Advanced Machine Learning

Lecture 2: Classification

Nora Ouzir : nora.ouzir@centralesupelec.fr
Lucca Guardiola : lucca.guardiola@centralesupelec.fr

Oct. - Nov. 2020

CentraleSupélec

Content

Today's Lecture

1. Introduction

2. Logistic Regression

3. Support Vector Machines
   1. Reminders on linear SVMs
   2. Handling non-linear boundaries: Kernel Machines
   3. Multiclass Extension

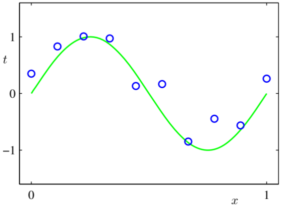
Today's course

1. Introduction
2. Logistic Regression
3. Support Vector Machines
    1. Reminders on linear SVMs
    2. Handling non-linear boundaries: Kernel Machines
    3. Multiclass Extension

Regression vs Classification

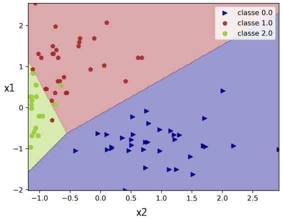
Regression

- $y \in \mathbb{R}$ is a continuous variable
- Predict a numerical value



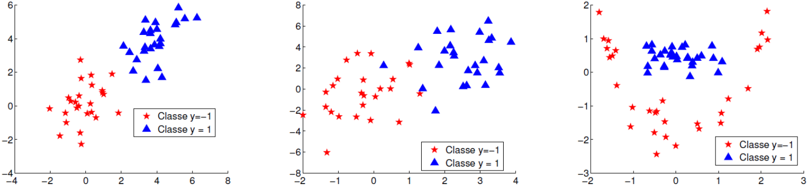
Classification

- labels are discrete variables
- Binary Classification
  $y \in \{0, 1\}, y \in \{-1, 1\}, \ldots$
- Multiclass $y \in \{1, \ldots, K\}$



Linear classification

Linearly separable data: their exists a separating hyperplane which classifies correctly the samples.



Applications

- Sentiment analysis from text features
- Handwritten digits recognition
- Gene expression data classification
- Object recognition in images
- Marketing data: is a client going to buy the product?

Today's course

1. Introduction
2. Logistic Regression
3. Support Vector Machines
    1. Reminders on linear SVMs
    2. Handling non-linear boundaries: Kernel Machines
    3. Multiclass Extension

Logistic Regression: Principle

Goal: Learn linear functions $f_k(\cdot)$ dividing the input space into a collection of K regions.

- Map a linear function on $\Pr(G = k|X = x) \sim$ linear regression
- More generally, map a linear function to a transformation of $\Pr(G = k|X = x)$

Key idea

Learn directly a scoring function

$$f(x) = \log\left(\frac{P(G = 1)|X = x)}{1 - P(G = 0)|X = x)}\right)$$

- Avoid to learn the conditional distributions $p(x/y)$ and the prior $p(y)$ to get the posterior probabilities $P(y/x)$

Logistic Regression: Model

Use an increasing monotone function $\mathbb{R} \mapsto [0, 1]$

$$
\begin{aligned}
\log \frac{\Pr(G = 1 | X = x)}{\Pr(G = K | X = x)} &= \beta_{10} + \boldsymbol{\beta}_1^\top x \\
\log \frac{\Pr(G = 2 | X = x)}{\Pr(G = K | X = x)} &= \beta_{20} + \boldsymbol{\beta}_2^\top x \\
&\vdots \\
\log \frac{\Pr(G = K-1 | X = x)}{\Pr(G = K | X = x)} &= \beta_{(K-1)0} + \boldsymbol{\beta}_{K-1}^\top x
\end{aligned}
$$

- Describe the probability ratios by a linear function of the parameters
- System of $K - 1$ equations

Logistic Regression: Model

Use an increasing monotone function $\mathbb{R} \mapsto [0, 1]$
$\Rightarrow$ For every $k = 1, \ldots, K-1$,

$$\Pr(G = k|X = x) = \frac{\exp(\beta_{k0} + \boldsymbol{\beta}_k^\top x)}{1 + \sum_{\ell=1}^{K-1} \exp(\beta_{\ell 0} + \boldsymbol{\beta}_\ell^\top x)}$$

and

$$\Pr(G = K|X = x) = \frac{1}{1 + \sum_{\ell=1}^{K-1} \exp(\beta_{\ell 0} + \boldsymbol{\beta}_\ell^\top x)}$$

