Métricas em Machine Learning

T2 - Medidas de semelhanças para dados nominativos/categóricos

Relatório do microprojeto

Grupo H: Artur Ribeiro (a82516), Filipa Silva (a81015), Gonçalo Costeira (a79799), Hugo Nogueira (a81898), Luís Abreu (a82888), Miguel Ribeiro (pg44419)

6 de junho de 2021

Resumo

Este documento foi elaborado no âmbito da Unidade Curricular de Métricas em Machine Learning e tem como objetivo estudar as semelhanças entre dados categóricos.

Para isso é realizado um enquadramento de como as medidas de similaridade operam em dados deste tipo, sendo dado um pequeno exemplo com uma base de dados artificial onde são calculadas as similaridades entre atributos e as frequências de todos os atributos que pertencem ao espaço de atributos definido.

Após este processo, implementam-se estas métricas num *script* com um volume de dados substancialmente elevado para que se possam realizar abordagens críticas acerca do assunto em questão.

Palavras-chave: Dados Categóricos, Medidas de Similaridade, Semelhança, Distância, *Machine Learning*.

Conteúdo

Capítulo 1

Introdução

1.1 Motivação

Clustering de dados é uma técnica que permite identificar grupos de objetos com elementos similares, esta medição de distância ou semelhança é uma etapa fundamental no que concerne à mineração de dados e à descoberta de conhecimento.

Qualquer base de dados real, é composta por uma mistura de dados categóricos, numéricos, ordinais, entre outros. Se falarmos em dados do tipo numérico ou ordinal, a sua noção de similaridade é bem compreendida, uma vez que estes apresentam uma ordenação natural para os seus valores. O mesmo não acontece para o tipo de dados categórico, estes dados são rudimentares, ou seja, não têm nenhuma estrutura particular definida. Portanto, calcular a semelhança entre dois objetos deste tipo não é simples e torna o *clustering* uma tarefa de *matching* (correspondência) e não de medição de distância.

Posto isto, o *clustering* de dados categóricos enfrenta dois problemas: processamento de variáveis não categóricas, um procedimento necessário para aplicar as medidas de similaridade para o processo de *matching*; escolha da medida de similaridade mais apropriada para um dado domínio.

1.2 Objetivo

Este microprojeto tem como objetivo realizar um trabalho de estudo acerca deste tipo de métrica. Assim sendo, os autores têm por base dois artigos científicos fornecidos pela equipa docente, bem como uma base de dados relacionada com cogumelos venenosos.

Relativamente aos artigos, espera-se uma leitura e compreensão daquilo que são os dados categóricos, como também das medidas de similaridade, em que é fulcral avaliar as métricas usadas e o seu desempenho dependendo do domínio em questão. Por fim, e após esta vasta recolha de conhecimento, implementam-se e aplicam-se as métricas à base de dados.

Concluído o processo descrito, o grupo propõe-se a avaliar quais as métricas mais vantajosas a ser implementadas, dependendo do tipo de situação a que são aplicadas. Para isto, será necessário definir diferentes tipos de métricas e identificar em que pontos é que umas são melhores do que as outras por forma a facilitar as escolhas futuras no que concerne objeto de estudo.

Capítulo 2

Medidas de Similaridade

As medidas de similaridade são usadas para avaliar semelhanças entre dois objetos, um valor alto é indicativo de que estes são idênticos e um valor baixo de que são completamente diferentes. O cenário contrário acontece com as medidas de dissimilaridade ou distância, em que se medem as diferenças entre dois objetos.

Sendo que $\mathcal{A}$ define um conjunto finito de dados categóricos com I objetos distintos, e A_i um conjunto finito de J variáveis (atributos) que descrevem propriedades de cada objeto A_i pertencente a $\mathcal{A}$.

$$\mathcal{A} = A_1 \times A_2 \times A_3 \times ... \times A_i \times ... \times A_I, \text{ onde } A_i = \{a_{i1}, a_{i2}, a_{i3}, ..., a_{ij}, ..., a_{iJ}\} \ \forall i = 1...I, j = 1...J$$

Um *dataset* D indica uma coleção de dados de $\mathcal{A}$ e pode ser definido matematicamente da seguinte forma:

$$D = (x^n)_{n=1}^N \text{ , onde } x^n \in \mathcal{A} \text{ e } x^n = (x_1^n, x_2^n, .., x_i^n, .., x_I^n) \ \forall i = 1...I$$

