Relatório Trabalho Perceptron

Arthur Haickel Nina (18203783)
Universidade Federal de Santa Catarina (UFSC)

I. Problema e implementação

O objetivo do trabalho se tratava de gerar datasets por meio de distribuições normais utilizando média e desvio padrão como centro e raio respectivamente e treinar um perceptron nesses datasets, observando o comportamento do modelo dadas variações em número de classes, balanceamento e a separabilidade de forma linear das classes. Para isso, implementaram-se funções em Python que geravam os pontos com média e desvio padrão do usuário e então, pela biblioteca NumPy [1], eles eram distribuídos normalmente dado um valor padronizado de amostras ou um valor multiplicado em uma das classes, caso o usuário deseje desbalanceá-las. Também foi programado o Perceptron em si para casos binários e multiclasse, que ao final do treinamento, classifica os pontos do dataset e traça as fronteiras de decisão. Para o caso multiclasse, a abordagem de classificação utilizada foi *One vs All*, em que cada neurônio destaca sua classe objetivo do resto, considerando como se houvesse sempre somente 2 e ao fim fazendo as comparações e intersecções entre fronteiras.

II. Validação Cruzada

Utilizou-se KFolds[2] como técnica de validação cruzada (CV), onde se divide o conjunto de dados em pastas (folds) para teste e validação. Optou-se por 5 folds, para cada fold escolhido, ele será utilizado para validação enquanto os outros 4 são utilizados para treinar o modelo. Isto garante que todo o dataset é coberto pela rede, o que pode evitar overfitting em datasets desbalanceados. Para a implementação da CV, utilizou-se o modelo da biblioteca *Scikit Learn* [3], as métricas avaliadas foram acurácia e área sob curva ROC (AUC).

III. Métricas de avaliação

- Acurácia - Taxa de acerto do modelo, medida pela proporção de predições corretas, não leva em consideração balanceamento do dataset, ams é de fácil leitura e dá uma ideia ras, porém geral de como o modelo se comporta em casos mais simplórios.
- Área sob Curva ROC - Descreve de forma generalizada o desempenho do modelo em separar classes ao traçar a taxa de verdadeiros positivos em função da taxa de falsos positivos em diferentes pontos de corte do modelo. Probabilidade do modelo atribuir um ranqueamento mais alto para um objeto positivo do que para um objeto negativo.[2]

IV. Performance

Nesta seção são mostrados resultados de treinamento e validação das redes para as diferentes condições conferidas aos datasets gerados para cada fold de validação cruzada.

A. Perceptron binário

- 100 amostras de cada classe
- Para desbalancear, multiplicaram-se as amostras da classe 0 por **3**
- 300 iterações
- Learning rate = 0.1

Tabela I: Dados de geração do dataset LS

Dado	**Classe 0**	**Classe 1**
Média	5	1
Desvio padrão	0.1	0.1

Para testar um datset não linearmente separável, escolheram-se valores cujos scatters poderiam sobrepor um ao outro.

Tabela II: Dados de geração do dataset não LS

Dado	**Classe 0**	**Classe 1**
Média	5	4
Desvio padrão	0.5	0.1

Tabela III: Acurácia em diferentes condições do dataset

K folds	**B e LS**	**Db e LS**	**B e não LS**	**Db e não LS**
Fold 1	1.0	1.0	0.725	0.275
Fold 2	1.0	1.0	0.725	0.975
Fold 3	1.0	1.0	0.575	0.2875
Fold 4	1.0	1.0	0.725	0.975
Fold 5	1.0	1.0	0.900	0.2875

Legenda: B - balanceado, Db - desbalanceado, LS - linearmente separável

A média para o dataset não linearmente separável cai para 0.73, o que já indica a dificuldade do Perceptron de lidar com esse tipo de problema, quando se desbalanceia este mesmo dataset ele passa a errar com ainda mais frequência, praticamente classificando de forma aleatória.

Tabela IV: AUC ROC em diferentes condições do dataset

K folds	**B e LS**	**Db e LS**	**B e não LS**	**Db e não LS**
Fold 1	1.0	1.0	0.738	0.532
Fold 2	1.0	1.0	0.710	0.984
Fold 3	1.0	1.0	0.575	0.500
Fold 4	1.0	1.0	0.750	0.963
Fold 5	1.0	1.0	0.889	0.500

Para a AUC ROC média, ainda se manteve um certo valor, indo de 0.732 para 0.696, o que indica que ele é capaz de trabalhar com datasets desbalanceados e não linearmente separáveis de forma bem limitada, porém, por o erro ser tão grande nesse caso, a fronteira de decisão nem mesmo pôde ser traçada para o pior cenário.

