KERNELS

Kernels bridge the world of linear and non-linear optimization.

INTRO: FEATURE MAPPING

How did you solve your perceptron problem on the mystery dataset?

$(x, y, z) \rightsquigarrow$ $\dots$ $\left(\sqrt{x^2+y^2+z^2}\right) \approx (x^2+y^2+z^2)$

$\dots$ $(x, y, z, x^2, y^2, z^2, 1)$

$\dots$

Algorithm 5 PerceptronTrain($\mathbf{D}$, *MaxIter*)

$w_d \leftarrow 0$, for all $d = 1 \dots D$ // initialize weights
$b \leftarrow 0$ // initialize bias
for $iter = 1 \dots MaxIter$ **do**
  for all $(x,y) \in \mathbf{D}$ **do**
    $a \leftarrow \sum_{d=1}^{D} w_d\, x_d + b$ // compute activation for this example
    if $ya \leq 0$ **then**
      $w_d \leftarrow w_d + yx_d$, for all $d = 1 \dots D$ // update weights
      $b \leftarrow b + y$ // update bias
    end if
  end for
end for
return $w_0, w_1, \dots, w_D, b$

Algorithm 6 PerceptronTest($w_0, w_1, \dots, w_D, b, \hat{x}$)

$a \leftarrow \sum_{d=1}^{D} w_d\, \hat{x}_d + b$ // compute activation for the test example
return SIGN(a)

We can always write w as linear combination of inputs! (REPRESENTER THEOREM)

Algorithm 30 $\textsc{KernelizedPerceptronTrain}(\mathbf{D}, \mathit{MaxIter})$

```
α ← 0, b ← 0                                   // initialize coefficients and bias
for iter = 1 ... MaxIter do
    for all (x_n, y_n) ∈ D do
        a ← Σ_m α_m φ(x_m) · φ(x_n) + b         // compute activation for this example
        if y_n a ≤ 0 then
            α_n ← α_n + y_n                      // update coefficients
            b ← b + y                            // update bias
        end if
    end for
end for
return α, b
```

Now we can replace $\langle \phi(x_m), \phi(x_n) \rangle$ with $K(x_m, x_n)$!

The same idea works for many other methods.

WHAT MAKES A KERNEL?

K-means:

1) Start with a guess for K centers

2) Loop while not converged:

   - assign points to centers
   - recompute centers

Linear regression

$$f_w(\hat{x}) = \langle w, \Phi(\hat{x}) \rangle$$

$$L(w) = \sum_i \left(f_w(x_i) - y_i \right)^2$$

$$= \sum_i \left(\langle w, \Phi(x_i) \rangle - y_i \right)^2$$

Representer thm:

$$f(\hat{x}) = \sum c_i \, K(x_i, \hat{x}) \qquad = \sum c_i \langle \Phi(x_i), \Phi(\hat{x}) \rangle$$

Then loss function will be given by:

$$L(f) = \|Kc - \vec{y}\|^2 + \lambda \|c\|^2$$

(Why?)

$$2Kc + 2\lambda c = 2\vec{y}$$

$$(K + \lambda I)c = \vec{y}$$

SUPPORT VECTOR MACHINES