CS446 Introduction to Machine Learning (Spring 2015)
University of Illinois at Urbana-Champaign
http://courses.engr.illinois.edu/cs446

LECTURE 11:
SOFT SVMS

Prof. Julia Hockenmaier
juliahmr@illinois.edu

Midterm (Thursday, March 5, in class)

Format

Closed book exam (during class):

- You are not allowed to use any cheat sheets, computers, calculators, phones etc. (you shouldn’t have to anyway)
- Only the material covered in lectures (Assignments have gone beyond what’s covered in class)
- Bring a pen (black/blue).

Sample questions

What is *n*-fold cross-validation, and what is its advantage over standard evaluation?

Good solution:

- Standard evaluation: split data into test and training data (optional: validation set)
- *n*-fold cross validation: split the data set into *n* parts, run *n* experiments, each using a different part as test set and the remainder as training data.
- Advantage of *n*-fold cross validation: because we can report expected accuracy, and variances/standard deviation, we get better estimates of the performance of a classifier.

Question types

- *Define* X:
  Provide a mathematical/formal definition of X
- *Explain* what X is/does:
  Use plain English to say what X is/does
- *Compute* X:
  Return X; Show the steps required to calculate it
- *Show/Prove* that X is true/false/…:
  This requires a (typically very simple) proof.

Back to the material…

Last lecture’s key concepts

Large margin classifiers:

– Why do we care about the margin?

– Perceptron with margin

– Support Vector Machines

Today's key concepts

Review of SVMs

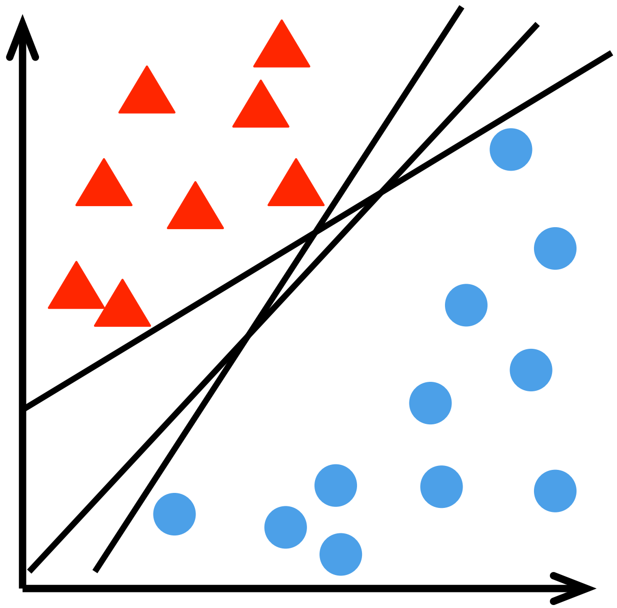
Dealing with outliers: Soft margins

Soft margin SVMs and Regularization

SGD for soft margin SVMs

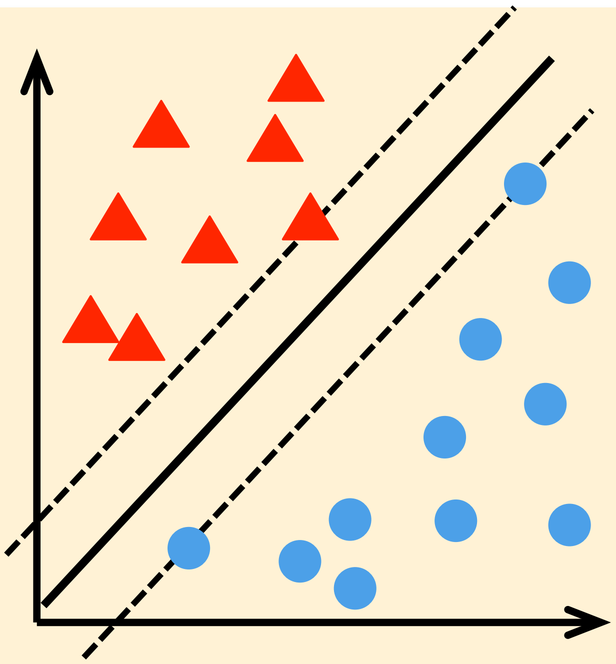
Review of SVMs

Maximum margin classifiers



These decision boundaries are very close to some items in the training data.
They have small margins.

