IMPLEMENTACJA SIECI NEURONOWYCH MLP Z WALIDACJĄ KRZYŻOWĄ

Celem ćwiczenia jest zapoznanie się ze sposobem działania sieci neuronowych typu MLP (multi-layer perceptron) uczonych nadzorowaną (z nauczycielem, supervised) metodą propagacji wstecznej błędów (backpropagation) z wykorzystaniem metody walidacji krzyżowej (cross-validation) dla zbioru Irys (z ML Repository).

Jak rozpocząć?

1. Wczytać dane Irysów do tablicy lub tabeli.
2. Zaprojektować strukturę sieci: 4 wejścia, 3 wyjścia, a warstwa ukryta powinna zawierać od 5 do 30 neuronów, ew. można zastosować dwie lub trzy takie warstwy.
3. Zainicjować wagi tej sieci małymi liczbami z zakresu od -0,1 do +0,1.
4. Napisać metodę realizującą pobudzanie kolejnych neuronów w kolejnych warstwach sieci od wejścia do wyjścia.
5. Obliczyć błąd na wyjściach neuronów klasyfikujących na podstawie danych uczących określających pożądane klasy dla danych wejściowych.
6. Napisać metodę realizującą przekazywanie wsteczne błędów do kolejnych neuronów w poprzednich warstwach sieci od wyjścia do wejścia i obliczania korekty wag sieci.
7. Następnie po prezentacji całego zbioru danych dokonać korekty wag (off-line, batch).
8. Na końcu dostosuj algorytm tak, żeby zastosować 10-krotną walidację krzyżową.

JAK POPRAWIAMY WAGI POŁĄCZEŃ MIĘDZY NEURONAMI?

Wagi połączeń między neuronami można poprawiać (aktualizować) w trakcie uczenia metodą:

- On-line (po prezentacji każdego wzorca uczącego i obliczeniu korekty wag)
- Off-line (batch learning, uczenie wsadowe) – polegające na sumowaniu w pomocniczej tabeli wszystkich korekt obliczonych dla wszystkich wzorców uczących w jednym cyklu uczącym. Na koniec te sumy dzielimy przez ilość wszystkich wzorców uczących, wyznaczając średnią korektę dla wszystkich wzorców uczących i dopiero ją wykorzystujemy do korekty wag sieci.

Najczęściej wykorzystywana jest metoda off-line, gdyż zapewnia większą stabilność uczenia, ponieważ wszystkie wzorce prezentowane są niejako tej samej sieci w danym cyklu uczenia (tj. sieci o tych samych wagach), zaś w przypadku uczenia metodą on-line kolejność prezentacji wzorców uczących wpływa na sposób adaptacji sieci, a kolejne wzorce prezentowane są sieci po jej dopasowaniu do poprzednich wzorców. Taki sposób uczenia jest charakterystyczny dla ludzi, lecz w przypadku uczenia zdefiniowanych zbiorów uczących kolejność prezentacji wzorców nie powinna być powodem do ich faworyzowania w procesie uczenia.

ZAIMPLEMENTUJ METODĘ UCZENIA PROPAGACJI WSTECZNEJ

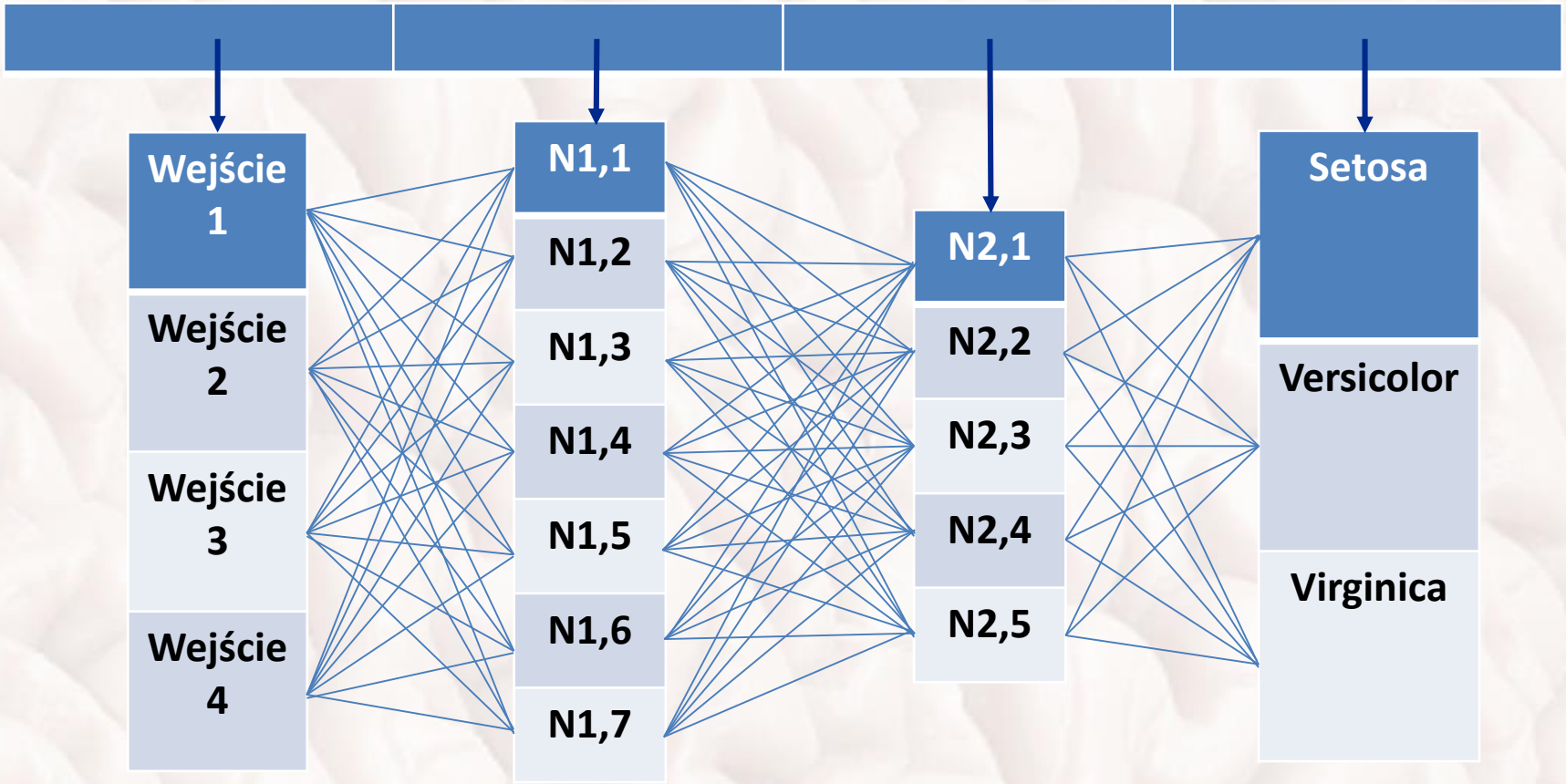
DO DYSKRYMINACJI 3 KLAS Irysów

Zbuduj 3 warstwową sieć neuronową składającą się z neuronów sigmoidalnych (lub tangens hiperboliczny) do dyskryminacji wzorców 3 klas Irysów: Setosa, Versicolor i Virginica na podstawie wszystkich czterech parametrów.

