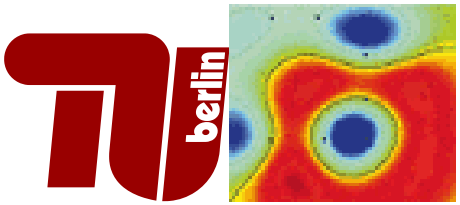
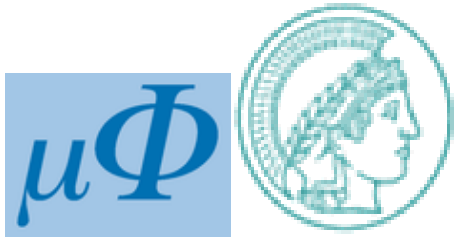




Introduction to Machine Learning

Felix Brockherde[12] Kristof Schütt[1]

[1]Technische Universität Berlin
[2]Max Planck Institute of Microstructure Physics

IPAM Tutorial 2013

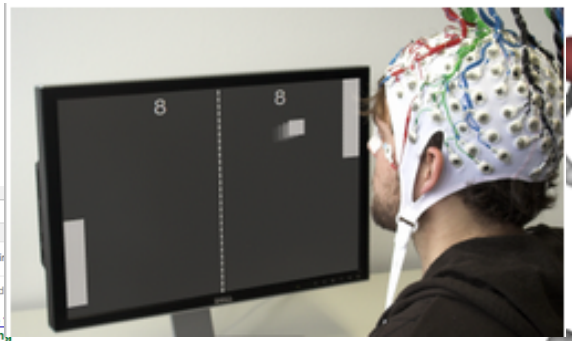
What is Machine Learning?

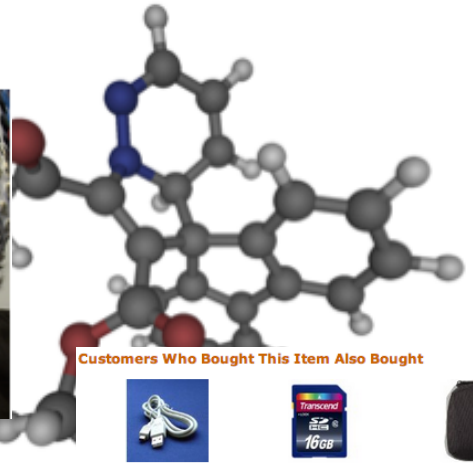
Data with Pattern | Algorithm | ML Model

ML is about learning structure from data

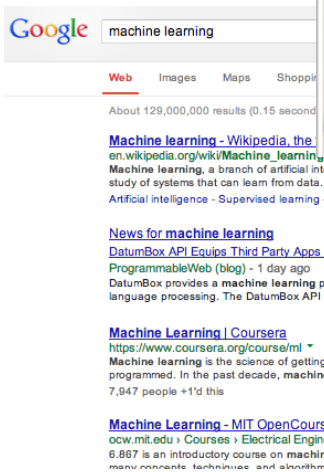
Examples

Drug discovery



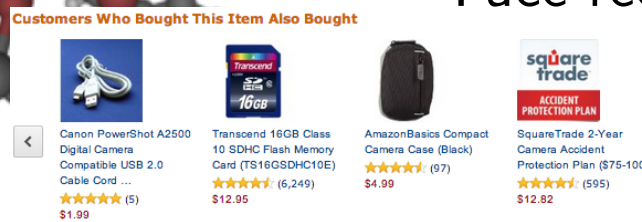


Face recognition

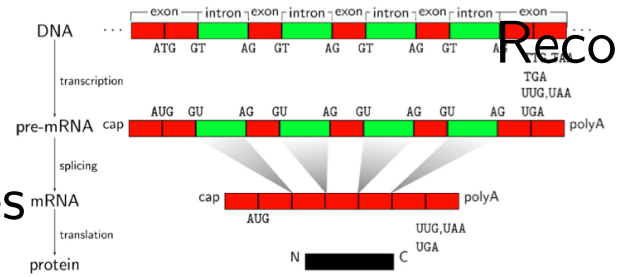


Search engines

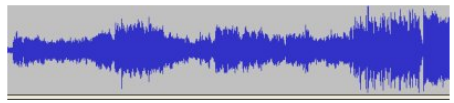
BCI



Recommender systems



DNA splice site detection



Speech recognition

This Talk

Part 1: Learning Theory and Supervised ML

- Basic Ideas of Learning Theory
- Support Vector Machines
- Kernels
- Kernel Ridge Regression

Part 2: Unsupervised ML and Application

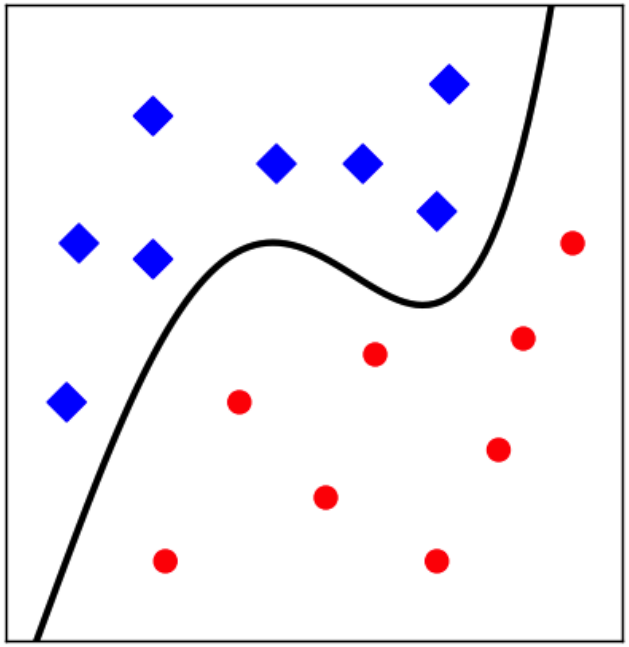
- PCA
- Model Selection
- Feature Representation

Not covered

- Probabilistic Models
- Neural Networks
- Online Learning
- Reinforcement Learning
- Semi-supervised Learning
- etc.

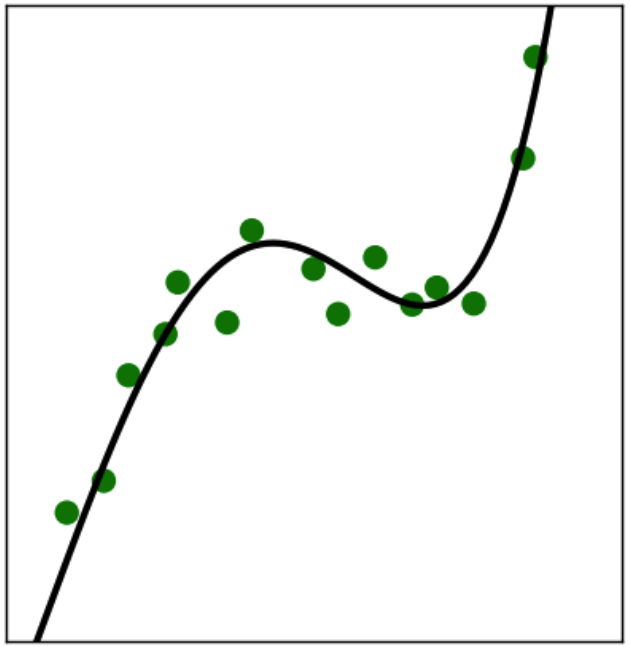
Supervised Learning

Classification



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

Regression



$y_i \in \mathbb{R}$

- Given: *Points* $X = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ with $\mathbf{x}_i \in \mathbb{R}^d$ and *Labels* $Y = (y_1, \ldots, y_n)$ generated by some joint probability distribution.
- Learn underlying unknown mapping $f(\mathbf{x}) = y$
- Important: Performance on unseen data

Basic Ideas in Learning Theory

Risk minimization (RM)

Learn a model function f from examples

$$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N) \in \mathbb{R}^d \times \mathbb{R} \text{ or } \{+1, -1\}, \text{ generated from } P(\mathbf{x}, y)$$

such that the expected number of errors on test data (drawn fom $P(\mathbf{x}, y)$),

$$R[f] = \int \frac{1}{2} |f(\mathbf{x}) - y|^2 \mathrm{d}P(\mathbf{x}, y),$$

is minimal.

Problem: Distribution $P(\mathbf{x}, y)$ is unknown

Empirical Risk Minimization (ERM)

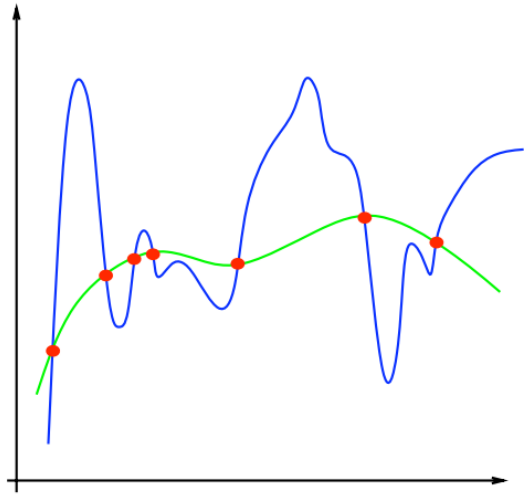
Replace the average over P(x, y) by average of training samples (i.e. *minimize the training error*):

$$R_{\text{emp}}[f] = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} |f(\mathbf{x}_i) - y_i|^2$$

Law of large numbers: $R_{\text{emp}}[f] \to R[f]$ as $N \to \infty$.

Question: Does $\min_f R_{\text{emp}}[f]$ give us $\min_f R[f]$ for sufficiently large N?

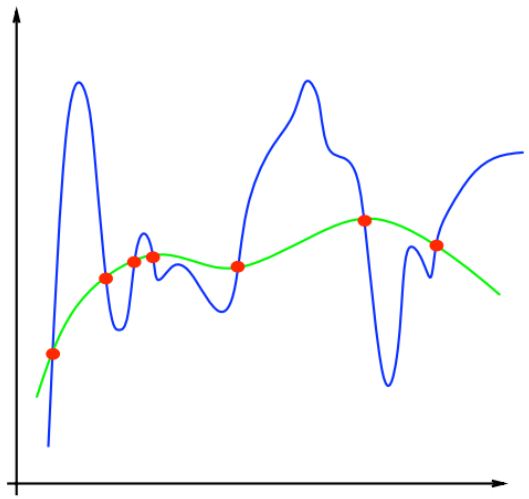
No: uniform convergence needed



Law of large numbers: $R_{\text{emp}}[f] \to R[f]$ as $N \to \infty$.

Question: Does $\min_f R_{\text{emp}}[f]$ give us $\min_f R[f]$ for sufficiently large N?

