

Nursery

Danilo Souza Hugo Santos Iago Medeiros
Welton Araújo

¹Universidade Federal do Pará

16 de Julho de 2013

1 **Introdução**

- Descrição da base de dados
- A ferramenta WEKA

2 **Rede Neural**

- O Momentum
- A simulação
- Resultados

3 **Floresta Randômica**

- A simulação
- Resultados

4 **Prisma**

- Simulação

O Problema

- Modelo de decisão hierárquica
- Desenvolvido para classificar requerimentos à escolas infantis
- Muito utilizado nos anos 80 na Slovênia
- Os requerimentos rejeitados precisavam de uma explicação objetiva
- A decisão final dependia de 3 principais fatores
 - Ocupação dos pais e berçário da criança
 - Estruturas familiar e financeira
 - Condições sociais e de saúde da família
- O modelo foi desenvolvido baseado no sistema DEX [M. Bohanec, V. Rajkovic: Expert system for decision making. Sistemica 1(1), pp. 145-157, 1990]

Atributos e seus estados

- *parents* (Ocupação dos pais)
 - usual, pretentious, great_pret
- *has_nurs* (Berçário da criança)
 - proper, less_proper, improper, critical, very_crit
- *form* (Estrutura familia)
 - complete, completed, incomplete, foster
- *children* (Número de crianças)
 - 1, 2, 3, more
- *housing* (Condições de moradia)
 - convenient, less_conv, critical
- *finance* (Condições financeiras)
 - convenient, inconv
- *social* (Condições sociais)
 - non-prob, slightly_prob, problematic
- *health* (Condições de saúde)
 - recommended, priority, not_recom

- Programa que possui uma coleção de algoritmos de aprendizagem de máquina para usar em tarefas de mineração de dados
- Desenvolvido pelo Machine Learning Group da Universidade de Waikato
- O Weka permite que se trabalhe sobre o dataset e traz ferramentas para: pré-processamento, classificação, regressão, clusterização, regras de associação e visualização
- Todos os nossos resultados foram feitos com o apoio do Weka

O *momentum*

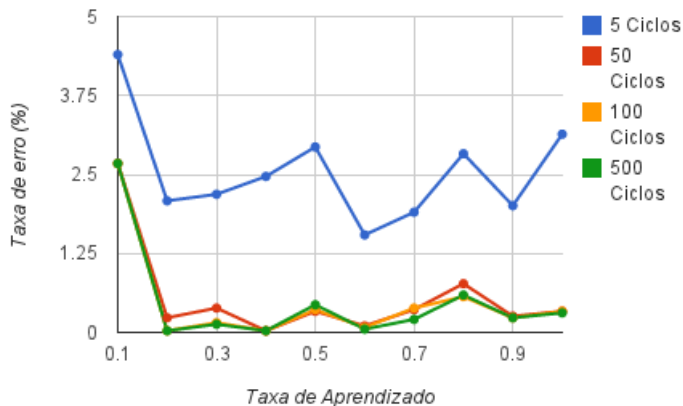
- Utilizado para acelerar a convergência de rede
- Adiciona uma fração proporcional à alteração anterior no cálculo dos pesos sinapticos
- Aumenta a estabilidade do processo de aprendizagem

$$w_{ij}(n+1) = w_{ij}(n) + \alpha e_i(n)x_j(n) + \beta[w_{ij}(n) - w_{ij}(n-1)]$$

- Parâmetros fixos
 - Função de ativação: sigmóide
 - Rede de 1 Camada
- Parâmetros variados para simulação
 - Porcentagem da base de dados para treinamento da rede
 - Taxa de aprendizado
 - Coeficiente de inércia (*momentum*)
 - Número de ciclos (iterações) realizados
- Parâmetros analisados nas simulações
 - Taxa de erro de classificação
 - Tempo de construção do modelo

