Pattern Recognition for Neuroimaging Data

Edinburgh, SPM course

April 2017

C. Phillips, GIGA – *In Silico* Medicine, ULg, Belgium
c.phillips@ulg.ac.be
http://www.giga.ulg.ac.be

Overview

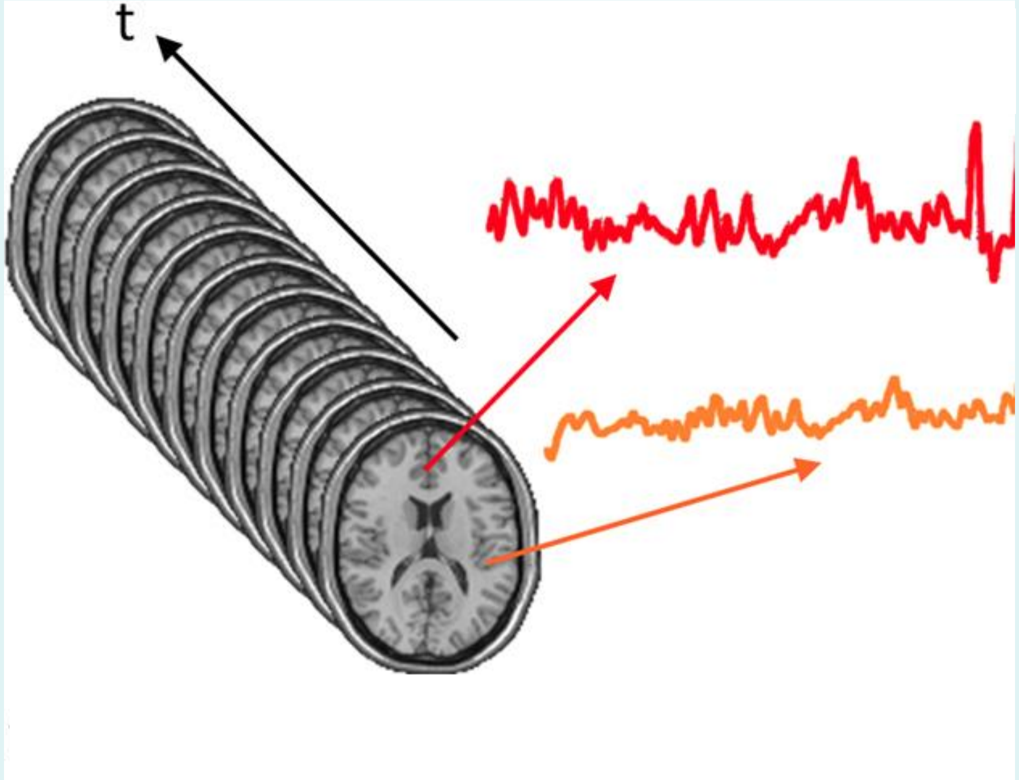
- Introduction
  - Univariate & multivariate approaches
  - Data representation
- Pattern Recognition
  - Machine learning
  - Validation & inference
  - Weight maps & feature selection
  - Applications: groups & fMRI
- Conclusion & Toolboxes

Overview

- Introduction
  - Univariate & multivariate approaches
  - Data representation
- Pattern Recognition
  - Machine learning
  - Validation & inference
  - Weight maps & feature selection
  - Applications: groups & fMRI
- Conclusion & Toolboxes

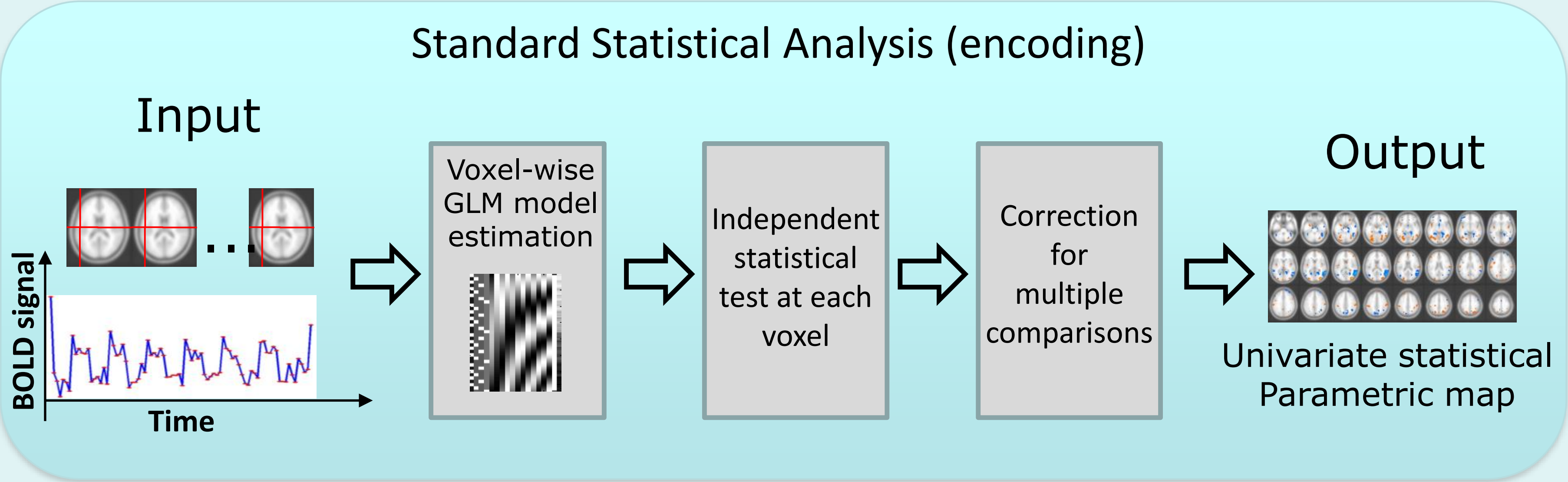
Introduction

fMRI time series = 4D image
= time series of 3D fMRI's
= 3D array of time series.



Univariate vs. multivariate

Standard univariate approach (SPM)



Standard Statistical Analysis (encoding)

Input

Voxel-wise GLM model estimation

Independent statistical test at each voxel

Correction for multiple comparisons

Output

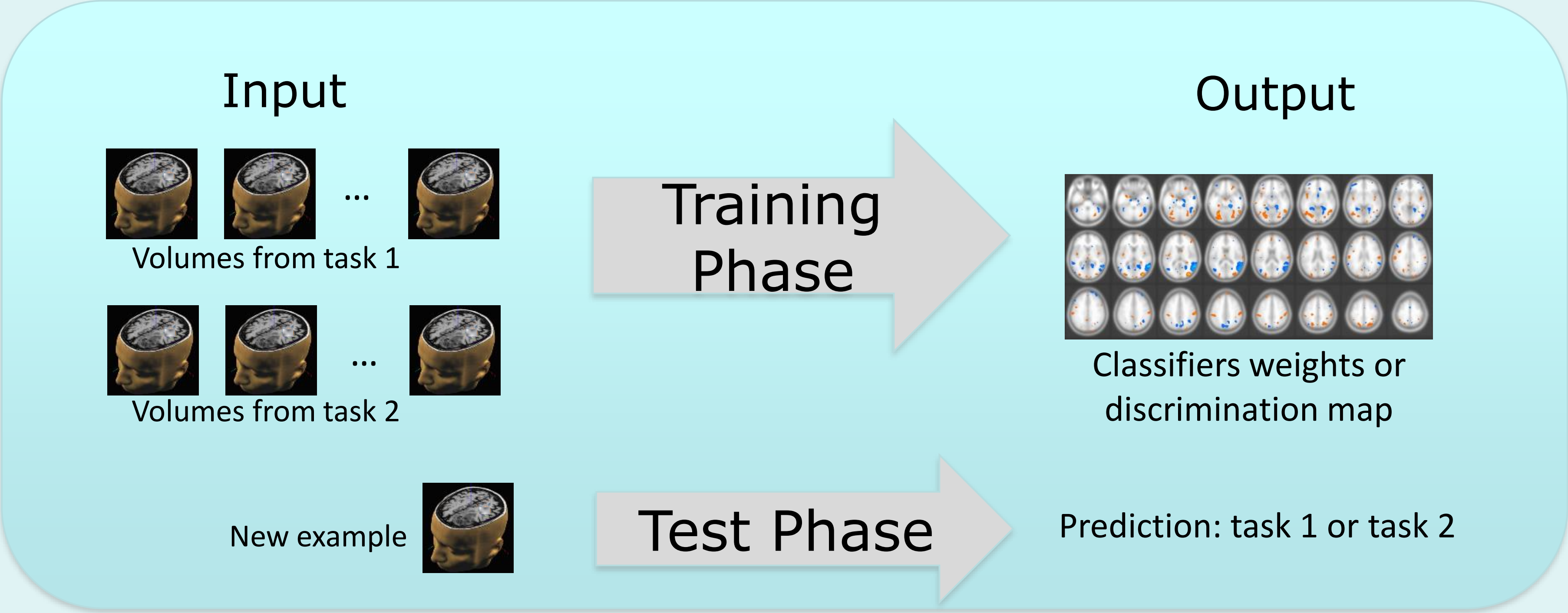
Univariate statistical Parametric map

Find the mapping g from explanatory variable X to observed data Y

$$g: X \rightarrow Y$$

Univariate vs. multivariate

Multivariate approach, aka. "pattern recognition"



