Pontifícia Universidade Católica
do Rio de Janeiro

Oscar Hernan Samudio Legarda

RandomFIS: um Sistema de Classificação Fuzzy para Problemas de Alta Dimensionalidade

Dissertação de Mestrado

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da PUC-Rio como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica.

Orientadora: Prof. Marley Maria Bernardes Rebuzzi Vellasco
Co-orientador: Prof. Ricardo Tanscheit

Rio de Janeiro

Setembro de 2016

PONTIFÍCIA UNIVERSIDADE CATÓLICA
DO RIO DE JANEIRO

Oscar Hernan Samudio Legarda

RandomFIS: um Sistema de Classificação Fuzzy para Problemas de Alta Dimensionalidade

Dissertação apresentada como requisito parcial para obtenção do título de Mestre pelo Programa de Pós-Graduação em Engenharia Elétrica da PUC-Rio. Aprovada pela Comissão Examinadora abaixo assinada.

Prof. Marley Maria Bernardes Rebuzzi Vellasco
Orientador
Departamento de Engenharia Elétrica PUC-Rio

Prof. Ricardo Tansheit
Co-orientador
Departamento de Engenharia Elétrica PUC-Rio

Prof. Daniel Furtado Leite
Faculdade de Engenharia – UFLA

Prof. Jorge Luís Machado do Amaral
Faculdade de Engenharia – UERJ

Prof. Karla Tereza Figueiredo Leite
Departamento de Engenharia Elétrica - PUC-Rio

Prof. Márcio da Silveira Carvalho
Coordenador(a) Setorial do Centro Técnico Científico - PUC-Rio

Rio de Janeiro, 13 de Setembro de 2016

Oscar Hernan Samudio Legarda

Graduou-se em Engenharia Eletrônica pela Universidade Nariño (Nariño, Colômbia). Durante a graduação trabalhou principalmente em atividades do tipo biomédico e industrial, envolvendo manutenção, calibração e ajuste dos mesmos.

Ficha Catalográfica

Samudio Legarda, Oscar Hernan

RandomFIS: um sistema de classificação fuzzy para problemas de alta dimensionalidade / Oscar Hernan Samudio Legarda; orientadora: Marley Maria Bernardes Rebuzzi Vellasco; co-orientador: Ricardo Tanscheit. – 2016.

72 f. : il. color. ; 30 cm

Dissertação (mestrado) – Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Elétrica, 2016.

Inclui bibliografia

1. Engenharia elétrica – Teses. 2. Sistema de inferência fuzzy. 3. Classificação de padrões. 4. Bootstraping. 5. Random subspace. 6. Bag of Little Bootstrap. I. Vellasco, Marley Maria Bernardes Rebuzzi. II. Tanscheit, Ricardo. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Elétrica. III. Título.

CDD: 621.3

Dedico este trabalho aos meus pais, Jorge e Doris, a minha irmã Leydi, pelo apoio indispensável e incondicional

Agradecimentos

Aos meus orientadores Professores Marley Vellasco e Ricardo Tanscheit pelo apoio, simpatia de sempre, e incentivo para a realização deste trabalho.

Ao CNPq, CAPES e à PUC–Rio, pelos auxílios concedidos e estrutura concedida, sem os quais este trabalho não poderia ter sido realizado.

Aos meus pais, Jorge e Doris, pela formação baseada nos valores, disciplina e amor. Além da compreensão e pacense nos momentos difíceis. À minha irmã Leidy por ser uma boa ouvinte, pelos ânimos e apoio.

Ao pessoal do DEE-PUC-Rio pela ajuda de todos os dias.

A meus amigos Adriano Koshiyama e Jorge Paredes pelas recomendações, motivações e longas discussões acerca da presente dissertação.

Finalmente a meus colegas do LIRA, aos amigos da Colômbia e outros países

Resumo

Samudio Legarda, Oscar Hernan. Vellasco, Marley Maria Bernardes Rebuzzi (Orientadora); Tanscheit, Ricardo (Co-orientador). **RandomFIS: um Sistema de Classificação Fuzzy para Problemas de Alta Dimensionalidade.** Rio de Janeiro, 2016. 72p. Dissertação de Mestrado - Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

Hoje em dia, grande parte do conhecimento acumulado está armazenada em forma de dados. Dentre as ferramentas capazes de atuar como modelos representativos de sistemas reais, os Sistemas de Inferência Fuzzy têm se destacado pela capacidade de fornecer modelos precisos e, ao mesmo tempo, interpretáveis. A interpretabilidade é obtida a partir de regras linguísticas, que podem ser extraídas de bases de dados bases históricas e que permitem ao usuário compreender a relação entre as variáveis do problema. Entretanto, tais sistemas sofrem com a maldição da dimensionalidade ao lidar com problemas complexos, isto é, com um grande número de variáveis de entrada ou padrões, gerando problemas de escalabilidade. Esta dissertação apresenta um novo algoritmo de geração automática de regras, denominado RandomFIS, especificamente para problemas de classificação, capaz de lidar com grandes bases de dados tanto em termos de número de variáveis de entrada (atributos) quanto em termos de padrões (instâncias). O modelo RandomFIS utiliza os conceitos de seleção de variáveis (Random Subspace) e Bag of Little Bootstrap (BLB), que é uma versão escalável do Bootstrapping, criando uma estrutura de comitê de classificadores. O RandomFIS é avaliado em várias bases *benchmark*, demostrando ser um modelo robusto que mantém a interpretabilidade e apresenta boa acurácia mesmo em problemas envolvendo grandes bases de dados.

Palavras-chave

Sistema de Inferência Fuzzy; Classificação de Padrões; Bootstraping, RandomSubspace; Bag of Little Bootstrap.

Abstract

Samudio Legarda, Oscar Hernan. Vellasco, Marley Maria Bernardes Rebuzzi (Advisor); Tanscheit, Ricardo (Co-advisor). **RandomFIS: a Fuzzy Classification System for High Dimensional Problems.** Rio de Janeiro, 2016. 72p. MSc. Dissertation - Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

Nowadays, much of the accumulated knowledge is stored as data. Among the tools capable of acting as representative models of real systems, Fuzzy Inference Systems are recognized by their ability to provide accurate and at the same time interpretable models. Interpretability is obtained from linguistic rules, which can be extracted from historical databases. These rules allow the end user to understand the relationship between variables in a specific problem. However, such systems experience the curse of dimensionality when handling complex problems, i.e. with a large number of input variables or patterns in the dataset, giving origin to scalability issues. This dissertation presents a new algorithm for automatic generation of fuzzy rules, called RandomFIS, specifically for classification problems, which is able to handle large databases both in terms of number of input variables (attributes) and in terms of patterns (instances). The RandomFIS model makes use of feature selection concepts (Random Subspace) and Bag of Little Bootstrap (BLB), which is a scalable version of Bootstrapping, creating a classifier committee structure. RandomFIS is tested in several benchmark datasets and shows to be a robust model that maintains interpretability and good accuracy even in problems involving large databases.

Keywords

Fuzzy Inference System; Pattern Classification; Bootstrapping; Random Subspace; Bag of Little Bootstrap.

Sumário

Lista de figuras

Lista de tabelas

1
Introdução

Hoje em dia, grande parte do conhecimento está armazenada em forma de dados. Muitos algoritmos para tarefas de classificação têm sido desenvolvidos para a extração deste conhecimento [1, 2]. Dentre as ferramentas capazes de atuar como modelos representativos de sistemas reais, os Sistemas de Inferência Fuzzy [3] são consagrados no que diz respeito à capacidade de representar e extrair conhecimento em problemas de classificação, além de sua aplicabilidade em problemas que envolvem incerteza, imprecisão e não linearidade [4].

Abordagens baseadas na lógica fuzzy têm a capacidade de fornecer modelos precisos e, ao mesmo tempo, interpretáveis. A interpretabilidade é caracterizada por regras linguísticas que informam ao usuário a relação explícita entre as variáveis de entrada e as classes existentes no problema [5]. Em várias aplicações, tais como bioinformática [6], aprendizado de máquina [7] e controle [8], é importante compreender como as variáveis de entrada influenciam o comportamento da variável de saída.

Com o objetivo de criar automaticamente sistemas de inferência fuzzy que sejam interpretáveis, sem prescindir de uma boa acurácia, a maior parte dos trabalhos tem trilhado duas linhas principais: Sistemas Fuzzy Evolucionários [9, 10, 11], que fazem uso de algoritmos evolucionários para a elaboração das bases de regras, com ou sem ajustes das funções de pertinência, e Sistemas de Inferência Fuzzy Evolutivos [12, 13], que criam e adaptam a base de regras fuzzy a partir do reagrupamento de novas observações. Os primeiros geralmente demandam um alto custo computacional, devido ao uso de um algoritmo evolucionário. Já os sistemas fuzzy evolutivos, por não assumirem uma alocação inicial e fixa para as funções de pertinência, normalmente apresentam interpretabilidade reduzida. De forma a combinar as vantagens das duas abordagens, em [14] foi proposta uma nova abordagem (conhecida como AutoFIS-Class) para a geração automática de um Sistema de Inferência Fuzzy com foco em classificação, buscando boa acurácia, com reduzido tempo computacional quando comparado aos sistemas

fuzzy-evolucionários, mas privilegiando a interpretabilidade linguística. Os resultados apresentados em [14] demonstram que, para bases de dados modestas, o AutoFIS-Class apresenta uma boa interpretabilidade é competitivo em termos de acurácia,.

Estudos realizados com o AutoFIS-Class evidenciaram o fenômeno conhecido como "maldição da dimensionalidade" (*Curse of dimensionality*) [7] na presença de conjuntos de dados com alta complexidade, ou seja, com um grande número de variáveis (atributos) ou padrões (instâncias). Problemas de escalabilidade em termos de tempo e consumo de memória afetaram de forma significativa o desempenho do AutoFIS-Class nestas circunstâncias.

Algumas técnicas vêm sendo usadas para lidar com este problemas de escalabilidade em Sistemas de Inferência Fuzzy. Estudos realizados em [15, 16, 17] descrevem sistemas de combinação de classificadores fuzzy, demostrando ser esta uma boa solução. Estes sistemas de combinação de classificadores (também conhecidos como ensembles) [18], fazem uso de técnicas como *Bootstrapping* (*Bagging*) e *Random Subspace* [19, 20].

Propõe-se neste trabalho um novo algoritmo de geração automática de regras fuzzy, denominado RandomFIS, capaz de lidar com grandes bases de dados, tanto em termos de número de variáveis de entrada (atributos) quanto em termos de padrões (instâncias). O modelo RandomFIS utiliza os conceitos de *Random Subspace* e *Bag of little Boostrap* (BLB) [21, 22], que é uma versão escalável do *Bootstrapping*, criando uma estrutura de comitê de classificadores. Essa combinação faz do RandomFIS um modelo interpretável e apresentando boa acurácia, graças à alta qualidade do classificador base, conforme será descrito no Capítulo 3.

1.1. Objetivos

1.1.1. Objetivos Primários

O principal objetivo deste trabalho é propor um modelo que permita gerar um sistema de inferência fuzzy com foco em classificação, denominado

RandomFIS, para cenários com elevado número de atributos e padrões, visando, simultaneamente, boas acurácia e interpretabilidade linguística.

1.1.2. Objetivos Secundários

1. Analisar as características do modelo RandomFIS em relação ao sistema AutoFIS e a outros Sistemas de Inferência Fuzzy voltados para alta dimensionalidade.
2. Comparar os resultados obtidos pelo modelo RandomFIS, em problemas de alta dimensionalidade, com seu algoritmo base (o modelo AutoFIS) e com outros Sistemas Fuzzy da literatura.

1.2. Contribuições

A principal contribuição desta dissertação é a proposição e o desenvolvimento do modelo RandomFIS, com as seguintes características:

- Uso de comitês de classificadores que permitam gerar um Sistema de Inferência Fuzzy para lidar com problemas de alta dimensionalidade, tanto em termos do número de atributos quanto em termos do número de amostras na base de dados histórica.

- Aplicação de métodos de combinação de classificadores, como voto ou média, além de uma combinação a partir da fusão de regras, para aproveitar a diversidade gerada pelo sistema, resultando em um sistema interpretável e com boa acurácia.

1.3. Descrição e Organização da Dissertação

Esta dissertação está assim organizada:

- O Capítulo 2 apresenta uma revisão bibliográfica de metodologias para gerar um sistema escalável a partir de comitês de classificadores. São apresentados os métodos mais conhecidos de geração de comitês de classificadores, tais como *Bagging*, *Boosting* e *Random Subspace*, além da nova metodologia conhecida como *Bag of Little Bootstrap*.
- O Capítulo 3 apresenta o modelo proposto, denominado RandomFIS, discorrendo sobre os principais passos na geração do comitê, além das etapas empregadas na geração automática de cada um dos Sistemas de Inferência Fuzzy que compõem o comitê: Fuzzificação, Formulação, Associação, Agregação, Decisão.
- O Capítulo 4 descreve os experimentos realizados e os resultados obtidos. O procedimento adotado consiste nas seguintes etapas: (i) avaliação das diferentes parametrizações do modelo RandomFIS para um conjunto de benchmarks, de modo a avaliar a influência de cada um dos parâmetros; (ii) a partir da melhor configuração geral, efetua-se uma comparação com seu classificador base (AutoFIS) para assim demonstrar como o sistema se torna escalável; (iii) comparação do modelo com outros Sistemas de Inferência Fuzzy voltados para problemas de alta dimensionalidade.
- Por fim, o Capítulo 5 conclui o trabalho e discute os trabalhos futuros, sugerindo alguns novos direcionamentos, tanto para o aprimoramento do modelo, quanto em termos de potenciais aplicações e comparações inexploradas no trabalho.

2
Fundamentos de Comitê de Classificadores

Comitês constituem-se em um paradigma de aprendizado que oferece um número finito de propostas de solução para um dado problema, denominadas componentes do comitê, que têm suas saídas combinadas com a finalidade de alcançar uma solução única. Esta abordagem tem sido amplamente utilizada na última década, tanto para problemas de regressão quanto para classificação de padrões, uma vez que os comitês são comprovadamente capazes de aumentar a capacidade de generalização e, consequentemente, o desempenho geral do sistema [18, 23, 24]. Intuitivamente, a combinação de múltiplos componentes é vantajosa, uma vez que componentes diferentes podem representar aspectos distintos e, ao mesmo tempo, relevantes para a solução de um dado problema. A Figura 1 ilustra a estrutura geral de um comitê. No caso de um comitê de classificadores (objeto deste trabalho), cada componente será um classificador (rede neural, árvore de decisão, classificador baseado em regras, etc...) proposto independentemente e capaz de atuar isoladamente. Para cada conjunto de dados de entrada, os *M* componentes gerarão *M* saídas que serão então combinadas empregando um método de combinação (*f*) para produzir a saída final do comitê.

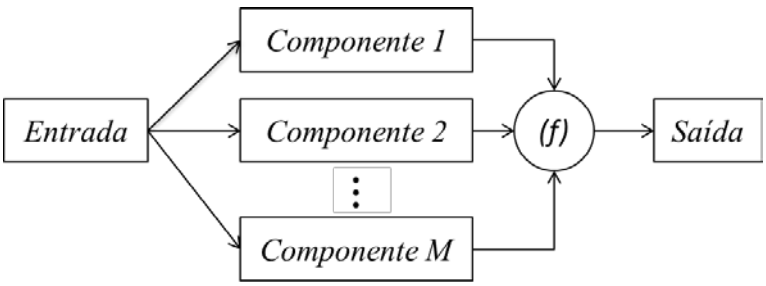

Figura 1. Estrutura geral de um comitê

O aumento na capacidade de generalização e, consequente, a melhora no desempenho total do sistema, apóia-se na *diversidade do erro* apresentado pelos

Componentes [25], ou seja, cada um deles deve apresentar um bom desempenho quando aplicado isoladamente ao problema e, simultaneamente, cometer erros distintos dos demais. Ao se combinarem vários componentes que apresentem erros coincidentes, não haverá ganho de generalização, já que os componentes apresentam o mesmo padrão de erro e de acerto, fazendo com que a combinação traga apenas um aumento no custo computacional.

No restante deste capítulo, será feito um breve resumo sobre a questão da diversidade e serão abordados alguns dos principais métodos de criação de diversidade apresentados na literatura. Para facilitar a exposição de tais métodos, será parcialmente seguida a taxonomia proposta por [26].

2.1. Diversidade

O objetivo da construção de comitês é obter ganho de generalização a partir da combinação das respostas dadas por cada componente. Tal combinação visa à minimização do erro final. Portanto, é necessário que haja diferença na generalização de cada componente, ou seja, diversidade [18], como citado na seção anterior, fazendo com que eles não apresentem erros coincidentes ou correlatos, mas sim uma dissimilaridade do erro entre as demais regiões do problema. Com isso, quando se trata de comitês, a questão da diversidade torna-se se um fator crucial para o seu sucesso [18].

