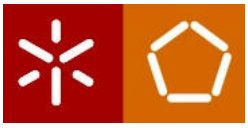


Universidade do Minho
Departamento de Informática

Aprendizagem e Decisão Inteligentes

Trabalho Prático

Grupo 12

Gonçalo Costa - A100824

Marta Rodrigues - A100743

Ema Martins - A97678

Henrique Malheiro - A97455

Índice

Índice de imagens

1. Introdução

Este projeto foi desenvolvido em prol da unidade curricular de Aprendizagem e Decisão Inteligentes, cujo objetivo era desenvolver as nossas capacidades de análise e extração de conhecimento atendendo à distribuição de dados disponíveis aplicando modelos de aprendizagem abordados ao longo do semestre em aulas teórico-práticas a dois diferentes *datasets*, sendo um desses atribuído pela equipa docente e sendo o restante à escolha pelo grupo.

De forma a proporcionar uma compreensão dos resultados obtidos e decisões tomadas, será explicada e justificada a abordagem aplicada durante a preparação, exploração e análise de ambos os *datasets*, assim como mostrar como essas decisões alteraram os resultados obtidos.

2. Metedologia

O **CRISP-DM** (Cross Industry Standard Process for Data Mining) foi a metodologia que decidimos usar para a realização deste trabalho prático.

Seguindo a sua estrutura, começamos por **entender o negócio**, no sentido de estabelecer os objetivos que pretendemos atingir com o modelo que iremos criar. Em seguida, procuramos **entender os dados** para posteriormente os **preparar** corretamente. Após isso, realizamos a **modelação** do problema e procedemos à **avaliação** do mesmo. A última fase desta metodologia corresponde à **implementação**, que no âmbito do trabalho proposto não será abordada. De salientar que, de forma a melhorar o trabalho, sempre que foi necessário por voltamos algumas etapas atrás.

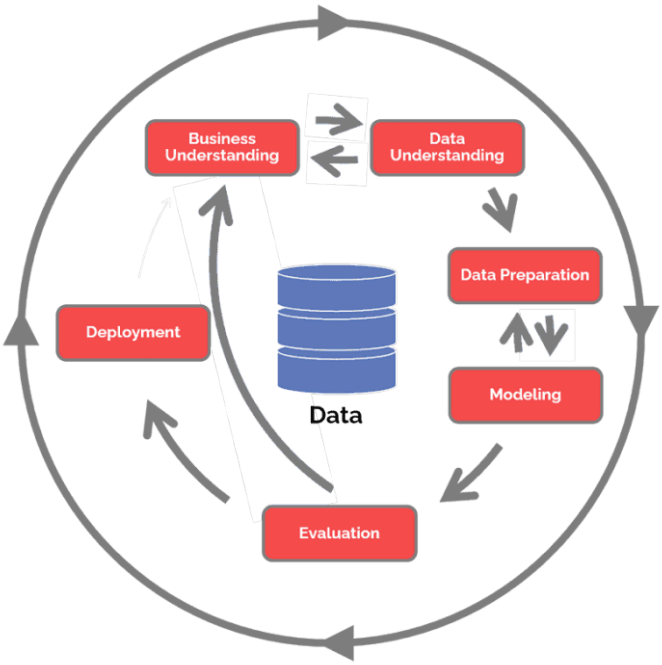


Figure 1: Modelo CRISP-DM

3. Dataset Par

3.1. Business Understanding

Dois *datasets* foram fornecidos pela equipa docente, dependendo do número do grupo o mesmo iria ter de preparar e analisar um deles, como o nosso grupo possui um número par (12) teríamos de analisar o *dataset par*.

Esse conjunto de dados representa as informações relacionadas com a condição do fígado de 538 pacientes e é composto por uma série de atributos sendo esses:

- **Age** - Idade do paciente, qualquer paciente que ultrapasse os 89 anos é listado como tendo 90 anos.
- **birth_year** - Ano em que o paciente nasceu.
- **birth_month** - Mês em que paciente nasceu.
- **birth_date** - Data de nascimento do paciente.
- **Gender** - Género com que o paciente nasceu.
- **TB** - *Total Bilirubin*, ou seja, quantidade total de Bilirrubina no sangue. A bilirrubina é um pigmento amarelo produzido quando as células vermelhas no sangue são quebradas. A bilirrubina é responsável pela cor amarela das fezes e pela cor amarela da pele e dos olhos em condições como icterícia, que ocorre quando há um acumulo anormal de bilirrubina no corpo indicando assim possivelmente problemas no fígado.
- **DB** - *Direct Bilirubin*, ou seja, **bilirrubina direta**, é uma forma de bilirrubina que foi processada pelo fígado e está pronta para ser excretada na bílis. Ela é uma parte importante do processo de excreção da bilirrubina do corpo. Quando há um aumento nos níveis de bilirrubina direta no sangue, isso pode indicar problemas no fígado ou nas vias biliares.
- **Alkphos** - *Alkaline Phosphotase*, ou seja, **Fosfatase Alcalina**, é uma enzima encontrada em vários tecidos do corpo, incluindo fígado, ossos, rins, intestinos e placenta durante a gravidez. Ela desempenha um papel importante na regulação de várias funções celulares, como o metabolismo de fosfatos e a formação óssea. Valores anormais de fosfatase alcalina podem indicar uma série de condições, como doenças do fígado.
- **Sgpt** - *Alamine Aminotransferase*, ou seja, **Transaminase Glutâmico-Pirúvica(ALT)**, é uma enzima encontrada principalmente no fígado, mas também em quantidades menores em outros tecidos do corpo, como os rins e os músculos. Ela desempenha um papel essencial no metabolismo dos aminoácidos.
- **Sgot** - *Aspartate Aminotransferase*, ou seja, **Transaminase Glutâmico-Oxalacética(AST)**, é uma enzima encontrada principalmente no fígado, coração, músculos esqueléticos, rins, pâncreas e pulmões, desempenha, também, um papel fundamental no metabolismo dos aminoácidos. Quando esses órgãos estão danificados, as células liberam AST na corrente sanguínea em quantidades maiores do que o normal, resultando em níveis elevados de AST no sangue.
- **TB(#1)** - *Total Proteins*, nível total de proteínas no sangue, se esses forem elevados, por si só, não são especificamente indicativos de doença hepática no fígado. No entanto, certos tipos de proteínas, como albumina e globulinas, são produzidos pelo fígado.
- **ALB** -*Albumin*, ou seja, **albumina**, é uma proteína produzida pelo fígado. Ela desempenha um papel importante na regulação do volume sanguíneo e no transporte de substâncias, como hormonas e nutrientes e é um indicador importante da função hepática. Baixos níveis de albumina podem ser um indicador importante da função hepática. Níveis baixos de albumina podem indicar disfunção hepática, uma vez que o fígado não está produzindo adequadamente essa proteína.
- **CHOL** - *Cholesterol*, ou seja, **colesterol**, é um tipo de lípidos encontrado no sangue e desempenha um papel essencial na estrutura das membranas celulares e na produção de hormonas. Elevações nos níveis de colesterol podem estar associadas a problemas hepáticos.

- **AG_Ratio** - *Albumin and Globulin Ratio*, ou seja, a razão entre **albumina** e **globulina**, é uma medida que pode ser útil na avaliação da função hepática e na identificação de certas condições hepáticas e não hepáticas. A albumina e as globulinas são duas das principais classes de proteínas encontradas no sangue. Em condições normais, a proporção entre albumina e globulinas no sangue é tipicamente alta (por exemplo, superior a 1). No entanto, em algumas condições, como doenças hepáticas crônicas, a proporção pode diminuir devido à diminuição na produção de albumina pelo fígado ou ao aumento na produção de globulinas, como resultado da inflamação hepática.
- **BILmg** - **Bilirrubina em Miligramas**, é uma medida mais precisa da concentração de bilirrubina no sangue, importante para detetar possíveis problemas no fígado.
- **Selector** - Indica se há presença de doença hepática no paciente. Valor que servirá como *target* no estudo deste *dataset*.

Esses atributos oferecem uma gama diversificada de dados que podem ser empregados na análise e modelagem dos dados relacionados à saúde hepática dos pacientes. Ao investigar as interações entre esses atributos e a ocorrência de doenças do fígado, torna-se viável a elaboração de modelos preditivos valiosos para o diagnóstico e tratamento dessas condições médicas. Por exemplo, uma baixa razão entre albumina e globulinas pode ser indicativa de doença hepática, mas não garantidamente, tal conclusão pode ser tirada especialmente em conjunto com outros marcadores de função hepática alterados, como níveis elevados de enzimas hepáticas (ALT, AST) ou bilirrubina.

