



UNIVERSIDADE DO MINHO

LICENCIATURA EM ENGENHARIA INFORMÁTICA

Aprendizagem e Decisão Inteligentes  
Trabalho Prático: Conceção de modelos de aprendizagem  
Grupo 22

José Pereira (A89596)      Rui Monteiro (A93179)  
Rodrigo Rodrigues (A93201)      Daniel Azevedo (A93324)

Ano Letivo 2021/2022



# 1 Introdução e Objetivos

Este projeto surge no âmbito da unidade curricular de Aprendizagem e Decisão Inteligentes, com o intuito de analisar e extrair conhecimento, aplicando modelos de aprendizagem abordados ao longo do semestre de forma a obter e analisar resultados úteis no contexto dos problemas subjacentes aos *datasets* trabalhados.

O projeto encontra-se dividido em duas fases. Na fase inicial, pretende-se explorar, analisar e preparar dois *datasets* - *Weather Conditions in World War Two*, que possui dados sobre as condições climáticas durante a 2<sup>a</sup> Guerra Mundial (escolhido pelo grupo) e classificação da qualidade de vinhos (atribuído ao grupo).

Adicionalmente, é fulcral decidir quais os domínios a tratar, quais os objetivos a alcançar e como os atingir e, finalmente, qual a metodologia a seguir. A segunda fase incide sobre a conceção de modelos de aprendizagem para ambos os problemas, obtenção e interpretação/análise crítica de resultados.

Com vista a aplicar os conhecimentos adquiridos nas aulas de Aprendizagem e Decisão Inteligentes, será utilizado o software de análise de dados **KNIME**.

# Conteúdo

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Introdução e Objetivos</b>                                    | <b>2</b>  |
| <b>2</b> | <b><i>Dataset</i>: “Wine Quality Classification”</b>             | <b>4</b>  |
| 2.1      | Relevância e Utilidade do <i>Dataset</i> no mundo real . . . . . | 4         |
| 2.2      | Análise Exploratória . . . . .                                   | 4         |
| 2.3      | Particionar os dados . . . . .                                   | 5         |
| 2.4      | Modelo KNIME . . . . .   | 6         |
| 2.5      | Conceção dos Modelos de Aprendizagem . . . . .                   | 6         |
| 2.5.1    | Decision Trees . . . . .   | 6         |
| 2.5.2    | Random Forests . . . . .   | 7         |
| 2.5.3    | XGBoost Linear Ensemble . . . . .                                | 8         |
| 2.5.4    | Redes Neurais Artificiais . . . . .                              | 9         |
| 2.5.5    | Obtenção e análise de resultados . . . . .                       | 11        |
| <b>3</b> | <b><i>Dataset</i>: “Weather Conditions in World War Two”</b>     | <b>12</b> |
| 3.1      | Relevância e Utilidade do Dataset no mundo real . . . . .        | 13        |
| 3.2      | Análise Exploratória . . . . .                                   | 13        |
| 3.3      | Particionar os dados . . . . .                                   | 14        |
| 3.4      | Conceção de Modelo de Aprendizagem . . . . .                     | 14        |
| 3.4.1    | Regressão Linear . . . . .                                       | 14        |
| 3.4.2    | Decision Trees . . . . .   | 16        |
| 3.4.3    | Random Forest . . . . .  | 17        |
| 3.4.4    | Redes Neurais Artificiais . . . . .                              | 18        |
| 3.4.5    | Obtenção e análise de resultados . . . . .                       | 19        |
| <b>4</b> | <b>Conclusão</b>   | <b>21</b> |

## 2 *Dataset*: “Wine Quality Classification”

O dataset atribuído *dataset* pela equipa docente ao grupo 22 é o *Wine Quality Classification*, que tem como objetivo determinar a qualidade de um vinho tinto produzido em Portugal, com base num conjunto de parâmetros como o pH, a percentagem de álcool, a quantidade de ácido cítrico, entre outros.

### 2.1 Relevância e Utilidade do *Dataset* no mundo real

Uma simples pesquisa no *Google* mostra que Portugal é um dos países com maior consumo de vinho *per capita*. Um cidadão comum que tenha adquirido recentemente o gosto pelos vinhos, e até mesmo aqueles que disfrutam à mais tempo, pode não saber realmente o que faz um bom vinho. A aplicação de modelos de *Machine Learning* permite-nos descobrir o que faz um vinho de boa qualidade, i.e, determinar quais características são mais indicativas de um vinho de boa qualidade. Assim, é possível fornecer aos consumidores um indicador de qualidade dos vinhos, permitindo que possam escolher e optar por escolhas possivelmente mais atrativas.

### 2.2 Análise Exploratória

Uma exploração aprofundada dos dados permite que se tirem ilações, muitas vezes escondidas, que poderão ser importantes para se compreender o domínio e o problema em mãos.

Começando por observar o *dataset* conclui-se que há um total de 1599 linhas e 12 colunas. Uma rápida análise das primeiras linhas do *dataset* indica que a tendência é a não existência de *missing values* e que todas as variáveis são numéricas (double) o que indica que, à partida, não teremos de realizar conversões. Ainda assim, devemos assegurar que não existem *missing values* e fazer o respetivo tratamento dos dados.

Usando nodos de exploração do KNIME como *Statistics* e *Numeric Outliers* vemos que todos os parâmetros possuem outliers, embora uns mais que outros.

De seguida, observamos as correlações entre as variáveis com as quais estamos a trabalhar. Imediatamente, é possível ver que existem algumas variáveis que estão fortemente correlacionadas com a qualidade. É provável que essas variáveis também sejam as *features* mais importantes no nosso modelo de *machine learning*.

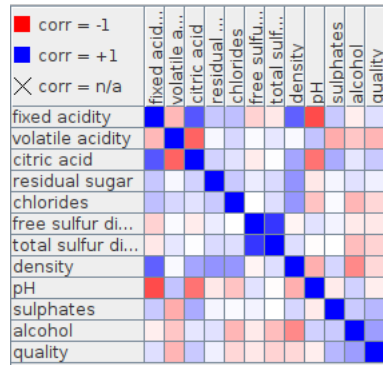


Figura 1: Matriz de correlação obtida pelo nodo *Rank correlation*

Quando a correlação de cada variável é considerada, podemos ver que algumas características estão correlacionadas entre si:

- O álcool está positivamente correlacionado com a qualidade do vinho tinto.
- O álcool tem uma correlação positiva fraca com o valor do pH.
- O ácido cítrico e a densidade têm uma forte correlação positiva com a acidez fixa.
- O pH tem correlação negativa com densidade, acidez fixa, ácido cítrico e sulfatos.

