Relatório Ciência de Dados

1 Introdução

Neste relatório iremos explicar que abordagens foram escolhidas para os dois *datasets Heart Failure Prediction* e *QSAR Oral Toxicity*, e também explicar o porquê das mesmas. Por questões de simplicação iremos referir-nos ao *Heart Failure Prediction* como *dataset 1* e ao *QSAR Oral Toxicity* como *dataset 2*. À medida que formos apresentando os resultados iremos também fazer a sua avaliação e análise crítica.

2 Perfil dos dados

Em relação ao perfil dos dados o *dataset 1* tem 299 registos e 13 variáveis das quais 6 são categóricas (*anaemia, diabetes, high_blood_pressure, sex, smoking e DEATH_EVENT*), 3 variáveis de virgula flutuante (*age, placetes e serum_creatinine*) e por fim 4 variáveis continuas (*creatinine_phosphokinase, ejection_fraction, serum_sodium e time*), havendo um rácio de 23.33 registos por variável. Neste *dataset* a variável que pretendemos prever é a ***DEATH_EVENT***, sendo os valores possíveis 0 (não ocorreu o evento) ou 1 (ocorreu o evento). Ao fazer a sua analise conferimos que não existem valores em falta.

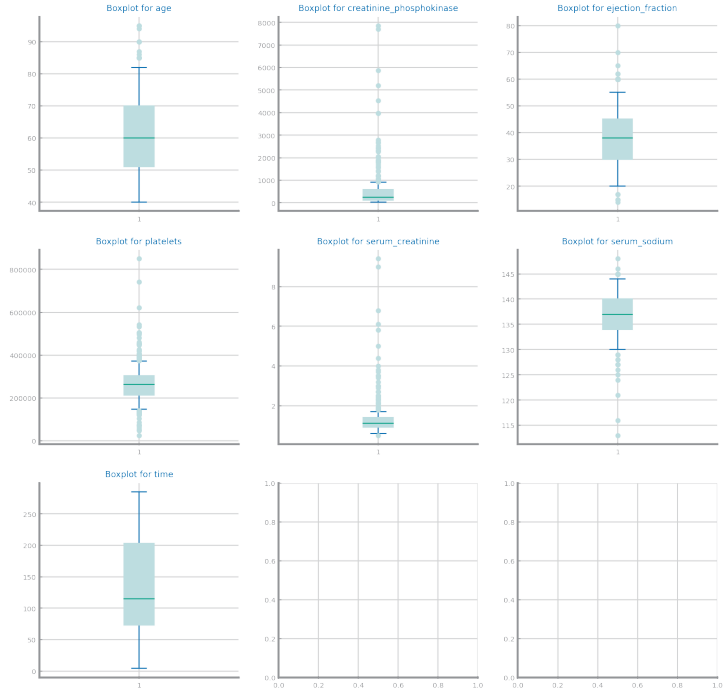

Figura 1: *Boxplot das variáveis não categóricas, não normalizadas*

Ao criarmos o *boxplot* das variáveis, percebemos que algumas delas tinham uma escala muito diferente das outras, por exemplo, algumas [0,1]e outras [0,8000000], sendo assim, foi necessário fazer a normalização como vamos explicar no próximo tópico.

Os dados não estavam balanceados uma vez que a nossa variável alvo tinha 203 entradas do valor 0 e 96 entradas do valor 1, tendo a classe negativa mais do dobro das entradas da classe positiva, sendo assim foi necessário fazer o balanceamento dos dados. Através da matriz de correlação foi possível perceber que a correlação entre as variáveis era bastante baixa visto que obtivemos uma matriz esbranquiçada uma vez que a cor branca foi escolhida para o valor 0. Ainda no *dataset 1* é possível encontrarmos alguns *outliers*, no entanto não é aconselhado fazer a sua remoção uma vez que a quantidade de dados é relativamente pequena. Ao fazermos os histogramas juntamente com as distribuição normal, exponencial e inclinação normal (*skew normal*), percebemos que a maioria dos pacientes tinha idades entre os 50 e os 65 anos com cerca de um terço dos pacientes acima dos 65 anos, as plaquetas variavam entre 150 mil e 400 mil, a fracção de ejecção variou significativamente entre os pacientes indo de 15% até 55% já excluindo os *outliers*, e o sódio variou principalmente entre os valores de 132 e 142.

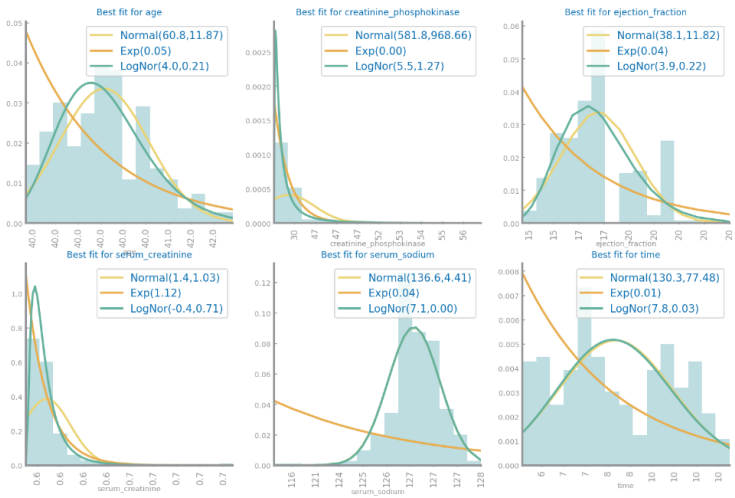

Figura 2: *Histograma com apoio da distribuição normal, exponencial e inclinação normal*

No *dataset 2* existem 8892 registos e 1025 variáveis todas elas categóricas, neste *dataset*, existe um rácio de 8.68 registos por variável. A variável que pretendemos prever neste *dataset* é ***IS_TOXIC*** que pode variar entre o valor *negative "not very toxic"* e *positive "very toxic"* (tendo sido o nome da variável atribuído por nós). Neste *dataset*, assim como no 1º não existem valores em falta. Tal como foi dito anteriormente, a variáveis presentes no *dataset 2* são categóricas binárias, sendo assim todas elas tem a mesma escala não necessitando de normalização. Em re-

lação aos *outliers* estes não fazem sentido neste conjunto de dados, no entanto o balanceamento é muito importante visto que temos 8251 (91.76%) registos *negative* e apenas 741 (08,24%) registos *positive* da variável alvo. Ao contrário do 1º *dataset*, não é possível visualizar a matriz de correlação, no entanto é possível interpretá-la, percebendo assim a baixa correlação existente entre as variáveis.

3 Preparação dos dados

A preparação dos dados teve uma grande importância no decorrer do trabalho visto que a qualidade dos resultados obtidos em ambos os *datasets* estavam directamente relacionados com a qualidade desta preparação. No dataset 1 não foram encontrados valores em falta. Caso existissem, seria necessário fazer o seu tratamento uma vez que a biblioteca *scikit-learn* não tem a capacidade de lidar com valores omissos, sendo assim, deveríamos substituir cada uma das entradas em falta por um valor (exemplo: *NA*). Após temos a certeza que não existiam valores em falta, deixamos de fazer esta validação no decorrer do trabalho.

