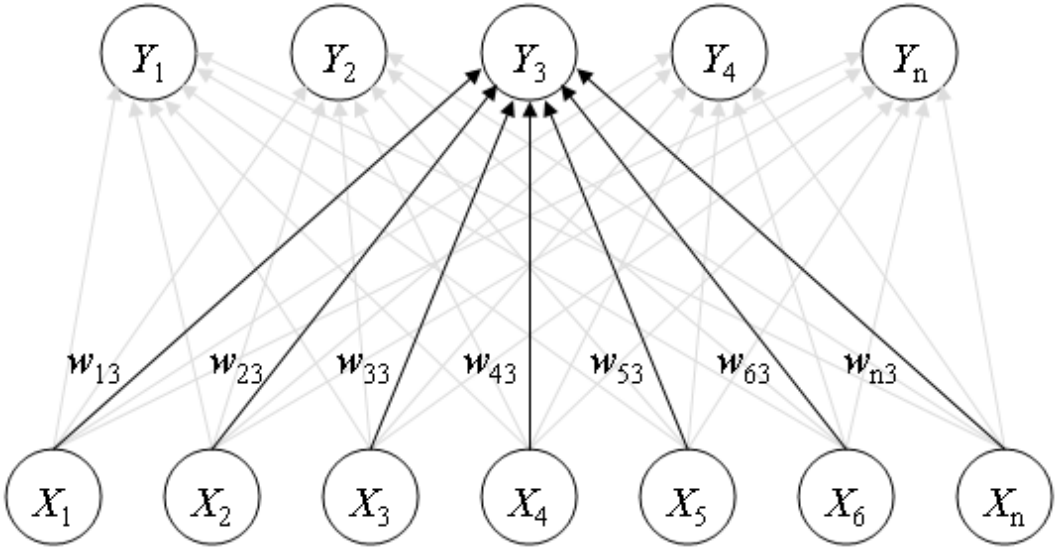
Daniel Wenecki
Podstawy Sztucznej Inteligencji
Sprawozdanie – Projekt nr 5

1.Cel ćwiczenia
Celem ćwiczenia jest poznanie budowy i działania sieci Kohonena przy wykorzystaniu reguły WTA do odwzorowywania istotnych cech kwiatów.

2. Syntetyczny opis budowy użytej sieci i algorytmu uczenia
Do wykonania ćwiczenia użyłem sieci Kohonena przy użycie reguły WTA.

Schemat sieci:



Schemat modyfikacji wag:

$$w_{ij}(new) = w_{ij}(old) + \alpha[x_i - w_{ij}(old)]$$

Odległość pomiędzy punktami znajdowałem za pomocą funkcji:

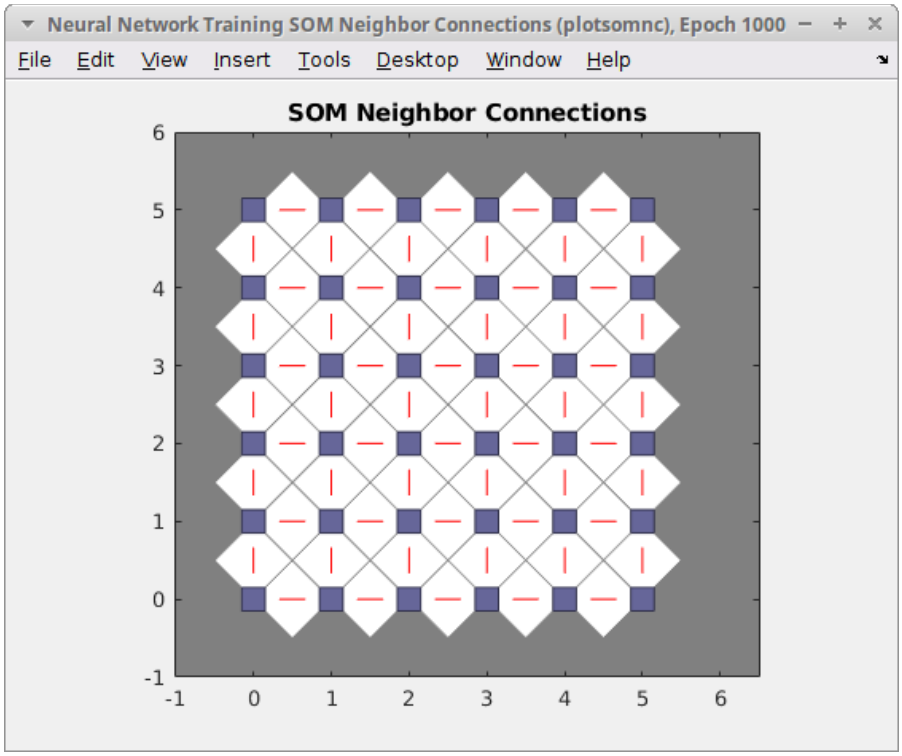
$$d(A,B) = \sqrt{(x_{1A} - x_{1B})^2 + (x_{2A} - x_{2B})^2 + \ldots + (x_{nA} - x_{nB})^2} = \sqrt{\sum_{i=1}^{n}((x_{iA} - x_{iB})^2)}$$

