

# Trabalho Prático - Fase III

Mineração de Dados

Daniel Xavier  
PG50310

Diogo Rebelo  
PG50327

Lídia Sousa  
PG50551

## I. INTRODUÇÃO

O presente relatório visa a compreensão sucinta e exploratória do projeto desenvolvido no âmbito da unidade curricular de Mineração de Dados.

Este projeto tinha como mote, extrair conhecimento útil e não óbvio, pelo que a nossa abordagem seguiu para a vertente da qualidade do ar. Para o efeito utilizamos diversas fontes de dados, que passaremos a descrever na secção do relatório Recolha de Dados.

Por conseguinte, para a implementação do nosso caso de estudo e aplicação fizemos a recolha, processamento, análise e mineração dos dados pelo que, para tornar mais fácil e acessível a visualização deste tratamento desenvolvemos uma aplicação web.

Deste modo, todo o desenvolvimento do projeto será explicado com mais detalhe de seguida.

## II. MOTIVAÇÃO E OBJETIVOS

### A. Motivação

O estudo e a análise da qualidade do ar são de importância inestimável, especialmente nos tempos atuais de mudanças climáticas aceleradas. A previsão precisa do Índice de Qualidade do Ar (AQI) é crucial não apenas para a gestão ambiental, mas também para a saúde pública, porque a poluição atmosférica tem impactos significativos na saúde humana, incluindo doenças respiratórias e cardiovasculares.

A cidade de Braga, sendo uma região em crescimento rápido apresenta um caso de estudo especialmente interessante. Ao prever o AQI nesta área podemos fornecer informações vitais para a gestão da qualidade do ar da cidade, ações preventivas de saúde e também contribuir para o desenvolvimento de políticas públicas de meio ambiente.

Com a mineração de dados e a capacidade de processar grandes volumes de dados e identificar padrões complexos podemos explorar a interação entre as condições meteorológicas e a qualidade do ar e, assim, prever o AQI.

Este trabalho é motivado pelo desejo de aplicar as aprendizagens obtidas na unidade curricular de Mineração de Dados para resolver problemas práticos e interessantes como as questões ambientais.

### B. Objetivos

O objetivo deste projeto centra-se na previsão do AQI (índice de qualidade do ar) para determinadas coordenadas

num dia e hora através de condições meteorológicas e da atmosfera.

Os objetivos organizados pelas fases de um trabalho de Mineração de dados são:

- **Recolha de dados:**

- Recolher e organizar dados históricos e em tempo real sobre o AQI, condições meteorológicas e atmosféricas em Braga.
- Integrar diferentes fontes de dados e garantir a consistência e a precisão.

- **Pré-processamento de Dados:**

- Transformar os dados em um formato adequado para análise.

- **Análise de Dados:**

- Realizar uma análise exploratória dos dados para identificar tendências e detetar anomalias nos dados.
- Visualizar os dados para compreender a distribuição dos valores e a correlação entre diferentes variáveis.
- Normalizar e preparar os dados para a modelagem.

- **Modelagem**

- Treinar diferentes modelos de aprendizado de máquina para prever o AQI com base nas condições meteorológicas e atmosféricas.
- Avaliar e comparar o desempenho desses modelos usando métricas apropriadas.
- Ajustar os modelos com base no desempenho para melhorar a precisão das previsões.
- Utilizar técnicas como ajuste de hiperparâmetros ou *cross-validation* para otimizar o desempenho do modelo.

- **Validação**

- Validar as previsões do modelo com recurso a um dataset destinado para esse objetivo

- **Implementação**

- Desenvolver um sistema web que integre o modelo desenvolvido para fazer previsões do AQI em Braga.

## III. METODOLOGIA

Neste projeto utilizamos a metodologia CRISP-DM (*Cross Industry Standard Process for Data Mining*) que forneceu uma estrutura sólida que ajudou a guiar todo o processo de mineração de dados, desde a compreensão do problema até a implantação do modelo.

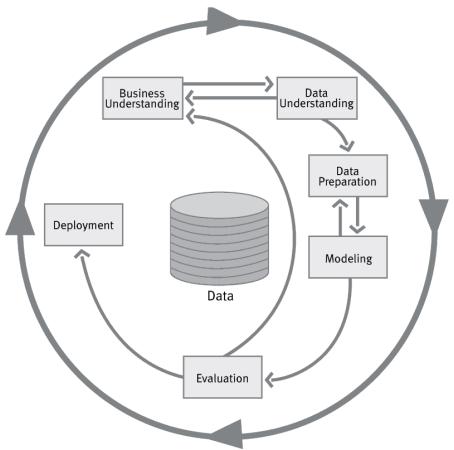


Fig. 1. Imagem representativa da metodologia CRISP-DM

- **Compreensão do Negócio:** Esta é a fase inicial onde definimos claramente o problema e entendemos os objetivos do negócio, assim como os requisitos do projeto a partir de uma perspectiva de negócio. Neste trabalho, a fase de compreensão do negócio envolve a identificação do problema da previsão da qualidade do ar e a definição do objetivo do projeto.
  - **Compreensão dos Dados:** Recolhemos os dados necessários e procedemos com a exploração inicial dos dados para entender as variáveis que temos. Neste trabalho, isto traduziu-se na recolha de dados de diferentes fontes, incluindo dados de qualidade do ar e dados meteorológicos, e realizando uma análise exploratória dos dados.
  - **Preparação dos Dados:** Limpamos e pré-processamos os dados para torná-los adequados para modelagem. Isto pode incluir a imputação de valores em falta, tratamento de outliers, entre outros.
  - **Modelação:** Seleccionamos e aplicamos várias técnicas de modelação de dados e ajustamos os parâmetros do modelo para obter o melhor modelo possível. Neste trabalho, usamos métodos como LSTM e ARIMA.
  - **Avaliação:** Avaliamos os modelos de forma apropriada, assegurando que estes satisfazem os critérios definidos na primeira fase.
  - **Implementação:** Intregamos o nosso modelo num ambiente de produção para que possa ser usado para fazer previsões futuras sobre a qualidade do ar em Braga.

