



UNIVERSIDADE DO MINHO  
LICENCIATURA EM ENGENHARIA INFORMÁTICA

Aprendizagem e Decisão Inteligentes  
Grupo 19

Eduardo Pereira (A94881)      Gonçalo Freitas (A96136)  
Gonçalo Vale (A96923)

Ano Letivo 2022/2023



# Introdução

No âmbito da UC de Aprendizagem e Decisão Inteligentes, foi-nos pedida a exploração, análise e preparação de um *dataset* escolhido por nós e de um *dataset* do conhecimento dos docentes, assim como o desenvolvimento de modelos de aprendizagem através da ferramenta KNIME, com o intuito de analisar e extrair conhecimento dos mesmos.

## Metodologia

Para o desenvolvimento deste trabalho, em ambas as tarefas, o grupo decidiu adotar a metodologia **CRISP-DM** (*Cross Industry Standard Process for Data Mining*). Esta metodologia visa definir um "guião" para projetos de Análise de Dados. Este "guião" divide-se em 6 passos (Estudo do negócio, Estudo dos dados, Preparação dos dados, Modelação, Avaliação e Desenvolvimento). Como estes passos são bastante similares aos conteúdos lecionados no âmbito desta UC, decidimos então adotar esta metodologia.

## Tarefa A

### Dataset Link

### *Dataset Global Video Game Sales & Ratings*

### Objetivos

Depois de uma primeira análise aos conteúdos do *dataset*, o grupo achou que seria mais interessante tentar elaborar modelos de forma a prever as vendas globais e a faixa etária à qual o jogo se destina. Para isso, começaremos por estudar os dados, de seguida elaborar um tratamento de dados, de forma a eliminar *missing values*. Só depois disso avançaremos para a construção dos modelos em si e faremos uma avaliação dos resultados.

### Estudo de Dados

Por consenso do grupo, foi decidido fazermos o estudo do dataset "*Dataset Global Video Game Sales & Ratings*". Este *dataset* consiste em registos de vários jogos do website "metacritic.com". Ao descarregar o *dataset*, são-nos apresentados três ficheiros CSV. Destes, escolhemos utilizar o ficheiro "Raw data.csv", pois permite-nos um maior controlo sobre os dados, visto que não apresentam qualquer pré-tratamento. Mais concretamente, este *dataset* possui os dados referentes a 16718 video-jogos e contém os seguintes 16 campos de informação(colunas):

- **Name** - Nome do jogo;
- **Platform** - Plataforma para o qual o jogo foi construído;
- **Year\_of\_Release** - Ano em que o jogo foi lançado para o público geral;
- **Genre** - Género em qual o jogo se enquadra;

- **Publisher** - Empresa que publicou o jogo;
- **NA\_Sales** - Número de vendas na região da América do Norte;
- **EU\_Sales** - Número de vendas na região da Europa;
- **JP\_Sales** - Número de vendas no Japão;
- **Other\_Sales** - Número de vendas no resto do mundo;
- **Global\_Sales** - Número de vendas total;
- **Critic\_Score** - Avaliação dos críticos;
- **Critic\_Count** - Número de críticos que contribuíram para o *Critic\_Score*;
- **User\_Score** - Avaliação dos jogadores;
- **User\_Count** - Número de jogadores que contribuíram para o *User\_Score*;
- **Developer** - Empresa que fez o jogo;
- **Rating** - Faixa etária à qual o jogo se destina.

De forma a explorar os dados, decidimos recorrer aos nodos "*Rank Correlation*" (devido ao facto de haverem variáveis discretas e contínuas), "*Data Explorer*", "*Scatter Plot*" e "*Statistics*" do KNIME, de forma a observar que tipos de relações existem entre os dados, quais os dados mais relevantes (isto é, de onde se pode extrair mais conhecimento) e ver a qualidade dos mesmos.

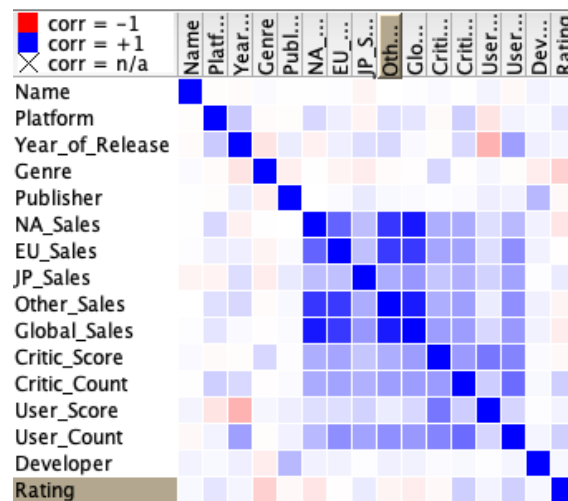


Figura 1: Correlação entre as variáveis do *dataset*

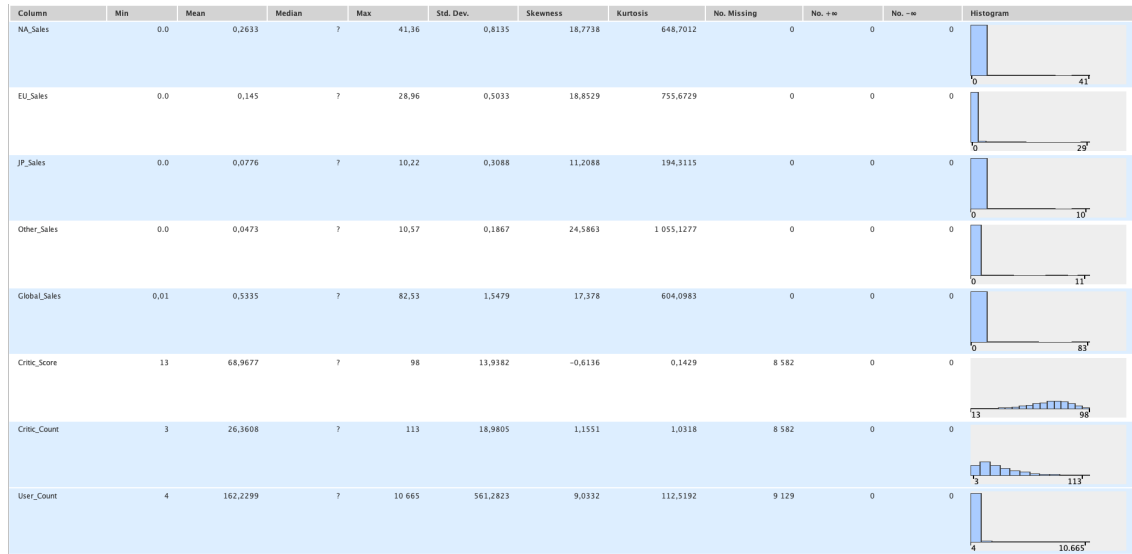


Figura 2: Resultados do nodo Statistics

Após uma análise aos resultados dos nodos "*Rank correlation*", "*Data Explorer*" e "*Statistics*", podemos afirmar que este *dataset* apresenta vários problemas, nomeadamente uma grande quantidade de *missing values* e uma *skewness* muito elevada. Para além disso, e focando mais nos resultados do nodo "*Rank correlation*", podemos destacar os campos que têm uma maior influência no valor das *Global\_Sales* e *Rating*, variáveis estas que queremos prever, usando modelos de regressão e classificação, respetivamente. Assim, decidimos destacar os seguintes campos para a colunas das *Global\_Sales*:

- Platform
- NA\_Sales
- EU\_Sales
- JP\_Sales
- Other\_Sales
- Critic\_Score
- Critic\_Count
- User\_Score
- User\_Count

Assim os campos *Name*, *Developer*, *Year\_of\_Release*, *Genre*, *Publisher* e *Rating* não irão estar em grande plano na previsão deste modelo.

