Universidade do Minho

Departamento de Informática

Aprendizagem e Decisão inteligentes

Conceção de modelos de aprendizagem.

Grupo 34




Bento João Concieiro Guimarães A96296
Gonçalo Araújo Brandão A100663
Maya Gomes A100822
Luís de Castro Rodrigues Caetano A100893

3 de maio de 2024

1 Introdução

O relatório deste projeto envolve a análise e extração de informações dos dados fornecidos, assim como o desenvolvimento de modelos de aprendizagem. Este trabalho está dividido em duas fases distintas, sendo a primeira a fase sobre o *Dataset Atribuído* e a segunda sobre o *Dataset Grupo* escolhido pelo grupo. O objetivo deste documento é fornecer suporte ao trabalho realizado, explicando as decisões tomadas ao longo do projeto. Sendo assim, o presente relatório apresenta uma análise detalhada e justificada de todo o procedimento efetuado em cada um dos *datasets*.

2 Metodologia escolhida

É importante evidenciar que a metodologia escolhida e seguida ao longo do projeto foi a metodologia CRISP-DM. Esta metodologia divide-se em 6 fases, como está representado na figura 1.

Neste trabalho, não iremos realizar a fase de *deployment*, pois não se aplica ao nosso contexto. Desta forma, iremos:

- Perceber o contexto;
- Perceber os dados;
- Preparar os dados;
- Modelar;
- Avaliar

Figura 1: Metodologia CRIPS-DM.

A metodologia *Cross-Industry Standard Process for Data Mining* é amplamente reconhecida como sendo uma *framework* robusta para projetos de análise de dados e processamento de conjuntos de dados. Para além disso, esta metodologia foi atrativa para nós, pois possui uma estrutura clara e bem definida para cada etapa do processo de análise de dados. Isto permite manter o projeto organizado e orientado.

3 Dataset Atribuído

3.1 Perceber o contexto Atribuído (Buisness Understanding)

O dataset proposto visa a previsão da presença ou não de uma patologia no fígado. Assim, fomo percebendo um pouco esta área analizando certos termos como o de *bilirubin* que se refere a um pigmento amarelo produzido naturalmente pelo corpo sendo normalmente usado para analisar o funcionamento do fígado. Para além disso, a *Alkaline Phosphotase* ou ainda a *Alamine Aminotransferase* que representam exames de sangue que ajudam a identificar doenças do fígado. Desta forma, depois de entendermos melhor o contexto avançamos para o *Data Understanding*.

3.2 Perceber o Dataset Atribuído (Data Understanding)

3.2.1 Descrição do Dataset

O dataset proposto pela equipa docente é um dataset de classificação. Com este pretende-se prever a presença ou não de doenças no fígado (*liver disease* ou *no liver disease*).

Desta forma, trata-se de um problema de classificação binária, pois contem variáveis independentes que serão os recursos usados para prever a presença de doença hepática, o target é se o paciente tem ou não doença hepática sendo esta uma variável categórica e porque pretende-se classificar cada paciente numa das duas classes possíveis com base nas suas características. A classificação binária é apropriada quando estamos lidando com um problema onde existem apenas duas classes distintas (*liver disease* ou *no liver disease*, neste caso).

O dataset consiste em 583 observações e em 17 *features*, sendo 14 delas numéricas e 3 categóricas.

id_code	Age	birth_year	birth_month	birth_date	Gender	TB	DB	Alkphos	Sgpt	Sgot	TB	ALB	CHOL	AG_Ratio	BILmg	Selector
1	65	1958	10	1958/00/00	Female	0,7	0,1	187	16	18	6,8	3,3	0	0,9	0,0409	1=liver disease
2	62	1961	8	1961/00/00	Male	10,9	5,5	699	64	100	7,5	3,2	0	0,7	0,6374	1=liver disease
3	62	1961	7	1961/00/00	Male	7,3	4,1	490	60	68	7,0	3,3	0	0,9	0,4269	1=liver disease
4	58	1965	11	1965/00/00	Male	1,0	0,4	182	14	20	6,8	3,4	0	1,0	0,0585	1=liver disease
5	72	1951	8	1951/00/00	Male	3,9	2,0	195	27	59	7,3	2,4	0	0,4	0,2281	1=liver disease
6	46	1977	8	1977/00/00	Male	1,8	0,7	208	19	14	7,6	4,4	0	1,3	0,1053	1=liver disease
7	26	1997	8	1997/00/00	Female	0,9	0,2	154	16	12	7,0	3,5	0	1,0	0,0526	1=liver disease
8	29	1994	7	1994/00/00	female	0,9	0,3	202	14	11	6,7	3,6	0	1,1	0,0526	1=liver disease
9	17	2006	1	2006/00/00	Male	0,9	0,3	202	22	19	7,4	4,1	0	1,2	0,0526	2=no liver disease
10	55	1968	3	1968/00/00	Male	0,7	0,2	290	53	58	6,8	3,4	0	1,0	0,0409	1=liver disease

Figura 2: Alguns casos do dataset atribuído.

3.2.2 Contextualização do tema do dataset (Data Describing)

As colunas dos nosso dataset são definidas por:

- Age : Idade de uma pessoa.
- Year of birth : O ano em que uma pessoa nasceu.
- Month of birth : O mês em que uma pessoa nasceu.
- Birthday : A data completa do dia de nascimento de uma pessoa.
- Gender : O gênero de uma pessoa.
- Total bilirubin : A quantidade total de bilirrubina presente no sangue.
- Direct Bilirubin : A porção da bilirrubina total que está diretamente conjugada com a glicuronidação no fígado e pronta para ser excretada na bile.
- Alkaline Phosphotase : Quantidade de uma enzima encontrada especialmente no fígado.
- Alamine Aminotransferase : Quantidade de uma enzima encontrada principalmente no fígado.
- Aspartate Aminotransferase : Quantidade de uma enzima presente no fígado.
- Total Proteins : A concentração total de proteínas no sangue.
- Albumin : Quantidade de uma das principais proteínas produzidas pelo fígado.
- Cholesterol : Quantidade de um tipo de lipídio encontrado em todas as células do corpo e na maioria dos alimentos de origem animal.
- Albumin and Globulin Ratio : Relação entre a concentração de albumina e globulinas no sangue.
- Bilirubin (mg/dL) : Quantidade de um pigmento amarelo produzido no fígado.
- Selector (*target variable*) : A presença ou não de doença no fígado.

3.2.3 Leitura: Feature id_code e Decimal Separator

Desde já, notamos que a *feature* "id_code". Esta não faz qualquer sentido já que funciona como um ID, isto é, representa um valor único para cada entrada e por tal, não traz conhecimento para o modelo a treinar futuramente. Desta forma, utilizamos a opção "*Has RowID*"no node *CSV Reader* de forma a ele atribuir somente uma coluna para o id, mas não considerá-la.

Para além disso, foi também necessário alterar o *decimal separator* para ","de forma a garantir a leitura correta dos tipos de dados das colunas.

3.2.4 Distribuição dos dados

De forma a verificamos novamente a distribuição de dados ou ainda possíveis dados assimétricos, utilizamos o Node *Data Explorer*.

Column	Exclude Column	Minimum	Maximum	Mean	Standard Deviation	Variance	Skewness	Kurtosis	Overall Sum
Age	☐	4	90	44.746	16.190	262.111	-0.029	-0.560	26087
birth_year	☐	1933	2019	1978.254	16.190	262.111	0.029	-0.560	1153322
birth_month	☐	1	11	5.909	2.900	8.413	0.053	-1.148	3445
TB	☐	0.400	75	3.299	6.210	38.558	4.907	37.164	1923.200
DB	☐	0.100	19.700	1.486	2.808	7.888	3.212	11.353	866.400
Alkphos	☐	63	2110	290.576	242.938	59018.867	3.765	17.753	169406
Sgpt	☐	10	2000	80.714	182.620	33350.194	6.549	50.579	47056
Sgot	☐	10	4929	109.911	288.919	83473.916	10.546	150.920	64078
TB (#1)	☐	2.700	9.600	6.483	1.085	1.178	-0.286	0.233	3779.700
ALB	☐	0.900	5.500	3.142	0.796	0.633	-0.044	-0.388	1831.700
CHOL	☐	0	0	0	0	0	0	0	0
AG_Ratio	☐	0.300	2.800	0.949	0.320	0.102	1.004	3.278	549.500
BILmg	☐	0.023	4.386	0.191	0.358	0.128	5.067	40.122	107.204

Figura 3: Distribuição dos dados - Node *Data Explorer*.

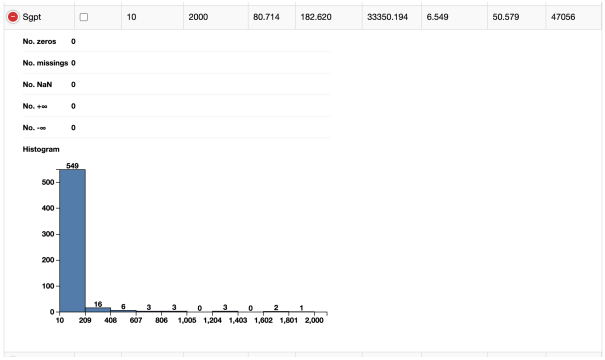


Figura 4: Detalhes da coluna "Sgpt".

No caso da *feature* "Sgpt", por exemplo, notamos um certo desequilíbrio e assimetria na distribuição dos dados, pois notamos que grande parte localiza-se do lado esquerdo do histograma. Esta assimetria denomina-se de *positive skew*. Uma distribuição com assimetria positiva ocorre quando a cauda direita da distribuição é mais longa ou mais ampla do que a cauda esquerda. Isso significa que a maioria dos dados está concentrada do lado esquerdo do histograma, enquanto os valores extremos ou outliers tendem a estar mais dispersos no lado direito. Dado o tema do dataset e a nossa interpretação isto tem significado, pois no contexto do dataset, onde a feature "Sgpt"está relacionada à presença de bilirrubina no fígado, uma assimetria positiva indica que a maioria dos pacientes tem níveis de Sgpt relativamente baixos, mas há alguns pacientes com níveis muito altos, o que pode indicar uma presença aumentada de bilirrubina no fígado em casos de doença hepática. Assim, a assimetria torna-se relevante porque irá ajudar-nos, no futuro, a perceber se a pessoa tem ou não doença, trazendo novo conhecimento relevante ao nosso modelo.