#### IV. FASES DE TRABALHO

#### *A. Recolha de dados*

Nesta fase, com o objetivo de obter dados de diversas fontes recolhemos dados de APIs públicas, como a OpenAQ, OpenWeather, Air Quality Index e Visual Crossing.

Um vez que o objetivo é prever o índice de qualidade do ar em Braga recolhemos nestas APIs dados relativos à cidade de Braga e zonas envolventes como Vila Verde e Ponte de Lima.

De seguida descreveremos cada uma dessas APIs e os dados obtidos de cada uma.

**1) OpenAQ:** Plataforma comunitária que fornece uma API para aceder a dados em tempo real bem como dados históricos sobre a qualidade do ar de diversos locais pelo mundo. Fornece informações sobre vários poluentes, como PM2.5, PM10, CO, SO<sub>2</sub>, NO<sub>2</sub> e O<sub>3</sub>. A OpenAQ utiliza fontes oficiais de dados governamentais e não governamentais para obter os dados disponibilizados.

No *dataset* obtido pela OpenAQ cada linha representa uma única medição, feita num horário específico, de um poluente específico para a cidade de Braga.

- locationId: Identificador numérico único para cada local.
  - location: Código de localização específico para este ponto de medição e estação de monitorização específica.
  - city: Cidade onde a medição foi feita. Neste dataset, todas as medições foram feitas em Braga.
  - country: País onde a medição foi feita. Neste dataset, todas as medições foram feitas em Portugal.
  - utc: A data e hora em que a medição foi feita, em horário UTC (horário a zero graus de longitude).
  - local: A data e hora local em que a medição foi feita.
  - parameter: O poluente a ser medido. Neste dataset, as medições são para PM10 (partículas menores que 10 micrômetros) e NO2 (dióxido de nitrogênio).
  - value: O valor medido para o poluente.
  - unit: A unidade em que a medição é relatada - microgramas por metro cúbico ( $\mu\text{g}/\text{m}^3$ ).
  - latitude e longitude: As coordenadas geográficas do local onde a medição foi feita.

2) ***OpenWeather***: API de dados meteorológicos que permite aceder a informações meteorológicas em tempo real, previsões, histórico e outros dados relacionados ao clima de diversos locais pelo mundo.

Para o dataset obtido através dos dados da OpenWeather os dados são:

- name: A cidade para a qual os dados se aplicam, neste caso, Braga.
  - datetime: A data dos dados meteorológicos.
  - tempmax e tempmin: As temperaturas máximas e mínimas do dia em graus Fahrenheit.
  - temp: A temperatura média do dia em graus Fahrenheit.
  - feelslikemax, feelslikemin e feelslike: As temperaturas máximas, mínimas e médias do dia, ajustadas para levar em consideração a umidade e a velocidade do vento (ou seja, a "sensação térmica").
  - dew: O ponto de orvalho médio do dia em graus Fahrenheit. O ponto de orvalho é a temperatura na qual o ar deve estar para se tornar saturado com vapor de água.
  - humidity: A umidade média do dia em percentagem.
  - precip: A quantidade total de precipitação do dia em polegadas.
  - precipprob: A probabilidade de precipitação do dia em percentagem.
  - precipcover: A percentagem de tempo em que ocorreu precipitação durante o dia.

- precipType: O tipo de precipitação (por exemplo, chuva).
- snow e snowdepth: A quantidade total de neve e a profundidade da neve em polegadas.
- windgust e windspeed: A velocidade do vento em milhas por hora, incluindo a rajada de vento (velocidade mais alta).
- winddir: A direção do vento em graus.
- sealevelpressure: A pressão ao nível do mar em milibares.
- cloudcover: A cobertura de nuvens média do dia em percentagem.
- solarradiation e solarenergy: A radiação solar e a energia solar do dia.
- uvindex: O índice UV médio do dia.
- severerisk: Um indicador do risco de tempo severo.
- sunrise e sunset: Os horários do nascer e do pôr do sol.
- moonphase: A fase da lua.
- conditions: Condições climáticas gerais do dia.
- description: Uma descrição mais detalhada das condições climáticas.
- stations: As estações meteorológicas das quais os dados foram coletados.

3) **Open Air:** Interface que permite aceder a dados de qualidade do ar. Fornece informações em tempo real sobre qualidade do ar, poluição do ar, emissões de gases e condições climáticas de várias estações de monitorização em todo o mundo.

