Sieci neuronowe

Sieci jednokierunkowe wielowarstwowe typu perceptron wielowarstwowy

Matlab

Proces tworzenia i posługiwania się siecią neuronową w środowisku Matlab można podzielić na trzy etapy:

1. Tworzenie modelu sieci
2. Uczenie sieci
3. Symulacja/Testowanie sieci

Pierwszy etap obejmuje proces definiowania parametrów sieci, wynikiem czego jest stworzenie odpowiedniej struktury danych zawierającej opis poszczególnych składników sieci. W etapie tym możliwe jest określenie

1. Liczby neuronów w poszczególnych warstwach
2. Kształtu neuronów w poszczególnych warstwach (domyślnie w Matlabie cała warstwa powinna posiadać identyczną funkcję aktywacji)
3. Funkcji uczącej sieć (algorytm uczenia)
4. Ilości epok uczenia
5. itp.

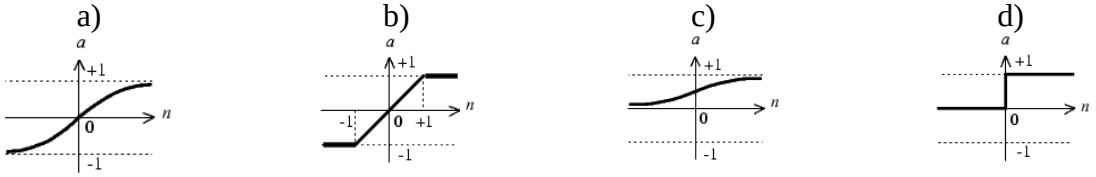
Proces ten realizowany jest poleceniem:

```
net = newff(X,Y,S,T,BTF);
```

gdzie

- net – struktura opisująca sieć
- X –wejścia sieci (uwaga w poziomie cechy, w pionie wektory)
- Y – wyjścia sieci (etykiety wektorów wej.) każdy wiersz opisuje pojedyncze wyjście
- S_i – współczynnik określający ilość neuronów w poszczególnych warstwach oprócz ostatniej (ostatnia warstwa jest zdeterminowana przez liczbę wyjść)
- T – tablica komórkowa zawierająca informację w postaci stringów opisujących kształt funkcji aktywacji dla poszczególnych warstw. Możliwe funkcje to:
  a. tansig – sigmoidalna funkcja aktywacji (+1,-1)
  b. satlins - liniowa funkcja aktywacji z ograniczeniami
  c. logsig - logarytmiczno sigmoidalna sunkcja aktywacji (0, +1)
  d. hardlim – binarna funkcja aktywacji typu y=1(x)

a) b) c) d)



- BTF – algorytm uczenia sieci. Możliwe opcje to:
  - traingd - największego spadku z propagacją wsteczną
  - traingda - największego spadku z adaptacją wsp. uczenia i propagacją wsteczną
  - traingdm - Największego spadku z funkcją momentum z propagacją wsteczną

- traingdx - Największego spadku z momentum i adaptacyjnym wsp. uczenia i propagacją wsteczną
- trainlm - algorytm Levenberg-Marquardt

Po stworzeniu struktury sieci wartości wag w sposób automatyczny są inicjalizowane wg. określonego algorytmu.
Kolejnymi istotnymi parametrami posiadającymi istotny wpływ na proces uczenia sieci są:

```
net.trainParam.epochs = 200; - liczba epok/iteracji uczenia
net.trainParam.showWindow = false; - czy ma być wyświetlane okno uczenia
net.performFcn='mse'; - typ funkcji kosztu
```

Etap drugi to proces uczenia sieci, w którym następuje dopasowanie wag neuronów. Zadanie to – kluczowe realizowane jest za pomocą funkcji:

```
nnet = train(net,X,Y);
```

gdzie

- net – struktura opisująca zainicjowaną sieć
- X – zbiór danych wejściowych (patrz wyżej)
- Y – zbiór danych wyjściowych – informacje które sieć ma się nauczyć (patrz wyżej)
- nnet – struktura sieci opisująca nauczoną sieć nauronową

Ostatnim etapem działania sieci jest proces jej symulacji, w którym następuje wykorzystanie nauczonej sieci w celu predykcji czyli wyznaczenia wartości wyjściowych dla określonych wejść. Etap ten realizowany jest poprzez polecenie:

```
D = sim(nnet,Xt);
```

