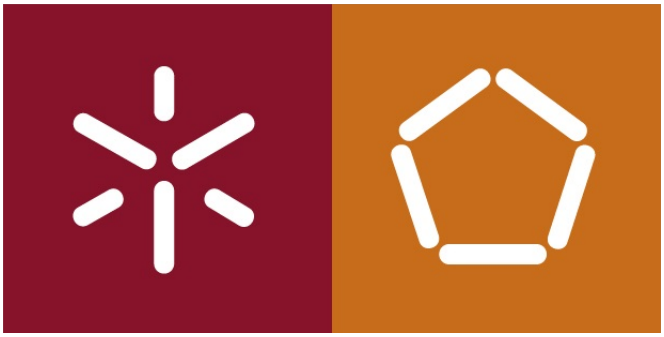


Universidade do Minho
Departamento de Informática

Sistemas Baseados em Similaridade

Trabalho Prático Individual 7

Gonçalo Almeida (A84610)

Dezembro 2020

1 Fase 1

1.1 *Open AQ Platform*

A primeira tarefa a realizar para este trabalho prático foi, após analisar a plataforma *Open AQ*, obter todas as cidades portuguesas que esta disponibiliza. Para isto, atendendo à *API* da plataforma, criou-se um nodo *Get Request* com o respetivo URL.

Figure 1: Nodo *Get Request*

O *JSON* obtido na coluna *body* foi transformado numa tabela com o nodo *JSON to Table* e foram removidas todas as colunas exceto as dos resultados. De seguida foi obtida a transposta e, novamente, aplicado o *JSON to Table*.

Row ID	country	name	city	count	locations
results (#1)	PT	Aveiro	Aveiro	385333	4
results (#2)	PT	Braga	Braga	149788	3
results (#3)	PT	Castelo Branco	Castelo Branco	118875	1
results (#4)	PT	Coimbra	Coimbra	177790	3
results (#5)	PT	Évora	Évora	130508	1
results (#6)	PT	Faro	Faro	379740	4
results (#7)	PT	Ilha da Madeira	Ilha da Madeira	321590	3
results (#8)	PT	Ilha do Faial	Ilha do Faial	143012	1
results (#9)	PT	Leiria	Leiria	134375	1
results (#10)	PT	Lisboa	Lisboa	1288997	15
results (#11)	PT	Porto	Porto	916685	15
results (#12)	PT	Santarém	Santarém	113011	1
results (#13)	PT	Setúbal	Setúbal	1217327	12
results (#14)	PT	Viana do Castelo	Viana do Castelo	72726	1
results (#15)	PT	Vila Real	Vila Real	107862	1
results (#16)	PT	Viseu	Viseu	81580	1

Figure 2: Cidades portuguesas disponibilizadas pela *Open AQ*

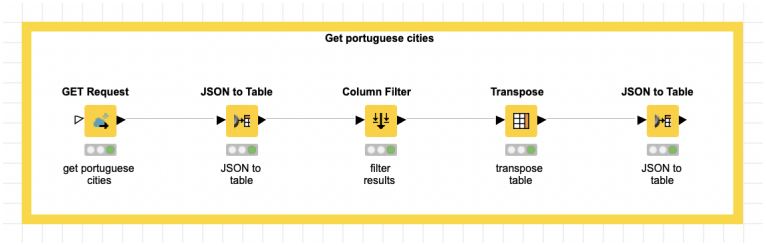


Figure 3: Workflow

Para a próxima tarefa foi necessário obter os níveis de ozono para as cidades anteriormente obtidas. Foi necessário formatar os nomes das cidades de modo a que os espaços fossem substituídos pelo *encode* "%20" e gerar um *request* para cada cidade.

