Escola de Engenharia da Universidade do Minho
Departamento de Informática
Mestrado em Matemática e Computação
Mestrado Integrado em Engenharia Informática

Árvores de Decisão

Joel Morais, A70841
Luísa Caldas, PG35974
Tiago Fraga, A74092
Valéria Romanciuc, PG33724

18 de novembro de 2018

Conteúdo

1 Introdução **2**

2 *Dataset* da intensidade de trânsito **3**

2.1 Análise preliminar dos dados 3

2.2 Tratamento de Dados 5

2.2.1 *Feature Selection* 7

2.3 *Workflows* utilizados 8

2.3.1 *K-fold Cross-validation* 8

2.3.2 *Tuning* 10

2.4 Modelos desenvolvidos e resultados obtidos 12

2.5 Submissão no *Kaggle* 14

3 *Dataset* de emissões de CO2 **16**

3.1 Contextualização 16

3.2 Análise preliminar dos dados 16

3.3 Tratamento de Dados 18

3.3.1 Escolha de atributos 19

3.4 *Workflows* utilizados 19

3.4.1 *K-fold Cross-validation* 19

3.4.2 *Tuning* 20

3.5 Modelos desenvolvidos e resultados obtidos 21

4 Conclusão e trabalho futuro **24**

Capítulo 1

Introdução

Este trabalho foi realizado no âmbito da UC de Sistemas Baseados em Similaridade, com o objetivo de desenvolver modelos preditivos através de Árvores de Decisão. Para isso, foram considerados dois *datasets*: um para a modelação de tráfego na cidade de Braga e outro para a previsão das emissões de CO2 de veículos. O modelo desenvolvido para o primeiro *dataset* fez, ainda, parte de uma competição na plataforma *Kaggle*.

A elaboração destes modelos dividiu-se em algumas etapas, nomeadamente: a análise exploratória de dados, o pré-processamento (incluindo a seleção de atributos relevantes), a validação, o *tuning* e testes. Este processo foi desenvolvido na plataforma KNIME.

Os objetivos para este trabalho passavam, ainda, pela familiarização com os contextos subjacentes aos *datasets* em estudo, pela aplicação de Árvores de Decisão para a modelação dos problemas utilizando o KNIME, e pelo desenvolvimento da análise crítica dos resultados obtidos.

Capítulo 2

Dataset da intensidade de trânsito

2.1 Análise preliminar dos dados

Comecemos com uma breve análise do *dataset* a ser tratado. Este *dataset* contém dados relativos ao tráfego em Braga, compreendido entre 24 de julho e 11 de outubro de 2018 e inclui os seguintes atributos (retirado de [1]):

- **city_name:** nome da cidade em causa (neste caso, apenas a cidade de Braga);
- **record_date:** o *timestamp* do registo;
- **average_confidence:** confiança nos valores do registo;
- **average_free_flow_speed:** a velocidade máxima que os carros podem atingir em cenários sem trânsito;
- **average_speed_diff:** a diferença de velocidade corresponde à diferença entre (1.) a velocidade máxima que os carros podem atingir em cenários sem trânsito e (2.) a velocidade que realmente se verifica num dado momento. Quanto mais alto o valor, maior é a diferença entre a velocidade no momento e a velocidade sem trânsito, i.e., valores altos deste atributo implicam que se está a andar mais devagar;
- **average_free_flow_time:** o valor médio do tempo que se demora a percorrer um determinado conjunto de ruas quando não há trânsito;
- **average_time_diff:** o valor médio da diferença do tempo que se demora a percorrer um determinado conjunto de ruas. Quanto mais alto o valor, maior é a diferença entre o tempo que demora para se percorrer as ruas e o que se deveria demorar sem trânsito, i.e., valores altos implicam que se está a demorar mais tempo a atravessar o conjunto de ruas;
- **average_temperature:** o valor médio da temperatura para o **record_date**;

- **average_atmosp_pressure:** o valor médio da pressão atmosférica para o **record_date**;
- **average_humidity:** o valor médio da humidade para o **record_date**;
- **average_wind_speed:** o valor médio da velocidade do vento para o **record_date**;
- **average_cloudiness:** o valor médio da percentagem de nuvens para o **record_date**;
- **average_rain:** o valor médio de precipitação para o **record_date**.

Importa destacar o atributo "Average_Speed_Diff", que indica, na escala qualitativa ("None", "Low", "Medium", "High"e "Very High"), qual a intensidade de trânsito num dado momento. O nosso objetivo é prever o trânsito utilizando esta escala.

No *training set*, podemos verificar que este atributo tem a seguinte distribuição:

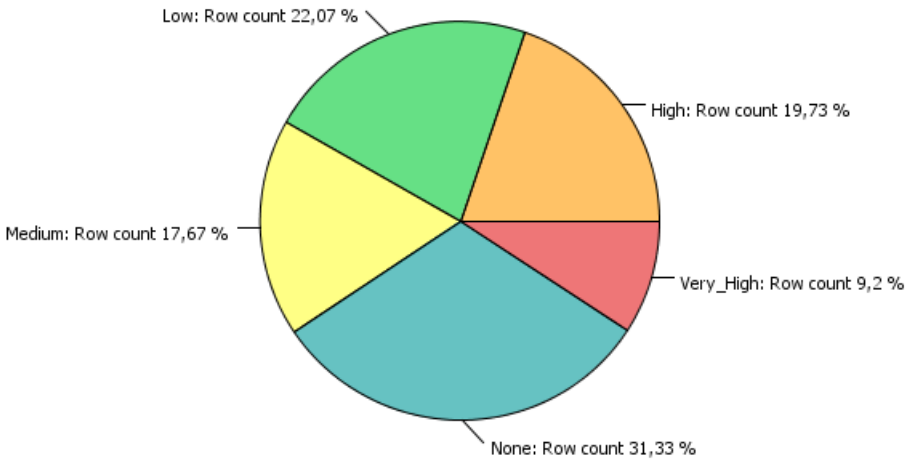

Figura 2.1: Distribuição do atributo "Average_Speed_Diff".

Pelo gráfico acima, podemos observar que o valor "None" é aquele que tem o maior número de ocorrências, representando cerca de 31% das observações, seguindo-se do "Low", com cerca de 22%. Os valores de "Medium" e "High" ocorrem com uma frequência semelhante (aproximadamente 18% e 20%, respetivamente) e, por fim, o valor com menor número de ocorrências é o "Very High", contando com cerca de 9% das observações. Estes dados vão ao encontro da nossa experiência diária, uma vez que as situações com maior congestionamento ocorrem maioritariamente em dois momentos do dia (uma no início da manhã e a outra ao final da tarde), sendo que nas restantes alturas observamos pouco ou nenhum trânsito.

Um atributo fortemente relacionado com este é o "Average_Time_Diff", como podemos observar pelo seguinte gráfico de dispersão.

