Histograms and Density Trees

- categorization of classification alg.
  - generative vs. discriminative (learn. RHS vs. CRS of Bayes rule)
  - parametric vs. non-parametric models
    - parametric: model uses an analytic prob. distrib (Gaussian, Beta, Exponential) ⇒ optimize the parameters (e.g. mean, variance, co-variance)
    - non-parametric: no fixed analytic formula for prob. distr. ⇒ more parameters

	generative	discriminative
parametric	QDA, LDA (Gauss)	LR (logistic distribution, O(D) storage)
non-parametric	Histograms ($D=1$) Density Trees ($D>1$)	NN, k-NN (parameters: memorized TS, $O(N \cdot D)$ storage) Decision Trees

- Histogram: finite number of bins, for each bin ⇒ return constant prob. (learn)
- if $X \in \{1, \dots, M\}$ discrete with M categories: bins ≙ categories, prob $\hat{p}_m = \frac{N_m}{N}$, $m = 1, \dots, M$
- if $X \in \mathbb{R}$: we must discretize X and then count instances per bin

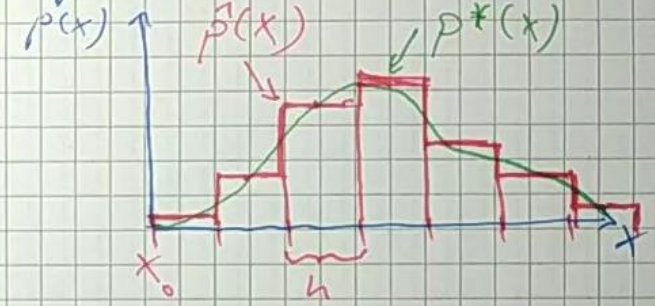
  $m = 1, \dots M$: $\text{bin}_m = \{ X \mid X_0 + m \cdot \Delta X \leq X < X_0 + (m+1) \Delta X \}$

  split X axis into intervals of size $\Delta X := h$

  ⇒ calculate bin index of instance X: $m = \left\lfloor \frac{x - x_0}{h} \right\rfloor$ $\quad \lfloor \cdot \rfloor$ ≙ "floor function" ≙ round down

- property of non-parametric methods: two types of parameters
  - learnable parameters ≙ fit by the training alg, here: $\hat{p}_m$ probs in each bin
  - hyper-parameters ≙ fixed by the data analysist before hand, or optimized via cross-validation (trial-and-error), here: x_0, h
    (x_0: left boundary of 1st bin, h: bin width)

⇒ result: true prob. density is approximated by a piecewise constant function



- given fixed values for the hyper-parameters, how do we learn optimal $\hat{p}_m$ for each bin?
  - express $\hat{p}(x)$ by a single formula:

$$\hat{p}(x) = \sum_{m=1}^{M} \hat{p}_m \, \mathbb{1}[x \in \text{bin}_m]$$

$\mathbb{1}[\text{bin}_m]$ ≙ indicator function $\quad \mathbb{1}[\text{cond.}] = \begin{cases} 1 & \text{if cond = true} \\ 0 & \text{if cond = false} \end{cases}$

- minimize least-squares error of $\hat{p}(x)$

$$\{\hat{p}_m\}_{m=1}^{M} = \arg\min_{\{p_m\}} \int \left[p^*(x) - \hat{p}(x) \right]^2 dx$$

$$= \arg\min_{\{p_m\}} \left[\underbrace{\int p^*(x)^2 \, dx}_{\text{independent of } \hat{p}} - 2 \underbrace{\int p^*(x) \cdot \hat{p}(x) \, dx}_{} + \int \hat{p}(x)^2 \, dx \right]$$

independent of $\hat{p}$ ⇒ no effect on the optimal solution ⇒ drop

$$= \int p^*(x) \sum_m \hat{p}_m \, \mathbb{1}[x \in \text{bin}_m] \, dx$$

$$= \sum_m \hat{p}_m \int p^*(x) \, \mathbb{1}[x \in \text{bin}_m] \, dx$$

$$= \sum_m \hat{p}_m \, \mathbb{E}_{p^*(x)}\left[\mathbb{1}[x \in \text{bin}_m] \right] \approx \sum_m \hat{p}_m \cdot \frac{N_m}{N}$$

