Universidade do Minho
Escola de Engenharia

Aprendizem e Decisão Intelegentes

Conceção de modelos de aprendizagem e decisão

Grupo 8:
A96357 - Luís Borges
A96854 - João Ferreira
A97352 - Flávio Silva
A100763 - José Vasconcelos

May 3, 2024

Índice

1 Introdução **4**

2 Domínios e Objetivos **4**

3 Metodologia **5**

3.1 Estudo do negócio 5

3.2 Estudo dos dados 5

3.3 Preparação dos dados 5

3.4 Modelação 5

3.5 Avaliação 5

3.6 Desenvolvimento 6

4 Descrição, exploração e tratamento de dados **6**

4.1 Dataset 1: Indian Liver Patient Dataset 6

10subsection5 4.2**Descrição dos modelos desenvolvidos** **16**

5.1 Dataset 1: Indian Liver Patient Dataset 16

5.1.1 Decision Tree 16

5.1.2 Random Forest 17

5.1.3 Naive Bayes 17

5.1.4 Logistic Regression 18

5.1.5 K Nearest Neighbor 18

5.1.6 SVM 18

5.1.7 Artificial Neural Network 19

5.1.8 K-Medoids 19
5.2 Dataset 2: NBA Team Summaries 20
5.2.1 Linear Regression 20
5.2.2 Simple Regression Tree 21
5.2.3 Polynomial Regression 21
5.2.4 Random Forest 21
5.2.5 Artifical Neural Network 22

6 Análise e Resultados **23**
6.1 Dataset 1: Indian Liver Patient Dataset 23
6.2 Dataset 2: NBA Team Summaries 25

7 Sugestões e Recomendações **29**

8 Conclusão **30**

1 Introdução

O objetivo deste trabalho passa por utilizar o software *Knime* para a conceção de um projeto focado em modelos de aprendizagem.

Serão utilizados dois *datasets*, um cedido pela equipa docente e um escolhido pelo grupo, para o processo de conceção do projeto. O grupo deve explorar, analisar e preparar os datasets, de forma a tirar partido dos vários modelos disponíveis e, por fim, realizar uma análise crítica dos resultados obtidos.

2 Domínios e Objetivos

Como referido acima, neste projeto serão explorados dois *datasets* completamente diferentes. O primeiro *dataset*, providenciado pela equipa docente, consiste num conjunto de dados médicos sobre pacientes indianos. Para este conjunto de dados, o objetivo será classificá-los como pacientes doentes ou pacientes saudáveis.

Para o segundo *dataset*, a equipa docente deu ao grupo a oportunidade de escolher um, a seu gosto. O nosso grupo tomou a decisão de trabalhar com um conjunto de dados relacionados com a liga de basquetebol norte-americana (NBA). A descrição de ambos os *datasets* será aprofundada na secção 4, mas, para já, cabe referir que os dados relacionados à NBA registam, para cada época regular, valores estatísticos ligados a cada equipa, tão variados como a idade média dos jogadores, se avançaram ou não para os *playoffs* ou a média de ressaltos ofensivos, por jogo. Estes valores serão alvo de uma clarificação, porém, para já, importa apenas indicar que o objetivo, neste caso, será determinar a percentagem de vitórias da equipa durante a fase regular.

Devido à própria natureza dos problemas, iremos tirar partido de modelos de classificação para trabalhar com o primeiro *dataset* e modelos de regressão para o segundo, demonstrando o nosso conhecimento e capacidade em ambos os tipos de problema.

3 Metodologia

A metodologia seguida, para os dois *datasets*, foi CRISP-DM (*Cross Industry Standard Process for Data Mining)*, que se subdivide em estudo do negócio, estudo dos dados, preparação dos dados, modelação, avaliação e desenvolvimento. Seguidamente, aprofunda-se cada um destes passos.

3.1 Estudo do negócio

Este passo passa pela compreensão do problema que o grupo tem em mãos, o objetivo a alcançar e todo o contexto relevante.

3.2 Estudo dos dados

Ao estudo do negócio segue-se o estudo dos dados, em que o objetivo é analisar os dados em jogo. Esta fase inclui a examinação dos vários atributos, incluindo o tipo (texto, número inteiro, etc.), a variância ou como se relacionam, por exemplo.

3.3 Preparação dos dados

Nesta fase, os dados devem ser preparados para ser fornecidos ao modelo de aprendizagem. É nesta parte que ocorrem os processos de normalização, seleção e eliminação de atributos e tratamento de *missing values*, entre outros.

3.4 Modelação

Aqui, um conjunto de dados de treino é aplicado a um modelo.

3.5 Avaliação

Por fim, os vários modelos são testados, através dos dados de teste, e avaliados, tirando partido das mais diversas métricas, de forma a perceber quais são os mais úteis na resolução do problema.

3.6 Desenvolvimento

Este passo passa pela colocação do modelo em produção, porém, vai para além do âmbito deste projeto.

4 Descrição, exploração e tratamento de dados

4.1 Dataset 1: Indian Liver Patient Dataset

Este dataset contém registos médicos de 583 pacientes indianos. Para além de um código de identificação, contém também a idade do paciente, a sua data de nascimento, o seu género e um conjunto de 10 indicadores médicos. Por fim, existe uma coluna que indica se o paciente possui mesmo uma doença no fígado.

O objetivo deste *dataset* passa pela caraterização dos pacientes em doentes ou saudáveis. Nesse sentido, concluímos que é um problema de classificação.

A exploração detalhada do *dataset* levou às seguintes conclusões e respetivas soluções:

- Existem duas colunas com o nome "TB", sendo que a primeira corresponde ao indicador *Total bilirubin* e a segunda ao indicador *Total Proteins*. A solução passou por mudar o nome da coluna referente ao total de proteínas para "TP".

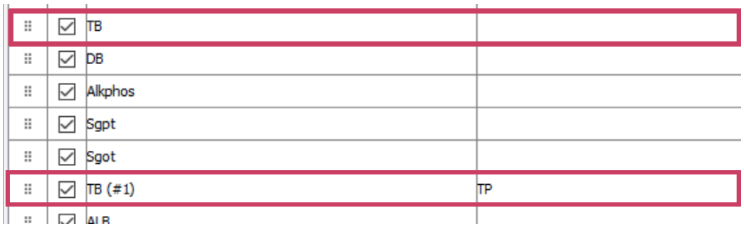


Figure 1: Colunas com o mesmo nome

- Os valores correspondentes ao género não estão uniformizados (por exemplo, para o género masculino, existem os valores "Male", "male" e "Masculine". Aqui, a solução passou pela aplicação de um nó *Java Snippet* que converte qualquer valor masculino no inteiro 1 e qualquer valor feminino no inteiro 2.

- O mesmo problema acontece na categoria Selector. Neste caso, os pacientes que não estão doentes são caraterizados por "2=no liver disease" ou "2=without liver disease". Para resolver esta situação, foi acrescentado ao *Java Snippet* de cima código para converter os valores correspondentes aos pacientes doentes em "1" e os valores dos pacientes saudáveis em "2". Importante notar que, sendo este o valor do *target*, estes valores continuam a ser *strings*.

