Trabalho Prático
CPS844 - 2024.P1
Gabriel Gazola Milan

# Introdução

Esse trabalho foi desenvolvido para a disciplina de Inteligência Computacional I (CPS844) durante o período 2024.P1. Tanto o enunciado desse trabalho como o código utilizado e as imagens geradas poderão ser encontrados publicamente no repositório https://github.com/gabriel-milan/cps844.

# Perceptron

O código utilizado para essa seção pode ser encontrado no arquivo `perceptron.py`. Para obtenção dos valores médios mencionados a seguir, foram realizadas 1000 execuções de cada experimento (N=10 ou N=100), com *random seeds* diferentes em cada uma.

Na questão (1), o resultado de média de iterações para convergir foi 9,26. O resultado mais próximo seria a alternativa b, 15. Como o PLA ajusta seus parâmetros um ponto por vez, é natural que o número médio de iterações que demora para convergir seja proporcional ao número de amostras do conjunto de treino.

Já para a questão (2), a abordagem utilizada foi a de gerar 10.000 novos pontos e testá-los na função aprendida pelo algoritmo. Nesse caso, o valor médio foi de 0.102768, sendo a alternativa c a mais próxima, 0.1. Por se tratarem de poucos pontos para treinamento (N=10), esses podem ser uma má representação dos pontos fora da amostra, fazendo que o modelo aprenda uma função que não represente razoavelmente bem a função objetivo. Isso pode ser observado em imagens geradas com esses 10.000 novos pontos, como na Figura 1.

Para a questão (3), o valor obtido foi 79,36, mais próximo da alternativa b, 100. De forma similar à questão (1), o número médio de iterações cresce proporcionalmente ao número de pontos de treinamento.

Na questão (4), o valor obtido foi 0.013227, se aproximando da alternativa b, 0.01. Similarmente à questão (2), aqui, por termos mais pontos de treinamento, representando melhor o universo de onde essas amostras foram obtidas, temos uma menor probabilidade de classificação errada em pontos fora da amostra. A Figura 2 deixa claro que a função aprendida com N=100 se aproxima muito mais da objetivo.

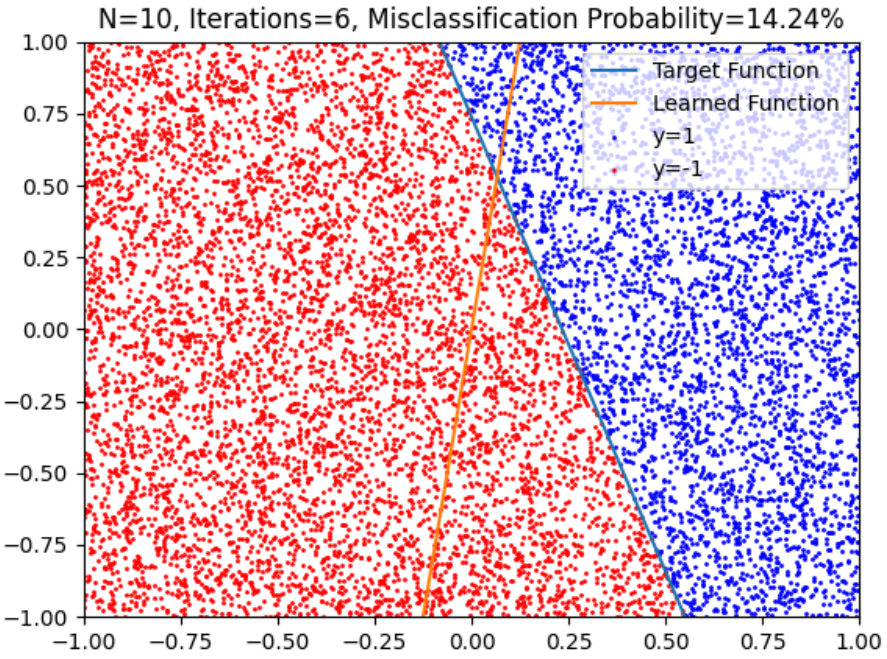


Figura 1 - Exemplo de classificação incorreta com pontos fora da amostra (N=10)

