Dados e Aprendizagem Automática

MEI - 1º ano - 1º semestre
Universidade do Minho

Trabalho Prático

Trabalho realizado por: *Número*

Pedro Paulo Queirós Tavares PG50698
Tiago Daniel Lopes Aroso da Costa PG50777
Diogo Casal Novo PG50342

Braga, 14 de janeiro de 2023

Índice

1 Introdução **2**

2 Dataset Grupo **3**
2.1 Exploração e tratamento dos dados 3
2.1.1 Análise de uma estação 4
2.1.2 Análise a todo o dataset 5
2.1.3 Gráficos interessantes para visualizar os dados 7
2.2 Treino dos modelos 8
2.3 Discussão de resultados 9

3 Dataset Competição **10**
3.1 Exploração de dados 10
3.2 Tratamento de dados 11
3.3 Gráficos sobre os dados 12
3.4 Treino dos modelos 13
3.5 Discussão de resultados 15

4 Conclusão **16**

1. Introdução

No âmbito da unidade curricular de Dados e Aprendizagem Automática, foi proposto o desenvolvimento de um trabalho prático com vista a aprofundar e consolidar os conhecimentos sobre Machine Learning abordados ao longo do semestre.

O proposto foi assim a análise e construção de modelos de *machine learning* de dois datasets.

O dataset competição é o fornecido pela equipa docente e contém dados referentes à quantidade e características dos incidentes rodoviários que ocorreram numa cidade portuguesa em 2021. O objetivo deste dataset passa por desenvolver modelos de Machine Learning capazes de prever o nível de incidentes rodoviários, numa determinada hora, na referida cidade.

O dataset de grupo, por sua vez, foi escolhido pelo grupo e contém dados relativos ao clima de uma determinada região do Brazil. O objetivo foi também escolhido pelo grupo e a sua escolha será fundamentada mais à frente no documento.

Para ambos os datasets a metodologia utilizada foi a SEMMA porque foi a que o grupo achou que seria a mais indicada para este tipo de trabalho.

Ao longo deste documento serão assim expostas todas as decisões tomadas pelo grupo e será feita uma análise crítica dos resultados obtidos pelos modelos.

2. Dataset Grupo

O dataset escolhido pelo grupo, como já referido, foi um dataset com os dados sobre o clima no Brasil. Este dataset contém, entre outros, dados recolhidos por várias estações sobre a quantidade de precipitação na última hora, a radiação, a temperatura na última hora, etc. Para poder entender melhor os atributos e as relações entre eles, foi então dado início à exploração de dados. Este dataset pode ser encontrado em `https://www.kaggle.com/datasets/PROPPG-PPG/hourly-weather-surface-brazil-southeast-region`

2.1 Exploração e tratamento dos dados

O primeiro passo a tomar na análise dos dados é obter o número de linhas e colunas do dataset para perceber com que dimensão estamos a trabalhar. Este dataset possui 11427120 linhas e 27 colunas. Estamos perante um dataset considerável em termos de tamanho e com bastante informação.

O segundo passo foi perceber quais colunas eram do tipo categórico. Esta análise foi feita recorrendo ao comando *info()* no qual pudemos também observar o nome das colunas.

```
(11427120, 27)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11427120 entries, 0 to 11427119
Data columns (total 27 columns):
 #   Column                                          Dtype
---  ------                                          -----
 0   index                                           int64
 1   Data                                            object
 2   Hora                                            object
 3   PRECIPITAÇÃO TOTAL, HORÁRIO (mm)                float64
 4   PRESSAO ATMOSFERICA AO NIVEL DA ESTACAO, HORARIA (mB)  float64
 5   PRESSÃO ATMOSFERICA MAX.NA HORA ANT. (AUT) (mB)  float64
 6   PRESSÃO ATMOSFERICA MIN. NA HORA ANT. (AUT) (mB)  float64
 7   RADIACAO GLOBAL (Kj/m²)                         int64
 8   TEMPERATURA DO AR - BULBO SECO, HORARIA (°C)    float64
 9   TEMPERATURA DO PONTO DE ORVALHO (°C)            float64
 10  TEMPERATURA MÁXIMA NA HORA ANT. (AUT) (°C)      float64
 11  TEMPERATURA MÍNIMA NA HORA ANT. (AUT) (°C)      float64
 12  TEMPERATURA ORVALHO MAX. NA HORA ANT. (AUT) (°C)  float64
 13  TEMPERATURA ORVALHO MIN. NA HORA ANT. (AUT) (°C)  float64
 14  UMIDADE REL. MAX. NA HORA ANT. (AUT) (%)        int64
 15  UMIDADE REL. MIN. NA HORA ANT. (AUT) (%)        int64
 16  UMIDADE RELATIVA DO AR, HORARIA (%)             int64
 17  VENTO, DIREÇÃO HORARIA (gr) (° (gr))            int64
 18  VENTO, RAJADA MAXIMA (m/s)                      float64
 19  VENTO, VELOCIDADE HORARIA (m/s)                 float64
 20  region                                          object
 21  state                                           object
 22  station                                         object
 23  station_code                                    object
 24  latitude                                        float64
 25  longitude                                       float64
 26  height                                          float64
dtypes: float64(15), int64(6), object(6)
memory usage: 2.3+ GB
```

Figura 2.1: Informação sobre os atributos do dataset grupo

Como observado na Figura 2.1, temos 6 colunas do tipo categórico que precisarão de serem transformadas para os modelos poderem ser aplicados.