Por outro lado, os valores que se destacam em relação à coluna do *Rating* são:

- Platform

- NA\_Sales
- EU\_Sales
- Global\_Sales
- Other\_Sales
- Critic\_Score
- Critic\_Count
- User\_Score

Desta forma, aos campos: *JP\_Sales*, *User\_Count* e *Developer* não será dada uma grande importância na previsão deste modelo.

Através do nodo "*Statistics*", notamos que as colunas relativas às vendas, têm um *skewness* muito elevado o que significa que a distribuição dos dados é altamente assimétrica, com uma cauda mais longa em uma direção do que noutra. Sendo o *skewness* positivo, significa que há mais valores concentrados na cauda esquerda da distribuição e a média é maior do que a mediana. Este problema já era esperado devido as grandes flutuações no sucesso de jogos observável no dia a dia.

## Preparação de Dados

Numa primeira vista do dataset depara-mo-nos com o facto da coluna *User\_Score* estar em formato String. Verificamos que a razão disto acontecer é o facto de algumas linhas terem o valor "tbd", que significa *to be determined*. Então, utilizamos o nodo "*String Replacer*" de forma a substituir qualquer ocorrência deste valor por um *missing value*. Agora apenas com Strings de numeros e *missing values*, podemos transformar esta coluna de Strings para Doubles, através do nodo *String to Number*, de forma a melhor poder-mos analisar a informação que nos é dada. Usamos também este tratamento para o campo *Year\_of\_Release*, o qual contém instâncias da string "N/A", ou valor indisponível, a qual é transformada num *missing value* e posteriormente este campo é transformado em inteiros. Este processo é realizado de modo a melhorar os resultados das estatísticas colecionadas pelo nodo *Statistics*.

Após a análise inicial do dataset escolhido é óbvio que este possui um número considerável de *missing values*. De forma a combater o problema dos *missing values* e manter a credibilidade dos dados a estudar escolhemos inicialmente a remoção das linhas que contém *missing values*. Deste modo, e aplicando um nodo "*Extract Table Dimension*" após o nodo "*Missing Value*", obtemos o seguinte resultado:

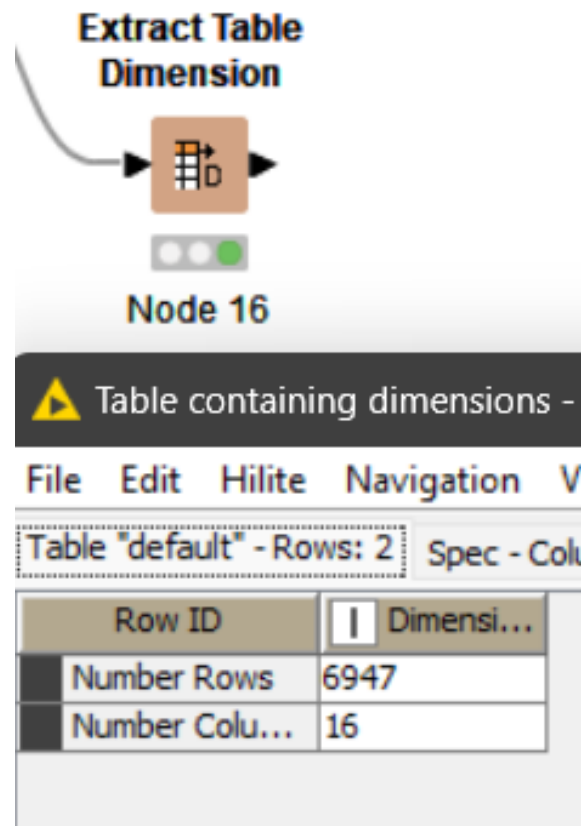


Figura 3: Table dimension da tabela resultante da remoção das linhas que contêm missing values

Com este resultado concluímos que o uso dessa técnica de tratamento de *missing values* retirava muita informação à base de conhecimento (cerca de 10000 linhas), por isso decidimos adotar outra. Então, optamos por uma técnica mista. Para campos com o tipo String, quando existem *missing values*, estes são substituídos pelo valor mais frequente. Para o resto, são substituídos pela média.

Para complementar este nodo, decidimos recorrer ao nodo *Numeric Outliers*, de forma a retirar os *outliers* da base de conhecimento e obter uma aprendizagem mais correta.

A utilização de técnicas de integração de dados é útil em diversas situações, tais como na análise de dados de diversas fontes para obter uma visão mais completa dos negócios. Neste caso, esse passo não será necessário visto que esta técnica não se aplica quando apenas temos uma fonte de dados, como é o caso.

## Modelação

Findada a etapa de preparação dos dados, passamos á fase de realizar os modelos de aprendizagem, onde iremos tentar atingir o objetivo proposto anteriormente, de fazer a previsão das *Global\_Sales* e *Rating*, com base nos restantes atributos.

Primeiramente, como o atributo *Global\_Sales* é um atributo não discreto, trata-se de um problema de regressão, pelo que iremos, primeiramente, recorrer a *Linear Regression*, utilizando os nodos do KNIME *Linear Regression Learner* e *Regression Predictor*, com supervisão.

## Modelo de Regressão

De forma a validarmos os resultados da aprendizagem, e como apenas temos uma fonte de dados, iremos recorrer ao nodo *Partitioning* do KNIME, utilizando 75% dos dados como dados de treino, e os restantes 25% como dados de teste. Apesar de termos apenas uma fonte de dados, essa fonte de dados apresenta um tamanho grande, por isso decidimos não recorrer ao nodo *X-Partitioner* do KNIME, visto não acharmos necessário fazer mais do que uma validação.

