Gaussian Distribution - MLE

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

(Biased) $$\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{\mu})^2$$

(Unbiased) $$\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \hat{\mu})^2$$

$$p_x(k) = p(x \mid y = k)\, p_k = \pi_k \frac{1}{\sqrt{2\pi}\,\sigma_k} e^{-\frac{1}{2}\left(\frac{x-\mu_k}{\sigma}\right)^2}$$

Discriminant Classifiers

LDA - equal covariance between classes

$$\delta_k(x) = x\frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log \pi_k$$

QDA - different covariances

$$\delta_k(x) = -\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1}(x-\mu_k) - \frac{1}{2}\log|\Sigma_k| + \log \pi_k$$

Quadratic decision boundary

Higher variance, lower bias than LDA

Naïve Bayes - same equation as QDA but features are independent (covariance matrix is diagonal)

Classification Performance Evaluation

		Predicted class −	Predicted class +
True Class	−	TN	FP
True Class	+	FN	TP

$$\boxed{\begin{aligned} \text{Prec.} &= \frac{TP}{TP+FP} \\ \text{Rec.} &= \frac{TP}{TP+FN} \\ \text{F-Score} &= \frac{2PR}{P+R} \\ \text{FPR} &= \frac{FP}{FP+TN} \end{aligned}}$$

* "Positive" class is rare one

ROC curve shows FPR vs. TPR - want high TPR, low FPR

Decision Trees

$$RSS = \sum_{j=1}^{J}\sum_{i \in R_j}(y_i - \hat{y}_{R_j})^2$$

$\hat{y}_{R_j}$ is mean response of training obs. in region J

y_i is response of i^{th} training example

Recursive binary splitting: select predictor X_j and cutpoint s such that splitting the space into regions with s leads to the largest reduction in the RSS

→ Find feature X_j and s that min.

$$\sum_{i: x_i \in R_1(j,s)}(y_i - \hat{y}_{R_1})^2 + \sum_{i: x_i \in R_2(j,s)}(y_i - \hat{y}_{R_2})^2$$

Pre-pruning: set stopping criteria to prevent overfitting

Post-pruning (pruning): grow long tree using training data, then prune it to obtain subtree

$\hat{p}_{mk}$: proportion of training obs. in m^{th} region from k^{th} class

Classification error rate:

$$E = 1 - \max_k(\hat{p}_{mk})$$

Gini index:

$$G = \sum_{k=1}^{K}\hat{p}_{mk}(1-\hat{p}_{mk})$$

"Pure nodes" → G = 0

Cross Entropy:

$$D = -\sum_{k=1}^{K}\hat{p}_{mk}\log \hat{p}_{mk}$$

Ensemble Methods

Bagging (bootstrap): create multiple training sets and train base algorithms on different sets, then get overall response using all

Random forests: build decision trees on bootstrapped ex., randomly select subset of features, then find best for splitting

Boosting: models are grown sequentially - focus on training examples that previous models got wrong

AdaBoost: loop over classifiers

$$err_m = \sum_{i=1}^{n} w_i I(y_i \neq G_m(x_i))$$

$$\alpha_m = 0.5 \ln\left(\frac{1-err_m}{err_m}\right)$$

$$w_i \longleftarrow w_i e^{-\alpha_m y_i \hat{y}_i}$$

Divide by $\sum_{i=1}^{n} w_i$ to normalize

Stacking: different base methods

SVC & SVM

Find hyperplane that best separates the classes

MMC - maximize gap between (linearly sep.) classes

support vectors are on the margin

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

SVC generalizes (soft margin)

$y_i(w^T x_i + b) \geq 1 - \xi_i$

support vectors lie on or within margin

SVC's decision is determined by support vectors

$$f(x) = b + \sum_{j=1}^{s}\alpha_{t_j} y_{t_j} \langle x_{t_j}, x \rangle$$

SVM uses kernel function to get nonlinear decision boundary (change all inner product to kernel function)

Kernel types:

linear (same as linear SVC)