Para cada atributo i, identifica-se uma métrica específica para A_i notada por $s_i(x, y)$ como a medida de semelhança entre os dois objetos x e y e $d_i(x, y)$ como a medida de distância.
De uma forma geral, pode definir-se distância através dos valores de *matching* sendo que:

$$d_i(x, y) = \begin{cases} 0, & \text{se } x_i = y_i \\ 1, & \text{se } x_i \neq y_i \end{cases}$$

Porém, esta definição não é tão linear como se espera, e como tal é fundamental realizar a sua extensão:

$$e_i(x, y) = \begin{cases} \alpha, & \text{se } x_i = y_i \\ \beta, & \text{se } x_i \neq y_i \end{cases}$$

Portanto, a noção de complementaridade destas duas métricas deve incluir esta extensão da definição de distância por forma a contribuir com um maior rigor nos seus cálculos, e respetivos resultados.

$$e_i(x, y) = \beta * d_i(x, y) + \alpha * (1 - d_i(x, y)) \text{ , onde } s_i(x, y) = 1 - d_i(x, y)$$

Dados dois objetos $x, y \in \mathcal{A}$, é possível determinar a similaridade $S(x, y)$ entre o par, o qual $S_i(x, y)$ corresponde à similaridade entre dois valores do atributo i e ω_i representa o peso da similaridade S_i.

$$S(x, y) = \sum_{i=1}^{I} \omega_i * S_i(x, y) \text{ , onde } \omega_i > 0$$

2.1 Base de dados

Para, posteriormente, se realizar uma melhor compreensão de alguns conceitos abordados neste trabalho, e aplicação de forma prática desses mesmos conceitos, adoptou-se uma Base de Dados (artificial) exemplo, construída pelos autores. Esta Base de Dados é composta por 4 conjuntos de atributos, com 12 registos cada, não existindo qualquer valor nulo nas observações.

Caracterização do espaço dos atributos:

$\mathcal{A} = A_1 \times A_2 \times A_3 \times A_4$

Caracterização do conjunto dos atributos:

$A_1 = \{Azul, Verde, Amarelo, Roxo, Vermelho, Preto\}$
$A_2 = \{Circulo, Quadrado, Rectangulo, Losango\}$
$A_3 = \{Vertice, Aresta, Angulo\}$
$A_4 = \{Miguel, Duarte, Filipa\}$

índice	A_1	A_2	A_3	A_4
x^1	Preto	Rectângulo	Vértice	Miguel
x^2	Azul	Quadrado	Vértice	Miguel
x^3	Preto	Rectângulo	Aresta	Duarte
x^4	Vermelho	Quadrado	Vértice	Miguel
x^5	Azul	Circulo	Ângulo	Filipa
x^6	Amarelo	Circulo	Ângulo	Miguel
x^7	Azul	Rectângulo	Vértice	Filipa
x^8	Verde	Losango	Aresta	Duarte
x^9	Amarelo	Quadrado	Vértice	Duarte
x^{10}	Azul	Losango	Aresta	Filipa
x^{11}	Verde	Losango	Aresta	Filipa
x^{12}	Roxo	Rectângulo	Vértice	Filipa

Tabela 2.1: Tabela de Registos

2.2 Definição da ponderação de ω_i

O peso da similaridade, ω_i, pode ser calculado de diversas formas, sendo que se valorizam os dois tipos mais comuns:

- O caso uniforme onde $\omega_i = \frac{1}{I}$, sendo I o número de objetos distintos na base de dados ;
- O caso geral onde $\omega_i = |A_i|^\alpha$, sendo A_i um conjunto finito de atributos que descrevem propriedades de cada objeto i, e $\alpha \in \mathbb{Z}^*$.

Neste trabalho, para efeitos de estudo e diferentes análises da base de dados, aquando da implementação da ponderação $\omega_i = |A_i|^\alpha$, consideraremos valores para α :

- $\alpha = 1$, então $\omega_i = |A_i|^1$, onde se dá mais importância a atributos mais numerosos, e menos importância a atributos menos numerosos, valorizando assim os atributos iguais entre dois registos diferentes da base de dados pertencentes a atributos numerosos;
- $\alpha = -1$, então $\omega_i = \frac{1}{|A_i|^1}$, onde atributos mais numerosos serão mais diferenciadores e por isso menos valorizados, ao contrário de atributos menos numerosos que serão mais valorizados e irão originar categorias mais grosseiras, obtendo assim uma visão mais colectiva do problema.

