Universidade do Minho
Escola de Engenharia

# Dados e Aprendizagem Automática
## Grupo 31
## 2023/2024

Ana Filipa da Cunha Rebelo pg53624
André Castro Alves pg53651
Maria Eugénia Bessa Cunha pg54042
Tomás Cardoso Francisco pg54263

Janeiro 2024

# Índice

# List of Figures

# 1 Introdução

Neste trabalho realizado no âmbito da UC de *Dados e Aprendizagem Automática* foi proposta a exploração, modelação e análise de dois *datasets*.

O primeiro conjunto de dados foi escolhido pelo grupo e tem como foco o objetivo de compreender e extrair informações relevantes relacionadas à ocorrência de ataques cardíacos para prever se os pacientes sofrerão ou não de um ataque cardíaco no futuro.

O outro, escolhido pelos docentes, tem o objetivo de prever, com precisão, a quantidade de energia elétrica, em *kWh*, gerada por painéis solares e injetada na rede elétrica, a cada hora do dia. Assim, faz-se uso de uma grande diversidade de atributos, que vão desde dados meteorológicos e informações geográficas, a históricos de gasto e produção energética elétrica para concluir esta previsão.

O presente relatório apresenta uma análise detalhada de todo o procedimento efetuado em cada um dos *datasets*, até à aplicação dos modelos de *Machine Learning*. Todas as etapas serão documentadas devidamente, sendo possível observar quais os métodos de visualização e exploração de dados escolhidos, assim como os modelos de aprendizagem aplicados e os respetivos resultados.

# 2 *Dataset* Livre

## 2.1 Exploração

A primeira etapa da realização deste trabalho consistiu em explorar o *dataset* de forma a prepará-lo para posteriormente aplicar os modelos de *Machine Learning*. Além disso, a exploração dos dados também auxilia na escolha do modelo de extração de conhecimento mais adequado. A aplicação de técnicas de preparação dos dados é essencial dado que os *datasets* do mundo real contém muitas inconsistências, podem ser incompletos e conter lixo.

A primeira ação tomada para cada um dos *datasets* na exploração dos dados foi a observação do *dataset* e a análise de cada atributo que o constitui, observando a contagem dos registos de cada um. Nesta secção é possível observar qual o tratamento de dados aplicado a cada *dataset* e os métodos de visualização e exploração de dados que conduziram a uma dada decisão.

Inicialmente, explorámos todos os atributos e informação relevante do dataset escolhido. Este contém 8763 linhas e 26 colunas.

```
Index(['Patient ID', 'Age', 'Sex', 'Cholesterol', 'Blood Pressure',
       'Heart Rate', 'Diabetes', 'Family History', 'Smoking', 'Obesity',
       'Alcohol Consumption', 'Exercise Hours Per Week', 'Diet',
       'Previous Heart Problems', 'Medication Use', 'Stress Level',
       'Sedentary Hours Per Day', 'Income', 'BMI', 'Triglycerides',
       'Physical Activity Days Per Week', 'Sleep Hours Per Day', 'Country',
       'Continent', 'Hemisphere', 'Heart Attack Risk'],
      dtype='object')
```

Figure 1: Colunas do *Dataset* Livre

Depois de uma breve observação dos dados presentes decidimos prosseguir para a fase de pré processamento de dados. Isto é, removemos as colunas '*Patient ID*' devido à sua irrelevância, e '*Hemisphere*' devido à sua redundância.

Seguidamente, testamos a presença de valores nulos dentro dos dados, mas como verificámos que não existiam, não foi necessário proceder a nenhum tratamento destes.

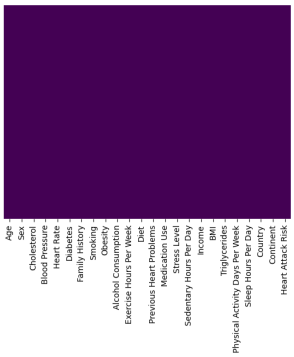


Figure 2: Nenhum valor nulo detetado

Posteriormente, decidimos investigar a coluna 'Blood Pressure', nomeadamente como as organizações médicas classificam os variados valores.

***Categorização da coluna Blood Pressure por categoria*** A coluna *Blood Pressure* é uma variável importante que tem que ser classificada com alguns detalhes em mente.

Neste momento, os valores desta coluna estão a seguir o seguinte formato: Pressão Sistólica/Diastólica. Assim sendo, este formato não nos é muito útil. Deste modo, recorremos a classificações categóricas, recorrendo a informações provinientes das seguintes instituições : *WHO* (*World Health Organization*), *Harvard Health* e *American Heart Association*.

**Normal** : menos de 120 Pressão Sistólica(mm Hg) e menos de 80 Pressão Diastólica(mm Hg).

**Elevated** : Pressão Sistólica entre 120-129 e menos de 80 Pressão Diastólica.

**High Blood Pressure (Hypertension) Stage 1** : Pressão Sistólica entre 130-139 ou Pressão Diastólica entre 80-89.

**High Blood Pressure (Hypertension) Stage 2** : 140 ou mais de Pressão Sistólica ou Pressão Diastólica a 90 ou mais.

**Hypertensive Crisis :** Pressão Sistólica a 180 ou mais e/ou Pressão Diastólica a 120 ou mais.

Figure 3: Classificação de valores de pressão sanguínea

Assim, tendo em atenção a estes valores, decidimos investigar os dados que nos foram fornecidos de modo a identificar se havia propensidade nos pacientes do *dataset* a serem identificados com pressão arterial mais elevada (podendo ter uma correlação positiva com ataques cardíacos).

