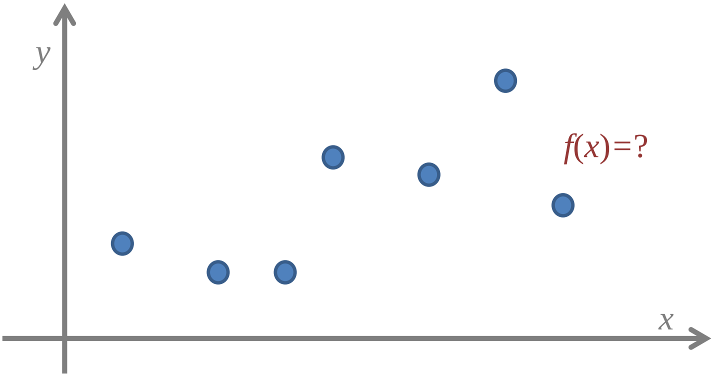
Lecture 15

CM50264: Machine Learning 1

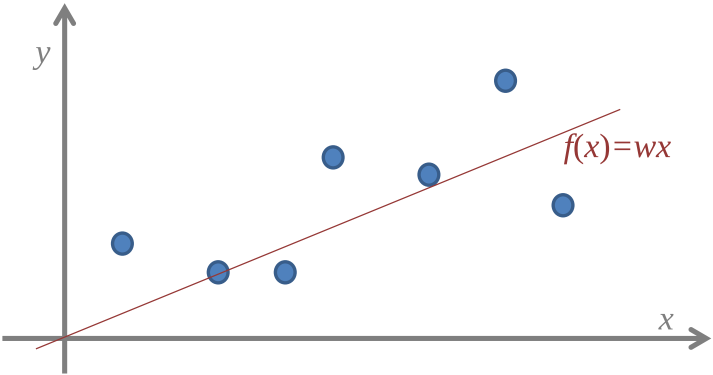
Regularized Regression and Support Vector Machines

Kwang In Kim

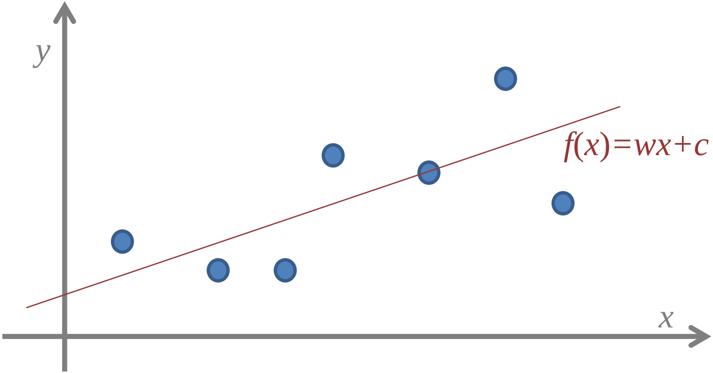
Previously in machine learning ...
Linear regression: One-D example



Previously in machine learning ...
Linear regression: One-D example



Previously in machine learning ...
Affine regression: One-D example



Previously in machine learning ...
Regression problem

We are given a training dataset (pairs of input and output)

$$D = \{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\} \subset \mathbb{R}^n \times \mathbb{R}.$$

Our goal is to find a function

$$f : \mathbb{R}^n \to \mathbb{R}$$

such that its output $f(\mathbf{x}^*)$ for an unseen input $\mathbf{x}^* \notin D$ is close to the underlying ground-truth output y^*:

More formally, we want to find a function f^* that minimizes

$$\int l(f(\mathbf{x}), y) dP(\mathbf{x}, y),$$

for a loss function $l(\cdot, \cdot) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, e.g.

$$l(f(\mathbf{x}), y) = (f(\mathbf{x}) - y)^2.$$

Previously in machine learning ... Least-squares regression

Training data:

$$D = \{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\} \subset \mathbb{R}^n \times \mathbb{R}.$$