## 2.3 Particionar os dados

Em seguida, dividimos os dados em conjuntos de treino (80%) e de teste (20%) para que possamos validar os nossos modelos e determinar a sua eficácia.

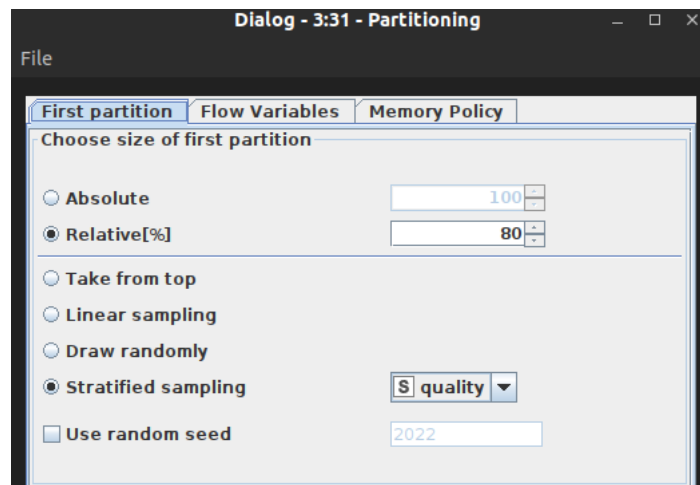


Figura 2: Configuração do nodo *Partitioning*

## 2.4 Modelo KNIME

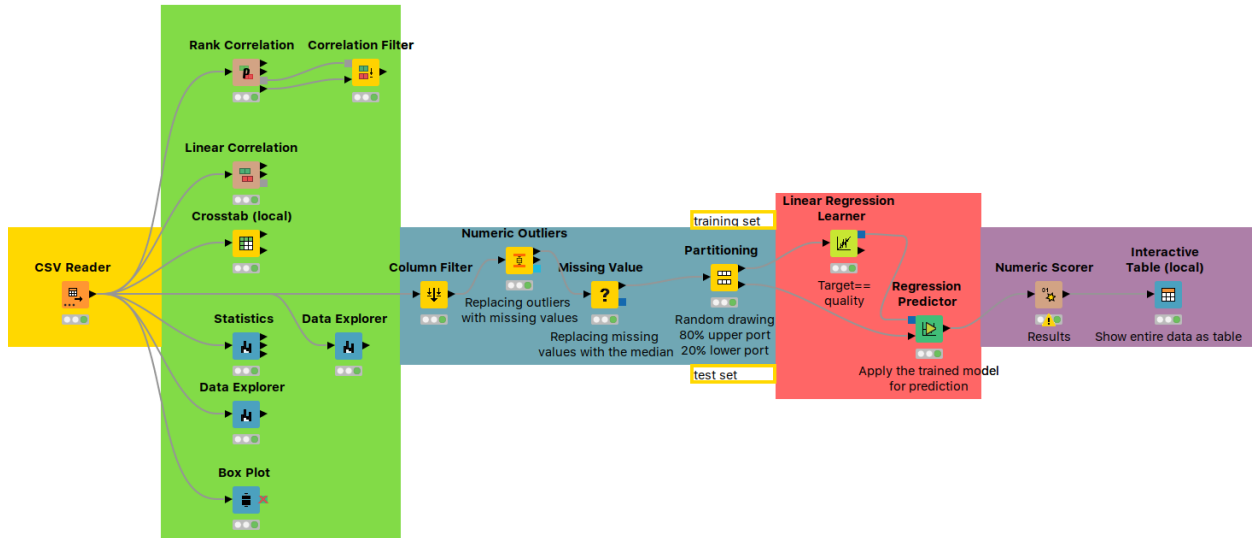


Figura 3: Modelo KNIME com uso de Regressão Linear

A partir do Modelo KNIME representado acima, construído pelo grupo, serão concebidos os modelos de aprendizagem fazendo as alterações necessárias. Inicialmente temos um nó *Linear Regression Learner* para treinar um modelo de regressão e um nó *Linear Regression Predictor* para obter previsões utilizando o modelo treinado. Aquando da conceção de diferentes modelos de aprendizagem, estes nodos podem ser alterados para satisfazer os objetivos pretendidos. Adicionalmente, podem ser adicionados nodos como *Scatter plot* como forma de observar os valores previstos versus valores reais, ou nodos para calcular o erro residual e analisar a sua distribuição.

## 2.5 Conceção dos Modelos de Aprendizagem

Para o propósito deste projeto, iremos comparar três modelos diferentes de *Machine Learning*: Decision Trees, Random Forests e Linear Regression.

### 2.5.1 Decision Trees

Modelo onde se usa uma árvore de decisão (como modelo preditivo) para ir de observações sobre  $X$  (representado nos ramos) até conclusões sobre o valor alvo de  $X$  (representado nas folhas). Intuitivas e fáceis de construir, mas ficam aquém quando se trata de precisão.

