BRUNO BUTILHÃO CHAVES

ESTUDO DO ALGORITMO ADABOOST DE APRENDIZAGEM DE MÁQUINA APLICADO A SENSORES E SISTEMAS EMBARCADOS

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Mestre em Engenharia Mecânica.

SÃO PAULO

2012

BRUNO BUTILHÃO CHAVES

ESTUDO DO ALGORITMO ADABOOST DE APRENDIZAGEM DE MÁQUINA APLICADO A SENSORES E SISTEMAS EMBARCADOS

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Mestre em Engenharia Mecânica.

Área de Concentração:

Engenharia de Controle e Automação Mecânica

Orientador:

Prof. Dr. Delson Torikai

SÃO PAULO

2012

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, de fevereiro de 2012.

Assinatura do autor _____

Assinatura do orientador _____

FICHA CATALOGRÁFICA

Chaves, Bruno Butilhão
Estudo do algoritmo Adaboost de aprendizagem de máquina aplicado a sensores e sistemas embarcados / B.B. Chaves. – ed.rev. -- São Paulo, 2012.
119 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos.

1.Reconhecimento de padrões 2.Aprendizado de máquina 3.Sistemas embutidos 4.Sensor 5.Inteligência artificial 6.Aprendizado computacional 7.Combustíveis veiculares I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos II.t.

Dedico este trabalho aos meus pais, Jorge e Vanda, e à minha irmã, Dani, por todo apoio, compreensão em minhas escolhas e incentivo em todos os projetos de minha vida.

AGRADECIMENTOS

Muitas foram as pessoas que estiveram comigo e contribuíram de forma direta ou indireta para a realização deste trabalho. A todas elas, muito obrigado.

Obrigado ao Prof. Dr. Delson Torikai, por toda dedicação e comprometimento como orientador, sempre disposto a ajudar, e também pela confiança e amizade que foram essenciais para que este trabalho fosse concluído com êxito.

Agradeço à Massey University da Nova Zelândia por me proporcionar a oportunidade de desenvolver parte dessa pesquisa em sua universidade. Agradeço especialmente aos Professores Dr. André Barczak e Dr. Napoleon Reyes, e também ao doutorando Teo Susnjak por toda a orientação, aprendizado e companheirismo durante os meses de convívio em seu país.

Obrigado ao Prof. Dr. Ricardo Ibrahim pela ajuda dada para a realização deste trabalho.

Agradeço também aos meus colegas de laboratório, Lucas Mendonça e Bruno Barazani, pelas valiosas ajudas durante as etapas experimentais e pelos conhecimentos compartilhados.

Aos meus pais e irmã pela educação, carinho e apoio de sempre.

À Cecilia pelo carinho, incentivo e compreensão em todos os momentos.

Aos meus grandes amigos com quem sempre posso contar.

Por fim, ao CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) pelo apoio financeiro deste trabalho, por meio da bolsa de mestrado concedida.

RESUMO

O estudo da Inteligência Artificial e de suas técnicas tem trazido grandes resultados para a evolução da tecnologia em diversas áreas. Técnicas já conhecidas como as Redes Neurais e Árvores de Decisão vêm sendo aprimoradas por técnicas de *Boosting* como o *Adaptive Boosting*. Esta técnica é uma das que apresenta maior perspectiva de crescimento devido a seu potencial, flexibilidade e simplicidade para ser implementada em diferentes cenários, como por exemplo, no tratamento de imagens para reconhecimento de padrões.

Um mercado com grande potencial para se beneficiar da técnica de *Boosting*, e em especial do AdaBoost, é o mercado de sensores. É cada vez mais comum a utilização de sensores isolados ou sistemas de múltiplos sensores trabalhando concomitantemente para se atingir um objetivo comum. Na utilização de sistemas embarcados compostos por sensores para realização de análises e tomadas de decisão são cada vez mais requisitados, principalmente onde se requer algum tipo de reconhecimento de padrão. O objetivo desta dissertação é estudar e desenvolver o conhecimento do algoritmo AdaBoost para aplicação em sensores, de forma a aprimorar a sensibilidade e precisão das medições, tanto de sensores isolados como de sistemas complexos com vários sensores, sem que seja necessário realizar modificações no próprio sensor. O estudo estende-se também em como implementar o algoritmo inteligente a um dispositivo autônomo composto por sensores e um microprocessador que contenha um classificador embarcado de reconhecimento de padrões. Para demonstrar a utilidade da técnica, foi realizado um estudo de caso utilizando um sistema composto de sensores capacitivos interdigitalizados e microfabricados, sensores de temperatura e sensor a fibra óptica, para verificar adulterações em combustíveis automotivos, em especial, do etanol combustível. Sete experimentos são apresentados no trabalho. Índices acima de 90% de classificações corretas foram obtidos, indicando a viabilidade da utilização do algoritmo para calibração de sensores ou rede de sensores. Por fim, foi desenvolvida com sucesso uma forma de embarcar o classificador treinado em um microprocessador, confirmando assim ser possível desenvolver dispositivos embarcados contendo essa tecnologia.

Palavras-chave: AdaBoost, *Boosting*, Aprendizagem de Máquina, Reconhecimento de Padrão, Dispositivos Embarcados, Sensores, Adulteração de Combustível.

ABSTRACT

Studies on Artificial Intelligence and its techniques have provided great results for the whole technology evolution in several areas. Techniques known as Neural Networks and Decision Trees have been improved by Boosting techniques such as Adaptive Boosting. This particular technique presents great growth prospects due to its potential, flexibility and simplicity to be implemented in different scenarios, such as image analysis for pattern recognition.

A specific market that can greatly benefit from the technique of Boosting and particularly AdaBoost is the sensor market. The use of isolated sensors or multiple sensor systems working together in order to reach a common goal is increasingly common. Embedded systems consisting of sensors for analysis and decision-making are also increasingly common especially in cases in which some sort of pattern recognition is necessary. Therefore, the purpose of this thesis is to study and to develop some knowledge about the AdaBoost algorithm applied to sensors in order to improve the sensitivity and accuracy of its measurements, both in isolated sensors and in complex systems with multiple sensors, without requiring any change in the sensor itself. The study also approaches how to implement the intelligent algorithm in an autonomous device composed by sensors and a microprocessor that contains an embedded classifier for pattern recognition. Accordingly, a case study was conducted using a system composed of microfabricated capacitive sensors, temperature sensors and fiber optical sensor with the purpose of analyzing the amount of automobile fuels, especially ethanol fuel. Seven experiments were performed in order to demonstrate the usefulness of this technique and they are presented in the study. Rates above 90% of correct classifications were obtained, which indicates the feasibility of using the algorithm for sensor calibration or sensor network calibration. Finally, a way to embed a trained classifier into a microprocessor was successfully developed, confirming that it is possible to develop embedded devices containing this technology.

Keywords: AdaBoost, Boosting, Machine Learning, Pattern Recognition, Embedded Systems, Sensors, Fuel Adulteration

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Símbolos

D_t	- Distribuição de pesos atribuída aos exemplos de treinamento
t	- Ciclo de aprendizagem que o algoritmo se encontra
T	- Número total de ciclos que o algoritmo será iterado
h_t	- Hipóteses geradas a cada iteração t
e_t	- Erro cometido durante a iteração
γ_t	- Importância associada do classificador
E_0	- Campo elétrico entre os eletrodos de um capacitor
E	- Campo elétrico resultante
Q	- Carga na superfície das placas do capacitor
ε_o	- Permissividade no vácuo
ε	- Permissividade elétrica de um material
V	- Tensão elétrica aplicada entre as placas do capacitor
P	- Conjunto de exemplos positivos
N	- Conjunto de exemplos negativos
TP	- Conjunto de exemplos positivos verdadeiros
FP	- Conjunto de exemplos positivos falsos
TN	- Conjunto de exemplos negativos verdadeiros
FN	- Conjunto de exemplos negativos falsos

Lista de Abreviaturas

AdaBoost	Algoritmo *Adaptive Boosting*
AM	Aprendizagem de Máquina
ºINPM	Porcentagem de álcool em peso ou grau alcoólico
AEHC	Álcool Etílico Hidratado Combustível
AEAC	Álcool Etílico Anidro Combustível
MEMS	Sistemas Micro-Eletro-Mecânicos (Micro-Electro-Mechanical Systems)
NEMS	Sistemas Nano-Electro-Mecânicos (Nano-Electro-Mechanical Systems)
IA	Inteligência Artificial

1 Introdução

O presente trabalho visa o desenvolvimento de conhecimento do potencial de algoritmos inteligentes como o AdaBoost para aplicações no campo de sensores. Dentre os diversos benefícios que este estudo pretende promover, pode-se destacar a possibilidade de projetar dispositivos embarcados que tenham a tecnologia de classificação de dados e que poderão ser aproveitados nas mais diversas áreas de conhecimento.

Nos últimos anos o mercado de sensores tem apresentado um expressivo crescimento no âmbito global (YURISH; KIRIANAKI; MYSHKIN, 2005; SALOMON, 2006; MENDEZ, 2007; SON et al., 2010). A forte inclinação na curva de demanda por esses dispositivos é influenciada diretamente por indústrias de diversos ramos de atividade, como automobilística, telefonia móvel, petroquímica, aeronáutica, espacial, médica, ou mesmo do entretenimento interativo.

Vazamento de gás com detecção remota em dutos, monitoramento da poluição das indústrias, controles de combustão em tempo real em veículos automotivos, controles de processos industriais, diagnósticos médicos, detecção de armas químicas, monitoração e detecção de possíveis falhas no processo de extração de petróleo. Esses são alguns exemplos de atividades que podem ser muito beneficiadas com o aprimoramento de sensores ou sistemas de sensores utilizados.

Há atualmente um grande número de sensores disponíveis no mercado, utilizando diferentes princípios físicos, onde vantagens e desvantagens devem ser analisadas caso a caso para uma determinada aplicação, e deve ser analisado se o uso de um sensor ou um conjunto de sensores seria o mais adequado. Dentre as características que devem ser avaliadas pode-se destacar a precisão e resolução, concisão e facilidade na interpretação dos dados, faixa de operação ou flexibilidade, simplicidade ou facilidade de operação e manutenção, redundância, consumo de energia, dimensões, calibração, confiabilidade e custo (FACELI, 2001; GROOVER AT. AL., 1989; EVERETT, 1995).

Devido à constante busca por desenvolvimento de produtos e processos de ponta e de alto desempenho ao menor custo possível apresentada pelos diversos setores industriais, o mercado de sensores tem apresentado inovações que estão cada vez mais presentes no setor industrial, como componentes microeletromecânicos *MEMS*, nanoeletromecânicos *NEMS* e sensores *plug-and-play* (RAY, 2005). Outra inovação tecnológica que se destaca são os sensores inteligentes (*smart sensors*) que utilizam microcontroladores com o objetivo de

acrescentar funcionalidades, inteligência e melhor desempenho às mais variadas tecnologias de sensores, além de fornecer mais simplicidade, flexibilidade e apresentarem de certa forma manipulação mais intuitiva (YURISH; KIRIANAKI; MYSHKIN, 2005).

Independentemente do tipo de sensor ou da finalidade de utilização é cada vez maior a exigência e a necessidade de se obter sinais de resposta com maior precisão, confiabilidade e rapidez. Além disso, a possibilidade de se classificar dados de leitura de sensores de acordo com padrões de diferentes classes e utilizá-los para tomadas de decisão é um grande diferencial e tem se apresentado de grande importância e utilidade para as mais diversas aplicações, como por exemplo, a classificação de sinais de imagem de acordo com formatos e cores, análises biomédicas sobre DNA e particularidades encontradas em amostras de sangue, ou mesmo a classificação de produtos químicos pela concentração de determinadas substâncias. Esses exemplos fortalecem a crescente utilização dos sensores inteligentes ou *smart sensors*.

Podem-se definir tais classificações de padrões como sendo o ato de realizar uma ação baseada na categoria de padrões retirados de dados não trabalhados previamente (DUDA; HART; STORK, 2001). Essa classificação de padrões está inserida no contexto de aprendizado de máquina, que é uma área estudada na inteligência computacional e é focada em pesquisas de técnicas capazes de extrair conceitos a partir de amostra de dados (MITCHELL,1997). Dentre os vários métodos de aprendizado de máquina, um método que vem sendo cada vez mais utilizado para se definir as características do classificador é a utilização de amostras de padrões para ensinar e aprimorar de forma contínua o classificador (DUDA; HART; STORK, 2001).

Esse desafio de classificação de padrões é estudado já há algumas décadas e existem diversas técnicas já amplamente divulgadas, como Redes Neurais, Árvores de decisão e Máquinas de Vetores de Suporte (SVM). No entanto, muitos estudos nessa área de conhecimento têm sido publicados e novos algoritmos e técnicas têm sido apresentados.

Uma alternativa que vem se mostrando viável e promissora em especial para ser utilizada junto a sensores é o algoritmo denominado *Adaptive Boosting*. Mais popularmente conhecido por AdaBoost, o algoritmo se destaca principalmente devido ao seu potencial, flexibilidade e simplicidade para ser implementado em diferentes cenários (SCHAPIRE; SINGER, 1998; SCHAPIRE, 2000), o que é de grande valia para as aplicações em sensores. Essas características são essenciais para se projetar mecanismos autônomos e que cada vez mais são utilizados nas mais diversas áreas de conhecimento, pois permitem que a análise seja embarcada em dispositivos.

Estudos sobre o algoritmo têm sido publicados sem diferentes contextos e bons resultados têm sido atingidos (SUSNJAK, 2009; LESHEM, 2005; LESHEM; RITOV, 2007; VIOLA; JONES, 2004; CHEN; NICPONSKI; RAY, 2004; CREAMER; FREUND, 2005; WANG; CHAKRABORTY; COMANICIU, 2006; LESLIE et al., 2004; NABATAME; IBA, 2006; TRIPP; HUNG; PONTIKAKIS, 2006; ECK et al., 2007).

Apesar das aplicações já publicadas, o AdaBoost ainda pode ser explorado em muitos outros cenários, entre eles os relacionados a sensores. A grande automatização de ferramentas e máquinas, aumento da produção de equipamentos embarcados e o desenvolvimento de novos produtos que trabalham com multi-sensores nas mais diversas áreas e mercados tornam os estudos do AdaBoost aplicados a sensores muito relevantes.

Exemplos de possíveis aplicações desse algoritmo são o reconhecimento de tipos e procedências de combustíveis, melhoria no desempenho de automóveis e diminuição de poluentes, desenvolvimento e aprimoramento de produtos como a língua eletrônica, aplicações em sistemas embarcados que operam no fundo dos oceanos, entre outros.

Portanto, independentemente do tipo de sensor ou da finalidade de utilização, a possibilidade de se classificar dados de leitura de sensores de acordo com padrões de diferentes classes e utilizá-los para tomadas de decisão é um grande diferencial e o AdaBoost vem demonstrando ter potencial para auxiliar nesse desafio.

1.1 Justificativa

A principal motivação desse projeto é a grande variedade de aplicações em que podem ser utilizados os sistemas embarcados compostos por sensores e um algoritmo de aprendizado de máquina que permita que decisões, classificações ou ações sejam realizadas de forma autônoma sem que seja necessário grande aparato eletrônico.

Outro fator importante é que os sensores fabricados por tecnologias de miniaturização, como por exemplo, os MEMS e NEMS, necessitam muitas vezes de calibrações individuais, o que requer muito tempo. Um algoritmo de aprendizado de máquina como o AdaBoost apresenta-se como uma alternativa para acelerar essa etapa de calibração, pois com poucos dados iniciais é possível calibrar o sensor, não sendo necessário construir curvas de calibração com muitos dados experimentais.

Em muitos cenários ocorre sobreposição de informações e algumas propriedades influenciam outras características do sistema analisado. Assim, muitas vezes um único sensor não é capaz de fazer uma análise real do sistema, fazendo-se necessária a utilização de um

sistema de sensores. Nesse caso, um algoritmo como o AdaBoost auxilia na análise de todas as informações dos diversos sensores e propriedades do sistema, sendo capaz de fornecer uma resposta final mais precisa.

Por fim, o algoritmo AdaBoost foi escolhido para essa pesquisa devido ao seu destaque perante os demais algoritmos se comparado o seu baixo custo computacional, característica importante para o desenvolvimento de dispositivos embarcados, além de ser flexível para ser utilizado em diferentes aplicações e também junto a algoritmos base distintos.

1.2 Objetivo

O presente trabalho tem como objetivo principal desenvolver o conhecimento sobre possíveis aplicações do algoritmo AdaBoost em sensores em geral, de forma a aprimorar a sensibilidade e precisão das medições, tanto de sensores isolados como de sistemas complexos com vários sensores, significativamente sem que seja necessário realizar modificações no próprio sensor. Almeja-se também obter benefícios da utilização dessa técnica durante a etapa de calibração dos sensores, gerando conhecimento para acelerar essa etapa quando sensores forem fabricados em escala industrial. Por fim, tem-se também como objetivo o desenvolvimento de um dispositivo autônomo e que tenha a tecnologia de classificação de dados utilizando o algoritmo AdaBoost embarcado.

O desenvolvimento desse tipo de trabalho envolve problemas de computação, eletrônica e princípios básicos sobre o funcionamento de sensores além de verificações experimentais. A etapa experimental será focada na detecção de adulteração de combustíveis automotivos, utilizando sensores de diferentes princípios, como o sensor capacitivo e sensor de fibra óptica. Dessa forma, pretende-se reconhecer e classificar diferentes tipos de combustíveis ou misturas de combustíveis, verificando assim concentrações, procedência e particularidades de forma mais precisa e rápida. Assim, o experimento viabilizará a análise confiável e em tempo real sem que seja necessário um grande acréscimo de equipamento eletrônico para apuração e análise dos dados adquiridos.

1.3 Organização da Dissertação

A organização deste trabalho é realizada conforme descrito a seguir.

O Capítulo 2 aborda os principais tópicos da Aprendizagem de Máquina, abrangendo uma breve introdução sobre o tema, classificações das diferentes técnicas existentes, além de discursar sobre algumas técnicas de combinação de classificadores.

O Capítulo 3 reúne a compilação de informações sobre o algoritmo *Adaptive Boosting*, abordando o funcionamento do algoritmo (item 3.2) e um exemplo didático para exemplificar o seu funcionamento (item 3.3). Em seguida são apresentadas estruturas do algoritmo para treinamento de classificadores (item 3.4), é levantada uma relação não exaustiva sobre as variações do AdaBoost (item 3.5) e também sobre aplicações já publicadas sobre o algoritmo (item 3.6).

O Capítulo 4 fornece uma visão geral sobre sensores, mostrando suas principais classificações e aplicações, bem como um detalhamento maior sobre os sensores capacitivos e também os de fibra óptica (item 4.2), além de discutir sobre os benefícios de sua aplicação em sensores (item 4.4). O Capítulo 5 apresenta o caso experimental estudado incluindo informações sobre o software utilizado para realizar as simulações e análises dos dados. Em seguida, o Capítulo 6 apresenta os sete casos experimentais estudados de forma detalhada.

O Capítulo 7 aborda a etapa de embarcação do classificador treinado em um microprocessador para permitir assim o desenvolvimento de dispositivos autônomos e que contenham a tecnologia de reconhecimento de padrões.

Finalmente, o Capítulo 8 apresenta as discussões sobre os resultados, conclusões e também sugestões para as próximas etapas da pesquisa.

2 Aprendizagem de Máquina

2.1 Visão Geral

Inteligência Artificial, ou simplesmente AI, teve origem na área de estudo computacional e foi introduzida no meio acadêmico por volta dos anos 1950 (BABENKO; MARMANIS, 2009). De acordo com pesquisas sobre o tema (RUSSEL; NORVIG, 2002; BUCHANAN, 2005) os objetivos delineados no início para esse novo campo da pesquisa eram extremamente ambiciosos, almejando desenvolver e construir máquinas e robôs que fossem capazes de pensar de forma independente, assim como os humanos. No entanto, com o desenvolvimento de novas pesquisas e projetos, novos objetivos mais modestos e práticos, porém não menos inovadores e com grande potencial de implementação, foram traçados. Devido a essa mudança de metas, fez-se necessária a criação de sub-áreas de pesquisa dentro da Inteligência Artificial, dentre elas a Mineração de Dados e a Aprendizagem de Máquina.

Mineração de Dados tem como objetivo principal a extração de informações implícitas, previamente desconhecidas e potencialmente úteis e proveitosas a partir de uma base de dados. A ideia principal é desenvolver programas de computador que sejam capazes de garimpar a base de dados fornecida de forma automática em busca de regularidades ou padrões. Por outro lado, a Aprendizagem de Máquina é responsável por fornecer a base ou suporte técnico para que a Mineração de Dados atinja seu objetivo. Por meio de algoritmos, a Aprendizagem de Máquina é utilizada para viabilizar a extração de informações das bases de dados para que sejam utilizadas para finalidades específicas. Essas duas sub-áreas da Inteligência Artificial serão abordadas a seguir (Seções 2.2 e 2.3). O capítulo também contém uma série de definições utilizadas nesses campos de estudo (Seção 2.4), além de abordar alguns algoritmos comumente utilizados e também as técnicas de classificação de classificadores conhecidas por *Boosting* e *Bagging* (Seções 2.5 e 2.6).

2.2 Mineração de Dados

Mineração de Dados pode ser definida como o processo de investigação e descoberta de padrões em elementos presentes em uma base de dados (WITTEN; FRANK, 2005). Outra possível definição desse campo de pesquisa é fornecida por (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996a) que afirma que extração de conhecimento da base de dados é o

processo de identificação de padrões válidos, novos, potencialmente úteis e compreensíveis embutidos nos dados.

Essa base de dados é armazenada eletronicamente e o processo de pesquisa é realizado de forma automática ou semi-automática pelo computador. Não é novidade que bases de dados têm informações valiosas, como tendências e padrões, e que podem ser utilizadas para tomadas de decisão em diferentes cenários, porém com o desenvolvimento acentuado da eletrônica nas últimas décadas, tornou-se viável o melhor aproveitamento dessas informações, acarretando no crescimento desse campo de pesquisa de Mineração de Dados.

Teorias e pesquisas sugerem diferentes abordagens para organizar o processo de extrair informações de bases de dados (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996b; WEISS; INDURKHYA, 1998). No entanto, (REZENDE, 2003) sugere uma organização composta por três etapas, como ilustrado abaixo.

Figura 2.1 - Etapas do processo de Mineração de Dados

2.2.1 Pré-processamento

Na grande maioria dos casos os dados a serem trabalhados não se encontram padronizados, em formato adequado ou em um único arquivo, podendo estar contidos parcialmente em planilhas e em arquivos de texto, por exemplo. Assim, faz-se necessária a unificação e padronização dos dados em uma única base de dados. É importante que os dados estejam em formato adequado para ser compatível com o algoritmo de extração de padrões escolhido para ser utilizado.

Nessa etapa também é interessante efetuar a limpeza dos dados, com o objetivo de corrigir eventuais problemas advindos do processo de coleta, minimizar tempo de treinamento do algoritmo, reduzir custo computacional, ou ainda como estratégia para remoção de ruído e tratamento de conjunto de exemplos não balanceados. As possíveis reduções de dados podem ser realizadas por meio da diminuição do número de exemplos, número de atributos ou ainda do número de valores possíveis de um atributo.

Nessa etapa deve-se priorizar um princípio básico da mineração de dados de que a qualidade dos dados é crítica e de muito mais importância do que a quantidade. Por isso, nessa etapa é necessário obedecer cinco características durante a captação e manutenção dos dados (BRAGA, 2005):

- Acurácia: é necessário eliminar ou ao menos minimizar erros de medição e digitação;
- Consistência: os dados presentes na base de dados e fornecidos ao algoritmo devem ser coerentes e fazer sentido no contexto abordado;
- Completude: campos de informação faltantes ou informações incompletas prejudicam ou até inviabilizam os resultados;
- Relevância: deve-se buscar compor a base de dados somente com informações concernentes ao problema estudado;
- Não-redundância: duplicações de informação não agregam valor, além de prejudicar o desempenho do algoritmo.

A etapa de pré-processamento é realizada necessariamente no início do processo de Mineração de Dados. No entanto, por esse ser um processo iterativo, algumas das atividades dessa etapa descritas acima podem ser realizadas após a etapa de extração de padrões.

2.2.2 Extração de padrões

A Aprendizagem de Máquina é essencial nessa etapa do processo. Nessa etapa é realizada a escolha, configuração e execução de um ou mais algoritmos para analisar e extrair o conhecimento diluído na base de dados.

Primeiramente é necessário definir qual a tarefa a ser realizada de forma a atingir os objetivos desejáveis para o problema abordado. Segundo (REZENDE, 2003), essas tarefas podem ser organizadas como descrito a seguir.

Atividades preditivas são aquelas capazes de classificar um novo exemplo em um dos possíveis grupos a partir de experiências e conhecimentos anteriores. Essas atividades podem ainda ser subdivididas em classificação, que trabalham com dados discretos, e em regressão, que envolve atributos com valores contínuos. Esse trabalho, por trabalhar com dados obtidos por sensores, tem como foco os problemas de classificação.

Por sua vez, as atividades descritivas são aquelas em que se almeja identificar comportamentos dos dados sem que haja classes pré-definidas.

Por fim, é necessário escolher um algoritmo de Aprendizado de Máquina que satisfaça as necessidades do problema. Existe uma série de algoritmos disponíveis, além de técnicas de combinação de algoritmos que serão discutidas no próximo capítulo desse trabalho.

Novamente, devido à iteratividade do processo, pode ser necessário executar o algoritmo ou conjunto de algoritmos diversas vezes objetivando, por exemplo, a melhoria da precisão ou refino dos padrões extraídos.

2.2.3 Pós-processamento

A última etapa do processo é o pós-processamento. Essa fase final é direcionada principalmente para analisar os padrões adquiridos visando avaliar se os resultados satisfazem os objetivos inicialmente propostos. Normalmente esses resultados são utilizados por humanos ou mesmo por máquinas para tomadas de decisão e, por isso, essa avaliação dos resultados torna-se extremamente relevante.

Nem sempre todos os padrões ou respostas encontradas pelo algoritmo satisfazem e realmente fazem sentido para o problema específico abordado. Segundo (SILBERSCHATZ; TUZHILIN, 1995) é de vital importância desenvolver algumas técnicas de apoio com o objetivo de fornecer aos usuários finais apenas os padrões mais interessantes. Por isso, em alguns casos é necessário realizar alguns ajustes e retornar à etapa anterior para refinar os resultados.

2.3 Conceitos da Aprendizagem de Máquina

As classificações de padrões podem ser definidas como sendo o ato de realizar uma ação baseada na categoria de padrões retirados de dados não trabalhados previamente (DUDA; HART; STORK, 1997). Essa classificação de padrões está inserida no contexto de aprendizado de máquina, que é uma área estudada na inteligência computacional e é focada em pesquisas de técnicas capazes de extrair conceitos a partir de amostra de dados (MITCHELL, 1997). Em outras palavras, pode-se afirmar que essas técnicas são programas de computador que permitem que máquinas tomem decisões baseadas em informações e experiências acumuladas e armazenadas em bases de dados a partir de soluções de problemas anteriores. Portanto, o objetivo dos algoritmos de AM é aprender como discriminar e

classificar exemplos pertencentes a diferentes grupos por meio da identificação de aspectos e características comuns dentre os exemplos presentes na base de dados.

Dentre os vários métodos de aprendizado de máquina, o que vem sendo cada vez mais utilizado para se definir as características do classificador é a utilização de amostras de padrões para ensinar e aprimorar de forma contínua o classificador (DUDA; HART; STORK, 1997).

As diversas técnicas desse campo de estudo podem ser classificadas segundo o paradigma empregado por elas na análise de dados (LORENA; CARVALHO, 2003), sendo o paradigma de aprendizado definido como responsável por determinar a maneira como o algoritmo de aprendizado de máquina se relaciona com seu meio ambiente, ou seja, o modo como se dará o seu aprendizado por meio dos dados (LORENA; CARVALHO, 2003; HAYKIN, 1999). Segundo esse critério, pode-se dividir as técnicas em dois grandes grupos: aprendizagem de forma supervisionada e aprendizagem de forma não supervisionada. Os dois grupos são comparados na Tabela 1.

Tabela 1 - Tipo de Aprendizagem de Máquina

Tipo de Aprendizagem de Máquina	Características
Aprendizagem supervisionada	Tem-se um "professor externo" que indica o conhecimento do ambiente por conjuntos de exemplos na forma (entrada, saída) para que a máquina aprenda a função desejada (LORENA; CARVALHO, 2003). Em outras palavras, o professor fornece o custo de cada padrão em um conjunto de treinamento com o intuito de minimizar a soma dos custos desses padrões (DUDA; HART; STORK, 2001).
Aprendizagem não supervisionada	A própria máquina aprende a formar conjuntos de forma autônoma a partir dos dados de entrada captados. Essas técnicas são utilizadas principalmente quando o objetivo do estudo é encontrar padrões ou tendências que auxiliam no entendimento dos dados (LORENA; CARVALHO, 2003; SOUTO et al., 2003).

No caso da aprendizagem supervisionada, o conjunto de exemplos fornecido para o algoritmo é geralmente composto por uma tabela de informações, onde cada linha representa as informações referentes a um exemplo específico e cada coluna representa um atributo (característica ou parâmetro) do determinado exemplo, além de conter uma última coluna referente à classe ou grupo a que o exemplo pertence. Nesses casos, o objetivo principal do algoritmo é classificar, a partir da análise de todas as informações presentes na base de dados, os novos exemplos que ainda não estejam rotulados quanto à classe pertencente. Todos esses

conceitos mencionados serão definidos no item 2.4 deste trabalho. A análise supervisionada pode ainda ser subdividida em dois outros grupos.

O primeiro deles é denominado regressão e é composto por problemas discretos, ou seja, a saída pode assumir valores discretos. Um exemplo pode ser a análise da tendência dos valores de ações da bolsa de valores, em que a saída não é uma classe pré-definida, mas sim um número discreto referente ao valor da ação analisada.

O segundo grupo é aquele que compreende problemas em que os valores da saída são classes. Por exemplo, classificar as condições climáticas em determinada região em: Ensolarado, Nublado ou Chuvoso. Esses são conhecidos como problemas de classificação.

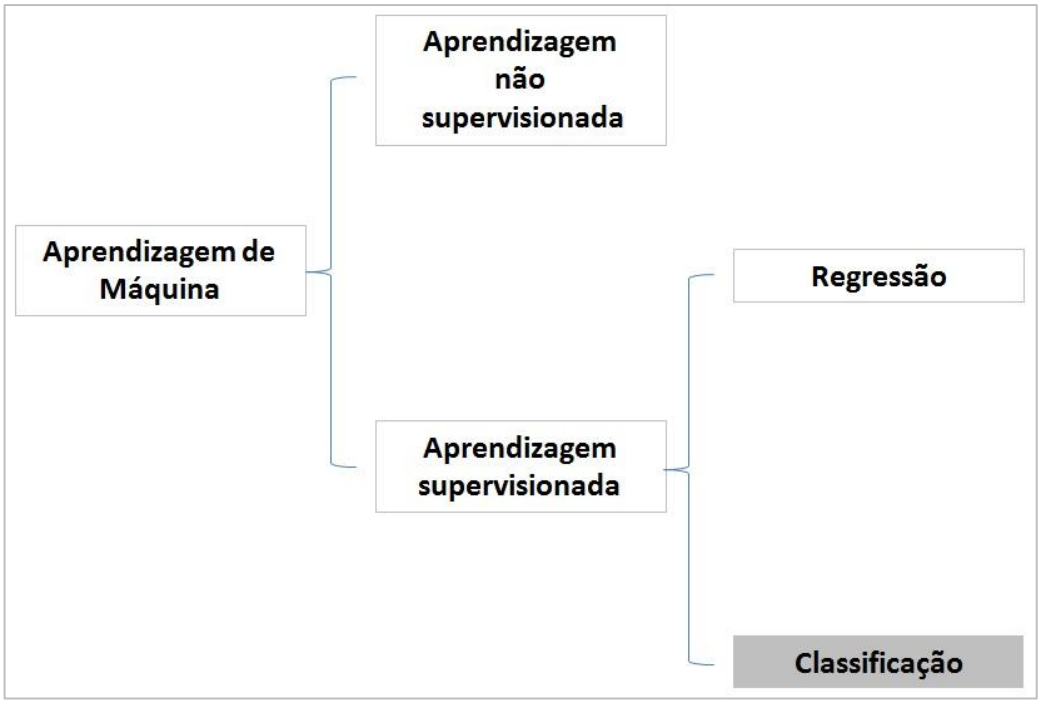

Figura 2.2 - Hierarquia de Aprendizagem de Máquina

O problema estudado nesse trabalho se enquadra em problemas de aprendizagem supervisionada e de classificação, pois almeja-se classificar novos dados fornecidos por sensores em classes pré-definidas a partir de dados previamente coletados e classificados.

2.4 Definições da Aprendizagem de Máquina

Algumas definições utilizadas aqui neste trabalho e também na literatura de Aprendizado de Máquina referentes à aprendizagem supervisionada serão fornecidas nessa seção. Não é pretendido fazer uma lista exaustiva e que contemple todas as definições desse campo de

estudo, mas sim as definições fundamentais para o entendimento do trabalho. Para essa seção foram utilizadas como base as obras (REZENDE, 2003; WEISS; INDURKHYA, 1997; WITTEN; FRANK, 2005; HE; GARCIA, 2009).

Conceito

Conceito é o que se espera de fato que seja aprendido. Por exemplo, dada uma base de dados composta por informações sobre componentes e respectivas concentrações em amostras de água retiradas ao longo de um determinado rio, determinar em quais localidades o rio se encontra poluído ou não-poluído. Nesse caso, o conceito é aprender quando o rio está poluído ou não.

Exemplo

Pode ser referenciado na literatura também como caso, registro ou dado. É um conjunto ou vetor de informações sobre o cenário analisado em um determinado instante e que contém as informações dos atributos referentes àquele momento. Pode ser descrito como:

$$E_1 = (x_1, x_2, x_3 \dots, x_n, y) \quad (1)$$

Onde:

- E_1: é o exemplo atual
- X_n: são os valores aos N atributos analisados
- Y: representa a classe a que pertence

Atributo

Descreve uma característica do cenário analisado e normalmente, em um problema real, diversos atributos são analisados concomitantemente. O atributo pode ser do tipo nominal, quando não existe ordem contínua dos valores (Ex. Coloração da amostra analisada: Branca, Amarela ou Laranja). Há também a possibilidade de o atributo ser do tipo contínuo, que engloba casos em que existe uma ordem contínua dos valores (Ex. Temperatura da amostra: um número real).

Usualmente existe um símbolo que indica quando o atributo é desconhecido ou não-determinado. Não há uma padronização desse símbolo nos diversos programas disponíveis de aprendizagem de máquina, porém muitas vezes o símbolo utilizado é o ponto de interrogação ‘?’.

Classe

Também denominado rótulo, a classe é um atributo especial dos exemplos e que descreve o fenômeno que se deseja aprender para fazer as previsões ou classificações. As classes são os grupos que podem ser classificados os exemplos da base de dados. Esta informação é fornecida nos dados de treinamento pelo usuário e será o dado de saída do programa no caso de novos dados analisados.

Conjunto de Treinamento

É um conjunto de exemplos contendo valores de atributos bem como o valor da classe associada a cada exemplo. O conjunto de treinamento pode ser exemplificado na forma mostrada na Tabela 2.

Tabela 2 - Modelo de conjunto de treinamento

	X_1	X_2	X_3	...	X_N	Y
E_1	X_{11}	X_{12}	X_{13}	...	X_{1N}	Y_1
E_2	X_{21}	X_{22}	X_{23}	...	X_{2N}	Y_2
E_3	X_{31}	X_{32}	X_{33}	...	X_{3N}	Y_3
⋮	⋮	⋮	⋮	...	⋮	⋮
E_N	X_{N1}	X_{N1}	X_{N1}	...	X_{NN}	Y_N

O conjunto de treinamento, assim como o nome sugere, tem como objetivo ensinar ao algoritmo a ser utilizado o conceito a ser estudado. Assim, nota-se a importância de se seguir os passos de pré-processamento, discutidos na Seção 2.2.1.

Conjunto de Teste

Apresenta a mesma estrutura do conjunto de treinamento. No entanto, esses exemplos são utilizados para mensurar a eficiência da etapa de treinamento. Esse teste é essencial para encontrar possíveis melhorias e correções no aprendizado do conceito, assegurando bons resultados na classificação de novos dados.

Classificador ou Preditor

É a hipótese gerada a partir de um determinado conjunto de treinamento, de forma a tornar-se possível a classificação de um novo exemplo quando requisitado.

Seja $Y(x_i)$ a saída de um determinado exemplo de treinamento com entrada x_i. O algoritmo ajusta seus parâmetros internos a partir do dado de treinamento para induzir uma hipótese de função h(x) que se aproxima da função Y(x). No exemplo fornecido, pode-se afirmar que $h(x_i) \approx Y(x_i)$.

Erro

Para avaliar o desempenho de um classificador, normalmente utiliza-se uma grandeza denominada taxa de erro. Para que seja obtida, compara-se o "gabarito" de classes dos exemplos (as classes verdadeiras de cada exemplo) com a resposta fornecida pelo classificador. Seja err(h) a taxa de erro de um classificador h, tem-se:

$$err(h) = \frac{1}{N}\sum_{i=1}^{N} \|Y_i \neq h(x_i)\| \quad (2)$$

Precisão

Precisão de um classificador é o complemento do erro. Assim, pode-se determinar que a precisão acc(h) é:

$$acc(h) = 1 - err(h) \quad (3)$$

Overfitting

Ao realizar o treinamento do algoritmo, é possível que seus parâmetros sejam ajustados em excesso de forma a gerar hipóteses muito específicas para o conjunto de treinamento, em detrimento da generalização e, assim, prejudicando a classificação de novos exemplos. Esse problema é avaliado geralmente por meio do conjunto de dados de teste. Ao aplicar o teste, a taxa de erro é calculada e então é possível concluir se o treinamento está ou não como esperado.

Underfitting

Como o nome sugere, é justamente o oposto do overfitting. Nesse caso, as hipóteses geradas a partir dos dados de treinamento não geram taxas de erro baixas. Por conseqüência, o mesmo ocorre para os dados de teste. Assim, é necessário reajustar o algoritmo para que o desempenho seja melhorado.

Matriz de confusão ou Confusion Matrix

Matriz de confusão é uma tabela em que as classes verdadeiras são apresentadas nas linhas e as hipóteses geradas (ou predições) são apresentadas nas colunas. Dessa forma, os elementos na diagonal principal representam os exemplos corretamente classificados, enquanto que a diagonal secundária apresenta os exemplos incorretamente classificados.

Tabela 3 - Matriz de confusão

Saída
(Hipótese gerada)

Entrada
(Classe verdadeira)

	Exemplos positivos	Exemplos negativos
Exemplos positivos	TP (Verdadeiro Positivo)	FN (Falso Negativo)
Exemplos negativos	FP (Falso Positivo)	TN (Verdadeiro Negativo)

Onde:

Total de exemplos positivos: $P = TP + FN$

Total de exemplos negativos: $N = FP + TN$

Erro Total

Como o nome sugere, é o total de classificações incorretas geradas. Dessa forma, pode-se afirmar que:

$$Erro\ total = \frac{(FP+FN)}{(P+N)} \tag{4}$$

Taxa de falso positivo

É a porcentagem de falsos positivos gerado, ou seja:

$$Fp(\%) = \frac{FP}{N} \tag{5}$$

Taxa de verdadeiro positivo ou comumente conhecido por *Hit ratio*

É a porcentagem de exemplos positivos corretamente classificados, ou seja:

$$Tp(\%) = \frac{TP}{P} \tag{6}$$

Curva ROC

Curva ROC – *Receiver Operating Characteristic* – é uma representação gráfica da taxa de hit ratio versus taxa de falsos positivos para sistemas de classificações binárias. Os pontos da curva representam o desempenho de um único classificador em uma determinada distribuição. Essa forma de representação de desempenho é muito útil, pois ela proporciona uma representação visual do custo/benefício entre os benefícios (representados pelo Hit ratio) e os custos (representados pelos falsos positivos). A figura a seguir ilustra um exemplo de curva ROC.

