SPRAWOZDANIE

Zajęcia: Analiza Procesów Uczenia

Prowadzący: prof. dr hab. Vasyl Martsenyuk

Laboratorium 5

Data 22.05.2019r.

Temat: "Użycie sztucznych sieci neuronowych"

Wariant 4

Artur Kozieł

Informatyka II stopień,

stacjonarne

1 semestr

1. **Polecenie**

   Celem ćwiczenia było stworzenie i nauczenie sieci neuronowej przewidywania wartości funkcji $\boldsymbol{f(x) = log\ x^2}$

2. **Wprowadzane dane**

   Jako dane treningowe została stworzona ramka danych – wektor *output* to 50 liczb losowych z rozkładu jednostajnego z przedziału 1 do 10; natomiast wektor *input* to wartości funkcji $f(x)$ dla wektora x

3. **Wykorzystane komendy**

```
library("neuralnet")                    # dołączenie pakietu neuralnet
output <- as.data.frame(runif(50, min = 1, max = 10))      #określenie zbioru wektora losowych wartości
input <- log(output^2)      # wyliczenia wartości podanej funkcji
trainingData <- cbind(input, output)    # połączenie danych w jedną ramkę
colnames(trainingData) <- c("Input", "Output")      #nadanie nazw kolumnom
normalize <- function(x) {          # stworzenie funkcji normalizującej wartości
  return ((x - min(x)) / (max(x) - min(x)))}                    danych treningowych
maxmindf <- as.data.frame(lapply(trainingData, normalize))
net.price <- neuralnet(Output~Input, maxmindf, hidden = c(4, 7, 3), threshold = 0.01)  # uczenie sieci neuronowej z trzema warstwami ukrytymi o wielkościach 4, 7 i 3
plot(net.price) # narysowanie schematu sieci neuronowej
```