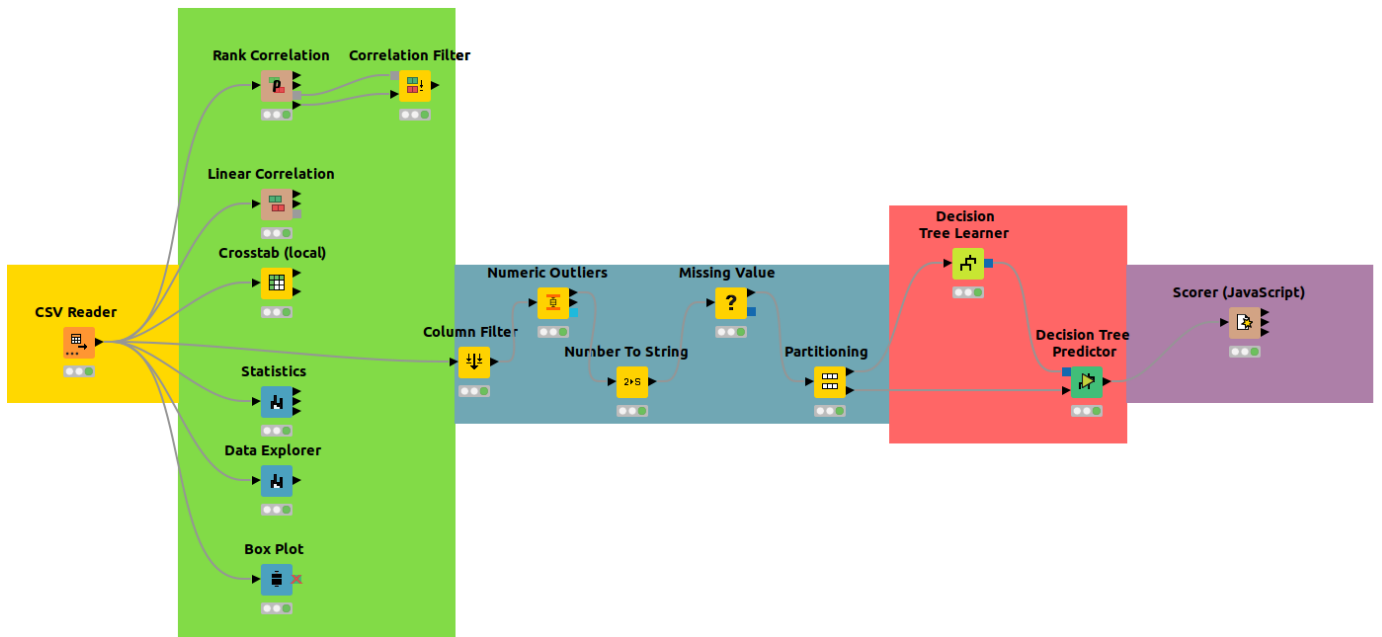


Figura 4: Modelo KNIME para *Decision Trees*

Scorer View

Confusion Matrix

|            | 0 (Predicted) | 1 (Predicted) |        |
|------------|---------------|---------------|--------|
| 0 (Actual) | 264           | 19            | 93.29% |
| 1 (Actual) | 16            | 21            | 56.76% |
|            | 94.29%        | 52.50%        |        |

Overall Statistics

| Overall Accuracy | Overall Error | Cohen's kappa ( $\kappa$ ) | Correctly Classified | Incorrectly Classified |
|------------------|---------------|----------------------------|----------------------|------------------------|
| 89.06%           | 10.94%        | 0.483                      | 285                  | 35                     |

Figura 5: Resultados *Decision Trees* - *Confusion Matrix*

### 2.5.2 Random Forests

Aprendizagem supervisionada que se baseia em árvores de decisão para resolver problemas de regressão construindo uma infinidade de árvores de decisão quando do *training* e dando como *output* a média da *prediction* das árvores individuais. Ao contar com um modelo de “ganhos da maioria”, reduz o risco de erro de uma árvore individual.

O projeto KNIME para o modelo de aprendizagem *Random Forest* é semelhante ao do modelo da aprendizagem baseada em *Decision Trees* (Figura 4), mas sofreu algumas alterações. Naturalmente, agora são usados os nodos *Random Forest*

*Learner e Random Forest Predictor.*

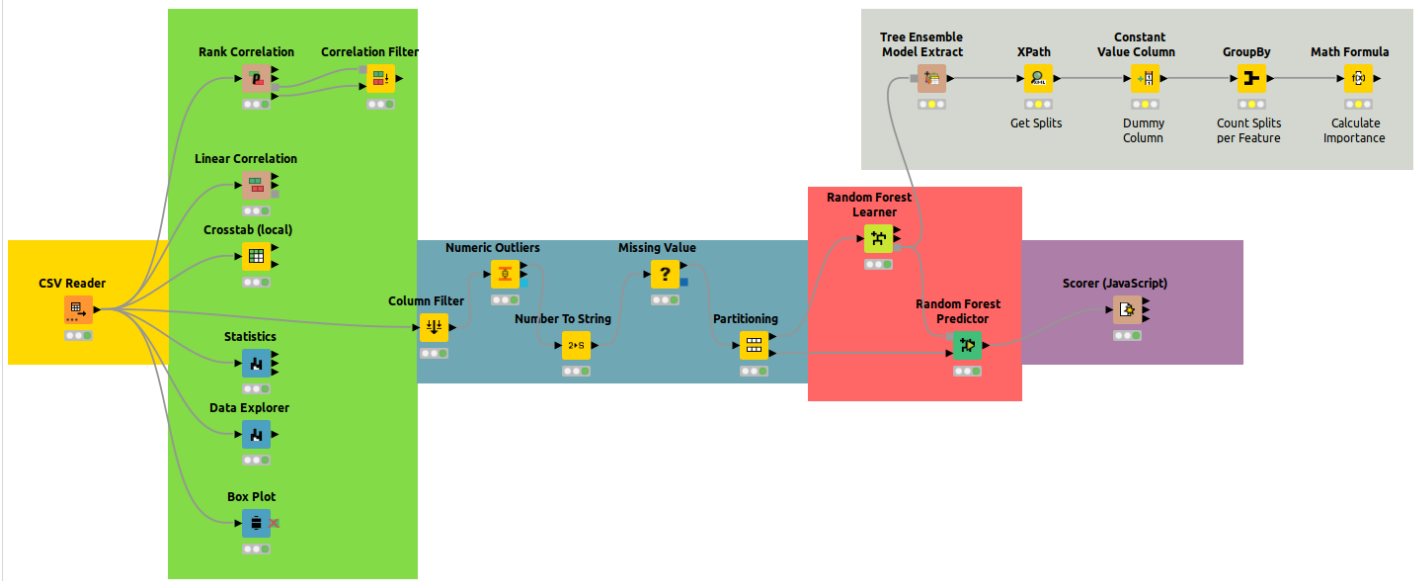


Figura 6: KNIME - Random Forest

Scorer View

Confusion Matrix

|            | 0 (Predicted) | 1 (Predicted) |        |
|------------|---------------|---------------|--------|
| 0 (Actual) | 275           | 2             | 99.28% |
| 1 (Actual) | 29            | 14            | 32.56% |
|            | 90.46%        | 87.50%        |        |

Overall Statistics

| Overall Accuracy | Overall Error | Cohen's kappa ( $\kappa$ ) | Correctly Classified | Incorrectly Classified |
|------------------|---------------|----------------------------|----------------------|------------------------|
| 90.31%           | 9.69%         | 0.433                      | 289                  | 31                     |

Figura 7: Resultados *Random Forest* - *Confusion Matrix*

### 2.5.3 XGBoost Linear Ensemble

XGBoost é uma biblioteca de *machine learning* escalável e distribuída de árvore de decisão impulsionada por gradiente, usa aproximações mais precisas para encontrar o melhor modelo de árvore. É um algoritmo de *boosting* que transforma *learners* fracos em fortes. O nodo *XGBoost Linear Ensemble Learner* em específico, aprende um modelo XGBoost baseado em modelo linear para classificação.



