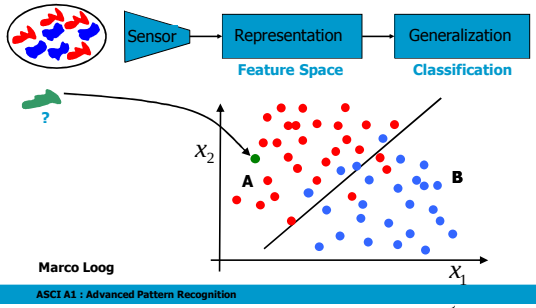
Classifiers



Marco Loog

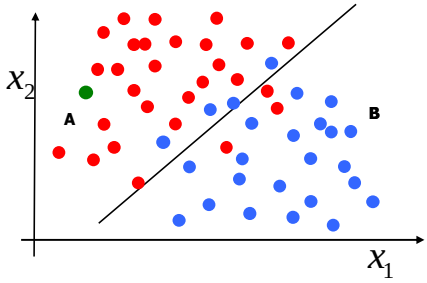
TU Delft
Delft University of Technology

The Problem of Classification

- Learn a decision rule, from finite training data, that assigns a new object x to one of K classes
- Rule aims to minimize the error of classification
- Typically there is no perfect, i.e., zero-error, rule
- Decision rule is obtained by training a classifier

TU Delft

Ill-posed Problem



TU Delft

Some Principles

- Parametric classifiers vs. nonparametric classifiers
- Linear vs. nonlinear
- Generative classifiers : focus on each class separately, model class conditional densities [likelihoods] and reason about discrimination
- Discriminative classifiers : focus on discrimination directly, model decision function [or posterior probabilities]

TU Delft

Normal Density-based Classifiers

$$p(\mathbf{x}\,|\,\omega_i) = \frac{1}{\sqrt{2\pi^k \det(\boldsymbol{\Sigma}_i)}} \exp\left(-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)\right) = \Phi(x,\mu_i,\Sigma_i)$$

TU Delft

Normal Density-based Classifiers

Bayes classifier $S(x) = p(x\,|\,\mathrm{A})p(\mathrm{A}) - p(x\,|\,\mathrm{B})p(\mathrm{B})$

logs don't matter

$$R(x) = \log(p(x\,|\,\mathrm{A})p(\mathrm{A})) - \log(p(x\,|\,\mathrm{B})p(\mathrm{B}))$$

$$R(x) = \log(p(x\,|\,\mathrm{A})) - \log(p(x\,|\,\mathrm{B}) + \log(p(\mathrm{A})/p(\mathrm{B}))$$

normal distribution

$$p(\mathrm{x}\,|\,\mathrm{A}) = \frac{1}{\sqrt{2\pi^k \det(\boldsymbol{\Sigma}_\mathrm{A})}} \exp\left(-\frac{1}{2}(x-\mu_\mathrm{A})^T \Sigma_\mathrm{A}^{-1}(x-\mu_\mathrm{A})\right)$$

$$\log(p(\mathrm{x}\,|\,\mathrm{A})) = -\frac{1}{2}(x-\mu_\mathrm{A})^T \Sigma_\mathrm{A}^{-1}(x-\mu_\mathrm{A}) - \log(\sqrt{2\pi^k \det(\boldsymbol{\Sigma}_\mathrm{A})})$$

substitute

$$R(x) = -\frac{1}{2}(x-\mu_\mathrm{A})^T \Sigma_\mathrm{A}^{-1}(x-\mu_\mathrm{A}) + \frac{1}{2}(x-\mu_\mathrm{B})^T \Sigma_\mathrm{B}^{-1}(x-\mu_\mathrm{B}) + \mathrm{c}$$

