Universidade do Minho

Universidade do Minho

Sistemas baseados em Similaridade

Trabalho Prático Nº1

Autores:
Luís Freitas, PG38347
Daniel Pereira, PG42821
João Queirós, A82422
Pedro Queirós, A84220

Grupo:
7

25 de junho de 2021

Conteúdo

1 Introdução

Este trabalho foi realizado no âmbito da cadeira de *Sistemas Baseados em Similaridade*, que se enquadra no 1º semestre do 1º ano do Mestrado em Engenharia Informática da Universidade do Minho.

Este trabalho possui 2 componentes distintos, abrangindo 2 *datasets* diferentes:

1. O 1º: Um dataset comum a todos os grupos, onde o objetivo passa pela exploração e pre-processamento dos dados, para posteriormente criar um modelo de *Machine Learning* baseado em árvores, criando assim uma competição entre os grupos para tentar gerar o melhor modelo.

2. O 2º: Um dataset escolhido por cada grupo, onde o principal objetivo é a exploração aprofundada dos dados, de modo a retirar informações e conclusões interessantes acerca dos dados.

2 Metodologias

A metodologia utilizada neste projecto para ambos os *datasets* é a ***CRISP-DM***. Em termos de algoritmos, utilizamos algoritmos baseados em árvores, especificamente *Random Forest*.

2.1 *CRISP-DM*

A metodologia ***CRISP-DM*** (*Cross Industry Standard Process for Data Mining*) descreve uma referência de fases bem estruturadas que se devem seguir e tarefas a executar para obtenção de resultados, tanto em projectos de *Data Mining* como em projetos de *Machine Learning*.

As fases principais em projetos de *Data Mining* são as seguintes:

1. **Compreensão do Negócio** - Compreende-se os objectivos do negócio ou do estudo e analisa-se os requisitos do mesmo;

2. **Compreensão dos Dados** - Os dados, nesta fase, são explorados de maneira a serem compreendidos e entendidos. São identificados os problemas dos dados e a qualidade dos mesmos. É retirada informação útil e conhecimento que está implícito nos dados;

3. **Preparação dos Dados** - Nesta fase os dados são analisados na utilização de vários algoritmos de *Data Mining*. É uma das fases mais importantes do projecto pois é onde são estudadas as correlações entre as variáveis ou colunas e então decididas as *features* a utilizar para o modelo. São utilizados também processos de transformação e limpeza dos dados;

4. **Modelação** - São usadas técnicas de modelação, ajustadas, para obter os melhores resultados possíveis. O retrocesso a fases anteriores é bastante normal para se conseguir melhorar o modelo;

5. **Avaliação** - No fim de se construir o modelo é decidido se vale a pena ou não avançar com a fase de implementação consoante os resultados obtidos. Se os resultados satisfazem as necessidades então o modelo é implementado;

6. **Desenvolvimento** - No caso deste projecto, por ser em âmbito curricular não será executada esta fase. Normalmente esta fase é quando se faz chegar a informação do modelo de forma visual, em relatório ou mesmo sendo uma parte integrante de um software, de forma a ser útil para o cliente ou a organização;

2.2 Modelo Baseados em Árvores

O algoritmo utilizado no projecto, nos dois *datasets*, foi o *Random Forest*, um algoritmo de aprendizagem-máquina que é bastante flexível e na maioria das vezes atinge bons resultados: o algoritmo de florestas aleatórias combina árvores de decisão para obter maior *accuracy* e estabilidade.

Uma das vantagens é que este algoritmo pode ser usado para modelos de regressão ou de classificação e este algoritmo adiciona aleatoriedade ao modelo na criação de árvores.

2.3 Ferramentas Utilizadas

As ferramentas utilizadas para a fase de análise de dados foram tanto ferramentas como *LibreOffice* e *Excel* como o *Knime*. Em relação à preparação de dados e ao desenvolvimento do modelo a única ferramenta utilizada foi o *Knime*.

3 *Dataset* da Competição

3.1 Contexto do Dataset

O principal objectivo é aprimorar um modelo de *Machine Learning* baseado em *decision trees*. Para isso é necessária uma análise exploratória do *dataset* para preparar o *dataset*, compreender o problema e obter o melhor modelo possível.

O *dataset* utilizado é referente a uma competição da plataforma *Kaggle*, onde o modelo vai competir com outros modelos no âmbito da disciplina de *SBS*.

O *dataset* utilizado neste projecto é referente ao nível de incidentes rodoviários na zona de Braga. Contém, por ocorrência, várias *features* que a caracterizam desde as condições atmosféricas (precipitação, velocidade do vento, temperatura) até às estradas afectadas. Enumerando-as:

1. ***city_name*** - nome da cidade: neste caso *Braga*;
2. ***record_date*** - o *timestamp* associado ao registo
3. ***magnitude_of_delay*** - magnitude do atraso provocado pelas ocorrências que se verificam no *record_date* correspondente;
4. ***delay_in_seconds*** - atraso, em segundos, provocado pelas ocorrências;
5. ***affected_roads*** - lista de todas as estradas afectadas pelas ocorrências;
6. ***luminosity*** - o nível de luminosidade que se verificava;
7. ***avg_temperature*** - valor médio da temperatura à data e hora da ocorrência;
8. ***avg_atm_pressure*** - valor médio da pressão atmosférica pà data e hora da ocorrência;
9. ***avg_humidity*** - valor médio da humidade à data e hora da ocorrência;
10. ***avg_wind_speed*** - valor médio da velocidade do vento à data e hora da ocorrência;
11. ***avg_precipitation*** - valor médio de precipitação à data e hora da ocorrência;
12. ***avg_rain*** - avaliação qualitativa do nível de precipitação à data e hora da ocorrência;
13. ***accidents*** - nível de gravidade das ocorrências registadas;

3.2 Análise e Transformação dos Dados

Previamente à modelação, foi transformado e analisado o *dataset* de forma a obter o melhor modelo possível.

As transformações ao *dataset* foram baseadas, tanto em lógica - por exemplo, a divisão da data em várias colunas - como em análises ao *dataset*.

3.2.1 Limpeza do *Dataset*

Inicialmente, de forma a facilitar e potenciar melhores análises, antes de fazer qualquer tipo de reconhecimento, procedeu-se a uma limpeza do *dataset*:

- Como a análise engloba apenas os dados da cidade de Braga, **dividimos e eliminamos a coluna *city_name***: não faz sentido incluir uma coluna com dados recorrentes para todas as observações;
- Para esclarecimentos com base na sazonalidade e análise temporal, **dividimos a coluna *record_date*** para caracterizar a hora, o mês, o dia e o trimestre;
- Também **eliminamos** a coluna *affected_roads* criando novas colunas: uma divisão por **tipo de estrada**, uma **contagem** do número de estradas envolvidas e até uma coluna para sabermos se a ocorrência **envolveu uma estrada ou não**.

3.2.2 Análise ao *Dataset*

O objectivo deste modelo é, tendo em conta vários factores, prever a magnitude de risco de uma dada ocorrência. As ocorrências podem tomar **cinco** valores: *Very High*, *High*, *Medium*, *Low* ou *None*.

Numa primeira análise ao *dataset* podemos ver que as ocorrências estão divididas da seguinte maneira:

Nível da Ocorrência	None	Low	Medium	High	Very High
% Total Obs.	28.0%	27.4%	21.1%	14.0%	9.6%

Tabela 1: Tabela a mostrar a distribuição geral das ocorrências no *dataset*.

As ocorrências estão distribuídas quase por igual: O maior grupo são as ocorrências *None*; juntando as ocorrências de *High* e *Very High*, quase teríamos uma divisão homogénea entre cada categoria.

Neste capítulo do relatório vamos analisar cada coluna/dimensão do *dataset* e perceber a sua distribuição, as interacções que existem entre elas e de que maneira cada dimensão influencia o nível da ocorrência.

3.2.2.1 Affected Roads

Como foi dito antes, foi uma coluna alvo de um tratamento pré-análise. Da coluna **surgiram** 8 **dimensões novas**: 6 **dimensões binárias** que indicam se a ocorrência **envolveu um tipo específico de estrada** (*road A*, *road CM*, *road M*, *road EM*, *road IP* ou *road N*) e 1 **dimensão numérica** que indica o número total de estradas afectadas (*N roads*) e, finalmente, 1 *dimensão binária* para perceber se, em cada ocorrência houve estradas repetidas.

As ocorrências, maioritariamente, acontecem em estradas to tipo **N** e **CM**:

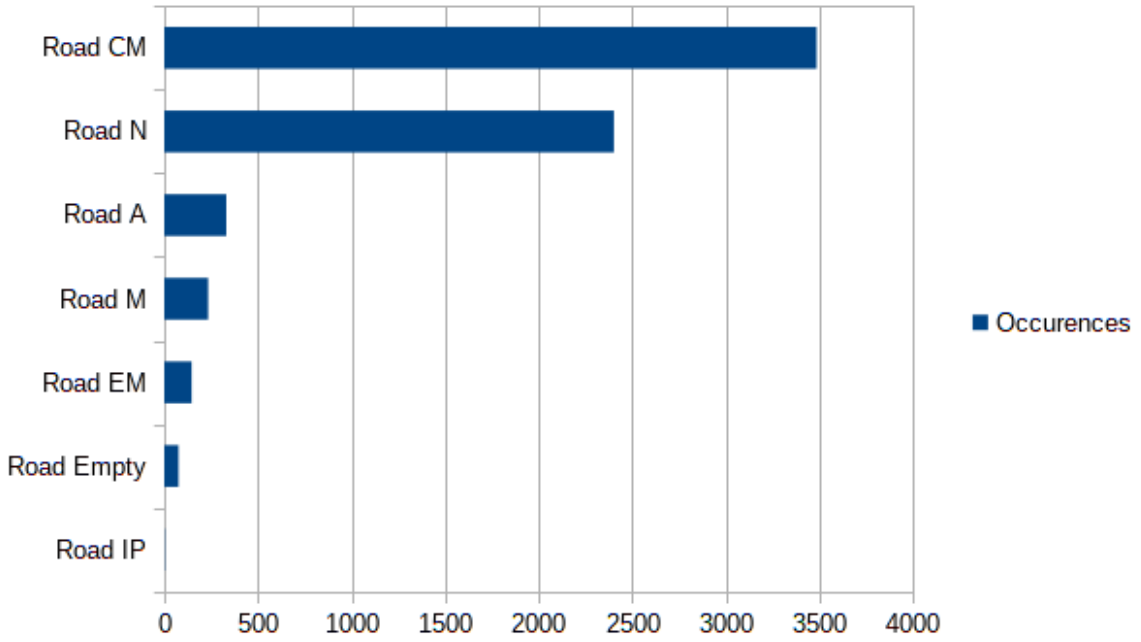

Figura 1: Número de ocorrências *flagged* por cada tipo de estrada

Como podemos ver, as ocorrências em estradas **N** e **CM**, tendo em conta que as ocorrências podem estar duplamente representadas (pode haver mais que uma estrada *flagged* por ocorrência), compõem 88.2% do total de ocorrências por estrada distinta. Visto de outra maneira, das **5000** ocorrências registadas, 48% tiveram um incidente numa estrada **N** e 71% tiveram numa estrada **CM**.

O passo seguinte foi perceber a distribuição de gravidade da ocorrência por tipo de estrada; analisamos apenas as estradas do tipo *N, M* e *A* - Analisamos as estradas do tipo *A* mesmo que, comparando com as estradas *N* e *CM*, tenha poucas ocorrências por duas causas: primeiro porque, embora tenha poucas ocorrências, tem ocorrências suficientes para ser útil e também pela sua distribuição peculiar, como vamos observar a seguir.

