Classifiers, Learning

Tomáš Svoboda and Matěj Hoffmann
thanks to Daniel Novák and Filip Železný, Ondřej Drbohlav

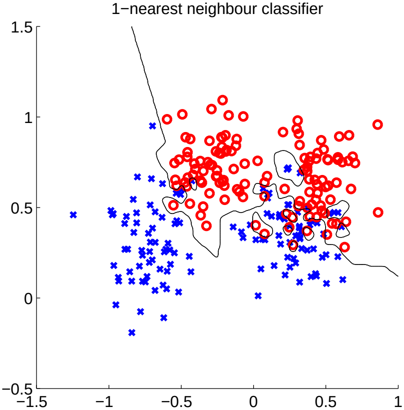
Vision for Robots and Autonomous Systems, Center for Machine Perception
Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University in Prague

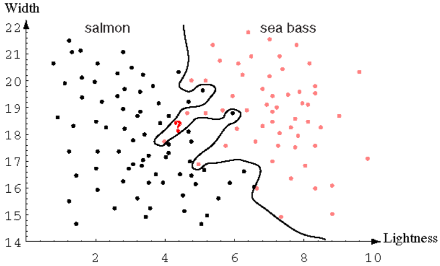
May 13, 2019

K-Nearest neighbors classification

For a query $\vec{x}$:

- Find K nearest $\vec{x}$ from the tranining (labeled) data.
- Classify to the class with the most exemplars in the set above.





$K-$ Nearest Neighbor and Bayes

Assume data:

- N points $\vec{x}$ in total.
- N_j points in s_j class. Hence, $\sum_j N_j = N$.

We want classify $\vec{x}$. We draw a sphere centered at $\vec{x}$ containing K points irrespective of class. V is the volume of this sphere. $P(s_j|\vec{x}) = ?$

$$P(s_j|\vec{x}) = \frac{P(\vec{x}|s_j)P(s_j)}{P(\vec{x})}$$

$$P(s_j) = \frac{N_j}{N}$$

$$P(\vec{x}) = \frac{K}{NV}$$

$$P(\vec{x}|s_j) = \frac{K_j}{N_j V}$$

$$P(s_j|\vec{x}) = \frac{P(\vec{x}|s_j)P(s_j)}{P(\vec{x})} = \frac{K_j}{K}$$

$K-$ Nearest Neighbor and Bayes

Assume data:

- N points $\vec{x}$ in total.
- N_j points in s_j class. Hence, $\sum_j N_j = N$.

We want classify $\vec{x}$. We draw a sphere centered at $\vec{x}$ containing K points irrespective of class. V is the volume of this sphere. $P(s_j|\vec{x}) =?$

$$P(s_j|\vec{x}) = \frac{P(\vec{x}|s_j)P(s_j)}{P(\vec{x})}$$

$$P(s_j) = \frac{N_j}{N}$$

$$P(\vec{x}) = \frac{K}{NV}$$

$$P(\vec{x}|s_j) = \frac{K_j}{N_j V}$$

$$P(s_j|\vec{x}) = \frac{P(\vec{x}|s_j)P(s_j)}{P(\vec{x})} = \frac{K_j}{K}$$

$K-$ Nearest Neighbor and Bayes

Assume data:

- N points $\vec{x}$ in total.
- N_j points in s_j class. Hence, $\sum_j N_j = N$.

We want classify $\vec{x}$. We draw a sphere centered at $\vec{x}$ containing K points irrespective of class. V is the volume of this sphere. $P(s_j|\vec{x}) = ?$

$$P(s_j|\vec{x}) = \frac{P(\vec{x}|s_j)P(s_j)}{P(\vec{x})}$$

$$P(s_j) = \frac{N_j}{N}$$

$$P(\vec{x}) = \frac{K}{NV}$$

$$P(\vec{x}|s_j) = \frac{K_j}{N_j V}$$

$$P(s_j|\vec{x}) = \frac{P(\vec{x}|s_j)P(s_j)}{P(\vec{x})} = \frac{K_j}{K}$$

$K-$ Nearest Neighbor and Bayes

Assume data:

- N points $\vec{x}$ in total.
- N_j points in s_j class. Hence, $\sum_j N_j = N$.

We want classify $\vec{x}$. We draw a sphere centered at $\vec{x}$ containing K points irrespective of class. V is the volume of this sphere. $P(s_j|\vec{x}) = ?$

$$P(s_j|\vec{x}) = \frac{P(\vec{x}|s_j)P(s_j)}{P(\vec{x})}$$

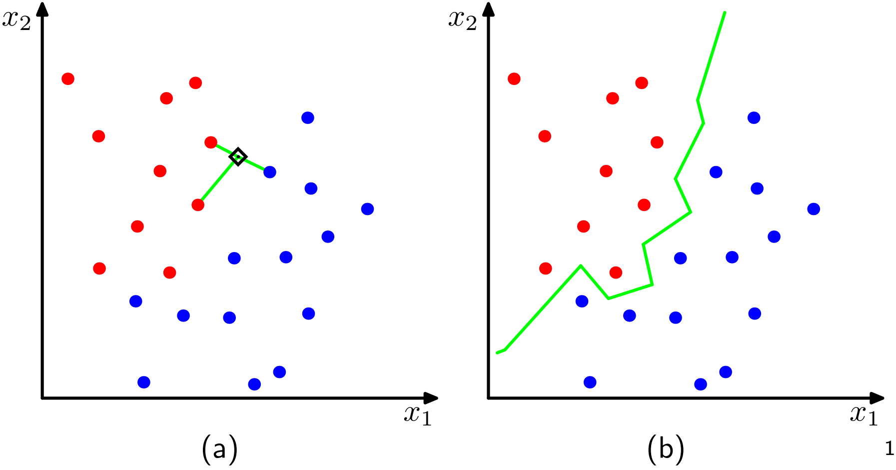
$$P(s_j) = \frac{N_j}{N}$$

$$P(\vec{x}) = \frac{K}{NV}$$

$$P(\vec{x}|s_j) = \frac{K_j}{N_j V}$$

$$P(s_j|\vec{x}) = \frac{P(\vec{x}|s_j)P(s_j)}{P(\vec{x})} = \frac{K_j}{K}$$

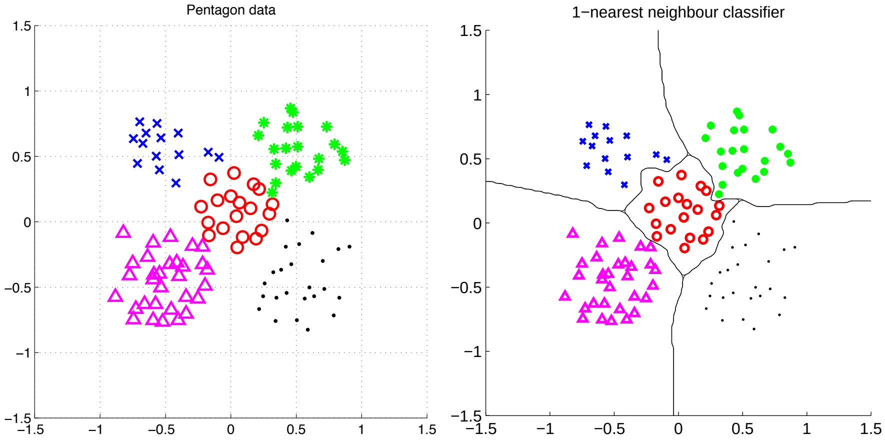
NN classification example



[1]