3.3 Análise do dataset

3.3.1 Nodes utilizados para análise dos dados

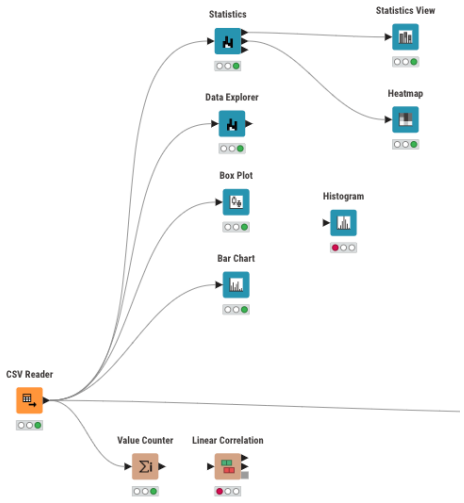


Figura 5: Nodes utilizados para análise dos dados.

3.3.2 Estatísticas

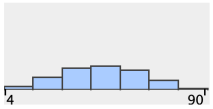
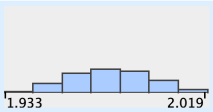
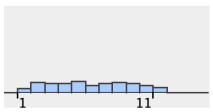
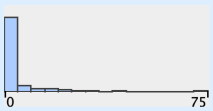
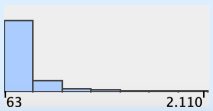
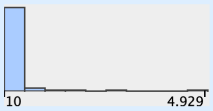
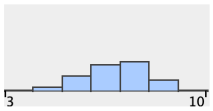
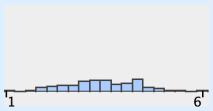
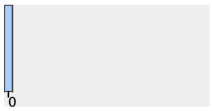
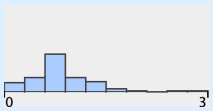
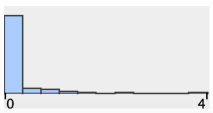
Column	Min	Mean	Median	Max	Std. Dev.	Skewness	Kurtosis	No. Missing	No. +∞	No. -∞	Histogram
Age	4	44,7461	?	90	16,1898	-0,0294	-0,5601	0	0	0	
birth_year	1 933	1 978,2539	?	2 019	16,1898	0,0294	-0,5601	0	0	0	
birth_month	1	5,9091	?	11	2,9005	0,0533	-1,1478	0	0	0	
TB	0,4	3,2988	?	75	6,2095	4,9075	37,1638	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	

Figura 6: Estatísticas do dataset.

Desde já, com base no resultado dos Histogramas conseguimos chegar à conclusão que há um grande desbalanceamento de valores nalgumas colunas. Conseguimos também visualizar as colunas com *missing values* ou ainda algumas estatísticas como a média dos valores.

Média: A média fornece uma medida da tendência central dos dados. Se a média é significativamente diferente do valor esperado ou se varia muito entre diferentes colunas, isso pode indicar distribuição não normal ou presença de outliers.

Mínimo e Máximo: Esses valores indicam os extremos dos dados. Eles fornecem uma ideia dos limites inferiores e superiores dos valores observados em cada coluna. Valores extremos podem indicar a presença de *outliers* ou erros nos dados, como podemos já notar nalgumas colunas ("Sgot", "Sgpt", etc).

3.3.3 Desbalanceamento

Com base no gráfico da figura, já conseguimos identificar a existência de desbalanceamento, algo crucial de se tratar. Conseguimos observar que existem muitos mais registos quando o target é 1 (liver disease), quando comparado com o outro número de ocupação (2 - no liver disease). A diferença de valores traduz-se numa distribuição assimétrica, que pode induzir a *overfitting* no posterior treino do modelo. O desbalancemaneto deve ser corrigido e tratado para que o modelo seja robusto e generalize bem para novos casos. Generalizar bem é ser capaz de prever corretamente tanto para o caso positivo (minoria) como para o negativo (maioria).

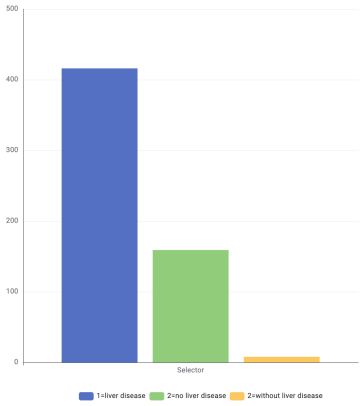


Figura 7: Bar Chart da coluna "Selector".

Além disso, com este gráfico, conseguimos perceber que os dados do tipo 2 podem ser representados por "2-no liver disease"ou por "2-without liver disease". Tendo em conta que ambos se referem ao mesmo, devemos corrigir de forma a que esses dois ocupem um só valor.

3.3.4 Missing Values

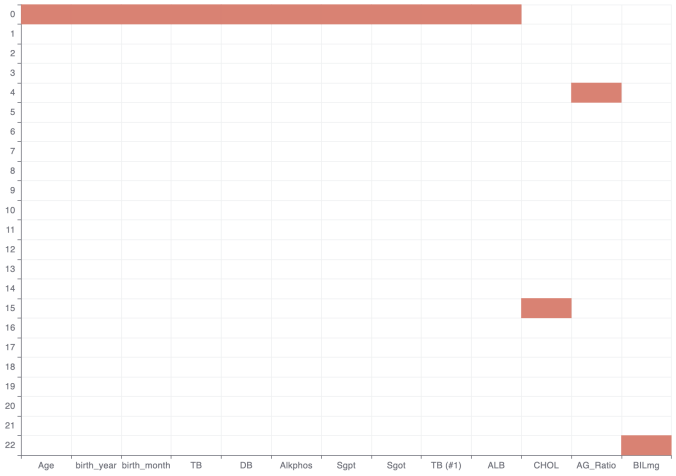


Figura 8: Heatmap dos *missing values*.

Com base no gráfico da figura, conseguimos identificar a existência de alguns *missing values*, nomeadamente nas colunas: "CHOL", "AG_Ratio"e "BILMG". Para tratar dos *missing values* temos várias opções:

- substituir pela moda;
- substituir pela mediana;
- substituir pela média;
- apagar a linha.

Desta forma devemos procurar a opção que nos parece mais adequada. A presença de *missing values* pode ter um impacto significativo na qualidade da análise e no desempenho dos modelos, por diversas razões tais como distorção dos resultado ou dificuldade de interpretaação dos modelos.

3.3.5 Outliers

De forma a localizar os *outliers* do nosso dataset foi relaizado um diagrama *Box Plot*.
Conseguimos visualizar bastantes *outliers* tal como na coluna "Sgot"em que temos os seguintes valores:

- Max = 180;
- Q3=87;
- Median = 42;
- Q1 = 25;
- Min = 10;

Neste caso, rapidamente avaliamos a dispersão e a tendência central de o conjunto de dados. Mais concretamente, temos que:

- Q1 = 25 : 25% dos valores da coluna Sgot são inferiores a 25
- Q2 = Mediana = 42 : 50% dos dados são menores ou iguais a 42
- Q3 = 87 : 75% dos dados são menores ou iguais a 87

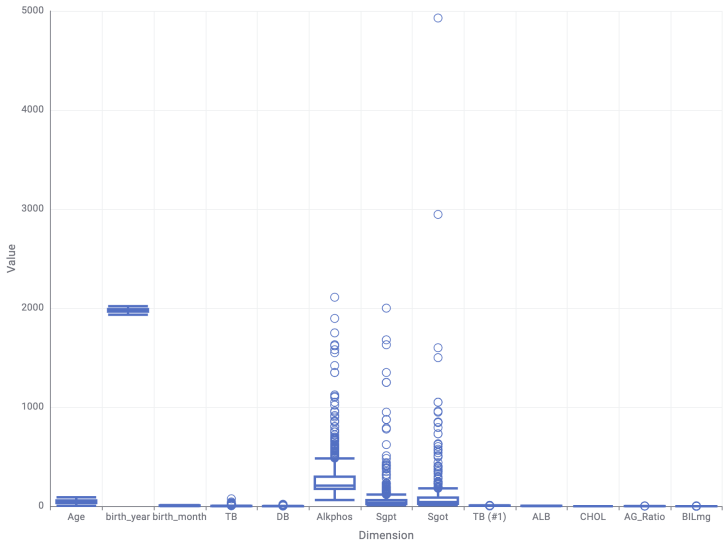


Figura 9: Box Plot dos dados.

Com base nas conclusões anteriores, podemos inferir que a distribuição dos dados na coluna Sgot é assimétrica, com uma concentração maior de valores inferiores a 42, visto que a mediana está abaixo do terceiro quartil. Além disso, a presença de valores extremos acima do terceiro quartil (Q3 = 87) pode indicar uma cauda longa à direita na distribuição dos dados, o que sugere que alguns valores podem estar distantes da maioria dos dados (*outliers*), o que pode influenciar a interpretação estatística e a análise dos dados.
Assim, para identificar os outliers utilizamos a *IQR Distance*, também conhecida como Distância Interquartil. Esta medida de robustez estatística é utilizada para identificar outliers (valores atípicos) num conjunto de dados. Ao contrário de medidas tradicionais como o desvio padrão, o IQR Distance é menos sensível à presença de valores extremos, tornando-o mais adequado para analisar dados com outliers. O IQR por si só, representa a dispersão dos dados medindo o intervalo central, onde se encontram 50% dos valores.

- $IQR = Q3 - Q1 \quad IQRDistance = |x - Q2|/(IQR/1.35)$

Para tratar dos *outliers*, tal como nos *missing values*, temos várias opções: substituir pela moda, substituir pela mediana, substituir pela média ou apagar a linha. Desta forma, mais uma vez, devemos procurar a opção que nos parece mais adequada.

3.3.6 Duplicados

Sendo o objetivo garantir a precisão e confiabilidade dos resultados da análise é recomendado remover linhas duplicadas. De facto, dados duplicados podem introduzir imprecisões nos modelos de *Machine Learning*, pois aumentam artificialmente a frequência de determinados valores, distorcendo a distribuição real dos dados e influenciando indevidamente o processo de aprendizado do modelo. Isto pode levar a previsões incorretas. Para além disso, a sua remoção permite melhorar a eficiência do treinamento e da análise, pois modelos de *Machine Learning* podem ser mais lentos e menos eficientes para treinar em datasets com linhas duplicadas, visto que necessitam de processar as mesmas informações várias vezes.
Através do node *Duplicate Row Filter* do KNIME visualizamos as linhas duplicadas do nosso dataset.Com o mesmo, irá-se mais tarde removê-los.

3.3.7 Relação entre variáveis - Correlação

Com base nas matrizes de correlação estabelecidas conseguimos localizar as colunas cuja a correlação aproxima se de 1 ou -1. A correlação estar próxima de 1 significa que quando uma aumenta a outra também, ou seja, indica que as categorias das variáveis tendem a coincidir. A correlação estar próxima de -1 significa que quando uma aumenta a outra diminui, ou seja, indica que as categorias das variáveis tendem a ser opostas.