Jako dane do klasyfikacji użyłem danych o 3 różnych odmianach kwiatów, dane można znaleźć pod linkiem: https://en.wikipedia.org/wiki/Iris_flower_data_set

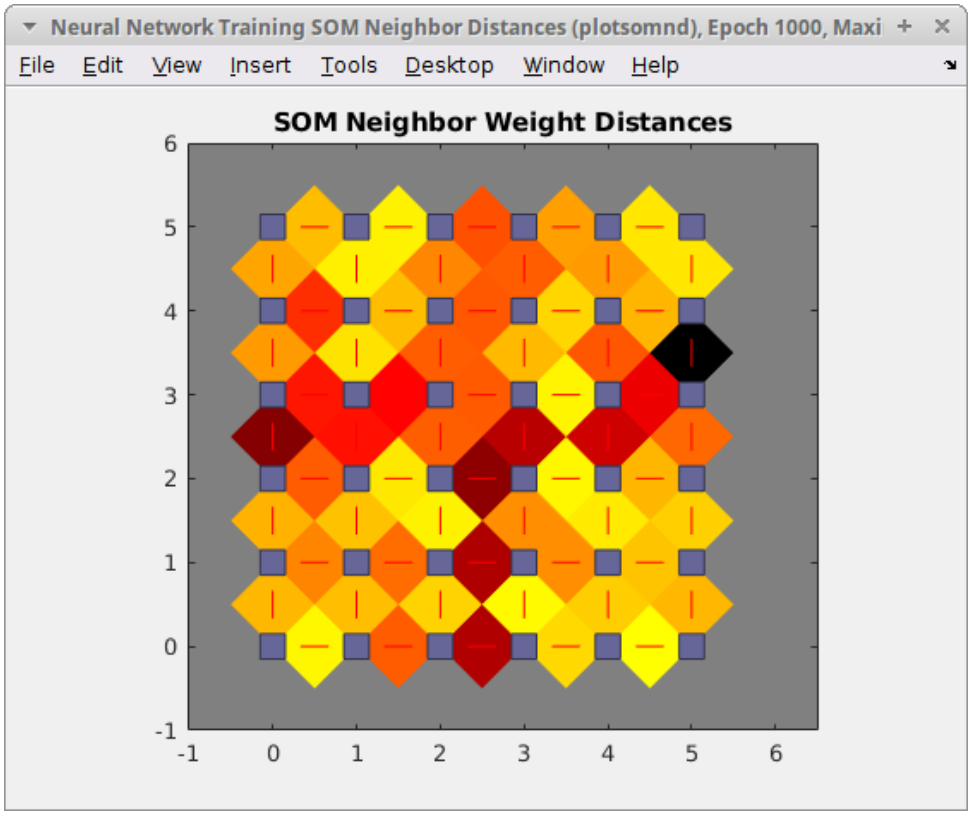
Do stworzenia wyżej wymienionej sieci użyłem programu matlab i narzędzia nntool.

3. Zestawienie otrzymanych wyników oraz ich analiza

Początkowo stworzyłem sieć 6 na 6 przypominającej siatkę, w której neurony miały maksymalnie 4 połączenia.