Durante a construção de um comitê, várias técnicas para geração de diversidade podem ser aplicadas. Dentre as técnicas existentes, as mais citadas na literatura são aquelas que fazem uso de estratégias como:

- *Algoritmos de aprendizagem iguais com parâmetros diferentes*: nesta abordagem, a diversidade pode ser alcançada através do uso de diferentes parâmetros de ajuste inicial dos algoritmos de aprendizagem. Mesmo que um comitê seja em princípio homogêneo, i.e. formado por um mesmo tipo de classificador, pode-se obter um comitê diverso como resultado de os parâmetros do algoritmo de aprendizagem terem sindo inicializados com valores diferentes, gerando, assim, modelos diferentes. Em uma rede neural, por

exemplo, isso significaria variar a topologia do modelo de rede neural.

- *Algoritmos de aprendizagem diferentes*: nesta abordagem, a diversidade pode ser obtida pelo uso de diferentes algoritmos de aprendizagem, ou seja, diferentes tipos de classificadores. Estes são os chamados comitês heterogêneos. Por exemplo, um comitê composto por uma rede neural e uma árvore de decisão é mais diversificado do que um comitê composto apenas de redes neurais ou de árvores de decisão.
- *Conjuntos de dados diferentes na construção do classificador*: neste caso, a diversidade se dá pela utilização de métodos de distribuição de atributos ou de estratégias de seleção de padrões de treinamento, tais como Bagging e Boosting, que selecionam conjuntos de exemplos distintos para cada classificador. Dessa forma, os classificadores componentes do comitê terão, em princípio, capacidade de generalização diversa, visto que os estímulos de entrada serão distintos.

Existem propostas de métricas para avaliar quantitativamente a diversidade entre classificadores e auxiliar na escolha de componentes mais diversos para a construção de um comitê. Porém, nenhuma delas é aceita uniformemente, pois ainda não há prova acerca de uma relação formal entre as métricas e o erro total do comitê. Segundo [18], as métricas podem ser divididas em dois grupos:

- *Medidas com paridade*: são calculadas para cada par de classificadores e a diversidade total do comitê é obtida pela média dos pares. Pode-se citar como exemplos a medida de desacordo, que mede a probabilidade de dois classificadores apresentarem discordância, e a medida de dupla falta, que mede a probabilidade de dois classificadores estarem errados em suas decisões;
- *Medidas sem paridade*: consideram todos os classificadores juntos, calculando diretamente um valor para o comitê. Essas métricas se baseiam em entropia ou na correlação de cada classificador com a saída média de todos os classificadores.

As métricas acima não foram utilizadas neste trabalho, pois, apesar de seu caráter intuitivo, sua eficácia não foi comprovada formalmente.

2.2. Abordagens de Geração de Componentes

Na construção de comitês, a etapa de geração dos componentes é de vital importância. Uma vez que esta metodologia é sustentada pela existência de diversidade entre os componentes, é necessário garantir que exista dissimilaridade entre eles na construção para que o comitê seja capaz de proporcionar ganho de acurácia.

Uma das estratégias mais pesquisadas de geração de diversidade consiste no fornecimento de conjuntos de treinamento diferentes para cada um dos componentes. Existem várias maneiras de gerar tais conjuntos de treinamento. Uma delas é fornecer, a cada componente, subconjuntos de amostras diferentes com todos os atributos do conjunto de treinamento original. Outra forma é a apresentação de todas as amostras presentes na base de dados, mas formando subconjuntos com atributos diferentes. Uma terceira maneira é pré-processar os atributos de forma a obter uma representação diferente, como, por exemplo, aplicar uma análise de componentes principais [27]. Às duas primeiras alternativas dá-se o nome de técnicas de *re-amostragem*, enquanto que a terceira é chamada de técnica de *distorção* [28]. Dentre as mais conhecidas na geração de conjuntos de treinamento, as técnicas de *Bagging*, *Boosting* e *Random Subspace* são descritas a seguir.

2.2.1. Bagging

O método *Bagging* (*Bootstrap Agregating*) ou agregação *bootstrap* [19] é um dos representantes da técnica de re-amostragem de dados com reposição. Esta técnica gera conjuntos de treinamento distintos, que são utilizados na obtenção dos componentes de um comitê. O fato de os conjuntos de treinamento serem distintos proporciona a geração de componentes que realizam generalizações de forma também distinta.

A diversidade provida pela técnica de agregação bootstrap [29] se deve à redistribuição aleatória dos dados, ou seja, dado um único conjunto de treinamento *S* com *n* amostras, gera-se, por re-amostragem uniforme com reposição, um subconjunto de dados S^t com *n* amostras. A probabilidade de uma amostra do conjunto *S* ser escolhida para compor o conjunto S^t é igual para todas. Com isso, a diferença entre os conjuntos S^t gerados está na presença de amostras repetidas e, consequentemente, na ausência de algumas amostras que fazem parte do conjunto *S*. Assim, a probabilidade de uma amostra ser escolhida é de $1 - \left(\frac{n-1}{n}\right)^n$, isto é, aproximadamente 63.2 % do conjunto de dados S^t gerado é composto por amostras únicas e o restante, por amostradas duplicadas [30], não havendo praticamente nenhuma chance dos conjuntos de dados gerados serem idênticos. O processo de geração de componentes através do método Bagging é ilustrado na Figura 2, onde c_t indica os classificadores gerados para os S^t conjuntos de dados, Σ é o método de combinação dos classificadores e *P* é a predição final.

O método Bagging emprega os *n* padrões na geração do comitê, o qual se torna ineficiente computacionalmente se a quantidade de dados for muito grande. O *Bag of Little Bootstrap* (BLB) [21] emprega o conceito de Bagging e o combina com técnicas de re-amostragem como *Subsampling* [31] e *m-out-of-n* [32] para alcançar a eficiência computacional. O BLB consiste em criar *s* sub-amostras de tamanho $b = n^\gamma$ com $0.5 < \gamma \leq 0.9$ do conjunto original de dados sem substituição; para cada sub-amostra, são gerados *r bootstraps*. O BLB cria um comitê de múltiplos classificadores com pequenos subconjuntos de Bagging. A titulo de exemplo, se $\gamma = 0{,}6$ e $n = 1.000.000$, cada conjunto de Bagging conteria aproximadamente 632.000 pontos distintos. Considerando, agora, que $b = n^{0{,}6}$, cada *BLB* sub-amostrado apresentaria aproximadamente 3.981 pontos distintos. Em termos de memória, se cada ponto emprega 1 MB de espaço, a base de dados original ocuparia 1TB e um Bagging ocuparia 632 GB, ao passo que cada *BLB* sub-amotrado ocuparia aproximadamente 4GB.

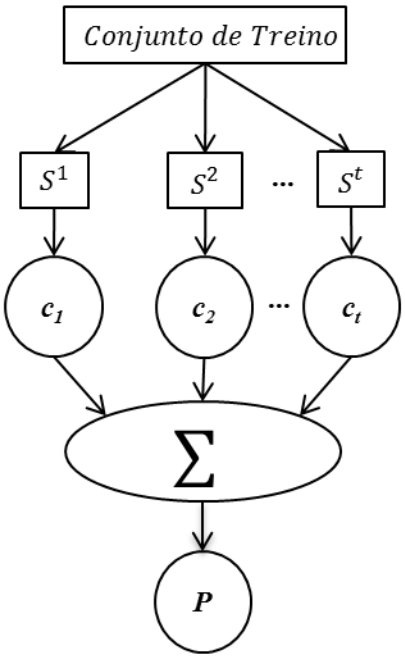

Figura 2. Método *Bagging*

2.2.2. Boosting

No algoritmo de *Boosting* [33, 34], os conjuntos de treinamento não são gerados via amostragem uniforme, como ocorre no algoritmo *Bagging*. A probabilidade de uma dada amostra ser escolhida depende da contribuição desta para o erro dos componentes já treinados. Tomando como exemplo um problema de classificação, uma amostra que não tenha sido corretamente classificada pelos componentes já treinados terá uma maior probabilidade de ser selecionada do que uma amostra corretamente classificada.

Com isso, esta amostra terá uma chance maior de compor o conjunto de treinamento do próximo componente a ser treinado. Um dos problemas desta técnica é a necessidade de treinamento sequencial dos componentes do comitê, uma vez que as probabilidades devem ser redefinidas de acordo com o comportamento dos componentes já treinados. Isso faz com que os últimos componentes gerados na sequência possam lutar com as regiões mais difíceis do espaço de busca, fazendo com que esta técnica seja inadequada na presença de ruído [35].

A versão mais popular do algoritmo de Boosting é chamada AdaBoost.M1 [36], onde cada amostra **x,** de um conjunto de dados de treinamento *S* com *n*

amostras, recebe um peso inicial w_i=1= n. O primeiro classificador é treinado com todas as amostras. Em seguida, é testado utilizando as mesmas amostras. O peso das amostras classificadas erroneamente tem seu valor acrescido, ao passo que as amostras classificadas de forma correta têm seus pesos mantidos. O classificador que acertar as amostras mais difíceis receberá um peso maior. Esse procedimento se repetirá para t componentes (valor informado previamente) ou quando o erro agregado ε_t for igual a zero ou maior do que 0,5.

A resposta final do comitê é resultante de um voto ponderado de todos os componentes. As principais diferenças entre os algoritmos Boosting é o método de ponderação dos dados de treinamento e a combinação dos classificadores [37].

A Figura 3 ilustra o processo de geração de componentes via método *Boosting*, sendo c_i o classificador gerado para a re-amostragem$_i$, p_i sua predição, Σ o método de combinação dos classificadores e P a predição final.

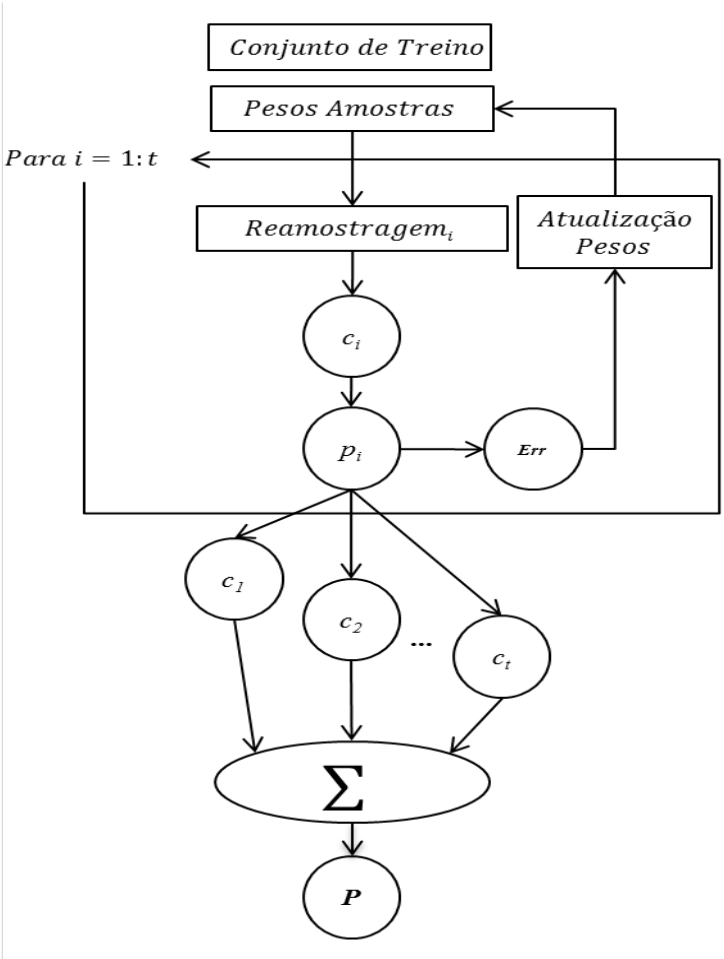

Figura 3. Método *Boosting*

Existem diferenças substanciais entre as estratégias *Bagging* e *Boosting*. A diferença mais imediata é a capacidade de se treinarem múltiplos classificadores em paralelo através de *Bagging*, enquanto ao aplicar *Boosting* o treinamento deve ser sequencial.

Nos experimentos realizados em [30] conclui-se que, na presença de ruído, técnicas de Boosting são inadequadas. Essa característica é previsível, pois durante a aplicação de Boosting os exemplos incorretamente classificados são ressaltados, buscando-se a minimização do erro de classificação durante a etapa de treinamento. Conclui-se ainda em [30] que, apesar de técnicas de *Boosting* serem mais efetivas que *Bagging* (em média) na redução do erro de generalização, o desempenho da estratégia *Boosting* é degradado no caso de conjuntos de dados ruidosos [34]. Por outro lado, o Bagging mostrou-se efetivo em todos os conjuntos de dados estudados em [30].

2.2.3. Random Subspace

Existem ainda outras técnicas para promover diversidade em comitês de classificadores. Conceitualmente, um comitê pode ser construído a partir de uma abordagem conhecida como “dividir e conquistar” no espaço das variáveis de entrada, a qual pode ser sustentada por dois motivos principais: (i) muitos algoritmos de aprendizagem sofrem da maldição da dimensionalidade, ou seja, o tempo computacional do algoritmo aumenta de forma considerável e indesejável com o aumento no número de atributos, dificultando a construção do modelo; (ii) a presença de atributos ruidosos, irrelevantes ou redundantes na base de dados pode confundir o algoritmo de aprendizagem, ajudando a esconder as distribuições de pequenos conjuntos de atributos realmente relevantes, prejudicando, assim, a construção de um classificador com boa acurácia [38]. Um dos métodos mais conhecidos é a seleção aleatória de variáveis (*Random Subspace*) [20], resultando na geração de subamostras do espaço original de variáveis.

A presente dissertação pretende aplicar comitês a diversos problemas de classificação de padrões. Dessa forma, buscando melhorar o desempenho em problemas de classificação em bases de dados com alta dimensionalidade, esta

utilizar-se-á *Bagging* e seleção aleatória de variáveis como métodos de promoção de diversidade em comitês.

2.3. Métodos de Combinação de Comitê

Os diferentes métodos de combinação podem ser distinguidos em função de sua capacidade de treinamento, adaptabilidade e exigência na informação de saída dos componentes individuais. Métodos de combinação como o voto (*'Voting'*) e média (*'Averaging'*) são estáticos, isto é, prescindem de treinamento, enquanto outros são treináveis [39]. Os métodos de combinação treináveis tendem a fornecer um melhor desempenho, a um custo computacional maior e à exigência de dados de treinamento adicionais.

Alguns esquemas de combinação são adaptativos, no sentido de que os métodos avaliam as decisões dos componentes individuais em função do padrão de entrada. Em contraste, abordagens de combinação não adaptativas tratam igualmente todos os padrões de entrada.

Diferentes abordagens de combinação levam em conta o tipo de saída apresentada pelos componentes individuais [40] e são agrupadas em três níveis, conforme apresentado na Figura 4.

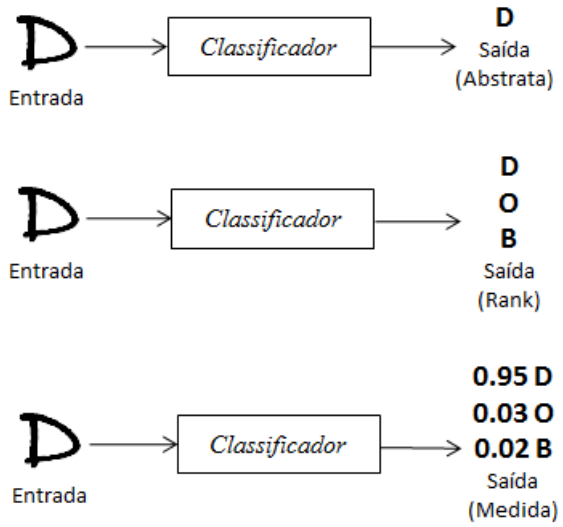

Figura 4. Tipos de saídas dos classificadores

1) *Nível Abstrato*: cada classificador apresenta somente o rótulo da classe.

2) *Nível Rank*: cada classificador atribui um nível de classificação a cada classe. Este nível não pode ser usado separadamente, pois o posto mais alto não significa necessariamente uma alta confiança na classificação.
3) *Nível de medidas* (confiança): cada classificador produz um valor numérico para cada classe indicando a confiança ou probabilidade de que o padrão de entrada pertença àquela classe.

O nível de confiança transmite informação mais rica, enquanton que o nível abstrato contém a menor quantidade de informação sobre a tomada de decisão.

A Tabela 1 fornece uma taxonomia de métodos de combinação e suas respectivas características.