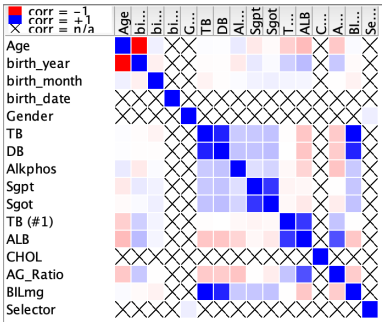


Figura 10: Matriz de correlação - *Linear Correlation*.

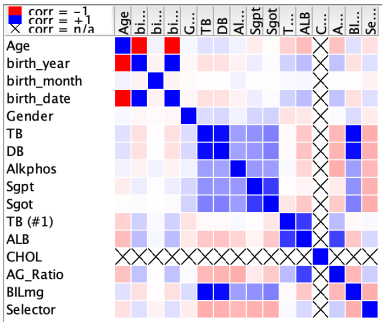


Figura 11: Matriz de correlação - *Rank Correlation*.

A correlação é uma medida estatística que quantifica a força e direção da associação entre duas variáveis.

A correlação linear é útil para analisar relações entre variáveis contínuas (que podem assumir qualquer valor dentro de um intervalo), como altura e peso ou temperatura e umidade. Esta é frequentemente utilizada em diversas áreas, como a das Ciências para estudar relações entre variáveis químicas ou biológicas, sendo adequada no nosso caso.

A correlação por classificação mede a força e a direção da associação entre duas variáveis categóricas (que só podem assumir valores discretos e pré-definidos), como o gênero, por exemplo. Desta forma, também se revela útil no caso de algumas *features* do nosso dataset.

3.4 Preparação dos dados do dataset atribuído

3.4.1 Tipos de dados.

Numa primeira fase, procuramos perceber se o tipo de dados correspondia efetivamente aos dados de cada coluna. De facto, notamos que algumas colunas eram de tipo *String* quando na verdade o seu conteúdo era do tipo Decimal ou Inteiro.

Aí, apercebemo-nos que os parâmetros do *CSV Reader* podiam não estar bem configurados, pelo que, alteramos os mesmos. Uma das alterações foi colocar a vírgula como separador decimal.

3.4.2 Remover duplicados

Com o objetivos de remover os elementos duplicados identificados na fase anterior, utilizamos o node *Duplicate Row Filter* de forma a guardar o primeiro elemento e remover todos os seus duplicados. Assim, pelos motivos referidos na análise dos dados, tornamos a previsão já mais confiável.

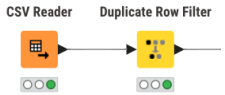


Figura 12: Node *Duplicate Row Filter*.

3.4.3 Remover Features

Na observação dos dados notamos que a coluna "CHOL"era preenchida somente por valores iguais a 0. Desta forma, ela acaba por ser irrelevante para o nosso problema, pois os seus valores são sempre iguais, não trazendo novo conhecimento, pelo que, decidimos removê-la.

Para além disso, decidimos proceder a remoção de *features* correlacionadas, pois essas dificultam a aprendizagem de padrões pelo modelo.

Com a analise das matrizes de correlação, chegamos a conclusão que a coluna "birth_day"era repetitiva, pois já possuímos a coluna "Age"ou ainda "birth_year". Para além disso, a coluna "birth_day", apesar de ter um padão *year/month/day* essa era somente preenchida com o ano, estando sempre na forma de: *year/00/00* o que se torna inútil, pois já possuímos a coluna "birth_year". Assim, também iremos proceder a remoção desta coluna.

A coluna TB foi também removida de forma a sua forte correlação (igual a 1) com a coluna DB que mantivemos. Para tal, utilizamos o node *Column Filter*.

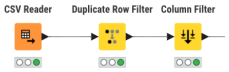


Figura 13: Node *Column Filter*.

3.4.4 Converter os dados Categóricos para Numéricos

A coluna Géneros (masculino, feminino) do nosso dataset é categórica. Para além disso, também reparamos que existiam diversos valores para o mesmo significado (Male=Masculine=male) como se segue na Figura 11.

Desta forma, com o objetivo de transformar o masculino para "M"e o feminino para "F"utilizamos o node *Rule Engine* com as seguites equações:

- $\$Gender\$LIKE"Male" => "M"$
- $\$Gender\$LIKE"Masculine" => "M"$
- $\$Gender\$LIKE"male" => "M"$
- $\$Gender\$LIKE"Female" => "F"$
- $\$Gender\$LIKE"female" => "F"$

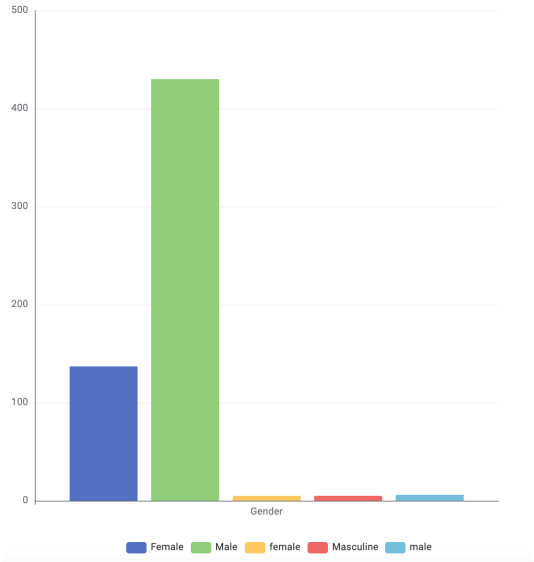


Figura 14: Bar Chart da coluna Gênero .

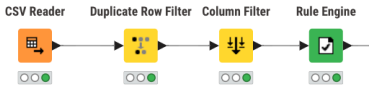


Figura 15: Node *Rule Engine*.

De seguida, faltava-nos transformar a coluna para numérica, pois é necessário de forma a garantir compatibilidade com os modelos de *machine learning*, tal como *Regressão Linear* e permitir a realização de tarefas como previsão, classificação e agrupamento.

Neste caso existem várias soluções. Podíamos usar o ***Ordinal Encoding*** passando os "M"para "0"e os "F"para "1". No entanto, é importante notar que esta resolução não é perfeita, pois os modelos acaba por atribuir uma ordem, mesmo não havendo, ou seja, ele considera o valor 1 mais importante que o 0 quando, neste caso, não é verdade. Outra solução seria, por exemplo, o ***One hot encoding*** na qual são criadas duas colunas, uma para cada valor existente, que serão preenchidas por 1 ou 0. Este algoritmo pode não ser o melhor caso sejam geradas muitas colunas a partir de uma coluna original, aumentando assim a dimensionalidade do conjunto de dados o que faz com que, pelo aumento da dimensionalidade, o tratamento do desbalanceamento seja muito mais demorado, não sendo eficiente. O aumento do numero de *features*, em geral, diminui a performance do modelo. No entanto, no caso da coluna *Gender* seriam somente criadas duas colunas (Feminino e Masculino). Desta forma, obtamos por esta resolução.

Desta forma, na nossa opinião, o *One hot encoding*, em termos de relação *accuracy-speed*, é mais vantajoso, porque o número de colunas geradas é reduzido e o *encoding* mantém a relação da *feature* (não há uma ordem atribuída entre os valores e essa relação deve ser preservada).

Para tal, utilizamos o Node *Column Aggregator* criando-se duas colunas (Masculine e Feminine). De seguida, usou-se o Node *Rule Engine* com as seguintes equações:

$Feminine$LIKE"F"→"1"; $Feminine$LIKE"M"→"0"
$Masculine$LIKE"M"→"1"; $Masculine$LIKE"F"→"0"
Atribuindo assim, os valores 0 e 1 em cada coluna.
Para além disso, foi necessário corrigir a coluna "Selector", igualando os valores referentes a "2=no liver disease"e "2=without liver disease". Desta forma, utilizamos as seguintes equações:

- $Selector$ LIKE "2=no liver disease"= "0"
- $Selector$ LIKE "2=without liver disease"= "0"
- $Selector$ LIKE "1=liver disease"= "1"

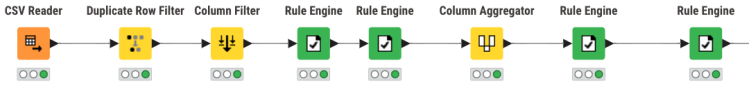


Figura 16: Nodes *Column Aggregator* e *Rule Engine*.

3.4.5 Outliers e Missing Values

Relativamente aos *outliers*, como verificamos anteriormente no gráfico *box plot* da figura 5, algumas *features* necessitam de ser corrigidas neste sentido.
Como referido anteriormente, tanto os *outliers* como os *missing values* podem ser tratados de diversas formas: substituir pela moda, substituir pela mediana, substituir pela média, apagar a linha,
Como sabemos a opção de apagar é somente utilizada em último recurso ou quando são poucos *outliers*/*missing values*, pois neste caso perdemos dados.
Na nossa opinião, o mais assertivo seria começar por substituir pela moda, pois esse é o valor mais neutro e portanto não gera desequilíbrio nos dados. A moda é o número mais comum de um conjunto de dados.
Utilizando os histogramas calculamos a moda:

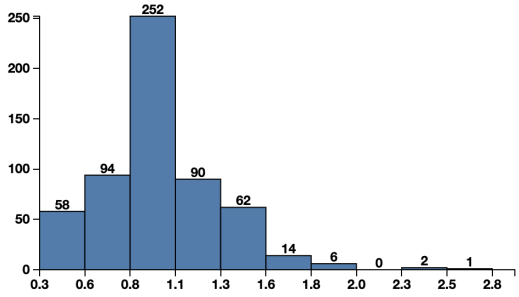


Figura 17: Histograma da *AG_Ratio*.

#	RowID	count *Number (integer)*
1	?	4
2	0.3	4
3	0.4	17
4	0.5	37
5	0.6	36
6	0.7	58
7	0.8	72
8	0.9	66
9	1.0	114

Figura 18: Resultados do Node *Value Counter*.

Por exemplo, no caso da coluna AG_Ratio, iremos substituir os *outliers* e *missing values* por 1,0, pois com o Node *Value Counter* conseguimos ver que este valor é o que tem mais ocorrências.

Numa primeria fase, foram tratados os *missing values* separadamente e de seguida decidimos passar os *outliers* para *missing values* e tratar estes últimos. Desta forma, conseguíamos observar as alterações de cada correção separadamente.

Para realizar a substituição dos *missing values* utilizamos o Node *Missing Values* com os *settings* da figura 16.
Para a substituição dos *outliers* para *missing values* utilizamos o Node *Numeric Outliers* com as opções : "Replace outlier values"e "Missing values".

Figura 19: Settings do Node *Missing Values*

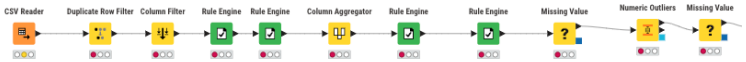


Figura 20: Nodes *Numeric Outlier* e *Missing Value*.