Visualmente, analisando assim a gravidade das ocorrências por tipo de estrada, observamos que:

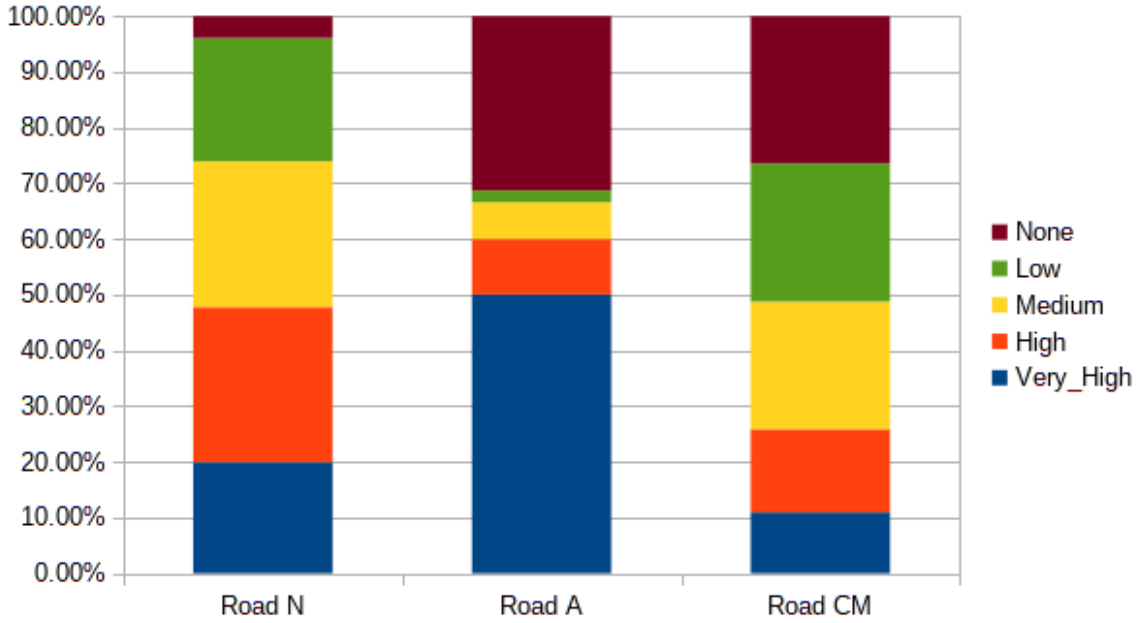

Figura 2: % de cada categoria de gravidade da ocorrência, por tipo de estrada.

- 50% das ocorrências nas estradas **A** têm gravidade *Very High*; algo completamente diferente da distribuição das outras estradas;
- A % de ocorrências **de gravidade *Low* e *Medium* são muito parecidas entre as estradas do tipo *N* e *CM***, mesmo assim, vemos que as ocorrências sem classificação (*None* é uma **percentagem ínfima** nas estradas do tipo **N**, contrastando com as estradas do tipo **CM**, onde é uma percentagem já mais semelhante tanto à categoria *Medium* como a categoria *Low*.

São variáveis interessantes para o modelo.
Outro ponto a analisar, como também foi referido anteriormente, foi de que maneira o **número de estradas afetada influenciava o *outcome*** da ocorrência.

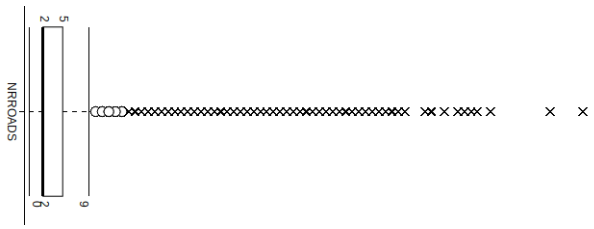

Figura 3: *Boxplot* a representar a distribuição do número de estradas afectadas no *dataset*

Para isso analisamos primeiro a distribuição do número de estradas afectadas: podemos ver que 75% das ocorrências tiveram menos que 5 estradas afectadas. Existe uma **média** é 5.6 estradas por ocorrência e uma *mediana* de 2.

Além disso, se olharmos para a distribuição de estradas afectadas por gravidade de acidente, percebemos claramente uma diferença no número de estradas afectadas:

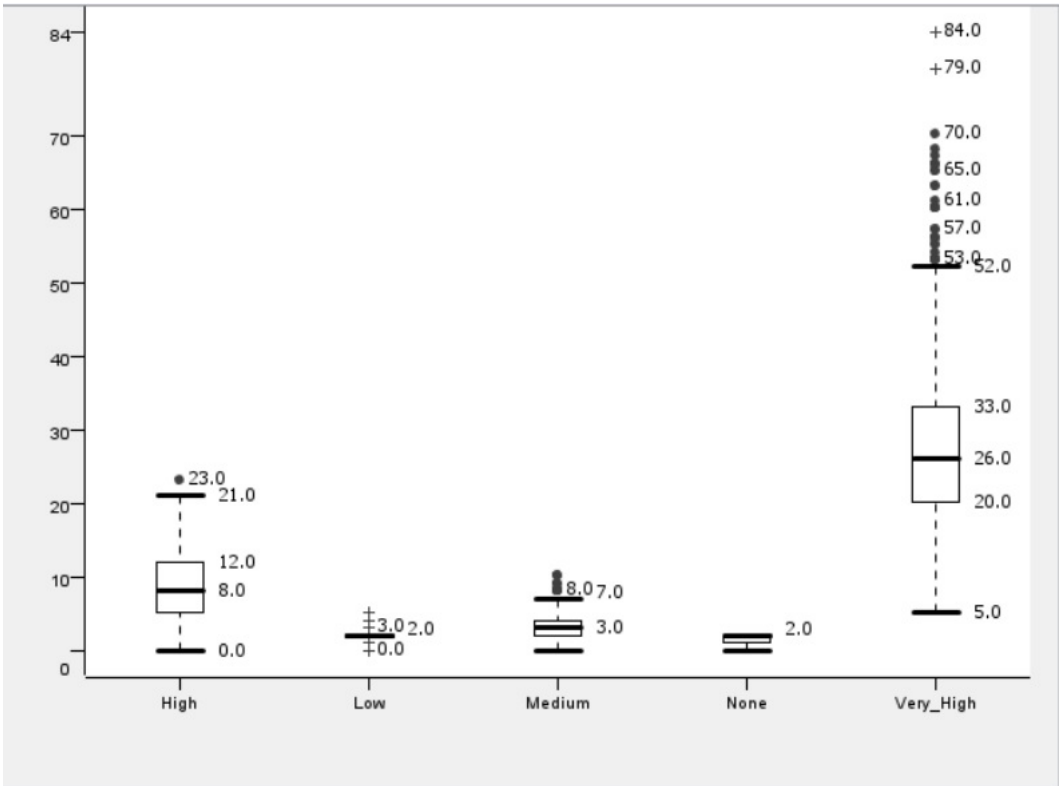

Figura 4: *boxplots* com a divisão do número de estradas afectadas por tipo de ocorrência

Como podemos ver:

- quando são ocorrências com gravidade *Very High* o número de estradas afectadas é consistentemente maior que nos outros níveis de gravidade;
- O número de estradas afetadas dos níveis *Low*,*Medium* e *None* são muito próximas, mas existe um ligeiro aumento nas de gravidade *Medium*;
- As ocorrências de gravidade *High* destacam-se das de *None*, *Low* e *Medium*; mas não tanto como as *Very High*.

Tendo estes dois pontos em conta, para facilitar a análise e para mais tarde adicionar ao modelo, criamos uma install extensions from surprise import Reader from surprise import Dataset from surprise import SVD, KNNBasic import pandas as pd import $mong_acessasmaimportsurp_optparassopimportrec_genrasrgfromsurpriseimportPredictionImpossib$
from sklearn.metrics.pairwise import $cosine_similarityfromsklearn.feature_extraction.textimpor$
import mysql.connector as mysql from sqlalchemy import $create_engine$
def $import_mtsql(table) : db_connection_str =' mysql+pymysql : //root : 1234567Ll.@localhost/rec$
$create_engine(db_connection_str)df = pandas.read_sql("SELECTISBN, lower(concat(Book-$

$Title,'',replace(Book-Author,'',''),'',Year-Of-Publication,'',replace(Publisher,'',''))aswords$
$str(table) + "limit3000;", con = db_connection)$
return df
book$_inf = ma.import_msql('book')print(book_inf)$
count=CountVectorizer() count$_matrix = count.fit_transform(book_inf['words'])$
gerando a matriz de similaridade de cosseno cosine$_sim = cosine_similarity(count_matrix, count_mat$
print(cosine$_sim)$
creating a Series for the movie titles so they are associated to an ordered numerical list I will use in the function to match the indexes
creating a Series for the movie titles so they are associated to an ordered numerical list I will use in the function to match the indexes indices = pd.Series(book$_inf.index)$
defining the function that takes in movie title as input and returns the t install extensions from surprise import Reader from surprise import Dataset from surprise import SVD, KNNBasic import pandas as pd import mong$_acessasmaimportsurp_optparassopimportrec_$
from sklearn.metrics.pairwise import cosine$_similarityfromsklearn.feature_extraction.textimport$
import mysql.connector as mysql from sqlalchemy import create$_engine$
def import$_msql(table) : db_connection_str =' mysql+pymysql : //root : 1234567Ll.@localhost/rec$
$create_engine(db_connection_str)df = pandas.read_sql("SELECTISBN, lower(concat(Book-Title,'',replace(Book-Author,'',''),'',Year-Of-Publication,'',replace(Publisher,'','')))aswords$
$str(table) + "limit3000;", con = db_connection)$
return df
book$_inf = ma.import_msql('book')print(book_inf)$
count=CountVectorizer() count$_matrix = count.fit_transform(book_inf['words'])$
gerando a matriz de similaridade de cosseno cosine$_sim = cosine_similarity(count_matrix, count_mat$
print(cosine$_sim)$
creating a Series for the movie titles so they are associated to an ordered numerical list I will use in the function to match the indexes
creating a Series for the movie titles so they are associated to an ordered numerical list I will use in the function to match the indexes indices = pd.Series(book$_inf.index)$
defining the function that takes in movie title as input and returns the top 10 recommended movies def recommendations(title, install extensions from surprise import Reader from surprise import Dataset from surprise import SVD, KNNBasic import pandas as pd import mong$_acessasmaimportsurp_optparassopimportrec_genrasrgfromsurpriseimportPredic$
from sklearn.metrics.pairwise import cosine$_similarityfromsklearn.feature_extraction.textimport$
import mysql.connector as mysql from sqlalchemy import create$_engine$
def import$_msql(table) : db_connection_str =' mysql+pymysql : //root : 1234567Ll.@localhost/rec$
$create_engine(db_connection_str)df = pandas.read_sql("SELECTISBN, lower(concat(Book-Title,'',replace(Book-Author,'',''),'',Year-Of-Publication,'',replace(Publisher,'','')))aswords$
$str(table) + "limit3000;", con = db_connection)$
return df
book$_inf = ma.import_msql('book')print(book_inf)$
count=CountVectorizer() count$_matrix = count.fit_transform(book_inf['words'])$
gerando a matriz de similaridade de cosseno cosine$_sim = cosine_similarity(count_matrix, count_mat$

print(cosine$_sim$)

creating a Series for the movie titles so they are associated to an ordered numerical list I will use in the function to match the indexes

creating a Series for the movie titles so they are associated to an ordered numerical list I will use in the function to match the indexes indices = pd.Series(book$_inf.index$)

defining the function that takes in movie title as input and returns the top 10 recommended movies def recommendations(title, cosine$_sim = cosine_sim$) : $initializingtheemptylistofreco$ []

gettin the index of the movie that matches the title idx = indices[indices==title].index[0]

creating a Series with the similarity scores in descending order score$_series = pd.Series(cosine_sim[$ $False)$

getting the indexes of the 10 most similar movies top$_10_indexes = list(score_series.iloc[1:11].index)$

populating the list with the titles of the best 10 matching movies for i in top$_10_indexes$: $recommended_movies.append(list(book_inf.index)[i])$

return recommended$_movies cosine_sim = cosine_sim)$: $initializingtheemptylistofrecommendedm$ []

gettin the index of the movie that matches the title idx = indices[indices==title].index[0]

creating a Series with the similarity scores in descending order score$_series = pd.Series(cosine_sim[$ $False)$

getting the indexes of the 10 most similar movies top$_10_indexes = list(score_series.iloc[1:11].index)$

populating the list with the titles of the best 10 matching movies for i in top$_10_indexes$: $recommended_movies.append(list(book_inf.index)[i])$

return recommended$_movies op10recommendedmoviesdefrecommendations(title, cosine_sim = cosine_sim)$: $initializingtheemptylistofrecommendedmoviesrecommended_movies =$ []

gettin the index of the movie that matches the title idx = indices[indices==title].index[0]

creating a Series with the similarity scores in descending order score$_series = pd.Series(cosine_sim[$ $False)$

getting the indexes of the 10 most similar movies top$_10_indexes = list(score_series.iloc[1:11].index)$

populating the list with the titles of the best 10 matching movies for i in top$_10_indexes$: $recommended_movies.append(list(book_inf.index)[i])$

return recommended$_movies$**variável categórica***chamada**roads_affect**quevisadividiravarivela* $tomaovalordeAseonmerodeestradasformenor\ que\ 3, Bseestiverentre\ 3,\ inclusive,\ e\ 6eCseformaio$

Para a criar, no *Knime*, foi utilizado um *node* de *javascript*.

