CS 559: Kernel Methods, Gaussian Process & Supportive Vector Machine

Lecture 6

Supportive Vector Machine (SVM)

Supportive Vector Machine

- Maximum Margin Classifiers
- Overlapping class distributions
- Classification
- Regression
- Kernel SVM - Regression

SVM

$$y(x) = w^T \phi(x) + b$$

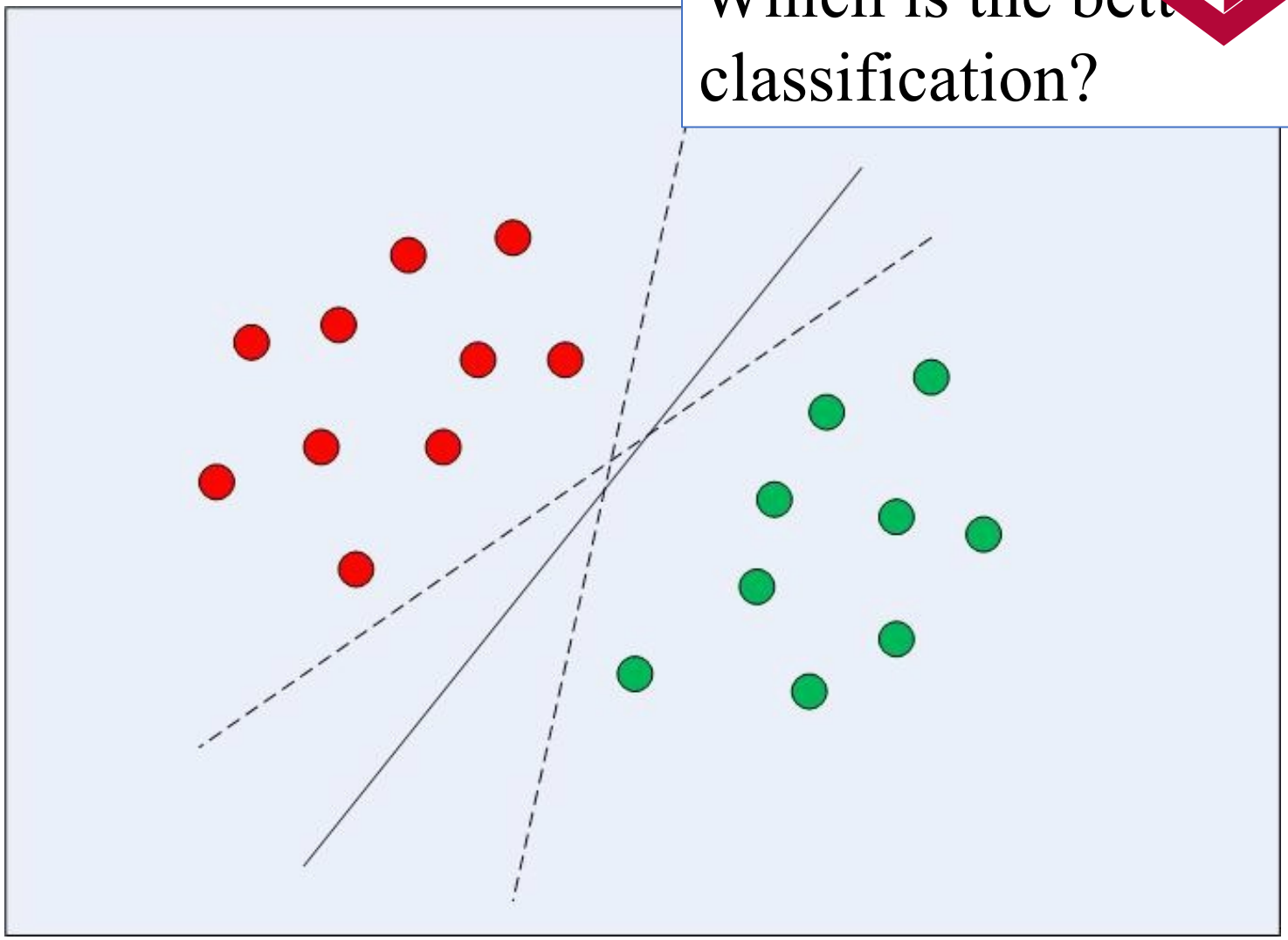
- $\phi(x)$ a fixed feature-space transformation
- b the bias parameter
- training set - linearly separable in feature space
- Data: $\{(x_i, t_i)|n = 1, \ldots, n\}$
- $t_i \in \{-1,1\}$

SVM vs. Perceptron Algorithm

Perceptron Algorithm

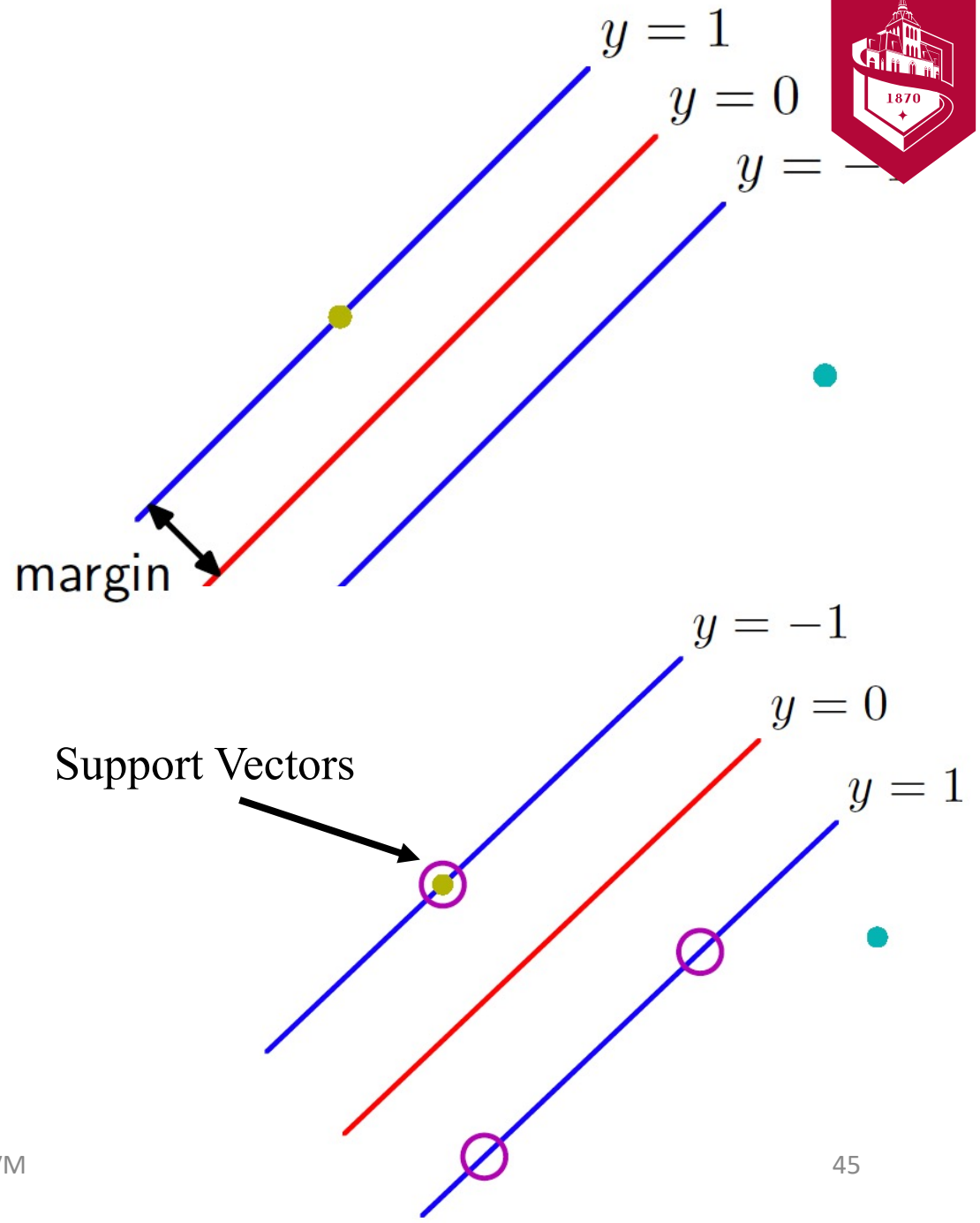
- guarantees to find a solution in a finite number of steps
- initial value for w and b starts from arbitrary chosen value..

If there are multiple solutions of all of which classify the training data set exactly, then we should try to find the one that give **the smallest generalization error!**



SVM

- An extension of the Perceptron developed by Rosenblatt in 1958.
- Concept of the margin - the smallest distance between the decision boundary and any of the samples.
- Choose the max margin
- Using the optimal boundary, determine the best ***hyperplane*** by minimizing the probability of error relative to the learned density mode.
- hyperplane becomes independent of data points that are not ***support vectors (SV)***.



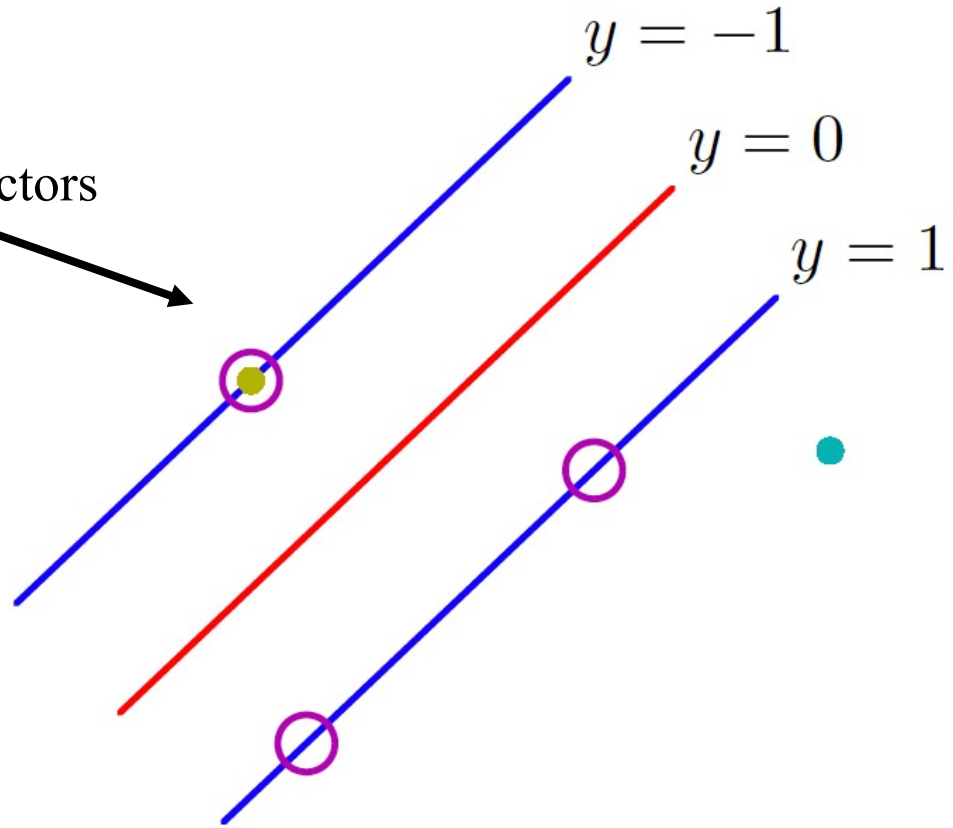
SVM

Hyperplane is $y(x) = 0$

Right classification $t_n y(x_n) > 0$

$$\frac{t_n y(\boldsymbol{x_n})}{||\boldsymbol{w}||} = \frac{t_n[\boldsymbol{w^T}\phi(\boldsymbol{x_n}) + b]}{||\boldsymbol{w}||}$$

Support Vectors



Optimize the parameters w and b to maximize the distance.