Logistic Regression: Model

Use an increasing monotone function $\mathbb{R} \mapsto [0, 1]$
$\Rightarrow$ For every $k = 1, \ldots, K-1$,

$$\Pr(G = k|X = x) = \frac{\exp(\beta_{k0} + \boldsymbol{\beta}_k^\top x)}{1 + \sum_{\ell=1}^{K-1} \exp(\beta_{\ell 0} + \boldsymbol{\beta}_\ell^\top x)}$$

and

$$\Pr(G = K|X = x) = \frac{1}{1 + \sum_{\ell=1}^{K-1} \exp(\beta_{\ell 0} + \boldsymbol{\beta}_\ell^\top x)}$$

Objective: Supervised Learning

Using a *training* set of n observations $\mathbf{x}_i$ with known labels g_i

- Learn the parameters $\boldsymbol{\beta}$
- Use the learnt model to predict the unknown label of a new observation $\rightsquigarrow$ classification

$$\hat{k} = \underset{k}{\text{argmax}}\, Pr(G = k|X = x)$$

Logistic Regression: Learning

Minimization of a Loss Function

$$F(\Theta) = \sum_{i=1}^{n} -\log \Pr(G = g_i | X = \mathbf{x}_i; \Theta)$$

where Θ contains all the parameters, and g_i is the class label associated to entry $\mathbf{x}_i$.

Two cases

1. Binary classification $K = 2$
2. Multiclass problem $K > 2$

Logistic Regression: Learning

Minimization of a Loss Function

$$F(\Theta) = \sum_{i=1}^{n} -\log \Pr(G = g_i | X = \mathbf{x}_i; \Theta)$$

where Θ contains all the parameters, and g_i is the class label associated to entry $\mathbf{x}_i$.

Two cases

1. Binary classification $K = 2$
2. Multiclass problem $K > 2$

How do we solve the optimization problem?

Logistic Regression: Binary case

Signed response

$$\forall i = 1, \dots, n \quad y_i = \begin{cases} -1 & \text{if } g_i = 1 \\ +1 & \text{if } g_i = 2 \end{cases}$$

Leads to

$$F(\beta) = \sum_{i=1}^{n} \log(1 + \exp(-y_i \beta^\top \mathbf{x}_i))$$

Logistic Regression: Binary case

Signed response

$$\forall i = 1, \ldots, n \quad y_i = \begin{cases} -1 & \text{if } \ g_i = 1 \\ +1 & \text{if } \ g_i = 2 \end{cases}$$

Leads to

$$F(\beta) = \sum_{i=1}^{n} \log(1 + \exp(-y_i \beta^\top \mathbf{x}_i))$$

Properties of F

- Function F is differentiable
- F is convex $\rightsquigarrow$ global minimizer
- Can use gradient descent! But ....

White board

White board

Logistic Regression: Binary Case

We want to use the IRLS algorithm to minimize

$$F(\beta) = \sum_{i=1}^{n} \log[1 + \exp(\left[\boldsymbol{L}\beta]_i)\right]$$

where $\boldsymbol{L} = -\mathrm{Diag}(\mathbf{y}) \times \boldsymbol{X}$ with $\boldsymbol{X} = [\mathbf{x}_1, ..., \mathbf{x}_n]^T \in \mathbb{R}^{n \times d}$

IRLS algorithm

$$\beta^{k+1} = \beta^k - \Omega(\beta^k)^{-1} \nabla F(\beta)$$

where $\Omega(\beta) = \boldsymbol{L}^T \mathrm{Diag}(w(\boldsymbol{L}\beta))\boldsymbol{L}$

Logistic Regression: Binary Case

- Useful inequality for $f(x) = \log(1 + e^x)$

$$(\forall (x, y) \in \mathbb{R}^2) \quad f(x) \leq f(y) + \dot{f}(y)(x - y) + \frac{1}{2}\omega(y)(x - y)^2$$

with $\dot{f}(y) = \frac{e^y}{1+e^y}$ and $\omega(y) = \frac{1}{y}(\frac{1}{1+e^{-y}} - \frac{1}{2})$

- F is differentiable with $\nabla F(\beta) = \boldsymbol{L}^T \dot{f}(\boldsymbol{L}\beta)$

Logistic Regression: Binary Case

- Useful inequality for $f(x) = \log(1 + e^x)$

$$(\forall (x,y) \in \mathbb{R}^2) \quad f(x) \leq f(y) + \dot{f}(y)(x-y) + \frac{1}{2}\omega(y)(x-y)^2$$

with $\dot{f}(y) = \frac{e^y}{1+e^y}$ and $\omega(y) = \frac{1}{y}\left(\frac{1}{1+e^{-y}} - \frac{1}{2}\right)$

- F is differentiable with $\nabla F(\beta) = \boldsymbol{L}^T \dot{f}(\boldsymbol{L}\beta)$

What if n is very large?

