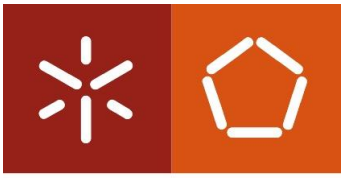


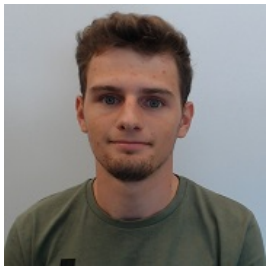
Universidade do Minho

Licenciatura em Engenharia Informática

Aprendizagem e Decisões Inteligentes

Conceção de modelos de aprendizagem.

Grupo 39



Bohdan Malanka
a93300

Diogo Rebelo
a93278

Henrique Alvelos
a93316

Lídia Sousa
a93205

23 de setembro de 2022

Conteúdo

1 Introdução

Este relatório é alusivo ao projeto prático desenvolvido com recurso à ferramenta KNIME, no âmbito da Unidade Curricular de Aprendizagem e Decisões Inteligentes que integra a Licenciatura em Engenharia Informática da Universidade do Minho.

Este projeto consiste na **análise e extração de conhecimento** atendendo à distribuição de dados disponíveis, e o respetivo **desenvolvimento de modelos de aprendizagem**.

Este projeto encontra-se dividido em duas fases de trabalho de forma a simplificar e fomentar a organização do trabalho, contribuindo para a sua melhor compreensão. Pretende-se, assim, que o relatório sirva de suporte ao trabalho realizado para esta fase, mais propriamente, dando uma explicação e elucidando o conjunto de decisões tomadas ao longo da construção de todo o projeto.

1.1 Estrutura do Relatório

De acordo com os requisitos do enunciado definimos uma estrutura para o relatório de forma a facilitar a compreensão do mesmo. O relatório está dividido em **duas partes**: uma parte para o *dataset* proposto pela equipa docente e outra para o *dataset* selecionado pelo grupo de trabalho.

Cada uma destas partes contém, inicialmente, uma breve explicação daquilo que consiste numa primeira análise do *dataset*, aspetos que achamos que poderão ser relevantes para o trabalho prático. De seguida, e correspondendo à **1.º fase deste trabalho**, a explicação e acompanhamento daquilo que foi toda a preparação de dados do *dataset*. Por fim, uma secção correspondente à **2.º fase do trabalho** que dividimos de forma a descrever os modelos desenvolvidos e detalhes dos mesmos e uma parte em que procedemos à análise e interpretação de resultados, bem como, sugestões e recomendações pós-análise.

1.2 Metodologia de trabalho

Neste projeto, em grupo, definimos inicialmente alguns prazos de forma a que fosse mais simples dividir tarefas e não deixar acumular trabalho. Portanto, decidimos inicialmente explorar o *dataset* proposto procedendo à preparação de dados do mesmo. De seguida começamos por explorar alguns *datasets*. Nesta fase foi quando optamos por procurar *datasets* de regressão. Quando este estava selecionado tratamos da preparação dos dados.

Mais à frente fomos explorando e testando vários modelos de aprendizagem, tendo em mente que seria necessário testar bastante e usar diferentes metodologias de

forma a encontrar aquela que se aproximaria mais da solução correta.

Algo que gostaríamos de ter feito de forma mais sistemática e cuidadosa era acompanhar todos os passos, principalmente da preparação de dados, com exemplos e capturas de forma a permitir acompanhar melhor todo o processo.

De forma a explorar o KNIME de forma mais eficiente, ao longo do trabalho tiramos partido do **KNIME Hub**, que consiste num fórum com explicações acerca dos nodos bem como *workflows* exemplo e algumas dicas. De forma análoga, fomos realizando pesquisa em diversos *websites* que consideramos que continham informação relevante para um melhor aproveitamento neste trabalho prático.

2 *Dataset* Proposto

2.1 Análise do *dataset*

O *dataset* proposto pela equipa docente é um *dataset de classificação*, cujo objetivo é prever o salário de um cliente (*salary*) - cujas possibilidades são >50k ou <=50k.

O dataset consiste em 32560 observações e 15 *features* (6 numéricas e 9 categóricas) (Figura 1.).

age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	salary
39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50K
50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50K
38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=50K
53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	<=50K
28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba	<=50K
37	Private	284582	Masters	14	Married-civ-spouse	Exec-managerial	Wife	White	Female	0	0	40	United-States	<=50K
49	Private	160187	9th	5	Married-spouse-absent	Other-service	Not-in-family	Black	Female	0	0	16	Jamaica	<=50K
52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	45	United-States	>50K
31	Private	45781	Masters	14	Never-married	Prof-specialty	Not-in-family	White	Female	14084	0	50	United-States	>50K

Figura 2.1: *Dataset: salary classification*

2.1.1 *Features*

- **age**: contínua
- **Workclass**: *Private*, *Self-emp-not-inc*, *Self-emp-inc*, *Federal-gov*, *Local-gov*, *State-gov*, *Without-pay*, *Never-worked.*
- **fnlwgt**: contínua
- **education**: *Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.*
- **education-num**: contínua
- **marital-status**: *Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.*
- **occupation**: *Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces.*
- **relationship**: *Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.*

- **race**: *White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.*
- **sex**: *Female, Male.*
- **capital-gain**: contínua
- **capital-loss**: contínua
- **hours-per-week**: contínua
- **native-country**: *United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, TrinadadTobago, Peru, Hong, Holand-Netherlands.*

2.1.2 Conclusões acerca do *dataset*

Analisando o *dataset*, nas colunas categóricas existiam cerca de 6000 valores em falta que, ao invés de estarem vazios estavam preenchidos com um ponto de interrogação portanto terão de ser tratados na preparação de dados.

Recorrerremos no entanto a *data visualization*, ou seja, à representação dos dados em gráficos como por exemplo *plots*, permitindo que relações de dados complexas se tornem percepções orientadas por dados de uma forma que é fácil de compreender.

