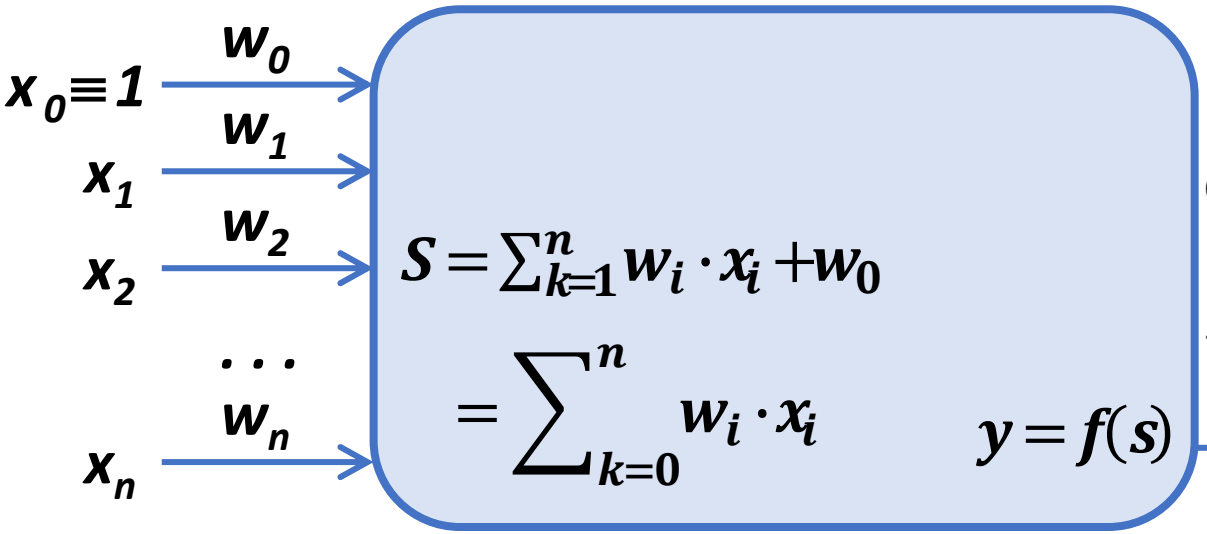
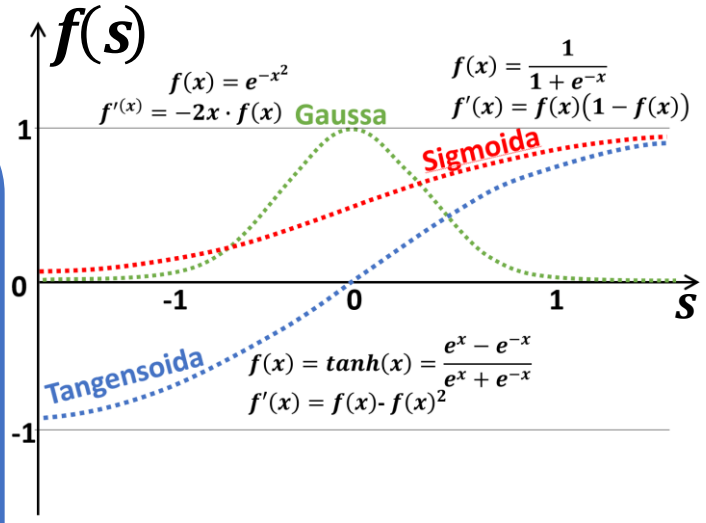
Inteligencja obliczeniowa w analizie danych

Wstęp do sztucznych sieci neuronowych

Prof. dr hab. inż. Norbert Skoczylas

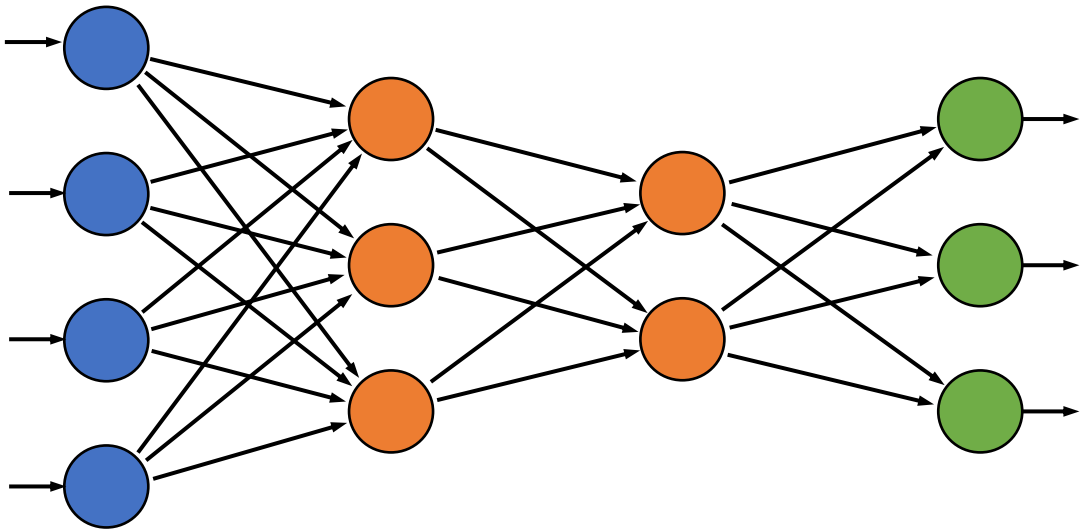
Model neuronu





Sieć MLP

Połączenie od warstw wcześniejszych do kolejnych (od wejścia, poprzez warstwy ukryte, do wyjścia) – **sieci jednokierunkowe - feedforward**



Uczenie i funkcja błędu

Błąd popełniany przez sieć neuronową zależny jest od współczynników wag występujących w sieci i poprawianych przez algorytmy uczenia.

Aby móc ocenić ten błąd najczęściej korzysta się z sumy kwadratów odchyłek …

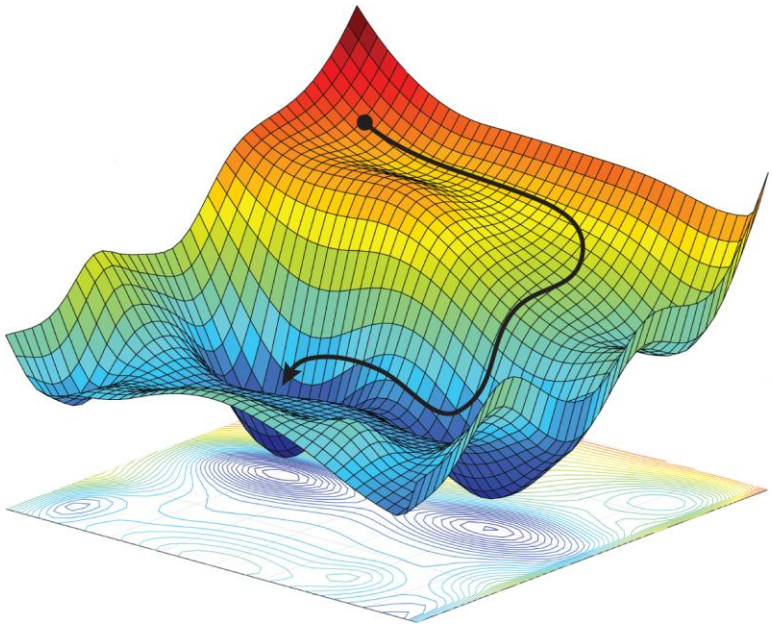
Błąd średniokwadratowy Mean Squared Error MSE

$$MSE = \frac{1}{N}\sum (z - y)^2$$

z – oczekiwane wyjścia
y – wyjścia sieci

Czym wobec tego formalnie jest uczenie sieci ? Jest problemem optymalizacyjnym, poszukiwania takiej kombinacji wag połączeń w sieci, aby błąd średniokwadratowy pomiędzy tym co oblicza sieć, a tym co oczekujemy aby obliczała, był jak najmniejszy ..

