POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Week 10-1

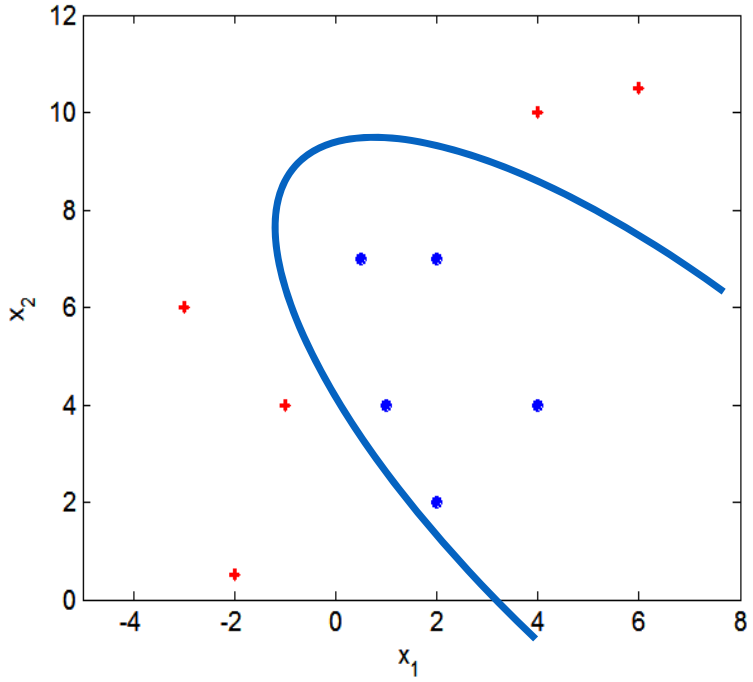
Machine Learning 5: Nonlinear SVM, Multiclassification, Ranking SVM