Para proceder à *data visualization* recorremos a nodos como por exemplo ***Statistics***, ***Data Explorer***, ***Box Plot***, entre outros.

Com o **Box Plot** pretendemos verificar possíveis desvios e existência de *outliers*.

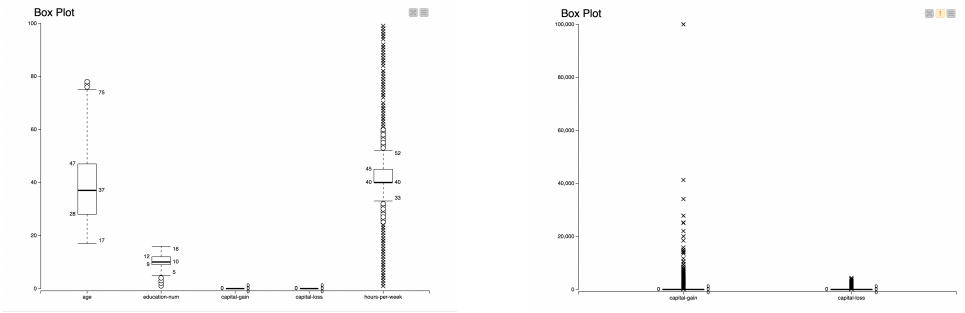


Figura 2.2: Box Plot

Com este gráfico concluímos que, sendo a mediana representada pela linha na caixa, percebemos que no caso das *hours per week* e do *education num* há um desvio, ou seja, que a maioria dos dados estão localizados no lado alto ou baixo do gráfico. Identificamos também a existência de outliers, que são valores de dados que estão muito longe de outros valores de dados. No caso do *capital gain* e *capital loss* surgem muitos *outliers* porque dado o contexto das *features* estes raramente estão diferentes de 0, sendo que os que estão são considerados *outliers*.

Recorrendo ao ***Data Explorer*** verificamos novamente a distribuição de dados, possíveis dados assimétricos, existência de *outliers* ou dados multimodais.

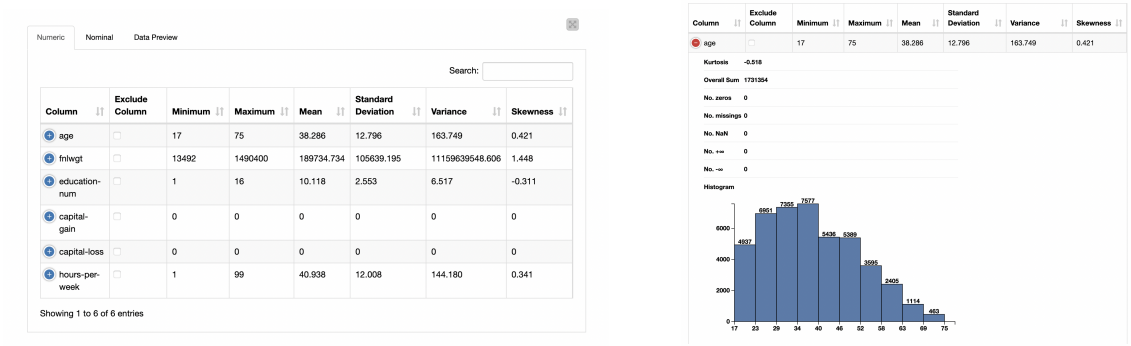


Figura 2.3: Data Explorer

No caso de *education num* verificamos que há uma má distribuição dos dados, havendo assimetria, sendo a maioria dos mesmos localizados no lado superior do gráfico. No entanto, dada a nossa interpretação isto tem significado no *dataset*, representando uma população mais formada e educada (dado que 9-13 anos de escolaridade representam escola secundária concluída e ensino universitário iniciado/-concluído)

Quanto à *feature fnlwgt* reparamos que esta não faria qualquer sentido, que funciona aparentemente quase como um ID, único para cada entrada e por tal, com o nodo **Column Filter** pretendemos remover para que este não seja considerado.

Usamos nodos de correlação (como por exemplo **Rank Correlation**) para procurar testar algo que nos chamou a atenção mas não funcionou dado que é uma correlação entre coluna categórica e númerica. As *features education-num* e *education* têm a mesma informação (nível de escolaridade que conseguiram e o número de anos que demorou a alcançar esse nível de escolaridade). Recorremos ao nodo **Crosstab** e **Statistics** para provar então a correlação que identificamos:

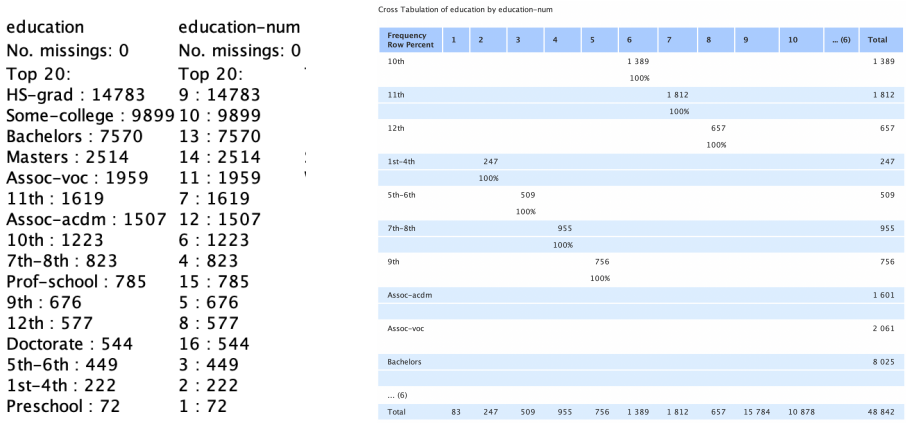


Figura 2.4: Correlação entre *education-num e education*

Como podemos observar, o número de ocorrências de 9 e HS-grad são iguais, sendo que HS-grad, por exemplo, corresponde a 9 anos de escolaridade no *dataset*.