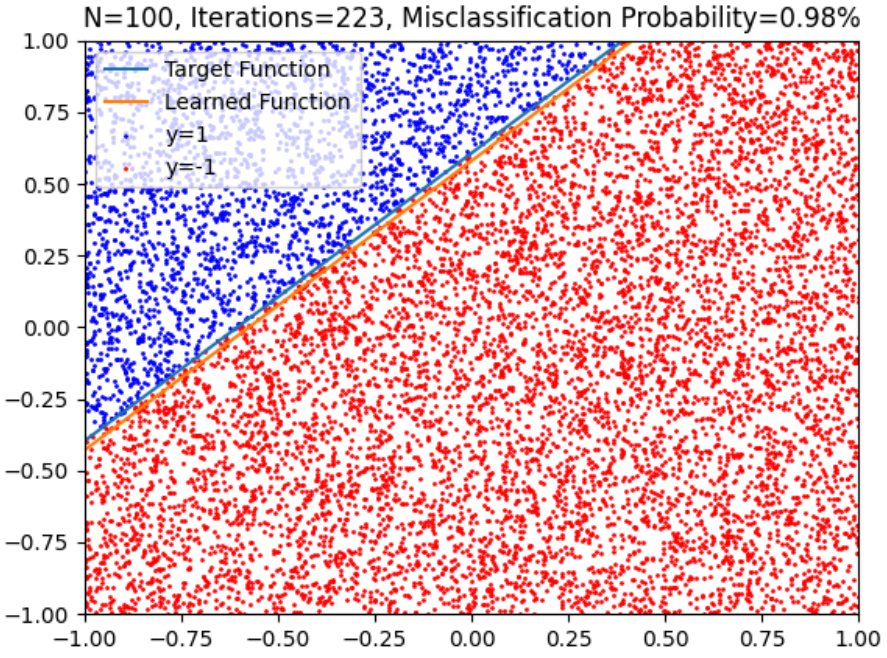


Figura 2 - Exemplo de classificação incorreta com pontos fora da amostra (N=100)

Finalmente, na questão (5), podemos relacionar o número de iterações até a convergência e a probabilidade de classificação incorreta. Pela inequação *generalization bound* $E_{\text{out}}(g) \leq E_{\text{in}}(g) + \sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}}$, enunciada no material do curso, vê-se que a probabilidade de classificação incorreta em pontos fora da amostra está relacionada com a probabilidade de classificação incorreta na amostra somada a um termo que cresce em função de $\sqrt{\frac{1}{2N}}$. Como, no caso desse exercício, as amostras são linearmente separáveis, erro dentro da amostra seria zero, tornando o erro fora da amostra proporcional somente a $\sqrt{\frac{1}{2N}}$. Já com relação ao número de iterações, é diretamente proporcional a N, dada a natureza do algoritmo de iterar os pontos um a um fazendo correções nos pesos.

# Regressão linear

O código utilizado para essa seção pode ser encontrado no arquivo `linear_regression.py`. Para obtenção dos valores médios mencionados a seguir, foram realizadas 1000 execuções de cada experimento, com *random seeds* diferentes em cada uma.

Na questão (1), obteve-se o valor 0.03865, aproximando-se melhor da alternativa c, 0.01. Observa-se que, diferentemente do PLA, o erro dentro da amostra é diferente de zero. Isso acontece pelo fato de que é executada uma operação em uma única iteração, utilizando todo o volume de dados de uma vez.

Já na questão (2), o erro fora da amostra obtido foi de 0.046892, aproximando-se melhor novamente da alternativa c, 0.01. É possível notar que o erro fora da amostra aqui é 0.008242 maior que o erro dentro da amostra, valor bem próximo do fator $\sqrt{\frac{1}{2N}}$ encontrado na questão (5) da seção Perceptron, que denota essa relação entre o erro dentro da amostra e o erro fora da amostra.

Na questão (3), o valor médio de iterações encontrado foi de 3.919, aproximando-se melhor da alternativa a, 1. É notável a redução do número médio de iterações quando são fornecidos os pesos iniciais calculados pela regressão linear. No caso, houve até alguns casos, como na Figura 3, em que não foram necessárias iterações para convergir. Esses pesos iniciais possuem uma tendência a se aproximar da solução final e, portanto, podem ser utilizados para acelerar o PLA.

Por fim, na questão (4), temos os seguintes resultados:

(a) Ein = 0.17536; Eout = 0.115014
(b) Ein = 0.12844; Eout = 0.061614
(c) Ein = 0.15427; Eout = 0.085992
(d) Ein = 0.12436; Eout = 0.053383

Como o número de pontos de treinamento (N1) era sempre 100, é notável que, ao limitarmos o número de iterações (i) para algo muito inferior a N1, como i=10 em (a) e (c),

os erros tendem a ser maiores do que em valores de i mais próximos de N1, como em (b) e (d).