Na subsecção seguinte, aplicam-se estes três tipos de cálculos à base de dados artificial, para completar a noção abordada anteriormente.

2.2.1 Exemplo

Cálculo da similaridade com ponderações ω_i entre o elemento x^1 e o elemento x^2, onde as suas componentes x_1 e x_2 são iguais, e entre o elemento x^1 e o elemento x^3, onde as suas componentes x_3 e x_4 são iguais. Os registos x^1, x^2 e x^3 pertencem á Base de Dados exemplo observada na Tabela 2.1 :

$$S(x^1, x^2) = \omega_1 * S_1(x^1, x^2) + \omega_2 * S_2(x^1, x^2) + \omega_3 * S_3(x^1, x^2) + \omega_4 * S_4(x^1, x^2)$$

- Utilizando ponderação $\omega_i = \frac{1}{I}$:

  1. Cálculo da similaridade entre os elementos x^1 e x^2

     $S(x^1, x^2) = \frac{1}{4} * 0 + \frac{1}{4} * 0 + \frac{1}{4} * 1 + \frac{1}{4} * 1 = \frac{2}{4}$

  2. Cálculo da similaridade entre os elementos e x^1 e x^3

     $S(x^1, x^3) = \frac{1}{4} * 1 + \frac{1}{4} * 1 + \frac{1}{4} * 0 + \frac{1}{4} * 0 = \frac{2}{4}$

- Utilizando ponderação $\omega_i = |A_i|$:

  1. Cálculo da similaridade entre os elementos x^1 e x^2

     $S(x^1, x^2) = 6 * 0 + 4 * 0 + 3 * 1 + 3 * 1 = 6$

  2. Cálculo da similaridade entre os elementos x^1 e x^3

     $S(x^1, x^3) = 6 * 1 + 4 * 1 + 3 * 0 + 3 * 0 = 10$

- Utilizando ponderação $\omega_i = \frac{1}{|A_i|}$:

  1. Cálculo da similaridade entre os elementos x^1 e x^2

     $S(x^1, x^2) = \frac{1}{6} * 0 + \frac{1}{4} * 0 + \frac{1}{3} * 1 + \frac{1}{3} * 1 = \frac{2}{3}$

  2. Cálculo da similaridade entre os elementos x^1 e x^3

     $S(x^1, x^3) = \frac{1}{6} * 1 + \frac{1}{4} * 1 + \frac{1}{3} * 0 + \frac{1}{3} * 0 = \frac{10}{24}$

2.3 Tabela de frequências

Seja $A_i = \{a_{i1}, a_{i2}, \cdots, a_{iJ}\}$ o atributo número i, definimos por $S_{ij} = \{x^n \in D \text{ tal como } x_i^n = a_{ij}\}$. A frequência associada é dada por :

$$f_{ij} = \frac{\#D_{ij}}{N}$$

2.3.1 Exemplo cálculo tabela de frequências

a_{1j}	$f_1(j)$
$_j = Azul$	$\frac{4}{12}$
$_j = Verde$	$\frac{2}{12}$
$_j = Amarelo$	$\frac{2}{12}$
$_j = Roxo$	$\frac{1}{12}$
$_j = Vermelho$	$\frac{1}{12}$
$_j = Preto$	$\frac{2}{12}$

Tabela 2.2: Tabela de frequências do atributo A_1

Conjunto de frequências para o atributo $A_1 \rightarrow \{\frac{4}{12}, \frac{2}{12}, \frac{2}{12}, \frac{1}{12}, \frac{1}{12}, \frac{2}{12}\}$

a_{2j}	$f_2(j)$
$_j = Circulo$	$\frac{2}{12}$
$_j = Quadrado$	$\frac{3}{12}$
$_j = Rectângulo$	$\frac{4}{12}$
$_j = Losango$	$\frac{3}{12}$

Tabela 2.3: Tabela de frequências do atributo A_2

Conjunto de frequências para o atributo $A_2 \rightarrow \{\frac{2}{12}, \frac{3}{12}, \frac{4}{12}, \frac{3}{12}\}$

a_{3j}	$f_3(j)$
$_j = Vertice$	$\frac{6}{12}$
$_j = Aresta$	$\frac{4}{12}$
$_j = Angulo$	$\frac{2}{12}$

