Linear classifiers
which line is better?
Pick the one with the largest margin!

Data:
\[(x_1^{(1)}, \ldots, x_1^{(m)}), y_1 \]
\[\vdots\]
\[(x_n^{(1)}, \ldots, x_n^{(m)}), y_n\]

\[w \cdot x + b > 0\]
\[w \cdot x + b < 0\]

```
Margin
```

```
\text{"confidence"} = (w^T x_j + b) y_j
```
Classification rule:

Classify as.. \(+1\) if \(w \cdot x + b \geq 1\)

\(-1\) if \(w \cdot x + b \leq -1\)

Universe explodes if 
\(-1 < w \cdot x + b < 1\)

How large is the margin of this classifier?

**Goal:** Find the maximum margin classifier
Computing the margin width

Let \( x^+ \) and \( x^- \) be such that

- \( w \cdot x^+ + b = +1 \)
- \( w \cdot x^- + b = -1 \)
- \( x^+ = x^- + \lambda w \)
- \( |x^+ - x^-| = M = ? \) (Margin)

Maximize \( M \equiv \text{minimize } w \cdot w \)
The Primal Hard SVM

- Given $D = \{(x_i, y_i), i = 1, \ldots, n\}$ training data set.
- Assume that $D$ is **linearly separable**.

\[
\hat{w} = \arg \min_{w \in \mathbb{R}^m} \frac{1}{2}||w||^2
\]

subject to $y_i \langle x_i, w \rangle \geq 1, \forall i = 1, \ldots, n$

**Prediction:** $f_{\hat{w}}(x) = \text{sign}(\langle \hat{w}, x \rangle)$

This is a QP problem (m-dimensional) (Quadratic cost function, linear constraints)
Quadratic Programming

Find
\[ \arg \min_{\omega \in \mathbb{R}^n} \omega^T H \omega + \omega^T q + e \]

Subject to
\[ A \omega \leq b \]
\[ A \in \mathbb{R}^{n\times m}, \quad \omega \in \mathbb{R}^m, \quad b \in \mathbb{R}^n \]

and to
\[ C \omega = d \]
\[ C \in \mathbb{R}^{s\times m}, \quad d \in \mathbb{R}^s \]

Efficient Algorithms exist for QP. They often solve the dual problem instead of the primal.
Constrained Optimization

$$\min_x x^2$$

s.t. \quad x \geq b

$$\min_x x^2$$

s.t. \quad x \geq -1

$$\min_x x^2$$

s.t. \quad x \geq 1

\(x^* = 0\)

\(x^* = 0\)

\(x^* = 1\)
Lagrange Multiplier

Moving the constraint to objective function

\[ L(x, \alpha) = x^2 - \alpha(x - b) \]

\[ \text{s.t. } \alpha \geq 0 \]

Solve:

\[ \min_x \max_{\alpha} \ L(x, \alpha) \]

\[ \text{s.t. } \alpha \geq 0 \]

Constraint is active when \( \alpha > 0 \)
Lagrange Multiplier – Dual Variables

Solving:

\[ L(x, \alpha) = \min_x \max_\alpha \ x^2 - \alpha(x - b) \]

s.t. \( \alpha \geq 0 \)

\[ \frac{\partial L}{\partial x} = 0 \Rightarrow x^* = \frac{\alpha}{2} \]

\[ \frac{\partial L}{\partial \alpha} = 0 \Rightarrow \alpha^* = \max(2b, 0) \]

When \( \alpha > 0 \), constraint is tight

\[ 2x^* - \alpha = 0 \quad \Rightarrow \quad x^* = \frac{\alpha}{2} \]

\[ L(x^*, \alpha) = \frac{\alpha^2}{4} - \alpha \left( \frac{\alpha}{2} - b \right) \]

\[ \frac{\partial L(x^*, \alpha)}{\partial \alpha} = -\frac{\alpha^2}{4} + b \alpha \]

\( \alpha^* = 2b \)
Primal problem:

\[ \hat{w} = \arg \min_{w \in \mathbb{R}^m} \frac{1}{2} \|w\|^2 \]

subject to \( y_i \langle x_i, w \rangle \geq 1, \forall i = 1, \ldots, n \)

Lagrange function:

\[ \alpha = (\alpha_1, \ldots, \alpha_n)^T \geq 0 \text{ Lagrange multipliers} \]

\[ L(w, \alpha) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{n} \alpha_i \left( y_i \langle x_i, w \rangle - 1 \right) \]
The Lagrange Problem

\[ L(w, \alpha) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{n} \alpha_i (y_i \langle x_i, w \rangle - 1) \]

