Aprendizagem e Mineração de Dados

Projeto Final – A

Semestre de inverno 2023/2024 - MI1D

Grupo 04
Gonçalo Silva – 47255
João Rocha – 47196
Luís Morgado - 51358

Docente
Eng.º Paulo Trigo

Data: 1/10/2023

Índice

Conteúdo

Projeto A

Introdução

Com este trabalho, pretende-se explorar e compreender inter-relações entre dados, informação e conhecimento, aplicando esses conceitos em cenários práticos e explorando diversas técnicas e métodos que permitam converter dados em conhecimento, de modo a desenvolver a competência de analisar, modelar e validar um projeto de mineração de dados.

Enquadramento

O centro médico especializado em oftalmologia, MedKnow, utiliza um sistema de gestão de bases de dados (SGBD) que armazena todos os dados reunidos ao longo do tempo referentes a cada consulta. A MedKnow forneceu o arquivo "d01_lenses.xls", contendo um conjunto de dados específico relacionado à atividade de prescrição de lentes nas últimas duas semanas (conforme apresentado na Tabela 1), algumas premissas e padrões conseguem ser identificados, fornecendo informação valiosa para a identificação de tendências na atividade diária da clínica oftalmológica.

age	prescription	astigmatic	tear_rate	lenses
young	myope	yes	normal	hard
young	myope	no	normal	soft
young	hypermetrope	yes	reduced	none
young	hypermetrope	no	normal	soft
young	hypermetrope	no	reduced	none
presbyopic	myope	yes	reduced	none
presbyopic	myope	yes	normal	hard
presbyopic	hypermetrope	yes	reduced	none
presbyopic	hypermetrope	yes	normal	none
presbyopic	hypermetrope	no	normal	soft
presbyopic	hypermetrope	no	reduced	none
pre-presbyopic	myope	yes	reduced	none
pre-presbyopic	myope	yes	normal	hard
pre-presbyopic	myope	no	normal	soft
pre-presbyopic	hypermetrope	yes	normal	none
pre-presbyopic	hypermetrope	no	normal	soft

Tabela 1 - snippet de dados relacionado com a atividade de prescrição de lentes

Com base nos dados da Tabela 1, podemos afirmar que a *Medknow* categoriza os seus clientes com base na sua idade (jovem, pré-presbiópico, presbiópico), tipos de prescrição (miopia e hipermetropia), presença de astigmatismo (sim/não), taxa de lágrima (normal/reduzida) e para cada um deles, podem ser prescritos 3 tipos diferentes de lentes (nenhuma, moles, rígidas).

Análise dos dados

Depois de uma análise completa deveremos ser capazes de saber que tipo de lentes devem ser escolhidas para cada paciente dependendo da sua idade e dos seus aspetos de saúde ocular.

Podemos observar que os dados da Tabela 1, são estruturados num formato tabular, com valores nominais em domínio discreto, quanto à sua semântica, o dataset apresenta 5 valores, que podem ser divididos em 4 atributos, sendo estes a sua idade (jovem, pré-presbiópico, presbiópico), tipos de prescrição (miopia e hipermetropia), presença de astigmatismo (sim/não), taxa de lágrima (normal/reduzida) e 1 classe/alvo que são as lentes, que podem ser representadas por 3 tipos diferentes de lentes (nenhuma, moles, rígidas).

Modelo de dados

Foi construído um modelo concetual de dados utilizando a notação Entidade-Relação (Entity-Relationship notation) para o apoio operacional (trabalho diário) e um modelo lógico derivado do mesmo. Tais modelos são cruciais para a organização dos dados e futura análise.

