AGH	AKADEMIA GÓRNICZO – HUTNICZA KRAKÓW	Tomasz Kąkol
Sztuczne Sieci Neuronowe		
SSN – lab01, lab02, lab03		
Data wykonania ćwiczenia: 12.03.2018 19.03.2018 26.03.2018	Data złożenia sprawozdania: I termin 14.04.2018 Poprawa 30.04.2018	Ocena:

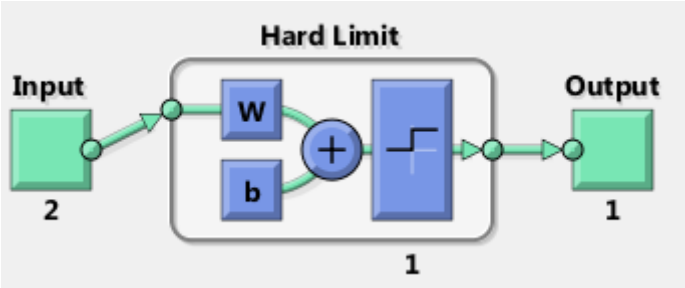
Laboratorium 1

Podczas pierwszych zajęć laboratoryjnych, zajmowaliśmy się uczeniem i testowaniem perceptronu (prostszej sieci neuronowej, składającej się z jednego lub wielu niezależnych neuronów McCullocha-Pittsa), który umożliwia min. stworzenie odwzorowania bramek logicznych typu AND, OR oraz XOR – co było naszym celem. Dla każdego z przypadków funkcji logicznych, jako wejścia służą 2 wektory wejścia. Uczony perceptron powinien wyznaczyć wynik na podstawie 1 z 4 możliwych kombinacji danych wejściowych.

Wejście nr 1	0	0	1	1
Wejście nr 2	0	1	0	1
Wyjście	x	x	x	x

Tab. 1. Ogólna reguła nauczania perceptronu.

Wizualizacja zdefiniowanej struktury sieci w omawianych przypadkach funkcji logicznych:



Rys.1. Zastosowana struktura sieci jako pojedynczy perceptron.

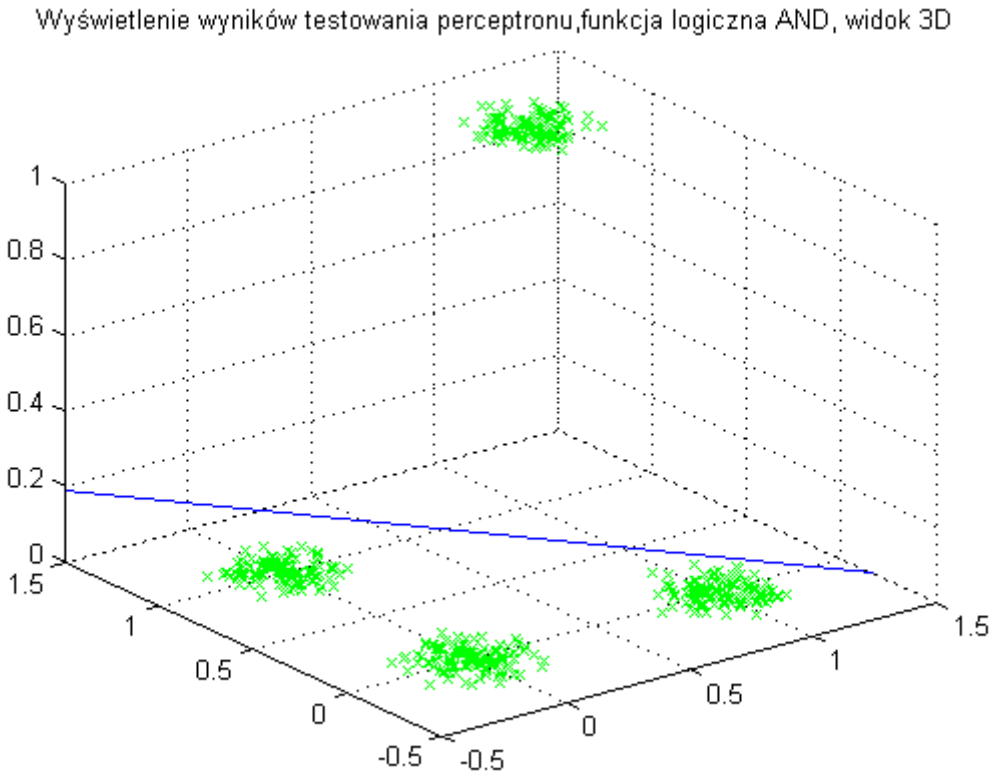
Przypadek 1 – bramka logiczna AND

W celu nauczenia perceptronu reguły logicznej AND, na wejście perceptronu wprowadziłem 50 wartości uczących dla każdej z 4 kombinacji wejścia (łącznie próba 200 wartości uczących). Dla wykonania testu, do wartości podawanych na wejścia perceprtonu dodane zostały losowe zakłócenia, o odchyleniu standardowym równym 0.1 i amplitudzie 1. Dla sprawdzenia skuteczności nauczania, wykonałem zdefiniowanie wektora wejściowego testującego o liczebności 500 wartości (zachowując wartości odchylenia standardowego i amplitudy wprowadzanych zakłóceń), dla których nie są znane wartości na wyjściu perceptronu.

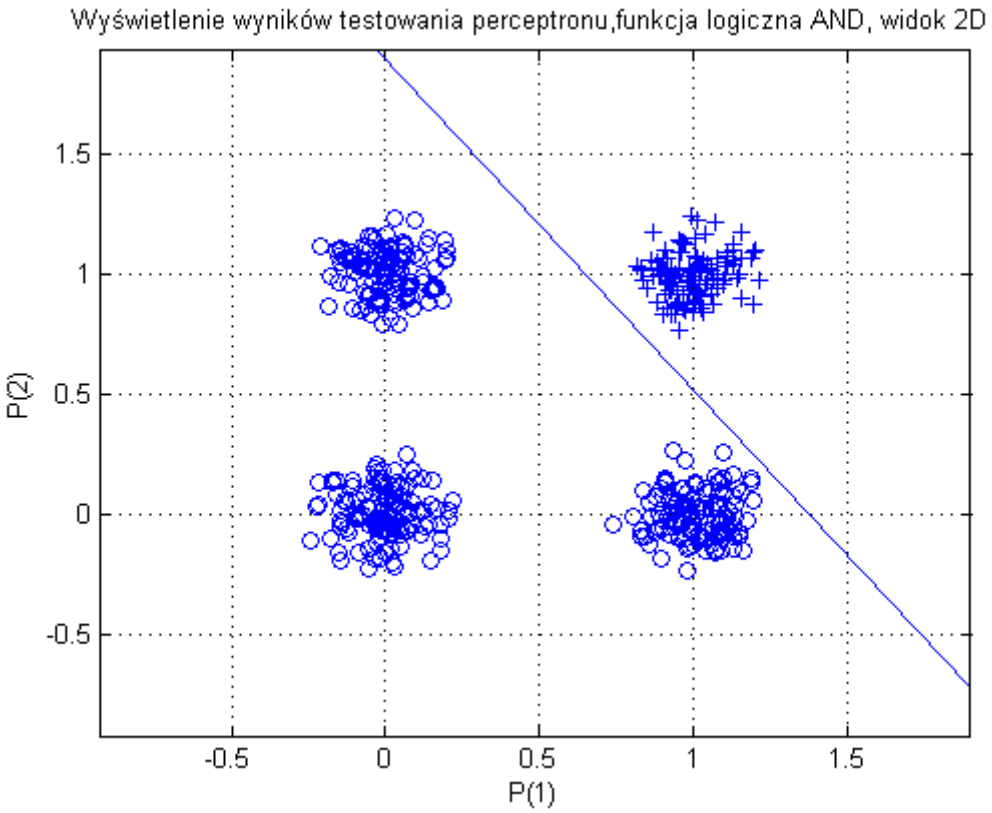
Wejście nr 1	0	0	1	1
Wejście nr 2	0	1	0	1
Wyjście	0	0	0	1

Tab. 2. Reguła nauczania perceptronu – funkcja AND.

Wizualizacja wyników testowania perceptronu:



Rys. 2a. Wizualizacja wyników testowania perceptronu, widok 3D.



Rys. 2b. Wizualizacja wyników testowania perceptronu, widok 2D.

W przypadku, w którym suma iloczynów wartości wejść perceptronu i wag połączeń jest większa od wartość progowej, tj. bias*(-1), wyjście neuronu ma wartość 1. Prosta dzieląca płaszczyznę na dwa obszary jest prostą funkcji aktywacji neuronu. Została ona wyznaczona na podstawie uzyskanych wartości wag i biasu, które w tym przypadku wynoszą odpowiednio:

```
Wartości współczynników:

Wag:     3.6179     2.6357 , Biasu:      -5.00
```

Dla przedstawionego przypadku, wielkość błędu wyznaczenia wartości wyjściowych dla zadanych wartości testowych przez perceptron wyniosła 0%:

```
Wartość błędnego podziału wartości testujących wynosi:     0.0000
```

(Błąd działania perceptronu, wyrażony w procentach, jako iloraz błędnych przypisań danych testowych względem wszystkich danych testowych. W rzeczywistości wartości wektora wyjść nie są znane dla danych testujących !)