- Need for regularization to avoid over-fitting
- Online minimization technique (*e.g.*, SGD).

Today's course

1. Introduction

2. Logistic Regression

3. Support Vector Machines
   1. Reminders on linear SVMs
   2. Handling non-linear boundaries: Kernel Machines
   3. Multiclass Extension

Today's course

1. Introduction
2. Logistic Regression
3. Support Vector Machines
    1. Reminders on linear SVMs
    2. Handling non-linear boundaries: Kernel Machines
    3. Multiclass Extension

Linear SVM: Problem Formulation

- Training set of pairs $(x_i, y_i), i = 1, .., n$
- $x_i \in \mathbb{R}^d$ and $y \in \{-1, 1\}$

Objective

Find a linear function $f(x) = w^T x + b, \quad w \in \mathbb{R}^d, b \in \mathbb{R}$ that classifies input samples such that

$$f(x) > 0 \quad x \text{ is assigned to class 1}$$
$$f(x) < 0 \quad x \text{ is assigned to class -1}$$

Linear SVM: Problem Formulation

- Training set of pairs $(x_i, y_i), i = 1, .., n$
- $x_i \in \mathbb{R}^d$ and $y \in \{-1, 1\}$

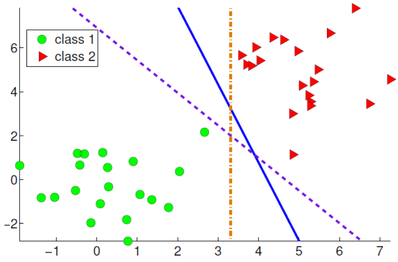
Objective

Find a linear function $f(x) = w^T x + b, \quad w \in \mathbb{R}^d, b \in \mathbb{R}$ that classifies input samples such that

$$f(x) > 0 \quad x \text{ is assigned to class 1}$$
$$f(x) < 0 \quad x \text{ is assigned to class -1}$$

- Classification rule is $\text{sign}(f(x))$

Max Margin Classifier



Best classifier?

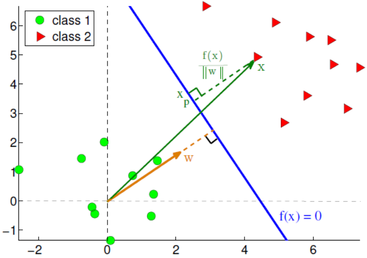
- Decision boundary that is more “stable,” we are confident in all decisions
- We want observations to be as far from the decision boundary as possible

$\rightsquigarrow$ large margin

Max Margin Classifier

The margin is the smallest distance $d(H, x)$ between the boundary (H) and any of the observations

$$d(x_i, H) = \frac{y_i(w^T x_i + b)}{\|w\|} = \frac{|f(x_i)|}{\|w\|}$$

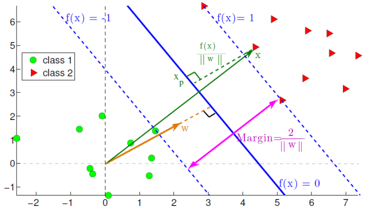


Max Margin Classifier: Canonical Hyperplane

Constraints for the hyperplane: one forces the training samples that are the closest to the boundary to satisfy

$$y_i(w^T x_i + b) = 1 \implies \min_{x_i} |w^T x + b| = 1$$

- The x_i satisfying $y_i(w^T x_i + b) = 1$ are the support vectors



The geometrical margin $M = \frac{2}{\|w\|}$

Linear SVM: Optimization Problem

Goal: Maximize the margin while correctly classifying each sample $\rightsquigarrow$ constrained optimization problem

Primal problem

$$\min_{w,b} \frac{1}{2}\|w\|^2 \quad \text{s.t} \quad y_i(w^T x_i + b) \geq 1, \quad \forall i = 1, \ldots, n$$

Simple problem since the cost function to optimize is quadratic and the constraints are linear!

