L1-3

: $H(X)=-\sum_x p(x)\log p(x)$

: $H(X,Y)=-\sum_{x,y} p(x,y)\log p(x,y)$

: $H(Y|X)=-\sum_{x,y} p(x,y)\log p(y|x)$

: $H(p,q)=-\sum_x p(x)\log q(x)$

KL : $D(p\|q)=\sum_x p(x)\log\frac{p(x)}{q(x)}$

: $I(X;Y)=\sum_{x,y} p(x,y)\log\frac{p(x,y)}{p(x)p(y)}$

$H(X,Y)=H(X)+H(Y|X)=H(Y)+h(X|Y)$

$I(X;Y)=H(X)-H(X|Y)=H(X)+H(Y)-H(X,Y)$

Vene $I(X;Y)=H(X)\cap H(Y); H(X;Y)=H(X)\cup H(Y)$

: Bagging→Random Forest Boosting→GBDT

Splitting Criterion: split feature Training ErrorRate $\text{Error}(x_j)=\sum_{i=1}^m \frac{|D_i|}{|D|}\cdot\text{Error}(D_i)$

$I(Y;X_i)$:

$I(y;x_d)=H(y)-H(y|x_d)=H(y)-\sum_{v\in V(x_d)} f_v\cdot H(Y_{x_d=v})$,

$H(y)$ $V(x_d)$ x_d $f_v=\frac{|D_v|}{|D|}$ v $Y_{x_d=v}$ $x_d=v$

pruning Evaluate each split using a *validation* dataset by comparing the validation error rate **with and without** that split (Greedily) remove the split that most decreases the validation error rate Stop if no split improves validation error, otherwise repeat

L4 KNN

M N Naive Train$\mathcal{O}(1)$,Predict$\mathcal{O}(MN)$ k-d Tree Train:$\mathcal{O}(MN\log N)$,Predict,$\mathcal{O}(2^M\log N)$

Experimental Design: train-subset+validation(all-train)

L5 perceptron

$\boldsymbol{w}\leftarrow\boldsymbol{w}+\eta y_i\boldsymbol{x}_i$; $b\leftarrow b+\eta y_i$

w b decision boundary;

$\mathbf{w}=\sum_{i=1}^N \alpha_i y^{(i)}\mathbf{x}^{(i)}$

$\frac{\|\mathbf{w}^T(\mathbf{x}''-\mathbf{x}')\|}{\|\mathbf{w}\|_2}=\frac{\|\mathbf{w}^T\mathbf{x}''+b\|}{\|\mathbf{w}\|_2}=\frac{y_i(\boldsymbol{w}^T\boldsymbol{x}_i+b)}{\|\boldsymbol{w}\|}$

L6 SVM

KernelMethods:$K(\mathbf{x},\mathbf{z})=\Phi(\mathbf{x})\Phi(\mathbf{z})$;$\Phi(\mathbf{x})$=x poly ,

$w_t=a_{i_1}x_{i_1}+\cdots+a_{i_k}x_{i_k}$,$w_t\cdot x=a_{i_1}x_{i_1}\cdot x+\cdots+a_{i_k}x_{i_k}$.

x replace with $a_{i_1}K(x_{i_1},x)+\cdots+a_{i_k}K(x_{i_k},x)$

Mercer: 1.$K(x,z)=K(z,x)$; 2.$\mathbf{a}^\top\mathbf{K}\mathbf{a}\geq 0$(semi-definite)

: $K(x,z)=c_1K_1(x,z)+c_2K_2(x,z)$, $\phi(x)=\left(\sqrt{c_1}\phi_1(x),\sqrt{c_2}\phi_2(x)\right)$

: $\max_{w,\alpha}\gamma$, : $\|\mathbf{w}\|=1$ $y_i(\mathbf{x}_i\cdot\mathbf{w}+\alpha)\geq\gamma$

$\mathbf{w}'=\frac{\mathbf{w}}{\gamma}$ $\alpha'=\frac{\alpha}{\gamma}$

$\min_{\mathbf{w}',\alpha'}\|\mathbf{w}'\|^2$ $y_i(\mathbf{x}_i\cdot\mathbf{w}'+\alpha')\geq 1$

SVM Optimizationl

Primal (soft-margin) form: minimize over $w,b,\{\xi_i\}$ the objective $\frac{1}{2}\|w\|^2+C\sum_{i=1}^m\xi_i$ subject to $y_i(w^\top x_i+b)\geq 1-\xi_i$ and $\xi_i\geq 0$.

Lagrangian: introduce multipliers $\alpha_i\in[0,C]$, $\mu_i\geq 0$, and form $\mathcal{L}=\frac{1}{2}\|w\|^2+C\sum_i\xi_i-\sum_i\alpha_i\left[y_i(w^\top x_i+b)-1+\xi_i\right]-\sum_i\mu_i\xi_i$.