Como podemos observar através dos resultados obtidos, só os pacientes no grupo de idade *60+* estão dentro da média apropriada para pressão sanguínea de acordo à sua idade e sexo. Assim, podemos concluir que a maior parte destes pacientes tem propensidade a ter um ataque cardíaco.

```
Blood pressure means on men (age group 18-39) :
 Systolic 134.558413 ; Diastolic 84.931247
Blood pressure means on women (age group 18-39) :
 Systolic 135.510740 ; Diastolic 85.658711

Blood pressure means on men (age group 40-59) :
 Systolic 134.773538 ; Diastolic 85.468308
Blood pressure means on women (age group 40-59) :
 Systolic 135.515328 ; Diastolic 85.979562

Blood pressure means on men (age group 60+) :
 Systolic 135.363367 ; Diastolic 85.067853
Blood pressure means on women (age group 60+) :
 Systolic 135.053144 ; Diastolic 84.401240
```

Figure 4: Classificação de valores de pressão sanguínea

Depois de executada esta análise seguiu-se o *encoding* de dados para colunas relevantes. A coluna '*Blood Pressure*' do tipo *object* foi dividida em duas : '*Systolic Pressure*' e '*Diastolic Pressure*' a conter os devidos valores em formato *int64*. Similarmente, procedeu-se ao *hot encoding* das colunas '*Sex*', '*Diet*' e '*Age_Group*'. Por fim, também se recorreu a um *label encoder* para colunas como '*Country*' e '*Continent*', resultando assim num código pronto a ser modelado.

Finalmente, concluímos a exploração dos dados com uma visualização das correlações entre os dados presentes no *dataset*.

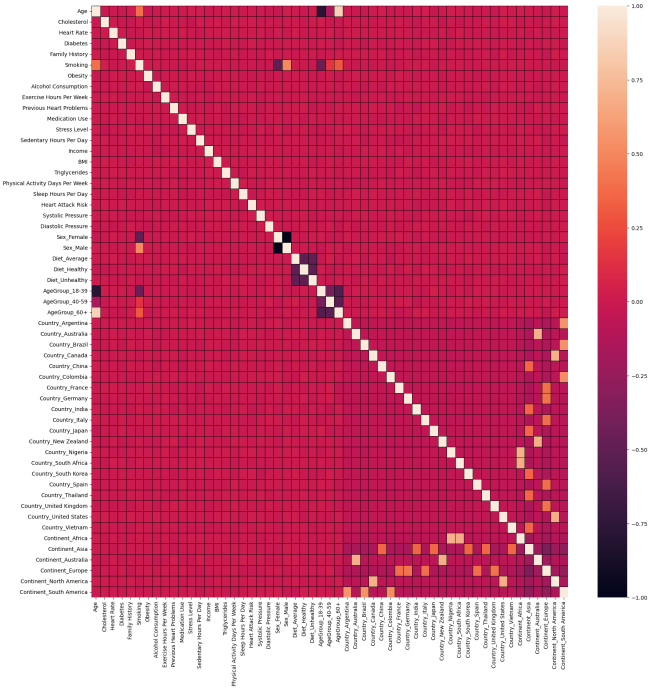

Figure 5: Matriz de Correlação

Conseguimos concluir que há uma leve correlação positiva entre todos os pacientes de sexo masculino e pacientes dentro dos grupos de idade *40-59* e *60+* e a coluna '*Smoking*'. Similarmente, há uma forte correlação negativa entre os pacientes de sexo feminino, os pacientes dentro do grupo de idades *18-39* e essa mesma coluna.

## 2.2 Análise de Dados e Modelação

O problema a ser tratado deste *dataset* é um de classificação binária, assim, tendo esse aspeto em consideração, o grupo escolheu os seguintes métodos para modelação: ***Decision Tree Classifier***, ***Support Vector Classifier*** (**SVC**), ***SVC com acesso a GridSearchCV***, ***Random Forest Classifier com acesso a GridSearchCv*** e, por fim, ***Logistic Regression***.

### 2.2.1 Decision Tree Classifier

*Decision Tree Classifier* é um poderoso algoritmo de *machine learning* supervisionado, especialmente adequado para problemas de classificação binária, como a previsão de ataques cardíacos que pretendemos atingir. A eficácia deste método reside na capacidade de manipular *datasets* de forma eficiente, dividindo-os em subconjuntos menores com base em decisões criteriosas sobre os valores das colunas.

Assim, o algoritmo constrói uma estrutura de árvore, onde os ramos representam as condições de decisão e as folhas denotam as classes ou valores previstos. Esta organização hierárquica proporciona uma interpretação visual intuitiva do processo de classificação, conferindo uma vantagem significativa em termos de compreensão e explicabilidade do modelo.