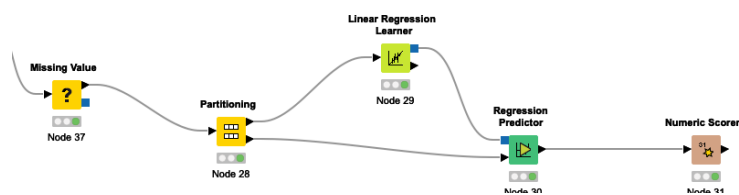


Figura 4: Modelo de Regressão

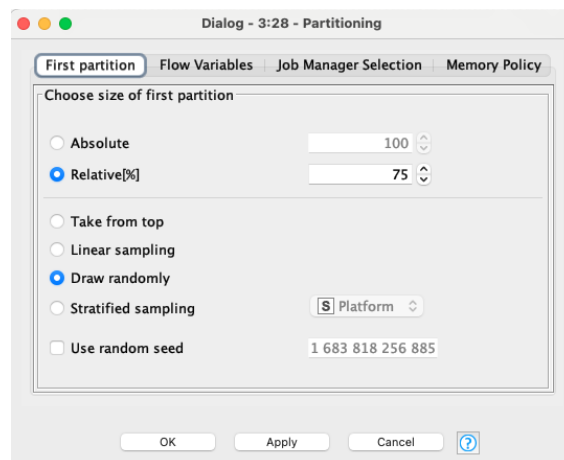


Figura 5: Partitioning

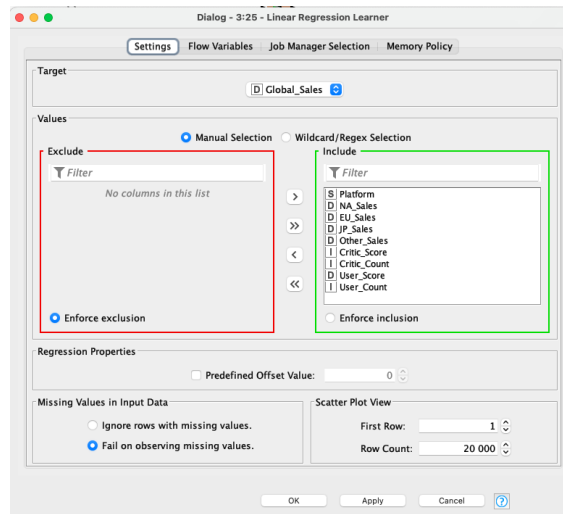


Figura 6: Linear Regression Learner

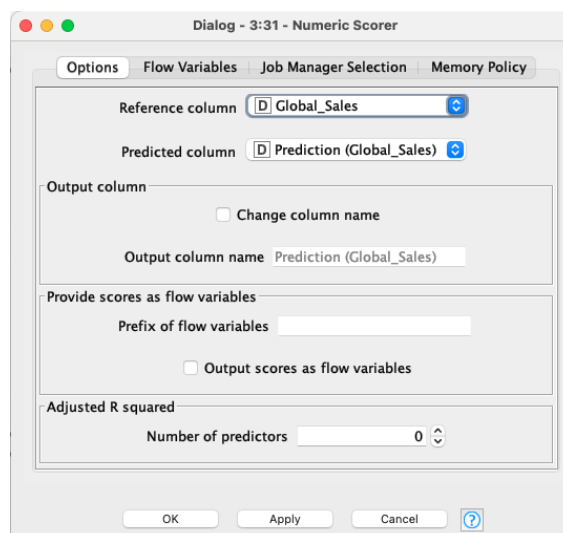


Figura 7: Numeric Scorer

Assim, aplicando os nodos de *Linear Regression* no KNIME ao *dataset* já preparado, obtivemos os seguintes resultados através do nodo *Numeric Scorer*:



File	
R <sup>2</sup> :	0,538
Mean absolute error:	0,093
Mean squared error:	0,024
Root mean squared error:	0,155
Mean signed difference:	-0,003
Mean absolute percentage error:	0,893
Adjusted R <sup>2</sup> :	0,538

Figura 8: Resultados da previsão utilizando o nodo Numeric Scorer

Depois de obtermos estes resultados, podemos afirmar que a previsão efetuada pelo modelo foi razoável, visto que se obteve um  $R^2$  de 0.538.

Devido ao facto de este *dataset* apresentar uma grande quantidade de *missing values*, e apesar de este método não ser muito aconselhado devido á grande perda de informação que este proporciona, decidimos testar o modelo, mas agora removendo todas as linhas que tivessem *missing values*. Estes foram os resultados:

File	
R <sup>2</sup> :	0,999
Mean absolute error:	0,004
Mean squared error:	0
Root mean squared error:	0,006
Mean signed difference:	-0
Mean absolute percentage error:	0,043
Adjusted R <sup>2</sup> :	0,999

Figura 9: Resultados da previsão utilizando o nodo Numeric Scorer, removendo linhas com missing values

### Modelo de Classificação

Realizado o modelo de regressão para o atributo das *Global\_Sales*, avançamos para o modelo de classificação com o objetivo de prever o atributo *Rating*. Para resolver este problema de classificação, recorremos a *Árvores de Decisão*, utilizando os nodos do *KNIME: Decision Tree Learner* e *Decision Tree Predictor*, com supervisão.

Quanto ao estudo dos dados, decidimos escolher as variáveis para a previsão segundo o nodo ”*Rank Correlation*”. Depois da análise dos resultados desse nodo (imagem já colocada no problema anterior), escolhemos os atributos que apresentavam mais correlação em relação ao campo de informação *Rating*. As variáveis escolhidas foram: *Platform*, *Year\_of\_Release*, *JP\_Sales*, *Critic\_Count*, *User\_Score* e *User\_Count*.

Quanto ao tratamento de dados, a única diferença para o outro modelo foi a mudança do nodo ”*Column Filter*”, no qual selecionamos apenas as variáveis referenciadas no último parágrafo. Para além disso, decidimos que o tratamento de *missing values* seria feito com substituição em strings pela mais frequente e nos restantes tipos de dados pela média

De forma a validarmos os resultados da aprendizagem, e como apenas temos uma fonte de dados, recorreremos ao nodo *Partitioning* do KNIME, utilizando 70% dos dados como dados de treino, e os restantes 30% como dados de teste. Apesar de termos apenas uma fonte de dados, essa fonte de dados apresenta um tamanho grande, por isso decidimos não recorrer ao nodo *X-Partitioner* do KNIME, visto não acharmos necessário fazer mais do que uma validação.