The Lagrange problem:

\[ (\hat{w}, \hat{\alpha}) = \arg \min_{w \in \mathbb{R}^m} \max_{0 \leq \alpha \in \mathbb{R}^n} L(w, \alpha) \]

\[ 0 = \frac{\partial L(w, \alpha)}{\partial w} \bigg|_{w=\hat{w}} = \hat{w} - \sum_{i=1}^{n} \alpha_i y_i x_i \]

\[ \Rightarrow \hat{w} = \sum_{i=1}^{n} \alpha_i y_i x_i \]
The Dual Problem

\[ L(w, \alpha) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{n} \alpha_i \left( y_i \langle x_i, w \rangle - 1 \right) \]

\[ \Rightarrow \widehat{w} = \sum_{i=1}^{n} \alpha_i y_i x_i \]

\[ \Rightarrow L(\widehat{w}, \alpha) = \frac{1}{2} \|\widehat{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left( y_i \langle x_i, \widehat{w} \rangle - 1 \right) \]

\[ = \frac{1}{2} \left\| \sum_{i=1}^{n} \alpha_i y_i x_i \right\|^2 + \alpha^T 1_n - \sum_{i=1}^{n} \alpha_i y_i \langle x_i, \sum_{j=1}^{n} \alpha_j y_j x_j \rangle \]

\[ = \alpha^T Y G Y \alpha \]

\[ = \alpha^T 1_n - \frac{1}{2} \alpha^T Y G Y \alpha \]

\[ Y \doteq \text{diag}(y_1, \ldots, y_n), \ y_i \in \{-1, 1\}^n \]

\[ G \in \mathbb{R}^{n \times n} \doteq \{G_{ij}\}_{i,j}^{n,n}, \ \text{where} \ G_{ij} \doteq \langle x_i, x_j \rangle \ \text{Gram matrix.} \]
The Dual Hard SVM

\[ Y \doteq \text{diag}(y_1, \ldots, y_n), \ y_i \in \{-1, 1\}^n \]

\[ G \in \mathbb{R}^{n \times n} \doteq \{G_{ij}\}_{i,j}^{n,n}, \text{ where } G_{ij} \doteq \langle x_i, x_j \rangle \text{ Gram matrix.} \]

\[ \hat{\alpha} = \arg \max_{\alpha \in \mathbb{R}^n} \alpha^T 1_n - \frac{1}{2} \alpha^T YGY \alpha \]

subject to \( \alpha_i \geq 0, \ \forall i = 1, \ldots, n \)

Quadratic Programming (n-dimensional)

**Lemma**

\[ \hat{w} = \sum_{i=1}^{n} \hat{\alpha}_i y_i x_i \]

**Prediction:**

\[ f_\hat{w}(x) = \text{sign}(\langle \hat{w}, x \rangle) = \text{sign}(\sum_{i=1}^{n} \hat{\alpha}_i y_i \langle x_i, x \rangle) \]

\[ \frac{k(x_i, x)}{k(x_i, x)} \]
The Problem with Hard SVM

It assumes samples are linearly separable...

What can we do if data is not linearly separable???
If the data set is not linearly separable, then adding new features (mapping the data to a larger feature space) the data might become linearly separable.
Hard 1-dimensional Dataset

Make up a new feature!
Sort of...
... computed from original feature(s)

\[ z_k = (x_k, x_k^2) \]

Separable! MAGIC!

Now drop this “augmented” data into our linear SVM.
Feature mapping

• *n* general! points in an *n*-1 dimensional space is always linearly separable by a hyperspace! ⇒ it is good to map the data to high dimensional spaces

• Having *n* training data, is it always enough to map the data into a feature space with dimension *n*-1?
  • *Nope... We have to think about the test data as well!* 
  
  *Even if we don’t know how many test data we have and what they are...* 

  • *We might want to map our data to a huge (∞) dimensional feature space* 
  
  • *Overfitting? Generalization error?... We don’t care now...*
How to do feature mapping?

Let us have $n$ training objects: $\vec{x}_i = [\vec{x}_{i,1}, \vec{x}_{i,2}] \in \mathbb{R}^2$, $i = 1, \ldots, n$

The possible test objects are denoted by $\vec{x} = [\vec{x}_1, \vec{x}_2] \in \mathbb{R}^2$

Let $\phi(\vec{x}) \doteq [\sin(\vec{x}_2), \exp(\vec{x}_2 + \vec{x}_1), \vec{x}_1, \vec{x}_2^{\tan(\vec{x}_1)}, \ldots]$
The Problem with Hard SVM

It assumes samples are linearly separable...

Solutions:

1. Use feature transformation to a larger space
   ⇒ each training samples are linearly separable in the feature space
   ⇒ Hard SVM can be applied 😊
   ⇒ overfitting... 😞

2. **Soft margin** SVM instead of Hard SVM
   - Slack variables... We will discuss them now
The Hard SVM problem can be rewritten:

\[ \hat{w}_{hard} = \arg \min_{w \in \mathbb{R}^m} \frac{1}{2} ||w||^2 \]

subject to \( y_i \langle x_i, w \rangle \geq 1, \ \forall i = 1, \ldots, n \)

\[ \hat{w}_{hard} = \arg \min_{w \in \mathbb{R}^m} \sum_{i=1}^{n} l_{0-\infty}(\langle x_i, w \rangle, y_i) + \frac{1}{2} ||w||^2 \]

where

\[ l_{0-\infty}(a, b) = \begin{cases} \infty : ab < 1 \\ 0 : ab \geq 1 \end{cases} \]

Misclassification, or inside the margin

Correct classification and outside of the margin
From Hard to Soft constraints

Instead of using hard constraints (points are linearly separable)

\[
\hat{w}_{\text{hard}} = \arg \min_{w \in \mathbb{R}^m} \sum_{i=1}^{n} l_{0-\infty}(\langle x_i, w \rangle, y_i) + \frac{1}{2} \| w \|^2
\]

We can try solve the soft version of it:. Introduce a \( \lambda \) parameter!

