Atividade prática: Árvores de Decisão

Objetivo da atividade:

- Treinar árvores de decisão para uma tarefa de classificação
- Observar o impacto de diferentes hiperparâmetros na árvore de decisão gerada relacionados a controle da complexidade da árvore e estratégias de poda
- Observar o impacto de diferentes amostras de dados na árvore de decisão gerada
- Interpretar as árvores de decisão obtidas

Instruções iniciais:

Neste exercício, o objetivo é aplicar árvores de decisão para uma tarefa de classificação. Assim, sugere-se o uso de pacote, bibliotecas ou ferramentas que forneçam a implementação pronta do algoritmo, assim como de funções para manipulação de dados. Algumas recomendações:

- Weka (com interface gráfica): possui algoritmos como **J48** (implementação do C4.5), **simpleCart**[1] e **REPTree**, para os quais é possível definir o uso ou não de técnicas de poda (pruning), bem como o valor para parâmetros que controlam a complexidade da árvore induzida: profundidade máxima (maxDepth), número/proporção mínima de instâncias em cada nó folha (minNumObj/minNum), e a confiança usada na poda (C, onde valores menores resultam em podas mais drásticas). Perceba que alguns destes parâmetros não estão disponíveis para ambos os algoritmos mencionados.
    - Existe bastante material de apoio ao uso do Weka no Google. Uma indicação de tutorial para iniciantes é: https://youtu.be/m7kpIBGEdkI
- R, com o pacote caret: possui a implementação do algoritmo CART (método 'rpart' ou 'rpart2') e C4.5 (método 'J48'). Para o primeiro método, é possível alterar o valor do pârametro cp (complexity parameter, no caso de rpart) ou maxDepth (profundidade máxima, no caso de rpart2), onde ambos possuem impacto na complexidade da

[1] Este algoritmo precisa ser instalado manualmente em Tools > Package Manager

árvore. Para o 'J48', é possível controlar C (confiança usada na poda) e M (número mínimo de objetos em cada nó folha).

- Python, com o pacote scikit-learn: possui o método DecisionTreeClassifier, que permite configurar hiperparâmetros como 'criterion' (usar índice Gini ou Entropia), max_depth (profundidade máxima da árvore), min_samples_leaf (número mínimo de objetos no nó folha), e ccp_alpha (parâmetro de complexidade usado na poda. Atenção: por padrão, não aplica a poda).

1. Faça download dos dados fornecidos no Moodle. Este conjunto de dados é conhecido como *Pima Indians Diabetes* (um famoso benchmark em ML) e foi obtido do repositório *Kaggle*[2]. O objetivo da tarefa de classificação é prever se um paciente terá ou não diabetes de acordo com um conjunto de 8 atributos fornecidos. O conjunto possui 768 instâncias e a classe alvo está na coluna *Outcome* (1 = tem diabetes, 0 = não tem diabetes).

2. Com o pacote ou ferramenta de sua preferência, faça o treinamento de árvores de decisão a fim de abordar o problema de classificação acima.

a. Para tanto, utilize o método *holdout* estratificado adotando a acurácia (taxa de acerto) como medida de desempenho. Sugere-se dividir os dados em 80% para treinamento e 20% para teste.
b. Durante o treinamento das árvores, faça variações nos hiperparâmetros conforme o "*Guia de experimentos*" abaixo. Forneça no seu relatório os resultados para cada experimento em forma de tabela ou gráficos de desempenho e a estrutura das árvores geradas (existem funções em R ou Python que geram boas visualizações da árvore. No Weka, após treinamento do modelo, é possível clicar com o botão direito sobre o resultado e selecionar 'Visualize Tree'). Além disso, comente brevemente acerca do resultado, ressaltando como a variação do hiperparâmetro impacta na acurácia do modelo e na complexidade da árvore.

Guia de experimentos:

A. Treine e teste árvores de decisão utilizando como critério de seleção de atributos o Gain Ratio/Entropy (C4.5/J48) ou Índice Gini (CART) - especifique no relatório sua escolha. Qual o desempenho do modelo? Qual parece ser o atributo mais relevante para a classificação, de acordo com o modelo gerado?

[2] https://www.kaggle.com/uciml/pima-indians-diabetes-database

B. Repita o treinamento do modelo utilizando o mesmo algoritmo do item anterior (isto é, mesmo critério de seleção de atributos) e mesma divisão de dados em treino e teste, mas agora variando os hiperparâmetros relacionados à complexidade do modelo (como profundidade máxima e/ou número de instâncias no nó folha). Informe no relatório os valores testados para cada hiperparâmetro variado. Demonstre e comente como a variação destes hiperparâmetros impacta na acurácia e complexidade do modelo (uso o modelo obtido no item A como “baseline”). Compare também as regras de classificação extraídas a partir de ambos os modelos, comentando brevemente como estes parâmetros parecem impactar no poder de generalização das regras (isto é, se as regras de classificação extraídas parecem ser mais “genéricas” ou “mais especializadas” para subconjunto de instâncias de treinamento). Inclua no seu relatório exemplos de regras de classificação obtidas a partir dos modelos gerados.

C. Utilize o mesmo algoritmo do "baseline" gerado no item A e agora repita o processo de treinamento com e sem estratégia de poda. Caso não seja possível optar por treinamento sem poda, varie o hiperparâmetro associado ao controle da complexidade na etapa de poda. Compare os modelos obtidos quanto à estrutura da árvore, número de testes, atributos usados, etc., e seus respectivos desempenhos. Informe no relatório os valores testados para cada hiperparâmetro. Intuitivamente, qual modelo você imagina ser melhor para classificar novas instâncias: o que utiliza ou não utiliza estratégia de poda (ou, de forma alternativa, aquele com uma poda mais ou menos drástica)?

D. De acordo com as suas observações para os itens A, B e C, escolha uma configuração para treinamento de árvores de decisão (critério de seleção de nós, controle de complexidade, estratégia de poda). Repita o processo de treino e teste com 10 partições aleatórias distintas dos dados (isto é, repita o processo de *holdout* 10 vezes usando seeds aleatórias diferentes). Compare os desempenhos obtidos e as estruturas das 10 árvores de decisão geradas. Descreva de forma objetiva se observou variações no desempenho e na estrutura das árvores geradas.

Entregáveis:

- Relatório (em **pdf**) devidamente identificado, com a apresentação dos resultados para os itens A-D acima. A apresentação de resultados pode ser feita por meio de gráficos ou tabelas. Exemplos das estruturas das árvores geradas devem ser incluídos, seja em modo texto ou em imagens. O/A aluno/a deve interpretar e comentar os resultados, apontando os principais achados e conclusões em relação a cada experimento.

- [opcional] Código utilizado para execução dos experimentos

O prazo final de entrega deste exercício é dia **06 de janeiro às 23:59h**.