

Signály a informace

Přednáška č.12 Biologicky inspirované systémy

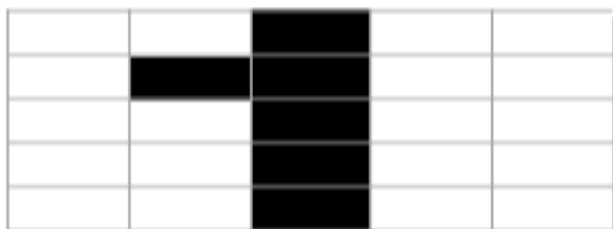


Opakování z minulé přednášky

- Signály v sobě nesou větší či menší množství informace
- Získáváním informace ze signálů/dat se zabývá obor data mining
- Jednou z dílčích úloh je úloha klasifikace, tj. roztřídění dat do předem daných tříd
- Ukázali jsme si jednoduchou metodu založenou na **měření vzdálenosti a nalezení nejbližšího souseda** mezi vzory
 - Metoda funguje relativně dobře, jsou-li jednotlivé třídy reprezentované mnoha (např. tisíce) vzory
 - **Pracovat s mnoha vzory je ale neefektivní**, s každým dalším vzorem vzrůstají výpočetní a paměťové nároky
- **Existují i efektivnější metody dosahující zároveň i lepších výsledků**

Vektor příznaků

- Klasifikovaný objekt lze reprezentovat jako vektor příznaků s rozměrem D
- Např. jak bude vypadat vektor příznaků pro černobílý obrázek a jaká bude dimenze toho vektoru?



| | | | | |
|-----|-----|---|-----|-----|
| 255 | 255 | 0 | 255 | 255 |
| 255 | 0 | 0 | 255 | 255 |
| 255 | 255 | 0 | 255 | 255 |
| 255 | 255 | 0 | 255 | 255 |
| 255 | 255 | 0 | 255 | 255 |

Vektor příznaků vznikne zřetězením řádků:

$x = [255, 255, 0, 255, 255, 255, 0, 0, 255, 255 \dots \dots 0, 255, 255]$

Lze zřetěžit sloupce?

ANO, prováděno v Matlabu v rámci příkazu RESHAPE

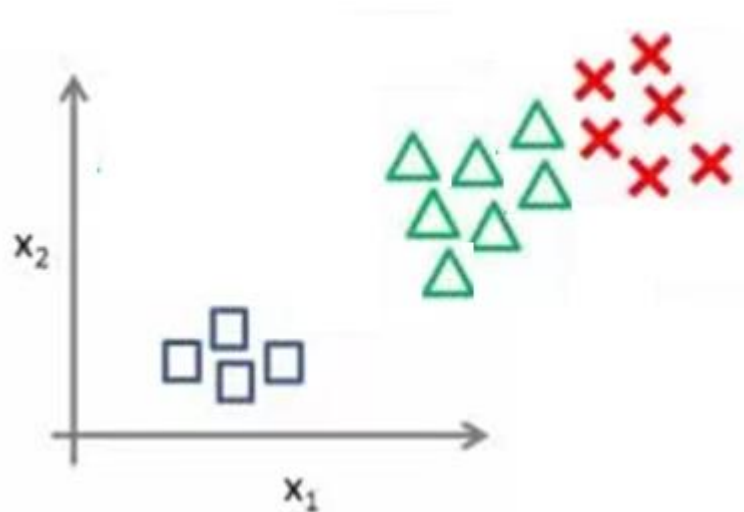
- Hodnoty vektoru budou odpovídat hodnotám jasu jednotlivých pixelů
- $D = \text{dimenze} = a * b$, kde a je šířka obrázku a b výška v pixelech

Prostor příznaků

- Každý objekt respektive jeho vektorová reprezentace určuje v daném v D -dimenzionálním prostoru příznaků jeden bod
- Prostory o velkém počtu dimenzí nelze vykreslovat na 2D ploše 😞
 - Počet dimenzí si proto pro další výklad omezíme na dvě až tři
 - Nebudeme chvíli pracovat s obrázky 😞

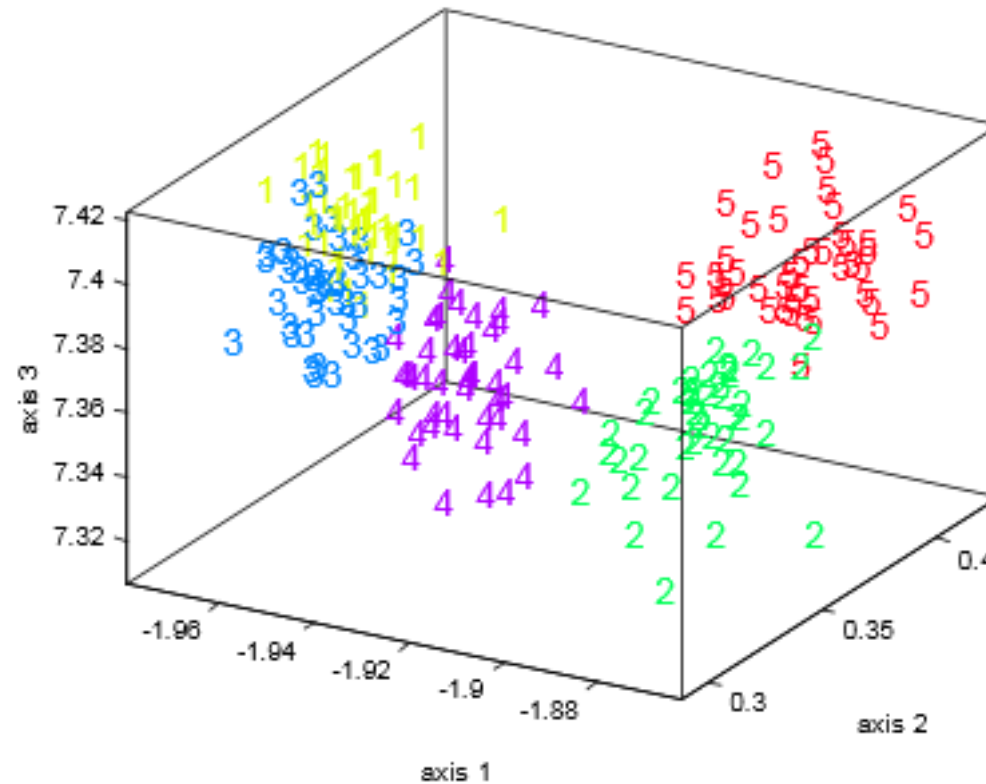
Prostor příznaků – 2D příklad

- Pro objekty, popsané jen 2 příznaky, x_1 a x_2 , je prostor příznaků 2D rovina
- Jednotlivé třídy se pak znázorňují různými symboly a/nebo barvami
- Objekty patřící do jednotlivých tříd mají podobné hodnoty příznaků
 - Vytváří v prostoru příznaků shluky (jsou blízko sebe)



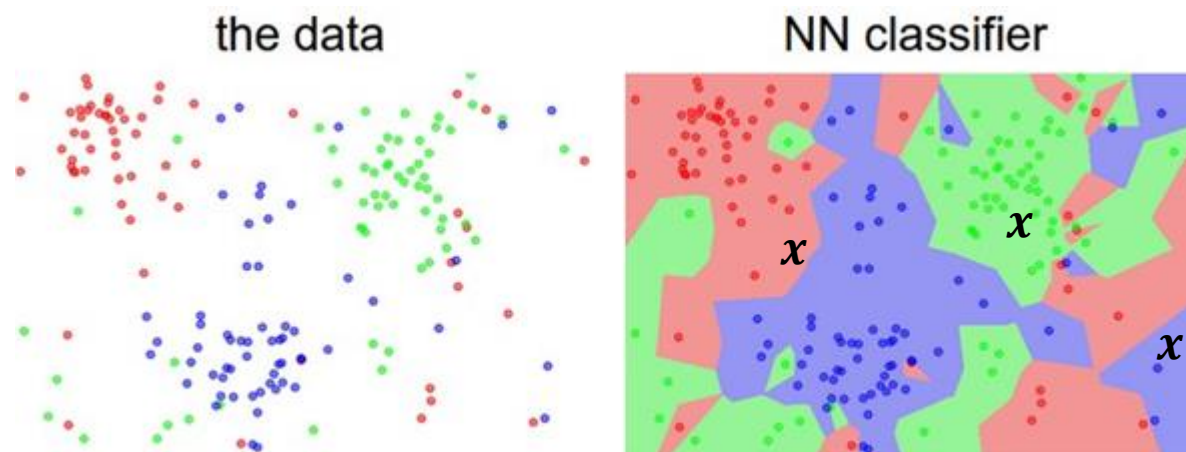
- Klasifikátor osob na třídy **Dítě**, **Muž**, **Žena**
- x_1 ...výška, x_2 ...hmotnost
- Čtverečky...třída Děti
- Křížky... třída Muži
- Trojúhelníky...třída Ženy

Prostor příznaků – 3D ilustrace pro 5 tříd



Jak funguje NN klasifikátor v prostoru příznaků?

