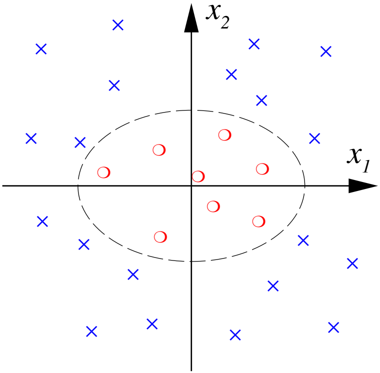
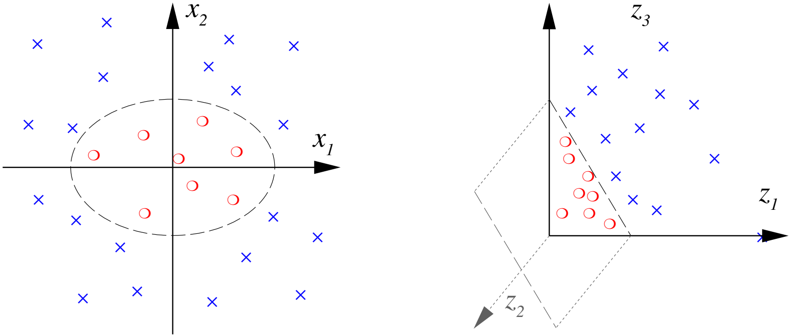
Non-separable Case



This data set is not properly separable with lines (also when using many slack variables)

Separate in Higher-Dim. Space

Map data in higher-dimensional space and separate it there with a hyperplane



$$\begin{aligned} \Phi : \mathbb{R}^2 &\to \mathbb{R}^3 \\ (x_1, x_2) &\mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}x_1x_2, x_2^2) \end{aligned}$$

Feature Space

Apply the mapping

$$\begin{aligned}\Phi : \mathbb{R}^N &\rightarrow \mathcal{F} \\ \mathbf{x} &\mapsto \Phi(\mathbf{x})\end{aligned}$$

to the data $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m \in \mathcal{X}$ and construct separating hyperplane in $\mathcal{F}$ instead of $\mathcal{X}$. The samples are preprocessed as $(\Phi(\mathbf{x}_1), y_1), \ldots, (\Phi(\mathbf{x}_m), y_m) \in \mathcal{F} \times \{\pm 1\}$.

Obtained decision function:

$$\begin{aligned}f(\mathbf{x}) &= \operatorname{sgn}\left(\sum_{i=1}^{m} y_i \alpha_i \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}_i) \rangle + b\right) \\ &= \operatorname{sgn}\left(\sum_{i=1}^{m} y_i \alpha_i\, k(\mathbf{x}, \mathbf{x}_i) + b\right)\end{aligned}$$

How about patters $\mathbf{x} \in \mathbb{R}^N$ and product features of order d? $\mathrm{Dim}(\mathcal{F})$ grows like N^d. Example $N = 16 \times 16$, and $d = 5 \longrightarrow$ dimension 10^{10}.

Kernels

A *kernel* is a function k, such that for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$

$$k(\mathbf{x}, \mathbf{y}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle,$$

where Φ is a mapping from $\mathcal{X}$ to an dot product feature space $\mathcal{F}$.

The $m \times m$ matrix K with elements $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ is called kernel matrix or Gram matrix. The kernel matrix is symmetric and positive semi-definite, i.e. for all $a_i \in \mathbb{R}$, $i = 1, \ldots, m$, we have

$$\sum_{i,j=1}^{m} a_i a_j K_{ij} \geq 0$$

Positive semi-definite kernels are exactly those giving rise to a positive semi-definite kernel matrix K for all m and all sets $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\} \subseteq \mathcal{X}$.

