Cvičení: Perceptron

1. Implementujeme si učící algoritmy pro perceptron (Hebbovo učení, Rosenblattovo učení - vybrané varianty)
2. Příklad 1: učení jednoducých logických funkcí
   - Zobrazíme si data a dělící nadrovinu ve 2D a ve 3D.
3. Příklad 2: Náhodně vygenerovaná data.
4. Příklad 3: Data s odlehlými vzory.
5. Příklad 4: Písmena.
6. Příklad 5: Ručně psané číslice.

Vytvářené funkce:

- Odezva sítě na vzor

```
function [y, xi] = response(x, w)
```

- Odezva sítě na matici vzorů

```
function [y, xi] = responseM(X, w)
```

- Chyba klasifikace

```
function [e, y] = classification_error(X, w, d)
```

- Hebbovo učení

```
function w = training_hebb (X, d)
```

- Rosenblattovo učení

```
function [w, epochs] = training_rosenblatt (X, d, ...
    alpha, max_epochs)
```

Vytvářené funkce:

Případné další varianty Rosenblattova algoritmu:

- Rosenblattovo učení s přihrádkou

```
function [w, epochs] = training_rosenblatt_best (X, d, alpha, max_epochs)
```

- Dávkový algoritmus učení

```
function [w, epochs] = training_rosenblatt_batch (X, d, alpha, max_epochs)
```

Příklad 1: učení jednoducých logických funkcí

1a) Příklad z přednášky

x_0	x_1	x_2	d
+1	-1	-1	+1
+1	-1	+1	+1
+1	+1	-1	+1
+1	+1	+1	-1

Zobrazení dat a dělící nadroviny ve 2D

```
function [hf] = plot_decision_boundary_2d (X, d, w)
```

Příklad 1: učení jednoducých logických funkcí

1b) Žlučník

vlašák	bůček	léky	bude mi zle?
+1	-1	-1	+1
+1	-1	+1	-1
+1	+1	-1	+1
-1	+1	+1	-1
+1	+1	-1	+1
+1	+1	+1	+1

Zobrazení dat a dělící nadroviny ve 3D

```
function [hf] = plot_decision_boundary_3d (X, d, w)
```

Příklad 1: učení jednoducých logických funkcí

1c) Hospoda

Pavel	Pepa	Honza	jde se na pivo?
+1	-1	-1	-1
+1	-1	+1	+1
-1	+1	-1	-1
-1	+1	+1	+1
+1	+1	-1	+1
+1	+1	+1	+1
-1	-1	-1	-1
-1	-1	+1	-1

Příklad 2: Náhodně generovaná data ve 2D

- Nejprve zkusíme vygenerovat trénovací množinu náhodně.
- Na datech naučíme perceptron, podíváme se na chybu a zobrazíme si data a dělící nadrovinu.

```
% rand randn randi rng(0)
```

- Potom vygenerujeme data obsahující dva shluky vzorů (využijeme připravený skript)
- Na datech naučíme perceptron, podíváme se na chybu a zobrazíme si data a dělící nadrovinu.)

```
% vygenerujeme data
p = [100,1,1,2,1;100,-2,1,0,2];
v = generate_random_clusters_2d(p);
...
% zobrazime vysledek
 plot_decision_boundary_2d (X, d, w)
```

Příklad 3: Data s odlehlými vzory

- Máme data, kde vstupní vektory mají různou délku (zde je odlehlý 3. vzor)
- Na datech naučíme perceptron, podíváme se na chybu a zobrazíme si data a dělící nadrovinu.)

```
X = [1  -0.5  -0.5;
  1  +0.3  -0.5;
  1  -40  +50;
  1  -0.5  +0.5;
  1  -0.1  +1.0]
d = [1;  1;  -1;  -1;  1];
```

- Jak dlouho se bude perceptron učit?
- Jak dlouho se bude perceptron učit, pokud normalizujeme všehny vstupní vektory na délku 1?

```
X1 = normr(X)
```

Příklad 4: Písmena

- Využijeme připravenou datovou sadu **pismena.mat**
- Písmena byla segmentována z **pismena.png**
- Prohlédněte si datovou sadu a zobrazte si některá písmenka (využijte předpřipravený skript)
- Naučte perceptron pomocí různých algoritmů (a variant) rozpoznávat jednotlivá písmena.
  (bude třeba vyrobit trénovací množiny)
- Určete chybu klasifikace na trénovací množině (popř. i počet epoch / čas učení)
- Určete chybu klasifikace na testovací množině (s přidaným šumem) (viz předpřipravený skript)
- Podívejte se, se kterými písmenky měl perceptron největší problémy.
- Soutěž o nejlepší učící algoritmus :-)

Příklad 5: Ručně psané číslice

- Využijeme připravenou datovou sadu **OcrData.mat** s ručně psanými číslicemi
- Prohlédněte si datovou sadu a zobrazte si některé číslice (využijte předpřipravený skript)
- Naučte perceptron pomocí různých algoritmů (a variant) rozpoznávat jednotlivé číslice.
- Určete (a porovnejte) chybu klasifikace na trénovací množině (popř. i počet epoch / čas učení)
- Podívejte se, se kterými číslicemi měl perceptron největší problémy