Pattern Recognition for Neuroimaging Data

Edinburgh, SPM course

April 2019

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

Overview

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

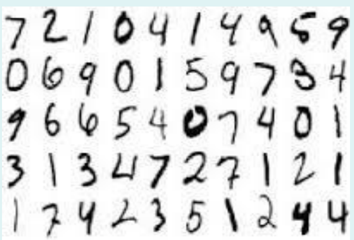
Overview

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

Pattern recognition concept

- Pattern recognition aims to find *patterns/regularities* in the data that can be used to take actions (e.g. make predictions), aka. machine learning, AI,…

Digit Recognition



Face Recognition

Recommendation Engines

- Types of Learning:
  - supervised learning: trained with labeled data (classification & regression)
  - unsupervised learning: trained with unlabeled data (clustering)
  - reinforcement learning: actions and rewards (robotics)

Pattern recognition framework

Input
(patterns)

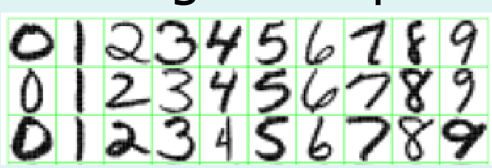
No mathematical model available

Output
(labels)
0 1 2 3 4 5 6 7 8 9

Machine Learning Methodology

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

Training Examples:



0 1 2 3 4 5 6 7 8 9

Learning/Training Phase

Generate a function or classifier f such that f(**input**) = label

f

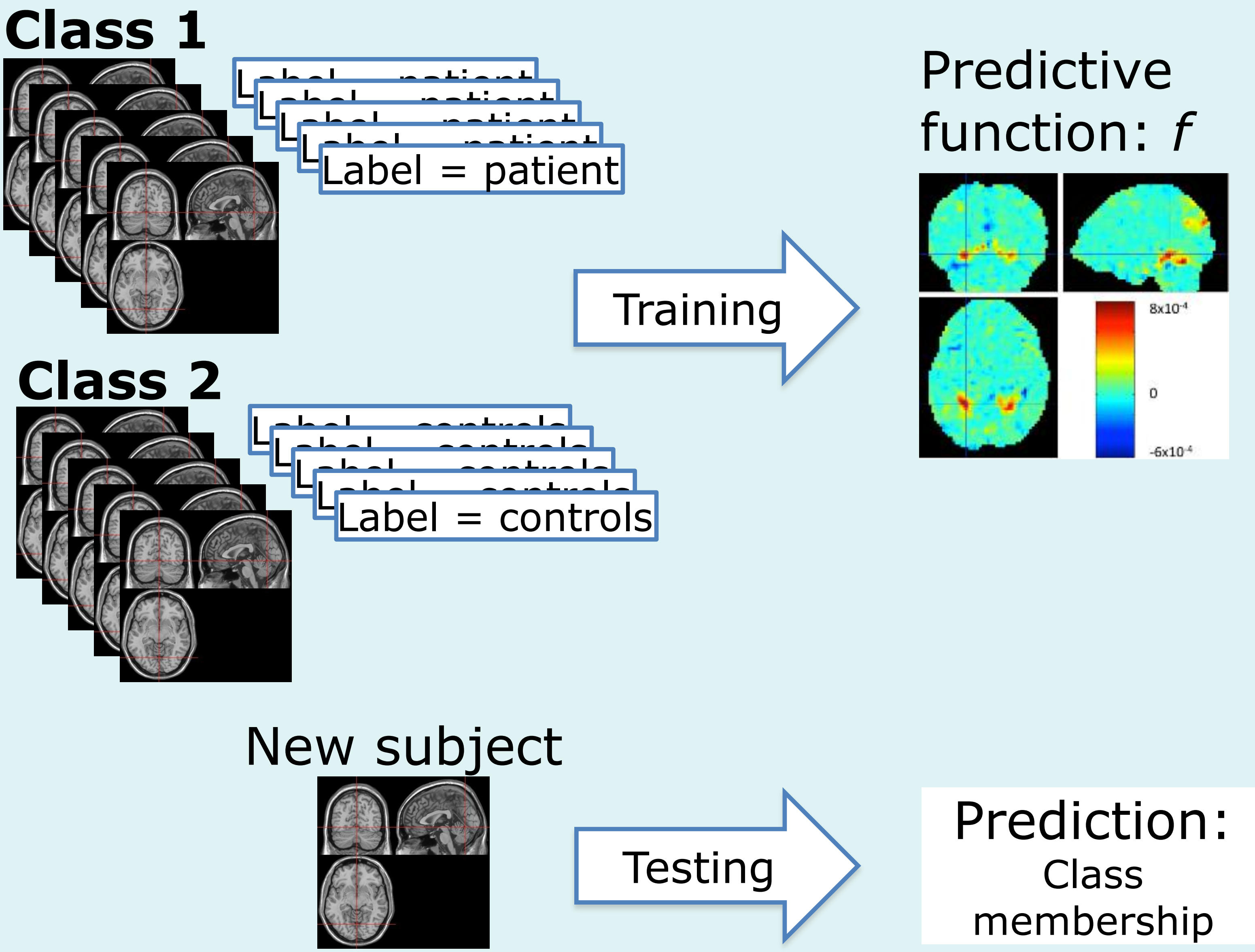
Test Example

Testing Phase

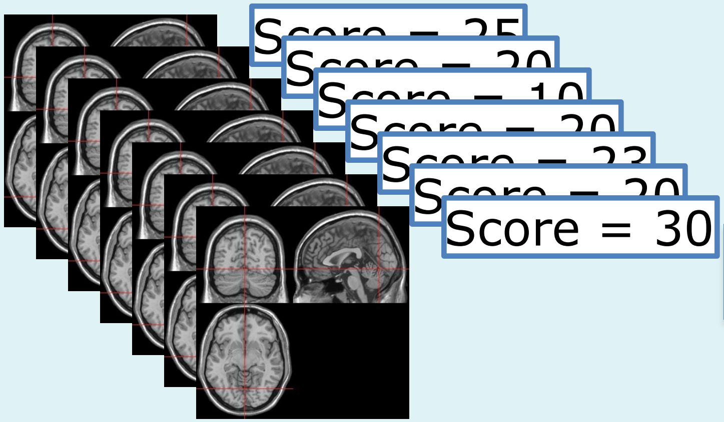
Prediction

3

Classification model

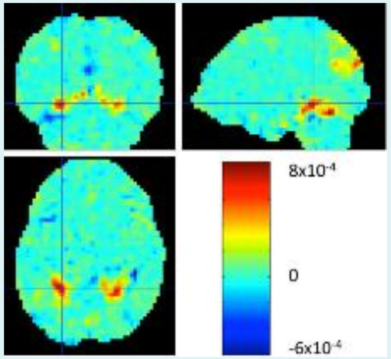


Regression model



Training

Predictive function: *f*



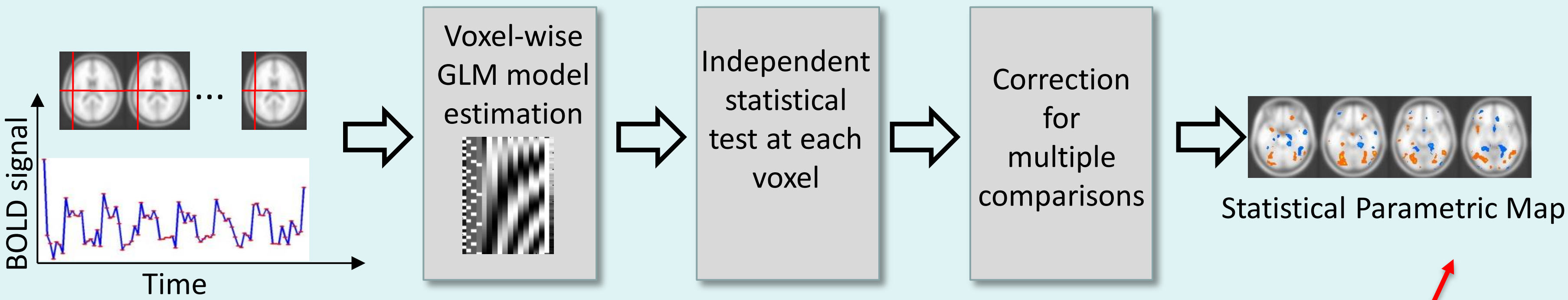
New subject

Testing

Prediction:
Score = 28

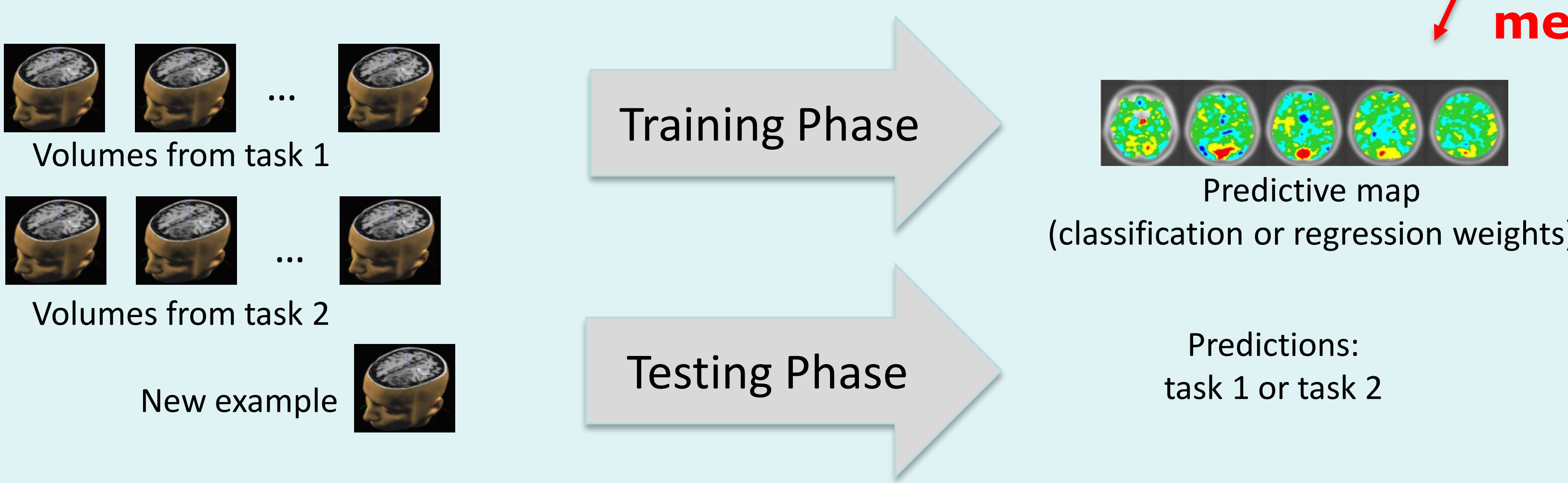
Mass-univariate vs Pattern recognition

Standard Statistical Analysis (mass-univariate)



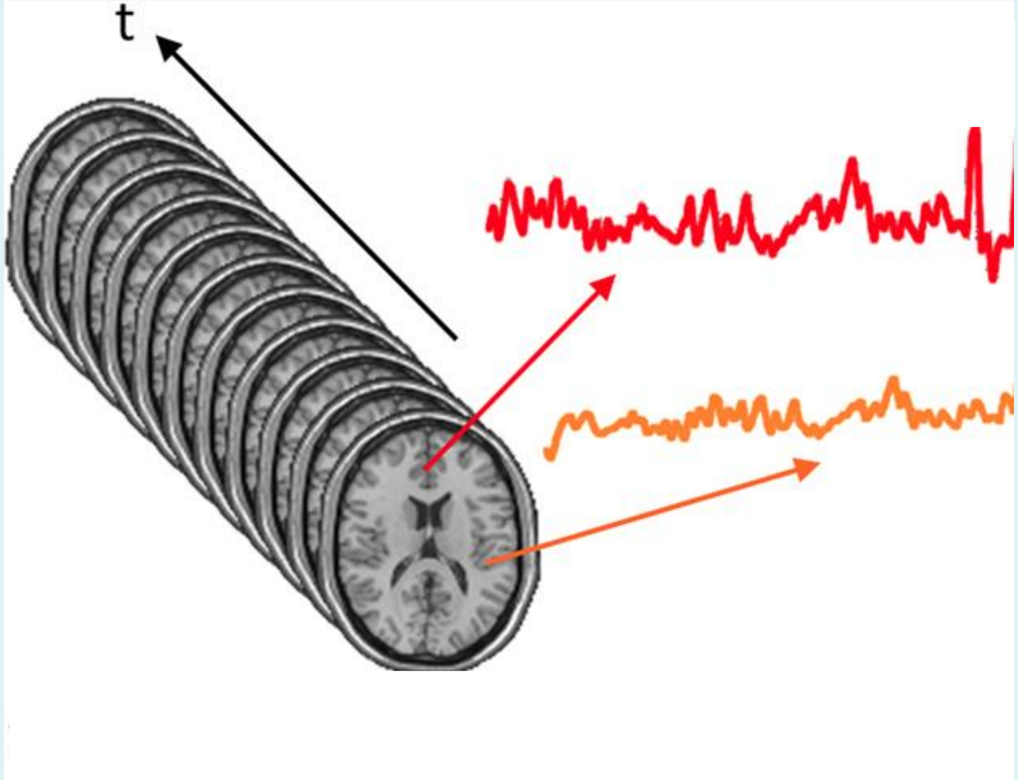
Very different meaning!

Pattern Recognition Analysis (multivariate & predictive)



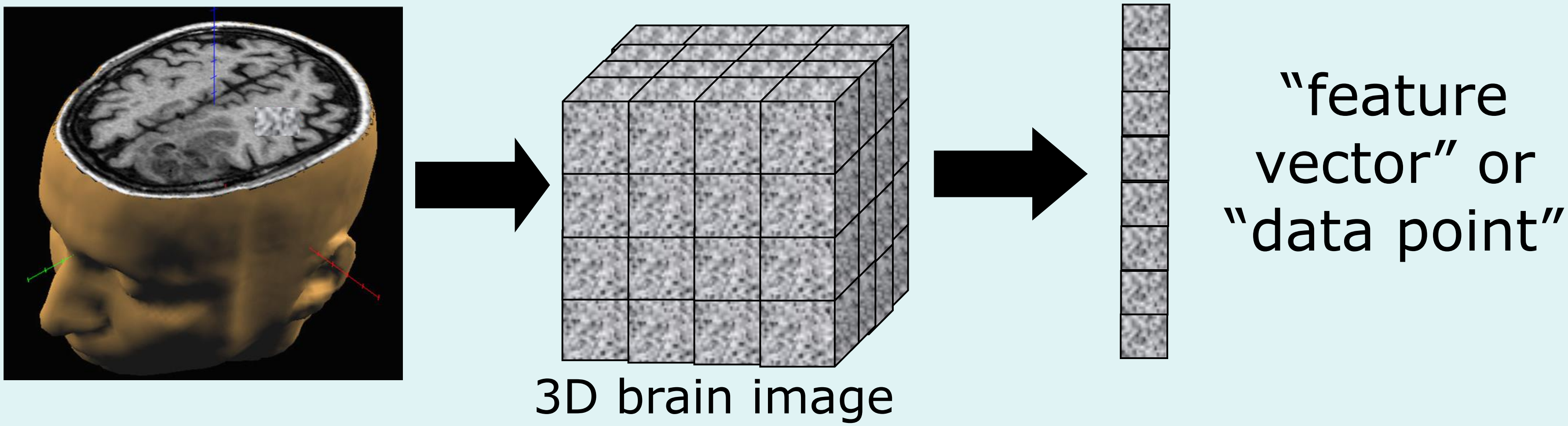
Neuroimaging data

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



About the same for a series of structural MRIs

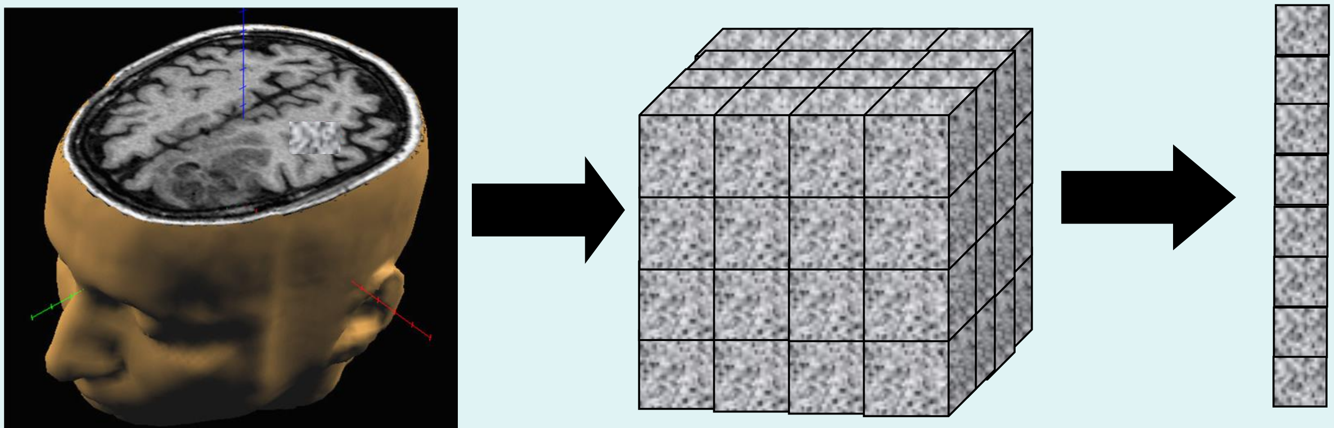
Neuroimaging data features



Data dimensions

- dimensionality of a “data point”, aka. features
  = #voxels considered
- number of “data points”, aka. samples
  = #scans/images considered

Neuroimaging data features



3D brain image

"feature vector" or "data point"

Types of features:

- fMRI:
  BOLD signal, contrast image, connectivity maps/matrix, …
- sMRI:
  GM maps, volume change map, cortical thickness,…
- PET images
- EEG/MEG

