Universidade do Minho

Mestrado em Engenharia Informática

Dados e Aprendizagem Automática

Grupo 5

Conceção de Modelos de Aprendizagem

David Duarte (pg50315)

Joana Alves (pg50457)

Tânia Teixeira (A89613)

Vicente Moreira (pg50799)

Janeiro 2023

Conteúdo

1 Introdução

Este trabalho prático foi realizado no âmbito da unidade curricular de Dados e Aprendizagem Automática e tem como objetivo aprofundar o conhecimento na matéria lecionada ao longo do semestre, através do desenvolvimento de dois modelos inteligentes capazes de prever e gerar conhecimento através da informação disponibilizada.

Este relatório contém todos os objetivos, decisões e problemas encontrados na exploração e tratamento de dois *data sets* distintos, assim como o desenvolvimento dos modelos de previsão. Para este modelos, utilizamos uma abordagem de aprendizagem por supervisão visto que ambos os *data sets* de teste utilizados contêm informação sobre os resultados pretendidos.

O primeiro *data set*, fornecido pelos docentes, trata acidentes rodoviários na cidade de Guimarães, apresentando informação sobre a cidade, o *timestamp* associado ao registo, magnitude do atraso, o atraso em segundos, as estradas afetadas pelo incidente, o nível de luminosidade, o valor médio de temperatura, o valor médio de pressão atmosférica, entre outros, sendo o objetivo deste desenvolver um modelo de aprendizagem utilizando técnicas de **classificação** de modo a prever o nível de incidentes rodoviários numa determinada hora na cidade.

O segundo *data set*, escolhido pelo grupo, incide sobre músicas. Possui os atributos nome, popularidade, duração, acústica, dançabilidade e energia da música, instrumental, vivacidade, entre outros. Para a construção do modelo deste problema, pretendemos utilizar e explorar várias técnicas de **regressão**, prevendo a popularidade de cada música de acordo com os seus diferentes atributos.

1.1 Metodologia Utilizada

Para a realização deste trabalho prático, o grupo de trabalho optou por seguir a metodologia **SEMMA** (*Sample, Explore, Modify, Model, Assess*), dada a importância da sua utilização em contextos de desenvolvimento de modelos de aprendizagem, de modo a efetuar um melhor planeamento na execução do projeto, assim como permitir a sua replicação entre os dois *data sets*. Ao longo deste relatório será possível verificar o trabalho desenvolvido em cada uma das fases: desde a análise e compreensão dos dados até à avaliação dos modelos.

2 Crash DataSet

2.1 Contexto e Introdução

O *data set* fornecido pelos docentes tem como objetivo a construção de um modelo de **classificação**. Este incide sobre a recolha de informações dos incidentes de trânsito na cidade de Guimarães durante o ano de 2021. Este *data set* é constituído por 13 *features*.

2.1.1 Objetivo do Modelo

Para este *data set*, decidimos que seria relevante criar um modelo de *Machine Learning* no qual, através da informação de cada registo, fosse capaz de prever o nível de incidentes rodoviários, numa determinada hora, na cidade de Guimarães.

2.2 Atributos do *Data Set*

O *data set* possui as seguintes *features* :

Atributos *Data Set* "Crash DataSet"		
Atributo (Antes)	**Tipo de Dado**	**Descrição**
city_name	Object	Nome da cidade
record_date	Object	Timestamp associado ao registo
magnitude_of_delay	Int64	Magnitude do atraso
delay_in_seconds	Object	Atraso em segundos
affected_roads	Object	Estradas afetadas pelos incidentes
luminosity	Object	Nível de luminosidade
avg_temperature	Float64	Valor médio da temperatura
avg_atm_pressure	Float64	Valor médio da pressão atmosférica
avg_humidity	Float64	Valor médio de humidade
avg_wind_speed	Float64	Valor médio da velocidade do vento
avg_precipitation	Float64	Valor médio de precipitação
avg_rain	Object	Avaliação qualitativa do nível de precipitação
incidents	Object	Indicação acerca do nível de incidentes rodoviários

2.3 Exploração e Visualização dos Dados

Antes de efetuar o tratamento dos dados, é necessário fazer a exploração dos mesmos. A exploração e visualização destes ajuda a entender melhor o conjunto de dados, as relações entre as variáveis e como elas podem afetar o desempenho do modelo. Desta forma, ajuda-nos a identificar problemas como dados em falta ou até mesmo *outliers*, e que medidas podemos tomar para os corrigir. Além disso, a exploração de dados também pode ajudar a selecionar as melhores *features* para incluir no modelo e ajustar os hiperparâmetros de forma mais informada. Em geral, a exploração de dados é uma etapa importante no processo de construção de modelos de *machine learning* e pode ajudar a garantir que o modelo seja tão preciso e robusto quanto possível.

- **Missing Values:** Recolhemos um gráfico de *missing values* para análise dos mesmos, onde verificamos que apenas existem *missing values* na feature *affected_roads*.
- **Matriz de Correlação:** Obtivemos uma matriz de correlação para uma melhor compreensão das relações entre os dados, denotando uma forte relação entre *avg_temperature* e *avg_humidity* (-0.55).
- **Outliers:** Avaliamos os *outliers* dos dados numéricos presentes no *data set*, sendo possível visualizar anormalidades nos dados no atributo *delay_in_seconds*.
- **Entradas Duplicadas:** Apuramos a existência de entradas repetidas, não sendo encontradas repetições nos dados.