3.4.6 Desbalanceamento dos dados

Para corrigirmos o desbalanceamento surgiram duas opções: o *Random Under Sampling* e o *Synthetic Minority Oversampling Technique*(SMOTE).
No *Random Under Sampling*, o objetivo é balancear a distribuição de classes eliminando aleatoriamente exemplos da classe maioritária. Isso pode ser vantajoso para melhorar o tempo de execução e reduzir o custo computacional, mas também pode resultar na perda de informação. Por outro lado, o SMOTE é uma técnica estatística que aumenta o número de casos no conjunto de dados de forma equilibrada, gerando novas entradas a partir dos casos minoritários existentes.
Desta forma, decidimos utilizar o SMOTE. Na sua configuração optamos por selecionar a opção *Oversample Minority Classes* de forma a acrescentar exemplos sintéticos na classe minoritária.

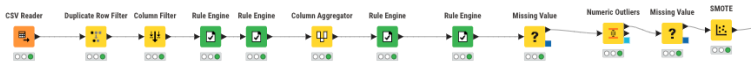


Figura 21: Node *SMOTE*.

Desta forma, conseguimos uma saída que contém o mesmo número de filas para cada uma das classes possíveis (figura 22).
Depois das várias etapas referidas anteriormente consideramos o nosso modelo já mais limpo.
Segue um exemplo do *Box Plot* após o tratamento, nomeadamente o de *outliers*, por exemplo (figura 23).

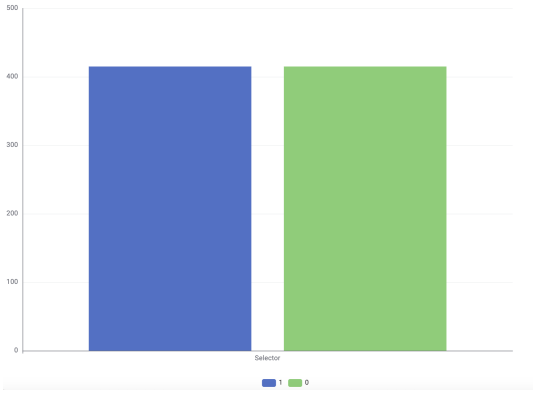


Figura 22: Bar Chart do Selector atual.

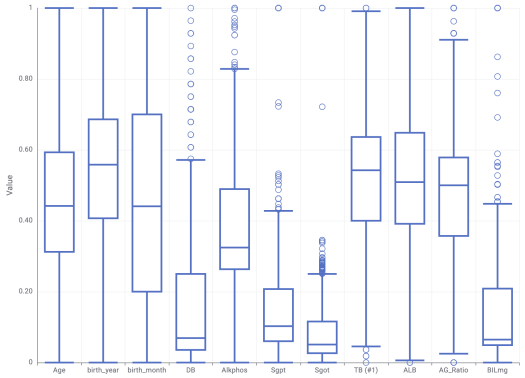


Figura 23: Node *Box Plot após o processo de preparação dos dados.*

3.4.7 Normalização

Relativamente à normalização, o nosso modelo necessita que os dados estejam normalizados no intervalo [0,1]. As variáveis com intervalos maiores afetam o modelo, pois são consideradas como "mais importantes". Desta forma, podem acabar por comprometer a precisão do algoritmo.
Como tal, em ambos os casos usamos a normalização gaussiana.

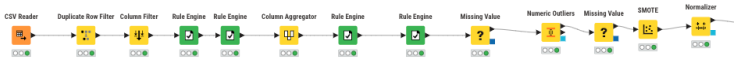


Figura 24: Node *Normalizer.*

3.4.8 Partitioning

De forma a dividirmos o dataset em treino e teste utilizamos o Node *Partitioning.*

Figura 25: Nodes *Partitioning.*

3.5 Modelos

.
Depois das várias etapas referidas anteriormente consideramos o nosso modelo limpo e iniciamos o processo de teste e treino utilizando diversos modelos.

3.5.1 Decision Tree

Ferramenta poderosa para análise de dados. Ele utiliza uma estrutura hierárquica em forma de árvore para prever ou classificar novos dados com base em padrões identificados num conjunto de dados de treinamento.

Figura 26: Node *Decision Tree Learner* e *Decision Tree Predictor*.

3.5.2 Random Forest

Algoritmo que constrói um conjunto de árvores de decisão para melhorar a previsão e generalização. Invés de confiar numa única árvore, o *Random Forest* combina as previsões de todas as árvores para obter um resultado final mais preciso e robusto.

Figura 27: Node *Random Forest Learner* e *Random Forest Predictor*.

3.5.3 Logistic Regression

Algoritmo utilizado para tarefas de classificação binária. Diferentemente da Regressão Linear, que prevê valores contínuos, a Regressão Logística estima a probabilidade de um evento ocorrer, classificando novos dados em duas categorias.

Figura 28: Node *Logistic Regression Learner* e *Logistic Regression Predictor*.

3.5.4 Gradient Boosted Trees

Algoritmo poderoso para classificação e regressão. Combina múltiplas árvores de decisão fracas sequencialmente para criar um modelo de previsão forte.

Figura 29: Node *Gradient Boosted Trees Learner* e *Gradient Boosted Trees Predictor*.

3.5.5 Tabela dos Resultados dos modelos

Com a tabela abaixo conseguimos perceber que o modelo que nos oferece uma melhor previsão é o *Random Forest*. Desta forma, passou-se para um tratamento individual do modelo criando-se vários cenários.

O *Random Forest* é um modelo, no qual o tratamento de outliers não melhora propriamente o modelo, pois esse último é bastante robusto. Assim, conseguimos notar que ao tratar os outliers de formas distintas, como veremos nos cenários a seguir, não haverá grandes variações nos resultados. Iremos notar no entanto, que removê-los já afeta um pouco mais negativamente.

Também experimentou-se variações na normalização, mas não teve grande impacto no modelo, tendo vindo piorar o modelo caso não sejam normalizados os dados. É também importante referirmos que o próprio algoritmo já possui alguma normalização.

Modelos				
	Decision Tree	Random Forest	Logistic Regression	Gradient Boosted Trees
True Positive	55	75	48	59
True Negative	69	61	65	76
False Positive	23	17	30	19
False Negative	19	13	23	12
Accuracy	74,699%	81,928%	68,072%	81,325%
Error	25,301%	18,072%	31,928%	18,675%
Cohen's kappa	0,491	0,636%	0,356%	0,623%
Correct classified	124	136	113	135
Wrong classified	42	30	53	31

Tabela 1: Resultados dos modelos

3.5.6 Accuracy e Confusion Matrix

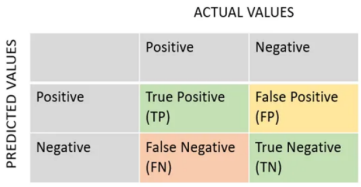


Figura 30: Esquema da *confusion matrix*.

Figura 31: Fórmula da *Accuracy*.

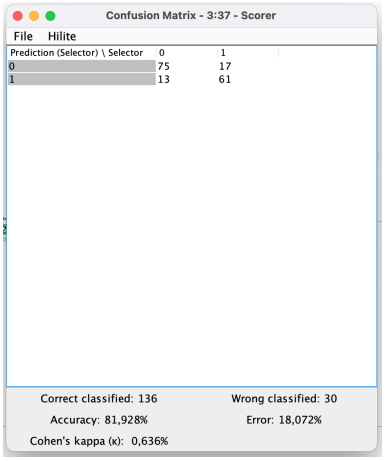


Figura 32: *Confusion matrix* do cenário 1.

Para percebermos os resultados dos nossos modelos e mais a frente cenários é importante esclarecermos alguns conceitos.

- **Accuracy:** Taxa de acerto do modelo, uma métrica comum, porém potencialmente enganosa em conjuntos de dados desbalanceados. Dado que corrigimos o desbalanceamento do nosso conjunto de dados, consideramos essa métrica adequada para avaliação. Em cenários desbalanceados iremos focar nos parâmetros da matriz de confusão.
- **Matriz de Confusão:** Tabela que regista quantas vezes o modelo classificou cada classe corretamente ou incorretamente. Os valores referentes aos TRUE (positive e negative) indicam quantas vezes o modelo acertou a classificação de cada classe, enquanto que os elementos FALSE mostram quantas vezes o modelo errou a classificação de cada classe.

3.5.7 Precision e Recall

De forma a avaliarmos ainda mais os resultados, podemos efetuar mais cálculos, nomeadamente os da *Precision* e o *Recall*:

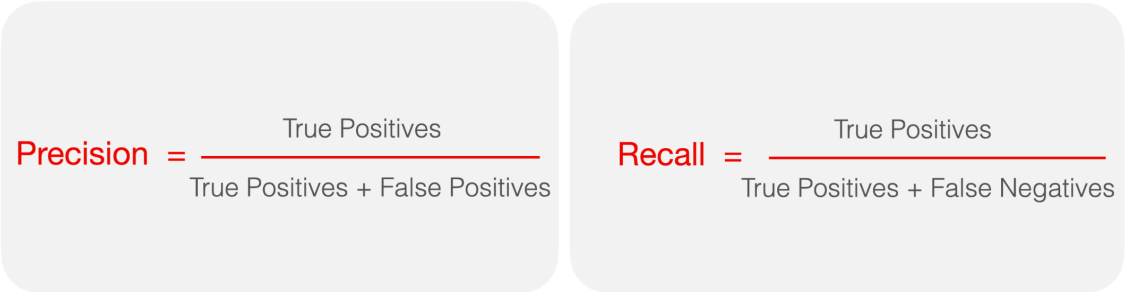


Figura 33: Fórmulas da Precision e Recall.

A *Precision* é uma métrica que mede com que frequência um modelo prevê corretamente a classe positiva. O *Recall*, por sua vez, é uma métrica que mede com que frequência um modelo identifica corretamente instâncias positivas (verdadeiros positivos) entre todas as amostras positivas reais no conjunto de dados.

Com base nessas duas fórmulas podemos também calcular a :

- Balanced Accuracy = $\frac{Precision+Recall}{2}$

É importante referir que esses parâmetros são maioritariamente utilizados em dados desbalanceados, nos quais a *accuracy* não tão é fiável.

No caso da matriz de confusão do *Random Forest* do nosso cenário 1 (Figura 30) teríamos então:

- Precision = $\frac{75}{75+17}$ = 0,815
- Recall = $\frac{75}{75+13}$ = 0,852
- Balanced Accuracy = $\frac{0,815+0,852}{2}$ = 0,833

Com esses valores, sendo todos relativamente perto do 1, podemos perceber que o cenário em questão está a ter um boa previsão do modelo.

