UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA DE TELEINFORMÁTICA
PROGRAMA DE PÓS-GRADUACÃO EM ENGENHARIA DE TELEINFORMÁTICA

David Nascimento Coelho

TÉCNICAS DE APRENDIZAGEM DE MÁQUINA PARA DETECÇÃO DE FALHAS EM MOTORES DE INDUÇÃO TRIFÁSICOS

Fortaleza

2015

David Nascimento Coelho

TÉCNICAS DE APRENDIZAGEM DE MÁQUINA PARA DETECÇÃO DE FALHAS EM MOTORES DE INDUÇÃO TRIFÁSICOS

Dissertação apresentada à coordenação do Programa de Pós-Graduação em Engenharia de Teleinformática, da Universidade Federal do Ceará, como requisito parcial à obtenção do título de Mestre Em Engenharia de Teleinformática.
Área de concentração: Sinais e Sistemas.
Prof. Dr. Guilherme de Alencar Barreto

Fortaleza

2015

David Nascimento Coelho
TÉCNICAS DE APRENDIZAGEM DE MÁQUINA PARA DETECÇÃO DE FALHAS EM MOTORES DE INDUÇÃO TRIFÁSICOS/ David Nascimento Coelho. – Fortaleza, 2015-
80 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Guilherme de Alencar Barreto

Dissertação (Mestrado) – Universidade Federal do Ceará
Centro de Tecnologia
Departamento de Engenharia de Teleinformática
Programa de Pós-Graduação em Engenharia de Teleinformática, 2015.

1. Machine Learning 2. Fault Detection 3. Induction Motor I. Guilherme Alencar Barreto. II. Universidade Federal do Ceará. III. Departamento de Engenharia de Teleinformática. IV. Título

CDU 02:141:005.7

David Nascimento Coelho

TÉCNICAS DE APRENDIZAGEM DE MÁQUINA PARA DETECÇÃO DE FALHAS EM MOTORES DE INDUÇÃO TRIFÁSICOS

Dissertação apresentada à coordenação do Programa de Pós-Graduação em Engenharia de Teleinformática, da Universidade Federal do Ceará, como requisito parcial à obtenção do título de Mestre Em Engenharia de Teleinformática.
Área de concentração: Sinais e Sistemas.

Aprovado em: Fortaleza, 20 de Setembro de 2015.

Prof. Dr. Guilherme de Alencar Barreto
Universidade Federal do Ceará

Prof. Dr. Cláudio Marques de Sá Medeiros
Instituto Federal de Educação Ciência e Tecnologia do Ceará

Prof. Dr.
Universidade bla

Prof. Dr.
Examinator 3

Agradecimentos

A Deus, por sempre se mostrar presente em minha vida.

A meus pais, que sempre me apoiaram em todos os momentos e são exemplos em tudo para mim.

Aos meus irmãos, que mesmo já não dividindo o mesmo teto são extremamente presentes e importantes.

A Nice, por ter enfrentado cada fase dessa dissertação, e por sempre me aconselhar e apoiar nas decisões mais importantes.

Ao Prof. Guilherme Alencar Barreto e ao Prof. Cláudio Marques de Sa Medeiros, por toda a paciência e ensinamentos durante o mestrado.

Aos professores participantes da banca examinadora (Xxxx) pelo tempo, pelas valiosas colaborações e sugestões.

Aos amigos do mestrado (Ananda, César, Daniel, Amauri, Davyd, Rodrigo, Rafael) pelas reflexões, críticas, sugestões recebidas e ajudas com o LaTex.

Aos amigos e família, os quais contribuíram para a finalização deste trabalho direta ou indiretamente.

À CAPES, pelo apoio financeiro com a manutenção da bolsa de auxílio.

Resumo

Esta dissertação visa a detecção de falhas incipientes por curto-circuito entre espiras do motor de indução trifásico gaiola de esquilo alimentado por um conversor de frequência senoidal. Para detectar este tipo de falha, diferentes condições de operação são aplicadas ao motor, e cada amostra do conjunto de dados real foi extraída das correntes de linha do conversor de frequência supracitado. Para extração de características, a análise da assinatura de corrente do motor foi utilizada. Para solucionar este problema, a detecção desta falha é tratada como um problema de classificação, por isso, diferentes algoritmos supervisionados de aprendizado de máquina são utilizados: Mínimos Quadrados Ordinários, Redes Perceptron Multicamadas, Máquina de Aprendizado Extremo, Máquina de Vetor de Suporte, Máquina de Vetor de Suporte por Mínimos Quadrados, Máquina de Aprendizado Minimo, e Classificadores Gaussianos. Juntamente com a técnica de opção de rejeição, estes classificadores são testados e os resultados destes são comparados entre si e com outros trabalhos que utilizam o mesmo banco de dados. Taxas de acerto máximo de até 100% com os classificadores SVM e LSSVM mostram que, em um futuro próximo, um sistema embarcado pode ser equipado com estes algoritmos.

Palavras-Chave: Aprendizado de Máquina. Motores de Indução. Detecção de Falhas.

Abstract

This dissertation aims at the detection of short-circuit incipient fault condition in a three-phase squirrel-cage induction motor fed by a sinusoidal PWM inverter. In order to detect this fault, different operation conditions are applied to an induction motor, and each sample of the real data set is taken from the line currents of the PWM inverter aforementioned. For feature extraction, the Motor Current Signature Analysis is used. The detection of this fault is treated as a classification problem, therefore different supervised algorithms of machine learning are used so as to solve it: Ordinary Least Squares, Multi-layer Perceptron, Extreme Learning Machine, Support-Vector Machine, Least-Squares Support-Vector Machine, the Minimal Learning Machine, and Gaussian Classifiers. Together with Reject Option technique, these classifiers are tested and the results are compared with other works with the same data set. Accuracy rates of almost 100% with SVM and LSSVM classifiers shows that, in near future, an embedded system can be equipped with these algorithms.

Key-words: Machine Learning. Induction Motors. Fault Detection.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

BP	Back Propagation
ELM	Extreme Learning Machine
FFT	Fast Fourier Transform
LSSVM	Least-squares Support Vector Machine
MAP	Maximum a Posteriori
MCSA	Motor Current Signature Analysis
MIT	Motor de Indução Trifásico
MLM	Minimal Learning Machine
MLP	Multi-layer Perceptron
SVD	Singular Value Decomposition
SVM	Support Vector Machine
OLS	Ordinary Least Squares
PS	Perceptron Simples
RBF	Radial Basis Function

Lista de símbolos

Vs	Velocidade do motor (em RPM)
f_s	Frequência de alimentação do motor
P	Pares de polos do motor
s	Escorregamento do motor
f_h	Componentes de frequência
$k1$	Ordem dos harmônicos temporais
$k2$	Ordem dos harmônicos espaciais
N	Número total de amostras
n	Índice da amostra atual
$\mathbf{x}_n$	Vetor de amostras
$\mathbf{X}$	Matriz de amostras
C_n	Classe da amostra
p	Número de atributos do problema
c	Número de classes do problema
$\mathbf{d}_n$	Rótulo da amostra
$\mathbf{D}$	Matriz de rótulos
$p()$	Probabilidade
μ_i	Vetor médio da i-ésima classe
$\mathbf{\Sigma}_i$	Matriz de Covariância da i-ésima classe

$\|.\|$	Determinante de uma matriz
$g_i()$	função discriminante da i-ésima classe
$ln()$	logaritmo natural
$Q_i()$	Distância de Mahalanobis da i-ésima classe
b	Bias
$\mathbf{y}_n$	Saída do classificador
$\mathbf{Y}$	Matriz de saídas do classificador
$N1$	Número de amostras para treinamento e validação
$N2$	Número de amostras para teste
q	Número de neurônios ocultos
$\|\|.\|\|^2$	Norma Euclidiana quadrática
T	Transposição de uma matriz
l	Constante da regularização de Tikhonov
L	Quantidade de parâmetros de um classificador
$\mathbf{W}$	Matriz de pesos
$\mathbf{w}_i$	Vetor de pesos do i-ésimo neurônio da camada oculta
$\mathbf{m}_k$	Vetor de pesos do k-ésimo neurônio da camada de saída
t	Iteração
u_i	Ativação do i-ésimo neurônio
$\varphi()$	Função de ativação
$\phi'()$	Derivada da função de ativação
δ_k	Gradiente local do k-ésimo neurônio
e_k	Erro do k-ésimo neurônio
λ	Taxa de aprendizagem
η	Fator de momento
$\mathbf{h}_n$	Saída do n-ésimo neurônio da camada oculta do ELM

$\mathbf{H}$	Matriz de saídas dos neurônios da camada oculta do ELM
$\boldsymbol{\beta}$	Matriz de pesos dos neurônios de saída da rede ELM
$j()$	Função custo
$L()$	Função lagrangeana
α_i	Multiplicador de Lagrange
ξ_i	Variável de folga
C	Constante de regularização do SVM
$\kappa(.,.)$	Função de Kernel
σ^2	Hiperparâmetro da função de Kernel gaussiano
γ	Constante de regularização do LSSVM
K	Pontos de referência do MLM
$\mathbf{m}_x$	Entradas de referência do MLM
$\mathbf{D}_x$	Distância entre as entradas
$\mathbf{t}_y$	Saída de referência do MLM
$\mathbf{\Delta}_y$	Distância entre as saídas
$\mathbf{B}$	Modelo de regressão do MLM
r	Número de divisões da validação cruzada

Sumário

Capítulo 1

Introdução

Neste capítulo, é feita uma breve introdução sobre o motor de indução trifásico, a detecção de falhas neste, e como este problema será tratado. Além disso, este capítulo também contém os objetivos, a organização, e contribuições desta dissertação.

1.1 Motores de Indução

Os motores de indução trifásicos (MIT), devido a sua robustez, eficiência e simplicidade, são a principal força motora da indústria atual (BACHA et al., 2008), (SESHADRINATH; SINGH; PANIGRAHI, 2014). De acordo com Thomson e Fenger (2001), estes podem consumir tipicamente de 40 a 50% de toda capacidade gerada de uma nação industrializada.

Dentre as aplicações destas máquina, podemos citar o acionamento de ventiladores, bombas, bobinadeiras, esteiras transportadoras e elevadores (JUNIOR, 2013).

De modo a adequar este motor às diversas aplicações, vários estudos sobre o controle de velocidade e torque desta máquina já foram realizados. Dentre os meios para controlar a velocidade desta, podemos citar controle vetorial e escalar (VENKADESAN; HIMAVATHI; MUTHURAMALINGAM, 2013), (SAWA; KUME, 2004), (NIRALI; SHAH, 2011).

Quando, em determinada aplicação, a variação de velocidade do motor de indução se mostra necessária, os inversores de frequência são muito utilizados. Estes dispositivos variam a velocidade do motor de indução modificando a frequência da tensão de alimentação deste.

A equação que relaciona a velocidade do motor à frequência da tensão de alimentação deste, pode ser descrita como:

$$Vs = \frac{120 f_s}{2P}(1 - s), \tag{1.1}$$

onde Vs é a rotação mecânica (em RPM), fs é a frequência de alimentação, em Hertz (Hz), do motor, P é o número de pares de polos, e s é o escorregamento deste.

Mesmo com todos os estudos sobre esta máquina e toda a sua robustez, devido ao envelhecimento da máquina, condições adversas do ambiente, aplicações inadequadas, ou falta de um programa de manutenção, o motor de indução é sujeito a várias falhas

(GHATE; DUDUL, 2010), (NANDI; TOLIYAT; LI, 2005). De acordo com Vico e Hunt (2010), a taxa de falhas nestes motores é conservadoramente estimada de 3 a 5% por ano.

Os tipos mais comuns destas são falhas em rolamento, falhas de isolamento no estator ou rotor, abertura de barras ou quebra de anéis, e excentricidade (NANDI; TOLIYAT; LI, 2005), (BONNETT, 2010). Dentre todas estas falhas, a perda de isolamento nos enrolamentos do estator corresponde a aproximadamente 40% de todos as falhas de motores (NANDI; TOLIYAT; LI, 2005), (MARTINS; PIRES; PIRES, 2007).

Devido a todas estas falhas, se não houver um monitoramento periódico destes motores, podem existir paradas não programadas em linhas de produção, e o custo destas é muito alto (OLIVEIRA; SA, 2013). Segundo Avelar, Baccarini e Amaral (2011), Custos de manutenção podem representar de 15% a 40% de muitos produtos.

Por isso, a detecção de falhas e o monitoramento periódico das condições de máquinas rotativas trazem um diferencial competitivo para as indústrias, visto que estes procedimentos podem garantir eficiência, segurança e bom funcionamento destas máquinas, conduzindo, assim, a uma maior produtividade e redução na perda de capital (YADAV, 2011).

No caso do curto circuito entre espiras no enrolamento do estator, este demora apenas alguns minutos para evoluir (THOMSON; FENGER, 2001). Assim, um monitoramento em tempo real desta falha é um ferramenta importante para reduzir custos e proteger a máquina.

1.2 O Problema da Detecção de Falhas em MIT

Muitos estudos sobre detecção de falhas por curto circuito entre espiras do estator do motor de indução trifásico foram feitos recentemente. Como exemplos, Asfani et al. (2012) utilizaram densidade espectral, transformada Wavelet Haar, e redes neurais para detecção de curto circuito. Por outro lado, Avelar, Baccarini e Amaral (2011) utilizou máquinas de vetores de suporte, e leituras da componente fundamental de tensão e das componentes fundamental e 3ª harmônica de corrente para este mesmo tipo de falha. Por fim, (SESHADRINATH; SINGH; PANIGRAHI, 2014) utilizou uma rede neural probabilística baseada em Wavelets para detectar falhas entre espiras.

Nestes e em outros trabalhos, durante a pesquisa realizada na presente dissertação, não foram encontradas comparações entre classificadores de diversos paradigmas de aprendizado de máquina. Ou seja, normalmente utiliza-se um método de extração de características e um ou dois classificadores para a detecção de falhas.

Além disso, na pesquisa feita, não foram encontrados muitos trabalhos com detecção de falhas em motores alimentados por inversores de frequência.

Por fim, quando é necessária a detecção de uma determinada falha, tenta-se dividir as amostras coletadas de motores entre funcionamento normal ou com falha. Assim a detecção de falhas pode ser tratada como um problema de reconhecimento de padrões, onde os dados devem ser divididos em classes.

De acordo com Príncipe (2000), O problema central no reconhecimento de padrões é definir a forma e a colocação de um limite, de modo que os erros de atribuição das classes são minimizados.

Vários algoritmos de aprendizado de máquina podem ser utilizados para resolver este tipo de problema, e estes algoritmos não necessitam de um modelo matemático do motor (MARTINS; PIRES; PIRES, 2007), porém é necessário uma quantidade consistente e significativa de dados que pode representar adequadamente o problema específico.

Por isso, há várias técnicas para extrair informações dos motores, de forma a gerar dados para detecção de falhas. Análise de sinais de vibração, análise da assinatura de corrente do motor, análise de óleo, fusão de sensores, análise de temperatura, análise de ruído audível, ultrassom, filtro de Kalman e filtro seguidor alfa-betha-gama são alguns exemplos destas técnicas (LEE, 2014).

Dentre estas, a análise da assinatura de corrente do motor (*Motor Current Signature Analysis* - MCSA) é muito utilizada. Particularmente, na detecção de curto circuito entre espiras do estator, é comum utilizar algumas componentes do espectro de frequências de corrente para compor a entrada para o sistema de detecção.

Porém, este problema não é simples, visto que as mesmas componentes de frequência utilizadas para detecção de falhas podem existir previamente no sistema elétrico ou serem afetadas por mais de uma falha, e o ruído proveniente do ambiente pode estar fortemente incorporado aos sinais de corrente (GHATE; DUDUL, 2010). Essas condições podem criar muitas dificuldades na detecção de falhas, e tornar o problema não linear.

Por isso, de modo a resolver este problema, diversos estados da arte de classificadores supervisionados com diferentes paradigmas de aprendizagem de máquina foram utilizados e seus desempenhos foram comparados.

O classificador dos mínimos quadrados é do tipo linear, Perceptron Multi camadas e a máquina de aprendizado extremo são redes neurais, a máquina de vetor de suporte e a máquina de vetor de suporte por mínimos quadrados tem sua teoria baseada na teoria do aprendizado estatístico, e os classificadores gaussianos partem do pressuposto que a distribuição de probabilidade intraclasse é gaussiana. Por fim, a máquina de aprendizado mínimo tem seu princípio baseado em distâncias dos padrões.

Como nesta dissertação, o uso de um conversor de frequências é pressuposto, um sistema de detecção de falhas (baseado em um algoritmo de aprendizado de máquina) previamente embarcado neste pode ser um vantagem competitiva para as empresas, porque,

ao embarcar este, não há a necessidade de alterações consideráveis no hardware deste dispositivo, e nem na aplicação a ser monitorada.

1.3 Objetivos

Dado o exposto no tópico anterior, os seguintes objetivos foram traçados para esta dissertação.

1.3.1 Objetivo Geral

O objetivo geral desta pesquisa é, através da comparação de diversas técnicas de aprendizagem de máquina, detectar falhas incipientes por curto circuito em motores de indução trifásicos gaiola de esquilo, alimentados por um conversor de frequências.

1.3.2 Objetivos Específicos

Os objetivos específicos desta dissertação estão listados a seguir:

1 Analisar estatisticamente o banco de dados reais, o qual foi adquirido através de sensores de efeito hall.

2 Implementar diferentes paradigmas de classificadores de padrões

3 Estudar a eficácia de diferentes classificadores de padrões na detecção de falhas de motores de indução trifásicos.

4 Utilizar estratégias de classificação com opção de rejeição, visando aumentar a confiabilidade do sistema de classificação

5 Comparar, através de testes estatísticos, classificadores de padrões quanto a sua capacidade de detecção de falhas.

1.4 Produção Científica

Os resultados parciais deste trabalho foram reunidos no artigo a seguir.

COELHO, D. N. ; SANTOS, J. D. A. ; MEDEIROS, C. M. S. ; BARRETO, G. A. Performance Comparison of Classifiers in The Detection of Short Circuit Incipient Fault in a Three-phase Induction Motor. In: IEEE Symposium Series on Computational Intelligence, 2014, Orlando, FL, USA. IEEE-SSCI, 2014.

Além deste artigo, durante o período no qual esta dissertação foi feita, o autor desta também foi co-autor nos seguintes artigos:

SILVA, R. D. C. ; COELHO, D. N. ; THE, G. A. P. . Comparison Between K-Nearest Neighbors, Self-organizing Maps and Optimum-path Forest in the recognition of Packages using Image Analysis by Zernike Moments. In: International Conference on Industry Applications, 2014, Juiz de Fora. INDUSCON, 2014.

SILVA, R. D. ; COELHO, D. N. ; THE, G. A. P. Performance Analysis of Classifiers to Recognition of Objects from Low-resolution Images Indutrial Sensor. In: Simpósio Brasileiro de Automação Inteligente, 2013, Fortaleza. SBAI/DINCON 2013, 2013.

OLIVEIRA, A. G. ; COELHO, D. N. ; BESSA, R. ; MEDEIROS, C. M. S. ; PONTES, R. T. . Técnicas Computacionais Para a Detecção de Falhas por Curto-circuito Entre Espiras de um Motor de Indução Acionado por Conversor de Frequência. In: Simpósio Brasileiro de Automação Inteligente, 2013, Fortaleza. SBAI/DINCON 2013, 2013.

1.5 Organização Geral da Dissertação

Por fim, o restante desta dissertação está organizada na seguinte formatação:

No capítulo 2, é mostrado um estudo sobre a falha por curto circuito entre espiras do estator do MIT, e sobre a detecção de falhas neste, além de descrever o conjunto de dados utilizado neste trabalho.

No capítulo 3, o problema de classificação é apresentado, e uma breve explicação dos algoritmos utilizados e das especificidades do problema tratado são feitas.

Já No capítulo 4, os detalhes da implementação dos algoritmos, a metodologia geral de comparação destes, e as técnicas computacionais (utilizadas juntamente com os classificadores) são discriminadas.

No capítulo 5 os resultados do trabalho desenvolvido são discutidos e analisados.

Por fim, no capítulo 6, são feitas as considerações finais e as perspectivas para trabalhos futuros

Capítulo 2

Detecção de Falhas em Motores de Indução

Neste capítulo, as características do problema investigado nesta dissertação são especificadas. Primeiramente a falha por curto circuito e os meios para a sua detecção são evidenciados. Por fim, é mostrado o procedimento para formação do banco de dados.

É importante mencionar que esta dissertação é a continuação de outros dois trabalhos. Em (COELHO, 2013), tentou-se detectar o início da falha de curto-circuito e supervisionar sua evolução através de Mapas Auto-organizáveis (Self-organizing Maps - SOM). Neste, há uma explicação mais detalhada da montagem da bancada de testes e da aquisição dos dados reais. Na seção 2.2 será mostrada uma visão geral desta montagem.

Já em (OLIVEIRA; SA, 2013), verificou-se a capacidade das redes neurais Perceptron Multicamadas (Multi-layer Perceptron - MLP) e Máquina de aprendizado extremo (Extreme Learning Machine - ELM) na detecção de falhas por curto-circuito. Neste, há uma explicação detalhada dos testes realizados para a seleção dos atributos.

2.1 Descrição do problema

Como já mencionado anteriormente, as falhas por curto circuito entre espiras do estator dos motores de indução trifásicos estão entre as de maior ocorrência dentre todas as falhas.

Os maiores contribuidores para este tipo de falha podem ser agrupados como estresses térmicos, elétricos, mecânicos e do ambiente (SESHADRINATH; SINGH; PANIGRAHI, 2014). Na presença de um ou mais destes estresses, o motor pode vir a falhar.

Especificamente, a falha por curto circuito no estator do motor de indução trifásico inicia como uma falha de alta impedância (NATARANJAN, 1989) e, em seguida, a corrente de falha pode causar um aquecimento local, fazendo com que a falha se espalhe rapidamente no bobinamento (TALLAM, 2003), configurando uma falha de baixa impedância.

No geral, existem duas formas de detectar falhas: através do diagnóstico ou de um prognóstico (LEE, 2014).

O diagnóstico é feito após ocorrer a falha, e serve tanto para consertar a máquina

como para determinar as causas da falha. É uma abordagem reativa ao problema, não previne as falhas.

Já o prognóstico é a detecção do início da falha, podendo estimar (predizer) a vida útil remanescente da máquina e programar a sua manutenção. Esta é uma abordagem que previne o problema (proativa).