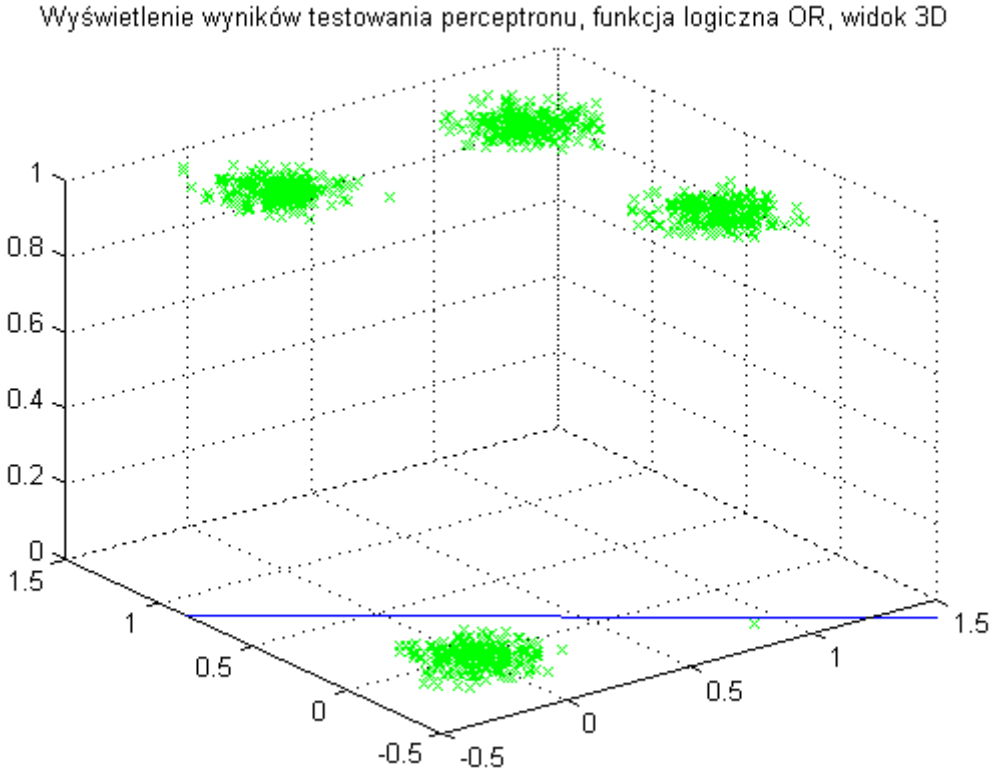
Przypadek 2 – bramka logiczna OR

W celu nauczenia perceptronu reguły logicznej OR, zastosowałem liczebność 100 danych uczących i 1000 danych testujących oraz takie same wartości odchylenia standardowego i amplitudy dla wprowadzanego odpowiednika szumu jak poprzednio. W tym przypadku różnicą w implementacji, w porównaniu z bramką AND, są wyłącznie inne wartości wektora wyjściowego.

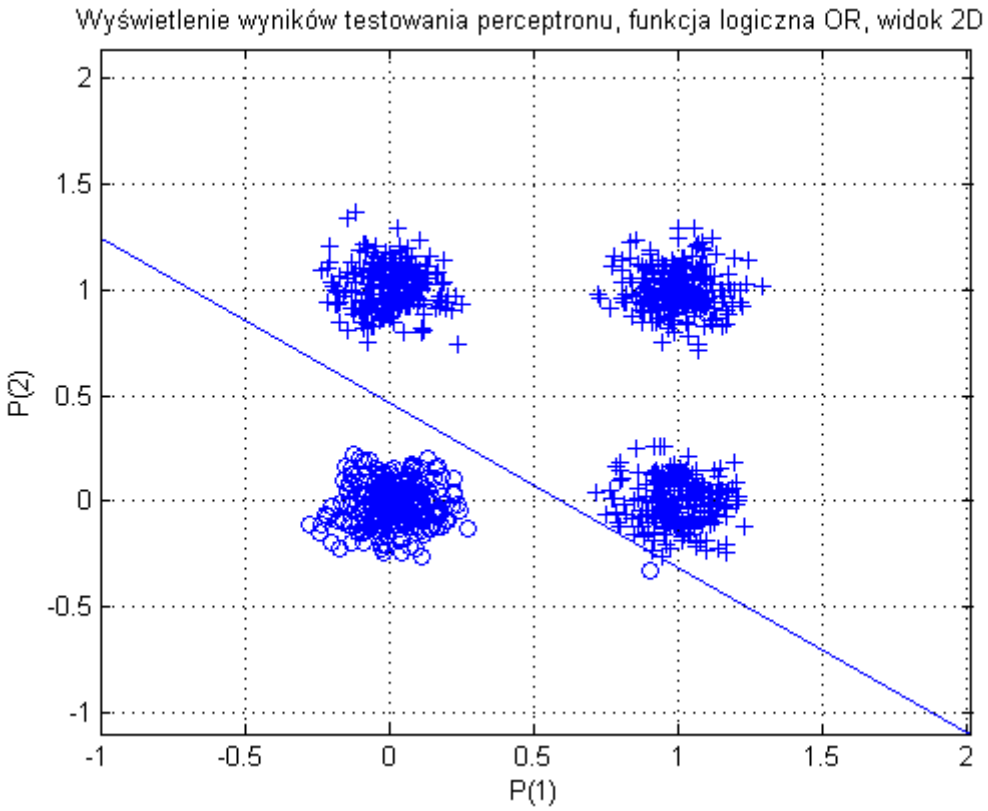
Wejście nr 1	0	0	1	1
Wejście nr 2	0	1	0	1
Wyjście	0	1	1	1

Tab. 3. Reguła nauczania perceptronu – funkcja OR.

Wizualizacja wyników testowania perceptronu:



Rys. 3a. Wizualizacja wyników testowania perceptronu, widok 3D.



Rys. 3b. Wizualizacja wyników testowania perceptronu, widok 2D.

Podobnie jak w poprzednim przypadku, prosta koloru niebieskiego przedstawia prostą progową funkcji aktywacji neuronu. Została ona wyliczona na podstawie uzyskanych wartości wag i biasu.

```
Wartości współczynników:

Wag:     1.6622     2.1375 , Biasu:      -1.00
```

Analogicznie jak we wcześniejszym przypadku (AND), dla przypadku, w którym suma iloczynów wartości wejść perceptronu i wag połączeń jest większa od wartość progowej (tj. bias*(-1), wyjście neuronu ma wartość 1.

Dla przedstawionego przypadku, wielkość błędu wyznaczenia wartości wyjściowych (dla zadanych wartości testowych) przez perceptron wyniosła:

```
Wartość błędnego podziału wartości testujących wynosi:     0.1000%
```

Otrzymany wynik jest zgodny z przedstawionymi wynikami na wykresie powyżej, gdzie 1 dana testowa (z grupy 1000 danych) znajduje się w niepożądanym (przez nas) dla niej obszarze.

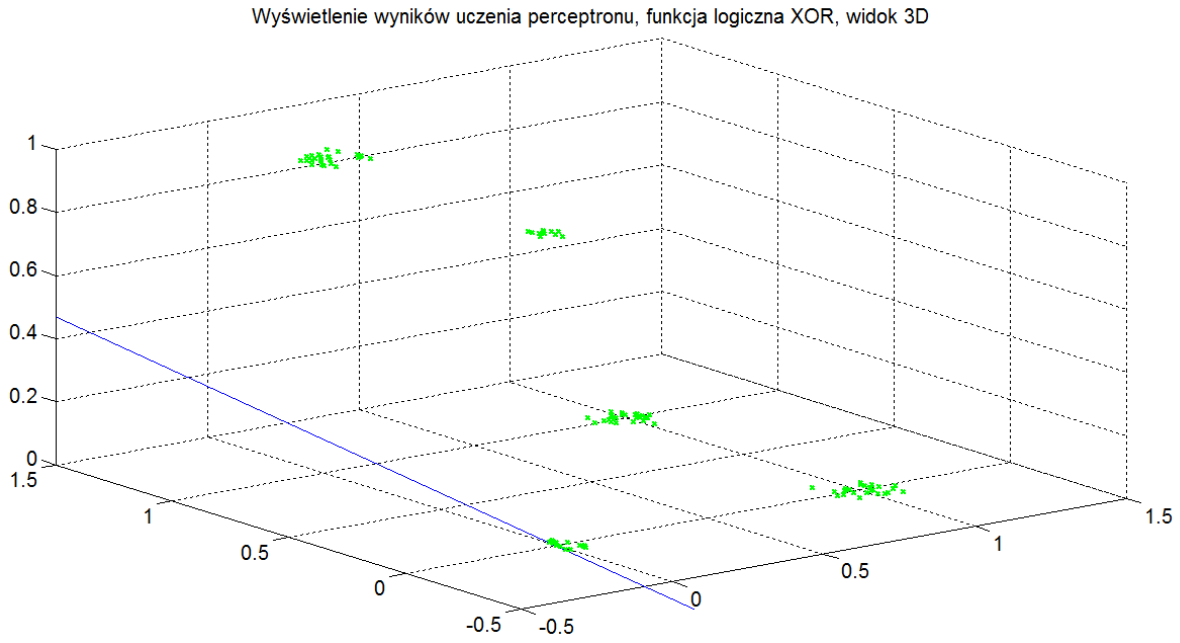
Przypadek 3 – bramka logiczna XOR

W celu nauczenia perceptronu reguły logicznej XOR, zastosowałem liczebność 100 danych uczących i 1000 danych testujących oraz takie same wartości odchylenia standardowego i amplitudy szumu jak poprzednio. Analogicznie, w tym przypadku różnicą są wyłącznie inne wartości wektora wyjściowego.