3.2. Data Understanding

Após compreender os diferentes atributos que compõem o *dataset* começamos a explorar os dados que o mesmo continham.

Inicialmente conglomerados os intervalos em que cada atributo deve se encontrar para serem considerados "saudáveis".

Atributo	Unidade	Valor
Bilirrubina Total	mg/dL	0,3 - 1,2
Bilirrubina Direta	mg/dL	0,1 - 0,3
Fosfatase Alcalina	IU/L	44 - 147
Transaminase Glutâmico-Pirúvica	U/L	7 - 56
Transaminase Glutâmico-Oxalacética	U/L	8 - 33
Protein	g/L	60 - 83
Albumin	g/L	35 - 55
Bilirubin	mg	3 - 12
AG_Ratio	—	1,0 - 2,2

Após isso, verificamos que a maior parte dos valores encontravam se por dentro desses intervalos e reparamos na existência de *missing values* nas colunas referentes aos valores de Colestrol, AG_Ratio e Bilirrubina em Miligramas.

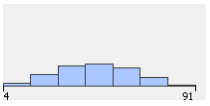
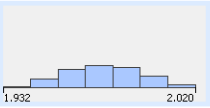
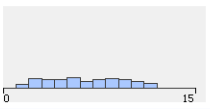
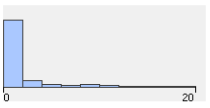
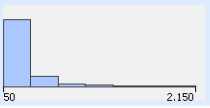
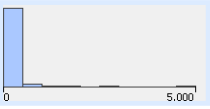
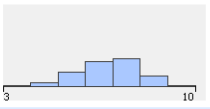
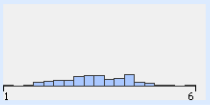
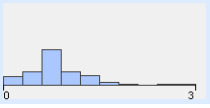
Column	Min	Mean	Median	Max	Std. Dev.	Skewness	Kurtosis	No. Missing	No. +∞	No. -∞	Histogram
Age	4	44,7461	?	90	16,1898	-0,0294	-0,5601	0	0	0	
birth_year	1 933	1 978,2539	?	2 019	16,1898	0,0294	-0,5601	0	0	0	
birth_month	1	5,9091	?	11	2,9005	0,0533	-1,1478	0	0	0	
TB	0,4	3,2988	?	75	6,2095	4,9075	37,1638	0	0	0	
DB	0,1	1,4861	?	19,7	2,8085	3,2124	11,3525	0	0	0	
Alkphos	63	290,5763	?	2 110	242,938	3,7651	17,7528	0	0	0	
Sgpt	10	80,7136	?	2 000	182,6204	6,5492	50,5794	0	0	0	
Sgot	10	109,9108	?	4 929	288,9185	10,5462	150,9199	0	0	0	
TB (#1)	2,7	6,4832	?	9,6	1,0855	-0,2857	0,233	0	0	0	
ALB	0,9	3,1419	?	5,5	0,7955	-0,0437	-0,3879	0	0	0	
CHOL	0.0	0.0	?	0.0	0.0	0.0	0.0	15	0	0	
AG_Ratio	0,3	0,9491	?	2,8	0,3201	1,0043	3,2785	4	0	0	
BILmg	0,0234	0,1911	?	4,386	0,3581	5,0668	40,1222	22	0	0	

Figure 2: Visão do nodo de “Statistics”

No entanto, foram encontradas algumas inconsistências nos dados observados. Principalmente na coluna referente ao colesterol, em que todos os valores estavam a 0 e na *birth_date* todos os valores referentes ao dia e mês estavam também a 0. Mais a frente no relatório iremos referir como abordamos estas duas colunas.

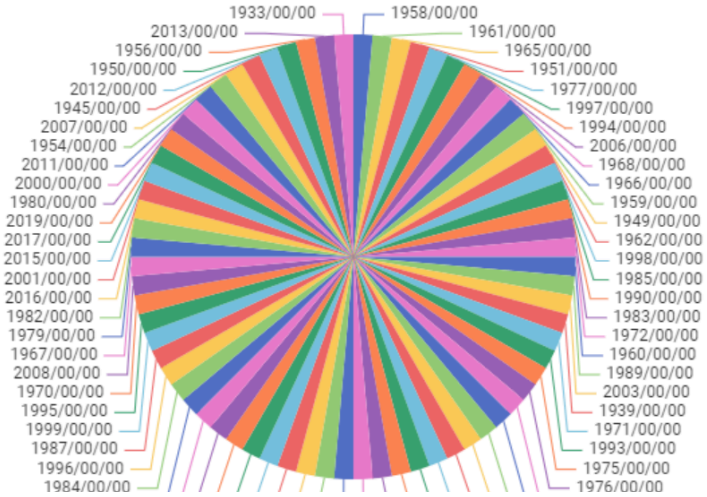


Figure 3: Distribuição dos valores da *birth_date*

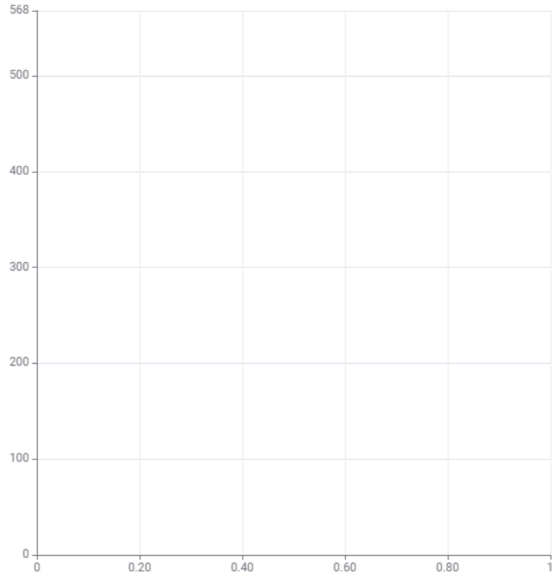


Figure 4: Distribuição dos valores de colesterol

Foi também observado inconsistências nos valores do género e do *selector*, ambos possuíam valores nominais diferentes mas que representavam a mesma informação, como se pode observar em baixo:

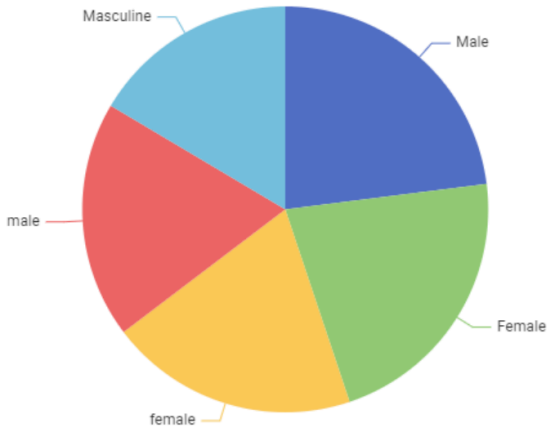


Figure 5: Distribuição dos valores do género

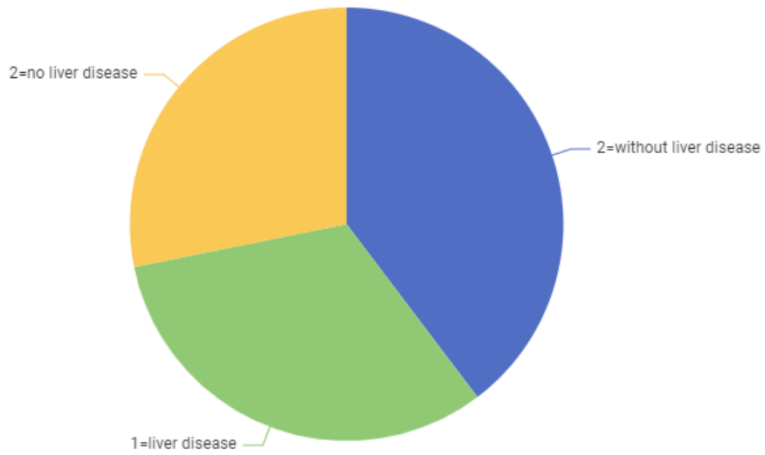


Figure 6: Distribuição dos valores da coluna *selector*

3.3. Data Preparation

3.3.1. Normalização dos valores e colunas filtradas

Para tratar dos problemas acima referidos, inicialmente foram filtradas as colunas do colesterol e da *birth_date* com recurso ao nodo **column filter**, pois ambas não traziam informações novas nem relevantes para o estudo do *dataset*, sendo que mesmo a *birth_date* possuindo a data de nascimento na sua informação, a mesma também já se encontrava na coluna *birth_year*

Foram também usados três nodos **rule engine** para tratar da discrepância dos dados do género e do *selector*, garantindo a uniformidade dos dados e facilitando análises futuras. Vale realçar que ambos os valores possíveis da coluna género (*Male* ou *Female*) foram passados para binários (1 e 0, respetivamente) utilizando a técnica *One-Hot Encoding*, uma técnica regularmente usada regularmente utilizada em aprendizagem de máquina para converter variáveis categóricas em representações numéricas adequadas para modelos de *machine learning*. Esta foi uma técnica essencial ao trabalho, pois após de ter sido implementada os resultados, em geral, melhoraram significativamente.

