# Sztuczne sieci neuronowe

dr inż. Piotr Szczuko

# Uczenie maszynowe

- Wnioskowanie na podstawie zależności wydobytych z analizy bardzo dużych zbiorów danych
- Automatyczne odkrywanie zależności z przykładów uczących
- Zastosowanie wcześniejszej wiedzy do nowych przypadków

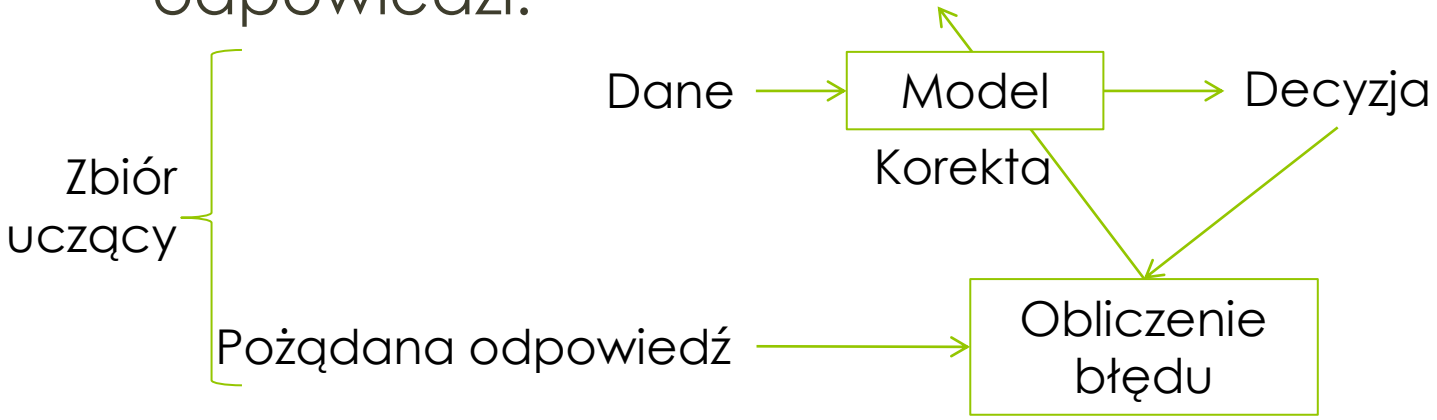
# Kategorie uczenia maszynowego

- Nadzorowane – dla wszystkich danych znany jest pożądany wynik
- Nienadzorowane – wynik nie zawsze jest znany

- Klasyfikacja – wynik to „etykieta”, „kategoria”, „klasa”
- Regresja – wynik to wartość ciągła

# Model nadzorowany

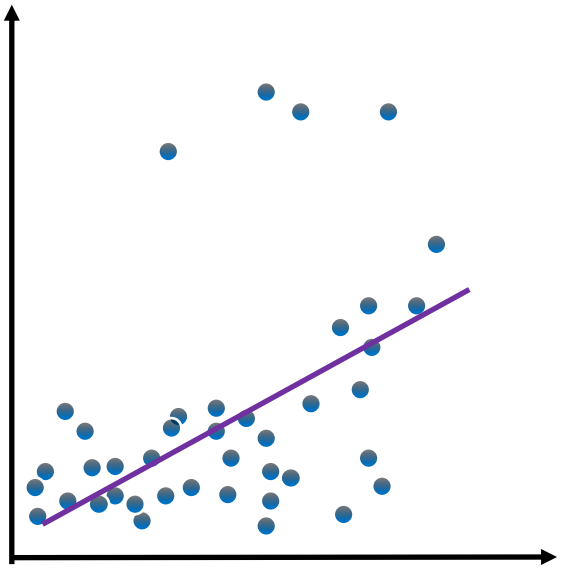
- Trening w oparciu o znane dane i odpowiedzi:




- Predykcja:

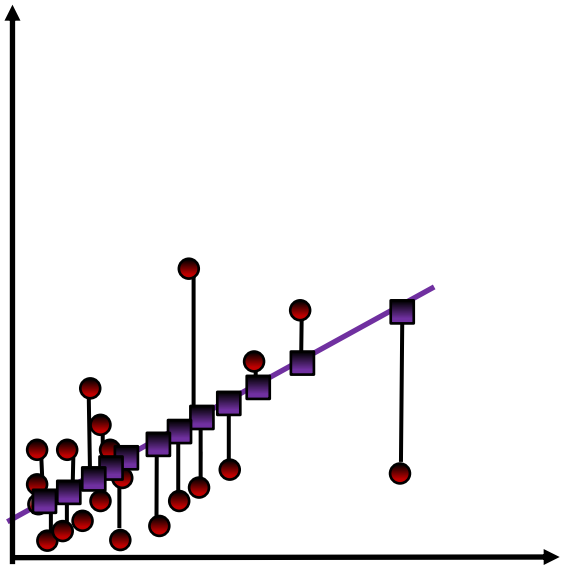
# Ocena trafności modelu

**Dane treningowe**



Dopasowanie modelu

**Dane testowe**



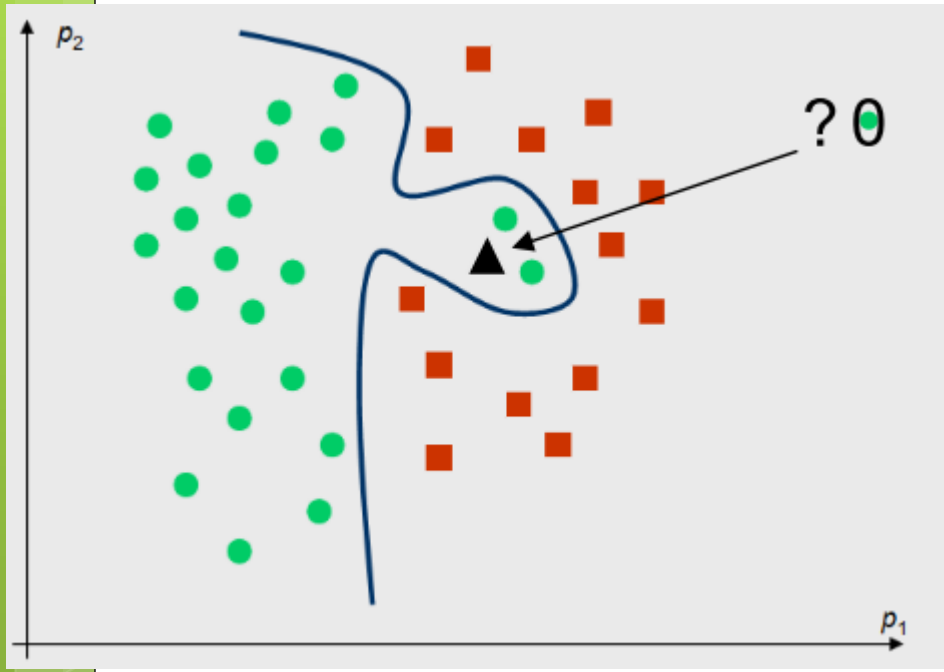
Predykcja i pomiar błędów

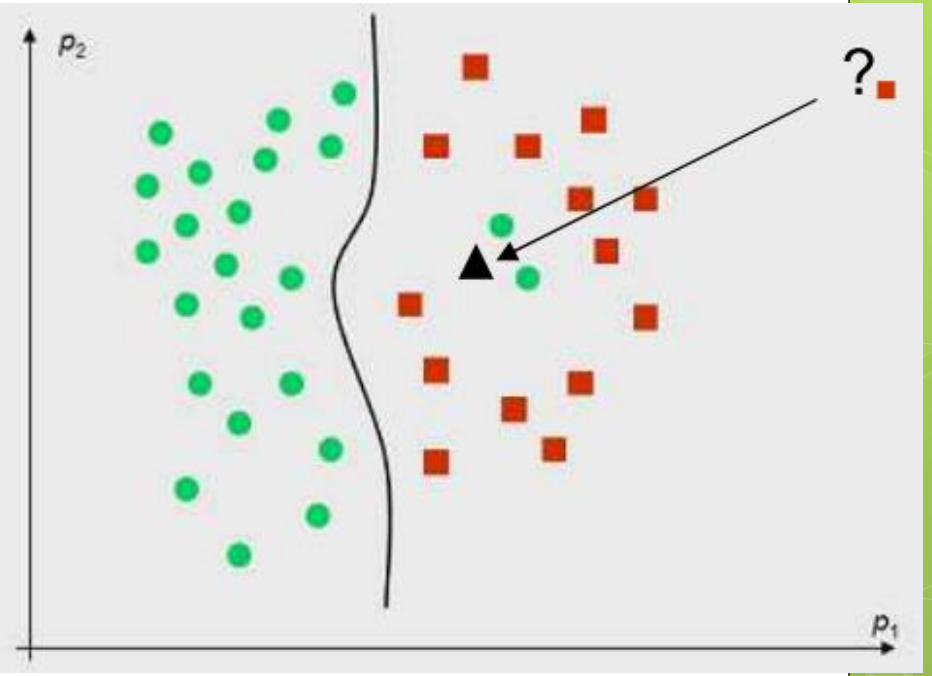
# Błąd – np. średniokwadratowy

- Niezależny od znaku
- Uwzględnia wszystkie punkty

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

# Zdolność generalizacji

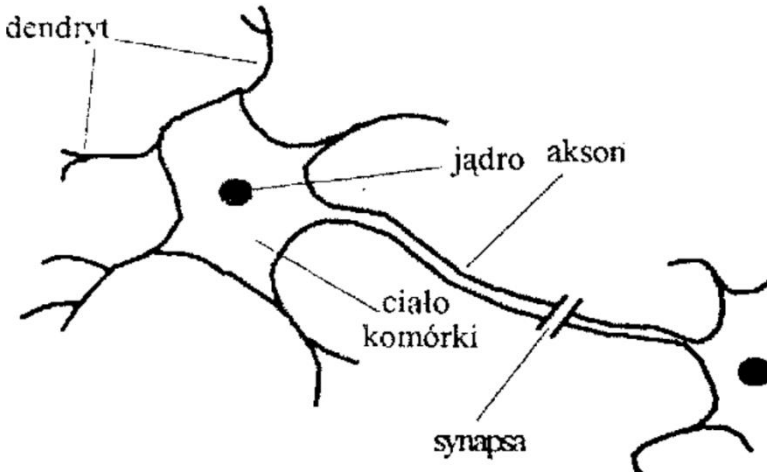



- Przetrenowanie – *overfitting*
- Niedotrenowanie - *underfitting*

# Model neuronu

- Inspiracja komórką nerwową



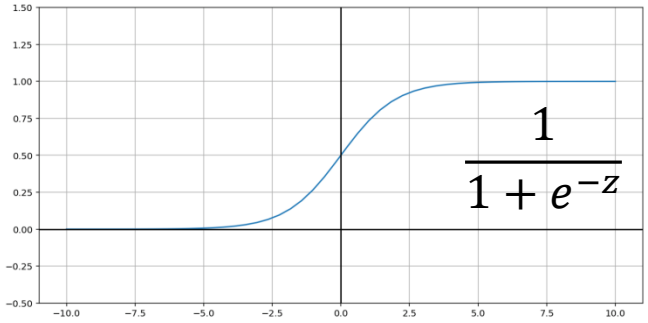

Pierwszy matematyczny opis komórki nerwowej:
McCulloch W. S. , Pitts W. (1943) *A logical calculus of the ideas immanent in nervous activity*. Bulletin of Mathematical Biophysics 5, pp. 115-133.