Minor changes in the data could lead to different decision boundaries



This decision boundary is as far away from any training items as possible.
It has a large margin.

Minor changes in the data result in (roughly) the same decision boundary

Euclidean distances

If the dataset is linearly separable, the Euclidean (geometric) distance of $\mathbf{x}^{(i)}$ to the hyperplane $\mathbf{wx} + b = 0$ is

$$\frac{\left|\mathbf{wx}^{(i)} + b\right|}{\|\mathbf{w}\|} = \frac{y^{(i)}(\mathbf{wx}^{(i)} + b)}{\|\mathbf{w}\|} = \frac{y^{(i)}\left(\sum_n w_n x_n^{(i)} + b\right)}{\sqrt{\sum_n w_n w_n}}$$

The *Euclidean distance* of the data to the decision boundary will depend on the dataset.

Support Vector Machines

Distance of the training example **x**(i) from the decision boundary **wx** + b = 0:

$$\frac{y^{(i)}(\mathbf{w}\mathbf{x}^{(i)} + b)}{\|\mathbf{w}\|}$$

Learning an SVM = find parameters **w**, b such that the decision boundary **wx** + b = 0 is furthest away from the training examples closest to it:

functional distance to the closest training examples

$$\underset{\mathbf{w},\, b}{\operatorname{argmax}} \left\{ \frac{1}{\|\mathbf{w}\|} \min_{n} \left[y^{(n)}(\mathbf{w}\mathbf{x}^{(n)} + b) \right] \right\}$$

Find the boundary **wx** + b = 0 with maximal distance to the data

Support vectors and functional margins

Functional distance of a training example $(\mathbf{x}^{(k)}, y^{(k)})$ from the decision boundary:

$$y^{(k)} f(\mathbf{x}^{(k)}) = y^{(k)}(\mathbf{wx}^{(k)} + b) = \gamma$$

Support vectors: the training examples $(\mathbf{x}^{(k)}, y^{(k)})$ that have a functional distance of 1

$$y^{(k)} f(\mathbf{x}^{(k)}) = y^{(k)}(\mathbf{wx}^{(k)} + b) = 1$$

All other examples are further away from the decision boundary. Hence $\forall k$: $y^{(k)} f(\mathbf{x}^{(k)}) = y^{(k)}(\mathbf{wx}^{(k)} + b) \geq 1$

Rescaling $\mathbf{w}$ and b

Rescaling $\mathbf{w}$ and b by a factor k to $k\boldsymbol{w}$ and kb changes the functional distance of the data but does not affect geometric distances (see last lecture)

We can therefore decide to fix the functional margin (distance of the closest points to the decision boundary) to 1, regardless of their Euclidean distances.

Support Vector Machines

Learn **w** in an SVM = maximize the margin:

$$\underset{\mathbf{w},\, b}{\operatorname{argmax}} \left\{ \frac{1}{\|\mathbf{w}\|} \min_{n} \left[y^{(n)} (\mathbf{wx} + b) \right] \right\}$$

Easier equivalent problem:

- We can always rescale **w** and *b* without affecting Euclidean distances.
- This allows us to set the functional margin to 1: $\min_n(y^{(n)}(\mathbf{wx}^{(n)} + b) = 1$

Support Vector Machines

Learn **w** in an SVM = maximize the margin:

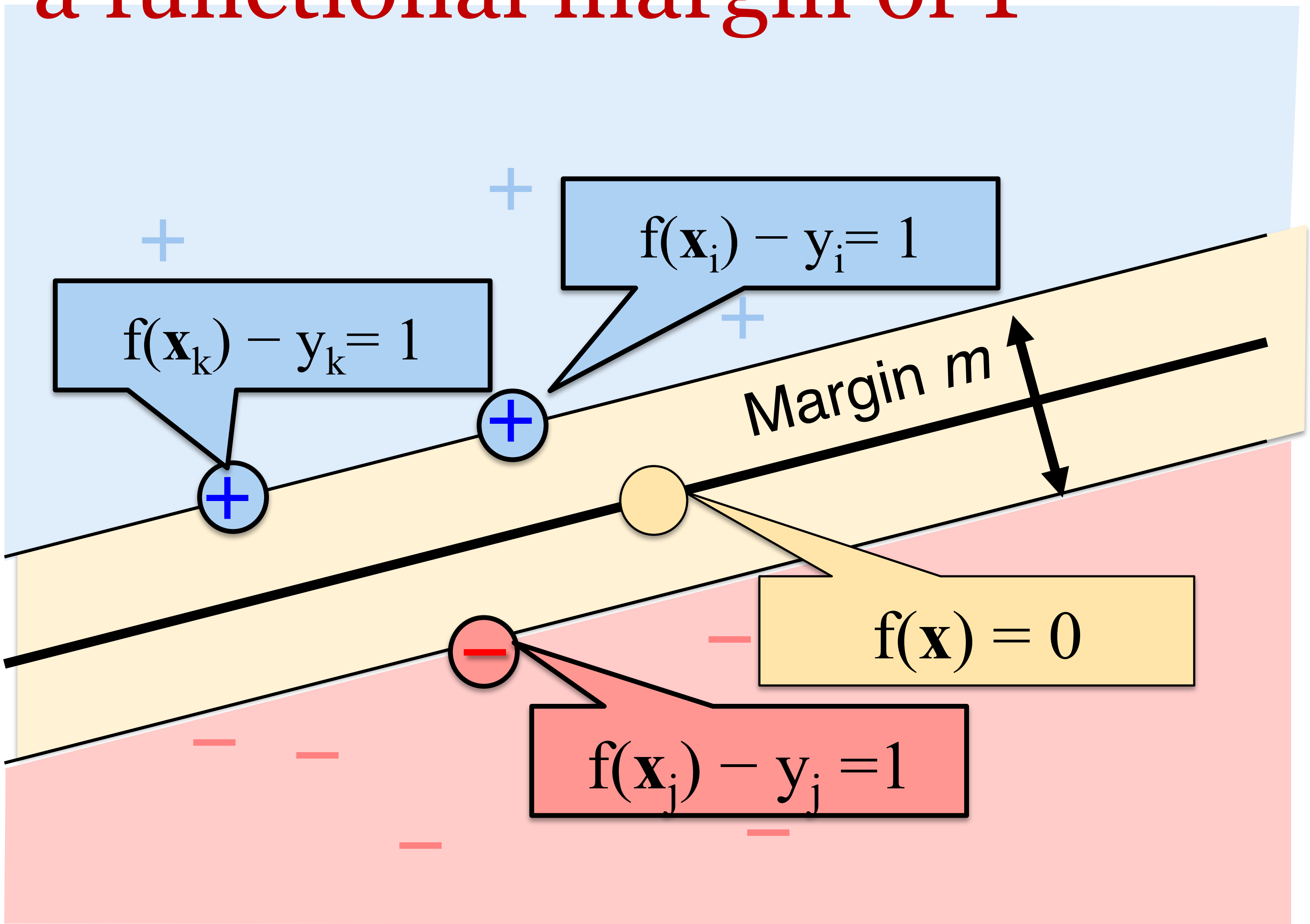
$$\underset{\mathbf{w},\,b}{\operatorname{argmax}} \left\{ \frac{1}{\|\mathbf{w}\|} \min_{n} \left[y^{(n)}(\mathbf{wx} + b) \right] \right\}$$