Modelo Entidade-Relação

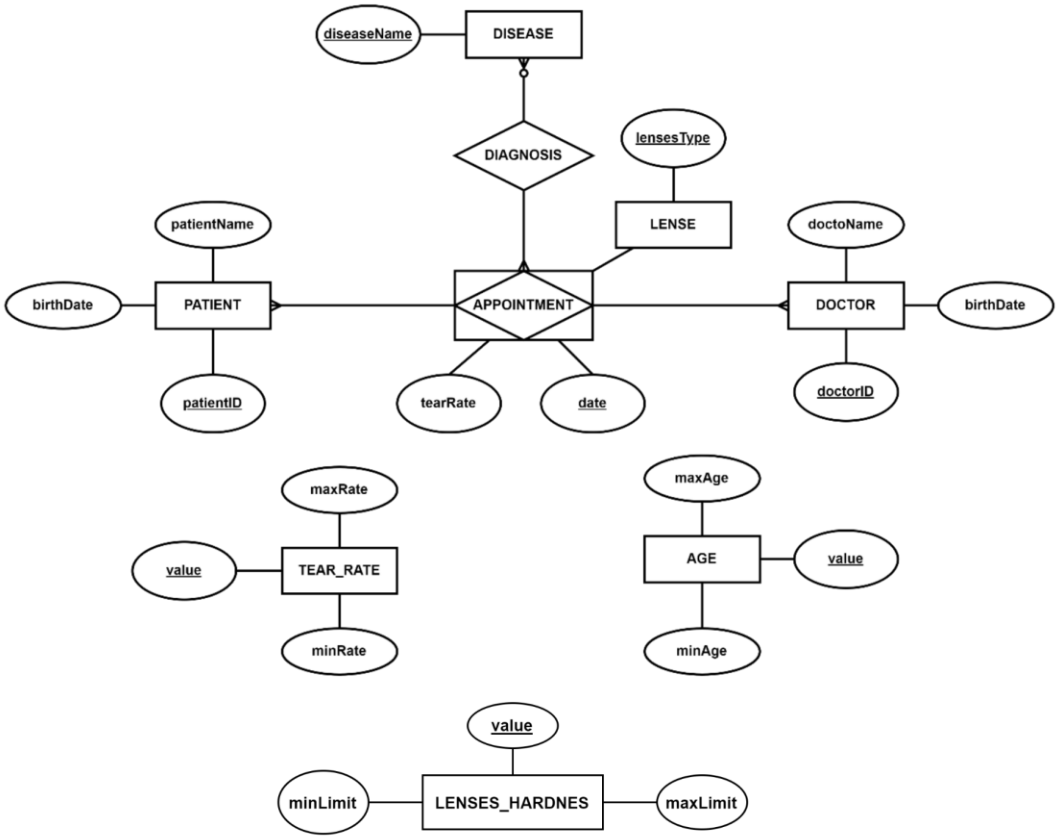


Figura 1 – Modelo Entidade-Relação

Modelo Lógico

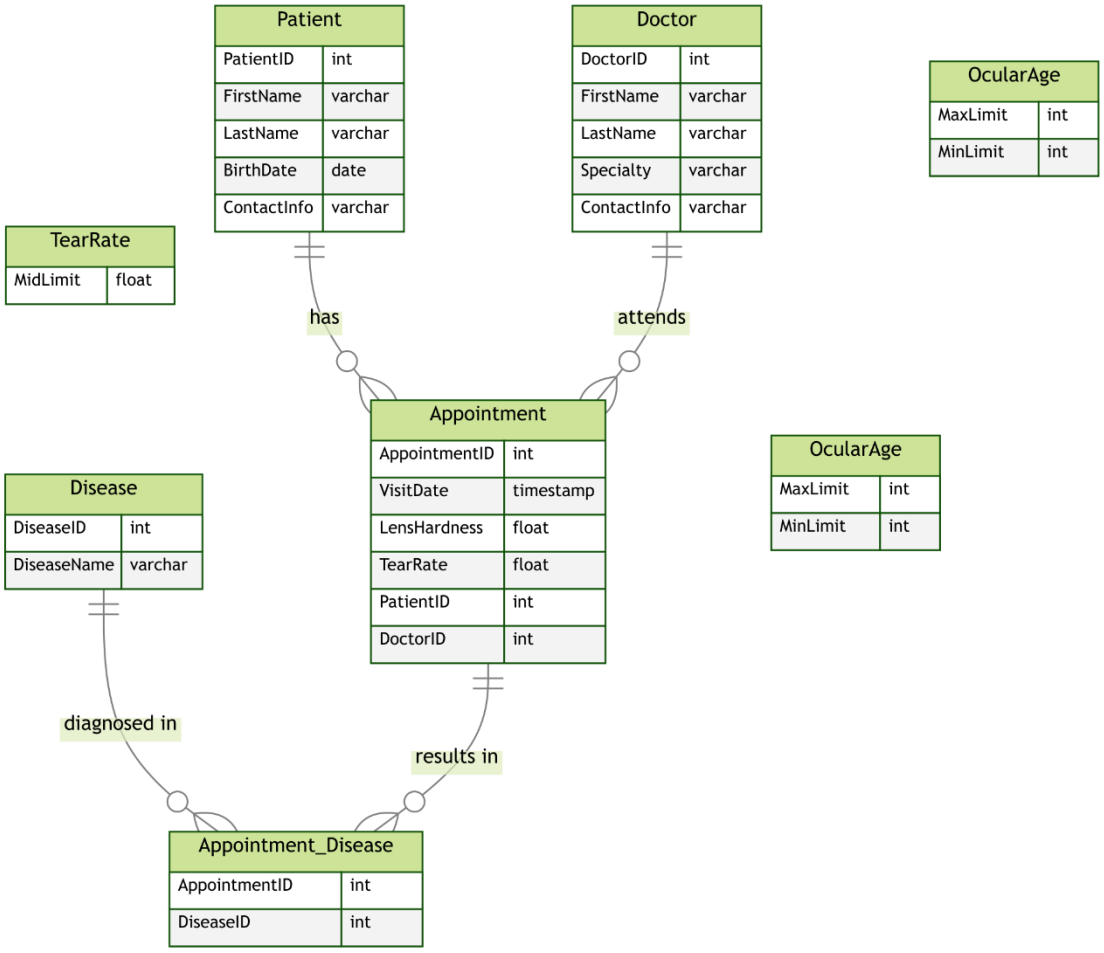


Figura 2 – Modelo Lógico

Detalhes de Implementação Base de dados

Modelo de Dados e Associações entre Tabelas

O modelo de dados é centrado na tabela "Appointment", que serve como o núcleo das informações clínicas. Essa tabela está relacionada com todas outras tabelas do sistema. Para associar as informações sobre as doenças a uma consulta, foi criada a tabela "DIAGNOSIS." Nessa associação, reconhecemos que uma única consulta pode estar relacionada a mais de uma doença. No entanto, no contexto clínico, esperamos que haja exatamente uma doença do tipo "myope" ou "hypermetrope" associada a cada consulta. Além disso, opcionalmente, pode estar associada a uma doença do tipo "astigmatic."

Atributos Chave e Associações Diretas:

O valor de "TearRate," assume-se que é atribuído um único valor à consulta, refletindo uma medição específica, estando este valor diretamente descrito na tabela “APPOINTMENT”. Da mesma forma, "LensHardness" também é atribuído como um único resultado, no entanto, esses resultados foram armazenados numa tabela separada, uma vez que representam os resultados específicos obtidos durante cada consulta.

Discretização

Com base no conjunto de dados apresentado na *tabela 1,* podemos concluir que todos os valores dos atributos são valores nominais, com um intervalo de valores bem definido.
Para chegar a estes valores, é necessário um processo de discretização, pois normalmente dados como a idade e a taxa de lágrima costumam ser valores numéricos.
Como mencionado anteriormente, precisamos converter todos os dados numéricos em valores nominais e podemos fazer isso atribuindo nomes a determinados intervalos de valores.

Intervalos de Valores e Mapeamento:

Para caracterizar os intervalos de valores possíveis para as medidas de idade, TearRate e LensHardness, foram criadas três tabelas de mapeamento. Estas tabelas servem como referências para definir faixas de valores aceitáveis. Cada intervalo é mapeado por meio das tabelas "OcularAge," "TearRate," e "LensHardness," proporcionando uma maneira padronizada de representar e consultar os resultados clínicos dentro dessas faixas.

