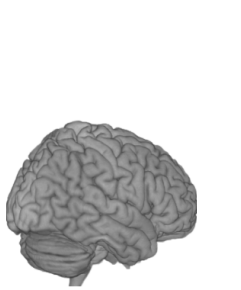
Kernel Methods for Pattern Recognition

Janaina Mourao-Miranda,
Machine Learning and Neuroimaging Lab,
University College London, UK

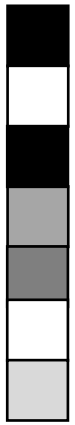
Notation

- Examples ($\mathbf{x}_i$)



fMRI/sMRI
3D matrix of voxels

Feature Vector



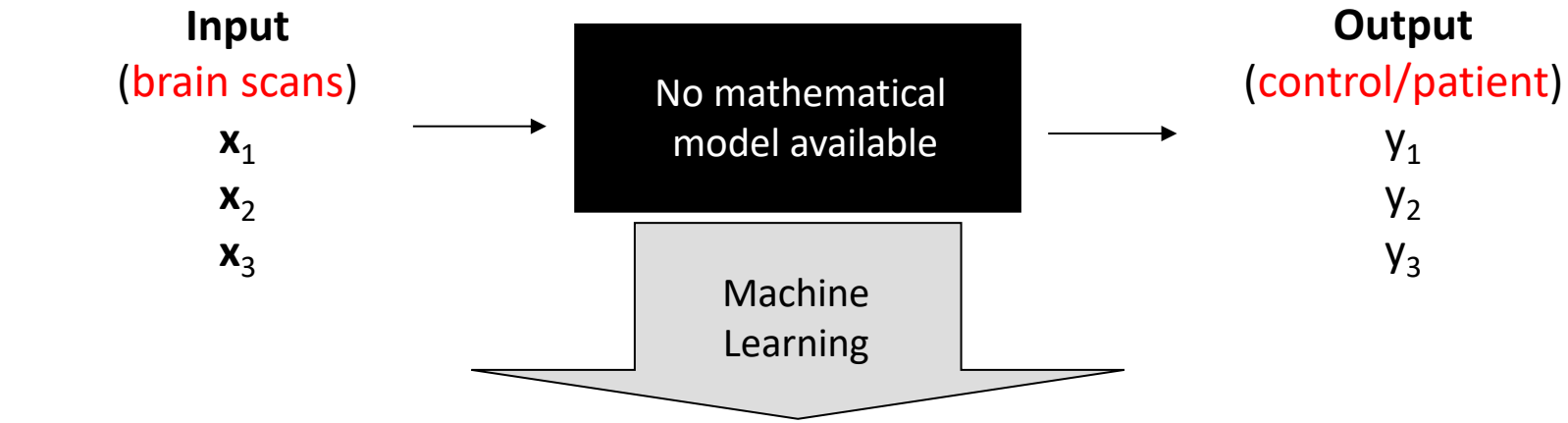
$\mathbf{x}_i$ is a vector of size dx1 where
d is the number of voxels

- Labels (y_i)
  - categorical value for classification (e.g. class 1 = patients, class 2 = healthy controls)
  - continuous value for regression (e.g. age or clinical scale).

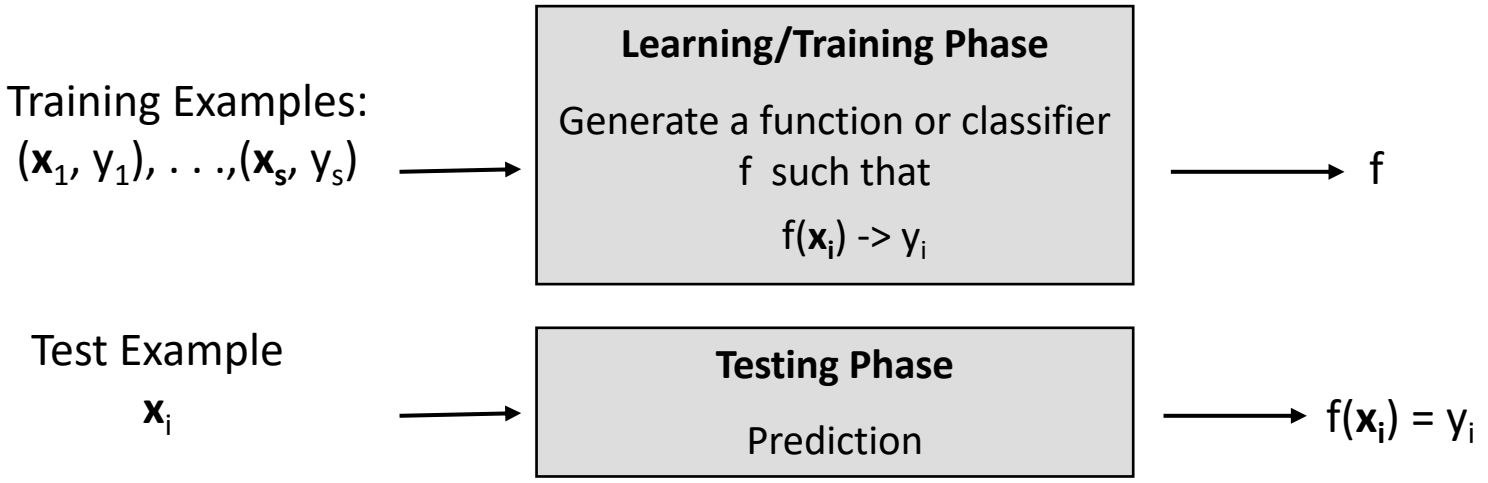
- Matrix notation (one example per row)

$$\mathbf{X} = [\mathbf{x}_1\ \mathbf{x}_2\ \ldots\ \mathbf{x}_N]^T$$
$$\mathbf{y} = [y_1\ y_2 \ldots\ y_N]^T$$

Pattern Recognition Framework



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



Linear models

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

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

where $f(\mathbf{x}_*)$ is the predicted score for regression or the distance to the decision boundary for classification models.