A versatilidade deste método permite a atribuição de uma leve importância em certas *features*, facilitando a identificação daquelas que desempenharam um papel crucial no processo de decisão. Este aspeto é crucial para a compreensão do modelo, pois permite-nos identificar os principais contribuintes para as previsões, fornecendo conclusões valiosas sobre os fatores de risco associados a ataques cardíacos.

Assim, tendo em conta todas estas características, consideramos este modelo uma escolha robusta e eficaz para abordar desafios de classificação binária em contextos clínicos.

Adicionalmente, também fizemos modelação com este método em combinação com *cross validation* (k=10). *Cross validation* é um método essencial para avaliar o desempenho de modelos. Neste caso em concreto, o uso desta técnica pode reduzir a variância, aprimorar a generalização, auxiliar na sintonia otimizada de hiperparâmetros e ajudar a detectar o *overfitting*, proporcionando uma estimativa de desempenho mais confiável. Assim tomamos esta decisão pois acreditamos que a combinação do modelo com esta técnica assegura efetivamente uma avaliação mais robusta e um aprimoramento das capacidades do modelo.

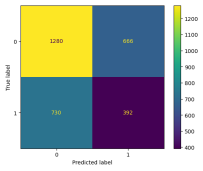


Figure 6: Matriz de Confusão de *Decision Tree Classifier*

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.64 | 0.66 | 0.65 | 1946 |
| 1 | 0.37 | 0.35 | 0.36 | 1122 |
| accuracy | | | 0.54 | 3068 |
| macro avg | 0.50 | 0.50 | 0.50 | 3068 |
| weighted avg | 0.54 | 0.54 | 0.54 | 3068 |

Figure 7: Matriz de Confusão de *Relatório de Classificação referente à Decision Tree Classifier*

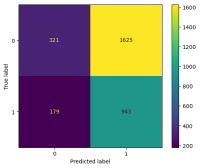


Figure 8: Matriz de Confusão de *Decision Tree Classifier* com *Cross Validation* (k=10)

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.64 | 0.16 | 0.26 | 1946 |
| 1 | 0.37 | 0.84 | 0.51 | 1122 |
| accuracy | | | 0.41 | 3068 |
| macro avg | 0.50 | 0.50 | 0.39 | 3068 |
| weighted avg | 0.54 | 0.41 | 0.35 | 3068 |

Figure 9: Third figure.

Rapidamente conseguimos concluir que a performance é melhor com o auxílio de *cross validation* mas a discussão destas conclusões será guardada para a secção de *Resultados* presente neste relatório.

### 2.2.2 Support Vector Classifier (SVC)

O *Support Vector Classifier (SVC)* é um algoritmo projetado para encontrar um hiperplano ideal que separe eficientemente as classes de um conjunto de dados de previsão de ataques cardíacos. Destacando-se em tarefas de classificação tanto lineares quanto não lineares, o *SVC* identifica vetores de suporte cruciais para decisões precisas. Contudo, ao analisar o relatório de classificação, observam-se desafios, especialmente na busca por altos índices de precisão e *recall* para ambas as classes. Estas observações indicam áreas potenciais para otimização e a consideração de alternativas algorítmicas.

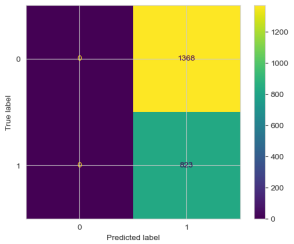


Figure 10: Matriz de Confusãio de SVC

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.00 | 0.00 | 0.00 | 1368 |
| 1 | 0.38 | 1.00 | 0.55 | 823 |
| accuracy | | | 0.38 | 2191 |
| macro avg | 0.19 | 0.50 | 0.27 | 2191 |
| weighted avg | 0.14 | 0.38 | 0.21 | 2191 |

Figure 11: Relatório de Classificação referente a SVC

### 2.2.3 SVC com acesso a GridSearchCV

A técnica de *Grid Search CV* é uma estratégia que visa otimizar o desempenho dos modelos através de *hypertuning* de *hyperparameters*. Este método realiza uma iteração sistemática para identificar a combinação mais eficaz de valores do modelo, contribuindo para aprimorar a eficiência durante o treino do *dataset*. A escolha desta abordagem é devido pela sua eficácia na otimização do modelo, aliada à implementação de validação, como o *cross validation*, que desempenha um papel crucial na prevenção do *overfitting*.

### 2.2.4 Random Forest Classifier com acesso a GridSearchCV

*Random Forest Classifier* é um método semelhante a *Decision Tree Classifier* pois opera da mesma maneira, mas com múltiplas árvores de decisão. Assim, consideramos que este modelo em conjunto com o uso de *GridSearchCV* tem potencial para apresentar melhores resultados.