Stationarity w.r.t. w,b,ξ gives $w=\sum_i\alpha_i y_i x_i$, $\sum_i\alpha_i y_i=0$, and $\alpha_i+\mu_i=C$. Eliminating w,b,ξ yields the dual QP: maximize $\sum_{i=1}^m\alpha_i-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j x_i^\top x_j$ subject to $\sum_{i=1}^m\alpha_i y_i=0$ and $0\leq\alpha_i\leq C$.

Decision function: $f(x)=\text{sign}\left(\sum_{i=1}^m\alpha_i y_i x_i^\top x+b\right)$, where support vectors satisfy $0<\alpha_i\leq C$.

L7 Regression

KNN: store all (x,y) pairs; k=1 return nearest y; k=2 return weighted average y.

Decision Tree Regression: model as a binary tree with each internal node testing $x_j\leq s$ and each leaf predicting a constant $\hat{y}_\ell$.

Split criterion: choose feature j and threshold s to minimize $\Delta_{\text{MSE}}(j,s)=\frac{|\mathcal{D}_{\text{left}}|}{|\mathcal{D}|}\text{MSE}(\mathcal{D}_{\text{left}})+\frac{|\mathcal{D}_{\text{right}}|}{|\mathcal{D}|}\text{MSE}(\mathcal{D}_{\text{right}})$, where $\text{MSE}(\mathcal{D})=\frac{1}{|\mathcal{D}|}\sum_{(x,y)\in\mathcal{D}}(y-\bar{y})^2$.

Leaf prediction: in leaf ℓ with data $\mathcal{D}_\ell$, set $\hat{y}_\ell=\frac{1}{|\mathcal{D}_\ell|}\sum_{(x,y)\in\mathcal{D}_\ell}y$.

Optimization Method #1: Gradient Descent

Residual: $e^{(i)}=y^{(i)}-\hat{y}^{(i)}$.

MSE objective: $J(\theta)=\frac{1}{N}\sum_{i=1}^N\left(y^{(i)}-(\theta^T x^{(i)}+b)\right)^2$.

Gradient: $\nabla_\theta J(\theta)=\left[\frac{\partial J}{\partial\theta_1}\ \cdots\ \frac{\partial J}{\partial\theta_M}\right]^T=\sum_{i=1}^N\left(\theta^T x^{(i)}-y^{(i)}\right)x^{(i)}$.

Gradient descent update: $\theta\leftarrow\theta-\gamma\nabla_\theta J(\theta)$.

Algorithm: initialize $\theta^{(0)}$; while not converged do $g=\sum_{i=1}^N(\theta^T x^{(i)}-y^{(i)})x^{(i)}$, $\theta\leftarrow\theta-\gamma g$; end; return θ.

Test time: $\hat{y}=h_\theta(x)=\theta^T x$.

Closed-form Solution for Linear Regression

Minimize MSE: $J(\theta)=\frac{1}{2N}\sum_{i=1}^N\left(y^{(i)}-\theta^T x^{(i)}\right)^2=\frac{1}{2N}(X\theta-y)^T(X\theta-y)=\frac{1}{2N}\left(\theta^T X^T X\theta-2\theta^T X^T y+y^T y\right)$.

Set gradient to zero: $\nabla_\theta J(\theta)=\frac{1}{2N}(2X^T X\theta-2X^T y)=0\Longrightarrow X^T X\hat{\theta}=X^T y$. Closed-form solution: $\hat{\theta}=(X^T X)^{-1}X^T y$.

Uniqueness: ,collinearity,

Core formula: $\hat{\theta}=(X^T X)^{-1}X^T y$, valid iff $X^T X$ invertible.

Q1: Invertibility holds when $\text{rank}(X)=D+1$, e.g. $N\gg D+1$ and no feature collinearity (if e.g. $x_3=2x_1+5x_2$, then not).

Complexity: compute $X^T X\in\mathbb{R}^{(D+1)\times(D+1)}$ in $O(ND^2)$; invert it in $O(D^3)$; total $O(ND^2+D^3)$.

closed-form is fast unique but requires full rank; otherwise use GD/SGD or regularization

SGD: Sample index $i\sim\text{Uniform}\{1,2,\ldots,N\}$. Compute gradient $g=\nabla_\theta J^{(i)}(\theta)$( ). Update $\theta\leftarrow\theta-\gamma g$.