Essas associações e representações foram projetadas para garantir uma estrutura de dados coerente e eficaz. O uso de tabelas de mapeamento ajuda a manter a consistência e a integridade dos dados, fornecendo uma estrutura clara para as informações clínicas em questão. Os valores foram organizados de acordo com a seguinte tabela:

mínimo	máximo	valor
Ocular Age		
0	30	"young"
30	45	"pre-presbyopic"
45	99	"presbyopic"
Lenses Hardness		
0	0.1	"none"
0.1	0.3	"soft"
0.3	0.99	"hard"
Tear Rate		
0	0.5	"reduced"
0.5	1.0	"normal"

Tabela 2 - Discretização dos dados

Este processo é especialmente importante neste projeto, pois o Classificador One Rule, só pode trabalhar com valores discretos.

1R – One Rule

O classificador "One Rule" tem como ideia principal encontrar a regra que melhor discrimina as classes alvo com base em um único atributo.
De forma a descobrir qual o atributo que melhor classifica os dados, é construída uma regra que relaciona cada um dos valores desse atributo com um dos valores do conceito (classe).

Processo para descobrir a "One Rule":

1. Escolher um atributo para análise (o processo deve ser repetido para cada atributo).
2. Calcular a frequência de cada valor do atributo em relação às classes alvo.
3. Calcular o erro para cada valor do atributo. $Erro = \frac{Total - Frequência}{Total}$
4. Escolher os pares (valor-atributo, valor-classe) com menor erro, aleatório se igual.

5. Calcular o erro do atributo somando os erros dos pares escolhidos.
6. Depois de obter as regras e os erros de cada atributo (itens acima), é escolhido o atributo com o menor erro e considera-se as regras associadas a esse atributo como representativas do conjunto de dados.

Implementação 1R – One Rule

Para a implementação do algoritmo 1R foram desenvolvidas 2 implementações, uma primeira, denominada OneR, baseada no código fornecido em aulas laboratoriais, tendo sido utilizadas e adaptadas às respectivas funções necessárias nomeadamente as funções de geração de matrizes de frequência, probabilidades e erros, entre outras funções. e uma outra implementação, denominada NativeOneR, desenvolvida sem utilização prévia de código fornecido

O seguinte diagrama de fluxo ilustra a execução e processamento interno do modelo OneR tendo em conta a implementação da classe OneR:

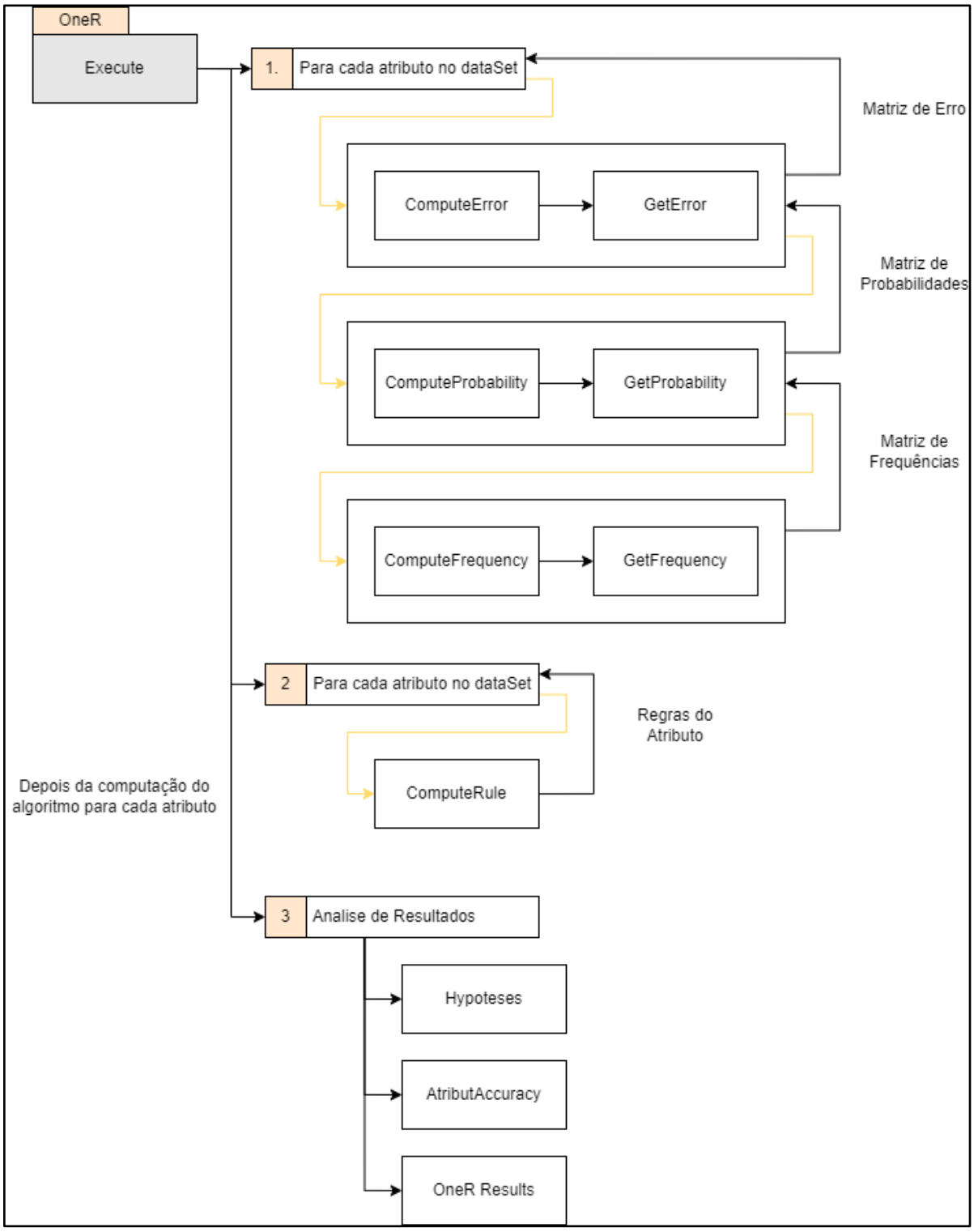


Figura 3 - Discretização dos dados

O algoritmo começa por carregar um ficheiro de input para o dataset sobre o qual vai ser executado o algoritmo. Inicialmente são feitas as computações necessárias para obter as regras/erro associadas a cada atributo, gerando por exemplo o seguinte resultado para o atributo tear_rate:

```
_____________________________________
Rule and error for: tear_rate :
(tear_rate, normal, soft) : 0.500
(tear_rate, reduced, none) : 0.000

Total Atribut Error: 0.3125
```