Já os da coluna *selector* (1=liver disease ou 2=no liver disease) foram tornados em booleanos que pertencerão a uma nova coluna *IsSick*. O que levou a que a coluna *selector* fosse, também, filtrada.

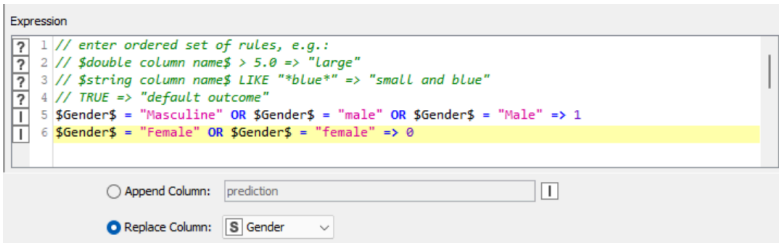


Figure 7: Uniformização da coluna género e após passado para binário

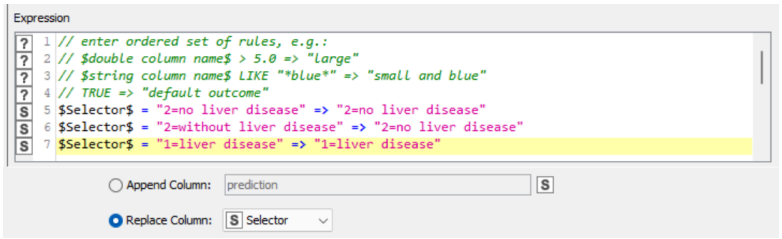


Figure 8: Uniformização do atributo *selector*

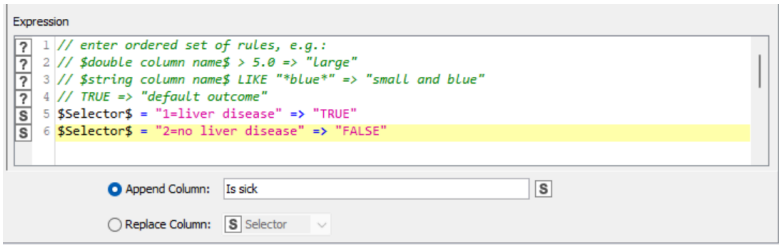


Figure 9: Transformação dos resultados da coluna *selector* para booleanos

3.3.2. Renomeamento de Colunas

De forma a melhor entendimento e facilidade de compreensão dos resultados de análise assim como o seu tratamento, com o auxilio de um nodo **column renamer** foram renomeadas as colunas para nomes mais facilmente compreensíveis.

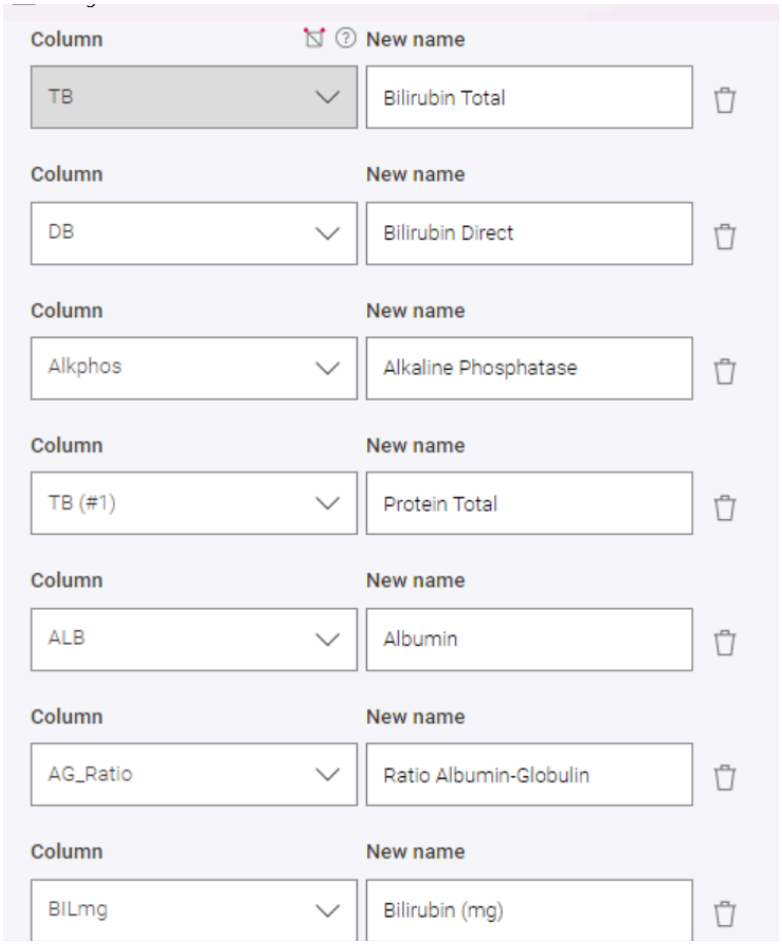


Figure 10: Excerto do nodo *column renamer*

3.3.3. Seleção de atributos a partir de correlação linear

Após isto de forma a compreender as relações entre os atributos, foi utilizado o nó **Linear Correlation**, obtendo assim a matriz de correlação.

A partir dessa análise, identificamos que alguns atributos apresentavam correlações muito fortes, indicadas por valores próximos de 1. Como pode ser observado a seguir.

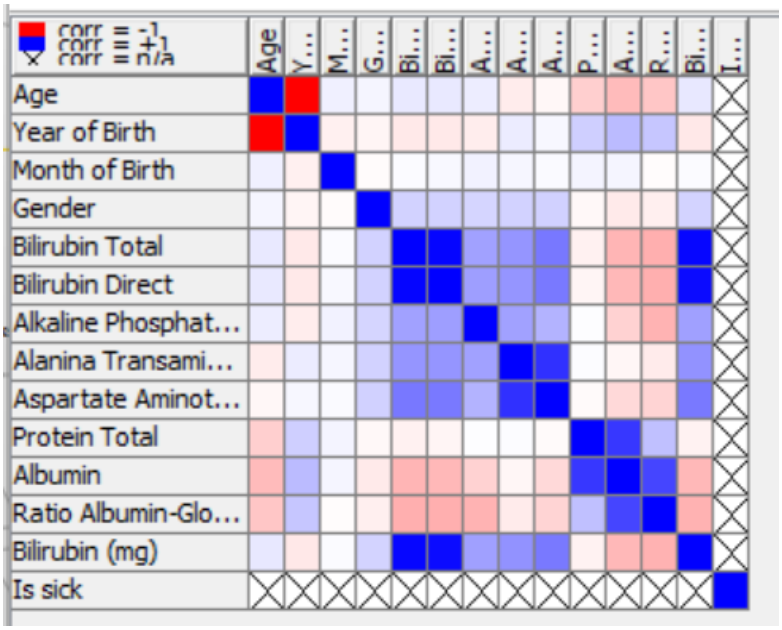


Figure 11: Matriz de correlação antes do tratamento

Após comparação de resultados, decidimos remover quatro colunas do dataset. As colunas *Birulin Direct* e *Birulin(mg)* foram eliminadas devido à alta correlação do atributo *Birulin Total* e entre si!

