# Sprawozdanie laboratorium 2 (sztuczne sieci neuronowe)

**Autorzy:** Remigiusz Nowakowski  
**Grupa:** INMN1(hybryda)\_1.1, INMN1(hybryda)\_sp\_wszyscy  
**Data:** 02.12.2023  
**Przedmiot:** Inteligencja obliczeniowa

|  |  |
| --- | --- |
| **Cel zadania (co chcemy klasyfikować)** | Przewidywanie, czy czas oczekiwania na następną erupcję gejzeru Old Faithful w Parku Narodowym Yellowstone w Stanach Zjednoczonych jest dłuższy niż średnia. |
| **Skąd pobrano zbiór danych?** | data(faithful) |
| **Jakie dane zawiera każda kolumna zbioru danych?** | 1. eruptions: Czas trwania erupcji gejzera w minutach. 2. waiting: Czas oczekiwania od poprzedniej erupcji do następnej erupcji, również w minutach. |
| **Jak jest dokładność klasyfikacji?** | 95.6044% |
| **Wnioski i podsumowanie (analiza otrzymanych wyników)** | Podział Danych: Przy użyciu metody hold-out, dane zostały skutecznie podzielone na zbiór treningowy i testowy, co umożliwiło ocenę skuteczności modelu na niezależnym zestawie danych.  Struktura Sieci Neuronowej: Wybrana struktura sieci neuronowej składała się z warstwy wejściowej, jednej warstwy ukrytej i warstwy wyjściowej. Wielkości warstw oraz funkcje aktywacji zostały dostosowane do charakterystyki danych.  Trenowanie Sieci: Proces trenowania sieci został zakończony po określonej liczbie iteracji. Wartości błędów były monitorowane, a proces trenowania był stopniowo optymalizowany przy użyciu algorytmu adaptacyjnego gradientu.  Ocena Klasyfikacji: Model został oceniony na danych testowych za pomocą macierzy kontyngencji, co pozwoliło na analizę liczby poprawnie i błędnie sklasyfikowanych przypadków.  Dokładność Klasyfikacji: Uzyskana dokładność klasyfikacji wyniosła X%. Wynik ten wskazuje na skuteczność modelu w przewidywaniu odpowiednich kategorii na podstawie dostępnych danych.  Eksperymentacja i Optymalizacja: W przypadku, gdy uzyskana dokładność nie spełnia oczekiwań, zaleca się eksperymentację z różnymi parametrami sieci, takimi jak liczba neuronów, współczynniki uczenia, funkcje aktywacji itp.  Wnioski: W celu dalszej poprawy skuteczności modelu, można przeprowadzić dodatkową analizę danych, dostosować strukturę sieci, zoptymalizować parametry uczenia, lub rozważyć zastosowanie innych metod klasyfikacji dostępnych w pakiecie AMORE. |
| **Bibliografia** | [1] Biblioteka AMORE  [2] Moodle  [3] data(faithful)  [4] |

## Kod źródłowy rozwiązania wraz z dodatkowymi (szczegółowymi) komentarzami, (w szczególności proszę opisać parametry użytych funkcji z pakietu AMORE)

|  |
| --- |
| # Zakładamy, że chcemy przewidzieć, czy czas oczekiwania jest dłuższy niż średnia  # Instalacja pakietu AMORE (odkomentuj linię poniżej, jeśli nie jest zainstalowany)  # install.packages("AMORE")  library(AMORE)  # Załaduj dane  data(faithful)  # Określenie funkcji target  target <- function(x) {  ifelse(x$waiting > mean(faithful$waiting), 1, 0)  }  # Zastosowanie funkcji target dla kategorii "waiting"  wZadane <- target(faithful)  # Ustawienie ziarna dla powtarzalności wyników  set.seed(888)  # Podział danych na dane trenujące i testowe (2/3 - 1/3)  ile <- nrow(faithful)  idxTren <- sample(1:ile, 2 \* ile / 3)  idxTest <- setdiff(1:ile, idxTren)  # Tworzymy strukturę sieci  siec <- newff(n.neurons = c(2, 5, 1),  learning.rate.global = 0.05,  momentum.global = 0.5,  hidden.layer = "sigmoid",  output.layer = "purelin",  method = "ADAPTgdwm",  error.criterium = "LMS")  # Trenujemy sieć  wynik <- train(siec,  faithful[idxTren, -2], # Zakładając, że druga kolumna to waga  wZadane[idxTren],  error.criterium = "LMS",  report = TRUE,  show.step = 10,  n.shows = 800)  # Wyświetlamy wartości błędów  plot(wynik$Merror, type = "l", xlab = "Iteracja (x10)",  ylab = "Błąd", col = "darkred")  # Stosujemy wytrenowaną sieć do danych testowych  y <- sim(wynik$net, faithful[idxTest, -2])  y  # Definiujemy funkcję oceny klasyfikacji (zamieniamy liczby na etykietę)  testKlasyfikacji <- function(zad, wy) {  zadane <- max.col(zad)  rozpoznane <- max.col(wy)  print(table(zadane, rozpoznane))  }  # Wyniki klasyfikacji  wynik <- testKlasyfikacji(wZadane[idxTest], y)  # Określamy dokładność klasyfikacji  dokladnosc <- sum(ifelse(y > 0.5, 1, 0) == wZadane[idxTest]) / length(wZadane[idxTest]) \* 100  cat("Dokładność klasyfikacji:", dokladnosc, "%\n") |