No caso das falhas por curto circuito no estator do motor de indução trifásico acionado por um inversor de frequências, é muito importante fazer o prognóstico da falha, visto que, se esta falha for detectada quando se está iniciando sua ocorrência, a equipe de manutenção pode atuar e economizar custos de produção, proteger o inversor de frequências, e a máquina de indução pode ser reutilizada após o rebobinamento (THOMSON, 2001).

Para detectar este tipo de falha, primeiramente, é necessário algum método para extrair características do motor. Dentre os diversos métodos, a análise da assinatura de corrente do motor (MCSA) é um grande candidato, visto que este é não invasivo, não precisa ser adaptado para áreas com risco de explosão e pode ser aplicado, sem restrição de potência, a qualquer máquina (THORSEN; DALVA, 1997).

Este método consiste em utilizar algumas componentes do espectro de frequência da corrente do motor como características da falha. Para definir quais são estas frequências, (PENMAN, 1994) perceberam que as harmônicas geradas pelas falhas por curto circuito obedeciam a seguinte equação:

$$f_h = \left[\frac{k_1 \pm k_2(1-s)}{P}\right] f_s, \tag{2.1}$$

onde f_h são as componentes de frequência em função do curto-circuito entre espiras; $k_1 = 1, 3, 5...$ é a ordem das harmônicas temporais; $k_2 = 1, 2, 3...$ é a ordem das harmônicas espaciais; s é o escorregamento; P é o número de pares de polos; e f_s é a frequência fundamental de alimentação do motor.

Assim, no caso da utilização de um inversor de frequências, sabendo o valor da frequência fundamental com a qual este alimenta o motor, pode-se extrair as demais componentes para a detecção de falhas.

A partir da equação 2.1, alguns autores tentaram encontrar componentes de frequência para a detecção de falhas por curto circuito entre espiras. (THOMSON; FENGER, 2001) realizaram experimentos com motores de baixa potência aplicando curto circuito entre espiras nestes. As componentes encontradas apenas para curto circuito, ocorreram quando, na equação 2.1, $k_1 = 1$ e $k_2 = 3$, ou $k_1 = 1$ e $k_2 = 5$. Em um motor com dois pares de polos e com escorregamento próximo de 0, estas componentes equivalem aproximadamente a $2,5f_s$ e $3,5f_s$.

Já Coelho (2013), após aplicar a Transformada rápida de Fourier (*Fast Fourier*

Transform - FFT) a diversas amostras de sinais de correntes do motor de indução trifásico sujeitos a vários níveis de curto circuito, verificou que as componentes do espectro de frequências que mais variavam eram as seguintes: $0,5f_s$, $1f_s$ $1,5f_s$, $2f_s$, $3f_s$, $5f_s$ e $7f_s$

Por fim, (OLIVEIRA; SA, 2013), após coletar, do espectro de frequência de um motor sujeito a vários níveis de curto-circuito, 16 componentes entre $0,5f_s$ e $8f_s$ (com distância de $0,5f_s$ entre cada componente), aplicou uma rede perceptron multicamadas a diversas combinações destas componentes para tentar detectar este tipo de falha. Os melhores resultados de classificação foram obtidos através da seguinte combinação de componentes: $0,5f_s$, $1,5f_s$, $2,5f_s$, $3f_1$, $5f_s$ e $7f_s$

Nesta presente dissertação, os mesmo sinais usados para gerar as componentes em COELHO, 2013 e (OLIVEIRA; SA, 2013) foram utilizados. Estes e a bancada de testes que serviu para gerá-los são explicados no tópico a seguir.

2.2 Descrição do Banco de Dados

Nesta seção, é explicada a formação do banco de dados e o fluxo do problema a ser tratado. Nesta, é evidenciada a sequência de passos para obter a quantidade total dos dados, a distribuição destes entre as classes, as características de cada amostra, e as técnica para seleção e extração de atributos.

Primeiramente, sabe-se que o sucesso da aplicação de reconhecimento de padrões depende da construção de um modelo confiável para os dados.

Por isso, para gerar o conjunto de dados representantes de diversas condições de operação do MIT, uma bancada de testes foi montada.

Os componentes desta bancada são evidenciados na figura 1, onde podem ser vistos um motor de indução trifásico, um eixo conectando este a um disco de alumínio e duas bobinas perpendiculares a este disco. Cada uma das partes deste sistema será especificada a seguir.

Já na figura 2, tem-se uma visão geral dos componentes do sistema para detecção de falhas e da sequência de passos para obter cada amostra do banco de dados gerado.

O motor de indução trifásico gaiola de esquilo (I) foi utilizado como base para este banco de dados. Suas principais características são: 0.75 kW, 220/380 V, 3.02/1.75 A, eficiência de 79.5%, 1720 rpm, Ip/In = 7.2, e fator de potência 0.82.

Para aplicar três níveis de carga a este motor (0% - sem carga, 50%, e 100% da carga nominal), um Freio de Foucault foi utilizado (II). A bancada de testes utilizada para aplicação destes níveis de carga pode ser vista na figura 1.

Também, para variar a velocidade do motor, um inversor de frequências WEG

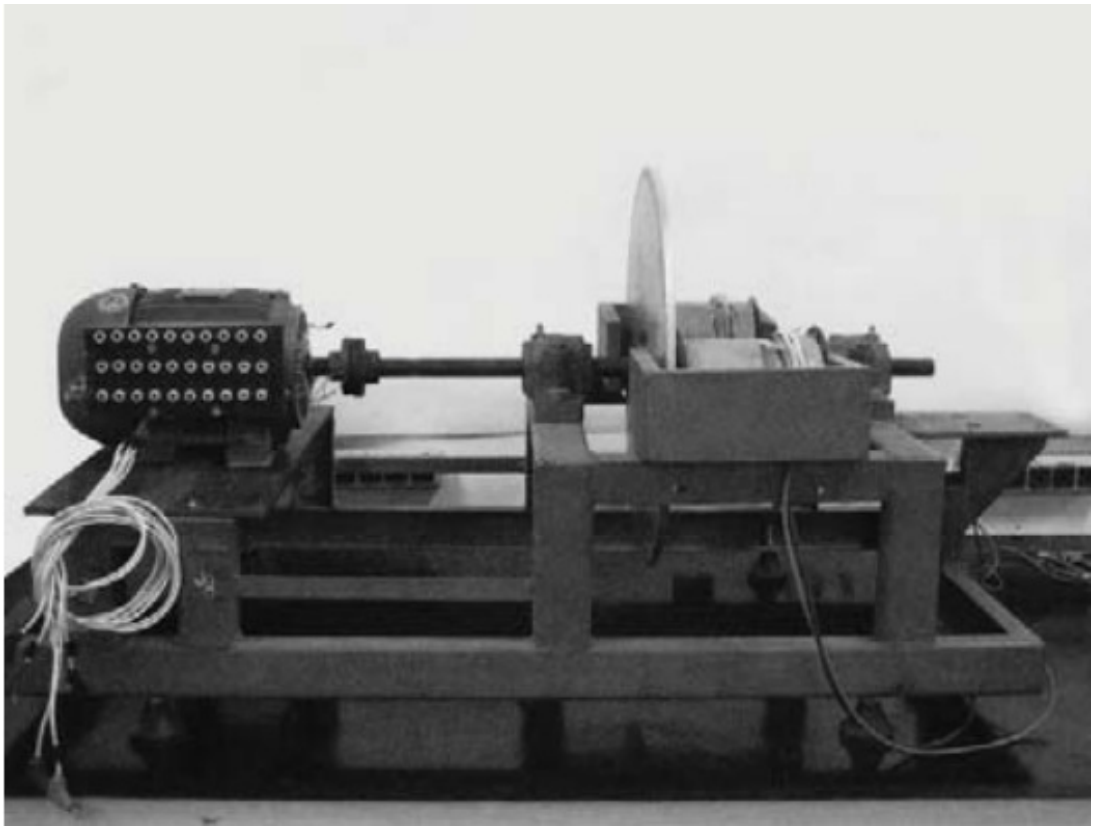

Figura 1 – Bancada de Testes.

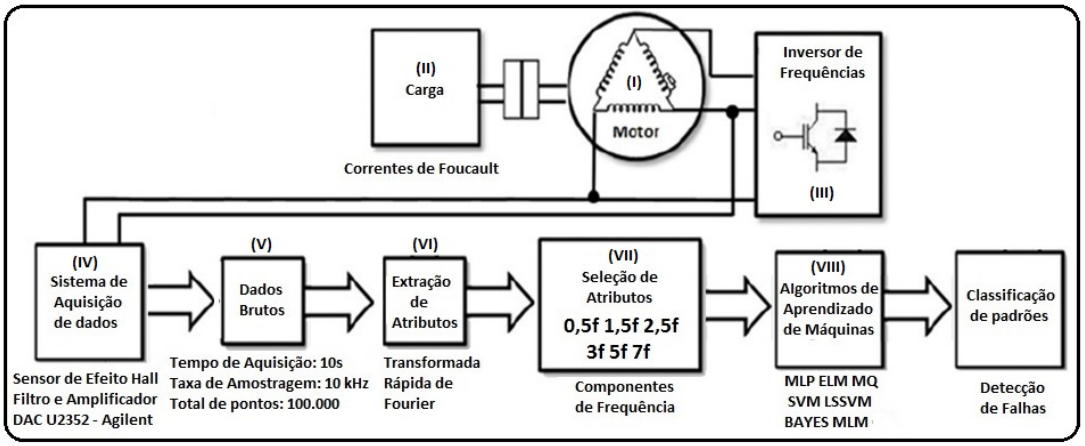

Figura 2 – Formação do Banco de Dados Reais. Adaptado. OLIVEIRA, 2014.

CFW-09 (III) foi utilizado com sete (7) frequências diferentes: 30 Hz, 35 Hz, 40 Hz, 45 Hz, 50 Hz, 55 Hz e 60 Hz. Este inversor pode ter controle vetorial, porém, neste trabalho, apenas a operação em malha aberta foi utilizada.

Além disso, três sensores baseados em efeito Hall (IV) foram utilizados para medir as correntes de linha de cada fase do inversor de frequências.

Normalmente, este motor possui apenas dois terminais disponíveis por fase (para que este seja alimentado). Porém, para gerar este conjunto de dados, o motor foi rebobinado,

de modo que mais terminais estão disponíveis, expondo derivações dos enrolamentos de cada fase.

Deste modo, pode-se emular diferentes níveis de curto circuito entre espiras do mesmo enrolamento. Por isso, neste trabalho, falhas por curto circuito entre fases não são consideradas.

Dentre os níveis de curto circuito, três foram utilizados. No menor nível (1), cinco espiras de um enrolamento são curto-circuitadas, totalizando 1,41% das espiras de uma fase. No nível seguinte (2), dezessete (17) espiras são curto circuitadas (correspondendo a 4,8% do enrolamento). Por fim, no nível três (3), 32 espiras (9,26% do enrolamento) são deixadas em curto circuito.

Finalmente, um sistema auxiliar de comando foi construído para executar dois tipos de curto circuito: alta impedância (A - emulando o início do processo de curto circuito), e baixa impedância (B - emulando a propagação da falha). Com estes dois tipos de curto circuito, e com os três níveis destes, existem seis condições diferentes de falha: A1, A2, A3, B1, B2 e B3.

Todas estas condições de operação estão representadas na figura 3. Os níveis de carga, a fase do inversor, a frequência da tensão aplicada ao motor, e a extensão da falha são respectivamente apresentados.

	Características das Amostras						
Nível de Carga	0%		50%		100%		
Fase do Inversor	Fase 1		Fase 2		Fase 3		
Frequência do Inversor	30 Hz	35 Hz	40 Hz	45 Hz	50 Hz	55 Hz	60 Hz
Extensão da Falha	Normal	A1	A2	A3	B1	B2	B3

Figura 3 – Características das Amostras. COELHO, 2013.

Todas estas condições totalizam 441 (3 x 3 x 7 x 7) amostras no domínio do tempo.

É importante mencionar que, como representado na figura 2, o motor foi conectado em delta. Nesta configuração, duas correntes de linha do inversor de frequências foram diretamente conectadas à fase do motor que continha as falhas. Como um dos objetivos deste projeto é fazer um sistema que possa detectar falhas utilizando apenas uma fase do inversor (independentemente de o sensor estar conectado diretamente ou não à fase com falha), apenas uma das duas fases citadas acima fora utilizadas para que não houvesse informação redundante.

Assim, no conjunto de dados final, são utilizadas 294 amostras (3 x 2 x 7 x 7): 147 da fase 1 (diretamente conectada à corrente de falha), e 147 da fase 3 (indiretamente conectada à corrente de falha). Todas estas amostras são representadas na figura 4.

Como pode ser visto nesta figura, o problema pode ser tratado a partir de 7 classes,

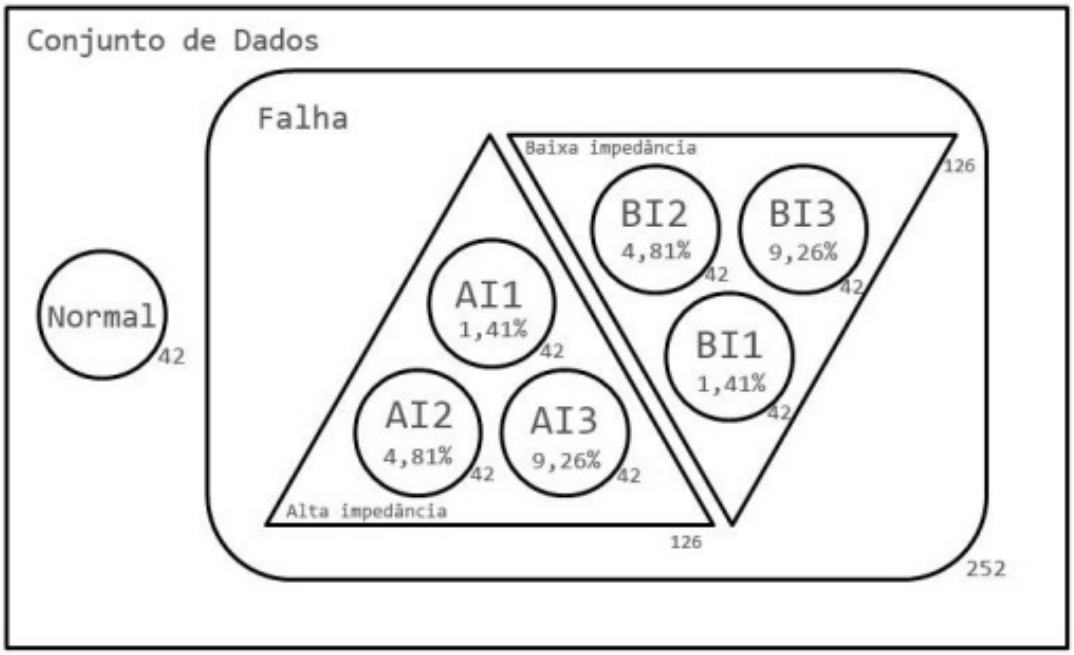

Figura 4 – Características das Amostras. OLIVEIRA, 2013.

se cada nível de falha for considerado uma classe (normal, AI1, AI2, AI3, BI1, BI2, BI3). Com esta configuração, cada classe possui 42 amostras.

Por outro lado, quando quer-se classificar os dados entre condição normal de operação, falha de alta impedância ou falha de baixa impedância, o problema mantém-se multi classes, porém existem apenas 3 classes: uma com 42 amostras (normal), outra com 126 amostras (AI1, AI2, AI3) e a última também com 126 amostras (BI1, BI2, BI3).

Porém, é importante mencionar que, para uma aplicação industrial, não é um problema considerável confundir os níveis ou tipos de falhas. Assim, o problema aqui estudado pode ser tratado como binário: motor em funcionamento normal ou com falha.

Neste último caso, existem 42 amostras de motor em funcionamento normal e 252 de motor em curto circuito (AI1, AI2, AI3, BI1, BI2, BI3).

Vale salientar que esta distribuição de dados não é comum em um meio industrial, visto que amostras de motores em funcionamento normal são mais frequentes que as amostras de falhas.

Todos estes dados foram adquiridos através de um sistema de aquisição Agilent U2352, de 16 bits de resolução, passando por um filtro analógico passa-baixa de 1 kHz e um amplificador de sinal.

Cada amostra do banco de dados é resultado de uma aquisição de 10 s à uma taxa de amostragem de 10 kHz. Assim, cada amostra possui 100.000 pontos (V).

É importante mencionar que os sinais adquiridos foram estacionários, e as correntes de falha foram limitadas a duas vezes a corrente de rotor bloqueado. Esta limitação foi

necessária para que o motor não fosse danificado após várias emulações de falhas.

Como apenas se capturou a situação estacionária dos sinais de corrente, após esta aquisição, foi aplicada a FFT (VI) para obter o espectro de frequência de cada uma das amostras. Como existe um filtro analógico de 1 kHz, pode-se escolher componentes de frequência de até 500 Hz como um atributo do problema.

Além disso, como a máxima frequência de alimentação do inversor é de 60 Hz, a maior componente múltipla desta frequência, considerando um máximo de 500 Hz, é a oitava (8a) harmônica (8 x 60 = 480).

Por fim, como em (OLIVEIRA; SA, 2013) foi feito um estudo minucioso com diversas combinações de componentes de frequência na detecção de falhas através de uma rede perceptron multicamadas, as mesmas componentes utilizadas pelo trabalho anteriormente citado foram usadas nesta dissertação. Estas componentes são: $0,5f_s$, $1,5f_s$, $2,5f_s$, $3f_s$, $5f_s$ e $7f_s$, onde f_s é a frequência fundamental do inversor (VII).

Assim, o conjunto final de dados, é composto por 294 amostras, cada uma possuindo 6 atributos, cada um correspondendo a uma harmônica da frequência fundamental de alimentação do inversor.

Após a compreensão do banco de dados gerado, serão apresentadas, nos capítulos 3 e 4, as ferramentas e metodologias utilizadas para detecção de falhas em motores.

Capítulo 3

Classificadores

Neste capítulo, o problema da classificação é abordado e, em seguida, os classificadores utilizados nessa dissertação são especificados e seus algoritmos são explicados.

Inicialmente, é muito importante definir alguns termos que serão utilizados durante esta dissertação:

- Atributo: uma característica do sistema a ser tratado, que serve de entrada para os classificadores.
- Amostra: se refere a um vetor de atributos, descrevendo um objeto a ser classificado.
- Hiperparâmetro: uma variável de um classificador, que deve ser definida, antes do treinamento, para que este classificador se adapte ao problema ao qual ele se propõe a solucionar.
- Parâmetro: uma variável do classificador que é definida durante o treinamento deste.
- Treinamento: etapa na qual os parâmetros de um classificador são atualizados
- Iteração de treinamento: por iteração, entende-se a utilização de uma amostra de treinamento para atualizar os parâmetros de um classificador.
- Época de treinamento: ocorre após todas as amostras de treinamento terem sido apresentadas (uma única vez cada) a um classificador, de modo a atualizar os parâmetros deste.
- Teste: etapa na qual o algoritmo é aplicado a dados não utilizados durante o treinamento e as estatísticas deste algoritmo são geradas.
- Realização: Quando finaliza-se as etapas de treinamento e validação, e testa-se o classificador com os dados restantes, ocorre uma realização do classificador.
- Generalização: é a característica mais desejada de um classificador, visto que, quanto maior a capacidade de generalização deste, maior será sua taxa de acerto para novos dados (dados de teste).

A partir dos termos anteriores, o problema da classificação será definido.

3.1 O problema da classificação

Formalmente, nos problemas de classificação, assume-se que se está de posse de um conjunto de N pares $\{\mathbf{x}_n, C_n\}_{n=1}^{N}$, em que o vetor coluna $\mathbf{x}_n \in \mathbb{R}^p$ representa a n-ésima amostra de entrada[1] e C_n é a classe à qual pertence $\mathbf{x}_n$. Assume-se ainda que se tem um número finito e pré-definido de c classes ($c \ll N$), i.e. $C_n \in \{C_1, C_2, \ldots, C_c\}$. Por fim, seja n_i o número de exemplos da i-ésima classe (i.e. C_i), temos que $N = n_1 + n_2 + \cdots + n_c = \sum_{i=1}^{c} n_i$.

Dependendo do problema, há várias formas de codificar os rótulos das classes. Nesta dissertação, escolheu-se um vetor coluna $\mathbf{d}_n \in \mathbb{R}^c$, no qual a posição referente a classe a qual o dado de entrada pertence tem o valor +1 e as demais posições tem o valor -1.

Como exemplo, se um problema tem 3 classes, e a amostra atual pertence à classe 2, seu rótulo será: $\mathbf{d}_n$ = [-1 +1 -1]^T. Assim, o problema aqui estudado possui N pares $\{\mathbf{x}_n, \mathbf{d}_n\}_{n=1}^{N}$.

Como já mencionado anteriormente, Segundo Príncipe, J. 2000, O problema central no reconhecimento de padrões é definir a forma e localização de um limite entre as amostras de diferentes classes, de modo a minimizar os erros de atribuição destas.

O modo como este limite é definido, depende do algoritmo de aprendizado de máquina utilizado. Dentre estes algoritmos, existem os supervisionados e os não-supervisionados.

No aprendizado não-supervisionado, não há a necessidade de um "professor" externo, durante o treinamento, que forneça a resposta desejada de saída do problema. Ou seja, os rótulos das amostras não são utilizados no algoritmo de treinamento dos classificadores baseados neste aprendizado.

Já no aprendizado supervisionado, já se sabe, durante o treinamento, as classes das amostras que estão sendo utilizadas. Ou seja, a saída desejada $\mathbf{d}_n$ do problema já é conhecida. Nesta dissertação, são utilizados apenas algoritmos baseados neste tipo de aprendizado.

Além disso, o problema de detecção de falhas por curto circuito pode ser tratado como binário ou multi classes. Se os dados forem divididos entre motores em funcionamento normal ou com falha, o problema em questão se torna binário. Porém, se determinados níveis de curto circuito forem tratados como uma classe, o problema se torna multi classes. Esta divisão dos dados entre as classes será melhor explicada no capítulo 4.

No restante do presente capítulo, tanto os algoritmos utilizados para o desenvolvimento deste trabalho, como alguns conceitos importantes são explicados.