Existem várias métricas que a OpenAir API fornece, incluindo:

- PM2.5 e PM10: Estes são dados sobre partículas finas suspensas no ar com diâmetros de 2,5 e 10 micrões, respectivamente. Essas partículas podem ser prejudiciais à saúde humana se inaladas.
- NO<sub>2</sub>, SO<sub>2</sub>, CO, O<sub>3</sub>: Estes são dados sobre a presença de vários gases poluentes no ar. NO<sub>2</sub> é dióxido de nitrogênio, SO<sub>2</sub> é dióxido de enxofre, CO é monóxido de carbono e O<sub>3</sub> é ozono.
- Índice de Qualidade do Ar (AQI): Este é um indicador geral da qualidade do ar. O AQI pode variar dependendo do país, pois diferentes países podem ter os seus próprios padrões para o que constitui "bom" ou "mau" qualidade do ar.

Para a cidade de Braga os dados que conseguimos foram para os poluentes PM2.5 e PM10.

- date: Data e hora em que os dados foram recolhidos no formato ISO 8601, que é um padrão internacional para representação de datas e horas.
- min: Valor mínimo registrado para a qualidade do ar naquela data.
- max: Valor máximo registrado para a qualidade do ar naquela data.
- median: Valor mediano da qualidade do ar naquela data.
- q1: Primeiro quartil dos dados de qualidade do ar naquela data. Isso significa que 25% dos valores são menores que o valor q1.
- q3: Terceiro quartil dos dados de qualidade do ar naquela data. Isso significa que 75% dos valores são menores que o valor q3.
- stdev: Desvio padrão dos dados de qualidade do ar naquela data. O desvio padrão é uma medida de quanto dispersos estão os valores.
- count: Quantidade de observações que foram feitas naquela data.

4) **Air Quality Index:** API que permite o acesso a informações sobre a qualidade do ar em tempo real e previsões para vários locais. Usa uma escala colorida para indicar diferentes níveis de qualidade do ar, de "Bom" (Verde) a "Perigoso" (Roxo). A AQI fornece dados para poluentes como PM2.5, PM10, CO, SO<sub>2</sub>, NO<sub>2</sub>, O<sub>3</sub>, entre outros.

5) **Visual Crossing:** Plataforma de análise e previsão do tempo que fornece uma API de dados meteorológicos desde dados históricos a atuais e de previsão para qualquer local do mundo. Oferece também a capacidade de visualizar dados de previsão numa variedade de gráficos.

#### B. Pré-processamento dos dados

Dado que obtivemos os dados de diferentes fontes foi necessário proceder ao pré-processamento dos dados.

Primeiramente, procedemos à leitura de dois conjuntos de dados de CSVs para a cidade de Braga, um para PM10 e outro para PM2.5. Adicionamos as colunas de latitude e longitude para as coordenadas utilizadas pela **OpenAir**. De seguida, selecionamos apenas as colunas que consideramos relevantes ('date', 'latitude', 'longitude', 'median') e renomeia a coluna 'median' para 'pm10' ou 'pm25' dependendo do *dataset*. Num *dataset* denominado braga\_41.5549\_-8.4067.csv guardamos os resultados do merge de ambos os *datasets* anteriores com base na data, latitude e longitude. O processo de *merge* para os *datasets* relativos a Ponte de Lima e Vila Verde seguiram o processo descrito para a cidade de Braga.

Por fim, com os datasets relativos às três cidades prontos criamos uma coluna chamada 'parameter' que contém os tipos de partícula (PM1, PM10 e PM2.5) como valores, e colocando seus respectivos valores em outra coluna chamada 'value'. Isso é feito usando o método *melt()* do pandas.

O resultado foi um dataset com as *features* seguintes:

```
date,latitude,longitude,parameter,value
```

Para o *dataset* da **OpenAQ** relativo a Braga foi necessário selecionar as colunas relevantes ('utc', 'parameter', 'value', 'longitude' e 'latitude') Isto foi feito para simplificar o dataset e concentrar-se apenas nos dados que são relevantes para a sua análise bem como manter a coerência com os dados obtidos pelas outras fontes de dados. Renomeia-se as colunas para 'date', 'parameter', 'value', 'longitude', 'latitude', respectivamente e usando o método *pd.to\_datetime()* converte-se a coluna 'date' num objeto *datetime*, que é mais fácil de trabalhar quando se manipula ou analisa dados baseados no tempo.

Numa fase final do pré-processamento transformou-se o último *dataset* de um formato longo para um formato largo

para que cada tipo de partícula (pm1, pm10, pm25) tenha sua própria coluna. Depois disso, demos o *merge* de todos os quatro num único DataFrame.

De seguida, recorrendo ao dataset meteorológicos e encontrando as datas comuns entre o dataset meteorológicos e o dataset da qualidade do ar.

Foi necessário proceder à manipulação dos dados meteorológicos para o *dataset* final. Ou seja, se o dia é o mesmo, os atributos 'tempmax', 'tempmin', 'temp', 'feelslike', 'dew', 'humidity', 'severerisk', 'precip', 'snow', 'windspeed', 'sealevelpressure', 'cloudcover', 'visibility', 'solarradiation', 'solarenergy', 'uvindex' são os mesmos.

### C. Visualização dos dados

Para proceder à visualização de dados usamos várias bibliotecas Python, incluindo pandas para manipulação de dados, numpy, matplotlib e seaborn para visualização de dados, scipy, sklearn, entre outras.