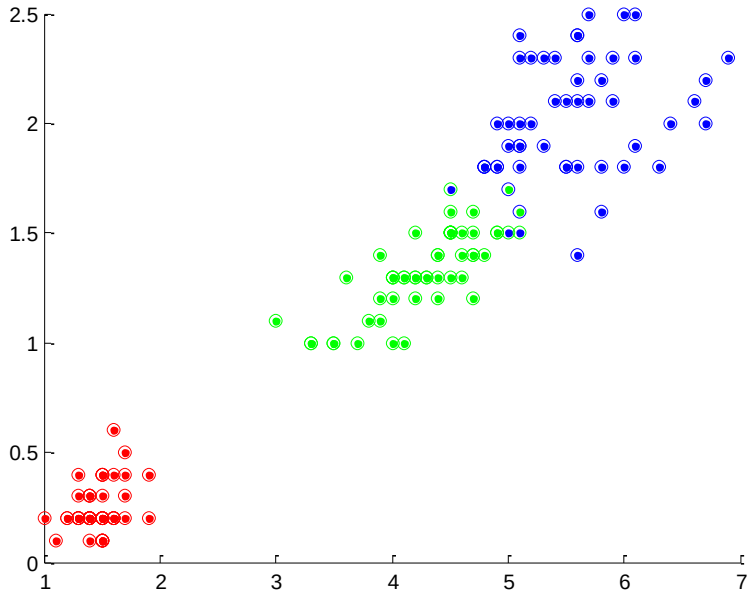
Gdzie

- nnet – jest strukturą opisującą nauczoną sieć
- Xt – jest macierzą opisującą dane dla których nasza sieć ma dokonać predykcji, czyli odpowiedzieć jakie powinno być wyjście z sieci
- D – odpowiedź sieci

Zadania

Zad 1

Wczytaj zbiór danych Iris.data, stwórz zmienne: X opisującą wejście sieci (na wstępie wykorzystaj tylko 3 i 4 cechę zbioru Iris) pamiętaj że toolbox matlaba wymaga by dane były zapisane w postaci kolumn, gdzie jedna kolumna opisuje jeden przypadek oraz Y opisującą wyjście sieci.
Zbuduj funkcję dokonującą binaryzacji zmiennej wyjściowej, tak aby każda kategoria była reprezentowana przez osobny wiersz którego elementy przyjmują wartości 1 jeśli wystąpiła dana kategoria. (Szczegóły u prowadzącego). Przekształć wyjście numeryczne na binarne.
Na podstawie X i Y stwórz sieć neuronową o jednej warstwie ukrytej i 3 neuronach.
Dokonaj symulacji sieci nie nauczonej na danych X i wynik zapisz do zmiennej D1.
Dokonaj procesu uczenia sieci na danych X i Y, a następnie dokonaj ponownie symulacji sieci na danych X, a wynik zapisz do D2.
Zbuduj funkcję rysującą dane tak aby każda kategoria rysowana była innym kolorem.
Przykład:



I wyświetl na dwóch różnych rysunkach wyniki działania sieci.
Zbuduj funkcję obliczającą dokładność uczenia sieci jako ilość poprawnie odgadniętych wzorców w stosunku do całkowitej liczby wzorców.

Zad 2

Dokonaj identycznych operacji jak w zad. 1 z tą różnicą iż weź pełny zbiór danych uczących zawierający wszystkie zmienne wejściowe a nie tylko 3 i 4. Korzystając z funkcji z ostatnich zajęć dokonaj podziału na trening/test.
Naucz sieć na zbiorze treningowym i dokonaj symulacji na zbiorze testowym. Oblicz dokładność uczenia sieci. Wykonaj powyższe operacje kilka razy. Czy wyniki dokładności ulegają zmianie?

Zad 3

Wyżej opisane zad 2 wykonaj w pętli 10 razy każdorazowo zapisując wynik dokładności uczenia. Na koniec wyznacz średnią dokładność sieci.

Zad 4

Wykonaj operacje z zad.3 zmieniając liczbę neuronów w poszczególnych warstwach i dodając kolejne warstwy sieci. Wyniki zapisuj w tabelce.

Zad 5

Powtórz operacje z zad 4 dla różnych algorytmów uczących zawartych w opisie.

Zad 6

Powtórz obliczenia dla innego zbioru danych zaproponowanego przez prowadzącego.