As colunas "index"e "station"podem ser descartadas porque a informação por elas dada não é relevante para o modelo. Em relação às colunas "Data"e "Hora"podemos transformar numa única convertendo-a para o tipo *datetime*.

A coluna "region"pode também ser descartada porque apenas apresenta um valor que é a região

que foi alvo de análise. Outro ponto importante que foi descoberto na exploração de dados foi o número de atributos nulos ou NAN. Numa primeira análise, não foram detetados valores nulos ou NAN, porém, através do comando *describe()* pudemos observar que todas as colunas tinham como mínimo o valor "-9999". Tendo em conta os tipos de atributos que estamos a lidar, por exemplo, a temperatura, é notável que estes valores são supostos serem tratados como NAN. Para o tratamento desta coluna foram então substituídas as linhas que tivessem este valor por NAN. Desta alteração surgiram então várias linhas com valores NAN como pode ser observado na Figura 2.2

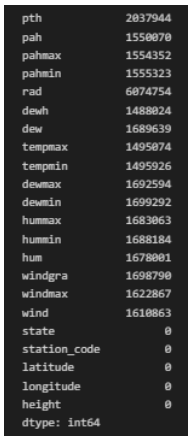
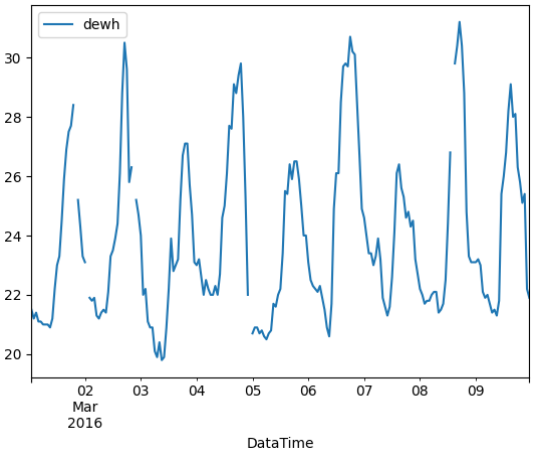


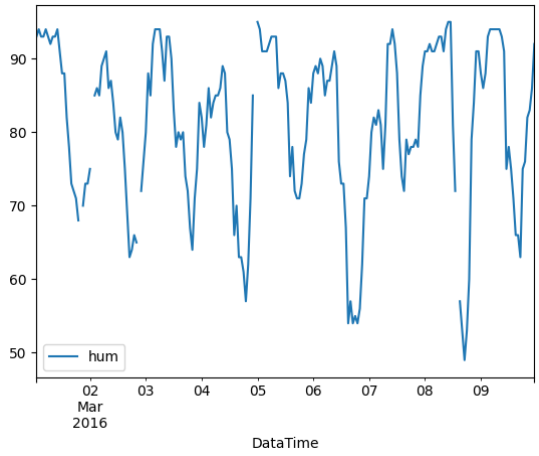
Figura 2.2: Valores NaN dataset grupo

2.1.1 Análise de uma estação

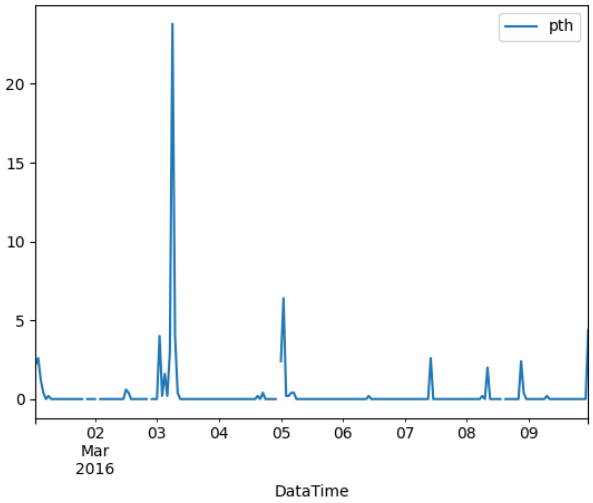
A grande quantidade de dados NAN implica uma maior e mais detalhada exploração dos dados para poder não introduzir *bias* ou remover dados demasiado importantes. Foi então feita uma análise aos dados de estações específicas, sendo a primeira em análise a estação "A027". As colunas alvo de análise foram as que podem eventualmente ser a *target* para previsão, nomeadamente, a precipitação, a radiação, a humidade e a temperatura.