1) *Descrição do dataset*: Recorrendo aos métodos `.describe()` e `.info()` obteve-se um resumo estatístico e informações básicas (como o número de linhas, colunas e tipos de dados) do conjunto de dados, respectivamente.

2) *Verificação de ocorrências*: Criou-se um ficheiro `occurrences.txt` que armazena a contagem de valores para cada coluna no conjunto de dados. Essa contagem de valores pode ajudar a identificar valores comuns, valores únicos e potenciais outliers.

Procedeu-se também à verificação da existência de valores duplicados bem como de *missing values*. No caso dos *missing values* criou-se um mapa de calor para visualizar onde os valores nulos se encontram no *dataset* de forma a entender se há algum padrão nestes valores.

3) *Visualização da distribuição geográfica*:

4) *Visualização de séries temporais de poluentes*: Recorrendo a gráficos de linha para visualizar as séries temporais de diferentes poluentes, conclui-se que

5) *Visualização de séries temporais de outras colunas*:

6) *Visualização de outliers*: Recorreu-se a boxplots para cada coluna (exceto 'date', 'time', 'latitude' e 'longitude') para visualizar a distribuição de seus valores e identificar possíveis outliers.

### D. Tratamento dos dados

No tratamento dos dados removemos as *features* 'feelslike', 'uvindex' e 'solarradiation' pelo facto de terem correlação muito alta com outras *features*.

1) *Missing Values*: A abordagem por remoção de linhas, embora fácil de implementar, leva à perda de muitas informações, especialmente neste caso em que devido ao *merge* há colunas com muitos valores em falta.

Efetuamos o cálculo de estatísticas para 'pm25' e 'pm10' imprimindo o valor mínimo, máximo e médio.

Ponderamos uma abordagem baseada na substituição dos valores em falta pelo valor médio do poluente mas, na nossa opinião, esta abordagem não faz sentido dado que temos dados

de diferentes coordenadas, em diferentes dias que não estão necessariamente relacionados entre si.

Efetuamos uma pesquisa e, em séries temporais é comum recorrer-se a Last Observation Carried Forward (LOCF) em que o último valor observado é usado para preencher o próximo valor em falta.

No entanto consideramos que o Iterative Imputer é a abordagem com mais sentido neste contexto porque é uma técnica com baixo peso computacional de imputação multivariada que considera a correlação entre as variáveis. Neste caso é particularmente útil porque a poluição do ar e as condições meteorológicas são processos interdependentes.

**Iterative Imputer** é uma técnica de imputação que modela cada recurso com *missing values* como uma função das restantes *features* e usa essa estimativa para imputação.

Para cada *feature* com *missing values* o IterativeImputer considera todas as outras características para estimar os valores de forma iterativa: a cada iteração, uma *feature* é considerada como *label* (y) e todas as outras como *input* (X).

Para efetuar este algoritmo usamos um Iterative Imputer da biblioteca sklearn com um máximo de 10 iterações e uma semente aleatória definida como 0 para garantir a reprodutibilidade. A imputação é realizada nas colunas 'no2', 'pm10', 'pm25' e 'pm1'.

2) *Outliers*: Para o tratamento de *outliers* exploraram-se algumas abordagens mas optamos por não alterar os *outliers* porque no contexto deste trabalho os *outliers* contêm significado dado que a qualidade do ar e concentrações de poluentes são, normalmente, constantes. Ou seja, os *outliers* podem representar eventos extremos, como picos na poluição do ar devido a eventos não usuais, como incêndios florestais, acidentes industriais, ou variações meteorológicas extremas, que têm impacto significativo na qualidade do ar.

Portanto, embora os outliers possam ser um problema, neste caso, fornecem informações valiosas e ajudam a criar um modelo de previsão mais realista e robusto.

### E. Modelos de aprendizagem

1) *ARIMA*: O modelo ARIMA (*Autoregressive Integrated Moving Average*) é uma abordagem amplamente utilizada para análise e previsão de séries temporais, que combina elementos de modelos autorregressivos (AR), modelos de médias móveis (MA) e diferenciação. Estes elementos permitem lidar com a natureza complexa das séries temporais, levando a que se torne uma ótima escolha de algoritmo para o nosso caso de estudo.

A própria sigla ARIMA refere-se a três componentes principais:

- **Autoregressive (AR)**: componente que incorpora a dependência linear das observações anteriores. Desta forma, o valor atual da série temporal é obtido por combinação linear dos seus valores anteriores, conhecidos como termos autor-regressivos. A ordem do componente AR, denotada por p, indica o número de observações anteriores a serem consideradas;

$$X(t) = c + \phi_1 X(t-1) + \phi_2 X(t-2) + \dots + \phi_p X(t-p) + \varepsilon(t)$$

Nesta equação,  $X(t)$  representa o valor atual da série temporal,  $\phi_1, \phi_2, \dots, \phi_p$  são os coeficientes autorregressivos, que indicam o impacto das observações anteriores na atual,  $p$  é a ordem do componente AR (número de observações anteriores consideradas),  $\varepsilon(t)$  é o termo de erro no tempo  $t$  e  $c$  é uma constante estimada normalmente durante o processo de treino

- **Integrated (I):** componente que lida com a estacionariedade da série temporal. A estacionariedade é uma propriedade desejada em muitas análises de séries temporais, pois implica que as propriedades estatísticas da série permanecem constantes ao longo do tempo. A diferenciação é aplicada à série para torná-la estacionária. A ordem de diferenciação, denotada por  $d$ , indica o número de vezes que a série precisa de ser diferenciada, de modo a tornar-se estacionária;

$$Y(t) = X(t) - X(t-d)$$

Nesta equação,  $Y(t)$  é a série temporal diferenciada de ordem  $d$ ,  $X(t)$  é a série temporal original e  $d$  é a ordem de diferenciação (número de vezes que a série precisa ser diferenciada) para torná-la estacionária.

