Prof. Carlos Eduardo Pedreira
PESC/COPPE/UFRJ

Trabalho prático - Inteligência Computacional I

Data de entrega: 31 de maio de 2024

Neste *Trabalho Prático*, você irá implementar alguns dos exercícios computacionais descritos no Homework #1 e no Homework #2 do professor Yaser Abu-Mostafa, disponibilizados em:

- https://work.caltech.edu/homework/hw1.pdf
- https://work.caltech.edu/homework/hw2.pdf

Um relatório prático deve ser escrito e entregue. Para cada questão, você deverá apresentar não só a resposta obtida, mas também um parágrafo contendo suas interpretações a respeito dos resultados (qual a relação entre esse resultado e os conceitos de aprendizado de máquina vistos em sala? quais as conexões entre o resultado desta questão e os das questões anteriores?). Respostas sem nenhuma discussão serão penalizadas, assim como as respostas desnecessariamente prolixas. Seu relatório também deverá conter o código fonte da implementação devidamente comentado, usando a linguagem de programação da sua escolha (preferencialmente Python).

Uma tradução livre dos enunciados presentes em ambos os Homeworks foi gerada com algumas alterações. Caso ocorram dúvidas na interpretação, recorram aos textos originais em inglês.

Perceptron

Neste problema, você criará a sua própria função *target* f e uma base de dados $\mathcal{D}$ para que possa ver como o Algoritmo de Aprendizagem Perceptron funciona. Escolha $d = 2$ pra que você possa visualizar o problema, e assuma $\mathcal{X} = [-1, 1] \times [-1, 1]$ com probabilidade uniforme de escolher cada $\mathbf{x} \in \mathcal{X}$.

Em cada execução, escolha uma reta aleatória no plano como sua função *target* f (faça isso selecionando dois pontos aleatórios, uniformemente distribuídos em $[-1, 1] \times [-1, 1]$, e pegando a reta que passa entre eles), de modo que um lado da reta mapeia pra $+1$ e o outro pra -1. Escolha os inputs $\mathbf{x}_n$ da base de dados como um conjunto de pontos aleatórios (uniformemente em $\mathcal{X}$), e avalie a função *target* em cada $\mathbf{x}_n$ para pegar o output correspondente y_n.

Agora, pra cada execução, use o Algoritmo de Aprendizagem Perceptron (PLA) para encontrar g. Inicie o PLA com um vetor de pesos $\mathbf{w}$ zerado (considere que $\text{sign}(0) = 0$, de modo que todos os pontos estejam classificados erroneamente ao início), e a cada iteração faça com que o algoritmo escolha um ponto aleatório dentre os classificados erroneamente. Estamos interessados em duas quantidades: o número de iterações que o PLA demora para convergir pra g, e a divergência entre f e g que é $\mathbb{P}[f(\mathbf{x}) \neq g(\mathbf{x})]$ (a probabilidade de que f e g vão divergir na classificação de um ponto aleatório). Você pode calcular essa probabilidade de maneira exata, ou então aproximá-la ao gerar uma quantidade suficientemente grande de novos pontos para estimá-la (por exemplo, 10.000).

A fim de obter uma estimativa confiável para essas duas quantias, você deverá realizar 1000 execuções do experimento (cada execução do jeito descrito acima), tomando a média destas execuções como seu resultado final.

Para ilustrar os resultados obtidos nos seus experimentos, acrescente ao seu relatório gráficos scatterplot com os pontos utilizados para calcular E_{out}, assim como as retas correspondentes à função *target* e à hipótese g encontrada.

1. Considere $N = 10$. Quantas iterações demora, em média, para que o PLA convirja com $N = 10$ pontos de treinamento? Escolha o valor mais próximo do seu resultado.

(a) 1
(b) 15
(c) 300
(d) 5000
(e) 10000

2. Qual das alternativas seguintes é mais próxima de $\mathbb{P}[f(\mathbf{x}) \neq g(\mathbf{x})]$ para $N = 10$?

   (a) 0.001
   (b) 0.01
   (c) 0.1
   (d) 0.5
   (e) 1

3. Agora considere $N = 100$. Quantas iterações demora, em média, para que o PLA convirja com $N = 100$ pontos de treinamento? Escolha o valor mais próximo do seu resultado.

   (a) 50
   (b) 100
   (c) 500
   (d) 1000
   (e) 5000

4. Qual das alternativas seguintes é mais próxima de $\mathbb{P}[f(\mathbf{x}) \neq g(\mathbf{x})]$ para $N = 100$?

   (a) 0.001
   (b) 0.01
   (c) 0.1
   (d) 0.5
   (e) 1

5. É possível estabelecer alguma regra para a relação entre N, o número de iterações até a convergência, e $\mathbb{P}[f(\mathbf{x}) \neq g(\mathbf{x})]$?

Regressão Linear

Nestes problemas, nós vamos explorar como Regressão Linear pode ser usada em tarefas de classificação. Você usará o mesmo esquema de produção de pontos visto na parte acima do Perceptron, com $d = 2$, $\mathcal{X} = [-1, 1] \times [-1, 1]$, e assim por diante.