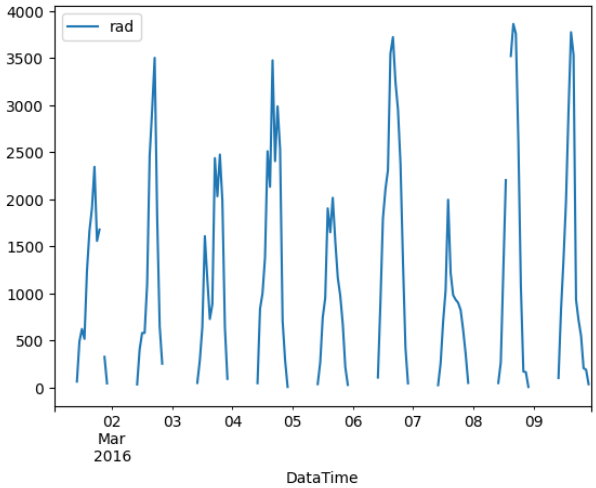
Temperatura Estação A027



Humidade Estação A003



Precipitação Atmosférica Estação A027



Radiação Estação A027

O observado é a falta de alguns valores intermédios no gráfico da temperatura e da humidade e, no caso da radiação e da precipitação, os valores em faltam são normalmente zero. Estes valores em falta que deveriam ser 0 podem ser explicados pela estação não conseguir recolher e portanto registar como NAN, como por exemplo no caso da radiação, os dados em falta são sempre à noite, o que indica a falta de radiação.
Foi então feita a interpolação para o gráfico da temperatura o que já retificou e melhorou os dados obtidos. Para os dados da precipitação e radiação foram substituídos os valores NAN por 0. O método seguido para as outras colunas foi a interpolação.

2.1.2 Análise a todo o dataset

Feita a análise para uma estação podemos extrapolar os resultados obtidos numa estação para o dataset todo e, deste modo, realizar as mesmas operações já descritas mas desta vez para todo o conjunto de dados.

Após estas alterações já não são então observados valores NAN e podemos explorar melhor os dados para decidirmos qual será a *target class*. Começamos por analisar os valores da temperatura pelos anos de recolha de dados e, como é possível observar na Figura 2.5, existem alguns anos onde há falta de dados que pode ser explicado pelo facto de a estação ter estado em manutenção por exemplo.

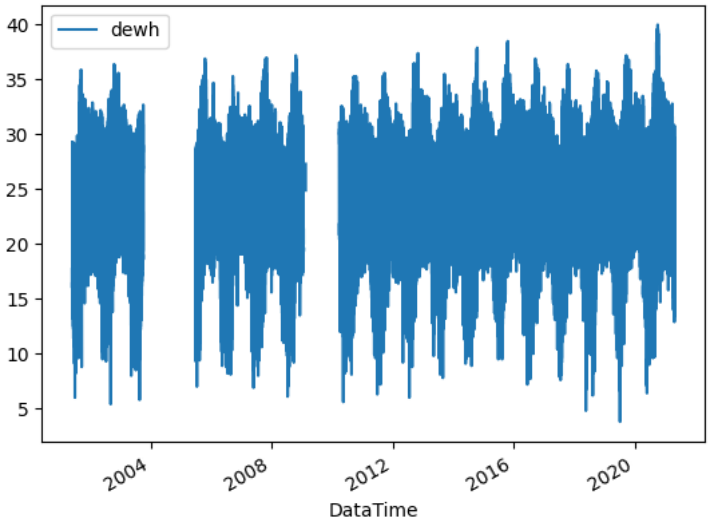
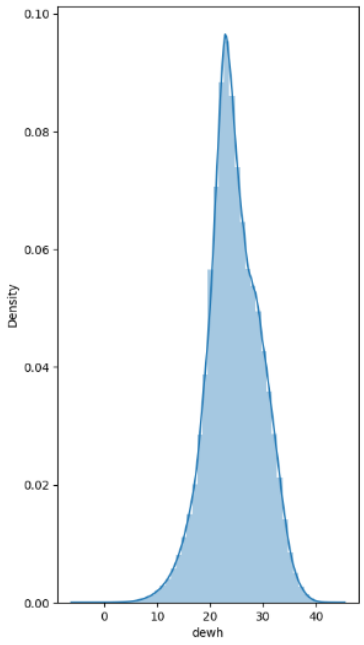


Figura 2.5: Valores da temperatura ao longo dos anos

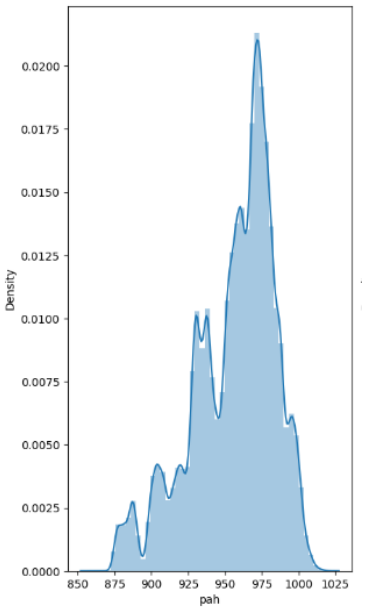
Os valores observados indicam que nos primeiros anos de recolha de dados houveram várias inconsistências e falhas, o que pode complicar o treino do modelo, por este motivo escolhemos apenas considerar os dados do dataset a partir do ano 2016.

