PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE TELEINFORMÁTICA
DEPARTAMENTO DE ENGENHARIA DE TELEINFORMÁTICA
UNIVERSIDADE FEDERAL DO CEARÁ

TESE

REVISITANDO O PROBLEMA DE CLASSIFICAÇÃO DE PADRÕES NA PRESENÇA DE *OUTLIERS* USANDO TÉCNICAS DE REGRESSÃO ROBUSTA

Autor: ANA LUIZA BESSA DE PAULA BARROS
Orientador: Dr. Guilherme de Alencar Barreto

FORTALEZA
2013

ANA LUIZA BESSA DE PAULA BARROS

REVISITANDO O PROBLEMA DE CLASSIFICAÇÃO DE PADRÕES NA PRESENÇA DE *OUTLIERS* USANDO TÉCNICAS DE REGRESSÃO ROBUSTA

Tese submetida à Coordenação do Curso de Pós-Graduação em Engenharia de Teleinformática, da Universidade Federal do Ceará, como parte dos requisitos exigidos para obtenção do grau de Doutor em Engenharia de Teleinformática.

Orientador: Prof. Dr. Guilherme de Alencar Barreto

FORTALEZA
2013

Dedico este trabalho a todos
que estiveram junto comigo nesta caminhada.

Agradecimentos

Ao Barros e à Laura, que deram início a tudo isso.

Ao Barros Neto e à Ana Carolina pelo apoio.

Ao Bernardo, meu dileto assistente, que está sempre me despertando para novas questões.

Ao Prof. Guilherme de Alencar Barreto, pelo convívio, conhecimento e orientação, e pela oportunidade de discussões que enriqueceram este trabalho.

Aos amigos do laboratório, pela convivência durante todo esse tempo e pelo ótimo ambiente de trabalho.

Aos professores e funcionários do Departamento de Engenharia de Teleinformática que, de forma direta ou indireta, participaram do desenvolvimento deste trabalho.

À UECE, pelo apoio e pela oportunidade de realizar este trabalho.

Aos professores do curso de computação da UECE. Em especial, ao Gustavo, pelas longas conversas sobre redes neurais, agradeço o despertar para esta área e a amizade durante todo este tempo. Ao Celestino, pela amizade e pela torcida durante todo este processo. Ao Thelmo, pelo apoio e pelas discussões filosóficas e matemáticas. Ao Valdísio, pela amizade e por estar sempre pronto a ajudar.

Ao Fabiano, pela dedicação, presteza e disponibilidade.

Aos amigos da Rua Alegre, pela ajuda no desenvolvimento do pensar.

A todos os amigos, pelo apoio, amizade e paciência.

“O que vale na vida não é o ponto de partida e sim a caminhada.
Caminhando e semeando, no fim terás o que colher.
(Cora Coralina)

Resumo

Nesta tese, aborda-se o problema de classificação de dados que estão contaminados com padrões atípicos. Tais padrões, genericamente chamados de *outliers*, são onipresentes em conjunto de dados multivariados reais, porém sua detecção *a priori* (i.e antes de treinar um classificador) é uma tarefa de difícil realização. Como conseqüência, uma abordagem reativa, em que se desconfia da presença de *outliers* somente após um classificador previamente treinado apresentar baixo desempenho, é a mais comum. Várias estratégias podem então ser levadas a cabo a fim de melhorar o desempenho do classificador, dentre elas escolher um classificador mais poderoso computacionalmente ou promover uma limpeza dos dados, eliminando aqueles padrões difíceis de categorizar corretamente. Qualquer que seja a estratégia adotada, a presença de *outliers* sempre irá requerer maior atenção e cuidado durante o projeto de um classificador de padrões. Tendo estas dificuldades em mente, nesta tese são revisitados conceitos e técnicas provenientes da teoria de regressão robusta, em particular aqueles relacionados à estimação M, adaptando-os ao projeto de classificadores de padrões capazes de lidar automaticamente com *outliers*. Esta adaptação leva à proposição de versões robustas de dois classificadores de padrões amplamente utilizados na literatura, a saber, o classificador linear dos mínimos quadrados (*least squares classifier*, LSC) e a máquina de aprendizado extremo (*extreme learning machine*, ELM). Através de uma ampla gama de experimentos computacionais, usando dados sintéticos e reais, mostra-se que as versões robustas dos classificadores supracitados apresentam desempenho consistentemente superior aos das versões originais.

Palavras-chave: Classificação de padrões, *outliers*, mínimos quadrados ordinário, regressão robusta, estimação-*M*, máquina de aprendizado extremo.

Abstract

This thesis addresses the problem of data classification when they are contaminated with atypical patterns. These patterns, generally called *outliers*, are omnipresent in real-world multivariate data sets, but their *a priori* detection (i.e. before training the classifier) is a difficult task to perform. As a result, the most common approach is the reactive one, in which one suspects of the presence of outliers in the data only after a previously trained classifier has achieved a low performance. Several strategies can then be carried out to improve the performance of the classifier, such as to choose a more computationally powerful classifier and/or to remove the detected *outliers* from data, eliminating those patterns which are difficult to categorize properly. Whatever the strategy adopted, the presence of outliers will always require more attention and care during the design of a pattern classifier. Bearing these difficulties in mind, this thesis revisits concepts and techniques from the theory of robust regression, in particular those related to M-estimation, adapting them to the design of pattern classifiers which are able to automatically handle outliers. This adaptation leads to the proposal of robust versions of two pattern classifiers widely used in the literature, namely, least squares classifier (LSC) and extreme learning machine (ELM). Through a comprehensive set of computer experiments using synthetic and real-world data, it is shown that the proposed robust classifiers consistently outperform their original versions.

Key words: Pattern classifiers, outliers, ordinary least squares, robust regression, M-estimation, extreme learning machine.

Lista de Figuras

Lista de Tabelas

Lista de Siglas

BIP	*Batch Intrinsic Plasticity*
ELM	*Extreme Learning Machine*
ELM-BIP	*Extreme Learning Machine-Batch Intrinsic Plasticity*
IRLS	*Iteratively Reweighted Least-Squares*
LAM	*Linear Associative Memory*
LSC	*Least-Squares Classifier*
MAR	Mediana dos valores Absolutos dos Resíduos
MIMO	*Multiple-Input and Multiple-Output*
MLP	*MultiLayer Perceptron*
MQ	Mínimos Quadrados
MQO	Mínimos Quadrados Ordinário
NMF	*Non-negative Matrix Factorization*
OLAM	*Optimal Linear Associative Memory*
PSO	*Particle Swarm Optimization*
RBF	*Radial Basis Function*
ROB-ELM	*Robust Extreme Learning Machine*
ROB-ELM-BIP	*Robust Extreme Learning Machine-Batch Intrinsic Plasticity*
ROLAM	*Robust OLAM Classifier*
SLFN	*Single-hidden Layer Feedforward Network*
SVD	*Singular Value Decomposition*
WDBC	*Wisconsin Diagnostic Breast Cancer*

Lista de Símbolos

f	função para variável contínua
$\hat{f}$	aproximação da função para variável contínua
$\mathbb{R}$	conjunto dos números reais
x	variável escalar, i.e., $x \in \mathbb{R}$
$\mathbf{x}$	vetor de variáveis
$\bar{\mathbf{x}}$	vetor médio
$\mathbf{X}$	matriz de regressão
y	variável resposta
$\mathbf{y}$	vetor de variáveis resposta
$\hat{y}$	aproximação da variável resposta
$e_{i\mu}$	contribuição de cada erro para a função objetivo
$\mathbf{e}$	vetor de erros de predição
ψ	derivada da função ρ
$w(e_i)$	função de ponderação do erro i
$\mathbf{W}$	matriz de pesos
$\mathbf{d}$	vetor de valores desejados (rótulos da classe alvo)
$\mathbf{D}$	matriz de valores desejados (rótulos da classe alvo)
$\mathbf{I}$	matriz identidade
$E[.]$	operador valor esperado
J	função custo
λ	constante de regularização
N	quantidade total de pares de dados
N_1	quantidade de pares de dados usados para treinar o modelo
N_2	quantidade de pares de dados usados para validar o modelo
N_{out}	número de *outliers*
μ_k	vetor médio da k-ésima classe
$\mathbf{C}_k$	matriz de covariância da k-ésima classe
L	valor crítico da distribuição Chi-quadrado
β_i	vetor peso conectando os neurônios ocultos ao i-ésimo neurônio de saída
k	limiar de um estimador-M

K	quantidade de classes
b	limiar (bias)
σ	desvio padrão
$\hat{\sigma}$	estimativa robusta do desvio padrão
σ_ε^2	variância do erro
h	valor de alavancagem
$\mathbf{h}$	vetor de saídas dos neurônios ocultos
$\mathbf{H}$	matriz de saídas dos neurônios ocultos
$\mathbf{m}$	vetor de pesos dos neurônios ocultos
$\mathbf{\Omega}$	aplicação de mapeamento
χ	conjunto de entradas
γ	conjunto de saídas
p	dimensão do vetor de entrada de uma rede neural
m	dimensão do vetor de saída de uma rede neural
ρ	função objetivo
ρ'	derivada da função ρ
ε	erro (ruído) aleatório
ε^2	erro médio quadrático
β	vetor de parâmetros
$\hat{\beta}$	estimativa do vetor de parâmetros
β_i	vetor de pesos do i-ésimo neurônio de saída
η	taxa de aprendizagem das redes neurais artificiais
$w_{i\mu}$	peso associado à ligação entre entrada j e neurônio μ
$d_{i\mu}$	valor alvo do i-ésimo neurônio de saída para o μ-ésimo vetor $\mathbf{h}_\mu$

Sumário

1 Introdução

A *teoria da estimação* é uma ramo da Estatística e do Processamento de Sinais que lida com o problema de determinar valores de parâmetros a partir de dados empíricos que possuem uma componente estocástica (SCHONHOFF; GIORDANO, 2006). Basicamente, os parâmetros a serem determinados ocorrem em dois tipos de situações: (1) parâmetros de uma função densidade de probabilidade ou função linear ou não-linear genérica (e.g. polinômio, hiperplano, etc.) que modela os dados a fim de agrupá-los ou categorizá-los; ou (2) coeficientes de uma função genérica, linear ou não-linear, que é ajustada aos dados para fazer previsões numéricas sobre uma variável específica, chamada variável resposta. A primeira situação trata de problemas de classificação de padrões, enquanto a segunda trata de problema de regressão.

Independentemente do tipo de problema a ser tratado, classificação ou regressão, os verdadeiros valores dos parâmetros do modelo probabilístico ou da função matemática geradora dos dados observados são desconhecidos. Um *estimador* é um método, técnica ou procedimento cujo o objetivo é prover valores aproximados para os parâmetros usando os dados observados.

A meta geral da teoria da estimação para um dado problema consiste, portanto, em desenvolver um estimador adequado ao modelo escolhido, que seja, de preferência, fácil de implementar em software e/ou hardware.

É também desejável que um estimador, ao tomar os dados medidos como entrada, produza uma estimativa dos parâmetros com a acurácia desejada; ou seja, os parâmetros estimados devem estar o mais próximo possível dos parâmetros verdadeiros. Em outras palavras, o estimador deve exibir *otimalidade*, que refere-se à capacidade do estimador em prover os melhores valores possíveis segundo um certo critério de avaliação.

Tais critérios usualmente envolvem alguma medida de erro entre o valor observado e o valor predito de uma variável de interesse (por exemplo, a variável resposta de um problema de regressão). Por exemplo, o famoso estimador dos mínimos quadrados ordinários (MQO), proposto inicialmente por Legendre (1805) e posteriormente, mas de maneira independente, por Gauss (1809), é ótimo pois provê estimativas que minimizam a soma dos quadrados dos

erros entre as saídas observadas e as saídas preditas por uma dada função de regressão.

Contudo, estimadores ótimos e, por extensão, estimativas ótimas para os parâmetros, nem sempre são possíveis de obter. Isto ocorre, por exemplo, quando as condições que garantem a otimalidade do estimador não são satisfeitas. Neste caso, o estimador deixa de ser ótimo. No caso de estimadores baseados no critério MQO, supõe-se que os erros estão normalmente distribuídos, são não-correlacionados e possuem média nula e variância constante.

Em muitas aplicações práticas, contudo, as condições de otimalidade do estimador MQO não são observadas, principalmente a de que os erros estão distribuídos segundo uma distribuição normal. Outra situação na qual a otimalidade do estimador MQO não está garantida é quando amostras discrepantes (*outliers*) estão presentes nos dados.

Segundo Barnett & Lewis (1994), um *outlier* é uma observação (ou um conjunto de observações) que aparenta ser inconsistente com o restante do conjunto de dados. Estes autores enfatizam que a frase "aparenta ser inconsistente" é crucial para o processo de caracterização de amostras discrepantes, uma vez que remete à idéia de que o modelo matemático ou probabilístico subjacente aos dados seria diferente caso estas amostras discrepantes não estivessem presentes.

Modelos diferentes geralmente implicam em valores diferentes para os parâmetros a serem estimados, pois os valores dos erros tendem a ser bem maiores para amostras discrepantes do que para amostras "normais". Isto posto, uma questão importante emerge aqui. Como proceder se há uma suspeita da presença de *outliers* nos dados?

Se for possível visualizar os dados, o que normalmente ocorre quando o modelo é unidimensional (seja na entrada, seja na saída), pode-se fazer uma inspeção visual e buscar diretamente pelas observações que produzem os maiores erros, eliminando-as em seguida[1]. Este procedimento de limpeza dos dados *a posteriori* foi em si discutido pelo próprio Legendre (1805) quando da proposição do critério MQO:

> *If among these errors are some which appear too large to be admissible, then those observations which produced these errors will be rejected, as coming from too faulty experiments, and the unknowns will be determined by means of the other observations, which will then give much smaller errors.*

Outra opção é utilizar estimadores que tenham capacidade de lidar de forma automática com *outliers*, diminuindo sua influência no resultado final da estimação de parâmetros do modelo.

[1] Para sistemas com múltiplas saídas pode-se tentar fazer uma inspeção de cada saída individualmente.

Esta classe de estimadores são chamados de *estimadores robustos* e compõem uma área da Estatística conhecida como *Estatística Robusta* (RONCHETTI, 2006; MARONNA et al., 2006) ou *Regressão Robusta* (ROUSSEEUW; LEROY, 1987), no contexto de problemas de regressão.

Contudo, não há concordância geral sobre qual procedimento adotar quando se suspeita que *outliers* estão presentes nos dados, pois alguns autores argumentam que pode haver informação importante nos *outliers*, descartada a opção de que eles tenham sido gerados por erros grosseiros de medição. Pode-se também optar por uma abordagem híbrida no pior caso, em que mesmo usando estimadores robustos, resultam erros que se destacam dos demais, justificando aí uma limpeza dos dados.

Qualquer que seja a opção escolhida para lidar com *outliers*, estes são onipresentes em dados estatísticos, conforme observa Barnett & Lewis (1994). Reside aí, portanto, o interesse desta tese na análise de dados contendo *outliers*. Em particular, nesta tese aborda-se o problema de classificação de padrões com *outliers*. Neste sentido, são apresentadas a seguir as motivações por trás das propostas a serem apresentadas nesta tese.

1.0.1 Motivação

A pesquisa em Estatística tem um longo histórico de contribuições relevantes no contexto de análise de regressão na presença de *outliers* nos dados (ANDREWS, 1974; HILL; HOLLAND, 1977; BRANHAM, 1982; STEVENS, 1984; BAI; WU, 1997; CHATTERJEE; MÄCHLER, 1997). Uma destas contribuições é conhecida como *Estimação-M* (BAI; WU, 1997), cujo principal objetivo é desenvolver estimadores de parâmetros que não sejam tão sensíveis à presença de *outliers* como o é o estimador MQO. Na verdade, o estimador MQO é um caso particular de estimador-*M*, conforme será discutido mais adiante nesta tese.

Ao contrário do que ocorre com a área de regressão robusta, em que há diversas contribuições ao problema de regressão na presença de *outliers*, o mesmo não se verifica nas áreas de Aprendizado de Máquinas e Reconhecimento de Padrões, embora este seja um tópico que eventualmente é abordado na literatura especializada (RITTER; GALLEGOS, 1997; STANIMIROVA; WALCZAK, 2008; KIM; GHAHRAMANI, 2008; NETO; BARRETO, 2009; GLAVIN; MADDEN, 2010; SMITH; MARTINEZ, 2011; FAWZYA et al., 2013).

Assim como em problemas de regressão, a abordagem mais comum em Classificação de Padrões envolve justamente realizar uma limpeza *a posteriori* dos dados a fim de eliminar *outliers*. Mais especificamente, treina-se primeiro um ou mais classificadores sobre um conjunto de dados e, baseando-se nos desempenhos dos classificadores treinados, tenta-se entender por

que os resultados foram aquém do esperado. Uma hipótese a ser testada é a de que a causa dos maus resultados é a presença de *outliers*. Tal hipótese pode ser comprovada se os desempenhos dos classificadores melhorar caso os supostos *outliers* sejam removidos do conjunto original.

Contudo, apesar dos trabalhos supracitados envolvendo classificação supervisionada na presença de *outliers* e da importância do tema para as áreas de Aprendizado de Máquinas e Reconhecimento de Padrões, não existe até o presente momento um arcabouço geral para o projeto de classificadores robustos, tal como a estimação-*M* o é para a teoria de regressão robusta.

Esta observação, confirmada através da realização de extensa revisão bibliográfica, serviu de motivação principal para o desenvolvimento das idéias a serem discutidas nesta tese. Diante deste fato, o objetivo geral da tese, bem como seus objetivos mais específicos, são apresentados a seguir.

1.1 Objetivos Geral e Específicos

O problema pricipal tratado nesta tese é o problema de classificação de padrões na presença de *outliers*. Como conseqüência, o objetivo maior a ser perseguido envolve a proposição de uma nova abordagem ao problema de classificação de padrões, abordagem esta baseada em conceitos oriundos da teoria de regressão robusta, em particular, no arcabouço da estimação-*M*.

Em outras palavras, deseja-se utilizar técnicas da área de regressão robusta em outra área, classificação de padrões, a fim de projetar classificadores de padrões robustos a *outliers*. A principal hipótese de trabalho é a de que o uso de estimadores-*M* é um arcabouço adequado para o projeto de tais classificadores robustos.

Com o intuito de testar esta hipótese, serão propostas extensões robustas de classificadores de padrões lineares e não-lineares. Em particular, serão utilizados classificadores que originalmente adotam o critério MQO para estimação de parâmetros, tais como o classificador LS (*least-squares*) (DUDA et al., 2006; WEBB, 2002) e a rede ELM (*Extreme Learning Machine*) (HUANG et al., 2006).

Em face do objetivo geral e da hipótese de trabalho supracitados, vários objetivos específicos serão trabalhados ao longo do desenvolvimento da tese, de tal modo a permitir que o objetivo geral seja alcançado. Tais objetivos específicos são listados a seguir.

1. Desenvolver e avaliar uma extensão robusta do classificador LS usando estimadores-*M*.
2. Desenvolver e avaliar uma extensão robusta do classificador ELM usando estimadores-*M*.

3. Desenvolver e avaliar uma extensão robusta do classificador ELM treinado com o algoritmo BIP (*batch intrinsic plasticity*) (NEUMANN; STEIL, 2013) usando estimadores-*M*.

4. Otimizar os parâmetros da versão robusta do classificador ELM usando otimização por enxame de partículas (*particle swarm optimization*) (KENNEDY; EBERHART, 1995).

1.2 Produção Científica

A seguir é fornecida uma lista com os trabalhos publicados em decorrência do desenvolvimento das propostas a serem apresentadas nesta tese.

- **BARROS, A. L. B. P.** & BARRETO, G. A. - Extreme Learning Machine Robusta para Reconhecimento de Faces. In: Brazilian Conference on Intelligent Systems, Curitiba-PR, *Proceedings of the 1st Brazilian Conference on Intelligent Systems* (BRACIS/ENIA'2012), p. 1-13, 2012.

- **BARROS, A. L. B. P.** & BARRETO, G. A. - Improving the Classification Performance of Optimal Linear Associative Memory in the Presence of Outliers, *Proceedings of the International Work Conference on Artificial Neural Networks 2013* (IWANN'2013), Puerto de la Cruz, Tenerife, Espanha, LNCS 7902, pp. 622-632, 2013.

- **BARROS, A. L. B. P.** & BARRETO, G. A. - Building a Robust Extreme Learning Machine for Classification in the Presence of Outliers, *Proceedings of the International Conference on Hybrid Artificial Intelligence Systems 2013* (HAIS'2013), Salamanca, Espanha.

- **BARROS, A. L. B. P.** & BARRETO, G. A. - Máquina de Aprendizado Extremo Robusta para Classificação com Outliers. In: Brazilian Congress on Computational Intelligence, aceito para publicação nos *Anais do 11. Congresso Brasileiro de Inteligência Computacional* (CBIC'2013), Porto de Galinhas, Pernambuco, Brasil.

- **BARROS, A. L. B. P.** & BARRETO, G. A. - A Robust Extreme Learning Machine for Face Recognition. Artigo selecionado entre os melhores do evento BRACIS/ENIA'2012 para submissão de versão estendida para o periódico *International Journal of Natural Computing Research* (IJNCR).

- **BARROS, A. L. B. P.** & BARRETO, G. A. - On the Design of Robust Linear and Nonlinear Classifiers Based on *M*-Estimation. Artigo submetido ao periódico *International Journal of Pattern Recognition and Artificial Intelligence* (IJPRAI).

1.3 Estrutura da Tese

1.3.1 Metodologia de Organização

Essa tese é subdividida em capítulos, os quais são organizados de forma a serem o mais auto-contidos possível em termos de conteúdo. Esta organização permite discutir mais especificamente a fundamentação teórica sobre análise de regressão linear, tanto padrão quanto robusta, além de discutir as propostas de classificadores robustos lineares, baseados no classificador OLAM, e não-lineares, baseados no classificador ELM e extensões. Ao final de cada capítulo, é apresentado um breve resumo relacionado ao assunto abordado.

A forma como a tese está organizada permite que o leitor interessado em uma determinada estratégia se restrinja a um determinado capítulo de interesse. Os detalhes contidos em cada capítulo são descritos de forma resumida na subseção a seguir.

1.3.2 Organização Geral do Restante do Projeto

O restante deste trabalho está organizado em sete capítulos que descrevem os diversos conceitos relacionados à análise de regressão robusta. Um breve comentário sobre cada um deles é feito a seguir.

No Capítulo 2 são apresentados os fundamentos do problema de regressão linear, simples e múltipla. Neste capítulo são apresentados também os fundamentos da teoria de estimação de parâmetros, um ponto crucial em regressão linear, sendo discutido em detalhes o estimador dos mínimos quadrados ordinário (MQO). A implementação em Matlab do estimador MQO e um exemplo numérico são apresentados ao final do capítulo.

No Capítulo 3 tem-se por objetivo inicial apresentar o conceito de *outlier* e discutir a influência desses pontos em problemas de análise de regressão. Em seguida, é discutida a fundamentação teórica sobre regressão robusta e estimação-M, que é apresentada como um arcabouço teórico geral que produz estimadores alternativos ao método de mínimos quadrados ordinários para problemas de regressão linear. A implementação em Matlab de métodos de estimação robusta e exemplos numéricos são apresentados ao longo do capítulo.

O Capítulo 4 apresenta a proposta de um classificador linear robusto, o qual tem como base o modelo de memória associativa linear ótima (*Optimal Linear Associative Memory*, OLAM). O Capítulo discute os fundamentos do modelo OLAM e mostra que sua versão padrão utiliza o método de mínimos quadrados para estimar a matriz de pesos. Em seguida, discute-se o

desenvolvimento de um classificador robusto baseado na rede OLAM, a partir da utilização de estimadores-*M*. A implementação em Matlab dos classificadores OLAM original e robusto, bem como exemplos numéricos discutindo os desempenhos destes classificadores na presença de *outliers*, são apresentados ao longo do capítulo.

No Capítulo 5 discute-se a proposta de um classificador não-linear robusto. O método tem como bases a rede neural ELM (*Extreme Learning Machine*) e a teoria de regressão robusta, discutida no Capítulo 3. Inicialmente, o Capítulo discute os fundamentos de uma rede neural ELM. Esta utiliza originalmente o método dos mínimos quadrados, baseado no cálculo da pseudoinversa, para estimar a matriz de pesos que conecta os neurônios ocultos aos neurônios de saída. Neste capítulo são também discutidos os fundamentos de um método recente de estimação de parâmetros da rede ELM, o método BIP (*Batch Intrinsic Plasticity*). Por fim, é proposto o desenvolvimento de uma rede ELM Robusta, que também utiliza estimação-*M*. A implementação em Matlab dos classificadores ELM original e robusto, bem como exemplos numéricos discutindo os desempenhos destes classificadores na presença de *outliers*, são apresentados ao final do capítulo.

Uma série abrangente de experimentos computacionais são levados a cabo do Capítulo 6 a fim de avaliar com mais profundidade o classificador OLAM robusto na presença de *outliers*. Já os experimentos computacionais visando uma melhor avaliação do desempenho do classificador ELM robusto são apresentados nos Capítulos 7 e 8. Em particular, no Capítulo 8 são apresentados os resultados referentes ao uso de uma técnica metaheurística baseada em inteligência de enxame para a otimização dos parâmetros do classificador ELM robusto.

No Capítulo 9 são feitas as considerações finais, comentários e análise dos resultados obtidos nesta tese. São analisadas as contribuições propostas, bem como são sugeridos trabalhos futuros relacionados com o tema abordado.

2 *Fundamentos de Regressão Linear*

Dada a importância do tema para a apresentação das propostas desenvolvidas nesta tese, neste capítulo são apresentados conceitos básicos sobre o problema de regressão linear simples e múltipla, bemo como sobre o problema de estimação de parâmetros através do método dos mínimos quadrados.

2.1 O Problema de Regressão Linear

Em muitos problemas práticos, há duas ou mais variáveis numéricas que parecem estar intrisicamente relacionadas, sendo necessário analisar a natureza matemática dessa relação de maneira mais formal a fim de entender melhor o problema. A análise de regressão é uma técnica estatística cujo objetivo principal reside justamente na investigação das relações entre duas ou mais variáveis e na consequente modelagem matemática do problema. A análise de regressão pode ser usada, por exemplo, na construção de um modelo que expresse o resultado de uma variável como função de uma ou mais variáveis. Esse modelo pode, então, ser usado para predizer o resultado de uma variável em função da outra (HINES et al., 2006).

Na descrição que se segue, assume-se que exista uma única variável dependente, ou de resposta, $y \in \mathbb{R}$, relacionada com p variáveis independentes, ou *regressoras*, $x_1, x_2, \ldots, x_p$, $x_j \in \mathbb{R}$. A variável de resposta y é uma variável aleatória, enquanto que as variáveis regressoras $x_1, x_2, \cdots, x_p$ são medidas com erro desprezível e são frequentemente controladas pelo experimentador (usuário). Isto posto, a relação entre y e as p variáveis regressoras é comumente escrita da seguinte forma:

$$y = f(x_1, x_2, \ldots, x_p|\beta) + \varepsilon, \tag{2.1}$$

$$= f(\mathbf{x}|\beta) + \varepsilon, \tag{2.2}$$

em que $f(\cdot|\cdot)$ é denominada de função de regressão, $\mathbf{x} \in \mathbb{R}^p$ é o vetor de variáveis regressoras, $\beta \in \mathbb{R}^p$ é o vetor de parâmetros da função regressora, e ε denota o erro (ruído) aleatório, de

média zero e variância σ_ε^2, presente na medição de y. Assume-se também que ε é uma variável aleatória independente, ou seja, amostras de ε são independentes entre si.

Note que o modelo descrito na Equação (2.1) é um modelo teórico, uma vez que, em geral, nem a função de regressão $f(\cdot|\cdot)$, nem a componente aleatória ε são conhecidas. A escolha da forma funcional da equação de regressão $f(\cdot)$ é feita com base em informação *a priori*, fruto de conhecimento prévio acerca do problema; ou então, através de experimentação com diferentes formas funcionais. A escolha da forma funcional mais adequada a um dado problema é feita (ou pelo menos deveria ser) através de rigoroso processo de análise dos resultados das predições para cada forma funcional escolhida.

Qualquer que seja a forma funcional da equação de regressão, o seu vetor de parâmetros β deve ser estimado. Para isso, faz-se necessário medir um conjunto de n valores de y e de suas variáveis regressoras $\{x_1, x_2, \ldots, x_p\}$:

$$(y_i, x_{i1}, x_{i2}, \cdots, x_{ip}), \qquad i = 1, \ldots, n, \tag{2.3}$$

ou, em forma condensada, faz-se necessário coletar n pares entrada-saída $(y_i, \mathbf{x}_i)$, $i = 1, \ldots n$.