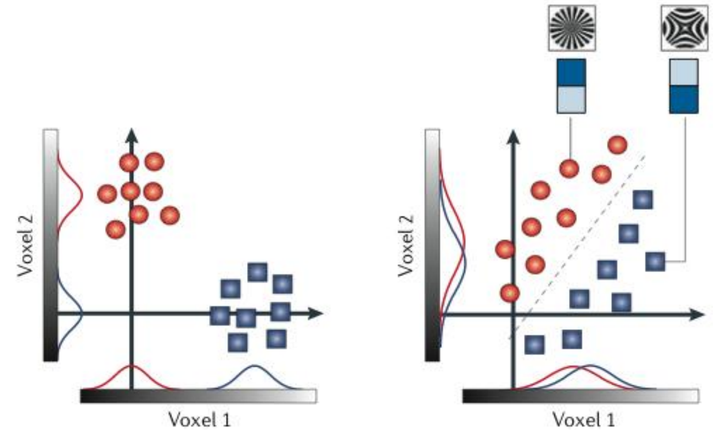
Advantages of pattern recognition

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 new samples

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

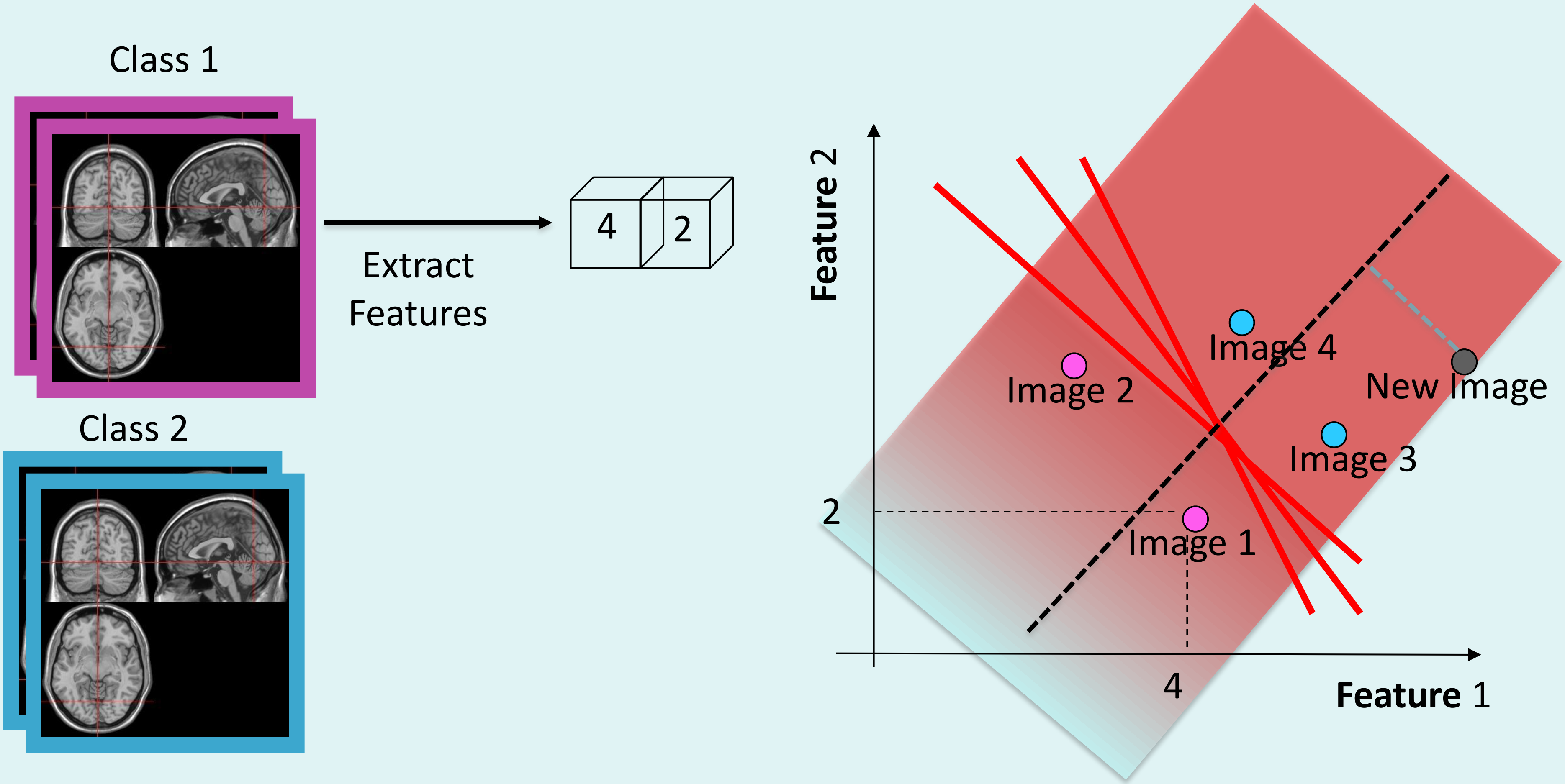


Haynes & Rees, 2006

Overview

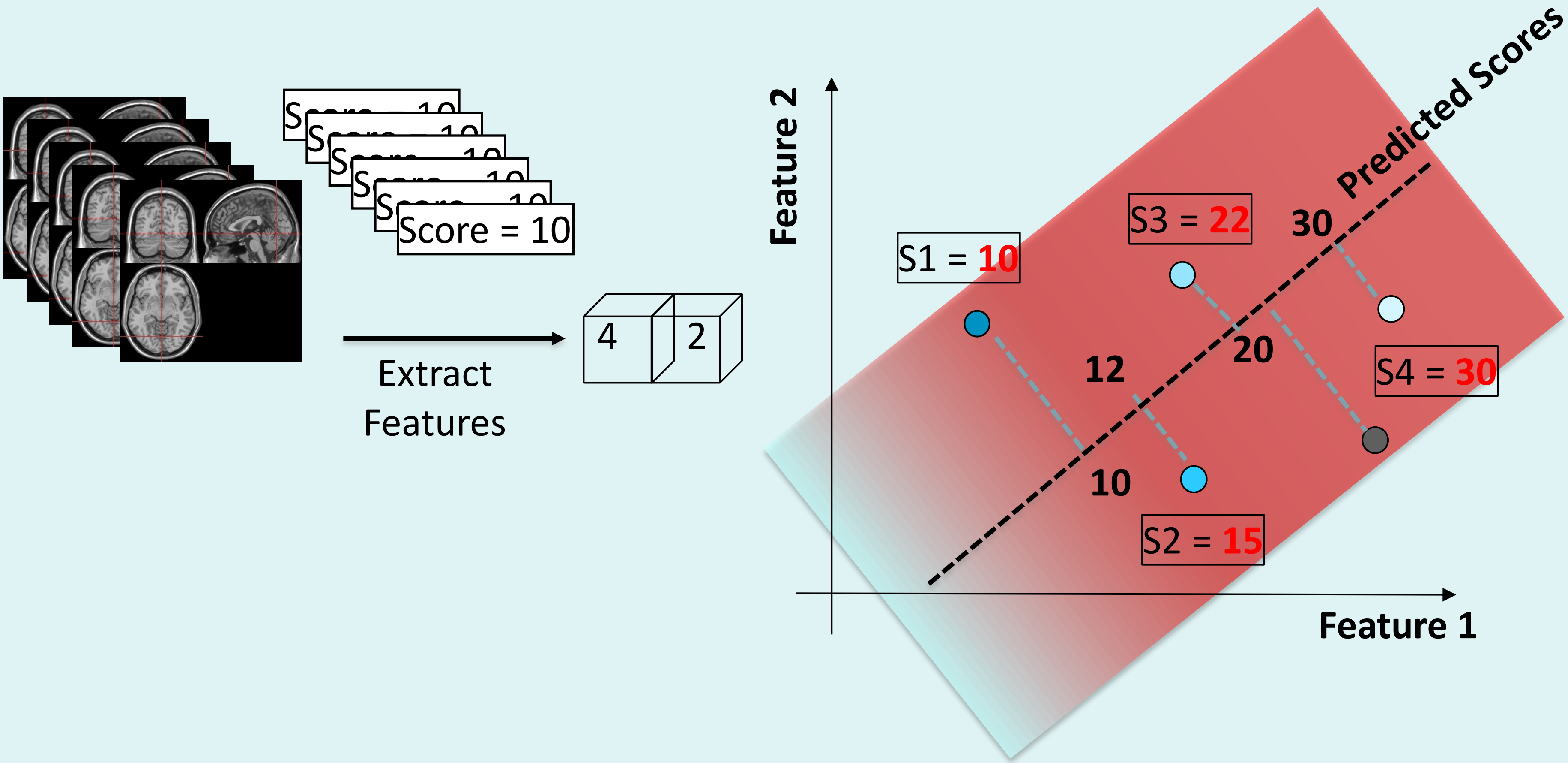
- Introduction
  - Pattern recognition
  - Univariate & multivariate approaches
  - Data representation
- Pattern Recognition
  - Machine learning
  - Validation & inference
  - Weight maps & feature selection
  - Applications: groups & fMRI
- Conclusion & Toolboxes

Classification model



Different classifiers will compute different hyper-planes!

Regression model



Linear predictive models

- Linear predictive models (classifier or regression) are parameterized by a weight vector **w** and a bias term *b*.