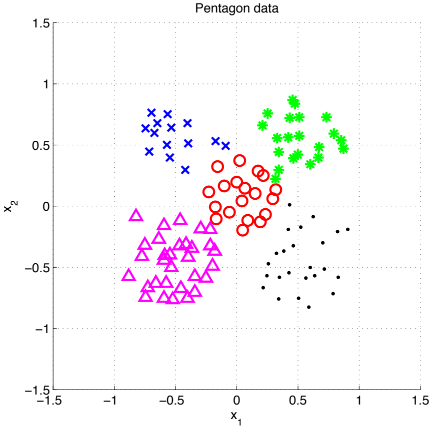
[1]Figs from [1]

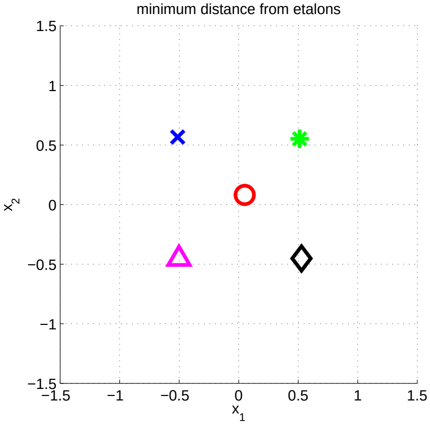
NN classification example



Metrics for NN classification

Etalon based classification

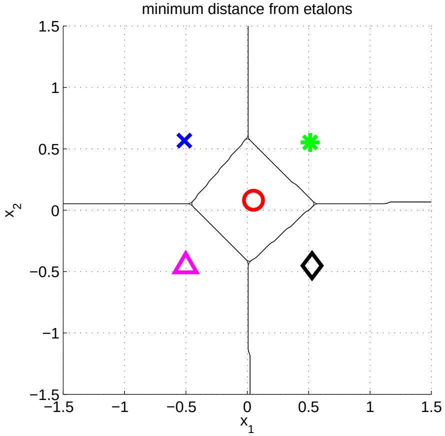




Represent $\vec{x}$ by etalon , $\vec{e}_s$ per each class $s \in S$

Separate etalons

$$s^* = \arg\min_{s \in S} (||\vec{x} - \vec{e}_s||^2 + o_s)$$

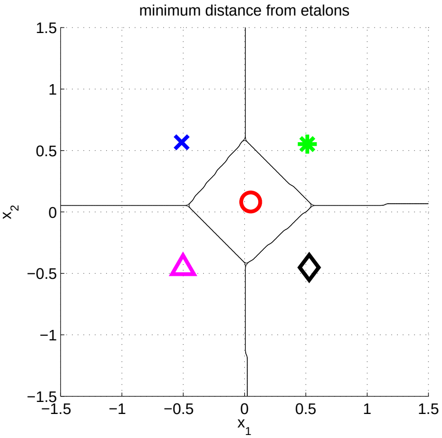


What etalons?

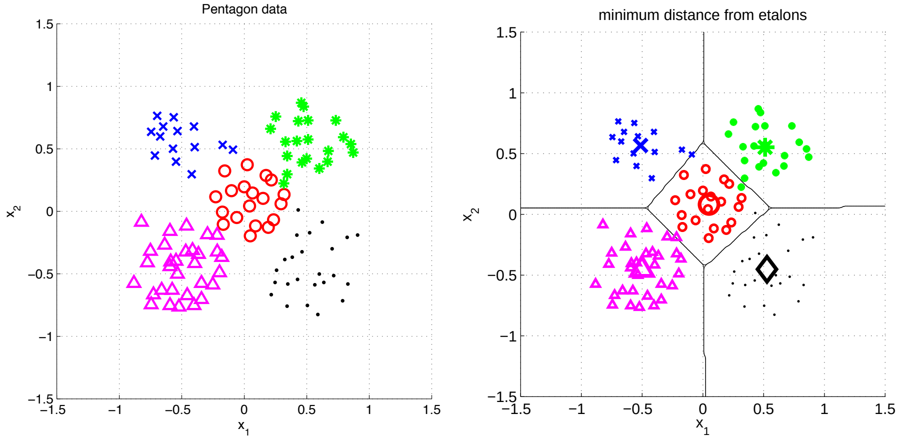
If $\mathcal{N}(\vec{x}|\vec{\mu},\Sigma)$; all classes same covariance matrices, then

$$\vec{e}_s \stackrel{\text{def}}{=} \vec{\mu}_s = \frac{1}{|\mathcal{X}^s|}\sum_{i\in\mathcal{X}^s}\vec{x}_i^s$$

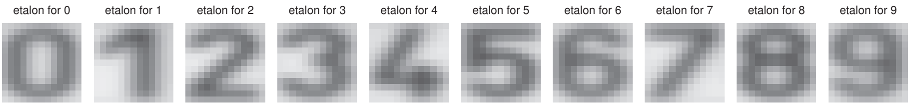
and separating hyperplanes halve distances between pairs.



Etalon based classification, $\vec{e}_s = \vec{\mu}_s$

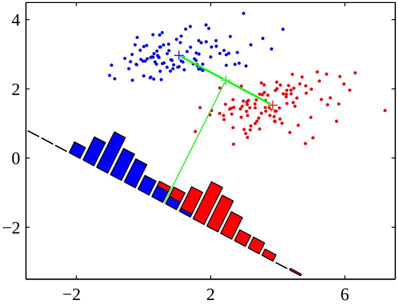


Digit recognition - etalons $\vec{e}_s = \vec{\mu}_s$



Figures from [5]

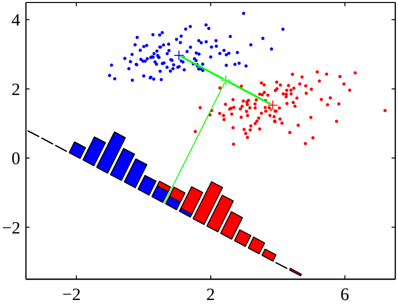
Better etalons – Fischer linear discriminant

