Graphical Representation of Numerical Data

Kernel density estimator:

$$\hat{f}_h(x) = \frac{1}{2nh}\sum_{i=1}^{n} I(|x-x_i| \le h) \quad \text{(Histogram)}$$

$$\Rightarrow \hat{f}_h(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x-x_i}{h}\right)$$

with Kernel $K(u) = I(|u| \le 0.5)$ (uniform Kernel), h...Degree of smoothness

Decision Trees

Impurity functions $u(R)$: $p_k(R) = \frac{1}{n}\sum I(y_i = k)$ (purity)

- classification error: $1 - \max_k p_k(R)$
- Gini index: $1 - \sum_k p_k(R)^2$
- entropy: $-\sum_k p_k(R) \cdot \log(p_k(R))$

⇒ are all maximized when p_k is uniform on the K classes in R; all are minimized when $p_k = 1$ for some k (R has one class)

Growing a classification tree: $R \to R^+$ and R^-

- calculate $u(R)$, $u(R^+)$, $u(R^-)$
- Gini improvement: $u(R) - (p(R^-)\cdot u(R^-) + p(R^+)\cdot u(R^+)) \to \max$

⇒ reduces uncertainty

Growing a regression tree:

$$\hat{y} = \sum_{m=1}^{M} c_m I(x \in R_m)$$

$$\Rightarrow \hat{c}(R_m) = \frac{1}{n(R_m)}\sum_{i=1}^{n} I(y_i \mid x_i \in R_m)$$

Stopping parameters:

- #splits, #observations per region, Δ objective function
- tree prunning: $R_\alpha(T) = \frac{1}{\sum(y_i-\bar{y})^2}\sum_{m=1}^{|T|}\sum_{i:x_i\in R_m}(y_i - f(x_i))^2 + \alpha|T| \to \min$
- α...complexity parameter (CP)

Optimal subtree with CV:

- trees T_0 (0 splits), ..., T_m (m splits) ⇒ $\infty, \alpha_1, ..., \alpha_{min}$
- $\beta_i = \sqrt{\alpha_i \cdot \alpha_{i+1}}$ (average)
- subsets $G_1, ..., G_B$
- trees with $\beta_1, ..., \beta_m$ for each subset (leave one out → forecast)

⇒ smallest β over all G_i's

Bagging: bootstrapping from training data, tree T_i

- prediction: $\frac{1}{n}\sum_{i=1}^{n} \text{pred}(T_i, x)$
- when correlation in bootstrap samples → effect decreases

Random Forests: bootstrapping + subset of explanatory variables

- importance of variable: how much increase of MSE or classification error when variable is permuted in left-out-sample

Boosting: AdaBoost:

- bootstrapping from training data with distribution w_t ($w_0 = \frac{1}{n}$)
- train classifier f_t
- $\varepsilon_t = \sum w_t \cdot I(y_t \ne f_t(x_i))$, $\alpha_t = \frac{1}{2}\log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$
- scale $w_{t+1}(i) = w_t(i)\cdot\exp(-\alpha_t y_i f(x_i))$ and normalize
- training error $\frac{1}{n}\sum I(y_i \ne f_{boost}(x_i)) \le \exp\left(-2\sum\left(\frac{1}{2}-\varepsilon_t\right)^2\right)$

⇒ $f_{boost} = \text{sgn}\left(\sum \alpha_t f_t\right)$

CHAID: not binary, split so that ANOVA has smallest p-value

Nearest Neighbors Classifiers

NN classifier: label x with label of closest point (default euclidean distance)

k-NN classifier: majority vote of k closest points

Linear classifier and Perceptron

Hyperplane H: $p-1$ dimensional subspace $H = \{x \mid \langle x, w\rangle = 0\}$

- affine Hyperplane $H = \{x \mid \langle x, w\rangle + w_0 = 0\}$

⇒ linear classifier: $\text{sgn}(\langle x, w\rangle + w_0)$

Perceptron:

- $L = -\sum(y_i \cdot \langle x_i, w\rangle) I(y_i \ne \text{sgn}(\langle x, w\rangle)) \to \min$
- $\frac{\partial L}{\partial w}$ direction in which L is increasing ⇒ $w' = w - \eta\frac{\partial L}{\partial w}$ (gradient descent, Perceptron uses a stochastic version): find one $(x_i y_i)$ where $y_i \ne \text{sgn}(\langle x_i, w\rangle)$ and then $w_{t+1} = w_t + \eta y_i x_i$
- problems: data separable → one H will be found (not necessary the best), data not separable → algorithm won't stop

Maximum Margin Classifiers and SVM

Maximum margin classifier:

- margin: distance $H \leftrightarrow$ closest point → max
- convex hull: every point can be reached by linear combination of convex-hull-points
- How to find H?