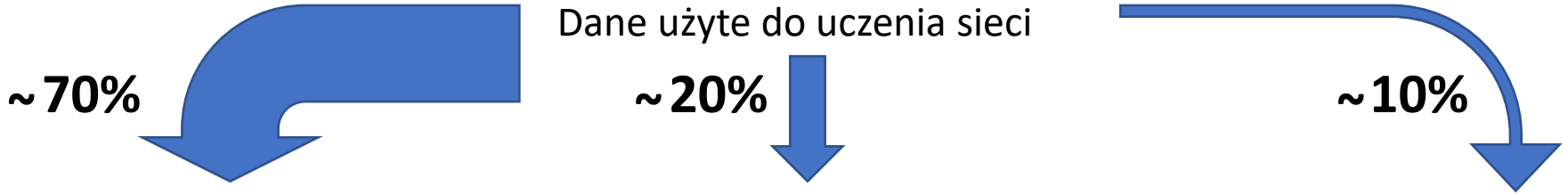
Algorytm wstecznej propagacji błędów polega na szukaniu kierunku spadku wielowymiarowej przestrzeni błędu … i na takich modyfikacjach wartości wag, żeby zmniejszać wartość funkcji błędu w kierunku jej najszybszego spadku ..



Często wykorzystywano metody ewolucyjne do optymalizacji wag sieci i minimalizacji funkcji błędu ..

Podział zbioru wejściowego na kategorie

Zbiór przypadków uczących, czyli zadań zawierających dane wejściowe oraz skojarzone z nimi odpowiedzi wzorcowe losowo dzielony jest na zbiory:



zbiór uczący
uczenia sieci z użyciem wybranego algorytmu uczenia

zbiór walidacyjny
wykorzystywany w procesie uczenia jedynie pośrednio do okresowej walidacji mającej na celu zapobieganie zjawisku przeuczenia..

zbiór testowy
przeprowadzenia po zakończeniu uczenia jednorazowej kontroli (czy w wyniku zbiegu okoliczności mimo okresowej walidacji nie doszło w trakcie uczenia do utraty zdolności generalizacji).

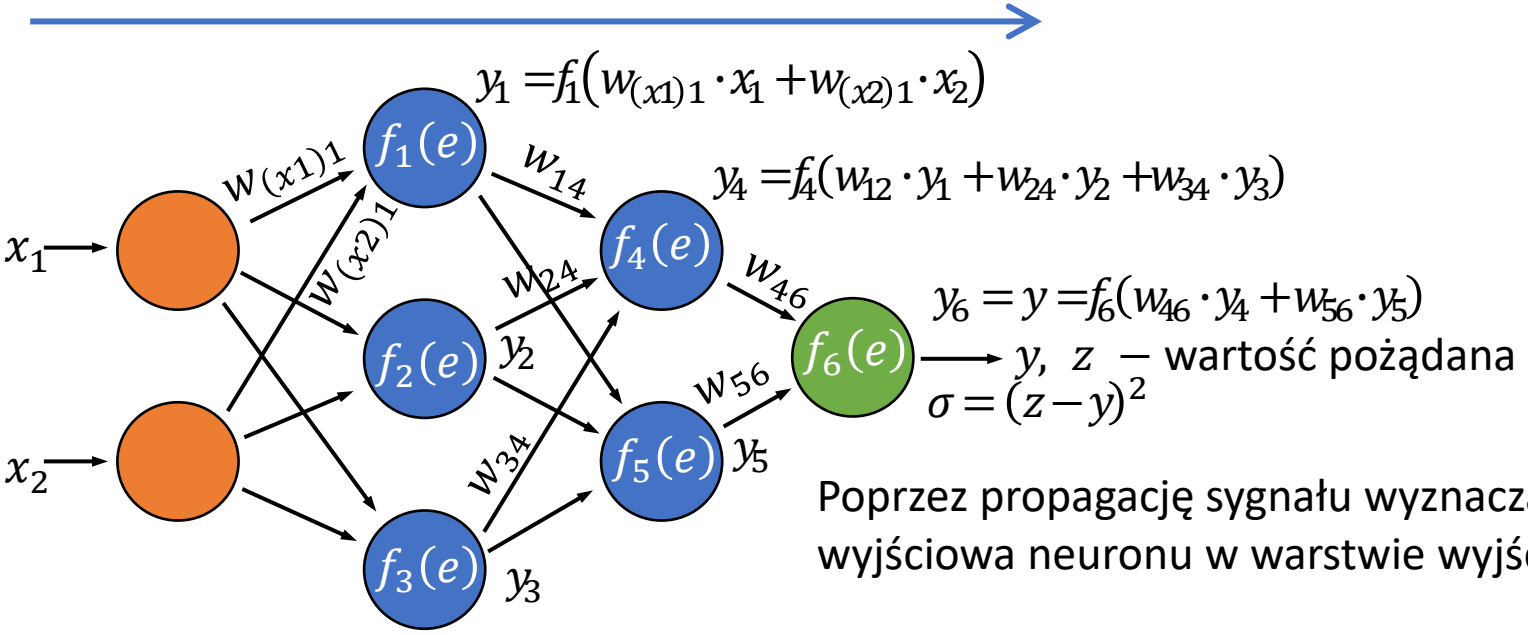
Główny problemem związany z wydzieleniem podzbiorów polega na konieczności posiadania w każdym zbiorze przypadków uczących reprezentatywnych dla całego zbioru. Przypadki unikatowe bądź wyłącznie standardowe, nie mogą pojawić się wyłącznie w jednym ze zbiorów.

Wielkość błędu dla zbioru testowego powinna być zbliżona do wielkości błędu dla zbioru walidacyjnego. Jeśli tak nie jest, to uczenie sieci trzeba powtórzyć, ponownie dzieląc w sposób losowy zbiór uczący i wydzielając nowy zbiór walidacyjny oraz zbiór testowy.

Metoda wstecznej propagacji błędów

umożliwiająca uczenie sieci wielowarstwowych poprzez propagację różnicy pomiędzy pożądanym a otrzymanym sygnałem na wyjściu sieci

Faza propagacji sygnału od wejść (wektora x) do wyjścia.



Poprzez propagację sygnału wyznaczana jest wartość wyjściowa neuronu w warstwie wyjściowej y_6.

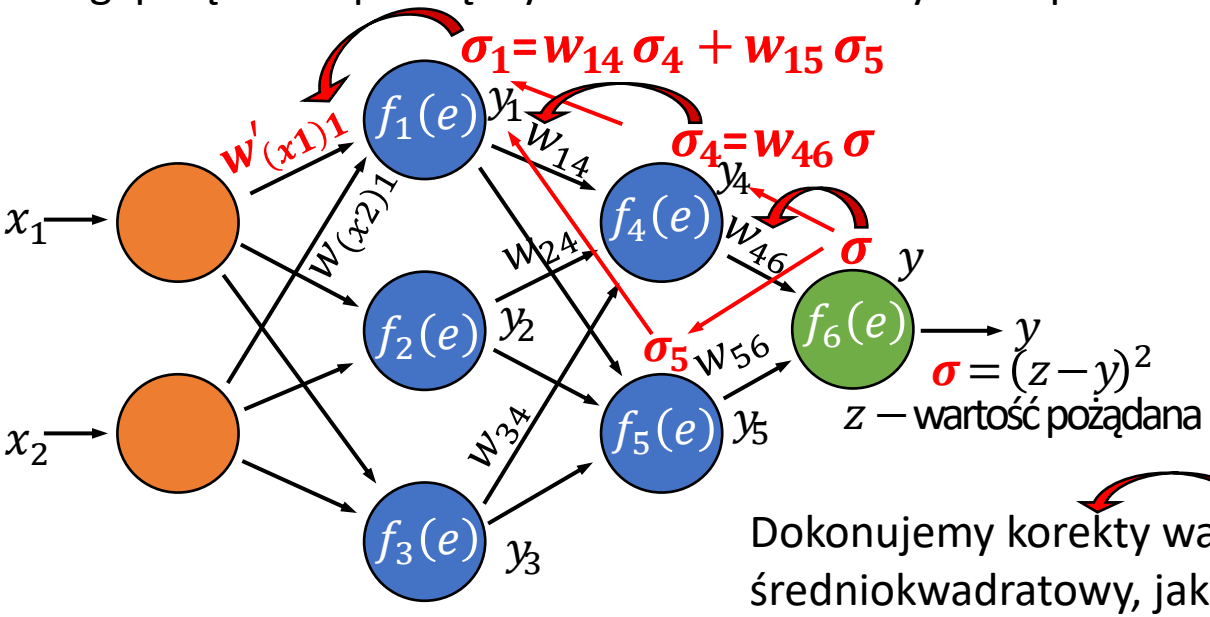
Obliczona wartość wyjściowa jest porównywana z wartością pożądaną (z) wyznaczoną w zbiorze uczącym przez nauczyciela.
Różnica pomiędzy wartością pożądaną i uzyskaną wynosi $\sigma = (z-y)^2$.

Metoda wstecznej propagacji błędów

Wartość błędu jest propagowana wstecz, ważona zgodnie z wagą połączenia między neuronami i sumowana w tych neuronach celem wyznaczenia ich błędu.

