**Relatório Projeto 2**

Inteligência Artificial (2018/19)

Grupo A023 – Margarida Morais, 86473 – Mafalda Mendes, 83502

**Parte 1 – Redes Bayesianas**

1. **Descrição dos Resultados Obtidos**

Todos os resultados obtidos através da nossa implementação da rede, estão de acordo com os resultados já conhecidos, que estão no ficheiro *mainBN.py*.

1. **Métodos Implementados**

De forma a poder ser construída uma Rede Bayesiana na forma de um grafo acíclico, foram implementadas duas classes (**Node** e **BN**).

Dentro destas classes estão definidos métodos correspondentes à mesma que irão permitir utilizar *Métodos de Inferência Exata* dentro da rede.

Tais como:

* ***computeProb***

O método *computeProb* permite calcular a probabilidade de um nó recebendo um tuplo de evidências que irá servir para sabermos qual o valor do acontecimento do qual o próprio nó depende, ou seja, os pais desse nó no grafo.

O valor retornado pela função é um array que contém a probabilidade do nó ser falso na primeira posição, ou seja de esse acontecimento ser falso, e a probabilidade de ser verdadeiro na segunda posição, que corresponde ao acontecimento ser verdadeiro.

Se o nó, no qual está a ser calculada a probabilidade, não tiver pais, então a probabilidade é meramente o valor dentro do array dado no atributo *prob* quando o nó é criado.

Por outro lado, quando o nó tem um ou mais pais, é necessário primeiro guardar os valores das evidências de todos os pais desse nó, e só depois iterar o respetivo array *prob* de modo a encontrar a probabilidade do nó sabendo as evidências dos nós antecessores. No nosso caso isto é feito, acedendo ao  *prob*, pode-se dizer recursivamente, nos indexes correspondentes às evidências dos pais, visto que todos os arrays têm tamanho 2.

A complexidade computacional do *computeProb* é O(N), sendo N o número de nós existentes.

* ***computePostProb***

O método *computePostProb* permite inferir a probabilidade de um nó da rede conhecendo um conjunto de evidências, em que pode haver algumas que são desconhecidas.

O valor retornado pela função é a probabilidade do acontecimento deste nó ser verdadeiro.

Para poder ser feita a inferência da probabilidade desconhecida, têm de ser primeiro calculadas todas as combinações possíveis de evidências para aquelas cujo valor é desconhecido.

Para cada combinação possível de evidências temos ainda de ter dois valores, um que corresponde à probabilidade conjunta de todos os nós no grafo com a evidência do próprio nó a 1 e outro em que esta é 0. Depois de terem sido então calculadas probabilidades conjuntas para todos as combinações de evidências, devem ser somadas as respetivas (\*).

A probabilidade à posteriori é calculada recorrendo a uma variável denominada **“constante de normalização”**, à qual chamamos *alpha* e que corresponde ao inverso da soma dos dois valores calculados anteriormente (\*).

O valor da probabilidade à posteriori desse nó é então o valor da soma das probabilidades conjuntas em que a evidência do próprio nó é 1, multiplicado pela constante *alpha*.

Ver quais são os outros métodos para calcular a probabilidade à posteriori e ver se há outros mais eficientes.

A complexidade computacional do *computePostProb* é O(?).

* ***computeJointProb***

O método *computeJointProb* calcula a probabibilidade conjunta de uma rede conhecendo um tuplo de evidências que está completo (todas as evidências são conhecidas)