The Kernel Trick Example

Example : compute 2nd order products of two "pixels", i.e.

$$\mathbf{x} = (x_1, x_2) \text{ and } \Phi(\mathbf{x}) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

$$\begin{aligned}
\langle \Phi(\mathbf{x}), \Phi(\mathbf{z}) \rangle &= (x_1^2, \sqrt{2}x_1x_2, x_2^2)(z_1^2, \sqrt{2}z_1z_2, z_2^2)^T \\
&= ((x_1, x_2)(z_1, z_2)^T)^2 \\
&= (\mathbf{x} \cdot \mathbf{z}^T)^2 \\
&= : k(\mathbf{x}, \mathbf{z})
\end{aligned}$$

Kernel without knowing Φ

Recall: mapping $\Phi : \mathbb{R}^N \to \mathcal{F}$. SVM depends on the data through dot products in $\mathcal{F}$, i.e. functions of the form

$$\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$$

- With k such that $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$, it is not necessary to even know what $\Phi(\mathbf{x})$ is.

Example: $k(\mathbf{u}, \mathbf{v}) = \exp\left(-\frac{\|\mathbf{u}-\mathbf{v}\|^2}{\gamma}\right)$, in this example $\mathcal{F}$ is infinite dimensional.

Feature Space (Optimization Problem)

Quadratic optimization problem (soft margin) with kernel:

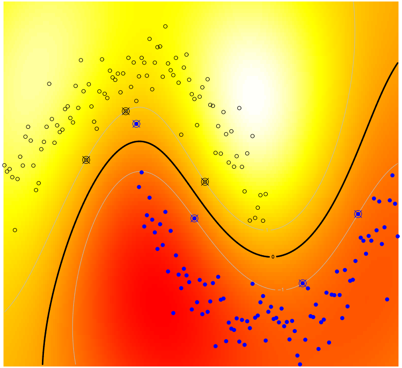
$$\text{maximize} \qquad W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$$

$$\text{subject to} \qquad 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, m \text{ and } \sum_{i=1}^{m} \alpha_i y_i = 0$$

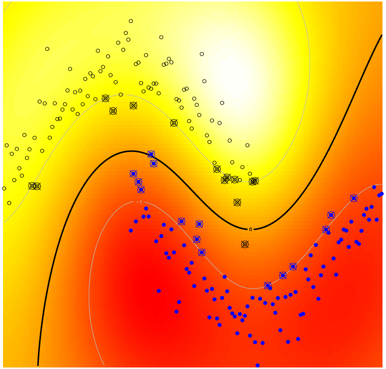
(Standard) Kernels

$$
\begin{aligned}
\text{Linear} \quad & k_0(\mathbf{u},\mathbf{v}) && = \langle \mathbf{u},\mathbf{v}\rangle \\
\text{Polynomial} \quad & k_1(\mathbf{u},\mathbf{v}) && = (\langle \mathbf{u},\mathbf{v}\rangle + \Theta)^d \\
\text{Gaussian} \quad & k_2(\mathbf{u},\mathbf{v}) && = \exp\left(-\frac{\|\mathbf{u}-\mathbf{v}\|^2}{\gamma}\right) \\
\text{Sigmoidal} \quad & k_3(\mathbf{u},\mathbf{v}) && = \tanh(\kappa\langle \mathbf{u},\mathbf{v}\rangle + \Theta)
\end{aligned}
$$

SVM Results for Gaussian Kernel

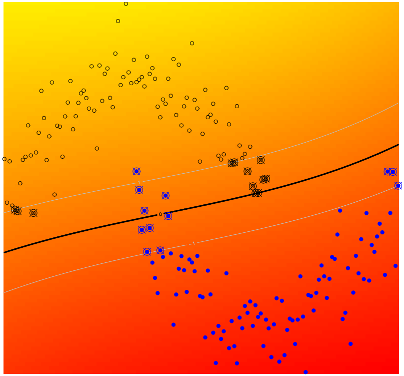


$\gamma = 0.5,\ C = 50$

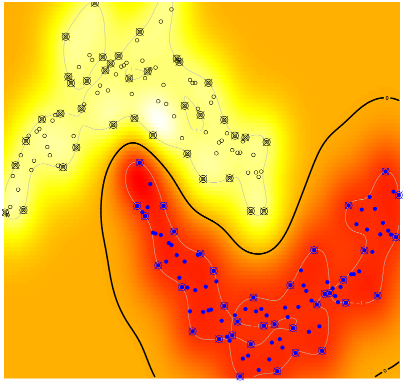


$\gamma = 0.5,\ C = 1$

SVM Results for Gaussian Kernel (cont.)



