Lecture 17 - 12-05-2020

1.1 Kernel functions

We use a notion of feature expansion. They are different but somehow they reach something similar. In fact Linear classifier have high bias.
Linear predictor use hyper plane as basic brick to build prediction.

1.1.1 Feature expansion

Given $\phi : \mathbb{R}^d \longrightarrow V \qquad V$ is typically $\mathbb{R}^N \qquad N >> d$

For example:
$d = 2 \quad N = 6 \quad \phi = \mathbb{R}^2 \longrightarrow \mathbb{R}^6$

$$\phi(x_1, x_2) = (1, x_1^2, x_2^2, x_1, x_2, x_1 x_2)$$

We have a homogenous hyper plane.
$w \in \mathbb{R}^6 \ \{z \in \mathbb{R}^6 : w^T z = 0\} \qquad z = \phi(x) \qquad x \in \mathbb{R}^2$

$$\forall x \in \mathbb{R}^2 \qquad w^t \phi(x) = w_1 + w_2 x_1^2 + w_3 x_2^2 + w_4 x_1 + w_5 x^2 + w_6 x_1 x_2 = 0$$

$$w^T \phi(x) = 0$$

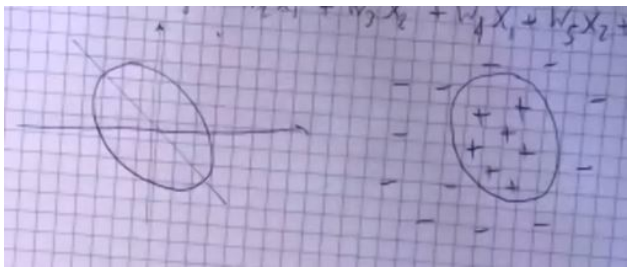

Figure 1.1:

$$\phi : \mathbb{R}^d \longrightarrow \mathbb{R}^N \qquad \Pi_{s=1}^M x_{Vs} \quad v \in \{1, ...d\}^k \quad k = 0, ..., n$$

$$h(x) = sgn(w^T \phi(x)) \qquad w^T \phi(x) = \sum_{i=1}^{N} w_i \phi(x)_i$$

The problem of this feature expansion is the degree of the monomials!

$$N = \sum_{i=0}^{n} |\{1, ...d\}^k| = \sum_{k=0}^{n} d^k = \frac{d^{n+1} - 1}{d - 1} = \Theta(d^n)$$

So it's exponential! But this feature expansion can be implemented in a efficient way.

1.1.2 Kernels implements feature expansion (Efficiently

$w^T\phi(x)$ Perception $w \leftrightarrow w + y_t x_t I\{y_t w^T x_t \leq 0\}$ MANCA quadlcosa

$$w = \sum_{s \in S} y_s x_s \rightsquigarrow \sum_{s \in S} y_s \phi(x_s)$$

where S is a subset of traning set where updates occurred.
Every time i make a mistake i add some of this product of data points.
If I run this using example that are images accourding to some feature expansion map (ϕ), I will get the perceptron after the mapping.

$$w^T\phi(x) = \sum_{s \in S} y_s \phi(x)^T \phi(x_s)$$

It's a inner problem and can have exponentially degree of the component.
Kernels help me compute this inner product $\phi(x)^T\phi(x_s)$ quickly

$$\phi : \mathbb{R}^2 \longrightarrow \mathbb{R}^6 \qquad \phi(x_1, x_2) = (1,\ x_1^2,\ x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2)$$

$$\phi(x)^T\phi(z) \ = \ 1 + x_1^2z_1^2 + x_2^2z_2^2 + 2x_2z_2 + 2x_1x_2z_1z_2 \ = \ (1 + x^Tz)^2 \ = \ k(x, z)$$

$$w^T\phi(x) = \sum_{s \in S} y_s \ k(x, x_2)$$

$k(x, z)$ implements $\phi(x)^T\phi(z)$ $\forall x, z$ and ϕ defined as before

How to we generalise this?

$$k_n\,(x, x') = (1 + x^2x')^n$$

This is called polynomial kernel.

I want to check now what is the ϕ for K_n?
I want to compute ϕ s.t. $\phi(x)^T\phi(x') = k_n(x,x') = (1+x^Tx')^n$
We can use Newtons bynomial theorem:

$$(1+x^Tx')^n = \sum_{k=0}^{n} \binom{n}{k} (x^tx')^k$$

$$(x^Tx')^k = \left(\sum_{i=1}^{d} x_i x_i'\right)^k = \sum_{v\in\{1,...d\}^k} \left(\prod_{s=1}^{k} x_{Vs}x'_{Vs}\right)$$

$$\phi(x) = \left(\sqrt{\binom{n}{k}} \prod_{s=1}^{k} x_{Vs}\right) \qquad k=0,...,n \qquad v\in\{1,...d\}^k$$

When I am using polynomial kernel I am implicitely using the feature expansion ...
Can an algorithm work using kernel?
Perceptron works!