(Your loss is only 1 instead of \( \infty \) if you misclassify an instance)

\[
\hat{w}_{\text{soft}} = \arg \min_{w \in \mathbb{R}^m} \sum_{i=1}^{n} l_{0-1}(\langle x_i, w \rangle, y_i) + \frac{\lambda}{2} \| w \|^2
\]

where

\[
l_{0-1}(y, f(x)) = \begin{cases} 1 : yf(x) < 0 & \text{Misclassification} \\ 0 : yf(x) > 0 & \text{Correct classification} \end{cases}
\]
Problems with $l_{0-1}$ loss

$$\hat{w}_{soft} = \arg \min_{w \in \mathbb{R}^m} \sum_{i=1}^{n} l_{0-1}(\langle x_i, w \rangle, y_i) + \frac{\lambda}{2} \| w \|^2$$

$$l_{0-1}(y, f(x)) = \begin{cases} 1 : yf(x) < 0 \\ 0 : yf(x) > 0 \end{cases}$$

It is not convex in $yf(x) \Rightarrow$ It is not convex in $w$, either...

... and we only like convex functions...

Let us approximate it with convex functions!
Approximation of the Heaviside step function

\[ l_{0-1}(y, \langle w, x \rangle) = \begin{cases} 
1 & : y \langle w, x \rangle < 0 \\
0 & : y \langle w, x \rangle > 0 
\end{cases} \]

\[ l_{lin}(y, f(x)) \]

\[ l_{quad}(y, f(x)) \]

Picture is taken from R. Herbrich
Approximations of $l_{0-1}$ loss

- Piecewise linear approximations (hinge loss, $l_{\text{lin}}$)

$$l_{\text{lin}}(f(x), y) = \max\{1 - yf(x), 0\}$$

[We want $yf(x) > 1$]

- Quadratic approximation ($l_{\text{quad}}$)

$$l_{\text{quad}}(f(x), y) = \max\{1 - yf(x), 0\}^2$$
The hinge loss approximation of $l_{0-1}$

\[
\hat{w} = \arg \min_{w \in \mathbb{R}^m} \sum_{i=1}^{n} l_{lin}(\langle x_i, w \rangle, y_i) + \frac{\lambda}{2} ||w||^2 \quad \xi_i \geq 0
\]

Where,

\[
\xi_i \doteq l_{lin}(f(x_i), y_i) = \max\{1 - y_i f(x_i), 0\}
\]

\[
\geq 1 - y_i \langle w, x_i \rangle \geq l_{0-1}(y_i, f(x_i))
\]
The Slack Variables

\[ M = \frac{2}{\sqrt{\mathbf{w} \cdot \mathbf{w}}} \]

\[ \xi_i \equiv l_{lin}(f(x_i), y_i) = \max\{1 - y_i f(x_i), 0\} \]
\[ = \max\{1 - y_i (\mathbf{w}^T \mathbf{x}_i), 0\} \]
The Primal Soft SVM problem

\[ \hat{w}_{soft} = \arg \min_{w \in \mathbb{R}^m} \sum_{i=1}^{n} l_{\text{lin}}(\langle x_i, w \rangle, y_i) + \frac{\lambda}{2} \|w\|^2 \quad \xi_i \geq 0 \]

where

\[ \xi_i = l_{\text{lin}}(f(x_i), y_i) = \max\{1 - y_i(w^T x_i), 0\} \]

Equivalently,

\[ \hat{w}_{soft} = \arg \min_{w \in \mathbb{R}^m, \xi \in \mathbb{R}^n} \sum_{i=1}^{n} \xi_i + \frac{\lambda}{2} \|w\|^2 \]

subject to \( y_i \langle x_i, w \rangle \geq 1 - \xi_i, \forall i = 1, \ldots, n \)

\( \xi_i \geq 0, \forall i = 1, \ldots, n \)

\( \xi_i : \text{ Slack variables} \)
The Primal Soft SVM problem

\[ \hat{w}_{soft} = \arg \min_{w \in \mathbb{R}^m, \xi \in \mathbb{R}^n} \sum_{i=1}^{n} \xi_i + \frac{\lambda}{2} \|w\|^2 \]

subject to \( y_i \langle x_i, w \rangle \geq 1 - \xi_i, \forall i = 1, \ldots, n \)

