Perceptron

Gustavo Aparecido de Souza Viana

***Abstract*— Redes Neurais Artificiais ou mais conhecida como *RNA*, é uma área de machine learning que vem crescendo exponencialmente, com intuíto de desenvolver algoritmos ou aplicações para analisar grande bases de dados e assim tirar conclusões. Esse modelo basedado no conceito de Neurônios, onde possuimos entradas e é retornado "0 ou 1", ou seja, binário, se tornou popular graças ao *Deep Learning*. O objetivo deste trabalho é implementar um modelo simples de Perceptron para realizar a classificação binária da base de dados Iris com apenas dois tipos de flores e para classificação de operadores como *AND, OR e XOR*. Os resultados mostraram que o é possível aplicar o *Perceptron* como classificador mas há uma limitação onde se que os dados não são linearmente separáveis, ou seja, não é possível traçar uma reta para dividilos em dois grupos.**

I. INTRODUÇÃO

A necessidade de classificações está sendo bem comum na atualidade, podemos dar o exemplo de uma empresa que necessita predizer qual estado da máquina (bom, normal, ou ruim) e assim com base no histórico dela e com base nos dados atuais, conseguimos classificar a mesma.

Conseguimos predizer algum informação baseada em uma base conhecida contendo resultados utilizando machine learning ou com métodos estatísticos. É de extrema importância conhecer o ramo de atividade da aplicação pois métodos de machine learning geralmente estão relacionados a rede neural, consquentemente necessitam de mais dados para que as predições tenham uma maior acurácia. De contra partida, métodos estatísticos não necessitam.

Neste trabalho foi abordado a implementação do *Perceptron* proposto por *McCulloch e Pitts* com objetivo de executar tarefas computacionais baseadas no modelo cerebral humano. Este modelo foi aplicado a base de dados "Iris" mas com apenas dois tipos de flores e testado a operadores como *AND, OR e XOR*.

II. CONCEITOS FUNDAMENTAIS

Nesta seção, será apresentado os conceitos básicos utilizados para o *Perceptron*.

A. Perceptron

Como citado na introdução o *Perceptron* [1] segue o modelo neural, mas de âmbito binário baseado na estrutura de neurônios. Dado uma entrada x com N características e assim temos uma saída $f(x)$, como podemos ver na Equação 1, onde o w representa os pesos, x as entradas e b o bias da função.

$$f(x) = \begin{cases} 1 \; se \; w.x + b >= 0 \\ 0 \; caso \; contrário \end{cases} \tag{1}$$

O *Perceptron* utiliza um momento supervisionado onde necessita de treinamento ou como chamado na literatura uma função *fit*. O objetivo desse treinamento é atualizar os valores do pesos fazendo com que tenha uma função linear que separe os grupos afim de classifica-los. Para isso é utiliado a Equação 1, onde w é o peso em questão, n é a taxa de aprendizado, t é o valor desejado e o valor estimado pelo perceptron, e por fim x_i é a entrada.

$$w_i = w_i + n(t - o)x_i \tag{2}$$

Na Figura 1 ilustra o conceito utilizado pelo perceptron.

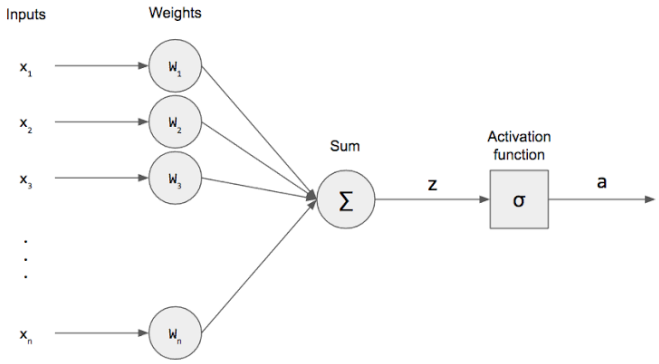

Fig. 1: Exemplo do modelo utilizado pelo *Perceptron*

III. METODOLOGIA

Nesta seção será apresentado a metodologia utilizada para implementar o *Perceptron* juntamente, implementado em Python. O código fonte pode ser encontrado em https://github.com/apparecidoo/master-special-learning-topic.