Row ID	S o3_request
results (#1)	https://api.openaq.org/v1/latest?city=Aveiro&parameter=o3&limit=1
results (#2)	https://api.openaq.org/v1/latest?city=Braga&parameter=o3&limit=1
results (#3)	https://api.openaq.org/v1/latest?city=Castelo%20Branco&parameter=o3&limit=1
results (#4)	https://api.openaq.org/v1/latest?city=Coimbra&parameter=o3&limit=1
results (#5)	https://api.openaq.org/v1/latest?city=Évora&parameter=o3&limit=1
results (#6)	https://api.openaq.org/v1/latest?city=Faro&parameter=o3&limit=1
results (#7)	https://api.openaq.org/v1/latest?city=Ilha%20da%20Madeira&parameter=o3&limit=1
results (#8)	https://api.openaq.org/v1/latest?city=Ilha%20do%20Faial&parameter=o3&limit=1
results (#9)	https://api.openaq.org/v1/latest?city=Leiria&parameter=o3&limit=1
results (#10)	https://api.openaq.org/v1/latest?city=Lisboa&parameter=o3&limit=1
results (#11)	https://api.openaq.org/v1/latest?city=Porto&parameter=o3&limit=1
results (#12)	https://api.openaq.org/v1/latest?city=Santarém&parameter=o3&limit=1
results (#13)	https://api.openaq.org/v1/latest?city=Setúbal&parameter=o3&limit=1
results (#14)	https://api.openaq.org/v1/latest?city=Viana%20do%20Castelo&parameter=o3&limit=1
results (#15)	https://api.openaq.org/v1/latest?city=Vila%20Real&parameter=o3&limit=1
results (#16)	https://api.openaq.org/v1/latest?city=Viseu&parameter=o3&limit=1

Figure 4: *Requests* gerados para cada cidade

Figure 5: Nodo *Get Request*

Após o nodo *Get Request* o processo é semelhante ao da tarefa anterior. De seguida são filtradas as colunas de interesse, é realizado o *cast* do valor de ozono de *double* para inteiro e a tabela é ordenada por este valor.

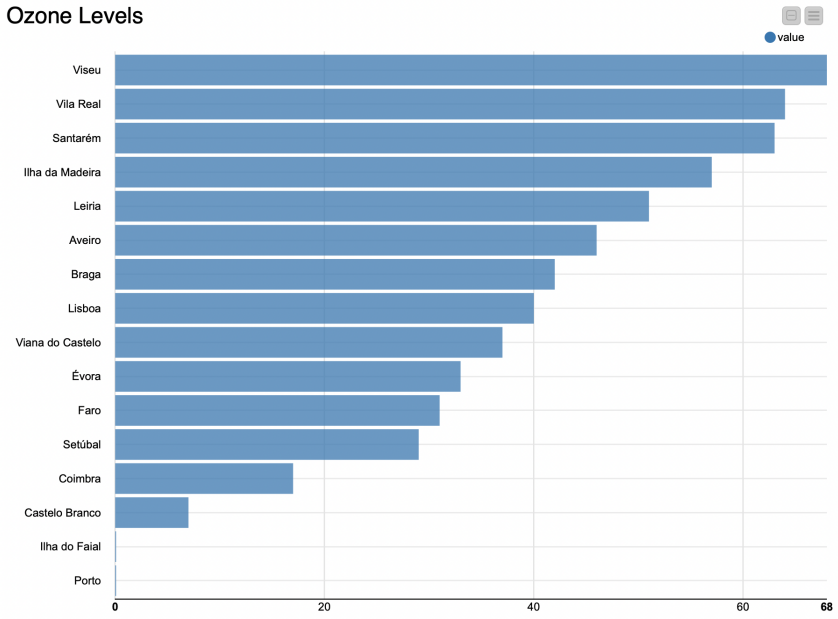
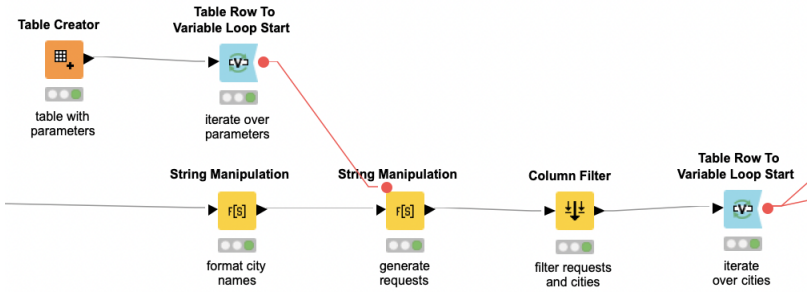
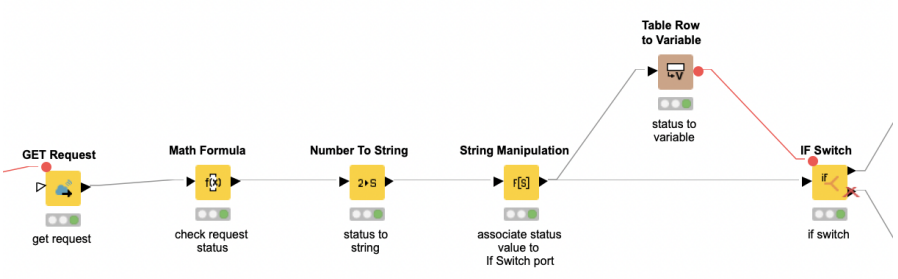


