Universidade do Minho
Escola de Engenharia

Modelos de Aprendizagem

Trabalho Prático
Aprendizagem e Decisão Inteligentes

Grupo 39

Gabriela Santos Ferreira da Cunha - a97393

João António Redondo Martins - a96215

João Pedro Antunes Gonçalves - a95019

Nuno Guilherme Cruz Varela - a96455


a97393


a96215


a95019


a96455

maio, 2023

Conteúdo

1 Introdução **3**

2 Estrutura dos *Workflows* **3**

3 Tarefa A **4**
3.1 *Dataset* 4
3.2 Exploração Prévia dos Dados 5
3.3 Pré-Processamento dos Dados 8
3.3.1 Tipo de imóvel 8
3.3.2 Preço do imóvel 9
3.4 Exploração dos Dados após o Pré-Processamento 10
3.5 Modelos Desenvolvidos 11
3.5.1 *Price* 12
3.5.2 *Type* 12
3.6 Análise dos Resultados Obtidos 14

4 Tarefa B **17**
4.1 *Dataset* 17
4.2 Exploração Prévia dos Dados 17
4.3 Pré-Processamento dos Dados 20
4.4 Exploração dos Dados após o Pré-Processamento 21
4.5 Modelos Desenvolvidos 22
4.6 Análise dos Resultados Obtidos 24

5 Conclusão **28**

1 Introdução

No âmbito da unidade curricular de Aprendizagem e Decisão Inteligentes, foi-nos proposta a conceção e desenvolvimento de um projeto utilizando os modelos de aprendizagem abordados ao longo do semestre.

Este projeto é dividido em duas tarefas, onde o objetivo de cada consiste em:

- Explorar, analisar e preparar um *dataset*, extraindo conhecimento relevante no contexto do problema em questão;
- Conceber e otimizar diversos modelos de *Machine Learning*;
- Realizar uma análise crítica dos resultados.

Deste modo, ambas as tarefas diferem nos *datasets*, sendo a primeira dedicada ao *dataset* escolhido pelo grupo, "Melbourne Housing Market", e a última ao *dataset* fornecido pela equipa docente, relativo à produção de vestuário.

2 Estrutura dos *Workflows*

Com vista a manter uma boa organização, criamos diversos metanodos que vão conter diferentes secções. Os *workflows* desenvolvidos na plataforma KNIME encontram-se, portanto, divididos em três fases essenciais na criação de modelos de aprendizagem:

- Exploração dos dados;
- Pré-processamento dos dados;
- Criação dos modelos;

A primeira etapa, exploração dos dados, permite entender melhor os dados que são usados para treinar e testar os modelos, ajudando a identificar padrões, tendências e anomalias nos dados, que podem vir a influenciar na escolha do modelo e dos seus parâmetros. Para além disso, a exploração dos dados permite analisar a importância de cada variável para o modelo e pode ajudar a identificar problemas como *missing values* ou *outliers*, que podem afetar o desempenho do modelo de forma negativa.

Sabemos que, numa situação real, é comum que os *datasets* precisem de ser limpos e preparados antes de serem usados para treinar e testar os modelos, de modo a garantir que os dados de entrada são relevantes e confiáveis. Nesta segunda etapa, são removidos os ruídos e erros dos dados e tratados os *missing values*, através de estratégias como a simples remoção das linhas ou colunas que contêm os mesmos ou a imputação de valores. Os dados devem também ser transformados para formatos adequados e para uma escala comum entre os mesmos.

Posto isto, é possível avançar para a criação dos modelos seguindo diferentes tipos de aprendizagem, supervisionada ou não, usando diferentes tipos de nodos para obter os melhores resultados.

Desta forma, realizamos diferentes pré-processamentos dos dados que vão ser posteriormente testados com a realização de diversos modelos. Exploramos, ainda, os dados antes e depois deste processamento.

3 Tarefa A

3.1 *Dataset*

O *dataset* selecionado para esta tarefa está relacionado com a venda de imóveis na região metropolitana de Melbourne, sendo útil para quem deseja realizar análises sobre o mercado imobiliário da zona e para quem procura informações sobre os fatores que influenciam o preço dos imóveis. Este *dataset* inclui cerca de 35000 registos e 21 *features* que são as seguintes:

- ***Suburb***: Subúrbio em que a propriedade se localiza;
- ***Address***: Morada;
- ***Rooms***: Número de quartos;
- ***Type***: Tipo de imóvel;
- ***Price***: Preço de venda;
- ***Method***: Método de venda;
- ***SellerG***: Agente imobiliário que agenciou a venda;
- ***Date***: Data de venda;
- ***Distance***: Distância ao centro da cidade em km;
- ***Postcode***: Código postal;
- ***Bedroom2***: Informação adicional sobre o número de quartos (proveniente de outra fonte);
- ***Bathroom***: Número de casas de banho;
- ***Car***: Número de lugares na garagem;
- ***Landsize***: Tamanho do terreno em m^2;
- ***BuildingArea***: Tamanho da área de construção em m^2;
- ***YearBuilt***: Ano de construção;
- ***CouncilArea***: Área do concelho local;
- ***Lattitude***: Coordenada de latitude;
- ***Longtitude***: Coordenada de longitude;
- ***Regionname***: Nome da região;
- ***Propertycount***: Número de propriedades que existem no subúrbio.

Neste *dataset*, o objetivo do grupo passa por prever o tipo e o preço do imóvel, ou seja, as *features* 'Type" e "Price".

3.2 Exploração Prévia dos Dados

Nesta secção, apresentamos a exploração do *dataset* antes do pré-processamento que irá ser feito.

Primeiramente, recorremos ao nodos "Statistics" e "Data Explorer" para analisar todas as *features* do *dataset*. A maior parte das colunas apresentam *missing values* e algumas contêm valores "#N/A", pelo que terão de ser manipulados com o objetivo de os tornar nulos. Verificamos, ainda, que há algumas *features* com demasiados *unique values*.

Para a verificação de existência de *outliers*, fizemos uso do "Box Plot" e analisamos a existência de vários *outliers* para as diferentes colunas.

O *dataset* possui a coluna "Type" que nos indica o tipo do imóvel que foi vendido. Inicialmente, começamos por explorar a quantidade de vendas por tipo de imóvel. Do seguinte gráfico observamos claramente que a maior parte dos imóveis vendidos são do tipo "h", ou seja, são uma casa. Em contrapartida, os *duplex* e as casas na cidade encontram-se em desvantagem.

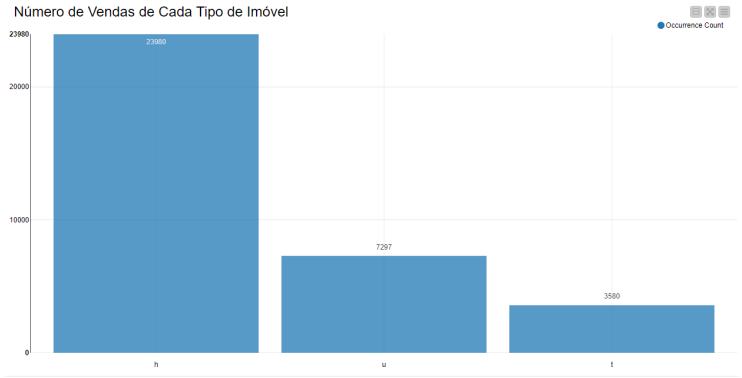


Figura 1: Número de vendas por tipo de imóvel.

Outra das distribuições exploradas foi o preço médio por tipo de imóvel. Concluímos que, em média, os tipos de imóveis que apresentam maior e menor valores monetários são a casa e o *duplex*, respetivamente. Realizamos ainda o gráfico presente na figura 3 que nos permite visualizar a variação dos dados para os três tipos de casa.

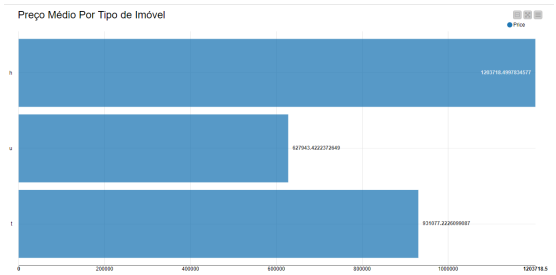


Figura 2: Preço médio por tipo de imóvel.

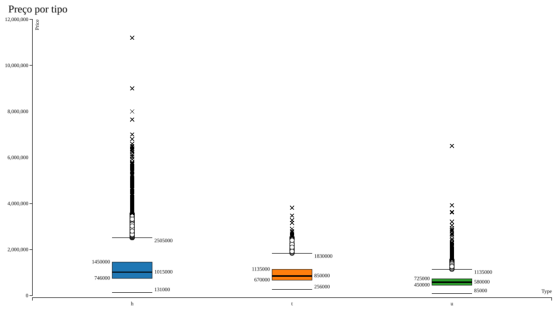


Figura 3: Dispersão do preço por tipo de imóvel.

De maneira a perceber qual é a região mais cara de "Melbourne", fizemos a distribuição do preço por região. Da análise do gráficos, percebemos que a região "Southern Metropolitan" se apresenta com o preço médio mais elevado.