Gender	☐	0	5	Male, Female, male, Masculine, female	
Selector	☐	0	3	1=liver disease, 2=no liver disease, 2=without liver disease	

Figure 2: Valores não-uniformizados em *Gender* e *Selector*

- As colunas "CHOL", "AG_Ratio" e "BILmg" contêm valores em falta, que devem ser tratados. Após vários ensaios, conclui-se que a melhor forma de tratar estes valores seria utilizar um nó *Missing Value* com a opção *Mean* para inteiros e *doubles*. Esta opção calcula a média de todos os valores existentes e substitui os não existentes pelo valor calculado.

Numeric | Nominal | Data Preview

Search:

Column	Exclude Column	Minimum	Maximum	Mean	Standard Deviation	Variance	Skewness	Kurtosis	Overall Sum	No. zeros	No. missings	No. NaN	No. +∞	No. -∞
id_code	☐	1	583	292	168.442	28372.667	0	-1.200	170236	0	0	0	0	0
Age	☐	4	90	44.746	16.190	262.111	-0.029	-0.560	26087	0	0	0	0	0
birth_year	☐	1933	2019	1978.254	16.190	262.111	0.029	-0.560	1153322	0	0	0	0	0
birth_month	☐	1	11	5.909	2.900	8.413	0.053	-1.148	3445	0	0	0	0	0
TB	☐	0.400	75	3.299	6.210	38.558	4.907	37.164	1923.200	0	0	0	0	0
DB	☐	0.100	19.700	1.486	2.808	7.888	3.212	11.353	866.400	0	0	0	0	0
Alkphos	☐	63	2110	290.576	242.938	59018.867	3.765	17.753	169406	0	0	0	0	0
Sgpt	☐	10	2000	80.714	182.620	33350.194	6.549	50.579	47056	0	0	0	0	0
Sgot	☐	10	4929	109.911	288.919	83473.916	10.546	150.920	64078	0	0	0	0	0
TP	☐	2.700	9.600	6.483	1.085	1.178	-0.286	0.233	3779.700	0	0	0	0	0
ALB	☐	0.900	5.500	3.142	0.796	0.633	-0.044	-0.388	1831.700	0	0	0	0	0
CHOL	☐	0	0	0	0	0	0	0	0	568	15	0	0	0
AG_Ratio	☐	0.300	2.800	0.949	0.320	0.102	1.004	3.278	549.500	0	4	0	0	0
BILmg	☐	0.023	4.386	0.191	0.358	0.128	5.067	40.122	107.204	0	22	0	0	0

Showing 1 to 14 of 14 entries

Figure 3: Valores em falta

- Muitos dos indicadores médicos têm valores que, claramente, constituem *outliers*, pelo que necessitam de ser tratados. Para este efeito, foi usado outro nó *Java Snippet* onde foi introduzido código com o intuito de eliminar estes valores discrepantes.

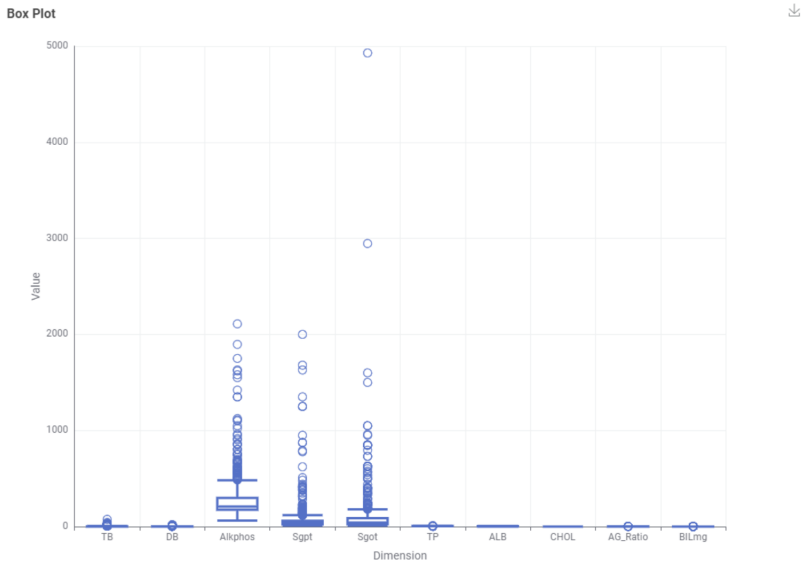


Figure 4: Presença de outliers

- Analisando as tabelas de correlação, concluímos que existem indicadores médicos com grande correlação entre si. Nomeadamente, os seguintes pares:

Atributo	**Correlação alta com:**
age	abbreviation; birth_year; birth_date;
birth_year	birth_date
TB	DB
TB	BILmg
DB	BILmg

Table 1: Elementos com alta correlação

Ademais, existem vários valores, como o id_code, que têm uma correlação desprezível. Existe também a coluna CHOL cujos valores estão todos a 0 e, por isso, apenas introduziriam confusão no modelo. Assim, após várias rondas de teste, optou-se por eliminar as seguintes colunas: id_code, birth_year, birth_month, birth_date, Gender, DB, CHOL, BILmg.

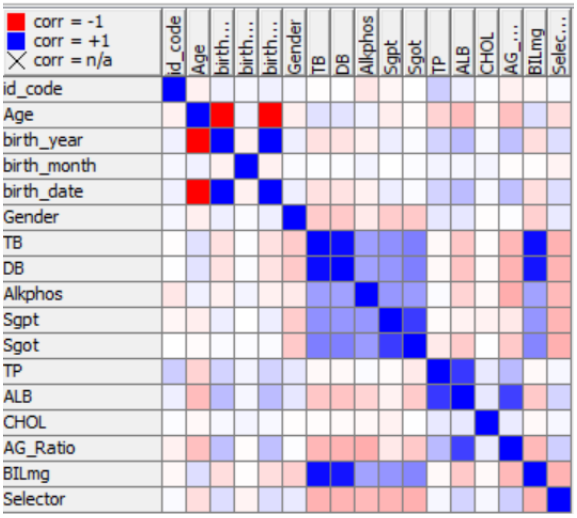


Figure 5: Tabela de correlação

- É também observável que o *dataset* não é balanceado, uma vez que existem 416 registos de pacientes doentes e apenas 167 registo de pacientes saudáveis. A resolução deste problema passou pelo uso de técnicas de *oversampling*, através da aplicação de um nó *SMOTE* aos dados de treino Este tema é abordado mais profundamente à frente.