Derivative of per-example loss $J^{(i)}(\theta)=\frac{1}{2}(\theta^T x^{(i)}-y^{(i)})^2$:

$\frac{\partial}{\partial\theta_k}J^{(i)}(\theta)=\frac{\partial}{\partial\theta_k}\frac{1}{2}\left(\theta^T x^{(i)}-y^{(i)}\right)^2=\left(\theta^T x^{(i)}-y^{(i)}\right)\frac{\partial}{\partial\theta_k}\left(\theta^T x^{(i)}-y^{(i)}\right)=\left(\theta^T x^{(i)}-y^{(i)}\right)x_k^{(i)}$

Gradient for example i: $\nabla_\theta J^{(i)}(\theta)=(\theta^T x^{(i)}-y^{(i)})x^{(i)}$.

Full-batch gradient of $J(\theta)=\frac{1}{N}\sum_i J^{(i)}(\theta)$: $\nabla_\theta J(\theta)=\frac{1}{N}\sum_{i=1}^N\nabla_\theta J^{(i)}(\theta)=\frac{1}{N}\sum_{i=1}^N(\theta^T x^{(i)}-y^{(i)})x^{(i)}$.

Why SGD Works: Unbiased Gradient Estimation

Let i be sampled uniformly from $\{1,\ldots,N\}$, so $P(i)=1/N$.

ERM objective: $J(\theta)=\frac{1}{N}\sum_{i=1}^N J^{(i)}(\theta)$.

Expectation definition: $E_i[f(i)]=\sum_{i=1}^N P(i)f(i)$.

SGD gradient expectation:

$E_i[\nabla_\theta J^{(i)}(\theta)]=\sum_{i=1}^N\frac{1}{N}\nabla_\theta J^{(i)}(\theta)=\nabla_\theta J(\theta)$.

$\Rightarrow$ single-sample gradient is an unbiased estimator of the full gradient.

Intuition: noisy but in expectation follows the true direction.

L12-13 Neural Network

Perceptron: $a=\boldsymbol{w}^T\boldsymbol{x}+b, z=\phi(a)$.

MLP Forward: for layer l: $a^{(l)}=W^{(l)}z^{(l-1)}+b^{(l)}, z^{(l)}=\phi(a^{(l)})$. Final output $\hat{y}=z^{(L)}$. Loss: $\mathcal{L}(\hat{y},y)$.

δ : define $\delta^{[l]}=\frac{\partial\mathcal{L}}{\partial z^{[l]}}\frac{\partial z^{[l]}}{\partial a^{[l]}}=\frac{\partial\mathcal{L}}{\partial z^{[l]}}\circ\phi'(a^{[l]})$.

For output layer L: $\delta^{[L]}=\frac{\partial\mathcal{L}}{\partial\hat{y}}\circ\phi'(a^{[L]})$

$\delta^{[l-1]}=(W^{[l]})^T\delta^{[l]}\circ\phi'(a^{[l-1]})$.

Gradients: $\frac{\partial\mathcal{L}}{\partial W^{[l]}}=\delta^{[l]}(z^{[l-1]})^T$, $\frac{\partial\mathcal{L}}{\partial b^{[l]}}=\delta^{[l]}$.

Update: $W^{[l]}:=W^{[l]}-\eta\frac{\partial\mathcal{L}}{\partial W^{[l]}}$, $b^{[l]}:=b^{[l]}-\eta\frac{\partial\mathcal{L}}{\partial b^{[l]}}$.

L11 Feature Engineering & Regularization

Learned Embedding: $f_{\text{deep}}(x)=\text{NN}(x)$.

Polynomial Basis: $\phi_k(x)=x^k, k=0,\ldots,m$.

Kernel Trick: $K(x,x')=\langle\phi(x),\phi(x')\rangle$.

Reg Obj: $\min_\theta\mathcal{L}(\theta)+\lambda r(\theta)$.

$\|\theta\|_q=\left(\sum_{m=1}^M|\theta_m|^q\right)^{\frac{1}{q}}$

L2 Ridge: $r(\theta)=\|\theta\|_2^2\Rightarrow\theta^*=(X^T X+\lambda I)^{-1}X^T y$.

L1 Lasso: $r(\theta)=\|\theta\|_1\Rightarrow$sparse θ^*.

Gradient Update: $\theta:=\theta-\eta\left(\nabla\mathcal{L}+\lambda\nabla r\right)$.

CV Tune λ: pick $\lambda=\text{argmin}_\lambda\mathcal{L}_{\text{val}}$.

L14-15 CNN

k: ,p: ,s:

$y_{i,j}^{(c)}=\sum_{u=1}^k\sum_{v=1}^k\sum_{c'=1}^{C_{\text{in}}}W_{u,v}^{(c,c')}x_{i+u,j+v}^{(c')}$.

$H_{\text{out}}=\left\lfloor\frac{H_{\text{in}}+2P-k}{S}\right\rfloor+1, W_{\text{out}}=\left\lfloor\frac{W_{\text{in}}+2P-k}{S}\right\rfloor+1$.