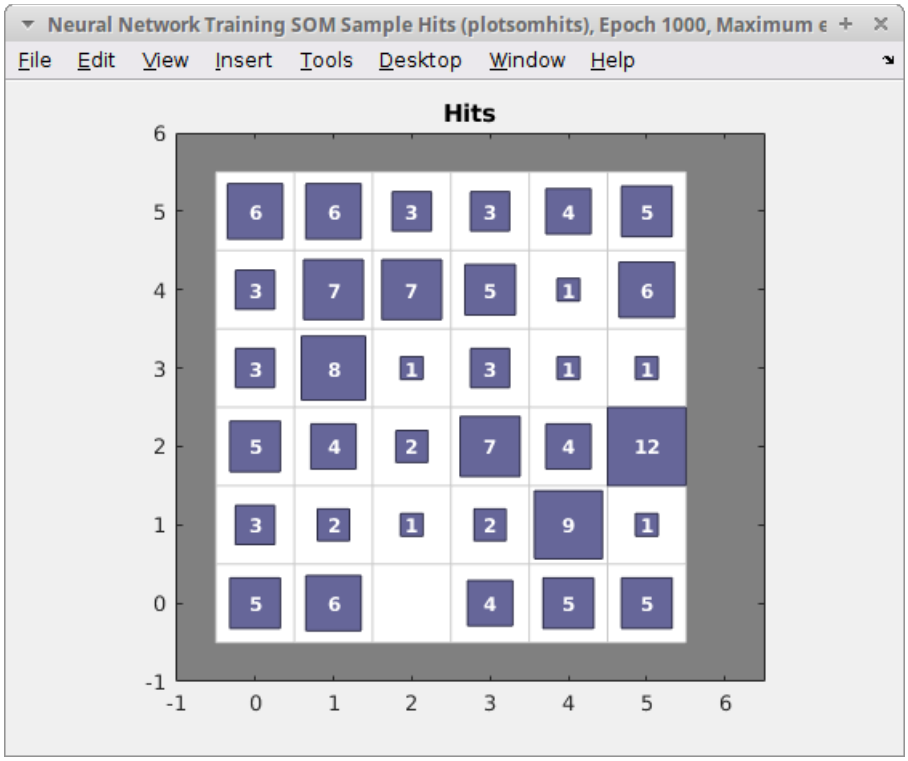


Po 1000 epokach odległości pomiędzy neuronami prezentowały się naastępująco:



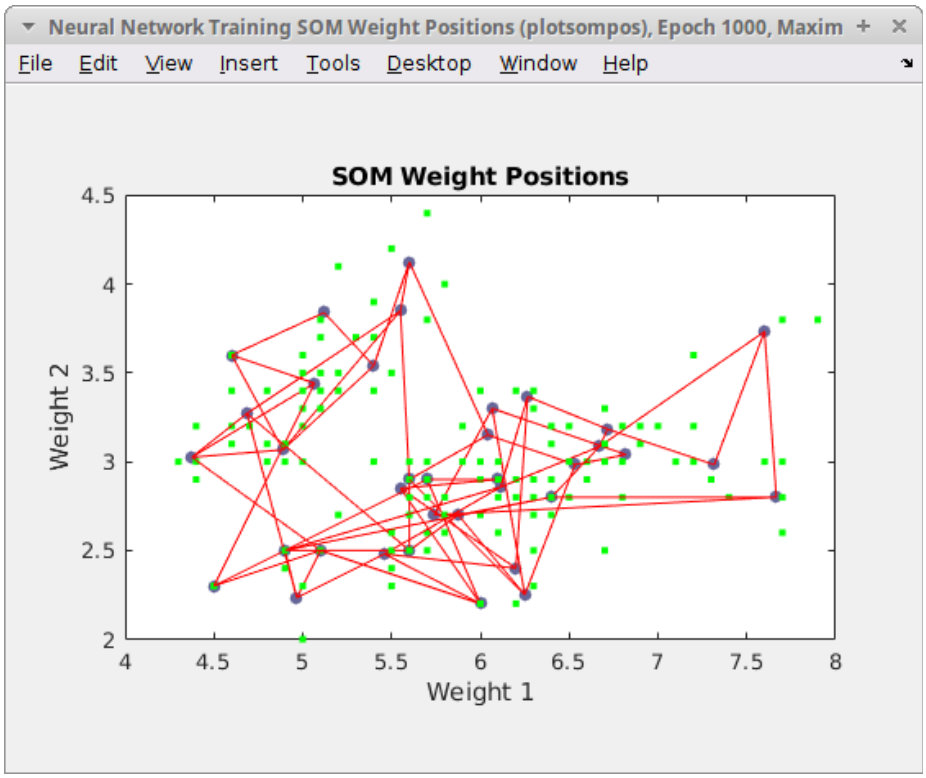
Na podanym rysunku kolor ciemnniejsze kolory oznaczają większy dystans, a kolory jaśniejsze mniejszy dystans. Na naszym rysuknu widzimy jedną wyraźną przerwę między jedną grupą neruonów a pozostałą. Reszta neuronów nie ma zbyt dużych odległości pomiędzy sobą, co możemy w naszym przypadku wyjaśnić tym, że dwa gatunki kwiatów mają zbliżone do siebie cechy.

