Introduction to Bayes' classifier and to the on-the-fly bayesian domain adaptation

Sébastien Piérard

INTELSIG, Montefiore Institute, University of Liège, Belgium

June 19, 2014
(version updated on July 14, 2014)

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$$

Thomas Bayes
1702 (London, England) — 1761 (Tunbridge Wells, Kent, England)

Outline

Outline

An introductory example

- Can you decide which silhouettes are those of humans ?
- Try to write an algorithm to solve this problem !

An introductory example

Observation

Most of the tasks related to video scene interpretation are complex. A human expert can easily take the right decision, but usually without being able to explain how he does it.

Solution

Machine learning techniques are indispensable in computer science.

Outline

Notations

Let us denote :

- o an object (*i.e.* a sample) to be classified
- $\overrightarrow{x}(o)$ the information about o (a vector of attributes)
- c_i a class ("human silhouettes", "non-human silhouettes", *etc.*)
- $\hat{y}(o)$ the class of o estimated by the classifier
- $y(o)$ the ground-truth class of o
- $P[\cdot]$ a probability ($\in \mathbb{R} \cap (0, 1)$)
- $\rho(\cdot)$ a *pdf* ($\rho(x) \geq 0 \, \forall x$, $\int_{-\infty}^{+\infty} \rho(x) \, dx = 1$)

To shorten expressions, let us denote the probability density function of the objects belonging to class c_i by

$$\rho_i\left(\overrightarrow{x}(o)\right) = \rho\left(\overrightarrow{x}(o) \,|\, y(o) = c_i\right) \tag{1}$$

and the priors (that is the proportion of objects belonging to a given class) as

$$p_i = P\left[y(o) = c_i\right] \tag{2}$$

Bayes' classifier in case of a continuous attribute space I

Bayes' classifier minimizes the error rate, when the samples to classify are assumed independent, by predicting the most probable class :

$$\hat{y}(o) = \arg\max_{c_i} \left(P\left[y(o) = c_i | \overrightarrow{x}(o) \right] \right) \tag{3}$$

Using Bayes' rule is not straightforward since

$$P\left[y(o) = c_i | \overrightarrow{x}(o) \right] = \frac{P\left[y(o) = c_i \right] \overbrace{P\left[\overrightarrow{x}(o) | y(o) = c_i \right]}^{=0}}{\underbrace{P\left[\overrightarrow{x}(o) \right]}_{=0}}$$

Let us consider a small neighborhood ϵ around $\overrightarrow{x}(o)$ in the attribute space.

$$P\left[y(o) = c_i | \overrightarrow{x}(o) \right] = \lim_{V_\epsilon \to 0} P\left[y(o) = c_i | \overrightarrow{x}(o) \in \epsilon \right] \tag{4}$$

We have

$$\begin{aligned} P\left[\overrightarrow{x}\left(o\right)\in\epsilon\wedge y\left(o\right)=c_i\right] &= P\left[y\left(o\right)=c_i\right]\,P\left[\overrightarrow{x}\left(o\right)\in\epsilon | y\left(o\right)=c_i\right] \\ &= p_i\int_{\overrightarrow{x}(o)\in\epsilon}\rho_i\left(x\right)dx \\ &\simeq p_i\,\rho_i\left(\overrightarrow{x}\left(o\right)\right)\,V_\epsilon \end{aligned} \tag{5}$$

where V_ϵ denotes the volume of ϵ. We also have

$$\begin{aligned} P\left[\overrightarrow{x}\left(o\right)\in\epsilon\right] &\simeq \sum_{c_i} p_i\,\rho_i\left(\overrightarrow{x}\left(o\right)\right)\,V_\epsilon \\ &= V_\epsilon\sum_{c_i} p_i\,\rho_i\left(\overrightarrow{x}\left(o\right)\right) \\ &= V_\epsilon\,\rho_\star\left(\overrightarrow{x}\left(o\right)\right) \end{aligned} \tag{6}$$

where $\rho_\star\left(\cdot\right)=\sum_{c_i} p_i\,\rho_i\left(\cdot\right)$ denotes the overall probability density function.

Using Bayes' rule, we have

$$\begin{aligned} P\left[y(o)=c_i|\overrightarrow{x}(o)\in\epsilon\right] &= \frac{P\left[\overrightarrow{x}(o)\in\epsilon \wedge y(o)=c_i\right]}{P\left[\overrightarrow{x}(o)\in\epsilon\right]} \\ &\simeq \frac{p_i\,\rho_i\left(\overrightarrow{x}(o)\right)\,V_\epsilon}{V_\epsilon\,\rho_\star\left(\overrightarrow{x}(o)\right)} \\ &= \frac{p_i\,\rho_i\left(\overrightarrow{x}(o)\right)}{\rho_\star\left(\overrightarrow{x}(o)\right)} \end{aligned} \tag{7}$$

and therefore,

$$\begin{aligned} P\left[y(o)=c_i|\overrightarrow{x}(o)\right] &= \lim_{V_\epsilon\to 0} P\left[y(o)=c_i|\overrightarrow{x}(o)\in\epsilon\right] \\ &= \lim_{V_\epsilon\to 0}\frac{p_i\,\rho_i\left(\overrightarrow{x}(o)\right)}{\rho_\star\left(\overrightarrow{x}(o)\right)} \\ &= \frac{p_i\,\rho_i\left(\overrightarrow{x}(o)\right)}{\rho_\star\left(\overrightarrow{x}(o)\right)} \end{aligned} \tag{8}$$

Bayes' classifier in case of a continuous attribute space IV

In summary, Bayes' classifier computes

$$\hat{y}(o) = \arg\max_{c_i} \left(P\left[y(o) = c_i | \overrightarrow{x}(o) \right] \right)$$

where [4]

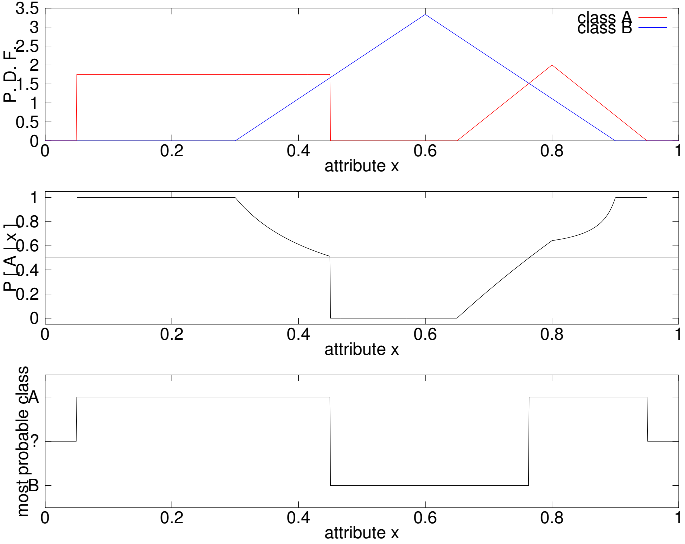
$$P\left[y(o) = c_i | \overrightarrow{x}(o) \right] = \frac{p_i \, \rho_i \left(\overrightarrow{x}(o) \right)}{\rho_\star \left(\overrightarrow{x}(o) \right)}$$

As $\rho_\star \left(\overrightarrow{x}(o) \right)$ depends only on $\overrightarrow{x}(o)$ (and not on c_i),

$$\hat{y}(o) = \arg\max_{c_i} \left(p_i \, \rho_i \left(\overrightarrow{x}(o) \right) \right) \tag{9}$$

The intrinsic difficulty of a classifier is that it is very difficult to estimate $\rho_i(\cdot)$ from a learning set because the space is not densely sampled.

An example in a 1D attribute space ($P[y = A] = 0.50$)



Outline

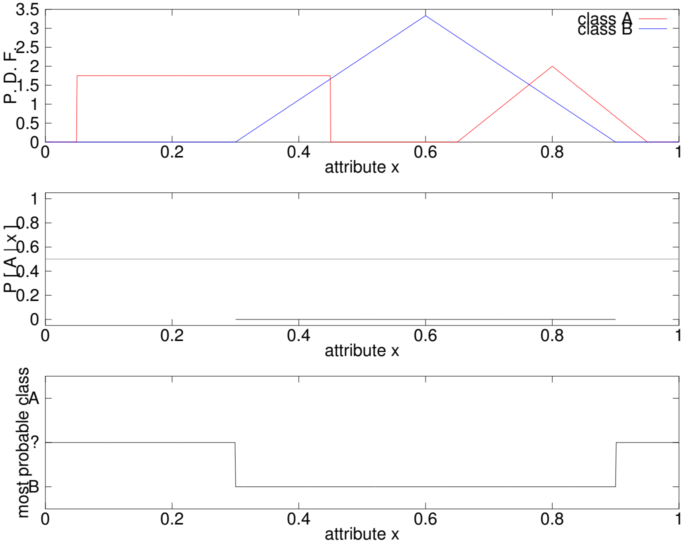
The class overlapping

There is some class overlapping when the supports of the probability density functions underlying the various classes are not mutually exclusive sets.

