Caderno de Exercícios elaborado por
Paulo Gouveia

1. Para um primeiro contacto com a linguagem Python e com o editor VS Code, crie um pequeno programa que comece por perguntar ao utilizador o nome e a idade, e lhe diga a seguir se é maior ou menor de idade, tratando-o pelo seu nome.

2. Considere um conjunto S formado por n_1 elementos de um determinado tipo e n_2 elementos doutro tipo. A entropia é uma medida que quantifica de alguma forma a desordem (impureza) de um conjunto de elementos e, para o caso de existirem só 2 classes de elementos, é expressa da seguinte forma:

$$\text{Entropia}(S) = -p_1 \log_2(p_1) - p_2 \log_2(p_2),$$

   em que $p_1=n_1/(n_1+n_2)$ e $p_2=n_2/(n_1+n_2)$ são as proporções do primeiro e do segundo tipos de elementos, respetivamente.
   Escreva em Python a função entropia, que comece por validar devidamente os dados de entrada do problema, invocando-a depois para conjuntos com as seguintes proporções: 0/1, 0.5/0.5, 0.1/0.9 e 0.9/0.1. Não se esqueça de documentar devidamente a função.

3. Escreva uma função em Python, idêntica à do problema anterior, mas que permita calcular a entropia de um conjunto contendo elementos de n classes distintas, a qual se pode expressar matematicamente da seguinte forma:

$$\text{Entropia}(S) = -\sum_{i=1}^{n} p_i \log_2(p_i),$$

   sendo $p_i=n_i/n$ a proporção de elementos da i-ésima classe que estão contidos no grupo.

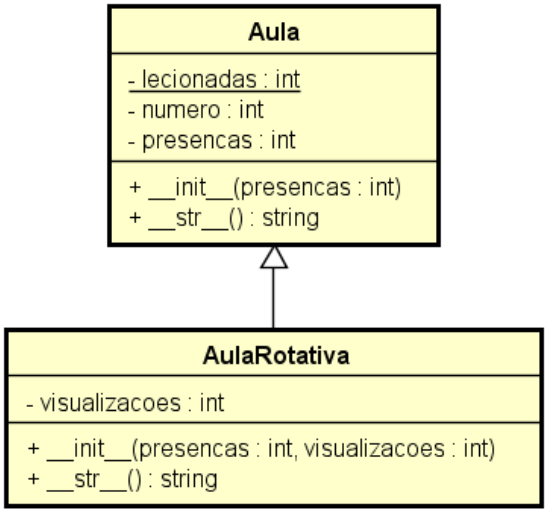
4. Ler um texto e apresentar por ordem alfabética o número de ocorrências de cada palavra contida nesse texto.

5. Defina uma função que verifique se um dado texto é capicua, que seja insensível a maiúsculas e minúsculas e que ignore a pontuação, espaços e outros carateres não alfanuméricos. Por exemplo, o texto “Rise to vote, sir.” é capicua.

6. Defina uma classe que represente uma coleção muito simples de elementos de um qualquer tipo, que suporte, para além das operações elementares de inserção e remoção, a consulta do número de elementos colecionados, através da função len(), bem como a possibilidade de ordenação dos seus elementos, sempre que não esteja em causa um conjunto heterogéneo. Para além de permitir a inserção, *a posteriori*, de elementos individuais, a coleção deve poder ser criada já com alguns elementos. De forma a simplificar a solução, a classe deve armazenar os elementos numa lista do Python. Torne a sua coleção iterável e teste-a devidamente.

7. Guarde na coleção que definiu no exercício anterior instâncias de uma classe criada por si e tente ordená-la. Se essa operação falhar, aperfeiçoe a definição da coleção, tratando devidamente esse erro. Posteriormente, torne as instâncias da sua classe ordenáveis.