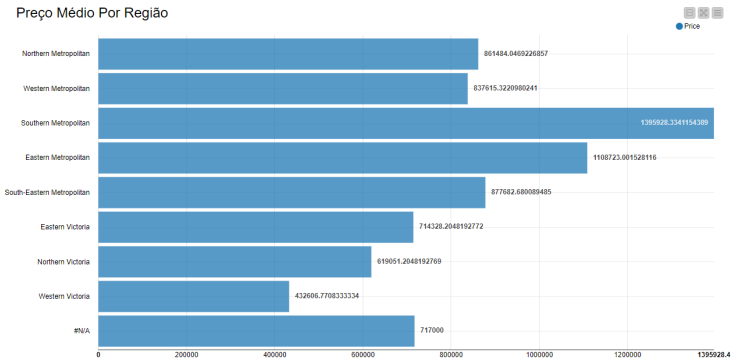


Figura 4: Preço médio por região.

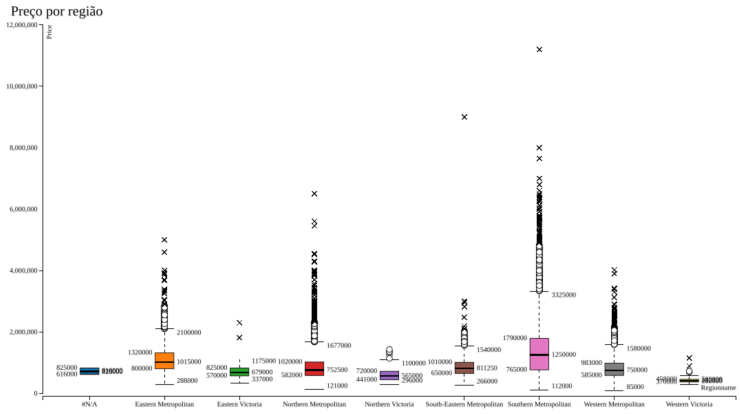


Figura 5: Dispersão do preço por região.

Por último, visualizamos o comportamento do preço das casas conforme o número de quartos presentes.

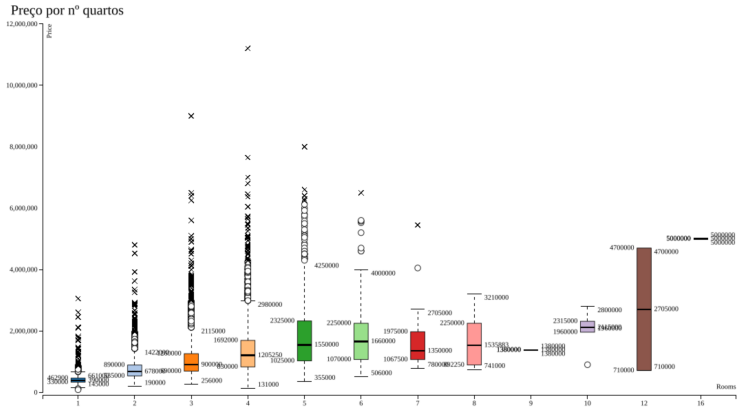
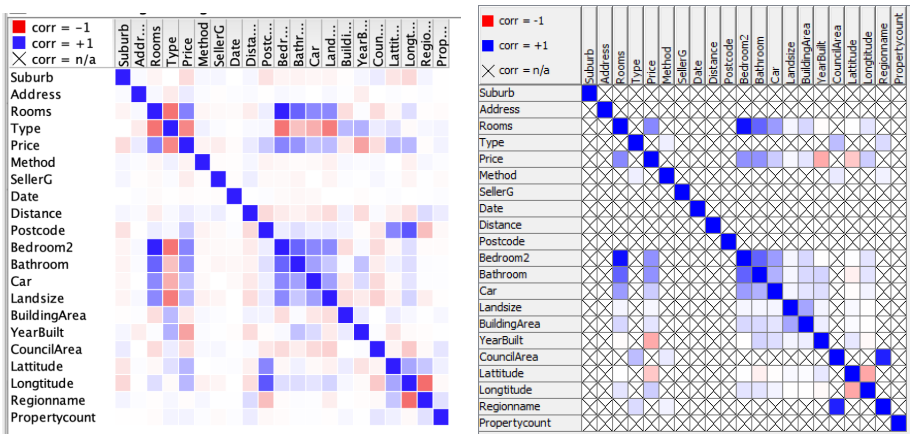


Figura 6: Dispersão do preço por número de quartos.

Através da análise dos nodos de correlação, verificamos que existe uma correlação positiva forte entre as colunas “rooms“ e “bedroom2“, o que era de esperar, visto que a segunda se trata de uma informação adicional à primeira.



(a) Correlação por *rank*.

(b) Correlação linear.

Figura 7: Matrizes de correlação.

3.3 Pré-Processamento dos Dados

Como já foi referido, neste *dataset* são previstas 2 *features* diferentes. Desta forma, realizamos diferentes pré-processamentos tendo em conta as diferentes colunas que pretendemos prever.

3.3.1 Tipo de imóvel

O tratamento dos dados realizado pelo grupo, com vista a prever o tipo de imóvel, encontra-se representado na figura 8.

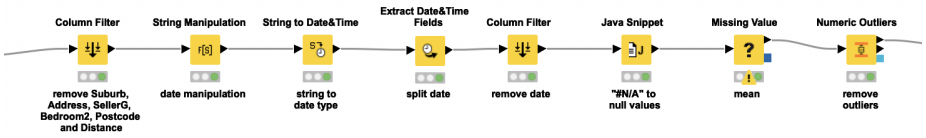


Figura 8: Pré-processamento dos dados realizado para a previsão do tipo.

Inicialmente, começamos por remover as colunas "Suburb", "Address", "SellerG" e "PostCode", dado que contêm muitos *unique values*, não possuindo estas informação significativa para o modelo. Como verificamos na análise da correlação e, visto que a coluna "bedroom2" apresenta *missing values*, optamos por removê-la igualmente, assim como a coluna "Distance" que, praticamente, não apresenta correlação com o tipo de imóvel.

Em seguida, recorrendo aos nodos "String Manipulation", "String to Date&Time" e "Extract Data&Time Fields", descompactamos a data, que era uma *string* com o formato DD/MM/YYYY, em componentes individuais (dia, mês e ano) representadas por inteiros, para percebermos os anos e meses em que mais vendas foram feitas. Por fim, removemos a coluna original por ser redundante, já que temos, de momento, três colunas para o mesmo efeito.

Uma das principais tarefas deste processamento de dados passava por tratar dos *missing values* presentes no *dataset*. Para tal, começamos por substituir os valores que se encontravam com a *string* "#N/A" por efetivamente *missing values*. Assim, é feita esta atribuição e os restantes valores, válidos, são transformados para o seu respetivo tipo, uma vez que alguns números ainda se encontravam representados por *strings*, como é o caso dos valores da coluna "BuildingArea".

Posto isto, recorremos ao nodo "Missing Value" para lidar com o tratamento dos *missing values*. Para valores inteiros, realizamos 4 tipos distintos de tratamento, representados na figura 9: *linear interpolation*, *average interpolation*, *mean* e *median*. De notar que, em cada um destes tratamentos, as linhas que contêm *missing values* das colunas como "CouncilArea" e "RegionName", que os seus valores são *strings*, são removidas.

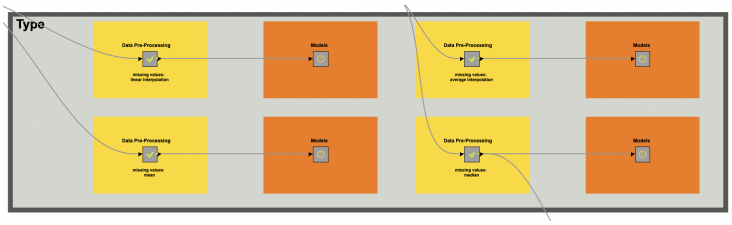


Figura 9: Diferentes tipos de tratamento para os *missing values* realizados.

Finalmente, experimentamos remover todos os *outliers* das colunas, pelo que chegamos à conclusão que todos são *outliers* que ajudam na previsão do modelo, à exceção dos das colunas "Price" e "Landsize". Os valores da coluna "Price" são também normalizados, devido à elevada variância dos valores.

3.3.2 Preço do imóvel

Já o tratamento de dados para a previsão do preço do imóvel, encontra-se representado pela figura 10.

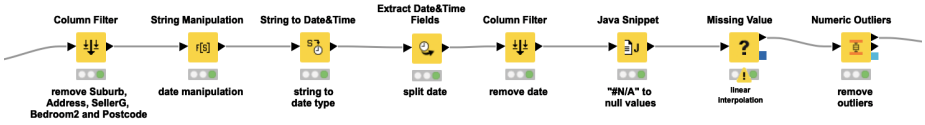


Figura 10: Pré-processamento dos dados realizado para a previsão do preço.

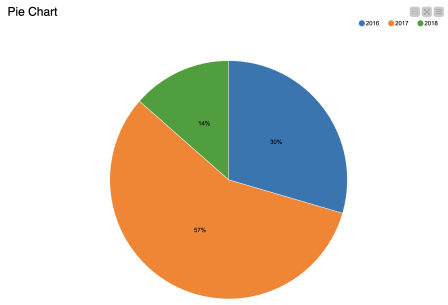
Podemos observar que o pré-processamento apenas difere nas colunas removidas, sendo que a coluna "Distance" é mantida. Pela exploração prévia dos dados, verificamos que a correlação desta coluna com o "Type" é praticamente nula. Por outro lado, a correlação com a coluna "Price" já é maior, traduzindo-se a sua inclusão num impacto positivo no modelo.

