Metody sztucznej inteligencji
Zadanie 1: Perceptron Rosenblatt'a w wersji nieliniowej

dr inż. Przemysław Klęsk

1 Zbiór danych dla zadania do wykonania w domu

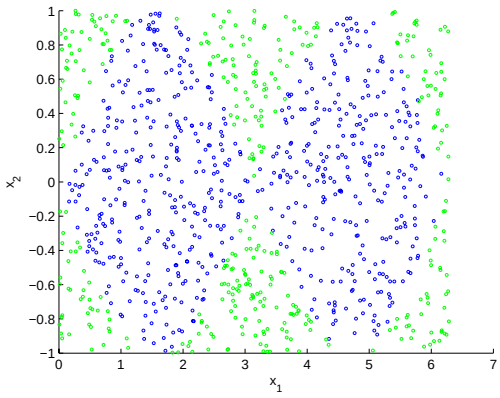
Zgodnie z tym, co zostało podane na laboratoriach, dany jest pewien zbiór par

$$\left\{(\mathbf{x}_i, y_i)\right\}_{i=1,\ldots,I} \tag{1}$$

gdzie $\mathbf{x}_i = (1, x_{i1}, x_{i2})$ są punktami rozmieszczonymi losowo w prostokącie $[0, 2\pi] \times [-1, 1]$ (pomijając zerową współrzędną ustawioną zawsze na 1), natomiast $y_i \in \left\{-1, 1\right\}$ są numerami klas nadawanymi zgodnie z regułą:

$$y_i = \begin{cases} -1, & \text{jeżeli } |\sin x_{i1}| > |x_{i2}|; \\ 1, & \text{w przeciwnym razie.} \end{cases} \tag{2}$$

Zbiór ten pokazano na rys. 1.



Rysunek 1: Zbiór danych.

Jak widać zbiór ten nie jest separowalny liniowo, tzn. nie istnieje taka prosta, która by oddzielała punkty przynależne do różnych klas.

2 Normalizacja zmiennych wejściowych

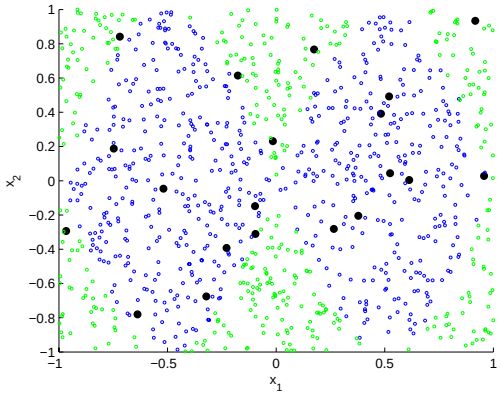
Do dalszej pracy potrzebne jest znormalizowanie zmiennych wejściowych — niech obie zmienne x_1 i x_2 zostaną znormalizowane do przedziału $[-1, 1]$. Innymi

słowy należy przeskalować współrzędne każdego punktu $\mathbf{x}_i$ w danym zbiorze, i tym samym zbiór ten nie będzie jest już zawarty w prostokącie $[0, 2\pi] \times [-1, 1]$ a w kwadracie $[-1, 1] \times [-1, 1]$.

3 Podniesienie wymiarowości przestrzeni wejściowej

Aby za pomocą algorytmu uczenia perceptronu Rosenblatt'a móc znaleźć krzywą separacji potrzeba będzie zastosować zabieg podniesienia wymiarowości przestrzeni wejściowej. Jeżeli dany zbiór nie jest separowalny liniowo w oryginalnej przestrzeni $\mathbb{R}^n$ (u nas w zadaniu $n = 2$), to można spróbować za pomocą pewnego odwzorowania podnieść ten zbiór do przestrzeni $\mathbb{R}^m$ o wyższym wymiarze, $m > n$, i być może wówczas w tej nowej przestrzeni zbiór ten będzie już można separować liniowo — tzn. za pomocą pewnej hiperpłaszczyzny. Innymi słowy każdemu punktowi $\mathbf{x}_i = (1, x_{i1}, \ldots, x_{in})$ należy przyporządkować pewien nowy dłuższy zestaw współrzędnych $\mathbf{z}_i = (1, z_{i1}, \ldots, z_{im})$. Nowe współrzędne nazywa się *cechami*, a nową nazywa się przestrzeń *przestrzenią cech*. Jak matematycznie należy to zrobić i jakie wielkości można wybrać jako cechy?

