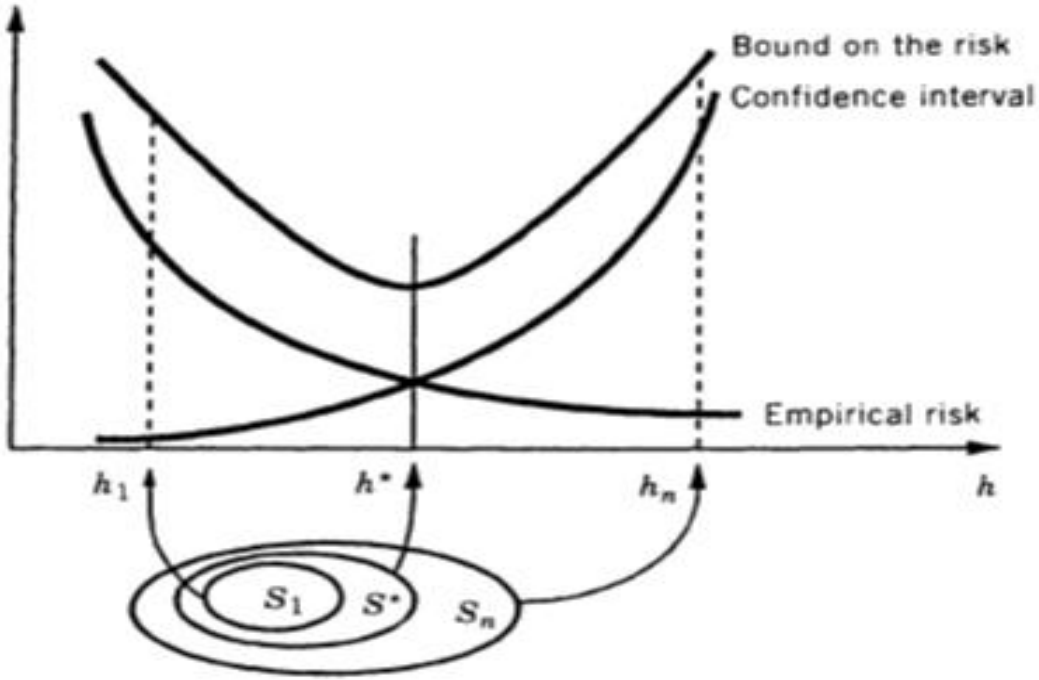
Support Vector Network / Machine

Risk Minimization



Bound on the risk = Empirical risk + Confidence interval

As h increases Empirical risk decreases and Confidence interval increases

h is VC dimension

Contd…

- Empirical Risk Minimization(ERM): Keep the confidence interval fixed (chosen a priori) while minimizing empirical risk.
- Structural Risk Minimization(SRM): Minimize both the confidence interval and empirical risk simultaneously.
- ANN uses ERM principle.
- SVM uses SRM principle.

Disadvantages of ANN

ANN requires tunable parametres:

- Number of neurons in the hidden layer
- Momentum constant
- Learning rate parameter
- Number of iterations to converge

Support Vector Machine

- SVM uses supervised learning algorithm.
- SVMs are used both for classification and regression analysis.
- Basic working principle: SVM defines a hyper-plane or a set of hyper-planes in a higher dimensional space.
- For a two class data, SVM selects a set of input data to predict for each given input which of the two possible classes the output belongs.

  Two class data are divided into
  - Separable data
  - Non-separable data
  - Nonlinear transformation with kernels