Easier equivalent problem: a quadratic program

- Setting $\min_n(y^{(n)}(\mathbf{wx}^{(n)} + b) = 1$
  implies $(y^{(n)}(\mathbf{wx}^{(n)} + b) \geq 1$ for all n
- argmax(1/**ww**)= argmin(**ww**) = argmin(1/2·**ww**)

$$\underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{2}\mathbf{w}\cdot\mathbf{w} \\ \textit{subject to} \\ y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 \;\; \forall i$$

Support vectors: Examples with a functional margin of 1


$f(\mathbf{x}_i) - y_i = 1$
$f(\mathbf{x}_k) - y_k = 1$
Margin *m*
$f(\mathbf{x}) = 0$
$f(\mathbf{x}_j) - y_j = 1$
+
−

Support Vector Machines

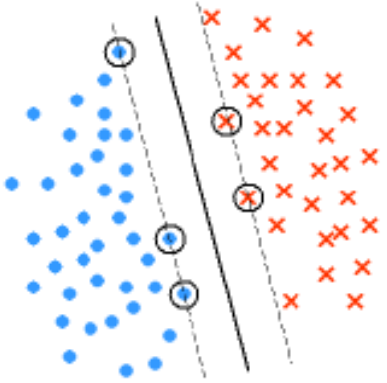
The name "Support Vector Machine" stems from the fact that **w*** is supported by (i.e. is the linear span of) the examples that are exactly at a distance 1/||**w***|| from the separating hyperplane. These vectors are therefore called **support vectors**.

Theorem: Let **w*** be the minimizer of the SVM optimization problem for $S = \{(\mathbf{x}_i, y_i)\}$.

Let $I = \{i: y_i (\mathbf{w}^* \mathbf{x}_i + b) = 1\}$.

Then there exist coefficients $\alpha_i > 0$ such that:

$$\mathbf{w}^* = \sum_{i \in I} \alpha_i \, y_i \, \mathbf{x}_i$$



Support vectors = the set of data points $\mathbf{x}_j$ with **non-zero weights** α_j

Summary: (Hard) SVMs

If the training data is linearly separable, there will be a decision boundary $\mathbf{wx} + b = 0$ that perfectly separates it, and where all the items have a functional distance of at least 1: $y^{(i)}(\mathbf{wx}^{(i)} + b) \geq 1$

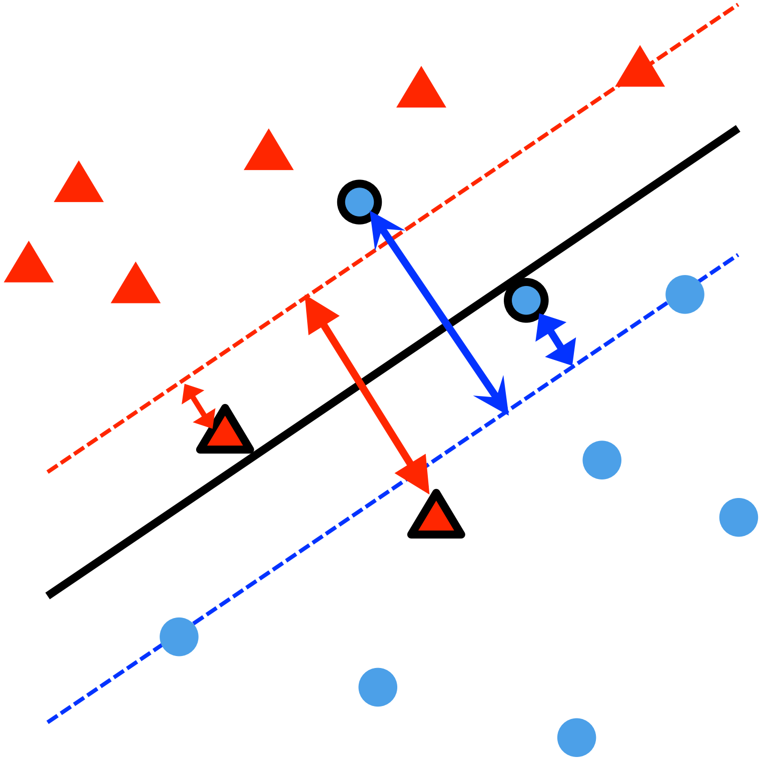
We can find $\mathbf{w}$ and b with a quadratic program:

$$\boxed{\begin{gathered}\underset{\mathbf{w},b}{\operatorname{argmin}} \frac{1}{2}\mathbf{w}\cdot\mathbf{w} \\ \textit{subject to} \\ y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 \;\; \forall i\end{gathered}}$$