- The general equation for making predictions for a test example x_* is:

Parameters learned/estimated from training data

$$f(\mathbf{x}_*) = \mathbf{w} \times \mathbf{x}_* + b$$

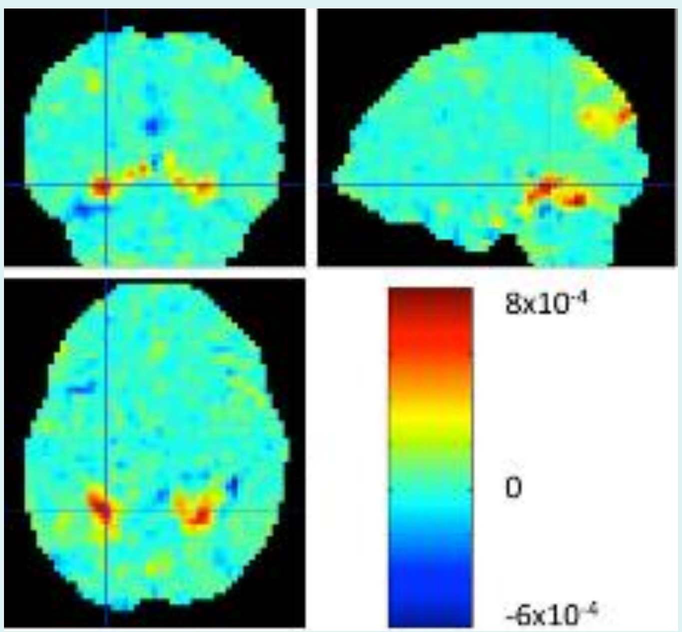
- In the linear case **w** can be expressed as a linear combination of training examples $\mathbf{x}_i$ (N = number of training examples).

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

Weight maps

= predictive patterns !

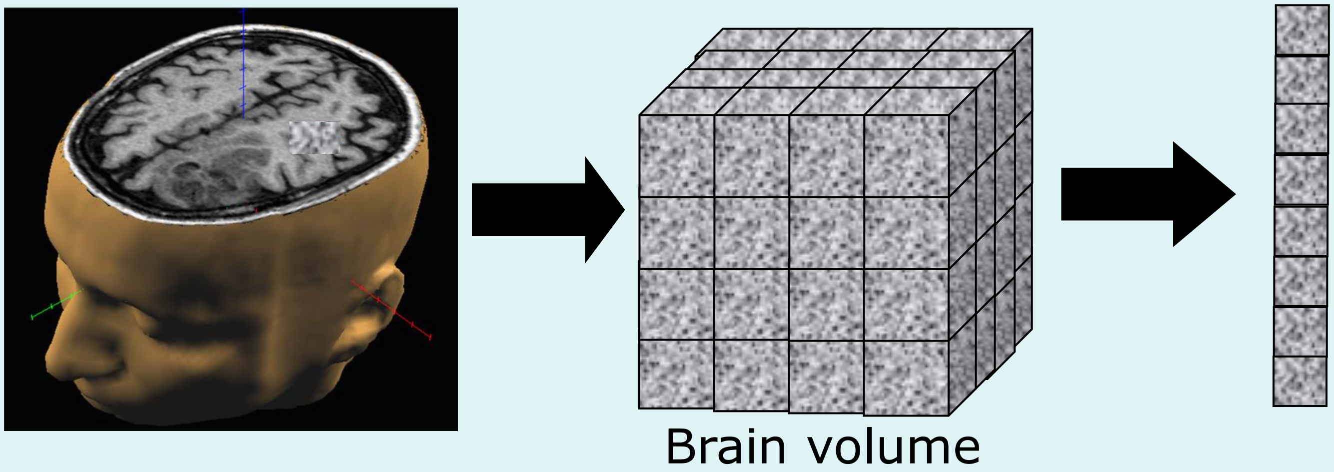
$$f(\mathbf{x}_*) = \mathbf{w} \times \mathbf{x}_* + b$$



- Shows the relative contribution of each feature for the decision

- No local inferences can be made!

Neuroimaging data



"feature vector" or "data point"

Problem: #features >> #samples

➔ "ill posed problem"

Possible solutions :

– Fewer features
  → ROIS, feature selection, searchlight
– **Regularization & Kernel Methods**

Regularization

- **Regularization** is a technique used in an attempt to **solve ill-posed problems** and to **prevent overfitting** in statistical/machine learning models.
- Regularized methods find **w** minimizing an objective function consisting of a data fit term ***E*** and a penalty/regularization term ***J***

Regularization parameter

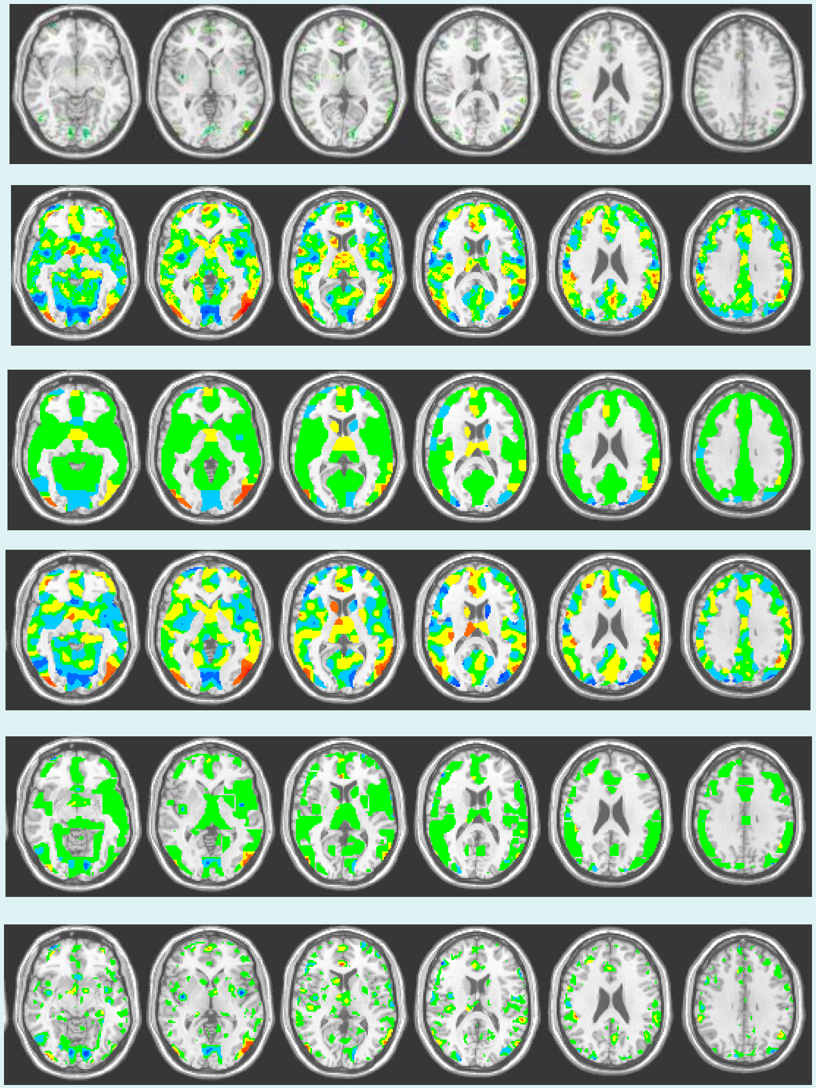
$$\min_{w \in R^p} \{E(w) + \lambda J(w)\}$$

Data fit term
= **loss function *L***

The **regularisation term *J***

- Many machine learning algorithms are particular choices of ***L*** and ***J*** (e.g. Kernel Ridge Regression (KRR), Support Vector Machine (SVM)) .

The role of regularization



LASSO
86.31%

Elastic Net
88.02%

Total Variation (TV)
85.79%

Laplacian (LAP)
83.71%

Sparse TV
85.86%

Sparse LAP
87.05%

- Weight maps for classifying fMRI images during visualization of pleasant vs. unpleasant pictures.
- **All models used a square loss + a different type of regularization.**

Baldassarre et al. (2017)

Kernel approaches

Mathematical trick!

➔ powerful and unified framework (e.g. classification & regression)

Consist of two parts:

- Use of a kernel function

  ➔ kernel matrix (mapping into the feature space)

- Learning algorithm operating with kernel

Advantages:

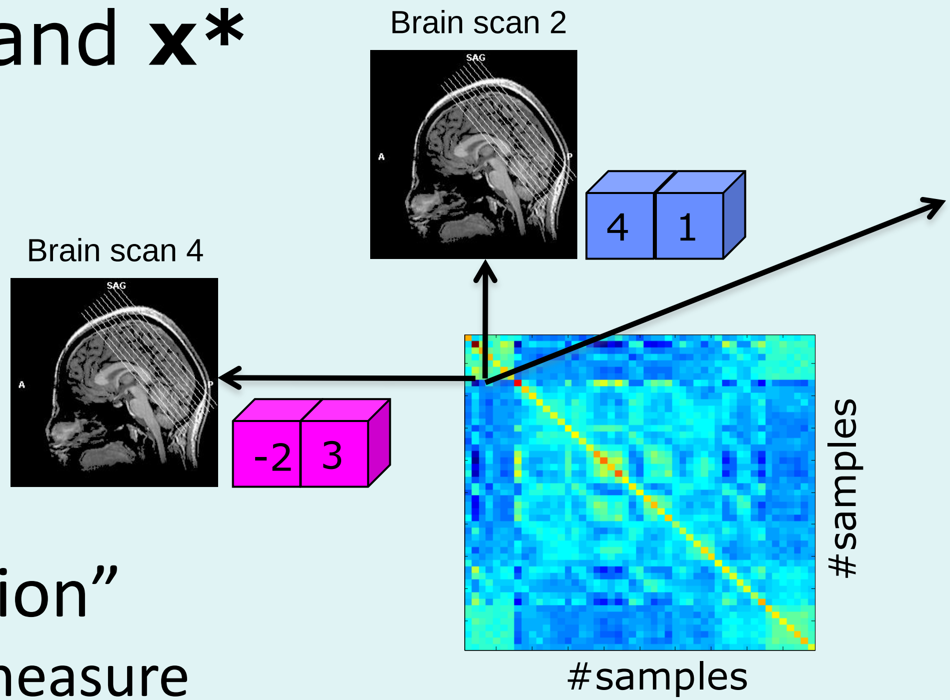
- Computational shortcut ➔ computational efficiency
- Kernel trick (linear & non-linear) + regulariaztion

  ➔ efficient solution of ill-conditioned problems.