Tabela 1. Métodos de combinação de classificadores [39]

Método	**Arquitetura**	**Treinável**	**Adaptativo**	**Nível de informação**	**Comentários**
Votação	Paralela	Não	Não	Abstrato	Considera estimadores independentes
Soma, Média, Mediana	Paralela	Não	Não	confiança	Robusto, considera avaliadores independentes de confiança
Produto, mínimo, máximo	Paralela	Não	Não	confiança	Considera características independentes
Ensemble generalizado	Paralela	Sim	Não	confiança	Considera correlação de erro
Ponderação adaptativa	Paralela	Sim	Sim	confiança	Examina experiência local
Stacking	Paralela	Sim	Não	Confiança	Boa utilização de dados de treinamento
Borda count	Paralela	Sim	Não	*Rank*	Converte *ranks* em confiança
Regressão logística	Paralela	Sim	Não	*Rank* confiança	Converte *ranks* em confiança
Class set reduction	Paralela Cascata	Sim/Não	Não	*Rank* confiança	Eficiente
Dempster-Shafer	Paralelo	Sim	Não	*Rank* confiança	Fusão de confiança não probabilística
Fuzzy integrals	Paralelo	Sim	Não	Confiança	Fusão de confiança não probabilística
Mixture of local experts (MLE)	Paralelo fechado	Sim	Sim	Confiança	Explora experiência local; otimiz. conjunta

Hierárquico MLE	Paralelo fechado hierárquico	Sim	Sim	Confiança	Idêntico ao MLE; hierárquico
Associative switch	Paralelo	Sim	Sim	Abstrato	Idêntico ao MLE, mas sem otimiz. conjunta
Bagging	Paralelo	Sim	Não	Confiança	Necessita de muitos classificadores de comparação
Boosting	Paralelo hierárquico	Sim	Não	Abstrato	Melhora as fronteiras; Não deve entrar em over-training, sensível à mislabels; necessita de muitos classificadores comparáveis
Random subspace	Paralelo	Sim	Não	Confiança	necessita de muitos classificadores comparáveis
Neural Trees	Hierárquico	Sim	Não	Confiança	Lida com um grande número de classes.

A presente dissertação faz uso de dois métodos de combinação para avaliar o desempenho do RandomFIS: Voto (‘*Voting’*) e Média(‘*Averaging’*), definidos na seção 3.4.

3
Sistema de Inferência Fuzzy para Classificação: RandomFIS

Este capítulo apresenta o Modelo RandomFIS, uma extensão do sistema de inferência fuzzy para classificação AutoFIS [14]. O RandomFIS foi especialmente desenvolvido para lidar com problemas de alta dimensionalidade, isto é, com bases de dados com grande número de atributos ou instâncias.

O RandomFIS baseia-se na criação de múltiplos classificadores, desenvolvidos a partir de bases de dados reduzidas, tanto em termos de atributos quanto em termo de padrões. É composto de cinco módulos principais, apresentados na Figura 5 e detalhados nas próximas seções.

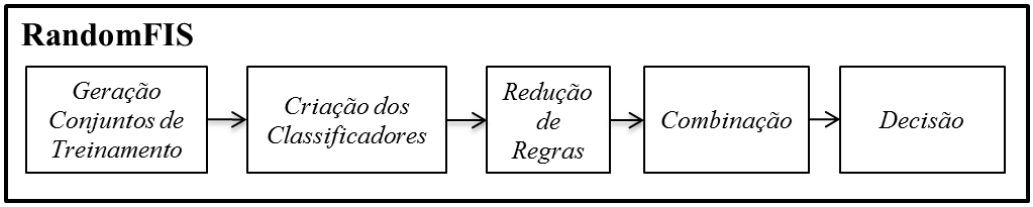

Figura 5. Estrutura RandomFIS

3.1.
Geração Conjuntos de Treinamento

Nesta primeira etapa, o modelo RandomFIS cria diferentes bases de dados, compostas por uma quantidade reduzida de instâncias e de variáveis de entrada, de forma a treinar os múltiplos classificadores, conforme ilustrado na Figura 6. A geração das diferentes bases de dados utiliza os conceitos de *bag of little bootstrap* (BLB) [21] e *Random Subspace* [20], conforme detalhado a seguir.

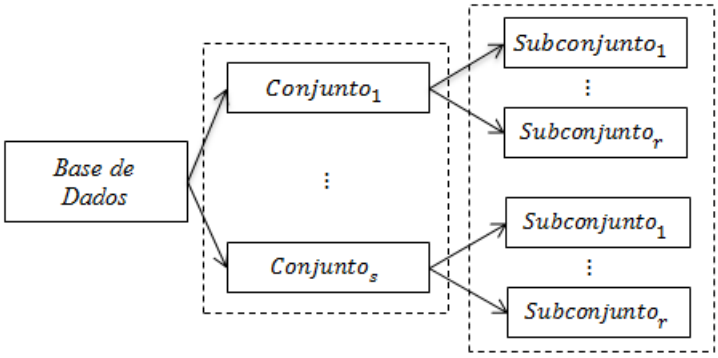

Figura 6. Geração múltiplos conjuntos de treinamento

Como pode ser observado na Figura 6, a geração dos múltiplos conjuntos de treinamento emprega dois níveis de processamento de dados: *Subsampling* e *Resampling* com seleção aleatória de variáveis (*Random Subspace*) do conjunto de atributos. O primeiro nível utiliza a base de dados original de tamanho *n* e gera *s* conjuntos de tamanho *b*, tal que:

$$b = n^{\gamma} \tag{1}$$

onde *n* é o número de instâncias da base de dados original, *b* é o tamanho da base de dados reduzida e $0.5 < \gamma \leq 0.9$, tipicamente.

A técnica de *Subsampling* usada para gerar os *s* conjuntos é baseada em uma seleção aleatória de padrões sobre uma distribuição uniforme, *sem substituição.*

O segundo nível encarrega-se de gerar, para cada $Conjunto_i,\ i \in \{1, \ldots, s\}$, um $Subconjunto_j,\ j \in \{1, \ldots, r\}$ de *Bootstraping* (*Resampling*), agora *com substituição*. Cada um destes *r* subconjuntos contém um número reduzido de variáveis selecionadas aleatoriamente (conhecido como *Random Subspace*), dado por:

$$J^* = \lfloor \log_2(J) + 1 \rfloor \tag{2}$$

onde *J* é o número total de atributos na base de dados original e *J** é o número de variáveis selecionadas em cada $Subconjunto_j$.

3.2. Criação dos Classificadores

Após a geração de todos os conjuntos de dados, a etapa seguinte envolve a criação de *s* grupos de *r* classificadores cada. Cada classificador é treinado com

um $Subconjunto_j$, com base nas etapas de Fuzzificação e Extração de Regras do modelo AutoFIS [14], como ilustrado na Figura 7.

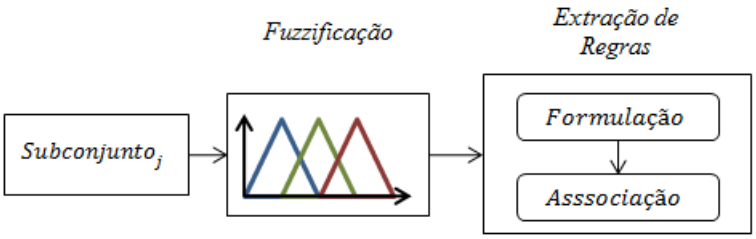

Figura 7. Processo da Geração de Classificadores

3.2.1. Fuzzificação

A etapa de Fuzzificação leva em conta três fatores: a forma das funções de pertinência, o suporte de cada função de pertinência $\mu_{A_{jl}}(x_{ij})$ e o rótulo linguístico apropriado, qualificando o subespaço compreendido pela função de pertinência com um adjetivo correspondente ao contexto. Em teoria, essas tarefas devem ser desempenhadas por um especialista no assunto, proporcionando ganhos de interpretabilidade nas regras fuzzy geradas. Na prática, entretanto, devido à dificuldade de se encontrar um profissional capaz de qualificar o problema (em alguns casos ele não existe), é bastante comum [10, 41, 42] empregar uma disposição dita Fortemente Particionada. Outra abordagem faz uso da informação dos quartis: 0º quartil (valor mínimo), 1º quartil, 2º quartil (mediana), 3º quartil e 4º quartil (valor máximo). Este tipo de particionamento é denominado Tukey e dá origem a disposições do tipo mostrado na Figura 8 e usado pelo modelo RandomFIS.

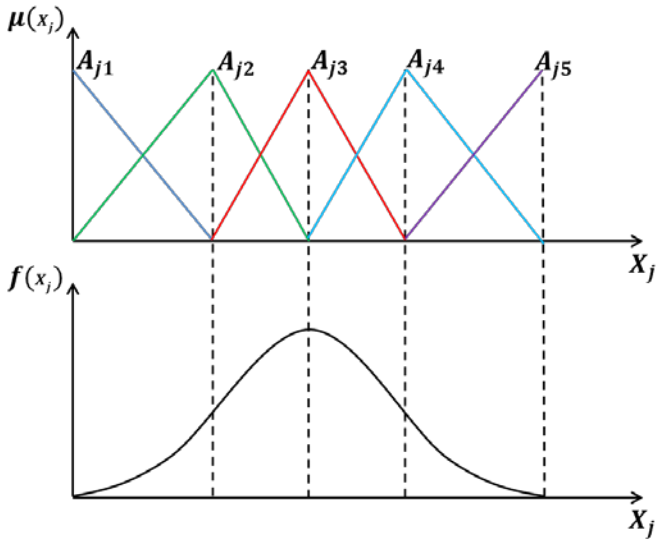

Figura 8. Tipo de funções de pertinência Tukey

Em problemas de classificação, a base de dados consiste em n vetores $\mathbf{x}_i = [x_{i1}, x_{i2}, \dots, x_{iJ}]$, compostos de J atributos X_j de entrada ($i = 1, \dots, n$ e $j = 1, \dots, J$), cada um associado a uma classe C_k das K possíveis, isto é, $C_k \in \{1, 2, \dots\ k, \dots, K\}$, conforme apresentado na Tabela 2,

Tabela 2. Representação de uma base de dados

	Atributos						
Padrão	$\boldsymbol{X_1}$	$\boldsymbol{X_2}$	$\cdots$	$\boldsymbol{X_j}$	$\cdots$	$\boldsymbol{X_J}$	**Saída**
1	x_{11}	x_{12}	$\cdots$	x_{1j}	$\cdots$	x_{1J}	$\vdots$
2	x_{21}	x_{22}	$\cdots$	x_{2j}	$\cdots$	x_{2J}	$\vdots$
3	x_{31}	x_{32}	$\cdots$	x_{3j}	$\cdots$	x_{3J}	$\vdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	C_k
i	x_{i1}	x_{i2}	$\cdots$	x_{ij}	$\cdots$	x_{iJ}	$\vdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	C_k
n	x_{n1}	x_{n2}	$\cdots$	x_{nj}	$\cdots$	x_{nJ}	$\vdots$

A cada j-ésimo atributo são associados L conjuntos fuzzy $A_{jl} = \left\{\left(x_{ij}, \mu_{A_{jl}}(x_{ij})\right) | x_{ij} \in X_j\right\}$, em que $\mu_{A_{jl}}: X_j \rightarrow [0,1]$ é uma função de pertinência

que associa a cada observação x_{ij} um grau de pertinência (compatibilidade) $\mu_{A_{jl}}(x_{ij})$ ao conjunto fuzzy A_{jl} (), conforme ilustrado na Tabela 3.

Tabela 3. Associação de funções de pertinência a um atributo

	Atributos				**Conjuntos Fuzzy associados**					
Padrão	$\cdots$	X_j	$\cdots$	$\cdots$	X_j					$\cdots$
1	$\cdots$	x_{1j}	$\cdots$	$\cdots$	$\mu_{A_{j1}}(x_{1j})$	$\cdots$	$\mu_{A_{jl}}(x_{1j})$	$\cdots$	$\mu_{A_{jL}}(x_{1j})$	$\cdots$
2	$\cdots$	x_{2j}	$\cdots$	$\cdots$	$\mu_{A_{j1}}(x_{2j})$	$\cdots$	$\mu_{A_{jl}}(x_{2j})$	$\cdots$	$\mu_{A_{jL}}(x_{2j})$	$\cdots$
3	$\cdots$	x_{3j}	$\cdots$	$\cdots$	$\mu_{A_{j1}}(x_{3j})$	$\cdots$	$\mu_{A_{jl}}(x_{3j})$	$\cdots$	$\mu_L(x_{3j})$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
i	$\cdots$	x_{ij}	$\cdots$	$\cdots$	$\mu_{A_{j1}}(x_{ij})$	$\cdots$	$\mu_{A_{jl}}(x_{ij})$	$\cdots$	$\mu_{A_{jL}}(x_{ij})$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
n	$\cdots$	x_{nj}	$\cdots$	$\cdots$	$\mu_{A_{j1}}(x_{nj})$	$\cdots$	$\mu_{A_{jl}}(x_{nj})$	$\cdots$	$\mu_{A_{jL}}(x_{nj})$	$\cdots$

Ressalte-se que, nesta abordagem, os atributos categóricos e as classes são binarizadas. A Tabela 4 ilustra um exemplo de transformação de uma base de dados com dois atributos, sendo o Atributo 2 do tipo categórico (quatro categorias diferentes), e com três classes de saída.

Tabela 4. Fuzzificação dos atributos (não categóricos) e binarização dos atributos categóricos e saída.

	Atributo 1		**Atributo 2**				**Saída Binarizada**		
Padrão	$\boldsymbol{A_{11}}$	$\boldsymbol{A_{1L}}$	$\boldsymbol{A_{21}}$	$\boldsymbol{A_{22}}$	$\boldsymbol{A_{23}}$	$\boldsymbol{A_{24}}$	$\boldsymbol{C_1}$	$\boldsymbol{C_2}$	$\boldsymbol{C_3}$
x_1	$\mu_{A_{11}}(x_{11})$	$\mu_{A_{1L}}(x_{11})$	1	0	0	0	1	0	0
x_2	$\mu_{A_{11}}(x_{21})$	$\mu_{A_{1L}}(x_{21})$	0	0	1	0	0	1	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_n	$\mu_{A_{11}}(x_{n1})$	$\mu_{A_{1L}}(x_{n1})$	0	1	0	0	0	0	1

3.2.2. Extração de Regras

A etapa de extração de regras é dividida em duas sub-etapas: Formulação das premissas de regras; e Associação do consequente (classe) mais apropriado para cada premissa. Estas duas etapas são apresentadas a seguir.

3.2.2.1. Formulação

Esta etapa consiste na elaboração das premissas das regras fuzzy. Uma premissa é comumente definida por: "$Se\ X_1$ é A_{1l} *e* ... *e* X_j é A_{jl} *e* ... *e* X_J *e* A_{Jl}", ou em termos matemáticos, como:

$$\mu_{A_d}(\mathbf{x}_i) = \mu_{A_{1l}}(x_{i1}) * \ldots * \mu_{A_{1l}}(x_{ij}) * \ldots * \mu_{A_{Jl}}(x_{iJ}) \quad (3)$$

onde $\mu_{A_d}(\mathbf{x}_i)$ é o grau de pertinência conjunto do *i*-ésimo padrão na *d*-ésima premissa $(d = 1, \ldots, D)$, computado geralmente a partir de uma t-norma * que combina cada $\mu_{A_{1l}}(x_{ij})$. De forma mais genérica, uma premissa pode ser construída a partir de uma combinação das $\mu_{A_{1l}}(x_{ij})$ com o uso de t-normas, t-conormas, operadores de negação e modificadores linguísticos. Neste trabalho, considera-se a t-norma produto. Além disso, o operador de negação pode incidir sobre cada elemento que compõe uma premissa de forma independente, em oposição ao procedimento de negar o resultado como um todo:

"$Se\ X_1$ é *não* A_{1l} *e* ... *e* X_j é A_{jl} *e* ... *e* X_J é *não* A_{Jl}"

Apesar do emprego restrito da t-norma produto e do operador de negação por elemento, o número de possíveis premissas, de diferentes tamanhos, para formar uma base de regras fuzzy cresce de maneira exponencial à medida que mais atributos são adicionados à base de dados. A título de exemplo, considere-se um problema com três atributos (X_1, X_2, X_3), cada um deles associado a dois termos linguísticos (*L*, *H*), como mostrado na Figura 9(a). Supondo a possibilidade de existirem regras com diferentes tamanhos de premissas (neste caso, o tamanho máximo é três, devido à existência de três atributos), a Figura 9(b) ilustra o número total de premissas geradas para cada um dos possíveis tamanhos de premissa (número de elementos no antecedente de cada regra): seis possíveis regras de tamanho 1 (Se X_1 é A_{1L}", $Se\ X_1$ é A_{1H}", ..., e "$Se\ X_3$ é A_{1L}");

doze de tamanho 2 ("$Se\ X_1$ é A_{1L} $e\ X_2$ é A_{2L}",..., e "$Se\ X_3$ é A_{3H} $e\ X_1$ é A_{1H}"); e oito de tamanho 3 (Se X_1 é A_{1L} e X_2 é A_{2L} e X_3 é A_{3L}", ..., $Se\ X_1$ é A_{1H} $e\ X_2$ é A_{2H} $e\ X_3$ é A_{3H}"). Quando se eleva o número de atributos e termos linguísticos, gera-se, consequentemente, uma quantidade maior de possíveis premissas. Contudo, nem todas as premissas são necessárias para formar uma base de regras, pois muitas delas ou são redundantes ou geram conflitos na decisão da classe mais apropriada.

