Support Vector Machines

Support vector machines (SVMs) learn a hypothesis:

\[ h(x) = b + \sum_{i=1}^{m} y_i \alpha_i k(x, x_i) \]

\((x_1, y_1), \ldots, (x_m, y_m)\) are the training exs., \(y_i \in \{-1, 1\}\)

\(b\) is the bias weight.

\(\alpha_1, \ldots, \alpha_m\) are the Lagrange multipliers, \(\alpha_i \geq 0\),
a nonnegative weight for each training example.

\(k\) is a kernel function, e.g., we might choose the dot product,

\[ k(x, x') = x \cdot x' = \sum x_j x'_j \]

if \(\alpha_i > 0\), then \(x_i\) is a support vector.

Maximizing the Margin

The margin is the region \(-1 \leq h(x) \leq 1\).

The goal of learning is to maximize the width of the margin with positive exs. \(\geq 1\) and negative exs. \(\leq -1\).

A SVM is defined by its support vectors, the positive exs. \(\leq 1\) and negative exs. \(\geq -1\).

The dot product, linear, kernel function

\[ k(x, x') = x \cdot x' = \sum x_j x'_j \]

leads to a classifier similar to perceptrons with a margin.

Other kernel fns. lead to nonlinear decision boundaries.
Example SVM for Separable Examples

Example SVM for Nonseparable Examples
Example Gaussian Kernel SVM

Example Gaussian Kernel, Zoomed In
Hyperplane Classification

Consider the class of hyperplanes, i.e., dot product plus a bias:

\[(\mathbf{w} \cdot \mathbf{x}) + b = 0\]

For linearly separable examples, there a unique optimal hyperplane defined by maximizing the margin.

This can be expressed as:

\[
\max_{\mathbf{w},b} \min \left\{ \|\mathbf{x} - \mathbf{x}_i\| \mid (\mathbf{w} \cdot \mathbf{x}) + b = 0 \right\}
\]

Choose \(\mathbf{w}\) and \(b\) to maximize the minimum distance from an example to the hyperplane.

Hyperplane Example

Figure from Scholkopf and Smola, *Learning with Kernels*, MIT Press, 2002.
The optimal hyperplane can be found by solving:

\[
\begin{align*}
\text{minimize} & \quad \|w\|^2/2 \\
\text{s.t.} & \quad y_i((w \cdot x_i) + b) \geq 1, \ i \in \{1, \ldots, m\}
\end{align*}
\]

That is, we require the smallest weights such that positive examples \( \geq 1 \) and negative examples \( \leq -1 \).

The smallest weights correspond to the maximum margin. Note that:

\[1 = w \cdot \frac{w}{\|w\|^2}\]

so the width of the margin is equal to:

\[2 \frac{\|w\|}{\|w\|^2} = \frac{2}{\|w\|}\]

---

**Math Tricks**

Convert to a Lagrangian:

\[\|w\|^2/2 - \sum_{i=1}^{m} \alpha_i (y_i((w \cdot x_i) + b) - 1)\]

The derivatives are zero when:

\[w = \sum_{i=1}^{m} \alpha_i y_i x_i \quad \text{and} \quad 0 = \sum_{i=1}^{m} \alpha_i y_i\]

Substituting for \( w \) in the Lagrangian leads to the objective function:

\[\sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j)\]

which we want to maximize subject to

\[\alpha_i \geq 0 \quad \text{and} \quad \sum_{i=1}^{m} \alpha_i y_i = 0\]
Substituting for \( \mathbf{w} \) in the hypothesis:

\[
h(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{x}) + b
\]

leads to:

\[
h(\mathbf{x}) = b + \sum_{i=1}^{m} y_i \alpha_i (\mathbf{x} \cdot \mathbf{x}_i)
\]

In this case (dot product kernel, linearly separable examples), \( \alpha_i > 0 \) implies \( y_i ((\mathbf{w} \cdot \mathbf{x}_i) + b) = 1 \), i.e., support vectors will be on the margin boundary.

Kernel Tricks

The kernel trick is to substitute other functions \( k(\mathbf{x}, \mathbf{x}') \) in place of \( (\mathbf{x} \cdot \mathbf{x}') \). The most popular are

- **Polynomial**: \( (\mathbf{x} \cdot \mathbf{x}')^d \), where \( d \in \{2, 3, \ldots\} \)
  
  \[a(\mathbf{x} \cdot \mathbf{x}') + c]^d, \text{ where } d \in \{2, 3, \ldots\}, a, c > 0 \]

- **Gaussian**: \( \exp\{-\|\mathbf{x} - \mathbf{x}'\|^2/(2\sigma^2)\} \), where \( \sigma > 0 \).