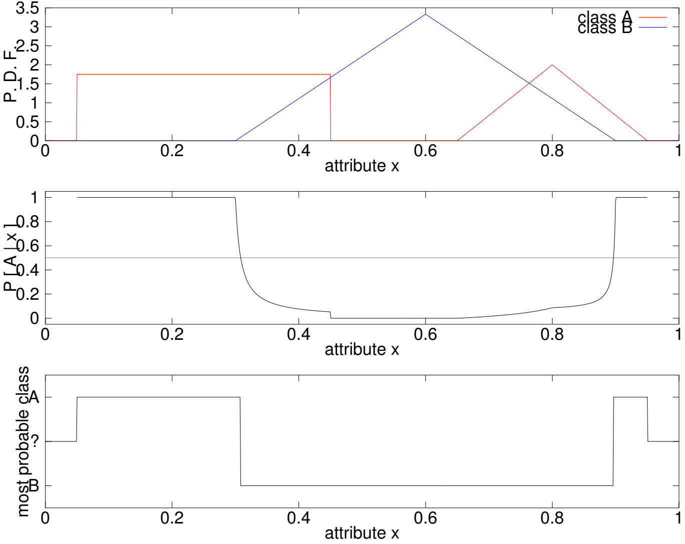
When there is no class overlapping, estimating correctly the probability density functions is not a matter of concern for the machine learning algorithm ; only their supports matter. Moreover, the classifier is insensitive to the priors (when > 0).

Taking the priors into account is most often important in case of class overlapping. But is some case, the classifier can still be insensitive to the priors when the range of expected priors is restricted (see example on the next slides).

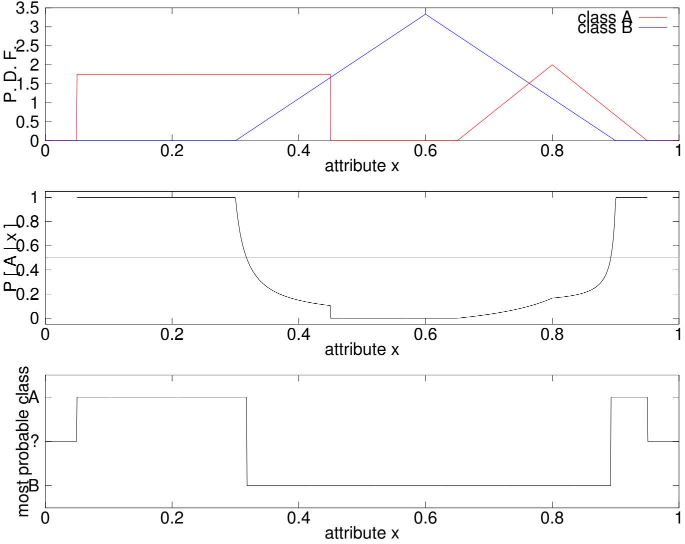
An example in a 1D attribute space ($P[y = A] = 0.00$)



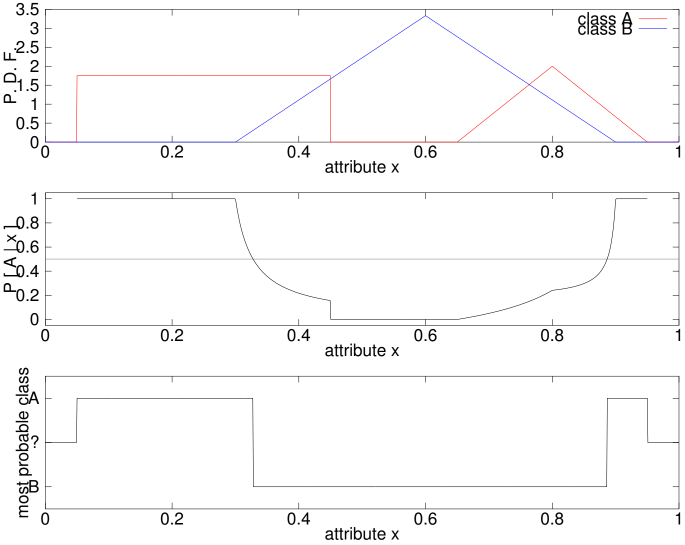
An example in a 1D attribute space ($P[y = A] = 0.05$)



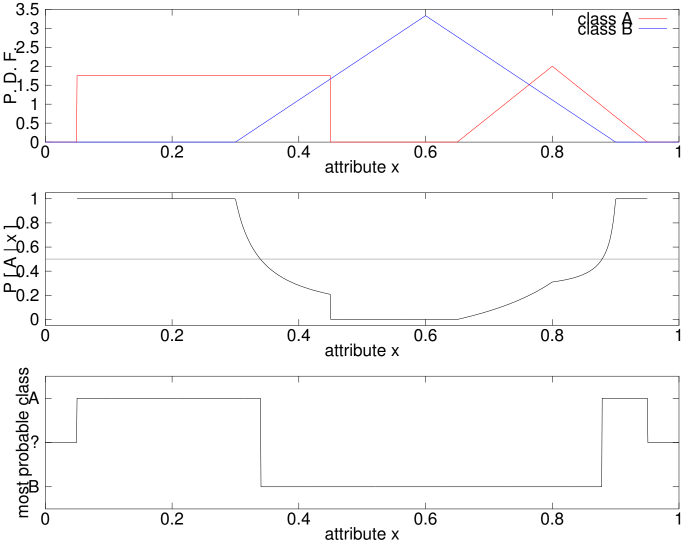
An example in a 1D attribute space ($P[y = A] = 0.10$)



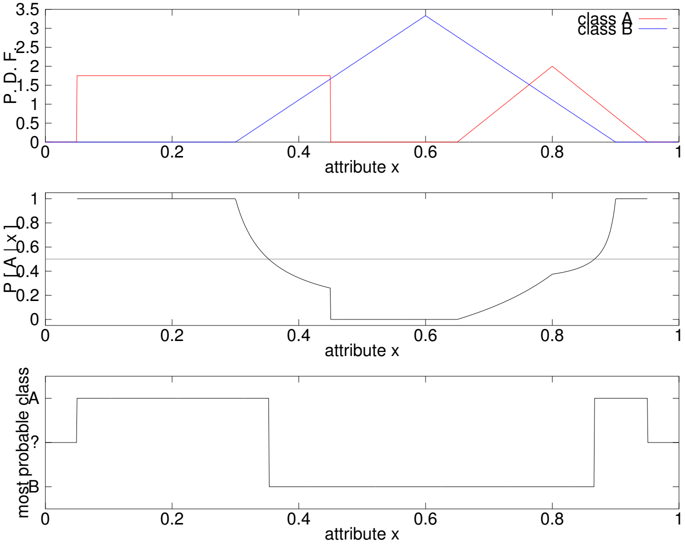
An example in a 1D attribute space ($P[y = A] = 0.15$)



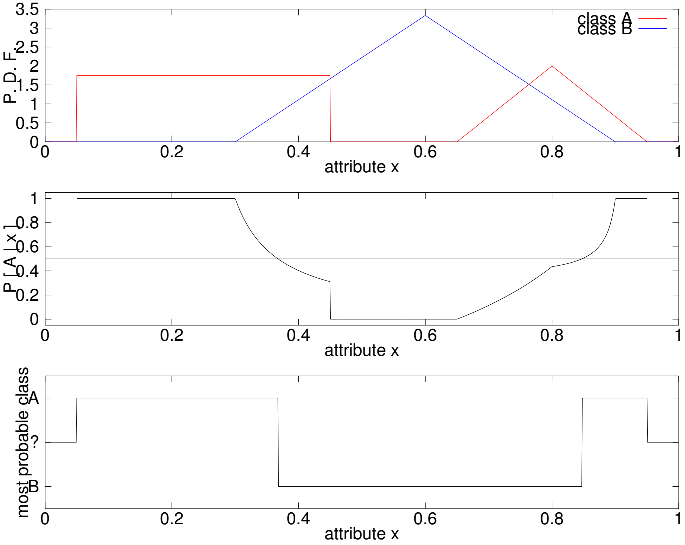
An example in a 1D attribute space ($P[y = A] = 0.20$)



