Sprawozdanie z laboratorium:
Uczenie Maszynowe i Sieci Neuronowe

3 lipca 2013

Prowadzący: dr hab. inż. Maciej Komosiński

Autorzy:	**Maciej Trojan**	inf94378	ISWD	maciek.trojan@me.com
	Paweł Rychły	inf94362	ISWD	pawelrychly@gmail.com

Zajęcia środowe, 11:45.

1 Uczenie nadzorowane sztucznych sieci neuronowych

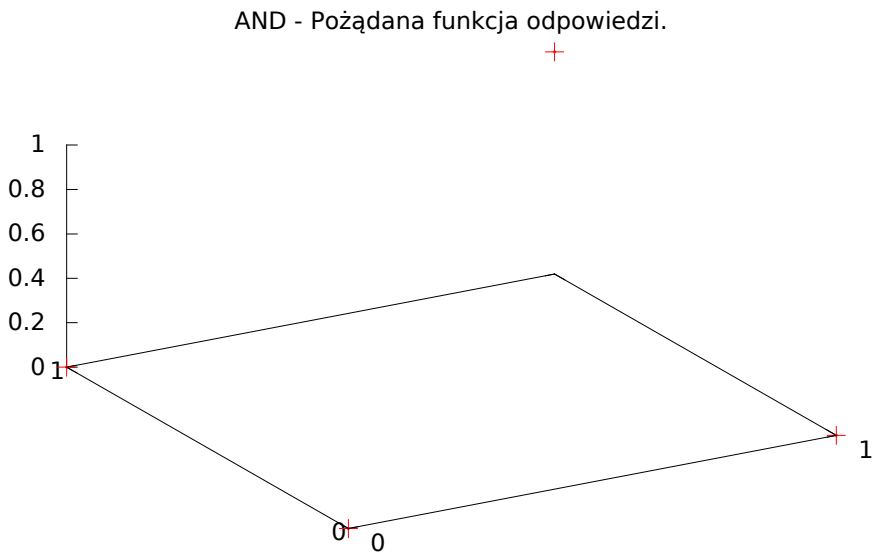
Zadanie 2. Różnice funkcjonalne pomiędzy sieciami jedno- i wielowarstwowymi oraz pomiędzy sieciami liniowymi a nieliniowymi (uczenie sieci warstwowych funkcji logicznej AND i funkcji różnicy symetrycznej XOR)

1. Skonstruuj zbiór przykładów definiujący dwuargumentową funkcję ("bramkę") AND (File—New—Data set) i zachowaj go. Wszystkie 4 przykłady mają stanowić zbiór uczący. Jakie są klasy decyzyjne w tym zbiorze przykładów i jakie są ich liczności?

Zbiór zawiera dwie klasy decyzyjne: 1 i 0. Ich liczności wynoszą odpowiednio: 1 i 3.

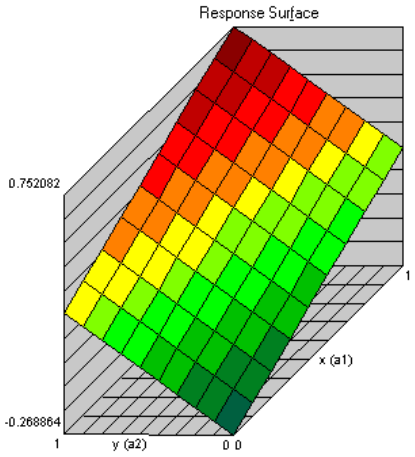
VAR1	VAR2	decyzja
0	0	0
0	1	0
1	0	0
1	1	1

Tabela 1: Zbiór przykładów uczących

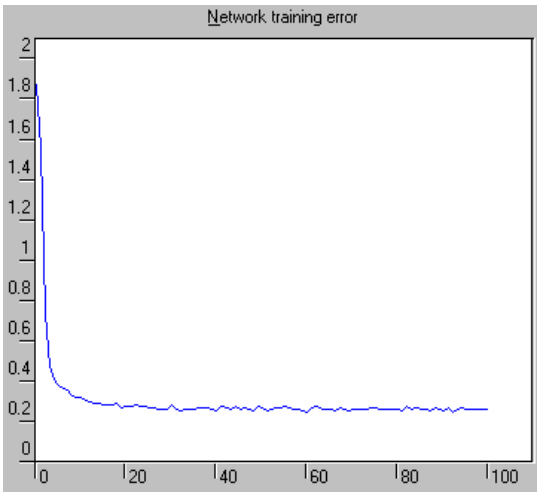


Rysunek 1: Pożądana funkcja odpowiedzi sieci

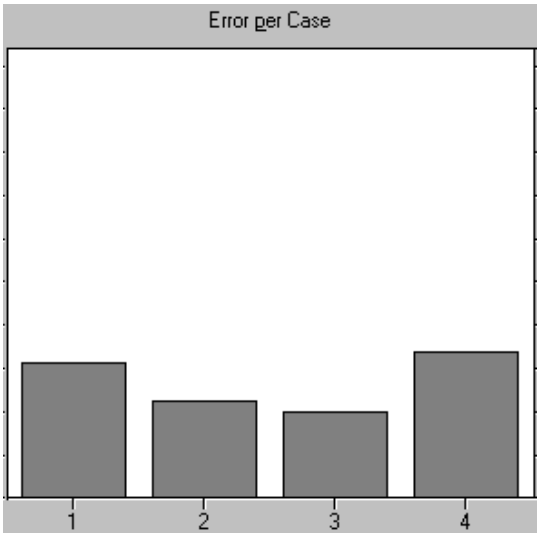
3. Wyobraź sobie (narysuj) pożądaną funkcję odpowiedzi sieci (trójwymiarowy wykres zależności wyjścia od dwóch wejść)