Recorrendo a uma pesquisa intensiva acabamos por encontrar alguns gráficos que foram relevantes para a análise do nosso *dataset*:

Através da análise destes gráficos concluímos que há um **desbalanceamento dos**

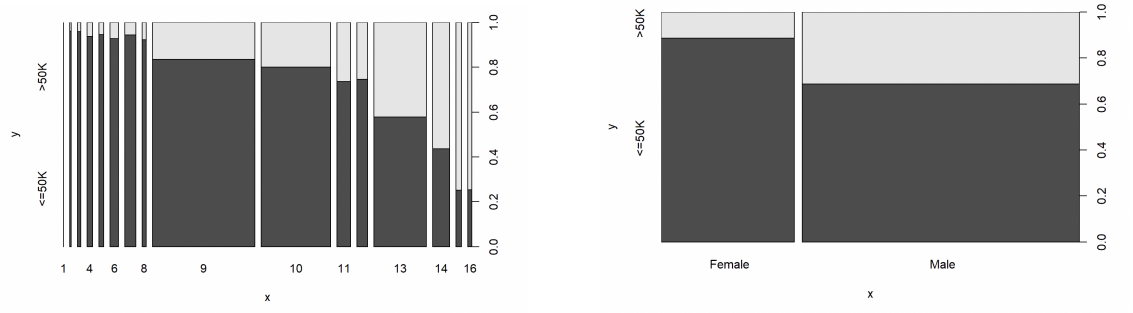


Figura 2.5: Gráficos - *Education-num / Sex*

dados, havendo muito mais ocorrências de <=50k do que >50k. Por outro lado, interpertando primeiramente a influência do *education-num* no salário percebemos que a maior ocorrência de salários ">50k" corresponde a uma maior escolaridade. Quanto ao segundo gráfico percebemos que há mais ocorrências de salários ">50k" no caso do sexo masculino do que no feminino.

De forma a corrigir os erros e interpretar as inferências a que chegamos, trataremos os dados porque um modelo de aprendizagem só é eficiente e fidedigno com *clean data*.

2.2 Preparação dos dados

A **preparação de dados** consiste em tratar *outliers*, células vazias ou *missing values*, possíveis erros, dados redundantes ou irrelevantes e adequar os dados.

A figura seguinte corresponde a uma preparação de dados numa fase inicial, ainda não muito eficiente, quando estávamos ainda a testar e a avaliar todo o processo.

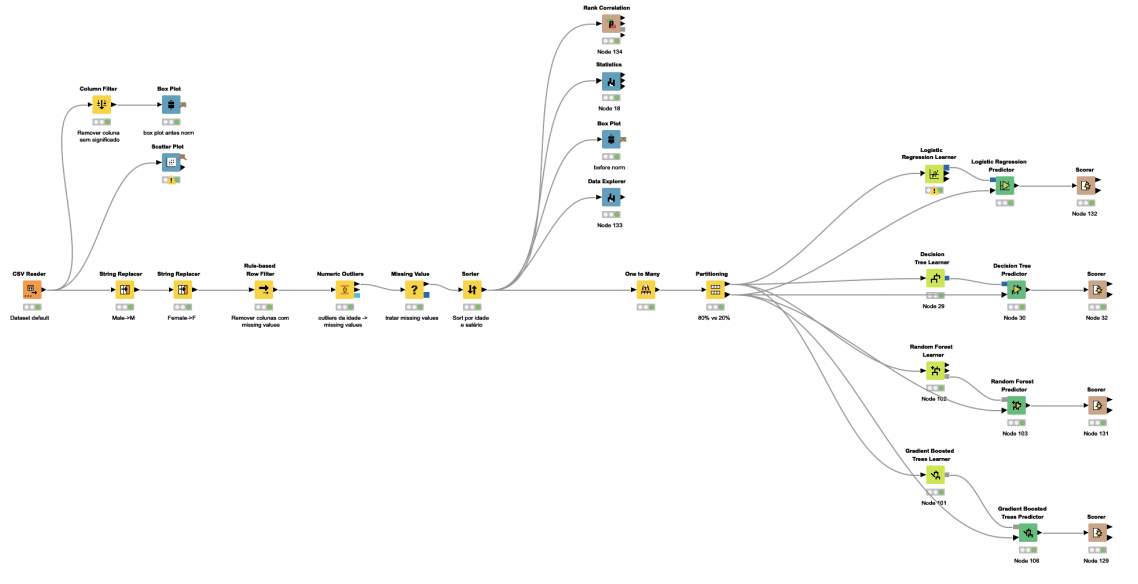


Figura 2.6: *Preparação de dados numa fase inicial*

No entanto, com o avanço do trabalho prático e à medida que fomos adquirindo mais conhecimento percebemos que poderíamos melhor em vários aspetos a nossa preparação de dados bem como explorar algumas opções:

- Substituir *Missing Values* ao invés de remover
- Corrigir o desbalanceamento dos dados
- Testar *feature selection*
- Otimização para cada modelo

A preparação de dados a que procedemos nesta fase inicial e a evolução da mesma será explicada de forma detalhada de seguida:

String Replacer

De modo a simplificar a complexidade do *dataset*, recorremos ao nodo **String Replacer** para substituir na feature *sex*, as strings *Female* e *Male* por F e M, respetivamente.

