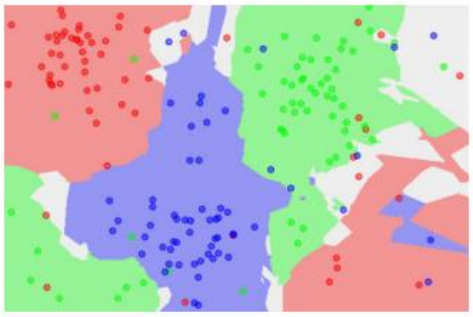
13. Nelineární klasifikace a neuronové sítě typu vícevrstvý perceptron, učení neuronových sítí- algoritmus zpětné propagace.

Metody nelineární klasifikace:

- Vzdálenostní metody
  - NN, kNN – neparametrický přístup
- Pametrické metody
  - Zapojení vyšších mocnin příznaků v rámci lineárního klasifikátoru
  - Kernelové transformace
    - Kernel SVM
  - Gaussovské mixturové modely
    - Viz přednáška 11
  - ...
  - Neuronové sítě typu vícevrstvý perceptron



Nelineární klasifikace a neuronové sítě typu vícevrstvý perceptron

Vzdálenostní metody

• Patří mezi nelineární metody

• Mají výhody i nevýhody (viz předchozí přednášky)

•Hlavní nevýhoda – Nejsou parametrické a s klasifikátorem je nutno distribuovat i trénovací data!

Parametrické metody

• Problém nelineární klasifikace se řeší **převodem na úlohu lineární klasifikace v jiném prostoru příznaku**

• Tento prostor vznikne rozšířením nebo transformací původního prostoru

Lineární klasifikace v rámci nelineárních metod

Metoda zapojení vyšších mocnin příznaků

- Lineární klasifikace probíhá v prostoru o vyšším počtu dimenzí, nové dimenze jsou dány vyššími mocninami

Kernelové transformace

- Lineární klasifikace probíhá v (nelineárně) transformovaném prostoru s vyšší dimenzí

Neuronové sítě (sériové zapojení lin. klasifikátorů)

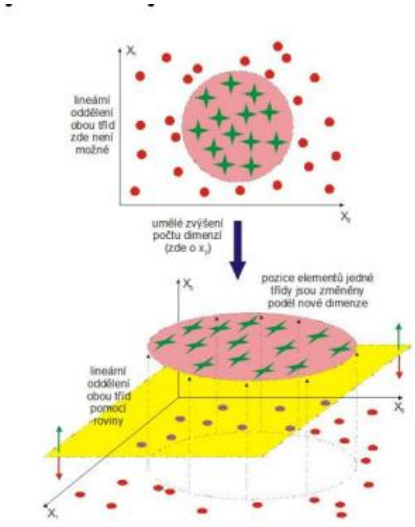
- Lineární klasifikace probíhá postupně v každé vrstvě
  - Např. v poslední vrstvě probíhá nad příznaky danými výstupem z předchozí (předposlední) vrstvy
    - Jde také prostor, oproti prostoru příznaků je ale nelineárně transformovaný a může mít zcela jiný počet dimenzí

Zapojení vyšších mocnin příznaků

• Obdobný jako u řešení problému polynomické regrese, který se převede na řešení problému lineární regrese s novými příznaky danými vyššími mocninami původních příznaků

• **Obecně platí, že v prostoru s dostatečným počtem dimenzí lze nakonec vždy najít lineární oddělovač (nadrovinu).**

• N datových bodů je možno lineárně oddělit v prostoru s $N - 1$ nebo více dimenzemi



Obr. 1. Princip vzniku možnosti lineárního oddělení dvou tříd s nelineárními hranicemi pomocí přidané dimenze.