Finalmente, derivamos uma variável para **perceber se, por ocorrência, havia estradas repetidas ou ocorrências sem estradas afectadas**. Visamos acima de tudo perceber se, por, à partida, acontecer várias ocorrências **no mesmo período de tempo e no mesmo local** a **gravidade** da ocorrência **aumentava**.

As ocorrências com a estrada repetida na coluna *affected_roads* são *muitas*, cerca de 58% das ocorrências totais.

```
String[] parts = c_affected_roads.split(",");
out_NRROADS = parts.length;

if (out_NRROADS >= 3 && out_NRROADS < 6 ){
        out_roads_affect = "B";
}else if (out_NRROADS >= 6){
        out_roads_affect = "C";
}else {
        out_roads_affect = "A";
}
```

Figura 5: Transformação efectuada no *Knime*

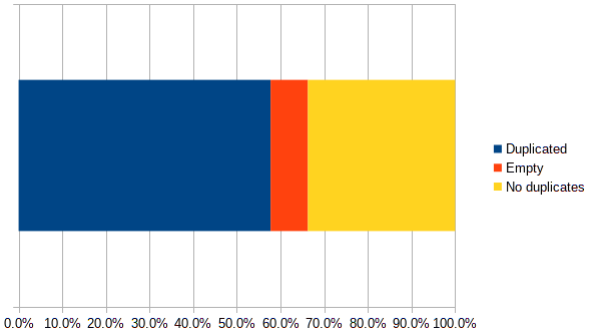

Figura 6: *Bar plot* a mostrar a distribuição das ocorrências

Olhando também à distribuição da gravidade dos acidentes por tipo de *repeated_road*, vemos que, quando são ocorrências com *empty*, são maioritariamente ocorrências sem gravidade atribuída (*None*). Já a hipótese inicial desmente-se: 50% das ocorrências quando não existem estradas duplicadas são de gravidade *High* ou *Very High*, comparando com 10%, quando existe duplicação.

3.2.2.2 *Record Date*

Outra coluna que tratamos foi a *record_date*. Dividimo-la em 6 *features*: *hour*, *month*, *quarter*, *day of week*, *week* e *parte do dia*.A divisão clara e a divisão e muito *self-explanatory*, mas implica comportamentos diferentes em termos da gravidade das ocorrências.

Por exemplo, olhando para a *week*, tem uma divisão equilibrada, isto é não há diferenças notórias no número de ocorrências por semana, mas, olhando para a distribuição da gravidade das ocorrências conseguimos perceber que existe um período em que as ocorrências de gravidade *None* deixam de existir.

Olhando em termos de *parte do dia*, em termos de volume de acidentes, não há uma

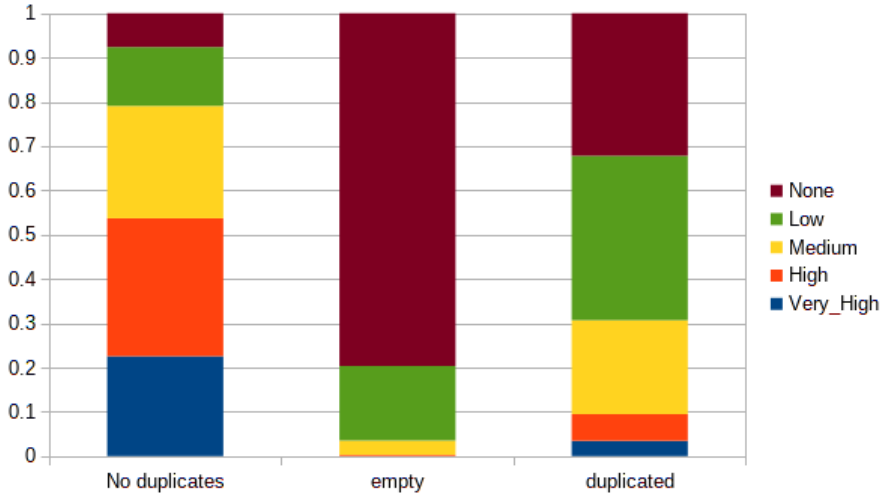

Figura 7: *Bar plot* a mostrar a distribuição das ocorrências por *repeated_road*

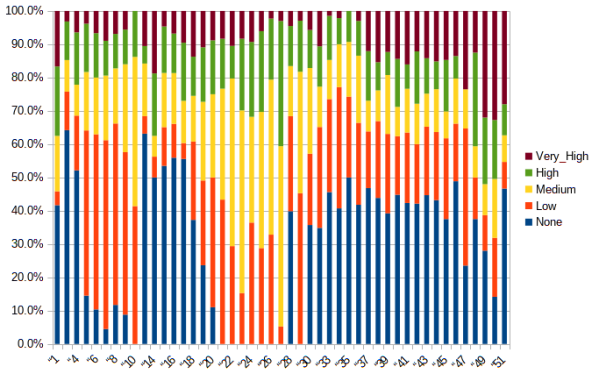

Figura 8: Distribuição da gravidade das ocorrências por *week*

diferença significativa. Quanto à distribuição da gravidade de ocorrências, embora não haja um padrão vísivel, percebemos que de *madrugada* não há ocorrências de gravidade *Very High*.

Quanto ao *quarter*, percebemos que os acidentes não estão homogeneamente distribuídos. O *quarter* com menos acidentes é o ***1*** (19%) e o com mais ocorrências é o ***3*** (30%). A distribuição da gravidade de ocorrências não é esclarecedora.

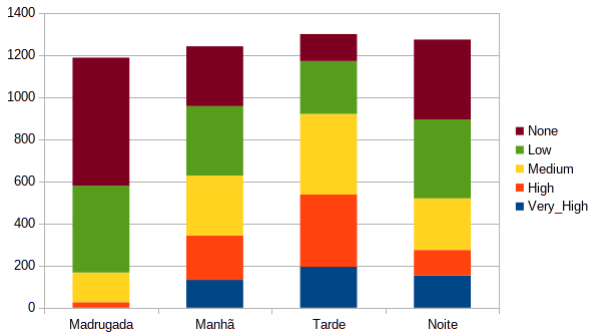

Figura 9: Distribuição da gravidade das ocorrências por *parte do dia*

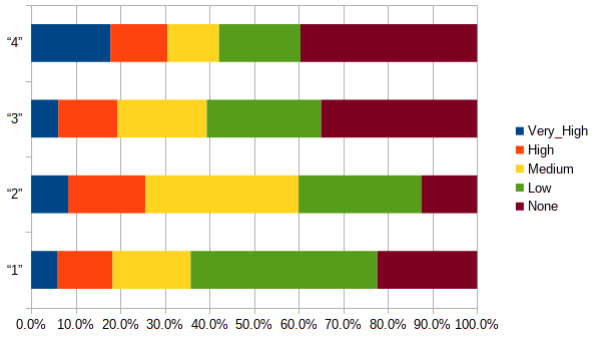

Figura 10: Distribuição da gravidade das ocorrências por *quarter*

3.2.2.3 Magnitude

Foi analisado também a magnitude do *delay* do acidente, mostrado pela coluna *magnitude_ of_ delay* e *delay_ in_ seconds*.

Figura 11: *Boxplot* a distribuição do *delay*

Primeiramente, olhamos à distribuição do *delay* da ocorrência. Observamos que o último *quarter* tem o intervalo maior entre os dados (de 3000 a 68000).

Dado esta distribuição, decidimos criar uma variável chamada *delay_ class*, que, ao contrário da *magnitude of delay*, está dividida em quatro categorias.

É claro que há uma distribuição completamente diferente entre estas categorias: primeiramente, quando o *delay_ class* é *zero*, não existem ocorrências de gravidade *Very*

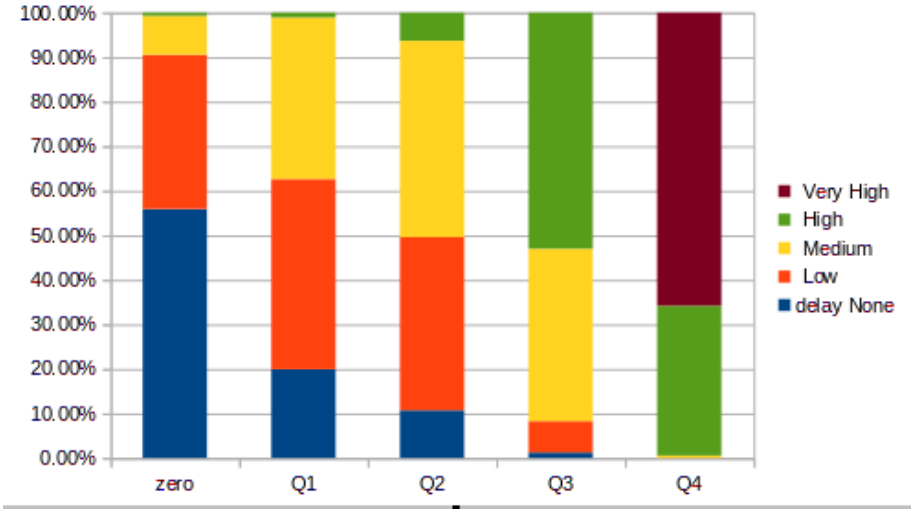

Figura 12: *Bar plot* com a distribuição da gravidade das ocorrências consoante o *quarter* do *delay*

High e quase nenhuma *High*, contrastando com o valor *Q4*, onde apenas há ocorrências de gravidade *High* e *Very High*. Entre elas, nota-se que a distribuição está dividida de forma gradual, com, no sentido *Q1* a *Q3*, as ocorrências de gravidade *Low* e *None* gradualmente a diminuírem.
A distribuição da *feature magnitude_ of_ delay* também foi estudada, onde observamos que 46% das ocorrências não tinham uma magnitude definida(*Undefined*).

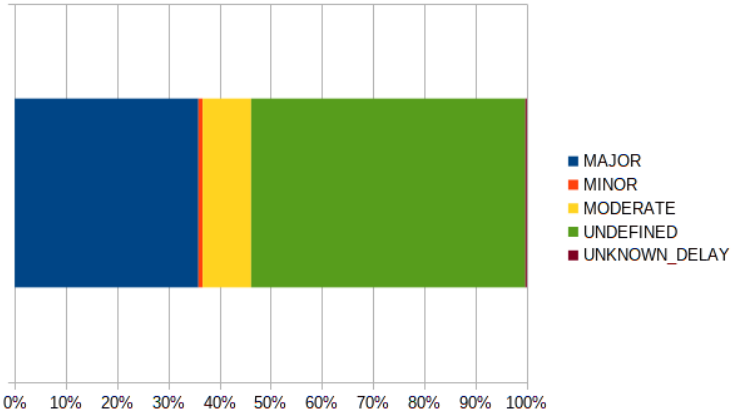

Figura 13: Distribuição da variável *magnitude_ of_ delay*

Devido ao facto de já utilizarmos outra variável para dividir e categorizar o tempo de espera, decidimos, criar uma variável chamada *magnitude_ defined* onde apenas indicaria se a magnitude está definida ou não.
Consegue-se perceber que a distribuição é muito parecida com a previamente mostrada.