Figura 4 - Regras do atributo tear_rate

À medida que são geradas regras para os atributos, estas são armazenadas num dicionário. No fim da execução da geração de regras para cada atributo é amostrado um conjunto de resultados "Hypoteses" que representam todas as regras e respectivas % de erro geradas para cada atributo, posteriormente é mostrado o conjunto de resultados AtrrAccuracy que representa a soma de %erro para cada atributo e por fim é escolhida a regra de menor erro que é a que melhor representa o dataSet

```
_________
HYPOTHESES
- ( attr, valueAttr, valueTarget ) : ( error, total )
(age, pre-presbyopic, none) : (3, 5)
(age, presbyopic, none) : (2, 6)
(age, young, none) : (3, 5)
(prescription, hypermetrope, none) : (3, 9)
(prescription, myope, hard) : (4, 7)
(astigmatic, no, soft) : (2, 7)
(astigmatic, yes, none) : (3, 9)
(tear_rate, normal, soft) : (5, 10)
(tear_rate, reduced, none) : (0, 6)


____________
attrACCURACY
- attr : ( error, total ) # error / total
age : (8, 16)  # 0.5
prescription : (7, 16)  # 0.4375
astigmatic : (5, 16)  # 0.3125
tear_rate : (5, 16)  # 0.3125


________________
One-R (Best Rules)
Atributo com menor erro: astigmatic, 0.3125
- Reading the rules: if attr is valueAttr then targetClass is valueTarget with x probability of error
- ( attr, valueAttr, valueTarget ) : (error, total)
(astigmatic, no, soft) : (2, 7)
(astigmatic, yes, none) : (3, 9)
```

Figura 5 - Modelo 1R: Resultados finais

As funções denominadas como "get_" são responsáveis por efetivamente gerar a matriz respetiva pretendida enquanto que as funções denominadas como "compute_" são responsáveis por efetivamente chamar as funções get e realizar a amostragem da matriz resultante a chamada da função "get".

Arvore de decisão – ID3

As árvores de decisão são uma técnica de aprendizagem que permite a classificação de instâncias em categorias tendo como base um conjunto de atributos observados.
O algoritmo ID3 (*inductive decision tree*) é um dos algoritmos mais utilizados na construção de árvores de decisão.
O processo de construção da árvore de decisão com o algoritmo ID3 pode ser resumido nas seguintes etapas:

1. Inicialmente, o algoritmo avalia todos os atributos disponíveis para determinar qual deles resulta no maior ganho de informação, e escolhe esse atributo para ser o nó raiz da árvore de decisão.
2. É criado um nó filho para cada valor do atributo escolhido na etapa anterior
3. Para cada nó filho criado, o algoritmo ID3 repete o processo, avaliando quais atributos são mais informativos para a classificação daquele subconjunto específico.
4. A condição de paragem ocorre quando todos os exemplos em um subconjunto têm o mesmo valor para um atributo, tornando o subconjunto homogêneo e não exigindo mais divisões.

Naive bayes - NB

O classificador Naïve Bayes, fundamentado no Teorema de Bayes, é um algoritmo de aprendizagem que trata todos os atributos como igualmente importantes durante a fase de treinamento, ao contrário do classificador 1R, que se baseia em apenas um atributo.

Este classificador presume que os atributos são independentes entre si, o que pode não ser totalmente verdade na prática. No entanto, essa simplificação permite um treinamento mais eficiente e é particularmente útil para lidar com problemas de classificação de texto e mineração de dados.

Implementação ID3 e NB

Ao contrário do modelo 1R, a implementação dos classificadores ID3 e NB, não tiveram que ser implementados de raiz, então para tal foi utilizada a biblioteca *sklearn* para a implementação de ambos os modelos.

Foram criadas 2 classes, *'ModelID3'* e *'ModelNaiveBayes'*, onde cada uma contém a implementação do algoritmo correspondente.

Cada uma destas classes tem como base a implementação do método **fit()**, e **predict()**.

O método **fit()** é usado para treinar o modelo de Naive Bayes com o dataset de treino.

Depois de ser realizado o treino, o modelo está pronto para fazer previsões, para tal é usado o método **predict()** que faz previsões com base no modelo treinado, recebendo um conjunto de dados *X* como entrada e usa o modelo de Naive Bayes para prever as classes correspondentes.

Classificação dos modelos

Foi feita a análise da classificação dos três modelos implementados anteriormente, para tal foram usadas as métricas Accuracy, Precision, Recall e f1 score para comparação dos resultados.

Os dados foram classificados 5 vezes para cada um dos três classificadores, tendo-se obtido os seguintes resultados:

*Classificação do modelo **1R***			
Accuracy	*Precision*	*Recall*	*f1 score*
0.22	0.17	0.22	0.19
0.55	0.62	0.55	0.52
0.33	0.11	0.33	0.16
0.33	0.13	0.33	0.18
0.55	0.62	0.55	0.52
Média: 0.39	Média: 0.33	Média: 0.39	Média: 0.31

Tabela 3 – Desempenho modelo 1R

*Classificação do modelo **ID3***			
Accuracy	*Precision*	*Recall*	*f1 score*
1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0
0.6	0.4	0.6	0.46
1.0	1.0	1.0	1.0
Média: 0.92	Média: 0.88	Média: 0,92	Média: 0.89