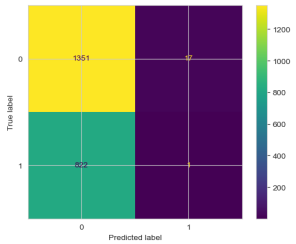


Figure 15: Matriz de Confusão de *Random Forest Classifier com GridSearchCV*

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.62 | 0.99 | 0.76 | 1368 |
| 1 | 0.06 | 0.00 | 0.00 | 823 |
| accuracy | | | 0.62 | 2191 |
| macro avg | 0.34 | 0.49 | 0.38 | 2191 |
| weighted avg | 0.41 | 0.62 | 0.48 | 2191 |

Figure 16: Relatório de classficação de *Random Forest Classifier com GridSearchCV*

### 2.2.5 Logistic Regression

*Logistic Regression* é um método destinado à resolução de problemas de classificação. Este algoritmo é particularmente eficaz na categorização de dados binários utilizando uma função logística para modelar a probabilidade de pertencer a uma determinada classe. É um método amplamente utilizado em diversos campos devido à sua

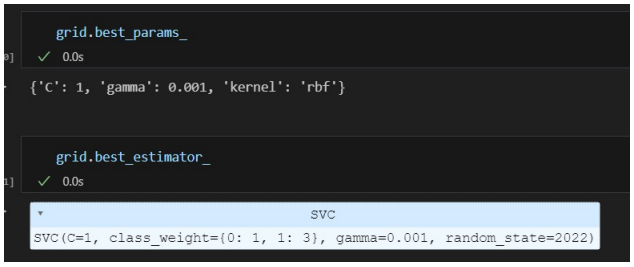


Figure 12: Melhores parâmetros e estimador para *SVC com GridSearchCV*

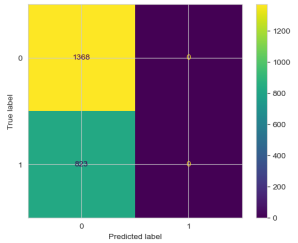


Figure 13: Matriz de Confusão de *SVC com GridSearchCV*

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.62 | 1.00 | 0.77 | 1368 |
| 1 | 0.00 | 0.00 | 0.00 | 823 |
| accuracy | | | 0.62 | 2191 |
| macro avg | 0.31 | 0.50 | 0.38 | 2191 |
| weighted avg | 0.39 | 0.62 | 0.48 | 2191 |

Figure 14: Relatório de classficação de *SVC com GridSearchCV*

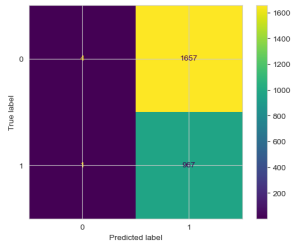


Figure 17: Matriz de Confusão de *Logistic Regression (solver: newton-cg)*

With solver 'newton-cgs' :

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.80 | 0.00 | 0.00 | 1661 |
| 1 | 0.37 | 1.00 | 0.54 | 968 |
| accuracy | | | 0.37 | 2629 |
| macro avg | 0.58 | 0.50 | 0.27 | 2629 |
| weighted avg | 0.64 | 0.37 | 0.20 | 2629 |

Figure 18: Relatório de classficação de *Logistic Regression (solver: newton-cg)*

simplicidade interpretativa e capacidade de generalização. Assim, decidimos fazer uso deste utilizando dois *solvers* distintos: *newton-cg* e *liblinear*.

## 2.3 Resultados

O nosso *dataset* tenta prever uma ocorrência médica e como tal, é mais imperativo a previsão correta de casos positivos mesmo que seja à custa de um maior número de casos falsamente positivos. Assim, ponderamos que a previsão da classe 1 é a mais importante e assim tendo só essa consideração, o modelo com mais sucesso será o modelo de *Support Vector Classifier*. Mesmo assim, tentando obter um modelo que não sacrifique totalmente a classificação correta de casos negativos, consideramos que o modelo ***Decision Tree Classifier com CV*** teve o melhor comportamento de todos, sendo este parecido ao comportamento de *SVC* na classificação de casos positivos, obtendo uma ***precisão correta 84% das vezes*** com **precisão de 37%** e para os casos negativos esta previsão é correta **16%** das vezes mas com uma **precisão de 64%**.

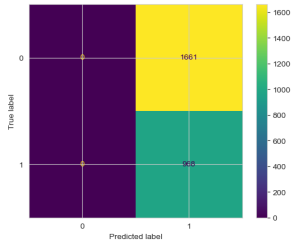


Figure 19: Matriz de Confusão de *Logistic Regression (solver: liblinear)*

With solver 'liblinear' :

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 1.00 | 0.00 | 0.00 | 1661 |
| 1 | 0.37 | 1.00 | 0.54 | 968 |
| accuracy | | | 0.37 | 2629 |
| macro avg | 0.68 | 0.50 | 0.27 | 2629 |
| weighted avg | 0.77 | 0.37 | 0.20 | 2629 |

Figure 20: Relatório de classficação de *Logistic Regression (solver: liblinear)*

| Classe 1 | Precisão | *Recall* |
|---|---|---|
| Decision Tree Classifier | 0.37 | 0.35 |
| Decision Tree Classifier com CV | 0.37 | 0.84 |
| Support Vector Classifier (SVC) | 0.38 | 1.00 |
| SVC com GridSearchCV | 0.00 | 0.00 |
| Random Forest Classifier com GridSearchCV | 0.06 | 0.00 |
| Logistic Regression Method | 0.37 | 1.00 |