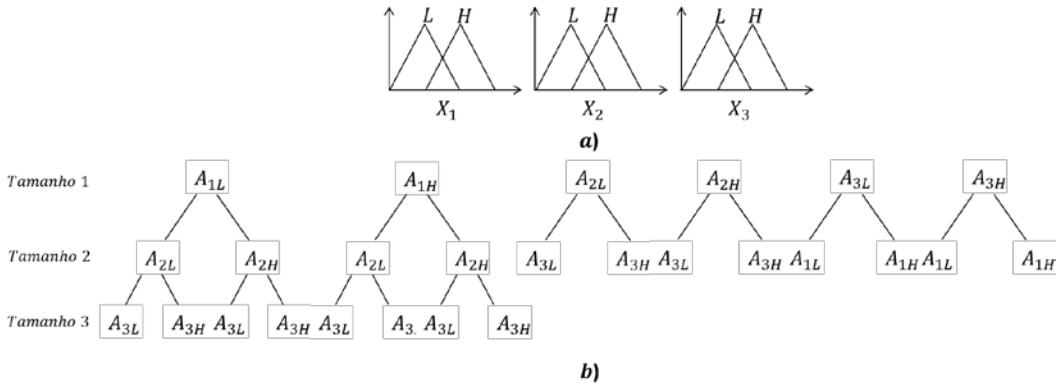

Figura 9. Possíveis premissas a gerar com: a) três Atributos e dois termos linguísticos associados; b) Número total de premissas geradas para cada tamanho.

Do ponto de vista da interpretabilidade, é desejável que haja poucas regras e que estas contenham poucos elementos em seus antecedentes. Assim, propõe-se o seguinte procedimento para a geração de premissas no RandomFIS:

i. Limitação no tamanho máximo de premissas (geralmente inferior ao número de atributos existentes na base de dados);
ii. Avaliação da viabilidade de uma premissa a partir de um conjunto de filtros: suporte, similaridade e grau de conflito na classificação, descritos a seguir;
iii. Geração de premissas de complexidade crescente, em um esquema do menor para o maior número de condições, isto é, criação, inicialmente, de premissas de tamanho 1, em seguida criação de premissas de tamanho 2 a partir das viáveis de tamanho 1, criação de premissas de tamanho 3 a partir das viáveis de tamanho 2, e assim sucessivamente.

O terceiro item acima tem como objetivo principal gerar premissas mais interpretáveis, com poucos antecedentes, assim como não incorrer em uma sobrecarga computacional desnecessária com a geração de premissas inapropriadas (cf. resultado do item ii). Descrevem-se a seguir os diferentes procedimentos de filtragem utilizados no RandomFIS (item ii).

Filtro do Suporte: O filtro do suporte visa à elaboração de premissas que possam cobrir um número elevado de padrões na base de dados. Dada uma premissa $\mu_{A_d}(\mathbf{x}_i)$, o seu Suporte é dado por:

$$Sup_d = \frac{\sum_{i=1}^{n} \mu_{A_d}(\mathbf{x}_i)}{n} \tag{4}$$

onde *n* é o número de instâncias (padrões) na base de dados.

Dada uma tolerância ε_{Sup} definida pelo usuário, uma premissa é viável se $Sup_d > \varepsilon_{Sup}$. A título de exemplo, considere-se uma premissa $\mu_{A_{1L}}(\mathbf{x}_i)$ que não obtenha *Suporte* maior do que ε_{Sup} (Figura 10). Qualquer combinação de $\mu_{A_{1L}}(\mathbf{x}_i)$ com outra função de pertinência irá gerar premissas inviáveis; o trapézio da Figura 11 indica as premissas eliminadas. Isto pode ser verificado por meio da propriedade de degeneração ou não idempotência da maior parte das t-normas [43].

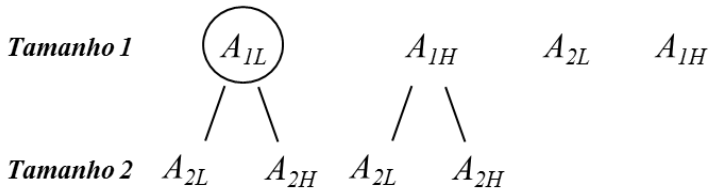

Figura 10. Premissa que não excede o limiar do Suporte

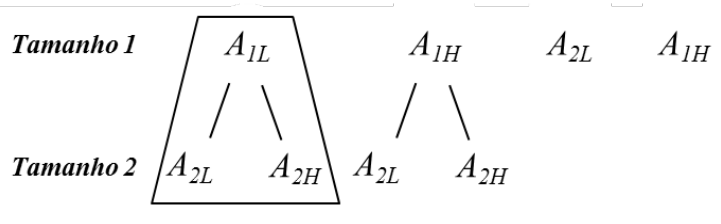

Figura 11. Premissas excluídas do espaço de busca

Filtro da Similaridade: O objetivo deste filtro é reduzir a ocorrência de premissas semelhantes ou iguais. Dadas duas premissas $\mu_{A_d}(\mathbf{x}_i)$ e $\mu_{A_v}(\mathbf{x}_i)$, a similaridade entre elas pode ser medida como:

$$Sim_{d,v} = \frac{\sum_{i=1}^{n} min\left\{\mu_{A_d}(\mathbf{x}_i),\ \mu_{A_v}(\mathbf{x}_i)\right\}}{\sum_{i=1}^{n} max\left\{\mu_{A_d}(\mathbf{x}_i),\ \mu_{A_v}(\mathbf{x}_i)\right\}} \tag{5}$$

Dada uma tolerância ε_{sim} definida pelo usuário, duas premissas são similares se $Sim_{d,v} > \varepsilon_{sim}$. Identificada a similaridade, remove-se a premissa com o menor Suporte Relativo.

Filtro do PCD: este filtro busca reduzir a ocorrência de regras conflitantes. Para tanto, calcula-se o Grau de Confiança Penalizado à classe k [44] – PCD_k. Dada uma premissa $\mu_{A_d}(\mathbf{x}_i)$, o seu PCD_k é dado por:

$$PCD_k = max\left\{\frac{\sum_{i\in k}\mu_{A_d}(\mathbf{x}_i) - \sum_{i\notin k}\mu_{A_d}(\mathbf{x}_i)}{\sum_{i=1}^{n}\mu_{A_d}(\mathbf{x}_i)}, 0\right\} \tag{6}$$

Buscam-se, assim, regras que são mais específicas a uma determinada classe e não as mais generalistas. Somente são viáveis as premissas com $PCD_k > 0$. Após a geração de D premissas $\mu_{A_1}(\mathbf{x}_i)$, $\mu_{A_2}(\mathbf{x}_i)$ …, $\mu_{A_D}(\mathbf{x}_i)$ que atendam aos limites de tamanho e que tenham passado por todos os filtros, o próximo passo é associar cada premissa a uma classe consequente.

3.2.2.2. Associação

Determina-se, nesta etapa, a classe consequente mais compatível com a premissa $\mu_{A_d}(\mathbf{x}_i)$. A d-ésima premissa associada à classe k (isto é, uma regra fuzzy) é denotada por $\mu_{A_d\,(k)}(\mathbf{x}_i)$, que descreve, em termos linguísticos:

"$Se\ X_1$ é $A_{1l}\ e \ldots e\ X_J\ e\ A_{Jl}\ então\ \mathbf{x}_i\ pertence$ à $Classe\ k$"

O RandomFIS emprega o método dos Mínimos Quadrados Restritos (MQR) [11], que parte do princípio de que toda premissa pode ser reescrita como uma combinação convexa das *K* classes que busca explicar. Nesse sentido, para cada premissa *d*, determina-se uma configuração linear de pesos, no intervalo [0, 1], que pode ser descrita como um problema de otimização:

$$\begin{gathered} min \sum_{i=1}^{n}\left(\mu_{C_i\in k}\left(\mathbf{x}_i\right)-\sum_{k=1}^{K}\beta_k\mu_{A_d}\left(\mathbf{x}_i\right)\right)^2 \\ sujeito \sum_{k=1}^{K}\beta_k=1 \text{ e } \beta_k\geq 0 \end{gathered} \tag{7}$$

onde $\mu_{C_i\in k}(\mathbf{x}_i) \in \{0,1\}$ é o grau de pertinência observado do padrão $\mathbf{x}_i$ à classe *k* (representação binária da classe *k*) e β_k é o grau de influência da classe *k* na descrição da premissa $\mu_{A_d}(\mathbf{x}_i)$. Esse é um problema típico de Programação Quadrática, cuja solução usada nesta dissertação é a mesma de [42] com uso dos Gradientes Conjugados. A classe *k* descarta premissas $\mu_{A_d}(\mathbf{x}_i)$ com $\beta_k = 0$. Se $\beta_k = 0$ para todo *k*, a premissa não é endereçada a nenhum termo consequente. Neste tipo de associação, é permitido que uma premissa possa ser endereçada a vários consequentes; isto pode ser de utilidade para algumas bases de múltiplas classes que intrinsecamente compartilhem certas características.

3.3. Redução de Regras

Este módulo tem por objetivo reduzir o número total de regras geradas na etapa anterior. Como mencionado na seção 3.1, cada $Conjunto_i,\ i \in \{1,\dots,s\}$ gera *r* Classificadores Fuzzy, treinados com um diferente $Subconjunto_j,\ j \in \{1,\dots,r\}$ que contém uma base de dados reduzida (em termos de instâncias e variáveis). Este módulo inspeciona todas as regras geradas pelos *r* classificadores, obtidos pelo $Conjunto_i$ e combina-os para obter uma base de regras compacta para cada classe *k*.

O processo de combinação (poda) é simples e direto (ver Figura 12): as bases de regras criadas por cada classificador BR_1, BR_2,..., BR_r são agrupadas de acordo com o seu consequente (Classe) e regras idênticas são eliminadas.

A Figura 12 exemplifica este processo. A partir de um $Conjunto_i$ de dados, *r* bases de regras são geradas independentemente. Regras em azul, associadas à

Classe 1, são idênticas, o mesmo ocorrendo com as regras vermelhas associadas à Classe 2. Somente uma amostra de regra azul e uma amostra de regra vermelha serão consideradas na base de regras final, extraída do $Conjunto_i$ de dados.

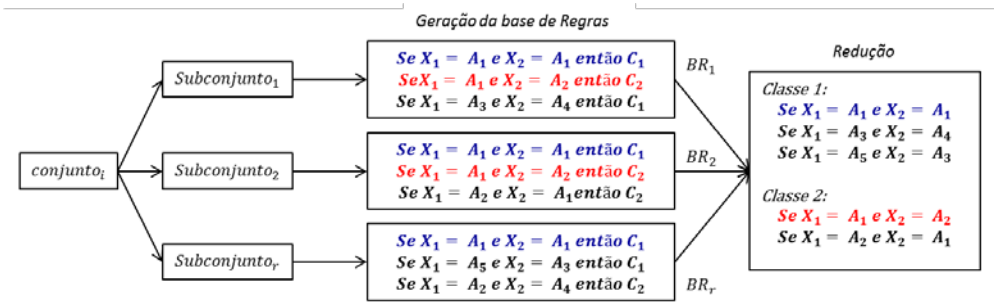

Figura 12. Exemplo do Processo de Redução de Regras

Este processo é realizado para cada $Conjunto_i$, $i \in \{1, \ldots, s\}$ de dados, gerando *s* diferentes classificadores, compostos de regras fuzzy para cada classe *k*.

O próximo módulo do modelo RandomFIS envolve a combinação de todos os classificadores.

3.4. Combinação de Classificadores

Seja uma coleção de *s* classificadores e um novo padrão $\mathbf{x}_i^*$. A saída inferida pelo classificador $s_j, j \in \{1, \ldots, s\}$ para este padrão é dada por $Y_j^*(\mathbf{x}_i^*) = \left[\hat{\mu}_{C_i \in 1}(\mathbf{x}_i^*), \ldots, \hat{\mu}_{C_i \in K}(\mathbf{x}_i^*)\right]$. A combinação de *s* resultados como este pode ser descrita matematicamente por [45]:

$$Y^*\left(\mathbf{x}_i^*\right) = f\left(Y_1^*\left(\mathbf{x}_i^*\right), \ldots, Y_j^*\left(\mathbf{x}_i^*\right), \ldots, Y_S^*\left(\mathbf{x}_i^*\right)\right) \quad (8)$$

em que $Y^*(\mathbf{x}_i^*)$ é a saída do comitê formado pelos *s* classificadores base e *f*(.) é um método de combinação.

O RandomFIS emprega dois métodos de combinação conhecidos: voto e média, definidos como segue.

Voto (*Voting*) é uma das formas mais simples de realizar a combinação anterior. Cada classificador base tem direito a votar em uma classe, sendo a classe mais votada a escolhida para representar a amostra na entrada do comitê. Para este tipo de combinação, a saída deve ser do tipo rótulo; assim, o RandomFIS torna as

saídas de cada classificador $s_j, j \in \{1, \ldots, s\}$ em um rótulo tipo binário, como segue:

$$Y_j^*\left(\mathbf{x}_i^*\right) = arg_k max\left\{\hat{\mu}_{C_i \in 1}\left(\mathbf{x}_i^*\right), \ldots, \hat{\mu}_{C_i \in k}\left(\mathbf{x}_i^*\right), \ldots, \hat{\mu}_{C_i \in K}\left(\mathbf{x}_i^*\right)\right\} \quad (9)$$

onde $Y_j^*(\mathbf{x}_i^*)$ é a saída do classificador em forma binária, resultado do *k*-ésimo argumento que maximiza (9). Sendo $\mathbf{v}(\mathbf{x}_i^*)$ o vetor de saída do *s*-ésimo classificador base para uma entrada $\mathbf{x}_i^*$, ele possui, em *K*, um valor proporcional à frequência de escolha da classe *k* entre os classificadores base:

$$\mathbf{v}\left(\mathbf{x}_\mathbf{i}^*\right) = \sum_{j=1}^{s} \mathbf{Y}_j^*(\mathbf{x}_\mathbf{i}^*) = \left[v_1(\mathbf{x}_\mathbf{i}^*), \ldots, v_C(\mathbf{x}_\mathbf{i}^*)\right] \quad (10)$$

A Figura 13 exemplifica o método de votação para três classificadores com 3 classes de saída.

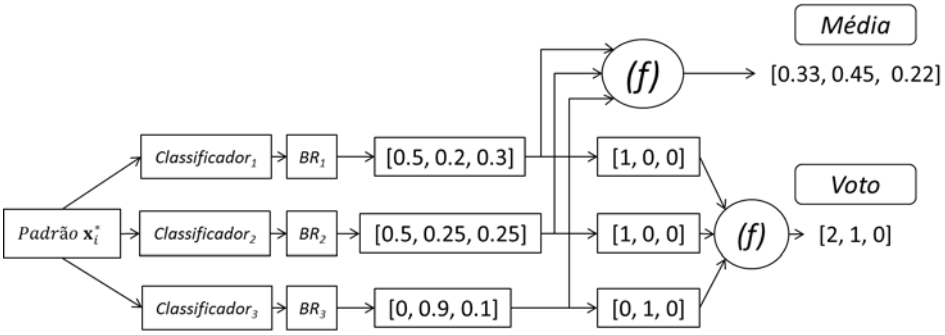

Figura 13. Combinação de Classificadores pelo voto e pela média

A classe $Y^*(\mathbf{x})$ inferida pelo comitê para a amostra $\mathbf{x}_i^*$ é aquela que maximiza a eq. 17 da etapa de Decisão (próxima seção). No caso da Figura 13, a saída é classe 2.

Média: este método é aplicado sem a necessidade de tornar a saída binária. Tomam-se os graus de pertinência fornecidos pelos *s* classificadores e, de forma similar ao voto, o vetor $\mathbf{v}(\mathbf{x}_i)$ é dado por:

$$\mathbf{v}\left(\mathbf{x}_i^*\right) = \sum_{j=1}^{S} \frac{\mathbf{Y}_j^*(\mathbf{x}_i^*)}{s} = \left[m_1(\mathbf{x}_i^*), \ldots, m_K(\mathbf{x}_i^*)\right] \quad (11)$$

A Figura 13 exemplifica, também, o método de média para três classificadores com 3 classes de saída.

Finalmente, a classe $Y^*(\mathbf{x})$ inferida pelo comitê para a amostra $\mathbf{x}_i$ é aquela que maximiza a eq. 17 da etapa de Decisão.

Propõe-se, neste trabalho, uma combinação de classificadores para gerar um só classificador. Para se obter um sistema com maior interpretabilidade (poucas regras), tomam-se as bases de regras BR_1, BR_2,…, BR_s geradas por cada classificador e se aplica uma redução de regras para gerar a base de regras final *BR* (Figura 14).

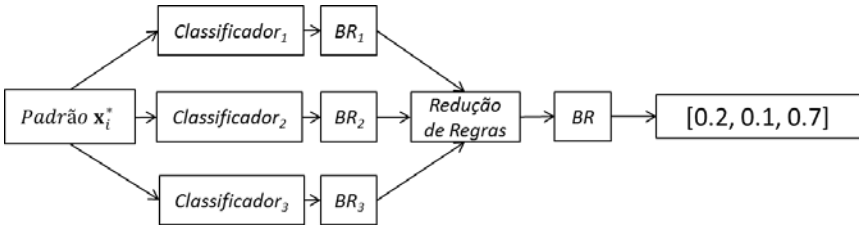

Figura 14**.** Combinação de Classificadores pela fusão das bases de regras

Esta versão do RandomFIS, denominada RandomFIS-Especialista, gera uma base de regras mais compacta e com maior acurácia. A denominação RandomFIS-Comitê será usada quando forem empregados os métodos de voto e média.