Faza wstecznej propagacji błędu od wyjścia y w kierunku wejść sieci

Przechodząc z błędem wstecz z 3 warstwy do 2, błąd jest ważony zgodnie z aktualną wartością wagi połączenia pomiędzy neuronem warstwy 3 i odpowiednim neuronem warstwy 2.



Po powrocie do warstwy 1 mamy rozpropagowane błędy po całej sieci…

Następnie dokonywana jest korekta wartości wag sieci.

Dokonujemy korekty wag sieci tak, żeby zmniejszyły błąd średniokwadratowy, jaki był obliczony na wyjściu sieci. W tym celu korzystamy z uogólnionej reguły delta, korzystając z pochodnej cząstkowej funkcji aktywacji $\frac{df_i(e)}{de}$.

Współczynnik a służy stopniowej adaptacji sieci oraz określa szybkość uczenia się (początkowo duży, na końcu uczenia mały – umożliwia przejścia od minimów lokalnych do minimum globalnego).

$$w'_{(x1)1} = w_{(x1)1} + a \cdot \sigma_1 \frac{df_1(e)}{de} x_1$$

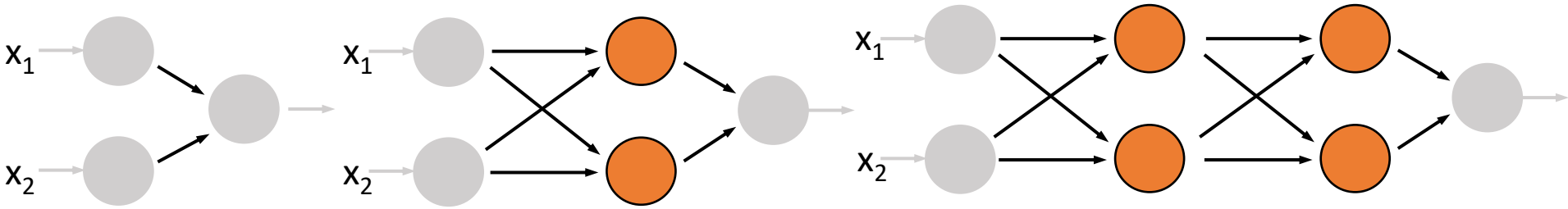
$$w'_{14} = w_{14} + a \cdot \sigma_4 \frac{df_4(e)}{de} y_1$$

$$w'_{46} = w_{46} + a \cdot \sigma \frac{df_6(e)}{de} y_4$$

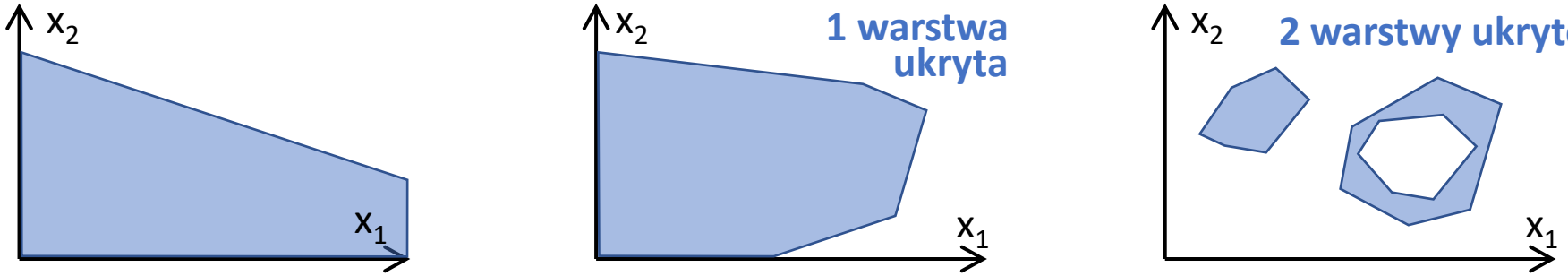
Kroki podczas budowy sieci: Utworzenie sieci:

Sieć nieliniowa – **wybór struktury sieci** ..

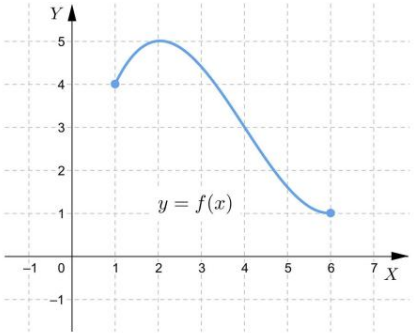
brak warstw ukrytych **1 warstwa ukryta** **2 warstwy ukryte**



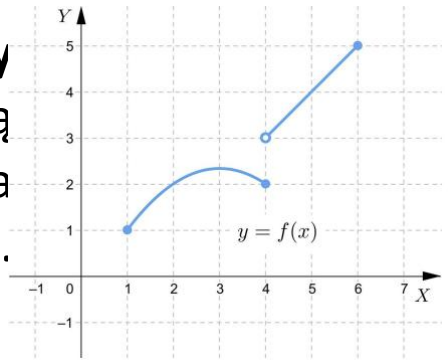
kształty obszarów decyzyjnych, jakie mogą tworzyć sieci o rożnej liczbie warstw ukrytych



sieć MLP z ***jedną warstwą ukrytą*** jest w stanie aproksymować dowolną ***funkcję ciągłą*** z dowolną dokładnością.



Dwie warstwy ukryte rozszerzają możliwości na **funkcje nieciągłe**.

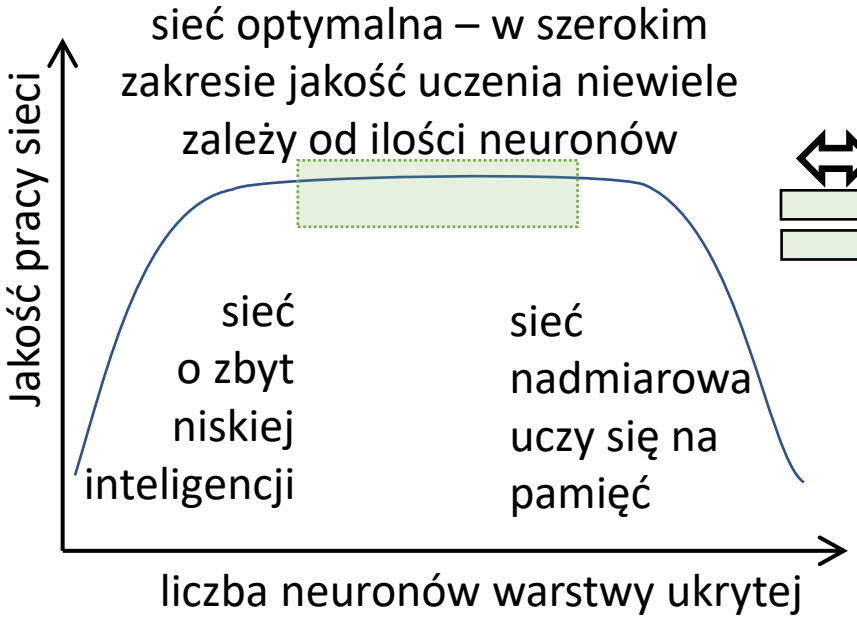


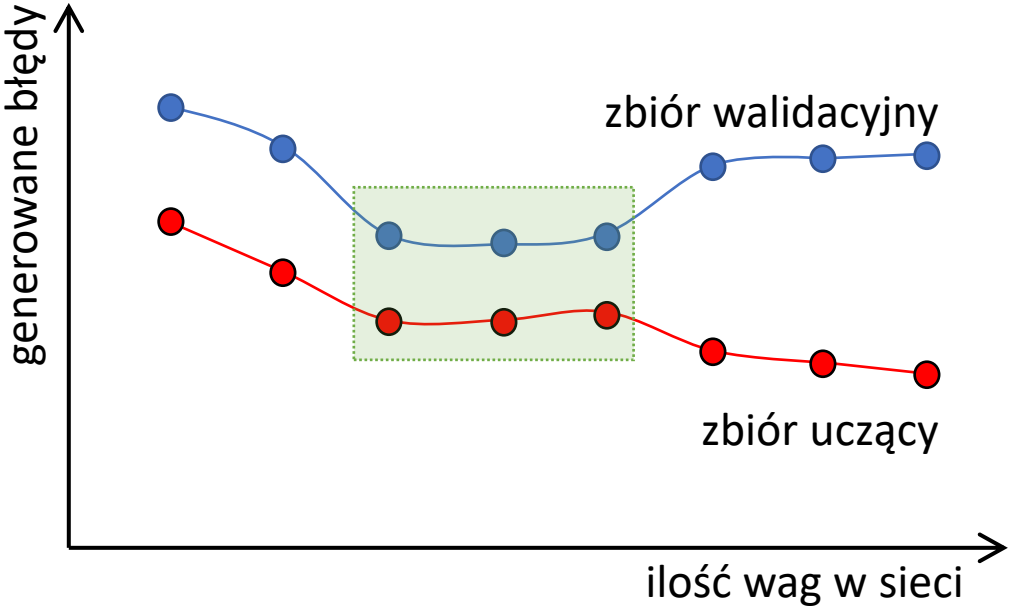
Ile neuronów warstwy ukrytej ? pierwsze przybliżenie: $k = \sqrt{n \cdot m}$ średnia geometryczna ilości wejść n i wyjść m sieci