A metodologia consiste em duas funções, *predict* e *train*. Sendo que a função *train* utiliza a Equação 2, que tem como objetivo atualizar os pesos, e a função *predict* utiliza a Equação 1, que tem como objetivo predizer uma entrada baseada nos pesos encontrados pelo método de treinamento.

IV. EXPERIMENTOS E RESULTADOS

Nesta seção será apresentado os experimentos e seus respectivos resultados.

O experimento consiste na aplicação do *Perceptron* para a base de dados Iris. Além disso, foram feitos testes nos operadores *AND, OR e XOR*. Nos testes citados, foram realizados o treinamento utilizando a taxa de aprendizado $n = 0.1$, e 10 épocas ou iterações.

A Figura 2 e a Figura Figura 3 representa os resultados graficamente obtidos do experimento do operador *AND* e *OR*.

Sendo que X e Y representam as entradas, e os quadrados em vermelho representam o valor 1 ou *True* como saída e os triângulos azuis o valor 0 ou *False* como saída. Ambos experimentos foram classificados corretamente com sucesso.

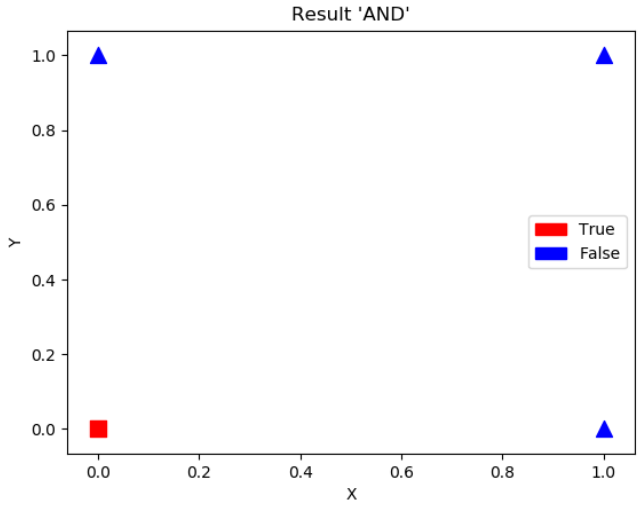

Fig. 2: Resultado do *Perceptron* no operador *AND*

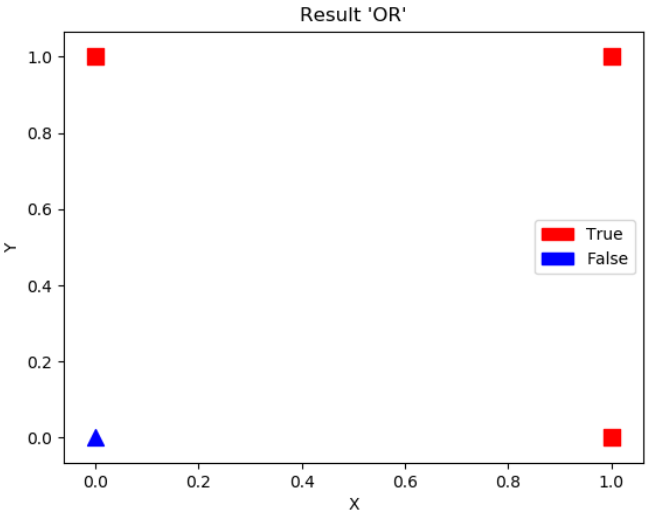

Fig. 3: Resultado do *Perceptron* no operador *OR*

Na Figura 4 representa os resultados graficamente obtidos do experimento do operador *XOR* da mesma forma citada com os operadores *AND* e *OR*. De contrapartida, o resultado do experimento não foi satisfatório, não foi possível classificar de forma correta.