Input

…

Volumes from task 1

…

Volumes from task 2

Training Phase

Output

Classifiers weights or discrimination map

New example

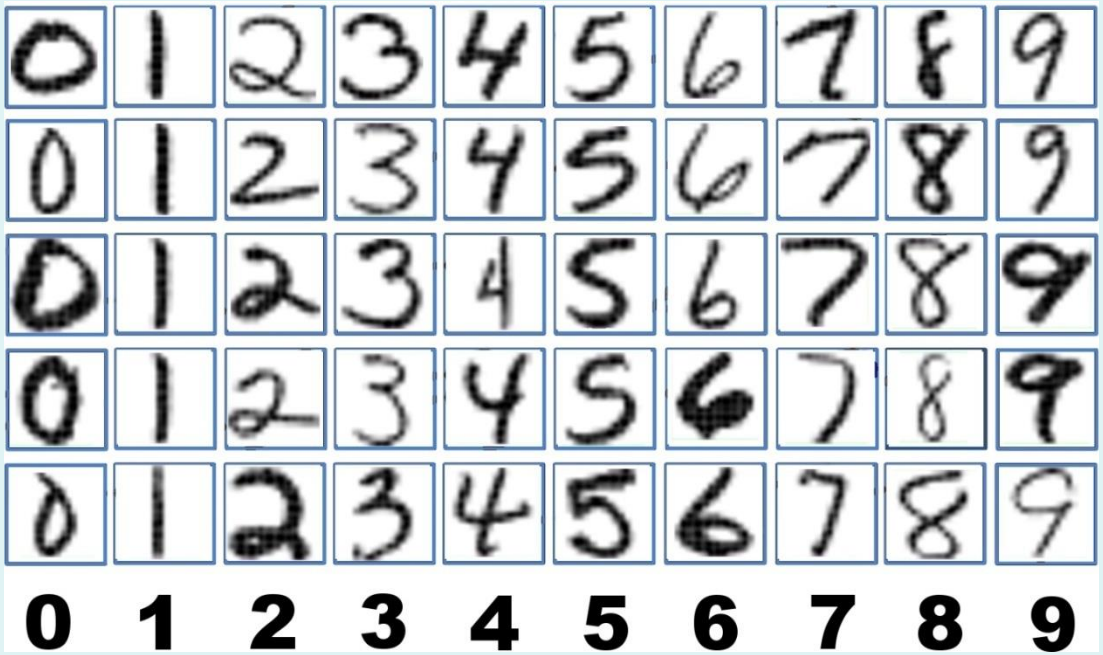
Test Phase

Prediction: task 1 or task 2

Find the mapping `h` from observed data `Y` to explanatory variable `X`

$$h: Y \rightarrow X$$

Pattern recognition concept



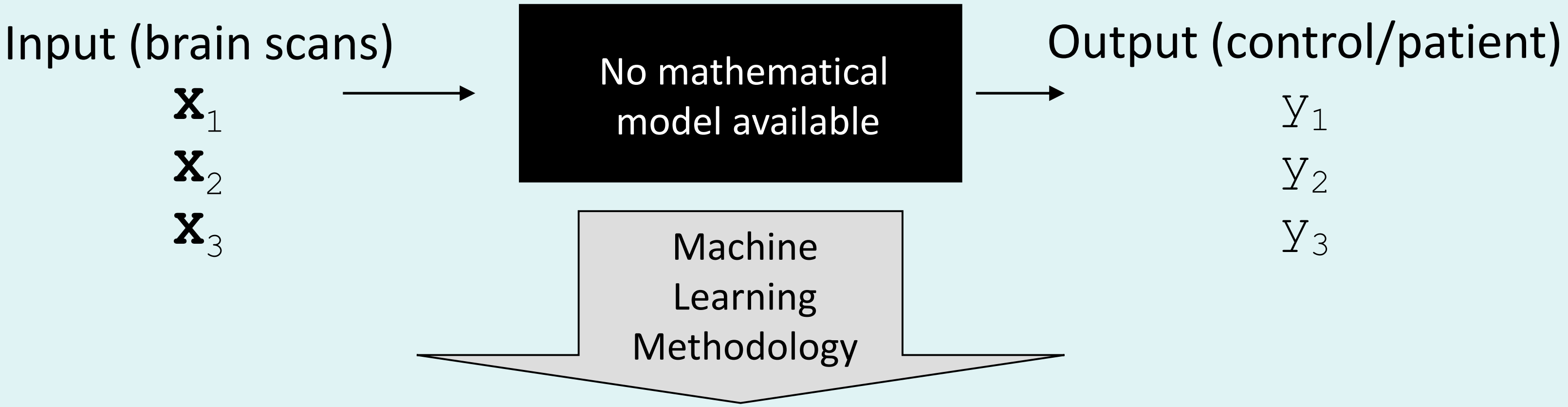
Data $\mathbf{X}$

Labels $\mathbf{y}$

$$f : \mathbf{X} \to \mathbf{y}$$

$$f : \mathbf{x}_* \to y_*$$

Pattern recognition framework



Input (brain scans)

$\mathbf{x}_1$

$\mathbf{x}_2$

$\mathbf{x}_3$

No mathematical model available

Output (control/patient)

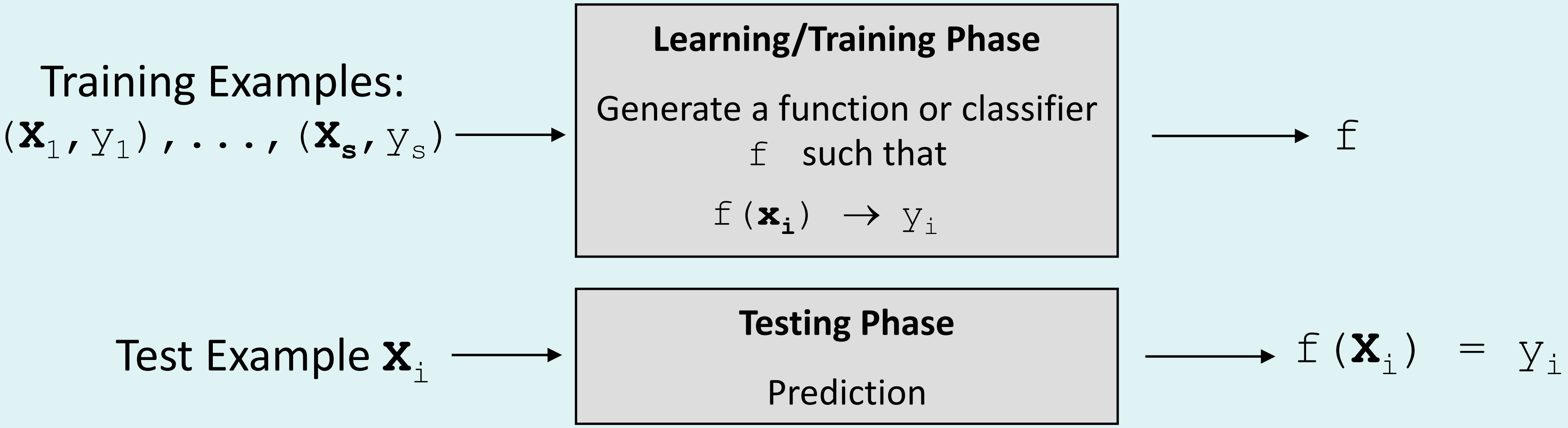
y_1

y_2

y_3

Machine Learning Methodology

Computer-based procedures that learn a function from a *series* of examples



Training Examples:

$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_s, y_s)$

Learning/Training Phase

Generate a function or classifier f such that

$f(\mathbf{x}_i) \rightarrow y_i$

f

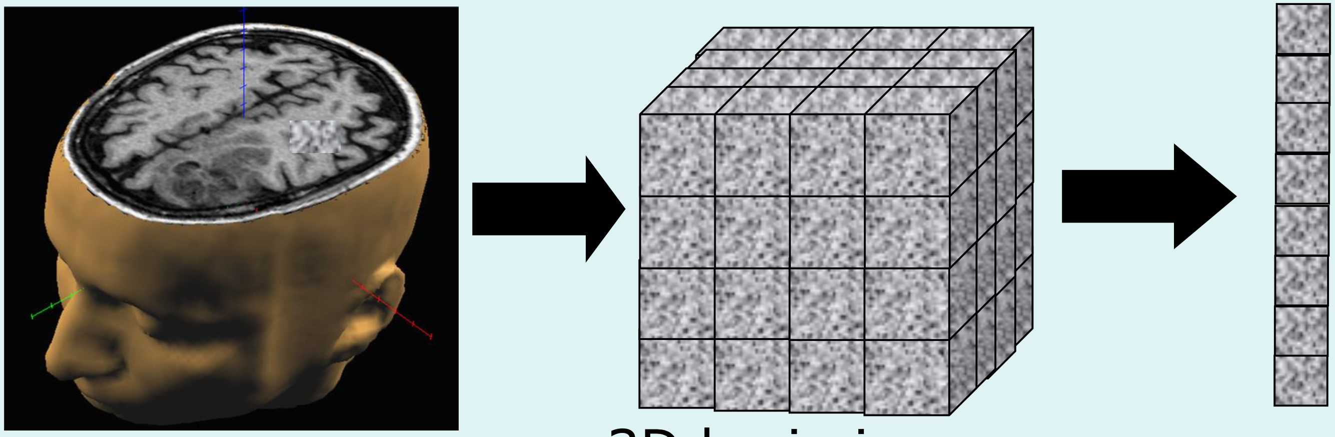
Test Example $\mathbf{x}_i$

Testing Phase

Prediction

$f(\mathbf{x}_i) = y_i$

Neuroimaging data



3D brain image

"feature vector" or "data point"

Data dimensions

- dimensionality of a "data point" = #voxels considered
- number of "data point" = #scans/images considered

Note that #voxels >> #scans !

➔ "ill posed problem"

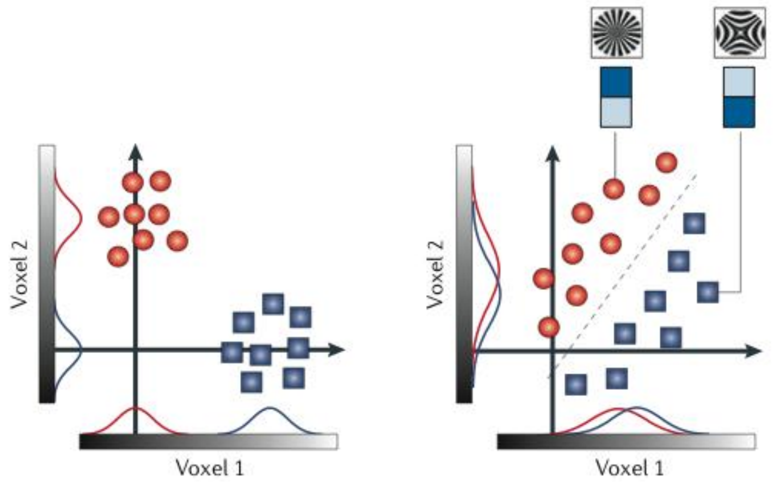
Advantages of pattern recognition

Accounts for the spatial correlation of the data (multivariate aspect)

- images are multivariate by nature.
- can yield greater sensitivity than conventional (univariate) analysis.