Figura 10: Modelo de Classificação

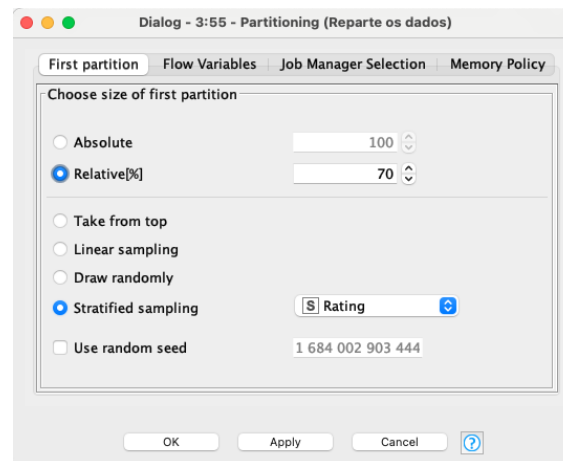


Figura 11: Partitioning

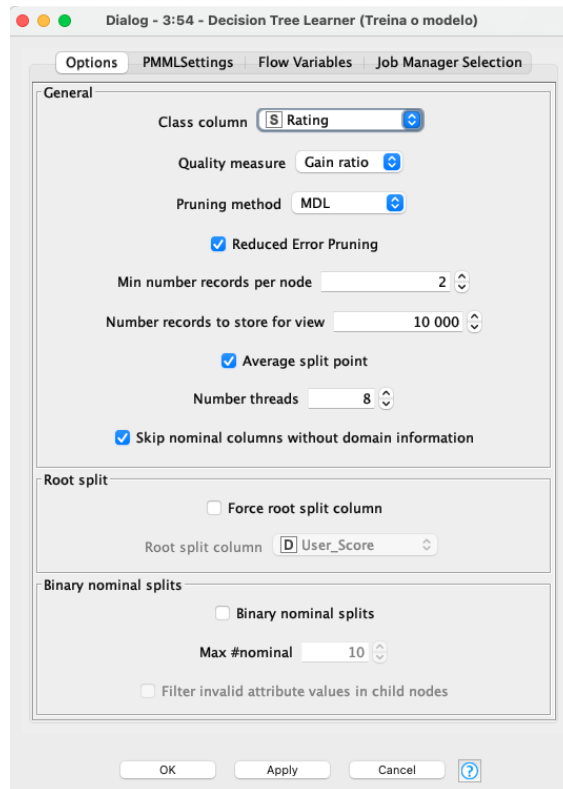


Figura 12: Decision Tree Learner

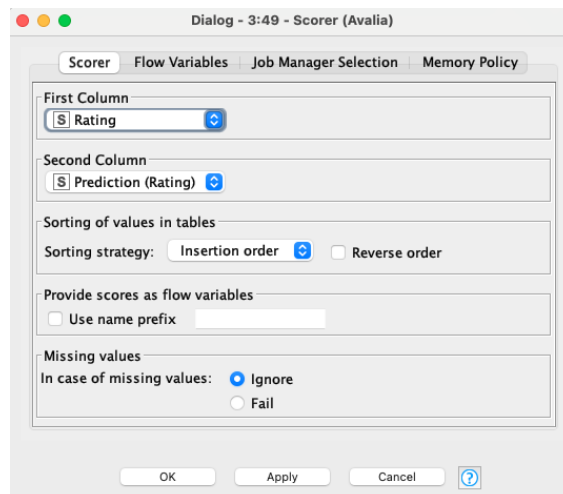
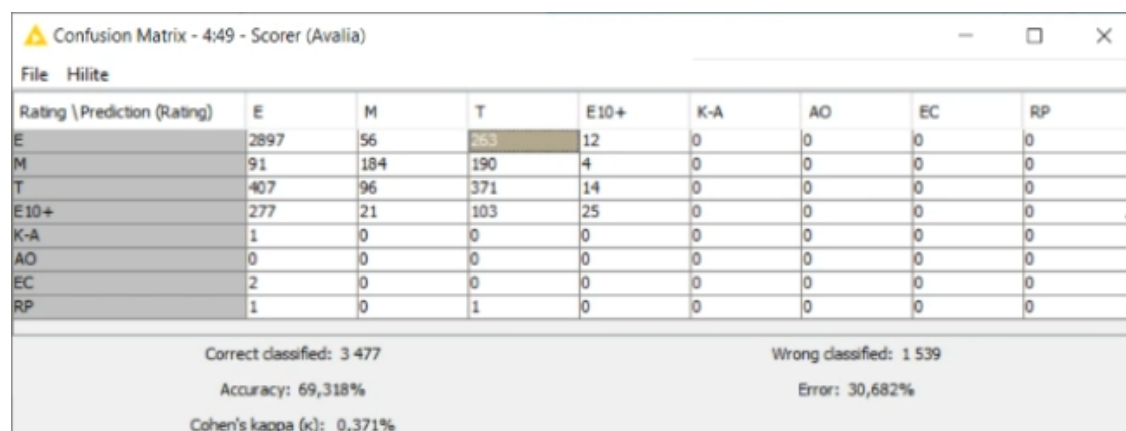


Figura 13: Scorer

Assim, aplicando os nodos de *Decision Tree Learner* no KNIME ao *dataset* já preparado, obtemos os seguintes resultados através do nodo *Scorer*:



The screenshot shows the 'Confusion Matrix - 4:49 - Scorer (Avalia)' window. It contains a confusion matrix table and summary statistics.

Rating \ Prediction (Rating)	E	M	T	E10+	K-A	AO	EC	RP
E	2897	56	263	12	0	0	0	0
M	91	184	190	4	0	0	0	0
T	407	96	371	14	0	0	0	0
E10+	277	21	103	25	0	0	0	0
K-A	1	0	0	0	0	0	0	0
AO	0	0	0	0	0	0	0	0
EC	2	0	0	0	0	0	0	0
RP	1	0	1	0	0	0	0	0

Summary statistics:

- Correct classified: 3 477
- Wrong classified: 1 539
- Accuracy: 69,318%
- Error: 30,682%
- Cohen's kappa (κ): 0,371%

Figura 14: Resultados da previsão utilizando o nodo Scorer

## Avaliação

### Modelo de Regressão

Depois de uma análise mais cuidada dos resultados obtidos pelo modelo que construímos, podemos concluir que, apesar da grande perda de dados na base de conhecimento, a segunda técnica de tratamento de *missing values* permitiu ao nosso modelo realizar uma previsão mais precisa, visto o  $R^2$  ter sido aproximadamente 1 quando removemos linhas, e aproximadamente 0.53 quando não removemos linhas. Para além disso, podemos associar esta melhoria na precisão em relação à primeira técnica ao facto de que, na primeira técnica, ao substituir Strings pelo valor mais frequente e ao substituir números inteiros e reais pelas médias, estamos a inserir conhecimento potencialmente errado e distante do valor real. Para além disso, podemos concluir que, devido à precisão elevada que a previsão do nosso modelo obteve, os campos que consideramos mais relevantes depois de estudarmos os dados são, de facto, importantes para determinar o valor do campo *Global\_Sales*.

### Modelo de Classificação

Quanto ao desempenho do modelo de classificação com o intuito de prever os valores da variável "Rating", podemos concluir que o resultado obtido de aproximadamente 69%, com um erro de cerca de 31%, foi bastante razoável dado os problemas do *dataset*. A partir dos 5016 valores previstos, dos quais 1539 estavam errados, e da *confusion matrix* resultante podemos confirmar que o número de previsões erradas para cada *Rating* é diretamente proporcional ao seu número no *dataset* original. Isto serve para confirmar de forma prática que as previsões feitas têm como base as instâncias já dadas pelo *dataset* original.

### Sugestões e Recomendações

Para melhorar os resultados do modelo de classificação, apesar do *dataset* já ser grande, podíamos ter aplicado o nodo "X-Partitioner" de forma a obter mais validações no particionamento dos

dados e, se calhar, obter uma percentagem de precisão mais elevada. Para além disso poderíamos hipoteticamente aplicar um nodo "*Numeric Outliers*" de forma a remover os *outliers* dos dados e ter uma amostra mais regular. Adicionalmente poderíamos desenvolver um modelo de *clustering* de forma a comparar os seus resultados a métodos de aprendizagem supervisionada como os que desenvolvemos.

## Tarefa B

### *Produção de Vestuário*

#### Dataset Link

#### Objetivos

Tal como definido no enunciado, o objetivo dos modelos deste *dataset* será prever os valores do campo "*actual\_productivity*". Para isso, vamos começar por fazer o estudo dos dados, depois iremos fazer o seu tratamento para mais tarde utilizar nos modelos de forma a alcançar o objetivo e obter os melhores resultados possíveis.

#### Estudo de Dados

Este *dataset* contém 1197 linhas, sendo cada uma dividida em 14 campos de informação. Mais especificamente, as seguintes colunas:

Variáveis Independentes:

- **rowID:** ID de registo;
- **date:** Data em DD/MM/YYYY;
- **department:** Departamento associado com a instância;
- **team:** Numero da equipa associada com a instância;
- **targeted\_productivity:** Produtividade desejada decidida pela Autoridade para cada equipa, para cada dia;
- **smv**(*Standard Minute Value*): Tempo alocado a uma tarefa;
- **wip**(*Work in progress*): Inclui o numero de *items* inacabados para produtos;
- **over\_time:** Representa a quantidade de tempo extra de cada equipa em minutos;
- **incentive:** Representa a quantidade de apoio financeiro(em BDT) que permite ou motiva um determinado curso de ação;
- **idle\_time:** A quantidade de tempo em que a produção esteve interrompida devido a diversas razões;
- **idle\_men:** Numero de trabalhadores que estiveram parados devido a interrupções na produção;
- **no\_of\_workers:** Numero de trabalhadores em cada equipa;
- **no\_of\_style\_change:** Numero de mudanças de estilo num produto particular;

Variáveis Dependentes:

- **actual\_productivity:** A percentagem real de produtividade que foi fornecida pelos trabalhadores;

Depois da análise dos resultados do nodo "Statistics", podemos observar que este *dataset* não possui *missing values*. Com o objetivo de analisar as variáveis que apresentam uma maior correlação em relação à variável objetivo, foi utilizado o nodo "Rank Correlation", o nodo "Rank Correlation" é um método estatístico utilizado para avaliar a relação entre duas variáveis quando os dados não possuem uma distribuição normal ou quando os dados são ordinais ou categóricos. Com essa análise, concluímos que os campos de informação a ter em conta, ou seja, os campos de informação que apresentam uma maior relevância para esta previsão são: "targeted\_productivity", "smv", "wip" e "incentive". Pelo contrário, os campos de informação "id", "date", "department", "team", "over\_time", "idle\_time", "idle\_men", "no\_of\_workers" e "no\_of\_style\_change", apresentam um valor de correlação baixo, pelo que não iremos ter em conta no modelo de *machine learning*.