Tabela 2.4: Tabela de frequências do atributo A_3

Conjunto de frequências para o atributo $A_3 \rightarrow \{\frac{6}{12}, \frac{4}{12}, \frac{2}{12}\}$

a_{4j}	$f_4(j)$
$_j = Miguel$	$\frac{4}{12}$
$_j = Duarte$	$\frac{3}{12}$
$_j = Filipa$	$\frac{5}{12}$

Tabela 2.5: Tabela de frequências do atributo A_4

Conjunto de frequências para o atributo $A_4 \rightarrow \{\frac{4}{12}, \frac{3}{12}, \frac{5}{12}\}$

Podemos assim, também definir uma semelhança entre atributos associada à sua frequência na base de dados com

$$S_i(x, y) = \begin{cases} f_{ij}, & \text{se } x_i = y_i = a_{ij} \\ 0, & \text{se } x_i \neq y_i \end{cases}$$

Definimos assim, que o cálculo da similaridade entre elementos de uma base de dados é dada por:

$$S(x, y) = \sum_{i=1}^{I} S_i(x, y)$$

2.3.2 Exemplo

O cálculo da similaridade entre elementos através da semelhança dos seus atributos associados ás suas frequências na base de dados, é influenciado largamente pela base de dados em estudo, ao contrário do cálculo da similaridade com ponderações. Tal é possível observar no exemplo seguinte, onde pretendemos calcular a similaridade entre os elementos x^1 e x^2, e entre os elementos x^1 e x^3 da nossa Base de Dados na Tabela 2.1.

1. Cálculo da similaridade entre os elementos x^1 e x^2 :

   $S(x^1, x^2) = S_1(x^1, x^2) + S_2(x^1, x^2) + S_3(x^1, x^2) + S_4(x^1, x^2)$

   $S(x^1, x^2) = 0 + 0 + f_{3Vertice} + f_{4Miguel}$

   $S(x^1, x^2) = 0 + 0 + \frac{6}{12} + \frac{4}{12}$

   $S(x^1, x^2) = \frac{10}{12}$

2. Cálculo da similaridade entre os elementos x^1 e x^3 :

   $S(x^1, x^3) = S_1(x^1, x^3) + S_2(x^1, x^3) + S_3(x^1, x^3) + S_4(x^1, x^3)$

   $S(x^1, x^3) = f_{1Preto} + f_{2Rectângulo} + 0 + 0$

   $S(x^1, x^3) = \frac{2}{12} + \frac{4}{12} + 0 + 0$

   $S(x^1, x^3) = \frac{6}{12}$

Capítulo 3

Implementação

Agora que introduzimos e exemplificamos as métricas que iremos aplicar a dados categóricos, como forma de medição de semelhanças entre estes, podemos começar a pensar como iremos implementar estas métricas, dado o dataset relativo a características de diversos cogumelos.

Além de uma primeira abordagem da implementação das métricas explicada acima, realizada na base de dados feita pelo grupo, é também feita uma exemplificação com uma porção mais pequena do dataset (30 elementos escolhidos aleatoriamente), de forma a podermos validar os resultados obtidos pelos cálculos das distâncias, em 2 datasets distintos.

Assim, a partir do estudo das métricas a utilizar, seguido de processo elementar de validação da aplicação destas métricas, é possível começar a sua implementação no dataset destinado para este trabalho prático. Este será abordado nos tópicos seguintes, bem como os resultados obtidos pelas diferentes métricas, assim como as conclusões que o grupo retirou após realizar este trabalho.

3.1 Descrição da implementação

Antes de falarmos da implementação das métricas abordadas acima neste documento, é importante começarmos por analisar o dataset em questão. Isto porque, como podemos ver acima, algumas métricas são mais influenciadas pelo próprio dataset do que outras.

Assim, existem 23 diferentes características de cogumelos no nosso dataset (23 atributos), entre 8124 cogumelos, variando sobre a forma que o cogumelo tem ou se é comestível/venenoso, até à sua cor, apesar de todos estes valores corresponderem a valores nominativos/categóricos.

Após uma primeira análise do dataset, foram retirados alguns valores em falta, sendo retiradas as linhas que continham estes valores, ficando com 5644 dados no nosso dataset, com as mesmas 23 colunas.

Depois, guardamos os dados numa lista, contendo os diversos cogumelos e as suas respectivas características. Temos também um array com os diferentes valores únicos que cada coluna pode ter, pois será usada nas nossas métricas para atribuir maior ou menor importância aos atributos, dependendo se estes são numerosos ou não (linha 5, com o nome nbitem). É desta forma que iremos fazer para aplicar ponderações diferentes aos atributos do dataset, dependendo da métrica utilizada.