Rysunek 2: Funkcja odpowiedzi sieci



Rysunek 3: Wykres błędu

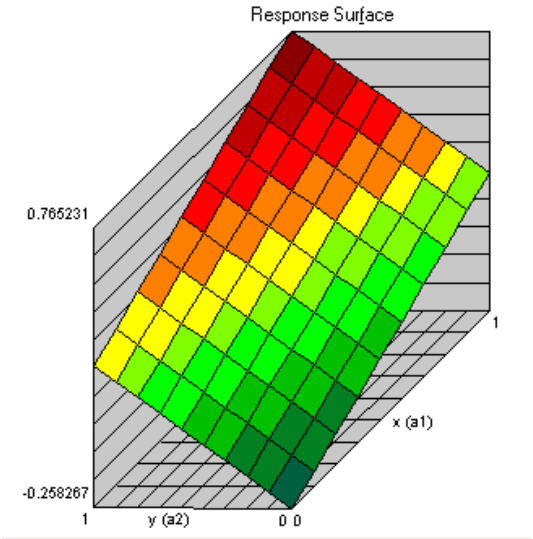


Rysunek 4: Błędy dla poszczególnych przypadków

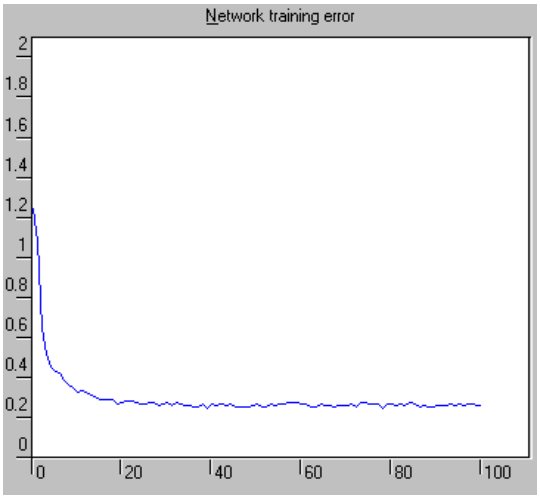
4. Skonstruuj liniową sieć jednowarstwową o architekturze 2-1 (File—New—Network, Type=Linear, przycisk Advise). Uaktywnij okno wykresu błędu średniokwadratowego (Statistics—Training graph). Naucz sieć na problemie AND (Train—Multilayer perceptron—Back propagation). Obejrzyj funkcję odpowiedzi sieci (Run—Responce surface) i błędy dla poszczególnych przypadków (Statistics—Case errors)

5. Spróbuj utworzyć sieć liniową dla problemu AND o liczbie warstw większej niż 2.

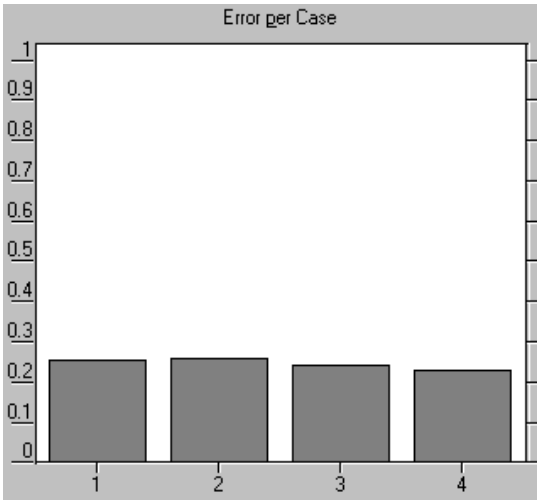
Program Statistica uniemożliwia utworzenie takiej sieci.



Rysunek 5: Funkcja odpowiedzi sieci



Rysunek 6: Wykres błędu

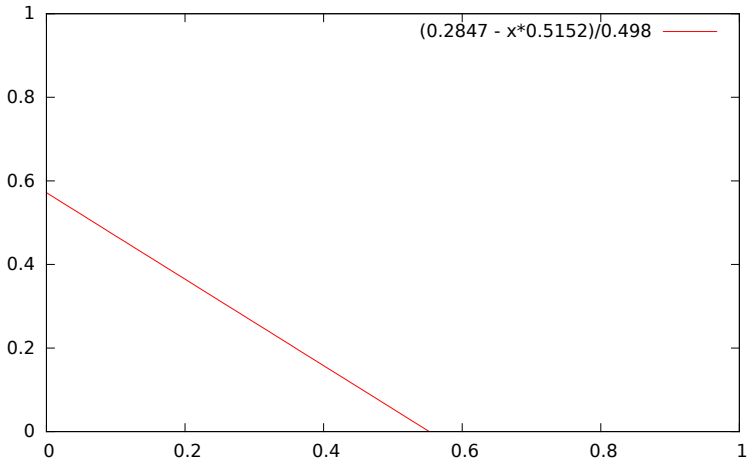


Rysunek 7: Błędy dla poszczególnych przypadków

6. Przerób sieć na nieliniową sieć jednowarstwową (ustawiając Act fn w Edit—Network na Logistic) i naucz ją na tym samym problemie.

7. Wejdź do edytora sieci (Network—Edit) i przypatrz się wagom neuronu wyjściowego. Wykreśl w przestrzeni wejść prostą, którą definiuje neuron wyjściowy i oceń, czy i jak realizuje ona separację klas decyzyjnych.

wagi:
w0 = 0.2847
w1 = 0.5152
w2 = 0.4980



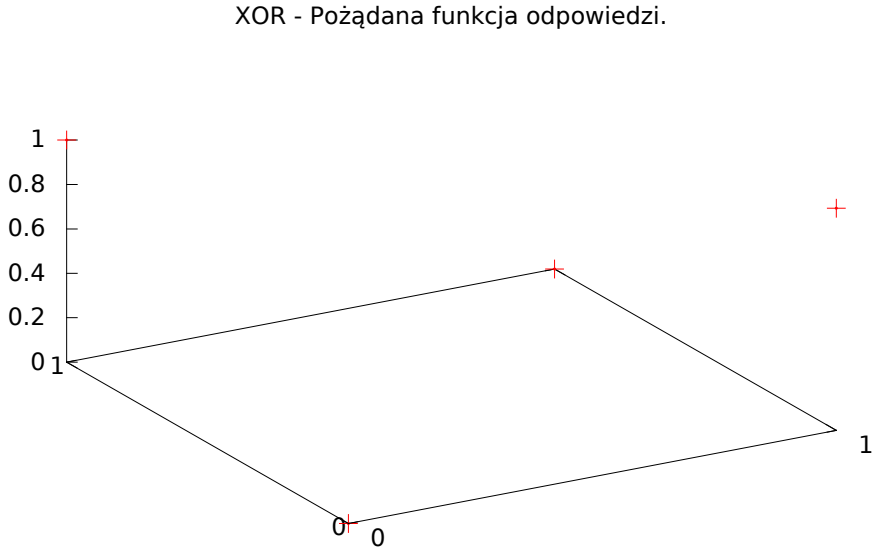
Rysunek 8: Prosta w przestrzeni wejść definiująca neuron wyjściowy

Prosta dokonuje niepoprawnej separacji klas decyzyjnych. Zgodnie z narysowaną prostą przypadki 0, 1 oraz 1, 0 należałyby do klasy 1.