# Model neuronu

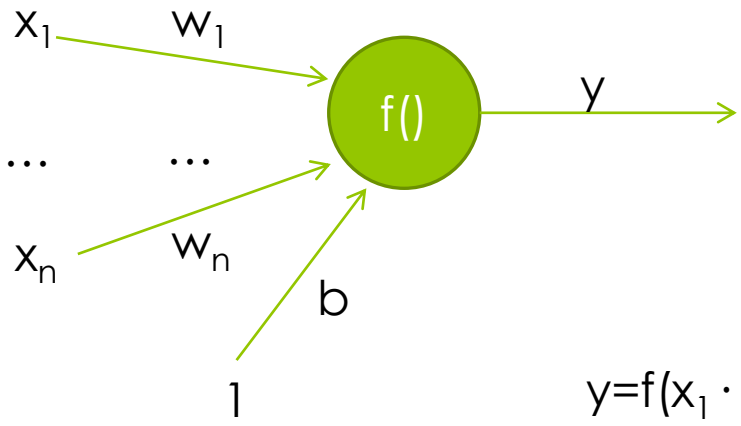
- Wejścia – suma ważona: $\mathbf{x}^T \cdot \mathbf{w}$
- Przetwarzanie – normalizacja sumy za pomocą **funkcji aktywacji**: $f(\mathbf{x}^T \cdot \mathbf{w})$
- Wyjście: $y = f(\mathbf{x}^T \cdot \mathbf{w})$

- Neuron reaguje nawet na jedną niezerową wartość na wejściu
- Na wyjściu wartości z ustalonej dziedziny

# Perceptron




- Pojedynczy neuron




$y=f(x_1 \cdot w_1+\ldots+x_n \cdot w_n+b)$
$=f(b + \Sigma x_i w_i)$

$\mathbf{x}=[x_1 \ldots x_n, 1]$

$\mathbf{w}=[w_1 \ldots w_n, b]^T$

„*bias*" Wartość progowa (zwykle ujemna, ustala kiedy argument funkcji jest większy od zera)

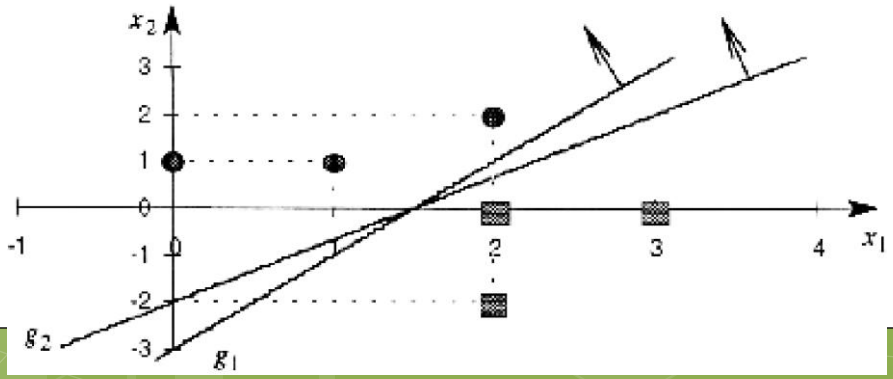
# Funkcje aktywacji

- Ciągła
- Ograniczona dziedzina wartości
- Łatwa do obliczenia i ciągła pochodna
- Ustalanie kształtu za pomocą parametru

| funkcja | Wzór funkcji | Wzór pochodnej |
|---|---|---|
| Sigmoida | $f(x) = \frac{1}{1+e^{-(beta*x)}}$ | $f'(x) = beta*(1-f(x))*f(x)$ |
| Tangens hiperboliczny | $f(x) = \tanh(beta*x)$ | $f'(x) = beta(1-f^2(x))$ |
| Sinusoida | $f(x) = \sin(beta*x)$ | $f'(x) = beta\sqrt{1-f^2(x)}$ |
| Cosinusoida | $f(x) = \cos(beta*x)$ | $f'(x) = -beta\sqrt{1-f^2(x)}$ |

# Perceptron – klasyfikator

- Klasyfikacja n-wymiarowego obiektu do jednej z dwóch klas
- Wyjście y≥0 -> klasyfikacja do klasy 1.
- Wyjście y<0 -> klasyfikacja do klasy 2.
- $y_1 = -2 \cdot x_1 + x_2 + 3$ (dla y=0: $x_2 = 2 \cdot x_1 - 3$)
- $y_2 = -4 \cdot x_1 + 3 \cdot x_2 + 6$ (dla y=0: $x_2 = 4/3 \cdot x_1 - 2$)



# Przykład

- https://playground.tensorflow.org
- (klasyfikacja skupisk gaussowskich)
- (funkcje aktywacji)