Tratamento de Missing Values

De seguida, nas *features* categóricas - *occupation*, *native country* e *workclass* - encontramos um conjunto de valores (cerca de 6000) que, como referimos acima estavam preenchidos com um ponto de interrogação. Inicialmente, usamos o nodo **Rule-based Row Filter** para remover todas as linhas com ocorrências do "?" nas *features* acima mencionadas. Por outro lado, de forma a serem tratados pelo nodo **Missing Values**, testamos também tratar os mesmos usando o nodo **String Manipulation**. Neste nodo, usamos a seguinte configuração modelo para atingir o objetivo mencionado acima:

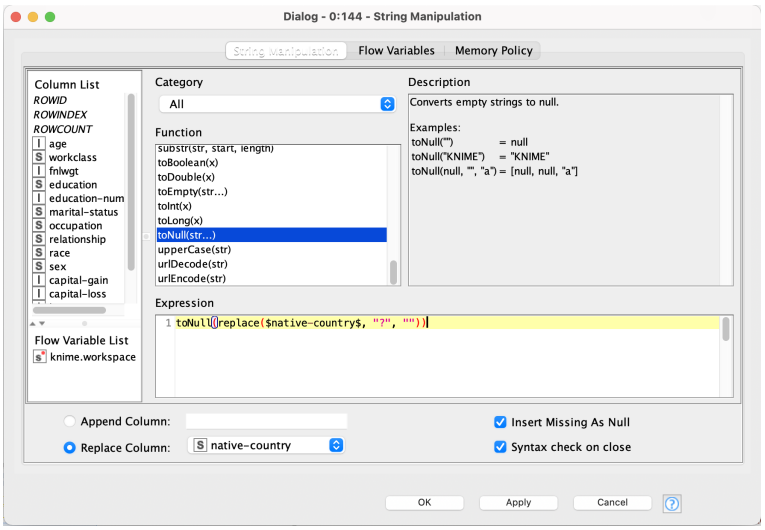


Figura 2.7: String Manipulator - *native country*

No nodo **Missing Values**, para o caso das *Strings* usamos o valor mais frequente e para o caso de números usamos a mediana porque como consideramos os *outliers* da idade *missing values* não poderíamos considerar a média - resultaria em valores sem significado.

A abordagem que acabamos por selecionar no caso dos ***missing values*** teve como motivação a quantidade de dados em falta (cerca de 6000) que não faria sentido substituir os mesmos porque poderíamos estar a enviesar os dados e ao remover os mesmos temos observações suficientes para resultar numa análise fiável.

Outliers

Quanto ao tratamento dos ***outliers***, identificados na análise do *dataset* consideramos que os outliers continham conhecimento importante. No entanto, no processo de descobrimento testamos tratar todos os *outliers*, mas decidimos tratar apenas a idade.

No caso das *hours per week* faz sentido aos nossos olhos que alguém que trabalhe mais horas semanalmente tenha um rendimento superior a alguém que trabalhe menos e portanto, os outliers desta *feature* trazem maior conhecimento. Por outro lado, como referido na secção de análise, os *outliers* do *capital gain* e *capital loss* correspondem aos valores diferentes de 0, ou seja, são valores com conhecimento.

No caso do *education num* analisamos acima a relação que esta tinha com o *target* e portanto, são, na nossa opinião, valores que podem ser muito informativos para nós. Por exemplo, se todos os casos com escolaridade abaixo da média tiverem um salário "<50k" para nós isto é conhecimento importante.

Portanto, a nossa decisão foi manter estes *outliers* porque podem conter informação importante. Remover estes valores pode distorcer os dados relativamente à variabilidade inerente à nossa àrea de estudo: salarial. Ou seja, poderíamos estar a forçar a que esta parecesse menos variável do que aquilo que é de facto.

Desbalanceamento de dados

Relativamente ao **tratamento do desbalanceamento dos dados**, ou seja, temos cerca de 37000 observações de "<=50" e cerca de 11000 de ">50k". Algoritmos de classificação como *Decision Tree Learner* e *Logistic Regression* vão ter enviesamento, porque tendem a prever os dados da maioria. As características da classe minoritária (neste caso ">50k") como ruído. Desta forma, há uma probabilidade muito grande de a classificação estar errada da classe minoritária em comparação com a classe maioritária.

No processo de decisão surgiram 2 possibilidades de resolver este problema: **Random Under Sampling** ou **Synthetic minority oversampling technique**, que corresponde ao SMOTE. No caso do *Random Under Sampling* este visa equilibrar a distribuição de classes através da eliminicação aleatória de exemplos da classe maioritária. Isto poderia ser vantajoso para melhorar o tempo de execução e o custo computacional. No entanto poderia descartar informação potencialmente útil. No caso do SMOTE, é uma técnica estatística para aumentar o número de casos no seu conjunto de dados de uma forma equilibrada, gerando novas entradas a partir de casos minoritários existentes.

No entanto, no contexto do KNIME, recorremos ao SMOTE. O SMOTE fez com que a *accuracy* dos nossos modelos. No entanto, a *accuracy* não é uma boa medida de desempenho em *datasets* desbalanceados. Como tal vamos interpretar a matriz de confusão no caso de, por exemplo, *Logistic Regression* antes e depois do balanceamento dos dados.

Row ID	<=50K	>50K
<=50K	6900	530
>50K	929	1410

Row ID	<=50K	>50K
<=50K	5986	1444
>50K	376	1963

Figura 2.8: *Confusion Matrix* - Sem balanceamento de dados / Com balanceamento de dados

Resumidamente, ao atribuir maior peso à classe mais pequena tornamos o modelo mais tendencioso para ela. Ou seja, o modelo vai prever agora esta classe com maior precisão, mas a precisão global vai diminuir. Como tal, na nossa opinião, faz sentido manter a utilização do SMOTE.

Relativamente ao SMOTE, na sua configuração optamos por selecionar *Oversample Minority Classes* por ser a opção que acrescenta exemplos sintéticos na classe minoritária. A saída contém o mesmo número de filas para cada uma das classes possíveis. Contrariamente, o *Oversample by* corresponde a especificar a quantidade

de dados sintéticos a introduzir. Ou seja, ao selecionar o valor de 2 estamos a introduzir mais 2 porções em cada classe (para 50 linhas na tabela de entrada como "<=50k" a saída irá conter 150 linhas desta classe).

Column Filter

Como referido na secção anterior, nesta fase inicial removemos a coluna *fnlwgt* dado que esta não possui nenhuma informação adicional, no entanto, é um tratamento a que vamos proceder ao longo de todo o processo e nos variados *workflows* porque esta coluna não adiciona efetivamente conhecimento nem informação.