#Params: $k^2C_{\text{in}}C_{\text{out}}+bias(C_{out})$.

Receptive Field: k

Pooling: Strid S

Equivariance: $f(Tx)=Tf(x)$; Invariance via pooling.

ReLU: $f(a)=\max(0,a)$.

Leaky ReLU: $f(a)=\max(\alpha a,a)$.

ELU: $f(a)=\begin{cases}a, & a>0\\ \alpha\left(e^a-1\right), & a\leq 0\end{cases}$.

Sigmoid: $\sigma(a)=\frac{1}{1+e^{-a}}$.

tanh: $\tanh(a)=\frac{e^a-e^{-a}}{e^a+e^{-a}}$.

Maxout: $f(a)=\max_k\left(w_k^\top x+b_k\right)$.

Model Complexity

FLOPs (Conv): $2k^2C_{\text{in}}C_{\text{out}}H_{\text{out}}W_{\text{out}}$.

Same-Pad Rule: keep size when $P=\frac{k-1}{2}$,S=1.

Representative Architectures

Residual Block: $y=F(x,W)+x$.

DenseNet: $x_l=H_l\left([x_0,x_1,\ldots,x_{l-1}]\right)$.

1×1 Conv: feature mixing per pixel, acts like FC layer.

Upsampling & Dilated Conv

Transpose Conv (Size): $H_{\text{out}}=(H_{\text{in}}-1)S-2P+k_{\text{eff}}$.

Dilated kernel: $k_{\text{eff}}=k+(k-1)(d-1)$.

L18 RNN

$h_t=\tanh(W_{xh}x_t+W_{hh}h_{t-1}+b_h)$ ( )

$y_t=W_{hy}h_t+b_y$ ( )

LSTM

$i_t=\sigma\left(W_{xi}x_t+W_{hi}h_{t-1}+b_i\right)$ ( )

$f_t=\sigma\left(W_{xf}x_t+W_{hf}h_{t-1}+b_f\right)$ ( )

$o_t=\sigma\left(W_{xo}x_t+W_{ho}h_{t-1}+b_o\right)$ ( )

$g_t=\tanh\left(W_{xg}x_t+W_{hg}h_{t-1}+b_g\right)$ ( )

$c_t=f_t\odot c_{t-1}+i_t\odot g_t$ ( )

$h_t=o_t\odot\tanh(c_t)$ ( )

$y_t=W_{hy}h_t+b_y$ ( )

L19 Attn $S=\frac{QK^\top}{d_k^{\frac{1}{2}}}$. A=softmax($S+M$) (mask $M=-\infty$ above diag for causal).

$X'=AV$. $Q=XW_q, K=XW_k, V=XW_v$.

$X=[x_1,\ldots,x_T]^\top\in\mathbb{R}^{T\times d_{\text{model}}}$.

Multi-Head Attn (H heads)

Per-Head: $Q^{(i)}=XW_q^{(i)}, K^{(i)}=XW_k^{(i)}, V^{(i)}=XW_v^{(i)}$.

Head Out: $X'^{(i)}=\text{softmax}(\frac{Q^{(i)}K^{(i)\top}}{d_k^{\frac{1}{2}}}+M)V^{(i)}$.

Concat: $X=\text{concat}\left(X'^{(1)},\ldots,X'^{(H)}\right)$.

Pre-Training

Init: start from *random* weights.

Mode A (unsup.): maximise likelihood / reconstr. on huge unlabeled set.

Mode B (sup.): train on huge labeled set (e.g. ImageNet-21k, 14 M imgs).

Vision ex.: autoencoder on MNIST; ImageNet cls. 21 k classes.

Language ex.: The Pile (800 GB), Dolma (3 T tokens).

Fine-Tuning

Init: load *pretrained* weights.

(Opt.) Head: add small randomly-init prediction head.

Train: back-prop on task-specific dataset.

Vision ex.: COCO det. (200 k imgs), ADE20K seg.

NLP ex.: MMLU few-shot (57 tasks); MBPP code-gen .

Recommender Systems & Collaborative Filtering:

Task: predict unknown user–item ratings in a sparse matrix $R \in \mathbb{R}^{m\times n}$; quality often measured by RMSE.

Paradigms: content-based (use side features) vs. collaborative (use interaction data only).

CF families: neighborhood methods (user- / item-based similarity) and latent-factor methods (low-rank matrix factorization).

Matrix Factorization (MF) Core Equations:

Model: $R \approx UV^\top$ with $U \in \mathbb{R}^{m\times k}, V \in \mathbb{R}^{n\times k}$.

