$$F'(S) = F(S \cap W) \quad \text{submodular}$$

Conditioning: for F submodular over V and $S, W \subseteq V$

$$F'(S) = F(S \cup W) \quad \text{submodular}$$

Reflection: for F submodular over V

$$F'(S) = F(V \backslash S) \quad \text{submodular}$$

For $F_{1,2}(A)$, $\max\{F_1(A), F_2(A)\}$ or $\min\{F_1(A), F_2(A)\}$ **not** submodular in general.

Example:

$$\max_{j \in A \cup \{s\}} M_{ij} - \max_{j \in A} M_{ij} = \max\{M_{is}, \max_{j \in A} M_{ij}\} - \max_{j \in A}$$
$$= \max\{M_{is} - \max_{j \in A} M_{ij}, 0\}$$

If $A \subseteq B$ then $\max_{j \in A} M_{ij} \leq \max_{j \in B} M_{ij}$

Lazy Greedy Algorithm

start with $A_0 = \{\}$
keep ordered list of marginal benefits Δ_i from prev. it.
for i=1:k **do**
  $\Delta_i = F(A_{i-1} \cup \{s^*\})$ where s^* is for top element
  if i is still top element **then**
    $A_i = A_{i-1} \cup \{s^*\}$
  else
    resort Δ_i and assign greedily