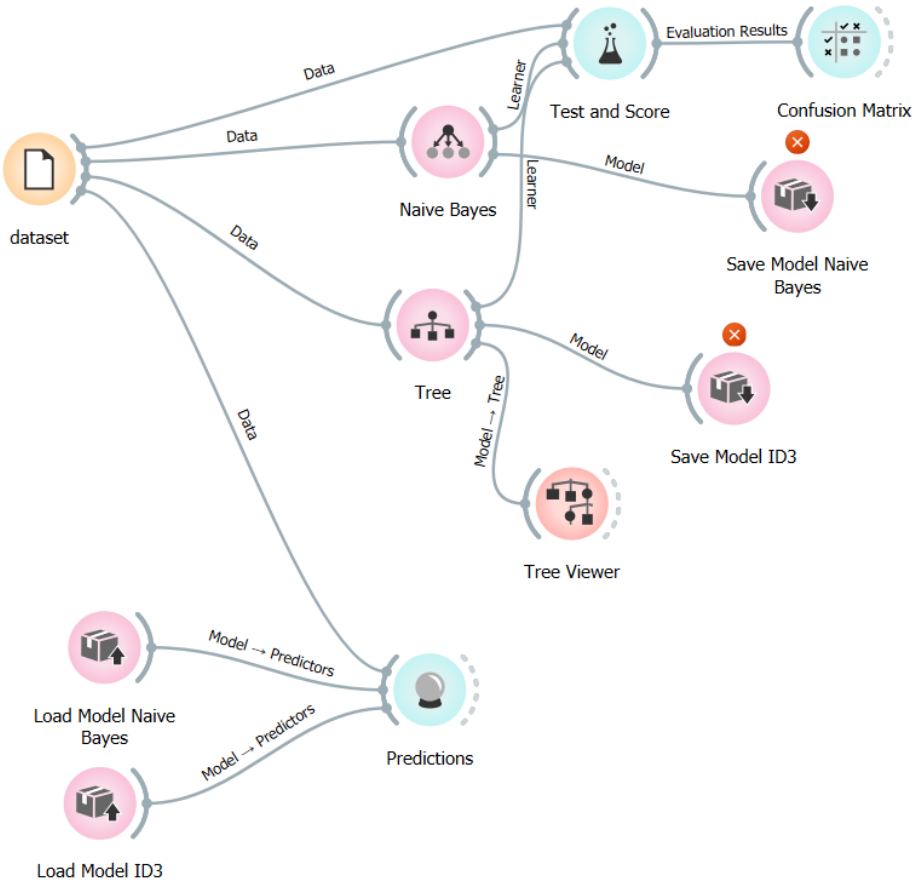
Tabela 4 – Desempenho modelo ID3

*Classificação do modelo **ID3***			
Accuracy	*Precision*	*Recall*	*f1 score*
1.0	1.0	1.0	1.0
0.6	0.46	0.6	0.5
1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0
Média: 0.92	Média: 0,89	Média: 0.92	Média: 0.9

Tabela 5 – Desempenho modelo NB

Tendo em conta as tabelas relativas ao desempenho de cada um dos classificadores, podemos observar que o classificar que demonstrou o menor desempenho foi o classificador 1R, devido ao facto deste modelo considerar apenas 1 atributo para classificar ao contrario dos modelos ID3 e NB.

Implementação D3 e Naïve Bayes - Orange



Desempenho dos modelos:

Model	AUC	CA	F1	Prec	Recall	MCC
Naive Bayes	1.000	1.000	1.000	1.000	1.000	1.000
Tree	1.000	1.000	1.000	1.000	1.000	1.000

Projeto A1

Introdução

Com o Projeto A1, pretende-se usar o classificador 1R construído anteriormente, mas desta vez, sobre um conjunto de dados mais extenso e com *missing values*.
Este conjunto de dados denomina-se por "FungiData" e consiste na descrição de amostras de 23 espécies diferentes de cogumelos da família Agaricus e Lepiota.
Para este dataset também serão usadas diversas ferramentas do Orange para analisar e classificar melhor os respetivos dados.

Análise dos dados

Ao analisar o conjunto de dados presente em *dataset_long_name_ORIGINAL.csv* podemos dizer que o dataset FungiData contem 8416 instâncias e tem uma estrutura tabular com valores nominais em domínio discreto, quanto à sua semântica, o dataset apresenta 23 valores com existência de omissões, de todos eles 22 são atributos e 1 classe com 2 possíveis valores, sendo eles EDIBLE ou POISONOUS.

Conversão de um ficheiro csv para um ficheiro tab

Visto que o ficheiro *dataset_long_name_ORIGINAL.csv* não apresenta o dataset em conformidade com o formato adequado para que possa ser processado pelo Orange, foi realizado um script python *CsvToTab.py*. Este script começa por criar 2 headers, sendo o primeiro para descrever o tipo de domínio do atributo, e o segundo para assinalar qual a classe do dataset e por fim é feita a leitura de todo o dataset ao qual são concatenados os 2 headers criados.
O ficheiro .tab criado terá o nome de *dataset_long_name_ORIGINAL.tab*.