- The weight vector can be expressed as a linear combination of training examples $\mathbf{x}_i$ (where $i = 1,\ldots,N$ and N is the number of training examples).

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

Pattern Recognition in Neuroimaging

Main difficulties:

- Very high dimensional data: computational issues
- Often the dimensionality of the data is greater than the number of examples: ill-conditioned problems

Potential Solutions:

- Feature Selection
- Region of Interest
- Searchlight
- **Kernel Methods + Regularisation -> PRoNTo**

Kernel Methods

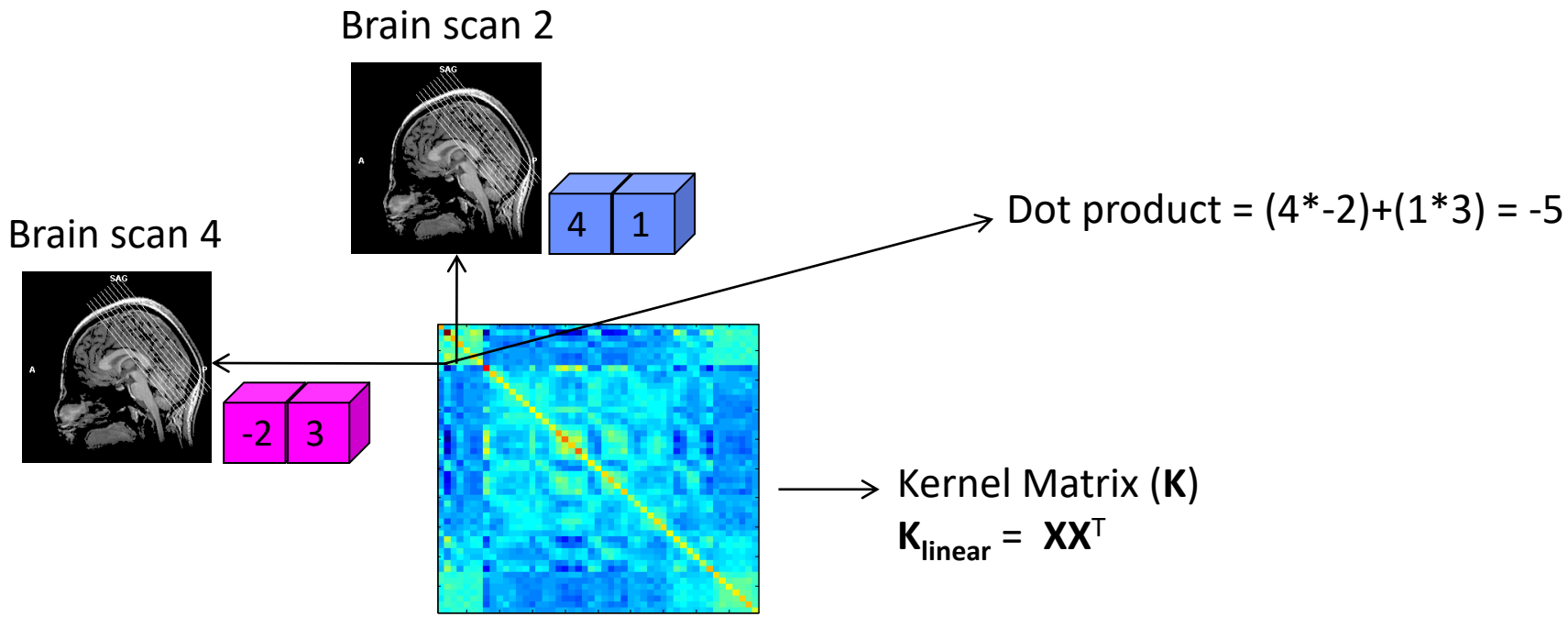
- The kernel methodology provides a powerful and unified framework for investigating general types of relationships in the data (e.g. classification, regression, etc).

- **Kernel methods consist of two parts:**
  - ✓ Computation of the kernel matrix (mapping into the feature space).
  - ✓ A learning algorithm based on the kernel matrix (designed to discover linear patterns in the feature space).

- **Advantages:**
  - ✓ Represent a **computational shortcut** which makes possible to represent linear patterns efficiently in high dimensional space.
  - ✓ Using the dual representation with proper **regularization*** enables efficient solution of ill-conditioned problems.

* e.g. restricting the choice of functions to favor functions that have small norm.