1. **Wczytaj zbiór danych Iris**
2. **Zbuduj sieć neuronową wielowarstwową, posługując się 2D tablicami double do reprezentacji wag (w_{ij}) pomiędzy neuronami sąsiednich warstw neuronów, oraz 1D tablicami rekordów do reprezentacji sum ważonych (s_i), obliczonych wartości wyjściowych (y_i) oraz sum ważonych propagowanych wstecz błędów (δ_{ik}) dla neuronów.**
3. **Zaimplementuj metodę propagacji wstecznej błędów.**
4. **Zastosuj metodę walidacji krzyżowej do nauki.**
5. **Zastosuj algorytmy genetyczne do wybrania optymalnej ilości neuronów w warstwie ukrytej.**
6. **Można poeksperymentować z 2. warstwą ukrytą (sieć 4-warstwowa).**

BUDOWA STRUKTURY SIECI NEURONOWEJ

Zaleca się posłużenie się tablicami tablic lub listami list implementującymi strukturę sieci w taki sposób, iż główna tablica/lista reprezentuje warstwy sieci, a wewnętrzne tablice/listy reprezentują neurony w tych warstwach:



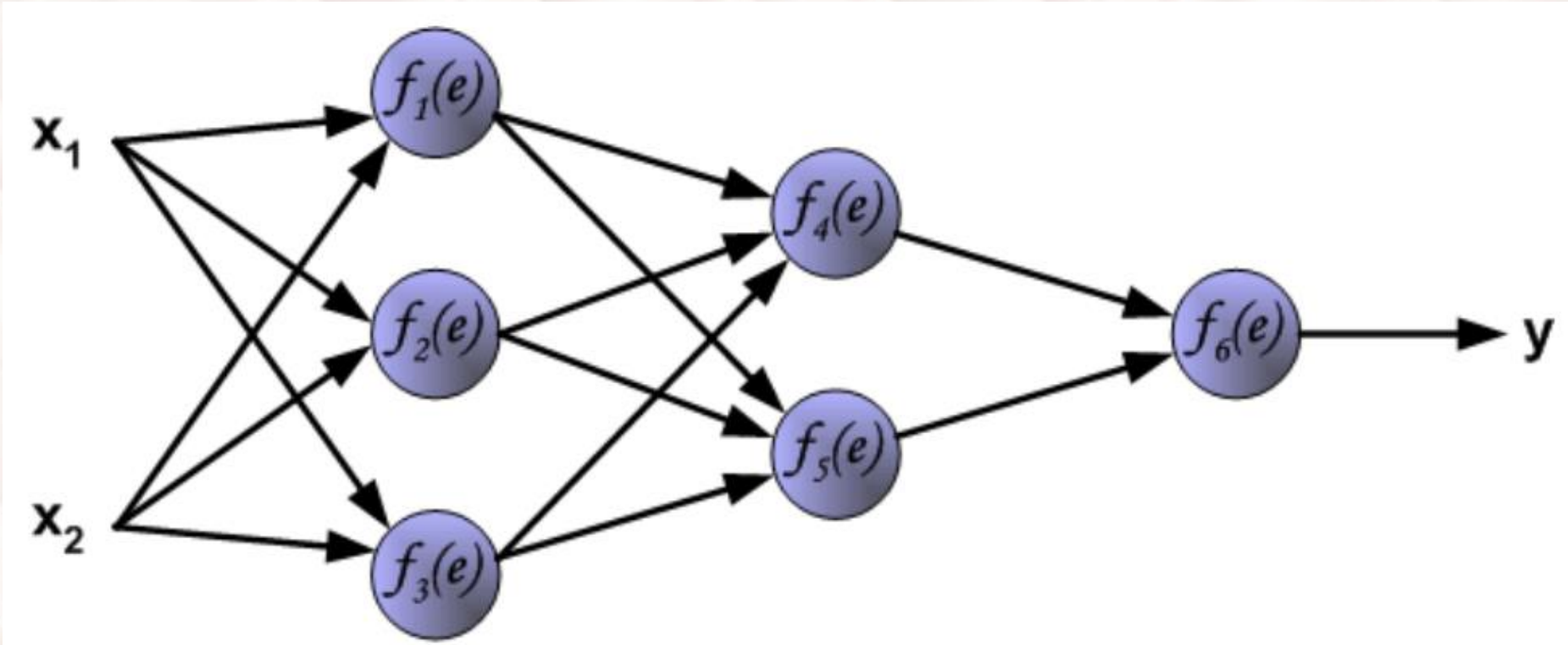
TABLICE DO IMPLEMENTACJI WAG I WARTOŚCI NEURONÓW

Zaleca się posłużenie się tablicami wag w następującej postaci:

	x1	x2
F1	w11	w12
F2	w21	w22
F3	w31	w32

	y1	y2	y3
F4	w41	w42	w43
F5	w51	w52	w53

	y4	y5
F6	w64	w65



SIECI NEURONOWE WIELOWARSTWOWE

MLP – Multilayer Perceptron

Sieci wielowarstwowe to odpowiednio połączone warstwy neuronów zwykle o nieliniowych funkcjach aktywacji (np. neurony sigmoidalne, radialne), aczkolwiek czasami w warstwie wyjściowej pojawiają się neurony liniowe.