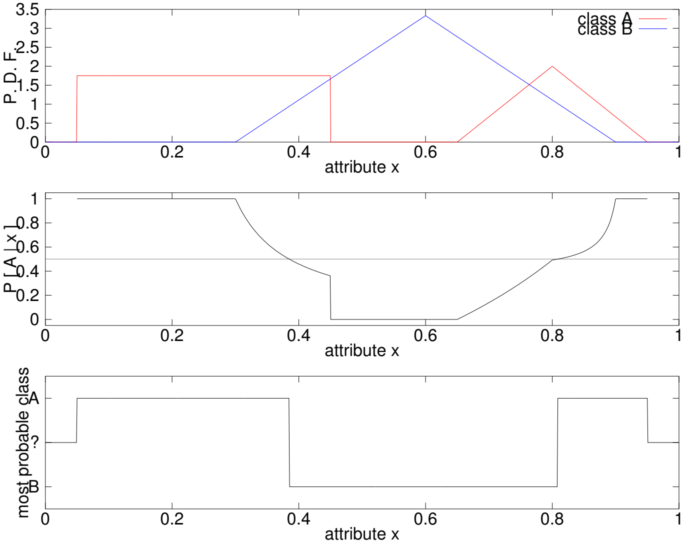
An example in a 1D attribute space ($P[y = A] = 0.25$)



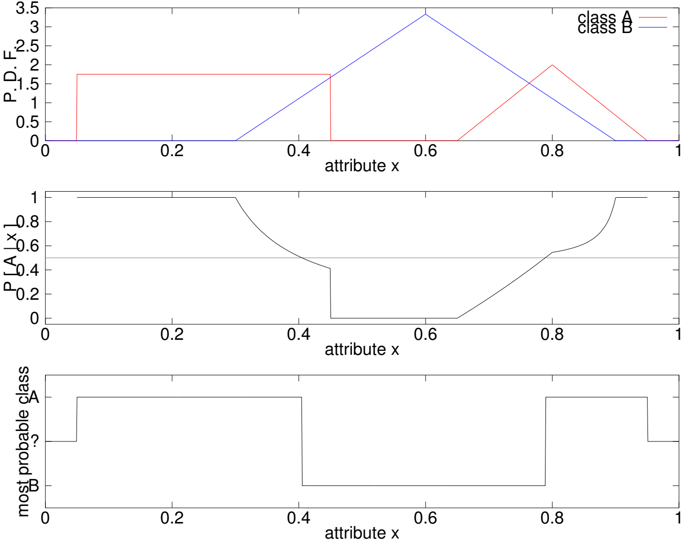
An example in a 1D attribute space ($P[y = A] = 0.30$)



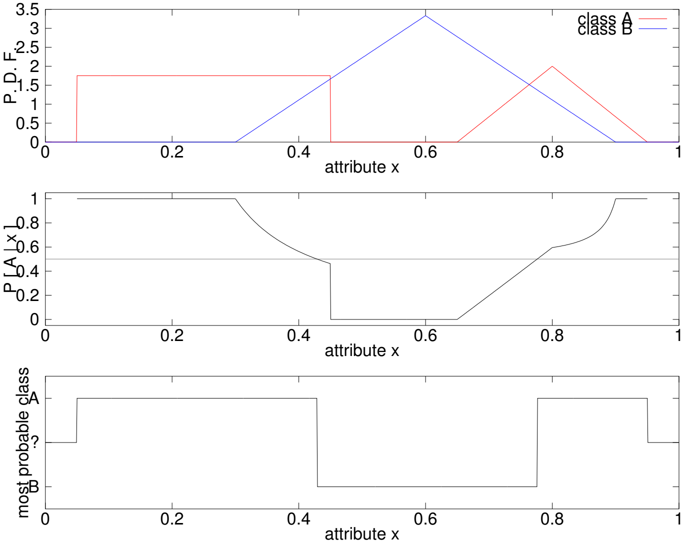
An example in a 1D attribute space ($P[y = A] = 0.35$)



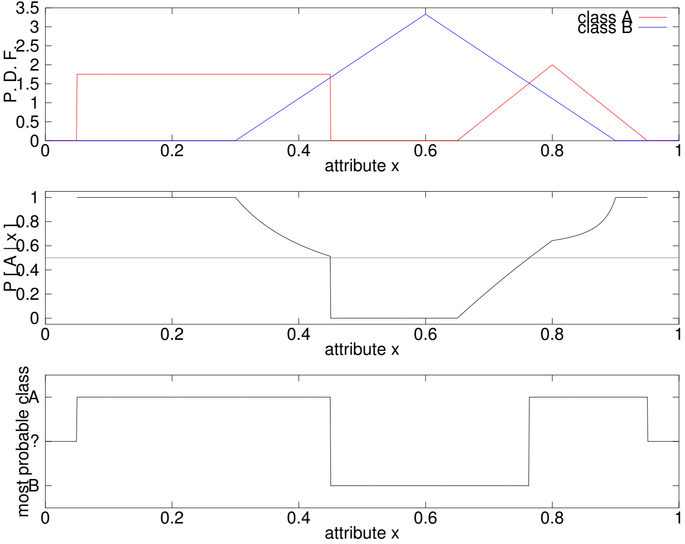
An example in a 1D attribute space ($P[y = A] = 0.40$)



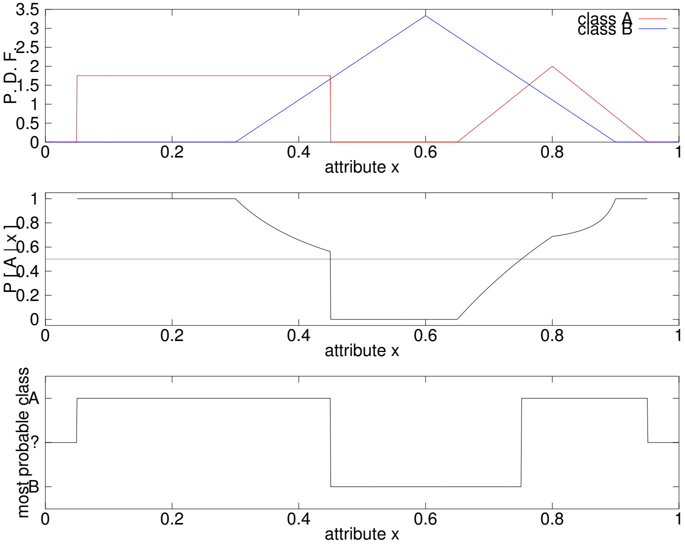
An example in a 1D attribute space ($P[y = A] = 0.45$)



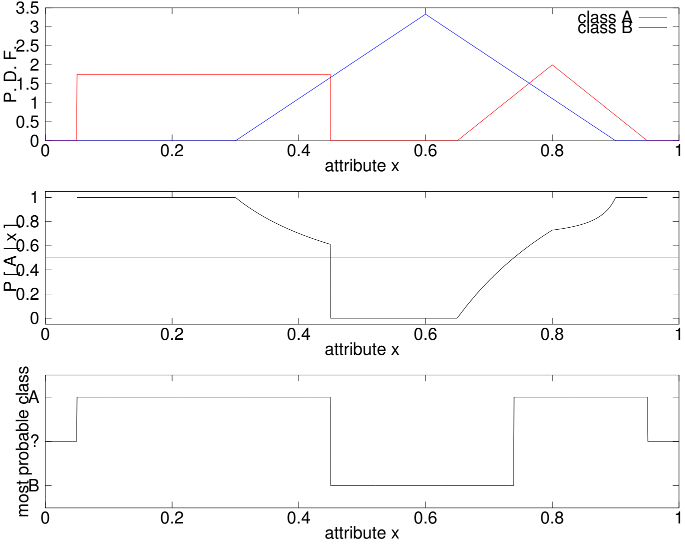
An example in a 1D attribute space ($P[y = A] = 0.50$)