Wyobraźmy sobie, że w naszej oryginalnej przestrzeni $\mathbb{R}^n$ rozmieścimy m ustalonych punktów — oznaczmy je jako $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_m$. To rozmieszczenie może być przypadkowe, patrz rys. 2. Punkty $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_m$ zwykle nazywa się *centrami*.



Rysunek 2: Przykładowe losowe rozmieszczenie punktów $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_m$ (zaznaczone kolorem czarnym).

Wówczas, jako *cechy* $(z_{i1}, \ldots, z_{im})$ — czyli współrzędne w przestrzeni o wyższym wymiarze — dla danego punktu $\mathbf{x}_i$ możemy obrać np. odległości[1] od tego punktu do wszystkich centrów. I tak otrzymujemy nowy zbiór danych, przedstawiony poniżej w formie macierzy (takiej, z jakiej korzystaliśmy na lab-

[1]W praktyce używane będą pewne funkcje pokrewne funkcji odległości. Póki co w celu wyjaśnienia można mówić o zwykłej odległości.

oratoriach):

$$\begin{pmatrix} 1 & z_{11} & z_{12} & \cdots & z_{1m} & y_1 \\ 1 & z_{21} & z_{22} & \cdots & z_{2m} & y_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & z_{I1} & z_{I2} & \cdots & z_{Im} & y_I \end{pmatrix}, \tag{3}$$

gdzie z_{ij} jest odległością i-tego punktu od j-tego centrum:

$$z_{ij} = \sqrt{(x_{i1} - c_{j1})^2 + (x_{i2} - c_{j2})^2 + \cdots + (x_{in} - c_{jn})^2}. \tag{4}$$

I tak przygotowany nowy zbiór danych można by już poddać pod działanie zwykłego algorytmu perceptronu Rosenblatt'a (poznanego na laboratoriach) w poszukiwaniu zestawu wag $(w_0, w_1, \ldots, w_m)$, który separuje liniowo nowy zbiór w przestrzeni $\mathbb{R}^m$ za pomocą hiperpłaszczyzny o równaniu:

$$w_0 + w_1 z_1 + w_2 z_2 + \cdots + w_m z_m = 0, \tag{5}$$

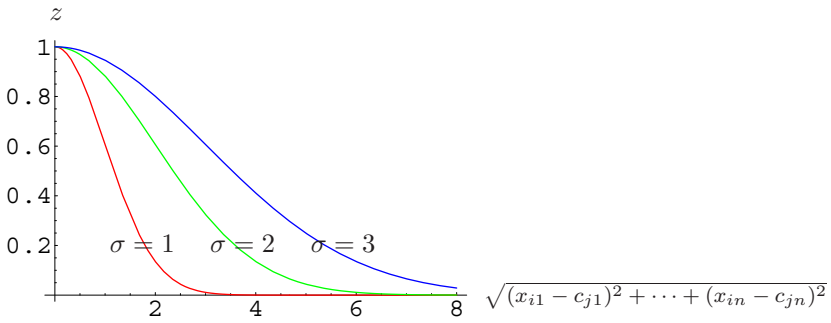
które jest liniowym odpowiednikiem prostej z przestrzeni $\mathbb{R}^2$.

Warto wyobrazić sobie, że znaleziona liniowa granica separacji w przestrzeni cech tj. w przestrzeni $\mathbb{R}^m$ odpowiada pewnej krzywoliniowej granicy w oryginalnej przestrzeni $\mathbb{R}^n$.

W praktyce zamiast zwykłej funkcji odległości, patrz wzór (4), stosuje się nieco inne funkcje, ale bazujące na wyrażeniu odległości. I tak w faktycznej implementacji w miejsce funkcji (4) proszę zastosować tzw. *jądrowe funkcje gaussowskie* o wzorze:

$$z_{ij} = \exp\Big(-\frac{(x_{i1} - c_{j1})^2 + (x_{i2} - c_{j2})^2 + \cdots + (x_{in} - c_{jn})^2}{2\sigma^2}\Big). \tag{6}$$

Jak można zauważyć, pod działaniem funkcji wykładniczej w liczniku znajduje się właśnie wyrażenie związane z odległością — dokładnie jest to kwadrat odległości (pominięty został pierwiastek). Wykresem funkcji Gaussa jest krzywa o kształcie dzwonowym (jeżeli na oś argumentów nanieść właśnie wyrażenie związane z odległością). Patrz rys. 6. Krzywa ta osiąga wartość 1 w punkcie 0,



Rysunek 3: Jądrowe funkcje gaussowskie.

tj. wtedy, gdy rozpatrywana odległość pomiędzy punktem $\mathbf{x}_i$ a centrum $\mathbf{c}_j$ jest

równa zeru, natomiast gdy ta odległość zmierza do nieskończoności, to krzywa Gaussa zbiega do wartości 0. O szerokości dzwonu decyduje parametr σ. Im większe σ, tym szersze zbocze dzwonu, im σ bliższe zeru, tym krzywa bliższa impulsowi w punkcie zero. Jak widać funkcja gaussowska jest tak naprawdę funkcją „bliskości" tzn. maleje wraz z odległością, co w niczym nie przeszkadza, aby stosować ją jako cechę.