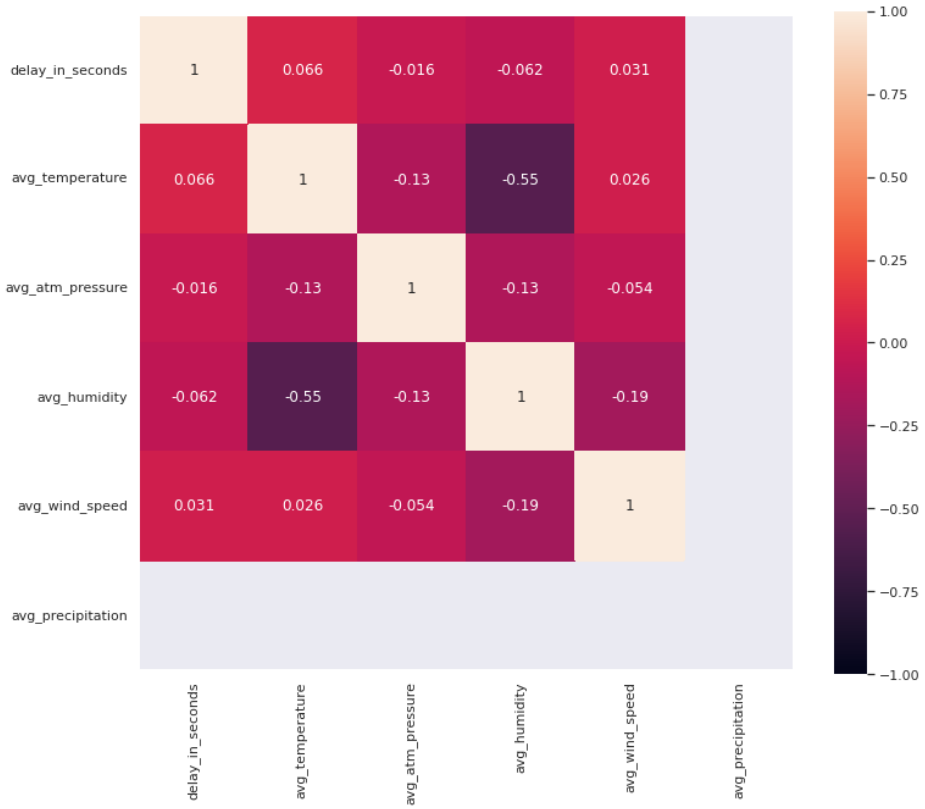

Figura 1: Matriz de Correlação

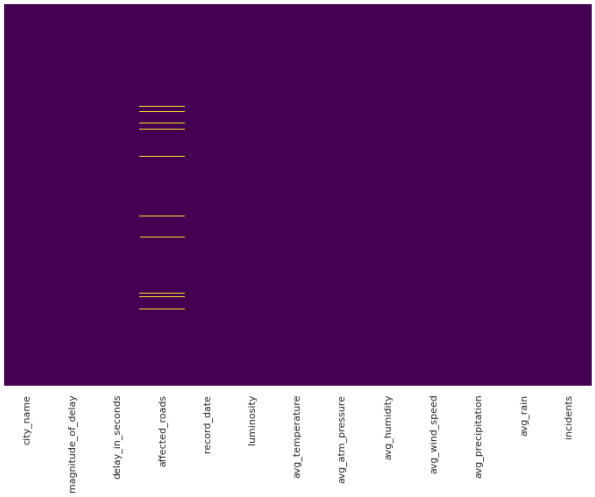

Figura 2: *Missing Values*

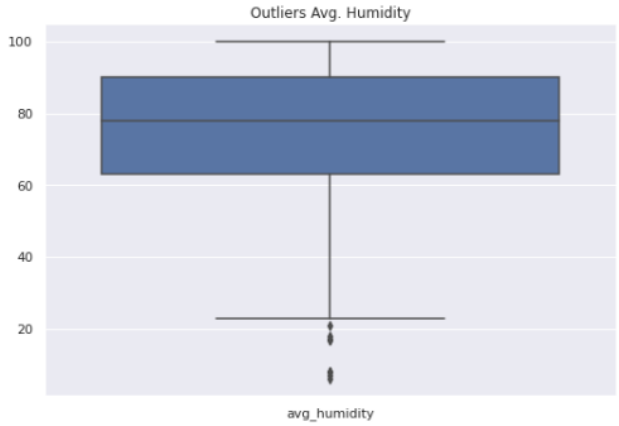

Figura 3: *Outlier - Humidity.*

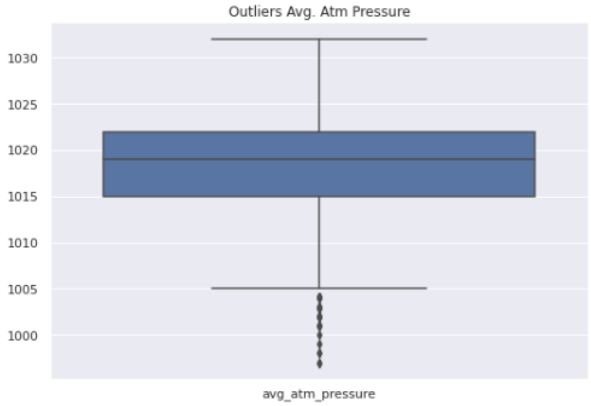

Figura 4: *Outlier - Pressure.*

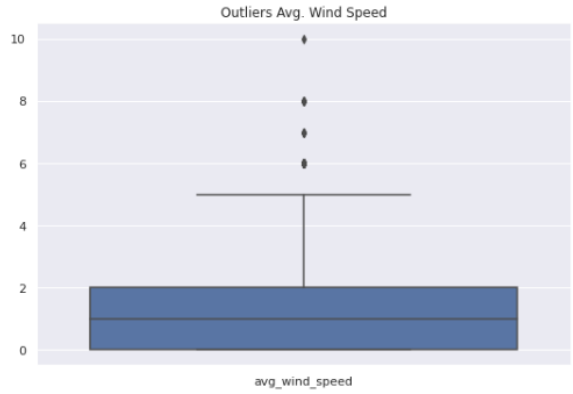

Figura 5: *Outlier - Wind Speed.*

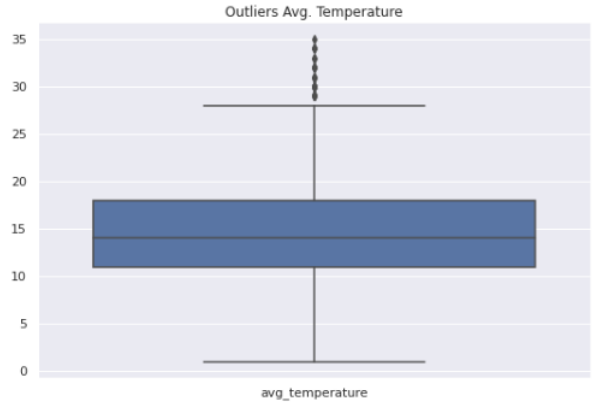

Figura 6: *Outlier - Temperature.*

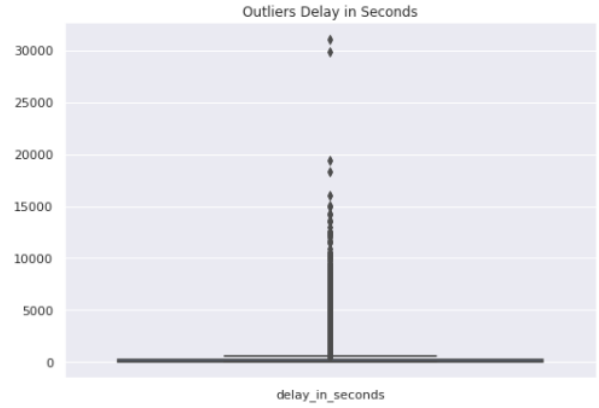

Figura 7: *Outlier - Magnitude do Delay.*

2.4 Tratamento de Dados

Depois de devidamente analisado, começamos por tratar o *data set*, com o objetivo de extrair informação adicional, assim como consertar quaisquer problemas presentes nos dados, detetados na fase anterior. Assim, apresentamos os vários tratamentos que foram realizados de modo a melhorar a qualidade dos nossos dados e que achamos que irão garantir que o nosso modelo seja o mais robusto e preciso quanto possível.

2.4.1 Remoção de Colunas Redundantes

Após a análise das *features* do *data set* verificamos que todos os incidentes se referem à cidade de Guimarães, desta forma removemos a coluna *city_name* dado o seu valor ser sempre referente à mesma cidade. Também foi efetuada a remoção da coluna (*avg_precipitation*) que diz respeito à precipitação média, uma vez que os valores deste atributo se encontravam todos a 0 o que não traria qualquer tipo de conhecimento pertinente ao nosso modelo.

2.4.2 Extração de Novas Features

Através da observação da *feature affected_roads* verificamos que existe uma repetição das estradas afetadas. De modo a extrair algum conhecimento disto optamos por fazer a contabilização do número estradas indicadas (incluindo as repetidas) e adicionamos uma nova coluna que contabiliza o número de incidentes, dando-lhe o nome de *number_of_affected_roads*. Isto é: cada indicação de estrada afetada simboliza um acidente decorrido naquele tempo.