3.6 Cenários

Depois de criarmos o nosso melhor cenário, procuramos variar algumas das correções realizadas procurando sempre uma melhor previsão. Seguem-se alguns exemplos de cenários escolhidos de forma a variarmos os valores da previsão:

Cenário 1: Este cenário foi o nosso melhor cenário que abrange todos os nodos referidos anteriormente. Procuramos corrigir as correlações, substituímos os *outliers* e *missing values* pela mode, corrigimos o desbalanceamento, passamos os dados todos para numéricos, removemos duplicados etc.

Cenário 2 (com string to number): Este cenário, foi o nosso primeiro cenário antes de notarmos que o nosso *CSV reader* não estava corretamente configurado. Desta forma, o *CSV reader* mostrava-nos colunas do tipo String com valores numéricos. Assim utilizamos o *String to number* de forma a converter as respetivas colunas. Mais tarde apercebemo-nos que o nosso *CSV reader* estava mal configurado e por isso foi corrigido colocando a ","como *decimal separator*. Este cenário ainda possui o String to number.
Aí reparamos que o desempenho até é significativo, no entanto, a a utilização correta do node *CSV Reader* é importante, pelo que optamos por corrigir o mesmo, não sendo necessário o uso do *String to number*.

Cenário 3 (missing values e outliers substituídos pela média): Neste cenário a ideia foi corrigir os *outliers* e *missing values* substituindo-os pela média em vez da moda. Assim procuramos ver se com a media conseguíamos uma *accuracy* melhor. De um modo geral, mantiveram se os resultados. Assim, podemos concluir que a substituição da forma como tratamos os *outliers* e *missing values* era pouco relativa desde que fossem tratados.

Cenário 4 (missing values e outliers substituídos pela mediana): Neste cenário a ideia foi corrigir os *outliers* e *missing values* substituindo-os pela mediana em vez da moda. Assim procuramos ver se com a mediana conseguíamos uma *accuracy* melhor. Mais uma vez, mantiveram se os resultados.

Cenário 5 (missing values e outliers removidos): Neste cenário a ideia foi corrigir os *outliers* e *missing values* usando o drop. Desta forma, queríamos remover os *outliers* e *missing values* procurando verificar se ocorria grandes variações.
Neste cenário, notamos que o desempenho foi menor podendo-se concluir que a remoção dos *outliers* e *missing values* acaba por afetar, pois estamos a perder dados que podem ser relevantes para a previsão, por exemplo.

Cenário 6 (com uma só coluna para o *Gender*): Neste cenário a ideia foi corrigir a coluna *Gender* transformando os M e F por 0 e 1. Este método não é o melhor, pois tal como referido anteriormente, os modelos irão considerar o 1 mais importante que o 0, o que não se revela verdadeiro no caso desta coluna *Gender*. Mais tarde adotamos portanto a ideia de criar 2 colunas (uma coluna F e uma coluna M), como no cenário 1.
Desta forma, neste cenário optamos por usar o ***Ordinal Encoding*** passando os "M"para "0"e os "F"para "1". Assim, utilizamos o nó *String Manipulation* com a seguinte equação:
$replace(replace(\$Gender\$,"M","1"),"F","0")$.
É importante notar que esta resolução não é perfeita, tal como referimos anteriormente. Noutros cenários, adotamos portanto a ideia de criar 2 colunas (uma coluna F e uma coluna M), *One hot encoding*.
O desempenho, apesar de pouco relevante, diminui. Desta forma, notamos que o *One hot encoding* revela-se mais eficiente do que *Ordinal Encoding*, no nosso caso.

Cenário 7 (sem SMOTE): Neste cenário, retiramos o SMOTE de forma a não ser corrigido o desbalanceamento, verificando assim se influencia ou não a previsão.
Sem o SMOTE notamos imediatamente um decréscimo na *accuracy* quase de 10%. Assim, percebemos a importância do SMOTE e de tratarmos o desbalanceamento.

Cenário 8 (sem normalização): Neste cenário, retiramos o Normalizer de forma a não normalizarmos o modelo. Assim, os valores não serão colocados no mesmo intervalo de valores fazendo com que o modelo considere os mais altos como mais importantes.
Houve uma deterioração no desempenho, mesmo não tendo sido significativa.

Cenário 9 (sem SMOTE e sem normalização): Neste cenário, retiramos o SMOTE e o Normalizer. Notando-se imediatamente um decréscimo significativo na *accuracy* e nos valores referentes aos TRUES da matriz de confusão, provavelmente mais devido a retirarmos o SMOTE do que o Normalizer, pelos dois cenários anteriores.

Cenário 10 (normalizado com z-score): Neste cenário, procuramos normalizar com o Z-score em vez de utilizar a min-max Normaliation. Os resultados foram bastante bons, sendo este o nosso segundo melhor cenário.

Apesar de ser representado como cenário 1 para uma melhor visualização das comparações, este não foi o nosso primeiro cenário, mas sim o melhor ao qual conseguimos chegar. Desta forma, procuramos comparar as previsões dos diversos cenários com as do cenário 1.

Tendo em conta que com os vários cenários apercebemo-nos que o melhor modelo seria o Random Forest, fez-se uma tabela de forma a comparar os diferentes resultados:

	C1	C2	C3	C4	C5
Alteração		String to Number	Missing Values & Outliers (mean)	Missing Values & Outliers (median)	Missing Values & Outliers removed
TP	75	73	74	73	31
TN	61	57	57	58	43
FP	17	21	21	20	18
FN	13	15	14	15	7
Accuracy	81.93%	78.31%	78.92%	78.92%	74.75%
Error	18.07%	21.69%	21.08%	21.08%	25.25%
Cohen's kappa	0.636	0.563	0.575	0.575	0.494
Correct	136	130	131	131	74
Wrong	30	36	35	35	25

Tabela 2: Resultados dos cenários (parte 1)

	C6	C7	C8	C9	C10
Alteração	1 coluna Gender (0 e 1)	Sem SMOTE	Sem Normalização	Sem SMOTE e Normalizer	Z-score
TP	72	72	71	73	74
TN	58	10	58	11	60
FP	20	24	20	23	18
FN	16	11	17	10	14
Accuracy	78.31%	70.09%	77.71%	71.80%	80.72%
Error	21.69%	29.91%	22.29%	28.20%	19.28%
Cohen's kappa	0.563	0.182	0.552	0.229	0.612
Correct	130	82	129	84	134
Wrong	36	35	37	33	32

Tabela 3: Resultados dos cenários (parte 2)

3.6.1 Cross-validation

O *cross validation* é conhecido por tentar atenuar o desbalanceamento de classes e o overfitting.

Desta forma, também com o propósito de ter uma análise mais fundamentada, escolhemos aplicar *cross validation* no melhor cenário (cenário 1).

	Modelos			
	Decision Tree	Random Forest	Logistic Regression	Gradient Boosted Trees
True Positive	291	369	376	397
True Negative	85	46	43	51
False Positive	124	46	39	69
False Negative	82	121	124	116
Accuracy	64,605%	71,306%	71,993%	68,213%
Error	35,395%	28,694%	28,007%	31,7875%
Cohen's kappa	0,195	0,552%	0,193%	0,152%
Correct classified	376	415	419	397
Wrong classified	206	167	163	185

Tabela 4: Resultados dos modelos do cenário 1 com cross validation

Com base nos resultados desta tabela iremos efetuar algumas contas de forma a comparar com os resultados anteriores (sem *cross validation*).

Decision Tree correct classified : $376 \div 582 = 0,65 - -124 \div 166 = 0,75$
wrong classified: $206 \div 582 = 0,35 - -42 \div 166 = 0,25$
Assim, no *Decision Tree*, utilizando os casos corretamente classificados e os casos erradamente classificados como métricas, podemos concluir que o modelo não melhora com cross validation, sendo mais robusto sem.

Random Forest correct classified : $415 \div 582 = 0,71 - -136 \div 166 = 0,81$
wrong classified: $167 \div 582 = 0,29 - -30 \div 166 = 0,18$
Assim, no *Random Forest*, utilizando os casos corretamente classificados e os casos erradamente classificados como métricas, podemos concluir que o modelo não melhora com cross validation, sendo mais robusto sem.

Logistic Regression correct classified : $419 \div 582 = 0,72 - -113 \div 166 = 0,68$
wrong classified: $163 \div 582 = 0,28 - -53 \div 166 = 0,31$
Assim, no *Logistic Regression*, utilizando os casos corretamente classificados e os casos erradamente classificados como métricas, podemos concluir que o modelo melhora com cross validation. Notamos que há mais casos bem previstos e menos casos erradamente previstos, sendo o resultado melhor. Assim, com cross validation o modelo é mais robusto.

Gradient Boosted Trees correct classified : $397 \div 582 = 0,68 - -135 \div 166 = 0,81$
wrong classified: $185 \div 582 = 0,32 - -31 \div 166 = 0,19$
Assim, no *Gradient Boosted Trees*, utilizando os casos corretamente classificados e os casos erradamente classificados como métricas, podemos concluir que o modelo não melhora com cross validation, sendo mais robusto sem.
Para além disso, também utilizamos a opção *seed* de forma a estabelecer um *thresholds* que bloqueia os valores das métricas de forma a dar sempre o mesmo valor, independentemente do número de vezes de execuções.

Como sabemos, o cross-validation é particularmente importante para evitar problemas de *overfitting* e *underfitting*.

> *The main purpose of cross validation is to prevent overfitting, which occurs when a model is trained too well on the training data and performs poorly on new, unseen data. By evaluating the model on multiple validation sets, cross validation provides a more realistic estimate of the model's generalization performance, i.e., its ability to perform well on new, unseen data.*

Figura 34: Nodes utilizados para o cross validation.

De seguida, procuramos utilizar *Cross Validation* com SMOTE no *Logistic Regression*, que tinha sido o melhor modelo com cross validation. Assim, devemos colocar o SMOTE depois do nodo X-PARTICIONER e antes do LEARNER. Usar o SMOTE antes do X-particioner seria errado pois as amostras sintéticas geradas pelo SMOTE podem influenciar a forma como os dados são divididos durante a validação cruzada, resultando num modelo que é otimizado para o conjunto de teste específico, em vez de generalizar bem para dados não vistos.
Assim, temos que:
Desta forma, podemos concluir que neste caso, os resultados são melhores sem SMOTE.

Modelos		
	Logistic Regression sem SMOTE	Logistic Regression com SMOTE
True Positive	376	261
True Negative	43	119
False Positive	39	154
False Negative	124	48
Accuracy	71,993%	65,292%
Error	28,007%	34,708%
Cohen's kappa	0,193%	0,287%
Correct classified	419	380
Wrong classified	163	202

Tabela 5: Resultados do Logistic Regression com e sem SMOTE.