Loss : $J = \frac{1}{2}\sum_{(i,j)\in\Omega}(R_{ij} - u_i^\top v_j)^2 + \frac{\lambda}{2}(\|U\|_F^2 + \|V\|_F^2)$.

SGD update: $u_i \leftarrow u_i + \eta(e_{ij}v_j - \lambda u_i), v_j \leftarrow v_j + \eta(e_{ij}u_i - \lambda v_j)$ where $e_{ij} = R_{ij} - u_i^\top v_j$.

ALS: alternately fix V and solve U, then fix U and solve V via least squares.

Variants: SVD for fully observed data, non-negative MF, implicit-feedback MF, bias terms, etc.

Ensemble Learning:

Bagging (parallel bootstrap) and Boosting (sequential reweighting).

Bagging & Random Forests:

Bootstrap Bagging: draw S bootstrap samples, train T base models, aggregate by majority vote(regression compute mean).

Feature Bagging: each learner sees only a random subspace of features.

Random Forest: bootstrap samples *plus* random feature subset at every tree split.

(OOB error: 37 % out-of-bag)

Weighted Majority Algorithm:

Online setting with N experts; start equal weights; prediction by weighted vote; if expert errs, multiply its weight by $\beta \in (0,1)$.

AdaBoost:

Initial distribution $D_1(i) = 1/N$.

At round t: train weak learner h_t w, error ε_t; set $\alpha_t = \frac{1}{2}\ln\frac{1-\varepsilon_t}{\varepsilon_t}$.

Update $D_{t+1}(i) = \dfrac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t}$.

Final classifier $H(x) = \operatorname{sign}\left(\sum_{t=1}^{T}\alpha_t h_t(x)\right)$.

Properties: empirical error drops exponentially; margin theory explains generalization.

Key Comparison:

Bagging —variance reduction, fully parallel, excels with high-variance bases.

Boosting —bias reduction, turns weak into strong, sensitive to noisy labels/outliers.

Random Forest —efficient, parallelizable, offers OOB validation and feature interpretability.

L10 Logistic

$f(x) = \sigma(\theta^\top x + b)$, where $\sigma(z) = \dfrac{1}{1+e^{-z}}$.

$P(y=1|x) = \sigma(\theta^\top x + b)$,

Objective (negative log-likelihood):

$\ell(\theta) = -\frac{1}{N}\log P(y^{(1)},\dots,y^{(N)}|x^{(1)},\dots,x^{(N)},\theta)$

$= -\frac{1}{N}\log\prod_{n=1}^{N} P(y^{(n)}|x^{(n)},\theta)$

$= -\frac{1}{N}\log\prod_{n=1}^{N}\left(P(Y=1|x^{(n)},\theta)\right)^{y^{(n)}}\left(P(Y=0|x^{(n)},\theta)\right)^{1-y^{(n)}}$

$= -\frac{1}{N}\sum_{n=1}^{N} y^{(n)}\log P(Y=1|x^{(n)},\theta) + (1-y^{(n)})\log P(Y=0|x^{(n)},\theta)$

$= -\frac{1}{N}\sum_{n=1}^{N} y^{(n)}\theta^\top x^{(n)} - \log\left(1+e^{\theta^\top x^{(n)}}\right)$.

Gradients:

$J(\theta) = \ell(\theta)$.

$\nabla_\theta J(\theta) = \dfrac{1}{N}\sum_{n=1}^{N} x^{(n)}\left(\hat{y}^{(n)} - y^{(n)}\right), \hat{y}^{(n)} = \sigma(\theta^\top x^{(n)} + b)$.

Weight update: $\theta := \theta - \eta\nabla_\theta J(\theta)$.

Bias gradient / update: $\dfrac{\partial J}{\partial b} = \dfrac{1}{N}\sum_{n=1}^{N}(\hat{y}^{(n)} - y^{(n)}), b := b - \eta\dfrac{\partial J}{\partial b}$.

Bayes Decision Rule:

$\hat{y} = \operatorname{argmax}_{y\in\{0,1\}} P(y|x) = \begin{cases} 1, & \sigma(\theta^\top x + b) \geq 0.5 \\ 0, & \text{otherwise} \end{cases} \iff$

$\begin{cases} 1, & \theta^\top x + b \geq 0 \\ 0, & \text{otherwise.} \end{cases}$

Decision boundary: $\theta^\top x + b = 0$.

Why log-likelihood, not likelihood:

1. Turns products into sums → easier calculus.
2. log is monotone optimum unchanged.
3. Prevents underflow from tiny products.
4. Simplifies gradients, convergence analysis.

Cross-entropy loss (same expression as $\ell(\theta)$) is thus obtained by maximizing the Bernoulli likelihood of the labels.