No: uniform convergence needed



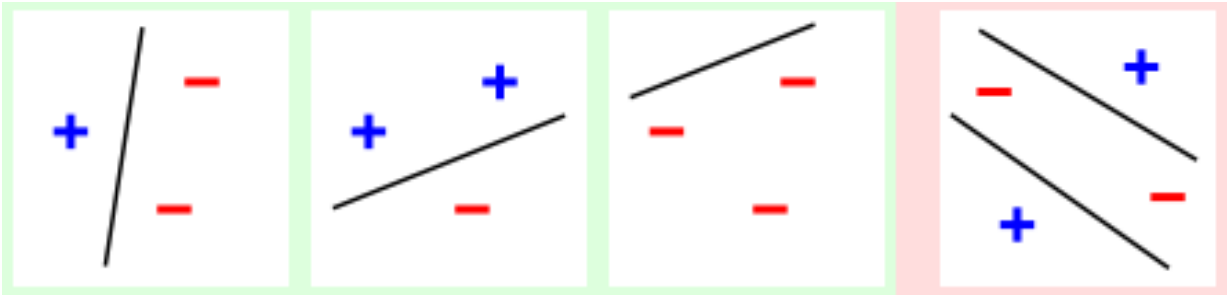
Error bound for classification

With probablity of at least $1 - \eta$:

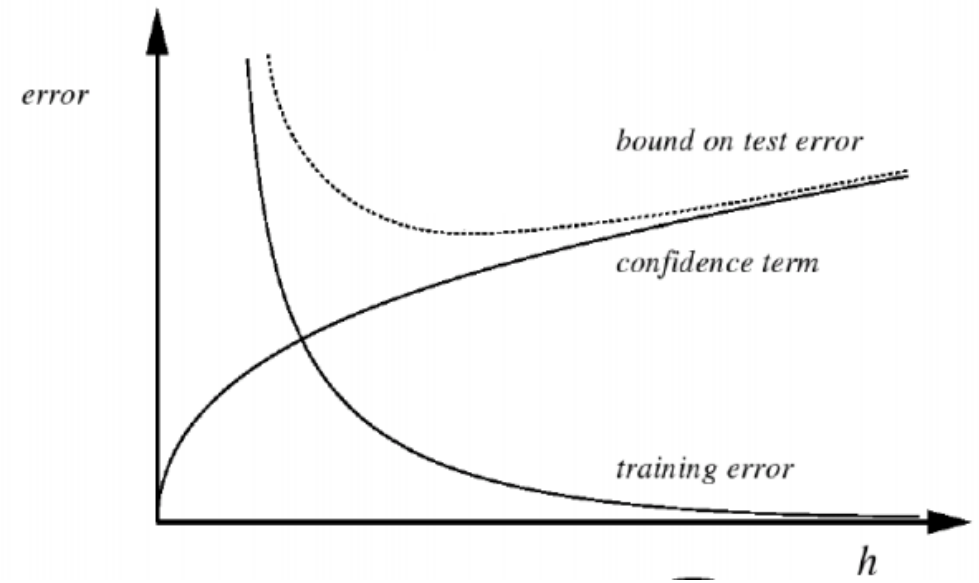
$$R[f] \leq R_{\text{emp}}[f] + \sqrt{\frac{D(\log \frac{2N}{D} + 1) - \log(\frac{\eta}{4})}{N}}$$

where D is the VC dimension (Vapnik and Chervonenkis (1971)).

Introduce structure on set of possible functions and use Structural Risk Minimization (SRM).

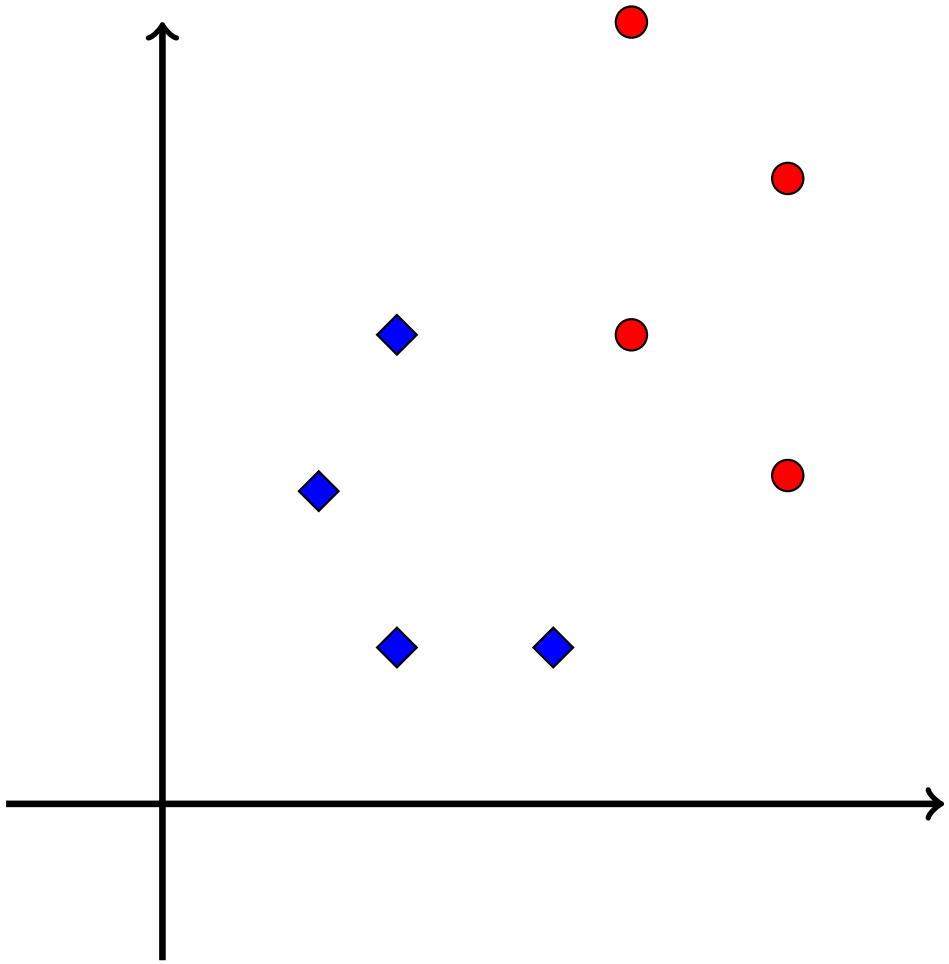


The linear function class has VC-dimension $D = 3$

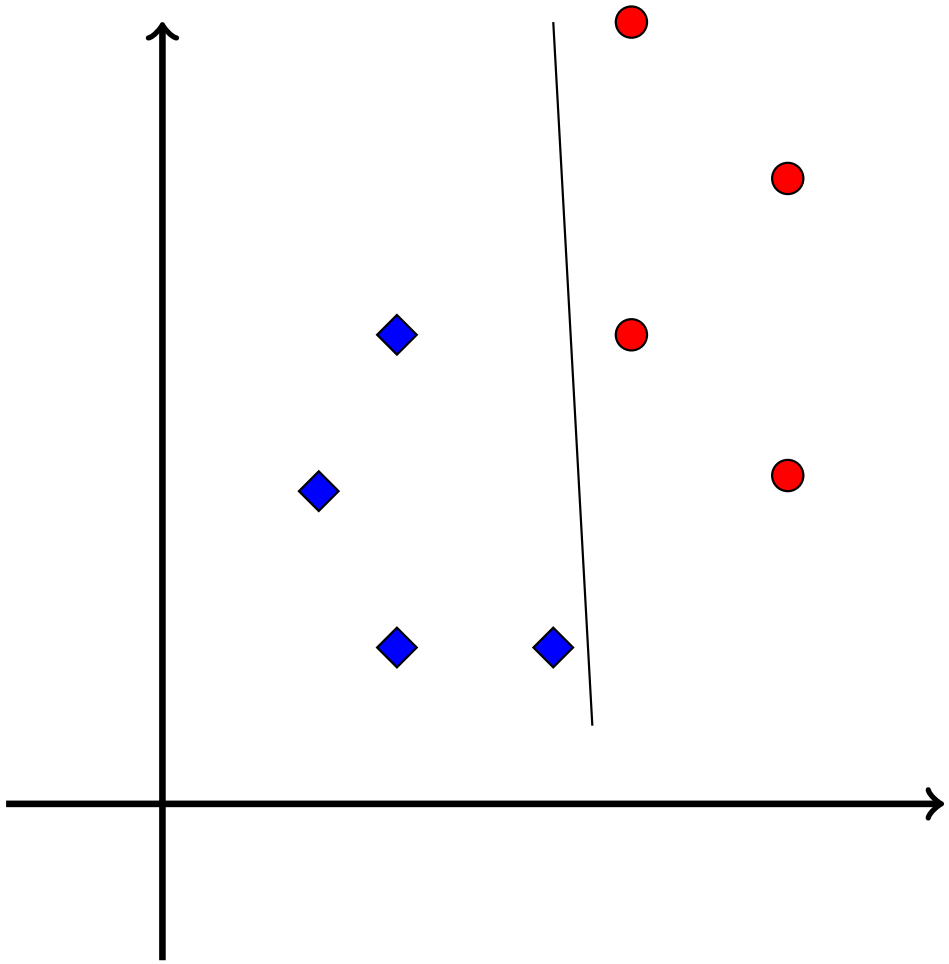


$$\min_f \quad R_{\text{emp}}[f] + \text{Complexity}[f]$$

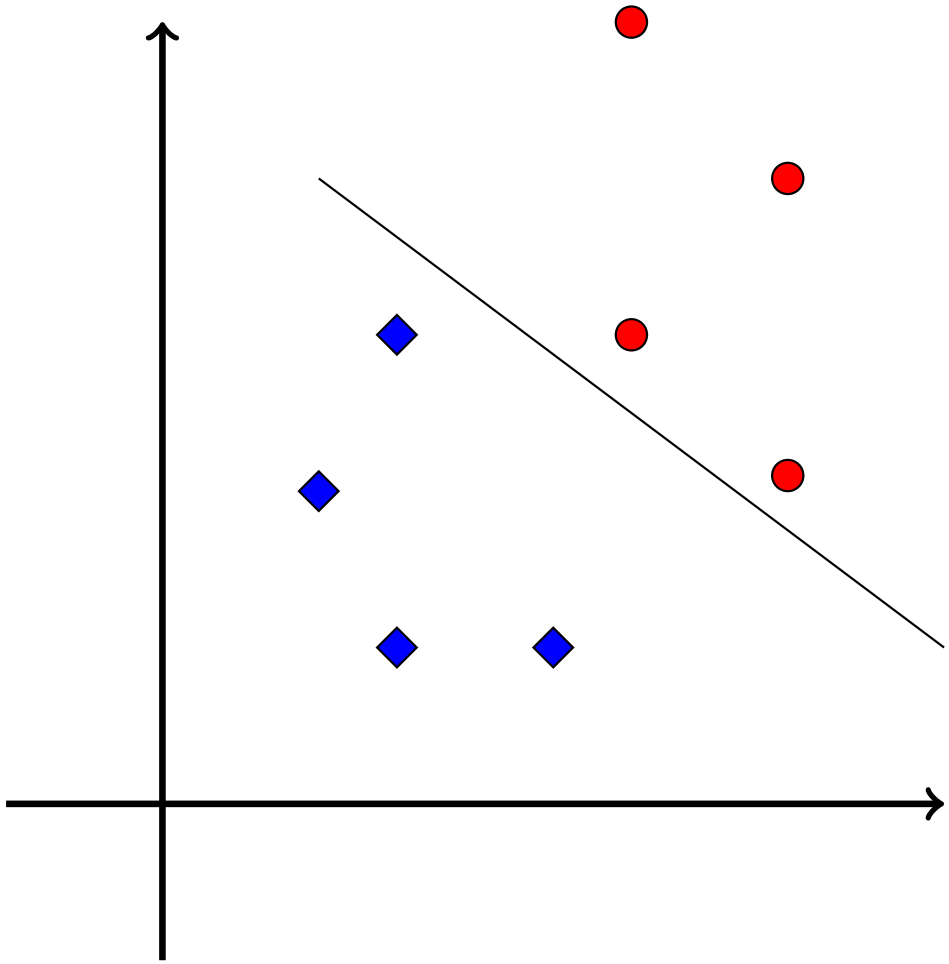
Support Vector Machines (SVM)