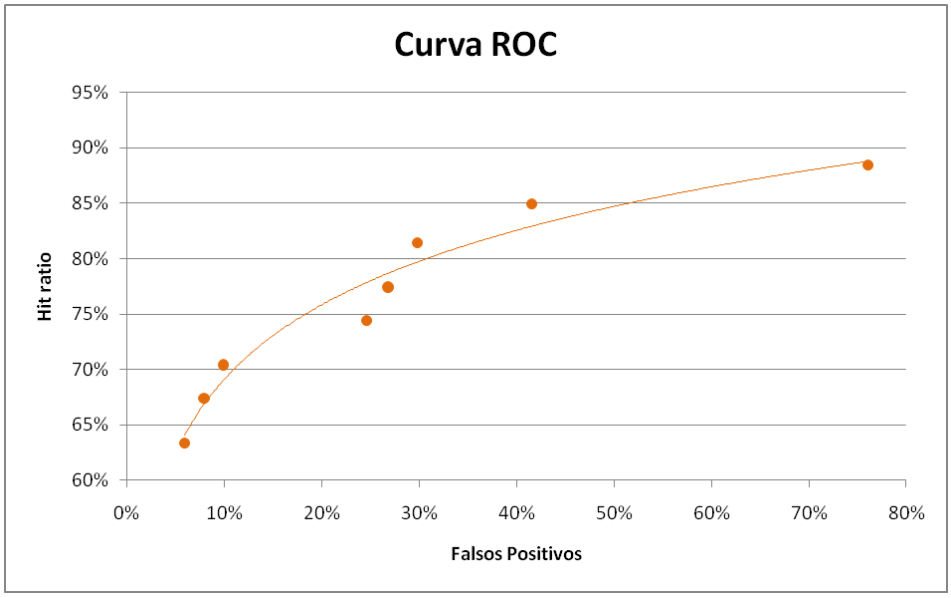

Figura 2.3 - Ilustração de uma curva ROC

2.5 Combinação de Classificadores

No processo de aprendizagem a partir de uma base de dados previamente coletados e selecionados, as técnicas utilizadas geram primeiramente os preditores. Esses são os produtos da etapa conhecida por "Extração de padrões" e que tem por objetivo encontrar modelos de acordo com os dados (PUGLIESI; SINOARA; REZENDE, 2003). Em seguida, ocorre o chamado pós-processamento em que o resultado é avaliado de acordo com a qualidade ou utilidade e então, aplicado tomada de decisão. Nessa etapa, os preditores mais adequados no contexto são utilizados.

Há uma busca constante pelo aprimoramento dos algoritmos de Aprendizado de Máquina, tanto para aumentar a acurácia e minimizar os erros, como para otimizar o tempo de aprendizado. Nesse sentido, foram desenvolvidos métodos para combinar preditores e utilizá-

los de forma conjunta. Pesquisas e casos de uso indicam que essas aplicações têm gerado resultados mais precisos se comparados com os obtidos com um único preditor (FREUND; SCHAPIRE, 1998).

Ensemble, também conhecido por comitê, é um conjunto de classificadores cujas conclusões individuais são combinadas de alguma forma a classificar o problema abordado (BERNARDINI, 2002). A proposta principal dos ensembles é combinar classificadores e aproveitar as contribuições de cada um deles para conseguir melhores resultados de classificação em detrimento da complexidade computacional e também maior custo computacional (BABENKO; MARMANIS, 2009). Essa técnica é muito utilizada nos problemas de classificação (FACELI, 2001; TUMER; GHOSH, 1996; MERZ, 1998).

Existem três principais motivos que sugerem a causa pela qual um conjunto de classificadores tem um funcionamento melhor do que um único classificador trabalhando de forma isolada (BERNARDINI, 2002; DIETTERICH, 2000):

I. *Estatística:* Quando o número de hipóteses do sistema analisado é muito maior do que o número de pontos a serem utilizados para o treinamento, o classificador pode tender a fornecer uma resposta errada sobre o sistema. Por outro lado, um conjunto de classificadores é capaz de avaliar as respostas de todos os classificadores um a um, e determinar qual é a mais possível de estar correta, aumentando assim as chances de representar o sistema real.

II. *Computacional:* Segundo Dietterich, os classificadores são muitas vezes sensíveis ao treinamento. Assim, combinando classificadores é possível fornecer melhores resultados, suavizando as sensibilidades de cada classificador na coleta dos dados durante o treinamento. Dessa forma é possível obter uma aproximação maior à função do sistema real se comparada à resposta de cada um dos classificadores de forma isolada.

III. *Representacional:* Na maioria dos casos reais não é possível atingir com perfeição a função exata do sistema analisado. No entanto, por meio de ponderações ou fusão das diversas hipóteses, é possível expandir o espaço das funções representáveis.

Diversos métodos de construção de *ensembles*, ou conjunto de classificadores, têm sido apresentados, entre eles o método de manipulação de exemplos de treinamento. O

funcionamento ocorre por meio da execução de um mesmo algoritmo de aprendizado por diversas vezes, porém alterando os subconjuntos de treinamento. Melhores resultados desse método são obtidos principalmente quando o algoritmo de aprendizado (algoritmo fraco ou base) é considerado instável como, por exemplo, redes neurais e árvores de decisão (BERNARDINI, 2002). Sobre a forma com que os conjuntos de treinamento são manipulados, pode-se subdividir este grupo em duas classes, de acordo com:

i. *Perturb and Combine:* Para garantir a geração de diferentes hipóteses, o conjunto de treinamento ou método de construção da hipótese é perturbado. Ex: *Bagging*.

ii. *Adaptively Resample and Combine:* Nessa técnica as amostras de treinamento são combinadas e refeitas de forma adaptativa. Ex: *Boosting*.

Existem diversas formas para realizar a combinação das respostas de diferentes classificadores e assim atingir a resposta final do *ensemble*. Entre elas, pode-se destacar as duas técnicas a seguir (FACELI, 2001) :

i. *Decisão por maioria:* Nesse caso todas as saídas têm o mesmo peso e a resposta final é aquela que foi apresentada em maior quantidade de vezes.

ii. *Decisão por ponderação das saídas:* Nesse caso as saídas recebem pesos geralmente obtidos de acordo com a precisão de cada classificador individual no conjunto de treinamento ou teste.

A seguir serão introduzidos os métodos de combinação de classificadores denominados *Bagging* e *Boosting*.

2.5.1 Bagging

Publicado no meio acadêmico pela primeira vez por Breiman em 1996 (BREIMAN, 1996), o método *Bootstrap Aggregating* tem o propósito de combinar preditores gerados por um mesmo algoritmo base com o objetivo de reduzir a variância de funções preditivas. Conhecido popularmente por *Bagging*, o método apresenta bons resultados em algoritmos base instáveis como redes neurais e, principalmente, árvores de decisão (CREAMER; FREUND, 2005). O

procedimento é classificado como instável se pequenas mudanças nos dados acarretarem em grandes alterações nos classificadores e na previsão final.

No método *Bagging* são gerados conjuntos sucessivos e independentes de amostras, denominadas *bootstrap*, a partir do conjunto de treinamento com n exemplos. Assim, cada *bootstrap* é gerado escolhendo de forma aleatória n exemplos, com substituição, de forma que todos apresentem a mesma quantidade de exemplos de T (PUGLIESI; SINOARA; REZENDE, 2003). Porém, alguns exemplos podem aparecer mais de uma vez e outros simplesmente não serem escolhidos.

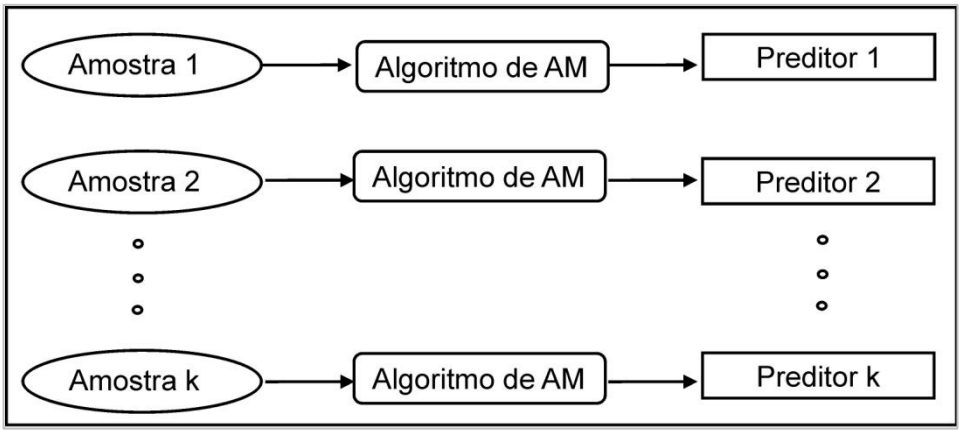

Figura 2.4 - Geração paralela de classificadores

Os classificadores são gerados de forma paralela a partir das amostras de *bootstraps* (Figura 2.4) e o classificador final é obtido por meio de uma média desses classificadores parciais.

$$C_f = \frac{1}{n}\sum_{i=1}^{n} C_i \tag{7}$$

2.5.2 Boosting

De acordo com Feund e Schapire, pode-se definir *Boosting* como sendo um método para aprimorar o desempenho de qualquer algoritmo de aprendizado (FREUND; SCHAPIRE, 1996). Portanto, essa técnica não é utilizada de forma isolada, sendo aplicada de forma combinada com outras técnicas, como Árvores de Decisão e Redes Neurais, com o objetivo de transformar os classificadores “fracos” em classificadores “fortes”. O *Boosting* trabalha com a combinação de classificadores gerados por um mesmo algoritmo de aprendizado

(conhecido por algoritmo base ou fraco) e seu funcionamento é ajustado de acordo com os erros cometidos pelo classificador anterior.

Neste método, o classificador final é denominado *classificador forte*, que por sua vez é composto por diversos outros classificadores, denominados *classificadores fracos* ou *classificadores base*. Cada classificador fraco, de forma isolada, é somente parcialmente correlacionado com a verdadeira classificação almejada. No entanto, quando combinados pelo método geram o chamado classificador forte, que fornece uma classificação completa e com maior acurácia do problema.

Assim, o diferencial do *Boosting* é a busca por gerar novos classificadores melhores e mais adequados para o problema, principalmente por corrigirem e aumentarem a eficiência dos classificadores gerados de forma isolada.

A principal diferença ente *Bagging* e *Boosting* é que este último gera conjuntos de treinos e classificadores de forma sequencial, baseados nos resultados da iteração anterior (Figura 2.5), enquanto que o Bagging gera esses conjuntos de treino de forma aleatória e pode gerar os classificadores paralelamente, como já discutido anteriormente.

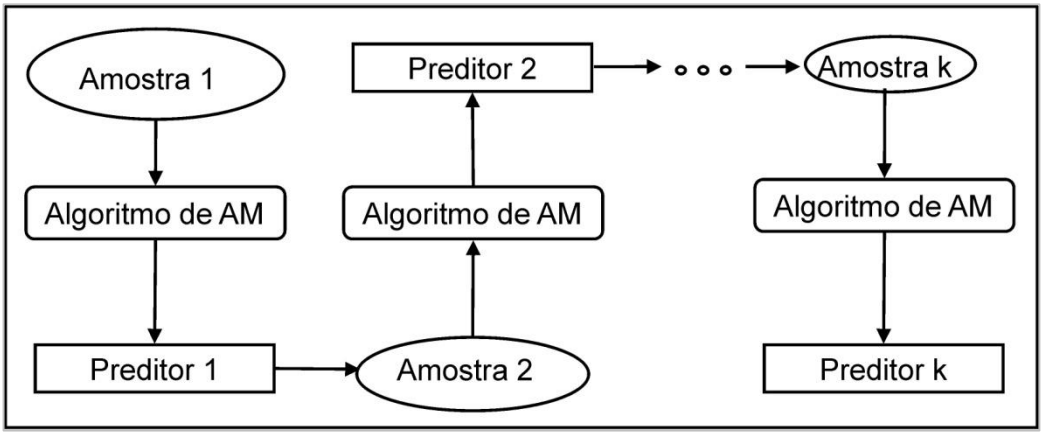

Figura 2.5 - Geração sequencial de classificadores

2.6 Algoritmos base

Na Seção 2.5.2 e de acordo com o conceito de *Boosting*, os *algoritmos base* ou *algoritmos fracos* foram definidos em linhas gerais como sendo uma função que produz classificadores fracos. Os classificadores fracos não devem apresentar por si só uma habilidade muito forte para fazer discriminações, pois caso isso aconteça, muito provavelmente ocorrerão problemas de *overfitting* durante a etapa de treinamento (SUSNJAK, 2009). Três possíveis

classificadores fracos serão discutidos a seguir: Redes Neurais, Árvores de Decisão e *Decision Stump*.

2.6.1 Redes Neurais

Redes Neurais Artificiais (ANNs), também conhecidas por neurocomputadores, redes conexionistas, processadores paralelamente distribuídos e redes neurônicas, são modelos computacionais que têm como motivação o funcionamento do cérebro humano, que é um sistema de processamento de informação altamente complexo, não-linear e paralelo. O cérebro organiza seus componentes estruturais, denominados neurônios, de forma a realizar certos processamentos, como reconhecimento de padrões, muito mais rapidamente que qualquer computador desenvolvido pelo homem.

A semelhança dessa técnica com o cérebro deve-se principalmente a dois aspectos. O primeiro deles é que o conhecimento é adquirido pela rede a partir de seu ambiente por meio de um processo de aprendizagem. Outra semelhança é que as forças de conexão entre neurônios, conhecidas por pesos sinápticos, são utilizadas para armazenar o conhecimento adquirido (HAYKIN, 1999). Assim, a fácil adaptabilidade a mudanças do meio ambiente por meio dos pesos sinápticos é um dos grandes diferenciais dessa técnica. Destaca-se também a capacidade de trabalhar sem grandes alterações com sinais de entrada não lineares e permitir extrair informações sobre a confiança da tomada de decisão, permitindo assim uma análise mais coerente em casos de ambiguidade no reconhecimento de padrões, por exemplo. Devido ao seu funcionamento, algumas linhas de pesquisa afirmam que as Redes Neurais fornecem resultados de classificação superiores para base limitada de dados de treinamento se comparadas com outros métodos convencionais (KAVZOGLU, 2009).

O modelo de redes neurais é formado por uma densa interconexão de elementos computacionais mais simples e denominados neurônios (Figura 2.6), também chamados de nós ou células.

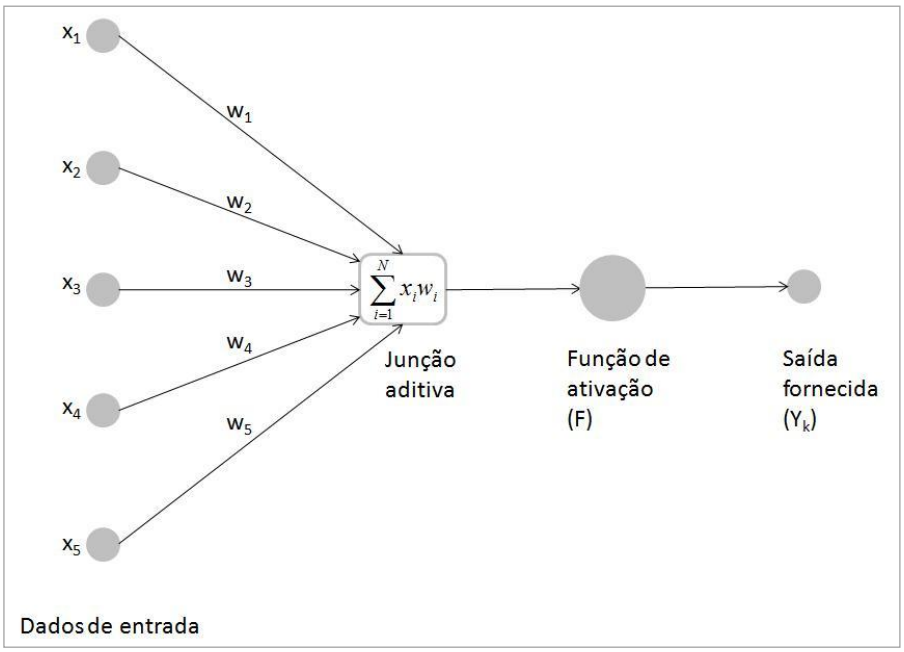

Figura 2.6 - Esquema de um neurônio

Como demonstrado, os componentes básicos deste modelo são os sinais de entrada, os pesos sinápticos, a Junção Aditiva, função de ativação não linear e a saída fornecida do neurônio. A junção aditiva é um somador simples de N entradas já com os respectivos pesos definidos para fornecer o valor de saída.

$$Y_k = F(\sum_{i=1}^{N} x_i w_i) \tag{8}$$

A Figura 2.7 mostra o esquema de uma rede neural em camadas, mais conhecida por Multi Layer Perceptron. Esse modelo é classificado como acíclico, pois as informações percorrem sempre no sentido da próxima camada, ou seja, não retornam em nenhum momento para a camada anterior. A primeira delas é a camada de entrada de neurônios de fonte, que fornecem os dados iniciais para a rede. Em seguida aparecem algumas camadas de neurônios ocultos que têm por objetivo refinar e agregar valor aos dados de saída. O maior número dessas camadas intermediárias permite que a rede extraia estatísticas de ordem mais elevadas. Por isso, a determinação da quantidade dessas camadas depende da aplicação, complexidade e precisão desejada.

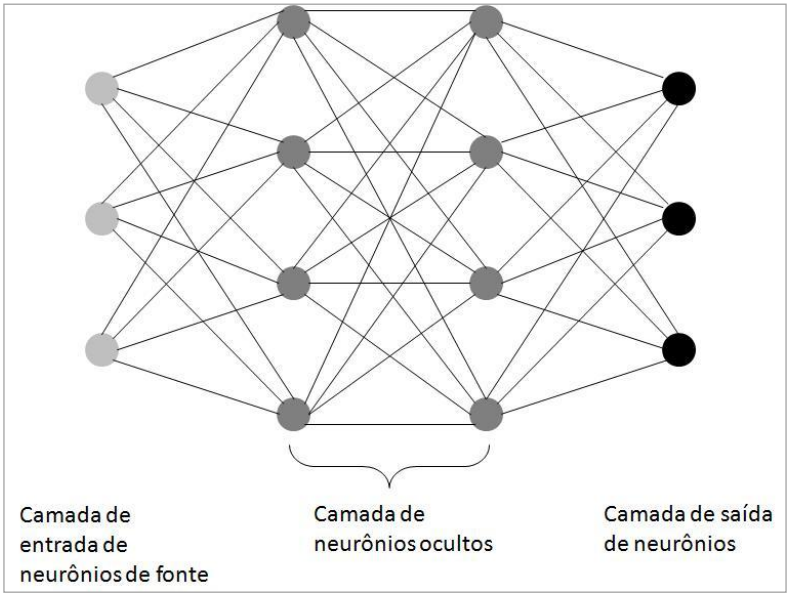

Figura 2.7 - Redes Neurais em camadas

2.6.2 Árvore de Decisão

Trata-se de um modelo prático de uma função recursiva que gera uma estrutura de árvore de forma a simplificar e ajudar a classificação e predição de amostras desconhecidas. A ideia principal desse modelo é subdividir um problema grande e complexo em sub-problemas mais simples de serem resolvidos. Essa estratégia é assim aplicada também para cada sub-problema até se atingir a forma mais simples. Assim, o modelo de árvore é montado baseado nas informações disponíveis na base de dados de treinamento. Dessa forma, é viável classificar a amostra analisada sem que seja necessário realizar testes com todos os valores dos seus atributos.

O esquema de uma Árvore de decisão pode ser verificado na Figura 2.8 apresentada a seguir. Os elementos básicos presentes nesse algoritmo são o nó raiz ou dado de entrada, os nós de decisão ou comuns, que dividem um atributo e geram ramificações, e por fim os nós folha que apresentam as informações de classificação do algoritmo.

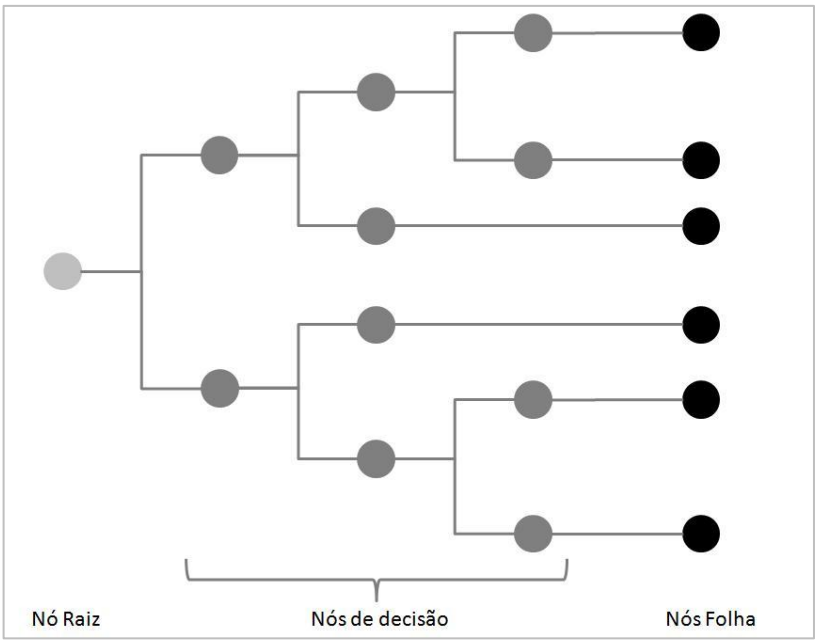

Figura 2.8 - Modelo de Árvore de decisão

Nesse modelo, cada nó representa um teste de um atributo e cada percurso na árvore, da raíz até a folha, corresponde a uma regra de classificação. Assim, a classificação de uma amostra é realizada percorrendo toda a estrutura da árvore utilizando os valores e atributos da amostra analisada.

Como diferencial desse modelo pode-se destacar o elevado grau de interpretabilidade, já que uma decisão complexa é decomposta em decisões elementares. Além disso, é possível afirmar que as Árvores de decisão são robustas à presença de atributos redundantes ou irrelevantes devido ao mecanismo de seleção de atributos. Por outro lado, o modelo também é classificado como instável, pois pequenas perturbações do conjunto de treinamento podem provocar grandes alterações no modelo aprendido final.

2.6.3 Decision Stump

Decision Stump são Árvores de Decisão que consistem de apenas duas folhas, ou seja, um nível apenas de árvore. Assim, cada classificador fraco gerado por ele é capaz de analisar apenas uma decisão do conceito geral do problema analisado.

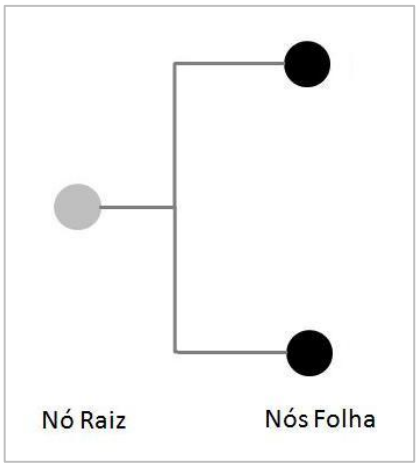

Figura 2.9 - Modelo *Decision Stump*

É considerado um dos mais simples algoritmos fracos e bastante utilizado em estudos com o método *Boosting*. Portanto, essa análise simplificada que o algoritmo é capaz de realizar pode não ser tão eficiente para solucionar problemas muito complexos ou com muitos parâmetros analisados ou até mesmo ocorrer problemas de *underfitting* durante a etapa de treinamento. Por outro lado, sua simplicidade implica em classificações mais rápidas.

2.6.4 Random Forest

Este também é um classificador baseado em árvores de decisão, porém nesse caso consiste em várias árvores de decisão combinadas de forma a gerarem apenas um classificador final (BREIMAN, 2001). O primeiro nome do algoritmo se deve à maneira aleatória como são escolhidas as análises em cada etapa. Apesar de ter como princípio a combinação de classificadores do modelo *Bagging*, existem já pesquisadores utilizando o algoritmo como algoritmo base no modelo *Boosting* (Zhang; Xie, 2010; LESHEM; RITOV, 2007; LESHEM, 2005).

3 Adaptive Boosting

O modelo *Boosting*, descrito no capítulo anterior, compreende na verdade uma família de algoritmos dentre os quais o algoritmo chamado *Adaptive Boosting* é o mais influente e popular. O algoritmo é popularmente conhecido por *AdaBoost*. No capítulo anterior deste trabalho foi introduzido o conceito do modelo de *Boosting*. Por sua vez, esse capítulo tem como objetivo abordar de forma sucinta o histórico do algoritmo *AdaBoost* (Seção 3.1), discutir seu funcionamento (Seções 3.2 e 3.3), abordar algumas estruturas de treinamento de classificadores (Seção 3.4), variações desse algoritmo (Seção 3.5), bem como apresentar alguns casos de uso e aplicações práticas do algoritmo (Seção 3.6).

3.1 Visão Geral

Publicado pela primeira vez em 1995 por Feund e Schapire (FREUND; SCHAPIRE, 1996; FREUND; SCHAPIRE, 1996), o algoritmo resolveu diversos problemas e dificuldades encontradas anteriormente sobre *Boosting* e atualmente apresenta crescente número de pesquisas sobre a metodologia e suas aplicações por estudiosos de diversas partes do mundo (REYES; BARCZAK; MESSOM, 2006; BARCZAK; JOHNSON; MESSOM, 2008). Ainda segundo os inventores desse algoritmo, o AdaBoost apresenta algumas propriedades específicas que o tornam mais prático de ser utilizado e implementado se comparado aos outros algoritmos predecessores a ele (FREUND; SCHAPIRE, 1997). Dentre os principais diferenciais desse algoritmo é possível destacar:

- **Diferenças de parâmetros utilizados:** Para análises de grandes espaços dimensionais, na casa de milhões, os valores de margem entre classes podem se apresentar muitas vezes bastante diferentes e mais precisos do que outros métodos, como o SVM (FREUND; SCHAPIRE, 1999). Ressaltando que valor de margem indica o quanto duas ou mais classes estão distantes das fronteiras de decisão formadas para sua classificação (LORENA; CARVALHO, 2003).

- **Requer baixo valor computacional:** AdaBoost corresponde a um programa linear e, portanto, não precisa de componentes eletrônicos pesados para funcionar e fazer as análises (FREUND; SCHAPIRE, 1999).

Outras vantagens do AdaBoost que merecem ser destacadas são a simplicidade de implementação, flexibilidade para ser utilizado em diferentes áreas de conhecimento e ocasiões, além da possibilidade de ser aplicado junto a diferentes classificadores base (SCHAPIRE; SINGER, 1998; SCHAPIRE, 2000).

Um ponto de atenção sobre o AdaBoost é em relação aos possíveis ruídos no sistema. Devido ao princípio do algoritmo de enfatizar os dados mais difíceis de serem classificados, os ruídos podem ser enquadrados nesse grupo e assim prejudicar os resultados gerados. Existem alguns princípios e formulações para minimizar essas influências dos ruídos nos dados de treinamento, porém não serão discutidos nesse trabalho.

Como já mencionado, o AdaBoost é utilizado junto a algoritmos base para cruzar duas ou mais informações sobre dados analisados e, ao comparar com a base de dados de treinamento, reconhecer padrões e fazer as classificações. Sendo assim, suponha que duas informações diferentes (x e y) foram coletadas sobre diversos objetos de uma amostra. Ao serem analisadas por um algoritmo já treinado previamente, o resultado final adquirido será a classificação automática de acordo com os padrões encontrados e comparados com os dados de treinamento, como ilustrado no gráfico abaixo (Figura 3.1). Este resultado pode servir de entrada para uma tomada de decisão da própria máquina ou do controlador dela.

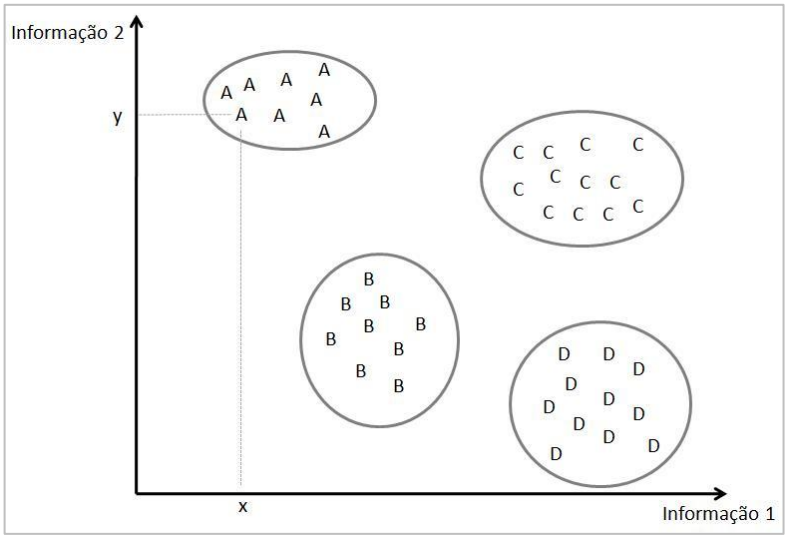

Figura 3.1 - Exemplo de classificação fornecida pelo AdaBoost

Este exemplo com apenas dois dados de entrada – informação 1 e informação 2 – encontrou quatro diferentes grupos que apresentam o mesmo padrão. No entanto, a utilização do algoritmo é justificada quando são utilizadas grandes quantidades de dados de entrada e maior quantidade de propriedades analisadas (informações), pois tornam o problema em

análises de informações com dimensões não triviais – problemas com 3 ou mais dimensões. Dessa forma todo o seu potencial é demonstrado e, a cada análise correta de novos dados, o algoritmo "aprende" e, assim, aprimora a acurácia das análises.

Após a proposta original do AdaBoost ter sido apresentada aos cientistas e pesquisadores, muitas variações e extensões deste algoritmo foram sugeridas e desenvolvidas como alternativas para fornecerem melhores resultados para problemas específicos. Alguns dos motivos para o surgimento dessas novas propostas são as aplicações analisadas (ex. tratamento de imagens) e também o tipo do problema abordado (ex. binário ou multiclasse).

3.2 Funcionamento do AdaBoost

Os trabalhos (FREUND; SCHAPIRE, 1996; OZA, 2001) foram utilizados como referência para a explicação do funcionamento do algoritmo dada a seguir.

A primeira etapa para o funcionamento do algoritmo é o treinamento. Nessa fase, a entrada para o AdaBoost é um conjunto de exemplos na forma $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$, onde:

a. Cada x_i representa um vetor de atributos do sistema, ou seja, um conjunto de dados referente aos parâmetros (ou propriedades) analisados e que representam um dado instante do sistema.

b. Dentre todos os possíveis grupos de classificação predefinidos para o sistema analisado, cada y_i representa o grupo de classificação associado ao x_i.

c. O número total de exemplos da amostra de treinamento é igual a m.

O algoritmo base escolhido para trabalhar junto ao AdaBoost é convocado de forma repetitiva por uma série de rodadas, sendo que para cada rodada o AdaBoost fornece ao algoritmo base uma distribuição de pesos referentes a cada um dos dados de treinamento. Os classificadores iniciam com pesos associados igualmente distribuídos.

$$D_0 = \frac{1}{m} \tag{9}$$

A cada ciclo de aprendizagem, o algoritmo base gera uma hipótese h_t baseada nos pesos atuais com o objetivo de priorizar a classificação correta dos dados que apresentam os maiores pesos associados. Logo, pode-se afirmar que o objetivo do algoritmo base é gerar uma hipótese de forma a minimizar o erro de treinamento e_t.

$$e_t = \sum_{i:h_t(x_t)\neq y_t} D_t(i) \tag{10}$$

Em seguida, esses pesos são reavaliados de forma a aumentar aqueles que são relacionados aos dados incorretamente classificados e uma nova rodada se inicia. Por fim, ao serem realizadas todas as T rodadas determinadas previamente, o AdaBoost combina todas as hipóteses intermediárias ht de forma a gerar uma única hipótese final H(x).

De acordo com (FREUND; SCHAPIRE, 1999) o método para se obter a hipótese final H(x) fornecida pelo AdaBoost é a combinação ponderada das diversas saídas das iterações. Assim, uma votação ponderada das T hipóteses fracas é realizada, onde γ_t é a importância associada à hipótese h_t.

$$H(x) = sign(\sum_{t=1}^{T} \gamma_t h_t(x)) \tag{11}$$

O esquema a seguir (Figura 3.2) resume o funcionamento do algoritmo.

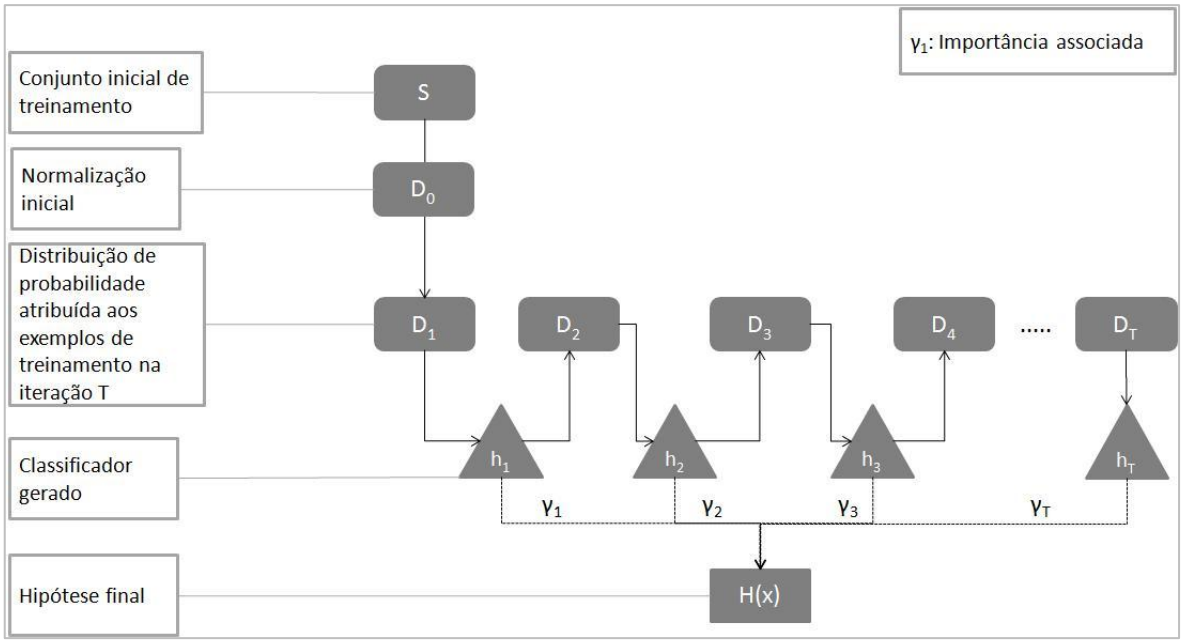

Figura 3.2 - Esquema do funcionamento do AdaBoost

Na forma de linguagem, também é fornecido um exemplo para ilustrar o funcionamento do algoritmo AdaBoost.

1: Dado um conjunto de treinamento $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, com $y_i \in \{-1, +1\}$
2: Inicializar $D_t(\mathbf{x}_i) \leftarrow \frac{1}{n}$ e $t \leftarrow 1$
3: **repita**
4: Treinar o algoritmo base utilizando a distribuição D_t
5: Calcular o erro ϵ_t da hipótese h_t gerada

$$\epsilon_t \leftarrow \Pr_{i \sim D_t}\left[h_t(\mathbf{x}_i) \neq y_i\right] = \sum_{i:h_t(\mathbf{x}_i) \neq y_i} D_t(\mathbf{x}_i)$$

6: Escolher $\gamma_t \leftarrow \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$ e $Z_t \leftarrow \sum_{i=1}^{n} D_t(\mathbf{x}_i)\exp\left(-\gamma_t y_i h_t(\mathbf{x}_i)\right)$, em que Z_t é um termo normalizador
7: Atualizar D_t:

$$D_{t+1}(\mathbf{x}_i) \leftarrow \frac{D_t(\mathbf{x}_i)}{Z_t} \times \begin{cases} \exp(-\gamma_t) & se\, h_t(\mathbf{x}_i) = y_i \\ \exp(\gamma_t) & se\, h_t(\mathbf{x}_i) \neq y_i \end{cases}$$
$$= \frac{D_t(\mathbf{x}_i)\exp(-\gamma_t y_i h_t(\mathbf{x}_i))}{Z_t}$$

8: $t = t + 1$
9: **até** $t = T$
10: Hipótese Final:

$$H(\mathbf{x}) \leftarrow \operatorname{sgn}\left(\sum_{t=1}^{T} \gamma_t h_t(\mathbf{x})\right)$$

Figura 3.3 - Algoritmo AdaBoost (LORENA; CARVALHO, 2003)

Onde:

D_t: Distribuição de pesos atribuída aos exemplos de treinamento

t: O ciclo de aprendizagem que o algoritmo se encontra

T: Número total de ciclos que o algoritmo será iterado

h_t: Hipóteses geradas a cada iteração t

e_t: Erro cometido durante a iteração

Z_t: Termo normalizador

γ_t: Importância associada do classificador, calculada a partir do erro e_t

De acordo com o algoritmo (Figura 3.3), a importância de um classificador fraco h_t é dada pela equação:

$$\gamma_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) \tag{12}$$

Assim, é possível observar e confirmar que quanto menor for a taxa de erro (e_t) maior será a importância atribuída ao classificador, como demonstrado na figura a seguir.

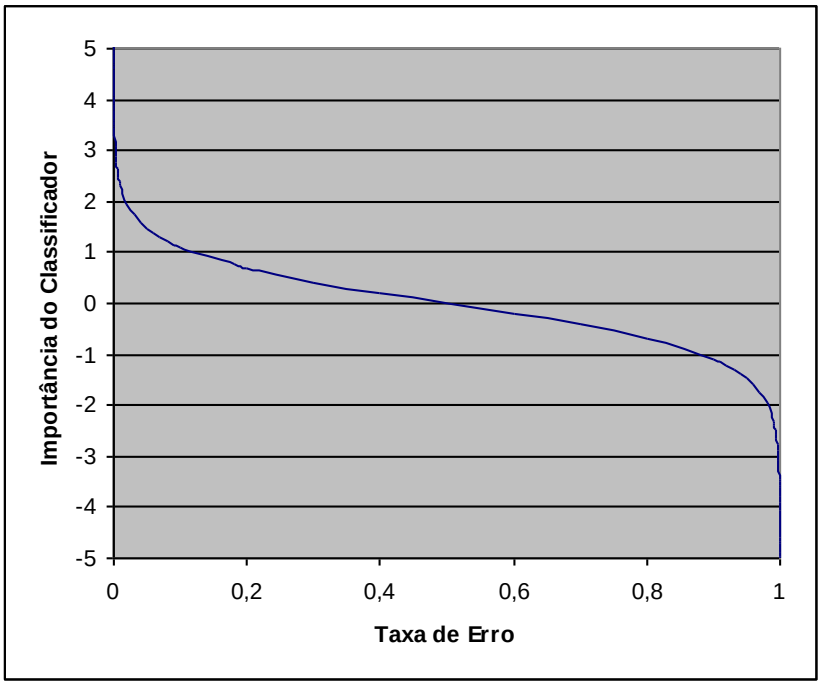

Figura 3.4 - Taxa de erro versus Importância do classificador

Ainda de acordo com a Figura 3.3, durante a etapa 7, a distribuição de pesos atribuída aos exemplos de treinamento é atualizada, de acordo com a seguinte equação.