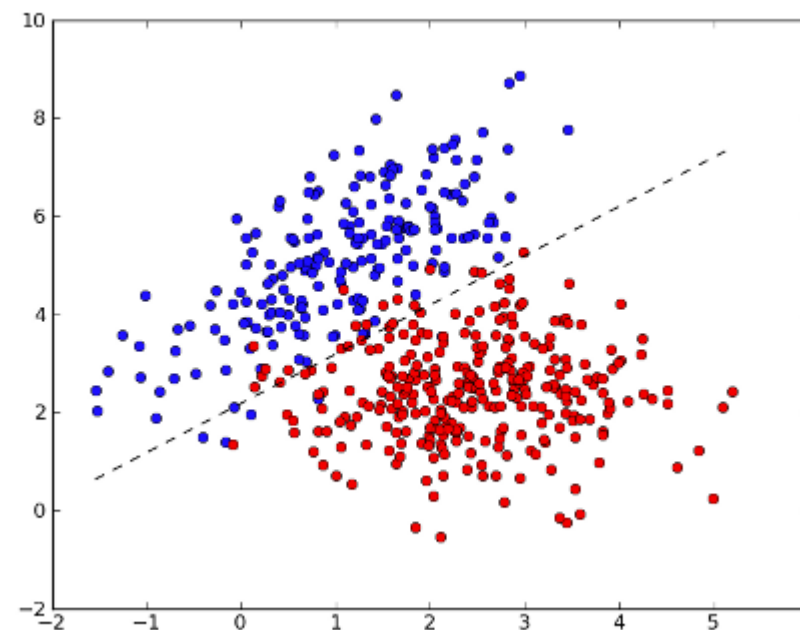
- NN klasifikátor definuje oblasti odpovídající jednotlivým třídám
 - Hranice oblastí jsou určeny rozložením všech trénovacích bodů v prostoru příznaků
 - Jsou to často složité nelineární křivky



- Neznámý objekt x , bod v prostoru, je zařazen do dané třídy podle toho, ve které oblasti leží

Co když je ale problém jednodušší?

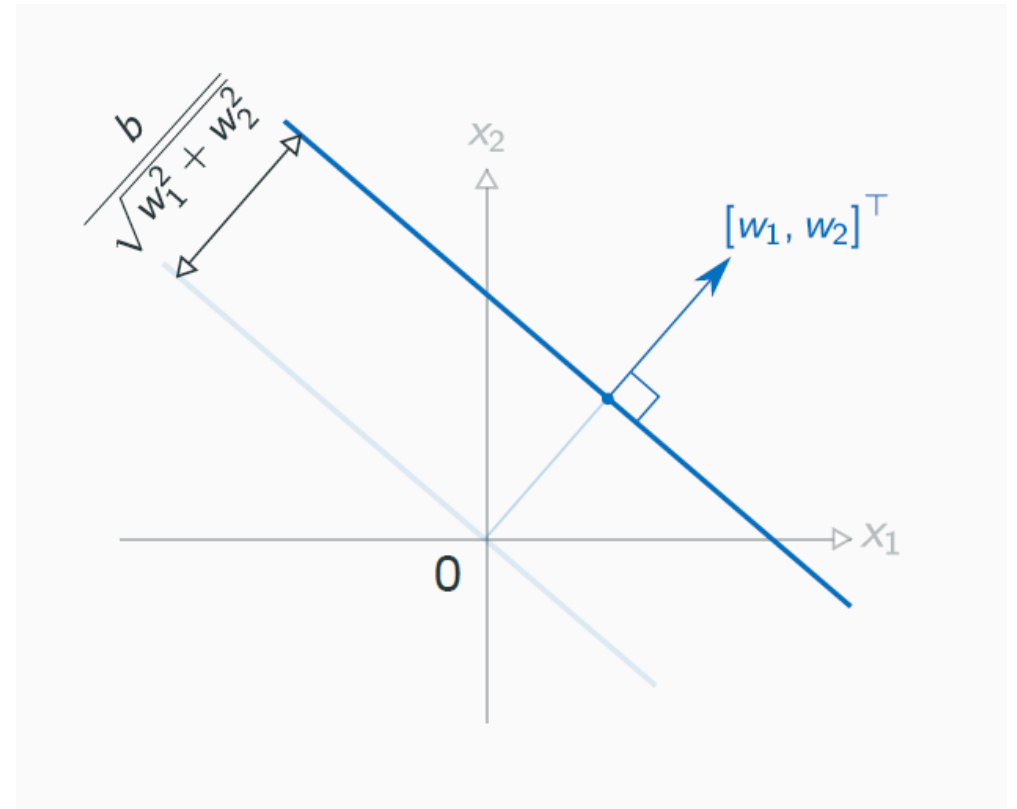
- V některých úlohách je počet tříd omezen jen na dvě = binární klasifikace
 - Pacient má/nemá nemoc, výrobek je bezvadný/zmetek
- Dané dvě třídy lze navíc někdy v prostoru příznaků separovat přímkou
 - I když třeba za cenu určité chyby klasifikace....
- Vyplatí se pak používat NN klasifikátor?
 - Když NN klasifikace trvá dlouho, protože je třeba vyhodnocovat vzdálenost ke všem objektům v trénovací množině?
- Nebyl by lepší klasifikátor definovaný rovnicí přímky, který by určil příslušnost ke třídě na základě toho, v jaké polorovině vzhledem k dané přímce leží objekt x ?



Nejprve opakování z matematiky:

Obecná rovnice přímky ve 2D

- Obecný tvar rovnice přímky v prostoru o souřadnicích x_1, x_2 má tvar:
 - $w_1x_1 + w_2x_2 + b = 0$
 - \mathbf{w} je normálový vektor (kolmý na směr přímky)
 - b souvisí s posunem od počátku (bias)
- Příklad:
 - Rovnice přímky: $-2x_1 + x_2 - 1 = 0$
 - $\mathbf{w} = [w_1; w_2] = [-2; 1]$
 - $b = -1$



Rovnice přímky pomocí skalárního součinu

- Obecnou rovnici přímky ve 2D lze zapsat jako:

$$w_1x_1 + w_2x_2 + b = \sum_{i=1}^2 w_ix_i + b = \mathbf{w}^T \mathbf{x} + b = \mathbf{x}^T \mathbf{w} + b$$

kde $\mathbf{w}^T \mathbf{x}$ je skalární součin vektoru \mathbf{w} a \mathbf{x}

- Někdy se bias b označuje jako w_3 , pak platí:

$$w_1x_1 + w_2x_2 + w_3 = \sum_{i=1}^3 w_i\tilde{x}_i = \mathbf{w}^T \tilde{\mathbf{x}} = \tilde{\mathbf{x}}^T \mathbf{w}$$

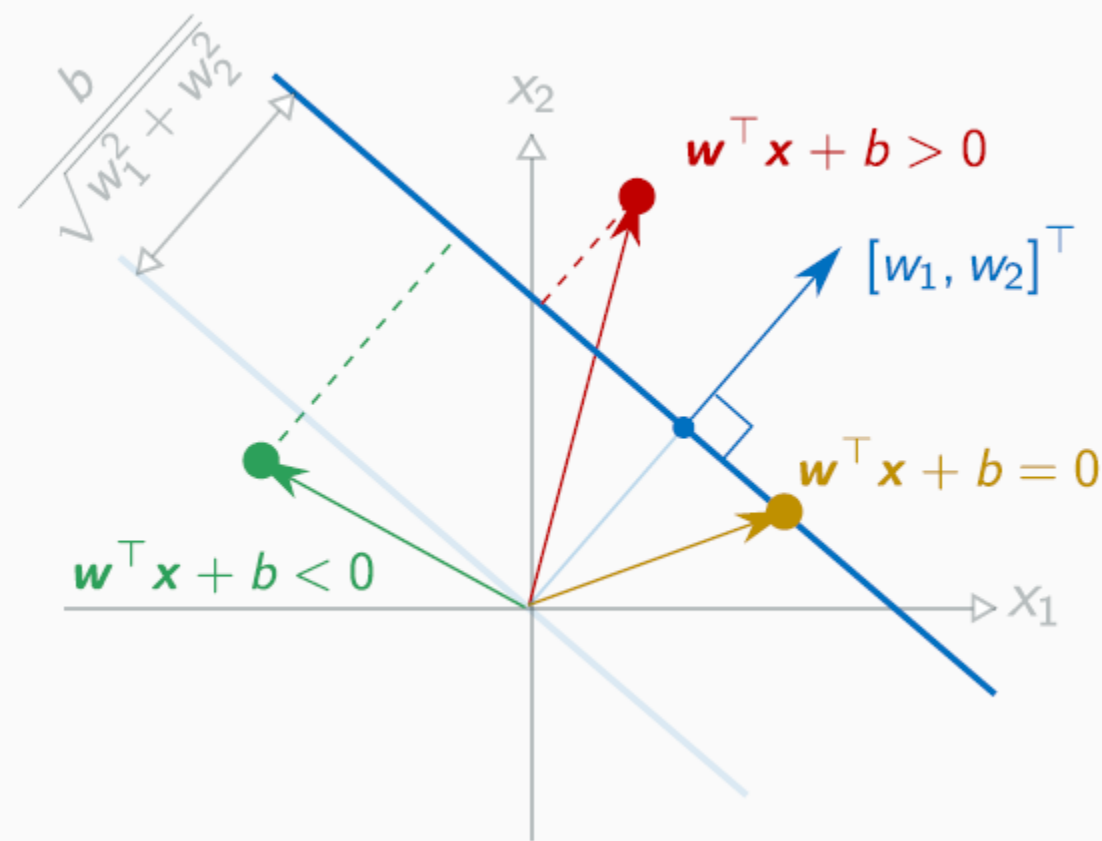
kde $\tilde{\mathbf{x}}$ je vektor \mathbf{x} rozšířený o složku x_3 s hodnotou 1: $\tilde{\mathbf{x}} = [x_1; x_2; 1]$

Poloha bodu vůči přímce ve 2D

- Orientovaná vzdálenost d bodu \mathbf{x} od přímky s param. \mathbf{w} se definuje jako:

$$d = \frac{w_1 x_1 + w_2 x_2 + b}{\sqrt{w_1^2 + w_2^2}} = \frac{\mathbf{w}^T \mathbf{x} + b}{\sqrt{w_1^2 + w_2^2}}$$

- Bod \mathbf{x} leží ve vzdálenosti d
 - ve směru vektoru \mathbf{w} (nad přímkou)
 $d > 0$ tedy $\mathbf{w}^T \mathbf{x} + b > 0$
 - proti směru vektoru \mathbf{w} (pod přímkou)
 $d < 0$ tedy $\mathbf{w}^T \mathbf{x} + b < 0$
 - přímo na přímce
 $d = 0$ tedy $\mathbf{w}^T \mathbf{x} + b = 0$