L21 KMeans

: N , K . : K center

choose the nearest centre: $z^{(i)} = \operatorname{argmin}_j \left\|\mathbf{x}^{(i)} - \mathbf{c}_j\right\|_2^2$.

$\hat{\mathbf{C}} = \operatorname{argmin}_{\mathbf{C}}\sum_{i=1}^{N}\min_j \left\|\mathbf{x}^{(i)} - \mathbf{c}_j\right\|_2^2 = \operatorname{argmin}_{\mathbf{C},\mathbf{z}}\sum_{i=1}^{N}\left\|\mathbf{x}^{(i)} - \mathbf{c}_{z^{(i)}}\right\|_2^2 = \operatorname{argmin}_{\mathbf{C},\mathbf{z}} J(\mathbf{C},\mathbf{z})$.

$J(\{C_k\}_{k=1}^K) = \sum_{k=1}^{K}\sum_{\mathbf{x}\in C_k}\left\|\mathbf{x} - \boldsymbol{\mu}_k\right\|_2^2$, $\boldsymbol{\mu}_k = \frac{1}{|C_k|}\sum_{\mathbf{x}\in C_k}\mathbf{x}$.

Lloyd : repeat until converge:

a. Assignment step: $z^{(i)} \leftarrow \operatorname{argmin}_j \left\|\mathbf{x}^{(i)} - \mathbf{c}_j\right\|_2^2$, $i = 1,\dots,N$.

b. Update step: $\mathbf{c}_j \leftarrow \frac{1}{|C_j|}\sum_{\mathbf{x}^{(i)}; z^{(i)}=j}\mathbf{x}^{(i)}$, $j = 1,\dots,K$,

FPH: $D(x) = \min_{c\in C}\|x - c\|_2$, Select the $c_k = \operatorname{argmax}_{x\in X} D(x)$ for next center until have chosen K centers

K-means++: $P(\mathbf{x}) = \dfrac{D(\mathbf{x})^2}{\sum_{\mathbf{x}'} D(\mathbf{x}')^2}$ center

:$\mathbb{E}[\text{Cost}_{\text{K++}}] \leq O(\log K)\cdot\text{OPT}$

L22 EM+GMM

Parameter Estimation with Latent Variables:

$\log p(\mathbf{X}|\Theta) = \log\sum_z p(\mathbf{X},z|\Theta) = \log\sum_z q(z)\frac{p(\mathbf{X},z|\Theta)}{q(z)}$

$\geq \sum_z q(z)\log\frac{p(\mathbf{X},z|\Theta)}{q(z)}$ (concave f, Jensen: $f(\sum\lambda_i x_i \sum\lambda_i f(x_i))$

$\log p(\mathbf{X}|\Theta) \geq \sum_z q(z)\log p(\mathbf{X},z|\Theta) - \sum_z q(z)\log q(z) = \sum_z q(z)\log p(\mathbf{X},z|\Theta) + \text{const.}$ $\left(\text{term independent of }\Theta\right)$

If we set $q(z) = p(z|\mathbf{X},\Theta)$, the inequality becomes equality $\sum_z q(z)\log\frac{p(\mathbf{X},z|\Theta)}{q(z)} = \sum_z p(z|\mathbf{X},\Theta)\log\frac{p(z|\mathbf{X},\Theta)p(\mathbf{X}|\Theta)}{p(z|\mathbf{X},\Theta)} = \sum_z p(z|\mathbf{X},\Theta)\log p(\mathbf{X}|\Theta) = \log p(\mathbf{X}|\Theta)$ Thus for $q(z) = p(z|\mathbf{X},\Theta)$ we have

$\log p(\mathbf{X}|\Theta) = \sum_z p(z|\mathbf{X},\Theta)\log p(\mathbf{X},z|\Theta) + \text{const.} = \mathbb{E}_{p(z|\mathbf{X},\Theta)}\left[\log p(\mathbf{X},z|\Theta)\right] + \text{const.}$ Therefore $\log p(\mathbf{X}|\Theta)$ is tightly lower-bounded by $\mathbb{E}\left[\log p(\mathbf{X},z|\Theta)\right]$, which the EM algorithm maximizes.

Expectation Maximization (EM) Algorithm:

E (Expectation) step:

Compute the posterior $p(\mathbf{Z}|\mathbf{X},\Theta^{\text{old}})$ over latent variables $\mathbf{Z}$ using Θ^{old}. Compute the expected complete-data log-likelihood with respect to this posterior:

$Q(\Theta,\Theta^{\text{old}}) = \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\Theta^{\text{old}})}\left[\log p(\mathbf{X},\mathbf{Z}|\Theta)\right] = \sum_{\mathbf{z}} p(\mathbf{Z}|\mathbf{X},\Theta^{\text{old}})\log p(\mathbf{X},\mathbf{Z}|\Theta)$.