A primeira intuição com este dataset foi treinar um modelo capaz de prever a precipitação, porém, após realizarmos a distribuição dos dados, reparamos que não seguia uma boa distribuição sendo todos os valores bastante próximos de 0. Isto faz todo o sentido porque os dados são relativos a uma região do Brasil onde raramente chove, e portanto, a escolha da precipitação como target não seria uma boa ideia.

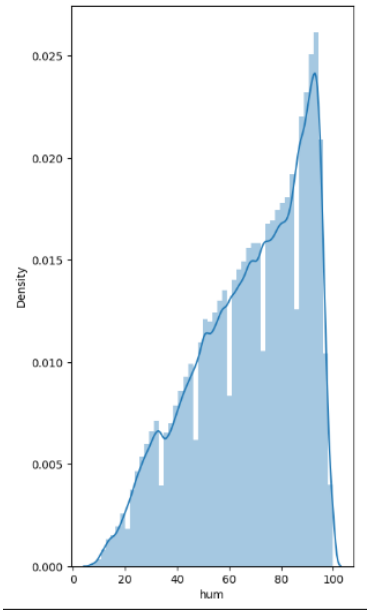
Analisamos portanto as outras colunas com maior interesse, nomeadamente a temperatura, a pressão atmosférica e a humidade.



Distribuição da temperatura após 2016



Distribuição da pressão após 2016



Distribuição da humidade após 2016

Claramente, devemos usar a temperatura como target class pois é a que possui a melhor distribuição.

2.1.3 Gráficos interessantes para visualizar os dados

Para obtermos algum contexto e informação sobre os dados, exploramos alguns gráficos que nos dão a possibilidade de vermos a variação dos mesmos.

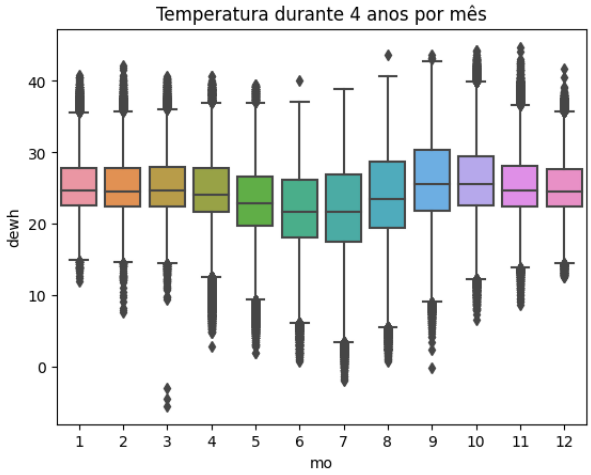


Figura 2.7: Valores da temperatura entre 2016-2020 por mês

Os valores observados correspondem com o esperado, dado que o Brasil tem um clima tropical e, por isso, o seu clima é mais quente e seco. Podemos também observar que os meses com menor temperatura correspondem aos meses de Maio até Julho que são os meses de inverno.
Outro gráfico que foi analisado foi a temperatura por estado.

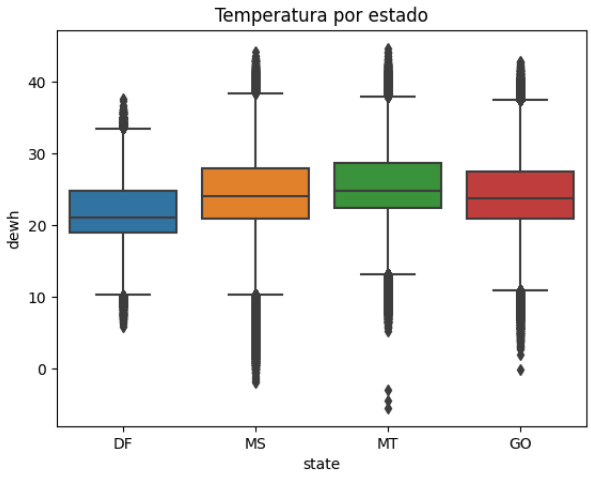


Figura 2.8: Valores da temperatura por estado

2.2 Treino dos modelos

Estabelecida a coluna target, sendo neste caso a coluna da temperatura, começamos o treino dos modelos. sendo esta uma variável do tipo contínuo devemos usar métodos de treino adequados. Os dois métodos escolhidos para os modelos foram o *Linear Regression* e o *Decision Tree Regressor*. A divisão do dataset foi de 30% em teste e 70% em treino.

Para validar que o dataset de teste está bem distribuído analisamos o gráfico da sua distribuição, como observado na figura 2.9.