VAR1	VAR2	decyzja
0	0	0
0	1	1
1	0	1
1	1	0

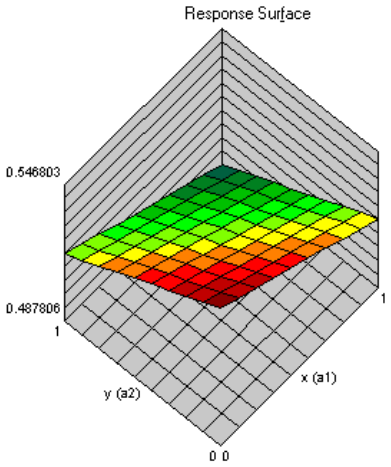
Tabela 2: Zbiór przykładów uczących

8. Skonstruuj zbiór przykładów definiujący dwuargumentową funkcję XOR i zachowaj go. Wszystkie 4 przykłady mają stanowić zbiór uczący.

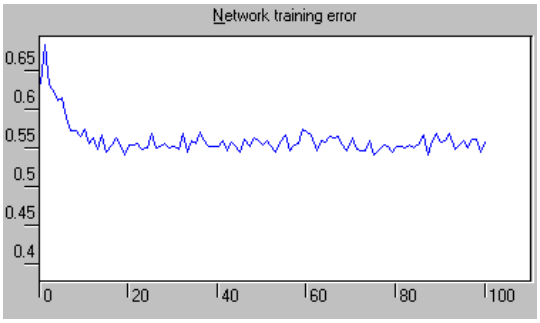


Rysunek 9: Pożądana funkcja odpowiedzi sieci

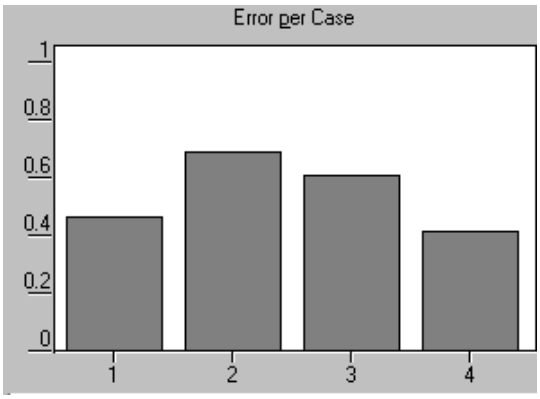
9. Wyobraź sobie (narysuj) pożądaną funkcję odpowiedzi sieci.



Rysunek 10: Funkcja odpowiedzi sieci

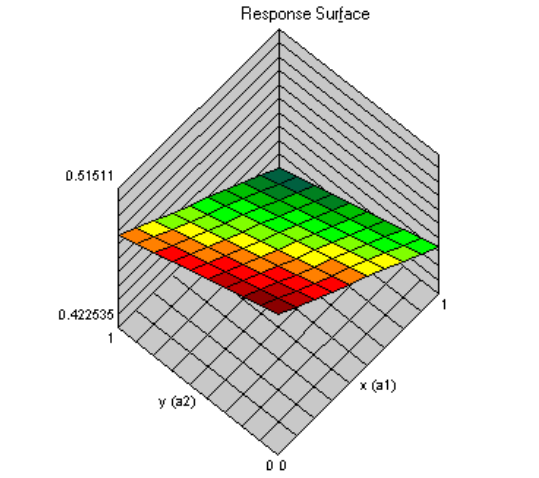


Rysunek 11: Wykres błędu

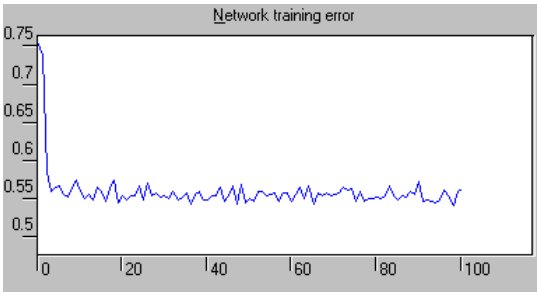


Rysunek 12: Błędy dla poszczególnych przypadków

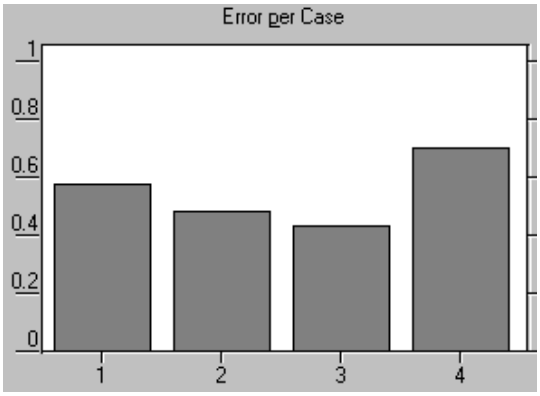
10. Skonstruuj liniową sieć jednowarstwową o architekturze 2-1 i naucz ją na problemie XOR. Obejrzyj funkcję odpowiedzi sieci i błędy dla poszczególnych przypadków.