Maximum margin solution is

$$\underset{\boldsymbol{w},b}{\operatorname{argmax}} \left\{ \frac{1}{||\boldsymbol{w}||} \min_n \left[t_n \left(\boldsymbol{w^T}\phi(\boldsymbol{x_n}) + b \right) \right] \right\}$$

w does not depend on n. canonical representation of the decision hyperplane.

SVM

Direction solution is quite complex, we convert it into an equivalent problem (rescale)

$$t_n(w^T\phi(x)+b)=1$$

for the point that is closest to the surface. All data points will satisfy the constrains

$$t_n(w^T\phi(x)+b)\geq 1$$

once the margin has been maximized there will be at least two active constraints.

So we maximize $||w||^{-1}$ and this is same as minimizing $||w||^2$

$$\underset{w,b}{\operatorname{argmin}}\frac{1}{2}||w||^2$$

Lagrange multipliers

We introduce Lagrange Multipliers $a_n \geq 0$

$$L(\boldsymbol{w}, b, \boldsymbol{a}) = \frac{1}{2}\left|\left|w\right|\right|^2 - \sum_{n=1}^{N} a_n \left\{t_n\left(\boldsymbol{w}^T \phi(\boldsymbol{x_n}) + b\right) - 1\right\} \quad (6\text{-}28)$$

- a strategy for finding the local maxima and minima of a function subject to equality constraints

Derivative:

$$w = \sum_{n=1}^{N} a_n \, t_n \phi(x_n)$$

$$0 = \sum_{n=1}^{N} a_n t_n$$

Dual Representation

Dual representation of the maximum margin problem

$$L(a) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{N} a_n\, a_m t_n t_m k(x_n, x_m) \qquad (6\text{-}29)$$

Use the kernel function.

This takes the form of a quadratic programming problem

$$a_n \geq 0$$

$$0 = \sum_{n=1}^{N} a_n t_n$$

Prediction

$$y(\boldsymbol{x}) = \sum_{n=1}^{N} a_n t_n k(\boldsymbol{x}, \boldsymbol{x_n}) + b \qquad (6\text{-}30)$$

Satisfies the following conditions:

$$a_n \geq 0$$
$$t_n y(x_n) - 1 \geq 0$$
$$a_n\{t_n y(x_n) - 1\} = 0$$

for point with $a_n = 0$ will not appear in the sum and plays no role making predictions for the new point.

The remaining data points are called support vectors because they satisfy $t_n y(x_n) = 1$ and correspond to points that lie on the maximum margin hyperplanes in feature space.

Prediction

Using Eq (6-30)

$$t_n\left(\sum_{m\in S} a_m t_m k(x_n, x_m) + b\right) = 1 \tag{6-31}$$

S denotes the set of indices of the support vectors (SV).

Making a use of $t_n^2 = 1$ averaging over all SV and solve for b

$$b = \frac{1}{N_s}\left(\sum_{n\in S}\left(t_n - \sum_{m\in S} a_m t_m k(x_n, x_m)\right)\right) \tag{6-32}$$

N_s is the total number of SV.

The maximum margin classifier in terms of the minimization of an error function

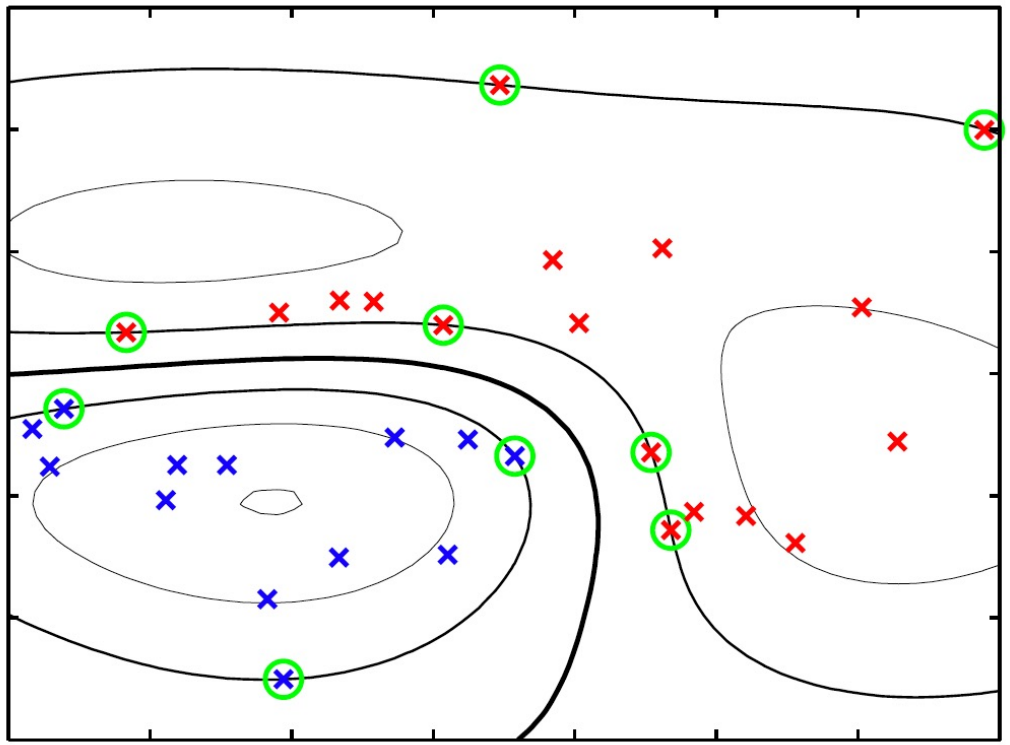
$$\sum_{n=1}^{N} E_\infty(y(x_n)t_n - 1) + \lambda\left|\left|w\right|\right|^2 \tag{6-33}$$

Prediction

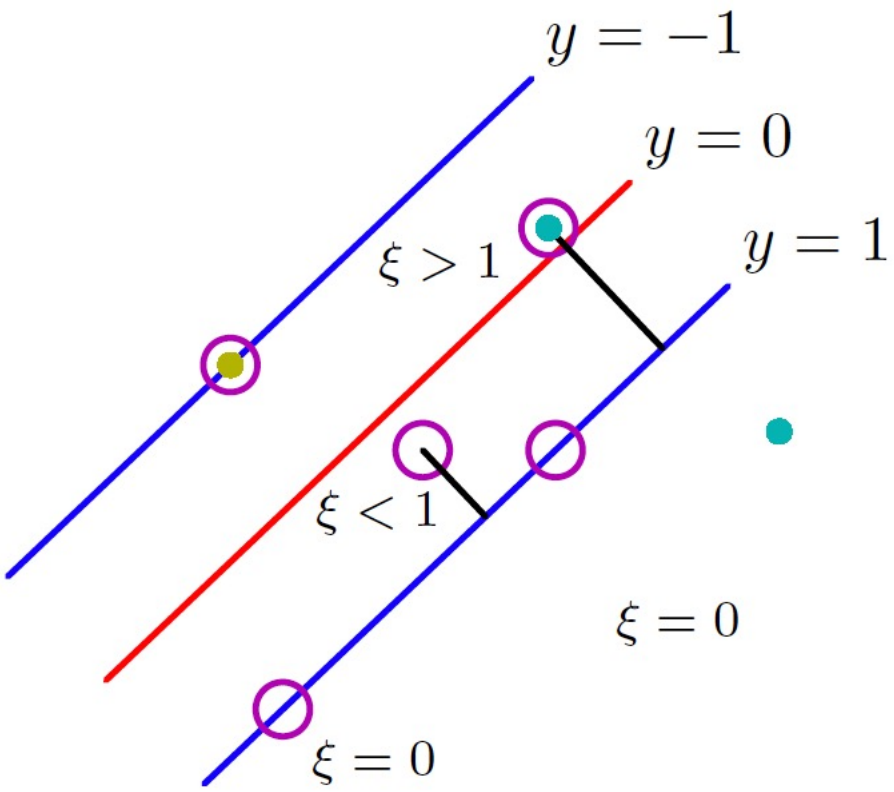
The maximum margin classifier in terms of the minimization of an error function

$$\sum_{n=1}^{N} E_{\infty}(y(x_n)t_n - 1) + \lambda\left|\left|w\right|\right|^2 \qquad (6\text{-}34)$$

where E is a function that is zero if $z \geq 0$ and ∞ otherwise to ensure that the constraints satisfies.



Overlapping Class Distributions



The class-conditional distributions may overlap, in which case exact separation of the training data can lead to poor generalization.

Need to modify SVM

Modify so that data points are allowed to be on the wrong side of the margin boundary, but with penalty that increases with the distance from that boundary.