8. Considere que se pretende registar o número de presenças dos alunos nas aulas duma unidade curricular lecionada durante o contexto pandémico que enfrentámos em anos anteriores. Nesse período houve a necessidade de implementar um sistema de rotatividade das presenças dos alunos, com as aulas a funcionarem em duas tipologias diferentes: no seu modelo normal, onde todos os alunos assistiam presencialmente, e numa nova tipologia, designada rotativa, em que alguns dos alunos assistiam presencialmente e os restantes assistiam online.



   a) Respeitando a solução que se representa no diagrama de classes, contrua as classes do problema, assegurando a numeração sequencial automática das aulas e que para o caso das aulas rotativas são registados quer o número de presenças físicas, quer a quantidade de visualizações online (número de alunos que assistem remotamente à aula).
   b) Acrescente à(s) classe(s) a possibilidade de reiniciar a numeração das aulas.
   c) Usando listas por compreensão, crie, numa única linha de código, uma coleção com as primeiras aulas, todas presenciais e com um decréscimo de 3 presenças por aula, começando a 1ª delas com 50 presenças e terminando a última com 20.
   d) Volte a criar a coleção anterior, mas agora usando expressões geradoras, em vez de listas por compreensão.
   e) Junte à coleção anterior três aulas rotativas e tire proveito do polimorfismo.

9. Crie o array NumPy apresentado, sem escrever explicitamente os seus valores.

```
[[ 1  4  7 10 13 16 19 22 25 28]
 [ 2  5  8 11 14 17 20 23 26 29]
 [ 3  6  9 12 15 18 21 24 27 30]]
```

10. Suponha que se pretende avaliar o índice de massa corporal (IMC) de um grupo de pessoas, a partir dos seus registos de altura e peso. Sabendo que o IMC de cada pessoa é dado pelo quociente entre o seu peso e o quadrado da sua altura, implemente todas as tarefas que se seguem, usando as funcionalidades do *package* NumPy.

   a) Crie 2 arrays. O primeiro unidimensional, contendo os nomes de um grupo de 10 pessoas, que escolherá arbitrariamente, e o outro bidimensional 2x10, com os pesos e alturas dessas pessoas, guardados nas respetivas linhas. Para não ter que introduzir, um a um, todos esses valores, gere-os aleatoriamente respeitando as seguintes restrições: todos valores inteiros; os pesos deverão situar-se entre 50 e 100 Kg; e as alturas entre 150 e 200 cm.
   b) Calcule, e guarde num novo array, o índice de massa corporal de cada uma das pessoas.
   c) Mostre como se classificam os IMC das pessoas de acordo com as seguintes categorias: “Baixo” se IMC<18.5; “Elevado” se IMC>25; e “Ideal” se 18.5≤IMC≤25.
   d) Mostre todos os atributos das pessoas por ordem crescente do IMC.
   e) Mostre o desvio padrão dos pesos e das alturas.
   f) Mostre as pessoas com peso superior à média.

11. Crie um array NumPy 3x10, com valores aleatórios reais, uniformemente distribuídos entre 10 e 20 (este, exclusivo). Depois, identifique automaticamente qual o valor mais próximo de 15 em cada uma das linhas.

12. Socorrendo-se do *package* Pandas e, em particular, da sua estrutura DataFrame, recorde os números fatídicos da COVID-19 no primeiro ano em que assolou o mundo. Comece por importar para um DataFrame do Pandas o *dataset* com o relatório diário, a 8/nov/2020[1], da COVID-19 no mundo, disponibilizado pela Johns Hopkins University[2] através do seguinte link: https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_daily_reports/11-08-2020.csv

    Analise de seguida esse dataset, realizando as operações descritas nas alíneas que se seguem.

    a) Comece com uma inspeção rápida do *dataset*, consultando as colunas que o compõem (quantidade, nomes e tipos de valores) e o número de registos.
    b) Tente descobrir um método da própria DataFrame que lhe dê logo toda a informação anterior. Conclua depois a sua breve inspeção do *dataset* com a visualização das suas primeiras e últimas 5 linhas.
    c) Crie um novo *dataset* com o número de casos confirmados, recuperados, ativos e de fatalidades por país, e ordene-o por ordem decrescente dos casos confirmados.
    d) A partir da tabela anterior, verifique em que posição se encontra Portugal no *ranking* dos países com mais casos confirmados, e situe o nosso país no panorama mundial relativamente às medidas de localização e de dispersão. Por fim, visualize a totalidade das linhas da tabela com um único *print*.
    e) Mostre, por ordem decrescente, os 10 países com mais fatalidades.
    f) Determine as taxas de letalidade por COVID-19 portuguesa e mundial.
    g) Contabilize o acréscimo que os principais indicadores sofreram nas 24 horas anteriores, por país: número de casos confirmados, recuperados, ativos e fatalidades. Para o efeito, comece por carregar o dataset do dia anterior e por verificar se há uma total concordância entre as linhas dos dois *datasets*.

