Systemy uczące się
Metoda wektorów wspierających
Przekształcenie z funkcjami jądrowymi
wykład 4

Jerzy Stefanowski
Instytut Informatyki PP
2021

Akademia Innowacyjnych Zastosowań Technologii Cyfrowych (AI-TECH)
projekt finansowany z środków Programu Operacyjnego Polska Cyfrowa
POPC.03.02.00-00-0001/20

Fundusze Europejskie
Polska Cyfrowa

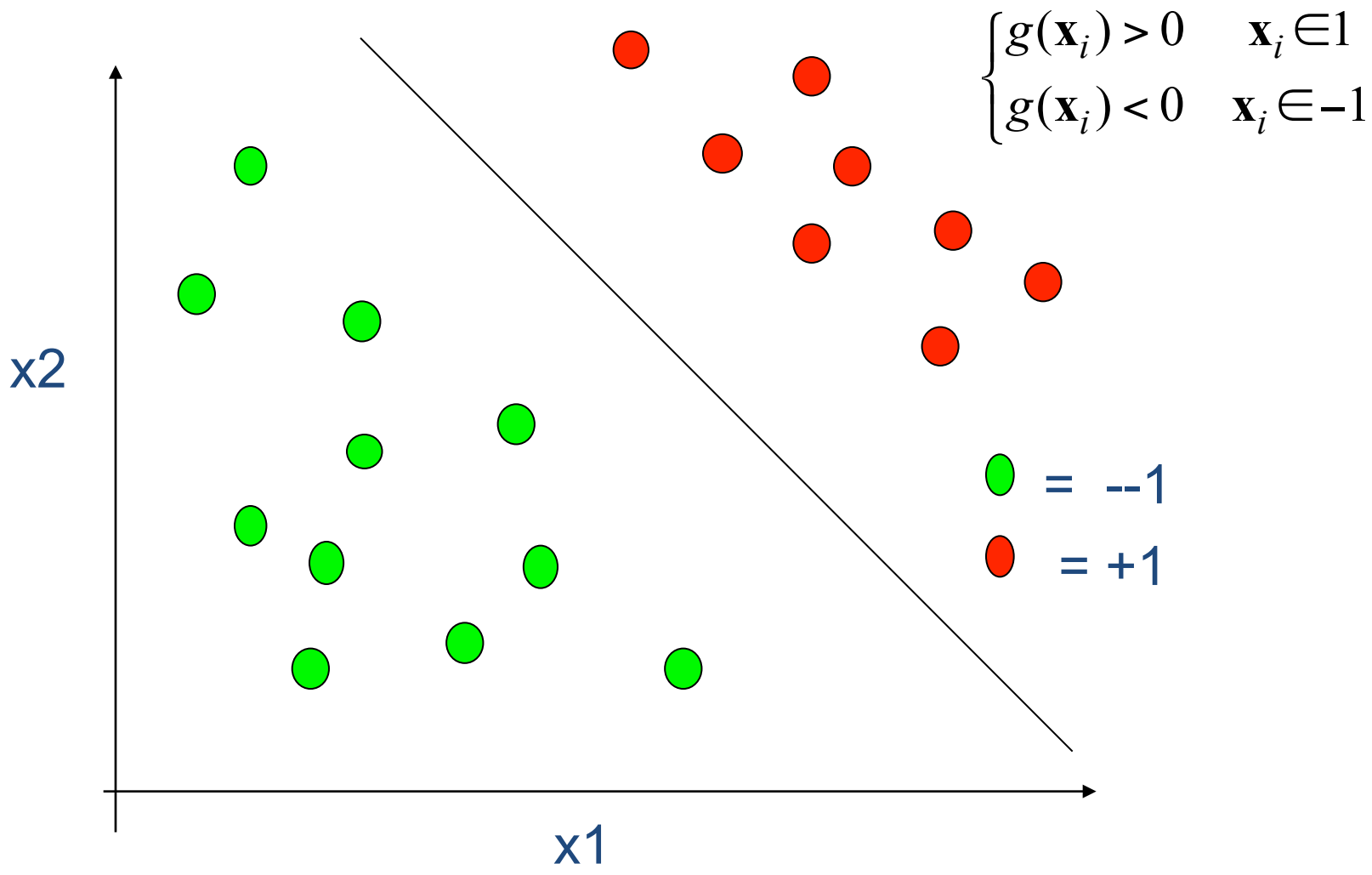
Rzeczpospolita Polska

Unia Europejska
Europejski Fundusz Rozwoju Regionalnego

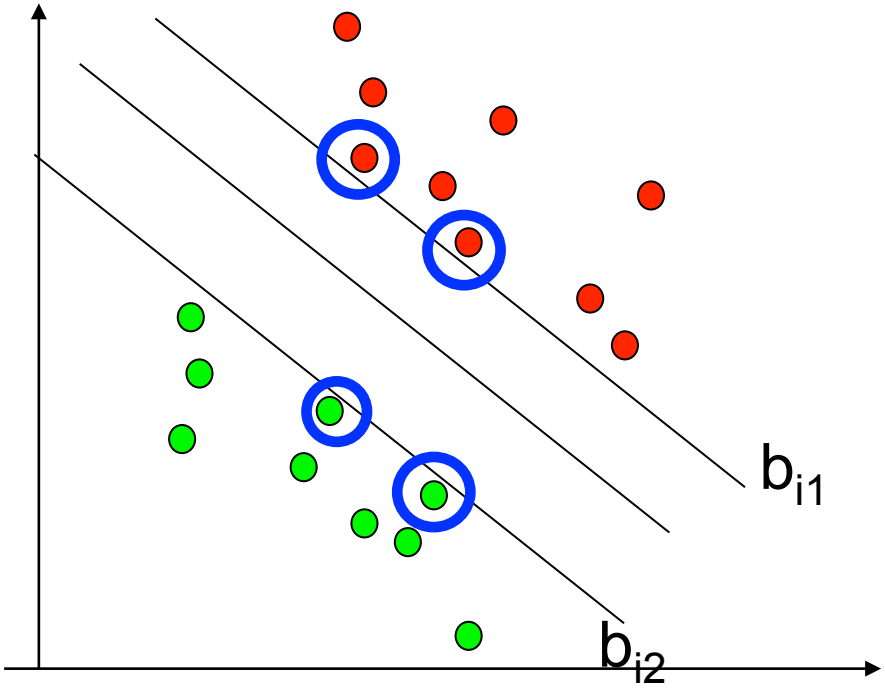
Plan wykładu

1. Przypomnienie liniowego SVM
    1. Ogólna zasada
    2. Sformułowanie zadania optymalizacji
2. Uogólnienie SVM (dane nie w pełni separowalne liniowo)
3. Funkcje jądrowe (tzw. kernel functions)
4. SVM dla danych z nieliniowymi granicami
5. Podsumowanie
6. Inne możliwości SVM oraz kernelizacji
7. Gdzie szukać więcej?

Poszukiwanie hiperpłaszczyzny separującej



Wektory nośne (wspierające)



Przykłady wspierające przesunięte hiperpłaszczyzny do granic rozkładów przykładów (wyznaczające tzw. margines klasyfikatora liniowego)

L-SVM zadanie optymalizacji

Sformułowanie mat. problemu:

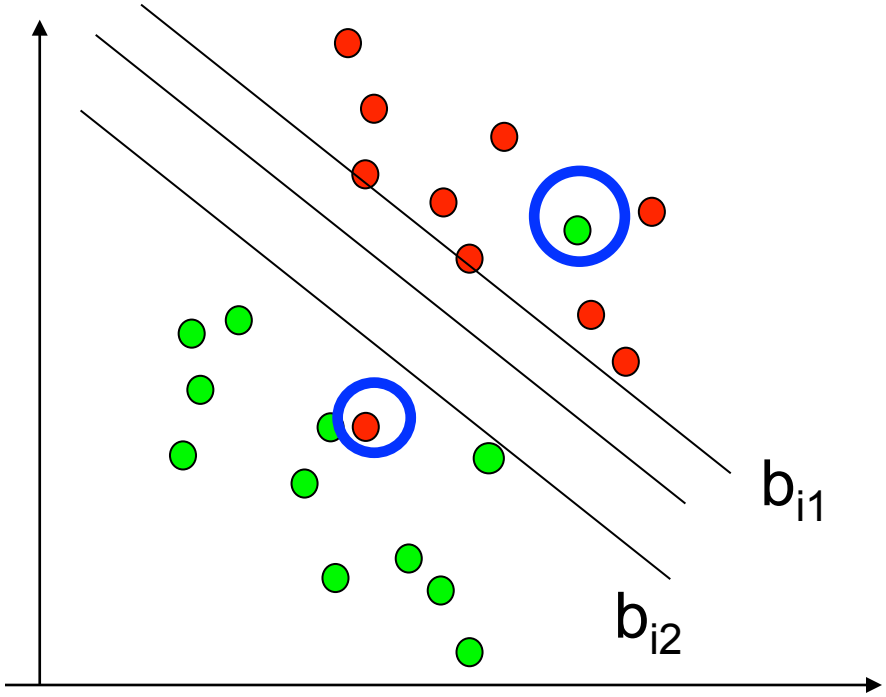
$$\min_{\mathbf{w}} = \frac{\| \mathbf{w} \|^2}{2}$$

- Przy warunkach ograniczających

$$y_i(w \cdot x_i + b) \geq 1 \quad i = 1,2,\ldots,N$$

- Jest to problem optymalizacji kwadratowej z liniowymi ogr. → uogólnione zadanie optymalizacji rozwiązywany metodą mnożników Lagrange'a (tak aby np. nie dojść do **w** → 0)
- Poprzedni wykład zawiera szczegóły − przekształcenia (zadanie dualne, …)

Dane uczące nie są liniowo separowalne (nakładanie się klas i przykłady położone po niewłaściwej stronie)



Przykłady położone po niewłaściwej stronie hiperpłaszczyzny w stosunku do rozkładów przykładów z danej klasy;

Ograniczenie $y(wx_i+b) \geq 1$ − może nie być spełnione dla niektórych przykładów