De seguida, começamos por implementar em código as diferentes métricas que iremos utilizar e que pode ser visto em anexo, da linha 2 à linha 38. Depois, fizemos uma validação elementar que poderá ser vista no tópico seguinte.

O objetivo será escolher um cogumelo e, aplicando diferentes métricas para descobrir a sua vizinhança, com cogumelos semelhantes x% ao cogumelo escolhido, analisar se estes são da mesma classe predicativa (venenosos ou comestíveis) que o nosso cogumelo, verificando que diferentes métricas dão resultados diferentes e dependem do contexto em que são usadas.

Atributo	Nr^{o} Valores únicos
1	2
2	6
3	4
4	10
5	2
6	9
7	4
8	3
9	2
10	12
11	2
12	6
13	4
14	4
15	9
16	9
17	2
18	4
19	3
20	8
21	9
22	6
23	7

Tabela 3.1: Tabela com os diferentes valores únicos que os atributos podem ter

3.2 Validações da implementação

Para podermos validar as implementações das diferentes métricas, foi feito uma experiência com apenas 30 elementos do dataset *mushrooms* e com a base de dados criada pelo grupo. Isto permitiu, não só entender os resultados obtidos pelas diferentes métricas num universo mais pequeno, como também comprovar os mesmos, fazendo os cálculos nós próprios. Para isso, são escolhidos três elementos de cada dataset, sendo estes comparados pelos valores das semelhanças entre estes, com as diferentes métricas.

Abaixo é possível ver os resultados obtidos pela implementação das diferentes métricas.

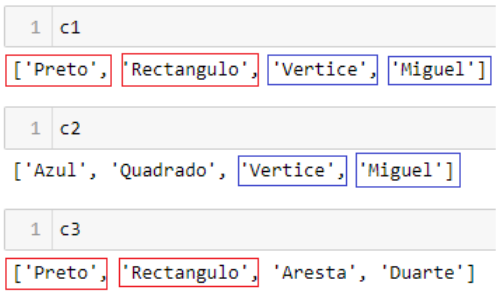

Figura 3.1: Comparação entre os 3 primeiros elementos da base dados criado pelo grupo

ω_i	$S(x_1, x_2)$	$S(x_1, x_3)$
$\frac{1}{I}$	0.5	0.5
$\lvert A_i \rvert$	6	10
$\frac{1}{\lvert A_i \rvert}$	0.667	0.417
f_{ij}	0.833	0.5

Tabela 3.2: Tabela de resultados para validação - dataset personalizado

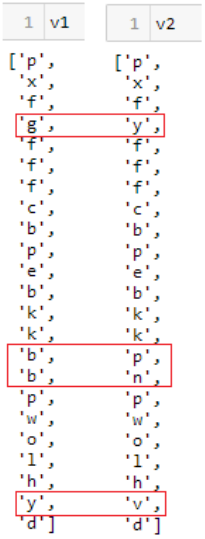

Figura 3.2: Comparação entre o primeiro e o segundo cogumelo do dataset

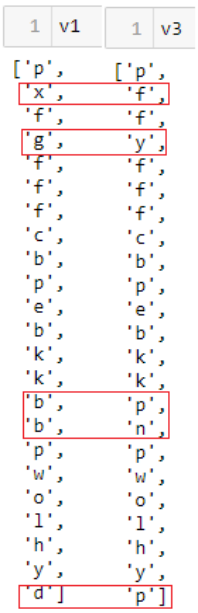

Figura 3.3: Comparação entre o primeiro e o terceiro cogumelo do dataset

ω_i	$S(x_1, x_2)$	$S(x_1, x_3)$
$\frac{1}{I}$	0.826	0.783
$\lvert A_i \rvert$	93	86
$\frac{1}{\lvert A_i \rvert}$	5.323	5.181

Tabela 3.3: Tabela de resultados para validação - dataset *Mushrooms*

Capítulo 4

Aplicação

Após já termos abordado teoricamente as métricas a implementar,termos validado na prática essas mesmas métricas e, de forma a obtermos resultados mais exactos, termos retirado os cogumelos com *missing values* presentes na nossa base de dados neste iremos, neste capitulo, aplicar todas essas métricas à totalidade da base de dados *Mushroom*.