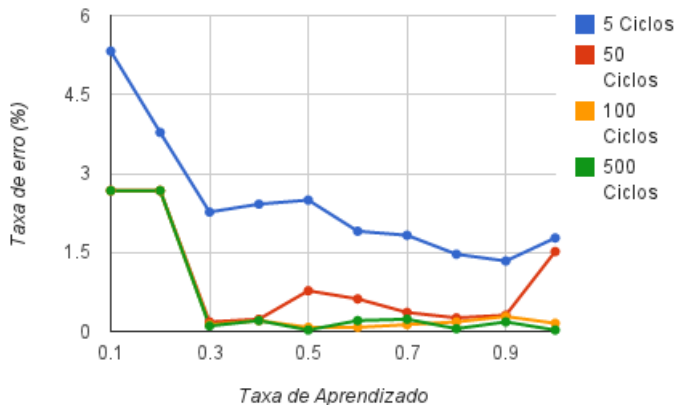
Momento constante

Momento = 0,2 - % Treino = 70



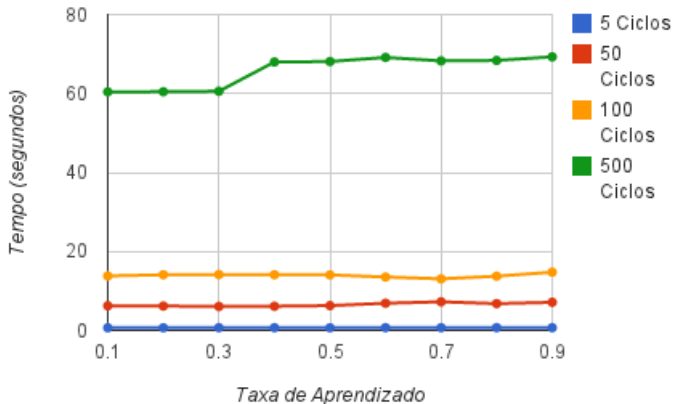
Sem momento

Momento = 0 - % Treino = 70



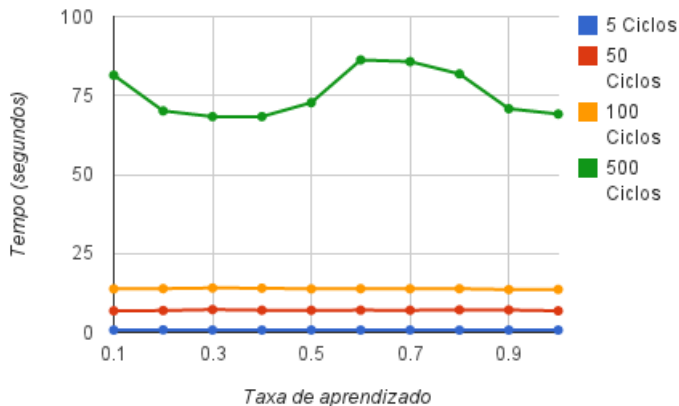
Momento constante

% Treino = 70 - Momento = 0,2

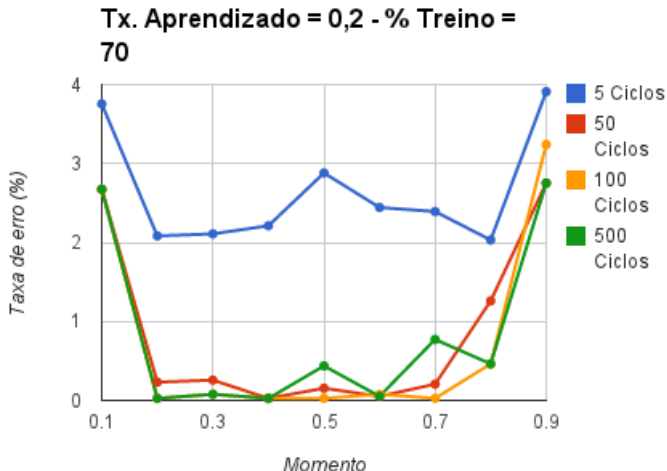


Sem momento

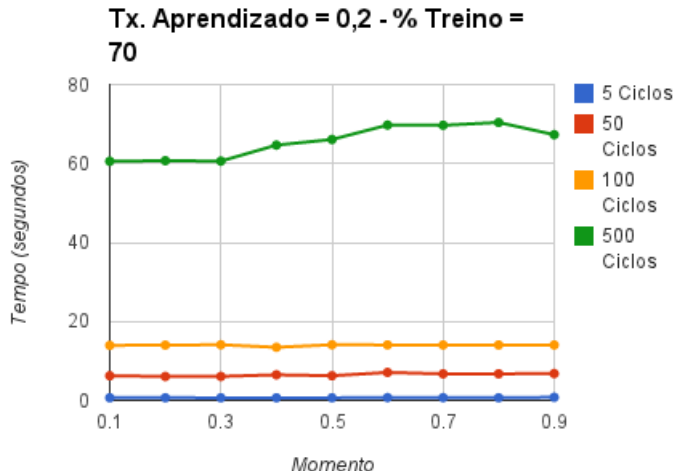
Momento = 0 - % Treino = 70%