Enable classification/prediction of individual subjects

- ‘Mind-reading’ or decoding applications
- Clinical application

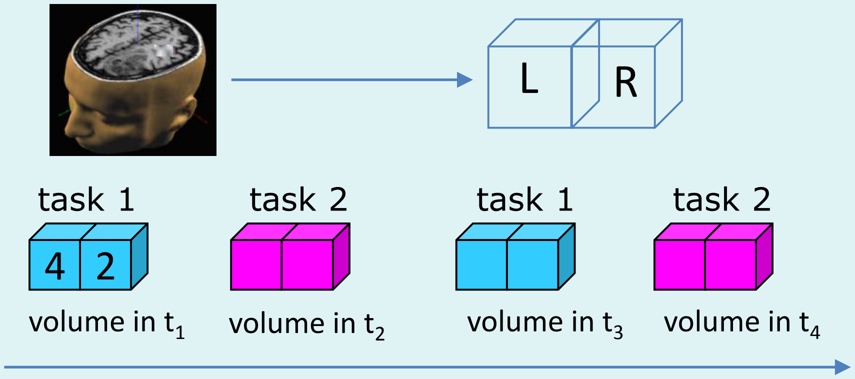


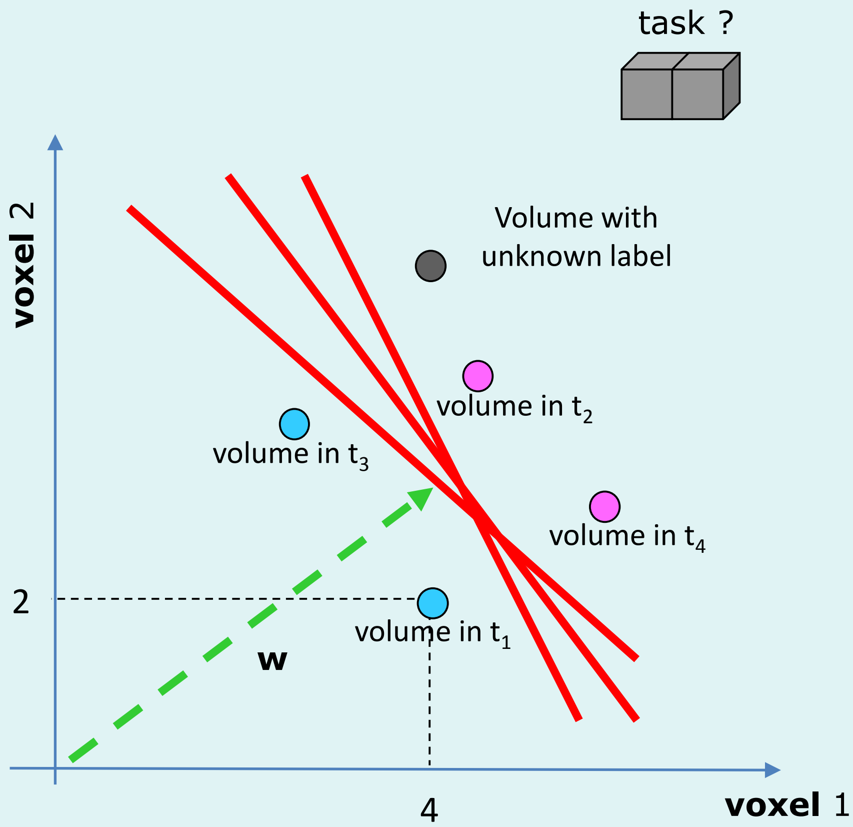
Haynes & Rees, 2006

Overview

- Introduction
  - Univariate & multivariate approaches
  - Data representation
- Pattern Recognition
  - Machine learning
  - Validation & inference
  - Weight maps & feature selection
  - Applications: groups & fMRI
- Conclusion & Toolboxes

Classification example

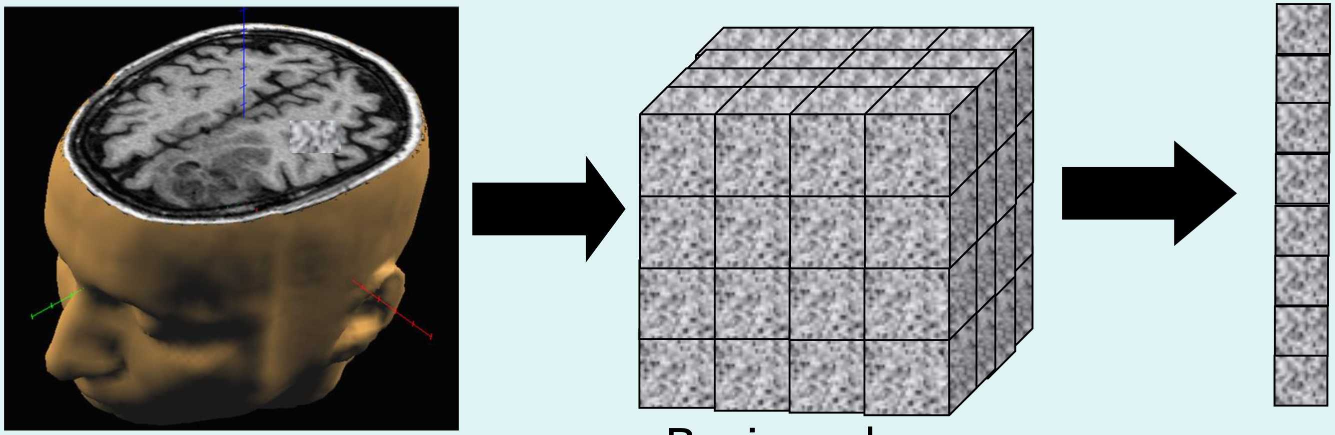




Different classifiers will compute different hyperplanes!

Note: task1/2 ~ disease/controle

Neuroimaging data



Brain volume

“feature vector” or “data point”

Problem: 1000’s of features vs. 10’s of data points

Possible solutions to dimensionality problem:

- Feature selection strategies (e.g. ROIS, select only “activated” voxels)
- (Searchlight)
- **Kernel Methods**

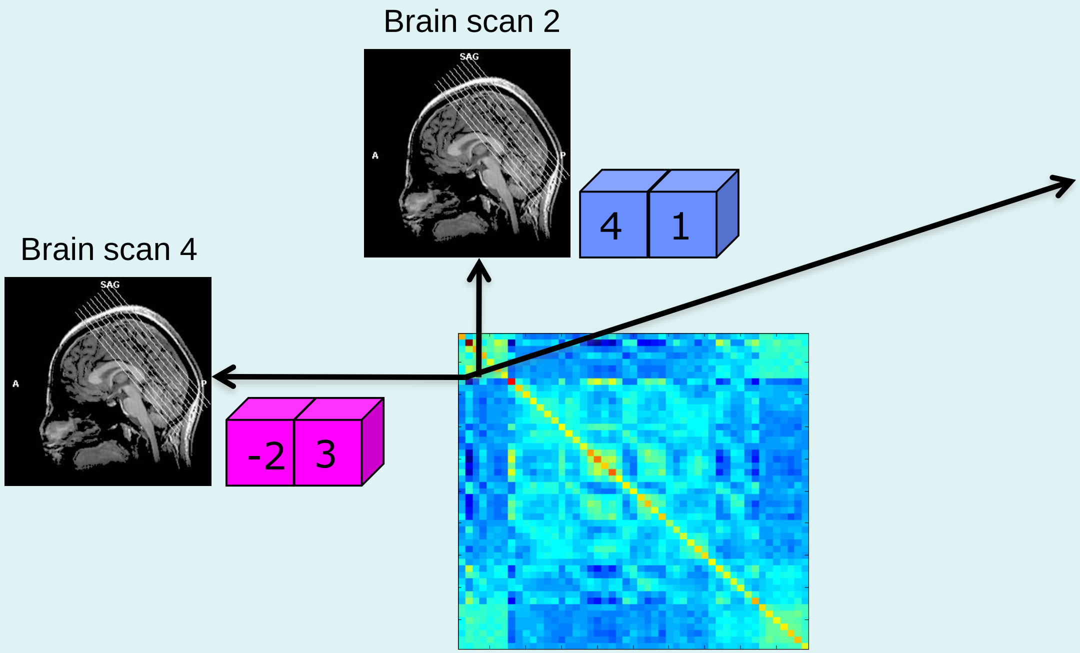
Kernel approaches

- Mathematical trick! ➔ powerful and unified framework (e.g. classification & regression)
- Consist of two parts:
  - build the kernel matrix (mapping into the feature space)
  - train using the kernel matrix (designed to discover linear patterns in the feature space)
- Advantages:
  - computational shortcut ➔ represent linear patterns efficiently in high dimensional space.
  - Using the dual representation with proper regularization ➔ efficient solution of ill-conditioned problems.
- Examples ➔ Support Vector Machine (SVM), Gaussian

  Processes (GP), Kernel Ridge Regression (KRR),…

Kernel matrix

Kernel matrix = “similarity measure”



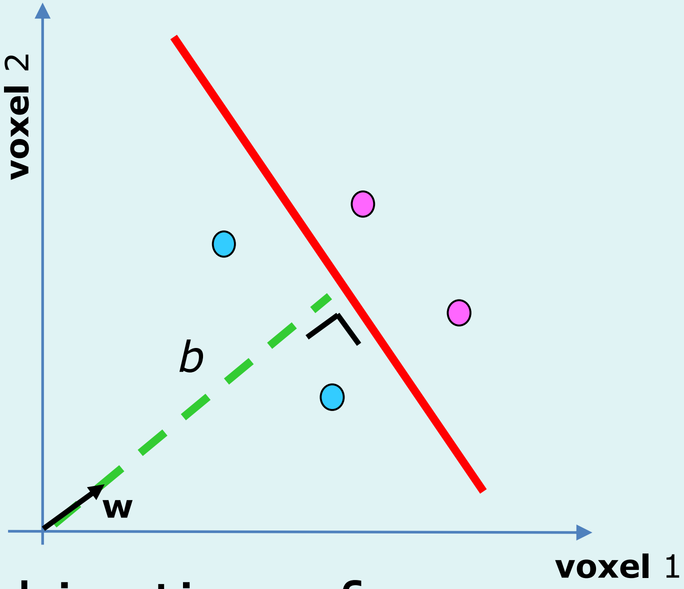
Linear kernel
= dot product
= (4*-2)+(1*3) = -5

The “kernel function”

- 2 patterns **x** and **x*** → a real number characterizing their similarity (~distance measure).
- simple similarity measure = a dot product →linear kernel.