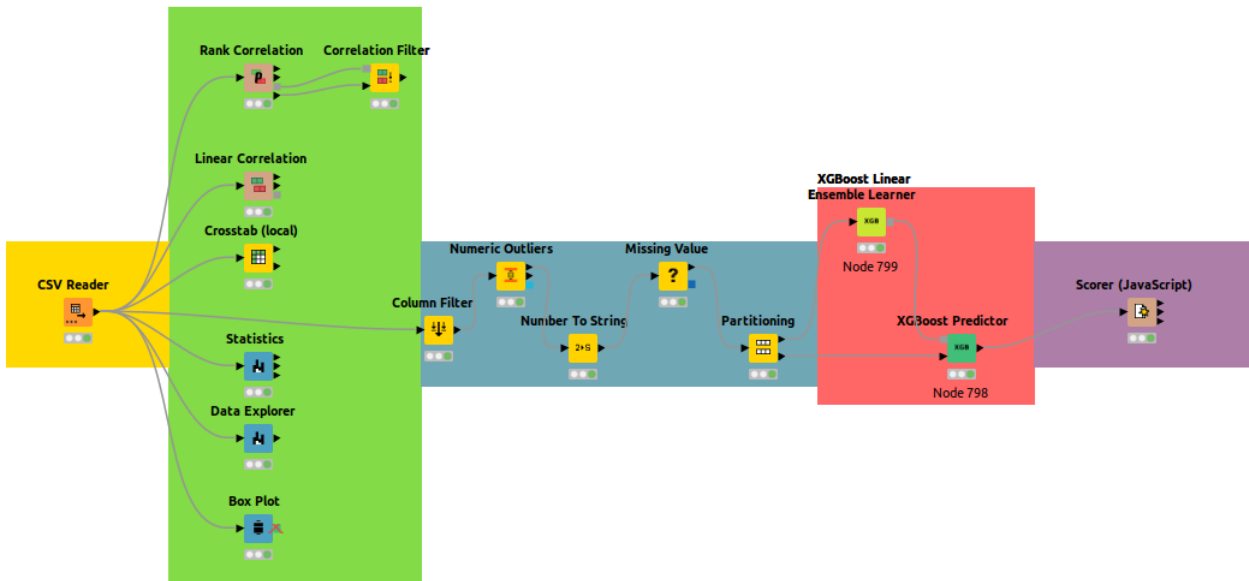


Figura 8: KNIME - XGBoost

**Scorer View**  
Confusion Matrix

|            | 0 (Predicted) | 1 (Predicted) |        |
|------------|---------------|---------------|--------|
| 0 (Actual) | 267           | 10            | 96.39% |
| 1 (Actual) | 30            | 13            | 30.23% |
|            | 89.90%        | 56.52%        |        |

Overall Statistics

| Overall Accuracy | Overall Error | Cohen's kappa ( $\kappa$ ) | Correctly Classified | Incorrectly Classified |
|------------------|---------------|----------------------------|----------------------|------------------------|
| 87.50%           | 12.50%        | 0.331                      | 280                  | 40                     |

Figura 9: Resultados XGBoost - Confusion Matrix

## 2.5.4 Redes Neurais Artificiais

Para utilizar este modelo foi convertido a *feature quality* do tipo *double* para *string*, com recurso ao nodo *Number to String*, de modo a ser possível utilizar um nodo *Scorer (JavaScript)* para obtenção dos resultados. As ANN são modelos computacionais inspirados pelo sistema nervoso central de um animal que são capazes de realizar aprendizagem automática. Neste caso, foram aplicadas a um problema de classificação.



Figura 10: KNIME - ANN

Usando como parâmetros iniciais no nodo *RProp MLP Learner* o número máximo de iterações igual a 1000, 4 camadas escondidas e 8 neurônios escondidos por camada foram obtidos os seguintes resultados:

#### Scorer View

Confusion Matrix

|            | 0 (Predicted) | 1 (Predicted) |        |
|------------|---------------|---------------|--------|
| 0 (Actual) | 262           | 23            | 91.93% |
| 1 (Actual) | 22            | 13            | 37.14% |
|            | 92.25%        | 36.11%        |        |

Overall Statistics

| Overall Accuracy | Overall Error | Cohen's kappa ( $\kappa$ ) | Correctly Classified | Incorrectly Classified |
|------------------|---------------|----------------------------|----------------------|------------------------|
| 85.94%           | 14.06%        | 0.287                      | 275                  | 45                     |

Figura 11: Resultados ANN (Parâmetros iniciais) - *Confusion Matrix*

## Scorer View

Confusion Matrix

|            | 0 (Predicted) | 1 (Predicted) |        |
|------------|---------------|---------------|--------|
| 0 (Actual) | 276           | 9             | 96.84% |
| 1 (Actual) | 24            | 11            | 31.43% |
|            | 92.00%        | 55.00%        |        |

Overall Statistics

| Overall Accuracy | Overall Error | Cohen's kappa ( $\kappa$ ) | Correctly Classified | Incorrectly Classified |
|------------------|---------------|----------------------------|----------------------|------------------------|
| 89.69%           | 10.31%        | 0.348                      | 287                  | 33                     |

Figura 12: Resultados ANN (Após *tuning*) - *Confusion Matrix*

### 2.5.5 Obtenção e análise de resultados

#### Modelos produtores de melhores resultados

Após a obtenção de resultados dos quatro modelos de aprendizagem (*Random Forest*, *Decision Trees*, *XGBoost* e *ANN*), conclui-se que *Random Forest* aparenta produzir o mais alto nível de precisão, segundo a nossa experiência, mostrando-se superior às *Decision Trees*. Tal seria de esperar, uma vez que são uma técnica muito mais robusta do que uma única árvore de decisão, pois agregam um grande número de árvores de decisão para limitar o *overfitting* (quando o modelo funciona bem em dados de treino, mas generaliza mal para dados não vistos), bem como o erro devido ao enviesamento e, portanto, produzem melhores resultados.

