13/06/2019

Segundo Trabalho de Programação I

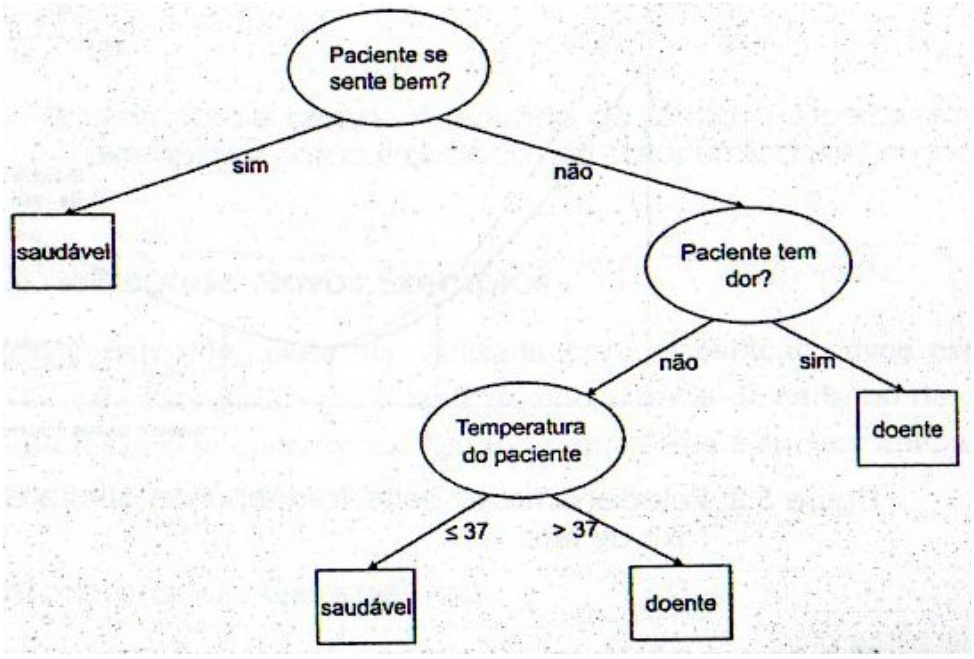
Prof. Flávio Miguel Varejão

I. Descrição do Problema

Classificação de dados multidimensionais é um dos problemas mais comuns na área de aprendizado de máquina. Esse problema consiste em classificar um conjunto de dados multidimensional representando um objeto (ou indivíduo) em uma classe (e somente uma) dentre um possível conjunto pré-determinado de possíveis classes. Por exemplo, diagnosticar uma pessoa apresentando dor e febre como saudável ou doente é uma tarefa de classificação.

Existem várias técnicas que permitem criar automaticamente um classificador para uma dada tarefa a partir de um conjunto de exemplos, isto é, um conjunto de casos com classificação conhecida. Uma dessas técnicas são os indutores de árvores de decisão. Uma árvore de decisão é uma árvore onde cada nó não terminal representa uma decisão a ser tomada e cada nó terminal (folha da árvore) é uma possível classificação. O processo de classificação de um caso por uma árvore de decisão consiste em percorrer a árvore da raiz para uma das folhas de acordo com as características do caso.

A figura abaixo ilustra um exemplo de árvore de decisão capaz de diagnosticar se um paciente está doente ou saudável. Por exemplo, se o paciente não se sente bem, não tem dor mas tem temperatura acima de 37 graus Celsius, ele é classificado como doente.



Uma árvore de decisão pode ser representada em forma de regras. A representação em regras da árvore acima é dado por:

```
se paciente está bem então retorne saudável
senao
      se paciente não está bem então
            se paciente tem dor então retorne doente
            senao
                  se temperatura <= 37 então retorne saudável
                  senão retorne doente
            fim-se
      fim-se
fim-se
```

Para um algoritmo aprender como construir uma árvore de decisão para resolver um problema de classificação é necessário utilizar uma base de dados com exemplos com classificação conhecida. A figura seguinte mostra um exemplo de base de dados *Ex* contendo 5 exemplos (T1, T2, T3, T4 e T5). Os exemplos representam dias nos quais alguém viajou ou não. Cada exemplo é descrito pelas características aparência, temperatura, umidade e ventando, que determinam as condições ambientais do dia específico, e pela decisão tomada (viajar ou não viajar). No exemplo T4, o dia foi com sol, 23 graus Celsius de temperatura média, 95% de umidade relativa do ar e sem vento. Neste dia, não houve viagem.

Exemplo Nº	Aparência	Temperatura	Umidade	Ventando	Viajar?
T1	sol	25	72	sim	vá
T2	sol	28	91	sim	não_vá
T3	sol	22	70	não	vá
T4	sol	23	95	não	não_vá
T5	sol	30	85	não	não_vá

Um indutor de árvores de decisão utiliza os exemplos da base para construir a árvore de decisão. Um algoritmo indutor de árvores de decisão é composto dos seguintes passos:

Passo 1: Se a Base **Ex** contém um ou mais exemplos, todos pertencentes à mesma classe *Cj*. Nesse caso, a árvore de decisão para a base *Ex* é um nó folha identificando a classe *Cj*.

