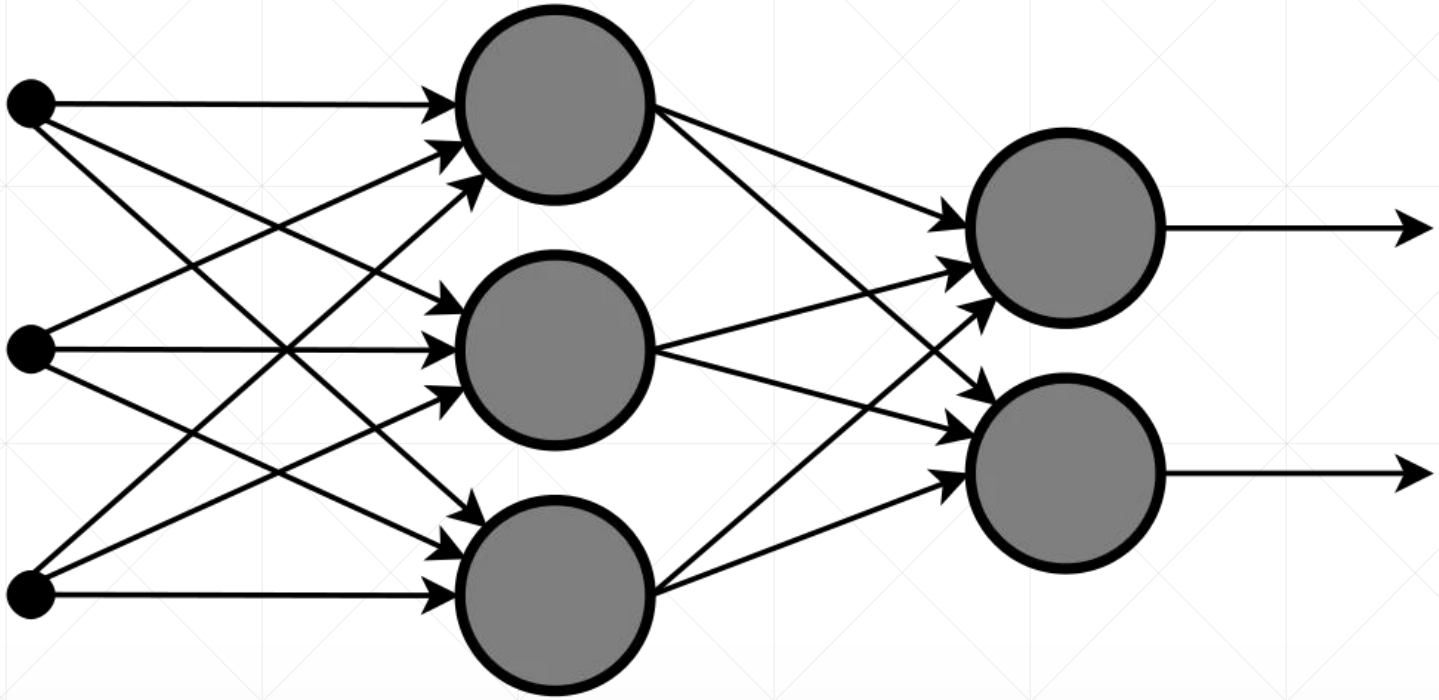
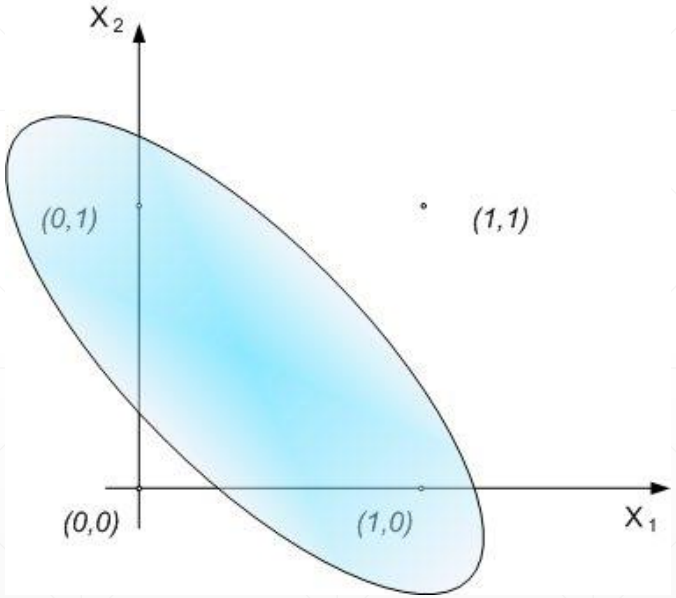