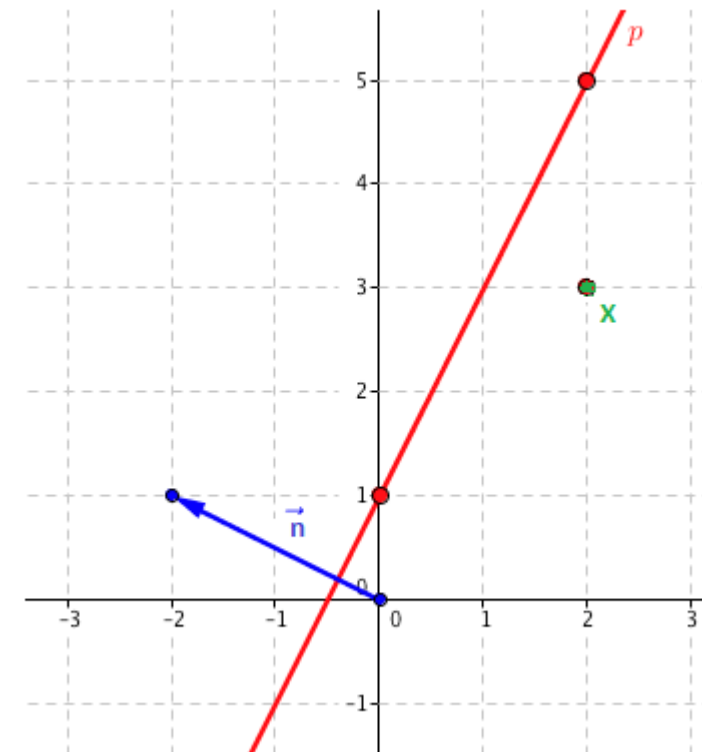
Příklad:

- Rovnice přímky: $-2x_1 + x_2 - 1 = 0$
- Vektor parametrů $\mathbf{w} = [w_1; w_2; w_3] = [-2; 1; -1]$
- Bod $\mathbf{x} = [2; 3]$ $\Rightarrow \tilde{\mathbf{x}} = [\mathbf{x}; x_{n+1} = 1] = [2; 3; 1]$
 - Leží tedy pod přímkou proti směru vektoru $\mathbf{n} = [-2; 1]$

- Dosazení do rovnice:

$$-2 * 2 + 1 * 3 - 1 * 1 = \sum_{i=1}^{n+1} w_i \tilde{x}_i = \mathbf{w}^T \tilde{\mathbf{x}} = [-2, 1, -1][2; 3; 1] = -2$$

- Algoritmicky lze realizovat pomocí skalárního součinu, nikoli dosazováním do sumy pomocí cyklu FOR



Vícerozměrné prostory

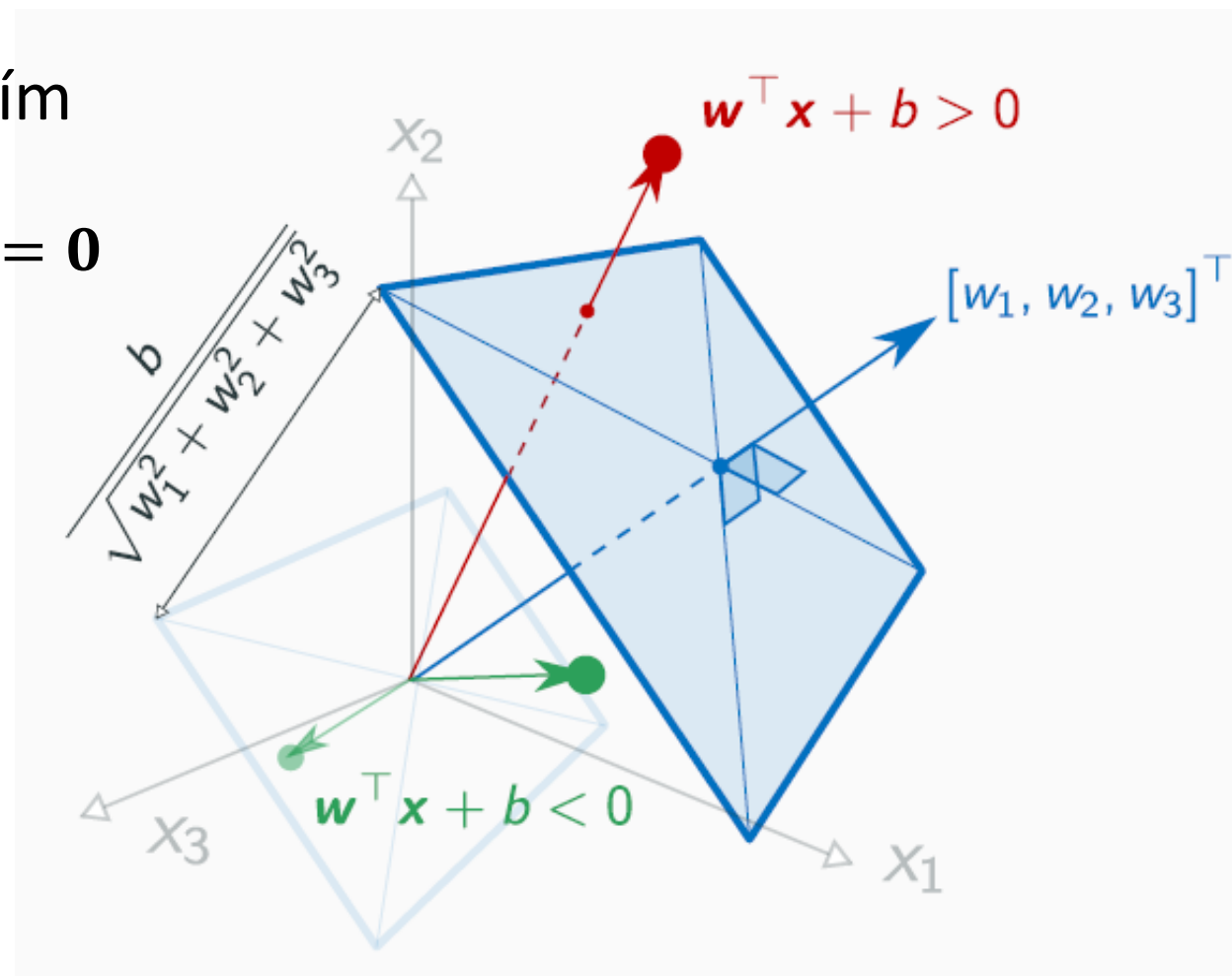
- Obecná rovnice v N dimenzionálním prostoru má tvar

$$w_1x_1 + w_2x_2 + \dots + w_nx_n + w_{n+1} = 0$$

- Např. pro 3D je o rovnici roviny

$$w_1x_1 + w_2x_2 + w_3x_3 + w_4 = 0$$

- Vše funguje principiálně stejně



Funkce sigmoida

- Funkce definovaná jako:

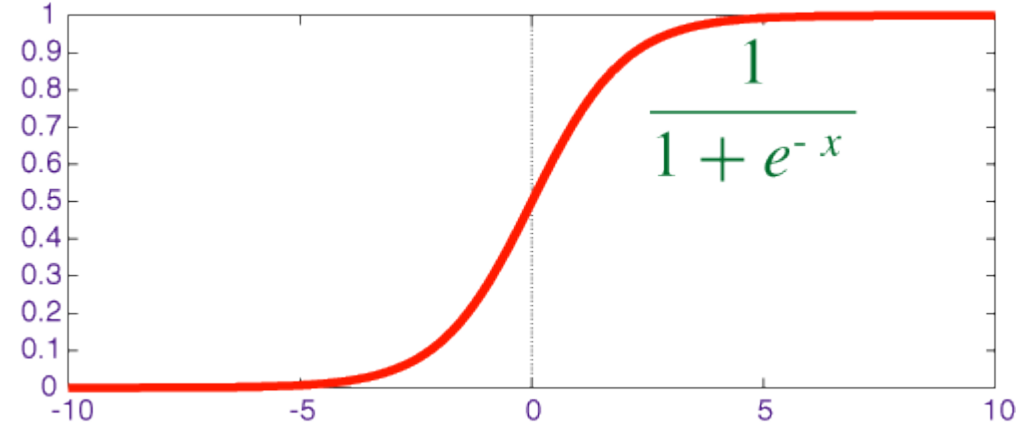
- $\text{sigmoid}(x) = \sigma(x) = \frac{e^x}{1+e^x} = \frac{1}{1+e^{-x}}$

- Má zajímavé vlastnosti:

- definiční obor $(-\infty, +\infty)$
 - obor hodnot $(0,1)$

\Rightarrow „přiřazuje reálným číslům pravděpodobnost“

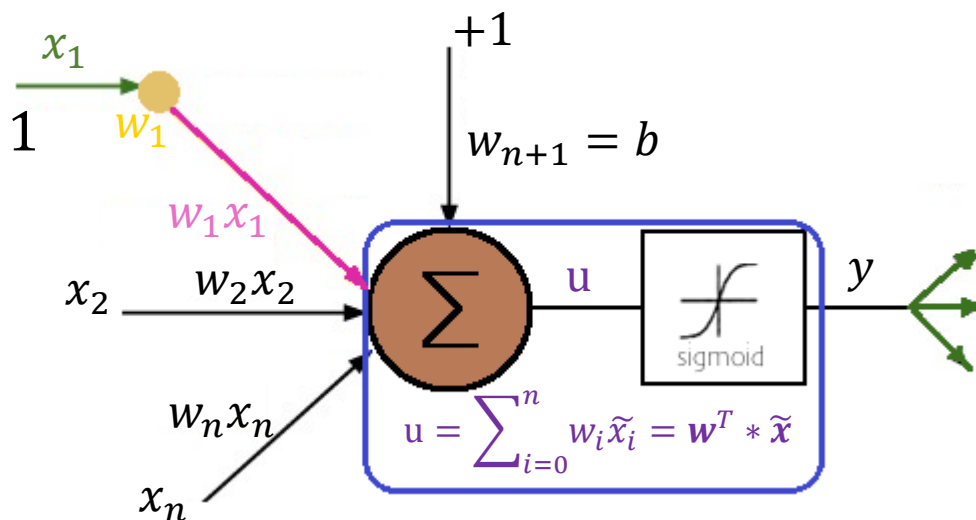
$$\sigma(x) > 0.5 \text{ pro } x > 0 \quad \sigma(x) < 0.5 \text{ pro } x < 0$$