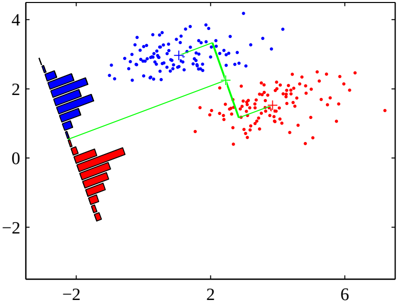


- Dimensionality reduction
- Maximize distance between means, ...
- ...and minimize within class variance. (minimize overlap)

Figures from [1]

Better etalons – Fischer linear discriminant

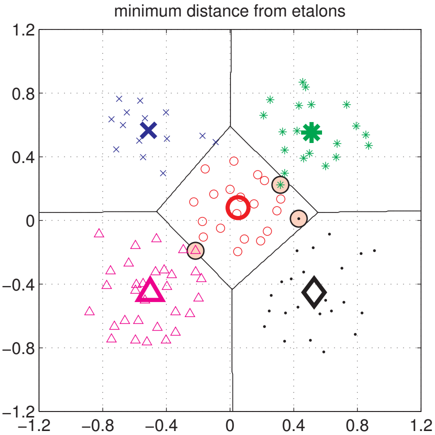




- Dimensionality reduction
- Maximize distance between means, ...
- ...and minimize within class variance. (minimize overlap)

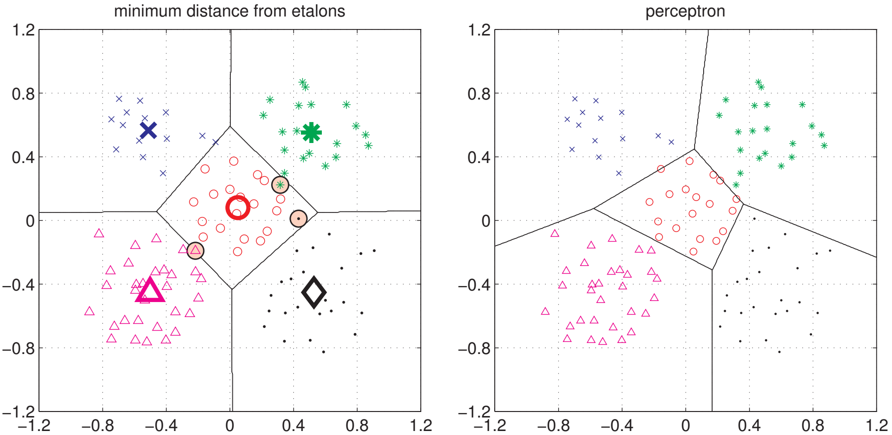
Figures from [1]

Better etalons?



Figures from [5]

Better etalons?



Figures from [5]

Etalon classifier – Linear classifier

$$s^* = \arg\min_{s\in S}\left(\|\vec{x} - \vec{e}_s\|^2 + o_s\right) = \arg\min_{s\in S}(\vec{x}^\top \vec{x} - 2\,\vec{e}_s^\top \vec{x} + \vec{e}_s^\top \vec{e}_s + o_s) =$$

$$= \arg\min_{s\in S}\left(\vec{x}^\top \vec{x} - 2\left(\vec{e}_s^\top \vec{x} - \frac{1}{2}(\vec{e}_s^\top \vec{e}_s + o_s)\right)\right) =$$

$$= \arg\min_{s\in S}\left(\vec{x}^\top \vec{x} - 2\left(\vec{e}_s^\top \vec{x} + b_s\right)\right) =$$

$$= \boxed{\arg\max_{s\in S}(\vec{e}_s^\top \vec{x} + b_s)} = \arg\max_{s\in S} g_s(\vec{x}). \qquad b_s = -\frac{1}{2}(\vec{e}_s^\top \vec{e}_s + o_s)$$

Linear function (plus offset)

$$g_s(\mathbf{x}) = \mathbf{w}_s^\top \mathbf{x} + w_{s0}$$

Etalon classifier – Linear classifier

$$s^* = \arg\min_{s \in S} \left(\|\vec{x} - \vec{e}_s\|^2 + o_s \right) = \arg\min_{s \in S} (\vec{x}^\top \vec{x} - 2\, \vec{e}_s^\top \vec{x} + \vec{e}_s^\top \vec{e}_s + o_s) =$$

$$= \arg\min_{s \in S} \left(\vec{x}^\top \vec{x} - 2\left(\vec{e}_s^\top \vec{x} - \frac{1}{2}(\vec{e}_s^\top \vec{e}_s + o_s)\right)\right) =$$

$$= \arg\min_{s \in S} \left(\vec{x}^\top \vec{x} - 2\left(\vec{e}_s^\top \vec{x} + b_s\right)\right) =$$

$$= \boxed{\arg\max_{s \in S} (\vec{e}_s^\top \vec{x} + b_s)} = \arg\max_{s \in S} g_s(\vec{x}). \qquad b_s = -\frac{1}{2}(\vec{e}_s^\top \vec{e}_s + o_s)$$

Linear function (plus offset)

$$g_s(\mathbf{x}) = \mathbf{w}_s^\top \mathbf{x} + w_{s0}$$

Etalon classifier – Linear classifier

$$s^* = \arg\min_{s\in S}\big(\|\vec{x}-\vec{e}_s\|^2 + o_s\big) = \arg\min_{s\in S}(\vec{x}^\top\vec{x} - 2\,\vec{e}_s^\top\vec{x} + \vec{e}_s^\top\vec{e}_s + o_s) =$$

$$= \arg\min_{s\in S}\Big(\vec{x}^\top\vec{x} - 2\,\big(\vec{e}_s^\top\vec{x} - \frac{1}{2}(\vec{e}_s^\top\vec{e}_s + o_s)\big)\Big) =$$

$$= \arg\min_{s\in S}\big(\vec{x}^\top\vec{x} - 2\,(\vec{e}_s^\top\vec{x} + b_s)\big) =$$

$$= \boxed{\arg\max_{s\in S}(\vec{e}_s^\top\vec{x} + b_s)} = \arg\max_{s\in S} g_s(\vec{x}). \qquad b_s = -\frac{1}{2}(\vec{e}_s^\top\vec{e}_s + o_s)$$