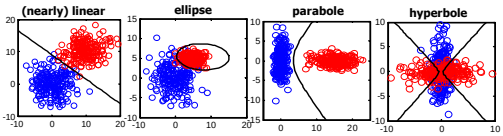
Quadratic expression

TU Delft

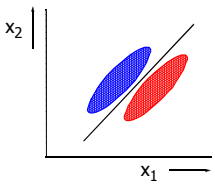
Normal Density-based Classifiers

$$R(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}-\hat{\boldsymbol{\mu}}_A)^T\hat{\Sigma}_A^{-1}(\mathbf{x}-\hat{\boldsymbol{\mu}}_A) + \frac{1}{2}(\mathbf{x}-\hat{\boldsymbol{\mu}}_B)^T\hat{\Sigma}_B^{-1}(\mathbf{x}-\hat{\boldsymbol{\mu}}_B) + c$$

$$c = \log(p(A)/p(B)) + \frac{1}{2}\log[\det(\hat{\Sigma}_B)/\det(\hat{\Sigma}_A)]$$

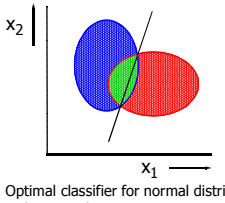


Normal Density-based Classifiers



Normal distributions with equal covariance matrices Σ are optimally separated by a linear classifier

$$S(\mathbf{x}) = (\boldsymbol{\mu}_A - \boldsymbol{\mu}_B)^T\Sigma^{-1}\mathbf{x} + c$$



Optimal classifier for normal distributions with unequal covariance matrices Σ_A and Σ_B can be approximated by:

$$S(\mathbf{x}) = (\boldsymbol{\mu}_A - \boldsymbol{\mu}_B)^T(p(A)\Sigma_A + p(B)\Sigma_B)^{-1}\mathbf{x} + c$$

Linear Discriminants

- General form : $S(x) = w^T x + w_0$
- E.g. ldc / LDA : normal density-based linear classifier

$$S(x) = (\mu_A - \mu_B)^T(p(A)\Sigma_A + p(B)\Sigma_B)^{-1}x + c$$

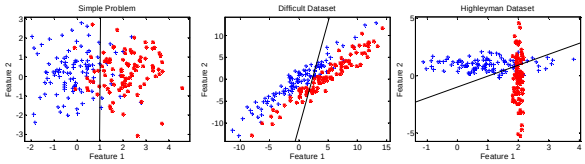
- E.g. fisherc : minimum variance linear classifier

$$S(x) = (\mu_A - \mu_B)^T\Sigma^{-1}x + c$$

  - Solves linear regression to indicator variables

Nearest Mean Classifier

$$S(x) = (\mu_A - \mu_B)^T x - (\mu_A - \mu_B)^T(\mu_A + \mu_B)/2$$



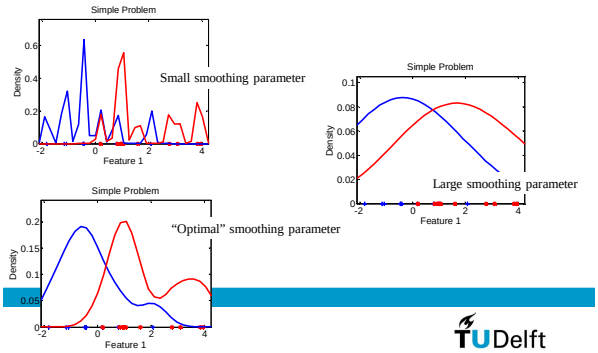
Parzen Classifier

- Substitute Parzen density estimates

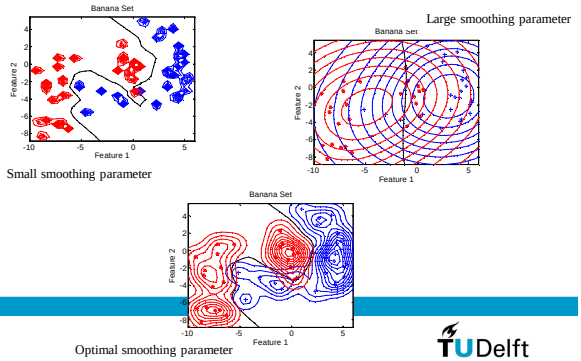
$$p(x \mid \omega) = \frac{1}{n}\sum_{x_i \in \omega}^{n}\Phi\left(\frac{x_i - x}{h}\right)$$