Overlapping Class Distributions – soft SVM

Make the penalty a linear function of the distance.

slack variable $\xi_n \geq 0$

$\xi = 0$ for data points that are on or inside the correct margin boundary and $\xi = |t_n - y(x_n)|$ for other points.
On the decision boundary, $y(x_n) = 0$ and have $\xi = 1$ and with $\xi > 1$ will be misclassified

$$t_n y(x_n) \geq 1 - \xi_n$$

Goal is to maximize the margin while **softly penalizing points** that lie on the wrong side of the margin boundary

Overlapping Class Distributions – soft SVM

$$C\sum_{n=1}^{N}\xi_n+\frac{1}{2}\left|\left|w\right|\right|^2 \tag{6-37}$$

where parameter C>0 controls the trade-off between the slack variable penalty and the margin.

$$L(w,b,\xi,a,\mu) \tag{6-38}$$
$$=\frac{1}{2}\left|\left|w\right|\right|^2+C\sum_{n=1}^{N}\xi_n+\sum_{n=1}^{N}a_n\{t_n y_n(x_n)-1+\xi_n\}-\sum_{n=1}^{N}\mu_n\xi_n$$

Lagrange Multipliers - $\{a_n\geq 0\}$ and $\{\mu_n\geq 0\}$:

$a_n=0, t_n y_n(x_n)-1+\xi_n\geq 0, a_n\{t_n y_n(x_n)-1+\xi_n\}=0$

$\mu_n\geq 0, \xi_n\geq 0, \mu_n\xi_n=0$

Overlapping Class Distributions – soft SVM

Optimize out w, b, and ξ

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{n=1}^{N} a_n t_n \phi(x_n)$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{n=1}^{N} a_n t_n = 0$$

$$\frac{\partial L}{\partial \xi_N} = 0 \Rightarrow a_n = C - \mu_n$$

$0 \leq a_n \leq C$ Box Constraints

$$L(a) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{m=1}^{N} \sum_{n=1}^{N} a_n \, a_m t_n t_m k(x_n, x_m)\Big) \qquad (6\text{-}39)$$

Overlapping Class Distributions – soft SVM

Solution

$$t_n\left(\sum_{m\in S} a_m t_m k(x_n, x_m) + b\right) = 1 \qquad (6\text{-}40)$$

$$b = \frac{1}{N_{\mathcal{M}}}\left(\sum_{n\in\mathcal{M}}\left(t_n - \sum_{m\in S} a_m t_m k(x_n, x_m)\right)\right) \qquad (6\text{-}41)$$

where M is the set of indices of data points having $0 < a_n < C$.

Overlapping Class Distributions – soft SVM

IF we wish to use the SVM as a module in a larger probabilistic system

Platt (2000) has proposed fitting a logistic sigmoid to the outputs of a trained SVM.

$$p(t = 1|x) = \sigma(Ay(x) + B)$$

A & B are found by minimizing the cross-entropy error function defined by a training set consisting of pairs of values y and t.

- The data used to fit the sigmoid needs to be independent to avoid sever over-fitting.
- Give a poor approximation to the posterior prob.

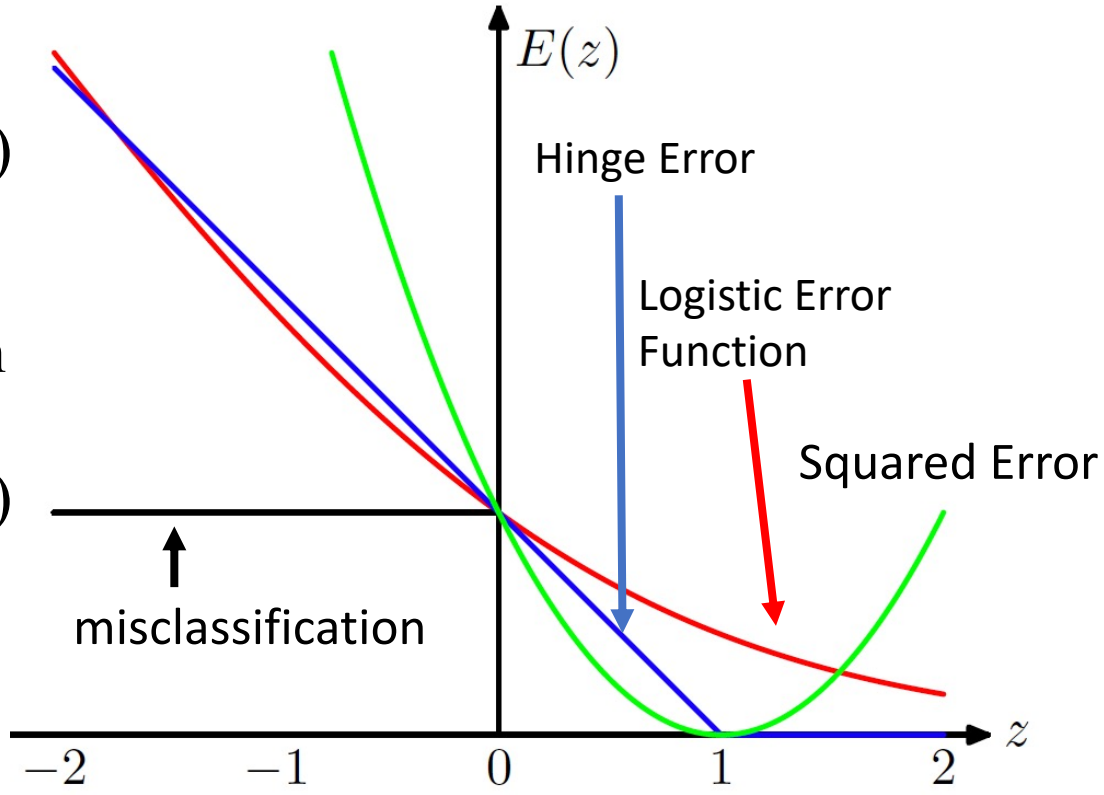
Relation to logistic regression – Hinge Error

Update Eq (7)

$$\sum_{n=1}^{N} E_{SV}(y_n t_n) + \lambda ||w||^2 \qquad (6\text{-}42)$$

where $\lambda = (2C)^{-1}$ and E_{SV} is the **hinge error** function

$$E_{SV}(y_n t_n) = [1 - y_n t_n]_+ \qquad (6\text{-}43)$$



Relation to logistic regression

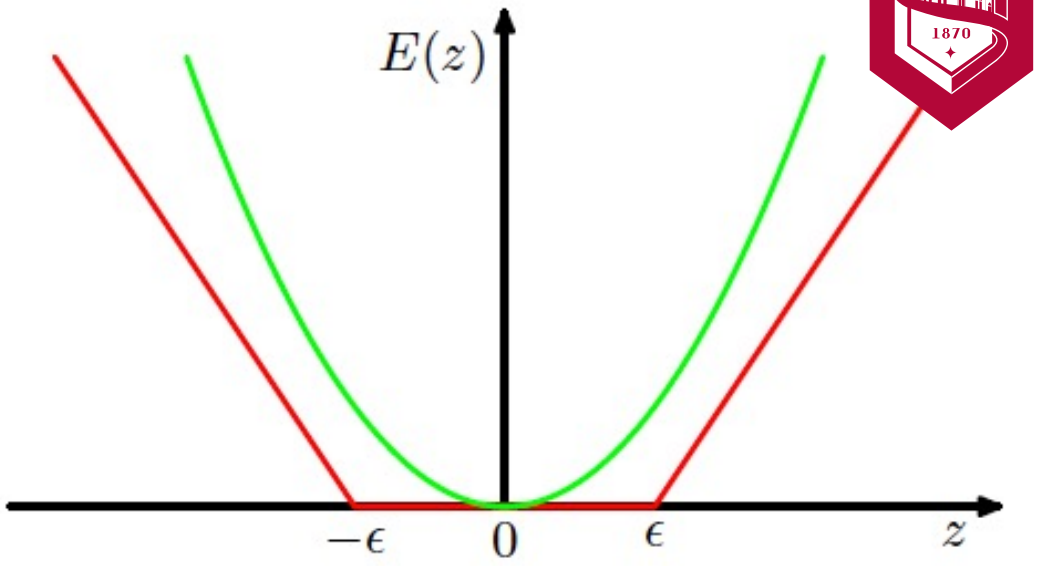
	SVMs	Logistic Regression
Loss Function	Hinge Loss	Log-loss
High dimensional features	Yes	No
Solution Sparse	Yes	Almost no
Output	Margin	Real Probabilities!

SVM for regression

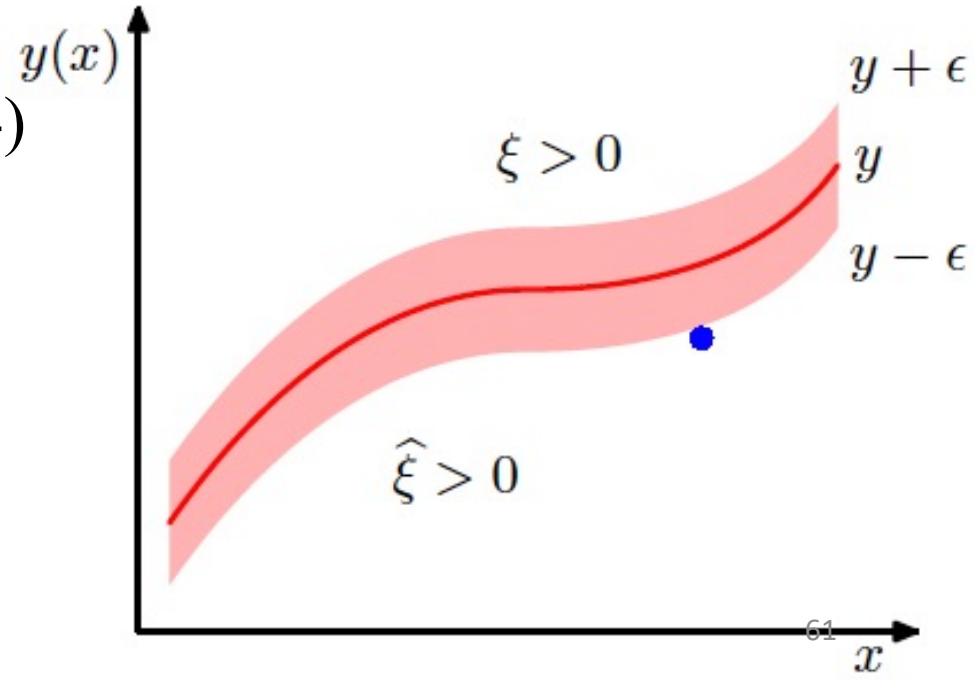
$$\frac{1}{2}\sum_{n=1}^{N}\{y_n - t_n\}^2 + \frac{\lambda}{2}||w||^2 \qquad (6\text{-}43)$$