Essa seleção de atributos permitiu simplificar a estrutura do *dataset* e eliminar redundâncias, garantindo que os atributos remanescentes fornecessem informações úteis e independentes para análises e modelagens futuras.

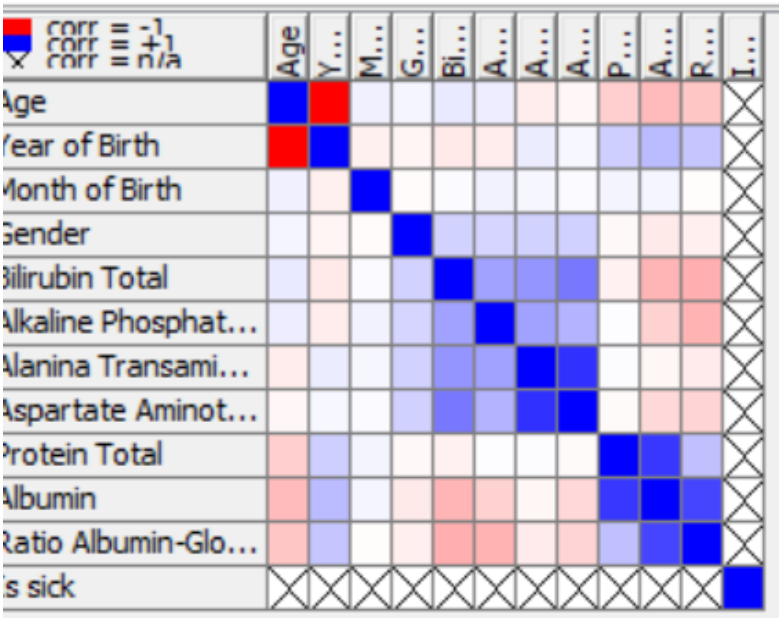


Figure 12: Matriz de correlação depois do tratamento

Vale realçar que apesar dos atributos *Alanina Transaminase* e *Aspartate Aminotransfarese*, assim como, *Protein Total* e *Albumin*, terem também altas correlações entre si, não são suficientemente grandes para ser justificado a sua filtragem.

Assim como, o atributo *Age* e *Year of Birth*,que inicialmente tinha, também sido retirado, mas que foi mantido pois após testes, os resultados eram ligeiramente melhores quando a coluna *Year of Birth* era mantido.

3.3.4. Tratamento de outliers e missing values

Por último mas não menos importante, com auxilio de nodos de exploração e visão como **box plot** , como foi dito anteriormente, foram verificados a existência de grandes valores de outliers e alguns missing values.

No caso dos, utilizamos o nó **Numeric Outliers**, que nos permitiu substituir os seus valores utilizando a técnica de substituição permitida pelo método “Closest Permitted Value”.

Esta decisão de substituir os outliers em vez de remove-los foi tomada após obter melhores resultados em vários testes e análises que vão ser explicadas mais à frente no relatório. Isto deve-se ao facto do seu elevado número, sendo que a sua remoção resultava numa perda significativa de informação, afetando negativamente os modelos desenvolvidos.

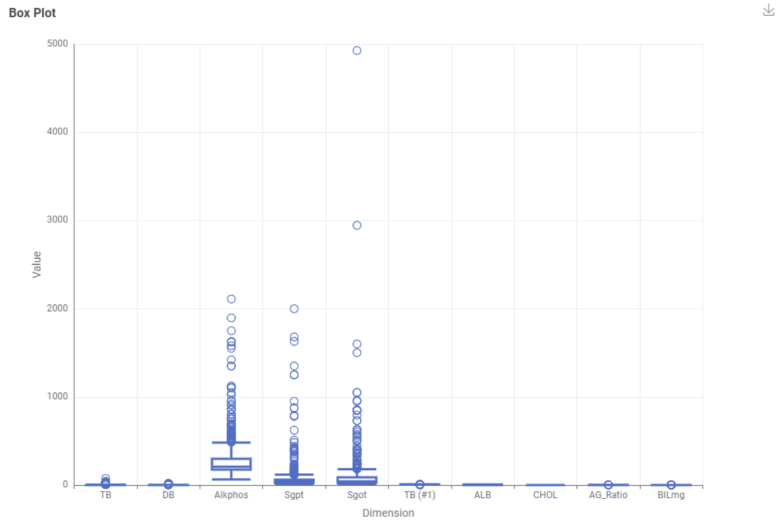


Figure 13: Visão de um **box plot** antes do tratamento

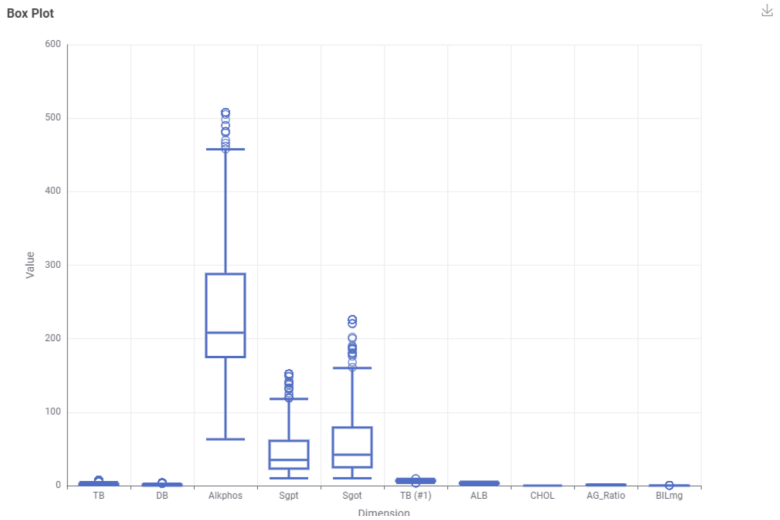


Figure 14: Visão de um **box plot** depois do tratamento

Quanto aos missing values, foi usado o nodo **Missing Valeus** e em geral, foi aplicada a estratégia de trocar os inteiros e strings para o Valor Mais Frequente e os doubles para a Mediana. Isto deve se ao mesmo motivo explicado em cima, pois foram obtidos os melhores valores nos modelos com esta estratégia.

3.4. Modeling e Evaluation

É importante denotar que durante a fase de desenvolvimento do modelo foram aplicadas várias estratégias diferentes a cada modelo de forma a obter e a poder comparar resultados, sendo essas:

- Utilizar os nodos **X-Partitioner** e **X-Aggregator**
- Utilizar o nodo **Partitioning**
- Utilizar o nodo **SMOTE** para *data augmentation*
- Utilizar o nodo **Normalizer** de forma a aplicar a normalização dos dados.

Mais a frente essas estratégias e as suas diferenças serão explicadas em mais detalhes.

Foi também utilizada a mesma *random seed* em todos os divisores nos modelos de forma a tentar manter a consistência dos resultados.

3.4.1. Decision Tree

Decision Tree é um dos modelos mais simples e recorrentes na criação de modelos de avaliação. A partir de decisões hierárquicas e intuitivos, ela divide o conjunto de dados em subgrupos durante o processo de aprendizagem. Após observar os dados e as suas características, separa as instâncias em diferentes categorias de destino.

Fora testadas diversas configurações, desde alteração do *prunning* ao aumento do número de threads, mas consequentemente, todos esses testes levaram a piores resultados, sendo mantidas assim as configurações iniciais do nodo.

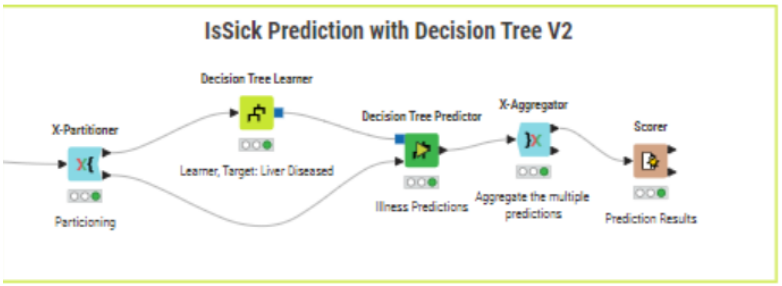


Figure 15: Modelo Decision Tree com X-Partitioner e X-Aggregator

3.4.1.1. Resultados Obtidos

Resultados obtidos com Missing Values serem tratados por Valor Mais Frequente para inteiros e strings e Madiana para os doubles. E Outliers com o valor mais perto permitido:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	81,4%	0,603%
X-Partitioner sem SMOTE	79,81%	0,554%
Partitioning com SMOTE	79,452%	0,587%
X-Partitioner com SMOTE	75,3%	0,569%

Resultados obtidos com Missing Values e Outliers a serem removidos:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	68,124%	0,323%
X-Partitioner sem SMOTE	66,11%	0,294%
Partitioning com SMOTE	67,142%	0,277%
X-Partitioner com SMOTE	63,13%	0,209%

3.4.2. Logistic Regression

Logistic Regression é um modelo estatístico que é usado para prever a probabilidade de ocorrência de um evento binário. Este modelo vai ajustando os coeficientes de cada variável independente para maximizar a probabilidade de ocorrência do evento a ser estudado.