W trakcie wykonywania zadania w domu, wartości parametru σ trzeba będzie dobierać eksperymentalnie metodą prób i błędów, obserwując uzyskiwane granice separacji. Jednocześnie należy przy tym brać pod uwagę liczbę centrów m (tj. jak wiele ich rozstawiono). Jest to również parametr dobieralny. Jeżeli rozstawi się dużą liczbę centrów i będą one gęsto pokrywały oryginalną przestrzeń wejściową, to można wówczas ustawić stosunkowo małe σ. Każde centrum $\mathbf{c}_j$ będzie wówczas miało mały lokalny zasięg i wartości cechy (związanej z tym centrum) bliskie jedności będą nadawane tylko tym punktom $\mathbf{x}_i$ leżącym naprawdę blisko tego centrum. Nieco dalsze punkty będą miały tę cechę szybko wygaszoną do wartości bliskiej zeru. Natomiast w sytuacji przeciwnej, gdy rozstawimy mniej centrów, sugeruje się obdarzać je szerszym zasięgiem działania poprzez większe σ.

4 Warunek stopu

Przeniesienie danego zbioru do przestrzeni cech — przestrzeni o wyższej wymiarowości — nie daje nam całkowitej pewności, że zbiór w tejże przestrzeni będzie już liniowo separowalny, a jedynie tylko taką nadzieję. W każdym razie separacja powinna być tam ułatwiona. Może jednakże istnieć pewna mała liczba punktów, które pomimo wielu iteracji algorytmu, nie będą klasyfikowane poprawnie. Istnieje więc groźba, że algorytm mógłby pracować w nieskończonej pętli, jako że zbiór aktualnie błędnie sklasyfikowanych punktów nigdy nie byłby pusty.

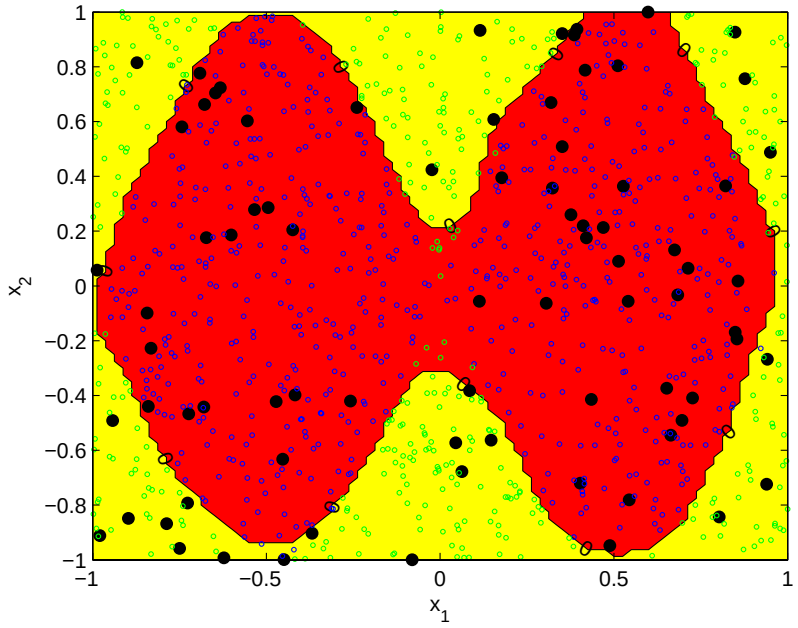
Potrzeba zatem wprowadzić dodatkowy warunek stopu tj. ograniczenie na maksymalną liczbę iteracji — k_{max}. I tak algorytm należy przerywać wtedy, gdy zajdzie jedno z dwojga: zbiór błędnie sklasyfikowanych punktów będzie pusty lub licznik k aktualizacji wag osiągnie k_{max}.