Figure 6: Níveis de ozono nas cidades portuguesas

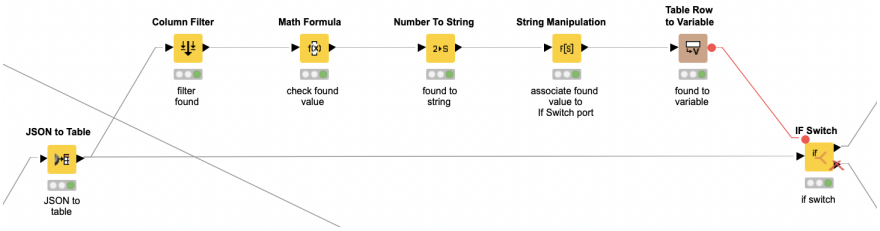
Não ficando apenas pelo ozono, foram obtidos também os níveis de outros parâmetros ambientais. Os primeiros passos foram formatar os nomes das cidades, criar uma tabela com alguns dos parâmetros disponibilizados pela plataforma e criar um ciclo, iterando sobre estes. Para cada parâmetro são gerados os pedidos referentes a cada cidade e é criado um novo ciclo que itera sobre cada pedido. Estes dois ciclos foram criados de modo a poder aplicar um nodo *Wait...* para não sobrecarregar o servidor de pedidos.



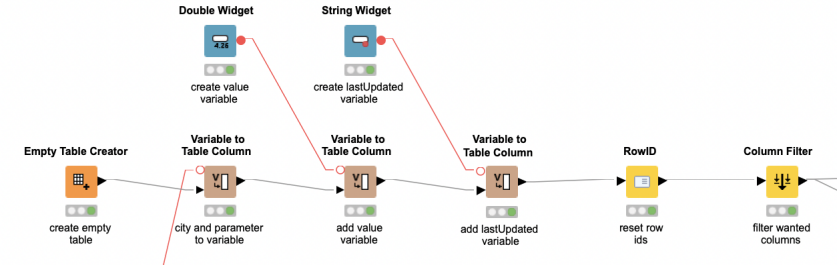
Após receber a resposta ao pedido enviado, é verificado se o *HTTP Request Status* é 500 pois, no decorrer deste trabalho, foi algo que aconteceu multiplas vezes.