- **Moving Average (MA):** componente que modela a média móvel dos erros de previsão anteriores. Desta forma, assume-se que o valor atual da série temporal é uma combinação linear de erros de previsão anteriores, conhecidos como termos de médias móveis. A ordem do componente MA, denotada por  $q$ , indica o número de erros anteriores a ser considerado.

$$X(t) = \mu + \theta_1\varepsilon(t-1) + \theta_2\varepsilon(t-2) + \dots + \theta_q\varepsilon(t-q) + \varepsilon(t)$$

Nesta equação,  $X(t)$  representa o valor atual da série temporal,  $\mu$  é a média da série,  $\theta_1, \theta_2, \dots, \theta_q$  são os coeficientes das médias móveis, que indicam o impacto dos erros de previsão anteriores no atual,  $q$  é a ordem do componente MA (número de erros passados considerados) e  $\varepsilon(t)$  é o termo de erro no tempo  $t$ .

Antes de proceder ao treino propriamente dito do modelo ARIMA, foi necessária efetuar algumas alterações aos dados de *input*, de forma a tornar possível a previsão no *dataset* de validação (*forecast*). Assim sendo, esse conjunto de alterações segue-se adiante:

- 1) **Remoção de colunas desnecessárias:** Uma vez que o *dataframe* já estava indexado por *datetime* no formato YYYY-MM-DD HH:MM:SS, removeram-se as colunas individuais *date* e *time*. A coluna *snow* continha todos os casos iguais a 0, não acrescentando conhecimento para o treino do modelo, pelo que, foi retirada de igual modo: **Remoção de colunas que não constam nos dados de validação:** Uma vez que deve existir coerência de *features* entre os dados de treino e os dados de validação, o modelo deve ser treinado com os mesmos atributos que os dados de validação. Então, após obtenção dos dados de validação, observou-se que algumas colunas estavam vazias, pelo que não poderiam ser utilizadas

durante a previsão, tendo sido removidas. Essas colunas consistiam em dados meteorológicos ou de poluentes que não constavam em ambos os datasets. No final, ambos os datasets continham as mesmas *features*.

Passou-se à divisão dos dados em 80% de dados de treino e os restantes para teste. Como se referiu anteriormente, um dos pontos mais importantes para o correto uso do modelo consiste na seleção dos parâmetros de ordem de cada componente ( $p, d, q$ ):

- **$d$ :** determinou-se através do teste de Augmented Dickey-Fuller (ADF) para estacionariedade, se o  $p-value$  for superior ao nível de significância (0.05), então, a série temporal não é estacionária e precisa de diferenciação, até se tornar estacionária. O teste ADF teve um valor de -5.690062146472456. Quanto mais negativo (menor) for o valor absoluto, mais evidência existe de que a série é estacionária. Neste caso, há uma forte evidência de estacionariedade na série temporal. O valor  $p$  é 8.109855336789639e-07, que é extremamente baixo. O valor  $p$  representa a probabilidade associada à estatística ADF. Como o valor  $p$  é menor que o nível de significância pré-determinado (0.05), rejeita-se a hipótese nula de que a série é não estacionária e considera-se que a série é estacionária. Neste caso, o valor  $p$  muito baixo (8.109855336789639e-07) indica uma forte evidência para rejeitar a hipótese nula e concluir que a série temporal é estacionária, não prescindendo de diferenciação, daí que  $d = 0$ .
- **$p$  e  $q$ :** determinaram-se através da Função de Autocorrelação (ACF) e Função de Autocorrelação Parcial (PACF). A ACF mede a correlação entre uma observação e suas observações anteriores em diferentes desfasagens. A PACF mede a correlação entre uma observação e suas observações anteriores, excluindo a influência das observações intermediárias. Através da análise dos gráficos ACF e PACF, verificou-se quando ocorria um pico de valores seguido de uma queda abrupta para valores próximos de 0. Utilizou-se esse valor para cada ordem,  $p = 0$  (há descida contínua) e  $q = 25$  (correlação é 0 a partir de 26).

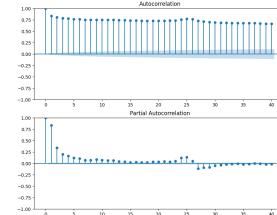


Fig. 2. Estimação dos parâmetros de ordem (p,d,q).

De seguida, efetuou-se a previsão nos dados de teste e obtiveram-se as métricas de regressão relevantes (MAE, RMSE e R2). De seguida, previu-se o Índice de Qualidade do Ar para os 12 dias seguintes (2023-05-01 a 2023-05-12). Através da análise destas métricas, concluiu-se que o modelo

não tinha as melhores capacidades para prever, obtendo-se um MAE de 62.4787. De forma a atenuar este erro, efetuou-se *grid-search* dos parâmetros de ordem de cada componente, tendo-se utilizado estes mesmos parâmetros na implementação do algoritmo com *sliding window*. Desta forma, o erro passou para 40.3, tendo-se obtido, então, uma melhoria.