Następną przydatną mapą do interpretacji jest mapa trafień:



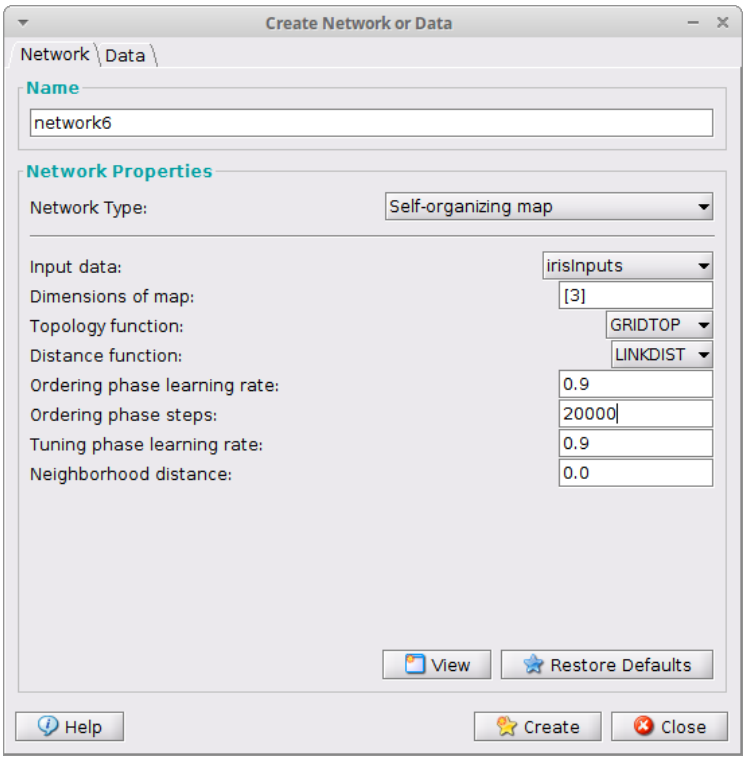
Przedstawia ona, jak wiele danych wejściowych zakwalfikuje każdy neuron. Widzimy na niej blisko 50 trafień w prawym dolnym rogu, co oznacza, że nasz kawałek sieci dobrze rozpoznaje jeden z gatunków na podstawie dostarczonych danych. Pozostałe dwa gatunki nie są dla sieci łątwo rozróżnialne.

To jak sieć odwzorowuje dane dobrze przedstawia wizualizacja położenia wektorów wag i wartości uczących:

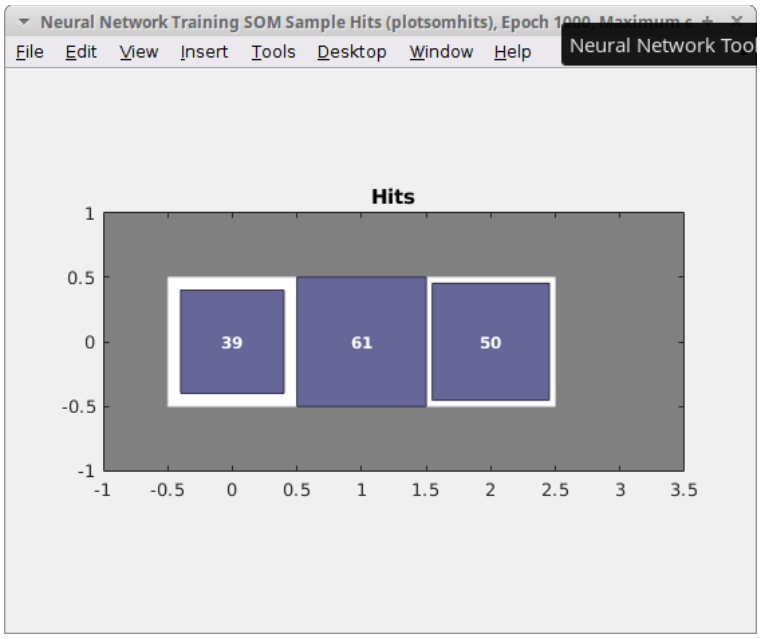


W tym przypadku jest to wykres dwóch pierwszych wymiarów wektora danych uczących i wektora wag. Z rysunku widzimy, że sieć przyjmuje wartości wag, tak aby ustawić się jak najbliżej punktów danych początkowych.

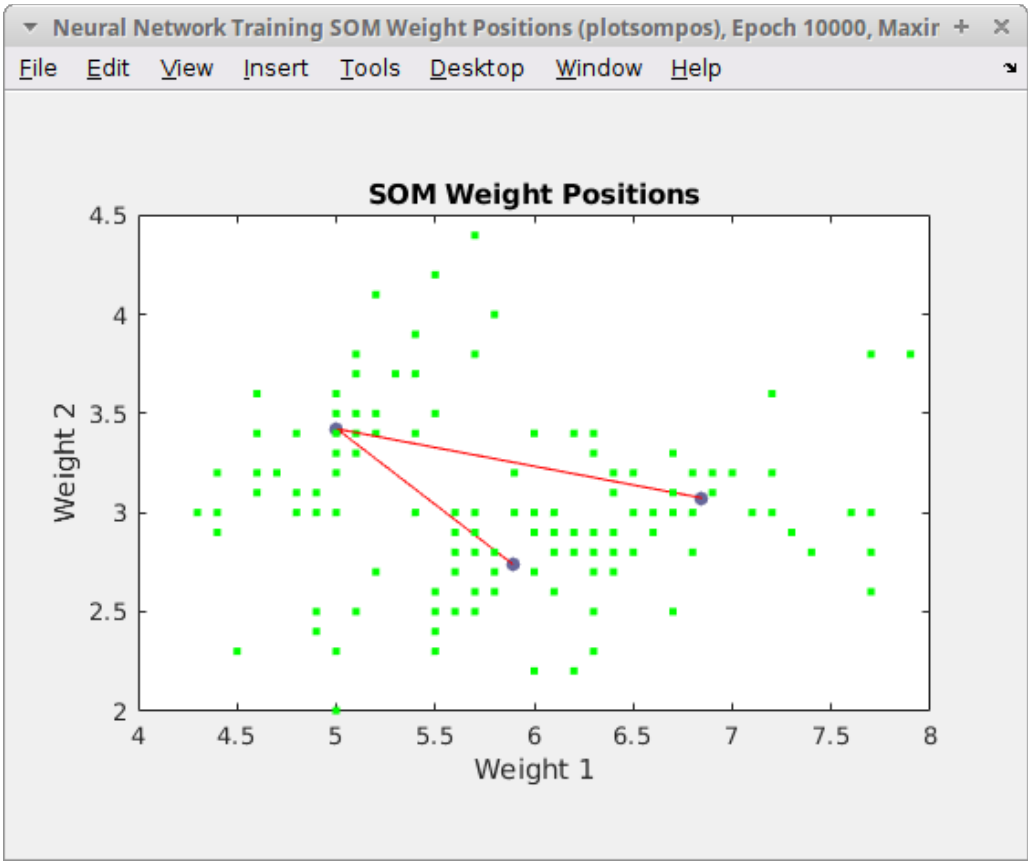
Aby kwalfikować dane wejściowe do gatunku, stworzymy nową sieć, złożoną z 3 neuronów, która nam to umożliwi.



Tu także po nauczeniu się sieci obesrwujemy błędy, sieć nie nauczyła się więc rozpoznawać gatunków dokładnie.



Jak widzimy sieć po 10000 iteracji przy współczynniku uczenia równym 0.1 bezbłędnie rozpoznała tylko jeden gatunek. Pozostałe dwa nie są rozpoznawalne w 100%.



Ponownie widzimy próbę znalezienia przez sieć 3 oddzielnych zbiorów wartości.