Linear function (plus offset)

$$g_s(\mathbf{x}) = \mathbf{w}_s^\top\mathbf{x} + w_{s0}$$

Etalon classifier – Linear classifier

$$
\begin{aligned}
s^* &= \arg\min_{s\in S}\left(\|\vec{x}-\vec{e}_s\|^2 + o_s\right) = \arg\min_{s\in S}(\vec{x}^\top\vec{x} - 2\,\vec{e}_s^\top\vec{x} + \vec{e}_s^\top\vec{e}_s + o_s) = \\
&= \arg\min_{s\in S}\left(\vec{x}^\top\vec{x} - 2\left(\vec{e}_s^\top\vec{x} - \frac{1}{2}(\vec{e}_s^\top\vec{e}_s + o_s)\right)\right) = \\
&= \arg\min_{s\in S}\left(\vec{x}^\top\vec{x} - 2\left(\vec{e}_s^\top\vec{x} + b_s\right)\right) = \\
&= \boxed{\arg\max_{s\in S}(\vec{e}_s^\top\vec{x} + b_s)} = \arg\max_{s\in S} g_s(\vec{x}).
\end{aligned}
$$

$$b_s = -\frac{1}{2}(\vec{e}_s^\top\vec{e}_s + o_s)$$

Linear function (plus offset)

$$g_s(\mathbf{x}) = \mathbf{w}_s^\top\mathbf{x} + w_{s0}$$

Etalon classifier – Linear classifier

$$
\begin{aligned}
s^* &= \arg\min_{s \in S} \left(\|\vec{x} - \vec{e}_s\|^2 + o_s \right) = \arg\min_{s \in S} (\vec{x}^\top \vec{x} - 2\,\vec{e}_s^\top \vec{x} + \vec{e}_s^\top \vec{e}_s + o_s) = \\
&= \arg\min_{s \in S} \left(\vec{x}^\top \vec{x} - 2\left(\vec{e}_s^\top \vec{x} - \frac{1}{2}(\vec{e}_s^\top \vec{e}_s + o_s) \right) \right) = \\
&= \arg\min_{s \in S} \left(\vec{x}^\top \vec{x} - 2\left(\vec{e}_s^\top \vec{x} + b_s \right) \right) = \\
&= \boxed{\arg\max_{s \in S} (\vec{e}_s^\top \vec{x} + b_s)} = \arg\max_{s \in S} g_s(\vec{x}).
\end{aligned}
$$

$$
b_s = -\frac{1}{2}(\vec{e}_s^\top \vec{e}_s + o_s)
$$

Linear function (plus offset)

$$
g_s(\mathbf{x}) = \mathbf{w}_s^\top \mathbf{x} + w_{s0}
$$

Etalon classifier – Linear classifier

$$
\begin{aligned}
s^* &= \arg\min_{s \in S} \left(\|\vec{x} - \vec{e}_s\|^2 + o_s \right) = \arg\min_{s \in S} (\vec{x}^\top \vec{x} - 2\, \vec{e}_s^\top \vec{x} + \vec{e}_s^\top \vec{e}_s + o_s) = \\
&= \arg\min_{s \in S} \left(\vec{x}^\top \vec{x} - 2 \left(\vec{e}_s^\top \vec{x} - \frac{1}{2} (\vec{e}_s^\top \vec{e}_s + o_s) \right) \right) = \\
&= \arg\min_{s \in S} \left(\vec{x}^\top \vec{x} - 2 \left(\vec{e}_s^\top \vec{x} + b_s \right) \right) = \\
&= \boxed{\arg\max_{s \in S} (\vec{e}_s^\top \vec{x} + b_s)} = \arg\max_{s \in S} g_s(\vec{x}).
\end{aligned}
$$

$$b_s = -\frac{1}{2} (\vec{e}_s^\top \vec{e}_s + o_s)$$

Linear function (plus offset)

$$g_s(\mathbf{x}) = \mathbf{w}_s^\top \mathbf{x} + w_{s0}$$

Learning and decision

Learning stage - learning models/function/parameters from data.

Decision stage - decide about a query $\vec{x}$.

What to learn?

- Generative model : Learn $P(\vec{x}, s)$. Decide by computing $P(s|\vec{x})$.
- Discriminative model : Learn $P(s|\vec{x})$
- Discriminant function : Learn $g(\vec{x})$ which maps $\vec{x}$ directly into class labels.

Linear discriminant function - two class case

$$g(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + w_0$$

Decide s_1 if $g(\mathbf{x}) > 0$ and s_2 if $g(\mathbf{x}) < 0$

Figure from [2]

Linear discriminant function - two class case

$$g(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + w_0$$

Decide s_1 if $g(\mathbf{x}) > 0$ and s_2 if $g(\mathbf{x}) < 0$

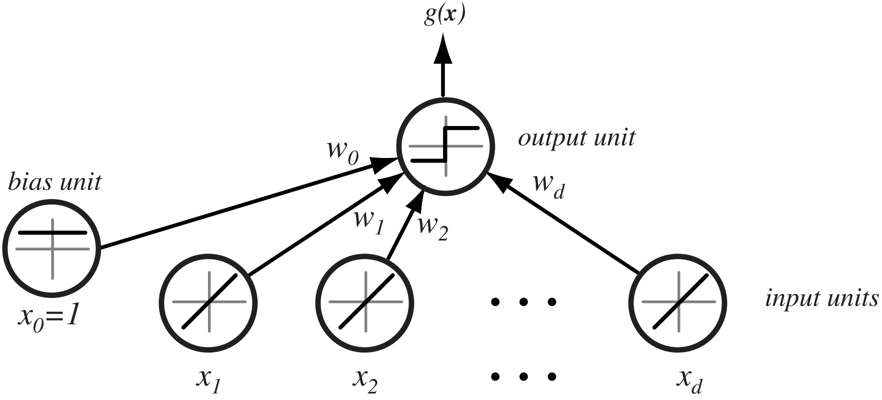


Figure from [2]

Separating hyperplane

$$\mathbf{w}^\top \mathbf{x}_1 + w_0 = \mathbf{w}^\top \mathbf{x}_2 + w_0$$
$$\mathbf{w}^\top (\mathbf{x}_1 - \mathbf{x}_2) = 0$$

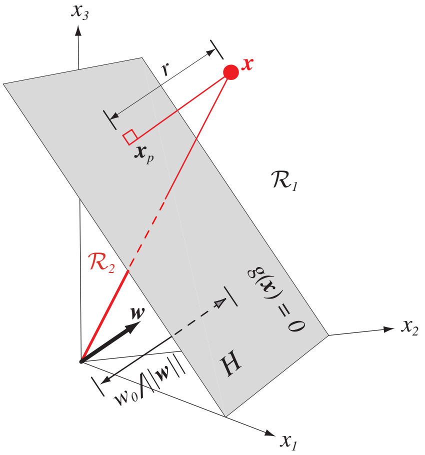


Figure from [2]

Multiclass case

Two classes set-up

$|S| = 2$, i.e. two states (typically also classes)

$$g(\mathbf{x}) = \begin{cases} s = 1\,, & \text{if} \quad \mathbf{w}^\top \mathbf{x} + w_0 > 0\,, \\ s = -1\,, & \text{if} \quad \mathbf{w}^\top \mathbf{x} + w_0 < 0\,. \end{cases}$$

$\mathbf{x}'_j = s_j \begin{bmatrix} 1 \\ \mathbf{x}_j \end{bmatrix}$, $\mathbf{w}' = \begin{bmatrix} w_0 \\ \mathbf{w} \end{bmatrix}$

for all $\mathbf{x}'$

$$\mathbf{w}'^\top \mathbf{x}' > 0$$

drop the dashes to avoid notation clutter.