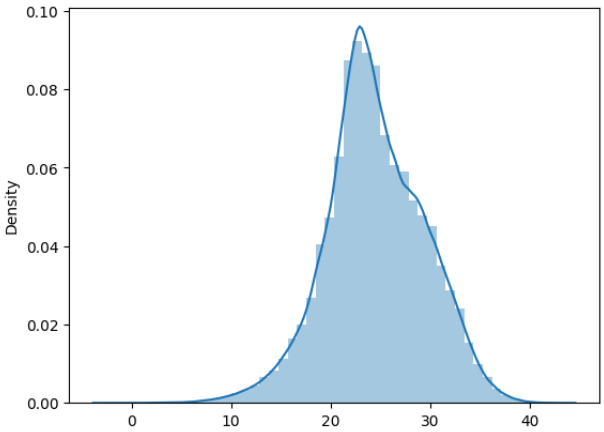


Figura 2.9: Distribuição do dataset de teste

A avaliação dos modelos foi feita recorrendo aos parâmetros R^2 e MSE. O primeiro explica a variância dos dados que é explicada pelo modelo, sendo valores mais próximos de 100% indicativos que o modelo é melhor. O segundo calcula a média dos erros quadrados e valores mais próximos de 0 indicam um melhor modelo.

Modelo/Resultados	Linear Regressor	Decision Tree Regressor
R^2	0.991	0.994
MSE	0.230	0.153

Tabela 2.1: Resultados obtidos sem *cross-validation*

Estes resultados podem indicar *overfitting* e, por isso, realizamos *cross-validation* para atestarmos a validade destes resultados.

Modelo/Resultados	Linear Regressor	Decision Tree Regressor
R^2	0.9898	0.9889
MSE	0.252	0.259

Tabela 2.2: Resultados obtidos com *cross-validation*

Em relação ao Decision Tree Classifier foi também usado o *tunning* de certos hiperparâmetros usando *GridSearch*. Os parâmetros que foram alvo de *tunning* foram os seguintes : *max_features,min_samples_split,min_samples_leaf*. Os valores foram primeiramente definidos de forma *random* e, após uma iteração foi selecionada uma gama de valores onde os melhores resultados se encontravam para, na seguinte iteração se perceber quais os melhores valores. Os resultados obtidos podem então ser vistos na seguinte tabela.

Modelo/Resultados	Decision Tree Regressor
R^2	0.9932
MSE	0.169

Tabela 2.3: Resultados obtidos com *cross-validation* e *tunning*

2.3 Discussão de resultados

Inicialmente a escolha de um dataset com bastante informação trouxe uma dificuldade acrescida devido ao facto de a análise de dados ter de ser mais intensiva e mais minuciosa. Para além disto, revelou também que os datasets contêm bastantes "impurezas"que necessitam de ser tratadas de acordo com o contexto do problema.
Posteriormente, a maior dificuldade centrou-se no treino dos modelos porque, numa primeira fase, era usado todo o dataset para treino e isto tornava os tempos de treino bastante grandes. Superada esta dificuldade, obtivemos os modelos treinados e verificamos que os modelos iniciais (Tabela 2.1) possuíam ótimos resultados o que poderia indicar *overfitting*.
Com a construção dos novos modelos onde há *cross-validation* podemos verificar que não havia *overfitting* (Tabela 2.2) porque os resultados mantiveram-se ótimos. Foi também realizado *tunning* dos hiperparâmetros para tentar ainda assim melhorar os resultados e também, como forma de preparação para o dataset de grupo. Os resultados obtidos mostram que o *tunning* foi efetivo, aumentando a eficácia do modelo a fazer previsões (Tabela 2.3).

Concluindo, achamos que alcançamos o pretendido, ou seja, fizemos uma boa exploração e tratamento dos dados, usando também dois modelos que obtiveram excelentes resultados.

3. Dataset Competição

O segundo dataset alvo de análise e tratamento por parte do grupo é um dataset pré definido pela equipa docente. Este contém dados referentes à quantidade e características dos incidentes rodoviários que ocorreram na cidade de Guimarães em 2021 (o dataset cobre um período que vai desde o dia 01 de Janeiro de 2021 até ao dia 31 de Dezembro do mesmo ano). O objetivo é, portanto, prever o número de incidentes rodoviários que irão acontecer na cidade de Guimarães a uma determinada hora.

