Statistical Learning Theory and the C-Loss cost function

Jose Principe, Ph.D.

Distinguished Professor ECE, BME

Computational NeuroEngineering Laboratory and

principe@cnel.ufl.edu

Statistical Learning Theory

In the methodology of science there are two primary methodologies to create undisputed principles (knowledge):

Deduction – starts with an hypothesis that must be scientific validated to arrive at a general principle that then can be applied to many different specific cases.

Induction – starts from specific cases to reach universal principles. Much harder than deducation.

Learning from samples uses an inductive principle and so must be checked for generalization.

Statistical Learning Theory

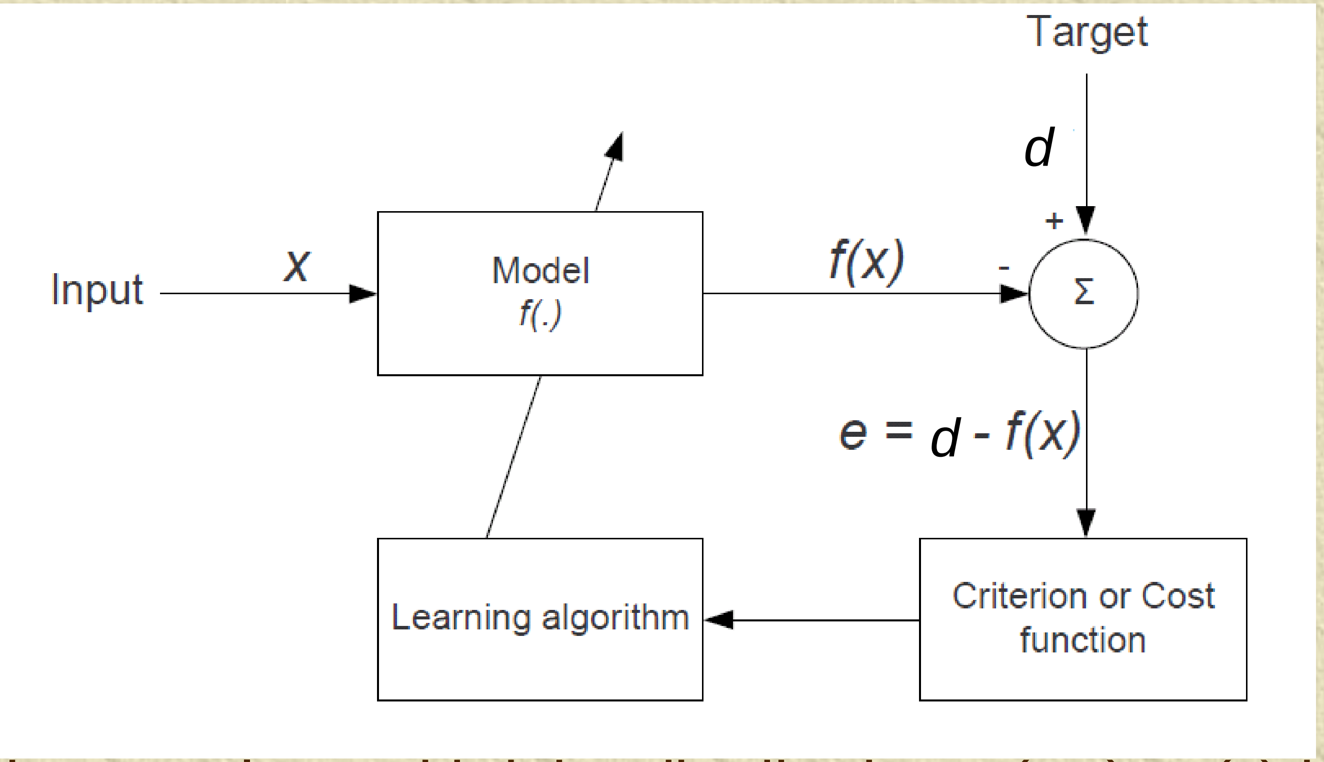
Statistical Learning Theory uses mathematics to study induction.

The theory has received lately a lot of attention and major advances were achieved.

The learning setting needs to be first properly defined. Here we will only treat the case of classfication.

Empirical Risk Minimization (ERM) principle

Let us consider a learning machine



x,d are real r.v. with joint distribution P(x,y). F(x) is a function of some parameters w, i.e. f(x,w).

Empirical Risk Minimization (ERM) principle

How can we find the possible best learning machine that generalizes for unseen data from the same distribution?

Define the Risk functional as

$$R(w) = \int L(f(x,w),d)\,dP(x,d) = E_{xd}[L(f(x,w),d)]$$

L(.) is called the Loss function, and minimize it w.r.t. w achieving the best possible loss.

