ISEL
INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Final Project A

Aprendizagem e Mineração de Dados

Grupo 3:

Duarte Valente | A47657

João Valido | A51090

Docente:

Eng. Paulo Trigo

Curso: MEIM

2023

Índice

Índice de ilustrações

Introdução

a) Descrição

Este trabalho tinha como objetivo explorar a relação entre as noções de dados, informação e conhecimento. Aplicar esses conceitos em cenários específicos e experimentar/desenvolver as técnicas que permitem "passar" dos dados ao conhecimento. Explorar a classificação e agrupamento que são nucleares para a extração de dados, a descoberta de conhecimentos, a aprendizagem automática e a recuperação de informação. As técnicas recorrem à estatística (e.g., 1R e regra de Bayes), à indução de árvores de decisão (e.g., J4.8/C4.5, ID3) baseadas em instâncias (e.g., KNN com suporte KDTree). Desenvolver a competência de análise, modelação e validação de um projeto de extração de dados. Utilizar ferramentas para gerir dados (p. ex., PostgreSQL), para descobrir conhecimentos (p. ex., extração de dados Orange) e para implementar algoritmos específicos (por exemplo, através de Python).

b) Âmbito

Este relatório insere-se no âmbito da realização da ficha laboratorial unidade curricular de Aprendizagem e Mineração de Dados, do Mestrado em Engenharia Informática e Multimédia do DEETC do ISEL.

c) Cenário

O centro médico "MedKnow" utiliza um sistema de gestão de bases de dados (SGBD) que contém todos os dados recolhidos ao longo do tempo sobre a visita de cada doente a um médico (que trabalha no "MedKnow"). O objetivo atual da equipa de oftalmologia é analisar toda a informação acumulada (ao longo do tempo) de forma a extrair padrões que forneçam indicadores úteis para apoiar a atividade de prescrição (e diagnóstico). Para atingir esse objetivo, decidiram contactar a empresa "SoftKnow" e enviar-lhes o ficheiro "d01_lenses.xls" com um excerto de dados (relacionados com a atividade de prescrição de (Numa linha, escreveram o seguinte desafio: "envie-nos um protótipo de um sistema que forneça à "MedKnow" não só o apoio operacional (trabalho quotidiano), mas também a perspetiva estratégica (padrões úteis) que pode ser extraída desses dados de trabalho quotidiano".

Desenvolvimento

a) Análise

Com base no excerto dos dados fornecido pelo "MedKnow", está relacionado com oftalmologia, uma vez que inclui informações sobre a idade dos pacientes, a prescrição, o astigmatismo, a taxa de lacrimação e o tipo de lentes que utilizam.

Seguem-se algumas suposições sobre o trabalho diário do "MedKnow" com base nos dados: Estabelecimento médico, atende pacientes de diferentes idades, incluindo indivíduos jovens, pré-presbitas e presbitas. Os dados sugerem que a "MedKnow" oferece serviços relacionados com a miopia e a hipermetropia. Prescrevem diferentes tipos de lentes para corrigir a visão, incluindo lentes duras e moles. Ofereçam consultas e exames para determinar a prescrição adequada para cada paciente. A presença das colunas "astigmatic" e "tear rate" indica que a "MedKnow" pode avaliar e tratar o astigmatismo, uma condição ocular comum, e fornecer recomendações com base na taxa de lágrimas do paciente, o que é importante para o conforto quando se usam lentes de contacto.

Os dados também mostram que o "MedKnow" oferece opções para taxas de lágrimas reduzidas, que podem exigir tratamentos especiais. Isto sugere uma abordagem abrangente aos cuidados oftalmológicos que tem em conta as necessidades e desafios individuais dos doentes.