Rysunek 13: Funkcja odpowiedzi sieci



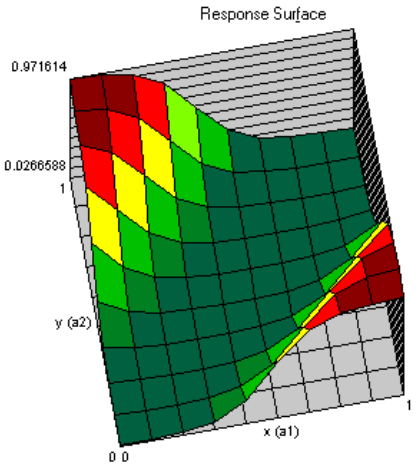
Rysunek 14: Wykres błędu



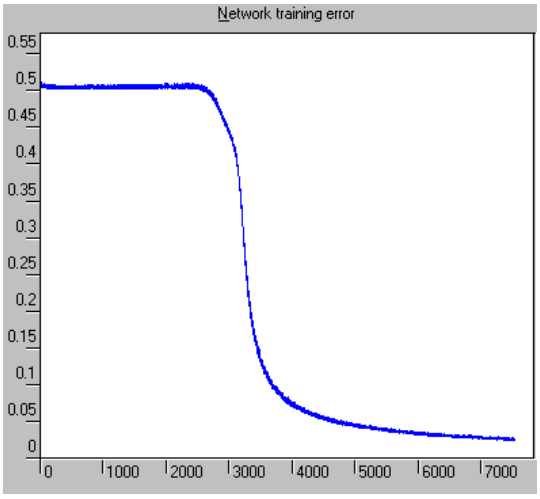
Rysunek 15: Błędy dla poszczególnych przypadków

11. Przerób sieć na nieliniową sieć jednowarstwową i naucz ją na tym samym problemie (zwróć uwagę, jak sieć stara się minimalizować błąd). Obejrzyj, jak zmienia się rozkład wag podczas procesu uczenia (Statistics—Weight distribution).

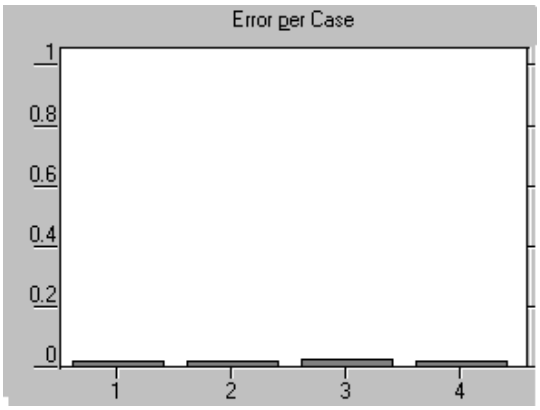
W trakcie procesu uczenia wartości wag zbiegały się do siebie.



Rysunek 16: Funkcja odpowiedzi sieci



Rysunek 17: Wykres błędu

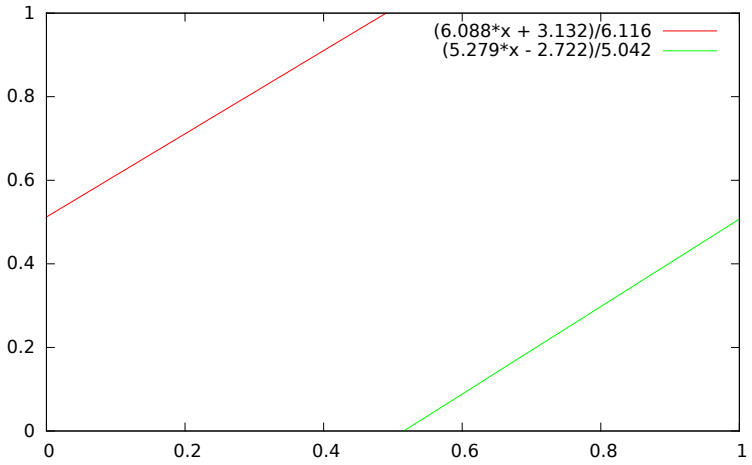


Rysunek 18: Błędy dla poszczególnych przypadków

12. Skonstruuj nieliniową sieć dwuwarstwową o architekturze 2-2-1 (File—New—Network, Type=Multilayer perceptron) i naucz ją na problemie XOR. Obejrzyj funkcję odpowiedzi.

0	h1#01	h1#02
Th	-3.132	2.722
a1	-6.088	-5.279
a2	6.116	5.042

Tabela 3: Wagi neuronów w warstwie ukrytej



Rysunek 19: Proste wyznaczone przez neurony warstwy ukrytej

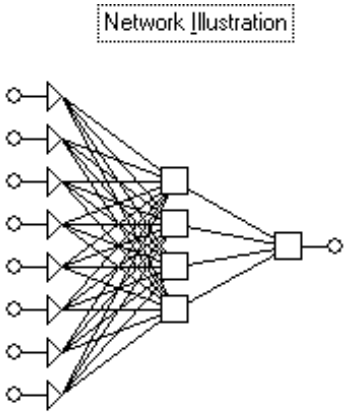
13. Obejrzyj wagi sieci w edytorze sieci (Edit—Network). Jakie proste definiują neurony w warstwie ukrytej (spróbuj je narysować w przestrzeni wejść) ? Jak można interpretować działanie neuronu wyjściowego ?

Neuron wyjściowy agreguje informacje pochodzące z neuronów ukrytych. Wagi neuronów warstw ukrytych definiują płaszczyzny w przestrzeni wejść. Neuron wyjściowy dokonuje klasyfikacji, w oparciu o te płaszczyzny.

14. Obejrzyj, jak zmienia się rozkład wag podczas procesu uczenia. Jaka jest przyczyna takiego zachowania wag i jakie to może mieć konsekwencje (z "informatycznego" punktu widzenia) ?

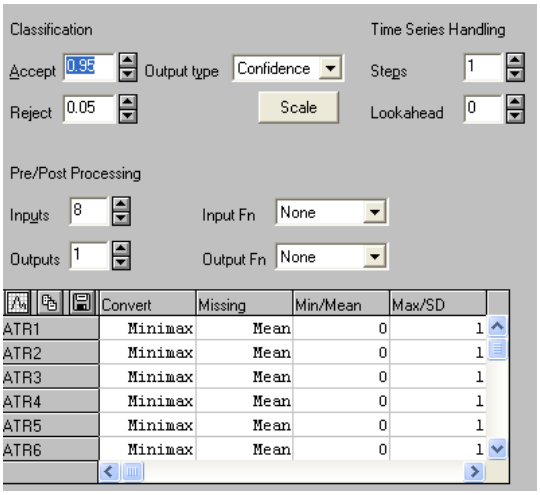
W trakcie uczenia sieci, wartości bezwzględne wag systematycznie rosną. Może to doprowadzić do przekroczenia zakresu liczb.