Nevýhody zapojení vyšších mocnin příznak:

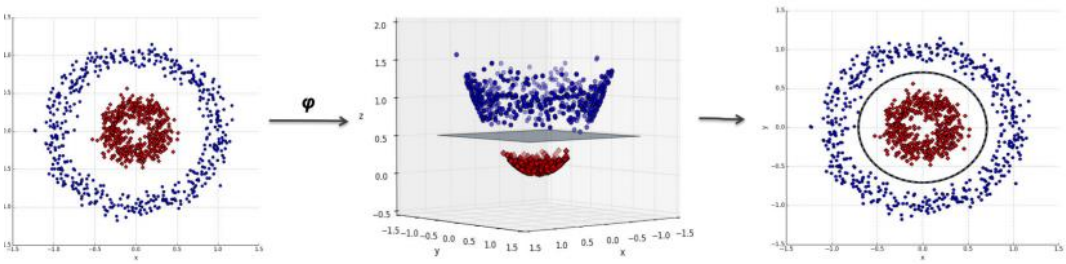
- Při zapojení druhých mocnin se počet příznaků zvýší na druhou
- Při zahrnutí třetích mocnin na třetí atd.
- Problém vzrůstajícího počtu příznaků se projeví zvláště např. při zpracování obrazu
    - Jas každého pixelu zde může být jeden příznak
- V praxi je proto obtížné řešit nelineární klasifikaci tímto způsobem

Jádrová (kernel) transformace

Jádrová funkce (transformace) transformuje data z prostoru příznaků do jiného prostoru (typicky o vyšší dimenzi), kde jsou tato data již lineárně separovatelná:

Používá se celá řada:

- Polynomiální
- Gaussova radiální bázová funkce = Gaussian Radial Basis Function (RBF):

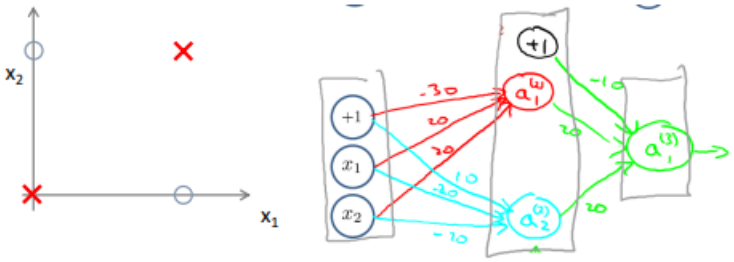


Shrnutí

- U kernelových metod neroste počet parametrů stejně jako při pouhém zapojení vyšších mocnin příznaků
  - Výhoda pro systémy s velkým počtem příznaků
- Počet parametrů ovšem roste s velikostí trénovací množiny !!
  - Pro rozsáhlé trénovací množiny je proto použitelnost kernelů obtížná
  - Nutno řešit dalšími modifikacemi – škálováním
- Použití pro více tříd je možné různými způsoby, podobně jako např. u lineárního SVM

Sériové zapojení lineárních klasifikátorů

Řešení: sériové zapojení lineárních klasifikátorů



x_1	x_2	$a_1^{(2)}$	$a_2^{(2)}$	$y = a_1^{(3)}$
0	0	0	1	1
0	1	0	0	0
1	0	0	0	0
1	1	1	0	1

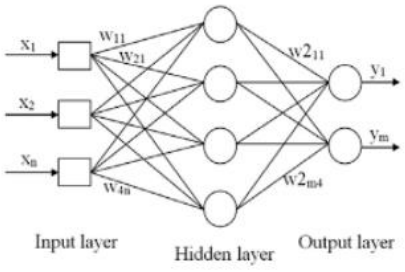
- Příznaky a_1 a a_2 dané jako výstup skryté vrstvy jsou pomocné
  - Příznak a_1 určuje, zda jsou oba vstupy rovny číslu 1
  - Příznak a_2, zda jsou oba vstupy rovny číslu 0
- Pomocí těchto dvou příznaků je možné v další vrstvě rozhodnout o výsledku
- *Pozn.: hodnoty vah v tomto příkladu byly nastaveny ručně*
  - *Reálně se nastaví během trénování podle trénovacích dat a počáteční inicializace*
  - *Mohly by vyjít podobně jako v příkladu nebo číselně odlišně, ale příznaky by fungovaly i pokud by například vektory vah u a_1 a a_2 byly úplně prohozené*

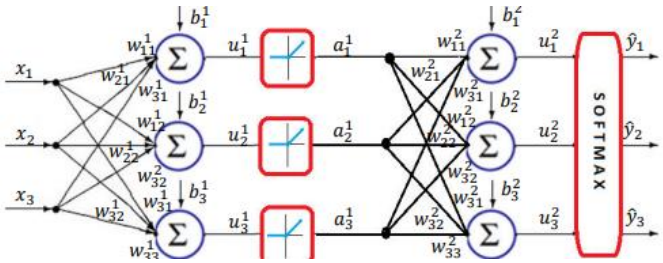
Jde o efektivní řešení problému nelineární klasifikace

• **Počet parametrů sice roste s každou další vrstvou ale neroste s velikostí dat** (jako u kernelů) ani s nějakou mocninou velikosti příznakového vektoru.