We wish to minimize

$$\mathcal{O}(f) = \int (f(\mathbf{x}) - y)^2 dP(\mathbf{x}, y).$$

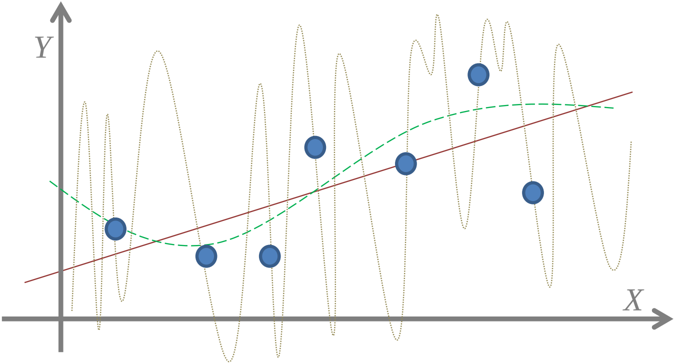
However, in practice we do not have access to the underlying data generating process $P(\mathbf{x}, y)$.

Instead, we minimize the empirical mean squared error (aka training error):

$$\mathcal{O}'(f) = \frac{1}{N} \sum_{i=1}^{N} (f(\mathbf{x}^i) - y^i)^2.$$

Overfitting

Fitting a function f to finitely many training data points $D = \{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\}$ can lead to overfitting.



The function represented as a doted line fits perfectly to data (circles). But it may not well represent the underlying data generating process.

Tikhonov regularization

To avoid overfitting, Tikhonov regularization enforces smoothness of the potential solution f.[1]

[1] This process was conceived as a systematic framework for solving mathematical inverse problems. An in-depth discussion and examples can be found in Ch16 of [Kre].

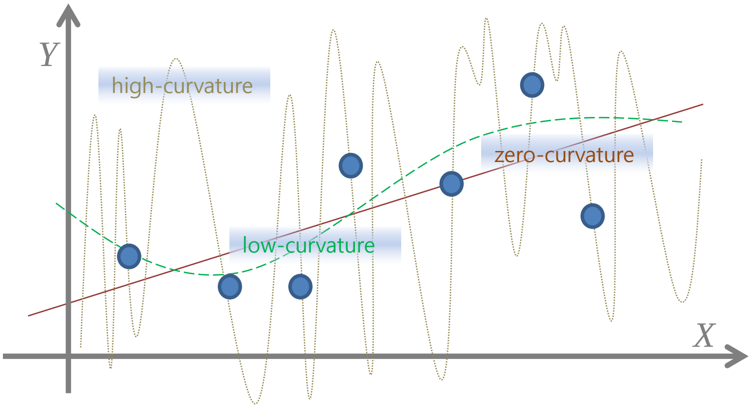
Notion of smoothness

- There are various notions of smoothness or inverse complexity:
  e.g. number of parameters (of physical system) and length of the source code (of a software).
- When the solution is a function $f : \mathbb{R}^n \to \mathbb{R}$,
  a common measure of (inverse) smoothness is

$$\|Df\|^2 := \int \|Df(\mathbf{x})\|^2 \mathrm{d}\mathbf{x},$$

  where D is a derivative operator.
- E.g. when D is the second order derivative operator (i.e. $Df = \frac{\partial^2 f}{\partial x_i x_j}$), $\|Df\|^2$ is called thin-plate spline energy which measures the curvature of f.

One-D regression examples



The function represented with a doted line has high energy

Linear regression

We are given a set of *training* data points

$$D = \{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\} \subset \mathbb{R}^n \times \mathbb{R}.$$

Our goal is to find a function $f : \mathbb{R}^n \to \mathbb{R}$ that minimizes

$$\mathcal{O}'(f) = \frac{1}{N} \sum_{i=1}^{N} (f(\mathbf{x}^i) - y^i)^2.$$

In linear regression, f is linear:

$$f(\mathbf{x}) = \mathbf{x}^\top \mathbf{w}$$

or affine:

$$f(\mathbf{x}) = \mathbf{x} \cdot \mathbf{w} + c = [\mathbf{x}^\top, 1][\mathbf{w}^\top, c]^\top := \mathbf{x}'^\top \mathbf{w}'.$$