[1] p é o número de atributos do problema em questão

3.2 Classificadores Estatísticos e Critério MAP

Nesta seção introduz-se o critério de decisão ótima, conhecido como critério *Maximum a Posteriori* (MAP), além de quatro classificadores gaussianos obtidos a partir da suposição de que os exemplos de uma dada classe seguem uma lei de distribuição de probabilidade normal.

3.2.1 Maximum a Posteriori

Inicialmente, define-se $p(C_i)$ como a probabilidade *a priori* da i-ésima classe, ou seja, a probabilidade de a classe C_i ser selecionada antes do experimento ser realizado, sendo o experimento o ato de observar e classificar um certo padrão. Este é um experimento aleatório, visto que não sabemos de antemão a que classe o padrão será atribuído. Logo, uma modelagem probabilística se torna justificável.

Em seguida, levando em conta apenas os dados da classe C_i, ou seja, ao subconjunto de padrões $\mathbf{x}_n$ cujos rótulos são iguais a C_i, um modelo probabilístico comum para estes dados é a densidade normal multivariada, denotada por $p(\mathbf{x}_n|C_i)$, de vetor-médio μ_i e matriz de covariância $\mathbf{\Sigma}_i$. Matematicamente, este modelo é dado pela seguinte expressão:

$$p(\mathbf{x}_n|C_i) = \frac{1}{(2\pi)^{\frac{p}{2}}|\mathbf{\Sigma}_i|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}(\mathbf{x}_n - \mu_i)^T\mathbf{\Sigma}_i^{-1}(\mathbf{x}_n - \mu_i)\right\}, \tag{3.1}$$

em que $|\mathbf{\Sigma}_i|$ denota o determinante da matriz de covariância $\mathbf{\Sigma}_i$, e $\mathbf{\Sigma}_i^{-1}$ denota a inversa desta matriz. Alguns detalhes sobre a implementação da matriz inversa estão dispostos na seção 3.3.

A densidade $p(\mathbf{x}_n|C_i)$, no contexto de classificação de padrões, também é chamada de *função de verossimilhança*[2] da classe C_i. A função de verossimilhança da classe C_i pode ser entendida como o modelo probabilístico que "explica" como os dados estão distribuídos nesta classe.

Além disso, dado que um novo padrão $\mathbf{x}_n$ é observado, qual seria a probabilidade de que este padrão pertença à classe C_i? Esta informação pode ser modelada através da função densidade *a posteriori* da classe, $p(C_i|\mathbf{x}_n)$.

Através do Teorema da Probabilidade de Bayes, a densidade a posteriori $p(C_i|\mathbf{x}_n)$ pode ser relacionada com a densidade a priori $p(C_i)$ e a função de verossimilhança $p(\mathbf{x}_n|C_i)$ por meio da seguinte expressão:

$$p(C_i|\mathbf{x}_n) = \frac{p(C_i)p(\mathbf{x}_n|C_i)}{p(\mathbf{x}_n)}. \tag{3.2}$$

Um critério comumente usado para tomada de decisão em classificação de padrões é o critério *máximo a posteriori* (MAP). Ou seja, um determinado padrão $\mathbf{x}_n$ é atribuído

[2] Do inglês, *likelihood function*.

à classe C_j se a moda da densidade a posteriori $p(C_j|\mathbf{x}_n)$ for a maior dentre todas. Em outras palavras, tem-se a seguinte regra de decisão:

$$\text{Atribuir } \mathbf{x}_n \text{ à classe } C_j, \text{ se } p(C_j|\mathbf{x}_n) > p(C_i|\mathbf{x}_n),\ \forall i \neq j.$$

Este critério MAP também pode ser definido como

$$C_j = \arg \max_{i=1,\ldots,c} \{p(C_i|\mathbf{x}_n)\}, \tag{3.3}$$

em que o operador "arg max" retorna o "argumento do máximo", ou seja, o conjunto de pontos para os quais a função de interesse atinge seu valor máximo.

Ao substituir a Eq. (3.2) na regra de decisão do critério MAP, obtém-se uma nova regra de decisão, dada por

$$\text{Atribuir } \mathbf{x}_n \text{ à classe } C_j, \text{ se } p(C_j)p(\mathbf{x}_n|C_j) > p(C_i)p(\mathbf{x}_n|C_i),\ \forall i \neq j,$$

em que o termo $p(\mathbf{x}_n)$ é eliminado por estar presente em ambos os lados da inequação. Em outras palavras, o termo $p(\mathbf{x}_n)$ não influencia na tomada de decisão feita por meio do critério MAP.

Na verdade, o critério MAP pode ser generalizado para usar qualquer *função discriminante* $g_i(\mathbf{x}_n)$, passando a ser escrito como

$$\text{Atribuir } \mathbf{x}_n \text{ à classe } C_j, \text{ se } g_j(\mathbf{x}_n) > g_i(\mathbf{x}_n),\ \forall i \neq j.$$

É importante ressaltar que, em um sentido amplo, uma função discriminante $g_i(\mathbf{x}_n)$ é qualquer função matemática que fornece um valor numérico que permita quantificar a pertinência do padrão $\mathbf{x}_n$ à classe C_i. Assim, as classes podem ser ranqueadas (ordenadas) em função dos valores de suas respectivas funções discriminantes.

A partir da definição de função discriminante, as deduções desta, para cada classificador, serão iniciadas.

3.2.2 Classificadores Gaussianos

No contexto dos classificadores bayesianos gaussianos, uma das funções discriminantes mais utilizadas é dada por

$$\begin{aligned} g_i(\mathbf{x}_n) &= \ln p(C_i|\mathbf{x}_n), \\ &= \ln p(C_i)p(\mathbf{x}_n|C_i), \\ &= \ln p(C_i) + \ln p(\mathbf{x}_n|C_i), \qquad (3.4) \\ &= g_i^{(1)}(\mathbf{x}_n) + g_i^{(2)}(\mathbf{x}_n), \qquad (3.5) \end{aligned}$$

em que $\ln(u)$ é a função logaritmo natural de u e a função $g_i^{(2)}(\mathbf{x}_n) = \ln p(\mathbf{x}_n|C_i)$ é chamada de função log-verossilhança da classe C_i.

Substituindo a função de verossimilhança mostrada na Eq. (3.1) em $g_i^{(2)}(\mathbf{x}_n)$, chega-se à seguinte expressão:

$$g_i^{(2)}(\mathbf{x}_n) = \ln\left[\frac{1}{(2\pi)^{\frac{p}{2}}|\mathbf{\Sigma}_i|^{\frac{1}{2}}}\exp\left\{-\frac{1}{2}Q_i(\mathbf{x}_n)\right\}\right], \tag{3.6}$$

$$= -\frac{1}{2}Q_i(\mathbf{x}_n) - \frac{p}{2}\ln 2\pi - \frac{1}{2}\ln|\mathbf{\Sigma}_i|, \tag{3.7}$$

em que $Q_i(\mathbf{x}_n) = (\mathbf{x}_n - \mu_i)^T\mathbf{\Sigma}_i^{-1}(\mathbf{x}_n - \mu_i)$.

Dado que o termo $-\frac{p}{2}\ln 2\pi$ é constante e aparece nas funções discriminantes de todas as classes ($i = 1, \ldots, c$). Logo, este termo não influencia na tomada de decisão, podendo ser eliminado.

A partir disto, podem-se definir algumas configurações de discriminantes para o classificador bayesiano gaussiano:

- **C**aso 1: A função discriminante geral para o classificador gaussiano é dada por

$$g_i(\mathbf{x}_n) = -\frac{1}{2}Q_i(\mathbf{x}_n) - \frac{1}{2}\ln|\mathbf{\Sigma}_i| + \ln p(C_i). \tag{3.8}$$

- **C**aso 2: A partir do discriminante anterior, uma suposição comumente feita na prática é a de que as densidades a priori das classes são iguais, ou seja

$$p(C_1) = P(C_2) = \cdots = P(C_c), \tag{3.9}$$

  o que equivale a supor que as classes são equiprováveis[3]. Com isto, é possível simplificar ainda mais a função discriminante mostrada na Eq. (3.8):

$$g_i(\mathbf{x}_n) = -\frac{1}{2}Q_i(\mathbf{x}_n) - \frac{1}{2}\ln|\mathbf{\Sigma}_i|, \tag{3.10}$$

  uma vez que o termo $\ln p(C_i)$ é igual para todas as c funções discriminantes. Vale ressaltar que usar esta função discriminante equivale a reescrever o critério MAP como

  Atribuir $\mathbf{x}_n$ à classe C_j, se $\ln p(\mathbf{x}_n|C_j) > \ln p(\mathbf{x}_n|C_i)$, $\forall i \neq j$,

  de tal forma que a regra de decisão passa a depender somente das funções de log-verossimilhança das classes. Neste caso, o critério MAP passa a ser chamado de critério da máxima verossimilhança (*maximum likelihood criterion*, ML).

  Além desta aproximação, existem algumas considerações que levam a outras configurações de classificadores gaussianos. São elas:

[3] Esta suposição pode ser encontrada na prática em situações nas quais o número de exemplos (padrões) por classe é aproximadamente igual.

- **C**aso 3: As estruturas de covariâncias das I classes são iguais, ou seja, suas matrizes de covariância são iguais. Em outras palavras,

$$\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \cdots = \boldsymbol{\Sigma}_I = \boldsymbol{\Sigma}. \tag{3.11}$$

Neste caso, a função discriminante da classe C_i passa a ser escrita simplesmente como

$$g_i(\mathbf{x}_n) = -\frac{1}{2}Q_i(\mathbf{x}_n) = -\frac{1}{2}(\mathbf{x}_n - \mu_i)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_n - \mu_i), \tag{3.12}$$

em que o termo $-\frac{1}{2}\ln|\boldsymbol{\Sigma}_i|$ foi eliminado por não influenciar mais na tomada de decisão. É possível notar que a função discriminante $g_i(\mathbf{x}_n)$ é proporcional a $Q_i(\mathbf{x}_n)$, que é a distância de Mahalanobis quadrática. Assim, pode-se fazer $g_i(\mathbf{x}_n) = Q_i(\mathbf{x}_n)$, de tal forma que o critério de decisão passa a ser escrito como

$$\text{Atribuir } \mathbf{x}_n \text{ à classe } C_j, \text{ se } Q_j(\mathbf{x}_n) < Q_i(\mathbf{x}_n),\ \forall i \neq j,$$

o que, em palavras, significa classificar $\mathbf{x}_n$ como sendo da classe C_j se a distância (de Mahalanobis) de $\mathbf{x}_n$ ao centróide da classe C_j (i.e. μ_j) for *menor* que as distâncias de $\mathbf{x}_n$ aos centróides restantes.

Matriz de Covariância Agregada - $\boldsymbol{\Sigma}_{pool}$: Uma forma muito comum de se implementar o classificador gaussiano cuja função descriminante é mostrada na Eq. (3.12) envolve o uso da matriz de covariância agregada, definida como

$$\begin{aligned}
\boldsymbol{\Sigma}_{pool} &= \left(\frac{n_1}{N}\right)\boldsymbol{\Sigma}_1 + \left(\frac{n_2}{N}\right)\boldsymbol{\Sigma}_2 + \cdots + \left(\frac{n_I}{N}\right)\boldsymbol{\Sigma}_I, \\
&= p(C_1)\boldsymbol{\Sigma}_1 + p(C_2)\boldsymbol{\Sigma}_2 + \cdots + p(C_K)\boldsymbol{\Sigma}_I, \\
&= \sum_{i=1}^{I} p(C_i)\boldsymbol{\Sigma}_i,
\end{aligned} \tag{3.13}$$

em que $p(C_i)$ é a probabilidade a priori da classe i. Percebe-se assim que a matriz $\boldsymbol{\Sigma}_{pool}$ é a média ponderada das matrizes de covariância das c classes, com os coeficientes de ponderação sendo dados pelas respectivas probabilidades a priori.

A matriz $\boldsymbol{\Sigma}_{pool}$ costuma ser mais bem condicionada que as matrizes de covariância individuais e, por isso, sua inversa tende a causar menos problemas de instabilidade numérica.

- **C**aso 4: Os atributos de $\mathbf{x}_n$ são descorrelacionados entre si e possuem mesma variância (que pode ser feita igual a 1). Neste caso, tem-se que a matriz de covariância de todas as classes é dada por

$$\boldsymbol{\Sigma} = \mathbf{I}_p, \tag{3.14}$$

em que $\mathbf{I}_p$ é a matriz identidade de ordem p. Logo, tem-se que $\mathbf{\Sigma}^{-1} = \mathbf{I}_p$. Neste caso, a função discriminante da classe C_i passa a ser escrita como

$$g_i(\mathbf{x}_n) = -\frac{1}{2}(\mathbf{x}_n - \mu_i)^T \mathbf{I}_p (\mathbf{x}_n - \mu_i), \tag{3.15}$$

$$= -\frac{1}{2}(\mathbf{x}_n - \mu_i)^T (\mathbf{x}_n - \mu_i), \tag{3.16}$$

$$= -\frac{1}{2}\|\mathbf{x}_n - \mu_i\|^2, \tag{3.17}$$

em que $\|\mathbf{u}\|^2$ denota a norma euclidiana quadrática de $\mathbf{u}$. Assim, pode-se fazer $g_i(\mathbf{x}_n) = \|\mathbf{x}_n - \mu_i\|^2$, de tal forma que o critério de decisão passa a ser escrito como

$$\text{Atribuir } \mathbf{x}_n \text{ à classe } C_j, \text{ se } \|\mathbf{x}_n - \mu_j\|^2 < \|\mathbf{x}_n - \mu_i\|^2, \forall i \neq j,$$

o que significa dizer que $\mathbf{x}_n$ deve ser classificado como pertencente à classe C_j se a distância euclidiana de $\mathbf{x}_n$ ao centróide μ_j for *menor* que as distâncias de $\mathbf{x}_n$ aos centróides restantes.

Este classificador também é chamado de classificador de máxima verossimilhança, pois o rótulo atribuído ao padrão de entrada é definido pela máxima proximidade deste ao centróide de determinada classe.

Nesta dissertação, os classificadores gaussianos cujas funções discriminantes estão descritas nos casos 2 (função não linear) e 4 (equação 3.10 e 3.17 respectivamente) foram utilizados.

No caso 4, chegou-se a um classificador linear a partir da suposição inicial de que os dados são distribuídos de forma Gaussiana. Na próxima seção, será mostrada a teoria sobre classificadores lineares, bem como um classificador linear específico: o baseado em mínimos quadrados ordinários

3.3 Classificadores Lineares

Nos classificadores lineares, busca-se obter um mapeamento linear entre os dados de entrada (no caso, vetores de atributos $\mathbf{x}_n \in \mathbb{R}^{p+1}$), e os dados de saída correspondentes (no caso, vetores $\mathbf{d}_n \in \mathbb{R}^c$ que indicam a qual classe pertence determinada amostra).

É importante mencionar que a dimensão do vetor $\mathbf{x}_n$ é $p+1$ pela adição de um viés b (também conhecido como *bias* ou *threshold*) aos atributos do problema.

Dado que tantos os vetores de atributos como os rótulos das classes são vetores coluna, este mapeamento pode ser realizado matematicamente pela equação

$$\mathbf{d}_n = \mathbf{W}\mathbf{x}_n, \tag{3.18}$$

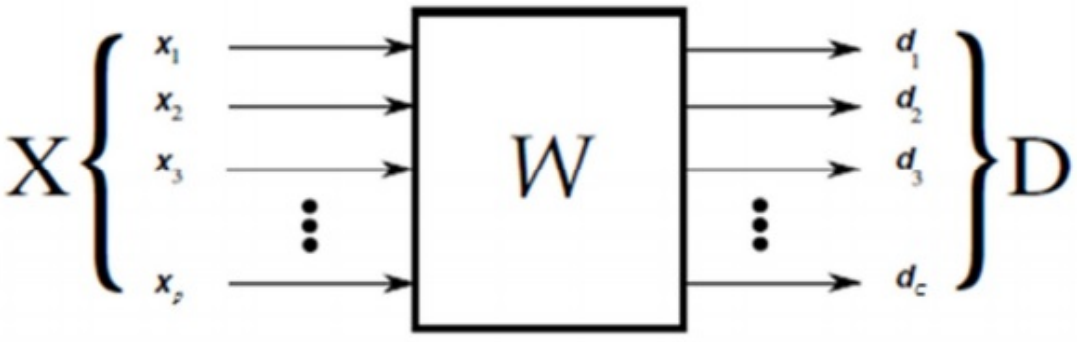

Figura 5 – Mapeamento linear dos dados. MONTEIRO, 2012.

onde $\mathbf{W} \in \mathbb{R}^{c\times(p+1)}$ é a matriz responsável por este. A representação gráfica deste classificador está contida na figura 5.

Assim, a função discriminante destes classificadores consiste na multiplicação do vetor de entrada atual pela matriz $\mathbf{W}$, e na atribuição deste vetor a uma classe C_j, onde C_j corresponde a posição do vetor de saída no qual é encontrado o máximo valor desta multiplicação. Isto pode ser descrito como

$$\text{Atribuir } \mathbf{x}_n \text{ à classe } C_j, \text{ se } d_{nj} > d_{ni}, \forall i \neq j.$$

A Priori, não se sabe o valor dos elementos da matriz $\mathbf{W}$, mas existem várias formas, durante o treinamento dos classificadores, de estimar os pesos desta matriz de forma a minimizar os erros de atribuições das entradas às classes.

Uma destas formas é através do classificador baseado nos mínimos quadrados ordinários, cujo algoritmo será especificado a seguir.

3.3.1 Mínimos Quadrados Ordinários

Inicialmente, assume-se que se está de posse N pares $\{\mathbf{x}_n, d_n\}_{n=1}^{N}$, onde $\mathbf{x}_n$ e $\mathbf{d}_n$ são, respectivamente, os vetores colunas de atributos e os de rótulos de determinada classe.

Agrupando-se estes vetores em matrizes, temos que $\mathbf{X} = [\mathbf{x_1}|\mathbf{x_2}|...|\mathbf{x_n}]$ é a matriz das amostras de entrada, e $\mathbf{D} = [\mathbf{d_1}|\mathbf{d_2}|...|\mathbf{d_n}]$ é a matriz dos rótulos de saída. Assim, a partir destas matrizes, a equação 3.18 pode ser representada matricialmente por

$$\mathbf{D} = \mathbf{WX}, \tag{3.19}$$

na qual todos os vetores $\mathbf{d_n}$ são calculados simultaneamente.

Se a matriz $\mathbf{X}$ for quadrada, ou seja, se o número n de amostras for igual à p + 1, a matriz $\mathbf{W}$ pode ser obtida através da seguinte equação:

$$\mathbf{W} = \mathbf{DX^{-1}}, \tag{3.20}$$

Porém, normalmente, nos problemas de classificação, tem-se uma quantidade bem superior de amostras comparada ao número de classes e atributos. Assim, $\mathbf{X}$ não pode ser invertida diretamente.

Por isso, para obter $\mathbf{W}$, uma técnica bastante difundida é a matriz pseudo-inversa (Moore-Penrose Pseudoinverse), também conhecida como técnica dos mínimos quadrados ordinários (Ordinary Least Squares - OLS). Através desta, visa-se minimizar a soma quadrática dos erros na aproximação representada na equação 3.19.

Para isto, isola-se $\mathbf{W}$ através da seguinte equação:

$$\mathbf{W} = \mathbf{D}\mathbf{X}^{\mathbf{T}}(\mathbf{X}\mathbf{X}^{\mathbf{T}})^{-\mathbf{1}}, \tag{3.21}$$

onde a matriz $\mathbf{X}^{\mathbf{T}}(\mathbf{X}\mathbf{X}^{\mathbf{T}})^{-\mathbf{1}}$ é chamada de matriz pseudoinversa de $\mathbf{X}$. A implementação desta será detalhada na seção a seguir.

Finalmente, após o cálculo da matriz $\mathbf{W}$, pode-se testar o desempenho do classificador OLS para as amostras que não foram utilizadas para treinamento, comparando a saída estimada pelo classificador, dada por

$$\mathbf{Y} = \mathbf{W}\mathbf{X}, \tag{3.22}$$

com os rótulos destas amostras, para validar se esta matriz representa (modela) bem o mapeamento entrada-saída dos dados.

3.3.2 Implementação da Inversão de Matrizes

Todos os algoritmos aplicados nesta dissertação foram implementados no software MATLAB. Assim, os meios para inverter matrizes partem do pressuposto que se está utilizando este software.

Inicialmente, considera-se o mesmo problema da equação 3.18, onde existe uma transformação linear de $\mathbf{X} \in \mathbb{R}^{(p+1)\times N}$ em $\mathbf{D} \in \mathbb{R}^{C\times N}$, através de $\mathbf{W} \in \mathbb{R}^{C\times(p+1)}$.

Neste caso, se quer descobrir o valor da matriz $\mathbf{W}$, através de $\mathbf{X}$ e $\mathbf{D}$, sabendo-se que $\mathbf{D} = \mathbf{W}\mathbf{X}$.

Dependendo dos valores de $p+1$ e c, pode-se formular este problema de dois modos:

- $\mathbf{W} = \mathbf{D}\mathbf{X}^{-1}$ caso $c = p + 1$ e
- $\mathbf{W} = \mathbf{D}\mathbf{X}^{\mathbf{T}}(\mathbf{X}\mathbf{X}^{\mathbf{T}})^{-\mathbf{1}}$ caso $c \neq p + 1$

onde a matriz $\mathbf{X}^{\mathbf{T}}(\mathbf{X}\mathbf{X}^{\mathbf{T}})^{-\mathbf{1}}$ é chamada de matriz pseudoinversa de $\mathbf{X}$.

Tanto no primeiro como no segundo modo, as matrizes a serem invertidas podem estar mal condicionadas, ou até serem singulares, e isto pode levar a resultados numericamente instáveis, o que não é desejável.

Para evitar este problema, pode-se utilizar o método de regressão de cumeeira (*ridge regression*), definido para lidar com problemas mal-condicionados. O uso desse método, também conhecido como regularização de Tikhonov, leva à seguinte estimativa das matrizes anteriores:

- $\mathbf{W} = \mathbf{D}(\mathbf{X} + l\mathbf{I})^{-1}$ caso $c = p + 1$ e
- $\mathbf{W} = \mathbf{D}\mathbf{X}^{\mathbf{T}}(\mathbf{X}\mathbf{X}^{\mathbf{T}} + l\mathbf{I})^{-1}$ caso $c \neq p + 1$

em que $l > 0$ é a constante de regularização e $\mathbf{I}$ é a matriz identidade de dimensões $p + 1 \times p + 1$.

