EE769 Intro to ML

Linear Classification

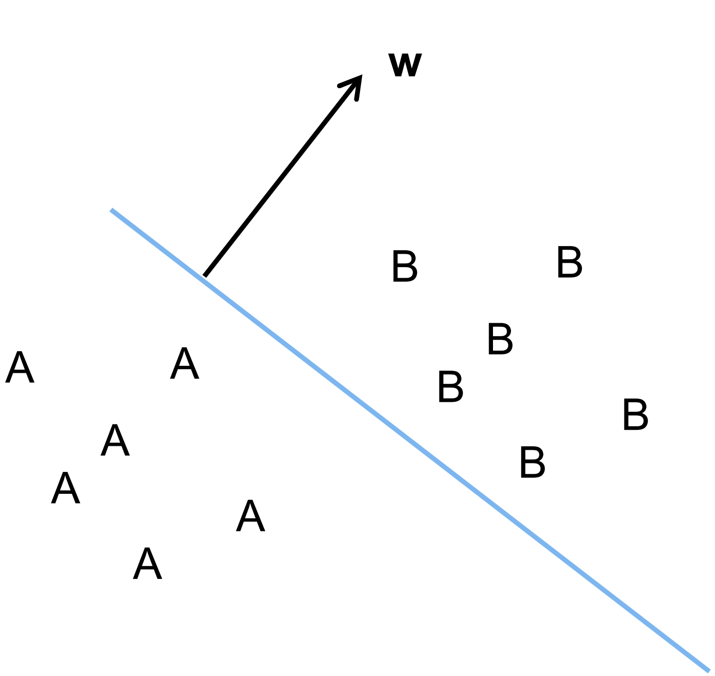
Amit Sethi
Faculty member, IIT Bombay

Learning objectives

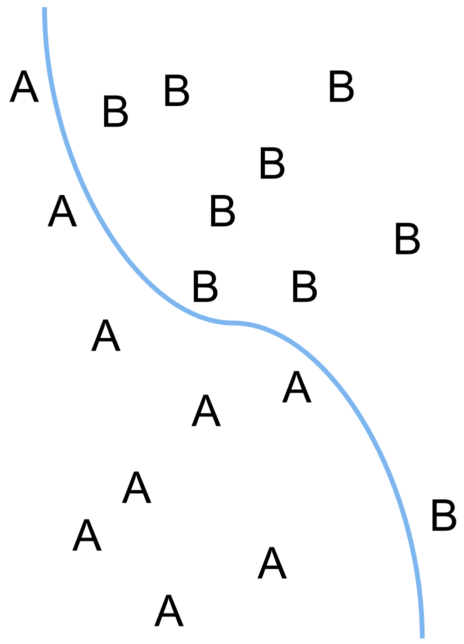
- Write the linear classification equation
- Write the Bayesian decision function
- Ground logistic regression in theory
- Derive gradient descent for logistic regression
- Derive the loss function for primal support vector machine

Linear classification function

- Class $y_i = \mathrm{Sign}(\mathbf{w}^T x_i + b)$ □ $\{-1,1\}$



Linear in **x**



Nonlinear in **x** (possibly linear in **ϕ**)

Why study linear classifiers?

- It is one of the simplest classifiers to analyze

- It seems to be a natural outcome for a familiar useful types of class conditional densities

- Many nonlinear problems can be linearized

- Multi-class classification can be modeled as a combination of several binary classification problems

Linearizing nonlinear problems

- Add derived features
  - Powers
  - Interaction terms
  - Kernels
- Extract features
  - Using pre-trained neural networks

Bayesian decision rule for classification

- Decision rule: $\boxed{\text{Class is 1, if } p(t=1|\mathbf{x}) > p(t=0|\mathbf{x})}$