Linear least-squares regression

$$f^* = \operatorname*{arg\,min}_{f \in \text{All linear functions from } \mathbb{R}^n \text{ to } \mathbb{R}} \frac{1}{N} \sum_{i=1}^{N} (f(\mathbf{x}^i) - y^i)^2. \qquad (1)$$

A linear function $f : \mathbb{R}^n \to \mathbb{R}$ is represented as

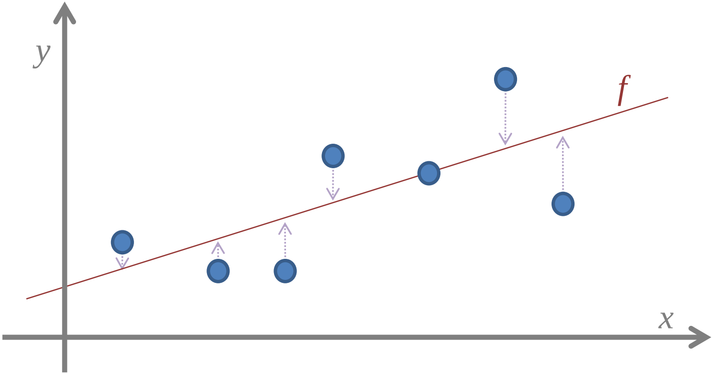
$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}.$$

Equivalently, a linear function f is parametrized by a vector $\mathbf{w} \in \mathbb{R}^n$.

Problem (1) is equivalent to

$$\begin{aligned}
\mathbf{w}^* &= \operatorname*{arg\,min}_{\mathbf{w} \in \mathbb{R}^n} \frac{1}{N} \sum_{i=1}^{N} (f(\mathbf{x}^i) - y^i)^2 \\
&= \operatorname*{arg\,min}_{\mathbf{w} \in \mathbb{R}^n} \frac{1}{N} \sum_{i=1}^{N} (\mathbf{w}^\top \mathbf{x}^i - y^i)^2 \\
&= \operatorname*{arg\,min}_{\mathbf{w} \in \mathbb{R}^n} \sum_{i=1}^{N} (\mathbf{w}^\top \mathbf{x}^i - y^i)^2.
\end{aligned}$$

Linear regression: One-D example



Linear least-squares regression

$$D = \{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\} \subset \mathbb{R}^n \times \mathbb{R}$$

$$\mathbf{w}^* = \arg\min_{\mathbf{w} \in \mathbb{R}^n} \sum_{i=1}^{N} (\mathbf{w}^\top \mathbf{x}^i - y^i)^2 \qquad (2)$$

- With a data matrix $\mathbf{X}$ and label vector $\mathbf{y}$:

$$\mathbf{X} = [\mathbf{x}^1, \ldots, \mathbf{x}^N]$$
$$\mathbf{y} = [y^1, \ldots, y^N]^\top,$$

we obtain a vectorized representation of the original optimization problem (Eq. 2):

$$\mathbf{w}^* = \arg\min_{\mathbf{w} \in \mathbb{R}^n} \|\mathbf{X}^\top \mathbf{w} - \mathbf{y}\|^2. \qquad (3)$$

- This problem is convex and differentiable. Therefore, the optimal solution is found by setting the derivative of r.h.s. of Eq. 3 w.r.t. $\mathbf{w}$ equal to zero:

$$\mathbf{X}\mathbf{X}^\top \mathbf{w} = \mathbf{X}\mathbf{y}.$$

Linear least-squares regression

$$D = \{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\} \subset \mathbb{R}^n \times \mathbb{R}$$

$$\mathbf{X}\mathbf{X}^\top \mathbf{w}^* = \mathbf{X}\mathbf{y}. \tag{4}$$

- The data matrix $\mathbf{X} = [\mathbf{x}^1, \ldots, \mathbf{x}^N] \subset \mathbb{R}^{n \times N}$ has rank $\min(n, N)$.
- The system matrix $\mathbf{X}\mathbf{X}^\top \subset \mathbb{R}^{n \times n}$ has rank $\min(n, N)$:
  - If $N > n$, $\mathbf{X}\mathbf{X}^\top$ has full rank.
  - If $N < n$, $\mathbf{X}\mathbf{X}^\top$ is rank deficient:
    In this case, the linear system (Eq. 4) has infinitely many solutions $\mathbf{w}$.
- In high-dimensional spaces, even linear regression can and will overfit! We need regularization.