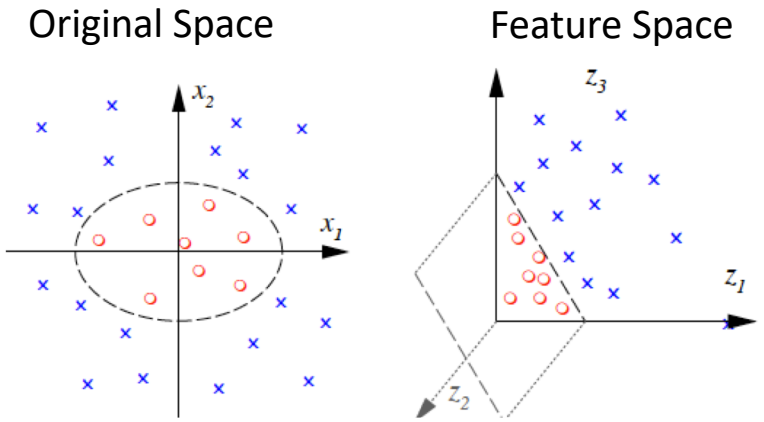
Kernel Function ("similarity measure")



• Kernel is a function that, for given two pattern **x** and **x***, returns a real number characterizing their similarity.

•A simple type of similarity measure between two vectors is a dot product (linear kernel).

Nonlinear Kernels



• There are more general “similarity measures”, i.e. nonlinear kernels: Gaussian kernel, Polynomial kernel, etc.

• Nonlinear kernels are used to map the data to a higher dimensional space as an attempt to make it linearly separable.

• The kernel trick enables the computation of similarities in the feature space without having to compute the mapping explicitly.

Advantage of linear models

- Neuroimaging data are extremely high-dimensional and the sample sizes are very small, therefore non-linear kernels often don’t bring any benefit.

- Linear models reduce the risk of overfitting the data and allow direct extraction of the weight vector as an image (i.e. predictive map).

Learning with kernels

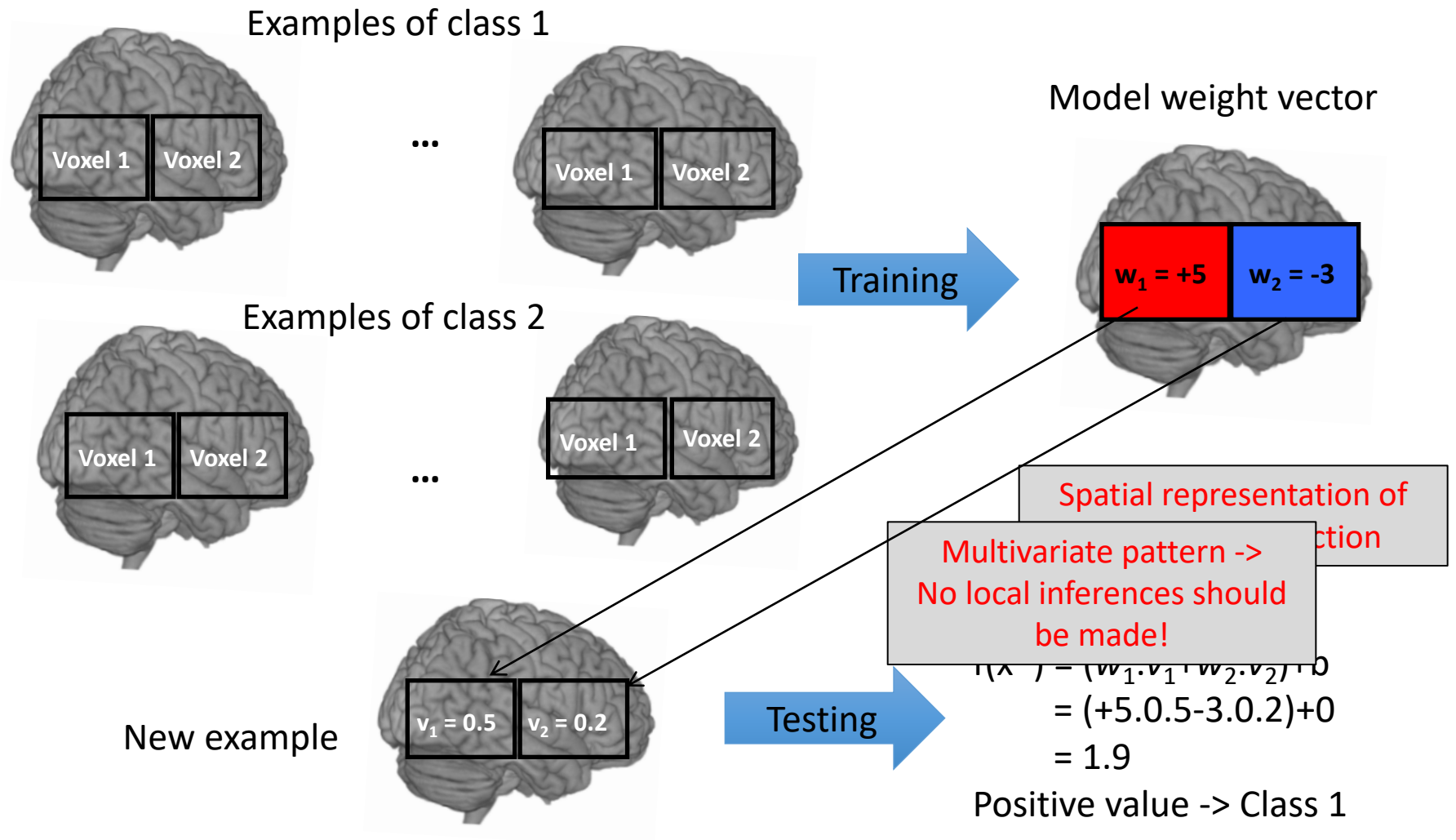
- Making predictions with kernel methods

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

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

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

How to interpret the weight vector (w)?



