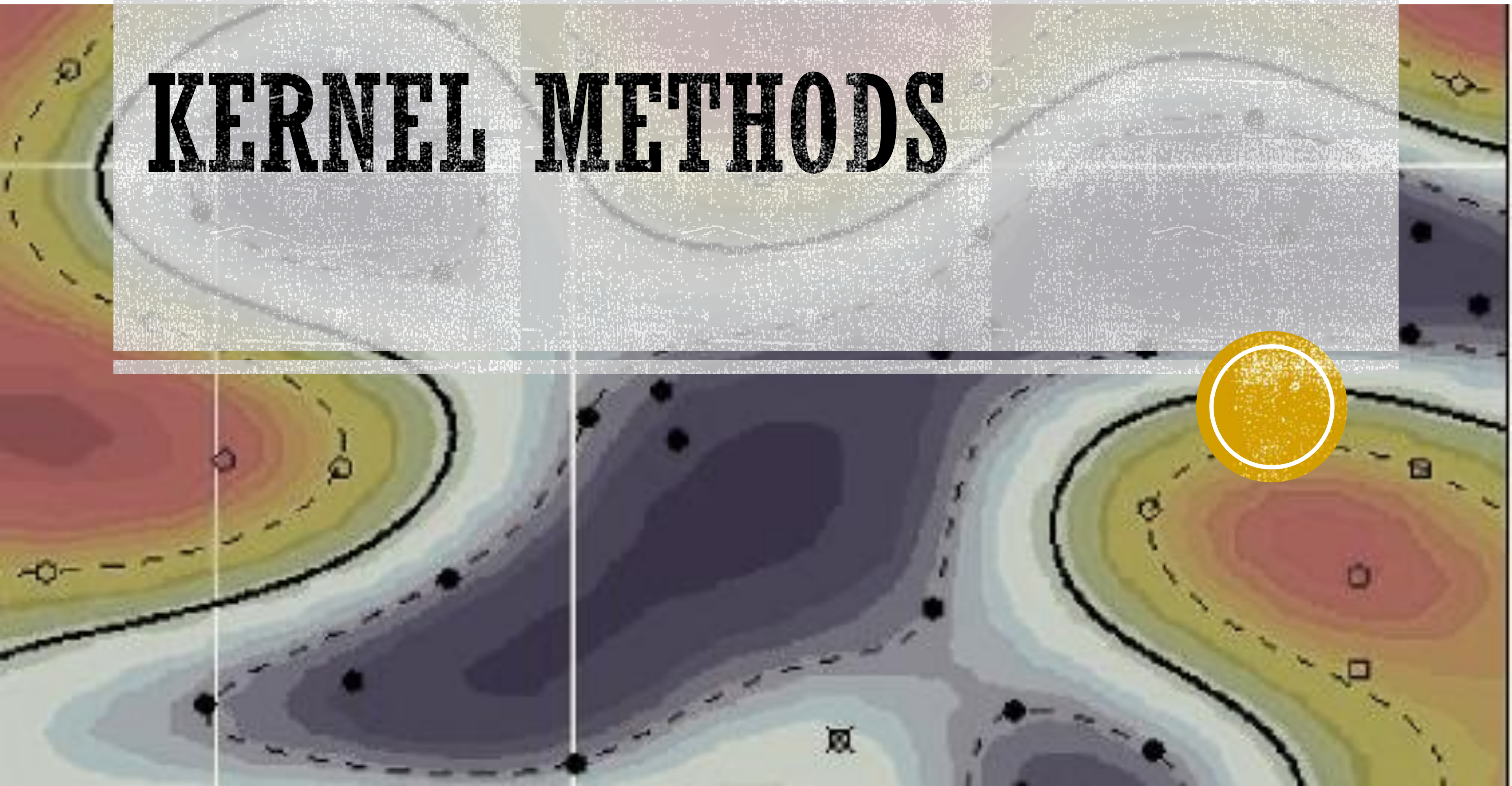
KERNEL METHODS



FEATURE MAPS

Example. Non-linear classifiers.