A estimativa do vetor β é simbolizada como $\hat{\beta}$, sendo ela utilizada na seguinte equação para predizer novos valores da variável de resposta:

$$\hat{y} = \hat{f}(x_1, x_2, \ldots, x_p|\hat{\beta}), \tag{2.4}$$

$$= \hat{f}(\mathbf{x}|\hat{\beta}), \tag{2.5}$$

em que $\hat{f}(\cdot|\cdot)$ denota uma aproximação da função de regressão do modelo teórico.

A análise de regressão é dita linear quando se assume que a relação matemática entre as variáveis de interesse é uma função linear de seus parâmetros. Neste caso, o modelo de regressão teórico passa a ser chamado modelo de *regressão linear múltipla*, sendo escrito como

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \varepsilon \tag{2.6}$$

$$= \beta^T \mathbf{x} + \varepsilon \tag{2.7}$$

em que o vetor de parâmetros $\beta \in \mathbb{R}^{p+1}$ contém $p+1$ componentes, $\beta_j, j = 0, 1, \cdots, p$, chamadas genericamente de coeficientes de regressão. Como conseqüência, a primeira componente do vetor $\mathbf{x} \in \mathbf{R}^{p+1}$ é igual a 1, sendo as restantes as próprias variáveis regressoras $\{x_1, x_2, \ldots, x_p\}$.

Os modelos de regressão linear múltipla são usados, em geral, como funções aproximadoras, e a equação de regressão é ajustada ao conjunto de pares entrada saída $(y_i, \mathbf{x}_i)$, $i = 1, \ldots, n$. Lembrando que a verdadeira relação funcional entre y e $x_1, x_2, \cdots, x_p$ é geralmente descon-

hecida, mas em muitos casos práticos o modelo de regressão linear apresenta-se como uma aproximação adequada (HINES et al., 2006). Nestes casos, a equação de predição passa então a ser escrita como

$$\hat{y} = E[y|\mathbf{x}, \hat{\beta}] = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \ldots + \hat{\beta}_p x_p = \hat{\beta}^T \mathbf{x}, \tag{2.8}$$

em que $E[y|\mathbf{x}, \hat{\beta}]$ denota o valor esperado da variável de resposta y condicionado ao vetor de variáveis regressoras $\mathbf{x}$ e à estimativa do vetor de parâmetros $\hat{\beta}$. A Equação (2.8) define um hiperplano no espaço p-dimensional das variáveis regressoras x_j.

Além de sua simplicidade, uma vantagem do modelo linear reside na interpretação direta que pode ser dada ao parâmetro β_j, como representando a mudança esperada na resposta y por unidade de mudança em x_j quando todas as demais variáveis independentes $x_i (i \neq j)$ são mantidas constantes, ou seja, $\beta_j = \frac{dy}{dx_j}$. Isto permite identificar diretamente quais variáveis regressoras são mais relevantes para a variável de saída ou, dito de outra forma, quais variáveis regressoras influenciam mais a variável de resposta.

Quando o problema de regressão lienar envolve apenas uma única variável regressora, x, tem-se uma regressão linear simples. Neste caso, a relação matemática entre uma única variável de entrada x e uma variável de saída y é definida por uma reta, ou seja,

$$y = \beta_0 + \beta_1 x_1 + \varepsilon, \tag{2.9}$$

em que β_0 é o intercepto e β_1, a inclinação da reta. Consequentemente, a equação de predição para o problema de regressão linear simples é dada por

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1, \tag{2.10}$$

em que $\hat{\beta}_0$ e $\hat{\beta}_1$ são, respectivamente, as estimativas do intercepto e do coeficiente angular da reta de regressão.

Como o modelo de regressão linear múltipla é mais geral que o de regressão simples, todo o desenvolvimento teórico que se segue será feito com base nesta formulação do problema de regressão.

2.2 Estimador de Mínimos Quadrados Ordinários

Como já mencionado na seção anterior, os dados a serem usados para estimar o vetor de parâmetros β consistem de n observações do par entrada-saída $(y_i, \mathbf{x}_i)$, $i = 1, \ldots, n$. Em palavras,

a i-ésima observação inclui uma resposta escalar y_i e o vetor de variáveis regressoras correspondente $\mathbf{x}_i = [x_{i1} \;\; x_{i2} \;\; \cdots \;\; x_{ip}]^T$, em que x_{ij} denota a i-ésima observação da j-ésima variável regressora.

Assim, para o modelo de regressão linear múltipla, a variável resposta é uma função linear das variáveis regressoras:

$$\begin{aligned} y_i &= \beta^T \mathbf{x}_i + \varepsilon_i, && (2.11)\\ &= \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \varepsilon_i && (2.12) \end{aligned}$$

em que ε_i corresponde à i-ésima observação do erro aleatório. Para a formulação que se segue, assume-se que as seguintes suposições são verdadeiras:

1. Existem (muito) mais observações que incógnitas (i.e. $n \gg p$).
2. O erro ou ruído no modelo (ε) tem média 0 e variância σ_ε^2.
3. As observações $\{\varepsilon_i\}$ são não-correlacionadas.

Em forma expandida, o modelo de regressão mostrado na Equação (2.11) corresponde a um sistema de equações com n equações e $p+1$ incógnitas, ou seja

$$\begin{aligned} y_1 &= \beta_0 + \beta_1 x_{11} + \beta_2 x_{12} + \cdots + \beta_p x_{1p} + \varepsilon_1 \\ y_2 &= \beta_0 + \beta_1 x_{21} + \beta_2 x_{22} + \cdots + \beta_p x_{2p} + \varepsilon_2 \\ &\vdots \qquad \vdots \qquad \vdots \\ y_n &= \beta_0 + \beta_1 x_{n1} + \beta_2 x_{n2} + \cdots + \beta_p x_{np} + \varepsilon_n \end{aligned} \qquad (2.13)$$

O sistema de equações mostrado na Equação (2.13) pode ser escrito em notação matricial como

$$\mathbf{y} = \mathbf{X}\beta + \varepsilon, \qquad (2.14)$$

em que os vetores $\mathbf{y} \in \mathbb{R}^n$ e $\varepsilon \in \mathbb{R}^n$, assim como a matriz de regressão $\mathbf{X} \in \mathbf{R}^n \times \mathbf{R}^{p+1}$, são definidos como

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}_{n\times 1} \quad \text{e} \quad \mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1p} \\ 1 & x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix}_{n\times(p+1)}, \qquad (2.15)$$

com o vetor de parâmetros $\beta \in \mathbb{R}^{p+1}$ e o vetor aleatório $\varepsilon \in \mathbb{R}^n$ sendo definidos por

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix}_{(p+1)\times 1} \quad \text{e} \quad \varepsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}_{n\times 1}. \tag{2.16}$$

Usar a técnica de mínimos quadrados para encontrar estimativas para os coeficientes de regressão β_j, $j = 1, \ldots, n$, corresponde a minimizar a seguinte função-custo:

$$J(\beta_1, \beta_2, \ldots, \beta_p) = \sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2. \tag{2.17}$$

Dessa forma, minimizar a função-custo $J(\beta_1, \beta_2, \ldots, \beta_p)$ equivale a fazer com que a soma dos quadrados dos desvios ε_i entre os valores observados de y_i e o hiperplano de regressão seja mínima. Em forma vetorial, a função-custo $J(\beta_1, \beta_2, \ldots, \beta_p)$ pode ser escrita como

$$J(\beta) = \|\varepsilon\|^2 = \varepsilon^T \varepsilon = (\mathbf{y} - \mathbf{X}\beta)^T (\mathbf{y} - \mathbf{X}\beta), \tag{2.18}$$

em que $\|\cdot\|$ denota a norma euclidiana de um vetor.

A forma vetorial permite também adicionar uma outra interpretação ao problema de minimizar a função-custo $J(\beta)$: minimizar esta função-custo corresponde a encontrar uma estimativa do vetor de parâmetros β que produza o vetor de erros aleatórios com menor norma quadrática.

As expressões para cálculo das estimativas de β, denotadas como $\hat{\beta}$, podem ser obtidas a partir da minização da função-custo dos erros quadráticos na forma escalar (Equação 2.17) ou na forma vetorial (Equação 2.18). Ambos os casos são apresentados a seguir.

2.2.1 Equações Normais dos Mínimos Quadrados (Forma Escalar)

A função $J(\beta_1, \beta_2, \ldots, \beta_p)$ deve ser minimizada individualmente em relação a cada um dos parâmetros $\beta_0, \beta_1, \cdots, \beta_p$, ou seja, parâmetro a parâmetro. Para isso, a derivada parcial de $J(\beta_1, \beta_2, \ldots, \beta_p)$ deve ser tomada em relação a cada parâmetro β_j, $j = 1, \ldots, p$ e igualada a zero, ou seja

$$\frac{\partial J}{\partial \beta_0} = -2 \sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p} \hat{\beta}_j x_{ij} \right) = 0 \tag{2.19}$$

e

$$\frac{\partial J}{\partial \beta_j} = -2\sum_{i=1}^{n} x_{ij}\left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p}\hat{\beta}_j x_{ij}\right) = 0, \qquad j = 1,2,\ldots,p. \tag{2.20}$$

Resolvendo as Equações (2.19) e (2.20), obtemos um sistema de equações conhecido como *equações normais de mínimos quadrados*, em sua forma escalar:

$$n\hat{\beta}_0 + \hat{\beta}_1\sum_{i=1}^{n} x_{i1} + \hat{\beta}_2\sum_{i=1}^{n} x_{i2} + \cdots + \hat{\beta}_p\sum_{i=1}^{n} x_{ip} = \sum_{i=1}^{n} y_i, \tag{2.21}$$

$$n\hat{\beta}_0\sum_{i=1}^{n} x_{i1} + \hat{\beta}_1\sum_{i=1}^{n} x_{i1}^2 + \hat{\beta}_2\sum_{i=1}^{n} x_{i1}x_{i2} + \cdots + \hat{\beta}_p\sum_{i=1}^{n} x_{i1}x_{ip} = \sum_{i=1}^{n} x_{i1}y_i, \tag{2.22}$$

$$\vdots \qquad\qquad \vdots \qquad\qquad \vdots \qquad\qquad \vdots$$

$$n\hat{\beta}_0\sum_{i=1}^{n} x_{ip} + \hat{\beta}_1\sum_{i=1}^{n} x_{ip}x_{i1} + \hat{\beta}_2\sum_{i=1}^{n} x_{ip}x_{i2} + \ldots + \hat{\beta}_p\sum_{i=1}^{n} x_{ip}^2 = \sum_{i=1}^{n} x_{ip}y_i. \tag{2.23}$$

Note que existem $p+1$ equações normais, uma para cada coeficiente de regressão; logo, o sistema acima é quadrado. A solução das equações normais produz as estimativas de mínimos quadrados dos coeficientes de regressão $\hat{\beta}_0, \hat{\beta}_1, \cdots, \hat{\beta}_p$ (HINES et al., 2006).

2.2.2 Equações Normais dos Mínimos Quadrados (Forma Vetorial)

Para o caso vetorial, a função-custo $J(\beta)$ mostrada na Equação (2.18) precisa primeiro ser decomposta da seguinte forma:

$$\begin{aligned} J(\beta) &= = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta), && (2.24)\\ &= \mathbf{y}^T\mathbf{y} - \beta^T\mathbf{X}^T\mathbf{y} - \mathbf{y}^T\mathbf{X}\beta + \beta^T\mathbf{X}^T\mathbf{X}\beta, && (2.25)\\ &= \mathbf{y}^T\mathbf{y} - 2\beta^T\mathbf{X}^T\mathbf{y} + \beta^T\mathbf{X}^T\mathbf{X}\beta, && (2.26) \end{aligned}$$

em que se fez uso do fato de o termo $\beta^T\mathbf{X}^T\mathbf{y}$ ser um escalar e, portanto, seu transposto $\beta^T\mathbf{X}^T\mathbf{y}^T = \mathbf{y}^T\mathbf{X}\beta$ resulta no mesmo escalar.

Portanto, para minimizar o funcional $J(\beta)$, deve-se tomar a sua derivada parcial em relação ao vetor de parâmetros β e igualá-la ao vetor nulo $\mathbf{0} \in \mathbb{R}^{p+1}$, ou seja

$$\frac{\partial J}{\partial \beta} = -2\mathbf{X}^T\mathbf{y} + 2\mathbf{X}^T\mathbf{X}\hat{\beta} = \mathbf{0}, \tag{2.27}$$

que, simplificando, resulta em

$$\mathbf{X}^T\mathbf{X}\hat{\beta} = \mathbf{X}^T\mathbf{y}, \tag{2.28}$$

que corresponde à versão vetorial das equações normais dos mínimos quadrados mostradas nas

Equações (2.21) a (2.23).

Para encontrar a solução das equações normais, multiplica-se ambos os lados da Equação 2.28 pela inversa de $\mathbf{X}^T\mathbf{X}$. Assim, a estimativa de mínimos quadrados ordinários (MQO) do vetor de parâmetros β é dada por

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}, \tag{2.29}$$

que também é conhecida como solução pelo método da pseudoinversa de Moore-Penrose (GOLUB; VAN LOAN, 1996).

O modelo de predição linear baseado na estimativa MQO do vetor de parâmetros é dado por

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\beta}, \tag{2.30}$$

em que $\hat{\mathbf{y}} \in \mathbb{R}^n$ é o vetor de predições da variável resposta. O vetor de erros de predição é então dado por $\mathbf{e} = \mathbf{y} - \hat{\mathbf{y}}$, sendo que a sua norma quadrática é a menor possível $\|\mathbf{e}\|^2$, segundo o critério dos mínimos quadrados.

Na notação escalar, a equação de predição é dada por

$$\hat{y}_i = \hat{\beta}^T\mathbf{x}_i = \hat{\beta}_0 + \sum_{j=1}^{p}\hat{\beta}_j x_{ij}, \tag{2.31}$$

para $i = 1, 2, \ldots, n$, e o erro correspondente é dado por $e_i = y_i - \hat{y}_i$. A soma dos erros quadráticos (SEQ), $\sum_{i=1}^{n} e_i^2$, que nada mais é do que a norma quadrática $\|\mathbf{e}\|^2$, é comumente usada como critério de avaliação da qualidade da regressão.

2.3 Estimador de Mínimos Quadrados Regularizado

Muitas vezes, a matriz $\mathbf{X}^T\mathbf{X}$ é singular (i.e. não é de posto completo) ou muito próxima da singularidade. Neste caso, tem-se que

$$det(\mathbf{X}^T\mathbf{X}) \approx 0, \tag{2.32}$$

fato este que pode comprometer toda a validade do processo de inferência da regressão linear múltipla, pois é fonte de instabilidades numéricas durante o cálculo da estimativa MQO do vetor de parâmetros mostrada na Equação (2.29). Isto ocorre geralmente quando as variáveis de entrada são intercorrelacionadas. Quando essa intercorrelação é grande, dizemos que existe *multicolinearidade*, ou seja, as linhas da matriz $\mathbf{X}^T\mathbf{X}$ não são linearmente independentes.

A fim de evitar problemas numéricos, faz-se necessário utilizar estratégias que permitam estimar β de modo confiável. Para este fim, um dos métodos mais conhecidos é o *método de regularização de Thikonov* (HOERL; KENNARD, 1970). Utilizando regularização de Thikonov, o estimador de mínimos quadrados de β é dado por

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}. \tag{2.33}$$

em que

- $0 \leq \lambda \ll 1$ é uma constante de valor bem pequeno, e
- $\mathbf{I}$ é uma matriz identidade de dimensão $(p+1) \times (p+1)$.

A regressão que utiliza esse tipo de regularização é chamada de regressão de cumeeira (*ridge regression*) e a função-custo associada é dada por

$$J(\beta) = \|\varepsilon\|^2 + \lambda\|\beta\|^2, \tag{2.34}$$

que permite interpretar a Equação (2.33) como aquela que produz uma estimativa do vetor de parâmetros $\hat{\beta}$ que tenta satisfazer dois critérios de otimalidade:

- Um que procura minimizar a norma do erro quadrático, ou seja, $\|\mathbf{e}\|^2$,
- E um outro que procura minimizar a norma do vetor de parâmetros, ou seja, $\|\beta\|^2$,

tal que a importância do segundo critério frente ao primeiro é regulada pelo valor de λ.

2.4 Regressão Linear no Matlab

A versão vetorial da solução dos mínimos quadrados (Equação (2.29)), assim como a sua versão regularizada mostrada na Equação (2.33), são úteis não apenas pela notação matemática compacta, mas principalmente porque sua implementação em ambientes de computação científica, tais como Matlab©, Octave e Scilab, é possível sem maior esforço de programação.

Como estes ambientes são amplamente utilizados em Engenharia e Ciências, existem diversos comandos e funções que geram, em princípio, os mesmos resultados numéricos para um dado problema de regressão linear simples ou múltipla. Contudo, alguns comandos são mais eficientes, seja porque consomem menos tempo, seja porque são menos susceptíveis a erros numéricos. Isto posto, vale a pena fazer alguns comentários sobre este tema e sugerir

formas eficientes de implementação das equações de estimação de parâmetros pelo método dos mínimos quadrados, ordinário ou regularizado.

Assumindo que o vetor de observações da variável de resposta $\mathbf{y}$ e a matriz de variáveis regresssoras $\mathbf{X}$ são denotadas como `y` e `X` no Matlab, então a Equação (2.29) pode ser implementada da forma que se lê, ou seja,

```
» B = inv(X'*X)*X'y;
```

em que `B` denota a estimativa MQO do vetor de parâmetros β. Contudo, esta forma de se estimar β não é a mais recomendada para problemas de maior escala por duas razões: (*i*) Possui elevado custo computacional, e (*ii*) é muito susceptível a erros numéricos.

Para problemas maiores recomenda-se usar o operador *barra invertida* (\), ou seja

```
» B = X\y;
```

Os cálculos realizados pelo uso do operador *barra invertida* (\) baseiam-se em grande parte no método de ortogonalização conhecido como fatoração QR.

Uma segunda maneira de estimar o vetor β é através do comando `PINV`:

```
» B = pinv(X)*y;
```

Embora a obtenção da solução via comando `pinv` seja computacionalmente mais custosa que a obtida pelo uso do operador *barra invertida*, visto que é baseada na técnica conhecida como SVD (*singular value decomposition*), ela é preferível em problemas em que a matriz de variáveis regressoras $\mathbf{X}$ possui deficiência de posto (i.e. tem posto incompleto).

Por fim, uma terceira maneira de se estimar o vetor β no Matlab é por meio do comando `REGRESS`:

```
» B = regress(y,X);
```

Para a versão regularizada do estimador MQO, também é possível estimar $\hat{\beta}$ através da escrita direta da Equação (2.33) no prompt do Matlab:

```
» l = 0.01;
» I=ones(size(X'*X));
» B = inv(X'*X + l*I)*X'y;
```

Porém, pelas mesmas razões apontadas anteriormente, recomenda-se também usar o operador *barra invertida* (\). Neste caso, a seqüência de comandos passa ser a seguinte:

```
» l = 0.01;
» I=eye(size(X'*X));
» A=X'*X + l*I;
» r=X'*y;
» B = A\r;
```

Um exemplo de aplicação dos comandos acima em um problema real será apresentado logo a seguir. Este exemplo, além de ser útil do ponto de vista didático, será retomado no próximo capítulo, quando abordaremos o problema de regressão robusta.

2.4.1 Exemplo Numérico: Regressão Linear Simples

A título de ilustração, vamos aplicar os conceitos de regressão linear apresentados neste capítulo ao conjunto de dados mostrado na Tabela 2.1. Este conjunto está disponível em Freedman et al. (2007) e envolve duas variáveis: uma é variável regressora $x \in \mathbb{R}$ (consumo per capita de cigarros em um dado país em 1930), enquanto a variável resposta $y \in \mathbb{R}$ corresponde ao número de mortes (por milhão de pessoas) por câncer de pulmão naquele país em 1950.

Índice	País	Cigarro per capita	Mortes por milhão de pessoas
1	Austrália	480	180
2	Canadá	500	150
3	Dinamarca	380	170
4	Finlândia	1100	350
5	Grã Bretanha	1100	460
6	Islândia	230	60
7	Holanda	490	240
8	Noruega	250	90
9	Suécia	300	110
10	Suíça	510	250

Tabela 2.1: Consumo per capita de cigarros em vários países em 1930 e as taxas de morte por câncer de pulmão em 1950.

Como o conjunto de dados envolve apenas um variável regressora é possível visualizar os pares (x_i, y_i), $i = 1, \ldots, 10$ em um diagrama de dispersão (*scatterplot*), marcando um círculo em cada coordenada, conforme ilustrado na Figura 2.1.

Pode-se perceber pelo diagrama de dispersão que há uma tendência linear nos dados, ou seja, há uma tendência de os pontos se organizarem ao longo de uma reta hipotética. Esta reta é, na verdade, a reta de regressão, cujos parâmetros podem ser calculados facilmente usando os

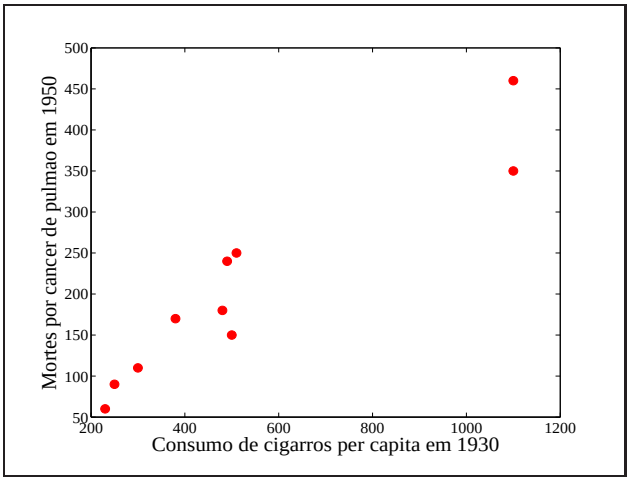


Figura 2.1: Diagrama de dispersão para o conjunto de dados da Tabela 2.1.

comandos do Matlab discutidos na seção anterior. Para isso, podemos usar a seguinte seqüência de comandos:

```
» x=[480; 500; 380; 1100; 1100; 230; 490; 250; 300; 510];
» y=[180; 150; 170; 350; 460; 60; 240; 90; 110; 250];
» n=length(x);
» X=[ones(n,1) x];
» B=X\y
B=
9.1393
0.3687
```

Assim, tem-se que $\hat{\beta}_0$ = 9,14 e $\hat{\beta}_1$ = 0,37, com a equação da reta de regressão sendo dada por $\hat{y}_i = 9,14 + 0,37x$. O gráfico da reta de regressão superposta aos pontos do conjunto de dados é mostrado na Figura 2.2.

No próximo capítulo, este mesmo conjunto de dados, adicionado de mais um par de pontos (x_i, y_i) relativo ao consumo de cigarros nos Estados Unidos, será usado para introduzir conceitos de regressão robusta, ou seja, na presença de pontos discrepantes (*outliers*).

2.5 Resumo do Capítulo

Esse capítulo apresentou os conceitos fundamentais sobre o problema de regressão linear, principalmente o de regressão linear múltipla. A análise de regressão trata da modelagem e investigação das relações entre duas ou mais variáveis, ou seja, da relação entre uma variável

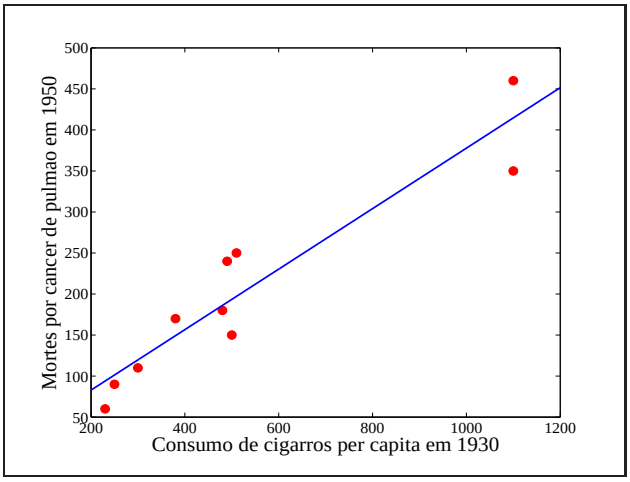


Figura 2.2: Gráfico da reta de regressão ajustada aos dados da Tabela 2.1.

resposta (ou de saída), e uma ou mais variáveis regressoras (independentes). A regressão linear simples se dá quando existe apenas uma variável regressora, e é definida por uma reta, enquanto que a regressão linear múltipla envolve mais de uma variável regressora, sendo definida por um plano.

Para estimar os parâmetros do modelo de regressão linear apresentado neste capítulo foi utilizado o método dos mínimos quadrados ordinário, bem como uma variante regularizada deste método. Além disso, foram apresentadas e discutidas diferentes meios para se implementar o método dos mínimos quadrados, ordinário e regularizado, em ambientes de computação científica, tais como Matlab© e Octave, tendo em vista questões relacionadas a problemas de natureza numérica.

Por fim, um conjunto de dados real foi utilizado com o propósito de ilustrar os conceitos introduzidos neste capítulo, principalmente para mostrar que, graças à formulação vetorial do estimador dos mínimos quadrados, a implementação deste método no Matlab é bem simples e direta.

No próximo capítulo este mesmo conjunto de dados, adicionado de mais um par de pontos (x_i, y_i) relativo ao consumo de cigarros nos Estados Unidos, será usado para introduzir conceitos de regressão robusta, ou seja, na presença de pontos discrepantes (*outliers*).

3 *Fundamentos de Regressão Robusta*

Esse capítulo apresenta, primeiramente, o conceito de amostra discrepante (*outlier*) e discute a influência desses pontos em problemas de análise de regressão. Como ilustração, é apresentado um exemplo numérico com o objetivo de enfatizar o efeito da presença de outliers no processo de estimação dos parâmetros da reta de regressão.

Em seguida, é introduzido o conceito de *regressão linear robusta* como um arcabouço teórico para o projeto de estimadores de parâmetros para modelos de regressão que sejam menos sensíveis a *outliers*. Em poucas palavras, será apresentada uma técnica de estimação alternativa ao método MQO em problemas de regressão linear. A base da regressão robusta é a estimação-*M*, conceito introduzido por Huber (1964), e que também é tratado no capítulo com bastante detalhe.

3.0.1 *Outliers*: Conceituação

Para uma determinada massa de dados, usualmente é possível conceber algum modelo estatístico que explique a geração de seus dados e, a partir deste modelo, novas observações podem ser avaliadas como oriundas ou não da mesma distribuição dos dados que geraram o modelo. Aquelas observações que não puderem ser satisfatoriamente explicadas pelo modelo muitas vezes são consideradas *outliers*.

Ben-Gal (2005) apresenta um resumo de algumas definições de *outliers* obtidas de diversos autores, como as que se seguem.

1. Uma observação que desvia tanto das outras observações, que levanta suspeitas de ter sido gerada por um mecanismo diferente (Hawkins).

2. Uma observação que parece desviar claramente dos outros membros da amostra na qual ela ocorre (BARNETT; LEWIS, 1994).

3. Uma observação em um conjunto de dados que parece ser inconsistente com o restante

daquele conjunto de dados (JOHNSON; GEISSER, 1983).

Mesmo com todas essas definições, não existe, no entanto, uma definição matemática rígida do que constitui um *outlier*, e determinar se uma observação é ou não um desses pontos é, em última análise, um exercício subjetivo. Diversos autores propuseram ao longo dos anos várias definições para esse termo, sem que se tenha ainda uma definição universalmente aceita (HODGE; AUSTIN, 2004). Grosso modo, pode-se resumir as idéias comuns a todas elas na seguinte definição:

> *Um **outlier** é uma observação (escalar ou vetorial) que difere marcadamente dos outros membros da amostra a que ela pertence, tal como indicado pelo modelo escolhido para representar esta amostra.*

Coloquialmente, pode-se dizer que *outliers* são *pontos fora da curva*, pois não partilham da mesma distribuição de probabilidades que a maioria das outras observações, ou seja, são observações diferentes das usuais.

Segundo Stevens (1984), é comum distinguir dois tipos de *outliers*, a saber: (*i*) *outliers* na variável resposta (saída), o que pode ser um indicativo de falhas no modelo. Este é o tipo de *outlier* que as pessoas em geral tem em mente quando falam de *outliers* em problemas de regressão. (*ii*) **Pontos de alavancagem**, que são *outliers* nas variáveis regressoras (entradas) e que também podem afetar o modelo de regressão, mas que não necessariamente implicam na produção de *outliers* na variável resposta.

