UNIVERSIDADE FEDERAL
DO RIO DE JANEIRO

Inteligência Computacional I – CPS844

Relatório Trabalho Prático

Aluno: Sylvio Silveira da Costa Mello
DRE: 119093388

Perceptron

1 – Resultado obtido: 9.441. Resposta: b. Esse número relativamente baixo de iterações pode ser explicado pela simplicidade do problema e pelo fato de que o PLA é eficiente para conjuntos de dados que são linearmente separáveis ou quase linearmente separáveis com um pequeno número de pontos. No caso de N=10, o espaço de busca para encontrar uma solução que separe perfeitamente os pontos é limitado, permitindo que o algoritmo convirja rapidamente.

2 – Resultado obtido: 0.1051567. Reposta: c. Esse nível de erro pode ser atribuído à simplicidade do modelo de perceptron e à pequena quantidade de dados de treinamento, que muitas vezes não consegue capturar toda a variabilidade dos dados. Em termos de machine learning, essa probabilidade de erro reflete o trade-off entre a capacidade do modelo e a quantidade de dados disponíveis para treinamento. Com apenas 10 pontos, o PLA pode não ser capaz de generalizar de forma boa para novos pontos fora da amostra de treinamento.

3 – Resultado obtido: 105.786. Resposta: b. Esse número de iterações reflete a maior complexidade do problema quando o número de pontos de treinamento aumenta, fazendo com que o algoritmo precise de mais ajustes para encontrar uma fronteira de decisão que separe perfeitamente os dados. Em termos de machine learning, este resultado demonstra como o aumento no tamanho do conjunto de dados impacta a eficiência do algoritmo. Com mais pontos, a quantidade de possíveis ajustes nos pesos aumenta, o que pode resultar em mais iterações necessárias para alcançar a convergência.

4 – Resultado obtido: 0.0132319. Resposta: b. Este resultado indica que, em média, cerca de 1.32% dos pontos são classificados de maneira diferente pela função target f e pela hipótese aprendida g pelo PLA. Esse nível relativamente baixo de erro sugere que, com um conjunto maior de pontos de treinamento, o PLA consegue aprender uma hipótese que generaliza bem para novos dados. No contexto de machine learning, essa probabilidade de erro reflete uma maior eficácia do modelo quando se dispõe de um número maior de pontos de treinamento. Com N=100, o PLA tem acesso a mais informações sobre a distribuição dos dados, permitindo que ele ajuste a fronteira de decisão de forma mais precisa. Comparando esse resultado com os itens anteriores, é possível perceber que a probabilidade de erro foi menor com um número maior de

pontos, o que indica que o perceptron está generalizando melhor (e consequentemente errando menos) com um número maior de pontos.

5 – Baseando-se nos resultados obtidos empiricamente, para N pequeno: Quando N é pequeno (por exemplo, N=10), o PLA tende a convergir rapidamente (poucas iterações), mas P[f(x)=g(x)] pode ser relativamente alto devido à limitação de informações nos dados de treinamento. Em contrapartida, para N alto: Conforme N aumenta (no caso apresentado: N = 100) o número de iterações até a convergência aumenta, mas P[f(x) ≠ g(x)] tende a diminuir, indicando uma melhor generalização do modelo. Dessa forma, embora não seja possível estabelecer uma regra fixa, podemos nos basear nos resultados e inferir que: aumentar N significa aumentar o número de iterações pois significa aumentar a complexidade no ajuste da fronteira de decisão e, além disso, aumentar N também significa estabelecer uma tendência de diminuir $P[f(x) \neq g(x)]$ pois o modelo possui mais informações para aprender e generalizar melhor.