Regularized linear least-squares regression

$$f(\mathbf{x}) = \mathbf{x}^\top \mathbf{w}$$

Linear functions have only first-order derivatives:
The only option for Df is $\nabla f = \mathbf{w}$.

- (Plain) linear least-squares regression minimizes

$$\mathcal{O}(\mathbf{w}) = \sum_{i=1}^{N} (\mathbf{w}^\top \mathbf{x}^i - y^i)^2.$$

- Regularized linear least-squares regression minimizes

$$\mathcal{O}(\mathbf{w}) = \sum_{i=1}^{N} (\mathbf{w}^\top \mathbf{x}^i - y^i)^2 + \lambda \|\mathbf{w}\|^2,$$

for a regularization (hyper-)parameter $\lambda \geq 0$.

Regularized linear regression is also called ridge regression.

Regularized linear least-squares regression

- (Plain) linear least-squares regression minimizes

$$\mathcal{O}(\mathbf{w}) = \sum_{i=1}^{N} (\mathbf{w}^\top \mathbf{x}^i - y^i)^2.$$

The minimizer $\mathbf{w}^*$ is obtained by solving a linear system

$$\mathbf{X}\mathbf{X}^\top \mathbf{w} = \mathbf{X}\mathbf{y}.$$

For high-dimensional problems ($N < n$), $\mathbf{X}\mathbf{X}^\top \mathbf{w}$ is rank deficient: Infinitely many solutions exist.

- Regularized linear least-squares regression minimizes

$$\mathcal{O}(\mathbf{w}) = \sum_{i=1}^{N} (\mathbf{w}^\top \mathbf{x}^i - y^i)^2 + \lambda \|\mathbf{w}\|^2,$$

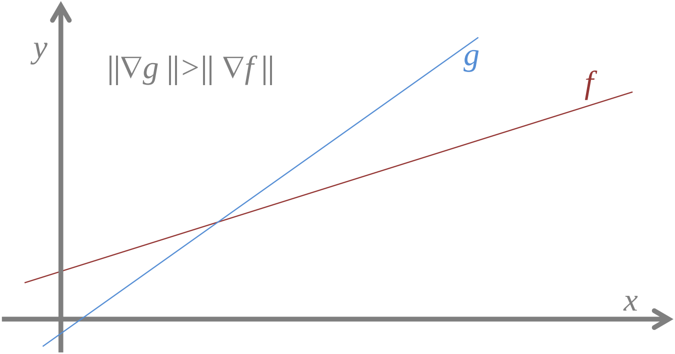
for a regularization (hyper-)parameter $\lambda \geq 0$.
The minimizer $\mathbf{w}^*$ is obtained by solving a linear system

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

For $\lambda > 0$, $\mathbf{X}\mathbf{X}^\top + \lambda \mathbf{I}$ is always full rank: A unique solution exists.

Regularized linear least-squares regression

$\nabla f = \|\mathbf{w}\|$ measures the slant of f: How much the output can change when the input changes.



Empirical error minimization for classification

We are given a training dataset (pairs of input and output)

$$D = \{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\} \subset \mathbb{R}^n \times \{-1, 1\}.$$

Our goal is to find a function

$$f : \mathbb{R}^n \to \{-1, 1\}$$

that minimizes the empirical classification error

$$\mathcal{O}(f) = \sum_{i=1}^{N} \mathbf{1}[f(\mathbf{x}^i) \neq y^i].$$

$$\mathbf{1}[A] = \begin{cases} 1, & \text{if } A \text{ is true} \\ 0, & \text{otherwise.} \end{cases}$$

A continuous approximation

$$f : \mathbb{R}^n \to \{-1, 1\}$$
$$\mathcal{O}(f) = \sum_{i=1}^{N} \mathbf{1}[f(\mathbf{x}^i) \neq y^i].$$

Minimizing $\mathcal{O}$ is a challenging discrete optimization problem.
Instead, we minimize a continuous approximation of $\mathcal{O}$:

$$\mathcal{O}'(f) = \sum_{i=1}^{N} l(f(\mathbf{x}^i), y^i) := \sum_{i=1}^{N} \max(0, 1 - f(\mathbf{x}^i) y^i)$$
$$f : \mathbb{R}^n \to \mathbb{R}.$$

$l(a, b) := \max(0, 1 - ab)$ is called the hinge loss.
Interpretation: we want f evaluation $f(\mathbf{x}^i)$
to have the same sign as the label y^i ($f(\mathbf{x}^i) y^i > 0$),
with a high-confidence (or margin) ($|f(\mathbf{x}^i)| > 1$).