Linear classifier

- hyperplane through the feature space
- parameterized by
  - a weight vector **w** (=direction of hyperplane),
  - a bias term *b* (=constant term).



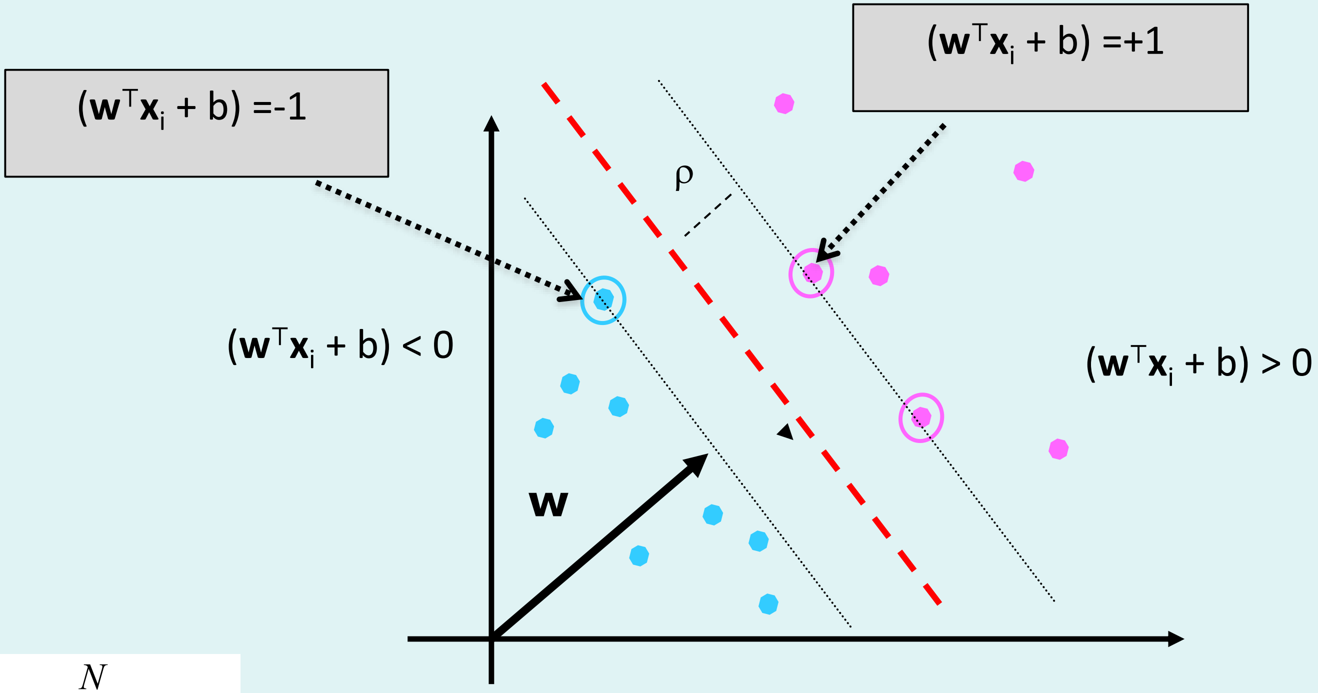
- weight vector **w** = linear combination of training examples $\mathbf{x}_i$ (where *i* = 1,…,N and N is the number of training examples)

$$\mathbf{w} = \mathop{\mathring{a}}_{i=1}^{N} a_i \mathbf{x}_i$$

➔ Find the α_i !!!

Support Vector Machine

SVM = “maximum margin” classifier



$(\mathbf{w}^T\mathbf{x}_i + b) = -1$

$(\mathbf{w}^T\mathbf{x}_i + b) = +1$

$(\mathbf{w}^T\mathbf{x}_i + b) < 0$

$(\mathbf{w}^T\mathbf{x}_i + b) > 0$

$$\mathbf{w} = \sum_{i=1}^{N} a_i \mathbf{x}_i$$

Support vectors have $\alpha_i \neq 0$

Data: $<\mathbf{x}_i, y_i>$, i=1,..,N

Observations: $\mathbf{x}_i \in R^d$

Labels: $y_i \in \{-1,+1\}$

Linear classifier prediction

General equation: making predictions for a test example $\mathbf{x}_*$ with kernel methods

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i \mathbf{x}_i$$

$$f(\mathbf{x}_*) = \mathbf{w} \cdot \mathbf{x}_* + b \longrightarrow \text{Primal representation}$$

$$f(\mathbf{x}_*) = \sum_{i=1}^{N} \alpha_i \mathbf{x}_i \cdot \mathbf{x}_* + b$$

kernel definition

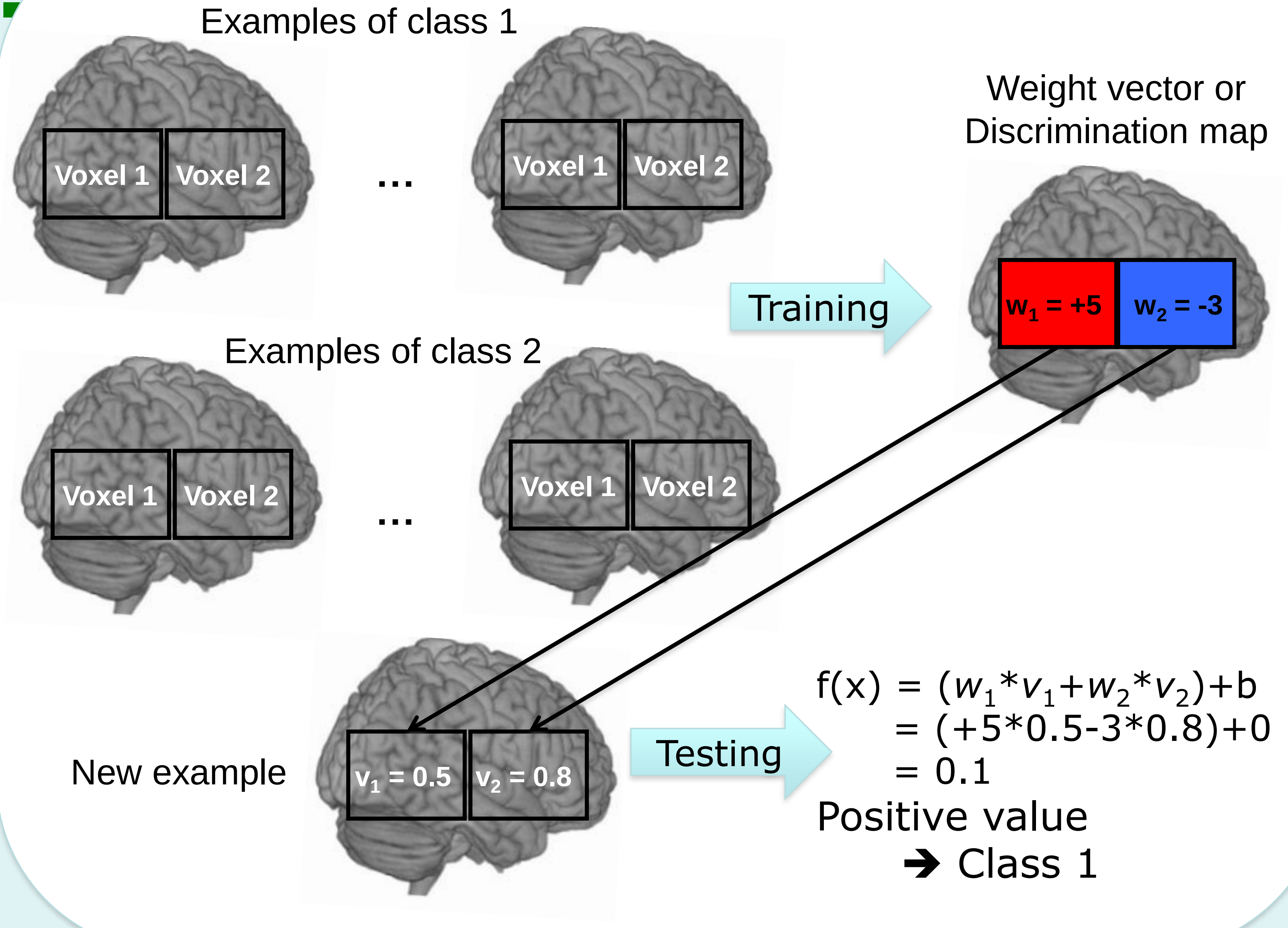
$$f(\mathbf{x}_*) = \sum_{i=1}^{N} \alpha_i K(\mathbf{x}_i, \mathbf{x}_*) + b \longrightarrow \text{Dual representation}$$

f($\mathbf{x}_*$) =

signed distance to boundary (classification)

predicted score (regression)

Illustrative example: Classifiers as decision functions



Examples of class 1

Voxel 1 Voxel 2 … Voxel 1 Voxel 2

Examples of class 2

Voxel 1 Voxel 2 … Voxel 1 Voxel 2

Training

Weight vector or Discrimination map

$w_1 = +5$ $w_2 = -3$

New example

$v_1 = 0.5$ $v_2 = 0.8$

Testing

$$
\begin{aligned}
f(x) &= (w_1 * v_1 + w_2 * v_2) + b \\
&= (+5*0.5-3*0.8)+0 \\
&= 0.1
\end{aligned}
$$