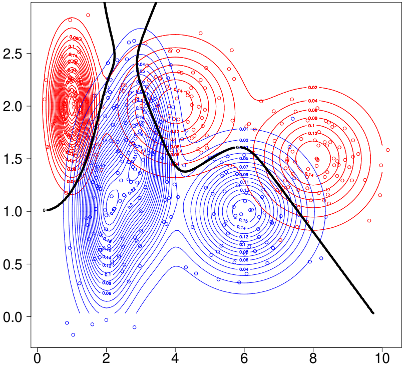
$\gamma = 0.02,\ C = 50$

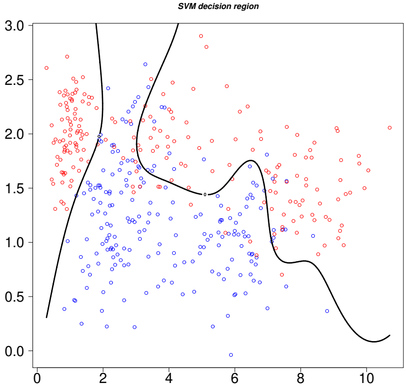


$\gamma = 10,\ C = 50$

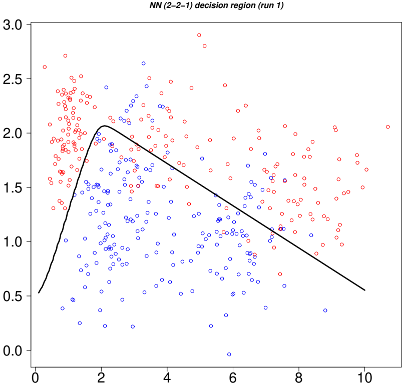
See interactive demo.

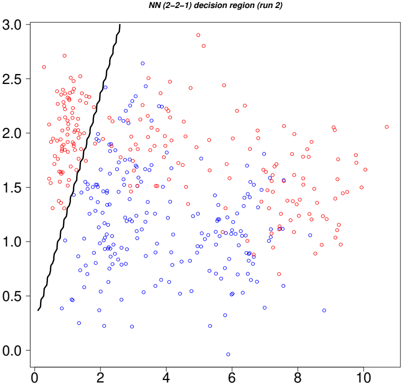
Bayes Decision and SVM (Gaussian Kernel)



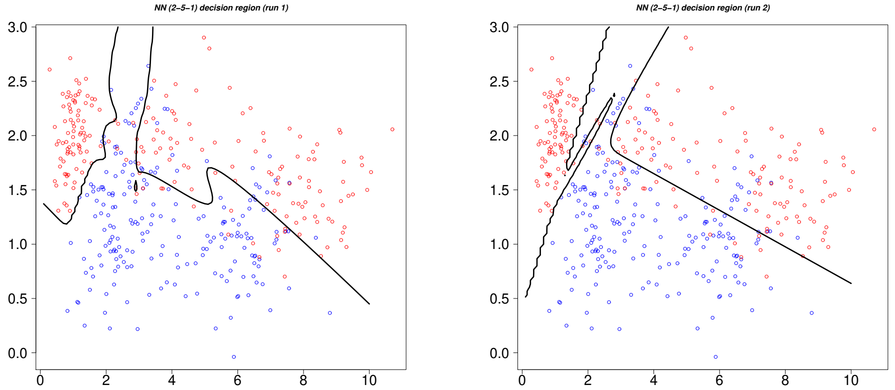


Neural Networks (2-2-1)