Linear support vector machines

Training data: $D = \{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\} \subset \mathbb{R}^n \times \{-1, 1\}$.
A plain linear classifier minimizes:

$$\mathcal{O}(f) = \sum_{i=1}^{N} \max(0, 1 - f(\mathbf{x}^i) y^i),$$
$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}.$$

A regularized linear classifier (support vector machine) minimizes:

$$\mathcal{O}(f) = \sum_{i=1}^{N} \max(0, 1 - f(\mathbf{x}^i) y^i) + \lambda \|\mathbf{w}\|^2,$$
$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}.$$

Representer theorem [Sch]

A solution to

$$\mathcal{O}(f) = \sum_{i=1}^{N} \max(0, 1 - f(\mathbf{x}^i)y^i) + \lambda\|\mathbf{w}\|^2$$
$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$$

takes the form

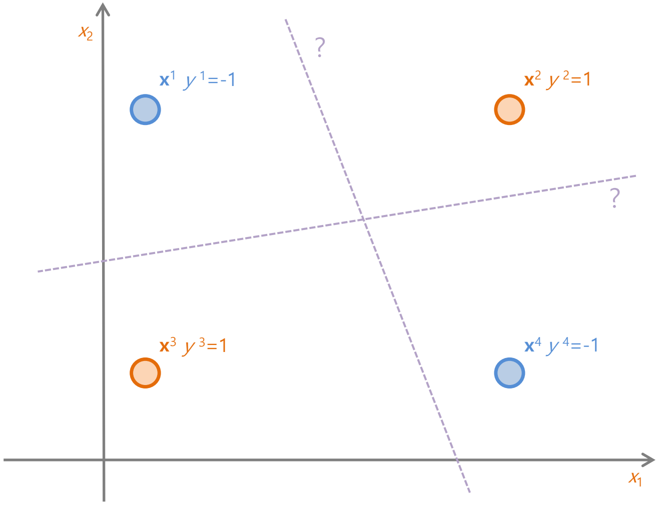
$$\mathbf{w}^* = \sum_{i=1}^{N} \alpha^i \mathbf{x}^i,$$

for $\{\alpha^i\}_{i=1}^N$.

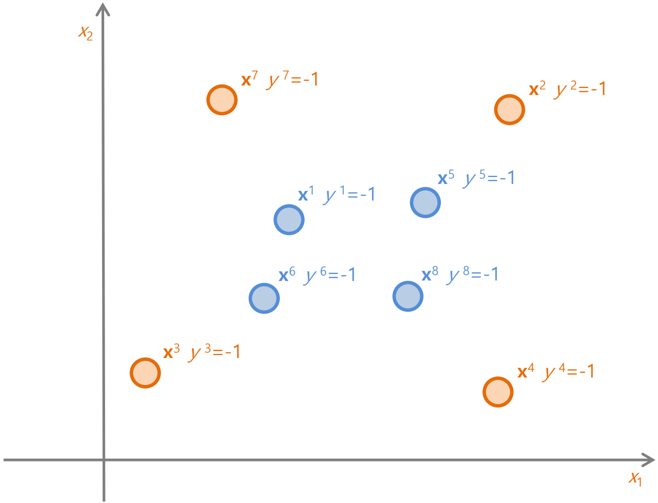
- The optimum $\mathbf{w}^*$ is expanded in training data points: $\{\alpha^i\}_{i=1}^N$ determines $\mathbf{w}^*$.
- $f(\mathbf{x}) = (\mathbf{w}^*)^\top \mathbf{x} = \sum_{i=1}^N \alpha^i (\mathbf{x}^i)^\top \mathbf{x} = \sum_{i=1}^N \alpha^i \langle \mathbf{x}^i, \mathbf{x}\rangle$, $\langle \mathbf{x}, \mathbf{x}'\rangle$: inner-product of $\mathbf{x}$ and $\mathbf{x}'$.