Sieci wielowarstwowe muszą posiadać minimalnie dwie warstwy neuronów:

warstwę wejściową i **warstwę wyjściową**

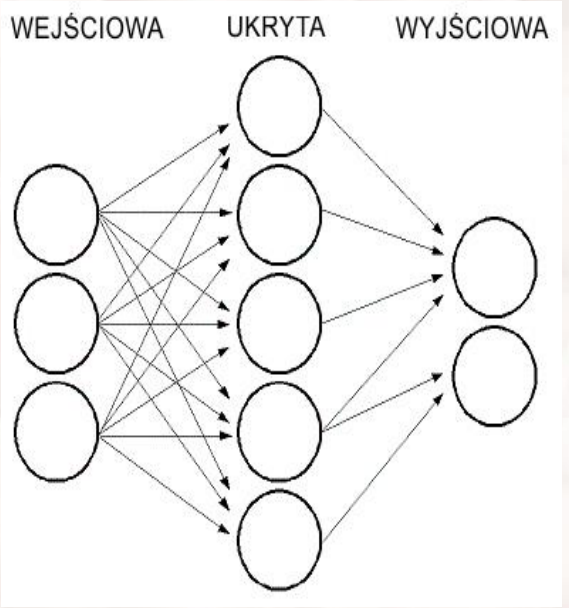
pomiędzy którymi może być jedna lub więcej **warstw ukrytych**.

W każdej warstwie może występować różna ilość neuronów:

W warstwie wejściowej odpowiada ona zwykle ilości parametrów opisujących dane wejściowe, w warstwie wyjściowej np. ilości klas, natomiast ilość neuronów w warstwach ukrytych odpowiada za możliwości modelu.

Neurony łączą się tylko pomiędzy (zwykle) sąsiednimi warstwami, zaś wewnątrz warstw nie występują połączenia pomiędzy neuronami.

Neurony zwykle łączymy pomiędzy sąsiednimi warstwami na zasadzie każdy z każdym.



UCZENIE SIECI WIELOWARSTWOWYCH

Sieci wielowarstwowe uczymy zwykle metodami nadzorowanymi (tzn. z nauczycielem):

1. **Podajemy na wejście sygnał wejściowy (zwykle w postaci wektora lub macierzy danych)**
2. **Obliczamy wartości wyjściowe neuronów w 1. warstwie i poprzez ich połączenia z neuronami w 2. warstwie podajemy te wartości jako sygnały wejściowe dla neuronów w 2. warstwie.**
3. **Obliczamy wartości wyjściowe w kolejnych warstwach tym samym sposobem.**
4. **Wartości wyznaczone w ostatniej warstwie stanowią zarazem odpowiedź sieci na podany sygnał wejściowy.**
5. **Sygnał ten porównujemy z wzorcowym (określonym przez nauczyciela) i wyznaczamy błąd.**
6. **Korzystając metod gradientowych propagujemy błąd wstecz przez sieć i dokonujemy korekty wartości wag połączeń synaptycznych.**

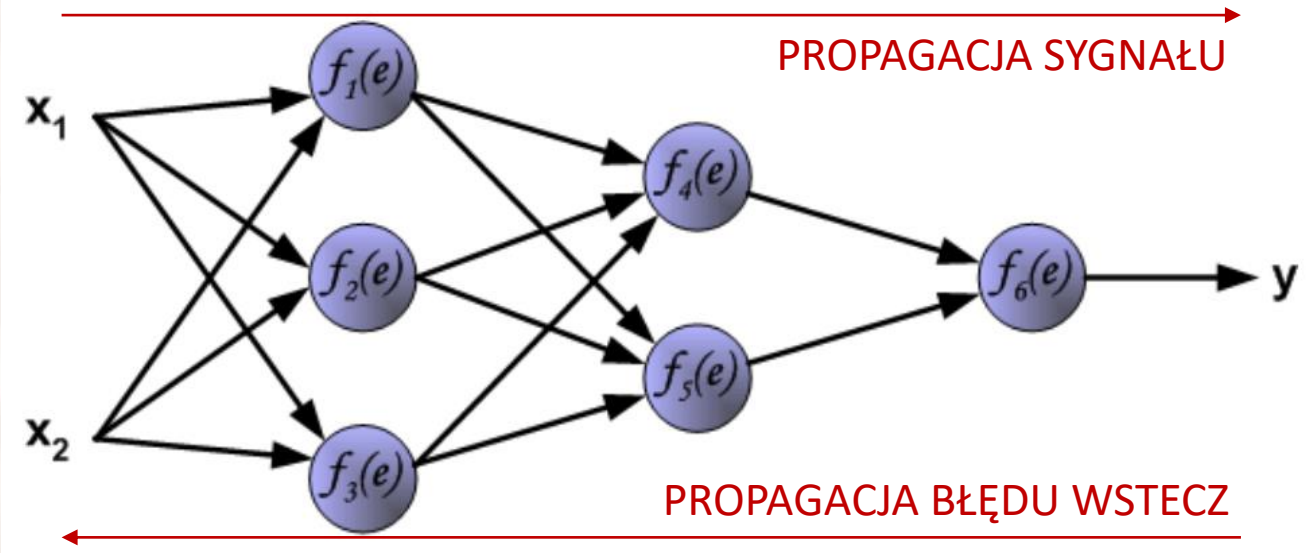
ALGORYTM WSTECZNEJ PROPAGACJI BŁĘDÓW

Pierwszym i zarazem najbardziej popularnym algorytmem do uczenia sieci wielowarstwowych jest algorytm wstecznej propagacji błędów (*backpropation algorithm*), którego działanie oparte jest na regule delta.

Wyznaczamy średniokwadratową funkcję błędu dla sieci Q(w), a następnie dążymy do znalezienia minimum tej funkcji względem wektora w.

Uczenie składa się z dwóch naprzemiennych faz:

1. **Fazy propagacji sygnału od wejść (wektora x) do wyjścia.**
2. **Fazy wstecznej propagacji błędu od wyjścia y w kierunku wejść sieci.**