Figure 21: Tabela com os resultados de Precisão e *Recall* para a **Classe 1** de cada modelo

| Classe 0 | *Precisão* | *Recall* |
|---|---|---|
| *Decision Tree Classifier* | 0.64 | 0.66 |
| *Decision Tree Classifier com CV* | 0.64 | 0.16 |
| *Support Vector Classifier (SVC)* | 0.00 | 0.00 |
| *SVC com GridSearchCV* | 0.62 | 1.00 |
| *Random Forest Classifier com GridSearchCV* | 0.62 | 0.99 |
| *Logistic Regression (solver newton-cg)* | 0.80 | 0.00 |
| *Logistic Regression (solver liblinear)* | 1.00 | 0.00 |

Figure 22: Tabela com os resultados de Precisão e *Recall* para a **Classe 0** de cada modelo

É de notar que não se efetuou a diferenciação dos diferentes *solvers* no método de *Logistic Regression* para a Classe 1 por não diferenciarem nos resultados.

# 3 *Dataset* de Competição

## 3.1 Exploração

Pelas mesmas razões, já identificadas na secção sobre o *dataset* alternativo, a exploração dos dados de um dataset para obter bons resultados é crucial. Assim, para a previsão da quantidade de energia foram utilizados seis datasets, três de energia e três de metereologia. Entre estes, dois de cada foram usados para treinar os modelos e os restantes usados para teste na competição. Os datasets de energia contém 6 recursos onde o usado para treino possui 2256

entradas, o de ajuste possui 8760 e o de teste também possui 2256. O datasets de metereologias contem 15 recursos onde o usado para treino tem 2928 entradas, o de ajuste possui 8760 e o de teste tem 1752.

### 3.1.1 Missing Values

Nesta secção nós seguimos predominante os seguintes 4 passos:

1. Remover valores em falta, valores anómalos e linhas/colunas desnecessárias
2. Verificar e imputar valores nulos

Primeiramente fomos verificar se havia valores em falta, e se sim como era as suas distribuições. Para isto realizamos o *display* de um *heatmap* de cada um dos *datasets*, como pode ser visto abaixo.

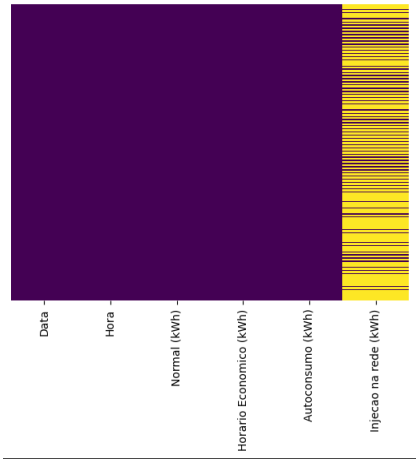


Figure 23: Missing values do dataset de energia de 2021

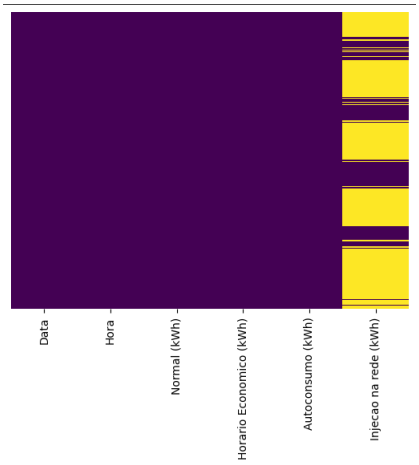


Figure 24: Missing values do dataset de energia de 2022

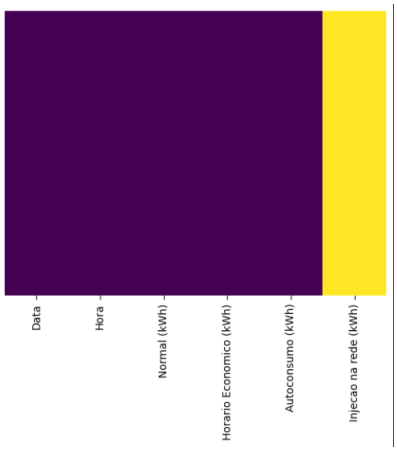


Figure 25: Missing values do dataset de energia de 2023

Como podemos ver acima, a única coluna que apresenta valores em falta é a coluna referente a "Injeção na rede (kWh)". Como estes valores não são realmente valores em falta, sendo que a coluna é uma categoria onde *None* tem valor real, logo resolvemos manter esta coluna. Para isto preenchemos a coluna com zeros ao invés de *None*.

A seguir é demonstrado a dispersão de valores em falta nos *datasets* de meteorologia.