3.7 Conclusões e Análise crítica dos resultados dos vários cenários e modelos

Com base nos cenários apresentados, podemos fazer uma análise comparativa dos diferentes métodos e abordagens utilizadas no pré-processamento dos dados antes e depois da aplicação do algoritmo *Random Forest* para previsão.

Os **Cenários 3 e 4**, que substituem *outliers* e *missing values* pela média e mediana, respetivamente, mantêm resultados semelhantes ao Cenário 1. As variações da accuracy, do erro ou ainda dos TRUE positive/negative e FALSE positive/negative são poucas significativas. Isso sugere que, em alguns casos, a escolha entre média e mediana pode não ter um impacto significativo na precisão do modelo.

Por outro lado, o **Cenário 5**, que opta por remover *outliers* e *missing values*, resulta num desempenho inferior. Isso indica que a remoção de dados pode prejudicar a capacidade do modelo em realizar previsões precisas, pois notamos, por exemplo, que o erro aumenta.

No **Cenário 6**, a transformação da coluna de género utilizando o *Ordinal Encoding* resulta em uma pequena diminuição no desempenho em comparação com o *One-Hot Encoding*, demonstrando que a representação inadequada de dados categóricos pode impactar a precisão do modelo.

Os **Cenários 7, 8 e 9** destacam a importância de certas etapas de pré-processamento. A remoção do SMOTE resulta em uma queda significativa na precisão do modelo (Cenário 7) aumentando o erro de mais de 10% e indicando a importância do tratamento de desbalanceamento de classes. Além disso, a remoção do Normalizer leva a uma pequena deterioração no desempenho (Cenário 8), demonstrando a utilidade da normalização dos dados para garantir que todos os recursos contribuam igualmente para o modelo.

O **Cenário 10**, com Z-score revela-se bastante adequado, pois já atinge os 80% de *accuracy*. O Z-score normaliza os dados para ter uma média de zero e um desvio padrão de um, o que facilita a interpretação dos valores em termos de desvios padrão da média. Enquanto isso, a normalização min-max (utilizada no cenário 1)redimensiona os dados para um intervalo específico, facilitando a compreensão da escala dos valores.

Como já referido anteriormente, o **Cenário 1** representa o **cenário mais completo e robusto**, abordando uma variedade de técnicas de pré-processamento, como correção de correlações, tratamento de *outliers* e *missing values* (com a moda), correção de desbalanceamento e conversão de dados categóricos em numéricos. Este cenário fornece uma base sólida para a construção do modelo, garantindo a qualidade dos dados e potencialmente melhorando a precisão da previsão. Para além disso, foi o cenário com menor erro, maior accuracy e maiores valores acertados na previsão (matriz de confusão). Desta forma, concluímos que este é o mais eficaz em termos de precisão do modelo. **No entanto, é importante referir que a escolha das técnicas de pré-processamento deve ser feita considerando as características específicas do conjunto de dados e o algoritmo de previsão utilizado.**

No caso do cenário 1, com cross validation conseguimos melhor resultados no modelo *Logistic Regression* mas mesmo assim, os resultados do *Random Forest* sem cross validation revelam se melhor, ficando este como **final**.

4 Dataset Selecionado

4.1 Perceber o contexto selecionado (Buisness Understanding)

O dataset escolhido procura prever a velocidade do vento. Possuímos vários parâmetros que nos oferecem uma certa contextualização. Assim, para prever essa velocidade é relevante avaliar temperaturas, tanto minimas como máximas ao longo dos dias. Para além disso, a chuva ou taxa de precipitação. Desta forma, consegue-se prever uma estimativa da velocidade do vento.

4.2 Perceber o Dataset Selecionado (Data Understanding)

4.2.1 Descrição do dataset

Visto que o dataset atribuído era um dataset de classificação, procuramos encontrar um dataset de regressão para esta segunda etapa do trabalho. Desta forma, encontramos no *kaggle* um dataset denominado de "Wind Speed Prediction".
O dataset é um dataset de regressão, isto é, que procura prever um valor, neste caso, o valor da velocidade do vento. O dataset consiste em 6574 observações e possui 9 *features*, sendo 8 delas numéricas e uma categórica.

DATE	WIND	IND	RAIN	IND.1	T.MAX	IND.2	T.MIN	T.MIN.G
1961-01-01	13.67	0	0.2	0	9.5	0	3.7	-1
1961-01-02	11.5	0	5.1	0	7.2	0	4.2	1.1
1961-01-03	11.25	0	0.4	0	5.5	0	0.5	-0.5
1961-01-04	8.63	0	0.2	0	5.6	0	0.4	-3.2
1961-01-05	11.92	0	10.4	0	7.2	1	-1.5	-7.5
1961-01-06	10.67	0	0	0	6.5	0	1.2	-2
1961-01-07	9.17	0	1.9	0	9.2	1	-2.4	-7.1
1961-01-08	14.29	0	0	0	6.6	0	3.1	0

Figura 35: Alguns casos do dataset Escolhido.

4.2.2 Contextualização do tema do dataset (Data Describing)

A previsão da velocidade do vento é uma tarefa importante para obter avisos avançados das condições climáticas severas. Desta forma, este conjunto de dados contém as respostas de um sensor meteorológico que coletou diferentes variáveis meteorológicas, como temperaturas e precipitação. As colunas do nosso dataset são definidas por:

- DATE: Data
- WIND: Velocidade média do vento
- IND: Primeiro valor indicador
- CHUVA: Quantidade de precipitação
- IND.1: Segundo valor indicador
- T.MÁX: Temperatura máxima (°C)
- IND.2: Terceiro valor indicador
- T.MIN: Temperatura mínima (°C)
- T.MIN.G: Temperatura mínima da grama às 09h UTC (°C)

4.3 Análise do Dataset Selecionado

4.3.1 Estatísticas

Desde já, conseguimos analisar alguns parâmetros.

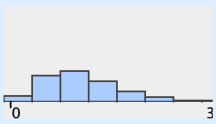
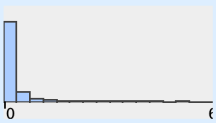
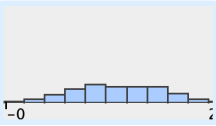
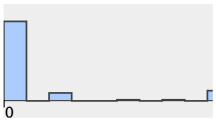
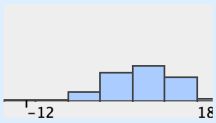
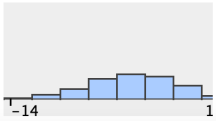
Column	Min	Mean	Median	Max	Std. Dev.	Skewness	Kurtosis	No. Missing	No. +∞	No. −∞	Histogram
WIND	0.0	9,7968	9,21	30,37	4,9773	0,6458	0,2129	0	0	0	
IND	0.0	0,3915	0.0	4	1,1791	2,7207	5,4387	0	0	0	
RAIN	0.0	1,8852	0,2	67	4,0305	4,4951	32,4782	0	0	0	
IND.1	0.0	0,3564	0.0	4	1,1286	2,8854	6,3878	61	0	0	
T.MAX	−0,1	13,3391	13,3	26,8	4,8905	−0,0306	−0,7347	621	0	0	
IND.2	0.0	0,4645	0.0	4	1,1776	2,5159	4,6943	61	0	0	
T.MIN	−11,5	6,4117	6,5	18	4,6372	−0,2007	−0,4825	674	0	0	
T.MIN.G	−14,4	2,7365	3	15,8	5,5692	−0,1957	−0,4815	360	0	0	

Figura 36: Dataset Escolhido.

Mediana: A mediana é especialmente útil para entender a tendência central dos dados, especialmente em conjuntos de dados com *outliers*.

Média: Serve como um ponto de referência para o modelo, permitindo comparações dos valores de cada *feature* individual com a medida central dos dados. Isso irá facilitar a identificação de padrões e relações entre as *features* e a variável alvo.

Máx e Min: Mais uma vez de forma a percebermos entre que valores se enquadram as variações.

4.3.2 Desbalanceamento e Missing Values

Com base no gráfico da figura 37, conseguimos identificar a existência de algum desbalanceamento, mas também da grande variedade de valores.

Com base no gráfico da figura 38, conseguimos identificar a existência de alguns *missing values*, nomeadamente nas colunas T.MAX, T.MIN e T.MIN.G.

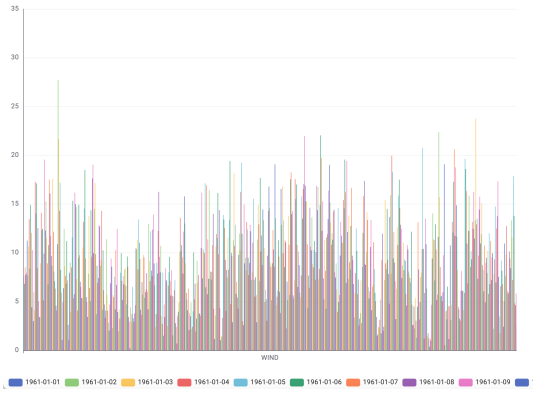


Figura 37: Bar Chart da coluna "WIND".

Figura 38: HeatMap dos *missing values*.

4.3.3 Outliers e Correlações (Relação entre variáveis)

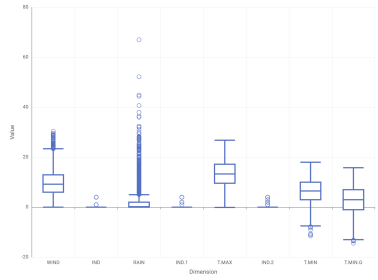


Figura 39: Box Plot dos *Outliers*

.

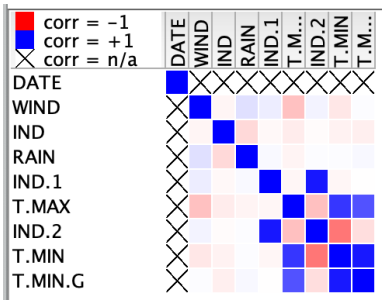


Figura 40: Matriz de correlação - *Linear Correlation*.

Com base no gráfico da figura 39, conseguimos identificar a existência de alguns *Outliers*, nomeadamente na coluna RAIN, por exemplo.

Com base na matriz de correlação conseguimos identificar diversas colunas com uma forte correlação (=1). Assim, temos que o IND.1 e o IND.2 têm uma correlação de 1, pelo que, uma delas poderá ser removida. A T.MAX e a T.MIN também, ou ainda a T.MIN com a T.MIN.G.

Neste dataset, sendo que as variáveis são maioritariamente contínuas (assumem um valor dentro de um intervalo, como a temperatura) escolhemos utilizar a correlação linear.

4.3.4 Duplicados

Através do node *Duplicate Row Filter* fomos analisando a presença de duplicados. Com o mesmo, conseguimos concluir que o nosso dataset não possui duplicados.