$$x = (x_1, x_2)$$

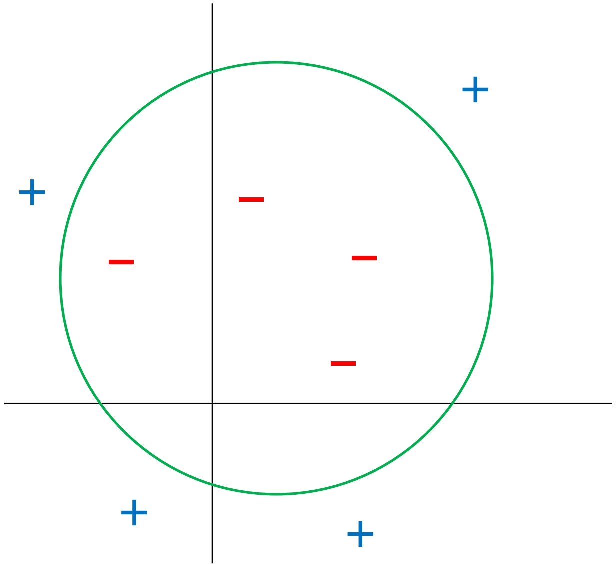
$$\phi(x) = \left(1, 2x_1, 2x_2, \sqrt{2}\, x_1 x_2, x_1^2, x_2^2\right)$$

$$\begin{aligned} h(x; \theta, \theta_0) &= \operatorname{sign}\big(\theta \cdot \phi(x)\big) \\ &= \operatorname{sign}\left(\theta_1 + \theta_2 2x_1 + \theta_3 2x_2 + \theta_4 \sqrt{2} x_1 x_2 + \theta_5 x_1^2 + \theta_6 x_2^2\right) \end{aligned}$$

FEATURE MAPS

Example. Non-linear classifiers.

$$h(x;\theta,\theta_0)$$
$$= \operatorname{sign}\big((x_1-1)^2+(x_2-2)^2-9\big)$$
$$= \operatorname{sign}\big(-4-2x_1-4x_2+x_1^2+x_2^2\big)$$



CHALLENGES

High-Dimensional Features.

$$x = (x_1, x_2, \dots, x_{1000}) \in \mathbb{R}^{1000}$$

$$\phi(x) = \left(1, \dots, x_i\,, \dots, \sqrt{2}x_i x_j, \dots, x_i^2\,, \dots\right) \in \mathbb{R}^{501501}$$

Inner Products.

Computing $\phi(x) \cdot \phi(x')$ for $x, x' \in \mathbb{R}^{1000}$ requires about 2,004,000 floating point operations.

KERNEL FUNCTIONS

Fortunately, many inner products simplify nicely.

$$
\begin{aligned}
K(x, x') &= \phi(x) \cdot \phi(x') \\
&= 1 + 2\textstyle\sum_i x_i x_i' + 2\sum_{i<j} x_i x_j x_i' x_j' + \sum_i x_i^2 x_i'^2 \\
&= 1 + 2(\textstyle\sum_i x_i x_i') + (\sum_i x_i x_i')^2 \\
&= (x \cdot x' + 1)^2
\end{aligned}
$$

For $x, x' \in \mathbb{R}^{1000}$, computing this requires only about 2000 floating point operations, less than the 501,501 operations needed for $\phi(x)$.

KERNEL FUNCTIONS

‘Infinite dimensional’ positive definite matrices

Definition.

A function $K: \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a *kernel function* if

1. $K(x, y) = K(y, x)$ for all $x, y \in \mathbb{R}^d$,
2. given $n \in \mathbb{N}$ and $x^{(1)}, x^{(2)}, \ldots, x^{(n)} \in \mathbb{R}^d$,

   the *Gram matrix* K with entries

   $$K_{ij} = K\left(x^{(i)}, x^{(j)}\right)$$

   is positive semidefinite.

Example. $K(x, x') = \phi(x) \cdot \phi(x')$

Can be shown that all kernel functions are of this form!