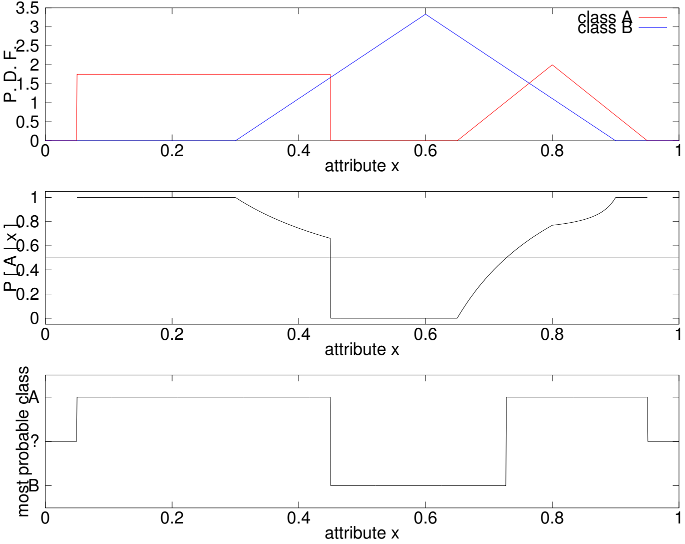
An example in a 1D attribute space ($P[y = A] = 0.55$)



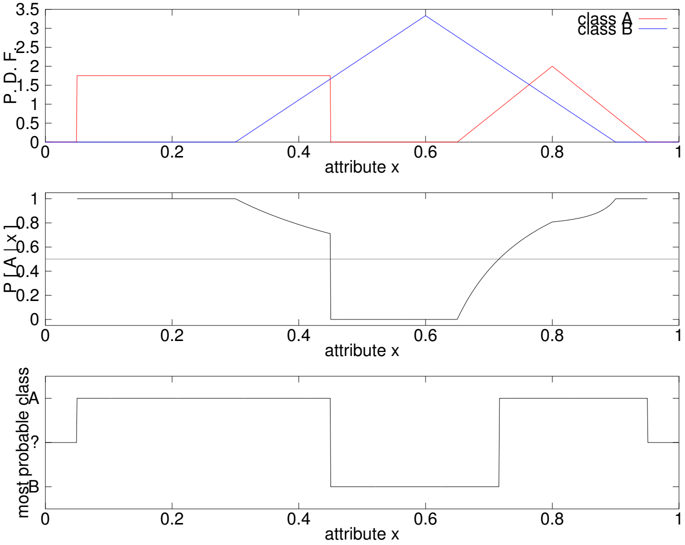
An example in a 1D attribute space ($P[y = A] = 0.60$)



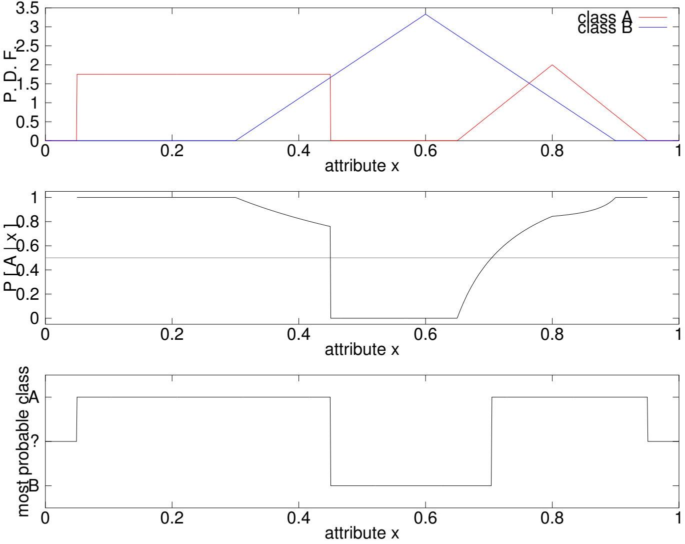
An example in a 1D attribute space ($P[y = A] = 0.65$)



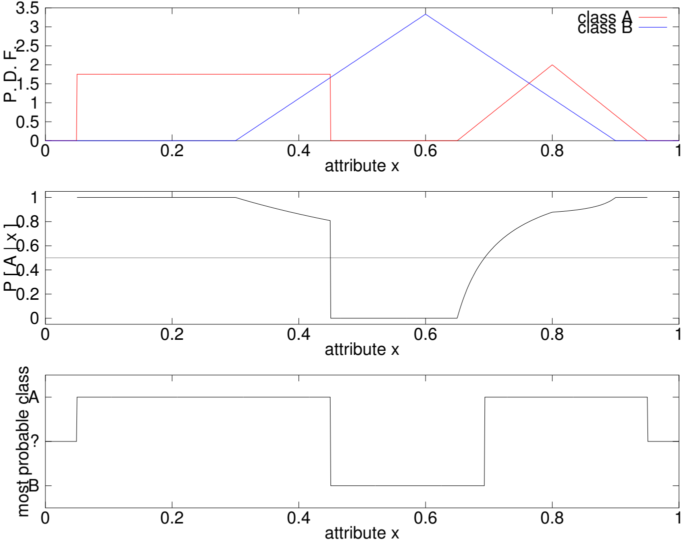
An example in a 1D attribute space ($P[y = A] = 0.70$)



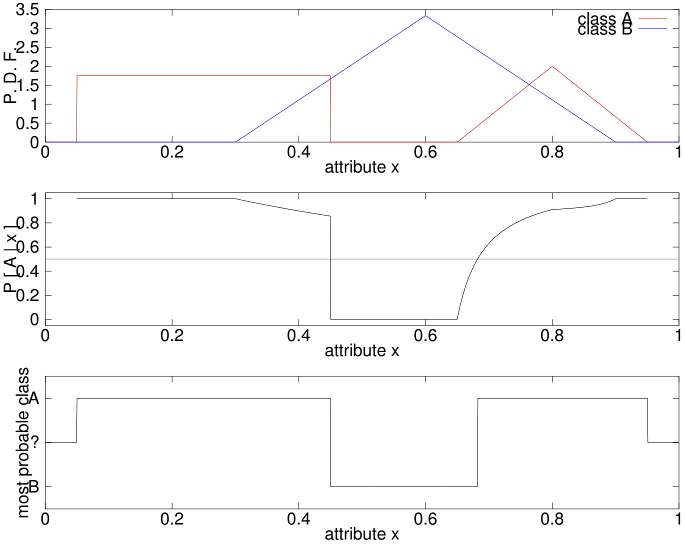
An example in a 1D attribute space ($P[y = A] = 0.75$)



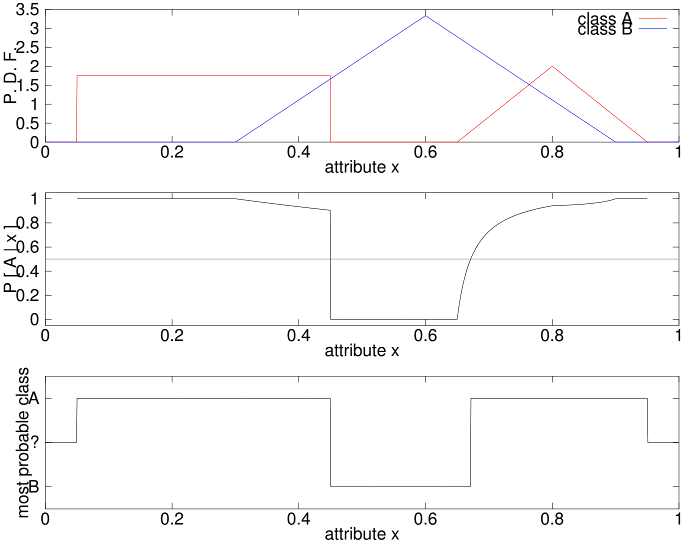
An example in a 1D attribute space ($P[y = A] = 0.80$)



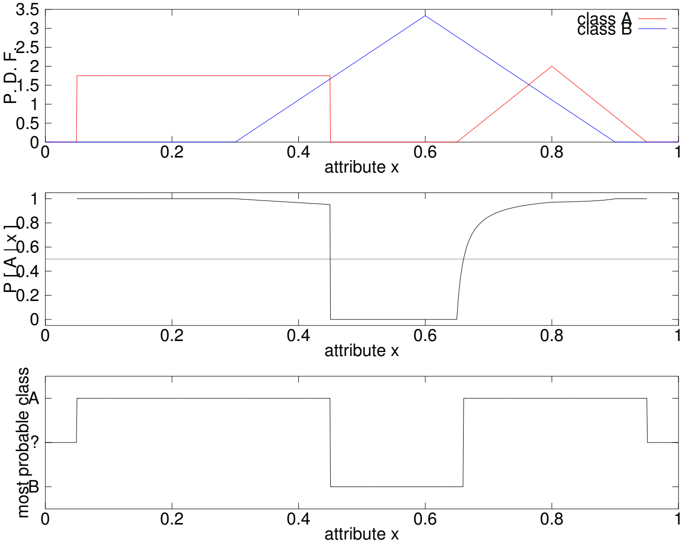
An example in a 1D attribute space ($P[y = A] = 0.85$)