4.4 Preparação do Dataset Selecionado

4.4.1 Tipos de dados

Tal como no dataset atribuído notamos que algumas colunas cujo o tipo da coluna era STRING eram na verdade constituídas por números. Desta forma, foi necessário adaptarmos os parâmetros do nosso Node CSV Reader de forma a ele considerar o ponto como separador decimal.

4.4.2 Remover Features

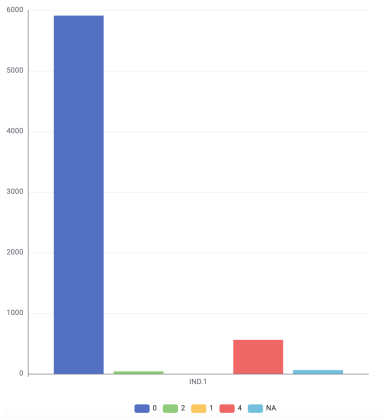


Figura 41: Bar Chart da coluna *IND.1*.

Relativamente a remoção de *features*, analizamos as várias colunas e decidimos remover algumas delas. A primeira coluna a ser removida foi a "IND.1". Como podemos observar no dataset, esta coluna tem um desbalanceamento enorme sendo praticamente somente constituída por zeros. Para além, disso, pela matriz de correlação conseguimos observar que a "IND.1"contem uma correlação de +1 com a "IND.2". Desta forma, uma delas pode e deve ser removida.

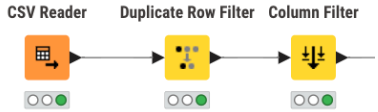


Figura 42: Node *Column Filter*.

Continuando com a análise da matriz de correlação, podemos observar que as colunas "T.MIN.G"e "T.MIN"têm também uma correlação forte entre elas. Para além disso, a "T.MIN"possui também correlação forte com a "T.MAX". Desta forma, a "T.MIN"será também removida, pois possui correlação forte com duas *features*.

4.4.3 Separação de colunas

Durante a analise do nosso dataset reparamos que o mesmo possuía uma coluna "DATE"com valores semelhantes aos seguintes: 1961-01-01; 1961-01-02; 1961-01-03 ...

Assim, devemos separar essa coluna em três colunas diferentes: uma para o ano, uma para o mês e outra para o dia. Removendo a coluna "DATE"e ficando apenas com as três restantes colunas. Para tal, utilizamos o Node Cell Splitter com o delimitador adequado(-) e ativamos a opção "Remove input column"para ser removida a coluna "DATE".

DATE *String*
1961-01-01
1961-01-02
1961-01-03
1961-01-04
1961-01-05

Figura 43: Coluna "DATE"antes do tratamento de dados.

DATE_Arr[0] *Number (integer)*	DATE_Arr[1] *Number (integer)*	DATE_Arr[2] *Number (integer)*
1961	1	1
1961	1	2
1961	1	3
1961	1	4
1961	1	5

Figura 44: Colunas para a data depois do tratamento de dados.

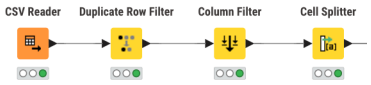


Figura 45: Node *Cell Splitter*.

4.4.4 Outliers e Missing Values

Para os missing values, decidimos manter a lógica do dataset atribuído. Utilizamos o node Value Counter de forma a recolher as modas das colunas. De seguída, com o Node Missing Value, fomos numa primeira fase substituindo os missing values pelas modes e depois, com o Node Numeric Outliers substituímos os outliers. É importante tratar individualmente dessas substituições para termos controlo sobre a mesma. É tambem de notar, que no caso dos outliers, escolhemos convertê-los para missing values antes de os tratar para conseguirmos a substituição dos mesmos.

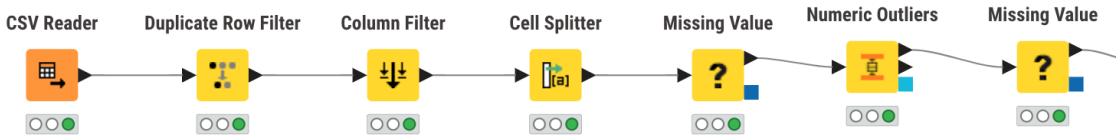


Figura 46: Nodes *Missing Value* e *Numeric Outliers*.

4.4.5 Desbalanceamento dos dados

Apesar de ter sido identificado algum desbalanceamento, o processo de correção de desbalanceamento em regressão, aborda *data argumentation*, que normalmente se faz com *GANs* e *VANs*. No entanto, isso já seria do âmbito generativo pelo que optamos por não tratar.

De forma a procurar controlar um pouco mais o desbalanceamento optamos por utilizar métricas robustas à desbalanceamento, como por exemplo o *Random Forest*.

4.4.6 Normalização

De forma a normalizar os nossos dados entre 0 e 1 para garantir que não sejam considerados valores e *features* mais importantes que outros utilizamos o node Normaizer.

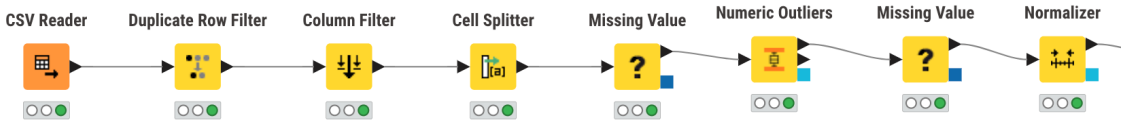


Figura 47: Nodes *Normalizer*.

4.4.7 Partitioning

De forma a dividir os dados e podermos aplicar os modelos utilizamos o Node *Partitioning* dividindo em 80% e 20%.

4.5 Modelos

4.5.1 Linear Regression

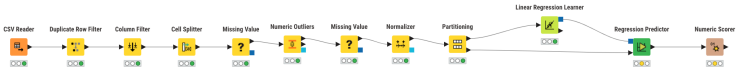


Figura 48: Modelo *Linear Regression*.

4.5.2 Random Forest

Figura 49: Modelo *Random Forest*.

4.5.3 Gradient Boosted Trees Learner

Figura 50: Modelo *Gradient Boosted Trees*.

4.5.4 Tabela dos Resultados dos modelos

Modelos			
	Linear Regression	Random Forest	Gradient Boosted Trees
R^2	0,098	-3,988	-2,13
Mean absolute error	0,156	0,407	0,143
Mean squared error	0,037	0,206	0,031
Root mean squared error	0,193	0,453	0,177
Mean signed difference	-0,009	-0,047	0,005
Mean absolute percentage error	0,591	0,945	0,355
Adjusted R^2	0,098	-3,988	-2,13

Tabela 6: Resultados dos modelos

Para percebermos os resultados dos nossos modelos e mais a frente cenários é importante esclarecermos alguns conceitos, nomeadamente as Métricas de Avaliação Utilizadas:

- **Erro Médio Absoluto:** É a média do erro absoluto entre as previsões do modelo e os valores reais. Calcula-se somando os erros absolutos e dividindo pelo número de observações. Esta métrica não é sensível a *outliers*.
- **Erro Médio Quadrático:** É a média dos erros ao quadrado entre as previsões do modelo e os valores reais. Calcula-se somando os erros ao quadrado e dividindo pelo número de observações. Esta métrica é sensível a *outliers*.
- **Raiz do Erro Médio Quadrático:** É a raiz quadrada da média dos erros ao quadrado entre as previsões do modelo e os valores reais. Representa o desvio médio das previsões do modelo em relação aos valores reais. É sensível a *outliers*.

- **Raiz do Erro Médio Logarítmico Quadrático:** É a raiz quadrada do logaritmo do erro médio ao quadrado entre as previsões do modelo e os valores reais. Representa o desvio médio das previsões do modelo em relação aos valores reais, sendo mais robusta em relação a *outliers* do que o anterior. Utilizamos esta métrica porque é uma medida mais estável do desempenho do modelo, levando em consideração possíveis desequilíbrios nos dados.

O **Coeficiente de Determinação (R^2 Score)**, que mede o quão bem o modelo se ajusta aos dados, variando de 0 a 1, onde 1 indica ajuste perfeito e 0 o pior ajuste possível, não será utilizado como métrica.

Se o modelo escolhido for muito simples para capturar a complexidade dos dados, o R^2 pode ser baixo. Isso pode ocorrer se o modelo não incluir todas as características relevantes dos dados ou se a relação entre as variáveis independentes e dependentes for não linear.

Também o *overfitting* pode ser um dos motivos da sua ineficiência. Se o modelo estiver muito ajustado aos dados de treinamento e não generalizar bem para novos dados, o R^2 pode ser baixo quando aplicado a dados de teste. O *overfitting* pode ocorrer quando o modelo é muito complexo em relação ao tamanho do conjunto de dados de treinamento.

A inclusão de variáveis irrelevantes ou altamente correlacionadas (multicolinearidade) no modelo pode também reduzir a capacidade do modelo de estimar com precisão os efeitos das variáveis independentes sobre a variável dependente, levando a um R^2 baixo.

Se os dados contiverem erros de medição ou ruído, pode ser reduzida a capacidade do modelo de ajustar os dados e explicar a variabilidade na variável dependente, resultando em um baixo R^2.

Se as pressuposições da regressão linear, como normalidade dos resíduos, homocedasticidade e independência dos resíduos, não forem atendidas, isso pode levar a estimativas imprecisas dos parâmetros do modelo e, consequentemente, a um baixo R^2.

Embora o coeficiente de determinação seja uma métrica usada para avaliar a qualidade de um modelo de regressão, existem situações em que pode não ser uma escolha viável ou suficiente como única métrica de avaliação. Aqui estão alguns casos em que o R^2 pode não ser adequado:

- Modelos não lineares: O R^2 é mais apropriado para modelos de regressão linear, onde a relação entre as variáveis independentes e dependentes é assumida como linear.
- Relação não linear entre variáveis: Se a relação entre as variáveis independentes e dependentes for não linear, o R^2 pode subestimar a qualidade do modelo.
- Dados desbalanceados: Para conjuntos de dados desbalanceados, onde uma classe é muito mais prevalente do que outras, o R^2 pode não fornecer uma avaliação precisa da qualidade do modelo, especialmente em modelos de classificação.
- Modelos de séries temporais: Para modelos de séries temporais, o R^2 pode não ser adequado devido à natureza dinâmica e autocorrelacionada dos dados.