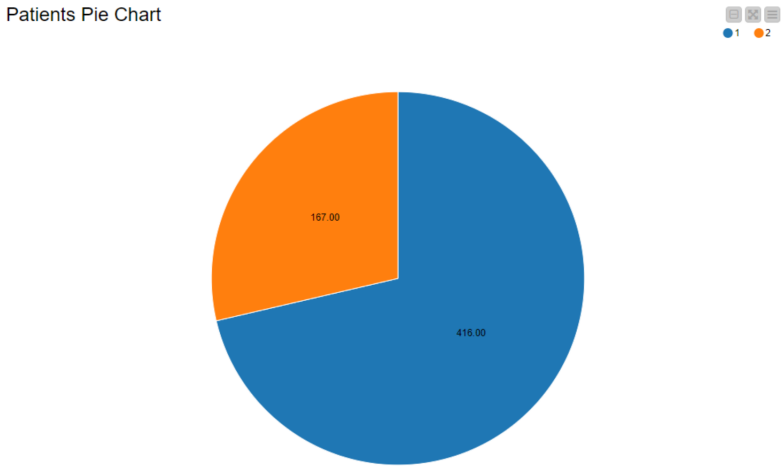


Figure 6: Pie chart do atributo *Selector*

Para além dos pontos previamente mencionados, a utilização de um nó de normalização foi um passo importante, garantindo que atributos com uma maior gama de valores não sobressaiam em relação aos outros.

4.2 Dataset 2: NBA Team Summaries[1]

O segundo *dataset* contém registos coletivos da liga de basquetebol norte-americana desde a época de 1950. Para cada temporada, inclui os registos estatísticos de cada equipa. Segue-se um breve glossário, descrevendo a informação presente em cada coluna:

- season: segundo ano da temporada (ou seja, o valor 2024 corresponde à temporada 2023/2024).
- lg: liga (o *dataset* contém registos de outras ligas, entretanto extintas, como a ABA).
- playoffs: Indica se a equipa se classificou para os *playoffs*.
- age: idade média dos jogadores da equipa.
- w: número total de jogos ganhos durante a época regular.
- l: número total de jogos perdidos durante a época regular.
- pw: Pythagorean wins (indicador correspondente ao número total de vitórias esperado, baseado nas médias de pontos marcados e concedidos).
- pl: Pythagorean losses (indicador correspondente ao número total de derrotas esperado, baseado nas médias de pontos marcados e concedidos).
- mov: margem de vitória média.
- sos: dificuldade do calendário (na NBA, as equipas não enfrentam todas as outras o mesmo número de vezes, gerando calendários mais ou menos difíceis).
- srs: sistema de classificação das equipas que tira partido das duas métricas anteriores.
- o_rtg: estimativa de pontos marcados por 100 posses de bola.
- d_rtg: estimativa de pontos concedidos por 100 posses de bola.

[1]Este dataset pode ser encontrado em: `https://www.kaggle.com/datasets/sumitrodatta/nba-aba-baa-stats`. É, no entanto, relevante mencionar que, entre a data atual e a data em que o grupo fez *download* do dataset, este foi atualizado, pelo que os registos da época 2024 devem ser diferentes.

- n_rtg: estimativa da diferença de pontos por 100 posses de bola.
- pace: estimativa do número de posses de bola por 48 minutos.
- f_tr: número de lances livres tentados por cada lançamento de campo tentado.
- x3p_ar: percentagem de lançamentos de campo atrás da linha de triplo.
- ts_percent: métrica de eficácia de lançamento que leva em consideração lançamentos de 2 pontos, lançamentos de 3 pontos e lançamentos livres.
- e_fg_percent: métrica de eficácia de lançamento que ajusta o facto de um lançamento de 3 pontos é mais valioso do que um lançamento de 2 pontos.
- tov_percent: estimativa de perdas de bola por 100 posses de bola.
- orb_percent: estimativa da percentagem de ressaltos ofensivos possíveis que a equipa obteve.
- ft_fga: lances livres marcados por cada lançamento de campo tentado.
- opp_e_fg_percent: e_fg_percent (consultar acima) do adversário.
- opp_tov_percent: tov_percent (consultar acima) do adversário.
- opp_drb_percent: estimativa da percentagem de ressaltos defensivos possíveis que a equipa adversária obteve.
- opp_ft_fga: ft_fga (consultar acima) do adversário.
- arena: nome da arena da equipa.
- attend: total de adeptos nos jogos em casa.
- attend_g: média de adeptos nos jogos em casa.

Tendo em conta que o primeiro *dataset* resultou num problema de classificação, o grupo decidiu, de forma a demonstrar amplamente os conhecimentos obtidos na UC, resolver um problema de regressão com este conjunto de dados. Assim, o objetivo passa por prever a percentagem de vitória de uma certa equipa numa certa época, a partir das métricas estatísticas correspondentes.

Dessa forma, a análise do conjunto de dados resultou nas seguintes conclusões e soluções:

- Existem dados sobre a época atual que, à data de recolha do *dataset*, ainda não tinha acabado, pelo que os dados seriam imprecisos. A solução passou pelo uso de um nó *Row Filter* para eliminar as linhas cuja coluna "season" tenha o valor 2024.

Table View

Rows: 1845 | Columns: 31

RowID	season Number (inte...	lg String	team String	abbreviat... String	playoffs String	age String	w String	l String	pw String	pl String	mov String
Row0	2024	NBA	Atlanta Hawks	ATL	FALSE	26.1	26	33	26	33	-2.14
Row1	2024	NBA	Boston Celtics	BOS	FALSE	28.4	46	12	45	13	10.43
Row2	2024	NBA	Brooklyn Nets	BRK	FALSE	26.4	23	36	25	34	-2.39
Row3	2024	NBA	Chicago Bulls	CHI	FALSE	28	28	31	27	32	-1.24
Row4	2024	NBA	Charlotte Hor...	CHO	FALSE	25.2	15	44	13	46	-10.54

Figure 7: Valores da época atual

- Existem dados sobre épocas de equipas noutras ligas que não a NBA. Estes dados não são relevantes para o problema, logo, devem ser eliminados, pelo que se tirou partido de um nó *Row Filter* para remover todas as linhas cujo valor da coluna "lg" não fosse "NBA"

Column	Exclude Column	No. missings	Unique values	All nominal values	Frequency Bar Chart
lg	☐	0	3	NBA, ABA, BAA	

Figure 8: Dados sobre outras ligas

- Cada época tem, para além de uma linha por equipa, uma linha onde, na coluna da equipa, está presente o valor "League Average". Estas linhas iriam apenas introduzir redundância estatística, uma vez que contém informação derivada de outras linhas, pelo que devem ser eliminadas. Para isso, mais uma vez, foi usado um nó *Row Filter*.

Row ID	season	lg	team	abbreviation	playoffs	age	w	l	pw	pl	mov	sos
Row30	2024	NBA	League Average	NA	FALSE	26.6	NA	NA	30	29	0	0
Row61	2023	NBA	League Average	NA	FALSE	26.3	NA	NA	41	41	0	0
Row92	2022	NBA	League Average	NA	FALSE	26.3	NA	NA	41	41	0	0

Figure 9: Valor "League Average" na coluna team

- Grande parte das colunas com métricas estatísticas foram automaticamente classificadas pelo nó *CSV Reader* como strings, apesar de serem, na sua grande maioria, números. Isto

acontece porque, quando não há informação nos campos, em vez de vazios, contém uma string "NA", tornando-se assim impossível definir os valores como *double* no nó *CSV Reader*. Para combater este problema, é usado um nó *String to Number*, convertendo estas colunas para *doubles*.