→ $\langle x_i, w\rangle + w_0 \ge 1$ ($y_i = 1$), $\langle x_i, w\rangle + w_0 \le -1$ ($y_i = -1$) ⇒ $y_i(\langle x_i, w\rangle + w_0) - 1 \ge 0$

→ $d_+ = d_- = \frac{1}{\|w\|} \Rightarrow d_+ + d_- = \frac{2}{\|w\|} \to \max$

→ primal problem: $L = \frac{1}{2}\|w\|^2 \to \min$ s.t. $y_i(\langle x_i, w\rangle + w_0) \ge 1$ (with Lagrange coefficients α)

→ dual problem $L = \sum\alpha_i - \frac{1}{2}\sum_i\sum_j \alpha_i\alpha_j y_i y_j\langle x_i, x_j\rangle \to \max$ s.t. $\sum\alpha_i y_i = 0$ (find closest points in convex hull)

→ solve dual problem for α, $w = \sum\alpha_i y_i x_i$, pick i for $\alpha_i > 0$ and solve $y_i(\langle x_i, w\rangle + w_0) \ge 1$ for w_0

→ problems: robustness, if not linear separable → not working

Support Vector Classifier:

- slack variables ξ_i represent violation from strict separation, $y_i(\langle x_i, w\rangle + w_0) \ge 1 - \xi_i$
- $L = \frac{1}{2}\|w\|^2 + \lambda\sum\xi_i$ s.t. $y_i(...) \ge 1 - \xi_i$
- problems: non separability

Support Vector Machine: maps data in higher dimensions with $\phi(x)$

- primal problem: SVC with $x \to \phi(x)$
- dual problem: MMC with $x \to \phi(x)$ s.t. $0 \le \alpha_i \le \lambda$, $\sum\alpha_i y_i = 0$

⇒ if K exists such that $K(x_i, x_j) = \langle\phi(x_i), \phi(x_j)\rangle$ we can use SVM without knowing ϕ (kernel trick)

- $K(x_i, x_j)$ pos. def. ⇔ $\sum\sum\lambda_i\lambda_j K(x_i, x_j) > 0$
  - linear kernel ($\langle x_i, x_j\rangle$)
  - polynomial kernel ($(\langle x_i, x_j\rangle + 1)^p$)
  - radial kernel ($\exp(-\|x_i - x_j\|/2\sigma^2)$)
  - sigmoid kernel ($\tanh(k\langle x_i, x_j\rangle - \delta)$)
- SVM with K classes:
  - 1-vs-1: $\binom{K}{2}$ pairs, $\binom{K}{2}$ classifiers ⇒ majority vote
  - 1-vs-all: compare one of K classes to rest, assign x^* to $b_k + w_{1k}x_1^* + ... \to \max$

K-Means

K-Means:

- objective function: $\hat{\mu}, \hat{c} = \arg\min L = \sum_k\sum_{i:c_i=k}\|x_i - \mu_k\|^2$
- non convex objective function (no global optimum findable, no derivatives, no gradient descent)
- update classes, update centers until nothing changes (start with random centers) → run multiple times

K-Means in compression: similar colors → same color (cluster)

choose K: advanced knowledge, increasing K leads to more relative reduction of L when K is to small than K is to big

Extensions:

- K-mediods: use L_1 norm instead of L_2 norm → more robust to outliers
- weighted K-means: $L = \sum_i\sum_k \phi_i(k)\frac{\|x_i-\mu_k\|^2}{\beta}$ where $\phi_i(k) > 0$ and $\sum_k\phi_i(k) = 1$, $\beta > 0$

$$\phi_i(k) = \frac{\exp\left(-\frac{1}{\beta}\|x_i - \mu_k\|^2\right)}{\sum_k \exp\left(-\frac{1}{\beta}\|x_i - \mu_k\|^2\right)}$$

$$\mu_k = \frac{\sum_i x_i\phi_i(k)}{\sum_i \phi(k)}$$

Generalized mixture model building:

EM-Algorithm

- each cluster contains points from normal distribution with μ_i, Σ_i (different sizes of clusters possible)
- E-Step: update $\phi_i(k)$ (to which of the K normal distributions a point belongs)
- M-Step: update parameters of each normal distribution

Cluster Analysis

Proximity for non-metric data, distances for metric data

- binary data: $\frac{a_1+\delta a_4}{a_1+\delta a_4+\lambda(a_2+a_3)}$ → matching coefficient $\lambda = \delta = 1$
- mixed scales:
  - nominal/ordinal: $d_{ij}^{(k)} = I(x_{ik} \ne x_{jk})$
  - metric: $d_{ij}^{(k)} = \frac{|x_{ik}-x_{jk}|}{\max x_{mk} - \min x_{mk}}$
  - $\delta_{ij}^k = 0$ if missing, else 1