Em geral este modelo foi o que obteve pior resultados, mesmo após alterar o numero de *epochs*.

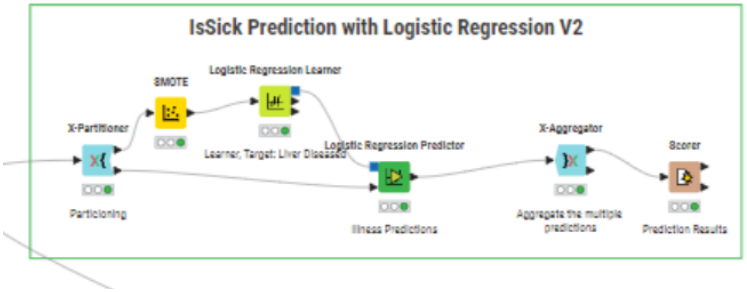


Figure 16: Exemplo de um Modelo Logistic Regression

3.4.2.1. Resultados Obtidos

Resultados obtidos com Missing Values serem tratados por Valor Mais Frequente para inteiros e strings e Madiana para os doubles. E Outliers com o valor mais perto permitido:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	82,192%	0,502%
X-Partitioner sem SMOTE	78,559%	0,394%
Partitioning com SMOTE	74,678%	0,441%
X-Partitioner com SMOTE	75,81%	0,461%

Resultados obtidos com Missing Values e Outliers a serem removidos:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	74,124%	0,533%
X-Partitioner sem SMOTE	71,131%	0,51%
Partitioning com SMOTE	71,422%	0,477%
X-Partitioner com SMOTE	66,123%	0,311%

3.4.3. Gradient Boosted Trees

Gradient Boosted Trees é uma técnica de aprendizagem de máquina baseada em *ensemble* em que diversas árvores de decisão são treinadas sequencialmente. Cada árvore tenta corrigir os os resultados errados da anterior, obtendo uma previsão final combinando as previsões individuais de todas as árvores.

Devido à falta de total compreensão de como funciona o nó e de tempo, não foi possível alterar e testar as diferentes definições.

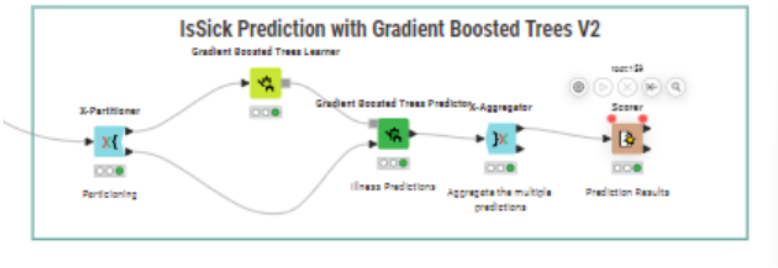


Figure 17: Exemplo de um Modelo Gradient Boosted Trees

3.4.3.1. Resultados Obtidos

Resultados obtidos com Missing Values serem tratados por Valor Mais Frequente para inteiros e strings e Madiana para os doubles. E Outliers com o valor mais perto permitido:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	90,024%	0,8%
X-Partitioner sem SMOTE	86,976%	0,71%
Partitioning com SMOTE	84,972%	0,616%
X-Partitioner com SMOTE	82,612%	0,574%

Resultados obtidos com Missing Values e Outliers a serem removidos:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	81,14%	0,533%
X-Partitioner sem SMOTE	77,1%	0,484%
Partitioning com SMOTE	79,151%	0,573%
X-Partitioner com SMOTE	75,13%	0,459%

3.4.4. Tree Ensemble

Tree Ensemble, à semelhança do anterior, é uma técnica que combina várias árvores de decisão individuais num único modelo.

Múltiplas árvores de decisão são treinadas com diferentes subconjuntos de dados, e as suas previsões são combinadas para obter uma previsão final.

Neste algoritmo foram aumentados o número de modelos criados assim como o *split criterion*, obtendo o resultado mais satisfatório quando criados 100 modelos com *Information Gain Ratio*.

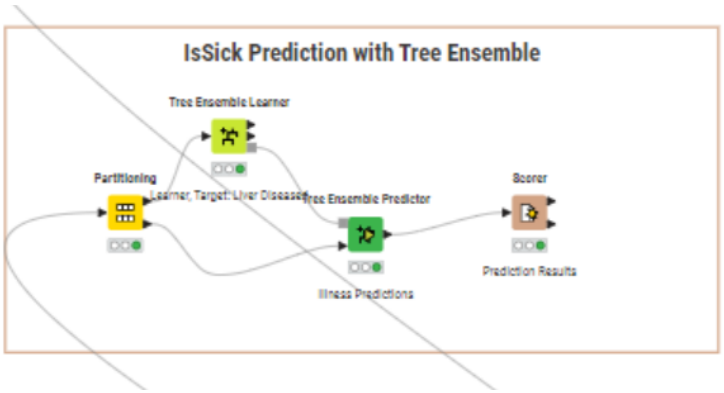


Figure 18: Exemplo de um Modelo Tree Ensemble

3.4.4.1. Resultados Obtidos

Resultados obtidos com Missing Values serem tratados por Valor Mais Frequente para inteiros e strings e Mediana para os doubles. E Outliers com o valor mais perto permitido:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	91,16%	0,83%
X-Partitioner sem SMOTE	87,31%	0,68%
Partitioning com SMOTE	82,877%	0,591%
X-Partitioner com SMOTE	81,818%	0,563%

Resultados obtidos com Missing Values e Outliers a serem removidos:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	83,4%	0,64%
X-Partitioner sem SMOTE	78,12%	0,524%
Partitioning com SMOTE	77,1%	0,507%
X-Partitioner com SMOTE	75,44%	0,477%

3.4.5. Random Forest

Random Forest é um modelo de aprendizagem de máquina baseado, também, em *ensemble* como os anteriores. Seguindo o método de treinamento do *Tree Ensemble*.

Após mexer com as definições do nodo como *split criterion* e número de modelos, os valores selecionados foram *Information Gain Ratio* com 150 modelos.

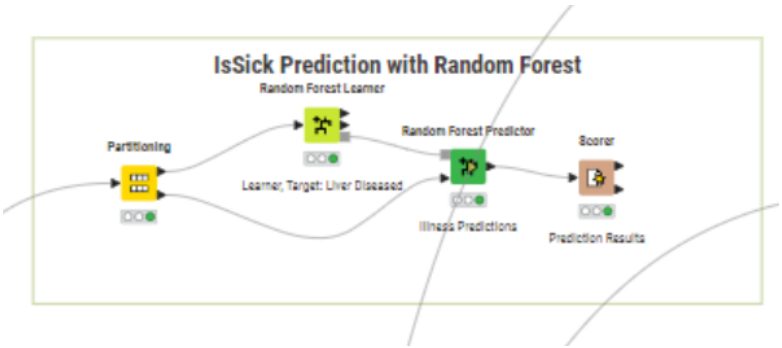


Figure 19: Exemplo de um Modelo Random Forest

3.4.5.1. Resultados Obtidos

Resultados obtidos com Missing Values serem tratados por Valor Mais Frequente para inteiros e strings e Madiana para os doubles. E Outliers com o valor mais perto permitido:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	88,741%	0,662%
X-Partitioner sem SMOTE	84,274%	0,583%
Partitioning com SMOTE	83,562%	0,577%
X-Partitioner com SMOTE	82,161%	0,582%

Resultados obtidos com Missing Values e Outliers a serem removidos:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	86,4%	0,641%
X-Partitioner sem SMOTE	81,8%	0,621%
Partitioning com SMOTE	80,02%	0,498%
X-Partitioner com SMOTE	78,163%	0,409%

3.4.6. Cluster (k-means)

O método de *clustering*, ou agrupamento, é uma técnica de aprendizado de máquina não supervisionado que visa identificar padrões naturais nos dados, agrupando-os em conjuntos ou *"clusters"* com base em suas características semelhantes. O algoritmo **k-means** é uma das abordagens mais comuns para realizar o *clustering*.