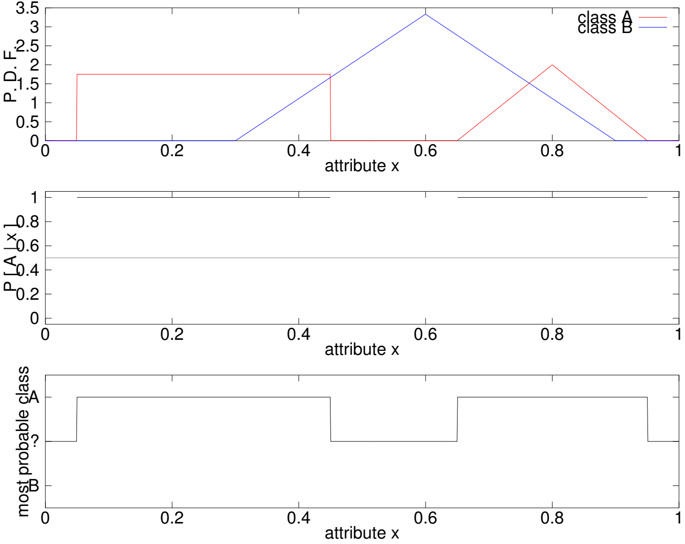
An example in a 1D attribute space ($P[y = A] = 0.90$)



An example in a 1D attribute space ($P[y = A] = 0.95$)



An example in a 1D attribute space ($P[y = A] = 1.00$)



Outline

The balance of the learning set

We have seen that Bayes' classifier computes

$$\begin{aligned}\hat{y}(o) &= \arg\max_{c_i} \left(P\left[y(o) = c_i | \overrightarrow{x}(o)\right]\right) \\ &= \arg\max_{c_i} \left(p_i\, \rho_i\left(\overrightarrow{x}(o)\right)\right)\end{aligned}$$

Therefore, a machine learning algorithm approximating Bayes' classifier needs to estimate the priors and the probability density functions (*pdfs*) from the learning set, either implicitly or explicitly.

- Intuitively, in order to correctly learn the *pdfs*, we need more samples drawn from the *pdfs* with complicated shapes than from the ones that are smooth.
- In order to estimate correctly the priors, the proportions of learning samples from the various classes need to reflect the priors.

These two aims can be contradictory. But we can focus on the first one, and compensate for the second one (if 2 classes) !

The case of the two-classes classifier I

- Let us consider the two classes c_- and c_+.
- Let n_-^{LS} be the amount of samples $\in c_-$ in the learning set.
- Let n_+^{LS} be the amount of samples $\in c_+$ in the learning set.

Since $P\left[y\left(o\right)=c_-|\overrightarrow{x}\left(o\right)\right]+P\left[y\left(o\right)=c_+|\overrightarrow{x}\left(o\right)\right]=1$, we can only focus of $P\left[y\left(o\right)=c_+|\overrightarrow{x}\left(o\right)\right]$. We would like to compute

$$P\left[y\left(o\right)=c_+|\overrightarrow{x}\left(o\right)\right]=\frac{p_+\,\rho_+\left(\overrightarrow{x}\left(o\right)\right)}{p_+\,\rho_+\left(\overrightarrow{x}\left(o\right)\right)+p_-\,\rho_-\left(\overrightarrow{x}\left(o\right)\right)} \tag{10}$$

but, when $\frac{n_-^{LS}}{n_-^{LS}+n_+^{LS}}\neq p_-\Leftrightarrow\frac{n_+^{LS}}{n_-^{LS}+n_+^{LS}}\neq p_+$, the machine learning algorithm computes

$$z\left(\overrightarrow{x}\left(o\right)\right)=\frac{\frac{n_+^{LS}}{n_-^{LS}+n_+^{LS}}\,\rho_+\left(\overrightarrow{x}\left(o\right)\right)}{\frac{n_+^{LS}}{n_-^{LS}+n_+^{LS}}\,\rho_+\left(\overrightarrow{x}\left(o\right)\right)+\frac{n_-^{LS}}{n_-^{LS}+n_+^{LS}}\,\rho_-\left(\overrightarrow{x}\left(o\right)\right)} \tag{11}$$

Can we get $P\left[y\left(o\right)=c_+|\overrightarrow{x}\left(o\right)\right]$ back from $z\left(\overrightarrow{x}\left(o\right)\right)$? Yes !

The case of the two-classes classifier II

$$
\begin{aligned}
z\left(\overrightarrow{x}(o)\right) &= \frac{\frac{n_+^{LS}}{n_-^{LS}+n_+^{LS}}\,\rho_+\left(\overrightarrow{x}(o)\right)}{\frac{n_+^{LS}}{n_-^{LS}+n_+^{LS}}\,\rho_+\left(\overrightarrow{x}(o)\right)+\frac{n_-^{LS}}{n_-^{LS}+n_+^{LS}}\,\rho_-\left(\overrightarrow{x}(o)\right)} \\
&= \frac{1}{1+\frac{n_-^{LS}}{n_+^{LS}}\,\frac{\rho_-\left(\overrightarrow{x}(o)\right)}{\rho_+\left(\overrightarrow{x}(o)\right)}} \\
P\left[y(o)=c_+|\overrightarrow{x}(o)\right] &= \frac{p_+\,\rho_+\left(\overrightarrow{x}(o)\right)}{p_+\,\rho_+\left(\overrightarrow{x}(o)\right)+p_-\,\rho_-\left(\overrightarrow{x}(o)\right)} \\
&= \frac{1}{1+\frac{p_-}{p_+}\,\frac{\rho_-\left(\overrightarrow{x}(o)\right)}{\rho_+\left(\overrightarrow{x}(o)\right)}}
\end{aligned}
$$

So we can compute

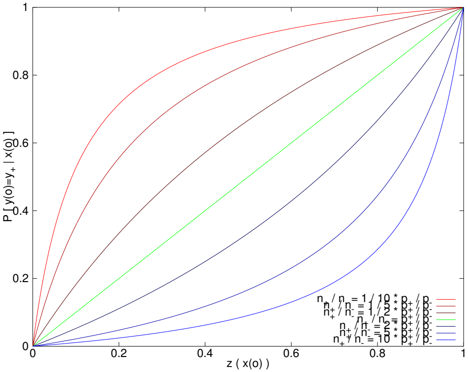
$$
z\left(\overrightarrow{x}(o)\right) \longmapsto \frac{\rho_-\left(\overrightarrow{x}(o)\right)}{\rho_+\left(\overrightarrow{x}(o)\right)} \longmapsto P\left[y(o)=c_+|\overrightarrow{x}(o)\right]
$$

$$
\begin{aligned}
z\left(\overrightarrow{x}(o)\right) &= \frac{1}{1+\frac{n_-^{LS}}{n_+^{LS}}\frac{\rho_-\left(\overrightarrow{x}(o)\right)}{\rho_+\left(\overrightarrow{x}(o)\right)}} \\
\Longleftrightarrow \frac{\rho_-\left(\overrightarrow{x}(o)\right)}{\rho_+\left(\overrightarrow{x}(o)\right)} &= \frac{n_+^{LS}}{n_-^{LS}}\frac{1-z\left(\overrightarrow{x}(o)\right)}{z\left(\overrightarrow{x}(o)\right)} \\
P\left[y(o)=c_+|\overrightarrow{x}(o)\right] &= \frac{1}{1+\frac{p_-}{p_+}\frac{\rho_-\left(\overrightarrow{x}(o)\right)}{\rho_+\left(\overrightarrow{x}(o)\right)}} \\
&= \frac{1}{1+\frac{p_-\,n_+^{LS}\left(1-z\left(\overrightarrow{x}(o)\right)\right)}{p_+\,n_-^{LS}\,z\left(\overrightarrow{x}(o)\right)}} \\
&= \frac{p_+\,n_-^{LS}\,z\left(\overrightarrow{x}(o)\right)}{p_-\,n_+^{LS}+\left(p_+\,n_-^{LS}-p_-\,n_+^{LS}\right)\,z\left(\overrightarrow{x}(o)\right)}
\end{aligned}
$$

The case of the two-classes classifier IV

Generalizing a result presented in [4], we have

$$P\left[y(o) = c_+ | \overrightarrow{x}(o)\right] = \frac{p_+ \, n_-^{LS} \, z\left(\overrightarrow{x}(o)\right)}{p_- \, n_+^{LS} + \left(p_+ \, n_-^{LS} - p_- \, n_+^{LS}\right) \, z\left(\overrightarrow{x}(o)\right)} \tag{12}$$



The 3 kinds of priors

There are 3 different kinds of priors :

- the priors in the context of use of the classifier : p_-, p_+
- the priors in the learning set (LS) : $\frac{n_-^{LS}}{n_-^{LS}+n_+^{LS}}$, $\frac{n_+^{LS}}{n_-^{LS}+n_+^{LS}}$
- the priors in the test set (TS) : $\frac{n_-^{TS}}{n_-^{TS}+n_+^{TS}}$, $\frac{n_+^{TS}}{n_-^{TS}+n_+^{TS}}$

The goal is to be able to predict the performance of the classifier in the target context, even when the 3 kinds of priors are different.