Gráficos para o Perceptron:

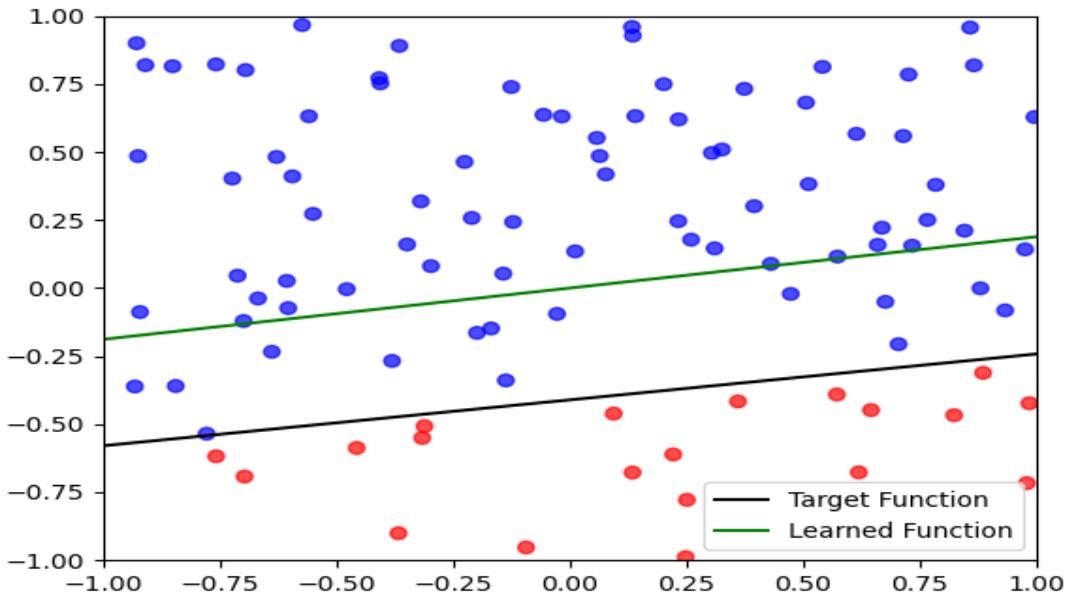

Fig 1. N = 10

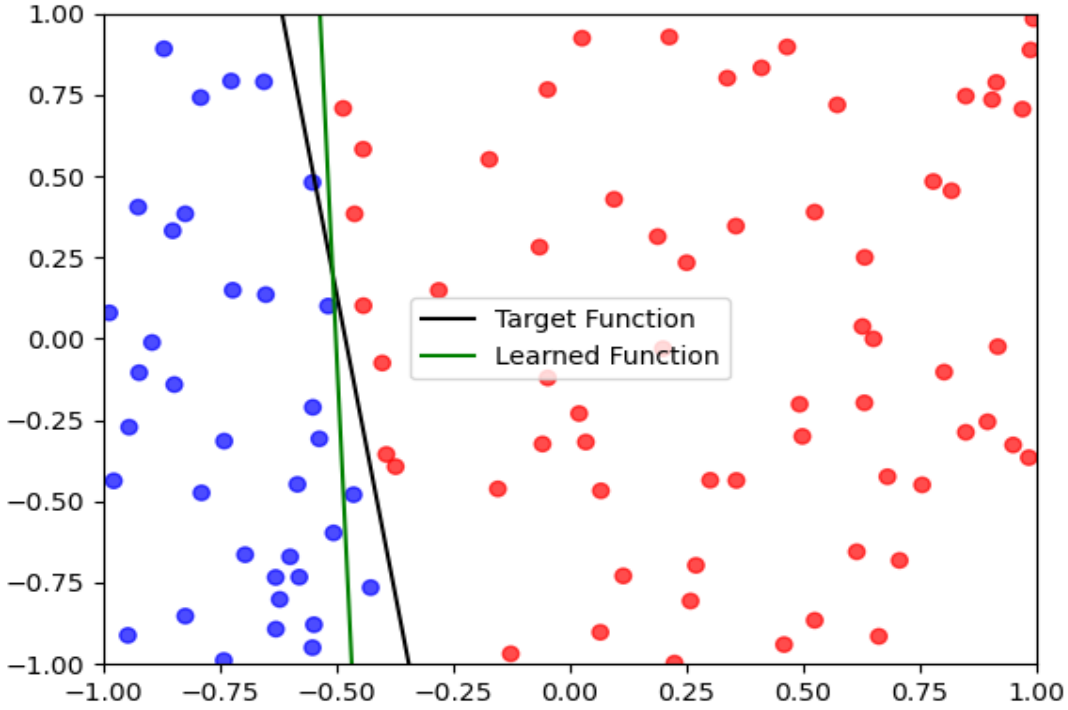

Fig 2. N = 100

Regressão Linear

1 - Resultado obtido: 0.027800000000000002. Resposta: c. Esse valor de erro, o qual não é extremamente baixo, é relativamente pequeno e demonstra que a Regressão Linear consegue ajustar uma fronteira de decisão que se aproxima de maneira aceitável da função target, apesar da possível presença de ruído nos dados. Em termos de machine learning, isso reflete a eficácia do modelo em aprender a estrutura subjacente dos dados quando os pontos são distribuídos de maneira linearmente separável ou quase linearmente separável. A presença de ruído e a linearidade da função target influenciam diretamente o desempenho do modelo, mas a Regressão Linear ainda consegue minimizar os erros de classificação dentro da amostra. Este resultado ilustra a capacidade do modelo de generalizar a partir de um conjunto de dados de tamanho razoável.

2 – Resultado obtido: 0.049289. Resposta: d. A diferença entre Ein e Eout reflete a capacidade de generalização do modelo. No contexto de aprendizado de máquina, este resultado sugere que a Regressão Linear conseguiu capturar a estrutura subjacente dos dados de treinamento, mas ainda apresenta uma certa taxa de erro ao generalizar para novos dados, possivelmente devido à presença de ruído e à simplicidade da função target. A capacidade de generalização de um modelo é crucial para seu desempenho em aplicações reais, e o Eout relativamente baixo indica que a Regressão Linear é eficaz