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} * \begin{cases} \exp(-\gamma_t) \; se \; h_t(x_i) = y_i \\ \exp(\gamma_t) \quad se \; h_t(x_i) \neq y_i \end{cases} \tag{13}$$

Traduzindo em palavras, essa etapa é responsável por aumentar os pesos dos exemplos erroneamente classificados e diminuir os pesos dos exemplos corretamente classificados.

Por fim, uma vez treinado o algoritmo, os parâmetros e coeficientes calculados durante a etapa de treinamento são armazenados. Assim, esses coeficientes são utilizados para classificar novos dados fornecidos ao algoritmo.

3.3 Exemplo Didático

Nessa seção tem-se como objetivo demonstrar como funciona a etapa de treinamento do algoritmo AdaBoost por meio de um exemplo didático.

Seja um conjunto de treinamento composto por 10 exemplos e duas possíveis classes: quadrados e circunferências. Abaixo está ilustrada a distribuição original.

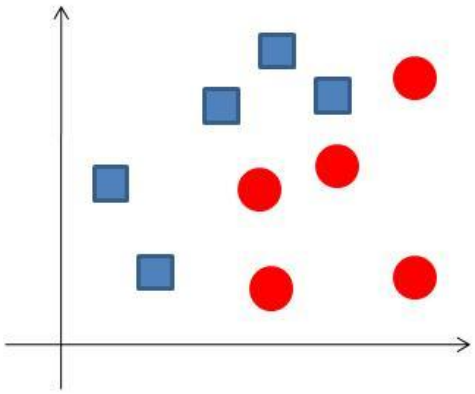

Figura 3.5 - Conjunto de treinamento inicial

Inicialmente tem-se a distribuição dos pesos da forma:

$$D_0 = \frac{1}{10} \tag{14}$$

Durante a primeira iteração, dois quadrados foram classificados erroneamente:

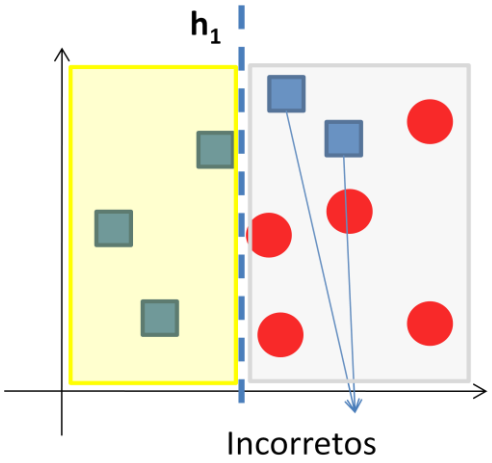

Figura 3.6 - Conjunto de treinamento - Iteração 1

Gerando assim um erro e_1, além da importância associada γ_1:

$$e_1 = \frac{1}{10} + \frac{1}{10} = 0{,}200$$
$$\gamma_1 = \frac{1}{2}\ln\left(\frac{1-0{,}20}{0{,}20}\right) = 0{,}693 \tag{15}$$

Ainda nessa primeira iteração a distribuição de pesos é reavaliada, gerando a distribuição de pesos D_t:

$$D_t = \begin{cases} \frac{0,1}{1} \times e^{-0,693} = 0,500; \ para\ os\ pontos\ classificados\ corretamente \\ \frac{0,1}{1} \times e^{0,693} = 1,999; \ para\ os\ pontos\ classificados\ incorretamente \end{cases}$$

Como próxima etapa, é necessário normalizar os pesos calculados acima. Assim, o termo normalizador é calculado a seguir:

$$Z_t = (8 * 0,500) + (2 * 1,999) = 7,998 \tag{16}$$

Portanto, após a etapa de normalização, os respectivos pesos e a representação gráfica são:

$$D_t = \begin{cases} \frac{0,500}{7,998} = 0,062; \ para\ os\ pontos\ classificados\ corretamente \\ \frac{1,999}{7,998} = 0,250; \ para\ os\ pontos\ classificados\ incorretamente \end{cases}$$

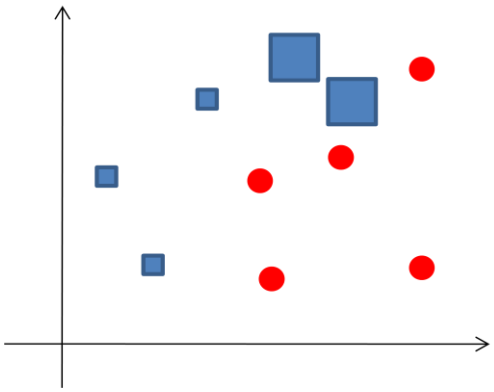

Figura 3.7 - Conjunto de treinamento - Iteração 1 após distribuição dos pesos

Seguindo os mesmos passos descritos acima ocorre a segunda iteração, gerando o erro e importância associada abaixo.

$$e_2 = 0,062 + 0,062 + 0,062 = 0,186$$
$$\gamma_2 = \frac{1}{2}\ln\left(\frac{1-0,186}{0,186}\right) = 0,738 \tag{17}$$

Obtém-se também a nova distribuição de pesos:

$$D_t = \begin{cases} 0{,}062 \times e^{-0{,}738} = 0{,}029;\ para\ os\ 5\ pontos\ classificados\ corretamente\ nas\ duas\ rodadas \\ 0{,}250 \times e^{-0{,}738} = 0{,}119;\ para\ os\ 2\ pontos\ classificados\ incorretamente\ na\ 1^a\ rodada \\ 0{,}062 \times e^{0{,}738} = 0{,}130;\ para\ os\ 3\ pontos\ classificados\ incorretamente\ na\ 2^a\ rodada \end{cases}$$

O termo normalizador e a normalização dos pesos obtidos na segunda iteração são apresentados a seguir.

$$Z_t = (5 * 0{,}029) + (2 * 0{,}119) + (3 * 0{,}062) = 0{,}780 \quad (18)$$

$$D_t = \begin{cases} \frac{0{,}029}{0{,}780} = 0{,}038;\ para\ os\ 5\ pontos\ classificados\ corretamente\ nas\ duas\ rodadas \\ \frac{0{,}119}{0{,}780} = 0{,}153;\ para\ os\ 2\ pontos\ classificados\ incorretamente\ na\ 1^a\ rodada \\ \frac{0{,}130}{0{,}780} = 0{,}167;\ para\ os\ 3\ pontos\ classificados\ incorretamente\ na\ 2^a\ rodada \end{cases}$$

De forma ilustrativa, tem-se que a iteração 2 é dada pela Figura 3.8:

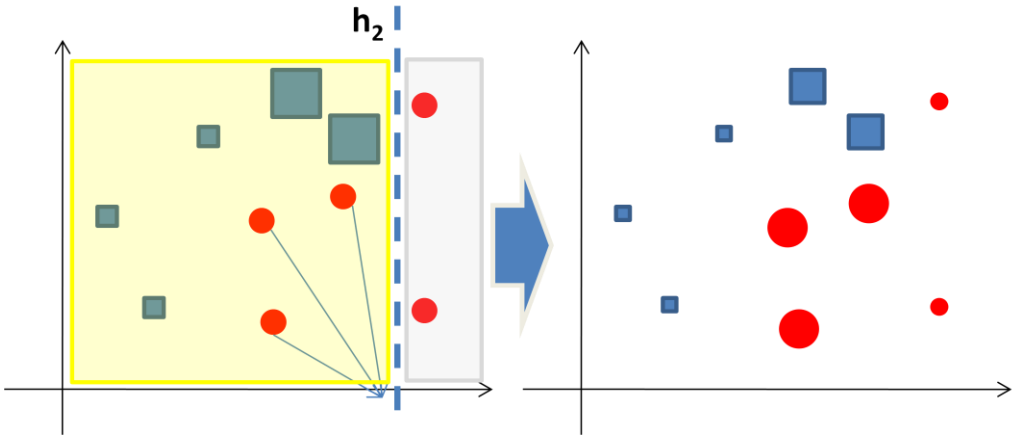

Figura 3.8 - Conjunto de treinamento - Iteração 2

Por sequência, a iteração 3 será:

$$e_3 = 0{,}038 + 0{,}038 + 0{,}038 = 0{,}115$$
$$\gamma_3 = \frac{1}{2}\ln\left(\frac{1-0{,}115}{0{,}115}\right) = 1{,}021 \quad (19)$$

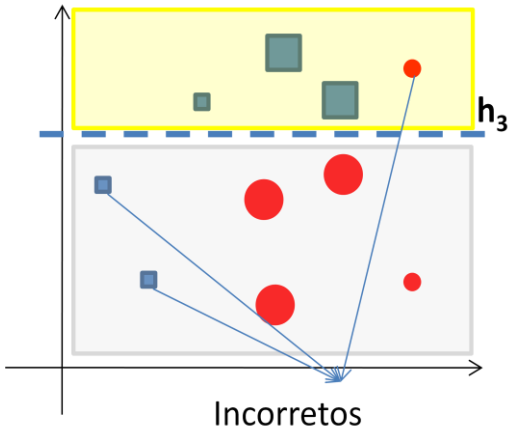

Figura 3.9 - Conjunto de treinamento - Iteração 3

Por fim, o classificador final H(x) é gerado a partir da combinação dos classificadores fracos gerados em cada iteração.

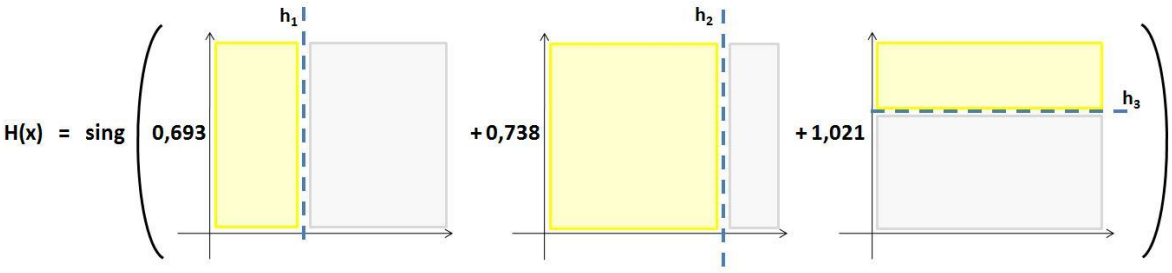

Figura 3.10 - Cálculo do classificador final H(x)

Portanto, retomando a distribuição inicial é possível verificar que os exemplos foram classificados de acordo com as classes da seguinte maneira.

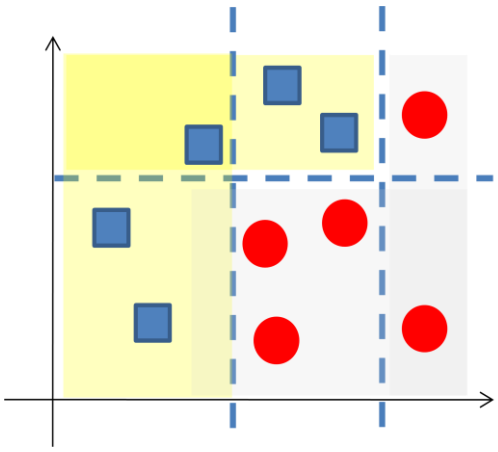

Figura 3.11 - Classificação após treinamento

3.4 Estruturas de AdaBoost para treinar classificadores

Diferentes abordagens podem ser utilizadas durante o processo de treinamento de um classificador e essa escolha impacta diretamente nos resultados obtidos. Cada estrutura tem sua forma particular de lidar com os níveis de independência e separação entre as diferentes iterações do *Boosting*, quais exemplos de treinamento serão apresentados em cada etapa, e por fim, como os classificadores fracos são combinados de forma a gerar a hipótese final. (SUSNJAK, 2009).

Como esperado, cada configuração apresenta vantagens e desvantagens. Assim, é necessário analisar qual estrutura é mais apropriada para cada problema específico. Existem diversas formas de se comparar as diferentes estruturas e duas delas serão apresentadas a seguir.

O primeiro aspecto importante a ser considerado para escolher qual configuração será utilizada refere-se à eficácia. Em outras palavras, qual o número de classificadores fracos que devem ser gerados durante o processo de treinamento para que o erro de treinamento estipulado seja atingido. Quanto menor for a quantidade de classificadores fracos, mais rápida e consequentemente eficaz é a estrutura. Esse parâmetro é especialmente importante nos casos em que é desejado embarcar classificadores em dispositivos com memória disponível limitada. Nestes casos, também é importante optar pela estrutura que apresente menor número de classificadores fracos.

O segundo aspecto refere-se aos resultados gerados, ou seja, analisar qual estrutura fornece melhores resultados, minimiza os falsos positivos e maximiza os índices de hit ratio.

As estruturas de *Boosting* mais populares são Monolítica (FREUND; SCHAPIRE, 1999), Cascata (VIOLA; JONES, 2004), Paralela (SCHNEIDERMAN; KANADE, 2000) e Árvore (LIENHART; LIANG; KURANOV, 2003). As duas primeiras estruturas serão detalhadas nos tópicos a seguir.

3.4.1 Estrutura Monolítica

A estrutura monolítica consiste em organizar os classificadores fracos em uma única camada. Essa foi a primeira estratégia para o processo *Boosting* utilizada e é o processo explicado em detalhes na Seção 3.2 deste trabalho. O principal objetivo da estrutura monolítica é minimizar o erro total e isso é realizado por meio da geração de classificadores fracos horizontalmente até que a taxa máxima de erro de treinamento estipulada seja atingida.

Essa configuração utiliza um processo iterativo que foca esforços nos exemplos mais difíceis de serem classificados e para isso, fornece pesos para cada exemplo de treinamento que são reavaliados a cada iteração.

Como mostrado anteriormente, o diagrama a seguir (Figura 3.12) resume o funcionamento dessa estrutura.

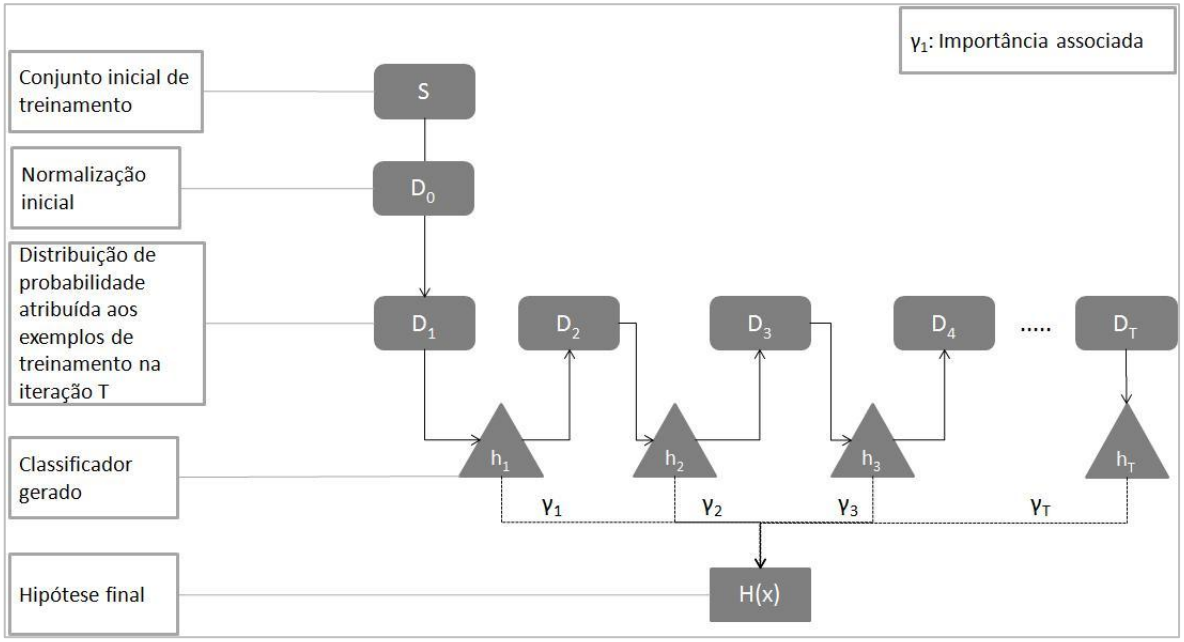

Figura 3.12 – Diagrama da estrutura monolítica

3.4.2 Estrutura Cascata

Introduzida por Viola e Jones no início deste século (VIOLA; JONES, 2001), essa estrutura divide o processo de treinamento em camadas. Dessa forma, o processo de treinamento é simplificado e uma das conseqüências importantes é a possibilidade de se utilizar essa estrutura em dispositivos que trabalham em tempo real, como por exemplo, o sistema de detecção de face em tempo real. A Figura 3.13 ilustra a configuração da estrutura cascata.

Cada iteração dessa estrutura corresponde à geração de um classificador fraco. Consequentemente, cada camada representa uma rodada completa da estrutura monolítica do AdaBoost apresentada na seção anterior. A estrutura cascata prioriza a minimização dos falsos positivos, diferentemente da estrutura monolítica que tenta minimizar o erro total.

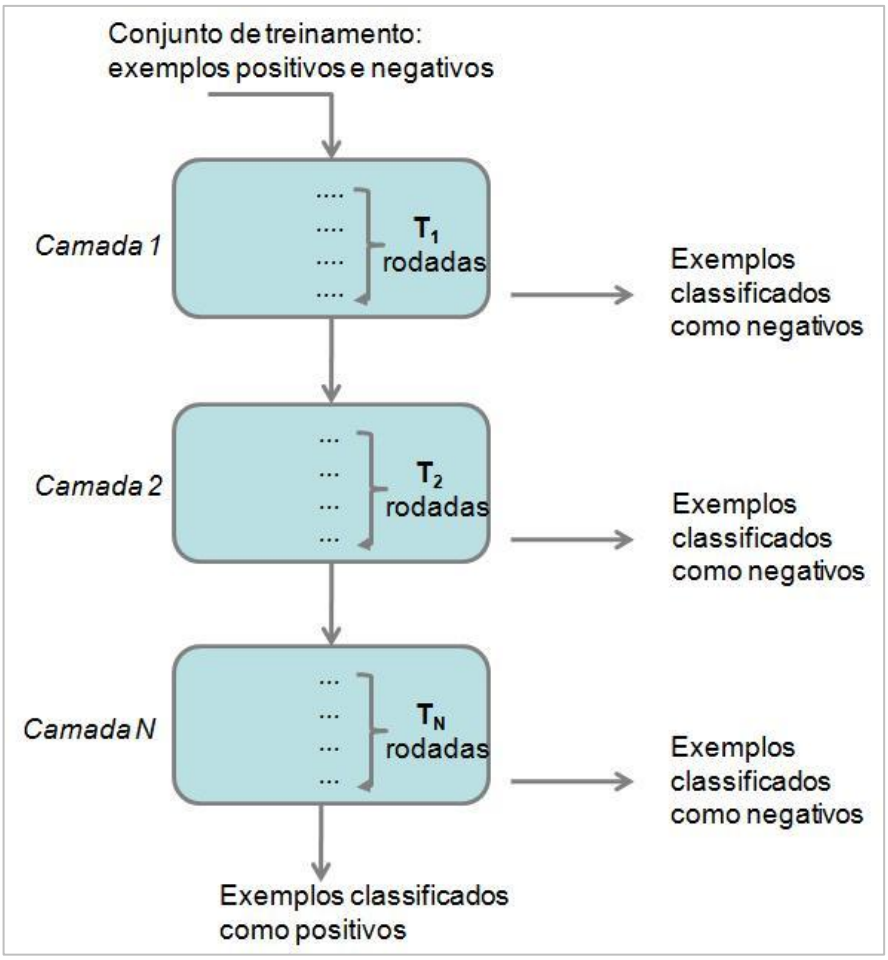

Figura 3.13 - Diagrama da estrutura cascata

Cada camada recebe uma meta simplificada para atingir. Por exemplo, mínima taxa de classificações corretas dos exemplos positivos é 99% e a máxima taxa de falsos positivos por camada é 50%, ou seja, pelo menos 50% dos exemplos negativos de treinamento que entram na camada são eliminados. Os exemplos restantes seguem para a próxima camada e o processo se inicia novamente. Como resultado, essa estrutura força o algoritmo de forma crescente a focar nos exemplos difíceis de serem classificados, ou seja, os exemplos não eliminados nas camadas anteriores. No final do processo, a taxa de exemplos positivos classificados corretamente (Hit ratio) e a taxa de falsos positivos (false positives) do classificador forte final se resume ao produto das respectivas taxas obtidas em cada camada (SUSNJAK, 2009).

A estrutura cascata acelera o processo de treinamento devido a sua configuração de eliminar exemplos em cada camada. Assim, é esperado que esta estrutura forneça menos classificadores fracos durante todo o processo se comparado à estrutura monolítica. No

entanto, esse desempenho depende diretamente do caso analisado e, portanto, não é uma regra.

3.5 Variações do AdaBoost

Após o AdaBoost original ter sido apresentado aos cientistas e pesquisadores, muitas variações e extensões deste algoritmo foram propostas e desenvolvidas como alternativas para fornecerem melhores resultados para problemas diversos.

Os problemas binários, ou seja, que existem somente duas classes para que os exemplos sejam classificados, foram os primeiros a serem pesquisados. Diversas variações para resolver problemas dessa ordem foram desenvolvidas, como o *Discrete AdaBoost* (FREUND; SCHAPIRE, 1996), *Real AdaBoost* (SCHAPIRE; SINGER, 1999), *Gentle AdaBoost* (FRIEDMAN; HASTIE;TIBSHIRANI, 2000) e o *Modest AdaBoost* (VEZHNEVETS; VEZHNEVETS, 2005).

Em muitas tarefas, porém, o exemplo pode ser classificado dentre diversas classes ao invés de apenas duas. Esses problemas, denominados multi-classes, são enfrentados por outras variações do AdaBoost. Pode-se destacar: AdaBoost.MH e AdaBoost.MR (SCHAPIRE; SINGER, 1999), LogitBoost (FRIEDMAN; HASTIE; TIBSHIRANI, 2000), além do AdaBoost.M1 e AdaBoost.M2 que serão brevemente introduzidos.

AdaBoost.M1

Considerada a mais simples dentre as versões do *Adaptive Boosting*, o AdaBoost.M1 também foi sugerido pelos pesquisadores Yoav Freund e Robert E. Schapire. Os autores provaram também que caso o erro do classificador fraco seja somente ligeiramente maior que ½, o erro de treinamento da hipótese final H(x) diminui de forma exponencial (FREUND; SCHAPIRE, 1996). O AdaBoost.M1 é praticamente o mesmo algoritmo apresentado na Figura 3.3, com exceção de que os algoritmos base podem ser classificadores de multi-classes ao invés de serem binários. Este foi o modelo adotado para a realização dos experimentos dessa pesquisa.

AdaBoost.M2

É similar à versão do AdaBoost.M1, sendo capaz também de resolver problemas de multi-classes, e foi desenvolvido pelos mesmos cientistas no mesmo ano. A diferença encontra-se na forma como trabalham as multi-classes. No AdaBoost.M1 o peso do classificador fraco é

função do erro calculado em relação a todos os exemplos da base. Por outro lado, no AdaBoost.M2 o peso do classificador fraco é função do erro calculado somente em relação aos exemplos classificados incorretamente. Dessa forma, o objetivo não é focar apenas nos exemplos difíceis de serem classificados como ocorre no AdaBoost.M1, mas sim nas classes utilizadas de forma incorreta durante a classificação e que portanto são difíceis de serem discriminadas (FREUND; SCHAPIRE, 1996).

3.6 Aplicações em diferentes áreas

A utilização de algoritmos de reconhecimento de padrão já pode ser encontrada não somente em projetos de alta tecnologia, mas também em produtos comerciais mais acessíveis como, por exemplo, em câmeras fotográficas digitais.

Disponível atualmente em máquinas de diferentes fabricantes, a tecnologia *Smile Shutter* é um exemplo de utilização desse princípio. Ao reconhecer o padrão de uma pessoa sorrindo na imagem focada, a máquina fotografa automaticamente sem que seja necessária nenhuma intervenção do portador da câmera. Estudos para essa finalidade já foram publicados utilizando algoritmos como Vetores de Suporte (SVM) e GentleBoost (WHITEHILL et al., 2007), porém, o algoritmo AdaBoost também apresenta um grande potencial para essa aplicação.

Por outro lado, devido a sua versatilidade e qualidade dos dados gerados, o algoritmo AdaBoost já apresenta aplicações em diversas áreas de estudo. Esses casos ocorrem tanto de forma comercial, disponíveis em patentes, como em estudos acadêmicos. Alguns exemplos são destacados a seguir.

3.6.1 Reconhecimento de Imagens

As primeiras aplicações do AdaBoost realizadas de forma mais aprofundada até os dias de hoje são para o reconhecimento de imagens para diversas finalidades. O reconhecimento de imagens é um trabalho que envolve um grande número de dados de forma simultânea e, devido aos pontos fortes desse algoritmo já apresentados neste trabalho, esses experimentos têm apresentado resultados muito positivos. Alguns exemplos são: detecções de faces (VIOLA; JONES, 2004; CHEN; NICPONSKI; RAY, 2004), auxílio para máquinas e robôs se locomoverem de forma autônoma prevendo possíveis impactos (BREED, 2005),

reconhecimento e classificação de elementos (AVIDAN, 2006), trabalhos com superposição de imagens (AVIDAN; BUTMAN; BUTMAN, 2006), técnicas para comparar duas imagens diferentes (LI et al., 2005), reconhecimento de manuscritos, entre outros.

3.6.2 Meio corporativo

Creamer e Freund (CREAMER; FREUND, 2005) realizaram um estudo sobre a utilização do algoritmo para auxiliar na análise do Balance Scored Card (BSC) e desenvolvimento de ações a partir dos dados presentes nesse documento. Segundo os autores, a aplicação foi comparada com outras metodologias como Bagging e Random Forest e os resultados foram promissores.

Além disso, o AdaBoost apresenta grande potencial nessa área já que outras técnicas de Inteligência Artificial, principalmente as Redes Neurais, são bastante utilizadas como na análise do mercado de ações, por exemplo.

3.6.3 Biomedicina

A grande quantidade de projetos multi-disciplinares nesse campo da ciência é um dos fatores da acelerada taxa de crescimento das pesquisas e descobertas. Devido ao grande volume de informação que esses projetos envolvem, muitas pesquisas usufruem também das técnicas da Inteligência Artificial para diversas finalidades, como por exemplo, para o descobrimento de novas drogas e cura de doenças de origem genética. Estudos utilizando o algoritmo AdaBoost já foram publicados, dentre eles, sobre auxílio em análises e diagnósticos médicos baseados em anticorpos (ITOH et al., 2007), predição e estatísticas de diagnósticos sobre esclerose (WANG; CHAKRABORTY; COMANICIU, 2006) e estudos e mapeamentos de genes (LESLIE et al., 2004; NABATAME; IBA, 2006).

3.6.4 Música

Já é possível encontrar publicações sobre a utilização de técnicas de inteligência artificial e, em especial o AdaBoost, para reconhecer padrões e classificar músicas de forma automática. As plataformas populares de listas de reproduções e vendas de músicas crescem cada vez mais e milhões de canções podem ser encontradas na loja iTunes da empresa americana Apple e também no Youtube, por exemplo. Por isso, é cada vez maior o desafio de

classificar as músicas pelo tipo e gênero. Assim, alguns estudos que utilizam o algoritmo AdaBoost para reconhecimento de padrões musicais para classificação e recomendação automática para os usuários já foram publicados (ECK et al., 2007; BERGSTRA et al., 2007; TRIPP; HUNG; PONTIKAKIS, 2006). Segundo os pesquisadores, os resultados obtidos até o momento apresentaram-se muito promissores.

4 Sensores

4.1 Sensores: Visão Geral

Devido aos sensores operarem baseados em diferentes princípios físicos fundamentais, o entendimento do funcionamento desses dispositivos geralmente requer um conhecimento interdisciplinar prévio tais como física, eletrônica, química e em alguns casos, biologia. Devido também a essa generalidade, existem diversas formas de classificação para sensores. Assim, o objetivo desse capítulo é apresentar algumas dessas formas de classificação e introduzir com mais detalhes os sensores capacitivos e sensores de fibra óptica (Seção 4.2). Será discutida também na Seção 4.3 uma das tendências para o futuro dos sensores, abordando os sensores inteligentes (*smart sensors*) e a fusão de sensores, além de discussão sobre benefícios da utilização de sistemas inteligentes junto a sensores (Seção 4.4).

4.2 Sensores: Tipos e classificações

De forma geral é possível definir sensores como dispositivos que têm a função de converter uma grandeza física - variável de entrada - em um sinal - variável de saída (WEBSTER, J. G., 1998). Assim, o objetivo principal dos sensores é responder a alguma forma de estímulo de entrada – quantidade, condição ou propriedade física a ser mensurada – e convertê-lo proporcionalmente em algum sinal elétrico de forma compatível com a interface ou circuito eletrônico a que está relacionado.

Esses dispositivos podem ser classificados em dois grupos: Sensores Ativos e Sensores Passivos. O primeiro grupo não gera uma tensão e, por isso, necessita de uma alimentação externa para então variar uma propriedade física e assim, fornecer o dado de saída ou resposta esperada. Já o segundo grupo, gera por si só um sinal elétrico em resposta a um estímulo (FRADEN, 1996).

Diversos conceitos envolvem o funcionamento dos sensores. Alguns são relacionados com a forma como o dispositivo irá operar, como sensibilidade, saturação e faixa de utilização. Já outros conceitos estão diretamente ligados aos erros das medições apresentadas pelo sensor, como erro de calibração, erro de histerese e reprodutibilidade.

A classificação de sensores em grupos pode ser realizada de diversas formas, desde elaboradas e complexas regras de semelhança até as mais simples. A tabela a seguir, adaptada

de (FRADEN, 1996; GARDNER, 1994), sugere quatro formas de classificação de sensores: material do sensor, fenômeno trabalhado, estímulo de entrada e áreas de aplicação.

Tabela 4- Algumas formas de classificação de sensores

Tipo de classificação	Grupos	Exemplos
Material do sensor	Inorgânico	
	Condutor	
	Semicondutor	
	Substância biológica	
	Orgânico	
	Outros...	
Fenômeno de conversão	Físico	Termoelétrico, fotoelétrico, fotomagnético, termoelástico, termomagnético
	Químico	Transformação química, processo eletroquímico
	Biológico	Transformação biológica, efeitos no organismo
Estímulo	Acústico	Amplitude de onda, espectro de fase, velocidade de onda
	Biológico	Concentração de açúcar, proteínas, hormônios, antígenos
	Químico	Umidade, concentração de gases, poluentes, íons
	Elétrico	Corrente, voltagem, resistência, capacitância, indutância, permissividade dielétrica, frequência
	Magnético	Campo magnético, condutividade, permeabilidade magnética
	Óptico	Fase, velocidade e amplitude da onda, índices de reflexão e refração
	Mecânico	Aceleração, velocidade, força, deslocamento, torque, pressão, massa, densidade, viscosidade
	Radiação	Tipo (raio X, infra-vermelho, onda de rádio), energia, intensidade
	Térmico	Temperatura, entropia, condutividade térmica
Áreas de aplicação	Automotivo	
	Espacial	
	Agricultura	
	Militar	
	Transporte	
	Domestico	
	Telecomunicações	
	Outros...	

Dentre os diversos tipos de sensores apresentados na Tabela 4, os sensores capacitivos e sensores de fibra óptica serão introduzidos a seguir.

4.2.1 Sensor Capacitivo

Como o próprio nome sugere, o princípio de funcionamento do sensor capacitivo é medir o parâmetro capacitância. Nesses sensores a substância a ser analisada faz o papel de dielétrico e preenchem os espaçamentos entre os eletrodos do capacitor. Assim, a constante dielétrica da substância influenciará diretamente no valor da capacitância registrado pelo sensor.

O primeiro cientista a pesquisar os efeitos do dielétrico em capacitores foi Faraday. Após pesquisas e experimentos, Faraday concluiu que quando o espaço entre as placas de um capacitor era preenchido por um dielétrico, o valor da capacitância detectado aumentava de um fator K, que é característico de cada material e que foi batizado por constante dielétrica (TIPLER; MOSCA, 1990). Essa constante é determinada em relação à permissividade do vácuo ε_o da seguinte forma:

$$\varepsilon = K * \varepsilon_0 \tag{20}$$

Sendo V_0 a tensão elétrica aplicada entre as placas do capacitor e Q_0 a carga formada nas superfícies internas das placas, como mostrado na figura a seguir, um campo elétrico E_0 é formado.

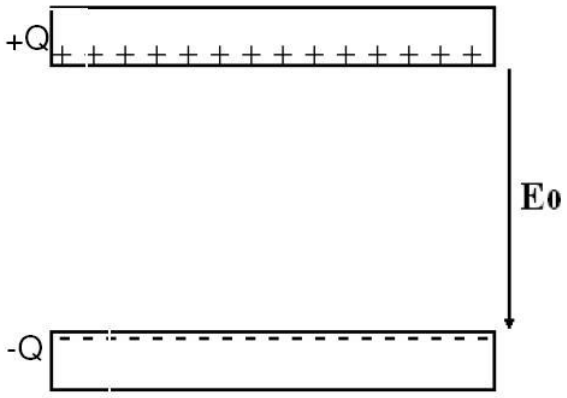

Figura 4.1 - Funcionamento de um capacitor

Assim, valor da capacitância em um capacitor sem dielétrico é dada por:

$$C_0 = \frac{Q_0}{V_0} \tag{21}$$

Portanto, quando há a presença de dielétrico entre as placas do capacitor, a constante dielétrica se relaciona com o valor da capacitância da seguinte forma:

$$C = K * C_0 \tag{22}$$

Como a constante dielétrica é uma característica do material, ela pode ser utilizada para identificação de uma substância ou mistura de substâncias. Uma das formas de se agregar valor aos sensores e utilizá-los em dispositivos cada vez mais autônomos é trabalhando os sinais captados por eles. Trabalhar os dados captados por um sensor capacitivo sobre a concentração das substâncias presentes em uma amostra de combustível, por exemplo, é de extrema utilidade. Utilizando metodologias e algoritmos específicos, é possível diminuir os erros presentes nas medições e trabalhar de forma conjunta os diversos dados captados pelo sensor. Além disso, nesse exemplo específico, é possível classificar a procedência do combustível, afirmar se é adulterado ou até mesmo servir de entrada para calibrar e melhorar a potência/eficiência do motor. É com esse objetivo que o estudo pretende utilizar o algoritmo AdaBoost.

Para o estudo de caso, serão utilizados os sensores capacitivos desenvolvidos por Lucas Gonçalves Dias Mendonça em sua dissertação de mestrado, neste Departamento, para analisar misturas de combustíveis (MENDONÇA, 2008). Os sensores foram fabricados por técnicas de microfabricação, tipo clássico utilizado em MEMS.

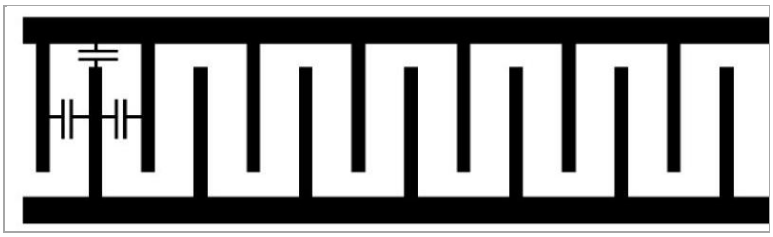

Figura 4.2 - Estrutura comb-drive (MENDONÇA, 2008)

Segundo o autor, optou-se por utilizar essa estrutura no sensor desenvolvido devido à possibilidade de o combustível a ser analisado preencher o espaçamento entre os eletrodos e, assim, permitir a detecção do valor de capacitância dos combustíveis analisados. A seguir tem-se uma figura que ilustra o dispositivo desenvolvido de forma completa.

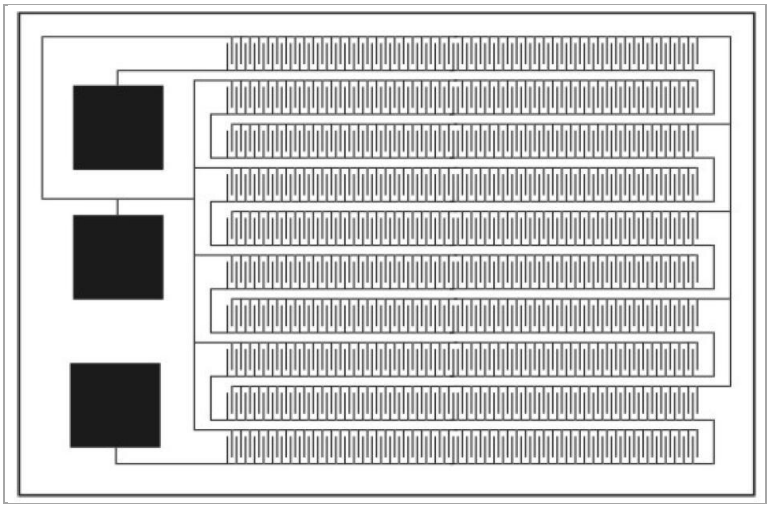

Figura 4.3 - Estrutura completa do sensor capacitivo (MENDONÇA, 2008)

4.2.2 Sensor de Fibra Óptica

Sensor de fibra óptica é atualmente tido como um tipo de sensor com grande potencial de crescimento e aplicação. Isso se deve principalmente ao fato de esses sensores poderem ser utilizados em condições consideradas mais severas como altas temperaturas, ambientes molhados, presença de taxas de vibração elevadas, ambientes potencialmente explosivos ou corrosivos.

O princípio de funcionamento do sensor é baseado no fenômeno da refletividade que ocorre na interface entre o sensor e o meio, líquido ou gasoso, em que se deseja analisar. A intensidade da luz refletida na ponta de prova é função da diferença entre o índice de refração do sensor e o do meio. Como o índice de refração de uma mistura é função dos índices das substâncias puras e de suas concentrações, é possível correlacionar a intensidade da luz refletida com a concentração da mistura (DALMON; GUSKEN; SUZUKI, 2007).

A refletividade é um fenômeno que ocorre quando um feixe de luz se move de um meio inicial para um segundo meio que apresentam índices de refração distintos. Parte do feixe de luz é refletida e outra parte refratada e essas proporções podem ser calculadas a partir do equacionamento de Fresnel (SALEH; TEICH, 1999). Sendo n_1 e n_2 os índices de refração reais dos meios e R a reflectância, tem-se que:

$$R = \left(\frac{n_2 - n_1}{n_2 + n_1}\right)^2 \tag{23}$$

Para um dos experimentos deste trabalho foi utilizado um sensor óptico desenvolvido no Laboratório de Materiais e Dispositivos Fotônicos, situado na Faculdade de Engenharia Mecânica da UNICAMP. O sensor consiste de um refletômetro à fibra óptica, cujo princípio operacional baseia-se na medição indireta do índice de refração da amostra através da intensidade luminosa refletida na interface entre a extremidade exposta da fibra e a amostra analisada.

4.3 Perspectiva para o futuro dos sensores

O sucesso da utilização de dispositivos para automatização das mais diversas áreas tem estimulado o mercado de sensores a buscar por inovações que forneçam mais funcionalidades e miniaturizações sem que se perca o foco no baixo custo de fabricação e do produto final. Além disso, os problemas enfrentados e pesquisados são cada vez mais complexos, envolvendo maior número de variáveis a serem mensuradas, propriedades analisadas e a uma maior velocidade de resposta requerida. Portanto, pode-se afirmar que o desafio atual para o crescimento do mercado de sensores é a procura pelo desenvolvimento de dispositivos menores, mais inteligentes e autônomos.