The ROC (receiver operating characteristic) and PR (precision-recall) evaluation spaces are insensitive to the priors in LS (the explanation follows on the next slides).

The PR space depends on the priors in TS (because the precision does). The ROC space does not depend on the priors in TS (because of the focus on the true positive and true negative rates).

The classifier does not depend on the priors in LS I

$$
\begin{aligned}
P\left[y(o) = c_+ | \overrightarrow{x}(o)\right] &< t \\
\iff \frac{1}{1 + \frac{p_-}{p_+} \frac{\rho_-\left(\overrightarrow{x}(o)\right)}{\rho_+\left(\overrightarrow{x}(o)\right)}} &< t \\
\iff \frac{\rho_-\left(\overrightarrow{x}(o)\right)}{\rho_+\left(\overrightarrow{x}(o)\right)} &> \frac{p_+}{p_-}\left(\frac{1}{t} - 1\right) \\
\iff \frac{1}{1 + \frac{n_-^{LS}}{n_+^{LS}} \frac{\rho_-\left(\overrightarrow{x}(o)\right)}{\rho_+\left(\overrightarrow{x}(o)\right)}} &< \frac{1}{1 + \frac{n_-^{LS}}{n_+^{LS}} \frac{p_+}{p_-}\left(\frac{1}{t} - 1\right)} \\
\iff z\left(\overrightarrow{x}(o)\right) &< \frac{1}{1 + \frac{n_-^{LS}}{n_+^{LS}} \frac{p_+}{p_-}\left(\frac{1}{t} - 1\right)}
\end{aligned}
$$

The classifier does not depend on the priors in LS II

$$P\left[y\left(o\right)=c_{+}|\overrightarrow{x}\left(o\right)\right]<t\Longleftrightarrow z\left(\overrightarrow{x}\left(o\right)\right)<t' \tag{13}$$

$$t'=\frac{p_{-}\,n_{+}^{LS}\,t}{p_{+}\,n_{-}^{LS}+\left(p_{-}\,n_{+}^{LS}-p_{+}\,n_{-}^{LS}\right)\,t} \tag{14}$$

$$t=\frac{p_{+}\,n_{-}^{LS}\,t'}{p_{-}\,n_{+}^{LS}+\left(p_{+}\,n_{-}^{LS}-p_{-}\,n_{+}^{LS}\right)\,t'} \tag{15}$$

For a binary Bayes' classifier, the priors in the learning set (LS) do not matter. We can simulate any prior in LS by tuning the decision threshold. Therefore, the shape of the performance curves in the ROC or precision-recall spaces does not depend on the priors in LS.

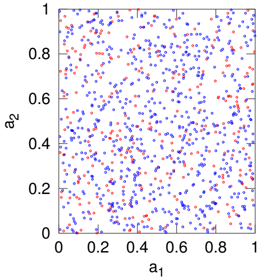
$$P\left[y\left(o\right)=c_{+}|\overrightarrow{x}\left(o\right)\right]<\frac{1}{2}\Longleftrightarrow z\left(\overrightarrow{x}\left(o\right)\right)<\frac{p_{-}\,n_{+}^{LS}}{p_{+}\,n_{-}^{LS}+p_{-}\,n_{+}^{LS}}$$

Outline

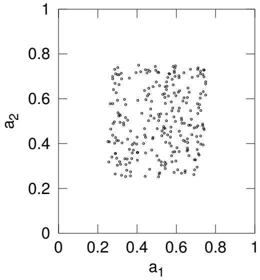
Experiment

- we assume the prediction given by the *ExtRaTrees* about the class of an object o depends only on the learning samples located in a neighborhood around $\overrightarrow{x}(o)$
- $\hookrightarrow$ we focus on small parts of the attribute space and assume the *pdfs* are uniform in first approximation in these parts
- our experiment is for a 2D attribute space
- we consider only the case where the trees are fully developed

learning set : example with 30 % positive samples

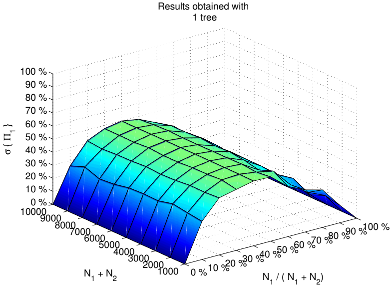


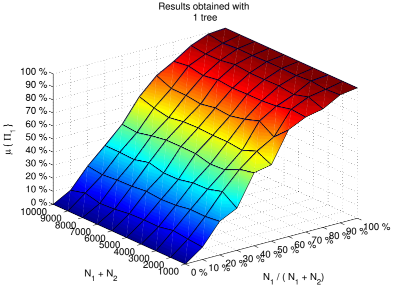
test set



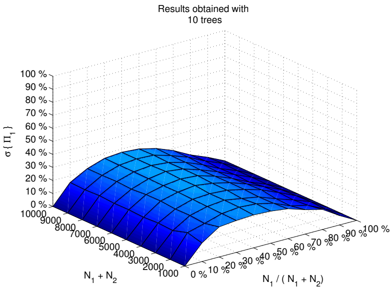
Results obtained with 1 tree in the forest

We observe the proportion $\Pi_+\left(\overrightarrow{x}(o)\right)$ of trees voting for the class c_+. As it is a random variable (we can draw many learning sets from the same *pdfs*), we plot its standard deviation and mean depending on the total amount of learning samples in the neighborhood around $\overrightarrow{x}(o)$ and the proportion of positive learning samples in this neighborhood.

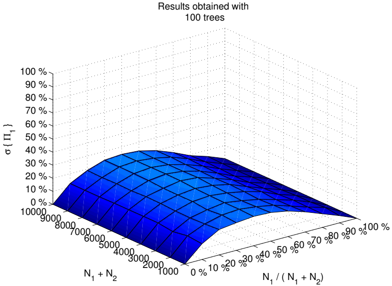


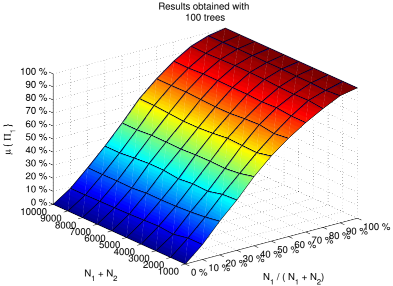


We observe the proportion $\Pi_+ \left(\overrightarrow{x}(o) \right)$ of trees voting for the class c_+. As it is a random variable (we can draw many learning sets from the same *pdfs*), we plot its standard deviation and mean depending on the total amount of learning samples in the neighborhood around $\overrightarrow{x}(o)$ and the proportion of positive learning samples in this neighborhood.

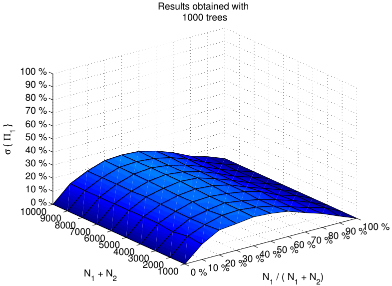


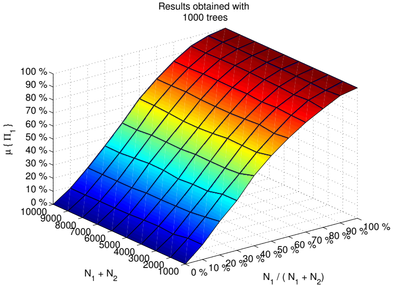
We observe the proportion $\Pi_+ \left(\overrightarrow{x}(o)\right)$ of trees voting for the class c_+. As it is a random variable (we can draw many learning sets from the same *pdfs*), we plot its standard deviation and mean depending on the total amount of learning samples in the neighborhood around $\overrightarrow{x}(o)$ and the proportion of positive learning samples in this neighborhood.