para este problema específico. No entanto, a diferença notável entre Ein e Eout destaca a importância de utilizar técnicas adicionais, como validação cruzada e regularização, para melhorar a robustez e a precisão do modelo em cenários mais complexos e ruidosos.

3 – Resultado obtido: 3.658. Resposta: a. Este resultado pode ser atribuído à simplicidade do problema e ao pequeno número de pontos de treinamento, o que facilita o processo de encontrar uma fronteira de decisão que separe perfeitamente os pontos. Mas além disso, a atribuição da tarefa de gerar os pesos para a regressão linear também facilitou o trabalho do PLA, ou seja, essa rápida convergência do PLA quando inicializado com os pesos da Regressão Linear sugere que a solução inicial fornecida pela Regressão Linear está muito próxima da solução ideal.

4 – Resultados obtidos para cada um dos casos:

(a) Inicializando com 0, i = 10; N1 = 100; N2 = 1000.

Média de E_in: 0.17298000000000002

Desvio padrão de E_in: 0.04634781116730325

Média de E_out: 0.11264300000000001

Desvio padrão de E_out: 0.06351808837646171

(b) Inicializando com 0, i = 50; N1 = 100; N2 = 1000.

Média de E_in: 0.12949000000000002

Desvio padrão de E_in: 0.02658269926098552

Média de E_out: 0.060843

Desvio padrão de E_out: 0.039620554652856646

(c) Inicializando com Regressão Linear, i = 10; N1 = 100; N2 = 1000.

Média de E_in: 0.12937

Desvio padrão de E_in: 0.023492617989487673

Média de E_out: 0.053623000000000004

Desvio padrão de E_out: 0.03237404625622197

(d) Inicializando com Regressão Linear, i = 50; N1 = 100; N2 = 1000.