4.1 Descrição da implementação

Iniciaremos por escolher dois registos da base de dados, em que um registo é referente a um cogumelo *edible*, e o outro registo é referente a um cogumelo *poisonous*. Com estes cogumelos de diferentes *classes*, pretendemos estudar quais as métricas, de entre as anteriormente mencionadas, mais apropriadas para determinar a *class* de um cogumelo.

Para o número de cogumelos da vizinhança relativamente ao cogumelo de referencia ser estatisticamente representativo devíamos considerar que este número corresponderia a 5% dos dados presentes na base de dados *Mushroom*, no entanto, o resultado do calculo usando este método, era sempre 1 portanto optamos por realiza-lo usando 5 números de vizinhança diferentes,sendo um o recomendado e os restantes 25%, 35%, 50% e 75% dos dados presentes. A escolha destas percentagens foi feita com base em alguns testes para se verificarem valores de *Accuracy* diferentes de 1 para as diversas métricas.

Primeiramente foram calculados os valores de similaridade (para as 3 métricas diferentes) entre os cogumelos de referência e os restantes cogumelos presentes na base de dados, sendo depois todos esses mesmos valores ordenados de forma decrescente ou seja, de forma a ficar-se com os cogumelos com maior similaridade em primeiro. Depois de ordenados os valores, retira-se apenas os primeiros N valores, sendo N o número de vizinhos que pretendemos ter na vizinhança. Para cada um desses N elementos, é verificado se a sua *class* corresponde à mesma do cogumelo de referência sendo depois calculado o nível de *accuracy* de cada métrica para ambos os cogumelos de referência.

4.2 Cogumelos referência

Como referido anteriormente, seleccionamos dois cogumelos, cada um com uma *class* diferente, *edible* (cogumelo com index 2147 no *Dataframe Mushroom*) e *poisonous* (cogumelo com index 5101 no *Dataframe Mushroom*). Os atributos de cada um desses cogumelos são:

class	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
edible	f	f	e	t	n	f	c	b	u	t	b	s	s	w	w	p	w	o	p	n	y	d
poisonous	x	y	y	f	f	f	c	b	p	e	b	k	k	p	p	p	w	o	l	h	v	p

Tabela 4.1: Tabela de atributos dos cogumelos de referência

4.3 Calculo da *Accuracy*

O Calculo da *Accuracy*, tal como foi referido anteriormente, foi realizado com 5 tamanhos de vizinhança diferentes. Sendo os valores dos cinco tamanhos diferentes de vizinhanças, representados por percentagem do tamanho da base de dados utilizada como vizinhança, como é demonstrado na tabela seguinte:

$Vizinhanca$	*Tamanho*
75%	4250
50%	2850
35%	2000
25%	1500
5%	300

Tabela 4.2: Tabela de Vizinhanças

Para determinar qual o sucesso da implementação de cada métrica na determinação do atributo *class*, foi aplicada a seguinte fórmula:

$$Accuracy = \frac{\#class}{\#vizinhanca}$$

Onde $\#class$ corresponde ao número de registos na vizinhança que têm a mesma *class* que o registo de referência, e $\#vizinhanca$ corresponde ao número total de registos que se encontram dentro da vizinhança de cada um dos registos de referência. Com esta fórmula obtemos assim, o rácio de acerto das métricas por nós estudadas.

Relembrando que o valor que define quais os registos que estão na vizinhança ou não de um registo de referência, foi abordado e definido previamente.

Os resultados obtidos, após aplicação das três métricas mencionadas anteriormente nestes estudos, são os seguintes:

ω_i	*edible*	*poisonous*
$\frac{1}{I}$	1	1
$\lvert A_i \rvert$	1	1
$\frac{1}{\lvert A_i \rvert}$	1	1

Tabela 4.3: Tabela de resultados - Vizinhança de 5%

ω_i	*edible*	*poisonous*
$\frac{1}{I}$	1	0.88
$\lvert A_i \rvert$	0.964	0.912
$\frac{1}{\lvert A_i \rvert}$	1	0.8913

Tabela 4.4: Tabela de resultados - Vizinhança de 25%

ω_i	*edible*	*poisonous*
$\frac{1}{I}$	0.954	0.718
$\lvert A_i \rvert$	0.926	0.758
$\frac{1}{\lvert A_i \rvert}$	0.882	0.714

Tabela 4.5: Tabela de resultados - Vizinhança de 35%

ω_i	*edible*	*poisonous*
$\frac{1}{I}$	0.8505	0.5786
$\lvert A_i \rvert$	0.8568	0.5839
$\frac{1}{\lvert A_i \rvert}$	0.8863	0.5586