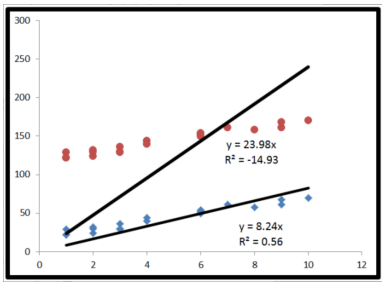


Figura 51: Gráfico exemplo do (R^2 Score)

Embora o R^2 seja uma métrica útil, é importante evitar a obsessão por valores altos do mesmo. O valor ideal de R^2 depende do contexto do problema e da qualidade dos dados.
O R^2 pode não ser fiável em certas situações, especialmente quando os dados não se ajustam bem a uma linha reta. Isso acontece porque o R^2 assume que os dados seguem um padrão linear e que a relação entre a variável independente e a variável dependente é linear.

Se os dados não seguem esse padrão, o R^2 pode não fornecer uma medida precisa de quão bem o modelo se ajusta aos dados. No nosso caso, sendo que os valores do nosso target variam imensos a métrica não é viável, pelo que, não será utilizada.

A otimização de hiperparâmetros é uma das principais estratégias para melhorar o R^2 e o desempenho geral de um modelo de regressão. Os hiperparâmetros são configurações do modelo que influenciam a forma como ele aprende

com os dados. Desta forma, procuramos utilizar o "Limit number of levels"e o "Minimum node size"no *Random Forest* e o "Limit number of levels (tree depth)"no *Gradient Boosted Trees* como hiperparâmetros e ir variando-os procurando melhores valores.
No caso do *Random Forest* não tivemos muitos resultados mas no caso do *Gradient Boosted Trees* ainda conseguimos algumas melhorias:

Limit number of levels (tree depth) do *Gradient Boosted Trees*				
	3	4	5	6
R^2	0,25	0,26	0,266	0,268
Mean absolute error	3,263	3,236	3,224	3,221
Mean squared error	16,649	16,418	16,287	16,24
Root mean squared error	4,08	4,052	4,036	4,03
Mean signed difference	-0,079	-0,076	-0,07	-0,067

Tabela 7: Resultados dos modelos

Conseguimos visualizar um aumento do R^2 e diminuição dos erros a medida que aumentamos o *Limit number of levels*.

4.6 Cenários

Procuramos variar algumas das correções realizadas procurando sempre uma melhor previsão. Seguem-se alguns exemplos de cenários escolhidos de forma a variarmos os valores da previsão:

Cenário 1: Este cenário foi o nosso melhor cenário que abrange todos os nodos referidos anteriormente. Procuramos corrigir as correlações, substituímos os *outliers* e *missing values* pela mode, removemos duplicados e normalizamos os dados. Este é o com menor erro nos parâmetros definidos na tabela abaixo.

Cenário 2: Neste cenário procuramos variar na substituição dos *outliers* e *missing values*. Desta vez, escolhemos a média.
Neste cenário, os valores dos parâmetros referentes aos erros revelam se menores sendo que podemos concluir que este cenário é melhor que o anterior.

Cenário 3: Neste cenário procuramos variar na substituição dos *outliers* e *missing values*. Desta vez, escolhemos a mediana.
Neste cenário, os valores dos parâmetros referentes aos erros aumentam ligeiramente, sendo que a previsão deste cenário não é tão boa.

Cenário 4: Neste cenário procuramos variar na substituição dos *outliers* e *missing values*. Desta vez, escolhemos remover os mesmos, apesar de não ser aconselável, pois estamos a perder dados.
Neste cenário, os valores dos parâmetros referentes aos erros aumentam também sendo que podemos chegar a conclusão que este cenário não tem menor previsão que o cenário 2.

Cenário 5: Neste cenário, retiramos o Normalizer de forma a não normalizarmos o modelo. Assim, os valores não serão colocados no mesmo intervalo de valores fazendo com que o KNIME considere os mais altos como mais importantes.
Neste cenário, os valores dos parâmetros referentes aos erros disparam, atingindo até valores que triplicam os erros obtidos no cenário 2.

	Cenário 1	Cenário 2	Cenário 3	Cenário 4
Alteração		Missing values e outliers substituídos pela média	Missing values e outliers substituídos pela mediana	Missing values e outliers removidos
R^2	0,098	0,113	0,09	0,115
Mean absolute error	0,156	0,152	0,158	0,161
Mean squared error	0,037	0,035	0,039	0,04
Root mean squared error	0,193	0,188	0,197	0,199
Mean signed difference	-0,009	0,016	-0	-0,003
Mean absolute percentage error	0,591	0,639	0,611	0,665
Adjusted R^2	0,098	0,113	0,09	0,115
	Cenário 5	Cenário 6		
Alteração	Sem normalizar	Normalização com Z-score		
R^2	0,161	0,102		
Mean absolute error	3,602	0,801		
Mean squared error	19,989	0,968		
Root mean squared error	4,471	0,984		
Mean signed difference	-0,332	-0,072		
Mean absolute percentage error	0,656	2,556		
Adjusted R^2	0,161	0,102		

Tabela 8: Resultados dos cenários

4.7 Cross validation

De forma a conseguirmos uma análise mas detalhada decidimos aplicar *cross validation* ao nosso melhor cenário, isto é, ao cenário 2.

É importante referir utilizamos 25 *folds*, ou seja, 25 no número de validações. Na verdade, fomos variando os valores sendo este último o melhor e portanto o escolhido para esta tabela:

Modelos			
	Linear Regression	Random Forest	Gradient Boosted Trees
R^2	0,106	0,226	-0,169
Mean absolute error	3,594	3,334	3,436
Mean squared error	19,857	17,178	18,447
Root mean squared error	4,456	4,145	4,295
Mean signed difference	0,023	0,023	-0,127
Mean absolute percentage error	1305176302,363	1114321129,208	0,355
Adjusted R^2	0,106	0,226	-2,13

Tabela 9: Resultados dos modelos com cross validation

Conseguimos visualizar em todos os modelos que as métricas aumentam, tornando os erros maiores. Desta forma, os modelos não parecem ser mais robusto com o *cross validation*.

4.8 Random sampling

Ainda no cenário 2, procuramos também utilizar o *random sampling* em vez do *Stratified sampling* no node *X-Particioner* de forma a fazer variar os resultados mais uma vez.

A principal diferença entre os dois métodos é que o *random sampling* não considera a estrutura subjacente da população. Desta forma, este último pode resultar em uma amostra que não seja representativa de subgrupos importantes da população, enquanto que o *Stratified sampling* garante a representação proporcional desses subgrupos.

Modelos			
	Linear Regression	Random Forest	Gradient Boosted Trees
R^2	0,112	0,256	0,26
Mean absolute error	3,573	3,262	3,238
Mean squared error	19,704	16,502	16,423
Root mean squared error	4,439	4,062	4,053
Mean signed difference	0	0,006	-0,074
Mean absolute percentage error	1,308376374,876	107315520,596	1,019 659 872,215
Adjusted R^2	0,112	0,256	0,26

Tabela 10: Resultados dos modelos com Random Sampling

Desta forma, conseguimos perceber que o *Random Sampling* tem melhores resultados, pois apesar de variação ser ligeira, notamos uma diminuição dos erros comparativamente aos resultados da tabela 8.

4.9 Conclusões e Análise crítica dos resultados do dataset escolhido

Analisando os diferentes cenários de pré-processamento de dados antes da aplicação de outro modelo de previsão, podemos observar as seguintes tendências:

O Cenário 1 é o mais completo, abordando diversas técnicas de pré-processamento. Os parâmetros de erro são os menores, indicando uma boa qualidade de previsão. No entanto, com a análise percebemos que a previsão realizada pelo cenário 2 revela-se ainda melhor.

Nos Cenários 3 e 4, onde os outliers e valores ausentes são substituídos pela mediana e removidos, respetivamente, observamos que as métricas relativas ao erro aumentam, sugerindo que essas abordagens podem não ser ideais para a previsão.

Nos Cenários 5 e 6, onde o Normalizer é removido e o String to Number é utilizado, respectivamente, observamos um grande aumento nos parâmetros de erro. Isso sugere que a normalização dos dados e a abordagem de conversão de strings para números podem ser importantes para a qualidade da previsão.

No cenário 2, conseguimos valores baixos nos parâmetros de erro. Assim, podemos concluir que o processamento realizado no cenário 2 é o mais adequado obtendo-se as melhores previsões. O cenário 2 proporciona uma boa qualidade de previsão com baixos parâmetros de erro.

5 Conclusão e trabalhos futuros

Após a conclusão deste projeto, é essencial refletir sobre os desafios enfrentados e os resultados alcançados. O trabalho realizado proporcionou uma imersão valiosa na plataforma KNIME, permitindo uma exploração das suas funcionalidades.

Durante o processo de pesquisa e análise, encontramos uma dificuldade em equilibrar a atenção dedicada aos diferentes conjuntos de dados. Inicialmente, houve uma concentração significativa no dataset fornecido, o que resultou em um desequilíbrio no tratamento dos dados. No entanto, reconhecemos que essa experiência serviu como um aprendizado importante para futuros projetos, onde a gestão equitativa dos dados será uma prioridade.

Além disso, reconhecemos a importância de uma abordagem crítica ao nosso trabalho. Consolidamos os conhecimentos adquiridos na disciplina de ADI, especialmente no que diz respeito ao tratamento de dados e desenvolvimento de modelos de Machine Learning.

Em termos de organização e metodologia, seguimos de perto o framework CRISP-DM, garantindo que nosso trabalho fosse estruturado e coerente com as etapas estabelecidas. Os modelos implementados foram abrangentes e a exploração realizada respondeu eficazmente aos desafios apresentados pelos datasets.

Em resumo, fazemos um balanço positivo. Superamos as dificuldades encontradas, cumprimos os requisitos propostos e adquirimos um entendimento mais profundo das ferramentas e técnicas utilizadas. Este projeto serviu não apenas como uma oportunidade de aplicar conceitos aprendidos em sala de aula, mas também como um catalisador para o nosso crescimento profissional contínuo.

6 Referências

https://www.kaggle.com

https://hub.knime.com/knime/extensions/org.knime.features.base/latest

https://www.datascience-pm.com/crisp-dm-2/

https://www.kdnuggets.com/2022/02/random-forest-decision-tree-key-differences.html

https://www.geeksforgeeks.org/advantages-and-disadvantages-of-logistic-regression/

https://medium.datadriveninvestor.com/logistic-regression-essential-things-to-know-a4fe0bb8d10a

https://www.evidentlyai.com/classification-metrics/accuracy-precision-recall

https://neptune.ai/blog/balanced-accuracy

https://pt.khanacademy.org/math/statistics-probability/describing-relationships-quantitative-data/scatterplots-and-correlation/a/correlation-coefficient-review

https://www.scribbr.com/statistics/correlation-coefficient/

https://towardsdatascience.com/feature-engineering-deep-dive-into-encoding-and-binning-techniques-5618d55a6b38

https://developers.google.com/machine-learning/data-prep/transform/normalization?hl=pt-br

https://towardsdatascience.com/which-models-require-normalized-data-d85ca3c85388