Apesar das vantagens que esperavam obter teoricamente na utilização do *XGBoost*, observamos que *Random Forests* apresentaram mais *accuracy* (90.31% *versus* 87.50%) do que o algoritmo em questão, além disso, também apresentou mais *accuracy* que as Redes Neurais Artificiais (90.31% *versus* 89.69%). Logo, o que podemos concluir destes resultados é que algoritmos diferentes necessitam de ser ajustados de maneiras diferentes. Por exemplo, um parâmetro importante no *XGBoost* é a taxa de aprendizagem. Ainda assim, após a realização de *tuning*, não conseguimos obter resultados tão ótimos a nível de *accuracy* neste algoritmo como tínhamos obtido com as florestas aleatórias.

#### Importância das *features*

Observando os resultados obtidos, podemos ver que vinhos de boa qualidade têm em média maiores teores de álcool, menor acidez volátil em média, maiores teores de sulfatos em média e teores de açúcares residuais maiores em média.

Abaixo, encontra-se o gráfico da importância das *features* com base no modelo *Random Forest*. As 3 principais características são : álcool, acidez volátil e sulfatos.

| Row ID | S Splits             | D importance |
|--------|----------------------|--------------|
| Row0   | alcohol              | 0.101        |
| Row1   | chlorides            | 0.088        |
| Row2   | citric acid          | 0.092        |
| Row3   | density              | 0.094        |
| Row4   | fixed acidity        | 0.089        |
| Row5   | free sulfur dioxide  | 0.077        |
| Row6   | pH                   | 0.089        |
| Row7   | residual sugar       | 0.083        |
| Row8   | sulphates            | 0.104        |
| Row9   | total sulfur dioxide | 0.095        |
| Row10  | volatile acidity     | 0.089        |

Figura 13: Importância das *features* com base no modelo de *Random Forest*

## Resultado final

O que faz um "Bom vinho" ser bom, de um ponto de vista de *Machine Learning*?

Recordando novamente a matriz de correlação apresentada na *Análise Exploratória* deste problema, e tendo completado treinos e testes de diversos modelos de *Machine Learning*, retiramos algumas conclusões finais com base na (1) análise da correlação das *features* e nos (2) resultados obtidos acerca da importância das mesmas.

É imediato observar que *Features* como acidez fixa e ácido cítrico fazem parte de *features* como valor de pH. Quando o coeficiente se aproxima de -1, também é possível dizer que as variáveis são correlacionadas, mas nesse caso quando o valor de uma variável aumenta (ex: valor de acidez aumenta) o da outra diminui (ex: valor de PH diminui) pois estamos perante uma correlação negativa ou inversa.

As principais descobertas são que a acidez teve um efeito importante na determinação da qualidade do vinho.

Além disso, os consumidores afirmaram que um vinho de alta qualidade não deve conter um excesso de açúcares residuais (açúcares da uva que sobraram após o processo de fermentação). Isto significa que um vinho muito doce não é uma característica de um vinho de alta qualidade. O aumento do nível de álcool também tem sido visto como uma característica de um bom vinho, mas não deve aumentar para quantidades em que o vinho seja categorizado como destilado (os destilados contêm muito mais álcool do que as bebidas fermentadas (*vodka versus cerveja*)).a

## 3 Dataset: "Weather Conditions in World War Two"

Para concretizar o objetivo deste trabalho, i.e, conceção de modelos de aprendizagem, é importante selecionar um *dataset* com utilidade no contexto dos seus problemas no mundo real.

Inicialmente, tinha sido selecionado pelo grupo um *dataset* de classificação do preço de telemóveis em função das suas características, mas após o *checkpoint* foi

feita a decisão de selecionar um *dataset* de regressão para termos oportunidade de trabalhar esse tema, visto que o *dataset* atribuído também era de classificação.

O *dataset* escolhido contém informação sobre as condições climáticas ao longo da 2ª Guerra Mundial. Partindo destes dados, pretende-se prever se há uma relação entre o mínimo e o máximo diário de temperatura, além de se prever o máximo diário dado o mínimo diário de temperatura.

### 3.1 Relevância e Utilidade do Dataset no mundo real

Esta informação é relevante no mundo real, por exemplo no contexto de planejar operações militares em função das temperaturas diárias, de forma a terem uma maior probabilidade de serem bem sucedidas.

Num contexto civil, tem a utilidade de ao descobrir a relação entre o mínimo e máximo diários de temperatura ser possível planejar todo o tipo de atividades, desde momentos ao ar livre, como averiguar a necessidade de utilizar aquecimento ou arrefecimento num dia, ou descobrir o máximo diário de temperatura em função do mínimo diário que normalmente ocorre durante a madrugada, possibilitando planejar o dia em função dessa temperatura mínima medida.

### 3.2 Analise Exploratória

Primeiro, faremos uma análise exploratória de dados para entender como os recursos dos dados estão relacionados uns aos outros. Esta análise ajudará a entender como modelar os dados mais tarde.

Por observação do *dataset* vemos o ficheiro *comma-separated values* possui 119041 linhas e 31 colunas, embora muitas destas sejam desnecessárias para o objetivo a que nos propomos. Existe ainda outro ficheiro CSV de tamanho mais reduzido que contém a localização das estações de medição do tempo, mas que para o caso de uso deste trabalho prático será descartado.

Vemos que existem *missing values* e variáveis numéricas, bem como *strings*. Apesar disto, para o contexto do nosso problema serão removidas todas as colunas não numéricas, além de algumas numéricas, o que indica que, à partida, não teremos de realizar conversões, mantendo apenas as necessárias para a previsão da temperatura, sendo elas: *MaxTemp* e *MinTemp*, que representam o valor da temperatura em graus *celsius* com três casas decimais, respetivamente, a temperatura máxima diária e a temperatura mínima diária para um determinado dia.