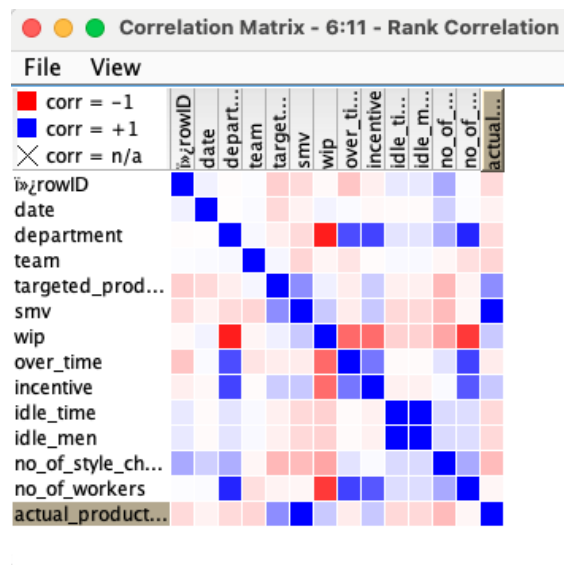


Figura 15: Correlação entre as variáveis do *dataset*

De seguida, analisamos as colunas que apresentaram uma correlação alta em relação à variável objetivo usando o nodo "*Scatter Plot*" que permite visualizar a relação entre duas variáveis num gráfico de dispersão. Através do gráfico gerado é possível identificar se existe alguma correlação entre as variáveis, qual é a direção e a força dessa correlação, assim como identificar possíveis *outliers* e padrões nos dados.