$S = 0$
For $t = 1, 2, ...$
1) Get (x_t, y_t)
2) $\hat{y}_t = sgn\left(\sum_{s\in S} y_s\ K(x, x_s)\right)$
3) If $\hat{y}_t \neq y_t \quad S \longleftarrow S\cup\{t\} \qquad w \leftarrow w + y_t\phi(x_t)$

So I am representing y as a sum and not as a vector. In fact, $w = \sum_{s\in S} y_s\phi(x_s)$

ù

1.2 Gaussian Kernel

$$\gamma > 0 \qquad k_\gamma(x,x') = \exp\left(-\frac{1}{2\ \gamma}\ \|x-x'\|^2\right)$$

$$e^{-\frac{1}{2\ \gamma}\ (x-x')^2}$$

I can controll the distribution changing the value of γ

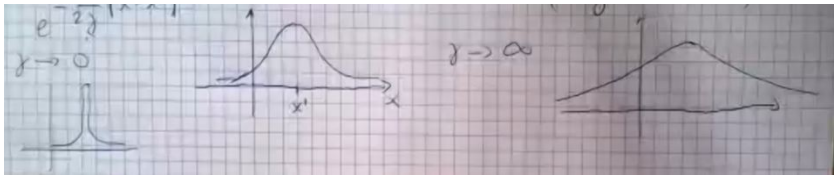

Figure 1.2:

$$\hat{y}_t = sgn\left(\sum_{s\in S} y_s \; K_\gamma(x, x_s)\right)$$

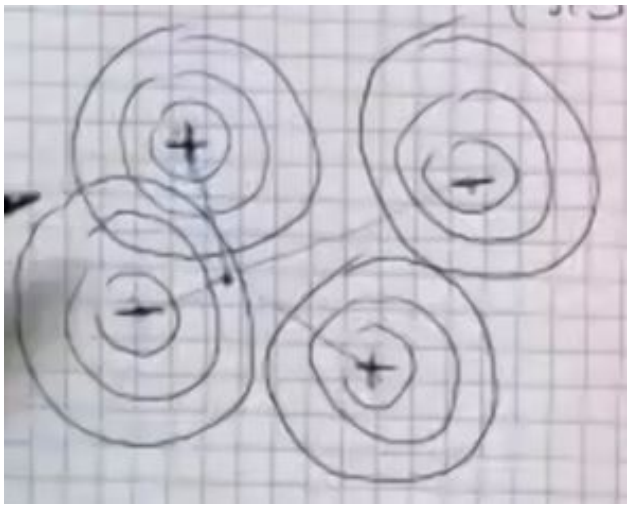

Figure 1.3:

Negative or positive gaussin component looking at the distance.
Now I want to compute: $\phi_\gamma : \mathbb{R}^n \longrightarrow V$

$$\exp\left(-\frac{1}{2\;\gamma}\|x - x'\|^2\right) = \exp\left(-\frac{1}{2\;\gamma}\left(\|x\|^2 + \|x'\|^2\right)\right) \cdot \exp\left(\frac{1}{\gamma}x^T x'\right) =$$

where $e = x + \frac{x^2}{!2}...$

$$= \exp\left(-\frac{1}{2\;\gamma}\|x\|^2\right) \cdot \exp\left(-\frac{1}{2\;\gamma}\|x'\|^2\right) \cdot \sum_{n=0}^{\infty} \frac{1}{n!}\frac{\left(x^T x'\right)^2}{\gamma^n}$$

Gaussian Kernel is a linear combination of infinitely many poly kernels.
The higher I go the small is $n!$. Gaussian kernel mapping into a space that is very large. So large that it has infinitely many dimension. Why? Because each polynomial kernel maps to infinitely dimensions.

ϕ_γ maps $\mathbb{R}^d$ into a space of infinitely many dimensions.

$$\phi_\gamma : \mathbb{R}^d \to V \qquad k_\gamma(x, x') = \phi_\gamma(x)^T \phi_\gamma(x)$$

It maps to infinetely many dimension, so it maps to a function!
$\phi_\gamma(x)$ **is a function.**
In general, when I learn a linear predictor using k_γ
I learn $\sum_s \alpha_s \; k(x_s, \cdot) = f$
$w^T \phi(x)$

$$H_\gamma \equiv \{ \sum_{i=1}^{N} \alpha_i \; k(x_i, \cdot) \;\; : \;\; x_1, ..., x_N \in \mathbb{R}^d, \;\; \alpha_1, ..., \alpha_N \in \mathbb{R}, \;\; N \in \mathbb{N} \}$$

Theorem
$\forall \gamma > 0 \; \forall f : \mathbb{R}^d \to \mathbb{R}$ continous, $\forall \varepsilon > 0$
$\exists g \in H_\gamma$ that approximates f with error ε
We define a function with H. We see the $\cdot$ and this tell us is a function. So we can evaluate every kind of x point in $\cdot$ position.

We are able to get a super parametric algorithm and transform it in a non-parametric algorithm. Parametric algorithms is defined by an arbitrary number of parameter we cannot adapt it for every case.

Gaussian Kernels enable consistency by using feature expansion with infinitely many components.