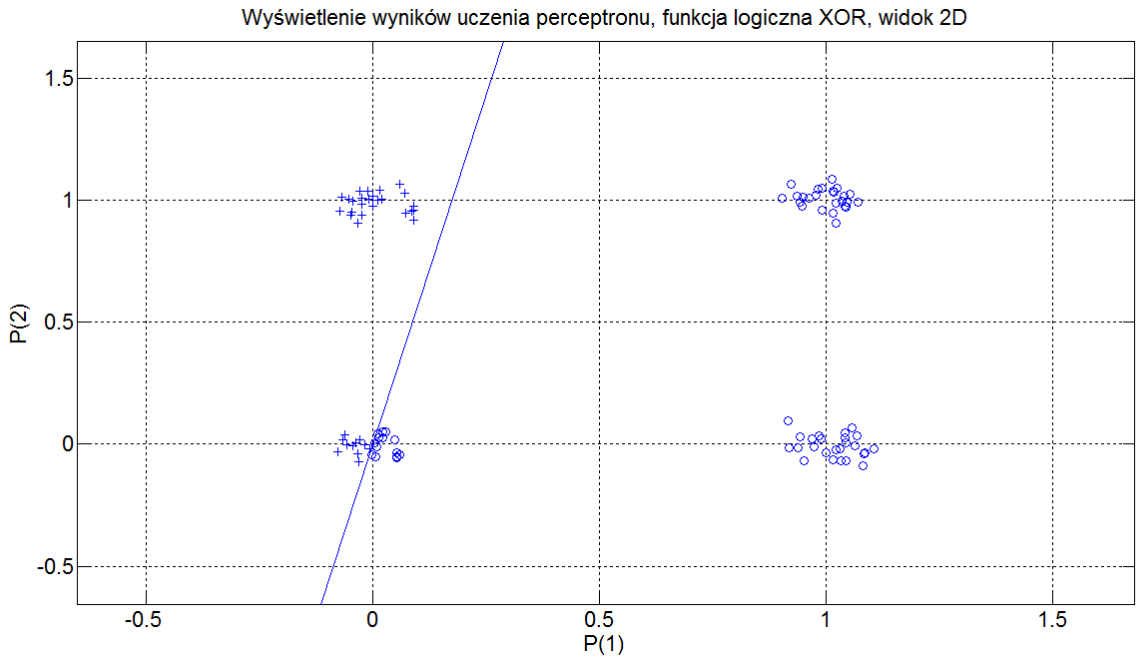
Wejście nr 1	0	0	1	1
Wejście nr 2	0	1	0	1
Wyjście	0	1	1	0

Tab. 4. Reguła nauczania perceptronu – funkcja XOR.

Wizualizacja wyników uczenia perceptronu:



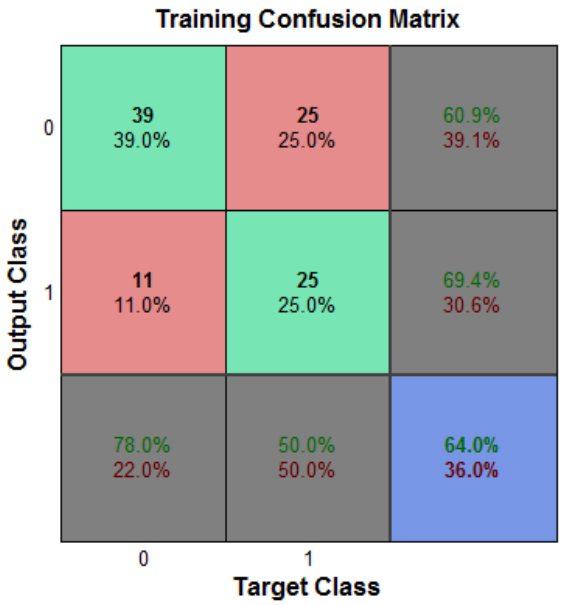
Rys. 4a. Wizualizacja wyników uczenia perceptronu, widok 3D.



Rys. 4b. Wizualizacja wyników uczenia perceptronu, widok 2D.

Wartość błędu uzyskana dla przypadku zastosowana takiej struktury nie jest zadawalająca. Nie jest możliwe wyuczenie pojedynczego perceptronu reguły logicznej XOR. Przyczyną jest wykorzystanie progowej (skokowej) funkcji aktywacji. Nie jest możliwe poprawne rozdzielenie obszarów dla funkcji logicznej XOR z użyciem 1 prostej (progowej). W tym przypadku niezbędne jest zastosowanie min. 2 prostych do podziału obszaru.

```
Wartości współczynników:

Wag:    -1,3433    -0.2337, Biasu:       0.50

Wartość błędnego podziału wartości uczących wynosi:    36 %
```

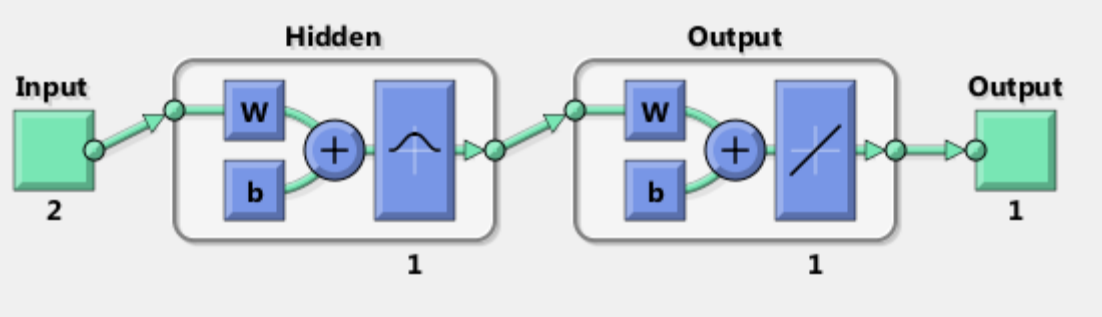


Rys. 4c. Wizualizacja wyników uczenia perceptronu.

Powiedzenie wyznaczonej wartości błędu: Interpretacja wyników przedstawionych na Rys. 4c jest następująca: *64% poprawnych klasyfikacji danych uczących.* W celu stworzenia poprawnej symulacji funkcji logicznej XOR, należy zastąpić wcześniejszą progową funkcję aktywacji na np. gaussowską funkcję aktywacji.

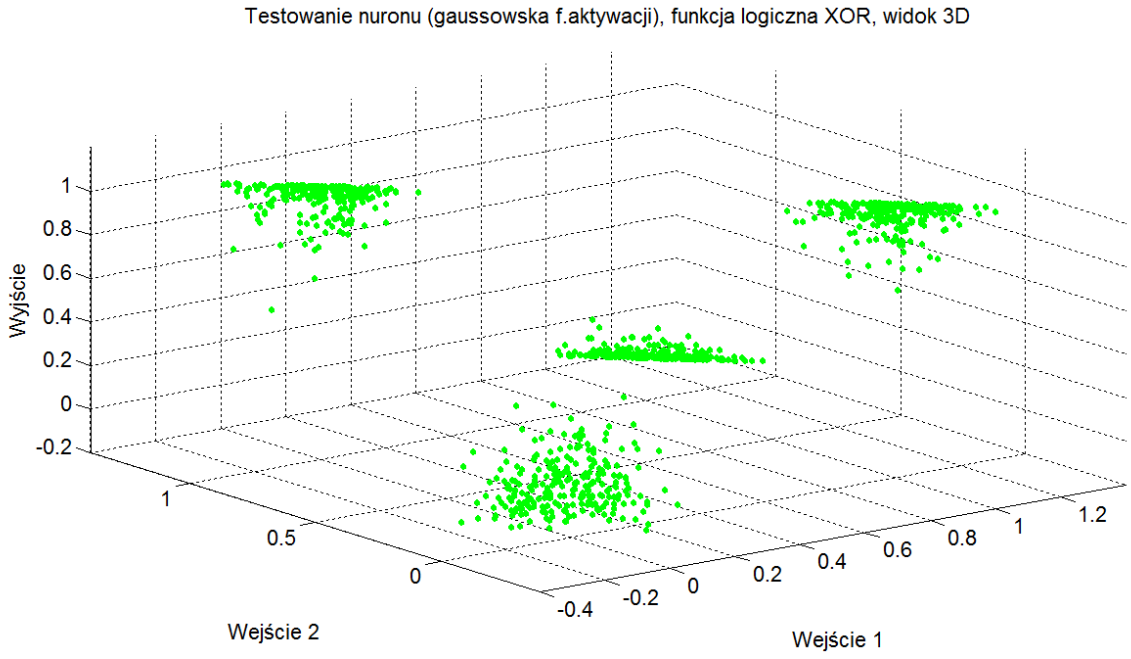
W tym przypadku, dla tak zdefiniowanego neurona, jest możliwe poprawne wyznaczenie 2 obszarów klas dla zdefiniowanych wyjść w tabeli 4. Należy wiedzieć, że zdefiniowana funkcja gaussowska nie zwraca wartości 0 i 1 (takie zwraca progowa funkcja aktywacji), lecz zwrca wartości z przedziału (0, 1). W tym celu otrzymany wektor wyjściowy (dla danych uczących) trzeba pogrupować (z użyciem funkcji skokowej) i przypisać otrzymanym podzbiorom wartości 0 lub 1. Jako wartość funkcji progowej rozdzielającej zbiór wartości wektora wyjściowego należy przyjąć wartość 0.5, ponieważ jest to najbardziej symetryczny podział względem przedziału (0,1). W przypadku, gdy nie zastosujemy progowej funkcji dla podziału wyjściowych wartości na 0 lub 1, wartość błędu wyniesie 100%.

Wizualizacja zastosowanej struktury sieci.



Rys.5. Zastosowana struktura sieci, warstwa ukryta – 1 rząd .