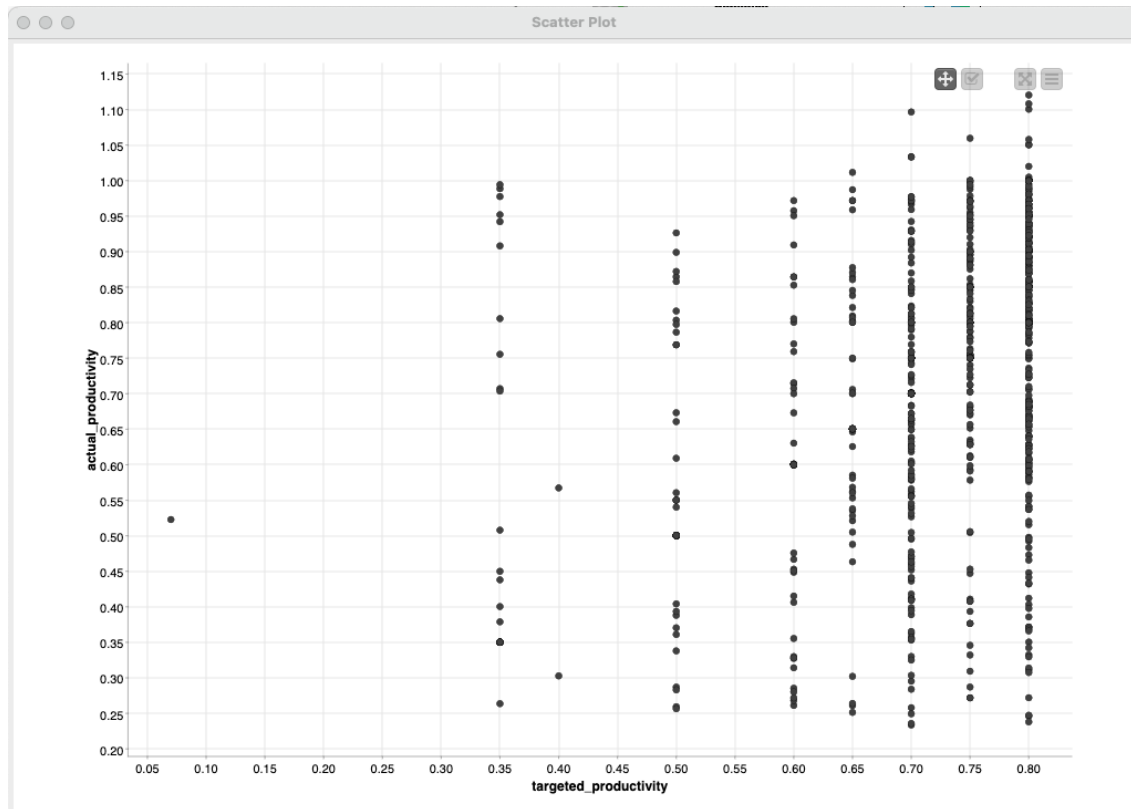


Figura 16: Relação entre *actual\_productivity* e *targeted\_productivity*



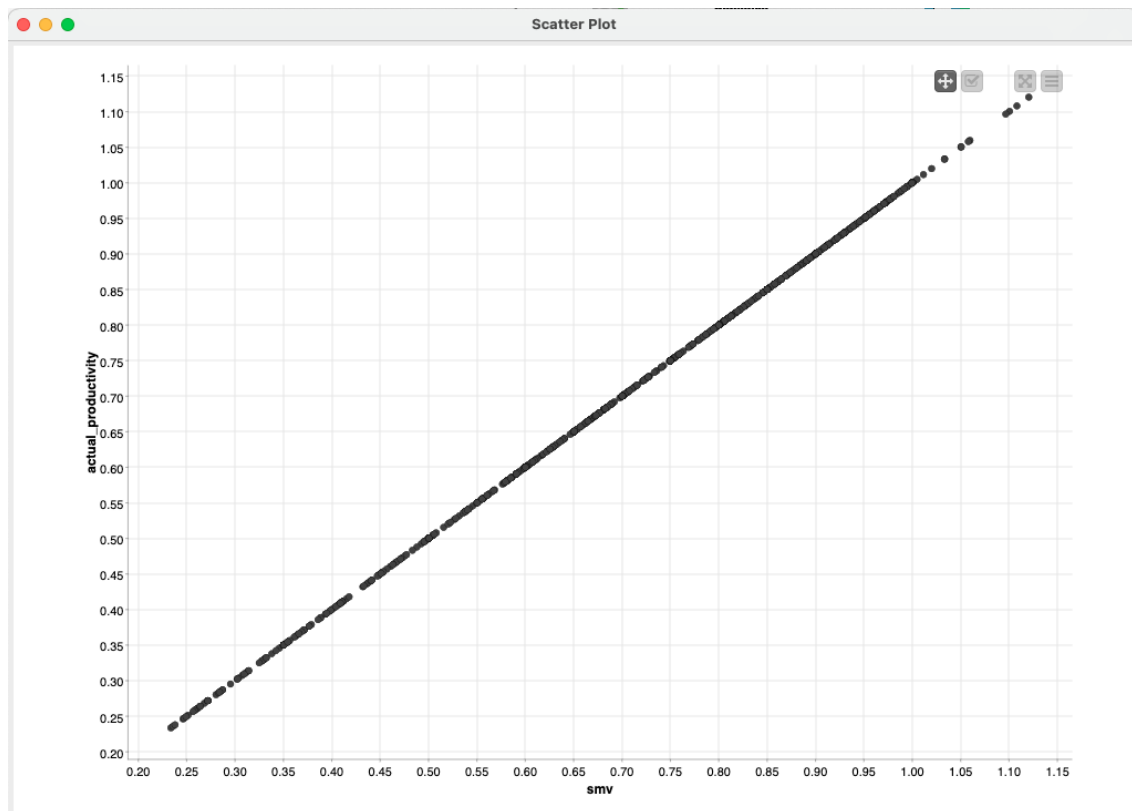


Figura 17: Relação entre *actual\_productivity* e *smv*

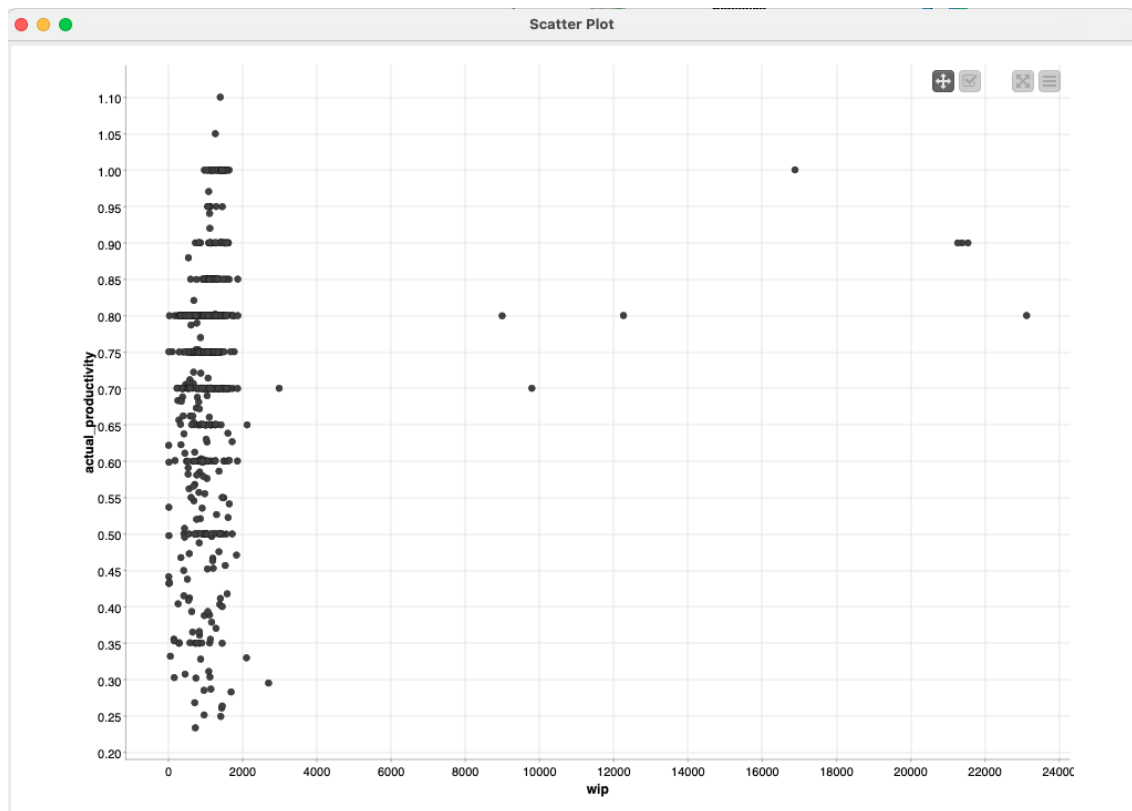


Figura 18: Relação entre *actual\_productivity* e *wip*

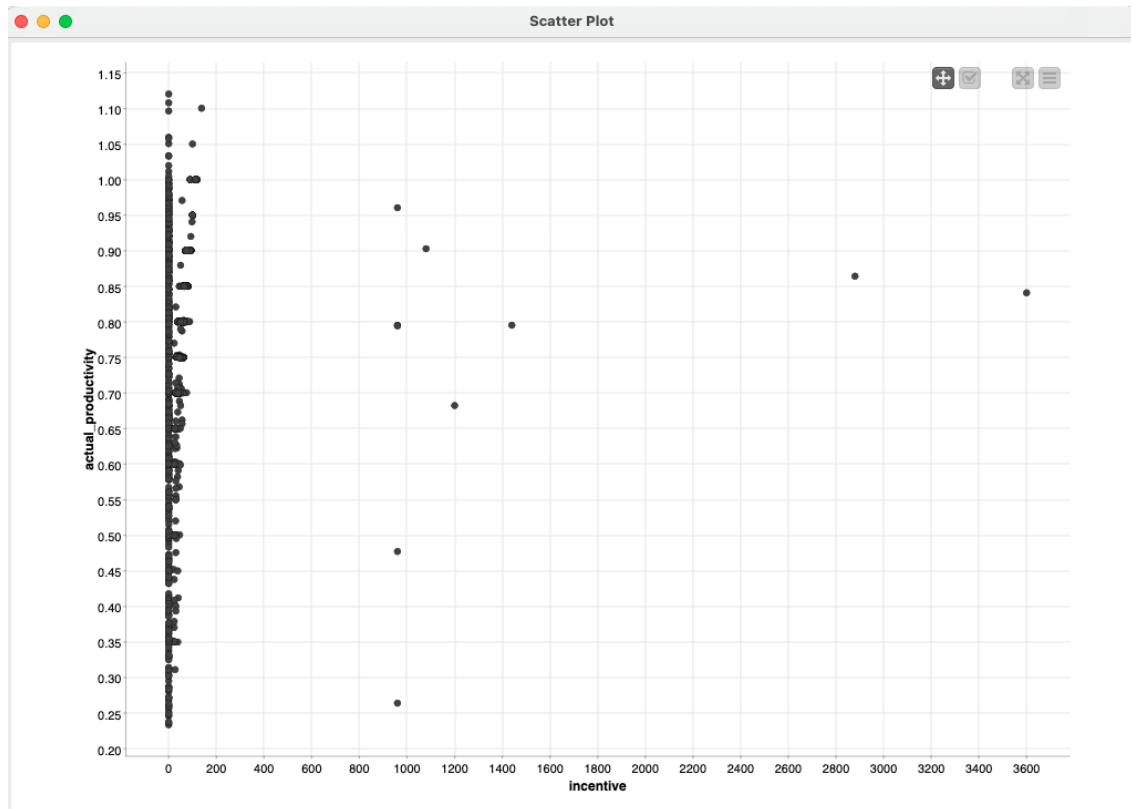


Figura 19: Relação entre *actual\_productivity* e *incentive*

No primeiro diagrama, podemos observar que, quanto maior é o valor do campo *targeted\_productivity*, mais alto é o valor da *actual\_productivity* na área onde apresenta maior concentração de pontos, sendo que essa área se situa á volta do valor da *targeted\_productivity*. Já no segundo gráfico, podemos ver que é mais ou menos traçada uma função linear de declive positivo, o que nos mostra que quanto maior o smv (*standart medium value*) maior a *actual\_productivity*. No quarto diagrama podemos observar que a maioria dos incentivos têm um valor de aproximadamente 0. Mas, quando este valor aumenta, podemos reparar numa reta a ser formada, em que quanto maior o *incentive*, maior a *actual\_productivity*.