Nesse sentido, os sensores inteligentes - do inglês *smart sensors* - já estão presentes no mercado, porém tendem a crescer mais nos próximos anos. Apesar de não haver uma definição consagrada sobre o tema atualmente, pode-se afirmar que sensores inteligentes são dispositivos integrados com unidade de pré-processamento (Ex. filtro, amplificador) e com uma unidade de processamento (microprocessador composto por memória e algoritmo ou função), sendo todos os componentes presentes em um único chip (MAKINWA et al, 2007; MEIJER, 2008; FRENCH, 2010). De acordo com (GARDNER, 1994), um sensor inteligente deve ser capaz de realizar ao menos uma das três tarefas:

- Desenvolver uma função lógica com êxito;
- Se comunicar com um ou mais dispositivos;
- Tomar decisão por meio de métodos

Gardner mencionou também em sua obra de 1994 a expectativa da utilização de métodos inteligentes para realizar a tomada de decisão, como as redes neurais. Tal previsão é pesquisada e desenvolvida no presente.

A utilização de sistemas compostos por multisensores também ocorre atualmente e é esperado um crescimento para este mercado. A fusão de dados adquiridos por diferentes sensores fornece diversas vantagens se comparada a um único sensor, destacando-se três principais benefícios (HALL; LLINAS, 2001):

1. *Estatística:* combinação de maneira otimizada da resposta de sensores idênticos aprimora a resposta final e se aproxima da real mensuração do fenômeno observado;

2. *Relatividade:* colocação de sensores em posições ou velocidades diferentes pode enriquecer as informações do cenário analisado;

3. *Aprimorar o entendimento:* Sensores com finalidades ou princípios diferentes atuando paralelamente podem captar maior quantidade de informações, construindo um cenário mais próximo ao real.

Outra forma de compor sistemas com multisensores é por meio da utilização de maiores quantidades de um mesmo sensor para analisar o mesmo sistema. Por exemplo, mais sensores de temperatura ou capacitância utilizados de forma concomitante. Neste caso, os dois primeiros benefícios destacados acima também são válidos.

Este trabalho não tem a pretensão de investigar os benefícios dos sensores inteligentes e da fusão de sensores. No entanto, o conhecimento gerado pelo estudo está de alguma forma relacionado com essas pesquisas e poderá ser utilizado em alguns casos e avanços dessas áreas.

4.4 Benefícios da utilização de Aprendizagem de Máquina em sensores

Monitoração parcial ou simples das variáveis de entrada de um sistema obtidas diretamente pelo sensor gera uma “nuvem de dados”, ou seja, não é uma função claramente definida.

Desta forma, os dados obtidos por um único sensor ou mesmo a mensuração de uma única propriedade do sistema indicam apenas o parcial de sua influência no comportamento do sistema analisado. Em outras palavras, o sistema não é representado ou analisado de forma completa. Logo, pode-se afirmar que a função xy referente ao comportamento do sistema não

é perfeita, e sim difusa. Assim, de acordo com os pontos utilizados para analisar o sistema, pode resultar em uma conclusão equivocada sobre a tendência do sistema.

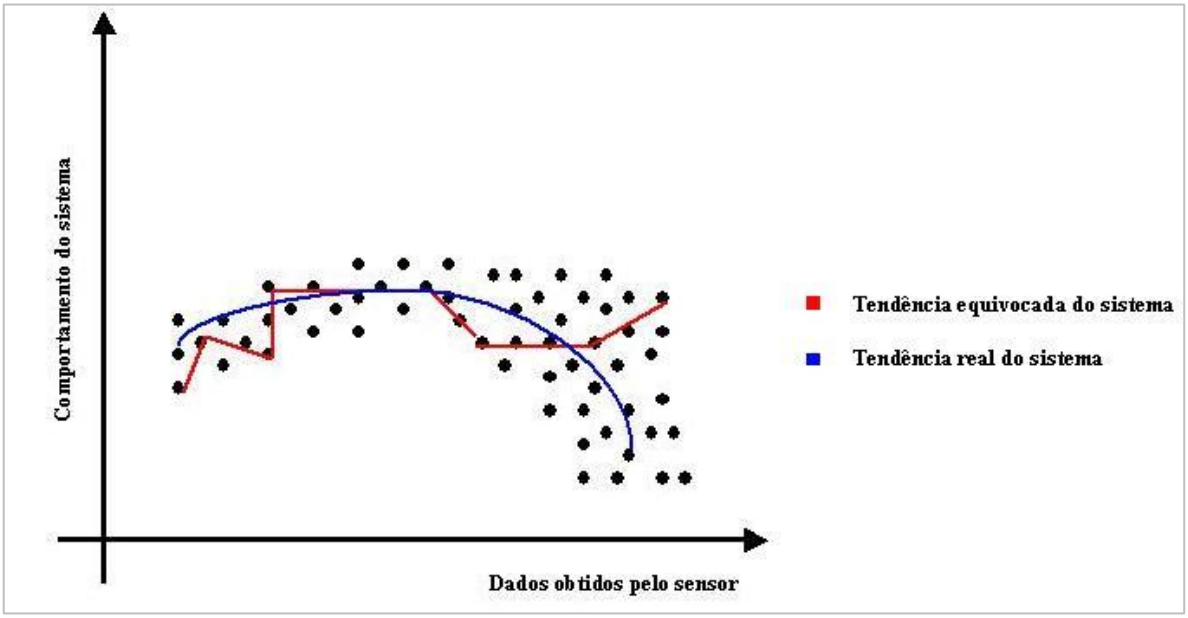

Figura 4.4 - Exemplo de possíveis erros de leitura sobre o sistema analisado

Esse comportamento difuso deve-se também aos ruídos existentes no mundo real e que influenciam os dados de entrada e de saída, não permitindo que a função final obtida seja perfeita.

Outros problemas que acarretam este tipo de comportamento são as chamadas medições "ruidosas", ou seja, aquelas que apresentam interferências. Por exemplo, uma propriedade pode interferir no comportamento de outra tornando-se necessário cruzar os dados de todas elas para se tirar uma conclusão do sistema com um todo. Por causa do meio, é natural que haja interferência entre sensores e entre fenômenos. Devido a esses motivos, os sensores geralmente fornecem informações incompletas, inconsistentes ou imprecisas sobre o sistema analisado.

Portanto, não é uma questão simples descobrir qual é a complexidade e forma da equação do comportamento do sistema. Devido à dificuldade de se responder essa pergunta é que os algoritmos de aprendizado de máquina podem auxiliar nas medições dos sensores ou redes de sensores.

Outro fator de grande potencial de contribuição que os algoritmos de aprendizado de máquina podem fornecer aos sensores é na etapa de calibração. A calibração é uma etapa de grande importância para garantir o funcionamento adequado do sensor. Uma das

contribuições que os algoritmos de aprendizado de máquina podem fornecer durante essa etapa é simplificar o processo.

Simplificar o processo de calibração é possível, pois a partir de uma amostra relativamente pequena de dados é possível abranger todos os grupos de classificações e precisão almejadas para o sensor. Assim, essa amostra pode ser utilizada para treinar o algoritmo para que este faça as futuras classificações de forma automática. Além disso, mesmo que a curva de calibração apresente não-linearidades ou formas de ordem elevada, o algoritmo de aprendizagem de máquina é capaz de mapear toda a curva sem elevar a complexidade da solução. Por outro lado, caso a calibração fosse realizada por meio de equações matemáticas, por exemplo, esses casos necessitariam de muito custo computacional para serem solucionados com a mesma qualidade.

Essa contribuição apresenta-se muito importante, pois se mostra flexível para ser utilizada junto a qualquer sensor ou rede de sensores, independente da propriedade física mensurada ou formato da curva de calibração.

5 Estudo de Caso – Adulteração de Combustível

5.1 Visão Geral

Com o objetivo de verificar a viabilidade deste projeto, foi escolhida uma aplicação prática de grande interesse comercial e também da comunidade acadêmica: a análise de adulterações de combustíveis automotivos. Outra motivação para a escolha desse tema para a realização dos experimentos refere-se à variedade de sensores utilizados para esse mesmo objetivo. A crescente quantidade de pesquisas sobre o tema permitiu que sensores com princípios distintos e até mesmo o uso concomitante de sensores possa ser utilizado. Neste capítulo será fornecida uma breve introdução sobre o tema.

5.2 Adulteração de Etanol Combustível Automotivo

O etanol teve iniciada sua utilização como combustível automotivo no Brasil na década de 1970. Pesquisas e investimentos impulsionaram o Brasil e atualmente o colocam em evidência mundial sobre este combustível automotivo, tanto na produção como na utilização do produto, pois iniciou-se naquele momento a procura por uma alternativa para o combustível proveniente do petróleo. Atualmente, o aumento do número de veículos portadores de tecnologia biocombustível também é parcialmente responsável pelo crescimento da comercialização de etanol combustível no país.

No país, dois tipos de etanol combustível são comercializados atualmente:

- Álcool Etílico Hidratado Combustível: conhecido por AEHC, este tipo é utilizado como alternativa à gasolina;
- Álcool Etílico Anidro Combustível: ou simplesmente AEAC, é utilizado para misturar com gasolina do tipo A com o objetivo de formar gasolina do tipo C.

Regras e legislações determinam que o AEHC contenha um teor alcoólico entre 92,6º e 93,8º INPM, ou de forma equivalente, concentração volumétrica entre 5,0% e 6,0% a uma temperatura de 20ºC de água diluída. No entanto, em alguns casos esses índices não são respeitados, de forma ilegal, prejudicando o bom funcionamento do veículo e o pagamento de impostos. A forma mais comum de adulteração do etanol é por meio da adição de água de

torneira, no AEAC para convertê-lo em AEHC ou no próprio AEHC com o objetivo de aumentar o volume total do produto a ser comercializado.

Assim, é crescente o interesse pelo desenvolvimento de tecnologia que permita a detecção de adulteração do etanol, seja para fiscalização nos estabelecimentos ou para prevenção nos próprios automóveis.

Portanto, essa é a motivação principal para a escolha de sensores com o objetivo específico de determinar as concentrações presentes no etanol combustível para a realização dos experimentos dessa pesquisa.

5.3 Software Utilizado

A evolução da área de extração de conhecimento de bases de dados e o progresso das pesquisas sobre o tema, tanto acadêmicas como comerciais, estimularam o desenvolvimento de diversos softwares capazes de auxiliar na resolução de problemas desse tipo. Algumas ferramentas são comerciais e, portanto, não são de acesso gratuito. No entanto, muitas ferramentas abertas – *open source* – e que fornecem ótimos resultados estão disponíveis no mercado. Algumas dessas ferramentas permitem inclusive que os pacotes de programação sejam acessados. A Tabela 5 fornece uma relação de algumas das mais conhecidas ferramentas existentes e disponíveis atualmente, incluindo programas, bibliotecas e pacotes matemáticos. Uma relação mais extensa e detalhada dessas ferramentas pode ser encontrada em (MIKUT; REISCHL, 2011).

Para este trabalho optou-se por utilizar o programa WEKA para avaliar a estrutura monolítica do algoritmo AdaBoost. Desenvolvido em linguagem Java pela Universidade de Waikato, na Nova Zelândia, o WEKA – *Waikato Environment for Knowledge Analysis* – é um software que se caracteriza por ser flexível, podendo assim funcionar em diversas plataformas. É um software de domínio público e que disponibiliza diversos algoritmos de Inteligência Artificial. A obra (WITTEN; FRANK, 2005) fornece informações detalhadas sobre como utilizar as diversas funcionalidades do programa.

Tabela 5 - Lista de algumas ferramentas disponíveis para Mineração de Dados

Nome	Link	Tipo de aplicativo
ADAPA (Zementis)	www.zementis.com	Comercial
CART	www.salford-systems.com	Comercial
IBM SPSS Modeler	www.spss.com/software/modeling/modeler	Comercial
IBM SPSS Statistics	www.spss.com/software/statistics	Comercial
ImageJ	rsbweb.nih.gov/ij	Livre
ITK	www.itk.org	Livre
KNIME	www.knime.org	Livre
KXEN	www.kxen.com	Comercial
MATLAB	www.mathworks.com	Comercial
Oracle Data Mining (ODM)	www.oracle.com/technology/products/bi/odm/index.html	Comercial
Orange	www.ailab.si/orange	Livre
Pentaho	sourceforge.net/projects/pentaho	Livre
R	www.r-project.org	Livre
RapidMiner	www.rapidminer.com	Livre
SAP Netweaver Business Warehouse (BW)	www.sap.com/platform/netweaver/components/businesswarehouse	Comercial
SAS Enterprise Miner	www.sas.com/products/miner	Comercial
SQL Server Analysis Service	www.microsoft.com/sql	Comercial
STATISTICA	www.statsoft.com/products/data-mining-solutions/G259	Comercial
Teradata Database	www.teradata.com	Comercial
TIBCO Spotfire	spotfire.tibco.com	Comercial
WEKA	http://www.cs.waikato.ac.nz/ml/weka/	Livre

Nos casos de avaliação da estrutura em cascata do algoritmo AdaBoost, foi utilizada uma implementação própria desenvolvida, em linguagem C++, na Massey University da estrutura de Cascata junto com *Decision Stump* como classificador fraco (BARCZAK, JOHNSON, MESSOM, 2008), seguindo a metodologia proposta por Viola, Jones, 2001.

6 Procedimento Experimental, Resultados e Análises

6.1 Visão Geral

Seis experimentos independentes foram realizados com o objetivo de avaliar a aplicação do algoritmo *Adaptive Boosting* junto a sensores de princípios físicos distintos e também em diferentes cenários. A seguir serão descritos os sete experimentos, a forma como foram obtidos e trabalhados os dados, além de apresentar e discutir sobre os resultados obtidos.

6.2 Estudo de caso 1: Sensor Capacitivo

6.2.1 Aspectos Gerais

Na primeira experiência o objetivo principal foi verificar a possibilidade de se utilizar o algoritmo AdaBoost para abordagem de problemas de classificação de dados obtidos por sensores. A estrutura monolítica foi escolhida para ser utilizada nessa primeira experiência. Para a experiência aqui descrita, foi utilizado um micro-sensor capacitivo construído por técnicas de microfabricação sobre um substrato de alumina e em configuração de eletrodos interdigitados. O sensor em questão foi desenvolvido em trabalho anterior (MENDONÇA, 2008) e já estava disponível no laboratório para ser utilizado. Quando totalmente submerso na amostra de etanol combustível, o sensor utiliza o líquido como dielétrico e, assim, é capaz de mensurar a capacitância da amostra combustível mais o substrato. Por sua vez, tendo conhecido o valor de capacitância, é possível mensurar as concentrações de água e álcool presentes na mistura.

Nessa primeira experiência foram utilizados dois algoritmos base para a análise e classificação dos dados: *Decision Stump* e *Random Forest*. Já o algoritmo principal foi o AdaBoost.M1.

6.2.2 Obtenção dos dados

Os dados foram coletados pelo autor do trabalho em parceria com pesquisadores do laboratório, sendo um dos autores da publicação (MENDONÇA et al., 2010).

Com o objetivo de analisar amostras conformes e não conformes em relação ao padrão adotado no país (Seção 5.2), foram escolhidas oito concentrações diferentes. Dentre elas, três concentrações respeitam as normas, duas concentrações apresentam quantidade de água diluída na mistura inferior ao permitido, enquanto as outras três concentrações restantes apresentam quantidade de água acima do permitido.

Por meio da utilização de duas buretas, uma delas contendo álcool anidro e a outra contendo água deionizada, foi possível preparar as amostras presentes na Tabela 6. Todo o procedimento foi realizado por três vezes consecutivas como forma de minimizar possíveis erros de preparação presentes nos dados coletados. Devido à menor escala das buretas serem de 0,05 ml, foi necessário fazer concentrações extremamente próximas à programada, porém não exatas. No entanto, para efeito de análise dos dados com o algoritmo AdaBoost, foi adotado o valor exato de cada concentração programada. A Tabela 6 a seguir resume as informações.

Tabela 6 – Concentrações utilizadas na experiência 1

Volume de álcool anidro presente na mistura (ml)	Volume de água presente na mistura (ml)	Concentração volumétrica de água em etanol exata (%)	Concentração volumétrica de água em etanol adotada para a análise (%)	Classificação da mistura
50	2,10	4,03	4,0	Não conforme
50	2,35	4,49	4,5	Não conforme
50	2,65	5,03	5,0	Conforme
50	2,90	5,48	5,5	Conforme
50	3,20	6,02	6,0	Conforme
50	3,50	6,54	6,5	Não conforme
50	3,80	7,06	7,0	Não conforme
50	4,40	8,09	8,0	Não conforme

A cada preparação de concentração o sensor era imerso e os dados aquisitados por meio de um medidor de impedância modelo PM6306 da marca FLUKE, tendo comunicação entre este aparelho e um computador desktop utilizando-se uma porta serial. Por sua vez, os dois fios do sensor capacitivo foram conectados ao aparelho. Desta forma, as propriedades das misturas medidas pelo aparelho foram capacitância, resistência e ângulo de fase.

6.2.3 Análise dos Dados

Para cada uma das concentrações da Tabela 6 foram coletados 600 dados de leitura na forma de vetor (capacitância, ângulo de fase, resistência), sendo 200 dados por ciclo de preparação. Portanto a base de dados é constituída por 4800 exemplos. A Tabela 7 a seguir resume as quantidades de exemplos por medição e por concentração.

Tabela 7 – Quantidade de exemplos por concentração

Concentrações	4,0%	4,5%	5,0%	5,5%	6,0%	6,5%	7,0%	8,0%
1ª Medição	200	200	200	200	200	200	200	200
2ª Medição	200	200	200	200	200	200	200	200
3ª Medição	200	200	200	200	200	200	200	200

Utilizando o programa Excel do pacote Microsoft Office 2007 foi calculada a média de cada atributo (capacitância, resistência e ângulo de fase) em cada grupo de 200 exemplos (Medição X Concentração). Os gráficos a seguir demonstram essas distribuições para cada um dos atributos.

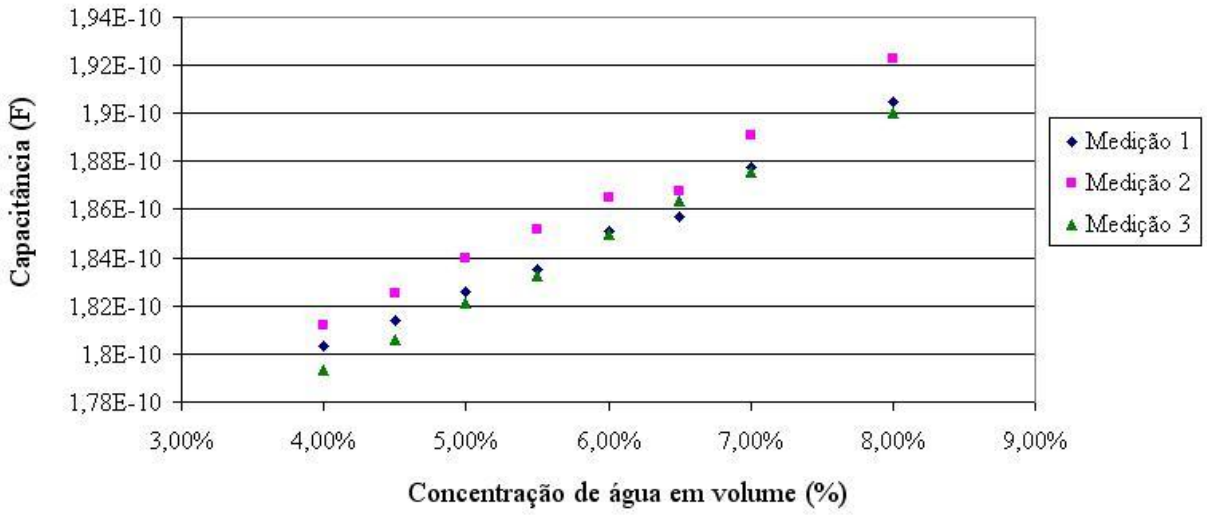

Figura 6.1 – Distribuição da média dos valores de Capacitância

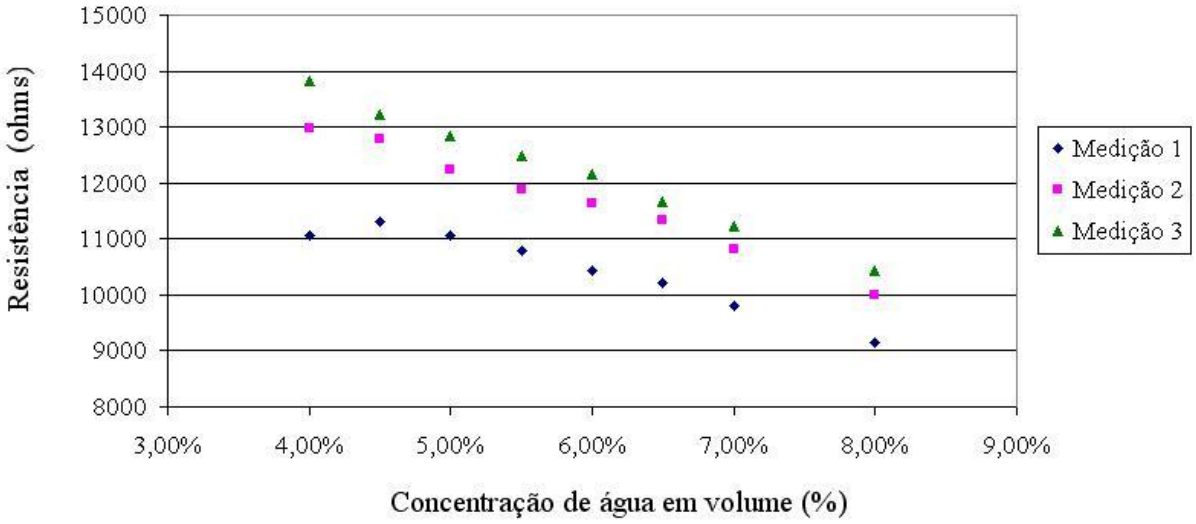

Figura 6.2 – Distribuição da média dos valores de Resistência

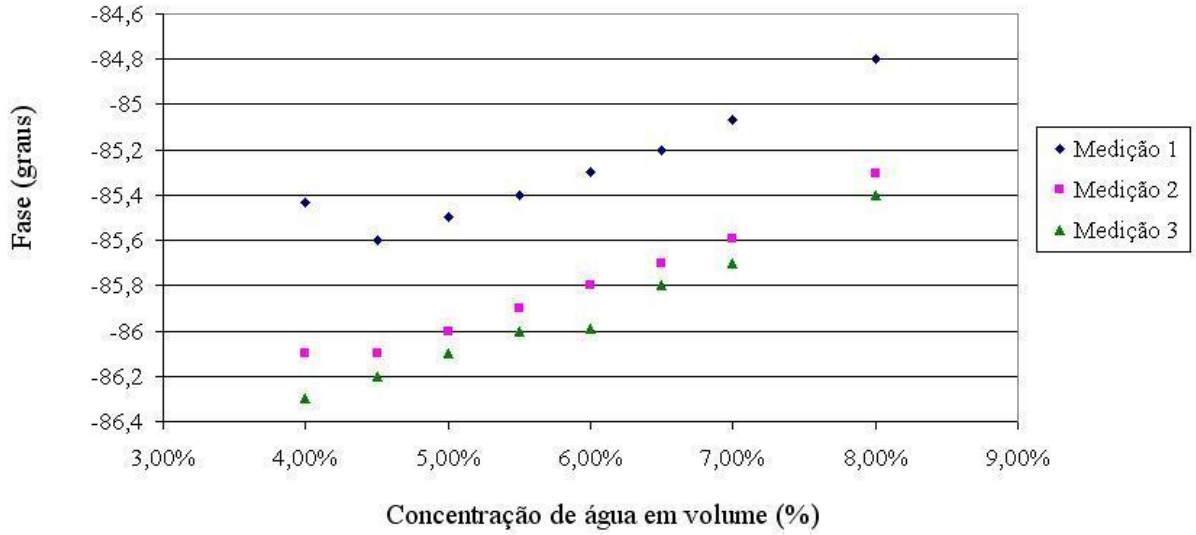

Figura 6.3 – Distribuição da média dos valores de ângulo de Fase

Ao analisar e comparar os valores entre as medições realizadas, ou seja, os pontos entre os 600 exemplos para cada concentração é possível verificar que um mesmo valor de capacitância, ângulo de fase ou resistência pode englobar diferentes valores de concentrações. Por exemplo, o valor de -85,6º de ângulo de fase (Figura 6.3) pode representar a concentração de 4,5% ou 7,0% de água em volume na mistura. A partir da Figura 6.1 nota-se que o valor de capacitância aproximadamente igual a 183 pF pode representar 4,5%, 5,0% ou 5,5% de água em volume na mistura. O gráfico a seguir ilustra essa distribuição do atributo Capacitância. Foi realizada uma média dos valores obtidos nas três medições para a construção demonstrada na Figura 6.4 abaixo.

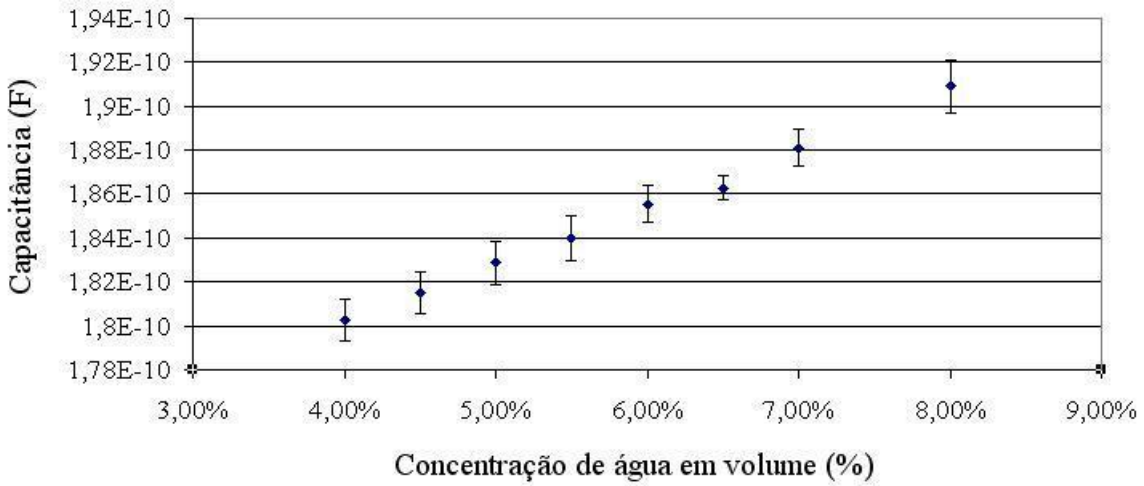

Figura 6.4 – Distribuição dos valores de Capacitância obtidos nas 3 medições

Analisando o gráfico e focando na região permitida (entre 5,0% e 6,0%) verifica-se que há grande sobreposição de possíveis valores de capacitância, tendo inclusive valores obtidos em concentrações da região não permitida (4,5% e 6,5%) que se sobrepõem à faixa permitida. Essa sobreposição torna a classificação de combustível em dois grupos, conforme e não conforme, a ser realizada pelo AdaBoost um problema não trivial. Caso a classe do problema seja a concentração da amostra, ou seja, o algoritmo tenha que classificar as amostras por concentrações, a sobreposição entre esses 8 possíveis grupos torna o problema ainda mais complexo.

6.2.4 Pré-processamento dos dados e extração de padrões

Antes de serem utilizados, os conjuntos presentes na base de dados foram pré-processados de forma a serem compatíveis com o programa WEKA e também organizados para as análises almejadas. A base de dados consiste em um arquivo de texto. Um exemplo de arquivo apropriado para ser utilizado com o programa WEKA pode ser observado no Apêndice A deste trabalho. A mesma base de dados foi utilizada para a realização de duas análises. Inicialmente os dados foram trabalhados em documento Excel e depois transformados em documentos de texto.

Na primeira delas o objetivo era classificar os exemplos em dois grupos: Conforme e Não-conforme. Assim, cada exemplo foi composto por 3 atributos (Capacitância, Resistência e Ângulo de Fase) e a classe associada a cada exemplo, ou seja, o grupo a que pertence, foi inserido manualmente. Utilizando a função *Aleatório*() do programa Excel, exemplos da base

de dados foram escolhidos para compor o grupo de exemplos de treinamento. O restante dos exemplos foi utilizado como dados de teste. Foram preparados 6 arquivos, com a distribuição explicada na Tabela 8 abaixo.

Tabela 8 – Distribuição de exemplos em treinamento e teste

Nº de exemplos na base de treinamento	Nº de exemplos na base de teste
20	4780
40	4760
60	4740
80	4720
100	4700

Esse procedimento foi repetido para a segunda análise em que o objetivo era classificar os exemplos de acordo com a concentração da amostra. Portanto, nesse caso são 8 possíveis classes, que também foram inseridas na tabela de forma manual. A mesma quantidade de arquivos foi gerada e com as mesmas distribuições de exemplos entre dados de treinamento e teste.

6.2.5 Avaliação dos resultados obtidos

Conforme descrito na Seção 6.2.1, foi utilizado o algoritmo AdaBoost.M1 como algoritmo principal e os algoritmos *Decision Stump* e *Random Forest* para gerar os classificadores base ou fracos.

Uma análise mais detalhada do comportamento dos algoritmos pode ser realizada observando-se a curva de aprendizado obtida, ou seja, o índice de acertos obtidos de acordo com a quantidade de exemplos fornecidos para o treinamento.

Os gráficos das Figuras 6.5 e 6.6 a seguir mostram as curvas de aprendizado obtidas utilizando como algoritmo base o *Decision Stump* e o *Random Forest*, respectivamente, tendo como objetivo a classificação do problema binário nas classes Conforme e Não-conforme em relação à concentração de água no álcool combustível.

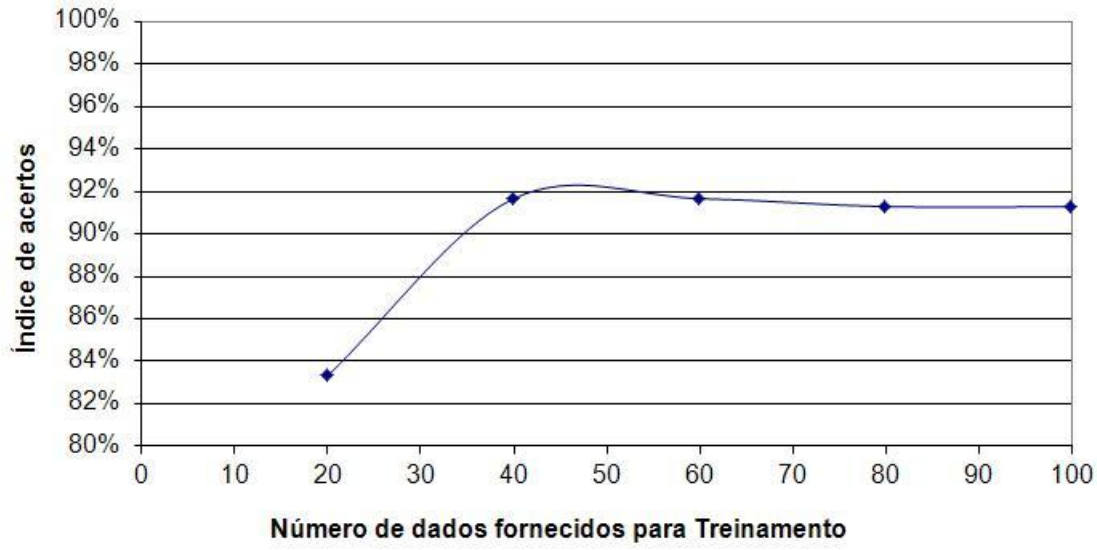

Figura 6.5 – Resposta de problema binário utilizando *Decision Stump* como algoritmo base

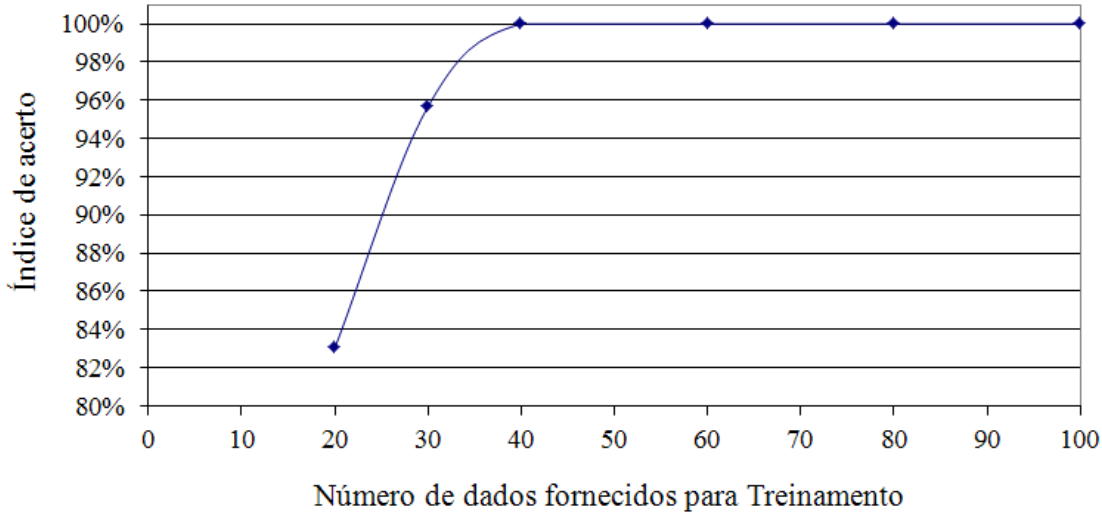

Figura 6.6 - Resposta de problema binário utilizando *Random Forest* como algoritmo base

O primeiro gráfico obtido com o algoritmo base *Decision Stump* apresentou saturação em 91,9% de classificações corretas, obtido a partir de uma amostra de 40 exemplos de treinamento. Estima-se que a saturação nesse patamar tenha ocorrido devido à simplicidade do algoritmo base e que por meio de uma investigação mais aprofundada desse algoritmo seja possível alterar parâmetros específicos para que o índice de acertos apresente uma melhora.

No caso do Random Forest, o algoritmo AdaBoost não foi requisitado, tendo o problema sido resolvido apenas com o algoritmo base. Em outras palavras, o primeiro classificador fraco fornecido pelo Random Forest na primeira iteração do AdaBoost já foi capaz de minimizar os erros de classificação, não necessitando de novas iterações, quando o classificador foi submetido ao conjunto de teste. Essa ocorrência provavelmente deve-se ao

fato de haver uma linearidade nos dados coletados dos parâmetros analisados e assim o cenário pode ser analisado sem a necessidade do algoritmo AdaBoost. No entanto, observa-se mais uma vez que o maior índice de classificação foi atingido a partir de 40 exemplos de treinamento.

Os resultados obtidos na análise desse primeiro cenário indicam ser possível acelerar a etapa de calibração dos sensores, podendo esta ser realizada baseada em um algoritmo de aprendizagem de máquina a partir de poucos exemplos de treinamento.

Em uma segunda análise, tendo como objetivo a classificação do problema multiclasse em relação à concentração de água no álcool combustível. As possíveis classes foram: 4,0%; 4,5%; 5,0%; 5,5%; 6,0%; 6,5%; 7,0%; 8,0%. Para essa análise foi utilizado somente o algoritmo Random Forest como algoritmo base. O gráfico a seguir ilustra a curva de aprendizado obtida.

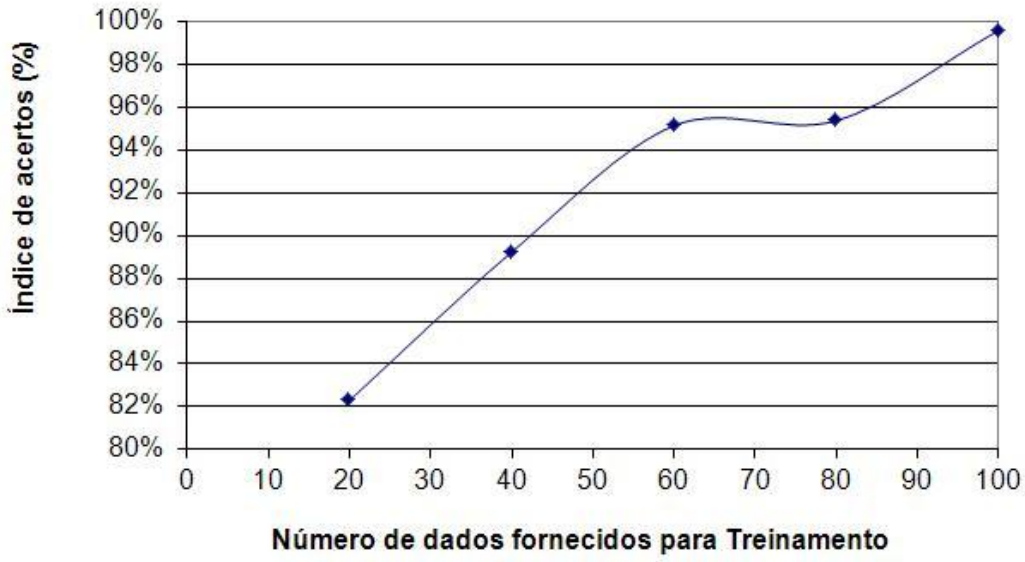

Figura 6.7 - Resposta de problema multiclasse utilizando Random Forest como algoritmo base

Assim como na análise anterior, índices de acertos elevados foram obtidos para o cenário estudado. Mais uma vez o algoritmo AdaBoost não foi requisitado e assim, o algoritmo Random Forest realizou as classificações de forma independente.

A curva de aprendizado obtida apresentou menor inclinação, atingindo índices de acerto próximos a 100% apenas com 100 exemplos de treinamento. Porém, ainda assim essa abordagem de classificação das amostras em grupos de concentração de combustível mostrou-se viável para calibrar o sensor de forma rápida com poucos exemplos iniciais e também com uma precisão muito significativa por meio de algoritmos de aprendizagem de máquina.

6.3 Estudo de caso 2: Conjunto de Sensores Capacitivos e de Temperatura

6.3.1 Aspectos Gerais

Para a segunda experiência, foi almejado aplicar o algoritmo AdaBoost com estrutura monolítica para analisar dados obtidos por uma rede de sensores. Nesse sentido, foram utilizados quatro micro-sensores capacitivos construídos por técnicas de microfabricação da mesma forma como o utilizado na primeira experiência. No entanto, os sensores utilizados nesse momento foram fabricados em substrato de fibra de vidro com eletrodos em cobre. Além disso, os quatro sensores apresentam estruturas semelhantes de eletrodos interdigitados, porém com *gaps* – distância entre dedos – distintas entre si. Dessa forma, apesar de os sensores apresentarem o mesmo princípio físico de funcionamento, podem ser considerados sensores diferentes e registram valores distintos de capacitância para uma mesma amostra analisada.

Além dos sensores capacitivos foi utilizado um sensor de temperatura LM35 fabricado pela National Semiconductor Corporation com o objetivo de verificar a influência da temperatura no problema de avaliação da qualidade de combustíveis.

Nessa segunda experiência o AdaBoost.M1 disponível no software WEKA foi utilizado junto a dois algoritmos base para a análise e classificação dos dados: J48 e *Random Forest*.

6.3.2 Obtenção dos dados

Os dados foram novamente coletados pelo autor do trabalho em parceria com pesquisadores do laboratório.