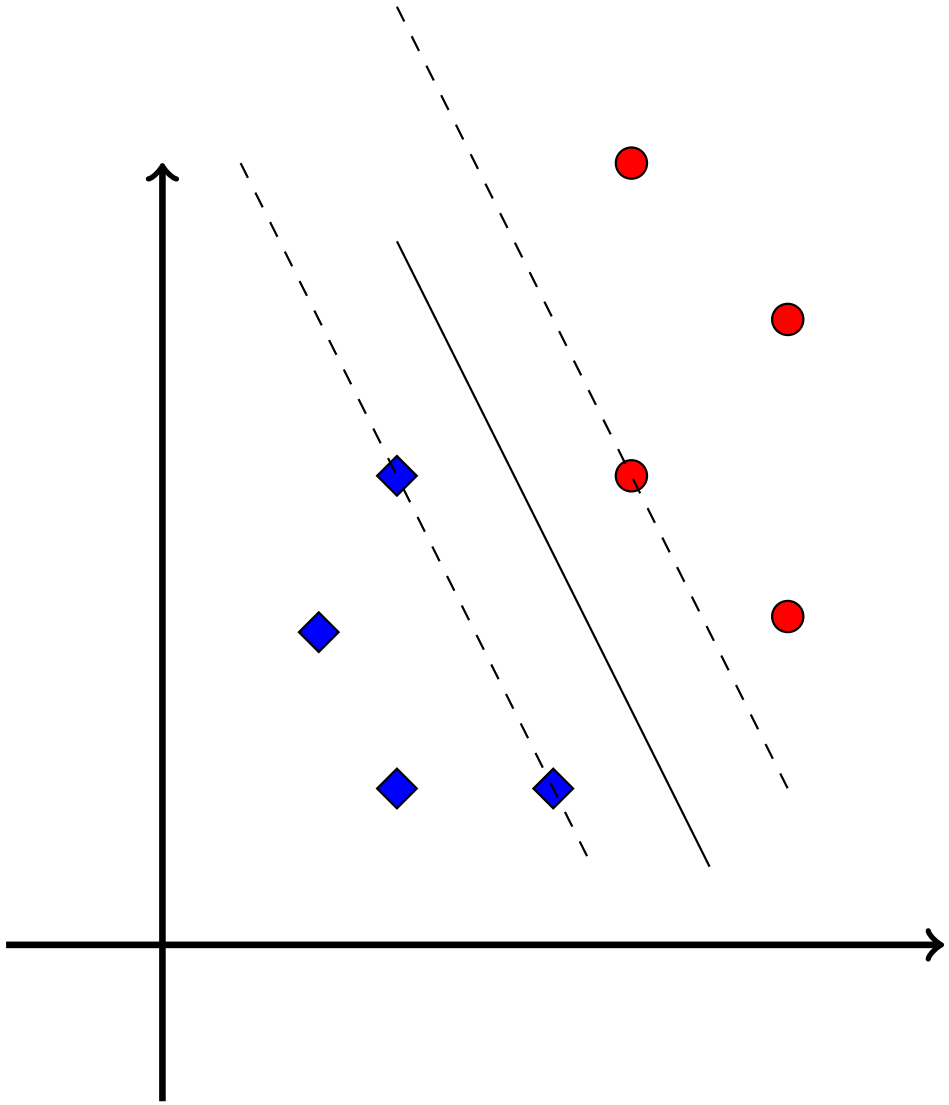
Support Vector Machines (SVM)



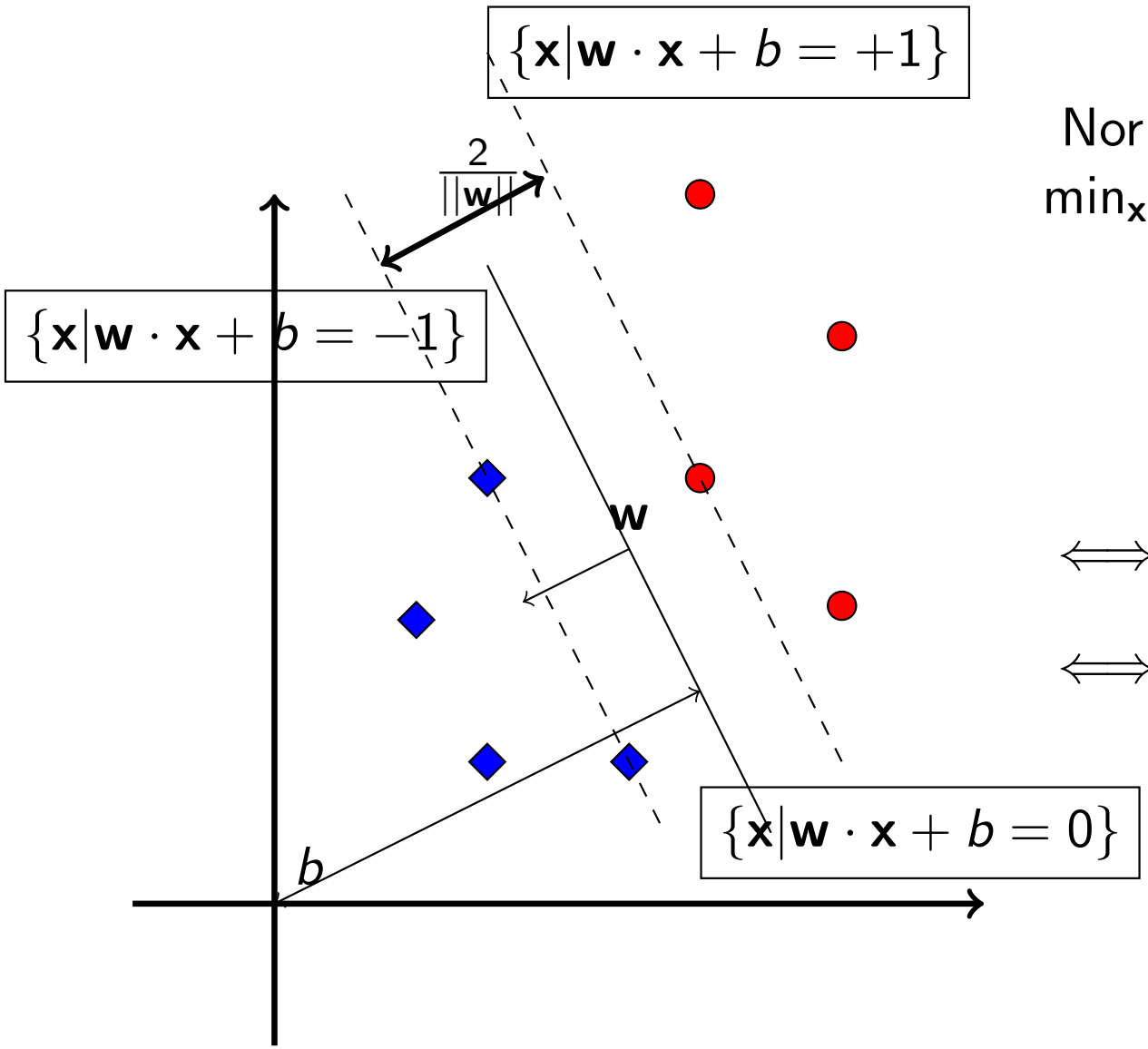
Support Vector Machines (SVM)



Support Vector Machines (SVM)



Support Vector Machines (SVM)



Normalize $\mathbf{w}$ so that $\min_{\mathbf{x}_i} \mathbf{w}\cdot\mathbf{x}_i + b = 1$.

$$\begin{aligned}
&\mathbf{w}\cdot\mathbf{x}_1 + b = +1 \\
&\mathbf{w}\cdot\mathbf{x}_2 + b = -1 \\
\iff &\mathbf{w}\cdot(\mathbf{x}_1 - \mathbf{x}_2) = 2 \\
\iff &\frac{\mathbf{w}}{||\mathbf{w}||}\cdot(\mathbf{x}_1 - \mathbf{x}_2) = \frac{2}{||\mathbf{w}||}
\end{aligned}$$

VC Dimension of Hyperplane Classifiers

Theorem (Cortes and Vapnik (1995))

Hyperplanes in canonical form have VC Dimension

$$D \leq \min\{R^2||\mathbf{w}||^2 + 1,\ N + 1\}$$

where R the radius of the smallest sphere containing the data.

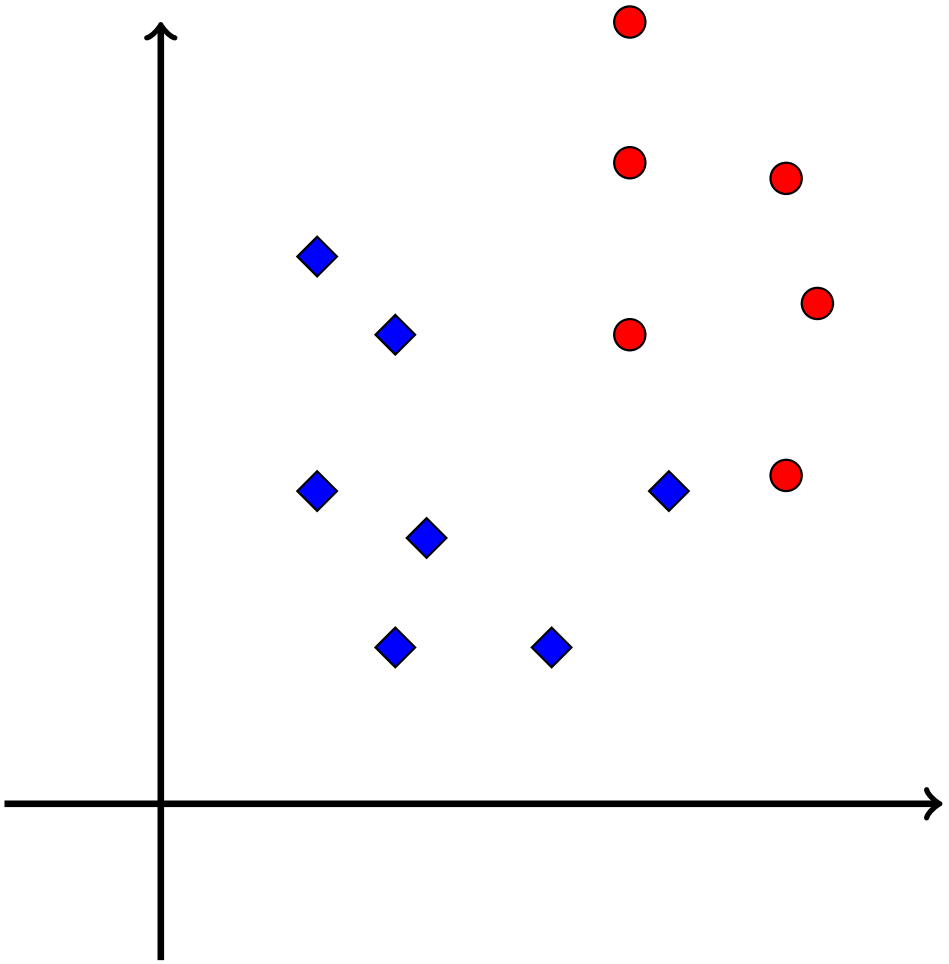
SRM Bound:

$$R[f] \leq R_{\text{emp}}[f] + \sqrt{\frac{D(\log \frac{2N}{D} + 1) - \log(\frac{\eta}{4})}{N}}$$