b) Modelo Conceptual dos dados

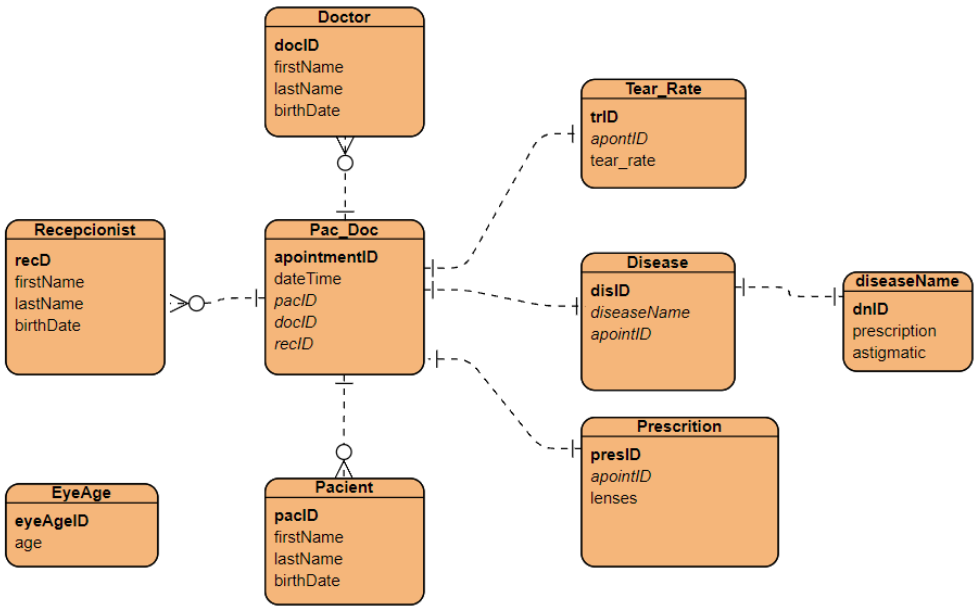

Figura 1 - Modelo Conceptual

c) Modelo lógico

1. **Entidade: "MedKnow"**
   i. Descrição: Clínica especializada em cuidados com os olhos e correção da visão.

2. **Categorias de Pacientes**
   i. **Jovens**: Pacientes de um grupo etário mais jovem.
   ii. **Presbiopia**: Pacientes com presbiopia (vista cansada relacionada à idade).
   iii. **Pré-presbiopia**: Pacientes em um grupo etário pré-presbiópico.

3. **Condições de Visão**
   i. **Miopia**: Visão ao perto, condição comum.
   ii. **Hipermetropia**: Visão ao longe, condição comum.

4. **Serviços Oferecidos**
   i. **Exames Oftalmológicos**: Exames oftalmológicos para determinar a condição de visão do paciente e suas necessidades.
   ii. **Prescrição**: Com base no exame, prescrevem lentes corretivas adaptadas à condição de visão do paciente.

5. **Tipos de Lentes Oferecidas**
   i. **Lentes Rígidas**: Lentes de contato rígidas adequadas para alguns pacientes.
   ii. **Lentes Macias**: Lentes de contato macias adequadas para outros.
   iii. **Nenhuma Lente**: Alguns pacientes podem não necessitar de lentes corretivas.

6. **Avaliações Adicionais**
   i. **Astigmatismo**: Avalia e potencialmente trata o astigmatismo.
   ii. **Taxa de Lágrimas**: Consideram a taxa de lágrimas do paciente e podem fornecer recomendações para o conforto ao usar lentes de contato.

7. **Abordagem Baseada na Idade**
   i. Personaliza os serviços com base no grupo etário do paciente, adaptando as recomendações para atender às necessidades específicas dos diferentes pacientes.

8. **Atendimento Individualizado**
   i. Adota uma abordagem individual e centrada no paciente, oferecendo diferentes tipos de lentes e avaliações para atender às necessidades únicas.

d) Implementação

1º - Criação das tabelas

De forma a implementar a solução encontrada, o primeiro passo foi criar a base de dados que suporta-se este sistema. Assim, tendo em conta o modelo lógico desenvolvido e explicado anteriormente, foi então desenvolvido o script “01_script_CREATE_TABLES_DB_MEDKNOW”, responsável por criar as tabelas necessárias, adicionar os atributos das tabelas, as chaves primarias e as relações entre elas. Assim que finalizada a construção do script, este foi executado e verificada a estrutura da base de dados.