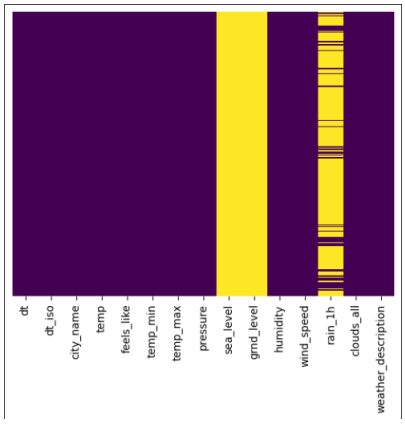


Figure 26: Missing values do dataset de meteorologia de 2021

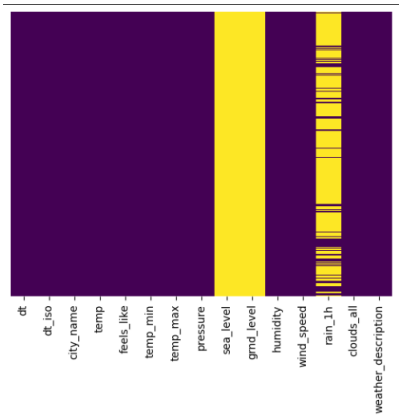


Figure 27: Missing values do dataset de meteorologia de 2022

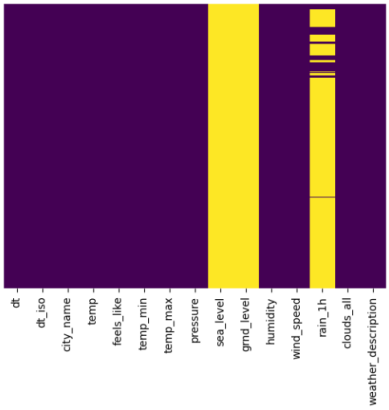


Figure 28: Missing values do dataset de meteorologia de 2023

Com base nestas figuras conseguimos ver que existem três colunas com valores em falta, a "sea_level", "grnd_level" e "rain_1h". Tendo em conta a representação dos dados nesta coluna e a importância que estas trazem para o nosso objetivo, decidimos remover as colunas "sea_level" e "grnd_level", onde decidimos preencher com zero os valores em falta na coluna "rain_1h".

### 3.1.2 Tratamento de datas

Como se consegue ver acima, os *datasets* têm em cada várias colunas para retratar o tempo daquelas amostras, mais propriamente nos de energia as colunas "Data" e "Hora", sendo estas auto explicativas. Nos *datasets* de meteorologia, no entanto, temos "dt" e "dt_iso". Algo que podemos retirar destas é que "dt_iso" é a data quando foi retirada essa entrada com o dia e hora juntos. Logo o que fizemos foi juntar ambas as colunas "Data" e "Hora" numa só para posteriormente juntar os *datasets* de energia e meteorologia, removendo a coluna da "Hora". Ao analisarmos o que a coluna "dt" significava, decidimos removê-la do *dataset*, sendo que esta apenas representa um número em segundos da data dessa entrada.

Como estas datas eram consideradas objetos, decidimos converter-las todas para "date-time".

### 3.1.3 Lidar com dados categóricos

Ao analisarmos melhor o *dataset* de meteorologia, conseguimos ver que a coluna "city_name" detém apenas um único valor, sendo este "local" como pode ser visto abaixo. Consoante isto decidimos remover esta coluna sendo que não nos traz valor para os nossos modelos.

```
meteo_1_copy['city_name'].nunique()

1
```

Figure 29: Valores únicos de city_name

Agora vamos analisar a coluna "weather_description" também do *dataset* da meteorologia. Olhando para a imagem abaixo conseguimos ver a distribuição destes valores.

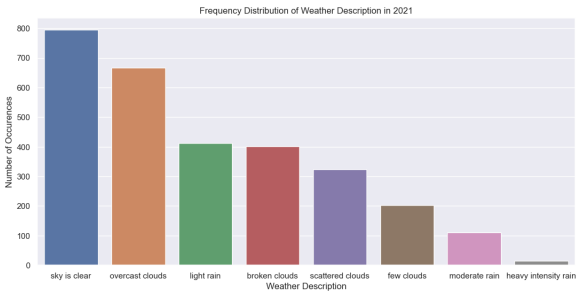


Figure 30: Ocorrências da weather_description de meteorologia de 2021

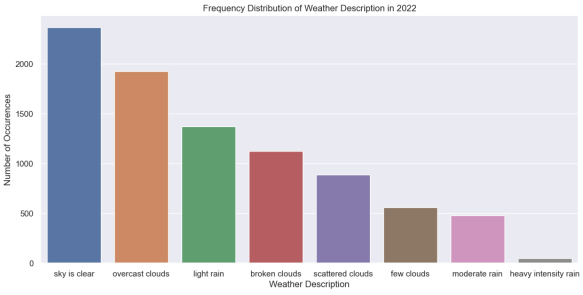


Figure 31: Ocorrências da weather_description de meteorologia de 2022

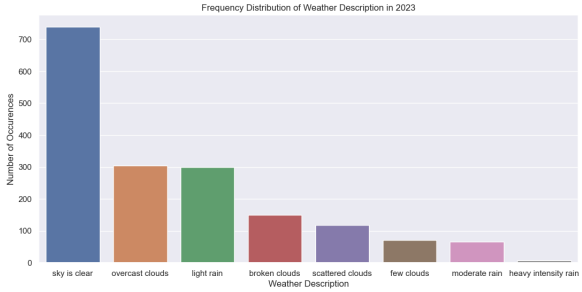


Figure 32: Ocorrências da weather_description de meteorologia de 2023

Consoante estas distribuições decidimos que o melhor rumo a tomar foi fatorizar estes valores, tornando os seus dados num tipo enumerado.