Um *outlier* pode ter diferentes definições, tais como, um valor esperado bastante incomum que é demasiado grande ou pequeno quando comparado com a média das variáveis ou seja, um valor menor que μ-$n\sigma$, ou maior que μ+$n\sigma$, outra definição pode ser qualquer valor que seja inferior ao ***Q1 − 1,5*IQR*** ou superior a ***Q3 + 1,5*IQR***. No nosso trabalho testamos ambas, no entanto a primeira definição era demasiada restritiva, sendo assim a segunda definição foi mais aceite. Esta não pôde ser aplicada na sua integra uma vez que este *dataset* tinha poucas entradas, sendo assim foi necessário fazer uma adaptação, considerando um *outlier* uma entrada que não estivesse dentro dos valores aceites pelo menos 3 vezes. Com esta abordagem reduzimos a quantidade de elementos seleccionados como *outliers* e obtivemos uma ligeira melhoria de resultados, contudo esta melhoria não foi significativa.

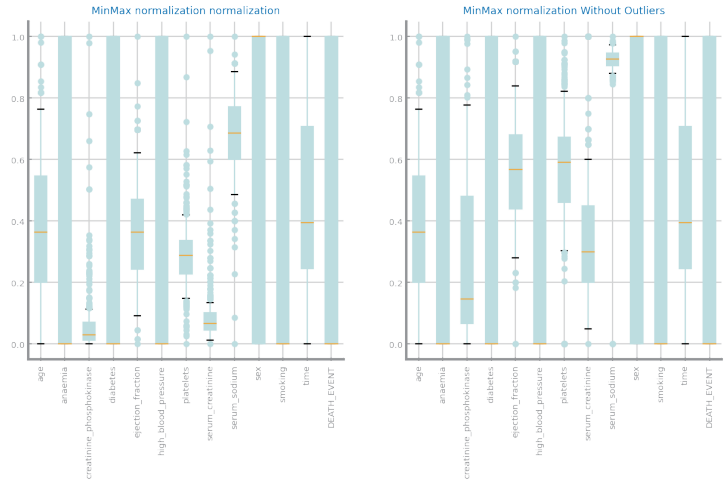

Figura 3: *Normalização MinMax com e sem outrliers*

Por fim, também testámos substituir os *outliers* pelo valor mínimo aceite caso este fosse inferior ao mesmo ou então pelo valor máximo aceite no caso de ser este ser superior, no entanto os resultados foram semelhantes à abordagem da remoção. No presente *dataset* existiam várias escalas entre as variáveis o que faria com que algumas destas tivessem um peso superior perante alguns modelos, sendo assim começamos por aplicar a normalização **Z-score** onde obtivemos uma escala no intervalo [-6,8], no entanto não foi esta normalização adoptada para o restante trabalho mas sim a **minmax** passando a ter um intervalo [0,1], é importante salientar que a normalização apenas alterou a escala e não a distribuição original.

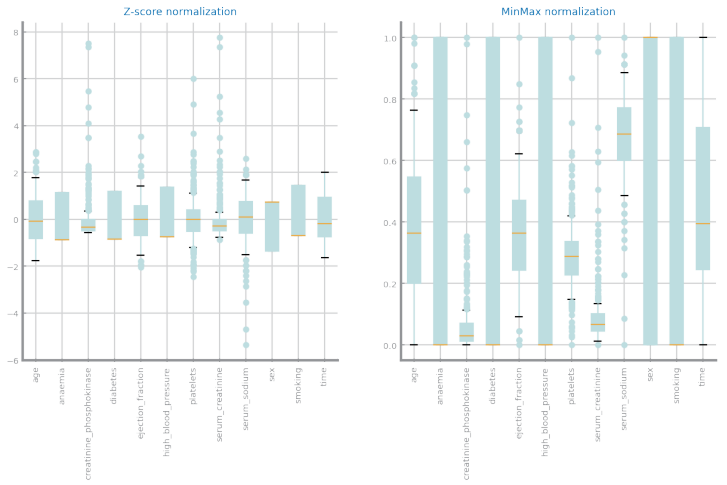

Figura 4: *Boxplot com as diferentes normalizações*

O *dataset* 1 não estava balanceado uma vez que a nossa variável alvo tinha 203 (67,9%) entradas do valor 0 e 96 (33,1%) entradas do valor 1, sendo assim, de forma a obtermos melhores resultados fizemos o balanceamento do dados. Começamos por aplicar a técnica de **oversampling** uma vez que este era um dataset bastante pequeno, após esta aplicação notamos de imediato uma melhoria significativa nos resultados. Testámos também a técnica **SMOTE** para balancearmos os dados, obtendo resultados semelhantes à primeira técnica no entanto com ligeiras melhorias. Por fim, apenas por curiosidade testamos a técnica de **undersampling** obtendo resultados piores como era esperado, salvo alguns casos excepcionais. De forma a termos um bom modelo e tentarmos evitar *overfiting*, aplicamos **K-Fold Cross Validation** que consiste em dividir os dados em dobras (fold's) e garantido que cada uma delas é utilizada como um conjunto de testes em algum momento, percebendo que o número ideal de dobras para este *dataset* variava entre 7 e 10. Desta forma também foi possível vermos o desempenho do nosso modelo com diferentes dados de entrada. Para alem deste modelo aplicamos também **Stratified K-Fold Cross Validation** que é muito semelhante ao anterior no entanto as dobras são feitas preservando a percentagem

de amostras para cada classe.

No segundo *dataset* grande parte das abordagens escolhidas anteriormente tiveram que ser reavaliadas visto que a estrutura deste era completamente diferente do *dataset 1*, sendo assim abordagens que tiveram um bom resultado anteriormente, neste deixaram de o ter. É viável afirmar que o oposto também é verdade. Tal como foi dito anteriormente, não existem valores omissos no presente *dataset*, posto isto não foi necessário fazer o seu tratamento. Uma vez que este *dataset* é constituído por variáveis categóricas não faz sentido procurarmos *outliers*, no entanto no inicio do projecto fizemos a sua procura por mera curiosidade. Devido à natureza dos dados não existiu a necessidade de fazermos normalização uma vez que todos os registos estavam na mesma escala, no entanto perante este conjunto de dados também não era possível fazer a normalização uma vez que as variáveis são categóricas (*very toxic* ou *not very toxic*), sendo assim não faria sentido. Este *dataset* estava extremamente desbalanceado uma vez que a nossa variável alvo tinha 11 vezes mais entradas da categoria "negative", do que da categoria "positive"tendo 8251 e 741 respectivamente.