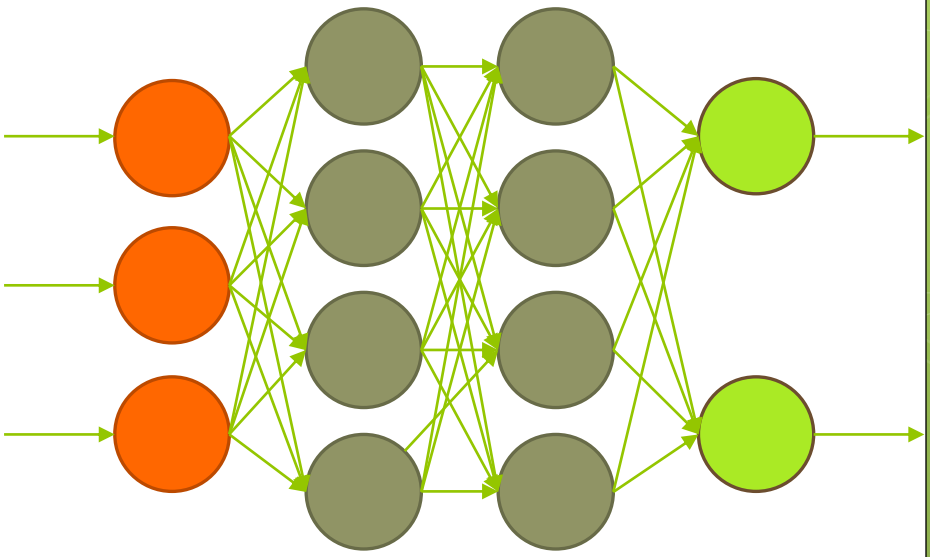
# Jeden neuron – sieć neuronów?

- Jeden neuron – liniowa granica między klasami
- Warstwy neuronów – wyliczanie cech/atrybutów z danych *(feature)* przydatnych dla kolejnych neuronów

Warstwy:
- Wejściowa
- N ukrytych
- Wyjściowa

Macierze wag np. 4x2

Funkcje aktywacji każdego neuronu



# Przykład

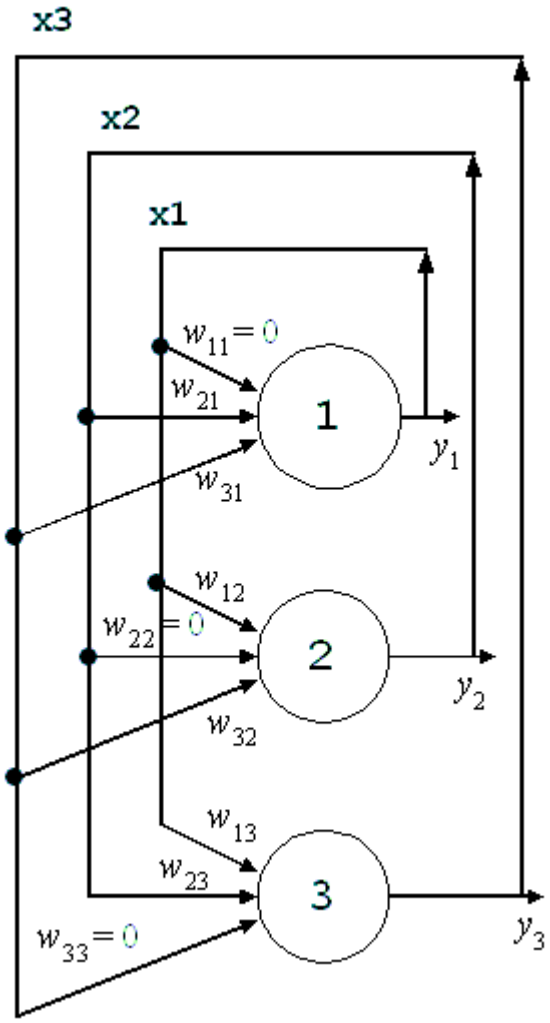
- https://playground.tensorflow.org
- (klasyfikacja XOR)
- (minimum lokalne)

# Architektury sieci

- Jednokierunkowe
- Ze sprzężeniem zwrotnym (proste wyjście na wejście)
- Komórkowe (złożone połączenia dwukierunkowe miedzy sąsiednimi neuronami)
- Z pamięcią (Long Short-Term Memory): w sieci wielowarstwowej zamiast neuronów „bramki”:
- Głębokie (wykład 2)