Foram utilizados apenas dois *clusters* devido a só haver dois valores possíveis para o *target*.

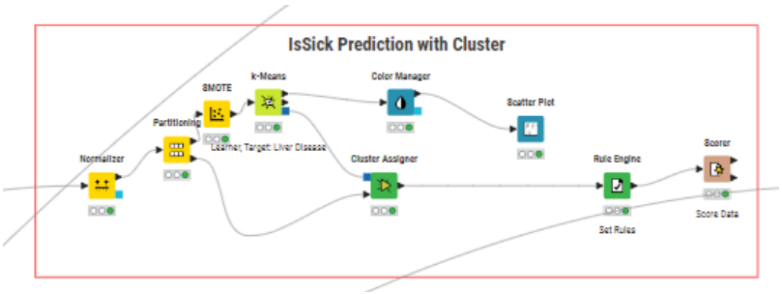


Figure 20: Exemplo de um Modelo K-means

3.4.6.1. Resultados Obtidos

Resultados obtidos com Missing Values serem tratados por Valor Mais Frequente para inteiros e strings e Madiana para os doubles. E Outliers com o valor mais perto permitido:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	38,42%	0,102%
X-Partitioner sem SMOTE	55,981%	0,184%
Partitioning com SMOTE	36,986%	0,355%
X-Partitioner com SMOTE	57,804%	0,077%

Resultados obtidos com Missing Values e Outliers a serem removidos:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	20,012%	−0,277%
X-Partitioner sem SMOTE	22,841%	0,034%
Partitioning com SMOTE	38,152%	0,207%
X-Partitioner com SMOTE	33,31%	0,109%

3.4.7. Naive Bayes

Naive Bayes é um modelo de aprendizagem de máquina baseado no teorema de *Bayes*, que assume independência condicional entre as características.

O modelo calcula a probabilidade de uma instância pertencer a uma determinada classe com base nas probabilidades condicionais das características dadas.

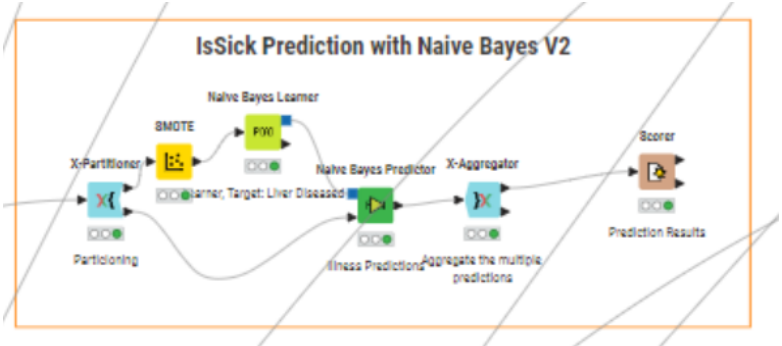


Figure 21: Exemplo de um Modelo Naive Bayes

3.4.7.1. Resultados Obtidos

Resultados obtidos com Missing Values serem tratados por Valor Mais Frequente para inteiros e strings e Madiana para os doubles. E Outliers com o valor mais perto permitido:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	79,452%	0,363%
X-Partitioner sem SMOTE	79,76%	0,375%
Partitioning com SMOTE	80,137%	0,411%
X-Partitioner com SMOTE	80,617%	0,421%

Resultados obtidos com Missing Values e Outliers a serem removidos:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	59,14%	0,213%
X-Partitioner sem SMOTE	53,127%	0,103%
Partitioning com SMOTE	55,2%	0,167%
X-Partitioner com SMOTE	53,12%	0,06%

3.4.8. RProp MLP

RProp MLP, ou *MultiLayer Perceptron*, é um modelo de rede neuronal artificial baseado no algoritmo de retro propagação com atualização de pesos resilientes.

Este algoritmo ajusta os pesos das conexões entre neurónios para minimizar a diferença entre as previsões do modelo e os valores reais dos dados de treinamento.

Neste modelo foi aumentado o número de *hidden layers* para 2, o número de *hidden neurons per layer* 95 e o número de iterações para 200, isto devido ao facto que foi com estas configurações que se encontrou maior *accuracy* e precisão.

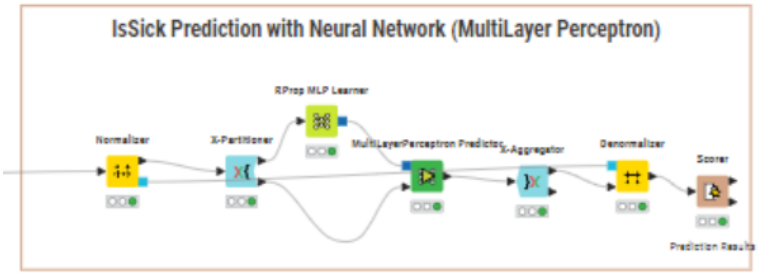


Figure 22: Exemplo de um Modelo RProp MLP

3.4.8.1. Resultados Obtidos

Resultados obtidos com Missing Values serem tratados por Valor Mais Frequente para inteiros e strings e Madiana para os doubles. E Outliers com o valor mais perto permitido:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	76,712%	0,379%
X-Partitioner sem SMOTE	75,986%	0,395%
Partitioning com SMOTE	82,192%	0,593%
X-Partitioner com SMOTE	74,099%	0,4%

Resultados obtidos com Missing Values e Outliers a serem removidos:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	69,142%	0,227%
X-Partitioner sem SMOTE	64,181%	0,027%
Partitioning com SMOTE	48,192%	0,037%
X-Partitioner com SMOTE	43,13%	0,033%

3.4.9. DL4J (Deep Learning for Java)

Os modelos criados com DL4J empregam algoritmos sofisticados de otimização, como o gradiente descendente estocástico (SGD), para ajustar os pesos das conexões entre as unidades neuronais.

Esse ajuste é realizado de maneira iterativa, visando a minimização de uma função de perda que mensura a discrepância entre as previsões do modelo e os valores reais dos dados de treinamento.

É importante realçar, para que seja possível a utilização deste tipo de modelos é necessária a criação de um initializer da rede neural, o mesmo foi composto por 3 *layers* sendo que cada um deles possui 3 neurónios e usa o método ReLU.

Vale também denotar, que devido ao elevado tempo que cada modelo demora a ser constituído foram mantidas as pré-definições dadas nas aulas teórico-práticas, aumentando apenas o número de iterações por 120. O mesmo não foi aumentado mais pois demorava cada vez mais tempo e a melhora dos resultados era quase insignificativa.

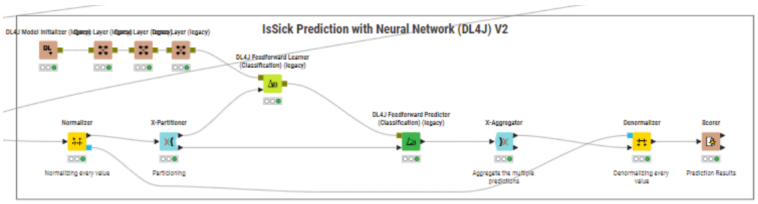


Figure 23: Exemplo de um Modelo DL4J

3.4.9.1. Resultados Obtidos

Resultados obtidos com Missing Values serem tratados por Valor Mais Frequente para inteiros e strings e Madiana para os doubles. E Outliers com o valor mais perto permitido:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	43,152%	0,127%
X-Partitioner sem SMOTE	37,05%	0,071%
Partitioning com SMOTE	69,863%	0,027%
X-Partitioner com SMOTE	72,556%	0,306%

Resultados obtidos com Missing Values e Outliers a serem removidos:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	40,124%	−0,126%
X-Partitioner sem SMOTE	30,181%	0,03%
Partitioning com SMOTE	42,112%	0,167%
X-Partitioner com SMOTE	42,229%	0,219%

3.4.10. PNN (Probabilistic Neural Network)

O método PNN (Probabilistic Neural Network), ou Rede Neural Probabilística, é uma técnica de aprendizagem supervisionada que pertence à categoria das redes neurais. Ela é conhecida pela capacidade de modelar distribuições de probabilidade dos dados de entrada para realizar classificação.