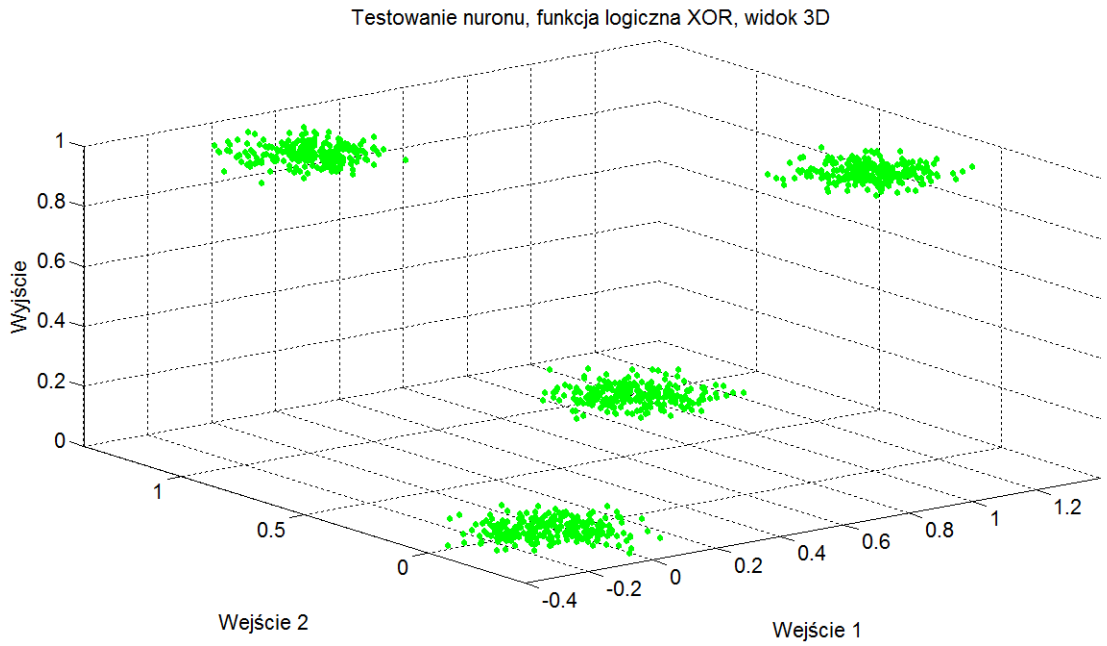
Wizualizacja wyników testowania sieci, przypadek - gaussowska funkcja aktywacji:



Rys. 6a. Wizualizacja wyników testowania neuronu, widok 3D.

```
Wartości współczynników:

Wag:      1.2080      1.2148 , Biasu:       -0.06

Wartość błędnego podziału wartości testujących wynosi:      100%
```

Wizualizacja wyników po wprowadzeniu dodatkowej funkcji progowej dla wartości wektora wyjściowego:



Rys. 6b. Wizualizacja wyników testowania neuronu, widok 3D.

Po wprowadzeniu wyżej omówionej struktury, wartość błędu dla tego przypadku wyniosła 0%:

```
Wartość błędnego podziału wartości testujących wynosi:      0%
```

co jest dla nas wynikiem zadawalającym (udało się zasymulować funkcję logiczną XOR).

Laboratorium 2

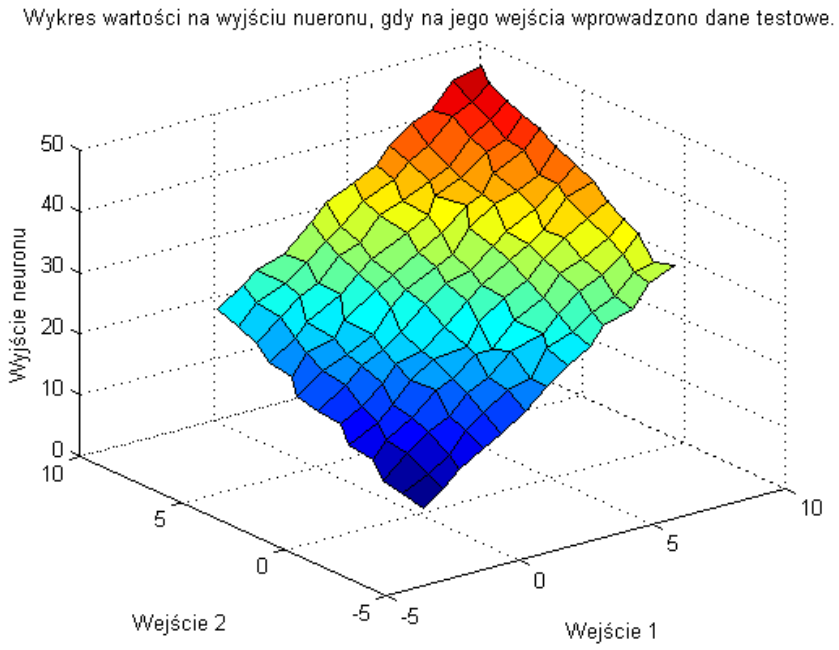
Podczas drugich zajęć laboratoryjnych zadaniem było stworzenie neuronu typu liniowego, posiadającego umiejętność wykonywania prostych obliczeń dodawania i mnożenia; stosując 2 wejścia neuronu, uwzględniając oczekiwanych wartości wag połączeń.

Przypadek 1 – dodawanie

Równanie postaci:

$$3 \cdot x_1 + 2 \cdot x_2 = \theta$$

Oczekiwanym przez nas rezultatem nauczania neuronu jest uzyskanie wag o wartości 3 i 2 dla kolejno wejścia x_1 oraz x_2. W celu przygotowania prostego widoku graficznego, początkowe wartości wektorów wejść x_1 oraz x_2są równe, zawierając się w przedziale (0, 10), z przrostem równym 0.1. Następnie, analogicznie jak podczas laboratorium 1, dane uczące zaszumiano z wykorzystaniem znormalizowanej funkcji randn. Wartość odchylenia standardowego i amplitudy dla zastosowanego szumu to odpowiednio 0.1 i 1. Dla przypadku danych testujących zakres wartości ustawiłem taki sam, ale zmniejszyłem próbę danych testujących, poprzez zwiększenie wartości przyrostu wektorów testujących. Dla zastosowanego typu sieci, powszechnym wskaźnikiem wielkości błędu jest błąd średniokwadratowy(ang. MSE, Mean squared error). Wyniki przeprowadzonego uczenia sieci:



Rys. 7a. Wizualizacja wyników testowania neuronu, przypadek dodawanie.

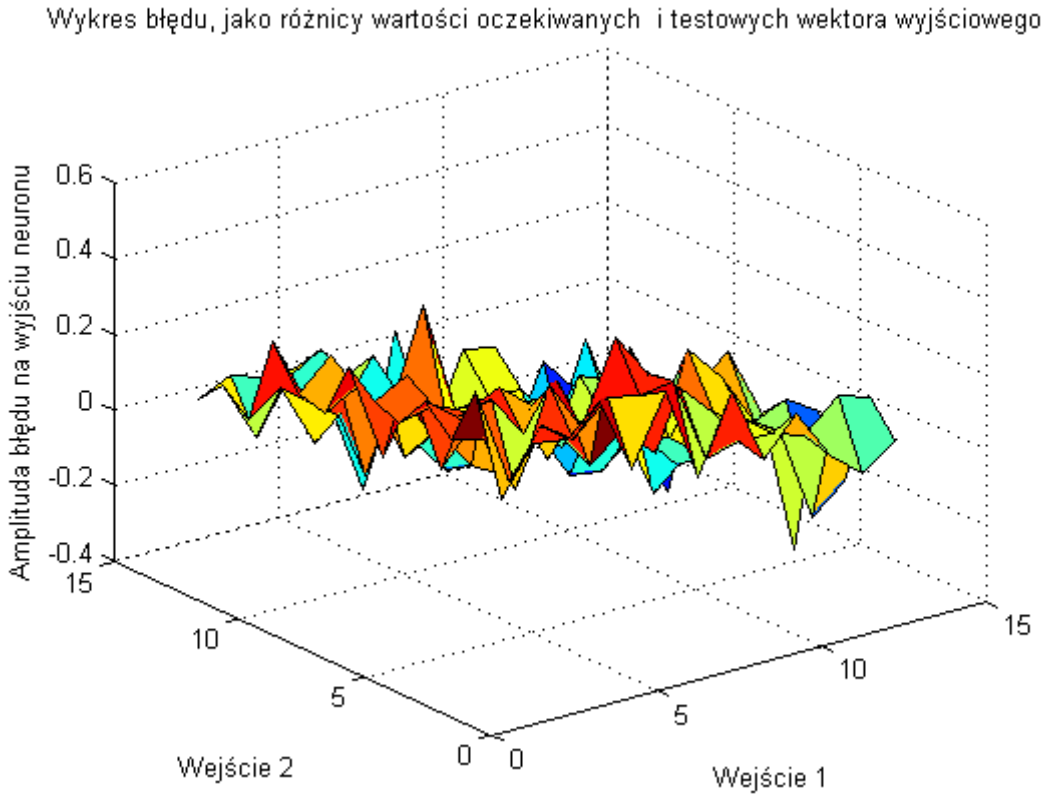
Dla wykonanego przypadku otrzymano następujące wartości wag:

```
Początkowe wartości wag połączeń:

Wagi:      0.1533     -0.9492

Wartości wag połączeń sieci nauczonej:

Wagi:      2.9642      1.9635
```

Co świadczy o tym, że otrzymane wartości są bliskie pożądanym (2.9642 ≈ 3 oraz 1.9635 ≈ 2). Należy pamiętać o wprowadzonym zaszumieniu do wartości.