$$\int \hat{p}(x)^2 \, dx = \int \left(\sum_m \hat{p}_m \, \mathbb{1}[x \in \text{bin}_m] \right)^2 dx$$

$$= \sum_{m,m'} \hat{p}_m \hat{p}_{m'} \int \mathbb{1}[x \in \text{bin}_m] \, \mathbb{1}[x \in \text{bin}_{m'}]$$

$$= \sum_m \hat{p}_m^2 \underbrace{\int \mathbb{1}[x \in \text{bin}_m]^2}_{= \mathbb{1}[x \in \text{bin}_m]} + 2 \sum_m \sum_{m'>m} \hat{p}_m \hat{p}_{m'} \underbrace{\int \mathbb{1}[x \in \text{bin}_m] \, \mathbb{1}[x \in \text{bin}_{m'}]}_{= 0 \text{ bins do not overlap}}$$

$$= \sum_m \hat{p}_m^2 \int \mathbb{1}[x \in \text{bin}_m] \, dx = \sum_m \hat{p}_m^2 \cdot h$$

$$\{\hat{p}_m\} = \underset{\{p_m\}}{\arg\min} \sum_m p_m^2 \frac{\mathbb{1}[x \in \text{bin}_m]}{h} - 2 \sum_m p_m \frac{N_m}{N}$$

$$\frac{\partial}{\partial p_m} \sum_m p_m^2 \frac{\mathbb{1}[x \in \text{bin}_m]}{h} - 2 \sum_m p_m \frac{N_m}{N} = 2 p_m \cdot h - 2 \frac{N_m}{N} \stackrel{!}{=} 0$$

$$\boxed{\hat{p}_m = \frac{N_m}{N \cdot h}}$$

$\hat{p}_m$ is a density, because $X \in \mathbb{R}$ $\Rightarrow$ devide by h (the "volume" of the bin)

$$\hat{p}(x) = \sum_m \frac{N_m}{N \cdot h} \mathbb{1}[x \in \text{bin}_m]$$

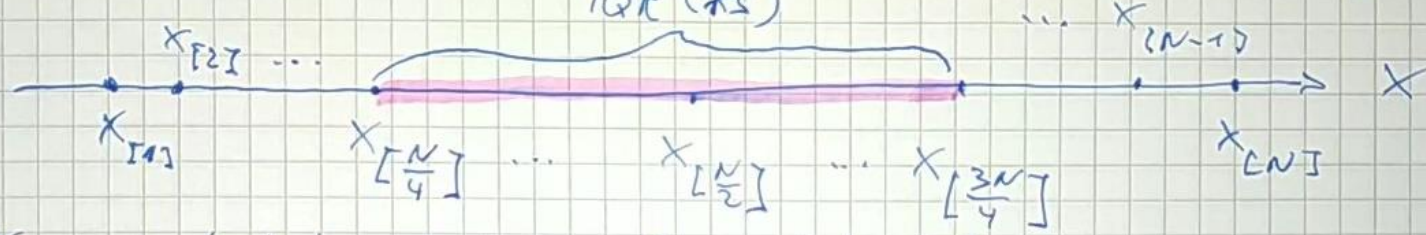
histogramm approximation of density $p^*(x)$

How to choose good values for h?

- rules of thumb: – scotts rule $h = \dfrac{3.5 \, \hat{\sigma}(TS)}{\sqrt[3]{N}}$ (has smallest error, when data are Gaussian distributed)

  – Freedman-Diaconis rule $h = \dfrac{2 \cdot IQR(TS)}{\sqrt[3]{N}}$ (equivalent to Scott when data is Gaussian, but more robust otherwise)

IQR (TS): "Inter Quartile Range" ≙ central interval containing 50% of the data



$x_{[i]}$: training instances in sorted order

Quiz: How many data instances are needed, when the true density is ~~$p^*(x)$ = Uniform~~ $p^*(x) = \text{Uniform}(0,1)$ and we want 100 bins?