Polynomial (degree > 1)

RBF (Gaussian similarity measure)

$$K(x_i, x_{i'}) = \exp\left(-\gamma \sum_{j=1}^{p}(x_{ij} - x_{i'j})^2\right)$$

Neural Networks

Activation functions:

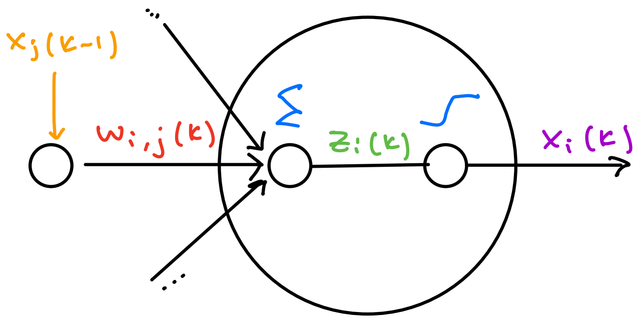
$$\sigma(v) = \frac{1}{1+e^{-v}}$$

$$\tanh(v) = \frac{2}{1+e^{-2v}} - 1$$

$$\text{relu}(v) = \max(0, v)$$

Softmax function: (done at output)

$$g_k(T) = \frac{e^{T_k}}{\sum_{\ell=1}^{K} e^{T_\ell}}$$



$$w_{i,j}(k) := w_{i,j}(k) - \alpha \frac{\partial J}{\partial w_{i,j}(k)}$$

$$w_{i,j}(k) - \alpha\, \delta_i(k)\, x_j(k-1)$$

$$\delta_i(k) = x_i(k)(1 - x_i(k)) \sum_{i=1}^{n} \delta_i(k-1)$$

at output: $\frac{\partial J}{\partial \hat{y}} = -1(y_d - \hat{y})$, can use to get δ_s for previous layer

Regularization - shrink coefficients

Data augmentation - train model on noisy versions of input

Dropout - randomly setting to 0 the output of some neurons during each training iteration

Deep Learning

Feature extraction → output pred.

CNNs: use convolution, which extracts low-level features at first and increases

Sparse connectivity: kernel is small

Output dimension = $\frac{(\text{input} - \text{kernel size})}{\text{stride}} + 1$

RNNs: process sequential data

$h(t) = f(h(t-1), x(t))$

Sequence-to-sequence (word by word translation)

Sequence-to-vector (predict next word)

Vector-to-sequence (image captioning)

Sequence-to-sequence, different length (translate sentence)

Hard to capture long-term dependencies - gradient will vanish or explode

LSTM: try to learn what info. to keep in memory and what to discard

Autoencoder: final output = input

Dimensionality Reduction

Preprocessing for unsupervised learning

PCA: how many features are needed to explain variability in data

Find eigenvalues/eigenvectors of covariance matrix

$$\underbrace{[z_{i1}, z_{i2}, \ldots, z_{im}]}_{n \times M} = \underbrace{[x_{i1}, x_{i2}, \ldots, x_{im}]}_{n \times p} \cdot \underbrace{W}_{p \times M}$$

LDA for dimensionality reduction: max. component axes for class separation

Clustering

k-means:

Initialize randomly

(E) Assign point to cluster based on centroid

(M) Re-evaluate centroids

Agglomerative

merge least dissimilar at each step

dendrogram - height represents dissimilarity of merged clusters

Linkage: measure dissimilarity

Single: smallest b/t clusters (PW)

Complete: largest b/t clusters (PW)

Average: average b/t clusters (PW)

Centroid: dist b/t centroids

DBSCAN

If there are min_samples within distance eps, they are in same cluster & point is a core point

Mixture of Gaussians

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} \overbrace{\gamma(z_{nk})}^{\text{prob. it lies in class } k} x_n$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu_k)(x_n - \mu_k)^T$$

$$\pi_k = \frac{N_k}{N}$$

$$N_k = \sum_{i=1}^{n} \gamma(z_{nk})$$