RowID	season Number (inte...	lg String	team String	abbreviat... String	playoffs String	age String	w String	l String	pw String	pl String	mov String	sos String	srs String
Row0	2024	NBA	Atlanta Hawks	ATL	FALSE	26.1	26	33	26	33	-2.14	-0.09	-2.22
Row1	2024	NBA	Boston Celtics	BOS	FALSE	28.4	46	12	45	13	10.43	0.05	10.48
Row2	2024	NBA	Brooklyn Nets	BRK	FALSE	26.4	23	36	25	34	-2.39	0.43	-1.96
Row3	2024	NBA	Chicago Bulls	CHI	FALSE	28	28	31	27	32	-1.24	-0.3	-1.54
Row4	2024	NBA	Charlotte Hor...	CHO	FALSE	25.2	15	44	13	46	-10.54	-0.03	-10.57

Figure 10: Métricas classificadas como string

- Excluindo a coluna *season*, todas as colunas numéricas têm, pelo menos, um valor em falta. Tal como no primeiro *datset*, os testes permitiram perceber que a melhor solução seria utilizar um nó *Missing Value* que substitui os valores ausentes pela média dos existentes.

Column	Exclude Column	Minimum	Maximum	Mean	Standard Deviation	Variance	Skewness	Kurtosis	Overall Sum	No. zeros	No. missings	No. NaN	No. +∞
season	☐	1950	2023	1994.171	19.184	368.009	-0.454	-0.677	3196656	0	0	0	0
age	☐	22.100	32	26.734	1.587	2.520	0.385	-0.010	42106.600	0	28	0	0
w	☐	7	73	39.918	12.348	152.462	-0.109	-0.604	63949	0	1	0	0
l	☐	9	73	39.918	12.291	151.057	0.208	-0.576	63949	0	1	0	0
pw	☐	7	70	39.988	11.832	139.994	-0.115	-0.600	64061	0	1	0	0
pl	☐	11	72	39.848	11.790	139.001	0.166	-0.594	63837	0	1	0	0
mov	☐	-15.200	12.280	-0.003	4.525	20.472	-0.180	-0.321	-5.230	0	1	0	0
sos	☐	-1.700	1.170	-0.000	0.401	0.161	-0.123	0.111	-0.760	11	1	0	0
srs	☐	-14.680	11.920	-0.006	4.340	18.839	-0.196	-0.273	-8.900	1	1	0	0
o_rtg	☐	80.400	119.400	104.454	6.599	43.553	-0.797	0.623	165559.400	0	18	0	0
d_rtg	☐	79.300	120	104.452	6.511	42.396	-0.954	0.932	165555.900	0	18	0	0
n_rtg	☐	-15.200	13.400	0.002	4.589	21.058	-0.197	-0.288	3.500	13	18	0	0
pace	☐	82.300	136.300	98.889	8.961	80.295	1.219	1.309	156739.200	0	18	0	0
f_tr	☐	0.194	0.554	0.313	0.048	0.002	0.583	0.846	500.785	0	1	0	0
x3p_ar	☐	0.011	0.519	0.192	0.115	0.013	0.308	-0.631	234.558	0	379	0	0
ts_percent	☐	0.377	0.610	0.526	0.035	0.001	-0.925	1.610	842.899	0	1	0	0
e_fg_percent	☐	0.310	0.575	0.481	0.041	0.002	-1.127	2.189	770.218	0	1	0	0

Figure 11: Valores ausentes

- Existem também um atributo nominal com o valor "NA", que não é reconhecido como um *missing value* por se tratar de uma *string*. No caso, corresponde à coluna "arena" e a melhor solução passou por remover essa linha.

_tr	x3p_ar	ts_percent	e_fg_perc...	tov_perce...	orb_perc...	ft_fga	opp_e_fg...	opp_tov_...	opp_drb_...	opp_ft_fga	arena	attend	attend_g
Number (dou...	Number (dou...	Number (dou...	Number (dou...	Number (dou...	Number (dou...	Number (dou...	Number (dou...	Number (dou...	Number (dou...	Number (dou...	String	Number (dou...	Number (dou...
.285	0.161	0.511	0.465	13.9	31.7	0.219	0.464	14	73.1	0.226	New Orleans ...	641,683	15,651
.315	0.192	0.478	0.429	16.5	28.3	0.232	0.464	17.5	68.9	0.263	Nassau Veter...	622,710.876	16,477.033
.313	0.192	0.526	0.481	14.077	28.742	0.235	0.492	14.077	71.239	0.229	NA	622,710.876	16,477.033
.299	0.192	0.489	0.44	17.5	28.1	0.229	0.478	18.4	68.7	0.25	Municipal Au...	622,710.876	16,477.033

Figure 12: Valor NA na coluna arena

- Tendo em conta o objetivo, que é calcular a percentagem de vitórias de cada equipa, é necessário, previamente, criar esse atributo, uma vez que ele não está presente. No entanto, uma vez que temos acesso ao número de vitórias e número de derrotas, para cada equipa, podemos facilmente calcular o valor deste atributo, para cada linha, através de um nó *Math Formula* que resulta na divisão entre o número de vitórias e a soma do número de vitórias e número de derrotas (ou seja, o número de jogos, uma vez que na NBA não há empates). A fórmula utilizada foi:

$$(w_pctg = w/(w + l))$$

  Aqui, é relevante mencionar que, uma vez que, durante a existência da NBA, o número de jogos por época não foi constante, a percentagem de vitória acaba por ser uma métrica mais vantajosa do que o número de vitórias ou de derrotas, uma vez que pode ser adaptada a qualquer época.

- A análise das tabelas de correlação permite encontrar os seguintes pares de elementos com grande correlação:

Atributo	**Correlação alta com:**
team	abbreviation
w	pw; mov; srs; n_rtg; w_pctg
l	pl; mov; srs; n_rtg; w_pctg
pw	mov; srs; n_rtg; w_pctg
pl	mov; srs; n_rtg; w_pctg
mov	srs; n_rtg; w_pctg
srs	n_rtg; w_pctg
o_rtg	ts_percent
n_rtg	w_pctg
f_tr	ft_fga
ts_percent	e_fg_percent

Table 2: Elementos com alta correlação

A solução passou pela aplicação de um nó *Column Filter* em que foram excluídos os seguintes atributos: lg, abbreviation, w, l ,pw, pl, mov, srs, n_rtg, e_fg_percent, ft_fga. É de notar aqui,

principalmente, a exclusão dos atributos w e l, uma vez que a previsão da percentagem de vitórias através de métricas que incluíssem o número de vitórias e o número de derrotas seria trivial.

