Sisteme de Recunoastere a Formelor – Lab 12

Clasificare cu Support Vector Machine

1. Obiective

In aceasta lucrare se va implementa clasificatorul SVM liniar si se va studia mecanismele de clasificare bazate pe vectori de suport si clasificatori cu soft margin.

2. Fundamente teoretice

2.1. Clasificatori cu separare stricta (hard-margin)

Pentru a explica ce inseamna un clasificator SVM cu separare stricta se va porni de la un exmplu simplu de problema de clasificare. Se doreste separarea liniara a unei multimi de puncte in spatiul cartezian bidimensional. Un exemplu este prezentat in Fig 1.1.

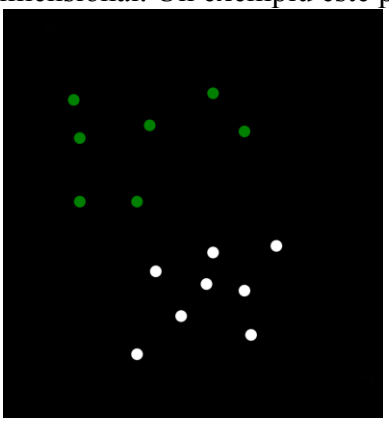


Fig. 1.1 O multime de puncte din doua clase liniar separabile

Cum putem clasifica aceste puncte folosind o functie discriminanta liniara care minimizeaza eroarea de clasificare? Exista un numar infinit de raspunsuri, cateva drepte separatoare cu eroare 0 sunt ilustrate in Fig. 1.2.

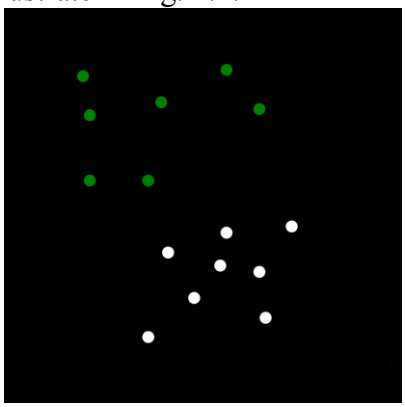


Fig. 1.2 Exemplu de clasificitori liniari cu eroare de clasificare zero

Din aceasta multime de solutii este necesar sa determinam clasificatorul optim. Pentru SVM consideram clasificatorul optim clasificatorul care maximizeaza zona de margine. Zona de margine este definita ca si latimea bandei la care poate fi extinsa linia de separare inainte sa intalnim un punct de date. (vezi Fig. 1.3).

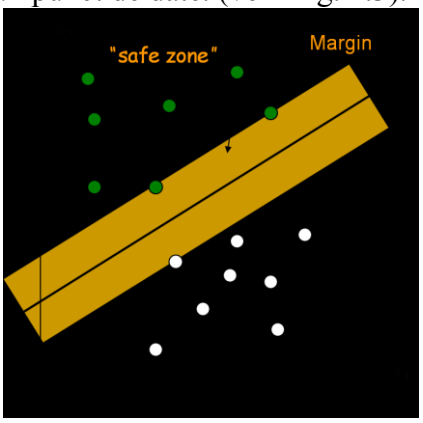


Fig. 1.3 Zona de margine pentru un clasificator liniar

Clasificatorul cu zona de margine maximala este considerata optimala fiindca este robusta la outliers si are o abilitate de generalizare puternica.

Fiind data o multime de puncte etichetate: $\{x_i, y_i\}, i = 1,2, \dots, n$ unde

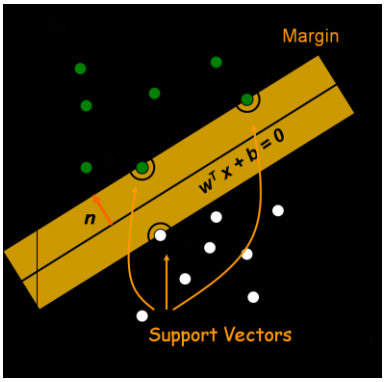
$$y_i = +1, w^T x_i + b > 0$$
$$y_i = -1, w^T x_i + b < 0$$