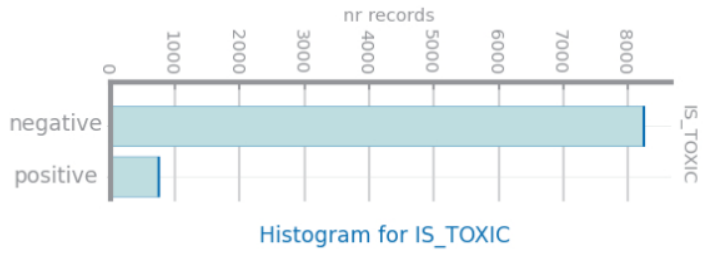

Figura 5: *Histograma da variável alvo*

Confrontados com estes dados foi necessário fazer o seu balanceamento. Já que o *dataset 2* tinha uma quantidade de registos considerável decidimos aplicar ***undersampling*** como método de balanceamento reduzindo assim a quantidade de dados para analisar, o que foi bastante importante para os passos seguintes uma vez que reduzimos o tempo de execução dos modelos. Testamos também o método de ***oversampling***, no entanto não era a metodologia correta pois estávamos a aumentar ainda mais a discrepância entre as duas categorias, logo optamos pelo primeiro método. Por ultimo testamos uma mistura entre ***undersampling*** e ***oversampling***, esperando encontrar melhores resultados, no entanto estes não tiveram uma grande melhoria. Neste *dataset* não utilizámos **K-fold Cross Validation**, apenas fizemos a divisão do dataset original onde colocamos 70% dos dados no dataset de treino e 30% no de teste.

Após a aplicação dos métodos de *scaling*, nomeadamente normalização para o *dataset 1*, não sendo realizado qualquer pré-processamento para o *dataset 2*, são realizadas técnicas de **redução de dimensionalidade**. Tais técnicas contribuem para guiar nas aprendizagens (focar nas regiões informativas e discriminatórias). Não obstante, removem possível ruído de componentes e aceleram a aprendizagem, tornando praticáveis técnicas de complexidade exponencial, tanto temporal como espacial. A realização do *balancing* é feita após esta metodologia pois pretendemos que não haja dados virtuais (não pré-existentes) a contribuir para as estimativas dos métodos de redução de dimensionalidade. Deste modo, destacamos o *Feature Selection* e o *Feature Extraction*.

No *Feature Selection* aplicámos três possíveis tipos de modelo: *Filter*, *Wrapper* e *Ensembled*. Dada a simplicidade e eficácia, foi considerada como primeira técnica remover as variáveis do dataset 2 cujo **variância** fosse igual ou inferior a $0.95 \times (1 - 0.95) = 0.0475$, isto é, cujas variáveis tivessem o mesmo valor (0 ou 1) em 95% ou mais dos registos ($0.95 \times 8892 = 8448$). De igual forma aplicámos *univariate feature selection* para cada um dos datasets, escolhendo a metodologia dos k-melhores ou, alternativamente, um dado percentil. No primeiro dataset, sendo as variáveis parcialmente numéricas e as restantes categóricas (booleanas), o mais adequado é aplicar *ANOVA* enquanto que para o segundo temos como mais apropriadas o χ^2 e o *Mutual Information*, uma vez que as variáveis são categóricas (booleanas). No dataset 1 a observação do *Heatmap* permitiu-nos determinar que as duas variáveis com maior correlação era inferior a 50% pelo que não considerámos nenhuma coluna como redundante. A aplicação do método *Decision Trees* permitiu excluir as variáveis com menor importância, com base na impureza, em que 3 tinham importância inferior a 3%. Para o dataset 2 obtivemos com o *Naive Bayes* uma *accuracy* de 83,6%, *recall* de 70,7% e *specificity* de 82,6% para o χ^2 quando consideramos um percentil de 20%, comparados com o dataset sem *Feature Selection* com valores de 79,2%, 67,1% e 80,3%. Para o *Mutual Information* com os mesmos parâmetros obtivemos uma *accuracy* e *specificity* superiores divergindo do *recall* que diminuiu para 53,15%, valor este não desejável. Para o *KNN* a *accuracy* e a *specificity* mantiveram-se constantes mas o *recall* teve um aumento de 35,1% para 41,4% no χ^2 e 37,9% para o *Mutual Information* quando considerados ambos com percentil de 0.20. Valores de k=400 conduziram em todos os casos para valores intermédios com e sem aplicação *Feature Selection*. Melhorias nas métricas consideradas devem-se essencialmente à eliminação de variáveis com menor relevância que têm impacto negativo no desempenho dos modelos. No entanto, se os parâmetros considerados forem demasiado restritivos tanto a *accuracy* como as restantes tenderão a obter piores resultados. O conjunto de treino é obtido com base nas variáveis selecionadas e não o oposto, sendo os valores aqui explicitados relativos ao conjunto de teste.

O ***Feature Extraction*** difere do *Feature Selection* em

que as variáveis são combinações (lineares ou não) das variáveis originais. Em particular, aplicámos *Principal Component Analysis* (PCA) de modo a reduzir as dimensões preservando informação dos dados tanto quanto possível. O PCA é uma técnica que procura uma base r-dimensional que melhor captura a variância nos dados. A direcção com a maior variância projectada é denominada primeira componente principal, e é também a que minimiza o erro médio quadrático. Numa primeira iteração e aplicado aos métodos de *Clustering*, ainda que mantendo o mesmo número de variáveis para cada um dos datasets, os valores de Índice de Silhueta para o *Expectation-Maximization*, *Hierarchical* e *KMeans* tiveram em média aproximadamente 25% do que se tinha obtido antes do *Feature Extraction*.

4 Sem supervisão

4.1 Regras de associação

Regras de Associação permitem fazer afirmações sobre a probabilidade de dois conjuntos co-ocorrerem ou de ocorrerem condicionalmente. Deste modo, o nosso objetivo é obter tais regras para as variáveis de ambos os datasets, para dados suportes e graus de confiança. Destacamos ainda o número de regras obtidas em função do suporte bem como a confiança e o *lift* em média em função da confiança (variando o valor destas variáveis independentes). Numa primeira abordagem recorremos ao algoritmo *Apriori*, sendo este o sugerido por ser de mais fácil compreensão. Contudo, em subsequentes cálculos que exigiam maior poder computacional demos preferência ao algoritmo *FPGrowth*, uma versão melhorada do *Apriori*, que permite que a procura de conjuntos de itens frequentes seja reduzida substancialmente: tal teve considerações na complexidade temporal e espacial. Os resultados seguintes corresponderão portanto ao *FPGrowth* onde naturalmente obteríamos (e efetivamente comparámos) os mesmos resultados com o algoritmo *Apriori*. No primeiro dataset obtivémos um tempo de execução do *Apriori* de 59,89 segundos, em comparação com o *FPGrowth* de 4,88s, isto é, 12 vezes menos tempo. Já para o *dataset 2* a diferença é mais notável dado termos tido 149,45s com o *Apriori* e 1,33s com o *FPGrowth* (menos de um centésimo do tempo do tempo). Estes valores para o *dataset 2* terão tido um limite de confiança mais elevado que para o dataset 1 que especificaremos de seguida. Relativamente ao dataset 1, aplicámos discretização unicamente para as variáveis numéricas recorrendo a *dummification* com 8 *bins*, dado que o *Pattern Mining* não se aplica diretamente a variáveis numéricas. Com isto, o número de padrões aumentou exponencialmente com o decréscimo de suporte, e, em particular, com suporte de 0,2, 0,1, 0,05 e 0,01 o número de padrões encontrados foi de 71, 291, 1123 e 19083, respetivamente, significando assim que existem 71 padrões em pelo menos 20% dos *records*. Consequentemente, também o número de regras em função da mesma escala de suporte apresenta o mesmo crescimento exponencial. Um número mais elevado de *bins* implicaria um número de reduzido de padrões. Obtivemos 4 571 796 regras de associação para um mínimo de confiança de 0,7, em que valores de confiança menores que 0,5 têm pouca utilidade. Verificámos também que a diminuição linear do suporte, com valores compreendidos entre 0,05 e 0,2 teve como resultado uma confiança aproximadamente constante, mas com o valor de *lift* a aumentar exponencialmente- destacando-se mais no top 10 de confiança de regras de associação. A título de exemplo, do top 10 correspondente ao *lift* do primeiro quartil de regras, verificámos que o conjunto *time* em $[4; 39, 125[$, não-fumador e *creatinine_phosphokinase* em $[23; 1002, 75[$ implicam o conjunto *anaemia* e *DEATH_EVENT* com suporte 0,06, confiança 0,77 e um *lift* de 3.47 que indica ser uma correspondência do nosso interesse. Por fim, a diminuição do grau de confiança, entre 60% e 100% teve como resultado um *lift* médio aproximadamente constante e um número de regras a aumentar exponencialmente. Em relação ao dataset 2, tal como a normalização, não é necessário aplicar discretização pois todas as variáveis são categóricas (booleanas), no entanto, reduzir o número de variáveis e aumentar o suporte mínimo foram essenciais para determinar os padrões existentes em tempo aceitável. Enquanto que no *dataset* anterior o mínimo de suporte para determinar os padrões era de 0,01, neste o mínimo terá sido de 0,6. Desta forma, foram identificados 847 padrões, em que a representação gráfica do número de padrões em função da diminuição do suporte de 1 a 0,6 corresponde a um crescimento exponencial. Com um mínimo de confiança de 0,8 obtemos 24 133 regras enquanto que para uma confiança de 0,6 temos 25 182 regras, um aumento pouco significativo face à diminuição de confiança. A representação gráfica da diminuição linear do suporte como função do número de regras apresenta-se como uma reta de declive negativo. Em contraste, o número de regras aumenta logaritmicamente com a diminuição linear da confiança. Por último, o gráfico da fig. 6 reflete o *lift* médio em função da confiança. O top 25% corresponde à média de confiança do primeiro quartil de regras.