Equivalently,
\[ \xi_i \geq 0, \forall i = 1, \ldots, n \]

We can use this form, too... where \( C = \frac{1}{\lambda} \)

\[ \hat{w}_{soft} = \arg \min_{w \in \mathbb{R}^m, \xi \in \mathbb{R}^n} C \sum_{i=1}^{n} \xi_i + \frac{1}{2} \|w\|^2 \]

subject to \( y_i \langle x_i, w \rangle \geq 1 - \xi_i, \forall i = 1, \ldots, n \)

\[ \xi_i \geq 0, \forall i = 1, \ldots, n \]

What is the dual form of primal soft SVM?
The Dual Soft SVM (using hinge loss)

\[ \hat{w}_{soft} = \arg \min_{w \in \mathbb{R}^m, \xi \in \mathbb{R}^n} C \sum_{i=1}^{n} \xi_i + \frac{1}{2} \|w\|^2 \]

subject to \( y_i \langle x_i, w \rangle \geq 1 - \xi_i, \ \forall i = 1, \ldots, n \)

\( \xi_i \geq 0, \ \forall i = 1, \ldots, n \)

\( \alpha = (\alpha_1, \ldots, \alpha_n)^T \geq 0 \) Lagrange multipliers

\( \beta = (\beta_1, \ldots, \beta_n)^T \geq 0 \) Lagrange multipliers

\[ (\hat{w}, \hat{\xi}, \hat{\alpha}, \hat{\beta}) = \arg \min_{w \in \mathbb{R}^m} \max_{0 \leq \alpha} \max_{0 \leq \xi \in \mathbb{R}^n} \max_{0 \leq \beta} L(w, \xi, \alpha, \beta) \]

where

\[ L(w, \xi, \alpha, \beta) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i (y_i \langle x_i, w \rangle - 1 + \xi_i) - \sum_{i=1}^{n} \beta_i \xi_i \]
The Dual Soft SVM (using hinge loss)

\[ L(w, \xi, \alpha, \beta) = \frac{1}{2} \|w\|^2 + C \xi^T 1_n - \sum_{i=1}^{n} \alpha_i y_i \langle x_i, w \rangle + \alpha^T 1_n - \xi^T (\alpha + \beta) \]

\[
\begin{align*}
0 &= \frac{\partial L(w, \xi, \alpha, \beta)}{\partial w} \bigg|_{w=\hat{w}} = \hat{w} - \sum_{i=1}^{n} \alpha_i y_i x_i \Rightarrow \\
\hat{w} &= \sum_{i=1}^{n} \alpha_i y_i x_i
\end{align*}
\]

\[
0 = \frac{\partial L(w, \xi, \alpha, \beta)}{\partial \xi} \bigg|_{\xi=\hat{\xi}} = C 1_n - \alpha - \beta \Rightarrow \beta = C 1_n - \alpha \geq 0
\]

\[
\Rightarrow 0 \leq \alpha \leq C
\]

\[
\Rightarrow (\hat{\alpha}, \hat{\beta}) = \arg \max_{0 \leq \alpha \leq C, \ 0 \leq \beta} L(\hat{w}, \hat{\xi}, \alpha, \beta)
\]

\[
\Rightarrow \hat{\alpha} = \arg \max_{0 \leq \alpha \leq C} \alpha^T 1_n - \frac{1}{2} \alpha^T Y G Y \alpha
\]
The Dual Soft SVM (using hinge loss)

\[ Y \doteq \text{diag}(y_1, \ldots, y_n) \in \{-1, 1\}^n \]

\[ G \in \mathbb{R}^{n \times n} \doteq \{G_{ij}\}_{i,j}^{n,n}, \text{ where } G_{ij} \doteq \langle x_i, x_j \rangle, \text{ Gram matrix.} \]

\[ \hat{\alpha} = \arg \max_{\alpha \in \mathbb{R}^n} \alpha^T 1_n - \frac{1}{2} \alpha^T YGY \alpha \]

subject to \( 0 \leq \alpha_i \leq C \)

where \( C = \frac{1}{\lambda} \)

If \( \lambda \to 0 \Rightarrow \) soft-SVM \( \to \) hard-SVM

This is the same as the dual hard-SVM problem, but now we have the additional \( 0 \leq \alpha_i \leq C \) constraints.
SVM classification in the dual space

Solve the dual problem

\[ \hat{\alpha} = \arg \max_{\alpha \in \mathbb{R}^n} \alpha^T 1_n - \frac{1}{2} \alpha^T YGY \alpha \]

subject to \( 0 \leq \alpha_i \leq C \)

where \( C = \frac{1}{\lambda} \). Let \( \hat{w} = \sum_{i=1}^{n} \hat{\alpha}_i y_i x_i \).