Kroki podczas budowy sieci: Utworzenie sieci:

Ile przypadków uczących i ?
Ilość wag w (połączeń neuronów) nie powinna przekraczać liczby przypadków uczących i – zalecane ($w < {}^{i}/_{10}$)

Czy więcej neuronów w warstwie ukrytej to zawsze lepiej ?
Najczęściej tak, ale nie wtedy, gdy mamy ograniczoną liczebność zbioru uczącego !





Jakość pracy sieci ↗ generowane błędy ↘

Generalizacja

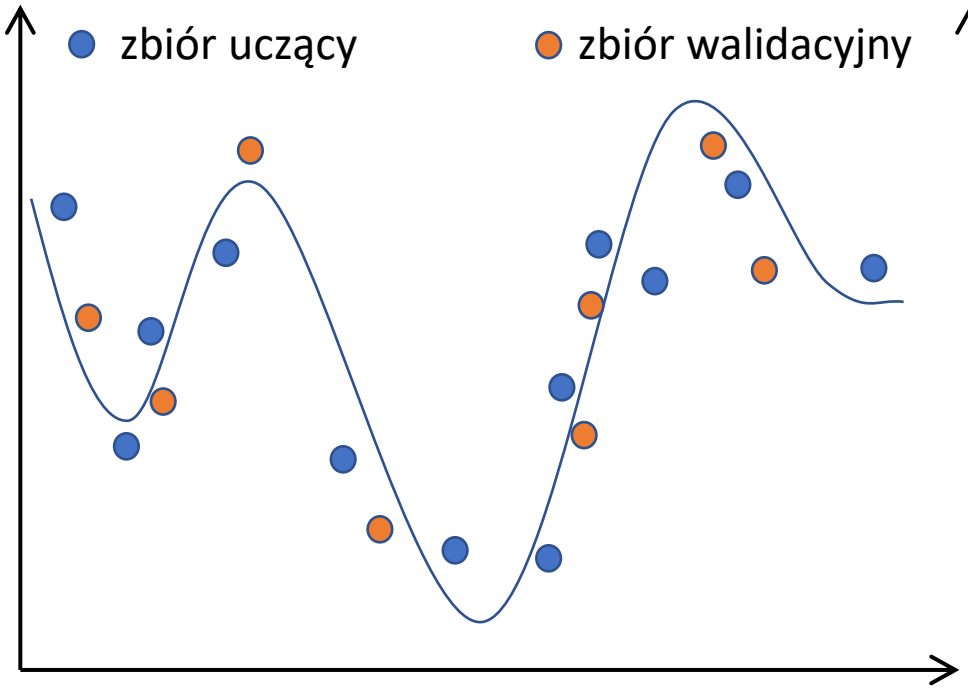
Sens użycia sieci neuronowej polega na tym, że musi ona (po nauczeniu) rozwiązywać zadania podobne do tych, na których była uczona – ale nie identyczne z nimi.

Takie przeniesienie nabytej wiedzy na nowe przypadki nazywane jest generalizacją.

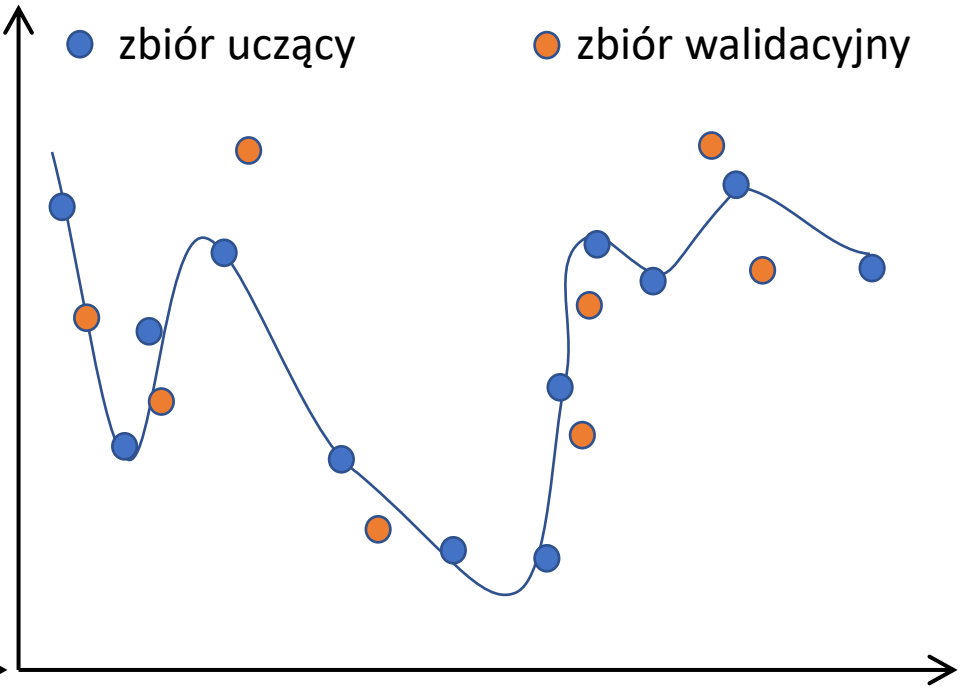
Zagrożeniem dla generalizacji jest przeuczenie.

Gdy sieć jest przeuczona – następuje nadmierne dopasowanie jej zachowania do drugorzędnych (nieistotnych) szczegółów konkretnych przypadków uczących – nie mających istotnego znaczenia z punktu widzenia istotnych cech rozwiązywanego zadania.

Sieć poprawnie generalizująca problem



Sieć która utraciła zdolność generalizacji



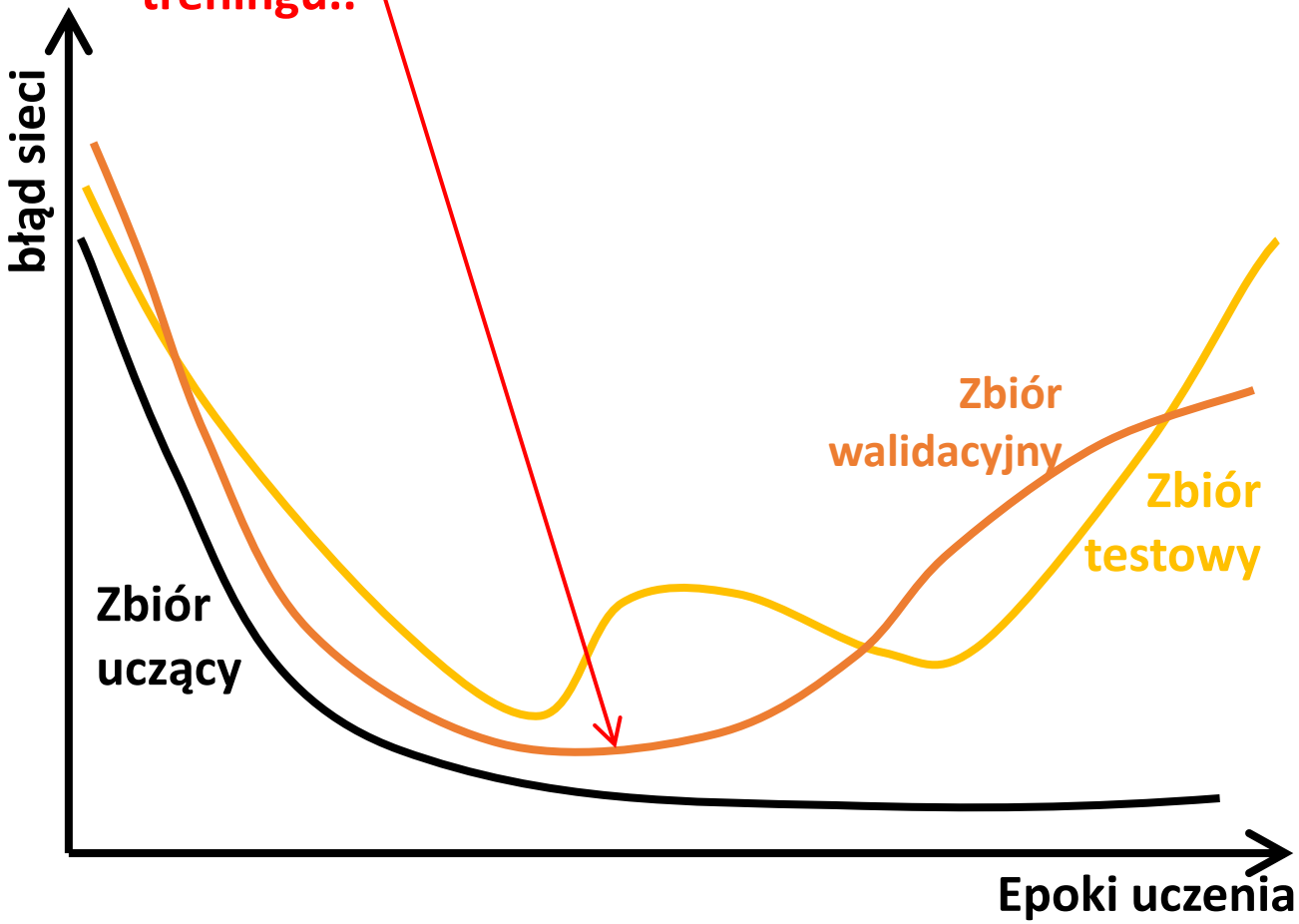
Generalizacja i przeuczenie (ilość epok nauczania)