Wyniki obliczonego błędu metodą MSE dla danych uczących i testujących:

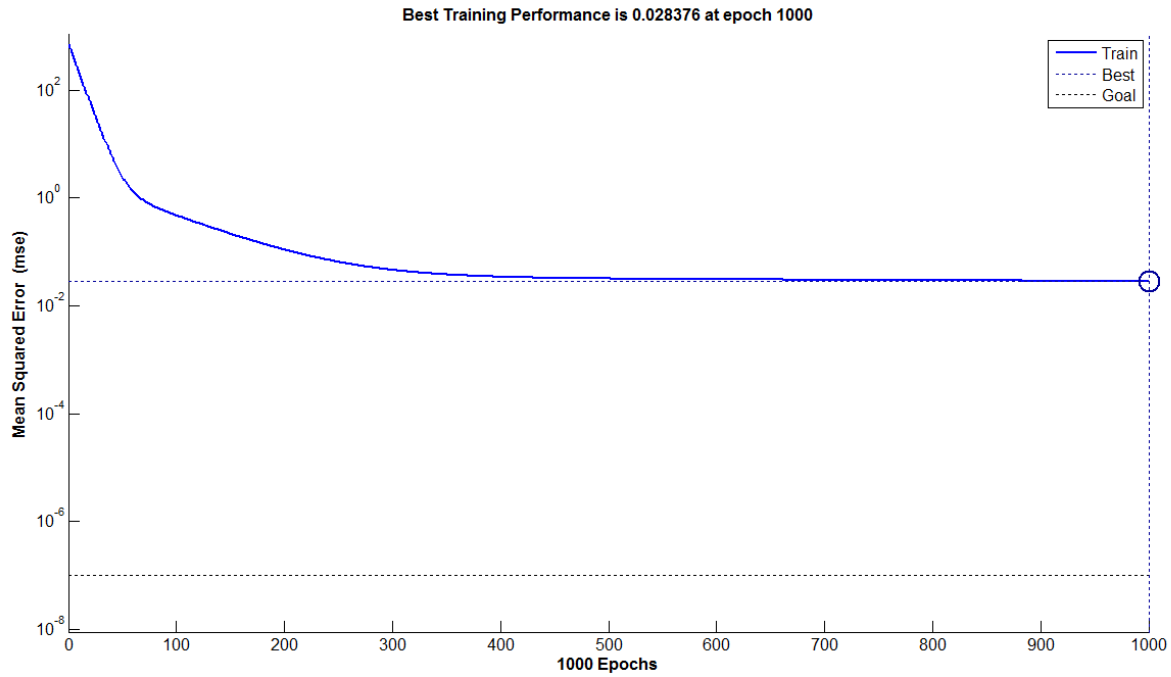
```
Wartość błędu danych uczących na wyjściu neuronu, obliczonego metodą
najmniejszych kwadratów wynosi:

MSE:      0.0368

Wartość błędu danych testujących na wyjściu neuronu, obliczonego metodą
najmniejszych kwadratów wynosi:

MSE:      0.0347
```



Rys. 7b. Wizualizacja wyników błędu na wyjściu neuronu, przypadek dodawanie.

Wartości wyznaczonych błędów są niskie, bliskie dla danych uczących i testujących, zgodnie z oczekiwaniami i akceptowalne.

Podczas wykonanych testów, widoczna jest poprawa minimalizacji wielkości błędu średniokwadratowego wraz z zwiększoną liczbą epok podczas testowania. Dużą zmienność wyników uzyskuje się poprzez edycję parametru learning rate - wydajność algorytmu jest bardzo wrażliwa na właściwe ustawienie szybkości uczenia się. Jeśli tempo uczenia się posiada wartość zbyt dużą, algorytm może oscylować i stać się niestabilny. Jeśli szybkość uczenia się jest zbyt mała, algorytm potrzebuje zbyt wiele czasu (epok), aby prawidłowo pogrupować dane podawane na wejścia. Ustalenie optymalnej wartości szybkości uczenia się przed treningiem nie jest w praktyce możliwe. W rzeczywistości optymalne ustawienie wartości szybkości uczenia się można otrzymać podczas procesu treningowego. Otrzymane wartości błędów są dla nas zadawalające, wynikają z wprowadzonych zakłóceń danych uczących i testujących.



Rys. 7c. Wizualizacja wyników błędu MSE w zależności od ilości epok, przypadek dodawanie. Wartość błędu zgodna z obliczeniami.

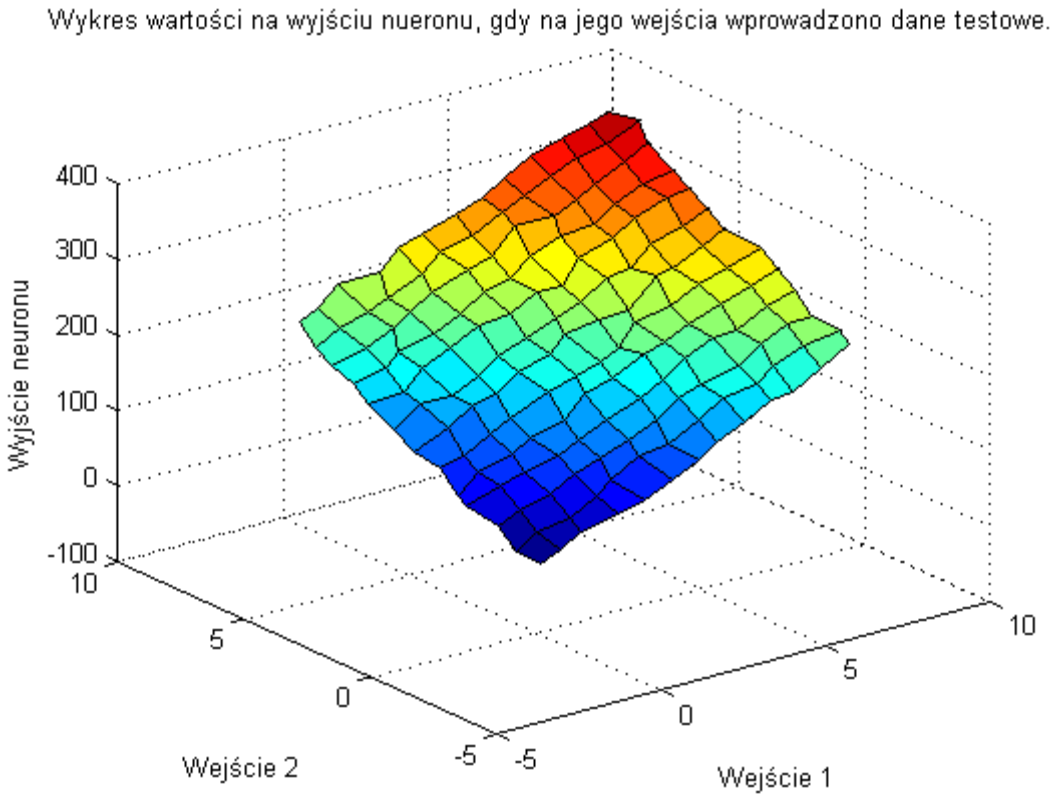
Przypadek 2 – mnożenie

Równanie postaci:

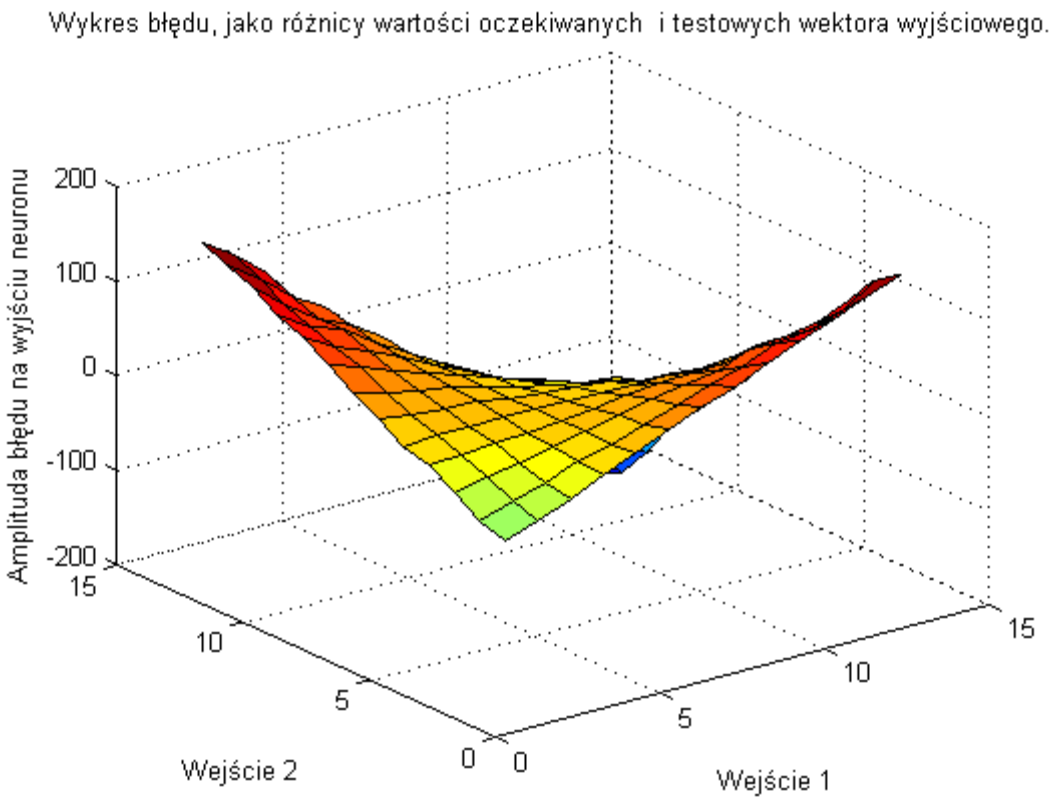
$$3 \cdot x_1 \cdot 2 \cdot x_2 = \theta$$

W przypadku mnożenia, analogicznie jak w punkcie 'Dodawanie', wygenerowano wektory wejściowe i wyjściowy, uczące oraz testujące. W tym przypadku jednak (wykonywania operacji mnożenia) nie ma znaczenia konkretne ustawienie współczynników przed zmiennymi x_1 i x_2, ponieważ mnożenie jest operacją przemienną. Może uprościć równanie do postaci:

$$6 \cdot x_1 \cdot x_2 = \theta$$



Rys. 8a. Wizualizacja wyników testowania neuronu, przypadek mnożenie.