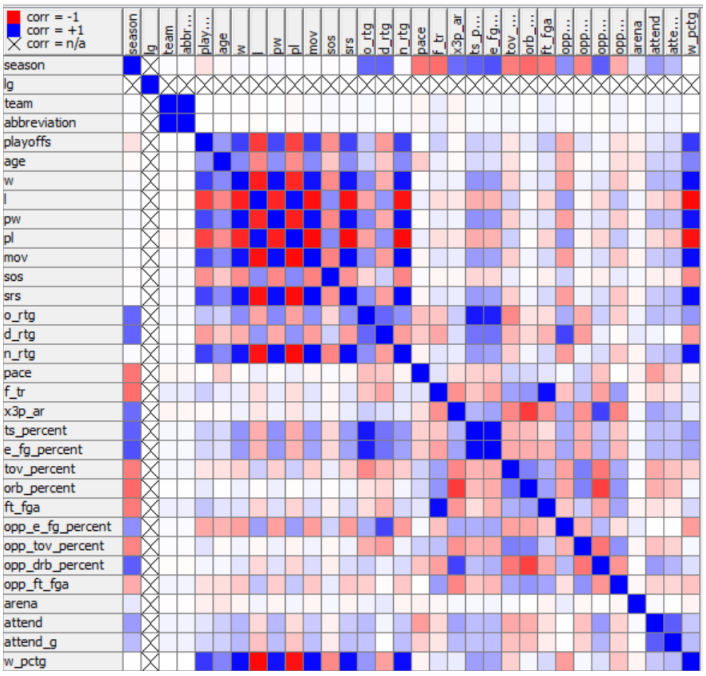


Figure 13: Tabela de correlação

- Por fim, é relevante mencionar a utilização de um nó de normalização no treino e teste do modelo, seguido de um nó de desnormalização antes da avaliação do modelo, tal como no primeiro *dataset*.

5 Descrição dos modelos desenvolvidos

5.1 Dataset 1: Indian Liver Patient Dataset

Para a resolução do primeiro problema, relacionado com o *dataset* providenciado pela equipa docente, foram aplicados os seguintes modelos:

- Decision Tree
- Random Forest
- Naive Bayes
- Logistic Regression
- K Nearest Neighbor
- SVM
- Artifical Neural Network
- K-Medoids

De forma a obter o melhor resultado possível e, simultaneamente, explorar e demonstrar o conhecimento obtido, foram desenvolvidas, para cada modelo, 3 versões diferentes: uma versão com partição em modo *holdout* e sem *oversampling*, uma outra versão com partição em modo *holdout* mas com *oversampling* e uma última versão, que não conta com *oversampling* mas conta com uma partição em modo *cross-validation*. A exceção a esta regra é o último modelo, para o qual foi aplicada apenas uma versão do modelo.

Segue-se uma breve descrição destes modelos e da forma como cada um foi aplicado:

5.1.1 Decision Tree

O modelo de árvore de decisão é um modelo que usa um conjunto de nós e ramos para chegar às melhores decisões possíveis, assemelhando-se a uma árvore. Cada um dos nós internos representa um teste num dos atributos, cada ramo representa o resultado do teste e cada uma das folhas representa a decisão tomada. Os parâmetros selecionados no *learner*, para cada versão, estão detalhados na seguinte tabela:

Parâmetro	Sem Oversampling	Com Oversampling	Cross-Validation
Quality Measure	Gain ratio	Gini index	Gini index
Pruning Method	No Pruning	No Pruning	No Pruning
Reduced Erro Pruning	Sim	Sim	Sim
Min number records per node	60	60	60
Nr records to store for view	10 000	10 000	10 000

Table 3: Parâmetros do *learner*

5.1.2 Random Forest

O modelo de *Random Forest* é semelhante ao modelo de árvore de decisão, porém, em vez de uma, constrói várias árvores de decisão. Para problemas de classificação, tal como este, o resultado é calculado através da análise de todas as árvores desenvolvidas, sendo a classe "vencedora" a selecionada pelo maior número de árvores. Segue-se a tabela com os parâmetros utilizados.

Parâmetro	Sem Oversampling	Com Oversampling	Cross-Validation
Enable Hilighting	Não	Não	Sim (2000 patterns)
Split Criterion	Information Gain Ratio	Information Gain Ratio	Information Gain Ratio
Number of Models	200	200	25

Table 4: Parâmetros do *learner*

5.1.3 Naive Bayes

O modelo de Naive Bayes é um modelo de classificação que tem a particularidade de assumir que todos os atributos são independentes uns dos outros, ou, por outras palavras, dá a mesma importância a todos os atributos. Para os *learners* deste modelo, foram usados os seguintes parâmetros:

Parâmetro	Sem Oversampling	Com Oversampling	Cross-Validation
Default Probability	0,0001	0,0001	0,0001
Minimum Standard Deviation	0,0001	0,0001	0,0001
Threshold standard deviation	3,0	0,001	3,0
Max nr of unique values	2	2	2

Table 5: Parâmetros do *learner*

5.1.4 Logistic Regression

O modelo de regressão logística aplica uma função logística para chegar a uma probabilidade, sendo que cada coeficiente quantifica o impacto de uma variável independente na ocorrência do *target*. Foram usados os seguintes parâmetros:

Parâmetro	**Sem Oversampling**	**Com Oversampling**	**Cross-Validation**
Solver	IRLS	IRLS	IRLS
Max number of epochs	5	5	5
Epsilon	100	100	100

Table 6: Parâmetros do *learner*

5.1.5 K Nearest Neighbor

O modelo de K Nearest Neighbor é um algoritmo que, para cada *input*, analisa os k vizinhos mais próximos, ou seja, com valores mais semelhantes, e classifica o *input* com valor mais comum entre eles. Os *leaners* foram parametrizados da seguinte forma:

Parâmetro	**Sem Oversampling**	**Com Oversampling**	**Cross-Validation**
Number of neighbors (k)	5	6	6
Weight Neighbors by distance	Sim	Sim	Sim

Table 7: Parâmetros do *learner*

5.1.6 SVM

O modelo SVM (*Support vector machine*) procura encontrar um hiperplano (generalização de plano para n dimensões) capaz de separar ambas as classes da melhor forma possível. Este hiperplano é escolhido de forma a maximizar a distância mínima entre os 2 pontos mais próximos de cada classe. Foram implementados os seguintes parâmetros:

Parâmetro	Sem Oversampling	Com Oversampling	Cross-Validation
Overlapping penalty	1.0	1.0	1.0
Kernel	RBF	RBF	RBF
Params	sigma=0,5	sigma=0,5	sigma=0,001

Table 8: Parâmetros do *learner*

5.1.7 Artificial Neural Network

As redes neuronais artificiais tiram partido de *perceptrons* com várias camadas para resolver problemas de classificação ou regressão. Este tipo de modelo é, dos usados, o mais semelhante à mente humana. No KNIME, foi usado o nó RProp MLP Learner, parametrizado da seguinte forma:

Parâmetro	Sem Oversampling	Com Oversampling	Cross-Validation
Max nr of iterations	75	75	75
Hidden layers	1	1	1
Hidden neurons per layer	30	30	30

Table 9: Parâmetros do *learner*

O nó DL4J Feedforward Leaner, do Knime também constitui um modelo de ANN. Como tal, também foi introduzido, usando os seguintes parâmetros:

Parâmetro	Sem Oversampling	Com Oversampling	Cross-Validation
Nr of iterations	10	10	10
Algorithm	SGD	SGD	SGD
Updater	ADAGRAD	ADAGRAD	ADAGRAD
Global Learning Rate	1.0	1.0	1.0
Batch Size	128	128	128
Epochs	15	15	15

Table 10: Parâmetros do *learner*

5.1.8 K-Medoids

Contrariamente aos anteriores, este modelo tira partido do paradigma de aprendizagem sem supervisão, que consiste num processo de aprendizagem em que os *learners* não têm informação

sobre os resultados esperados. O modelo separa os dados em k grupos, tentando minimizar a distância entre os pontos de um grupo e o centro. No modelo K-Medoids, especificamente, o centro do grupo é um dos próprios pontos.