Com isto, o nodo "JavaSnippet" trata também de alterar os valores desta coluna para *doubles*. Em relação ao tratamento dos *missing values*, são novamente utilizadas as 4 abordagens feitas para a previsão do tipo e os *outliers* removidos são os mesmos, sendo os *outliers* da coluna "Distance" significativos.

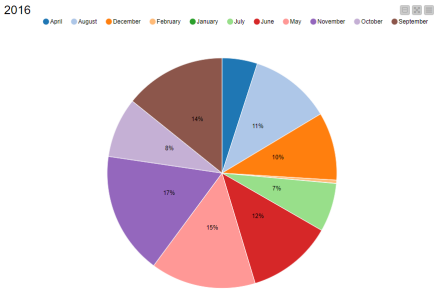
3.4 Exploração dos Dados após o Pré-Processamento

Após o pré-processamento e como agora possuímos colunas com o mês e o ano discriminados, exploramos algumas distribuições em função destas colunas.

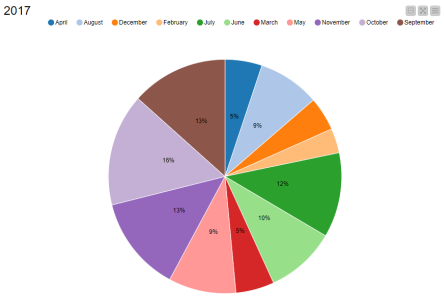
Deste modo, observamos a distribuição das vendas por ano e mês. Inicialmente verificamos que o número de vendas realizadas durante o ano de 2017 é maior do que a soma do número de vendas dos restantes anos. Contudo, no ano de 2016 e 2017 foram registadas vendas em 11 meses, sendo que não foram registadas vendas em janeiro de 2016 e em março no ano seguinte. Pelo contrário, no ano de 2018 só existem vendas até ao mês de março inclusivé, algo que explica a reduzida percentagem de vendas em 2018 no primeiro gráfico.



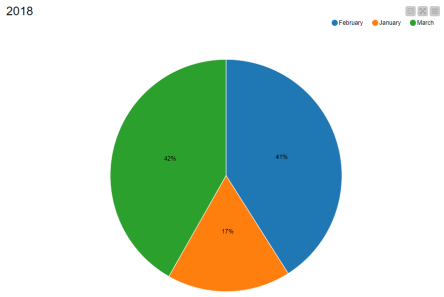
(a) Vendas realizadas por ano.



(b) Vendas realizadas em cada mês de 2016.



(c) Vendas realizadas em cada mês de 2017.



(d) Vendas realizadas em cada mês de 2018.

Figura 11: Distribuição de vendas por ano e mês.

Foram também analisadas novamente as matrizes de correlação, que se encontram presentes no *workflow* mas decidimos não apresentar no relatório, uma vez que têm valores diferentes para os 8 pré-processamentos realizados.

3.5 Modelos Desenvolvidos

Após o processamento dos dados, o *dataset* encontra-se pronto para seguir para a criação de modelos de *machine learning*. Neste *dataset* utilizamos 2 técnicas de aprendizagem supervisionada - regressão e classificação - e uma não supervisionada - segmentação. Como estamos perante diferentes problemas em que são previstas diferentes *features*, realizamos modelos distintos para o tipo e preço do imóvel.

Para cada modelo, realizamos dois tipos de validação: *hold-out validation* e *cross validation*. Para a primeira recorremos a um nodo "Partitioning" em que 75% dos dados são para treino e 25% para teste. Já na validação cruzada usufruímos do nodo "X-Partitioner" com 10 validações. Qualquer um destes nodos utiliza a mesma *seed*, de modo a que possa ser feita uma comparação entre os diferentes nodos utilizados.

Validação *hold-out*

Na validação *hold-out*, o conjunto de dados é dividido em 2 subconjuntos: um conjunto de treino e um conjunto de teste. O modelo é treinado no conjunto de treino e avaliado no conjunto de teste.

Validação cruzada

A validação cruzada é uma técnica mais robusta que envolve a divisão do *dataset* em k subconjuntos, chamados de *folds*. O modelo é treinado k vezes, cada uma com um conjunto diferente de dados de teste e de treino. Em cada treino, um dos k *folds* é usado como conjunto de teste e os restantes são usados como conjunto de treino. O desempenho do modelo é avaliado pela média dos resultados obtidos em cada uma das k execuções. A validação cruzada tem um custo computacional maior, mas é menos dependente da escolha aleatória dos conjuntos, o que a torna mais precisa.

3.5.1 *Price*

A previsão do preço de um imóvel constitui um problema de regressão, pelo que para esta *feature* efetuamos diversos modelos de regressão. Para tal, recorremos a diferentes nodos como o "Linear Regression Learner", "Random Forest Learner", "Tree Ensemble Learner", "Simple Regression Tree Learner", "Gradient Boosted Trees Learner', "Polynomial Regression Learner" e "RProp MLP Learner".

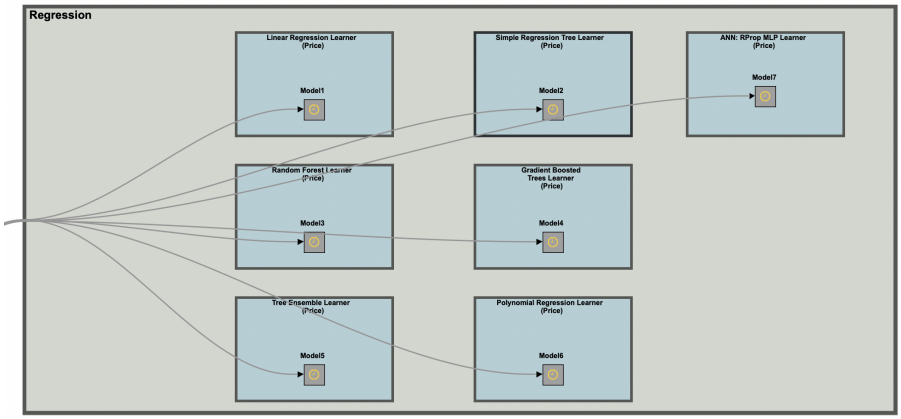


Figura 12: Modelos de regressão para prever o preço.

3.5.2 *Type*

Neste *dataset* recorremos, também, à classificação para resolver o problema de prever o tipo de cada imóvel. Para isso utilizamos o "Decision Tree Learner",

"Gradient Boosted Trees Learner", "Logistic Regression Learner", "Naive Bayes Learner", "Random Forest Learner", "Tree Ensemble Learner" e ao "RProp MLP Learner". De notar que para o nodo "Decision Tree Learner" e "Logistic Regression Learner" foram utilizadas diferentes configurações para serem comparadas posteriormente na análise de resultados.

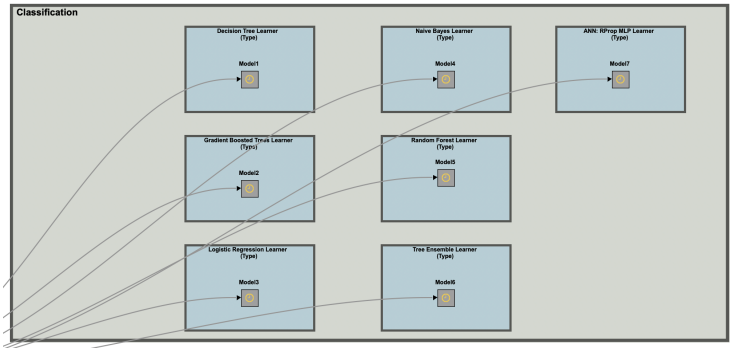


Figura 13: Modelos de classificação para prever o tipo.

O tipo de cada imóvel pode também ser previsto com recurso à técnica de aprendizagem não-supervisionada de segmentação/*clustering*. O *clustering* visa agrupar dados em grupos ou *clusters* com base nas suas características similares. Existem, portanto, vários algoritmos de agrupamento. O primeiro algoritmo utiliza o nodo "k-Means", que usa a média aritmética dos pontos dos dados de um *cluster* para identificar o seu ponto central. Por outro lado, o nodo "k-Medoids", usa um objeto do próprio conjunto de dados do *cluster* como representante desse mesmo *cluster*, chamado "medoid".

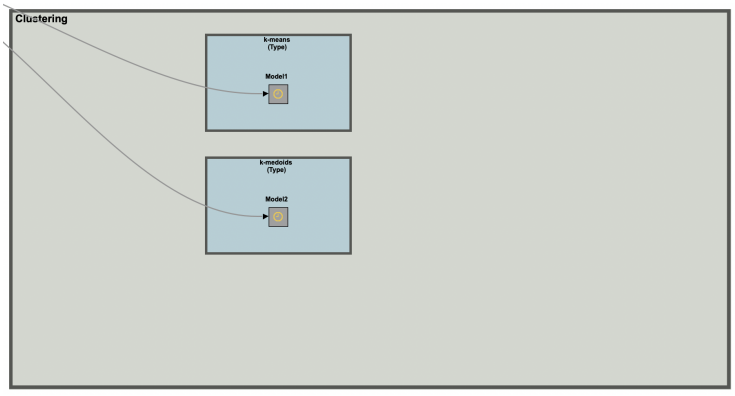


Figura 14: Modelos de segmentação para prever o tipo.