Zmienne osłabiające / dopełniające, tzw. slack variables – niezerowe dla przykładów położonych po niewłaściwej stronie granicy decyzyjnej

$$-in+1 \qquad \mathbf{w}\cdot\mathbf{x}+\mathbf{b}\geq 1-\xi$$

$$+in-1 \qquad \mathbf{w}\cdot\mathbf{x}+\mathbf{b}\leq -1+\varsigma$$

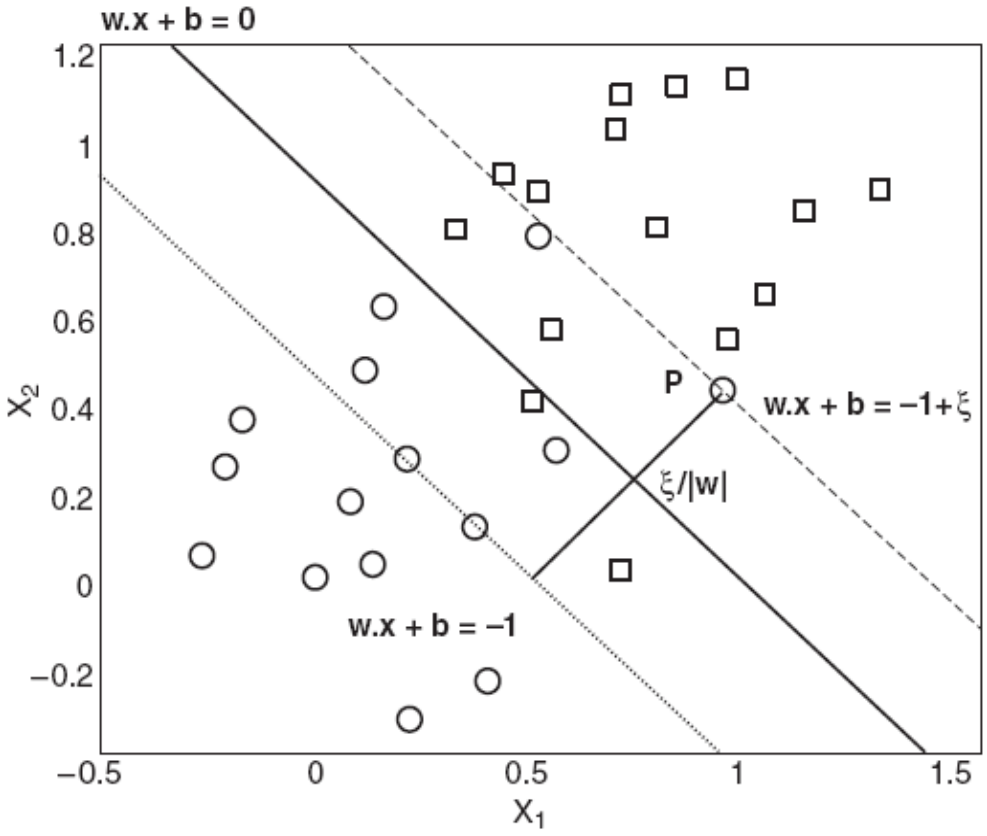


Figure 5.26. Slack variables for nonseparable data.

Zmienne osłabiające

- Zmienne $\xi_i \geq 0$ ocenią odstępstwa przykładu x_i od marginesu
  - Przykład x_i po niewłaściwej stronie granicy decyzyjnej i poza marginesem
  - Także przykład x_i po właściwej stronie granicy decyzyjnej lecz leży w obszarze marginesu zbyt blisko granicy
- Jeśli $\xi_i = 0$, to nie występują trudności z położeniem przykładu x_i
- Definiuje się tzw. soft error jako sumę zmiennych ξ_i = patrz dalsze sformułowanie problemu optymalizacyjnego z dodatkowych elementem funkcji „kary”

Zmienne osłabiające - interpretacja

- Zmienne $\xi_i \geq 0$ dobiera się dla każdego przykładu uczącego. Jej wartość zmniejsza margines separacji (rodzaj „zwisu” punktu poza hiperpłaszczyzną nośną)
- Jeżeli $0 \leq \xi_i \leq 1$, to punkt danych $(\mathbf{x}_i, d_i)$ leży wewnątrz strefy separacji, ale po właściwej stronie
- Jeżeli $\xi_i > 1$, punkt po niewłaściwej stronie hiperpłaszczyzny i wystąpi błąd klasyfikacji
- Modyfikacja wymagań dla wektorów nośnych

$$b_{i1} \qquad \mathbf{w} \cdot \mathbf{x} + \mathbf{b} = 1 - \xi$$

$$b_{i2} \qquad \mathbf{w} \cdot \mathbf{x} + \mathbf{b} = -1 + \varsigma$$

Rola zmiennych osłabiających w optymalizacji

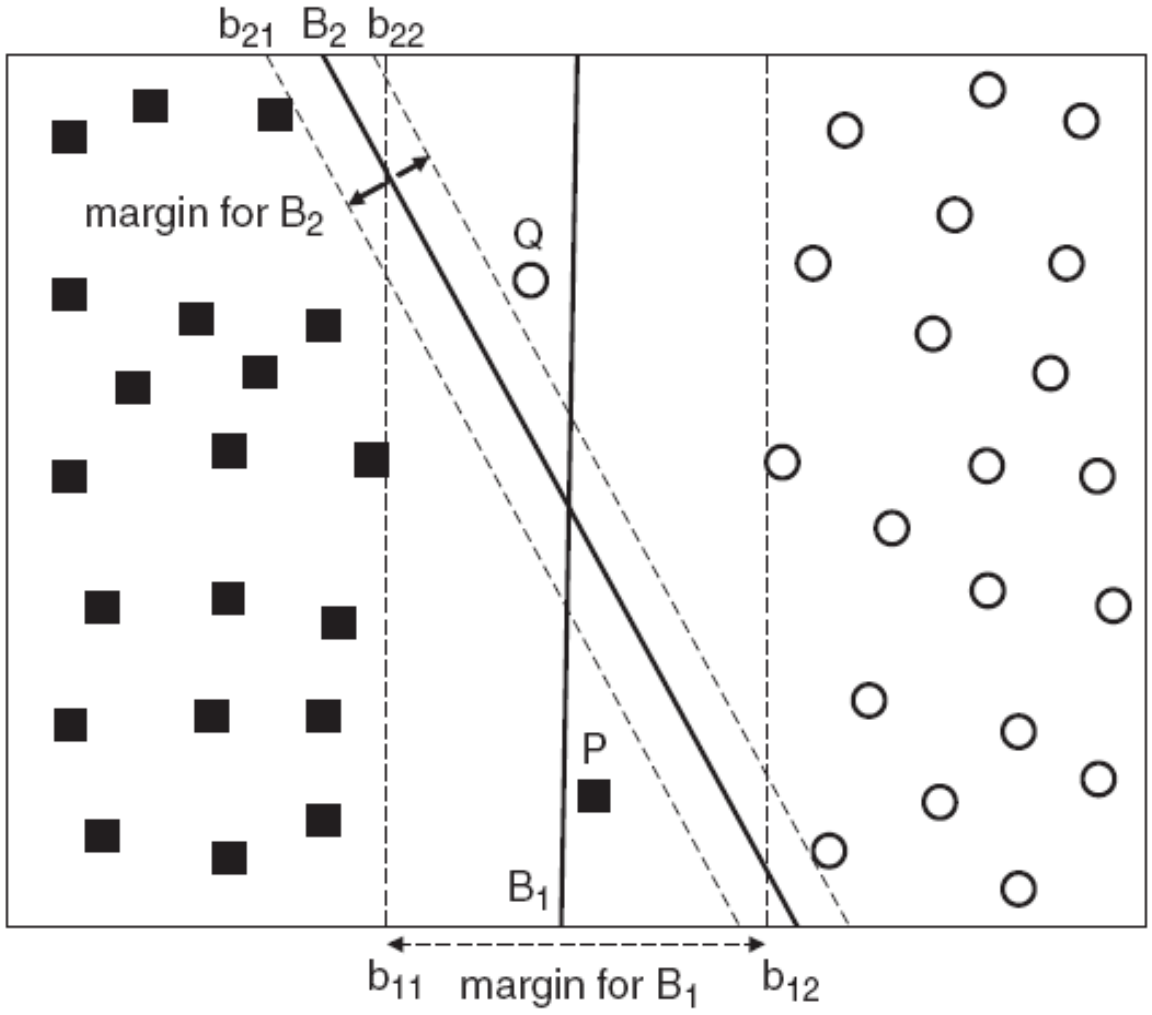


Figure 5.25. Decision boundary of SVM for the nonseparable case.

SVM z dodatkowymi zmiennymi

- Jak przedefiniować sformułowanie? Z dodatkowymi zmiennymi osłabiającym oraz kosztem błędu na danych uczących
  - Mimimalizuj funkcję:

$$L(w) = \frac{\| \vec{w} \|^2}{2} + C\left(\sum_{i=1}^{N} \xi_i^k \right)$$

  - z ograniczeniami:
  - $$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x}_i + b \geq 1 - \xi_i \\ -1 & \text{if } \vec{w} \bullet \vec{x}_i + b \leq -1 + \xi_i \end{cases}$$
  - Drugi czynnik odpowiada za ew. błędy klasyfikowania (górne oszacowanie tych błędów)
  - Parametr ***C*** ocena straty związanej z każdym błędnie klasyfikowanym punktem dla które ξ>0
  - Przetarg „szeroki margines” to dużo błędów i odwrotnie

Rozwiązanie problemu - przekształcenia

Programowanie kwadratowe (QP) : trudności rozwiązania - przeformułuj problem

Ponownie dojdziemy do dualnego problemu:

$$\textit{Max}: W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \left(\mathbf{x}_i \bullet \mathbf{x}_j \right)$$

przy ograniczeniach:

(1) $0 \le \alpha_i \le C, \ \forall i$

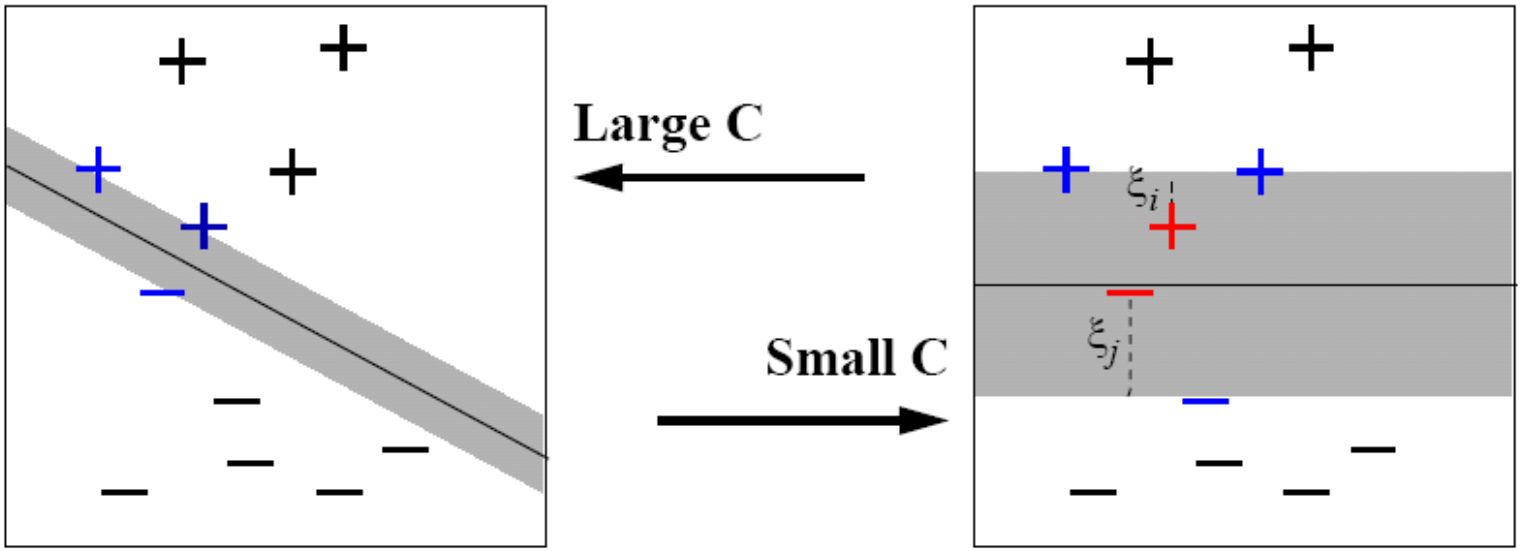
(2) $\sum_{i=1}^{m} \alpha_i y_i = 0$

Controlling Soft-Margin Separation

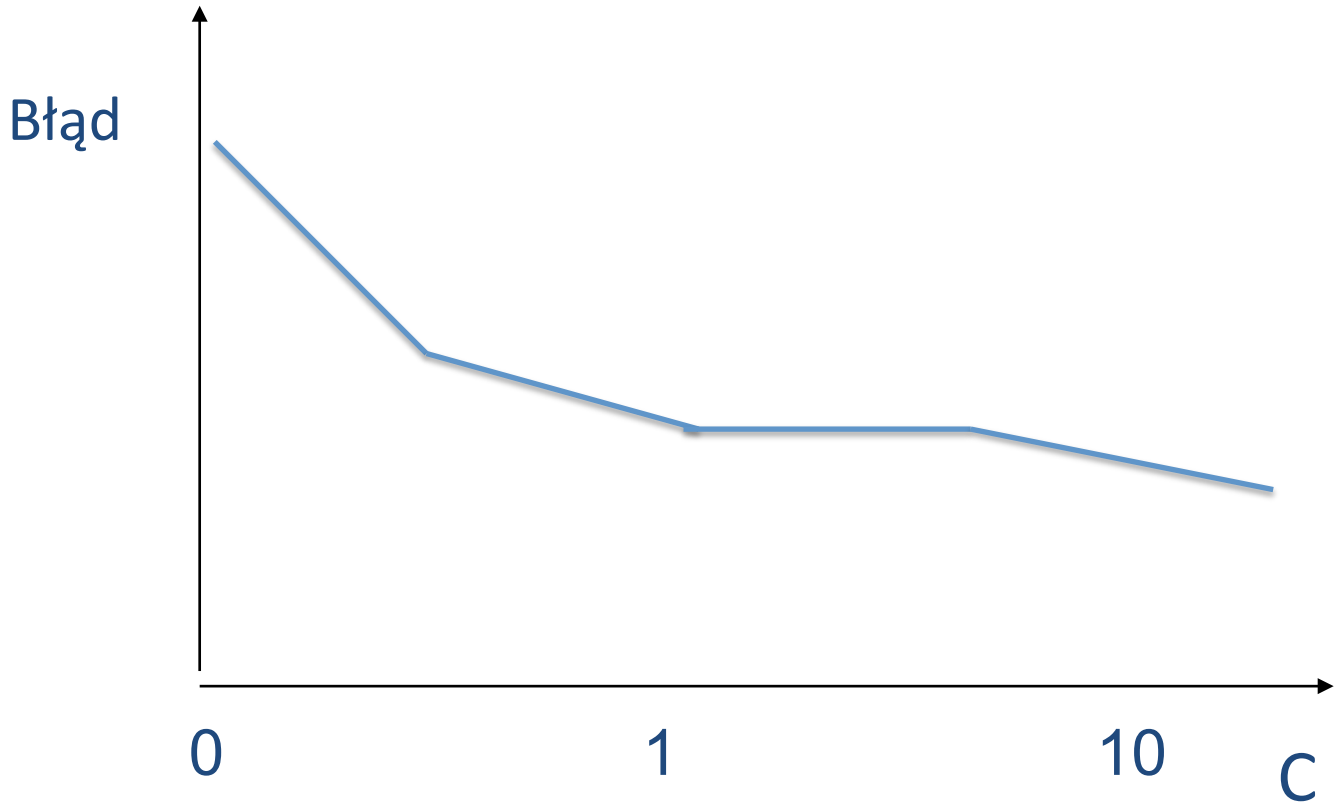
Soft Margin: minimize $P(\vec{w}, b, \vec{\xi}) = \frac{1}{2}\vec{w} \cdot \vec{w} + C \sum_{i=1}^{n} \xi_i$

s. t. $y_i[\vec{w} \cdot \vec{x}_i + b] \geq 1 - \xi_i$ *and* $\xi_i \geq 0$

- $\sum \xi_i$ is an upper bound on the number of training errors.
- C is a parameter that controls trade-off between margin and error.

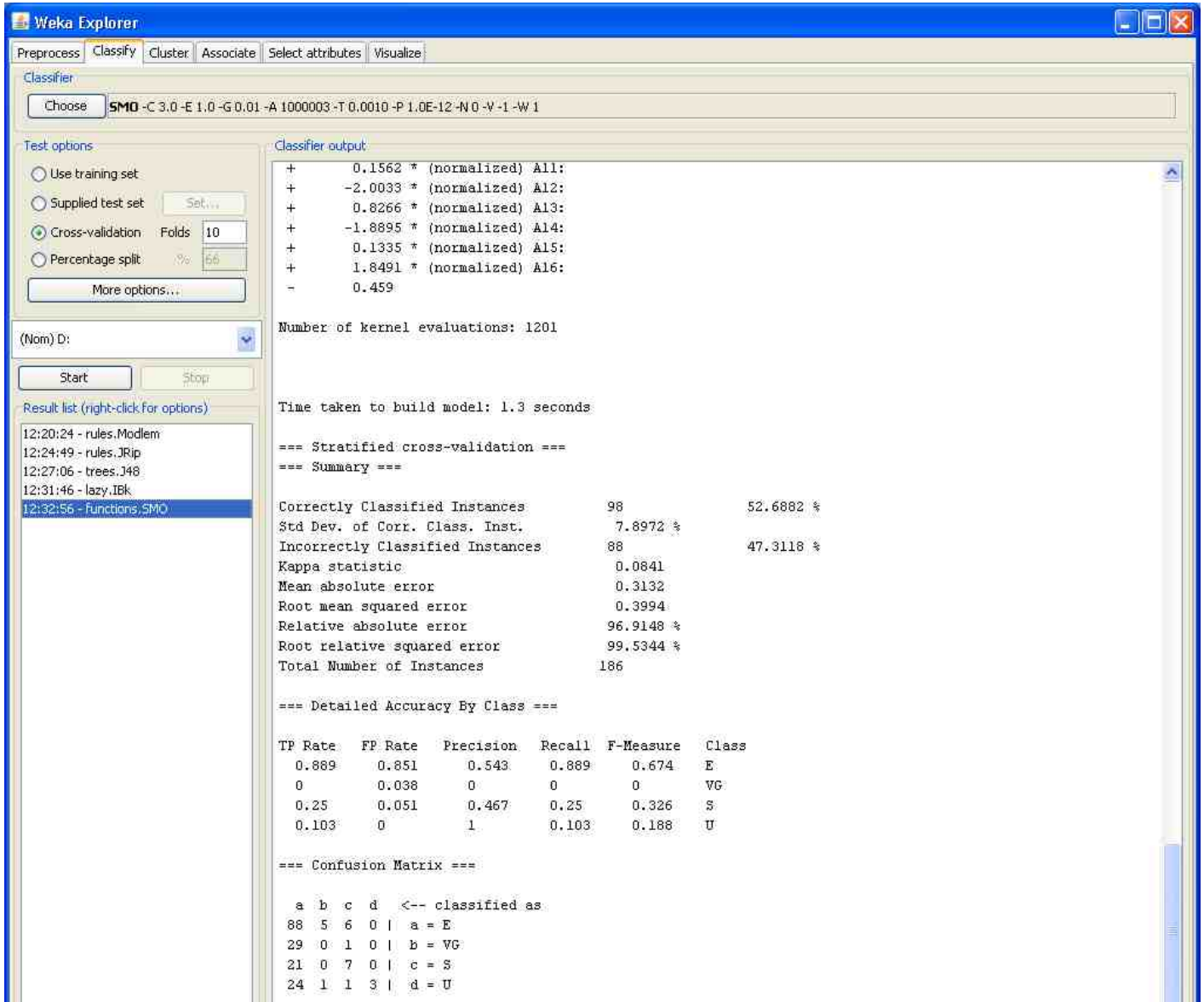


Dostrajanie par. C



Przykład danych Reuters;
Często domyślnie około 1, lecz dla trudnych danych warto podwyższyć

Dostrajanie par. C



Gdzie jesteśmy w wykładzie:

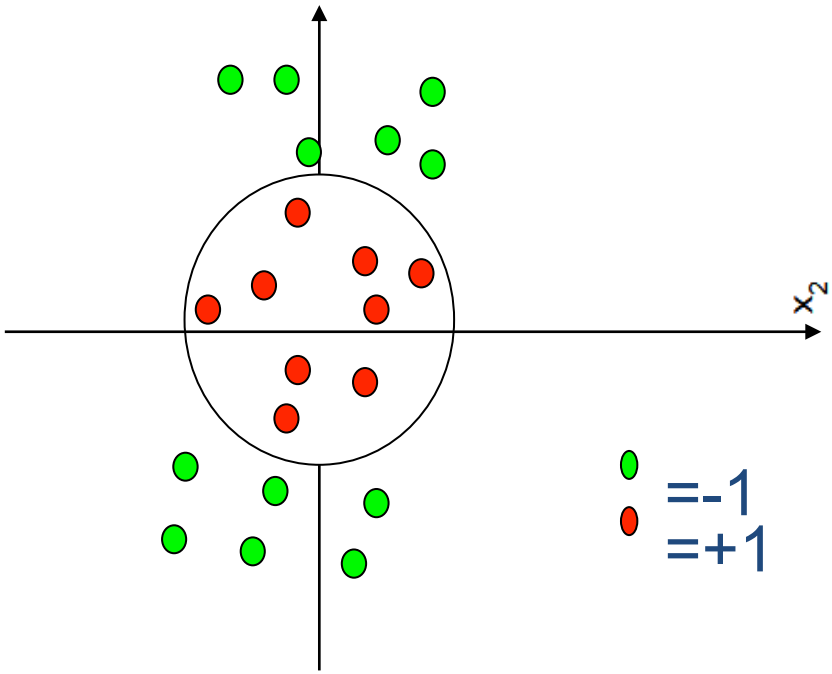
- Podstawy matematyczne SVM
- Przekształcenia do obliczalnych wersji zadania programowania matematycznego
- Rozszerzenie na tzw. overlapping nakładania się klas

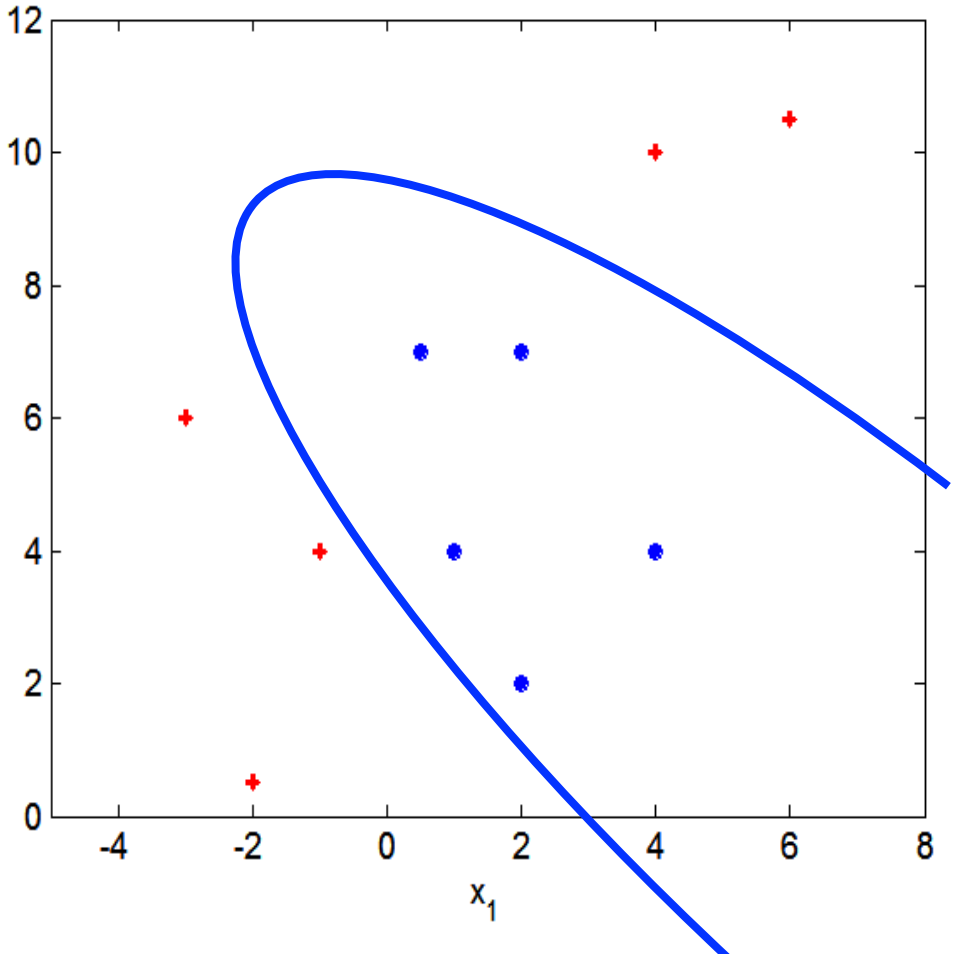
Lecz rzeczywiste problemy mają złożone nieliniowe granice decyzyjne!

Jak przejść z L-SVM na N-SVM?

Nieliniowy SVM

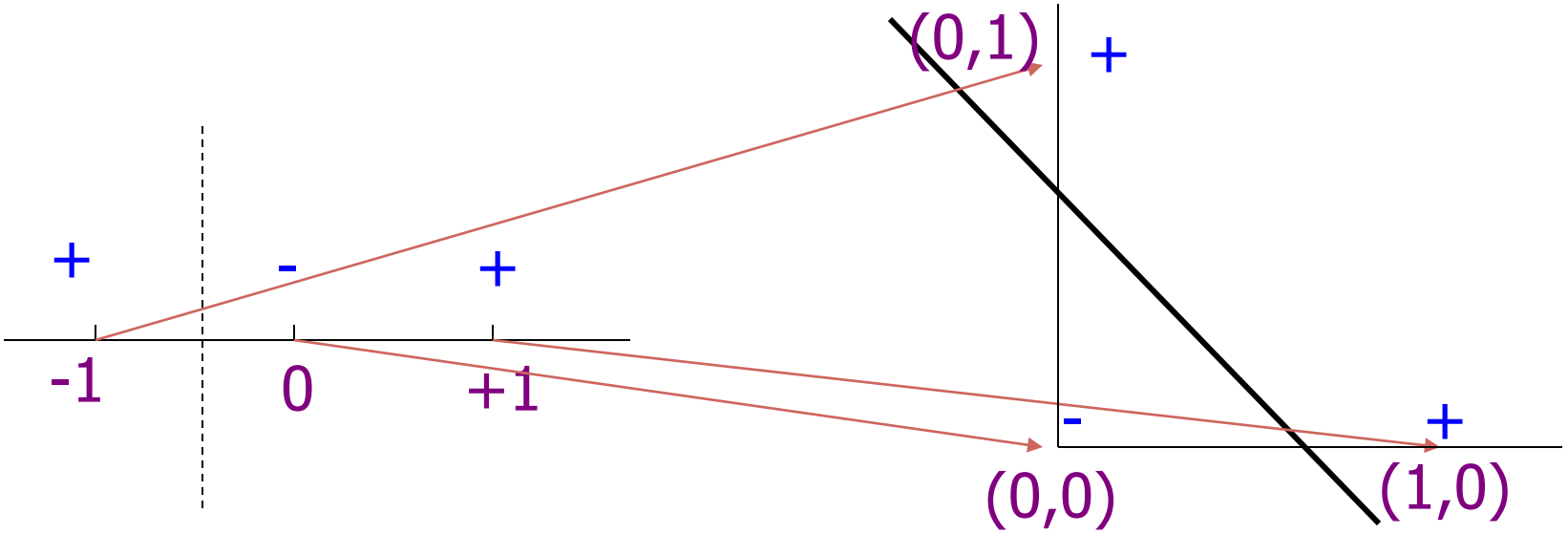
Kiedy klasy są nieliniowo separowalne oraz kształt granicy decyzyjnej jest dość złożony?





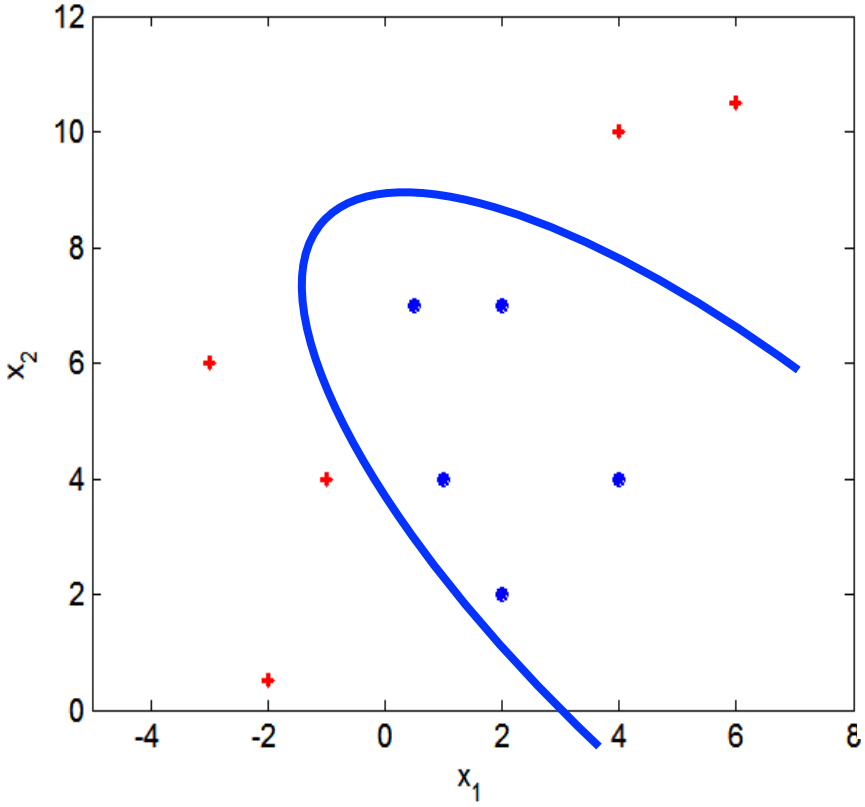
N-SVM – Transformacje

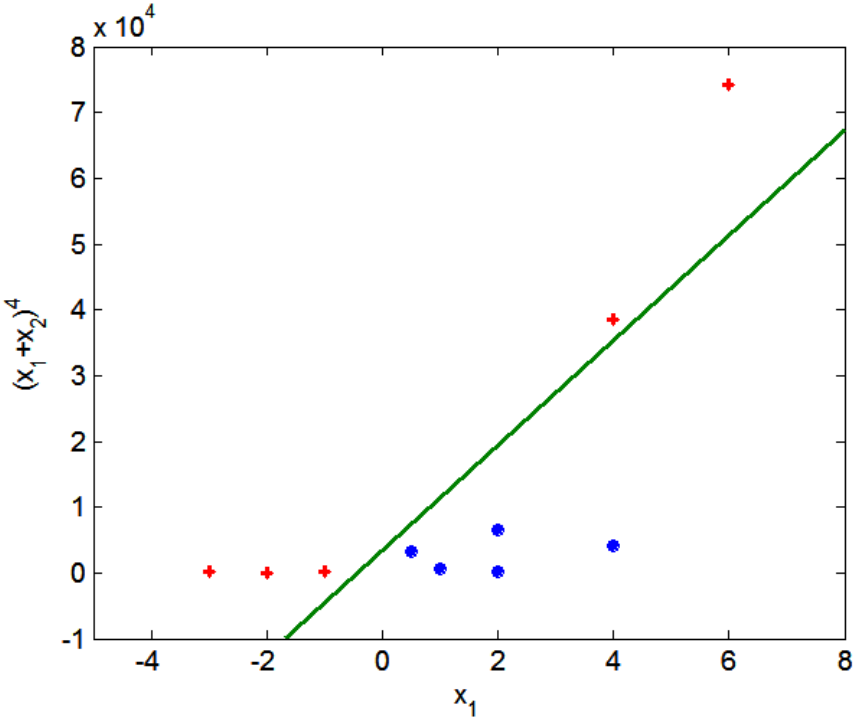
- Przykład transformacji 1D→ 2D
- Projekcja danych oryginalnych $x \in R^p$ w nową m>p wielowymiarową przestrzeń, w której z dużym prawdopodobieństwem będą separowalne liniowo (Twierdzenia matem. np. Covera)
- Przykład przekształcenia wielomianowe wyższego stopnia gdzie do zmiennych x dołącza się ich p-te potęgi oraz iloczyny mieszane zmiennych.
- W ogólności trzeba użyć większej liczbie wyżej wymiarowych przestrzeni przekształconych zmiennych