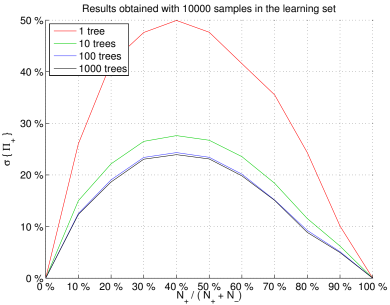
We observe the proportion $\Pi_+\left(\overrightarrow{x}(o)\right)$ of trees voting for the class c_+. As it is a random variable (we can draw many learning sets from the same *pdfs*), we plot its standard deviation and mean depending on the total amount of learning samples in the neighborhood around $\overrightarrow{x}(o)$ and the proportion of positive learning samples in this neighborhood.

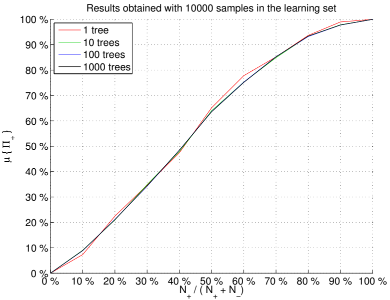




Summary of the results I

We observe the proportion $\Pi_+\left(\overrightarrow{x}(o)\right)$ of trees voting for the class c_+. As it is a random variable (we can draw many learning sets from the same *pdfs*), we plot its standard deviation and mean depending on the total amount of learning samples in the neighborhood around $\overrightarrow{x}(o)$ and the proportion of positive learning samples in this neighborhood.





Summary of the results II

The expected value of $\Pi_+ \left(\overrightarrow{x}(o)\right)$ is the proportion of positive learning samples in a neighborhood around $\overrightarrow{x}(o)$.
If other regions of the attribute space are populated into the learning set, this proportion is not $\frac{n_+^{LS}}{n_-^{LS}+n_+^{LS}}$ anymore. It is

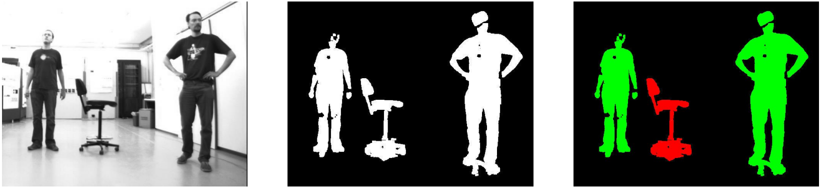
$$\begin{aligned}
\mu\left\{\Pi_+\left(\overrightarrow{x}(o)\right)\right\} &= \lim_{V_\epsilon \to 0} \frac{n_+^{LS} \int_{x\in\epsilon} \rho_+(x)\, dx}{n_-^{LS} \int_{x\in\epsilon} \rho_-(x)\, dx + n_+^{LS} \int_{x\in\epsilon} \rho_+(x)\, dx} \\
&\simeq \lim_{V_\epsilon \to 0} \frac{n_+^{LS} \left(V_\epsilon\, \rho_+\left(\overrightarrow{x}(o)\right)\right)}{n_-^{LS} \left(V_\epsilon\, \rho_-\left(\overrightarrow{x}(o)\right)\right) + n_+^{LS} \left(V_\epsilon\, \rho_+\left(\overrightarrow{x}(o)\right)\right)} \\
&= \frac{n_+^{LS}\, \rho_+\left(\overrightarrow{x}(o)\right)}{n_-^{LS}\, \rho_-\left(\overrightarrow{x}(o)\right) + n_+^{LS}\, \rho_+\left(\overrightarrow{x}(o)\right)} = z\left(\overrightarrow{x}(o)\right)
\end{aligned}$$

The *ExtRaTrees* behave like a slightly biased Bayes' classifier, with a high variance (it converges towards a minimum @ 100 trees).

Outline

Detecting human silhouettes for video-surveillance I

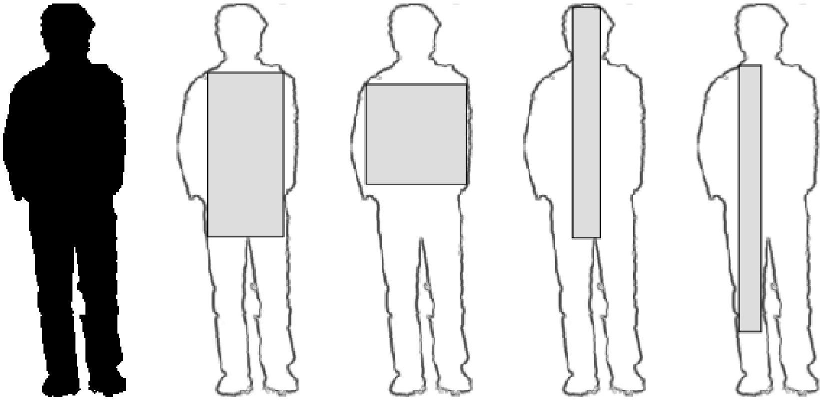
BGS [2], connected components, silhouettes classification [1, 3] :



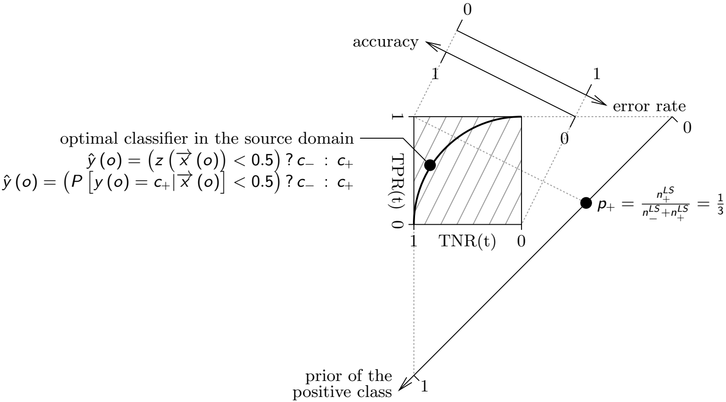
Silhouettes may present huge defects :

Detecting human silhouettes for video-surveillance II

For robustness against defects, silhouettes are analyzed by parts (overlapping maximal axis-aligned rectangles). We discriminate between the classes "part of a human silhouette" (c_+) and the "part of a non-human silhouette" (c_-) [1, 6].

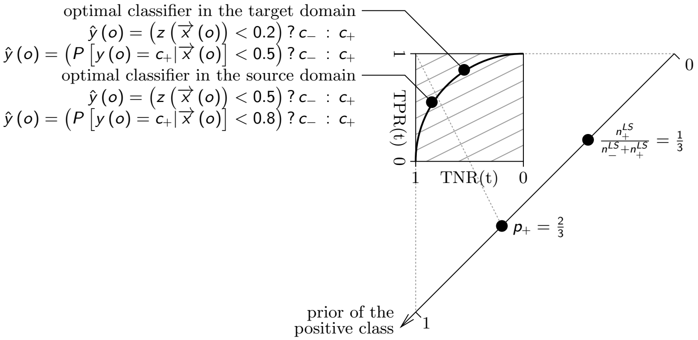


- Let t denote the decision threshold.
- The accuracy is $Q(t) = p_- \, TNR(t) + p_+ \, TPR(t)$.
- Minimizing the error rate (Bayes) $\Leftrightarrow$ maximizing the accuracy.



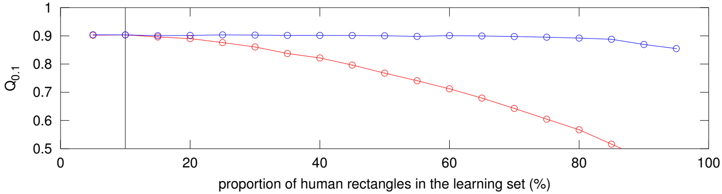
The accuracy and the ROC space II

- Let t denote the decision threshold.
- The accuracy is $Q(t) = p_- \, TNR(t) + p_+ \, TPR(t)$.
- Minimizing the error rate (Bayes) $\Leftrightarrow$ maximizing the accuracy.