Na zbiorze treningowym błąd jest w stanie zawsze osiągnąć 0..

Dążymy do minimalizacji błędu na danych, które nie zostały użyte w treningu..

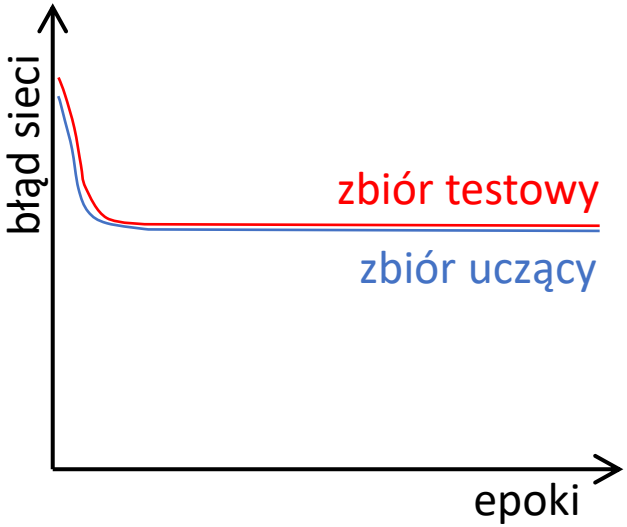
Generalizacja jest celem uczenia - uogólnienie reguły..

Zbyt długie uczenie sieci neuronowej powoduje, że sieć nadmiernie uzależnia swoje działanie od cech użytych do uczenia przypadków uczących – w tym także od cech drugorzędnych, nie dających podstaw do generalizacji.

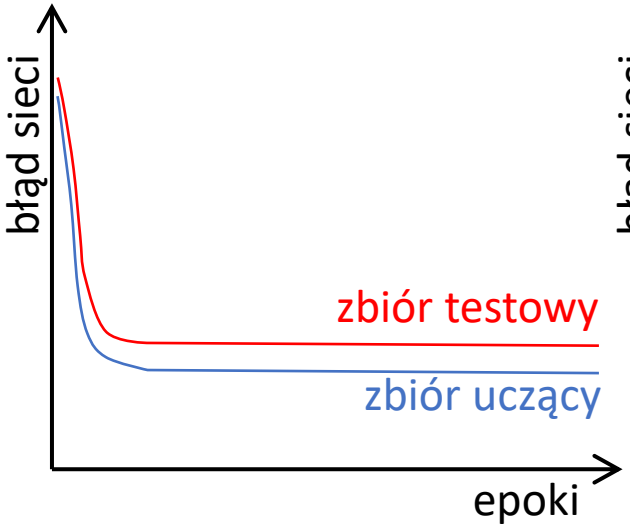


Przeuczenie (ilość epok nauczania) – interpretacja wykresów uczenia

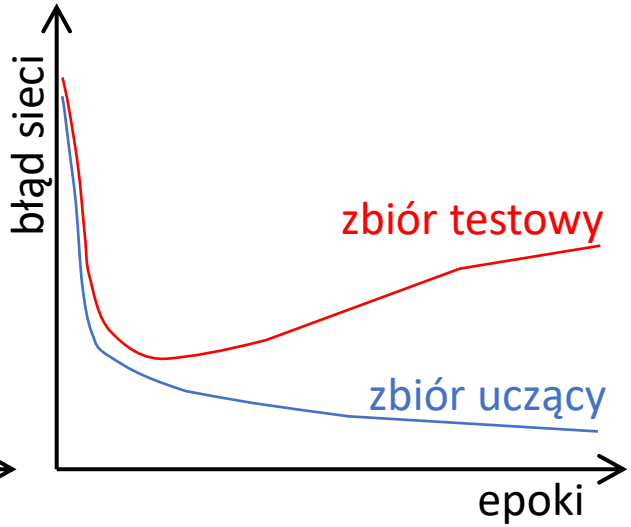
niedouczenie:
błąd testowy oraz
błąd treningowy
pozostają wysokie



