UNIVERSIDADE FEDERAL DE UBERLÂNDIA – Campus Santa Mônica
SISTEMAS DE INFORMAÇÃO
Computação Bioinspirada

Nicolli Freitas – 11911BSI241

Victor Hugo Buiatti Marçal - 11911BSI216

Redes Neurais:

Implementando Perceptron para a base de dados Iris

UBERLÂNDIA - MG

OUTUBRO 2023

1. Analisando Resultados para Diferentes Porcentagens de Treino

O código foi executado no ambiente Google Colab e utilizando duas classes de Iris para análise (Iris-setosa e Iris-versicolor), no gráfico de resultados os pontos azuis representam a classe Iris-versicolor e os pontos laranjas Iris-setosa. Além disso, utilizamos apenas 2 características da Iris, o tamanho da pétala e o tamanho da sépala. Na próxima página iremos mostrar os resultados utilizando diferentes porcentagens da base para treino, considerando 10 épocas e uma taxa de aprendizado de 0.01.
Como conclusão, observamos que o Data set Iris é um conjunto linearmente separável, sendo assim o modelo Perceptron funcionou corretamente para a predição de dados, sendo que utilizando 30% e 50% para treino obtivemos um resultado melhor do que utilizando 10%. Além disso, realizamos testes com 100 épocas, e nesse cenário considerando todas as porcentagens (10%, 30% e 50%) obtivemos precisão de 100%.

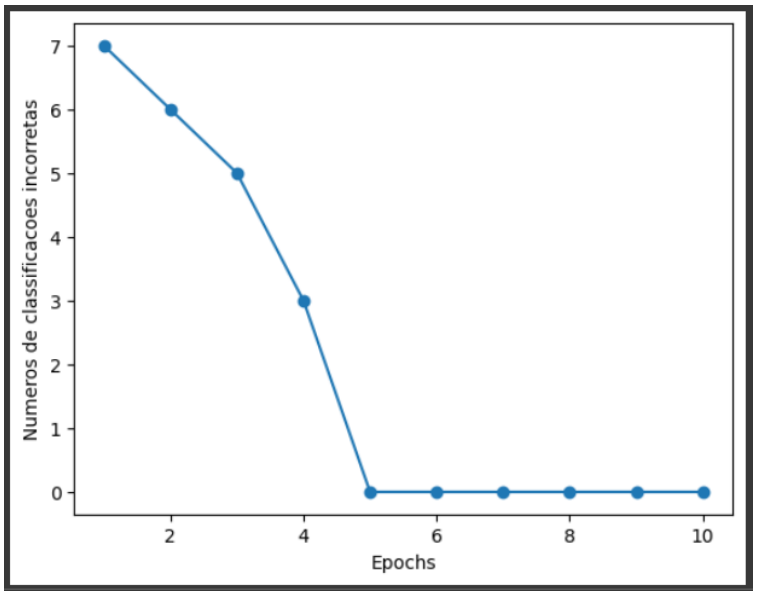
Para inclusão de uma terceira classe o ideal seria utilizar um MLP (Multilayer Perceptron) ou poderia ser usada a abordagem OVA (One VS All) para o modelo Perceptron. A ideia dessa abordagem é transformar um problema de classificação multiclasse em múltiplos problemas de classificação binária, um para cada classe. Dessa forma teremos um conjunto de pesos para cada classe, e a classe com o maior produto escalar é escolhida como a previsão final.

Utilizando 10% da base para treino:

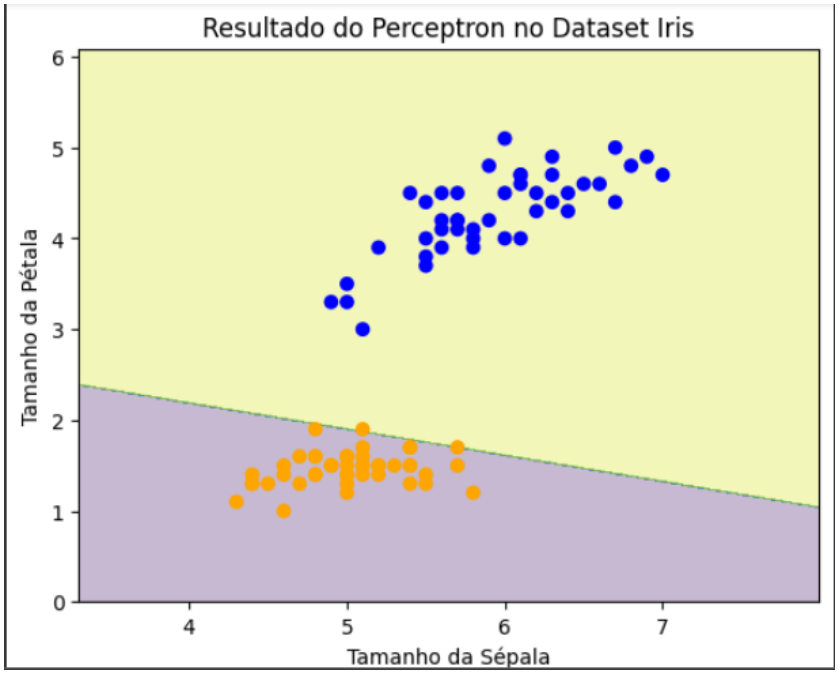
- Precisão de aproximadamente 99%.

```
Pesos Iniciais:  [-0.4351057   0.13121601  0.16419332]
Pesos apos treinamento:  [-0.45510569815229873, 0.03921601401549654, 0.1361933162871151]
Precisão do modelo: 0.9888888888888889
```

- Taxa de erro zerou a partir da 5º interação (época).



- Resultados das predições.



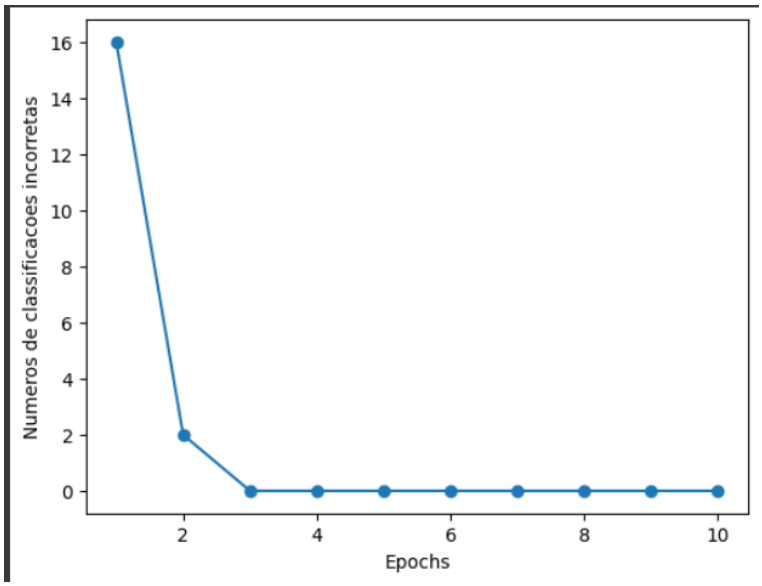
Como a precisão do modelo não chegou a ser de 100% podemos observar que algumas classes preditas não foram separadas corretamente.

Utilizando 30% da base para treino:

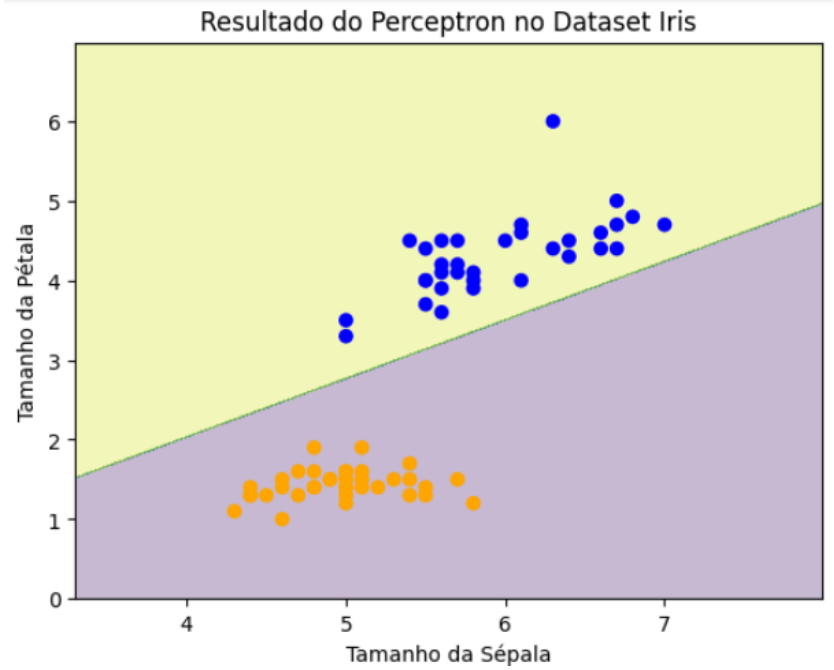
- Precisão de 100%.

```
Pesos Iniciais:  [ 0.2483406   0.1516102  -0.08255697]
Pesos apos treinamento:  [0.16834060213335134, -0.13638979644187627, 0.1854430290432507]
Precisão do modelo: 1.0
```