3.1 Exploração de dados

O método de exploração de dados seguirá o mesmo que foi utilizado no outro dataset, começando pela análise da dimensão. A diferença notável neste dataset é que são fornecidos dois ficheiros sendo um de treino e outro de teste. Isto implica que as operações realizadas no ficheiro de treino tenham que ser replicadas no de teste.
Assim, o ficheiro de treino contém 5000 linhas e 13 colunas e o de teste 1206 linhas e 12 colunas. Em termos de tipo de variáveis temos 7 do tipo não categóricas que terão de ser tratadas para uso nos modelos e 6 numéricas.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 13 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   city_name            5000 non-null   object 
 1   magnitude_of_delay   5000 non-null   object 
 2   delay_in_seconds     5000 non-null   int64  
 3   affected_roads       4915 non-null   object 
 4   record_date          5000 non-null   object 
 5   luminosity           5000 non-null   object 
 6   avg_temperature      5000 non-null   float64
 7   avg_atm_pressure     5000 non-null   float64
 8   avg_humidity         5000 non-null   float64
 9   avg_wind_speed       5000 non-null   float64
 10  avg_precipitation    5000 non-null   float64
 11  avg_rain             5000 non-null   object 
 12  incidents            5000 non-null   object 
dtypes: float64(5), int64(1), object(7)
memory usage: 507.9+ KB
```

Figura 3.1: Info dataset competição

Em relação aos valores NAN pudemos ver que existem 22 valores em falta na coluna "affected_roads".

3.2 Tratamento de dados

O primeiro passo a seguir é escolher como tratar os valores NAN, sendo que o procedimento que o grupo seguiu foi mudar os mesmos pelo valor zero, primeiramente porque simplesmente excluir estas colunas iria ser prejudicial aos modelos pela reduzida dimensão do dataset. Por outro lado, valores NAN nesta coluna indicam que não foram registadas ruas afetadas, logo podemos substituir o seu valor por zero.
Ainda em relação a esta coluna, dado que é categórica também teve de ser alvo de tratamento. Os valores apresentados nesta coluna são uma lista com as ruas afetadas, porém isto não acrescenta informação ao modelo. Então a abordagem seguida foi transformar esta coluna numa nova que tem o número de ruas afetadas.

A variável que foi tratada a seguir foi a "avg_precipitation"e esta foi excluida do dataset porque apenas apresentava um valor e, por isso, não iria acrescentar grande vantagem no modelo.

A coluna "city_name"foi também excluida porque possui apenas um valor possível.

Da variável categórica "record_date"foram extraídas 4 novas colunas que correspondem ao mês, hora, dia e dia da semana. Inicialmente apenas tinham sido extraídas 3 novas colunas, porém aquando da observação de resultados dos modelos já treinados, a presença da coluna "week_day"revelou uma melhor *accuracy* dos modelos.

As variáveis "magnitude_of_delay", "luminosity"e "avg_rain"foram transformadas usando *Ordinal Encoder*, uma vez que, claramente, se tratam de variáveis ordinais.

O *target* deste dataset (apenas presente no ficheiro de treino) foi, por sua vez, transformado usando *Label Encoder* porque se trata de uma *target*.

3.3 Gráficos sobre os dados