prawidłowy przebieg uczenia

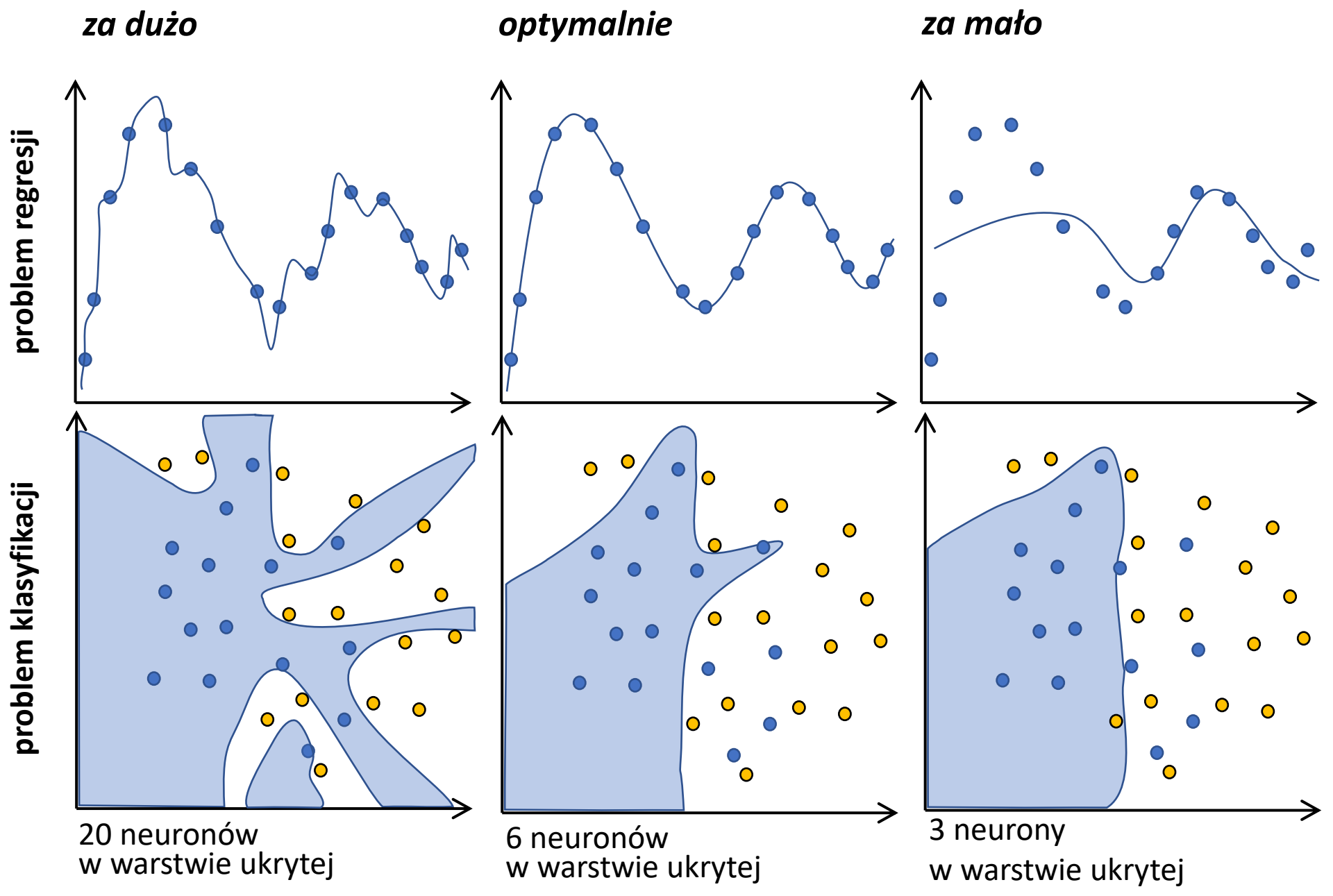


przeuczenie:
błąd testowy rośnie ↑,
a błąd treningowy maleje ↓



Rys: Riedmiller, Machine Learning

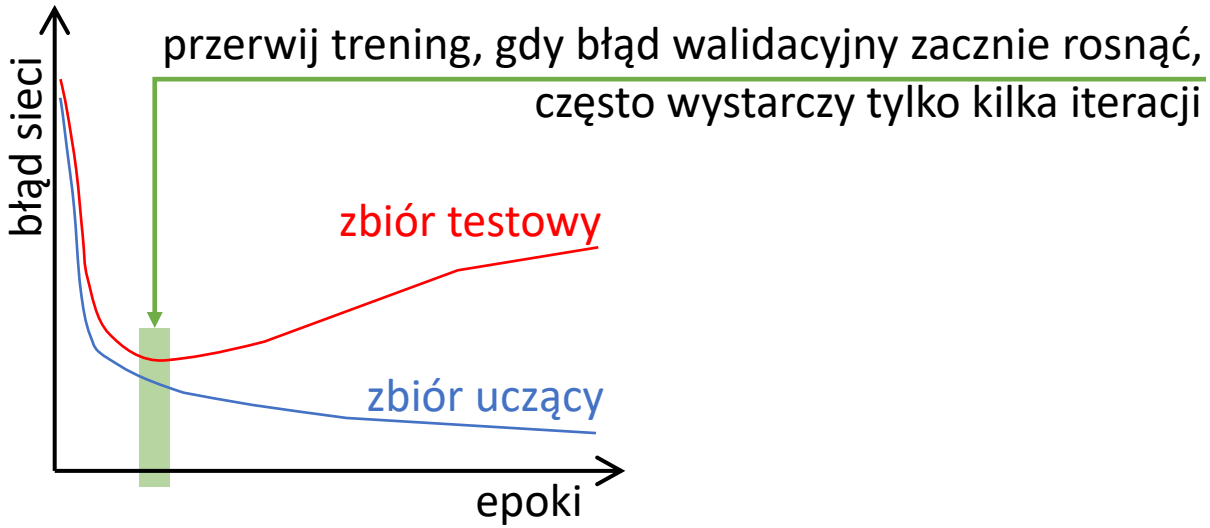
Generalizacja i przeuczenie (rozmiar warstwy ukrytej)



Co można zrobić aby poprawić generalizowalność problemu przez sieć ?

- ✓ dobór wielkości modelu (ilości neuronów i wag), preferowane są najprostsze rozwiązania
- ✓ z pośród wielu modeli wybierz ten o najmniejszym błędzie walidacyjnym i najmniejszej liczbie parametrów (neuronów)
- ✓ usuwanie nadmiarowych połączeń lub neuronów z wytrenowanej sieci (wagi zerowe oznaczają brak połączenia)
- ✓ zwiększenie liczby danych
- ✓ selekcja cech - wybór tylko istotnych cech do treningu
- ✓ dodanie szumu do wag lub danych treningowych
- ✓ wczesne zatrzymanie treningu, zanim błąd walidacyjny wzrośnie

zapamiętaj model o najmniejszym błędzie walidacyjnym



Co i kiedy ??

Kolejny wykład - sieci głębokie 28 maja..

2 czerwca – 18:10
Krzysztof Nojman - Infrastruktura Big Data

ostatnie obligatoryjne laborki –
projekt neuro – 22 i 23 maja..

18 czerwca – termin zero egzaminu

27 czerwca – pierwszy termin egzaminu

Maj 2025

Pn	Wt	Śr	Cz	Pt	SO	N
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

Czerwiec 2025

Pn	Wt	Śr	Cz	Pt	SO	N
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25		27	sesja	

25 maja/ fuzzy

TERMINY ODDANIA PROJEKTÓW

15 czerwca/ neuro

TERMINY ODDANIA PROJEKTÓW

Wszelkie problemy z genetycznym – piszcie albo przyjdźcie na nieobligatoryjne, albo do mnie (p.317)

Nazwi	Imię	Monte	Genetyc	Fuzzy	Neuro	Ile oddanych	Ocena z projektów
A	Łucja Weronika	3/21/2025	4/19/2025	5/17/2025		3	4
B	Jakub Marcel	3/24/2025				1	2
B	Maja	3/21/2025	5/18/2025			2	3
Ch	Jakub Hubert	4/3/2025	3/30/2025			2	3
Ch	Wiktor Jan	3/30/2025	5/11/2025	5/18/2025		3	4
D	Konrad Adam	4/6/2024				1	2
D	Kamil Stanisław	3/19/2025	4/19/2025	5/11/2025		3	4
D	Adam	4/4/2025	5/12/2025			2	3
D	Tomasz Piotr	3/31/2025	4/22/2025	5/15/2025		3	4
D	Gabriela	3/30/2025	5/25/2025			2	3
D	Jan Bartosz	3/21/2025	3/30/2025	5/11/2025		3	4
E	Maurycy	4/1/2025	5/9/2025			2	3
F	Julia	4/6/2025	5/13/2025			2	3
F	Aleksandra Maria	3/27/2025	4/27/2025	5/18/2025		3	4
G	Zuzanna Ewa	4/6/2025				1	2
G	Patrycja	3/19/2025		5/13/2025		2	3
G	Mikołaj	4/6/2025				1	2
G	Konrad Wojciech	4/3/2025	4/26/2025	5/9/2025		3	4
G	Jakub Przemysław	3/19/2025	5/11/2025			2	3
H	Karol	4/3/2025				1	2
H	Aleksander Jakub	3/21/2025	5/13/2025			2	3
I	Bartosz	4/6/2025	5/18/2025			2	3
J	Filip Andrzej	4/6/2025	4/29/2025	5/18/2025		3	4
J	Aleksandra Monika	4/6/2025				1	2
J	Julia Weronika	3/29/2025	4/30/2025			2	3
J	Roksana Kamila	3/24/2025	4/28/2025	5/13/2025		3	4
J	Weronika	4/6/2025	5/11/2025			2	3
Ki	Maria	3/31/2025	4/18/2025	5/9/2025		3	4
Kn	Maria	3/17/2025	4/8/2025	5/18/2025		3	4
K	Karolina Agnieszka	3/27/2025	5/7/2025	5/18/2025		3	4
K	Oliwier Piotr	3/15/2025	4/11/2025	5/13/2025		3	4
K	Julia Anita	3/27/2025	5/11/2025			2	3
K	Bartłomiej Mariusz	3/23/2025	4/30/2025			2	3
K	Dawid Tomasz	3/20/2025	5/11/2025			2	3
K	Natalia Katarzyna	3/21/2025	4/23/2025	5/15/2025		3	4
K	Kacper Bartłomiej	4/5/2025	5/11/2025			2	3
L	Patryk	4/3/2025	5/11/2025			2	3
M	Patrycja Anna	3/21/2025	5/11/2025	5/20/2025		3	4
M	Gerard	3/20/2025	5/4/2025			2	3
M	Jakub Franciszek	3/16/2025	4/23/2025	5/13/2025		3	4
M	Eliza Klaudia	3/28/2025	4/23/2025			2	3
M	Karolina	4/6/2025	4/24/2025			2	3
O	Joanna Julia	3/26/2025	5/9/2025			2	3
P	Miłosz	3/25/2025	5/9/2025	5/18/2025		3	4
P	Szymon Mateusz	4/3/2025	5/8/2025			2	3
Pie	Bartosz	4/6/2025	5/19/2025			2	3
Piw	Bartłomiej Jakub	3/24/2025	5/11/2025			2	3
P	Magdalena Maria	4/4/2025	5/9/2025			2	3
P	Dominika	3/31/2025	5/11/2025			2	3
R	Gabriel Mieczysław	4/4/2025	5/5/2025			2	3
S	Dominik	3/30/2025	4/23/2025	5/18/2025		3	4
S	Marcin Jan	3/17/2025	4/15/2025	5/16/2025		3	4
S	Julia Krystyna	3/25/2025	5/11/2025			2	3
S	Julita	3/28/2025	5/7/2025			2	3
S	Szymon Marcin	3/16/2025	4/22/2025	5/11/2025		3	4
S	Weronika					0	2
S	Martyna Joanna	3/26/2025	5/11/2025	5/19/2025		3	4
S	Wiktoria Danuta	4/4/2024	5/18/2025			2	3
S	Aleksandra	3/27/2025	5/5/2025	5/16/2025		3	4
S	Anna Sara	4/3/2025	5/11/2025			2	3
Szcz	Magdalena Anna	4/6/2025				1	2
Szt	Magdalena	3/19/2025	4/25/2025			2	3
Sz	Mikołaj	4/3/2025	5/12/2025			2	3
T	Katarzyna	4/1/2025	4/28/2025	5/18/2025		3	4
T	Witold	3/17/2025				1	2
W	Julia Barbara	3/26/2025				1	2
W	Katarzyna	4/6/2025	5/12/2025	5/20/2025		3	4
W	Oliwia Klaudia	3/25/2025	5/11/2025			2	3
Z	Michał	3/21/2025	5/7/2025			2	3
Z	Karolina	4/2/2025	5/11/2025	5/12/2025		3	4
	ile % ? / średnio	100,0	85,5	37,7	0,0	2,200	3,21