Aplicação do Algoritmo 1R

Para aplicar o algoritmo 1R ao conjunto de dados presentas no dataset "FungiData", foi utilizado o classificador 1R implementado anteriormente.

Os dados foram classificados 5 vezes para o classificador 1R, sendo o dataset dividido em 70% para dataset de treino e 30% para dataset de teste, tendo-se obtido os seguintes resultados:

*Classificação do modelo **1R***			
Accuracy	*Precision*	*Recall*	*f1 score*
0.98	0.98	0.98	0.98
0.98	0.98	0.98	0.98
0.98	0.98	0.98	0.98
0.99	0.99	0.99	0.99
0.98	0.98	0.98	0.98
Média: 0.98	Média: 0.98	Média: 0.98	Média: 0.98

Tabela 6 – Desempenho modelo 1R

Matriz de Confusão 1R		Predicted Labels	
		EDIBLE	POISONOUS
True Labels	EDIBLE	1396	0
	POISONOUS	38	1091

Tabela 7 – Matriz de Confusão do modelo 1R

Ao observarmos as tabelas 6 e 7, podemos concluir que a classificação do modelo 1r no dataset "FungiData" mostra um desempenho bastante elevado, sendo capaz de apresentar medias com valores muito próximos do 100%. A partir da matriz de confusao podemos observar que apenas 34 de 2525 amostras de cogumelos foram mal classificadas.

De seguida, guardou-se no ficheiro "out_oneR_mushrooms.txt" a informação que o algoritmo 1R escolheu como sendo o atributo com o menor erro.

```
(attr, valueAttr, valueTarget) : (error, total)
(odor, ALMOND, EDIBLE) : (0, 279)
(odor, ANISE, EDIBLE) : (0, 278)
(odor, CREOSOTE, POISONOUS) : (0, 134)
(odor, FISHY, POISONOUS) : (0, 414)
(odor, FOUL, POISONOUS) : (0, 1532)
(odor, MUSTY, POISONOUS) : (0, 34)
(odor, NONE, EDIBLE) : (81, 2617)
(odor, PUNGENT, POISONOUS) : (0, 182)
(odor, SPICY, POISONOUS) : (0, 421)
```

Figura 7 – model 1R output

Implementação Tree e Random Forest Classification – Orange

Utilizou-se o Orange Data Mining para analisar um dataset de cogumelos, visando classificá-los como comestíveis ou venenosos. O dataset foi dividido 70% para treino e 30% para teste.

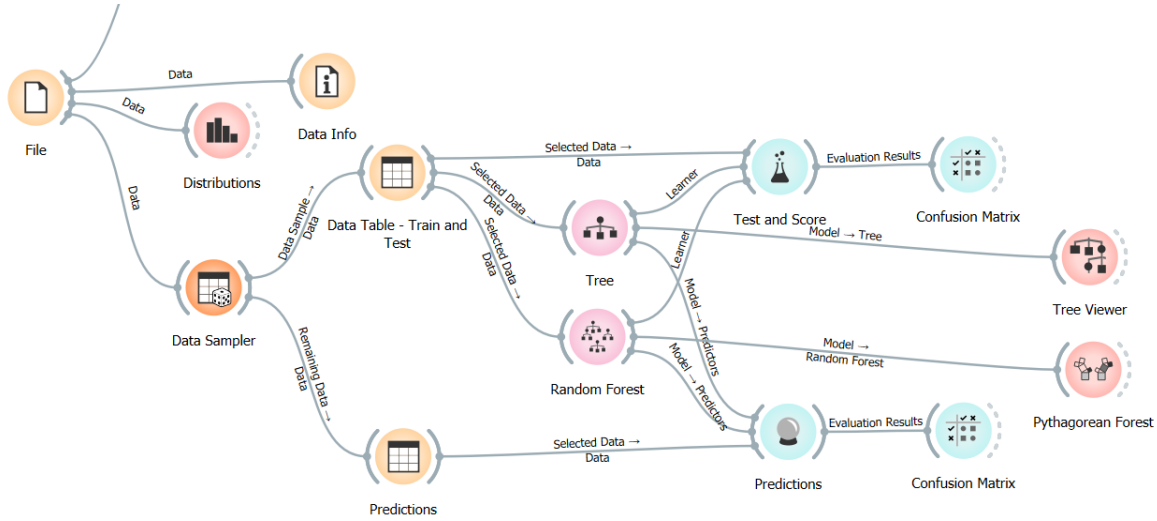
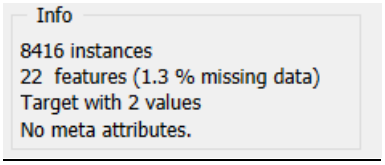