É importante mencionar que nem todos os pontos de alavancagem terão influência nos valores dos coeficientes de regressão. Uma dada observação pode ser um *outlier*, ou um ponto de alavancagem, ou ambos. Se uma observação é ao mesmo tempo um outlier e um ponto de alavancagem, então é costume chamá-la de **ponto de influência** (*influential point*).

Várias métricas podem ser usadas para caracterizar um ponto como *outlier*. Em Stevens (1984), são analisadas duas dessas métricas, o resíduo padronizado (*standardized residual*), r_i, e o resíduo studentizado (*studentized residual*), r_{-i}.

O resíduo padronizado, r_i, é calculado através da seguinte expressão:

$$r_i = \frac{e_i}{\hat{\sigma}\sqrt{1-h_{ii}}} \tag{3.1}$$

em que $e_i = y_i - \hat{y}_i$ é o valor do resíduo bruto, $\hat{\sigma}$ é uma estimativa do desvio padrão dos resíduos e h_{ii} é o i-ésimo elemento diagona principal da matriz *chapéu* $\mathbf{H} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$, com a matriz

$\mathbf{X}$ sendo definida como na Equação (2.15). A matriz $\mathbf{H}$ recebe esta denominação pelo fato de ela literalmente "por um chapéu" (acento circunflexo usado para simbolizar uma estimativa) em $\mathbf{y}$, já que $\hat{\mathbf{y}} = \mathbf{X}\hat{\beta} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{H}\mathbf{y}$ (VIEIRA; DAL BELLO, 2006).

Os valores de h_{ii} também podem ser calculados diretamente por meio da seguinte expressão:

$$h_{ii} = \mathbf{x}_i^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_i, \tag{3.2}$$

em que $\mathbf{x}_i^T$ corresponde à i-ésima linha da matriz $\mathbf{X}$.

O valor de h_{ii} fornece uma medida relativa do quanto a i-ésima observação $\mathbf{x}_i$ é um *outlier* no espaço das variáveis regressoras. Estes valores encontram-se entre 0 e 1, e quando h_{ii} é grande (i.e. próximo de 1), a variância para o i-ésimo resíduo é próxima de zero, pois $Var(e_i) = \hat{\sigma}^2(1-h_{ii})$. Isto significa que $y_i \approx \hat{y}_i$; ou seja, uma observação pode se ajustar bem ao modelo linear e, ainda assim, ser um ponto de influência. Tal diagnóstico sinaliza, portanto, observações que devem ser examinadas com cuidado (STEVENS, 1984).

É importante ressaltar que assume-se para r_i uma distribuição normal, com média zero. Se o modelo estiver correto, então 99% de r_i deve estar contido dentro de três desvios padrões da média. Desta forma, qualquer resíduo bruto e_i que produzir um resíduo padronizado com valor absoluto maior que três (i.e. $|r_i| \geq 3$) pode ser considerado um *outlier*.

Por sua vez, o resíduo studentizado é calculado da seguinte forma:

$$r_{-i} = \frac{e_i}{\hat{\sigma}_{-i}\sqrt{1-h_{ii}}}, \tag{3.3}$$

sendo que $\hat{\sigma}_{-i}$ é o desvio padrão dos resíduos com o i-ésimo caso removido. Uma relação entre os resíduos r_i e r_{-i} é mostrada a seguir:

$$r_{-i} = \frac{\hat{\sigma}e_i}{\hat{\sigma}_{-i}\hat{\sigma}\sqrt{1-h_{ii}}} = \frac{\hat{\sigma}}{\hat{\sigma}_{-i}}r_i. \tag{3.4}$$

Pela Equação (3.4), é possível notar que o resíduo studentizado tende a ser mais sensível na detecção de *outliers* e, portanto, preferível na maioria das aplicações. Para medir se uma dada observação é um ponto de influência, outras medidas fazem-se necessárias como, por exemplo, a distância de Cook (STEVENS, 1984), porém este tópico está fora do escopo desta tese e não será aprofundado.

Em relação à ocorrência, a presença de *outliers* nos dados tem causas variadas, tais como:

- Erros de medida durante a coleta dos dados;
- Erros de impressão ou gravação dos dados;

- Falhas no sistema (e.g. elétricas ou mecânicas);
- Comportamento inesperado do sistema (comportamento fraudulento);
- Mudança do ponto de operação do sistema (e.g. sistema não-estacionário);
- Flutuações estatísticas inerentes ao conjunto de dados.

Dados sintéticos (i.e. artificialmente gerados), gerados com grande controle do experimentador, não apresentam erros visíveis, grosseiros, ou valores equivocados. Dados reais, no entanto, normalmente contêm tais erros, os quais geralmente se mostram como *outliers*. Um *outlier* é definido, portanto, como uma observação que é separada de alguma forma do resto dos dados, sendo referenciado como uma observação extrema, ou seja, remota, distante, longínqua. Eles se desviam do padrão de dados do conjunto, e podem levar à especificação incorreta do modelo, estimação enviesada de parâmetros e resultados incorretos.

No entanto, apesar de *outliers* serem muitas vezes considerados erro ou ruído, eles podem também conter informações importantes (BEN-GAL, 2005). Alguns são genuinamente informativos e podem ser as observações mais importantes da amostra. Assim, alguns autores consideram que não é suficiente examinar os dados e remover os *outliers* e, em relação a isso, existem vários aspectos a considerar (RIPLEY, 2004):

1. Usuários, mesmo estatísticos experientes, nem sempre examinam os dados;
2. A difícil decisão de manter ou rejeitar uma observação é perda de tempo. Pode-se fazer melhor diminuindo o peso de observações dúbias do que rejeitando-as, apesar de se poder querer rejeitar observações erradas;
3. Pode ser difícil, ou mesmo impossível, localizar *outliers* em dados multivariados ou altamente estruturados;
4. Rejeitar *outliers* afeta a teoria da distribuição. Em particular, variâncias serão subestimadas a partir dos dados "limpos".

Um fato importante, portanto, a ser considerado em relação à remoção de *outliers*, é que corre-se o risco de remover esses pontos do conjunto de dados simplesmente porque eles parecem não se ajustar ao restante dos pontos. Dessa forma, corre-se o risco de escolher dados com base apenas no desejo de melhorar os resultados, excluindo dados que estejam em desacordo com uma determinada hipótese levantada. Nesse caso, a remoção de *outliers* poderia levar à obtenção de um resultado enviesado, em vez de a um resultado verdadeiramente significativo.

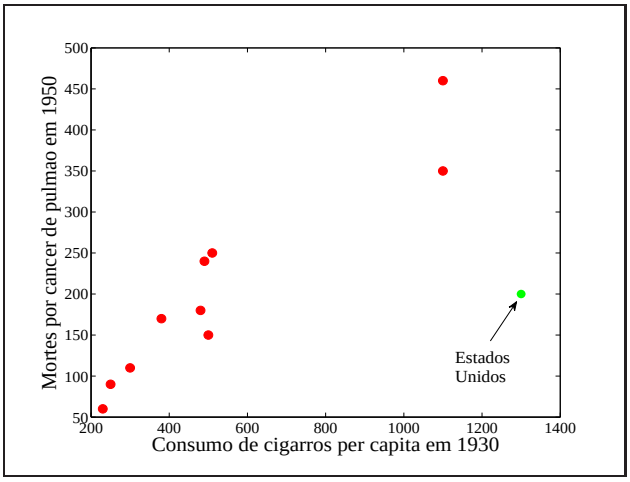


Figura 3.1: Diagrama de dispersão para o conjunto de dados da Tabela 3.1.

Após a definição de *outliers* e a discussão sobre a conveniência de sua remoção ou não, a seguir é apresentado um exemplo numérico cujo objetivo é mostrar como a presença de *outliers* pode influenciar no processo de obtenção do modelo de regressão linear.

3.0.2 Exemplo Numérico: *Outliers* na Regressão Linear Simples

A título de ilustração vamos aplicar os conceitos de regressão linear apresentados neste capítulo ao conjunto de dados mostrado na Tabela 2.1 acrescido do par entrada-saída (x_i, y_i) relativo aos dados dos Estados Unidos. O conjunto completo, com informação de 11 países, está mostrado na Tabela 3.1.

País		Cigarro per capita	Mortes por milhão de pessoas
1	Austrália	480	180
2	Canadá	500	150
3	Dinamarca	380	170
4	Finlândia	1100	350
5	Grã Bretanha	1100	460
6	Islândia	230	60
7	Holanda	490	240
8	Noruega	250	90
9	Suécia	300	110
10	Suíça	510	250
11	Estados Unidos	1300	200

Tabela 3.1: Consumo per capita de cigarros em vários países em 1930 e as taxas de morte por câncer de pulmão em 1950 com dados dos Estados Unidos.

O diagrama de dispersão dos pares (x_i, y_i), $i = 1, \ldots, 11$ está ilustrado na Figura 3.1, com o ponto relativo aos Estados Unidos em destaque.

Pode-se perceber pelo diagrama de dispersão que a hipótese de haver uma tendência linear nos dados fica comprometida, ou seja, não está tão claro mais de que há uma tendência de os pontos se organizarem ao longo de uma reta hipotética. Mesmo assim, podemos usar a mesma

seqüência de comandos do Matlab/Octave que foi usada no exemplo numérico discutido no Capítulo 2, tomando o cuidado para incluir o par entrada-saída (1300, 200) dos Estados Unidos, a fim de verificar o efeito da presença do novo ponto. Assim, temos

```
» x=[480; 500; 380; 1100; 1100; 230; 490; 250; 300; 510; 1300];
» y=[180; 150; 170; 350; 460; 60; 240; 90; 110; 250; 200];
» n=length(x);
» X=[ones(n,1) x];
» B=X\y
B =
67.5609
0.2284
```

É fácil notar que os parâmetros da reta de regressão foram consideravelmente alterados. As equações das retas de regressão *com* ou *sem* os dados dos Estados Unidos estão mostradas na Tabela 3.2.

Dados	**Parâmetro β_0** (intercepto)	**Parâmetro β_1** (inclinação)	**Reta de Regressão**
Sem EUA	9,14	0,37	$\hat{y}_i = 9,14 + 0,37x_i$
Com EUA	67,56	0,23	$\hat{y}_i = 67,56 + 0,23x_i$

Tabela 3.2: Resultado da estimação dos parâmetros da reta de regressão para os dados da Tabela 3.1 com e sem a informação dos Estados Unidos (EUA).

Os gráficos das duas retas de regressão resultantes, com e sem a inclusão do par $(1300, 200)$, estão mostrados na Figura 3.2. Fica claro que as retas de regressão para os dois casos levariam a conclusões diferentes, caso fossem usadas para prever o número de mortes (por milhão de pessoas) para um certo país que já estava contemplado na Tabela 3.1. O resultado seria ainda mais tendencioso (ou viesado, no jargão estatístico) para países que não estão contemplados na referida tabela, mas para os quais se deseja fazer um previsão.

Na próxima seção será introduzido um arcabouço teórico, conhecido como regressão robusta, desenvolvido com o intuito de lidar com problemas de regressão na presença de *outliers*. Este arcabouço é particularmente útil em problemas de regressão múltipla, pois a visualização dos dados em diagramas de dispersão não é possível ou bastante difícil.

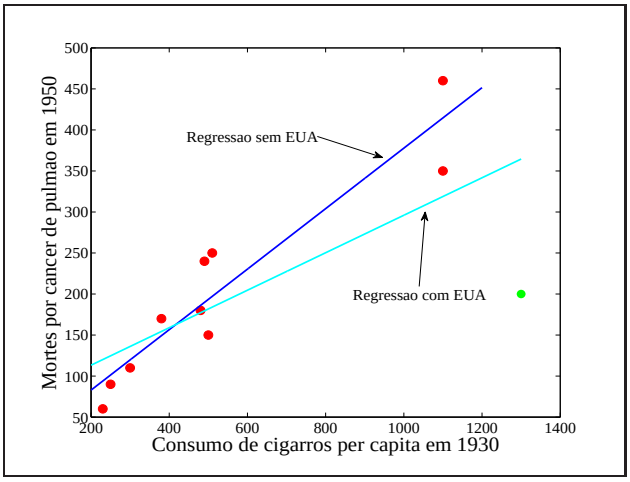


Figura 3.2: Gráfico das retas de regressão ajustadas aos dados da Tabela 2.1 com e sem a informação dos Estados Unidos (EUA).

3.1 Estatística Robusta: Um Breve Apanhado Histórico

O arcabouço teórico de estatística robusta tem como um de seus objetivos propor alternativas ao método de estimação MQ ordinário, devido às restrições deste em relação ao tratamento de dados do mundo real.

A primeira questão tratada pela estatística robusta é a da suposição de que os erros derivam de uma distribuição normal. No caso de normalidade, o método MQ produz os estimadores mais eficientes (RAO; TOUTENBURG, 1999), mas algumas vezes, em problemas reais, não é possível fazer tal suposição. Em casos nos quais esta suposição não se aplica, os resultados do estimador MQ podem ser afetados, estando inclusive, segundo Andrews (1974), muito longe do estimador ótimo para erros com distribuição não-normal e/ou com caudas longas.

Apesar de a média ser um estimador ótimo (segundo o critério MQ) do parâmetro de localização da distribuição normal, ela pode ser apenas subótima mesmo para distribuições próximas à normal, sendo assim necessária a utilização de métodos que visam obter estimadores de alta eficiência em situações não-ideais (ou aproximadas) para o método de estimação assumido (RIPLEY, 2004). Um exemplo desse tipo de método são os métodos de estatística robusta.

A suposição feita pelo método MQ, de que os erros são normalmente distribuídos, é uma suposição amplamente usada em modelagem. No entanto, medidas independentes de um mesmo experimento, tomadas separadamente, geralmente diferem em seus valores. Isto ocorre porque o erro entre elas é aleatório, e não pode (ou é muito difícil de) ser precisamente caracterizado. Para os valores observados, portanto, como definir a melhor estimativa do valor verdadeiro desconhecido?

Segundo Hampel (2001), essa questão já havia sido considerada por Gauss, que ao perceber que precisava da distribuição de erro para respondê-la, inverteu o problema ao perguntar qual seria a distribuição de erro que fazia com que uma regra geralmente aceita, a média aritmética, fosse considerada uma boa regra de estimação do valor verdadeiro da média. Isso foi o que levou Gauss a assumir a distribuição normal como a distribuição ideal para os erros observados em suas medidas.

As distribuições de erros reais, no entanto, geralmente não são normais. Podem ser bem próximas, mas tendem a possuir caudas mais longas. Além disso, Gauss tratou observações de igual acurácia, mas os dados reais têm acurácia diferente. Muitos problemas são resolvidos através do "dogma da normalidade", considerando que os erros assumem uma distribuição normal, mas o teorema do limite central, sendo um teorema limite, apenas sugere normalidade aproximada sob condições bem específicas em situações reais.

A crença (implícita ou explícita) de que, sob normalidade aproximada, o estimador MQ ainda seria aproximadamente ótimo, foi rechaçada por Tukey, que mostrou que já sob pequenos desvios da normalidade, o desvio médio era melhor que o desvio padrão, apesar de sua perda de eficiência de 12% sob normalidade estrita (HAMPEL, 2001).

Para Andrews (1974), métodos de regressão são resistentes quando o resultado não é grandemente alterado no caso de modificação em uma pequena fração dos dados. Além disso, tais métodos são robustos em eficiência quando sua eficiência estatística continua alta para condições mais realísticas que casos idealizados de distribuições normais com erros de variâncias constante (i.e. homocedasticas).

A teoria de estatística robusta lida com desvios a partir de suposições no modelo e preocupa-se com a construção de procedimentos estatísticos confiáveis e razoavelmente eficientes em uma vizinhança do modelo (RONCHETTI, 2006). Essa teoria pode ser vista, portanto, como uma teoria estatística lidando com modelos paramétricos aproximados, sendo uma ponte entre a abordagem paramétrica e a abordagem não-paramétrica. É um compromisso razoável entre a rigidez de um modelo paramétrico rigoroso e as potenciais dificuldades de interpretação de uma análise não-paramétrica.

Segundo Ronchetti (2006), algumas das principais contribuições da estatística robusta para a estatística moderna são listadas abaixo:

- **Modelos são apenas aproximações da realidade**. Esta é uma sentença padrão em ciência, mas a estatística robusta ajudou a reforçar e quantificar esse ponto. Iniciando com Tukey (1960), foi demonstrada a perda dramática de eficiência de procedimentos ótimos

na presença de pequenos desvios a partir do modelo estocástico assumido. Isso abriu a porta para procurar alternativas melhores e para múltiplas ferramentas de análise de dados.

- **Múltiplas variáveis e soluções para um problema de análise de dados**. Este ponto foi apresentado por Tukey (1964) em seu artigo pioneiro sobre o futuro da análise de dados. Estatística robusta contribuiu para desenvolver a idéia de que múltiplas variáveis são necessárias para analisar dados reais, e que problemas reais podem ter múltiplas soluções.

- **A abordagem minimax**. Esta abordagem, emprestada da teoria dos jogos, foi a solução elegante de Huber (1964) para o problema de robustez, visto como um jogo entre a Natureza, que escolhe uma distribuição dos dados em uma vizinhança do modelo, e o estatístico, que escolhe um estimador em uma dada classe. O saldo é a variância assintótica do estimador em uma dada distribuição. Algumas vezes, soluções minimax podem ser pessimistas, mas revelou-se que esse não era o caso aqui. O estimador resultante, estimador de Huber, se tornou o bloco de construção básico de qualquer procedimento robusto.

- **Estimação-*M***. Estimação-*M* (HUBER, 1964) representa um arcabouço teórico abrangente que cobre uma classe muito flexível de estimadores que vem exercendo um papel importante no desenvolvimento da estatística robusta e na construção de procedimentos de estimação robustos.

Dada a importância do tópico Estimação-*M* para o desenvolvimento das idéias presentes nesta tese, esta classe de estimadores será descrita em mais detalhes na próxima seção.

3.2 Fundamentos de Estimação-*M*

Huber (1964) introduziu a seguinte questão:

> O que acontece se a distribuição dos dados desvia um pouco da distribuição normal assumida?

Nesta situação, a média amostral pode ter um desempenho catastroficamente ruim, pois mesmo pequenos desvios podem explodir sua variância. Tukey e outros propuseram vários substitutos robustos para a média, como média truncada (*trimmed mean*), média winsorizada (*Winsorized mean*), etc., e exploraram seu desempenho para algumas violações de normalidade.

Huber, no entanto, queria definir uma teoria geral de estimação robusta. Sua questão era saber se era possível obter mais robustez minimizando uma outra função dos erros, em vez da soma de seus quadrados. Esta motivação o levou a se concentrar em estimadores que podiam ser definidos por um critério de minimização da seguinte forma:

$$T = T_n(x_1, \ldots, x_n) \quad \text{minimiza} \quad \sum_i \rho(x_i - T), \tag{3.5}$$

em que T é uma estimativa de x_i calculada a partir dos dados observados e $\rho(\cdot)$ é uma função não-constante. Esta classe de estimadores abrange em particular (HUBER, 1964):

(i) a média amostral, quando $(\rho(e) = e^2)$,

(ii) a mediana amostral $(\rho(e) = |e|)$,

(iii) todos os estimadores de máxima verossimilhança $(\rho(e) = -\log f(e))$, em que $f(\cdot)$ é a densidade de probabilidade de e.

Esses estimadores são conhecidos como estimadores-M, em que M significa 'do tipo Máxima Verossimilhança'. Huber (1964) introduziu o conceito de estimação-M como um processo para estimar os parâmetros de um estimador-M. De forma resumida, o que ele sugeriu para obter mais estimativas robustas dos parâmetros foi minimizar uma outra função dos erros em vez da soma de seus quadrados.

3.2.1 Estimação-M em Regressão

A introdução sobre estimação-M feita no início desta seção tomou como base a estimativa de parâmetros de uma distribuição, tais como média e mediana. Nesta tese, porém, a principal utilidade do arcabouço teórico provido pela estimação-M é na estimação dos parâmetros de modelos de regressão linear múltipla.

Da mesma forma que a média amostral, por ser uma estimativa MQ da média real da distribuição dos dados, as estimativas MQ dos parâmetros de um modelo de regressão linear podem ter seus valores fortemente influenciados quando a distribuição dos erros não é normal, em particular quando tal distribuição tem caudas pesadas. Em distribuições deste tipo, valores extremos (i.e. *outliers*) têm probabilidade alta de ocorrer.

Uma abordagem muito comum consiste em perscrutar os dados em busca de outliers, a fim de eliminá-los. Outra abordagem, denominada *regressão robusta*, consiste em empregar

um critério para ajuste do modelo de regressão aos dados que não seja tão vulnerável quanto o critério MQ para dados que desviam da suposição de normalidade.

Para isso, vamos considerar o modelo de regressão linear múltipla mostrado na Equação (2.6), repetido aqui para maior clareza de exposição:

$$y_i = \beta^T \mathbf{x}_i + \varepsilon_i, \tag{3.6}$$

$$= \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \varepsilon_i, \tag{3.7}$$

sendo que o modelo ajustado é representado por

$$\hat{y}_i = \hat{\beta}\mathbf{x}_i = \hat{\beta}_0 + \sum_{j=1}^{p} \hat{\beta}_j x_{ij}, \tag{3.8}$$

para $i = 1, 2, \ldots, n$, e o erro correspondente é dado por $e_i = y_i - \hat{y}_i$. Baseado na teoria de Huber, um estimador-*M* geral minimiza a seguinte função objetivo (FOX, 2002):

$$\sum_{i=1}^{n} \rho(e_i) = \sum_{i=1}^{n} \rho(y_i - \hat{\beta}^T \mathbf{x}_i) \tag{3.9}$$

em que o papel da função $\rho(\cdot)$ é computar a contribuição de cada resíduo à função objetivo, y_i é o valor observado da variável de resposta, $\mathbf{x}_i$ é o vetor de variáveis regressoras, e $\hat{\beta}$ é o vetor de parâmetros estimados.

A título de ilustração, o estimador MQ é um caso particular de estimador-*M*, em que $\rho(e_i) = e_i^2$. A função ρ deve, em geral, ter as seguintes propriedades (FOX, 1997), (OES; LIMA, 2010):

1. Ser não-negativa: $\rho(e_i) \geq 0$.
2. Passar pela origem: $\rho(0) = 0$.
3. Ser par: $\rho(e_i) = \rho(-e_i)$.
4. Ser monotonicamente crescente: $\rho(e_i) \geq \rho(e_{i'}), \quad \text{para} \quad |e_i| > |e_{i'}|)$.

Seja $\psi = \rho'$ a derivada de ρ em relação a e_i. Diferenciando-se ρ em relação ao vetor $\hat{\beta}$, tem-se

$$\sum_{i=1}^{n} \psi(y_i - \hat{\beta}^T \mathbf{x}_i)\mathbf{x}_i^T = \mathbf{0}, \tag{3.10}$$

em que $\mathbf{0} \in \mathbb{R}^p$ é o vetor nulo.

Em seguida, definimos a função de ponderação como $w(e_i) = \frac{\psi(e_i)}{e_i}$, e denotamos $w_i = w(e_i)$.

Assim, as equações de estimação dos parâmetros são dadas por

$$\sum_{i=1}^{n} w_i(y_i - \hat{\beta}^T \mathbf{x}_i)\mathbf{x}_i^T = \mathbf{0}, \tag{3.11}$$

de onde se conclui que solucionar estas equações corresponde a solucionar um problema de mínimos quadrados ponderado, minimizando $J(\beta) = \varepsilon^T \mathbf{W}\varepsilon = \sum_i w_i^2 e_i^2$.

É importante ressaltar, porém, que os pesos dependem dos resíduos (isto é, dos erros estimados), os resíduos, por sua vez, dependem dos parâmetros estimados, e os parâmetros estimados dependem dos pesos. Como consequência, não há uma fórmula fechada para estimação dos parâmetros $\hat{\beta}_j$, $j = 0, \ldots, p$, como no caso do estimador MQ. Neste caso, recorre-se a um método iterativo de estimação chamado *iteratively reweighted least-squares* (IRLS) (FOX, 1997), cujos passos são descritos a seguir.

Algoritmo IRLS (Regressão Robusta)

Passo 1 - Prover uma estimativa inicial $\hat{\beta}(0)$ usando o método MQ, tal como descrito na Eq. (2.29).

Passo 2 - A cada iteração t, calcular os resíduos a partir das iterações anteriores $e_i(t-1)$, bem como os pesos correspondentes $w_i(t-1) = w[e_i(t-1)]$.

Passo 3 - Obter uma nova estimativa de mínimos quadrados ponderados para $\beta(t)$:

$$\hat{\beta}(t) = [\mathbf{X}^T \mathbf{W}(t-1)\mathbf{X}]^{-1} \mathbf{X}^T \mathbf{W}(t-1)\mathbf{y}, \tag{3.12}$$

em que $\mathbf{W}(t-1) = \text{diag}\{w_i(t-1)\}$ é a matriz $n \times n$ de pesos da iteração atual.

Repetir os Passos 2 e 3 até a convergência do vetor $\hat{\beta}_i(t)$.

3.2.2 Funções Objetivo para Regressão Robusta

Uma das vantagens de se trabalhar com estimadores-*M* deve-se ao fato de a escolha da função ρ ser flexível, sendo que cada escolha leva a diferentes estimadores, incluindo aí o estimador MQ ordinário (MQO) se fizermos $\rho(e_i) = e_i^2$. As funções objetivo ($\rho(\cdot)$) e peso ($w(\cdot)$) de nove estimadores-*M* comuns, a saber, Andrews, Bisquare, Cauchy, Fair, Huber, Logistic, MQO, Talwar e Welsch, são apresentadas nas Tabelas 3.3 e 3.4, respectivamente, enquanto que seus gráficos são apresentados nas Figuras 3.3 e 3.4.

Nome	**Função Objetivo (ρ)**
Andrews	$\begin{cases} k^2[1-\cos(\frac{e_i}{k})], & \lvert\frac{e_i}{k}\rvert \leq \pi \\ 2k^2, & \lvert\frac{e_i}{k}\rvert > \pi \end{cases}$
Bisquare	$\begin{cases} \frac{k^2}{6}\left\{1-\left[1-\left(\frac{e_i}{k}\right)^2\right]^3\right\}, & \lvert\frac{e_i}{k}\rvert \leq 1 \\ \frac{k^2}{6}, & \lvert\frac{e_i}{k}\rvert > 1 \end{cases}$
Cauchy	$\frac{k^2}{2}\log(1+(\frac{e_i}{k})^2)$
Fair	$k^2\left[\frac{\lvert e_i\rvert}{k}-\log(1+\frac{\lvert e_i\rvert}{k})\right]$
Huber	$\begin{cases} \frac{1}{2}e_i^2, & \lvert\frac{e_i}{k}\rvert \leq 1 \\ k\lvert e_i\rvert-\frac{1}{2}k^2, & \lvert\frac{e_i}{k}\rvert > 1 \end{cases}$
Logistic	$k^2\log[\cosh(\frac{e_i}{k})]$
MQO	e_i^2
Talwar	$\begin{cases} \frac{e_i^2}{2}, & \lvert\frac{e_i}{k}\rvert \leq 1 \\ \frac{k^2}{2}, & \lvert\frac{e_i}{k}\rvert > 1 \end{cases}$
Welsch	$\frac{k^2}{2}\left[1-\exp(-(\frac{e_i}{k})^2)\right]$

Tabela 3.3: Funções Objetivo (ρ) para diversos estimadores-*M* comumente encontrados na literatura.

Analisando algumas funções peso da Tabela 3.4, fica claro que uma das causas da ineficiência do estimador MQO reside no fato de a sua função peso atribuir pesos iguais a cada erro. Isto faz com que os erros causados por *outliers*, tenham a mesma importância que erros causados por outras amostras. Por outro lado, os pesos em Huber caem quando $|e| > k$, e os pesos em Bisquare caem assim que e se afasta de 0, e são 0 para $|e| > k$ (FOX, 1997). Assim, como *outliers* produzem erros maiores que outras amostras, estimadores Huber e Bisquare reduzem a influência desses pontos.

Como pôde ser notado no parágrafo anterior, para cada estimador faz-se necessário escolher um limiar k. Valores pequenos de k produzem mais resistência a *outliers*, mas ao custo de menor eficiência quando os erros são normalmente distribuídos. Este limiar é geralmente escolhido para dar eficiência razoavelmente alta no caso de erros normalmente distribuídos; em particular, $k = 1,345\sigma$ para a função Huber e $k = 4,685\sigma$ para a função Bisquare, em que σ é o desvio padrão dos erros, os quais produzem 95% de eficiência quando os erros são normais, e ainda oferece proteção contra *outliers* (FOX, 1997).