Nonlinear Support Vector Machines

Przykład transformacji wielomianowej x2 –> $(x1+x2)^4$

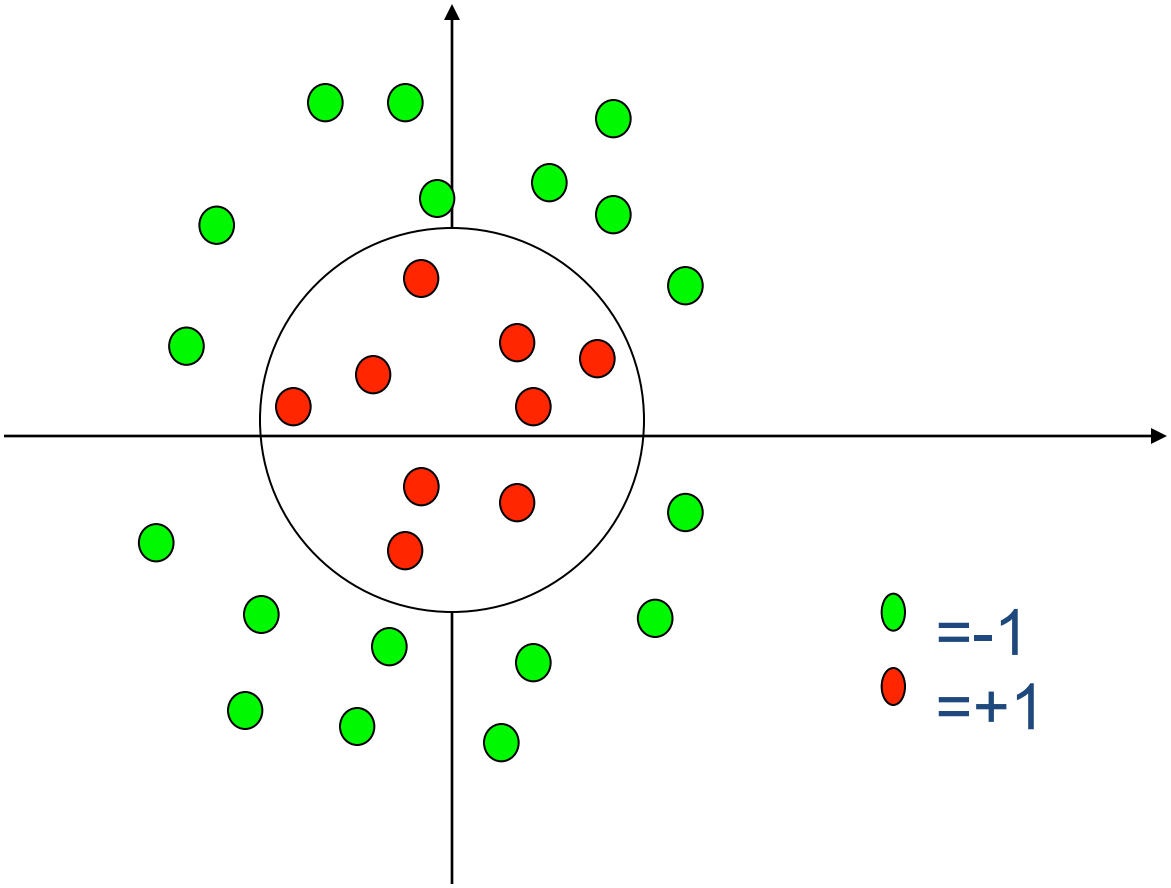




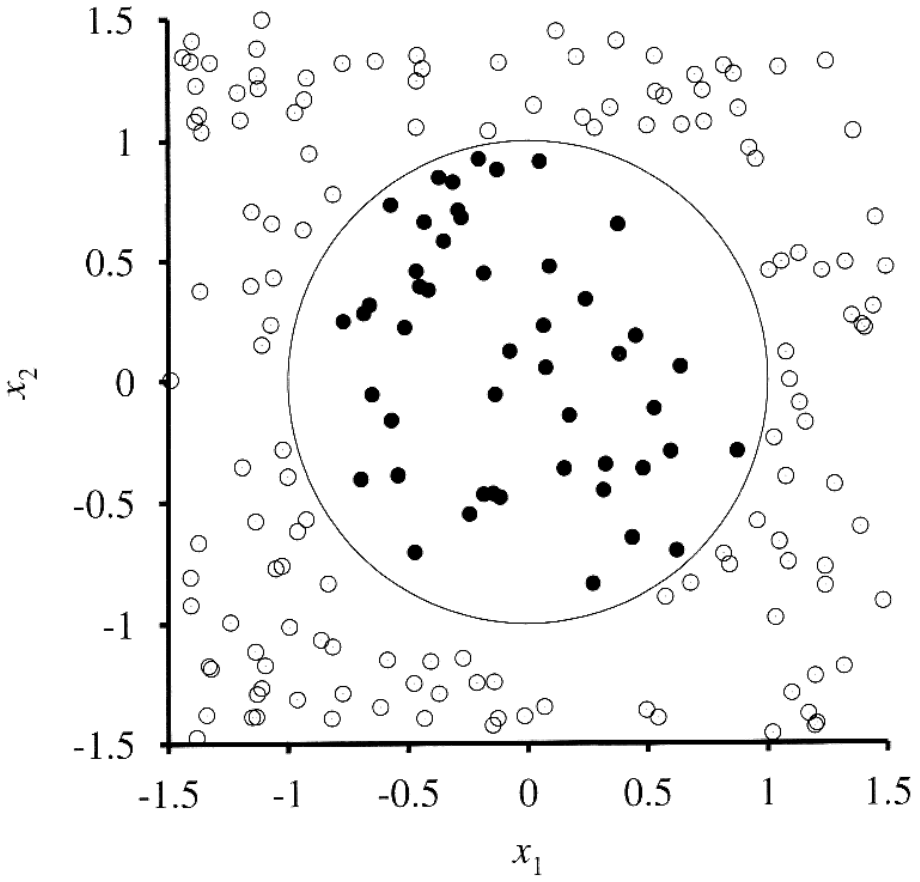
Trochę inspiracji statystyki mat.

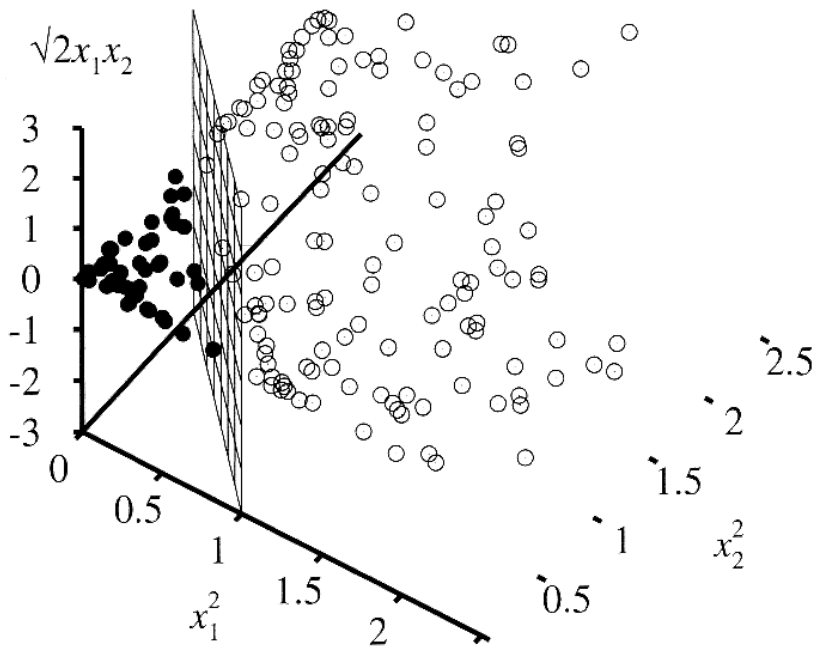
- W zagadnieniach regresyjnych i klasyfikacyjnych, dawno zauważono że można otrzymać bardziej efektywne rozwiązanie, jeżeli oprócz (zamiast) oryginalnego wektora danych $x \in R^d$ rozpatrzy się rozszerzony wektor $z=\phi(x)$ zawierający m składowych, gdzie $m \geq d$.
- Przykładem są przekształcenia wielomianowe
  - Do oryginalnych zmiennych dodajemy ich potęgi oraz iloczyny mieszane zmiennych
  - Inne b. złożone przekształcenia
  - Własności matematyczne − spójrz do literatury
- Przykład rozkładu kołowego

Przykład rozkładu kołowego klasy otoczonego przykładami z innej klasy



Przykład transformacji wielomianowej





Koło - przykład transformacji wielomianowej

- Oryginalna funkcja celu

$$y(x_1, x_2) = \begin{cases} 1 & \sqrt{(x_1 - 0.5)^2 + (x_2 - 0.5)^2} > 0.2 \\ -1 & otherwise \end{cases}$$

$$\Phi : (x_1, x_2) \rightarrow (x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, 1)$$