On test data \( x \): \( f_{\hat{w}}(x) = \langle \hat{w}, x \rangle = \sum_{i=1}^{n} \hat{\alpha}_i y_i \frac{\langle x_i, x \rangle}{k(x_i, x)} \)
\[ \alpha = (\alpha_1, \ldots, \alpha_n)^T \geq 0 \] 

Lagrangian multipliers

\[ L(w, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i (y_i \langle x_i, w \rangle - 1) \]

KKT conditions

Complementary slackness condition

\[ d_i > 0 \Rightarrow y_i \langle x_i, w \rangle = 1 \]

\[ \langle x_i, w \rangle = +1 \]

\[ \langle x_i, w \rangle = -1 \]

Either \( d_i = 0 \) or \( (y_i \langle x_i, w \rangle - 1) = 0 \) and \( d_i > 0 \)

\( x_i \) is on the margin lines

Support vectors
Dual SVM Interpretation: Sparsity

\[ w = \sum_{j} \alpha_j y_j x_j \]

Only few \( \alpha_j \)s can be non-zero: where constraint is tight

\[ (\langle w, x_j \rangle + b)y_j = 1 \]

Support vectors – training points \( j \) whose \( \alpha_j \)s are non-zero
Support Vectors

\[ w \cdot x + b > 0 \]

\[ w \cdot x + b < 0 \]

Linear hyperplane defined by “support vectors”

Moving other points a little doesn’t effect the decision boundary

only need to store the support vectors to predict labels of new points
Support vectors in Soft SVM

\[ \hat{w}_{soft} = \arg \min_{w \in \mathbb{R}^m, \xi \in \mathbb{R}^n} C \sum_{i=1}^{n} \xi_i + \frac{1}{2} ||w||^2 \]

s.t. \( y_i \langle x_i, w \rangle \geq 1 - \xi_i, \forall i = 1, \ldots, n \)

\( \xi_i \geq 0, \forall i = 1, \ldots, n \)
Support vectors in Soft SVM

\[
\hat{w}_{\text{soft}} = \arg \min_{w \in \mathbb{R}^m, \xi \in \mathbb{R}^n} C \sum_{i=1}^{n} \xi_i + \frac{1}{2} \|w\|^2
\]

s.t. \( y_i \langle x_i, w \rangle \geq 1 - \xi_i, \ \forall i = 1, \ldots, n \)

\( \xi_i \geq 0, \ \forall i = 1, \ldots, n \)

- **Margin support vectors**
  \( \xi_i \geq 0 \)
  \( y_i \langle x_i, w \rangle = 1 \)

- **Nonmargin support vectors**
  \( \xi_i > 0 \)
SVM classification in the dual space

“Without b”

\[ \hat{\alpha} = \arg \max_{\alpha \in \mathbb{R}^m} \alpha^T 1_m - \frac{1}{2} \alpha^T Y G Y \alpha \]

subject to \( 0 \leq \alpha_i \leq C \)

“With b”

\[ L(w, b, \alpha) = \frac{1}{2} ||w||^2 - \sum_{i=1}^{n} \alpha_i (y_i (x_i \cdot w + b) - 1) \]

\[ \hat{\alpha} = \arg \max_{\alpha \in \mathbb{R}^n} \alpha^T 1_n - \frac{1}{2} \alpha^T Y G Y \alpha \]

subject to \( 0 \leq \alpha_i \leq C \)

\[ \sum_{i} \alpha_i y_i = 0 \]
SVM with Linear Programs

**QP:**

\[
\min_{w \in \mathbb{R}^m, \xi \in \mathbb{R}^n} C \sum_{i=1}^{n} \xi_i + \frac{1}{2} \|w\|^2
\]

subject to \( y_i \langle x_i, w \rangle \geq 1 - \xi_i, \ \forall i = 1, \ldots, n \)

\[ \xi_i \geq 0, \ \forall i = 1, \ldots, n \]

**LP:**

\[
\min_{\alpha \in \mathbb{R}^n, \xi \in \mathbb{R}^n} C \sum_{i=1}^{n} \xi_i + \sum_{i=1}^{n} \alpha_i
\]

subject to \( y_i \langle x_i, w \rangle \geq 1 - \xi_i, \ \forall i = 1, \ldots, n \)

\[ \xi_i \geq 0, \ \forall i = 1, \ldots, n \]

\[ \alpha_i \geq 0, \ \forall i = 1, \ldots, n \]

\[ w = \sum_{j=1}^{n} \alpha_j y_j x_j \]
SVM for Regression

\[ \text{Loss}(y, \mathbf{w}^T \mathbf{x}) = (y - \mathbf{w}^T \mathbf{x})^2 \]

\[ \begin{cases} 0 & \text{if } |y - \mathbf{w}^T \mathbf{x}| \leq \varepsilon \\ 1^\circ - \mathbf{w}^T \mathbf{x} & \text{otherwise} \end{cases} \]
Ridge Regression

Linear regression:  \( f(x) = \langle \mathbf{w}, \phi(x) \rangle \)