Big Data

Prof. Hwanjo Yu
POSTECH

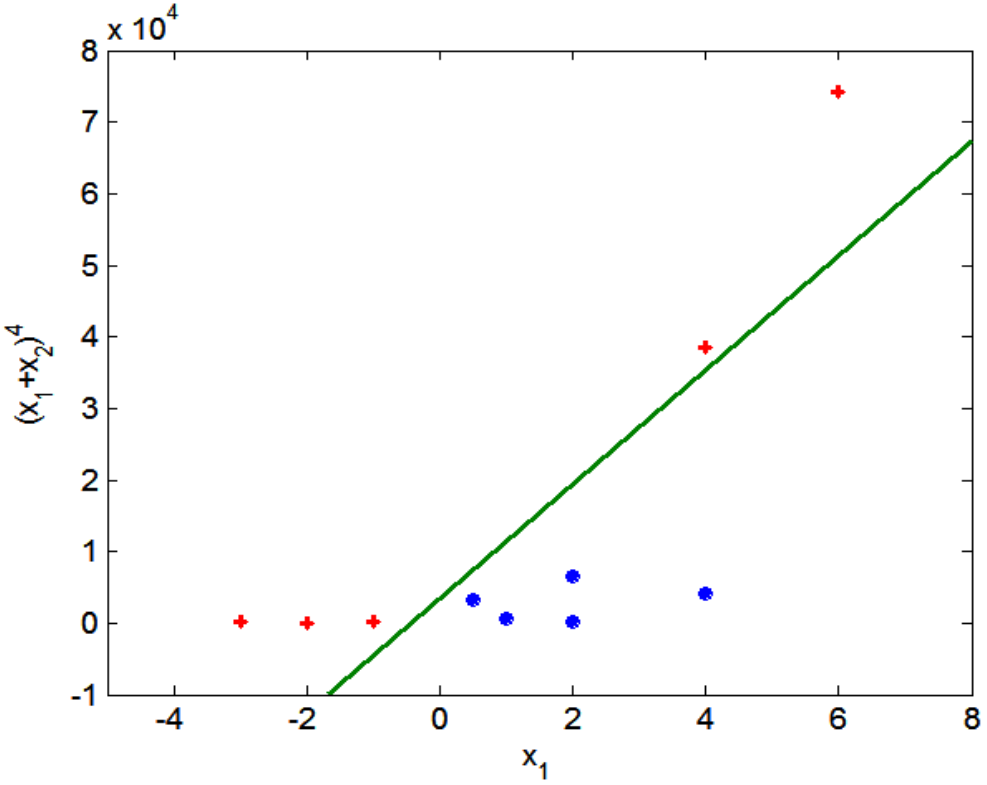
Nonlinear support vector machines

- What if decision boundary is not linear?



Nonlinear support vector machines

- Transform data into higher dimensional space



Nonlinear support vector machines

- A naive way
  - Transform data into higher dimensional space
  - Compute a linear boundary function in the new feature space
  - the boundary function becomes nonlinear in the original feature space

    => very time consuming though.

- SVM “Kernel trick”
  - Does all of these without explicitly transforming data into higher dimensional space

Nonlinear support vector machines

- Dual form:

$$\text{max. } M(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{m} \alpha_i \alpha_j y_i y_j X_i^T X_j$$

$$\text{Subject to } C \geq \alpha_i \geq 0, \ \sum_{i=1}^{m} \alpha_i y_i = 0$$

$$F(Z) = b + \sum \alpha_i y_i X_i Z \quad \rightarrow \quad K(X_i, X_j) = \phi(X_i)^T \phi(X_j)$$

Example for ϕ(.) and K(.,.)

- Suppose ϕ(.) is given as follows

$$\phi\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = \left(1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2\right)$$

original feature space

new feature space

- An inner product in the feature space is

$$\left\langle \phi\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right), \phi\left(\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}\right) \right\rangle = (1 + x_1y_1 + x_2y_2)^2$$

- So, if we define the kernel function as follows, there is no need to carry out f(.) explicitly

$$K(X, Y) = (1 + x_1y_1 + x_2y_2)^2$$

- This use of kernel function to avoid carrying out f(.) explicitly is known as the kernel trick

Kernel functions

- In practical use of SVM, the user specifies the kernel function; the transformation $\phi(.)$ is not explicitly stated

- Another view: kernel function, being an inner product, is really a similarity measure between the objects

Examples of kernel functions

- Polynomial kernel with degree p

$$K(X, Y) = \left(X^T Y + 1\right)^p$$

- Radial basis function (RBF) kernel with width σ

$$K(X, Y) = \exp\left(-\frac{\|X - Y\|^2}{2\sigma^2}\right)$$

  - Closely related to kNN model and RBF neural networks
  - The feature space is infinite-dimensional

Modification due to kernel function

- Change all inner products to kernel functions
- For training,

Original

$$\text{max. } M(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{m} \alpha_i \alpha_j y_i y_j X_i^T X_j$$

$$\text{Subject to } C \geq \alpha_i \geq 0, \ \sum_{i=1}^{m} \alpha_i y_i = 0$$

With kernel function

$$\text{max. } M(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{m} \alpha_i \alpha_j y_i y_j K(X_i, X_j)$$

$$\text{Subject to } C \geq \alpha_i \geq 0, \ \sum_{i=1}^{m} \alpha_i y_i = 0$$

Modification due to kernel function

- For testing, the new data Z is classified as class 1 if $F > 0$, and as class -1 if $F < 0$

Original

$$W = \sum_{j=1}^{s} \alpha_j y_j X_j$$

$$f = W^T Z + b = \sum_{j=1}^{s} \alpha_j y_j X_j^T Z + b$$

With kernel function

$$W = \sum_{j=1}^{s} \alpha_j y_j \phi(X_j)$$

$$f = \langle W, \phi(Z) \rangle + b = \sum_{j=1}^{s} \alpha_j y_j K(X_j, Z) + b$$

More on kernel functions

- Since the training of SVM only requires the value of $K(Xi, X_j)$, there is no restriction of the form of X_i and X_j
  - X_i can be a sequence or a tree, instead of a feature vector