See [Sch] for a strong generalization of this result:
Representer theorem applies to any regularization energy with convex losses.

Often, linear classifiers are not enough...

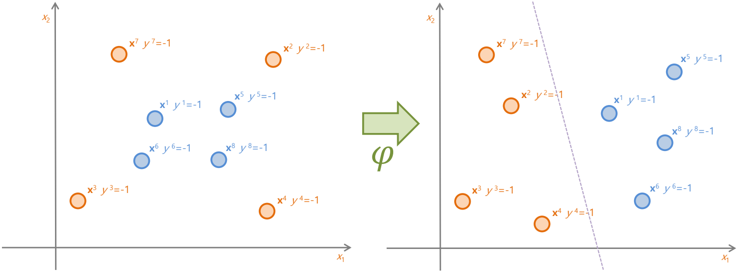


Often, linear classifiers are not enough...



Nonlinear classification

A simple approach to convert a linear classifier to nonlinear one:



Nonlinear classification

A simple approach to convert a linear classifier to nonlinear one:

1. Nonlinearly map data to a high-dimensional feature space $\mathcal{H}$: $\phi : \mathbb{R}^n \to \mathcal{H}$;
2. Build a linear classifier in $\mathcal{H}$;

- Why high-dimensional spaces?
- How high $\mathcal{H}$-dim. should be?

Nonlinear classification

Linear classifiers minimize:

$$\mathcal{O}(f) = \sum_{i=1}^{N} \max(0, 1 - f(\mathbf{x}^i)y^i) + \lambda\|\mathbf{w}\|^2$$
$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}, \mathbf{w} \in \mathbb{R}^n.$$

Nonlinear classifiers minimize:

$$\mathcal{O}(f) = \sum_{i=1}^{N} \max(0, 1 - f(\phi(\mathbf{x}^i))y^i) + \lambda\|\mathbf{w}\|^2$$
$$f(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x}), \mathbf{w} \in \mathcal{H}.$$

What if $\mathcal{H}$ has a very high-dimensionality, e.g. infinite?

Nonlinear classification

Nonlinear classifiers minimize:

$$\mathcal{O}(f) = \sum_{i=1}^{N} \max(0, 1 - f(\phi(\mathbf{x}^i))y^i) + \lambda\|\mathbf{w}\|^2$$

$$f(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x}), \mathbf{w} \in \mathcal{H}.$$

The representer theorem states that there are coefficients $\{\alpha^j\}_{j=1}^N$ s.t.

$$\mathbf{w}^* = \sum_{j=1}^{N} \alpha^j \phi(\mathbf{x}^j)$$

$$\Leftrightarrow f^*(\mathbf{x}) = (\mathbf{w}^*)^\top \phi(\mathbf{x}) = \sum_{j=1}^{N} \alpha^j \phi(\mathbf{x}^j)^\top \phi(\mathbf{x}) = \sum_{j=1}^{N} \alpha^j \langle \phi(\mathbf{x}^j), \phi(\mathbf{x}) \rangle.$$

$$\begin{aligned}
\mathcal{O}(f) &= \sum_{i=1}^{N} \max\left(0, 1 - \left(\sum_{j=1}^{N} \alpha^j \phi(\mathbf{x}^j)^\top \phi(\mathbf{x}^i)\right) y^i\right) + \lambda\|\mathbf{w}\|^2 \\
&= \sum_{i=1}^{N} \max\left(0, 1 - \left(\sum_{j=1}^{N} \alpha^j \langle \phi(\mathbf{x}^j), \phi(\mathbf{x}^i) \rangle\right) y^i\right) + \lambda \mathbf{w}^\top \mathbf{w} \\
&= \sum_{i=1}^{N} \max\left(0, 1 - \left(\sum_{j=1}^{N} \alpha^j \langle \phi(\mathbf{x}^j), \phi(\mathbf{x}^i) \rangle\right) y^i\right) + \lambda \sum_{i,j=1}^{N} \alpha^i \alpha^j \langle \phi(\mathbf{x}^i), \phi(\mathbf{x}^j) \rangle.
\end{aligned}$$