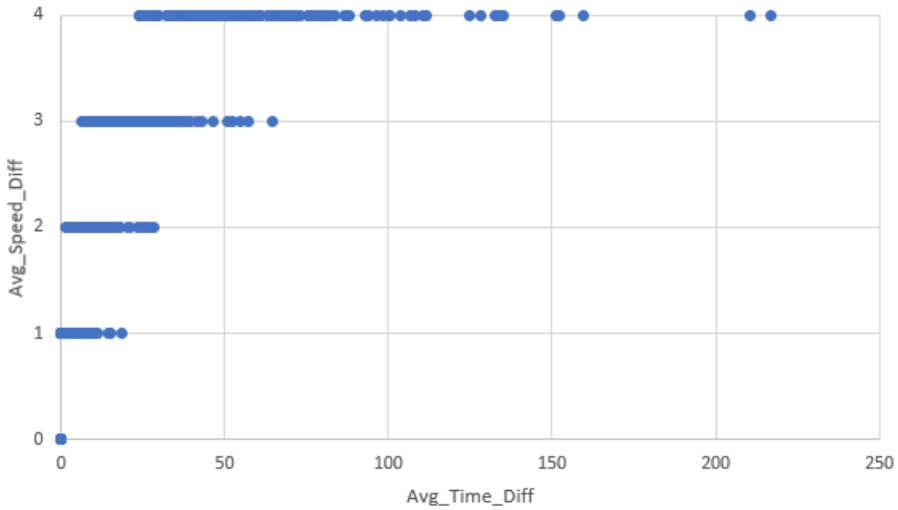

Figura 2.2: Gráfico de dispersão para os atributos "Average_Speed_Diff"e "Average_Time_Diff".

Com isto, concluímos que, em média, quanto mais elevada a "Average_Time_Diff", maior será a "Average_Speed_Diff". Mais uma vez, isto corresponde às nossas expectativas, uma vez que, intuitivamente, quanto mais elevada a diferença entre o tempo que levamos a percorrer um conjunto de ruas e o tempo que levaríamos sem trânsito, maior será o congestionamento naquele local.

Relativamente aos restantes atributos numéricos, começámos por verificar o intervalo de variação de cada *feature*, bem como a sua média e desvio padrão, de forma a identificar as características e possíveis irregularidades do *dataset*. Estes dados são apresentados na Figura 2.3. Como podemos observar, o atributo "Average_Rain" apresenta sempre o mesmo valor (zero), pelo que será pertinente excluí-lo *a priori* do modelo.

2.2 Tratamento de Dados

Numa primeira fase, antes de podermos testar o nosso modelo, temos de fazer um tratamento dos dados, isto é, uma análise crítica que nos permita identificar características importantes do *dataset* e quais os atributos que são essenciais no nosso modelo.

Antes de passarmos os dados ao modelo, começámos por extrair algumas informações do atributo "record_date", nomeadamente o dia da semana ("Day of week") e a hora do dia ("Hour") a que o

Column	Min	Max	Mean	Std. deviation
AVERAGE_CONFIDENCE	0.5	0.95	0.798	0.125
AVERAGE_FREE_FLOW_SPEED	37	46.5	40.545	1.698
AVERAGE_TIME_DIFF	0	217	11.748	21.064
AVERAGE_FREE_FLOW_TIME	50.1	86.6	64.928	4.385
AVERAGE_TEMPERATURE	7	35	19.468	4.496
AVERAGE_ATMOSP_PRESSURE	998	1,026	1,017.063	3.656
AVERAGE_HUMIDITY	14	100	76.948	20.808
AVERAGE_WIND_SPEED	0	10	2.328	1.805
AVERAGE_CLOUDINESS	0	92	24.022	33.451
AVERAGE_RAIN	0	0	0	0

Figura 2.3: Estatísticas relacionadas com os diferentes atributos.

registo foi realizado. Além destas duas *features*, inicialmente adicionámos os atributos "Is_Weekday" e "Is_Weekend" para diferenciar os dias úteis dos fins de semana, mas chegámos à conclusão de que estes não influenciavam o nosso modelo.

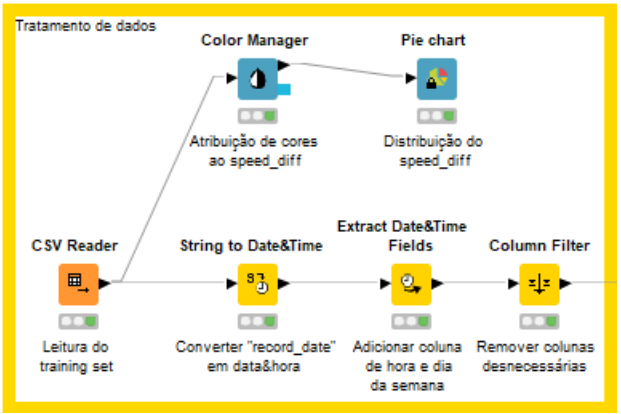

Figura 2.4: Tratamento de dados no *workflow*.

2.2.1 *Feature Selection*

De forma a escolher os atributos necessários para o nosso modelo, uma das estratégias a que recorremos foi utilizar a técnica de *Feature Selection*. Para isso, criámos um novo *workflow*, que podemos ver na Figura 2.5.

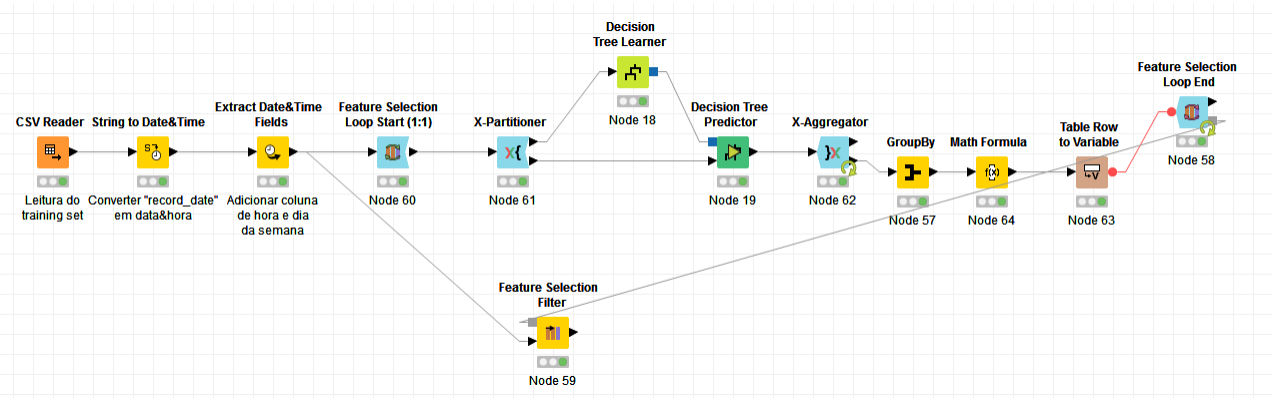

Figura 2.5: *Feature Selection workflow.*

Neste *workflow*, corremos várias vezes o modelo com os parâmetros *Gain ratio*, *MDL* e 7 registos mínimos no nodo "Decision Tree Learner", tendo sido estes os parâmetros com que obtivemos os melhores resultados.

Os resultados que obtivemos executando o modelo três vezes podem ser consultados na Tabela 2.1.