EXAMPLES

Linear Kernel.

$$K(x, x') = x \cdot x'$$

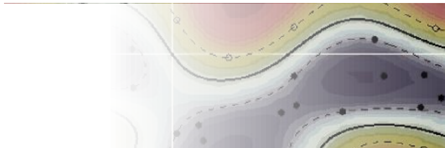
Polynomial Kernel.

$$K(x, x') = (x \cdot x' + 1)^k$$

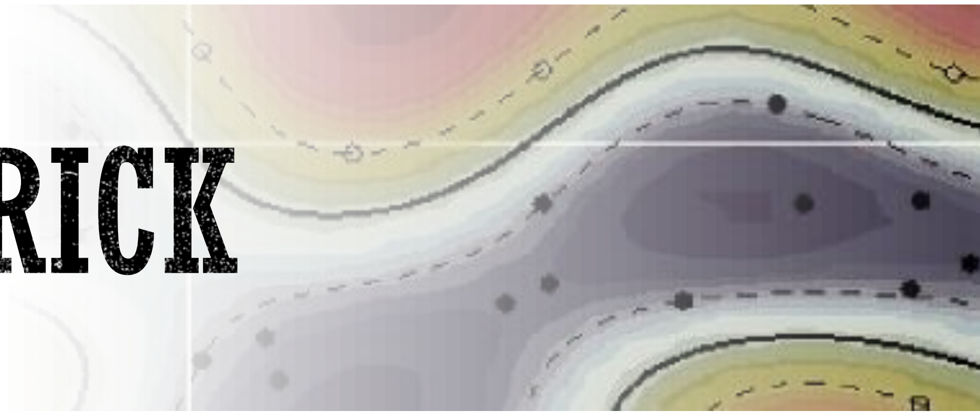
Radial Basis Kernel.

$$K(x, x') = \exp\left(-\frac{1}{2}\|x - x'\|^2\right)$$

Feature map $\phi(x)$ is infinite dimensional!

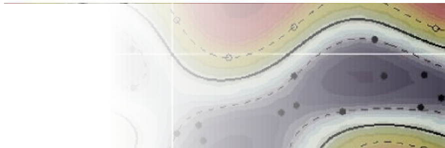


KERNEL TRICK



KERNEL TRICK

The kernel trick refers to the strategy of converting a learning algorithm and the resulting predictor into ones that involve only the computation of the kernel $K(x, x') = \phi(x) \cdot \phi(x')$ but not of the feature map $\phi(x)$.



SUPPORT VECTOR MACHINES

Learning.

maximize $\sum_{(x,y)} \alpha_{x,y} - \frac{1}{2} \sum_{(x,y)} \sum_{(x',y')} \alpha_{x,y}\, \alpha_{x',y'}\, y y' (x \cdot x')$

subject to $\alpha_{x,y} \geq 0$ for all (x, y)

Prediction.

$$h(x;\theta) = \mathrm{sign}(\theta \cdot x) = \mathrm{sign}\left(\sum_{(x',y')} \alpha_{x',y'}\, y' (x \cdot x')\right)$$

KERNEL SUPPORT VECTOR MACHINES

Learning.

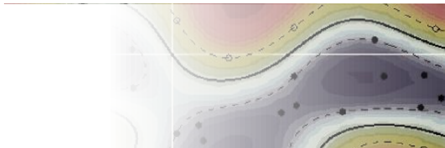
$$\text{maximize } \sum_{(x,y)} \alpha_{x,y} - \frac{1}{2} \sum_{(x,y)} \sum_{(x',y')} \alpha_{x,y}\, \alpha_{x',y'}\, y y' K(x, x')$$

$$\text{subject to } \alpha_{x,y} \geq 0 \text{ for all } (x, y)$$

Prediction.

$$h(x;\theta) = \mathrm{sign}(\theta \cdot x) = \mathrm{sign}\left(\sum_{(x',y')} \alpha_{x',y'}\, y' K(x, x')\right)$$

We may use the linear, polynomial, or radial basis kernels to get different kinds of decision boundaries.



PERCEPTRON

Learning.

1. Initialize $\theta = 0, \theta_0 = 0$.
2. Repeat until no mistakes are found:
   Select data $(x, y) \in \mathcal{S}_n$ in sequence:
   If $y(\theta^\top x + \theta_0) \leq 0$, then $\theta \longleftarrow \theta + yx,\ \ \theta_0 \longleftarrow \theta_0 + y$.

Prediction.

$$h(x; \theta, \theta_0) = \operatorname{sign}(\theta \cdot x + \theta_0)$$

From the learning algorithm, we see that

$\theta = \sum_{x,y} \alpha_{x,y} yx$ and $\theta_0 = \sum_{x,y} \alpha_{x,y} y$ for some $\alpha_{x,y} \in \mathbb{N}$.

PERCEPTRON

Learning.