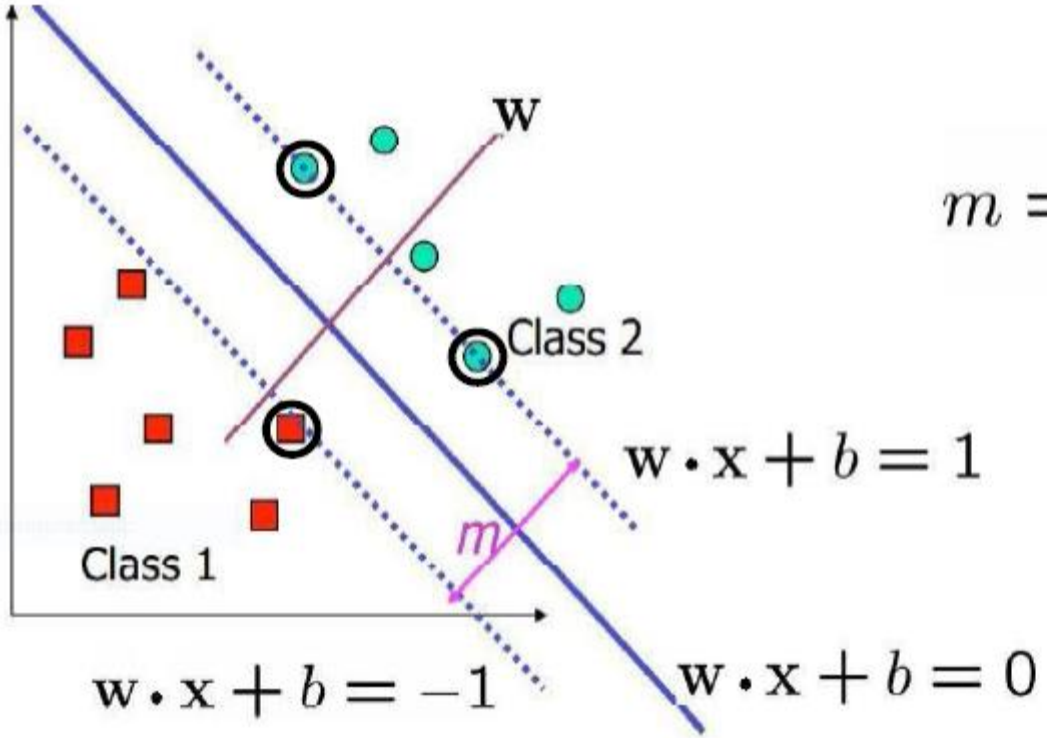
Separable Data

- Consider a two class, linearly separable classification problem
- The decision boundary should be as far away from the data of both classes as possible
  - We should maximize the margin, m
  - Perpendicular distance between the origin and the hyperplane (line) $\mathbf{w}.\mathbf{x} + b = 0$ is $|b|/||\mathbf{w}||$ where $\mathbf{w}$ is normal to the hyperplane, b is constant and $||\mathbf{w}||$ is the Euclidean norm of $\mathbf{w}$.
- Let $\{ x_1, x_2, \ldots, x_n \}$ be our data set and let $y_i \in \{ -1, 1\}$ be the class level of x_i

  $\mathbf{w}.\mathbf{x}_i + b \geq +1$ for $y_i = +1$

  $\mathbf{w}.\mathbf{x}_i + b \leq -1$ for $y_i = -1$
- The decision boundary should classify all points correctly $y_i (\mathbf{w}.\mathbf{x}_i + b) -1 \geq 0 \quad \forall i$