Nível da Ocorrência	None	Low	Medium	High	Very High
Magnitude Undefined	82.92%	76.54%	42.41%	3.43%	0%
Magnitude Defined	17.08%	23.46%	57.59%	96.57%	100%

Tabela 2: Tabela a mostrar a distribuição da variável *magnitude_ of_ delay*, tendo em conta a gravidade da ocorrência

3.2.2.4 Análise das variáveis climatéricas

Após analisar as variáveis temporais e das condições físicas da ocorrência, analisamos as variáveis climatéricas.

Nesta categoria inserem-se as *features avg_ temperature*, *avg_ atm_ pressure*, *avg_ humidity*, *avg_ wind_ speed*, *avg_ rain* e *avg_ precipitation*. Primeiro tratamos de perceber as variáveis; ao olhar para a *avg_ precipitation* percebemos que havia valores muito disformes: 4997 das 5000 observações toma o valor de 0: ainda por cima, olhando a *avg_ rain*, que pode tomar valores inteiros, 4645 desses valores correspondem a *Sem Chuva*, mesmo que todos esses estejam classificados com 0 na *avg_ precipitation*.

avg_rain / avg_precipitation	0	1	5	Total
aguaceiros	27			27
aguaceiros fracos	61			61
chuva	4			4
chuva forte	8			8
chuva fraca	171	1		172
chuva leve	16			16
chuva moderada	58		2	60
chuvisco e chuva fraca	3			3
chuvisco fraco	4			4
Sem Chuva	4645			4645
Total	4997	1	2	5000

Tabela 3: Tabela a especificar o *deepdive* feito às variáveis *avg_ precipitation* e *avg_ rain*

Após a análise ao *avg_ rain* e *avg_ precipitation* avaliamos as outras *features*.

Após perceber a distribuição geral de cada *feature*, precisávamos de entender de que maneira estão relacionadas a a gravidade de cada ocorrência. Para isso comparamos a média das *features*, para cada nível de gravidade, com a média geral dessa *feature*.

Percebemos que ao nível da pressão atmosférica, não houve uma alteração significante e na velocidade do vento, quando havia ocorrências de nível *Very High*, *High* e *Medium* acima da média, aumentando a diferença à medida que a gravidade aumenta.

Na temperatura também se pode observa quase o mesmo comportamento, mas, provavelmente, a diferença na gravidade *Very High* toma aquele valor devido à falta de observações.

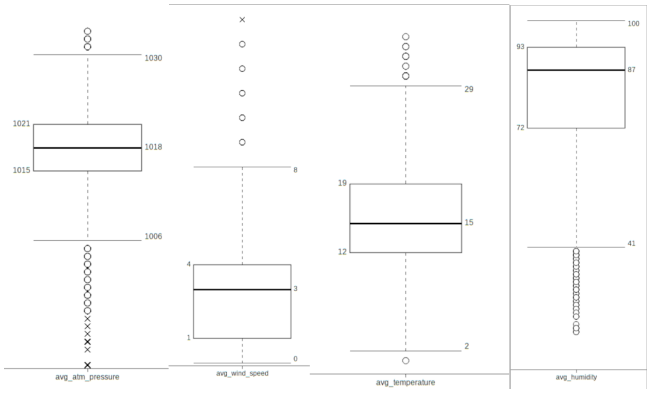

Figura 14: *Boxplots* das *features* climatéricas (excepto as relacionadas com chuva).

Accident Level	Temperature	ATM Pressure	Humidity	Wind Speed
Very_High	-0.79%	-0.04%	-2.06%	16.65%
High	10.28%	-0.08%	-4.78%	19.67%
Medium	10.91%	-0.02%	-4.13%	9.38%
Low	-3.50%	0.06%	0.60%	-5.77%
None	-9.68%	0.00%	5.62%	-16.97%
Geral	**15.5**	**1018.0**	**81.8**	**3.0**

Tabela 4: Comparação da média das *features* incluídas em cada nível de gravidade com a média geral

3.2.2.5 Correlações

Analisando as correlações entre features retiramos alguns pontos:

- Existe uma correlação enorme entre os vários tipos de *roads*;
- O *delay_in_seconds* tem uma correlação grande com os variós tipos de roads;
- A correlação mais baixa com *accidents* é a *road_N*.

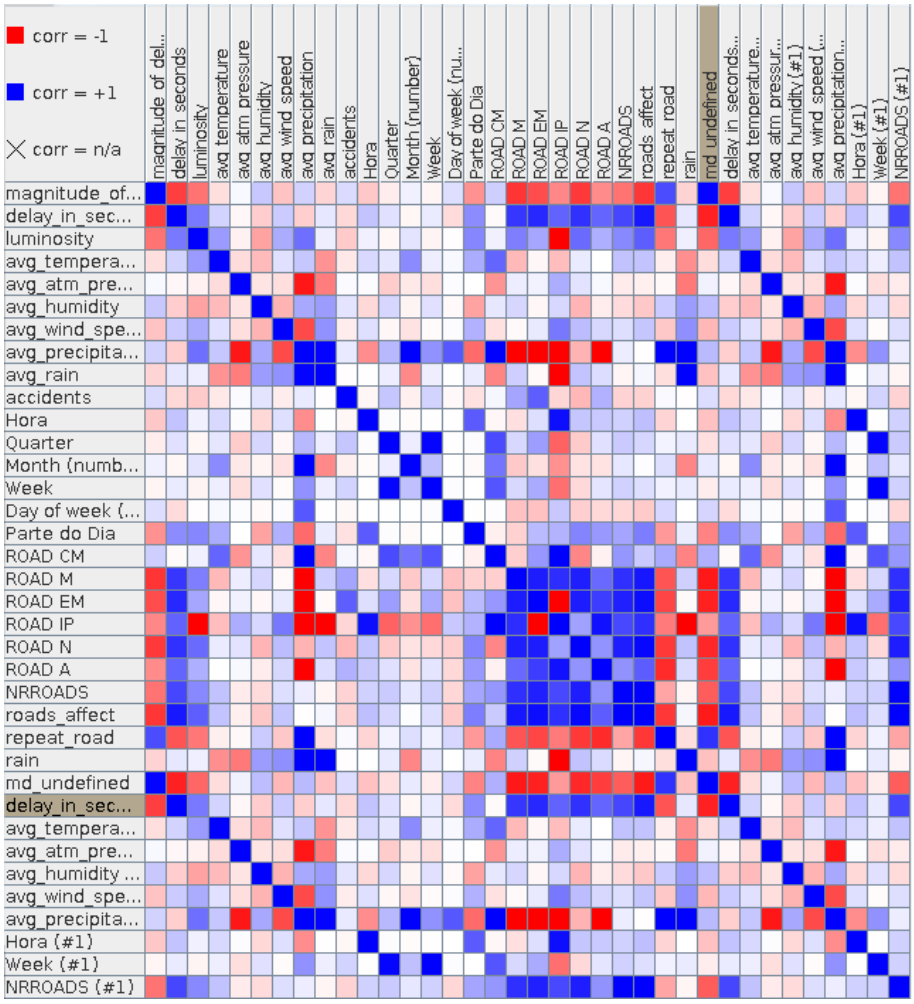

Figura 15: Correlação entre todas as *features* calculadas

3.2.3 Transformação de *Features*

No *Knime*, aplicamos as transformações que foram mencionadas anteriormente. Organizamos o *workflow* em vários conjuntos de *nodes*:

- ***Feature Engineer -Data***

  Neste *node* transformamos a *record_ date*, criando as features descritas na análise.

- ***Feature Engineer -Roads***

  Neste *node* aplicamos todas as transformações necessárias à coluna *affected_ roads*.

- ***Feature Engineer -Rain***

  Aqui criamos uma variável binária, embora, como explicado na análise não ser usada no modelo, com o valor de 1 quando chove e 0 quando não chove, baseada na *avg_ rain*.

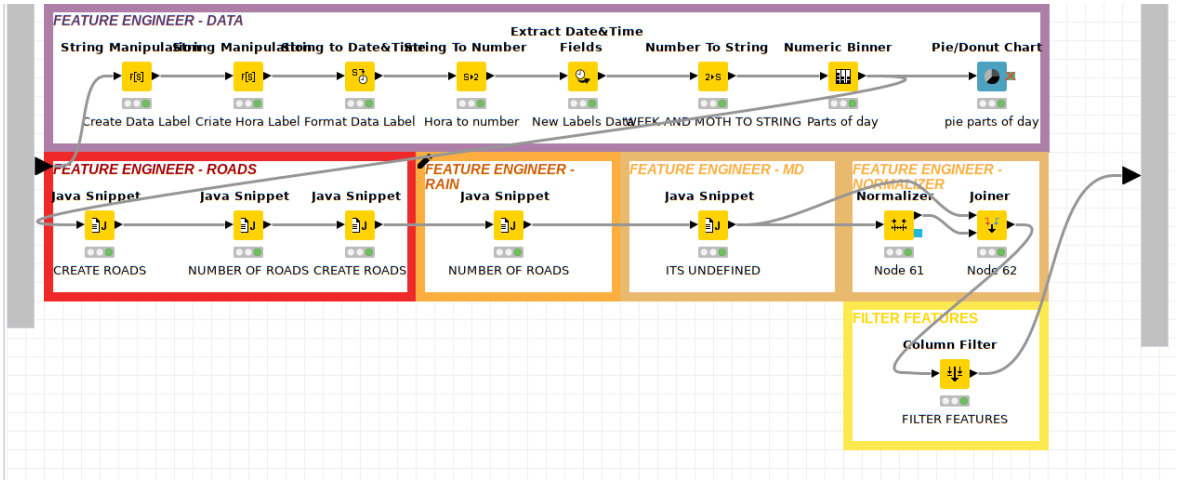

Figura 16: *Workflow Data Preparation*: Organização dos *nodes* para aplicar as transformações necessárias

- ***Feature Engineer -MD***

  Neste grupo tratamos a magnitude, criando as variáveis anteriormente mencionadas.

- ***Feature Engineer -Normalizer***

  Aqui, para ver se trazia alguma melhoria ao modelo, **normalizamos as variáveis climáticas**.

3.2.4 *Features* seleccionadas

Após a análise e a algumas iterações na selecção do modelo, seleccionaram-se as seguintes *features*:

- Variável binária que implica que a ***Road CM*** tenha sido afectada;
- Variável binária que implica que a ***Road A*** tenha sido afectada;
- Variável que nos diz o ***número de roads***;
- Variável que nos indica se choveu ou não: ***Has_rain***;
- Temperatura Normalizada ***avg_temperature_normalized***;
- Humidade Normalizada ***avg_humidade_normalized***;
- Velocidade do Vento Normalizada ***avg_windspeed_normalized***;
- Semana ***Week***;

- Variável que indica a Magnitude do delay ***magnitude_ of_ delay***;
- Variável com o delay em segundos ***seconds_ of_ delay***;
- Variável que nos indica a pressão média ***avg_ pressure***;
- Variável que nos indica a velocidade do vento média ***avg_ wind_ speed***;
- Hora da ocorrência ***hour***;
- Month da ocorrência ***month***;
- Dia da Semana da ocorrência ***day_ of_ week***;

3.3 Modelação

3.3.1 Arquitetura Principal

Em baixo podemos ver a arquitetura principal do workflow que conte com uma componente analítica para analise de dados(Analitycs Enviroment), uma componente para preparação de dados (Data Preparation Enviroment), um ambiente de optimização de parâmetros (Model Optimization Enviroment) e finalmente um ambiente experimental (Data Preparation Experiments Enviroment) para se poder fazer qualquer tipo de experiência sem comprometer o desenvolvido nos outros ambientes.