Rys. 8b. Wizualizacja wyników błędu na wyjściu neuronu, przypadek mnożenie.

```
Początkowe wartości wag połączeń:

Wagi:    -0.9922    -0.6448

Wartości wag połączeń sieci nauczonej:

Wagi:    18.6484    18.6994
```

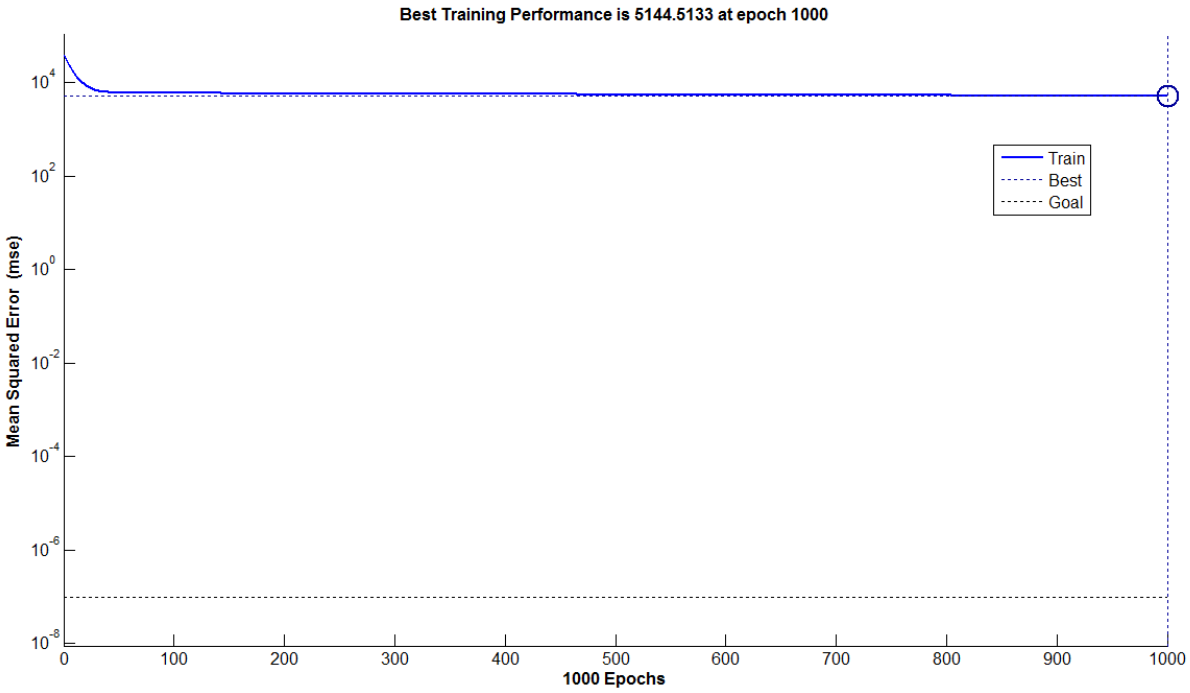
Obliczenie błędu metodą MSE dla danych uczących:

```
Wartość błędu danych uczących na wyjściu neuronu, obliczonego metodą
najmniejszych kwadratów wynosi:

MSE dla treningu:  5149.3529

Wartość błędu danych testujących na wyjściu neuronu, obliczonego metodą
najmniejszych kwadratów wynosi:

MSE dla testowania:  5273.6474
```

Podczas wykonanych testów, widoczna jest słaba poprawa minimalizacji wielkości błędu średniokwadratowego wraz z zwiększoną liczbą epok podczas testowania. Wartości błędów wyniosły odpowiednio:

```
MSE dla treningu:  5149.3529

MSE dla testowania:  5273.6474
```

Wartość błędu średniokwadratowego jest tym mniejsza, im korelacja wartości wektorów wejściowych jest większa (tzn. wartości bardziej zbliżone). Gdy wektory wejściowe były takie same, to wartości wag obu wejść byłyby sobie równe.



Rys. 8c. Wizualizacja wyników błędu MSE w zależności od ilości epok, przypadek mnożenie. Wartość błędu zgodna z obliczeniami.

Wniosek końcowy: Zgodnie z przypuszczeniami, zastosowana sieć nie jest w stanie rozróżnić wektorów wejściowych w celu wyznaczenia oczekiwanych wag połączeń, które wyniosły odpowiednio:

```
Wagi:     18.6484     18.6994
```

co świadczy o tym, że zastosowany neuron typu liniowego nie jest odpowiednim do wykonania mnożenia zmiennych wejść.

Laboratorium 3

Przypadek 1 – Iris

Celem Laboratorium 3 było stworzenie sieci neuronowej, potrafiącej pogrupować gatunki kwiatu irysu, a konkretnie:

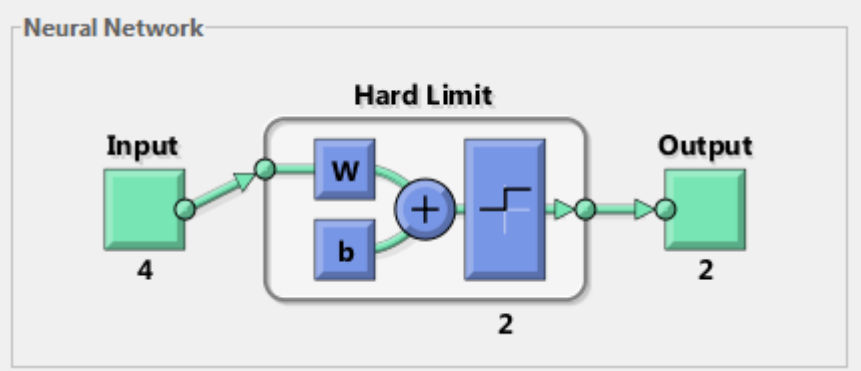
- Iris-setosa, w macierzy irisInputs elementy od 1 do 50.
- Iris-versicolor, w macierzy irisInputs elementy od 51 do 100.
- Iris-virginica, w macierzy irisInputs elementy od 101 do 150.

Posiadających cechy (wektory wejściowe):

- SepalLengthCm - tj. długość działki kielicha,
- SepalWidthCm - tj. szerokość działki kielicha
- PetalLengthCm - tj. długość płatka
- PetalWidthCm - tj. szerokość płatka

W celu prawidłowego pogrupowania danych gatunków kwiatów irysa oraz odpowiedniego nauczenia neuronu, należy zastosować sieć złożoną z minimum 2 perceptronów (ja zastosowałem 2 -podział na 3 klasy), zdolną wykonać taki podział. Aby przetestować wyuczoną sieć, załadowane dane iris_dataset podzielono na 2 grupy, dane uczące (40%) i dane testowe (60%). Dane losowano z każdego gatunku oddzielnie, tak aby z każdego gatunku otrzymać 20 danych uczących i 40 testujących.

Zastosowana struktura sieci w tym przypadku wygląda następująco:



Rys.9. Zastosowana struktura sieci dla kwiatu Irys'a.

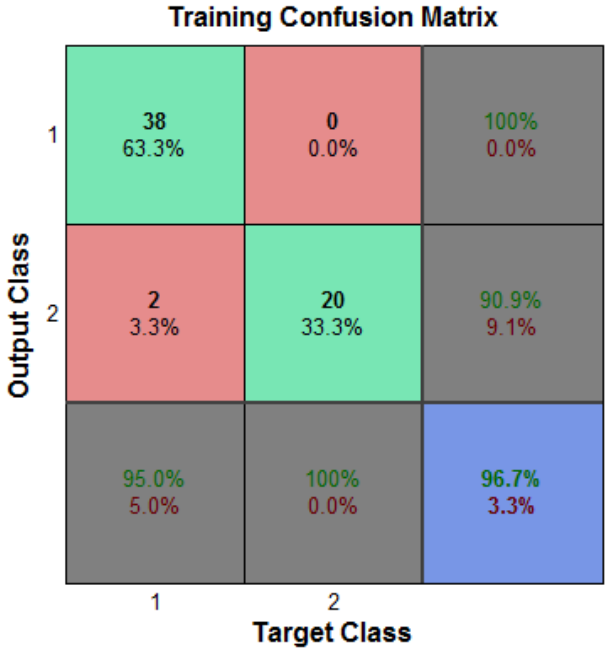
Wartość współczynnika uczenia learning rate ustawiłem 0.00001, a maksymalną liczbę epok podczas treningu na 200. Początkowe wartości wag połączeń zainicjowałem funkcją losową 'rands'.

Wyjścia perceptronów dobrano według następującej kombinacji:

Gatunek	Pierwszy perceptron	Drugi perceptron
IrisSetosa	*1*	*0*
IrisVirginica	*0*	*1*
IrisVersicolour	*1*	*1*

Tab. 5. Reguła nauczania –Irys

Istnieje 4 kombinacja (0 0), lecz dla naszego przypadku 3 klas, wystarczające są 3 kombinacje.



Rys.10. Rezultat uczenia sieci (2-krotny perceptron).

Dla tego przypadku otrzymano 96,7 % skuteczności uczenia (58 na 60 danych uczących poprawnie sklasyfikowanych) .