Com os resultados obtidos para as previsões no intervalo temporal definido, foi criado um objeto *geoJSON*, com as coordenadas e os respetivos valores do AQI. O *website* criado, apresenta, numa camada de calor do mapa, este conjunto de previsões na respetiva cidade. É possível, então, observar a camada de calor dos dados utilizados para treino, e, numa outra camada, observar os dados previsto, tirando conclusões.

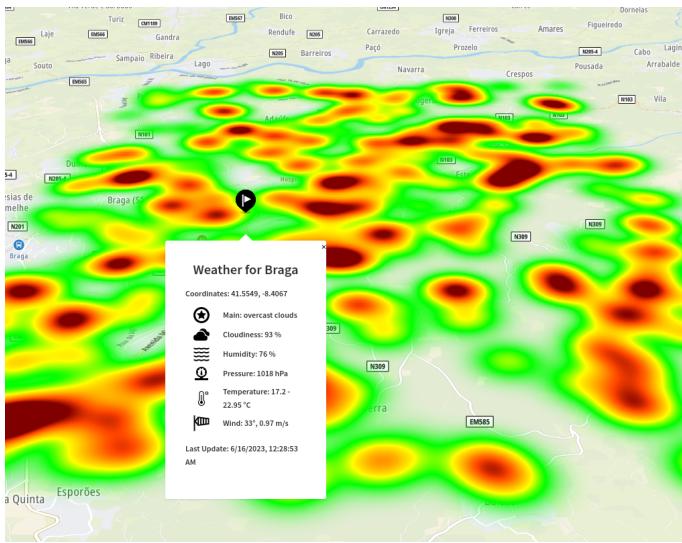


Fig. 3. Mapa de calor de Braga com os valores de AQI previstos com o ARIMA.

2) *LSTM*: Do mesmo modo foi também estudado e explorado o modelo LSTM (*Long short-term memory*). Exploramos este modelo uma vez que o que o diferencia de outras RNNs é a sua capacidade de armazenar e recuperar informações ao longo de períodos de tempo mais longos. O modelo consegue fazer isso uma vez que usa uma estrutura de célula de memória interna, que é composta por unidades de memória chamadas "células" interconectadas. Cada célula possui várias camadas e componentes que permitem que ela armazene, atualize e passe as informações ao longo da sequência. Dentro de uma célula LSTM, existem assim três componentes principais que são importantes distinguir:

- **Célula de memória:** é responsável por armazenar informações ao longo do tempo. Possui uma estrutura interna que permite que as informações sejam adicionadas ou removidas conforme necessário. Essa célula é projetada para reter informações relevantes e descartar informações menos importantes.
- **Input gate:** controla o fluxo de informações novas que entram na célula de memória. Decide que informações devem ser armazenadas e atualizadas na memória com base na entrada atual e na memória anterior. O *Input gate*

é ativado pela função de ativação sigmoidal, que decide a importância das informações.

- **Forget gate:** permite que a célula de memória descarte informações desnecessárias ou irrelevantes. Decide que informações antigas devem ser mantidas e quais devem ser esquecidas. O *Forget gate* é ativado pela função de ativação sigmoidal e determina a taxa de esquecimento das informações.

Esses componentes trabalham em conjunto para permitir que as LSTMs capturem dependências de longo prazo em sequências. As informações fluem através dos *gates*, permitindo que a célula de memória seja atualizada com base nas entradas atuais e na memória anterior. A saída da célula de memória é controlada pelo *forget gate*, que determina qual parte da memória será usada como saída da LSTM.

Desse modo, tal como no modelo ARIMA, foi necessário efetuar algumas alterações aos dados de *input*, de forma a tornar possível a previsão no *dataset* de validação (*forecast*). Assim sendo, esse conjunto de alterações segue-se adiante:

- 1) **Remoção de colunas desnecessárias:** Uma vez que o *dataframe* já estava indexado por *datetime* no formato YYYY-MM-DD HH:MM:SS, removeram-se as colunas individuais *date* e *time*. A coluna *snow* continha todos os casos iguais a 0, não acrescentando conhecimento para o treino do modelo, pelo que, foi retirada de igual modo;
- 2) **Remoção de colunas que não constam nos dados de validação:** Uma vez que deve existir coerência de *features* entre os dados de treino e os dados de validação, o modelo deve ser treinado com os mesmos atributos que os dados de validação. Então, após obtenção dos dados de validação, observou-se que algumas colunas estavam vazias, pelo que não poderiam ser utilizadas durante a previsão, tendo sido removidas. Essas colunas consistiam em dados meteorológicos ou de poluentes que não constavam em ambos os datasets. No final, ambos os datasets continham as mesmas *features*.
- 3) **Verificação das features:** teste para verificar se os conjuntos de dados de treinamento e validação possuem as mesmas características.
- 4) **Definição dos dados:** Define os dados de treino e validação, separando as variáveis de entrada (X) e as variáveis de saída (y).
- 5) **Normalização:** normalização dos dados de entrada usando a média e o desvio padrão dos dados de treino.
- 6) **Redimensionamento:** redimensionamento dos dados de entrada para o formato necessário para o modelo LSTM. Os dados de treino e validação são redimensionados usando o método *reshape()*. O objetivo é adicionar uma dimensão adicional aos dados, de modo que eles fiquem no formato (amostras, tempo, recursos). A primeira dimensão representa as amostras, a segunda dimensão representa o tempo (neste caso, temos apenas um ponto de tempo) e a terceira dimensão representa os recursos (neste caso, as características dos dados).