Zadanie 3. Obserwacja zjawiska przeuczenia na przykładzie zbioru PIMA



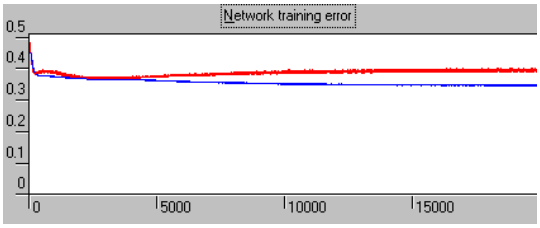
Rysunek 20: Dwuwarstwowa sieć neuronowa o architekturze 8-4-1.

2. Wczytaj zbiór PIMA i skonstruuj dla niego dwuwarstwową sieć nieliniową o architekturze 8-4-1.



Rysunek 21: Okno Pre/Post Processing

3. Obejrzyj ustawienia w okienku Pre/Post Processing.

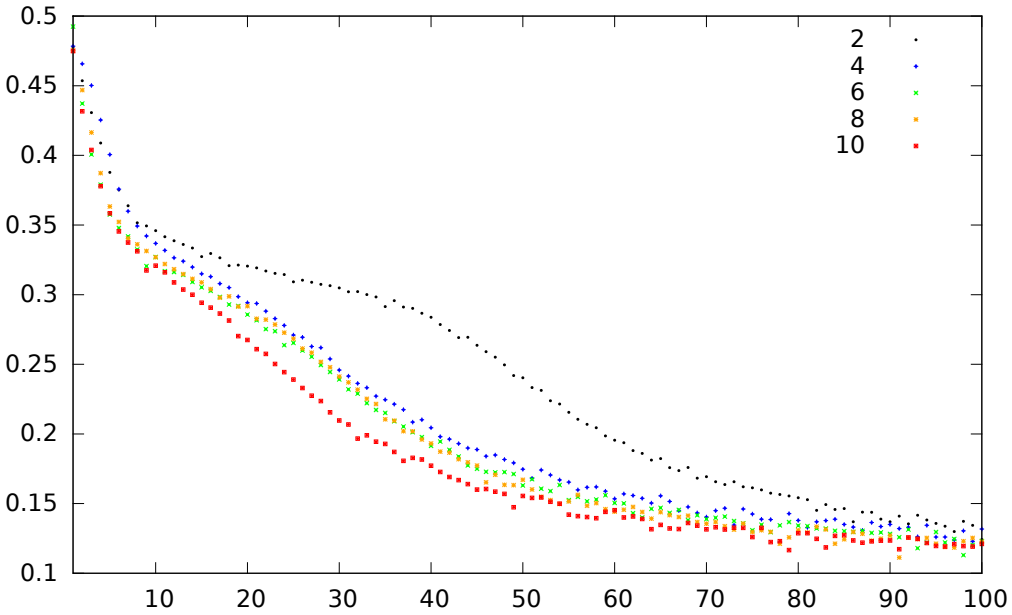


Rysunek 22: Przebieg błędu dla zbioru uczącego i weryfikującego

5. Przeprowadź uczenie algorytmem wstecznej propagacji błędu (może być bardzo długie, np. 20 000 epok); obserwuj przebieg błędu dla zbioru uczącego i weryfikującego.

Jak można zauważyć na powyższym wykresie, po około 5000 epok błąd średniokwadratowy dla zbioru weryfikującego zaczął rosnąć. Spowodowane jest to efektem przeuczenia.

Zadanie 4. Dobór liczby neuronów w warstwie ukrytej na przykładzie zbioru IRIS.



Rysunek 23: Przebieg błędu uczenia w zależności od liczby neuronów w warstwie ukrytej

3. Czy istnieje jednoznaczna zależność pomiędzy n a przebiegiem błędu średniokwadratowego? Czy biorąc pod uwagę tylko przebeg błędu dla zbioru uczącego można ustalić optymalną liczbę neuronów w warstwie ukrytej? Jeśli tak, to ile ona wynosi dla tego zbioru przykładów? Jak można zauważyć na powyższym wykresie, większa liczba neuronów w warstwie ukrytej spowodowała, że błąd w początkowych

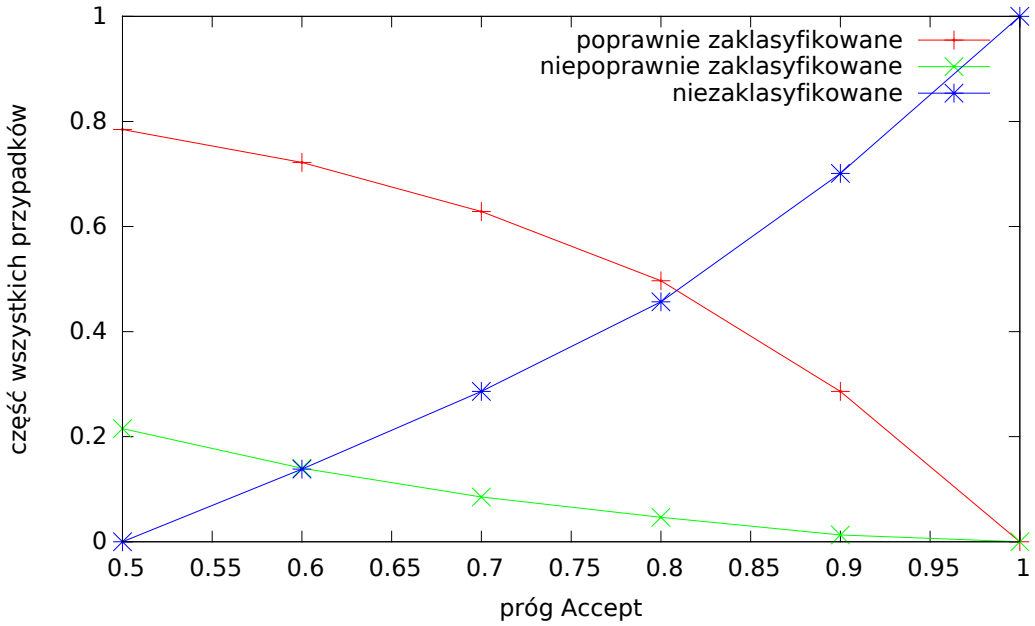
epokach uczenia spadał szybciej. Różnica ta malała jednak w kolejnych etapach. Nie można ustalić optymalnej liczby neuronów w warstwie ukrytej.