Média de E_in: 0.11958

Desvio padrão de E_in: 0.01866075025287033

Média de E_out: 0.04574600000000001

Desvio padrão de E_out: 0.029229907355309902

Será feito um parágrafo de explicação para cada caso:

a) Nesta configuração, os pesos foram inicializados com zero e o algoritmo foi executado por 10 iterações. A média de Ein relativamente alta indica que o modelo teve dificuldade em ajustar a fronteira de decisão com apenas 10 iterações, especialmente em um conjunto de dados não-linearmente separável com 10% de rótulos invertidos. O Eout também é elevado, refletindo a dificuldade do modelo em generalizar para novos dados.

b) Com 50 iterações, o modelo melhorou significativamente, como evidenciado pela redução tanto de Ein quanto de Eout. Este resultado demonstra que, com mais iterações, o pocket PLA consegue ajustar melhor a fronteira de decisão, mesmo em um cenário com dados ruidosos e não-linearmente separáveis.

c) Inicializar os pesos com os resultados da Regressão Linear e executar o PLA por 10 iterações resultou em uma melhora significativa em relação à inicialização com zeros e 10 iterações. A Regressão Linear fornece uma boa aproximação inicial, reduzindo o erro dentro da amostra e melhorando a generalização, conforme evidenciado pelo menor Eout.

d) Esta configuração combina a inicialização com Regressão Linear e 50 iterações do PLA, resultando nos melhores desempenhos tanto em Ein quanto em Eout. A Regressão Linear fornece uma boa aproximação inicial, e as 50 iterações adicionais permitem que o PLA refine a fronteira de decisão, melhorando ainda mais a classificação dentro da amostra e a generalização.