Binární lineární klasifikace

Binární lineární klasifikátor se sigmoidou #1

- Vstupem je vektor $\mathbf{x} = [x_1; x_2; \dots; x_n]$
 - Například hodnoty pixelů testovacího obrázku
- Každá vstupní hodnota x_i se vynásobí vahou w_i , výsledek se sečte a přičte se k němu váha w_{n+1}
 - To lze zapsat jako $\sum_{i=1}^{n+1} w_i \tilde{x}_i = \mathbf{w}^T \tilde{\mathbf{x}} = u$
 - Kde $\tilde{\mathbf{x}}$ je původní vektor \mathbf{x} rozšířený o hodnotu $x_{n+1} = 1$
- Hodnota tohoto součinu říká, zda bod \mathbf{x} leží v prostoru o n dimenzích a souřadných osách x_1, x_2, \dots, x_n nad, pod nebo přímo na podploše o dimenzi $n - 1$, která je definována rovnicí
$$w_1 x_1 + \dots + w_n x_n + w_{n+1} = 0$$
- Hodnota součinu u se dosadí do sigmoidy
 - “Převeď se na pravděpodobnost v rozmezí (0,1)”

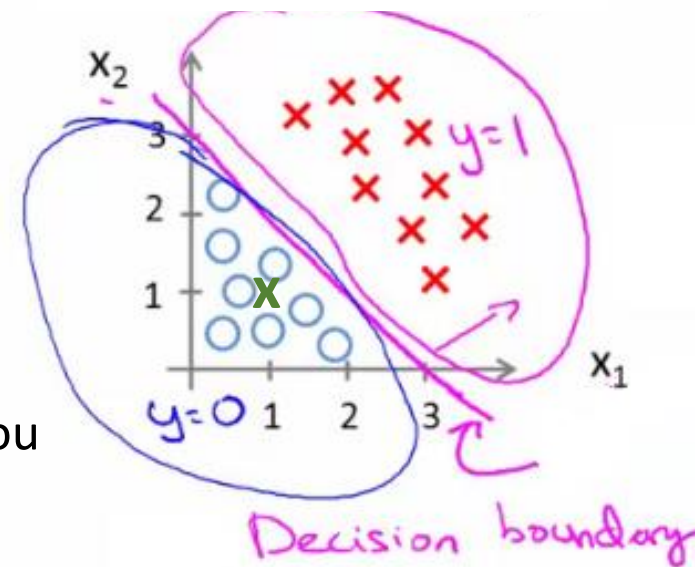


Binární lineární klasifikátor se sigmoidou #2

- Pokud je hodnota na výstupu sigmoidy ≥ 0.5 , přiřadíme x ke třídě 1
- Pokud je hodnota na výstupu sigmoidy < 0.5 , přiřadíme x ke třídě 0
- **Popsaný klasifikátor je binární a pravděpodobnostní**
 - Třída 1 má pro x pravděpodobnost $\sigma(u) = \sigma(\mathbf{w}^T \tilde{\mathbf{x}})$
 - Třída 0 má pro x pravděpodobnost $1 - \sigma(u) = 1 - \sigma(\mathbf{w}^T \tilde{\mathbf{x}})$
- **Umí klasifikovat jen třídy separovatelné lineárně (přímkou)**
 - $\mathbf{w}^T \tilde{\mathbf{x}} \geq 0 \Rightarrow \sigma(\mathbf{w}^T \tilde{\mathbf{x}}) \geq 0.5 \Rightarrow$ bod leží v jedné části příznakového prostoru
 - $\mathbf{w}^T \tilde{\mathbf{x}} < 0 \Rightarrow \sigma(\mathbf{w}^T \tilde{\mathbf{x}}) < 0.5 \Rightarrow$ bod leží ve druhé části příznakového prostoru
- **Klasifikátor je parametrický s vektorem parametrů \mathbf{w}**
 - Během trénování se používá jiné kritérium než pro učení obyčejné regrese !!

Příklad ve 2D

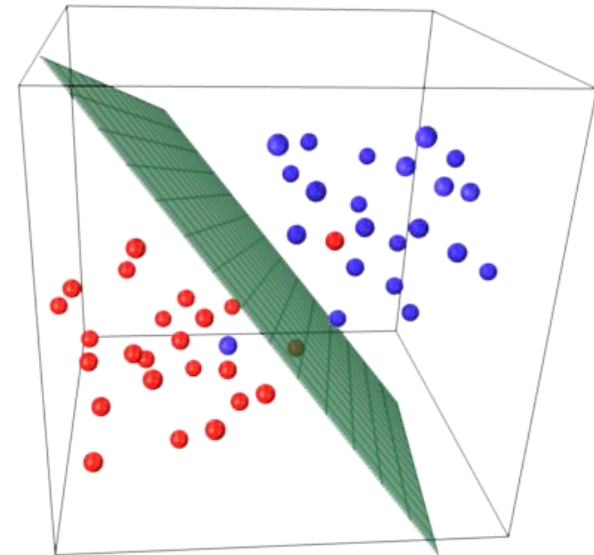
- Klasifikátor pracuje se dvěma vstupy (příznaky)
 - Vstup x_1 reprezentuje váhu, vstup x_2 výšku
- Na základě těchto dvou příznaků se určuje, zda
 - Klasifikovaný objekt spadá do třídy A (křížky) ...oblast nad přímkou
 - Nebo do třídy B (kolečka)... oblast pod přímkou
- Rovnice separující přímky je daná vektorem \mathbf{w}
 - Jeho hodnota $[1; 1; -3]$ byla určena během trénování
- Rozpoznáváme objekt \mathbf{x} , jeho váha = 1 (kg), výška = 1 (m)
- $\tilde{\mathbf{x}} = [\mathbf{x}; x_{n+1}] = [1; 1; 1]$
- Klasifikace:
 - $\mathbf{w}^T \tilde{\mathbf{x}} = [1, 1, -3][1; 1; 1] = 1 + 1 - 3 = -1$
 - $\sigma(-1) = \frac{1}{1+e^{-(-1)}} = 0,27$
 - $0.27 < 0.5 \Rightarrow$ objekt \mathbf{x} patří mezi kolečka s $P=1-0.27=0.73...73\%$



$$\begin{aligned}\mathbf{w} &= [w_1; w_2; w_3 = b] \\ \mathbf{w} &= [1; 1; -3] \\ x_1 + x_2 - 3 &= 0 \\ x_2 &= 3 - x_1\end{aligned}$$

Co když jsou příznaky více než dva?

- Vše funguje principiálně stejně
- Nemluvíme pak ale rovnici přímky, která by třídy separovala, ale o rovnici **rozdělující podplochy**, která má v daném prostoru dimenzi D-1
- Příklad pro 3D:
 - Separující plocha je zde 2D rovina
 - Rovnice 2D roviny má tvar $\mathbf{w}_1\mathbf{x}_1 + \mathbf{w}_2\mathbf{x}_2 + \mathbf{w}_3\mathbf{x}_3 + \mathbf{w}_4 = 0$



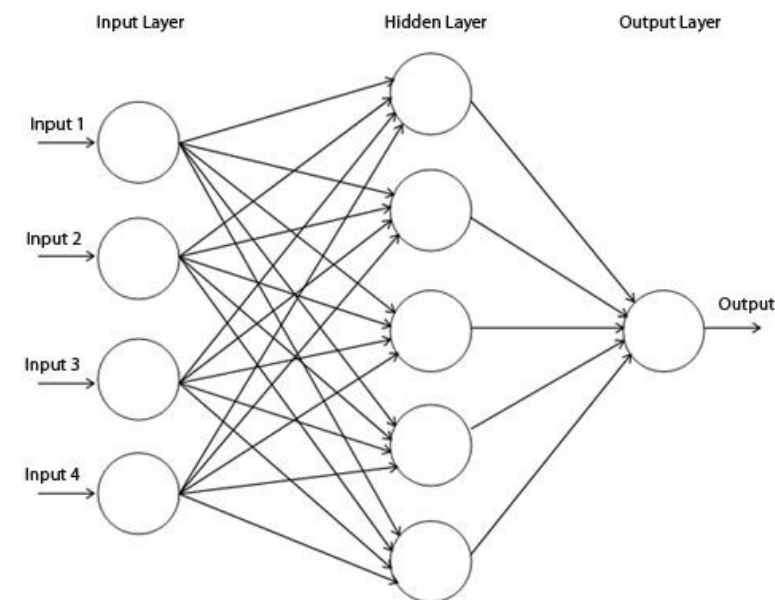
Shrnutí

- Představený klasifikátor se označuje jako perceptron
- Z hlediska úlohy klasifikace objektů do tříd je perceptron
 - BINÁRNÍ, (Proč?)
 - LINEÁRNÍ, (Proč?)
 - PARAMETRICKÝ (Proč?) a
 - PRAVDĚPODOBNOSTNÍ MODEL (Proč?)
- Perceptron zároveň představuje umělý neuron
 - Jde o ekvivalent biologické neuronové buňky
 - Je základním stavebním kamenem umělých neuronových sítí