Taxa de aprendizado constante

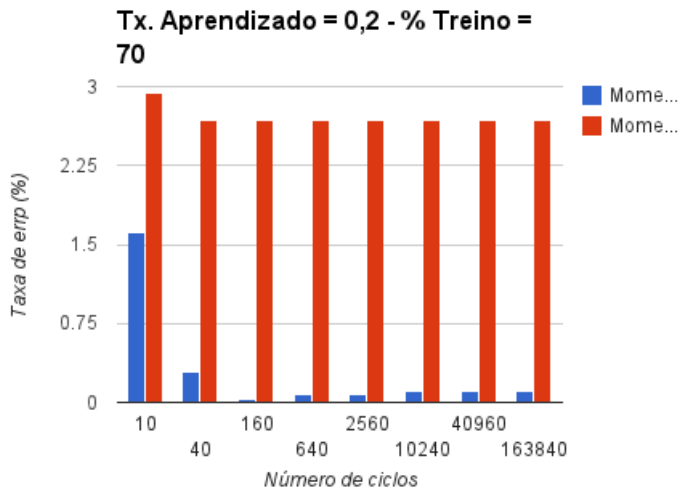


Resultados - Taxa de aprendizado constante



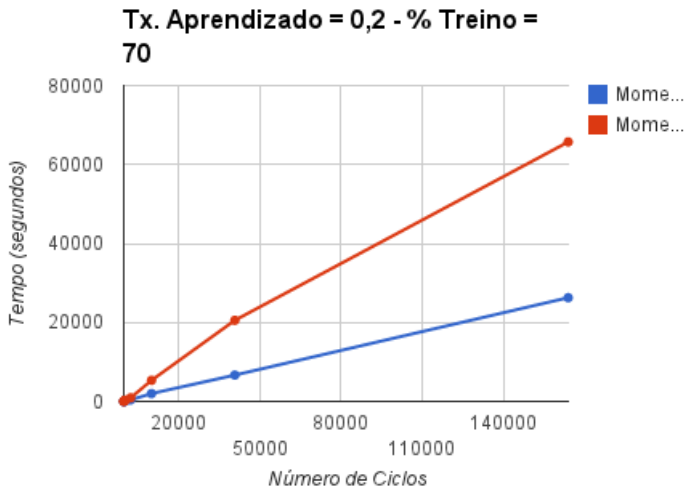
Resultados

Resultados - Taxa de aprendizado constante e momento = 0 & momento = 0.2



Resultados

Resultados - Taxa de aprendizado constante e momento = 0 & momento = 0.2



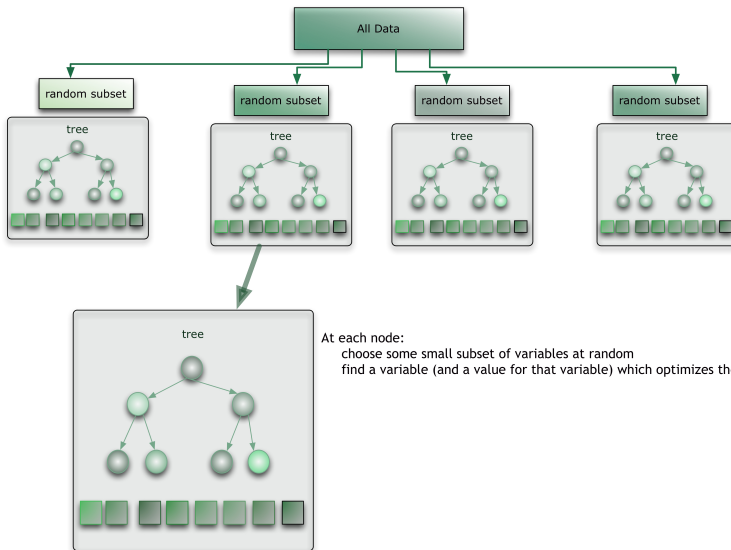
Floresta Randômica (Random Forest)