Examples of Kernel Methods in PRoNTo

- **Support Vector Machines (SVM)**
- Gaussian Processes (GP)
- **Kernel Ridge Regression (KRR)**
- Relevance Vector Regression (RVR)
- **Multiple Kernel Learning (MKL)**

Example of Kernel Methods

(1) Support Vector Machine

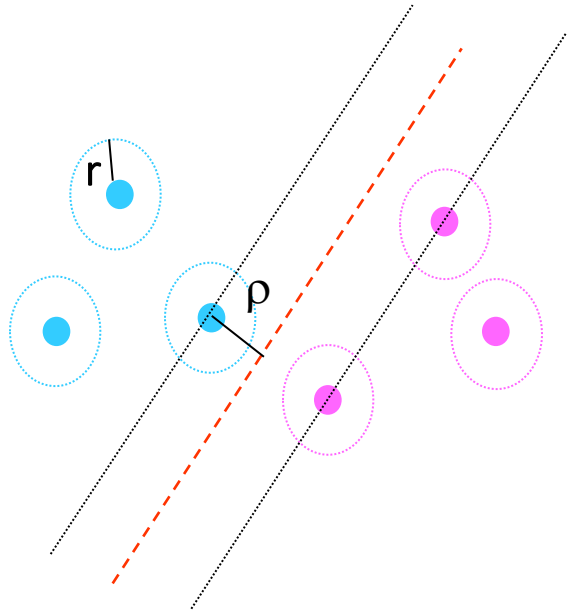
Support Vector Machines (SVMs)

- A classifier derived from statistical learning theory by Vapnik, et al. in 1992.
- SVM became famous when, using images as input, it gave accuracy comparable to neural-network in a handwriting recognition task.
- Currently, SVM is widely used in object detection & recognition, text recognition, biometrics, speech recognition, neuroimaging, etc.
- Also used for regression.

Largest Margin Classifier

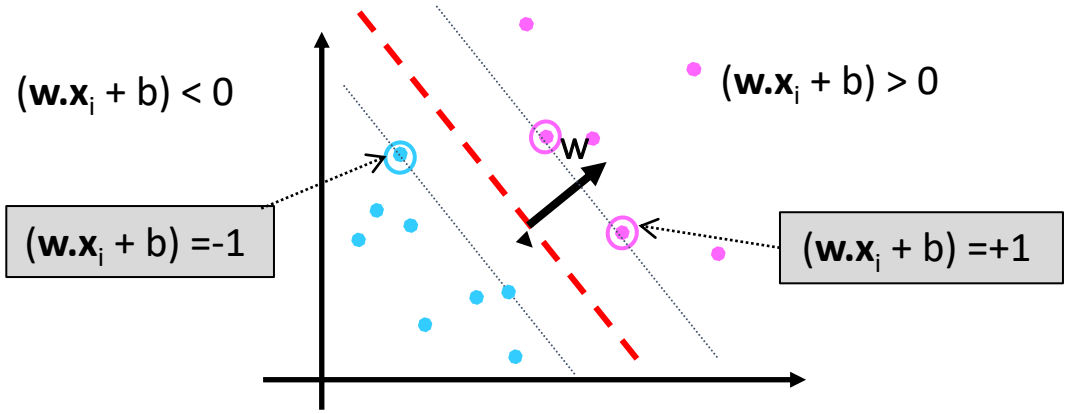
• Among all hyperplanes separating the data there is a unique optimal hyperplane, the one which presents the largest margin (the distance of the closest points to the hyperplane).

• Let us consider that all test points are generated by adding bounded noise (**r**) to the training examples (test and training data are assumed to have been generate by the same underlying dependence).



• If the optimal hyperplane has margin $\rho > r$ it will correctly separate the test points.

Linearly separable case (Hard Margin SVM)



- We assume that the data are linearly separable, that is, there exist $\mathbf{w} \in IR^d$ and $b \in IR$ such that $y_i(\mathbf{w}.\mathbf{x}_i + b) > 0$, i = 1,...,m.

- Rescaling **w** and *b* such that the points closest to the hyperplane satisfy $|(\mathbf{w}.\mathbf{x}_i + b)| = 1$ we obtain the canonical form of the hyperplane satisfying $y_i(\mathbf{w}.\mathbf{x}_i + b) > 0$

- The distance of a point $\mathbf{x}_i$ to a hyperplane $H_{w,b}$ is given by $\rho_x = |(\mathbf{w}.\mathbf{x}_i + b)|/||\mathbf{w}||$

- The distance from the closest point to the canonical hyperplane is $\rho = 1/||w||$.

- In this case, the margin, measured perpendicularly to the hyperplane, equals $2/||w||$.

- In order to maximize the margin we need to minimize $||w||/2$.