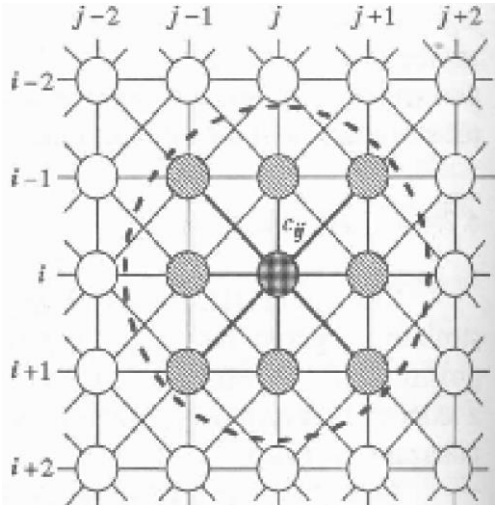
# Sieć Hopfielda

- Sprzężenie zwrotne
- Przetwarzanie szeregów czasowych



# Sieć komórkowa

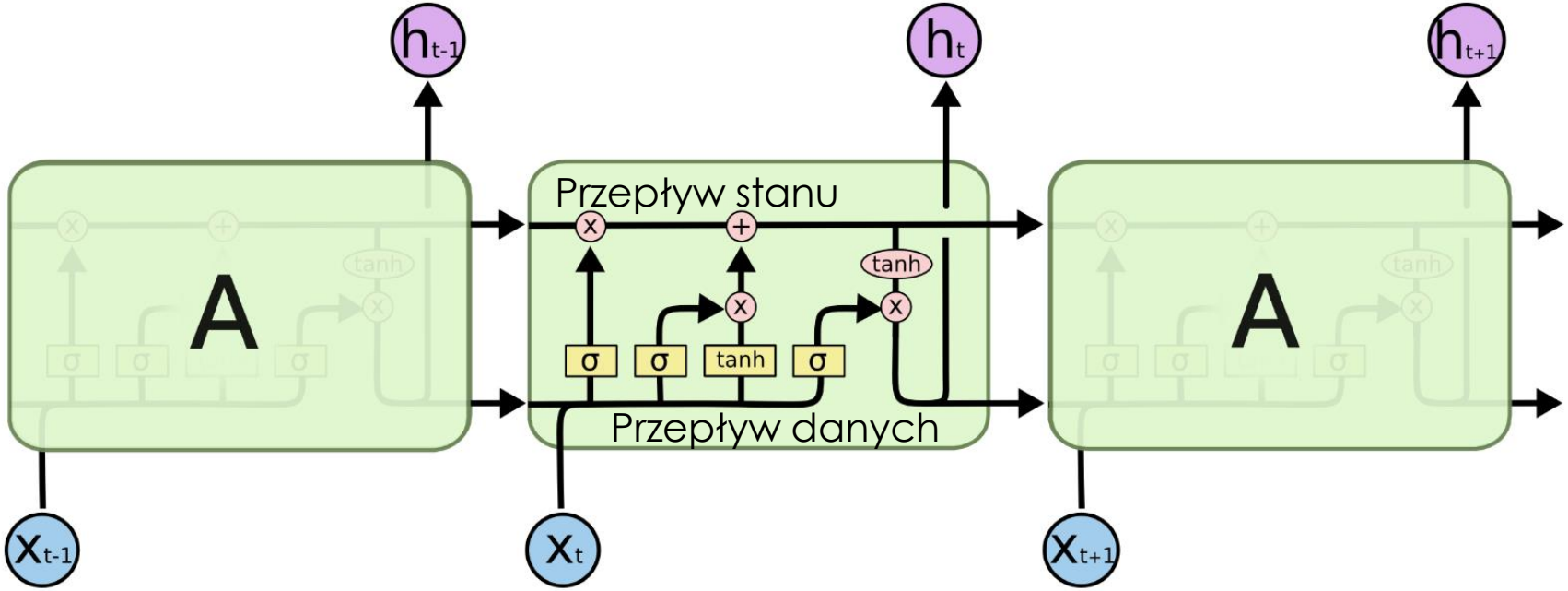
- Np. na regularnej siatce prostokątnej
- Z sąsiedztwem 1 do 8




- Filtracja obrazów

# Sieci z pamięcią krótkotrwałą

- Neuron σ – bramkowanie [0,1]
- Neuron tanh - skalowanie




Źródło: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

# Zastosowania sieci

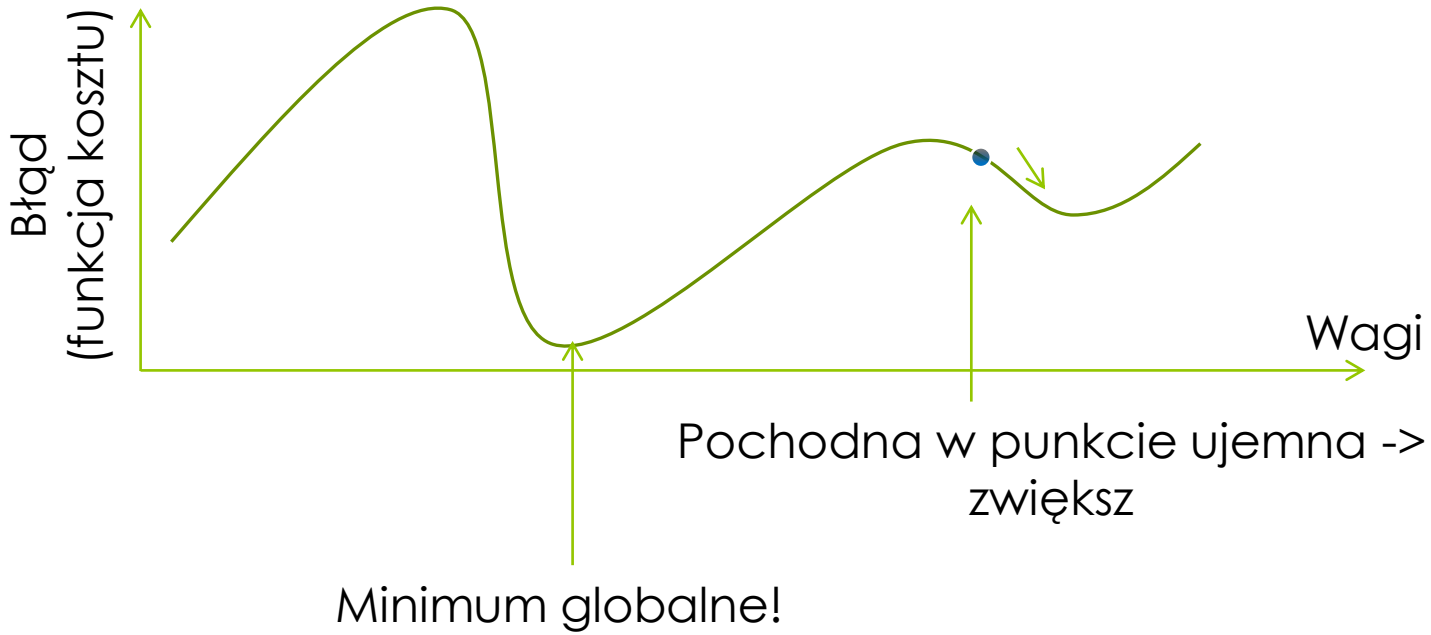
- Rozpoznawanie i klasyfikowanie na podstawie wcześniejszych wzorców
- Kompresja danych (wyznaczanie jednoznacznej prostszej reprezentacji)
- Predykcja wartości, sterowanie
- Filtracja rekonstrukcja sygnałów (sieci asocjacyjne)