- Tudo começou em 1996, com o surgimento do meta-algoritmo de Bagging (Bootstrap Aggregating), por Leo Breiman
- Bootstrap ajuda a reduzir variância e overfitting
 - Escolhe aleatoriamente amostras, D_i , de um conjunto de treino, D
 - Esse conjunto de treino D tem reposição (ou seja, uma amostra pode ser escolhida mais de uma vez)
 - Tamanho do conjunto de treino: n
 - Tamanho do conjunto de amostra: n'
 - Geralmente $n' < n$
 - Quando n' for praticamente n , nota-se uma chance de 63% de aparecer amostras não repetidas. Daqui que surgiu o nome de Bootstrap

Floresta Randômica (Random Forest)

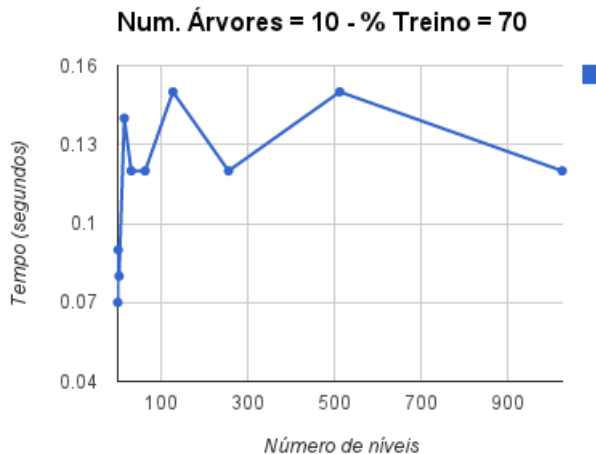
- Em 2001, Breiman lançou o Random Forest
- É um método usado para classificação (e regressão)
- O método faz a construção de várias árvores de decisão, com a diferença de que não usa pruning (poda)
- Algoritmo:
 - Escolhe-se aleatoriamente a amostra (bootstrap) D_i dentre D
 - Constrói a árvore T_i usando amostra D_i
 - Em cada árvore T_i , escolhe aleatoriamente M variáveis e encontra a melhor divisão (split)
- No fim, pode-se
 - pegar o voto majoritário (classificação)
 - calcular a média dos resultados (regressão)

Exemplo de Random Forest



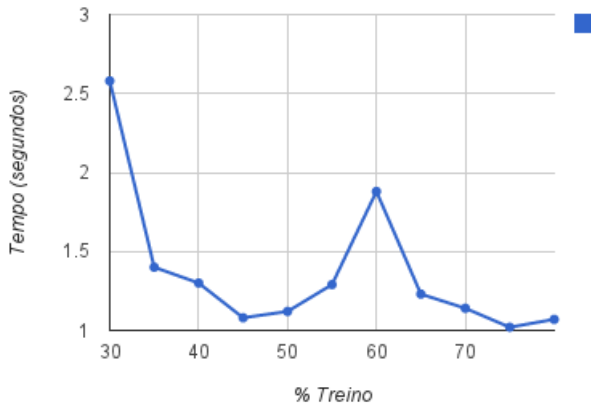
- Parâmetros variados para simulação
 - Número de Árvores
 - Número de Níveis (profundidade)
 - Número de Atributos Utilizados
 - Porcentagem da base de dados para treinamento da rede
- Parâmetros analisados nas simulações
 - Taxa de erro de classificação
 - Tempo de construção do modelo

