TP555 - AI/ML
Lista de Exercícios #7
Árvores de Decisão

1. Considere o conjunto de treinamento dado na tabela abaixo. Ele é composto por 3 atributos de entrada binários (A_1, A_2 e A_3) e uma saída binária, y. Usando o método ID3, encontre manualmente uma árvore de decisão para este conjunto de dados. Apresente os cálculos feitos para se determinar cada um dos nós.

Exemplo	A_1	A_2	A_3	Saída y
x_1	1	0	0	0
x_2	1	0	1	0
x_3	0	1	0	0
x_4	1	1	1	1
x_5	1	1	0	1

2. Considere o conjunto de treinamento dado na tabela abaixo. Ele é composto por 2 atributos de entrada binários (x_1 e x_2) e uma saída binária, y. Usando o método ID3, encontre manualmente uma árvore de decisão para este conjunto de dados. Apresente os cálculos feitos para se determinar cada um dos nós. Qual o valor do *Remainder* para os atributos x_1 e x_2 durante a escolha do primeiro nó? Qual dos dois atributos é escolhido como primeiro nó? Baseado nesses valores de *Remainder*, é possível termos uma outra versão da árvore que também classifique corretamente todos os dados do conjunto de treinamento?

XOR		
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

3. Treine e ajuste uma árvore de decisão para o conjunto de dados das luas (*moons dataset*).

   a. Gere um conjunto de dados das luas usando: `make_moons(n_samples = 10000, noise = 0.4, random_state=42)`.

   b. Divida-o em um conjunto de treinamento e um conjunto de testes usando: `train_test_split (X, y, test_size=0.25, random_state=42)`.

   c. Plote os dados do conjunto de treinamento em relação às classes a que pertencem. Ou seja, defina marcadores diferentes para identificar cada uma das classes na figura. Por exemplo, use círculos para denotar exemplos que pertencem à classe 0 e quadrados para denotar exemplos que pertencem à classe 1.

d. Use o *Grid Search* com validação cruzada (com a ajuda da classe `GridSearchCV`) para encontrar bons valores de hiperparâmetro para um `DecisionTreeClassifier`. **Dica**: tente vários valores para `max_leaf_nodes`.

e. Treine o modelo com o conjunto de treinamento usando os valores do hiperparâmetro e meça o desempenho do modelo no conjunto de teste. Você deve obter aproximadamente 85% a 87% de precisão.

f. Plote as seguintes informações:

- A árvore de decisão encontrada com o valor ótimo do hiperparâmetro.
- A matriz de confusão.
- A fronteira de decisão.
- A curva `ROC`.

4. Neste exercício você irá continuar o exercício anterior e criar uma floresta de árvores de decisão.

a. Continuando o exercício anterior, gere 1000 subconjuntos à partir do conjunto de treinamento, com cada um contendo 100 exemplos selecionados aleatoriamente. **Dica**: use a classe `ShuffleSplit` do `Scikit-Learn` para isso. O `ShuffleSplit` fornece índices para subconjuntos de treinamento e teste, porém, neste exercício você irá apenas utilizar os índices criados para o subconjunto de treinamento, podendo ignorar os índices do subconjunto de testes. O conjunto de testes que será utilizado é o criado no exercício anterior com a função `train_test_split`. A documentação da classe `ShuffleSplit` pode ser acessada através do seguinte link: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.ShuffleSplit.html.

b. Treine uma árvore de decisão em cada um dos 1000 subconjuntos de treinamento, usando os melhores valores de hiperparâmetros encontrados no exercício 3 ou execute o *Grid Search* novamente. Avalie cada uma das 1000 árvores de decisão no conjunto de teste original, ou seja o conjunto criado no exercício 3 (lembre-se, não é o subconjunto de testes gerado pelo `ShuffleSplit`). Como foram treinadas em conjuntos menores, essas árvores de decisão provavelmente terão desempenho pior que a árvore de decisão do exercício 3, atingindo provavelmente cerca de 80% de precisão.

c. Agora vem a mágica das florestas aleatórias. Para o conjunto de teste original, gere predições com as 1000 árvores de decisão e mantenha apenas a predição mais frequente. **Dica**: você pode usar a função `mode()` da biblioteca `SciPy` para isso: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.mode.html. Essa abordagem fornece previsões por maioria de votos à partir do conjunto de teste original.

d. Meça a precisão das predições obtidas com conjunto de teste original. **Dica**: utilize a função `accuracy_score` para medir a precisão. Você deve obter uma precisão um pouco maior que o modelo do exercício 3 (cerca de 0.5% a 1.5% maior). Ao final deste exercício, você terá treinado o que é conhecido como um classificador baseado em florestas aleatórias.