A partir disso, existem as seguintes possibilidades para a inversão de matrizes:

- inv(): único método que pode ser utilizado apenas para o primeiro caso, onde a matriz é quadrada. Por não tratar automaticamente problemas de condicionamento, recomenda-se a utilização da seguinte expressão: $inv(X + l * I)$
- Operadores Backlash ("/"ou "\"): este método pode ser aplicado para solução da pseudo inversa, e utiliza decomposição QR como algoritmo. Dada a formulação acima, a pseudo inversa pode ser implementada como: $(X')/(X * X')$ ou $(X')/(X * X' + l * I)$ (na versão regularizada).
- pinv(): Assim como os operadores Backlash, este método pode ser utilizado para solucionar a pseudo inversa, porém este utiliza decomposição em valores singulares (Singular Value Decomposition - SVD), e já trata, automaticamente, singularidades. Assim, a pseudo inversa pode ser implementada simplesmete por: $pinv(X)$.

No caso desta dissertação, o método pinv() foi utilizado toda vez que necessitou-se calcular uma matriz inversa ou pseudoinversa.

Após entender os classificadores lineares e suas implementações, nas sessões a seguir, diferentes classificadores não-lineares serão evidenciados.

3.4 Classificadores Neurais

As redes neurais (HAYKIN, 2000) possuem, como base para sua estrutura, o modelo matemático de um neurônio biológico (MCCULLOCH; PITTS, 1943). A partir deste, foram desenvolvidos vários algoritmos que são utilizados para a resolução de diversos problemas, incluindo a classificação de padrões.

Neta dissertação, dois classificadores foram utilizados, baseados nas seguintes redes neurais: Perceptron Multi Camadas (multi-layer perceptron - MLP) e a Máquina de Aprendizado Extremo (extreme learning machine - ELM).

É importante salientar que, durante este trabalho, ao se mencionar rede MLP, o autor se refere às redes MLP, com apenas uma camada escondida, treinadas pelo algoritmo de retropropagação do erro (Error Back Propagation - BP).

As redes MLP são conhecidas por serem aproximadores universais de funções (HORNIK; STINCHCOMBE; WHITE, 1989), já as redes ELM possuem arquitetura semelhante as redes MLP, porém o seu treinamento é mais rápido. O algoritmo de ambas é descrito a seguir.

3.4.1 Perceptron Multi Camadas

As redes MLP tiveram como base as redes Perceptron Simples (PS) (ROSENBLATT, 1958). Nestas, assim como no classificador OLS, busca-se um mapeamento linear entre os dados de entrada e saída.

Porém, as redes PS se diferenciam do classificador OLS pela sua regra de aprendizagem e pelo fato de que cada linha da matriz **W** representa um neurônio artificial. Na figura 6, a arquitetura de um neurônio da rede neural perceptron simples é representada. Nesta, cada x_i representa um atributo (uma variável de entrada), cada w_i representa um parâmetro do neurônio (peso sináptico), w_0 é o limiar (bias ou threshold) associado a este neurônio, e a bloco que relaciona $u(t)$ com $y(t)$ é definido como função de ativação do neurônio.

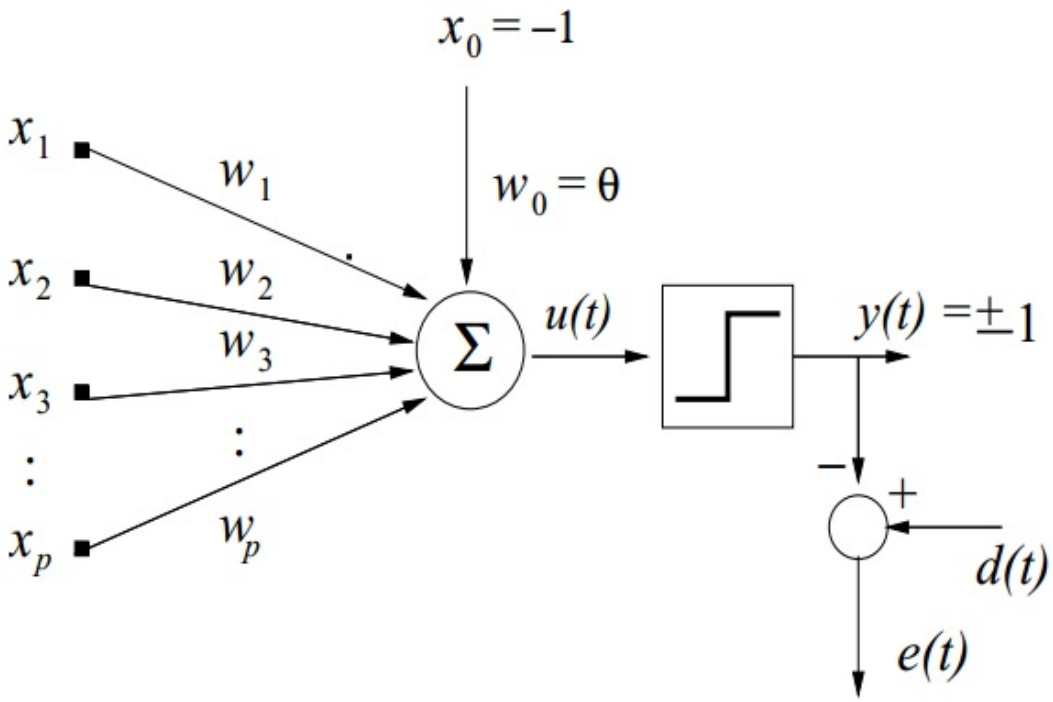

Figura 6 – Modelo Matemático de um neurônio da rede PS

A função de ativação representada na figura 6 é a função sinal, onde a saída desta

é -1 se sua entrada for negativa, e +1 se for positiva.

Assim, matematicamente, a relação entre entrada e saída de um neurônio é dada por

$$y(t) = sign(u(t)) = sign(\mathbf{w_n x_n}), \tag{3.23}$$

Diferentemente das redes PS, as redes MLP, além da camada de neurônios que tem contato diretamente com os dados de saída (camada de saída), possuem uma ou mais camadas de neurônios ocultos (camada oculta ou camada escondida), que são responsáveis pelo processamento não linear da informação de entrada, de modo a facilitar a resolução do problema para os neurônios da camada de saída.

Uma rede MLP com uma camada oculta pode ser representada por: MLP(p,q,c), onde p é o número de entradas, q é o número de neurônios ocultos e c o número de neurônios de saída.

Assim, o número de parâmetros deste classificador é dado por: $L_{mlp} = (p+1)q + (q+1)c$.

As especificações de p e c são ditadas pela forma como o problema em questão é tratado. Num caso específico de classificação de padrões por exemplo, p é definido pelo número de atributos, e c pelo número de classes do problema.

Por outro lado, não há uma heurística definida para calcular a quantidade de neurônios ocultos (q) de uma rede MLP. Esta depende da complexidade do problema, e devem ser realizados vários testes até que o valor mais adequado seja encontrado. Assim, este número é um hiperparâmetro da rede MLP.

Além das diferenças previamente citadas, a função de ativação utilizada pela rede MLP possui uma não linearidade suave, podendo obter em sua saída, diversos valores no intervalo $[0, 1]$ (caso a função utilizada seja do tipo sigmoidal), ou valores no intervalo $[-1, +1]$ (caso a função utilizada seja do tipo tangente hiperbólica).

Nesta dissertação, a tangente hiperbólica foi escolhida para ser utilizada como função de ativação. Esta é definida como

$$y_i(t) = \frac{1 - exp(-u_i(t))}{1 + exp(-u_i(t))}, \tag{3.24}$$

onde $y_i(t)$ é a saída do i-ésimo neurônio de uma camada na iteração t, e $u_i(t)$ é o produto escalar entre o vetor de pesos do neurônio e a sua atual entrada.

O algoritmo da retro propagação do erro (Error Back Propagation - BP) (RUMELHART, 1988) foi utilizado para atualizar os parâmetros da rede MLP. Este será descrito a seguir.

Inicialmente, define-se que o vetor associado a cada neurônio i da camada oculta, é representado por $\mathbf{w_i} = [w_{i0}...w_{ip}]'$ onde w_{i0} é o limiar associado ao neurônio i.

Por outro lado, a cada neurônio k da camada de saída, é associado um vetor de pesos representado por $\mathbf{m_k} = [m_{k0}...m_{kq}]'$ onde m_{k0} é o limiar associado ao neurônio k, e q é o número de neurônios da camada oculta. Ambas representações estão ilustradas na figura 7, onde $\varphi(u_i)$ representa a função de ativação (tangente hiperbólica).

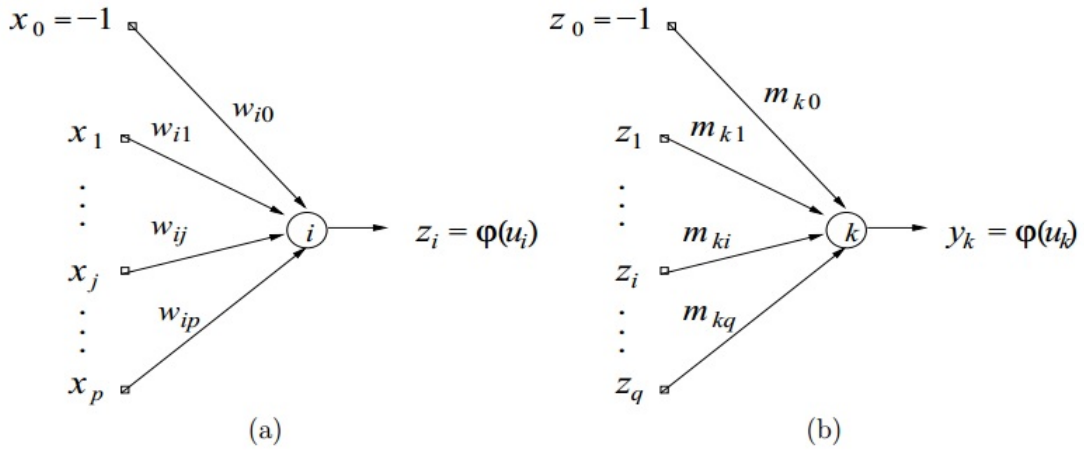

Figura 7 – Modelo dos neurônios da rede MLP (a) camada oculta (b) camada de saída

A partir dessas definições, o algoritmo de treinamento BP se divide em duas fases: sentido direto e sentido inverso.

No sentido direito (forward), o fluxo de informações se dá dos neurônios de entrada para os neurônios de saída, passando pelos da camada escondida. Essa fase se divide em dois passos:

Primeiramente, após a apresentação de um vetor de entrada $\mathbf{x_n}$, na iteração t, à rede MLP, a saída de cada neurônio da camada escondida é calculada por

$$z_i(t) = \phi_i(u_i(t)) = \phi_i(\sum_{j=0}^{p} w_{ij}(t)x_j(t)) = \phi_i(\mathbf{w_i'(t)x(t)}). \tag{3.25}$$

Em seguida, a saída de cada neurônio da camada de saída é dada por

$$y_k(t) = \phi_k(u_k(t)) = \phi_k(\sum_{i=0}^{q} m_{ki}(t)z_i(t)) = \phi_k(\mathbf{m_k'(t)z(t)}). \tag{3.26}$$

Após estes dois passos, inicia-se o sentido inverso (backward), que é a fase onde os parâmetros (pesos sinápticos) dos neurônios são calculados. Nesta fase, o fluxo de informações se dá dos neurônios de saída, para os neurônios da camada oculta.

Após calcular, no sentido direto, as ativações e saídas de cada neurônio, o primeiro passo da segunda fase de treinamento é calcular os gradientes locais dos neurônios de saída

$$\delta_k(t) = e_k(t)\phi'(u_k(t)), \tag{3.27}$$

em que $e_k(t)$ é o erro entre a saída desejada $d_k(t)$ para o neurônio k e saída gerada por ele, $y_k(t)$:

$$e_k(t) = d_k(t) - y_k(t), \tag{3.28}$$

Como a tangente hiperbólica foi escolhida como função de ativação, a derivada $\phi^{'}(u_k(t))$ é calculada por

$$\phi_k^{'}(u_k(t)) = \frac{\mathrm{d}\phi_k(u_k(t))}{\mathrm{d}u_k(t)} = \frac{1}{2}\left[1 - y_k^2(t)\right]. \tag{3.29}$$

O segundo passo do sentido inverso consiste em calcular os gradientes locais dos neurônios da camada escondida. Assim, para cada neurônio "i"desta camada, utiliza-se a seguinte expressão:

$$\delta_i(t) = \phi^{'}(u_i(t)) \sum_{k=1}^{n} m_{ki}\delta_k(t), \tag{3.30}$$

onde, assim como para os neurônios de saída, a derivada $\phi^{'}(u_i(t))$ é dada por:

$$\phi_i^{'}(u_i(t)) = \frac{\mathrm{d}\phi_i(u_i(t))}{\mathrm{d}u_i(t)} = \frac{1}{2}[1 - y_i^2(t)] \tag{3.31}$$

Por fim, o último passo da fase 2, corresponde à atualização ou ajuste de parâmetros para a camada de saída e para as camadas ocultas. No caso de uma MLP com apenas uma camada oculta, a regra de atualização dos pesos desta é dada por

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t) = w_{ij}(t) + \lambda\delta_i(t)x_j(t) \tag{3.32}$$

onde λ é a taxa de aprendizagem da rede (um hiperparâmetro desta). Por fim, para a camada de saída, a atualização dos pesos se dá por:

$$m_{ki}(t+1) = m_{ki}(t) + \Delta m_{ki}(t) = m_{ki}(t) + \lambda\delta_k(t)z_i(t) \tag{3.33}$$

A estas duas regras de aprendizagem, também pode ser adicionado um termo adicional, chamado "termo de momento", cujo objetivo é tornar o processo de aprendizagem mais estável. Com estes termos, as equações 3.32 e 3.33 passam a ser dadas por:

$$w_{ij}(t+1) = w_{ij}(t) + \lambda\delta_i(t)x_j(t) + \eta\Delta w_{ij}(t-1) \tag{3.34}$$

$$m_{ki}(t+1) = m_{ki}(t) + \lambda\delta_k(t)z_i(t) + \eta\Delta m_{ki}(t-1) \tag{3.35}$$

em que η é um hiperparâmetro da rede MLP, e é chamado "fator de momento". Este fator, por questões de estabilidade de aprendizagem, deve ser mantido, em geral, na faixa de valores entre [0.5 1].

Após mostrado o algoritmo de treinamento para a rede MLP, também será evidenciada uma classe de redes neurais supervisionadas, com uma única camada oculta, chamada *Máquina de Aprendizado Extremo* ou simplesmente rede ELM, proposta por Huang, Zhu e Ziew (2006).

3.4.2 Máquinas de Aprendizado Extremo

Diferentemente da rede MLP, na rede ELM, os pesos entre as camadas de entrada e oculta são escolhidos aleatoriamente, e os pesos entre as camadas oculta e de saída, são determinados analiticamente.

A rede ELM é uma rede que apresenta rápida velocidade de aprendizado e facilidade de implementação (HUANG; WANG; LAN, 2011) e, devido principalmente a isso, vários autores têm aplicado a rede ELM (e as mais sofisticadas variantes dela) a uma ampla gama de problemas complexos de classificação de padrões e regressão (NEUMANN; STEIL, 2013; HORATA; CHIEWCHANWATTANA; SUNAT, 2012; MOHAMMED et al., 2011; ZONG; HUANG, 2011; MICHE et al., 2011; MICHE et al., 2010; LIU; WANG, 2010; DENG; ZHENG; CHEN, 2009).

Mais especificamente, a rede ELM é uma rede *feedforward* com uma única camada oculta, que oferece menor necessidade de intervenção humana, no que se refere ao ajuste de seus parâmetros (pesos e limiares), quando comparada a redes *feedforward* mais tradicionais, tais como as redes MLP (*Perceptron multicamadas*) e RBF (*Funções de Base Radial*).

Assumindo que N pares de dados $\{(\mathbf{x}_n, \mathbf{d}_n)\}_{n=1}^{N}$ estejam disponíveis para construir e avaliar o modelo, em que $\mathbf{x}_n \in \mathbb{R}^{p+1}$ é o n-ésimo padrão de entrada[4] e $\mathbf{d}_n \in \mathbb{R}^c$ é o rótulo da classe alvo correspondente, com c denotando o número de classes.

Selecionando aleatoriamente, então, N_1 ($N_1 < N$) pares de dados a partir do conjunto de dados disponível e os organizando em colunas das matrizes $\mathbf{D}$ e $\mathbf{X}$, temos que:

$$\mathbf{X} = [\mathbf{x}_1 \mid \mathbf{x}_2 \mid \cdots \mid \mathbf{x}_{N_1}] \quad \text{e} \quad \mathbf{D} = [\mathbf{d}_1 \mid \mathbf{d}_2 \mid \cdots \mid \mathbf{d}_{N_1}]. \tag{3.36}$$

em que $\dim(\mathbf{X}) = (p+1) \times N_1$ e $\dim(\mathbf{D}) = c \times N_1$.

Para uma rede com $p+1$ unidades de entrada, q neurônios ocultos e c saídas, a i-ésima saída, para o n-ésimo padrão de entrada $\mathbf{x}_n$, é dada por

$$y_{in} = \boldsymbol{\beta}_i^T \mathbf{h}_n, \tag{3.37}$$

em que $\boldsymbol{\beta}_i \in \mathbb{R}^q$, $i = 1, \ldots, c$, é o vetor peso conectando os neurônios ocultos ao i-ésimo neurônio de saída, e $\mathbf{h}_n \in \mathbb{R}^q$ é o vetor de saídas dos neurônios ocultos para um dado padrão de entrada $\mathbf{x}(t) \in \mathbb{R}^p$. O vetor $\mathbf{h}(t)$ propriamente dito é definido como

$$\mathbf{h}_n = [f(\mathbf{m}_1^T \mathbf{x}_n + b_1) \;\; f(\mathbf{m}_2^T \mathbf{x}_n + b_2) \;\; \cdots \;\; f(\mathbf{m}_q^T \mathbf{x}_\mu + b_q)]^T, \tag{3.38}$$

em que b_l, $l = 1, \ldots, q$, é o limiar (*bias*) do l-ésimo neurônio oculto, $\mathbf{m}_l \in \mathbb{R}^{p+1}$ é o vetor de pesos do l-ésimo neurônio oculto e $f(\cdot)$ é uma função de ativação sigmoidal ou de base

[4] Primeira componente de $\mathbf{x}_n$ é igual a 1 para possibilitar a inclusão do bias.

radial. Usualmente, os vetores de peso $\mathbf{m}_l$ são aleatoriamente amostrados a partir de uma distribuição uniforme ou normal.

Seja $\mathbf{H} = [\mathbf{h}(1)\ \mathbf{h}(2)\ \cdots\ \mathbf{h}(N_1)]$ uma matriz $q \times N_1$ cujas N_1 colunas são os vetores de saída da camada oculta $\mathbf{h}_n \in \mathbb{R}^q$, $n = 1, \ldots, N_1$, em que N_1 é o número de padrões de entrada disponíveis para treinamento. Similarmente, seja $\mathbf{D} = [\mathbf{d}(1)\ \mathbf{d}(2)\ \cdots\ \mathbf{d}(N_1)]$ uma matriz $c \times N_1$ cuja n-ésima coluna é o vetor alvo (desejado) $\mathbf{d}_n \in \mathbb{R}^c$ associado com o padrão de entrada $\mathbf{x}_n$, $n = 1, \ldots, N_1$. Finalmente, seja $\boldsymbol{\beta}$ uma matriz $c \times q$, cuja i-ésima linha é o vetor de pesos $\boldsymbol{\beta}_i^T \in \mathbb{R}^q$, associado ao i-ésimo neurônio de saída, $i = 1, \ldots, c$.

Assim, essas três matrizes estão relacionadas pelo seguinte mapeamento linear:

$$\mathbf{D} = \boldsymbol{\beta}\mathbf{H}, \tag{3.39}$$

em que as matrizes $\mathbf{D}$ e $\mathbf{H}$ são conhecidas, enquanto a matriz de pesos $\boldsymbol{\beta}$ não. A solução baseada no critério OLS do sistema linear da Eq. (3.39) é dada pela inversa generalizada de Moore-Penrose como

$$\boldsymbol{\beta} = \mathbf{D}\mathbf{H}^T \left(\mathbf{H}\mathbf{H}^T\right)^{-1}. \tag{3.40}$$

A expressão mostrada na Eq. (3.40) pode ser dividida em c equações de estimação individuais, uma para cada neurônio de saída i, ou seja

$$\boldsymbol{\beta}_i = \left(\mathbf{H}\mathbf{H}^T\right)^{-1} \mathbf{H}\mathbf{D}_i^T, \qquad i = 1, \ldots, c, \tag{3.41}$$

em que $\mathbf{D}_i$ representa a i-ésima linha da matriz $\mathbf{D}$.

3.4.3 Implementação do Classificador ELM

Como mencionado anteriormente, o treinamento da rede ELM é mais rápido e de implementação mais simples que o MLP. Com isso, segue a implementação deste no software Matlab.

Primeiramente, vamos assumir que o número de neurônios ocultos está definido em q e que a dimensão dos vetores de atributos $\mathbf{x}_\mu$ está definida em p. Além disso, vamos considerar que os vetores de atributos $\mathbf{x}_\mu$, $\mu = 1, \ldots, N_1$, usados no treinamento do classificador ELM estão dispostos ao longo das colunas da matriz X e que os rótulos correspondentes estejam dispostos ao longo das colunas da matriz D.

Assim, a sequência de comandos que leva à determinação da matriz $\mathbf{H}$ e à estimação da matriz de pesos de saída $\boldsymbol{\beta}$ via Equação (3.40) é apresentada abaixo:

- M = 0.1*rand(q,p+1);
- U = M*X;
- H = 1./(1+exp(-U));
- B = D*H'*inv(H*H');

em que B denota a estimativa da matriz de pesos $\boldsymbol{\beta}$. É importante salientar que, como visto acima, o comando rand() do Matlab foi utilizado em todas as rotinas que necessitavam de geração de números aleatórios, e este comando, por padrão, implementa o algoritmo Mersenne Twister [5].