Adição dos dados á base de dados

De seguida, foi feita a adição de dados à base de dados, utilizando também o script "02_script_POPULATE_DB_MEDKNOW.

De forma a introduzir dados representativos, optamos por introduzir dados de forma a que o resultado fosse igual ao resultado do ficheiro “d01_lenses.csv”. Assim, introduzimos numa fase inicial 3 reacionistas, 3 médicos e 16 clientes. De forma a preencher as restantes tabelas, foram de seguida introduzidos dados: de 16 consultas (uma para cada paciente), os dados da tabela diseaseName que acabou por ter 4 instancias (todas as hipóteses possíveis como mostra a figura 2), os dados da tabela disease que remetem a diseaseName que foi identificada em cada consulta, os dados da tabela tear_rate que remetem à tear rate identificada na consulta, os dados da tabela prescription que indicam o tipo de lente receitado conforme as leituras feitas na consulta e por fim os dados da tabela eyeAge que conta com 3 instancias, uma para cada idade ocular como mostra a figura 3.

Apos a introdução destes dados, verificamos que estes poderiam ser poucos, logo acabamos por adicionar mais 32 pacientes e introduzimos os dados respetivos para novas 32 consultas.

dnid	prescription	astigmatic
1	myope	true
2	myope	false
3	hypermetrope	true
4	hypermetrope	false

Figura 2 - dados tabela diseaseName

minAge	maxAge	age
0	20	young
20	45	presbyopic
45	120	pre-presbyopic

Figura 3 - dados da tabela eyeAge

Criação e exportação das views

De forma a visualizar e compreender melhor os dados, foram criadas duas views através do script “03_script_EXPORT_DB_MEDKNOW” que desempenham o papel de calcular a idade ocular de cada paciente e de apresentar todas as informações relevantes sobre cada um dos pacientes.

A primeira view, denominada "PatientEyeAge," é responsável por calcular a idade ocular de cada paciente com base na sua data de nascimento, comparando-a com os valores definidos na tabela "eyeAge" (figura 3). Assim, esta view atribui uma faixa etária à idade ocular do paciente e a exibe-a.

Já a segunda view, denominada "exportView," combina as informações de várias tabelas, incluindo dados sobre a idade ocular calculada na view anterior. Assim, esta view reúne dados como a idade ocular (view PatientEyeAge), a prescrição (tabela diseaseName), astigmatismo (tabela diseaseName), taxa de lágrima (tabela tearRate) e tipo de lentes (tabela prescription) para cada paciente.

Resumindo, estas duas views conseguem fornecer uma maneira eficaz de calcular e acessar informações relacionadas à idade ocular de cada paciente e apresentam uma visão abrangente dos dados, permitindo análises e insights valiosos no contexto da clínica.

Exportação dos dados

Assim que reunidos os dados necessários para passar á próxima fase, exportamos então a view “exportView”, pois esta reúne todas as informações necessárias para a continuação do desenvolvimento do projeto. Para exportar a view foi utilizado comando \COPY do postgres, que permite selecionar a veiw pretendida, a pasta e nome de destino e o tipo de ficheiro, que no caso foi CSV.

Análise e classificação de dados

Após a estruturação da base de dados, e de os mesmos serem exportados. Passamos à fase de análise e classificação dos dados. Fase esta onde foram utilizados os registos de cada cliente (eyeAge, prescription, astigmatic, tear_rate) para serem analizados por três diferentes algoritmos, implementados em python, de forma a prever qual o tipo de lente que deve ser receitada a cada paciente.

Em relação à importação dos dados, foi utilizada a função read_csv do pandas. De seguida foram separados os atributos da classe a ser prevista (lenses) e foram divididos os dados para treino e teste, sendo que 80% foram para treino e 20% para teste.

