LECTURE NOTE #9

PROF. ALAN YUILLE

1. Kernel Trick

Note that the final classifier of an SVM depends on $\underline{x}$ only by dot products. The final classifier is $\hat{y}(\vec{x}) = sign(\sum_i \alpha_i y_i \vec{x}_i \cdot \vec{x}$. This depends on $\vec{x}$ only by: (i) the dot product $\underline{x} \cdot \underline{x}_\mu$, and (ii) the α's depend on solving the dual problem (maximizing the dual) which again depends only of the dot products of the data $\vec{x}_i \cdot \vec{x}_j$.

This motivates the Kernel Trick

Compute features $\underline{\varphi}(\underline{x})$ and reformulate the problem in feature space – i.e. seek a classifier of form:

$$sign(\underline{c} \cdot \underline{\varphi}(\underline{x}) + b)$$

Replace $\underline{x}$ by $\underline{\varphi}(\underline{x})$ everywhere in the primal & dual formulation. Then the classifier only depends on the dot product of the $\underline{\varphi}(\underline{x})$'s:

I.e. on the Kernel $K(\underline{x}, \underline{x}') = \underline{\varphi}(\underline{x}) \cdot \underline{\varphi}(\underline{x}')$

2. Why does this help?

First, using features $\phi(.)$ can make it possible to classify data by hyperplanes, which we could not classify in the original space.

Example

Logical X-OR, $\underline{x} = (x_1, x_2), x_j \in \{\pm 1\}, \omega \in \{\pm 1\}$

The X-OR (exclusive or), see figure (1), requires a decision rule

$\alpha(\underline{x})$ s.t
$\alpha(1,1) = \alpha(-1,-1) = 1$
$\alpha(1,-1) = \alpha(-1,1) = -1$

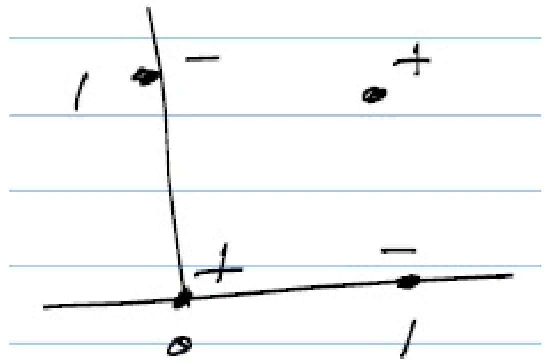

FIGURE 1. Data for the logical X-or problem. It is impossible to separate the positive and negative examples by a straight line (i.e. to classify them correctly by a linear classifier). But we can find features which will enable us to do this.

It is impossible to find a linear classifier to do this. But define feature $\underline{\varphi}(x_1, x_2) = (x_1, x_2, x_1 x_2)$. Now the classifier sign $\{(0,0,1) \cdot \underline{\varphi}(x_1, x_2)\}$ can separate the data.

Moral : increasing the dimensionality of the data by features, makes it possible to find separating hyperplanes.

Second, we do not need to specify the features $\underline{\varphi}(\underline{x})$ explicitly, we only need to specify the kernel

$K(\underline{x}, \underline{x}') = \underline{\varphi}(\underline{x}) \cdot \underline{\varphi}(\underline{x}')$

Remember: the dual problem reduces to maximizing

$$L_d(\{\alpha_\mu\}) = \sum_\mu \alpha_\mu - \frac{1}{2} \sum_{\mu,\nu} \alpha_\mu \alpha_\nu \omega_\mu \omega_\nu \underline{\varphi}(\underline{x}_\mu) \cdot \underline{\varphi}(\underline{x}_\nu)$$

$$= \sum_{\mu} \alpha_\mu - \frac{1}{2} \sum_{\mu,\nu} \alpha_\mu \alpha_\nu \omega_\mu \omega_\nu K(\underline{x}_\mu, \underline{x}_\nu)$$

The solution is $\underline{\hat{a}} = \sum_\mu \underline{\hat{\alpha}}_\mu \omega_\mu \underline{\varphi}(\underline{x}_\mu)$
$\underline{\hat{a}} \cdot \underline{\varphi}(\underline{x}) = \sum_\mu \underline{\hat{\alpha}}_\mu \omega_\mu \underline{\varphi}(\underline{x}) \cdot \underline{\varphi}(\underline{x}_\mu) = \sum_\mu \underline{\hat{\alpha}}_\mu \omega_\mu K(\underline{x}, \underline{x}_\mu)$

(Can solve for $\hat{\sigma}$ as before)

3. What kernels to use?

There are many choices of kernels. The difficulty is knowing which one to use. As always, cross-validation is useful for checking whether a kernel can generalize.

$K(\underline{x}, \underline{x}') = \{1 + \underline{x} \cdot \underline{x}'\}^d$

$K(\underline{x}, \underline{x}') = e^{-\frac{1}{\sigma^2}|\underline{x} - \underline{x}'|^2}$

$K(\underline{x}, \underline{x}') = \tanh\{C_1 \underline{x} \cdot \underline{x}' + C_2\}$

Choice of best kernel is problem dependent.

Some kernels$\rightarrow$ e.g. $\{1 + \underline{x} \cdot \underline{x}'\}^d$ naturally generalized the idea of hyperplanes.

Others $\rightarrow$ e.g. $e^{-\frac{1}{\sigma^2}|\underline{x} - \underline{x}'|^2}$ are similar to nearest neighbors.