1. Considere $N = 100$. Use Regressão Linear para encontrar g e calcule E_{in}, a fração de pontos dentro-de-amostra que foram classificados incorretamente (armazene os g's pois eles serão usados no item seguinte). Repita o experimento 1000 vezes. Qual dos valores abaixo é mais próximo do E_{in} médio?

   (a) 0
   (b) 0.001
   (c) 0.01

(d) 0.1
(e) 0.5

2. Agora, gere 1000 pontos novos e use eles para estimar o E_{out} dos g's que você encontrou no item anterior. Novamente, realize 1000 execuções. QUal dos valores abaixo é mais próximo do E_{out} médio?

   (a) 0
   (b) 0.001
   (c) 0.01
   (d) 0.1
   (e) 0.5

3. Agora, considere $N = 10$. Depois de encontrar os pesos usando Regressão Linear, use-os como um vetor de pesos iniciais para o Algoritmo de Aprendizagem Perceptron (PLA). Execute o PLA até que ele convirja num vetor final de pesos que separa perfeitamente os pontos dentro-de-amostra. Dentre as opções abaixo, qual é mais próxima do número médio de iterações (sobre 1000 execuções) que o PLA demora para convergir?

   (a) 1
   (b) 15
   (c) 300
   (d) 5000
   (e) 10000

4. Vamos agora avaliar o desempenho da versão *pocket* do PLA em um conjunto de dados que não é linearmente separável. Para criar este conjunto, gere uma base de treinamento com N_2 pontos como foi feito até agora, mas selecione aleatoriamente 10% dos pontos e inverta seus rótulos. Em seguida, implemente a versão *pocket* do PLA, treine-a neste conjunto não-linearmente separável, e avalie seu E_{out} numa nova base de N_2 pontos na qual você não aplicará nenhuma inversão de rótulos. Repita para 1000 execuções, e mostre o E_{in} e E_{out} médios para as seguintes configurações (não esqueça dos gráficos scatterplot, como anteriormente):

   (a) Inicializando os pesos com 0; $i = 10$; $N_1 = 100$; $N_2 = 1000$.
   (b) Inicializando os pesos com 0; $i = 50$; $N_1 = 100$; $N_2 = 1000$.
   (c) Inicializando os pesos com Regressão Linear; $i = 10$; $N_1 = 100$; $N_2 = 1000$.
   (d) Inicializando os pesos com Regressão Linear; $i = 50$; $N_1 = 100$; $N_2 = 1000$.

Regressão Não-Linear

Nestes problemas, nós vamos novamente aplicar Regressão Linear para classificação. Considere a função *target*:

$$f(x_1, x_2) = \text{sign}(x_1^2 + x_2^2 - 0.6)$$

Gere um conjunto de treinamento de $N = 1000$ pontos em $\mathcal{X} = [-1, 1] \times [-1, 1]$ com probabilidade uniforme escolhendo cada $\mathbf{x} \in \mathcal{X}$. Gere um ruído simulado selecionando aleatoriamente 10% do conjunto de treinamento e invertendo o rótulo dos pontos selecionados.

1. Execute a Regressão Linear sem nenhuma transformação, usando o vetor de atributos $(1, x_2, x_2)$ para encontrar o peso $\mathbf{w}$. Qual é o valor aproximado de classificação do erro médio dentro-de-amostra E_{in} (medido ao longo de 1000 execuções)?

   (a) 0

   (b) 0.1

   (c) 0.3

   (d) 0.5

   (e) 0.8

2. Agora, transforme os $N = 1000$ dados de treinamento seguindo o vetor de atributos não-linear $(1, x_2, x_2, x_1x_2, x_1^2, x_2^2)$. Encontre o vetor $\widetilde{\mathbf{w}}$ que corresponda à solução da Regressão Linear. Quais das hipóteses a seguir é a mais próxima à que você encontrou? Avalie o resultado médio obtido após 1000 execuções.

   (a) $g(x_1, x_2) = \text{sign}(-1 - 0.05x_1 + 0.08x_2 + 0.13x_1x_2 + 1.5x_1^2 + 1.5x_2^2)$

   (b) $g(x_1, x_2) = \text{sign}(-1 - 0.05x_1 + 0.08x_2 + 0.13x_1x_2 + 1.5x_1^2 + 15x_2^2)$

   (c) $g(x_1, x_2) = \text{sign}(-1 - 0.05x_1 + 0.08x_2 + 0.13x_1x_2 + 15x_1^2 + 1.5x_2^2)$

   (d) $g(x_1, x_2) = \text{sign}(-1 - 1.5x_1 + 0.08x_2 + 0.13x_1x_2 + 0.05x_1^2 + 0.05x_2^2)$

   (e) $g(x_1, x_2) = \text{sign}(-1 - 0.05x_1 + 0.08x_2 + 1.5x_1x_2 + 0.15x_1^2 + 0.15x_2^2)$

3. Qual o valor mais próximo do erro de classificação fora-de-amostra E_{out} de sua hipótese na questão anterior? (Estime-o gerando um novo conjunto de 1000 pontos e usando 1000 execuções diferentes, como antes).

   (a) 0

   (b) 0.1

   (c) 0.3

   (d) 0.5

   (e) 0.8