# Atividade prática: Árvores de Decisão

Objetivo da atividade:

- Treinar árvores de decisão para tarefas de classificação
- Observar impacto de diferentes parâmetros na árvore de decisão gerada, como critérios de seleção, parâmetros para controlar complexidade da árvore, e uso de estratégias de poda
- 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. 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 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

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

'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, por padrão, não aplica a poda).

1. Faça download dos dados fornecidos no Moodle. Estes dados foram obtidos do repositório *Penn Machine Learning Benchmarks*[2], o qual realiza alguns pré-processamentos básicos sobre o conjunto de dados original (como por exemplo, imputar valores faltantes). O conjunto de dados fornecido é denominado *1984 United States Congressional Voting*, no qual o objetivo da tarefa de classificação é predizer o partido, democratas ou republicano, ao qual cada membro da Câmara dos Representantes do Congresso dos Estados Unidos é afiliado a partir dos seus votos registrados (dados de 1984). São 16 atributos categóricos e um atributo alvo (*target*), e um total de 435 instâncias.

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 dois critérios distintos de seleção de atributos, Gain Ratio/Entropy (C4.5/J48) e Índice Gini (CART), mantendo os demais hiperparâmetros com valor padrão. As árvores obtidas possuem desempenho e estrutura similar? Os mesmos atributos são utilizados em ambas as árvores? Qual parece ser o atributo mais relevante para a classificação, de acordo com cada modelo gerado?

[2] https://epistasislab.github.io/pmlb/

B. Selecione um dos algoritmos utilizados no experimento anterior e repita o treinamento variando o valor de hiperparâmetros relacionados à complexidade do modelo (como profundidade máxima e/ou número de atributos no nó folha). Informe no relatório os valores testados para cada hiperparâmetro. 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 exemplos de regras de classificação obtidas a partir dos modelos gerados.

C. Selecione um dos algoritmos utilizados no experimento do item A e 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 (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)?

Entregáveis:

- Relatório (em **pdf**) devidamente identificado, com a apresentação dos resultados para os itens A-C acima. A apresentação de resultados pode ser feita por meio de gráficos ou tabelas. A estrutura das árvores geradas deve ser incluída, 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 **16 de fevereiro às 23:59h**.