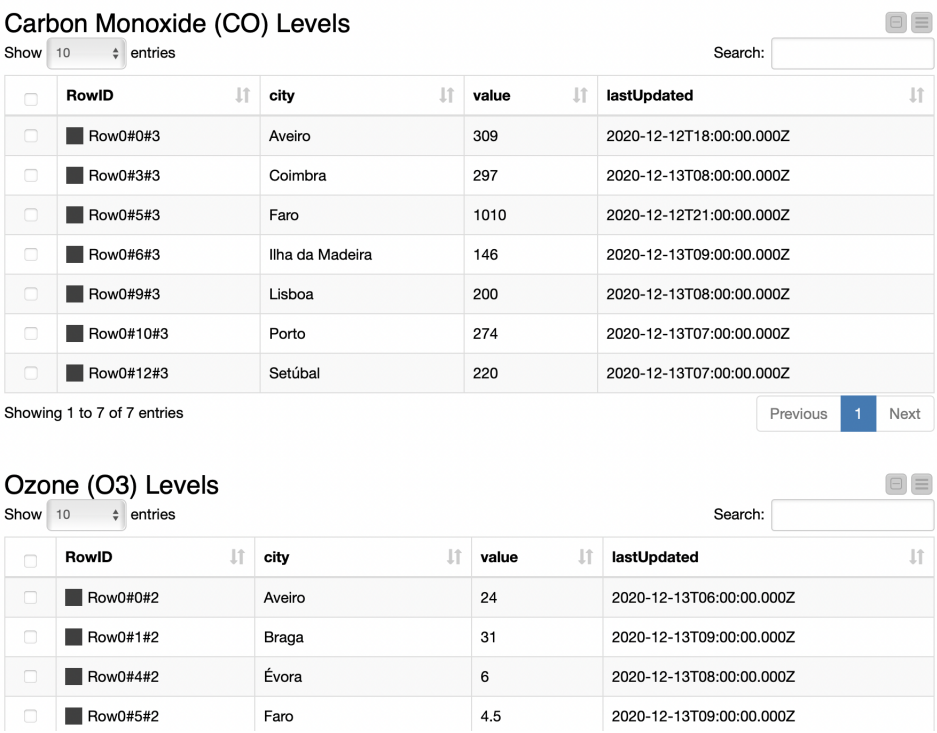
Caso o *Status* não seja 500, é validado o valor na coluna *found* da resposta, isto é, se existem dados relativos ao pedido feito.



No caso de falharem ambas as validações, é criada uma tabela com valores *default* para as colunas que se pretende obter.



No final, são aplicadas *views* interativas de forma a separar e melhor vizualizar os dados obtidos.



Carbon Monoxide (CO) Levels

Show 10 entries Search:

RowID	city	value	lastUpdated
Row0#0#3	Aveiro	309	2020-12-12T18:00:00.000Z
Row0#3#3	Coimbra	297	2020-12-13T08:00:00.000Z
Row0#5#3	Faro	1010	2020-12-12T21:00:00.000Z
Row0#6#3	Ilha da Madeira	146	2020-12-13T09:00:00.000Z
Row0#9#3	Lisboa	200	2020-12-13T08:00:00.000Z
Row0#10#3	Porto	274	2020-12-13T07:00:00.000Z
Row0#12#3	Setúbal	220	2020-12-13T07:00:00.000Z

Showing 1 to 7 of 7 entries Previous 1 Next

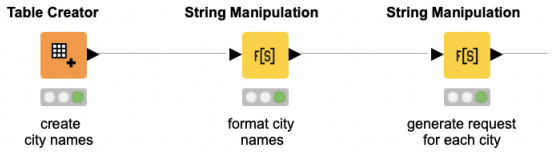
Ozone (O3) Levels

Show 10 entries Search:

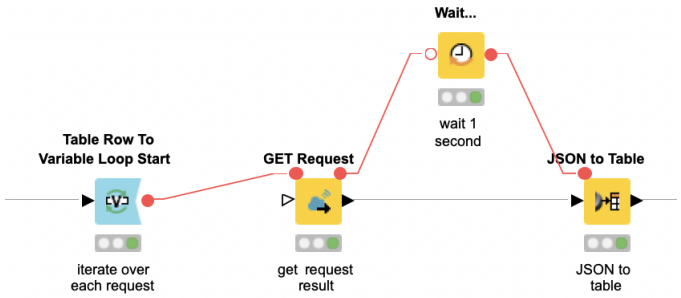
RowID	city	value	lastUpdated
Row0#0#2	Aveiro	24	2020-12-13T06:00:00.000Z
Row0#1#2	Braga	31	2020-12-13T09:00:00.000Z
Row0#4#2	Évora	6	2020-12-13T08:00:00.000Z
Row0#5#2	Faro	4.5	2020-12-13T09:00:00.000Z

1.2 *OpenWeatherMaps*

De modo a obter e vizualizar dados relativos a outras plataformas, foi escolhida a *OpenWeatherMaps* com o propósito de obter as temperaturas atuais em várias cidades portuguesas. Criou-se uma tabela com as cidades pretendidas e, com esta, geraram-se os *requests*.



De forma a não atingir o limite de pedidos ao servidor por unidade de tempo, criou-se um ciclo, iterando sobre cada *request*, que após cada um ser realizado, um nodo *Wait...* aplica a espera de 1 segundo.