Separable Data

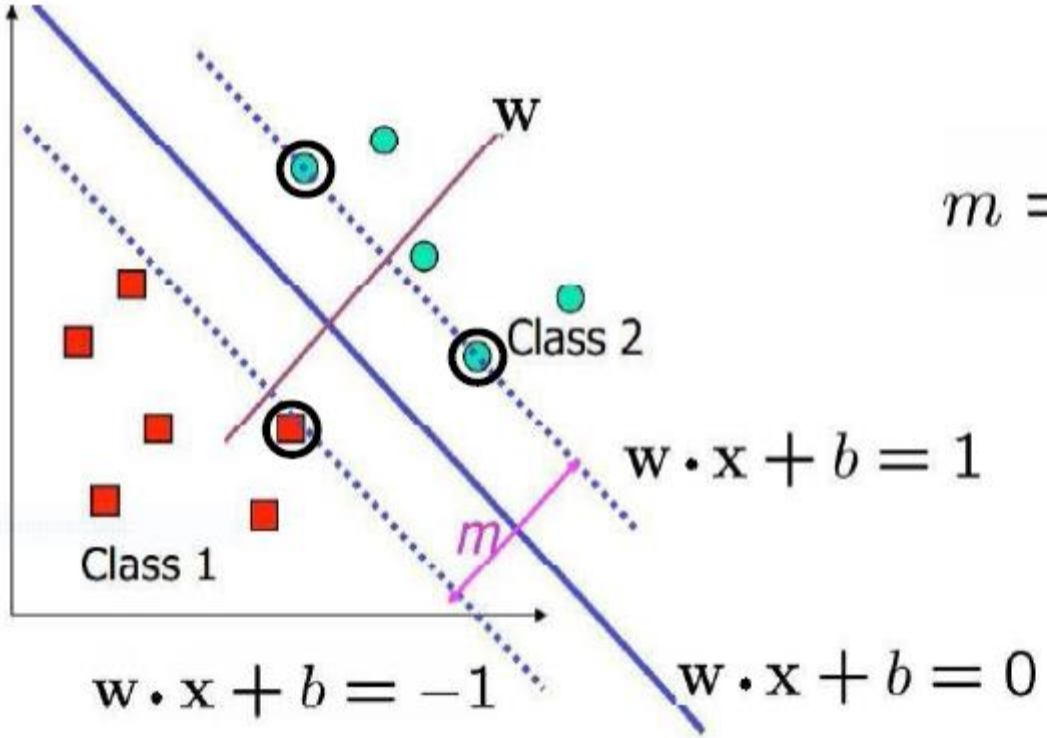


$$m = \frac{2}{||\mathbf{w}||}$$

Figure 1: Linear separating hyper-plane for separable data. The support vectors are circled.

Separable Data

$$m = \frac{|1-b| - |-1-b|}{||w||}$$



$$m = \frac{2}{||\mathbf{w}||}$$

Figure 1: Linear separating hyper-plane for separable data. The support vectors are circled.

Separable Data

The decision boundary can be found by solving the following constrained optimization problem

For the separable case, the objective function for the optimization problem is:

Minimize $\{0.5 \| \mathbf{w} \|^2 \}$

subject to $y_i (\mathbf{w} \cdot \mathbf{x_i} + b) - 1 \geq 0$

for $1 \leq i \leq n$

>>

Nonseparable Data

We allow "error" ξ_i in classification; it is based on the output of the discriminant function $\mathbf{w} \cdot \mathbf{x} + b$

- ξ_i approximates the number of misclassified samples.

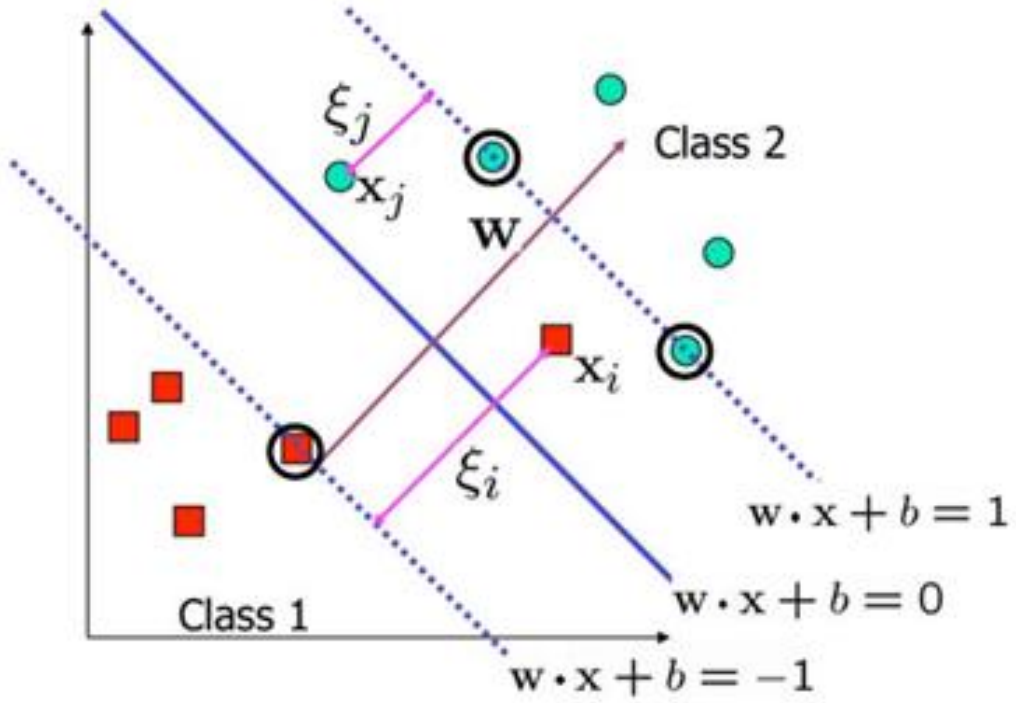


Figure 2: Linear separating hyper-plane for non-separable data. The support vectors are circled.