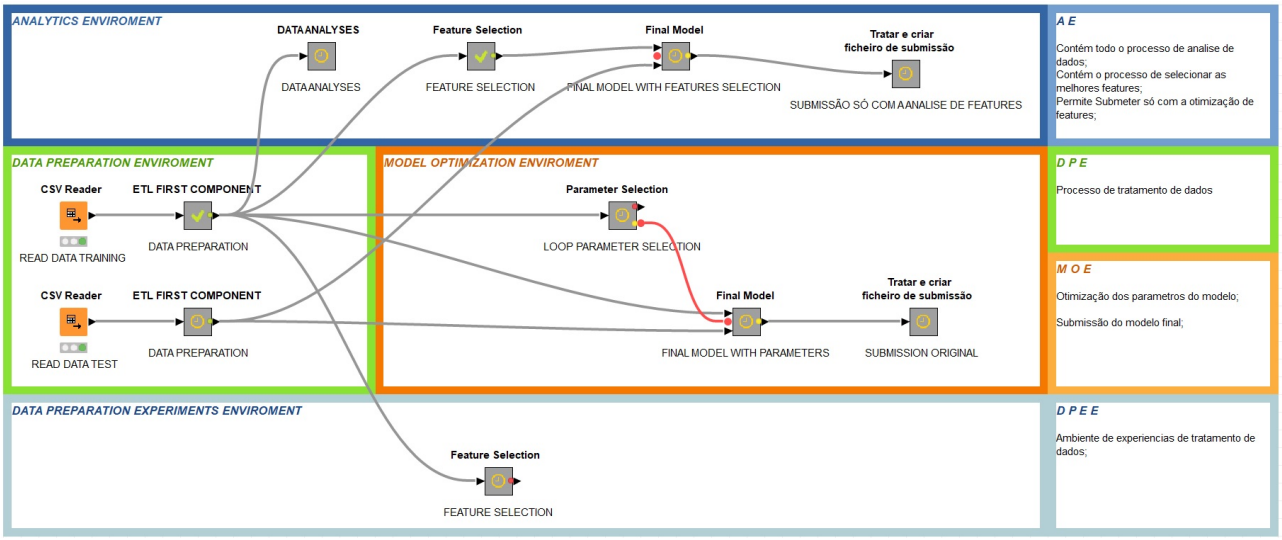

Figura 17: Arquitetura principal do Workflow

3.3.2 Loop Feature Selection

A próxima figura mostra o loop de seleção de features, esta componente pertence ao ambiente de analise de dados e foi indispensável para os bons resultados do modelo. Foi utilizada uma abordagem hibrida na forma em que os developers seleccionavam as features com base na analise de dados (capitulo 3.2) e também tinham em atenção ao resultado do loop das features, em que o tipo de loop mais utilizado foi o forward e o backward.

Com este loop foi possivel ter em atenção features que não pareciam tão importantes mas que se destacaram no modelo, como é o caso da feature week.

Se repararmos na figura em cima foram realizadas submissões só com a escolha das features porque nesta fase o modelo atingiu resultados ótimos que seram abordados no cápitulo 3.

Numa breve explicação do loop podemos ver que está dividido em 4 fases, a primeira (Decide All Parameters), é onde são escolhidos os parametros de execução do loop,

no caso deste projeto optou-se por o método forward e backward, e por defeito os parâmetros do random forest. Na segunda fase (Cross Validation), apesar de ter sido recomendado a não utilização da validação cruzada na loop da escolha das features, o grupo de trabalho decidiu usar mesmo assim porque foram realizadas submissões só com a selecção de features, sendo que era importante trazer confiança nos resultados ter confiança nos resultados. A terceira componente deste loop é onde é decidido o objectivo, neste caso, melhorar a accuracy e podemos ver também que é a fase do processo onde todos os resultados do loop vêm de um agrupamento para serem transportados para a fase 4, que simplesmente se encarrega de selecionar as features que trazem melhor resultado para o modelo, isto é, melhor accuracy.

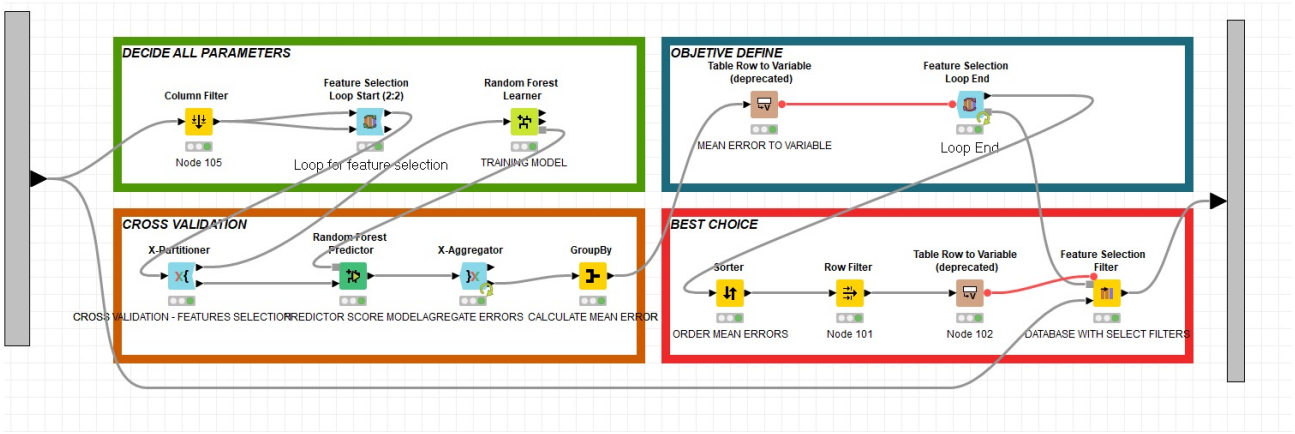

Figura 18: Loop de selecção de Features

3.3.3 Loop de Otimização de Parâmetros

Na parte de Otimização dos Parametros, foi desenvolvido um *loop* de igual forma distribuído pelas componentes do *loop* das *features* mas com os nodos correspondentes a escolha de parâmetros. A primeira parte do *loop* (*DECIDE ALL PARAMETERS*) é onde se selecionam os parâmetros e o intervalo deste que será testado. A segunda componente deste *loop* é a *cross validation* para garantir a confiança dos parâmetros. Em relação à terceira e quarta componente, representam o agrupamento dos resultados e a escolha dos melhores parâmetros para o modelo consoante as *features* escolhidas. O parâmetro mais trabalhado neste projeto foi o numero de modelos que a *random forest* criaria.

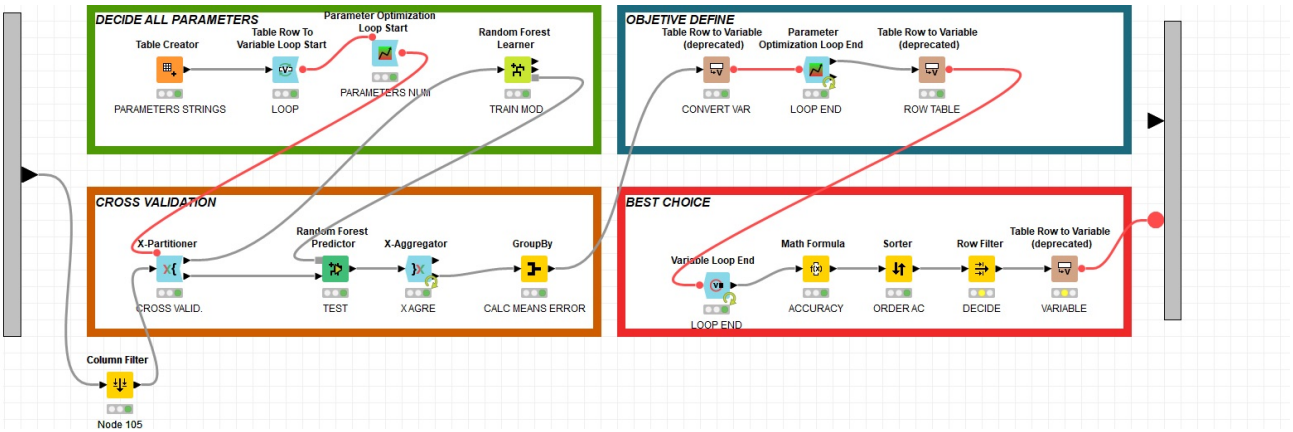

Figura 19: Loop de Optimização dos parâmetros

Depois de serem escolhidas as *features* e otimizados os parâmetros é desenvolvida a modelação final para prever os valores de teste e é criado o ficheiro de submissão.

3.4 Análise de Resultados

Em relação aos resultados, o modelo *Random Forest* desenvolvido teve óptimos valores, rondando, no final, os 92% de *accuracy*.

No *loop* dos parâmetros foram seleccionados os valores que trazem menos erro ao modelo, 282 Número de modelos, 23 no limite de níveis, o critério de divisão foi *Information Gain Ratio*, os outros parâmetros restantes ficaram como os *default* na ferramenta Knime.

Depois da análise da matriz de confusão e da precisão conseguimos perceber que as previsões com pior performance são para a categoria *Medium* e *High* que são as categorias intermédias do *dataset*. São estas Categorias que estão pior distribuídas no *dataset* e no futuro se se quiser obter melhores resultados terá de ser possível colmatar esta distribuição um pouco desbalanceada. De qualquer forma os resultados são considerados bons pois o modelo esta com cerca de 92% de *accuracy* e com uma precisão de

quase 84%. Se a precisão for melhorada com foco nas categorias *medium* e *high* muito provavelmente o modelo é melhorado.

Em relação a competição da plataforma *Kaggle*, foram feitas apenas 7 submissões, obtendo uma *accuracy* de cerca de 92% na parte pública. Foram feitas poucas submissões porque o grupo de trabalho usou o *cross validation* desde início para prever os dados e teve sempre confiança nos resultados obtidos sabendo, desta forma, quando a *accuracy* iria subir.

Row ID	Medium	Low	Very_High	None	High
Medium	923	58	0	1	96
Low	63	1275	0	19	0
Very_High	0	0	457	0	17
None	7	34	0	1379	0
High	61	1	22	0	587

Row ID	TruePo...	FalsePo...	TrueNe...	FalseN...	Recall	Precision	Sensitivity	Specifity	F-meas...	Accuracy	Cohen'...
Medium	923	131	3791	155	0.856	0.876	0.856	0.967	0.866	?	?
Low	1275	93	3550	82	0.94	0.932	0.94	0.974	0.936	?	?
Very_High	457	22	4504	17	0.964	0.954	0.964	0.995	0.959	?	?
None	1379	20	3560	41	0.971	0.986	0.971	0.994	0.978	?	?
High	587	113	4216	84	0.875	0.839	0.875	0.974	0.856	?	?
Overall	?	?	?	?	?	?	?	?	?	0.924	0.902

Figura 20: Matriz de Confusão, Accuracy e Precisão

4 *DataSet 1 - Students Performance*

4.1 Contexto do *Dataset*

O *dataset* que escolhemos aborda o sucesso académico de alunos do ensino secundário de duas escolas brasileiras, nomeadamente na disciplina de português. Os atributos que o *dataset* contem inclui as notas dos alunos, assim como atributos relacionados com dados sociais e demográficos.

O principal objectivo é explorar o *dataset* de modo a encontrar relações significativas entre os atributos dos alunos, em especial os que tenham impacto na nota final da disciplina de Português. Para tal construímos um modelo de *machine learning* baseado em árvores para realizar uma análise completa dos dados de modo a ajudar a retirar informações relevantes do *dataset*.

O que nos levou à escolha deste *dataset* foi principalmente a componente de exploração de dados do mesmo, visto que pode levar a descoberta de relações interessantes entre atributos que tenham impacto no sucesso ou insucesso escolar e deste modo tirar conclusões significativas sobre a magnitude do ambiente escolar e social dos alunos na performance escolar.