M (Maximization) step:

Maximize Q with respect to Θ.

For maximum-likelihood estimation (MLE): $\Theta^{\text{new}} = \operatorname{argmax}_\Theta Q(\Theta,\Theta^{\text{old}})$.

For maximum-a-posteriori (MAP) estimation: $\Theta^{\text{new}} = \operatorname{argmax}_\Theta\left\{Q(\Theta,\Theta^{\text{old}}) + \log p(\Theta)\right\}$.

If the log-likelihood or the parameter values have not converged, set $\Theta^{\text{old}} = \Theta^{\text{new}}$ and return to the E step.

The algorithm converges to a local maximum of $p(\mathbf{X}|\Theta)$.

EM for Gaussian Mixture Model (GMM):

Initialize parameters $\Theta = \{\pi_k,\mu_k,\Sigma_k\}_{k=1}^K$

Iterate until $\log p(\mathbf{X}|\Theta)$ convergence :

E-step:

$\gamma_{ik} = p(z_i = k|x_i,\Theta^{\text{old}}) = \dfrac{\pi_k^{\text{old}}\mathcal{N}(x_i|\mu_k^{\text{old}},\Sigma_k^{\text{old}})}{\sum_{j=1}^{K}\pi_j^{\text{old}}\mathcal{N}(x_i|\mu_j^{\text{old}},\Sigma_j^{\text{old}})}$. Expected complete-data log-likelihood:

$Q(\Theta,\Theta^{\text{old}}) = \mathbb{E}_{Z|X,\Theta^{\text{old}}}[\log p(\mathbf{X},\mathbf{Z}|\Theta)] = \sum_{i=1}^{N}\sum_{k=1}^{K}\gamma_{ik}\left(\log\pi_k + \log\mathcal{N}(x_i|\mu_k,\Sigma_k)\right)$.

M-step: maximize Q with respect to Θ: (with $\sum_k\pi_k = 1$):

$\pi_k^{\text{new}} = \dfrac{1}{N}\sum_{i=1}^{N}\gamma_{ik}$. $\mu_k^{\text{new}} = \dfrac{\sum_{i=1}^{N}\gamma_{ik}x_i}{\sum_{i=1}^{N}\gamma_{ik}}$.

$\Sigma_k^{\text{new}} = \dfrac{\sum_{i=1}^{N}\gamma_{ik}(x_i - \mu_k^{\text{new}})(x_i - \mu_k^{\text{new}})^\top}{\sum_{i=1}^{N}\gamma_{ik}}$.

$p(\mathbf{X},\mathbf{Z}|\Theta) = \prod_{i=1}^{N}\pi_{z_i}\mathcal{N}(x_i|\mu_{z_i},\Sigma_{z_i})$.

$\log p(\mathbf{X},\mathbf{Z}|\Theta) = \sum_{i=1}^{N}\left(\log\pi_{z_i} + \log\mathcal{N}(x_i|\mu_{z_i},\Sigma_{z_i})\right)$.

Gaussian PDF in D dimensions: $\mathcal{N}(x|\mu_k,\Sigma_k) = (2\pi)^{-D/2}|\Sigma_k|^{-1/2}\exp\left(-\frac{1}{2}(x-\mu_k)^\top\Sigma_k^{-1}(x-\mu_k)\right)$.

L23 PCA: Data Centering: Mean vector $\mu=\frac{1}{N}\sum_{n=1}^{N}\mathbf{x}^{(n)}$.

Centered samples $\tilde{\mathbf{x}}^{(n)}=\mathbf{x}^{(n)}-\mu\ \forall n$.

Stacked matrix $X=\left[\tilde{\mathbf{x}}^{(1)^T}\tilde{\mathbf{x}}^{(2)^T}\ldots\tilde{\mathbf{x}}^{(N)^T}\right]^T\in\mathbb{R}^{N\times D}$.

Sample Variance & Covariance:

One-dimensional variance $\hat{\sigma}^2=\frac{1}{N}\sum_{i=1}^{N}\left(x^{(i)}-\hat{\mu}\right)^2$.

$\Sigma_{jk}=\frac{1}{N}\sum_{i=1}^{N}\left(x_j^{(i)}-\hat{\mu}_j\right)\left(x_k^{(i)}-\hat{\mu}_k\right)$.

Matrix form $\Sigma=\frac{1}{N}X^TX$.

Projection & Reconstruction Error:

Projection of $\tilde{\mathbf{x}}^{(n)}$ onto unit vector $\mathbf{v}$: $z^{(n)}=\mathbf{v}^T\tilde{\mathbf{x}}^{(n)}$.

