LECTURE 10

BOGDAN ICHIM

Classification with Non-linear Decision Boundaries

Note: We discuss a general mechanism for converting a linear model into a non-linear model.

(1) Enlarge the feature space using quadratic, cubic or higher-order polynomial functions of the predictors:
$$X_1, X_2 \to X_1, X_1^2, X_2, X_2^2, X_1X_2, \dots$$
(2) Train the support vector classifier in the enlarged feature space.

Example: $X_1, X_2 \to X_1, X_1^2, X_2, X_2^2$. Our problem of finding **Max M** suffers the following transformation:

$$\begin{cases} \beta_0, \beta_1, \beta_2, \epsilon_i, i = \overline{1,n} \\ \beta_1^2 + \beta_2^2 = 1 \\ y_n(\beta_0 + \beta_1 X_{n1} + \beta_2 X_{n2}) \geq M(1-\epsilon_n) \\ \epsilon_n \geq 0, \sum_{i=1}^{n} \epsilon_i \leq c, n = \overline{1,N} \end{cases} \implies \begin{cases} \beta_0, \beta_1, \beta_2, \beta_{12}, \beta_{22}, \epsilon_i, i = \overline{1,n} \\ \beta_1^2 + \beta_2^2 + \beta_{12}^2 + \beta_{22}^2 = 1 \\ y_n(\beta_0 \sum_{i=1}^{2} \beta_i X_{ni} + \sum_{i=1}^{2} \beta_{i2} X_{ni}^2) \geq M(1-\epsilon_n) \\ \epsilon_n \geq 0, \sum_{i=1}^{n} \epsilon_i \leq c, n = \overline{1,N} \end{cases}$$

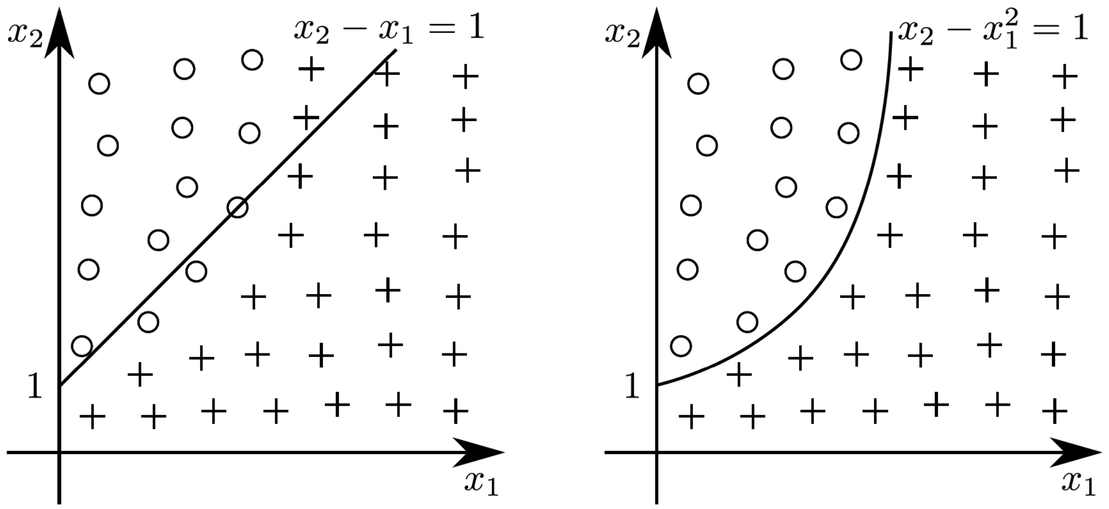


FIGURE 1. Transformation while finding **Max M**

The Support Vector Machine

Main Idea: Enlarging the feature space in a specific way that leads to efficient computations using **kernels**.

The support vector classifier is given by H_β where $\beta(x) = \beta_0 + \sum_{i=1}^{p} \beta_i x_i$.

Assume that $x_i, i = \overline{1, N}$ is a set of generators for $\mathbb{R}^p$ as a vector space $\Rightarrow \exists \alpha_i, i = \overline{1, N}$ such that $(\beta_1, ..., \beta_p) = \sum_{i=1}^{N} \alpha_i x_i \Rightarrow$

$$\beta(x) = \beta_0 + \langle(\beta_1, ..., \beta_p), x\rangle = \beta_0 + \langle\sum_{i=1}^{N} \alpha_i x_i, x\rangle = \beta_0 + \sum_{i=1}^{N} \alpha_i \langle x_i, x\rangle$$

In order to solve the optimization problem which gives the support vector classifier, that is to estimate the parameters $\alpha_i, i = \overline{1, N}$ and β_0, we *only* need the $\binom{N}{2}$ *scalar products* $\langle x_i, x_j\rangle$ between all pairs of training observations.

Moreover, it turns out that $\alpha_i \neq 0$ only for the support vectors:

$$\beta(x) = \beta_0 + \sum_{i \in S} \alpha_i \langle x_i, x\rangle, \text{where } S \subset N$$

So all that we need are the scalar products!

We can replace them each time they appear in the computation.

Kernel = function which is a generalization of the scalar product.