Tendo isto em conta, podemos avançar para a parte de preparação de dados.

## Preparação de Dados

Para começar, depara-mo-nos com um problema. Todos os valores que, na teoria, seriam do tipo *Double*, estavam em formato *String*. Então aplicamos o nodo "*String to Number*" para os converter para números, mas não foi suficiente, visto que os separadores decimais eram virgulas em vez de pontos. Então tivemos de recorrer ao nodo "*Column Expressions*" antes do "*String to Number*". Estes nodos apresentam as seguintes configurações:

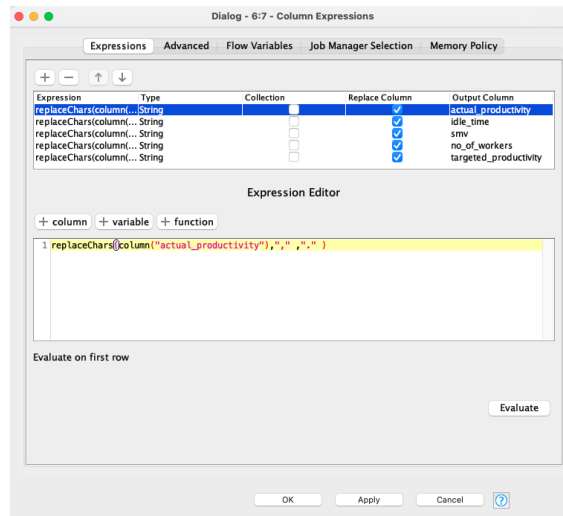


Figura 20: Configurações do nodo *Column Expressions*

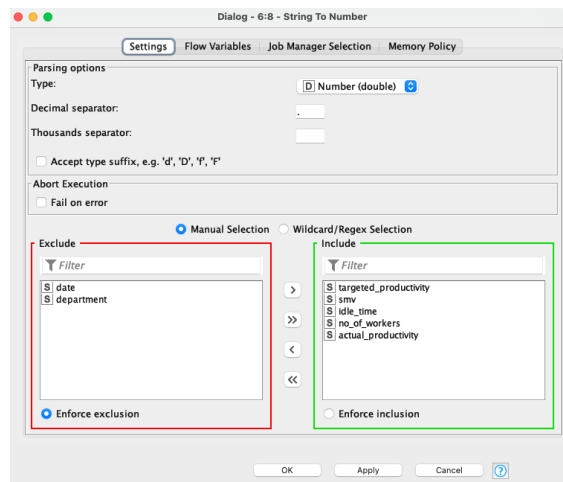


Figura 21: Configurações do nodo *String to Number*

Numa segunda análise, reparamos que a coluna *wip* continha valores NaN, ou *Not a Number*. Para retirar esses valores, que mais tarde viriam a dar problemas, adicionamos uma nova expressão ao nodo *Column Expressions* e, como essa coluna, diferente das outras, era um Double, antes do nodo *Column Expressions* adicionamos um nodo *Number to String*. Na nova expressão no nodo *Column Expressions*, adicionamos uma expressão igual às restantes e, de forma a tratar dos NaN's, substituímos os mesmos por *missing values*. As próximas imagens mostram como ficaram ambos os nodos.

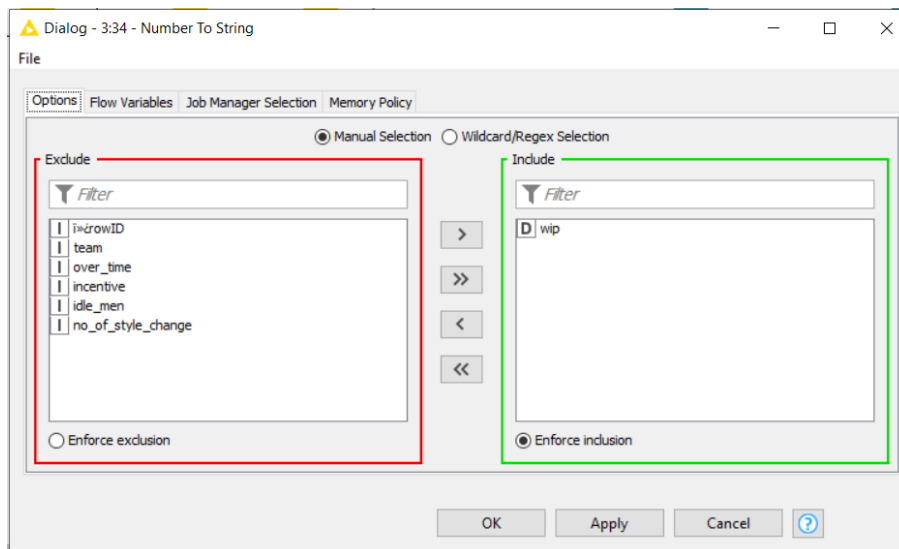


Figura 22: Nodo *Number to String*

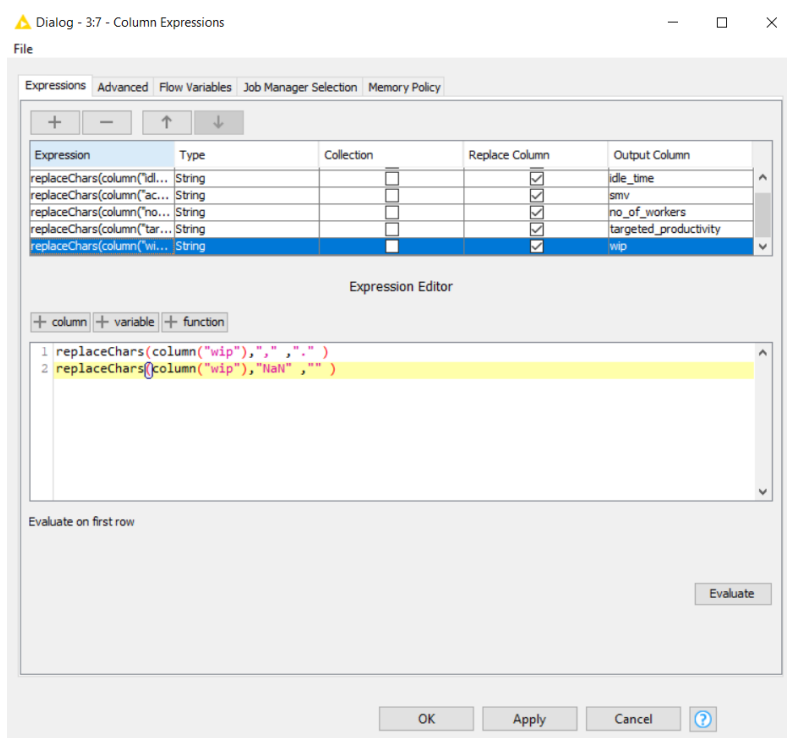


Figura 23: Nova expressão do nodo *Column Expressions*

Em relação aos valores em falta (*missing values*), devido ao tratamento de dados aplicado na coluna *wip*, tivemos de adicionar um nodo que tratasse deles. Neste nodo, apenas serão tratados valores do tipo Double, sendo que a técnica de substituição será pela média, pois remover as linhas tiraria muito conhecimento na base de conhecimento, esta que já tem um tamanho pequeno.