Na prática, é preciso estimar o desvio padrão dos erros para usar os valores de k sugeridos no parágrafo anterior para as funções Huber e Bisquare. Usualmente, uma medida robusta de dispersão é empregada no lugar do desvio padrão dos resíduos. Por exemplo, uma abordagem comum é fazer

$$\hat{\sigma} = \frac{MAR}{0,6745}, \tag{3.13}$$

em que *MAR* é a mediana dos valores absolutos dos resíduos (FOX, 1997). A constante 0,6745

Nome	Função Peso (w)
Andrews	$\begin{cases} (\frac{e_i}{k})^{-1}\sin(\frac{e_i}{k}), & \lvert\frac{e_i}{k}\rvert \leq \pi \\ 0, & \lvert\frac{e_i}{k}\rvert > \pi \end{cases}$
Bisquare	$\begin{cases} \left[1-\left(\frac{e_i}{k}\right)^2\right]^2, & \lvert\frac{e_i}{k}\rvert \leq 1 \\ 0, & \lvert\frac{e_i}{k}\rvert > 1 \end{cases}$
Cauchy	$\frac{1}{1+(\frac{e_i}{k})^2}$
Fair	$\frac{1}{1+\frac{\lvert e_i\rvert}{k}}$
Huber	$\begin{cases} 1, & \lvert\frac{e_i}{k}\rvert \leq 1 \\ \frac{k}{\lvert e_i\rvert}, & \lvert\frac{e_i}{k}\rvert > 1 \end{cases}$
Logistic	$(\frac{e_i}{k})^{-1}\tanh(\frac{e_i}{k})$
MQO	1
Talwar	$\begin{cases} 1, & \lvert\frac{e_i}{k}\rvert \leq 1 \\ 0, & \lvert\frac{e_i}{k}\rvert > 1 \end{cases}$
Welsch	$\exp(-(\frac{e_i}{k})^2)$

Tabela 3.4: Funções Peso (ω) de diversos estimadores-M.

torna a estimativa não viesada para a distribuição normal. Se existem $p+1$ colunas na matriz de variáveis regressoras $\mathbf{X}$, os menores $p+1$ desvios absolutos são excluídos ao se calcular a mediana.

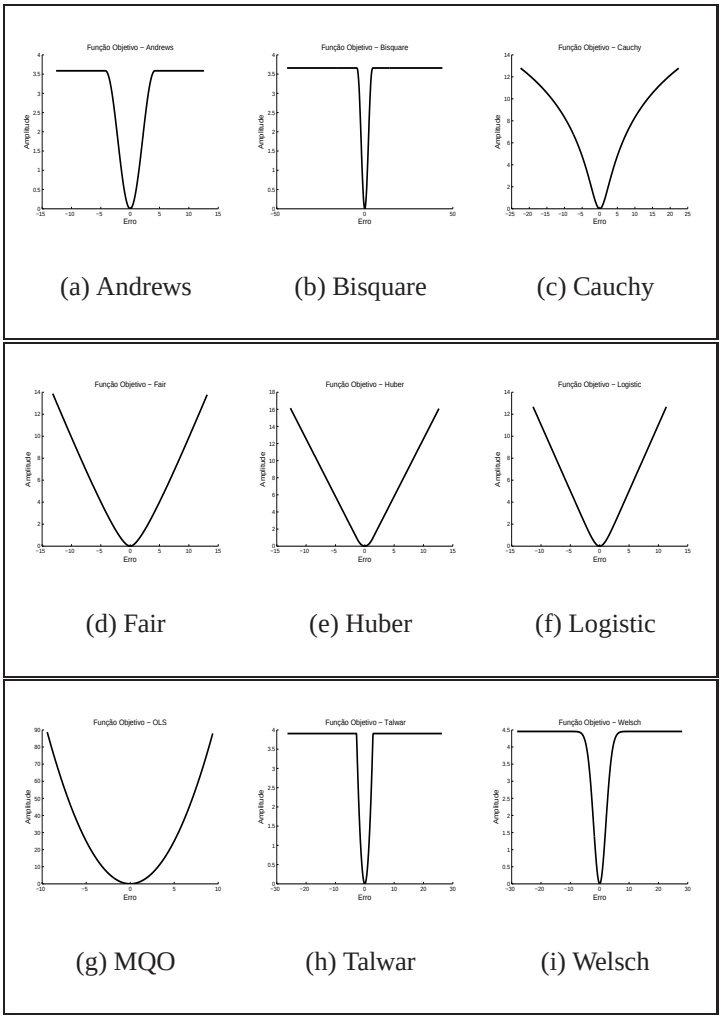
3.2.3 Regressão Robusta no Matlab

A fim de implementar técnicas de regressão robusta nesse trabalho, pode-se utilizar a função `ROBUSTFIT` do Matlab, que contempla os nove estimadores-M apresentados nas Tabelas 3.3 e 3.4. Valores *default* para o limiar resultam em estimativas do vetor de parâmetros que, conforme citado anteriormente, são aproximadamente 95% tão eficientes estatisticamente quanto as estimativas usuais de mínimos quadrados, desde que a resposta tenha uma distribuição normal sem *outliers*. Quando o valor do limiar diminui, diminuem os valores dos pesos atribuídos a resíduos de valor elevado (causados por *outliers*). Quando o valor do limiar aumenta, aumentam-se os valores dos pesos atribuídos a resíduos de valor elevado.

As funções peso presentes no comando `ROBUSTFIT` do Matlab e os valores *default* de seus respectivos limiares são mostrados na Tabela 3.5. Nesta tabela, o parâmetro r_i é definido como

$$r_i = \frac{e_i(t-1)}{k \cdot \hat{\sigma} \cdot \sqrt{(1-h_i)}} \tag{3.14}$$

em que $e_i(t-1)$ é o resíduo da iteração anterior do algoritmo IRLS, k é o limiar, h_i é o valor de alavancagem (*leverage value*) para a i-ésima observação $\mathbf{x}_i$ resultante do ajuste pelo método



(a) Andrews (b) Bisquare (c) Cauchy

(d) Fair (e) Huber (f) Logistic

(g) MQO (h) Talwar (i) Welsch

Figura 3.3: Gráfico das funções objetivo mostradas na Tabela 3.3.

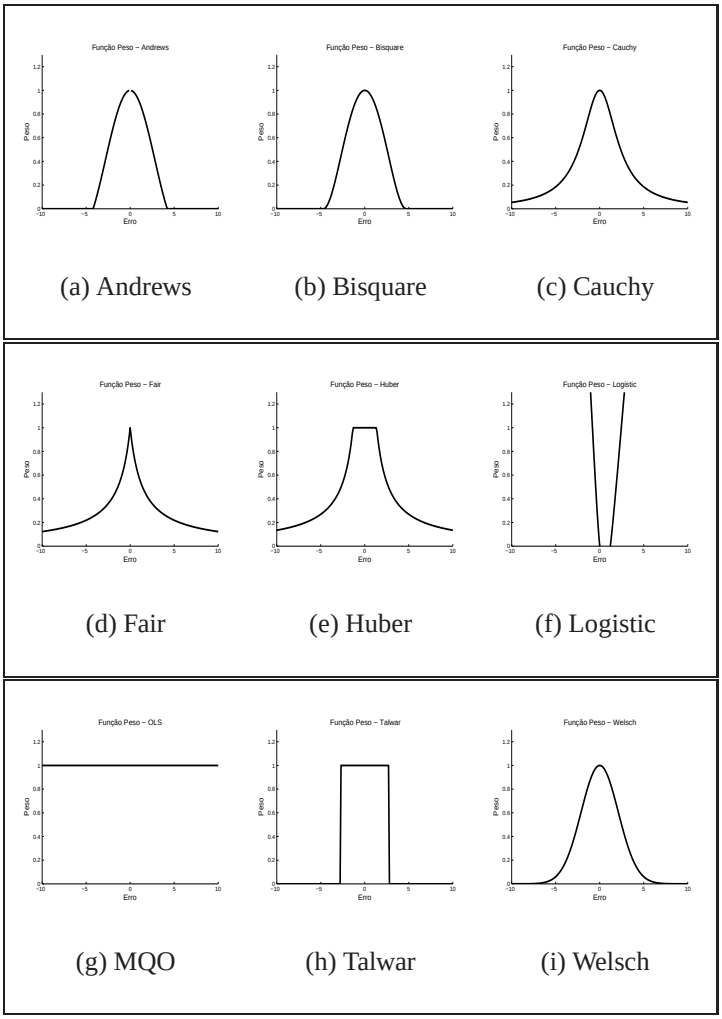
MQO, e $\hat{\sigma}$ é uma estimativa robusta do desvio padrão calculada como na Equação (3.13).

3.2.4 Exemplo Numérico: Regressão Linear Robusta

Nesta seção retomamos o exemplo apresentado no começo deste capítulo, no qual discutimos a influência da presença de *outliers* no processo de estimação dos parâmetros de um modelo de regressão linear simples. Aqui vamos incluir resultados da estimação usando o arcabouço de regressão robusta discutidos na Seção 3.2.1. Para isso, usaremos o comando `ROBUSTFIT` do Matlab.

Isto posto, vamos avaliar o estimador robusto para os casos sem e com a presença do *outlier* (dados dos EUA). Assim, a seqüência de comandos é dada por

```
» x=[480; 500; 380; 1100; 1100; 230; 490; 250; 300; 510; 1300];
» y=[180; 150; 170; 350; 460; 60; 240; 90; 110; 250; 200];
```



(a) Andrews (b) Bisquare (c) Cauchy

(d) Fair (e) Huber (f) Logistic

(g) MQO (h) Talwar (i) Welsch

Figura 3.4: Gráfico das funções peso mostradas na Tabela 3.4.

```
» Bs=robustfit(x(1:10),y(1:10)) % Usando dados SEM outlier
Bs =
6.9545
0.3737
» Bc=robustfit(x,y) % Usando dados COM outlier
Bc =
7.7198
0.3717
```

A função objetivo Bisquare é usada como *default* para o comando `ROBUSTFIT`, cujo valor *default* do limiar é 4,685, conforme mostrado na Tabela 3.5. Obviamente, é possivel especificar uma outra função objetivo dentre as nove listadas na Tabela 3.3, assim como um valor diferentes para o limiar. Por exemplo, usando a função Andrews com k=1,35, o comando passa a ser

```
» Bc=robustfit(x,y,'andrews',1.35) % Usando dados COM outlier
```

Nome	Função Peso	Limiar default
Andrews	$\begin{cases} r_i^{-1}\sin(r_i), & \lvert r_i\rvert \leq \pi \\ 0, & \lvert r_i\rvert > \pi \end{cases}$	1,339
Bisquare	$\begin{cases} (1-r_i^2)^2, & \lvert r_i\rvert < 1 \\ 0, & \lvert r_i\rvert \geq 1 \end{cases}$	4,685
Cauchy	$\frac{1}{(1+r_i^2)}$	2,385
Fair	$\frac{1}{(1+\lvert r_i\rvert)}$	1,400
Huber	$\frac{1}{\max(1,\lvert r_i\rvert)}$	1,345
Logistic	$\frac{\tanh(r_i)}{r_i}$	1,205
MQO	ausente	ausente
Talwar	$\begin{cases} 1, & \lvert r_i\rvert \leq 1 \\ 0, & \lvert r_i\rvert > 1 \end{cases}$	2,795
Welsch	$\exp(-(r_i^2))$	2,985

Tabela 3.5: Valores *default* do limiar e funções para cada estimador-*M* implementadas no comando `ROBUSTFIT` do Matlab.

```
Bc =
7.7241
0.3717
```

As retas de regressão e seus parâmetros estimados pelos métodos MQO e estimação-*M* usando os dados sem e com *outlier* estão compilados na Tabela 3.6.

Dados	Método	Parâmetro $\hat{\beta}_0$ (intercepto)	Parâmetro $\hat{\beta}_1$ (inclinação)	Reta de Regressão
1. Sem EUA	MQO	9,14	0,37	$\hat{y}_i = 9,14 + 0,37x_i$
2. Sem EUA	Estimador-*M*	6,95	0,37	$\hat{y}_i = 6,95 + 0,37x_i$
3. Com EUA	MQO	67,56	0,23	$\hat{y}_i = 67,56 + 0,23x_i$
4. Com EUA	Estimador-*M*	7,72	0,37	$\hat{y}_i = 7,72 + 0,37x_i$

Tabela 3.6: Resultado da estimação dos parâmetros da reta de regressão para os dados da Tabela 3.1 sem e com a informação dos EUA.

Os gráficos das duas retas de regressão para o caso sem *outlier* estão mostrados na Figura 3.5, enquanto para o caso com *outlier* as duas retas estão mostradas na Figura 3.6.

Uma rápida inspeção das Figuras 3.5 e 3.6 é suficiente para chegar à principal conclusão de que a reta cujos parâmetros foram estimados pelo método robusto praticamente não "sente" a presença do *outlier*, não tendo sua inclinação alterada. Já a reta estimada pelo método MQO tem sua inclinação fortemente modificada, sendo "atraída" pelo *outlier*. No cenário sem *outlier* as retas praticamente se superpõem, confirmando o fato de que estimadores-*M* tendem a preservar 95% da eficiência do estimador quando os erros são normais.

Para finalizar, foram coletadas para este experimento informações sobre os pesos atribuídos

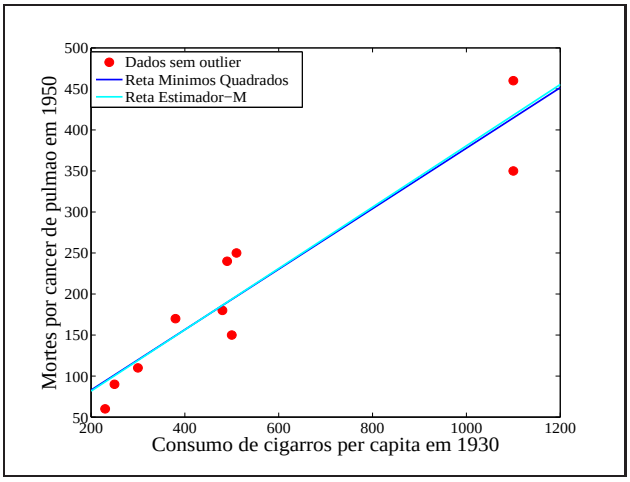


Figura 3.5: Gráfico das retas de regressão dos estimadores MQO e robusto ajustadas aos dados da Tabela 3.1 sem *outlier*.

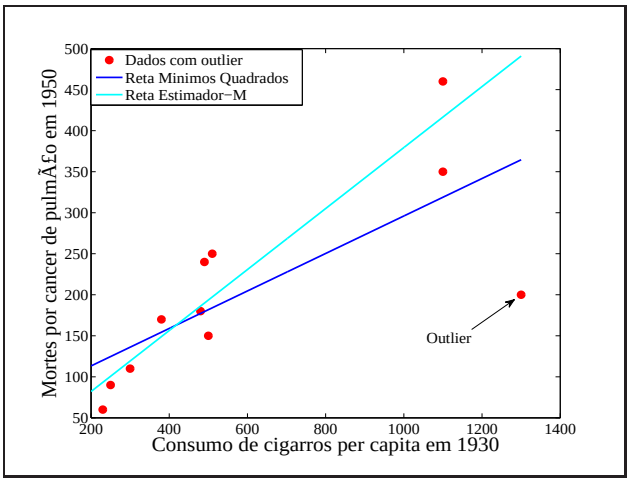


Figura 3.6: Gráfico das retas de regressão dos estimadores MQO e robusto ajustadas aos dados da Tabela 3.1 com *outlier*.

por cada um dos métodos de estimação para os pontos (x_i, y_i) na análise de regressão sem e com *outlier*. Os valores numéricos dos pesos atribuídos aos erros gerados estão mostrados na Tabela 3.7 e foram gerados, no caso do estimador-*M*, usando a função objetivo Bisquare com o valor default (k =4,685).

É possível perceber nesta tabela que o *outlier* exerce uma grande influência no MQO, devido ao seu resíduo (distância vertical do ponto y_i à reta $\hat{y}_i$) ser muito alto, enquanto que a influência dele no método robusto é nula.

País	Sem Outlier		Com Outlier	
	Peso (MQO)	Peso (Estimador-*M*)	Peso (MQO)	Peso (Estimador-*M*)
Islândia	0,2	0,97	0,19	0,98
Noruega	0,19	1	0,18	1
Suécia	0,16	1	0,16	1
Dinamarca	0,13	0,99	0,13	0,99
Austrália	0,1	1	0,1	1
Canadá	0,1	0,96	0,098	0,96
Holanda	0,1	0,95	0,1	0,95
Suíça	0,1	0,94	0,097	0,95
Finlândia	0,46	0,84	0,26	0,9
Grã-Bretanha	0,46	0,94	0,26	0,95
Estados Unidos	ausente	ausente	**0,43**	**0**

Tabela 3.7: Influência (peso) do resíduo na análise de regressão realizada pelos métodos MQO e estimação-*M*, para o conjunto de dados da Tabela 3.1, sem e com *outlier*.

3.3 Resumo do Capítulo

Esse capítulo apresentou os fundamentos de estatística robusta, que são métodos menos susceptíveis à presença de *outliers*. Os conceitos de estatística robusta, aplicados ao problema de regressão linear, conduzem à classe de estimadores conhecidos como *estimadores-M*. Tais estimadores conferem mais robustez ao problema de estimação dos parâmetros do modelo de regressão a partir da minimização de uma outra função dos erros, que não a soma dos quadrados dos erros, consequentemente, atribuem pesos diferentes aos erros gerados em um problema de regressão linear.

O método MQO utiliza a soma dos quadrados dos resíduos entre a resposta observada e a resposta estimada pelo modelo ajsutado, de forma tal que todos os erros têm o mesmo peso na soma, contribuindo igualmente para o resultado final. Este tipo de função objetivo, no entanto, não é eficiente na presença de *outliers*, os quais são pontos que estão distantes dos outros pontos do conjunto e, consequentemente, produzem erros maiores. Tais erros provocados por *outliers* exercem uma grande influência na estimação dos parâmetros, atraindo a reta de regressão e enviesando os resultados.

Uma observação que é substancialmente diferente de outras pode causar, portanto, uma grande diferença nos resultados da análise de regressão. *Outliers* ocorrem muito frequentemente em dados reais, e eles muitas vezes passam desapercebido, pois os dados são processados automaticamente, sem uma inspeção cuidadosa (BLATNÁ, 2006). A análise desses pontos, no entanto, merece uma atenção particular, pois a inclinação de uma reta de regressão linear simples é mais influenciada pelas observações $\mathbf{x}_i$ que têm o maior valor $||\mathbf{x}_i - \overline{\mathbf{x}}||$, em que $\overline{\mathbf{x}}$ denota o vetor médio das observações (TONG, 2010).

No método MQO, portanto, mesmo quando se aplica a condição de normalidade dos erros, seu comportamento fica comprometido na presença de *outliers*. Se *outliers* são gerados a partir

de algum tipo de erro durante o processo de medição, e não são extraídos de alguma distribuição estatística como as outras amostras, isso é um problema (RAO; TOUTENBURG, 1999). Uma reação comum a esse 'perigo' é a rejeição desses pontos, embora, em princípio, eles não devam ser descartados, podendo ser reservados para tratamento separado. Dessa forma, é importante ter uma solução que considere *outliers* e que, em vez de descartá-los, os trate apenas como pontos *fora da curva*, ou seja, como uma observação que não segue o mesmo padrão das outras amostras. Essa solução é o arcabouço estatístico de estimação de parâmetros conhecido como *regressão robusta* via estimadores-M.

No próximo capítulo vamos utilizar os conceitos de regressão robusta e estimadores-M para introduzir a primeira contribuição desta tese. Lá, iremos dar uma nova utilização e, quiçá, uma nova visão aos conceitos de regressão robusta, porém não mais aplicados a problemas de regressão, mais sim a problemas de clasificação de padrões. O resultado é o desenvolvimento de um classificador linear robusto.

4 *Proposta de um Classificador Linear Robusto*

Esse capítulo introduz inicialmente o classificador linear dos mínimos quadrados (*least-squares classifier*, LSC), que é um dos classificadores de padrões mais comuns na literatura (DUDA et al., 2006; WEBB, 2002). No campo de redes neurais artificiais, o classificador LSC possui a mesma estrutura e formulação matemática da regra de aprendizagem da rede OLAM (*Optimal Linear Associative Memory*), proposta por Kohonen & Ruohonen (1973), de modo que a denominação classificador OLAM será adotada de agora em diante.

Em seguida, analisaremos o desempenho do classificador OLAM em um problema de classificação binária sintético, a fim de ilustrar a influência de *outliers* no aprendizado do classificador OLAM e, consequentemente, no posicionamento da reta de decisão entre as classes.

Por fim, estabeleceremos conexão entre os conceitos de regressão robusta, principalmente com relação a estimadores-M, e classificação de padrões, como o intuito de adaptá-los ao projeto de uma versão robusta do classificador OLAM. Um exemplo numérico servirá como prova de conceito das idéias propostas.

4.1 Memória Associativa Linear Ótima

Aprender é a forma de adquirirmos conhecimento sobre o mundo ao nosso redor, e é através desse processo de aquisição de conhecimento que o ambiente nos torna conscientes de nossas respostas comportamentais. A aprendizagem nos permite armazenar e reter conhecimento; ela constrói nossas memórias (PRASAD et al., 2010).

Memória e aprendizagem estão intimamente ligadas. Quando um padrão particular de atividade é aprendido, ele é armazenado no cérebro, de onde ele pode ser recuperado mais tarde, quando necessário. Aprendizagem codifica informação. Um sistema aprende um padrão, se o sistema codifica o padrão na sua estrutura. A estrutura do sistema muda quando o sistema aprende a informação. Assim, a aprendizagem envolve mudança. Essa mudança pode ser re-

presentada na memória para comportamento futuro (PRASAD et al., 2010).

Uma memória associativa linear (*Linear Associative Memory*, LAM) é um algoritmo de aprendizado treinado para mapear entradas desejadas em saídas desejadas (EICHMANN; KASPARIS, 1989), através de um operador matricial.

Uma memória associativa é um modelo inspirado na forma com que o cérebro humano armazena e recorda informações por associação (MESQUITA, 2012). Estes modelos são projetados para armazenar um conjunto finito de associações $\{(\mathbf{x}_\mu, \mathbf{y}_\mu) : \mu = 1, \cdots, N\}$, em que $\mathbf{x}_\mu \subseteq \chi$ e $\mathbf{y}_\mu \in \gamma$ são, respectivamente, os padrões de entrada e saída a serem associados. Os conjuntos χ e γ englobam todos os possíveis itens memorizados. Em termos matemáticos, uma memória associativa corresponde a uma aplicação (i.e. mapeamento) $\Omega : \chi \to \gamma$ tal que $\Omega(\mathbf{x}_\mu) = \mathbf{y}_\mu$ para todo $\mu = 1, \cdots, N$.

Os primeiros algoritmos de memórias associativas lineares foram introduzidos em 1972, de forma independente, por Anderson (1972), Kohonen (1972) e Nakano (1972). Do ponto de vista matemático, as LAMs podem ser vistas como os modelos mais simples de memórias associativas, supondo que os conjuntos de todos os padrões de entrada e saída sejam $\chi = \mathbb{R}^n$ e $\gamma = \mathbb{R}^m$, e que a aplicação $\Omega : \chi \to \gamma$, que descreve a memória, seja linear.

Existem na literatura basicamente duas estratégias para determinar a matriz M, são elas (MESQUITA, 2012): armazenamento por correlação e armazenamento por projeção. O armazenamento por correlação é baseado no postulado de Hebb (1949), o qual pode ser descrito nos seguintes termos (BARRETO, 1998):

> "Quando um axônio da célula A está próximo o suficiente para excitar uma célula B e, repetida ou persistentemente, influenciar no seu disparo, algum processo de crescimento ou mudança metabólica acontece em uma ou ambas as células, tal que a eficiência de A em ser uma das células que dispara B é aumentada."

No contexto de redes neurais artificiais, o postulado de Hebb leva à regra de atualização de conexões conhecida como *regra de aprendizado de Hebb*. Matematicamente, a versão mais simples desta regra é escrita como

$$w_{ij}(t+1) = w_{ij}(t) + \eta y_i \cdot x_j, \tag{4.1}$$

em que w_{ij} é o peso sináptico que conecta a j-ésima entrada ao i-ésimo neurônio, t é o instante atual, $\eta > 0$ define o passo de aprendizagem, y_i é ativação do i-ésimo neurônio e x_j é a intensidade da j-ésima entrada da rede.

A regra de aprendizagem de Hebb na construção da matriz de pesos de conexão de uma memória associativa, portanto, leva a uma capacidade de memória significativamente baixa. Várias modificações e variações são propostas para maximizar a capacidade de memória, como é o caso do armazenamento por projeção (PRASAD et al., 2010). O armazenamento por projeção, proposto inicialmente por Kohonen & Ruohonen (1973), tem como objetivo determinar o melhor modelo LAM no sentido dos mínimos quadrados, sendo então chamado de Memória Associativa Linear Ótima (*Optimal Linear Associative Memory*, OLAM).

O modelo OLAM é um paradigma computacional bem conhecido de memória associativa. Como tal, a informação no modelo OLAM é armazenada de forma distribuída em um operador matricial, de modo que um vetor $\mathbf{y}_\mu$ armazenado pode ser recuperado através da especificação de toda ou de uma parte do padrão de entrada $\mathbf{x}_\mu$. O modelo OLAM tem a propriedade de disponibilizar recuperação rápida da informação, com desempenho tolerante a distorções (ruído) no padrão de entrada $\mathbf{x}_\mu$.

Contudo, o interesse no modelo OLAM para esta tese não é como um modelo de memória associativa, mas sim, como classificador de padrões. Neste contexto, pode-se entender o processo de classificação como um processo de associação de um dado padrão de entrada $\mathbf{x}_\mu$ com um vetor-código $\mathbf{y}_\mu$, vetor este que carrega uma representação numérica, chamada de rótulo (*label*), da classe a qual pertence o padrão de entrada.

Vale ressaltar aqui que, do ponto de vista teórico, o classificador OLAM é equivalente ao classificador de mínimos quadrados (DUDA et al., 2006; WEBB, 2002), e tem sido usado tanto como um classificador isolado (BARRETO; FROTA, 2013; EICHMANN; KASPARIS, 1989; KOHONEN; OJA, 1976), quanto como um bloco de construção de abordagens de classificação não-linear multicamadas, tais como funções de base radial (POGGIO; GIROSI, 1990), máquina de aprendizado extremo (ELM) (HUANG et al., 2011), e rede echo-state (ESN) (EMMERICH et al., 2010).

Uma descrição detalhada do modelo OLAM já no contexto de classificação de padrões é conduzida a seguir.

4.2 OLAM em Classificação de Padrões

Assumindo que N pares de dados $\{(\mathbf{x}_\mu, \mathbf{d}_\mu)\}_{\mu=1}^{N}$ estejam disponíveis para construção e avaliação do modelo, onde $\mathbf{x}_\mu \in \mathbb{R}^{p+1}$ é o μ-ésimo padrão de entrada[1] e $\mathbf{d}_\mu \in \mathbb{R}^K$ é o rótulo da classe alvo correspondente, com K denotando o número de classes. Para os rótulos, assumimos

[1] A primeira componente de $\mathbf{x}_\mu$ é igual a 1 para poder incluir o *bias* com parâmetro a ser estimado.

um esquema de codificação 1-de-K, ou seja, para cada vetor de rótulos $\mathbf{d}_\mu$, a componente cujo índice corresponde à classe do padrão $\mathbf{x}_\mu$ é definida como "+1", enquanto as outras $K-1$ componentes são definidas como "-1".

Então, primeiramente deve-se selecionar aleatoriamente N_1 ($N_1 < N$) pares de dados a partir do conjunto de dados disponível e os organizar ao longo das colunas das matrizes $\mathbf{D}$ e $\mathbf{X}$, como segue:

$$\mathbf{X} = [\mathbf{x}_1 \mid \mathbf{x}_2 \mid \cdots \mid \mathbf{x}_{N_1}] \quad \text{e} \quad \mathbf{D} = [\mathbf{d}_1 \mid \mathbf{d}_2 \mid \cdots \mid \mathbf{d}_{N_1}]. \tag{4.2}$$

em que $\dim(\mathbf{X}) = (p+1) \times N_1$ e $\dim(\mathbf{D}) = m \times N_1$. O objetivo é usar as matrizes $\mathbf{X}$ e $\mathbf{D}$ para obter o seguinte mapeamento linear:

$$\mathbf{D} = \beta \mathbf{X} \quad (\text{modo } \textit{batch}), \tag{4.3}$$

que pode também ser escrito em mapeamentos individuais do tipo

$$\mathbf{d}_\mu = \beta \mathbf{x}_\mu \quad (\text{modo padrão-a-padrão}), \tag{4.4}$$

para $\mu = 1, \ldots, N_1$. Para ambos os modos de operação, a dimensão da matriz β é $K \times (p+1)$.