- Parzenc : optimize *h* for classification
- Parzendc : optimize *h* for density estimation per class

Parzen Classifier : 1D Example

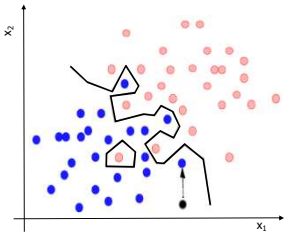


Parzen Classifier : 2D Example



TUDelft

Nearest Neighbor Rule

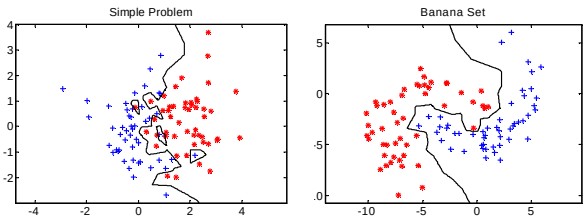


- The scaling issue…

TUDelft

More Nearest Neighbor

- Good for [almost] separable classes
- Useful for non-linear decision functions
- No training time, but long execution time
- All data should be stored



Asymptotics

- Nearest neighbor rule will, asymptotically, not perform worse than twice the best possible classifier

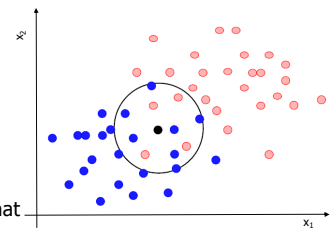
$$\varepsilon^* \leq \varepsilon_{1NN} \leq 2\varepsilon^*(1-\varepsilon^*) \leq 2\varepsilon^*$$

- Indeed, 1NN often performs rather well and provides good baseline results

TUDelft

k-Nearest Neighbor Classifier

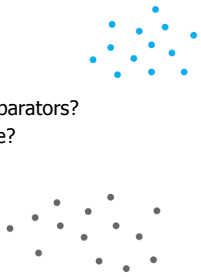
- More smooth
- Less local



- Generally, one can show that

$$\varepsilon^* \leq \varepsilon_{kNN} \leq (1+\sqrt{2/k})\varepsilon^*$$

TUDelft

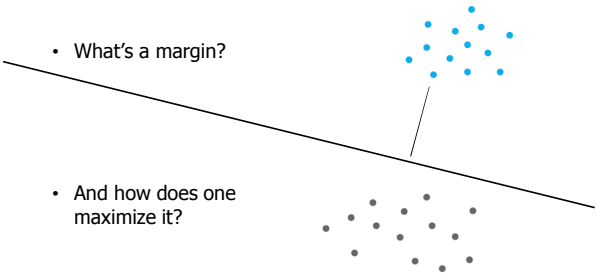


- How many perfect linear separators?
- What would be your favorite?

TUDelft

Support Vector Machines

- What's a margin?
- And how does one maximize it?



TUDelft

Naïve Bayes

- Assume features independent given class label
- Estimate class-conditional densities by
- Bayes rule does the rest…

$$p(x|A) = \prod_{i=1}^{k} p(x_i \mid A)$$
$$p(x|B) = \prod_{i=1}^{k} p(x_i \mid B)$$

TUDelft

Logistic Regression / Classification

- Based on assumption that logarithm of posterior odds is linear

$$\log\left(\frac{p(A \mid x)}{p(B \mid x)}\right) = \log\left(\frac{p(A)p(x \mid A)}{p(B)p(x \mid B)}\right) = w^T x + w_0$$