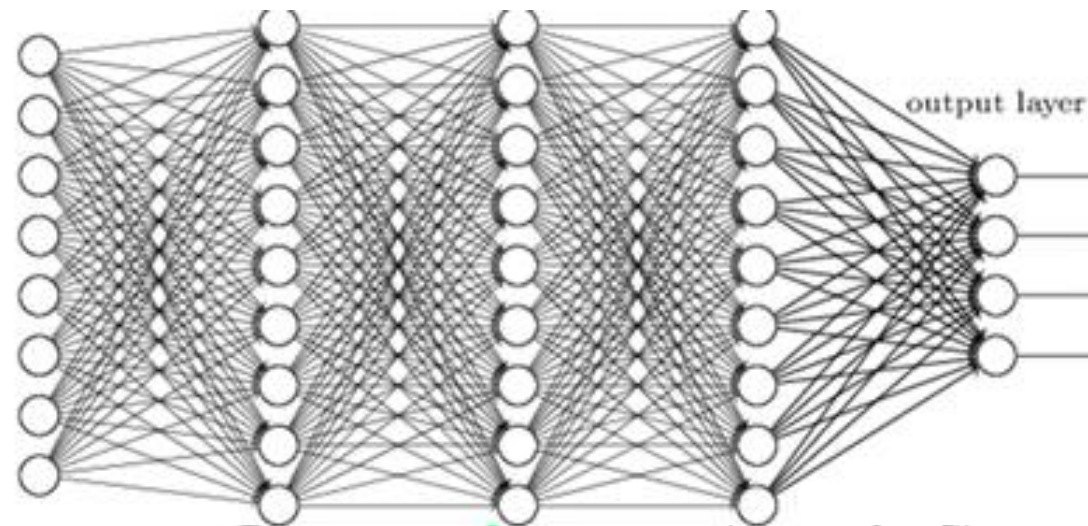
Neuronová síť

Co je to umělá neuronová síť?

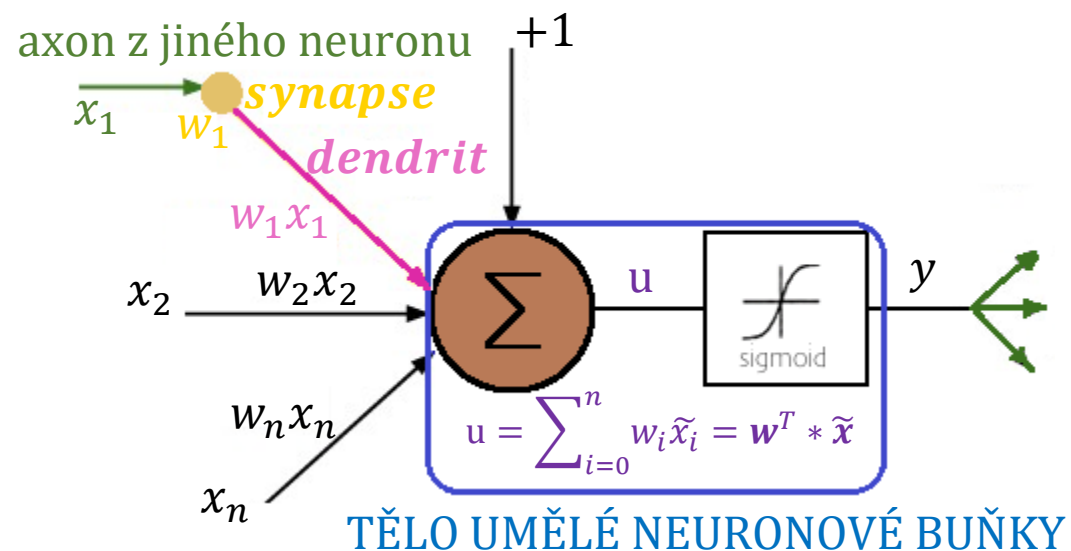
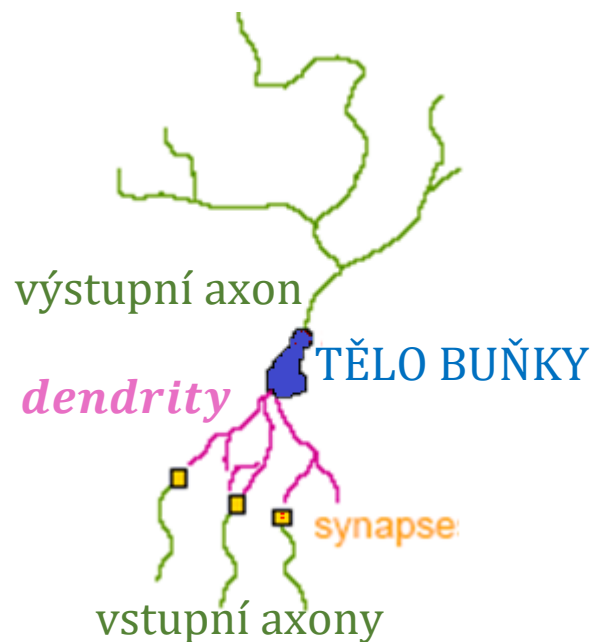
- Matematický model tvořený sériovým a paralelním spojením umělých neuronů
- Podle typů spojení a uspořádání neuronů existuje celá řada typů umělých neuronových sítí
- Následující výklad bude omezen na architekturu označovanou jako **Multi-Layer Perceptron (MLP)**, kde jsou
 - 1) Neurony uspořádány vedle sebe do vrstev a vrstvy jsou spojeny sériově za sebou
 - 2) Výstup z každého neuronu vrstvy i je přiveden na vstup všech neuronů vrstvy $i + 1$
- Inspirací je mozek a biologické neuronové sítě



| Mozek | Umělá neuronová síť typu MLP |
|---|---|
| Skládá se z asi 10^{11} buněk - neuronů | Skládá se z několika až několika desítek tisíc umělých neuronů |
| Neurony jsou různě nepravidelně pospojovány do sítě | Umělé neurony jsou pospojovány paralelně do vrstev, a vrstvy jsou pak spojeny mezi sebou sériově = pravidelná struktura |
| Proces učení spočívá v aktivování a zesilování/zeslabování synapsí (vazeb) mezi neurony | Proces učení spočívá v nastavení a následném neustálém přepočítávání váhových koeficientů w na každém vstupu do každého neuronu |



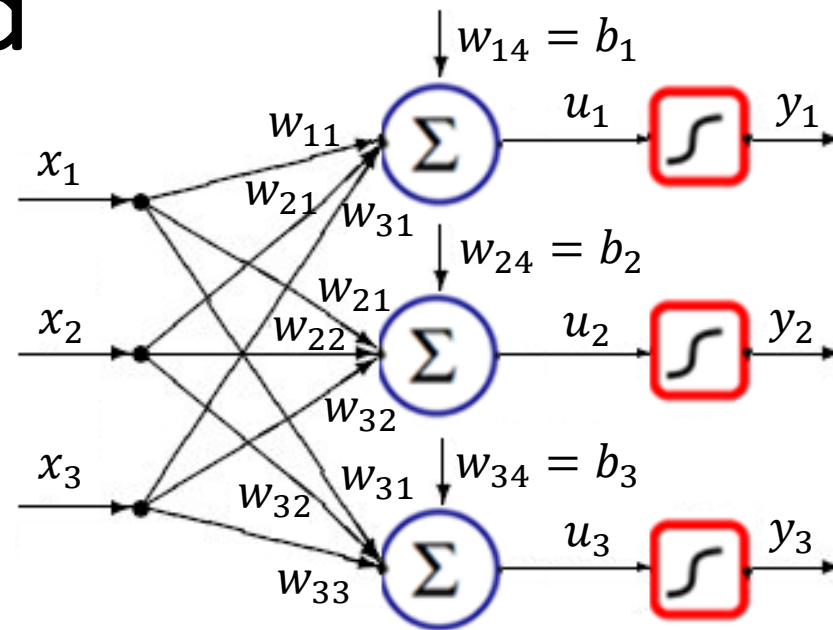
| Neurony | Umělé neurony |
|---|---|
| Spoje mezi neurony tvoří axony | Spoje jsou dány logickou strukturou sítě, kdy výstup z jedné vrstvy/vstupní data je naveden na vstupy do další/první vrstvy |
| Spoj je tvořen synapsí = parametr učení | Parametrem učení je vektor vah w |
| Samotný vstup do buňky tvoří dendrity | Dendrit lze chápat jako hodnotu součinu $w_i x_i$ |
| Podněty jsou v těle neuronu akumulovány | Vstupní signály jsou v modelu sčítány |
| Po překročení určitého prahu je podnět poslán dál | Po překročení prahové hodnoty je výstupní signál změněn podle typu použité aktivační funkce |



Paralelní spojení perceptronů (lineární klasifikace do více tříd)

Lineární klasifikace do více tříd

- Je možná spojením více neuronů paralelně
 - Vznikne jednovrstvá paralelní síť
 - Má tolik výstupů, kolik je tříd
- Síť pak nemá jeden vektor parametrů \mathbf{w} , ale matici parametrů \mathbf{W}
- Příklad:
 - \mathbf{x} má dimenzi 3 (máme tři příznaky)
 - Klasifikujeme shodou okolností také do 3 tříd
 - Jak klasifikace probíhá?
 - Na výstupu ze sítě jsou 3 čísla v intervalu (0,1)
 - **Maximum** z těchto čísel určuje výslednou třídu



$$\mathbf{W}^T = \begin{bmatrix} [w_{11}, w_{12}, w_{13}, w_{14} = b_1] \\ [w_{21}, w_{22}, w_{23}, w_{24} = b_2] \\ [w_{31}, w_{32}, w_{33}, w_{34} = b_3] \end{bmatrix}$$