Além disso, como já mencionado na seção 3.3, esta forma de se estimar $\boldsymbol{\beta}$ não é recomendada por ser muito susceptível a erros numéricos. Neste caso, recomenda-se usar o operador *barra* (/), ou o comando PINV, ou seja:

- B = D/H;
- B = D*pinv(H);

Para a versão regularizada do classificador ELM, também é possível estimar $\hat{\boldsymbol{\beta}}$ através da escrita direta no prompt do Matlab:

- l = 0.01;
- I = ones(size(H*H'));
- B = D*H'/(H*H' + l*I);

Após finalizarmos os estudos sobre os classificadores neurais, iniciaremos agora o estudo dos baseados em vetores de suporte.

3.5 Classificadores Via Vetores de Suporte

A base dos classificadores via vetores de suporte está fundamentada na teoria do aprendizado estatístico (VAPNIK, 1998), (VAPNIK, 2000), onde tenta-se obter a minimização do risco empírico, e do risco estrutural.

Entende-se por risco empírico, como o erro obtido pelo classificador, ao buscar separar as amostras de duas classes durante o treinamento. Já o risco estrutural, está relacionado à complexidade da função que o classificador gerou para separar as classes.

Assim, visando obter o melhor resultado de separabilidade dos padrões de treinamento, através de uma função menos complexa possível, o processo de aprendizagem supervisionado busca obter uma boa capacidade de generalização (capacidade de separar os dados de testes entre as classes do problema).

Inicialmente, as Máquinas de Vetores de Suporte (Support Vector Machine - SVM), foram introduzidas para solucionar problemas binários de reconhecimento de padrões (BURGES, 1998). Mesmo existindo formas de, com estes classificadores, solucionar problemas multiclasses, nesta dissertação, utilizou-se apenas o classificador binário.

[5] http://www.mathworks.com/help/matlab/ref/randstream.list.html

O algoritmo para a classificação binária de padrões, através do SVM, é explicado a seguir. Após este, também será detalhado o classificador baseado nas Máquinas de Vetores de Suporte por Mínimos quadrados (Least Squares SVM - LSSVM).

3.5.1 Máquinas de Vetor de Suporte

Para obtermos o algoritmo não-linear do classificador SVM, faz-se necessário apresentar os classificadores SVM linear de margem rígida e o de margem flexível.

De maneira geral, assim como outros algoritmos de aprendizado de máquina, no problema de classificação, os algoritmos baseados em vetores de suporte buscam um hiperplano (problema linear) ou uma hiper superfície (problema não linear) que separa os dados de duas classes, a partir de alguns exemplos destas.

Inicialmente, considerando um problema linearmente separável, as soluções deste podem ser representadas, matematicamente, pela equação de um hiperplano, dada por

$$\mathbf{w^T x_n} + b = 0 \tag{3.42}$$

onde w é um vetor de pesos, b é o viés (também conhecido como *bias*, *threshold* ou limiar), e x_n é uma amostra de entrada.

Caso o hiperplano consiga colocar todos as amostras de uma determinada classe em posição oposta ao da outra, este representará uma solução para o problema em estudo. Neste caso, este hiperplano deve obedecer as seguintes restrições:

$$\mathbf{w^T x_i} + b > a \rightarrow d_i = +1$$
$$\mathbf{w^T x_i} + b > a \rightarrow d_i = -1$$

onde $a > 0$, e x_i representa a i-ésima amostra da treinamento.

Neste tipo de problema, há vários hiperplanos que podem dividir as amostras de entrada em dois espaços multidimensionais distintos.

Dentre estes, deve-se escolher como solução o hiperplano que possua a máxima distância em relação aos padrões mais próximos de treinamento, ou seja, possua a maior margem de separação[6]. Este hiperplano é definido como ótimo e representado pela equação

$$\mathbf{w_o^T x_n} + b_o = 0. \tag{3.43}$$

Assim, para problemas de classificação binários, a função discriminante do classificador SVM pode ser definida como:

$$g(\mathbf{x}) = sign(\mathbf{w_o^T x_n} + b_o) \tag{3.44}$$

[6] mínima distância entre hiperplano e amostra de treinamento mais próxima

de modo que

$$\mathbf{w^T x_i} + b < 0 \rightarrow f(\mathbf{x}) = -1,$$
$$\mathbf{w^T x_i} + b \geqslant 0 \rightarrow f(\mathbf{x}) = +1.$$

Para obter o hiperplano ótimo, ou seja, para encontrar os valores de w_o e b_o a partir dos dados de treinamento, pode-se, inicialmente, reescrever o problema da equação 3.43 como

$$\mathbf{w^T x_i} + b \geqslant +1 \rightarrow d_i = +1$$
$$\mathbf{w^T x_i} + b \leqslant -1 \rightarrow d_i = -1$$

onde estas equações podem ser resumidas por

$$d_i(\mathbf{w^T x_i} + b) \geqslant +1. \tag{3.45}$$

As amostras de treinamento que satisfazem a igualdade na equação anterior são denominadas vetores de suporte e são as que possuem a menor distância ao hiperplano ótimo.

Durante o treinamento dos classificadores SVM, busca-se maximizar a margem de separação. Isto é conseguido através da minimização da norma do vetor de pesos (AJALMAR, 2013), representada pela função

$$j(\mathbf{w}) = \frac{1}{2}\mathbf{w^T w} \tag{3.46}$$

onde $j(\mathbf{w})$ é a função a ser minimizada.

Ao maximizar a margem de separação, infere-se que a dimensão VC (Vapnik-Chervonenkis)[7] é minimizada.

Assim, dada as equações 3.45 3.46, temos que o treinamento de um classificador SVM pode ser convertido em um problema de otimização, onde busca-se minimizar $j(\mathbf{w}) = \frac{1}{2}\mathbf{w^T w}$ com a restrição $d_i(\mathbf{w^T x_i} + b) \geqslant +1$.

O problema de otimização com restrição anteriormente citado é conhecido como problema primal, e pode ser solucionado pelo método de Lagrange, utilizando a seguinte função lagrangeana:

$$L\left(\mathbf{w}, b, \alpha\right) = \frac{1}{2}w^T w - \sum_{i=1}^{n} \alpha_i \left(d_i \left(\mathbf{x_i^T w} + b\right) - 1\right) \tag{3.47}$$

onde α_i são os multiplicadores de Lagrange, e estes são não-negativos.

[7] dimensão associada à complexidade da função discriminante, ou seja, ao risco estrutural.

A solução deste problema de otimização é determinado pelo ponto de sela da função lagrangeana (minimizada em relação a $\mathbf{w}$ e b e maximizada em relação à α).

Nestas condições, desenvolvendo-se a equação 3.47, chega-se ao problema de otimização dual, dado por

$$maxL(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j d_i d_j x_i^T x_j \tag{3.48}$$

$$\sum_{i=1}^{n} \alpha_i d_i = 0$$
$$\alpha_i \geq 0 \quad \forall i$$

Solucionando-se este problema, encontram-se os multiplicadores de lagrange ótimos α_i^o. Por fim, através dos multiplicadores de lagrange ótimos, computam-se o hiperplano ótimo w_o e o viés ótimo pelas equações

$$\mathbf{w_o} = \sum_{i=1}^{N} \alpha_i^o d_i x_i \tag{3.49}$$

e

$$b_o = 1 - \mathbf{w}_o^T \mathbf{x}^{(s)} \tag{3.50}$$

quando $d^{(s)} = 1$, em que $(x^{(s)}, d^{(s)})$ representam um vetor de suporte.

Assim, reescrevendo-se a função discriminante definida pela equação 3.44, utilizando os resultados das equações 3.49 e 3.50, temos que

$$g(\mathbf{x}) = sign \left(\sum_{i=1}^{N} \alpha_i^o d_i \mathbf{x}_i^T \mathbf{x} + b_o \right) \tag{3.51}$$

No problema de otimização primal desenvolvido até o momento (SVM de margem rígida), considerou-se que existe um hiperplano que separa totalmente as duas classes, sem que ocorra nenhum erro. Porém, muitas vezes, esta condição não é atingida por haver sobreposição entre alguns padrões pertencentes a classes distintas, ou pela presença de *outliers*.

A flexibilização da margem de separação entre as classes é um meio para levar em conta também estas situações. A partir desta, alguns erros, durante o treinamento do algoritmo, são permitidos de modo a reduzir a complexidade da função discriminante e melhorar o desempenho desta para o conjunto de teste.

Assim, esta flexibilização se dá a partir de um relaxamento das restrições do SVM de margens rígidas, e pode ser formulada por

$$d_i(\mathbf{w^T x_i} + b) \geqslant 1 - \xi_i \tag{3.52}$$

onde os limiares $\xi_i \geq 0$ são chamados de variáveis de folga.

A partir disto, o problema de otimização primal dos classificadores SVM de margem flexível pode ser definido como

$$j(\mathbf{w}, \xi) = \frac{1}{2}\mathbf{w^T w} + C\sum_{i=1}^{n}\xi_i \tag{3.53}$$

$$d_i(\mathbf{w}^{'}\mathbf{x_i} + b) \geqslant 1 - \xi_i \quad \forall i$$
$$\xi_i \geq 0 \quad \forall i$$

onde C é a constante responsável pela regularização entre o primeiro e o segundo termo da função a ser otimizada. Esta constante é um hiperparâmetro dos classificadores SVM de margem flexível.

A partir do problema formulador anteriormente, pode-se perceber que tanto tenta-se maximar a margem de separação entre as classes (através da minimização de $\frac{1}{2}\mathbf{w^T w}$), como tenta-se minimizar os valores das variáveis de folga (ξ_i).

Do mesmo modo que os classificadores de margem rígida, os de margem flexível também podem ser resolvidos através de uma função lagrangeana, formulada por:

$$L(\mathbf{w}, b, \xi, \alpha, \beta) = \frac{1}{2}\mathbf{w^T w} + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i\left(d_i\left(\mathbf{x_i^T w} + b\right) - 1 + \xi_i\right) - \sum_{i=1}^{n}\beta_i\xi_i, \tag{3.54}$$

em que todos os elementos dos conjuntos α, β e ξ são não-negativos.

A solução deste problema é determinada ao se minimizar a função lagrangeana em relação à $\mathbf{w}$, b e ξ_i. Nestas condições, e desenvolvendo-se a equação 3.54, chega-se ao problema de otimização dual, dado por

$$max L(\alpha) = max\left\{\sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{i=j}^{n}\alpha_i\alpha_j d_i d_j \mathbf{x}_i^T\mathbf{x}_j\right\} \tag{3.55}$$

$$\sum_{i=1}^{n}\alpha_i d_i = 0$$
$$0 \leq \alpha_i \leq C \quad \forall i$$

Como pode-se perceber, a única diferença entre este problema dual e o da equação 3.48, é a restrição aplicada aos multiplicadores de lagrange α_i. Assim, tanto o classificador de margem rígida, como o de margem flexível são formatados para solucionar problemas linearmente separáveis. Porém, existem diversos problemas reais onde uma separação linear dos dados entre classes não é possível.

Por isso, para tratar este tipo de problemas, funções de Kernel $\kappa(x, x_i)$ são aplicadas aos classificadores SVM, de modo que os dados possam ser mapeados em um espaço de

características de ordem superior, fazendo com que, neste novo espaço, o problema se torne linear.

Ao utilizar estas funções, evita-se o uso explícito de um espaço de características de elevada dimensão, trabalhando-se indiretamente neste.

Os kernels mais utilizados são o linear, polinomial, e gaussiano (AJALMAR, 2013). Nesta dissertação, optou-se pelo kernel gaussiano, formulado por

$$\kappa\left(\mathbf{x}, \mathbf{x}_i\right) = exp\left\{-\frac{\left\|\mathbf{x}-\mathbf{x_i}\right\|^2}{\sigma^2}\right\} \tag{3.56}$$

onde σ^2 é um hiperparâmetro a ser definido e $\left\|\mathbf{x}-\mathbf{x_i}\right\|^2$ é a distância euclidiana entre as amostras x e x_i

Com a aplicação de um kernel ao classificador SVM, o problema de otimização dual, para o caso de margem flexível, torna-se

$$maxL(\alpha) = max\left\{\sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{i=j}^{n}\alpha_i\alpha_j d_i d_j \kappa\left(\mathbf{x}_i, \mathbf{x}_j\right)\right\}, \tag{3.57}$$

$$\sum_{i=1}^{n}\alpha_i d_i = 0$$
$$0 \leq \alpha_i \leq C \quad \forall i$$

e, por fim, sua função discriminante pode ser reescrita como

$$g(\mathbf{x}) = sign\left(\sum_{i=1}^{n}\alpha_i^o d_i \kappa\left(\mathbf{x_i}, \mathbf{x}\right) + b_o\right) \tag{3.58}$$

Por fim, para solucionar o problema de otimização dual, no software Matlab, utilizou-se a função quadprog(). Esta tem por finalidade solucionar um conjunto de equações quadráticas, podendo utilizar diversos algoritmos. Nesta dissertação, optou-se pelo algoritmo *Interior Point Convex*.

Um variante do classificador SVM é o SVM por Mínimos Quadrados (Least Squares SVM - LSSVM), que reduz o problema de programação quadrático dos SVM a um sistema linear. O algoritmo do classificador LSSVM será descrito a seguir.

3.5.2 Máquinas de Vetor de Suporte por Mínimos Quadrados

Assim como o classificador SVM, o LSSVM (SUYKENS; VANDEWALLE, 1999) também é capaz de solucionar problemas binários de classificação, porém sua formulação faz com que seu treinamento seja simplificado em relação ao SVM.

Levando em consideração o classificador de margens flexíveis, o LSSVM apresenta duas diferenças em relação ao SVM. Primeiramente, a restrição de desigualdade presente no SVM torna-se uma de igualdade. Além disso, na função custo, leva-se em conta a soma dos quadrados das variáveis de folga.

Deste modo, o problema primal de otimização do classificador LSSVM torna-se

$$j(\mathbf{w}, \xi) = \frac{1}{2}\mathbf{w^T w} + \gamma \frac{1}{2} \sum_{i=1}^{n} \xi_i^2 \tag{3.59}$$

$$d_i \left(\left(\mathbf{w^T x} \right) + b \right) = 1 - \xi_i \quad \forall i$$

onde γ, assim como C nos classificadores SVM, tem por finalidade regularizar a função de otimização. Vale ressaltar que, no problema acima, as variáveis de folga γ podem possuir valores negativos.

Este problema também pode ser solucionado por uma função lagrangeana, dada por

$$L(\mathbf{w}, b, \xi, \alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \gamma \frac{1}{2} \sum_{i=1}^{n} \xi_i^2 - \sum_{i=1}^{n} \alpha_i \left(d_i \left(\left(\mathbf{w^T x} \right) + b \right) - 1 + \xi_i \right) \tag{3.60}$$

em que as condições de otimização (minimização de $\mathbf{w}$, b e ξ_i e maximização de α_i) geram, respectivamente, as seguintes funções:

$$\begin{aligned} \mathbf{w} &= \sum_{i=1}^{n} \alpha_i x_i d_i \\ \sum_{i=1}^{n} \alpha_i d_i &= 0 \\ \alpha_i &= \gamma \xi_i \\ d_i \left(\mathbf{x}_i^T \mathbf{w} + b \right) &- 1 + \xi_i \end{aligned} \tag{3.61}$$

Pode-se rearranjar as igualdades acima, de modo a representar o problema do classificador por um sistema de equações lineares, dado por:

$$\begin{bmatrix} 0 & \mathbf{d}^T \\ \mathbf{d} & \Omega + \gamma^{-1}\mathbf{I} \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{1} \end{bmatrix} \tag{3.62}$$

onde $\Omega_{i,j} = y_i y_j \mathbf{x}_i^T \mathbf{x}_j$, $\mathbf{d} = [d_1 d_2 ... d_n]^T$, $\alpha = [\alpha_1 \alpha_2 ... \alpha_n]^T$ e $\mathbf{1} = [11...1]^T$.

Para solucionar este sistema de equações lineares, utilizou-se, no software Matlab, a função linsolve(). Esta soluciona problemas do tipo $Ax = b$, e utiliza dois algoritmos principais para isto: Fatoração LU com pivoteamento parcial (se a matriz A for quadrada),

ou Fatoração QR com pivoteamento por coluna (se as dimensões da matriz A forem diferentes).

Parar aplicar este classificador à problemas não-lineares, também pode-se utilizar o mesmo conceito de Kernel usado nos classificadores SVM. Com isso, Ω pode ser redefinido como $\Omega_{i,j} = y_i y_j \kappa(\mathbf{x}_i, \mathbf{x}_j)$ e a saída do classificador (função discriminante), fica definida como:

$$g(\mathbf{x}) = sign\left(\sum_{i=1}^{I} \alpha_i \gamma_i \kappa(\mathbf{x}, \mathbf{x_i}) + b\right). \tag{3.63}$$

Como citado anteriormente, uma grande vantagem do LSSVM em relação ao SVM é a diminuição da complexidade para resolução do problema de otimização. Porém, o LSSVM também possui limitações, onde a principal se refere aos valores dos multiplicadores de lagrange. Estes possuem valores não-nulos, o que significa perda na esparcidade na resolução do problema, pois todos os elementos de treinamento deverão ser armazenados. Isto pode trazer problemas em um grande banco de dados.

Para solucionar esta característica, existem métodos de poda para este classificador, porém estes não são tratados nesta dissertação.

3.6 Classificação Via Regressão Baseada em Distâncias

A Máquina de Aprendizado Mínimo (*Minimal Learning Machine* - MLM) é um algoritmo recente (SOUZA JUNIOR, 2015), inicialmente projetado para problemas de regressão, mas também pode ser utilizado para classificação, adicionando um passo na etapa de teste do algoritmo.

"A ideia básica por trás da MLM é a existência de um mapeamento entre a configuração geométrica dos pontos nos espaços de entrada e saída (SOUZA JUNIOR, 2015)".

Primeiramente, o banco de dados deve ser dividido em duas partes: conjunto de treinamento e conjunto de teste. A partir disto, K pontos de referência devem ser aleatoriamente escolhidos. Estes pontos de referência correspondem a algumas amostras do conjunto de treinamento. Cada amostra possui suas entradas (atributos) e saídas (rótulos).

O número de pontos de referência é o único hiperparâmetro deste algoritmo. O modo como este número é definido será especificado no capítulo 4.

A etapa de treinamento é iniciada calculando-se $D_x \epsilon \Re^{N_t x K}$, onde cada elemento d_{nk} representa a distância da n-ésima amostra de treinamento até a entrada do k-ésimo ponto de referência (representada por $\mathbf{m}_x$). Neste trabalho, a distância Euclidiana é utilizada para calcular os elementos d_{nk}.

O segundo passo desta etapa, consiste em calcular $\Delta_y \epsilon \Re^{NxK}$, onde cada elemento desta matriz representa a distância de cada saída das amostras do conjunto de treinamento a cada saída dos pontos de referência (representadas por $\mathbf{t_y}$).

Por fim, deve-se estimar um modelo de regressão linear relacionando as matrizes D_x e Δ_y. Este modelo linear é representado na equação 3.64, onde E corresponde aos resíduos, e $B \epsilon \Re^{KxK}$ pode ser estimado pelo método da pseudo-inversa, representado na equação 3.65.

$$\Delta_y = D_x.B + E \tag{3.64}$$

$$\hat{B} = (D_x^{'} D_x)^{-1} D_x^{'} \Delta_y \tag{3.65}$$

Já a etapa de teste consiste de, para cada amostra de teste $\mathbf{x_i}$, calcular um vetor $d \epsilon \Re^{1xK}$, onde cada elemento representa a distância entre as entradas dos pontos de referências e da amostra de teste $\mathbf{x_i}$.

Em seguida, as distâncias entre as saídas da amostra $\mathbf{x_i}$ e dos pontos de referência são estimadas por

$$\hat{\delta}(\mathbf{y}, \mathbf{t_k}) = d\hat{B}, \tag{3.66}$$

onde cada elemento de $\hat{\delta}(\mathbf{y}, \mathbf{t_k}) \epsilon \Re^{1xK}$ representa a distância entre a saída dos pontos de referência e a da amostra $\mathbf{x_i}$.

Finalmente, a saída $\mathbf{y}$ é estimada pela solução de um problema de otimização descrito na equação 3.67.

$$\hat{y} = argmin_y \sum_{k=1}^{K} \left((\mathbf{y} - \mathbf{t_k})^{'} (\mathbf{y} - \mathbf{t_k}) - \hat{\delta}^2 (\mathbf{y}, \mathbf{t_k}) \right) \tag{3.67}$$

Neste trabalho, a função fsolve() do Matlab, juntamente com o algortimo Levenberg-Marquardt, é utilizada para solucionar o problema supracitado. Esta função é capaz de solucionar um sistema de equações não lineares, o que a tornou um bom candidato para ser utilizada.

Após a descrição de cada algoritmo de aprendizado de máquina, as técnicas utilizadas, juntamente com estes, para a detecção de falhas em MIT, serão detalhadas no capítulo a seguir.

Capítulo 4

Metodologia dos Experimentos

Neste capitulo, são mostradas as técnicas que foram utilizadas para: analisar os dados, para comparar os classificadores e que, juntamente com estes, foram utilizadas para detectar as falhas nos motores de indução trifásicos. Por fim, alguns testes, que foram feitos devido a características específicas de cada classificador, são evidenciados.

4.1 Análise dos Dados

Como mostrado no capítulo 2, já haviam sido realizados alguns estudos teóricos sobre as harmônicas que indicam falhas por curto circuito, e alguns estudos empíricos sobre as estatísticas destes atributos.

Porém, a única métrica estatística utilizada para analisar os dados, foi a variância destes. Assim, nesta dissertação, utilizaram-se mais duas ferramentas para analisar os dados.

Primeiramente, diagramas de caixa foram utilizados para verificar a faixa de valores nas quais os dados de cada classe variavam.

Isto foi feito para verificar se havia um limiar de separação, em uma dimensão, entre os dados de motores em funcionamento normal ou com falhas.

Além dos diagramas de caixas, foram feitos gráficos de dispersão com as harmônicas que seriam utilizadas para a avaliação dos classificadores.

A partir destes gráficos, buscava-se um limiar de separação, em duas dimensões, entre as classes de motores em funcionamento normal ou com falhas.

Se comprovada a existência de um destes limiares, não haveria a necessidade de aplicação de algoritmos complexos para solução do problema.

4.2 Banco de Dados

Como explicado no capítulo 2, o banco de dados reais utilizado nesta dissertação possui 294 amostras representantes de motores. Se o problema for tratado como binário, 42 destas amostras representam a classe de motores em funcionamento normal, e o restante (252) são representantes de motores com falhas por curto circuito entre espiras. Este banco

de dados é definido como Banco de dados 1. Pode-se perceber que o banco de dados 1 é bem desequilibrado, visto que a quantidade de amostras da classe de falhas (86% do total) equivale a seis vezes a quantidade de amostras da classe de motores em funcionamento normal.