4. When do Kernels Correspond to Features?

Suppose we specify $K(\underline{x}, \underline{x}')$, is it equal to $\underline{\varphi(\underline{x})} \cdot \underline{\varphi(\underline{x}')}$ for some features $\underline{\varphi(\underline{x})}$?

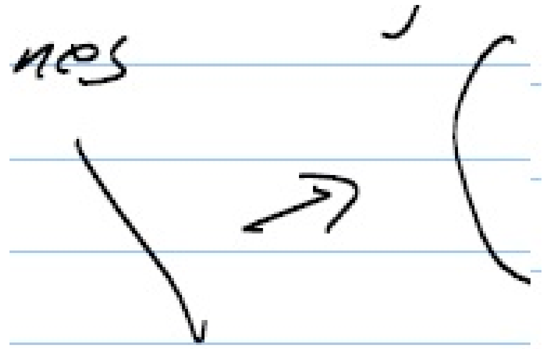

FIGURE 2. One type of kernel, e.g. $\{1+\vec{x}\cdot\vec{x}'\}^d$, corresponds to using curved surfaces to separate the data. The other type of kernel, $\exp\{-|\vec{x}-\vec{x}'|^2\}$ is like nearest neighbour.

Theoretical results can be obtained.

e.g. Mercer's Theorem

Compute <u>eigenfunctions</u> of $K(\underline{x},\underline{x}')$

$\int K(\underline{x},\underline{x}')\psi(\underline{x}')d\underline{x}' = \lambda\psi(\underline{x})$ with $\int\{\psi(\underline{x})\}^2 d\underline{x}$ finite.

Provided $K(\underline{x},\underline{x}')$ is positive definite, then the features are $\varphi^\mu(\underline{x}) = \sqrt{\lambda_\mu}\psi_\mu(\underline{x})$

<u>Similar</u> to linear algebra expansion of a symmetric matrix in terms of eigenvectors.

$A_{ij} = \sum_\mu \lambda_\mu e_i^\mu e_j^\mu$, where $\sum_j A_{ij} e_j^\mu = \lambda_\mu e_i^\mu$

$\underline{\underline{If}}$ A_{ij} is positive definite.

$A_{ij} = \sum_\mu \{\lambda_\mu^{1/2} e_i^\mu\}\{\lambda_\mu^{1/2} e_j^\mu\} = \sum_\mu \varphi_i^\mu \cdot \varphi_j^\mu$

5. <u>KERNEL PCA</u>

LECTURE NOTE #9

The kernel trick can be applied to an quadratic problem - e.g. PCA

$\underline{\underline{C}} = \frac{1}{m}\sum_{k=1}^{m}(\underline{x}_k - \bar{\underline{x}})(\underline{x}_k - \bar{\underline{x}})^T$

w.l.o.g. $\bar{\underline{x}} = \frac{1}{m}\sum_{k=1}^{m}\underline{x}_k = 0$

Go to feature space

$\underline{x} \to \underline{\varphi}(\underline{x})$

$\to \underline{\underline{C}} = \frac{1}{m}\sum_{k=1}^{m}\underline{\varphi}(\underline{x}_k)\underline{\varphi}^T(\underline{x}_k)$

All non-zero eigenvectors $\underline{e}$ of $\underline{\underline{C}}$ are of form

$\underline{e} = \sum_{j=1}^{m}\alpha_j\underline{\varphi}(\underline{x}_j)$, for some $\{\alpha_j\}$

<u>Substituting</u>: $\underline{\underline{C}}\underline{e} = \lambda\underline{e}$

$\to \frac{1}{m}\sum_{k=1}^{m}\underline{\varphi}(\underline{x}_k)\{\underline{\varphi}(\underline{x}_k)\cdot\underline{e}\} = \lambda\underline{e}$

$\to \frac{1}{m}\sum_{k=1}^{m}\underline{\varphi}(\underline{x}_k)\sum_{j=1}^{m}\alpha_j\{\underline{\varphi}(\underline{x}_k)\cdot\underline{\varphi}(\underline{x}_j)\} = \lambda\alpha_j\underline{\varphi}(\underline{x}_j)$

Equating coefficients of $\varphi(\underline{x}_j)$ gives new eigenvalue equations.

$\frac{1}{m}\sum_j K(\underline{x}_k, \underline{x}_j)\alpha_j = \lambda\alpha_k$

Index $\lambda^\mu, \alpha_k^\mu$

6.

$\frac{1}{m}\sum_j K(\underline{x}_k, \underline{x}_j)\alpha_j^\mu = \lambda^\mu\alpha_k^\mu \quad \mu = 1 \textit{ to } m$

Solving this, gives us the eigenvectors.

$\underline{e}^\mu = \sum_{j=1}^{m}\alpha_j^\mu\underline{\varphi}(\underline{x}_j)$, eigenvalue λ^μ. (depends on φ)

But the projections $\underline{e}^\mu \cdot \underline{\varphi}(\underline{x})$ of the data are

$\underline{e}^\mu \cdot \underline{\varphi}(\underline{x}) = \sum_{j=1}^m \alpha_j^\mu K(\underline{x}_j, \underline{x})$

which is independent of φ and depends only on $K(.,.)$).

<u>Hence</u>:

The projection of the data onto the eigenvectors requires only knowing the kernel$K(\underline{x}_i, \underline{x}_j)$ (i.e. not knowing φ)

Knowledge of the kernel is used twice :

(1) to compute the $\{\alpha_j^\mu\}$

(2) to compute the projections $\underline{e}^\mu \cdot \underline{\varphi}(\underline{x})$