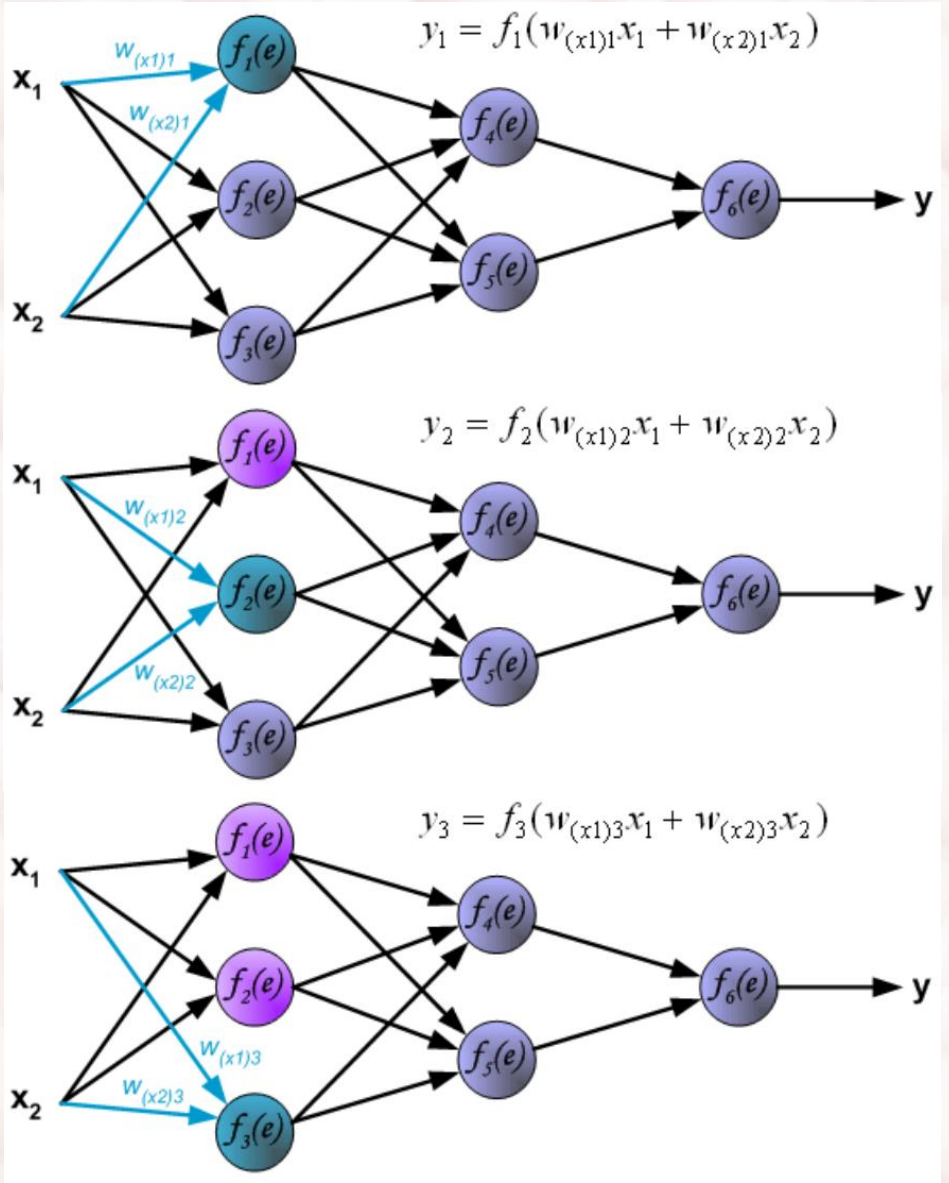


PROPAGACJA SYGNAŁU PRZEZ PIERWSZĄ WARSTWĘ SIECI

Sieć wielowarstwową pobudzamy sygnałami wejściowymi (x_1, x_2) kolejno warstwami neuronów.

Najpierw pobudzane są neurony w 1. warstwie i obliczana jest ich wartość wejściowa y_1, y_2 i y_3.

Wartości wyjściowe następnie są wykorzystane jako sygnały wejściowe w kolejnej warstwie neuronów.

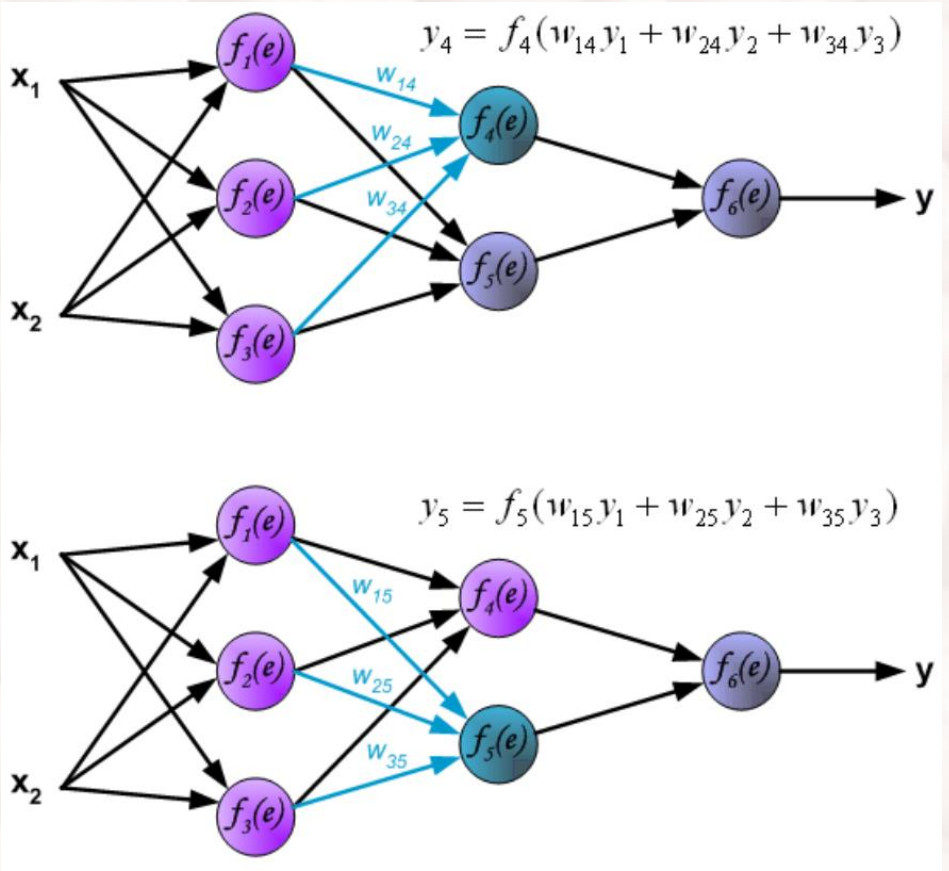


PROPAGACJA SYGNAŁU PRZEZ DRUGĄ WARSTWĘ SIECI

Drugą (ukrytą) warstwę neuronów sieci wielowarstwowej pobudzamy sygnałami wyjściowymi warstwy pierwszej (y_1, y_2 i y_3) obliczonymi w poprzednim kroku.