The “trick” of a kernel function is efficiently computing the dot product of a large number of basis functions.

\[
k(\mathbf{x}, \mathbf{x}') = (\Phi(\mathbf{x}) \cdot \Phi(\mathbf{x}'))
\]

where \( \Phi(\mathbf{x}) = (\Phi_1(\mathbf{x}), \Phi_2(\mathbf{x}), \ldots) \).
Example Kernel Function

Let \( k(\mathbf{x}, \mathbf{x}') = ((\mathbf{x} \cdot \mathbf{x}') + 1)^2 \). Then in two dimensions,
\[
((\mathbf{u} \cdot \mathbf{v}) + 1)^2 \\
= ((u_1, u_2) \cdot (v_1, v_2)) + 1)^2 \\
= (u_1v_1 + u_2v_2 + 1)^2 \\
= u_1^2v_1^2 + u_2^2v_2^2 + 1 + 2u_1u_2v_1v_2 + 2u_1v_1 + 2u_2v_2 \\
= (u_1^2, u_2^2, 1, \sqrt{2}u_1u_2, \sqrt{2}u_1, \sqrt{2}u_2) \\
\cdot (v_1^2, v_2^2, 1, \sqrt{2}v_1v_2, \sqrt{2}v_1, \sqrt{2}v_2)
\]

The kernel function obtains the same result as the dot product of the basis functions without explicitly computing all the basis functions. The Gaussian kernel corresponds to an infinite number of basis functions!

---

Nonlinear Classification

Using a kernel function, the hypothesis becomes:

\[
h(\mathbf{x}) = b + \sum_{i=1}^{m} y_i \alpha_i k(\mathbf{x}, \mathbf{x}_i)
\]

The Lagrangian conversion results in the problem:

Find the \( \alpha_i \) weights that solve:

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

subject to \( \alpha_i \geq 0 \) and \( \sum_{i=1}^{m} \alpha_i y_i = 0 \)

The support vectors are \( \{\mathbf{x}_i \mid \alpha_i > 0\} \).
They satisfy \( y_i h(\mathbf{x}_i) = 1 \).
Soft Margin Classification

It might not be possible or desirable to satisfy:

\[ y_i ((w \cdot x_i) + b) \geq 1 \]

To allow violations, an error term can be added:

\[ y_i ((w \cdot x_i) + b) \geq 1 - \xi_i \]

where \( \xi_i \geq 0 \) and the problem is to:

\[
\text{minimize } ||w||^2 / 2 + C \sum_{i=1}^{m} \xi_i
\]

where \( C \) is chosen by the user.

Note that \( \sum_{i=1}^{m} \xi_i \geq \) the number of training mistakes.

In the conversion, replace \( \alpha_i \geq 0 \) with \( 0 \leq \alpha_i \leq C \).
**ν-Parameterization**

Another type of soft margin classifier satisfies:

\[ y_i \left( (\mathbf{w} \cdot \mathbf{x}_i) + b \right) \geq \rho - \xi_i \]

where \( \rho > 0 \) is a free parameter, and solves:

\[
\text{minimize } \frac{\|\mathbf{w}\|^2}{2} - \rho + \frac{1}{\nu m} \sum_{i=1}^{m} \xi_i
\]

where \( \nu \) is chosen by the user.

In the conversion, \( \alpha_i \geq 0 \) is replaced with:

\[ 0 \leq \alpha_i \leq \frac{1}{\nu m} \quad \text{and} \quad \sum_{i=1}^{m} \alpha_i = 1 \]

\( \nu \in (0, 1) \) means at least \( m \nu \) support vectors.
$C = 1$, Gaussian Kernel, $\sigma^2 = 0.5$

$C = 10$, Gaussian Kernel, $\sigma^2 = 0.5$
$C = 100$, Gaussian Kernel, $\sigma^2 = 0.5$

$\nu = 0.3$, Gaussian Kernel, $\sigma^2 = 0.5$
\[ \nu = 0.2, \text{ Gaussian Kernel, } \sigma^2 = 0.5 \]

\[ \nu = 0.1, \text{ Gaussian Kernel, } \sigma^2 = 0.5 \]
Support Vector Regression

SV classification uses \( y \in \{-1, 1\} \), while regression tries to predict \( y \in \mathbb{R} \).

SV regression uses \( \epsilon \)-insensitive loss:

\[
|y - z|_\epsilon = \begin{cases} 
0 & \text{if } |y - z| \leq \epsilon \\
|y - z| - \epsilon & \text{otherwise}
\end{cases}
\]

or equivalently:

\[
|y - z|_\epsilon = \max(0, |y - z| - \epsilon)
\]

To obtain a hypothesis \( h(x) = b + w \cdot x \)

\[
\text{minimize } ||w||^2/2 + C \sum_{i=1}^{m} |y - h(x_i)|_\epsilon
\]

One can think of SV regression as fitting a tube of radius \( \epsilon \) to the data.

Figure from Scholkopf and Smola, Learning with Kernels, MIT Press, 2002.
SV Regression Problem

Applying the usual math tricks results in:

\[ h(x) = b + \sum_{i=1}^{m} \alpha_i k(x, x_i) \]

where the \( \alpha_i \) weights are found by solving:

\[
\begin{align*}
\text{maximize} & \quad -\epsilon \sum_{i=1}^{m} |\alpha_i| \\
& \quad + \sum_{i=1}^{m} \alpha_i y_i \\
& \quad - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j k(x_i, x_j) \\
\text{subject to} & \quad -C \leq \alpha_i \leq C \quad \text{and} \quad \sum_{i=1}^{m} \alpha_i = 0
\end{align*}
\]