Relativamente ao treino do modelo utilizamos diferentes

arquiteturas com mais ou menos complexidade, ajustando também os devidos hiperparâmetros.

Após a definição da arquitetura do modelo, foi necessário compilar o modelo antes de treiná-lo. Aqui, utilizamos a função de perda *mean squared error*, que é comumente usada para problemas de regressão, como prever um valor numérico como o AQI. O otimizador escolhido foi o Adam, pelo que também utilizamos diferentes taxas de aprendizagem.

Após o fit, fizemos as previsões e compilamos no gráfico seguinte:

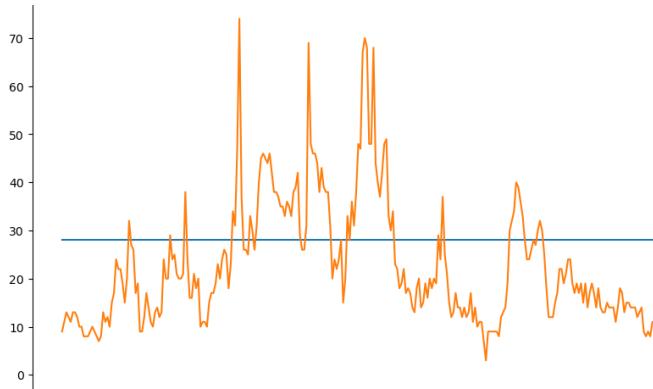


Fig. 4. Gráfico com os valores atuais e previstos pelo modelo LSTM.

Foi possível verificar uma convergência do valor previsto, mesmo após uma atualização dos hiperparâmetros, épocas, *batches*, complexidade das camadas do modelo, pelo que a solução encontrada foi que esta arquitetura do modelo inadequada para os nossos dados. Isto é, a arquitetura LSTM não é adequada para capturar a complexidade e as relações dos dados em questão pelo que mantivemos o nosso tratamento mais aprofundado com o modelo ARIMA uma vez que este se adapta melhor aos nossos dados.

#### F. Implementação de sistema web

Para integrar as fontes de dados utilizadas e os modelos de previsão desenvolvidos desenvolvemos um sistema *web* recorrendo às ferramentas React, JS, Firebase, Express e NodeJS.

Por uma questão de gestão do tempo não foi possível efetuar o *deployment* do sistema de forma a ser possível aceder de forma direta através de um url. Como tal, para visualizar a implementação do sistema *web* é necessário correr o comando

```
npm start
```

na pasta de back-end, aguardar que este recolha todos os dados em tempo real das APIs e de seguida correr o mesmo comando na pasta de front-end.

1) *Front-end*: No *front-end* optamos por um *design* simples com três páginas: página inicial, mapa e modelo.

A página inicial dispõe informações acerca do objetivo do trabalho bem como uma imagem ilustrativa.

Na secção do mapa temos um mapa mundo com possibilidade de dispor dados relativos ao trânsito (obtidos pela API

da TomTom - em determinada altura no projeto pensamos em relacionar o trânsito em determinada área com a qualidade de ar na mesma), um mapa de calor com as diferentes coordenadas e qualidade de ar nas mesmas e pontos de meteorologia com os dados da OpenWeather.

Por fim, a página relativa ao modelo dispõe a informação relativa ao plano de ação.

2) *Back-end*: O *back-end* está construído em código Node.js que usa o Express para criar rotas para aceder aos dados.

A rota `'map'` responde aos pedidos GET para `'/map'` que obtém os dados mais recentes e o AQI (Índice de Qualidade do Ar) para uma lista de cidades, e cria um objeto GeoJSON com esses dados.

A rota `'weather'` responde a pedidos GET para `'/maps/weather'`. É devolvido o registo mais recente de dados meteorológicos para uma lista de cidades.

Por fim, a rota `'air'` responde aos pedidos GET para `'/maps/air'` e obtém os dados mais recentes e o AQI para uma lista de cidades.

A base de dados está realizada em Firebase.

3) *Interfaces*: Resta nos por isso demonstrar a aplicação construída através das suas diversas interfaces.

As boas-vindas são dadas com imponente escadório do Bom Jesus. Este foi escolhido porque é o pulmão central da cidade de Braga cada vez mais poluída e em constante crescimento insustentável.



Fig. 5. Menu Inicial.

Na rota do Mapa encontramos o mapa mundial, onde é possível explorar o mesmo através das duas visões distintas (de frente e de cima). O nosso foco será contudo Portugal e a cidade de Braga pelo através das leituras em tempo real dos diversos sensores construímos uma tabela indicador com os dados recolhidos para cada cidade. São assim indicadas as coordenadas, o tempo, precipitação, nebulosidade, temperatura, vento e a data da última alteração.

Na figura abaixo vemos o exemplo para Santarém.