Também para este atributo, após a contagem, decidimos remover os valores duplicados visto que estes já se refletem no novo atributo e procedemos à criação de uma nova *feature unique_roads_affected* que contabiliza as diferentes estradas afetadas, sem contagem das repetidas.

2.4.3 Feature Engineering

Através do atributo *record_date*, efetuamos a extração para colunas separadas o mês, hora, dia da semana e a informação se este se tratava de um feriado ou não, tendo em consideração que o movimento nas estradas varia consoante estes fatores, sendo algo pertinente para o nosso modelo.

Para a atribuição dos dias da semana, assim como se este é feriado, utilizamos as bibliotecas *'datetime'* e *'holidays'*. Percorrendo a lista de feriados nacionais, pudemos extrair esta nova informação que será implementada para uma melhor aprendizagem do nosso modelo.

2.4.4 Label Encoding

Uma vez que os atributos *avg_rain* e *luminosity* se tratavam de valores na forma de *string*, optamos por utilizar a técnica de *Label Encoding* de modo a fazer corresponder a um atributo numérico ordinal. Para isso, efetuamos previamente a avaliação de todos os valores possíveis para cada *feature* e, posteriormente fizemos um dicionário de tradução dos valores, que aplicamos no *data set*.

No atributo *avg_rain*:

- Sem Chuva: 0,
- chuva moderada: 1,
- chuva fraca: 2,
- chuva forte: 3

No atributo *luminosity*:

- DARK: 0,
- LOW_LIGHT: 1,
- LIGHT: 2

2.5 Modelos de Aprendizagem

Para a próxima fase, decidimos criar modelos de aprendizagem automática de classificação utilizando vários algoritmos de aprendizagem lecionados nas aulas teóricas e práticas, com o objetivo de encontrar um modelo que aprenda da forma mais precisa o domínio do problema, sendo capaz de prever as taxas de incidentes com os dados fornecidos.