Examples:

(1) The **usual scalar product** in $\mathbb{R}^p$:

$$K(x_n, x_m) = \sum_{j=1}^{p} x_{nj} x_{mj}$$

(2) **Polynomial kernel** of degree d:

$$K(x_n, x_m) = (1 + \sum_{j=1}^{p} x_{nj} x_{mj})^d,$$

where $d \in \mathbb{N}^*$ is the tuning parameter.

(3) The **radial kernel**:

$$K(x_n, x_m) = e^{-\gamma \sum_{j=1}^{p} (x_{nj} - x_{mj})^2},$$

where $\gamma > 0$ is the tuning parameter.

The classifier given by:

$$f(x) = \beta_0 + \sum_{i \in S} \alpha_i K(x_i, x)$$

is called a **support vector machine**.

Remark 1. Using kernels one needs only to compute $K(x_n, x_m)$ for all distinct $\binom{N}{2}$ distinct pairs (n, m) in $\overline{1, N}$ without working explicitly in the enlarged feature space $\Rightarrow$ faster computations.

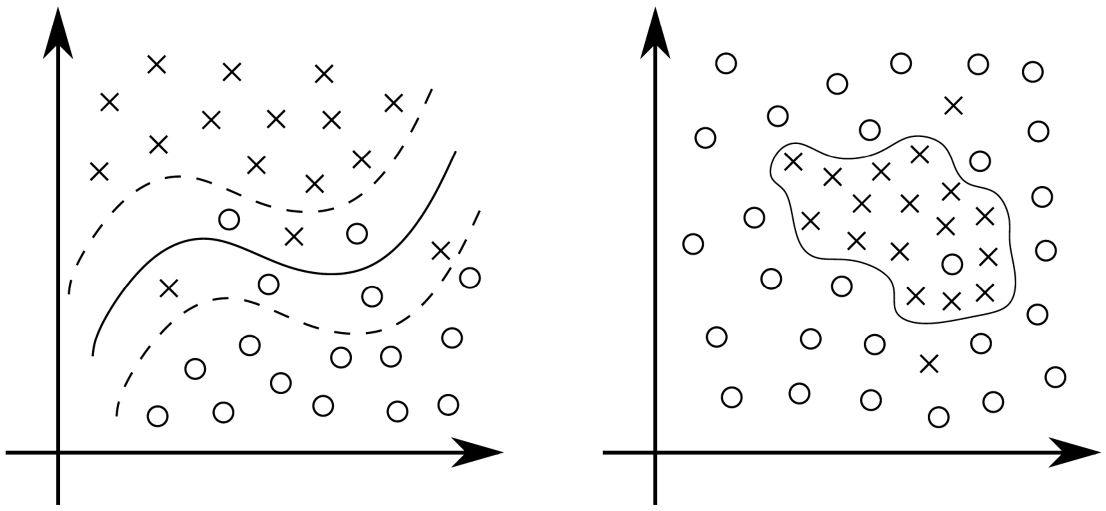


FIGURE 2. Left figure: polynomial kernel. Right figure: radial kernel

Multi-Class SVM Classsifier

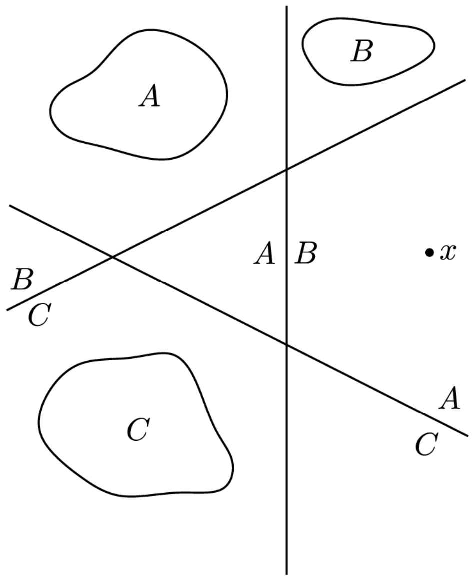
Assume that the number of classes $K > 2$. We have two possibilities:

One-vs-One Classification:

- We consider all possible pairs of classes, thus $\binom{K}{2}$ pairs. We then construct $\binom{K}{2}$ SVMs, each comparing a pair.
- We classify a test observation using each classifier and we assign it to the class that wins the most pairwise classifications.

Remark 2. This is very similar to the Condorcet majority vote!

Example: $k = 3 \Rightarrow \binom{3}{2} \Rightarrow 3$ SVMs
Let A, B, C be the classes and $(A|B), (B|C), (C|A)$ the SVMs.
What if $(A|B)(x) = B, (B|C)(x) = C, (C|A)(x) = A$.



	A	B	C
A	▨	$-$	$+$
B	$+$	▨	$-$
C	$-$	$+$	▨

FIGURE 3. An instance of Condorcet paradox

One-vs-All Classification:

We fit K SVMs, each time comparing one class to all remaining $K - 1$ classes, so we get K classifiers:

$$f_k(x) = \beta_{0k} + \sum_{i_k \in S_k} \alpha_{ik} \mathcal{K}(x_{ik}, x), k = \overline{1, K}$$

and we set $f_k(x) > 0$ for x in the k^{th} class. Then we assign a test observation x to the class k for which $f_k(x)$ is the largest.

Support Vector Regression

(Extension of SVM for Regression)

Main Idea: Minimize only residuals larger in absolute value than some positive constant.