**Primal:**

\[
\hat{\mathbf{w}} = \arg\min_{\mathbf{w} \in \mathcal{K}} \sum_{i=1}^{n} \xi_i^2 \\
\text{subject to } y_i - \langle \phi(x_i), \mathbf{w} \rangle = \xi_i, \; \forall i = 1, \ldots, n \\
\text{and } \|\mathbf{w}\| \leq B
\]

\[
L(\mathbf{w}, \xi, \alpha, \beta) = \sum_{i=1}^{n} \xi_i^2 + \sum_{i=1}^{n} \alpha_i (y_i - \langle \phi(x_i), \mathbf{w} \rangle - \xi_i) + \frac{\lambda(\|\mathbf{w}\|^2 - B^2)}{\lambda > 0}
\]

**Dual for a given** \( \lambda \):  ...after some calculations...

\[
\hat{\alpha} = \arg\max_{\alpha \in \mathbb{R}^n} \lambda \sum_{i=1}^{n} \alpha_i^2 - 2 \sum_{i=1}^{n} \alpha_i y_i + \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(x_i, x_j)
\]

This can be solved in closed form:
Kernel Ridge Regression Algorithm

Given \( D = \{(x_i, y_i), i = 1, \ldots, n\} \) training data set. \( k(\cdot, \cdot) \) kernel, \( \lambda > 0 \) parameter. \( y = (y_1, \ldots, y_n)^T \in \{-1, 1\}^n \)

- \( G \in \mathbb{R}^{n \times n} \triangleq \{G_{ij}\}_{i,j}^{n,n} \),
  where \( G_{ij} \triangleq \left\langle \begin{array}{c} x_i \\ \phi(x_i) \end{array}, \begin{array}{c} x_j \\ \phi(x_j) \end{array} \right\rangle_{\mathcal{K}}, \) Gram matrix.

- \( \hat{\alpha} = (G + \lambda I_n)^{-1}y \)

- \( \hat{w} = \sum_{i=1}^{n} \hat{\alpha}_i \phi(x_i) \).

- \( f(x) = \left\langle \hat{w}, \phi(x) \right\rangle = \sum_{i=1}^{n} \hat{\alpha}_i k(x_i, x) \)
SVM vs. Logistic Regression

**SVM**: Hinge loss
\[
\text{loss}(f(x_j), y_j) = (1 - (\mathbf{w} \cdot x_j + b)y_j)_{+}
\]

**Logistic Regression**: Log loss (log conditional likelihood)
\[
\text{loss}(f(x_j), y_j) = -\log P(y_j \mid x_j, \mathbf{w}, b) = \log(1 + e^{-(\mathbf{w} \cdot x_j + b)y_j})
\]
## Difference between SVMs and Logistic Regression

<table>
<thead>
<tr>
<th></th>
<th>SVMs</th>
<th>Logistic Regression</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Loss function</strong></td>
<td>Hinge loss</td>
<td>Log-loss</td>
</tr>
<tr>
<td><strong>High dimensional features with kernels</strong></td>
<td>Yes!</td>
<td>No (but there is kernel logistic regression too)</td>
</tr>
<tr>
<td><strong>Solution sparse</strong></td>
<td>Often yes!</td>
<td>Almost always no!</td>
</tr>
<tr>
<td><strong>Semantics of output</strong></td>
<td>“Margin”</td>
<td>“Real probabilities”</td>
</tr>
</tbody>
</table>
Constructing Kernels
Common Kernels

• Polynomials of degree $d$

$$K(u, v) = (u \cdot v)^d$$

• Polynomials of degree up to $d$

$$K(u, v) = (u \cdot v + 1)^d$$

• Gaussian/Radial kernels

$$K(u, v) = \exp \left( -\frac{||u - v||^2}{2\sigma^2} \right)$$

• Sigmoid

$$K(u, v) = \tanh(\eta u \cdot v + \nu)$$
Designing new kernels from kernels

\[ k_1 : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}, \quad k_2 : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R} \] are kernels \( \Rightarrow \)

1. \( k (x, \tilde{x}) = k_1 (x, \tilde{x}) + k_2 (x, \tilde{x}), \)
2. \( k (x, \tilde{x}) = c \cdot k_1 (x, \tilde{x}), \) for all \( c \in \mathbb{R}^+ , \)
3. \( k (x, \tilde{x}) = k_1 (x, \tilde{x}) + c, \) for all \( c \in \mathbb{R}^+ , \)
4. \( k (x, \tilde{x}) = k_1 (x, \tilde{x}) \cdot k_2 (x, \tilde{x}), \)
5. \( k (x, \tilde{x}) = f (x) \cdot f (\tilde{x}), \) for any function \( f : \mathcal{X} \rightarrow \mathbb{R} \)

are also kernels.