3.6 Análise dos Resultados Obtidos

Com todos os modelos realizados foi possível levantar os resultados para os diferentes pré-processamentos efetuados. Através das métricas de qualidade levantadas como o R^2, MAE (Erro Médio Absoluto), MSE (Erro Médio Quadrado) e RMSE (Raiz Quadrada do Erro Médio), podemos agora avaliar o desempenho dos modelos.

Pré-Processamento	Nodo	Validação hold-out				Validação cruzada			
		R^2	MAE	MSE	RMSE	R^2	MAE	MSE	RMSE
Linear Interpolation	Linear Regression Learner	0.541	0.098	0.018	0.133	0.534	0.100	0.018	0.135
	Random Forest Learner	0.638	0.084	0.014	0.118	0.633	0.084	0.014	0.120
	Tree Ensemble Learner	0.639	0.084	0.014	0.118	0.632	0.084	0.014	0.120
	Simple Regression Tree Learner	0.271	0.115	0.028	0.168	0.276	0.116	0.028	0.168
	Gradient Boosted Trees Learner	0.642	0.081	0.014	0.117	0.633	0.082	0.014	0.120
	Polynomial Regression Learner	0.429	0.113	0.022	0.148	0.264	0.115	0.029	0.169
	RProp MLP Learner	0.423	0.114	0.022	0.149	0.444	0.110	0.022	0.147
Average Interpolation	Linear Regression Learner	0.538	0.100	0.018	0.135	0.535	0.100	0.018	0.135
	Random Forest Learner	0.640	0.084	0.014	0.119	0.633	0.084	0.014	0.120
	Tree Ensemble Learner	0.640	0.084	0.014	0.119	0.635	0.084	0.014	0.120
	Simple Regression Tree Learner	0.254	0.117	0.029	0.171	0.271	0.115	0.029	0.169
	Gradient Boosted Trees Learner	0.642	0.082	0.014	0.118	0.635	0.082	0.014	0.119
	Polynomial Regression Learner	0.434	0.114	0.022	0.149	0.161	0.115	0.033	0.181
	RProp MLP Learner	0.452	0.111	0.021	0.147	0.442	0.111	0.022	0.148
Mean	Linear Regression Learner	0.488	0.106	0.018	0.758	0.500	0.105	0.018	0.132
	Random Forest Learner	0.639	0.085	0.013	0.113	0.648	0.084	0.012	0.111
	Tree Ensemble Learner	0.642	0.085	0.013	0.112	0.647	0.085	0.012	0.111
	Simple Regression Tree Learner	0.289	0.111	0.025	0.158	0.317	0.108	0.024	0.155
	Gradient Boosted Trees Learner	0.639	0.085	0.013	0.113	0.646	0.084	0.012	0.111
	Polynomial Regression Learner	0.383	0.117	0.022	0.147	0.358	0.117	0.023	0.150
	RProp MLP Learner	0.391	0.116	0.021	0.146	0.386	0.117	0.022	0.147
Median	Linear Regression Learner	0.445	0.107	0.019	0.137	0.455	0.106	0.018	0.136
	Random Forest Learner	0.600	0.086	0.014	0.116	0.607	0.085	0.013	0.115
	Tree Ensemble Learner	0.599	0.086	0.014	0.117	0.606	0.086	0.013	0.115
	Simple Regression Tree Learner	0.230	0.109	0.026	0.161	0.236	0.108	0.026	0.161
	Gradient Boosted Trees Learner	0.597	0.086	0.014	0.117	0.603	0.086	0.013	0.116
	Polynomial Regression Learner	0.346	0.117	0.022	0.149	0.330	0.116	0.023	0.151
	RProp MLP Learner	0.352	0.116	0.022	0.148	0.349	0.117	0.022	0.148

Figura 15: Resumo dos resultados nos vários modelos de regressão desenvolvidos.

Relativamente aos modelos de regressão é possível verificar que o melhor modelo obtido para o parâmetro R^2 foi com o nodo "Random Forest Learner" com um valor de 0.648 para o R^2 na validação cruzada, através da substituição dos *missing values* pela média.

Para a validação *hold-out* o melhor resultado de R^2 obtido foi de 0.642 em três modelos. Dois dos modelos utilizam o nodo "Gradient Boosted Trees Learner" com imputação da *linear interpolation* e *average interpolation* nos *missing values*. O modelo restante utiliza o nodo "Tree Ensemble Learner" através da utilização da média nos *missing values*.

De uma forma geral, o pré-processamento que utiliza a mediana no tratamento dos *missing values*, apresenta piores resultados relativamente aos restantes pré-processamentos.

Para os modelos de classificação, foram levantados os resultados para os nodos com diferentes configurações.

Pré-Processamento	Quality Measure	Pruning Method	Validação hold-out		Validação cruzada	
			Accuracy	Cohen's Kappa (k)	Accuracy	Cohen's Kappa (k)
Linear Interpolation	Gini index	MDL	82.315%	0.592	81.779%	0.586
	Gain ratio	MDL	80.918%	0.538	80.752%	0.537
	Gini index	No pruning	78.753%	0.544	77.678%	0.525
	Gain ratio	No pruning	76.602%	0.494	76.568%	0.496
Average Interpolation	Gini index	MDL	81.724%	0.582	82.011%	0.592
	Gain ratio	MDL	80.598%	0.545	80.799%	0.545
	Gini index	No pruning	77.272%	0.518	78.181%	0.536
	Gain ratio	No pruning	77.099%	0.509	77.291%	0.513
Mean	Gini index	MDL	83.055%	0.636	82.832%	0.633
	Gain ratio	MDL	81.593%	0.584	81.380%	0.577
	Gini index	No pruning	80.310%	0.594	80.087%	0.589
	Gain ratio	No pruning	79.425%	0.574	79.484%	0.574
Median	Gini index	MDL	82.299%	0.623	82.550%	0.625
	Gain ratio	MDL	81.491%	0.575	81.341%	0.571
	Gini index	No pruning	80.080%	0.588	80.030%	0.587
	Gain ratio	No pruning	79.515%	0.576	79.840%	0.582

Figura 16: Resultados obtidos alterando os hiperparâmetros do nodo "Decision Tree Learner".

De forma a obter os melhores resultados no modelo com o nodo *Decision Tree Learner*, alterámos os hiperpârametros e em todos os diferentes pré-processamentos o resultado com *accuracy* mais alta está associado à medida de qualidade *Gini Index* e com a utilização de *pruning*. De uma forma geral, a *accuracy* do modelo aumenta na presença de *pruning*.

Pré-Processamento	Solver	Validação hold-out		Validação cruzada	
		Accuracy	Cohen's Kappa (k)	Accuracy	Cohen's Kappa (k)
Linear Interpolation	Stochastic average gradient	81.239%	0.567	80.903%	0.557
	Iteratively reweighted least squares	81.214%	0.566	80.925%	0.557
Average Interpolation	Stochastic average gradient	81.068%	0.564	81.112%	0.564
	Iteratively reweighted least squares	81.031%	0.563	81.121%	0.564
Mean	Stochastic average gradient	80.862%	0.575	81.097%	0.575
	Iteratively reweighted least squares	80.811%	0.568	81.117%	0.576
Median	Stochastic average gradient	80.888%	0.574	81.149%	0.580
	Iteratively reweighted least squares	80.875%	0.574	81.232%	0.583

Figura 17: Resultados obtidos alterando os hiperparâmetros do nodo "Logistic Regression Learner".

Na tabela seguinte, encontra-se o resumo dos resultados obtidos nos diversos modelos de classificação, sendo os valores dos nodos "Decision Tree Leaner" e "Logistic Regression Learner" as melhores configurações obtidas nas tabelas anteriores.

Pré-Processamento	Nodo	Validação hold-out		Validação cruzada	
		Accuracy	Cohen's Kappa (k)	Accuracy	Cohen's Kappa (k)
Linear Interpolation	Decision Tree Learner	82.315%	0.592	81.779%	0.586
	Gradient Boosted Trees Learner	83.861%	0.630	83.507%	0.624
	Logistic Regression Learner	81.214%	0.566	80.925%	0.557
	Naive Bayes Learner	72.891%	0.432	72.991%	0.429
	Random Forest Learner	83.403%	0.615	83.346%	0.617
	Tree Ensemble Learner	83.206%	0.615	83.439%	0.619
	RProp MLP Learner	81.363%	0.550	81.040%	0.549
Average Interpolation	Decision Tree Learner	81.724%	0.582	82.011%	0.592
	Gradient Boosted Trees Learner	83.554%	0.626	83.613%	0.628
	Logistic Regression Learner	81.031%	0.563	81.121%	0.564
	Naive Bayes Learner	74.811%	0.442	73.281%	0.438
	Random Forest Learner	83.319%	0.619	83.752%	0.629
	Tree Ensemble Learner	83.529%	0.622	83.715%	0.628
	RProp MLP Learner	81.699%	0.565	81.118%	0.552
Mean	Decision Tree Learner	83.055%	0.636	82.832%	0.633
	Gradient Boosted Trees Learner	84.505%	0.667	84.699%	0.671
	Logistic Regression Learner	80.811%	0.568	81.117%	0.576
	Naive Bayes Learner	73.114%	0.467	73.638%	0.479
	Random Forest Learner	84.646%	0.667	84.821%	0.670
	Tree Ensemble Learner	84.531%	0.662	84.811%	0.670
	RProp MLP Learner	80.490%	0.537	80.203%	0.536
Median	Decision Tree Learner	82.299%	0.623	82.550%	0.625
	Gradient Boosted Trees Learner	84.351%	0.663	84.478%	0.666
	Logistic Regression Learner	80.875%	0.574	81.232%	0.583
	Naive Bayes Learner	73.486%	0.474	73.237%	0.477
	Random Forest Learner	84.774%	0.672	84.821%	0.672
	Tree Ensemble Learner	84.389%	0.660	84.911%	0.674
	RProp MLP Learner	80.772%	0.551	80.636%	0.549

Figura 18: Resumo dos resultados nos vários modelos de classificação desenvolvidos.

