Algoritmo de Wang-Mendel para aprendizagem de regras fuzzy

Renato Lopes Moura

16 de dezembro de 2016

Sumário

1 Visão Geral

2 Passo 1: Divisão dos espaços de entradas e saídas

3 Passo 2: Geração de regras fuzzy a partir dos dados

4 Passo 3: Associando um grau a cada regra

5 Passo 4: Criação de uma Base de Regras

6 Passo 5: Defuzzificação

7 Aplicação 1: Backer-Upper Truck

8 Aplicação 2: Série temporal de Mackey-Glass

9 Aplicação 3: Como ficar milionário?

10 Referências

Visão Geral

Esta apresentação descreve o método geral proposto por Li-Xin Wang e Jerry Mendel [1] para geração de uma base de regras fuzzy a partir de dados numéricos.

O método consiste de 5 passos, que serão detalhados a seguir. E após, serão apresentados alguns exemplos de aplicação em problemas diversos.

Para facilitar o entendimento do método vamos considerar um caso simples com **2 variáveis de entrada** e **1 variável de saída**, que poderá ser facilmente estendido para casos mais gerais ao final.

Passo 1: Divisão dos espaços de entradas e saídas

Inicialmente, é necessário dividir os intervalos de valores possíveis das variáveis de entrada e saída em regiões fuzzy.
Supondo que os domínios das variáveis de entrada e saída sejam respectivamente:

$$[x_1^-, x_1^+], [x_2^-, x_2^+], [y^-, y^+]$$

Podemos criar as seguintes regiões fuzzy utilizando funções de pertinência triangulares:

Passo 2: Geração de regras fuzzy a partir dos dados

Em seguida, para cada par **entradas-saída** são determinados os graus de pertinência e é gerada uma regra. Por exemplo considerando o seguinte par:

$$(x_1^{(n)}, x_2^{(n)}; y^{(n)})$$

Podemos gerar a seguinte regra:

$$(x_1^{(n)}, x_2^{(n)}; y^{(n)}) \Rightarrow [(x_1^{(n)}(0.8 \text{ em } B1), x_2^{(n)}(0.7 \text{ em } S1); y^{(n)}(0.9 \text{ em } CE)] \Rightarrow \text{Regra n}$$

SE x_1 é $B1$ e x_2 é $S1$, **ENTÃO** y é CE

Passo 3: Associando um grau a cada regra

Após o passo anterior devemos ter uma grande quantidade de regras (1 para cada par entradas-saída), e inclusive regras conflitantes entre si (antecedentes iguais mas consequentes diferentes).
Para resolver tais conflitos, é associado um "grau de confiabilidade"a cada regra dado por:

$$D(\text{Regra n}) = m_{B1}(x_1^{(n)}).m_{S1}(x_2^{(n)}).m_{CE}(y^{(n)})$$

E então nos casos de conflito são descartadas as regras com menor "grau de confiabilidade".

Nesta etapa também é possível agregar o conhecimento de um especialista ou controlar a influência de dados que possam conter erros de medição. Isto é feito acrescentando-se um novo parâmetro $m^{(n)}$ à equação anterior:

$$D(\text{Regra n}) = m_{B1}(x_1^{(n)}).m_{S1}(x_2^{(n)}).m_{CE}(y^{(n)}).m^{(n)}$$

Passo 4: Criação de uma Base de Regras

No caso simples do nosso exemplo são possíveis 35 regras distintas conforme a tabela a seguir:

		x2						
		S3	S2	S1	CE	B1	B2	B3
x1	S2							
	S1							
	CE							
	B1							
	B2							

Tabela: Base de regras

Passo 5: Defuzzificação

Aplicação 1: Back-Upper Truck

Aplicação 2: Série temporal de Mackey-Glass

Aplicação 3: Como ficar milionário?

Referências

L. Wang and J. M. Mendel, *Generating fuzzy rules by learning from examples* IEEE Trans. Syst., Man, Cybern., vol. 22, no. 6, pp. 1414?1427, (1992).