A solução de mínimos quadrados ordinários (MQO) do sistema linear na Equação (4.3) é dada pela inversa generalizada Moore-Penrose, ou seja

$$\hat{\beta} = \mathbf{D}\mathbf{X}^T \left(\mathbf{X}\mathbf{X}^T\right)^{-1}, \tag{4.5}$$

em que o símbolo ($\wedge$) indica uma estimativa do operador matriz β. A solução de norma-mínima para Equação (4.3) é dada pela versão regularizada da Equação (4.5):

$$\hat{\beta} = \mathbf{D}\mathbf{X}^T \left(\mathbf{X}\mathbf{X}^T + \lambda \mathbf{I}\right)^{-1}, \tag{4.6}$$

em que $\mathbf{I}$ é a matriz identidade de dimensão $(p+1) \times (p+1)$ e λ é um parâmetro de regularização positivo muito pequeno.

É importante notar que o vetor de parâmetros $\hat{\beta}_i \in \mathbb{R}^{p+1}$, $i = 1, \ldots, m$, pode ser calculado individualmente por meio da seguinte equação:

$$\hat{\beta}_i = \left(\mathbf{X}\mathbf{X}^T\right)^{-1} \mathbf{X}\mathbf{D}_i^T, \tag{4.7}$$

em que o vetor $\mathbf{D}_i$ corresponde à i-ésima linha da matriz $\mathbf{D}$. O vetor estimado $\hat{\beta}_i$ deve ser interpretado como sendo o vetor de pesos (incluindo o *bias*) do i-ésimo neurônio de saída do classificador OLAM.

É importante ressaltar também as diferenças entre a Equação (4.7) e a Equação (2.29) do

modelo de regressão linear múltipla, a despeito da similaridade entre as fórmulas. Uma diferença entre elas diz respeito ao tipo de variável de saída. Na Equação (2.29), as variáveis de resposta são variáveis contínuas, enquanto na Equação (4.7), as variáveis de saída são discretas (i.e. pertencem ao conjunto $\{-1,+1\}$), pois a tarefa é de classificação.

Outra diferença entre as equações está no uso do índice i pela Equação (4.7) para indicar que é o vetor de parâmetros ajustáveis do i-ésimo neurônio. Isto é necessário porque em classificação o problema é formulado como MIMO, ou seja, como tendo múltiplas entradas e múltiplas saídas, mesmo para problemas de classificação binária.

Predição da Classe para um Padrão Desconhecido: Uma vez de posse da matriz de pesos $\hat{\beta}$ estimada, os $N_2 = N - N_1$ pares de dados restantes são usados para validar o modelo. Dessa forma, para o modo de recuperação padrão-a-padrão, a saída do classificador OLAM é dada por

$$\mathbf{y}_\mu = \hat{\beta}\mathbf{x}_\mu, \tag{4.8}$$

para $\mu = 1,\ldots,N_2$, enquanto para o modo *batch*, tem-se

$$\mathbf{Y} = \hat{\beta}\mathbf{X}. \tag{4.9}$$

O índice da classe predita i^*_μ para o μ-ésimo padrão de entrada de teste é então dado pela seguinte regra de decisão:

$$i^*_\mu = \arg\max_{i=1,\ldots,K}\{y_{i\mu}\} = \arg\max_{i=1,\ldots,K}\{\hat{\beta}_i^T\mathbf{x}_\mu\}, \tag{4.10}$$

em que $y_{i\mu} = \hat{\beta}_i^T\mathbf{x}_\mu$ é a i-ésima componente do vetor $\mathbf{y}_\mu$ calculado como na Equação (4.8), com o vetor $\hat{\beta}_i^T$ sendo a i-ésima linha da matriz $\hat{\beta}$.

4.3 O Classificador OLAM no Matlab

As Equações (4.5) e (4.6) usadas, respectivamente, para estimar a matriz de pesos do classificador OLAM e de sua versão regularizada, podem ser facilmente implementadas no ambiente Matlab. De fato, os comandos a serem mostrados a seguir são extensões naturais dos comandos usados no Capítulo 2.

Assumindo que os vetores de atributos $\mathbf{x}_\mu$, $\mu = 1,\ldots,N_1$, usados no treinamento do classificador OLAM estejam dispostos ao longo das colunas da matriz `X` e que os rótulos correspondentes estejam dispostos ao longo das colunas da matriz `D`, então a Equação (4.5) pode ser implementada da forma que se lê, ou seja,

```
» B = D*X'*inv(X*X');
```

em que `B` denota a estimativa MQO da matriz de pesos β. Contudo, esta forma de se estimar β não é recomendada por ter elevado custo computacional e por ser muito susceptível a erros numéricos. Neste caso, recomenda-se usar o operador *barra* ($/$), ou seja

```
» B = D/X;
```

Uma outra maneira de estimar a matriz β é através do comando `PINV`:

```
» B = D*pinv(X);
```

Para a versão regularizada do classificador OLAM, também é possível estimar $\hat{\beta}$ através da escrita direta da Equação (4.6) no prompt do Matlab:

```
» l = 0.01;
» I=ones(size(X*X'));
» B = D*X'*inv(X*X' + l*I);
```

Porém, pelas mesmas razões apontadas anteriormente, recomenda-se o uso do operador *barra* ($/$). Neste caso, a seqüência de comandos passa ser a seguinte:

```
» l = 0.01;
» I=eye(size(X*X'));
» A=X*X' + l*I;
» R=D*X';
» B = R/A;
```

Um exemplo de aplicação dos comandos acima em um problema de classificação de dados sintéticos é mostrado na próxima seção. O objetivo do experimento computacional é avaliar o efeito da presença de *outliers* no posicionamento da reta de decisão entre classes para o classificador OLAM.

4.4 Exemplo Numérico: Classificador OLAM

Nesta seção vamos avaliar o desempenho do classificador em um problema sintético de classificação binária projetado com o objetivo de mostrar a influência de *outliers* na definição

da reta de decisão entre duas classes linearmente separáveis. O experimento envolve uma base de dados sintéticos bidimensionais consistindo de $N = 120$ amostras (69 da classe +1, 51 da classe -1) mais $N_{out} = 5$ *outliers* da classe +1, que está disponível publicamente na internet[2]. Assim, para este problema, tem-se $p = 2$ atributos de entrada e $K = 2$ neurônios de saída, um para cada classe. Não foi utilizado *bias* para este experimento.

Nesse experimento o classificador OLAM é treinado duas vezes. Na primeira vez, ele é treinado com o conjunto de dados limpo, ou seja, livre de *outliers*. Na segunda vez, o classificador OLAM é treinado com N_{out} *outliers* adicionados ao conjunto original de dados. É importante mencionar que todas as amostras de dados são usadas para treinar os classificadores, já que o objetivo é visualizar a posição final da reta de decisão, e não, calcular taxas de erro do classificador.

No Matlab, assumindo que os vetores de atributos estão dispostos ao longo das colunas da matriz `X`, enquanto os rótulos correspondentes estão dispostos ao longo das colunas da matriz `D`, a estimativa MQO da matriz de pesos β para os dados sem *outliers* é computada como

```
» B = D/X
B =
0.0872 0.0087
-0.0872 -0.0087
```

Já para os dados com *outliers*, tem-se a seguinte estimativa:

```
» Bout = D/X
Bout =
0.0748 0.0226
-0.0748 -0.0226
```

Na Tabela 4.1 estão mostradas as funções discriminantes estimadas da classe +1 ($\hat{y}_1(x_1,x_2)$), da classe -1 ($\hat{y}_2(x_1,x_2)$) e reta de decisão ($\hat{y}_1 - \hat{y}_2 = 0$) resultante para o problema de classificação binário sintético, sem e com *outliers*.

Dados	Função Discriminante da Classe +1	Função Discriminante da Classe -1	Reta de Decisão
1. Sem *outliers*	$y_1(x_1,x_2) = 0,0872x_1 + 0.0087x_2$	$y_2(x_1,x_2) = -0,0872x_1 - 0.0087x_2$	$x_2 = -10,023x_1$
2. Com *outliers*	$y_1(x_1,x_2) = 0,0748x_1 + 0.0226x_2$	$y_2(x_1,x_2) = -0,0748x_1 - 0.0226x_2$	$x_2 = -3,310x_1$

Tabela 4.1: Resultado da aplicação do classificador OLAM ao problema de classificação binário sintético sem e com *outliers*.

[2] `www.deti.ufc.br/~guilherme/Codes/dataset1.dat`

Analisando a Tabela 4.1, percebe-se facilmente como a inclinação da reta de decisão do classificador OLAM foi bastante alterada com inclusão de apenas 5 *outliers*, mudando de $-10,023$ para $-3,310$. Os respectivos gráficos das retas de decisão para os cenários sem e com *outliers* estão mostrados na Figura 4.1. Esta figura só faz deixar mais claro que a inclusão de *outliers* mudou significativamente a inclinação da reta de decisão do classificador OLAM.

A principal conclusão deste experimento é que, conforme esperado, o classificador OLAM é bastante sensível à presença de *outliers* nos dados. Este fato serve de motivação para o desenvolvimento de uma versão robusta desse classificador.

4.5 Classificador OLAM Robusto

Estudos prévios avaliaram empírica e/ou teoricamente a robustez do modelo OLAM a padrões de entrada com ruído (CHERKASSKY et al., 1991; STILES; DENQ, 1987, 1985) em problemas de memória associativa. A principal conclusão apontada por esses trabalhos é que quando vetores de entrada são degradados (através de ruído), o modelo se torna extremamente sensível (i.e. instável) e seu erro de associação se torna inaceitavelmente grande. Alguns autores têm tratado essa limitação do modelo OLAM incluindo mecanismos não-lineares no modelo de memória associativa (HUNT et al., 1993) ou levando em consideração, diretamente no desenvolvimento do sistema, as propriedades do ruído (BAEK; OH, 2006).

Contudo, conforme já mencionado na Seção 4.1, o interesse no modelo OLAM para esta tese não é como um modelo de memória associativa, mas sim, como classificador de padrões. Isto posto, é interessante observar que em muitos problemas de classificação do mundo real, os próprios rótulos atribuídos aos vetores de dados são ruidosos. Existem tipicamente dois tipos de ruído nos rótulos (KIM; GHAHRAMANI, 2008):

1. Ruído próximo às fronteiras das classes geralmente ocorrem devido à dificuldade de rotular, de forma consistente, dados ambíguos.

2. Ruído longe das fronteiras das classes podem ocorrer devido a erros grosseiros na rotulação ou a erros grosseiros na medição dos atributos de entrada.

Independentemente do tipo de *outlier* que possa estar presente em um dado conjunto de dados de classificação, *outliers* são considerados, em geral, indesejáveis e sua remoção do conjunto de dados é feita sempre que possível (HAMPEL, 2001). Contudo, identificar *outliers* é, por si só, uma tarefa árdua, que exige muito cuidado e experimentação por parte do usuário (BEN-GAL, 2005), principalmente quando o problema é de classificação multiclasses.

Uma das técnicas mais comum para se eliminar supostos *outliers* consiste em calcular a distância de Mahalanobis de cada exemplo de uma classe ao centróide da respectiva classe (NETO; BARRETO, 2009). Se a distância for maior que certo limiar, o exemplo é considerado um *outlier*.

Matematicamente, considera-se que $\mathbf{x} \in \mathbb{R}^{p+1}$ é um *outlier* da *k*-ésima classe, $k = 1, \ldots, K$, se a seguinte condição for satisfeita:

$$\sqrt{(\mathbf{x} - \mu_k)^T \mathbf{C}_k^{-1} (\mathbf{x} - \mu_k)} > \gamma L \tag{4.11}$$

em que μ_k e $\mathbf{C}_k$ são, respectivamente, o vetor médio e a matriz de covariância da *k*-ésima classe, $0 < \gamma \leq 1$ é uma constante arbitrária e L é o valor crítico da distribuição Chi-quadrado com $p+1$ graus de liberdade e nível de significância α. Pode-se, por exemplo, variar γ até que uma certa porcentagem dos dados da *k*-ésima classe (e.g. 95%) esteja dentro do limite dado por γL.

Vale mencionar que nem sempre *outliers* são entes indesejáveis. De fato, alguns autores (SINGH; MARKOU, 2004; AUGUSTEIJN; FOLKERT, 2002; VASCONCELOS et al., 1995) até sugerem a inclusão de exemplos negativos como **outliers conhecidos** ou **outliers falsos** durante o processo de construção do modelo dos dados. Este tipo de *outlier* é construído, ou artificialmente ou mudando o rótulo de alguns exemplos negativos para positivo, passando estes a integrar o conjunto de dados positivos que será usado para construir o modelo estatístico dos dados.

A fim de permitir que o classificador OLAM trate *outliers* de forma automática e eficiente, propomos nesta tese o uso de estimadores-*M* descritos no Capítulo 3, por ser um arcabouço largamente usado para lidar com *outliers* em problemas de regressão linear múltipla. Esta é a principal contribuição desta tese, pois apesar do fato de estimadores-*M* gozarem de ampla utilização em problemas de regressão (HORATA et al., 2012; LI et al., 2012), sua aplicação a problemas de classificação supervisionada de padrões é inédita. Na realidade, ao longo do desenvolvimento desta tese, não foi encontrado um único artigo que combinasse o uso de estimação-*M* e classificadores de padrões, neurais ou estatísticos. Além disso, até onde se conseguiu averiguar, essa parece ser também a primeira vez que o desempenho do modelo OLAM como um classificador está sendo avaliado sob a presença de *outliers*.

Isto posto, considerando as fortes limitações do método MQO em estimar adequadamente a matriz β em cenários com *outliers*, nesta tese propomos o desenvolvimento de um classificador OLAM robusto, doravante denominado classificador ROLAM (*Robust OLAM classifier*), utilizando idéias oriundas da teoria de regressão robusta. A idéia geral é bem simples, porém, de grande impacto no desempenho do classificador OLAM: enquanto que no classificador

OLAM descrito na Seção 4.2, a matriz β é estimada através do método MQO usando a Equação (4.5) ou a Equação (4.6), no classificador ROLAM, a matriz β será estimada através do uso de estimadores-*M*. Os detalhes são apresentados a seguir.

4.5.1 Estimação-*M* para Classificação de Padrões Usando a Rede OLAM

Conforme já discutido no Capítulo 3, a principal razão do mau desempenho do MQO em cenários com *outliers* é que este critério atribui a mesma importância a todas as amostras de erro; ou seja, todos os erros contribuem da mesma forma para a solução final. Uma abordagem comum para lidar com *outliers* consiste em identificá-los e removê-los dos dados, para em seguida tentar o ajuste por mínimos quadrados. A abordagem de interesse para esta tese, conhecida como *regressão robusta*, contudo, usa métodos de estimação capazes de lidar automaticamente com conjuntos de dados que possuam *outliers*.

Nesta seção iremos revisitar o conceito de estimação-*M*, introduzido por Huber (HUBER, 1964), a fim de adaptá-lo a problemas de classificação de padrões. Neste capítulo trataremos de classificadores lineares, cujas fronteiras de decisão são modelos lineares (i.e. hiperplanos). Em particular, vamos desenvolver uma versão robusta do classificador OLAM. No próximo capítulo faremos o mesmo para uma categoria de classificadores neurais multicamadas baseados na rede ELM (*Extreme Learning Machine*) (HUANG et al., 2006).

Com base na teoria de Huber, um estimador-*M* geral, aplicado ao *i*-ésimo neurônio de saída do classificador OLAM, deve minimizar a seguinte função objetivo:

$$J(\beta_i) = \sum_{\mu=1}^{N_1} \rho(e_{i\mu}) = \sum_{\mu=1}^{N_1} \rho(d_{i\mu} - y_{i\mu}) = \sum_{\mu=1}^{N_1} \rho(d_{i\mu} - \beta_i^T \mathbf{x}_\mu), \tag{4.12}$$

em que a função $\rho(\cdot)$ calcula a contribuição de cada erro $e_{i\mu} = d_{i\mu} - y_{i\mu}$ para a função objetivo, $d_{i\mu}$ é o valor alvo do *i*-ésimo neurônio de saída para o μ-ésimo vetor $\mathbf{x}_\mu$, $\mu = 1, \ldots, N_1$, com N_1 sendo o número de padrões de treinamento. O vetor β_i é o vetor de pesos do *i*-ésimo neurônio de saída, $i = 1, \ldots, K$, em que K é o número de classes.

Seja $\psi = \rho'(\cdot)$ a derivada da função $\rho(\cdot)$ em relação ao resíduo $e_{i\mu}$. Diferenciando a função $\rho(\cdot)$ em relação ao vetor de pesos estimado $\hat{\beta}_i$, temos

$$\sum_{\mu=1}^{N_1} \psi(y_{i\mu} - \hat{\beta}_i^T \mathbf{x}_\mu)\mathbf{x}_\mu^T = \mathbf{0}, \tag{4.13}$$

em que $\mathbf{0}$ é um vetor nulo de dimensão $p+1$. Então, definindo a função peso $w(e_{i\mu}) =$

$\psi(e_{i\mu})/e_{i\mu}$, e fazendo $w_{i\mu} = w(e_{i\mu})$, as equações de estimação são dadas por

$$\sum_{\mu=1}^{n} w_{i\mu}(y_{i\mu} - \hat{\beta}_i^T \mathbf{x}_\mu)\mathbf{x}_\mu^T = \mathbf{0}. \tag{4.14}$$

Portanto, resolver as equações de estimação corresponde a solucionar um problema de mínimos quadrados ponderado, minimizando $\sum_\mu w_{i\mu}^2 e_{i\mu}^2$.

É importante lembrar que os pesos dependem dos resíduos (ou seja, erros estimados), os resíduos dependem dos coeficientes estimados, e os coeficientes estimados dependem dos pesos. Logo, teremos que adaptar o algoritmo IRLS (FOX, 1997) descrito no Capítulo 3, a um problema de classificação.

Isto posto, os passos do algoritmo IRLS, no contexto de treinamento do classificador OLAM usando a Equação (4.7) como referência, são descritos a seguir.

Algoritmo IRLS para Treinamento do Classificador OLAM

Passo 1 - Prover uma estimativa inicial $\hat{\beta}_i(0)$ usando a solução MQO em Equação (4.7).

Passo 2 - Em cada iteração t, calcular os resíduos a partir das iterações anteriores $e_{i\mu}(t-1)$, $\mu = 1,\ldots,N_1$, associados com o i-ésimo neurônio de saída, e então calcular os pesos correspondentes $w_{i\mu}(t-1) = w[e_{i\mu}(t-1)]$.

Passo 3 - Obter uma nova estimativa de mínimos quadrados ponderados para $\beta_i(t)$:

$$\hat{\beta}_i(t) = \left[\mathbf{X}\mathbf{W}_i(t-1)\mathbf{X}^T\right]^{-1}\mathbf{X}\mathbf{W}_i(t-1)\mathbf{D}_i^T, \tag{4.15}$$

em que $\mathbf{W}_i(t-1) = \text{diag}\{w_{i\mu}(t-1)\}$ é uma matriz de pesos $N_1 \times N_1$. Repetir Passos 2 e 3 até a convergência do vetor de coeficientes estimados $\hat{\beta}_i(t)$.

Várias funções objetivos $\rho(e_{i\mu})$ e de peso $w_{i\mu} = w(e_{i\mu})$ podem ser usadas, tais como aquelas mostradas nas Tabelas (3.3) e (3.4). Apenas a título de ilustração, a função de peso de Huber é dada por

$$w(e_{i\mu}) = \begin{cases} \frac{k}{|e_{i\mu}|}, & \text{se } |e_{i\mu}| > k \\ 1, & \text{caso contrário.} \end{cases} \tag{4.16}$$

em que o parâmetro k é uma constante (limiar) positiva e $|\cdot|$ é operador valor absoluto. Valores pequenos de k conferem mais robustez a *outliers*, mas a um custo de menor eficiência quando os erros são normalmente distribuídos. Em particular, $k = 1.345\hat{\sigma}$ para a função Huber, em que $\hat{\sigma}$ é uma estimativa robusta do desvio padrão dos erros[3].

[3]uma abordagem comum é fazer $\hat{\sigma} = \text{MAR}/0.6745$, em que *MAR* é mediana dos valores absolutos dos resíduos.

Em suma, a ideia básica da abordagem proposta é muito simples: substituir a estimação MQO da matriz de pesos $\hat{\beta}$ descrita na Equação (4.5), por uma estimação que combine o uso do arcabouço teórico da estimação-*M* e o algoritmo IRLS. A partir de agora, nos referiremos à abordagem proposta como classificador *OLAM Robusto* ou pela sigla ROLAM (do inglês *Robust OLAM*).

4.6 Classificador ROLAM no Matlab

O modo mais fácil e rápido de se implementar o classificador ROLAM no Matlab é usando o comando *ROBUSTFIT*. Para problemas de classificação binária, basta organizar os vetores de atributos de treinamento ao longo das colunas da matriz `X` e os respectivos rótulos ao longo das colunas da matriz `D`.

Note que em classificação binária a matriz `D` reduz-se a um vetor-linha no qual cada componente assume o valor +1, para uma das classes (geralmente chamada de *classe positiva*), e -1, para padrões da outra classe (chamada de *classe negativa*). Um exemplo hipotético é dado abaixo para um problema de classificação binária.

Suponha que temos apenas 8 padrões de treinamento, cada padrão tem dimensão 3 (i.e. possui 3 atributos), sendo que 4 padrões são da classe positiva e 4 da classe negativa. Assim, a sequência de comandos permite estimar o vetor de pesos do único neurônio de saída:

```
» X = rand(3,8);
» D = [1 1 1 1 -1 -1 -1 -1];
» B=robustfit(X',D)
B =
0.9306
3.1752
-1.0708
-3.6318
```

em que foi usada a função objetivo *default* (i.e. Bisquare). Note que o comando *ROBUSTFIT* adiciona automaticamente uma linha de 1's à matriz `X`; por isso, o vetor `B` tem 4 componentes.

É importante ressaltar que o comando *ROBUSTFIT* foi desenvolvido para estimar parâmetros de modelos MISO (*multi-input/single-output*); ou mais especificamente para problemas com múltiplas entradas e uma única saída. Caso o problema de classificação seja formulado

como um problema MIMO (i.e. multiclasses), então o comando *ROBUSTFIT* não funcionará[4].

Este problema, porém, pode ser contornado se calcularmos o vetor de pesos de cada neurônio separadamente. A solução, neste caso, é dada pela seguinte seqüência de comandos:

```
» X = rand(3,8);
» D = [1 1 1 1 -1 -1 -1 -1;-1 -1 -1 -1 1 1 1 1]
D =
1 1 1 1 -1 -1 -1 -1
-1 -1 -1 -1 1 1 1 1
» B1=robustfit(X',D(1,:))
B1 =
1.1729
-0.3077
-1.8530
-0.6803
» B2=robustfit(X',D(2,:))
B2 =
-1.1729
0.3077
1.8530
0.6803
```

Usaremos esta segunda abordagem no exemplo numérico a ser descrito na próxima seção.

Para finalizar, é muito útil mencionar que o comando `ROBUSTFIT` mostra um mensagem automática de erro se o número de padrões de treinamento (N_1) for menor ou igual à dimensão do vetor de entrada $\mathbf{x}_\mu$; ou seja, o comando acusa erro se $N_1 \leq \dim(\mathbf{x}_\mu) + 1$. Um exemplo hipotético no Matlab é dado a seguir, em que temos apenas 4 padrões de treinamento (2 da classe +1, 2 da classe -1), cada padrão de dimensão 3.

```
» X = rand(3,4);
» D = [1 1 -1 -1];
» B=robustfit(X',D)
???  Error using ==> statrobustfit at 21
```

[4]Até problemas de classificação binária costumam ser formulados como problemas MIMO. Para isso, basta definir o rótulo da classe +1 como $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$, e o rótula da classe -1 como $\begin{pmatrix} -1 \\ 1 \end{pmatrix}$.

```
Not enough points to perform robust estimation.

Error in ==> robustfit at 106
[varargout:]  = statrobustfit(X,y,wfun,tune,wasnan,doconst);
```

Embora tais situações sejam raras em problemas regressão, em reconhecimento de padrões é muito comum se ter poucos padrões de alta dimensionalidade. É o caso, por exemplo, de certas aplicações em Processamento de Imagens (RAMALHO; MEDEIROS, 2006) e Bioinformática (BOULESTEIX, 2004). Nestas situações, uma possibilidade consiste em localizar e comentar a linha de controle de dimensões diretamente no arquivo *robustfit.m*. Este procedimento será usado nesta tese, principalmente na avaliação do classificador não-linear robusto que será introduzido no próximo capítulo.

4.7 Exemplo Numérico: Classificador ROLAM

Nesta seção, os desempenhos dos dois classificadores, OLAM e ROLAM, são agora comparados entre si no mesmo problema da Seção 4.4. Para este fim, a função objetivo *Bisquare* foi usada para implementar o classificador ROLAM e a constante de regularização necessária à implementação do classificador OLAM usando o estimador dos mínimos quadrados regularizado foi definida em $\lambda = 10^{-2}$. O limiar utilizado foi o default da função `ROBUSTFIT` do Matlab para a função objetivo *Bisquare* (i.e k=4,685).

Para avaliar as posições finais das retas de decisão dos classificadores OLAM e ROLAM na presença de *outliers*, foram adicionados $N_{out} = 5$ *outliers* ao conjunto de dados, os quais foram rotulados como pertencentes à classe $+1$. Os *outliers* foram posicionados propositadamente longe da fronteira da classe encontrada para o caso sem *outliers*; mais especificamente, na região de decisão da classe -1.

Os resultados para o treinamento sem *outliers* são apresentados na Figura 4.2a, na qual, como esperado, as retas de decisão de ambos os classificadores coincidem. Os resultados para o treinamento com *outliers* são apresentados na Fig. 4.2b, sendo que dessa vez, a reta de decisão do classificador OLAM moveu-se em direção aos *outliers*, enquanto que a reta de decisão do classificador ROLAM parmaneceu praticamente na mesma posição, confirmando então a robustez da abordagem proposta a *outliers*.

4.8 Resumo do Capítulo

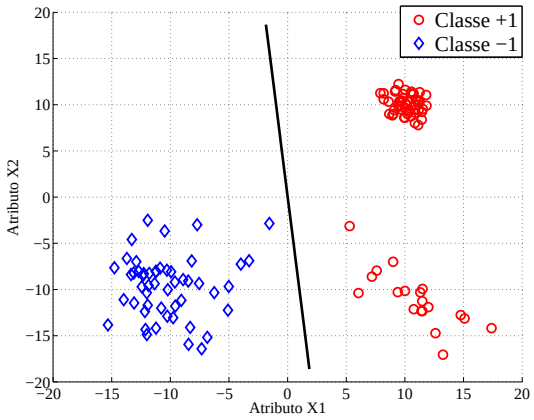
Neste capítulo foi introduzido um classificador linear robusto baseado na rede OLAM (*Optimal Linear Associative Memory*), que é uma arquitetura de rede neural originalmente proposto como modelo linear de memória associativa. Quando utilizada para classificador de padrões, a rede OLAM reduz-se ao classificador linear dos mínimos quadrados (*least-squares classifier*, LSC) (DUDA et al., 2006; WEBB, 2002).

A matriz de pesos do classificador OLAM é estimada por meio do método dos mínimos quadrados ordinário (MQO), cuja implementação requer o uso da matriz pseudoinversa, ou inversa generalizada de Moore-Penrose (GOLUB; VAN LOAN, 1996). Esse método, no entanto, apresenta desvantagens quando os dados a serem analisados envolvem *outliers*. A solução do método de mínimos quadrados é bastante influenciada por *outliers*, o que torna o posicionamento do hiperplano de decisão muito sensível a tais amostras discrepantes.

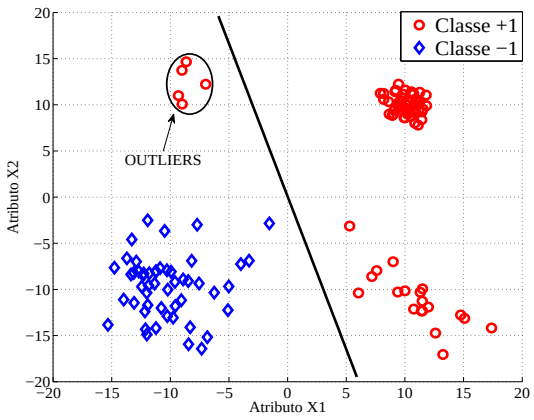
Isto posto, a principal contribuição deste capítulo à presente tese, envolve justamente a proposta de um classificador OLAM robusto a *outliers*, usando como base os conceitos de regressão robusta apresentados no Capítulo 3, principalmente o de estimadores-M. A ideia básica da abordagem proposta é muito simples, pois parte do pressuposto de substituir a estimação MQO da matriz de pesos $\hat{\beta}$ descrita na Equação (4.5), por uma estimação que combine o uso do arcabouço de estimação-M e o algoritmo IRLS. Ao novo classificador, deu-se o nome de classificador *OLAM Robusto* (ROLAM, do inglês *Robust OLAM*).