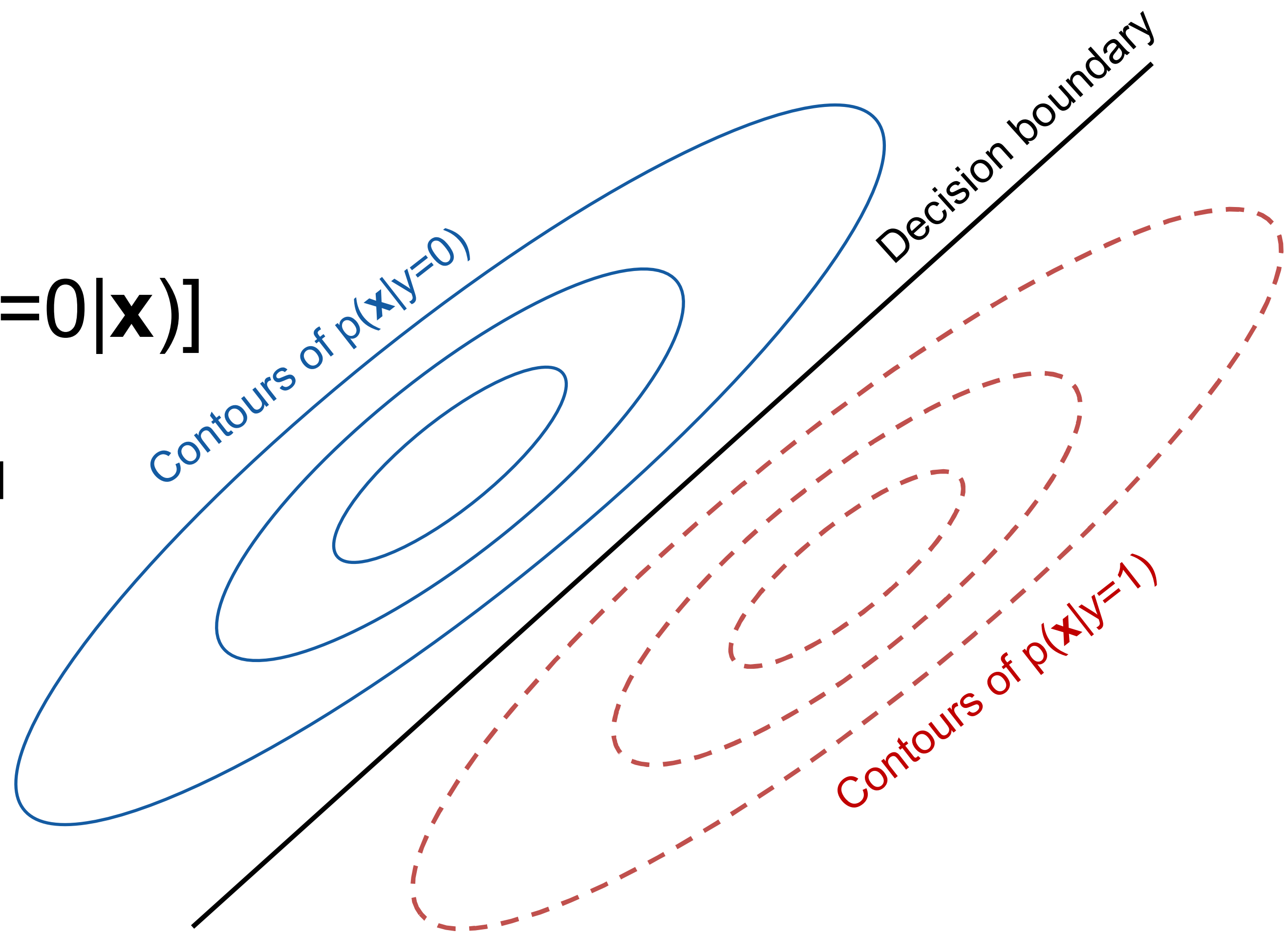
- Problem: Do not know how to model $p(t|\mathbf{x})$ directly

- Solution: Bayes rule $p(t|\mathbf{x}) = p(\mathbf{x}|t) \, . \, p(t) \, / \, p(\mathbf{x})$
- Posterior = Likelihood . prior / marginal
- Marginal is unknown, but common to both classes
- Class is 1, if $p(\mathbf{x}|t=1)\, p(t=1) > p(\mathbf{x}|t=0)\, p(t=0)$

Gaussian class conditionals in with the same covariance matrix

- $p(\mathbf{x}|t=j) = N(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$
- $p(t=j)$ □ $[0,1]$
- $\log [p(t=1|\mathbf{x}) / \log p(t=0|\mathbf{x})]$

  is linear in $\mathbf{x}$, if $\boldsymbol{\Sigma}_0 = \boldsymbol{\Sigma}_1$



Gaussian class conditionals in with the same covariance matrix

Derivation:

- To check if $p(t=1|\mathbf{x}) > p(t=0|\mathbf{x})$
- Check if $p(t=1|\mathbf{x}) / p(t=0|\mathbf{x}) > 1$
- $p(\mathbf{x}|t=1)\, p(t=1) / p(\mathbf{x}|t=0)\, p(t=0) > 1$
- $\log[p(\mathbf{x}|t=1)] + \log p(t=1) - \log[p(\mathbf{x}|t=0)] - \log p(t=0) > 0$
- $\log[\exp(-(\mathbf{x}-\boldsymbol{\mu}_1)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_1))] - \log[(2\pi)^{-d/2}\det(\boldsymbol{\Sigma})^{-1/2}] + \log p(t=1) -$
  $\log[\exp(-(\mathbf{x}-\boldsymbol{\mu}_0)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_0))] - \log[(2\pi)^{-d/2}\det(\boldsymbol{\Sigma})^{-1/2}] - \log p(t=0) > 0$
- $-(\mathbf{x}-\boldsymbol{\mu}_1)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_1) + \log p(t=1) + (\mathbf{x}-\boldsymbol{\mu}_0)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_0) - \log p(t=0) > 0$
- $-\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x} - \boldsymbol{\mu}_1^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 + 2\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 + \mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x} + \boldsymbol{\mu}_0^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_0 - 2\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_0$
  $+ \log[p(t=1)/p(t=0)] > 0$
- $[2\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_0)]^T\mathbf{x} + [\boldsymbol{\mu}_0^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 + \log[p(t=1)/p(t=0)]\,] > 0$
- $\mathbf{w}^T\mathbf{x} + b > 0$

Algorithm to build a Bayesian classifier

- Count number of samples n_j in each class j
- Estimate priors p(t=j) as n_j / N, where N = $\boldsymbol{\Sigma}_j n_j$
- Estimate class conditionals p(**x**|t=j), e.g. Gaussian
  - μ_j is the sample mean for class j
  - Σ_j is the sample covariance matrix for class j
- Decision rule: Class is 1, if

$$\log p(\mathbf{x}|t=1) + \log p(t=1) - \log p(\mathbf{x}|t=0) - \log p(t=0) > 0$$

Gradient descent for linear classifier

Loss

- Can use $(p(t_i|\mathbf{x}_i) - t_i)^2$, but we do not because this error is not Gaussian
- We use $D_{KL}\ (t_i \ || \ p(t_i|\mathbf{x}_i)) = -\sum_j \mathbf{1}_{t=j} \log[p(t_i=j|\mathbf{x}_i)/\mathbf{1}_{t=j}]$

- Which is BCE

$-[t_i \log p(t_i=j|\mathbf{x}_i) + (1-t_i) \log(1-(p(t_i=j|\mathbf{x}_i))]$

$-[t_i \log \sigma(h) + (1-t_i) \log(1-(\sigma(h))]$,

where $\sigma(h) = 1 / [1+\exp(-h)]$;

$h = \text{log of odds ratio} = \log [p(t_i=1|\mathbf{x}_i)/p(t_i=0|\mathbf{x}_i)]$

$= \mathbf{w}^T\mathbf{x}_i+b$

Gradient descent using BCE

- $h_i = \log[\ p(t_i=1|\mathbf{x}_i) \ /\ p(t_i=0|\mathbf{x}_i)\ ] = \mathbf{w}^T\mathbf{x}_i+b$
- $y_i = p(t_i=1|\mathbf{x}_i) = \sigma(h_i) = 1 \ /\ [1+\exp(-h_i)]$
- $\text{BCE} = L_i = -\ t_i \log y_i - (1-t_i) \log (1-y_i)$
- $\partial L_i/\partial w_k = \partial L_i/\partial y_i \ .\ \partial y_i/\partial h_i \ .\ \partial h_i/\partial w_k$

$$= [t_i/y_i - (1-t_i)/(1-y_i)] \ .\ y_i \ .\ (1-y_i) \ .\ x_{i,k}$$

$$= [(y_i-t_i) \ /\ y_i \ /\ (1-y_i)] \ .\ y_i \ .\ (1-y_i) \ .\ x_{i,k}$$

$$= (y_i-t_i) \ .\ x_{i,k}$$

$$\mathbf{w}_{\text{new}} = \mathbf{w}_{\text{old}} - \eta \textstyle\sum_i (y_i-t_i)\ \mathbf{x}_i$$