O melhor modelo de classificação realizado foi através do nodo "Tree Ensemble Learner" com 84.911% de *accuracy*, na validação cruzada, com a imputação da mediana. Para a validação *hold-out*, a melhor eficiência obtida foi também através da mediana com o nodo "Random Forest Learner". Desta forma, a mediana revela-se o melhor pré-processamento entre os 4 realizados.

Finalmente, levantamos os resultados para os modelos de segmentação/*clustering* criados, aos quais obtivemos os seguintes resultados.

Pré-Processamento	Validação hold-out		Validação cruzada	
	Accuracy	Cohen's Kappa (k)	Accuracy	Cohen's Kappa (k)
Linear Interpolation	45.313%	0.151	34.365%	0.076
Average Interpolation	44.516%	0.141	34.605%	0.006
Mean	54.079%	0.231	38.398%	0.055
Median	52.989%	0.171	35.615%	-0.007

Figura 19: Resumo dos resultados nos modelos de *clustering*.

Em relação aos resultados obtidos da resolução do problema de forma não supervisionada, recorrendo ao nodo "k-Means" concluímos que os dados deste problema não foram projetados para serem resolvidos de forma não supervisionada, visto que os grupos que estão a ser criados não correspondem aos tipos reais dos imóveis.

4 Tarefa B

4.1 *Dataset*

O *dataset* atribuído pela equipa docente tem como objetivo prever a produção de vestuário. Este *dataset* inclui cerca de 1200 registos e 14 *features* que são as seguintes:

- ***Row ID***: ID do registo;
- ***Date***: Data do registo no formato DD/MM/YYYY;
- ***Department***: Departamento associado;
- ***Team***: Número da equipa associada;
- ***Targeted productivity***: Objetivo de produtividade para cada dia estabelecida para cada equipa pela autoridade;
- ***SMV***: *Standard Minute Value* - tempo definido para uma tarefa;
- ***WIP***: *Work in Progress* - número de itens inacabados para os produtos;
- ***Over Time***: Tempo extra de cada equipa em minutos;
- ***Incentive***: Incentivo financeiro (em BDT) que motiva uma determinada ação;
- ***Idle Time***: Tempo de interrupção da produção em minutos;
- ***Idle Men***: Número de trabalhadores inativos devido à interrupção da produção;
- ***Number of workers***: Número de trabalhadores em cada equipa;
- ***Number of style change***: Número de mudanças no estilo de um determinado produto;
- ***Actual productivity***: Percentagem de produtividade dos trabalhadores;

No âmbito deste *dataset*, o objetivo passa por prever a produtividade real - "Actual productivity".

4.2 Exploração Prévia dos Dados

Nesta secção, apresentamos a exploração dos dados que, como já referimos, nos ajuda a identificar padrões, tendências e anomalias nas entradas do *dataset*.

Inicialmente, verificamos através do nodo "Data Explorer" que apenas a *feature* "wip" contém valores inválidos, "NaN", sobre os quais teremos de efetuar um tratamento.

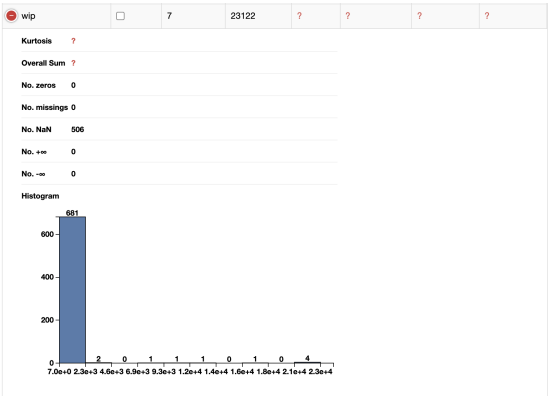


Figura 20: Valores nominais da *feature* "wip".

Através do seguinte tratamento de dados, averiguamos que os *missing values* presentes na coluna "wip" constituem cerca de 42% do *dataset* e encontram-se apenas no departamento "finishing".

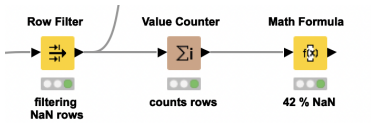


Figura 21: Contagem dos valores "NaN".

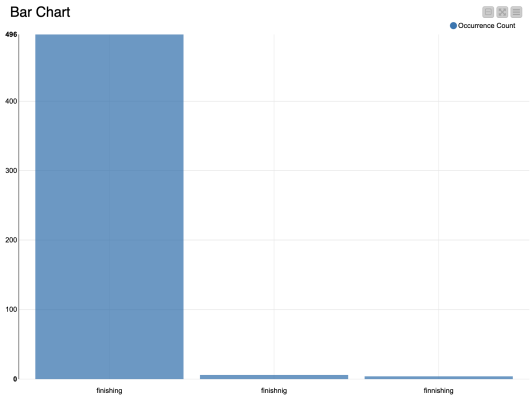


Figura 22: "NaN" no departamento "finishing".

Em seguida, verificamos os tipos de departamentos existentes. Observamos dois tipos - "sweing" e "finishing" -, sendo os restantes derivações com erros ortográficos dos anteriores.

Column	Exclude Column	No. missings	Unique values	All nominal values	Frequency Bar Chart
department	☐	0	5	sweing, finishing, finishnig, finnishing, swenig	

Figura 23: Departamentos existentes.

Posteriormente, são identificadas as equipas de trabalho, representadas por um inteiro, sendo as equipas 2 e 8 as que ocorrem com mais frequência. Podemos verificar que, em termos de equipas, o *dataset* se encontra bem distribuído, isto é, cada equipa contém um número similar de ocorrências, evitando o enviesamento dos dados.

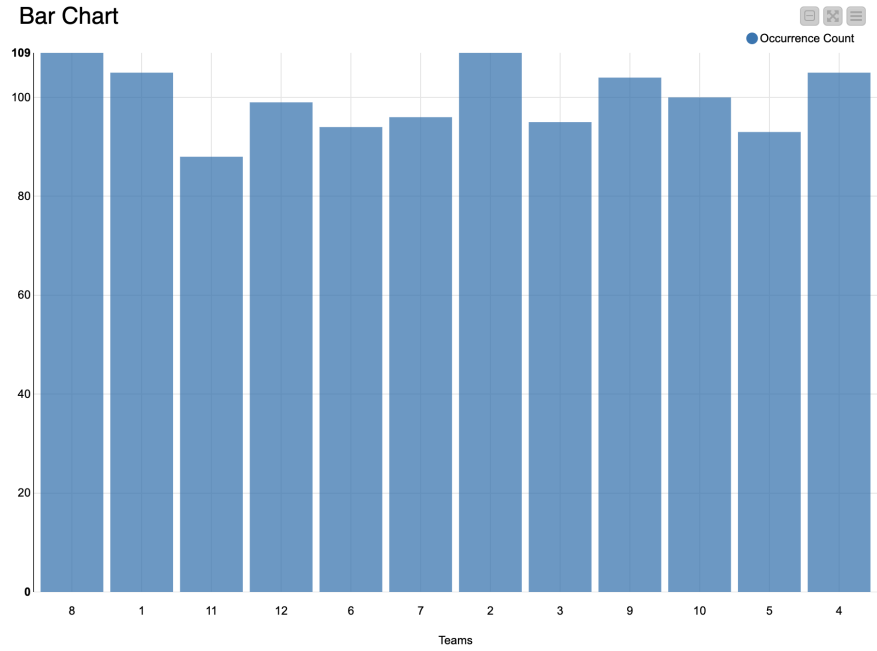
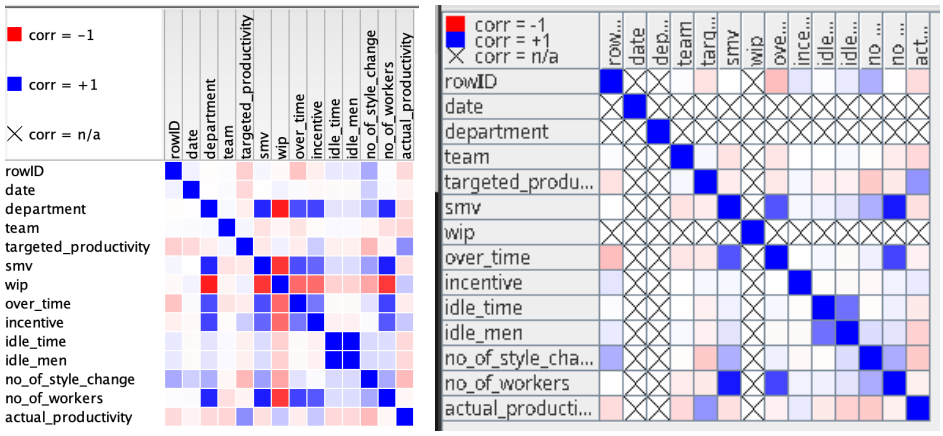


Figura 24: Equipas existentes.