Três concentrações na zona crítica de conformidade da mistura de etanol combustível e água foram escolhidas para o experimento: 4,5%, 5,5% e 6,5% de concentração volumétrica de água em etanol combustível. Mais uma vez o processo de utilização das buretas descrito na seção 6.2.2 foi utilizado para a preparação das amostras.

A cada preparação de concentração os quatro sensores capacitivos e também o sensor de temperatura eram imersos na amostra e os dados aquisitados por meio de um microprocessador Arduino duemilanove, tendo comunicação com um computador desktop utilizando-se uma porta USB. Por sua vez, os sensores capacitivos foram conectados a um circuito composto por resistores e circuitos integrados para possibilitar a comunicação com o

microprocessador. Cada amostra foi aquecida e depois resfriada durante a etapa de coleta dos dados.

Desta forma, as propriedades das misturas mensuradas pelo circuito foram período coletado pelo sensor em (µs) e a temperatura da amostra em (ºC). No entanto, por meio de calculo matemático é possível obter o valor da capacitância em (pF) a partir do valor de período mensurado. Portanto, os três atributos acima foram utilizados no experimento, apesar da redundância matemática entre dois dos atributos.

6.3.3 Análise dos Dados

Como já foi dito, as três amostras preparadas foram aquecidas e depois resfriadas durante a coleta dos dados. Dessa forma, a base de dados possui 12637 exemplos e três atributos. Na tabela abaixo é apresentada a distribuição das informações da base de dados.

Tabela 9 – Distribuição dos exemplos coletados pelos sensores

Concentração volumétrica de água em etanol (%)	Exemplos
4,5%	4601
5,5%	4168
6,5%	3868
Total	12637

A temperatura da amostra de combustível apresentou grande influência nos valores de capacitância coletados pelos sensores. Essa influência pode ser observada, por exemplo, nos gráficos temporais obtidos pelo primeiro sensor capacitivo para as concentrações de 4,5%, 5,5% e 6,5% respectivamente ilustrados nas Figuras 6.8 a 6.13.

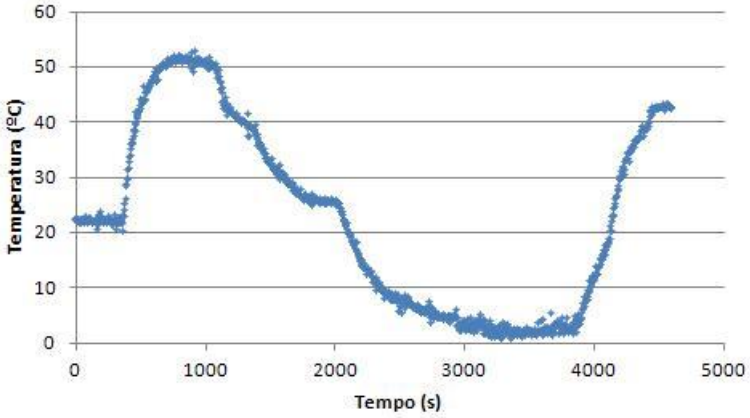

Figura 6.8 – Gráfico da temperatura no tempo obtido pelo sensor lm35 para concentração 4,5%

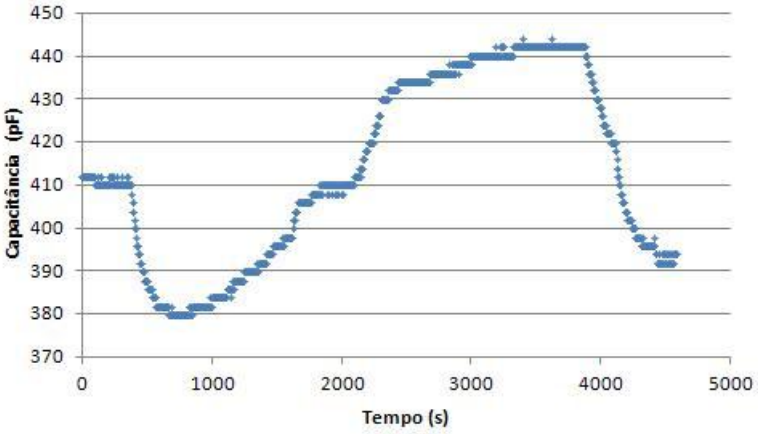

Figura 6.9 - Gráfico da capacitância no tempo obtido pelo sensor 1 para concentração 4,5%

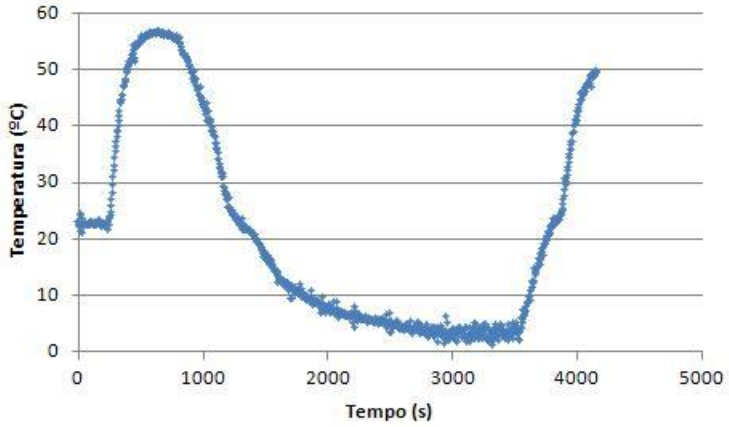

Figura 6.10 - Gráfico da temperatura no tempo obtido pelo sensor lm35 para concentração 5,5%

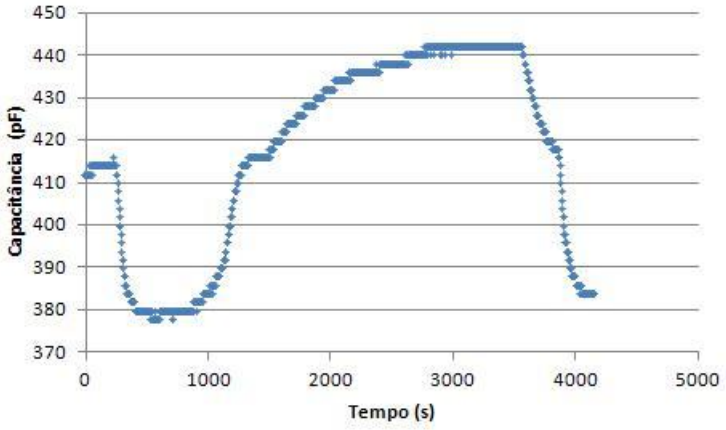

Figura 6.11 - Gráfico da capacitância no tempo obtido pelo sensor 1 para concentração 5,5%

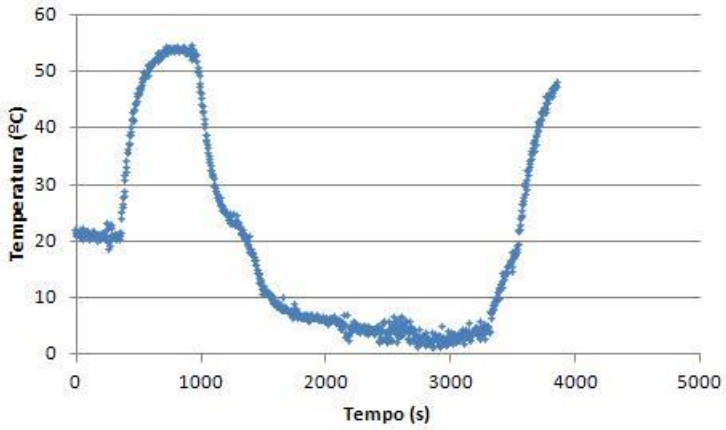

Figura 6.12 - Gráfico da temperatura no tempo obtido pelo sensor lm35 para concentração 6,5%

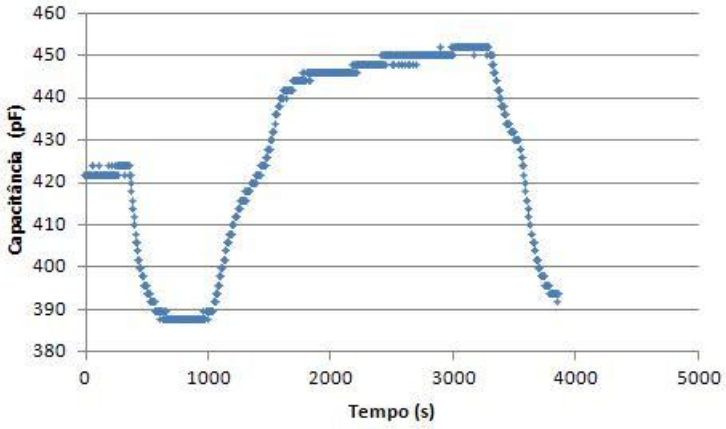

Figura 6.13 - Gráfico da capacitância no tempo obtido pelo sensor 1 para concentração 6,5%

Ao analisar os dados com enfoque nas concentrações preparadas, nota-se também que há grande sobreposição dos dados. O gráfico do sensor 1 a seguir (Figura 6.14) foi escolhido para mostrar que mais uma vez prováveis valores de capacitância para a mistura analisada com 4,5% de volume de água em etanol combustível coincidem com valores também

possíveis de 5,5% e 6,5%. Os valores plotados referem-se a dados de capacitância em função da variação de temperatura, inicialmente aquecida a partir da temperatura ambiente até cerca de 55°C, resfriado até próximo a 0°C e aquecido novamente a 55°C. Observa-se uma curva de histerese nesse ciclo, cujas interferências entre as curvas dificultam a classificação das amostras de acordo com suas concentrações de água e torna a tarefa não trivial. Essa histerese pode ter ocorrido devido à possível evaporação de álcool, e conseqüente alteração de concentração. Porém, essa hipótese não será adotada nesse momento.

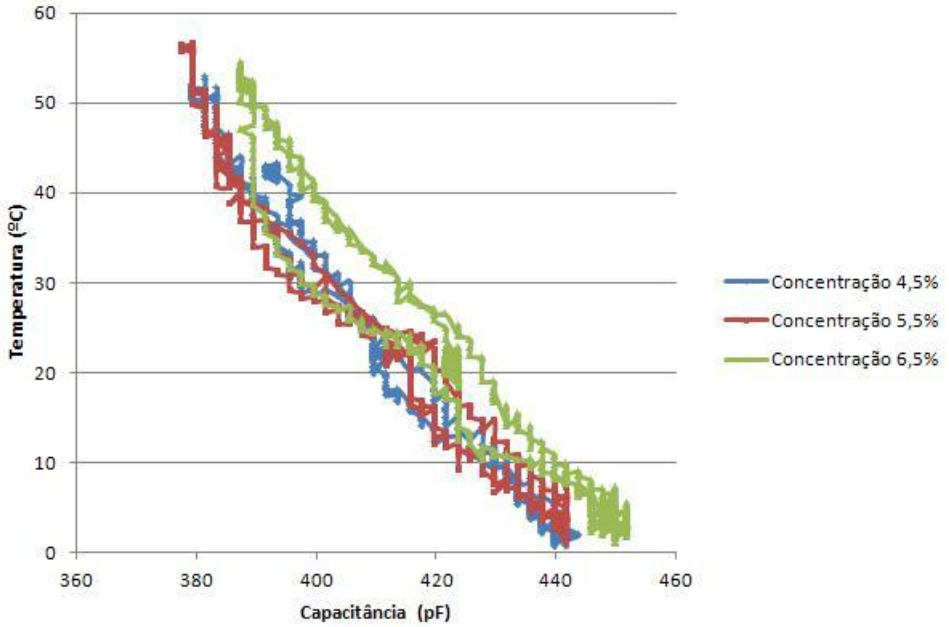

Figura 6.14 – Gráfico da Temperatura versus Capacitância obtido pelo sensor 1

6.3.4 Pré-processamento dos dados e extração de padrões

Como explicado na Seção 6.2.4, os conjuntos presentes na base de dados foram pré-processados de forma a serem compatíveis com o programa WEKA.

O objetivo desse experimento é classificar os exemplos em três classes de acordo com a concentração volumétrica de água na mistura. Mais uma vez essa informação foi inserida manualmente na base de dados. Utilizou-se a base de dados completa obtida pelos quatro sensores capacitivos e pelo sensor de temperatura e também foi utilizada a função *Aleatório()* do programa Excel para escolher exemplos da base de dados para compor o grupo de dados de treinamento. O restante dos exemplos foi utilizado como dados de teste. Foram preparados arquivos de acordo com a Tabela 10 a seguir.

Tabela 10 - Distribuição de exemplos em treinamento e teste

Nº de exemplos na base de treinamento	Nº de exemplos na base de teste
20	12617
40	12597
60	12577
80	12557
100	12537
150	12487
200	12437
300	12337
500	12137
700	11937

6.3.5 Avaliação dos resultados obtidos

Conforme descrito na Seção 6.3.1, foi utilizado o algoritmo AdaBoost.M1 como algoritmo principal e os algoritmos J48 e Random Forest para gerar os classificadores fracos.

As curvas de aprendizado novamente servirão como base para avaliar a classificação realizada pelo AdaBoost. O primeiro gráfico mostra a curva de aprendizado utilizando os resultados dos quatro sensores capacitivos além do sensor de temperatura. Nesse caso foi simulada uma rede maior de sensores, sendo 5 sensores no total.

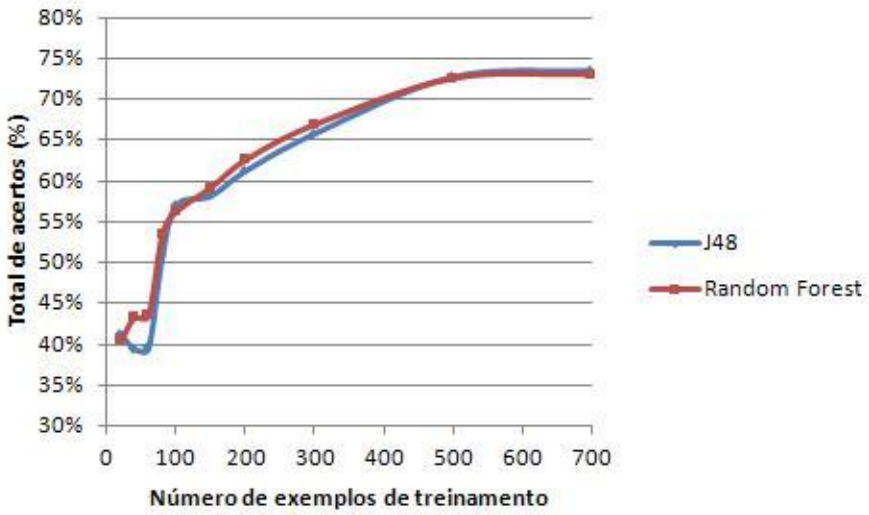

Figura 6.15 – Curva de aprendizado obtida pela análise do conjunto de sensores

A curva obtida pelos dois algoritmos base foi semelhante, tendo seguido a mesma tendência de evolução. Ambas apresentaram um salto no total de classificações corretas realizado com 60 exemplos de treinamento em comparação com 80 exemplos de treinamento. Em seguida a inclinação da curva começou a diminuir, indicando uma estabilização próxima aos 75% de acertos.

Estima-se que o patamar de saturação possa ser elevado por meio de modificações nos parâmetros dos algoritmos base ou por meio de inclusão de um novo atributo e assim incluir mais informações sobre a amostra realizada.

Uma análise semelhante foi realizada tendo como foco apenas os dados obtidos pelo sensor de temperatura e pelo sensor 1, eliminando as informações dos demais sensores. Nesse caso, a base total de exemplos reduziu de 12637 para 3160. A mesma proposta de testes com evolução no número de exemplos na base de treinamento foi utilizada e a curva de aprendizado obtida é ilustrada na Figura 6.16.

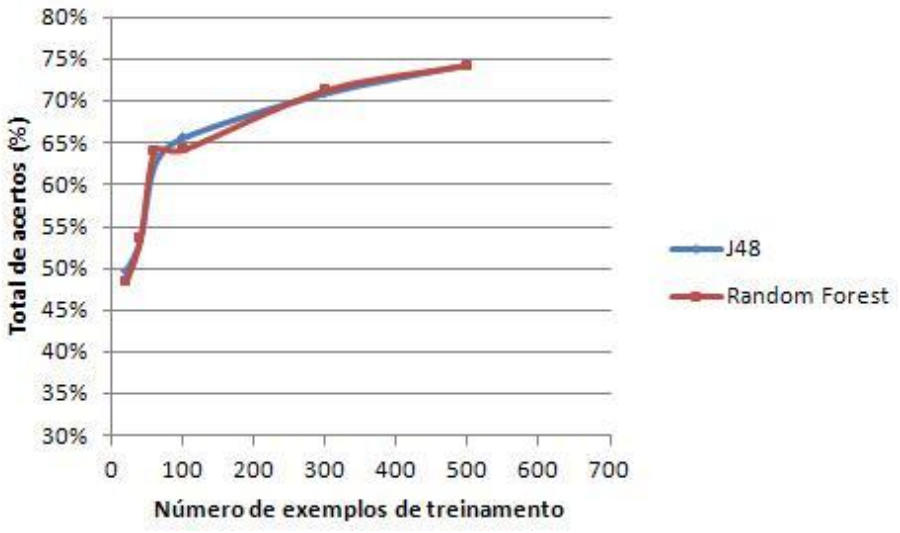

Figura 6.16 - Curva de aprendizado obtida pela análise de um sensor capacitivo e sensor de temperatura

Novamente é possível perceber uma forte inclinação no início da curva e uma tendência de saturação em torno dos 75% de êxito.

Os dois resultados obtidos e apresentados em forma de gráfico, apesar de necessitar ajustes para melhorar os índices de sucesso, indicam ser viável a utilização de algoritmo de aprendizagem de máquina como forma de calibrar sensores e também como forma de realizar tomadas de decisão.

6.4 Estudo de caso 3: Conjunto de Sensor de Fibra Óptica e de Temperatura

6.4.1 Aspectos Gerais

Para a terceira experiência aqui descrita, o objetivo foi avaliar o método de classificação utilizando-se a estrutura monolítica do algoritmo AdaBoost e aplicada a um sensor diferente e também a misturas diferentes dos experimentos anteriores.

Para isso, foi utilizada uma rede de sensores com o intuito de avaliar misturas compostas por etanol combustível e gasolina do tipo C em diferentes concentrações. Os sensores utilizados foram dois de temperatura e um sensor de fibra óptica para mensurar a refletância do laser na amostra.

Para essa experiência foi utilizado o AdaBoost junto com o Random Forest como algoritmo base para analisar os dados e realizar as classificações.

6.4.2 Obtenção dos dados

A base de dados foi cedida pelos pesquisadores do Laboratório de Materiais e Dispositivos Fotônicos da Unicamp e é composta por informações de Temperatura da amostra [°C], Temperatura interna do sensor [°C], Intensidade luminosa refletida [%] e Intensidade do Laser [%].

Onze misturas com concentrações diferentes de etanol combustível e gasolina tipo C foram preparadas e, por meio dos sensores, dados foram extraídos das amostras.

Essa base consiste de 548900 exemplos e 4 atributos, além da classe relativa à concentração da mistura a ser utilizada para classificar cada exemplo. Mais informações sobre esses dados serão fornecidas a seguir.

6.4.3 Análise dos Dados

As onze amostras preparadas continham desde etanol combustível puro até gasolina tipo C pura, além de misturas com concentrações intermediárias. As concentrações das amostram seguem o exposto na tabela 11.

Tabela 11 – Concentrações presentres nas amostras analisadas

Amostra	1	2	3	4	5	6	7	8	9	10	11
Concentração de etanol	100%	90%	80%	70%	60%	50%	40%	30%	20%	10%	0%
Concentração de gasolina tipo C	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%

Para cada amostra foram extraídos 49900 exemplos, totalizando os 548900 presentes na base de dados.

Os dados apresentaram linearidade ao serem analisados os atributos temperatura da amostra (^{0}C) e intensidade luminosa refletida (%), como ilustrado algumas concentrações no gráfico a seguir (Figura 6.17).

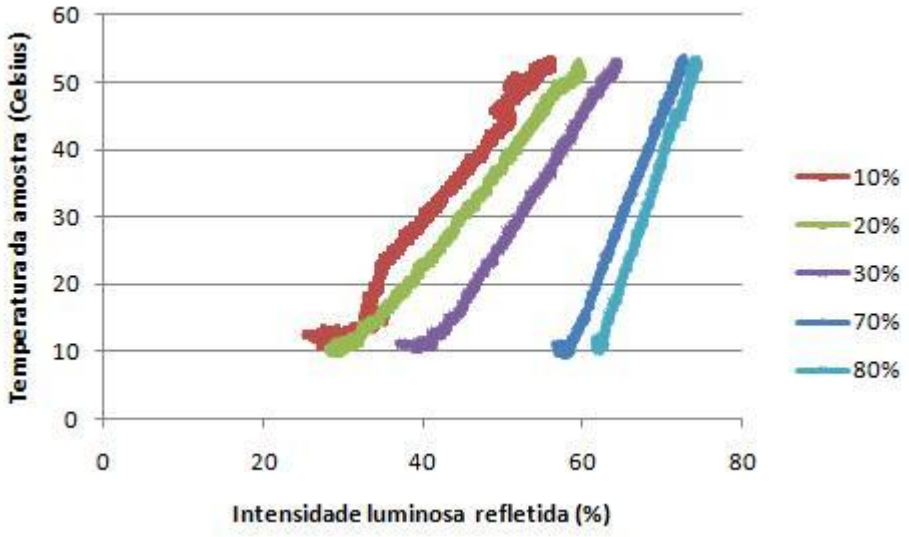

Figura 6.17 – Distribuição obtida pelo sensor de fibra óptica das diferentes concentrações de Etanol na mistura.

6.4.4 Pré-processamento dos dados e extração de padrões

O objetivo desse experimento é classificar os exemplos em onze possíveis classes de acordo com a concentração de etanol e gasolina na mistura. Novamente essa informação foi inserida manualmente na base de dados utilizando o programa Excel. Dessa vez os dados foram colocados em uma lista no programa Excel e escolhidos o exemplo número 500 da lista e os exemplos múltiplos de 500 para compor a base de exemplos de treinamento. Dessa

forma, foram escolhidos 100 exemplos por concentração, totalizando 1100 exemplos para treinamento. O restante dos exemplos foi utilizado para teste, totalizando 547800 nessa base de teste.

Após essas definições, os conjuntos de exemplos foram pré-processados de forma a serem entendidos pelo programa WEKA.

6.4.5 Avaliação dos resultados obtidos

Conforme descrito na Seção 6.4.1, foi utilizado o algoritmo AdaBoost.M1 como algoritmo principal e o algoritmo Random Forest para gerar os classificadores fracos.

O gráfico abaixo (Figura 6.18) ilustra os resultados de classificações corretas obtidos para cada uma das classes possíveis.

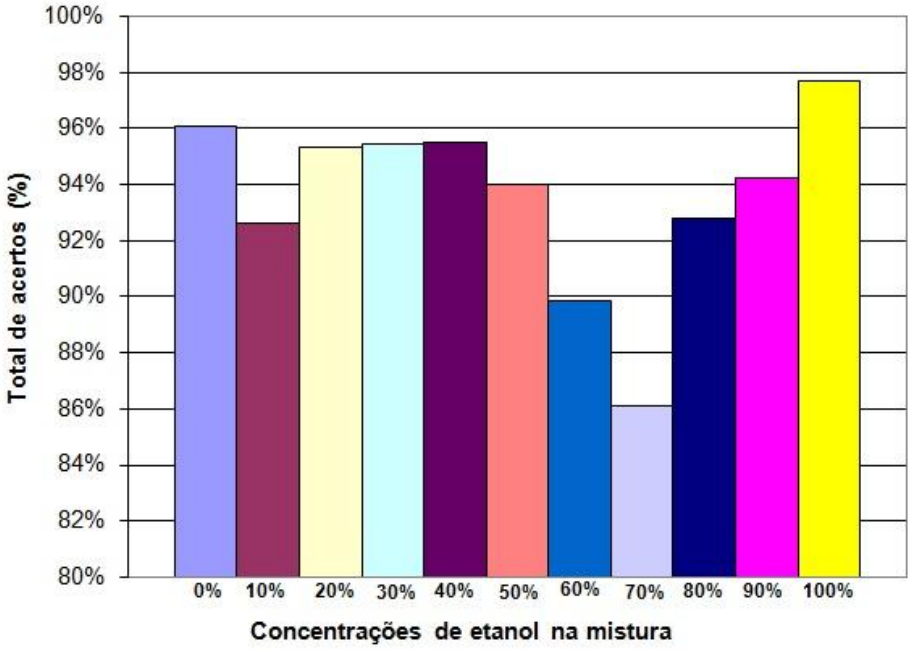

Figura 6.18 – Gráfico de sucesso na classificação das misturas de etanol e gasolina

Nesse caso o índice de acertos foi elevado, demonstrando ser viável a aplicação de algoritmos de aprendizagem de máquina também nesse sistema de sensores.

6.5 Estudo de caso 4: Conjunto de Sensores Capacitivos e de Temperatura – Comparação entre estruturas Monolítica e Cascata do AdaBoost

6.5.1 Aspectos Gerais

O principal objetivo desta quarta experiência foi comparar o desempenho entre as duas principais estruturas de AdaBoost. Para testar a estrutura monolítica foi utilizado o software WEKA (WITTEN; FRANK, 2005). Utilizou-se o algoritmo AdaBoost.M1 junto ao *Decision Stump*, este último sendo o classificador fraco. Para a segunda estrutura do AdaBoost, foi utilizada uma implementação própria desenvolvida na Massey University da estrutura de Cascata junto com *Decision Stump* como classificador fraco (VIOLA, JONES, 2001; BARCZAK, JOHNSON, MESSOM, 2008).

Foram utilizados três sensores capacitivos, sendo dois deles fabricados com eletrodos de cobre em substrato de fibra de vidro e outro com eletrodos de níquel em substrato de alumina, além dos três apresentarem dimensões distintas (MENDONÇA et al, 2011). Além disso, foram utilizados dois sensores de temperatura. Novamente o foco principal da classificação foi sobre a qualidade de amostras de etanol em relação à quantidade de água diluída.

6.5.2 Obtenção dos dados

A base de dados foi obtida preparando experimentalmente amostras de etanol anidro e etanol com água diluída em concentrações volumétricas de 4,5%, 5,5% e 6,5%. As amostras foram aquecidas a partir de aproximadamente 1^{o}C até 25 ^{o}C. Durante esse processo, todos os sensores permaneceram submersos na solução e os dados aquisitados a cada 4 segundos.

As cinco informações acompanhadas no experimento foram: três valores de capacitância em (pF) forcenidos pelos três diferentes sensores capacitivos, além de dois valores de temperatura mensurados em (^{o}C).

6.5.3 Análise dos Dados

A base de dados contém um total de 1455 exemplos, distribuídos nas quatro concentrações de água diluída em etanol anidro mencionadas acima, de acordo com a tabela 12 a seguir.

Tabela 12 – Distribuição de exemplos presentes na base de dados

Concentração volumétrica de água em etanol (%)	Exemplos
0,0%	335
4,5%	389
5,5%	380
6,5%	351
Total	1455

Os gráficos a seguir (Figuras 6.19 a 6.21) apresentam a distribuição da capacitância (pF) em relação à temperatura (^{o}C) obtidos pelos três sensores capacitivos utilizados no experimento.

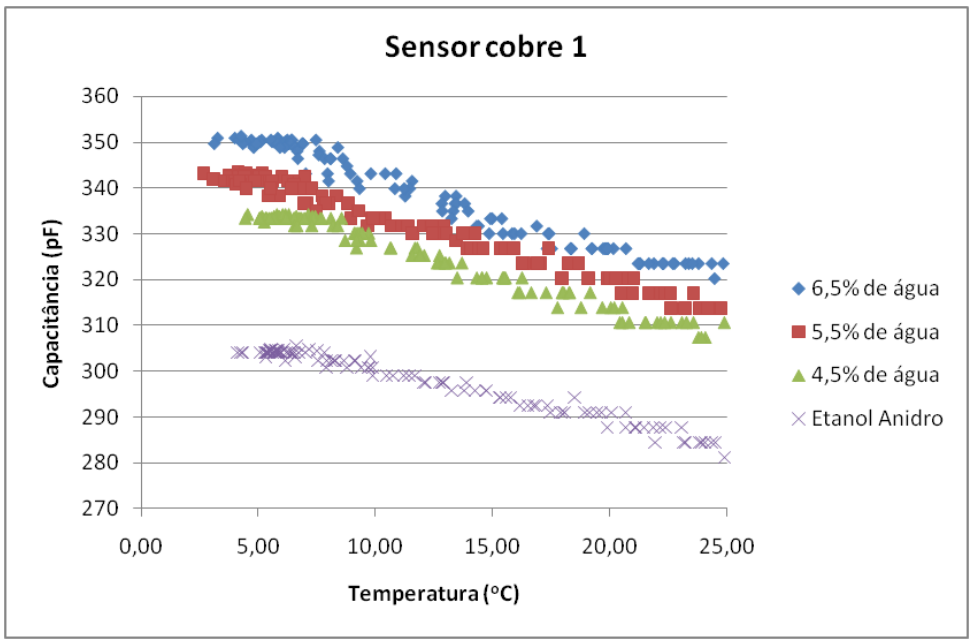

Figura 6.19 - Capacitância obtida pelo sensor capacitivo de cobre 1 versus temperatura obtida pelo primeiro sensor de temperatura

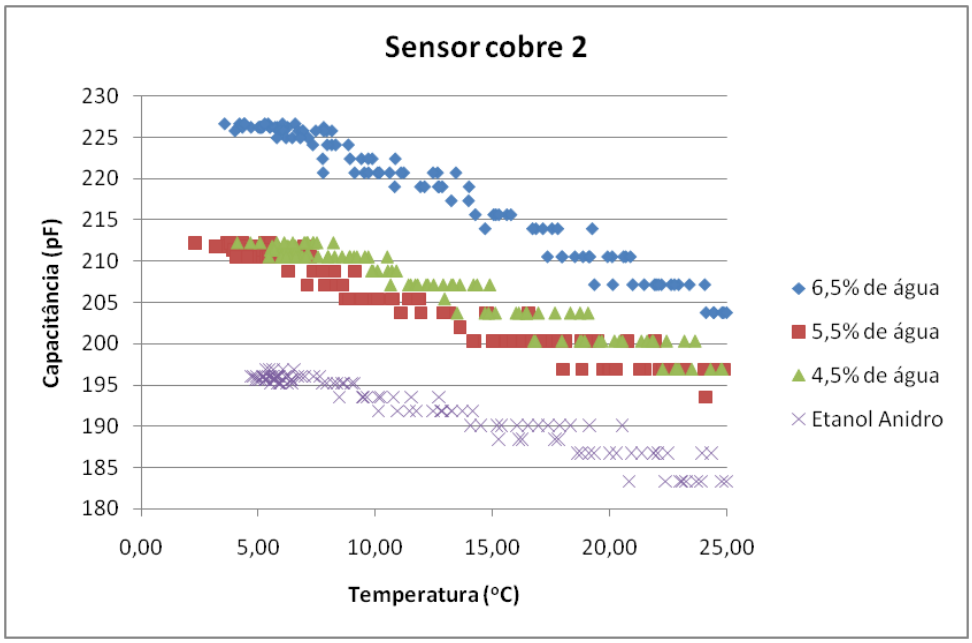

Figura 6.20 - Capacitância obtida pelo sensor capacitivo de cobre 2 versus temperatura obtida pelo primeiro sensor de temperatura

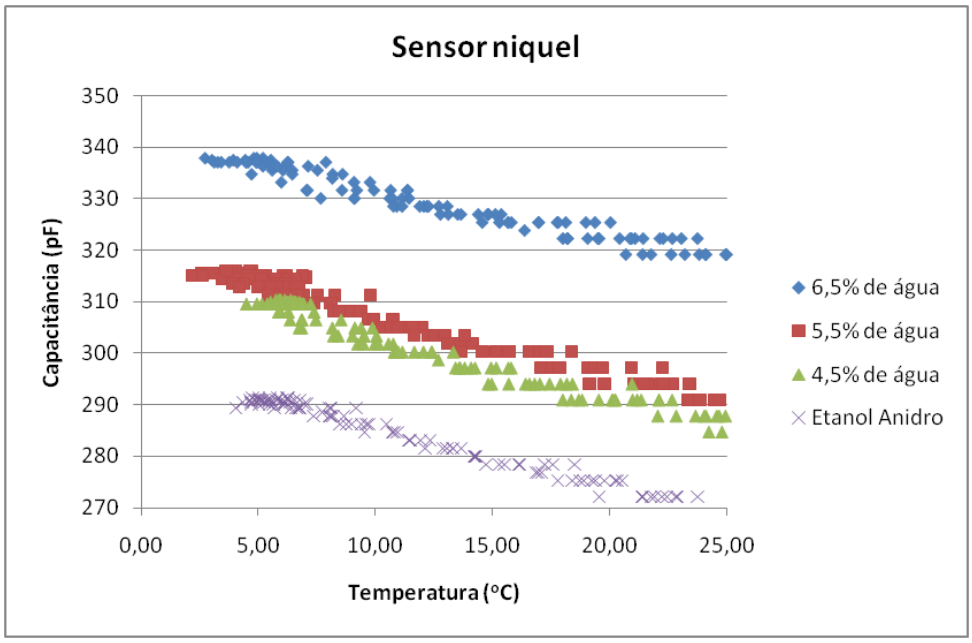

Figura 6.21 - Capacitância obtida pelo sensor capacitivo de niquel versus temperatura obtida pelo primeiro sensor de temperatura

A partir dos gráficos é possível identificar sobreposições parciais entre as concentrações de 4,5% e 5,5% de água diluída em etanol. Por outro lado, o etanol anidro e a concentração de 6,5% de água diluída são mais facilmente identificados. Apesar da baixa complexidade do problema, este caso mostra-se extremamente interessante para realizar a comparação entre as estruturas do AdaBoost.

6.5.4 Pré-processamento dos dados e extração de padrões

Utilizando-se o modelo de problema binário, os exemplos positivos e negativos presentes no caso avaliado têm a seguinte distribuição. A classe dos exemplos foi inserida manualmente, baseada na legislação brasileira.

Tabela 13 - Distribuição dos exemplos positivos e negativos

Concentração volumétrica de água em etanol (%)	Exemplos	Classificação	Classe
0,0%	335	Não conforme	Exemplos Negativos
4,5%	389	Não conforme	Exemplos Negativos
5,5%	380	Conforme	Exemplos Positivos
6,5%	351	Não conforme	Exemplos Negativos
Total	1455	-	-

Dois subconjuntos foram gerados de forma aleatória a partir da base total de dados. O primeiro deles, chamado de conjunto de teste, contendo 455 exemplos. O segundo subconjunto gerado continha 700 exemplos e foi utilizado como conjunto de treinamento. A partir de então, o conjunto de treinamento contento 700 exemplos foi utilizado para treinar um classificador e o conjunto de teste foi utilizado em seguida para mensurar o desempenho do classificador gerado. Esses resultados serão descutidos a seguir.

6.5.5 Avaliação dos resultados obtidos

Os resultados serão apresentados a seguir explorando dois aspectos. Em um primeiro momento, os totais de hit ratio (classificações positivas corretas) obtidos pelas duas estruturas são comparados, sendo que a estrutura monolítica foi treinada até o máximo de 2000 iterações. Os resultados são apresentados em curvas ROC. A figura 6.22 apresenta a comparação dos resultados das duas estruturas em um único gráfico, enquanto as figuras 6.23 e 6.24 apresentam os resultados de forma separada da estrutura Monolítica e da estrutura Cascata, respectivamente.

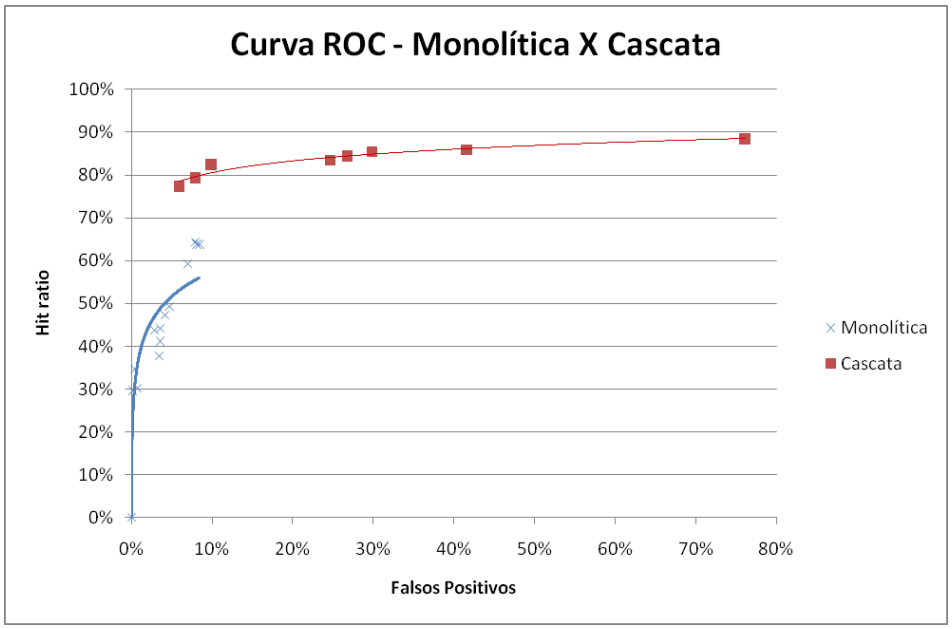

Figura 6.22 - Curva ROC obtida pela estrutura cascata e também pela estrutura Monolítica a partir de 700 exemplos de treinamento

A partir dos gráficos nota-se que a estutura monolítica atingiu um índice aproximado de 60% de hit ratio para 8,5% de falsos positivos (Figura 6.23). Por outro lado, a estrutura cascata atingiu aproximadamente 78% de hit ratio para os mesmos 8,5% de falsos positivos (Figura 6.24). Também é possível verificar que a estrutura monolítica converge para um patamar de cerca de 60% de hit ratio mesmo que se aumente o índice de falsos positivos, enquanto que a estrutura cascata permite a calibração de classificações corretas de cerca de 90% em detrimento do crescimento do índice de falsos positivos.