- $K(X_i, X_j)$ is just a similarity measure comparing X_i and X_j
- For a test object Z, the discriminant function essentially is a weighted sum of the similarity between Z and a pre-selected set of objects (the support vectors)

$$f(Z) = \sum_{j=1}^{s} \alpha_j y_j K(X_j, Z) + b$$

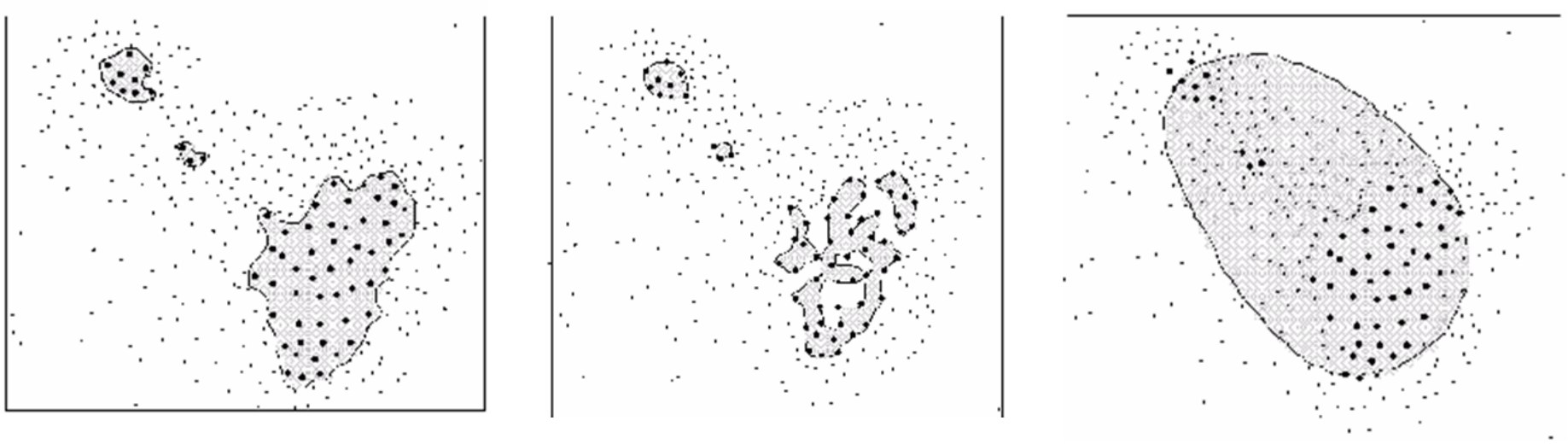
Modification due to kernel function

- Not all similarity measure can be used as kernel function
  - The kernel function needs to satisfy the Mercer function, i.e. the function is “positive-definite”
  - This implies that the *m*-by-*m* kernel matrix, in which the (i,j)-th entry is the $K(X_i, X_j)$, is always positive definite
  - This also means that the QP is convex and can be solved in polynomial time

SVM model with RBF kernel

- As σ moves, the boundary shape changes

$$K(X,Y) = \exp(-\frac{\|X-Y\|^2}{2\sigma^2})$$



Tuning parameters in SVM

- Soft margin parameter C
  - SVM becomes “soft”, as C approaches to zero
  - SVM becomes “hard”, as C goes up
- Polynomial kernel parameter p in $K(X,Y) = \left(X^T Y + 1\right)^p$
  - As p increases, boundary becomes more complex
  - Start with $p = 1$ and increase p by 1
  - The generalization performance will stop improving at some point
- RBF kernel parameter σ in $K(X,Y) = \exp(-\frac{\|X-Y\|^2}{2\sigma^2})$
  - Start with a small number like 2^{-6}, and multiply by 2 at each iteration up to 2^6
  - Once you find a good range of σ, you can tune it more finely within the range.