13. Com o auxílio dos *packages* Matplotlib e Seaborn, continue a analisar os *datasets* do exercício anterior, mas agora numa perspetiva mais visual, criando os gráficos solicitados nas alíneas que se seguem.

    a) Comece por transferir os DataFrames do exercício anterior para um novo *notebook*.
    b) Visualize graficamente o grau de correlação existente entre as variáveis numéricas do relatório diário da COVID-19 que importou do exercício anterior.
    c) Crie um gráfico de dispersão que relacione os casos confirmados com o número de fatalidades, usando quer o *package* Matplotlib quer o Seaborn. Diga, depois, se a partir do gráfico, será possível tirar alguma conclusão quanto à variabilidade da taxa de letalidade entre países?
    d) Sobreponha num gráfico de barras os casos ativos, recuperados e fatalidades dos 10 países com mais casos confirmados, e tire as suas conclusões.

[1] Foi a 11 de março de 2020 que a OMS caraterizou a COVID-19 como uma pandemia. Mas já a 30 de janeiro de 2020, a OMS tinha declarado que o surto do novo coronavírus constituía uma Emergência de Saúde Pública de Importância Internacional.
[2] COVID-19 Data Repository by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University (url: https://github.com/CSSEGISandData/COVID-19).

14. Com o *dataset* Titanic, disponibilizado no *package* Seaborn, produza os seguintes gráficos.
    a) Um gráfico de barras que mostre a idade média dos homens, mulheres e crianças que embarcaram no Titanic.
    b) Um gráfico que mostre a dispersão das idades em cada grupo de pessoas: homens, mulheres e crianças.
    c) Um gráfico que mostre a taxa de sobrevivência dos homens, mulheres e crianças.
    d) Um gráfico que mostre a taxa de sobrevivência dos homens, mulheres e crianças, por classe de cabine.
    e) Um gráfico que mostre o valor médio da tarifa paga pelos homens, mulheres e crianças, por classe de cabine.

15. Tendo por base o *dataset* Iris, também disponibilizado no *package* Seaborn, mostre graficamente que o comprimento e a largura das pétalas duma planta iris podem ser usados para determinar, com elevado grau de acerto, o tipo de planta em causa: setosa, versicolor ou virginica. Verifique depois se não existe outro par de atributos que conjuntamente permitam ainda uma melhor separação dos três tipos de plantas.

16. Como deve saber, num problema de regressão linear simples tenta-se estabelecer uma relação linear entre uma variável independente (variável explicativa) e uma variável dependente (a variável alvo). Mas neste exercício vai avançar mais um degrau na sua aprendizagem, criando um modelo de regressão linear múltipla, que relaciona uma variável dependente com várias independentes.
    a) Do IPB.virtual, descarregue para uma pasta local os ficheiros `housing_dataset.csv` e `housing_descricao.pdf`, que contêm um *dataset* clássico e a sua descrição, respetivamente. Trata-se de um *dataset* bastante popular entre a comunidade da Machine Learning, com os preços e outros dados da habitação em Boston. Comece então por importar para o seu ambiente de trabalho o respetivo *dataset* e analise-o. Mude para 'y' o nome da última coluna, para identificar bem a sua variável alvo.
    b) Ainda antes de tomar qualquer decisão relativamente ao modelo que vai desenvolver, ponha já de lado 20% dos dados do *dataset*, para que no fim possa testar o seu modelo com dados que ele nunca tenha visto.
    c) Este *dataset* adequa-se perfeitamente a problemas de regressão, usando para a variável alvo a mediana do preço da habitação. Nem todas as 12 variáveis restantes poderão ter o poder explicativo necessário para que venham a ter uma contribuição positiva no treino do modelo. Escolha então para variáveis preditoras apenas as duas que apresentem maior correlação com a variável a prever.
    d) Usando o Scikit-learn, crie finalmente o modelo de regressão linear múltipla, treine-o e teste-o com o subconjunto de dados que reservou para esse efeito e usando o R^2 como métrica de desempenho.
    e) Use agora o modelo desenvolvido para prever o preço duma habitação com as caraterísticas do primeiro exemplar do conjunto de teste. Compare-o com o preço real.
    f) Verifique se o desempenho do modelo pode ser melhorado com a inclusão de novas variáveis explicativas.
    g) Grave em disco os subconjuntos de treino e de teste para utilizações futuras.