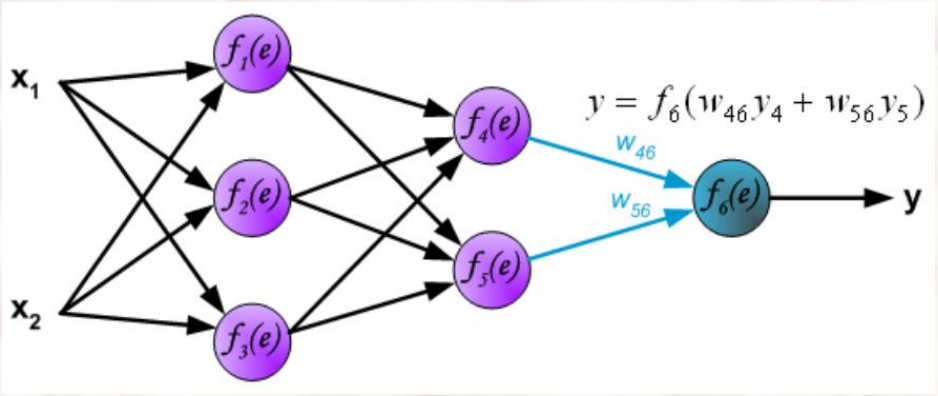
Na tej podstawie wyznaczane są wartości wyjściowe neuronów w warstwie drugiej (y_4 i y_5).

Obliczone wartości wyjściowe następnie są wykorzystane jako sygnały wejściowe w ostatniej warstwie neuronów, tzw. Warstwie wyjściowej sieci, która w naszym przypadku zawiera tylko 1 neuron.



PROPAGACJA SYGNAŁU PRZEZ TRZECIĄ WARSTWĘ SIECI

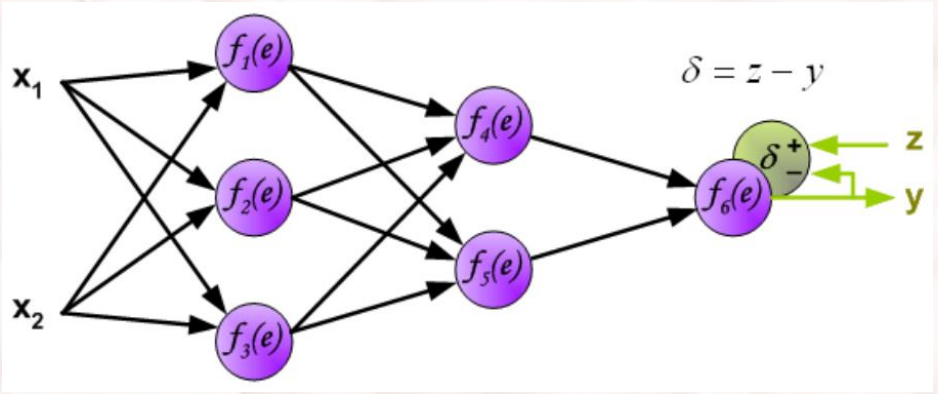
Trzecia warstwa neuronów sieci wielowarstwowej zawierająca pojedynczy neuron pobudzamy sygnałami wyjściowymi warstwy drugiej (y_4 i y_5) obliczonymi w poprzednim kroku.



Na tej podstawie wyznaczana jest wartość wyjściowa neuronu w warstwie trzeciej (y_6).

Obliczona wartość wyjściowa jest porównywana z wartością pożądaną (z) wyznaczoną w zbiorze uczącym przez nauczyciela, a różnica pomiędzy wartością pożądaną i uzyskaną ($\delta = z - y$) stanowią o dalszych krokach działania algorytmu.

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



PROPAGACJA WSTECZNA BŁĘDU DO DRUGIEJ WARSTWY

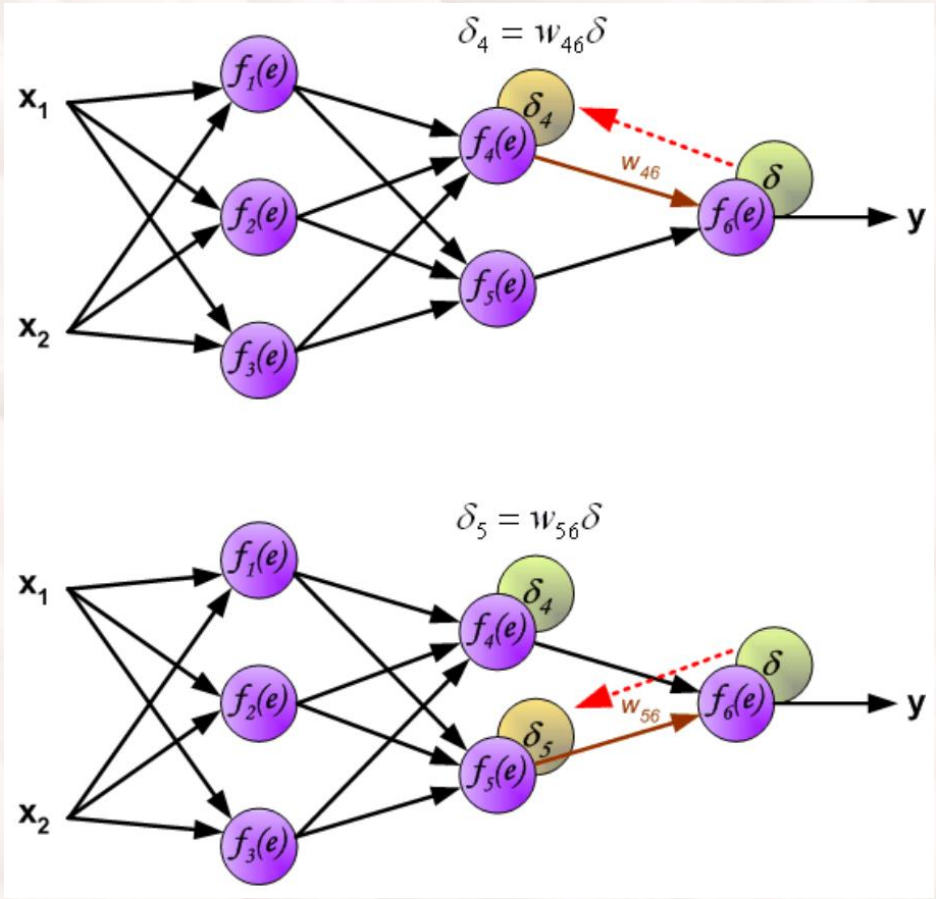
Przechodząc z błędem wstecz z trzeciej warstwy do drugiej, błąd jest ważony zgodnie z aktualną wartością wagi połączenia pomiędzy neuronem warstwy 3 i odpowiednim neuronem warstwy drugiej.