Picture is taken from R. Herbrich
Designing new kernels from kernels

1. \( k(x, \tilde{x}) = (k_1(x, \tilde{x}) + \theta_1)^{\theta_2}, \text{ for all } \theta_1 \in \mathbb{R}^+ \text{ and } \theta_2 \in \mathbb{N} \)

2. \( k(x, \tilde{x}) = \exp \left( \frac{k_1(x, \tilde{x})}{\sigma^2} \right), \text{ for all } \sigma \in \mathbb{R}^+ \)

3. \( k(x, \tilde{x}) = \exp \left( -\frac{k_1(x, x) - 2k_1(x, \tilde{x}) + k_1(\tilde{x}, \tilde{x})}{2\sigma^2} \right), \text{ for all } \sigma \in \mathbb{R}^+ \)

4. \( k(x, \tilde{x}) = \frac{k_1(x, \tilde{x})}{\sqrt{k_1(x, x) \cdot k_1(\tilde{x}, \tilde{x})}} \)}
Designing new kernels from kernels

The meaning of

\[
k(x, \tilde{x}) = \frac{k_1(x, \tilde{x})}{\sqrt{k_1(x, x)k_1(\tilde{x}, \tilde{x})}}
\]

is that we can normalize the data in the feature space without performing the explicit mapping.

Use the normalized kernel \( k_{norm} \):

\[
k_{norm}(x, \tilde{x}) = \frac{k(x, \tilde{x})}{\sqrt{k(x, x)k(\tilde{x}, \tilde{x})}} = \frac{\langle x, \tilde{x} \rangle}{\sqrt{\|x\|^2\|\tilde{x}\|^2}} = \langle \frac{x}{\|x\|^2}, \frac{\tilde{x}}{\|\tilde{x\|^2} \rangle}
\]
Higher Order Polynomials

\[ \text{num. terms} = \binom{d + m - 1}{d} = \frac{(d + m - 1)!}{d!(m - 1)!} \sim m^d \]

grows fast!

\( d = 6, m = 100 \)

about 1.6 billion terms
Dot Product of Polynomials

$\Phi(x) = \text{polynomials of degree exactly } d$

$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad z = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$

$d = 1 \quad \Phi(x) \cdot \Phi(z) = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \cdot \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = x_1 z_1 + x_2 z_2 = x \cdot z$

$d = 2 \quad \Phi(x) \cdot \Phi(z) = \begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_2^2 \end{bmatrix} \cdot \begin{bmatrix} z_1^2 \\ z_1 z_2 \\ z_2^2 \end{bmatrix} = x_1^2 z_1^2 + x_2^2 z_2^2 + x_1 x_2 z_1 z_2$

$= (x_1 z_1 + x_2 z_2)^2$

$= (x \cdot z)^2$

$d \quad \Phi(x) \cdot \Phi(z) = K(x, z) = (x \cdot z)^d$
\[ \text{dim}(\mathcal{X}) = N \]

<table>
<thead>
<tr>
<th>Name</th>
<th>Kernel function</th>
<th>dim (\mathcal{K})</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p \text{-th degree polynomial} )</td>
<td>( k(\bar{u}, \bar{v}) = (\langle \bar{u}, \bar{v} \rangle_{\mathcal{X}})^p )</td>
<td>( \binom{N+p-1}{p} )</td>
</tr>
<tr>
<td></td>
<td>( p \in \mathbb{N}^+ )</td>
<td></td>
</tr>
<tr>
<td>( \text{complete polynomial} )</td>
<td>( k(\bar{u}, \bar{v}) = (\langle \bar{u}, \bar{v} \rangle_{\mathcal{X}} + c)^p )</td>
<td>( \binom{N+p}{p} )</td>
</tr>
<tr>
<td></td>
<td>( c \in \mathbb{R}^+, \ p \in \mathbb{N}^+ )</td>
<td></td>
</tr>
<tr>
<td>( \text{RBF kernel} )</td>
<td>( k(\bar{u}, \bar{v}) = \exp \left( -\frac{|\bar{u} - \bar{v}|_{\mathcal{X}}^2}{2\sigma^2} \right) )</td>
<td>( \infty )</td>
</tr>
<tr>
<td></td>
<td>( \sigma \in \mathbb{R}^+ )</td>
<td></td>
</tr>
<tr>
<td>( \text{Mahalanobis kernel} )</td>
<td>( k(\bar{u}, \bar{v}) = \exp \left( -(\bar{u} - \bar{v})' \Sigma (\bar{u} - \bar{v}) \right) )</td>
<td>( \infty )</td>
</tr>
<tr>
<td></td>
<td>( \Sigma = \text{diag} \left( \sigma_1^{-2}, \ldots, \sigma_N^{-2} \right) )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>( \sigma_1, \ldots, \sigma_N \in \mathbb{R}^+ )</td>
<td></td>
</tr>
</tbody>
</table>
The RBF kernel

Note:

The RBF kernel maps the input space $\mathcal{X}$ onto the surface of an infinite dimensional hypersphere.