1. 1R

Como primeiro algoritmo, foi utilizado o "1R" (One Rule), que tem o objetivo de encontrar a regra que melhor consegue prever o tipo de lentes a utilizar. Assim, o algoritmo calcula regras de decisão com base na frequência de ocorrência de cada valor de cada atributo para cada tipo de lente e escolhe o atributo com o menor erro. Atributo esse utilizado por sua vez para prever o tipo de lente no conjunto de teste. Assim, descrevendo melhor este processo, podemos dividir o algoritmo implementado nas seguintes partes:

Para cada atributo:

- Contar a frequência com que se relaciona com o tipo de lentes;
- Calcular o erro para cada valor através da diferença do total de associações com a sua frequência.
- Escolher os pares com menor erro.
- Calcular o erro do atributo como a soma dos erros dos pares escolhidos

Assim que escolhidas as regras e os erros de cada atributo, escolher o atributo com menor erro para ser a regra que melhor consegue prever o tipo de lentes a utilizar.

Por fim, após encontrada a regra, esta é guardada em formato de *string* num ficheiro txt em anexo para ser possível de utilizar para prever outros dados caso sem ser necessário de recalcular qual a melhor regra.

2. Decision tree

Em relação ao segundo algoritmo utilizado, conhecido como ID3 (*Iterative Dichotomiser 3),* foi adotada uma maneira diferente de classificar os dados, em relação ao algoritmo 1R, visto que o ID3 cria uma árvore de decisão baseada na entropia dos valores. Entropia esta, utilizada para determinar a melhor divisão em cada nó da árvore. Possibilitando assim, separar os dados de treino nos melhores subconjuntos que representam os diferentes tipos de lentes.

Para tal, foi utilizada a função *DecisionTreeClassifier* do *SkLearn* para criar o modelo de classificação e treinar com os dados de treino.

Por fim, após a fase de treino do modelo, este foi armazenado, através da função *dump* da biblioteca *joblib,* num ficheiro, de forma a ser utilizado para fazer previsões nos dados de teste ou para ser utilizado no futuro com novos dados.

3. Naive Bayes

O último algoritmo utilizado, foi Naïve Bayes, que utiliza uma técnica de classificação probabilística baseada no *Teorema de Bayes* e na suposição ingênua (naïve) de independência condicional entre os atributos. O que torna possível prever qual o melhor tipo de lente com base numa certa instancia de dados.

Assim, para implementar o modelo Naïve Bayes, este foi criado através da função *CategoricalNB* do *SkLearn*. Tendo de seguida passado ao processo de treino com os dados de treino.

À semelhança do algoritmo ID3, este também acabou por ser armazenado num ficheiro, através da função *dump* da biblioteca *joblib,* para ser utilizado de uma maneira mais prática para classificar os dados sem ser necessário estar a treinar novamente o modelo com novos dados.

e) Avaliação dos modelos

De forma a analisar os resultados, foram então avaliados os modelos 1R, D3, e o Naïve Bayes, através de medidas como a accuracy, precision, recall e f1-score. Com o objetivo de verificar a capacidade de cada um fazer previsões precisas.

1R (One Rule):

	1R
Accuracy	0.7

Em relação ao algoritmo 1R, devido a seguir apenas uma regra simples, foi apenas avaliado através da accuracy. Tendo-se verificado que acerta em 70% das previsões que faz. O que acaba por ser um resultado positivo tendo em conta a sua maneira simples de fazer previsões.

ID3:

	ID3
Accuracy	1.0
Precision	1.0
Recall	1.0
f1-score	1.0

Navie Bayes

	Naive Bayes
Accuracy	1.0
Precision	1.0
Recall	1.0
f1-score	1.0

Em relação aos algoritmos ID3 e Navie Bayes, estes tiveram um resultado igual e bastante positivos, tendo ambos acertado todas as previsões que fizeram e obtido 100% de precisão, recall e F1-Score. O que indica que estes modelos conseguem fazer previsões perfeitamente precisas e sem erros. O que os torna altamente confiável para prever tipos de lentes.

Desenvolvimento – A1

a) Análise do dataset

Como segunda parte do projeto A foi sugerido a continuação do desenvolvimento destes classificadores, assim como o teste de outros algoritmos de classificação, mas agora para o dataset "Mushroom Database". Dataset esse que conta com 8416 amostra, onde são representadas 23 espécies de cogumelos pertencentes às famílias Agaricus e Lepiota.