4.2 Clustering

Neste tópico procuraremos minimizar a distância dentro dos *clusters* (coesão) e maximizar a distância entre os *clusters* (separação). Como métodos recorremos a *Partitioning*, *Hierarchical*, *Density-based* e *Model-based* para ambos os datasets, onde as duas variáveis-alvo foram descartadas. Determinámos ainda os *clusters* antes e após o PCA. Relativamente ao primeiro dataset aplicámos normali-

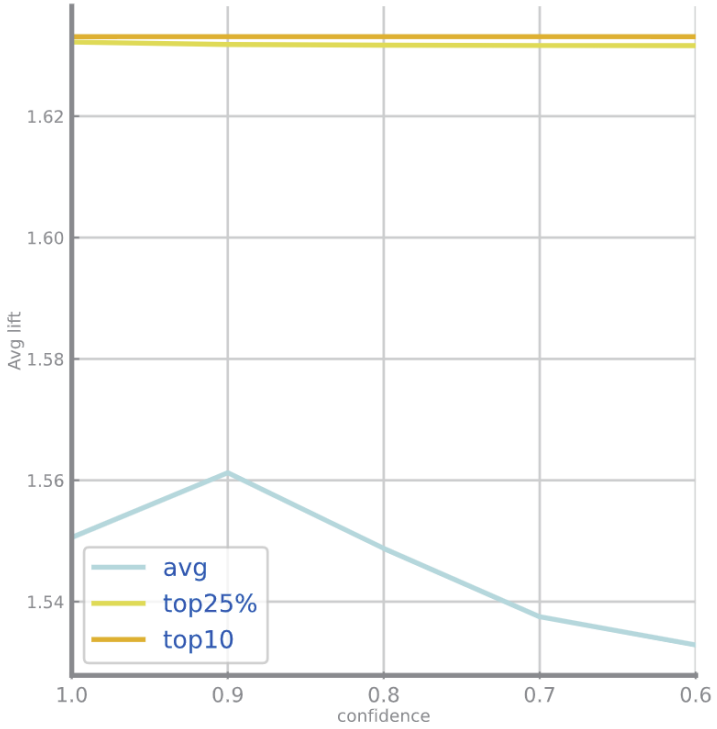

Figura 6: *Variação do lift com a diminuição da confiança*

zação com o objetivo de não distanciar variáveis com um intervalo de valores relativamente elevados, que resultariam numa distinção mais elevada dos pontos por estarem mais dispersos e, por conseguinte, tornando menos eficientes os métodos de *Clustering*. Consideremos as variáveis *age* e *ejection_fraction*. Para o método ***KMeans***, o Erro Quadrático Médio (MSE), como medida de coesão dos *clusters* considerados, decresce exponencialmente com o número k considerado, o número de *clusters*. Tendo em vista o *Princípio da Navalha de Occam* consideramos o ponto de inflexão (método do cotovelo) que considerámos ser quando k=7. Em comparação com Índice de Silhueta, os valores tiveram em média aproximadamente 0.6 com valor máximo quando k=9 (0.63): este valor não é o mais desejável uma vez que não está próximo de 1, mas indica que terá sido escolhido um razoável conjunto de duas variáveis. Para o *DBScan*, *Density-based*, não consideramos o número de *clusters* mas sim a distância máxima que deve haver entre diferentes pontos. O Erro Quadrático Médio é nulo em todos os valores do *Optimal Epsilon* exceto quando igual a 20. Para valores superiores, o MSE descresce ligeira e monotonamente. Por sua vez, como métricas de distância considerámos *Euclideana*, *Cityblock*, *Chebyshev* e *Jaccard*, onde este último obteve o menor erro quadrático médio. Quando utilizado o método *Hierarchical*, obtemos um número ideal de *clusters* quando k=9 com o Índice de Silhueta próximo de 0,6 ainda que elevado, não obteve um desempenho tão bom quanto o do *KMeans*. Das métricas consideradas, *Euclideana*, *Cityblock* e *Chebyshev* são as que presentam maior SC. Por fim, no EM, *Expectation-Maximization*, o Índice de Silhueta torna-se razoável para valores de $k \geq 15$ com máximo de 0.6 quando k=29. Quanto ao segundo dataset, uma vez mais, não foi necessário aplicar *scaling*. Consideremos duas variáveis aleatórias, as de índice 0 e 4 do registo. Para o método ***KMeans***, o Erro Quadrático Médio decresce exponencialmente com o número de *clusters*, no entanto com menor magnitude, isto é, maior dificuldade de interpretação do ponto de inflexão, potencialmente quando k=13. Não obstante, o Índice de Silhueta varia entre 0,01 e 0,04, o que indica um mau resultado de *clustering*. No **DBScan** o desempenho foi idêntico, e, sendo o MSE nulo quando $\epsilon \geq 10$ e Índice de Silhueta diferente de 0 se $\epsilon = 10$. De igual forma, como métricas de distância considerámos *Cityblock*, *Cosine* e *Jaccard*, em que o *Cityblock* obteve um valor ligeiramente inferior em relação aos restantes. Já o mínimo de Índice de Silhueta foi nulo para cada um. O método *Hierarchical* obteve para estas duas variáveis do dataset 2 o melhor desempenho, ainda que distante do ideal, quando consideradas as métricas *Euclideana* e *Cityblock*. Por último, o SC do *Model-based* (EM) apresenta valores em torno de 0,1, concluindo assim que, com os resultados obtidos, não é possível obter uma boa aproximação de *clusters* para as duas variáveis consideradas.