Gerada a base de regras, um novo padrão $\mathbf{x}_i^*$ pode ser avaliado agregando-se as regras pelo método de Média Ponderada apresentado na próxima seção.

3.5. Agregação de Regras

A operação de Agregação visa a reunir os graus de pertinência de um novo padrão $\mathbf{x}_i^*$ às regras pertencentes ao mesmo consequente, de modo a gerar um valor consensual.

Considere-se $D^{(k)}$ o numero de regras associadas ao k-ésimo termo consequente. Dado um operador de agregação $g: [0,1]^{D^{(k)}}$, o grau de pertinência estimado de $\mathbf{x}_i^*$ ao k-ésimo termo consequente ($\hat{\mu}_{C_i \in k}(\mathbf{x}_i^*)$) é:

$$\hat{\mu}_{C_1 \in k}\left(\mathbf{x}_i^*\right) = g\left[\ \mu_{A_1(1)}(\mathbf{x}_i^*),\ldots,\ \mu_{A_D(1)}(\mathbf{x}_i^*)\right] \tag{12}$$

$$\hat{\mu}_{C_2 \in k}\left(\mathbf{x}_i^*\right) = g\left[\ \mu_{A_1(2)}(\mathbf{x}_i^*),\ldots,\ \mu_{A_D(2)}(\mathbf{x}_i^*)\right] \tag{13}$$

…

$$\hat{\mu}_{C_i \in k}\left(\mathbf{x}_i^*\right) = g\left[\ \mu_{A_1(K)}(\mathbf{x}_i^*), \ldots,\ \mu_{A_D(K)}(\mathbf{x}_i^*)\right] \tag{14}$$

O RandomFIS emprega uma média ponderada como operador de agregação. O procedimento é similar ao da etapa de Associação (seção 3.2.2.2), ou seja, faz uso de uma Média Ponderada estimada via Mínimos Quadrados Restritos:

$$\hat{\mu}_{C_1 \in k}\left(\mathbf{x}_i^*\right) = \sum_{d^{(1)}=1}^{D^{(1)}} w_{d^{(1)}} \mu_{A_d(1)}\left(\mathbf{x}_i\right) \tag{15}$$

…

$$\hat{\mu}_{C_i \in k}\left(\mathbf{x}_i^*\right) = \sum_{d^{(K)}=1}^{D^{(K)}} w_{d^{(K)}} \mu_{A_d(K)}\left(\mathbf{x}_i\right) \tag{16}$$

onde $w_{d^{(k)}}$ é o peso ou grau de influencia de $\mu_{A_d(k)}(\mathbf{x}_i)$ na discriminação de padrões oriundos da classe *k*.

Computado o grau de pertinência predito de $\mathbf{x}_i^*$ para cada uma das *K* classes (para todos os *s* diferentes classificadores gerados), executa-se uma etapa de combinação dessas saídas para fornecer uma resposta do sistema em graus de pertinência. Finalmente, a etapa de Decisão fornece uma saída única do sistema.

3.6. Decisão

Dado um novo padrão de entrada $\mathbf{x}_i^*$, a classe $Y^*(\mathbf{x}_i^*)$ inferida pelo sistema é computada como:

$$Y^*\left(\mathbf{x}_i\right) = \arg_{k=1,2,\ldots,K} \max\left\{v_k(\mathbf{x}_i)\right\} \tag{17}$$

onde $Y^*(\mathbf{x}_i^*)$ é a classe predita, resultado do k-ésimo argumento que maximiza a expressão (17). Assim, esse método associa o padrão $\mathbf{x}_i^*$ à classe à qual é mais pertinente, segundo as saídas fornecidas pelos classificadores. Quando há empate, o sistema RandomFIS associa $\mathbf{x}_i^*$ à classe que possui mais padrões na base de dados.

O próximo capítulo apresenta diferentes testes do modelo RandomFIS, efetuados a partir de experimentos com benchmarks presentes na literatura. Os

resultados são comparados com os obtidos por alguns dos Sistemas Fuzzy voltados para lidar com alta dimensionalidade.

4
Estudo de Casos

Este capítulo apresenta uma série de estudos de casos, cuja finalidade principal é exibir a amplitude e qualidade das soluções fornecidas pelo modelo RandomFIS. Sua avaliação foi realizada em duas etapas: análise dos diferentes parâmetros do modelo e comparação do seu desempenho com os de outros modelos desenvolvidos para tratamento de grandes bases de dados, conforme descrito a seguir.

- Primeira etapa – Avaliação do comportamento do RandomFIS, em termos de acurácia, número de regras e tempo computacional, para diferentes parâmetros, considerando bases de dados para as quais o AutoFIS se mostrou ineficiente.
- Segunda etapa – Comparação do RandomFIS com outros modelos presentes na literatura, para diferentes bases de dados benchmark, avaliando-se acurácia e número de regras.

Empregou-se a abordagem [46] de efetuar análises estatísticas sempre que possível, com o objetivo de verificar qual método apresenta resultados significativamente superiores para um certo conjunto de bases de dados. Para verificar se a diferença de acurácia (maximização) e do número de regras (minimização) entre os modelo é significativa, foram aplicados os testes de Friedman e Holm [47]. Utilizaram-se as rotinas estatísticas do KEEL [48].

As bases de dados usadas nas aplicações benchmark foram, em sua maioria, obtidas da University of California Irvine (UCI) [49] e do repositório KEEL [48][48]. Todas as rotinas foram implementadas em Python 2.7.9, por ser esta uma linguagem livre e eficiente em comparação a outras, como Matlab, e executadas principalmente em um PC Windows 7 com processador Intel i7 3.4GHz, 16GB de RAM. Alguns experimentos foram realizados em um PC com processador Intel i7 3.2Ghz, 32GB de RAM, com Windows 7.

Foram consideradas 31 bases de dados, agrupadas em bases binárias e de múltiplas classes, cujas principais características são mostradas nas Tabelas 5 e 6, respectivamente.

Tabela 5. Principais características das bases de dados binárias

Base de dados	**Classes**	**Atributos**	Instâ**ncias**
German	2	20	1000
Ionosphere	2	33	351
Ring	2	20	7400
Sonar	2	60	208
Spambase	2	57	4597
Spectfheart	2	44	267
Two-norm	2	20	7400
Wdbc	2	30	569
Heart	2	13	270
Magic	2	10	19020
Phoneme	2	5	5404
Pima	2	8	768
Kddcup_DOS_vs_normal	2	41	4856151
Poker_0_vs_1	2	10	946799
Covtype_2_vs_1	2	54	495141
Census	2	41	141544
Fars_Fatal_Inj_vs_No_Inj	2	29	62123

Tabela 6. Principais características das bases de dados de múltiplas classes

Base de dados	**Classes**	**Atributos**	**Instancias**
Automobile	6	25	205
Dermatology	6	34	358
Movementlibras	15	90	360
Optdigits	10	64	5620
Penbased	10	16	10992
Satimage	6	36	6435
Segment	7	19	2310
Texture	11	40	5500
Wine	3	13	178
Vehicle	4	18	846
Page-blocks	5	10	5472
Letter	26	16	20000
Glass	7	9	214
Yeast	10	8	1484

As 31 bases de dados foram escolhidas para permitir uma comparação do modelo RandomFIS com o AutoFIS e com outros sistemas desenvolvidos para problemas de alta dimensionalidade. A avaliação do modelo segue as linhas de [14, 21, 22, 50, 51, 52], a saber:

- Validação cruzada de 10 pastas (10-fold-cv) e 5x2 pastas (5x2-fold-cv, a qual, segundo [53], é considerada superior a *k-folds* validação cruzada em problemas de classificação, por ter uma maior significância estatística. Em 5x2-fold-cv, cinco duplas de validação cruzada são executadas. O conjunto de dados é divido aleatoriamente em duas metades: um é usado para treinamento e o outro para teste e vice-versa. O procedimento é repetido cinco vezes, cada uma com uma nova partição metade/metade.
- Avaliação do parâmetro gamma (γ), número de classificadores *s* e métodos de combinação de classificadores.
- Obtenção do tempo computacional.
- Obtenção de acurácia do teste.
- Obtenção do número de regras

4.1. Investigação Empírica da Arquitetura do Modelo RandomFIS

A partir da descrição anterior, cria-se um experimento para cada combinação dos parâmetros; gamma (γ=[0.6-0.9]), número de classificadores (s=[5,7,15]), e método de combinação de classificadores (voto, média, e fusão de regras). O parâmetro *r* será fixado para 100 ao fim de garantir uma maior convergência, segundo avaliações levadas em [20], de modo a caracterizar o sistema RandomFIS, em termos de acurácia, número de regras e tempo computacional.

Os parâmetros usados para gerar cada classificador fuzzy foram definidos de acordo com os melhores resultados obtidos em [14] e apresentados na Tabela **7**.

Tabela 7. Parâmetros para cada classificador fuzzy

Parâmetro	**Valor**
Formato da função de pertinência	tukey
No. de Funções de pertinência	3
t-norma	Produto
Negação	Ativada
Máximo Tamanho da premissa	2
Limiar de Suporte (ε_{sup})	0.075
Limiar de Similaridade (ε_{sim})	0.95
Associação	MQR
Agregação	MQR

À Tabela **7** é adicionado outro parâmetro, conhecido como filtro PCD, aplicado somente em bases de dados binárias [14].

Nesta seção serão empregadas 16 das 31 bases de dados apresentadas na Tabela 5 e na Tabela 6. As bases selecionadas são aquelas para as quais o classificador base (AutoFIS) apresentou problemas de escalabilidade [14]. As principais informações das 16 bases de dados consideradas neste experimento são mostradas na Tabela 8.

Tabela 8. Bases de dados consideradas na caracterização.

Base de dados	**Classes**	**Atributos**	**Instâncias**
German	2	20	1000
Ionosphere	2	33	351
Ring	2	20	7400
Sonar	2	60	208
Spambase	2	57	4597
Spectfheart	2	44	267
Two-norm	2	20	7400
Wdbc	2	30	569
Automobile	6	25	205
Dermatology	6	34	358
Movementlibras	15	90	360
Optdigits	10	64	5620
Penbased	10	16	10992
Satimage	6	36	6435
Segment	7	19	2310
Texture	11	40	5500

A função de avaliação em acurácia é computada como:

$$Acurácia = \frac{\sum_{i=1}^{n} \left| C_{i\in k}(\mathrm{x}_i^*) - \hat{C}_{i\in k}(\mathrm{x}_i^*) \right|}{n} \quad (18)$$

onde $C_{i\in k}$ é a classe real do padrão $\mathbf{x}_i^*$ e $\hat{C}_{i\in k}$ é a classe predita. $\left| C_{i\in k} - \hat{C}_{i\in k} \right| = 0$, se $C_{i\in k} = \hat{C}_{i\in k}$ e 1, caso contrario. O próximo tópico exibe os resultados e análises do experimento. Os resultados foram avaliados independentemente para bases binárias e múltiplas classes.

4.1.1. Acurácia

4.1.1.1. Bases de Dados Binárias

As Tabelas 9-12 apresentam a acurácia (%) obtida para cada configuração descrita no experimento em bases binárias. É possível verificar que, para aproximadamente 81.25% das bases de dados em cada variação de γ, o método de combinação por fusão de regras obteve os melhores resultados, demostrando ser uma boa alternativa em termos de acurácia. Foi seguido de média, com aproximadamente 8.75%. Há uma semelhança de resultados entre média e voto, sendo a primeira um pouco superior. Este resultado indica que uma fusão de classificadores aplicada a partir das bases de regras possibilita ganhos de acurácia na maioria dos casos. A partir do melhor método de combinação, tenta-se determinar um número aproximado de classificadores e do parâmetro γ.

Tabela 9. Resultados em termos de acurácia (%) para bases binárias.

$\gamma = 0.6$	Média			Voto			Fusão regras		
Base de dados	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**
german	70.8	72.1	70.9	71.3	71.5	71.1	73	**75.1**	73.8
ionosphere	78.35	81.76	79.51	78.64	78.91	79.79	86.33	86.9	**87.48**
ring	68.5	70.11	67.52	69.05	70.42	68.03	72.42	**72.39**	72.38
sonar	72.98	74.02	77.86	67.26	70.19	77.86	**78.29**	72.02	76.36
spambase	88.95	89.1	88.69	88.73	88.95	88.58	89.86	89.95	**90.34**
spectfheart	79.42	79.79	79.42	79.42	79.79	79.42	78.66	80.16	**80.93**
two-norm	94.2	95.01	**96.58**	93.11	93.95	96.16	94.11	94.27	93.91
wdbc	90.86	93.14	93.14	91.38	92.79	92.44	92.62	92.97	**93.67**
Média	80.51	81.88	81.7	79.86	80.81	81.67	83.16	82.97	**83.61**

Tabela 10. Resultados em termos de acurácia (%) para bases binárias.

$\gamma = 0.7$	Média			Voto			Fusão Regras		
Bases de dados	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**
german	71.80	71.40	71.80	72.30	71.50	71.90	**74.1**	73.4	73.9
ionosphere	83.50	80.94	82.09	82.94	80.37	82.66	86.06	85.48	**86.35**
ring	70.46	72.19	69.34	70.95	72.05	70.35	72.54	**72.64**	72.28
sonar	74.43	76.36	76.40	73.45	75.88	76.36	77.31	**77.79**	76.4
spambase	89.80	89.58	89.69	89.58	89.54	89.54	90.02	90.12	**90.3**
spectfheart	79.42	79.42	79.42	79.79	79.79	79.42	**80.17**	79.42	79.79
two-norm	95.31	95.70	**96.41**	94.70	94.95	96.06	93.68	93.82	93.86
wdbc	93.14	91.91	93.32	92.97	92.26	93.49	**94.02**	93.32	93.85
Média	82.23	82.19	82.31	82.08	82.04	82.47	**83.49**	83.25	83.34

Tabela 11. Resultados em termos de acurácia (%) para bases binárias.

$\gamma = 0.8$	Média			Voto			Fusão Regras		
Bases de dados	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**
german	72.50	72.40	72.00	71.90	72.20	72.50	74	73.4	**74.8**
ionosphere	82.93	83.21	84.37	83.20	82.08	83.51	85.77	**86.92**	86.63
ring	72.08	70.46	71.43	71.95	71.09	71.57	72.45	**72.68**	72.39
sonar	74.48	76.81	**78.74**	73.48	75.38	**78.74**	77.33	**78.74**	75.88
spambase	89.76	89.95	89.69	89.67	89.82	89.54	90.1	**90.47**	90.34
spectfheart	79.42	80.17	79.80	79.42	80.17	79.80	79.79	**80.19**	79.79
two-norm	95.57	95.70	**96.11**	95.25	95.30	96.10	94.12	94.14	93.85
wdbc	93.14	93.14	93.15	92.97	92.79	92.80	93.15	**93.85**	93.5
Média	82.48	82.73	83.16	82.23	82.35	83.07	83.34	**83.80**	83.40

Tabela 12. Resultados em termos de acurácia (%) bases binárias.

$\gamma = 0.9$	Média			Voto			Fusão Regras		
Bases de dados	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**
german	73.50	73.20	73.60	73.70	72.90	73.80	73.9	**74.4**	74
ionosphere	86.63	85.50	86.07	85.78	85.79	86.07	86.64	**88.9**	86.62
ring	72.15	72.08	72.06	72.19	72.16	72.22	72.58	72.61	**72.65**
sonar	76.88	76.38	77.78	76.36	76.86	76.86	77.31	76.38	**78.29**
spambase	90.17	90.13	90.06	89.99	89.91	89.82	90.15	90.25	**90.34**
spectfheart	79.42	80.17	79.80	79.42	80.17	79.80	79.79	80.17	**80.53**
two-norm	95.69	95.72	**96.00**	95.42	95.55	95.98	94.36	94.09	93.91
wdbc	92.97	**93.85**	92.97	93.49	**93.85**	93.32	93.67	**93.85**	**93.85**
Média	83.42	83.38	83.54	83.29	83.40	83.48	83.55	83.83	**83.77**

A Tabela 13 agrupa os resultados, para bases binárias, do método de combinação por fusão de regras. É possível verificar que em torno de 62.5% das bases de dados os melhores resultados foram obtidos para sete classificadores, seguido de cinco classificadores em 25% dos casos. Também pode ser observado que $\gamma = 0.8$ fornece os melhores resultados em torno de 37.5% dos casos, seguido de 0.6 e 0.9 (25% das bases de dados). Observe-se, todavia, que não existe uma diferença significativa na maioria dos casos.

Tabela 13. Acurácia (%):combinação de classificadores por fusão de regras.