Folosind o transformare de scala aplicata pe w si b conditiile sunt echivalente cu:

$$y_i = +1, w^T x_i + b \geq 1$$
$$y_i = -1, w^T x_i + b \leq -1$$

Se stie ca:

$$w^T x^+ + b = 1$$
$$w^T x^- + b = -1$$



Latimea zonei de separare este:

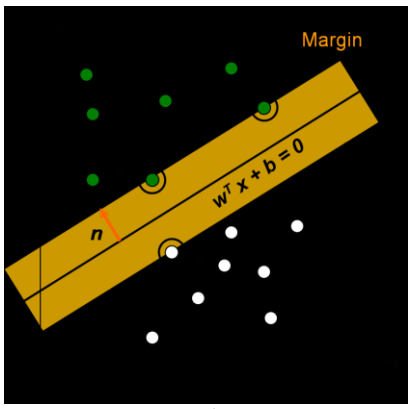
$$M = (x^+ - x^-) \cdot n = (x^+ - x^-) \cdot \frac{w}{||w||} = \frac{2}{||w||}$$

Problema maximizarii acestei expresii este dificila fiindca depinde de norma vectorului w, $||\mathbf{w}||$, care contine radical. Se poate minimiza $\frac{1}{2}||w||^2$ in locul normei, care duce la o problema de optimizare echivalenta dar rezolvabila.

Cu aceasta schimbare rezulta o problema de optimizare patratica:

Sa se minimizeze expresia $\frac{1}{2}\|w\|^2$ cu constrangerile:

- $y_i = +1, w^T x_i + b \geq 1$
- $y_i = -1, w^T x_i + b \leq -1$



O reformulare mai succinta este sa se minimize $\frac{1}{2}\|w\|^2$ cu constrangerile $y_i(w^T x_i + b) \geq 1$. Solutia la aceasta problema de optimizare se gaseste cu metoda multiplicatorilor lui Lagrange.

2.2. Clasificatori cu separare permisiva (soft-margin)

In 1995, Corinna Cortes si Vladimir Vapnik au propus un alt criteriu mai permisiv care permite exemple etichetate gresit. Daca nu exista un hiperplan care separa exemplele in doua clase atunci un clasificator cu separare permisiva va alege hiperplanul care separa datele cat mai corect, maximizand distanta de la cel mai apropiat exemplu corect clasificat. Aceasta metoda introduce variabilele auxiliare, ξ_i, care masoara gradul de clasificare gresita a punctului x_i.

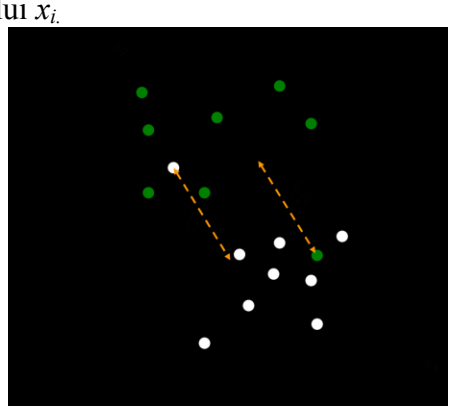


Fig. 1.4 Classificatare cu separare permisiva

Prin minizarea sumei $\sum_i \xi_i$, obtinem ξ_i ca fiind:

$$\begin{cases} \mathbf{w}^T \mathbf{x}_i + b \geq 1 - \xi_i & y_i = 1 \\ \mathbf{w}^T \mathbf{x}_i + b \leq -1 + \xi_i & y_i = -1 \\ \xi_i \geq 0 & \forall i \end{cases}$$

- ξ_i sunt variabilele auxiliare de optimizare;
- ξ_i =0 daca $\mathbf{x}_i$ este clasificat corect, si
- C este numarul maxim de erori

Trebuie sa minimizam $\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i$, supus la constrangerile $y_i(w^T x_i + b) \geq 1 - \xi_i$ si $\xi_i \geq 0$. Parametrul C poate fi vazut ca si un parametru care regleaza eroarea fata de zona de separare.