Cada amostra de espécie de cogumelo, pode então ser classificada como "edible" (comestível) ou "poisonous" (venenoso), tornando este conjunto de dados crucial para a identificação segura do tipo de cogumelos que podemos encontrar. Já os atributos de cada amostra, são representados por 22 tipos diferentes que fornecem informações detalhadas sobre características como forma do chapéu, odor, cor das lamelas e muitos outros (figura 4).

Atributo	Possíveis valores
cap-shape	bell | conical | convex | flat | knobbed | sunken
cap-surface	fibrous | grooves | scaly | smooth
bruises	brown | buff | cinnamon | gray | green | pink | purple | red | white | yellow | bruises | no
odor	almond | anise | creosote | fishy | foul | musty | none | pungent | spicy
gill-attachment	attached | descending | free | notched
gill-spacing	close | crowded | distant
gill-size	broad | narrow
gill-color	black | brown | buff | chocolate | gray | green | orange | pink | purple | red | white | yellow
stalk-shape	enlarging | tapering
stalk-root	bulbous | club | cup | equal | rhizomorphs | rooted | missing
stalk-surface-above-ring	fibrous | scaly | silky | smooth
stalk-surface-below-ring	fibrous | scaly | silky | smooth
stalk-color-above-ring	brown | buff | cinnamon | gray | orange | pink | red | white | yellow
stalk-color-below-ring	brown | buff | cinnamon | gray | orange | pink | red | white | yellow
veil-type	partial | universal
veil-color	brown | orange | white | yellow
ring-number	none | one | two
ring-type	cobwebby | evanescent | flaring | large | none | pendant | sheathing | zone
spore-print-color	black | brown | buff | chocolate | green | orange | purple | white | yellow
population	abundant | clustered | numerous | scattered | several | solitary
habitat	grasses | leaves | meadows | paths | urban | waste | woods

Figura 4 - Valores de cada atributo

Em relação à distribuição das classes podemos observar que os dados disponibilizados no arquivo "dataset_long_name_ORIGINAL.csv", onde a primeira linha contém os nomes dos atributos e classe, seguida por 8416 instâncias com os valores associados a cada atributo e classe, que 53,33% destas instâncias são do tipo “edible” (comestíveis), enquanto que 46,67% são do tipo "poisonous” venenosas (figura 5).

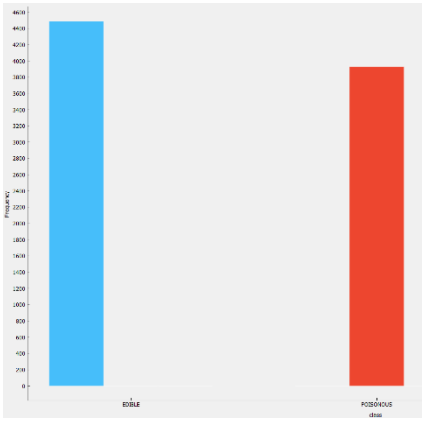

Figura 5 - Distribuição das classes

Assim, esta riqueza de informações consegue proporcionar uma base sólida para a análise e mineração destes dados, com o objetivo de contribuir para a compreensão e classificação eficiente de cogumelos.

b) Transformação dos dados

Após a analise dos dados, passamos então à fase de transformação dos dados, onde nos foi sugerido que transformássemos o ficheiro ".csv" para o formato ".tab" e com a estrutura idêntica à do ficheiro "dataset_long_name_PTS_INPUT_v01.tab" fornecido na pasta do trabalho.

Assim, para ir de encontro ao resultado pretendido, os dois ficheiros foram carregados para o OrangeDM e foram comparados os seus dados. Como primeiro passo verificamos que o atributo classe do ficheiro ".csv" era reconhecido como feature, logo foi mudado o role da "class" para "target". De seguida também verificamos que existia um tipo de classe marcado como "----------", logo foi utilizado o select row do OrangeDM para selecionar apenas as classes que não eram do tipo "----------". Para verificar no global se os datasets eram similares, utilizamos também o icon "Data Info" para verificar alguns valores como o número de colunas e linhas.