- **Transformacja**
- Poszukiwanie parametrów równania liniowego po transformacji

$$w_4 x_1^2 + w_3 x_2^2 + w_2 \sqrt{2} x_1 + w_1 \sqrt{2} x_2 + w_0 = 0$$

- Rozwiązanie

$$x_1^2 - x_1 + x_2^2 - x_2 = -0.46$$

Model nieliniowy SVM

- Funkcja decyzyjna po przekształceniu $g(\mathbf{x}) = \mathbf{w}\varphi(\mathbf{x}) + b$
- Zasady klasyfikacji

$$+1 \quad g(\mathbf{x}) > 0$$
$$-1 \quad g(\mathbf{x}) < 0$$

- Sformułowanie problemu nieliniowego SVM

$$\min_{\mathbf{w}} = \frac{\|\mathbf{w}\|^2}{2} \qquad y_i(w \cdot \Phi(x_i) + b) \geq 1 \quad i = 1,2,\ldots,N$$

- Podobnie jak poprzednio optymalizujemy funkcje z mnożnikami Lagrange'a

$$\sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \left(\Phi(\mathrm{x}_i)\Phi(\mathrm{x}_j)\right)$$

- Funkcja klasyfikująca

$$f(x) = sign(\sum_{i=1}^{N} \alpha_i y_i \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}) + b)$$

Wyzwanie obliczeniowe

- Oblicz $\Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x}_j)$
- **Problem:** Trudne obliczeniowo do wykonania!
- Wiele parametrów do oszacowania - wielomian stopnia *p* dla *N* atrybutów w oryginalnej przestrzeni prowadzi do $O(N^p)$ atrybutów w nowej rozszerzonej *F* przestrzeni cech
- Skorzystaj z dot product (iloczynu skalarnego) na wektorach wejściowych jako miary podobieństwa wektorów
- Iloczyn $\Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x}_j)$ może być odniesiony do podobieństwa wektorów $\mathbf{x}_i\cdot\mathbf{x}_j$ w **transformowanej** rozszerzonej przestrzeni
- Idea ”kerneli” (funkcji jądrowych)
  - Proste funkcje *K* dwóch argumentów wektorowych pozwalają obliczyć wartość iloczynu skalarnego w rozszerzonej przestrzeni

Co to są funkcje jądrowe (ang. Kernel function)

- Wywodzą się z badań liniowych przestrzeni wektorowych, przestrzeni Hilberta, Banacha
- Intuicyjnie są to stosunkowo proste symetryczne $K(\mathbf{x}_i,\mathbf{x}_j)$ zależne od odległości między $\mathbf{x}_i$ i $\mathbf{x}_j$ które spełniają pewne wymagania matem.

$$K(u) \geq 0, \int K(u)du = 1, \sigma_K^2 = \int uK(u)du > 0$$

Wniosek z twierdzenia Mercera

- Jeśli *K*(**x**,**y**) jest funkcją jądrową dla każdego **x,y** in *X* (R), to można określić przekształcenie $\Phi: X \text{-} \rightarrow F$ (przestrzeń przekształconych zmiennych) takie, że

$$K(\mathbf{x},\mathbf{y}) = \Phi(\boldsymbol{x})\ \Phi(\boldsymbol{y})$$

- Własność podstawą tzw. triku kernelowego (*kernel trick*)
- Cytat z literatury
  - „... map the data into some other scalar product space (feature space) *F* by means of a nonlinear mapping like the above, and perform the linear algorithm (like decision boundary for 2 classes) in the feature space *F*. In many cases the mapping Φ cannot be explicitly computed, due to the high-dimensionality of *F*. But this is not an obstacle, when the decision function requires the evaluation of scalar products $\Phi(x)\cdot\Phi(y)$, and not the pattern $\Phi(x)$ in explicit form.”[Camastra]
- **Każdy iloczyn funkcji – można zastąpić nieliniową funkcją jądrową na prostszym iloczynie wektorów (przykładów)**

SVM: tzw. trik kernelowy

Przykład prostego przekształcenia wielomianowego

- The kernel trick:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)^2 = \left(x_{i1}^2, \sqrt{2}x_{i1}x_{i2}, x_{i2}^2\right) \cdot \left(x_{j1}^2, \sqrt{2}x_{j1}x_{j2}, x_{j2}^2\right)$$
$$= \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$$

- zadanie optymalizacyjne

$$\sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i \cdot \mathbf{x}_j)$$

Nie musimy znać funkcji Φ, wystarczy znać jądro (kernel) i można "pracować" w nowej przestrzeni.

Typowo stosowane f. jądrowe w SVM

Normalne (Gaussowskie)	$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\{-\frac{(\mathbf{x}_i - \mathbf{x}_j)}{2\sigma^2}\}$
Wielomianowe	$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j + d)^p$
sigmoidalne	$K(\mathbf{x}_i, \mathbf{x}_j) = tgh(\kappa \mathbf{x}_i \cdot \mathbf{x}_j - \delta)$

Najpopularniejsze funkcje jądrowe − patrz też implementacje z b. złożonymi funkcjami

Przykład obliczeń

Najprostsza funkcja wielomianowa: $K\left(\mathbf{X},\mathbf{Y}\right)=\left(1+\mathbf{X}\cdot\mathbf{Y}\right)^{d}$

Zastosujmy dla przestrzeni 2D

$$K\left(\mathbf{X},\mathbf{Y}\right)=\left(1+X_1Y_1+X_2Y_2\right)^2$$

$$=1+2X_1Y_1+2X_2Y_2+\left(X_1Y_1\right)^2+\left(X_2Y_2\right)^2+2X_1Y_1X_2Y_2$$

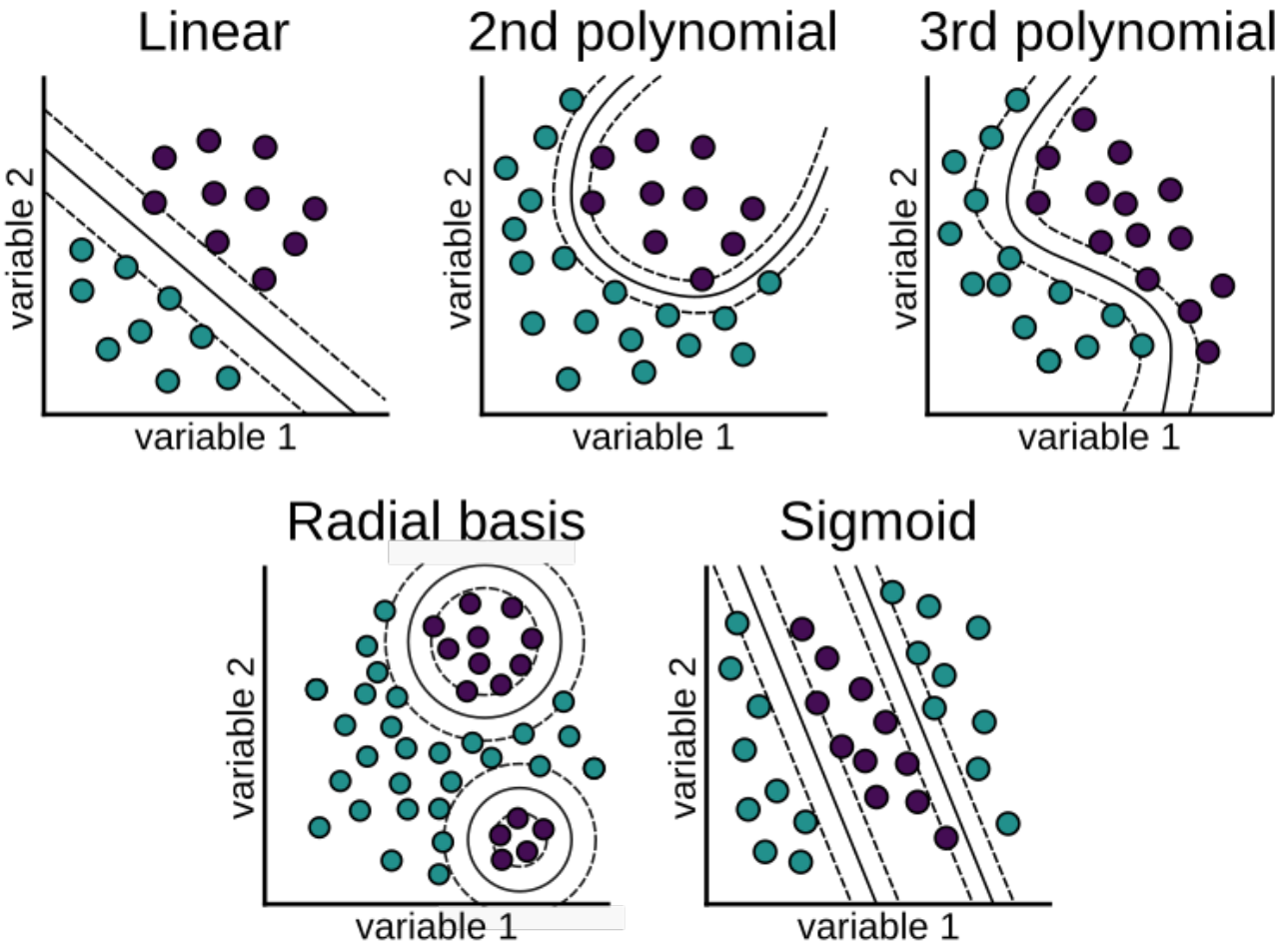
W ten sposób przenosimy się do przestrzeni 5 D

$$\mathbf{X}=\left(X_1,X_2\right)\Rightarrow\Phi\left(\mathbf{X}\right)=\left(1,\sqrt{2}X_1,\sqrt{2}X_2,X_1^2,X_2^2,\sqrt{2}X_1X_2\right)$$

Hiperpłaszczyzna w 5D – odnaleziona liniowym SVM – odpowiada funkcji kwadratowej w 2D