Neurony w warstwie drugiej sumują ważone sygnały błędów dochodzących do nich z warstwy trzeciej. Tutaj ze względu na to, iż w warstwie 3 występuje tylko jeden neuron, sumy składają się tylko z jednego członu:

$$\delta_i = \sum_{j=1}^{k} w_{ij}\delta_j$$

$$\delta_4 = w_{46}\delta_6$$

$$\delta_5 = w_{56}\delta_6$$

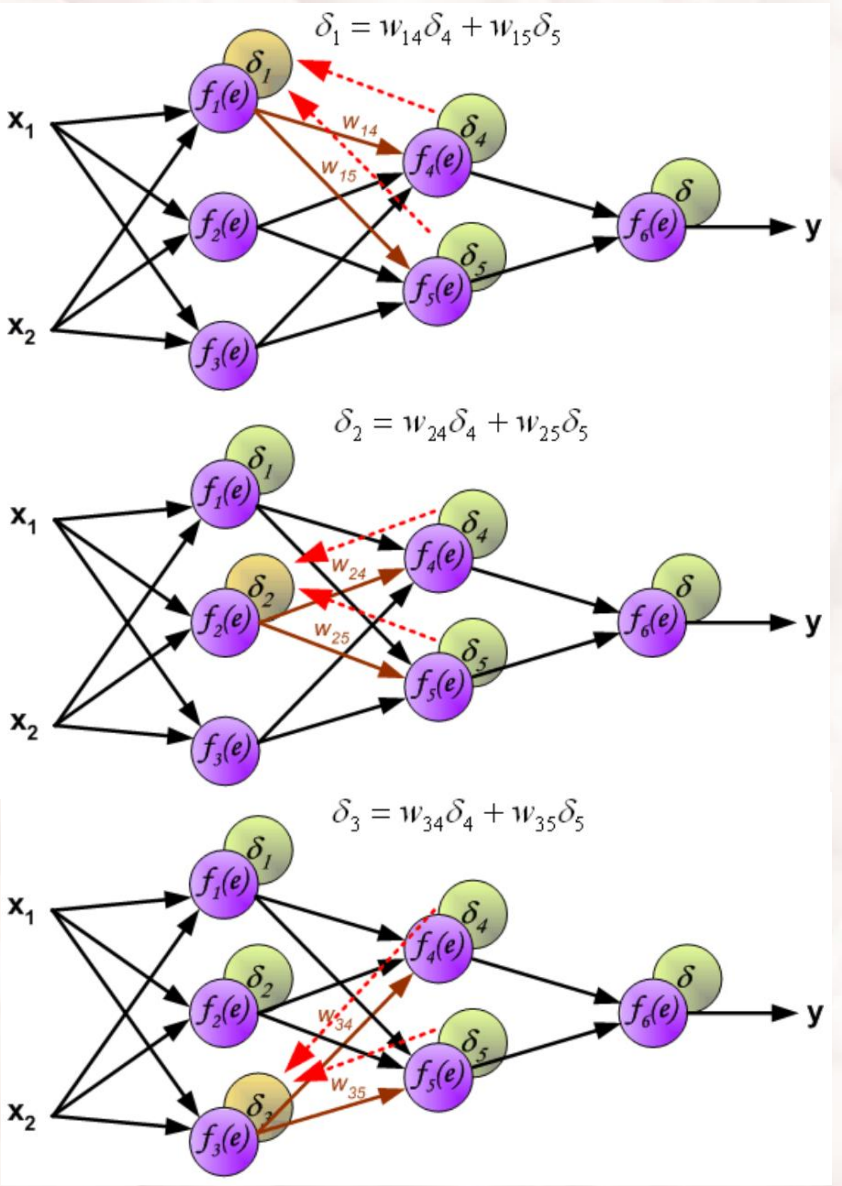


PROPAGACJA WSTECZNA BŁĘDU DO PIERSZWEJ WARSTWY

Przechodząc z błędem wstecz z drugiej do pierwszej warstwy sieci błędy obliczone dla drugiej warstwy (δ_4 i δ_5) są ważone zgodnie z aktualną wartością wag połączeń pomiędzy neuronami warstwy drugiej i pierwszej, a następnie sumowane neuronach warstwy pierwszej zgodnie z następującą zależnością:

$$\delta_i = \sum_{j=1}^{k} w_{ij}\delta_j$$

Następnie dokonywana jest korekta wartości wag sieci.



KOREKTA WAG SIECI W TRAKCIE PROPAGACJI WSTECZNEJ BŁĘDU

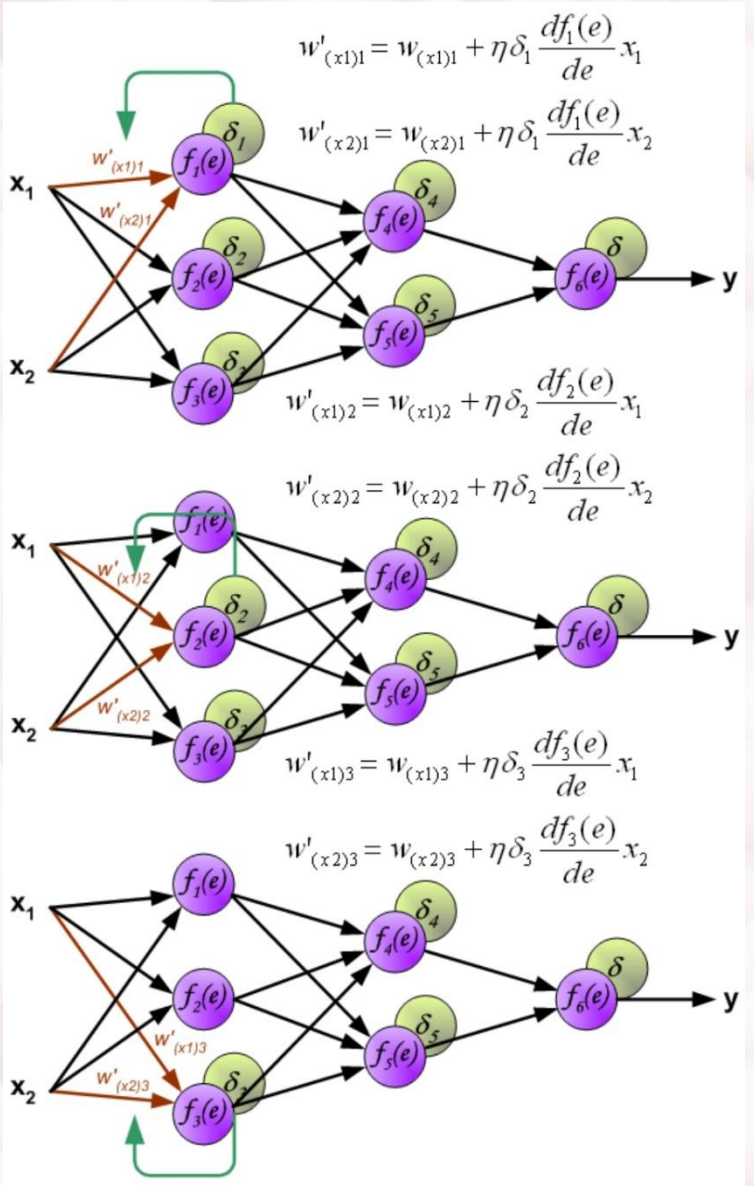
Dokonujemy korekty wag sieci tak, żeby zmniejszyły błąd średniokwadratowy, jaki był obliczony na wyjściu sieci.