2.3. Support vector machine cu mapare neliniara

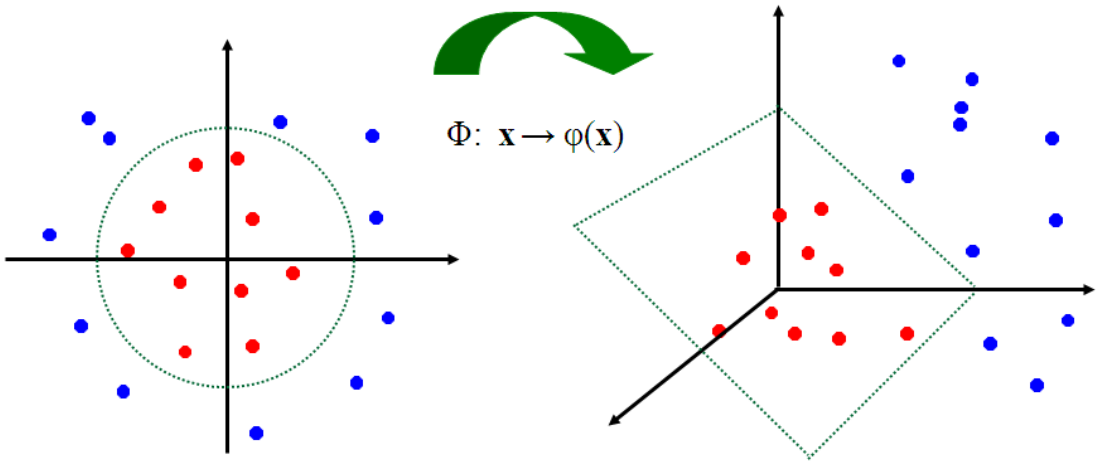
In cazul in care datele nu sunt liniar separabile se poate aplica o transformare $x_i \rightarrow \phi(x_i)$ care mapeaza fiecare punct intr-un spatiu cu dimensionalitate mai mare. Se doreste ca in acest spatiu punctele sa devina liniar separabile.
Fiind data o multime de perechi instanta si eticheta (x_i, y_i); i = 1… l unde $x_i \in R^n$ si $y \in \{+1,-1\}^l$, clasificatorul SVM rezolva urmatoarea problemei de optimizare:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i$$

cu constrangerile:

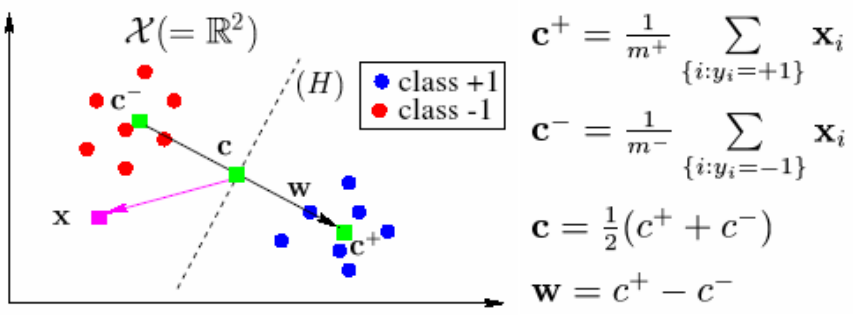
$$y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i,$$
$$\xi_i \geq 0.$$

In acest caz vectorii de antrenare x_i sunt mapati intr-un spatiu cu dimensionalitate mai mare (posibil infinita). Se gaseste hiperplanul separator optimal in acest spatiu. C > 0 este termenul care penalizeaza erorile de clasificare. Se defineste functia $K(x_i,x_j) = \Phi(x_i)^T \Phi(x_j)$ ca si functia de kernel (nucleu). Se poate reformula problema de optimizare astfel incat sa folosim aceasta functie (in loc sa mapam vectorii). In literatura exista o multitudine de functii kernel, se recomanda folosirea urmatoarelor tipuri: liniar, polinomial, radial basis function si sigmoid.



3. Clasificator liniar elementar

Ideea de baza pentru un clasificator elementar este sa se clasifice orice instanta pe baza distantei de la mediile claselor. Se aloca instanta la clasa cea mai apropiata. Pentru aceasta se calculeaza dreapta de separare ca fiind dreapta perpendiculara pe segmentul care uneste mediile claselor si trece prin mijlocul acestui segment.