	Fusão regras											
	$\gamma = 0.6$			$\gamma = 0.7$			$\gamma = 0.8$			$\gamma = 0.9$		
Base de dados	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**
german	73.00	**75.10**	73.80	74.10	73.40	73.90	74.00	73.40	74.80	73.90	74.40	74.00
ionosphere	86.33	86.90	87.48	86.06	85.48	86.35	85.77	86.92	86.63	86.64	**88.90**	86.62
ring	72.42	72.39	72.38	72.54	72.64	72.28	72.45	**72.68**	72.39	72.58	72.61	72.65
sonar	78.29	72.02	76.36	77.31	77.79	76.40	77.33	**78.74**	75.88	77.31	76.38	78.29
spambase	89.86	89.95	90.34	90.02	90.12	90.30	90.10	**90.47**	90.34	90.15	90.25	90.34
spectfheart	78.66	80.16	**80.93**	80.17	79.42	79.79	79.79	80.19	79.79	79.79	80.17	80.53
two-norm	94.11	94.27	93.91	93.68	93.82	93.86	94.12	94.14	93.85	**94.36**	94.09	93.91
wdbc	92.62	92.97	93.67	**94.02**	93.32	93.85	93.15	93.85	93.50	93.67	93.85	93.85

4.1.1.2. Múltiplas Classes

No caso das bases de dados de múltiplas classes, pode-se inferir, das Tabelas 14-17, que para aproximadamente 46.88% das bases de dados, em cada variação de γ, o método de combinação por fusão de regras auferiu os melhores resultados, demostrando ser novamente uma alternativa viável em termos de acurácia, seguido de média, com aproximadamente 40.63%. Há uma semelhança de resultados entre média e voto, sendo a primeira um pouco superior.

Diferentemente das bases de dados binárias, os três métodos apresentam acurácias substancialmente semelhantes, embora o método de fusão de regras proporcione melhores resultados de uma forma geral. A partir do melhor método de combinação, tenta-se determinar um número aproximado de classificadores e do parâmetro γ.

Tabela 14. Resultados em acurácia (%) para múltiplas classes.

$\gamma = 0.6$	**Média**			**Voto**			**Fusão Regras**		
Base de dados	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**
automobile	55.57	54.18	57.58	51.53	55.44	58.00	**68.34**	62.25	67.79
dermatology	94.98	96.06	94.94	93.81	95.50	96.08	96.35	96.62	**96.9**
movementlibras	44.17	48.33	50.83	41.11	49.44	51.11	53.61	59.17	**61.11**
optdigits	90.64	90.89	**91.66**	89.22	90.25	91.26	91.16	91.55	91.6
penbased	85.88	86.22	85.98	85.15	85.71	85.89	86.12	**86.37**	86.26
satimage	79.45	80.13	80.40	79.58	80.03	**80.85**	80.5	80.53	80.54
segment	86.62	86.32	86.84	86.02	86.50	87.14	87.97	88.4	**88.18**
texture	83.45	83.95	**84.93**	81.40	82.91	84.20	82.89	83.29	83.58
Média	77.60	78.26	79.14	75.98	78.22	79.32	80.87	81.02	**82.00**

Tabela 15. Resultados em acurácia (%) para múltiplas classes.

$\gamma = 0.7$	Média			Voto			Fusão Regras		
Base de Dados	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**
automobile	61.64	58.05	58.87	59.50	62.88	61.58	67.74	67.07	**66.77**
dermatology	95.51	96.62	**97.18**	94.93	96.63	96.61	96.62	96.62	96.34
movementlibras	56.39	56.67	62.50	53.33	55.28	65.28	61.94	65.28	**67.22**
optdigits	91.37	91.92	**92.31**	91.05	91.44	92.24	91.64	91.96	92.19
penbased	86.43	**86.59**	86.27	86.35	86.35	86.21	86.41	86.44	86.55
satimage	80.40	80.54	80.45	80.34	80.50	**80.67**	80.36	80.65	80.25
segment	87.19	87.58	87.10	86.62	87.79	87.10	88.14	**89.13**	88.83
texture	**84.98**	84.40	84.91	83.60	83.53	84.47	83.87	84.04	83.8
Média	80.49	80.30	**81.20**	79.47	80.55	81.77	82.09	82.65	**82.74**

Tabela 16. Resultados em acurácia (%) para múltiplas classes.

$\gamma = 0.8$	Média			Voto			Fusão Regras		
Base de Dados	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**
automobile	60.26	63.44	65.13	62.91	61.50	66.45	70.99	72.25	**73.12**
dermatology	96.91	96.92	96.90	96.61	96.63	**97.17**	96.91	96.91	96.33
movementlibras	60.00	61.67	63.61	60.56	62.78	65.00	67.22	64.17	**68.89**
optdigits	91.92	**92.30**	92.13	91.58	92.21	92.10	91.92	92.28	92.14
penbased	**86.58**	86.42	86.43	86.34	86.27	86.46	86.5	86.39	86.55
satimage	80.62	80.78	**80.95**	80.67	80.75	80.90	80.65	80.59	80.67
segment	87.92	88.36	88.01	87.66	88.53	88.14	88.4	**88.96**	88.61
texture	**85.15**	84.58	84.73	84.22	84.11	84.42	83.36	83.73	83.64
Média	81.17	81.81	82.24	81.32	81.60	82.58	83.24	83.16	**83.74**

Tabela 17. Resultados em acurácia (%) para múltiplas classes.

$\gamma = 0.9$	Média			Voto			Fusão Regras		
Base de Dados	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**
automobile	65.53	68.14	66.32	65.86	65.43	67.99	71.25	71.02	**73.61**
dermatology	96.05	96.90	96.89	96.06	**97.18**	97.17	96.62	96.9	96.62
movementlibras	63.05	66.39	66.39	65.00	66.67	64.72	**68.33**	68.33	68.06
optdigits	92.33	92.30	**92.54**	92.10	92.21	92.26	92.22	92.14	92.31
penbased	86.41	86.23	86.36	86.26	86.14	86.17	86.44	86.49	**86.59**
satimage	80.79	80.95	**81.28**	80.84	80.97	81.12	80.53	80.7	80.65
segment	88.70	88.18	88.66	88.44	87.92	88.66	**89.22**	88.61	88.44
texture	84.93	84.73	**85.22**	84.42	84.58	84.82	83.71	83.87	84
Média	82.22	82.98	**82.96**	82.37	82.63	82.86	83.54	83.51	83.79

A Tabela 18 agrupa os resultados do método de combinação por fusão de regras para múltiplas classes. É possível verificar que, para em torno de 50% das bases de dados, os melhores resultados foram obtidos para quinze classificadores, seguido de sete classificadores para 25% dos casos. Também pode ser observado que, em 62.5% dos casos, $\gamma = 0.9$ fornece os melhores resultados, seguido de 0.8 para 25% das bases de dados. Todavia, na maioria dos casos não existe uma diferença significativa.

Tabela 18. Acurácia (%): combinação de classificadores por fusão de regras

	Fusão Regras											
	$\gamma = 0.6$			$\gamma = 0.7$			$\gamma = 0.8$			$\gamma = 0.9$		
Base de dados	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**
automobile	68.34	62.25	67.79	67.74	67.07	66.77	70.99	72.25	73.12	71.25	71.02	**73.61**
dermatology	96.35	96.62	96.9	96.62	96.62	96.34	**96.91**	**96.91**	96.33	96.62	96.9	96.62
movementlibras	53.61	59.17	61.11	61.94	65.28	67.22	67.22	64.17	**68.89**	68.33	68.33	68.06
optdigits	91.16	91.55	91.6	91.64	91.96	92.19	91.92	92.28	92.14	92.22	92.14	**92.31**
penbased	86.12	86.37	86.26	86.41	86.44	86.55	86.5	86.39	86.55	86.44	86.49	**86.59**
satimage	80.5	80.53	80.54	80.36	80.65	80.25	80.65	80.59	80.67	80.53	**80.7**	80.65
segment	87.97	88.4	88.18	88.14	89.13	88.83	88.4	88.96	88.61	**89.22**	88.61	88.44
texture	82.89	83.29	83.58	83.87	**84.04**	83.8	83.36	83.73	83.64	83.71	83.87	84

4.1.2. Número de Regras

Como apresentado na seção 3.5, o RandomFIS apresenta duas versões, diferenciadas pela técnica de combinação empregada: RandomFIS-Comitê (voto e média) e RandomFIS-Especialista (fusão de regras). A partir das bases de regras dos classificadores de RandomFIS-Comitê para cada classe, o RandomFIS-Especialista executa uma redução de regras, gerando uma base de regras final. As Tabelas 19 e 20 exibem o fator de redução para bases de dados binárias e múltiplas classes, respectivamente. Verifica-se que a redução é de 2 a 10 vezes, aproximadamente, e que um aumento no número de classificadores gera um número maior de regras (apresenta comportamento linear) na maioria dos casos.

Tabela 19. Fator de redução do número de regras: RandomFIS-Especialista versus RandomFIS-Comitê em bases de dados binárias.

	$\gamma = 0.6$			$\gamma = 0.7$			$\gamma = 0.8$			$\gamma = 0.9$		
Base de dados	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**
german	**2.73**	3.49	6.40	2.85	4.17	7.31	3.39	4.46	7.82	3.21	4.42	7.69
ionosphere	**3.44**	4.33	9.10	3.76	5.22	10.03	4.23	5.55	10.97	4.09	5.42	9.98
ring	**2.54**	3.27	6.53	2.72	3.87	7.36	3.08	4.06	7.97	3.81	4.83	9.79
sonar	**2.91**	4.08	7.65	3.27	4.42	8.62	4.16	5.07	9.81	4.10	5.98	11.01
spambase	3.73	5.15	9.90	3.67	4.90	9.28	**3.60**	4.63	8.26	3.34	4.75	7.90
spectfheart	**2.71**	3.69	6.97	3.30	4.19	9.03	3.72	4.87	9.98	3.91	5.28	10.88
two-norm	**2.42**	3.03	5.76	2.67	3.53	6.68	3.09	3.95	7.52	3.29	4.24	7.64
wdbc	**4.28**	5.96	11.57	4.47	5.91	12.33	4.37	6.63	11.32	4.62	6.44	11.36

Tabela 20. Fator de redução do número de regras: RandomFIS-Especialista versus RandomFIS-Comitê para múltiplas classes.

	$\gamma = 0.6$			$\gamma = 0.7$			$\gamma = 0.8$			$\gamma = 0.9$		
Base de Dados	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**	**5**	**7**	**15**
automobile	4.27	5.05	10.07	**3.99**	5.44	10.18	4.26	5.66	10.67	4.05	5.35	10.60
dermatology	6.34	7.42	14.39	5.51	6.82	11.21	4.54	5.23	9.25	**3.61**	4.37	7.25
movementlibras	**2.94**	3.90	7.32	3.73	4.99	10.45	4.04	5.54	11.84	4.32	6.06	12.28
optdigits	**3.30**	4.13	7.29	3.10	4.09	6.71	3.13	3.89	6.95	3.07	3.93	7.01
penbased	**3.46**	4.60	8.89	3.60	5.04	10.23	3.83	5.06	10.69	3.93	5.30	11.16
satimage	**3.33**	4.38	8.21	3.61	4.50	8.77	3.53	4.52	8.53	3.48	4.63	8.52
segment	3.63	4.64	9.16	3.83	4.74	9.04	3.54	4.46	8.45	**3.55**	4.53	8.39
texture	4.59	6.07	11.58	4.27	5.94	10.95	4.16	5.31	10.16	**3.66**	5.01	9.69

4.1.3. Tempo Computacional

As figuras 15 e 16 exibem o tempo médio para cada conjunto de classificadores gerados para cada γ. Um aumento em qualquer um dos parâmetros implica em um maior investimento computacional, pois o RandomFIS gera os classificadores de forma serial. Verifica-se que, nos problemas de classificação com múltiplas classes, o tempo computacional é aproximadamente triplicado em comparação às bases binárias, independentemente do número de classes.

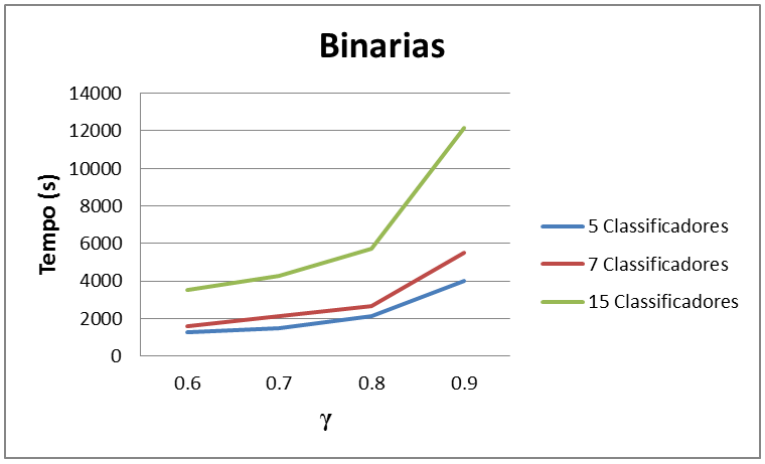

Figura 15. Tempo (s) médio para problemas binários.

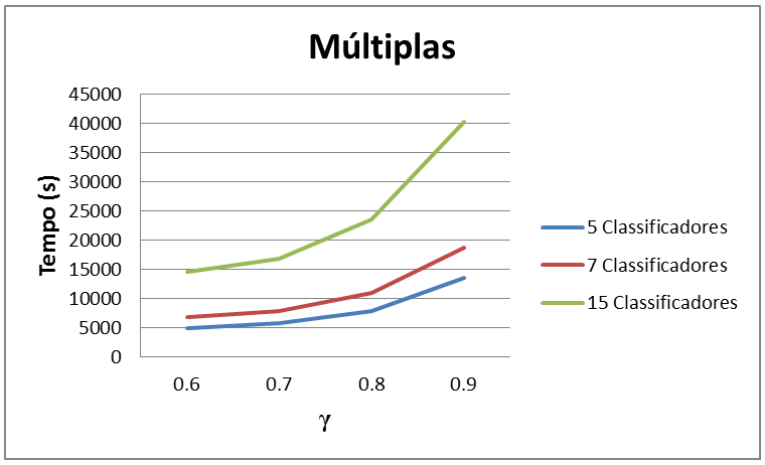

Figura 16. Tempo (s) médio para problemas com múltiplas classes.

A partir dos experimentos realizados, a configuração escolhida para o modelo RandomFIS, para bases binárias e múltiplas classes, é apresentada na Tabela 21.

Tabela 21. Configuração do modelo RandomFIS

Parâmetro	Binárias	Múltiplas
γ	0.8	0.9
Número de Classificadores	7	15
Método de Combinação	Fusão de Regras	Fusão de Regras

4.2. Comparação com o AutoFIS

As Figuras 17 e 18 apresentam os desempenhos dos modelos RandomFIS e AutoFIS. Verifica-se que o RandomFIS apresenta melhores resultados em termos de acurácia nos casos de *wdbc*, *spambase*, *ring* (classes binárias), *texture*, *segment* e *automobile* (classes múltiplas). O aumento na acurácia é reflexo do incremento no número de regras em relação àquele gerado pelo AutoFIS.

Em termos de memória, o AutoFIS não apresenta reposta alguma (gerando um erro de memoria no código) para as bases de dados *spectfheart*, *sonar* (classes binárias), *optidigits*, e *movementlibras* (classes múltiplas). Em termos de tempo computacional, o RandomFIS apresenta desempenho superior, reduzindo-o significativamente para todos os casos de bases binárias; em múltiplas classes, o AutoFIS apresenta melhor desempenho para as bases de dados *automobile*,

dermatology, *penbased* e *segment*. Isto se deve ao RandomFIS empregar uma verificação no número de classes, ao fim de garantir que todas elas estejam presentes na criação dos ***s*** conjuntos de dados, como apresentado na seção 3.1.