Tabela 4.6: Tabela de resultados - Vizinhança de 50%

ω_i	*edible*	*poisonous*
$\frac{1}{I}$	0.7955	0.4586
$\lvert A_i \rvert$	0.8054	0.4456
$\frac{1}{\lvert A_i \rvert}$	0.7849	0.4508

Tabela 4.7: Tabela de resultados - Vizinhança de 75%

Capítulo 5

Conclusão

Após feita uma reflexão sobre os resultados obtidos, podemos perceber de imediato que há uma diferença muito acentuada entre as duas *classes*, onde nos valores de vizinhança com percentagem mais baixa podemos reparar que os cogumelos que pertencem à *class edible* tem uma *accuracy* muito elevada, em alguns casos 100% . Relativamente à *class poisonous* já temos uma perda mais significativa com valor que já ficam por volta dos 88%.

À medida que o tamanho da vizinhança vai aumentando conseguimos observar quedas nos níveis *accuracy*, quando aplicadas com valores de tamanho da vizinhança de 35% da totalidade da base de dados, as métricas ainda mantêm valores altos quando aplicadas à *class edible* apesar destas descerem para os 88%, enquanto que quando aplicadas à *class edible* voltamos a ter perdas maiores já chegando a 70% de *accuracy*.

Este padrão mantém-se nos outros dois teste onde podemos ver uma queda na *accuracy* para valores a rondar os 80% quando aplicadas à *class edible* e uma queda bastante superior de mais de 50% quando aplicadas à *class poisonous*, no entanto esta diferença de valores tão grande é influencia por,como já foi referido anteriormente, terem sido retirados *missing values* da base de dados, sendo que nestes registos retirados, a *class poisonous* é a mais representada.

Relativamente à diferença de valores entre métricas, podemos confirmar que com tamanho da vizinhança de 25% da base de dados, as métricas que utilizam $\omega_i = \frac{1}{I}$ e $\omega_i = \frac{1}{|A_i|}$ tiveram uma *accuracy* de 100% enquanto que a métrica que utiliza $\omega_i = |A_i|$ teve uma *accuracy* de 96%, quando são aplicadas à *class edible*.

Quanto mais se vai aumentando o tamanho da vizinhança conseguimos perceber que a métrica com $\omega_i = |A_i|$ é a que se mantém mais constante, sendo que todas perdem *accuracy*. No entanto ao longo dos teste esta perde menos, tendo no último teste uma *accuracy* de cerca de 80% enquanto que as outras duas já chegam aos 78% e 79%.

Quando aplicadas á *class poisonous* acontece o oposto, em que a métrica com $\omega_i = |A_i|$ nos valor de percentagem mais pequenos é mais exacta, mas á medida que vai aumentando perde a exactidão e acaba por ser a métrica que utiliza $\omega_i = \frac{1}{I}$ a mais constante. Analisando estes factores podemos perceber que entre si as métricas não variam muito a *accuracy*, sendo métrica com $\omega_i = |A_i|$ quem demonstrou variar mais, em contraste com as métricas com $\omega_i = \frac{1}{I}$ e $\omega_i = \frac{1}{|A_i|}$ que tiveram valores extremamente semelhantes.

Podemos concluir, após esta análise, que a métrica mais eficiente para encontrar os valores que pertencem à *class edible*,quando se trabalha com percentagens de vizinhança maiores, é aquela que aplica a ponderação de $\omega_i = |A_i|$, no entanto, se se trabalhar com percentagens menores, a mais adequada seria aquela que se aplica a ponderação $\omega_i = \frac{1}{I}$. Relativamente à *class poisonous*, temos o oposto, em que aquela que é mais eficiente para se trabalhar com uma percentagem de vizinhança maior é que aplica a ponderação de $\omega_i = \frac{1}{I}$, porem quando se trabalhar com uma percentagem de vizinhança menor a mais adequada seria aquela que se aplica a ponderação de $\omega_i = |A_i|$.

Capítulo 6

Bibliografia

dos Santos, Tiago RL, and Luis E. Zárate. "Categorical data clustering: What similarity measure to recommend?."Expert Systems with Applications 42.3 (2015): 1247-1260.

Boriah, Shyam, Varun Chandola, and Vipin Kumar. "Similarity measures for categorical data: A comparative evaluation."Proceedings of the 2008 SIAM international conference on data mining. Society for Industrial and Applied Mathematics, 2008.