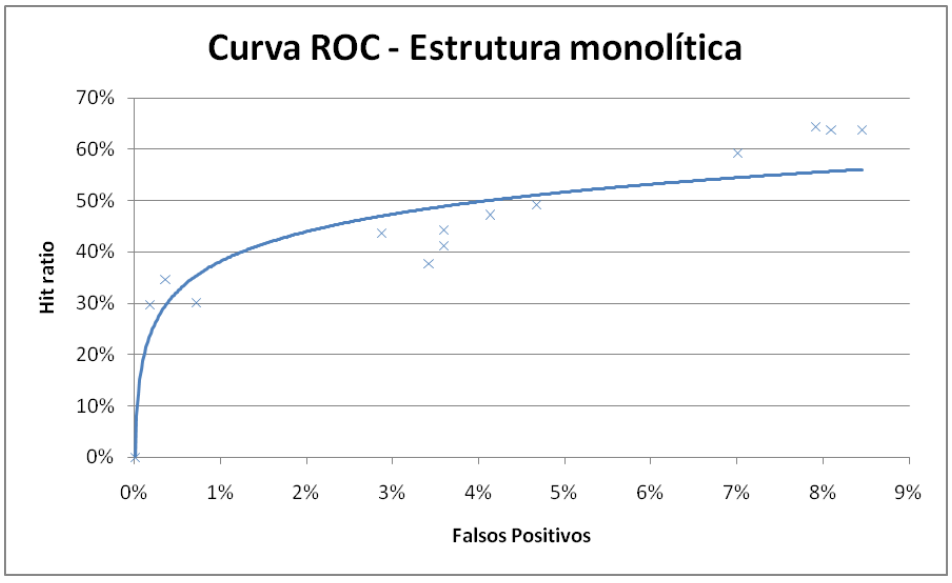

Figura 6.23 - Curva ROC obtida a partir da estrutura Monolítica

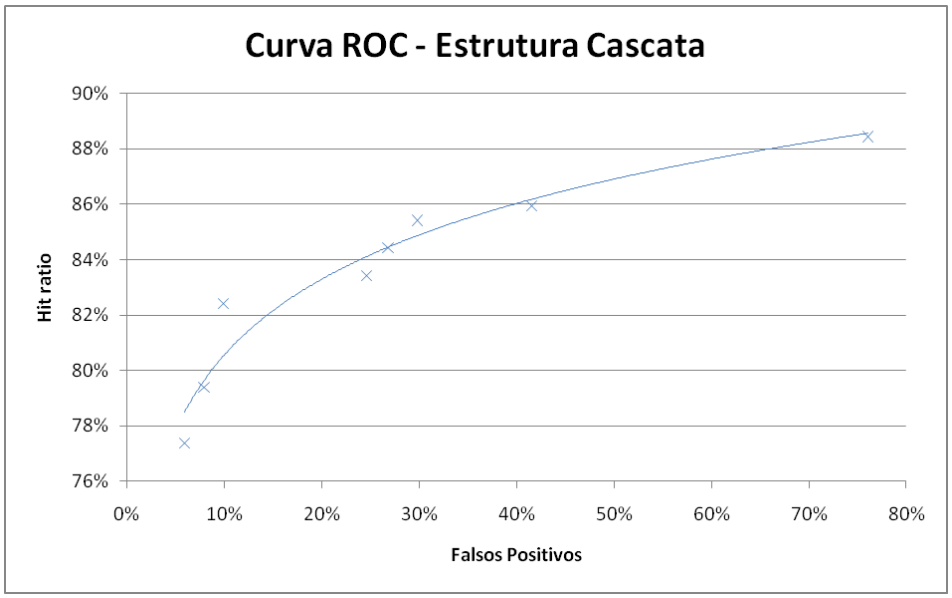

Figura 6.24 - ROC obtida a partir da estrutura Cascata

O segundo aspecto apresentado refere-se à evolução dos treinamentos de acordo com a quantidade de iterações (ou classificadores fracos) produzidos durante o processo. Este é um importante indicador de desempenho tendo em vista que quanto maior o número de iterações, menor serão os erros de treinamento e teste. Independentemente da estrutura utilizada é esperado que as curvas de erro convirjam para um determinado patamar devido ao algoritmo AdaBoost, mesmo que o número de iterações seja aumentado.

As figuras 6.25 e 6.26 apresentam as curvas de erro obtidas pela utilização das estruturas monolítica e cascata, respectivamente, para o caso de 700 exemplos no conjunto de treinamento. Mais uma vez nota-se que os resultados obtidos a partir da utilização da estrutura Cascata foram melhores se comparados aos obtidos a partir da estrutura Monolítica, tendo em vista que a primeira estrutura eliminou o erro de treinamento e atingiu aproximadamente 5% de erro de teste (Figura 6.26), enquanto que a segunda estrutura atingiu cerca 5% de erro de treinamento e 15% de erro de teste (Figura 6.25). Também se faz necessário destacar que o número total de iterações realizadas pela estrutura cascata foi de 607, enquanto que a estrutura monolítica atingiu as 2000 iterações máximas estipuladas. Este desempenho é importante para esse estudo, tendo em vista que é desejado embarcar o algoritmo e, assim, o classificador mais compacto, ou seja, com menor número de iterações (classificadores fracos), é mais interessante por ser de tamanho menor. Portanto, foi demonstrado que para o problema estudado nessa pesquisa, a utilização da estrutura Cascata apresenta melhores resultados e também melhor desempenho.

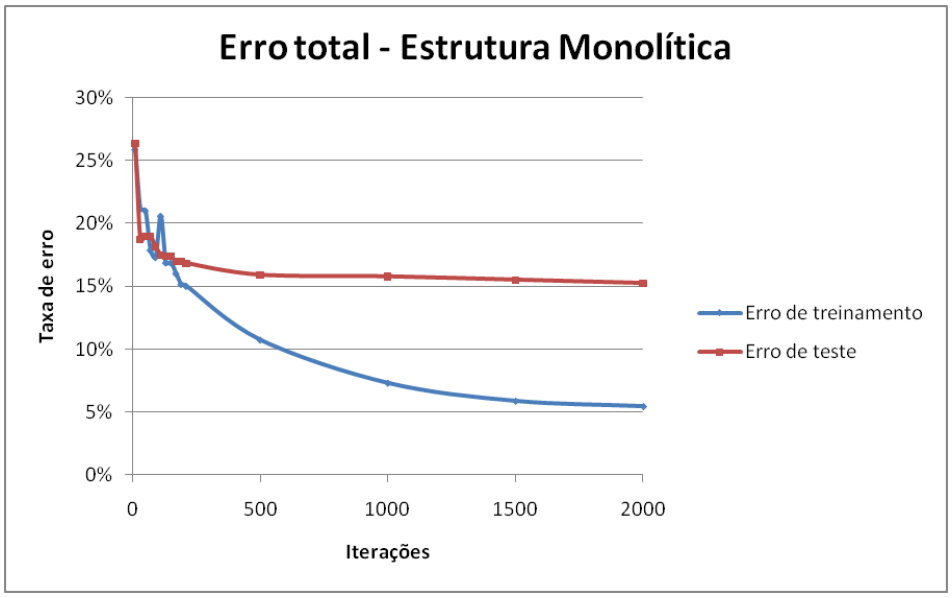

Figura 6.25 - Erro total fornecido pela estrutura Monolítica

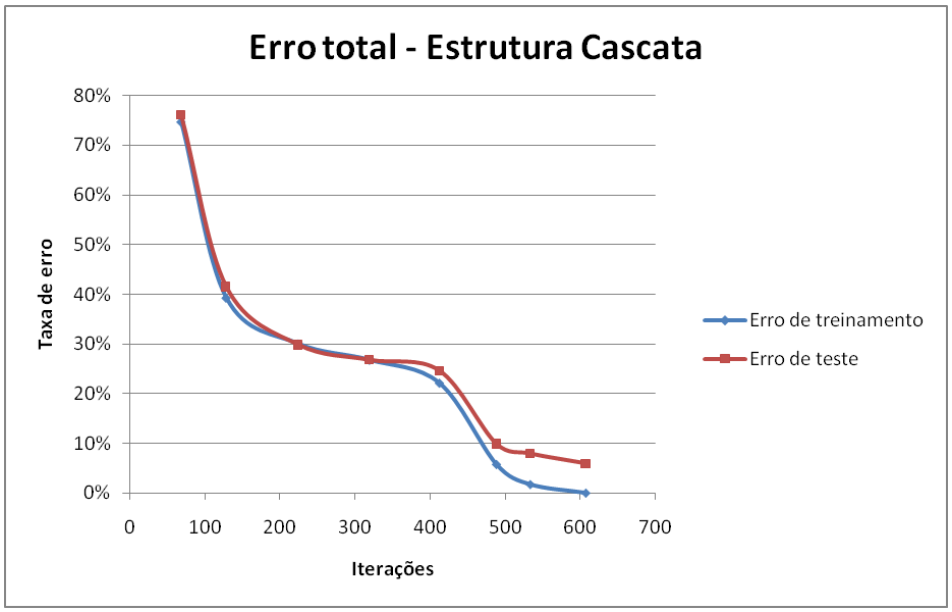

Figura 6.26 - Erro total fornecido pela estrutura Cascata

Os resultados dessa experiência sugerem ser possível atingir calibrações com razoável acurácia a partir da combinação de um algoritmo como, por exemplo, AdaBoost e sensores ou rede de sensores.

Uma vantagem de se utilizar o algoritmo AdaBoost é a compilação rápida de informação obtida a partir de diferentes sensores. Além disso, é possível calibrar de forma acurada qualquer sensor, independentemente da forma da curva, mesmo que esta seja em formato totalmente inusitado. Por exemplo, a relação entre temperatura e capacitância apresentadas

anteriormente (Figuras 6.19, 6.20 e 6.21) podem ser aproximadas a uma tendência de linha reta. No entanto, mesmo que essas curvas apresentassem funções desconhecidas e complexas, seria possível representá-las por meio da utilização do AdaBoost, contanto que sejam fornecidos exemplos suficientes de treinamento.

6.6 Estudo de caso 5: Conjunto de Sensores Capacitivos e de Temperatura

6.6.1 Aspectos Gerais

Como continuidade da experiência anterior, essa quinta experiência utiliza os mesmos dados obtidos e descritos no item anterior. Tendo sido verificado o melhor desempenho da estrutura cascata, o objetivo deste quinto experimento é gerar classificadores e analisar os respectivos desempenhos por meio da variação de exemplos de treinamento, além de verificar, ainda de que forma não aprofundada, a influência do desbalanceamento de exemplos positivos e negativos do problema.

6.6.2 Obtenção dos dados

Idem ao processo descrito no item 6.5.2 deste trabalho.

6.6.3 Análise dos Dados

A análise dos dados pode ser encontrada no item 6.5.3 deste trabalho.

6.6.4 Pré-processamento dos dados e extração de padrões

Novamente o modelo de problema binário foi utilizado para abordar o problema e a Tabela 13 apresentada no experimento anterior contendo a distribuição dos exemplos e classificação de acordo com a legislação brasileira foi utilizada.

Como primeira etapa, a base de dados foi dividida entre exemplos para teste e para treino. Tal divisão foi realizada de forma aleatória. Como segunda etapa, a partir dos dados de treinamento foram criados também aleatoriamente grupos de 700, 800, 900 e 1000 exemplos. Detalhes sobre os subconjuntos são apresentados na Tabela 14 a seguir.

Tabela 14 - Distribuição dos exemplos para treino e teste

Conjunto	Exemplos	Classe
Treino	1000	271 exemplos positivos
		729 exemplos negativos
Teste	455	109 exemplos positivos
		346 exemplos negativos
Total	1455	-

6.6.5 Avaliação dos resultados obtidos

A seguir (Figura 6.27) são apresentadas as curvas ROC obtidas a partir de 700, 800, 900 e 1000 exemplos de treinamento e utilizando-se a estrutura Cascata.

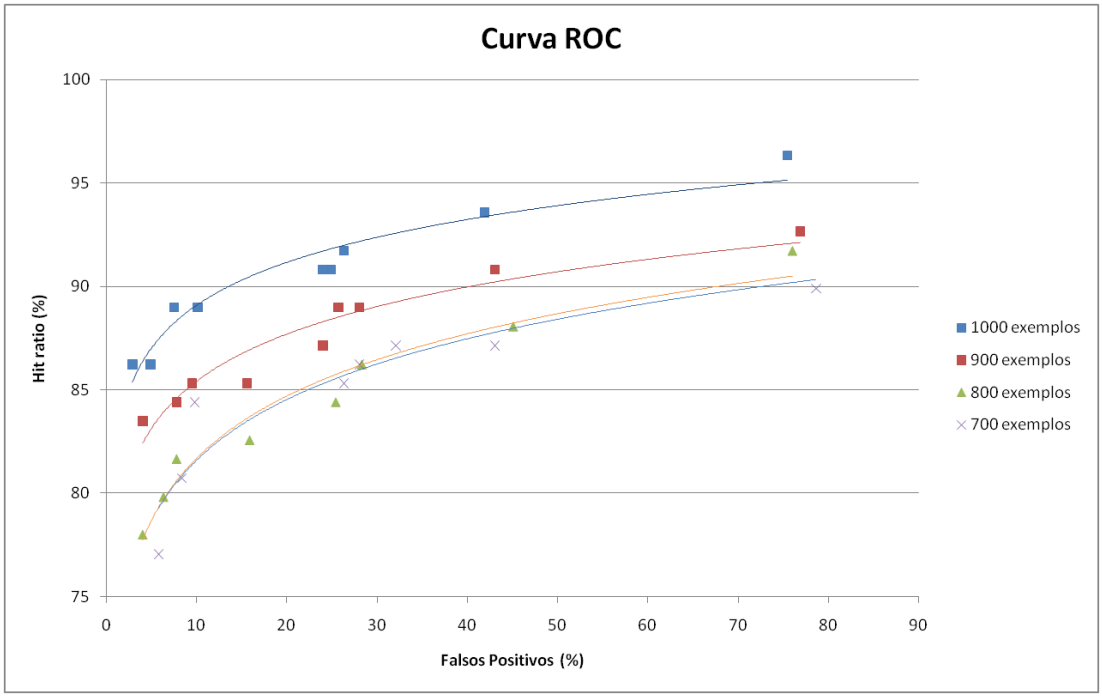

Figura 6.27 - Curvas ROC obtidas a partir de diferentes conjuntos de treinamento e utilizando-se a estrutura Cascata

Conforme esperado, é possível verificar a partir do gráfico que o desempenho do classificador melhora com o aumento do número de exemplos de treinamento. No entanto, de acordo com a teoria, é esperado que este desempenho converja para um dado patamar mesmo que o número de dados de treinamento continue aumentando.

Cada ponto da curva representa o resultado obtido por uma camada, sendo que a curva evolui da direita para a esquerda conforme o treinamento é realizado. Em outras palavras, a

primeira camada é o primeiro ponto da direita e a última camada é o último ponto da esquerda. Assim, é possível afirmar que o número de falsos positivos e hit ratio diminuem conforme o aumento do número de camadas geradas durante o treinamento. Neste caso, o melhor resultado foi obtido a partir de 1000 exemplos de treinamento, tendo atingido os índices de 85% de hit ratio e 3% de falsos positivos.

Com o objetivo de checar a repetibilidade dos resultados, a base de dados inicial contendo 1455 exemplos foi dividida de forma aleatória em conjunto de treinamento e teste por duas outras vezes, exatamente como já havia sido realizado até o momento. Devido aos melhores resultados gerados, os conjuntos de treinamento contendo 900 e 1000 exemplos foram gerados a partir dessas novas divisões também de forma aleatória. Os resultados obtidos a partir do treinamento das três diferentes divisões dos dados podem ser verificados a seguir (Figuras 6.28 e 6.29).

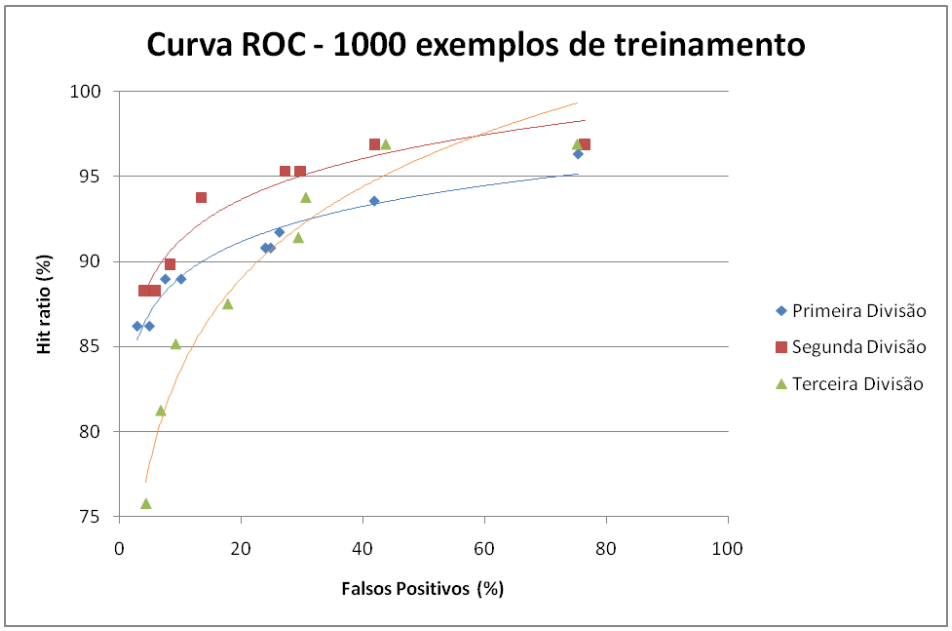

Figura 6.28 - Curva ROC produzida a partir de diferentes divisões da base de dados e conjunto de treinamento contendo 1000 exemplos

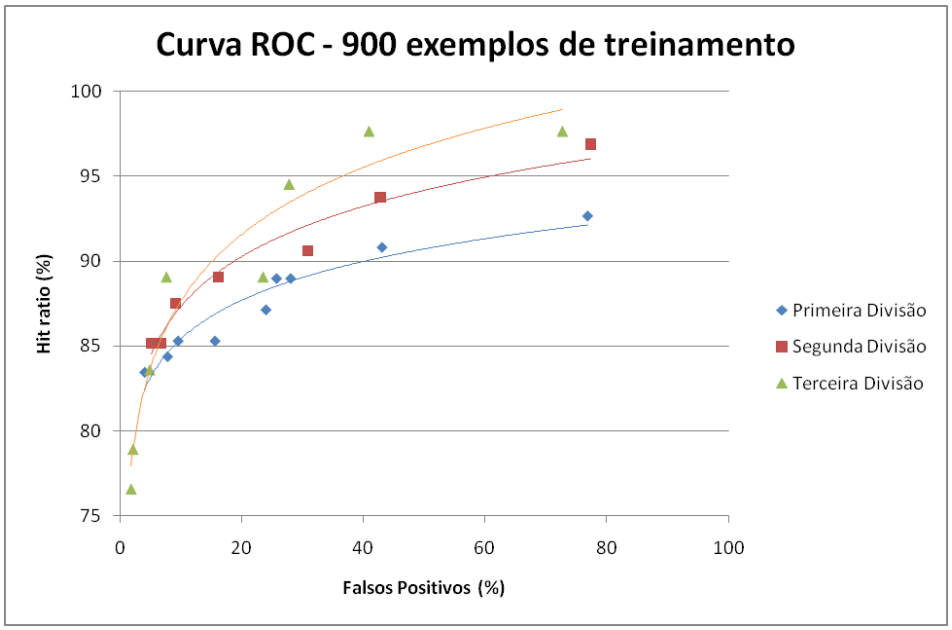

Figura 6.29 - Curva ROC produzida a partir de diferentes divisões da base de dados e conjunto de treinamento contendo 900 exemplos

Os resultados confirmaram que os classificadores gerados a partir de conjuntos de treinamento contendo 1000 exemplos apresentam melhores desempenhos se comparados aos gerados a partir de 900 exemplos de treinamento. Além disso, a segunda divisão dos dados gerou um classificador com melhores resultados, tendo atingido 88,3% de hit ratio e 3,97% de falsos positivos. O gráfico do erro total do respectivo classificador é ilustrado na Figura 6.30.

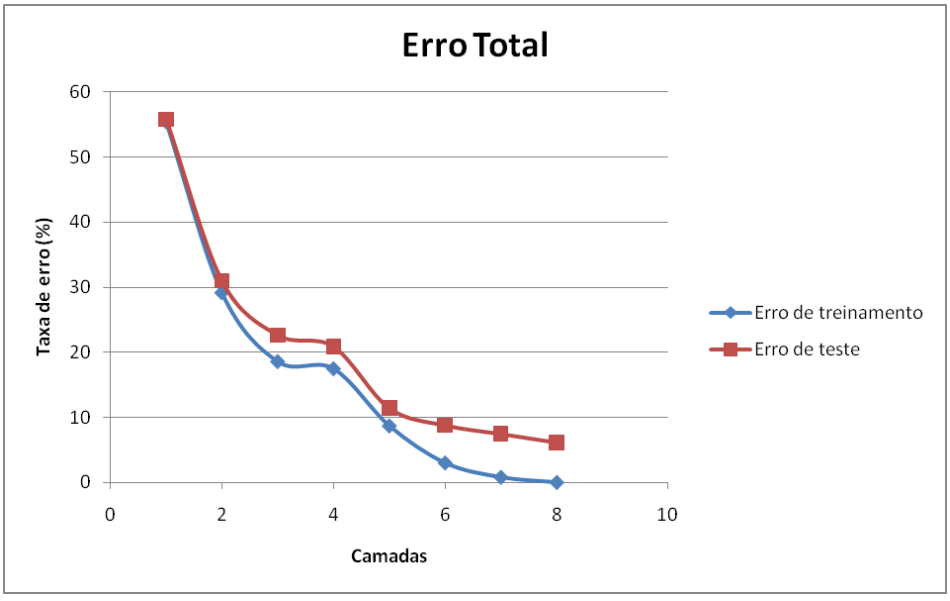

Figura 6.30 - Erro total produzido pelo classificador a partir da segunda divisão dos dados e 1000 exemplos de treinamento

Conforme mencionado previamente, essa pesquisa utiliza a abordagem binária para o problema. Exemplos positivos e negativos foram determinados até o momento na experiência conforme mostrado abaixo.

Tabela 15 – Conjuntos de exemplos positivos e negativos

Exemplos positivos	Combustível **não** adulterado	380 exemplos
Exemplos negativos	Combustível adulterado	1075 exemplos

Como pode ser observado na Tabela 15, este é um problema de aprendizagem a partir de base de dados desbalanceada, ou seja, os números de exemplos positivos e negativos são diferentes. Essa distinta distribuição de exemplos pode gerar complicações. Por exemplo, idealmente busca-se um classificador com acurária próxima a 100% para os dois grupos, com maioria e minoria dos exemplos. Porém, em um mundo real os classificadores tendem a produzir acurárias desbalanceadas também entre as duas classes. Em casos extremos, é possível que a classe em maioria apresente quase 100% de acurária, enquanto a classe em minoria apresente aproximadamente índices menores que 10% de classificações corretas (HE; GARCIA, 2009).

Com o objetivo de avaliar o impacto do desbalanceamento presente na base de dados utilizada na experiência, e por conseqüência no problema de detecção da qualidade de combustível, todo o processo descrito até o momento foi realizado novamente, porém invertendo os grupos de exemplos positivos e negativos, conforme demonstrado na Tabela 16.

Tabela 16 – Conjuntos de exemplos positivos e negativos

Exemplos positivos	Combustível adulterado	1075 exemplos
Exemplos negativos	Combustível **não** adulterado	380 exemplos

O mesmo processo de divisão da base de dados inicial de forma aleatória por três vezes em conjuntos de treinamento e teste e produção de classificadores treinados foi realizado. A seguir são ilustrados os resultados obtidos.

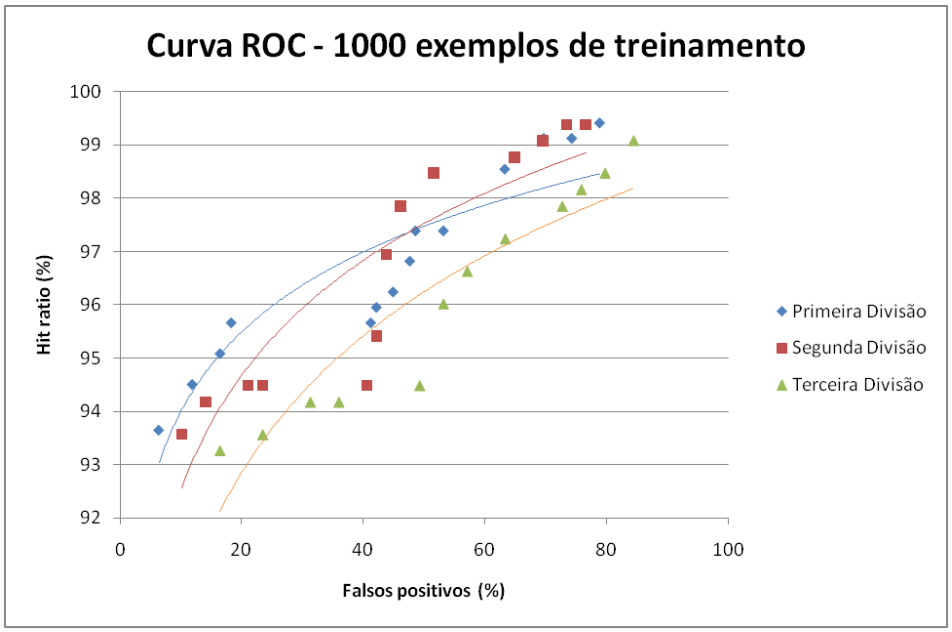

Figura 6.31 - Curvas ROC produzidas a partir de três diferentes divisões da base de dados após inverter os conjuntos positivos e negativos e conjunto de treinamento contendo 1000 exemplos

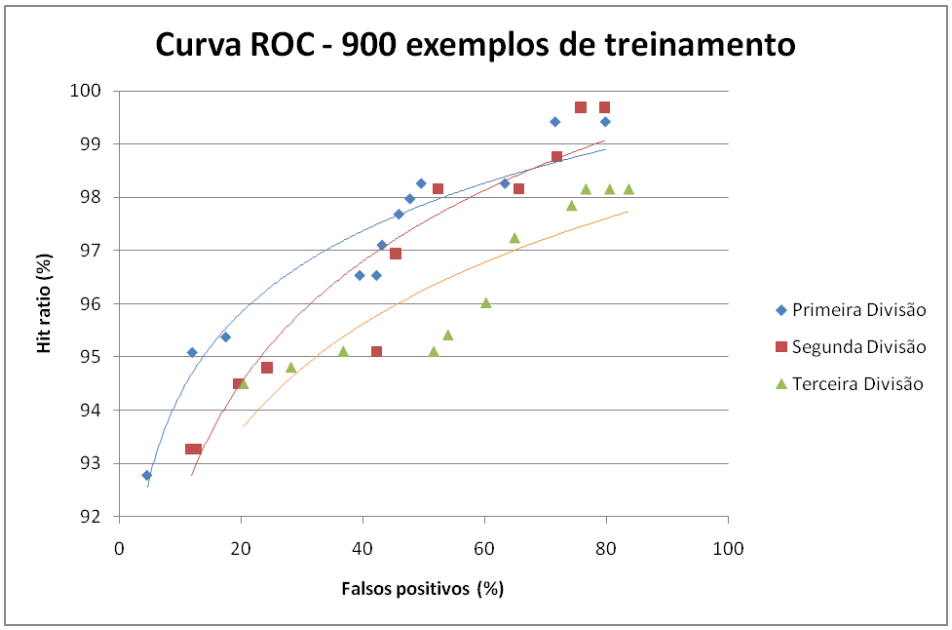

Figura 6.32 – Curvas ROC produzidas a partir de três diferentes divisões da base de dados após inverter os conjuntos positivos e negativos e conjunto de treinamento contendo 900 exemplos

Mais uma vez os melhores resultados foram produzidos a partir de 1000 exemplos na base de treinamento. Além disso, dentre as três divisões consecutivas e independentes, o melhor desempenho foi produzido pela primeira divisão, tendo sido obtido 93,6% de hit ratio e 6,4% de falsos positivos. O gráfico abaixo (Figura 6.33) apresenta a curva de erro do respectivo classificador.

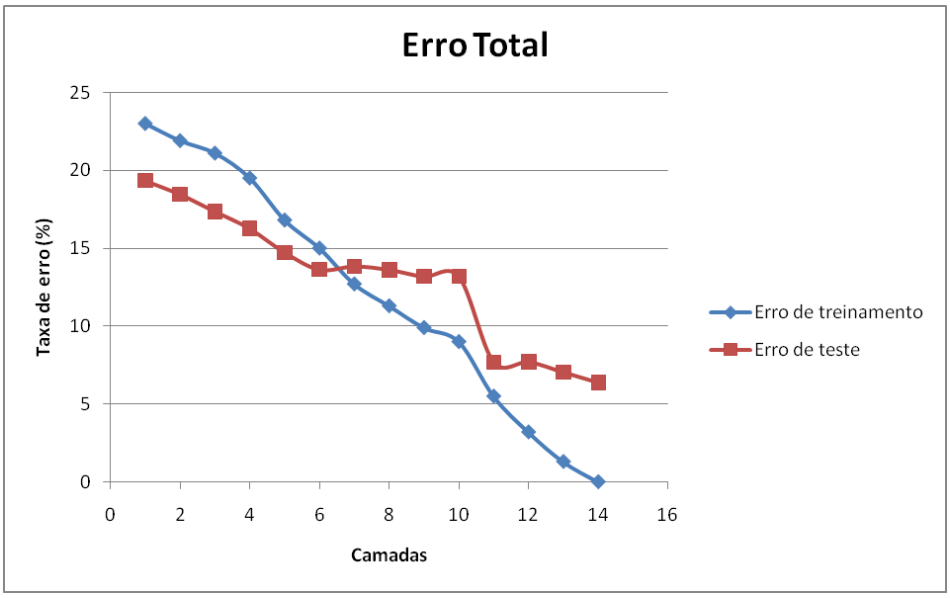

Figura 6.33 - Erro total produzido pelo classificador na primeira divisão e a partir de 1000 exemplos de treinamento

Embora haja uma pequena diferença entre os resultados obtidos pelas duas abordagens (exemplos positivos como combustíveis conformes e não conformes) é possível afirmar que neste caso o desbalanceamento de dados não gerou distúrbuio significativo. Porém, essa análise mostrou-se importante para futuros experimentos, pois apesar de pequena, houve influência nesse experimento. Assim, experimentos com maiores quantidades de exemplos ou desbalanceamentos, poderão gerar influências mais significativas.

É importante ressaltar também que a estrutura do AdaBoost em camadas minimiza esses distúrbios devido ao fato de essa estrutura minimizar os falsos positivos, como explicado na seção 3.4.2 deste trabalho. Por outro lado, esses distúrbios são mais propensos a acontecer na estrutura monolítica, pois esta é focada em minimizar o erro total das classificações (seção 3.4.1).

6.7 Estudo de caso 6: Conjunto de Sensores Capacitivos e de Temperatura

6.7.1 Aspectos Gerais

Para a sexta experiência aqui descrita, o objetivo foi avaliar o desempenho do algoritmo AdaBoost com estrutura de cascata como método de classificação de etanol combustível conforrme e não conforme de acordo com a legislação brasileira utilizando-se novos sensores capacitivos e maior faixa de temperatura se comparado às experiências anteriores.

Foram utilizados dois micro-sensores capacitivos construídos por técnicas de microfabricação. O primeiro deles foi fabricado em substrato de fibra de vidro e eletrodos de cobre, e o segundo, sobre o substrato de alumina com eletrodos de níquel. Além disso, os dois sensores apresentam estruturas semelhantes de eletrodos interdigitados, porém com *gaps* – distância entre dedos – distintas entre si.

Além dos sensores capacitivos, foram utilizados dois sensores de temperatura LM35 fabricados pela National Semiconductor Corporation.

6.7.2 Obtenção dos dados

Três concentrações na zona crítica de conformidade da mistura de etanol combustível e água foram escolhidas para o experimento: 4,5%, 5,5% e 6,5% de concentração volumétrica de água em etanol combustível.

A cada preparação de concentração os dois sensores capacitivos e também o sensor de temperatura foram imersos na amostra e os dados aquisitados por meio de um microprocessador Arduino duemilanove. Cada amostra foi resfriada até aproximadamente 2^{o}C em primeiro momento. A aquisição dos dados então foi iniciada e a amostra foi aquecida até aproximadamente 44^{o}C. Essa faixa de temperatura foi escolhida com o objetivo de simular temperaturas desde o inverno até o verão na maior parte do território brasileiro. A taxa de aquecimento foi controlada durante o processo e com valor aproximado de 1,6 ^{o}C/min.

Desta forma, as propriedades das misturas mensuradas pelo circuito foram dois valores de capacitância em (pF), dois valores de período coletado pelo sensor em (μs) e dois valores de temperatura da amostra em (ºC). Em outras palavras, foram seis propriedades analisadas na experiência.

6.7.3 Análise dos Dados

Na Tabela 17 é apresentada a distribuição das informações da base de dados. Neste caso, um exemplo significa um conjunto de informações coletadas em um mesmo instante sobre a amostra. Assim, cada exemplo é composto pelos seis valores descritos acima.

Tabela 17 - Distribuição dos exemplos coletados pelos sensores

Concentração volumétrica de água em etanol (%)	Exemplos
4,5%	858
5,5%	1063
6,5%	1169
Total	3090

A seguir (Figuras 6.34 a 6.37) são apresentados os gráficos dos dados obtidos pelos sensores. Os gráficos apresentam a relação Capacitância versus Temperatura, sendo o primeiro e o segundo gráficos de cada sensor referentes, respectivamente, à temperatura obtida pelo primeiro sensor de temperatura e pelo segundo sensor de temperatura.

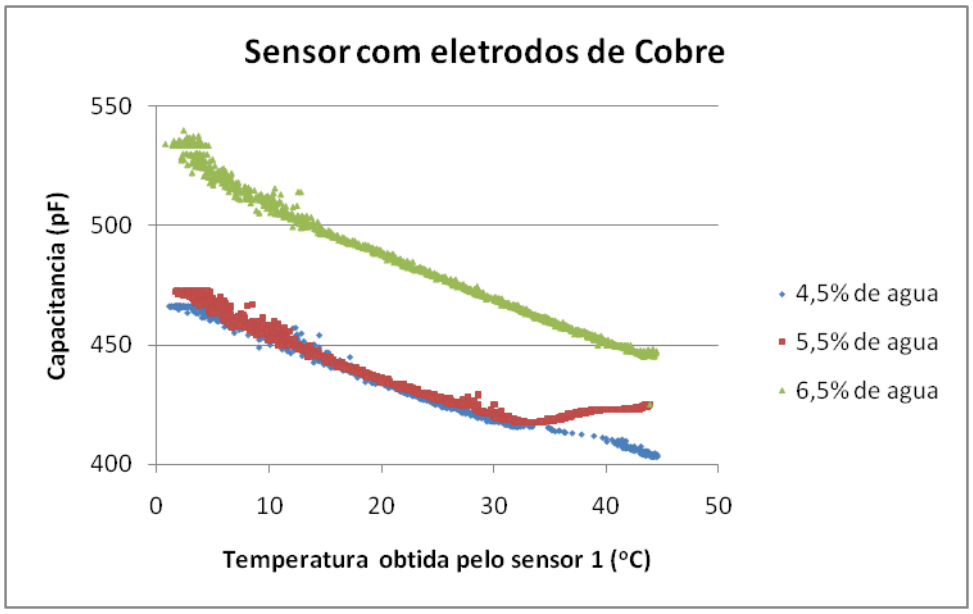

Figura 6.34 - Capacitância obtida pelo sensor capacitivo com eletrodos de cobre versus temperatura obtida pelo primeiro sensor de temperatura

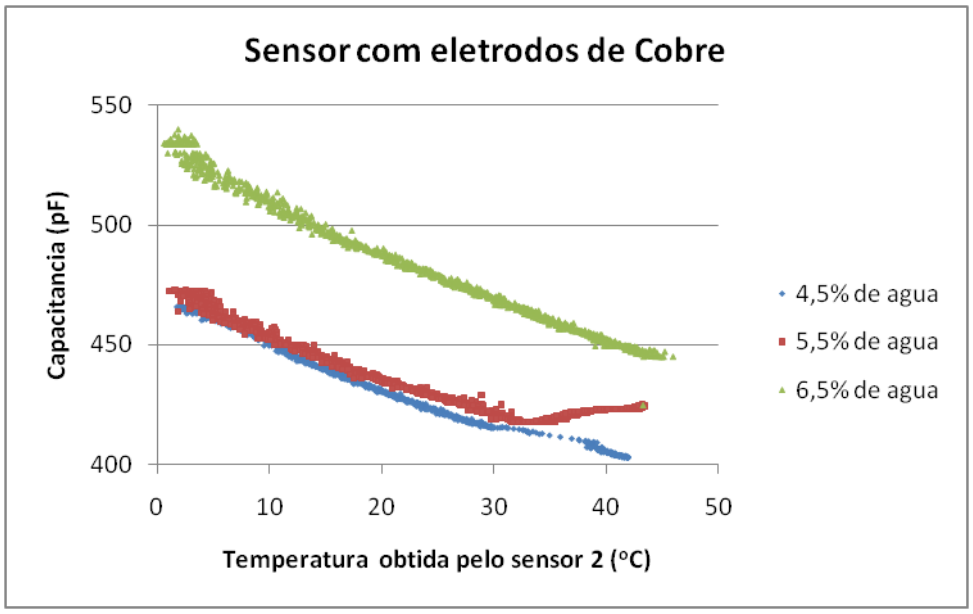

Figura 6.35- Capacitância obtida pelo sensor capacitivo com eletrodos de cobre versus temperatura obtida pelo segundo sensor de temperatura

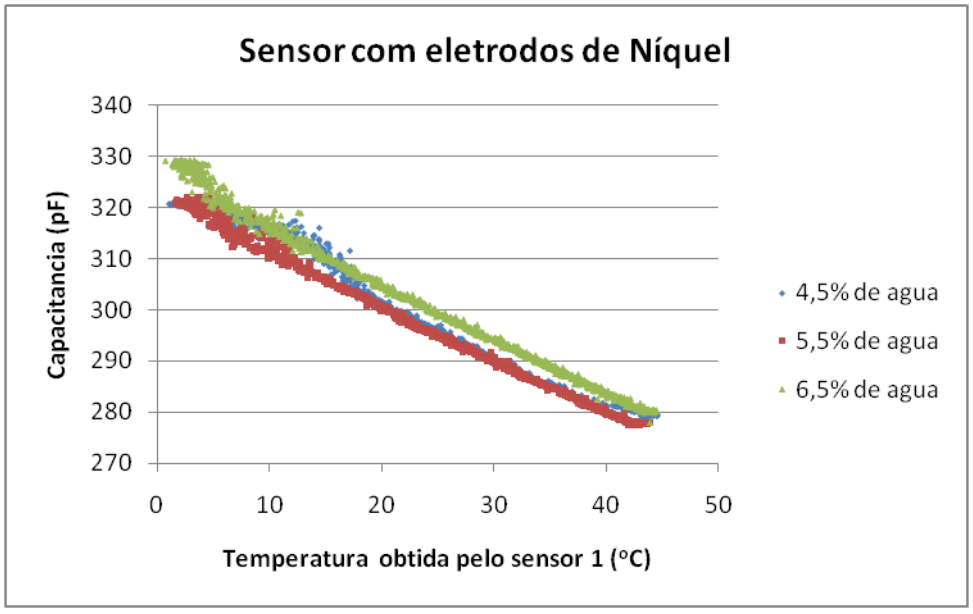

Figura 6.36- Capacitância obtida pelo sensor capacitivo com eletrodos de niquel versus temperatura obtida pelo primeiro sensor de temperatura

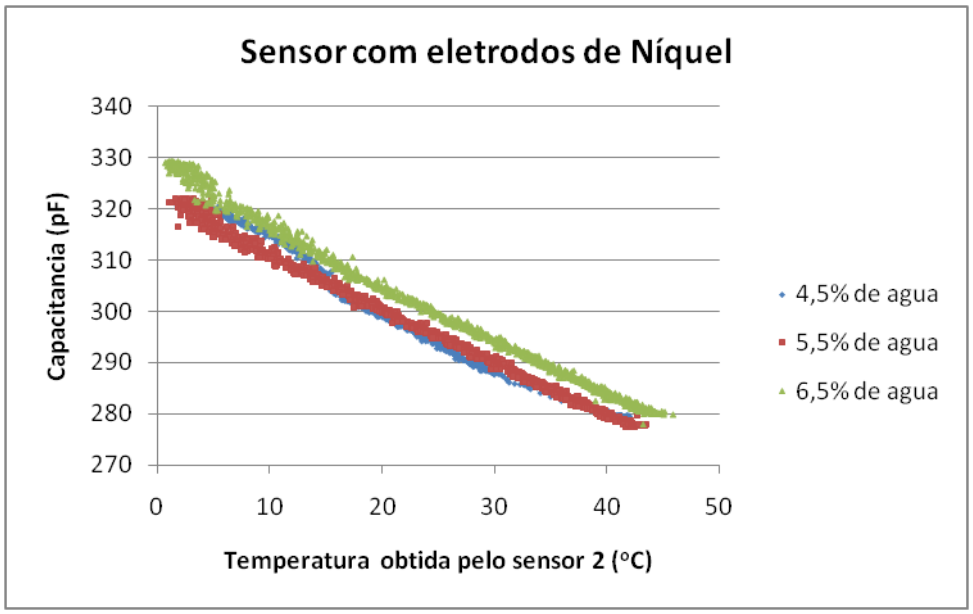

Figura 6.37- Capacitância obtida pelo sensor capacitivo com eletrodos de niquel versus temperatura obtida pelo segundo sensor de temperatura

Ao analisar os gráficos nota-se que há grande sobreposição dos dados, principalmente entre as concentrações de 4,5% e 5,5% de água diluída no etanol anidro. Devido à maior influência do substrato de fibra de vidro se comparado ao substrato de alumina, o distanciamento dos valores de capacitância obtidos referente à concentração de 6,5% de água diluída em etanol foi ampliado (Figuras 6.34 e 6.35), se comparado com os valores obtidos pelo sensor de níquel (Figuras 6.36 e 6.37).