Por fim, foi utiizado o "Save Data" para salvar os dados selecionados no formato ".tab". O esquema na figura 6 representa o esquema do canvas utilizado para efetuar esta tarefa.

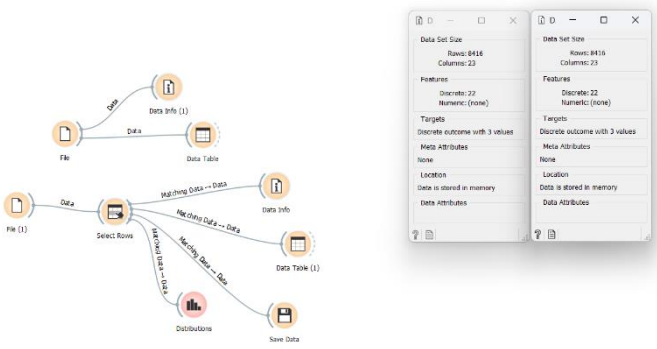

Figura 6 - Esquema do canvas no Orange DM

c) Aplicação do algoritmo 1R

Após obter o arquivo “dataset_long_name_ORIGINAL.tab” no formato desejado, carregamos o ficheiro para o script python desenvolvido e descrito no ponto **d.1** do projeto A, de forma a encontrar a melhor regra que consiga indicar a classe das amostras deste dataset.

Assim, após carregar os dados, foram divididos os dados dos atributos dos da classe e foram separados em dados para treino (80%) e dados para teste (20%).

Em relação ao algoritmo, nada foi mudado pois este foi desenvolvido na parte anterior de forma a ser possível a calcular a regra para qualquer dataset.

Após encontrado o melhor atributo e as regras para esse atributo, foram então guardados num ficheiro chamado “oneR_OUTPUT.txt” os dados calculados pelo algoritmo. Dados esses no formato: (attr, valueAttr, valueTarget) : (error, total), onde attr representa o atributo selecionado, valueAttr os valores que o atributo pode ter, valueTarget para o valor do atributo qual a classificação indicada, error o erro que essa decisão tem e o total representa o total de instancias que se existem com esse valor de atributo.

Como o dataset é bastante extenso em comparação ao dataset da MedKnow, verificamos que o algoritmo encontrou vários atributos com erro igual a zero pelo que a escolha do melhor atributo acabou por ser considerada aleatória dentro destes melhores atributos.

Exemplo do resultado armazenado no ficheiro “oneR_OUTPUT.txt”:

```
(stalk-color-above-ring, WHITE, EDIBLE) : (1363, 3777)
(stalk-color-above-ring, GRAY, EDIBLE) : (0, 477)
(stalk-color-above-ring, BUFF, POISONOUS) : (0, 350)
(stalk-color-above-ring, PINK, POISONOUS) : (452, 1511)
(stalk-color-above-ring, YELLOW, POISONOUS) : (0, 8)
(stalk-color-above-ring, ORANGE, EDIBLE) : (0, 151)
(stalk-color-above-ring, BROWN, POISONOUS) : (14, 345)
(stalk-color-above-ring, CINNAMON, POISONOUS) : (0, 36)
(stalk-color-above-ring, RED, EDIBLE) : (0, 77)
```

Figura 7 - Ficheiro oneR_OUTPUT.txt

Assim, após calculas e armazenadas estas informações, é possível utilizar o atributo selecionado e as regras encontradas para efetuar previsões nos dados de teste sendo no final mostrada a precisão destas mesmas previsões.

Resultado dados de teste: Accuracy = 0.7179334916864608

d) Classificação “Tree”

Por último foi sugerido que classificássemos estes mesmos dados utilizado o OrangeDM, nomeadamente através de uma classificação em arvore ou uma classificação Random Forest.