Pentru a gasi clasa unei instante este suficient sa se calculeze produsul scalar intre w si x-c. Definim clasificatorul simplu ca si f(x) = sign(<w, x-c>). Hiperplanul H (care este o dreapta) este suprafata de separare si are ca si normala vectorul w care uneste mediile claselor. Daca se inlocuieste in produsul scalar avem formula pentru clasificator:

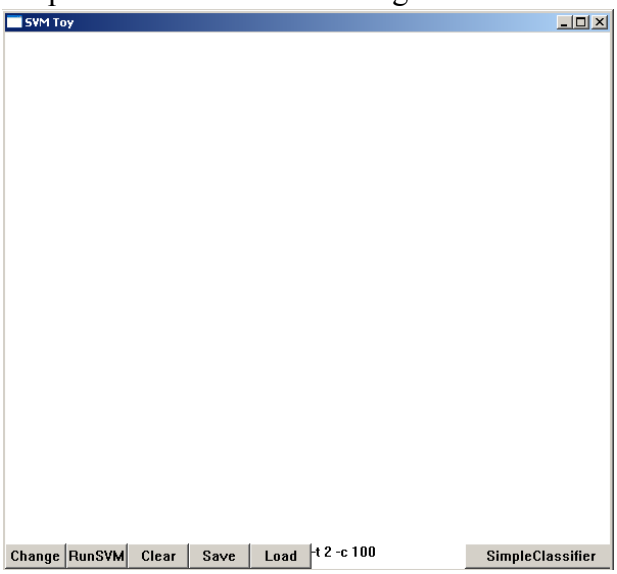
$$h(x) = \sum_{i=1,\ldots,m} \alpha_i \langle x_i, x \rangle + b$$

Unde m este numarul total de exemple, m^+ este numarul exemplelor pozitive, m^- este numarul exemplelor negative, $\alpha_i = 1/m^+$ daca $y_i=1$ si $\alpha_i = -1/m^-$ daca $y_i=-1$ si
$b = \langle c^+,c\rangle - \langle c^-,c\rangle$.

4. Exercitii

Ca si exercitiu veti folosi platforma denumita SVM-toy de la LIBSVM tools, care este o implementare in C++ a clasificatorilor cu separare permisiva folosind diferite functii de kernel.

1. Se descarca fisierul de zip care contine proiectul si se incarca in VisualStudio. Interfata grafica este prezentata in urmatoarea figura:



Butoanele de la interfata au urmatoarele functionalitati:

- ‘Change’: schimba clasa punctului care va fi adaugat in spatiul de clasificare. Se poate adauga un punct nou prin click stanga in zona

ferestrei albe. Numarul maxim de clase este 3 care corespunde la 3 culori diferite

- 'RunSVM': ruleaza antrenarea unui SVM cu parametri specificati in edit boxul de langa 'Load'
- 'Clear': sterge toate punctele din spatiul de clasificare
- 'Save': salveaza punctele din spatiul de clasificare curent intr-un fisier
- 'Load': incarca o poza bmp si deseneaza punctele din spatiul de clasificare
- In edit box se specifica parametri clasificatorului, contine: '–t 2 –c 100' Vom folosi doi parametri:
  - '-t kernel_type' specifica tipul functiei kernel, are valoarea implicita de 2. Se poate alege dintre urmatoarele:
    0 – kernel liniar: u*v
    1 – kernel polinomial: (gamma*u'*v + coef0)^degree
    2 – radial basis function: exp(-gamma*|u-v|^2)
    3 – sigmoid: tanh(gamma*u'*v + coef0)
  - '-c cost' specifica valoarea pentru parametrul C
- 'SimpleClassifier' butonul lanseaza codul care urmeaza sa fie implementat (se schimba *svm-toy.cpp*).

2. Pentru fiecare imagine din arhiva zip se ruleaza clasificatorul SVM cu parametri diferiti si se observa rezultatul.
3. Sa se implementeze 'SimpleClassifier' si sa se compare rezultatele cu un SVM cu kernel liniar.