Há que notar que para esta plataforma, ao contrário da anterior, foi necessário incluir alguns *headers* aos pedidos realizados.

Template:

Header Key	Header value	Value kind
x-rapidapi-key	40284daf08msh92aa7e140401892p124569j...	Constant
x-rapidapi-host	community-open-weather-map.p.rapidapi.com	Constant

Tal como nos casos anteriores, a resposta é tratada da mesma forma, convertendo estruturas *JSON* em tabelas e filtrando colunas de interesse, obtendo os dados pretendidos.

cidade	atual	min	max
Porto	16.11	13.32	16.11
Lisboa	17.13	15.25	17.26
Braga	15.28	11.67	15.28
Coimbra	16.86	13.32	17.15
Porto	16.11	13.32	16.11
Lisboa	17.13	15.25	17.26
Braga	15.28	11.67	15.28
Coimbra	16.86	13.32	17.15
Faro	17.73	14	17.73
Guimarães	15.59	12.5	15.59

Figure 7: Temperaturas nas cidades portuguesas

2 Fase 2

Na segunda fase deste trabalho prático foi necessário selecionar e tratar um dataset sobre o qual deverá ser aplicado um método de *clustering*. O *dataset* escolhido foi obtido na plataforma Kaggle e resume o comportamento de uso de cerca de 9000 titulares de cartões de crédito ativos durante um período de 6 meses.

2.1 Identificação Visual de *Clusters*

De forma a tentar identificar, visualmente, clusters no *dataset* foram aplicados diagramas de dispersão com o auxílio do nodo *Scatter Matrix (local)*.

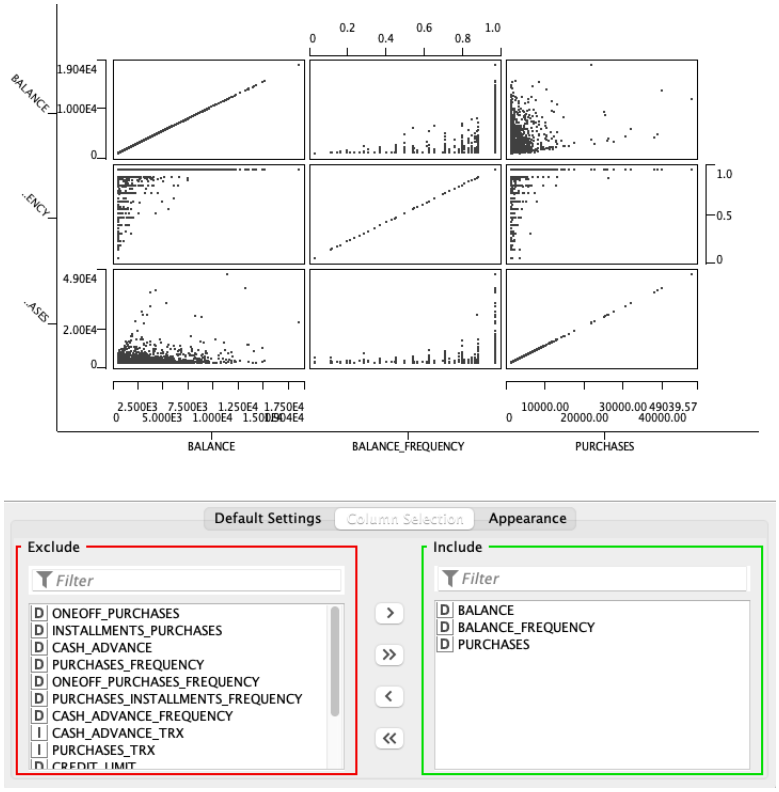


Figure 8: Diagramas de dispersão

Foi também aplicada uma Análise de Componentes Principais (PCA) de forma a projetar os dados em apenas duas dimensões.