Adding regularization

- $L = -\sum_i [t_i \log y_i + (1-t_i) \log (1-y_i)] + \lambda \sum_k |w_k|^q$
- $\mathbf{w}_{new} = \mathbf{w}_{old} - \eta\, [\, \sum_i (y_i - t_i)\, \mathbf{x}_i + 2\lambda\, \mathbf{w}_{old} \,]$, for $q=2$
- $\mathbf{w}_{new} = \mathbf{w}_{old} - \eta\, [\, \sum_i (y_i - t_i)\, \mathbf{x}_i + \lambda\, \mathrm{sign}(\mathbf{w}_{old})]$, for $q=1$

Detour -- Elastic Net

- Ridge (or L2 regularization or weight decay)
  - Minimize: $L_{error} + \lambda_2 \|w\|_2^2$
  - **Grouping effect** on correlated variables
  - Encourages two correlated variables to have the same weight
- LASSO (or L1 regularization)
  - Minimize: $L_{error} + \lambda_1 \|w\|_1$
  - May **eliminate variables**
  - Does not encourage two correlated variables to have the same weights
- Elastic net (or L1+L2 regularization) has both effects
  - Minimize: $L_{error} + \lambda_1 \|w\|_1 + \lambda_2 \|w\|_2^2$
  - May eliminate groups of correlated variables

Asymmetric risk

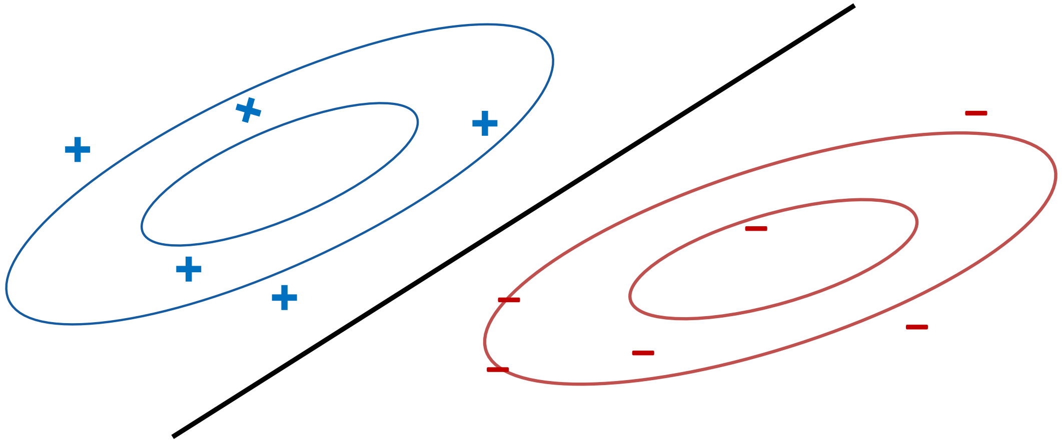
- Some risks are not symmetric
  - Calling a healthy person sick vs. vice versa
- We need a risk matrix
  - Perhaps, no risk for correct calls
  - But, different risks for Type I (FP) vs. Type II (FN) errors
- Minimize expected risk