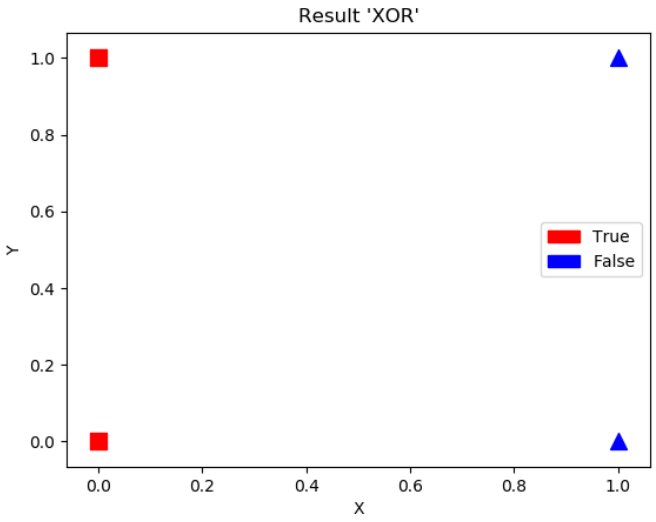

Fig. 4: Resultado do *Perceptron* no operador *XOR*

No último teste, foi aplicado o modelo de *Perceptron* para classificar a base de dados Iris em *Iris Setosa* e *Iris Versicolor*.

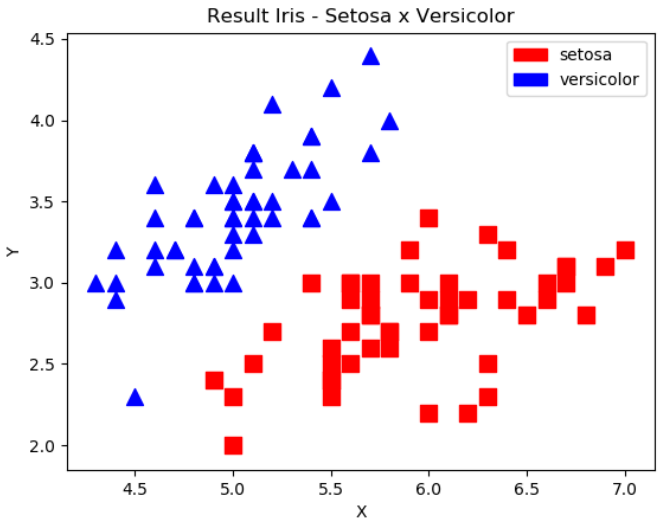

Fig. 5: Resultado do *Perceptron* na base de dados Iris

V. CONCLUSÃO

Neste trabalho foi implementado o modelo de *Perceptron*, sendo que este modelo segue o modelo neural, ou seja, a partir de neurônios podemos executar tarefas computacionais. O *Perceptron* é utilizado como classificador binário a partir de uma equação citada na Sessão dos Conceitos Fundamentais.

Após a análise dos conceitos e resultados obtidos após a aplicação do método *Perceptron*, podemos concluir que tem capacidade de classificar dados mas com uma certa limitação onde pode haver somente dois grupos e além disso nos deparamos com a falha ao classificar o operador *XOR* pois nesse caso necessita mais de uma reta para ser dividido corretamente.

Os próximos passos seriam a implementação do *Perceptron* com vários neurônios, que será caracterizado e tem

como nome *Multi-Layer Perceptron* [2] para ter uma maior abrangência de resolução de problemas.

REFERENCES

[1] Y. Freund and R. E. Schapire. Large margin classification using the perceptron algorithm. In *COLT*, 1998.

[2] S. K. Pal and S. Mitra. Multilayer perceptron, fuzzy sets, and classification. *IEEE transactions on neural networks*, 3 5:683–97, 1992.