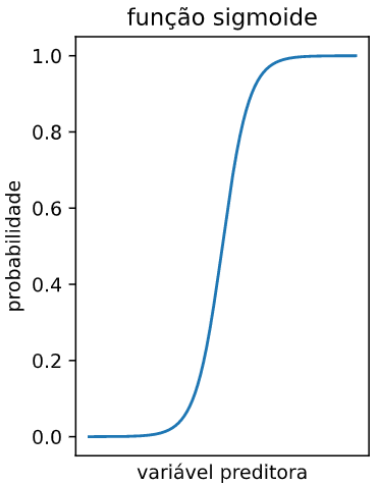
17. Nem sempre a regressão linear é a melhor solução para se capturar a relação entre variáveis. Por vezes essa relação pode ser modelada mais eficazmente, por exemplo, pela linha curva representada por uma função polinomial. Neste exercício vamos tentar aumentar a capacidade de previsão dos preços da habitação em Boston com essa nova forma de modelação: a regressão polinomial.

    a) Usando o Scikit-learn, crie, treine e teste um modelo de regressão polinomial de ordem 2, com duas variáveis independentes (use as duas mais correlacionadas com a variável dependente) – sendo x_1 e x_2 as variáveis independentes, a relação destas com a dependente pode ser representada pela função quadrática $y = a + bx_1 + cx_2 + dx_1x_2 + ex_1^2 + fx_2^2$.
    b) Use o modelo de regressão polinomial desenvolvido para prever o preço duma habitação com as caraterísticas do primeiro exemplar do conjunto de teste. Compare-o com o valor real.
    c) Veja se consegue aumentar ainda mais a capacidade preditiva do modelo, quer usando mais variáveis explicativas, quer aumentado a ordem polinomial do modelo.
    d) Questão final apenas para reflexão. Provavelmente até conseguiu chegar a um modelo que apresenta uma excelente capacidade de previsão. Mas será de esperar que mantenha essa capacidade de acerto quando for usado com outros conjuntos de teste? Na verdade, se pensar bem, a versão final do modelo a que chegou acabou por ser influenciada pelos próprios dados de teste, pois foi com base em sucessivos testes que foi tomando opções no sentido de o aperfeiçoar. Por isso, daria muito jeito dispormos agora de um outro conjunto de teste, completamente novo, que nos permitisse fazer a avaliação derradeira do modelo final. Este é que seria então o verdadeiro conjunto de teste. Ao outro conjunto de dados que foi usado nos sucessivos testes intermédios durante o aperfeiçoamento do modelo é dado o nome de "conjunto de validação". Portanto, sempre que se pretenda aperfeiçoar em várias iterações o modelo em desenvolvimento, deveremos começar por particionar o *dataset* inicial em três subconjuntos: dados de treino, dados de validação e dados de teste.

18. A regressão logística é outro algoritmo de ML supervisionada. Porém, apesar de se designar de "regressão", é um algoritmo de classificação. Em vez de prever o valor de uma variável numérica, estima a probabilidade do valor duma variável categórica assumir uma classe específica. Essa probabilidade é estimada pela função sigmoide,

$$y = \frac{1}{1 + e^{-(\beta_0 + \beta x)}}$$

que converte o valor da variável preditora num valor numérico entre 0 e 1. É durante o treino do modelo que os coeficientes β_0 e β são ajustados aos dados, de forma automática.
Neste exercício vamos criar um modelo simples de regressão logística para um *dataset* de classificação binária.