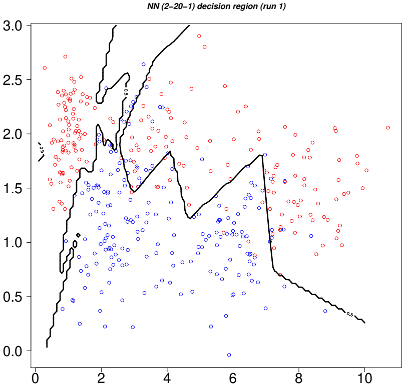


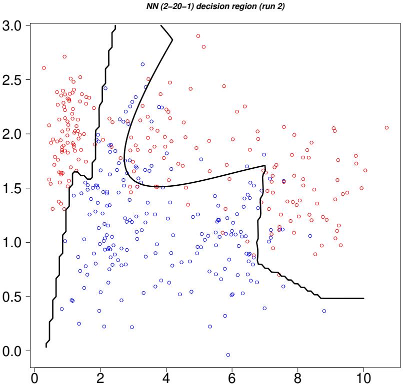


Neural Networks (2-5-1)



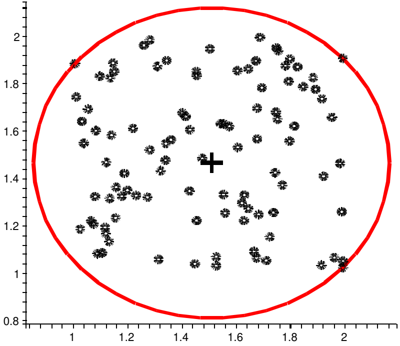
Neural Networks (2-20-1)





One-Class SVM for Novelty Detection

Idea: enclose data with a hypersphere and classify new data as *normal* if it falls within the hypersphere and otherwise as anomalous data.



Minimum Enclosing Hypersphere

Given normal data $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\} \in \mathbb{R}^d$ and let r be the radius of the hypersphere and $\mathbf{c} \in \mathcal{F}$ the center. To find the minimum enclosing hypersphere we have to solve the following optimization problem:

$$\begin{aligned} \text{minimize} \quad & r^2 \\ \text{subject to} \quad & \|\Phi(\mathbf{x}_i) - \mathbf{c}\|^2 \leq r^2, \quad i = 1, \ldots, m. \end{aligned}$$

Lagrangian multiplier $\alpha_i \geq 0$ for each constraint

$$L(\mathbf{c}, r, \boldsymbol{\alpha}) = r^2 + \sum_{i=1}^{m} \alpha_i \left\{ \|\Phi(\mathbf{x}_i) - \mathbf{c}\|^2 - r^2 \right\}$$

Minimum Enclosing Hypersphere (cont.)

Setting the derivatives with respect to $\mathbf{c}$ and r to zero

$$\begin{aligned}\frac{\partial L(\mathbf{c}, r, \boldsymbol{\alpha})}{\partial \mathbf{c}} &= 2\sum_{i=1}^{m} \alpha_i(\Phi(\mathbf{x}_i) - \mathbf{c}) = \mathbf{0} \\ \frac{\partial L(\mathbf{c}, r, \boldsymbol{\alpha})}{\partial r} &= 2r\left(1 - \sum_{i=1}^{m} \alpha_i\right) = 0\end{aligned}$$

one obtains the following equations

$$\sum_{i=1}^{m} \alpha_i = 1 \text{ and } \mathbf{c} = \sum_{i=1}^{m} \alpha_i \Phi(\mathbf{x}_i). \quad (1)$$

Minimum Enclosing Hypersphere (cont.)

Inserting relation (1) into

$$
\begin{aligned}
L(\mathbf{c}, r, \boldsymbol{\alpha}) &= r^2 + \sum_{i=1}^{m} \alpha_i \left\{ \|\Phi(\mathbf{x}_i) - \mathbf{c}\|^2 - r^2 \right\} \\
&= \sum_{i=1}^{m} \alpha_i \|\Phi(\mathbf{x}_i) - \mathbf{c}\|^2 \\
&= \sum_{i=1}^{m} \alpha_i k(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i,j=1}^{m} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j)
\end{aligned}
$$

gives the dual form.[3]

[3] Note: In dual form we got rid of **c** and $\Phi(\cdot)$.