Iteração	**1**	**2**	**3**	Interseção
Accuracy	86,933%	87,133%	87,2%	-
Average_Confidence	x	x	x	x
Average_Free_Flow_Speed		x		
Average_Time_Diff	x	x	x	x
Average_Free_Flow_Time				
Average_Atmos_Pressure		x		
Average_Temperature	x			
Average_Humidity		x	x	
Average_Wind_Speed			x	
Average_Cloudiness	x	x	x	x
Day of week	x	x	x	x
Hour	x	x	x	x

Tabela 2.1: Resultados da execução do *Feature Selection* três vezes.

Depois de obtidos estes resultados, decidimos passar para o nosso modelo cada um dos conjuntos de colunas. Daqui, obtivemos os valores apresentados na Tabela 2.2.

	Resultados *Feature Selection*	*Resultados no nosso modelo*
1	86,933%	87,2%
2	87,133%	87%
3	87,2%	86,933%

Tabela 2.2: Comparação da *accuracy* dada pelo *Feature Selection* com a dada pelo modelo.

No caso da interseção, a *accuracy* obtida no nosso modelo foi de 86,8%.

Além de utilizarmos o *Feature Selection*, consideramos mais algumas estratégias, nomeadamente:

1) Considerar todos os atributos;
2) Excluir as colunas que são do tipo *string* ("city_name"e "record_date");
3) Excluir as colunas do tipo *string* e a coluna do average_rain.

Nestas tentativas, a *accuracy* foi de 87,2%, 86,733% e 87,133%, respetivamente. Tendo em conta todas as opções consideradas, optámos por escolher esta última alternativa como a final para o modelo, uma vez que apresentou uma *accuracy* dentro do mesmo valor que as outras mas achámos que os atributos que usa fazem mais sentido para o problema. Apesar de termos tentado utilizar o *Feature Selection*, este não provou ser muito relevante para o modelo em questão.

2.3 *Workflows* utilizados

Para este *dataset*, utilizámos dois *workflows*: um para otimizar algumas variáveis do modelo da Árvore de Decisão, nomeadamente a medida de qualidade, o método de *pruning* e o número mínimo de registos por nodo, e outro *workflow* para testarmos a *feature selection*.

2.3.1 *K-fold Cross-validation*

Cross-validation é um método usado para prever o desempenho de um modelo de *Machine Learning*. Este método divide os dados em várias partições de *training*/*test* (neste caso, em k partições) de forma a percorrer todos os dados e, no fim, retorna os erros da previsão em cada partição que usou para teste. Desta forma, obtemos uma estimativa mais estável e confiável uma vez que percorremos todo o conjunto de dados para treinar o modelo. Por sua vez, num modelo sem *cross-validation*, estamos apenas a utilizar uma partição dos dados [2]. No nosso trabalho usamos *k-fold cross-validation* com $k = 10$.

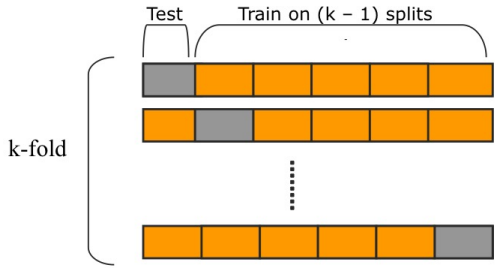

Figura 2.6: Visualização de *k-fold cross-validation*.

Uma das vantagens de usar *cross-validation* é diminuir o problema do *overfitting*, uma questão muito comum em *Machine Learning*. Este fenómeno ocorre quando o modelo memoriza e modela demasiado o *training set*, dando origem a previsões erradas quando recebe um novo conjunto de dados para prever [3]. Podemos visualizar este fenómeno na Figura 2.7.

Figura 2.7: Visualização de *overfitting*. Retirado de "Memorizing is not learning! - 6 tricks to prevent overfitting in machine learning.", Hackernoon, a 17.11.2018

Para aplicar este método no nosso *workflow*, utilizámos os nodos "X-Partitioner"e "X-Aggregator". O primeiro faz a repartição do *dataset* em 10 partes iguais, sendo o *sampling* aleatório. Já o segundo, seleciona as colunas a comparar (isto é, a coluna "alvo"e a da previsão) e retorna a tabela de previsões, bem como a percentagem de erro para cada *fold*.

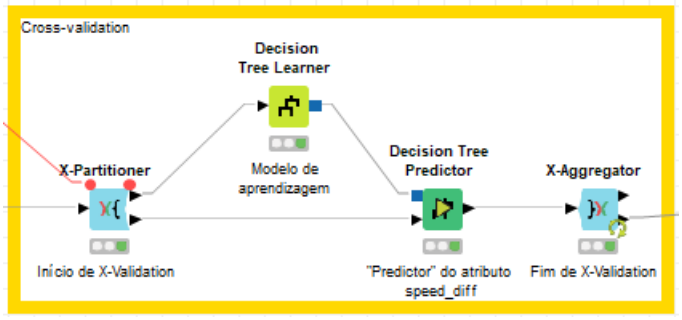

Figura 2.8: *Cross-validation* no *workflow*.

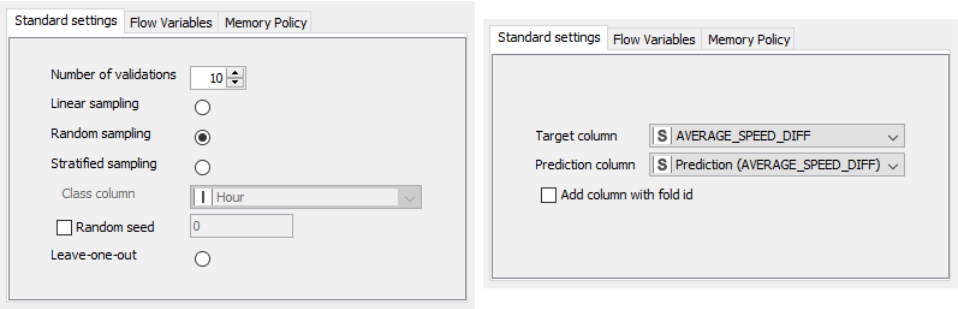

Figura 2.9: Definições dos nodos "X-Partitioner"e "X-Aggregator", respetivamente.

2.3.2 *Tuning*

O *tuning* foi feito a dois níveis: para parâmetros numéricos e para parâmetros nominais.

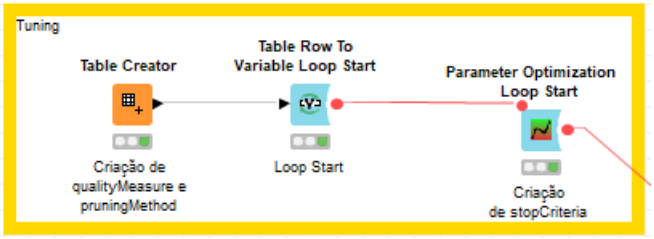

Figura 2.10: Primeira parte do *tuning* no *workflow*.