Two classes set-up

$|S| = 2$, i.e. two states (typically also classes)

$$g(\mathbf{x}) = \begin{cases} s = 1\,, & \text{if} \quad \mathbf{w}^\top \mathbf{x} + w_0 > 0\,, \\ s = -1\,, & \text{if} \quad \mathbf{w}^\top \mathbf{x} + w_0 < 0\,. \end{cases}$$

$\mathbf{x}'_j = s_j \begin{bmatrix} 1 \\ \mathbf{x}_j \end{bmatrix}$, $\mathbf{w}' = \begin{bmatrix} w_0 \\ \mathbf{w} \end{bmatrix}$

for all $\mathbf{x}'$

$$\mathbf{w}'^\top \mathbf{x}' > 0$$

drop the dashes to avoid notation clutter.

Two classes set-up

$|S| = 2$, i.e. two states (typically also classes)

$$g(\mathbf{x}) = \begin{cases} s = 1\,, & \text{if} \quad \mathbf{w}^\top \mathbf{x} + w_0 > 0\,, \\ s = -1\,, & \text{if} \quad \mathbf{w}^\top \mathbf{x} + w_0 < 0\,. \end{cases}$$

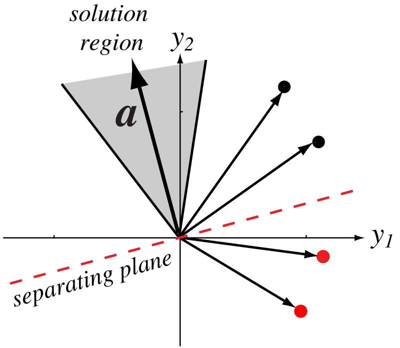
$\mathbf{x}'_j = s_j \begin{bmatrix} 1 \\ \mathbf{x}_j \end{bmatrix}$, $\mathbf{w}' = \begin{bmatrix} w_0 \\ \mathbf{w} \end{bmatrix}$

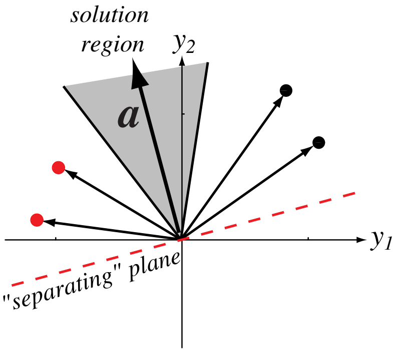
for all $\mathbf{x}'$

$$\mathbf{w}'^\top \mathbf{x}' > 0$$

drop the dashes to avoid notation clutter.

Solution (graphically)





Substitute $\mathbf{a} \leftarrow \mathbf{w}$ and $y_1, y_2 \leftarrow x_1, x_2$

Figure from [2]

Learning $\mathbf{w}$, gradient descent

A criterion to be minimized $J(\mathbf{w})$

Initialize $\mathbf{w}$, threshold θ, learning rate α

$k \leftarrow 0$

repeat

  $k \leftarrow k + 1$

  $\mathbf{w} \leftarrow \mathbf{w} - \alpha(k)\nabla J(\mathbf{w})$

until $|\alpha(k)\nabla J(\mathbf{w})| < \theta$

return $\mathbf{w}$

Learning **w** - Perceptron criterion

Goal: Find a weight vector $\mathbf{w} \in \Re^{D+1}$ (original feature space dimensionality is D) such that:

$$\mathbf{w}^\top \mathbf{x}_j > 0 \qquad (\forall j \in \{1, 2, ..., m\})$$

(Perceptron) Criterion to be minimized:

$$J(\mathbf{w}) = \sum_{\mathbf{x} \in \mathcal{X}} -\mathbf{w}^\top \mathbf{x}$$

where $\mathcal{X}$ is a set of missclassified $\mathbf{x}$.

$$\nabla J(\mathbf{w}) = \sum_{\mathbf{x} \in \mathcal{X}} -\mathbf{x}$$

Learning $\mathbf{w}$ - Perceptron criterion

Goal: Find a weight vector $\mathbf{w} \in \Re^{D+1}$ (original feature space dimensionality is D) such that:

$$\mathbf{w}^\top \mathbf{x}_j > 0 \qquad (\forall j \in \{1, 2, ..., m\})$$

(Perceptron) Criterion to be minimized:

$$J(\mathbf{w}) = \sum_{\mathbf{x} \in \mathcal{X}} -\mathbf{w}^\top \mathbf{x}$$

where $\mathcal{X}$ is a set of missclassified $\mathbf{x}$.

$$\nabla J(\mathbf{w}) = \sum_{\mathbf{x} \in \mathcal{X}} -\mathbf{x}$$

Learning **w** - Perceptron criterion

Goal: Find a weight vector $\mathbf{w} \in \Re^{D+1}$ (original feature space dimensionality is D) such that:

$$\mathbf{w}^\top \mathbf{x}_j > 0 \qquad (\forall j \in \{1, 2, ..., m\})$$

(Perceptron) Criterion to be minimized:

$$J(\mathbf{w}) = \sum_{\mathbf{x} \in \mathcal{X}} -\mathbf{w}^\top \mathbf{x}$$

where $\mathcal{X}$ is a set of missclassified **x**.

$$\nabla J(\mathbf{w}) = \sum_{\mathbf{x} \in \mathcal{X}} -\mathbf{x}$$

(Batch) Perceptron algorithm

Initialize **w**, threshold θ, learning rate α
$k \leftarrow 0$
repeat
  $k \leftarrow k + 1$
  $\mathbf{w} \leftarrow \mathbf{w} + \alpha(k) \sum_{\mathbf{x} \in \mathcal{X}(k)} \mathbf{x}$
until $|\alpha(k) \sum_{\mathbf{x} \in \mathcal{X}(k)} \mathbf{x}| < \theta$
return **w**