Figura 8 – Implementação no Orange

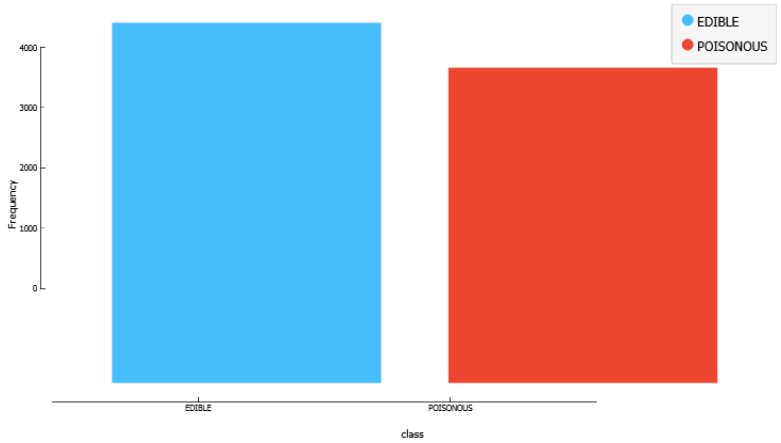
Total de Instâncias:

O widget *Data Table* mostrou que o dataset contém 8416 instâncias, 22 características (1,3% dados em falta).



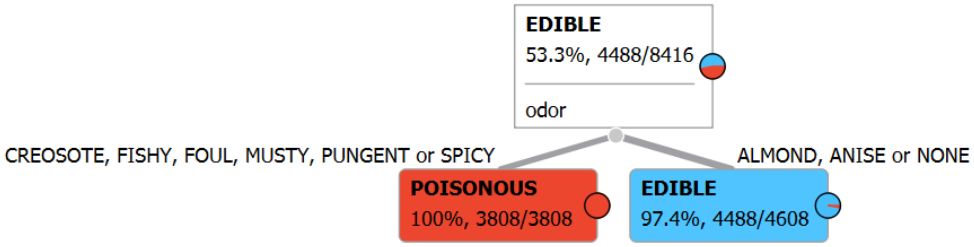
Número de Instâncias de Cada Classe:

O widget *Distributions* revelou a existência de 4488 (53,33%) instâncias de cogumelos comestíveis e 3928 (46,67%) venenosos.



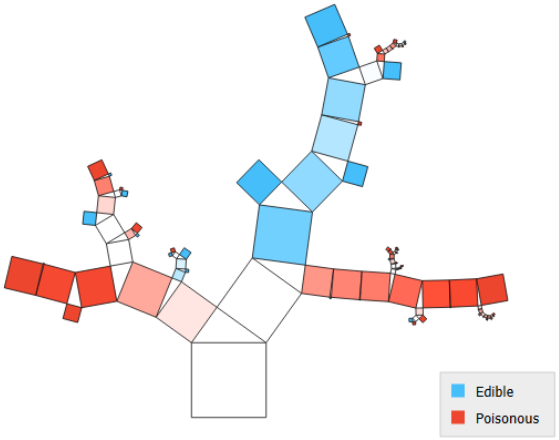
Classificação por Árvore de Decisão:

O widget *Tree* foi usado para gerar um modelo de árvore de decisão, e o *Tree Viewer* ajudou na visualização deste.



Classificação por Random Forest:

Implementou-se um modelo usando o widget *Random Forest*, com visualização através do *Pythagorean Forest*.



Previsões Usando os Dois Classificadores:

O widget *Predictions* permitiu visualizar as previsões feitas pelos modelos de Árvore de Decisão e Random Forest para os dados de teste.

O modelo de Decisão Random Forest apresenta melhores resultados comparando com o elevado numero de falso positivos.

Confusion matrix for Tree (showing number of instances)

		Predicted EDIBLE	POISONOUS	Σ
Actual	**EDIBLE**	1346	0	**1 346**
	POISONOUS	34	1144	**1 178**
	Σ	**1 380**	**1 144**	**2 524**

Confusion matrix for Random Forest (showing number of instances)

		Predicted EDIBLE	POISONOUS	Σ
Actual	**EDIBLE**	1345	1	**1 346**
	POISONOUS	2	1176	**1 178**
	Σ	**1 347**	**1 177**	**2 524**

Conclusão

Com a realização do projeto A e projeto A1 no âmbito da disciplina de Aprendizagem e Mineração de Dados, foram postos em prática os novos conhecimentos obtidos na teórica sobre a correta extração, análise e classificação de dados.

No projeto A, embora o conjunto de dados utilizado neste trabalho seja relativamente pequeno, o modelo One-R demonstrou ser uma abordagem simples para realizar classificações com base em um único atributo, permitindo a interpretação das regras de classificação. No entanto, ao avaliar o desempenho global dos modelos, observamos que outras técnicas mais complexas, como árvores de decisão (ID3) e o classificador Naïve Bayes, superaram o One-R em termos de precisão e capacidade de generalização.

No projeto A1, a dimensão do dataset passou a ser significativamente maior, o que refletiu num melhor desempenho na aprendizagem feita pelo modelo 1R.

De forma geral os objetivos do projeto A e A1 foram cumpridos de forma satisfatória.