Kernel function & matrix

Kernel matrix

= “similarity measure” between any pair of sample **x** and **x***



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

The “kernel function”

- simple similarity measure
  = a dot product → linear kernel
- more general measures
  = Gaussian, polynomial,… → non-linear kernel

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)

Example of kernel methods: Support Vector Machines (SVM), Kernel Ridge Regression (KRR), Gaussian Process (GP), Kernel Fisher Discriminant, Relevance Vector Regression

Multi-kernel learning

- Multiple Kernel Learning (**MKL**) can be applied to combine different sources of information (e.g. multimodal imaging or ROIs) for prediction.

- In MKL, the kernel **K** can be considered as a linear combination of *M* "basis kernels".

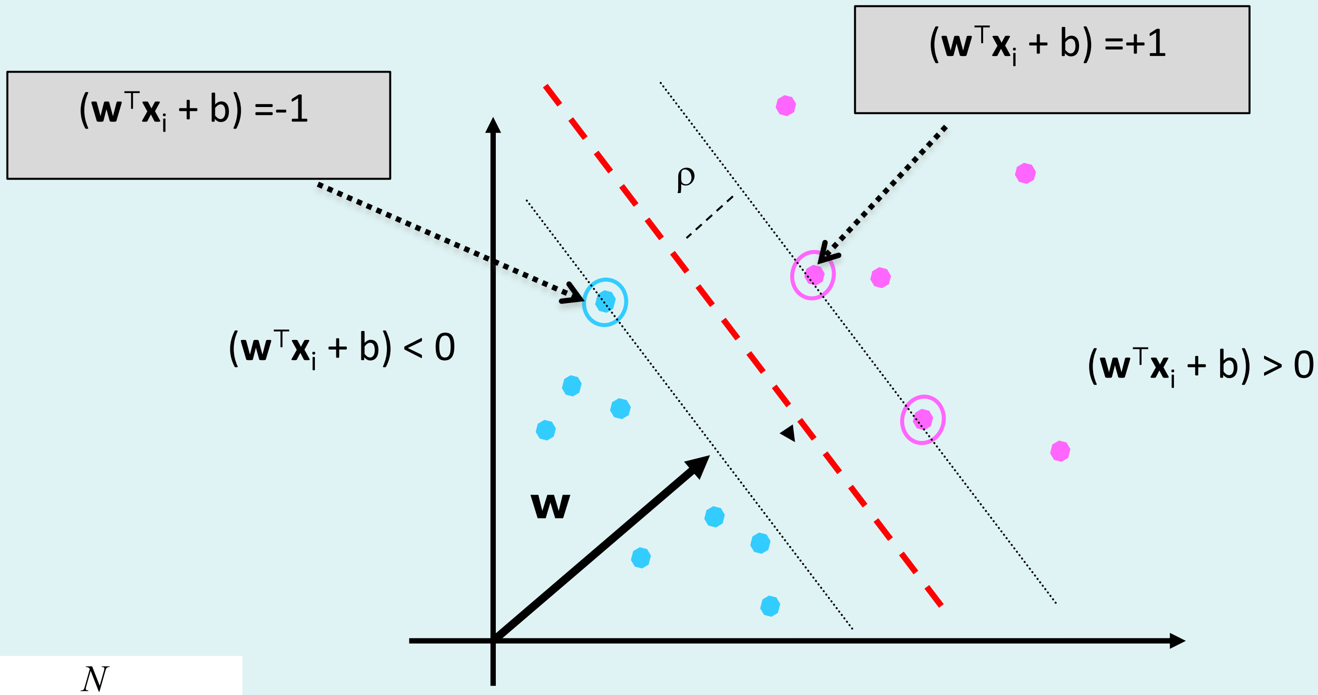
$$K(\mathbf{x}, \mathbf{x}') = \sum_{m=1}^{M} d_m K_m(\mathbf{x}, \mathbf{x}')$$

$$\text{with } d_m \geq 0, \sum_{m=1}^{M} d_m = 1$$

- MKL models simultaneously learn the kernel weights (d_m) and the associated decision function (**w**, *b*) in supervised learning settings.

Support Vector Machine

SVM = “maximum margin” classifier



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

Support vectors have $\alpha_i \neq 0$

Data: $<\mathbf{x}_i, y_i>$, $i=1,..,N$
Observations: $\mathbf{x}_i \in R^d$
Labels: $y_i \in \{-1,+1\}$

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

Other machines out there:

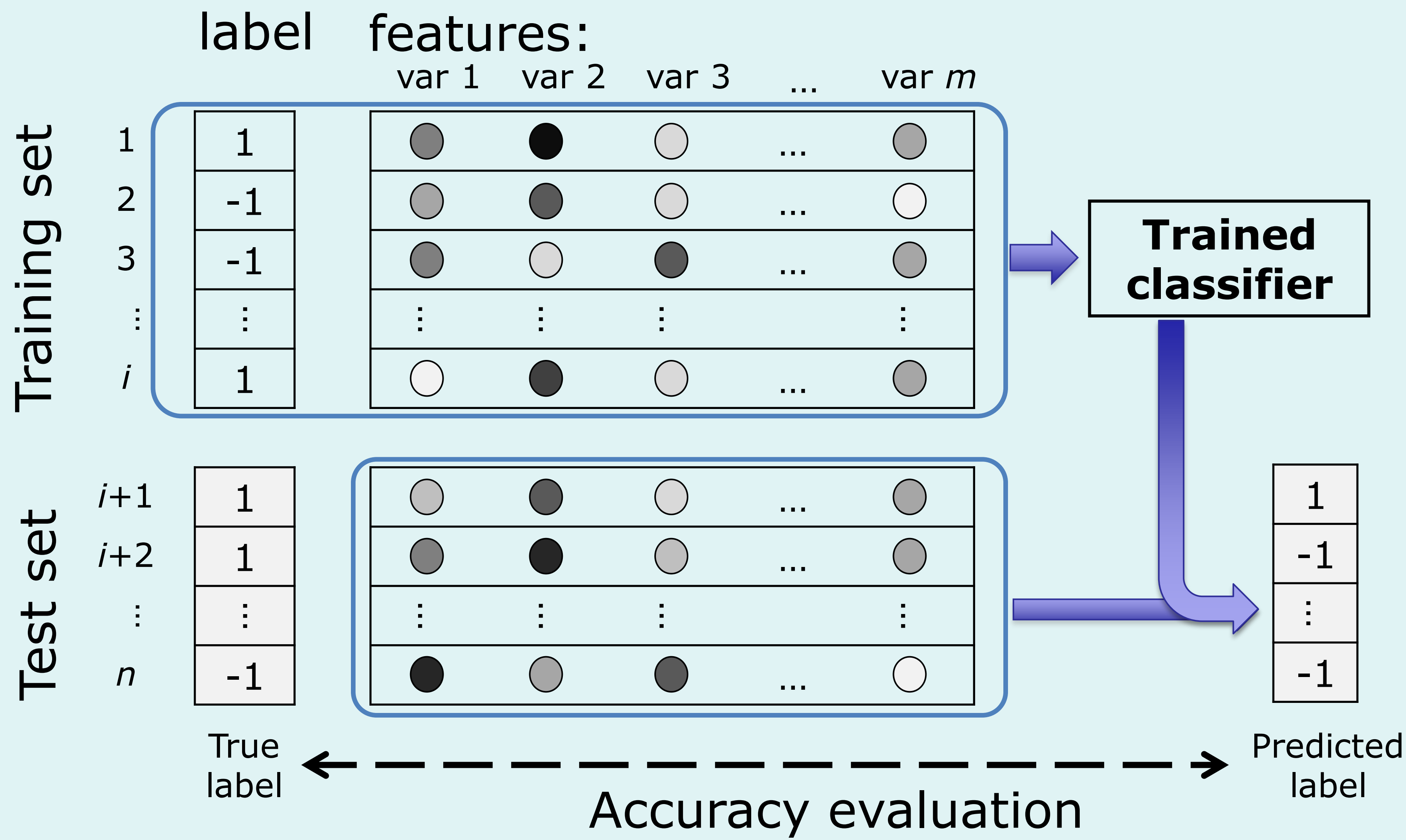
ex. tree-based, deep learning,…

Overview

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

Validation principle

Data set: Samples = {features, labels}



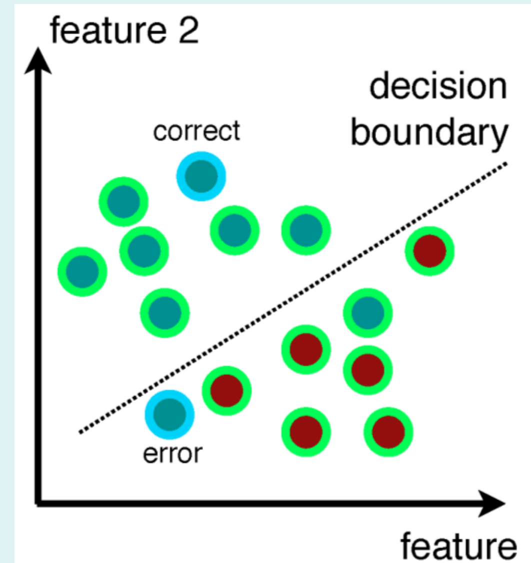
M-fold cross-validation

- 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-some-out (LSO) approach

- Accumulates metric over the M “folds”.