# Trening sieci

- Problem: big data
- Czas
- Pamięć

# Metody korekty wag

- Iteracyjne poprawki w stronę malejącego błędu



# Funkcja kosztu

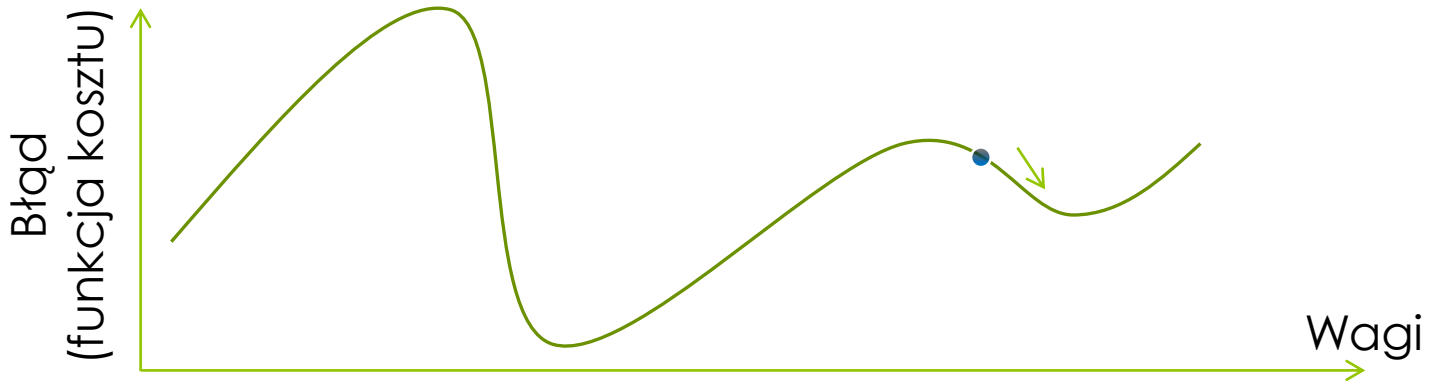
- $J(y, f(\mathbf{x}^T\mathbf{w}))$ zależy od:
  - konkretnego przykładu **x**
  - uzyskanego wyjścia $f(\mathbf{x}^T\mathbf{w})$ (**od wag**)
  - pożądanego wyjścia y
- Pochodne cząstkowe po wagach w każdej warstwie:
  - $\partial J / \partial \mathbf{W}_n$

# Wsteczna propagacja błędu

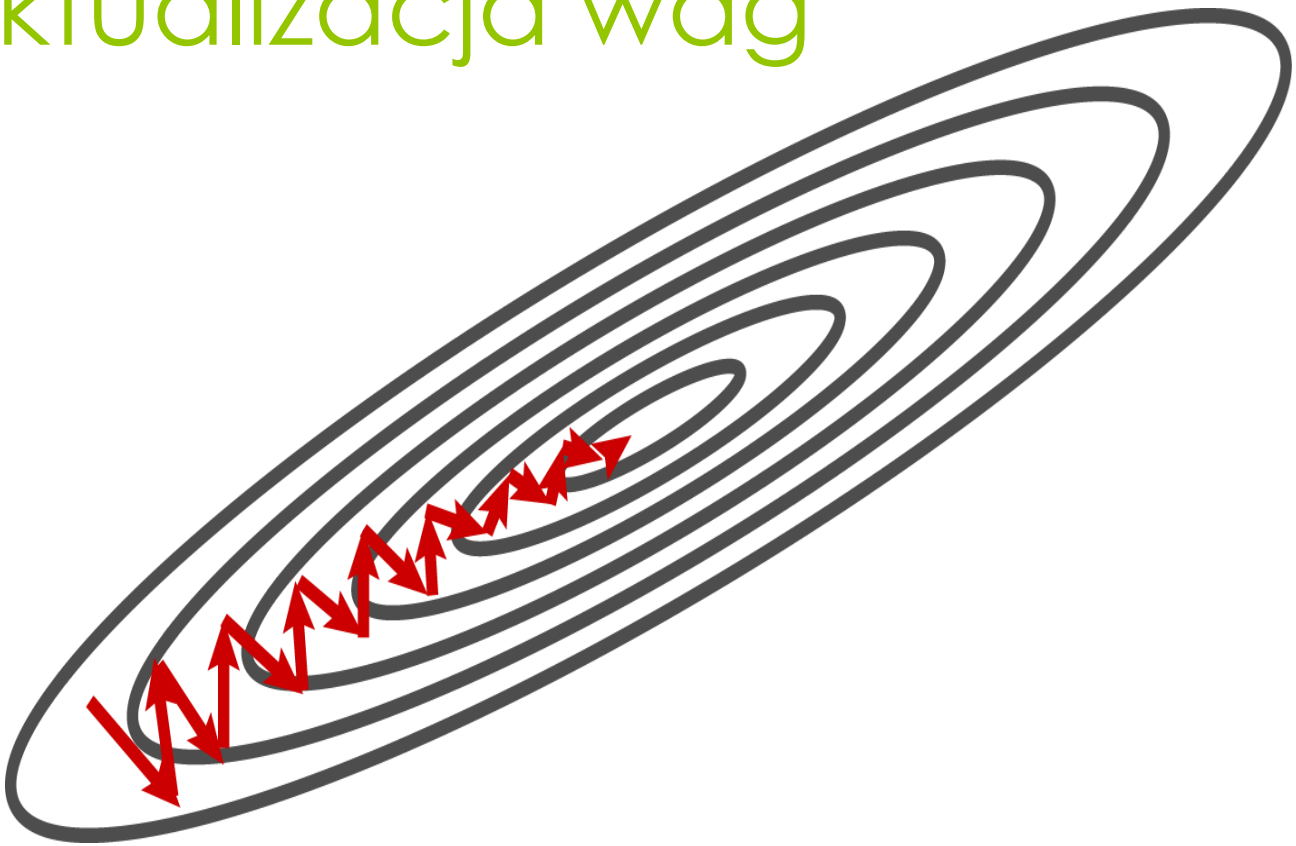
- *Backpropagation*
- Wyliczenie od ostatniej warstwy:
- Korekty wag $\mathbf{W}_n$ na podstawie funkcji kosztu
- Wstecz: korekty wag $\mathbf{W}_{n-1}$ na postawie wartości pochodnych z warstwy sąsiedniej