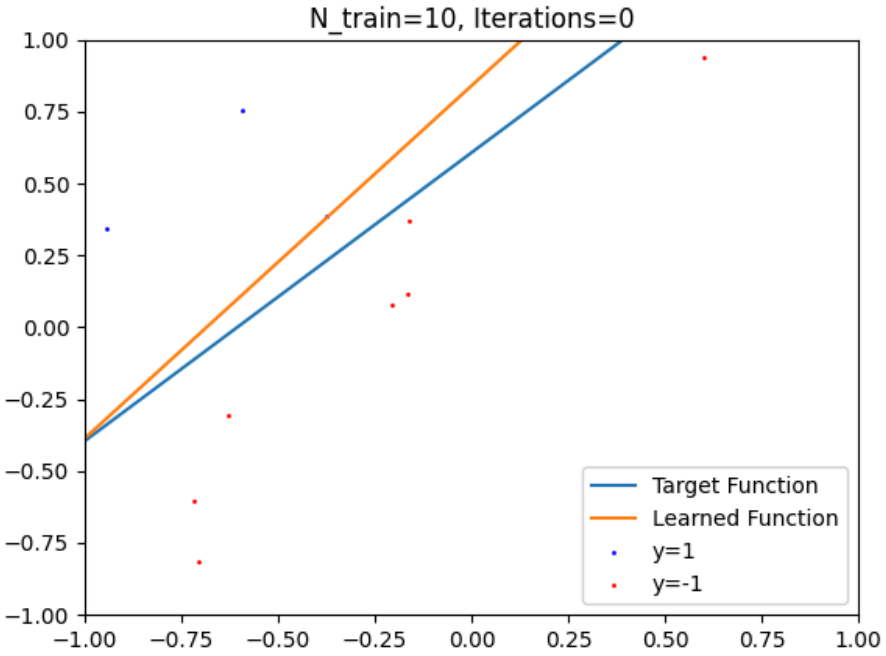


Figura 3 - PLA com pesos iniciais calculados via LR

Por outro lado, quando são fornecidos pesos iniciais calculados via regressão linear, como em (c) e (d), o número de iterações tende a fazer menor diferença, dada a tendência a convergir mais rápido explicitada na questão (3). A diferença da qualidade das funções aprendidas quando o peso inicial é fornecido e há grande limitação do número de iterações também é notável (Figuras 4 e 5).