	v1	v2	v1	v2	v1	v2
Total	403	219	51	18	46	31
Correct	111	0	11	0	12	0
Wrong	0	2	0	0	0	2
Unknown	292	217	40	18	34	29
v1	111	2	11	0	12	2
v2	0	0	0	0	0	0

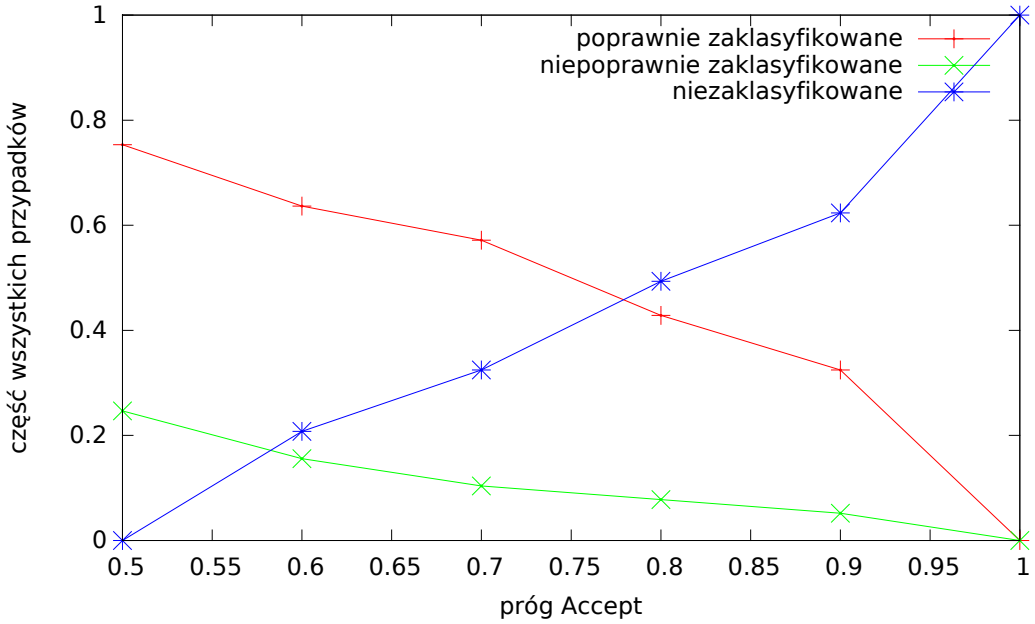
Rysunek 24: Przebieg błędu uczenia w zależności od liczby neuronów w warstwie ukrytej

Zadanie 7. Dobór liczby neuronów w warstwie ukrytej na przykładzie zbioru IRIS.

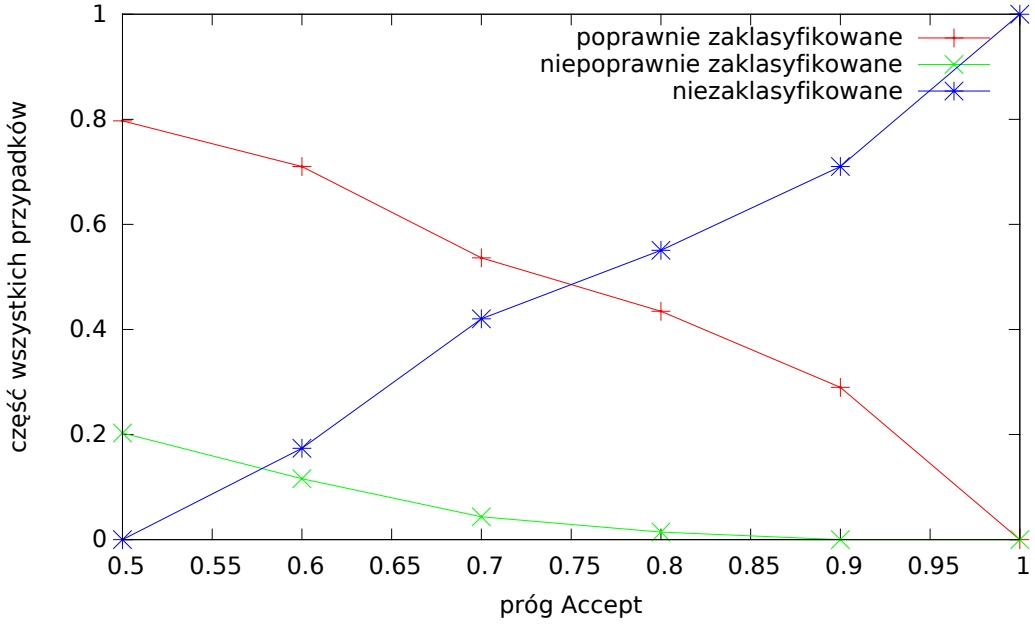
2. Przeprowadź klasyfikowanie przykładów ze zbioru testującego (okno Statistics—Classification). Znajdź w tym oknie macierz pomyłek (ang. confusion matrix)



Rysunek 25: Część przypadków zaklasyfikowanych poprawnie, błędnie i niezaklasyfikowanych w zależności od progu akceptacji, dla zbioru treningowego.



