**PODA**

* Não olhar para as folhas
* Podar um nível, substituir como folha a maioria do pai
* Não se remova os nós via backtracking

Quem podar primeiro?  
Avaliar a acurácia em cima do conjunto de validação  
Podar com a maior e plotar a decisão

Construção

* Dividir o conjunto em X pontos candidatos
* Apenas entregar slides e código comentado

Slides

* Código não deve falar de aspectos de engenharia
* Enfoque no desempenho da árvore
* Poda da árvore e decisões tomadas

Relatório de exibição

* Árvore: podemos usar o código JTree, disponível em: <http://docs.oracle.com/javase/tutorial/uiswing/components/tree.html>
* Gráfico: usa o código de exemplo JCharts, disponível em: <http://jcharts.sourceforge.net/>

Relatório de discretização:

* Número de pontos candidatos
* Técnica utilizada para definir o número de valores do domínio

Atributo com muitos valores, por exemplo *Country*: a solução seria usar o **Split Information** ao invés do ganho de informação – algoritmo C4.5.

# Algoritmo de Discretização

* Ordenar o conjunto pelo atributo desejado
* Identificar fronteiras (pontos onde ocorrem mudanças de classe)
* Para cada fronteira , calcular ganho total
* Escolher o maior ganho
* Definir o atributo como <= e > que o ganho escolhido
* Gravar no conjunto de dados

# Dados da discretização realizada

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributo** | **Num. Candidatos** | **Tempo de processamento** | **Resposta** |
| age | 10128 | 241 segundos | > 27; <= 27 |
| flnwgt | 10422 | 263 segundos | > 209912.0; <= 209912.0 |
| education-num | 9655 | 183 segundos | > 12; <= 12 |
| capital-gain | 9339 | 92 segundos | > 7073.5; <= 7073.5 |
| capital-loss | 10452 | 98 segundos | > 1820.5; <= 1820.5 |
| hours-per-week | 10445 | 175 segundos | > 41.0; <= 41.0 |

# Dificuldades de Implementação

* Entender e programar o algoritmo ID3 corretamente
  + PlayTennis não cobre todos os casos
  + Usamos algoritmo e exemplo do livro Artificial Intelligence – A Modern Approach – RUSSEL & NORVIG, pois está mais claro e exato
* Decidir e implementar a exibição
  + Método que abria a árvore inteira de uma vez travava o Adult
* Problemas de acesso e cópias de listas na linguagem Java
  + Implementada a interface Cloneable nas Instâncias e Atributos
* Dificuldade para definir um modelo para os dados
  + Classe No, Atributo, Conjunto, ect.

# Outros Detalhes

* Adult e demais bases foram alteradas, incluindo um campo cabeçalho com os nomes dos atributos, permitindo uma desenvolver classe Leitor genérica.

# Objetivos do EP

* Realizar a pós-poda e verificar o comportamento
  + Melhora o overfitting?
* Atenção ao gráfico do livro do MITCHELL
* Dividir o conjunto em 3
* Apresentação: 1 sorteado, mas todos devem saber.
* Saída em texto, conjunto de regras.