Mais uma vez, as interferências entre as curvas dificultam a classificação das amostras de acordo com suas concentrações de água e transforma a tarefa de classificação de combustível conforme e não conforme em um problema não trivial.

6.7.4 Pré-processamento dos dados e extração de padrões

O problema foi estudado como binário e assim, o objetivo desse experimento é classificar os exemplos em duas classes de acordo com a concentração volumétrica de água na mistura. Os exemplos positivos utilizados foram os de combustível conforme de acordo com a legislação brasileira e os exemplos negativos foram os não conformes. Mais uma vez essa informação foi inserida manualmente na base de dados.

Tabela 18 - Distribuição dos exemplos positivos e negativos

Concentração volumétrica de água em etanol (%)	Exemplos	Classificação	Classe
4,5%	858	Não conforme	Exemplos Negativos
5,5%	1063	Conforme	Exemplos Positivos
6,5%	1169	Não conforme	Exemplos Negativos
Total	3090	-	-

Os dados foram primeiramente divididos de forma aleatória entre conjunto de treinamento e conjunto de teste, na proporção apresentada na tabela abaixo.

Tabela 19 - Distribuição dos exemplos para treino e teste

Conjunto	Exemplos	Classe
Treino	2000	688 exemplos positivos
		1312 exemplos negativos
Teste	1090	375 exemplos positivos
		715 exemplos negativos
Total	3090	-

Em seguida, a partir do conjunto de exemplos de treinamento, foram gerados subconjuntos contendo 300, 500, 1000, 1500 e 2000 exemplos, também de forma aleatória.

Por fim, todos esses conjuntos foram utilizados para treinar classificadores e os resultados serão apresentados a seguir.

6.7.5 Avaliação dos resultados obtidos

Os resultados serão apresentados na forma de curva ROC – Receiver Operating Characteristic – para avaliar o desempenho dos classificadores gerados. As curvas de erro de treinamento e teste também são apresentadas.

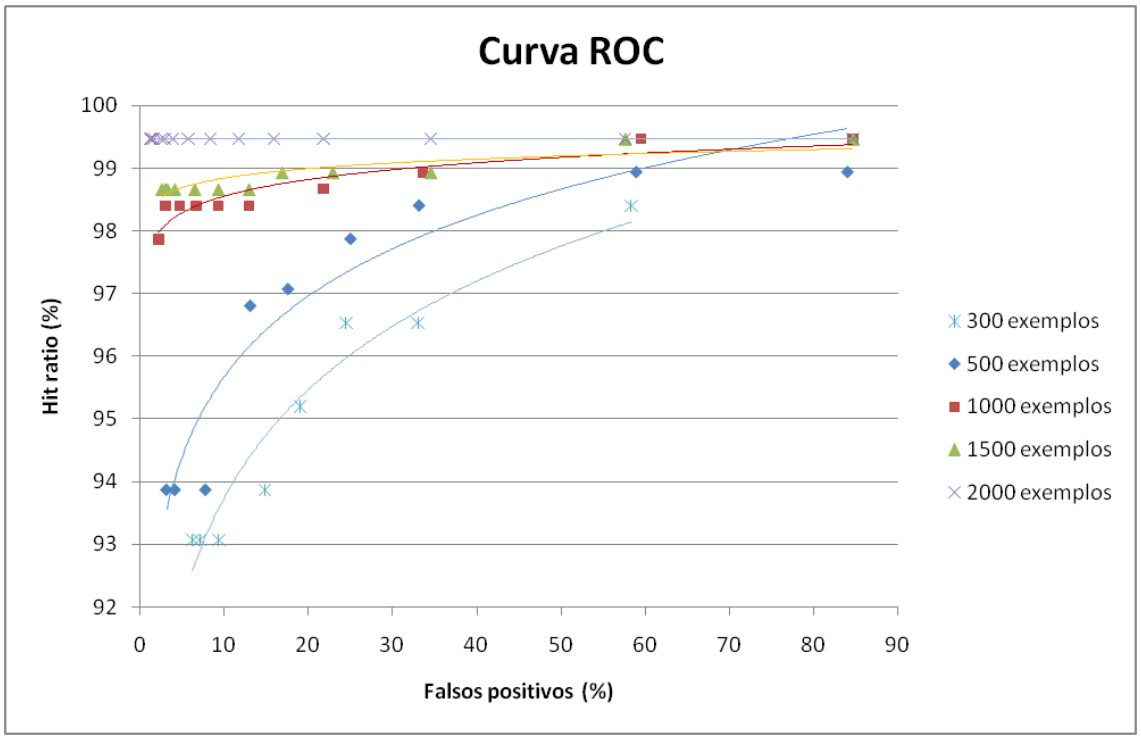

Figura 6.38 - Curva ROC obtida a partir de diferentes conjuntos de treinamento

Observando o gráfico acima (Figura 6.38) é possível verificar que o desempenho do classificador foi gradativamente melhorado conforme o número de exemplos no conjunto de treinamento foi aumentado. Por exemplo, para o conjunto de treinamento composto por 300 exemplos o classificador atingiu 93% de hit ratio (correta classificação de combustível conforme) e 6,5% de falso positivo (combustível não conforme classificados como conforme). Por outro lado, para o conjunto de treinamento formado por 2000 exemplos, foi possível atingir os índices de 99,4% de hit ratio e 1,3% de falso positivo.

É válido destacar neste experimento o êxito no processo de calibração, especialmente devido à curva apresentada pelo sensor de cobre (Figuras 6.34 e 6.35). Nota-se que há um ponto de não-linearidade nas curvas apresentadas referentes ao sensor de cobre, em aproximadamente 32^{o}C de temperatura. Tal não-linearidade é mais acentuada para a concentração de 5,5% em volume de água diluída na mistura. O processo de calibração por meio de aproximações de curvas ou funções seria bastante complicado, especialmente em problemas envolvendo sistema de sensores. Além disso, tal aproximação possivelmente acarretaria em erros e conseqüente depreciação da acurácia do sistema. No entanto, devido à forma de aprendizado e funcionamento do algoritmo AdaBoost, o processo de calibração foi realizado com sucesso.

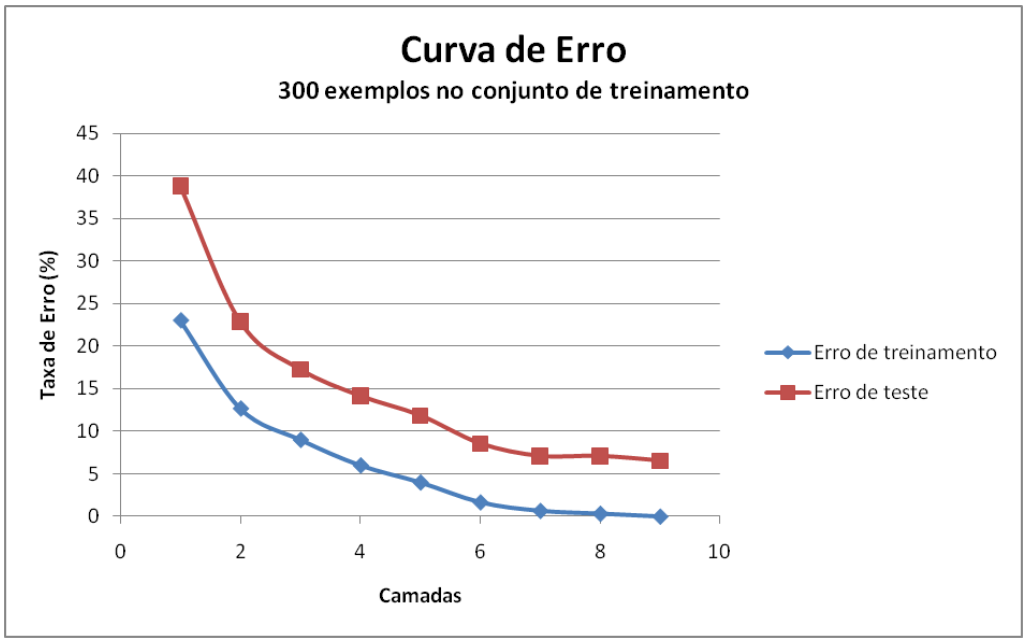

Figura 6.39 - Curvas de erro de treinamento e teste obtidas a partir de 300 exemplos de treinamento

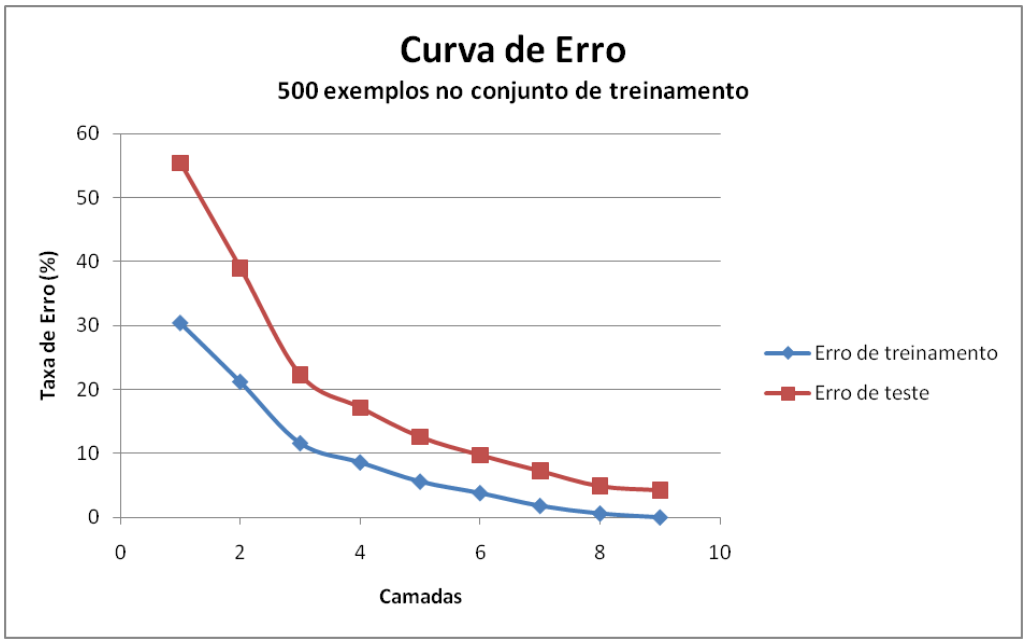

Figura 6.40 - Curvas de erro de treinamento e teste obtidas a partir de 500 exemplos de treinamento

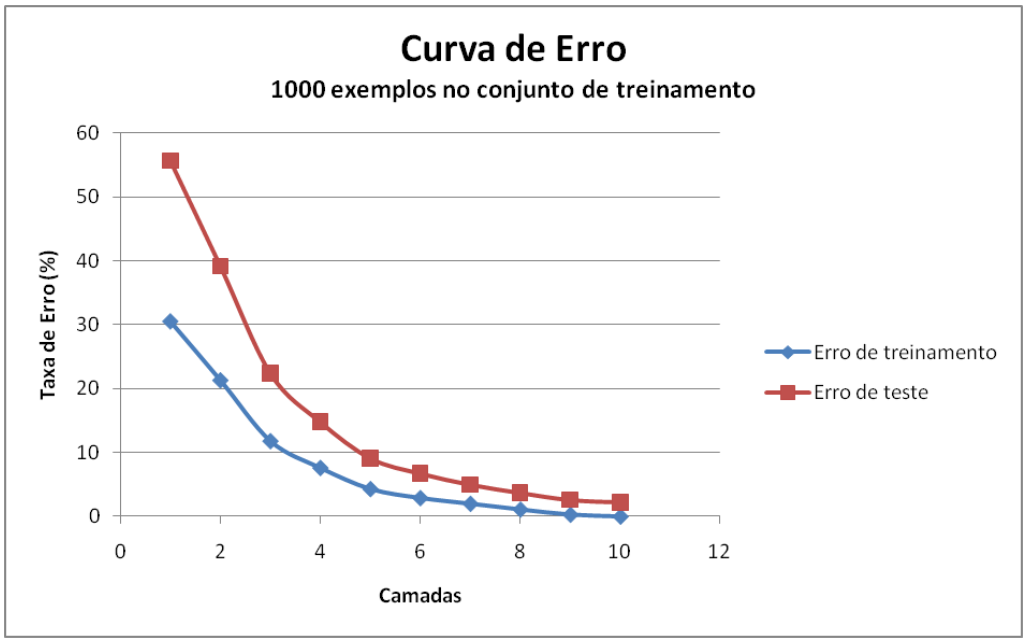

Figura 6.41 - Curvas de erro de treinamento e teste obtidas a partir de 1000 exemplos de treinamento

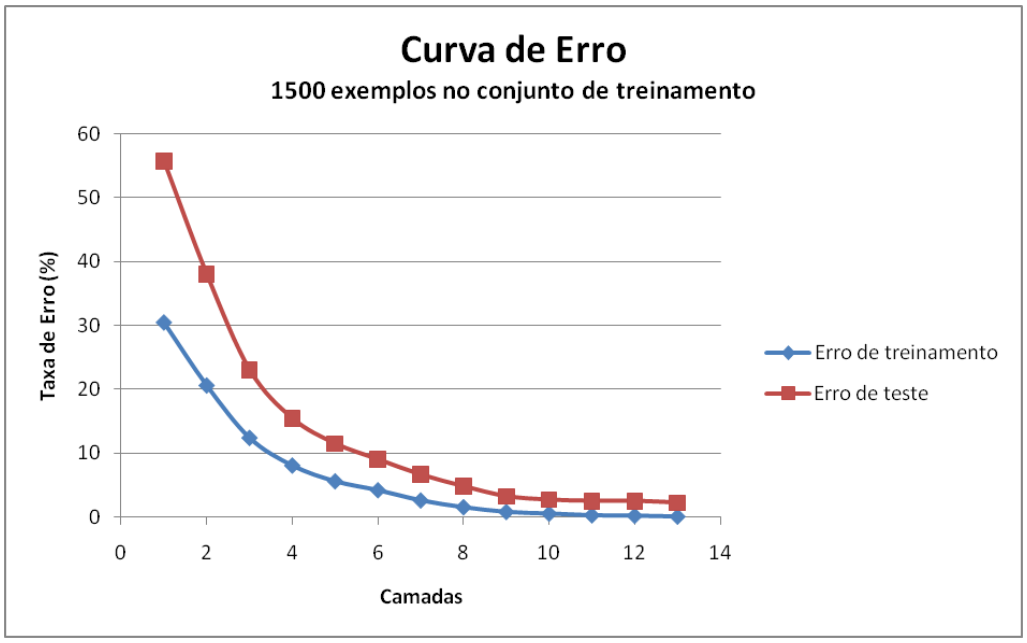

Figura 6.42 - Curvas de erro de treinamento e teste obtidas a partir de 1500 exemplos de treinamento

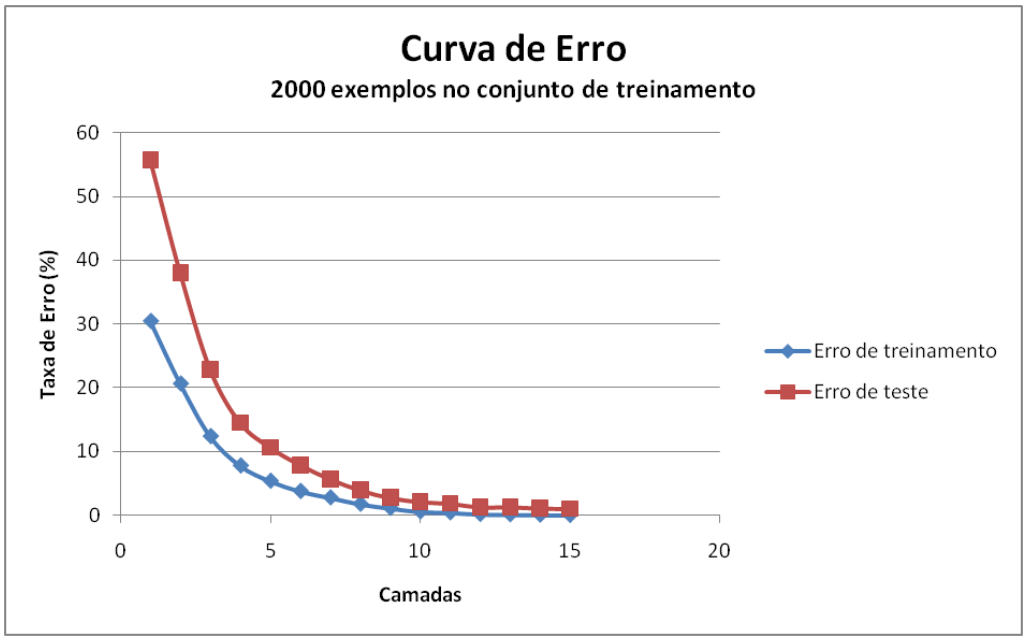

Figura 6.43 - Curvas de erro de treinamento e teste obtidas a partir de 2000 exemplos de treinamento

Os dados também foram analisados de forma a comparar o desempenho de cada sensor capacitivo separadamente com o desempenho do sistema composto pelos dois sensores capacitivos. Os dois sensores de temperatura foram utilizados em todos os casos. Desta forma, ao invés de 6 parâmetros presentes no sistema, cada subsistema contém um sensor capacitivo e dois sensores de temperatura, totalizando 4 parâmetros analisados (temperatura do primeiro sensor, temperatura do segundo sensor, período e capacitância do único sensor capacitivo).

Os mesmos procedimentos de obtenção de conjunto de treinamento e teste foram realizados. A seguir são apresentados os resultados dos classificadores gerados a partir de 300, 500 e 1000 exemplos no conjunto de treinamento. Os gráficos apresentam os desempenhos dos dois sensores capacitivos em comparação com o sistema composto pelos dois sensores.

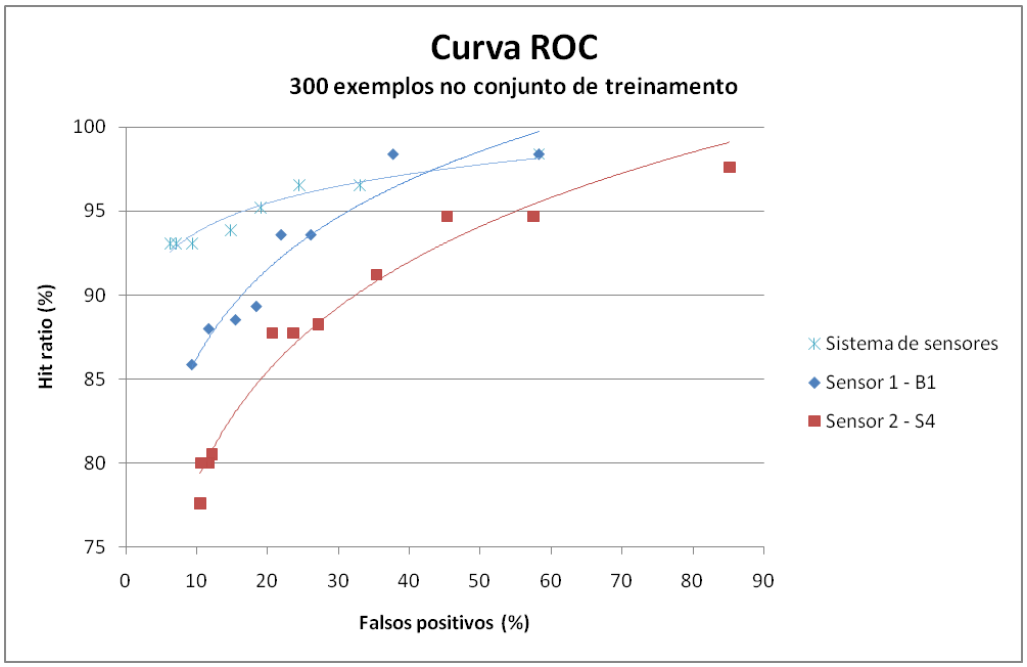

Figura 6.44 - Curva ROC obtida pelos sensores capacitivos de forma independente e também de forma conjunta a partir de 300 exemplos no conjunto de treinamento

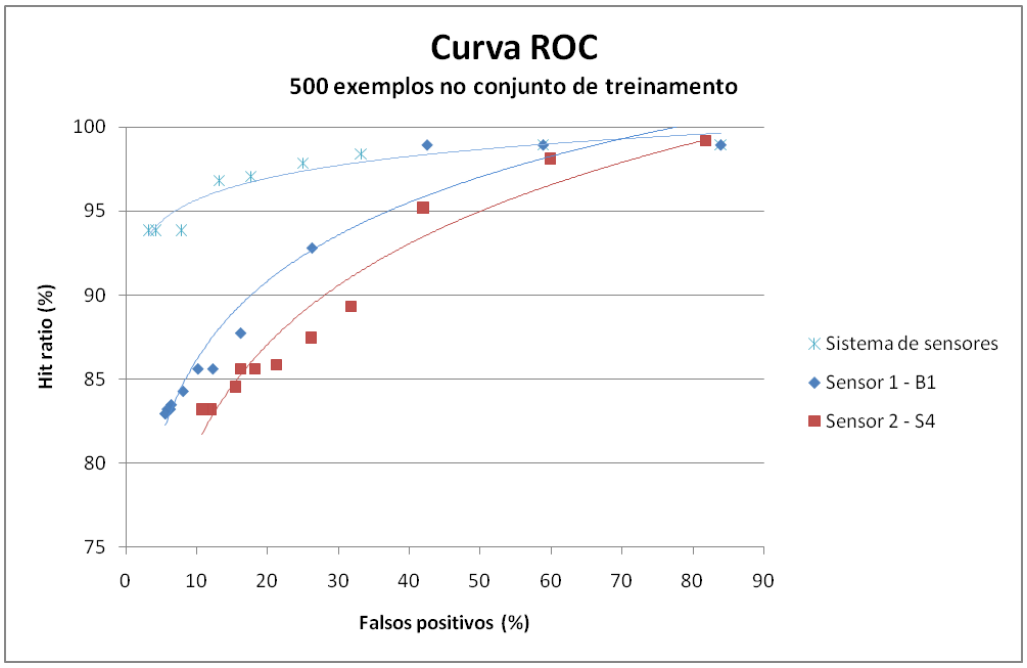

Figura 6.45 - Curva ROC obtida pelos sensores capacitivos de forma independente e também de forma conjunta a partir de 500 exemplos no conjunto de treinamento

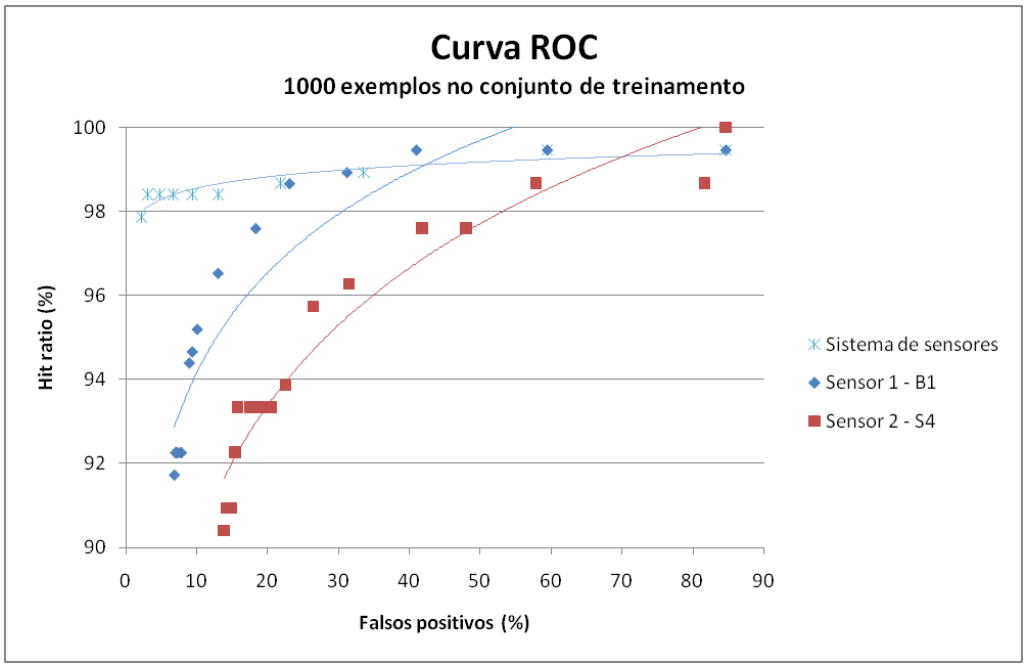

Figura 6.46 - Curva ROC obtida pelos sensores capacitivos de forma independente e também de forma conjunta a partir de 1000 exemplos no conjunto de treinamento

É possível verificar em todos os casos que o desempenho do sistema de sensores foi melhor se comparado a cada subsistema, tendo em vista que a respectiva curva encontra-se acima das outras duas curvas em todos os casos. Isso se deve ao fato de fornecer maior quantidade de informação para o algoritmo na etapa de treinamento. Em outras palavras, apesar de mensurarem a mesma propriedade física, os sensores apresentam estruturas distintas, além se terem sido fabricados com materiais diferentes para o substrato e eletrodos. Assim, as informações de cada sensor tornam-se, na prática, novos atributos para o caso do sistema de sensores.

Por exemplo, para o gráfico referente a classificadores obtidos a partir de conjunto de treinamento contendo 1000 exemplos, o subsistema formado pelo sensor capacitivo 2 obteve 90,4% de hit ratio e 13,8% de falso positivo. Por outro lado, o subsistema composto pelo sensor capacitivo 1 apresentou 91,73% de hit ratio e 6,8% de falso positivo. Por fim, o melhor desempenho apresentado foi o gerado pelo sistema que atingiu os índices de 97,8% de hit ratio e 2,3% de falso positivo.

Dois aspectos principais podem ser destacados após a análise desse experimento. O primeiro deles refere-se aos índices atingidos. O desempenho dos classificadores gerados sugere ser possível atingir um nível de classificações corretas muito elevado, acima de 95%. Esses resultados reforçam a grande contribuição e utilidade de se usar algoritmos de reconhecimento de padrões, como o AdaBoost, aliado a conjunto de sensores.

O segundo aspecto refere-se ao desempenho de sensores isolados e sistema composto por diferentes sensores. Os resultados sugerem ser possível aprimorar os resultados obtidos utilizando sensores diferentes e de forma paralela mesmo sem realizar modificações ou melhorias em cada sensor separadamente. Utilizando-se o algoritmo AdaBoost foi possível elevar a precisão e acurácia obtidas para a classificação das mesmas amostras de combustíveis adulterados e não adulterados por meio da combinação das respostas de cada sensor capacitivo.

6.8 Estudo de caso 7: Conjunto de Sensores de Temperatura e de Fibra óptica

6.8.1 Aspectos Gerais

Para a sétima e última experiência aqui apresentada, foram realizados os procedimentos da experiência anterior, com o objetivo de validar a utilização do algoritmo AdaBoost com estrutura de cascata como método de avaliação da qualidade de etanol combustível em diferentes temperaturas utilizando-se um sensor de fibra óptica e dois sensores de temperatura do tipo termopar K1.

6.8.2 Obtenção dos dados

Para a obtenção dos dados foi seguido o procedimento descrito no item 6.7.2 deste trabalho, com a coleta dos seguintes parâmetros: temperatura da amostra em (ºC), intensidade luminosa refletida (%), intensidade luminosa do laser (%) e temperatura interna do sensor (ºC). Portanto, quatro atributos foram utilizados no experimento.

6.8.3 Análise dos Dados

Sendo cada exemplo um conjunto de informações coletados em um mesmo instante sobre a amostra, a base de dados foi composta por um total de 3120 exemplos, sendo 1040 exemplos coletados em cada uma das três concentrações avaliadas (4,5%, 5,5% e 6,5%).

Fazendo-se uma análise estatística simples (Figura 6.47), observa-se uma grande sobreposição nos dados, especialmente nas concentrações 5,5% e 6,5% de água diluída em etanol, onde a classificação dos dados é dificultada pelas interferências apresentadas.

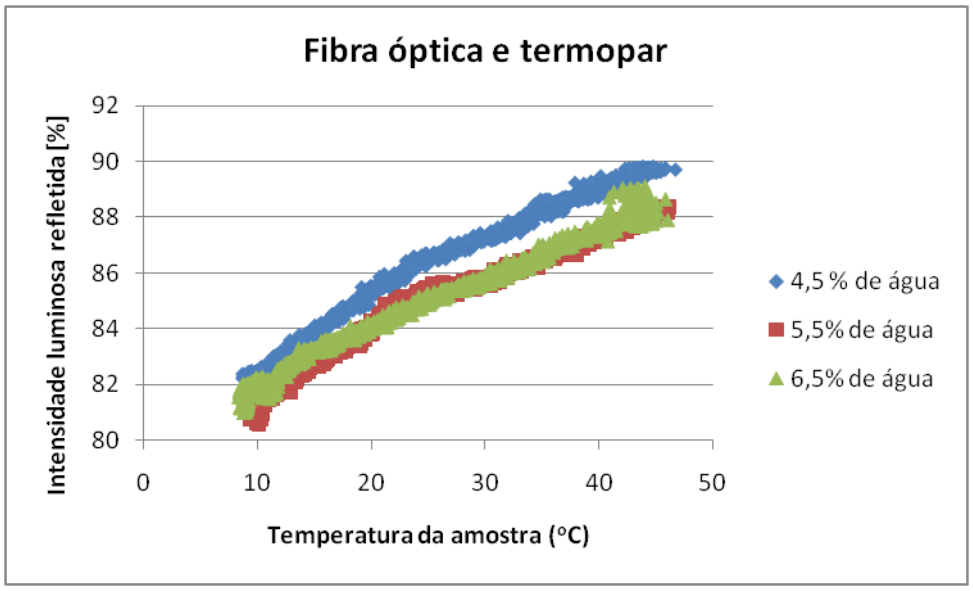

Figura 6.47 - Gráfico da relação dos dados analisados

6.8.4 Pré-processamento dos dados e extração de padrões

O problema foi estudado de forma binária, onde os exemplos positivos são aqueles referentes à amostra de 5,5% de água diluída em etanol, e os exemplos negativos, as amostras com 4,5% e 6,5% de água na mistura. Os dados foram divididos de forma aleatória entre o conjunto de treinamento e conjunto de teste, na proporção apresentada na tabela abaixo.

Tabela 20 - Distribuição dos exemplos para treino e teste

Conjunto	Exemplos	Classe
Treino	2000	667 exemplos positivos
		1333 exemplos negativos
Teste	1120	373 exemplos positivos
		747 exemplos negativos
Total	3120	-

A partir do conjunto de treinamento foram gerados aleatoriamente conjuntos contendo 1000, 1250, 1500, 1750 e 2000 exemplos. Então, esses conjuntos foram utilizados para treinar classificadores e os resultados serão apresentados a seguir.

6.8.5 Avaliação dos resultados obtidos

Novamente os resultados serão apresentados na forma de curva ROC - Receiver Operating Characteristic - para avaliar o desempenho dos classificadores produzidos.

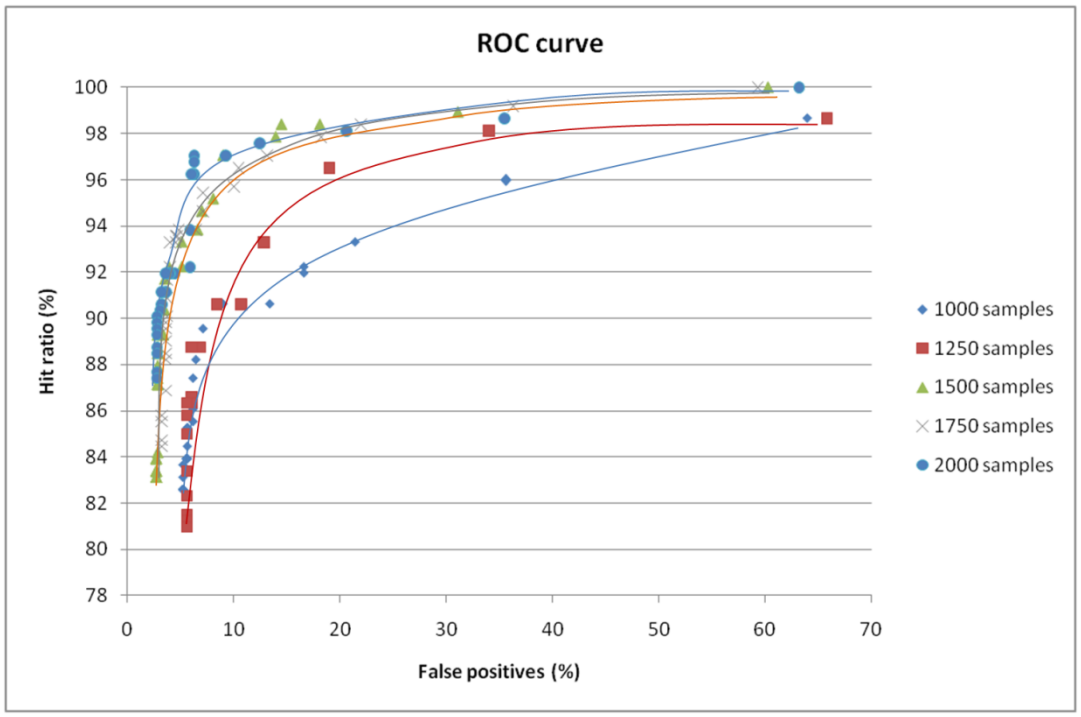

Figura 6.48 - Curva ROC obtida a partir dos dados da experiência

Como esperado, o desempenho dos classificadores foi melhorando conforme o aumento do número de exemplos de treinamento. No entanto, é possível verificar que a partir de 1500 exemplos de treinamento a melhoria do desempenho dos classificadores foi menos acentuada, sendo que a curva referente a 2000 exemplos de treinamento forneceu um classificador de destaque, tendo apresentado índices de 96,2% de hit ratio e 6,3% de falsos positivos.

As curvas de erro de treinamento e de teste referentes aos conjuntos de treinamento contendo 1500, 1750 e 2000 exemplos são apresentadas a seguir (Figuras 6.49 a 6.51).

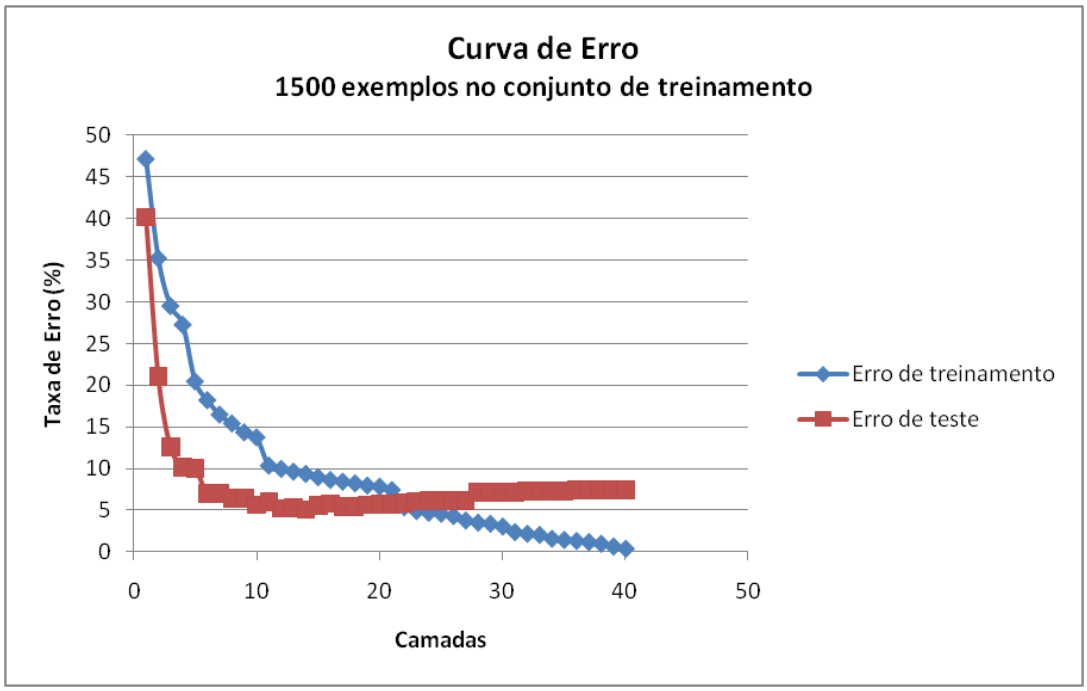

Figura 6.49 - Curva de erro referente ao conjunto de 1500 exemplos de treinamento

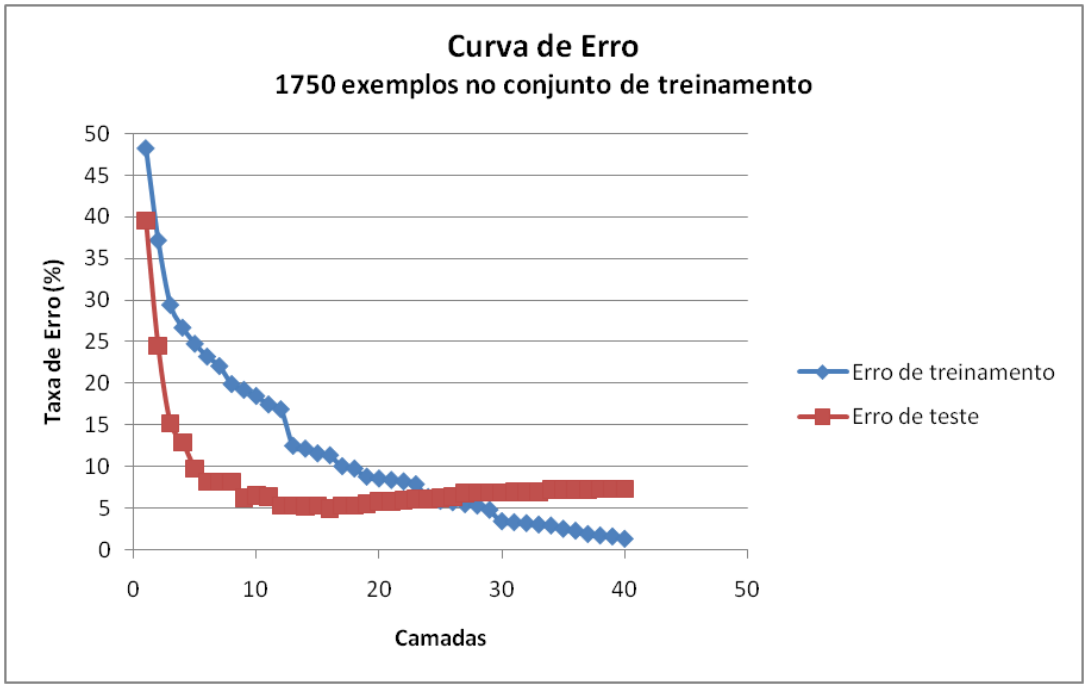

Figura 6.50 - Curva de erro referente ao conjunto de 1750 exemplos de treinamento

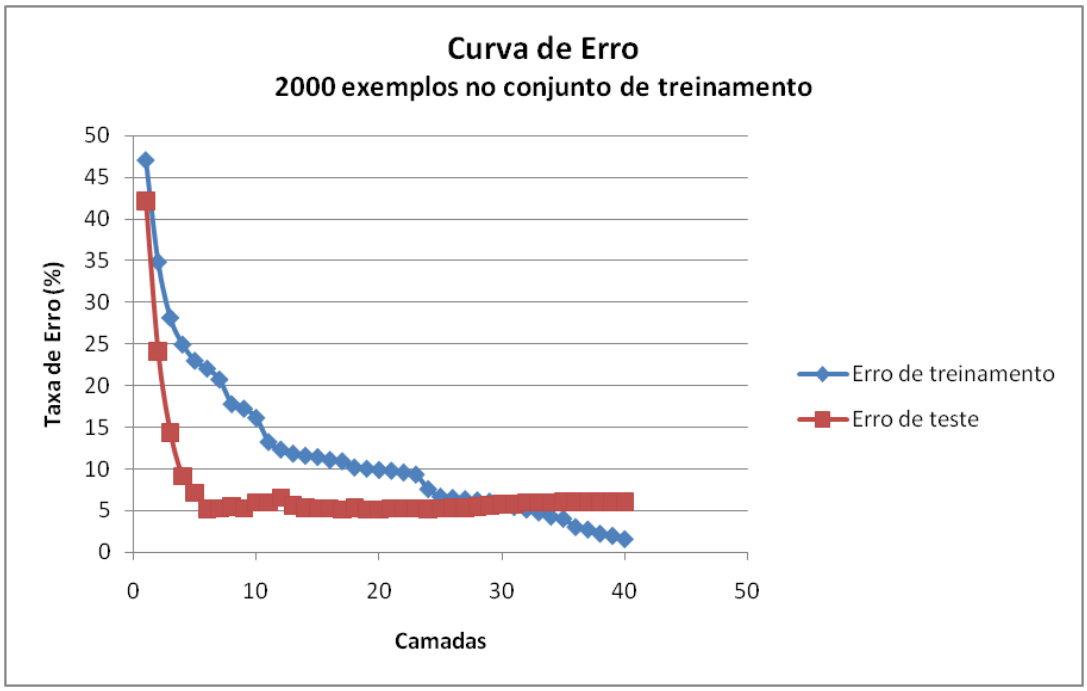

Figura 6.51 - Curva de erro referente ao conjunto de 2000 exemplos de treinamento

As curvas apresentam tendências semelhantes de zerar o erro de treinamento próximo a 40 camadas. Apesar disso, a curva de erro de teste apresenta a tendência de estabilizar em aproximadamente 5% após poucas camadas. Por exemplo, a curva de erro de teste referente a 2000 exemplos de treinamento atingiu este índice após 7 camadas. Em outras palavras, após poucas camadas de treinamento foi possível produzir classificadores com as melhores relações hit ratio versus falsos positivos, apresentados no gráfico da curva ROC (Figura 6.48). Consequentemente, essa é uma vantagem para o desenvolvimento de sistemas embarcados, pois esses classificadores são menores devido à menor quantidade de camadas de treinamento se comparados aos outros classificadores produzidos. Além disso, outra vantagem de se utilizar a estrutura em cascata para sistemas embarcados refere-se ao fato de o microprocessador não utilizar toda a estrutura do classificador, caso o dado de leitura seja classificado como negativo.