Para isso começamos por efetuar a divisão do *data set*, separando os atributos a serem analisados pelo modelo e o atributo alvo. O *data set* de treino, como já foi fornecido em separado, não necessitou desta divisão. Para cada modelo aplicado, incluímos dois valores de *accuracy*:

- **Avaliação Parcial:** Valor de *accuracy* obtido inicialmente, resulta da submissão efetuada na plataforma *Kaggle* antes do término da competição, sendo este o meio principal de avaliação de um modelo. Este apenas representa 30% dos dados de teste.
- **Avaliação Completa:** Resultado de *accuracy* obtido no final da competição.

2.5.1 Decision Tree

Começamos por criar um modelo mais básico utilizando uma *Decision Tree Classifier*. Visto que este não aceita atributos categóricos, quer no seu treino como no seu teste, removemos estes temporariamente do *data set*. O valor de *accuracy* obtido foi de cerca de 62%.

Após efetuarmos o *Label Encoding* assim como a introdução das novas *features* (extraídas a partir da data), refizemos o nosso modelo, obtendo assim um modelo de aprendizagem mais preciso com os seguintes valores de *accuracy*:

- **Avaliação Parcial:** 86.15%
- **Avaliação Completa:** 88.64%

2.5.2 Random Forest - Simples

De seguida decidimos utilizar o algoritmo de *Random Forest Classifier*. Este algoritmo é geralmente considerado como um modelo mais eficaz do que uma árvore de decisão simples, uma vez que combina várias árvores de decisão, o que fornece uma maior estabilidade e precisão para além de ajudar a reduzir o *overfitting*. Com o modelo de aprendizagem treinado, efetuamos testes, obtendo os seguintes valores de *accuracy*:

- **Avaliação Parcial:** 87.8%
- **Avaliação Completa:** 90.77%

2.5.3 Random Forest - HyperParameter

Decidimos aplicar técnicas de *HyperParameter* no algoritmo anterior. Utilizando diferentes valores de n^o *estimators*, *max_depth*, *max_features*, entre outros, podemos gerar um modelo de aprendizagem otimizado, evitando problemas de *overfitting*. Com este método obtemos um modelo com os seguintes valores de *accuracy*:

- **Avaliação Parcial:** 83.65%
- **Avaliação Completa:** 86.62%

2.5.4 Support Vector Machine

Decidimos utilizar o modelo *Support Vector Machine* uma vez que este é usado principalmente para problemas de classificação. Apesar de ser um algoritmo poderoso, tende a ser sensível aos valores dos parâmetros e pode ser afetado por ruído ou *outliers*. No entanto, este modelo foi o que obteve os piores resultados, não sendo surpreendente uma vez que o *data set* tende a apresentar alguns ruídos, exigindo um tratamento de dados mais minucioso.

- **Avaliação Parcial:** 52.07%
- **Avaliação Completa:** 56.33%

2.5.5 XGBoost

Por último, utilizamos um algoritmo de *XGBoost*, algoritmo este que consiste em adicionar sucessivamente modelos fracos para melhorar o desempenho geral do modelo. Foi o algoritmo escolhido como final uma vez que é um algoritmo de aprendizagem altamente eficaz e escalável, com capacidade para lidar com grandes conjuntos de dados tanto para problemas de classificação como de regressão. Com este método obtemos um modelo com os seguintes valores de *accuracy*:

- **Avaliação Parcial:** 87.53%
- **Avaliação Completa:** 91.12%

2.6 Resultados e Conclusões

Após a análise de todos os resultados apresentados pelos modelos acima referidos, verificamos que o modelo que apresenta melhor desempenho para os resultados pretendidos é o *XgBoost*, embora as percentagens de *accuracy* entre os modelos desenvolvidos não apresentem uma grande discrepância, este sem dúvida é o que apresenta melhor previsão sem estar *overfitted*.

Apresentamos de seguida um gráfico representativo dos valores de precisão de todos os modelos:

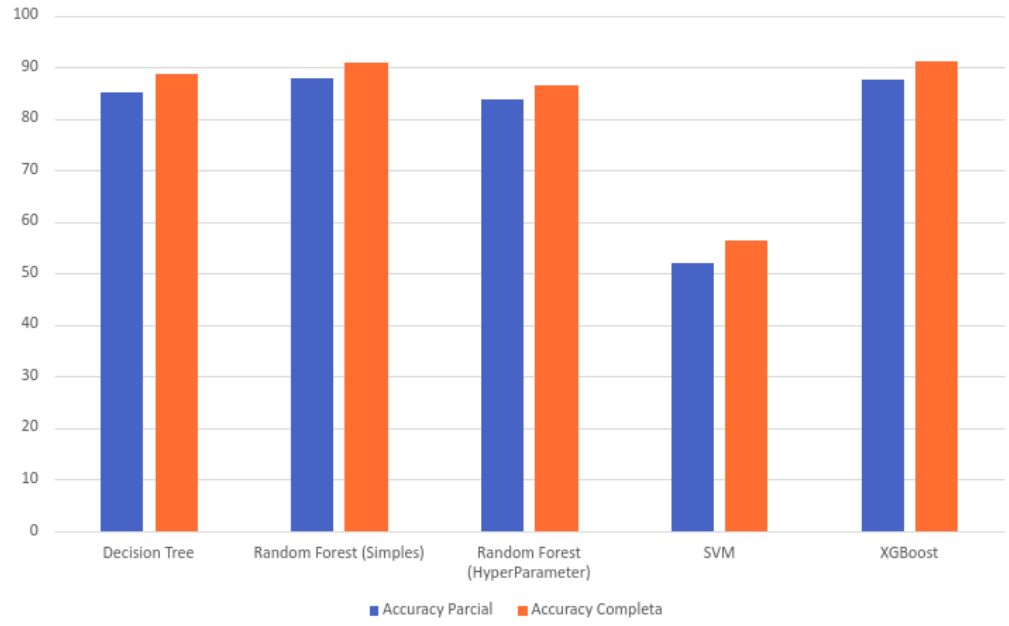

Figura 8: *Valores de Accuracy Finais*

3 Song Popularity

3.1 Contexto e Introdução

Visto que o *data set* anterior (fornecido pelos docentes) tem como objetivo desenvolver um modelo de classificação, o grupo decidiu procurar um *data set* que tivesse como objetivo um modelo de **regressão**, de forma a diversificar os casos de estudo.

O grupo escolheu um *data set* designado por "Song Popularity". Este oferece informação sobre músicas, tal como a sua popularidade, duração, acústica, dançabilidade, energia, entre outros. Este *data set* é constituído por 15 *features*, sendo a maior parte do tipo *float64* e *int64*, com a exceção do nome da música, sendo este um atributo nominal.

3.1.1 Objetivo do Modelo

O nosso principal objetivo, definido pelo enunciado, será criar um modelo capaz de prever a popularidade de uma música, utilizando os restantes atributos como auxílio.

3.2 Atributos do *Data Set*

Atributos *Data Set* "Song Popularity"		
Atributo	**Tipo de Dado**	**Descrição**
song_name	Object	Nome da Música
song_popularity	Int64	Popularidade da Música
song_duration_ms	Int64	Duração da Música em milissegundos
acousticness	Float64	Nível de Acústica da Música
danceability	Float64	Nível de Dançabilidade da Música
energy	Float64	Nível de Energia da Música
instrumentalness	Float64	Nível de Instrumental
key	Int64	Tom da Música
liveness	Float64	Vivacidade da Música
loudness	Float64	Sonoridade da Música
audio_mode	Int64	Modo de Áudio da Música
speechiness	Float64	Quantidade de Voz
tempo	Float64	Ritmo da Música
time_signature	Int64	Fórmula de Compasso da Música
audio_valence	Float64	Valência de Áudio da Música

3.3 Exploração e Visualização dos Dados

O primeiro passo tomado na avaliação do *data set* foi a exploração deste, ou seja, desenvolveu-se uma exploração inicial dos dados recebidos para que depois se conseguisse analisar corretamente estes. Sendo assim, preparamos os dados, através de alguns comandos *python*, para que fosse possível obter um bom tratamento dos dados fornecidos a serem aplicados nos modelos de regressão.

- **Distribuição dos Atributos:** Para a análise da distribuição dos valores dos diversos atributos, recorremos ao método *histplot* do módulo *seaborn*.
- **Missing Values:** Recolhemos um gráfico de *missing values* onde reparamos na ausência destes e portanto concluímos que não teremos de fazer qualquer tratamento específico para este problema.
- **Matriz de Correlação:** Utilizamos o método *heatmap* do módulo *seaborn* para obter uma matriz de correlação para uma melhor compreensão das relações entre os dados.
- **Análise de Duplicados:** Através do método *duplicated* obtivemos o número de linhas duplicadas, sendo estas no total 3909 linhas.
- **Outliers:** Visualizamos os *outliers* de todas as *features* não categóricas de modo a perceber a sua distribuição.