- Taxa de erros zerou a partir da 3º interação (época).



- Resultados das predições.



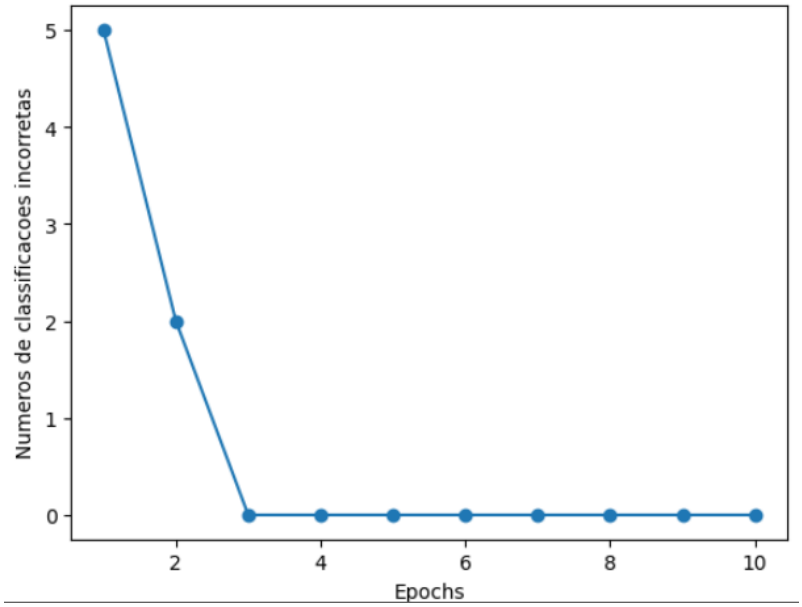
Em relação aos resultados tivemos uma separação correta entre as classes utilizadas para predição.

Utilizando 50% da base para treino:

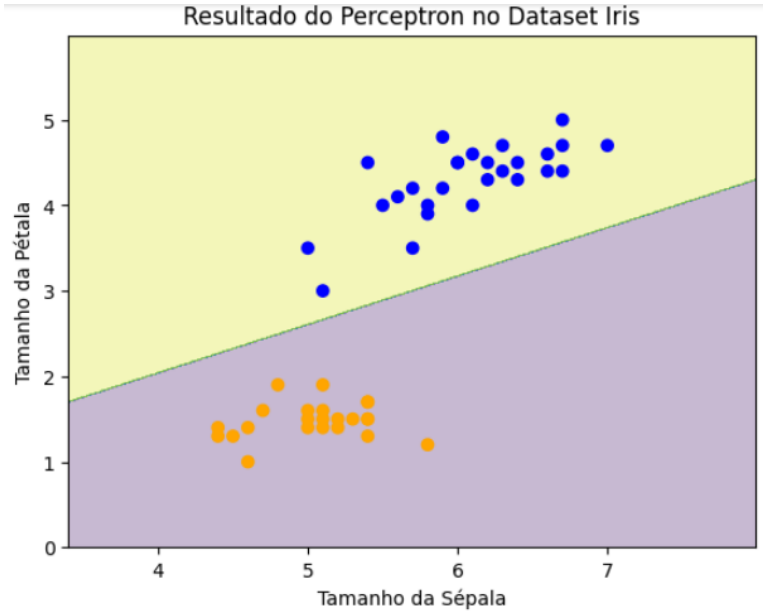
- Precisão de 100%.

```
Pesos Iniciais:  [-0.04038566  0.1303136   0.30233741]
Pesos apos treinamento:  [-0.1003856621197929, -0.13568640194466852, 0.28833741077177244]
Precisão do modelo: 1.0
```

- Taxa de erros zerou a partir da 3º interação (época).



- Resultados das predições.



Em relação aos resultados tivemos uma separação correta entre as classes utilizadas para predição.