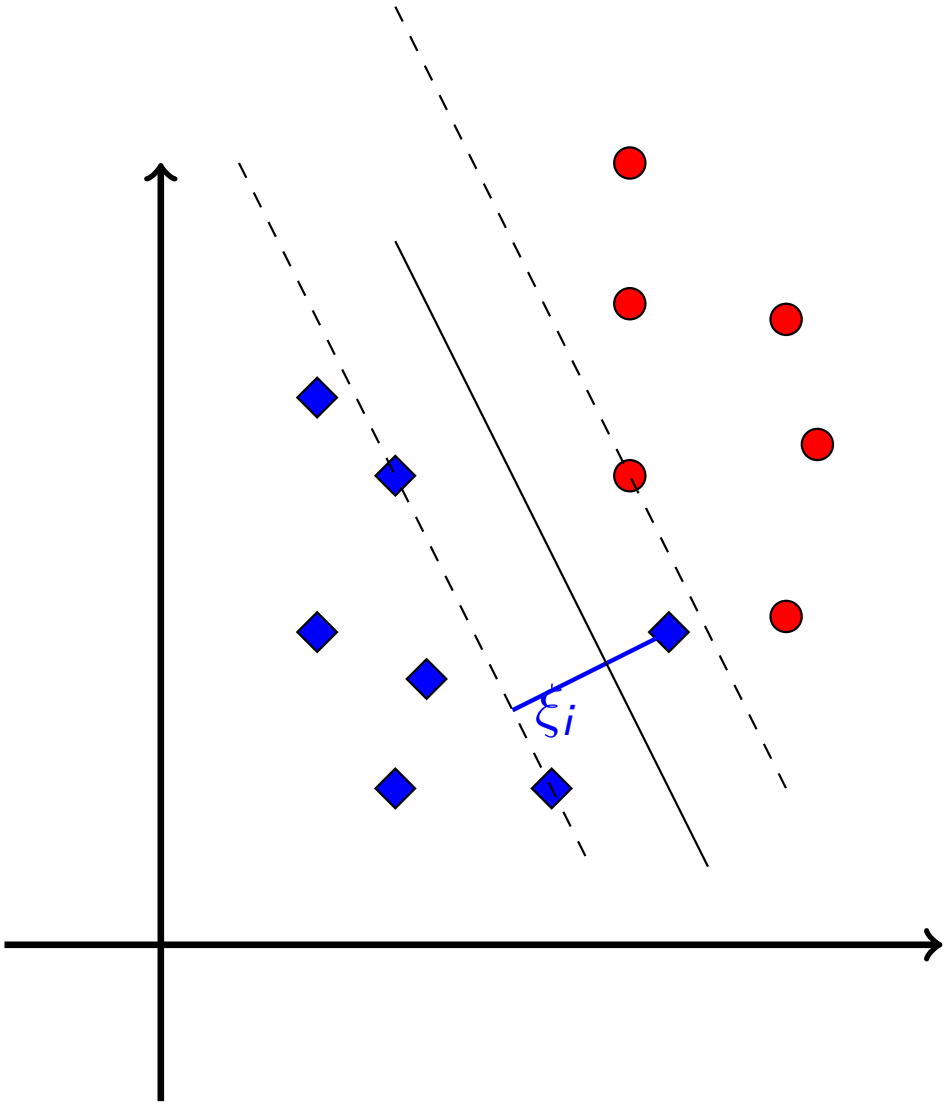
maximal margin = minimum $||\mathbf{w}||^2 \rightarrow$ good generalization, i.e. low risk:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & ||\mathbf{w}||^2 \\ \text{subject to} \quad & y_i(\mathbf{w} \cdot \mathbf{x_i} + b) \geq 1 \quad \text{for } i = 1 \ldots N \end{aligned}$$

Slack variables



Slack variables



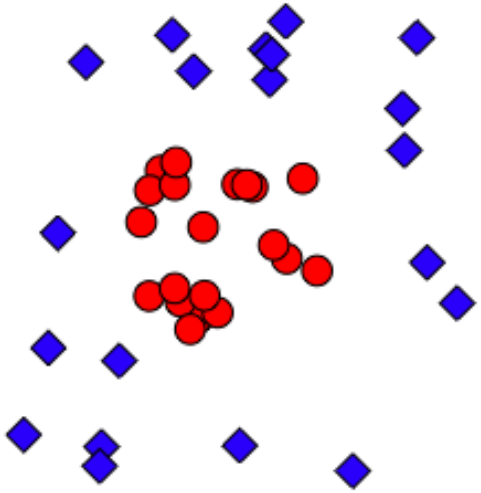
Introduce slack variables ξ_i:

$$\min_{\mathbf{w},b,\xi_i} \quad ||\mathbf{w}||^2 + C\sum_{i=1}^{N}\xi_i$$

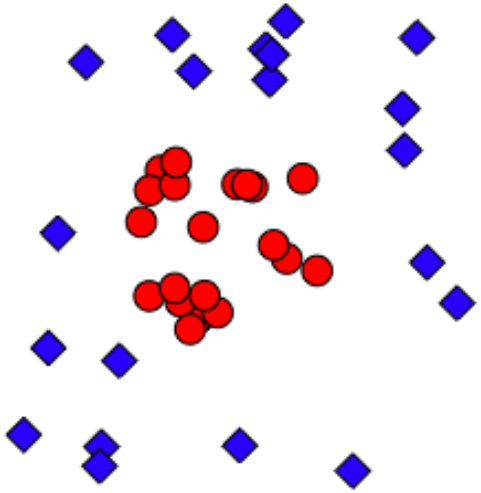
$$\text{subject to} \quad y_i(\mathbf{w}\cdot\mathbf{x_i}+b) \geq 1-\xi_i$$

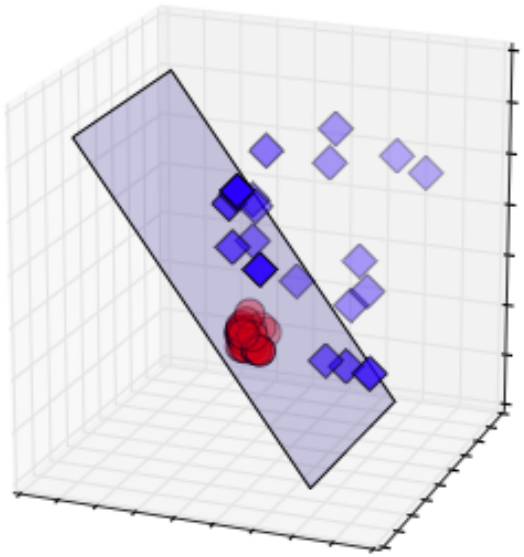
$$\xi_i \geq 0$$

Non-linear hyperplanes



Non-linear hyperplanes





Map into a higher dimensional feature space:

$$\Phi: \quad \mathbb{R}^2 \quad \rightarrow \quad \mathbb{R}^3$$

$$(x_1, x_2) \quad \mapsto \quad (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

Dual SVM

Primal

$$\min_{\mathbf{w},b,\xi_i} \quad ||\mathbf{w}||^2 + C\sum_{i=1}^{N}\xi_i$$

$$\text{subject to} \quad y_i(\mathbf{w}\cdot\Phi(\mathbf{x_i}) + b) \geq 1 - \xi_i \;\text{ and }\; \xi_i \geq 0 \quad \text{for } i = 1\ldots N$$

Dual

$$\max_{\alpha} \quad \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j y_i y_j \left(\Phi(\mathbf{x_i})\cdot\Phi(\mathbf{x_j})\right)$$

$$\text{subject to} \quad \sum_{i=1}^{N}\alpha_i y_i = 0 \;\text{ and }\; C \geq \alpha_i \geq 0 \quad \text{for } i = 1\ldots N$$

Data points x_i only appear in scalar products $(\Phi(\mathbf{x_i})\cdot\Phi(\mathbf{x_j}))$.

The Kernel Trick

Replace scalar products with kernel function (Müller et al. (2001)):

$$k(x, y) = \Phi(x) \cdot \Phi(y)$$

- Compute kernel matrix $K_{ij} = k(x_i, x_j)$, i.e. never use Φ directly
- Underlying mapping Φ can be unknown
- Kernels can be adopted to specific task, e.g. using prior knowledge (kernels for graphs, trees, strings, ...)