Positive definite kernels

A symmetric function $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is called positive definite if for any N, the matrix K formed by evaluating k on any N data points $\{\mathbf{x}^1, \ldots, \mathbf{x}^N\} \subset \mathbb{R}^n$ is positive definite:

$$[K]_{i,j} = k(\mathbf{x}^i, \mathbf{x}^j).$$

Kernel methods [Sch2]

For a positive definite kernel k, there is a (non-linear) feature map $\phi : \mathbb{R}^n \to \mathcal{H}_k$ that maps $\mathbf{x}$ to an element of the *space of functions* $\mathcal{H}_k$ such that:

$$\begin{aligned} \phi(\mathbf{x}) &= k(\mathbf{x}, \cdot) \\ \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle &= \langle k(\mathbf{x}, \cdot), k(\mathbf{x}', \cdot) \rangle = k(\mathbf{x}, \mathbf{x}'). \end{aligned} \tag{5}$$

- The feature map ϕ converts a data point $\mathbf{x} \in \mathbb{R}^n$ to a function $k(\mathbf{x}, \cdot)$ defined on $\mathbb{R}^n$.
- $\mathcal{H}_k$ (a space of functions) is called the reproducing kernel Hilbert space corresponding to kernel k.
- Eq. 5 is called the reproducing property.

Example kernels

- Gaussian kernel (or radial basis function (RBF) kernel):

$$k(\mathbf{x}, \mathbf{x}') = \frac{-\|\mathbf{x} - \mathbf{x}'\|^2}{\sigma_k^2}$$

  for a hyper-parameter σ_k^2:
  $\mathcal{H}$ is infinite dimensional; How do we know?
- Polynomial kernel: $k(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^\top \mathbf{x}' + c)^d$
  for hyper-parameters c, d.

How do we know that a function $k(\cdot, \cdot)$ is positive definite (p.d.)? k is p.d. if

- $k(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle$ for a map ϕ.
- $k(\mathbf{x}, \mathbf{x}') = k^1(\mathbf{x}, \mathbf{x}') + k^2(\mathbf{x}, \mathbf{x}')$ for p.d. functions k^1, k^2.
- $k(\mathbf{x}, \mathbf{x}') = k^1(\mathbf{x}, \mathbf{x}')k^2(\mathbf{x}, \mathbf{x}')$ for p.d. functions k^1, k^2.
- $k(\mathbf{x}, \mathbf{x}') = \exp(k^1(\mathbf{x}, \mathbf{x}'))$ for a p.d. function k^1.

Feature map ϕ is not uniquely defined

For $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^2$:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \mathbf{x}' = \begin{bmatrix} x_1' \\ x_2' \end{bmatrix},$$

two feature maps

$$\phi^1(\mathbf{x}) := [(x_1)^2, (x_2)^2, \sqrt{2} x_1 x_2]^\top$$

and

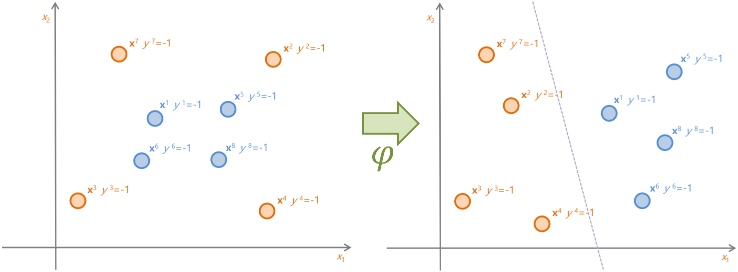
$$\phi^2(\mathbf{x}) := k(\mathbf{x}, \cdot)$$

with $k(\mathbf{a}, \mathbf{b}) = (\mathbf{a}^\top \mathbf{b})^2$ lead to the same inner-product:

$$\begin{aligned} \langle \phi^1(\mathbf{x}), \phi^1(\mathbf{x}') \rangle &= (x_1)^2 (x'_1)^2 + (x_2)^2 (x'_2)^2 + 2 x_1 x_2 x'_1 x'_2 \\ \langle k(\mathbf{x}, \cdot), k(\mathbf{x}', \cdot) \rangle &:= k(\mathbf{x}, \mathbf{x}') \\ &= (x_1)^2 (x'_1)^2 + (x_2)^2 (x'_2)^2 + 2 x_1 x_2 x'_1 x'_2. \end{aligned}$$