  ⇒ Gower coefficient: $d_{ij} = \frac{\sum w_k\delta_{ij}^{(k)}d_{ij}^{(k)}}{\sum w_k\delta_{ij}^{(k)}}$
- L_p norm
- French railway metric: over Paris
- Karlsruhe metric: along arcs
- Mahalanobis distance: $d_{ij}^2 = (x_i - x_j)^\top A(x_i - x_j)$
- Contingency tables: $d^2(r_1, r_2) = \sum\left(\frac{x_{..}}{x_{\cdot j}}\right)\left(\frac{x_{r_1 j}}{x_{r_1\cdot}} - \frac{x_{r_2 j}}{x_{r_2\cdot}}\right)^2$
- Q-Correlation distance: correlation between x_i and x_j in k-th variable

Distance between clusters:

- single linkage: nearest points → large groups
- complete linkage: farthest points
- average linkage: mean of all combinations
- centroid: $d(R, \text{center of gravity}(P+Q))$
- Ward: join groups that not increase heterogeneity to much ($I(R) = \frac{1}{n_R}\sum d^2(x_i, \bar{x}_R)$)

hierarchical: joins/splits groups, partitioning: exchange elements in given clustering

Missing data

Types of missing data:

- missing completely at random
- missing at random: depends on observed data
- missing not at random: depends on observed predictors or on missing value itself

$\Rightarrow$ not testable!

What to do?

- deletion: assumes MCAR
- pairwise deletion: assumes MCAR ("merges" rows to get full data)
- unconditional location (cold deck): use value from "closest" observation $\rightarrow$ mean of Y is wrong, variance of Y is wrong
- unconditional mean: mean of all other observations $\rightarrow$ mean of Y is good, variance of Y is wrong
- unconditional distribution (hot deck): use randomly selected observation $\rightarrow$ mean/variance of Y is good, $\mathrm{Cor}(X, Y)$ is wrong
- conditional mean (linear regression) $\rightarrow$ conditional mean of Y is good, $\mathrm{Cor}(X, Y)$ is good, conditional variance of Y is wrong
- conditional distribution (linear regression + ε) $\rightarrow$ conditional mean/variance of Y is good, $\mathrm{Cor}(X, Y)$ is good
- random Forests: good if MAR
- time series: last observation carried forward, next observation carried backward
- time series: interpolation linear, seasonal interpolation

$\Rightarrow$ multiple imputations: impute, estimate parameter, ...

Markov Chains

- M_{ij}...probability of going from state i to state $j \Rightarrow$ row sums $= 1$
- estimate M: $\hat{M}_{ij} = \frac{\#\text{transitions } i \to j}{\#\text{transitions } i \to *}$
- $w_{t+1} = w_t M$, stationary $w_\infty = w_\infty M$ $\xrightarrow[\text{Eigenvalue}]{\lambda=1} w_\infty = \frac{\gamma}{\|\gamma\|_1}$
- Markov Chains as ranking: A beats B, then B $\rightarrow$ A high and A $\rightarrow$ B low
- Markov Chains as classification: $\hat{M}_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{b}\right) \rightarrow$ normalize, if x_i has label: $M_{ii} = 1$, rest of row $0 \Rightarrow$ absorbing state

$$M = \begin{pmatrix} A & B \\ 0 & I \end{pmatrix} \Rightarrow w_\infty = w_0 M^\infty$$
$$= w_0 \begin{pmatrix} 0 & (I - A)^{-1} B \\ 0 & I \end{pmatrix}$$

- hidden Markov Model: hidden sequence of states, observation is drawn from distribution associated with state $\rightarrow$ EM-algorithm

Neural Networks

Activation functions:

$$f(I_j) = \begin{cases} 1 & I_j \geq \theta_j \\ 0 & \text{else} \end{cases}$$
$$f(I_j) = \frac{1}{1 + \exp(-\beta(I_j - \theta_j))} \quad \text{(sigmoid)}$$
$$f(I_j) = \max(0, I_j) \quad \text{(relu)}$$
$$f(I_j) = \log(1 + \exp(I_j)) \quad \text{(solftplus)}$$
$$S(y_j) = \frac{\exp(y_i)}{\sum \exp(y_i)} \quad \text{(softmax)}$$

Back-propagation: update weights with gradient decent

Reinforcement learning: find best policy to max rewards $\rightarrow$ Q-learning: $Q(s, a) = r + \gamma \max_a Q(s', a)$