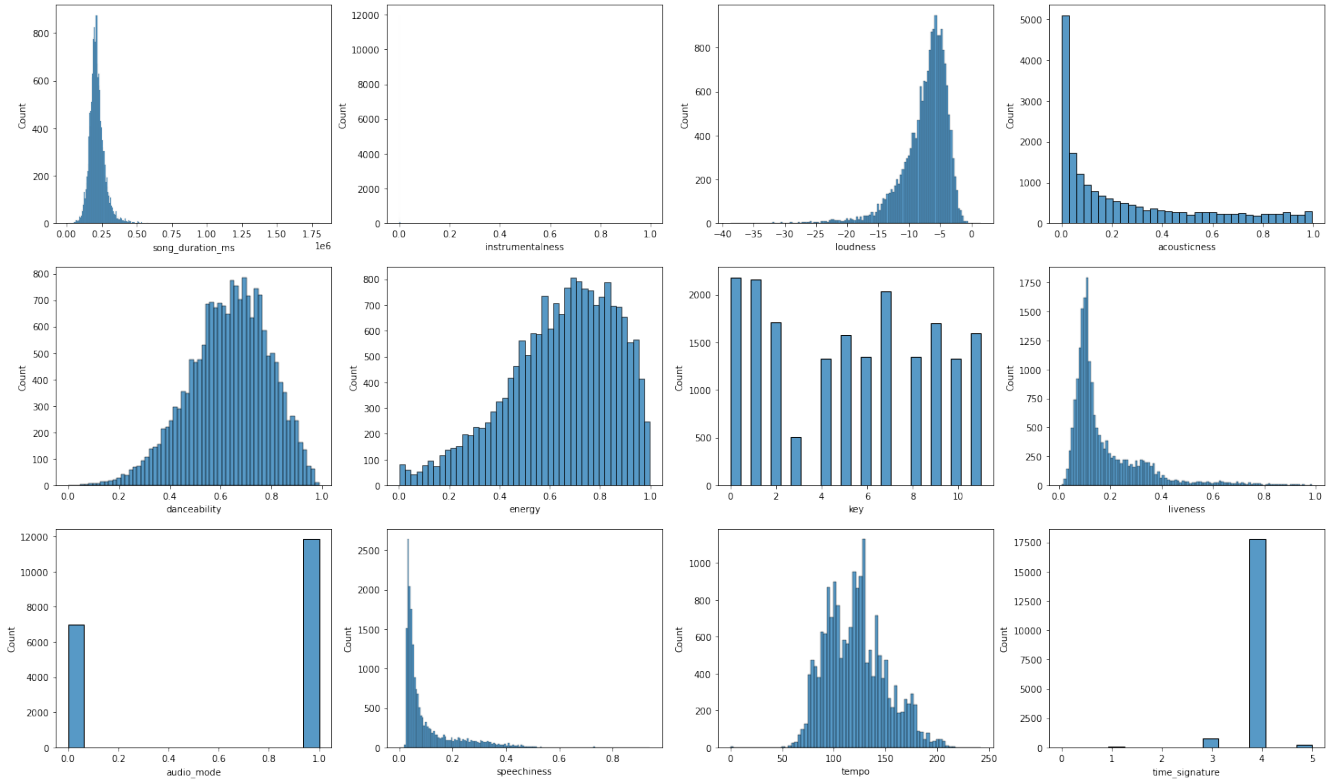

Figura 9: Distribuição dos Atributos

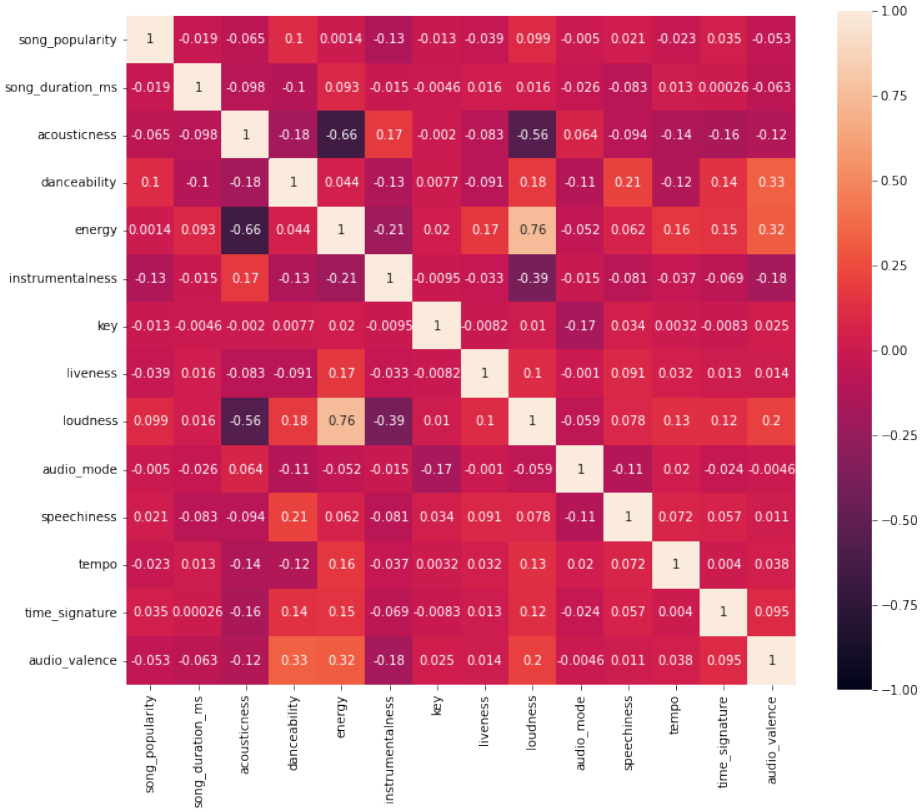

Figura 10: Matriz de Correlação

Pela observação da Figura 10, podemos encontrar algumas correlações pertinentes entre os dados, como, por exemplo, a correlação entre o nível de acústica (*acousticness*) e a energia da música (*energy*), mostrando que estes estão interligados. Observamos, ainda, que a energia (*energy*) está também fortemente relacionada com a sonoridade da música (*loudness*). Esta exploração deu-nos confiança no potencial da capacidade de previsão do modelo a ser desenvolvido para este problema.

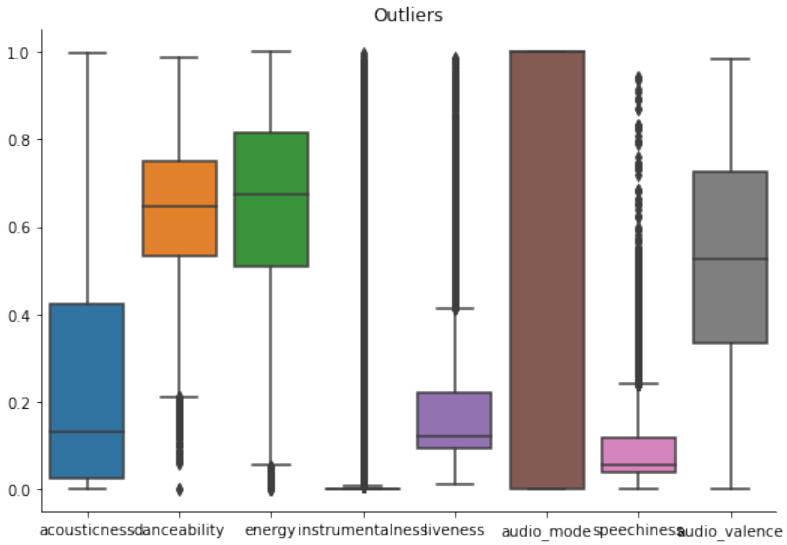

Figura 11: *Outliers*

3.4 Tratamento de Dados

Depois de devidamente analisado, começamos por tratar o *data set*, com o objetivo de extrair informação adicional, assim como consertar quaisquer problemas presentes nos dados. Assim, apresentamos os vários tratamentos realizados de modo a melhorar a qualidade dos nossos dados.