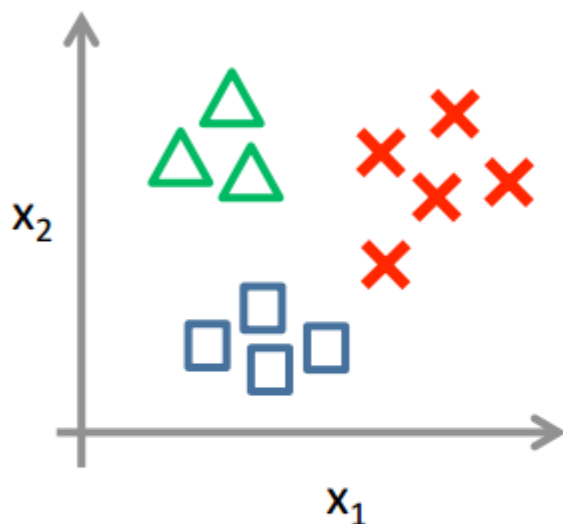
Trénování klasifikátoru na více tříd




- Síť se trénuje postupně jeden klasifikátor za druhým
- Trénuje se tedy vždy **jedna třída proti všem ostatním datům (třídám)**
 - Při trénování klas. pro danou třídu jsou všechna trénovací data náležící ostatním třídám brány dohromady jako jedna „negativní třída“ odpovídající výstupu s hodnotou nula
- Klasifikátor se nazývá 1 vs ALL nebo také 1 vs REST

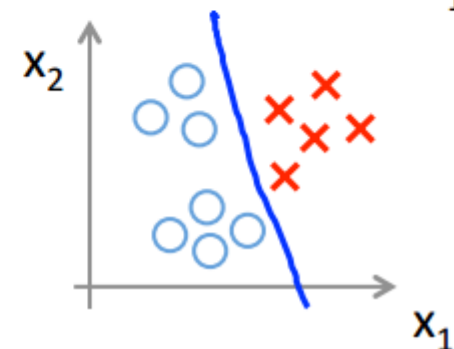
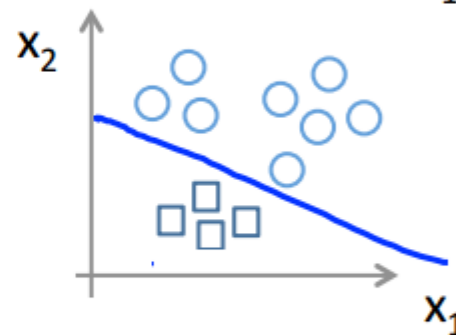
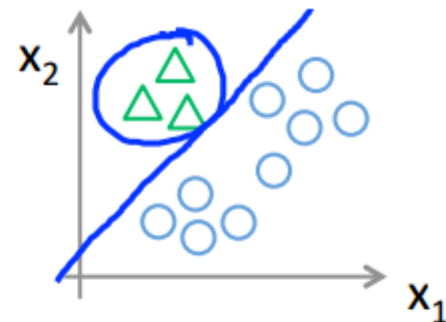
Pozn.: existují i jiné metody, jak trénování provádět a celý model paralelní sítě definovat, ale jejich vysvětlení přesahuje rámec této přednášky

A: klasifikátor typu MAX (1 vs ALL) - trénování

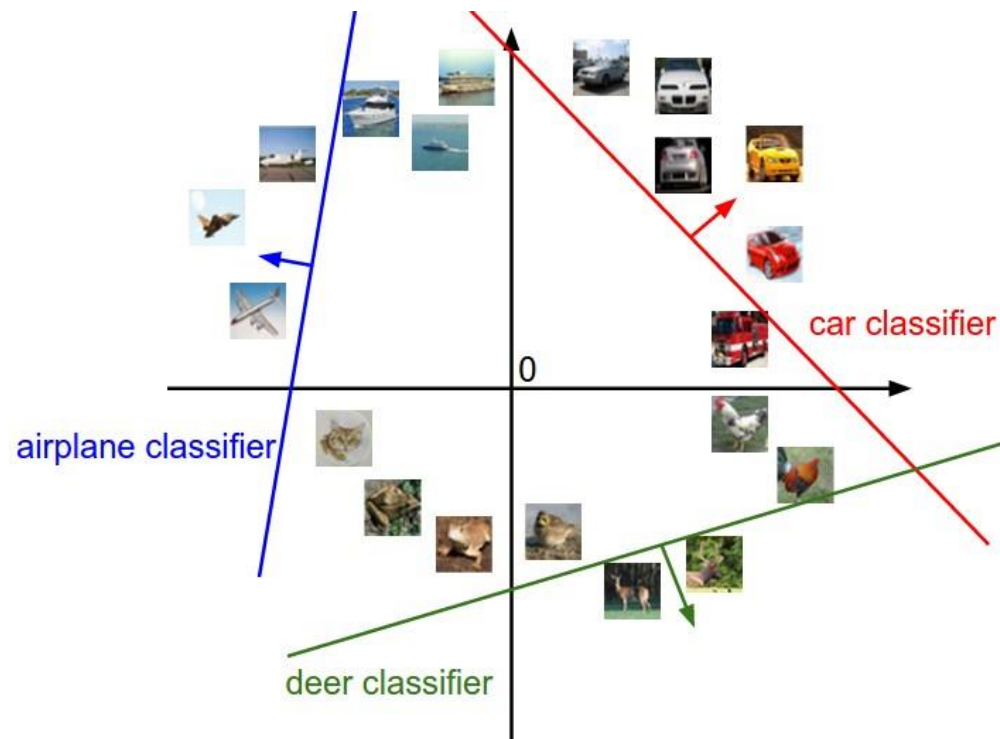
One-vs-all (one-vs-rest):



Class 1: 
Class 2: 
Class 3: 



Ilustrace výsledného klasifikátoru:

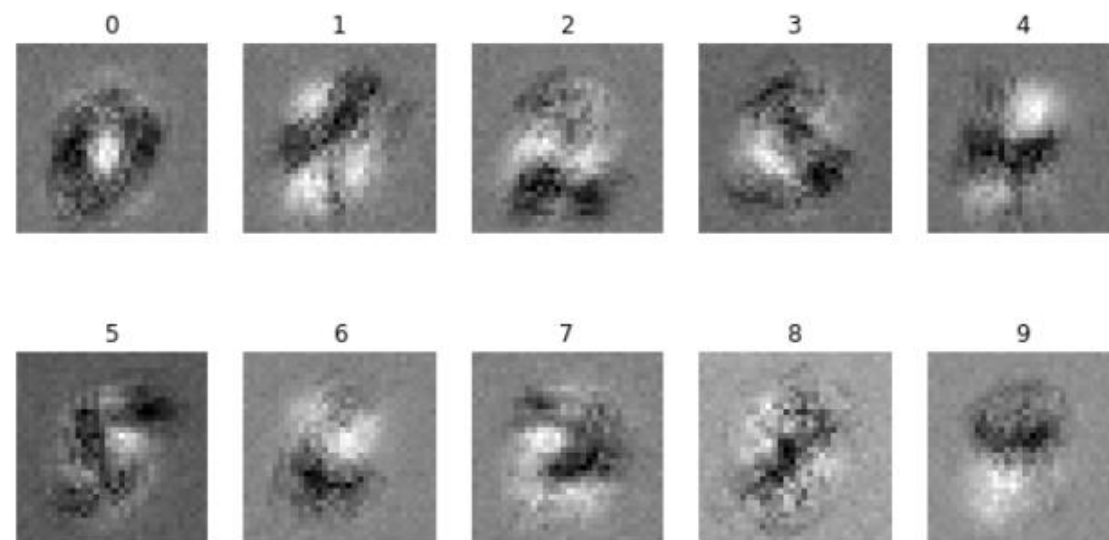


Příklad – úloha rozpoznávání psaných číslovek

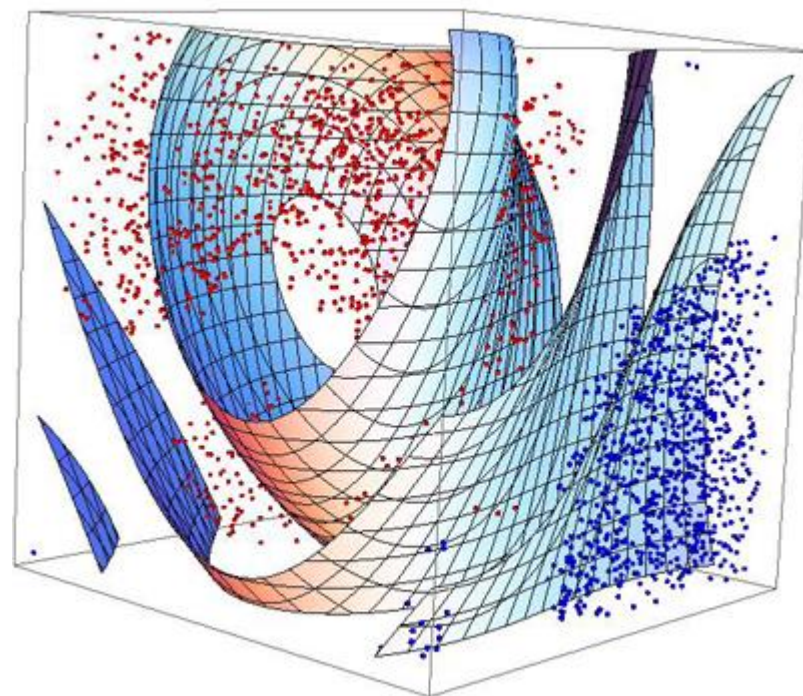
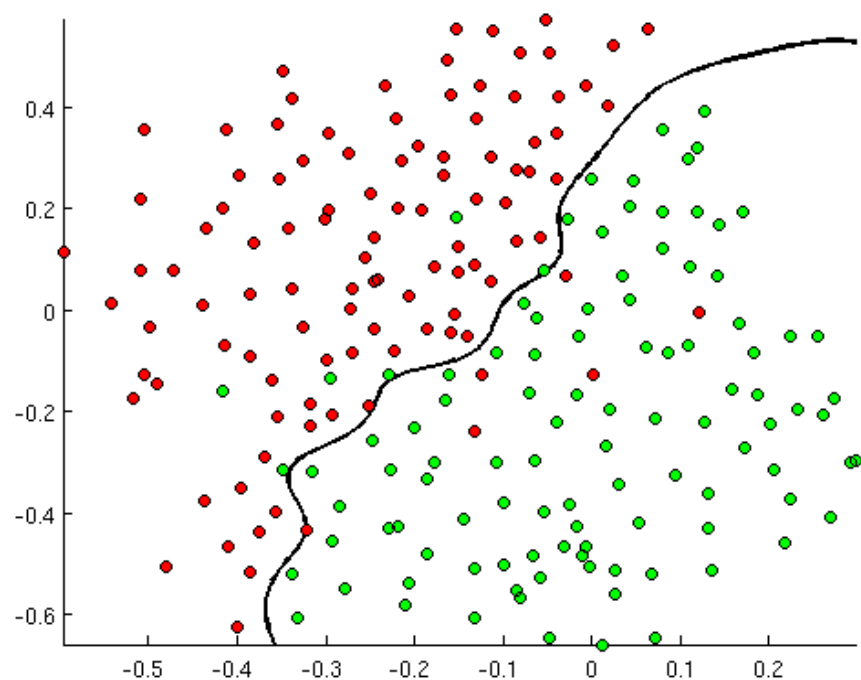
- Rozpoznávaný objekt = digitální obrázek dané číslovky = vektor čísel \mathbf{x}
- Třídy pro klasifikaci = deset kategorií číslovek (0,1,2,3,4,5,6,7,8,9)
- Lineární klasifikátor do více tříd rozpoznávanému objektu \mathbf{x} (obrázku) přiřadí třídu (0...9)
 - Třídy jsou vnitřně reprezentovány modelem s parametry = matice \mathbf{W}
 - Tato matice reprezentuje D-1 dimenzionální separující podroviny v prostoru o D dimenzích
 - Definuje celkem 10 podrovin
 - Je určena ve fázi trénování systému
 - Pomocí modelu (matice \mathbf{W}) se určí pravděpodobnost, že objekt patří do každé jednotlivé třídy (0...9)
 - Např.: $P(0) = 0.3$, $P(1) = 0.05$, ..., $P(9) = 0.1$
 - Třída s největší pravděpodobností (skóre) je vybrána jako výsledek rozpoznávání