5 Classificação

Após uma primeira iteração por todos os métodos de classificação apresentados em seguida foram retiradas três conclusões essenciais para os resultados aqui apresentados:

1. Os resultados dos algoritmos são muito dependentes dos conjuntos utilizados para treino e teste;

2. Dadas as variáveis alvo para os *datasets 1 e 2* serem o acontecimento de uma morte e a toxicidade de uma substância, respetivamente, é de extrema importância garantir o menor número possível de *Falsos Negativos*, ou seja, situações em que se prevê que uma dada substância não é tóxica quando esta o é ou situações em que se não prevê a morte de alguém apesar de tal acontecer. Dada esta natureza dos dados aqui classificados, mais relevante do que garantir uma *accuracy* elevada, é também garantir uma *precision* e *recall* elevados.

3. Em modelos como *Random Forest* e *Gradient Boosting*, com o aumento da sua complexidade estes tendem a adaptar-se ao ruído presente nos dados perdendo assim a sua capacidade de generalizar a informação (*overfitting*). Este fenómeno foi observável no aumento (significativo) do *training score* e consequente diminuição do *testing score*.

Para resolver ambos estes problemas recorremos a duas estratégias que demonstraram ser de extrema relevância:

1. Os modelos não só foram testados sobre um *validation set* com igual número de entradas de ambas as classes, como também, os resultados aqui apresentados são a média de várias iterações de cada modelo sobre diferentes *validation sets*, escolhidos aleatoriamente;

2. Como métricas de avaliação de um modelo foi usada não só a sua *accuracy* como o seu *F1-Score*, que contempla as duas métricas relevantes anteriormente mencionadas (Precision e Recall).
$$F1\text{-}Score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

3. Guiámos-nos pelo *princípio da Navalha de Occam* para resolvermos os principais problemas de *overfitting*, isto é: se dois modelos apresentassem resultados semelhantes aquele com menor complexidade seria o melhor. Uma estratégia prática para aplicação deste princípio foi, por exemplo, o aumento do número mínimo de entradas por folhas nas árvores (relativamente a *Decision Trees, Random Forest e Gradient Boosting*), impedindo assim que o modelo, para árvores mais profundas, criasse uma folha para cada entrada presente nos nossos dados.

Note-se que os gráficos apresentados ao longo desta secção possuem diferentes escalas para permitir uma melhor interpretação individual.

5.1 Naive Bayes

Foram testados os 3 modelos deste algoritmo de classificação (*Gaussian, Multinomial e Bernoulli*), para classificar ambos os datasets. Para o *dataset 1* os melhores resultados foram obtidos inesperadamente, dado o já pequeno conjunto de registos, recorrendo apenas a *Undersampling* como técnica de pré-processamento de dados. Assim, recorrendo ao **Gaussian Naive Bayes** obtivemos uma *accuracy* de 78.6% e um *F1-score* de 77%. Os resultados recorrendo a *SMOTE* como técnica de balanceamento foram quase tão bons com uma *accuracy* de 77.1% e um *F1-score* de 76.5%, também para o **Gaussian Naive Bayes**.

Para o *dataset 2* os melhores resultados foram obtidos também recorrendo a *Undersampling* como técnica de balanceamento de dados. Porém, desta vez recorrendo ao **Multinomial Naive Bayes**, obtivemos uma *accuracy* de 73.8% e um *F1-score* de 72.7%. É de salientar que ao reduzirmos as *features* do *dataset 2* àquelas que possuiam menos de 95% dos seus valores iguais (466 das 1024 features) a *accuracy* apenas piorou em 2%(71.7%) e o *F1-score* em 1.2%(76.5%).

Como podemos observar na Figura 7, a semelhança entre as métricas obtidas para o training e test sets permite concluir que não existe qualquer tipo de *overfitting* neste modelo.

Figura 7: *Métricas para Training e Test Sets para ambos os Datasets utilizando Naive Bayes*

5.2 KNN

Para este algoritmo de classificação, testou-se ambos os datasets variando o número de vizinhos (números ímpares de 1 a 31) e as medidas de distância (*Manhattan, Chebyshev e Euclidean*). Para o *dataset 1* os melhores resultados foram obtidos recorrendo ao *SMOTE* para balanceamento dos dados, seguindo-se o *scaling* (sendo que ambas as técnica, Z-Score e MinMax, obtiveram os mesmos resultados). Assim, recorrendo ao **KNN** com **27 vizinhos** e utilizando **euclidean** como medida de distância obtivemos uma *accuracy* de 74.2% e um *F1-score* de 73.2%. Para o *dataset 2* os melhores resultados foram obtidos recorrendo a *Undersampling* como técnica de balanceamento de dados. Porém, desta vez com **KNN** com **5 vizinhos** e utilizando **manhattan** como medida de distância. Obtivemos, assim, uma *accuracy* de 77.4% e um *F1-score* de 77.5%. Tal como no *Naive Bayes*, testámos os modelos reduzindo as *features* deste dataset àquelas que possuiam menos de 95% dos seus valores iguais o que levou a que tanto a *accuracy* como o *F1-score* piorassem em menos de 0.5% (para 77% e 77.4%, respetivamente).

Tal como se sucedeu com o modelo anterior, ao observarmos a Figura 8, a semelhança entre as métricas presentes, para o training e test sets, permite novamente concluir que também não existe qualquer tipo de *overfitting* neste modelo.

5.3 Decision Trees

Neste algoritmo de classificação, para ambos os datasets, variaram-se os parâmetros de critério (*gini e entropy*), profundidade máxima da árvore (*2,5,10,15,20 e 25*) e diminuição mínima de impureza (*0.025, 0.01, 0.005, 0.0025 e 0.001*) e mantivemos o número mínimo de folhas a *6*. Para o *dataset 1* os melhores resultados foram obtidos recorrendo ao *SMOTE* para balanceamento dos dados, e sem qualquer tipo de *scaling*. O modelo com melhor resultados foi a **Decision Tree com critério gini,**

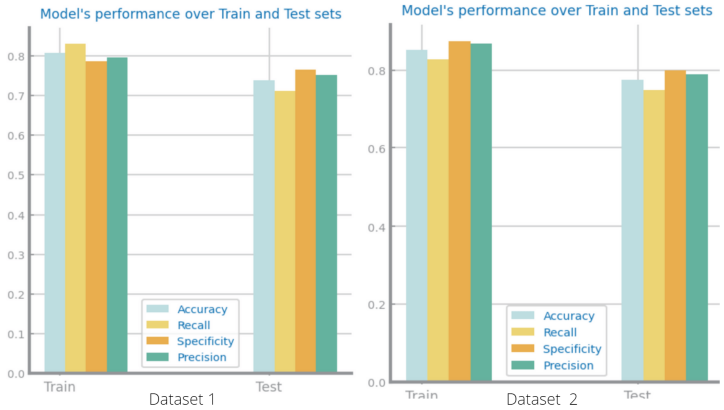

Figura 8: *Métricas para Training e Test Sets para ambos os Datasets utilizando KNN*

profundidade máxima igual a 5 e perda mínima de impureza de 0.025. Com este modelo foi obtida uma *accuracy* de 80.8% e um *F1-Score* de 81.0%. Ainda no *dataset 1*, foi possível observar que quanto maior a perda mínima de impureza (até 0.025) e quanto menor a profundidade máxima da àrvore (até 10), melhores os resultados.