Por fim, são analisadas as matrizes de correlação entre as variáveis com recurso aos nodos "Rank Correlation" e "Linear Correlation".



(a) Correlação por *rank*.

(b) Correlação linear.

Figura 25: Matrizes de correlação.

4.3 Pré-Processamento dos Dados

Relativamente ao pré-processamento dos dados, utilizamos metanodos com diferentes configurações que vão originar diferentes *outputs* nos dados. Seguimos esta abordagem com o objetivo de podermos comparar e avaliar o melhor pré-processamento realizado.

Esta fase revelou-se ser a mais complexa nesta tarefa, devido à coluna "wip". Como averiguado na exploração, esta coluna apresenta 42% de ocorrências "NaN", as quais interpretamos como *missing values*, o que levou a uma maior reflexão por parte do grupo. Deste modo, utilizamos 5 tipos diferentes de tratamento destes *missing values*: imputação do valor 0, imputação da média, *linear interpolation*, *average interpolation*, imputação da mediana e remoção da coluna "wip". De notar ainda que estes valores apenas se encontram em um dos dois departamentos, pelo que se optássemos por remover estas linhas, iriamos ficar apenas com dados de um departamento. Não nos pareceu a solução mais viável, uma vez que perderíamos toda a informação acerca de um departamento.

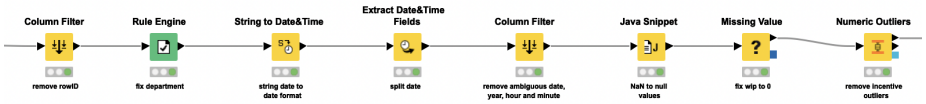


Figura 26: Pré-processamento da tarefa B.

Em relação ao tratamento de dados comum aos vários processamentos desenvolvidos, primeiramente, tendo em conta que a *feature* "rowID" é um identifi-

cador, ou seja, cada linha possui um valor diferente, é evidente que esta não contribui para a aprendizagem do nosso modelo.

Em seguida, vimos a necessidade de manipular os valores de algumas linhas da coluna relacionada com o departamento, visto que apresentavam alguns valores mal-escritos, pelo que recorremos ao nodo "Rule Engine".

Mais adiante, transformamos a *string* data no formato *date*. A este novo formato extraímos campos da data como o ano, mês, dia do mês e da semana, horas e minutos. Como o ano, horas e minutos são iguais em todas as linhas, optamos por remover essas colunas, visto que não ajudariam o modelo a prever.

Por conseguinte, identificamos que as *features* "wip", "over time", "incentive", "idle time", "idle men" e "no of workers" apresentam *outliers*. De acordo com os resultados obtidos, retirámos apenas os *outliers* da *feature* "incentive", visto que, os outros apresentam significado para o modelo.

4.4 Exploração dos Dados após o Pré-Processamento

Após o tratamento dos dados, pudemos explorar a relação entre o número de trabalhadores por equipa e o tempo necessário para cada tarefa através do *scatter plot* da figura 27. Deste gráfico concluímos que, para o departamento de "finishing", o tempo alocado para cada tarefa mantém-se com o número de trabalhadores.

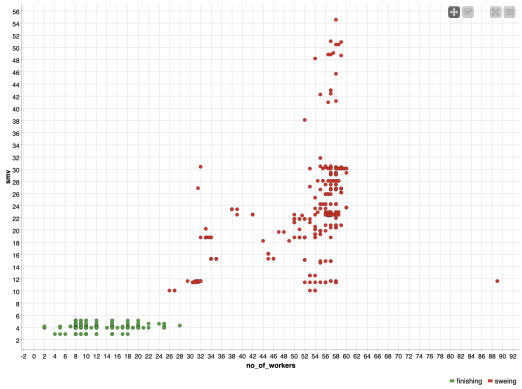


Figura 27: Número de trabalhadores.

Por fim, a partir da extração dos parâmetros da data, verificamos que a empresa trabalha todos os dias da semana com exceção da sexta-feira.

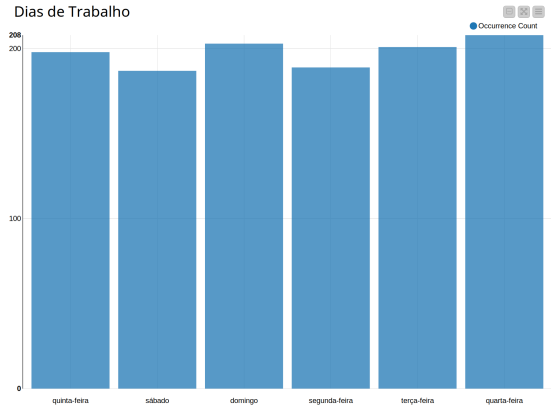


Figura 28: Dias da semana de trabalho.

4.5 Modelos Desenvolvidos

A *feature* prevista neste *dataset* foi a "actual_productivity" (produtividade atual) novamente com recurso a 3 diferentes técnicas de aprendizagem. À semelhança dos modelos desenvolvidos para a tarefa A, utilizamos dois tipos de validação.

De maneira a resolver o problema através de técnicas de regressão, apenas utilizamos nodos de regressão para criar modelos e, em seguida, comparar os reultados obtidos.

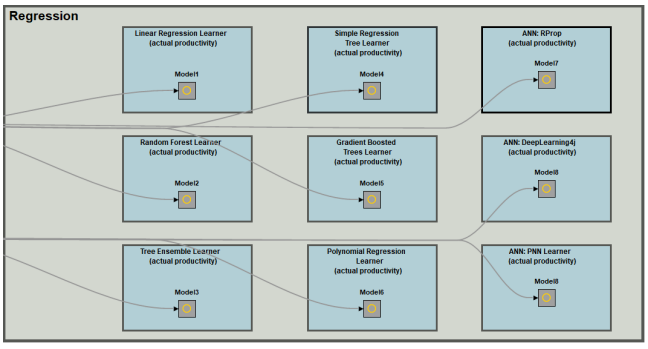


Figura 29: Modelos de regressão realizados.

De forma a resolver o problema através de técnicas de classificação, tivemos de discretizar a *feature* a ser prevista. Para isso, utilizamos 5 *bins* com igual largura, sendo, em seguida, atribuídos nomes às mesmas. Por conseguinte, a produtividade pode ser: *Very Bad, Bad, Normal, Good* e *Very Good*. Por fim,

retiramos a coluna com os valores contínuos para não ser usada nos modelos de classificação.

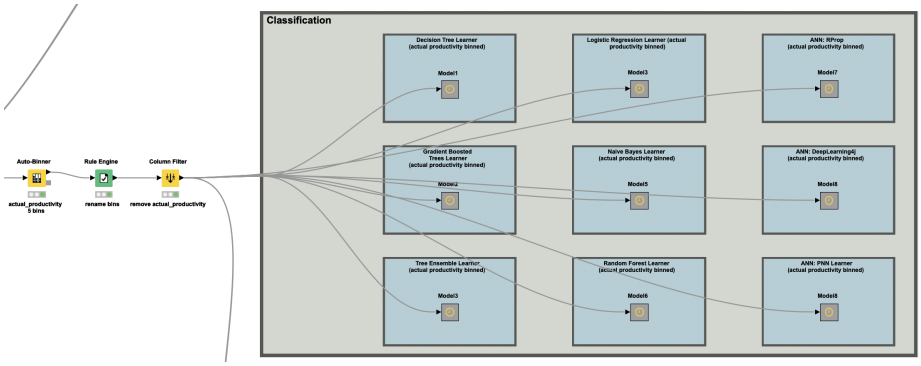


Figura 30: Modelos de classificação realizados.

Por fim, mesmo o problema sendo de regressão, recorremos a técnicas de aprendizagem não supervisionada como a segmentação para resolver o problema.

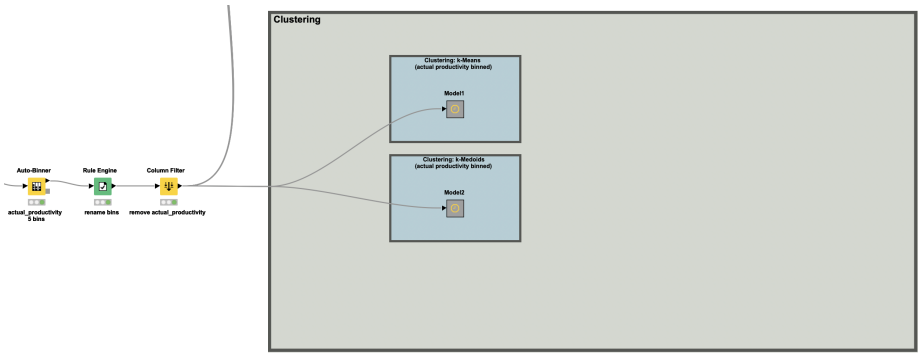


Figura 31: Modelos de *clustering* realizados.