ϵ-insensitive error function (Vapnik, 1995)

$$E_\epsilon(y(x) - t) = \begin{cases} 0 & if\ |y(x) - t| < \epsilon \\ |y(x) - t| & \text{Otherwise} \end{cases}$$



$$C\sum_{n=1}^{N} E_\epsilon(y_n(x_n) - t_n) + \frac{\lambda}{2}||w||^2 \qquad (6\text{-}44)$$



$\xi_n > 0$ corresponds to a point $t_n > y(x_n) + \epsilon$ and
$\hat{\xi}_n \geq 0$ for $t_n < y(x_n) - \epsilon$

$$C\sum_{n=1}^{N}\left(\xi_n + \hat{\xi}_n\right) + \frac{1}{2}||w||^2 \qquad (6\text{-}45)$$

SVM for regression - Optimize by slack variable

$$
\begin{aligned}
&\mathrm{L} \\
&= C\sum_{n=1}^{N}\left(\xi_n + \hat{\xi}_n\right) + \frac{1}{2}\left|\left|w\right|\right|^2 - \sum_{n=1}^{N}\left(\mu_n\xi_n + \hat{\mu}_n\hat{\xi}_n\right) - \sum_{n=1}^{N} a_n\left(\epsilon + \xi_n + y_n - t_n\right) \\
&- \sum_{n=1}^{N}\hat{a}_n\left(\epsilon + \hat{\xi}_n - y_n + t_n\right)
\end{aligned}
\tag{6-46}
$$

The derivative w.r.t. w, b, ξ_n, and $\hat{\xi}_n$

$$
\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{n=1}^{N}(a_n - \hat{a}_n)\phi(x_n)\,, \frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{n=1}^{N}(a_n - \hat{a}_n) = 0
$$

$$
\frac{\partial L}{\partial \xi_n} = 0 \Rightarrow a_n = C - \mu_n, \frac{\partial L}{\partial \hat{\xi}_n} = 0 \Rightarrow \hat{a}_n = C - \hat{\mu}_n
$$

SVM for regression - Optimize by slack variable

$$L(a,\hat{a}) = -\frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{N}(a_n-\hat{a}_n)(a_m-\hat{a}_m)k(x_n,x_m) - \epsilon\sum_{n=1}^{N}(a_n+\hat{a}_n) + \sum_{n=1}^{N}(a_n-\hat{a}_n)\,t_n \quad (6\text{-}47)$$

$$y(x) = \sum_{n=1}^{N}(a_n-\hat{a}_n)k(x,x_n) + b$$

$$a_n(\epsilon+\xi_n+y_n-t_n)=0$$
$$\hat{a}_n(\epsilon+\hat{\xi}_n-y_n+t_n)=0$$
$$(C-a_n)\xi_n=0$$
$$(C-\hat{a}_n)\hat{\xi}_n=0$$

$$b = t_n - \epsilon - w^T\phi(x_n) = t_n - \epsilon - \sum_{m=1}^{N}(a_m-\hat{a}_m)k(x_n,x_m) \quad (6\text{-}48)$$

$$L(a,\hat{a}) = -\frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{N}(a_n-\hat{a}_n)(a_m-\hat{a}_m)k(x_n,x_m) + \sum_{n=1}^{N}(a_n-\hat{a}_n)\,t_n \quad (6\text{-}49)$$

Kernel SVM

- Outputs represents decisions (can be an advantage but also can be a disadvantage)
- Originally formulated for two classes
- Predictions are linear combination of kernel functions **but centered on training data points and that are required to be positive-definite**.

- How can we overcome the limitation of centered kernel?

- The relevance vector machine or RVM (Tipping 2001)
  - Faster performance on test data while the generalization error is maintained.

RVM - Regression

The modified prior:

$$p(t|x, w, \beta) = \mathcal{N}(t|y(x), \beta^{-1}) \tag{6-50}$$

where $\beta = \sigma^{-2}$ is the noise precision.

The mean of the linear model

$$y(x) = \sum_{i=1}^{M} w_i \phi_i(x) = w^T \phi(x)$$

In RVM, we use kernels with one kernel associated with each of the data points in training set.

$$y(x) = \sum_{i=1}^{N} w_n k(x, x_n) + b \tag{6-52}$$

where b is the bias parameter and $M = N + 1$.

Here, **there is no restriction of positive-definite and the number nor the location to the training data sets.**

RVM - Regression

Suppose we have N observations of data matrix X. The likelihood function is

$$p(\boldsymbol{t}|\boldsymbol{X},\boldsymbol{w},\beta) = \prod_{n=1}^{N} p(t_n|\boldsymbol{x}_n,\boldsymbol{w},\beta)$$

and a weight prior distribution is

$$p(\boldsymbol{w}|\boldsymbol{\alpha}) = \prod_{n=1}^{M} \mathcal{N}(w_i|0,\alpha_i^{-1})$$

α_i is a separate hyperparameter for each of weight parameter.

If the posterior distribution for the weights is Gaussian, then it is

$$p(\boldsymbol{t}|\boldsymbol{X},\boldsymbol{w},\beta) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{m},\boldsymbol{\Sigma})$$

Mean: $\beta\boldsymbol{\Sigma}\boldsymbol{\Phi}^{\boldsymbol{T}}\boldsymbol{t} = \beta\boldsymbol{\Sigma}\mathbf{K}^{\mathbf{T}}\mathbf{t}$

Covariance: $\Sigma = \left(\boldsymbol{A} + \beta\boldsymbol{\Phi}^{\boldsymbol{T}}\boldsymbol{\Phi}\right)^{-1}$

$= \left(\boldsymbol{A} + \beta\boldsymbol{K}^{\boldsymbol{T}}\boldsymbol{K}\right)^{-1}$

The maximization of likelihood function by integration of the weight parameters

$$p(\boldsymbol{t}|\boldsymbol{X},\boldsymbol{\alpha},\beta) = \int p(\boldsymbol{t}|\boldsymbol{X},\boldsymbol{w},\beta)p(\boldsymbol{w}|\boldsymbol{\alpha})d\boldsymbol{w} \quad (6\text{-}53)$$

RVM - Regression

If the posterior distribution for the weights is Gaussian, then it is

$$p(\boldsymbol{t}|\boldsymbol{X},\boldsymbol{w},\beta) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{m},\boldsymbol{\Sigma})$$

The maximization of likelihood function by integration of the weight parameters

$$p(\boldsymbol{t}|\boldsymbol{X},\boldsymbol{\alpha},\beta) = \int p(\boldsymbol{t}|\boldsymbol{X},\boldsymbol{w},\beta)p(\boldsymbol{w}|\boldsymbol{\alpha})d\boldsymbol{w} \quad (6\text{-}54)$$

The log likelihood also can be formatted as

$$\ln p(\boldsymbol{t}|\boldsymbol{X},\boldsymbol{\alpha},\beta) = \ln \mathcal{N}(\boldsymbol{t}|\boldsymbol{0},\boldsymbol{C}) - \frac{1}{2}\{N\ln(2\pi) + \ln|\boldsymbol{C}| + \boldsymbol{t}^T\boldsymbol{C}^{-1}\boldsymbol{t}\} \quad (6\text{-}55)$$

where $\boldsymbol{C} = \beta^{-1}\boldsymbol{I} + \boldsymbol{\Phi}\boldsymbol{A}^{-1}\boldsymbol{\Phi}^T$.

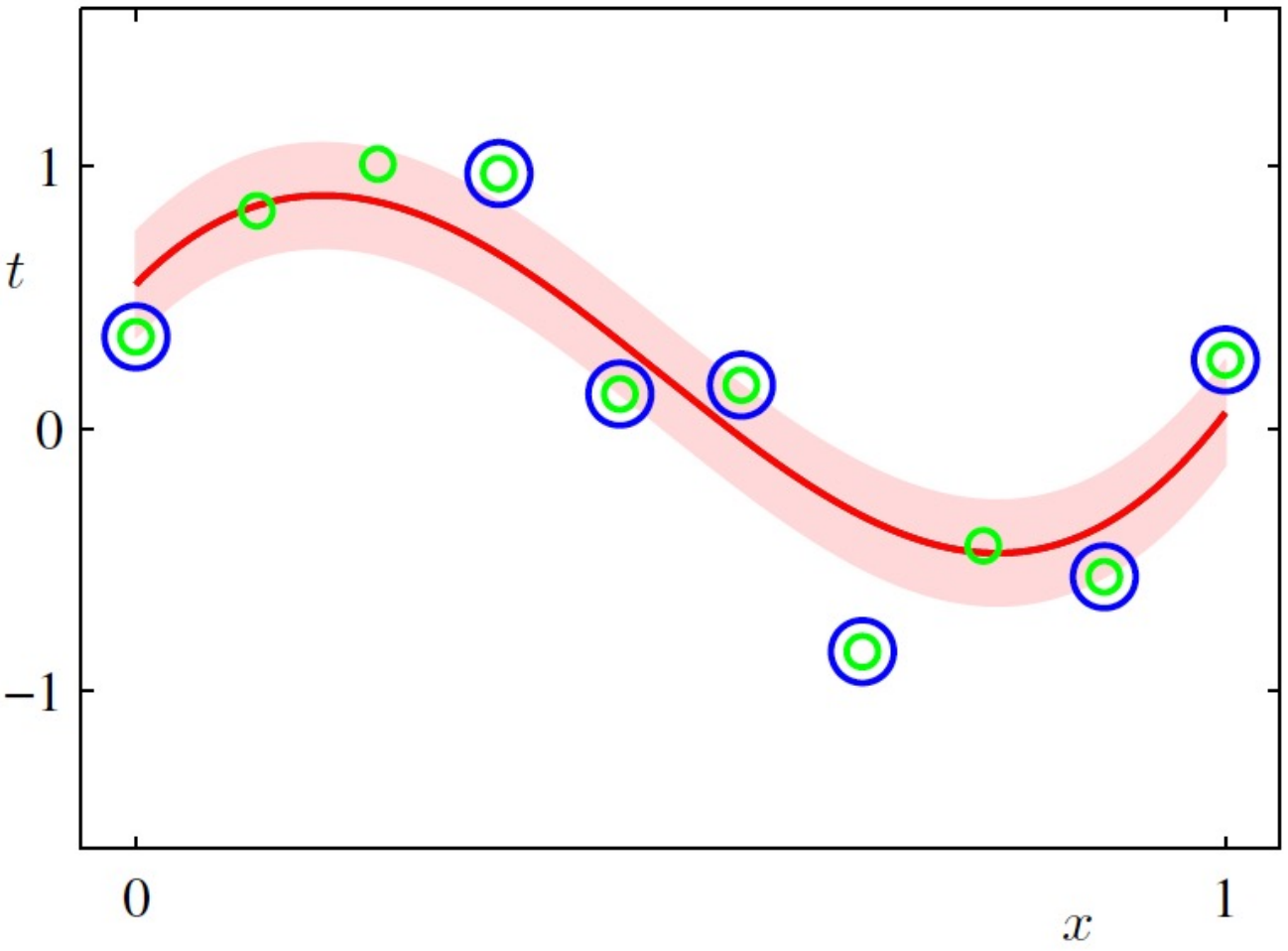
Set the required derivatives of the likelihood to zero and find the hyperparameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$:

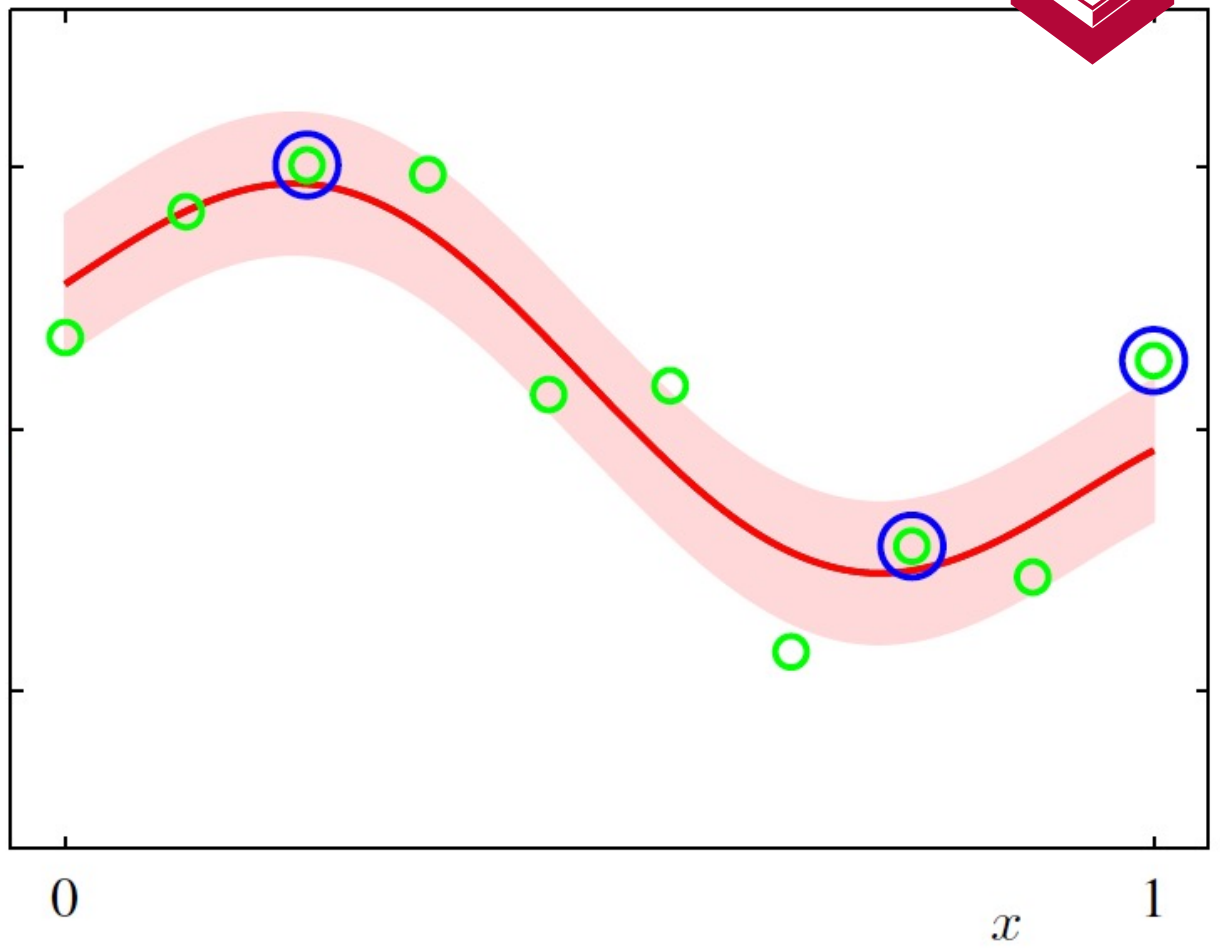
$$\alpha_i^{new} = \frac{\gamma_i}{m_i^2}$$

$$(\beta^{new})^{-1} = \frac{\left||\boldsymbol{t} - \boldsymbol{\Phi}\boldsymbol{m}|\right|^2}{N - \sum_i \gamma_i}$$

where m_i is the i^{th} component of the poster mean $\boldsymbol{m}$ and the quantity $\gamma_i = 1 - \alpha_i\Sigma_{ii}$ is the measurement of how well w_i is measured.

RVM - Regression





RVM - Regression

Linear Regression – the predictive variance becomes small in regions of input space.

SVM Regression - ϕ is centered on data points and the model will become increasingly certain of its predictions when extrapolating outside the domain of the data.

Gaussian Process – the computation cost is high.

RVM – A significant improvement in the speed of processing on test data. This greater sparsity is achieved with little or no reduction in generalization error compared with SVM.

But… Training involves optimizing a nonconvex function and takes longer.

RVM - Classification

Let's begin from the two-class problem with a binary target $t \in \{0,1\}$.

If the model takes the linear combination format of transformation function $\phi(\cdot)$ by a logistic sigmoid function

$$y(\boldsymbol{x}, \boldsymbol{w}) = \sigma\left(w^T \phi(\boldsymbol{x})\right).$$

The posterior distribution over w (assume it is a Gaussian prior) is

$$p(\boldsymbol{w}|\boldsymbol{t}) \propto p(\boldsymbol{w}) p(\boldsymbol{t}|\boldsymbol{w}).$$

We can solve for $\boldsymbol{\alpha}$ by following the way we saw in RVM for regression using Hassian matrix.

RVM - Classification

$$\ln p(\boldsymbol{w}|\boldsymbol{t},\boldsymbol{\alpha}) = \ln\{p(\boldsymbol{t}|\boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{\alpha})\} - \ln p(\boldsymbol{t}|\boldsymbol{\alpha})$$

$$= \sum_{n=1}^{N}\{t_n \ln y_n + (1-t_n)\ln(1-y_n)\} - \frac{1}{2}\boldsymbol{w}^T\boldsymbol{A}\boldsymbol{w} + const.$$

From logistic regression lecture,
Refer to equation 22.

Regularization comes from the prior in the previous slide,
$p(\boldsymbol{w}|\boldsymbol{t}) \propto p(\boldsymbol{w})p(\boldsymbol{t}|\boldsymbol{w})$

Note: $\ln p(w|t) = -\frac{1}{2}(\boldsymbol{w}-\boldsymbol{m}_0)^T\boldsymbol{S}_0^{-1}(\boldsymbol{w}-\boldsymbol{m}_0) + \sum_{n=1}^{N}\{t-y_n)\ln(1-y_n)\} + Const.$ when the Gaussian prior of **w** is $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{m_0},\boldsymbol{S}_0)$

From here, we can take the gradient of $\ln p(w|t,\alpha)$ be zero to obtain the mean and α_i^{new} is

$$\alpha_i^{new} = \frac{\gamma_i}{m_i^2}$$

Empirical Risk – Loss Functions

Empirical Risk Minimization

$$\min_{w} \frac{1}{n} \sum_{i=1}^{N} \underbrace{l(h_w(x_i), y_i)}_{\text{Loss}} + \underbrace{\lambda r(w)}_{\text{Regularization}}$$

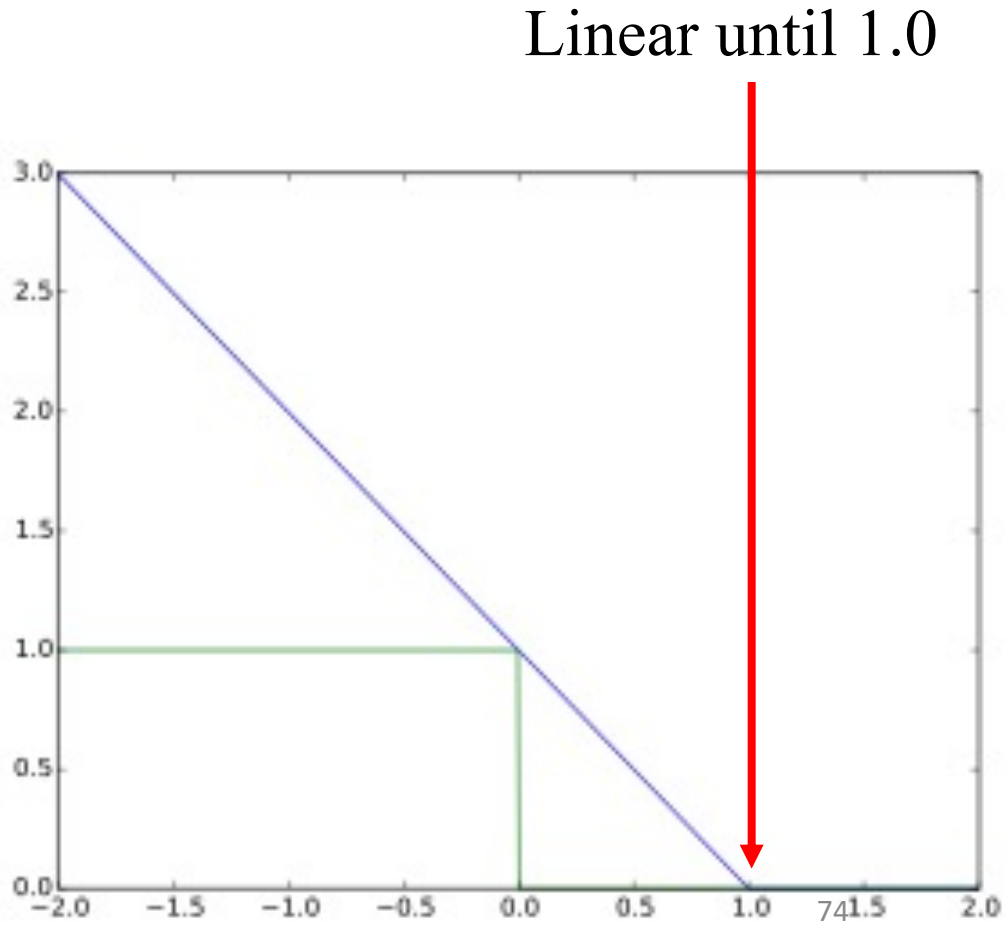
- The loss function – a continuous function penalizing training error
- The regularization – a continuous function penalizing classifier complexity

Classification: Hinge-Loss

$$\max[1 - h_w(x_i)y_i, 0]^p$$

- Usage: Supportive Vector Machine
- When $p = 1$, Standard SVM: the loss function denotes the size of the margin between linear separator and its closest points in either class.