5 Sugerowane ustawienia i przykładowe wyniki

Niech dany zbiór punktów będzie o rozmiarze $I = 1000$. Należy eksperymentalnie metodą prób i błędów dobrać liczbę centrów m (a tym samym wymiarowość przestrzeni cech). Sugeruje się badać m w zakresie od 20 do 100. Sugeruje się eksperymentować na wartościach σ w zakresie od 1 do 0.1. Górne ograniczenie na liczbę iteracji k_{max} sugeruje się badać w zakresie od 500 do 5000.

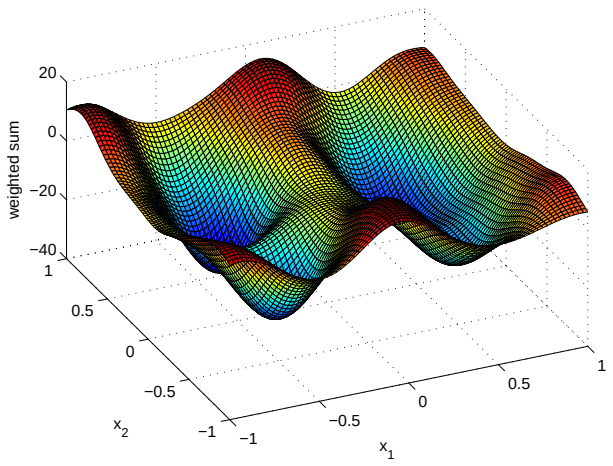
Po zatrzymaniu się algorytmu uzyskujemy wektor wag $(w_0, w_1, \ldots, w_m)$. Krzywoliniową granicę separacji wykreślamy w oryginalnej przestrzeni, u nas w zadaniu w przestrzeni $\mathbb{R}^2$. Wygodnie jest użyć do tego wykresu warstwicowego (poziomicowego) — polecenia MATLABA: `contour` lub `contourf`. Wykres otrzymuje się w ten sposób, że każdy kreślony punkt siatki (z polecenia `meshgrid`) należy najpierw odwzorować do przestrzeni cech, a dopiero potem obliczyć, jaką wartość $\{-1, 1\}$ zwraca dla niego perceptron w oparciu o wagi

$(w_0, w_1, \ldots, w_m)$ i nanieść do wykresu. Patrz rys. 4.

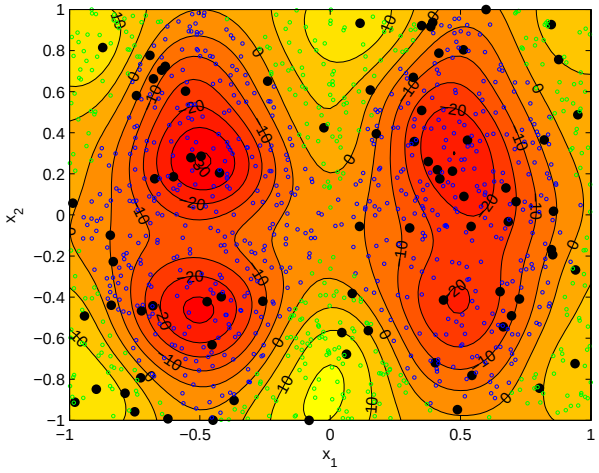


Rysunek 4: Wykres warstwicowy ilustrujący granicę separacji. Tylko jedna warstwica na wysokości równej 0.

Dodatkowo, na podobnej zasadzie można wykreślić sobie również wykres powierzchniowy wyjścia perceptronu (polecenie `surf` zamiast `contour`). Przy czym dla lepszego zobrazowania działania perceptronu warto wówczas wyłączyć progowanie w pereceptronie tzn. przyciąganie do wartości -1 lub 1. Czyli należałoby zwracać wartości samej sumy ważonej — da to na wykresie powierzchnię o gładkim przebiegu. Patrz rysunki 5 i 6.



Rysunek 5: Wykres powierzchniowy sumy ważonej $w_0+w_1z_1+w_2z_2+\cdots+w_mz_m$ perceptronu (bez progowania) narysowany nad układem zmiennych x_1, x_2. Każdy punkt (x_1, x_2) z dziedziny $[-1, 1]\times[-1, 1]$ zostaje najpierw odwzorowany w punkt $(z_1, z_2, \ldots, z_m)$, a dopiero później oblicza się dla niego wartość sumy ważonej i nanosi tę wartość na wykres.



Rysunek 6: Wykres warstwicowy sumy ważonej perceptronu (bez progowania). Większa liczba warstwic.