Capítulo 7

Anexo 1

```
1
2  ## caso uniforme
3  def distance_Hamming(x, y):
4    s = 0
5    for i in range(len(x)):
6      if x[i] == y[i]:
7        s += 1
8    return s/len(x)
9
10 ## caso geral
11 def distance_weighted_Hamming(x, y, weight):
12   s = 0
13   for i in range(len(x)):
14     if x[i] == y[i]:
15       s += 1 * weight[i]
16   return s
17
18 ## por frequencia
19 def frequency_table():
20   c = []
21   for attribute in LT:
22     c.append(Counter(attribute))
23   table = []
24   for counter in c:
25     row = {}
26     for x in counter.items():
27       row[x[0]] = x[1]/N
28     table.append(row)
29   return table
30
31 def similarity(x, y, table):
32   s = 0
33   for i in range(len(x)):
34     if(x[i] == y[i]):
35       print('S' + str(i) + ':', x[i], y[i], table[i][x[i]])
36       s += table[i][x[i]]
37   print('\n')
38   return s
39
40 #calculo da similaridade para o caso uniforme
41 def unif_similarity(x, y):
42     s = 0
```

```
    for i in range(len(x)-1):
        if(x[i+1] == y[i+1]):
            s += 1/(len(x)-1)
    return s

#calculo da similaridade para o caso geral com alfa igual a 1
def A_similarity(x, y, wei):
    s = 0
    for i in range(len(x)-1):
        if(x[i+1] == y[i+1]):
            s += wei[i+1]
    return s

#calculo da similaridade para o caso geral com alfa igual a -1
def invA_similarity(x, y, wei):
    s = 0
    for i in range(len(x)-1):
        if(x[i+1] == y[i+1]):
            s += 1/wei[i+1]
    return s

#calculo da vizinhanca do cogumelo com index "id_cog" usando caso uniforme
#com o numero de vizinhos igual a "nr_viz" e respetivo
#calculo da accuracy de metrica para esses valores
def unif_vizinhanca(id_cog, nr_viz):
    val = mushroom.values
    res = []
    sim = 0
    for i in range(len(val)):
        sim = unif_similarity(val[id_cog], val[i])
        res.append((i,sim))
        print("Similarity with", i , "->", sim)

    res = sorted(res, key=lambda tup: tup[1], reverse=True)

    e = 0
    for r in res[:nr_viz]:
        if(val[r[0]][0] == val[id_cog][0]):
            e += 1
            #print("val:", val[r[0]], r[1])

    print("Nr.", val[id_cog][0]+":", e)
    print("perc: ", e/nr_viz)


#calculo da vizinhanca do cogumelo com index "id_cog" usando
#caso geral com alfa igual a 1 com o numero de vizinhos igual a "nr_viz"
#e respetivo calculo da accuracy de metrica para esses valores
def A_vizinhanca(id_cog, nr_viz):
    val = mushroom.values
    res = []
    sim = 0
    for i in range(len(val)):
        sim = A_similarity(val[id_cog], val[i], nbitem)
        res.append((i,sim))
        print("Similarity with", i , "->", sim)

    res = sorted(res, key=lambda tup: tup[1], reverse=True)

```

```
    e = 0
    for r in res[:nr_viz]:
        if(val[r[0]][0] == val[id_cog][0]):
            e += 1
            #print("val:", val[r[0]], r[1])

    print("Nr.", val[id_cog][0]+":", e)
    print("perc: ", e/nr_viz)


#calculo da vizinhanca do cogumelo com index "id_cog" usando
#caso geral com alfa igual a -1 com o numero de vizinhos igual a "nr_viz"
#e respetivo calculo da accuracy de metrica para esses valores
def invA_vizinhanca(id_cog, nr_viz):
    val = mushroom.values
    res = []
    sim = 0
    for i in range(len(val)):
        sim = invA_similarity(val[id_cog], val[i], nbitem)
        res.append((i,sim))
        print("Similarity with", i , "->", sim)

    res = sorted(res, key=lambda tup: tup[1], reverse=True)
    e = 0
    for r in res[:nr_viz]:
        if(val[r[0]][0] == val[id_cog][0]):
            e += 1
            #print("val:", val[r[0]], r[1])

    print("Nr.", val[id_cog][0]+":", e)
    print("perc: ", e/nr_viz)
```