Dla wyuczonej sieci otrzymano następujące wartości wag połączeń:

```
Wagi połączeń 1-wszego perceptronu:

10.1270    23.1814    -26.0127    -12.0663

Wagi połączeń 2-ego perceptronu:

  -1.8105    -7.7648     8.8461     3.4728
```

Dla przypadku danych testowych (grupa o długości 90 danych) błędnie sklasyfikowane zostały 3 wartości, co daje wartość naszego wskaźnika błędu równą:

```
Błąd:      0.0333   (ilość błędnie sklasyfikowanych danych: 3 na 90)
```

Wniosek końcowy: Dla naszego przypadku prawidłowym (lecz nie jedynym) wyborem struktury sieci była struktura złożona z 2 perceptronów (dwie proste umożliwiają prawidłowe wydzielenie 3 klas, położonych w przestrzeni względem siebie tak jak nasze klasy).

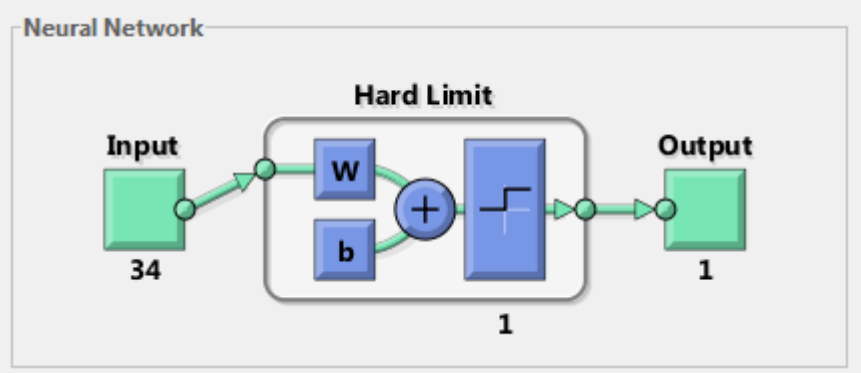
Przypadek 2 – ionosphere

Dla 3 przypadku celem było stworzenie sieci neuronowej, potrafiącej dobre lub złe powroty radaru (2 klasy). Zbiór danych zawiera 34 cech, które zostały wygenerowane z fazy składającej się z 16 anten wysokiej częstotliwości

Dane wejściowe: X <351x34 double>to jest 34 cech.
Dane wyjściowe: y <351x1 cell> po przekształceniu to jest 2 klasy.

W celu prawidłowego pogrupowania danych i odpowiedniego nauczenia neuronu, należy zastosować sieć złożoną z minimum 1 perceptronów (1 perceptron to jeszcze nie sieć, ja zastosowałem 1 - podział na 2 klasy), zdolną wykonać taki podział. Aby przetestować wyuczoną sieć, załadowane dane ionosphere iterowałem 9 razy, tak aby systematycznie zmieniać wartość wskaźnika stosunku liczebności danych testujących do wszystkich danych. Wyniki otrzymanych błędów przedstawiłem w tabeli poniżej.

Zastosowana struktura w tym przypadku do 1 –ego perceptron:

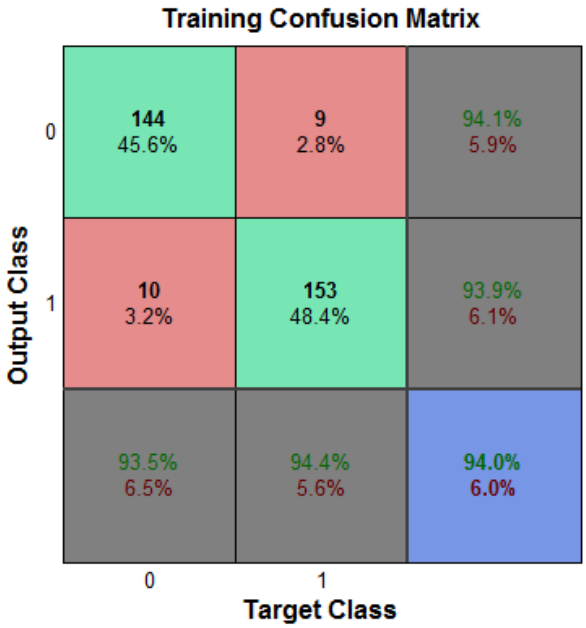


Rys.11. Zastosowana struktura sieci dla danych ionosphere.

D – wskaźnik podziału	Ilość błędnie sklasyfikowanych danych uczących	Dane uczące, MSE	Ilość błędnie sklasyfikowanych danych testowych	Dane testowe, MSE
1/10	140	0.4430	17	0.4857
2/10	135	0.4272	32	0.4571
3/10	137	0.4335	45	0.4286
4/10	134	0.4241	65	0.4643
5/10	116	0.3671	78	0.4457
6/10	96	0.3038	92	0.4381
7/10	80	0.2532	118	0.4816
8/10	61	0.1930	131	0.4679
9/10	19	0.0601	165	0.5238

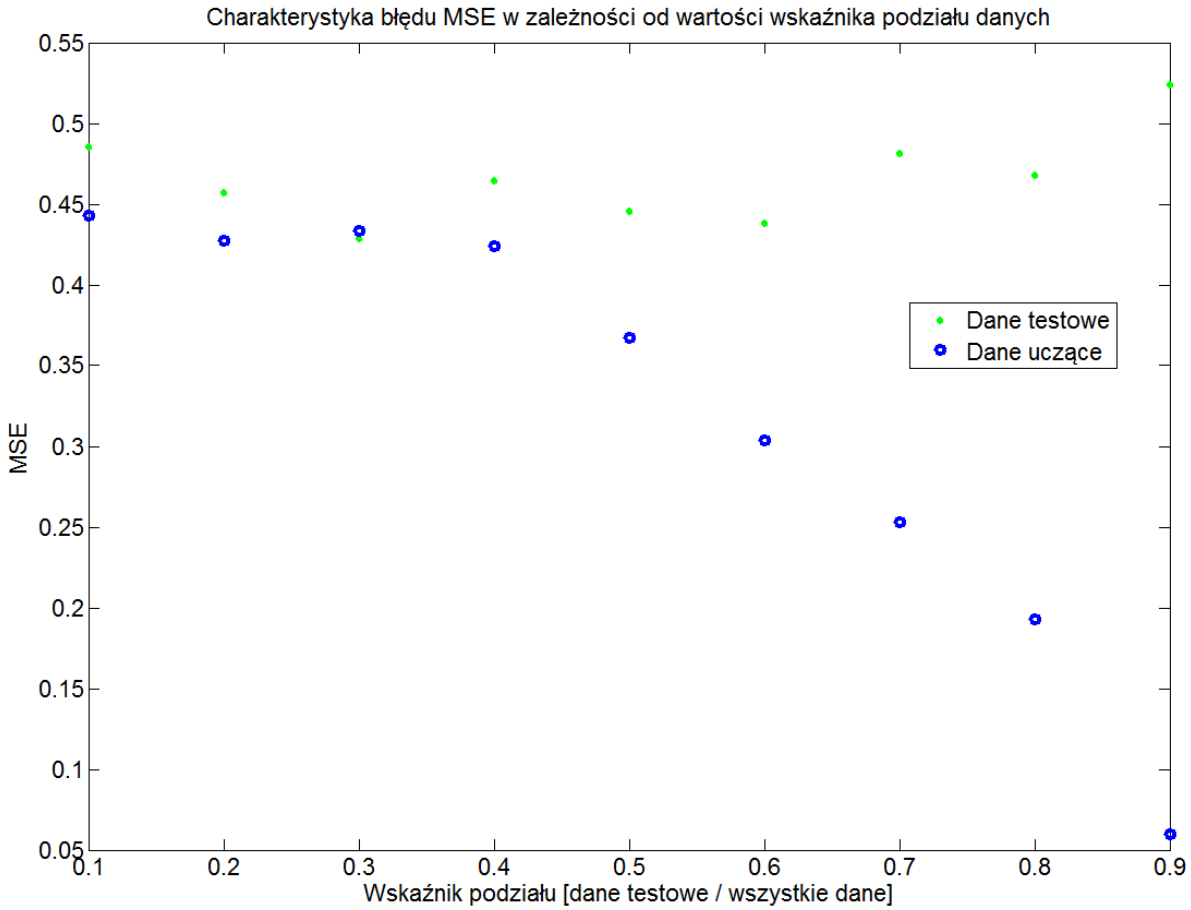
Tab. 6. Wyniki końcowe błędów dla przykładu danych 'ionosphere'.

Wartości wyznaczone za pomocą wzorów znajdują potwierdzenie na automatycznie wygenerowanym przypadku D = 0.9. Dla danych uczących otrzymano 19 błędnych dopasowań na 316, co daje wynik wartość 6% wskaźnika błędu (tj.'błędne' przez 'wszystkie')



Rys.12. Ilustracja potwierdzająca poprawność wzorów do wyznaczenia wartości w Tab.6.

Na podstawie wyznaczonych wartości w tabeli wykreśliłem zależność zmiany wartości wskaźnika błędu od wartości wskaźnika podziału danych. Na podstawie wykresu widoczne jest, że wraz ze wzrostem liczby danych uczących spada wartość błędu MSE wyznaczonego dla wyuczonej sieci i danych uczących. Błąd MSE dla zbioru danych testujących jest w tym przypadku bardziej stabilnie na większym przedziale wskaźnika D, i wzrasta w fazie powyżej D > 0.6. W tym przypadku zastosowano wartość parametru learning rate równy 0.00001. Podobnie jak w laboratorium 2, jeśli tempo uczenia się posiada wartość zbyt dużą, algorytm może stać się niestabilny. Jeśli szybkość uczenia się jest zbyt mała, algorytm potrzebuje zbyt wiele czasu (epok), aby prawidłowo pogrupować dane podawane na wejścia. Tutaj liczba epok była ustawiona na 200. Ustalenie optymalnej wartości szybkości uczenia się przed treningiem nie jest w praktyce możliwe. W rzeczywistości optymalne ustawienie wartości szybkości uczenia się można otrzymać podczas procesu treningowego. Otrzymane wartości błędów są dla nas zadawalające, poniższa charakterystyka była przez nas w przybliżeniu oczekiwaną.