W tym celu korzystamy z uogólnionej reguły delta, korzystając z pochodnej cząstkowej funkcji aktywacji oznaczonej jako $df_i(e) / de$.

Korekty wag możemy dokonywać:

od razu dla każdego wzorca uczącego (tzw. *on-line training*)

dopiero po zakończeniu propagacji błędów dla całego zbioru uczącego, sumując je dla wszystkich wzorców, a na końcu obliczając ich średnią (tzw. *off-line training* lub *batch training*).



KOREKTA WAG SIECI W TRAKCIE PROPAGACJI WSTECZNEJ BŁĘDU

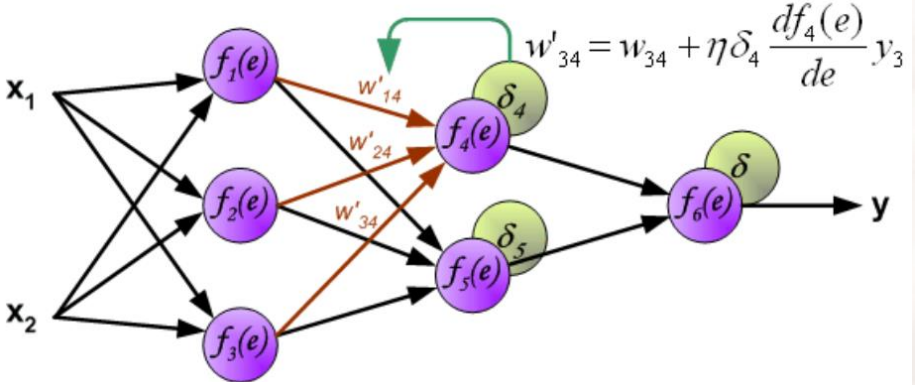
Podobnie dokonujemy korekty wag w drugiej i trzeciej warstwie sieci.

Współczynnik η służy stopniowej adaptacji sieci oraz określa szybkość uczenia się. Na początku jest zwykle duży, a następnie jego wartość jest stopniowo zmniejszana. Warunkuje on możliwość przejścia od minimów lokalnych do minimum globalnego.

$$w'_{14} = w_{14} + \eta\delta_4 \frac{df_4(e)}{de} y_1$$

$$w'_{24} = w_{24} + \eta\delta_4 \frac{df_4(e)}{de} y_2$$

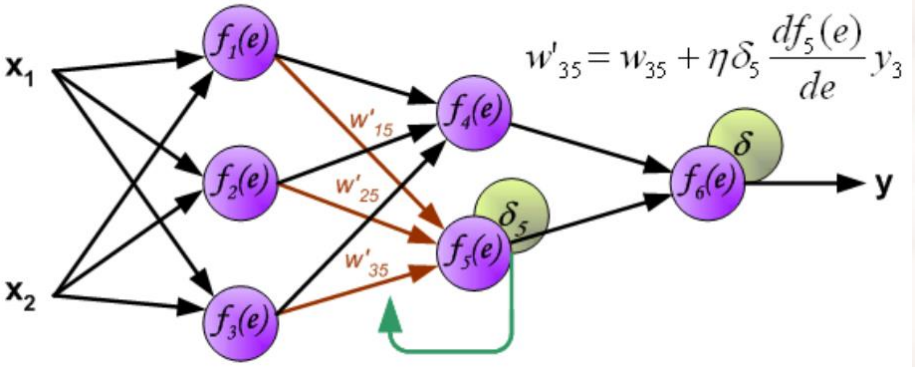
$$w'_{34} = w_{34} + \eta\delta_4 \frac{df_4(e)}{de} y_3$$



$$w'_{15} = w_{15} + \eta\delta_5 \frac{df_5(e)}{de} y_1$$

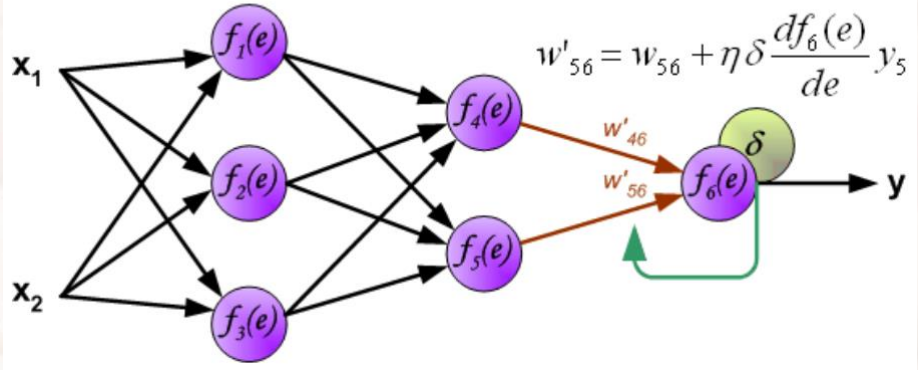
$$w'_{25} = w_{25} + \eta\delta_5 \frac{df_5(e)}{de} y_2$$