1. Initialize $\theta = 0, \theta_0 = 0$.
2. Repeat until no mistakes are found:
   Select data $(x, y) \in \mathcal{S}_n$ in sequence:
   If $\sum_{x',y'} \alpha_{x',y'}\, y y' (x \cdot x' + 1) \leq 0$, then $\alpha_{x,y} \longleftarrow \alpha_{x,y} + 1$.

Prediction.

$$h(x; \theta, \theta_0) = \operatorname{sign}\left(\sum_{x',y'} \alpha_{x',y'}\, y' (x \cdot x' + 1)\right)$$

KERNEL PERCEPTRON

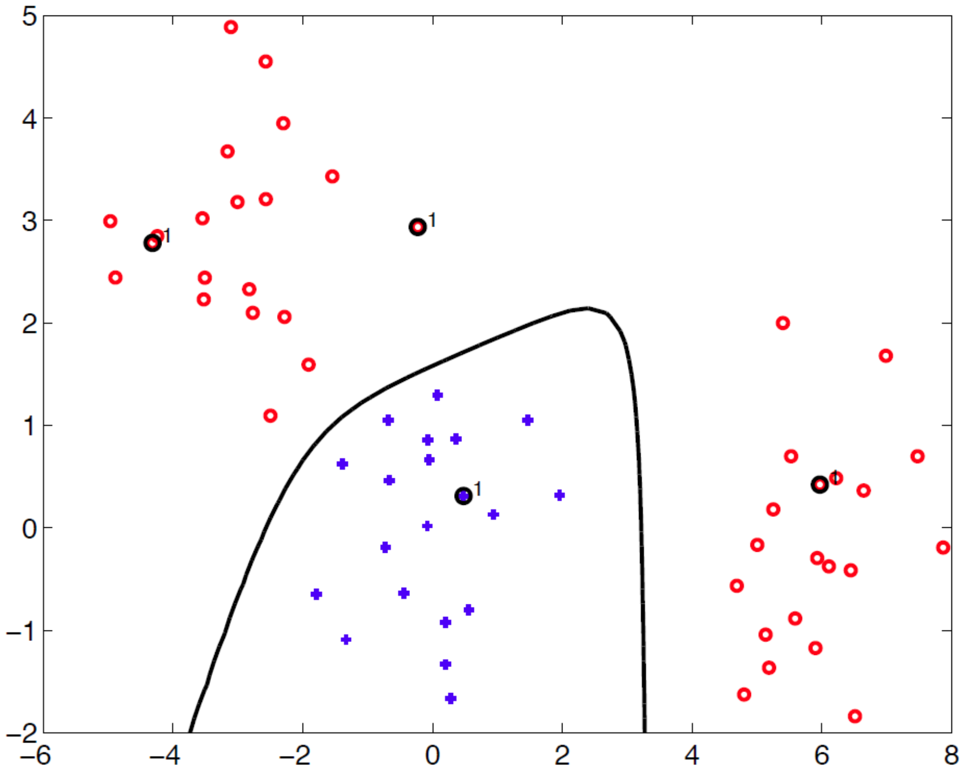
Learning.

1. Initialize $\theta = 0, \theta_0 = 0$.
2. Repeat until no mistakes are found:
   Select data $(x, y) \in \mathcal{S}_n$ in sequence:
   If $\sum_{x',y'} \alpha_{x',y'}\, yy'(K(x,x') + 1) \leq 0$, then $\alpha_{x,y} \longleftarrow \alpha_{x,y} + 1$.

Prediction.

$$h(x; \theta, \theta_0) = \operatorname{sign}\left(\sum_{x',y'} \alpha_{x',y'}\, y'(K(x,x') + 1)\right)$$

KERNEL PERCEPTRON



Data is always separable when using radial basis kernels!

LINEAR REGRESSION

Learning.

Let $n\lambda\alpha_{x,y} = y - \theta \cdot x$ so we have $n\lambda\alpha = Y - X\theta$.

Recall that the exact solution θ satisfies

$$(n\lambda I + X^\top X)\theta = X^\top Y$$

so we may derive the following:

$$X(n\lambda I + X^\top X)\theta = XX^\top Y$$

$$n\lambda(Y - n\lambda\alpha) + XX^\top(Y - n\lambda\alpha) = XX^\top Y$$

$$n\lambda Y - (n\lambda I + XX^\top)n\lambda\alpha = 0$$

$$\alpha = (n\lambda I + K)^{-1}Y$$

Gram matrix K with entries $K_{ij} = x^{(i)} \cdot x^{(j)}$

LINEAR REGRESSION

Prediction.