4.6 Análise dos Resultados Obtidos

Pré-Processamento	Nodo	Validação hold-out				Validação cruzada			
		R^2	MAE	MSE	RMSE	R^2	MAE	MSE	RMSE
Fix Value: 0	Linear Regression Learner	0.362	0.095	0.019	0.138	0.372	0.098	0.019	0.138
	Random Forest Learner	0.538	0.074	0.014	0.117	0.523	0.077	0.014	0.120
	Tree Ensemble Learner	0.524	0.075	0.014	0.119	0.529	0.076	0.014	0.119
	Simple Regression Tree Learner	0.117	0.088	0.026	0.162	0.077	0.094	0.028	0.167
	Gradient Boosted Trees Learner	0.506	0.072	0.015	0.121	0.526	0.072	0.014	0.120
	Polynomial Regression Learner	0.348	0.100	0.019	0.139	0.342	0.102	0.020	0.141
	RProp MLP Learner	0.348	0.098	0.025	0.157	0.370	0.101	0.024	0.156
	DL4J Feedforward Learner	-2.586	0.286	0.100	0.317	-2.216	0.280	0.097	0.312
	PNN Learner	-0.188	0.121	0.033	0.182	-0.309	0.135	0.040	0.199
Mean	Linear Regression Learner	0.362	0.095	0.019	0.138	0.372	0.098	0.019	0.138
	Random Forest Learner	0.538	0.075	0.014	0.117	0.526	0.077	0.014	0.120
	Tree Ensemble Learner	0.523	0.076	0.014	0.119	0.527	0.076	0.014	0.120
	Simple Regression Tree Learner	0.054	0.092	0.028	0.168	0.102	0.092	0.027	0.165
	Gradient Boosted Trees Learner	0.545	0.069	0.014	0.117	0.516	0.073	0.015	0.121
	Polynomial Regression Learner	0.346	0.100	0.020	0.140	0.340	0.102	0.020	0.141
	RProp MLP Learner	0.412	0.096	0.022	0.149	0.354	0.103	0.025	0.158
	DL4J Feedforward Learner	-2.586	0.286	0.100	0.317	NaN	NaN	NaN	NaN
	PNN Learner	-0.235	0.127	0.035	0.186	-0.235	0.132	0.037	0.193
Linear Interpolation	Linear Regression Learner	0.362	0.095	0.019	0.138	0.371	0.098	0.019	0.138
	Random Forest Learner	0.535	0.073	0.014	0.118	0.530	0.075	0.014	0.119
	Tree Ensemble Learner	0.532	0.073	0.014	0.118	0.536	0.073	0.014	0.119
	Simple Regression Tree Learner	0.263	0.084	0.022	0.148	0.075	0.095	0.028	0.167
	Gradient Boosted Trees Learner	0.493	0.072	0.015	0.123	0.528	0.073	0.014	0.119
	Polynomial Regression Learner	0.348	0.100	0.019	0.140	0.340	0.101	0.020	0.141
	RProp MLP Learner	0.222	0.111	0.030	0.172	0.357	0.101	0.025	0.157
	DL4J Feedforward Learner	-2.586	0.286	0.100	0.317	-2.240	0.282	0.098	0.313
	PNN Learner	-0.421	0.142	0.040	0.199	-0.533	0.151	0.046	0.215
Average Interpolation	Linear Regression Learner	0.362	0.095	0.019	0.138	0.371	0.098	0.019	0.138
	Random Forest Learner	0.535	0.073	0.014	0.118	0.530	0.075	0.014	0.119
	Tree Ensemble Learner	0.532	0.073	0.014	0.118	0.536	0.073	0.014	0.119
	Simple Regression Tree Learner	0.263	0.084	0.022	0.148	0.075	0.095	0.028	0.167
	Gradient Boosted Trees Learner	0.493	0.072	0.015	0.123	0.528	0.073	0.014	0.119
	Polynomial Regression Learner	0.348	0.100	0.019	0.140	0.340	0.101	0.020	0.141
	RProp MLP Learner	0.407	0.100	0.023	0.150	0.324	0.105	0.026	0.161
	DL4J Feedforward Learner	-2.586	0.286	0.100	0.317	NaN	NaN	NaN	NaN
	PNN Learner	-0.451	0.131	0.041	0.202	-0.294	0.132	0.039	0.198
Median	Linear Regression Learner	0.362	0.095	0.019	0.138	0.371	0.098	0.019	0.138
	Random Forest Learner	0.535	0.073	0.014	0.118	0.530	0.075	0.014	0.119
	Tree Ensemble Learner	0.532	0.073	0.014	0.118	0.536	0.073	0.014	0.119
	Simple Regression Tree Learner	0.263	0.084	0.022	0.148	0.075	0.095	0.028	0.167
	Gradient Boosted Trees Learner	0.493	0.072	0.015	0.123	0.528	0.073	0.014	0.119
	Polynomial Regression Learner	0.348	0.100	0.019	0.140	0.340	0.101	0.020	0.141
	RProp MLP Learner	0.352	0.104	0.025	0.157	0.356	0.104	0.025	0.157
	DL4J Feedforward Learner	-2.586	0.286	0.100	0.317	NaN	NaN	NaN	NaN
	PNN Learner	-0.200	0.125	0.034	0.183	-0.270	0.134	0.038	0.196
Column Filter	Linear Regression Learner	0.362	0.095	0.019	0.138	0.371	0.098	0.019	0.138
	Random Forest Learner	0.535	0.073	0.014	0.118	0.530	0.075	0.014	0.119
	Tree Ensemble Learner	0.532	0.073	0.014	0.118	0.536	0.073	0.014	0.119
	Simple Regression Tree Learner	0.263	0.084	0.022	0.148	0.075	0.095	0.028	0.167
	Gradient Boosted Trees Learner	0.493	0.072	0.015	0.123	0.528	0.073	0.014	0.119
	Polynomial Regression Learner	0.348	0.100	0.019	0.140	0.340	0.101	0.020	0.141
	RProp MLP Learner	0.281	0.107	0.027	0.165	0.368	0.104	0.024	0.156
	DL4J Feedforward Learner	-2.586	0.286	0.100	0.317	NaN	NaN	NaN	NaN
	PNN Learner	-0.392	0.128	0.039	0.197	-0.199	0.122	0.036	0.190

Figura 32: Resumo dos resultados nos vários modelos de regressão desenvolvidos.

Para a análise dos resultados é importante referir os hiperparâmetros dos nodos utilizados. No "RProp MLP Learner" utilizamos 950 iterações com 4 ca-

madas e 8 neurónios por camada. Já no "Gradient Boosted Tree Learner" apenas alteramos o limite do número de profundidade da arvóre para 5. Nos restantes nodos apresentados na tabela acima usamos as configurações por defeito.

Relativamente aos modelos de regressão é possível identificar que o melhor modelo foi obtido com o nodo "Gradient Boosted Trees Learner" com a substituição dos *missing values* da coluna "wip" pela média, onde obtivemos o valor mais alto do R^2, 0.545, e os valores mais baixos de erros, com o MAE igual a 0.069, MSE igual a 0.014 e RMSE igual a 0.117, na validação *hold-out*. Para a validação cruzada, o melhor modelo obtido foi com o nodo "Tree Ensemble Learner" com a substituição dos *missing values* da coluna "wip" pela mediana, *average interpolation*, *linear interpolation* e remoção da coluna.

Por outro lado, é possível verificar que o nodo "Random Forest Learner" apresenta melhores resultados em todos os diferentes tipos de pré-processamento para a validação *hold-out*, com exceção no pré-processamento em que é utilizada a imputação da média nos *missing values*. Neste caso, é o nodo "Gradient Boosted Trees Learner" que apresenta, de um modo geral, o melhor resultado.

Quanto aos nodos de redes neuronais artificiais, como é o caso da utilização do *DeepLearning4J* e do *PNN Learner*, estes não apresentam bons resultados não sendo neste problema uma boa técnica de previsão.

Relativamente aos modelos de classificação utilizados para prever as *bins* criadas para a produtividade, utilizamos diferentes configurações para os nodos "Decision Tree Learner" e "Logistic Regression Learner" com o objetivo de perceber o impacto das alterações dos hiperparâmetros na precisão dos modelos.

Pré-Processamento	Quality Measure	Pruning Method	Validação hold-out		Validação cruzada	
			Accuracy	Cohen's Kappa (k)	Accuracy	Cohen's Kappa (k)
Fix Value: 0	Gini index	MDL	68.350%	0.512	67.960%	0.521
	Gain ratio	MDL	67.003%	0.489	66.189%	0.488
	Gini index	No pruning	66.330%	0.519	65.852%	0.508
	Gain ratio	No pruning	64.646%	0.498	66.610%	0.522
Mean	Gini index	MDL	68.350%	0.512	67.960%	0.521
	Gain ratio	MDL	67.003%	0.489	66.020%	0.487
	Gini index	No pruning	65.993%	0.514	65.852%	0.508
	Gain ratio	No pruning	64.646%	0.498	66.695%	0.522
Linear Interpolation	Gini index	MDL	70.370%	0.550	67.622%	0.513
	Gain ratio	MDL	67.003%	0.489	65.599%	0.475
	Gini index	No pruning	67.677%	0.539	65.177%	0.500
	Gain ratio	No pruning	67.003%	0.531	67.201%	0.530
Average Interpolation	Gini index	MDL	69.697%	0.538	67.454%	0.510
	Gain ratio	MDL	67.003%	0.489	66.526%	0.491
	Gini index	No pruning	69.697%	0.566	67.538%	0.534
	Gain ratio	No pruning	66.697%	0.531	68.550%	0.547
Median	Gini index	MDL	68.350%	0.512	67.960%	0.521
	Gain ratio	MDL	67.003%	0.489	66.020%	0.487
	Gini index	No pruning	65.993%	0.514	65.852%	0.508
	Gain ratio	No pruning	64.646%	0.498	66.695%	0.522
Column Filter	Gini index	MDL	68.350%	0.512	67.538%	0.515
	Gain ratio	MDL	67.003%	0.489	66.779%	0.496
	Gini index	No pruning	66.667%	0.525	66.442%	0.517
	Gain ratio	No pruning	64.983%	0.502	66.863%	0.525

Figura 33: Resultados obtidos alterando os hiperparâmetros do nodo "Decision Tree Learner".