Dealing with outliers: Soft margins

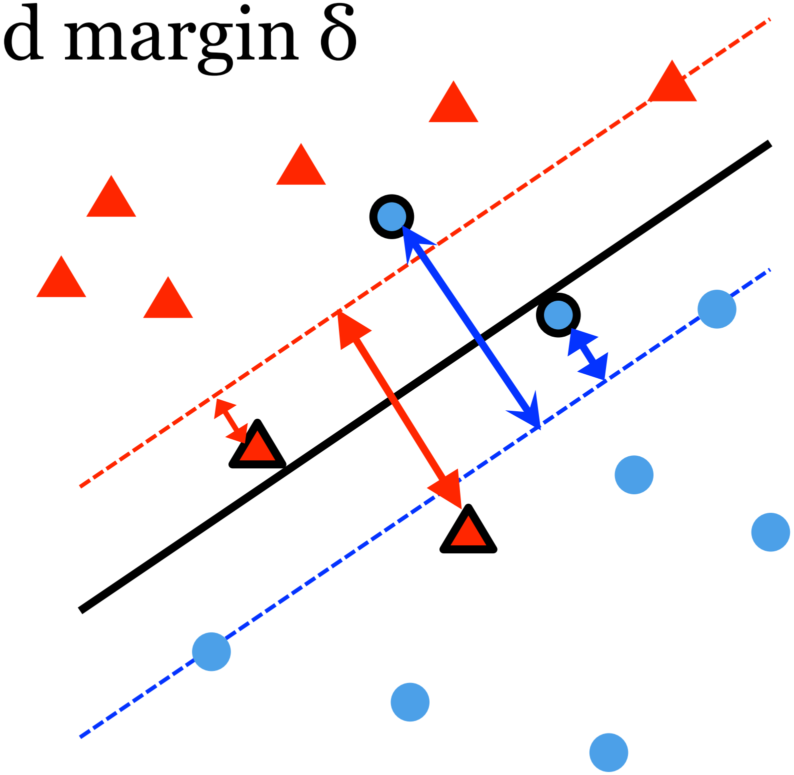
Dealing with outliers

Not every dataset is linearly separable.
There may be outliers:



Dealing with outliers: Slack variables ξ_i

Associate each $(\mathbf{x}^{(i)}, y^{(i)})$ with a slack variable ξ_i that measures by how much it fails to achieve the desired margin δ



Dealing with outliers: Slack variables ξ_i

If $\mathbf{x}^{(i)}$ is on the correct side of the margin:

$\mathbf{wx}^{(i)} + b \geq 1$: $\xi_i = 0$

If $\mathbf{x}^{(i)}$ is on the wrong side of the margin:

$\mathbf{wx}^{(i)} + b < 1$: $\xi_i > 0$

If $\mathbf{x}^{(i)}$ is on the decision boundary:

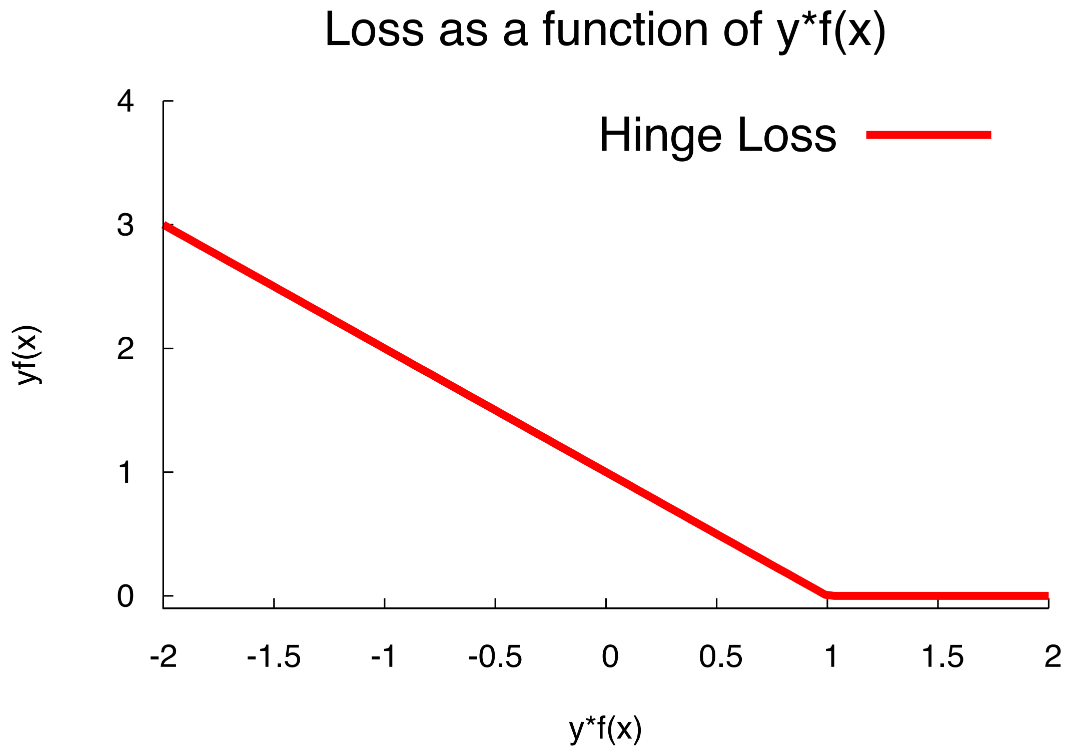
$\mathbf{wx}^{(i)} + b = 1$: $\xi_i = 1$

Hence, we will now assume that

$$\mathbf{wx}^{(i)} + b \geq 1 - \xi_i$$

Hinge loss and SVMs

$$L_{hinge}(y^{(n)}, f(\mathbf{x}^{(n)})) = \max(0, 1 - y^{(n)} f(\mathbf{x}^{(n)}))$$



Case 0: $f(\mathbf{x}) = 1$
$\mathbf{x}$ is a support vector
Hinge loss = 0

Case 1: $f(\mathbf{x}) > 1$
$\mathbf{x}$ outside of margin
Hinge loss = 0

Case 2: $0 < yf(\mathbf{x}) < 1$:
$\mathbf{x}$ inside of margin
Hinge loss = $1 - yf(\mathbf{x})$

Case 3: $yf(\mathbf{x}) < 0$:
$\mathbf{x}$ misclassified
Hinge loss = $1 - yf(\mathbf{x})$

From Hard SVM to Soft SVM

Replace $y^{(n)}(\mathbf{wx}^{(n)} + b) \geq 1$ (hard margin)
with $y^{(n)}(\mathbf{wx}^{(n)} + b) \geq 1 - \xi^{(n)}$ (soft margin)

$y^{(n)}(\mathbf{wx}^{(n)} + b) \geq 1 - \xi^{(n)}$ is the same as
$\xi^{(n)} \geq 1 - y^{(n)}(\mathbf{wx}^{(n)} + b)$

Since $\xi^{(n)} > 0$ only if $\mathbf{x}^{(n)}$ is on the wrong side of the margin, i.e. if $y^{(n)}(\mathbf{wx}^{(n)} + b) < 1$, this is the same as the **hinge loss:**

$$L_{\text{hinge}}(y^{(n)}, f(\mathbf{x}^{(n)})) = \max(0, 1 - y^{(n)} f(\mathbf{x}^{(n)}))$$

Soft margin SVMs

$$\underset{\mathbf{w},b,\xi_i}{\operatorname{argmin}} \frac{1}{2}\mathbf{w}\cdot\mathbf{w} + C\sum_{i=1}^{n}\xi_i$$

$$subject\ \ to$$

$$\xi_i \geq 0 \ \ \forall i$$

$$y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq (1-\xi_i)\forall i$$

ξ_i (slack): how far off is $\mathbf{x}_i$ from the margin?