No que toca a parâmetros numéricos, o parâmetro otimizado foi o de número mínimo de registos por nodo. Para isso, utilizámos o *loop* "Parameter Optimization Loop", sendo que no nodo "Start" criámos uma variável que pudesse servir de critério de paragem, denominada por "stopCriteria" (ver Figura 2.11).

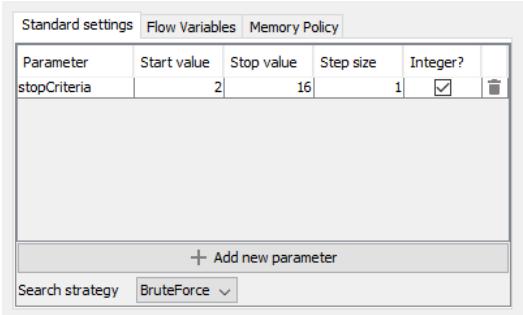

Figura 2.11: Definições do nodo "Parameter Optimization Loop Start".

Quanto aos parâmetros nominais, pretendíamos otimizar as variáveis "Quality Measure" (sendo possível utilizar o "Gini index" ou o "Gain ratio") e "Pruning Method" (sendo possível não utilizar nenhum método de *pruning*, ou utilizar o MDL). Os nodos utilizados no KNIME foram o "Table Creator" e o *loop* "Table Row to Variable Loop" (ver Figura 2.12). O primeiro, permitiu criar as variáveis "qualityMeasure" e "pruningMethod", que pretendemos otimizar, enquanto que o segundo permitiu transformar as variáveis mencionadas em variáveis de fluxo.

	S qualityMeasure	S pruningMethod
Row0	Gain ratio	No pruning
Row1	Gain ratio	MDL
Row2	Gini index	No pruning
Row3	Gini index	MDL

Figura 2.12: Definições do nodo "Table Creator".

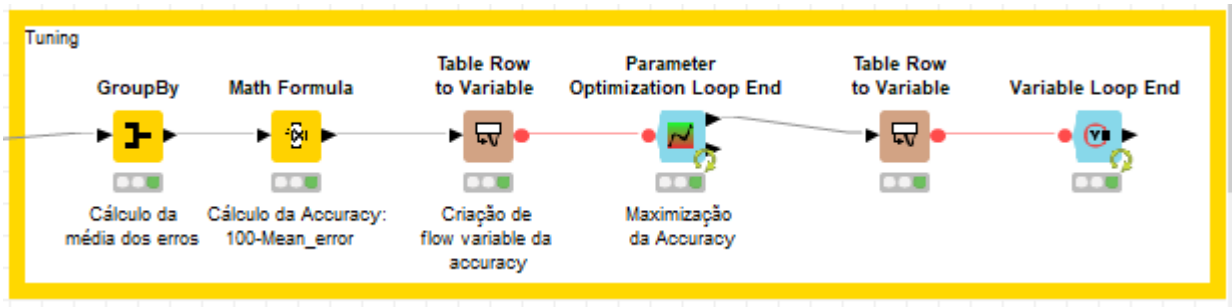

Figura 2.13: Segunda parte do *Tuning* no *workflow*.

Para o cálculo da *accuracy*, utilizámos as percentagens de erro obtidas no nodo "X-Aggregator" para primeiro calcular a média dos erros no nodo "GroupBy". De seguida, utilizámos um nodo "Math Formula"para o cálculo da $accuracy = 100 - mean\_error$.

No nodo "Parameter Optimization Loop End", definimos como objetivo a maximização da *accuracy*, passando-a ao nodo final "Variable Loop End" através de um "Table Row to Variable". Aqui, as variáveis a otimizar foram as que podem ser observadas na Figura 2.14.

Com isto, a melhor *accuracy* (de 87,133%) foi obtida para os seguintes valores dos parâmetros: 7 como número mínimo de registos por nodo e utilizando o "Gain ratio" como *quality measure* e MDL como método de *pruning*.

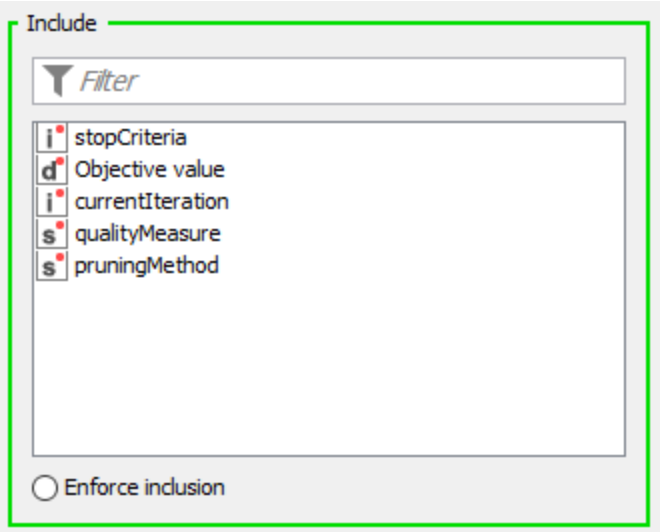

Figura 2.14: Parâmetros a otimizar no nodo "Variable Loop End".

Row ID	I stopCr...	D Objective...	I curr...	S qualityMea...	S pruningMe...
Row0	9	85.533	0	Gain ratio	No pruning
Row1	7	87.133	1	Gain ratio	MDL
Row2	15	85.733	2	Gini index	No pruning
Row3	6	87.067	3	Gini index	MDL

Figura 2.15: Resultados obtidos no nodo "Variable Loop End".

2.4 Modelos desenvolvidos e resultados obtidos

Ao longo do trabalho, testámos diferentes abordagens no KNIME antes de obtermos o modelo que apresentámos anteriormente como final. Nesta secção, os parâmetros do nodo "Decision Tree Learner"serão *Gain ratio*, *MDL* e 7 registos mínimos no nodo. Como colunas iremos usar as definidas anteriormente.

Numa primeira fase, começámos por considerar um modelo sem *cross-validation* e sem *tuning* (pode ser consultado na Figura 2.16). Este modelo, deu-nos uma *accuracy* de 83,7%.

Numa segunda fase, considerámos o modelo anterior mas acrescentando *cross-validation* (ver Figura 2.17). Aqui, a *accuracy* obtida foi de 86,667%. Podemos considerar este modelo melhor do que o primeiro, porque esta *accuracy* é dada tendo por base os erros de todo o *dataset*, ao contrário do primeiro que apenas é baseado nos dados para treino.

Para testar uma otimização do primeiro modelo, criámos um *workflow* em que fazemos uso do nodo "Parameter Optimization" (ver Figura 2.18).