Proof:

$$\|\phi(x)\| = \sqrt{k(x, x)} = \sqrt{\exp(0)} = 1$$

Note:

The RBF kernel is shift invariant:

$$k(u + a, v + a) = k(u, v), \ \forall a$$
Overfitting

• Huge feature space with kernels, what about overfitting???
  • Maximizing margin leads to sparse set of support vectors
  • Some interesting theory says that SVMs search for simple hypothesis with large margin
  • Often robust to overfitting
String kernels

P-spectrum kernel:

$P=3: \quad s=\text{“statistics”} \quad t=\text{“computation”}$

They contain the following substrings of length 3

“com”, “omp”, “mpu”, “put”, “uta”, “tat”, “ati”, “tio”, “ion”

Common substrings: “tat”, “ati”

$k(s,t)=2$
Distribution kernels

Euclidean:

$$K(P, Q) = \int \rho(x) q(x) \, dx$$

Bhattacharyya's affinity:

$$K(P, Q) = \int \sqrt{\rho(x)} \sqrt{q(x)} \, dx$$

Mean map:

$$\Phi(P) = \mathbb{E}_{X \sim P} K(\cdot, X)$$

$$K(P, Q) = \mathbb{E}_{X \sim P} \mathbb{E}_{Y \sim Q} K(X, Y)$$
Set kernels

Mean map:
\[ \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} k_{2}(x_i, y_j) = \left\langle \{x_1, \ldots, x_n\}, \{y_1, \ldots, y_m\} \right\rangle \]

Intersection kernel:
\[ k_{2}(A_1, A_2) = \int_{A_1 \cap A_2} (x) \, dM(x) \]
\[ = M(A_1 \cap A_2) \]

Union complement kernel:
\[ 1 - M(A_1 \cup A_2) \, M(\emptyset) \]
What about multiple classes?
One against all

Learn 3 classifiers separately:
Class k vs. rest
\((w_k, b_k)_{k=1,2,3}\)

\[y = \arg \max_{k} w_k \cdot x + b_k\]

But \(w_k\)'s may not be based on the same scale.

Note: \((aw) \cdot x + (ab)\) is also a solution
Learn 1 classifier: Multi-class SVM

Simultaneously learn 3 sets of weights

\[ w^{(y_j)} \cdot x_j + b^{(y_j)} \geq w^{(y')} \cdot x_j + b^{(y')} + 1, \quad \forall y' \neq y_j, \quad \forall j \]

Margin - gap between correct class and nearest other class

\[ y = \arg \max \ w^{(k)} \cdot x + b^{(k)} \]
Learn 1 classifier: Multi-class SVM

Simultaneously learn 3 sets of weights

\[
\begin{align*}
\text{minimize}_{w,b} & \quad \sum_y w(y).w(y) + C \sum_j \sum_{y \neq y_j} \xi_j(y) \\
& \quad w(y_j).x_j + b(y_j) \geq w(y).x_j + b(y) + 1 - \xi_j(y), \quad \forall y \neq y_j, \forall j \\
& \quad \xi_j(y) \geq 0, \quad \forall y \neq y_j, \forall j
\end{align*}
\]

\[y = \arg \max w^{(k)}.x + b^{(k)}\]

Joint optimization: \(w_k\)s have the same scale.
Steve Gunn’s svm toolbox
Results, Iris 2vs13, Linear kernel

No. of Support Vectors: 2 (1.7%)
Results, Iris 1vs23, 2nd order kernel

No. of Support Vectors: 12 (100%)
\[ K(u, v) = \exp\left(-\frac{||u - v||^2}{2\sigma^2}\right) \] 

\( \sigma \to 0 \Rightarrow \text{MORE SUPPORT VECTORS} \)
Results, Iris 1vs23, RBF kernel

No. of Support Vectors: 41 (34.2%)
No. of Support Vectors: 103 (61.0%)
No. of Support Vectors: 102 (34.0%)
Results, Chessboard, RBF kernel

No. of Support Vectors: 174 (56.0%)
Sinc = \frac{\sin(\pi x)}{\pi x}, RBF kernel
\( \text{Sinc} = \frac{\sin(\pi x)}{\pi x} \), RBF kernel
Sinc = \frac{\sin(\pi x)}{\pi x}, RBF kernel
$Sinc = \frac{\sin(\pi x)}{\pi x}$, RBF kernel
Sinc = \frac{\sin(\pi x)}{\pi x}, RBF kernel
Sinc = \frac{\sin(\pi x)}{(\pi x)}, RBF kernel
Sinc = \frac{\sin(\pi x)}{\pi x}, \text{ poly kernel}
Sinc = \sin(\pi x) / (\pi x), poly kernel
Sinc = \frac{\sin(\pi x)}{\pi x}, poly kernel
Sinc = \sin(\pi x) / (\pi x), poly kernel
Sinc = \frac{\sin(\pi x)}{\pi x}, poly kernel
$\text{Sinc} = \frac{\sin(\pi x)}{\pi x}$, poly kernel
What you need to know…

- Dual SVM formulation
  - How it’s derived
- Common kernels
- Differences between SVMs and logistic regression
Thanks for your attention 😊