Nonseparable Data

For the non-separable case, the objective function for the optimization problem modifies to:

$$\text{Minimize } \{0.5 \,||\, \mathbf{w} \,||^2 + C \sum_{i=1}^{n} \xi_i \}$$

subject to $y_i (\mathbf{w} \cdot \mathbf{x_i} + b) \geq 1 - \xi_i$

for $i = 1, 2, 3, \ldots\ldots, n$ and $\xi_i \geq 0$

C: tradeoff parameter between error and margin

Nonlinear Transformation with Kernels

- Original problem is stated in a finite dimensional space, and the data sets to be discriminated are not linearly separable in that space.

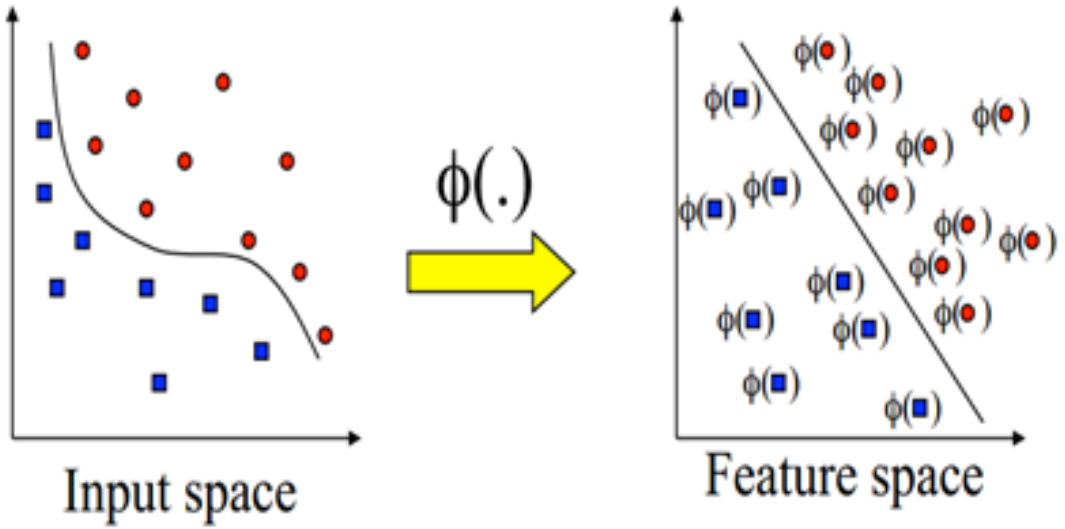


Figure 3: Original finite dimensional input space is mapped into a higher dimensional feature space

Contd…

- Key idea: transform $\mathbf{x}_i$ to a higher dimensional space
  - Input space: the space the point $\mathbf{x}_i$ are located
  - Feature space: the space of $\varphi(\mathbf{x}_i)$ after transformation. It does not need to represent the space explicitly, simply by defining a kernel function.
- Computation in the feature space can be costly because it is high dimensional
  - The feature space is typically infinite-dimensional!
- The kernel trick comes to rescue
- In the dual SVM optimization problem the data points only appear as dot product
- As long as we can calculate the dot product in the feature space, we do not need the mapping explicitly

Contd…

- The kernel function plays the role of the dot product in the feature space.

- Define the kernel function K by

$$K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)$$

- The linear classifier relies on dot product between vectors $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j$

- This use of kernel function to avoid carrying out $\varphi(.)$ explicitly is known as the kernel trick

Some commonly used kernels

Polynomial: $K(\mathbf{x},\mathbf{y}) = (\mathbf{x}.\mathbf{y}+1)^p$

Gaussian radial basis function: $K(\mathbf{x},\mathbf{y}) = e^{-\|\mathbf{x}-\mathbf{y}\|^2/2\sigma^2}$

Neural net: $K(\mathbf{x},\mathbf{y}) = \tanh(k\,\mathbf{x}.\mathbf{y} - \delta)$

Parameters that the user must choose

SVM Applications

- SVM has been used successfully in many real-world problems
  - text (and hypertext) categorization
  - image classification
  - bioinformatics (Protein classification, Cancer classification)
  - hand-written character recognition