But we can not do this integration because the joint is normally not known in functional form.

Empirical Risk Minimization (ERM) principle

The only hope is to substitute the expected value by the empirical mean to yield

$$R_E(w) = \frac{1}{N}\sum_i L(f(x_i, w), d_i)$$

Giovani and Cantelli proved that the ER converges to the true Risk, and Kolmogorov proved the convergence rate is exponential. So there is hope to achieve inductive machines.

What should the best loss function be for classification?

Empirical Risk Minimization (ERM) principle

The only hope is to substitute the expected value by the empirical mean to yield

$$R_E(w) = \frac{1}{N}\sum_i L(f(x_i, w), d_i)$$

Giovanni and Cantelli proved that the ER converges to the true Risk functional, and Kolmogorov proved that the convergence rate is exponential.

So there is hope to achieve inductive machines.

Empirical Risk Minimization (ERM) principle

What should the best loss function be for classification?

We know from Bayes theory that the classification error is the integral over the tails of the likelihoods, but this is very difficult to do in practice.

In the confusion tables, what we do is to count errors, so this seems to be a good approach. Therefore the ideal Loss is

$$l_{0/1}(f(x,w),d) = \begin{cases} 1 & df(x,w) < 0 \\ 0 & otherwise \end{cases}$$

Which makes the Risk

$$R(w) = P(Y \neq sign(f(x,w)) = E[l_{0/1}(f(X,w),D)$$

Empirical Risk Minimization (ERM) principle

Again, the problem is that the l0/1 loss is very difficult to work in practice. The most widely used family of losses are the polynomial losses that take the form

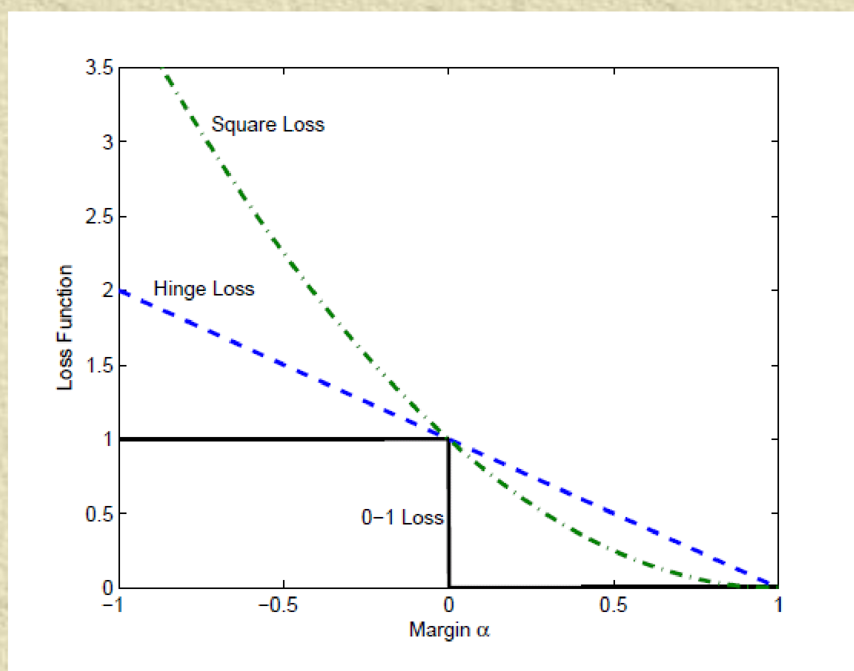
$$R(w) = E[(f(X,w) - D)^p]$$

Let us define the error as $e = d - f(x,w)$. If d={-1,1} and the learning machine has an output between [-1,1], the error will be between [-2,2]. Errors beyond |e|>1 correspond to wrong class assignments.

Sometimes we define the margin α as $\alpha = df(x,w)$. The margin is therefore in [-1,1] and for α >0 we have perfect class assignments.

Empirical Risk Minimization (ERM) principle

In the space of the margin the $l_{0/1}$ loss and the l_2 norm look as in the figure.



The hinge loss is a l_1 norm of the error. Notice that the square loss is convex, but the hinge is a limiting case, and $l_{0/1}$ is definitely non convex.

Empirical Risk Minimization (ERM) principle

It turns out that the quadratic loss is easy to work with for the minimization (we can use gradient descent). The hinge loss requires dynamic programming in the minimization, but the current availability of fast computers and optimization software is becoming practical.

The $l_{0/1}$ loss is still impractical to work with.

The down side of the quadratic loss (our well known MSE) is that machines trained with it are unable to control generalization, so they do not lead to useful inductive machines. The user must find additional ways to guarantee generalization (as we have seen – early stopping, weight decay).