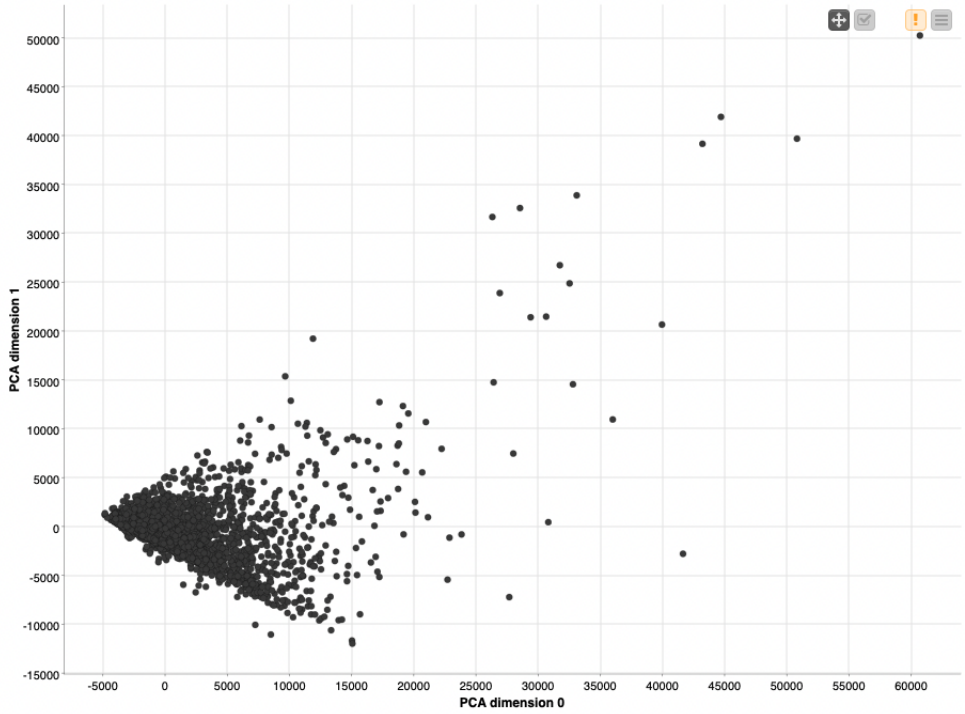


Figure 9: Análise de Componentes Principais

Observando os gráficos obtidos, pode-se, possivelmente, identificar dois clusters, um contendo a maior parte do aglomerado de pontos à esquerda e outro que contendo os pontos mais dispersos à direita.

2.2 Método do Cotovelo

O próximo passo foi aplicar o método do cotovelo de forma a identificar o número ótimo de *clusters*. Este método foi aplicado utilizando dois métodos distintos de *clustering*, *k-Medoids* e *k-Means* e a métrica *Mean Average Error (MAE)* como medida de qualidade.

Começou-se por normalizar os valores dos atributos e criar um ciclo que itera sobre o número de *clusters*, neste caso, entre 1 e 12. Há que ter em conta que, como a variável de iteração começa com valor 0, foi necessário criar uma nova com a ajuda do nodo *Java Edit Variable* que comece com o valor 1.

Enquanto que o método *k-Means* consegue tratar o *dataset* completo num curto período de tempo, no método *k-Medoids* isto não acontece, sendo que foi necessário reduzir o conjunto de dados. Para tal foi aplicado o nodo *Shuffle* para obter uma amostra de forma aleatória.

Após ser aplicado o método de *clustering* é calculado o *Mean Average Error (MAE)* para cada entrada.

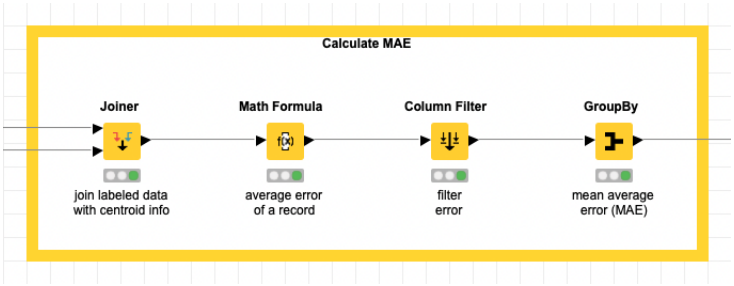


Figure 10: Cálculo do MAE

Tendo, então, os valores do *MAE* para cada número de *clusters* k, é possível visualizar o "cotovelo" com o nodo *Scatter Plot*.