Os atributos do *dataset* que escolhemos são apresentados da seguinte forma:

1. **school** - nome da escola do aluno;
2. **sex** - o sexo do aluno.
3. **age** -Idade do aluno.
4. **address** - tipo da morada do aluno,atributo binário, U - Urbano ou R - Rural
5. **famsize** - Tamanho da família do aluno,atributo binário, LE3 - 3 ou menos pessoas ou GT3 - Mais de 3 pessoas.
6. **Pstatus** - estado de coabitação dos pais,atributo binário, T - Juntos ou A - Separados.
7. **Medu** - Nível de educação da mãe,(numérico: 0 - nenhuma, 1 - escola primaria, 2 – ensino básico 3 – ensino secundário ou 4 – Licenciatura ou superior) ;
8. **Fedu** - Nível de educação do pai,(numérico: 0 - nenhuma, 1 - escola primaria, 2 – ensino básico 3 – ensino secundário ou 4 – Licenciatura ou superior) ;
9. **Mjob** - Trabalho da mãe,(nominal: "teacher", "health"care related, civil "services"(e.g. administrative or police), "at_home"or "other")
10. **Fjob** - Trabalho do pai,(nominal: "teacher", "health"care related, civil "services"(e.g. administrative or police), "at_home"or "other");

11. **reason** - Razao para escolha da escola,(nominal: close to "home", school "reputation", "course"preference or "other").
12. **guardian** - Encarregado de educação do aluno,(nominal: "mother", "father"or "other").
13. **traveltime** - Tempo entre casa e escola,(numeric: 1 - <15 min., 2 - 15 to 30 min., 3 - 30 min. to 1 hour, or 4 - >1 hour).
14. **studytime** - Tempo de estudo semanal,(numeric: 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4 - >10 hours).
15. **failures** - Numero de disciplinas reprovadas em anos anteriores,(numeric: n if 1<=n<3, else 4).
16. **schoolsup** - Suporte educacional extra,(binary: yes or no).
17. **famsup** - Suporte educacional familiar,(binary: yes or no).
18. **paid** - Pagou aulas extra da disciplina em questao,(binary: yes or no).
19. **activities** - Atividades extra curriculares,(binary: yes or no).
20. **nursery** - Frequentou escola de enfermaria,(binary: yes or no).
21. **higher** - Quer seguir educação de nível superior,(binary: yes or no).
22. **internet** - Tem acesso a Internet em casa,(binary: yes or no).
23. **romantic** - Está numa relação romântica ,(binary: yes or no).
24. **famrel** - Qualidade da relação com a família,(numeric: from 1 - very bad to 5 - excellent).
25. **freetime** - Quantidade de tempo livre depois da escola,(numeric: from 1 - very low to 5 - very high).
26. **goout** - Frequência com que sai com amigos,(numeric: from 1 - very low to 5 - very high).
27. **Dalc** - Quantidade de álcool consumido durante a semana,(numeric: from 1 - very low to 5 - very high).
28. **Walc** -Quantidade de álcool consumido durante o fim de semana,(numeric: from 1 - very low to 5 - very high).
29. **health** - Situação atual de saúde,(numeric: from 1 - very bad to 5 - very good).
30. **absences** - Numero de faltas a aulas,(numeric: from 0 to 93).
31. **G1,G2,G3** - Notas da disciplina de português no final do primeiro, segundo e terceiro período respetivamente

4.2 Análise e Compreensão dos Dados

Previamente à modelação, foi transformado e analisado o *dataset* de forma a obter o melhor modelo possível. As transformações ao *dataset* foram baseadas, tanto em lógica como em análises ao *dataset*.

4.2.1 Limpeza do *Dataset*

O *dataset* escolhido já tinha grande parte das *features* bastante especificadas e por isso para a exploração de dados não fizemos qualquer alteração dos dados, no entanto para a modelação foram feitas alterações para ajudar o modelo a ter um melhor desempenho e desse modo indicar algumas características interessantes. As alterações feitas ao *dataset* para a parte da modelação foram as seguintes:

- Começamos por retirar as colunas G1 e G2 visto que são redundantes devido a coluna G3 ser a média do ano todo e por isso englobar as notas que levaram aos parâmetros G1 e G2
- Para uma melhor precisão do modelo criámos dois *bins* para o atributo G3, dividindo em duas categorias, notas de 0 a 10 e de 10 a 20, sendo posteriormente cada categoria renomeada para "Mau"e "Bom"respetivamente.
- Para concluir a preparação dos dados fizemos uma partição dos dados para o modelo treinar como uma percentagem,80%, e depois ser avaliado com os restantes 20%.

4.2.2 Análise ao *Dataset*

O objectivo deste *dataset* é, tendo em conta os vários factores sociais e demográficos, prever a nota de português no final do ano lectivo, parâmetro G3.Este parâmetro inicialmente é numérico e pode ir de 0 a 20 valores, visto isto fizemos uma analise com o auxilio de um *boxplot* ao atributo G3, e identificamos que 50% das notas dos alunos estavam entre 10 e 14 valores.Foi devido a esta análise do *boxplot* e a acharmos que os restantes atributos do *dataset* não terem um correlação forte o suficiente com o sucesso escolar, no contexto em que seria muito difícil para o modelo distinguir um aluno de 12 a outro de 15 só com parâmetros sociais e demográficos visto que retiramos as colunas que continham a notas do primeiro e segundo período, que decidimos através de um *numeric binner* dividir as notas em duas categorias, Mau (0-10) e Bom (10-20).Isto permite ao modelo ser bastante mais preciso e da mesma maneira poder tirar conclusões sobre atributos que tenham impacto no sucesso escolar.

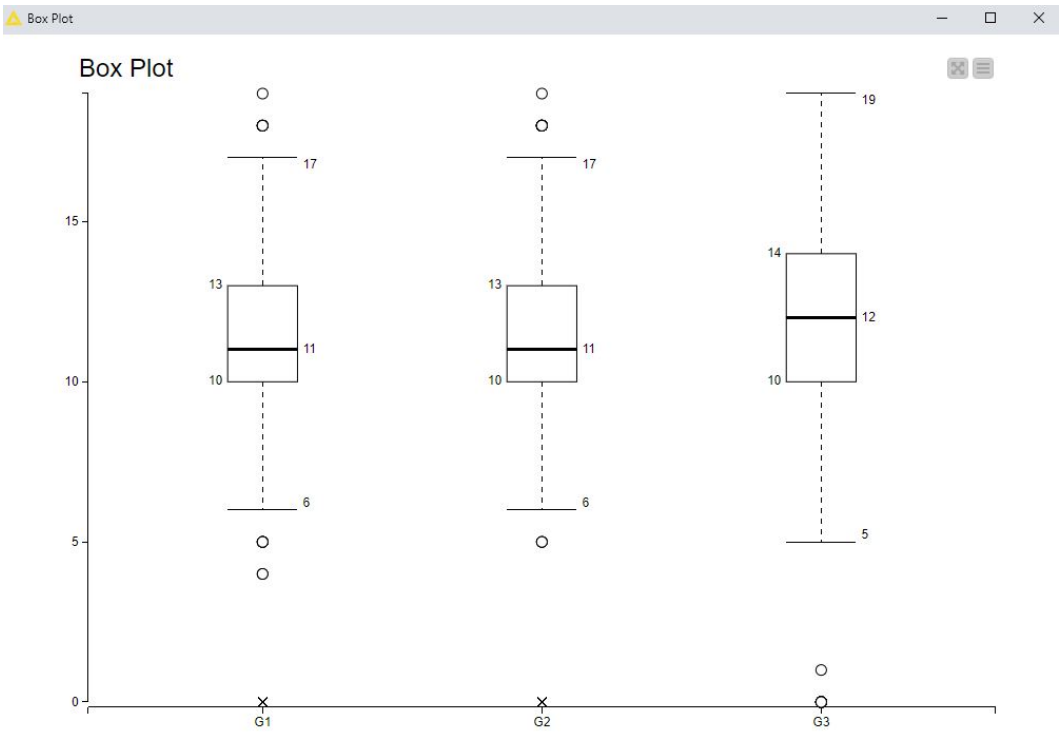

Figura 21: Distribuição das notas dos alunos nos 3 períodos

A seguinte Matriz de correlação confirma a preposição que o grupo fez anteriormente sobre os atributos terem pouca correlação com o sucesso escolar,excluindo o atributo "*failures*", que representa o numero de disciplinas reprovadas em anos anteriores.

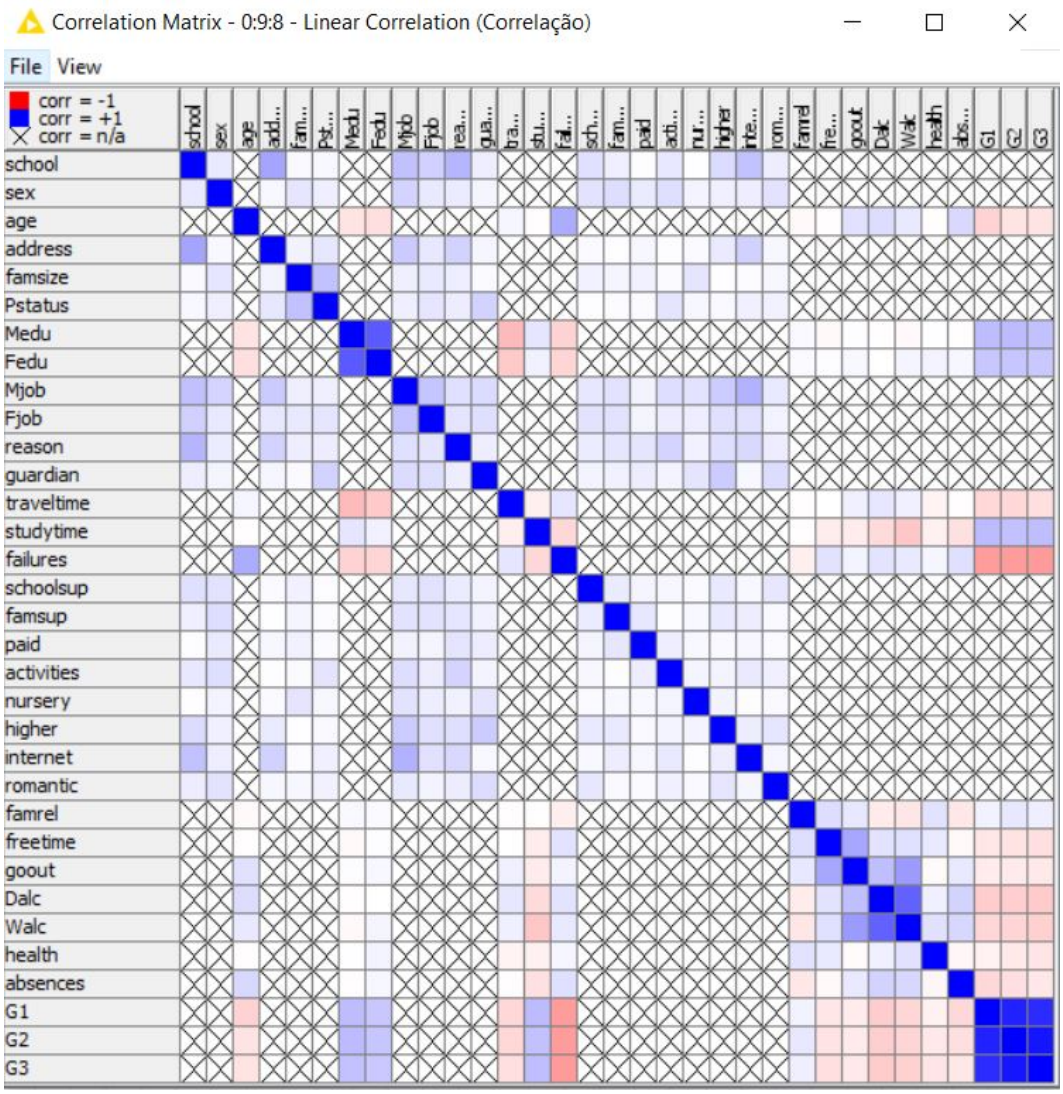

Figura 22: Matriz de correlação do dataset1

Em relação aos restantes atributos, que são 30, excluindo G1 G2 G3,verificamos que 13 eram binários e outros 13 numéricos sendo que desses 13 numéricos 11 eram atributos de frequência de 0 a 5.Deste modo concluímos que seria difícil adicionar mais atributos relevantes para o modelo visto que os existentes já eram bastantes específicos e por isso não foi criada mais nenhuma coluna após analisarmos o *dataset* em profundidade.