Passo 2: Se a Base *Ex* contém exemplos que pertencem a várias classes, refina-se *Ex* em subconjuntos de exemplos que são (ou aparentam ser) conjuntos de exemplos pertencentes a uma única classe. Nesse caso, deve ser escolhida uma característica para servir como teste do nó não terminal a ser adicionado a árvore. Os possíveis valores da característica escolhida são denotados por {*O1, O2, ...,Or*}. A Base *T* é então particionada em subconjuntos *Ex1, Ex2, ...,Exr*, nos quais cada *Exi* contém todos os exemplos em *Ex* cujo valor da característica seja *Oi.* A árvore de decisão para *Ex* consiste em um nó interno identificado pela característica escolhida e uma aresta para cada um dos resultados possíveis.

Passo 3: Se a Base *Ex* não contém exemplos ou não existem mais características para serem utilizadas como teste, a árvore de decisão para a Base *Ex* é novamente um nó folha e a classe mais comum deve ser utilizada.

Passo 4: Os passos 1, 2 e 3 são aplicados recursivamente para cada subconjunto de exemplos de maneira que, em cada nó, as arestas levam para as subárvores construídas a partir do subconjunto de exemplos *Exi*.

O pseudo-código seguinte detalha esse algoritmo:

```
arvoreDecisao (exemplos, caracteristicas, maisComum): arvore
    se (exemplos é vazio) entao retorne maisComum;
    senão se (todos os exemplos têm a mesma classificação)
     entao retorne (a classificação);
    senão se (não há mais características)
      então retorne maioria(exemplos);
    senão
        melhor <- melhorTeste(características,exemplos);
        árvore <- nova árvore com raiz "melhor";
        para cada valor vi de melhor faça
            exemplosi <- exemplos onde melhor = vi;
            subárvore <- arvoreDecisao(exemplosi,
                caracteristicas-{melhor}, maioria(exemplos));
            adicione subárvore como um ramo à árvore com
                rótulo vi;
    retorne arvore;
```

A função `maioria` recebe uma base de dados *Ex* e retorna a classificação majoritária na base, isto é, a classe *Ci* mais presente no conjunto de exemplos que pertencem a base.

A função `melhorTeste` recebe o conjunto de características a ser considerado e uma base de exemplos e retorna a característica que deve ser usada como teste no nó não terminal de decisão a ser incluído na árvore. A característica escolhida por essa função é a que gera a melhor razão de ganho de informação *IGR*. O cálculo da razão de ganho de uma característica é a realizado em 3 passos:

Passo 1: Calcular a entropia da base de exemplos *Ex*. Para calculá-la é necessário determinar as percentagens de ocorrência p_i de cada classe presente nos exemplos da base.

$$p_i = n_i / N$$

onde n_i é o número de exemplos com classificação *i* na base *Ex* e *N* é o número total de exemplos na base *Ex.*

A entropia *H(Ex)* é calculada da seguinte forma:

$$H(Ex) = - \Sigma\, pi \;\; log2 \;\; (pi\,)$$

Passo 2: Calcular a razão de ganho de informação (*IGR*) de cada característica *a* a ser avaliada na base de exemplos *Ex*, dividindo-se o ganho de informação *IG(Ex, a)* pelo valor intrínseco *IV(Ex, a)* da característica *a*.

$$IGR(Ex, a) = IG(Ex, a)/IV(Ex, a)$$

O ganho de informação *IG(Ex, a)* é calculado da seguinte forma:

$$IG(Ex,a) = H(Ex) - \sum_{v \in values(a)} \left(\frac{|\{x \in Ex | value(x,a) = v\}|}{|Ex|} \cdot H(\{x \in Ex | value(x,a) = v\}) \right)$$

O valor intrínseco *IV(Ex, a)* é calculado da seguinte forma:

$$IV(Ex,a) = -\sum_{v \in values(a)} \frac{|\{x \in Ex | value(x,a) = v\}|}{|Ex|} \cdot \log_2 \left(\frac{|\{x \in Ex | value(x,a) = v\}|}{|Ex|} \right)$$

Passo 3: Escolher a característica com maior valor de razão de ganho de informação IGR*(Ex, a)*.

Para o caso de características numéricas, é necessário realizar um processo de discretização para poder avaliá-la e eventualmente incluí-la na árvore de decisão. O processo de discretização a ser efetuado consiste em determinar intervalos de valores para os quais há correlação entre os valores numéricos da característica e uma determinada classificação. Portanto, o processo de discretização pode ser dividido em 3 passos:

Passo 1: Selecionar da base *Ex* as colunas correspondentes a característica a ser discretizada e a que contem a classificação dos exemplos.

Passo 2: Ordenar os valores da característica mantendo a correspondência com a coluna de classificação.