- One can derive that

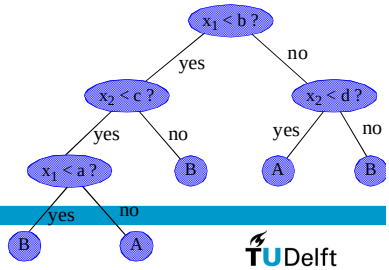
$$p(A \mid x) = \frac{p(A)p(x \mid A)}{p(A)p(x \mid A) + p(B)p(x \mid B)} = \frac{e^{w^T x + w_0}}{1 + e^{w^T x + w_0}} = \frac{1}{1 + e^{-w^T x - w_0}}$$

$$p(B \mid x) = \frac{p(B)p(x \mid B)}{p(A)p(x \mid A) + p(B)p(x \mid B)} = \frac{1}{1 + e^{w^T x + w_0}}$$

- Product over all terms in training set is optimized

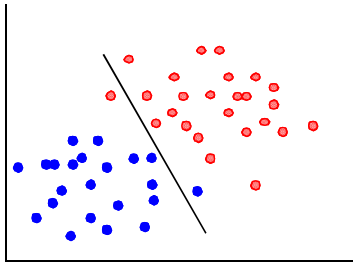
TUDelft

Decision Trees



TUDelft

Perceptron



- Linear classifier
- Trained trough simple update rule

$$w^{k+1} = w^k + \alpha \Delta w(w^k, x_i^{'}) = w^k + \alpha x_i^{'}$$

TUDelft

Somewhat more General View

- Given class M of functions / models m parameterized by θ Model $m(.|\theta)$ takes x as input and provides posterior, label, whatever as output
- Given training data (xs and ys), find that θ that minimizes the empirical risk :

$$\sum_i \ell(m(x_i|\theta), y_i)$$

- Loss ℓ defines how to penalize deviations op model prediction for x from true value y

TUDelft

Some Losses

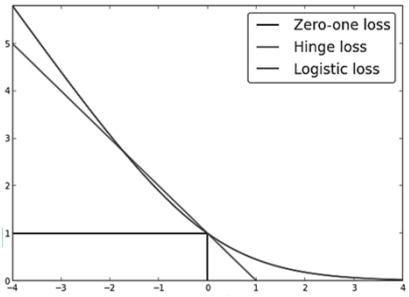
- 0/1 : $\begin{cases} 0, m(x|\theta) = y \\ 1, \text{otherwise} \end{cases}$
- Squared : $(m(x|\theta) - y)^2$
- Logistic : $\log(1 + \exp(-y\, m(x|\theta)))$
- Soft-margin / hinge : $\max(0, 1 - y\, m(x|\theta))$

- What loss is often used in generative modeling?

TUDelft

Why Approximate?

- In fact, we would like to optimize 0/1 in classification



TUDelft

Additional Remarks

- Rather than empirical risk, one often optimizes regularized / penalized / … risk

$$\sum_i \ell(m(x_i|\theta), y_i) + R(\theta)$$

- A priori, R-term biases solution to simple ones
- Needed when models too flexible / little training data

- Not all classifiers can be formulated in terms of above formalism… Or?

TUDelft

PRTools : Some Generative Classifiers

- NMC - nearest mean classifier
- NMSC - nearest mean scaled classifier
- FISHERC - Fisher linear discriminant
- LDC - linear discriminant
- QDC - quadratic discriminant
- UDC - quadratic discriminant diagonal covariance
- MOGC - mixture of Gaussians classification
- PARZENC - Parzen classifier
- NAIVEBC - naïve Bayes classifier

TUDelft

PRTools : Some Discriminative Classifiers

- TREEC - binary decision tree classifier
- BPXNC - backpropagation feed forward neural network
- LMNC - Levenberg-Marquardt feed forward neural net
- PERLC - linear perceptron
- RBNC - radial basis neural network classifier
- SUBSC - subspace classifier
- SVC - support vector machine
- KNNC - k-nearest neighbor rule

TUDelft