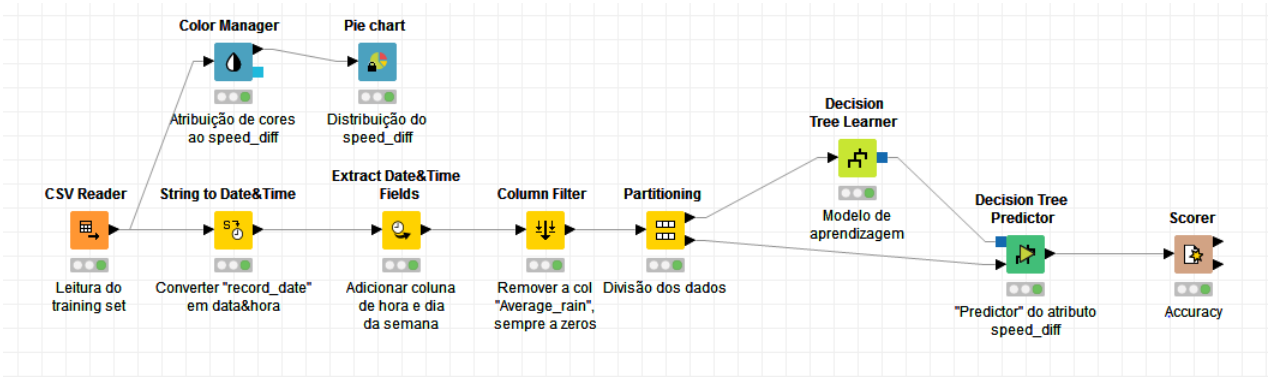

Figura 2.16: Primeiro modelo.

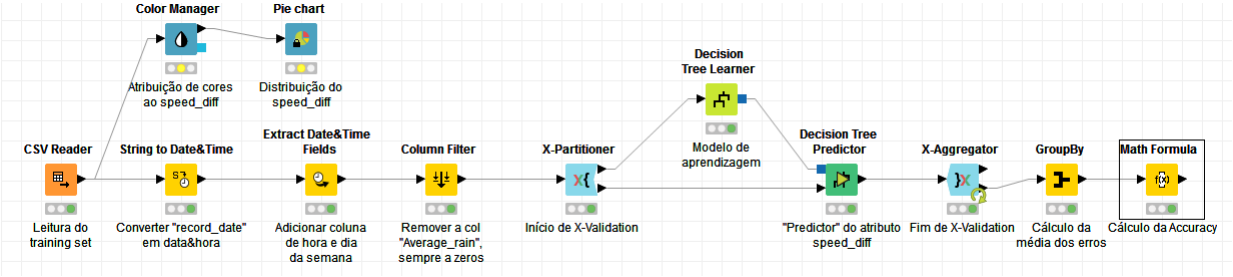

Figura 2.17: Segundo modelo.

Este modelo dá uma *accuracy* de 90,7%, que é uma melhoria signficativa em relação ao primeiro modelo. No entanto, é um pouco instável porque só tem em conta uma parte dos dados.

Dado isto, decidimos, por último, tentar juntar esta melhoria que nos dá o nodo "Parameter Optimization" com a segurança que nos dá a técnica de *cross-validation*. Podemos consultar este modelo na Figura 2.19.

Este modelo dá-nos uma *accuracy* de 87,133% (ver Figura 2.15). Apesar de ser mais baixa que a do modelo anterior, podemos dizer que é melhor que a do primeiro, uma vez que além de estar otimizada, também é mais confiável. Desta forma, esta foi a nossa escolha como modelo final, pois em sucessivas execuções do modelo, a *accuracy* não sofre grandes alterações, o que nos dá uma segurança maior.

Apesar de podermos fazer mais testes, pensámos que estes foram suficientes para criar o nosso modelo. Isto deve-se ao facto de sabermos que o modelo é todo percorrido para dar a melhor *accuracy* e que este também está otimizado para dar sempre os melhores parâmetros.

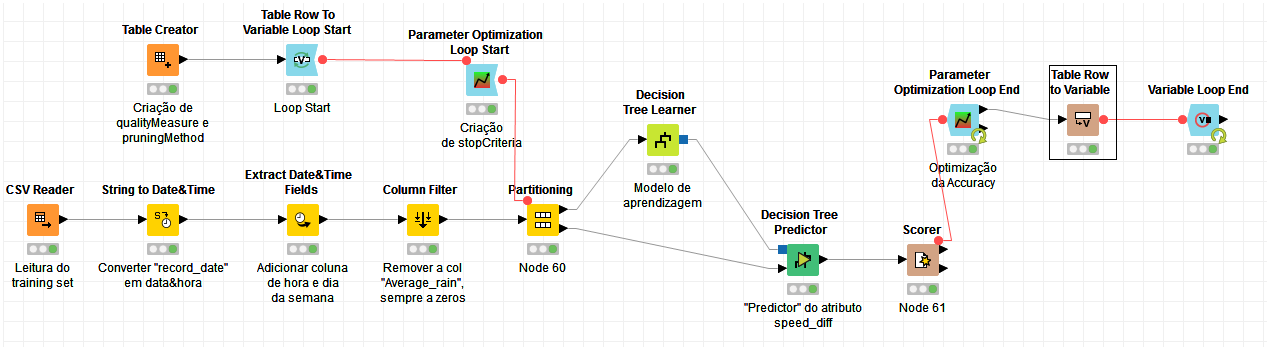

Figura 2.18: Terceiro modelo.

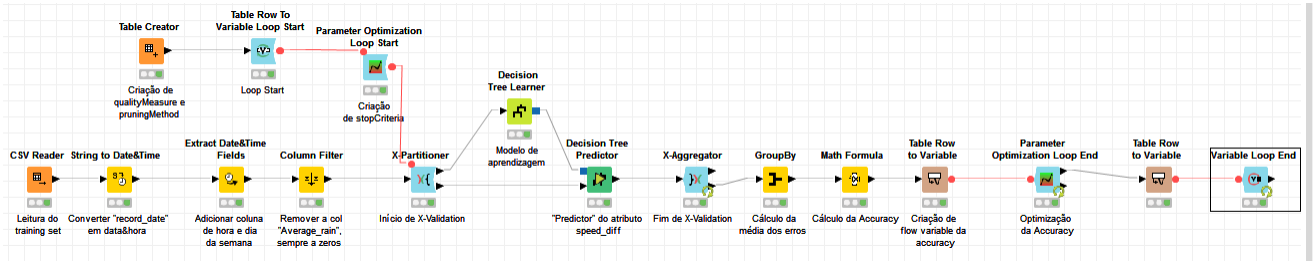

Figura 2.19: Quarto modelo.

2.5 Submissão no *Kaggle*

Tal como foi mencionado, o modelo elaborado teve de ser submetido numa competição na plataforma *Kaggle*, sendo que a avaliação da *accuracy* era medida de duas formas: uma feita com 30% dos dados de teste e outra com 70%. A pontuação obtida em cada uma delas contava para um *leaderboard* público e um *leaderboard* privado, respetivamente.

Para o CSV submetido, foi necessário fazer ajustes ao *output* gerado inicialmente pelo nosso modelo, de forma a estar de acordo com o exemplo deixado na competição. Para isto, foi necessário criar no nosso *workflow* uma coluna com a identificação da linha de previsão ("RowId"), reajustar a contagem (que, por omissão, começava em 0 e não em 1, como pretendido) e reordenar as colunas. O *workflow* completo pode ser visto na Figura 2.20.

A submissão do nosso modelo obteve uma *accuracy* 89,09% no *leaderboard* público e 86,821% no privado.