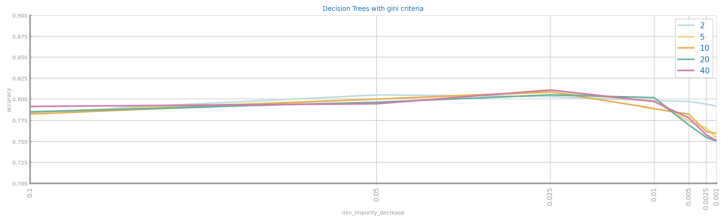

Figura 9: *Accuracy x Perda Mínima de Impureza, para Decision Trees com várias profundidades,para o dataset 1*

Quanto ao *dataset 2* obtivemos uma *accuracy* de 73.0% e um *F1-score* de 72.7% com *Undersampling* dos dados para o modelo de **Decision Tree com critério gini, profundidade máxima igual a 20 e perda mínima de impureza de 0.0025**. Testámos também este modelo recorrendo à redução das *features* deste dataset às 466 com menos de 95% dos valores iguais. Os resultados foram idênticos aos obtidos com todas as 1024 *features* (*accuracy* igual a 72.9% e *F1-score* igual a 72.3%). Neste dataset, observou-se uma melhoria nos resultados com a diminuição da perda mínima de impureza (até 0.0025) e o aumento da profundidade máxima da árvore (até uma profundidade de 20).

Neste modelo, ao fazermos variar a perda mínima de impureza, e profundidade das árvores, foi possível compreender como poderíamos evitar *overitting*, como podemos observar pelas métricas obtidas na Figura 11.

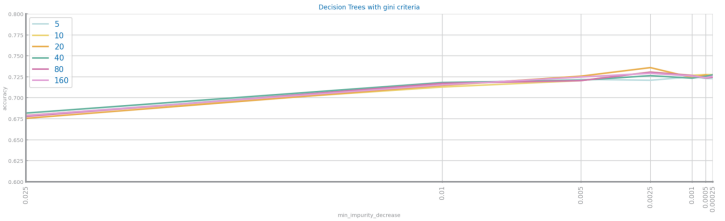

Figura 10: *Accuracy x Perda Mínima de Impureza, para Decision Trees com várias profundidades, para o dataset 2*

Figura 11: *Métricas para Training e Test Sets para ambos os Datasets utilizando Decision Trees*

5.4 Random Forests

Para ambos os datasets, variaram-se os parâmetros de profundidade máxima da árvore (*5,10 e 25*), número de estimadores (*5, 10, 25, 50, 75, 100, 150, 200, 250, 300*) e número de *features* (*0.1, 0.3, 0.5, 0.7, 0.9, 1*). É de salientar que as profundidades máximas obtidas para os melhores resultados são idênticas às obtidas no algoritmo anterior. Quanto a *overfitting*, utilizámos os valores mínimos de perda de impureza obtidos anteriormente para garantirmos que não eram obtidas métricas de treino iguais a 1, ou seja, para evitar que o modelo se adaptasse ao ruído presente nos dados tornando-se assim incapaz de generalizar, o que resultava em métricas de teste sub-ótimas.

Para o *dataset 1* os melhores resultados foram obtidos recorrendo novamente ao *SMOTE* para balanceamento dos dados, e sem qualquer tipo de *scaling*. O modelo com melhor resultados foi a **Random Forest com profundidade máxima igual a 5, 0.5 features, 75 estimadores e perda mínima de impureza de 0.025**. Com este modelo foi obtida uma *accuracy* de 82.1% e um *F1-Score* de 81.0%. Com o aumento da profundidade máxima da àrvore os resultados pioraram. Quanto às features obtemos uma melhoria dos resultados com o seu aumento, até 0.5, voltando a piorar quanto maior o valor acima de 0.5. Por fim, existe uma melhoria significativa na *accuracy* associada ao aumento dos estimado-

res até 75, obtendo-se resultados relativamente constantes daí em diante (ver Fig 12). Dado o poder computacional associado ao aumento de estimadores e seguindo o *Princípio da Navalha de Occam*, considera-se então que 75 estimadores é o valor ótimo para esta classificação.

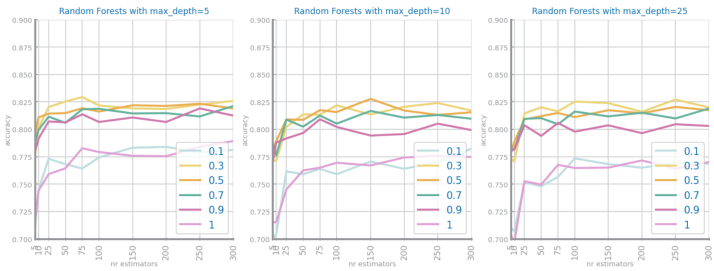

Figura 12: *Accuracy x Número de estimadores, para Random Forests de àrvores com várias profundidades e features, para o dataset 1*

Já no *dataset 2* para o algoritmo de **Random Forest com profundidade máxima igual a 20, tal como obtivemos no algoritmo anterior, 0.5 features, 75 estimadores e perda mínima de impureza de 0.0025** obtivemos uma *accuracy* de 80.0% e um *F1-score* de 78.7% com *Undersampling* dos dados. Testámos também este modelo recorrendo à redução das *features* tal como foi feito anteriormente. Os resultados foram idênticos aos obtidos com todas as 1024 *features* (*accuracy* igual a 78.9% e *F1-score* igual a 77.4%). Com o aumento da profundidade máxima da àrvore os resultados melhoraram, obtendo-se os melhores resultados com 20 features. Na Figura 13, podemos também observar que com a diminuição de features os resultados melhoram, apesar destas já terem sido previamente selecionadas das 1024 iniciais, obtendo-se o melhor resultado com 0.1 features das 466 selecionadas. Por fim, existe uma melhoria significativa na *accuracy* com o aumento dos estimadores até 150, obtendo-se resultados relativamente constantes daí em diante (ver Fig. 13). Novamente, dado o poder computacional associado ao aumento de estimadores, considera-se então que 150 é o valor mais apropriado para esta classificação.

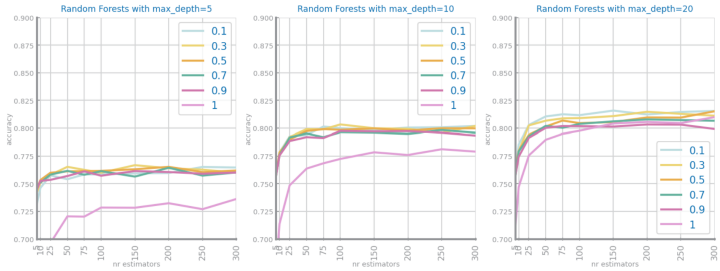

Figura 13: *Accuracy x Número de estimadores, para Random Forests de àrvores com várias profundidades e features, para o dataset 2 com redução de features*

Como podemos observar na Figura 14, a semelhança entre as métricas obtidas para o training e test sets permite concluir que não existe qualquer tipo de *overfitting* neste modelo.