Parâmetro	**Valor**
Partition count (k)	2
Chunk Size	10
Number of iterations	100

Table 11: Parâmetros do *learner*

5.2 Dataset 2: NBA Team Summaries

O segundo problema, baseado no *dataset* relacionado com a NBA, constitui um problema de regressão e, como tal, foram testados modelos diferentes, sendo estes:

- Linear Regression
- Simple Regression
- Polynomial Regression
- Random Forest
- Artifical Neural Network
- DL4J

Mais uma vez, para cada um dos modelos, foram construídas duas versões. Uma em que a divisão dos dados entre dados de treino e teste é feita em modo *holdout* e outra versão em que é usado o método de *cross validation*. No entanto, para este problema, não foram usados métodos de *oversampling*, uma vez que o *dataset* é relativamente balanceado.

Segue-se uma breve descrição destes modelos e da forma como cada um foi aplicado:

5.2.1 Linear Regression

Um modelo de regressão linear estima o valor de uma variável (neste caso, a percentagem de vitórias de uma equipa numa época) através de um conjunto de atributos, tirando partido de uma

equação linear em que os coeficientes são ajustados de forma a minimizar a diferença entre os valores observados e esperados. Foram utilizados, para cada *learner*, os parâmetros por defeito.

5.2.2 Simple Regression Tree

Este modelo é semelhante às árvores de decisão usadas acima, porém, adaptado a problemas de regressão

Parâmetro	**Holdout**	**Cross-Validation**
Use binary splits for nominal attributes	Sim	Sim
Missing value handling	Surrogate	Surrogate

Table 12: Parâmetros do *learner*

5.2.3 Polynomial Regression

O modelo de regressão polinomial é uma adaptação do modelo de regressão linear que permite calcular a relação entre as variáveis através de um polinómio, permitindo um melhor ajuste à complexidade dos dados. Os *learners* aplicados foram parametrizados da seguinte forma:

Parâmetro	**Holdout**	**Cross-Validation**
Maximum Polynomial degree	4	4

Table 13: Parâmetros do *learner*

5.2.4 Random Forest

O modelo *Random Forest* para regressão é semelhante ao modelo com o mesmo nome para classificação, sendo que, neste, são usadas árvores de regressão. Seguem-se os parâmetros dos *learners*:

Parâmetro	**Holdout**	**Cross-Validation**
Enable highlighting	Sim (2 000 patterns)	Não
Number of models	200	200

Table 14: Parâmetros do *learner*

5.2.5 Artifical Neural Network

As redes neuronais artificiais já foram abordadas acima, sendo que funcionam tanto para problemas de classificação como para problemas de regressão. Para os *learners* RProp, foram usados os seguintes parâmetros:

Parâmetro	**Holdout**	**Cross-Validation**
Number of iterations	3000	3000
Number of hidden layers	1	1
Number of hidden neurons per layer	1	1

Table 15: Parâmetros do *learner*

Foi também usado um nó *DL4J*, com os seguintes parâmteros:

Parâmetro	**Holdout**	**Cross-Validation**
Number of iterations	2	1
Optimization Algortihm	SGD	SGD
Updater	ADADELTA	ADADELTA
Global Learning Rate	0.001	0.001
Batch Size	128	200
Epochs	15	15

Table 16: Parâmetros do *learner*

Para cada modelo, foram usadas duas *layers*, ambas parametrizadas da seguinte forma:

Parâmetro	**Valor**
Number of Output Units	3
Learning Rate	0.01
Updater	ADADELTA
Weight Initialization Strat.	RELU_UNIFORM
Activation Function	Softsign

Table 17: Parâmetros do *learner*

6 Análise e Resultados

6.1 Dataset 1: Indian Liver Patient Dataset

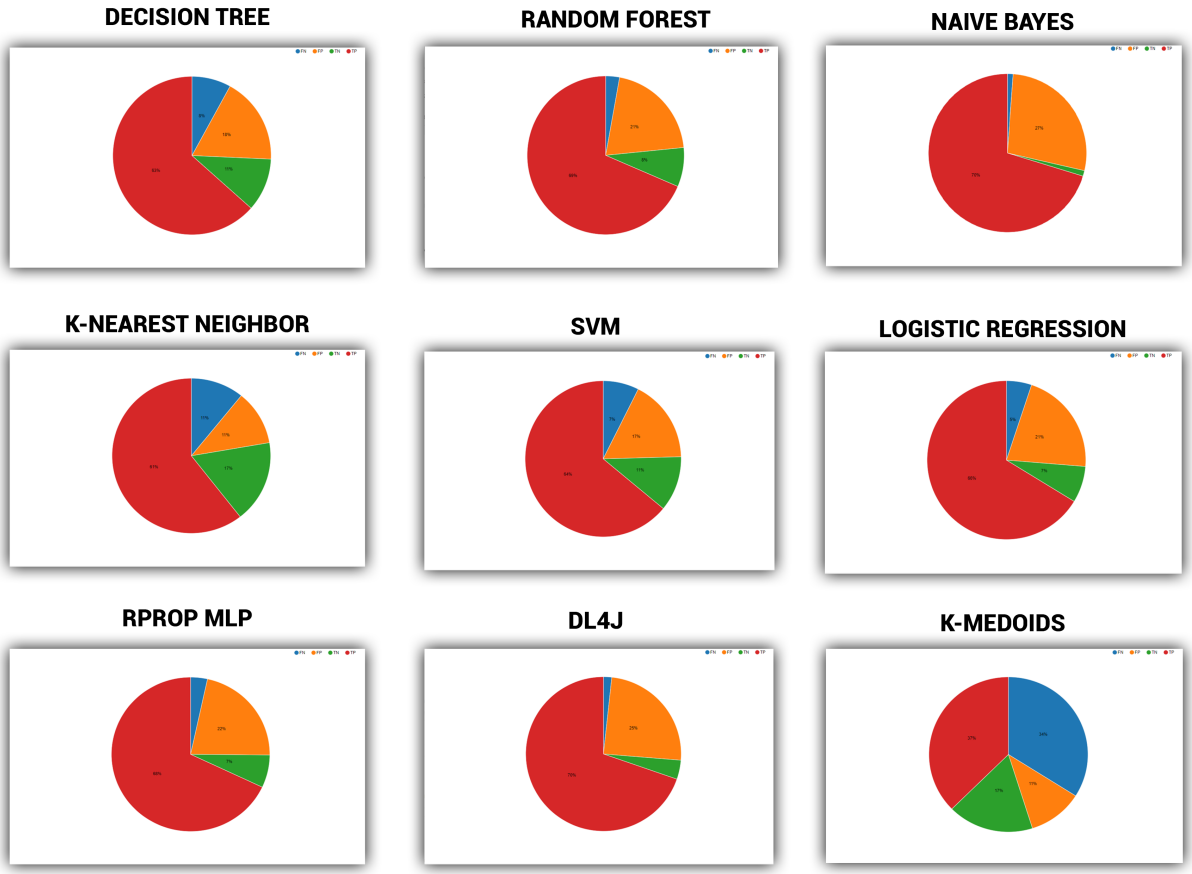
Para este *dataset*, foram obtidos os seguintes resultados:

Modelo	**Sem Oversampling**	**Com Oversampling**	**Cross-Validation**
Decision Tree	74.286	72.571	71.527
Random Forest	76.571	74.857	71.698
Naive Bayes	71.429	62.286	71.527
Logistic Regression	73.714	66.286	73.242
K Nearest Neighbor	77.714	70.286	70.840
SVM	76.000	73.143	72.384
RProp MLP	74.857	70.286	72.041
DL4J	73.714	46.857	72.014

Table 18: Campo *Accuracy* do *scorer* ligado a cada modelo

Nota-se, aqui, que o último modelo desenvolvido, *K-Medoids*, teve uma *accuracy* de 54.889%.