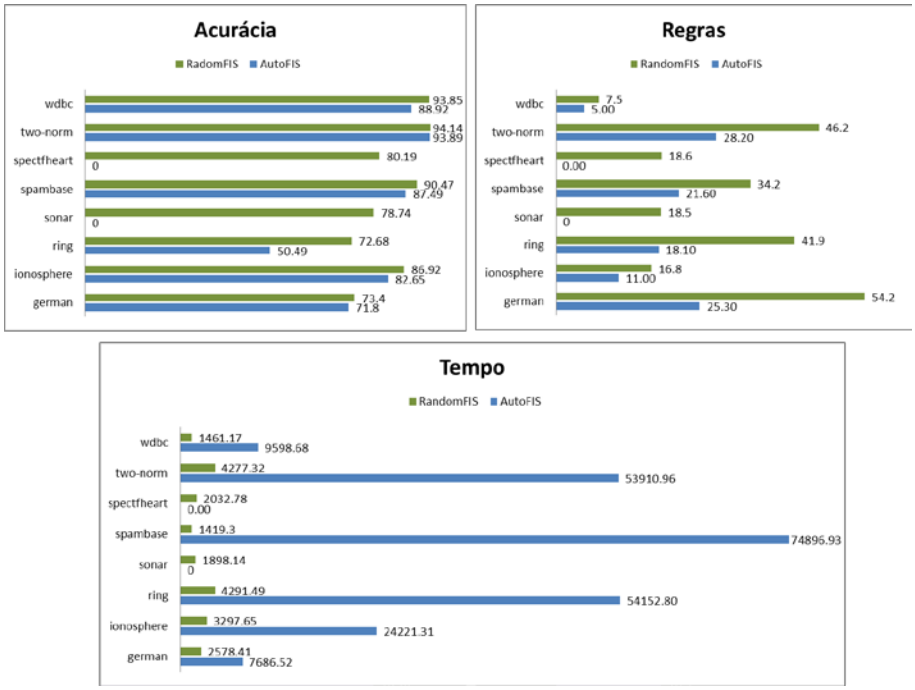

Figura 17. Desempenho RandomFIS vs AutoFIS em termos de acuracia (%), número de regras, tempo computacional para bases de dados binárias

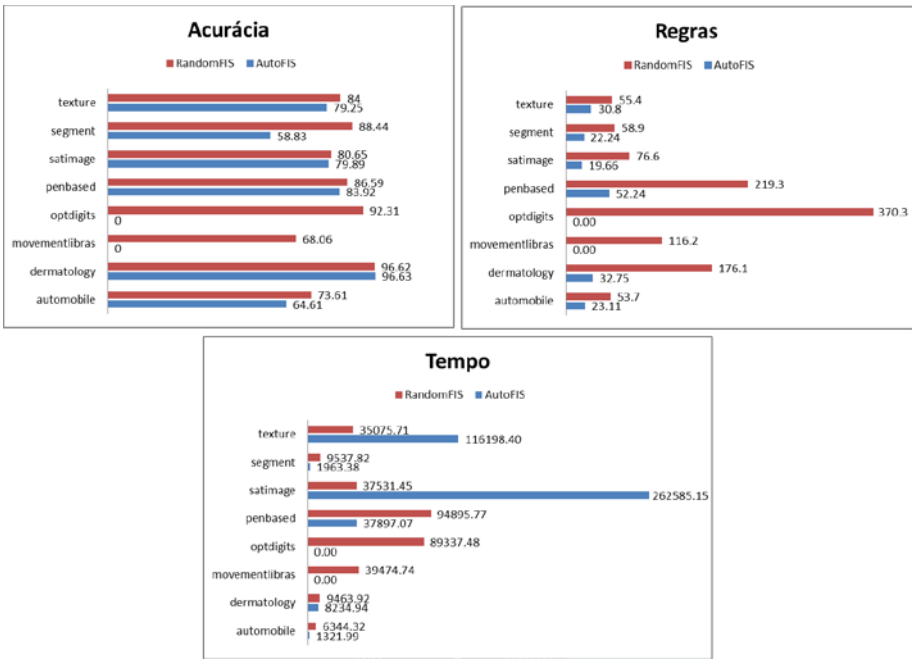

Figura 18 Desempenho RandomFIS vs AutoFIS em termos de acuracia (%), número de regras, tempo(s) para bases de dados em multiplas classes

4.3. Comparação com Sistemas de Inferência Fuzzy Voltados para Alta Dimensionalidade.

Esta seção apresenta comparações do modelo RandomFIS com outros modelos da literatura, divididas em quatro grupos:

- Grupo 1, do artigo [54], empregando-se 11 das 31 bases de dados, comparando-se os resultados de Acurácia (média 10 pastas de validação cruzada) e Número de regras com os do algoritmo FARC-HD, desenvolvido para mineração de regras fuzzy em bases de dados de elevada dimensionalidade e empregando uma inspiração de árvore e metodologias evolutivas na simplificação e aprimoramento da base de regras. O FARC-HD foi comparado com os algoritmos 2SLAVE, FH-GBML, SGERD, sendo estes do tipo GA.
- Grupo 2, do artigo [51], tomando-se 13 das 31 bases de dados, comparando-se os resultados de Acurácia (média 5x2 pastas de validação cruzada) com os do algoritmo FURIA-MCSs, voltado para problemas de alta dimensionalidade como extensão da abordagem FURIA [55], empregando os conceitos de *bagging* e seleção de variáveis. Foi comparado com Ishibuchi-MCS, abordagem de classificação fuzzy, e com C4.5-Bagging e RandomForest, algoritmos de aprendizado de máquina clássicos.
- Grupo 3, do artigo [52], usando-se 8 das 31 bases de dados, comparando-se os resultados de Acurácia (média 5x2 pastas de validação cruzada) e Número de regras com os de Ishibushi [7], um clássico sistema de inferência fuzzy e denotado como FRBCS – abordagem para alta dimensionalidade que emprega técnicas de *bagging* e de seleção de variáveis (FRBMCSs); outra versão faz uso de GA na seleção do número de classificadores do comitê (FRBMCSs-GA).
- Grupo 4, do artigo [56], utilizando-se 5 das 31 bases de dados, comparando-se resultados de Acurácia (média 10 pastas de validação cruzada) e Número de regras com os de Chi-FRBCS-BigData, baseado na abordagem Chi *et al's* [57], um clássico sistema de

inferência fuzzy e adaptado para lidar com problemas de alta dimensionalidade (*Big-Data*) via um esquema MapReduce [58].

Nas análises de desempenho dos modelos, foram empregados os testes estáticos de Friedman e Holm.

4.3.1. Grupo 1

A Tabela 22 apresenta os principais resultados para cada modelo na fase de teste, em termos de acurácia e número de regras, considerando diferentes bases de dados.

O modelo FARC-HD obteve os melhores resultados dentre todos os algoritmos para 81.82% das bases de dados; apresentou desempenho inferior somente para as bases *German* (binária) e *Spectfheart* (múltiplas classes). A acurácia média do FARC-HD foi em torno de 5.41% superior à do RandomFIS.

A Tabela 23 exibe os resultados fornecidos pelo Teste de Friedman na acurácia de cada algoritmo. Verifica-se que o posto médio do FARC-HD foi o menor de todos (1.2727), enquanto que o do RandomFIS foi o segundo menor (2.3636) – quase o dobro. Assim, o modelo FARC-HD foi selecionado como método de controle para comparação com as demais abordagens. O teste de Holm (Tabela 24) para múltiplas comparações evidencia que o modelo FARC-HD proporcionou, para as bases de dados analisadas, acurácia significativamente superior às dos demais algoritmos (p-valor <0.05).

Dado que o modelo FARC-HD obteve bons resultados em termos de acurácia, uma questão a considerar é se isto foi fruto do número reduzido de regras. A Tabela 25 apresenta igualmente o número de regras geradas para cada base de dados. Em todos os casos analisados, o FARC-HD gerou menos regras do que o modelo RandomFIS. Porém, a Tabela 25 no teste Friedman indica que o RandomFIS e o FARC-HD ocupam postos semelhantes (com 4.7273 e 4.2727, respectivamente), tornando-se necessário aplicar o teste de Holm (Tabela 26). Este denota que FARC-HD é estatisticamente semelhante ao RandomFIS. A compacidade do modelo FARC-HD deve-se ao emprego de rotinas evolutivas na simplificação ou aprimoramento da base de regras, ao passo que o RandomFIS não as utiliza.

Tabela 22. Comparação de acurácias e regras no Grupo 1

	Acurácia					**Regras**				
Dataset	**2SLAVE**	**FH-GBML**	**SGERD**	**FARC-HD**	**RandomFIS**	**2SLAVE**	**FH-GBML**	**SGERD**	**FARC-HD**	**RandomFIS**
German	70.53	87.01	67.97	72.8	**73.4**	6.5	5.1	**3.4**	85.7	54.2
Ringnorm	79.63	86.92	72.63	**94.08**	72.68	40	18.4	**15.9**	152.8	219.3
Sonar	71.42	68.24	71.9	**80.19**	78.74	**4.6**	6.9	6.8	24	41.9
Spambase	70.14	77.22	72.98	**91.93**	90.47	9.6	10.3	**3.2**	18	18.5
Spectfheart	79.17	72.36	78.16	79.83	**80.19**	6.1	10.8	**2.1**	12.9	18.6
Twonorm	86.99	85.97	73.98	**95.28**	94.14	26.5	12	**3.1**	60.9	46.2
Wdbc	92.33	92.26	90.68	**95.25**	93.85	5.2	7.2	**3.7**	10.4	7.5
Movementlibras	67.04	68.89	68.09	**76.67**	68.06	47.4	**12.1**	22.9	83.1	116.2
Pen-based	81.16	50.45	67.93	**96.04**	86.59	7.9	3.9	**3.7**	29.8	34.2
SatImage	81.69	74.72	77.1	**87.32**	80.65	57.9	16.5	**12.2**	76.1	76.6
Texture	81.57	70.15	71.66	**92.89**	84	34.9	**14.6**	18.6	54.5	55.4
Média	78.33	75.84	73.92	**87.48**	82.07	22.42	10.71	**8.69**	55.29	62.60

Tabela 23. Posto médio obtido por cada algoritmo no teste de Friedman para acurácias no Grupo 1.

Algoritmo	Posto
SGERD	4.1818
FH-GBML	3.7273
2SLAVE	3.4545
RandomFIS	2.3636
FARC-HD	1.2727

Tabela 24. p-valores obtidos na aplicação do método post hoc sobre os resultados de Friedman para acurácias no Grupo 1.

Algoritmo	$z = (R_0 - R_i / SE)$	*p-valor*	Holm
SGERD	4.314879	0.000016	0.0125
FH-GBML	3.640679	0.000272	0.016667
2SLAVE	3.236159	0.001211	0.025
RandomFIS	1.61808	0.105645	0.05

Tabela 25. Posto médio obtido por cada Algoritmo no teste de Friedman para regras no Grupo 1.

Algoritmo	Posto
RandomFIS	4.7273
FARC-HD	4.2727
2SLAVE	2.5455
FH-GBML	2.1818
SGERD	1.2727

Tabela 26. p-valores obtidos na aplicação do método post hoc sobre os resultados de Friedman para regras no Grupo 1.

Algoritmo	$z = (R_0 - R_i / SE)$	*p-valor*	Holm
RandomFIS	5.123919	0.0001	0.0125
FARC-HD	4.449719	0.00001	0.016667
2SLAVE	3.236159	0.001211	0.025
FH-GBML	1.3484	0.17753	0.05

4.3.2. Grupo 2

A Tabela 27 apresenta os principais resultados para cada modelo na fase de teste, em termos de acurácia, considerando diferentes bases de dados. Para 46% e 38.6% das bases de dados RandomForest e RandomFIS geram os melhores resultados, porém o FURIA-MCSs gera a melhor média.

A Tabela 28 exibe os resultados fornecidos pelo Teste de Friedman para a acurácia. Observa-se que o algoritmo FURIA-MCSs apresentou menor posto médio (2.0000), seguido de RandomForest com 2.2692, ao passo que o RandomFIS ocupou um posto médio de 2.9231. O FURIA-MCSs foi selecionado para ser o método de controle. O teste de Holm da Tabela 29 evidencia que o FURIA-MCSs obteve, dentre as bases de dados analisadas, acurácia significativamente maior às dos demais algoritmos (p-valor < 0.05).

Tabela 27. Comparação de acurácias no Grupo 2

Base de dados	FURIA-MCSs	C45-Bagging	RandomForest	Ishibuchi-MCSs	RandomFIS
heart	82.2	80.6	78.9	78.7	**86.89**
ionosphere	86.6	85.1	86	87.1	**87.75**
magic	86.4	**86.6**	**86.6**	79.8	80.66
sonar	80.2	75.3	76.1	75.5	**81.15**
spambase	93.9	93.3	**94**	77.7	90.47
wine	96.4	90.3	95.2	94.4	**96.76**
vehicle	72.4	71.1	**73.1**	60.2	65.65
page-blocks	**97.2**	97	96.9	92.5	94.84
satimage	89.5	88.8	**89.6**	82.5	80.73
segment	96.5	95.8	**96.6**	83.4	88.84
optdigits	37.2	30.3	30.5	37.1	**92.3**
texture	**96.4**	94.9	96	74.4	84.13
letter	90.9	89.7	**92**	58	59.1
Média	**85.06**	82.98	83.96	75.48	83.79

Tabela 28. Posto médio obtido por cada algoritmo no teste de Friedman para acurácia no Grupo 2.

Algoritmo	Posto
Ishibuchi-MCSs	4.3846
C45-Bagging	3.4231
RandomFIS	2.9231
RandomForest	2.2692
FURIA-MCSs	2.0000

Tabela 29. p-valores obtidos na aplicação do método post hoc sobre os resultados de Friedman para acurácia no Grupo 2

Algoritmo	$z = (R_0 - R_i / SE)$	*p-valor*	Holm
Ishibuchi	3.845077	0.000121	0.0125
C45	2.294643	0.021754	0.016667
RandomFIS	1.488417	0.136641	0.025
RandomForest	0.434122	0.6642	0.05

4.3.3. Grupo 3

A Tabela 30 apresenta os principais resultados para cada modelo na fase de teste, em termos de acurácia e número de regras, considerando diferentes bases de dados. Em 62.5% das bases de dados, o modelo RandomFIS obteve os melhores resultados dentre todos os algoritmos baseados em comitê; teve desempenho inferior para as bases de dados Phoneme e Pima (classes binárias) e Vehicle (múltiplas classes). O RandomFIS apresentou acurácia média em torno de 0.15% superior à do FRBCMCSs-GA.

A Tabela 31 exibe os resultados fornecidos pelo Teste de Friedman para acurácia. É possível verificar que o posto médio do RandomFIS foi o menor de todos (1.75), enquanto que o do FRBMCSs-GA foi o segundo menor (1.875) embora quase iguais. O RandomFIS foi então selecionado como método de controle. O teste de Holm (Tabela 32) evidencia que o modelo RandomFIS obteve, dentre as bases de dados analisadas, acurácia significativamente superior às dos demais sistema de comitê fuzzy (p-valor <0.05).

Dado que o modelo RandomFIS obteve bons resultados em termos de acurácia, uma questão a considerar é se isto foi fruto do número reduzido de

regras. A Tabela 30 mostra que o RandomFIS gerou menos regras (uma média de quase três vezes) do que o modelo FRBMCSs-GA.

A maior compacidade do modelo RandomFIS (RandomFIS-Especialista) deve-se à etapa de redução de regras. Ressalte-se que o RandomFIS, ao contrário do FRBMCSs-GA, não faz uso de estratégias do tipo GA para selecionar e reduzir o número de classificadores no comitê.

Tabela 30. Comparação de acurácias e regras no Grupo 3

Base de dados	Acurácia			Regras		
	FRBMCSs	**FRBMCSs-GA**	**RandomFIS**	**FRBMCSs**	**FRBMCSs-GA**	**RandomFIS**
glass	60.4	64	**64.89**	257.4	125.4	**43.7**
heart	83.4	83	**86.74**	76.43	80.1	**38.9**
page-blocks	93.47	93.66	**94.67**	671.98	698.7	**39.7**
phoneme	81.8	**82**	74.93	2877	2872.8	**19.7**
pima	**76.6**	76.3	76.15	594.95	592	**16.2**
sonar	78	78.8	**82.6**	601.36	632	**20.4**
vehicle	67	**67.1**	65.32	1359	1356.3	**33.7**
yeast	55.6	55.9	**56.66**	487.77	1703.5	**83.2**
Média	74.53	75.10	**75.25**	865.74	1007.60	**36.94**

Tabela 31. Posto médio obtido por cada algoritmo no teste de Friedman para acurácia no Grupo 3

Algoritmo	Posto
FRBMCSs	2.375
FRBMCSs-GA	1.875
RandomFIS	1.75

Tabela 32. p-valores obtidos na aplicação do método post hoc sobre os resultados de Friedman para acurácias no Grupo 3

Algoritmo	$z = (R_0 - R_i \,/\, SE)$	p-valor	Holm
FRBMCSs	1.25	0.2113	0.025
FRBMCSs-GA	0.25	0.8026	0.05

4.3.4. Grupo 4

As Tabelas 33 e 34 exibem a acurácia e o número de regras, respectivamente, para cada modelo na fase de teste, considerando diferentes bases de dados.

Em 60% das bases de dados, o modelo RandomFIS obteve os melhores resultados; mostrou-se inferior somente para Kddcup_DOS_vs_normal e Poker_0_vs_1. Outra questão importante é que Chi-BigData faz uma divisão da base de dados sem uso de técnicas de re-amostragem, gerando perda de informação, com reflexo na acurácia quando o número de partições aumenta. O RandomFIS, em contrapartida, gera melhores resultados pelo uso dessas técnicas. As duas abordagens empregam etapas de fusão de regras; o RandomFIS é superior em suas duas versões, especialmente o RandomFIS-Especialista.

Tabela 33. Comparação de acurácias (%) no Grupo 4.

	Chi-BigData (CBD)						**RandomFIS (RF)**					
	8		**16**		**32**		**8**		**16**		**32**	
Base de Dados	**CBD-Max**	**CBD-Ave**	**CBD-Max**	**CBD-Ave**	**CBD-Max**	**CBD-Ave**	**RF-Comitê**	**RF-Esp.**	**RF-Comitê**	**RF-Esp.**	**RF-Comitê**	**RF-Esp.**
Kddcup_DOS_vs_normal	**99.93**	99.93	99.93	99.93	99.92	99.92	99.61	99.91	99.71	99.90	99.75	99.92
Poker_0_vs_1	**60.74**	60.91	59.88	60.35	58.93	59.30	57.07	58.46	56.77	59.27	56.98	60.02
Covtype_2_vs_1	74.63	74.61	74.72	74.69	74.62	74.85	73.15	76.58	73.23	76.54	73.18	**76.66**
Census	93.89	93.86	93.75	93.52	93.48	93.32	94.29	94.53	94.31	94.56	94.29	**94.58**
Fars_Fatal_Inj_vs_No_Inj	95.07	95.25	94.75	95.01	94.26	94.63	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Média	84.85	84.91	84.61	84.70	84.24	84.40	84.05	85.90	84.04	86.05	84.08	**86.24**

Tabela 34. Comparação número de regras (10^3) no Grupo 4.