$$w'_{35} = w_{35} + \eta\delta_5 \frac{df_5(e)}{de} y_3$$



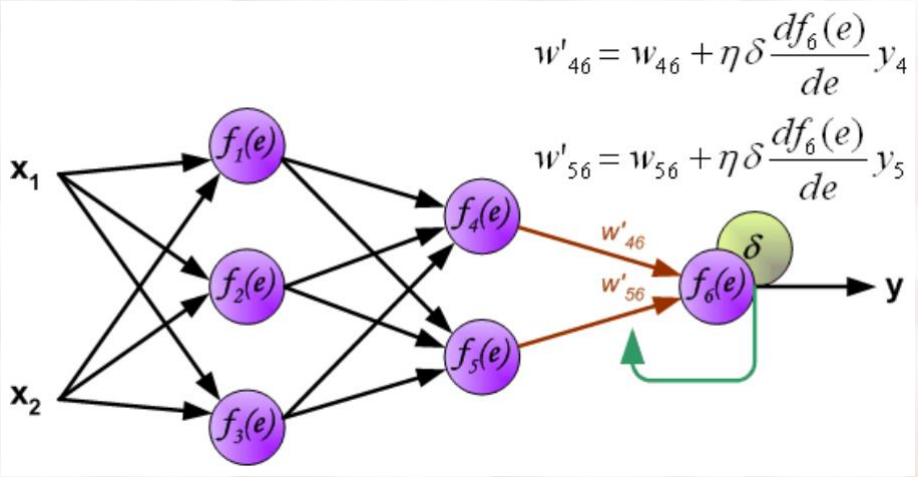
$$w'_{46} = w_{46} + \eta\delta \frac{df_6(e)}{de} y_4$$

$$w'_{56} = w_{56} + \eta\delta \frac{df_6(e)}{de} y_5$$



KOREKTA WAG SIECI W TRAKCIE PROPAGACJI WSTECZNEJ BŁĘDU

Wyznaczenie wartości pochodnej we wzorach na aktualizację wag zależne jest od postaci funkcji aktywacji f.



Dla najczęściej stosowanych funkcji:

sigmoidalnej:

$$f(x) = \frac{1}{1 + e^{-\beta * x}}$$

$$f'(x) = \beta * f(x) * \left(1 - f(x)\right) = \beta * y * (1 - y)$$

tangensa hiperbolicznego:

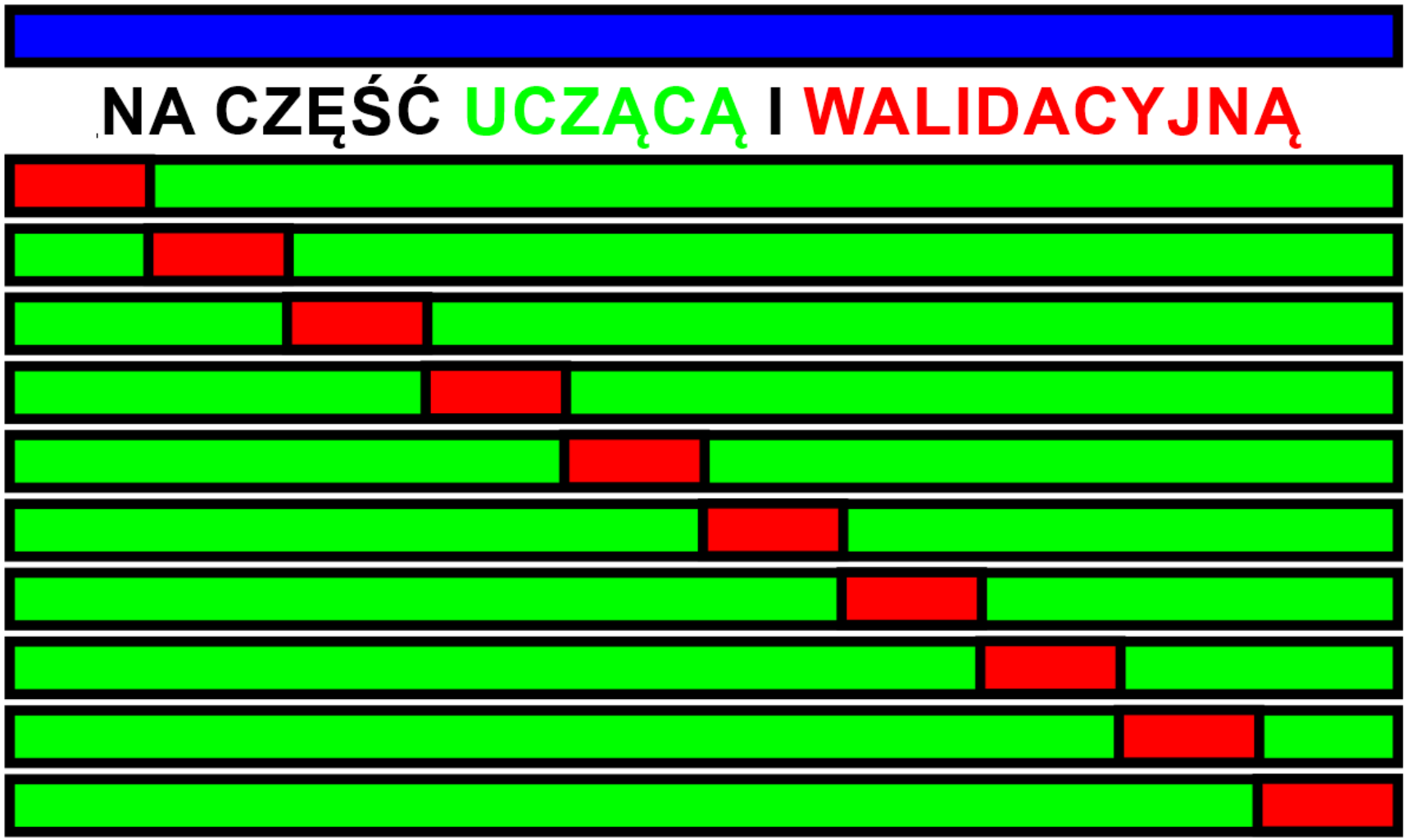
$$f(x) = tgh(\beta * x)$$

$$f'(x) = \beta * \left(1 - tgh^2(\beta * x)\right) = \beta * y\left(1 - y^2\right)$$

Zbuduj pętlę umożliwiając uczenie z wykorzystaniem walidacji krzyżowej

CAŁY ZBIÓR DANYCH DZIELONY JEST

NA CZĘŚĆ UCZĄCĄ I WALIDACYJNĄ



Zbuduj pętlę umożliwiając uczenie z wykorzystaniem walidacji krzyżowej

CAŁY ZBIÓR DANYCH DZIELONY JEST PROPORCJONALNIE

DO LICZNOŚCI KLAS NA CZĘŚĆ UCZĄCĄ I WALIDACYJNĄ