Minimum Enclosing Hypersphere (cont.)

To find $\boldsymbol{\alpha}$ in dual form, solve optimization problem:

$$\text{maximize} \quad W(\boldsymbol{\alpha}) = \sum_{i=1}^{m} \alpha_i k(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i,j=1}^{m} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j)$$

$$\text{subject to} \quad \sum_{i=1}^{m} \alpha_i = 1, \text{ and } \alpha_i \geq 0, \quad i = 1, \ldots, m.$$

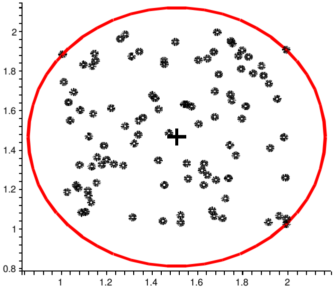
Recall: Lagrange multiplier can be non-zero only if the corresponding inequality constraint is an equality at the solution.

Minimum Enclosing Hypersphere (cont.)

The KKT complementarity conditions are satisfied by the optimal solutions $\boldsymbol{\alpha}, (\mathbf{c}, r)$

$$\alpha_i \left\{ \|\Phi(\mathbf{x}_i) - \mathbf{c}\|^2 - r^2 \right\}, \quad i = 1, \ldots, m.$$

This implies that only training examples $\mathbf{x}_i$ that lie on the surface of the optimal hypersphere have their corresponding $\alpha_i > 0$.



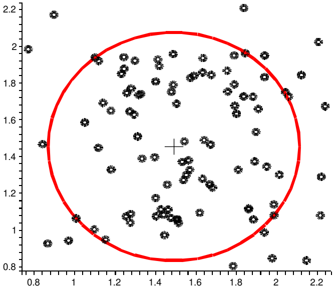
Decision Function

$$
\begin{aligned}
f(\mathbf{x}) &= \operatorname{sgn}(r^2 - \|\Phi(\mathbf{x}) - \mathbf{c}\|^2) \\
&= \operatorname{sgn}\left(r^2 - \left\{(\Phi(\mathbf{x}) \cdot \Phi(\mathbf{x})) - 2\sum_{i=1}^{m} \alpha_i (\Phi(\mathbf{x}) \cdot \Phi(\mathbf{x}_i)) \right.\right. \\
&\quad \left.\left. + \sum_{i,j=1}^{m} \alpha_i \alpha_j (\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)) \right\}\right) \\
&= \operatorname{sgn}\left(r^2 - \left\{k(\mathbf{x}, \mathbf{x}) - 2\sum_{i=1}^{m} \alpha_i k(\mathbf{x}, \mathbf{x}_i) \right.\right. \\
&\quad \left.\left. + \sum_{i,j=1}^{m} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \right\}\right)
\end{aligned}
$$

Soft Enclosing Hypersphere

If we have some noise in our training set the “hard” enclosing hypersphere approach may force a larger radius than should really be needed. In other words, the solution would not be *robust*.

Aim: Find minimum enclosing hypersphere that contains (allmost) all training examples, but not some small portion of extreme training examples.



Soft Enclosing Hypersphere (cont.)

Introduce slack variables $\boldsymbol{\xi}, \xi_i \geq 0, i = 1, \ldots, m$

$$\begin{aligned} \text{minimize} \quad & r^2 + C \sum_{i=1}^{m} \xi_i \\ \text{subject to} \quad & \|\Phi(\mathbf{x}_i) - \mathbf{c}\|^2 \leq r^2 + \xi_i, \quad \xi_i \geq 0,\ i = 1, \ldots, m. \end{aligned}$$

Lagrangian multiplier $\alpha_i, \beta_i \geq 0$ for each constraint

$$\begin{aligned} L(\mathbf{c}, r, \boldsymbol{\alpha}, \boldsymbol{\beta}) \quad = \quad & r^2 + C \sum_{i=1}^{m} \xi_i \\ & + \sum_{i=1}^{m} \alpha_i \left\{ \|\Phi(\mathbf{x}_i) - \mathbf{c}\|^2 - r^2 - \xi_i \right\} - \sum_{i=1}^{m} \beta_i \xi_i \end{aligned}$$

Soft Enclosing Hypersphere (cont.)