Os gráficos que mais estamos interessados em visualizar são aqueles que mostram a distribuição das variáveis que acabamos de transformar para perceber como vão influenciar os modelos.

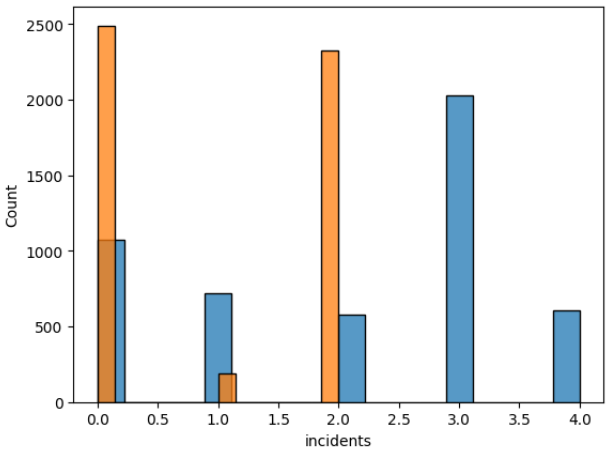


Figura 3.2: Histograma sobre as variáveis *incidents* e *luminosity*

A variável "incidents"(a azul) possui uma distribuição de dados razoável para podermos aplicar o método de classificação, ou seja, os número de dados presentes em cada classe é significativo e estão todos dentro da mesma escala de valores. Já relativamente à variável "luminosity"podemos ver que DARK e LIGHT são as mais representativas enquanto que LOW_LIGHT possui poucos valores, sendo que isto pode apresentar um ponto de falha no modelo se os novos dados que lhe forem apresentados tiverem este tipo de luminosidade.

Em relação à "magnitude_of_delay"podemos observar no gráfico seguinte que a sua distribuição também é não tão boa uma vez que a maior parte dos dados apresenta o valor UNDEFINED.

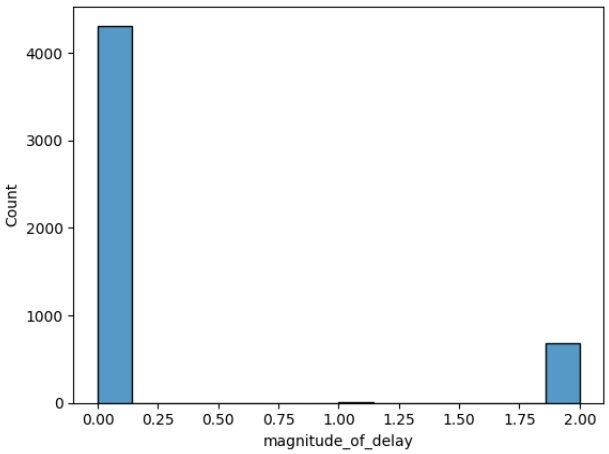


Figura 3.3: Histograma sobre a variável *magnitude of delay*

Por fim, a feature "avg_rain"também sofre de desbalanceamento dos dados.

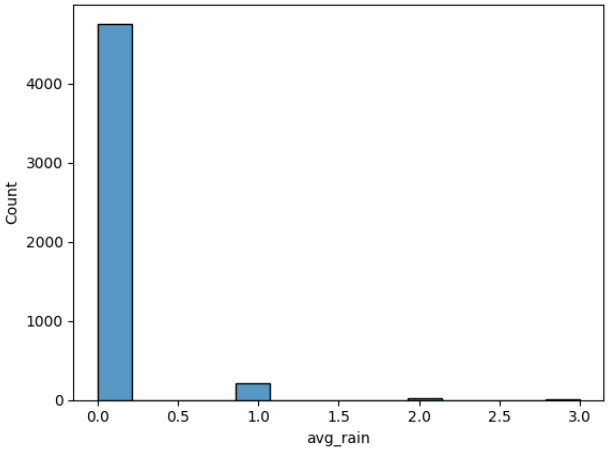


Figura 3.4: Histograma sobre a variável *avg rain*

3.4 Treino dos modelos

Uma vez que são fornecidos os ficheiros de teste e treino, não precisamos de realizar o *split* dos dados. A única alteração que precisamos de fazer é dividir o dataset de treino em *X_train* e *Y_train* para podermos treinar os modelos.

O primeiro modelo que foi utilizado foi o **Decision Tree Classifier**.Com este modelo, e com *cross-validation* de 10 iterações, obtivemos uma *accuracy* de 90.46%. Decidimos não apostar em realizar inicialmente o *tunning* do modelo pois a sua *base accuracy* já era boa e queríamos testar outros modelos primeiro, por isso, procuramos utilizar outros modelos que aumentassem a *accuracy*.

O segundo modelo utilizado foi o **SVC** e para explorarmos melhor este modelo decidimos utilizar *tunning* dos hiperparâmetros "C","kernel"e "gamma". Os resultados obtidos foram ainda piores que o decision tree classifer e, por isso, decidimos não apostar na exploração mais detalhada deste modelo.

O modelo seguinte que decidimos explorar foi o **MLP**. Neste modelo criamos então uma rede de *perceptrons* com 3 layers, sendo que a primeira layer possui 12 nodos (número de features), a segunda layer foi alvo de teste com valores entre 6 e 24 nodos (respetivamente metade e dobro do valor de nodos de input) e, por fim, como se trata de um problema em que a target é multiclass, a última layer possui o número de classes possíveis, ou seja, 5 nodos. Em relação às funções de ativação foram experimentadas várias para a camada de input e a hidden layer, mas, para a última layer apenas fazia sentido usar a "softmax"porque é uma target multiclass.O outro parâmetro que foi alvo de *tunning* foi o learning rate.
Os resultados obtidos podem ser vistos na próxima figura.

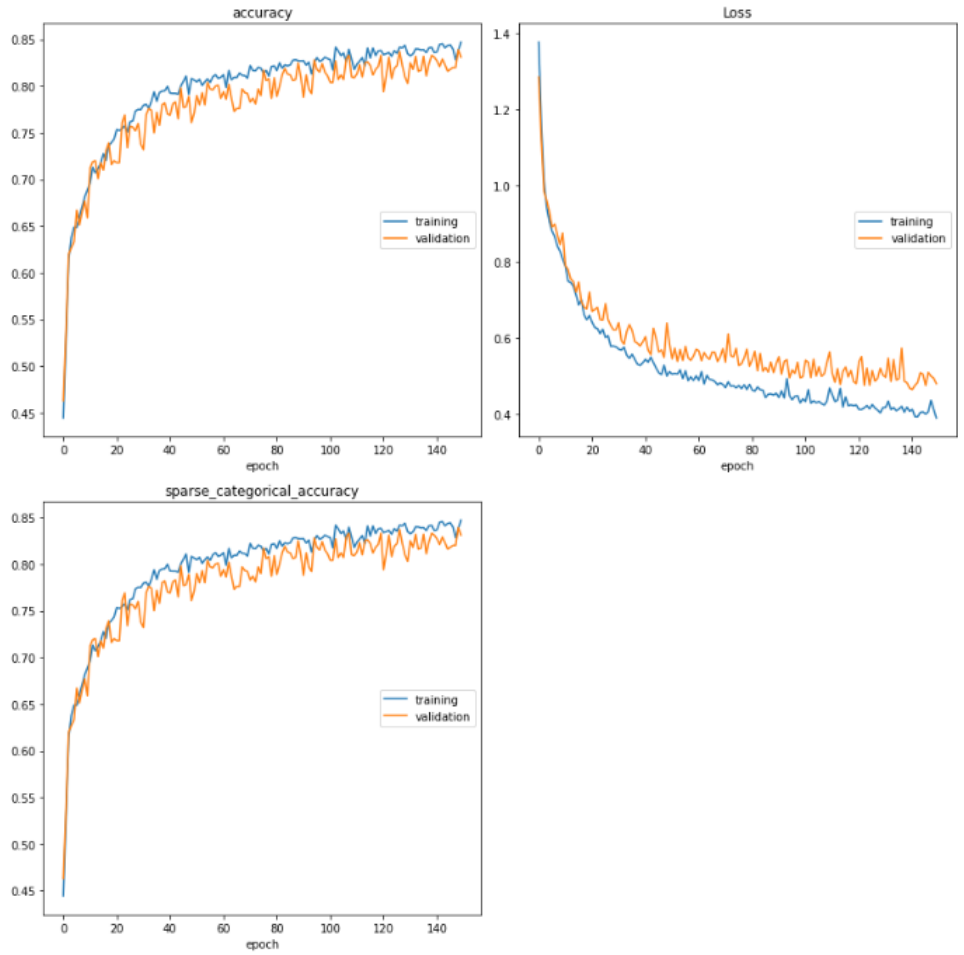


Figura 3.5: Análise do treino MLP

Como podemos observar o modelo atingiu uma boa *accuracy* e podemos notar que a partir das 150 épocas o modelo entraria em *overfitting* o que prejudicaria os resultados. Outro ponto que podemos referir é que o particionamento dos dados em validação e treino foi também bem feito porque as curvas de validação e treino estão bastante próximas uma da outra.