Partitioning

Para a separação do *dataset* em treino e teste, usamos o nodo ***Partitioning***. No projeto testamos *stratified sampling* e *random sampling* pelos seguintes motivos:

- Quanto ao *random sampling*, que é a forma mais simples de criar o dataset de testes este escolhe parte dos dados aleatoriamente. No entanto este método é bom se o dataset for suficientemente grande, caso contrário podemos estar a enviesar os dados.
- Quanto ao *stratified sampling*, reduz o erro no caso em que a população pode ser dividida em subgrupos. Assim, o dataset de teste é representativo da população.

Optamos pelo *stratified sampling* porque pode fornecer uma representação mais precisa da população com base no que é usado para a dividir em diferentes subconjuntos e porque produz estimativas mais precisas dos grupos, colocando indivíduos semelhantes nos grupos. Pretendemos que, neste caso em que a população tem diversos subgrupos (*race*, *sex*, etc.) a amostra inclua todos eles.

Com o avanço no projeto e com toda a pesquisa que este implicou acabamos por perceber que poderíamos explorar melhor cada um dos modelos de aprendizagem. Passaremos a explicar então aspetos de cada um dos modelos que se tornaram relevantes para as decisões tomadas numa fase mais avançada deste trabalho prático.

- **Decision Tree Learner:** Uma árvore de decisão é essencialmente uma série de declarações condicionais que determinam o caminho que uma amostra toma até chegar ao fundo. No caso deste modelo temos a vantagem da preparação de dados ser mais simples (os atributos utilizados para a tomada de decisões podem ser tanto nominais como numéricos). e dos valores em falta e *outliers* terem menos significado. Em termos de desvantagens, árvores de decisão são mais instáveis em comparação com outros modelos e, apesar de requer poucos cálculos, reduzindo assim o tempo de implementação, têm uma menor precisão.
- **Random Forest Learner:** Este modelo consiste em muitas árvores de decisão combinadas para obter um resultado mais preciso em comparação com uma única árvore. O custo computacional é por isso maior e o processo de geração e análise é mais demorado. No entanto este modelo resulta bastante

bem em *datasets* grandes e é bastante robusto no que diz respeito a *outliers*. Outra vantagem é a sua capacidade de lidar com valores categóricos sem ter de os transformar primeiro (por exemplo, utilizando técnicas de *feature engineering*).

- **Logistic Regression:** *Logistic Regression* é um método de análise estatística para prever um resultado binário com base em observações prévias de um conjunto de dados. Este modelo prevê uma variável de dados dependente através da análise da relação entre uma ou mais variáveis independentes existentes, utilizando um modelo linear - pelo que sofre dos mesmos problemas que a regressão linear. A preparação de dados neste caso é mais trabalhosa sendo necessário tratamento de *outliers* e *missing values*, remover *features* correlacionadas, entre outros. Este algoritmo assume que existe uma relação linear entre as variáveis de entrada com a saída pelo que a normalização é importante.

- **Gradient Boosted:** Normalmente uma precisão nas suas previsões dificilmente ultrapassada e a preparação de dados é normalmente simples - dado que funciona com valores categóricos e numéricos e lida com os *missing values*. Como requer normalmente muitas árvores (por vezes mais de 1000), pode ser exaustivo em termos de tempo e memória computacional. Por outro lado, este modelo é mais díficil de interpretar e a sua alta flexibilidade resulta em muitos parâmetros que interagem e influenciam muito o comportamento da abordagem (número de iterações, profundidade das árvores, parâmetros de regularização). Logo, este modelo requer uma pesquisa exaustiva sobre estes valores específicos.

Analisando o que cada um avaliava e como avaliava percebemos que faria sentido fazer uma preparação para cada modelo de aprendizagem que passaremos a explicar mais à frente. No entanto, tivemos por base a preparação de dados explicitada anteriormente dado que consideramos que esta estava adequada, especificando de acordo com as necessidades de cada modelo.

Como tal decidimos proceder a uma preparação de dados para o *Logistic Regression*.

Procuramos escolher aprofundar esta preparação tendo por base nodos e algoritmos que ainda não teríamos trabalhado ao longo deste trabalho. O *Logistic Regression* foi o primeiro a ser escolhido dado que, segundo a nossa pesquisa, este é o modelo que tem a preparação de dados mais específica.

2.2.1 Preparação de dados - *Logistic Regression*

Para proceder à preparação de dados para este modelo vamos proceder à normalização, conversão, ao tratamento de *missing values* e tratamento do desbalanceamento. Como referido acima, tendo por base parte da preparação de dados geral.

Conversão

Logistic Regression funciona, no geral, com atributos numéricos. Neste caso vamos tratar da conversão das categóricas. Para converter colunas nominais em uma ou

mais colunas numéricas podemos optar por *one hot enconding* ou *index encoding*.

No caso do *index encoding* este transforma cada valor nominal num número. No entanto, introduz uma distância numérica artificial entre dois valores devido à função de mapeamento. Para o *one hot encoding* cada valor existente cria uma coluna desse valor, que vão ser preenchidas com 1 ou 0. Este algoritmo gera muitas colunas a partir de uma coluna original, aumentando assim a dimensionalidade do conjunto de dados.

Para testar construímos 2 *workflows* com o mesmo tratamento de dados para testar a influência de cada um na precisão e rapidez de execução visto que a decisão final é um equilíbrio entre estas 2 variáveis.

Como o *one hot encoding* faz com que, pelo aumento da dimensionalidade, o tratamento do desbalanceamento seja muito mais demorado, na nossa opinião, o *index encoding* em termos de relação *accuracy-speed* é mais vantajoso.

Normalização

Relativamente à normalização, este modelo precisa que os dados estejam normalizados no intervalo [0,1] (preferencialmente em Z-Score Normalization - normalização gaussiana) isto porque as variáveis com intervalos maiores afetam o cálculo das variações e distâncias e podem acabar por comprometer a precisão do algoritmo.

Como tal, em ambos os casos usamos a normalização gaussiana.

No caso do *one hot encoding* optamos por normalizar as *features age, capital gain e loss, hours per week* e *education num* visto que as restantes correspondiam a colunas preenchidas com 0 e 1.