Correntropy:
A new generalized similarity measure

Define correntropy of two random variables X,Y as

$$v(X,Y) = E_{XY}(\kappa_\sigma(X-Y))$$

by analogy to the correlation function. K is the Gaussian kernel.

The name correntropy comes from the fact that the average over the dimensions of the r.v. is the information potential (the argument of Renyi's entropy)

We can estimate readily correntropy with the empirical mean.

$$\hat{v}(x,y) = \frac{1}{N}\sum_{i=1}^{N}\kappa(x(i)-y(i))$$

Correntropy:
A new generalized similarity measure

Some Properties of Correntropy:

- It has a maximum at the origin ($1/\sqrt{2\pi}\sigma$)
- It is a symmetric positive function
- Its mean value is the argument of the log of quadratic Renyi's entropy of X-Y (hence its name)
- Correntropy is sensitive to second and higher order moments of data (correlation only measures second order statistics)

$$v(x,y)=\sum_{n=0}^{\infty}\frac{(-1)^n}{2^n\sigma^{2n}n!}E\|X-Y\|^{2n}$$

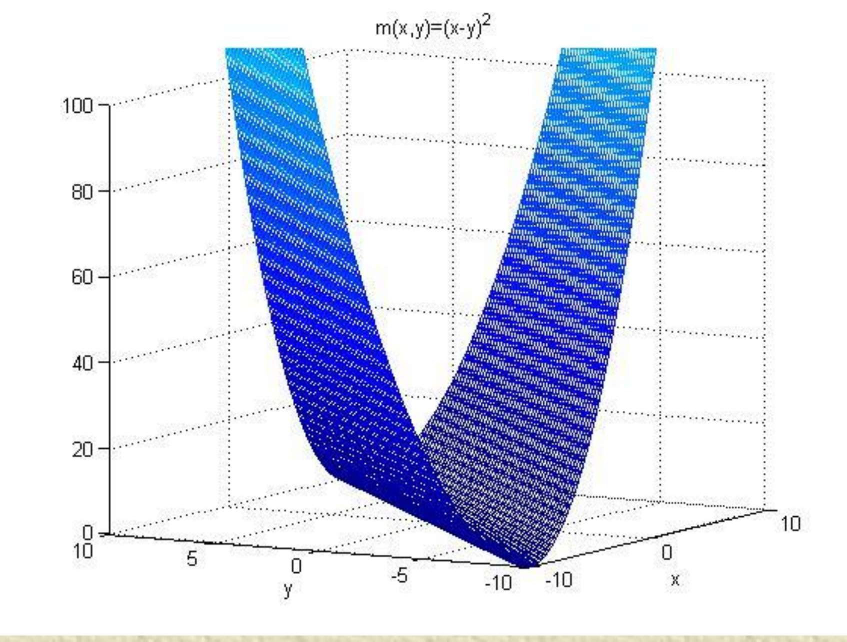
- Correntropy estimates the probability of X = Y.

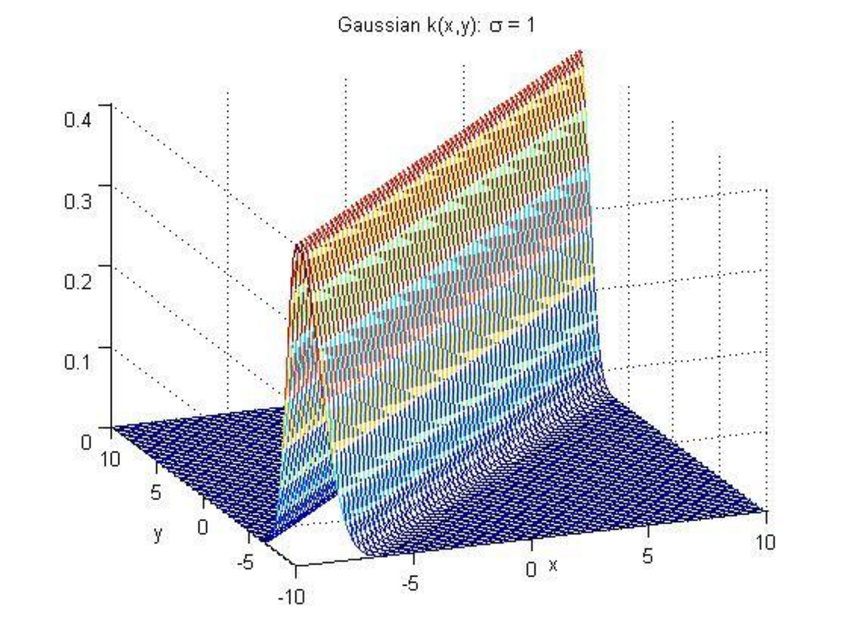
Correntropy:
A new generalized similarity measure