# Aktualizacja wag

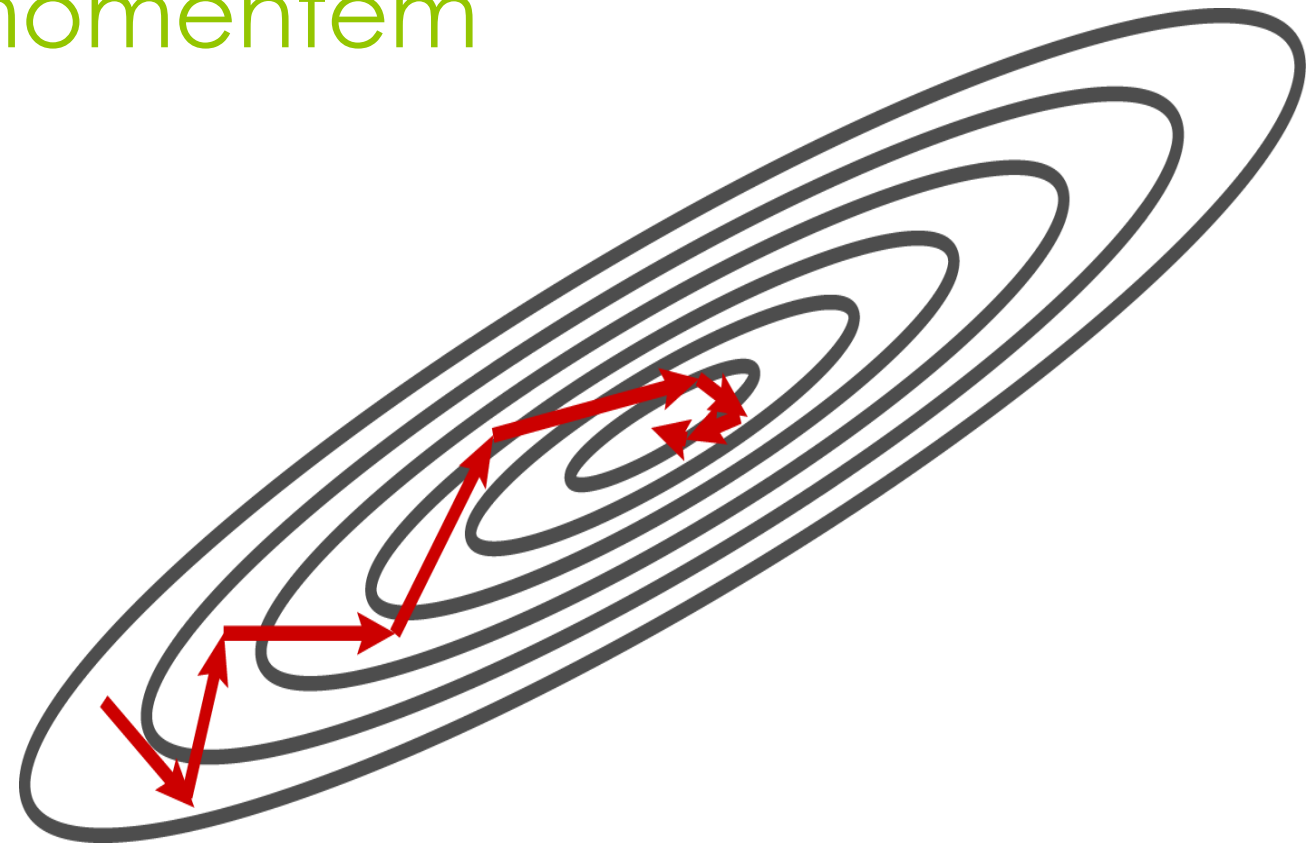
- $\mathbf{w}_{nowa} = \mathbf{w}_{stara} - \eta \cdot \text{gradient} \cdot \mathbf{x}^T + \text{moment}$