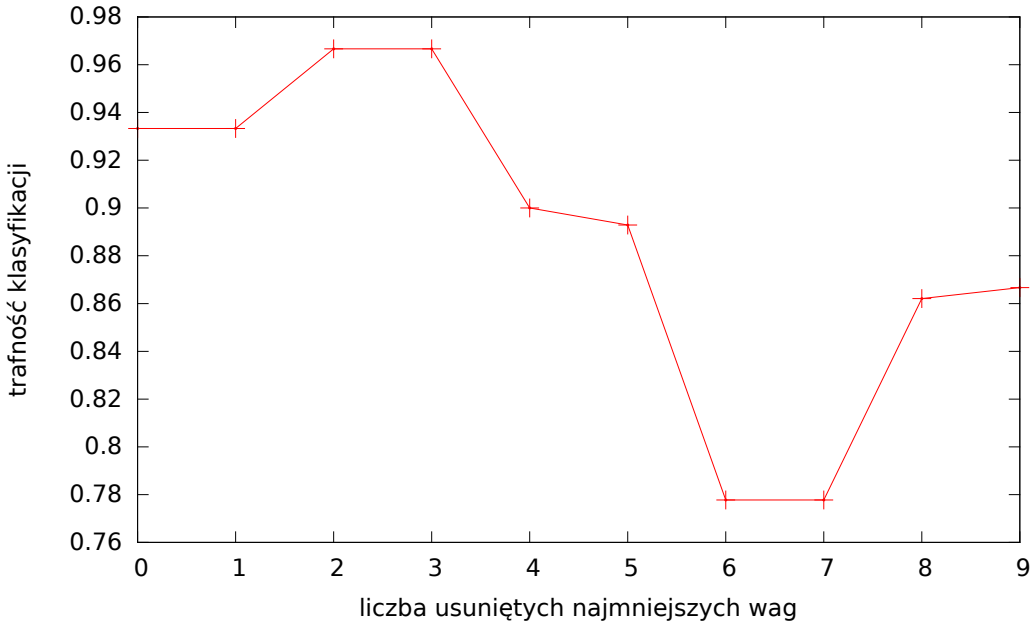
Rysunek 26: Część przypadków zaklasyfikowanych poprawnie, błędnie i niezaklasyfikowanych w zależności od progu akceptacji, dla zbioru testowego.



Rysunek 27: Część przypadków zaklasyfikowanych poprawnie, błędnie i niezaklasyfikowanych w zależności od progu akceptacji, dla zbioru weryfikującego.

4. Przy użyciu gnuplota sporządź wykres (trzy przebiegi na jednym wykresie) zależności procentu przypadków z jednego ze zbiorów (uczącego / weryfikującego / testującego): – poprawnie zaklasyfikowanych (do wszystkich klas decyzyjnych razem), – niepoprawnie zaklasyfikowanych, – niezaklasyfikowanych w funkcji progu Accept

5. Czy na podstawie otrzymanego wykresu można zasugerować jakąś optymalną wartość obu progów dla tego zbioru przykładów i tej sieci? Tak. Wartość ta zależy jednak od tego jak 'kosztowne' jest dokonanie błędnej klasyfikacji. Zakładając, że w analizowanym problemie koszt takiej pomyłki jest niewielki, sugerowanymi wartościami progów Accept i Reject są odpowiednio: 0.5 i 0.5.



Rysunek 28: Trafność klasyfikacji w zależności od liczby usuniętych wag.

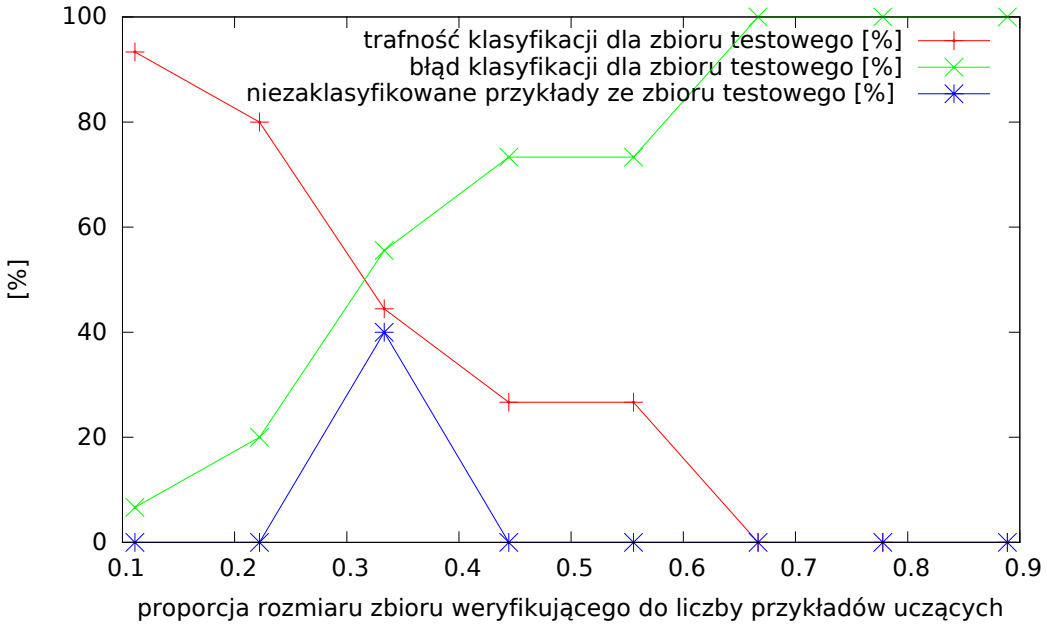
Zadanie 8. Badanie odporności sieci na uszkodzenia.

5. Czy odporność sieci na uszkodzenia (usunięcia wag) jest wysoka? Tak. Jak można zauważyć na wykresie, pomimo usunięcia wag, trafność klasyfikacji pozostała wysoka.

6. Czy jesteś w stanie na podstawie tak "zredukowanej" sieci powiedzieć coś o ważności poszczególnych atrybutów opisujących przykłady? Pomimo tego, że wagi dwóch pierwszych atrybutów zostały prawie całkowicie wyzerowane, trafność klasyfikowania pozostała wysoka. Wydaje się, że ważność przytoczonych atrybutów jest niewielka.