Reconstruction error summed over data:
$\sum_{n=1}^{N}\left\|\tilde{\mathbf{x}}^{(n)}-(\mathbf{v}^T\tilde{\mathbf{x}}^{(n)})\mathbf{v}\right\|_2^2=\sum_{n=1}^{N}\left\|\tilde{\mathbf{x}}^{(n)}\right\|_2^2-\sum_{n=1}^{N}\left(\mathbf{v}^T\tilde{\mathbf{x}}^{(n)}\right)^2$.

Minimizing the error $\Longleftrightarrow$ maximizing the second term (projected variance).

Variance Maximization Form:

$\hat{\mathbf{v}}=\text{argmax}_{\mathbf{v}:\|\mathbf{v}\|_2^2=1}\sum_{n=1}^{N}\left(\mathbf{v}^T\tilde{\mathbf{x}}^{(n)}\right)^2=$

$\text{argmax}_{\mathbf{v}:\|\mathbf{v}\|_2^2=1}\mathbf{v}^T\left(\sum_{n=1}^{N}\tilde{\mathbf{x}}^{(n)}\tilde{\mathbf{x}}^{(n)^T}\right)\mathbf{v}=$

$\text{argmax}_{\mathbf{v}:\|\mathbf{v}\|_2^2=1}\mathbf{v}^T(X^TX)\mathbf{v}$.

Eigenvalue Problem:

Lagrangian $\mathcal{L}(\mathbf{v},\lambda)=\mathbf{v}^T(X^TX)\mathbf{v}-\lambda(\|\mathbf{v}\|_2^2-1)$.

Gradient condition $2(X^TX)\hat{\mathbf{v}}-2\lambda\hat{\mathbf{v}}=0\Rightarrow(X^TX)\hat{\mathbf{v}}=\lambda\hat{\mathbf{v}}$.

Thus $\hat{\mathbf{v}}$ is an eigenvector of X^TX (or Σ) and λ is the associated eigenvalue (variance along that component).

Principal components are the eigenvectors ordered by decreasing λ; λ_i quantifies variance captured by component $\hat{\mathbf{v}}_i$.

PCA Algorithm:

1. Center data $\mathbf{X}\rightarrow X$.
2. Compute covariance matrix $\Sigma=\frac{1}{N}X^TX$.
3. Eigendecompose $\Sigma\Rightarrow\{(\hat{\mathbf{v}}_i,\lambda_i)\}$.
4. Select top K eigenvectors $V_K=[\hat{\mathbf{v}}_1,\ldots,\hat{\mathbf{v}}_K]$.
5. Low-dim projection $\mathbf{z}^{(n)}=V_K^T\tilde{\mathbf{x}}^{(n)}$.

Applications: visualization, noise reduction, improved generalization in downstream tasks.

$\min_{w,b}\frac{1}{2}\|w\|^2$s.t. $y_i(w^\top x_i+b)\geq 1$.

Lagrange Dual (Support Vectors emerge):

Lagrangian
$\mathcal{L}(w,b,\alpha)=\frac{1}{2}\|w\|^2-\sum_{i=1}^{m}\alpha_i\left[y_i(w^\top x_i+b)-1\right],\alpha_i\geq 0$.

Stationarity
$w=\sum_i\alpha_i y_i x_i,\sum_i\alpha_i y_i=0$.

Dual
$\max_\alpha\sum_i\alpha_i-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j x_i^\top x_j,\alpha_i\geq 0,\sum_i\alpha_i y_i=0$.

KKT
$\alpha_i\left[y_i(w^\top x_i+b)-1\right]=0$. Only $\alpha_i>0$ are *support vectors*.

Soft-Margin SVM (Hinge Loss):

Primal $\min_{w,b,\xi}\frac{1}{2}\|w\|^2+C\sum_{i=1}^{m}\xi_i,y_i(w^\top x_i+b)\geq 1-\xi_i,\xi_i\geq 0$.

Hinge loss $\ell=\max\{0,1-y_i(w^\top x_i+b)\}$.

Dual
$\max_\alpha\sum_i\alpha_i-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j x_i^\top x_j,0\leq\alpha_i\leq C,\sum_i\alpha_i y_i=0$.

SV classes: $0<\alpha_i<C$ (on/inside margin), $\alpha_i=C$ (errors).

Kernel Trick –Non-linear SVM:

Replace inner product: $x_i^\top x_j\rightarrow K(x_i,x_j)$.

Dual
$\max_\alpha\sum_i\alpha_i-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j K(x_i,x_j),0\leq\alpha_i\leq C,\sum_i\alpha_i y_i=0$.

Classifier $f(x)=\text{sign}\left(\sum_i\alpha_i y_i K(x_i,x)+b\right)$.