SVM implementation

- LIBSVM
  - Highly optimized implementation
  - Provide interfaces to other languages such as Python, Java, etc.
  - Support various types of SVM such as multi-class classification, nu-SVM, one-class SVM, etc.
  - Support very fast linear SVM called LibLinear
- SVM-light
  - One of the earliest implementations thus also widely used
  - Support “Ranking SVM” – we will discuss later

Multi-Class Classification

Big Data

Multiclass classification

- A multiclass classification can be reduced into a set of binary classifications.

- Two representative ways
  1. One-to-all
  2. Pairwise coupling(or one-to-one)

One-to-all and pairwise coupling (or one-to-one)

One-to-all

- For k-class classification, create k binary classifiers
  - Each binary classifier is trained from one class as positive and the others as negative
  - Classify new object by taking the majority vote of the classifiers
- Less number of binary classifiers will be created
- Size of training set for each classifier is larger

Pairwise coupling (or one-to-one)

- Create a binary classifier for each pair from k classes
  - ${}_kC_2$ binary classifiers will be created
  - Each classifier is constructed from two classes of data – one as positive and the other as negative
  - Classify new object by taking the majority vote of the classifiers
- More number of binary classifiers will be created
- Size of training set for each classifier is smaller

Ranking SVM (RankSVM)

Big Data

Rank learning

- Ranking model
  - Input: a set of partial orders $\{(Xi > Xj)\}$
  - Output: $F(X)$ such that $F(Xi) > F(X_j)$ when X_i is ranked higher than X_j
- For now, assume F is linear, e.g. X is a house object
  - $F(X) = W \cdot X$ (e.g. w_1*price$(x_1) + w_2$*size(x_2)+..)
- Denote $(X_i, X_j) \in \mathrm{R}$ when X_i is ranked higher than X_j according to an ordering R
- R* is the desired ordering of data (which is unknown to the user.)
- Train F (or compute W) from partial orders R'$(\subset \mathrm{R}^*)$
  - $\forall (X_i, X_j) \in$ R': $F(Xi) > F(X_j) \Leftrightarrow W \cdot Xi > W \cdot Xj$
  - If there exists such W, then R' is *linearly rankable.*

Rank learning

- Objective
  - $\forall (X_i, X_j) \in \mathrm{R}'$: $F(X_i) > F(X_j) \Leftrightarrow W \cdot X_i > W \cdot X_j$

  - $\forall (X_i, X_j) \in \mathrm{R}^*$: $F(X_i) > F(X_j) \Leftrightarrow W \cdot X_i > W \cdot X_j$

- We want to train F (or compute W) from partial orders $\mathrm{R}' (\subset \mathrm{R}^*)$, such that F is concordant with R' and also generalize well beyond R' to order unseen data with respect to R*
- Generalization performance: performance on unseen data