# Aktualizacja wag



# Aktualizacja wag z momentem



# Moment

- Moment wprowadza do algorytmu element bezwładności, który zmniejsza chwilowe i gwałtowne zmiany kierunku wskazywanego przez gradient funkcji błędy
- Uczenie nie wchodzi w płytkie minima lokalne
- Znaczne przyspieszenie nauki dla płaskich obszarów funkcji błędu

# Aktualizacja wag

- Aktualizacja wag wyliczana może być w oparciu o:
  - Wszystkie wektory **x** i **y**
    - Dokładne: każda poprawka uwzględnia wszystkie dane
    - Długotrwałe
  - Losowo wybrany podzbiór
    - Szybkie
    - Niedokładne: poprawka dla jednej próbki może prowadzić do pogorszenia dla pozostałych

# Stochastic Gradient Descent

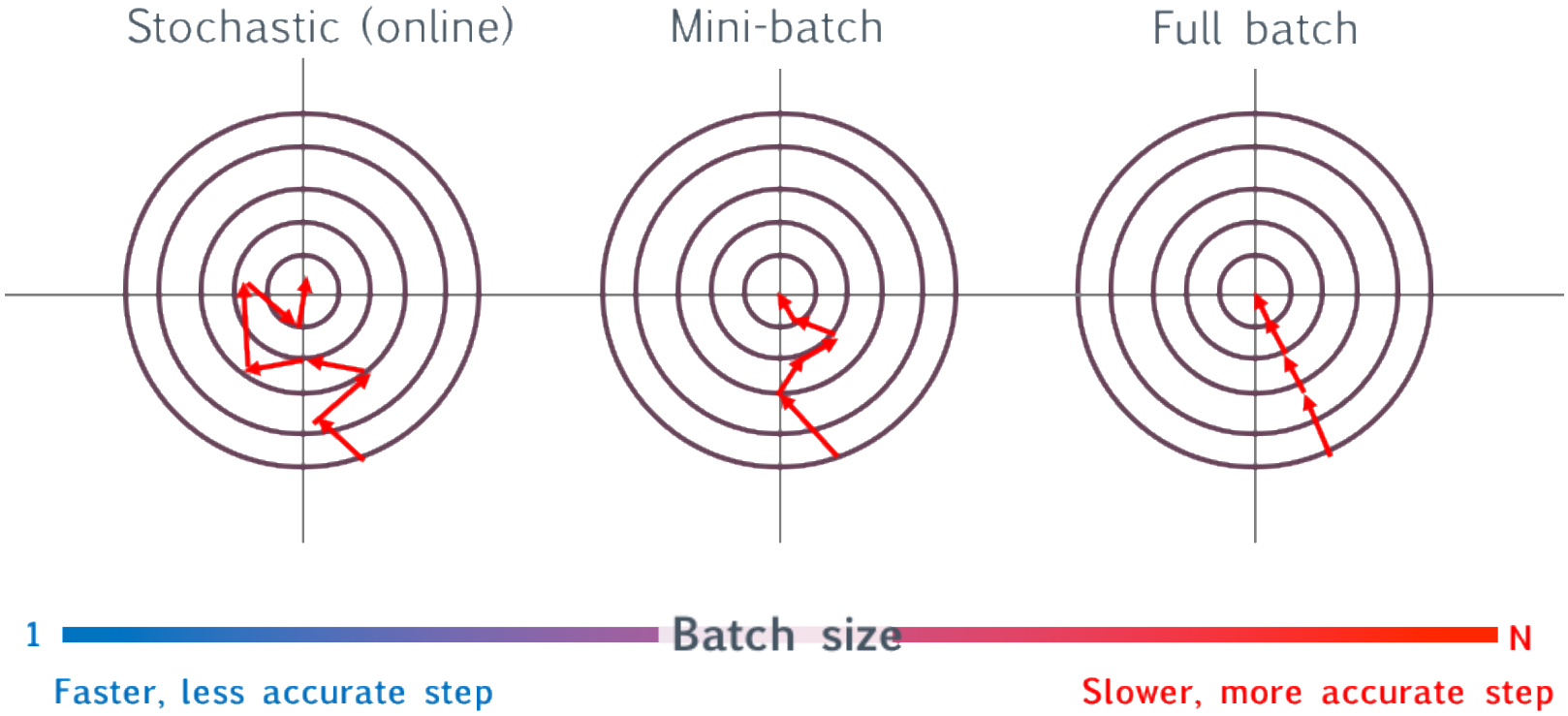
- Poprawka wag wykonywana po uwzględnieniu tylko jednej próbki
- Założenia:
  - Z czasem uzyska poprawę dla wszystkich
  - Zwykle stosuje się niewielki krok poprawki
  - Pomaga uzyskać regularyzację
- Czasochłonne

# Mini-batch

- Podzbiory danych wejściowych, np. 16, 32 próbki
- Aktualizacja wag w danym kroku
- Pobranie losowo kolejnego podzbioru
- Aktualizacja… itd.

- Kompromis między Full batch a SGD

# Aktualizacja wag - podsumowanie




- Źródło: INTEL AI Academy

- https://playground.tensorflow.org
- Learning rate
- Batch size

# Klasyfikacja – kodowanie typu „One-hot”

- Warstwa wyjściowa z n neuronów (n=liczba_klas)
- Każda klasa to „1” na odpowiednim wyjściu, pozostałe zera

| $\begin{bmatrix}1\\0\\0\end{bmatrix}$ | $\begin{bmatrix}0\\1\\0\end{bmatrix}$ | $\begin{bmatrix}0\\0\\1\end{bmatrix}$ |
|---|---|---|
| Rower | Osobowy | Ciężarowy |

# Dziękuję za uwagę

Wykład 2: sieci głębokie