Passo 3: Determinar como valores da característica os intervalos para os quais a classificação se mantém a mesma. Para fazer isso é necessário calcular a mediana dos valores de umidade subsequentes quando ocorre a mudança de classificação.

Considere, por exemplo, uma base de exemplos para a problema de decidir se devemos viajar ou não. Para discretizar a característica *umidade*, o primeiro passo deve inicialmente extrair da base o conteúdo da coluna *umidade* e a respectiva coluna de classificação *viajar*:

umidade	72	90	80	40	60	48
viajar	vá	não_vá	vá	não_vá	vá	não_vá

O segundo passo consiste em ordenar os dados de *umidade* mantendo a correspondência com a coluna *viajar*:

umidade	40	48	60	72	80	90
viajar	não_vá	não_vá	vá	vá	vá	não_vá

O terceiro passo calcula as medianas de 48 e 60 ((48+60)/2 = 54) e de 80 e 90 ((80+90)/2 = 85). Portanto, no exemplo dado, os valores da característica *umidade* são os intervalos: *umidade <= 54*, *54 < umidade <= 85* e *umidade > 85*.

II. Especificação do Sistema

Funcionalidades a serem implementadas:

*1.*Leitura da descrição da base de exemplos de um arquivo texto denominado "descricao.txt". Cada linha deste arquivo descreve uma característica da base na

sequência na qual ela será lida. Linhas com características numéricas contém apenas o nome da característica. Linhas com características nominais contém o nome da característica e os valores possíveis da característica. A classe é representada como característica nominal e sempre é a última linha do arquivo.
*2.*Leitura da base de exemplos de um arquivo texto denominado "base.txt". Cada linha corresponde a um exemplo. Os valores das características de um exemplo são colocados sucessivamente em uma linha separadas por espaço. O último valor representa a classe do exemplo.
3. Executar o algoritmo de indução de árvore de decisão e gravá-la no formato de regras no arquivo “arvore.txt”.
4. Leitura de um caso a ser classificado de um arquivo texto denominado "caso.txt". Esse arquivo possui apenas uma linha contendo os valores do caso (uma linha com a mesma formatação das linhas do arquivo base com exceção da classe que é omitida).
*5.*Realizar a classificação do caso lido e gravar a classe selecionada na única linha do arquivo “classe.txt”.

A seguir se apresenta um exemplo dos arquivos para a base do problema de decidir viajar:

Exemplo de arquivo descricao.txt:

```
Aparencia Sol Chuva Nublado
Temperatura
Umidade
Vento Sim Nao
Viajar Va NaoVa
```

Exemplo de arquivo base.txt:

```
Sol 25 72 Sim Va
Sol 28 91 Sim NaoVa
Sol 22 70 Nao Va
Sol 23 95 Nao NaoVa
Sol 30 85 Nao NaoVa
```

Exemplo de arquivo arvore.txt:

```
se umidade <= 78.5 então retorne Va
senao retorne NaoVa
fim-se
```

Exemplo de arquivo caso.txt:

```
Chuva 23 92 Sim
```

Exemplo de arquivo result.txt:

```
NaoVa
```

III. Requisitos da implementação

- Modularize seu código adequadamente.

- Crie códigos claros e organizados. Utilize um estilo de programação consistente, Comente seu código.
- Os arquivos do programa devem ser lidos e gerados na mesma pasta onde se encontram os arquivos fonte do seu programa.

IV. Condições de Entrega

O trabalho deve ser feito individualmente e submetido por e-mail até as 23:59 horas da data limite especificada para o endereço fvarejao@gmail.com com o subject PG_1_TRABALHO_2_NomedoAluno_SobrenomedoAluno. O e-mail deve conter também um arquivo .zip com o mesmo nome do subject do e-mail enviado. O arquivo principal (o que contém o main do trabalho) obrigatoriamente deve estar com o nome "main". Note que a data limite já leva em conta um dia adicional de tolerância para o caso de problemas de submissão via rede. Isso significa que o aluno deve submeter seu trabalho até no máximo um dia antes da data limite. Se o aluno resolver submeter o trabalho na data limite, estará fazendo isso assumindo o risco do trabalho ser cadastrado no sistema após o prazo. Em caso de recebimento do trabalho após a data limite, o trabalho não será avaliado e a nota será ZERO. Aluno que receber zero por este motivo e vier pedir para o professor considerar o trabalho estará cometendo um ato de DESRESPEITO ao professor e estará sujeito a perda adicional de pontos na média.

V. Data de Entrega: 08/07/2019

VI. Avaliação

Os trabalhos terão nota zero se:

A data de entrega for fora do prazo estabelecido;

O trabalho não compilar;

O trabalho não gerar o arquivo com o resultado e formato esperado;

For detectada a ocorrência de plágio.

Observação importante

Caso haja algum erro neste documento, serão publicadas novas versões e divulgadas erratas em sala de aula. É responsabilidade do aluno manter-se informado, freqüentando as aulas ou acompanhando as novidades na página da disciplina na Internet.