Common kernels

Gaussian Kernel:	$k(x, y)$	$=$	$\exp\left(-\frac{\|\|x-y\|\|^2}{2\sigma^2}\right)$
Linear Kernel:	$k(x, y)$	$=$	$x \cdot y$
Polynomial Kernel:	$k(x, y)$	$=$	$(x \cdot y + c)^d$

The Support Vectors in SVM

$$\max_{\alpha} \quad \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \left(\Phi(\mathbf{x_i}) \cdot \Phi(\mathbf{x_j})\right)$$

$$\text{subject to} \quad \sum_{i=1}^{N} \alpha_i y_i = 0 \;\text{ and }\; C \geq \alpha_i \geq 0 \quad \text{for } i = 1 \ldots N$$

KKT conditions

$$y_i \left[\mathbf{w}\Phi(\mathbf{x_i})) + b\right] > 1 \Longrightarrow \qquad a_i = 0 \longrightarrow \qquad x_i \text{ irrelevant}$$

$$y_i \left[\mathbf{w}\Phi(\mathbf{x_i})) + b\right] = 1 \Longrightarrow \quad \text{on/in margin} \longrightarrow \quad x_i \text{ Support Vector}$$

Old model $f(x) = w \cdot \Phi(x_i) + b$ becomes via $w = \sum_{i=1}^{N} \alpha_i y_i \Phi(x_i)$:

$$f(x) = \sum_{i=1}^{N} \alpha_i y_i k(x_i, x) + b \quad \longrightarrow \quad f(x) = \sum_{x_i \in SV} \alpha_i y_i k(x_i, x) + b$$

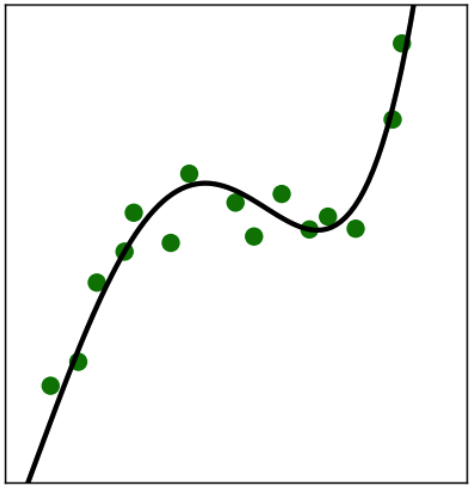
Kernel Ridge Regression (KRR)

Ridge Regression

$$\min_w \sum_{i=1}^{N} ||y_i - w \cdot \mathbf{x_i}||^2 + \lambda ||w||^2$$

Setting derivative to zero gives

$$w = \left(\lambda I + \sum_{i=1}^{N} x_i x_i^T \right)^{-1} \sum_{i=1}^{N} y_i x_i$$



Linear Model: $f(x) = w \cdot x$

Kernelizing Ridge Regression

Setting $X = (x_1, \ldots, x_N) \in \mathbb{R}^{d \times N}$ and $Y = (y_1, \ldots, y_n)^\intercal \in \mathbb{R}^N$:

$$w = (\lambda I + XX^\intercal)^{-1} XY$$

Apply Woodbury Matrix identity:

$$w = X (X^\intercal X + \lambda I)^{-1} Y$$

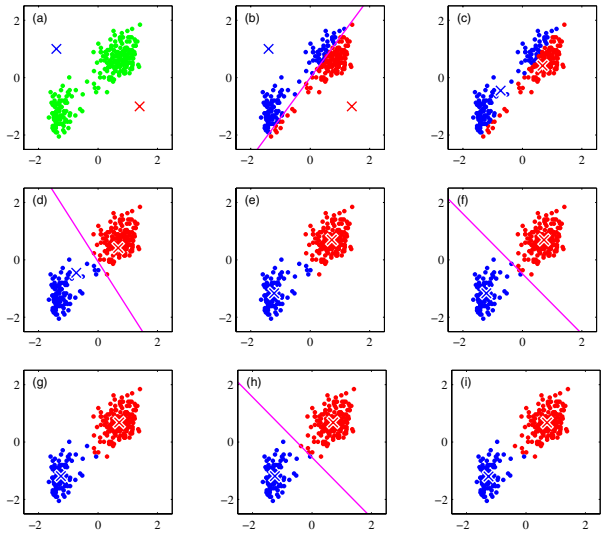
Introduce α:

$$\alpha = (K + \lambda I)^{-1} Y \quad \text{and} \quad w = \sum_{i=1}^{N} \Phi(\mathbf{x_i}) \alpha_i$$

Kernel Model: $f(x) = w \cdot \Phi(x) = \sum_{i=1}^{N} \alpha_i k(\mathbf{x}_i, \mathbf{x})$

Unsupervised Learning

- Learn structure from unlabeled data
- Fit an assumed model / distribution to the data
- Examples
  - clustering
  - blind source separation
  - outlier detection
  - dimensionality reduction



Principal Component Analysis (PCA)

Given centered data matrix $X = (x_1, \ldots, x_N)^\intercal \in \mathbb{R}^{N \times D}$

- best linear approximation

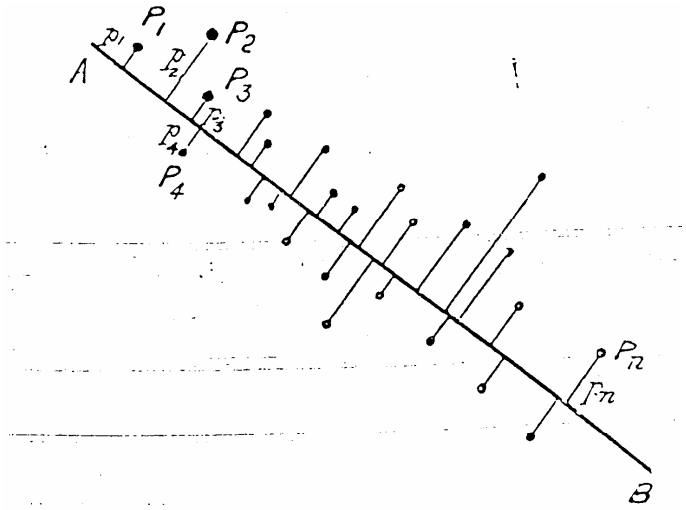
$$w_1 = \arg \min_{\|w\|=1} \left\{ \|X - X w w^\intercal\|^2 \right\}$$

- direction of largest variance

$$w_1 = \arg \max_{\|w\|=1} \left\{ \|X w\|^2 \right\}$$

- matrix reduction for further components

$$X_{k+1} = X_k - X_k w w^\intercal$$



Pearson (1901)

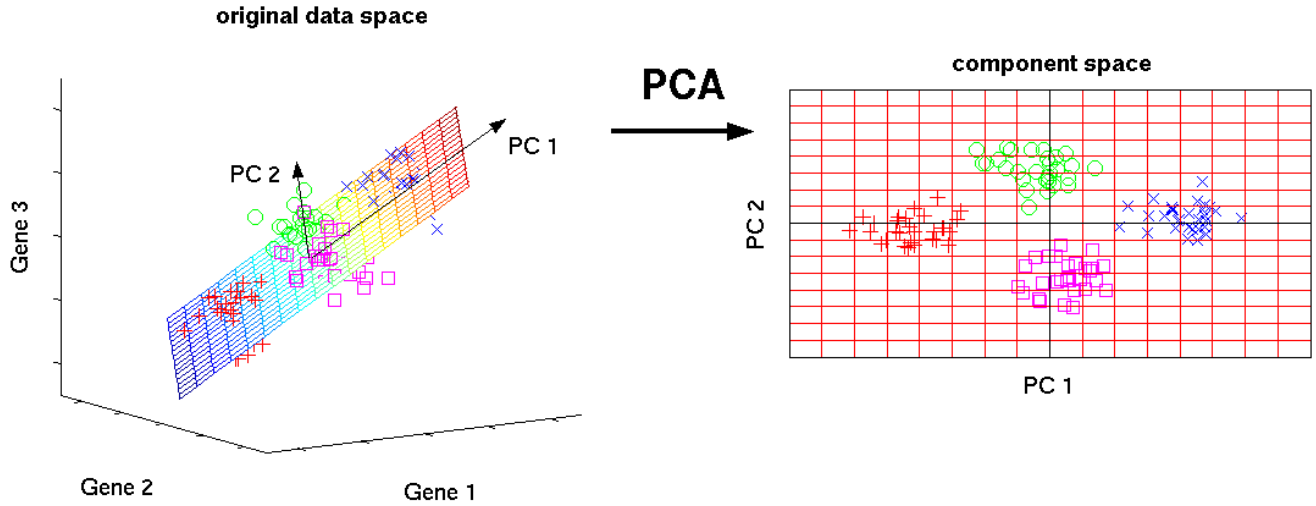
Principal Component Analysis (PCA)

Given centered data matrix $X \in \mathbb{R}^{N \times D}$, decompose correlated data matrix into uncorrelated, orthogonal *PCs*

- diagonalize covariance matrix $\Sigma = \frac{1}{N} X^\intercal X$

$$\Sigma \mathbf{w}_k = \sigma_k^2 \mathbf{w}_k$$

- order principal components $\mathbf{w}_k$ by variance σ_k^2
- project data to first n principal components



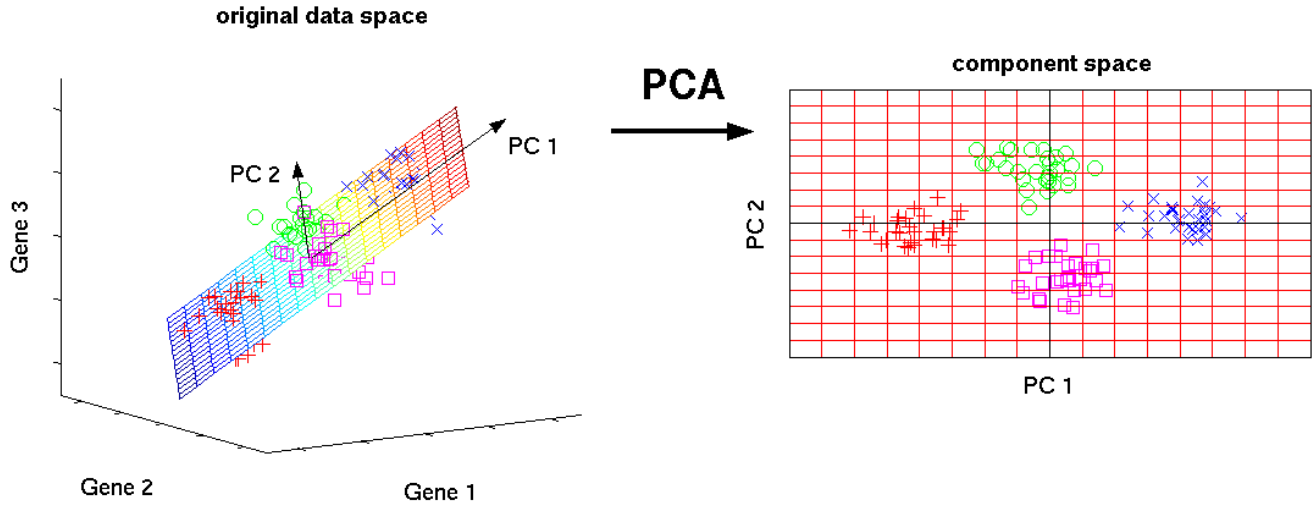
Principal Component Analysis (PCA)

Given centered data matrix $X \in \mathbb{R}^{N \times D}$, decompose correlated data matrix into uncorrelated, orthogonal *PCs*

- diagonalize covariance matrix $\Sigma = \frac{1}{N} X^\intercal X$

$$\Sigma \mathbf{w}_k = \sigma_k^2 \mathbf{w}_k$$

- order principal components $\mathbf{w}_k$ by variance σ_k^2
- project data to first n principal components



What about nonlinear correlations?