Figura 14: *Métricas para Training e Test Sets para ambos os Datasets utilizando Random Forest*

5.5 Gradient Boosting

Para ambos os datasets, variaram-se os mesmos parâmetros que no algoritmo das *Random Forests*, à excepção do número de features, variando-se neste caso o *learning rate* para os mesmos valores (*0.1, 0.3, 0.5, 0.7, 0.9*). É de salientar que no primeiro dataset os melhores resultados foram obtidos com profundidades máximas idênticas às obtidas nos algoritmos de *Decision Trees* e *Random Forests*.

Para evitar *overfitting* dos dados neste modelo, recorremos ao número mínimo de entradas por folha impedindo assim o modelo de criar ramos e folhas demasiado específicos (adaptados ao ruído).

Para o *dataset 1* as técnicas de *scaling* não levaram a uma melhoria nos resultados, contudo todas as técnicas de balanceamento levaram a uma *accuracy* mais elevada. No entanto, a melhor entre estas foi a de *Undersampling* que, apesar de ter uma *accuracy* idêntica às restantes, obteve um *F1-Score* significativamente maior, entre 10%-15% acima das restantes(*OverSampling* e *SMOTE*). Assim, o modelo com melhores resultados foi o **Grandient Boosting com profundidade máxima igual a 5, 100 estimadores, learning rate igual a 0.5, perda mínima de impureza de 0.025 e número mínimo de entradas por folha de 40**. Com este modelo foi obtida uma *accuracy* de 80.9% e um *F1-Score* de 80.3%.

Tal como podemos observar na Figura 15, o desempenho do algoritmo varia de forma idêntica ao que obtivemos no algoritmo de *Random Forests*. Neste caso obtemos uma *accuracy* significativamente pior com o aumento dos estimadores.

Já no *dataset 2*, utilizando *Undersampling*, o modelo com melhores resultados foi o **Gradient Boosting com**

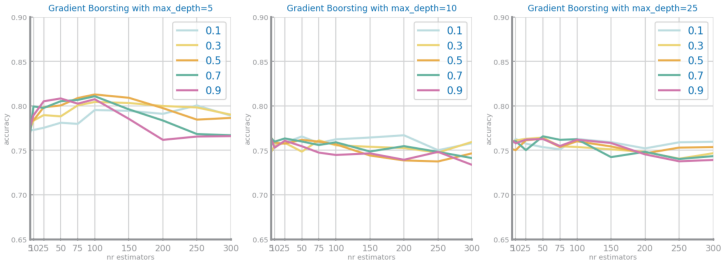

Figura 15: *Accuracy x Número de estimadores, para Gradient Boosting com àrvores de várias profundidades e learning rate, para o dataset 2 com redução de features*

profundidade máxima igual a 10, 150 estimadores, learning rate de 0.5 e número mínimo de entradas por folha de 150. Com este modelo foi obtida uma *accuracy* de 77.5% e um *F1-Score* de 77.5%. Também com este modelo, os resultados obtidos usando apenas as *features* com menos de 95% dos seus valores idênticos foram semelhantes aos obtidos com todas as 1024 *features*. Usando apenas 466 *features*, obtivemos uma *accuracy* e *F1-Score* 0.4% piores (ambos iguais a 77.1%).

Tal como podemos observar na Figura 16, o melhor desempenho é obtido quando a profundidade máxima das árvores é 10, sendo que este melhora com o aumento do learning rate, até 0.5, piorando com o seu aumento daí em diante até 1. Contrariamente ao obtido com Random Forests, o desempenho deste modelo piora significativamente quando aumentamos a profundidade máxima das àrvores para 20.

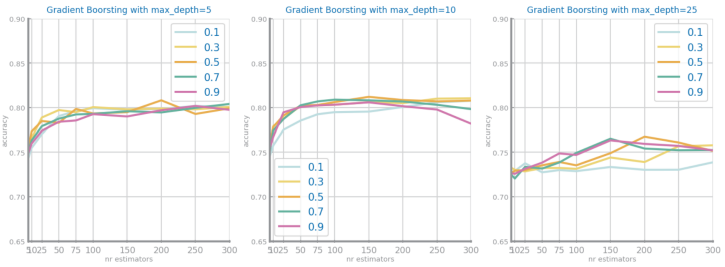

Figura 16: *Accuracy x Número de estimadores, para Gradient Boosting com àrvores de várias profundidades e learning rate, para o dataset 2 com redução de features*

Comparativamente aos restantes, este modelo, dado o seu elevado número de parâmetros e fácil adaptação aos dados, revelou-se mais complicado identificar os parâmetros ótimos para este modelo. Caso utilizássemos um número mínimo de entradas por folha demasiado baixo, rapidamente este modelo se encontrava em *overfitting*. Por outro lado, caso escolhêssemos um número mínimo de entradas por folha demasiado elevado, facilmente nos deparávamos com o modelo em *underfitting* - o modelo não se adaptava o suficiente aos dados. Isto foi observável ao aumentarmos o número mínimo de entradas por folha e, com este aumento, surgir uma diminuição tanto das métricas do conjunto de treino como do conjunto de teste. Com os números mínimos de entradas por folha escolhidos, obtivemos então o ponto ótimo entre *underfitting* e *overfitting* (ver Figura 17).

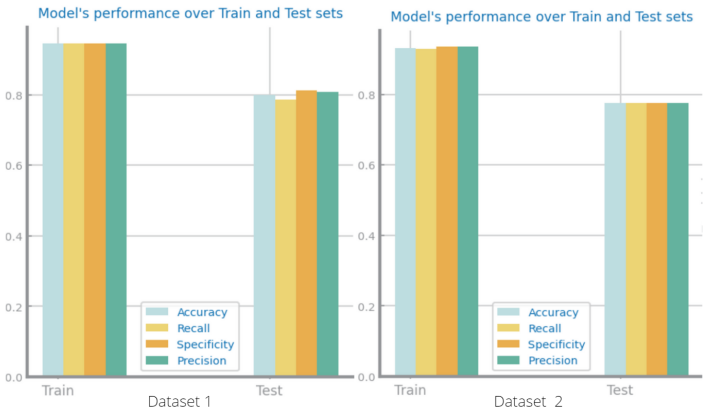

Figura 17: *Métricas para Training e Test Sets para ambos os Datasets utilizando Gradient Boosting*

5.6 Avaliação dos Resultados da Classificação

Após a aplicação dos vários métodos de classificação, é possível retirarmos diversas conclusões acerca dos nossos dados e das formas mais eficazes de os classificarmos.