7 Embarcação do sistema de classificação pelo algoritmo AdaBoost

7.1 Visão Geral

Em virtude da importância e utilidade de sistemas embarcados na análise e tomada de decisão nas mais diversas áreas e aplicações, o último objetivo desta pesquisa foi estudar e desenvolver um meio de embarcar o classificador treinado, capaz de realizar análises de novos cenários em tempo real e de forma autônoma, sem que seja necessário grande aparato eletrônico. Neste capítulo será detalhada a abordagem utilizada para viabilizar este processo.

7.2 Etapa de embarcação do classificador

O sistema de classificação embarcado proposto neste trabalho foi desenvolvido considerando-se o custo computacional e acurácia da resposta do sistema, visando o desenvolvimento de um dispositivo autônomo capaz de ser utilizado em aplicações em tempo real e de baixo custo. Desta forma, devido ao custo computacional e também conveniência para o processo, toda a etapa de treinamento do classificador foi realizada em um computador desktop. Uma vez construída a curva ROC do treinamento e tendo escolhido o classificador mais adequado para o problema, apenas os valores correspondentes ao respectivo classificador são embarcados.

O microprocessador escolhido para o projeto foi o Arduino Mega 1280 devido ao tamanho da memória disponível, facilidade de manipulação, plataforma de programação gratuita, baixo custo e linguagem de programação similar a C++. A Figura 7.1 mostra o microcontrolador montado em uma placa comercial para o Arduino Mega1280.

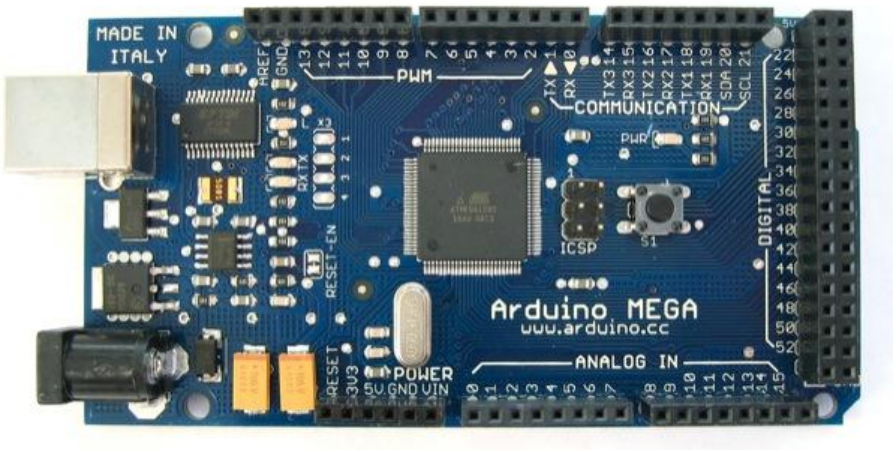

Figura 7.1 - Placa do microprocessador Arduino Mega 1280 utilizado no projeto

Por ter sido escolhida a estrutura cascata do algoritmo AdaBoost para esta pesquisa, o código de programação escrito para ser embarcado no microprocessador foi desenvolvido de forma a se adaptar a essa estrutura. A Figura 7.2 a seguir resume o algoritmo escrito.

```
// Dados do respectivo classificador a ser embarcado

void loop() {

   ... // Extração de dados – código referente à leitura da saída dos sensores conectados ao sistema

          Para todos os classificadores fracos de todas as camadas;


          Verificar se o atual exemplo é positivo ou negativo de acordo com os limites (thresholds) obtidos em
          cada iteração da etapa de treinamento;
          Soma as respostas de todos os classificadores fracos;


          A camada atual classifica o exemplo como positivo em caso da soma ser positiva ou igual a zero;
          Caso contrário, o exemplo é classificado como negativo;


          Por fim, se o exemplo atingir o final da última camada como positivo, este é um exemplo positivo.

}
```

Figura 7.2 - Código a ser embarcado no microprocessador

Portanto, de forma correspondente, o código segue o formato da estrutura cascata (Figura 7.3) que já foi apresentada anteriormente na Seção 3.4.2.

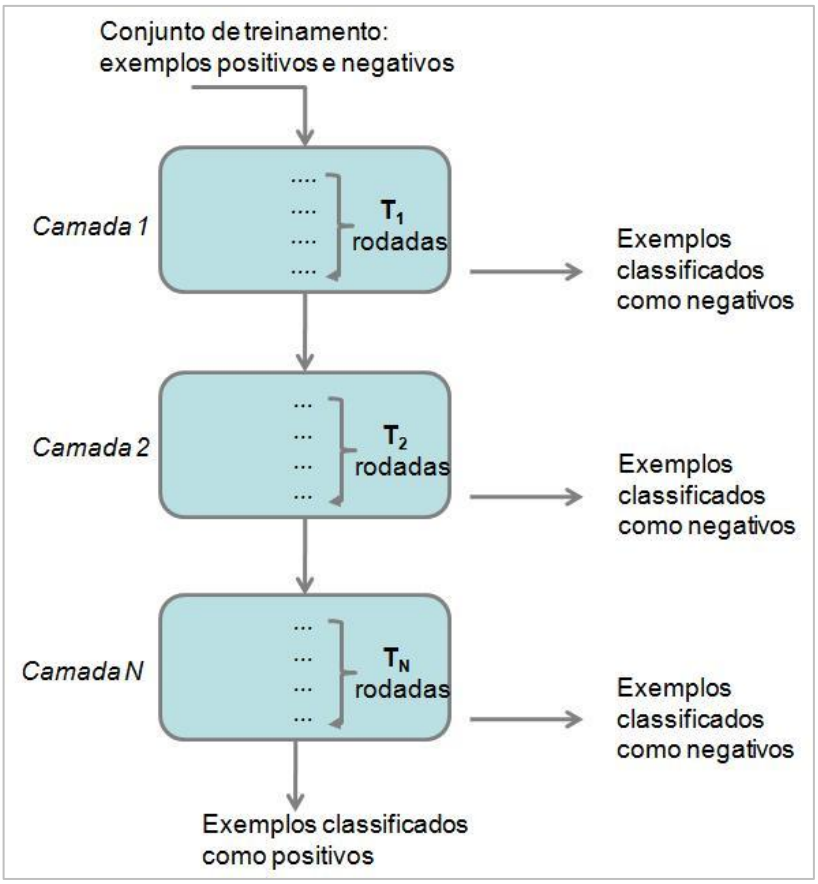

Figura 7.3 - Esquema da estrutura Cascata do algoritmo AdaBoost

Conforme explicado em detalhes durante o exemplo didático descrito na Seção 3.3, o classificador produzido após a etapa de treinamento é composto por informações como:

• Camadas: número de camadas do classificador

• Classificadores: número total de classificadores fracos

• Alfa: coeficiente alfa de cada iteração que representa a importância associada do classificador (Definido também como γ na Seção 3.2).

• Threshold: o limite ou borda estipulado por iteração

• Tipo: qual a dimensão avaliada em cada iteração

Por tanto, por meio da utilização dos valores dos limites (*threshold*) é possível classificar novos exemplos.

A partir da Figura 7.2 nota-se que este código desenvolvido é flexível e pode ser utilizado junto a diferentes sensores, independente de sua propriedade física analisada. Este pedaço de código de ativação e leitura dos sensores é incluído logo no início do programa, podendo então ser escrito e modificado de forma a se tornar conveniente para o caso particular estudado. No caso de estudo deste trabalho, os sensores capacitivos e de temperatura são ativados nessa etapa do processo para coletar informações sobre a amostra analisada. Além disso, a quantidade de sensores utilizados também é adaptada nessa parte do código. Porém, esse número de sensores deve ser coerente com aqueles utilizados durante a etapa de treinamento para produzir o classificador.

Portanto, o sistema desenvolvido tem apresenta vasta área de aplicações e utilidades. Tal sistema pode ser utilizado para desenvolver dispositivos portáteis da área médica, sistemas de agricultura de precisão, aplicações em automóveis ou no setor aeroespacial, entre outros.

Como caso de estudo, foi desenvolvido um protótipo de dispositivo portátil para avaliação da qualidade de etanol combustível contendo um classificador treinado pelo algoritmo AdaBoost com estrutura em cascata. Para isso, foi utilizado o classificador gerado durante o experimento 6 descrito neste trabalho. Além do microprocessador Arduino Mega 1280, foi utilizado também um módulo display LCD modelo HD44780 Pic Atmel para apresentar os resultados das classificações. Para esse dispositivo foi realizada uma simulação para verificar seu funcionamento. Assim, dados de amostras de etanol combustível adulterado e não adulterado foram fornecidos ao microprocessador de forma seqüencial sem qualquer classificação, para que este pudesse utilizar o classificador embarcado e então, imprimir no display se aquela amostra era adulterada ou não adulterada. Dessa forma, o display apresenta as seguitnes informações:

- Ex: número da amostra analisada
- T(C): Temperatura da amostra
- Classificação: Adulterado ou Não Adulterado

As Figuras 7.4 e 7.5 apresentam ilustrações obtidas durante a utilização do dispositivo. A Figura 7.6 apresenta o dispositivo completo, onde é possível verificar a fonte externa de alimentação, o microprocessador contendo o algoritmo embarcado e o display LCD.

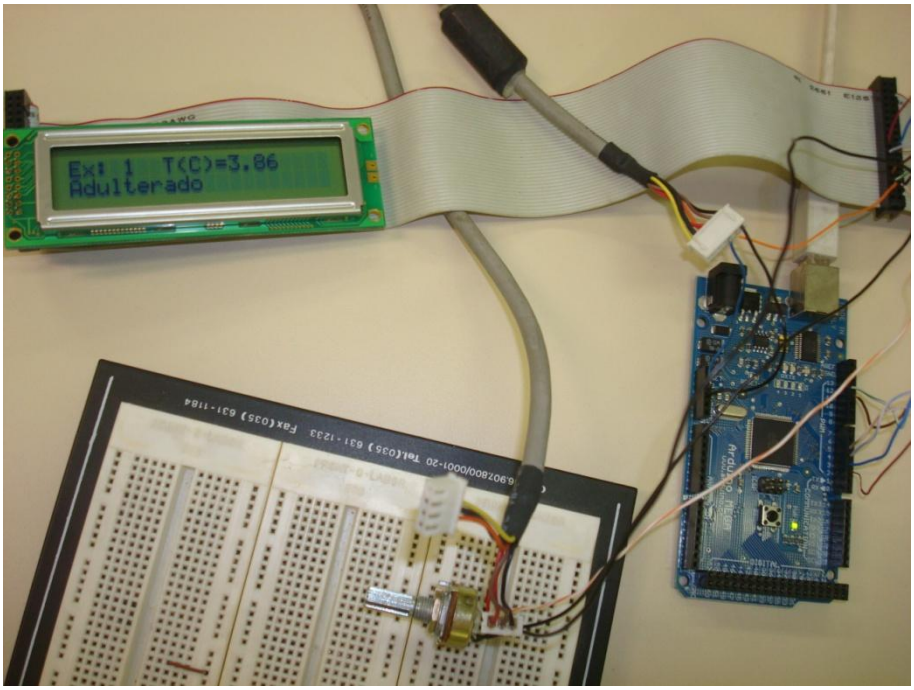

Figura 7.4 - Dispositivo em funcionamento analisando amostra de etanol adulterada

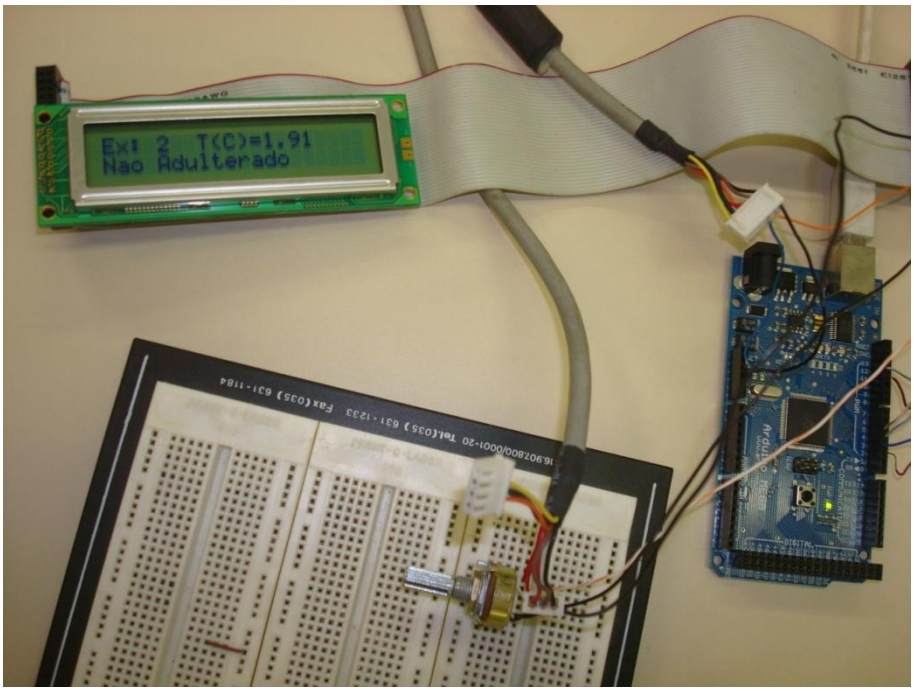

Figura 7.5 - Dispositivo em funcionamento analisando amostra de etanol não adulterada

De maneira geral, o dispositivo mostrou-se bastante eficiente, tendo trabalhado de forma adequada e em tempo real. O mesmo conjunto de exemplos de combustível adulterado e não adulterado utilizado no microprocessador foi classificado em um computador Desktop e seus

resultados foram comparados aos fornecidos pelo microprocessador, sendo estes 100% coerentes.

Outro teste realizado foi apresentar ao microprocessador os 1090 exemplos de teste utilizados durante a etapa de treinamento da experiência 6. Novamente todas as classificações apresentadas pelo microprocessador foram iguais aquelas fornecidas no computador Desktop, comprovando assim a eficiência e precisão do dispositivo.

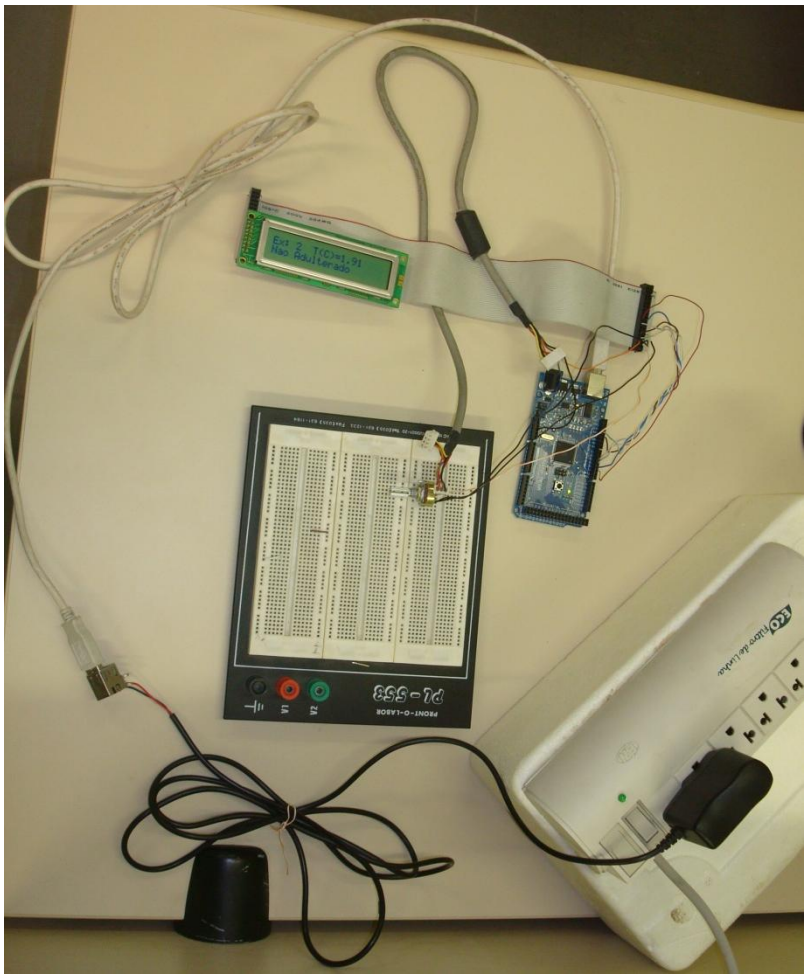

Figura 7.6 - Dispositivo completo

8 Conclusões

8.1 Do AdaBoost Aplicado a sensores

De forma geral, o projeto de aplicação do algoritmo AdaBoost para analisar e classificar dados obtidos por meio de sensores mostrou-se viável e atendeu aos objetivos propostos.

Os sete experimentos demonstraram ser possível aperfeiçoar a etapa de calibração dos sensores, uma vez que a partir um de baixo número de exemplos (dados) é possível atingir elevados índices de classificações corretas por meio da utilização de algoritmos de aprendizagem de máquina e, em especial do AdaBoost junto a algoritmos baseados em árvores de decisão para gerar os classificadores fracos. Índices acima de 90% de classificações corretas foram obtidos, indicando a viabilidade deste processo.

Dentre os exemplos analisados, concluiu-se também que maior a quantidade de parâmetros do cenário analisado podem acarretar na melhora da resposta do sistema. A extração de dados a partir de um maior número de sensores acrescenta informações como novos atributos e assim aprimoram análises e classificações de problemas mais complexos. Esse enriquecimento de informações também se mostrou viável para elevar o índice de acertos das classificações sem que seja necessário realizar melhorias ou modificações nos sensores propriamente ditos.

8.2 Do Sistema de Embarcação

Outro resultado importante obtido foi o sucesso na embarcação do classificador treinado em um microprocessador de baixo custo e memória. Isto só foi possível devido à estrutura de treinamento em cascata do algoritmo AdaBoost utilizada no estudo, que produz classificadores compactos.

A partir do teste de embarcação, aliado a rapidez nas etapas de aprendizado e classificação, além dos índices de acerto obtidos nos experimentos e também do baixo custo computacional requerido pelo algoritmo, é possível concluir ser viável desenvolver dispositivos totalmente independentes, que tenham embarcado o classificador treinado previamente, e assim, contenham a tecnologia de reconhecimento de padrões.

Como caso de estudo, foi desenvolvido um protótipo de dispositivo portátil para avaliação da qualidade de etanol combustível contendo um classificador treinado pelo algoritmo

AdaBoost com estrutura em cascata. Porém, o mesmo procedimento utilizado para reconhecimento de adulteração de etanol combustível pode ser utilizado para desenvolver dispositivos autônomos contendo a tecnologia de reconhecimento de padrões para outras finalidades ou aplicações em áreas de conhecimento distintas, como a área médica, agrícola de precisão, aeroespacial, automobilística, telefonia móvel, entre outros.

9 Trabalhos Futuros

O trabalho desenvolvido neste projeto mostrou-se bastante promissor e abrangente, permitindo o projeto de dispositivos capazes de realizar análises e classificações de cenários não triviais. Além de tudo, a aplicação do algoritmo AdaBoost no projeto de sensores ou rede de sensores mostrou um grande potencial de expansão.

Como próximas pesquisas seria interessante estudar novos classificadores fracos para serem utilizados de forma conjunta com o algoritmo AdaBoost, tendo como objetivo melhorar ainda mais os índices de acerto obtidos nos testes, além de ampliar as possibilidades de aplicação dessa tecnologia para casos ainda mais complexos de automação e reconhecimento de padrões.

Por fim, é importante estudar e ampliar a utilização do algoritmo AdaBoost aplicado a sensores para casos não apenas binários, mas multiclasses. Muitas aplicações interessantes necessitam dessa flexibilidade para ter sua utilização bem sucedida.

10 Referências Bibliográficas

Avidan, S. Object classification using image segmentation. Estados Unidos Patente 20060018521. 26 de Janeiro de 2006.

Avidan, S., M. Butman, e A. Butman. Method for secure component labeling in images. Estados Unidos Patente 20060123245. 8 de Junho de 2006.

Babenko, D., e H. Marmanis. *Algorithms of the Intelligence Web.* Manning Publications, 2009.

Barczak, A. L. C., Johnson, M. J., Messom, C. H. "Empirical Evaluation of a New Structure for AdaBoost." *Proceedings of the 2008 ACM symposium on Applied Computing.* Fortaleza, Ceará, Brazil, 2008.

Bergstra, J., Casagrande, N., Erhan, D., Eck, D., K´egl, B. "Aggregate Features and AdaBoost for Music Classification." *Machine Learning*, 2007: 473-484.

Bernardini, F. C. *Combinação de classificadores simbólicos para melhorar o poder preditivo e descritivo de ensembles.* Dissertação de Mestrado, São Carlos: Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2002.

Braga, L. P. V. *Introdução à mineração de dados.* Rio de Janeiro: E-papers Serviços Editoriais, 2005.

Breed, D. S. Vehicular impact reactive system and method. Estados Unidos Patente 20050278098. 15 de Dezembro de 2005.

Breiman, L. "Bagging Predictors." *Machine Learning 24*, 1996: 123 – 140.

Breiman, L. "Random Forests." *Machine Learning, Vol 45*, 2001: 5-32.

Buchanan, B.G. "A (Very) Brief History of Artificial Intelligence." *AI Magazine, Vol 26*, 2005: 53-60.

Chen, S., H. Nicponski, e L. Ray. Method and system for face detection in digital images. Estados Unidos Patente 20040179719. 16 de Setembro de 2004.

Creamer, G., e Y. Freund. "Using AdaBoost for equity investment scoredcards." *Workshop Machine Learning in Finance, NIPS 2005.* Whistler, British Columbia, Canada, 2005.

Dalmon, D. L., E. Gusken, e C. K. Suzuki. "Sensor óptico para determinação de concentração em misturas de combustíveis." *4º Congresso Brasileiro de Pesquisa e Desenvolvimento em Petróleo e Gás*, 2007: 376-1.

Dietterich, T. G. "Ensemble methods in machine learning." *In First International Workshop on Multiple Classifier Systems*, June 21-23 de 2000: 1-15.

Duda, R. O., P. E. Hart, e D. G. Stork. *Pattern Classification- second edition.* New York: John Wiley & Sons. Interscience, 2001.

Eck, D., P. Lamere, T. B. Mahieux, e S. Green. "Automatic Generation of Social Tags for Music Recommendation." *Advances in Neural Information Processing Systems, Vol. 20*, 2007.

Everett, H. R. *Sensors for Mobile Robots: Theory and Application.* Wellesley: A. K. Peters, 1995.

Faceli, K. *Combinação de métodos de inteligência artificial para fusão de sensores.* Dissertação de Mestrado, São Carlos: Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2001.

Fayyad, U., G. Piatetsky-Schapiro, e P. Smyth. "From data mining to knowledge discovery: an overview." In: *Advances in knowledge discovery and data mining*, 1-34. The MIT Press, 1996a.

Fayyad, U., G. Piatetsky-Schapiro, e P. Smyth. "The KDD process for extracting useful knowledge from volumes of data." *Communications of the ACM, Vol 39*, 1996b: 27-34.

Fraden, J. *Handbook of Modern Sensors: Physics, Designs, and Applications.* San Diego, Califórnia: Springer; 2 edição, 1996.

French, P. J. "Smart Sensors: Advantages and Pitfalls." *Advanced Materials and Technologies for Micro/Nano-Devices, Sensors and Actuators. Springer*, 2010: 249-259.

Freund, Y, e R. E. Schapire. "Boosting the margin: A new explanation for the Effectiveness of voting methods." *The Annals of Statistics 26*, 1998: 1651 – 1686.

Freund, Y., e R. E. Schapire. "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting." *J. of Computer and System Sciences 55*, 1997: 119-139.

Freund, Y., e R. E. Schapire. "A short introduction to Boosting." *Journal of Japanese Society for Artificial Intelligence, Vol. 14, No. 5.*, 1999: 771-780.

Freund, Y., e R. E. Schapire. "Experiments with a New Boosting Algorithm." *Proceedings of the Thirteenth International Conference on Machine Learning*, 1996: 148-156.

Friedman, J., T. Hastie, e R. Tibshirani. "Additive logistic regression: a statistical view of boosting." *The Annals of Statistics, Vol 28*, 2000: 337–374.

Gardner, J., W. *Microsensors: principles and applications.* John Wiley & Sons Ldt, 1994.

Groover, M. P., M. Weiss, R. N. Nagel, e N. G. Odrey. *Robótica: Tecnologia e Programação.* McGraw-Hill, 1989.

Hall, D. L., e J. Llinas. *Handbook of Multisensor data fusion.* CRC Press LLC, 2001.

Haykin, S. *Neural Networks – A Compreensive Foundation - 2 edition.* New Jersey: Pretice-Hall, 1999.

He, H. and Garcia, E. A. "Learning from Imbalanced Data." *IEEE Transactions on knowledge and data engineering*, Setembro, 2009: vol. 21, no. 9.

Itoh, K., N. Komatsu, S. Shichijo, T. Yanagawa, e K. Murotani. MEDICAL DIAGNOSIS PROCESSOR. Japão Patente WO2007026773. 31 de Agosto de 2005.

Kavzoglu, T. "Increasing the accuracy of neural network classification using refined training data." *Environmental Modelling & Software 24*, 2009: 850 – 858.

Leshem, G. *Improvement of Adaboost Algorithm by using Random Forests as Weak Learner and using this algorithm as statistics machine learning for traffic flow prediction.* Proposta de pesquisa para Tese de Doutorado, The Hebrew University of Jerusalem, 2005.

Leshem, G., e Y. Ritov. "Traffic Flow Prediction using Adaboost Algorithm with Random Forests as a Weak Learner." *Transactions on engireering, computing and technology, Vol 19*, 2007: ISSN 1305-5313.

Leslie, C., Y. Freund, C. Wiggins, A. Kundaje, e M. Middendorf. "Predicting genetic regulatory response using classification." *Bioinformatics - Oxford University Press - Vol. 20*, 2004: i230 - i240.

Li, M., L. Zhang, Y. Sun, H. Zhang, e D. R. Parlin. Integrated solution to digital image similarity searching. Estados Unidos Patente 20050084154. 21 de Abril de 2005.

Lienhart, R., Liang, L., Kuranov, A. "A detector tree of boosted classifiers for real-time object detection and tracking." *IEEE*, 2003: 277-280.

Lorena, A. C., e A. C. P. L. F. Carvalho. *Revisão de Técnicas para Geração de Classificadores de Margens Largas Multiclasses.* Relatório Técnico, São Carlos: Instituto de Ciências Matemática e de Computação - USP, 2003.

Makinwa, K.A.A., M.A.P. Pertijs, J.C. Van der Meer, e J.H. Huijsing. "Smart sensor design: The art of compensation and cancellation." *Solid State Circuits Conference, 33ª edição*, 2007: 76 - 82.

Meijer, G. *Smart Sensor Systems.* John Wiley & Sons, 2008.

Mendez, A. "Fiber Bragg grating sensors: a market overview." *Proceedings of SPIE, European Workshop on Optical Fibre Sensors No3*, 2007: 661905.1-661905.6.

Mendonça, L. G. D. *Micro-sensor Capacitivo para avaliação da qualidade de combustíveis automotivos.* Dissertação de Mestrado, São Paulo: Escola Politécnica da Universidade de São Paulo, 2008.

Mendonça, L. G. D., B. Barazani, B. B. Chaves, D. Torikai, e R. C. Ibrahim. "Desenvolvimento de sensor Mems capacitivo para classificação e determinação de conformidade de combustível automotivo." *Simpósio Internacional de Engenharia Automotiva*, 2010.

Mendonça, L. G. D., et al. "Study of a copper capacitive MEMS as a sensor for automotive fuel evaluation." *21st International Congress of Mechanical Engineering*, 2011.

Merz, C. J. *Classification and Regression by Combining Models.* Tese de Doutorado, Irvine: University of California, 1998.

Mikut R., Reischl M. "Data mining tools." *WIREs Data Mining Knowl Discov*, 2011: 431-443.

Mitchell, T. *Machine Learning.* Portland: McGraw Hill, 1997.

Nabatame, S., e H. Iba. "Integrative Estimation of Gene Regulatory Network by Means of AdaBoost." *Japanese Society for Bioinformatics (JSBI)*, 2006.

Oza, N. C. *Online Ensemble Learning.* Tese de Doutorado, Berkeley: University of California, 2001.

Pugliesi, J. B., R. A. Sinoara, e S. O. Rezende. *Combinação de Regressores Homogêneos e Heterogêneos: Precisão e Compreensibilidade.* Relatório Técnico, São Carlos: Instituto de Ciências Matemática e de Computação da Universidade de São Paulo, 2003.

R., Ray. "What's next in sensor technologies?" *Control engineering, Vol 52*, 2005: 24.

Reyes, N. H., Barczak, A. L. and Messom, C. H. "Fast Colour Classification for Real-time Colour Object Identification: Adaboost training of Classifiers." *Proceedings of The Third International Conference on Autonomous Robots and Agents (ICARA 2006).* Palmerston North, New Zealand: Massey University, 2006.

Rezende, S. O. *Sistemas Inteligentes - Fundamentos e Aplicações.* Manole Ltda, 2003.

Russel, S., e P. Norvig. *Artificial Intelligence: A Modern Approach, 2 edição.* Prentice Hall, 2002.

Saleh, B. E. A., e M. C. Teich. *Fundamentals of photonics.* New York: John Wiley & Sons Inc, 1991.

Salomon, P. R. "Micro sensors – World wide markets and economic impact." *In Proceedings of 20th Eurosensors Conference*, 2006: 30-31.

Schapire, R. E. "An adaptive version of the boost by majority algorithm." *Machine Learning, Vol 43* , 2001: 293 -318.

Schapire, R. E., e Y. Singer. "Improved boosting algorithms using confidencerated predictions." *Machine Learning, Vol 37*, 1999: 297–336.

Schneiderman, H., Kanade, T. “A statistical method for 3d object detection applied to faces and cars.” *IEEE Conference on Computer Vision and Pattern Recognition*, 2000: 1746-1759.

Silberschatz, A., e A. Tuzhilin. “On subjective measures of interestingness in knowledge discovery.” *Proceedings of the first international conference on knowledge discovery and data mining*, 1995: 275-281.

Son, H., C. Kim, H. Kim, S. H. Han, e M. K. Kim. “Trend Analysis of Research and Development on Automation and Robotics Technology in the Construction Industry.” *KSCE Journal of Civil Engineering, Vol 14*, 2010: 131-139.

Souto, M. C. P., A. C. Lorena, A. C. B. Delbem, e A. C. P. L. F. Carvalho. “Técnicas de Aprendizado de Máquina para problemas de Biologia Molecular - Minicursos de Inteligência Artificial.” *XXIII Congresso da Sociedade Brasileira de Computação.* São Carlos, 2003.

Susnjak, T. *Accelerating classifier training using AdaBoost within cascades of boosted ensembles.* Dissertação de Mestrado, Auckland, Nova Zelândia: Massey University, 2009.

Tipler, P., e G. Mosca. F*ísica, v.2 - Para Cientistas e Engenheiros.* LTC, 2009.

Tripp, C., Hung, H., Pontikakis, M. *Waveform-Based Musical Genre Classification.* Projeto final de conclusão de curso, Stanford University, 2006.

Tumer, K. and Ghosh, J. *Classifier combining: Analytical results and implications.* Portland: Proc. Nat'l Conf. Artificial Intelligence, 1996.

Vezhnevets, A., e V. Vezhnevets. “Modest adaboost - teaching adaboost to generalize better.” *Graphicon, Vol 12*, 2005: 987–997.

Viola, P. e Jones, M. J. “Robust Real-Time Face Detection.” *International Journal of Computer Vision, Vol 57*, 2004: 137–154.

Wang, L., A. Chakraborty, e D. Comaniciu. System and method for automatic molecular diagnosis of ALS based on boosting classification. Estados Unidos Patente 20060218109. 28 de Setembro de 2006.

Webster, J. G. *The Measurement, Instrumentation, and Sensors.* CRC Press, 1998.

Weiss, S. M., e N. Indurkhya. *Predictive Data Mining: A Practical Guide.* Morgan Kaufmann, 1997.

Whitehill, J., Littlewort, G., Fasel, I., Bartlett, M., e J. Movellan. “Developing a Practical Smile Detector.” *Machine Perception Laboratory, Submitted to PAMI, vol 3*, 2007: 5.

Witten, I. H., e E. Frank. *Data mining: practical machine learning tools and techniques, 2 edição.* San Francisco: Elsevier, 2005.

Yurish, S. Y., N. V. Kirianaki, e I. L. Myshkin. “World Sensors and MEMS Markets: Analysis and Trends.” *Sensors & Transducers Magazine, Vol.62*, 2005: 456-461.

Zhang, Z., e X. Xie. “Research on AdaBoost.M1 with Random Forest.” *2nd International Conference on Computer Engineering and Technology (ICCET)*, 2010: V1-647 - V1-652.

Apêndice A–Exemplo de entrada para o treinamento do AdaBoost

A.1 Exemplo de arquivo de treinamento para o software WEKA:

A seguir é exemplificado um arquivo compatível com a entrada do programa WEKA utilizado para a realização da etapa de treinamento do algoritmo. O símbolo "?" indica os exemplos que apresentam a classe não definida e terão que ser classificados pelo algoritmo. Esses são os exemplos de teste. O restante foi utilizado para exemplos de treinamento.

.....

@relation Analise_de_concentracao_de_agua_em_Alcool

@attribute Sensor {sensor_1, sensor_2, sensor_3, sensor_4}
@attribute temperatura_da_solucao real
@attribute Periodo_(ms) real
@attribute Capacitancia_(pF) real
@attribute Concentração {Conforme, Não_conforme}

@data
sensor_1,22.14,205,411.64,?
sensor_2,21.16,98,200.21,Conforme
sensor_3,22.78,191,373.41,?
sensor_4,22.63,225,477.45,?
sensor_1,22.39,205,411.64,?
sensor_2,22.58,98,200.21,?
sensor_3,22.48,191,373.41,?
sensor_4,22.92,225,477.45,?
sensor_1,22.14,205,411.64,Não_conforme
sensor_2,23.12,98,200.21,?
sensor_3,23.02,191,373.41,?
sensor_4,22.73,225,477.45,?
sensor_1,22.63,205,411.64,?
sensor_2,22.78,98,200.21,?
sensor_3,23.17,191,373.41, Não_conforme

sensor_4,23.22,225,477.45,?

sensor_1,22.09,205,411.64,?

sensor_2,22.19,98,200.21,?

sensor_3,23.26,191,373.41,?

sensor_4,22.92,225,477.45,?

sensor_1,21.85,205,411.64,?

sensor_2,22.53,98,200.21,?

sensor_3,22.87,191,373.41,?

sensor_4,22.29,225,477.45,?

sensor_1,21.99,205,411.64,Conforme

sensor_2,22.14,98,200.21,?

sensor_3,22.92,191,373.41,?

sensor_4,22.92,225,477.45,?

sensor_1,21.85,205,411.64,?

sensor_2,21.85,98,200.21,?

sensor_3,22.92,191,373.41, Não_conforme

sensor_4,22.53,225,477.45,?

sensor_1,22.09,205,411.64,?

sensor_2,22.29,98,200.21,?

sensor_3,23.02,191,373.41,?

sensor_4,23.02,225,477.45, Não_conforme

sensor_1,21.65,205,411.64,?

sensor_2,22.29,98,200.21,?

sensor_3,22.97,191,373.41,Conforme

sensor_4,22.73,225,477.45,?

sensor_1,22.24,205,411.64,?

sensor_2,21.99,98,200.21,?

sensor_3,22.83,191,373.41,?

sensor_4,22.73,225,477.45,?

sensor_1,22.09,205,411.64,?

sensor_2,21.6,98,200.21,?

sensor_3,22.73,191,373.41, Não_conforme

sensor_4,22.53,225,477.45,?

sensor_1,21.75,205,411.64, Não_conforme