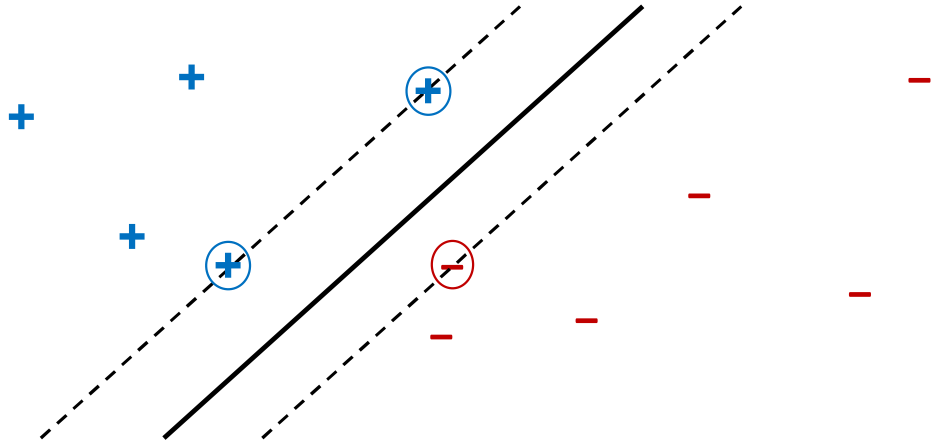
Some metrics for binary classification

- For a single threshold
  - Accuracy = (TP + TN) / (TP + TN + FP + FN)
  - Precision = TP / (TP + FP)
  - Recall, Sensitivity = TP / (TP + FN)
  - Specificity = TN / (TN + FP)
  - PPV, NPV, FDR, FOR etc.
  - Balanced metric: F1 score = 2 × precision × recall / (precision + recall)
- For all thresholds
  - Receiver operating characteristic (ROC) curve: Plot of sensitivity (y-axis) versus (1 − specificity) by varying decision threshold
  - Area under curve (AUC): area under ROC (from 0 to 1)

SVM for distribution-free learning

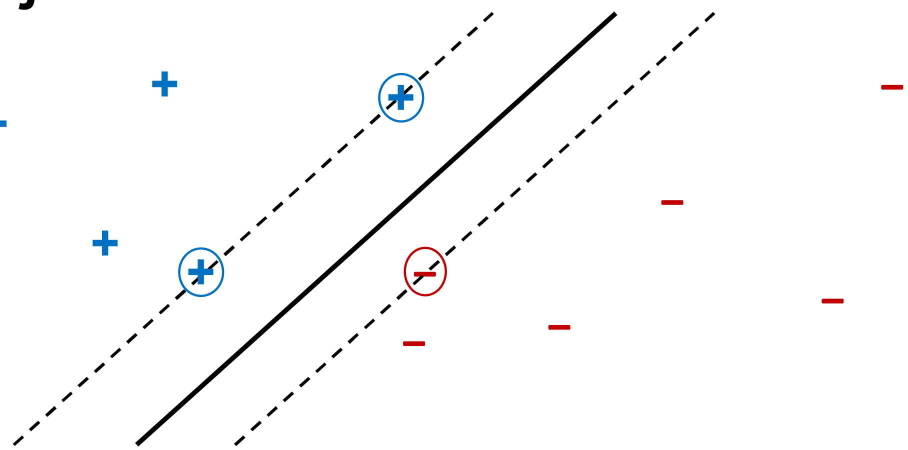
- Empirical risk: risk of misclassifying trianing data
- How to minimize empirical risk?
- How to pick the “best” among multiple solutions?
- Depends upon the assumptions:
  - Bayesian: Minimize expected risk (by assuming pdf)
  - SVM: Minimize structural risk => margin maximization





SVM maximizes the separating margin

1. $\text{Max}_{\mathbf{w},b}\, [\min_i ||\mathbf{w}^T\mathbf{x}_i+b||]$, subject to:
   a) $||\mathbf{w}|| = 1$
   b) $\forall\ i,\ t_i\,(\mathbf{w}^T\mathbf{x}_i+b) \geq 0$



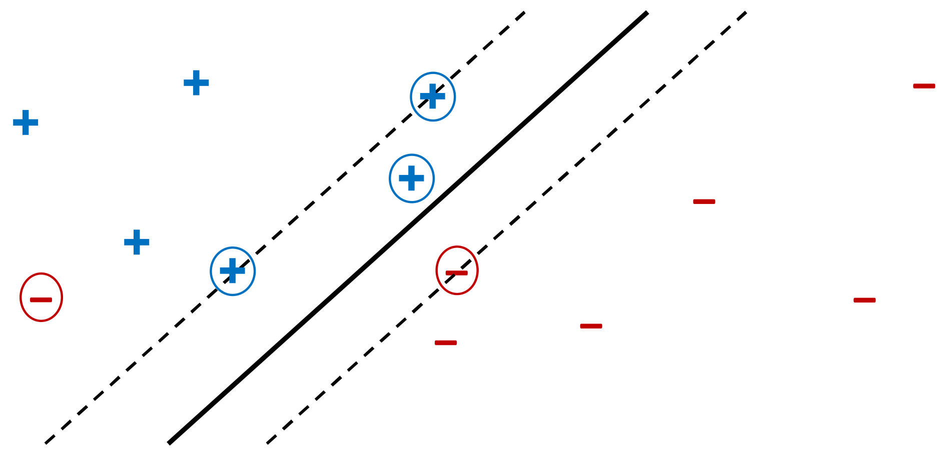
2. $\text{Min}_{\mathbf{w},b}\, ||\mathbf{w}||^2$, s.t.
   a) $\forall i,\ t_i\,(\mathbf{w}^T\mathbf{x}_i+b) \geq 1$