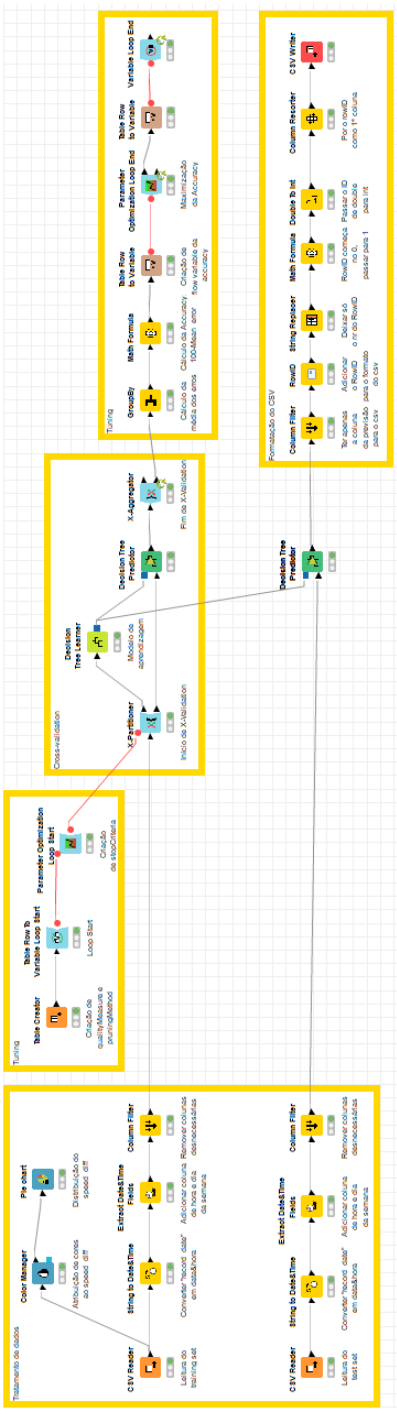

Figura 2.20: *Workflow* completo no KNIME.

Capítulo 3

Dataset de emissões de CO2

3.1 Contextualização

A preocupação crescente da população mundial com a *emissão de gases* poluentes para a atmosfera, aumentando assim o efeito de estufa, é um tema atual e cada vez mais atual e presente no nosso quotidiano.

Os recentes casos de manipulação dos resultados de emissões de CO2 por parte de algumas grandes marcas mundiais, bem como o gosto e a paixão pelo mundo automóvel, serviu como motivação para a escolha deste tema e elaboração de um modelo de previsão da emissão de CO2 no mundo automóvel.

Consideramos que com a construção deste modelo de *Machine Learning*, e com a obtenção de dados de alguns veículos, é possível prever os níveis de gases que estes emitem, evitando assim a adulteração dos resultados por parte de alguns agentes maliciosos, como pudemos assistir nestes últimos anos.

3.2 Análise preliminar dos dados

No presente *dataset* foi possível tratar os dados que em seguida enumeramos:

- **Manufacturer:** Nome do fabricante do veículo;
- **Model:** Modelo do veículo;
- **Description:** Descrição comercial sobre a motorização;
- **Transmission:** Tipo de transmissão utilizada, i.e, como funcionam as mudanças no veículo;
- **Engine Capacity:** A cilindrada do veículo em centímetros cúbicos;

- **Fuel type:** Tipo de combustível. Como estamos a tratar o nível de emissões de *CO2*, para este estudo, apenas foram considerados os veículos movidos a gasolina, diesel e híbridos.
- **Metric Urban(Cold):** Consumo médio de combustível apenas em cidade. Esta métrica é apresentada em litros gasto por cada 100km percorridos;
- **Metric Extra-Urban:** Consumo médio de combustível apenas fora da cidade (auto-estrada). Esta métrica é apresentada em litros gasto por cada 100km percorridos;
- **Metric combined:** Consumo médio de combustível misto, isto é, em cidade e auto-estrada. Esta métrica é apresentada em litros gasto por cada 100km percorridos;
- **CO2 g/km:** Nível de gás *CO2* no motor. Este atributo é apresentado em gramas por quilometro;
- **Fuel Cost:** Custo total de combustível em 20000 quilómetros;
- **Noise Level db(A):** Nível de barulho do motor. Este atributo é apresentado em decibéis.
- **Emissions CO2[mg/km]:** Quantidade de emissões de *CO2* emitidas pelo veículo. Esta métrica é apresentada em miligramas por quilometro;
- **Emissions NOx [mg/km]:** Quantidade de emissões de gases de nitrogénio. Esta métrica é apresentada em miligramas por quilometro;
- **THC + NOx Emissions [mg/km]:** Quantidade total de gases hidrocarbonetos e gases de nitrogénio. Esta métrica é apresentada em miligramas por quilometro;
- **Particulates[No.][mg/km]:** Quantidade de particular de nitrogénio do veículo. Esta métrica é apresentada em miligramas por quilometro;

O nosso objetivo será prever as emissões de CO2, sendo, para isso, necessário enfatizar o atributo Emissions CO2[mg/km]. Visto termos valores inteiros bastante dispersos, foi necessário aglomera-los em vários **Bins**.

Como é possível ver pela Figura 3.1, decidimos agrupar este atributo em quatro conjuntos. Como no *dataset* anterior o atributo de previsão de cálculo estava agrupado em cinco conjuntos, decidimos reduzir em um o número de conjuntos de maneira a poder verificar também de que forma este número influencia a variação da previsão final do trabalho. A divisão que efetuamos pode ter várias interpretações, sendo umas delas designar os *bins* por: **muito baixo**, **baixo**, **alto** e **muito alto**. Esta designação pode estar inserida numa legislação que pretenda reduzir ao máximo o numero de emissões de CO2. Por exemplo, desta forma, apenas os dois primeiros conjuntos eram aceites.

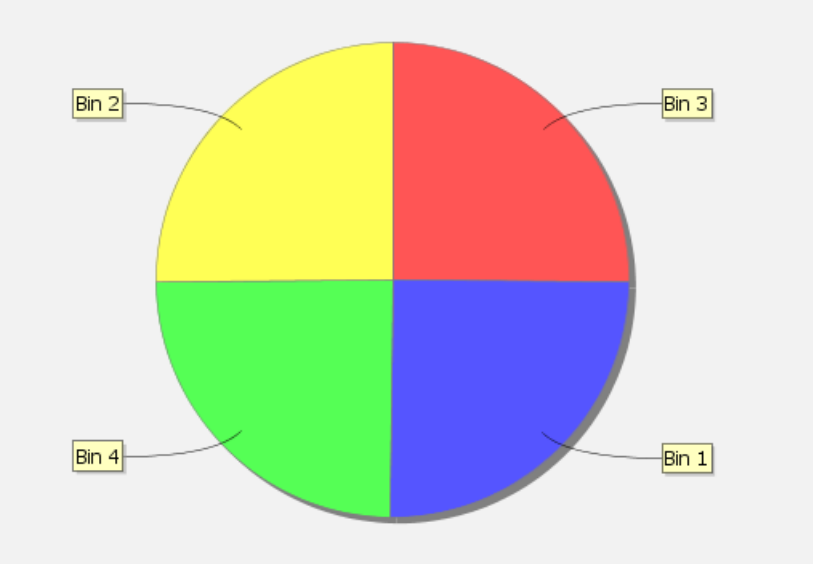

Figura 3.1: Distribuição do atributo Emissions CO2[mg/km]:

3.3 Tratamento de Dados

Foi necessário fazermos um tratamento prévio, antes de podermos começar a testar o nosso *dataset*, visto que não faria sentido a inclusão de alguns dos atributos.