No *dataset 1*, algo muito relevante é que à excepção do algoritmo de *Gradient Boosting*, todos os restantes obtiveram os melhores resultados utilizando *SMOTE* como técnica de balanceamento. Algo que era expectável dado o pequeno conjunto de dados que possuímos e a sua grande desproporcionalidade (aproximadamente 2 negativos para cada 1 positivo). Outro resultado relevante foi o KNN ser o único modelo que beneficiou com ambas as técnicas de *scaling* das variáveis pois tal permitiu retirar o peso associado a certas variáveis (como as plaquetas) apenas pela sua escala ser diferente. Na Figura 18 podemos observar o de-

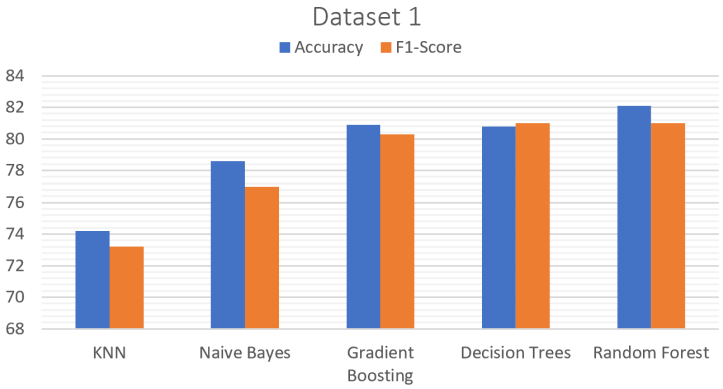

Figura 18: *Resultados dos vários classificadores, quando utilizados para modelar o dataset 1*

sempenho relativo dos vários classificadores para o *dataset 1* e perceber assim, que o método com o qual conseguimos melhores resultados para este conjunto de dados foi com as **Random Forests** com **profundidade máxima de 5** (pois profundidades superiores levaram a *overfitting* do modelo, **0.5 features e 75 estimadores**, observável na Figura 19.

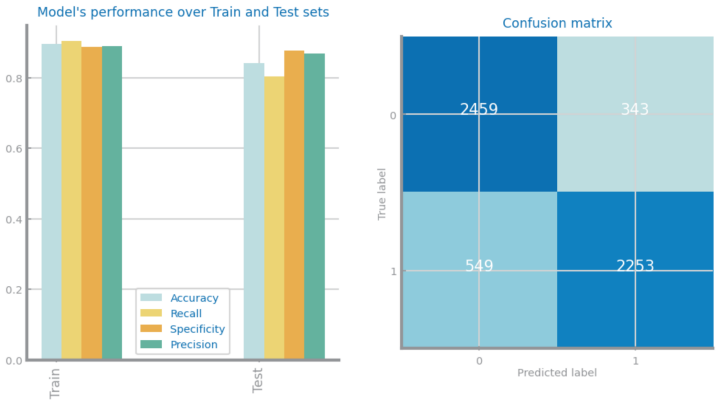

Figura 19: *Resultados de várias iterações de Random Forests com profundidade máxima de 5, 0.5 features, 75 estimadores e perda mínima de impureza de 0.025*

No *dataset 2*, foi relevante perceber que os resultados obtidos com estes classificadores foram idênticos tanto quando utilizavamos as 1024 *features* como quando utilizámos apenas as 466 *features* com menos de 95%. Com isto podemos inferir que as restantes 558 *features* são de pouca relevância para determinar a toxicidade das substâncias (variável alvo). Ao compararmos os resultados dos diversos métodos de classificação podemos perceber que aqueles com melhor capacidade de modelar os dados são o *Random Forest* e *Gradient Boosting* com as parametrizações previamente mencionadas. Ambos obtiveram *accuracy* e *F1-Score* semelhantes, contudo o *Gradient Boosting* obteve um Recall 5.6% acima do algoritmo de *Random Forest*. Como mencionámos no inicio desta secção um dos principais objetivos dado o cariz dos dados é reduzir o número de *Falsos Negativos* e, por isso, achámos que o **Gradient Boosting com profundidade máxima igual a 10, 150 estimadores, learning rate de 0.5 e número mínimo de entradas por folha de 150** seria o melhor modelo a aplicar a este conjunto de dados, como podemos observar na Figura 20.

6 Avaliação e análise crítica

Ao desenvolvermos este trabalho obtivemos bastante experiência prática, uma vez que aplicamos os modelos abordados durante as aulas teóricas.
O nosso grupo considera que desenvolveu um bom trabalho, uma vez que este foi desenvolvido de forma consistente, aplicando sempre as sugestões recebidas durante as aulas laboratoriais.

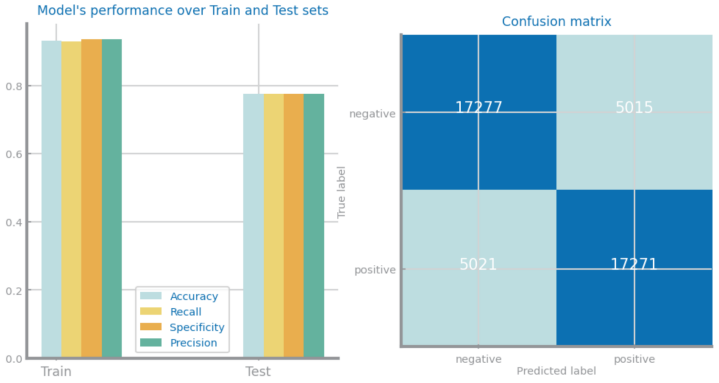

Figura 20: *Resultados de várias iterações de Gradient Boosting com profundidade máxima igual a 10, 150 estimadores, learning rate de 0.5 e número mínimo de entradas por folha de 150*

Com este trabalho percebemos que não é possível afirmar que um modelo é sempre melhor que o outro, uma vez que os resultados obtidos por esse dependem de vários factores, tais como, o tipo de dados, a dispersão, a quantidade, a qualidade, entre outros, sendo assim, antes de aplicar um modelo é necessário perceber as características dos nossos dados. Perante os resultados obtidos e percebendo a importância que uma boa classificação têm neste tipo de dados sensíveis, consideramos que valores aceitáveis de *recall* teriam que estar dentro do intervalo [98.0%,100.0%].

7 Estratégias de melhoria

Neste projecto tal como em todos os projectos de aprendizagem existe sempre espaço de melhoria uma vez que todo este é constituído por um fluxo continuo de aprendizagem. Durante o desenvolvimento do projecto a métrica de performance à qual demos primazia foi a *accuracy*, sendo assim, grande parte dos testes que efectuámos, tinham o objetivo de obter o melhor valor possível para esta métrica. Para trabalhos futuros, sugerimos, fazer o mesmo processo, no entanto para o *recall*, de forma para a avaliar os resultados com a variação dos parâmetros de cada método. Embora se tenha aplicado as técnicas de ANOVA para o primeiro dataset e χ^2 e *Mutual Information* para o segundo, é possível obter resultados interessantes ao intercalar diferentes métodos, tais como interpretar a variável de *output* como numérica, ou seja, utilizar os métodos Pearson e Kendall aos datasets. De facto, experimentámos para o laboratório correspondente com tais métodos embora não se tenha chegado a resultado algum percetivelmente superior. Na análise de *clusters*, pode ser considerada outras medidas de distância intra e inter-clusters, tais como o Índice de *Dunn* e o Índice de *Davies-Bouldin*.