Nome: Dennis José da Silva, NUSP: 9176517, dennis@ime.usp.br

Universidade de São Paulo

Instituto de Matemática e Estatística

MAC5832 - Aprendizagem Computacional: Modelos, Algoritmos e Aplicações

**Problem 1:**

**Resposta.:**

Exemplos positivos: 101, 1100, 1010101, 1111, 0011

Exemplos Negativos: 010, 1000, 101010, 111, 1

**Problem 2:**

**Resposta.:**

Exemplos positivos: 101, 1001, 1111, 0000, 110011, 1, 0

Exemplos negativos: 10, 1100, 1010, 1110, 011101, 01

**Problem 3:**

**Resposta:**

Exemplos positivos: (0.1, 0.1, 0.1), (0.1, 0.5), (0.5, 0.3, 0.6, 0.1)

Exemplos negativos: (1, 1, 1), (2, 1, 2), (3.5, 4.5, 1.1)

**Problem 4:**

**Resposta:**

Função alvo =

(1001111, 1) =

(0110110, 0) = Conjunto não modificado por que

(1011101, 1) =

(1011001, 1) =

**Problem 5:**

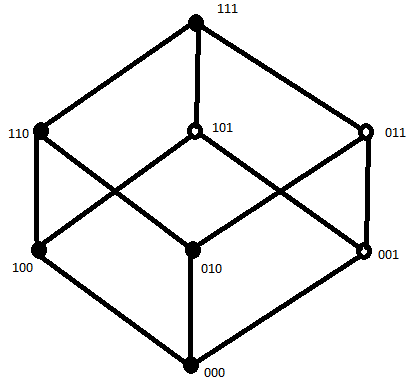
**Resposta:**

Três possíveis formas são por reticulado booleano, função booleana e portas lógicas, por exemplo, dada a seguinte tabela para a função 0, 1}:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

Pode ser representado como:

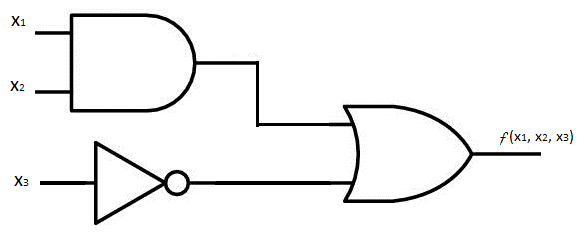
Por reticulado:



Repare que os pontos pintados em preto representam os pontos onde a função vale 1.

Por Função Booleana:

Por portas lógicas:



**Problem 6:**

**Resposta:**

**ATENÇÃO: Os programas foram desenvolvidos em python versão = 3.4.1 no S.O Windows 7**

Perceptron desenvolvido em linguagem Python. O arquivo fonte está na pasta problem 6 juntamente com as instruções de uso, resumo abaixo:

Executar programa de geração de amostras: “samples\_generator.py” que recebe os parâmetros:

-s = sample-number, quantidade de amostras a serem geradas. Valor padrão = 100

-t = threshold, valor de θ; valor padrão = 5.0

-i = ini, valor inicial do domínio da função. Valor padrão = 0.0

-e = end, valor final do domínio da função. Valor padrão = 10.0

-o = output file, arquivo de amostras no formato json, Valor padrão ./samples.json

A executar o gerador de amostras serão gerados amostras em json com o seguinte formato:

[[valor gerado, {1 ou 0}], [valor gerado, {1 ou 0}], [valor gerado, {1 ou 0}]]

Então executar o programa “main .py” que gera o classificador apartir do algoritmo de perceptron, esse programa recebe o seguinte parâmetro:

-t = training\_data\_file, arquivo json geradpo pelo programa samples\_generator. Valor padrão = “./samples.json”

Então entre com os valores de x, e o programa retorna o y de acordo com o conceito aprendido.

O algoritmo do perceptron está no arquivo “Perceptron.py”

**Problem 7:**

**Resposta:**

**ATENÇÃO: Os programas foram desenvolvidos em python versão = 3.4.1 no S.O Windows 7**

Os perceptrons foram desenvolvido em linguagem Python. O arquivo fonte está na pasta problem 7 juntamente com as instruções de uso, resumo abaixo:

Executar programa de geração de amostras: “samples\_generator.py” que recebe os parâmetros:

-b = bound, limitante superior ou inferior, pode receber os valores “upper” ou “lower”, sem valor padrão;

-s = sample-number, quantidade de amostras a serem geradas. Valor padrão = 100

-t = threshold, valor de θ; valor padrão = 5.0

-i = ini, valor inicial do domínio da função. Valor padrão = 0.0

-e = end, valor final do domínio da função. Valor padrão = 10.0

-o = output file, arquivo de amostras no formato json, Valor padrão ./samples.json

A executar o gerador de amostras serão gerados amostras em json com o seguinte formato:

[[valor gerado, {1 ou 0}], [valor gerado, {1 ou 0}], [valor gerado, {1 ou 0}]]

Para a solução do exercício, será necessário executar duas vezes o “sample\_generator.py”, um para o limitante inferior “lower” e outro para o superior “upper”, grave os arquivos json gerados com nomes diferentes e execute o “main.py”

Então executar o programa “main .py” que gera o classificador apartir do algoritmo de perceptron, esse programa recebe os seguinte parâmetro:

-l = lower\_tranning\_data\_file, arquivo json gerado pelo programa samples\_generator para limitante inferior. Valor padrão = “./lower\_samples.json”

-u = upper\_bound\_tranning\_data\_file, arquivo json gerado pelo programa samples\_generator para limitante superior. Valor padrão = “./upper\_samples.json”

O algoritmo do perceptron está no arquivo “Perceptron.py”, nesse o algoritmo gerara dois perceptron e juntara os dois para classificar a entrada pelo console.

O algoritmo do perceptron está no arquivo “Perceptron.py”

Então entre com os valores de x, e o programa retorna o y de acordo com o conceito aprendido.

**Problem 8:**

**Resposta:**

Sabendo que um monômio booleano representa um número qualquer com no máximo k literais podemos dizer que que Mk,n  = 2n , porque teremos uma função que mapeia um monômio qualquer em 1 e outra em 0, como temos “n“ monômios podemos ter 2n combinações de 1 e 0 como resultado de cada monômio, logo podemos ter 2n funções. Para Dmk cada função de Mmk se torna uma variável das disjunções de Dmk como sabemos que para qualquer disjunção representa a operação OU da álgebra booleana essas funções de Dmk só podem ser 0 quando todas as suas variáveis são 0 (quando todas as funções de Mmk assumem valor 0 ao mesmo tempo) e 1 quando possui alguma variável 1 (quando pelo menos uma função de Mmk assume valor 1), sabemos que podemos ter Dmké igual a números de combinações dos resultados das funções de Mmk e como sabemos que Mmk pode representar 2n funções, logo Dmk = 2n**. Logo** Podemos calcular |H| como:

**|H| = 2n . 2n = 22n**

**Problem 9:**

**Resposta:**

Rodar programa main.py na pasta problem9, ele gera duas imagens a problem9-1.png para o exercício 9 (1) e problem9-2.png para o exercício 9 (**2)**