Pod linkiem na dole slajdu
znajduje się zestawienie z datami oddania
kolejnych projektów i ..
ogólnie bieżącą sytuacją –
proszę sprawdzajcie, czy się nie pomyliłem –
jeśli się pomyliłem, piszcie do mnie …

genetyczny – 85%

fuzzy – 38%

https://docs.google.com/spreadsheets/d/1LYZjd4Fg9ql7hI4Qhglm06P6IkBDF_mb/edit?usp=sharing&ouid=102885865370465498299&rtpof=true&sd=true

Mini-projekt 4 z SSN – analiza efektów uczenia sieci..

```
clear all
close all

ile_danych = 500; % Liczba przypadków uczących
ile_epok = 100; % Liczba epok
a = 0; % przedział losowania zmiennych
b = 12;
hold on;

% Taką funkcję postaramy się aproksymować siecią
x = a:0.05:b;
y = 0.5*cos (0.2.*x.^2)+0.5;
plot(x, y, 'k', 'linewidth',2);

% generowanie przypadków uczących - ta sama funkcja z szumem
x_siec = a + (b-a).*rand(ile_danych,1); % Losowanie wektora zmiennych x_siec
y_siec = 0.5*cos (0.2.*x_siec.^2)+0.5 + 0.1.*randn(ile_danych,1); % Generowanie wektora y_siec na podstawie x_siec i dodatkowych szumów
plot(x_siec, y_siec, 'rx', 'MarkerSize',5);

% Definicja architektuy_siec sieci neuronowej
netconf= [12]; % [20 20] -> 2 warstwy uky_siecte po 20 neuronów w każdej
net = feedforwardnet(netconf);

% Ustawienie parametrów uczenia sieci
net.trainParam.epochs = ile_epok; % ile epok podczas uczenia ?
%net.trainParam.max_fail = ile_epok; % liczba niepowodzeń walidacji z rzędu - maksymalna liczby kolejnych epok, w któy_siecch błąd walidacji nie poprawił się *.
net.trainParam.goal = 0; % minimalna wartość błędu trenowania
net = train(net, x_siec.', y_siec.');

% Generowanie aproksymowanego przez sieć wektora odpowiedzi Y dla argumentów x
ypred = net(x);

% y_siecsowanie wykresów aproksymacji przez sieć
plot(x, ypred, 'r', 'linewidth',2); % y_siecsowanie predykcji sieci
legend('model do odwzorowania','punkty z szumem (przypadki wejsciowe)','wynik aproksymacji sieci','Location','northoutside')
```

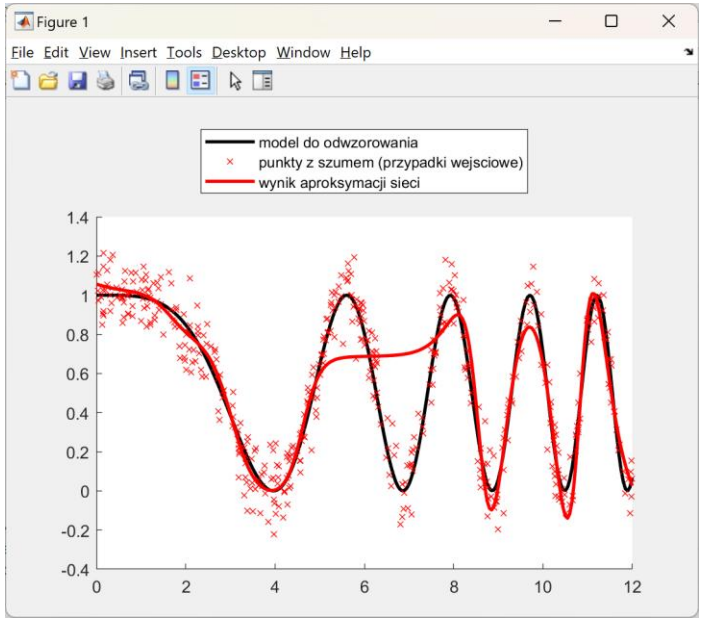
---

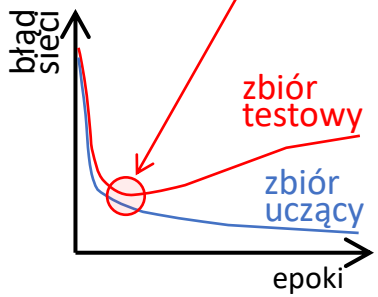
* Innymi słowy, jest to liczba epok, po któy_siecch trening sieci zostanie przerwany, jeśli błąd na zbiorze walidacyjnym (lub testowym) nie ulegnie poprawie. Parametr ten służy do kontroli procesu uczenia i pomaga uniknąć przeuczenia (overfittingu) sieci. Jeśli błąd walidacji nie poprawia się przez określoną liczbę kolejnych epok, oznacza to, że sieć nie jest w stanie poprawić jakości uczenia i dalsze uczenie może prowadzić do przeuczenia. Przerwanie treningu w takim przypadku pozwala zaoszczędzić czas i uniknąć uzyskiwania wyników, które nie będą generalizować dobrze dla innych zbiorów danych. Jeśli zaremujesz tą linię, algrytm przerwie uczenie automatycznie, gdy wykryje przeuczenie ..

Mini-projekt 4 z SSN – analiza efektów uczenia sieci w zależności od:

Liczebności przypadków uczących
Ilości epok
Budowy sieci (ilości neuronów i warstw ukrytych)

```
clear all
close all

ile_danych = 500; % Liczba przypadków uczących
ile_epok = 100; % Liczba epok
a = 0; % przedział losowania zmiennych
b = 12;
hold on;

% Taką funkcję postaramy się aproksymować siecią
x = a:0.05:b;
y = 0.5*cos (0.2.*x.^2)+0.5;
plot(x, y, 'k', 'linewidth',2);

% generowanie przypadków uczących - ta sama funkcja z szumem
x_siec = a + (b-a).*rand(ile_danych,1); % Losowanie wektora zmiennych x_siec
y_siec = 0.5*cos (0.2.*x_siec.^2)+0.5 + 0.1.*randn(ile_danych,1); % Generowanie wektora
plot(x_siec, y_siec, 'rx', 'MarkerSize',5);                % y_siec na podstawie x_siec i dodatkowych szumów

% Definicja architektuy_siec sieci neuronowej
netconf= [12]; % [20 20] -> 2 warstwy uky_siecte po 20 neuronów w każdej
net = feedforwardnet(netconf);

% Ustawienie parametrów uczenia sieci
net.trainParam.epochs = ile_epok; % ile epok podczas uczenia ?
%net.trainParam.max_fail = ile_epok; % liczba niepowodzeń walidacji z rzędu - maksymalna liczby kolejnych epok, w których błąd walidacji nie poprawił się *.
net.trainParam.goal = 0; % minimalna wartość błędu trenowania
net = train(net, x_siec.', y_siec.');

% Generowanie aproksymowanego przez sieć wektora odpowiedzi Y dla argumentów x
ypred = net(x);

% y_siecsowanie wykresów aproksymacji przez sieć
plot(x, ypred, 'r', 'linewidth',2); % y_siecsowanie predykcji sieci
legend('model do odwzorowania','punkty z szumem (przypadki wejsciowe)','wynik aproksymacji sieci','Location','northoutside')
```