Na validação *hold-out* verificamos que o melhor resultado é obtido com *gini index* no parâmetro "Quality Measure" e *pruning*, utilizando *linear interpolation* no tratamento dos *missing values* da coluna "wip". Na validação cruzada, observamos que o melhor resultado, de *accuracy* igual a 67.96% é obtido com *gini index* e *pruning*, tanto na substituição do *missing value* por 0 como pela média ou pela mediana.

De um modo geral, observamos que obtêm-se melhores resultados utilizando *pruning*. O *pruning* consiste numa técnica de otimização usada nas árvores de decisão, de modo a evitar *overfitting* - quando o modelo se ajusta aos dados de treino mas não generaliza bem para novos dados. Desta forma, são removidas partes da árvore que não são úteis para a previsão, obtendo-se uma melhoria na generalização dos dados e uma redução do tempo necessário para treinar o modelo.

Para a construção das árvores de decisão, podem ser utilizados dois critérios diferentes, o *gini index* e o *gain ratio*, de onde podemos concluir que se obtêm melhores resultados com a utilização do primeiro.

Pré-Processamento	Solver	Validação hold-out		Validação cruzada	
		Accuracy	Cohen's Kappa (k)	Accuracy	Cohen's Kappa (k)
Fix Value: 0	Stochastic average gradient	57.912%	0.356	57.251%	0.342
	Iteratively reweighted least squares	58.923%	0.382	58.179%	0.371
Mean	Stochastic average gradient	57.912%	0.356	57.336%	0.344
	Iteratively reweighted least squares	58.923%	0.382	58.179%	0.371
Linear Interpolation	Stochastic average gradient	58.249%	0.361	57.504%	0.347
	Iteratively reweighted least squares	59.596%	0.390	58.516%	0.377
Average Interpolation	Stochastic average gradient	58.249%	0.361	57.420%	0.346
	Iteratively reweighted least squares	59.596%	0.390	58.600%	0.378
Median	Stochastic average gradient	57.912%	0.356	57.336%	0.344
	Iteratively reweighted least squares	58.923%	0.382	58.179%	0.371
Column Filter	Stochastic average gradient	57.912%	0.356	57.083%	0.340
	Iteratively reweighted least squares	58.259%	0.387	58.263%	0.372

Figura 34: Resultados obtidos alterando os hiperparâmetros do nodo "Logistic Regression Learner".

No nodo "Logistic Regression Learner", os algoritmos para construir modelos de regressão logística variam entre "Stochastic Average Gradient (SAG)" e "Iteratively Reweighted Least Squares (IRLS)". Através da figura anterior, concluímos que obtêm-se melhores resultados com o uso do segundo algoritmo. Este é apropriado para *datasets* com distribuições desiguais ou variáveis fortemente correlacionadas, enquanto que o primeiro se trata de um algoritmo estocástico geralmente indicado para *datasets* mais extensos.

Pré-Processamento	Nodo	Validação hold-out		Validação cruzada	
		Accuracy	Cohen's Kappa (k)	Accuracy	Cohen's Kappa (k)
Fix Value: 0	Decision Tree Learner	68.350%	0.512	67.960%	0.521
	Gradient Boosted Trees Learne	73.401%	0.625	70.742%	0.575
	Tree Ensemble Learner	72.727%	0.590	72.007%	0.577
	Logistic Regression Learner	58.923%	0.382	58.179%	0.371
	Naive Bayes Learner	10.774%	0.011	13.322%	0.022
	Random Forest Learner	71.044%	0.567	71.164%	0.565
	RProp MLP Learner	60.606%	0.432	61.804%	0.432
Mean	Decision Tree Learner	68.350%	0.512	67.960%	0.521
	Gradient Boosted Trees Learne	71.380%	0.594	74.790%	0.630
	Tree Ensemble Learner	72.727%	0.590	70.995%	0.561
	Logistic Regression Learner	58.923%	0.382	58.179%	0.371
	Naive Bayes Learner	10.774%	0.011	9.244%	-0.002
	Random Forest Learner	72.391%	0.588	71.429%	0.561
	RProp MLP Learner	60.943%	0.409	62.395%	0.441
Linear Interpolation	Decision Tree Learner	70.370%	0.550	67.622%	0.513
	Gradient Boosted Trees Learne	75.084%	0.642	77.311%	0.661
	Tree Ensemble Learner	74.074%	0.616	72.597%	0.587
	Logistic Regression Learner	59.596%	0.390	58.516%	0.377
	Naive Bayes Learner	10.774%	0.011	9.244%	-0.002
	Random Forest Learner	72.727%	0.598	75.630%	0.633
	RProp MLP Learner	62.290%	0.429	62.816%	0.443
Average Interpolation	Decision Tree Learner	66.697%	0.531	68.550%	0.547
	Gradient Boosted Trees Learne	75.084%	0.642	77.311%	0.661
	Tree Ensemble Learner	74.747%	0.626	72.934%	0.593
	Logistic Regression Learner	59.596%	0.390	58.600%	0.378
	Naive Bayes Learner	10.774%	0.011	9.244%	-0.002
	Random Forest Learner	72.727%	0.598	75.630%	0.633
	RProp MLP Learner	60.943%	0.416	62.901%	0.448
Median	Decision Tree Learner	68.350%	0.512	67.960%	0.521
	Gradient Boosted Trees Learne	75.084%	0.642	77.311%	0.661
	Tree Ensemble Learner	72.054%	0.577	71.417%	0.569
	Logistic Regression Learner	58.923%	0.382	58.179%	0.371
	Naive Bayes Learner	10.774%	0.011	9.244%	-0.002
	Random Forest Learner	72.727%	0.598	75.630%	0.633
	RProp MLP Learner	59.596%	0.395	60.877%	0.417
Column Filter	Decision Tree Learner	68.350%	0.512	67.538%	0.515
	Gradient Boosted Trees Learne	75.084%	0.642	77.311%	0.661
	Tree Ensemble Learner	71.717%	0.584	71.164%	0.567
	Logistic Regression Learner	58.259%	0.387	58.263%	0.372
	Naive Bayes Learner	10.774%	0.011	9.244%	-0.002
	Random Forest Learner	72.727%	0.598	75.630%	0.633
	RProp MLP Learner	60.269%	0.402	64.924%	0.475

Figura 35: Resumo dos resultados nos vários modelos de classificação desenvolvidos.

Relativamente aos modelos de classificação é possível identificar que os melhores modelos foram obtidos com o nodo "Gradient Boosted Trees Learner" tanto no caso da validação *hold-out* como no caso da validação *cruzada*. No caso da primeira, o melhor modelo possui *accuracy* de 75.084% e, na segunda o melhor modelo possui *accuracy* de 77.311%. É possível identificar também uma semelhança nos resultados para os dois tipos de validação, principalmente nos seguintes pré-processamentos: *median, linear interpolation, average interpolation* e *column filter*, visto que os resultados em todos os nodos com exceção do "RProp MLP Learner" são iguais.

Pré-Processamento	Validação hold-out		Validação cruzada	
	Accuracy	Cohen's Kappa (k)	Accuracy	Cohen's Kappa (k)
Fix Value: 0	29.966%	0.046	23.862%	0.037
Mean	27.946%	0.035	23.356%	0.032
Linear Interpolation	27.946%	0.036	24.030%	0.027
Average Interpolation	27.946%	0.036	23.946%	0.027
Median	27.946%	0.036	23.609%	0.027
Column Filter	28.283%	0.039	23.187%	0.029

Figura 36: Resumo dos resultados nos modelos de *clustering*.

Em relação aos resultados obtidos da resolução do problema de forma não supervisionada com o nodo "k-Means". Os resultados são insatisfatórios sendo que a divisão por *clusters* não corresponde às classes de produtividade pretendidas.

5 Conclusão

Este trabalho prático teve como objetivo explorar, analisar e preparar 2 *datasets* para criar modelos de aprendizagem automática. A plataforma utilizada foi o KNIME, que providencia uma interface de alto nível para o desenvolvimento de trabalho no âmbito de *machine learning*, permitindo consolidar os conteúdos teóricos abordados, através da sua aplicação prática. A par destes conceitos teóricos, o grupo considera também que as aulas práticas foram bastante úteis para a realização do projeto, na medida em que este se aproxima bastante do que é proposto nas mesmas, mas em maior escala.

O processo de desenvolvimento dos modelos apresentados exigiu bastante reflexão por parte do grupo e, por vezes, a reconstrução dos modelos ou do tratamento dos dados. De uma forma geral, consideramos que a parte mais complexa e que proporcionou maior dedicação foram os diferentes pré-processamentos realizados. Deste modo, o grupo considera que os objetivos propostos foram alcançados com sucesso, uma vez que foram utilizadas diversas estratégias para a criação dos modelos.