Foi aplicado um tratamento aos *outliers* numéricos, sendo que foram transformados em *missing values* que depois foram substituídos pela mediana, sendo que os atributos que estamos a tratar são números (*doubles*).

### 3.3 Particionar os dados

Em seguida, dividimos os dados em conjuntos de treino (80%) e de teste (20%) para que possamos validar os nossos modelos e determinar a sua eficácia.

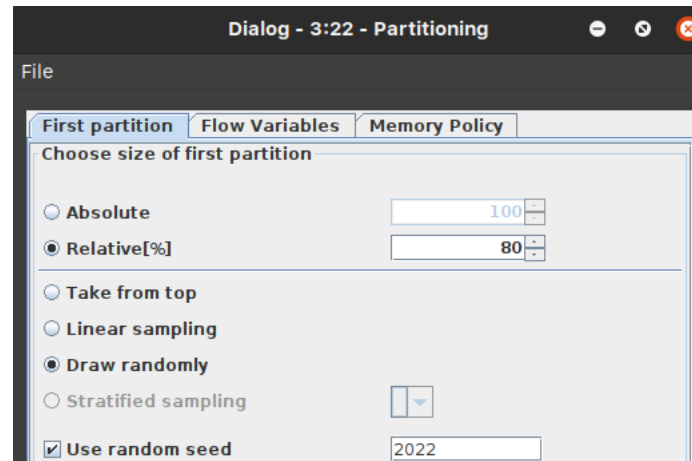


Figura 14: Configuração do nodo *Partitioning*

### 3.4 Conceção de Modelo de Aprendizagem

Neste caso, iremos criar e treinar diversos modelos e, por fim, comparar os resultados obtidos.

#### 3.4.1 Regressão Linear

Em primeiro, iremos conceber um modelo de regressão linear. Regressão linear é uma técnica de *machine learning* usada para estimar o objetivo de um caso de estudo, sendo dado um conjunto dados que caracterizam o comportamento de uma série de casos passados. O objetivo passa por desenvolver um modelo de regressão linear capaz de prever a temperatura máxima diária em função da temperatura mínima diária. Para isso, utilizamos *MinTemp* para prever a *MaxTemp*, através de um nodo de *Linear Regression Learner* e de um *Regression Predictor*.



Figura 15: Modelo KNIME com uso de Regressão Linear

O modelo da figura 15 foi concebido com o intuito de aplicar regressão linear para alcançar o objetivo a que nos propusemos com este *dataset*.

| Statistics - 0:36 - Numer...    |        |
|---------------------------------|--------|
| File                            |        |
| R <sup>2</sup> :                | 0,25   |
| Mean absolute error:            | 2,57   |
| Mean squared error:             | 12,031 |
| Root mean squared error:        | 3,469  |
| Mean signed difference:         | 0,007  |
| Mean absolute percentage error: | 0,094  |
| Adjusted R <sup>2</sup> :       | 0,25   |

Figura 16: Estatísticas obtidas no *Numeric Scorer*

Os resultados obtidos na figura 16, foram obtidos com a partição inicial representada na figura 14 e com os *missing values* a serem substituídos pela mediana.

| Statistics - 0:36 - Numer...    |        |
|---------------------------------|--------|
| File                            |        |
| R <sup>2</sup> :                | 0,263  |
| Mean absolute error:            | 2,555  |
| Mean squared error:             | 11,934 |
| Root mean squared error:        | 3,455  |
| Mean signed difference:         | 0,014  |
| Mean absolute percentage error: | 0,094  |
| Adjusted R <sup>2</sup> :       | 0,263  |

Figura 17: Estatísticas obtidas no *Numeric Scorer*

Na figura 17, foi feita a alteração de dividir os dados através de *linear sampling* em vez de escolha aleatória em função de uma *seed* randomizada e mantendo os valores de 80% para o conjunto de treino e 20% para o conjunto de teste.

Comparando ambos os valores, vemos que utilizando *linear sampling* o MAE e o MSQ foram menores, embora não de forma notória, e o R<sup>2</sup> foi ligeiramente superior.

### 3.4.2 Decision Trees

Sabendo que o modelo de *Decision Trees* é aplicável apenas a problemas de classificação, foi feita a conversão dos dados de *double* para *string* e foram aplicados os nodos correspondentes a árvores de decisão, *Decision Tree Learner* e *Decision Tree Predictor*. Este modelo foi feito com o intuito de verificar se os resultados obtidos seriam aceitáveis para o grupo.