Fixed-increment single-sample Perceptron

n patterns/samples, we are looping over all patterns repeatedly

Initialize $\mathbf{w}$

$k \leftarrow 0$

repeat

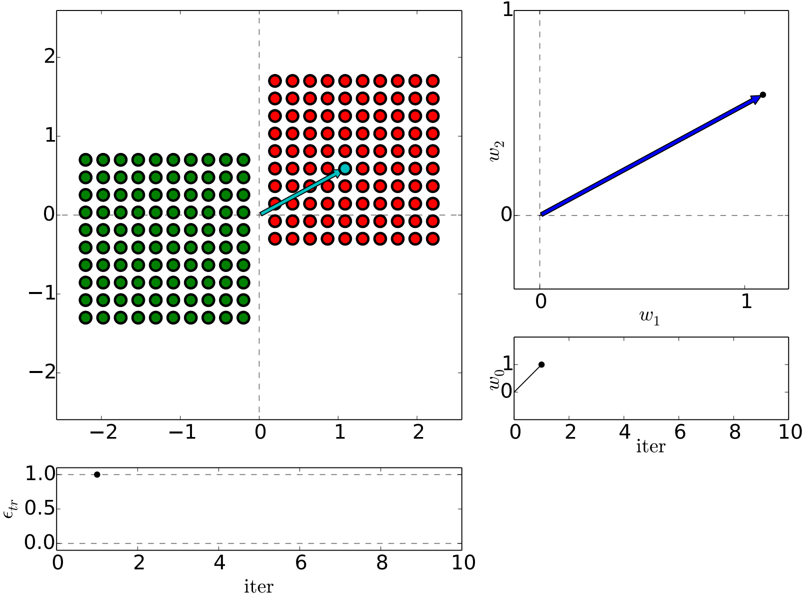
$k \leftarrow (k+1) \bmod n$

if $\mathbf{x}^k$ missclassified, **then** $\mathbf{w} \leftarrow \mathbf{w} + \mathbf{x}^k$

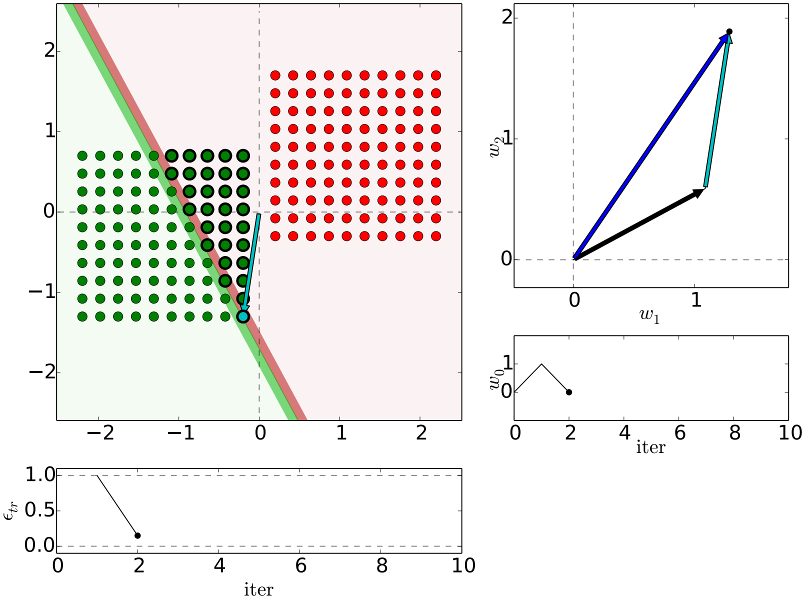
until all $\mathbf{x}$ correctly classified