Tratamento de missing values

Relativamente ao tratamento de missing values, dado que o *logistic regression* não trata os mesmos, optamos por manter a abordagem explicada anteriormente.

2.2.2 Desbalanceamento de dados

O balanceamento dos dados, a sua demonstração e o seu tratamento já foi explicado acima, no entanto achamos que poderia ser interessante testar o *Equal-Size Sampling* - O nó removendo linhas aleatórias pertencentes à classe maioritária. As linhas devolvidas por este nó conterão todos os registos da classe minoritária e uma amostra aleatória da classe maioritária.

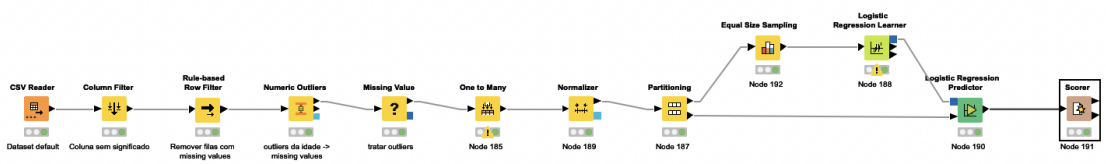


Figura 2.9: *Equal size sampling*

Relativamente aos resultados, tal como esperávamos este nodo funciona de forma muito mais rápida, dado que o custo computacional é obviamente menor, mesmo no caso do *one hot encoding*. No entanto, em termos de *accuracy* esta é mais flutuante dado que a remoção de dados é aleatória, pondendo remover informação importante, mas não foi nas experiências que fizemos substancialmente menor do que com o uso do SMOTE.

2.3 Análise crítica de resultados

Tal como esperávamos, a preparação de dados foi a parte a que precisamos de dedicar mais tempo, sendo uma das mais importantes. No entanto, ainda que com os nossos desafios e dúvidas tentamos ter uma aborgadem prática e eficiente (com baixo custo computacional) que resultasse num *dataset* com dados úteis e coerentes.

Consideramos que poderíamos ter explorado *feature selection*, no entanto, acabamos por não explorar este algoritmo porque implicaria dispender de tempo para explorar outro modelo, o *Naive Bayes*, porque árvores de decisão tratam de *feature selection* até um certo ponto.

Ou seja, para árvores de decisão, *feature selection* não é tão importante porque durante indução de árvores de decisão, os dados são selecionados com base em métricas como o ganho de informação, portanto, se houver *features* não-informativas, não serão seleccionadas para a decisão.

Poderíamos considerar trabalho futuro implementar este algoritmo de forma a diminuir o custo computacional e o tempo de construção das árvores por exemplo, para o *Random Forest* mas na nossa pesquisa sobre quando seria mais útil recorrer a *feature selection* encontramos o seguinte blog, que *Naive Bayes* não tratam de forma intrínseca a importância das *features*. Como tal, pensamos que seria vantajoso explorar este modelo.

Na nossa opinião exploramos bastantes nodos, bastantes configurações e fizemos um trabalho de pesquisa acerca de algoritmos de classificação muito eficaz.

3 *Dataset* Selecionado

O dataset escolhido no Kaggle foi acerca da popularidade de um *dataset*, tendo por objetivo prever a popularidade tendo em conta determinados fatores. Este *dataset* chamou-nos à atenção nomeadamente pelo tema, porque nos parecia desafiante prever a popularidade de uma música tendo em conta certas características. Por outro lado, tencionávamos testar treinar modelos de regressão de forma a diversificar o trabalho prático.

3.1 Análise do *dataset*

	_name	song_popularity	song_duration_ms	acousticness	danceability	energy	instrumentalness	key	liveness	loudness	audio_mode	speechine	tempo	time_signature	audio_valence
2	levard of Broker	73	262333	0.0055200000	0.496	0.682	2.94e-05	8	0.0589	-4.095	1	0.0294	167.06	4	0.474
3	e End	66	216933	0.0103	0.542	0.853	0.0	3	0.1080000	-6.407	0	0.0498	105.256	4	0.37
4	n Nation Army	76	231733	0.00817	0.737	0.46299	0.447	0	0.255	-7.827999	1	0.0792	123.881	4	0.324
5	he Way	74	216933	0.0264	0.451	0.97	0.00355	0	0.102	-4.938	1	0.107	122.444	4	0.198
6	You Remind M	56	223826	0.0009539999	0.447	0.76599	0.0	10	0.113	-5.065	1	0.0313	172.011	4	0.574
7	g Me To Life	80	235893	0.00895	0.316	0.945	1.85e-06	4	0.396	-3.168999	0	0.124	189.930	4	0.32
8	Resort	81	199893	0.000504	0.581	0.887	0.00110999999999	4	0.268	-3.659	0	0.0624	90.5779	4	0.7240000000
9	You Gonna Be M	76	213800	0.00148	0.613	0.953	0.000582	2	0.152	-3.435	1	0.0855	105.046	4	0.537
10	Brightside	80	222586	0.00108	0.33	0.93599	0.0	1	0.0926	-3.66	1	0.0917	148.112	4	0.2339999999
11	on Fire	81	203346	0.00172	0.542	0.905	0.0104	9	0.136	-5.653	1	0.0540000	153.398	4	0.374
12	Middle	78	168253	0.0424	0.629	0.897	0.0	2	0.263	-3.401000	1	0.0483	161.944	4	0.93
13	ıb	63	185586	0.0046	0.496	0.863	0.0	9	0.639	-4.153000	1	0.0381	110.017	4	0.243
14	oth Criminal	75	209266	0.00434	0.647	0.96400	0.0036	9	0.15	-4.225	0	0.06	126.942	4	0.875
15	t Stop	81	269000	0.0179	0.618	0.938	0.0	9	0.1669999	-3.441999	1	0.0456	91.455	4	0.875
16	p Suey!	69	210240	0.0003529999	0.42	0.929	0.00074699999999	7	0.122	-3.898999	0	0.121	127.204	4	0.3
17	Me Out	77	237026	0.0004230000	0.278	0.67599	0.00089900000000	9	0.136	-8.821	1	0.0371	104.545	4	0.494
18	s You	71	227240	0.0013599999	0.659	0.778	6.79e-06	11	0.0841	-6.422999	1	0.0379	110.022	4	0.623
19	of You	62	256600	0.00701	0.37	0.94400	2.89999999999999	1	0.135	-4.979	0	0.0767	130.315	4	0.345
20	te Sins Not Trag	77	187613	0.0938	0.5670000000	0.795	0.0	9	0.114	-4.985	0	0.134	170.06	4	0.635
21	tonite	79	233933	0.0066400000	0.545	0.865	1.12e-05	11	0.168	-5.707999	0	0.0286	99.01	4	0.5429999999