Na PNN, os dados de entrada são atribuídos a diferentes camadas de neurônios. Cada camada corresponde a uma classe ou categoria possível de saída. Durante o treinamento, a rede calcula a probabilidade de que um dado de entrada pertença a cada classe com base em sua semelhança com os exemplos de treinamento previamente apresentados.

A principal vantagem da PNN é sua capacidade de modelar distribuições de probabilidade complexas e lidar bem com problemas de classificação.

Este modelo foi também testado após aconselhamento de um dos docentes da cadeira. Devido à falta de tempo não foram testadas as configurações extras deste modelo.

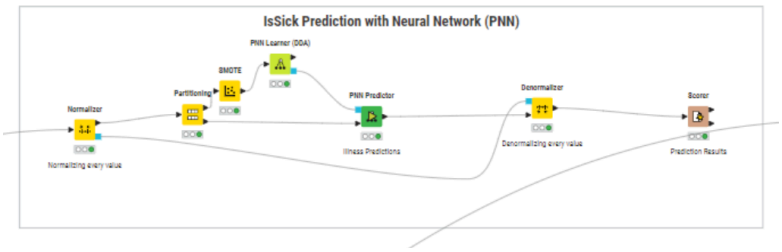


Figure 24: Exemplo de um Modelo PNN

3.4.10.1. Resultados Obtidos

Resultados obtidos com Missing Values serem tratados por Valor Mais Frequente para inteiros e strings e Madiana para os doubles. E Outliers com o valor mais perto permitido:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	80,137%	0,39%
X-Partitioner sem SMOTE	79,931%	0,0379%
Partitioning com SMOTE	71,233%	0,4%
X-Partitioner com SMOTE	70,84%	0,399%

Resultados obtidos com Missing Values e Outliers a serem removidos:

Estratégia	Accuracy	Cohen's kappa
Partitioning sem SMOTE	72,12%	−0,678%
X-Partitioner sem SMOTE	70,181%	0,244%
Partitioning com SMOTE	63,368%	0,176%
X-Partitioner com SMOTE	66,183%	0,205%

3.5. Conclusões e anotações extras

Após uma minuciosa revisão dos resultados derivados do treinamento dos modelos de aprendizagem, visando antecipar a ocorrência ou não de doenças hepáticas, torna-se patente que a estratégia empregada revelou-se promissora. Com tempo extra e modelos mais complexos poderia-se ter obtido valores bastantes altos e precisos. Apesar disso os variados modelos experimentados forneceram resultados preciosos acerca da interconexão entre os atributos demográficos e a variável alvo, contribuindo para uma apreciação mais aprofundada da questão em análise.

Em geral podemos observar que os Modelos que seguem a estratégia de Ensemble tem valores mais elevados. Este tipo de modelos apresentaram métricas de avaliação robustas, como alta precisão, *accuracy* e *kappa de Cohen*, sugerindo uma boa capacidade de generalização e adequação aos dados.

Podemos tirar também essa conclusão pois com o auxilio de um **scatter plot matrix** observamos que os valores são muito conglomerados e há pouca linearidade.

Foi usado um **SMOTE** (Synthetic Minority Over-sampling Technique) pois os dados eram ligeiramente desbalanceados, esta técnica consite e é usada quando uma classe é significativamente mais presente do que outra. Ele funciona sintetizando exemplos da classe minoritária, em vez de simplesmente replicá-los, como fazem abordagens de *oversampling* simples. Mas inicialmente o nodo era aplicado antes dos *partitionings* e levou a resultados com *accuracy* bastante elevados (acima de 93%), mas esses resultados, apesar de altos, não são bons nem confiáveis, pois o **SMOTE** desvirtua os testes. Foi então corrigido para ser aplicado após a partição dos dados, especificamente nos dados que são usados para treino, o que nos levou a compreender que em geral, tirando em redes neuronais, os resultados pioram quando usado o **SMOTE**. Quando usado em modelos de redes neuronais, conseguimos obter resultados ligeiramente melhores.

Podemos também observar que os resultados em modelos aplicados com **X-Partitioned** e **X-Aggregator**, em geral, tinham resultados ligeiramente piores, mas devido à sua alta complexidade, e há forma como agrupa os dados quando são divididos em subconjuntos, podemos afirmar que apesar dos "piores" resultados, eles estão mais aptos a receber novos dados enquanto com o **partitioning** está bastante "preso" aos dados atuais que possui.

Foram feitos vários testes com diferentes soluções para o tratamento de missing values e outliers,(como só tratar de outliers acima do limite superior ou inferior, ou trocar os missing values para média, entre outros) mas não foram especificados no relatório devido ao limite de páginas do mesmo!

4. Dataset escolhido pelo grupo

4.1. Business Understanding

O dataset escolhido pelo grupo apresenta um conjunto de atributos relativos aos funcionários de uma fábrica de vestuário, na tentativa de perceber a relação desses mesmos atributos com a produtividade dos funcionários.

O dataset é então composto pelos seguintes atributos:

- **date**: Data formatada seguindo o formato MM-DD-YYYY;
- **day**: Dia da semana;
- **quarter**: Parte do mês. Um mês é dividido em 4 quarters;
- **department**: Departamento associado á instância;
- **team_no**: Número da equipa a associado à instância;
- **no_of_workers**: Número de funcionários da equipa;
- **no_of_style_change**: Número de mudanças no estilo de um produto específico.
- **targeted_productivity**: Produtividade alvo definida pela Autoridade para cada equipe em cada dia;
- **smv**: Standard Minute Value, tempo alocado para uma tarefa;
- **wip**: Work in progress. Inclui o número de itens inacabados por produto;
- **over_time**: Representa o overtime por cada equipa em minuto;
- **incentive**: Representa um incentivo financeiro (em BDT);
- **idle_time**: Tempo que a produção foi interrompida, causada por diversos fatores;
- **idle_men**: O número de trabalhadores que estavam inativos devido à interrupção da produção;
- **actual_productivity**: Produtividade atual em percentagem dos trabalhadores. Este valor localiza-se entre 0-1.

4.2. Data Understanding

Para conseguir perceber melhor os dados do problema, utilizaram-se uma série de nodos. Como o nó **Data Explorer** conseguimos perceber que o atributo wip continha missing values. Além disso, percebemos que o atributo quarter por vezes continha o valor *quarter5*, valor esse inválido no contexto do problema. Com o uso do nodo **Estatistics**, foi possível perceber que algumas colunas, como o idle_time, o idle_men e o over_time não acrescentavam informações ao problema pois os seus valores eram praticamente sempre zero. Além disso, percebemos que o atributo actual_produtivity por vezes continha valores acima de 1, valores impossíveis no contexto do problema. Com o uso do nodo **Rank Correlation**, foi possível perceber que os dados não têm muita correlação entre si. Usamos ainda o nodo **Box Plot** que nos permitiu identificar alguns valores anómalos, sobretudo nos valores do atributo wip. Analisando ainda com o nodo **Scatter Plot**, conseguimos perceber ainda a existencia de mais valores anómalos, como é o caso do valor do over_time da linha 146, do incentive das linhas 1128,1080, 960,1440,1200,960,1130 e 1133 e o valor do no_of_workers da linha 355. Temos ainda anomalias nas linhas 570,572,568,561,563 e 569 do wip.

Foram ainda abordados alguns nodos que nos permitissem visualizar alguns gráficos de forma a detetarmos mais algumas anomalias.

4.3. Data Preparation

Para realizar a data preparation, usou-se como base o **Linear Regression Learner** e o **Regression Predictor** para que possamos decidir quais as melhores abordagens, relativas ao processo de preparação de dados, que devemos tomar.

Uma vez que os atributos quarter e *actual_produtivity* continham valores inválidos no contexto do problema, a nossa primeira abordagem passou por proceder à remoção das linhas dessas colunas que continham valores inválidos. Para o fazer, usamos dois nodos **Row Filter**.

Depois, decidimos tratar os *outliers* usando o nodo **Numeric Outliers**, passando os valores anormais para o valor mais próximo permitido.

Usamos ainda um **Column Filter** para remover as colunas do *idle_time, idle_men e no_of_style_change*, uma vez que praticamente só continham zeros.