Tempo x Profundidade



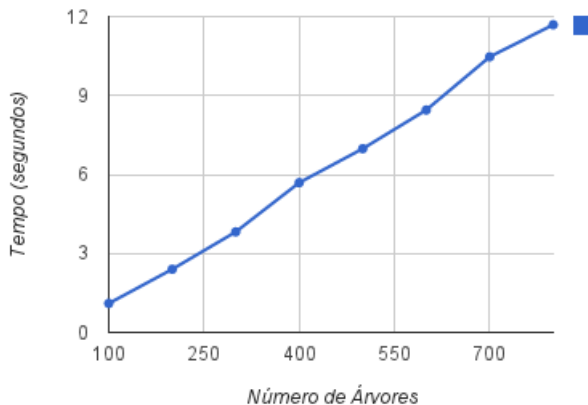
Tempo x Treino (Split)

Num. Árvores = 100 - Num. níveis =
16

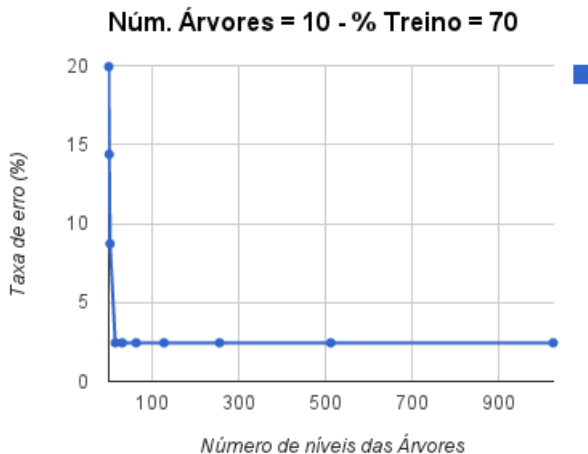


Tempo x Árvores

Num. níveis = 16 - % Treino = 70

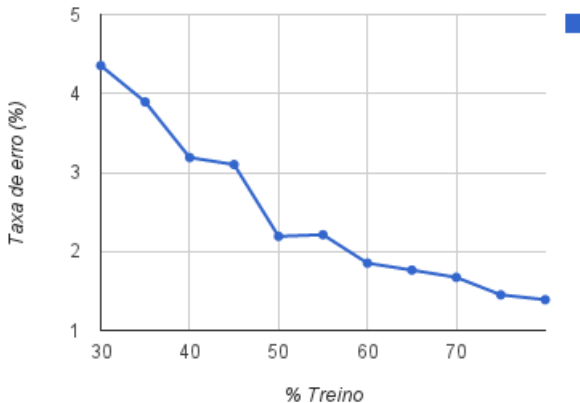


Erro x Profundidade

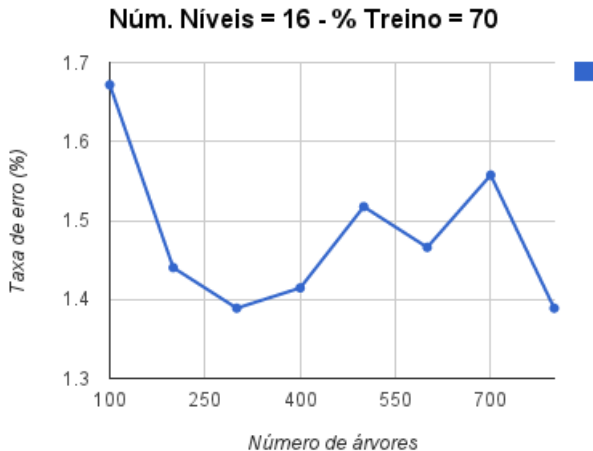


Erro x Treino (Split)