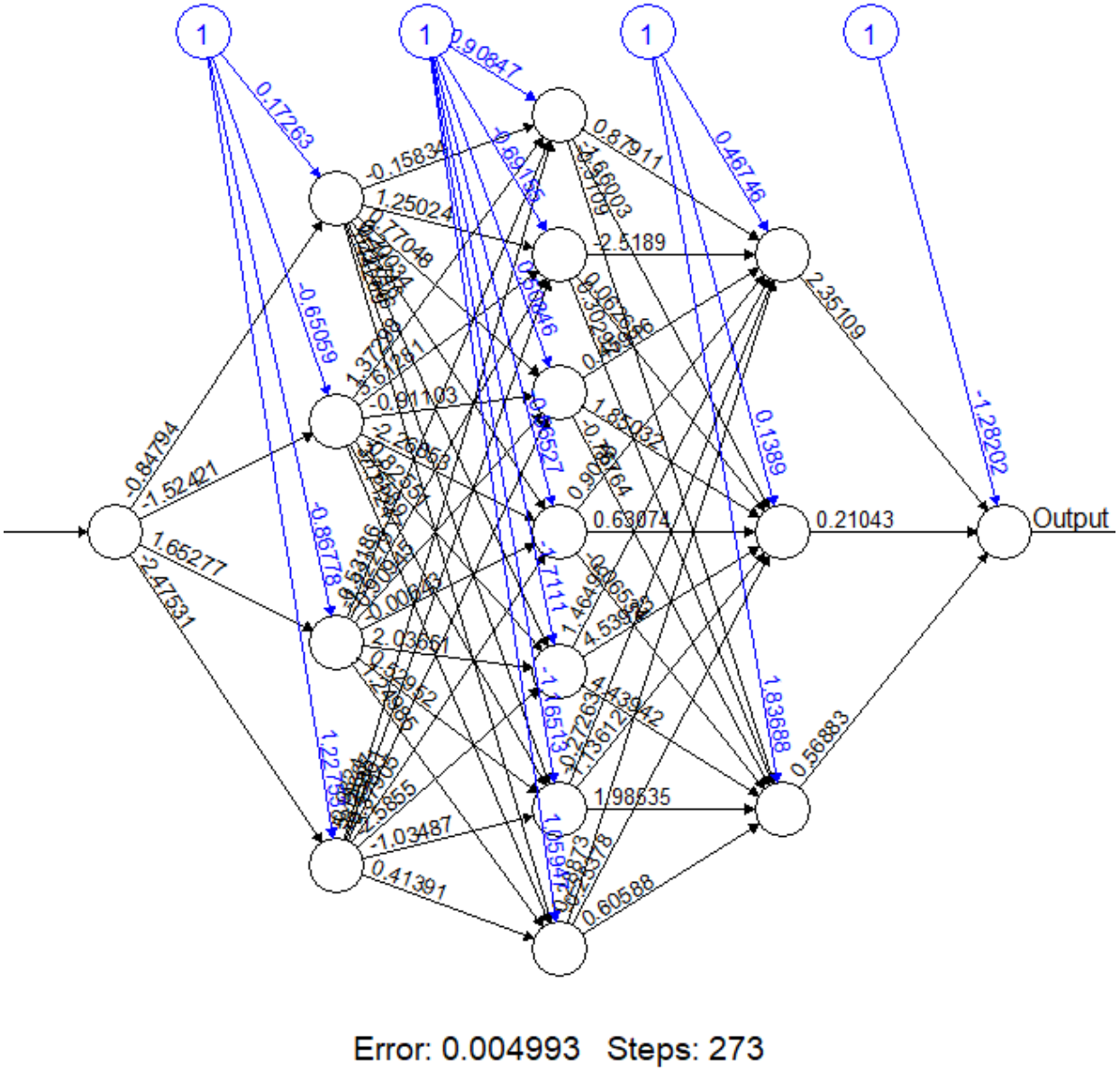
4. Wynik działania

Kod programu dostępny w repozytorium: https://github.com/arturkoziel/APU

Dane wygenerowane do nauki sieci neuronowej miały następującą postać:

	Input	Output
1	3.3307358734	5.287618401
2	4.4548934490	9.276151247
3	2.5375034802	3.556410466
4	3.9495221607	7.204898057
5	3.1902831643	4.929026885
6	3.6452453051	6.188066335
7	1.9892862585	2.703759277
8	4.5603086193	9.778189162
9	2.0122776175	2.735020165
10	1.9691757227	2.676708482
11	3.6751588441	6.281315431
12	1.8329750548	2.500492037
13	3.5695297572	5.958178888
14	4.5564346412	9.759267248
15	4.4618773712	9.308599828
16	2.3785478758	3.284695449
17	2.0539633549	2.792624061
18	0.6311330840	1.371035840
19	0.1733835532	1.090560492
20	2.4661761437	3.431810868
21	3.8280400565	6.780291046
22	3.7775084924	6.611127711
23	2.1219945350	2.889250909
24	3.6987172293	6.355741735
25	4.0027456132	7.399206810
26	0.7938532309	1.487246785
27	2.1679212953	2.956365516
28	2.1953504197	2.997190080
29	4.0006377950	7.391412826
30	1.7939670272	2.452194931
31	4.3785539987	8.928755285
32	3.8577987699	6.881931714
33	2.5899061039	3.650824571
34	1.6702280562	2.305076877
35	4.5684285682	9.817969058
36	4.4647942145	9.322185596
37	3.3135581990	5.242398379
38	4.5374296775	9.666969213
39	3.4138202537	5.511904054
40	2.3876259980	3.299638771
41	4.5803200515	9.876518052
42	0.1036583073	1.053195792
43	0.6860548082	1.409207379
44	2.3159829660	3.183532671
45	4.5168233362	9.567880122
46	3.3258475007	5.274710257
47	3.6973400805	6.351366841
48	4.2399017216	8.330728112
49	2.8205346809	4.097050565
50	1.3206015666	1.935374375

Schemat sieci neuronowej prezentuje się następująco:



Error: 0.004993 Steps: 273

Możemy z niego wyczytać wagi jakie algorytm dobrał dla poszczególnych węzłów. Widzimy także, że błąd wynosi **0.004993** i wynik tak dobry został osiągnięty po **273** iteracjach.

5. Wnioski

W trakcie pracy nad ćwiczeniem zaimplementowano model sieci neuronowej. Dzięki przeprowadzonemu uczeniu zostały ustalone wagi dla poszczególnych węzłów.