RankSVM

$$\text{minimize}: L(W, \xi_{ij}) = \frac{1}{2} W \cdot W + C \sum \xi_{ij}$$

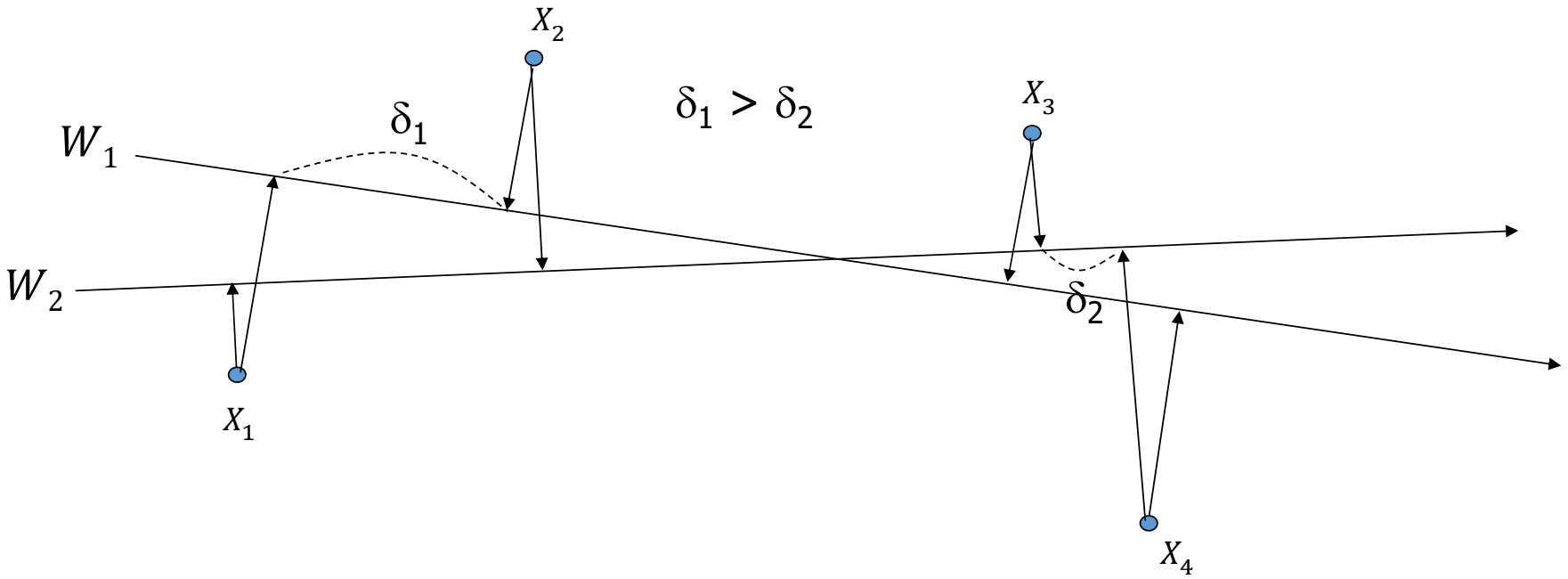
$$\text{subject to}: \forall (X_i, X_j) \in \mathrm{R}' : W \cdot X_i \geq W \cdot X_j + 1 - \xi_{ij}$$

$$\forall (i, j): \xi_{ij} \geq 0$$

- The constraints are to satisfy the training data
- The objective is to maximize the generalization performance

RankSVM

- $\forall(X_i, X_j) \in \mathrm{R}^*: F(X_i) > F(X_j) \Leftrightarrow W \cdot X_i > W \cdot X_j$
- $\forall(X_i, X_j) \in \mathrm{R}': W \cdot X_i \geq W \cdot X_j + 1 - \xi_j \Leftrightarrow W \cdot (X_i - X_j) \geq 1 - \xi_j$



- Both W_1 and W_2 rank $X_1 > X_2 > X_3 > X_4$, but which one is likely to generalize better?

RankSVM

- Ranking SVM learns $F(X)$, from partial orders R', that returns a ranking score of X, such that
  - F tries to satisfy R'
  - F tries to maximizes the minimal ranking difference in feature space

- Ranking SVM also supports kernel trick so that F becomes a nonlinear function

- SVM-light has an implementation of ranking SVM

Summary

- Classification and prediction are two forms of data analysis that can be used to extract models describing important data classes or to predict future data trends.
- Linear, nonlinear, and generalized linear models of regression can be used for prediction. Regression trees and model trees are also used for prediction.
- Stratified k-fold cross-validation is a recommended method for accuracy estimation. Bagging and boosting can be used to increase overall accuracy by learning and combining a series of individual models.

Remark

- Effective and scalable methods have been developed for decision trees induction, Naive Bayesian classification, Bayesian belief network, rule-based classifier, Backpropagation, Support Vector Machine (SVM), pattern-based classification, nearest neighbor classifiers, and case-based reasoning, and other classification methods such as genetic algorithms, rough set and fuzzy set approaches.
- There have been numerous comparisons of the different classification and prediction methods, and no single method has been found to be superior over all others for all data sets. Issues such as accuracy, training time, robustness, interpretability, and scalability must be considered and can involve trade-offs, further complicating the quest for an overall superior method