Além disso, este conjunto não representa uma situação real na indústria, visto que, nesta, a quantidade de amostras de motores em funcionamento normal é bem maior (apenas uma pequena parcela de motores falha).

Para tentar sanar este problema, de modo a equilibrar o número de dados nas duas classes, algumas amostras de motores em funcionamento normal foram geradas, artificialmente, de acordo com os passos a seguir:

- Selecionava-se, aleatoriamente, uma amostra da classe de funcionamento normal. - Adicionava-se, a cada atributo desta amostra, um valor aleatório, de distribuição uniforme, com valores entre $-0.01 * x_{nj}$ e $+0.01 * x_{nj}$, onde x_{nj} corresponde ao valor de determinado atributo.

Estes passos foram feitos até que fossem gerados 210 amostras de motores em funcionamento normal. Assim, o novo banco de dados é composto de 504 dados, contendo 252 dados de cada classe. Este banco de dados é definido como o Banco de Dados 2.

4.3 Classificadores

Nesta seção, alguns detalhes sobre a implementação e a utilização dos classificadores são discutidos.

Primeiramente, como não havia um conhecimento prévio da distribuição de probabilidades intraclasses, supôs-se uma distribuição gaussiana dos dados, visto que estes são reais e foram extraídos da bancada de testes explicada no capítulo 2. Assim, utilizaram-se Classificadores Gaussianos (Bayesianos).

Este classificador é não paramétrico (não possui hiperparâmetros), no sentido que apenas extrai informações do dados de treinamento (matriz de covariância e média destes) para classificar os dados de teste.

Uma etapa muito importante deste classificador é a verificação do condicionamento da matriz de covariâncias, visto que, como esta é invertida no processo de teste do classificador, um mal condicionamento desta poderá trazer problemas de instabilidade numérica.

Também, dado que a distribuição de probabilidade das classes não era conhecida, supôs-se uma separação linear destas, pois, se estas assim fossem, não haveria a necessidade de algoritmos não lineares ou mais complexos para a solução do problema em questão.

Vários algoritmos, tais como "Perceptron Simples", "Classificador Bayesiano Linear"e "Mínimos Quadrados Recursivos"(RLS), poderiam ser utilizados para atestar ou não esta separação, porém o método dos mínimos quadrados ordinários (OLS) foi utilizado, visto que, além da possibilidade de aplicá-lo diretamente aos dados como classificador, este método também faz parte dos algoritmos de outros classificadores.

Em relação à algoritmos não lineares, as Redes Perceptron Multicamadas (MLP), são conhecidas como aproximadores universais de funções.

Com esta topologia, foram utilizados dois algoritmos para treinamento: o treinamento pela retro-propagação do erro (Backpropagation) e um mais recentemente e com diversas aplicações: máquina de aprendizado extremo (extreme learning machine - ELM).

Além disso, também verificou-se se havia alguma relação entre os labels e o quão "dentro"da classe o dado estava, para isso, utilizou-se um classificador basado em distâncias: a máquina de aprendigem mínima (MLM).

Por fim, como os resultados ainda estavam não-satisfatórios, pensou-se em utilizar o classificador SVM, pois além deste estar entre os estados da arte de classificadores, também são selecionados apenas alguns dados de treinamento para traçarem o hiperplano de decisão ótimo.

Já o LSSVM parte da mesma idéia do SVM, porém este tem o treinamento bem mais rápido, e utiliza todos os dados de treinamento como vetores de suporte. Se a taxa de acerto deste classificador se mostrasse bem superior aos outros, valeria o custo computacional.

4.4 Metodologia de Comparação entre Classificadores

Os experimentos utilizados nesta dissertação para comparar o desempenho dos classificadores quanto a sua capacidade de detectar falhas seguem a metodologia ilustrada na figura 8.

Ao finalizar todos os passos desta metodologia, ocorre a realização de determinado algoritmo. Para cada conjunto de passos determinados, foram feitas 50 realizações para cada classificador.

Deste modo, as estatísticas (máximo, mínimo, média, desvio padrão, mediana) referentes à capacidade de generalização[1] (taxa de acerto na detecção) de cada ferramenta pôde ser observada.

Essas estatísticas serão, no próximo capítulo, evidenciadas através de matrizes de confusão, tabelas e gráficos de caixa.

[1] Entende-se por generalização como o poder de um classificador, após o seu treinamento, rotular corretamente novas amostras.

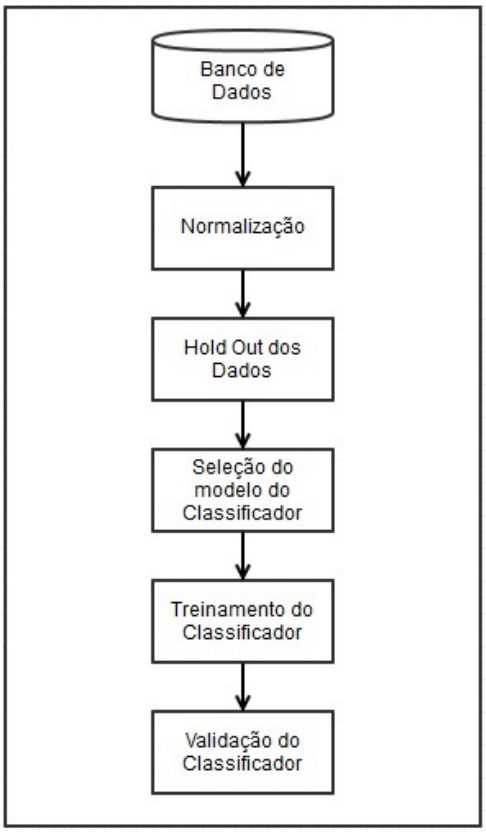

Figura 8 – Metodologia Dos Experimentos

Inicialmente, define-se um banco de dados a ser tratado pelos classificadores. Nesta dissertação, pode-se optar pelo banco de dados 1 (com a quantidade desequilibrada de amostras de motores com ou sem falhas) ou pelo banco de dados 2 (com equilíbrio entre a quantidade de amostra com ou sem falhas). É importante salientar que, em ambos, cada amostra possui 6 atributos (0.5fs 1.5fs 2.5fs 3fs 5fs 7fs).

Após esta definição, cada amostra de entrada, é normalizada. Isto é feito para que nenhum componente do vetor de atributos se sobreponha aos outros durante o processo de classificação.

Existem diversas formas de normalizar os dados, dentre elas, podem ser citadas a normalização no intervalo entre [-1 e +1], a normalização pela média e pelo desvio padrão de cada atributo, e a normalização no intervalo entre [0 e 1].

As duas primeiras citadas foram utilizadas (uma por vez) juntamente com os outros passos da metodologia. Estas normalizações podem ser resumidas, respectivamente, na equação

$$x_{nj}^{*} = 2\left(\frac{x_{nj} - x_{nj}^{max}}{x_{nj}^{max} - x_{nj}^{min}}\right) - 1, \tag{4.1}$$

onde x_{nj}^{max} e x_{nj}^{min} são, respectivamente, o valor máximo e mínimo de um determinado

atributo, dentre todas as amostras disponíveis; e na equação

$$x^*_{nj} = \frac{x_{nj} - \mu_j}{\sigma_j}, \tag{4.2}$$

onde μ_j é a média do atributo j e σ_j é o desvio padrão deste mesmo atributo.

Os demais passos da metodologia aplicada são descritos a seguir.

4.4.1 Hold Out

Esta etapa consiste em, aleatoriamente, separar algumas amostras do banco de dados, para que sejam utilizadas apenas para testar os classificadores.

Assim, o conjunto de dados com N amostras, é dividido entre $N1$ amostras de treinamento e $N2$ amostras de testes, onde $N = N1 + N2$.

Nesta dissertação, foram utilizados dois métodos de *Hold Out*:

- Hold Out 01: de todo o banco de dados utilizado, são retirados, aleatoriamente, (utilizando a função *rand()* do Matlab) 20% das amostras. Estas amostras só são apresentadas ao classificador para gerar as taxas de acerto, após o modelo deste ter sido determinado. Já as 80% das amostras restantes são utilizadas para selecionar o modelo do classificador e treiná-lo.
- Hold Out 02: as amostras são separadas por classe e, da classe que possui a menor quantidade de amostras, são separadas, aleatoriamente, 80% destas para seleção do modelo e treinamento do classificador. Das demais classes, uma quantidade equivalente de amostras é separada, de modo que, durante o treinamento, o número de amostras por classe seja equilibrado.

Este último método foi aplicado ao Banco de Dados 1, visto que, neste caso, o desequilíbrio entra a quantidade de amostras de motores com ou sem falhas é muito grande.

Após a separação de dados para treinamento e teste, é feita a seleção do modelo do classificador.

4.4.2 Seleção do modelo

Esta etapa só está presente se o algoritmo em questão possuir hiperparâmetros que devem se ajustados antes do treinamento. No caso da rede MLP, por exemplo, um dos hiperparâmetros a ser ajustado é o número de neurônios da camada oculta.

Esta seleção ocorre apenas com os dados separados para treinamento. Isto se deve ao fato de que o modelo do classificador também carrega informações dos dados, e para

que se meça a capacidade de generalização deste classificador, não deve haver interferência dos dados de teste.

Com isso, esta etapa consiste dos seguintes passos:

- Busca em grade (*grid search*) - Define-se uma faixa de valores para determinados parâmetros de um classificador. A partir disso, é fornecido, ao passo seguinte, todas as combinações possíveis de valores dos parâmetros.
- Validação cruzada: - Divide-se, aleatoriamente, as amostras de treinamento em r partes iguais (*r-fold cross-validation*). - A partir dessa divisão, são realizadas r rodadas de treinamento e teste, para cada conjunto de valores dos parâmetros disponibilizados pela busca em grade. - Com isso, é calculada a taxa de acerto médio, no teste, de cada conjunto de valores de parâmetros.
- Escolha dos hiperparâmetros: - O conjunto de valores de parâmetros que obtiverem a maior taxa de acerto médio é selecionado para a próxima etapa.

Primeiramente, levando em conta todos os classificadores, estes possuem os seguintes hiperparâmetros:

Tabela 1 – Hiperparâmetros dos Classificadores

Classificador	Hiperparâmetros
OLS	(não possui)
Gaussiano	(não possui)
MLP	q | λ | η
ELM	q
SVM	C | σ^2
LSSVM	γ | σ^2
MLM	K

Para a busca em grade, as seguintes faixas de valores foram definidas para os hiperparâmetros:

- OLS: Este classificador não possui hiperparâmetros, mas apenas parâmetros que são calculados diretamente dos dados.
- Gaussiano: Como o OLS, este não possui hiperparâmetros, mas apenas parâmetros que são calculados diretamente dos dados.
- MLP: os valores $\lambda = 0.05$ e $\eta = 0.75$ foram mantidos constantes, e variou-se o número de neurônios ocultos q no intervalo entre [2 20].
- ELM: o número de neurônios ocultos foi variado no intervalo entre [5 e 30].

- SVM: o parâmetro de regularização e o hiperparâmetro do kernel tiveram, para a busca em grade, respectivamente os seguintes valores: C =[0.5 5 10 15 25 50 100 250 500 1000] e σ^2 =[0.01 0.05 0.1 0.5 1 5 10 50 100 500].
- LSSVM: já para este classificador, o parâmetro de regularização e o hiperparâmetro do kernel tiveram, para a busca em grade, respectivamente os seguintes valores: γ =[2^{-5} a 2^{20}] σ^2 =[2^{-10} a 2^{10}]
- MLM: por fim, o número de pontos de referência K foi variado no intervalo entre [2 e 20].

Em seguida, definiu-se uma divisão r de 5 partes no conjunto de treinamento, ou seja, foi feita uma validação cruzada de 5 partes (5-fold cross-validation).

Por fim, verificou-se, para as diversas realizações dos classificadores, quais valores de hiperparâmetros eram comumente selecionados como sendo ótimos para o conjunto de treinamento. Se estes valores permanecessem constantes ou variassem pouco, eles poderiam ser definidos como os hiperparâmetros ótimos e o algoritmo da validação cruzada juntamente com a busca em grade se tornaria desnecessário.

4.4.3 Treinamento do Classificador

Após a seleção dos modelos, cada algoritmo é treinado com todos os dados separados para treinamento, de modo que os parâmetros fossem atualizados.

Nesta etapa, como todos os algoritmos utilizados nesta dissertação possuem o treinamento supervisionado, os rótulos das amostras são utilizados.

Levando em conta todos os classificadores, estes possuem os seguintes parâmetros:

Tabela 2 – Parâmetros dos Classificadores

Classificador	Parâmetros		
OLS	$\mathbf{W}$		
Gaussiano	μ_i	$\mathbf{\Sigma}_i$	
MLP	$\mathbf{W}$	$\mathbf{M}$	
ELM	$\mathbf{W}$	β	
SVM	n_{sv}	α	$\mathbf{X}$
LSSVM	n_{sv}	α	$\mathbf{X}$
MLM	$\mathbf{B}$	$\mathbf{M}_k$	

A partir da tabela 2, pode-se notar que existem muitos parâmetros que são vetores ou matrizes. Para quantificar o total de parâmetros, deve-se saber a dimensão destes vetores e matrizes. Em resumo, a quantidade total de parâmetros de cada classificador é:

- L_{OLS}: $c \times p + 1$ (referente a $\mathbf{W}$)

- Gaussiano: Para todas as classes, existem $c \times (p + p \times p)$ parâmetros.
- MLP: dado um problema MLP(p,q,c), a quantidade de parâmetros se resume a $((p + 1) \times q) + (q \times c)$
- ELM: dado um problema ELM(p,q,c), a quantidade de parâmetros se resume a $((p + 1) \times q) + (q \times c)$
- SVM: A quantidade total de parâmetros deste classificador depende dos vetores de suporte selecionados durante o treinamento. Assim, esta quantidade é de: $n_{sv} \times (1+p)$, onde o 1 se refere aos multiplicadores de lagrange e o p se refere às amostras de treinamento que devem ser armazenadas.
- LSSVM: Assim como no SVM, a quantidade total de parâmetros deste classificador depende dos vetores de suporte selecionados durante o treinamento. Assim, esta quantidade é de: $n_{sv} \times (1 + p)$, onde o 1 se refere aos multiplicadores de lagrange e o p se refere às amostras de treinamento que devem ser armazenadas.
- MLM: a quantidade de parâmetros deste classificador depende do hiperparâmetro K, definido este, esta quantidade se torna $(K \times K) + (K \times p)$

4.4.4 Teste do Classificador

Finalmente, após cada classificador ser devidamente treinado com as $N1$ amostras de treino, a cada realização deste, apresenta-se as $N2$ amostras restantes a este. Para calcular a taxa de erro/acerto de cada classificador, compara-se a saída destes com o rótulo original das amostras. Se a saída do classificador for igual ao rótulo da amostra, um acerto é computado, caso contrário, um erro é computado a este.

Assim, a taxa de acerto de cada classificador é a razão entre quantidade de amostras rotuladas corretamente pelo número total de amostras.

Para comparar a acurácia dos classificadores, utilizaram-se, inicialmente, diagramas de caixa (*boxplot*) e matrizes de confusão.

A utilização de diagramas de caixas é um meio muito útil para visualizar e comparar desempenho entre classificadores, visto que, através destes, a distribuição da taxa de acerto para um conjunto de realizações de cada algoritmo pode ser observada, evidenciando informações, tais como outliers (desempenhos atípicos). Além disso, cada quartil[2] desta distribuição pode ser observado.

Já através das matrizes de confusão F, pode-se verificar, em uma realização do algoritmo, entre quais classes estão ocorrendo os maiores erros do classificador.

[2] Quartis são medidas de posição para uma dada distribuição. Estes dividem os resultados em 4 partes iguais. Ou seja, o primeiro quartil indica que 25% dos dados estão abaixo deste. Já o segundo, indica que 50% dos dados estão abaixo deste (mediana), e assim sucessivamente.

Estas matrizes tem dimensão $c \times c$, onde as linhas representam a classe C_i a qual a amostra pertence, e as colunas representam as saídas $\hat{y}$ estimadas pelo classificador, ou seja, a qual classe o classificador atribuiu as amostras.

Deste modo, o elemento f_{12}, por exemplo, indica a quantidade de amostras pertencentes à classe 1, que foram atribuídas à classe 2. Assim, a soma dos elementos da diagonal da matriz de confusão indica a quantidade de dados classificados corretamente.

4.5 Opção de Rejeição

Em problemas de classificação, normalmente, atribui-se à uma nova amostra o rótulo da classe para qual a sua função discriminante obteve o maior valor, mesmo quando a probabilidade a posteriori de duas classes é muito próxima.

Esta característica pode levar ao aumento na taxa de erros de classificadores, visto que existem muitos casos onde há padrões difíceis de serem classificados.

Assim, para evitar este tipo de problema, a opção de rejeição é inserida para salvaguardar contra erros excessivos e tomadas de decisão difíceis (**??**).

Por definição, opção de rejeição é qualquer estratégia que permita ao classificador não-classificar determinada amostra de entrada caso a avaliação das saídas seja ambígua. Assim, a classificação não se resume apenas a separar as amostras entre classes, mas, também, converter potenciais erros em rejeição.

Sem a utilização desta estratégia, ao se avaliar classificadores, apenas leva-se em conta a taxa de erros destes quanto a solução de determinado problema de classificação. Ao incluí-la, também deve-se medir a taxa de rejeição dos dados.

Assim, o melhor classificador não será simplesmente o que atingiu a menor taxa de erros, mas, sim, o que obteve o melhor balanceamento entre esta métrica e a quantidade de dados rejeitados.

Com a inclusão da opção de rejeição, a taxa de rejeição é definida como

$$R = \frac{n^o \;\; de \;\; amostras \;\; rejeitadas}{n^o \;\; total \;\; de \;\; amostras}. \tag{4.3}$$

Já a taxa de erros é dada por

$$E = \frac{n^o \;\; de \;\; amostras \;\; classificadas \;\; erroneamente}{n^o \;\; total \;\; de \;\; amostras \;\; - \;\; n^o \;\; de \;\; amostras \;\; rejeitadadas}. \tag{4.4}$$

Existem várias formas de adicionar a opção de rejeição aos classificadores, podendo ser incorporada diretamente ao treinamento ou medindo-se apenas as saídas do classificador e aplicando alguma heurística.

Nesta dissertação, implementou-se o método da definição de um limiar ótimo para aceitar a saída de um classificador. Neste, para que uma amostra seja atribuída a uma classe, não basta apenas que a função discriminante desta classe tenha o valor máximo dentre as outras, mas, também, esta função deve ter um valor maior do que um limiar definido pelo usuário.

4.6 Comparação entre Classificadores

Além das ferramentas previamente citadas (tais como *Boxplot* e taxa de acerto) para comparar classificadores, foram utilizadas mais duas ferramentas estatísticas: o coeficiente de variação e o teste de Mc Nemar.

O coeficiente de variação é uma medida de variabilidade relativa da taxa de acerto do classificador (BOSLAUGH; WATTERS, 2008). Este é calculado dividindo-se o desvio padrão (σ) da taxa de acerto pelo seu valor médio (μ), ou seja:

$cv = 100 \times \frac{sd}{md}\%$

Numa comparação entre classificadores que apresentam desempenhos médios, equivalentes, quanto menor o valor de cv, melhor é o classificador.

Além disso, o teste de Mc Nemar (BOSTANCI; BOSTANCI, 2013) é um variante do teste χ^2, onde este é não paramétrico e é utilizado para avaliar estatisticamente pares de dados. Nesta dissertação, esta ferramenta é utilizada para verificar o quão diferentemente dois classificadores se comportam na solução de determinado problema de classificação.

Formalmente, de posse de dois classificadores nomeados como A e B, pode-se obter, em uma realização de teste, os resultados mostrados na tabela 3. onde Nff representa

Tabela 3 – Resultados de dois Classificadores

	Falha no Algoritmo A	Sucesso no Algoritmo A
Falha no Algoritmo B	Nff	Nsf
Sucesso no Algoritmo B	Nfs	Nss

o número de vezes que os dois classificadores falharam ao tentar classificar uma nova amostra, e Nss representa sucesso na classificação para os dois algoritmos. Por fim, Nfs e Nfs indicam a quantidade de vezes onde os classificadores divergiram quanto aos resultados de classificação das amostras.

Deste modo, estes dois últimos resultados são mais indicativos das discrepâncias entre dois algoritmos. Para quantificar estas divergências, o teste de Mc Nemar aplica "z score", formulado por

$$z = \frac{(|Nsf - Nfs| - 1)}{\sqrt{Nsf + Nfs}}. \tag{4.5}$$

Assim, quando $z = 0$, infere-se que os dois algoritmos possuem a mesma performance. Quanto mais diferente de 0 for o valor de z, maior é a significância da diferença de performance.

4.7 Testes Adicionais

Como será evidenciado no capítulo de resultados, pela metodologia principal de comparação, dois classificadores se sobressaíram em relação aos demais. Assim, algumas metodologias foram aplicadas aos demais classificadores, na tentativa de melhorar o resultado destes quanto a detecção de falhas.

A principal modificação destas metodologias em relação a principal, esta na rotulação geral do banco de dados. Nestas, começa-se a tratar o problema de detecção de falhas, durante o treinamento, como multiclasse, e considera-se este problema como binário durante o teste.

Assim, foram feitas três metodologias adicionais:

- Metodologia 1: problema composto de 7 classes, onde cada combinação de tipo e nível de falha é considerada uma classe (AI1, AI2, AI3, BI1, BI2, BI3). Neste problema, cada classe de falha possui 42 amostras, e a classe de funcionamento normal pode possuir 42 amostas (caso fosse utilizado o banco de dados 1) ou 252 amostras (caso o banco de dados utilizado fosse o 2).

- Metodologia 2: problema composto por 3 classes, onde cada tipo de falha é considerado uma classe (AI e BI). Neste problema, cada classe de falha possui 126 amostras, e a classe de funcionamento normal pode possuir 42 amostas (caso fosse utilizado o banco de dados 1) ou 252 amostras (caso o banco de dados utilizado fosse o 2).