Gráficos Regressão Linear

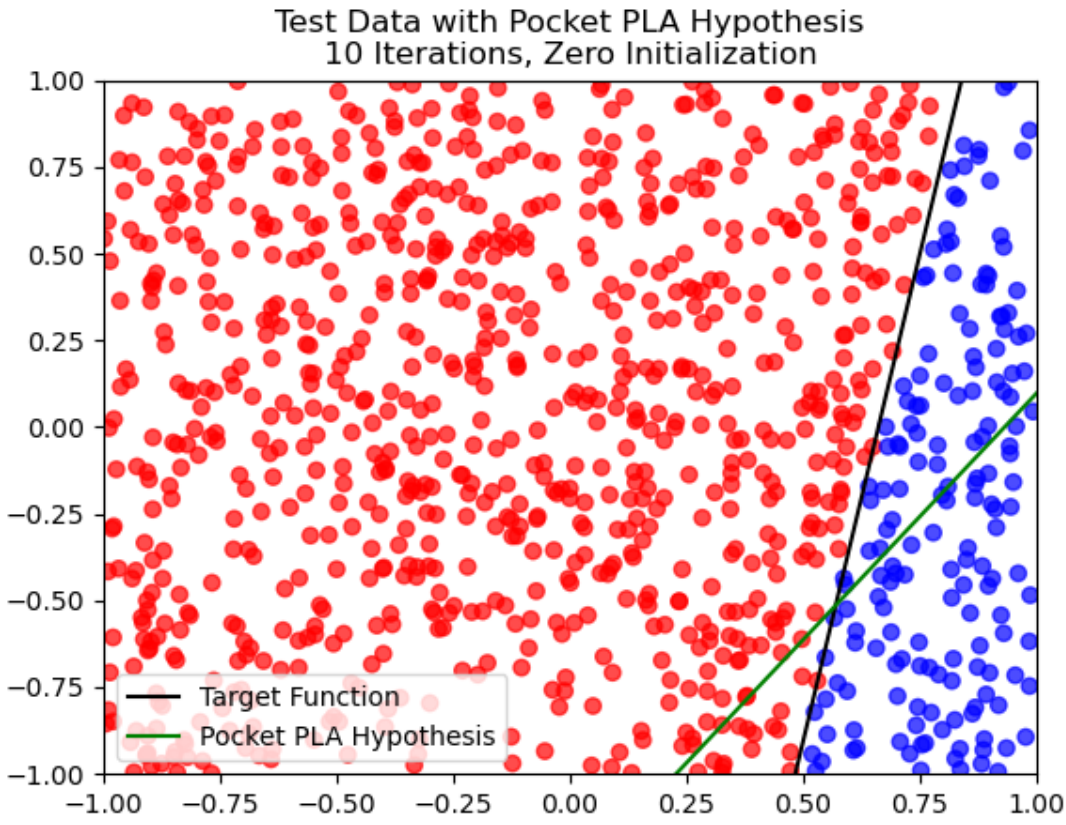

Fig 3. i=10, w=0

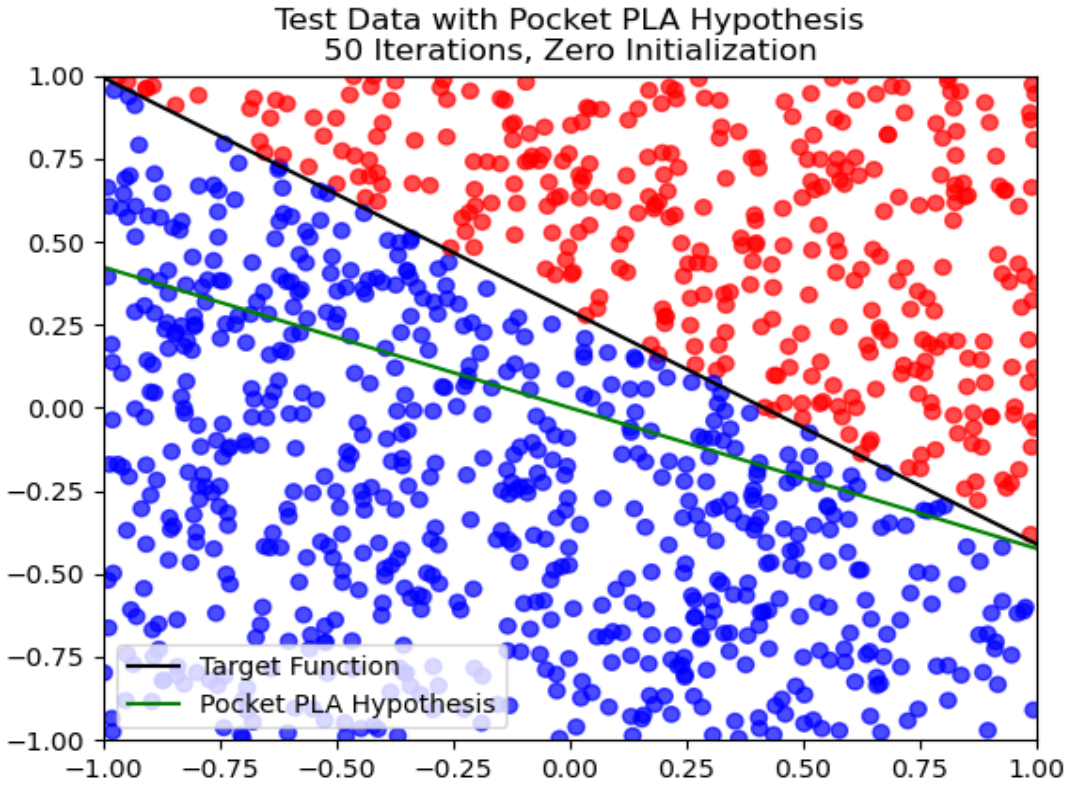

Fig 4. i = 50, w = 0

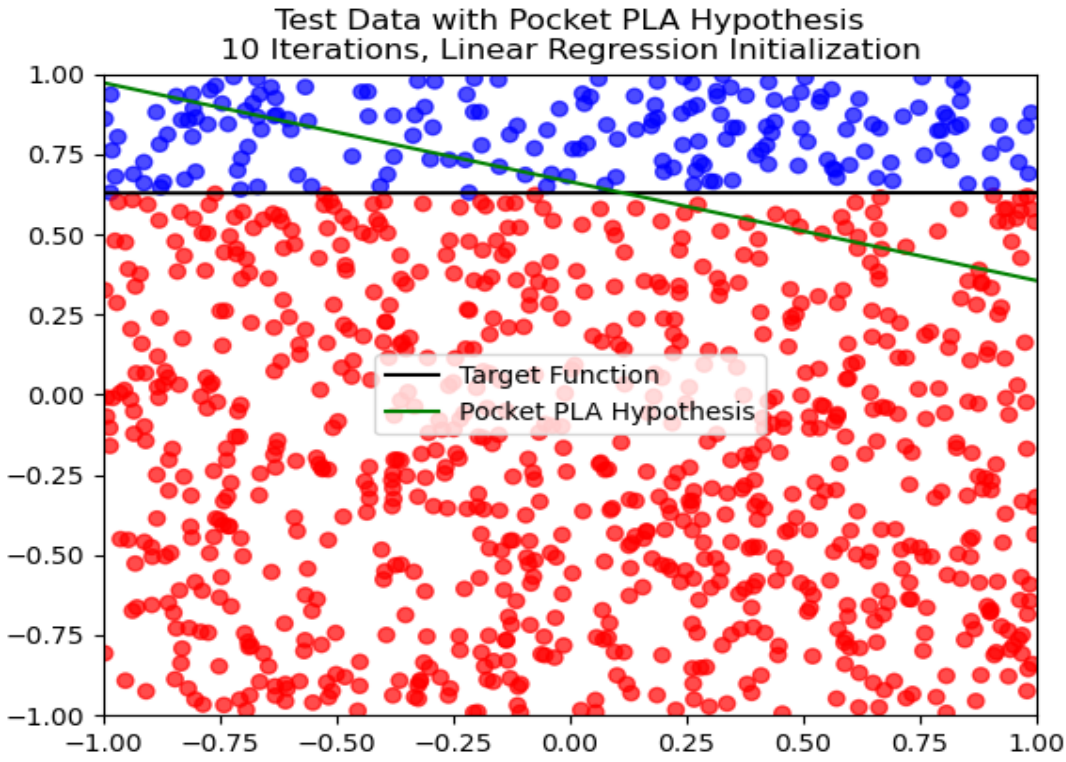

Fig 5. i = 10, w = Inicializado pela Regressão Linear

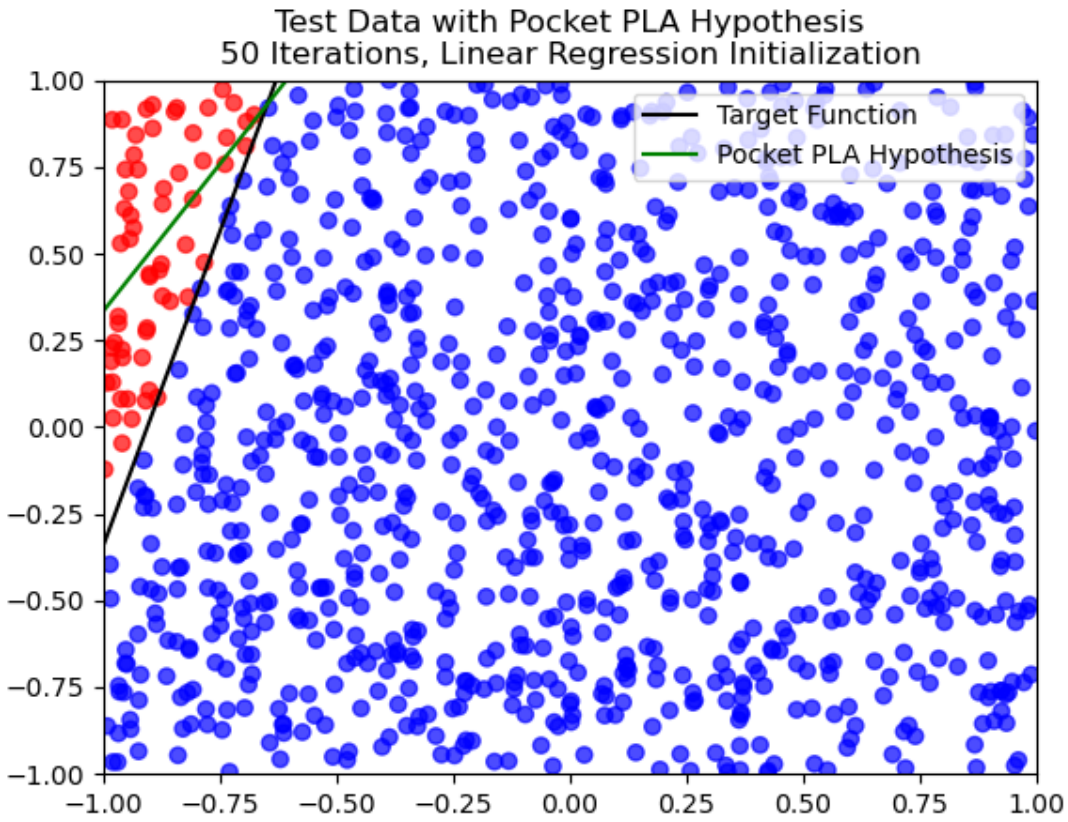

Fig 6. i = 50, w = Inicializado pela Regressão Linear

Regressão Não Linear

1 – Resultado obtido: 0.509463. Resposta: d. Este resultado alto de Ein mostra que não faz muito sentido se utilizar de uma abordagem de Regressão Linear simples para capturar a complexidade da função target dada, que é uma função não-linear. A função target depende dos termos quadráticos x_1^2 e x_2^2, mas ao utilizar apenas os termos lineares x1 e x2 como atributos, a Regressão Linear não consegue modelar corretamente, resultando em um alto erro de classificação dentro da amostra. Em termos de aprendizado de máquina, isso ressalta a importância da escolha adequada das características e da transformação dos dados para capturar corretamente a complexidade do problema, e a necessidade de outros tipos de modelos, para lidar com funções target não-lineares de forma eficaz.