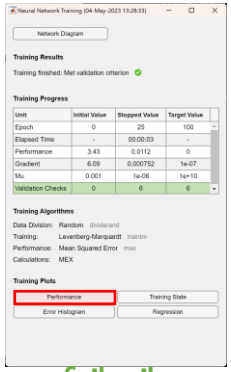




* Innymi słowy, jest to liczba epok, po których trening sieci zostanie przerwany, jeśli błąd na zbiorze walidacyjnym (testowym) nie ulegnie poprawie. Parametr ten służy do kontroli procesu uczenia i pomaga uniknąć przeuczenia (overfittingu) sieci. Jeśli błąd walidacji nie poprawia się przez określoną liczbę kolejnych epok, oznacza to, że sieć nie jest w stanie poprawić jakości uczenia i dalsze uczenie może prowadzić do przeuczenia. Przerwanie treningu w takim przypadku pozwala zaoszczędzić czas i uniknąć uzyskiwania wyników, które nie będą generalizować dobrze dla innych zbiorów danych. Jeśli „zaremujesz" tą linię, algrytm przerwie uczenie automatycznie, gdy wykryje przeuczenie ..

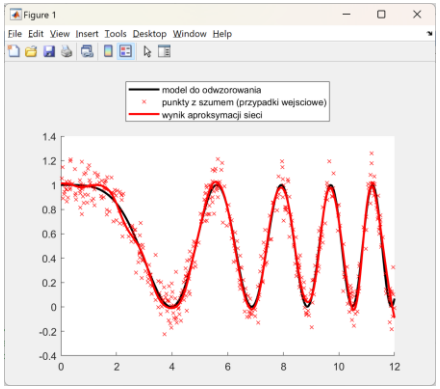
Mini-projekt 4 z SSN – analiza efektów uczenia sieci ..

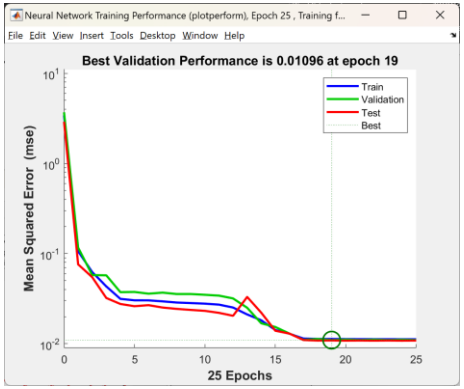
średnia geometryczna ilości wejść n i wyjść m sieci



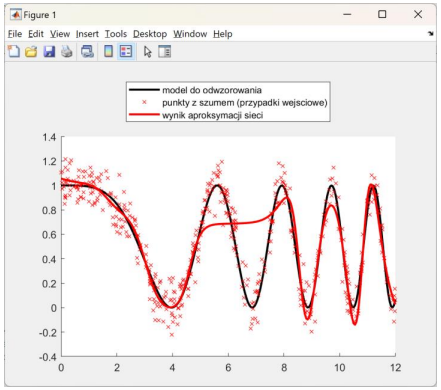
Sieć nieliniowa – wybór struktury sieci ..
Ile neuronów warstwy ukrytej ? pierwsze przybliżenie: $k = \sqrt{n \cdot m}$
Ilość wag w nie powinna przekraczać liczby przypadków uczących i – zalecane ($w < {}^{i}/_{10}$)

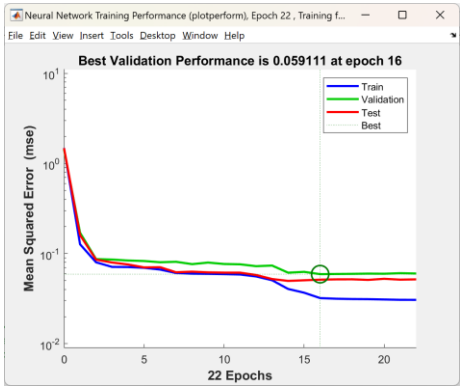
```
ile_danych = 500;
ile_epok = 100;
netconf= [12];
%net.trainParam.max_fail = ile_epok;
```



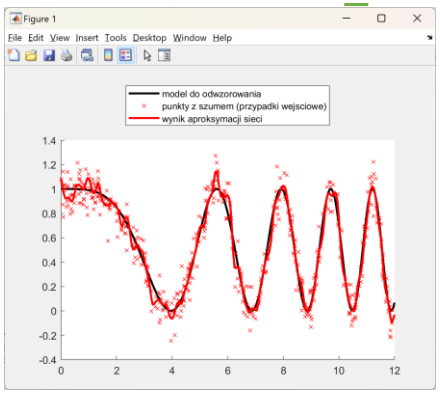


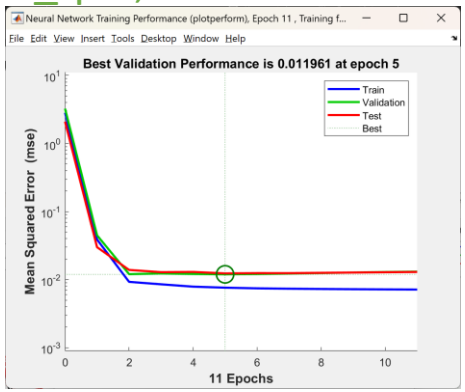

```
ile_danych = 500;
ile_epok = 100;
netconf= [8];
%net.trainParam.max_fail = ile_epok;
```



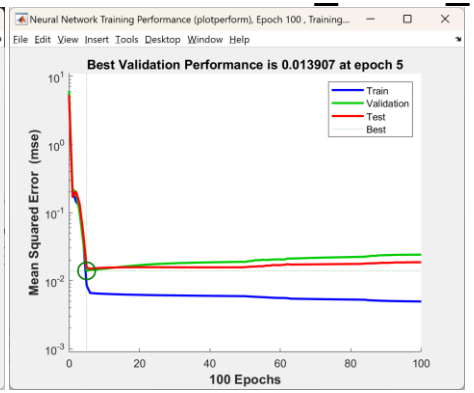


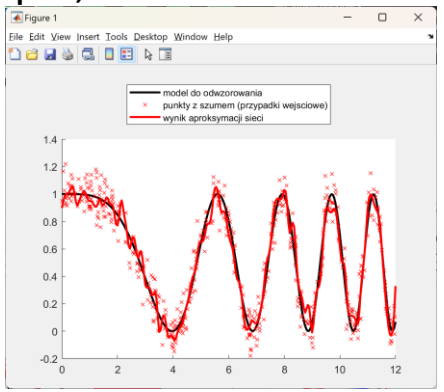

```
ile_epok = 100; netconf= [45 45];
%net.trainParam.max_fail = ile_epok;
```






```
ile_epok = 100; netconf= [45 45];
net.trainParam.max_fail = ile_epok;
```





Mini-projekt 4 z SSN – analiza efektów uczenia sieci w zależności od:

- Można zastosować własną funkcję, bądź zaproponowaną …

- Sprawdźmy dla ustalonej liczebności zbioru uczącego:
  - Jaka liczebność neuronów w warstwie ukrytej jest zbyt niska, aby zagwarantować poprawną aproksymację ?
  - Jak zbyt duża liczebność neuronów w sieci skutkuje utratą zdolności do generalizacji problemu ..
  - Czy dodanie drugiej warstwy ukrytej mocno zmienia sytuację ?
  - Dla ambitnych – przeróbka na funkcję nieciągłą ..

- Dla zoptymalizowanej liczby neuronów manipuluj liczebnością zbioru uczącego:
  - Jakie są konsekwencje zbyt małej liczebności zbioru uczącego ?
  - Czy zwiększanie liczebności zbioru uczącego jest korzystne ?

- Aktywuj linię: net.trainParam.max_fail = ile_epok; - proces uczenia nie będzie automatycznie przerywany..
  - Oceń wpływ długości procesu uczenia na efekt „przeuczenia”

Analogiczne „doświadczenia z siecią” można zrobić na dowolnej platformie ..

Alternatywny mini-projekt – dowolna własna implementacja SSN

Inteligencja obliczeniowa w analizie danych

dziękuję za uwagę !