Um experimento numérico que serviu como prova de conceito, verificou que, na presença de *outliers*, a reta de decisão gerada pelo classificador OLAM move-se em direção aos *outliers*, enquanto que a reta definida pelo classificador ROLAM, permanece na mesma posição, comprovando a maior robustez do método proposto a *outliers* em relação ao classificador OLAM. Mais resutlados experimentais com o classificador OLAM serão apresentados no Capítulo 6.

No próximo capítulo será apresentada a segunda proposta desta tese, a saber, uma versão robusta de um classificador não-linear de padrões baseado em uma arquitetura neural recente.

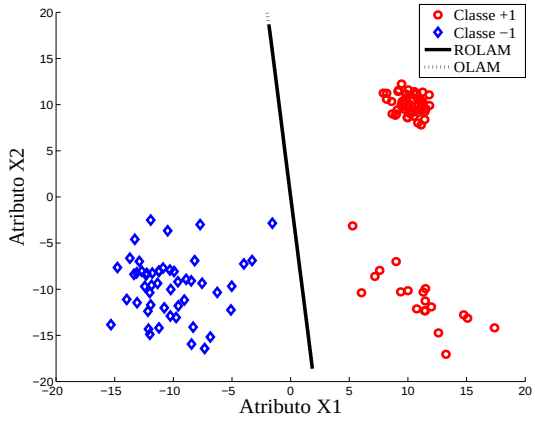


(a) Conjunto de dados sem *outliers*.

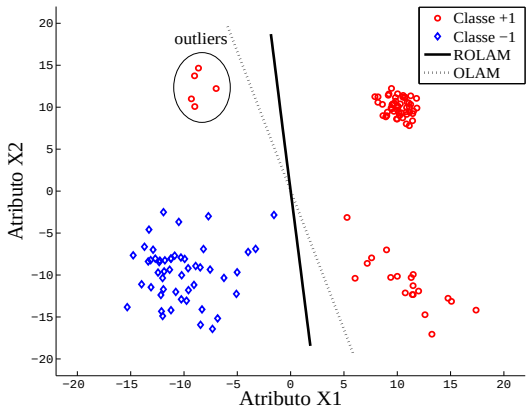


(b) Conjunto de dados com *outliers*.

Figura 4.1: Retas de decisão do classificador OLAM padrão para (a) Conjunto de dados sem *outliers*; (b) Conjunto de dados com *outliers*.



(a) Conjunto de dados sem *outliers*.



(b) Conjunto de dados com *outliers*.

Figura 4.2: Retas de decisão dos classificadores lineares OLAM e ROLAM. (a) Conjunto de dados sem *outliers*. (b) Conjunto de dados com *outliers*.

5 *Proposta de um Classificador Não-Linear Robusto*

Neste capítulo é apresentada a segunda proposta desta tese, a saber, uma arquitetura neural para classificação não-linear, robusta a *outliers*. O método tem como base uma rede neural multicamadas, conhecida como rede ELM (*Extreme Learning Machine*), e a teoria de estimadores-*M*, discutida no Capítulo 3.

Antes da apresentação da proposta serão apresentados os fundamentos da rede neural ELM, que usa o critério de mínimos quadrados ordinários (MQO) para estimar a matriz de pesos de saída, bem como um algoritmo recente de treinamento da rede ELM, conhecido como Plasticidade Intrínseca (*Intrinsic Plasticity*, IP), que será utilizado mais adiante para comparação com o método proposto.

Ao longo do capítulo são apresentados exemplos de simulação cujo os objetivos são avaliar, como prova de conceito, a robustez dos classificadores não-lineares descritos neste capítulo à presença de *outliers* no conjunto de dados, verificando o efeito que amostras discrepantes exercem sobre a posição da curva de decisão dos classificadores envolvidos.

5.1 Rede ELM: Fundamentos

Nos últimos anos, se tem visto um crescente interesse em uma classe de redes neurais supervisionadas, com uma única camada oculta, chamada *Máquina de Aprendizado Extremo* ou simplesmente rede ELM, proposta por Huang et al. (2006). 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 et al., 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 et al., 2012;

MOHAMMED et al., 2011; ZONG; HUANG, 2011; MICHE et al., 2011, 2010; LIU; WANG, 2010; DENG et al., 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 $\{(\mathbf{x}_\mu, \mathbf{d}_\mu)\}_{\mu=1}^{N}$ estejam disponíveis para construir e avaliar o modelo, em que $\mathbf{x}_\mu \in \mathbb{R}^{p+1}$ é o μ-ésimo padrão de entrada[1] e $\mathbf{d}_\mu \in \mathbb{R}^K$ é o rótulo da classe alvo correspondente, com K denotando o número de classes. Para os rótulos, assumimos um esquema de codificação 1-de-K, ou seja, para cada vetor de rótulos $\mathbf{d}_\mu$, a componente cujo índice corresponde à classe do padrão $\mathbf{x}_\mu$ é definida como "+1", enquanto as outras $K-1$ componentes são definidas como "-1".

Selecionemos aleatoriamente, então, N_1 ($N_1 < N$) pares de dados a partir do conjunto de dados disponível e os organizemos em colunas das matrizes $\mathbf{D}$ e $\mathbf{X}$, como segue:

$$\mathbf{X} = [\mathbf{x}_1 \mid \mathbf{x}_2 \mid \cdots \mid \mathbf{x}_{N_1}] \quad \text{e} \quad \mathbf{D} = [\mathbf{d}_1 \mid \mathbf{d}_2 \mid \cdots \mid \mathbf{d}_{N_1}]. \tag{5.1}$$

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

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

$$y_{i\mu} = \beta_i^T \mathbf{h}_\mu, \tag{5.2}$$

em que $\beta_i \in \mathbb{R}^q$, $i = 1, \ldots, K$, é o vetor peso conectando os neurônios ocultos ao i-ésimo neurônio de saída, e $\mathbf{h}_\mu \in \mathbb{R}^q$ 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

$$\mathbf{h}_\mu = [f(\mathbf{m}_1^T \mathbf{x}_\mu + b_1) \;\; f(\mathbf{m}_2^T \mathbf{x}_\mu + b_2) \;\; \cdots \;\; f(\mathbf{m}_q^T \mathbf{x}_\mu + b_q)]^T, \tag{5.3}$$

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 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

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

saída da camada oculta $\mathbf{h}_\mu \in \mathbb{R}^q$, $\mu = 1,\ldots,N_1$, 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 $K \times N_1$ cuja μ-ésima coluna é o vetor alvo (desejado) $\mathbf{d}_\mu \in \mathbb{R}^K$ associado com o padrão de entrada $\mathbf{x}_\mu$, $\mu = 1,\ldots,N_1$. Finalmente, seja β uma matriz $K \times q$, cuja i-ésima linha é o vetor de pesos $\beta_i^T \in \mathbb{R}^q$, associado ao i-ésimo neurônio de saída, $i = 1,\ldots,K$.

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

$$\mathbf{D} = \beta\mathbf{H}, \tag{5.4}$$

em que as matrizes $\mathbf{D}$ e $\mathbf{H}$ são conhecidas, enquanto a matriz de pesos β não. A solução baseada no critério MQO do sistema linear da Eq. (5.4) é dada pela inversa generalizada de Moore-Penrose (GOLUB; VAN LOAN, 1996) como

$$\beta = \mathbf{D}\mathbf{H}^T\left(\mathbf{H}\mathbf{H}^T\right)^{-1}. \tag{5.5}$$

A expressão mostrada na Eq. (5.5) pode ser dividida em K equações de estimação individuais, uma para cada neurônio de saída i, ou seja

$$\beta_i = \left(\mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{H}\mathbf{D}_i^T, \qquad i = 1,\ldots,K, \tag{5.6}$$

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

Em vários problemas do mundo real, é sabido que a matriz $\mathbf{H}\mathbf{H}^T$ $(q \times q)$ pode tornar-se singular, impossibilitando o uso da Eq. (5.5). Na verdade, uma matriz $\mathbf{H}\mathbf{H}^T$ quase singular (ainda inversível) já é também um problema, pois pode levar a resultados numericamente instáveis. Para evitar ambos os problemas, uma abordagem comum envolve o uso do 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 da matriz β:

$$\beta = \mathbf{D}\mathbf{H}^T\left(\mathbf{H}\mathbf{H}^T + \lambda\mathbf{I}\right)^{-1}. \tag{5.7}$$

em que $\lambda > 0$ é a constante de regularização e $\mathbf{I}$ é a matriz identidade de dimensões $q \times q$. De modo semelhante, a versão regularizada da Equação (5.6) é dada por

$$\beta_i = (\mathbf{H}\mathbf{H}^T + \lambda\mathbf{I})^{-1}\mathbf{H}\mathbf{D}_i^T, \qquad i = 1,\ldots,K. \tag{5.8}$$

5.1.1 Plasticidade Intrínseca

Usualmente, os desenvolvimentos teóricos e aplicações de modelos de redes neurais em regressão e classificação de padrões tentam explorar a plasticidade sináptica, ou seja, a adaptação dos enlaces (sinapses) entre as camadas, através da adaptação dos pesos entre as camadas oculta e de entrada, ou camadas oculta e de saída. Os problemas quase sempre têm o foco em encontrar melhores formar de calcular e ajustar esses pesos, mas dão pouca ou nenhuma importância à adaptação dos parâmetros da função de ativação do neurônio propriamente dita.

De acordo com Li (2011), teorias contemporâneas de aprendizagem e memória tratam quase que exclusivamente da plasticidade sináptica. Tais teorias postulam que a memória é armazenada nos pesos sinápticos e aprendizagem é o processo de alterar esses pesos. Por outro lado, existe um mecanismo, chamado Plasticidade Intrínseca (*Intrinsic Plasticity*) (CUDMORE; DESAI, 2008), que trata de alterações no neurônio propriamente dito, que pode ter grande influência na aprendizagem e memória.

Segundo Cudmore & Desai (2008), a plasticidade intrínseca é a modificação persistente das propriedades elétricas de um neurônio por atividade neural ou sináptica. Tal plasticidade é mediada por mudanças no nível de expressão ou propriedades biofísicas dos canais de íons da membrana celular, e podem afetar processos tão diversos quanto, integração sináptica, propagação sublimiar de sinais, geração de potenciais de ação, retropropagação de potenciais de ação e meta-plasticidade (TRIESCH, 2005).

Ainda segundo Cudmore & Desai (2008), a função da plasticidade intrínseca em animais é desconhecida em sua totalidade, mas há evidências experimentais que a atribuem vários papéis distintos: como parte do engrama de memória em si, como reguladora da plasticidade sináptica subjacente ao aprendizado e à memória, e como uma componente da regulação homeostática.

Assim, é importante enfatizar que a plasticidade intrínseca é diferente da plasticidade sináptica. Esta envolve mudanças na sinapse entre dois neurônios em vez de mudanças nas propriedades elétricas dentro de um único neurônio.

Matematicamente falando, a plasticidade intrínseca é a plasticidade pensada do ponto de vista do neurônio através de sua função de ativação $f(\cdot)$ e seus parâmetros. Modelos teórico-computacionais de plasticidade intrínseca tentam fazer com que a distribuição da saída de um neurônio oculto siga uma distribuição aproximadamente exponencial (TRIESCH, 2005). Distribuições exponenciais são usadas porque tais distribuições têm sido observadas em neurônios do cortex visual e, considerando um nível fixo de custo metabólico, esse tipo de função permite a transmissão da quantidade máxima de informação pelo neurônio (BADDELEY et al., 2005).

Neumann & Steil (2013) propõem um método chamado *Batch Intrinsic Plasticity* (BIP) para treinar redes ELM seguindo os conceitos de plasticidade intrínseca. Tal como o método proposto por Triesch (2005), o método BIP também tenta obter certas distribuições de saída para os neurônios ocultos. A diferença entre ambos é que a fase de estimação dos vetores de pesos $\mathbf{m}_l$, $l = 1, \ldots, q$, no BIP, trabalha em modo *batch*; ou seja, os padrões são apresentados todos de uma vez e os vetores $\mathbf{m}_l$ são determinados de uma só vez.

Um ponto importante a ser considerado ao usar ELM é a inicialização aleatória dos pesos de entrada. Dependendo desses valores iniciais, ELM pode apresentar uma boa generalização ou não pois, de acordo com Neumann & Steil (2013), essa inicialização aleatória pode levar ao problema de neurônios saturados ou quase lineares. No entanto, isso pode ser evitado encontrando funções de ativação que estejam em um regime favorável, isto é, que maximizem a transmissão de informações, tal como as funções exponenciais.

Tanto para o método iterativo inicialmente proposto por Triesch (2005) quanto para o método BIP de Neumann & Steil (2013), a forma de obter a distribuição desejada, no caso a distribuição exponencial, é através do ajuste dos parâmetros da função de ativação do l-ésimo neurônio oculto, mais especificamente, de sua inclinação (a_l) e de seu limiar b_l. A saída da função de ativação para cada neurônio oculto deve obedecer uma distribuição exponencial. No método BIP, esse processo de associação é feito todo de uma vez, em vez de um por um, tal como no método de Triesch (2005).

Grosso modo, o método BIP é então projetado da seguinte forma. Um problema de regressão linear é formulado e valores-alvos são obtidos a partir da distribuição de saída desejada que, no caso deste trabalho, foi a distribuição exponencial. Cada um desses valores-alvo deve corresponder a um dos neurônios ocultos. Dessa forma, a cada neurônio oculto está associado um valor de saída derivado de uma função exponencial. Os valores, ativações geradas pelos neurônios ocultos e valores-alvo, são então dispostos em ordem ascendente, e o objetivo é encontrar o par, inclinação a_l e bias b_l, para a função de ativação do l-ésimo neurônio oculto, a partir da relação entre a pseudoinversa das ativações e a função inversa dos valores-alvo. O método é não-supervisionado, pois apenas os padrões de entrada são usados para o treinamento, enquanto que os valores-alvos são obtidos a partir da função de saída desejada. O pseudocódigo do algoritmo BIP é mostrado a seguir, para o qual se assume que os vetores de pesos dos neurônios ocultos da rede ELM, $\mathbf{m}_l$, $l = 1, \ldots, q$, foram iniciados aleatoriamente.

entrada: $\mathbf{X} = [\mathbf{x}_1 \ \mathbf{x}_2 \ \cdots \ \mathbf{x}_{N_1}]^T$;
início
  para *cada neurônio oculto* **faça**
    calcular as ativações $s_l(t) = \mathbf{m}_l^T \mathbf{x}(t), t = 1, \ldots, N_1$;
    agrupá-las no vetor $\mathbf{s}_l = [s_l(1) \ s_l(2) \ \cdots \ s_l(N_1)]^T$;
    obter os valores-alvos $\mathbf{d} = [d_1 \ d_2 \ \cdots \ d_{N_1}]^T$ a partir da distribuição desejada f_{des};
    ordenar os vetores-alvos $\mathbf{d} \leftarrow \text{sort}(\mathbf{d})$ e as ativações $\mathbf{s}_l \leftarrow \text{sort}(\mathbf{s}_l)$;
    construir $\Phi(\mathbf{s}_l) = (\mathbf{s}_l^T, (1...1)^T)$;
    calcular pseudoinversa $\mathbf{v}_l = (a_l, b_l)^T = \Phi(\mathbf{s}_l)\dagger . f^{-1}(\mathbf{d})$;
  fim
  retornar $\mathbf{V} = [\mathbf{v}_1 \ \mathbf{v}_2 \ \cdots \ \mathbf{v}_q]^T$;
  incorporar as inclinações e limiares em $\mathbf{V}$ na rede ELM;
fim

Algorithm 1: Algoritmo *Batch Intrinsic Plasticity*

5.2 O Classificador ELM no Matlab

As Equações (4.5) e (4.6) usadas, respectivamente, para estimar a matriz de pesos do classificador ELM e de sua versão regularizada, podem ser facilmente implementadas no ambiente Matlab. De fato, os comandos a serem mostrados a seguir são extensões naturais dos comandos usados no Capítulo 4.

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 seqüência de comandos que leva à determinação da matriz $\mathbf{H}$ e à estimação da matriz de pesos de saída β via Equação (5.5) 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 MQO da matriz de pesos β. Como já mencionado em capítulos anteriores, esta forma de se estimar β não é recomendada por ter elevado custo computacional

e por ser muito susceptível a erros numéricos. Neste caso, recomenda-se usar o operador *barra* ($/$), ou seja

```
» B = D/H;
```

Uma outra maneira de estimar a matriz β é através do comando `PINV`:

```
» B = D*pinv(H);
```

Para a versão regularizada do classificador ELM, também é possível estimar $\hat{\beta}$ através da escrita direta da Equação (5.7) no prompt do Matlab:

```
» l = 0.01;
» I=ones(size(H*H'));
» B = D*H'*inv(H*H' + l*I);
```

Porém, pelas mesmas razões apontadas anteriormente, recomenda-se o uso do operador *barra* ($/$). Neste caso, seqüência de comandos passa ser a seguinte:

```
» l = 0.01;
» I=eye(size(H*H'));
» A=H*H' + l*I;
» R=D*X';
» B = R/A;
```

5.3 Classificador ELM Robusto

Como mostrado no exemplo anterior, no que diz respeito à robustez da ELM a *outliers* em problemas de classificação, não há ainda uma abordagem abrangente, baseada em um arcabouço teórico sólido, tal como a de estimadores-*M*. Tendo isso em mente, assim como foi feito para o classificador linear no Capítulo 4, propõe-se aqui o uso de estimadores-*M* para calcular a matriz de pesos de saída da rede ELM, em vez da abordagem clássica via MQO.

Nos últimos anos, tem-se observado um interesse crescente no desenvolvimento de arquiteturas de redes neurais que sejam robustas a *outliers*, incluindo propostas para o projeto de redes RBF (LEE et al., 2009, 1999), redes de ecos de estado (LI et al., 2012; BOCCATO, 2013) e até mesmo redes ELM (HORATA et al., 2012). Contudo, os trabalhos envolvendo a rede ELM

mencionados anteriormente não abordam pontos importantes relacionados ao desempenho do modelo na presença de *outliers* nos dados, com o trabalho de Horata et al. (2012) sendo a única exceção encontrada.

Vale ressaltar que todos os trabalhos anteriores, sem exceção desta vez, abordaram a questão da robustez a *outliers* para problemas de regressão, tais como aproximação de função e predição de séries temporais. No entanto, em muitos problemas de classificação de padrões do mundo real, os dados e os rótulos providos para as amostras de dados são ruidosos. Mesmo assim, ao longo da pesquisa bibliográfica realizada para o desenvolvimento desta tese não foi encontrado um único artigo sequer que avaliasse a robustez de modelos de redes neurais, especialmente as redes baseadas em ELM, em problemas de classificação de padrões na presença de *outliers*.

Existem tipicamente dois tipos de ruído em rótulos (KIM; GHAHRAMANI, 2008). Ruído próximo às fronteiras das classes, que muitas vezes ocorrem porque é difícil rotular de forma consistente pontos de dados ambíguos. Erros de rotulação longe das fronteiras das classes, que podem ocorrer por causa de enganos na rotulação ou erros grosseiros na medição dos atributos de entrada.

Os métodos MQR (*mínimos quadrados regularizado*) e BIP para treinamento da rede ELM introduzem melhorias em relação ao método de estimação baseado no critério MQO; porém, nenhum destes métodos trata da questão da robustez em relação a conjunto de dados com *outliers* para classificação de padrões. Isto posto, propõe-se neste capítulo estender a aplicação de estimação-M para classificadores não-lineares baseados na rede ELM que façam uso dos critérios de estimação MQO ou MQR no cômputo da matriz de pesos da camada de saída β.

5.3.1 Estimação-M para Classificação de Padrões Usando a Rede ELM

A formulação teórica apresentada a seguir é muito similar à apresentada no Capítulo 4 para o classificador linear robusto. A diferença está no fato de que os padrões de entrada da camada de saída são formados pelos vetores $\mathbf{h}_\mu$, $\mu = 1,\ldots,N_1$, organizados ao longo das colunas da matriz $\mathbf{H} \in \mathbb{R}^q \times \mathbb{R}^{N_1}$, e tal que q denota o número de neurônios ocultos.

Assim, um estimador-M aplicado ao i-ésimo neurônio de saída do classificador ELM deve minimizar a seguinte função objetivo:

$$J(\beta_i) = \sum_{\mu=1}^{N} \rho(e_{i\mu}) = \sum_{\mu=1}^{N} \rho(d_{i\mu} - y_{i\mu}) = \sum_{\mu=1}^{N} \rho(d_{i\mu} - \beta_i^T \mathbf{h}_\mu), \tag{5.9}$$

em que a função $\rho(\cdot)$ calcula a contribuição de cada erro $e_{i\mu} = d_{i\mu} - y_{i\mu}$ para a função objetivo,

$d_{i\mu}$ é o valor alvo do i-ésimo neurônio de saída para o μ-ésimo vetor $\mathbf{h}_\mu$, $\mu = 1,\ldots,N_1$, com N_1 sendo o número de padrões de treinamento. O vetor β_i é o vetor de pesos do i-ésimo neurônio de saída, $i = 1,\ldots,K$, em que K é o número de classes.

Seja $\psi = \rho'(\cdot)$ a derivada da função $\rho(\cdot)$ em relação ao resíduo $e_{i\mu}$. Diferenciando a função $\rho(\cdot)$ em relação ao vetor de pesos estimado $\hat{\beta}_i$, temos

$$\sum_{\mu=1}^{N_1} \psi(y_{i\mu} - \hat{\beta}_i^T \mathbf{h}_\mu)\mathbf{h}_\mu^T = \mathbf{0}, \tag{5.10}$$

em que $\mathbf{0}$ é um vetor-linha nulo de dimensão $p+1$. Então, definindo a função peso $w(e_{i\mu}) = \psi(e_{i\mu})/e_{i\mu}$, e fazendo $w_{i\mu} = w(e_{i\mu})$, as equações de estimação são dadas por

$$\sum_{\mu=1}^{n} w_{i\mu}(y_{i\mu} - \hat{\beta}_i^T \mathbf{h}_\mu)\mathbf{h}_\mu^T = \mathbf{0}. \tag{5.11}$$

Portanto, resolver as equações de estimação corresponde a solucionar um problema de mínimos quadrados ponderado, minimizando $\sum_\mu w_{i\mu}^2 e_{i\mu}^2$.

É importante lembrar, mais uma vez, que os pesos dependem dos resíduos (ou seja, erros estimados), os resíduos dependem dos coeficientes estimados, e os coeficientes estimados dependem dos pesos. Logo, teremos que utilizar o algoritmo IRLS (FOX, 1997) descrito no Capítulo 4 para estimar a matriz de pesos da camada de saída do classificador ELM.

Os passos do algoritmo IRLS, no contexto de treinamento do classificador ELM usando a Equação (5.6) como referência, são descritos a seguir.

Algoritmo IRLS para Treinamento do Classificador ELM

Passo 1 - Prover uma estimativa inicial $\hat{\beta}_i(0)$ usando a solução MQO em Equação (5.6).

Passo 2 - Em cada iteração t, calcular os resíduos a partir das iterações anteriores $e_{i\mu}(t-1)$, $\mu = 1,\ldots,N_1$, associados com o i-ésimo neurônio de saída, e então calcular os pesos correspondentes $w_{i\mu}(t-1) = w[e_{i\mu}(t-1)]$.

Passo 3 - Obter uma nova estimativa de mínimos quadrados ponderados para $\beta_i(t)$:

$$\hat{\beta}_i(t) = \left[\mathbf{H}\mathbf{W}_i(t-1)\mathbf{H}^T\right]^{-1}\mathbf{H}\mathbf{W}_i(t-1)\mathbf{D}_i^T, \tag{5.12}$$

em que $\mathbf{W}_i(t-1) = \text{diag}\{w_{i\mu}(t-1)\}$ é uma matriz de pesos $N_1 \times N_1$. Repetir Passos 2 e 3 até a convergência do vetor de coeficientes estimados $\hat{\beta}_i(t)$.

Várias funções objetivos $\rho(e_{i\mu})$ e de peso $w_{i\mu} = w(e_{i\mu})$ podem ser usadas para implementar

a matriz de ponderação $\mathbf{W}_i$, tais como aquelas mostradas nas Tabelas (3.3) e (3.4).

Para concluir, a ideia básica da abordagem proposta neste capítulo é muito simples: substituir o algoritmo de estimação MQO da matriz de pesos $\hat{\beta}$ descrito na Equação (5.6), por um que combine o uso do arcabouço teórico da estimação-*M* e o algoritmo IRLS. A partir de agora, nos referiremos à segunda abordagem proposta como classificador *ELM Robusto* ou pela sigla ROB-ELM (do inglês *Robust ELM*).

5.4 Classificador ROB-ELM no Matlab

O modo mais fácil e rápido de se implementar o classificador ROB-ELM no Matlab é usando o comando *ROBUSTFIT*, assim como foi feito para o classificador ROLAM. Para problemas de classificação binária, basta organizar os vetores de atributos de treinamento ao longo das colunas da matriz `X` e os respectivos rótulos ao longo das colunas da matriz `D`.

Suponha que tenhamos apenas $N_1 = 8$ padrões de treinamento, e que cada padrão tenha dimensão $p = 3$ (i.e. possui 3 atributos), sendo que 4 padrões são da classe positiva e 4 da classe negativa. A matriz de pesos **M**, da entrada para os neurônios ocultos, é iniciada aleatoriamente entre 0 e 0,1. Vamos assumir também que há quatro neurônios ocultos (i.e. $q = 4$) e que a função sigmóide logística é usada como função de ativação dos neurônios ocultos.

Assim, os comandos no Matlab para gerar as matrizes **M** e **H** são os seguintes:

```
» p=3; q=4; N1=8;
» X = rand(p+1,N1);
» M = 0.1*rand(q,p+1);
» U = M*X;
» H = 1./(1+exp(-U));
```

O exemplo hipotético a ser dado a seguir assume que há dois neurônios de saída, cada um representando uma função discriminante para o problema de classificação binária de interesse. Assim, o rótulo da classe +1 é definido como $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$, e o rótulo da classe -1 como $\begin{pmatrix} -1 \\ 1 \end{pmatrix}$. A solução, neste caso, é dada pela seguinte seqüência de comandos:

```
» D = [1 1 1 1 -1 -1 -1 -1;-1 -1 -1 -1 1 1 1 1]
D =
1 1 1 1 -1 -1 -1 -1
```

```
-1 -1 -1 -1 1 1 1 1
» B1=robustfit(H',D(1,:))
B1 =
47.2367
151.1743
-189.2802
4.8841
-62.3828
» B2=robustfit(H',D(2,:))
-47.2367
-151.1743
189.2802
-4.8841
62.3828
```

Para finalizar, embora tenha sido mencionado no capítulo anterior, é muito útil mencionar novamente no contexto da implementação do classificador ROB-ELM, que o comando `ROBUSTFIT` mostra um mensagem automática de erro se o número de padrões de treinamento (N_1) for menor ou igual à dimensão do vetor $\mathbf{h}_\mu$; ou seja, o comando acusa erro se $N_1 \leq q+1$.

Esta situação é possível de ocorrer principalmente durante os experimentos para determinar o número ótimo de neurônios ocultos da rede ELM para um determinado problema, em que valores elevados de q podem ser testados. Nestes casos, a solução encontrada consistiu em localizar e comentar a linha de controle de dimensões diretamente no arquivo *robustfit.m*.

5.5 Exemplo Numérico: Classificador ROB-ELM

Com o intuito de mostrar a influência de *outliers* no posicionamento final da curva de decisão entre duas classes de dados não-linearmente separáveis, mostraremos a seguir um experimento como prova de conceito.

Para tal, foi criado um conjunto de dados sintéticos, bidimensionais, consistindo de $N = 120$ amostras mais N_{out} *outliers*. Os classificadores ELM e ROB-ELM são treinados duas vezes. Na primeira vez, eles são treinados com uma base de dados sem *outliers*. Na segunda vez, eles são treinados com os *outliers* sendo adicionados ao conjunto de dados original. É importante mencionar que todas as amostras de dados são usadas para treinar os classificadores, já que

o objetivo é visualizar o posicionamento final da curva de decisão, e não, calcular taxas de reconhecimento.