2 – Resultado obtido: Pesos médios após 1000 execuções: [-9.89660266e-01 2.03830427e-03 -3.87205063e-04 -7.31579962e-04 1.55219616e+00 1.55751144e+00], a distância desse vetor mais próxima seria à opção (a) com uma distância de 0.17999930126050467. Resposta: a. Este resultado destaca a importância da transformação não-linear dos dados em problemas de classificação onde a função target também é não-linear. Ao incluir termos quadráticos, a Regressão Linear pôde modelar a complexidade da função target de forma mais eficaz, reduzindo significativamente o erro de classificação se comparado com o item anterior, no qual foi utilizada uma Regressão Linear simples. A transformação dos dados permitiu capturar a estrutura da função target, aproximando-se muito mais fielmente da hipótese verdadeira. Em termos de machine learning, fica demonstrada a eficácia de técnicas de feature engineering e a necessidade de modelos apropriados para tratar a não-linearidade presente nos dados.

3 – Resultado obtido: 0.125995. Resposta: b. Este desempenho, embora não perfeito, reflete a eficácia da abordagem de transformar os dados com características não-lineares, especialmente em um cenário com ruído adicionado. A inclusão de termos quadráticos permitiu que o modelo capturasse de melhor maneira a complexidade da função target $f(x_1, x_2) = sign(x_1^2 + x_2^2 - 0.6)$, resultando em uma hipótese que não apenas minimiza o erro dentro da amostra (Ein), mas também mantém um erro razoavelmente baixo fora da amostra (Eout). Em termos de machine learning, isso demonstra a importância da feature engineering adequada para lidar com a não-linearidade nos dados, permitindo que modelos simples, como a Regressão Linear, alcancem um desempenho relativamente robusto em problemas complexos. O valor de Eout de aproximadamente 0.126 reforça a ideia de que a transformação correta dos dados e a consideração do ruído são essenciais para melhorar a capacidade de generalização do modelo.

Link para o repositório no github com o código fonte:

https://github.com/SylvioMello/Computational-Inteligence-Class