Setting partial derivatives to **0** gives

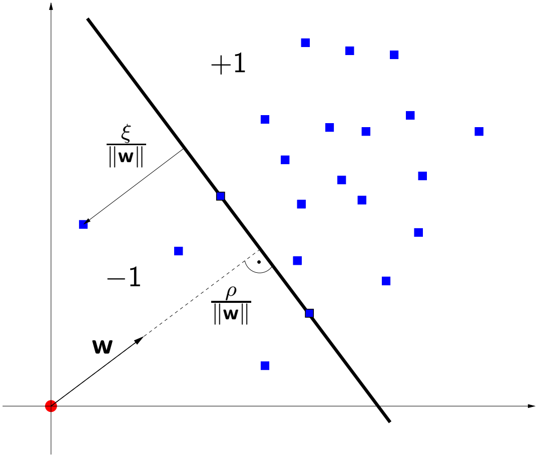
$$\sum_{i=1}^{m} \alpha_i = 1, \quad \mathbf{c} = \sum_{i=1}^{m} \alpha_i \Phi(\mathbf{x}_i)$$

This leads to the dual form

$$\begin{aligned} \text{minimize} \quad & \sum_{i,j=1}^{m} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i=1}^{m} \alpha_i k(\mathbf{x}_i, \mathbf{x}_i) \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C, \quad \sum_{i=1}^{m} \alpha_i = 1 \end{aligned}$$

Hyperplane One-Class SVM

Idea: Separate in high-dimensional feature space $\mathcal{F}$, the points from the origin (circled point) with a maximum distance, and allow $\nu \cdot m$ many "outliers" which lie between the origin and the hyperplane, i.e. the -1 side.



Hyperplane One-Class SVM (cont.)

Normal vector of the hyperplane is determined by solving the primal quadratic optimization problem

$$\text{minimize} \quad \tfrac{1}{2}\|\mathbf{w}\|^2 + \tfrac{1}{\nu m}\textstyle\sum_i \xi_i - \rho \tag{2}$$
$$\text{subject to} \quad \langle \mathbf{w}, \Phi(\mathbf{x}_i)\rangle \geq \rho - \xi_i, \xi_i > 0,\ i = 1, \ldots, m. \tag{3}$$

Lagrangian multiplier $\alpha_i, \beta_i \geq 0$ for each constraint

$$\begin{aligned} L(\mathbf{w}, \boldsymbol{\xi}, \rho, \boldsymbol{\alpha}, \boldsymbol{\beta}) &= \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{\nu m}\sum_i \xi_i - \rho \\ &\quad - \sum_{i=1}^{m} \alpha_i(\langle \mathbf{w}, \Phi(\mathbf{x}_i)\rangle - \rho + \xi_i) - \sum_{i=1}^{m} \beta_i \xi_i \end{aligned}$$

Reformulating (2) and (3) to a dual optimization problem in terms of a kernel function $k(\cdot, \cdot)$, one obtains

Hyperplane One-Class SVM (cont.)

$$\begin{aligned} &\text{maximize} && \tfrac{1}{2}\textstyle\sum_{i,j=1}^{m} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) && (4) \\ &\text{subject to } 0 \leq \alpha_i \leq \tfrac{1}{\nu m}, i = 1, \ldots, m \text{ and } \textstyle\sum_{i=1}^{m} \alpha_i = 1. && && (5) \end{aligned}$$

Differentiating the primal with respect to $\mathbf{w}$, one gets $\mathbf{w} = \sum_{i=1}^{m} \alpha_i \Phi(\mathbf{x}_i)$.

Recall KKT theorem: For $\alpha_i > 0$ the corresponding pattern $\mathbf{x}_i$ satisfies

$$\rho = \langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle = \sum_{j=1}^{m} \alpha_j k(x_j, x_i)$$

Hyperplane One-Class SVM (cont.)