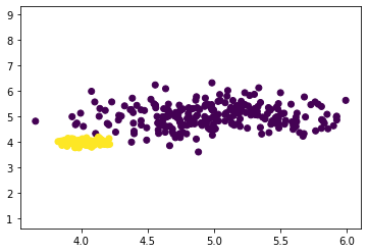

Figura 1: Ausência de fronteiras de decisão.

Uma solução para o problema foi iterá-lo o dobro de vezes, isso permitiu um melhor ajuste de pesos e viés e o perceptron foi capaz de traçar uma fronteira com erro consideravelmente mais baixo.

- Acurácia média = 0.83
- AUC ROC média = 0.85

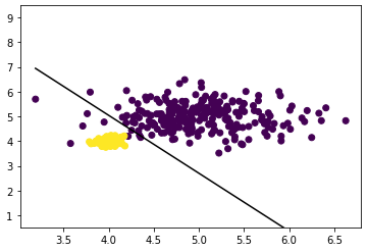

Figura 2: Fronteiras de decisão.

B. Perceptron multiclasse

- 100 amostras de cada classe
- 3 classes
- Para desbalancear, multiplicaram-se as amostras da classe 0 por **3**
- 300 iterações
- Learning rate = 0.1

Tabela V: Dados de geração do dataset LS

Dado	Classe 0	Classe 1	Classe 2
Média	(2,2)	(8,8)	(5,15)
Desvio padrão	0.1	0.5	0.5

No caso do perceptron multiclasse, a coordenada y do ponto central do scatter conta como fator de separabilidade das classes, para o caso não linearmente separável, utilizaram-se valores que se sobrepunham marginalmente, mas também eram paralelos.

Tabela VI: Dados de geração do dataset LS

Dado	Classe 0	Classe 1	Classe 2
Média	(2,2)	(8,8)	(5,5)
Desvio padrão	0.1	0.5	0.9

É possível observar um fenômeno interessante a partir dos datasets não linearmente separáveis: a acurácia roda em torno de 66% em vários casos, isso se deve ao fato de que as classes estão alinhadas, então na abordagem *One vs All* o neurônio observa as duas classes que não a objetivo como 1 só, isto é $\frac{2}{3}$ do dataset são reconhecidos como a mesma classe; e isso se confirma na matriz de confusão.

Tabela VII: Acurácia em diferentes condições do dataset

K folds	B e LS	Db e LS	B e não LS	Db e não LS
Fold 1	1.0	1.0	0.650	0.36
Fold 2	1.0	1.0	0.300	0.19
Fold 3	0.95	1.0	0.433	0.55
Fold 4	1.0	1.0	0.783	0.81
Fold 5	1.0	1.0	0.633	0.78

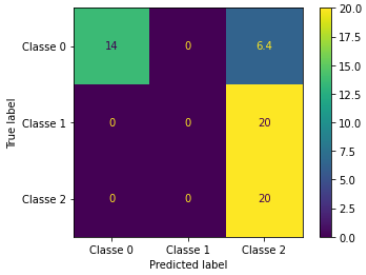

Figura 3: Matriz de confusão caso 3.

Tabela VIII: AUC ROC em diferentes condições do dataset

K folds	B e LS	Db e LS	B e não LS	Db e não LS
Fold 1	1.0	1.0	0.860	0.886
Fold 2	1.0	1.0	0.830	0.939
Fold 3	0.96	0.999	0.794	0.899
Fold 4	1.0	1.0	0.852	0.942
Fold 5	1.0	1.0	0.825	0.904

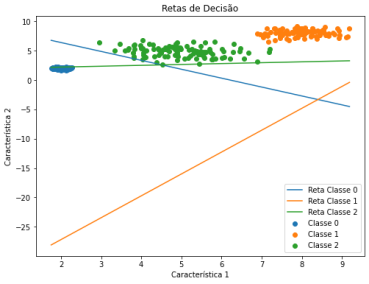

Figura 4: Fronteiras de decisão incorretas no caso 4.

Por outro lado, a AUC demonstra a verdadeira capacidade da rede de separar objetos de classes distintas, mesmo elas estando incorretamente agrupadas no plano e na matriz de confusão como uma classe única.

Referências

[1] "Numpy: The fundamental package for scientific computing with python." NumPy Steering Council, Open Source. [Online]. Available: https://numpy.org

[2] L. WEIHMANN and P. ANDRETTA, "Tópicos em aprendizado de máquina - notas de aula," 2023.

[3] "scikit-learn: Machine learning in python." Open Source. [Online]. Available: https://scikit-learn.org/stable/index.html