Assim, carregamos o ficheiro “dataset_long_name_ORIGINAL.tab” para o Orange, verificamos que os dados se encontravam bem formatados e utilizamos o Modelo “Tree” do Orange para fazer previsões e testar o modelo para estes dados.

Assim obtivemos os seguintes Resultados:

Modelo	AUC	CA	F1	Precision	Recall
Tree	0.985	0.986	0.986	0.986	0.986

Figura 8 - Resultados do Test e Score Tree

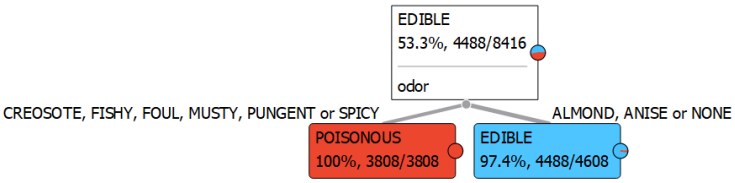

Figura 9 - Arvore encontrada

e) Classificação “Random Forest”

Já para o modelo Random Forest foi também utilizada o mesmo ficheiro e desta vez o Icon Random Forest para efetuar a classificação dos dados.

Assim para este modelo obtivemos os seguintes resultados:

Modelo	AUC	CA	F1	Precision	Recall
Random Forest	1.000	1.000	1.000	1.000	1.000

Figura 10 - Resultados Random Forest

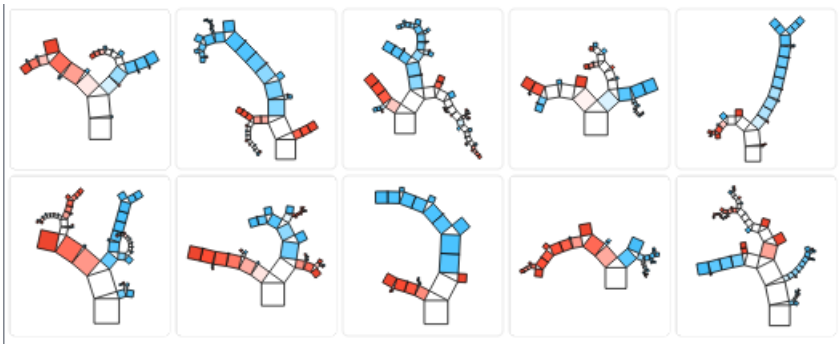

Figura 11 - Arvores do RF

f) Comparação de resultados

Para concluir, comparamos os 3 métodos de classificação utilizados para verificar o seu desempenho. E como esperado, verificamos que o método 1R apesar de conseguir classificar bem algumas amostras, tem um desempenho muito inferior aos dois outros modelos. Pois já o modelo "Tree" e o "Random Forest" obtiveram resultados muito satisfatórios, tendo até o Random Forest classificado 100% corretamente todas as amostras.

Algumas métricas utilizadas para avaliar o desempenho dos modelos:

Modelo	AUC	CA	F1	Precision	Recall
1R	0.717	-	-	-	-
Tree	0.985	0.986	0.986	0.986	0.986
Random Forest	1.000	1.000	1.000	1.000	1.000