7. Czy w konsekwencji "przerzedzenia" sieci można usunąć niektóre neurony? Jak należy to zrobić? (przemyśl dokładnie) Można usunąć tylko takie neurony, których zarówno wszystkie wagi wejściowe, jak i wyjściowe są wyzerowane.

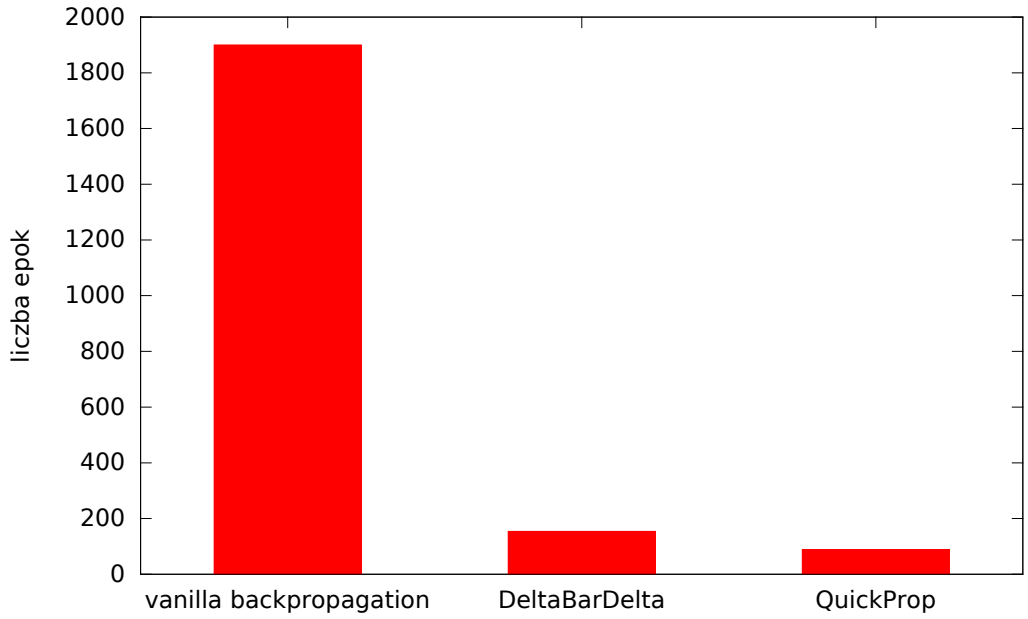
Zadanie 9. Eksperymentalny dobór rozmiaru zbioru weryfikującego Eksperyment przeprowadzono na zbiorze IRIS (150 przykładów) oraz sieci neuronowej o architekturze 4-3-3.



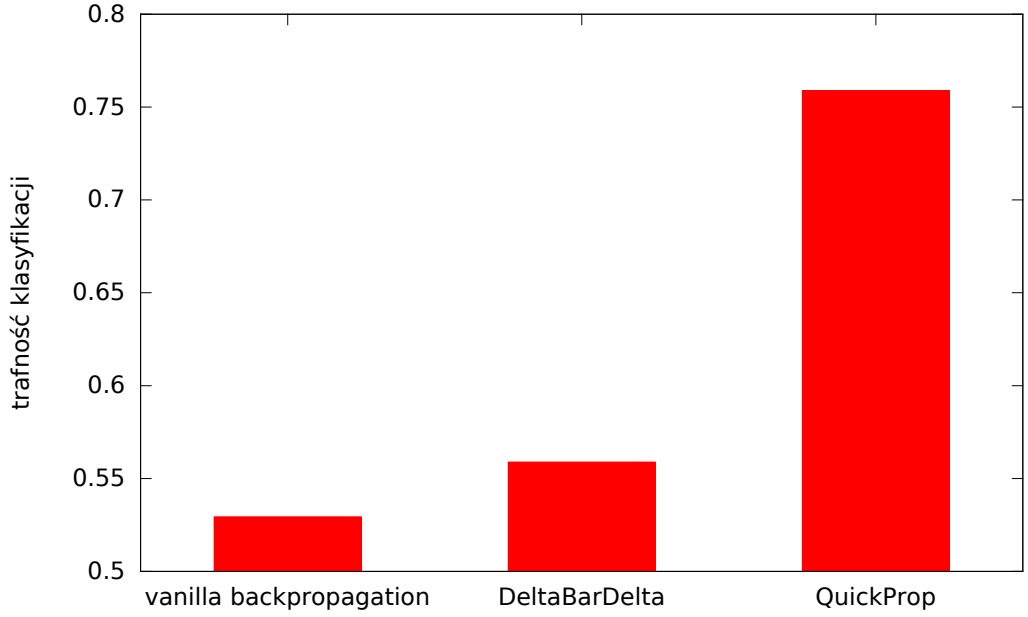
Rysunek 29: Trafność klasyfikowania, błąd klasyfikowania i procent przykładów niezaklasyfikowanych w funkcji proporcji rozmiaru zbioru weryfikującego do liczby przykładów uczących

Jak można zauważyć na wykresie, jakość klasyfikacji jest tym lepsza im większy jest stosunek wielkości zbioru treningowego do zbioru weryfikującego.

Zadanie 10. Porównanie 'vanilla' backpropagation z bardziej wyrafinowanymi algorytmami uczenia nadzorowanego sieci warstwowych Eksperyment przeprowadzono na zbiorze BUPA na sieci neuronowej o architekturze 6-6-1. Zbadano działanie algorytmów: vanilla backpropagation, QuickProp oraz DeltaBarDelta. Dla każdego z nich, sieć uczona była aż do osiągnięcia warunku stopu, którym było osiągnięcie wielkości błędu średniokwadratowego mniejszej lub równej 0.44. Dla każdego algorytmu, naukę sieci przeprowadzono pięciokrotnie, a uzyskane wyniki uśredniono. Na wykresach przedstawiono liczbę epok, jaka upłynęła aż do osiągnięcia warunku stopu oraz uśrednioną trafność klasyfikacji dla każdego algorytmu.



Rysunek 30: Liczba epok, potrzebnych do osiągnięcia warunku stopu w zależności od wykorzystanego algorytmu



Rysunek 31: Średnia trafność klasyfikacji w zależności od wykorzystanego algorytmu