Positive value

➔ Class 1

SVM vs. GP

SVM

➔ Hard binary classification

- – simple & efficient, quick calculation but
- – NO ‘grading’ in output {-1, 1}

Gaussian Processes

➔ probabilistic model

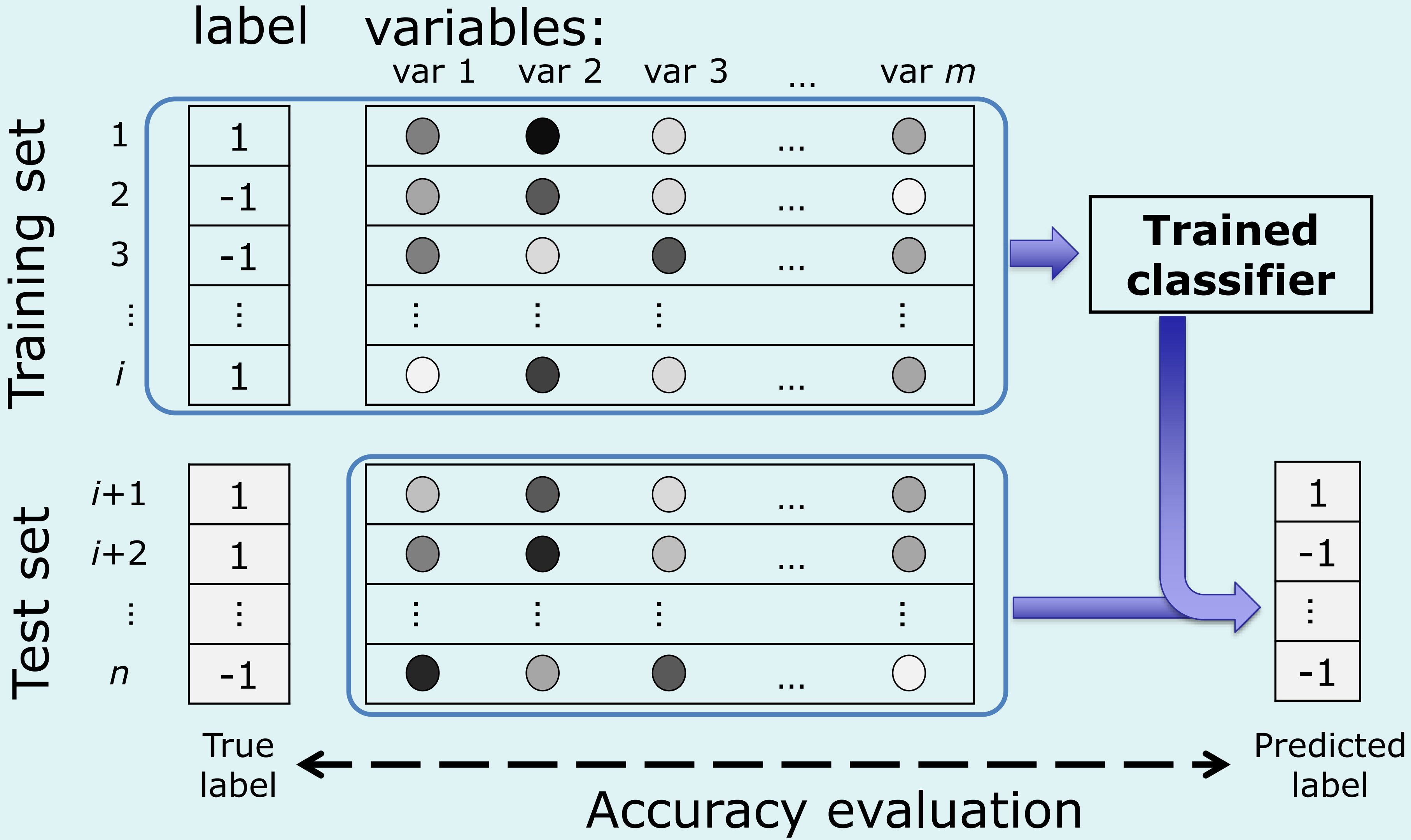
- – more complicated, slower calculation but
- – returns a probability [0 1]
- – can be multiclass

Overview

- Introduction
  - Univariate & multivariate approaches
  - Data representation
- Pattern Recognition
  - Machine learning
  - Validation & inference
  - Weight maps & feature selection
  - Applications: groups & fMRI
- Conclusion & Toolboxes

Validation principle

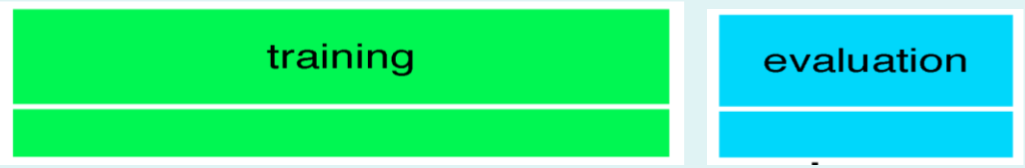
Samples = {variables, labels}

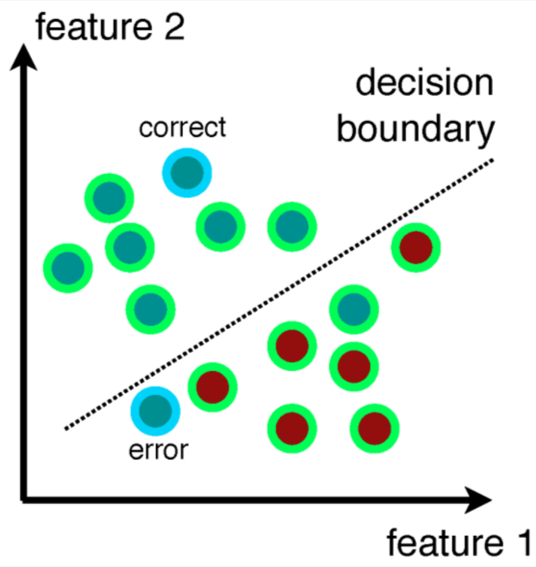


M-fold cross-validation

- Split data in 2 sets: “train” & “test”

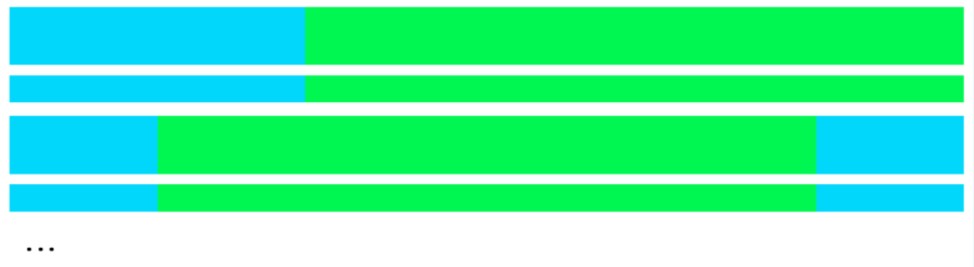
  ➔ evaluation on 1 “fold”





- Rotate partition and repeat

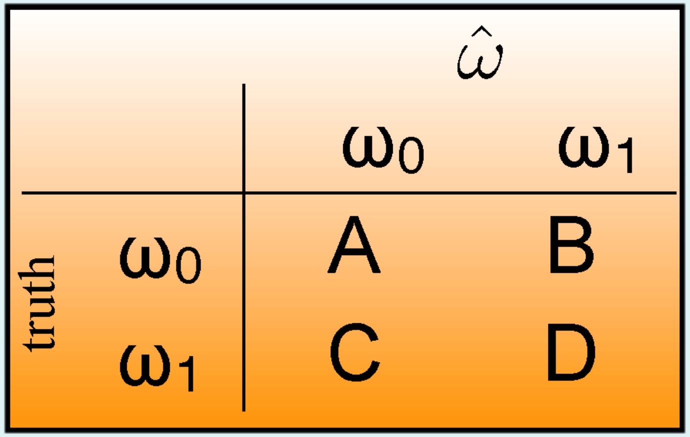
  ➔ evaluations on M “folds”



- Applies to scans/events/blocks/subjects/…

  ➔ Leave-one-out (LOO) approach

Confusion matrix & accuracy



$$\begin{array}{cc|cc} & & \multicolumn{2}{c}{\hat{\omega}} \\ & & \omega_0 & \omega_1 \\ \hline \text{truth} & \omega_0 & A & B \\ & \omega_1 & C & D \end{array}$$

Confusion matrix

= summary table

Accuracy estimation

- Class 0 accuracy, p_0 = A/(A+B)
- Class 1 accuracy, p_1 = D/(C+D)
- Accuracy, p = (A+D)/(A+B+C+D)

Other criteria

- Positive Predictive Value, PPV = D/(B+D)
- Negative Predictive Value, NPV = A/(A+C)

Accuracy & Dataset balance

Watch out if #samples/class are different!!!

Example: Classes A/B with 80/20 samples each

➔ observed $\mathbf{a_{tot} = 70\%}$ overall accuracy but

- within class A ($N_A = 80$), excellent accuracy (85%)
- within class B ($N_B = 20$), poor accuracy (10%)

➔ balanced accuracy $\mathbf{a_{bal} = 47{,}5\%}$!