Como verificado na análise anterior aos nodos *Scatter Plot*, onde verificamos a existência de outliers, decidimos aplicar o nodo *Numeric Outliers* de forma a melhorar a precisão do modelo.

## Modelação

Findada a etapa de preparação dos dados, passamos á fase de realização de modelos de aprendizagem, onde iremos tentar atingir o objetivo proposto anteriormente, o de fazer a previsão da *actual\_productivity*, com base nos atributos que seleccionamos como mais relevantes após a nossa análise.

Como se trata de um problema de regressão, visto que o atributo *actual\_productivity* é um atributo não discreto, iremos recorrer a uma rede neuronal, através dos nodos "*RProp MLP Learner*" e "*MultiLayer Perceptron Predictor*". Como queremos prever uma variável numérica, teremos de recorrer a uma aprendizagem supervisionada, logo teremos de incluir a variável que queremos prever no nodo *Column Filter*, bem como as que decidimos dar destaque no estudo dos dados.

Uma rede neuronal é um tipo de modelo de *machine learning* que é concebida com base num modelo simplificado do sistema nervoso central dos seres humanos. É um sistema conexionista e é definida por uma estrutura interligada de unidades computacionais, designadas neurónios, com capacidade de aprendizagem.

### Modelo de rede neuronal

Esta foi a rede neuronal realizada:

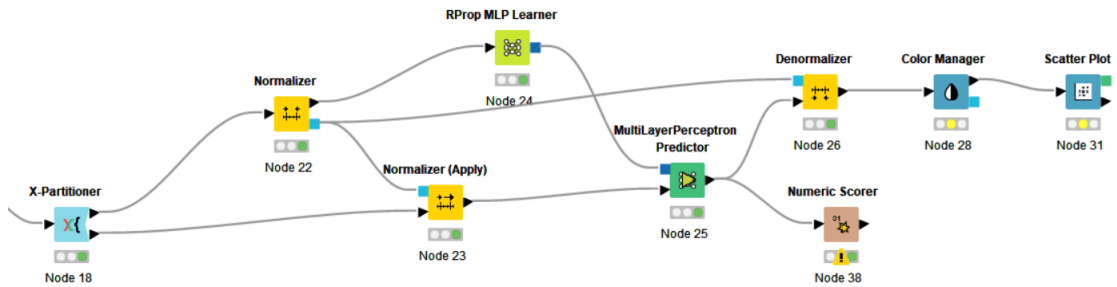


Figura 24: Modelo completo (sem os nodos de tratamento de dados)

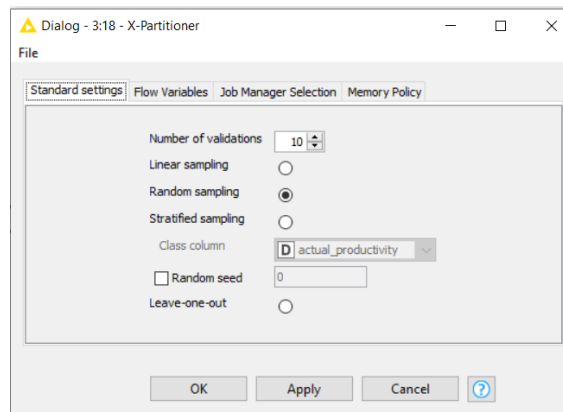


Figura 25: X-Partitioner

Como o *dataset* é relativamente pequeno, decidimos utilizar 10 validações.

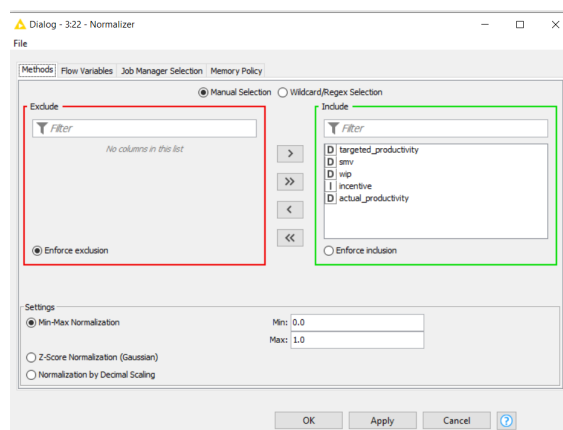


Figura 26: Normalizer

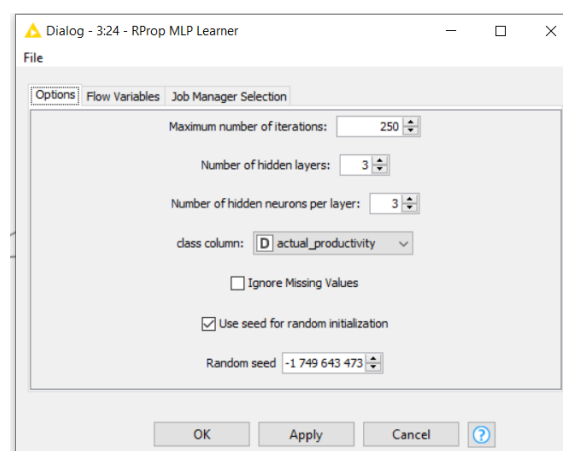


Figura 27: Learner

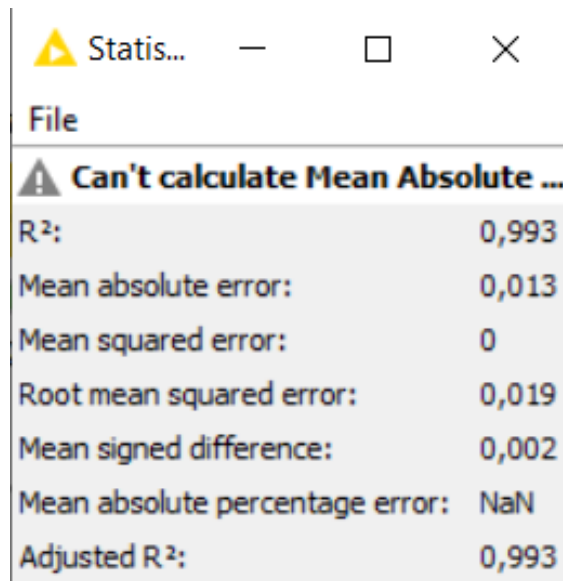


Figura 28: Numeric Scorer

## Avaliação

Analisando os resultados obtidos pelo nodo *Numeric Scorer*, podemos concluir que a previsão foi decente, visto que o valor do  $R^2$  está relativamente próximo de 1, e que os restantes valores providenciados pelo nodo são bastante próximos de 0. Também podemos concluir que o tratamento de dados que efetuamos foi decente e o método que escolhemos para remover *missing values* foi uma boa escolha. O resultado desta previsão ser decente desta forma deve-se provavelmente ao facto de termos utilizado o valor da média para substituir os *missing values* o que resultou nos valores previstos tenderem para esse valor mais frequentemente.

## Sugestões e Recomendações

Quanto a este modelo, para melhorar os resultados poderíamos eventualmente ter separado o atributo "data" em dia, mês, ano, hora e minuto e, a partir daí, fazer a análise dos resultados, de forma a ver se algum destes apresenta-se relevância para a previsão do atributo pretendido. Também poderíamos ter aplicado nodos de regressão linear de forma a testar se a sua precisão seria superior ao da rede neuronal.