No learner, foram retiradas as colunas *day, date, over_time* e *no_of_workers* uma vez que eram as que apresentavam as correlações mais baixas (menor de 0.1) com o *actual_produtivity*.

Todas este tratamento de dados levou-nos ao nosso primeiro *score*, obtido com o nodo **Numeric Scorer**. Em seguida, realizamos testes para perceber se existiam alternativas de tratamento de dados que pudessem melhorar o nosso *score*. As etapas realizadas seguem no quadro abaixo.

De reforçar que quando as alterações levavam a melhores valores eram mantidas. Caso contrário, a modelações anterior seria recuperada.

Procedimentos	R^2	MAE	MSE	RMSE
Tratamento inicial	0.703	0.047	0,005	0,07
Ao invés de retirar as linhas com *actual_produtivity* acima de 1, passar esses valores para 1.	0.803	0.041	0,004	0,062
Colocar missing values *wip* com o valor mais frequente	0.346	0.094	0,018	0,134
Colocar missing values *wip* com o da mediana	0.346	0.094	0,018	0,134
Colocar missing values *wip* com o da média	0.345	0.095	0,018	0,135
Eliminar as linhas que têm outliers	0.85	0.026	0,001	0,036
Eliminar outliers abaixo do limite inferior	0.808	0.029	0,002	0,047
Eliminar outliers acima do limite superior	0.757	0.039	0,004	0,067

Procedimentos	R^2	MAE	MSE	RMSE
Eliminar linhas com *overtime* inferior a 1800	0.744	0.027	0,002	0,041
Eliminar linhas com *wip* inferior a 400	0.761	0.032	0,002	0,046
Remover do learner colunas com correlação abaixo de 0.15 com o *actua_produtivity*	0.859	0.025	0,001	0,035
Remover do learner colunas com correlação abaixo de 0.2 com o *actua_produtivity*	0.855	0.025	0,001	0,035

Para passarmos os valores da *produtivity* que eram maiores que 1 para 1, usou-se o nodo **JavaSnippet**.

O remover as linhas que continham o *quarter* errado fez com que perdêssemos um número muito significativo de linhas. No entanto, foi o único tratamento que fez com que acabássemos um valor razoável de r^2.

No final, o melhor tratamento de dados passou por remover as linhas que continham missing values no campo *wid*, remover as linhas que tinham o *quarter* como sendo Quarter5, colocar os outliers com o valor mais próximo permitido, colocar os valores de produtividade que eram superiores a 1, a 1 remover as colunas com os valores quase todos nulos e remover do treino as colunas com correlação inferior a 0.15 com o *actual_produtivity*.

4.4. Modeling and Evaluation

Como dito anteriormente, começamos por utilizar **Linear Regression Learner** e o **Regression Predictor** para realizar a modelação. Realizou-se a partição dos dados recorrendo ao nodo **Partitioning**, definindo-se que 75% dos dados serviriam de treino e os restantes para teste. Definiu-se ainda que quando fosse necessário utilizar uma *seed* que esta seria 2024, de forma a podermos comparar os deferentes modelos. Nesta secção, abordaremos os diferentes nodos de treino que utilizamos de modo a perceber qual o modelo que melhor se adequa ao nosso problema. Assim, o seguinte quadro serve de síntese ao trabalho realizado.

Nodos	R^2	MAE	MSE	RMSE
• Linear Regression • Linear Predictor	0.859	0.025	0,001	0,035
• Random Forest Learner • Random Forest Predictor	0.842	0.021	0,001	0,037
• Gradient Boosted Trees Learner • Gradient Boosted Trees Predictor	0.859	0.019	0,001	0,037

Nodos	R^2	MAE	MSE	RMSE
• Linear Regression • Linear Predictor + Normalização dos valores entre 0 e 1	0.735	0.031	0,001	0,037
• Linear Regression • Linear Predictor + Normalização dos valores entre o maximo e o mínimo	0.711	0.027	0,001	0,036
• Gradient Boosted Trees Learner • Gradient Boosted Trees Predictor + Normalização dos valores entre 0 e 1	0.742	0.021	0,001	0,034
• RProp MLP Learner • MultiLayerPerceptores Predictor	0.846	0.025	0,001	0,036
• Gradient Boosted Trees Learner • Gradient Boosted Trees Predictor + Patição dos dados 70%-30%	0.846	0.02	0,001	0,036
• Gradient Boosted Trees Learner • Gradient Boosted Trees Predictor + Patição dos dados 80%-20%	0.774	0.018	0,002	0,039
• Gradient Boosted Trees Learner • Gradient Boosted Trees Predictor + *seed* com valor −4783817209010111339	0.878	0.015	0,001	0,03

É importante notar que sempre que mexemos com um modelo novo, tentamos ajustar os vários parâmetros possíveis para que os *scores* obtidos sejam o melhor possível. Assim, o melhor resultado obtido foi usando o **Gradient Boosted Trees Learner** e o **Gradient Boosted Trees Predictor** com uma partição dos dados de 30% para teste e 70% para treino e com uma *seed* de −4783817209010111339 .

Nodos	R^2	MAE	MSE	RMSE
• Gradient Boosted Trees Learner • Gradient Boosted Trees Predictor + *seed* com valor −4783817209010111339 + partição 70% treino e 30% teste	0.878	0.015	0,001	0,03

4.4.1. Transformar um problema de regressão em classificação

Achamos que seria pertinente para o estudo do nosso dataset tranforma-lo num problema de classificação. Isto deve-se ao facto de ser interessantes ter divisões entre as diferentes percentagens de produtividade dos funcionários para a realização de estatísticas associadas à fábrica. Para o fazer, recorremos ao nodo **Auto-Binner**. Seguimos duas abordagens diferentes na utilização deste nodo, descritas em seguida:

Forma	Accuracy	Error
Intervalos espaçados 0.2	0.859	0.025
4 intervalos delimitados pelo maior e menor valor existente de *actual_produtivity*	63.636%	36.364%

4.5. Conclusões

Após uma minuciosa avaliação dos desempenhos dos modelos de regressão, estamos bastante satisfeitos com os resultados alcançados. Os diferentes modelos testados forneceram resultados valiosos sobre a conexão entre os atributos demográficos e o *target*, enriquecendo nossa compreensão de como os diferentes atributos impactam a produtividade dos trabalhadores.

O processo de pré-processamento dos dados, que englobou técnicas como o tratamento de *outliers* e *missing values*, desempenhou um papel fundamental na otimização da qualidade dos modelos. Podendo observar que quando os alteramos, os resultados variam bastante. Essas abordagens auxiliaram na mitigação do impacto de dados discrepantes, resultando em uma maior capacidade de generalização dos modelos.

Além disso, a eliminação de colunas com altos valores de correlação e a normalização dos dados foi algo essencial e que nos trouxe resultados bem mais precisos.

5. Conclusão Final

Após a finalização deste projeto, sentimos que nosso conhecimento em Aprendizado de Máquina foi ampliado. Durante a elaboração e treinamento dos modelos em ambos os conjuntos de dados, adquirimos uma compreensão mais abrangente das técnicas de análise de dados, pré-processamento e construção de modelos, assim como os seus algoritmos e como as diferentes configurações e infinitas possibilidades alteravam o resultado. Sentimos que os nossos esforços de fazer os melhores resultados possíveis deram frutos.

Além disso, ao longo do projeto, reconhecemos a importância do pré-processamento de dados na melhoria da qualidade dos modelos. O tratamento de valores discrepantes, a normalização de variáveis e outras técnicas contribuíram significativamente para a precisão e generalização dos modelos desenvolvidos. E abriu-nos a visão das infinitas técnicas que podem ser usadas e tudo o que está envolvido no melhoramento de modelos de aprendizagem.

Em uma análise global, temos convicção de que nosso trabalho foi bem executado e que os resultados obtidos refletem isso. Entretanto, reconhecemos que sempre há espaço para melhorias. Algumas das melhorias potenciais incluem a exploração de técnicas mais avançadas de seleção de atributos para identificar as variáveis mais impactantes, e a criação de novas *features* a partir das já existentes nos conjuntos de dados. Estas medidas poderiam aprimorar ainda mais o desempenho dos modelos, assim como muitas outras variáveis.

6. Avaliação Pares

Gonçalo - 1 Ema - 1 Marta - 1 Henrique - 1