Figura 12 - Comparação das 3 classificações

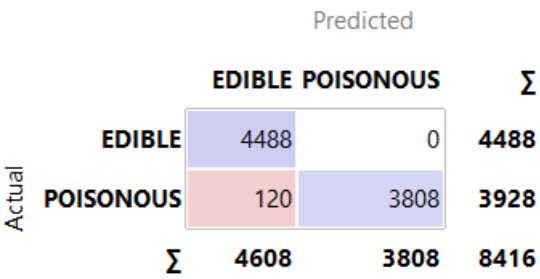

Figura 13 - Confusion matrix Tree

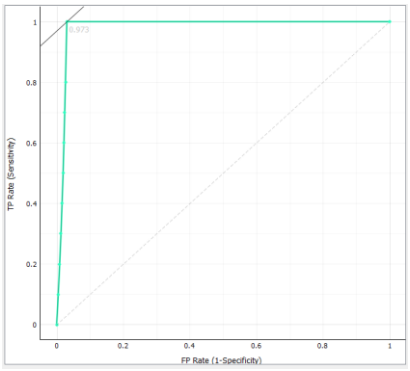

Figura 14 - Curva ROC Tree

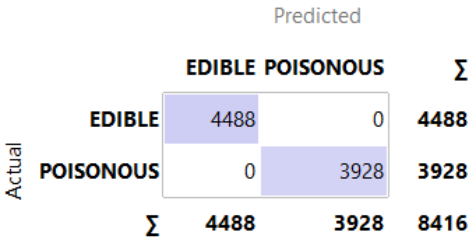

Figura 15 - Confusion matrix Random Forest

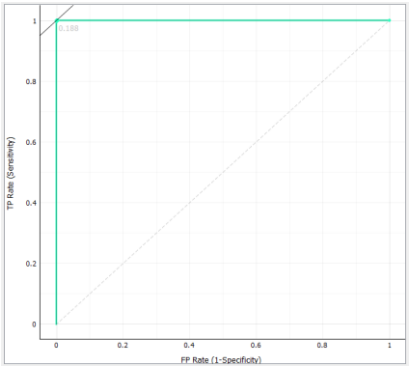

Figura 16 - Curva ROC Random Forest

Conclusão

Este trabalho teve como objetivo explorar o processo de transformação de dados em conhecimento, aplicando conceitos de mineração de dados em um cenário específico relacionado à área médica. Com início na análise de dados fornecidos pela MedKnow, identificamos os atributos relevantes e realizamos suposições sobre o funcionamento da clínica oftalmológica.

A modelação conceitual e lógica dos dados permitiu representar os elementos envolvidos no processo, desde pacientes de diferentes grupos etários até aos serviços oferecidos, tipos de lentes, avaliações adicionais e uma abordagem personalizada. A criação de tabelas no SGBD PostgreSQL e a inserção de dados proporcionaram uma estrutura sólida para as etapas seguintes.

O desenvolvimento da solução envolveu a criação de duas views, "PatientEyeAge" e "exportView," que possibilitaram calcular a idade ocular de cada paciente e reunir informações relevantes para análise. Informações essas por fim exportadas para dar seguimento aos objetivos propostos do projeto.

A análise e classificação dos dados constituíram uma parte fundamental do trabalho, onde foram utilizados três algoritmos distintos para prever o tipo de lente a ser prescrita. O algoritmo "1R" forneceu previsões com 70% de acerto, enquanto os algoritmos "ID3" e "Naïve Bayes" obtiveram 100% de precisão, recall e F1-Score, revelando-se altamente confiáveis.

Em resumo, este projeto demonstra como a mineração de dados e a análise de informações podem ser aplicadas na área médica para melhorar a tomada de certas decisões. Já em relação aos resultados obtidos com os algoritmos "ID3" e "Naïve Bayes", estes indicam que estas abordagens têm potencial para fornecer apoio valioso a uma clínica como por exemplo a MedKnow. No entanto, é importante salientar que a escolha do modelo a ser utilizado pode depender das circunstâncias específicas e da disponibilidade de dados. Este trabalho serve assim de exemplo, de como a mineração de dados pode ser aplicada na área da saúde, contribuindo para a melhoria dos cuidados oftalmológicos oferecidos aos pacientes.

Já em relação ao dataset "Mushroom Database", analisamos os dados, tendo em conta como estes são classificados e os seus atributos. Transformamos os dados de forma a pudermos classifica-los. E por fim, efetuamos a classificação e avaliação dos resultados utilizando o algoritmo 1R desenvolvido anteriormente e os modelos Tree e Random Forest do OrangeDM. Assim, podemos verificar que o 1R apesar da sua simplicidade consegue prever bem algumas amostras, mas, no entanto, não chega perto da perfeição de modelos mais complexos como o Random Forest por exemplo, que obteve um resultado perfeito. O que em situações de extrema importância, como é o caso da classificação correta do tipo de cogulemos, pode ser crucial utilizar um classificador com a menor taxa de falsos positivos possível.