Przeprowadziłem testy uczenia dla sieci złożonej z 3 neuronów przy stałej liczbie 1000 iteracji, dla różnych współczynników uczenia

lr	0.1	0.01	0.001
Stopień błędnych odpowiedzi	0,0933333333	0,0866666667	0,0866666667

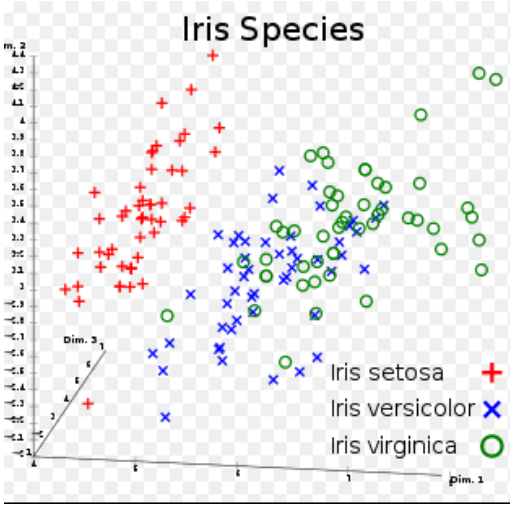
Jak widzimy stopień błędnych odpowiedzi był większy przy większym kroku uczenia, jest to spowodowane tym, że większy krok uczenia, oznacza szybszą naukę sieci, ale też i mniejszą jej dokładność.

5. Wnioski

Wybierając learning rate, powinniśmy wybrać optymalną wartość. Wartość lr zbyt duża spowoduje niedokładność lub w ogóle nie możliwość nauczenia się sieci, wartość zbyt mała bardzo wydłuży czas uczenia.

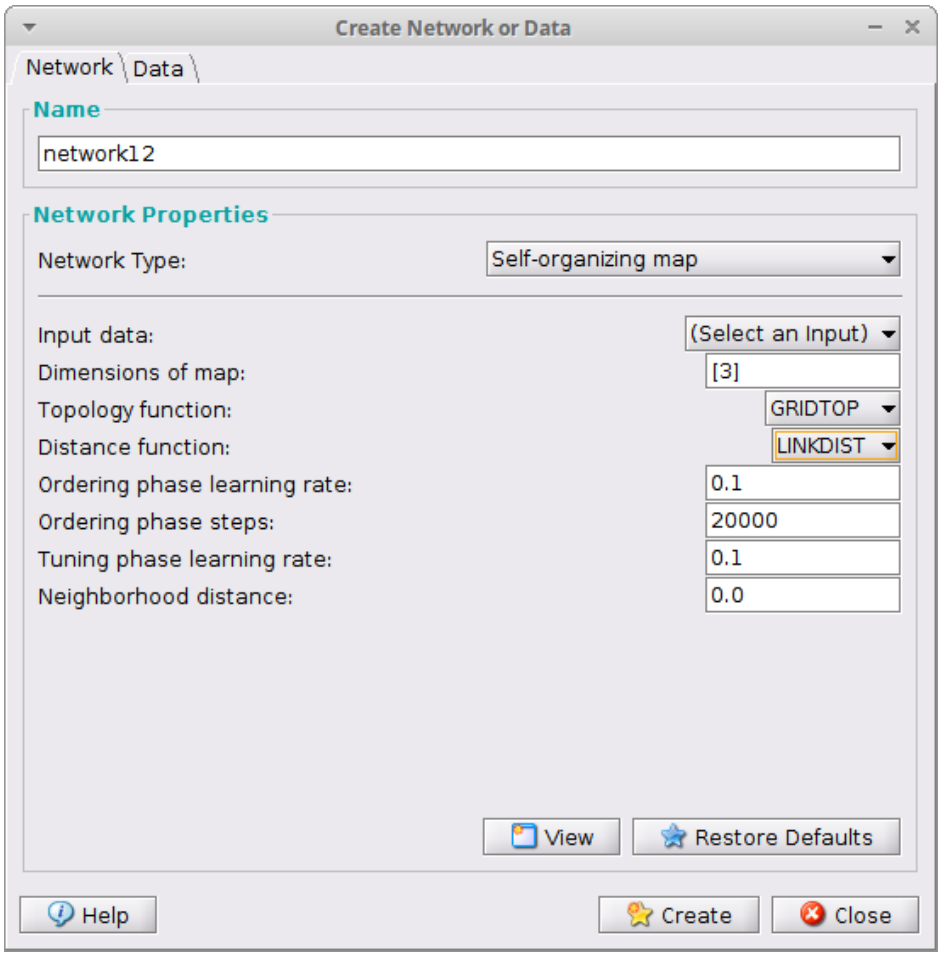
Jeśli chcemy sieć nauczyć rozpoznawać, tak jak w naszym przykładzie gatunki, powinniśmy wybierać zestaw parametrów, który da się pogrupować. Wybranie danych, w których wariancja w parametrach zależna od gatunku nachodzi na siebie, uniemożliwia stworzenia sieci, która by w blisko 100% rozpoznawała gatunek. Możemy za to uzyskać pewną dokładność, lub dodać dodatkowy parametr w celu zwiększenia dokładności.