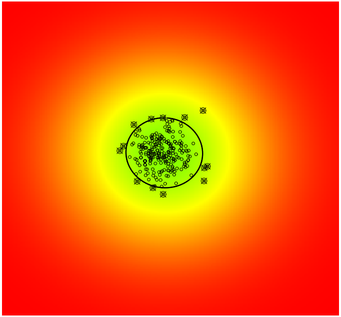
The decision function (left/right side of the hyperplane):

$$
\begin{aligned}
f(\mathbf{x}) &= \operatorname{sgn}(\langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle - \rho) \\
&= \operatorname{sgn}\left(\sum_{i=1}^{m} \alpha_i k(\mathbf{x}_i, \mathbf{x}) - \rho \right)
\end{aligned}
$$

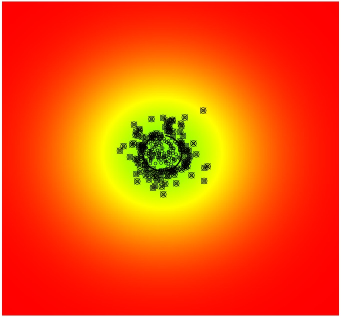
ν-**Property**:

- ν is an upper bound on the fraction of outliers.
- ν is a lower bound on the fraction of Support Vectors.

Hyperplane One-Class SVM Example



$\nu = 0.05$



$\nu = 0.5$

See interactive demo.

Support Vector Regression

Basic idea: map the data $\mathbf{x}$ into a high-dimensional feature space $\mathcal{F}$ via a nonlinear mapping Φ, and do *linear* regression in this space.

$$f(\mathbf{x}) = \langle \mathbf{w}, \Phi(\mathbf{x}) \rangle + b \text{ with } \Phi : \mathbb{R}^d \to \mathcal{F}, \mathbf{w} \in \mathcal{F}.$$

Linear regression in a high dimensional feature space corresponds to *nonlinear* regression in the low dimensional space $\mathbb{R}^d$.

Vapnik's ϵ-insensitive loss function:

$$|y - f(\mathbf{x})|_\epsilon := \max\{0, |y - f(\mathbf{x})| - \epsilon\}$$

Find function $f(\mathbf{x})$ that has at most ϵ deviation from all the targets y_i

Support Vector Regression (cont.)

Estimate linear regression $f(\mathbf{x}) = \langle \mathbf{w}, \Phi(\mathbf{x}) \rangle + b$ leads to the problem of minimizing the term

$$\frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{n} |y_i - f(\mathbf{x}_i)|_\epsilon$$

In the soft margin case one needs two types of slack variables $(\boldsymbol{\xi}, \boldsymbol{\xi}^*)$ for the two cases $f(\mathbf{x}_i) - y_i > \epsilon$ and $y_i - f(\mathbf{x}_i) > \epsilon$.

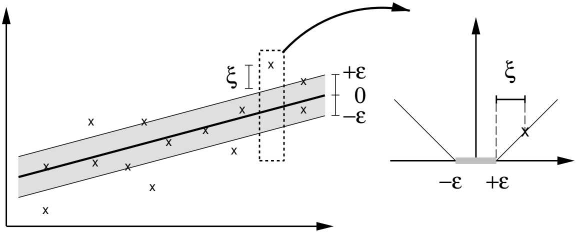


Figure is taken from Schölkopf's and Smola's book (Learning with Kernels)

Support Vector Regression (cont.)

Optimization problem is given by:

$$
\begin{array}{lrcl}
\text{minimize} & \multicolumn{3}{l}{\frac{1}{2}\|\mathbf{w}\|^2 + C \cdot \sum_{i=1}^{n}(\xi_i + \xi_i^*)} \\
 & f(\mathbf{x}_i) - y_i & \leq & \epsilon + \xi_i \\
\text{subject to} & y_i - f(\mathbf{x}_i) & \leq & \epsilon + \xi_i^* \\
 & \xi_i, \xi_i^* \geq 0 & & \text{for all } i = 1, \ldots, n
\end{array}
$$

Support Vector Regression (cont.)