$N = 1000$ (A) $= 10000$ (B) 10^5 (C) 10^6 (D) 10^7 (E)

- estimate h empirically: - candidate set for h: $h_1 = 0.1$ $h_2 = 0.3$, $h_3 = 1$, $h_4 = 3$, $h_5 = 10$
  - train the model for ~~of~~ each h_ℓ and determine the accuracy using an independent validation set (not reuse the training set, not use test set, because then the test set would become part of TS)
  - choose the model with winning h_ℓ and evaluate on test set
  - what to do if we don't have a validation set? ⇒ cross-validation

    here: N-fold cross-validation ≙ leave-one-out cross-validation
    - for each i: use (x_i, y_i) as a test point, and remaining TS for training
    - expensive: train the model N times

    exception: sometimes, one can train the model with entire TS and compute the effect of leave-one-out CV analytically

    this is possible for the histogram

analytic LOO-CV for the histogram

$$\hat{h} = \arg\min_h \int (\hat{p}^*(x) - \hat{p}(x))^2 \, dx$$

$\hat{p}^*$: truth; $\hat{p}$: optimal bin $\hat{p}$ for fixed h

$$= \arg\min_h \int \hat{p}^*(x)^2 \, dx - 2 \int \hat{p}^*(x)\hat{p}(x) \, dx + \underbrace{\int \hat{p}(x)^2 \, dx}_{= \sum_m \hat{p}_m^2 h = \sum_m \frac{N_m^2}{N^2 h}}$$

$$\int \hat{p}^*(x)\hat{p}(x) \, dx = \mathbb{E}_{\hat{p}^*(x)}[\hat{p}(x)] = \frac{1}{N}\sum_i \hat{p}(x_i)$$ not correct, because x_i is used twice:

- as training point to find $\hat{p}$
- as test point in $\hat{p}(x_i)$

$$= \frac{1}{N}\sum_i \underbrace{\hat{p}_{-i}(x_i)}_{\hat{p} \text{ trained without instance } x_i}$$

original hist. $\hat{p}_m = \frac{N_m}{N \cdot h}$ LOO histogram $\hat{p}_{-i,m} = \begin{cases} \frac{N_m}{(N-1)h} & \text{if } x_i \notin \text{bin}_m \\ \frac{N_m - 1}{(N-1)h} & \text{if } x_i \in \text{bin}_m \end{cases}$

[here, N_m, N are the values for full TS]

$$= \frac{N_m - \mathbb{1}[x_i \in \text{bin}_m]}{(N-1) \cdot h}$$

$$\frac{1}{N}\sum_i \hat{p}_{-i}(x_i) = \frac{1}{N}\sum_{i,m} \frac{N_m - \mathbb{1}[x_i \in \text{bin}_m]}{(N-1)\cdot h} \cdot \mathbb{1}[x_i \in \text{bin}_m]$$

$$= \frac{1}{N}\sum_i \sum_m \left(\frac{N_m}{(N-1)h}\mathbb{1}[x_i \in \text{bin}_m] - \frac{\mathbb{1}[x_i \in \text{bin}_m]^2}{(N-1)h} \right) = \frac{1}{N(N-1)h}\sum_m (N_m - 1) \cdot \underbrace{\sum_i \mathbb{1}[x_i \in \text{bin}_m]}_{N_m}$$