Moreover,

$$n\lambda X^\top \alpha = X^\top Y - X^\top X\theta$$

$$= (n\lambda I + X^\top X)\theta - X^\top X\theta = n\lambda\theta$$

So $\theta = X^\top \alpha = \sum_{(x',y')} \alpha_{x',y'}\, x'$. Therefore,

$$y = \theta \cdot x = \sum_{(x',y')} \alpha_{x',y'}\, x \cdot x'$$

KERNEL LINEAR REGRESSION

Gram matrix K with entries $K_{ij} = K\left(x^{(i)}, x^{(j)}\right)$

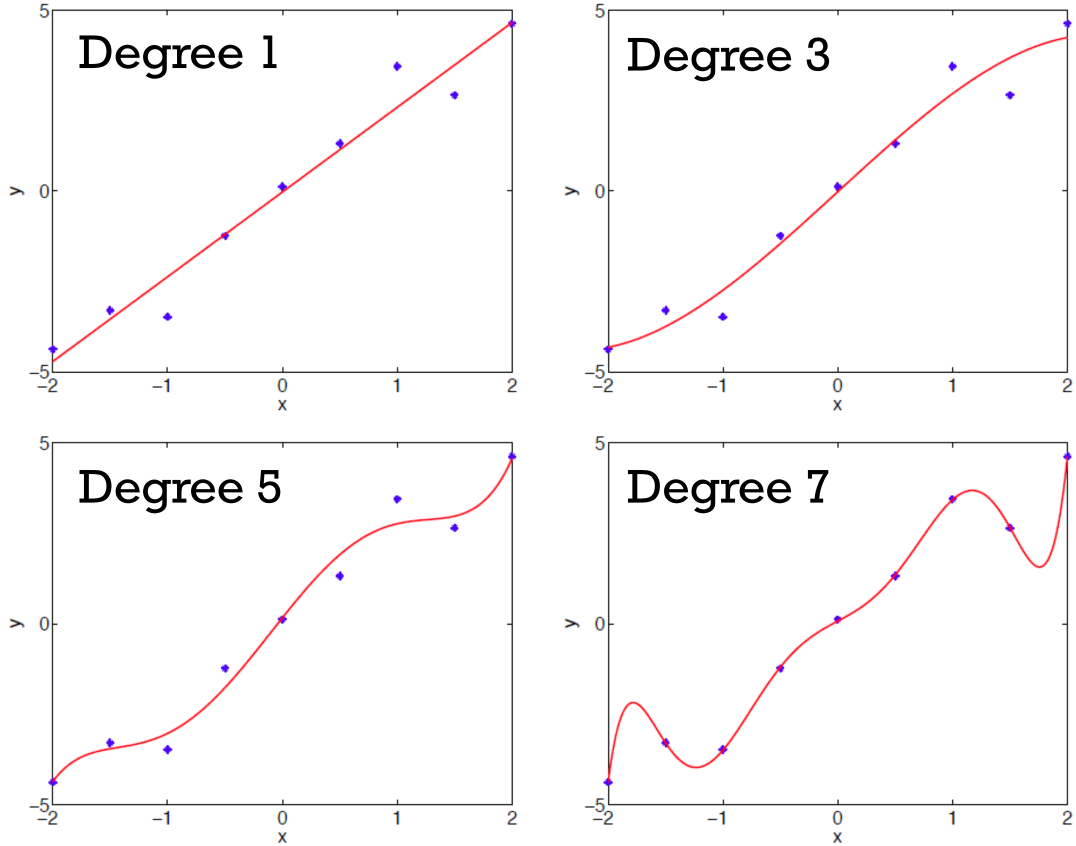
Learning.

$$\alpha = (n\lambda I + K)^{-1} Y$$

Prediction.

$$y = \sum_{(x',y')} \alpha_{x',y'} K(x, x')$$

KERNEL LINEAR REGRESSION



SUMMARY

- Kernel Functions
  - Feature Maps
  - Inner Products
  - Polynomial Kernel
  - Radial Basis Kernel
- Kernel Trick
  - Support Vector Machines
  - Perceptron
  - Linear Regression