3. $\text{Min}_{\mathbf{w},b} \sum_i \max[0, 1 - t_i\,(\mathbf{w}^T\mathbf{x}_i+b)] \quad + \lambda\, ||\mathbf{w}||; \quad \lambda \rightarrow 0.$
   $\text{Min}_{\mathbf{w},b} \sum_i [1 - t_i\,(\mathbf{w}^T\mathbf{x}_i+b)]_+ \quad + \lambda\, ||\mathbf{w}||; \quad \lambda \rightarrow 0.$
   $\text{Min}_{\mathbf{w},b} \sum_i \ell_{hinge}(t_i, \mathbf{x}_i, \mathbf{w}, b) \quad + \lambda\, ||\mathbf{w}||; \quad \lambda \rightarrow 0.$

Soft-margin SVM formulation

- Or, minimize $||\mathbf{w}||^2 + C \sum_i \xi_i$
- ξ_i are slack variables
- Subject to $\forall\ i,\ t_i\,(\mathbf{w}^T\mathbf{x}_i) \geq 1-\xi_i$

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

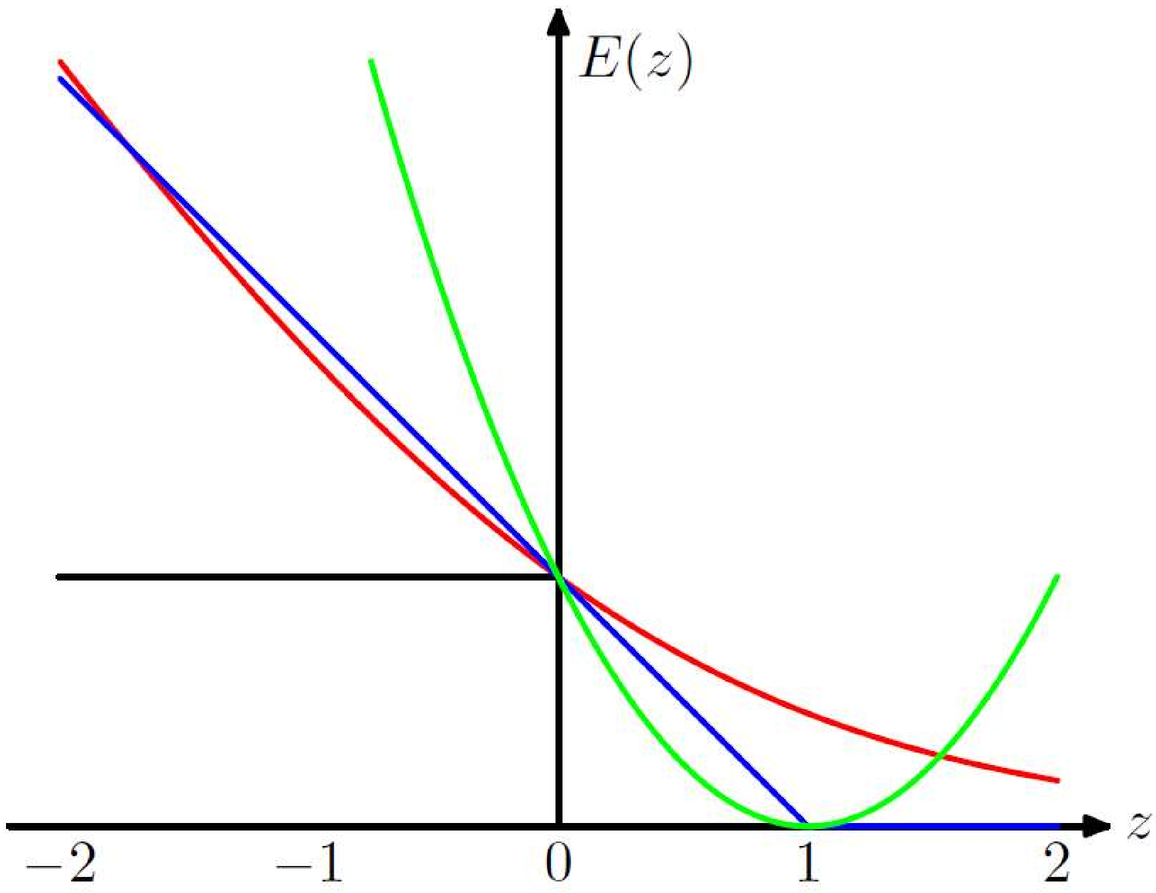


- $\mathrm{Min}_{\mathbf{w},b} \sum_i \ell_{\mathrm{hinge}}(t_i,\mathbf{x}_i,\mathbf{w},b) + \lambda\, ||\mathbf{w}||;\quad \lambda \geq 0.$

What are support vectors and slack variable?

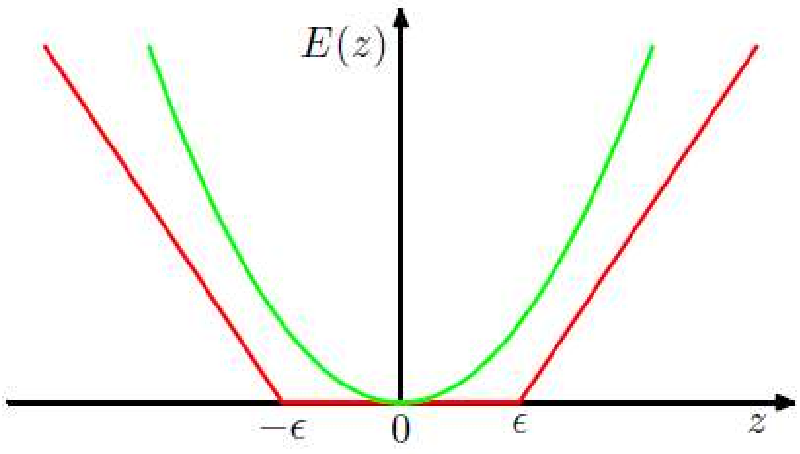
- In hard SVM
  - SVs define the hyperplane
  - They are closest to the hyperplane
  - There is a sparse set of such points
  - All other points don't matter
- In soft SVM
  - All missclassified points are SV too!