Confusion matrix & accuracy

		$\hat{\omega}$	
		ω_0	ω_1
truth	ω_0	A	B
	ω_1	C	D

Confusion matrix

= summary table

Accuracy estimation

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

Other criteria

- Sensitivity = D/(C+D)
- Specificity = A/(A+B)
- 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

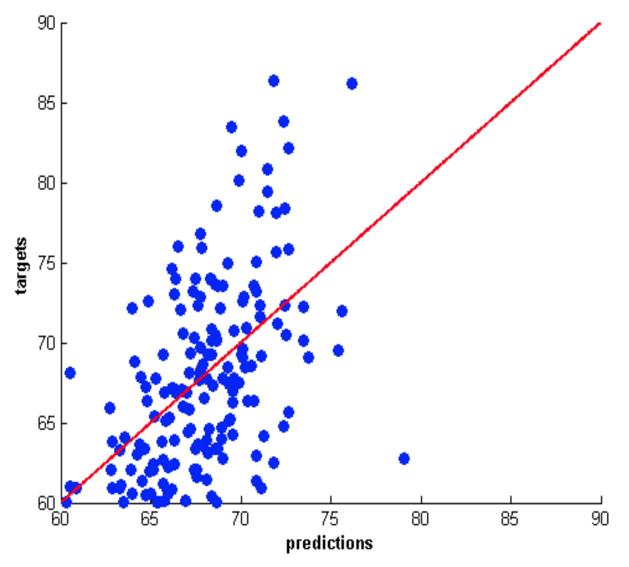
- class accuracies $[a_0, a_1, \ldots, a_C]$
- balanced accuracy $a_{bal} = (a_0 + a_1 + \ldots + a_C)/\#\text{Classes}$

Regression validation

"Mean squared error" (MSE):

- Squared error in one fold $SE_n = (y_n - f(\mathbf{x}_n))^2$
- Across all CV 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>

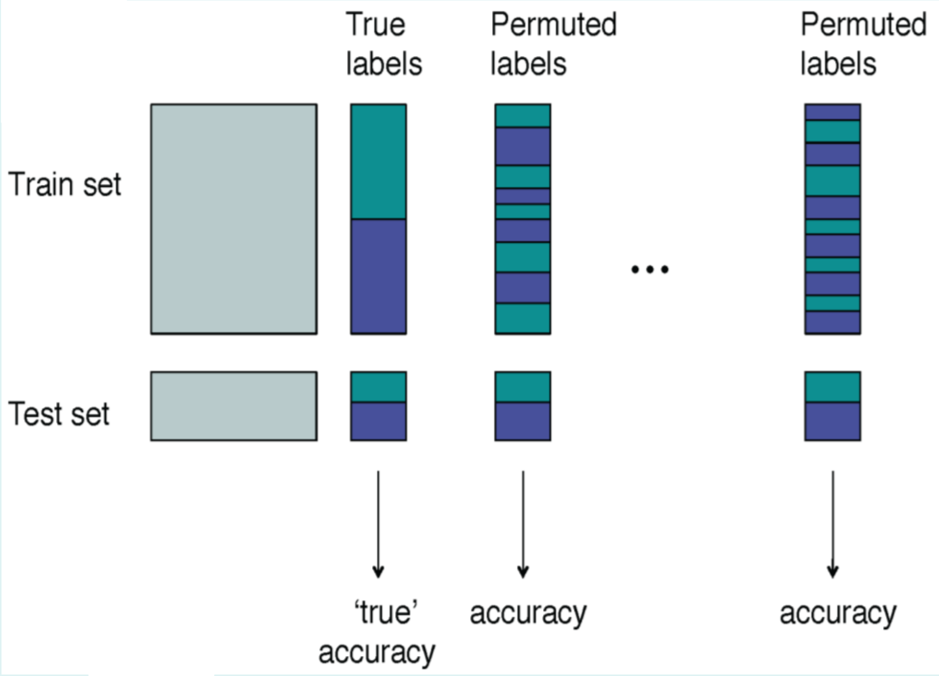
Correlation between:

- 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 H_0
  - ➔ Random permutation of labels
  - ➔ Estimate accuracy
  - ➔ Repeat M times
- Calculate p-value as

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



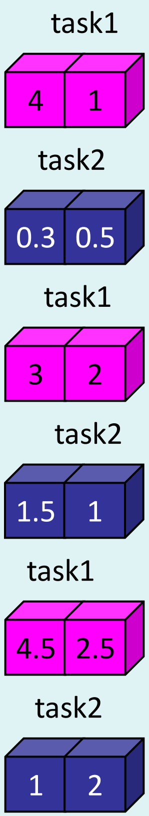
Overview

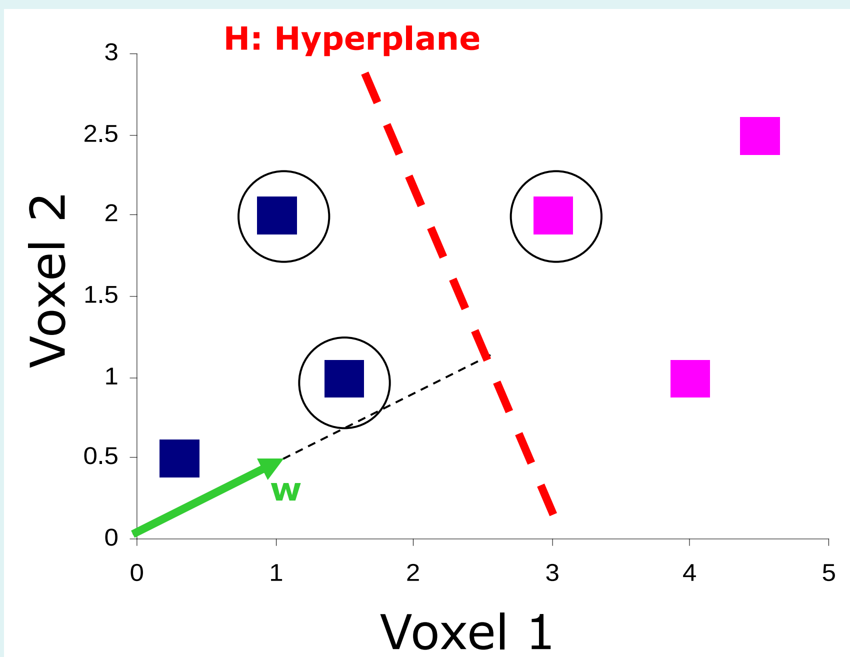
- Introduction
  - Pattern recognition
  - Univariate & multivariate approaches
  - Data representation
- Pattern Recognition
  - Machine learning
  - Validation & inference
  - Weight maps & feature selection
  - Applications: groups & fMRI
- Conclusion & Toolboxes