Para esse experimento, a função objetivo *Andrews* foi usada para implementar o classificador ROB-ELM, e a constante de regularização, necessária à implementação do classificador ELM padrão, foi definida como $\lambda = 10^{-2}$. Três neurônios ocultos com funções de ativação tangente hiperbólica foram usados para ambos os classificadores. Por uma questão de justiça, os classificadores ELM e ROB-ELM usaram os mesmos pesos entre as camadas entrada e oculta, que foram aleatoriamente amostrados a partir de uma distribuição uniforme entre $(-0.1, +0.1)$. O valor *default* do limiar do erro *k* para a função objetivo *Andrews* foi utilizado (i.e. $k = 1,339$).

A fim de avaliar as curvas de decisão finais dos classificadores ELM e ROB-ELM na presença de *outliers*, adicionamos $N_{out} = 10$ *outliers* ao conjunto de dados e os rotulamos como pertencentes à classe $+1$. Os *outliers* foram colocados propositadamente longe da fronteira das classes encontrada para o caso livre de *outliers*; mais especificamente, na região de decisão da classe -1.

Os resultados para o treinamento sem *outliers* são mostrados na Fig. 5.1a, na qual, como esperado, as curvas de decisão de ambos os classificadores são similares. Os resultados para o treinamento com *outliers* são mostrados em Fig. 5.1b, em que dessa vez a curva de decisão do classificador ELM padrão moveu (pendeu) em direção aos *outliers*, enquanto que a curva de decisão do classificador ROB-ELM permaneceu inalterada, revelando, portanto, uma maior robustez a *outliers* da abordagem proposta.

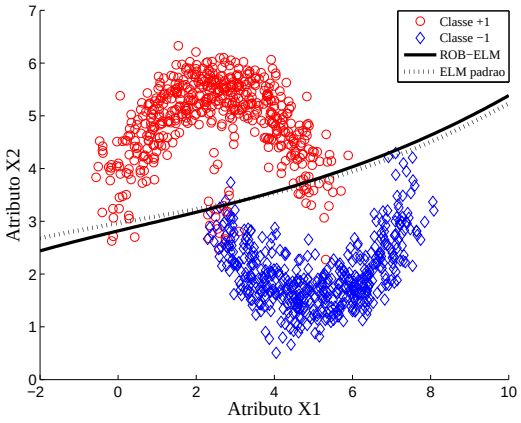
5.6 Resumo do Capítulo

A rede ELM é uma rede neural feedforward, com uma única camada oculta, em que os pesos das entradas para os neurônios ocultos são escolhidos aleatoriamente e os pesos dos neurônios ocultos para a saída são analiticamente determinados. A rede ELM oferece vantagens significativas, tais como velocidade rápida de aprendizagem, facilidade de implementação, e menos interveniência humana quando comparada a SLFNs mais tradicionais, tais como as redes MLP e RBF.

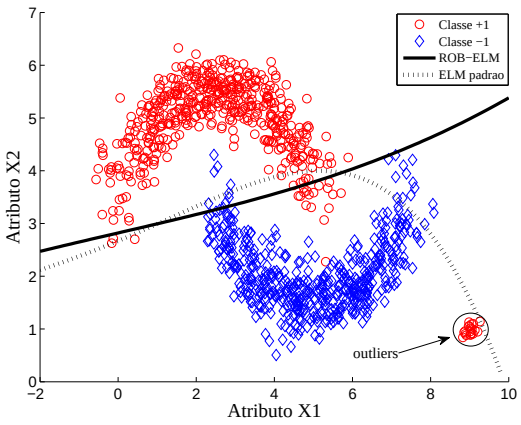
O método dos mínimos quadrados, baseado no cálculo da pseudoinversa, é o método original utilizado na ELM para calcular a relação linear entre os neurônios da camada oculta e os neurônios de saída. Esse método, no entanto, tem algumas desvantagens, tal como a possibilidade da presença de matrizes singulares no cálculo da pseudoinversa, e da ausência de robustez na presença de *outliers*.

Com o intuito de comparar nosso método proposto com outras extensões ELM, que buscam trazer melhorias à ELM em relação ao método tradicional dos mínimos quadrados, foi apresentado nesse capítulo o método ELM BIP. Esse método, no entanto, continua utilizando o artifício de inversão de matrizes para calcular a matriz de pesos β. Além disso, ele também não trata da robustez a *outliers*.

Para contornar tais problemas, foi proposto o desenvolvimento de uma rede ELM Robusta (ROB-ELM), que tem como base o método de regressão robusta e os estimadores-*M*, discutidos no Capítulo 3. Como prova de conceito da nossa proposta, foi definido um experimento que compara as curvas de decisão de dois métodos de regressão (ELM padrão e ELM Robusta), com e sem a presença de *outliers*.



(a) Conjunto de dados sem *outliers* .



(b) Conjunto de dados com *outliers* .

Figura 5.1: Curvas de decisão dos classificadores ELM e ROB-ELM. (a) Conjunto de dados sem *outliers* . (b) Conjunto de dados com *outliers* .

6 *Classificador Linear Robusto: Resultados Experimentais*

O objetivo deste capítulo é analisar, avaliar e validar a proposta de um classificador ROLAM, proposto no Capítulo 4.

Inicialmente, são apresentadas as bases de dados utilizadas nos experimentos, especificando as principais características de cada uma.

Em seguida, são definidas três formas de adição de *outliers* a um conjunto de dados.

São apresentadas também novas provas de conceito, que complementam as provas apresentadas no Capítulo 4, pois agora, as superfícies de decisão são definidas pelos dois classificadores lineares, OLAM e ROLAM, considerando a presença de diferentes percentuais de *outliers*.

Como forma de comparação de desempenho entre os classificadores OLAM e ROLAM, foram realizados testes com as bases de dados Iris e Coluna Vertebral. Para essa última, sendo apresentados testes para quatro diferentes cenários relativos à adição de *outliers*.

Ao final do capítulo são apresentados dois experimentos, com as bases de dados Pima Indians Diabetes e Wisconsin Diagnostic Breast Cancer, com o intuito de avaliar a flexibilidade do classificador ROLAM, mostrando que, além de disponibilizar diversas funções robustas, disponibiliza também, para cada função, um parâmetro de ajuste (k) que, com uma simples alteração em seu valor, permite melhorias na taxa de acerto.

6.1 Bases de Dados

As bases de dados utilizadas nos testes são as seguintes: Iris, Coluna Vertebral, Pima Indians Diabetes e Wisconsin Diagnostic Breast Cancer. Todas elas estão publicamente disponíveis para download a partir do repositório UCI Machine Learning (FRANK; ASUNCION, 2010).

6.1.1 Iris

A base de dados Iris contém 150 padrões divididos em três classes: Virginica, Versicolor e Setosa, cada uma contendo 50 pontos. Cada padrão é representado por quatro atributos, sendo eles: comprimento da sépala, largura da sépala, comprimento da pétala e largura da pétala. Todas as medidas são definidas em centímetros.

O problema abordado nesta base de dados é de classificação binária. Dessa forma, são utilizadas nos testes apenas duas classes dessa base de dados: Virginica, com rótulo -1, e Versicolor, com rótulo $+1$. A terceira classe, Setosa, é utilizada para gerar *outliers* para a classe Virginica, como será apresentado na Seção 6.2.

6.1.2 Coluna Vertebral

A base de dados Coluna Vertebral é relativa a pacientes que apresentam, ou não, um de dois tipos de patologias da coluna vertebral, hérnia de disco ou espondilolistese. Ela é composta por 310 indivíduos e possui três classes, uma classe relativa aos pacientes que não apresentam qualquer patologia, classe Normal, com 100 indivíduos, e duas classes que representam pacientes que possuem uma das duas patologias analisadas, classe Hérnia de disco, com 60 indivíduos, e classe Espondilolistese, com 150 indivíduos. Cada indivíduo é representado por 6 atributos biomecânicos: ângulo de incidência pélvica, ângulo de versão pélvica, declive sacral, ângulo de lordose, raio pélvico e grau de deslizamento.

Com o mesmo objetivo de tratar a classificação desta base de dados como uma classificação binária, nos testes são utilizadas apenas as classes Normal, com rótulo $+1$, e Espondilolistese, com rótulo -1.

6.1.3 Pima Indians Diabetes

A base de dados Pima Indians Diabetes é composta por 768 padrões, cada um com 8 atributos. Os dados estão divididos em duas classes: Positivo, com 268 indivíduos, que representa os pacientes que têm diabetes, e Negativo, com 500 indivíduos, que representa os pacientes que não têm a doença. A classe Positivo é representada pelo rótulo $+1$ e a classe Negativo é representada pelo rótulo -1.

6.1.4 Wisconsin Diagnostic Breast Cancer (WDBC)

A base de dados WDBC representa pacientes que apresentam, ou não, câncer de mama. Os dados estão dividios em duas classes: Benigno e Maligno. A classe Benigno é representada pelo rótulo -1, e conta com 357 pacientes que não apresentam a doença, e a classe Maligno, representada pelo rótulo $+1$, é composta por 212 pacientes que apresentam a doença, perfazendo um total de 569 indivíduos. Cada padrão da classe é representado por 30 atributos.

A Tabela 6.1 resume as principais características apresentadas para cada base de dados, considerando problemas de classificação binária.

BD	Classe $+1$	Classe -1	Qtd. Padrões Classe $+1$	Qtd. Padrões Classe -1	Qtd. Atributos
Iris	Versicolor	Virginica	50	50	4
Coluna	Normal	Espondilolistese	100	150	6
Diabetes	Positivo	Negativo	268	500	8
WDBC	Maligno	Benigno	212	357	30

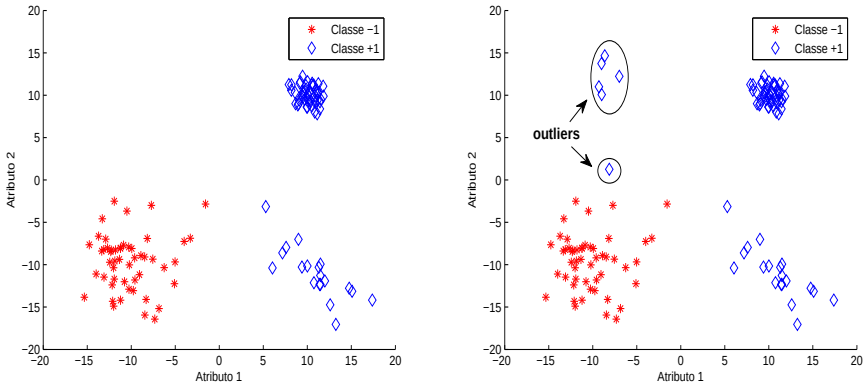
Tabela 6.1: Características das bases de dados utilizadas nos testes (classificação binária).

6.2 Adição de *Outliers*

Encontrar dados reais contendo *outliers* não é uma tarefa trivial. Pelos menos até onde sabemos, não existem bases de dados caracterizadas explicitamente como bases que contenham padrões considerados *outliers*. Com o objetivo de superar essa dificuldade, são propostas neste capítulo três formas de adicionar pontos de *outliers* a bases de dados.

Os problemas aqui tratados são problemas de classificação binária, nos quais podem ser definidas duas regiões de classificação, uma relativa aos dados rotulados como -1 e a outra contendo dados rotulados como +1. No problema de classificação, é definida uma superfície de decisão que separa essas duas regiões. Neste trabalho, são considerados *outliers*, pontos localizados na região oposta à região referente à sua classe de origem. Por exemplo, são considerados *outliers* da classe -1, pontos localizados, fisicamente, na região da classe +1, mas que são rotulados como pertencentes à classe -1. O contrário se aplica para *outliers* da classe +1, que possuem rótulo dessa classe, mas que são inseridos no gráfico na região referente à classe -1. Os três métodos de adição de *outliers* desenvolvidos neste trabalho são variações desta ideia.

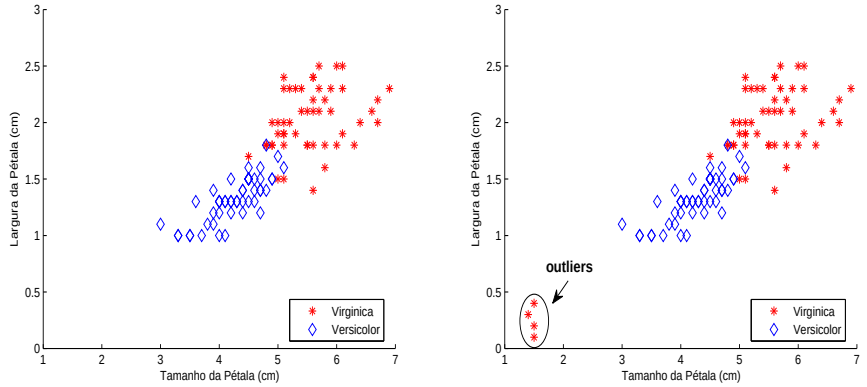
O primeiro método utiliza dados artificiais. Foi criado um conjunto de dados de forma que pudessem ser colocados em um plano bidimensional em regiões distintas, apresentando-se como dados linearmente separáveis. Em seguida, foram definidos pontos que, possuindo o rótulo +1, são inseridos, no plano bidimensional, na região dos dados rotulados como -1. A representação desse método pode ser visualizada na Figura 6.1.



(a) Conjunto sem *outliers*. (b) Conjunto com *outliers*.

Figura 6.1: Primeiro método de adição de *outliers*.

O segundo método trata da inserção de *outliers* nos dados da base Iris. Das três classes existentes nesta base de dados, apenas Virginica (rótulo = -1) e Versicolor (rótulo = +1) são consideradas classes, enquanto que os dados do conjunto Setosa que, considerando apenas duas regiões, fisicamente aparecem na região da classe Versicolor, são adicionados ao conjunto "Virginca + Versicolor", como sendo *outliers* da classe Virginica, ou seja, utilizando o rótulo da classe Virginica, -1. Os dados do conjunto Setosa a serem inseridos como *outliers* são selecionados, de forma aleatória, dentre os 50 pontos deste conjunto. A representação deste método pode ser visualizada na Figura 6.2.



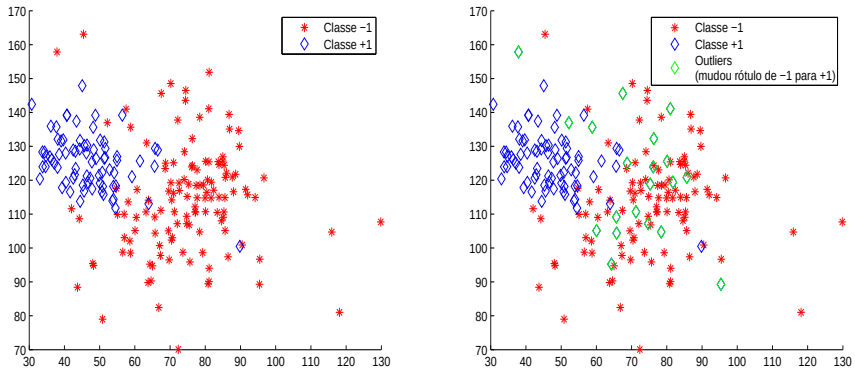
(a) Conjunto sem *outliers*. (b) Conjunto com *outliers*.

Figura 6.2: Segundo método de adição de *outliers*.

Por fim, o último método de adição de *outliers* definidos neste trabalho é o que será utilizado

nos experimentos envolvendo as bases de dados Coluna Vertebral, Pima Indians Diabetes e Wisconsin Diagnostic Breast Cancer.

Os *outliers*, neste caso, não são propriamente novos pontos adicionados ao conjunto inicial, como nos dois casos anteriores, mas sim, são pontos de uma das duas classes que passam a ser rotulados como sendo da outra classe. Por exemplo, dados da classe +1 que têm seus rótulos alterados para -1, ou vice versa, dados da classe -1 tendo seus rótulos alterados para +1. Essa forma de geração de *outliers* tem como base a metodologia introduzida por Kim e Ghahramani (KIM; GHAHRAMANI, 2008). A representação desse método pode ser visualizada na Figura 6.3.



(a) Conjunto sem *outliers*. (b) Conjunto com *outliers*.

Figura 6.3: Terceiro método de adição de *outliers*.

Nos dois capítulos anteriores, foram apresentadas provas de conceito para mostrar a influência dos *outliers* em um classificador linear e em um classificador não-linear. O objetivo era mostrar o desempenho de classificadores robustos e não-robustos em cada um dos casos. Neste Capítulo, apresentamos uma prova de conceito que visa mostrar não só a influência dos *outliers*, mas como se dá essa influência na presença de diferentes percentuais de *outliers*.

6.3 Mais Provas de Conceito

Como provas de conceito relativas à adição de diferentes percentuais de *outliers* aos dados, primeiramente é apresentado um caso com dados artificiais e, em seguida, um caso com dados reais. São analisados dois classificadores lineares, o OLAM e o ROLAM. Em ambos os casos, o ROLAM é executado com a função robusta Huber e o parâmetro de ajuste (k) recebe o valor *default* definido para esta função, igual a 1,345.

6.3.1 Conjunto de Dados Artificiais

Para este experimento, foi criado um conjunto de dados sintéticos com duas classes, representadas pelos rótulos $+1$ e -1. Ao longo do experimento, pontos de dados considerados *outliers* são adicionados, aos poucos, em diferentes percentuais, ao conjunto original sem *outliers*.

O objetivo do experimento é mostrar como a presença de diferentes percentuais de pontos considerados *outliers* influenciam na definição, ou melhor, no posicionamento e inclinação da reta de decisão, que faz a separação entre os dados das duas classes na superfície de decisão.

A Figura 6.4 apresenta as superfícies de decisão para os dados com 5%, 10%, 15% e 20% de *outliers* definidas para os classificadores OLAM e ROLAM.

É fácil ver a influência dos *outliers* nos gráficos. Esse pontos agem como se "puxassem"a reta na sua direção, exercendo uma clara influência sobre ela. Para o classificador OLAM, no entanto, essa influência é perceptivelmente maior que para o classificador ROLAM. Já com 5% de *outliers*, a reta sofre uma inclinação bem maior no OLAM que no ROLAM. Com 15% de *outliers*, a inclinação da reta definida pelo OLAM já faz com que os dados localizados na região de fronteira sejam classificados de forma errada, sendo posicionados na região oposta à sua região original, característica dos dados de sua classe. O classificador robusto, por outro lado, se mostra muito mais indiferente às perturbações geradas pelos *outliers*.

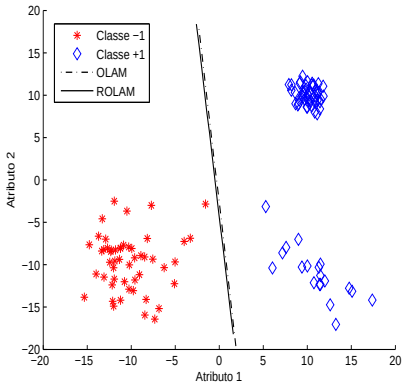
Após a apresentação da prova de conceito utilizando dados sintéticos, são apresentados, a partir de agora, resultados de experimentos realizados com dados reais.

6.4 Conjunto de Dados Reais

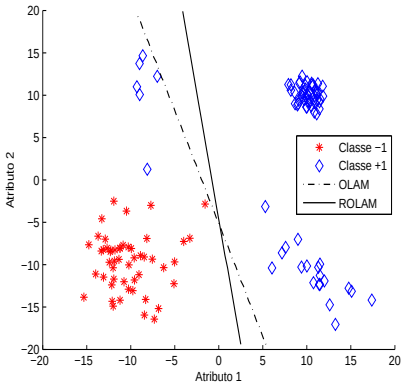
O experimento relativo à prova de conceito utilizando dados reais faz uso da base de dados Iris. São apresentados aqui os resultados dos experimentos de classificação para os dados com e sem *outliers*.

A Figura 6.5 apresenta as superfícies de decisão definidas pelo classificadores OLAM e ROLAM. Com o objetivo de possibilitar a visualização bidimensional dos dados, foram utilizados na construção dos gráficos apenas dois dos quatro atributos citados anteriormente, tamanho da pétala e largura da pétala.

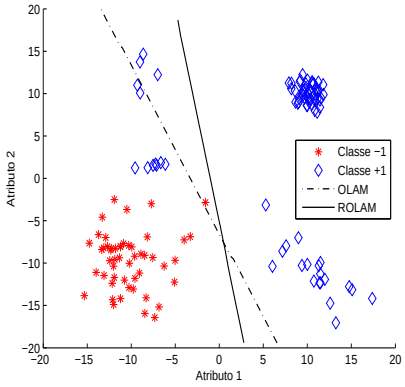
Acompanhando a sequência de gráficos, é possível perceber as alterações ocorridas na superfície de decisão à medida que os *outliers* são adicionados. E, mais uma vez, é possível



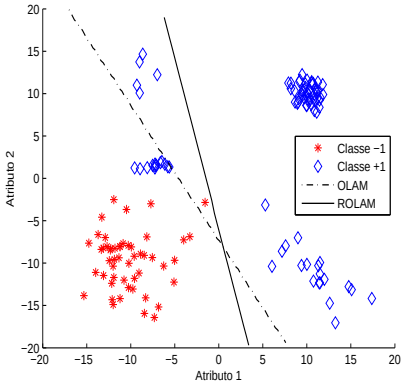
(a) Conjunto sem *outliers*.



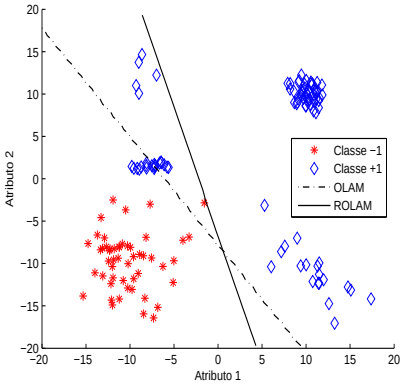
(b) Conjunto com 5% de *outliers*.



(c) Conjunto com 10% de *outliers*.



(d) Conjunto com 15% de *outliers*.



(e) Conjunto com 20% de *outliers*.

Figura 6.4: Retas de decisão dos classificadores OLAM e OLAM Robusto (ROLAM). (a) Dados sem outliers. (b) 5% de outliers. (c) 10% de outliers. (d) 15% de outliers. (e) 20% de outliers.

comprovar que o classificador OLAM é mais sensível à presença dos *outliers*, enquanto que o classificador ROLAM se apresenta mais resistente, sofrendo menor influência destes pontos. Com o aumento da quantidade de *outliers*, no entanto, até mesmo o classificador robusto vai perdendo eficiência gradativamente.

Após a apresentação das provas de conceito, a partir de agora são apresentados experimentos de classificação de dados com as base de dados Iris, Coluna Vertebral, Pima Indians Diabetes e Wisconsin Diagnostic Breast Cancer. Em todos os experimentos são utilizados cinco classificadores lineares, o OLAM e quatro tipos de ROLAM, de acordo com a função robusta implementada (Bisquare, Fair, Huber e Logistic). Como resultado, são apresentadas as taxas de acerto e os desvios padrões obtidos obtidos em cada experimento.

6.5 Resultados

6.6 Base de dados: Iris

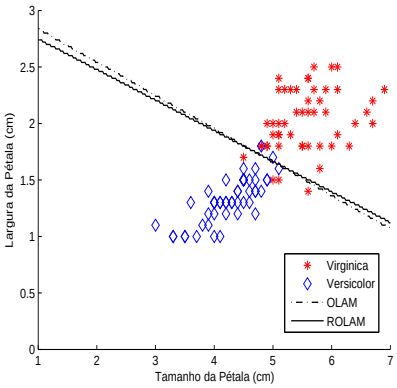
A Tabela 6.2 apresenta os resultados de classificação da base de dados Iris, com e sem *outliers*, utilizando os classficadores OLAM e ROLAM. Estes valores correspondem aos gráficos apresentados na Figura 6.5. Os valores mostram que, com 5% de *outliers*, o ROLAM já começa a ter uma pequena vantagem frente ao OLAM, ficando essa vantagem mais evidente nos resultados para 10% de *outliers*, chegando a diferença a ficar em torno de 8%, se compararmos o OLAM com o ROLAM-Bisquare. Além disso, o desvio padrão apresentado pelos classificadores robustos é menor.

	OLAM	ROLAM (Bisq.)	ROLAM (Fair)	ROLAM (Huber)	ROLAM (Log.)
0%	93.05±4.76	93.35±4.55	93.25±5.52	**94.20±4.75**	93.55±4.68
5%	91.75±5.52	**94.05±4.36**	93.35±5.13	93.10±5.31	93.50±5.29
10%	85.60±9.03	**93.40±5.02**	92.05±5.73	93.10±5.49	93.30±5.23
15%	76.25±9.93	77.20±10.28	**81.70±8.36**	76.30±10.36	79.65±9.85
20%	66.60±11.78	69.10±11.90	**70.20±11.28**	67.00±10.96	68.95±9.33

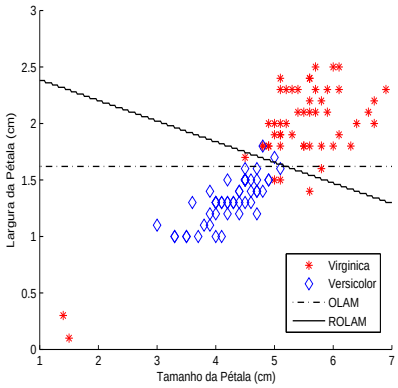
Tabela 6.2: Resultados de classificação dos classificadores OLAM e ROLAM para a base de dados Iris.

6.7 Base de dados: Coluna Vertebral

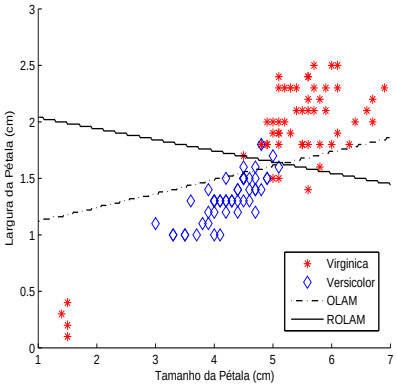
Os classificadores OLAM foram avaliados também em relação à base de dados Coluna Vertebral. Com o objetivo de tornar o experimento um caso de classificação binária, são considerados aqui apenas os dados relativos às classes dos pacientes normais, representados pelo rótulo $+1$, e os dados dos pacientes com espondilolistese, representados pelo rótulo -1.



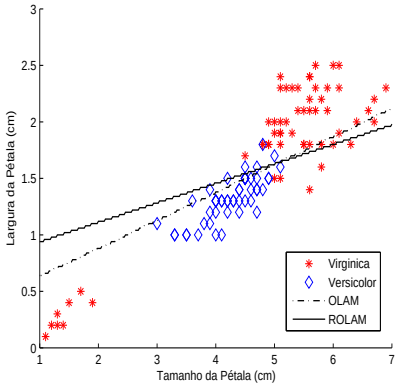
(a) Conjunto sem outliers.



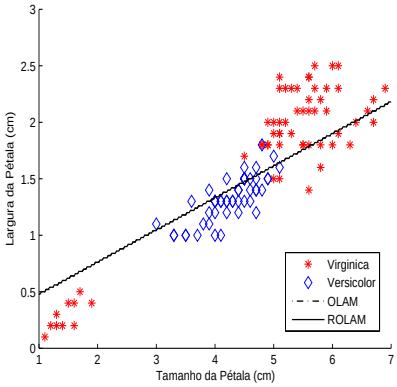
(b) Conjunto com 5% de outliers.



(c) Conjunto com 10% de outliers.



(d) Conjunto com 15% de outliers.



(e) Conjunto com 20% de outliers.

Figura 6.5: Retas de decisão do classificador OLAM para a base de dados Iris. (a) Dados sem outliers. (b) 5% de outliers. (c) 10% de outliers. (d) 15% de outliers. (e) 20% de outliers.

Os *outliers* para esta base de dados são adicionados de acordo com o método 3 apresentado na Seção 6.2. Os rótulos definidos são: +1, para a classe Normal, e -1, para a classe Espondilolistese. Assim, a alteração de rótulos se dá da seguinte forma: de +1 para -1, ao considerar um ponto da classe Normal como sendo da classe Espondilolistese e, dessa forma, *outlier* da classe Normal; e de -1 para +1, no caso contrário, ao considerar um ponto da classe Espondilolistese como sendo da classe Normal e, dessa forma, *outlier* da classe Espondilolistese.

6.7.1 Cenários de Treinamento

Os experimentos com a base de dados Coluna Vertebral foram divididos em quatro cenários. A definição dos cenários contempla as duas formas de geração de *outliers* descritas nessa seção (dados da classe Normal rotulados como da classe Espondilolistese, e vice versa). Além disso, são consideradas também nos cenários, duas formas de distribuição de outliers. Na primeira delas, os *outliers* podem ser qualquer um dos pontos do conjunto total de dados. Na segunda forma, os *outliers* são escolhidos como sendo pontos da classe que terá seu rótulo alterado e que estejam mais distantes da região de fronteira. O objetivo desta análise é observar se a definição de *outliers* como pontos mais afastados da fronteira aumenta a influência desses pontos sobre a superfície de decisão.