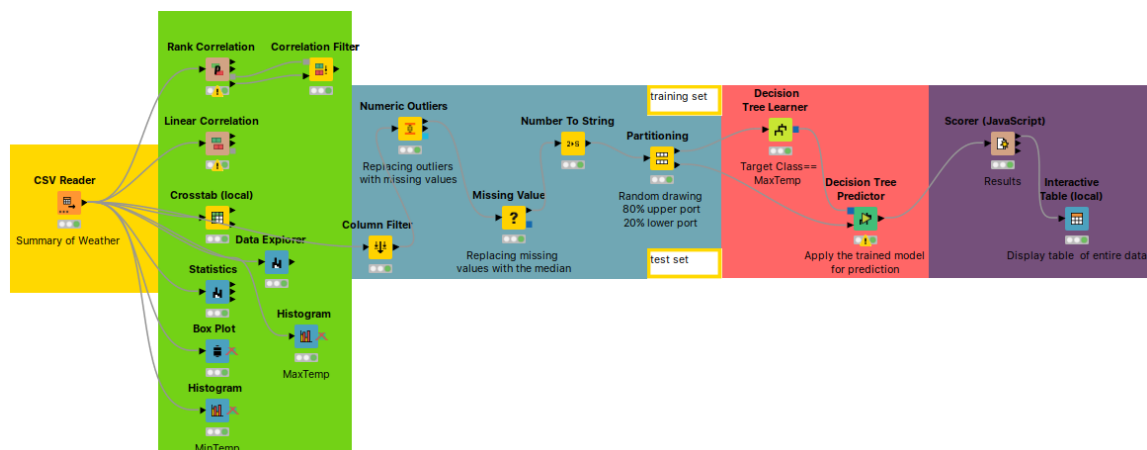


Figura 18: Modelo KNIME com uso de *Decision Trees*



Overall Statistics

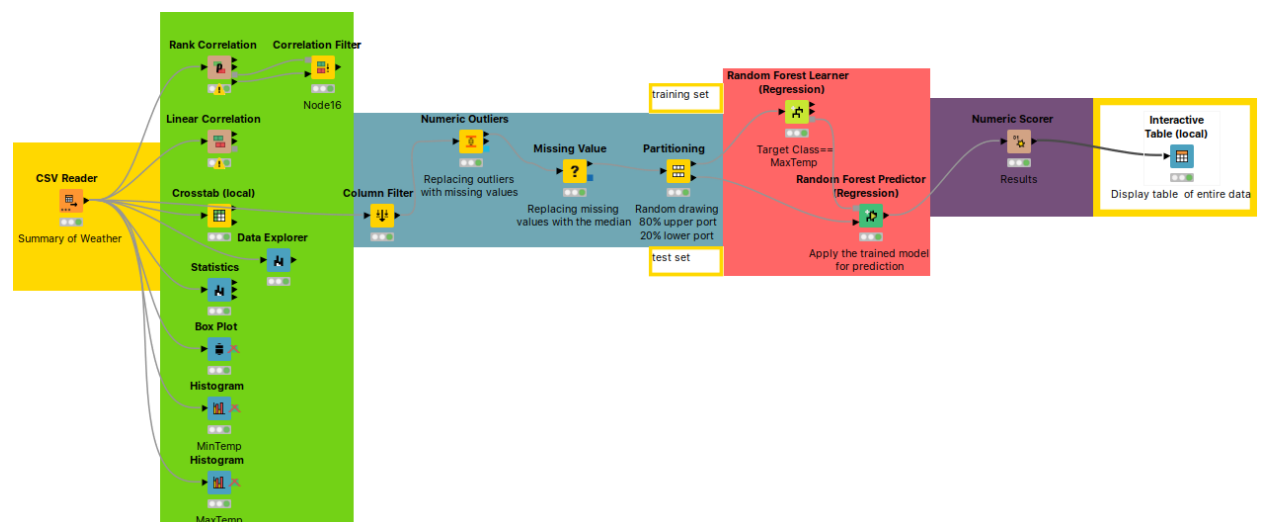
| Overall Accuracy | Overall Error | Cohen's kappa ( $\kappa$ ) | Correctly Classified | Incorrectly Classified |
|------------------|---------------|----------------------------|----------------------|------------------------|
| 19.51%           | 80.49%        | 0.041                      | 4645                 | 19163                  |

Figura 19: Estatísticas obtidas no nodo *Scorer (JavaScript)*

### 3.4.3 Random Forest

Random Forest é um algoritmo de aprendizagem supervisionada que se baseia árvore de decisão, aproveita o poder de diversa árvores de decisão para tomar decisões.

Para se desenvolver o projeto KNIME para o modelo de aprendizagem *Random Forest* são utilizados os nodos *Random Forest Learner* e *Random Forest Predictor*.

Figura 20: Modelo KNIME com uso de *Random Forest*

| Statistics - 3:56 - Numer...    |        |
|---------------------------------|--------|
| File                            |        |
| R <sup>2</sup> :                | 0,298  |
| Mean absolute error:            | 2,497  |
| Mean squared error:             | 11,257 |
| Root mean squared error:        | 3,355  |
| Mean signed difference:         | 0,013  |
| Mean absolute percentage error: | 0,091  |
| Adjusted R <sup>2</sup> :       | 0,298  |

Figura 21: Estatísticas obtidas no *Numeric Scorer*

### 3.4.4 Redes Neuronais Artificiais

As Redes Neuronais Artificiais são um modelo computacional que consiste em vários elementos de processamento que recebem *input* e produzem *outputs* baseados nas suas funções de ativação pré-definidas, sendo que podem ser aplicadas tanto para problemas de classificação como de regressão. Neste caso, serão aplicadas ao problema de regressão em causa. Para se desenvolver o projeto KNIME para o modelo de aprendizagem de ANN são utilizados os nodos *RProp MLP Learner* e *MultiLayerPerceptron Predictor*.

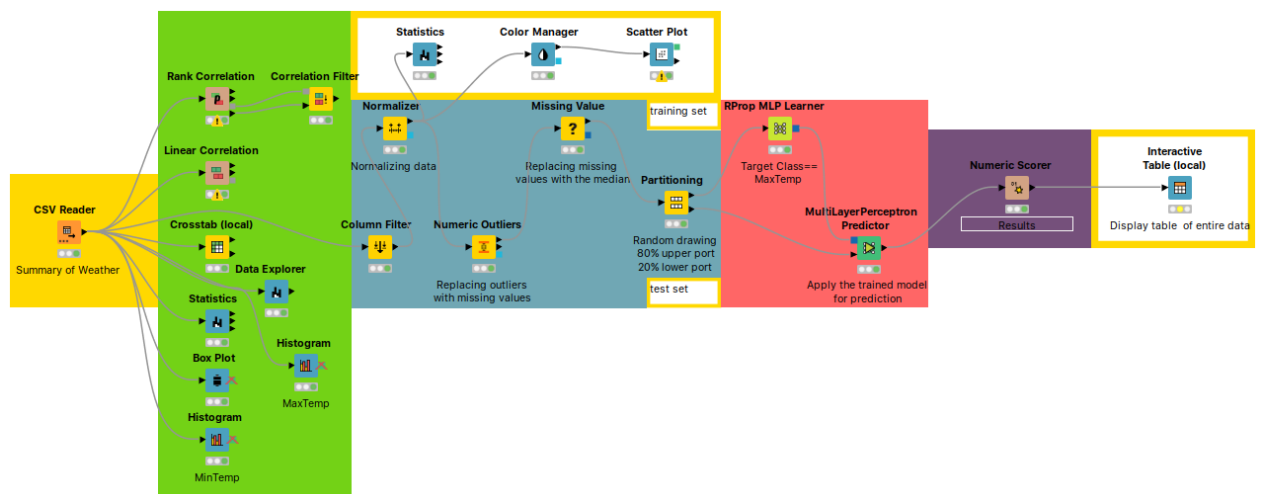
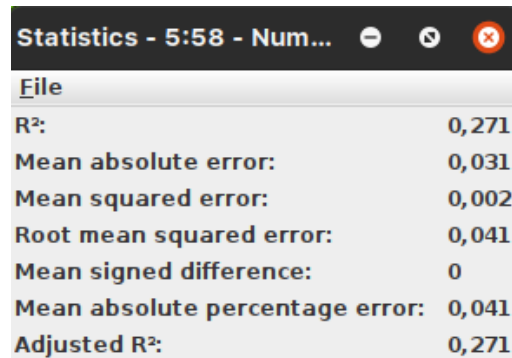