    a) Com a função load_breast_cancer() do módulo 'datasets' do Scikit-learn, carregue para o seu ambiente de trabalho um *dataset* clássico usado para classificação binária, contendo registos do cancro da mama no estado de Wisconsin, com 30 atributos

preditivos e respetivo diagnóstico (benigno/maligno). Comece por inspecionar esse *dataset* e converta-o para o formato DataFrame.

b) Mostre graficamente que os dois primeiros atributos do *dataset* ajudam a diagnosticar a gravidade da doença. Construa, para o efeito, um gráfico de dispersão com essas duas variáveis preditivas, assinalando os casos malignos a vermelho e os benignos a azul.

c) Reserve para a fase de teste 30% dos dados, optando por uma partição devidamente balanceada.

d) Usando o Scikit-learn, crie o modelo de regressão logística que permita classificar o tipo de doença (benigna/maligna) com base nos 30 preditores disponíveis, treine-o e calcule a sua taxa de acerto.

e) A taxa de acerto (ou acurácia) não é, por si só, suficiente para se avaliar o verdadeiro desempenho dum modelo, nomeadamente em *datasets* com desbalanceamento de classes. Averigue melhor a capacidade do modelo usando métricas que permitam uma avaliação mais consistente (mais fiável) do seu verdadeiro desempenho, como são os casos da matriz de confusão, medida f1, curva ROC e valor AUC.

f) Por fim, use o modelo desenvolvido para prever a gravidade da doença do primeiro exemplo do conjunto de teste, com base nos 30 indicadores. Qual a probabilidade de se concretizar esse diagnóstico?

19. As Random Forests (RF) são um método de aprendizagem baseado em comités (também designados *ensembles* ou especialistas), que tenta criar um classificador forte a partir de múltiplas árvores de decisão. Tal como a generalidade dos métodos de ML, as RF podem ser usadas quer para classificação, quer para regressão. De modo a exemplificarmos a sua utilização de uma forma simples, vamos aplicá-las ao nosso já bem conhecido *dataset* iris. Ainda que esteja vocacionado para a classificação, vamos neste exercício perceber como esse mesmo conjunto de dados pode ser usado para regressão – poderá depois o aluno facilmente desenvolver um modelo análogo das RF para classificação.

    a) Como sabe, o *dataset* iris do Scikit-learn é composto por 4 preditores (largura e comprimento da sépala e da pétala) e uma variável alvo categórica (espécie da planta). Para passar a ter um problema de regressão, assuma que é o comprimento da pétala que se pretende prever e que a sua espécie é uma das variáveis preditoras.

    b) Com essa troca de papeis, entre as variáveis ‘espécie’ e ‘comprimento da pétala’, passou a ter, entre as suas variáveis preditivas, uma de tipo categórico – a ‘espécie’. Codifique devidamente essa variável para que o modelo não seja induzido em erro. Na verdade, encontrando-se as 3 espécies (categorias) representadas por códigos inteiros (0, 1 e 2), sem a devida codificação os valores dessa variável acabariam por ser interpretados, erradamente, pelo modelo como quantidades numéricas.

    c) Comece então por criar os subconjuntos que vão ser usados para treino e para teste do respetivo modelo.

    d) Crie uma primeira versão, não afinada, do modelo RF, que permita prever o comprimento da pétala duma flor iris, treine-o e teste-o calculando o seu coeficiente de determinação (R^2).

    e) Tente otimizar o modelo, usando validação cruzada na afinação de alguns dos seus hiperparâmetros.

f) Visualize graficamente a relação entre as previsões do modelo e os valores reais.

g) Represente num gráfico de barras a importância de cada uma das variáveis explicativas na previsão do comprimento da pétala.

20. Repita o estudo do exercício anterior, mas suportando o seu modelo de previsão numa única árvore de decisão. Compare os resultados obtidos com os anteriores.