Figura 3.1: *Song Popularity Dataset*

3.1.1 *Features*

- *song name*: nome da música
- *popularity*: valores entre 0 e 100
- *durationms*: duração da música em milisegundos; tipicamente valores entre 200k e 300k
- *acousticness*: valores entre 0 e 1
- *danceability*: valores entre 0 e 1
- *energy*: valores entre 0 e 1

- *instrumentalness*: valores entre 0 e 1
- *key*: todas as chaves em oitava codificadas como valores entre 0 e 11, começando com C como 0, C como 1, etc.
- *liveness*: valores entre 0 e 1
- *loudness*: float com valores tipicamente entre -60 e 0
- *mode*: 0 = menor, 1 = maior
- *speechiness*: valores entre 0 e 1
- *tempo*: float com valores tipicamente entre 50 e 150
- *valence*: quão felizes a feliz a música é; valores entre 0 e 1
- *time-signature*: valores entre 1 e 5

3.1.2 Conclusões acerca do *dataset*

Relativamente à análise deste dataset, quanto a *missing values* não estavam presentes mas havia uma grande quantidade de *outliers*, confirmada pela observação do ***Box Plot***:

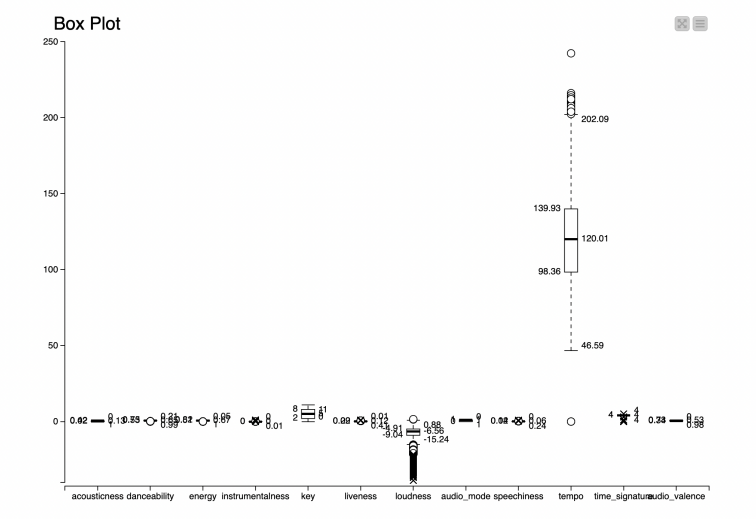


Figura 3.2: *Box plot*

Os dados não estão normalizados, o que podemos perceber pela escala dos gráficos. A *feature* mais evidente em termos de destaque de normalização é o *tempo* e a *song duration* (que não incluimos porque tornava ainda mais inpercetível a análise das restantes *featues*). As ordens de grandeza serão tratadas na preparação de dados de forma a corrigir esta má distribuição dos dados.

Como podemos ver na seguinte imagem, os dados, por exemplo, da duração das músicas tem valores tendenciosos que acabam por fazer com que hajam *skews*.

Ou seja, a duração de uma música tem valores tendenciais. Na preparação de dados vamos recorrer também a *binning* de forma a tentar tornar menos evidente.

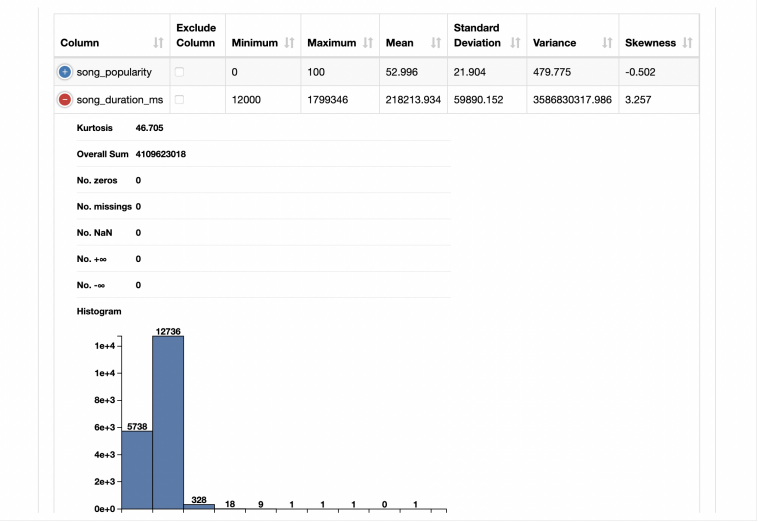


Figura 3.3: *Data Explorer - Song duration ms*

3.2 Preparação dos dados

A explicação da preparação de dados para o modelo de regressão será explicada detalhadamente de seguida:

Column Filter

O nodo ***Column Filter*** serviu para remover a coluna *song name* visto que esta pode ser vista como uma coluna de ID, que não adiciona informação relevante por ser diferente para todas as entradas.

Auto-Binner

Usamos o nodo ***Auto-Binner*** para agrupar os dados numéricos em intervalos - chamados *bins*. Normalmente, *binning* melhora a precisão dos modelos de previsão, reduzindo o ruído ou a não-linearidade no conjunto de dados.

Outliers

Na preparação de dados tratamos os *outliers* como *missing values* e substituindo-os à frente pela mediana no caso de inteiros e pela média no caso de doubles.