Sieci neuronowe w R i Python

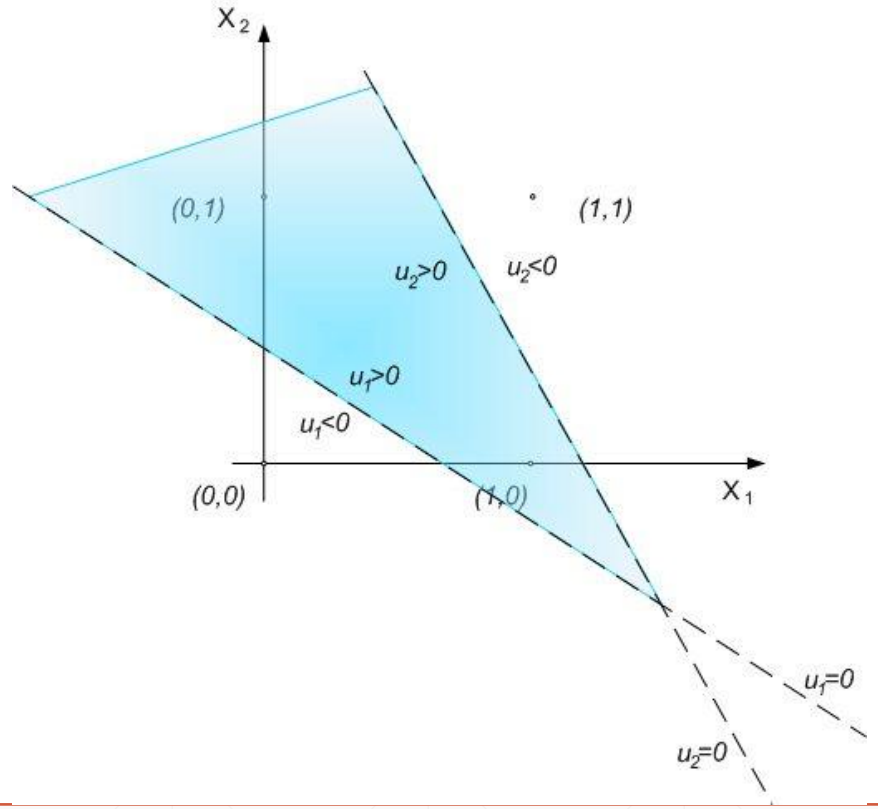
Inga Dyląg

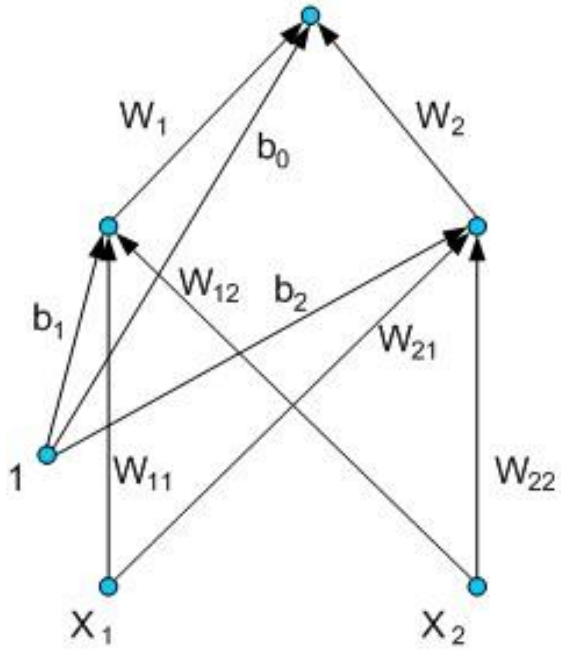
XOR

A	B	A **XOR** B
0	0	0
0	1	1
1	0	1
1	1	0



Struktura sieci mogąca realizować funkcję XOR



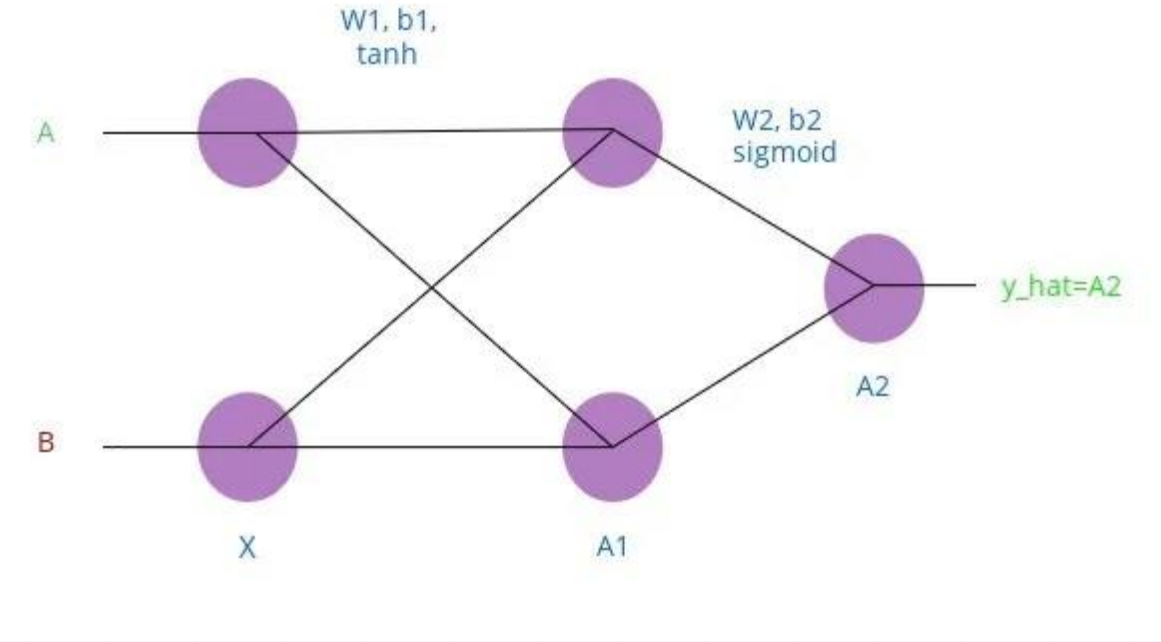


Warstwa wyjściowa

Warstwa ukryta

Warstwa wejściowa

XOR w języku Python




```
A1 = h(W1*X + b1)
A2 = g(W2*A1 + b2)
```

Binary Cross-Entropy Loss

- Dla binarnej klasyfikacji (zadanie klasyfikacji z dwiema klasami - 0 i 1), mamy binarną funkcję straty Cross-Entropy zdefiniowaną jako:

$$
\begin{aligned}
L &= -\sum_{i=1}^{2} t_i \log(p_i) \\
&= -[t_1 \log(p_1) + t_2 \log(p_2)] \\
&= -[t \log(p) + (1-t) \log(1-p)]
\end{aligned}
$$

where t_i is the truth value taking a value 0 or 1 and p_i is the Softmax probability for the i^{th} class. Since we have two classes 1 and 0 we can have $t_1 = 1$ and $t_2 = 0$ and since $p's$ are probabilities then $p_1 + p_2 = 1 \implies p_1 = 1 - p_2$. For the convenience of notation, we can then let $t_1 = t, t_2 = 1 - t, p_1 = p$ and $p_2 = 1 - p$.

Wnioski

Implementacja nn **bez użycia biblioteki**, jej główne zalety to:

- Dostępność pełnej kontroli nad każdym elementem sieci, włącznie z inicjalizacją, warstwami i propagacją
- Pełna elastyczność i możliwość modyfikacji każdego elementu sieci, co pozwala na łatwe dostosowywanie jej do różnych zadań i danych
- Możliwość zrozumienia i nauczenia się algorytmu krok po kroku, co może pomóc w zrozumieniu uczenia maszynowego w ogóle

Implementacja nn z użyciem **biblioteki sklearn**, jej główne zalety to:

- Szybkość implementacji modelu sieci neuronowej, która zwykle wymagałaby więcej kodu
- Możliwość łatwej zmiany hiperparametrów sieci, takich jak liczba neuronów w ukrytej warstwie, funkcja aktywacji i wiele innych
- Automatyczna obsługa różnych solverów i metod optymalizacji co pozwala na łatwe testowanie różnych opcji

Implementacja nn z użyciem **biblioteki neuralnet,** jej główne zalety to:

- Skupia się na prostych modelach sieci neuronowych, co może wpływać na efektywność obliczeniową i czas trenowania modelu, szczególnie dla prostych problemów.
- Łatwy w użyciu i zrozumieniu, szczególnie dla początkujących
- Pozwala na łatwe wizualizowanie sieci neuronowej
- Neuralnet oferuje dokładniejsze informacje na temat uczenia się sieci neuronowej, takie jak wartości wag, wartości błędów

Bibliografia

https://scikit-learn.org/s
https://towardsdatascience.com/how-to-build-a-simple-neural-network-from-scratch-with-python-9f011896d2f3
https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e
https://home.agh.edu.pl/~vlsi/AI/xor_t/glowna.htm
https://home.agh.edu.pl/~horzyk/lectures/biocyb/BIOCYB-SieciNeuronowe.pdf/

https://scikit-learn.org/stable/auto_examples/neural_networks/plot_rbm_logistic_classification.html
#sphx-glr-auto-examples-neural-networks-plot-rbm-logistic-classification-py
https://www.simplilearn.com/restricted-boltzmann-machines-rbms-article

https://cran.r-project.org/web/packages/neuralnet/neuralnet.pdf
https://datascienceplus.com/neuralnet-train-and-test-neural-networks-using-r/
https://journal.r-project.org/archive/2010-1/RJournal_2010-1_Guenther+Fritsch.pdf
https://www.kaggle.com/