Linearly separable case (Hard Margin SVM)

- Constrained optimization problem

$$\min \frac{1}{2} \| \mathbf{w} \|^2$$

Quadratic problem:
unique optimal solution

$$\text{s.t. } y_i(\mathbf{w}.\mathbf{x}_i + b) \geq 1, i = 1,..,m$$

- The solution of this problem is equivalent to determine the saddle point of the Lagrangian function

$$L(\mathbf{w}, b; \alpha) = \frac{1}{2} \| \mathbf{w} \|^2 - \sum_{i=1}^{N} \alpha_i \{ y_i(\mathbf{w}.\mathbf{x}_i + b) - 1 \}$$

where $\alpha_i \geq 0$ are the Lagrange multipliers.

- We minimize *L* over (**w**,*b*) and maximize over α.

Linearly separable case (Hard Margin SVM)

Differentiating *L* w.r.t. **w** and *b* we obtain:

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{N} y_i \alpha_i = 0$$

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$$

Substituting **w** in *L* leads to the dual problem

$$\max \; Q(\alpha) := -\frac{1}{2}\alpha^{\mathrm{T}} \mathbf{A} \alpha + \sum_i \alpha_i$$

$$\text{s.t.} \sum_i y_i \alpha_i = 0$$

$$\alpha_i \geq 0, \; \mathrm{i}=1,\ldots,N$$

where **A** is an $N \times N$ matrix and $\mathbf{A} = (y_i y_j \mathbf{x}_i . \mathbf{x}_j : i, j = 1, \ldots, N)$

Note that the complexity of this problem depends on *N* (number of examples), not on the number of input components *d* (number of dimensions).

Linearly separable case (Hard Margin SVM)

If $\bar{\alpha}$ is a solution of the dual problem then the solution (**w**, b) of the primal problem is given by

$$\bar{\mathbf{w}} = \sum_{i=1}^{N} \bar{\alpha}_i y_i \mathbf{x}_i$$

Note that $\bar{\mathbf{w}}$ is a linear combination of only the $\mathbf{x}_i$ for which $\alpha_i > 0$. These $\mathbf{x}_i$ are called support vectors (SVs).

Parameter *b* can be determined by $b = y_i - \mathbf{w} \cdot \mathbf{x}_i$, where $\mathbf{x}_i$ corresponds to a SV.

A new point $\mathbf{x}_*$ is classified as

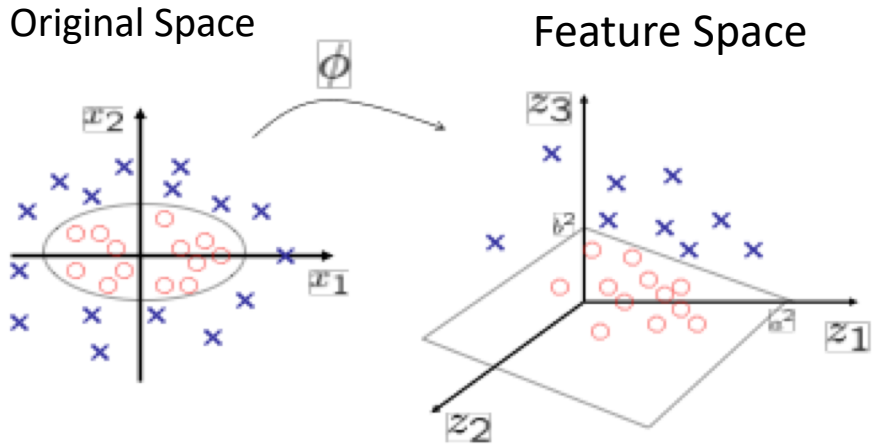
$$f(\mathbf{x}_*) = \operatorname{sgn}\left(\sum_{i=1}^{N} y_i \bar{\alpha} (\mathbf{x}_i . \mathbf{x}_*) + \bar{b} \right)$$

The dot product is simple type of similarity measure

Kernel Trick

- The dot product can be replaced by a kernel function which corresponds to a dot product in the feature space.

- The kernel trick is a way of mapping observations from the original space into a feature space, without ever having to compute the mapping explicitly.



Some remarks

- The fact that that the Optimal Separating Hyperplane is determined only by the support vectors is most remarkable. Usually, the support vectors are a small subset of the training data.

- **All the information contained in the data set is summarized by the support vectors.** The whole data set could be replaced by only these points and the same hyperplane would be found.

Soft Margin SVM

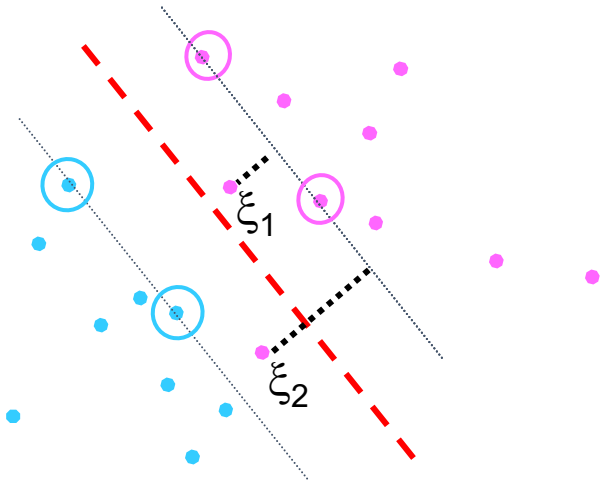
- If the data is not linearly separable the previous analysis can be generalized by looking at the problem

$$\min \frac{1}{2} \| \mathbf{w} \|^2 + C \sum_{i=1}^{N} \xi_i$$

$$\text{s.t.} \quad y_i (\mathbf{w}.\mathbf{x}_i + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0, \qquad i = 1,..,N$$

- The idea is to introduce the slack variables ξ_i to relax the separation constraints ($\xi_i > 0 \Rightarrow x_i$ has margin less than 1).