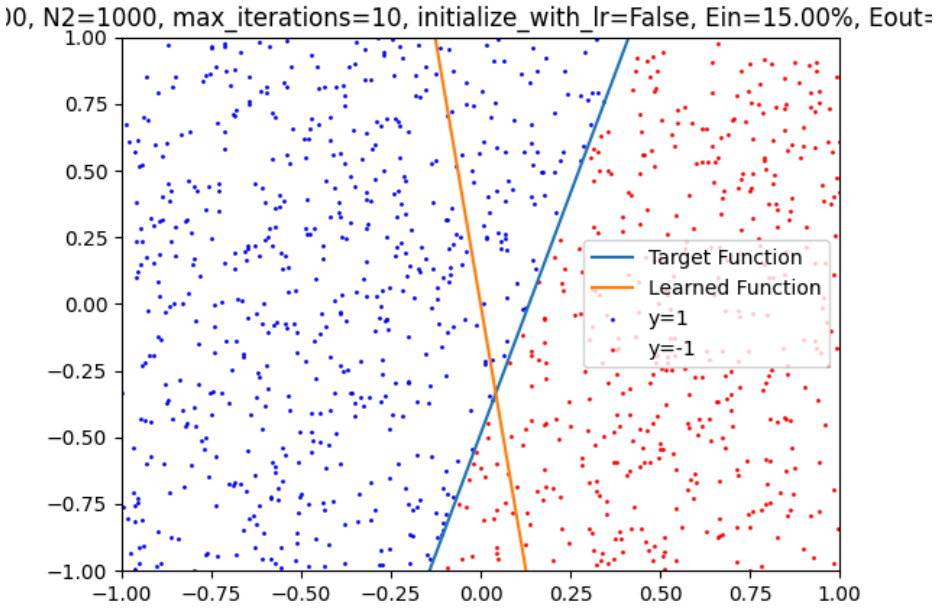


Figura 4 - Pocket PLA com (N1=100, i=10) e sem inicialização de pesos

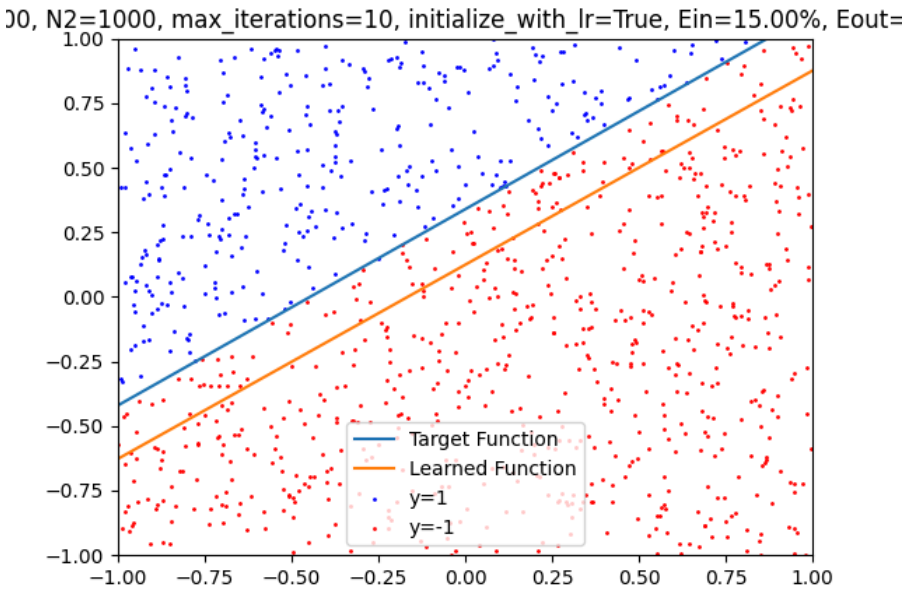


Figura 5 - Pocket PLA com (N1=100, i=10) e inicialização de pesos com regressão linear

## Regressão Não-Linear

O código utilizado para essa seção pode ser encontrado no arquivo `nonlinear_regression.py`. Para obtenção dos valores médios mencionados a seguir, foram realizadas 1000 execuções de cada experimento, com *random seeds* diferentes em cada uma.

Na questão (1), o valor médio obtido para o erro da classificação dentro da amostra foi 0.5084, estando mais próximo então da alternativa d, 0.5. Observando o scatter plot (Figura 6) é possível notar que a função objetivo se trata de uma função não-linear. Por esse motivo, sem nenhuma transformação nos dados, torna-se insuficiente a aplicação do modelo de regressão linear.

Já na questão (2), a equação média obtida para a função após a transformação foi a seguinte:

$g(x_1, x_2) = sign(-1.2403 - 0.0001x_1 + 0.0012x_2 + 0.0011x_1x_2 + 1.9449x_1^2 + 1.9480x_2^2$

sendo ela mais próxima da alternativa a. Aqui, com a transformação, foram incluídos os valores dos quadrados das coordenadas, sendo possível então desenhar elipses. Observando novamente a Figura 6, nota-se que a distribuição de fato se assemelha a uma elipse. Os valores mais significativos da equação também apontam para uma grande relevância dos atributos dos quadrados das coordenadas.

Por fim, na questão (3), o valor obtido para o erro médio fora da amostra foi 0.0299, sendo então mais próximo da alternativa a, 0. Conforme dito na questão anterior, a inclusão desses novos atributos possibilitou o ajuste de uma função que compreenda bem o universo de onde as amostras foram extraídas. Sendo assim, a redução drástica do erro médio nesse caso era esperada.

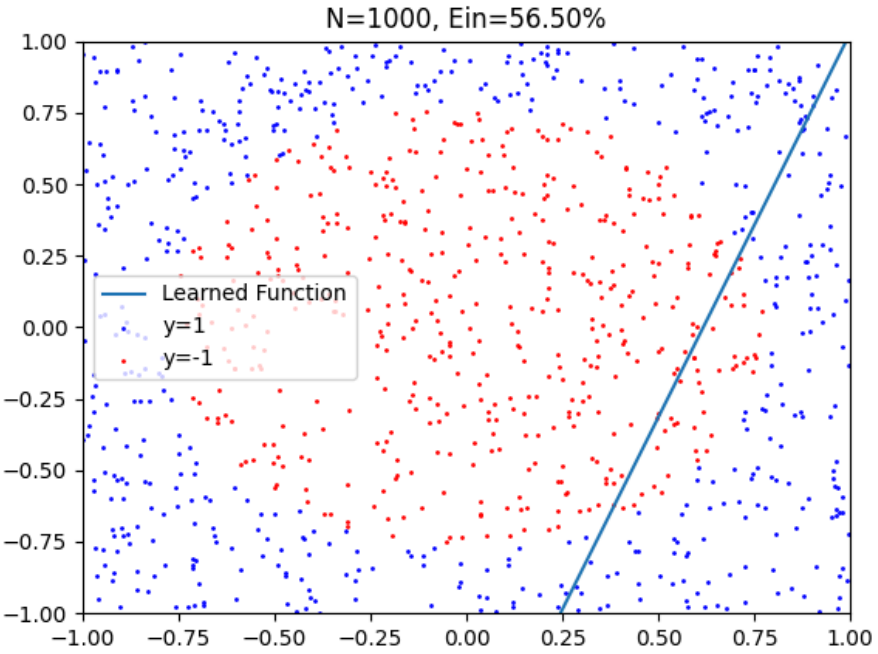


Figura 6 - Scatter plot do primeiro experimento da seção “Regressão Não-Linear”