Normalização

Como vimos anteriormente, os dados neste *dataset* não estão normalizados, e como tal vamos proceder à sua normalização visto que os modelos de regressão são sensíveis à distribuição dos dados.

Normalizaremos o conjunto de dados para que cada variável esteja no intervalo de 0 a 1. Isto significa que todas as nossas características terão o mesmo peso.

Clustering

De forma a testar novos algoritmos decidimos testar *clustering* que consiste em dividir a população ou pontos de dados em vários grupos, de modo a que os pontos de dados nos mesmos grupos sejam mais semelhantes a outros pontos de dados no mesmo grupo do que os de outros grupos. Em palavras simples, o objectivo é segregar grupos com traços semelhantes e atribuí-los em *clusters*.

Recorremos ao nodo ***K-means*** e ao ***Cluster Assigner***, visto que são relativamente simples de implementar e funcionam bem para grandes conjuntos de dados.

Partiotining / X-Partitioner

Para dividir os nossos dados em conjuntos de treino e testes vamos testar 2 formas diferentes: *partitioning* e *x-partitioner*. Para o partitioning usaremos *random sampling* numa proporção de 8:2 e para o caso do *x-partitioner* usamos *random sampling* com 10 iterações.

Mais à frente iremos apresentar os modelos que testamos em cada caso.

3.3 Conceção de modelos

3.3.1 Random Forest

Random Forest é um tipo de algoritmo de aprendizagem supervisionada que utiliza métodos de *bagging* para resolver tanto problemas de regressão como de classificação. O algoritmo funciona através da construção de uma multiplicidade de árvores de decisão no momento do treino e produz a média/modo de previsão das árvores individuais.

A implementação deste algoritmo poderia ser mais limitada no contexto de um problema de regressão dado que a gama de previsões que poderá fazer está limitada pelos valores mais altos e mais baixos nos dados de treino. Ou seja, este modelo não consegue extrapolar.

3.3.2 Linear Regression

Este modelo consiste num método de regressão estatística simples utilizado para análise preditiva e que mostra a relação entre as variáveis contínuas. A **regressão linear** mostra a relação linear entre a variável independente (eixo X) e a variável dependente (eixo Y), consequentemente chamada regressão linear. Se houver uma única variável de entrada (x), essa regressão linear é chamada de regressão linear simples. E se houver mais de uma variável de entrada, essa regressão linear chama-se regressão linear múltipla. O modelo de regressão linear dá uma linha recta inclinada descrevendo a relação dentro das variáveis.

Ou seja, de forma resumida, treinar um modelo em regressão linear é o processo de encontrar uma linha que melhor se ajuste aos pontos de dados disponíveis na parcela, para que possamos usá-la para prever valores de saída para entradas que não estão presentes no conjunto de dados que temos, acreditando que essas saídas cairiam na linha.

3.4 Análise crítica de resultados

No início deste trabalho, não sabíamos se seria possível prever, sem ser de forma quase aleatória a popularidade de uma música (quão popular seria). Depois de testarmos o nosso modelo percebemos que é mais fácil prever se uma canção será popular ou não (sim ou não) do que o quão popular seria. Com conhecimento da área percebemos que há fatores que não estão incluídos no estudo que terão peso, como o artista, o ano de lançamento ou mesmo o género da música. Os dados que temos são um pouco abstratos porque é difícil perceber o quão "dançável" uma música será.

4 Conclusão

Este trabalho foi muito útil na medida em que todo o processo de pesquisa e exploração de ferramentas se tornou muito vantajosa, permitindo um aprofundamento na plataforma KNIME.

Consideramos que concretizamos uma exploração devidamente organizada, no entanto, foi díficil equilibrar a atenção dada aos datasets dado que nos focamos inicialmente bastante no *dataset* fornecido e este acabou por ficar com um desíquilibrio no tratamento de forma positiva.

Por outro lado, consideramos que certas técnicas que não testamos ao longo do trabalho se resumem ao facto de que foi um desafio equilibrar também o aprofundamento e complexidade com os prazos e limites estipulados.

No entanto, consideramos que o trabalho está organizado, construído de forma metódica e que exploramos muito a ferramenta KNIME e respetivos nodos.

5 Referências

- kdnuggets.com/2020/10/explain-machine-learning-algorithms-interview.html
- https://www.kdnuggets.com/2022/02/random-forest-decision-tree-key-differences.html
- https://corporatefinanceinstitute.com/resources/knowledge/other/decision-tree/
- https://bookdown.org/acitofrank/TextbookDraft/tree-models.html
- https://hub.knime.com/knime/extensions/org.knime.features.base/latest/org.knime.base.node
- https://www.kdnuggets.com/2022/03/machine-learning-algorithms-classification.html
- https://towardsai.net/p/machine-learning/why-choose-random-forest-and-not-decision-trees
- https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/
- https://blog.paperspace.com/gradient-boosting-for-classification/
- https://medium.com/swlh/gradient-boosting-trees-for-classification-a-beginners-guide-596b594a14ea
- https://learn.g2.com/logistic-regression
- https://www.keboola.com/blog/logistic-regression-machine-learning
- https://www.geeksforgeeks.org/advantages-and-disadvantages-of-logistic-regression/
- https://medium.datadriveninvestor.com/logistic-regression-essential-things-to-know-a4fe0bb8d10a

  https://www.knime.com/blog/four-basic-steps-in-data-preparation

  https://datascience.stackexchange.com/questions/28227/why-will-the-accuracy-of-a-highly-unbalanced-dataset-reduce-after-oversampling

  https://forum.knime.com/t/how-to-deal-with-unbalanced-data-smote-vs-equalizer/9358

  https://statisticsbyjim.com/basics/remove-outliers/ https://statisticsbyjim.com/basics/stratifi sampling/

  https://blog.ineuron.ai/Feature-Importance-in-Naive-Bayes-Classifiers-5qob5d5sFW

  https://towardsai.net/p/data-science/how-when-and-why-should-you-normalize-standardize-rescale-your-data-3f083def38ff