Analisando a coluna "Injecao na rede (kWh)" vimos que podíamos tomar a mesma ação com esta, logo fatorizamos também esta coluna por todos os *datasets*.

### 3.1.4 Merge de *datasets*

Após estes tratamentos iniciamos o *merge* de todos os *datasets* com base na data dos segmentos. Com isto conseguimos ver que ambos os **datasets** de treino encontram-se em bom estado para avançar, porém, o *dataset* de teste após isto demonstra ter certos valores em falta, mais especificamente nas colunas que vieram do *dataset* de meteorologia, como se pode ver abaixo. Isto é óbvio sendo que este contém menos entradas que a sua contraparte de energia. Para resolver este problema nós fomos procurar pelos dados necessários para as datas que faltavam consoante o local que nos foi descrito o problema. Com isto obtivemos dois novos **datasets**, onde os passamos por um tratamento específico para ultimamente o juntar com o **dataset** final de teste.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2256 entries, 0 to 2255
Data columns (total 18 columns):
 #   Column                    Non-Null Count  Dtype         
---  ------                    --------------  -----         
 0   Data                      2256 non-null   datetime64[ns]
 1   Normal (kWh)              2256 non-null   float64       
 2   Horario Economico (kWh)   2256 non-null   float64       
 3   Autoconsumo (kWh)         2256 non-null   float64       
 4   Injecao na rede (kWh)     2256 non-null   int64         
 5   Data_month                2256 non-null   int32         
 6   Data_day                  2256 non-null   int32         
 7   Data_hour                 2256 non-null   int32         
 8   temp                      1752 non-null   float64       
 9   feels_like                1752 non-null   float64       
 10  temp_min                  1752 non-null   float64       
 11  temp_max                  1752 non-null   float64       
 12  pressure                  1752 non-null   float64       
 13  humidity                  1752 non-null   float64       
 14  wind_speed                1752 non-null   float64       
 15  rain_1h                   1752 non-null   float64       
 16  clouds_all                1752 non-null   float64       
 17  weather_description       1752 non-null   float64       
dtypes: datetime64[ns](1), float64(13), int32(3), int64(1)
memory usage: 290.9 KB
```

Figure 33: Entradas em falta no dataset combinado de 2023

Após isto esta é a nossa matriz de correlação entre as diferentes **features** dos datasets.

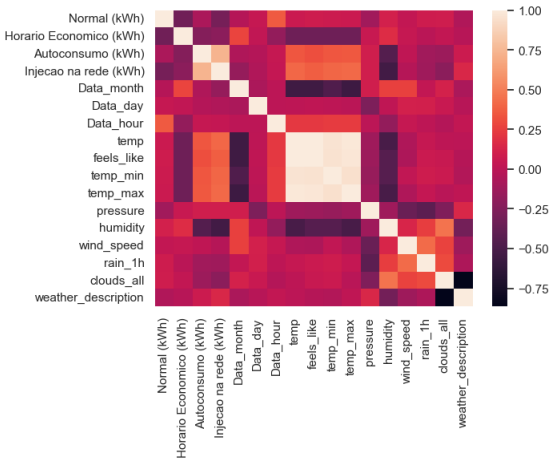


Figure 34: Correlação entre as features do dataset

# 4 Criação dos Modelos

O segundo passo na realização do projeto foi o desenvolvimento de modelos de *Machine Learning* como forma de dar resposta ao problema proposto em cada dataset. Todos os modelos desenvolvidos utilizaram o tratamento de dados realizado na primeira etapa do trabalho.

## 4.1 Dataset de Competição

Para o Dataset de competição usamos 3 modelos sendo estes: *Descision Tree Classifier*,Rede Neuronal LSTM e ARIMA. Cada um destes modelos foi dividido em 3 etapas:preparação de dados,treinamento do modelo e predição.

### 4.1.1 Decision Tree Classifier

- Preparação de dados:o conjunto de dados foi dividido em caraterísticas(x) e a variável alvo (y), onde x contém todas as colunas exceto a variável alvo, e y contém apenas a coluna alvo 'Injecao na rede (kWh)'.Os dados são divididos em conjuntos de treinamento e teste usando train_test_split, alocando 75% para treinamento e 25% para teste.
- Treinamento do Modelo:Um Classificador de Árvore de Decisão do Scikit-learn foi criado e treinado com os dados de treinamento (X_train, y_train) usando o método fit.
- Predição e Avaliação: foram feitas previsões nos dados de teste (X_test) usando o modelo de Árvore de Decisão treinado. A precisão do modelo é calculada usando accuracy_score comparando os valores previstos com os valores reais (y_test).