New dual problem

- A saddle point analysis (similar to that above) leads to the dual problem

$$\max \; Q(\alpha) := -\frac{1}{2}\alpha^{\mathrm{T}}\mathbf{A}\alpha + \sum_i \alpha_i$$

$$\text{s.t.} \sum_i y_i \alpha_i = 0$$

$$0 \le \alpha_i \le C, \qquad i = 1,\ldots,N$$

- This is like the previous dual problem except that now we have "box constraints" on α_i.

- Again we have $\overline{\mathbf{w}} = \sum_{i=1}^{N} \overline{\alpha}_i y_i \mathbf{x}_i$

The role of the parameter C

• The parameter C that controls the relative importance of minimizing the norm of **w** (which is equivalent to maximizing the margin) and satisfying the margin constraint for each data point.

•If C is close to 0, then we don't pay that much for points violating the margin constraint. This is equivalent to creating a very wide tube or safety margin around the decision boundary (but having many points violate this safety margin).

•If C is close to inf, then we pay a lot for points that violate the margin constraint, and we are close the hard-margin formulation we previously described - the difficulty here is that we may be sensitive to outlier points in the training data.

•C is often selected by cross-validation (nested cross-validation in PRoNTo).

Summary

•SVMs are prediction devices known to have good performance in high-dimensional settings.

• "The key features of SVMs are the use of kernels, the absence of local minima and the sparseness of the solution.” Shawe-Taylor and Cristianini (2004).

Example of Kernel Methods

(1) Multiple Kernel Learning (MKL)

Motivation for Multiple Kernel Learning

- Many practical learning problems involve multiple, heterogeneous data sources.
- It seems advantageous to combine different sources of information for prediction (e.g. multimodal imaging for diagnosis/prognosis).
- Need to learn with not only a single kernel but with multiple kernels.

Multiple Kernel Learning (MKL)

- Multiple Kernel Learning (MKL) has been proposed as an approach to simultaneously learn the kernel weights and the associated decision function in supervised learning settings.

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

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

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

- The decision function of an MKL problem can be then expressed in the form:

$$f(\mathbf{x}_*) = \sum_{i=1}^{m} \mathbf{w}_m \cdot \mathbf{x}_* + b$$

Multiple Kernel Learning (MKL)

- One example of MKL approach based on SVM is the SimpleMKL (Rakotomamonjy, et al. 2008).
- SimpleMKL optimization problem

$$\min \frac{1}{2}\sum_{m=1}^{M}\frac{1}{d_m}\| \mathbf{w}_m \|^2 + C\sum_{i=1}^{N}\xi_i$$

$$\text{s.t. } y_i\left(\sum_{m=1}^{M}\mathbf{w}_m.\mathbf{x}_i + b\right) \geq 1-\xi_i, \qquad i = 1,..,N$$