Os *Pie charts*, apresentados de seguida, formam uma excelente representação gráfica dos resultados. Neste caso, apresentam-se apenas os gráficos para os modelos gerados em modo *holdout* e sem oversampling, ou seja, correspondentes à primeira coluna da tabela acima.



Uma análise dos resultados requer, sempre, um conhecimento profundo do contexto e esse aspeto não poderia ser mais relevante neste problema. Pela análise puramente analítica da tabela, seria fácil chegar à conclusão de que o modelo a utilizar seria o *K-Nearest Neighbor*, uma vez que é o que tem maior *accuracy*.

No entanto, considerando o âmbito do problema, é possível que o(s) utilizador(es) do programa (imagine-se, a equipa médica de um hospital) pretenda(m), acima de tudo, minimizar os resultados falsos negativos, uma vez que o pior cenário seria não diagnosticar um paciente que esteja, de facto, doente. Deste perspetiva, pode argumentar-se que o modelo *Naive-Bayes* seria o melhor a aplicar aqui porque, apesar de ser o que tem menor *accuracy*, apresenta a percentagem inferior de falsos negativos.

É, todavia, relevante referir que isto acontece porque este modelo acaba por ser demasiado

seguro, classificando 97% dos pacientes como doentes. Como é lógico, um modelo que classifica quase todos os pacientes como doentes não irá gerar muitos falsos negativos, mas a sua utilidade também será menor. Seria interessante encontrar um modelo equilibrado, com alta *accuracy*, baixa taxa de falsos negativos e que não fosse excessivamente seguro. Dos modelos desenvolvidos, o *Random Forest* seria o que melhor encaixaria nesta descrição, possuindo a segunda *accuracy* mais alta e classificando apenas 2% dos pacientes como falsos negativos. Note-se também que este modelo não é demasiado cauteloso, uma vez que classifica cerca de 10% dos pacientes como saudáveis.

6.2 Dataset 2: NBA Team Summaries

Para este *dataset*, foram obtidos os seguintes resultados:

Modelo	**Holdout**				**Cross-Validation**			
	R^2	**MAE**	**MSE**	**RMSE**	R^2	**MAE**	**MSE**	**RMSE**
Linear Regression	0.937	0.03	0.001	0.037	0.926	0.03	0.002	0.041
Simple Regression Tree	0.829	0.048	0.004	0.062	0.760	0.054	0.005	0.073
Polynomial Regression	0.925	0.032	0.002	0.041	0.911	0.033	0.002	0.044
Random Forest	0.879	0.041	0.003	0.052	0.874	0.041	0.003	0.053
RProp MLP	0.934	0.03	0.001	0.038	0.919	0.032	0.002	0.043
DL4J	0.038	0.124	0.023	0.151	-0.3	0.142	0.029	0.17

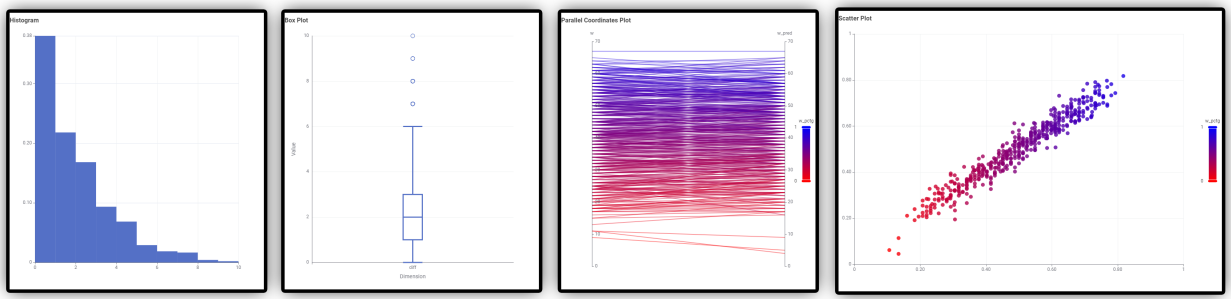
Table 19: Campo R^2 do *scorer* ligado a cada modelo

No entanto, a análise dos resultados não se fica apenas pela observação dos valores do *scorer*. Para cada modelo, após ter sido obtida uma predição da percentagem de vitórias, multiplicou-se esse valor por 82 e converteu-se para inteiro. O mesmo foi feito para o valor real da percentagem de vitórias, fazendo com que, nesta fase, exista uma tabela em que, em cada linha, há uma coluna com o número de vitórias real correspondente a uma época de 82 jogos, assim como o número de vitórias estimado. Foi também, posteriormente, criada uma coluna correspondente ao módulo da diferença entre os dois valores anteriores, de forma a aprofundar a análise dos resultados.

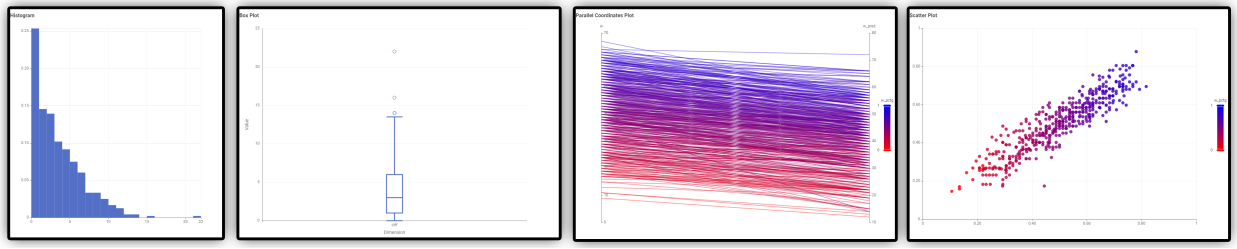
Foram também desenvolvidas representações gráficas dos resultados, nomeadamente:

- Um gráfico de barras com uma barra para cada valor da coluna "diff", representante da diferença de vitórias entre o valor real e estimado.
- Um *box plot* dos valores da coluna "diff"
- Um quadro de coordenadas paralelas entre os valores do número de vitórias real e estimado,
- Um gráfico de dispersão entre a percentagem de vitórias real e estimada.