Linear SVM: Dual problem

Lagrangian formulation

$$L(w, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i [y_i(w^T x_i + b) - 1]$$

- α_i are the Lagrange multipliers, dual variables
- Set derivatives wrt w and b to zero

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \quad \text{and} \quad w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

- Substitute the latter in L

Maximization problem

$$\max_{\alpha} \textstyle\sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \quad \text{s.t } \alpha_i \geq 0, \forall i \quad \text{and} \quad \sum_{i=1}^{n} \alpha_i y_i = 0$$

Linear SVM: solution

Once we have the dual problem...

1. Find the solution $\hat{\alpha}$ (quadratic function to optimize and linear constraints)
2. Compute the weights according to $\hat{w} = \sum_{i=1}^{n} \hat{\alpha}_i y_i x_i$
3. Two scenarios $\begin{cases} x_i \text{ is on the margin} \rightarrow \alpha_i > 0 \\ y_i(w^T x_i + b) > 1 \text{ and } \alpha_i = 0 \end{cases}$

   Only the support vectors play a role in prediction !!!
4. Compute b knowing that $\hat{\alpha}_i > 0$ satisfy $y_i(\hat{w}^T x_i + b) = 1$

Classification function:

$$f(x) = \hat{w}^T x + b = \sum_{i=1}^{n} \hat{\alpha}_i y_i x_i^T x + b$$

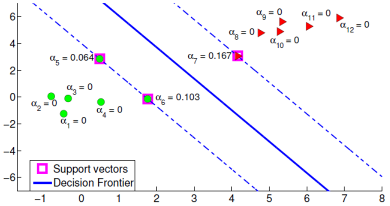
Linear SVMs: summary

In the primal problem

- Predictions are based on the learnt $(n+1)$ values of w and b
- Parametric approach

In the dual formulation...

- Only the support vectors play a role in prediction
- Central in practice because once the model is trained, a significant proportion of datapoints can be discarded



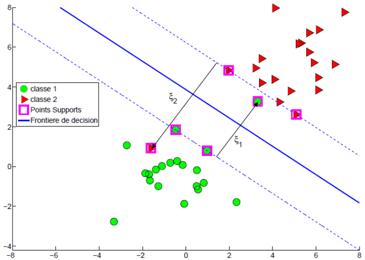
Soft SVM: The overlapping case

Key principle: If the classes are overlapping, we can't learn a perfect linear classifier

- Allow for some error or *slack* : $\xi_i \geq 0$
- The slack relaxes the classification constraint

$$y_i(w^T x_i + b) \geq 1 - \xi_i$$

- Minimize the sum of slacks $\sum_{i=1}^{n} \xi_i$



Soft SVM: Optimization problem

New optimization problem

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \quad \text{s.t} \quad y_i(w^T x_i + b) \geq 1 - \xi_i,\ \xi_i \geq 0\ \forall i$$

- C controls the trade-off between slack errors and margin maximization $\rightarrow$ user defined

Dual problem (after similar computations...)

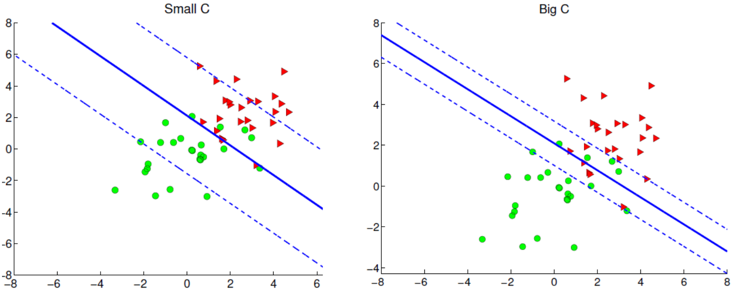
$$\max_{\alpha} \ \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

$$\text{s.t } 0 \leq \alpha_i \leq C, \forall i \quad \text{and} \quad \sum_{i=1}^{n} \alpha_i y_i = 0$$

Soft SVM : Examples

Influence of C

Larger C values penalize the slack more $\rightsquigarrow$ narrow margin



Relation to Logistic Regression

- Penalized problems with different losses : hinge loss and logistic loss
- Both are approximations of (SVM: sparse approximation)