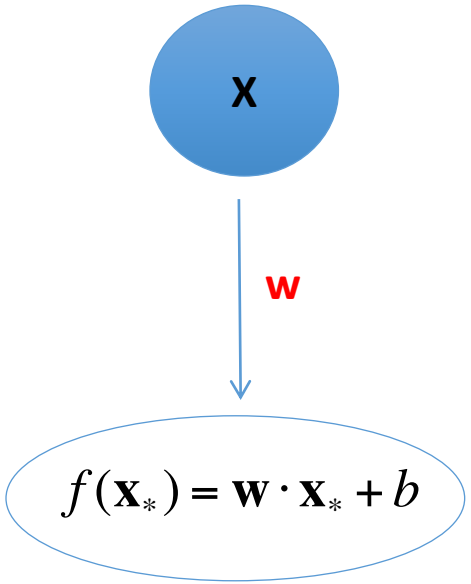
$$\xi_i \geq 0, \qquad i = 1,..,N$$

$$\sum_{m=1}^{M} d_m = 1, \; d_m \geq 0, \qquad m = 1,..,M$$

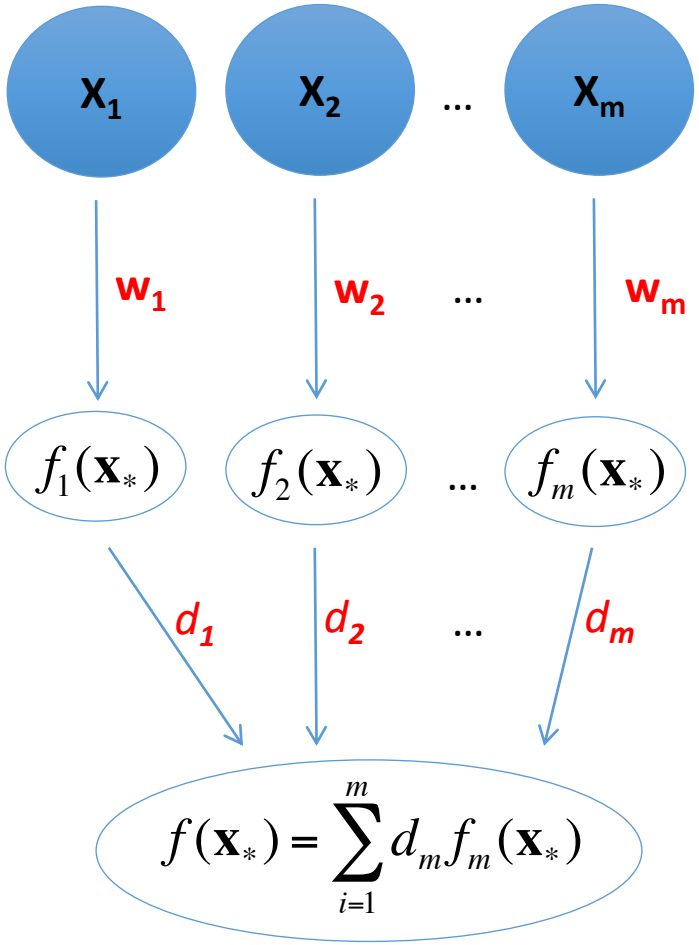
the L1 constrain on d_m enforces sparsity on the kernels with a contribution to the model.

Single vs. Multiple Kernel Learning

Single kernel SVM



Multiple kernel SVM



Example of Kernel Methods

(3) Kernel Ridge Regression

Kernel Methods

- The general equation for making predictions with kernel methods is

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

  where $f(\mathbf{x}_*)$ is the predicted score for regression or the distance to the decision boundary for classification.

- α_i is the dual weight vector and b is a constant offset, both of which are learnt from the training samples.

- We can simplify the equation for making predictions by adding a constant element to $\mathbf{x}_*$, so that $\mathbf{x}_* = [\mathbf{x}_* \; 1]^T$ and $\mathbf{w}=[\mathbf{w} \; b]^T$

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

Kernel Ridge Regression

- Kernel ridge regression is the dual representation of ridge regression, which is sometimes known as the linear Least Square Regression (LSR) with Tikhonov regularization (Chu et al. 2011).

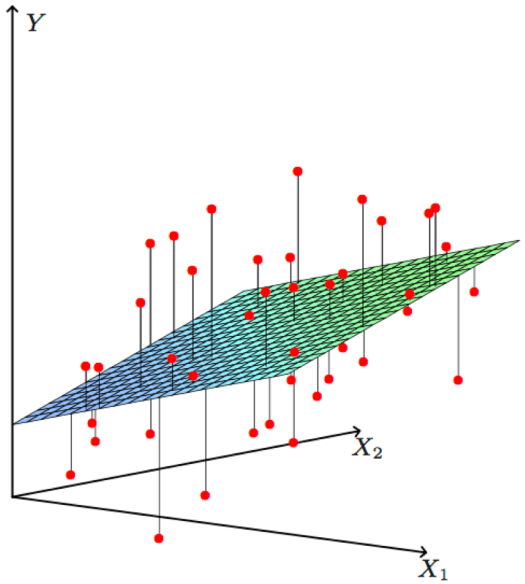


Illustration of a linear least squares fitting with $X \in \mathbb{R}^2$. We seek the linear function of X that minimizes the sum of squared residuals from Y.

Hastie, Tibshirani & Friedman, 2009

Least Squares Regression (LSR)

- In LSR we compute the weight vector **w** by minimizing the mean squared errors on all training examples:

$$\mathbf{w}^* = \operatorname{argmin}_w \frac{1}{N} \sum_{i=1}^{N} \left(\mathbf{x}_i \cdot \mathbf{w} - y_i \right)^2$$

Using a matrix notation where $\mathbf{X} = [x_1\ x_2\ ..\ x_N]^T$ is a matrix containing the training examples vectors as its row we can rewrite the cost function as

$$\mathbf{w}^* = \operatorname{argmin}_w \left(\mathbf{X}\mathbf{w} - \mathbf{y} \right)^T \left(\mathbf{X}\mathbf{w} - \mathbf{y} \right)$$

- To find the optimum **w** we set the derivative of the cost function with respect to **w** to 0, which yields to the following equation:

$$\mathbf{X}^T (\mathbf{X}\mathbf{w} - \mathbf{y}) = 0$$

$$\mathbf{X}^T \mathbf{X}\mathbf{w} = \mathbf{X}^T \mathbf{y}$$

$$\mathbf{w} = \left(\mathbf{X}^T \mathbf{X} \right)^{-1} \mathbf{X}^T \mathbf{y}$$

Regularized Least Squares Regression (LSR)

- When the sample size is limited, i.e. in order to solve ill-posed problems or to prevent over-fitting some form of regularization is often introduced into the model

$$\mathbf{w}^* = \operatorname{argmin}_w \frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{x}_i \cdot \mathbf{w} - y_i\right)^2 + \lambda\|\mathbf{w}\|^2$$

Error term/ Loss function

Regularization term

- The regularization parameter λ controls the amount of regularization.