Por fim, decidimos utilizar o modelo **Random Tree Classifier** que apresentou uma *accuracy* base de 91%. Como esta apresentava um valor mais elevado que todos os outros modelos, decidimos apostar mais na exploração do *tunning* deste modelo.

Os primeiros hiperparâmetros que decidimos afinar foram o número máximo de *features*, o critério, o número mínimo de splits, o número mínimo de *samples* numa folha e o número de estimadores. Estabelecidos estes hiperparâmetros, decidimos apostar no hiperparâmetro "*class_weight*"uma vez que algumas classes devem ter mais peso que outras.

Com todo este *tunning* os resultados obtidos foram os melhores até ao momento, com uma *accuracy* de 92.24%.

Voltamos portanto ao **Decision Tree Classifier** para verificar se conseguiríamos superar os valores do Random Forest. Após o *tunning* o valor máximo de *accuracy* obtido foi 92%.

Modelo/Resultados	Decision Tree Regressor	SVC	MLP	Random Tree Classifier
Accuracy	0.92	0.815	0.8352	0.9224

Tabela 3.1: Accuracy final dos diferentes modelos utilizados

3.5 Discussão de resultados

Resumindo, como observado na tabela 3.1, o modelo **Random Tree Classifier** foi o que obteve melhores resultados, embora muito próximo dos resultados do **Decision Tree Regressor**. Em termos de *tunning* de hiperparâmetros o vencedor é bastante mais complexo de afinar pela grande quantidade de valores possíveis de alteração.
Uma nota também tem de ser referida sobre os outros modelos porque embora não apresentem a melhor *accuracy*, apresentam bons resultados. O **MLP** provou-se o mais difícil de treinar e o mais complexo uma vez que as *layers* possuem vários parâmetros que podem ser alterados e o seu treino é bastante moroso.

Concluindo, os modelos apresentados possuem uma *accuracy* bastante alta o que indica um bom trabalho por parte do grupo, mas os melhores modelos podiam ainda ser alvos de mais minuciosidade em exploração de hiperparâmetros para obtenção de melhores resultados e, também, explorar técnicas de normalização de dados para obtenção de melhores resultados. Outro ponto importante de se referir é o facto, de não serem apresentados neste relatório os resultados intermédios resultantes das alterações a *feature engineering* porque os mesmos formaram piores resultados que não são importantes para análise.

4. Conclusão

Concluindo, após uma intensa exploração de dados, do seu tratamento e criação de modelos, o grupo considera que consolidou bem os conhecimentos sobre *machine learning* e que os objetivos deste trabalho foram atingidos com sucesso.

Foram também retiradas várias aprendizagens novas decorrentes de cada dataset, nomeadamente, através do dataset de grupo conseguimos entender as dificuldades e adversidades de datasets com grande dimensão e, através do dataset de grupo, pudemos explorar vários modelos, a sua utilidade e os resultados para um certo tipo de dados.

Por fim, o dataset de grupo proporcionou também uma experiência nova que foi a competição com os outros colegas para ver quem treina o modelo com melhor *accuracy*, o que torna desafiante a criação de modelos e o *tunning* e acrescenta um pouco de motivação para este trabalho mais minucioso.