Introducing Lagrange multipliers $\boldsymbol{\alpha}, \boldsymbol{\alpha}^*$ (dual form):

$$
\begin{aligned}
\text{maximize} \quad & -\epsilon \sum_{i=1}^{n} (\alpha_i^* + \alpha_i) + \sum_{i=1}^{n} (\alpha_i^* - \alpha_i) y_i \\
& -\frac{1}{2} \sum_{i,j}^{n} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) k(\mathbf{x}_i, \mathbf{x}_j) \\
\text{subject to} \quad & 0 \leq \alpha_i, \alpha_i^* \leq C \text{ for all } i = 1, \ldots, n \text{ and} \\
& \sum_{i=1}^{n} (\alpha_i^* - \alpha_i) = 0
\end{aligned}
$$

Regression estimate takes the form

$$
f(\mathbf{x}) = \sum_{i=1}^{n} (\alpha_i^* - \alpha_i) k(\mathbf{x}_i, \mathbf{x}) + b
$$

Support Vector Regression (cont.)

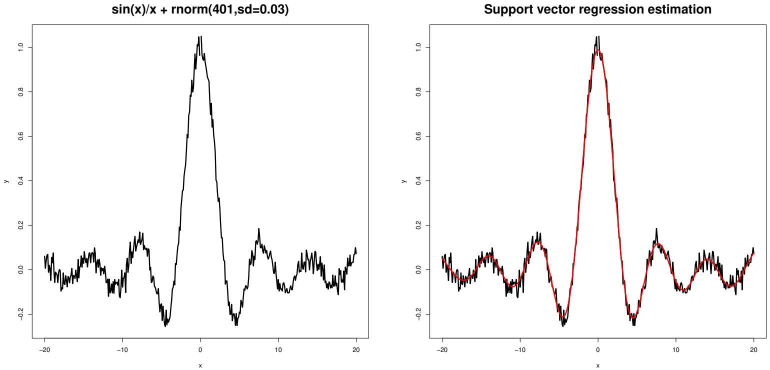
Offset b can be computed by exploiting Karush-Kuhn-Tucker conditions: $f(\mathbf{x}_i) - y_i \leq \epsilon + \xi_i$ becomes an equality with $\xi_i = 0$ if $0 < \alpha_i < C$ and $y_i - f(\mathbf{x}_i) \leq \epsilon + \xi_i^*$ becomes an equality with $\xi_i^* = 0$ if $0 < \alpha_i^* < C$ that is:

$$
\begin{aligned}
\alpha_i(\epsilon + \xi_i - y_i + \langle \mathbf{w}, \Phi(\mathbf{x}_i)\rangle + b) &= 0 \\
\alpha_i^*(\epsilon + \xi_i^* + y_i - \langle \mathbf{w}, \Phi(\mathbf{x}_i)\rangle - b) &= 0
\end{aligned}
$$

and leads to solution

$$
\begin{aligned}
b &= y_i - \langle \mathbf{w}, \Phi(\mathbf{x}_i)\rangle - \epsilon \quad \text{for } \alpha_i \in (0, C) \\
b &= y_i - \langle \mathbf{w}, \Phi(\mathbf{x}_i)\rangle + \epsilon \quad \text{for } \alpha_i^* \in (0, C)
\end{aligned}
$$

Support Vector Regression Example




```
library(kernlab);
x <- seq(-20,20,0.1);
y <- sin(x)/x + rnorm(401,sd=0.03);
# train SVM
reg.svm <- ksvm(x,y,epsilon=0.01,kpar=list(sigma=16),cross=3);
plot(x,y,type="l",lwd=3);
lines(x,predict(reg.svm,x),col="red",lwd=3);
```

Summary & End

- SVM's are (from my biased perspective) simpler to train than neural networks.
- SVM' are useful classification and regression techniques on "small" data sets.
- Should be in your machine learning toolbox along with deep neural networks.

Thank you for your attention

Feel free to contact me t.stibor@gsi.de