Kernel Principal Component Analysis (kPCA)

Transformation to feature space $X \mapsto X_f$:

$$\Sigma_f = \frac{1}{N} X_f^\intercal X_f,\ K = X_f X_f^\intercal, K_{ij} = k(x_i, x_j)$$

$$\Sigma_f \mathbf{w}_k = \sigma_k^2 \mathbf{w}_k$$

Kernel Principal Component Analysis (kPCA)

Transformation to feature space $X \mapsto X_f$:

$$\Sigma_f = \frac{1}{N} X_f^\intercal X_f,\ K = X_f X_f^\intercal, K_{ij} = k(x_i, x_j)$$

$$X_f^\intercal X_f \mathbf{w}_k = N\sigma_k^2 \mathbf{w}_k$$

Kernel Principal Component Analysis (kPCA)

Transformation to feature space $X \mapsto X_f$:

$$\Sigma_f = \frac{1}{N} X_f^\intercal X_f,\ K = X_f X_f^\intercal, K_{ij} = k(x_i, x_j)$$

$$\begin{aligned} X_f^\intercal X_f \mathbf{w}_k &= N\sigma_k^2 \mathbf{w}_k \\ \Downarrow \mathbf{w}_k &= X_f^\intercal \boldsymbol{\alpha}_k \\ X_f^\intercal X_f X_f^\intercal \boldsymbol{\alpha}_k &= N\sigma_k^2 X_f^\intercal \boldsymbol{\alpha}_k \end{aligned}$$

Kernel Principal Component Analysis (kPCA)

Transformation to feature space $X \mapsto X_f$:

$$\Sigma_f = \frac{1}{N} X_f^\intercal X_f,\ K = X_f X_f^\intercal, K_{ij} = k(x_i, x_j)$$

$$\begin{aligned} X_f^\intercal X_f \mathbf{w}_k &= N\sigma_k^2 \mathbf{w}_k \\ \Downarrow \mathbf{w}_k &= X_f^\intercal \boldsymbol{\alpha}_k \\ X_f^\intercal X_f X_f^\intercal \boldsymbol{\alpha}_k &= N\sigma_k^2 X_f^\intercal \boldsymbol{\alpha}_k \\ \Downarrow X_f &\cdot \\ X_f X_f^\intercal X_f X_f^\intercal \boldsymbol{\alpha}_k &= N\sigma_k^2 X_f X_f^\intercal \boldsymbol{\alpha}_k \end{aligned}$$

Kernel Principal Component Analysis (kPCA)

Transformation to feature space $X \mapsto X_f$:

$$\Sigma_f = \frac{1}{N} X_f^\intercal X_f,\ K = X_f X_f^\intercal, K_{ij} = k(x_i, x_j)$$

$$\begin{aligned} X_f^\intercal X_f \mathbf{w}_k &= N\sigma_k^2 \mathbf{w}_k \\ \Downarrow\ \mathbf{w}_k &= X_f^\intercal \boldsymbol{\alpha}_k \\ X_f^\intercal X_f X_f^\intercal \boldsymbol{\alpha}_k &= N\sigma_k^2 X_f^\intercal \boldsymbol{\alpha}_k \\ \Downarrow\ X_f\cdot & \\ K^2 \boldsymbol{\alpha}_k &= N\sigma_k^2 K \boldsymbol{\alpha}_k \end{aligned}$$

Kernel Principal Component Analysis (kPCA)

Transformation to feature space $X \mapsto X_f$:

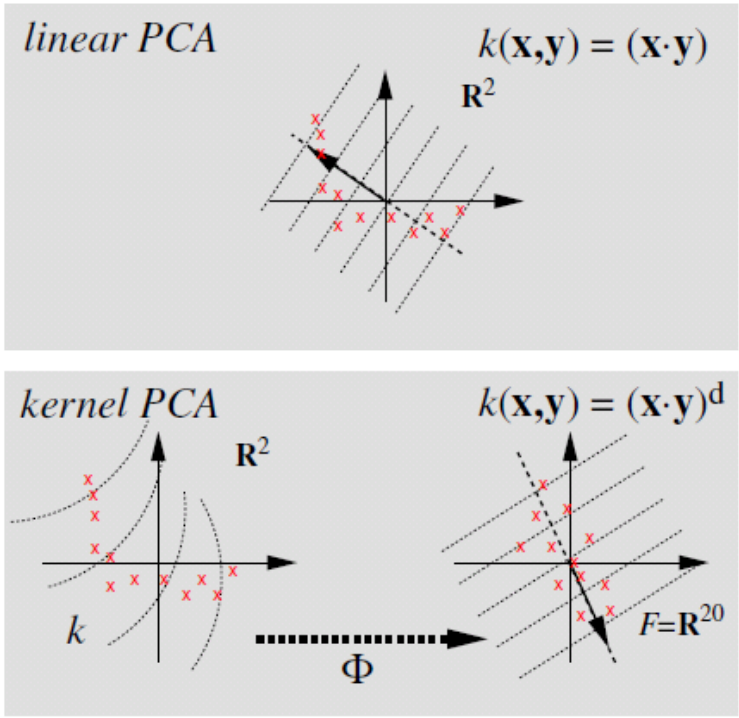
$$\Sigma_f = \frac{1}{N} X_f^\intercal X_f,\ K = X_f X_f^\intercal, K_{ij} = k(x_i, x_j)$$

$$\begin{aligned}
X_f^\intercal X_f \mathbf{w}_k &= N\sigma_k^2 \mathbf{w}_k \\
&\Downarrow \mathbf{w}_k = X_f^\intercal \boldsymbol{\alpha}_k \\
X_f^\intercal X_f X_f^\intercal \boldsymbol{\alpha}_k &= N\sigma_k^2 X_f^\intercal \boldsymbol{\alpha}_k \\
&\Downarrow X_f \cdot \\
K^2 \boldsymbol{\alpha}_k &= N\sigma_k^2 K \boldsymbol{\alpha}_k \\
&\Downarrow K^{-1} \cdot \\
K \boldsymbol{\alpha}_k &= N\sigma_k^2 \boldsymbol{\alpha}_k
\end{aligned}$$

Kernel Principal Component Analysis (kPCA)

Projection:

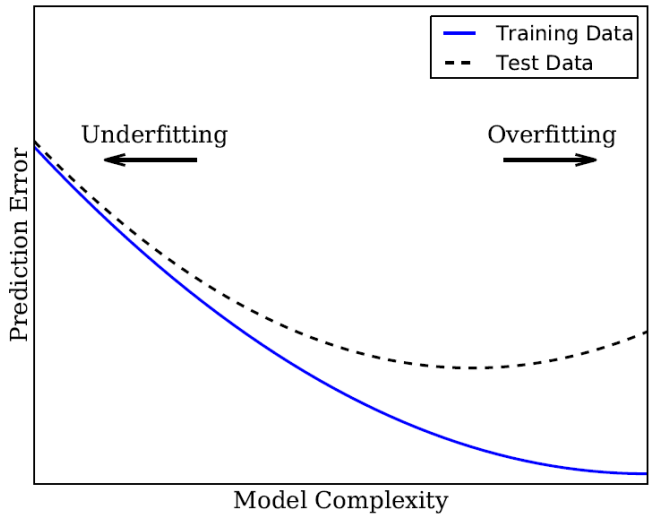
$$x_f^\mathsf{T} w_k = x_f^\mathsf{T} X_f^\mathsf{T} \boldsymbol{\alpha}_k$$
$$= \sum_{i=1}^{N} \alpha_{k,i} k(x, x_i)$$



Schölkopf et al. (1997)

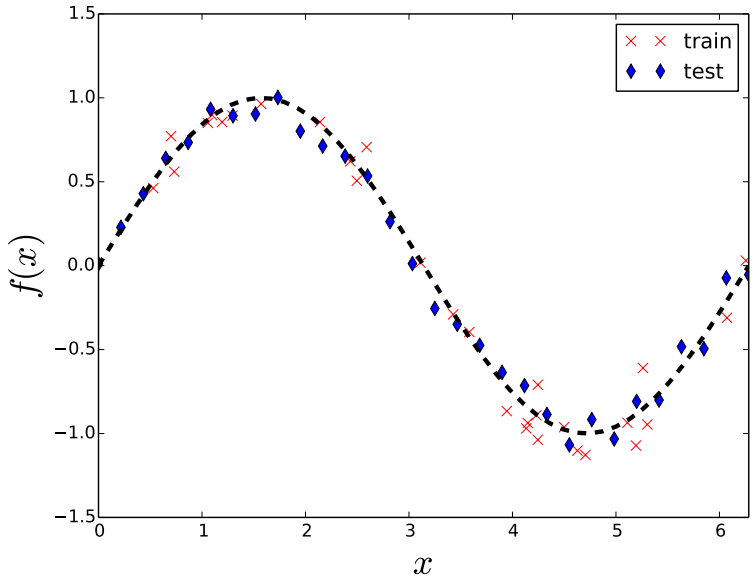
Model Selection

- Find the model that best fits the data distribution
- We can only estimate this distribution
- Consider
  - noise ratio / distribution
  - data correlation



Hyperparameters

- adjust model complexity
  - regularization, kernel parameters, etc.
- have to be tuned using examples **not** used for training
- standard solution: exhaustive search over parameter grid

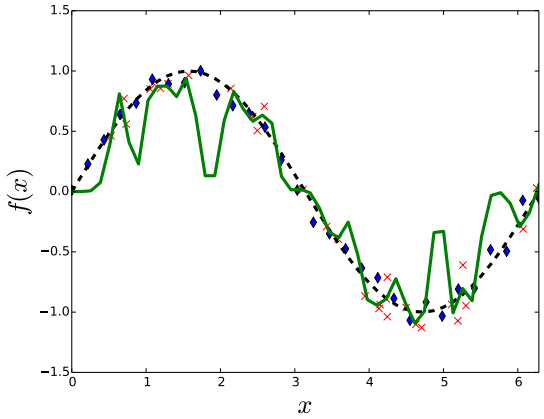


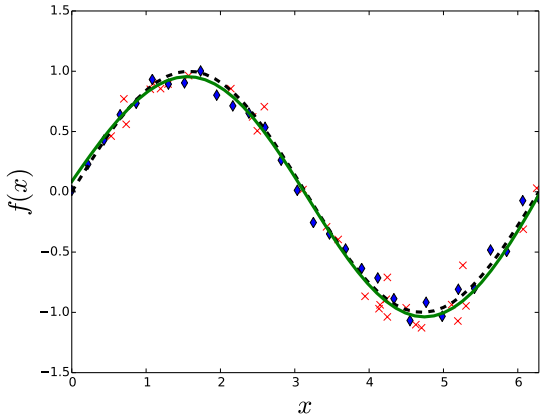
$$f(x) = \sin(x)$$

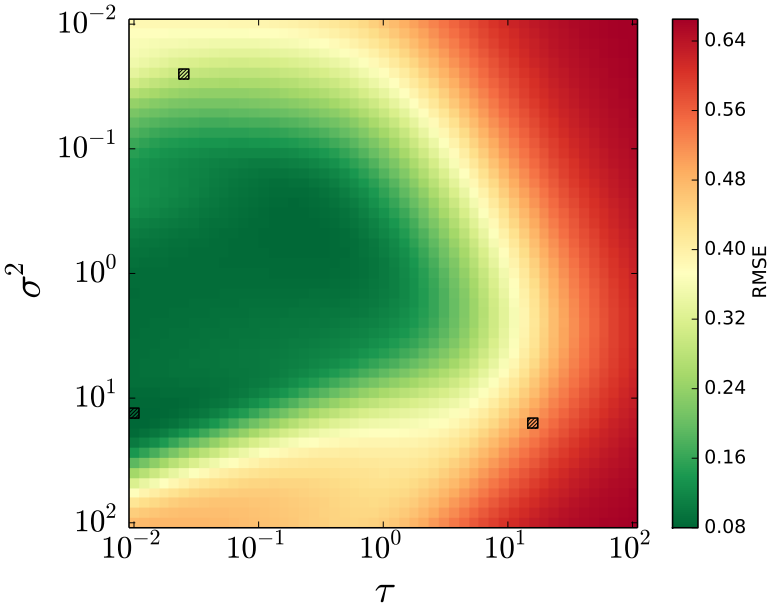
$$\bar{f}(x) = \sum_i \alpha_i \exp\left(\frac{\|x - x_i\|^2}{\sigma^2}\right)$$