- Correntropy as a cost function versus MSE.

$$MSE(X,Y) = E[(X-Y)^2]$$
$$= \iint_{x,y} (x-y)^2 f_{XY}(x,y)dxdy$$
$$= \int_e e^2 f_E(e)de$$

$$V(X,Y) = E[k(X-Y)]$$
$$= \iint_{x,y} k(x-y) f_{XY}(x,y)dxdy$$
$$= \int_e k(e) f_E(e)de$$



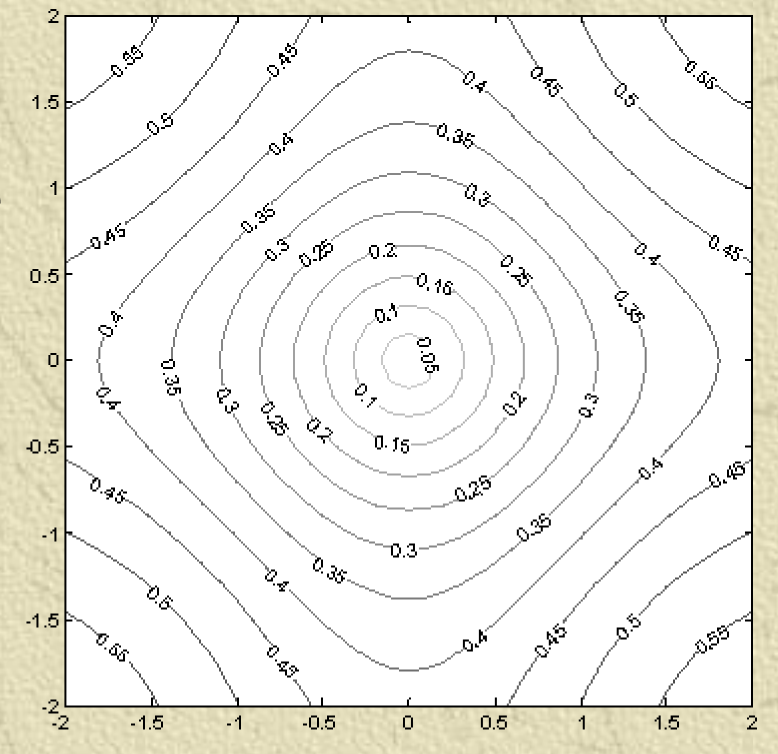


Correntropy:
A new generalized similarity measure

- Correntropy induces a metric in the sample space (CIM) defined by

$$CIM(X,Y) = (v(0,0) - v(x,y))^{1/2}$$

- Correntropy uses different L norms depending on the actual sample distances.



This can be very useful for outlier's control and also to improve generalization

The Correntropy Loss (C-loss) Function

We define the C-loss function as

$$l_C(d, f(x,w)) = \beta[1 - \kappa_\sigma(d - f(x,w))]$$

In terms of the classification margin α

$$l_C(\alpha) = \beta[1 - \kappa_\sigma(1-\alpha)]$$

β is a positive scaling constant that guarantees $l_C(\alpha = 0) = 1$

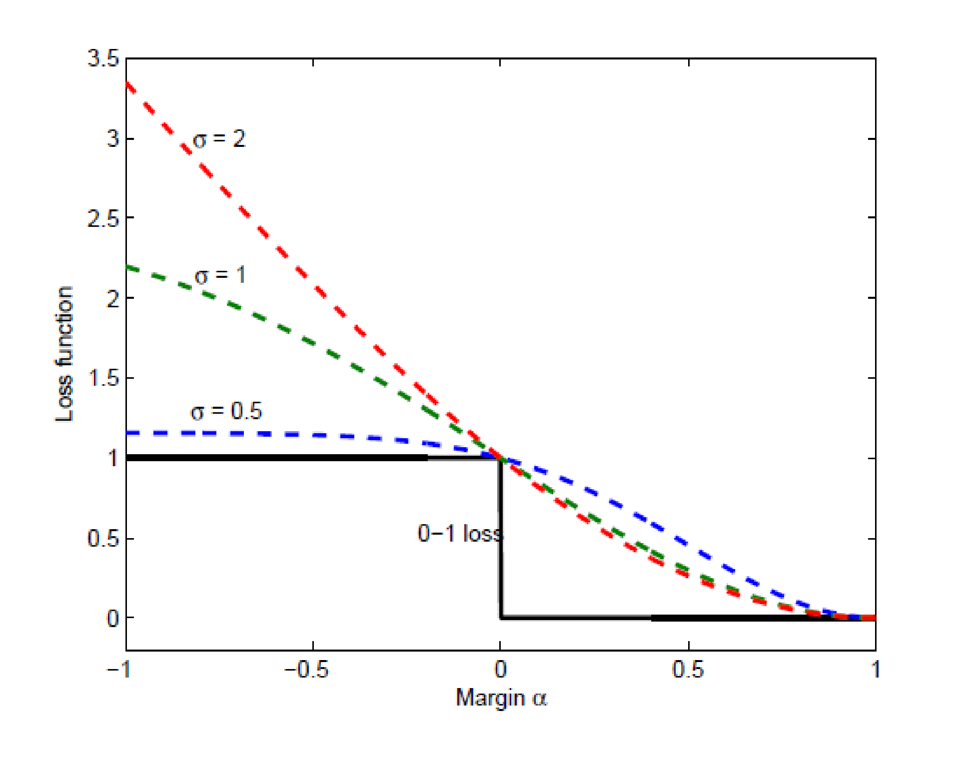
The expected risk of the C-Loss function is