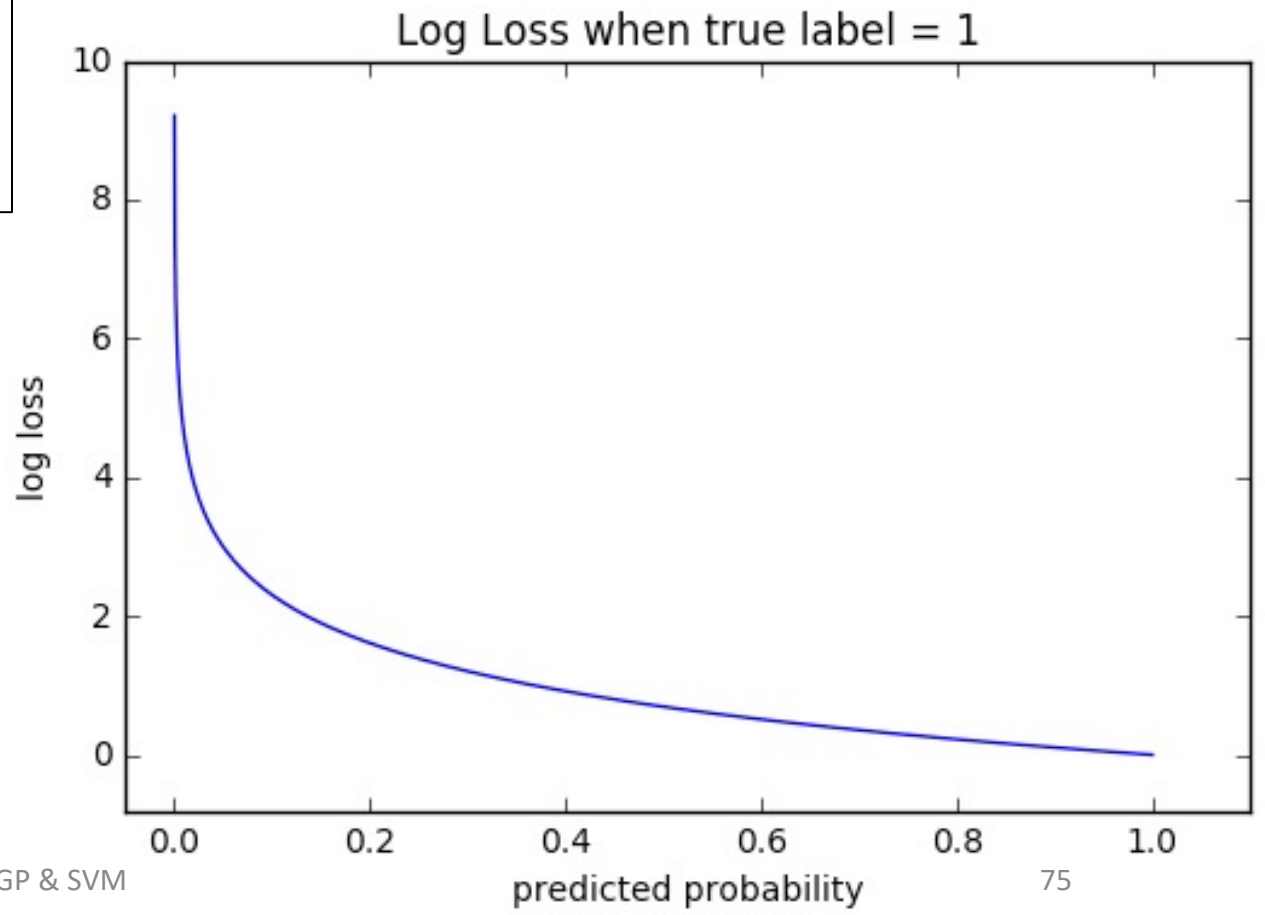
- When $p = 2$, Squared Hingeless SVM: Only differentiable everywhere with $p = 2$.



Classification: Log-Loss

$$\log\left(1 + e^{-h_w(x_i)y_i}\right)$$

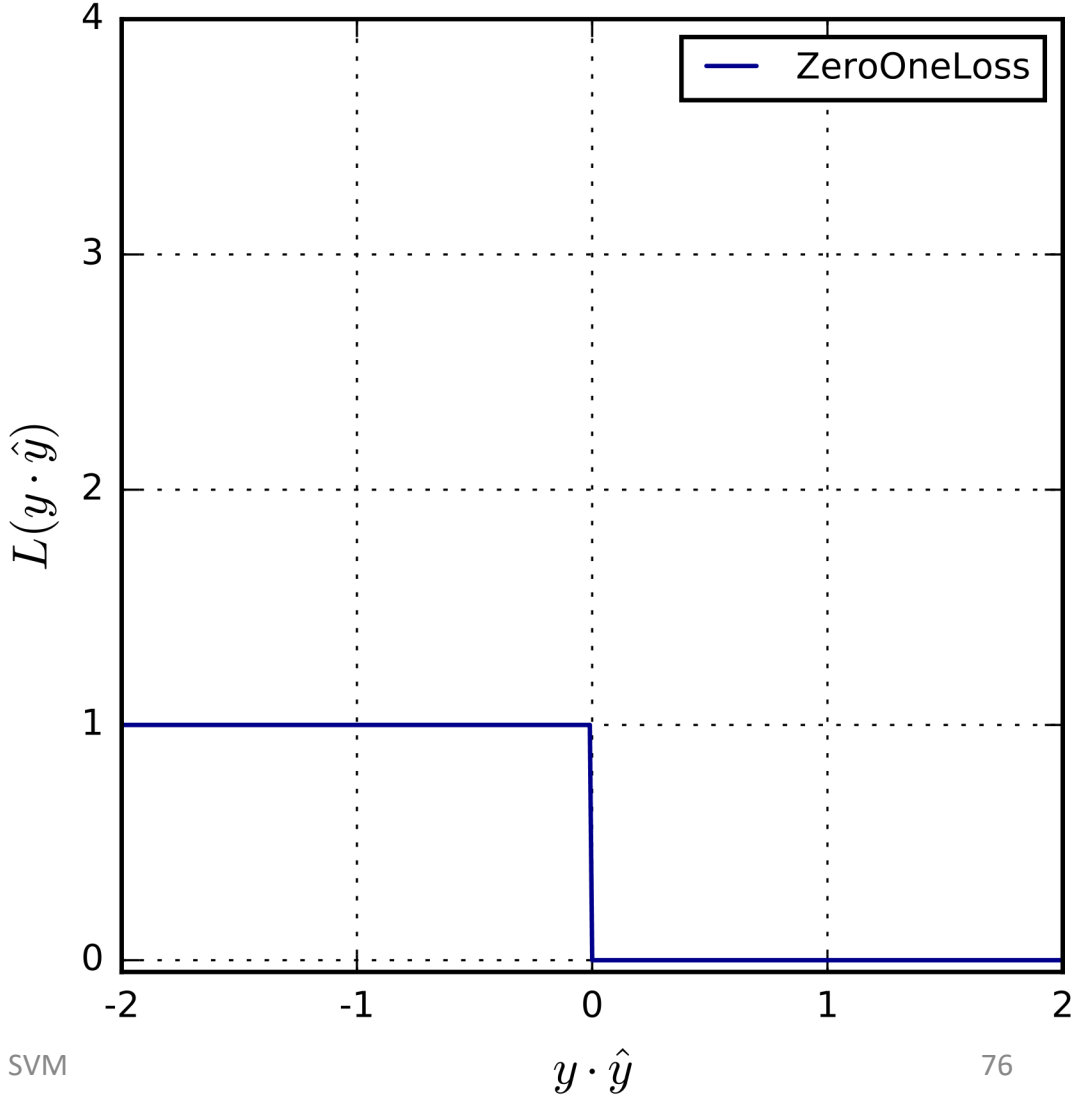
- Usage: Logistic Regression
- Its outputs are well-calibrated probabilities.
- Value range from 0 to 1.
- Goal: to minimize its value.



Classification: Zero-One Loss

$$\delta\big(sign\big(h_w(x_i)\big) \neq y_i\big)$$

- Usage: Actual Classification Loss
- Characteristic: Not convex nor continuous.



Loss Function: Classifications

Hinge-Loss:

$$\max[1 - h_w(x_i)y_i, 0]^p$$

- Usage: SVM
- When $p = 1$, Standard SVM: the loss function denotes the size of the margin between linear separator and its closest points in either class.
- When $p = 2$, Squared Hingeless SVM: Only differentiable everywhere with $p = 2$.

Log-Loss:

$$\log\left(1 + e^{-h_w(x_i)y_i}\right)$$

- Usage: Logistic Regression
- Its outputs are well-calibrated probabilities.
- Value range from 0 to 1.
- Goal: to minimize its value.