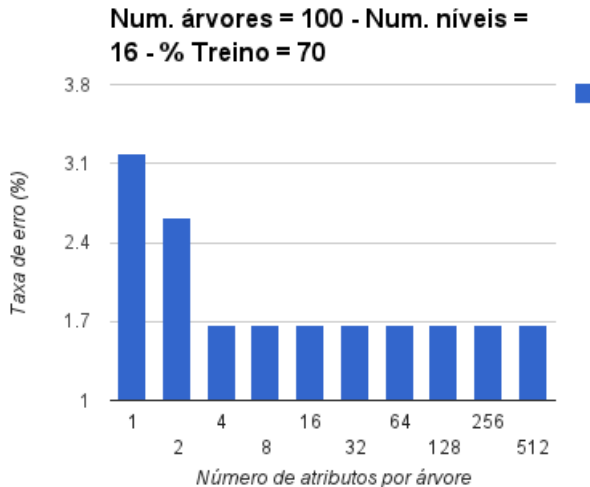
Num. Árvores = 100 - Num. Níveis = 16



Erro x Árvores



Erro x Atributos selecionados



Prisma

- Para cada classe c de 1 a n :
 - **Passo 1** - Calcular a probabilidade de ocorrência da classe c para cada par-atributo-valor
 - **Passo 2** - Selecionar o pa-atributo com a probabilidade máxima de ocorrência e crie um subconjunto de treinamento tomado como entrada compreendendo todas as instâncias que o par selecionado (para todas as classes)
 - **Passo 3** - Repetir os passos 1 e 2 para este subconjunto até o momento em que ele apresente apenas instâncias da classe c . A regra induzida é então a conjunção de todos os pares atributo-valor selecionados na criação deste subconjunto homogêneo.
 - **Passo 4** - Remover todas as instâncias, que satisfaçam a regra formada, do conjunto de treinamento.
- Repetir a sequência de 1 a até que todas as instâncias da classe c tenham sido removidas.

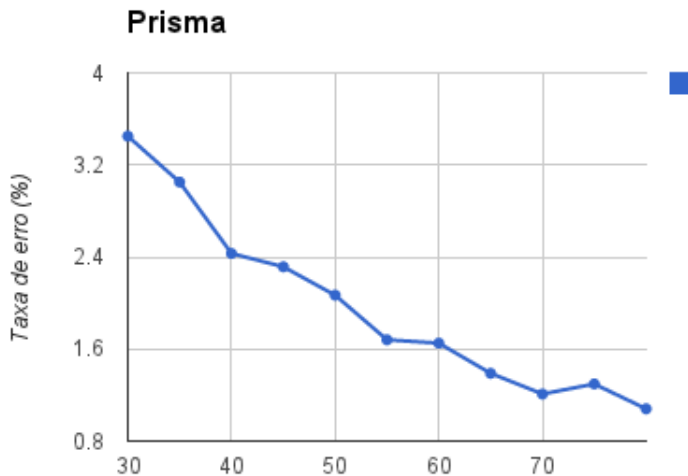
Exemplo - parte I

Atributo-valor	Frequência(P) para classe = recommend	Frequência (T) para atributo-valor	Probabilidade (P/T)
Finance = convenient	2	8	0.25
Finance = inconven	1	8	0.125
Housing = convenient	1	8	0.125
Housing = less_conv	4	12	0.33
Housing = critical	1	12	0.083
Children = 1	0	12	0
Children = 2	3	12	0.25
Children = 3	0	12	0
Children = more	4	12	0.33

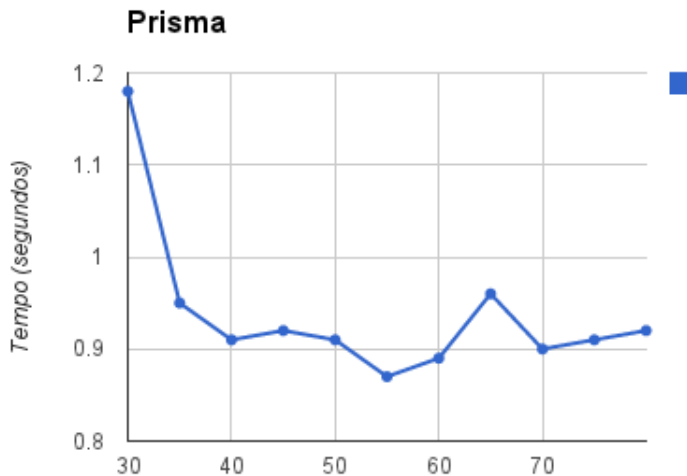
Exemplo - parte II

Atributo-valor AND Finance = convenient	Frequência(P) para classe = recommend	Frequência (T) para atributo-valor	Probabilidade (P/T)
Housing = convenient	4	4	1
Housing = less_conv	1	4	0.25
Children = 1	0	6	0
Children = 2	2	6	0.33
Children = 3	0	6	0
Children = more	3	6	0.5

Exemplo - parte II



Exemplo - parte II



Exemplo - parte II