return $\mathbf{w}$

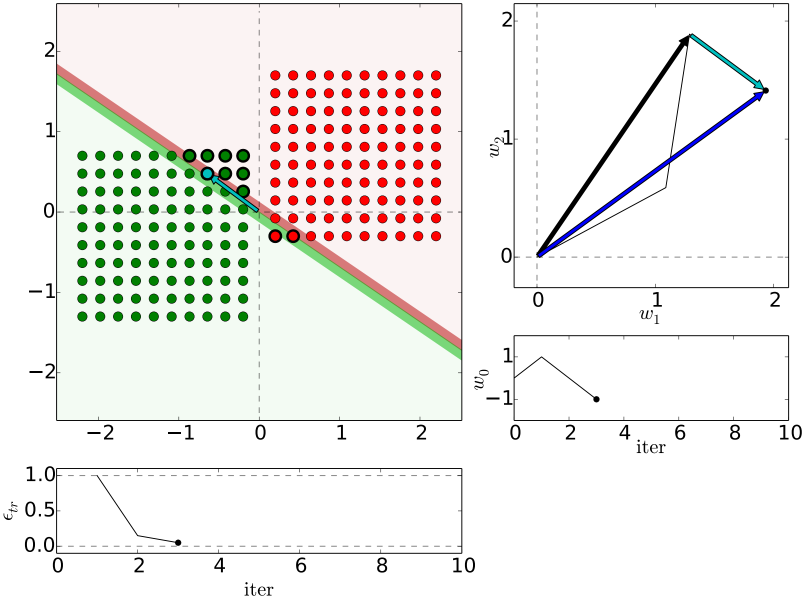
Perceptron iterations/loops



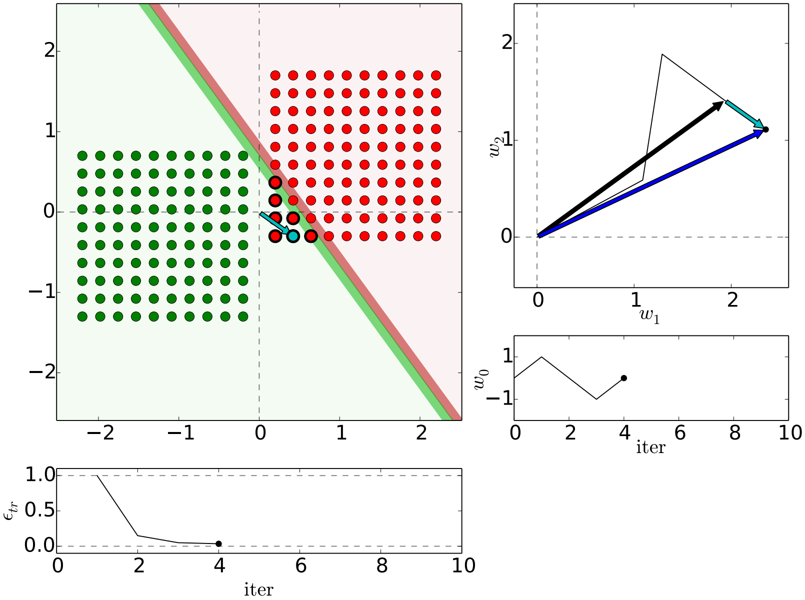
Perceptron iterations/loops



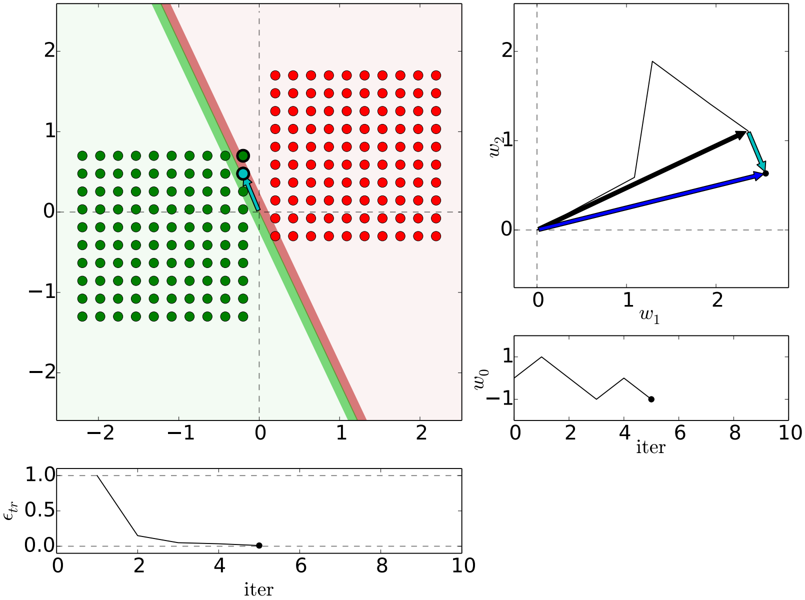
Perceptron iterations/loops



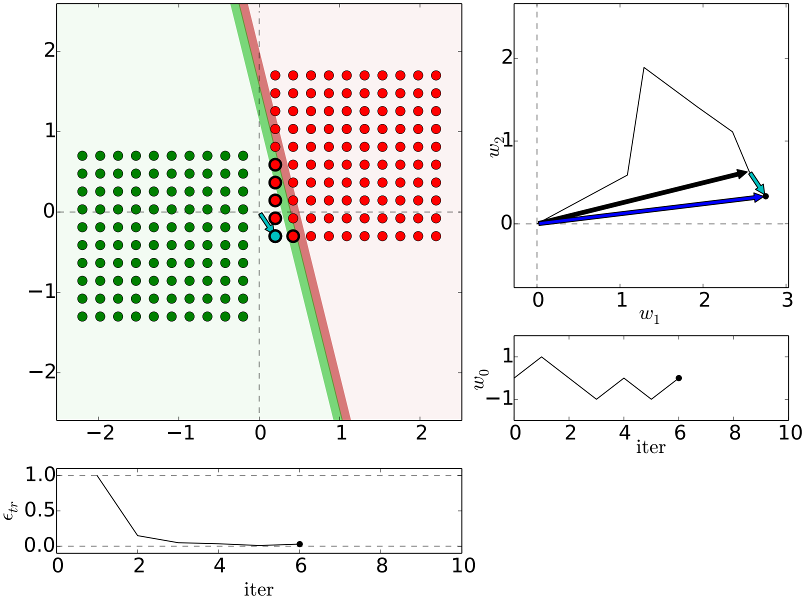
Perceptron iterations/loops



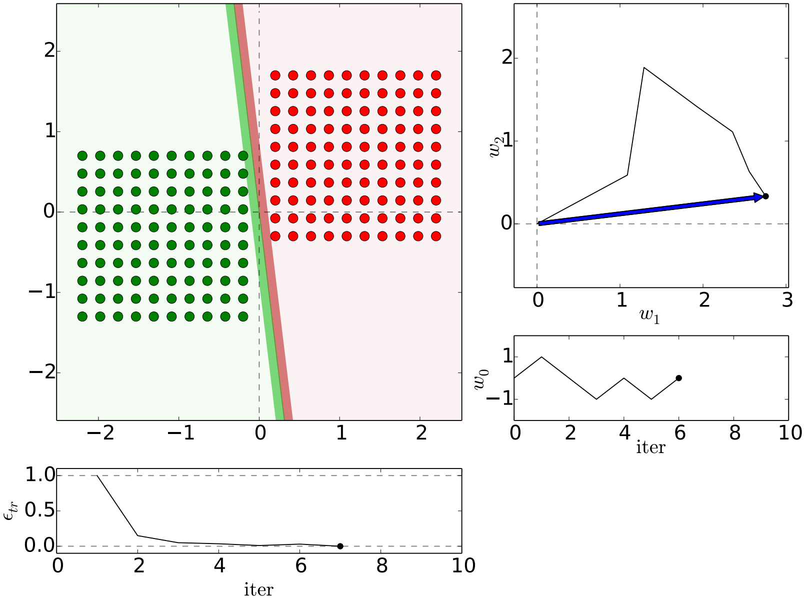
Perceptron iterations/loops



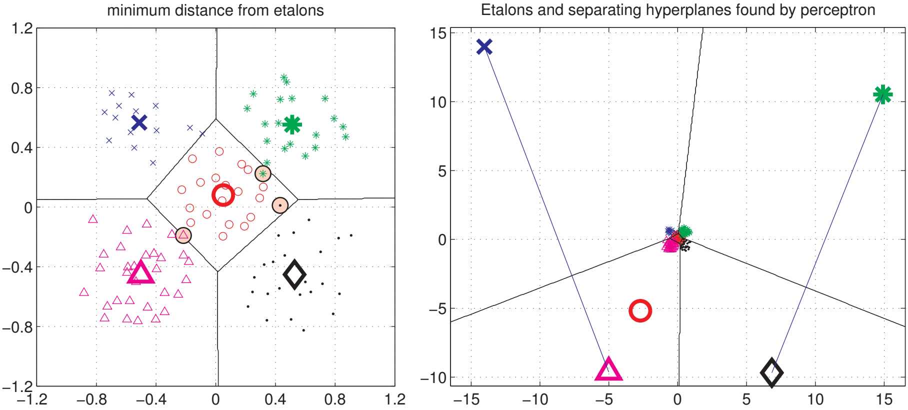
Perceptron iterations/loops



Perceptron iterations/loops

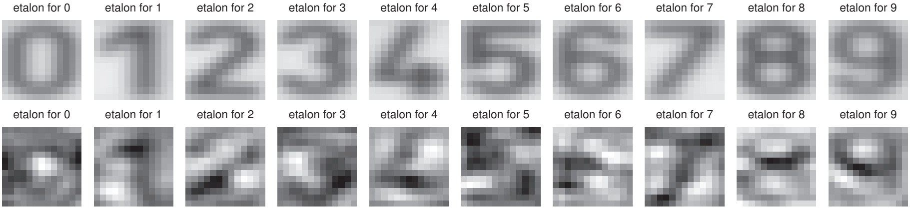


Etalons: means vs found be perceptron



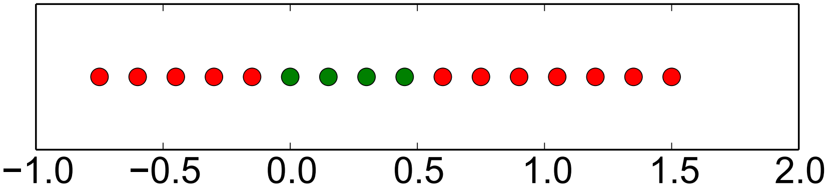
Figures from [5]

Digit recognition - etalons means vs. perceptron



Figures from [5]

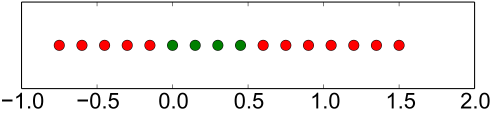
What if not lin separable?