NEURONOVÉ SÍTĚ

• Matematický model tvořený sériovým a paralelním spojením umělých neuronů





Skryté vrstvy a jejich funkce:

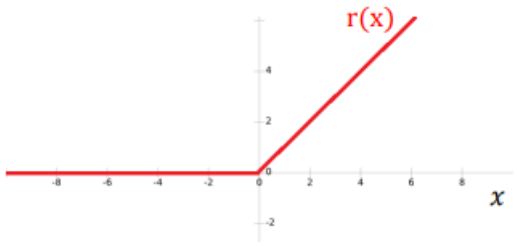
• Pomocí těchto vrstev si síť vytváří sama svoje vlastní příznaky

• Obecně příznaky ve vyšší vrstvě směrem od vstupu reprezentují vyšší úroveň rozhodování (abstrakce) na základě hodnot jednodušších příznaků z předcházející vrstvy

Aktivační funkce ReLU

ReLU je správně označení neuronu, který využívá aktivační funkci r definovanou jako:

$$r(x) = rectifier(x) = max(0, x)$$:



Učení neuronových sítí- algoritmus zpětné propagace

• Obdobným iteračním způsobem jako sítě s jedním neuronem respektive jednou vrstvou

• Podle vzniklé chyby se opraví váhy na poslední výstupní vrstvě a chyba se pak propaguje dále do předposlední vrstvy, kde se také upraví vektory vah

• Celý algoritmus se pak nazývá **Zpětná propagace**

Ve spojitosti se složenou derivací platí řetízkové pravidlo (chain rule)

Je-li $F(x) = f(g(x))$, pak $\frac{dF}{dx} = \frac{dF}{dg}\frac{dg}{dx}$

Příklad:

- $F(x) = \frac{(x+4)^3}{(x-1)^3} = g^3$, kde $g = \frac{x+4}{x-1}$
- $\frac{dF}{dx} = \frac{dF}{dg}\frac{dg}{dx} = 3g^2\frac{dg}{dx} = 3\frac{(x+4)^2}{(x-1)^2}\frac{-5}{(x-1)^2} = -15\frac{(x+4)^2}{(x-1)^4}$

• Pro učení konkrétního vektoru vah v dané vrstvě metodou SGD potřebujeme vyjádřit derivaci výstupu vzhledem k tomuto vektoru

• Jde o několikanásobně složenou funkci, vektor vah se postupně:

  • Násobí vstupem do dané vrstvy

  • Výsledek se dosadí do příslušné aktivační funkce a přenásobí dalším vektorem vah

  • …

  • Nakonec dojdeme až do funkce Softmax

• **Pro výpočet derivace podle daného parametru se použije řetízkové pravidlo**

• „Zpětně jsme přenesli gradient z výstupu funkce softmax na jeden z vektorů vah“

• Postupná aplikace řetízkového pravidla směrem z výstupu (sítě) na vstup (sítě) představuje zpětné šíření gradientu z výstupu na vstup

• Pokud známe hodnotu gradientu v bodě A, je možné gradient pomocí derivace složené funkce (řetízkové pravidla) vyjádřit i pro prvek sítě B, který je blíže vstupu

• V rámci učení se tak gradient šíří od výstupu směrem ke vstupu

Aspekty trénování

• Předzpracování dat - Pro trénování neuronových sítí je vhodné provádět standardizaci dat na nulovou střední hodnotu a jednotkový rozptyl

• Inicializace parametrů (váhových koeficientů) - Váhy neuronů nesmí být nastaveny na stejné hodnoty (např. 0)