- **Remoção de Linhas Duplicadas:** Começamos por remover as 3909 linhas duplicadas que encontramos no *data set* de modo a não sobrecarregar o mesmo com informação desnecessária.
- **Remoção de Colunas Redundantes:** Retiramos a coluna *song_name* uma vez que esta não apresenta informação relevante acerca da popularidade da música, servindo apenas como um tipo de identificador. Decidimos remover também as colunas *instrumentalness*, *audio_mode*, *key* e *time_signature*.
- **Normalização do Atributo *loudness*:** A visualização e exploração dos dados permitiu a observação da distribuição anormal dos valores da *feature loudness*. Assim, aplicamos a técnica de normalização sobre os mesmos, de modo a obter valores mais uniformizados.
- **Tratamento de *Outliers*:** De seguida, passamos para o tratamento de *outliers*, tendo sido decidido remover os mesmos, aplicando, no final, normalização aos valores restantes. Assim foram tratadas as seguintes *features*: *loudness*, *liveness* e *speechiness*.
- **Alteração da Escala:** Por último, alteramos a escala da duração da música de milissegundos para segundos.

3.5 Modelos Utilizados

Nesta fase decidimos aplicar modelos de aprendizagem automática de regressão, utilizando, para isso, vários algoritmos de aprendizagem, com o objetivo de encontrar um modelo que apresente da forma mais precisa a popularidade das músicas através dos dados fornecidos. Para isso começamos por dividir o *data set* separando os atributos a serem analisados pelo modelo do atributo alvo (*target*), estando estes armazenados em variáveis distintas. De seguida, dividimos novamente o *data set*, no entanto, desta vez dividimos o mesmo em dados de treino e dados de teste, verificando que o conteúdo do *target* se encontrava bem distribuído por ambos.

3.5.1 Simple Linear Regression

Para o primeiro modelo a ser testado escolhemos o *Simple Linear Regression*, um modelo de regressão linear que, após a sua aplicação, obteve os seguintes resultados:

- **MAE:** 16.059
- **RMSE:** 411.247

3.5.2 Random Forest Regressor

De seguida, aplicamos o modelo de *Random Forest Regressor*, tendo este obtido resultados muito parecidos ao anterior. Passamos a apresentar os valores obtidos, assim como um gráfico representativo das previsões obtidas (verde escuro) em comparação ao resultados esperados (verde claro) no *data set* de treino:

- **MAE:** 16.177
- **RMSE:** 419.241

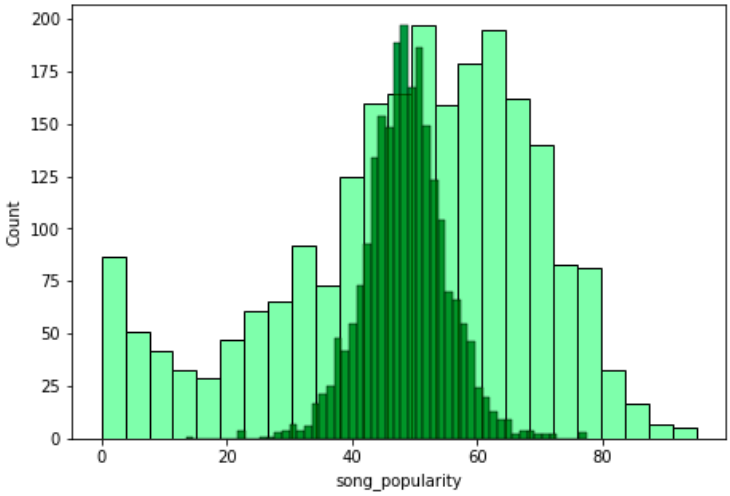

Figura 12: *Comparação de resultados reais com previsões do modelo*

3.5.3 Redes Neuronais Artificiais

A aplicação deste modelo necessitou um pouco mais de trabalho e atenção visto que engloba vários parâmetros como função de ativação, número de camadas e número de neurónios em cada camada. Assim, começamos por normalizar os dados presentes em todas as *features*, incluindo o *target*, passando, de seguida, à divisão do *data set* normalizado para treino e teste.