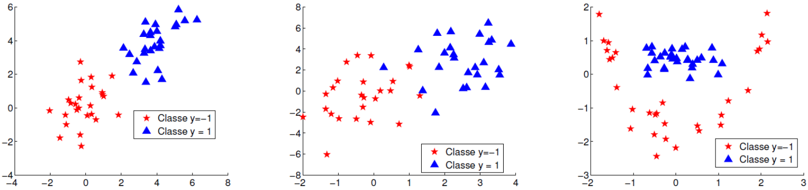
Today's course

1. Introduction

2. Logistic Regression

3. Support Vector Machines
    1. Reminders on linear SVMs
    2. Handling non-linear boundaries: Kernel Machines
    3. Multiclass Extension

Non-linear Boundaries



Linear SVM limitations

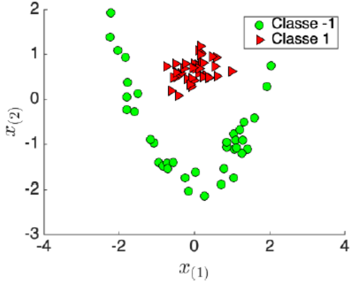
- The decision boundary is not always linear
- Data are not always vectors (e.g., string, time series, graphs, images ...)

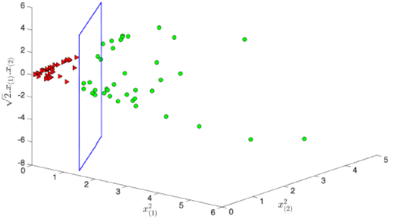
Higher Dimensional Embedding

Key Idea: Data might be linearly separable in a higher dimensional space

- Use a non-linear embedding $\Phi(x) : \mathbb{R}^p \mapsto \mathbb{R}^q$
- Train the SVM using pairs $(\Phi(x_i), y_i)$

(Non-linear transformation $\rightsquigarrow$ linear separabilty)





Example

Consider the binary case

The classes $\mathcal{C}_1 = \{(1,1),(-1,-1)\}$ and $\mathcal{C}_2 = \{(1,-1),(-1,1)\}$ are not linearly separable. Consider the application Φ defined by

$$\Phi : \begin{cases} \mathbb{R}^2 \mapsto \mathbb{R}^6 \\ (x_1, x_2) \mapsto (\sqrt{2}x_1, \sqrt{2}x_1x_2, 1, \sqrt{2}x_2, x_1^2, x_2^2) \end{cases}$$

The data are separable in the plane (Φ_1, Φ_2)

Non-linear SVM: Kernels

The decision function is now

$$f(x) = w^T \Phi(x) + b = \sum_{SV} \alpha_i y_i \Phi(x_i)^T \Phi(x)$$

The kernel trick

- Exploit the inner product in the dual formulation of SVM
- Define a function $k(.\,,.) : \chi \times \chi \mapsto \mathbb{R}$ (similarity in implicit higher dimensional space)
- Replace the inner product between samples by the kernel k
- Independent of the implicit feature dimension!
- $\rightsquigarrow$ reduce computational cost from $O(n^3)$, $O(n^2)$ to $O(n)$ using $k(x, y) = \Phi(x)^T \Phi(y)$

Non-linear SVMs: Kernels

A kernel k is a function $k(.\ ,.) : \chi \times \chi \mapsto \mathbb{R}$ such that

$$k(x, y) = \langle \Phi(x), \Phi(y) \rangle$$

What are the conditions on k?

The kernel must be positive-definite to ensure a well-defined dual problem

1. Symmetric $k(x, y) = k(y, x)$
2. And for any positive integer n

$$\forall \alpha_i \sum_i \sum_j \alpha_i^n \alpha_j^n k(x_i, x_j) \geq 0$$

- The associated Gram matrix $G \in \mathbb{R}^{n \times n}$ $G_{ij} = k(x_i, x_j)$ is positive definite

Common Kernels

Type	Name	$k(\mathbf{x}, \mathbf{z})$
radial	Gaussian	$\exp\left(-\frac{\lVert\mathbf{x}-\mathbf{z}\rVert^2}{2\sigma^2}\right)$
radial	Laplacian	$\exp(-\lVert\mathbf{x}-\mathbf{z}\rVert/\sigma)$
non stat.	χ^2	$\exp(-r/\sigma),\ r = \sum_k \frac{(x_k - z_k)^2}{x_k + z_k}$
projectif	polynomial	$(\mathbf{x}^\top \mathbf{z} + \sigma)^p$
projectif	cosinus	$\mathbf{x}^\top \mathbf{z}/\lVert\mathbf{x}\rVert \lVert\mathbf{z}\rVert$
projectif	correlation	$\exp\left(\frac{\mathbf{x}^\top \mathbf{z}}{\lVert\mathbf{x}\rVert\lVert\mathbf{z}\rVert} - \sigma\right)$

How to choose the right kernel?