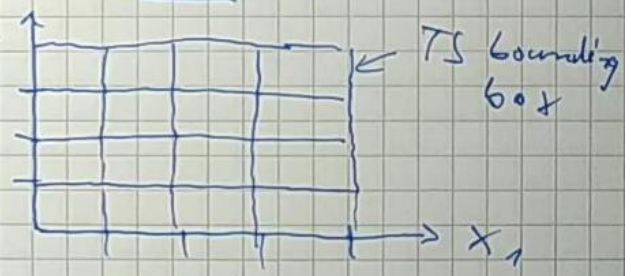
$$= \frac{1}{N(N-1)h} \sum_m (N_m - 1) N_m$$

$$(*) \quad \boxed{\hat{h} = \operatorname*{argmin}_h \; -\frac{2}{N(N-1)h} \sum_m (N_m - 1) N_m + \sum_m \frac{N_m^2}{N^2 h} = \operatorname*{argmin}_h \frac{1}{(N-1)h} \left(2 - (N+1) \sum_m \frac{N_m^2}{N^2} \right)}$$

Alg: for each candidate h_c, train the model on full TS ($\hat{=}$ determine N_m) and keep h_c where the RMS of $(*)$ is smallest

Generalization of histograms to D-dimensional features is not easy:

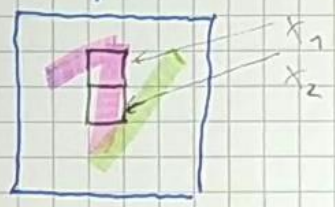
- suppose we want M bins per dimension
  split the bounding box along each feature $M = 4$
  ⇒ total number of bin explodes M^D
  (e.g. $M = 10$, $D = 50$: 10^{50} bins,
  you can never collect so many training instances (need 10^{150})
  only works for $D \leq 4 \ldots 5$

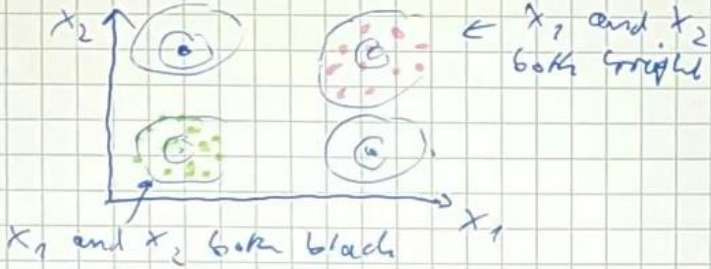


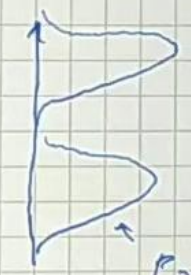
- in higher dimensions: traditional trick: assume that features are independent of each other
  ex: fruit, feature 1: color feature 2: size
  knowing color = red doesn't help to guess the size
  ⇒ joint prob of features factorizes $p(x_1, \ldots, x_D) = \prod_j p_j(x_j)$
  ⇒ represent the one-dimensional distributions $p_j(x_j)$ by 1-D histograms

unfortunately, the assumption is rarely true in practice,

ex: digits data from homework:







independence assumption:
compute marginals

product approx: $p(x_1, x_2) \approx p_1(x_1)\, p_2(x_2)$

$\Rightarrow$ spurious clusters at
$x_1 \uparrow x_2 \downarrow$ and $x_1 \downarrow x_2 \uparrow$

$\Rightarrow$ spurious cluster lead to incorrect densities,
and classification errors

<u>still, the method is used sometimes</u>: <u>Naive Bayes Classifier</u>

generative model, learn $p(X \mid Y=k) = \prod_j p_j(X_j \mid Y=k) = \prod_j \underbrace{p_{j,k}(X_j)}_{1\text{-D}}$

$p_{j,k}(X_j)$: one 1-dimensional density per feature per class

two possibilities: represent $p_{j,k}(X_j)$ as a 1-D histogram

1-D Gaussian distribution

$p_{j,k}(X_j) \approx \mathcal{N}(\mu_{j,k}, \sigma^2_{j,k}) \Rightarrow p(x \mid Y=k) = \mathcal{N}(\mu_k, \Sigma_k)$

diagonal matrix $\Sigma_k = \begin{pmatrix} \sigma^2_{1,k} & & 0 \\ & \ddots & \\ 0 & & \sigma^2_{d,k} \end{pmatrix}$

equivalent to QDA restricted to diagonal Σ
("mean-field approximation")