Zero-One Loss:

$$\delta\left(sign\left(h_w(x_i)\right) \neq y_i\right)$$

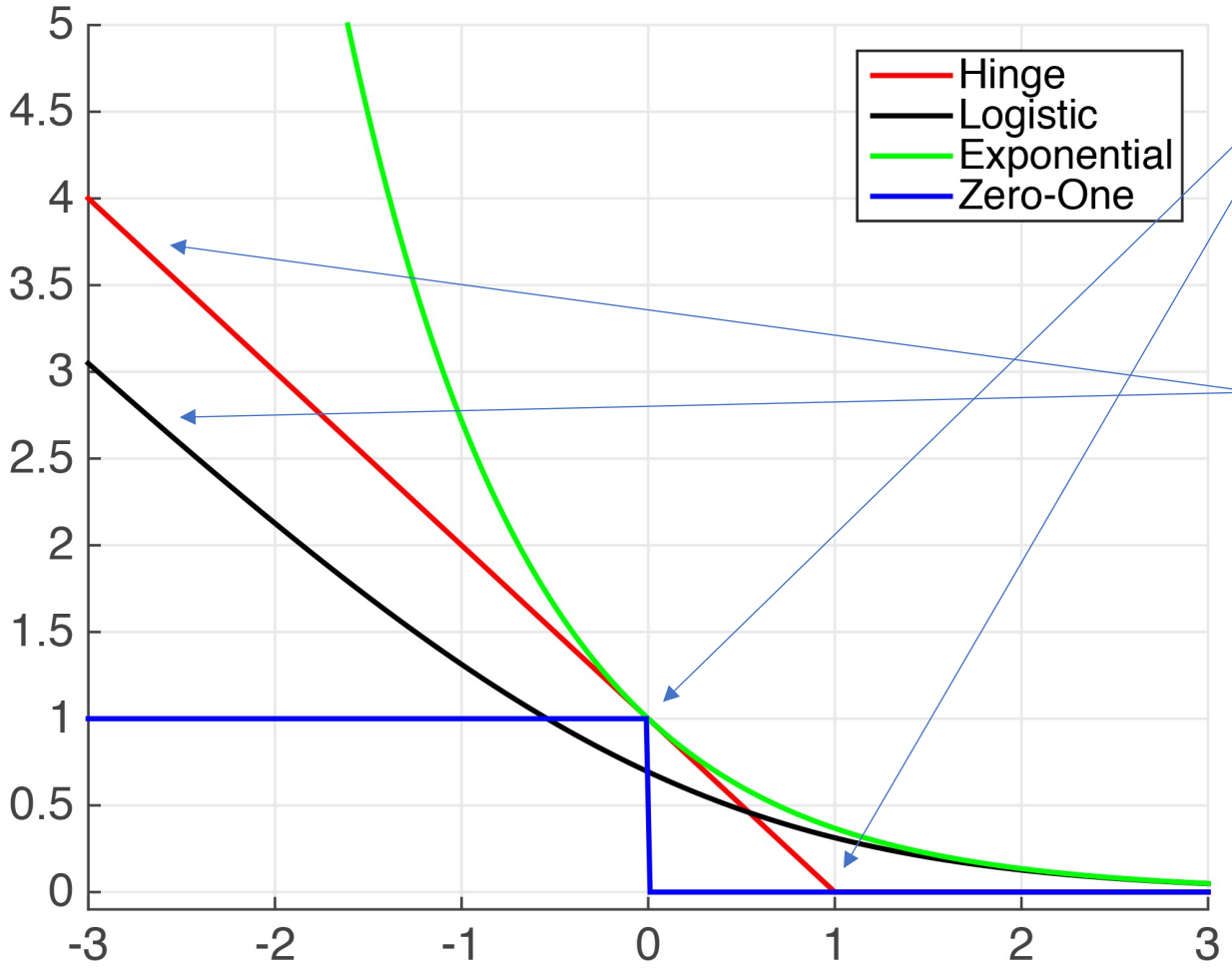
- Usage: Actual Classification Loss
- Characteristic: Not convex nor continuous.

Exponential Loss:

$$\exp(-h_w(x_i)y_i)$$

- Usage: AdaBoost
- Characteristic: Very aggressive
- Misprediction increases exponentially with the value of $-h_w(x_i)y_i$.
- May cause problems with noisy data

Classification Loss Functions Summary



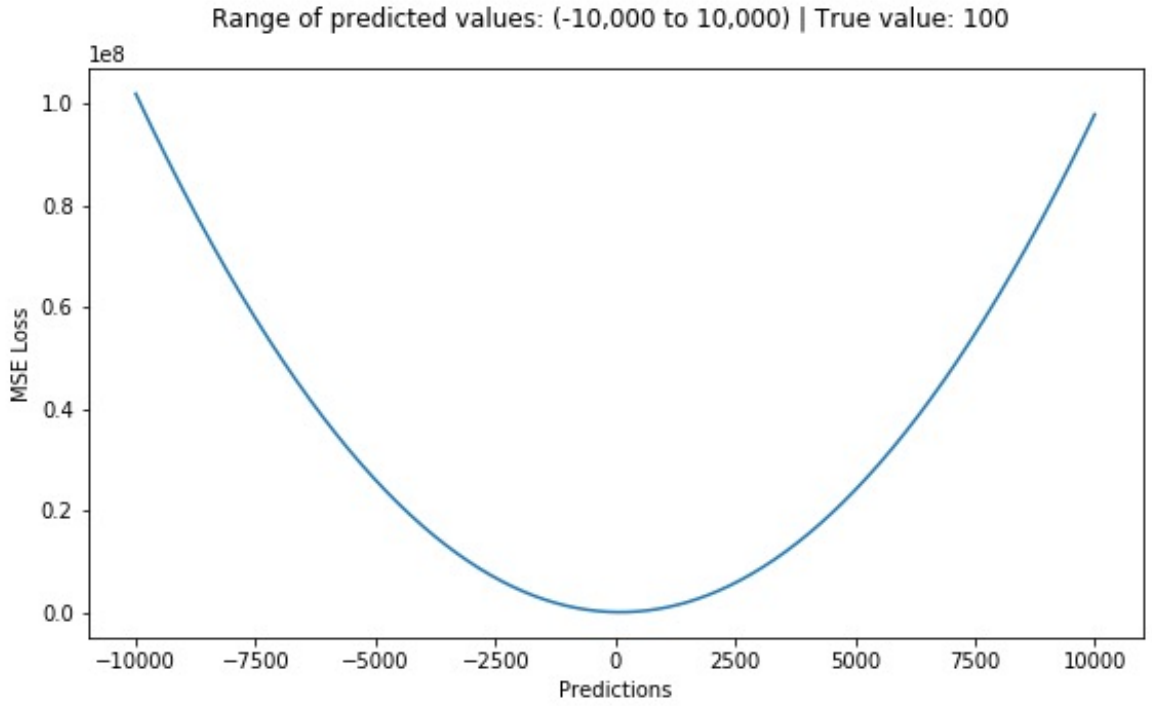
Which functions are strict upper bounds on the 0/1-loss?

What can you say about the hinge-loss and the log-loss as $z \to -\infty$?

Regression: Mean Square Loss

$$(h(x_i) - y_i)^2$$

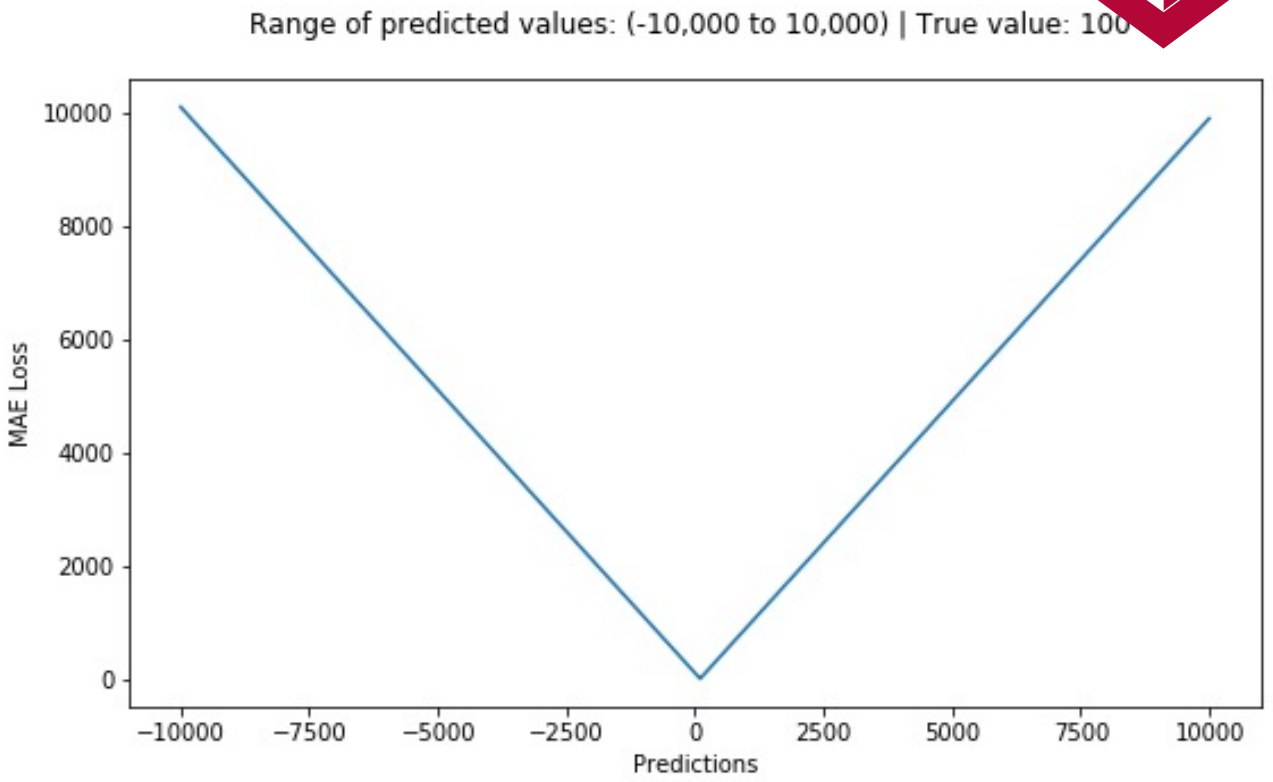
- Most popular regression loss function.
- Estimates Mean.
- It is differentiable everywhere.
- Sensitive to outliers.