Sendo que o nosso objetivo é prever a emissão de CO2, existem certos atributos que iremos eliminar logo à partida, como o "Manufacturer", "Model", "Description", "Electric energy consumption Miles/kWh", "wh/km", "Maximum range(Km)", "Maximum range(Miles)", "Imperial Urban(Cold)", "Imperial Extra-Urban", "Imperial Combined", "Fuel Cost", "Electricity Cost", "Total Cost"e "Euro Standard".

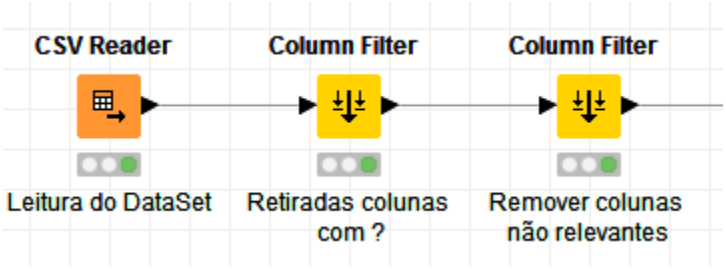

Figura 3.2: Tratamento de dados no *workflow*

3.3.1 Escolha de atributos

O processo de escolha de atributos ocorreu de forma fácil, uma vez que a escolha do tema foi sobre o tema que pretendêssemos. Decidimos escolher um *dataset* sobre um tema que nos sentíssemos à vontade para trabalhar e fazer a escolha dos atributos de uma maneira mais rápida e fácil.

De forma a obter um maior nível da *accuracy*, decidimos manter todos os atributos que estão diretamente ligados com as emissões de gases CO2, ou seja, todos os atributos que indicavam os níveis de diferentes gases no motor, bem como a cilindrada do veículo, o gasto médio de combustível e o tipo de combustível usado na alimentação da combustão do mesmo.

Em suma, os atributos usados para calcular a previsão da emissão de gases de CO2 foram:

- **Engine Capacity;**
- **Fuel type;**
- **Metric Urban(Cold);**
- **Metric Extra-Urban;**
- **Metric combined;**
- **CO2 g/km;**
- **Noise Level db(A);**
- **Emissions NOx [mg/km];**
- **THC + NOx Emissions [mg/km];**
- **Particulates[No.][mg/km];**

3.4 *Workflows* utilizados

Tal como no *dataset* anterior, e visto que obtemos uma *accuracy* bastante precisa, decidimos voltar a utilizar o mesmo tipo de *workflow*, um que irá otimizar as nossas novas variáveis da Árvore de Decisão.

3.4.1 *K-fold Cross-validation*

Tal como foi mencionado anteriormente, uma das vantagens de usar o *cross-validation* consiste em diminuir o problema do *overfitting*.

Para conseguirmos usar o *cross-validation*, foi utilizado o "X-Partitioner", que irá partir o nosso *dataset* em 10 partes iguais com os devidos dados aleatórios. Além deste, foi utilizado o "X-Aggregator", sendo que neste caso, a nossa coluna alvo será a "Emissions CO [mg/km]", como

podemos observar na seguinte figura. No fim, irá ser retornada a tabela de previsões, tal como a percentagem de erro de cada *fold*.

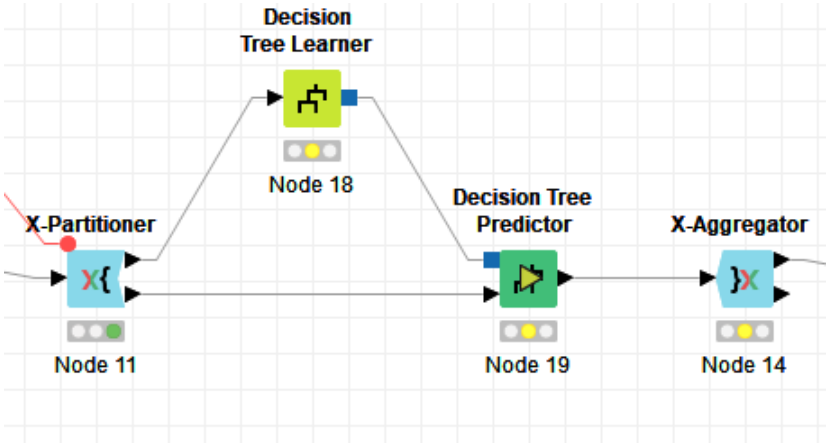

Figura 3.3: *Cross-validation* no *workflow*

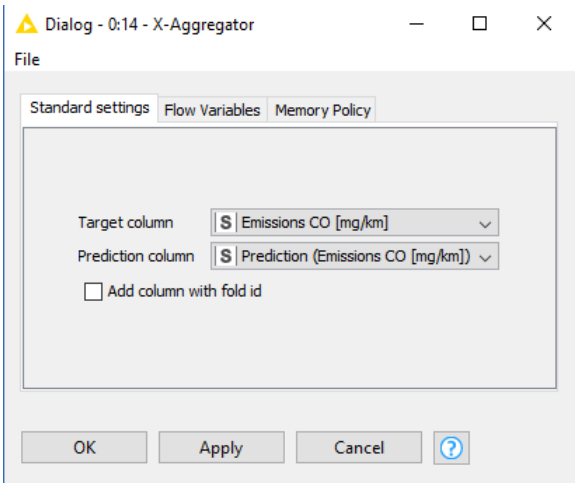

Figura 3.4: Definição do nodo X-Aggregator

3.4.2 *Tuning*

Como podemos observar pela seguinte figura, o *tuning* implementado neste *workflow* é semelhante ao utilizado no *dataset* anterior.

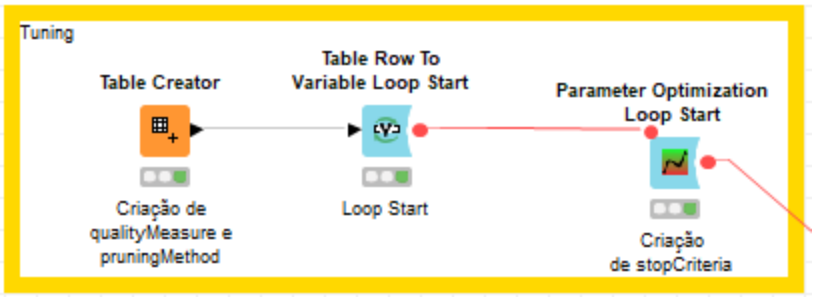

Figura 3.5: *Tuning* no *workflow*.

Foi também criada uma variável de paragem denominada **stop** no *loop* utilizando o "Parameter Optimization Loop".