Oczywiście wybór funkcji jądrowej wpływa na wynik przekształcenia

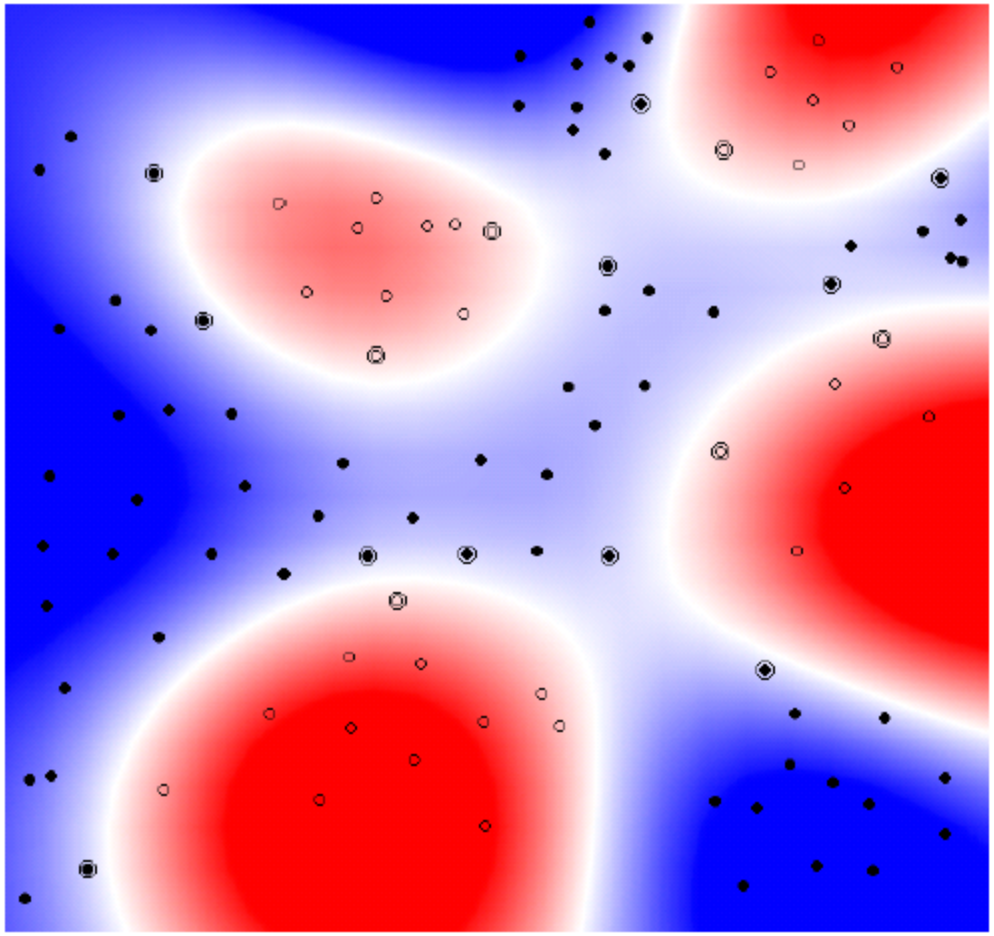
Ilustracyjne rady



Example: SVM with RBF-Kernel

Kernel: $K(\vec{x}_i,\vec{x}_j) = \exp(-|\vec{x}_i - \vec{x}_j|^2/\sigma^2)$

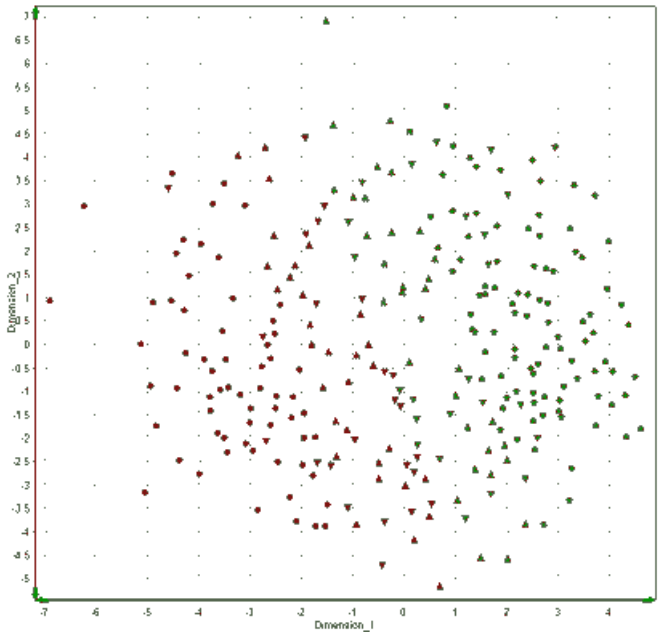
plot by Bell SVM applet

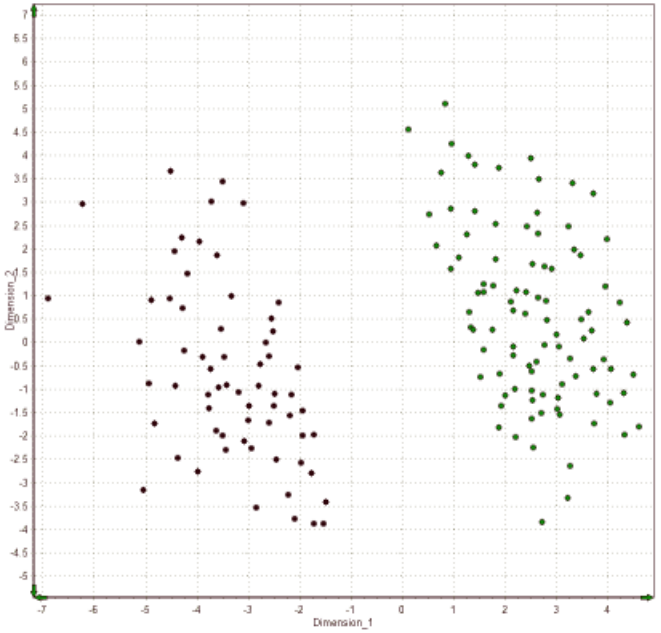


Przykład 2: Cleveland heart data

Left: 2D MDS features, linear SVM, C=1, acc. 81.9%

Right: support vectors removed, margin is clear.





Gaussian kernel, C=10000, 10xCV, 100% train, 79.3 ± 7.8 test
Gaussian kernel, C=1, 10xCV, 93.8% train, 82.6 ± 8.0 test
Auto C=32 and Gaussian dispersion 0.004: about 84.4 ± 5.1 on test

Funkcja decyzyjna

- Wykorzystanie funkcji jądrowych

$$f(\mathbf{x}) = \begin{array}{c} sign\left(\sum_{i=1}^{N} \alpha_i y_i \Phi(\mathbf{x}_i)\Phi(\mathbf{x}) + b\right) \\ sign\left(\sum_{i=1}^{N} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right) \end{array}$$

- Model klasyfikacji binarnej rozszerza się na zagadnienie wieloklasowe K > 2
  - Specyficzne konstrukcje złożone (jak w tzw. zespołach):
    - one-versus-all
    - Pairwise classification (one against one)

Inne definicje funkcji jądrowych

- Tworzenie dziedzinowych specjalizowanych funkcji jądrowych
  - K(x,y) przyjmuje wyższe wartości, jeśli x oraz y są podobne do siebie.
- Rozważmy, np. klasyfikacje tekstów: dwa dokumenty D1 oraz D2 – K skonstruowane z wykorzystaniem liczby wspólnych słów
- Inne: tzw. Tree kernels, graphs kernels. … (Scholkopf i inni)

Liczne zastosowania

Można się zapoznać z listą:
http://www.clopinet.com/isabelle/Projects/SVM/applist.html

Od końca poprzedniego wieku niezwykle popularny:

- Rozpoznawanie ręcznego pisma – historyczne prace Vapnik i współpracownicy
- Rozpoznawanie obiektów na zdjęciach
- Intrusion Detection Systems (IDSs)
- Klasyfikacja obrazów
- Zastosowania medyczne (diagnostyka, …)
- Fizyka wysokich cząstek
- Bioinformatyka: analiza mikromacierzy, własności białek
- Wyszukiwanie informacji
- Przetwarzanie dokumentów tekstowych

Trochę historii

- Wczesne lata sześćdziesiąte – została opracowana metoda “support vectors” w celu konstruowania hiperpłaszczyzn do rozpoznawania obrazu (Vapnik i Lerner 1963, Vapnik i Czervonenkis 1964 / Inst. Akademii Nauk ZSRR) – liniowa SVM.
- Początek lat 1990-siątych: uogólnienie metody pozwalające na konstruowanie nieliniowych funkcji separujących (Boser 1992, Cortes i Vapnik 1995).
- 1995: dalsze rozszerzenie pozwalające otrzymać estymację funkcji ciągłej na wyjściu – regresja (Vapnik 1995).
- Później grupowanie z SVM (Support Vector Clustering)
- Ciekawy wykład V.Vapnika (Complete Statistical Theory of Learning):

  https://www.youtube.com/watch?v=Ow25mjFjSmg

Kilka zagadn. efektywnego stosowania SVM

- Normalizuj sygnały wejściowe
- Dobrze wybierz wartość C
- Wybór funkcji jądrowej − ważna decyzja + parametryzacja wybranej funkcji
- Uogólnienia dla problemów wieloklasowych
- … co jeszcze?
  - Na ile są skuteczne SVM w analizie danych niezbalansowanych?
  - Tzw. częściowo-etykietowane SVM
- Spójrz na mój inny przedmiot Projekt Eksploracji Danych (sem. 10)

http://www.cs.put.poznan.pl/jstefanowski/PSE.html

Parę uwag podsumowujących

- **Dane odwzorowane (przy pomocy funkcji jądrowych) w nową przestrzeń cech – silna przewaga nad innymi metodami**
- **Wykorzystanie modelu optymalizującego − jedno rozwiązanie**
- W nowej przestrzeni dane powinny być liniowo separowalne
- W porównaniu do innych podejść wielowymiarowość przekształcenia jest „rozwiązana” przez trick kernelowy
- Pośrednio ogranicza się niebezpieczeństwo przeuczenia − ale nadal może wystąpić
- Teoretycznie poszukują minimum globalnego a nie lokalnego (jak podejścia heurystyczne – np. MLP, i inne ANN)
- Ograniczenia
  - Dobór parametrów − trudne, także wybór funkcji kernela
  - Odpowiednik podejście „black box”

Mocne strony SVM

Stopień skomplikowania/pojemność jest niezależna od liczby wymiarów.

Bardzo dobra podbudowa statystyczno-teoretyczna

Znajdowanie minimum glonalnego. Minimalizujemy funkcję kwadratową co gwarantuje zawsze znalezienie minimum.

SVM generuje “prawie optymalny” klasyfikator

Dobre uogólnianie dzięki wielowymiarowej “feature space”.

Najważniejsze: poprzez użycie odpowiedniej funkcji jądra SVM bardzo duża skuteczność w praktyce

Słabe strony SVM

Powolny trening – minimalizacja funkcji, szczególnie dokuczliwy przy dużej ilości danych użytych do treningu.