Rys.13. Charakterystyka wskaźnika błędu od wartości wskaźnika podziału zbioru danych.

Przypadek 3 – Wine

Dla 3 przypadku celem było stworzenie sieci neuronowej, potrafiącej pogrupować wina według wytwórni win (3 winnice) ze względu na charakterystykę chemiczną (13 cech).

Dane wejściowe: wineInputs <13x178 double>, to jest 13 cech.
Dane wyjściowe: wineTargets <3x178 double>, to jest 3 klasy.

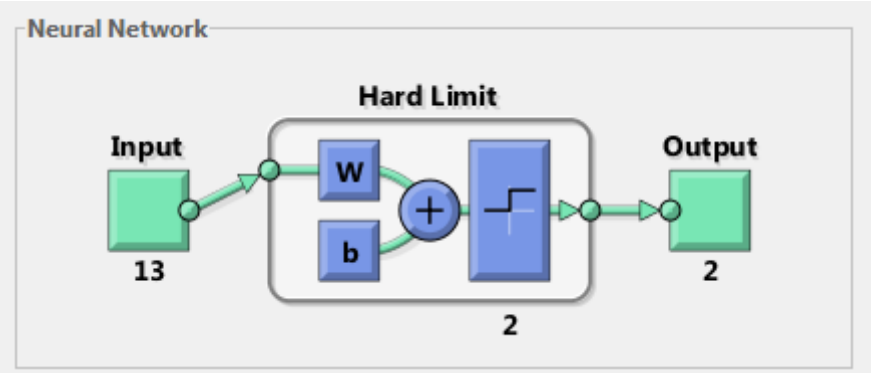
Liczba wystąpień każdej klasy jest następująca:

(źródło:https://www.kaggle.com/brynja/wineuci)

- Klasa 1 = 59 wystąpień,
- Klasa 2 = 71 wystąpień,
- Klasa 3 = 48 wystąpień,

W celu prawidłowego pogrupowania danych i odpowiedniego nauczenia neuronu, należy zastosować sieć złożoną z minimum 2 perceptronów (ja zastosowałem 2 -podział na 3 klasy), zdolną wykonać taki podział. Aby przetestować wyuczoną sieć, załadowane dane wine_dataset podzieliłem tak jak w przypadku 2 na 2 grupy, w zmiennym stosunku dane testowe do ogółu co 10% danych.

Zastosowana struktura w tym przypadku to połączenie 2 perceptronów::



Rys.14. Zastosowana struktura sieci dla danych Wine

Wyjścia perceptronów dobrano według następującej kombinacji:

Winnica	Pierwszy perceptron	Drugi perceptron
Winnica 1	*1*	*0*
Winnica 2	*0*	*1*
Winnica 3	*1*	*1*

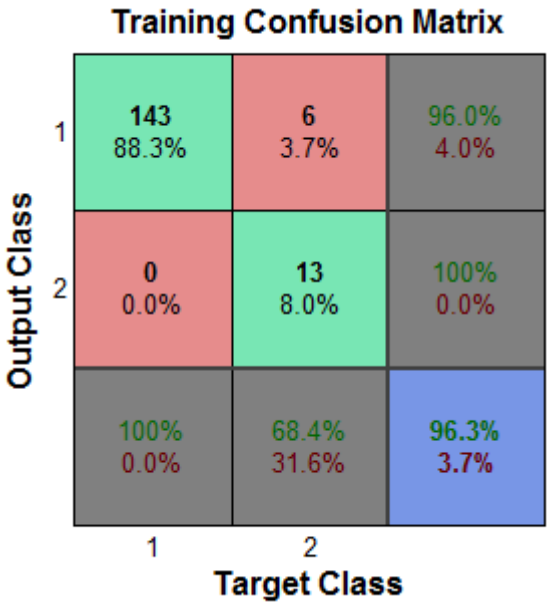
Tab. 7. Reguła nauczania –Wine

Analogicznie jak dla przypadku 2 (dane 'ionosphere') wykonałem porównanie wartości naszego klasyfikatora błędu dla 9 wartości wskaźnika podziału danych testujących do ogółu danych.

D – wskaźnik podziału	Ilość błędnie sklasyfikowanych danych uczących	Dane uczące, MSE	Ilość błędnie sklasyfikowanych danych testowych	Dane testowe, MSE
1/10	97	0.5988	10	0.6250
2/10	80	0.4938	16	0.4706
3/10	84	0.5185	37	0.7115
4/10	79	0.4877	51	0.7286
5/10	65	0.4012	62	0.7045
6/10	35	0.2160	51	0.4857
7/10	25	0.1543	62	0.5041
8/10	5	0.0309	56	0.3972
9/10	6	0.0370	69	0.4340

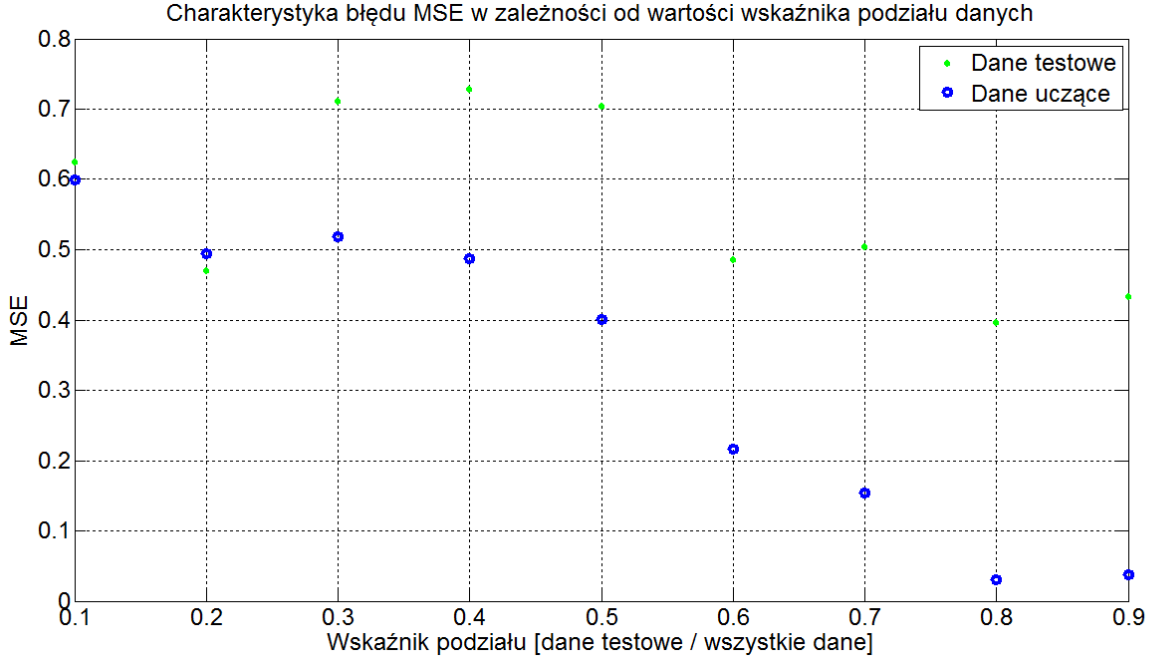
Tab. 8. Wyniki końcowe błędów dla przykładu danych 'wine'.

Wartości wyznaczone za pomocą wzorów znajdują potwierdzenie na automatycznie wygenerowanym przypadku D = 0.9. Dla danych uczących otrzymano 6 błędnych dopasowań na 162, co daje wynik wartość 3,7 % wskaźnika błędu (tj.'błędne' przez 'wszystkie')



Rys.15. Ilustracja potwierdzająca poprawność wzorów do wyznaczenia wartości w Tab.8.

Na podstawie wyznaczonych wartości w tabeli, podobnie jak w przypadku 2, wykreśliłem zależność zmiany wartości wskaźnika błędu od wartości wskaźnika podziału danych. Na podstawie wykresu widoczne jest, że wraz ze wzrostem liczby danych uczących spada wartość błędu MSE wyznaczonego dla wyuczonej sieci i danych uczących. Błąd MSE dla zbioru danych uczących zachował się analogicznie jak w przypadku danych *ionosphere',* tzn dla d>0.5 wartość błędu systematycznie maleje, jest to spowodowane większą ilością danych przeznaczonych do uczenia sieci. W tym przypadku zastosowano wartość parametru learning rate równy 0.00001. Liczba epok była ustawiona na 200. Wnioski dotyczące wartości parametru learning rate są takie same jak wymienione w przykładzie 2.



Rys.16. Charakterystyka wskaźnika błędu od wartości wskaźnika podziału zbioru danych.

Wartości wag połączeń perceptronu dlatego przypadku dla ostatniej iteracji (d=0.9) były następujące:

```
Wartości wag połączeń sieci nauczonej:

Wagi połączeń  perceprtonu 1:

-412.6145    -167.3639      -97.9736  -1154.5401    -2922.2317
-131.3419     -174.6275     -16.1671    -105.4521     158.9964
-42.8803   -172.8228     711.6776

Wagi połączeń  perceprtonu 2:

401.9797     199.5343       79.8191     888.9202     2682.2346
75.7677       77.2691       11.4245      71.2449      28.8304
23.4131    109.8593    -357.0008
```