  - All points with ξ_i>0 or just at the cusp
- Values of ξ_i
  - Correct side of margin = 0
  - On the margin = 0
  - Inside the margin > 0
  - On the boundary = 1
  - Misclassified > 1

Comparison of loss functions



Source: PRML book by Bishop

Training data sparsity in regression using SVMs



- We change the error function from:

$$\frac{1}{2}\sum_{n=1}^{N}\{y_n - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

- To a ε-insensitive (fault-tolerant) error function

$$C\sum_{n=1}^{N} E_\epsilon(y(\mathbf{x}_n) - t_n) + \frac{1}{2}\|\mathbf{w}\|^2$$

where, $$E_\epsilon(y(\mathbf{x}) - t) = \begin{cases} 0, & \text{if } |y(\mathbf{x}) - t| < \epsilon; \\ |y(\mathbf{x}) - t| - \epsilon, & \text{otherwise} \end{cases}$$

Source: PRML book by Bishop

Now we define slack variables

$$t_n \leqslant y(\mathbf{x}_n) + \epsilon + \xi_n$$
$$t_n \geqslant y(\mathbf{x}_n) - \epsilon - \widehat{\xi}_n.$$

$$\xi_n \geqslant 0 \text{ and } \widehat{\xi}_n \geqslant 0$$

- Cost function can be written as:

$$C \sum_{n=1}^{N} (\xi_n + \widehat{\xi}_n) + \frac{1}{2}\|\mathbf{w}\|^2$$

Source: PRML book by Bishop