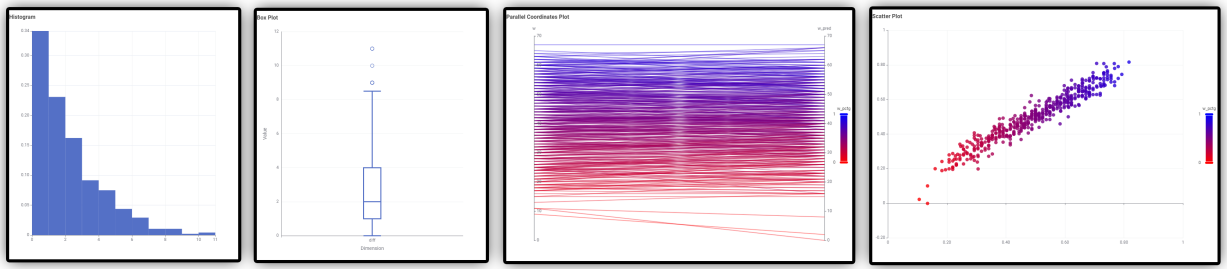
MODEL: LINEAR REGRESSION



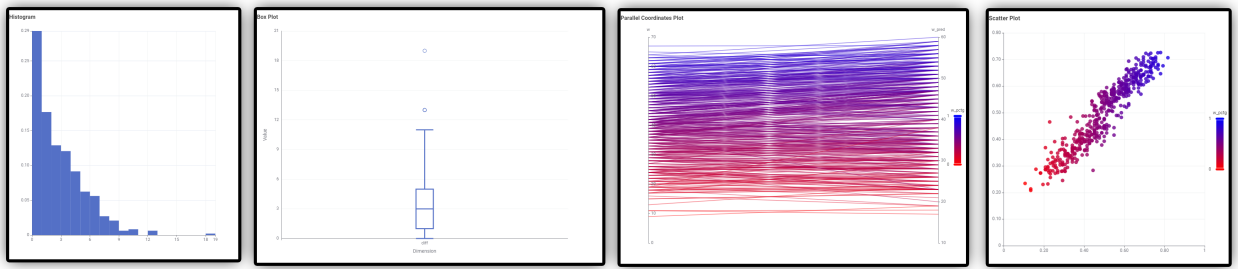
MODEL: REGRESSION TREE



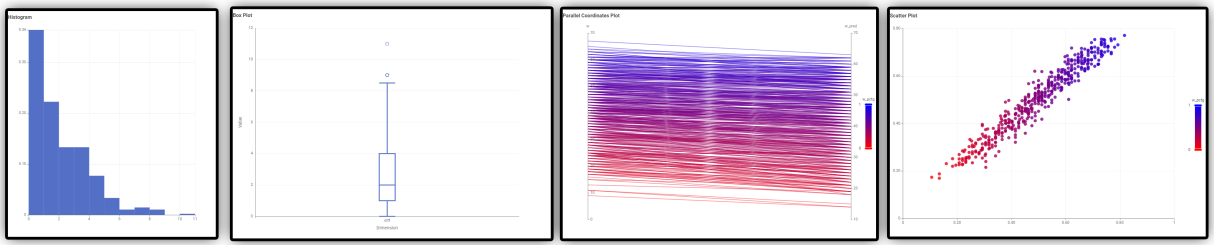
MODEL: POLYNOMIAL REGRESSION



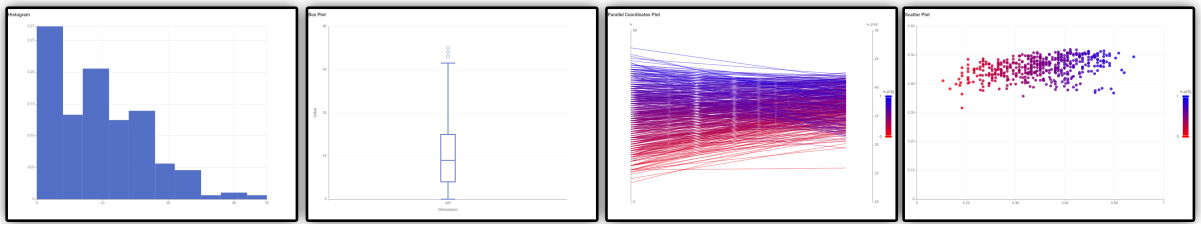
MODEL: RANDOM FOREST



MODEL: RPROP



MODEL: DL4J



A primeira conclusão, clara e imediata, é que o último modelo é claramente inferior aos outros. As métricas do *scorer* são todas inferiores e os gráficos mostram o mesmo. Apesar de, à primeira vista, inesperados, estes resultados não são necessariamente incomuns. Tudo indica que, para este conjunto de dados, o modelo DL4J acaba por ser demasiado complexo, introduzindo confusão nos resultados. Desta forma, é compreensível que os modelos mais simples tenham melhor desempenho.

Concluímos também, por observação dos gráficos, que, à exceção do último, os modelos comportam-se de forma satisfatória, uma vez que todos os gráficos de dispersão se parecem, mais ou menos, com uma linha y=x e que os gráficos de coordenadas paralelas apresentam linhas, na sua maioria, perfeitamente ou quase perfeitamente horizontais.

Para esta *dataset*, acreditamos que o melhor modelo é o primeiro, ou seja, o modelo de regressão linear. Par além do seu *scorer* apresentar as métricas mais favoráveis, sabemos também, através

da observação dos dados e gráficos, que é o modelo com a maior percentagem de acerto total no número de vitórias (38%) e o modelo em que o maior valor da diferença entre predição e valor real do número de vitórias é menor (10).

Assim, um responsável pela gestão de uma equipa da NBA ou, por exemplo, um jornalista interessado no assunto, escolheria provavelmente o modelo de regressão linear. No entanto, os modelos de regressão polinomial e RProp MLP também constituiriam escolhas válidas.

7 Sugestões e Recomendações

Para ambos os *datasets*, é interessante notar que os valores correspondentes aos modelos de *cross-validation* são sempre relativamente inferiores aos valores dos modelos treinados e testados em modo *holdout*. Este fator indica que os segundos poderão estar a ser alvo de *overfitting* e devem ser usados mais cautelosamente.

É de notar também que os resultados do problema correspondente ao primeiro conjunto de dados, na sua generalidade, pioram com o uso de *oversampling*, pelo que seria interessante procurar outras formas de ajustar os modelos ao facto de os dados não estarem balanceados.

8 Conclusão

Em conclusão, este projeto demonstrou a capacidade da plataforma *Knime* para todo o processo de desenvolvimento de modelos, quer para problemas de classificação, quer para problemas de regressão.

O grupo também demonstrou o seu conhecimento e perícia, tanto teoricamente como na prática, em todos os passos, desde a análise dos dados à avaliação dos modelos, tanto para o primeiro *dataset*, fornecido pela equipa docente, como para o segundo, escolhido por nós.

Este projeto permitiu aos elementos adquirir competências que, de outra forma, não conseguiriam, numa área tão relevante como a de *machine learning*, providenciando, ao mesmo tempo, um meio para os alunos desenvolverem as suas habilidades partindo de temas sobre os quais têm interesse, através da possibilidade de escolha de um *dataset*, tornando o projeto ainda mais cativante.