Fig. 6. Mapa com as diversas cidades sendo apresentada a tabela-indicador de Santarém.

Por outro lado, e uma vez que utilizamos diversas APIs distintas para a recolha de dados, incorporamos a congestão de tráfego em tempo real para as diversas rotas do mapa, sendo possível ver pela legenda a intensidade do mesmo em cada região de interesse. Na figura seguinte, é possível evidenciar o supracitado bem como a vista de cima do modelo.

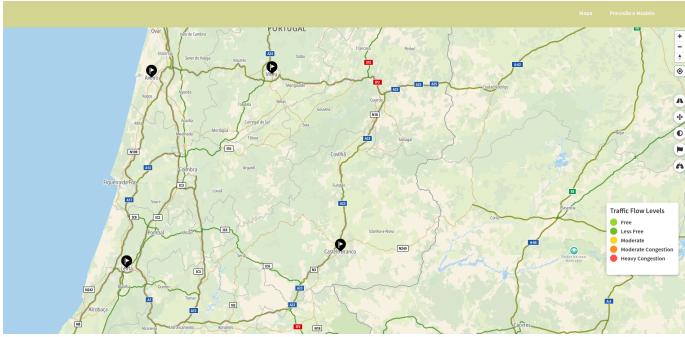


Fig. 7. Vista de cima sendo possível evidenciar a congestão do tráfego

Segue-se como cerne do nosso projeto a análise à qualidade do ar, sendo a métrica o AQI, pelo que através do processamento de todos os dados recolhidos construímos o mapa de calor para Portugal de modo a evidenciar as zonas mais afetadas pela pobre qualidade do ar. Esta visualização é acompanhada pela respetiva legenda com os níveis do AQI. É possível verificar na figura que como esperado as zonas do Grande Porto e da Grande Lisboa são as mais afetadas.

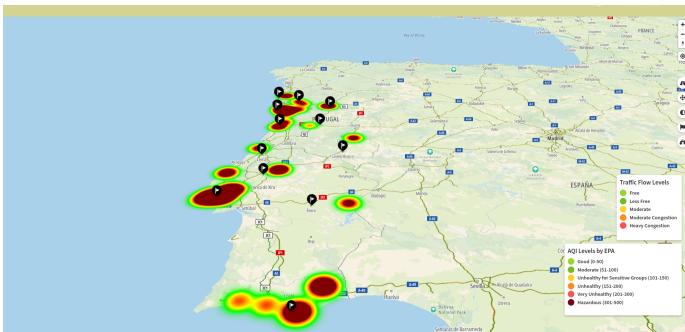


Fig. 8. Mapa de calor da qualidade do ar.

Por fim e tendo em vista o objetivo principal inicialmente proposto, criamos o mapa de calor com os valores previstos da qualidade do ar para a cidade de Braga através da utilização do modelo ARIMA, sendo esta previsão detalhadamente explicada anteriormente.

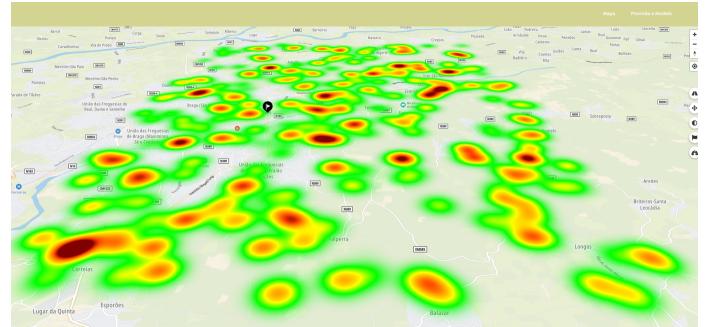


Fig. 9. Mapa de calor com os valores previstos da qualidade do ar para a cidade de Braga.

## V. CONCLUSÃO

Em suma, estamos bastante contentes com o trabalho desenvolvido e com os resultados obtidos, seguindo as técnicas aprendidas neste que foi um contacto mais profundo com o tratamento e mineração dos dados, num momento em que "os dados são o novo petróleo" e estas áreas de estudo são cada vez mais promissoras e poderosas.

Contudo, poderíamos estender o trabalho com novas funcionalidades de modo a aprimorar a aplicação e a extração de conhecimento.

### A. Trabalho futuro

Numa fase posterior e com o intuito de expandir e melhorar o nosso trabalho seriam propostos desenvolvimentos no que diz respeito a:

- Relações com AQI:** Seria interessante explorar mais a fundo as relações entre diferentes variáveis e o AQI. Por exemplo, poderíamos analisar a relação entre as condições meteorológicas (como temperatura, umidade e velocidade do vento) e o AQI. Por outro lado, considerávamos relevante explorar o impacto do trânsito em áreas urbanas no índice de qualidade do ar.
- Teste de hipóteses:** Além disso, poderíamos realizar testes de hipóteses estatísticas para confirmar ou refutar as nossas suposições sobre as relações entre diferentes variáveis e o AQI.
- Obtenção de dataset futuro para previsão:** Para fazer previsões futuras mais precisas, precisamos de datasets com dados futuros. Isto poderia incluir previsões meteorológicas futuras. Desta forma seria possível tentar prever o AQI conforme os dados previstos e explorar esta relação (ainda que possa ser prejudicado pelo acúmulo de previsões a que estaríamos sujeitos)