Já no caso dos parâmetros nominais, as variáveis que pretendíamos otimizar são a "Quality Measure"e o "Pruning Method". Ambas as variáveis foram criadas através da utilização do nodo "Table Creator", sendo que o "Table Row to Variable Loop"está encarregue de passar essas variáveis para variáveis de fluxo.

De forma a calcular a *accuracy*, começamos por recorrer ao nodo "GroupBy"de modo a podermos calcular a média dos erros. De seguida, utilizámos o nodo "Math Formula" com o objetivo de calcular a *accuracy* através da expressão matemática *100-mean_error*. Posto isto, foi necessário a criação da *flow variable* da *accuracy*, através da utilização do nodo "Table Row to Variable".

Por fim, no momento do tratamento da *accuracy* definimos, no nodo "Parameter Optimization Loop End"a maximização da mesma, passando-a ao nodo final "Variable Loop End"através de um nodo "Table Row to Variable". Conforme é possível ver na Figura 3.7, os melhores resultados obtidos para a previsão do nosso modelo, foram:

Row ID	I stop	D Objecti...	S RowID	I current...	I maxIte...	S quality...	S pruning...
Row0	2	87.686	Best parameters	0	4	Gain ratio	No pruning
Row1	3	80.067	Best parameters	1	4	Gain ratio	MDL
Row2	2	87.175	Best parameters	2	4	Gini index	No pruning
Row3	6	78.714	Best parameters	3	4	Gini index	MDL

Figura 3.6: Resultados obtidos no nodo "Variable Loop End".

3.5 Modelos desenvolvidos e resultados obtidos

Apesar do modelo aplicado a este *dataset* derivar do modelo do capítulo anterior, testamos novamente os mesmos processos até chegar ao produto final.

Em primeiro lugar, cometemos o erro de construir um modelo cuja previsão era baseada em valores inteiros de emissões. Após alguma análise, partimos para a construção do modelo de previsão com auxilio de **Bins**, assim como é explicado na secção *3.1.*

Demos seguimento à realização do trabalho com um modelo básico sem *cross-validation* e *tuning.* Deste modo, utilizamos os conceitos iniciais da Unidade Curricular, elaborando um modelo suportado em "Decision Trees". Como no capitulo anterior, obtivemos bons resultados, e com o objetivo de aumentar a *accuracy* foi possível chegar ao modelo final descrito ao longo deste capítulo.

Em suma, conseguimos atingir valores de previsão de 87,686% que consideramos bastante positivo.

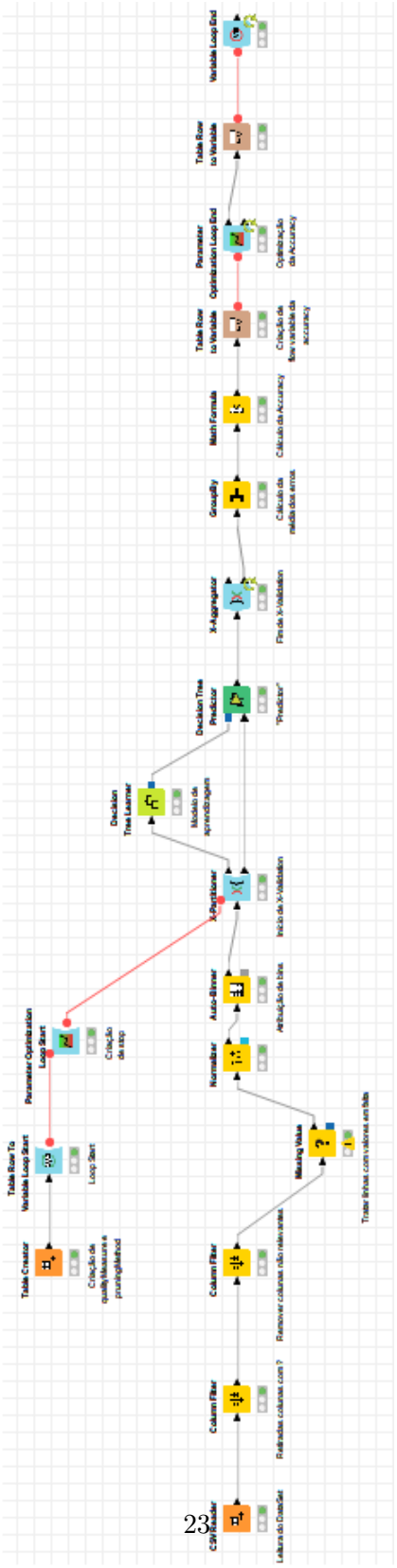

Figura 3.7: *Workflow* completo no KNIME..

Capítulo 4

Conclusão e trabalho futuro

Ao longo deste trabalho, fomos testando diferentes funcionalidades no KNIME, que nos permitiram tirar algumas conclusões, que apresentamos a seguir. A primeira é que, ao utilizarmos *cross-validation*, obtemos um modelo mais fidedigno, uma vez que este faz uso de todos os dados para o cálculo da *accuracy*. A segunda é que usando diferentes técnicas para fazer *tuning* ao modelo, a *accuracy* é melhorada substancialmente, pois otimiza os parâmetros desejados. Por último, concluímos que o *Feature Selection* não provou ser um método relevante no nosso modelo, pois não melhora o sistema como era suposto.

Relativamente ao primeiro *dataset*, modelação de tráfego na cidade de Braga, decidimos como modelo final o modelo que usa *cross-validation* e *tuning*. Este modelo dá-nos uma *accuracy* de 87,133% no KNIME e de 89,09% na *leaderboard* pública e de 86,821% na *leaderboard* privada do *Kaggle*.

No segundo *dataset*, conseguimos estudar um tema atual, o de emissões de CO2 de veículos. O modelo elaborado também faz uso de *cross-validation* e *tuning*, tendo apresentado uma *accuracy* de 87,686%.

Este trabalho permitiu ver algumas vantagens da utilização de Árvores de Decisão em modelos de previsão, nomeadamente a fácil interpretação de resultados. Porém, estas também apresentam algumas desvantagens, tais como a volatilidade do algoritmo, uma vez que, como pudemos observar, uma pequena variação em certos parâmetros produz resultados diferentes. Além disso, para que este algoritmo fosse aplicável ao segundo *dataset*, foi necessário adaptar os dados (transformar uma variável numérica em nominal).

Como trabalho futuro, seria interessante aplicar outros algoritmos de classificação a cada um dos problemas, nomeadamente o de "Random Forest", uma vez que está diretamente relacionado com "Decision Trees".

Bibliografia

[1] Trabalho Prático 2 - Competição para grupos pares (Traffic Flow). https://www.kaggle.com/c/sbs2p2018/data. Consultado a 10-11-2018.

[2] Kevyn Collins-Thompson. Applied machine learning in python. https://www.coursera.org/learn/python-machine-learning/lecture/Vm0Ie/cross-validation. Consultado a 10-11-2018.

[3] Philippe Esling and Carlos Agon. Time-series data mining. *ACM Computing Surveys (CSUR)*, 45(1):12, 2012.