Figura 22: Modelo KNIME com uso de *Redes Neuronais Artificiais*

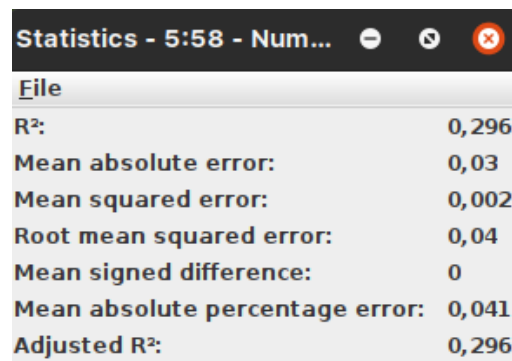
Sendo que é ser recomendado fazer normalização dos dados para ANN, optamos por *Min-Max Normalization* sendo o mínimo igual a 0.0 e o máximo igual a 1.0.



| Statistics - 5:58 - Num...      |       |
|---------------------------------|-------|
| File                            |       |
| R <sup>2</sup> :                | 0,271 |
| Mean absolute error:            | 0,031 |
| Mean squared error:             | 0,002 |
| Root mean squared error:        | 0,041 |
| Mean signed difference:         | 0     |
| Mean absolute percentage error: | 0,041 |
| Adjusted R <sup>2</sup> :       | 0,271 |

Figura 23: Estatísticas obtidas no nodo *Numeric Scorer* com 100 iterações, 5 camadas e 10 neurônios por camada

Após ter sido feito o *tuning* do modelo, alterando o número de iterações, número de camadas escondidas e número de neurônios por camada obteve-se o seguinte resultado no *Numeric Scorer*:



| Statistics - 5:58 - Num...      |       |
|---------------------------------|-------|
| File                            |       |
| R <sup>2</sup> :                | 0,296 |
| Mean absolute error:            | 0,03  |
| Mean squared error:             | 0,002 |
| Root mean squared error:        | 0,04  |
| Mean signed difference:         | 0     |
| Mean absolute percentage error: | 0,041 |
| Adjusted R <sup>2</sup> :       | 0,296 |

Figura 24: Estatísticas obtidas no nodo *Numeric Scorer* com 1000 iterações, 5 camadas e 10 neurônios por camada

Analisando ambos os resultados estatísticos obtidos, podemos averiguar que não foi possível alterar significativamente o **MAE**, o **MSE** e o **RMSE**, apenas o **R<sup>2</sup>** foi alterado através do *tuning* nesta situação, melhorando ligeiramente a confiança do modelo.

#### 3.4.5 Obtenção e análise de resultados

##### Modelos produtores de melhores resultados

Após a obtenção de resultados dos quatro modelos de aprendizagem (*Linear Regression*, *Random Forest*, *Decision Trees* e *Artificial Neural Networks (ANN)*), conclui-se que *Redes Neurais* aparenta produzir o mais alto nível de precisão, segundo a nossa experiência, mostrando-se superior aos outros. O resultado esperado era precisamente este, pois as *Redes Neurais* ..... O modelo que obteve

os piores resultados foi, como era esperado, o das *Árvores de Decisão*, visto ser um modelo para problemas de classificação foi necessário fazer alterações aos dados para ser possível aplicar os nodos necessários à utilização deste modelo. O modelo de *Random Forest* agrega um grande número de árvores de decisão sendo selecionadas algumas amostras dos dados, de maneira aleatória. Para a criação dos nós das árvores, também existirá uma etapa aleatória, onde algumas variáveis serão selecionadas da mesma forma e, por isso, obtém-se um resultado aceitável na visão do grupo e ligeiramente superior ao do modelo de *Regressão Linear*, que era esperado ser o que produzisse os resultados menos precisos entre os 3 modelos de regressão utilizados (Regressão Linear, *Random Forests* e Redes Neurais Artificiais).

Comparando o **MAE**, **MSE** e o **RMSE** obtidos nos três modelos de regressão, conseguimos observar que o modelo que fez uso de *Redes Neurais Artificiais* teve resultados inferiores, logo foi superior a determinar a temperatura máxima diária a partir da temperatura mínima. Modificando certos parâmetros é possível obter resultados diferentes, tanto para melhor como para pior, mas os resultados apresentados são fruto de certas experiências e variações de forma a tentar melhorar e produzir resultados com o menor erro possível, o chamado *tuning*.

### Resultado final

Após a obtenção do resultado dos modelos construídos, podemos inequivocamente afirmar que existe uma relação entre o máximo e o mínimo diários de temperatura e que é, de facto, possível prever, o primeiro em função do segundo, embora analisando o  $R^2$  dos modelos podemos inferir que a relação entre ambas as temperaturas não é a melhor, ou seja, menos de metade da variação observada pode ser explicada pelos *inputs* dos modelos criados pelo grupo.

## 4 Conclusão

A realização deste trabalho prático permitiu aprofundar a concepção e utilização de modelos de aprendizagem abordados ao longo do semestre nas aulas da Unidade Curricular de Aprendizagem e Decisão Inteligentes, abordando as várias etapas que constituem o *Machine Learning* desde a exploração, análise/preparação dos datasets, aplicação de metodologias de *Machine Learning* até à obtenção e análise crítica de resultados definindo a sua utilidade no contexto dos problemas subjacentes.

A natureza prática deste trabalho permitiu conciliar os conhecimentos adquiridos nas aulas teóricas com exemplos de aplicação com datasets do mundo real, concretizando os objetivos de aprendizagem.