Results with $p_+ = 0.1$

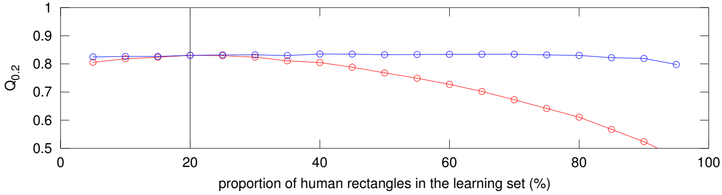
"part of a non-human silhouette" / "part of a human silhouette" :



- blue curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{p_- \; n_+^{LS}}{p_+ \; n_-^{LS} + p_- \; n_+^{LS}} \right) ? \, c_- \; : \; c_+$
- red curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{1}{2} \right) ? \, c_- \; : \; c_+$

Results with $p_+ = 0.2$

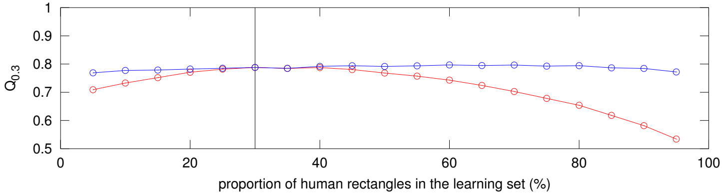
"part of a non-human silhouette" / "part of a human silhouette" :



- blue curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{p_- \, n_+^{LS}}{p_+ \, n_-^{LS} + p_- \, n_+^{LS}} \right) ?\, c_- \; : \; c_+$
- red curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{1}{2} \right) ?\, c_- \; : \; c_+$

Results with $p_+ = 0.3$

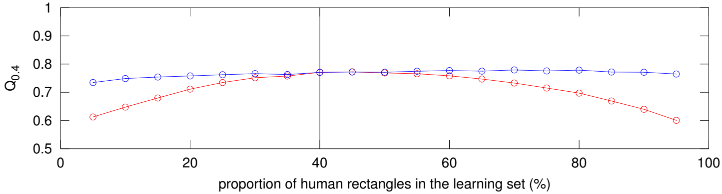
"part of a non-human silhouette" / "part of a human silhouette" :



- blue curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{p_- \ n_+^{LS}}{p_+ \ n_-^{LS} + p_- \ n_+^{LS}} \right) ?\, c_- \ : \ c_+$
- red curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{1}{2} \right) ?\, c_- \ : \ c_+$

Results with $p_+ = 0.4$

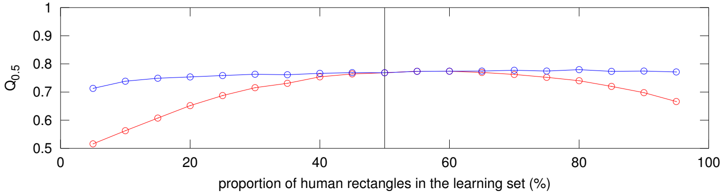
"part of a non-human silhouette" / "part of a human silhouette" :



- blue curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{p_- \; n_+^{LS}}{p_+ \; n_-^{LS} + p_- \; n_+^{LS}} \right) ? \, c_- \; : \; c_+$
- red curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{1}{2} \right) ? \, c_- \; : \; c_+$

Results with $p_+ = 0.5$

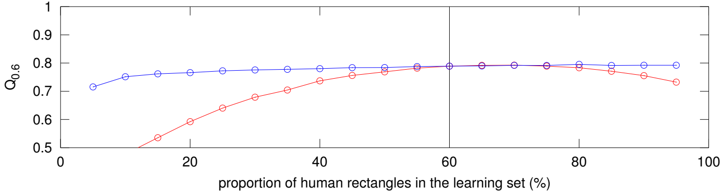
"part of a non-human silhouette" / "part of a human silhouette" :



- blue curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{p_- \, n_+^{LS}}{p_+ \, n_-^{LS} + p_- \, n_+^{LS}} \right) ? \, c_- \ : \ c_+$
- red curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{1}{2} \right) ? \, c_- \ : \ c_+$

Results with $p_+ = 0.6$

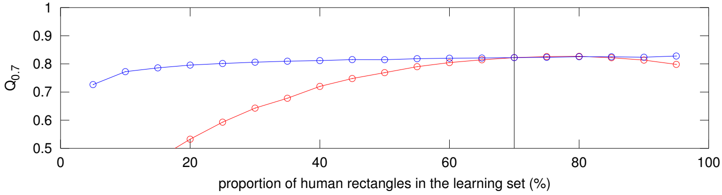
"part of a non-human silhouette" / "part of a human silhouette" :



- blue curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{p_- \; n_+^{LS}}{p_+ \; n_-^{LS} + p_- \; n_+^{LS}} \right) ? \, c_- \; : \; c_+$
- red curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{1}{2} \right) ? \, c_- \; : \; c_+$

Results with $p_+ = 0.7$

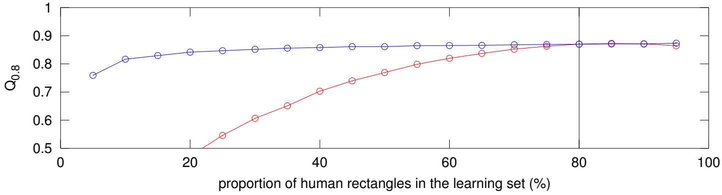
"part of a non-human silhouette" / "part of a human silhouette" :



- blue curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{p_- \, n_+^{LS}}{p_+ \, n_-^{LS} + p_- \, n_+^{LS}} \right) ? \, c_- \; : \; c_+$
- red curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{1}{2} \right) ? \, c_- \; : \; c_+$

Results with $p_+ = 0.8$

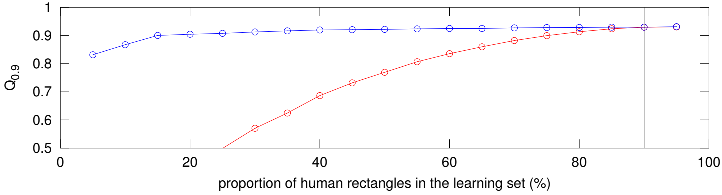
"part of a non-human silhouette" / "part of a human silhouette" :



- blue curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{p_- \, n_+^{LS}}{p_+ \, n_-^{LS} + p_- \, n_+^{LS}} \right) ? \, c_- \, : \, c_+$
- red curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{1}{2} \right) ? \, c_- \, : \, c_+$

Results with $p_+ = 0.9$

"part of a non-human silhouette" / "part of a human silhouette" :



- blue curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{p_- \, n_+^{LS}}{p_+ \, n_-^{LS} + p_- \, n_+^{LS}} \right) ? \, c_- \; : \; c_+$
- red curve : $\hat{y}(o) = \left(z\left(\overrightarrow{x}(o)\right) < \frac{1}{2} \right) ? \, c_- \; : \; c_+$

Outline

Conclusion

- In practice, the *ExtRaTrees* can be considered as an approximation of Bayes' classifier.
- With Bayes' classifier, when choosing the priors to populate the learning set, we can focus on the relative complexity of the underlying probability density functions (most often leading to balanced datasets). But we have to adapt the decision threshold ; the optimal one can be determined theoretically.
- It follows that we have a way to tackle problems in which the priors in the context of use of the classifier are unknown at learning time or change continuously. This is a kind of on-the-fly domain adaptation.
- In this presentation, we have assumed that, for each class, the *pdfs* in the source domain and in the target domain are identical. That is a more challenging domain adaptation problem. Please read [5] for solution to that problem !

Outline

O. Barnich, S. Jodogne, and M. Van Droogenbroeck.
Robust analysis of silhouettes by morphological size distributions.
In *Advances Concepts for Intelligent Vision Systems (ACIVS)*, volume 4179 of *Lecture Notes on Computer Science*, pages 734–745. Springer, September 2006.

O. Barnich and M. Van Droogenbroeck.
ViBe : A universal background subtraction algorithm for video sequences.
IEEE Transactions on Image Processing, 20(6) :1709–1724, June 2011.

- S. Piérard, A. Lejeune, and M. Van Droogenbroeck.
  3D information is valuable for the detection of humans in video streams.
  In *Proceedings of 3D Stereo MEDIA*, Liège, Belgium, December 2010.

- S. Piérard, A. Lejeune, and M. Van Droogenbroeck.
  A probabilistic pixel-based approach to detect humans in video streams.
  In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 921–924, Prague, Czech Republic, May 2011.

S. Piérard, A. Marcos Alvarez, A. Lejeune, and M. Van Droogenbroeck.
On-the-fly domain adaptation of binary classifiers.
In *Belgian-Dutch Conference on Machine Learning (BENELEARN)*, Brussels, Belgium, June 2014.

M. Van Droogenbroeck and S. Piérard.
Object descriptors based on a list of rectangles : method and algorithm.
In *International Symposium on Mathematical Morhology (ISMM)*, volume 6671 of *Lecture Notes on Computer Science*, pages 155–165, Ispra, Italy, 2011. Springer.