Interpretace a význam naučených vah - příklad

- Obrázky znázorňují naučené váhové koeficienty v jednovrstvé síti rozpoznávající psané číslice – viz předchozí příklad.
 - Koeficientů je $10 \times (32 \times 32)$
 - Každý pixel je napojen do každého z 10 neuronů.
- Hodnoty vah jsou znázorněny barvou přiřazené každému pixelu:
 - Černá - velké kladné hodnoty
 - Bílá - velké záporné hodnoty
 - Šedá - něco mezi tím
- Interpretace:
 - Číslice 0
 - Pro rozhodnutí je důležitá přítomnost oválného obrysu, uprostřed obrazu naopak nesmí nic být
 - Číslice 8
 - Je důležitá zejména přítomnost překřížení uprostřed číslovky

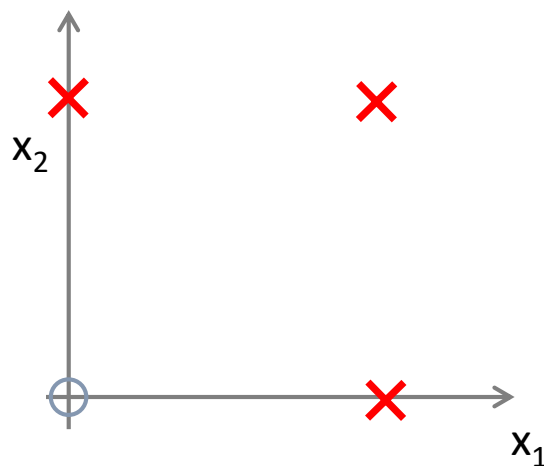


Sériové spojení neuronů (nelineární klasifikace)

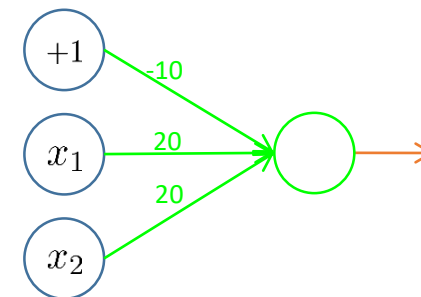


Jednovrstvý perceptron a jeho omezení

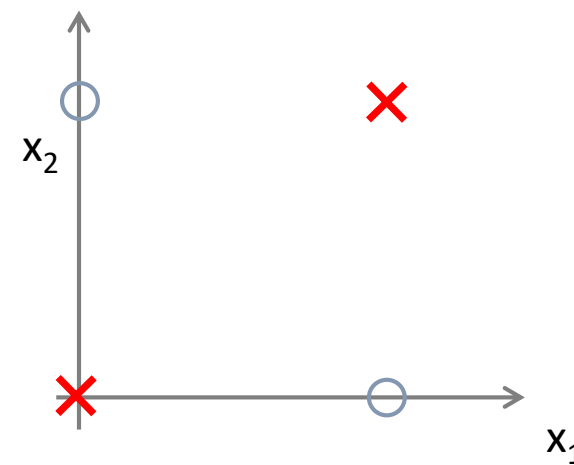
- Co klasifikovat umí?
 - Třídy oddělitelné přímkou



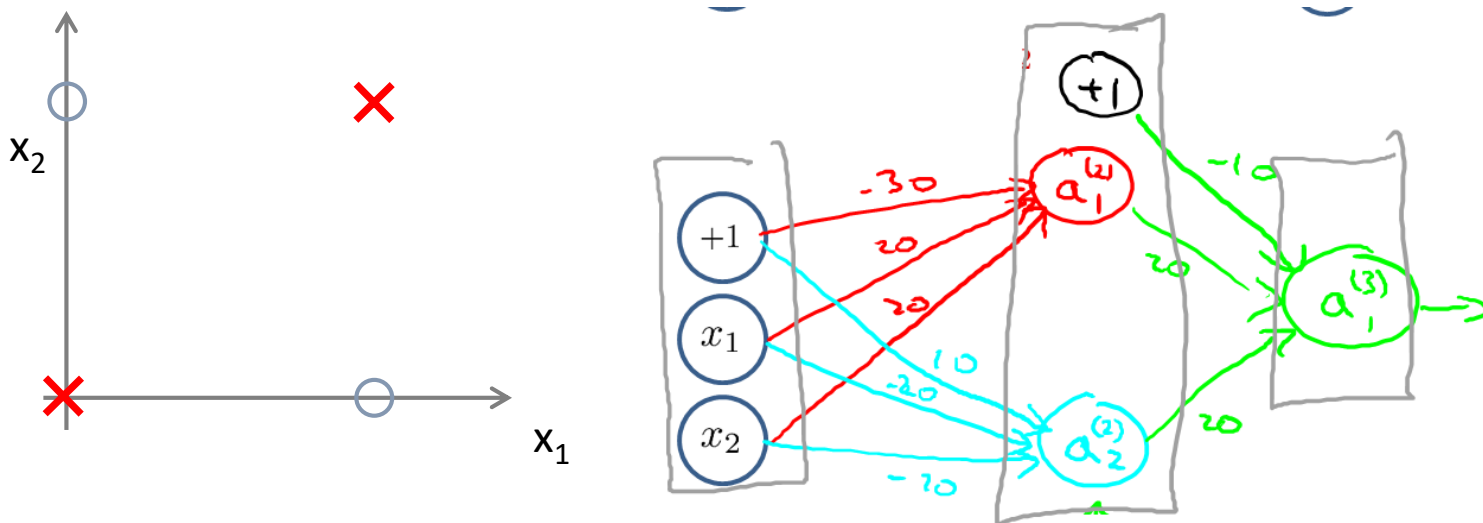
| x_1 | x_2 | y |
|-------|-------|-----|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |



- Co neumí? Nelineárně separovatelné třídy
 - Dokážete najít přímkou oddělující kolečka a křížky ?!?!



Řešení: nelineárního klasif. se skrytou vrstvou

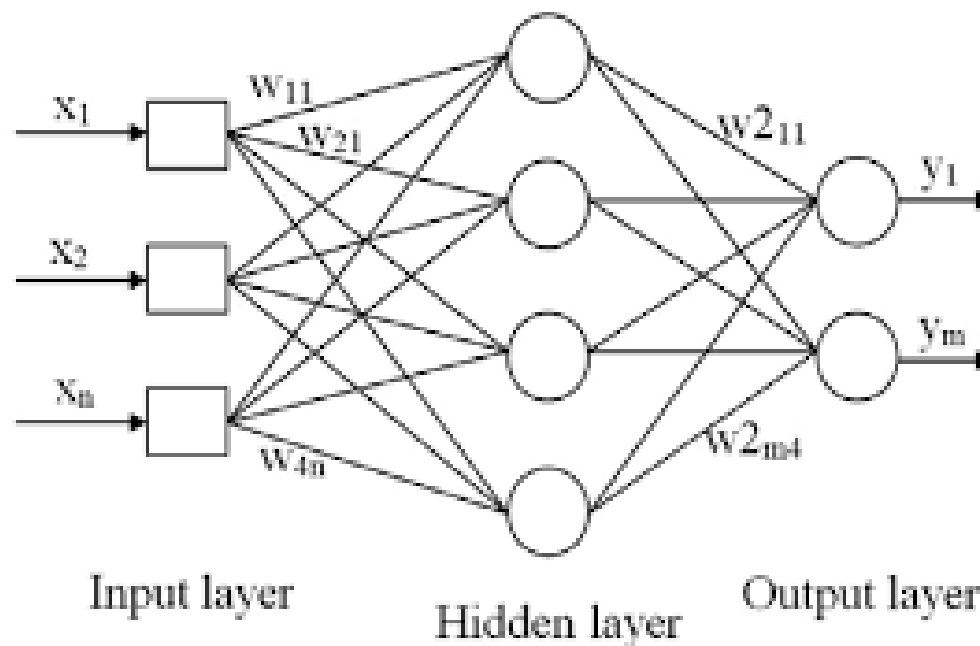


| x_1 | x_2 | $a_1^{(2)}$ | $a_2^{(2)}$ | $y = a_1^{(3)}$ |
|-------|-------|-------------|-------------|-----------------|
| 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 |