Good practice:

Report

- class accuracies $[a_0, a_1, \ldots, a_C]$
- balanced accuracy $a_{bal} = (a_0 + a_1 + \ldots + a_C)/\#\text{Classes}$

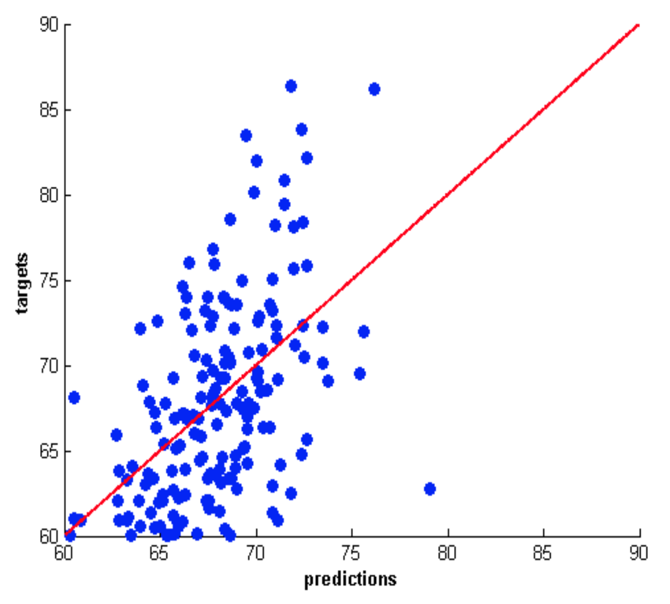
Regression validation

“Mean squared error” (MSE):

•LOO error in one fold $SE_n = (y_n - f(\mathbf{x}_n))^2$

•Across all LOO folds $R(f, \mathbf{X}) = MSE = \frac{1}{N}\sum_{n=1}^{N}(y_n - f(\mathbf{x}_n))^2$

➔ Out-of-sample “mean squared error” (MSE)



<u>Other measure:</u>

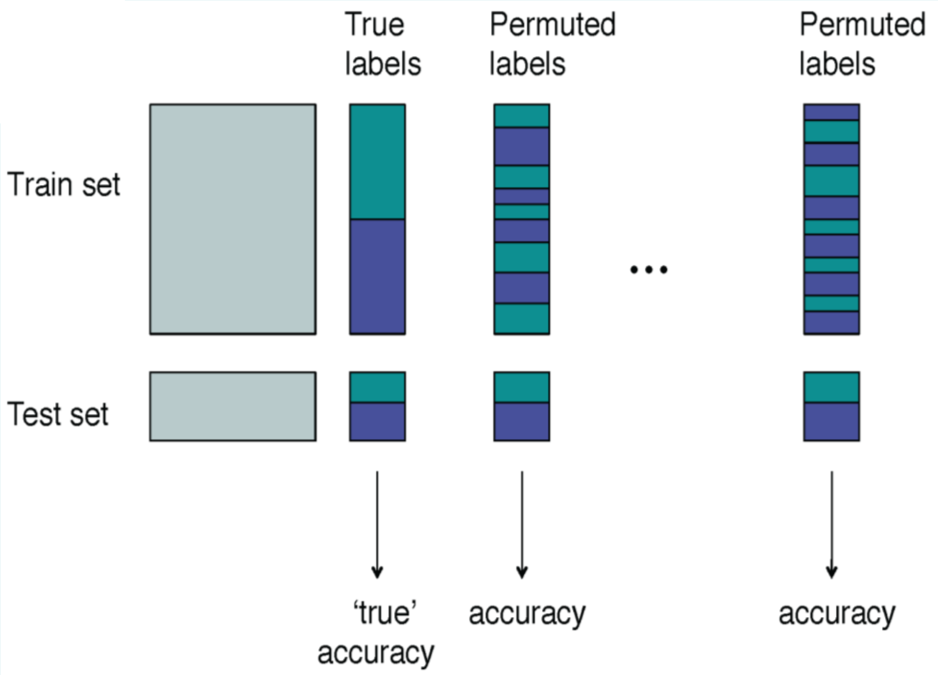
Correlation between:

•predictions (across folds!), and
•‘true’ targets

Inference by permutation testing

- H_0: “class labels are non-informative”
- Test statistic = CV accuracy (total or balanced)
- Estimate distribution of test statistic under H0
  - ➔ Random permutation of labels
  - ➔ Estimate accuracy
  - ➔ Repeat M times
- Calculate p-value as

$$p = \frac{1}{M} \sum_{m}^{M} (a_m^{\mathrm{perm}} \geqslant a^{\mathrm{true}})$$



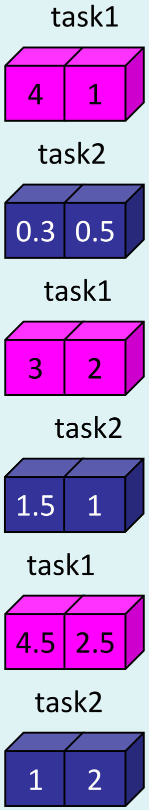
Overview

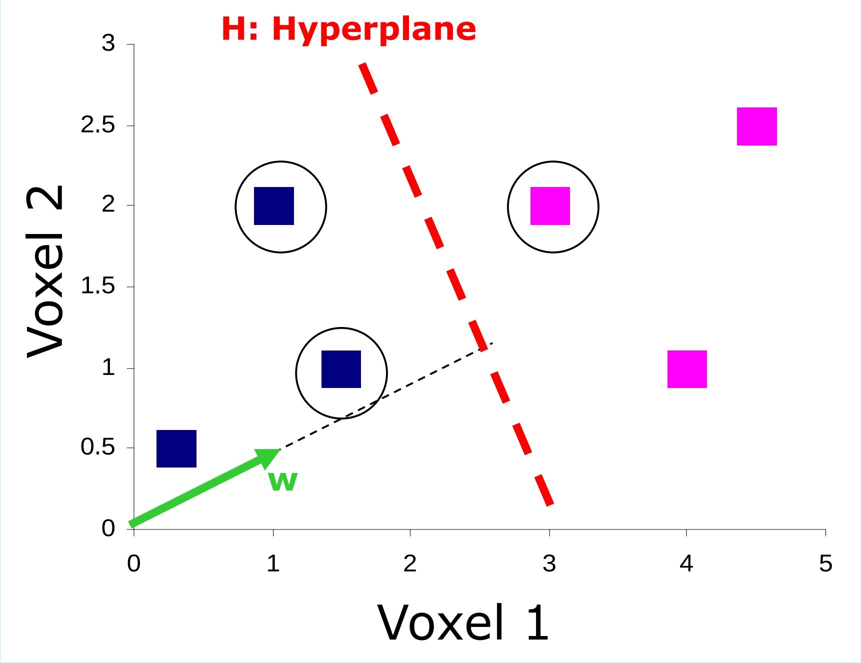
- Introduction
  - Univariate & multivariate approaches
  - Data representation
- Pattern Recognition
  - Machine learning
  - Validation & inference
  - Weight maps & feature selection
  - Applications: groups & fMRI
- Conclusion & Toolboxes

Weight vector interpretation

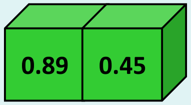
Weight vector

- ➔ weight (or discrimination) image !
- ➔ how important each voxel is
- ➔ for which class "it votes" (mean centred data & b=0)





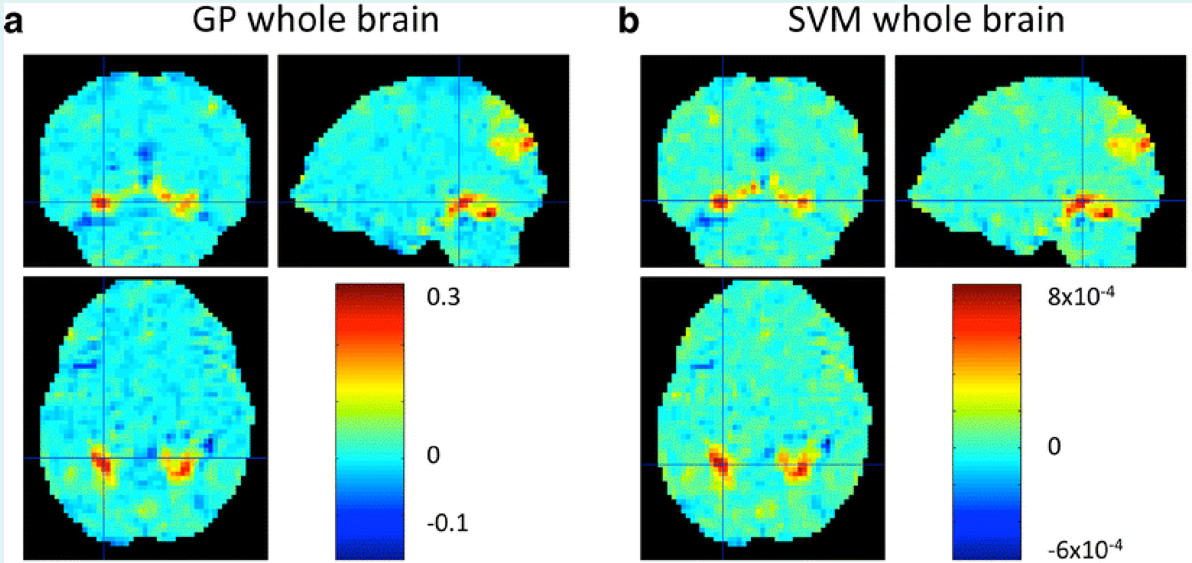
Weight vector
W = [0.89 0.45]
b = -2.8



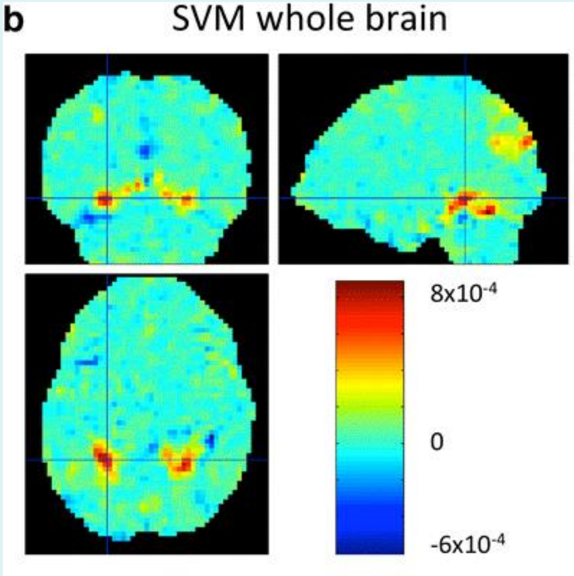
Weight maps for different masks

Linear machine
→ Weight map

Different mask
→ different feature set
→ different weight map



Feature selection



- 1 sample image

  ➔ 1 predicted value
- use ALL the voxels

  ➔ NO thresholding of weight allowed!

Feature selection:

- *a priori* mask
- *a priori* ‘filtering’
- Multiple Kernel Learning
- (Search Light)
- Recursive Feature Elimination/Addition

MUST be independent from test data!

