Support Vector Machine

# VC dimension

* Assume training T sample i=1,…T drawn from probability P(x,y) (IID) { }
* Consider family of machines (classifiers) : X -> {-1,1}
* Expect test error (risk):
  + Average error rate on all possible value (x,y) <~ cannot be computed in practice
  + Probability of misclassification
* Empirical risk: (on training data)

## Idea of VC:

* Generally using validation data to watching error rate. But we can use complexity of the classifier
* Complexity is high -> reduce error -> overfit
* We need to optimize between error rate and classifier complexity

## VC Dimension

* We have N points can be labelled {1,-1} => 2^N ways to labels
* H is set of machines (classifiers) . H **shatters** N of there **exists** a set of N so that for **all possible labellings,** there exists a consistent (no error)
* The maximum value N that H can shatters is called Vapnik Chervonenkis V(H) = N. With
* Expected error <= Estimated error + VC confidence (choose model with minimum upper bound)

# Support vector machine

## Mathematical things

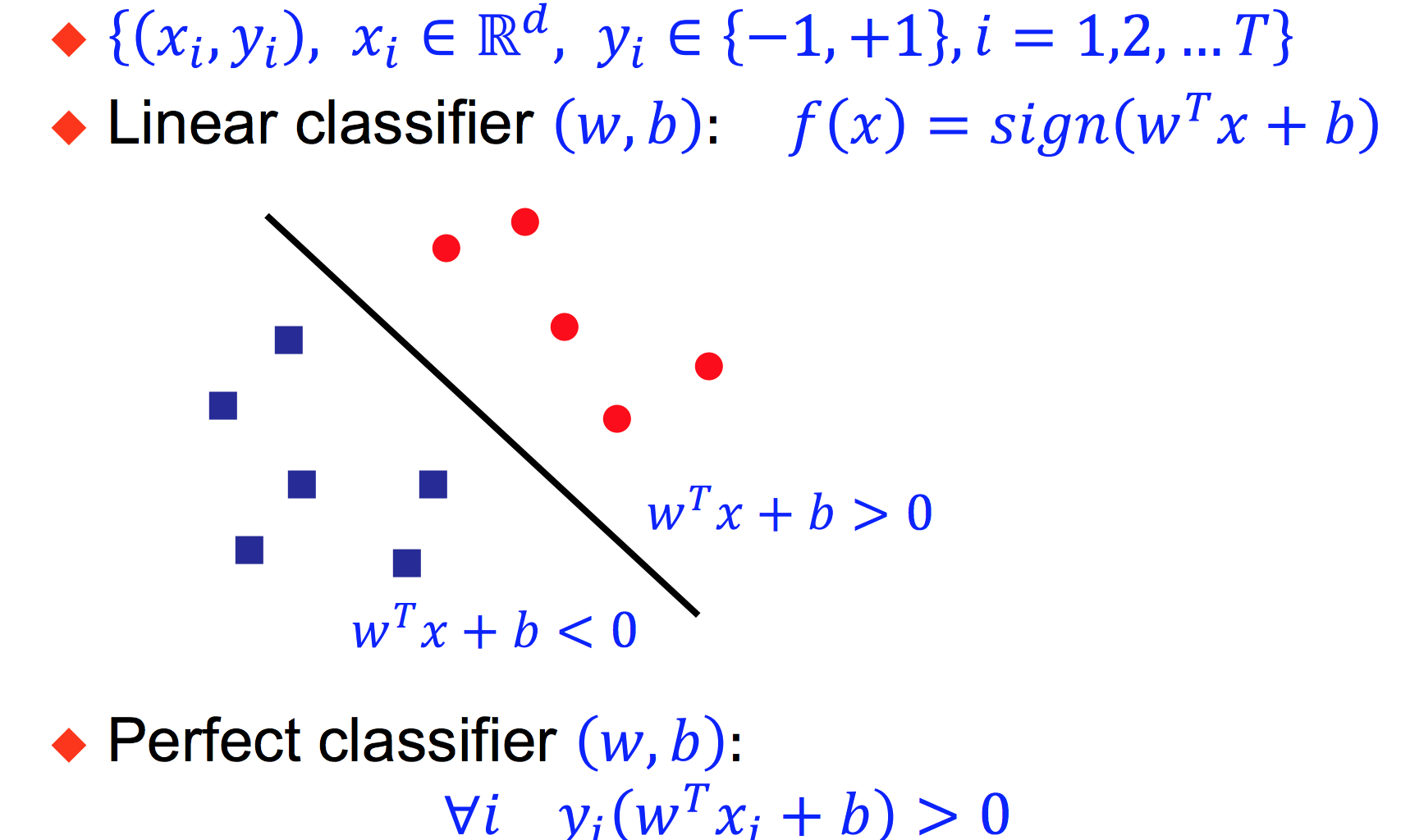
### Distance from a hyperplane

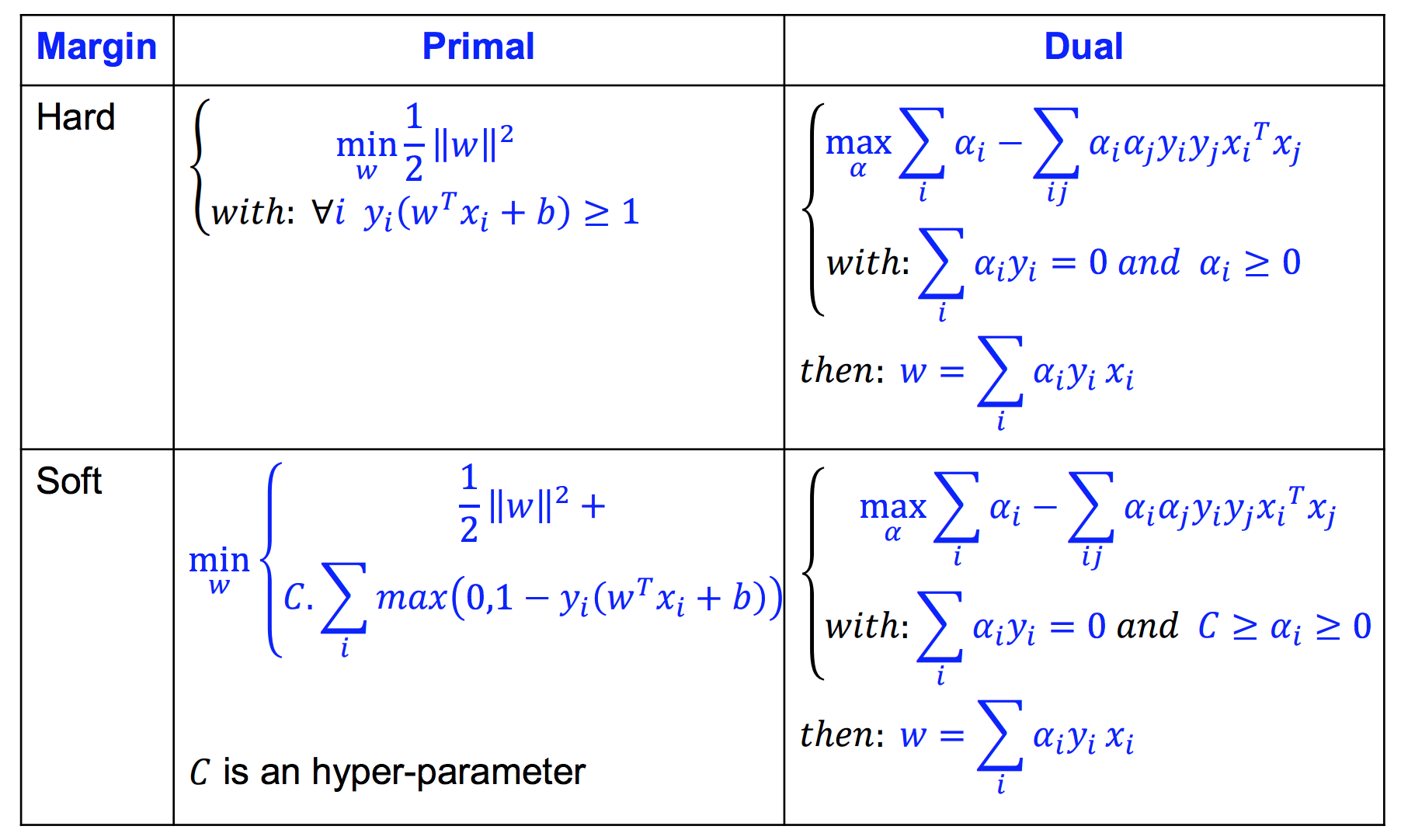
* Hyperplane equation
* Distance to hyperplane:
  + Project x on over H:
  + x’ belongs to H =>

### Boundary margin

* distance from class to boundary
* margin
* best boundary

## Hard margin





## Soft margin



## Non-linear boundary

* Move data into a higher dimension space and search for a linear separator
* Idea: transform input space into feature space
* Because solving dual problem we only need scalar product
* Kernel similarity measure between x\_i and x\_j
* Mercer’s theorem: every possitive semi definite symmetric function
* The dual problem:
* Weight vector is
* Discriminant function

### Kernel sample

* D-th degree polynomial , input space n => feature space combination(d,n+d)
* Radial basis
* Sigmoid with parameter alpha and beta