4.3 Modelação

Neste secção vamos explicar a modelação do dataset1, como funciona o *workflow* na generalidade e qual foi o nosso foco quando produzimos o modelo.

Para melhor organização e manutenção do *workflow* dividimos o mesmo em 3 ambientes e em cada ambiente usamos metanodos para cada componente específica do ambiente em questão.

Os ambientes do *workflow* do *dataset* 1 são:´

1. *Data Preparation Environment* - Consiste no ambiente em que os dados são carregados e acontece alterações ao *dataset* necessárias, limpeza de dados por exemplo;

2. *Analytics Environment* - Consiste no ambiente em que são analisados as melhores *features* do *dataset* e também são exploradas eventuais relações entre atributos;

3. *Model Optimization Environment* - Consiste no ambiente em que os parâmetros do modelo são optimizados para uma melhor *accuracy*.

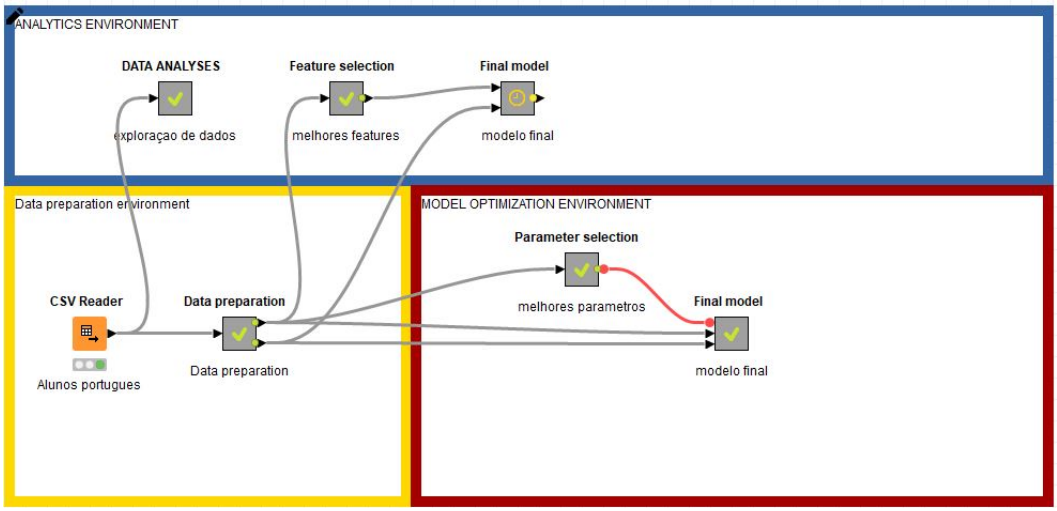

Figura 23: Arquitetura principal do *workflow* do *dataset* 1

4.3.1 *Data Preparation Environment*

Este ambiente é a onde o ficheiro do *dataset* é carregado para o *Knime* através de um *csv reader* e de seguida no *metanode Data Preparation* os dados são preparados para posterior modelação.

No *metanode Data Preparation* é feita a filtragem das colunas $G1$ e $G2$, devido a sua redundância com $G3$, de seguida é aplicado o *numeric binner* ao atributo $G3$ passando a ter duas categorias *Mau(0-10)* e *Bom (10-20)*.Por fim é feita a partição dos dados visto que este *workflow* só tem um *dataset* e então é necessário usar um *partitioning* para o modelo treinar uma percentagem de dados e testar com a restante.

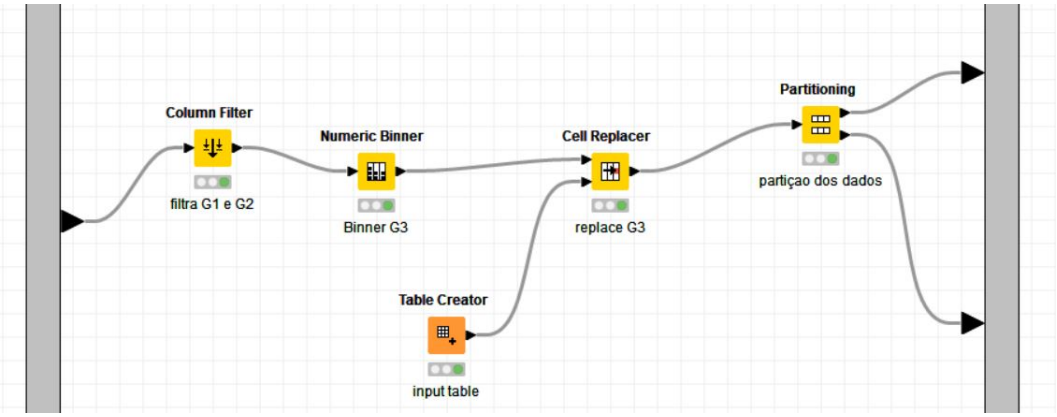

Figura 24: Arquitectura do *Metanode Data Preparation*

4.3.2 *Analytics environment*

Este ambiente consiste num *metanode* para exploração de dados, que vai ser explicado em profundidade mais a frente, e dois *metanodes* que fazem o ciclo para seleção das melhores *features*. No caso dos dois metanodos, que fazem a *feature selection*, vamos focar no que tem o ciclo que faz a *feature selection* que está representado na próxima figura.

Esta componente foi essencial para a *accuracy* do modelo aumentar drasticamente e também para potenciar posteriormente uma melhor exploração dos dados visto que nos indicou algumas features a explorar. A estratégia que usamos para este modelo foi usar a forward feature selection visto que a analise de dados não nos proporcionou grandes conclusões de quais podiam ser as features importantes para o modelo,excluindo o parâmetro failures. De seguida usamos o algoritmo de Backwards para as as melhores features devolvidas pelo modelo e assim sucessivamente sempre a procura da melhor acuracy do modelo.

Para uma breve descrição do modelo visto que é bastante idêntico ao usado para o dataset da competição, tem a primeira parte a onde decide os parâmetros, em que são filtrados parâmetros que o método forward não escolheu como óptimos. De seguida é

feita a cross validation para termos uma maior confiança nos resultamos que obtivemos, fazemos a media do erro e fixamos como objetivo a minimização do mesmo.Por fim é simplesmente escolhida as features com menores percentagens de erro para o dado modelo.

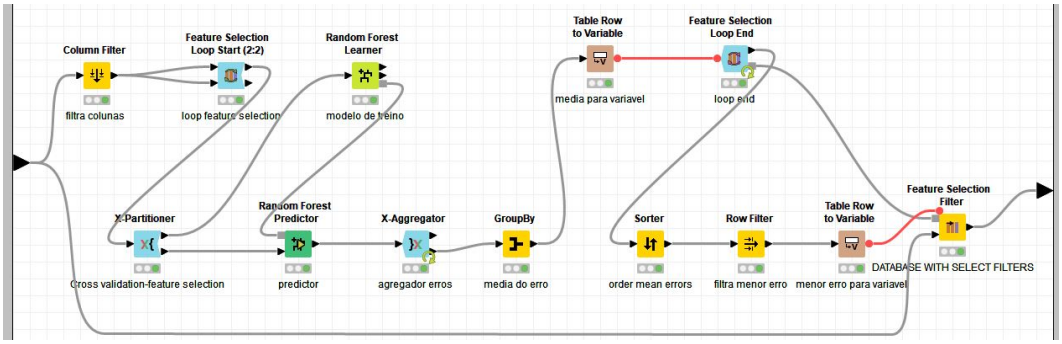

Figura 25: Arquitectura do *Metanode Feature Selection*

4.3.3 *Model Optimization Environment*

Neste ambiente é feita a optimização dos parâmetros especificamente no metanodo Parameter Selection. Nesse mesmo metanodo, próxima imagem, o nosso foco foi melhorar os parâmetros de Numero de modelos e o critério de divisão pois achamos que o foco nesses dois parâmetros nos traria o melhor modelo.

Numa breve explicação do metanodo visto que é idêntico ao do modelo da competição, numa primeira fase temos a filtragem das features, em que só entram no modelo as que tiveram melhores resultados na parte da feature optimization. Depois temos a decisão dos parâmetros e em que intervalos vão ser testados. De seguida tem a cross validation do modelo para garantir confiança nos resultados que temos. Por fim faz a media do erro, faz a conversão de erro para accuracy e escolhe os parâmetros com melhor accuracy para o modelo em questão.

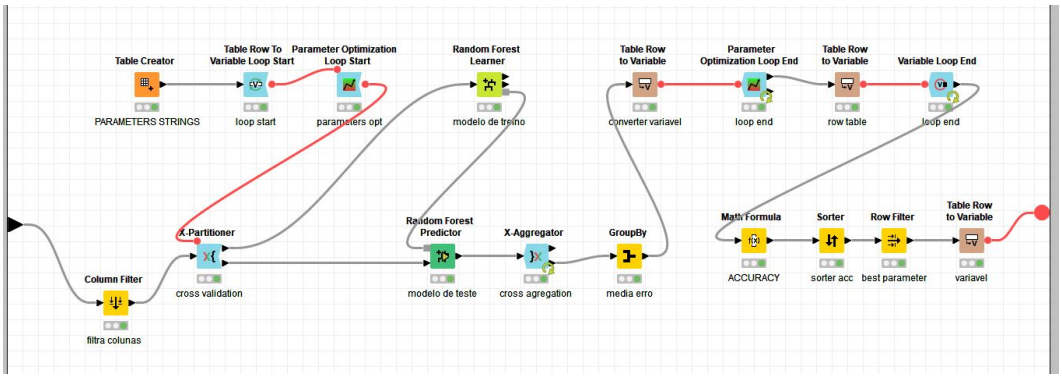

Figura 26: Arquitectura do *Metanode Parameter Selection*

4.4 Exploração dos dados

Na exploração dos dados, o foco principal foi procurar encontrar relações entre os vários atributos e o impacto que estes poderão ter, ou não, no desempenho escolar dos alunos.

4.4.1 Sexo dos alunos

Row ID	S Sexo	D Média G1	D Média G2	D Média G3	D Média das Relações Familiares	I Número de ocorrências
Row0	F	11.637	11.822	12.253	3.864	383
Row1	M	11.056	11.207	11.406	4.026	266

Figura 27: Relação entre o sexo dos alunos e as suas notas

É possível verificar que existe uma diferença significativa entre o sexo dos alunos e notas obtidas pelos mesmo, sendo que os alunos do sexo feminino apresentam uma média final 0.847 décimas mais alta da dos alunos do sexo masculino. Procuramos ainda verificar se o sexo dos alunos também se reflete numa diferença de relações familiares, no entanto a diferença da média é relativamente baixa, sendo que os alunos do sexo masculino reportaram uma relação com a sua família marginalmente superior à dos alunos do sexo feminino.

4.4.2 Álcool Consumido

Row ID	I Álcool consumido no fim de semana	D Média G1	D Média G2	D Média G3	I Número de ocorrências
Row0	1	11.688	11.951	12.36	247
Row1	2	11.747	11.82	12.26	150
Row2	3	11.308	11.5	11.667	120
Row3	4	10.655	10.736	11.034	87
Row4	5	10.333	10.444	10.556	45

Figura 28: Relação entre o álcool consumido durante o fim de semana pelos alunos e as suas notas

Row ID	I Álcool consumido semanalmente	D Média G1	D Média G2	D Média G3	I Número de ocorrências
Row0	1	11.698	11.896	12.299	451
Row1	2	11.091	11.157	11.364	121
Row2	3	10.442	10.744	11.14	43
Row3	4	9.941	9.471	8.941	17
Row4	5	9.529	10.059	10.235	17

Figura 29: Relação entre o álcool consumido durante a semana pelos alunos e as suas notas

Row ID	Relações familiares	Média de alcool durante a semana	Média de alcool durante o fim de semana	Média G1	Média G2	Média G3	Número de ocorrências
Row0	1	1.909	2.409	10.409	10.273	10.636	22
Row1	2	1.724	2.655	11.034	10.448	10.862	29
Row2	3	1.485	2.485	11.139	11.208	11.594	101
Row3	4	1.476	2.246	11.678	11.959	12.344	317
Row4	5	1.472	2.15	11.233	11.428	11.633	180

Figura 30: Relação entre as relações familiares dos alunos e a quantidade de álcool que consomem

Como seria de esperar, os alunos que consomem mais álcool apresentam piores resultados em termos de notas, com uma diferença entre os que não consomem e os que mais consomem de quase 2 valores relativos ao nível de álcool consumido durante o fim de semana, e ligeiramente mais de 2 valores relativos à quantidade de álcool consumido durante a semana.