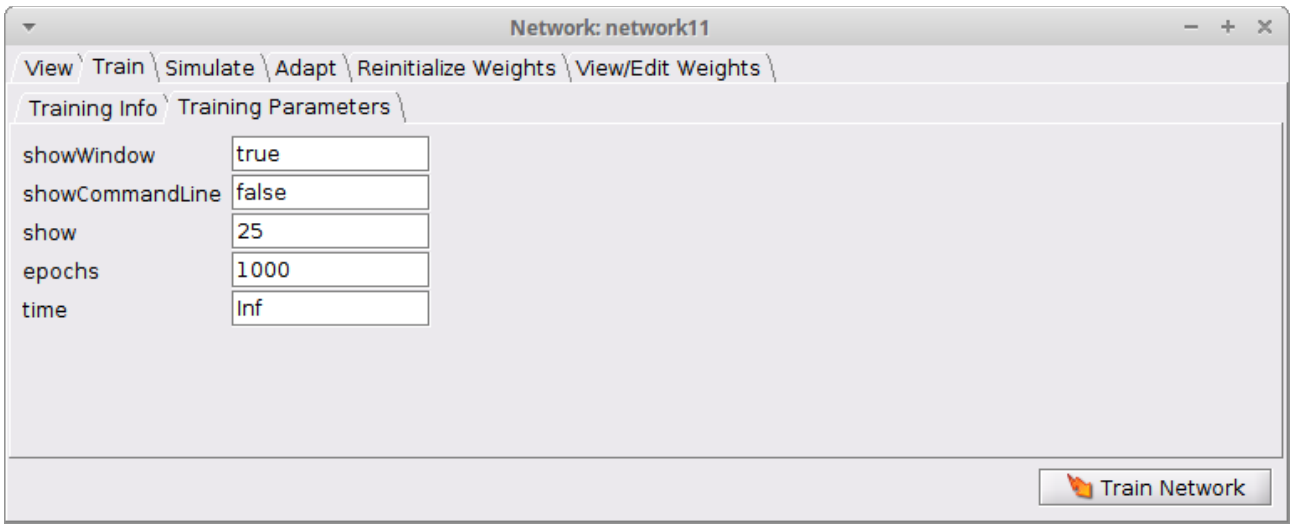
Nachodzenie się gatunków widać dla 3 wymiarów na screenie:

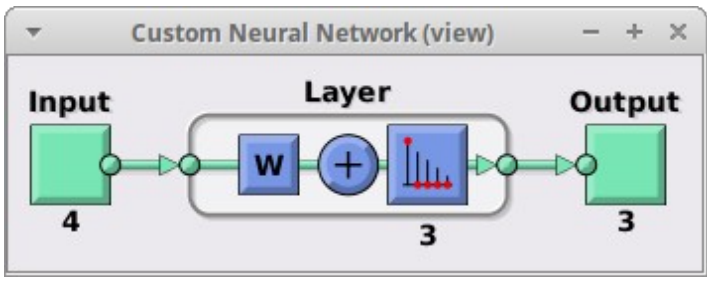


Jeśli chcemy tylko rozpoznawać kwiaty w danych uczących, możemy sieć nauczyć ich rozpoznawania poprzez zwiększenie ilości neuronów. Należy brać jednak pod uwagę, że w przypadku podania innego kwiatu sieć nadal będzie mogła się pomylić.

6. Screeny konfiguracji programu:







Bibliografia:

https://en.wikipedia.org
https://www.mathworks.com/help/nnet/self-organizing-maps.html
http://mnemstudio.org/neural-networks-kohonen-self-organizing-maps.htm