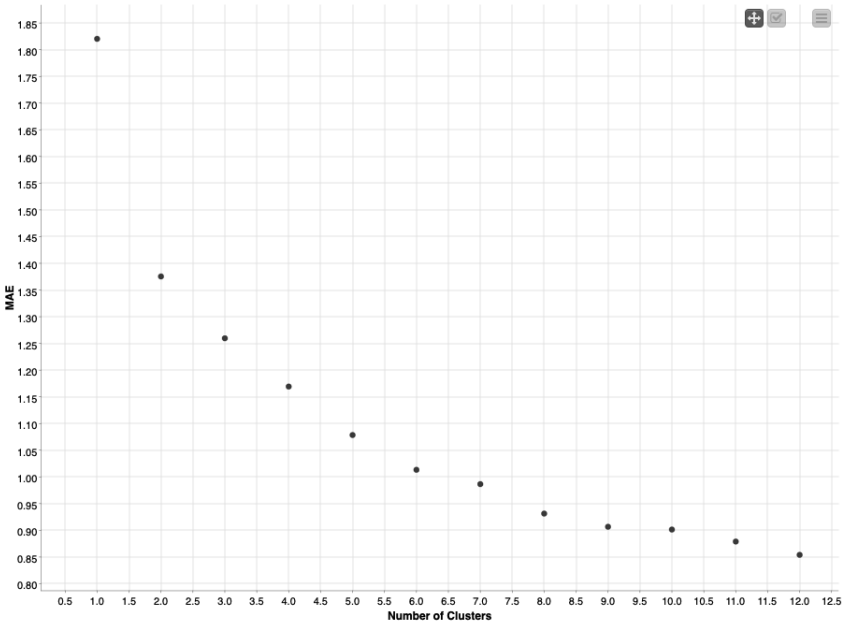


Figure 11: Cotovelo do método k-Medoids

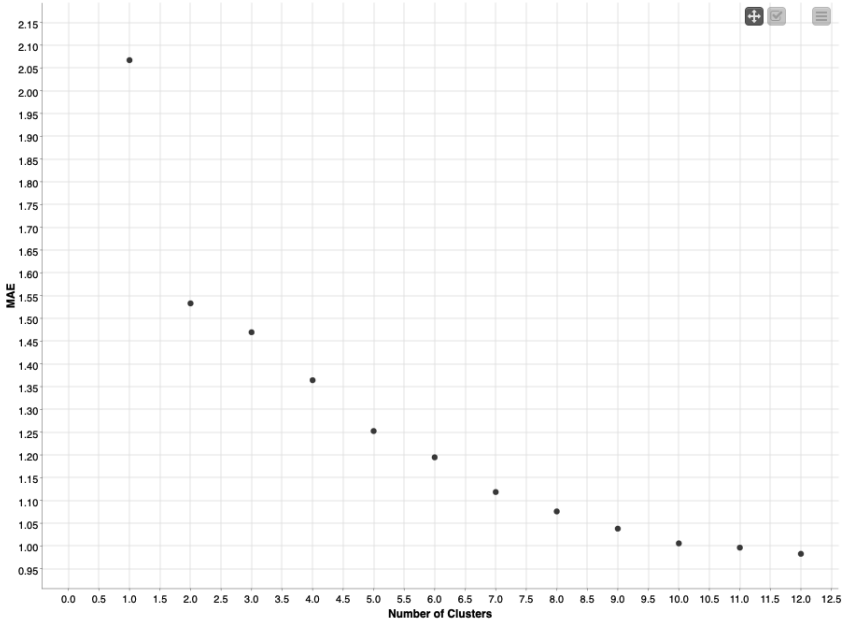


Figure 12: Cotovelo do método k-Means

De forma a calcular o número ótimo de *clusters*, é calculada a maior diferença entre os valores de *MAE* entre os diferentes valores de k.