```
from sklearn.tree import DecisionTreeClassifier
clf = DecisionTreeClassifier(random_state=2023)
clf.fit(X_train, y_train)
```

Figure 35: Modelo Decision Tree Classifier

### 4.1.2 Ensemble (Random Forest e Gradient Boosting)

Random Forest: É um método de ensemble baseado em árvores de decisão. Ele cria várias árvores de decisão e as combina para obter uma previsão mais robusta e geralmente mais precisa.
Gradient Boosting: Assim como o Random Forest, é um método de ensemble. No entanto, ao contrário do Random Forest, o Gradient Boosting constrói árvores sequencialmente, cada uma corrigindo os erros da anterior. Isso geralmente leva a um desempenho superior, mas pode ser mais sensível a overfitting.
Voting Regressor: Combina vários modelos de regressão para melhorar a precisão e a robustez da previsão.

```
# Modelagem com ensemble com hiperparâmetros ajustados
random_forest = RandomForestRegressor(n_estimators=100, max_depth=10, random_state
gradient_boosting = GradientBoostingRegressor(n_estimators=100, learning_rate=0.1

ensemble = VotingRegressor(estimators=[
    ('rf', random_forest),
    ('gb', gradient_boosting)
])

# Treinamento do modelo
ensemble.fit(X_train_scaled, y_train)
```

Figure 36: Modelo Ensemble

### 4.1.3 Random Forest Regressor

Este modelo é particularmente eficaz para problemas de regressão, onde o objetivo é prever valores contínuos. Nesse caso específico, o modelo está sendo aplicado para prever a "Injeção na rede (kWh)" com base em diversas features disponíveis nos dados.

O pipeline de treinamento do modelo inclui etapas essenciais de pré-processamento, como preenchimento de valores ausentes, normalização para features numéricas e codificação one-hot para features categóricas. Essas etapas visam garantir que os dados estejam em um formato adequado para serem processados pela Random Forest.

Após o treinamento do modelo, a avaliação é realizada usando o conjunto de teste, e a métrica $R^2$ (coeficiente de determinação) é utilizada para mensurar o desempenho preditivo do modelo. O valor de $R^2$ indica a proporção da variabilidade na variável alvo que é explicada pelas features do modelo.

# 5 Discussão e Análise crítica dos resultados

A terceira etapa do trabalho consiste na análise dos resultados obtidos com os modelos de *machine learning*. Desta forma, apresentamos um sumário dos resultados obtidos com cada *dataset* e a respetiva análise crítica.

Em relação ao *dataset* livre, houve complicações a determinar qual dos modelos o melhor devido a uma geral insatisfação com os resultados obtidos. De qualquer forma, mantemos que tal se sucedeu pelo elevado número de variáveis assim como um conjunto de dados não muito balanceado.

Em relação ao *dataset* da competição, os nossos resultados revelaram uma discrepância notável entre a nossa posição na competição e os resultados obtidos nos nossos testes internos. Embora tenhamos alcançado uma posição não ótima no cenário competitivo, os dados provenientes das nossas avaliações internas destacaram desempenhos consistentemente positivos. Essa disparidade suscita a necessidade de uma investigação mais aprofundada para compreender as *nuances* que influenciaram o nosso desempenho na competição. Os resultados internos positivos indicam que a nossa abordagem e estratégias têm um potencial promissor, apesar de não terem se manifestado completamente nos ambientes competitivos externos.

# 6 Sugestões e Recomendações

Nesta secção iremos abordar aspetos que consideramos relevantes relativos a sugestões e recomendações.

Consideramos fundamental explorar uma variedade mais ampla de modelos que não foram utilizados inicialmente. A adoção de diferentes algoritmos de machine learning pode proporcionar uma compreensão mais abrangente dos dados em estudo, contribuindo para uma análise mais completa e robusta.

Além disso, sugerimos a investigação e implementação de conjuntos de dados com dimensões ainda mais amplas. O aumento do tamanho do conjunto de dados pode ser uma estratégia eficaz para mitigar o overfitting, especialmente em contextos onde a complexidade do problema pode se beneficiar de uma quantidade maior de exemplos de treino. Esta abordagem tem o potencial de aprimorar significativamente o desempenho do modelo, tornando-o mais capaz de generalizar para dados não vistos.

É importante ressaltar que ao expandir o conjunto de dados, é crucial manter um equilíbrio entre quantidade e qualidade. Dados volumosos devem ser representativos e livres de viés, garantindo que o modelo seja treinado em exemplos diversificados e realistas.

Em suma, a exploração de uma gama mais ampla de modelos e a consideração de conjuntos de dados mais extensos são passos cruciais para aprimorar a solidez e eficácia da análise realizada, proporcionando uma base mais sólida para conclusões e insights derivados do estudo.

# 7 Conclusão

A realização deste trabalho permitiu-nos consolidar a matéria leccionada na cadeira de DAA, especialmente no que diz respeito ao tratamento de dados e desenvolvimento de modelos de Machine Learning.

Consideramos que o presente documento se encontra explicativo, estando de acordo com as etapas estabelecidas na metodologia adotada, CRISP-DM. Além disso, o conjunto de modelos implementados é completo e a exploração realizada consegue responder de forma eficaz e correta aos problemas evidenciados nos datasets.

Em suma, consideramos que o balanço do trabalho é positivo, as dificuldades sentidas foram superadas e os requisitos propostos foram cumpridos.