Weight vector interpretation

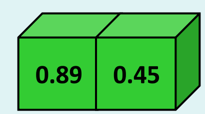
Weight vector

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





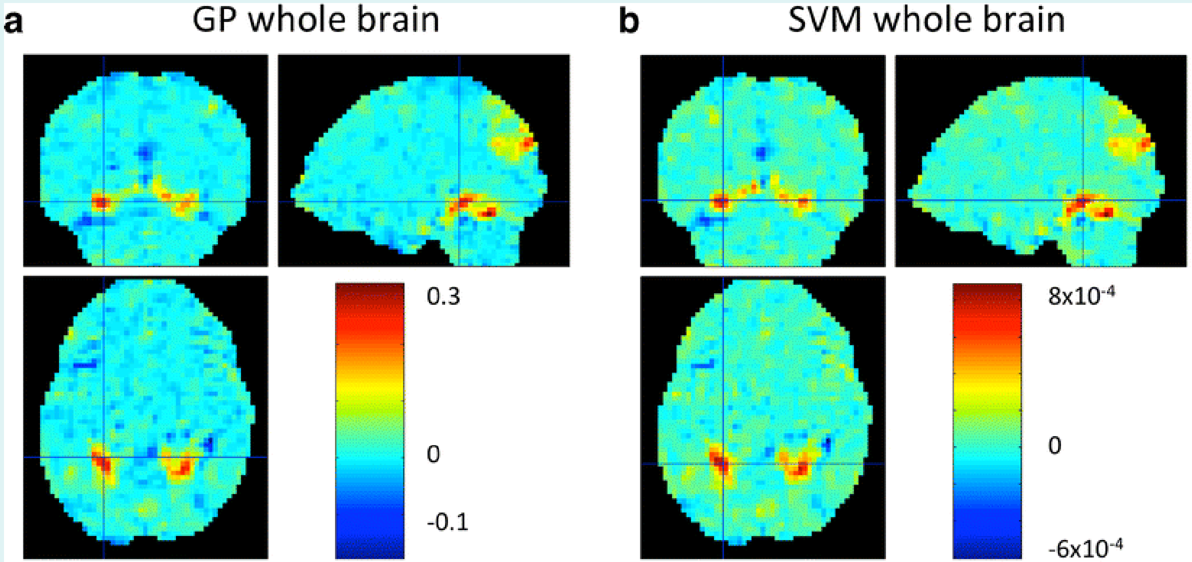
Weight vector
W = [0.89 0.45]
b = -2.8



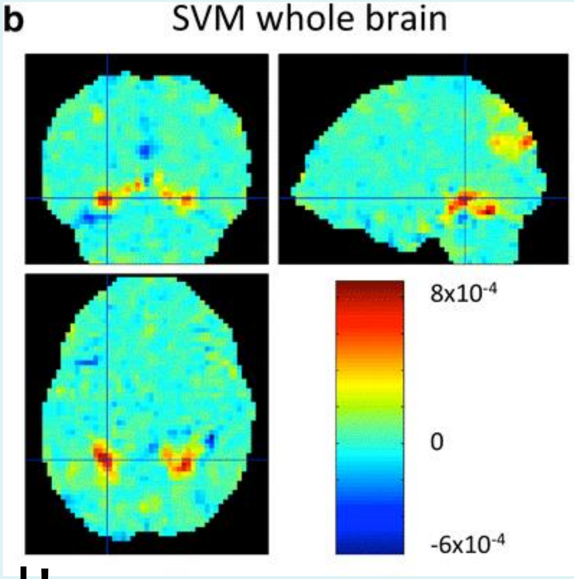
Weight maps for different masks

Linear machine
→ Weight map

Different mask/ROI
→ 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 or 'filtering'
- Multiple Kernel Learning
- Sparse methods
- (Search Light)
- Recursive Feature Elimination/Addition

MUST be independent from test data!

Overview

- Introduction
  - Pattern recognition
  - 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"

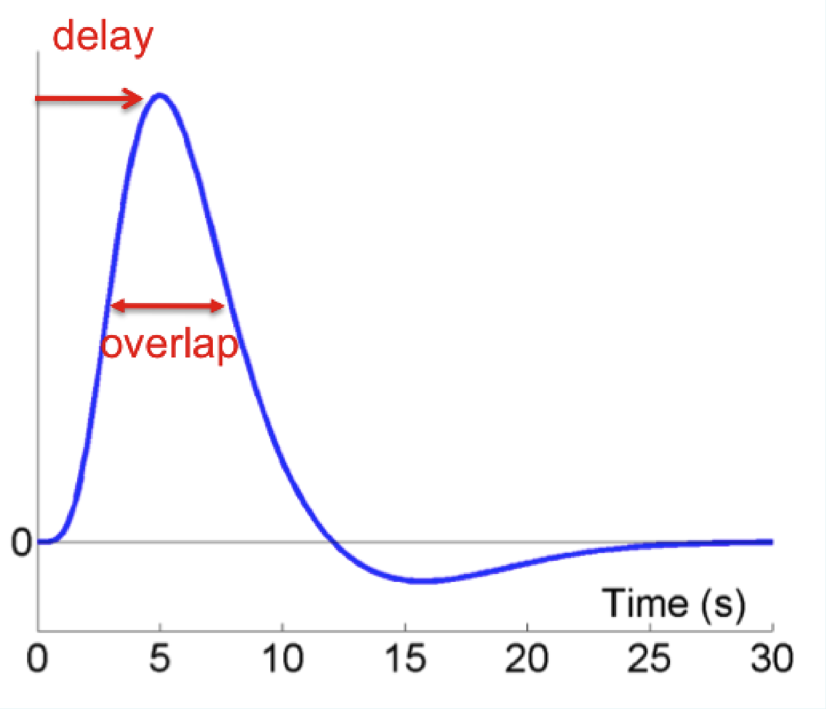
- within subject ≈ FFX with SPM
  - ➔ 'decode' subject's brain states
  - ➔ multiple images, e.g. fMRI time series

- between subjects ≈ RFX with SPM
  - ➔ 'classify' groups, e.g. patients vs. controls
    or regress subjects' parameter
  - ➔ 1 (or few) image(s)/subject

Within subject, fMRI

Activation design → decode stimuli

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

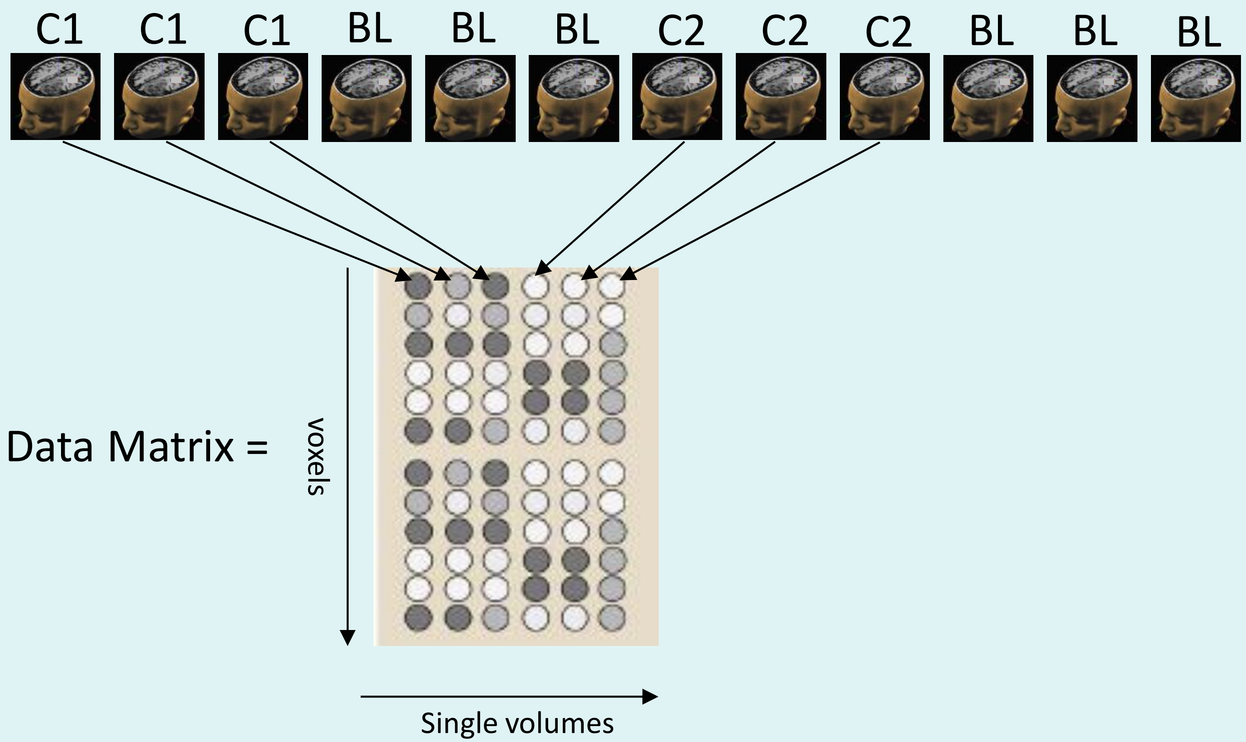


Within subject, fMRI

Rely on raw BOLD signal per event/block

$\rightarrow$ one label per image !