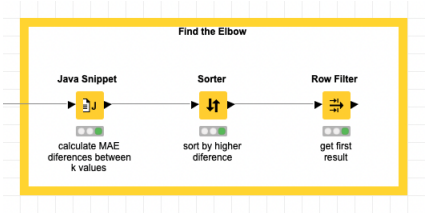


Figure 13: Cálculo do número ótimo de *clusters*

	MAE	k ótimo
k-Medoids	1.375	2
k-Means	1.533	2

Figure 14: *Workflow k-Medoids*

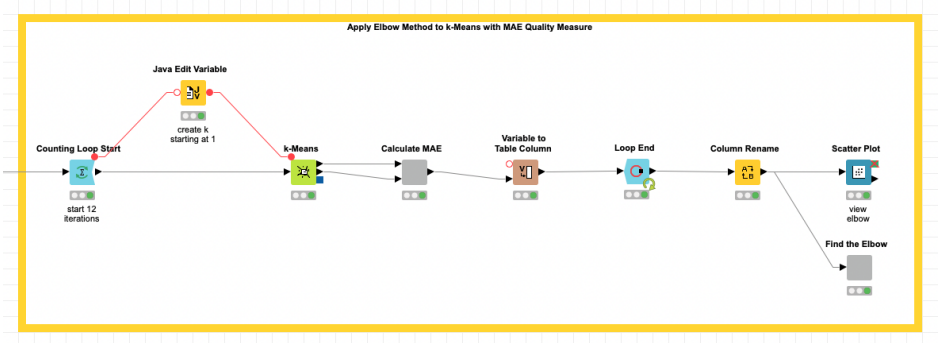


Figure 15: *Workflow k-Means*

2.3 Método de *Clustering* Interativo

Por fim, foi desenvolvido um *workflow* que permite o utilizador definir o número de *clusters* a ser utilizado pelo método *k-Medoids*.

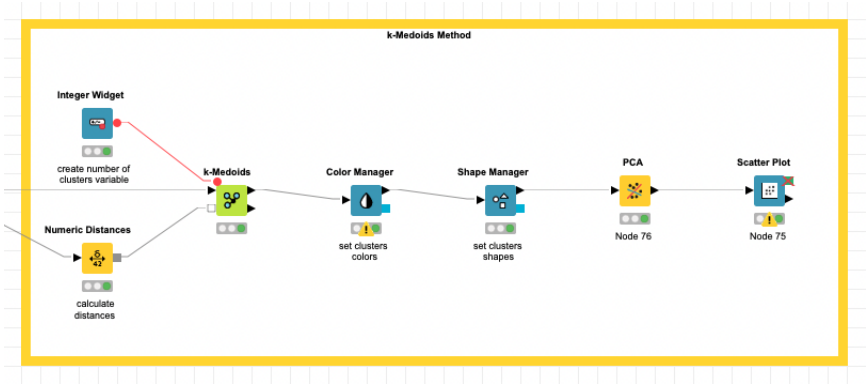


Figure 16: Workflow *k-Medoids*

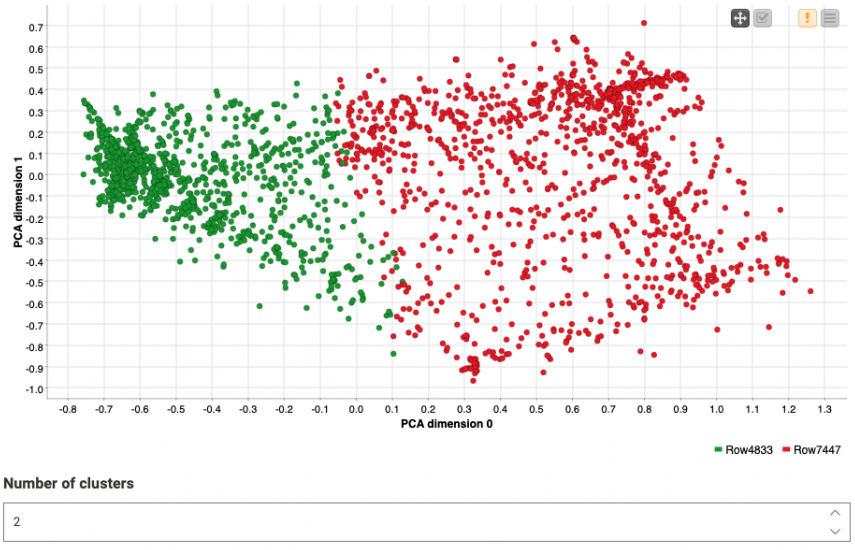


Figure 17: *View* interativa