$$R_C(w) = \beta(1 - E[\kappa_\sigma(d - f(x,w))]) = \beta(1 - v(D, f(X,w)))$$

Clearly, minimizing C-Risk is equivalent to maximizing the similarity in the correntropy metric sense between the true label and the machine output.

The Correntropy Loss (C-loss) Function

The C-Loss for several values of σ



The C-loss is non convex, but approximates better the $l_{0/1}$ loss and it is Fisher consistent.

The Correntropy Loss (C-loss) Function

Training with the C-Loss

Can use backpropagation with a minor modification: the injected error is now the partial of the C-Risk w.r.t. the error

$$\frac{\partial R_C(e)}{\partial e_n} = \frac{\partial l_C(e_n)}{\partial e_n}$$

or

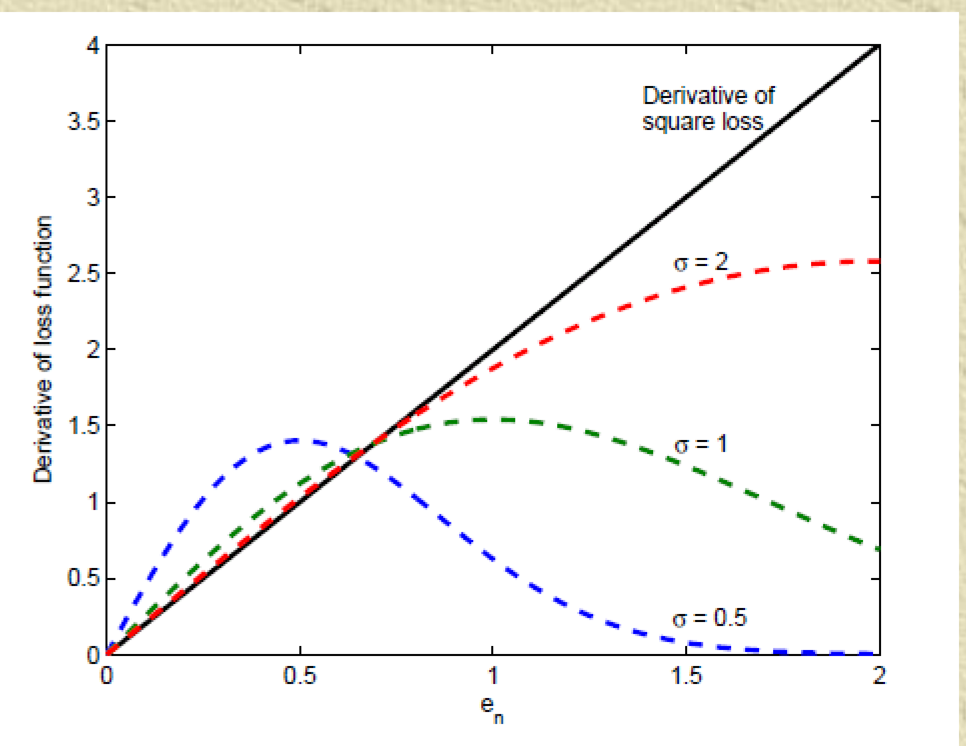
$$\frac{\partial l_C(e_n)}{\partial e_n} = \frac{\partial}{\partial e_n}\beta\left[1-\exp\left(\frac{-e_n^2}{2\sigma^2}\right)\right] = \frac{\beta e_n}{\sigma^2}\exp\left(\frac{-e_n^2}{2\sigma^2}\right)$$

All the rest is the same!

The Correntropy Loss (C-loss) Function
Automatic selection of the kernel size

An unexpected advantage of the C-Loss is that it allows for an automatic selection of the kernel size.

We select $\sigma = 0.5$ to give maximal importance to the correctly classified samples



The Correntropy Loss (C-loss) Function

How to train with the C-loss

The only disadvantage of the C-loss is that the performance surface is non convex and full of local minima.