- 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 (logický součet)
- *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

Sériové spojení = vícevrstvý perceptron obecně

- Obecně obsahuje vícevrstvý perceptron
 - **Vstupní vrstvu**: nemá neurony, reprezentuje vektor vstupních hodnot
 - Jednu nebo více **skrytých vrstev** s nelineární aktivační funkcí
 - **Výstupní vrstvu**
- Výstupní vrstva a skryté vrstvy mají vždy vlastní matice váhových koeficientů **W**

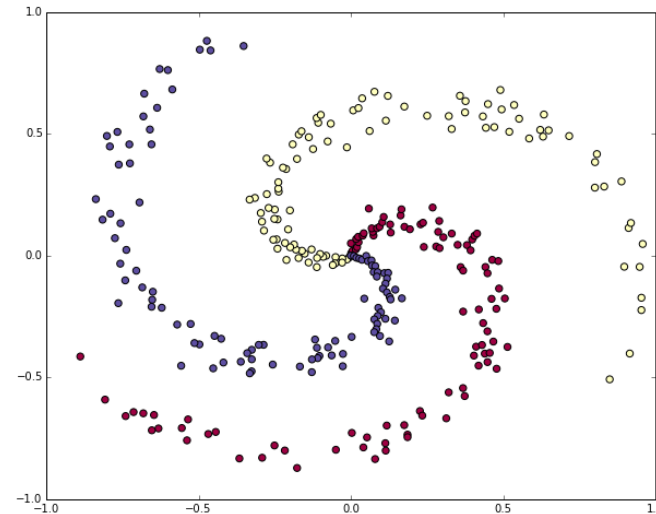


Skryté vrstvy obecně

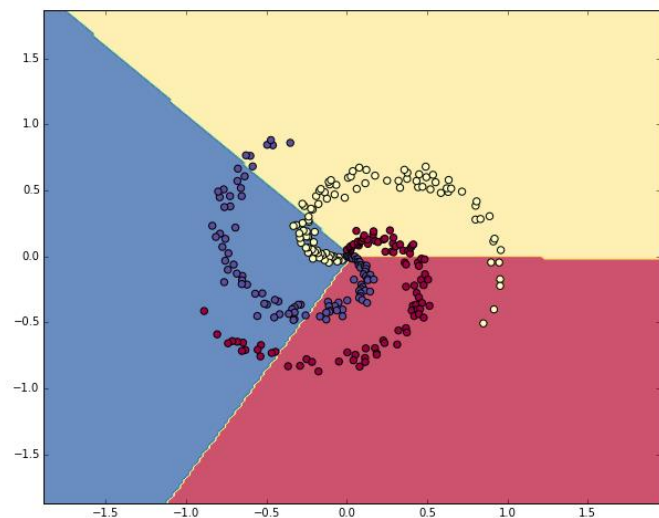
- Pomocí těchto vrstev si síť vytváří sama svoje vlastní příznaky
 - Umožňují nelineární klasifikaci
 - Nejsou vytvořeny člověkem, nějakou matematickou metodou, ale neuronovou sítí samotnou v době trénování
- Většinou je těžké interpretovat význam těchto příznaků
 - Jde to lehce např. u obrázků, viz další přednáška
- Obecně příznaky ve vyšší vrstvě směrem od vstupu reprezentují vyšší úroveň rozhodování (abstrakce) na základě hodnot jednodušších příznaků z předcházející vrstvy
- Sítě s více než 2 skrytými vrstvami se nazývají jako hluboké neuronové sítě (**Deep Neural Networks = DNNs**)

Porovnání výsledků lin. a nelin. klasifikace

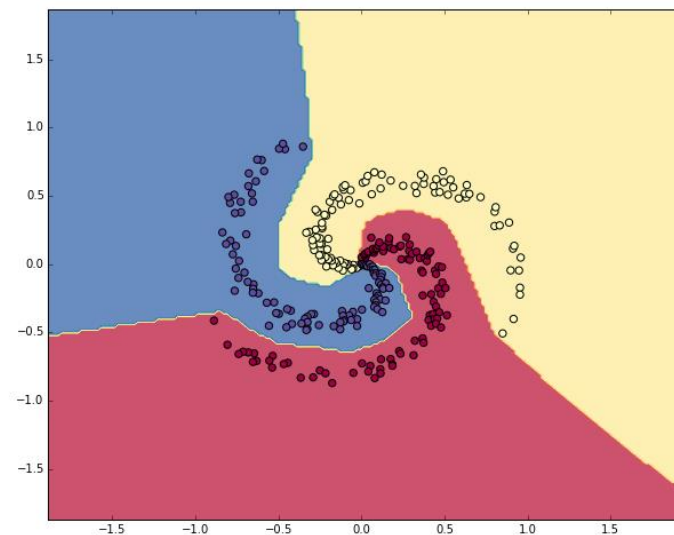
- Data – tři třídy, 2-rozměrný příznakový vektor



- Paralelní lineární klasifikátor



+ jedná skrytá vrstva = nelin. klas.



Závěr: MLP vs NN klasifikátor

| | MLP | NN klasifikátor |
|---|--|--|
| Co se vypočítává pro rozpoznávaný objekt? | Pravděpodobnosti, že objekt patří do jednotlivých tříd | Vzdálenost objektu od všech objektů v trénovací množině |
| Kolik výpočtů se provádí během rozpoznávání? | Jeden výpočet s celým modelem | Tolik, kolik objektů je v trénovací množině !! |
| Jak se určí výsledná třída? | Vybere se ta, pro kterou je vypočtená pravděpodobnost nejvyšší | Je daná třídou, do které patří trénovací objekt, který má nejmenší vzdálenost k rozpoznávanému objektu |
| Pracuje se vnitřně s nějakým modelem? | Ano, klasifikátor má vnitřní model s parametry (vektory či matice vah) | Ne, klasifikátor vnitřně pracuje se všemi trénovacími daty |
| Je třeba klasifikátor natrénovat? | Ano, je třeba určit parametry modelu | Ne, nic se netrénuje, při klasifikaci se pracuje přímo s trénovacími daty |
| Je možné klasifikovat i třídy separovatelné pouze nelineárně? | Ano, ale perceptrony musí být uspořádány i sériově | Ano, separující podplocha je nelineární a je určena rozložením všech trénovacích bodů v příznakovém prostoru |

Bonusová úloha

- Dostupná na e-learningu
- Multiclass_perceptron.m
 - 3 různě efektivní implementace klasif. tohoto typu (skóre vždy 49 %)
 - Každá využívá různě efektivní implementaci funkce sigmoida (nutno také vytvořit)
 - Na začátku programu se využívá funkce data_preprocessing.m (nutno také vytvořit)
- Data_preprocessing_fast.m
 - Výpočetně efektivní implementace data_preprocessing.m bez cyklů for
- DNN_128_128_10.m - program pro rozpoznávání ručně psaných číslic
 - Implementace MLP se 2 skrytými vrstvami bez cyklů for
 - Využívá aktivační funkci typu ReLU místo sigmoidy a natrénovaný model Image_DNN_128_128_10.mat (128 neuronů v 1. a 2. vrstvě)
 - Aplikujte na základní testovací set (skóre 75,9 %) a nezávislý testovací set z minulé úlohy
 - Následně otestujte model Image_centered_DNN_128_128_10.mat natrénovaný na vycentrovaných obrázcích
 - sami si vycentrujte oba testovací sety a porovnejte výsledky
- Odevzdání všech těchto programů = 1 bod + splnění cvičení

Extra - bonusová úloha

- Aplikujte odladěný program také na úlohu rozpoznávání mluvených číslic.
- Projděte si program a upravte ty části (je jich minimum), které se musí změnit, aby síť rozpoznávala slova zparametrizovaná do matice 64x64.
- Pro experiment použijte data z minulé bonusové úlohy
 - a to jak (SD – Speaker Dependent), tak i SI (Speaker Independent).
- Natrénováali jsme pro Vás síť se 128, 128 a 10 neurony, čili ve stejné struktuře jako pro psané číslice.
 - Váhy sítě jsou uloženy v souboru Spoken_DNN_128_128_10.mat, který si můžete stáhnout.
- Otestujte opět na obou testovacích setech (SD a SI) a zjistěte výsledek. Porovnejte dosažené výsledky s metodou NN a do mailu při odevzdávání uveďte oba výsledky.

Jak bonusové úlohy a cvičení řešit?

- Kód nelze spustit ... je na různých místech nedopsaný !
 - Řešení může být kód postupně kopírovat do vlastního m-filu
- V různých místech obsahuje pro kontrolu zakomentované mezivýsledky
- Nezapomenout na začátku programu vždy vymazat paměť !
- Může se hodit
 - Funkce whos
 - Vypíše obsah paměti a dimenze proměnných
 - Tužka a papír
 - Pro hledání logiky, jak výpočty správně zapsat maticově

Aktivační funkce ReLU

- ReLU je správně označení neuronu, který využívá aktivační funkci r definovanou jako:
$$r(x) = \text{rectifier}(x) = \max(0, x)$$
- Často se ale termín ReLU používá i pro samotnou aktivační funkci $r(x)$ jako takovou
- ReLU je zkratka Rectified Linear Unit
 - jde o typ umělého neuronu
 - představen v roce 2000
 - v současné době nejpoužívanější typ
 - DNN s těmi neurony dosahují nejlepších výsledků