	Chi-BigData (CBD)						**RandomFIS (RF)**					
	8		**16**		**32**		**8**		**16**		**32**	
Base de Dados	**CBD-Max**	**CBD-Ave**	**CBD-Max**	**CBD-Ave**	**CBD-Max**	**CBD-Ave**	**RF-Comitê**	**RF-Esp.**	**RF-Comitê**	**RF-Esp.**	**RF-Comitê**	**RF-Esp.**
Kddcup_DOS_vs_normal	0.30	0.30	0.30	0.30	0.30	0.30	0.10	0.04	0.19	0.05	0.40	**0.07**
Poker_0_vs_1	52.80	52.80	53.17	53.17	53.40	53.40	0.62	0.11	1.27	0.10	2.52	**0.10**
Covtype_2_vs_1	7.08	7.08	7.13	7.13	7.21	7.21	0.10	0.03	0.19	0.04	0.38	**0.04**
Census	34.28	34.28	34.34	34.34	34.38	34.38	0.25	0.05	0.51	0.07	1.01	**0.08**
Fars_Fatal_Inj_vs_No_Inj	17.11	17.11	17.16	17.16	17.18	17.18	0.36	0.19	0.73	0.33	1.45	**0.53**
Média	22.31	22.31	22.42	22.42	22.49	22.49	0.29	0.09	0.58	0.12	1.15	**0.16**

5
Conclusões e Trabalhos Futuros

Abordou-se, nesta disertação, o desenvolvimento de um Sistema de Inferência Fuzzy de forma automática, com foco em classificação, para problemas de alta dimensionalidade, denominado RandomFIS. Foram exibidas suas principais características, tais como (i) geração de bases de regras competitivas, sem uso de Algoritmo Evolucionário, reduzindo o impacto em termos de tempo computacional; (ii) capacidade de lidar com problemas de alta dimensionalidade (instância ou atributos), tornando-se um sistema escalável em termos de tempo computacional e uso de memória; (iii) possibilidade de uso de operadores de combinação de classificadores para proporcionar uma melhor acurácia e bases de regras compactas.

Analisou-se a influência de cada componente na acurácia e na compacidade proporcionadas pelo modelo proposto. Foi possível obter resultados superiores, em diferentes benchmarks, aos de alguns sistemas de inferência fuzzy voltados para lidar com problemas de alta dimensionalidade.

Os testes desenvolvidos tiveram como objetivo ilustrar o funcionamento do modelo e evidenciar a sua qualidade. Destaca-se que, apesar de o RamdomFIS ter sido desenvolvido para problemas de alta dimensionalidade, o modelo também pode ser usado, com bastante eficiência, em problemas de média ou baixa dimensionalidade.

O modelo RandomFIS abre diversas frentes para trabalhos futuros. A seguir são sugeridas algumas linhas, de acordo com os diferentes módulos que compõem o modelo: geração dos conjuntos de dados, criação dos classificadores, poda de regras, combinação de classificadores.

- **Técnicas de seleção de variáveis:** as técnicas de re-amostagem empregadas forneceram um bom desempenho em termos de tempo e de memória nas bases de dados analisadas. Entretanto, podem-se buscar identificar as variáveis mais relevantes para criar melhores classificadores.

- **Bases de dados com desbalanceamento de classes:** pode-se realizar um estudo mais aprofundado do desempenho do RandomFIS em bases de dados desbalanceadas e fazer uso de técnicas de criação de dados (dados sintéticos) ou de geração de cópias aleatórias dos dados para atingir um melhor balanceamento entre as classes. Pode-se, também, analisar uma estratégia de treinamento par a par, uma vez que o modelo apresentou melhor desempenho com bases de dados binárias.

- **Métodos de combinação:** durante os experimentos realizados, logrou-se analisar a influência dos métodos de combinação de classificadores do tipo não treinados (seção 2.3). É sempre possível explorar o uso de outras metodologias que permitam encontrar alguma relação entre as saídas fornecidas pelos classificadores (métodos de combinação treinados) para melhorar o desempenho e manter a interpretabilidade.

- **Análise da diversidade:** apesar do bom desempenho do modelo proposto, ele não faz uma análise da diversidade do comitê gerado. Sugere-se, como trabalho futuro, uma análise mais detalhada do comitê obtido pelo RandomFIS para eliminar os classificadores que sejam iguais ou aqueles que apresentem um baixo desempenho em comparação com os demais. Um estratégia deste tipo pode melhorar o desempenho do sistema e fornecer bases de regras com maior compacidade.

- **Arquitetura computacional de alto desempenho:** Atualmente, os diferentes classificadores são treinados de forma sequencial, o que é ineficiente em termos de tempo computacional. A realização do treinamento em uma arquitetura paralela, empregando programação de alto desempenho, certamente melhoraria este aspecto.

Referências bibliográficas

[1] BAITHARU, T. R; PANI, S. K. **A survey on application of machine learning algorithms on data mining**. Int. J. Innovative Technol. Explor. Eng, 3(7):17–20, 2013.

[2] PHYU, T. N. **Survey of classification techniques in data mining.** In: PROCEEDINGS OF THE INTERNATIONAL MULTICONFERENCE OF ENGINEERS AND COMPUTER SCIENTISTS, volumen 1, p. 18–20, 2009.

[3] ALONSO, J. M; MAGDALENA, L ; GONZÁLEZ-RODRÍGUEZ, G. **Looking for a good fuzzy system interpretability index: An experimental approach. International Journal of Approximate Reasoning**, 51(1):115–134, 2009

[4] RIZA, L. S; BERGMEIR, C. N; HERRERA, F ; BENÍTEZ SÁNCHEZ, J. M. **frbs: fuzzy rule-based systems for classification and regression in r**. American Statistical Association, 2015.

[5] ZHANG, Y; WU, X.-B; XING, Z.-Y ; HU, W.-L. **On generating interpretable and precise fuzzy systems based on pareto multiobjective cooperative co-evolutionary algorithm**. Applied Soft Computing, 11(1):1284–1294, 2011.

[6] ZHOU, S.-M; LYONS, R. A; BROPHY, S ; GRAVENOR, M. B. **Constructing compact takagi-sugeno rule systems: identification of complex interactions in epidemiological data**. PloS one, 7(12):e51468, 2012.

[7] ISHIBUCHI, H; NAKASHIMA, T ; NII, M. **Classification and modeling with linguistic information granules: Advanced approaches to linguistic Data Mining**. Springer Science & Business Media, 2006.

[8] BABUŠKA, R. **Fuzzy modeling for control**, volumen 12. Springer Science & Business Media, 2012.

[9] FAZZOLARI, M; ALCALA, R; NOJIMA, Y; ISHIBUCHI, H ; HERRERA, F. **A review of the application of multiobjective**

evolutionary fuzzy systems: Current status and further directions. IEEE Transactions on Fuzzy systems, 21(1):45–65, 2013.

[10] ISHIBUCHI, H; YAMANE, M ; NOJIMA, Y. **Rule weight update in parallel distributed fuzzy genetics-based machine learning with data rotation.** In: FUZZY SYSTEMS (FUZZ), 2013 IEEE INTERNATIONAL CONFERENCE ON, p. 1–8. IEEE, 2013.

[11] HERRERA, F. **Genetic fuzzy systems: taxonomy, current research trends and prospects.** Evolutionary Intelligence, 1(1):27–46, 2008.

[12] ANGELOV, P.; ZHOU, X.. **Evolving fuzzy-rule-based classifiers from data streams.** IEEE Transactions on Fuzzy Systems, 16(6):1462–1475, 2008.

[13] LEMOS, A; CAMINHAS, W ; GOMIDE, F. **Evolving intelligent systems: Methods, algorithms and applications.** In: EMERGING PARADIGMS IN MACHINE LEARNING, p. 117–159. Springer, 2013.

[14] PAREDES, J; TANSCHEIT, R; VELLASCO, M ; KOSHIYAMA, **A. Automatic synthesis of fuzzy inference systems for classification.** In: INTERNATIONAL CONFERENCE ON INFORMATION PROCESSING AND MANAGEMENT OF UNCERTAINTY IN KNOWLEDGE-BASED SYSTEMS, p. 486–497. Springer, 2016.

[15] CANUL-REICH, J; SHOEMAKER, L ; HALL, L. O. **Ensembles of fuzzy classifiers.** dermatology, 6(34):366, 2007.

[16] CORDÓN, O; QUIRIN, A ; SÁNCHEZ, L. **A first study on bagging fuzzy rule-based classification systems with multicriteria genetic selection of the component classifiers.** In: GENETIC AND EVOLVING SYSTEMS, 2008. GEFS 2008. 3RD INTERNATIONAL WORKSHOP ON, p. 11–16. IEEE, 2008.

[17] NOJIMA, Y; ISHIBUCHI, H. **Genetic rule selection with a multiclassifier coding scheme for ensemble classifier design.** International Journal of Hybrid Intelligent Systems, 4(3):157–169, 2007.

[18] KUNCHEVA, L. I. **Combining pattern classifiers: methods and algorithms.** John Wiley & Sons, 2004.

[19] BREIMAN, L. **Bagging predictors.** Machine learning, 24(2):123–140, 1996.

[20] HO, T. K. **The random subspace method for constructing decision**

forests. IEEE transactions on pattern analysis and machine intelligence, 20(8):832–844, 1998.

[21] KLEINER, A; TALWALKAR, A; SARKAR, P ; JORDAN, M. I. **A scalable bootstrap for massive data.** Journal of the Royal Statistical Society: Series B (Statistical Methodology), 76(4):795–816, 2014.

[22] KLEINER, A; TALWALKAR, A; SARKAR, P ; JORDAN, M. **The big data bootstrap**. arXiv preprint arXiv:1206.6415, 2012.

[23] HANSEN, L. K; SALAMON, P. **Neural network ensembles.** IEEE transactions on pattern analysis and machine intelligence, 12:993–1001, 1990.

[24] POLIKAR, R. **Ensemble based systems in decision making.** IEEE Circuits and systems magazine, 6(3):21–45, 2006.

[25] PERRONE, M. P; COOPER, L. N. **When networks disagree: Ensemble methods for hybrid neural networks.** Technical report, DTIC Document, 1992.

[26] BROWN, G; WYATT, J; HARRIS, R ; YAO, X. **Diversity creation methods: a survey and categorisation.** Information Fusion, 6(1):5– 20, 2005.

[27] MARDIA, K. V; KENT, J. T ; BIBBY, J. M. **Multivariate analysis.** 1980.

[28] SHARKEY, A. J; SHARKEY, N. E; GERECKE, U ; CHANDROTH, G. O. **The "test and select" approach to ensemble combination.** In: INTERNATIONAL WORKSHOP ON MULTIPLE CLASSIFIER SYSTEMS, p. 30–44. Springer, 2000.

[29] EFRON, B; TIBSHIRANI, R. J. **An introduction to the bootstrap.** CRC press, 1994.

[30] BAUER, E; KOHAVI, R. **An empirical comparison of voting classification algorithms: Bagging, boosting, and variants.** Machine learning, 36(1-2):105–139, 1999.

[31] POLITIS, D. N.; ROMANO, J. P.; WOLF, M. **Subsampling**. New York, NY: Springer New York, 1999.

[32] BICKEL, P. J; GÖTZE, F ; VAN ZWET, W. R. **Resampling fewer than n observations: gains, losses, and remedies for losses.** In: SELECTED WORKS OF WILLEM VAN ZWET, p. 267–297. Springer, 2012.

[33] SCHAPIRE, R. E. **The strength of weak learnability.** Machine learning,

5(2):197–227, 1990.

[34] FREUND, Y; SCHAPIRE, R. E ; OTHERS. **Experiments with a new boosting algorithm.** In: ICML, volumen 96, p. 148–156, 1996.

[35] COELHO, A. L. V. **Evolução, simbiose e hidridismo aplicados a engenharia de sistemas inteligentes modulares: investigações em redes neurais, comites de maquinas e sistemas multiagentes.** 2004.

[36] FREUND, Y; SCHAPIRE, R ; ABE, N. **A short introduction to boosting.** Journal-Japanese Society For Artificial Intelligence, 14(771- 780):1612, 1999.

[37] HAYKIN, S. S. **Redes neurais.** Bookman, 2001.

[38] PAPPA, G. L. **Seleção de atributos utilizando Algoritmos Genéticos multiobjetivos.** PhD thesis, Pontifícia Universidade Católica do Paraná, 2002.

[39] JAIN, A. K; DUIN, R. P. W ; MAO, J. **Statistical pattern recognition: A review.** IEEE Transactions on pattern analysis and machine intelligence, 22(1):4–37, 2000.

[40] XU, L; KRZYZAK, A ; SUEN, C. Y. **Methods of combining multiple classifiers and their applications to handwriting recognition.** IEEE transactions on systems, man, and cybernetics, 22(3):418–435, 1992.

[41] BERLANGA, F. J; RIVERA, A; DEL JESÚS, M. J ; HERRERA, F. **Gp-coach: Genetic programming-based learning of compact and accurate fuzzy rule-based classification systems for highdimensional problems.** Information Sciences, 180(8):1183–1200, 2010.

[42] KOSHIYAMA, A. S; VELLASCO, M. M ; TANSCHEIT, R. **Gpfis-class: a genetic fuzzy system based on genetic programming for classification problems.** Applied Soft Computing, 37:561–571, 2015.

[43] KLEMENT, E. P; MESIAR, R ; PAP, E. **Triangular norms**, 2000.

[44] FERNÁNDEZ, A; CALDERÓN, M; BARRENECHEA, E; BUSTINCE, H ; HERRERA, F. **Solving multi-class problems with linguistic fuzzy rule based classification systems based on pairwise learning and preference relations.** Fuzzy sets and systems, 161(23):3064–3080, 2010.

[45] KUNCHEVA, L. I; JAIN, L. C. **Designing classifier fusion systems by genetic algorithms.** IEEE Transactions on Evolutionary computation, 4(4):327–336, 2000.

[46] KOSHIYAMA, A. S.; VELLASCO, M. M. B. R.; TANSCHEIT, R. **GPFIS-CLASS: A Genetic Fuzzy System based on Genetic Programming for classification problems.** Applied Soft Computing, 37:561–571, 2015.

[47] DERRAC, J; GARCÍA, S; MOLINA, D ; HERRERA, F. **A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms.** Swarm and Evolutionary Computation, 1(1):3–18, 2011.

[48] ALCALÁ-FDEZ, J; SANCHEZ, L; GARCIA, S; DEL JESUS, M. J; VENTURA, S; GARRELL, J. M; OTERO, J; ROMERO, C; BACARDIT, J; RIVAS, V. M ; OTHERS. **Keel: a software tool to assess evolutionary algorithms for data mining problems.** Soft Computing, 13(3):307–318, 2009.

[49] ASUNCION, A; NEWMAN, D. **Uci machine learning repository,** 2007.

[50] DEL RÍO, S; LÓPEZ, V; BENÍTEZ, J. M ; HERRERA, F. **A mapreduce approach to address big data classification problems based on the fusion of linguistic fuzzy rules.** International Journal of Computational Intelligence Systems, 8(3):422–437, 2015.

[51] TRAWIŃSKI, K; CORDÓN, O ; QUIRIN, A. **On designing fuzzy rulebased multiclassification systems by combining furia with bagging and feature selection.** International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 19(04):589–633, 2011.

[52] CORDÓN, O; QUIRIN, A. **Comparing two genetic overproduce-andchoose strategies for fuzzy rule-based multiclassification systems generated by bagging and mutual information-based feature selection.** International Journal of Hybrid Intelligent Systems, 7(1):45–64, 2010.

[53] DIETTERICH, T. G. **Approximate statistical tests for comparing supervised classification learning algorithms.** Neural computation, 10(7):1895–1923, 1998.

[54] ALCALA-FDEZ, J; ALCALA, R ; HERRERA, F. **A fuzzy association rule-based classification model for high-dimensional problems with genetic rule selection and lateral tuning.** IEEE Transactions on Fuzzy Systems, 19(5):857–872, 2011.

[55] HÜHN, J; HÜLLERMEIER, E. **Furia: an algorithm for unordered fuzzy rule induction.** Data Mining and Knowledge Discovery, 19(3):293– 319,

2009.

[56] LÓPEZ, V; DEL RIO, S; BENITEZ, J. M ; HERRERA, F. **On the use of mapreduce to build linguistic fuzzy rule based classification systems for big data.** In: 2014 IEEE INTERNATIONAL CONFERENCE ON FUZZY SYSTEMS (FUZZ-IEEE), p. 1905–1912. IEEE, 2014.

[57] CHI, Z; YAN, H ; PHAM, T. **Fuzzy algorithms: with applications to image processing and pattern recognition,** volumen 10. World Scientific, 1996.

[58] DEAN, J; GHEMAWAT, S. **Mapreduce: simplified data processing on large clusters.** Communications of the ACM, 51(1):107–113, 2008.