I suggest to first train with MSE for 10-20 epochs, and then switch to the C-loss

Alternatively can use the composite cost function

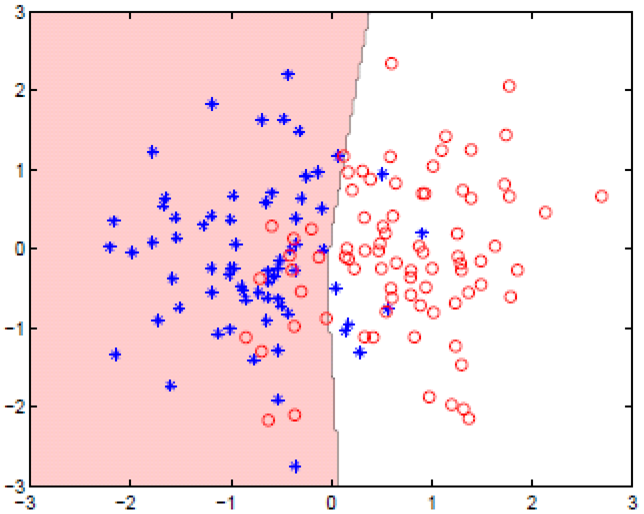
$$R(w) = (1-\frac{\lambda}{N})R_2(w) + \frac{\lambda}{N}R_C(w)$$

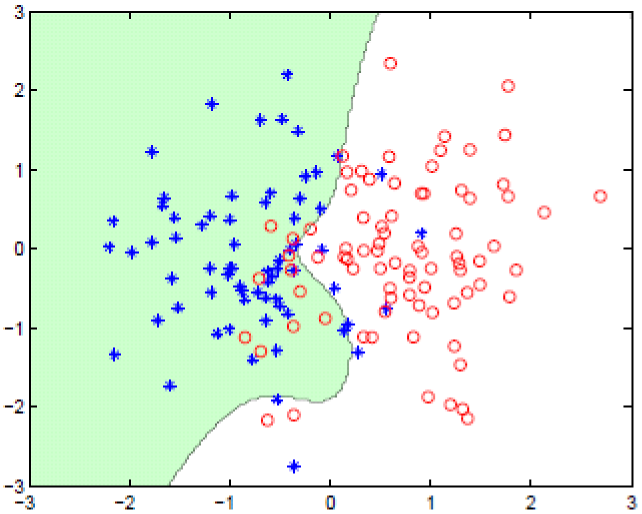
where N is the number of training iterations, and λ is set by the user.

The Correntropy Loss (C-loss) Function

Synthetic example: two Gaussian classes

Discriminant functions obtained using C-Loss and the Square loss functions:



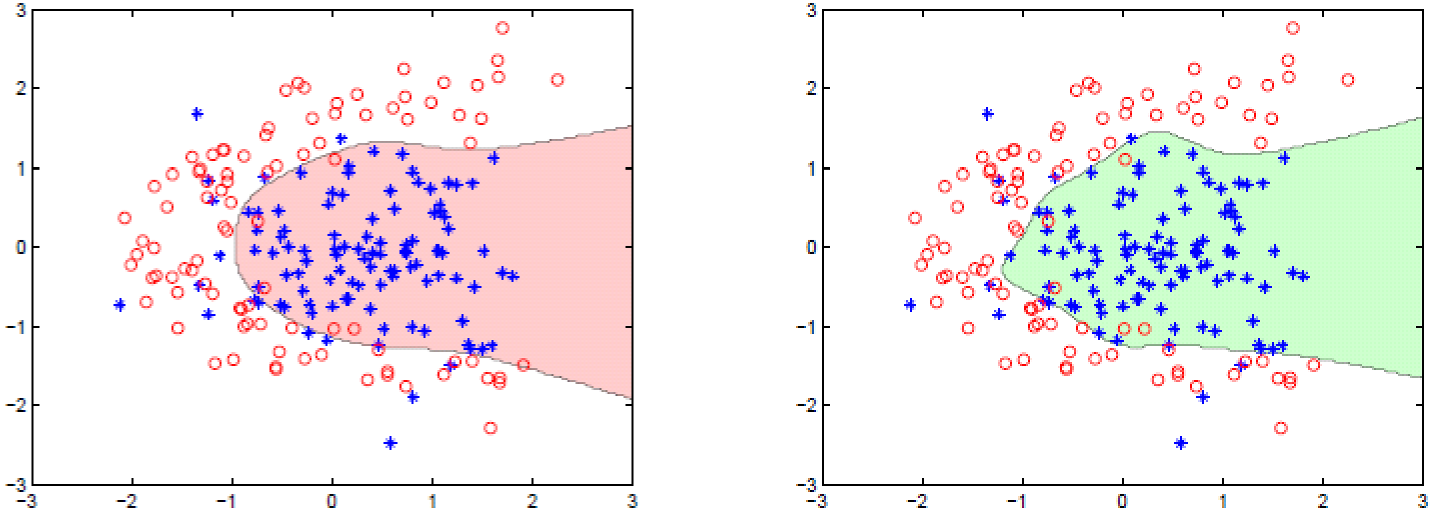


Notice how smooth is the separation surface

The Correntropy Loss (C-loss) Function

Synthetic example: more difficult case

Discriminant functions obtained using C-Loss and the Square loss functions:



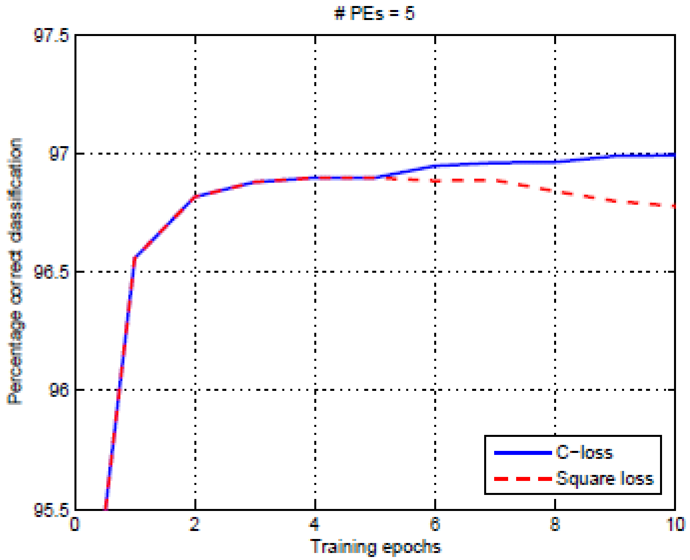
Notice how smooth is the separation surface

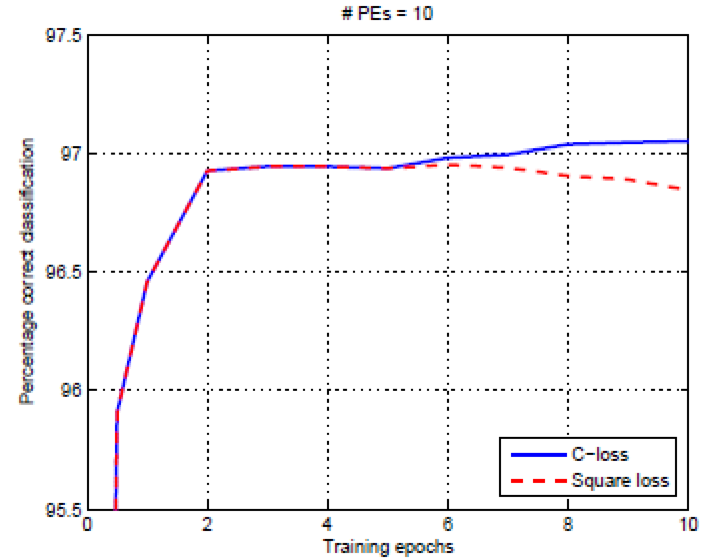
The Correntropy Loss (C-loss) Function

Wisconsin Breast Cancer Data Set

Train: 300 samples, Test: 383 samples

Classification performance vs. number of training epochs (with 5 and 10 PEs):