- 1 volume = 1 sample

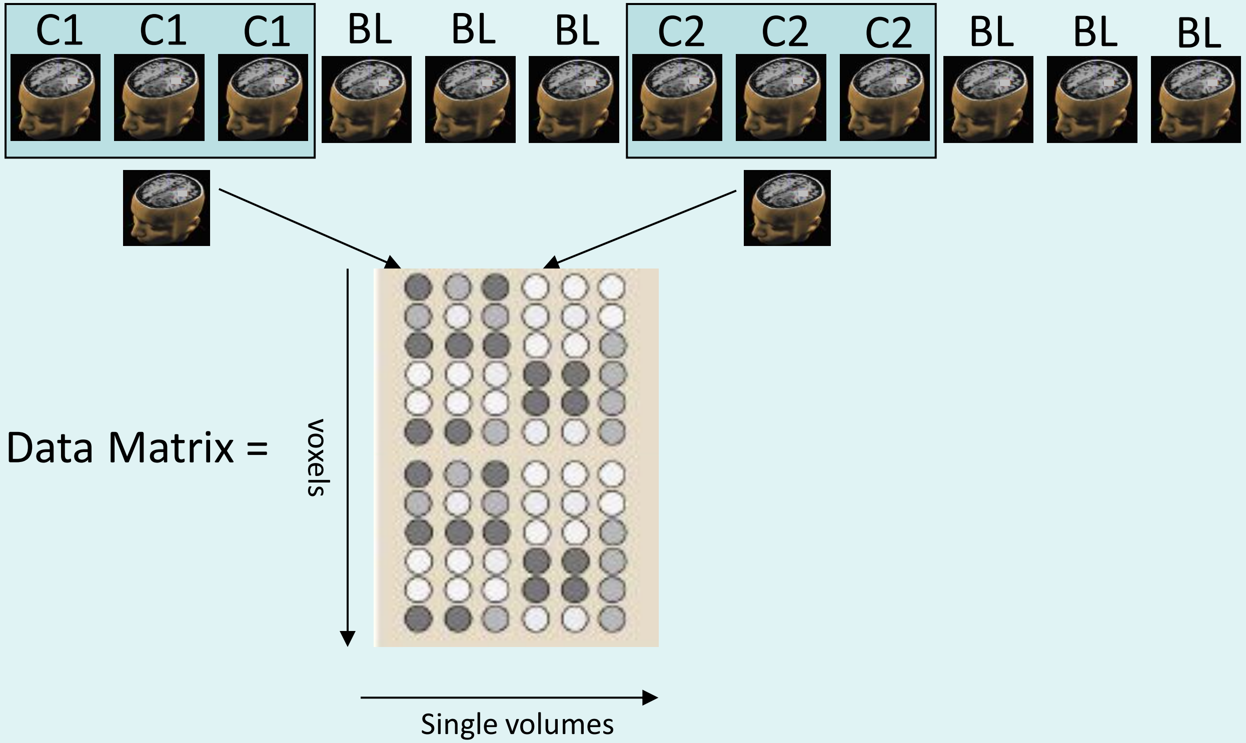


Within subject, fMRI

Rely on raw BOLD signal per event/block

$\rightarrow$ one label per image !

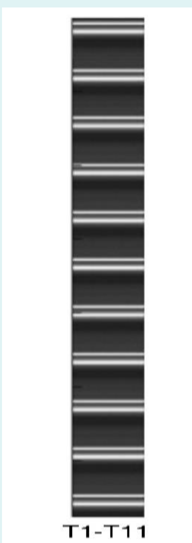
- 1 volume = 1 sample, or
- average over N volumes



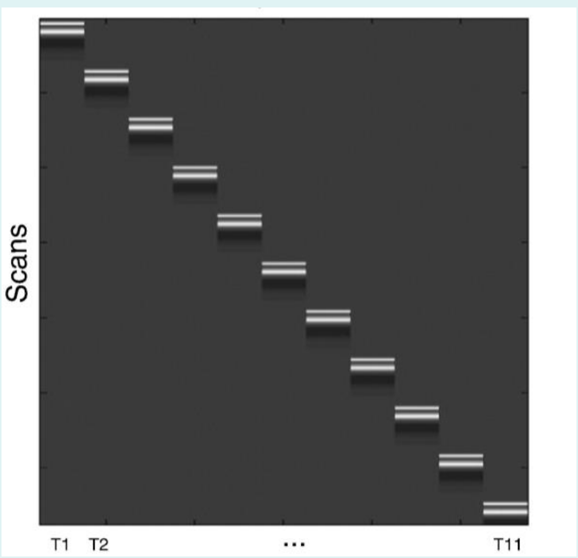
Within subject, fMRI

Rely on contrast image per event/block

- 1 contrast = 1 sample
- implicit averaging

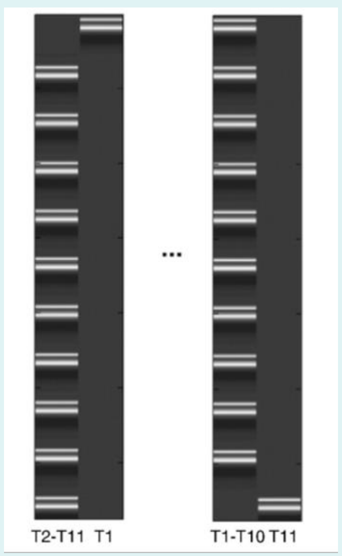


LSA



LSU

or



LSS

"Least Squares All" (LSA)
"Least Squares Unitary" (LSU)
"Least Squares Separate" (LSS)

Abdulrahman & Henson, Neuroimage, 2016

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

Example:

- contrast (a-fMRI), ICA/correlation map (rs-fMRI)
- GM/Jacobian maps (sMRI)
- FA/MD maps (DWI)
- PET
- etc.

Overview

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

"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

Conclusions

<u>Key points:</u>

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

Existing toolboxes

In Matlab

- The Decoding Toolbox, https://sites.google.com/site/tdtdecodingtoolbox/
- Pattern Component Modelling Toolbox (PCMtoolbox), https://github.com/jdiedrichsen/pcm_toolbox
- MVPA by cross-validated MANOVA, https://github.com/allefeld/cvmanova
- Princeton Multi-Voxel Pattern Analysis (MVPA) Toolbox, https://github.com/princetonuniversity/princeton-mvpa-toolbox

In Python

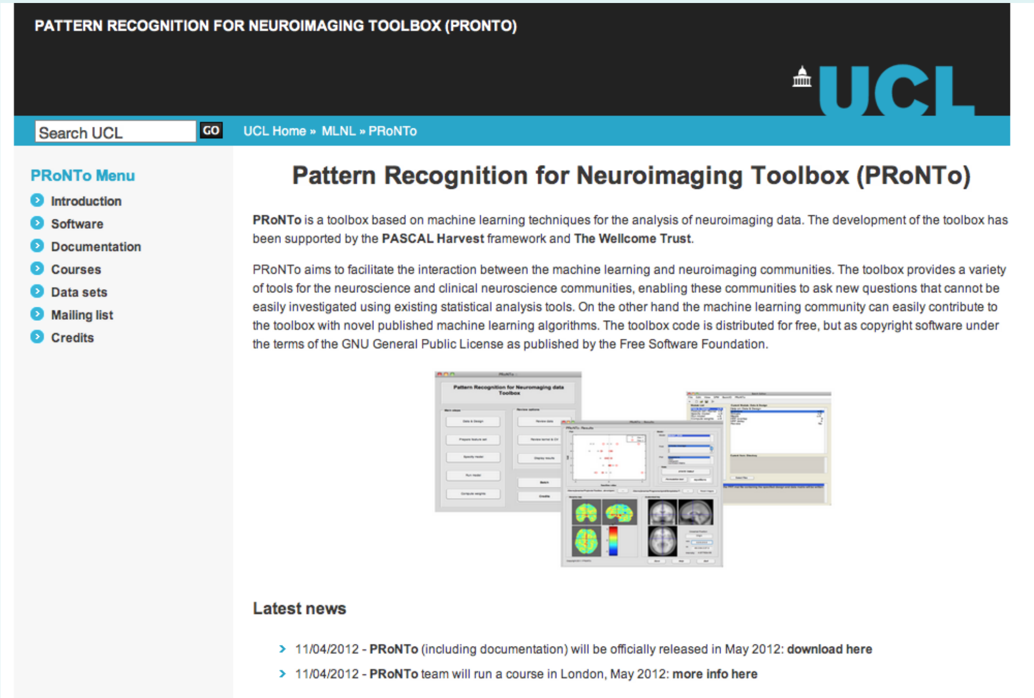
- pyMVPA, http://www.pymvpa.org/
- Nilearn, http://nilearn.github.io/
- Brain Imaging Analysis Kit (BrainAIK), https://brainiak.org/

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.



Schrouff et al, 2013.

Afternoon workshop

More about

- Weight interpretation
- Machines & “multi-kernel learning”
- Nested CV & parameter optimization
- Feature extraction
- …

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

- Baldassare L, et al. (2017). Sparsity Is Better with Stability: Combining Accuracy and Stability for Model Selection in Brain Decoding. Front. Neurosci.
- Hastie T, Tibshirani R & Friedman J. The Elements of Statistical Learning 2009. Springer Series in Statistics.
- Haynes JD, Rees G (2006) Decoding mental states from brain activity in humans. *Nat Rev Neurosci.*
- Mourão-Miranda J 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 Q, et al. (2014). Biased binomial assessment of cross-validated estimation of classification accuracies illustrated in diagnosis predictions. *Neuroimage Clin*. 4, 687–694.
- Pereira F, Mitchell TM, Botvinick M (2009) Machine Learning Classifiers and fMRI: a tutorial overview. *Neuroimage*
- Rakotomamonjy A et al. (2008) Simple MKL. *Journal of Machine Learning*, 2491-2521.
- Rasmussen C, Williams CKI (2006) Gaussian Processes for Machine Learning. Cambridge, Massachusetts: The MIT Press.
- Shawe-Taylor J, Christianini N (2004) Kernel Methods for Pattern Analysis. Cambridge: Cambridge University Press.
- Schrouff J et al. (2013) PRoNTo: Pattern Recognition for Neuroimaging Toolbox, *Neuroinformatics*.
- Schrouff J et al (2018) Embedding anatomical or functional knowledge in whole-brain multiple kernel learning models. *Neuroinformatics*.