- Setting the derivative of the cost function with respect to **w** to 0, which yields to the following equation:

$$\mathbf{X}^T(\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda\mathbf{w} = 0$$

$$(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})\mathbf{w} = \mathbf{X}^T\mathbf{y}$$

$$\mathbf{w} = \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y}$$

Statistical Learning – General Framework

- Consider the general equation for making predictions

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

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

- To estimate the weights **w** we seek to minimize the empirical risk which is penalized to restrict model flexibility

$$\mathbf{w}^* = \text{argmin}_w \frac{1}{N} \sum_{i=1}^{N} L(y_i, \mathbf{x}_i, \mathbf{w}) + \lambda J(\mathbf{w})$$

Loss function

Regularization term

Statistical Learning – General Framework

$$\mathbf{w}^* = \operatorname{argmin}_w \frac{1}{N}\sum_{i=1}^{N} L(y_i, \mathbf{x}_i, \mathbf{w}) + \lambda J(\mathbf{w})$$

Loss function

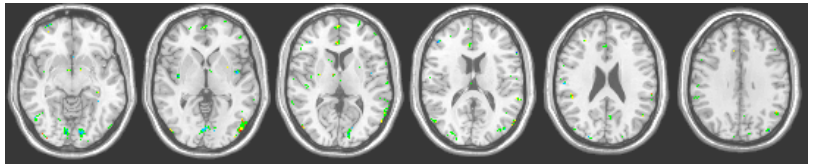
Regularization term

- Loss function: denotes the price we pay when we make mistakes in the predictions (e.g. squared loss, Hinge loss).
- Regularization term: favours certain properties and improves the generalisation over unseen examples (e.g. L2-norm, L1-norm).
- Many learning algorithms are particular choices of *L* and *J* (e.g. SVM, Kernel Ridge Regression) .

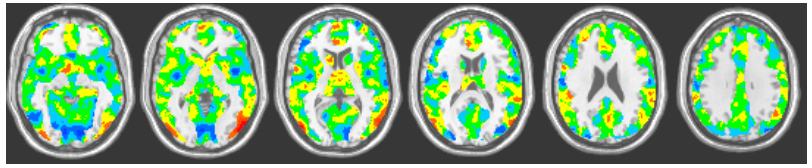
KRR $\longrightarrow$ $$\mathbf{w}^* = \operatorname{argmin}_w \frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{x}_i \cdot \mathbf{w} - y_i\right)^2 + \lambda \|\mathbf{w}\|^2$$

SVM $\longrightarrow$ $$\mathbf{w}^* = \operatorname{argmin}_w C\frac{1}{N}\sum_{i=1}^{N}\max\left[1 - y_i\left(\mathbf{x}_i \cdot \mathbf{w} + b\right), 0\right] + \lambda \|\mathbf{w}\|^2$$

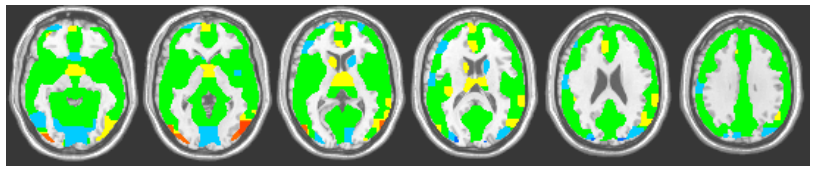
Impact of regularization on **w**



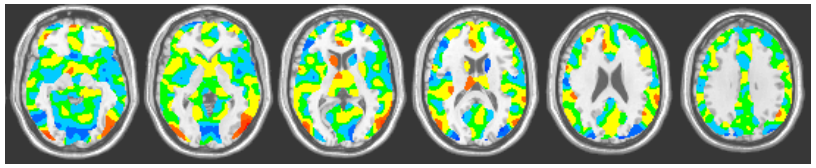
LASSO
86.31%



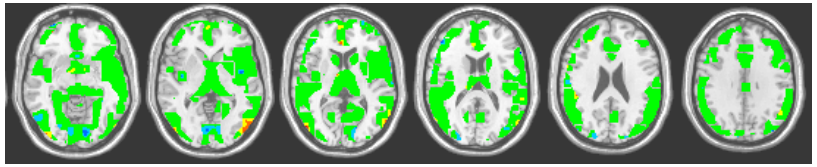
Elastic Net
88.02%



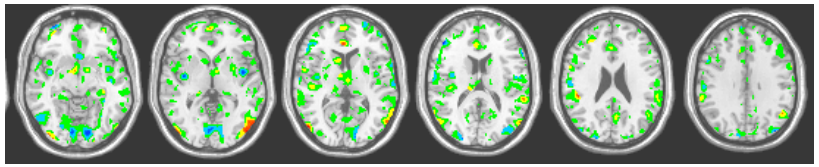
Total Variation (TV)
85.79%



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 + regularization.**

Baldassarre L, Pontil M, Mourao-Miranda J (2017)

References

•Shawe-Taylor J, Christianini N (2004) Kernel Methods for Pattern Analysis. Cambridge: Cambridge University Press.

•Schölkopf, B., Smola, A., 2002. Learning with Kernels. MIT Press.

•Burges, C., 1998. A tutorial on support vector machines for pattern recognition. Data Min. Knowl. Discov. 2 (2), 121–167.

•Chu C, Ni Y, Tan G, Saunders CJ & Ashburner J. (2011): Kernel Regression for fMRI pattern prediction. NeuroImage.

•Rakotomamonjy, Alain, Francis R. Bach, Stéphane Canu, et Yves Grandvalet. SimpleMKL. Journal of Machine Learning (2008): 2491-2521.

•Hastie T, Tibshirani R & Friedman J. The Elements of Statistical Learning 2009. Springer Series in Statistics.