Rozwiązania też są skomplikowane (czasami >60% wektorów użytych do nauki staje się wektorami wspierającymi), szczególnie dla dużych ilości danych.

> *Przykład (Haykin): poprawa o 1.5% ponad wynik osiągnięty przez MLP. Ale MLP używał 2 ukrytych węzłów, a SVM 285 wektorów.*

Trudno dodać własną wiedzę (prior / background knowledge) !

SVM oprogramowanie

- SVM Website
  - http://www.kernel-machines.org/
- Popularne
  - **LIBSVM**: efektywna implementacja, także dla problemów wieloklasowych, one-class SVM, dostępna dla java, python, etc.
  - SVM-light: prostsza niż LIBSVM, na ogół dla binarnych problemów
  - SVM-torch: inna napisana w C.
- scikit learn – implementacja oparta na libsvm
- Ponadto – liczne inne implementacje, Weka, R

Scikit-learn

- Proszę sprawdzić dokumentacje klasy sklearn.svm.SVC
- Główne parametry
  - Współczynnik C
  - Wybór funkcji jądrowej (‘linear’, ‘poly’, ‘rbf’, ‘sigmoid’, ‘precomputed’) − domyślnie rbf
  - Parametry funkcji (np. gamma w rbf, degree do funkcji wielomianowej)

Odnośniki literaturowe

Warto sprawdzić moje stronę:

Tradycyjnie strona materiały dodatkowe:
http://www.cs.put.poznan.pl/jstefanowski/ml/dodatki.html

Oprócz linków na stronie www dodatki, warto czytać książki:

- T.Hastie, R.Tibshirani, J.Friedman: The Elements of Statistical Learning. Springer → poszukaj wersji elektronicznej pdf
- J.Koronacki, J.Ćwik: Statystyczne systemy uczące się (rozdz. 6)
- M.Krzyśko, W.Wołyński, T.Górecki, M.Skorzybut: Systemy uczące się.
- S.Ossowski: Sieci neuronowe w przetwarzaniu informacji

- Poszukujcie dalej w Internecie − sprawdźcie sami i referujcie

Inne rozszerzenia SVM

- Specjalne wieloklasowe SVM - poza O-a-A lub Pairwise coupling – także inne sformułowanie problemu optymalizacyjnego
- Sformułowanie regresyjnego SVM
- One-class SVM (estymacja obszaru o wysokiej gęstości przykładów i dyskryminacja od obszarów o niskiej gęstości)
  - Przydatna dla wykrywania obserwacji nietypowych (ang. outlier detection) oraz uczenia się z rzadkich, danych niezbalansowanych.
- Specjalne wersje tzw. transductive support-vector machines – dla uczenia częściowo nadzorowanego

One-Class Kernel Machines

Naucz się sfery z centrum ***a*** oraz promieniem *R*

$$\min R^2 + C\sum_t \xi^t$$

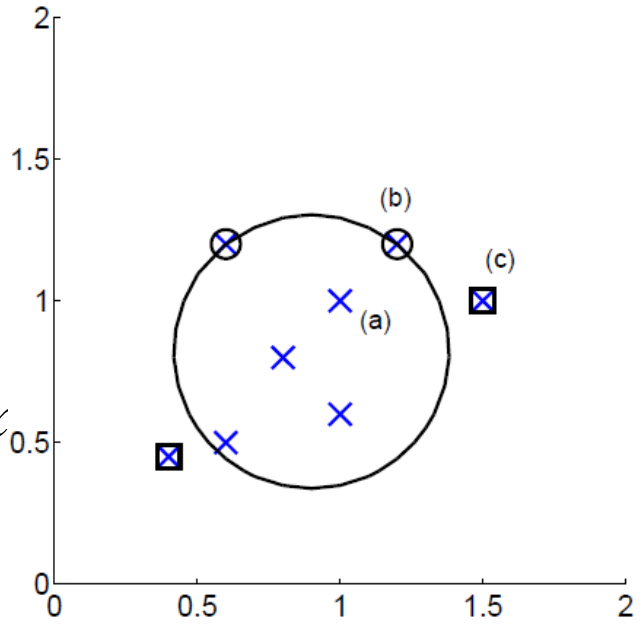
organiczenia

$$\left\|\mathbf{x}^t - a\right\| \le R^2 + \xi^t, \xi^t \ge 0$$

$$L_d = \sum_t \alpha^t \left(x^t\right)^T x^s - \sum_{t=1}^{N}\sum_s \alpha^t \alpha^s r^t r^s \left(x^t\right)^T x$$

organiczenia

$$0 \le \alpha^t \le C, \sum_t \alpha^t = 1$$



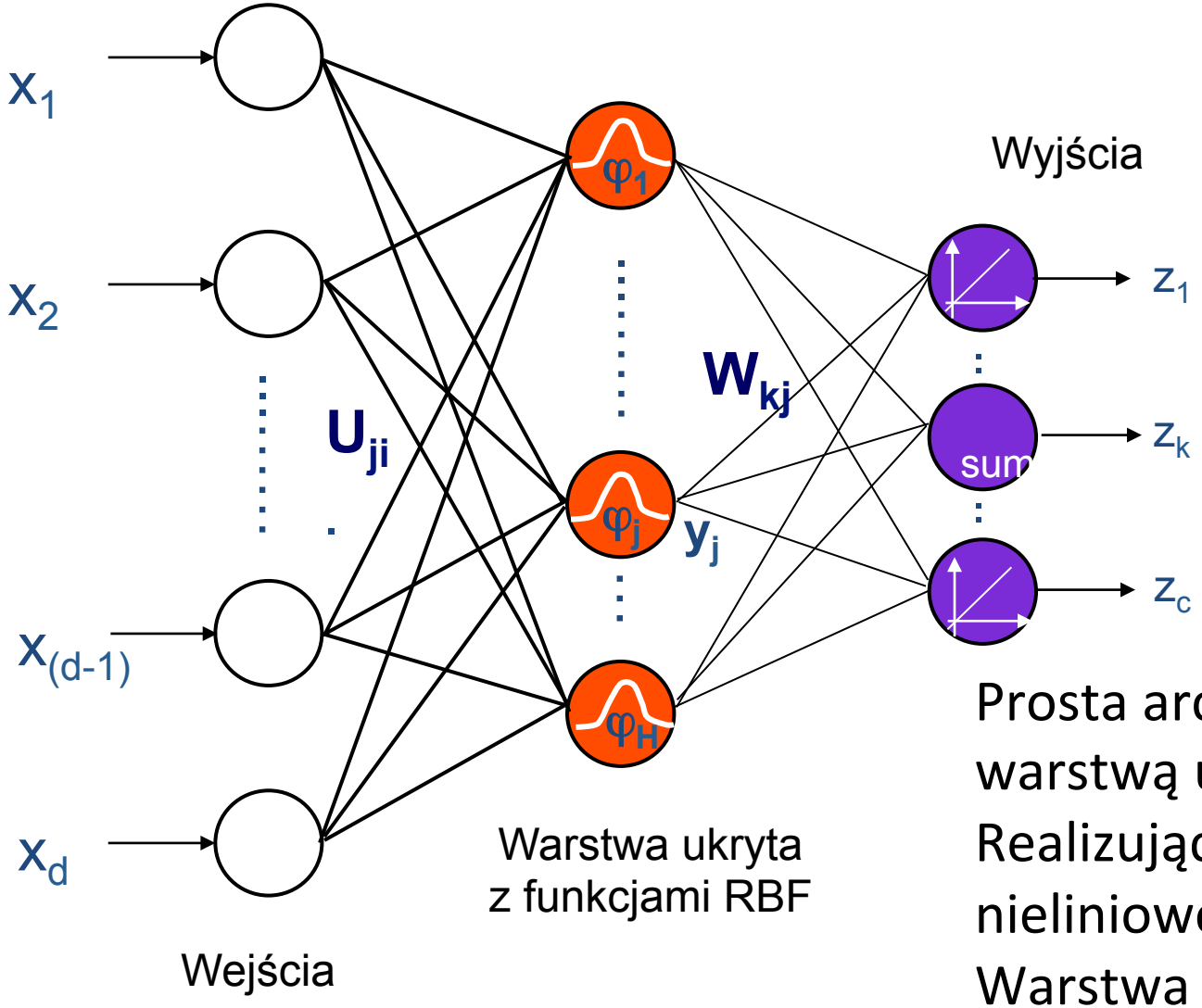
Inne materiały - internet

- A.Bartkowiak: Kernele, siecie SVM i sieci GDA.
- W.Duch: wyklady nt. Computational Intelligence
- Angielska wersja Wikipedii
- Thorsten Joachims: Support Vector and Kernel Methods – SIGIR Tutorial

Metody kernelowe

- Szersza klasa
  - Wykorzystują przekształcenie oryginalnej przestrzeni cech funkcjami jądrowymi
    - np. w zaawansowanej wizualizacji danych
  - Niektóre powrócą na laboratoriach
- **Przykład sieci neuronowych RBF**
  - Sieci z funkcjami o symetrii kołowej
  - Neurony ukryte realizują przekształcenie nieliniowe funkcjami jądrowymi

Typowa topologia sieci RBF



Prosta architektura z warstwą ukrytą – Realizującą przekształcenie nieliniowe (funkcje RBF)
Warstwa wyjściowa – proste (liniowe) ważone sumowanie

Aproksymacja RBF funkcji ciągłej

- Zadanie aproksymacji złożonej funkcji $f(\mathbf{x})=z$
- Przyjmijmy funkcje liniową względem parametrów w_i, wykorzystującą funkcje o symetrii kołowej RBF

$$f(\mathbf{x}) = \sum_{i=1}^{m} w_i \cdot \varphi(\|\mathbf{x} - c_i\|)$$

- Radialna funkcja bazowa $\rightarrow$ funkcja φ o postaci $\varphi(\mathbf{x},\mathbf{c})= \varphi(r(\mathbf{x},\mathbf{c}))$, gdzie **r** jest odległością między punktami **x** i **c**. Punkt c nazywamy „centrum”
- Związek funkcji radialnym z funkcjami jądrowymi (ang. kernals), z parametrem σ szerokością jądra
- Najczęściej funkcja Gaussowska

Pytanie i komentarze?

Dalszy kontakt:

jerzy.stefanowski@cs.put.poznan.pl

http://www.cs.put.poznan.pl/jstefanowski/

Fundusze Europejskie Polska Cyfrowa

Rzeczpospolita Polska

Unia Europejska Europejski Fundusz Rozwoju Regionalnego