Como também era esperado, alunos que apresentem piores relações familiares consomem, em média, quantidades maiores de álcool, e apresentam resultados escolares mais baixos.

4.4.3 Nível de educação dos pais

Row ID	Nivel de educação da mãe	Média G1	Média G2	Média G3	Número de ocorrências
Row0	0	10.833	11.167	11.667	6
Row1	1	10.399	10.524	10.797	143
Row2	2	11.204	11.285	11.661	186
Row3	3	11.324	11.561	11.921	139
Row4	4	12.503	12.749	13.069	175

Figura 31: Relação entre o nível de educação da mãe e as notas dos alunos

Row ID	Nível de educação do pai	Média G1	Média G2	Média G3	Número de ocorrências
Row0	0	11.429	11.571	12.143	7
Row1	1	10.523	10.695	10.937	174
Row2	2	11.368	11.411	11.785	209
Row3	3	11.733	11.992	12.382	131
Row4	4	12.297	12.586	12.922	128

Figura 32: Relação entre o nível de educação do pai e as notas dos alunos

Row ID	Relações Familiares	Média do nível de educação da mãe	Média do nível de educação do pai	Número de ocorrências
Row0	1	2.409	2.136	22
Row1	2	2.621	2.448	29
Row2	3	2.455	2.277	101
Row3	4	2.498	2.29	317
Row4	5	2.572	2.35	180

Figura 33: Relação entre o nível de relações familiares e o nível de educação dos pais

Row ID	Média do nível de educação da mãe	Média do nível de educação do pai
Row0	2.515	2.307

Figura 34: Média do nível de educação dos pais

Nestas tabelas procuramos descobrir se existe alguma relação entre o nível de educação de cada um dos pais com os resultados académicos dos alunos, e, caso exista, se o nível de educação de um dos pais tem em média maior impacto que o outro.

Como podemos ver na figura 31, o nível de educação da mãe parece ter um impacto bastante significativo nas notas dos filhos, sendo que alunos em que a mãe apenas tem nível 1 (ensino primário), têm uma média 2,272 valores abaixo dos alunos em que a mãe tem nível 4 (ensino superior).

Do mesmo modo, podemos ver na figura 32, que alunos em que o pai apenas tem nível 1, têm uma média 2 valores abaixo daqueles em que o pai tem nível 4. Como existem poucos casos em que o pai ou mãe têm nível 0 de ensino (nenhum ensino) os dados estatísticos são pouco viáveis e não podemos tirar conclusões.

Na figura 27, podemos ver que no geral, os alunos que têm melhores relações familiares têm, em média, pais com maior nível de ensino, sendo que existe uma exceção entre os níveis 2 e 3, onde o nível médio de educação dos pais diminui, mas isso também se pode dever ao facto de haver apenas 29 casos de alunos que classificaram a sua relação familiar como sendo de nível 2, em comparação com os 101 que reportaram nível 3.

Por fim, na figura 34, podemos ver que existe uma pequena diferença entre o nível de educação das mães e dos pais dos alunos, sendo que as mães possuem em média um nível de educação ligeiramente superior ao dos pais, o que, em conjunto com a figura 21, reforça a ideia que indivíduos do sexo feminino apresentam em média resultados escolares superiores aos indivíduos do sexo masculino.

4.4.4 Elementos auxiliares

Para além das tabelas mencionadas em cima, foram ainda utilizados nodos para ajudar na visualização dos dados, nomeadamente um *heatmap* para ter uma forma visual de observar a distribuição dos atributos numéricos do nosso *dataset*, assim como um *data explorer* para ter acesso às várias componentes estatísticas de cada atributo assim como um histograma da sua distribuição.

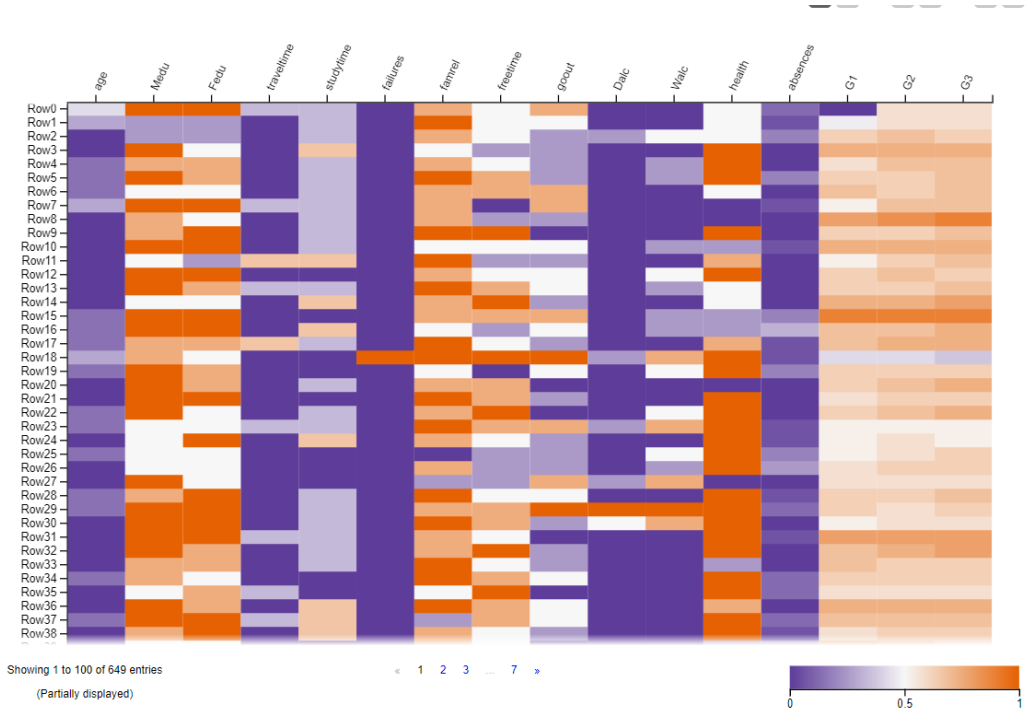

Figura 35: *Heatmap* dos atributos normalizados

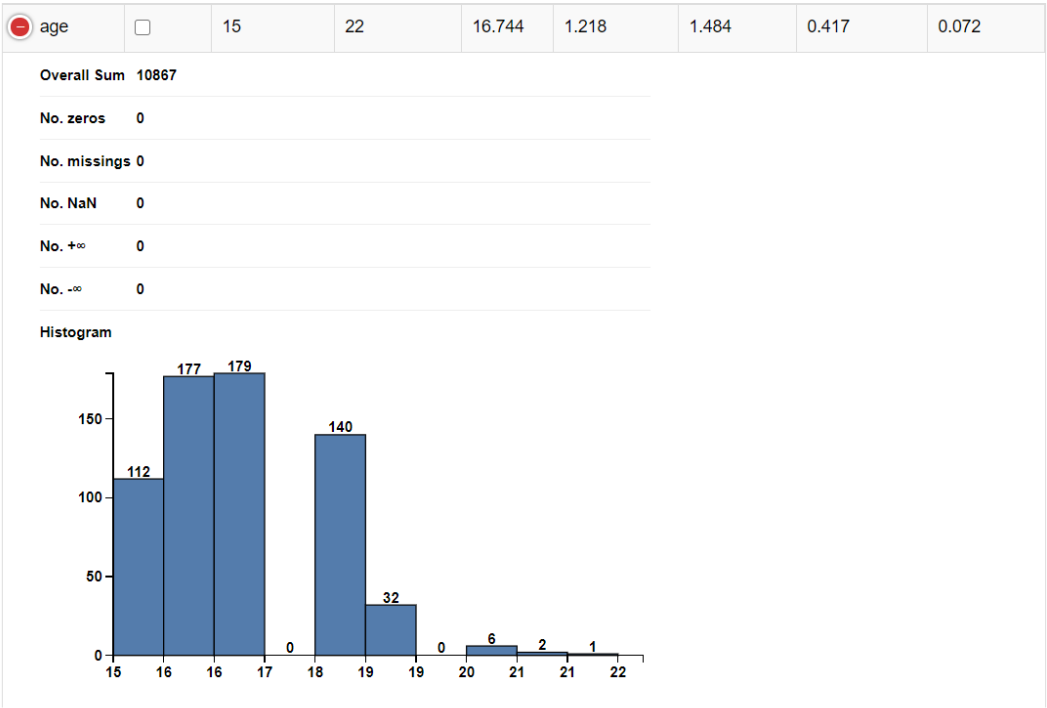

Figura 36: Resultados do nodo *Data Explorer* para o atributo da idade

4.5 Análise de Resultados

Os melhores resultados que obtivemos foram sempre no intervalo de 88%. Isto deveu-se à dificuldade do modelo em prever notas da categoria Mau(0-10), exemplificado pela próxima imagem. Isto deve-se no nosso entendimento, como já foi referido anteriormente, a 50% das notas estarem no intervalo de 10 a 14 valores o que faz que com seja difícil para o modelo diferenciar uma negativa alta, por exemplo 9 valores que é catalogada como Mau, de uma positiva baixa, por exemplo 10 que é catalogada como Bom.Para obtermos estes resultados com 701 Modelos e com o critério de divisão o *InformationGainRatio*.

Para concluir achamos que o nosso modelo prevê muito bem alunos bons, visto que no exemplo da imagem ele tem apenas 1 falso Negativo e 110 corretos, e ficamos com a sensação que embora pudéssemos ter feito um melhor trabalho a prever os casos negativos o modelo vai ter sempre problemas a identificar notas negativas altas, [8-10[, como "Mau"visto que maior parte das notas estão acima dos 10 valores.

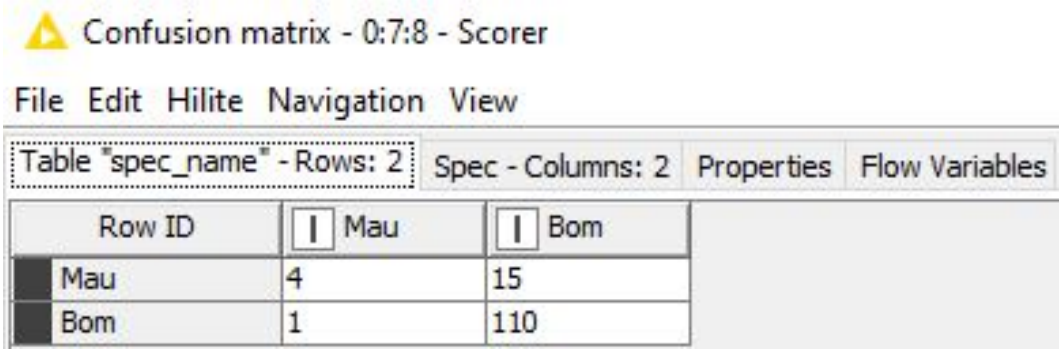

Figura 37: Matriz de confusão

5 Conclusão

Em suma, o trabalho realizado utilizando a ferramenta *Knime* envolveu dois *datasets* independentes resultando em dois *workflows*, em que o objectivo de ambos era desenvolver modelos baseados em árvores usando diferentes técnicas como *cross validation* e *feature selection* com o objectivo sempre de encontrar o melhor modelo possível para o *dataset* em questão. O grupo acha que em ambos os *datasets* fez um trabalho satisfatório embora tenha a perfeita noção que seria possível melhorar os modelos produzidos.

O grupo consciencializou-se da importância da análise dos dados, visto que foi um passo fulcral principalmente no desenvolvimento do modelo da competição pois permitiu uma limpeza e transformação do *dataset* extraindo mais dados e permitindo ao modelo ter mais informação e assim conseguir melhores resultados.