- Metodologia 3: problema composto de 4 classes, onde cada nível de falha é considerado uma classe(AI1 BI1 X AI2 BI2 X AI3 BI3). Neste caso, cada classe de falha possui 84 amostras, e a classe de funcionamento normal pode possuir 42 amostas (caso fosse utilizado o banco de dados 1) ou 252 amostras (caso o banco de dados utilizado fosse o 2).

Vale salientar que estas duas última metodologias foram aplicadas para verificar se os classificadores conseguiam diferenciar melhor ”níveis de falhas”ou ”tipos de falhas”.

Em todas estas metodologias, atualizava-se os parâmetros, durante o treinamento, a partir da quantidade de classes definida pelo problema. Por outro lado, durante o teste, a saída de cada classificador foi comparada ao rótulo de cada amostra. Se em determinada amostra, o classificador fizesse confusão entre níveis ou tipo de falhas, isto

não era considerado um erro. Apenas quando o classificador rotulava uma amostra normal como falha (ou vice-versa) um erro era computado.

Após explicados os passos e ferramentas utilizadas juntamente com os classificadores, no capítulo a seguir, serão discutidos os resultados gerais deste trabalho.

Capítulo 5

Análise dos Resultados

A partir da metodologia apresentada no capítulo anterior, aqui são indicados os resultados dos experimentos realizados com as técnicas e classificadores citados.

5.1 Análise dos Dados

Em relação à utilização dos diagramas de caixas, a partir das figuras 9 e 10, foi verificado que, para os dados não normalizados, existem faixas de valores onde apenas os motores com falhas apresentam amostras.

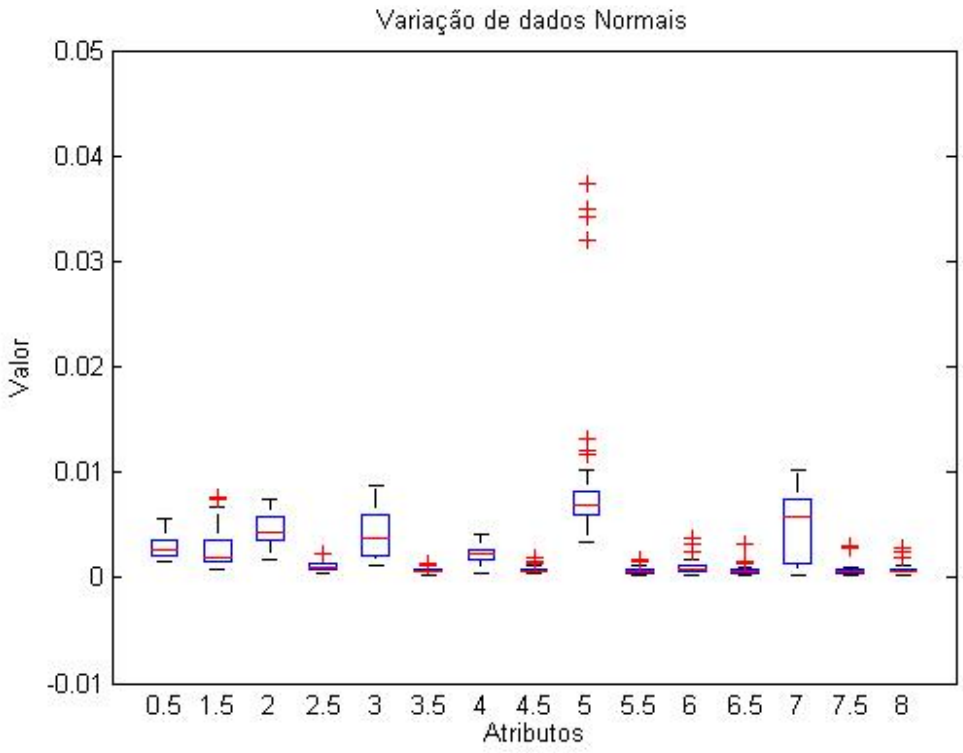

Figura 9 – Distribuição dos dados normais, não-normalizados

Porém, na faixa de valores de atributos onde se encontram as amostras do motor em funcionamento normal, também há amostras deste com falhas. Ou seja, não foi identificado um limiar que separasse as duas classes, mas sim um que demonstra a ocorrência de falhas.

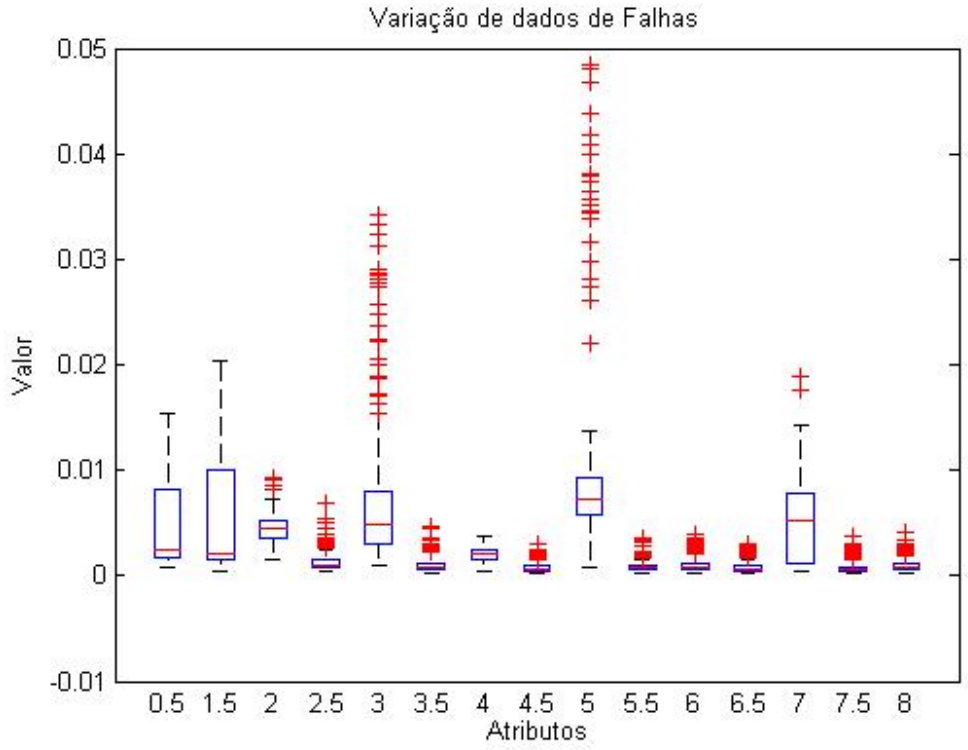

Figura 10 – Distribuição dos dados de falhas, não-normalizados

Isto, como será mostrado a seguir, é um dos fatores que contribuíram na classificação errônea de vários motores em funcionamento normal.

Em seguida, as amostras foram normalizadas pela média e pelo desvio padrão de cada atributo. Neste caso, como pode ser observado nas figuras 11 e 12, o mesmo padrão se repetiu: não há regiões bem distintas de dados do motor com ou sem falhas, apenas regiões onde só existem representantes de motores com falhas.

Da mesma forma, através de diversas combinações entre dois atributos do problema, não foi percebido nenhum padrão de separabilidade das classes, e nem uma distribuição de probabilidade intraclasses, fazendo com que fosse necessário a utilização de classificadores.

Nas figuras 13 e 14, são mostradas, respectivamente, combinações entre os atributos $0.5f_s$ e $1.5f_s$, e entre os atributos $5f_s$ e $3f_s$. Novamente pode-se observar o padrão anterior.

Este resultado poderia ser um ponto de partida para um sistema de detecção de falhas, visto que, se a amostra atingir um limiar definido em determinado atributo, pode-se diretamente considerar este dado como falha, sem a necessidade de aplicação dos classificadores.

Após esta análise dos dados, os classificadores foram comparados utilizando a metodologia descrita em 4.4. Os resultados desta comparação são discutidos a seguir.

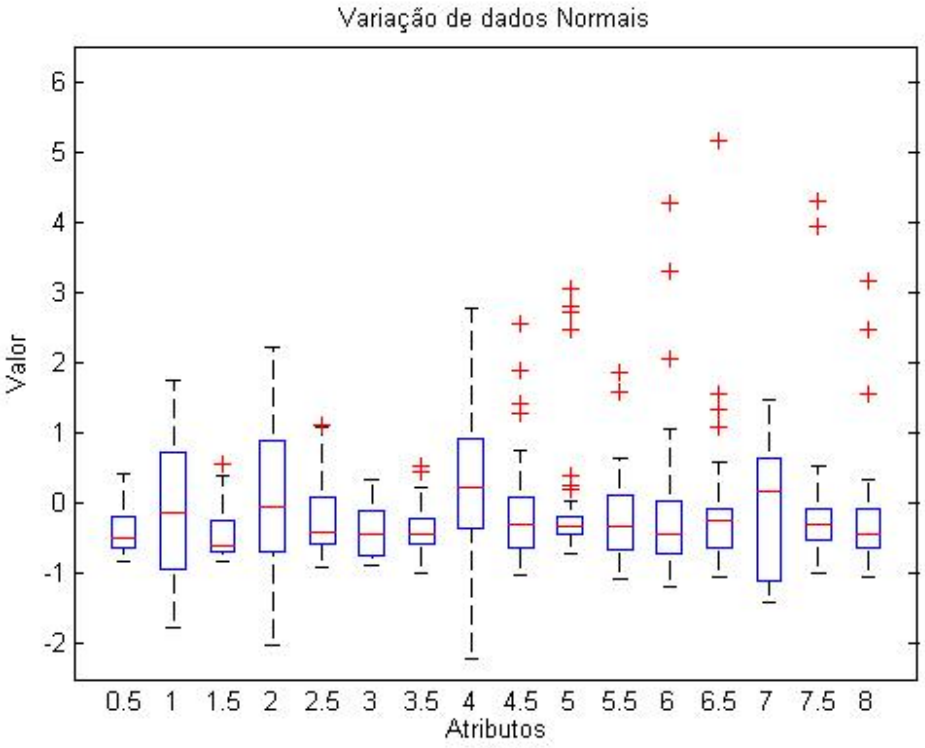

Figura 11 – Distribuição dos dados normais, não-normalizados

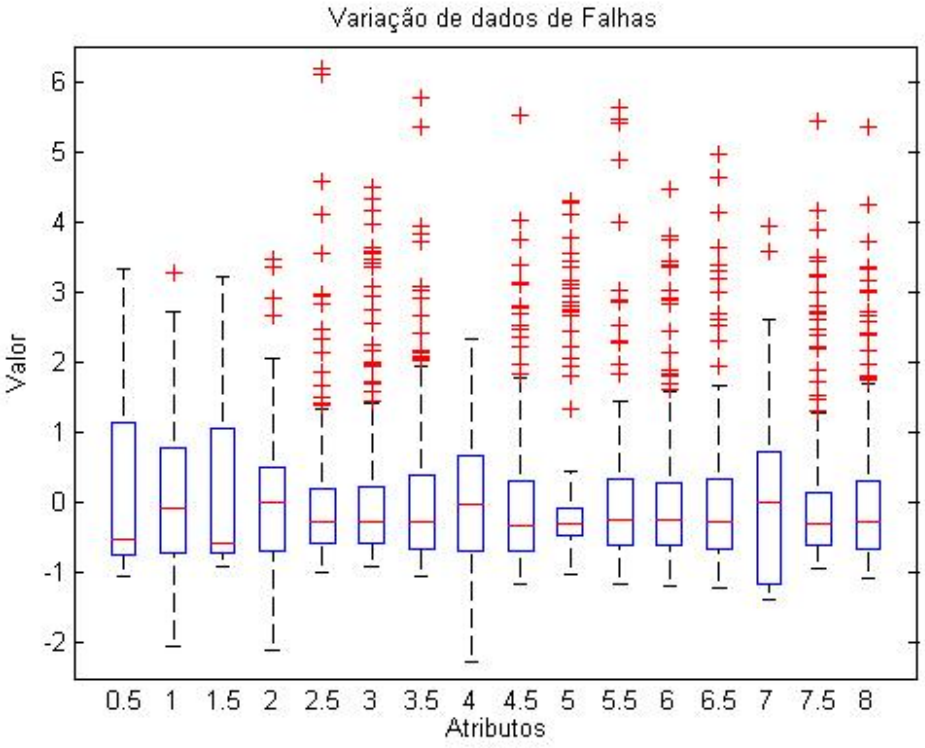

Figura 12 – Distribuição dos dados de falhas, não-normalizados

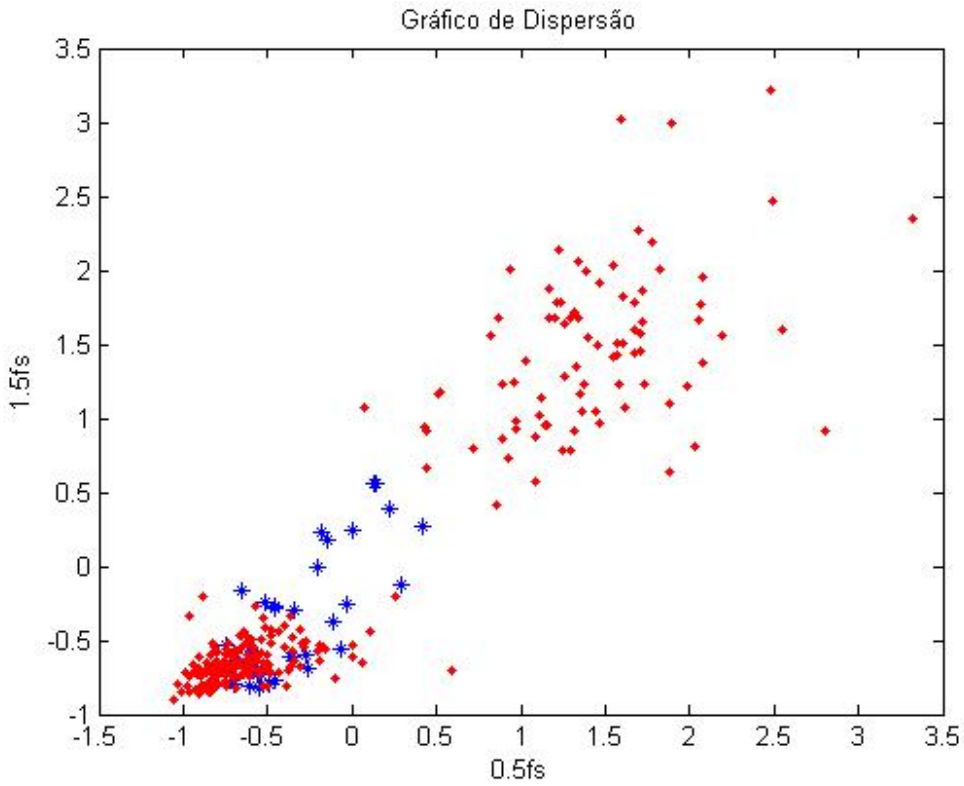

Figura 13 – Gráfico de Dispersão 1

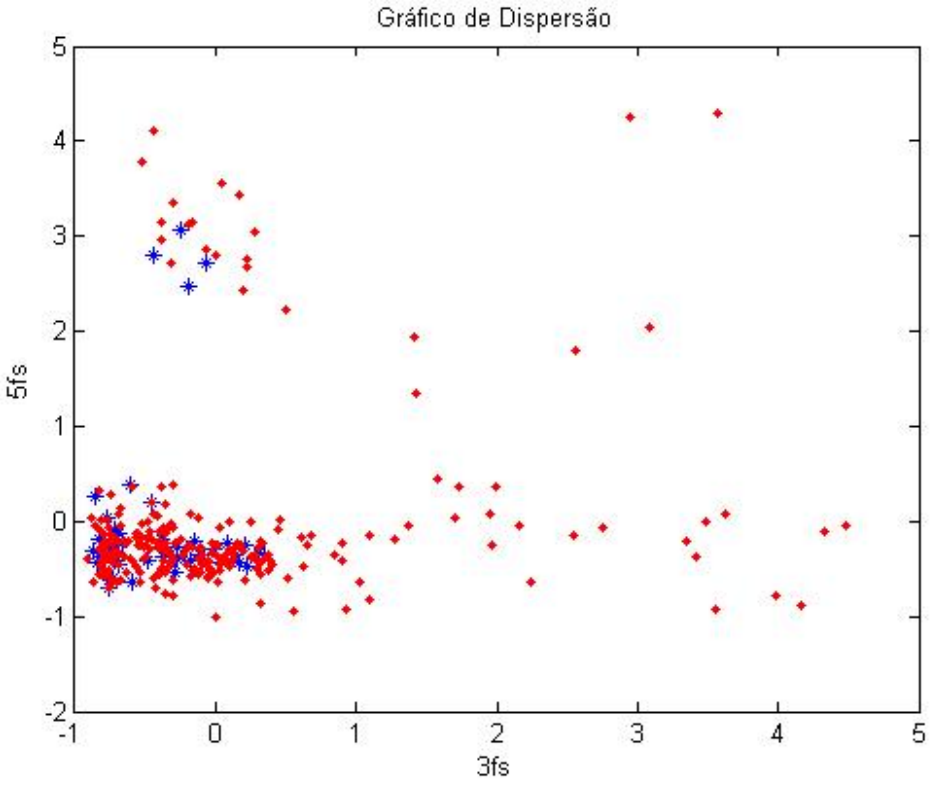

Figura 14 – Gráfico de Dispersão 3

5.2 Comparação entre Classificadores

Durante a comparação entre os classificadores, foram testadas diferentes combinações de características para avaliar o desempenho destes quanto a detecção de falhas. Cada combinação de características será considerada uma metodologia distinta, e será numerada sequencialmente.

Inicialmente, na metodologia 1 utilizaram-se as seguintes características para testar o classificador:

- Metodologia 01
- Banco de dados utilizado: 01 (42 dados sem falha e 252 dados com falha).
- Normalização: entre [-1 e +1]
- Hold out: 80% dos dados para treinamento e 20% para teste
- Seleção dos Hiperparâmetros: Validação Cruzada

onde os resultados desta primeira comparação são ilustrados no diagrama de caixas da figura 15.

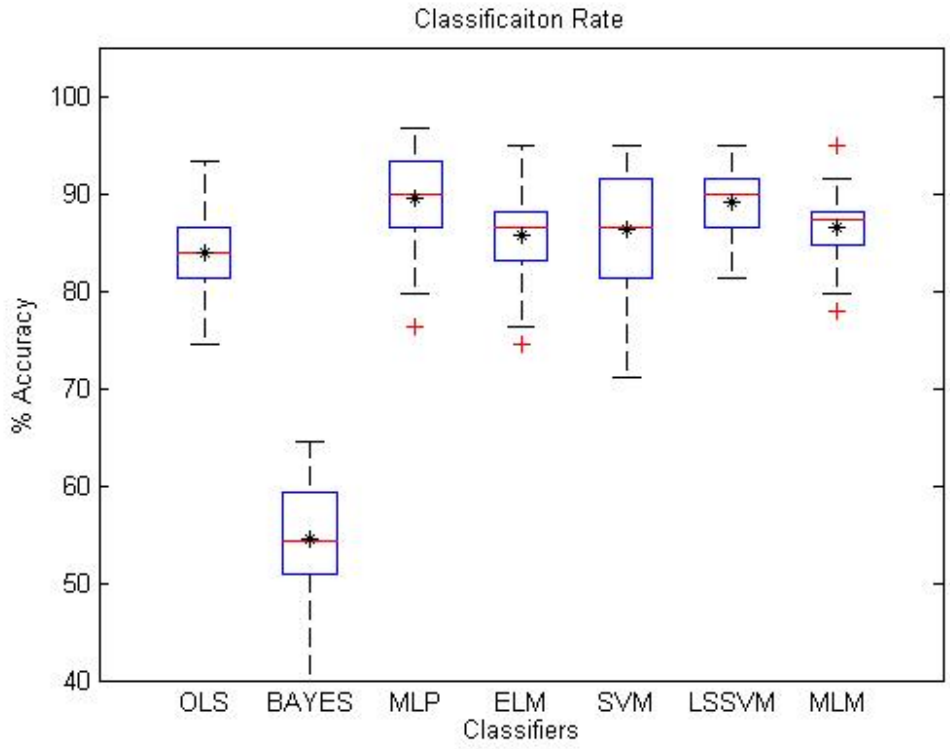

Figura 15 – Metodologia 01

A partir desta, pode-se verificar que, excluindo o classificador gaussiano quadrático (que obteve uma taxa de acerto média de 55%), a taxa de acerto média de todos os classificadores se situou entre 85% e 90%, com uma pequena vantagem do classificado MLP. Também pode-se notar que o classificador OLS, que é uma classificador linear e bem menos complexo que os demais, obteve um resultado similar aos outros algoritmos.

Este resultado poderia levar a conclusão de que, através da aplicação de um classificador linear, seria possível verificar a condição do motor com até mais de 90% de acerto.

Porém, ao analisar as matrizes de confusão de cada classificador, pode-se perceber que os dados de teste estavam sendo, quase em todas as situações, rotulados como falhas. Isto se deve ao grande desbalanceamento das classes, pois 86% do banco de dados é composto por amostra de falhas.

Se estes classificadores fossem aplicados em um ambiente real, ocorreriam vários falsos positivos (condição na qual um motor não tem falhas, e o classificador indica uma falha). Esta condição é indesejável, visto que a indicação de uma falha se converte em uma parada de processo, para que a equipe de manutenção possa atuar. Isto se refletiria em perda de tempo e capital para determinada empresa.

Além da metodologia 01, tentou-se inicialmente, na metodologia 02 (cujas características são listadas a seguir), alterar a normalização dos dados. Porém, como pode ser observado na figura 16 e nas matrizes de confusão, os resultados da metodologia anterior se repetiram.

- Metodologia 02
- Banco de dados utilizado: 01 (42 dados sem falha e 252 dados com falha).
- Normalização: pela média e desvio padrão dos atributos
- Hold out: 80% dos dados para treinamento e 20% para teste
- Seleção dos Hiperparâmetros: Validação Cruzada

Assim, para mitigar a taxa de falsos positivos, modificou-se, na metodologia 03, o Hold Out dos dados. Nesta nova etapa, os dados foram equilibrados de modo que, durante o treinamento, os parâmetros dos classificadores fossem atualizados pela mesma quantidade de dados de cada classe. As características desta metodologia estão listados a seguir, e os resultados são ilustradas na figura 17.

- Metodologia 03
- Banco de dados utilizado: 01 (42 dados sem falha e 252 dados com falha).