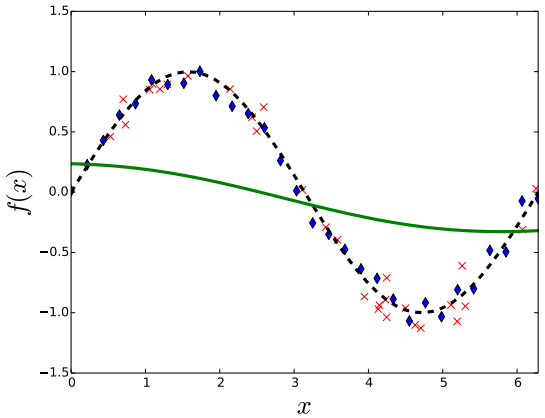
$$\boldsymbol{\alpha} = (K + \tau I)^{-1} y$$

Grid Search

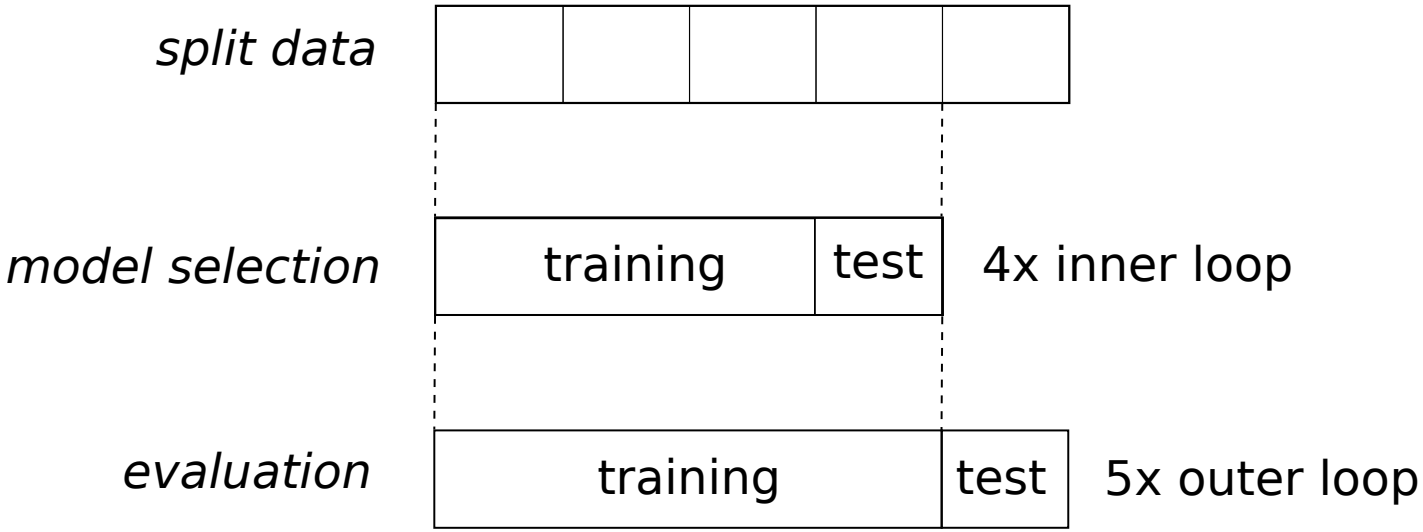




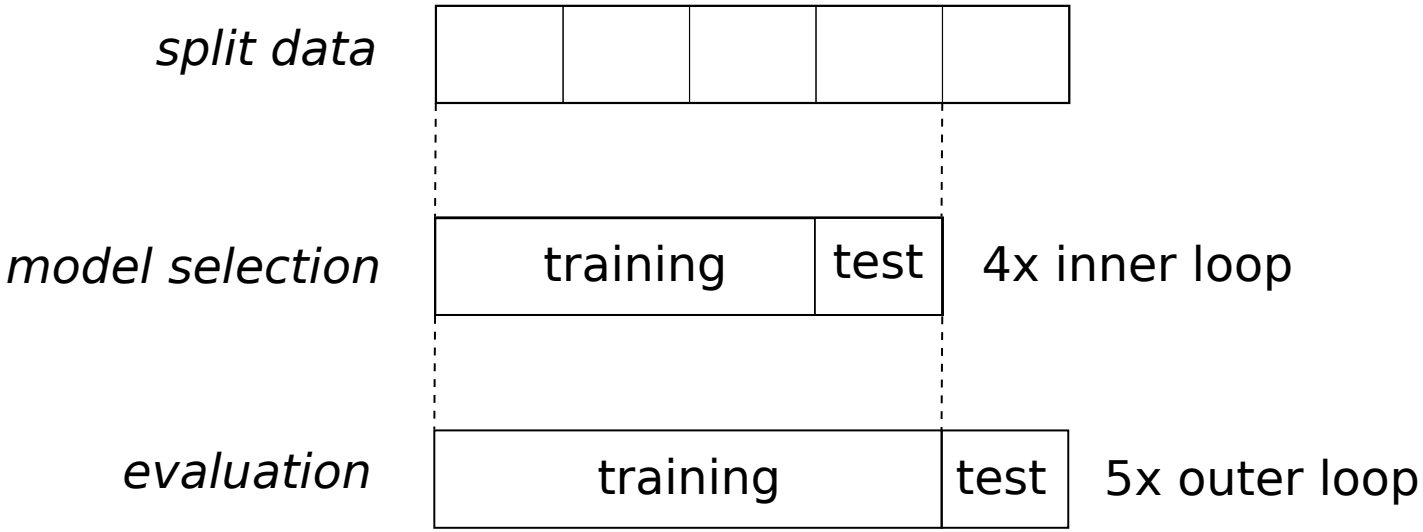




k-fold cross-validation



k-fold cross-validation



Don't even think about looking at the test set!

From objects to vectors

How to represent complex objects for kernel methods?

- explicit map to vector space: $\phi : M \to \mathbb{R}^n$
  - use standard kernel (e.g., linear, polynomial, gaussian) $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ on mapped features
- direct use of kernel function: $k : M \times M \to \mathbb{R}$

Feature Representation

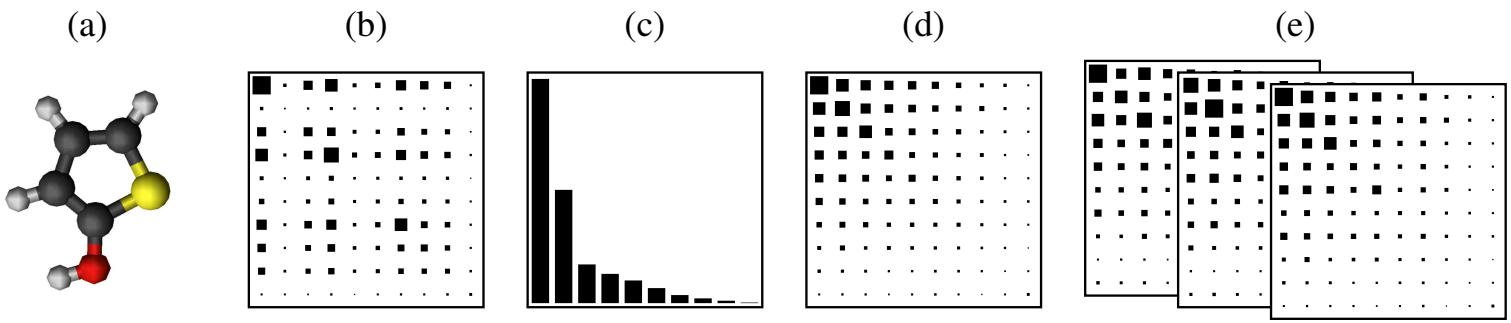
Given a physical object (molecule, crystal, etc.) and a property of interest, what is a good ML representation?

- no loss of valuable information
- support generalization
  - remove invariances
  - decompose problem
- incorporation of domain knowledge
- depends on data set, target function and learning method

Feature Representation - Molecules

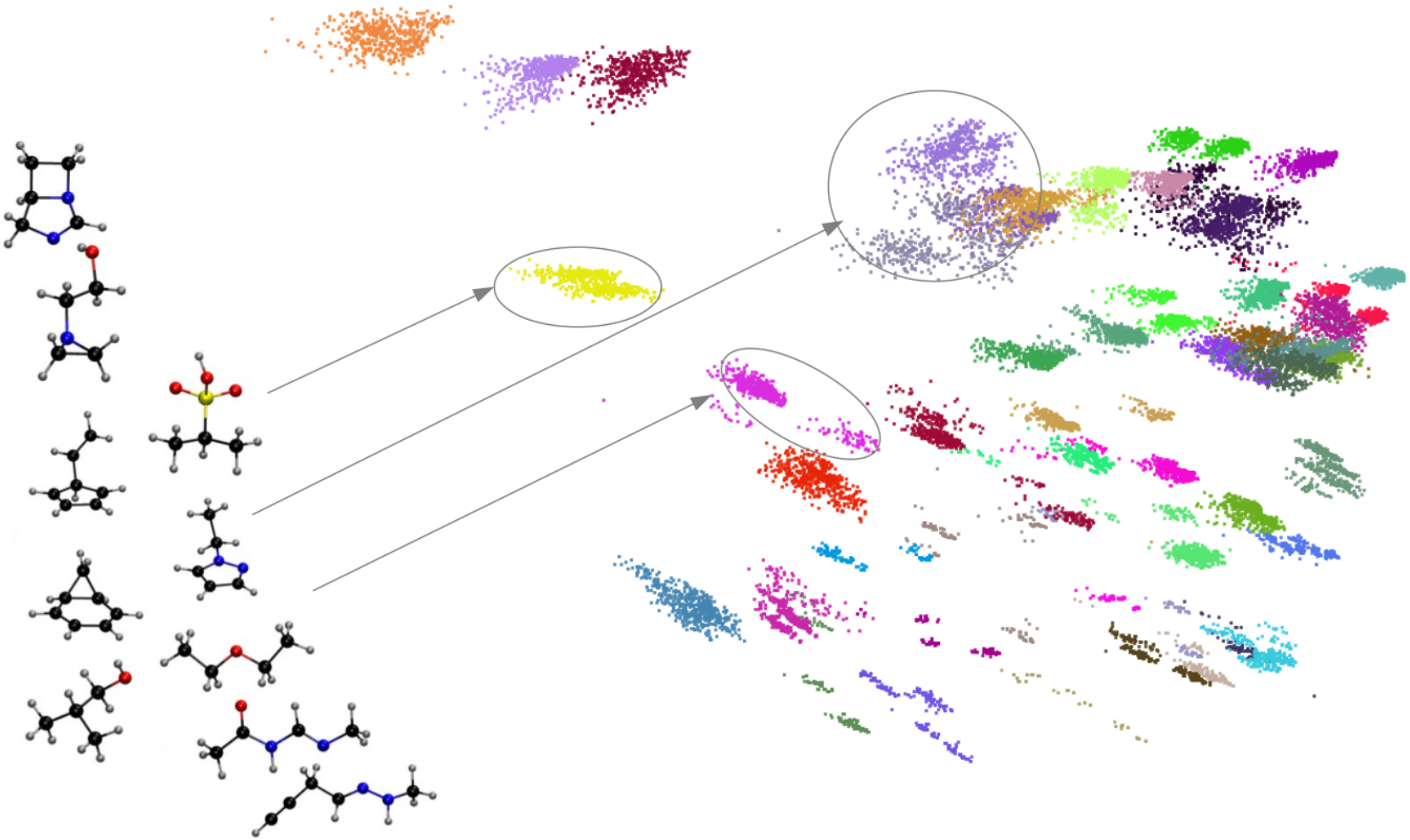
Coulomb matrix:

$$C_{ij} = \begin{cases} 0.5 Z_i^{2.4} & \text{if } i = j \\ \dfrac{Z_i Z_j}{\|r_i - r_j\|} & \text{if } i \neq j \end{cases}$$



(Rupp et al., 2012; Montavon et al., 2012)

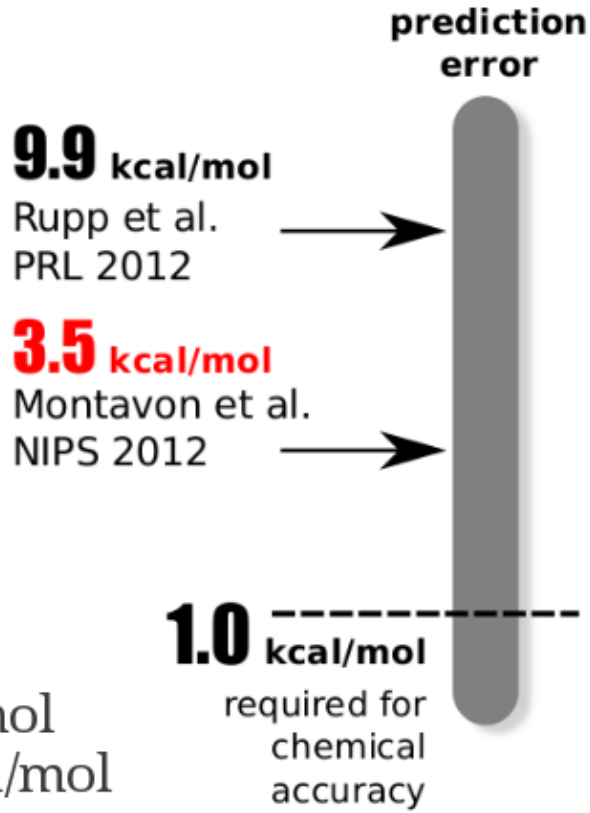
Feature Representation - Molecules



PCA of Coulomb matrices with atom permutations Montavon et al. (2013)

Results - Molecules

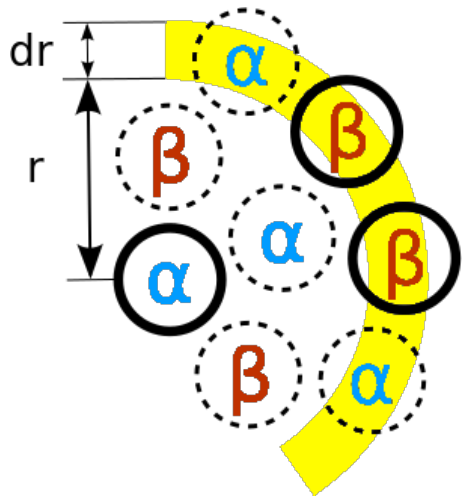
Results:
Mean predictor: **179** kcal/mol
K-nearest neighbors: **70** kcal/mol
Linear regression: **20.7** kcal/mol
Deep net, backprop: **11.8** kcal/mol
Gaussian kernel ridge regression: **8.7** kcal/mol
Deep net, backprop + permutations: **3.5** kcal/mol



G. Montavon, K. Hansen, S. Fazli, M. Rupp, F. Biegler, A. Ziehe, A. Tkatchenko, O. A. von Lilienfeld, K.-R. Müller, Learning Invariant Representations of Molecules for Atomization Energy Prediction, Advances in Neural Information Processing Systems (NIPS), 2012

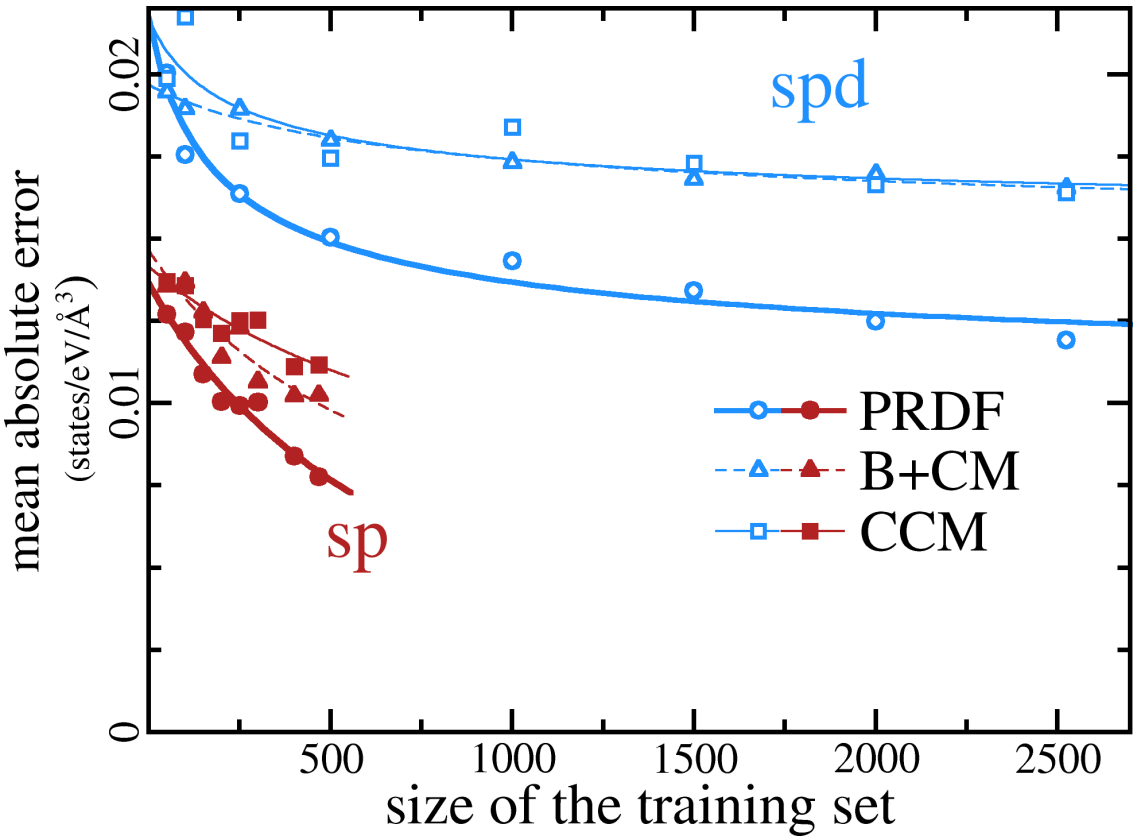
Feature Representation - Crystals

element pair	r_1	$\cdots$	r_n
$\alpha\ \alpha$	$g_{\alpha\alpha}(r_1)$	$\cdots$	$g_{\alpha\alpha}(r_n)$
$\alpha\ \beta$	$g_{\alpha\beta}(r_1)$	$\cdots$	$g_{\alpha\beta}(r_n)$
$\beta\ \alpha$	$g_{\beta\alpha}(r_1)$	$\cdots$	$g_{\beta\alpha}(r_n)$
$\beta\ \beta$	$g_{\beta\beta}(r_1)$	$\cdots$	$g_{\beta\beta}(r_n)$



Results - Crystals

Learning curve of DOS_{fermi} predictions



K.T. Schütt, H. Glawe, F. Brockherde, A. Sanna, K.-R. Müller, E.K.U. Gross, How to represent crystal structures for machine learning: towards fast prediction of electronic properties, arXiv, 2013

Machine Learning ...

... has been successfully applied to various research fields.

Machine Learning ...

... has been successfully applied to various research fields.

... is based on statistical learning theory.

Machine Learning ...

... has been successfully applied to various research fields.

... is based on statistical learning theory.

... provides fast and accurate predictions on previously unseen data.

Machine Learning ...

... has been successfully applied to various research fields.

... is based on statistical learning theory.

... provides fast and accurate predictions on previously unseen data.

... is able to model non-linear relationships of high-dimensional data.

Machine Learning ...

... has been successfully applied to various research fields.

... is based on statistical learning theory.

... provides fast and accurate predictions on previously unseen data.

... is able to model non-linear relationships of high-dimensional data.

Feature representation is key!

Literature I

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.

Montavon, G., Hansen, K., Fazli, S., Rupp, M., Biegler, F., Ziehe, A., Tkatchenko, A., Lilienfeld, A. V., and Müller, K.-R. (2012). Learning invariant representations of molecules for atomization energy prediction. In *Advances in Neural Information Processing Systems*, pages 449–457.

Montavon, G., Rupp, M., Gobre, V., Vazquez-Mayagoitia, A., Hansen, K., Tkatchenko, A., Müller, K.-R., and von Lilienfeld, O. A. (2013). Machine learning of molecular electronic properties in chemical compound space. *arXiv preprint arXiv:1305.7074*.

Müller, K.-R., Mika, S., Ratsch, G., Tsuda, K., and Scholkopf, B. (2001). An introduction to kernel-based learning algorithms. *Neural Networks, IEEE Transactions on*, 12(2):181–201.

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

Rupp, M., Tkatchenko, A., Müller, K.-R., and von Lilienfeld, O. A. (2012). Fast and accurate modeling of molecular atomization energies with machine learning. *Physical Review Letters*, 108(5):058301.

Schölkopf, B., Smola, A., and Müller, K.-R. (1997). Kernel principal component analysis. In *Artificial Neural Networks—ICANN'97*, pages 583–588. Springer.

Vapnik, V. N. and Chervonenkis, A. Y. (1971). On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280.