Overview

- Introduction
  - Univariate & multivariate approaches
  - Data representation
- Pattern Recognition
  - Machine learning
  - Validation & inference
  - Weight maps & feature selection
  - Applications: groups & fMRI
- Conclusion & Toolboxes

Application & designs

Levels of “inference”

- within subject ≈ FFX with SPM
  - ➔ ‘decode’ subject’s brain states
  - ➔ multiple images: typically fMRI

- between subjects ≈ RFX with SPM
  - ➔ ‘classify’ groups, e.g. patients vs. controls (or regress subjects’ parameter)
  - ➔ 1 (or few) image(s)/subject : contrast (fMRI), structural (GM, FA, Jacobian,…), PET, etc.

Between subjects

Design

- 2 groups: group A vs. group B
- 1 group: 2 conditions per subject (e.g. before/after treatment)
- 1 group: 1 target score

➔ Extract 1 (or a few) summary image(s) per subject, and classify/regress

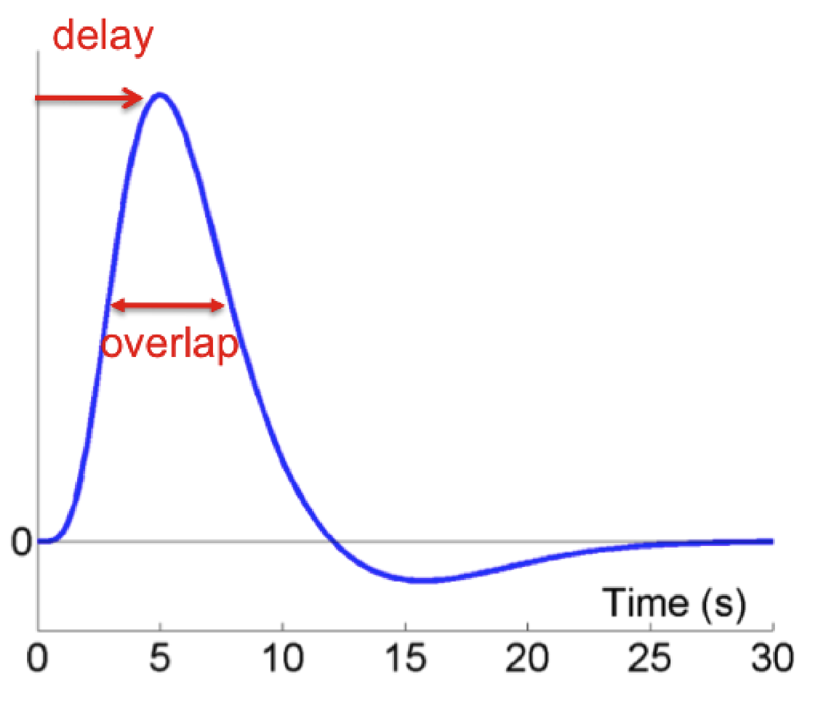
Leave-one-out (LOO) cross-validation:

- Leave one subject out (LOSO)
- Leave one subject per group out (LOSGO)

Within subject, fMRI

Design:

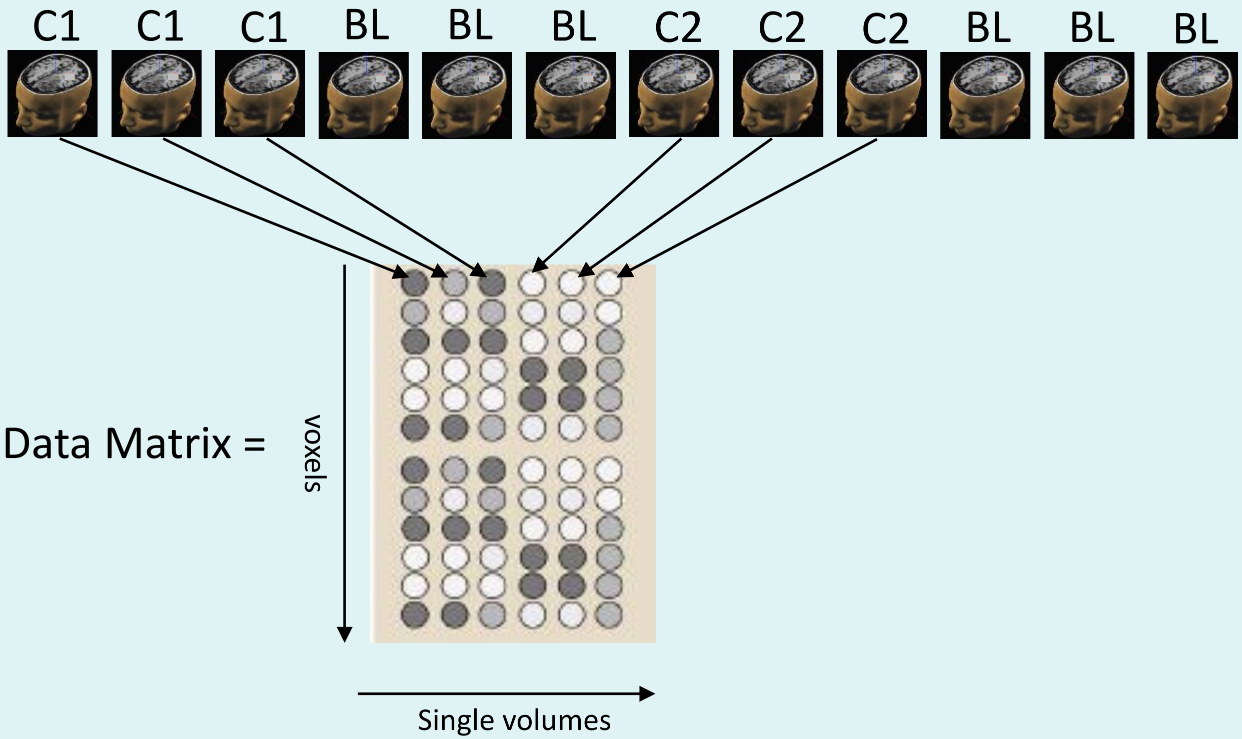
- Block or event-related design?
- How to account for haemodynamic function?



Within subject, fMRI

Design:

- Block design
- Use single scans

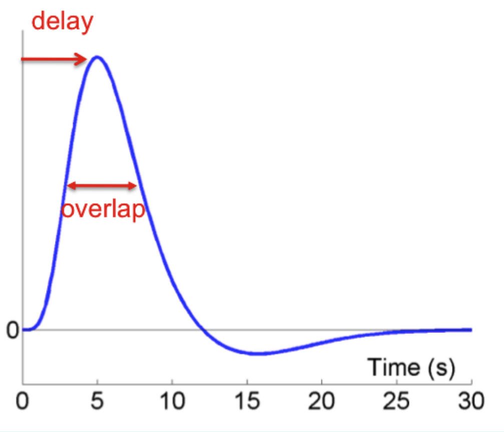


C1: Condition 1

C2: Condition 2

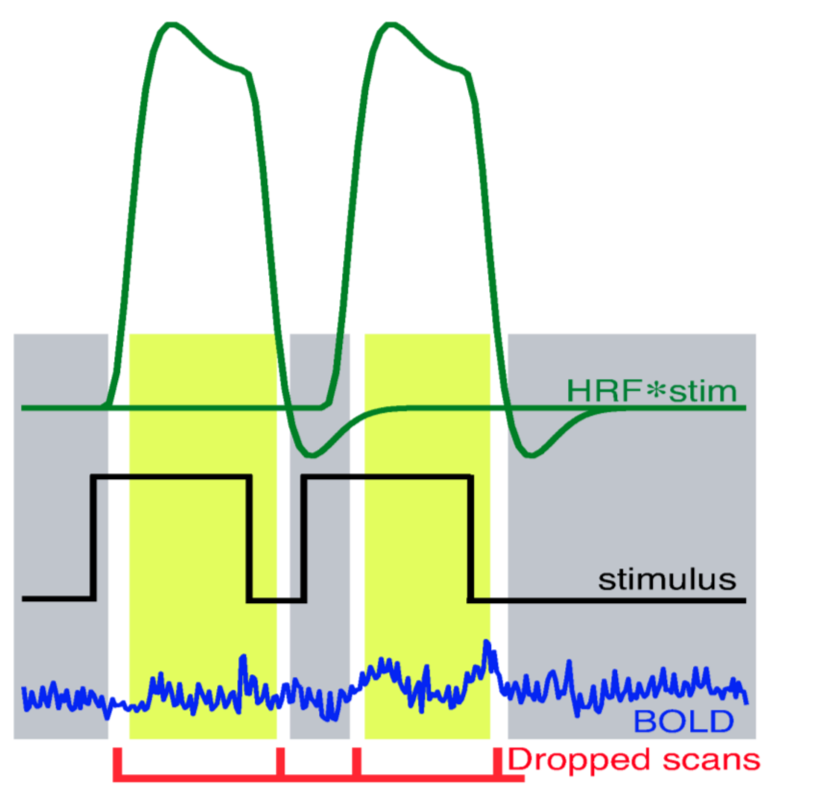
BL: Baseline

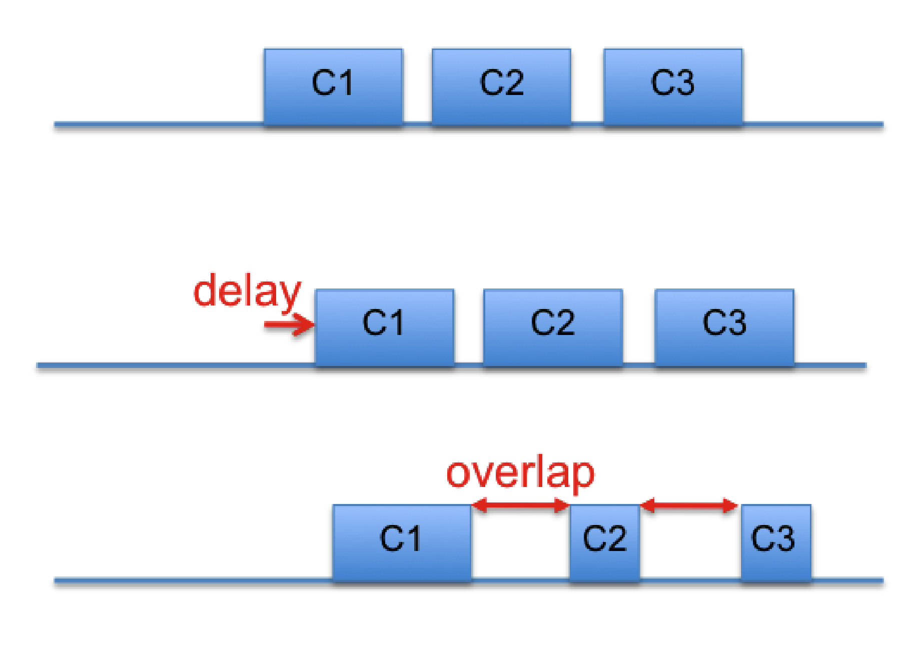
Within subject, fMRI



Design:

- Block or event-related design
- Accounting for haemodynamic function, with HRF ‘delay’ & ‘overlap’

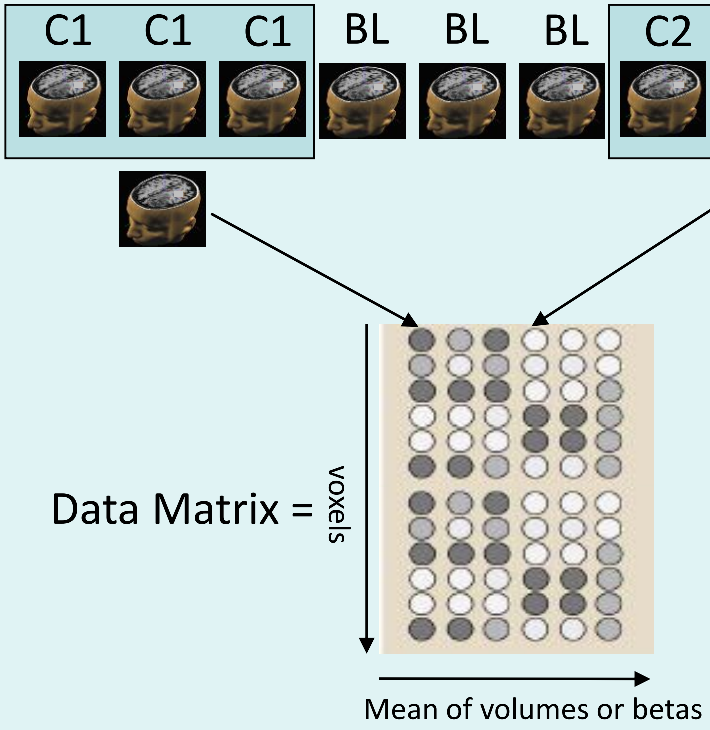




Within subject, fMRI

Design:

- Block or event-related design
- Accounting for haemodynamic function by averaging/deconvolution



C1: Condition 1
C2: Condition 2
BL: Baseline

How to?

- **Average scans** over blocks/events
- **Parameter (beta) estimate** from the GLM with 1 regressor per block/event

Mourão-Miranda et al. (2006)

Within subject

Design:

- Block or event-related design
- Accounting for haemodynamic function

Leave-one-out (LOO) cross-validation:

- Leave one session/run out
- Leave one block/event out

(danger of dependent data!!!)

Overview

- Introduction
  - Univariate & multivariate approaches
  - Data representation
- Pattern Recognition
  - Machine learning
  - Validation & inference
  - Weight maps & feature selection
  - Applications: groups & fMRI
- Conclusion & Toolboxes

Conclusions

Key points:

- NO local (voxel/blob) inference
  - ➔ CANNOT report coordinates nor thresholded weight map
- Require cross-validation (split in train/test sets)
  - ➔ report accuracy/PPV (or MSE)
- MUST assess significance of accuracy
  - ➔ permutation approach
- Could expect more sensitivity (~like omnibus test with SPM)
- Different questions & Different designs!?

“Univariate vs. multivariate” concepts

Univariate	Multivariate
• 1 voxel	• 1 volume
• target → data	• data → target
• look for difference or correlation	• look for similarity or score
• General Linear Model	• Specific machine (SVM, GP,...)
• GLM inversion	• training & testing cross-validation
• calculate contrast of interest	• estimate accuracy of prediction

Existing toolboxes

✓ Princeton Multi-Voxel Pattern Analysis (MVPA) Toolbox (http://code.google.com/p/princeton-mvpa-toolbox/)

- (+) Matlab based, AFNI/BV oriented but some SPM5 compatibility,
- (-) only(?) neural network classification, only(?) fMRI data, last version from 2009.

✓ Multivariate Pattern Analysis with Python – PyMVPA (http://www.pymvpa.org/)

- (+) looks powerful with loads of tools,
- (-) Python based, OS specific (Linux).

✓ The Decoding Toolbox – TDT (https://sites.google.com/site/tdtdecodingtoolbox/)

- (+) search-light & SVM, ROI/whole brain, feature selection, flexible CV & design, SPM8/12 compatibility
- (-) no SPM-style batching (?)

✓ Brain Voyager MVPA – BV-MVPA (http://www.brainvoyager.com/bvqx/)

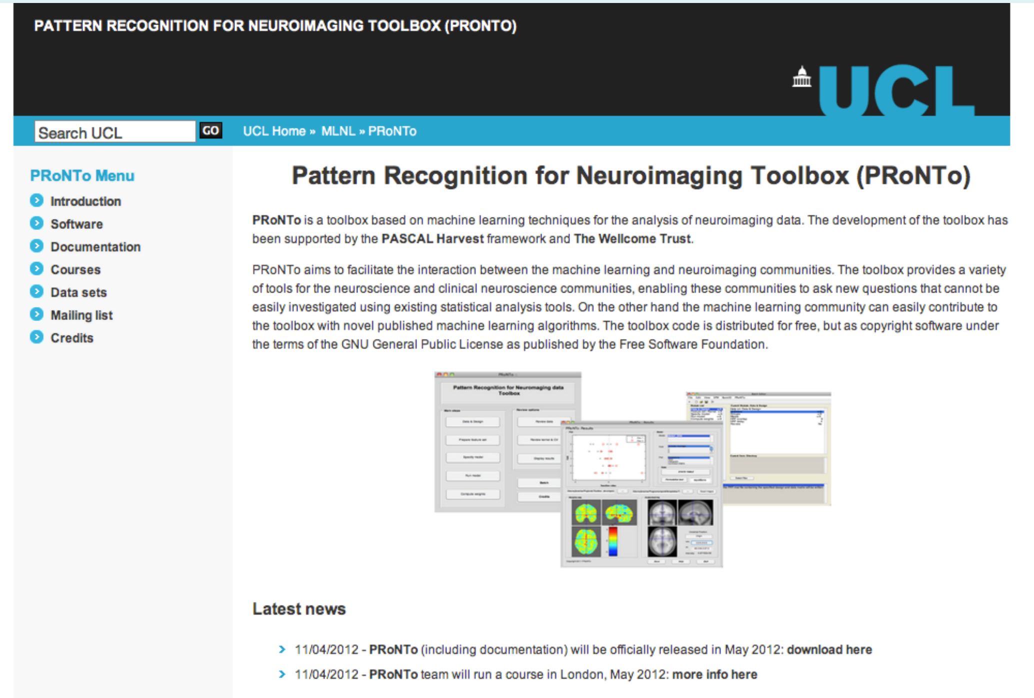
- (+) search-light & SVM,
- (-) proprietary code, fMRI only, no way of adding own machine

PRoNTo

Pattern **R**ecognition f**o**r **N**euroimaging **To**olbox

http://www.mlnl.cs.ucl.ac.uk/pronto/

with references, manual, demo data, course, etc.



PATTERN RECOGNITION FOR NEUROIMAGING TOOLBOX (PRONTO)

Pattern Recognition for Neuroimaging Toolbox (PRoNTo)

PRoNTo is a toolbox based on machine learning techniques for the analysis of neuroimaging data. The development of the toolbox has been supported by the **PASCAL Harvest** framework and **The Wellcome Trust**.

PRoNTo aims to facilitate the interaction between the machine learning and neuroimaging communities. The toolbox provides a variety of tools for the neuroscience and clinical neuroscience communities, enabling these communities to ask new questions that cannot be easily investigated using existing statistical analysis tools. On the other hand the machine learning community can easily contribute to the toolbox with novel published machine learning algorithms. The toolbox code is distributed for free, but as copyright software under the terms of the GNU General Public License as published by the Free Software Foundation.

Latest news

- 11/04/2012 - **PRoNTo** (including documentation) will be officially released in May 2012: **download here**
- 11/04/2012 - **PRoNTo** team will run a course in London, May 2012: **more info here**

Schrouff et al, 2013.

Afternoon workshop

More about

- Weight interpretation
- Machines & “multi-kernel learning”
- Nested CV & parameter optimization
- Feature selection & search light approach
- Multiclass (N>2) classification
- fMRI design & group analysis
- …

And practical demo of PRoNTo:

- fMRI & group analysis
- GUI and batching

Thank you for your attention!

Any question?

Thanks to the PRoNTo Team for the borrowed slides. ☺

References

- C. Bishop. Pattern Recognition and Machine Learning. Springer, 2006.
- Haynes and Rees (2006). Decoding mental states from brain activity in humans. *Nat. Rev. Neurosci*. 7, 523–534.
- Mourão-Miranda et al. (2006). The impact of temporal compression and space selection on SVM analysis of single-subject and multi-subject fMRI data. *Neuroimage* 33, 1055–1065.
- Noirhomme, et al. (2014). Biased binomial assessment of cross-validated estimation of classification accuracies illustrated in diagnosis predictions. *Neuroimage Clin*. 4, 687–694.
- Pereira et al., Machine learning classifiers and fMRI: A tutorial overview, NeuroImage 45, 2009.
- C. Rasmussen and C. K. I. Williams. Gaussian Processes for Machine Learning . The MIT Press, 2006.
- Schrouff et al. (2013). PRoNTo: Pattern Recognition for Neuroimaging Toolbox. *Neuroinformatics*, 11, 319-337.
- J. Shawe-Taylor and N. Christianini (2004) Kernel Methods for Pattern Analysis. Cambridge: Cambridge University Press.