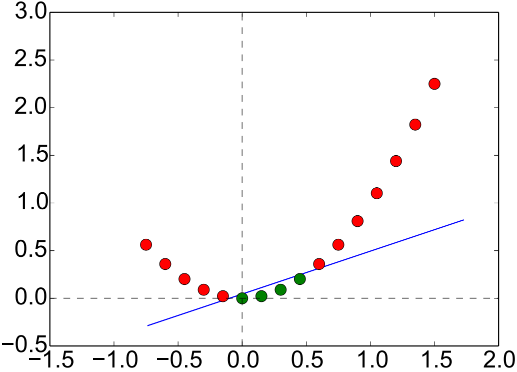


Dimension lifting

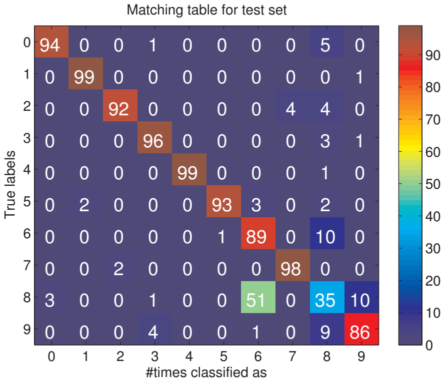
$$\mathbf{x} = [x, x^2]^\top$$

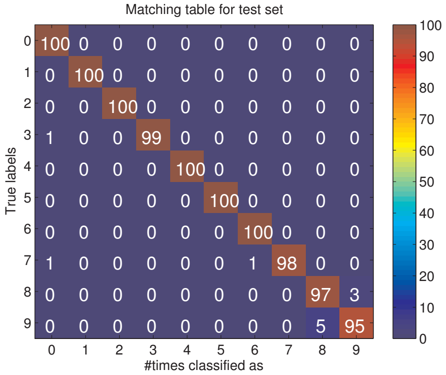
Dimension lifting, $\mathbf{x} = [x, x^2]^\top$





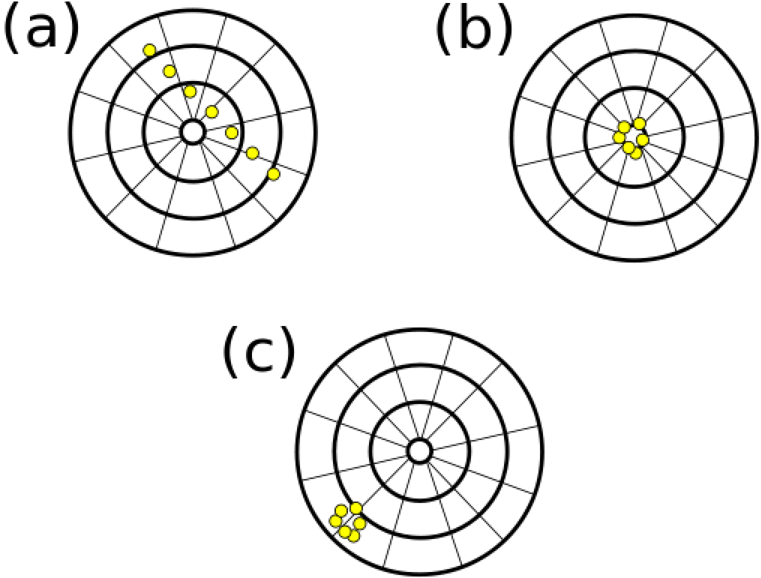
Performance comparison, parameters fixed





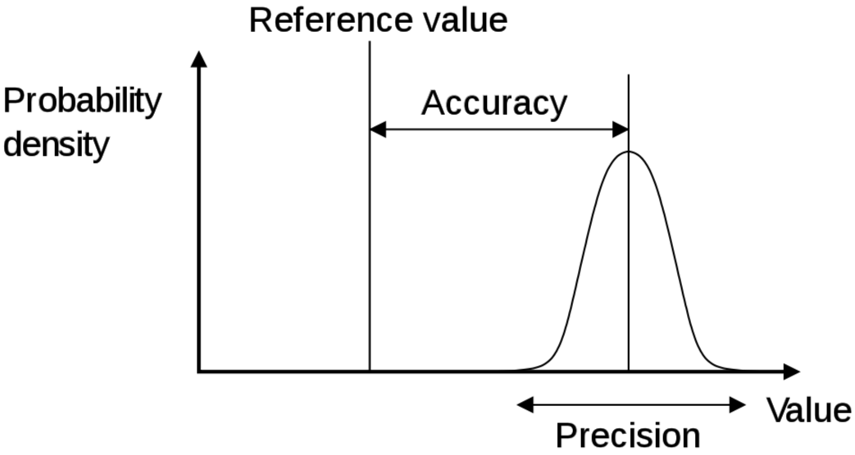
LSQ approach to linear classification

Accuracy vs precision



https://commons.wikimedia.org/wiki/File:Precision_versus_accuracy.svg

Accuracy vs precision



https://en.wikipedia.org/wiki/Accuracy_and_precision

References I

Further reading: Chapter 13 and 14 of [4]. Books [1] and [2] are classical textbooks in the field of pattern recognition and machine learning. Many Matlab figures created with the help of [3]

[1] Christopher M. Bishop.
Pattern Recognition and Machine Learning.
Springer Science+Bussiness Media, New York, NY, 2006.
PDF freely downloadable.

[2] Richard O. Duda, Peter E. Hart, and David G. Stork.
Pattern Classification.
John Wiley & Sons, 2nd edition, 2001.

[3] Votjěch Franc and Václav Hlaváč.
Statistical pattern recognition toolbox.
http://cmp.felk.cvut.cz/cmp/software/stprtool/index.html.

References II

[4] Stuart Russell and Peter Norvig.
Artificial Intelligence: A Modern Approach.
Prentice Hall, 3rd edition, 2010.
http://aima.cs.berkeley.edu/.

[5] Tomáš Svoboda, Jan Kybic, and Hlaváč Václav.
Image Processing, Analysis and Machine Vision — A MATLAB Companion.
Thomson, Toronto, Canada, 1^{st} edition, September 2007.
http://visionbook.felk.cvut.cz/.