Comparison with Neural Networks

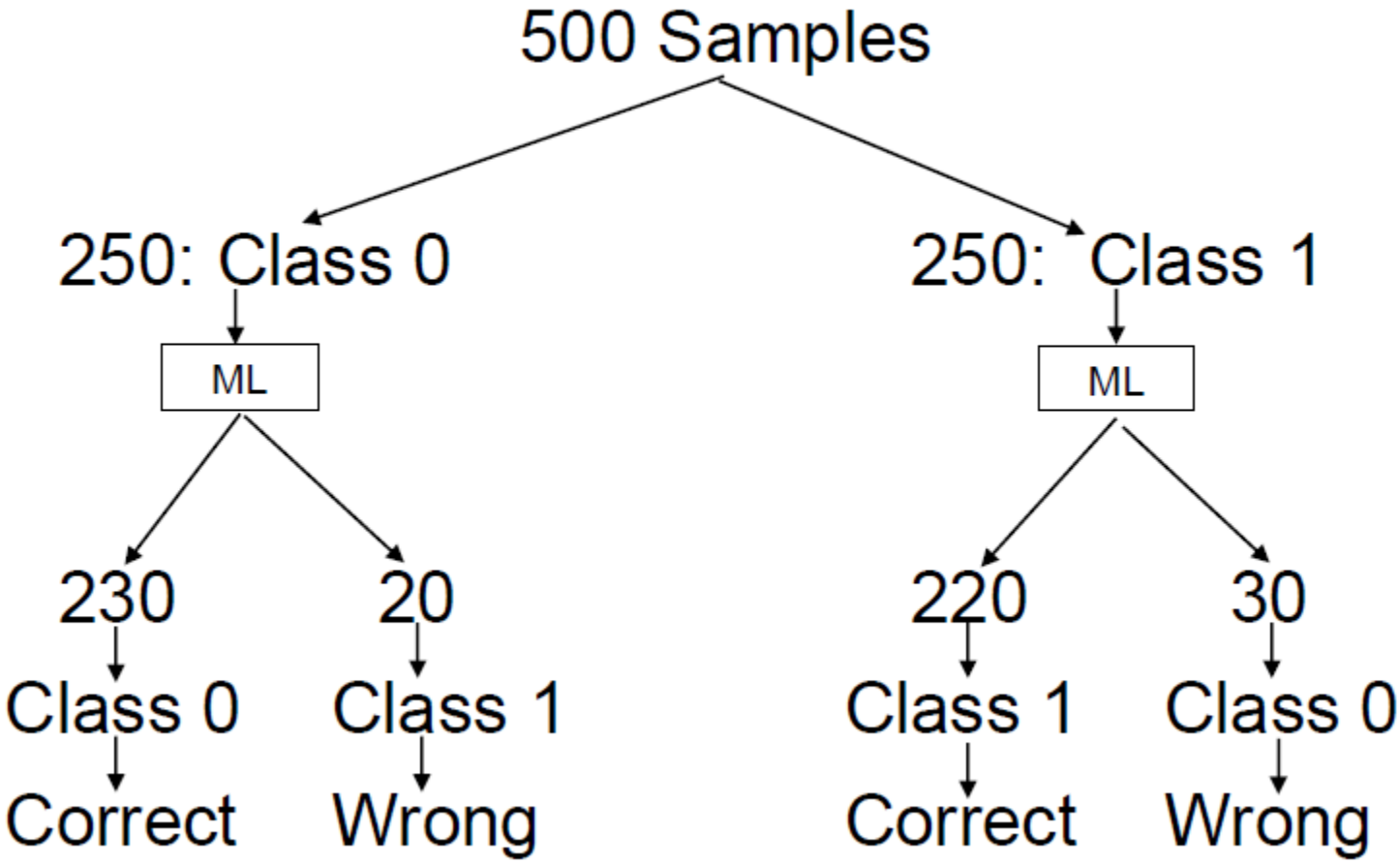
Neural Networks

- Hidden Layers map to lower dimensional spaces
- Search space has multiple local minima
- Training is expensive
- Classification extremely efficient
- Requires number of hidden units and layers
- Very good accuracy in typical domains

SVMs

- Kernel maps to a very-high dimensional space
- Search space has a unique minimum
- Training is extremely efficient
- Classification extremely efficient
- Kernel and cost, the two parameters to select
- Very good accuracy in typical domains
- Extremely robust

Confusion Matrix



Consider 0 as Negative and 1 as Positive

	Predicted 0	Predicted 1
Actual 0 (250)	TN 230	FP 20
Actual 1 (250)	FN 30	TP 220

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$\boldsymbol{F - score = 2 \times \frac{Precision \times Recall}{Precision + Recall}}$$