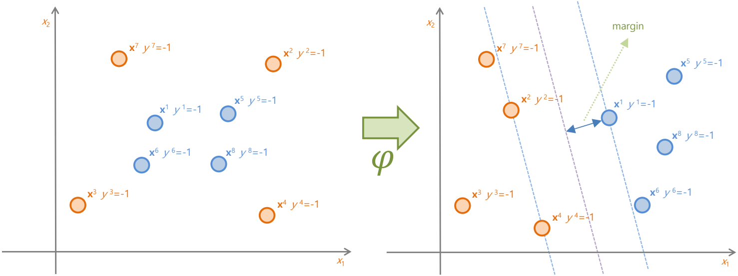
Support vector machines

find a linear classifier in a feature space:



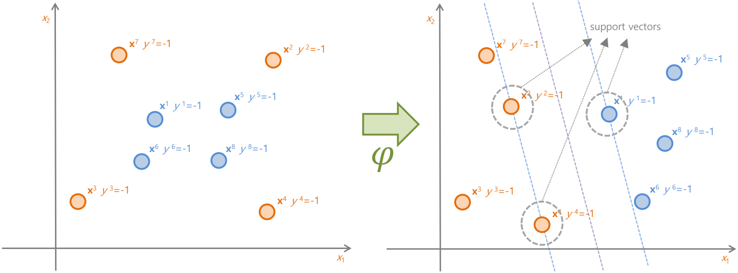
Support vector machines

maximizes margin:



Support vector machines

identifies support vectors:



Sparse expansion: our classifier f is represented as:
$f(\mathbf{x}) = \sum_{j=1}^{N} \alpha^j \langle \phi(\mathbf{x}^j), \phi(\mathbf{x}) \rangle$.
$\alpha^j = 0$ if $\mathbf{x}^j$ is not a support vector.

Support vector machines

Demo

Optimization methods

$$\{\alpha^j\}_{j=1}^N = \arg\min \sum_{i=1}^N \max\left(0, 1 - \left(\sum_{j=1}^N \alpha^j \langle \phi(\mathbf{x}^j), \phi(\mathbf{x}^i)\rangle\right) y^i\right)$$

$$+ \lambda \sum_{i,j=1}^N \alpha^i \alpha^j \langle \phi(\mathbf{x}^i), \phi(\mathbf{x}^j)\rangle \qquad (6)$$

$$= \arg\min \sum_{i=1}^N \eta^i + \lambda \sum_{i,j=1}^N \alpha^i \alpha^j \langle \phi(\mathbf{x}^i), \phi(\mathbf{x}^j)\rangle \qquad (7)$$

$$\text{s.t.} \left(\sum_{j=1}^N \alpha^j \langle \phi(\mathbf{x}^j), \phi(\mathbf{x}^i)\rangle\right) y^i \geq 1 - \eta^i,$$

$$\eta^i \geq 0, i = 1, \ldots, N.$$

Our objective is convex.

1. The original form (Eq. 6) is not differentiable. Solution can be found using sub-gradient descent.
2. The constrained optimization form (Eq. 7) can be eventually formulated as a quadratic optimization.

References

- **Pet** Petersen and Pedersen, *The Matrix Cookbook*
- **Kre** R. Kress, *Linear Integral Equations*, Springer (second edition)
- **Teu** Teukolsky, Vetterling, Flannery, *Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press (any edition) `http://www.nr.com/`
- **Sch** Schölkopf, Herbrich, and Smola, A Generalized representer theorem, *Proc. Computational Learning Theory*, 2001.
- **Sch2** Schölkopf and Smola, *Learning with Kernels*, MIT Press