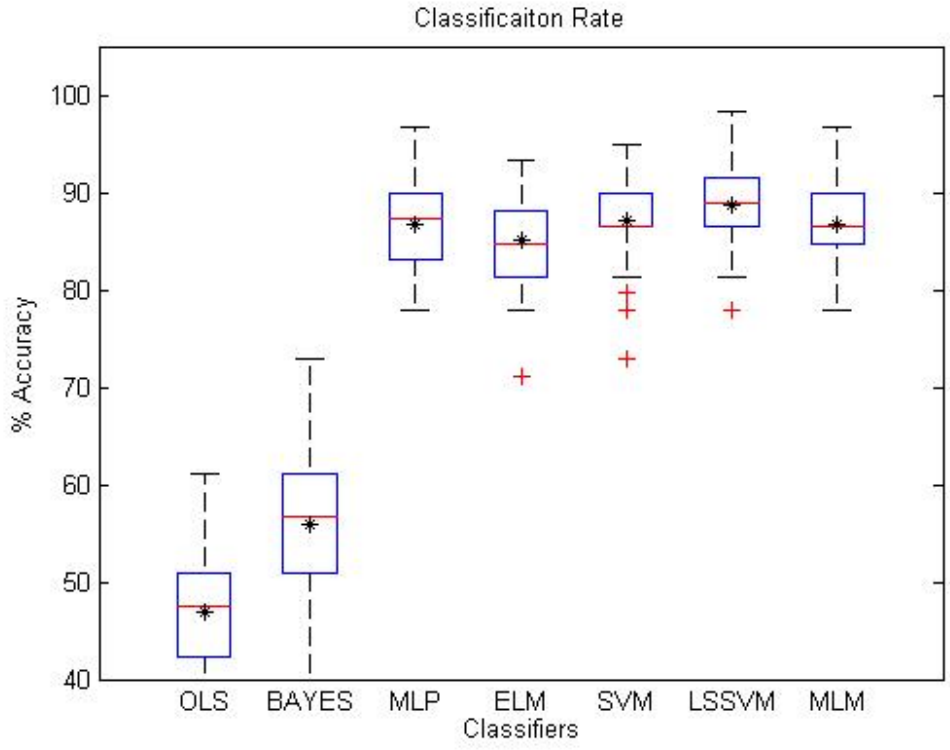

Figura 16 – Metodologia 02

- Normalização: pela média e desvio padrão dos atributos
- Hold out: Treinamento equilibrado
- Seleção dos Hiperparâmetros: Validação Cruzada

Como pode ser observado pelo Boxplot, a taxa média de acerto de todos os classificadores reduziu. Porém, analisando as matrizes de confusão, observou-se uma redução geral na taxa de falsos positivos (de mais de 95% para menos de 30%).

A taxa de falsos positivos reduziu bastante, porém um classificador com uma taxa de acerto média menor que 70% não se mostra satisfatório para a resolução deste problema. Este problema também se deu pelo desequilíbrio global dos dados, visto que, como haviam poucas amostras de motores sem falhas (42), apenas 33 amostras de cada classe (esta quantidade representando 80% da classe sem falhas) foram utilizadas para treino, e o restante (maioria de amostras com falhas) foi utilizada para teste.

Este fato aumentou bastante a taxa de falsos negativos, que levaria a um sistema de detecções de falhas com baixa taxa de acerto para o acontecimento de falhas.

Para solucionar o problema da pequena quantidade de amostras para treinamento, uma nova abordagem foi feita. Na metodologia 04, cujas características estão descritas a seguir, utilizou-se o banco de dados 02. Este banco de dados possui várias amostras,

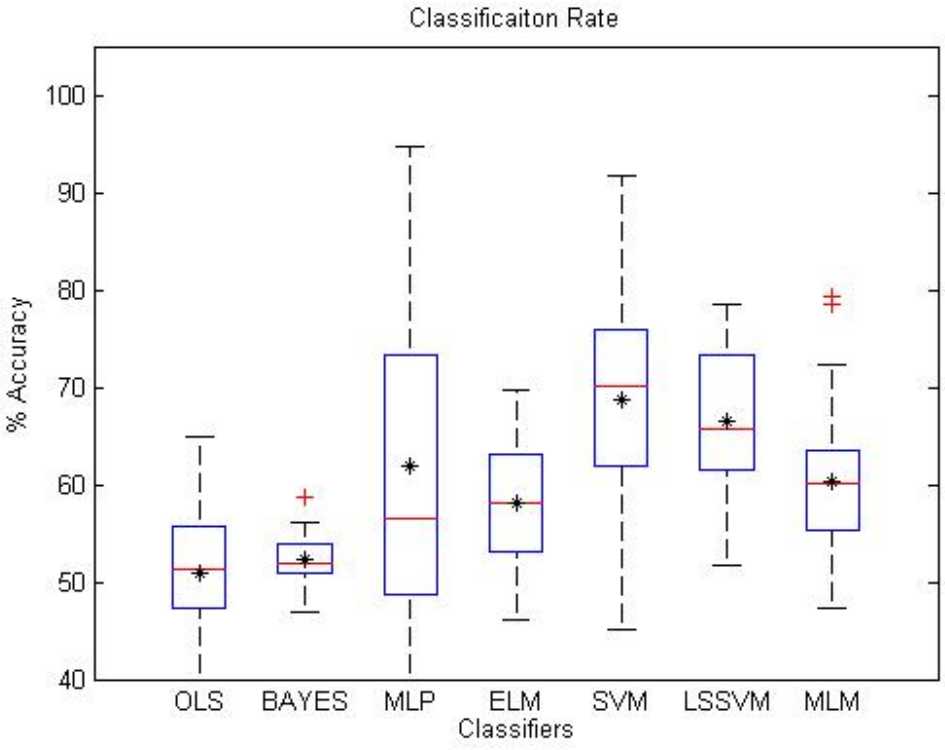

Figura 17 – Metodologia 03

geradas artificialmente, de motores sem falhas, o que faz com que este possua equilíbrio de amostras e uma maior quantidade de dados.

- Metodologia 04
- Banco de dados utilizado: 02 (252 dados sem falha e 252 dados com falha).
- Normalização: pela média e desvio padrão dos atributos
- Hold out: 80% dos dados para treinamento e 20% para teste
- Seleção dos Hiperparâmetros: Validação Cruzada

Com isso, pode-se aplicar diretamente o método de hold out considerando apenas 20% dos dados para teste, sem a preocupação de esta divisão gerar uma alta taxa de falsos positivos.

como pode ser visto na figura 18, os classificadores SVM e LSSVM conseguiram taxas médias de acerto próximas a 100% e o classificador baseado na rede MLP atingiu taxas médias de acerto de aproximadamente 90%. Juntamente a isso, houve uma redução na taxa de falsos positivos, uma situação desejada.

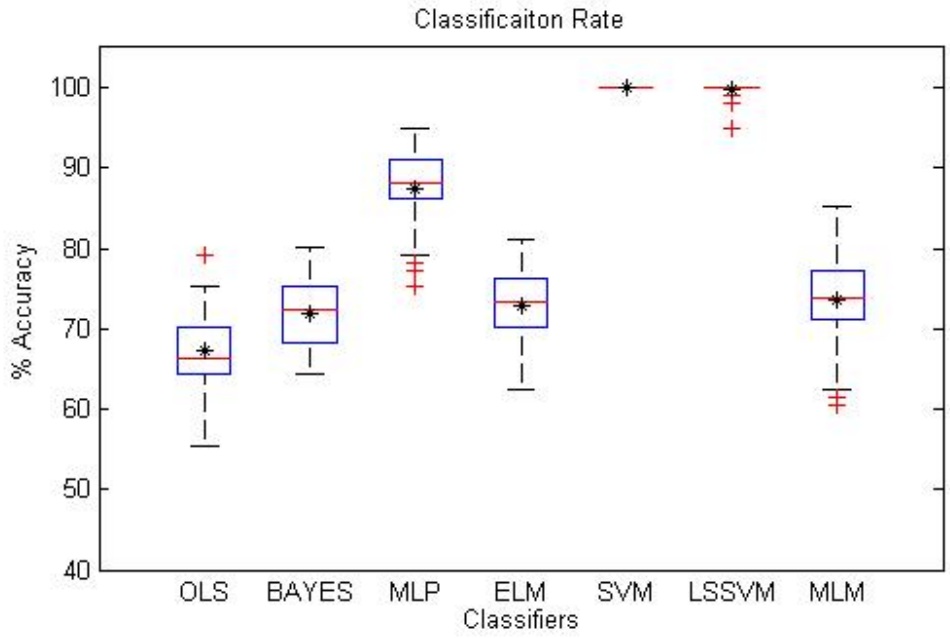

Figura 18 – Metodologia 04

Uma justificativa para esta diferença de resultados de todos os algoritmos aos classificadores SVM e LSSVM se dá pela quantidade de informação (parâmetros) guardados por estes classificadores.

Durante o treinamento, o classificador LSSVM acumula todos os dados desta etapa, o que, no caso, equivale a (403 dados). Também, verificou-se a quantidade de vetores suporte que o classificador SVM estava acumulando, e este valor chegou, em média, a 80% do banco (303 dados).

Na seção a seguir, a análise da seleção dos hiperparâmetros é evidenciada.

5.3 Seleção dos Hiperparâmetros

Como pode-se notar, na metodologia de comparação entre classificadores, a seleção dos hiperparâmetros ocorreu por meio de validação cruzada, ou seja, estes eram definidos e podiam ter valores variáveis a cada realização.

Com isso, após cada realização, guardava-se o valor de todos os hiperparâmetros selecionados para verificar se estes valores seguiam determinado padrão.

Na metodologia 1, na qual o banco de dados desequilibrado e o Hold Out do tipo 80% e 20% foram utilizados, os seguintes valores de hiperparâmetros foram atingidos:

- MLP: q = 2
- ELM: q = 26

- SVM: $C = 0{,}5$ ou 10; $\sigma^2 = 0{,}01$ ou 1
- LSSVM: $\gamma = 2$; $\sigma^2 = 0{,}5$
- MLM: $K = 10$;

Já na metodologia 3, na qual o banco de dados desequilibrado e o Hold Out equilibrando os dados durante o treinamento foram utilizados, os seguintes valores de hiperparâmetros foram atingidos:

- MLP: q = 9
- ELM: q = 26
- SVM: $C = 15$; $\sigma^2 = 5$
- LSSVM: $\gamma = 4$; $\sigma^2 = 0{,}5$
- MLM: $K = 9$;

Por fim, na metodologia 4, na qual o banco de dados equilibrado e o Hold Out do tipo 80% e 20% foram utilizados, os seguintes valores de hiperparâmetros foram atingidos:

- MLP: q = 12
- ELM: q = 29
- SVM: $C = 5$; $\sigma^2 = 0{,}01$
- LSSVM: $\gamma = 0{,}25$ ou 0,50 2; $\sigma^2 = 128$
- MLM: $K = 10$;

Os resultados da seção a seguir, se referem a tentativa de melhorar a taxa de acerto dos classificadores OLS, BAYES, MLP, ELM, e MLM, visto que estes tiveram desempenho inferior aos classificadores SVM e LSSVM e, em suas formulações, não precisam de modificações para se adequar a problemas multiclasses.

5.4 Resultados com Opção de Rejeição

5.5 Resultados Adicionais

Capítulo 6

Conclusão

6.1 Objetivo Geral

Neste trabalho, buscou-se detectar falhas incipientes por curto-circuito entre espiras do motor de indução trifásico através de técnicas de aprendizado de máquina, e análise da assinatura de corrente do motor.

A partir do banco de dados reais gerado, através dos diagramas de caixa e gráficos de dispersão, pôde-se perceber, primeiramente, que na faixa de valores onde há variação dos atributos das amostras de motores em funcionamento normal, também há amostras de motores com falhas, fazendo com que seja difícil a separação destas duas classes por níveis.

Também, percebeu-se, pela baixa taxa de classificação dos classificadores lineares e gaussianos, que a divisão entre as classes deste problema não é linear e que a distribuição de probabilidade intraclasses não se mostrou gaussiana.

A partir da aplicação de algoritmos não-lineares, verificou-se que o desbalanceamento entre amostras de motores com falhas e em funcionamento normal prejudicou a taxa de acerto dos classificadores e, também, fez com que ocorresse uma alta taxa de falsos positivos (que é a indicação de uma falha, quando na verdade não há).

Para resolver o problema supracitado, a probabilidade a priori das classes foi equilibrada com a geração artificial de amostras de motores em funcionamento normal.

Com isto, uma taxa média de acerto de 97% com o classificador SVM e de 99% com o classificador LSSVM foi atingida, mostrando que a maior dificuldade para separar os dados estava tanto no desbalanceamento entre classes destes, como na pequena quantidade de amostras.

Porém, para obter as elevadas taxas de acerto, estes classificadores se utilizavam de muitos parâmetros em comparação com os demais classificadores.

Como os resultados dos demais classificadores ainda não se mostravam satisfatórios, algumas técnicas no treinamento, juntamente com opção de rejeição foram utilizadas.

Estas técnicas elevaram as taxas de acerto dos classificadores, como por exemplo...

Por fim, em um futuro próximo, novos métodos de extração de características serão testados juntamente com novos e os já utilizados classificadores, de modo a escolher o

melhor método (extrator, classificador) para ser embarcado em um processador.

6.2 Objetivos Específicos

Todos os objetivos específicos foram atingidos, visto que o banco de dados foi analisado estatisticamente; cinco paradigmas de classificadores foram implementados (incluindo sete algoritmos distintos); a eficácia de cada classificador foi estudada com o auxílio de ferramentas estatísticas, tais como gráfico de caixas e matriz de confusão; a estratégia de opção de rejeição foi utilizada e, com esta, pode-se notar uma diminuição na taxa de erros dos classificadores; e, por fim, o teste de Mc Nemar foi aplicado para comparar os classificadores quanto à capacidade de detectar falhas.

6.3 Trabalhos Futuros

Mesmo com os diversos estudos dessa dissertação, o problema de detecção de falhas ainda está longe de ser esgotado. Algumas sugestões para estudos futuros são listadas a seguir.

Primeiramente, o único método utilizado para a extração de atributos foi a FFT, a partir da qual algumas harmônicas da tensão de alimentação do motor foram utilizadas como atributos do problema. Uma nova análise seria aplicar outras técnicas para extração de atributos, tais como estatísticas de alta ordem.

Além disso, existem diversas falhas em motores que não foram estudadas, tais como quebra de barras e falhas em rolamentos. Pode-se verificar as similaridades e diferenças dos métodos de extração de atributos e detecção destas falhas.

Em relação ao banco de dados já gerado, percebeu-se que as classes de falha e normal estavam muito desequilibradas. Com isso, foi necessário gerar dados artificialmente. Neste trabalho, apenas um método foi aplicado para esta geração, porém outros métodos, tais como os baseados em clusters, podem ser aplicados.

Em relação aos classificadores SVM e LSSVM, pode-se estudar os métodos de poda destes e os meios para utilizá-los em classificação multi-classes.

Em relação à técnicas de classificação, como as classes de falha representam valores crescentes de falhas, será verificada a utilização da classificação ordinal na detecção de falhas em motores.

Apêndice A

Motor de Indução Trifásico

Teste

Apêndice B

Fast Fourier Transform

Teste

Referências

ASFANI, D. et al. Temporary short circuit detection in induction motor winding using combination of wavelet transform and neural network. *Expert Systems with Applications*, Elsevier, v. 39, n. 5, p. 5367–5375, 2012. Cited on page 16.

AVELAR, V. S.; BACCARINI, L. M. R.; AMARAL, G. F. V. Desenvolvimento de um sistema inteligente para diagnostico de falhas nos enrolamentos do estator de motores de indução. *X SABI–Simpósio Brasileiro de Automação Inteligente. São João Del Rei–MG–Brasil*, 2011. Cited on page 16.

BACHA, K. et al. Induction machine fault detection using stray flux emf measurement and neural network-based decision. *Electric Power Systems Research*, Elsevier, v. 78, n. 7, p. 1247–1255, 2008. Cited on page 15.

BONNETT, A. H. Root cause failure analysis for ac induction motors in the petroleum and chemical industry. In: *2010 Record of Conference Papers Industry Applications Society 57th Annual Petroleum and Chemical Industry Conference (PCIC)*. [S.l.: s.n.], 2010. Cited on page 16.

BOSLAUGH, S.; WATTERS, P. *Statistics in a Nutshell: A Desktop Quick Reference (In a Nutshell (O'Reilly))*. [S.l.]: O'Reilly Media, Inc., illustrated edition ed, 2008. Cited on page 61.

BOSTANCI, B.; BOSTANCI, E. An evaluation of classification algorithms using mc nemar's test. In: SPRINGER. *Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2012)*. [S.l.], 2013. p. 15–26. Cited on page 61.

BURGES, C. J. C. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, v. 2, n. 2, p. 121–167, 1998. Cited on page 43.

DENG, W.; ZHENG, Q.; CHEN, L. Regularized extreme learning machine. In: *Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining (CIDM'09)*. [S.l.: s.n.], 2009. p. 389–395. Cited on page 41.

GHATE, V. N.; DUDUL, S. V. Optimal mlp neural network classifier for fault detection of three phase induction motor. *Expert Systems with Applications*, Elsevier, v. 37, n. 4, p. 3468–3481, 2010. Cited 2 times on pages 16 e 17.

HAYKIN, S. S. Redes neurais artificiais: principios e praticas. *2a Edicao, Bookman, Sao Paulo, Brasil*, 2000. Cited on page 36.

HORATA, P.; CHIEWCHANWATTANA, S.; SUNAT, K. Robust extreme learning machine. *Neurocomputing*, v. 102, p. 31–44, 2012. Cited on page 41.

HORNIK, K.; STINCHCOMBE, M.; WHITE, H. Multilayer feedforward networks are universal approximators. *Neural networks*, Elsevier, v. 2, n. 5, p. 359–366, 1989. Cited on page 37.

HUANG, G.-B.; WANG, D. H.; LAN, Y. Extreme learning machines: a survey. *International Journal of Machine Learning and Cybernetics*, v. 2, p. 107–122, 2011. Cited on page 41.

HUANG, G. B.; ZHU, Q. Y.; ZIEW, C. K. Extreme learning machine: Theory and applications. *Neurocomputing*, v. 70, n. 1–3, p. 489–501, 2006. Cited on page 40.

JUNIOR, C. D. Sistema de simulação de cargas mecânicas para motor de indução acionado por inversor de frequência. Curitiba, 2013. Cited on page 15.

LIU, N.; WANG, H. Ensemble based extreme learning machine. *IEEE Signal Processing Letters*, v. 17, n. 8, p. 754–757, 2010. Cited on page 41.

MARTINS, J. F.; PIRES, V. F.; PIRES, A. Unsupervised neural-network-based algorithm for an on-line diagnosis of three-phase induction motor stator fault. *Industrial Electronics, IEEE Transactions on*, IEEE, v. 54, n. 1, p. 259–264, 2007. Cited 2 times on pages 16 e 17.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, Springer, v. 5, n. 4, p. 115–133, 1943. Cited on page 36.

MICHE, Y. et al. OP-ELM: Optimally pruned extreme learning machine. *IEEE Transactions on Neural Networks*, v. 21, n. 1, p. 158–162, 2010. Cited on page 41.

MICHE, Y. et al. TROP-ELM: a double-regularized ELM using LARS and Tikhonov regularization. *Neurocomputing*, v. 74, n. 16, p. 2413–2421, 2011. Cited on page 41.

MOHAMMED, A. et al. Human face recognition based on multidimensional PCA and extreme learning machine. *Pattern Recognition*, v. 44, n. 10–11, p. 2588–2597, 2011. Cited on page 41.

NANDI, S.; TOLIYAT, H. A.; LI, X. Condition monitoring and fault diagnosis of electrical motors-a review. *Energy Conversion, IEEE Transactions on*, IEEE, v. 20, n. 4, p. 719–729, 2005. Cited on page 16.

NEUMANN, K.; STEIL, J. Optimizing extreme learning machines via ridge regression and batch intrinsic plasticity. *Neurocomputing*, v. 102, p. 23–30, 2013. Cited on page 41.

NIRALI, R.; SHAH, S. Fuzzy decsion based soft multi agent controller for speed control of three phase induction motor. *Transformation*, v. 2, p. 3Φ, 2011. Cited on page 15.

OLIVEIRA, A. G. de; SA, C. M. de. Stator winding interturns short circuit fault detection in a three phase induction motor driven by frequency converter using neural networks. In: *Energy Efficiency in Motor Driven Systems 2013 Conference, EEMODS*. [S.l.: s.n.], 2013. Cited 4 times on pages 16, 20, 22 e 26.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, American Psychological Association, v. 65, n. 6, p. 386, 1958. Cited on page 37.

SAWA, T.; KUME, T. Motor drive technology-history and visions for the future. In: IEEE. *Power Electronics Specialists Conference, 2004. PESC 04. 2004 IEEE 35th Annual*. [S.l.], 2004. v. 1, p. 2–9. Cited on page 15.

SESHADRINATH, J.; SINGH, B.; PANIGRAHI, B. K. Incipient interturn fault diagnosis in induction machines using an analytic wavelet-based optimized bayesian inference. *Neural Networks and Learning Systems, IEEE Transactions on*, IEEE, v. 25, n. 5, p. 990–1001, 2014. Cited 3 times on pages 15, 16 e 20.

THOMSON, W. T.; FENGER, M. Current signature analysis to detect induction motor faults. *Industry Applications Magazine, IEEE*, Ieee, v. 7, n. 4, p. 26–34, 2001. Cited 3 times on pages 15, 16 e 21.

VAPNIK, V. N. *Statistical learning theory*. [S.l.]: Wiley New York, 1998. v. 1. Cited on page 43.

VAPNIK, V. N. *The nature of statistical learning theory*. [S.l.]: Springer Science & Business Media, 2000. Cited on page 43.

VENKADESAN, A.; HIMAVATHI, S.; MUTHURAMALINGAM, A. Performance comparison of neural architectures for on-line flux estimation in sensor-less vector-controlled im drives. *Neural Computing and Applications*, Springer, v. 22, n. 7-8, p. 1735–1744, 2013. Cited on page 15.

VICO, J.; HUNT, R. Protection principles for electrical motors in the cement industry. In: IEEE. *Cement Industry Technical Conference, 2010 IEEE-IAS/PCA 52nd*. [S.l.], 2010. p. 1–13. Cited on page 16.

ZONG, W.; HUANG, G.-B. Face recognition based on extreme learning machine. *Neurocomputing*, v. 74, n. 16, p. 2541–2551, 2011. Cited on page 41.