Codul se introduce in ramura instructiunii case din *svm-toy.cpp*:

```
case ID_BUTTON_SIMPLE_CLASSIFIER:
{
/* ************************************
      TO DO:
      WRITE YOUR CODE HERE FOR THE SIMPLE CLASSIFIER
************************************** */
}
```

Se mentioneaza ca punctele pentru clasificare sunt stocate intr-o lista de puncte:

```
list<point> point_list;
```

cu structura unui punct fiind definita:

```
struct point {
      double x, y;
      signed char value;
};
```

Campul value corespunde la clasa punctelui (are valoarea 1,2 sau 3).

Coordonatele punctelor sunt normalizate intre 0 si 1, iar punctul de origine (0,0) este localizat in coltul de stanga sus.
Dimensiunea spatiului de clasificare este de XLEN * YLEN, in consecinta coordonatele reale corespunzatoare unui punct (x,y) vor fi (x*XLEN, y*YLEN).

Sa se determine *w* si *b* folosind formulele urmatoare:

$$\mathbf{c}^+ = \frac{1}{m^+} \sum_{\{i:y_i=+1\}} \mathbf{x}_i$$

$$\mathbf{c}^- = \frac{1}{m^-} \sum_{\{i:y_i=-1\}} \mathbf{x}_i$$

$$\mathbf{c} = \frac{1}{2}(c^+ + c^-)$$

$$\mathbf{w} = c^+ - c^-$$

$$b = \langle c, c^+ \rangle - \langle c, c^- \rangle$$

si sa se deseneze linia cu ecuatia:

$$\langle w, \boldsymbol{x} \rangle - b = 0, \boldsymbol{x} \in R^2$$

Desenarea unei linii intre doua puncte se face cu metoda:

```
DrawLine(buffer_dc,x1, y1, x2, y2, RGB(255,0,0));
```

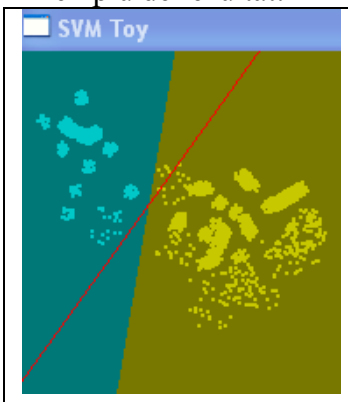
Se parcurge lista de puncte si se numara instantele din clasa 1 si clasa 2 folosind urmatorul cod:

```
//declare an iterator
list<point>::iterator p;
int nrSamples1=0;
int nrSamples2=0;
double xC1=0,xC2=0,yC1=0,yC2=0;

for(p = point_list.begin(); p != point_list.end(); p++)
{
	if ((*p).value==1) //point from class '1'
	{
		nrSamples1++;
		xC1 =(*p).x; //normalized x coordinate of the current point
		yC1 =(*p).y; //normalized y coordinate of the current point

	}
	if ((*p).value==2)  //point from class '2'
	{
		nrSamples2++;
		xC2 =(*p).x; //normalized x coordinate of the current point
		yC2 =(*p).y; //normalized y coordinate of the current point
	}
}
```

Exemplu de rezultat:



Detalii:
- punctele 2D care vor fi clasificate
- 2 clase, 2 trasaturi (x_1 si x_2)
- Separatorul generat de algoritmul 'Simple Classifier' este linia rosie
- Separarea indicata prin culori diferite este cea obtinuta de un SVM cu kernel liniar (-t 0) si cost C=100 (-c 100)

Se observa:
- Marginea maximizata obtinuta cu SVM
- Punctele clasificate gresit cu classificatorul simplu

4. References

[1] Jinwei Gu - *An Introduction to SVM*: *http://www1.cs.columbia.edu/~belhumeur/courses/biometrics/2009/svm.ppt*
[2] J. Shawe-Taylor, N. Cristianini: *Kernel Methods for Pattern Analysis*. Pattern Analysis (Chapter 1)
[3] B. Scholkopf, A. Smola: *Learning with Kernels*. A Tutorial Introduction (Chapter 1), MIT University Press.
[4] LIBSVM: http://www.csie.ntu.edu.tw/~cjlin/libsvm/