Regression: Absolute Loss

$$|h(x_i) - y_i|$$

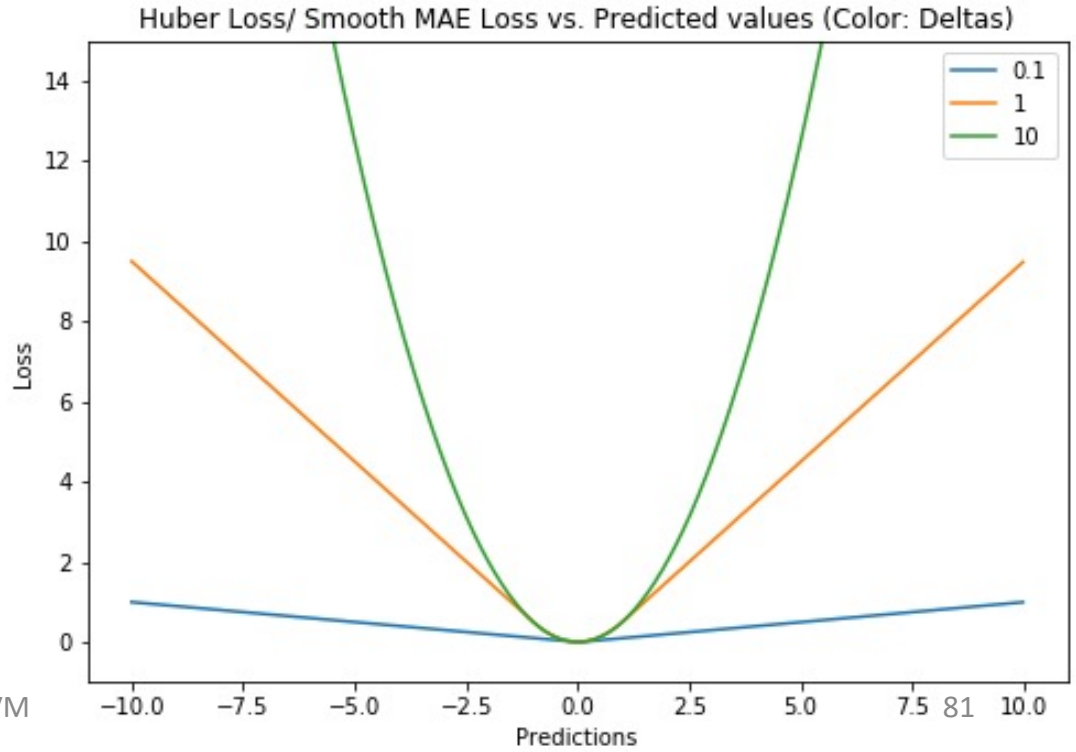
- Another Most popular regression loss function.
- Estimates Median.
- Less sensitive than squared loss function.
- Not differentiable at 0.



Regression: Huber Loss

$$\begin{cases} \frac{1}{2}(h(x_i) - y_i)^2, & if\ |h(x_i) - y_i| < \delta \\ \delta\left(|h(x_i) - y_i| - \frac{\delta}{2}\right), & Otherwise \end{cases}$$

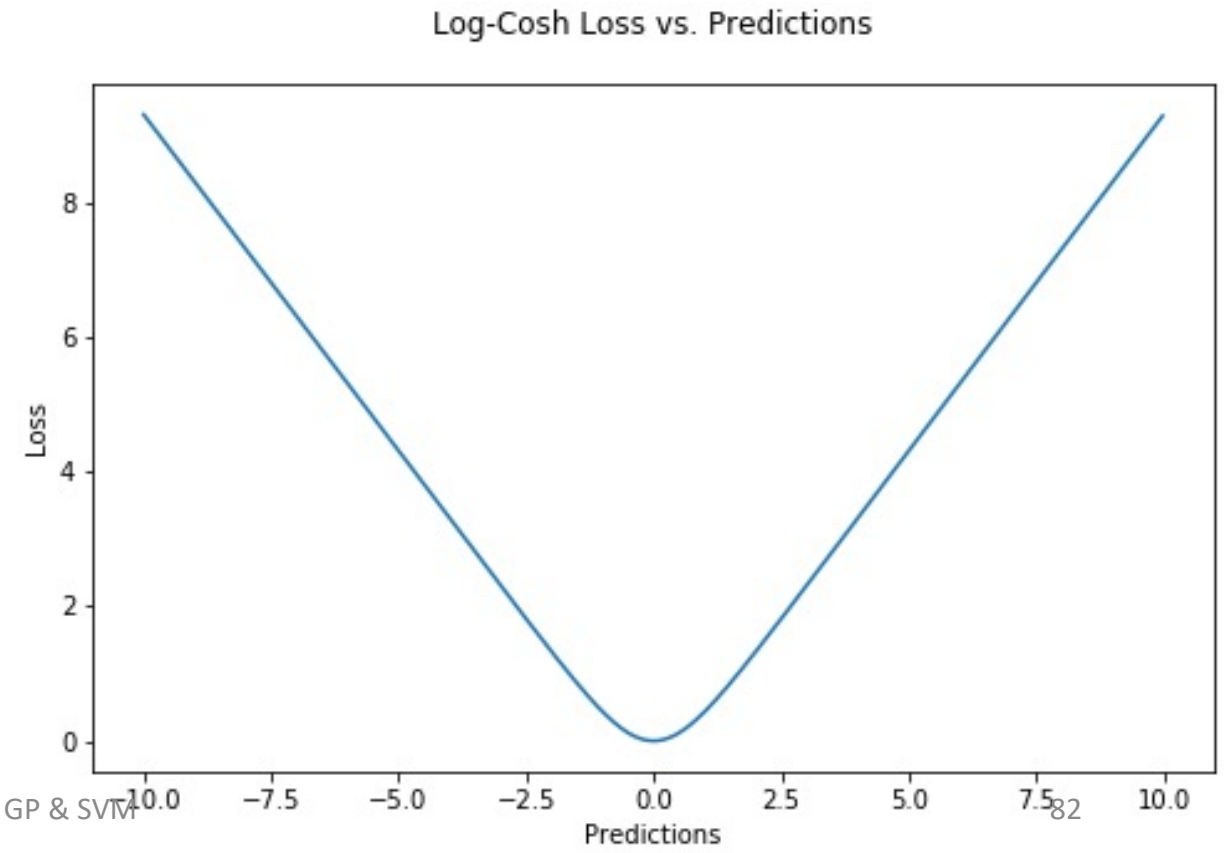
- Also known as Smooth Absolute Loss
- "Best of Both Worlds"
- Once-differentiable.
- When the loss is large, the squared loss.
- When the loss is small, the absolute loss.



Regression: Squared Loss

$$\log(\cosh(h(x_i) - y_i)), \cosh\left(\frac{e^x + e^{-x}}{2}\right)$$

- Similar to Huber Loss.
- Differentiable everywhere.



Loss Functions: Regressions

Squared Loss:

$$(h(x_i) - y_i)^2$$

- Most popular regression loss function.
- Estimates Mean.
- It is differentiable everywhere.
- Sensitive to outliers.

Absolute Loss:

$$|h(x_i) - y_i|$$

- Another Most popular regression loss function.
- Estimates Median.
- Less sensitive than squared loss function.
- Not differentiable at 0.

Huber Loss:

$$\begin{cases} \frac{1}{2}(h(x_i) - y_i)^2, & if\ |h(x_i) - y_i| < \delta \\ \delta(|h(x_i) - y_i| - \delta/2), & Otherwise \end{cases}$$

- Also known as Smooth Absolute Loss
- "Best of Both Worlds"
- Once-differentiable.
- When the loss is large, use absolute loss.
- When the loss is small, use squared loss.

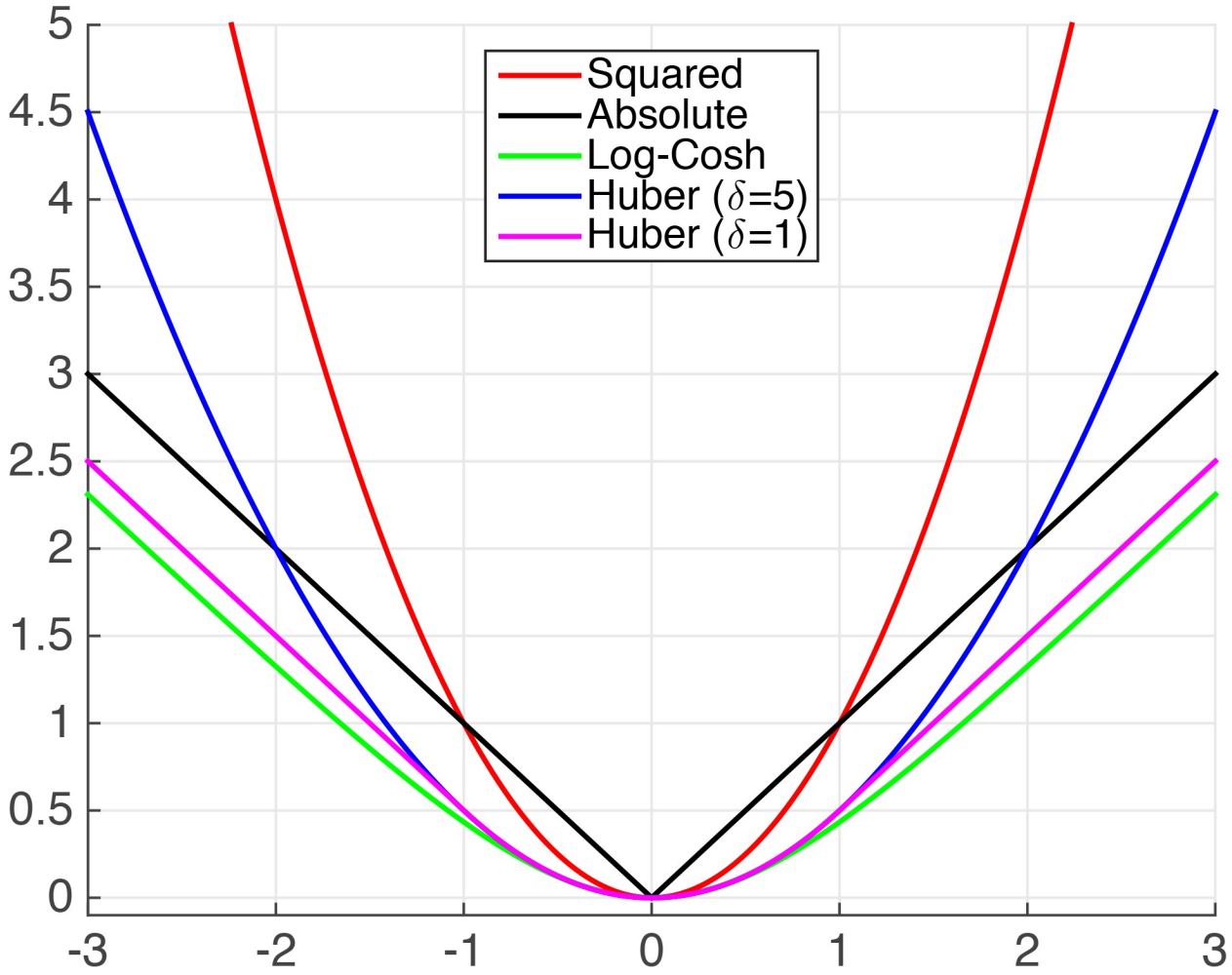
Log-Cosh Loss:

$$\log(\cosh(h(x_i) - y_i))$$

$$\cosh\left(\frac{e^x + e^{-x}}{2}\right)$$

- Similar to Huber Loss.
- Differentiable everywhere.

Regression: Loss Functions Summary



See how sensitive functions are!