Criamos uma função (*build_model*) onde definimos os vários parâmetros da rede, sendo estes: 4 camadas intermédias, com 18, 15, 12 e 9 nodos respetivamente, tendo a camada de *input* 9 nodos (número de *features*) e a de *output* apenas 1.

```
def build_model (activation='relu', learning_rate=0.01):
    model = Sequential()
    model.add(Dense(18, input_dim=9, activation=activation))
    model.add(Dense(15, activation=activation))
    model.add(Dense(12, activation=activation))
    model.add(Dense(9, activation=activation))
    model.add(Dense(1, activation=activation))

    model.compile(
        loss = 'mae',
        optimizer = tf.optimizers.Adam(learning_rate),
        metrics = ['mae', 'rmse'])

    return model
```

Figura 13: *Função de Configuração da Rede Neuronal*

Após a aplicação deste modelo, obtivemos os seguintes valores de erro, apresentando, também, uma representação gráfica das previsões em conjunto com o valor pretendido:

- **MAE:** 15.965
- **RMSE:** 399.556

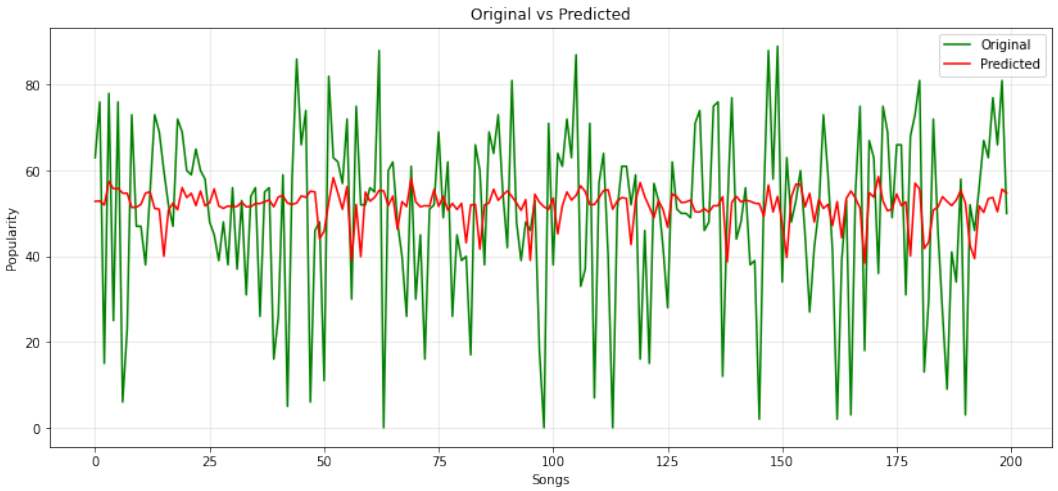

Figura 14: *Valores Obtidos vs Valores Pretendidos*

3.6 Resultados e Conclusões

Após a análise dos resultados obtidos pelos diferentes modelos, verificamos que todos se comportam da mesma forma, isto é, obtêm valores de previsão dentro de um intervalo inferior ao expectável. Examinando os dados divididos para treino e teste, verificamos que estes se encontram bem distribuídos, isto é, são representativos do intervalo de previsão, como mostra a figura 15.

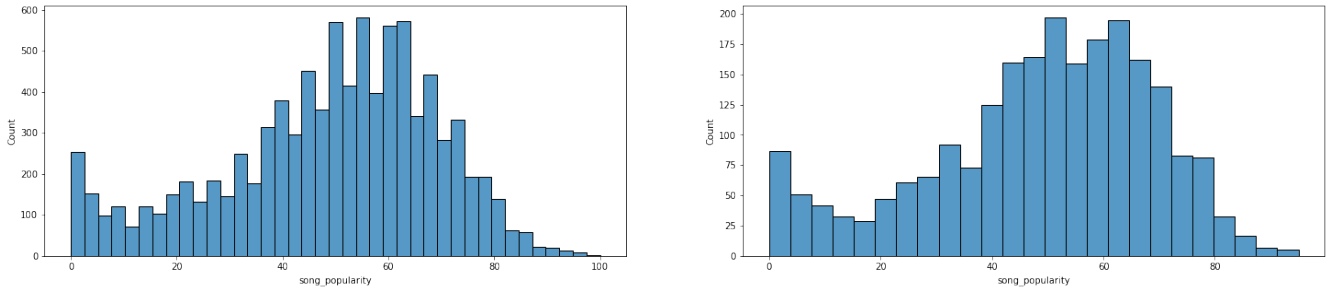

Figura 15: *Distribuição dos dados de Treino e Teste*

Com esta informação, concluímos que seria necessário uma análise mais profunda do data set para verificar se este apresenta problemas inerentes, visto que os vários modelos apresentam as mesmas falhas. Dado que, na recolha do data set, alguns atributos como *key* e *audio mode* não possuem informação disponível sobre o que representam, não pudemos tirar partido de tais dados.

4 Conclusão

Após a realização deste projeto, o grupo encontra-se satisfeito com o trabalho desenvolvido, tendo sido alcançados todos os objetivos estabelecidos quer pelos docentes, como pelo próprio grupo. No entanto, acreditamos que alguns dos modelos apresentados poderão ser melhorados no futuro, através de um tratamento mais aprofundado e meticuloso.

Relativamente aos *data sets* trabalhados:

- No *data set* ***Crash DataSet***, desenvolvemos modelos de classificação a partir das várias condições presentes num acidente, obtendo um modelo capaz de prever o nível de incidentes rodoviários, numa determinada hora, na cidade de Guimarães. No fim deste desenvolvimento, acreditamos ter obtido uma série de modelos com um bom nível de precisão.

- No *data set* ***Song Popularity***, o grupo desenvolveu modelos de regressão a partir dos dados inicialmente tratados e analisados, obtendo um modelo capaz de prever a popularidade de uma música. Consideramos ter obtido modelos com uma métrica de erro acima do expectável.

Em suma, aprofundamos o conhecimento lecionado nas aulas teóricas e práticas da unidade curricular em relação às várias fases de exploração, tratamento e criação de modelos de aprendizagem através de algoritmos de *machine learning*, aumentando também a nossa capacidade de utilização de *Python Notebooks* através da configuração de ambientes de desenvolvimento pelo *Anaconda*, que permitiu a realização deste projeto.