Short answer: test it !

- Use cross-validation for the hyperparameters (polynomial order p, bandwidth σ)

Non Linear SVM: kernel formulation

With similar computations of the Lagrangian we obtain...

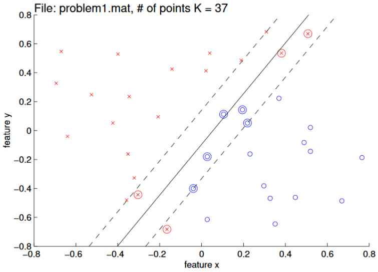
Dual problem

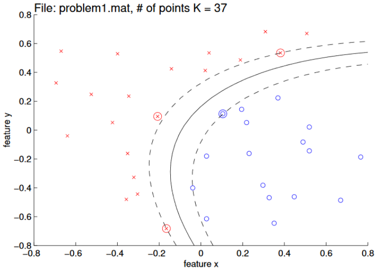
$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$
$$\text{s.t } 0 \leq \alpha_i \leq C, \forall i \quad \text{and} \quad \sum_{i=1}^{n} \alpha_i y_i = 0$$

Classification function

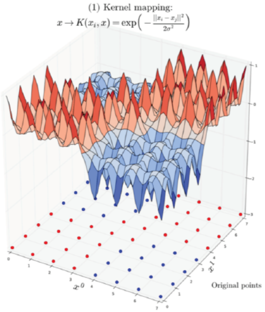
$$f(x) = \sum_{SV} \alpha_i y_i k(x_i, x)$$

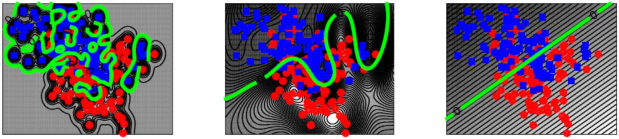
Non-linear SVM: Example





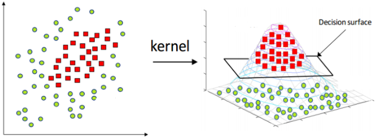
Example with Gaussian Kernel





σ too small | nice σ | σ too large

Non Linear SVM: Summary



- Exploit inner product in dual formulation
- No explicit representation of the non-linear embedding Φ
- Can be defined on any kind of data provided we are able to define a measure of similarity
- Need to save the support vectors : instance based approach (save data rather than parameters)
- In practice: no right way to choose the kernel, cross-validation for the hyperparameters
- In practice: small to moderate datasets

Today's course

1. Introduction

2. Logistic Regression

3. Support Vector Machines
    1. Reminders on linear SVMs
    2. Handling non-linear boundaries: Kernel Machines
    3. Multiclass Extension

Multiclass SVM Extensions

One against all

- Learn K SVM (a class against the others)
- Classify each sample according to the "winner takes all" strategy

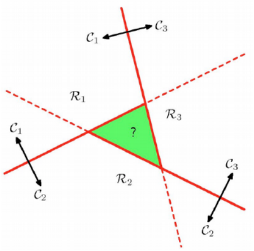
One against one

- Learn $K(K-1)/2$ SVM (one class against another one)
- Classify each sample wih a majority vote
- or estimate the posterior probabilities (pairwise coupling) ; classify according to the maximal posterior probability

The latter method is preferable but if K is too large, the former is to be used!

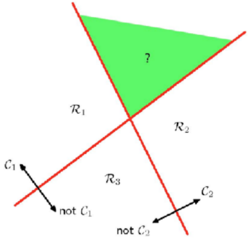
Multiclass Extensions

One Against One



- Learn binary SVMs $f_{kj}(x) = w_{kj}^T x + b_{kj}$
- $y_i = 1$ if $x_i \in C_k$ and -1 if $x_i \in C_j$
- if $f_{kj}(x_i) > 0$ vote for C_k, otherwise C_j
- Assign to max votes

One Against All



- Learn binary SVM for each class $f_k(x) = w_k^T x + b_k$
- $y_i = 1$ if $x_i \in C_k$ and -1 otherwise
- Winner takes it all: $\hat{k} = \underset{k}{argmax}\{f_1, ..., f_k\}$