C (cost): how much do we have to pay for misclassifying $\mathbf{x}_i$

We want to minimize $C\sum_i \xi_i$ and maximize the margin

C controls the tradeoff between margin and training error

Soft SVMs = Regularized Hinge Loss:

We can rewrite this as:

$$\underset{\mathbf{w},b}{\operatorname{argmin}} \frac{1}{2}\mathbf{w}\cdot\mathbf{w} + C\sum_{i=1}^{n} L_{hinge}(y^{(n)}, \mathbf{x}^{(n)})$$

$$= \underset{\mathbf{w},b}{\operatorname{argmin}} \frac{1}{2}\mathbf{w}\cdot\mathbf{w} + C\sum_{i=1}^{n} \max(0, 1 - y^{(n)}(\mathbf{w}\mathbf{x}^{(n)} + b))$$

The parameter C controls the tradeoff between choosing a large margin (small ||**w**||) and choosing a small hinge-loss.

Soft SVMs = Regularized Hinge Loss:

$$\underset{\mathbf{w},b}{\operatorname{argmin}} \frac{1}{2}\mathbf{w}\cdot\mathbf{w} + C\sum_{i=1}^{n} L_{hinge}(y^{(n)},\mathbf{x}^{(n)})$$

We minimize both the l2-norm of the weight vector ||**w**|| = √**ww** and the hinge loss.

Minimizing the norm of **w** is called regularization.

Regularized Loss Minimization

Empirical Loss Minimization: $\text{argmin}_{\mathbf{w}}\ L(D)$

$L(D) = \sum_i L(y^{(i)},\mathbf{x}^{(i)})$: Loss of **w** on training data *D*

Regularized Loss Minimization:

Include a regularizer R(**w**) that constrains **w**

e.g. L2-regularization: $R(\mathbf{w}) = \lambda \,||\, \mathbf{w} \,||\, ^2$

$$\text{argmin}_{\mathbf{w}}\ (L(D) + R(\mathbf{w}))$$

λ controls the tradeoff between empirical loss and regularization.

Training SVMs

Traditional approach:

Solve quadratic program.

– This is very slow.

Current approaches:

Use variants of stochastic gradient descent or coordinate descent.

Gradient of hinge loss at $\mathbf{x}^{(n)}$

$L_{hinge}(y^{(n)}, f(\mathbf{x}^{(n)})) = \max(0, 1 - y^{(n)}f(\mathbf{x}^{(n)}))$

Gradient

If $y^{(n)}f(\mathbf{x}^{(n)}) \geq 1$: set the gradient to 0

If $y^{(n)}f(\mathbf{x}^{(n)}) < 1$: set the gradient to $-y^{(n)}\mathbf{x}^{(n)}$

SGD for SVMs

Minimizing regularized hinge loss:

If $y^{(n)} f(\mathbf{x}^{(n)}) < 1$:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t+1)} + y^{(n)} \mathbf{x}^{(n)}$$

$$\mathbf{w}^{(t+1)} = \boldsymbol{\theta}^{(t)} / (\lambda t)$$

Dividing $\boldsymbol{\theta}^{(t)}$ by (λt) is a projection step.

Summary: SVMs

Hinge loss: Penalize misclassified items as well as items inside the margin

Hard SVMs assume linear separability

- Learning hard SVMs = Minimizing hinge loss

Soft SVMs allow for outliers.

- Each outlier is associated with a slack variable
- Learning soft SVMs = Minimizing regularized hinge loss