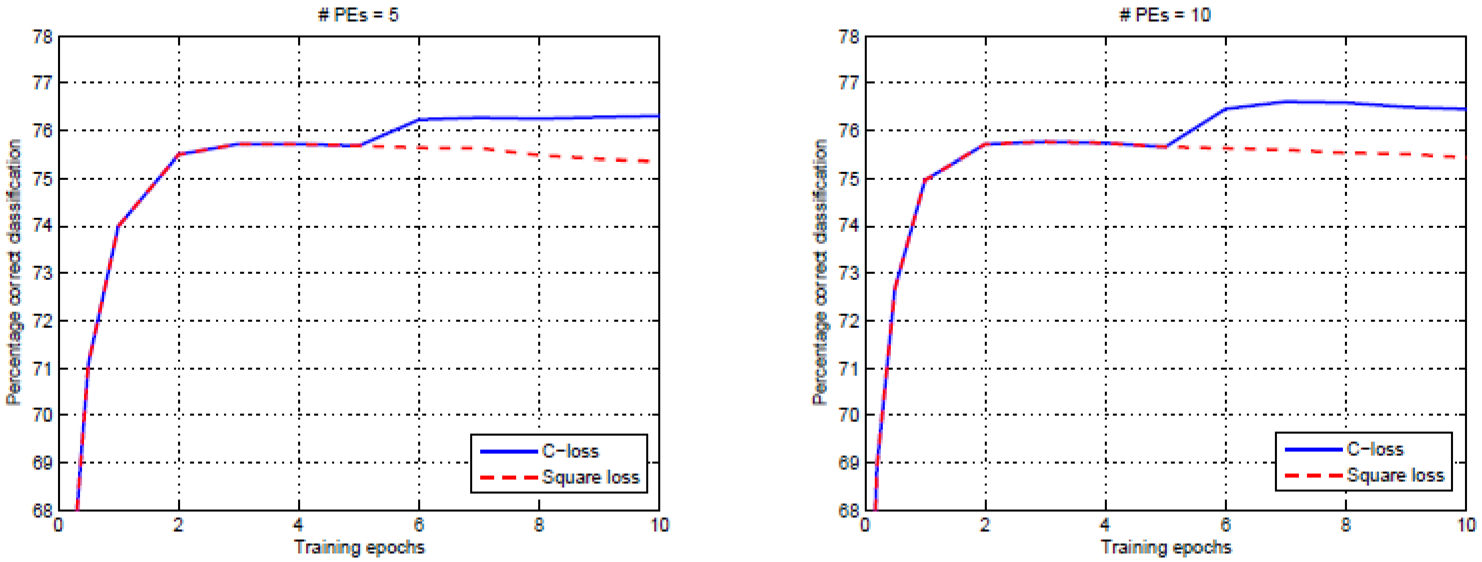
C-loss does NOT over train, so generalizes much better than MSE

The Correntropy Loss (C-loss) Function

Pima Indians Data Set

Train: 400 samples, Test: 368 samples

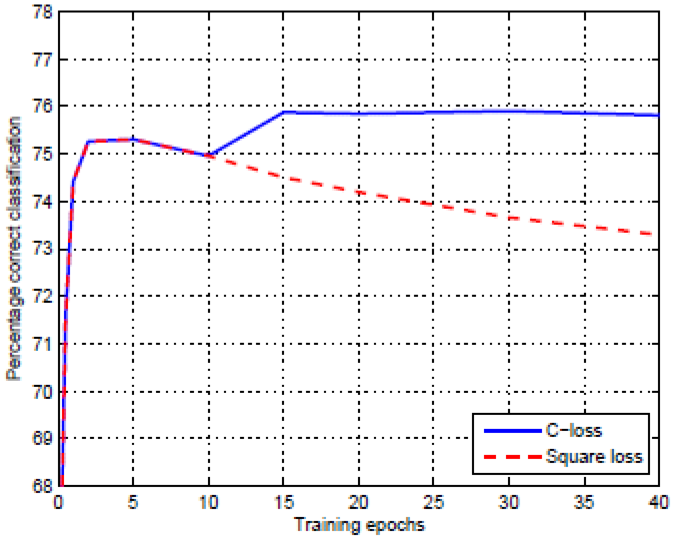
Classification performance vs. number of training epochs (with 5 and 10 PEs):

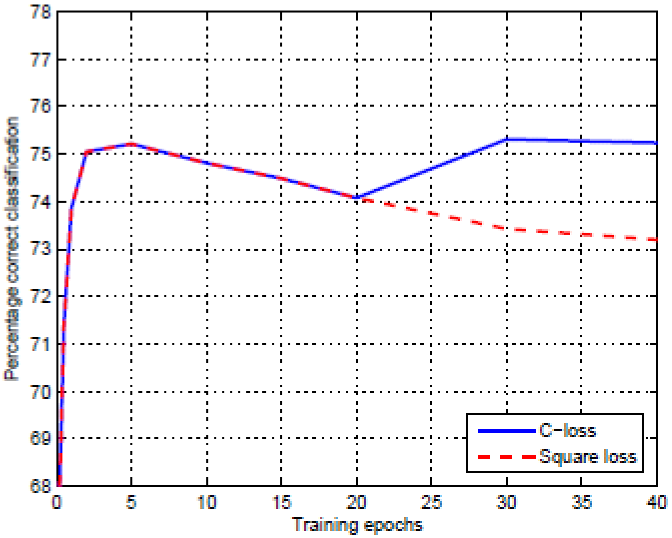


C-loss does NOT over train, so generalizes much better than MSE

The Correntropy Loss (C-loss) Function
But the point of switching affects performance

Early switching vs. late switching:





Conclusions

The C-loss has many advantages for classification:

- Leads to better generalization, as samples near the boundary have less impact on training (the major cause for overtraining with the MSE).
- Easy to implement - can be simply switched after training with MSE.
- Computation complexity is the same as MSE and backpropagation.

The open question is the search of the performance surface. The switching between MSE and C-loss afffects the final classification accuracy.