21. Volte a repetir o estudo de previsão anterior usando agora o algoritmo KNN (os K vizinhos mais próximos).

22. No exercício 16 criou um modelo simples de regressão linear para prever o preço duma habitação em Boston, tendo conseguido, provavelmente, um coeficiente de determinação R^2 que não superou os 80%. Neste exercício vai tentar aumentar ainda mais o acerto dessa previsão, suportando o seu modelo numa máquina de vetores de suporte (SVM), um algoritmo de ML mais sofisticado.

    a) Com os subconjuntos de treino e de teste que teve o cuidado de guardar no exercício 16, comece por criar rapidamente um modelo de regressão SVM com a sua configuração base, sem se preocupar com questões de afinação nem de normalização de dados. Treine e teste o modelo criado. Surpreendido com o resultado?... Avance para o passo seguinte.
    b) Com o método describe() das DataFrames pode facilmente verificar que o intervalo de variação das variáveis preditivas é bastante díspar, chegando algumas delas a assumir amplitudes várias centenas de vezes superiores às de outras. Uma vez que as SVM são bastante sensíveis à escala das variáveis preditivas, aperfeiçoe o seu modelo normalizando adequadamente os dados. Repare na diferença de desempenho.
    c) Tente melhorar ainda mais o desempenho do seu modelo, afinando-o devidamente.
    d) Preveja o preço duma habitação em Boston com as caraterísticas do primeiro exemplar do conjunto de teste. Compare-o com o preço real. Preveja também o preço do último exemplar.

23. Use agora redes neuronais artificiais para tentar aumentar ainda mais a capacidade de previsão do preço das habitações em Boston, replicando o procedimento seguido no exercício anterior. Compare os resultados.

24. Usando o *dataset* iris de forma mais convencional, desenvolva vários modelos de classificação multi-classe que consigam identificar a espécie da planta com base nas dimensões das suas pétalas e sépalas. Para construção de cada um desses modelos use um algoritmo de ML diferente, entre os que estudou (regressão logística, *random forest*, SVM e redes neuronais artificiais). Compare o desempenho alcançado pelos diferentes métodos.

25. Suponha que uma empresa de confeção de vestuário decidiu comercializar um novo modelo de bermudas, e lhe solicitou a si a difícil tarefa de encontrar o conjunto de tamanhos a produzir que melhor se ajuste ao seu público-alvo. O tamanho desse tipo de peça de vestuário é definido quer pelo perímetro de cintura quer pelo comprimento da parte superior da perna da pessoa padrão. Suponha então que lhe fornecem um *dataset* contendo essas medidas para uma amostra significativa da população em causa. De forma a conseguir identificar, a partir desse *dataset*, os vários tamanhos do modelo a produzir, vai precisar de

subdividir a população dessa amostra em vários subconjuntos, o mais homogéneos possível. Depois, só terá que propor um tamanho de bermudas específico para cada um desses grupos de pessoas.

É precisamente a este tipo de problemas que os algoritmos de *clustering* tentam dar resposta, suportando-se em técnicas de aprendizagem não supervisionada. Pretende-se, com este exercício, que utilize o K-Means, um dos algoritmos de *clustering* mais simples e mais utlizados. Repare que todos os modelos de ML que desenvolveu até agora tinham por base técnicas de aprendizagem supervisionada.

a) Comece por importar de https://data.world/rhoyt/body-measurements o *dataset* BMX_G.csv com a informação necessária. Nesse *dataset* encontram-se registadas 26 medidas padrão de 9338 indivíduos, como a altura, o peso, o IMC e outras medidas mais específicas, onde se incluem as duas de que vai precisar (colunas bmxleg e bmxwaist), e que passaremos a identificar simplesmente por perna (comprimento da parte superior da perna) e cintura (perímetro da cintura). Depois de proceder à seleção e limpeza dos dados, mostre graficamente a dispersão desses dois atributos.

b) Usando o algoritmo K-Means, aplique a técnica de *clustering* aos dados, de forma a encontrar os 3 tamanhos de bermudas que melhor se ajustem ao conjunto de pessoas considerado.

c) Mostre novamente a dispersão das duas medidas, perna e cintura, representado também agora a respetiva associação aos *clusters* e os centróides dos mesmos.

d) Avalie a qualidade do *clustering* que obteve, calculando o seu Coeficiente de Silhueta. Trata-se de uma medida de desempenho que mede quer a coesão dentro de cada *cluster* quer a separação entre eles. Varia entre -1 e 1. Valores próximos de 0 indiciam sobreposição de *clusters*, e próximos de -1 instâncias alocadas a *clusters* errados.

e) Tente agora encontrar o melhor conjunto de tamanhos, otimizando o parâmetro k.