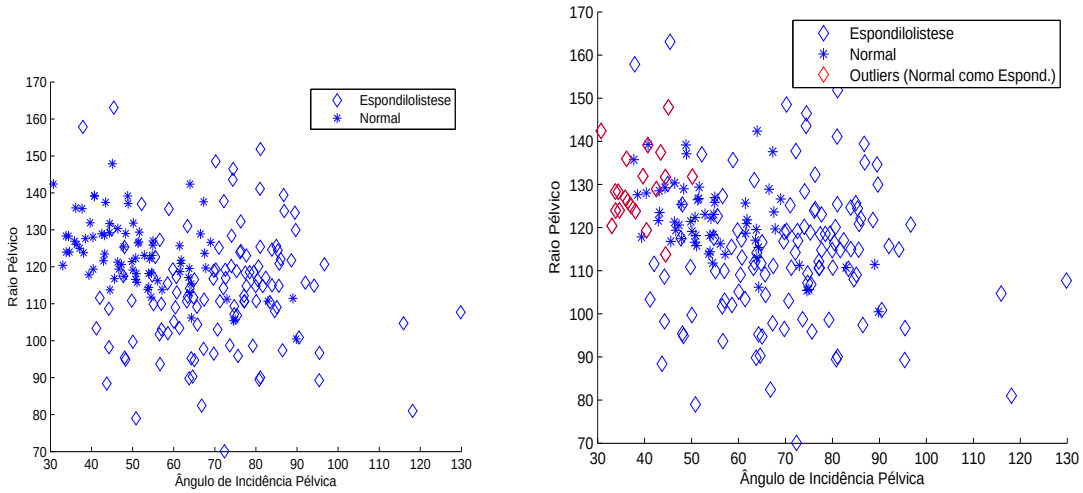
Considerando a combinação de duas formas de geração de *outliers* e duas formas de distribuição de *outliers*, é possível definir quatro cenários de testes, os quais são apresentados graficamente a seguir. Com o objetivo de possibilitar a visualização bidimensional dos dados, foram utilizados na construção dos gráficos apenas dois dos seis atributos da base de dados Coluna Vertebral, ângulo de incidência pélvica e raio pélvico. Nos gráficos contendo *outliers*, é considerada a presença de um percentual de 10% desses pontos.

6.7.2 Cenário 1

Neste cenário, apresentado na Figura 6.6, os *outliers* são pontos da classe Normal rotulados como da classe Espondilolistese, e encontram-se afastados da região de fronteira.

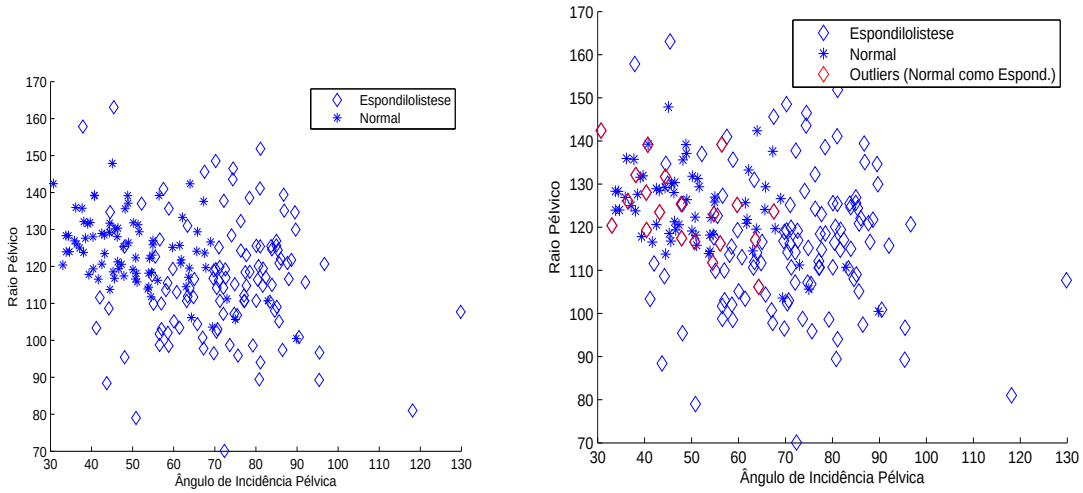
6.7.3 Cenário 2

Neste *cenário*, apresentado na Figura 6.7, os *outliers* são pontos da classe Normal rotulados como da classe Espondilolistese, e podem ser quaisquer pontos da classe Normal, independente de seu posicionamento.



(a) Cenário 1 (sem *outliers*). (b) Cenário 1 (com 10% de *outliers*).

Figura 6.6: Cenário 1 para experimentos com a base de dados Coluna. (a) sem *outliers*. (b) com 10% de *outliers*.

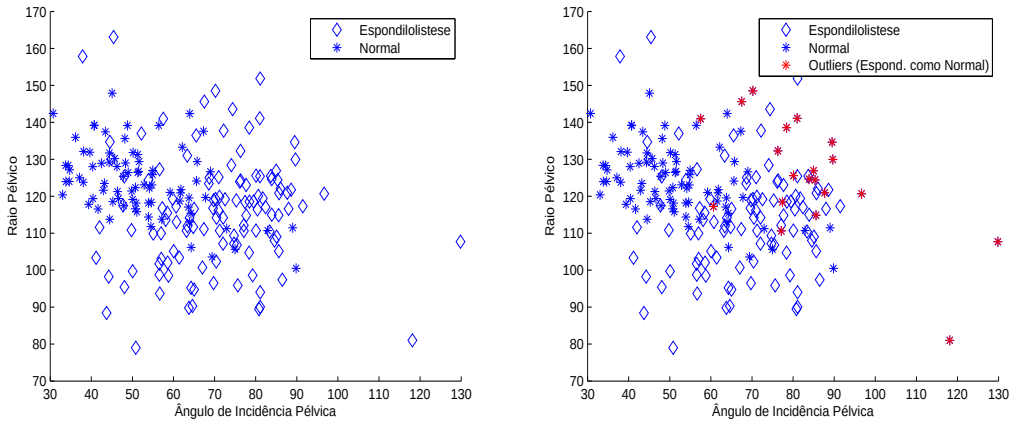


(a) Cenário 2 (sem outliers). (b) Cenário 2 (com 10% de outliers).

Figura 6.7: Cenário 2 para experimentos com a base de dados Coluna. (a) sem outliers. (b) com 10% de outliers.

6.7.4 Cenário 3

Neste cenário, apresentado na Figura 6.8, os *outliers* são pontos da classe Espondilolistese rotulados como da classe Normal, e encontram-se afastados da região de fronteira.

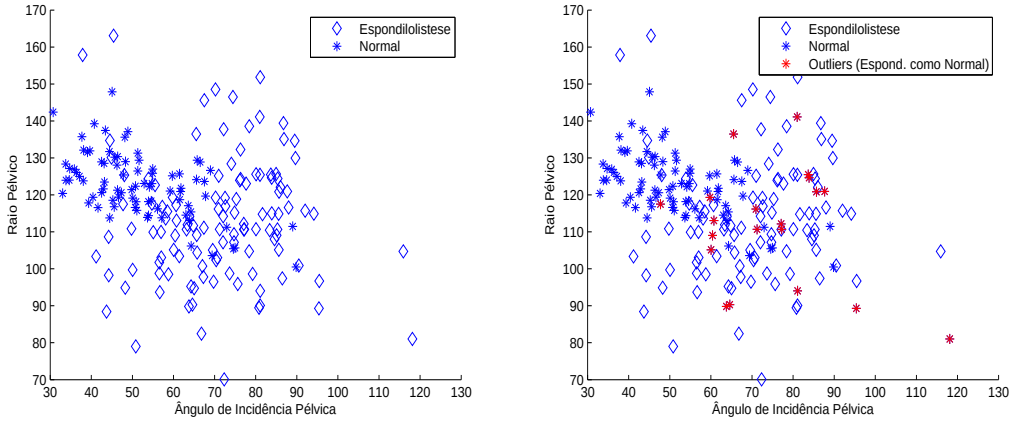


(a) Cenário 3 (sem outliers). (b) Cenário 3 (com 10% de outliers).

Figura 6.8: Cenário 3 para experimentos com a base de dados Coluna. (a) sem outliers. (b) com 10% de outliers.

6.7.5 Cenário 4

Neste cenário, apresentado na Figura 6.9, os *outliers* são pontos da classe Espondilolistese rotulados como da classe Normal, e podem ser quaisquer pontos da classe Espondilolistese, independente de seu posicionamento.



(a) Cenário 4 (sem outliers). (b) Cenário 4 (com 10% de outliers).

Figura 6.9: Cenário 4 para experimentos com a base de dados Coluna. (a) sem outliers. (b) com 10% de outliers.

A seguir são apresentados os resultados dos testes para a base de dados Coluna Vertebral,

considerando os classificadores lineares OLAM e ROLAM.

6.7.6 Resultados

Os resultados são avaliados para diferentes percentuais de *outliers* (0%, 5%, 10%, 15% e 20%) e são analisados para cada um dos quatro cenários definidos anteriormente.

Por motivo de clareza e legibilidade, os gráficos apresentados em todas as seções mostram apenas duas linhas, uma correspondendo ao OLAM e, a outra, ao ROLAM com melhor desempenho naquele teste específico.

Cenário 1

A Tabela 6.3 apresenta os resultados para o testes executados no cenário 1 e os gráficos correspondentes às taxas de acerto e desvios padrões são apresentados na Figura 6.10. Em todos os testes neste cenário, o ROLAM se mostrou superior ao OLAM, ficando essa diferença bastante visível no caso com 10% de *outliers*, chegando tal diferença a ser de até 10% na taxa de acerto, caso em que o desvio padrão também é consideravelmente menor, caindo pela metade.

Nos gráficos da Figura 6.10, o OLAM apresenta sempre uma taxa de acerto menor e maior desvio padrão para os casos com 0%, 5%, 10% e 20% de *outliers*.

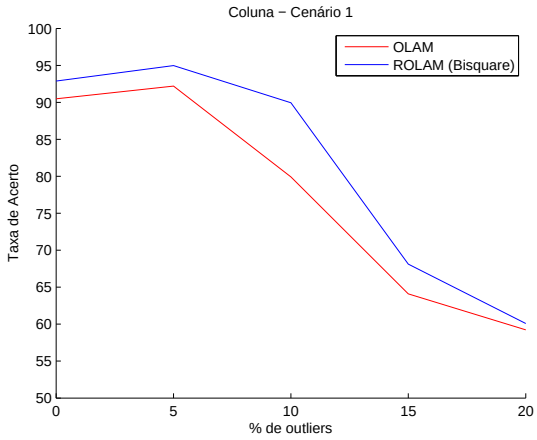
	OLAM	ROLAM (Bisq.)	ROLAM (Fair)	ROLAM (Huber)	ROLAM (Log.)
0%	90.48±4.37	**92.88±3.58**	92.60±3.23	92.04±3.49	91.96±3.97
5%	92.20±4.01	**94.98±2.93**	93.70±3.52	94.54±3.31	94.34±3.14
10%	79.92±8.64	**89.94±4.62**	84.00±8.81	87.94±6.12	88.76±6.03
15%	64.10±8.39	**68.12±9.58**	63.20±8.49	67.10±8.14	64.18±9.44
20%	59.22±6.80	60.08±6.75	59.88±6.62	**60.94±6.56**	59.34±5.96

Tabela 6.3: Resultados para a base de dados Coluna Vertebral no cenário 1 (*outliers*: pontos da classe Normal rotulados como da classe Espondilolistese; posicionamento dos outliers: afastados da fronteira).

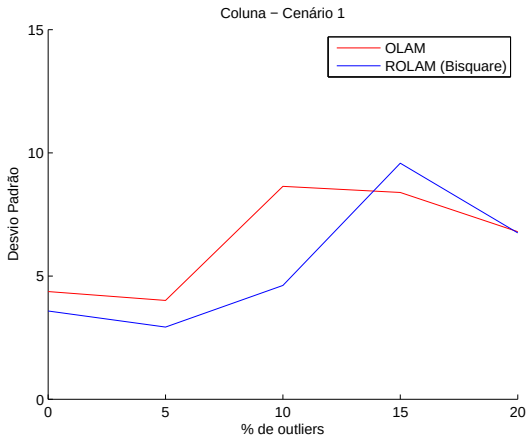
Cenário 2

A Tabela 6.4 apresenta os resultados para o testes executados no cenário 2 e os gráficos correspondentes às taxas de acerto e desvios padrões são apresentados na Figura 6.11. Em praticamente todos os testes, o ROLAM se mostrou superior. Para o caso com 10% de *outliers* nos dados, chega-se a uma diferença de 5% na taxa de acerto e com um desvio padrão menor.

Os gráficos da Figura 6.11 apresentam taxas de acerto maiores e desvio padrão geralmente menores para o ROLAM, ficando os valores de desvio padrão muito próximos para os dois classificadores nos casos com 15% e 20% de *outliers*.



(a) Taxa de Acerto.



(b) Desvio Padrão.

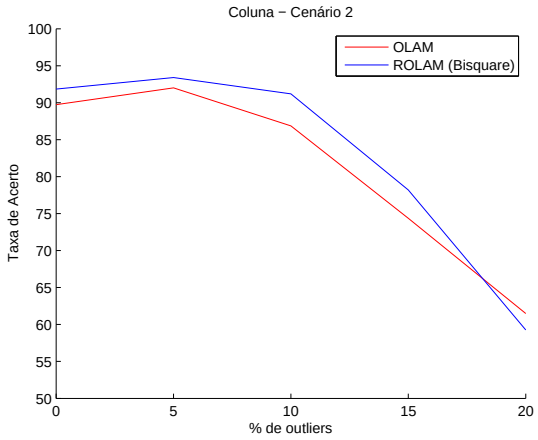
Figura 6.10: Coluna Vertebral - Cenário 1. (a) Taxa de Acerto. (b) Desvio Padrão.

Cenário 3

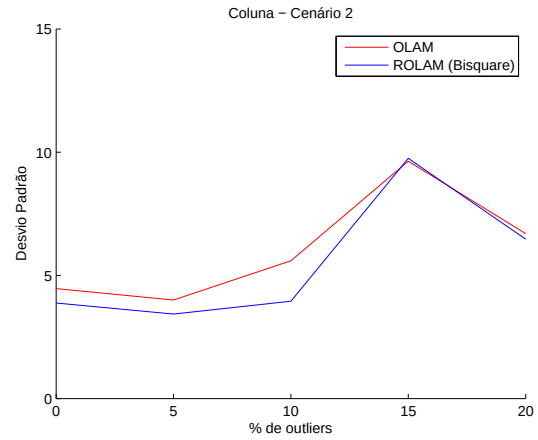
A Tabela 6.5 apresenta os resultados para o testes executados no cenário 3 e os gráficos correspondentes às taxas de acerto e desvios padrões são apresentados na Figura 6.12. Neste cenário, o ROLAM obteve taxas de acerto 9% maior para o caso de 5% de *outliers*, e 8% maior para o caso com 10% de *outliers*, ambos os casos com desvios padrões menores. A superioridade do ROLAM pode ser vista na Figura 6.12.

	OLAM	ROLAM (Bisq.)	ROLAM (Fair)	ROLAM (Huber)	ROLAM (Log.)
0%	89.74±4.46	91.84±3.88	91.60±3.60	91.96±3.47	**92.12±3.82**
5%	92.00±4.00	**93.40±3.43**	92.20±3.61	93.26±3.30	92.84±3.39
10%	86.86±5.59	**91.20±3.95**	90.28±4.22	90.48±5.07	90.14±4.85
15%	74.36±9.63	**78.20±9.75**	69.90±9.98	74.12±12.57	70.96±11.60
20%	61.48±6.69	59.26±6.47	**61.62±6.70**	59.54±6.64	59.52±6.37

Tabela 6.4: Resultados para a base de dados Coluna Vertebral no cenário 2 (*outliers*: pontos da classe Normal rotulados como da classe Espondilolistese; posicionamento dos *outliers*: qualquer lugar).



(a) Taxa de Acerto.



(b) Desvio Padrão.

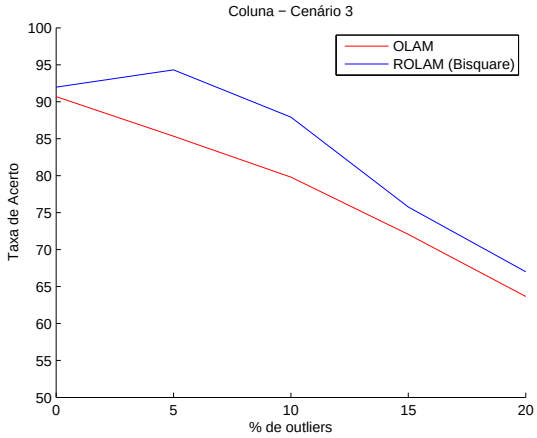
Figura 6.11: Coluna Vertebral - Cenário 2. (a) Taxa de Acerto. (b) Desvio Padrão.

Cenário 4

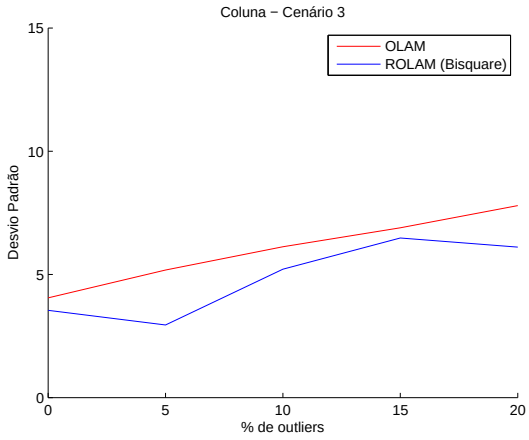
A Tabela 6.6 apresenta os resultados para o testes executados no cenário 4 e os gráficos correspondentes às taxas de acerto e desvios padrões são apresentados na Figura 6.13. Neste

	OLAM	ROLAM (Bisq.)	ROLAM (Fair)	ROLAM (Huber)	ROLAM (Log.)
0%	90.70±4.05	**91.98±3.54**	91.54±3.58	91.58±3.61	91.76±3.58
5%	85.34±5.18	**94.32±2.95**	89.16±4.43	90.14±4.29	90.10±4.25
10%	79.80±6.12	**87.92±5.21**	82.36±5.47	84.26±5.88	84.62±5.35
15%	72.06±6.89	**75.76±6.48**	72.30±6.69	74.60±5.89	73.36±7.15
20%	63.66±7.79	**67.00±6.11**	63.52±7.16	64.42±6.77	64.14±7.53

Tabela 6.5: Resultados para a base de dados Coluna Vertebral no cenário 3 (*outliers*: pontos da classe Espondilolistese rotulados como da classe Normal; posicionamento dos *outliers*: afastados da fronteira).



(a) Taxa de Acerto.



(b) Desvio Padrão.

Figura 6.12: Coluna Vertebral - Cenário 3. (a) Taxa de Acerto. (b) Desvio Padrão.

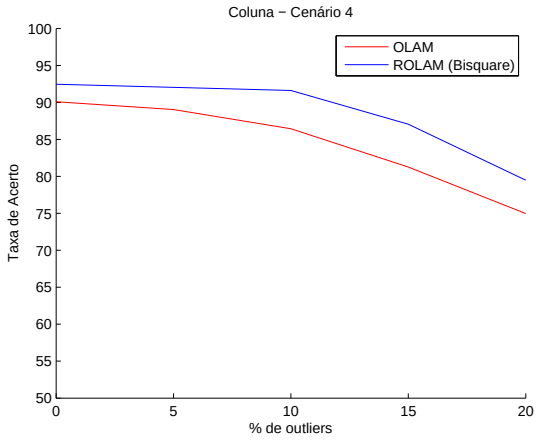
cenário, as diferenças entre as taxas de acerto dos dois classificadores ficou em torno de 3% em favor do ROLAM no caso com 5% de *outliers*, e diferença de 5%, também em favor do ROLAM, no caso com 10% de *outliers*.

Mais uma vez, os gráficos (Figura 6.13) são claros em relação ao melhor desempenho do

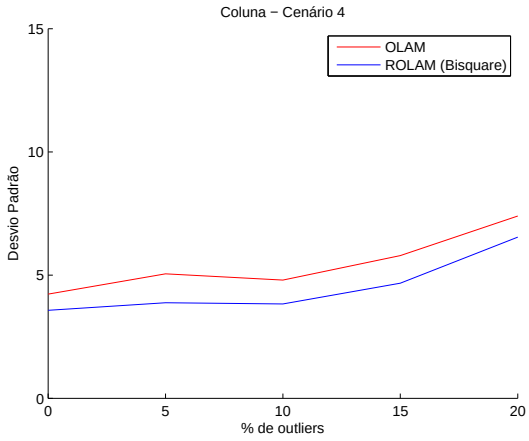
ROLAM.

	OLAM	ROLAM (Bisq.)	ROLAM (Fair)	ROLAM (Huber)	ROLAM (Log.)
0%	90.10±4.23	**92.46±3.57**	91.96±3.56	92.06±3.48	92.46±3.36
5%	89.04±5.05	**92.04±3.88**	91.38±3.81	91.44±4.16	90.94±3.63
10%	86.44±4.80	**91.62±3.83**	89.22±3.62	89.86±4.21	89.78±4.14
15%	81.26±5.79	**87.06±4.67**	84.76±5.61	86.30±4.72	85.52±4.81
20%	74.96±7.40	**79.48±6.54**	76.32±6.70	78.22±6.51	77.18±6.31

Tabela 6.6: Resultados para a base de dados Coluna no cenário 4 (outliers: pontos da classe espondilolistese rotulados como da classe normal; posicionamento dos outliers: qualquer lugar).



(a) Taxa de Acerto.



(b) Desvio Padrão.

Figura 6.13: Coluna Vertebral - Cenário 4. (a) Taxa de Acerto. (b) Desvio Padrão.

Um fato interessante de ser observado é que, os cenários para os quais as diferenças entre o OLAM e o ROLAM foram relativamente melhores, foram naqueles onde os pontos de *outliers* se localizaram mais afastados da fronteira (cenários 1 e 3). Isto se deu devido à maior complexidade gerada por estes pontos na tarefa de definição da superfície de decisão, pois quanto mais

afastados da fronteira de decisão, maiores os resíduos gerados por eles e, consequentemente, maior a influência exercida por eles no método de mínimos quadrados, que atribui o mesmo peso para todos os pontos, compromentendo, assim, o método OLAM.

Em relação ao desempenho das funções robustas, é importante notar que a função Bisquare apresentou um melhor resultado geral para a classificação da base de dados Coluna Vertebral.

6.8 Pima Indians Diabetes

No Capítulo 3, ao serem apresentadas as funções robustas, comentou-se que cada uma das funções possui um parâmetro de ajuste (k), e que esse parâmetro exerce uma influência sobre os pesos que os *outliers* têm na tarefa de classificação, mais especificamente, no cálculo dos resíduos da função erro.

Na função *robustfit* do Matlab, utilizada na implementação dos testes, cada função robusta possui um valor padrão para o parâmetro k, e todos os experimentos apresentados até aqui utilizaram tais valores padrões. Para a base de dados Diabetes, no entanto, foram realizados testes considerando valores de k diferentes dos valores padrões. Para tanto, foi realizada uma busca em grade onde os valores de k variaram de $0,1$ a 10, com incrementos de $0,5$.

A Tabela 6.7 apresenta, para cada função robusta e determinado percentual de *outliers*, os resultados de classificação usando dois valores de k, o padrão e o ótimo, considerado ótimo aqui o valor que gerou melhor resultado dentre todos para cada caso específico.

Neste experimento, foram considerados *outliers* pontos da classe Negativo, que passaram a ser rotulados como Positivos. Os *outliers* foram definidos como pontos afastados da região de fronteira entre as duas classes. Foram destacados na Tabela todos os casos com diferenças nas taxas de acerto maiores ou iguais a 3%.

O objetivo do experimento é analisar a flexibilidade do método robusto que, além de disponibilizar várias funções robustas, em vez de apenas uma única função, como a de cálculo de mínimos quadrados do método padrão, ele também disponibiliza um parâmetro de ajuste que, dependendo do valor que assume, possibilita aumentos significativos na taxa de acerto. Na Tabela 6.7, em quase todos os casos é possível perceber uma taxa de acerto maior para as situações onde não é utilizado o k padrão. Para alguns casos, a diferença na taxas de acertos é significativa, como por exemplo, para o caso com 10% de *outliers*, onde já é possível perceber uma diferença de 3% na taxa obtida pela função Talwar utilizando o valor $kpadrao = 2,795$ e o valor $kotimo = 1,5$.

Com 20% de *outliers*, chega-se a diferenças em torno de 6% em favor do *k* ótimo, nos casos das funções Bisquare e Welsch, e a diferenças em torno de 5%, também em favor do *k* ótimo, para funções como Andrews e Talwar.

Estes são resultados importantes, considerando-se que a alteração do valor de apenas um único parâmetro pode gerar diferenças expressivas nas taxas de acerto dos classificadores robustos, tornando-os mais flexíveis do ponto de vista de possibilidades de configuração.

	0%	5%	10%	20%
OLAM	77.42±2.90	77.21±3.00	73.45±2.92	66.89±3.42
ROLAM (Andrews)				
k_padrão (1.339)	77.17±3.25	77.36±3.32	74.75±3.07	67.64±3.61
k_ótimo	77.68±3.06 (6)	77.74±2.76 (2.5)	75.77±3.22 (0.5)	**72.77±3.74** (0.5)
ROLAM (Bisquare)				
k_padrão (4.685)	77.85±3.04	76.71±3.09	74.71±2.73	67.60±4.58
k_ótimo	77.38±2.70 (9)	77.77±2.92 (7.5)	76.25±3.49 (3)	**73.27±3.68** (1.5)
ROLAM (Cauchy)				
k_padrão (2.385)	77.28±2.95	77.16±3.56	74.19±3.46	68.05±3.79
k_ótimo	78.00±3.09 (5.5)	77.55±3.06 (3.5)	75.90±3.35 (1)	**71.84±3.24** (0.5)
ROLAM (Fair)				
k_padrão (1.400)	76.98±3.05	77.07±3.36	74.08±3.35	67.73±3.42
k_ótimo	77.59±2.80 (9)	77.68±3.08 (0.1)	75.64±3.01 (0.5)	**70.87±3.47** (0.1)
ROLAM (Huber)				
k_padrão (1.345)	76.99±3.14	76.73±2.81	74.86±2.97	67.18±4.12
k_ótimo	77.54±3.05 (7.5)	77.66±2.96 (0.5)	75.97±2.75 (0.5)	**70.95±3.17** (0.1)
ROLAM (Logistic)				
k_padrão (1.205)	76.95±3.28	77.49±3.25	74.64±3.15	67.45±3.79
k_ótimo	77.51±3.01 (9)	77.35±3.46 (6)	75.94±2.62 (0.5)	**71.05±3.62** (0.1)
ROLAM (MQO)				
k_padrão (1)	76.68±3.01	77.16±2.85	74.53±3.26	66.86±3.81
k_ótimo	78.10±2.83 (2)	77.68±2.87 (8)	74.55±3.20 (9.5)	67.81±4.16 (2.5)
ROLAM (Talwar)				
k_padrão (2.795)	75.86±2.76	76.70±3.12	73.60±3.24	67.29±3.82
k_ótimo	78.01±2.87 (3.5)	77.75±3.20 (2)	**76.62±3.22** (1.5)	**72.62±3.75** (1)
ROLAM (Welsch)				
k_padrão (2.985)	77.47±3.19	77.47±3.13	74.50±3.55	67.51±3.93
k_ótimo	77.36±3.14 (9)	77.92±3.00 (4)	76.70±2.99 (1.5)	**73.08±3.41** (1)

Tabela 6.7: Desempenho do ROLAM para diferentes valores de *k* (BD: Diabetes)

6.9 Base de dados: WDBC

Assim com para a base de dados Diabetes, os testes para a base de dados WDBC também foram realizados considerando valores de *k* diferentes dos padronizados. Foi realizada também uma busca em grade, variando os valores de *k* no intervalo entre $0,1$ a 10, com incrementos de $0,5$.

A Tabela 6.8 apresenta, para cada função robusta e determinado percentual de *outliers*, os resultados de classificação usando o *k* padrão e o *k* ótimo, considerado ótimo por ter gerado o melhor resultado dentre todos para cada caso específico. Foram destacados na Tabela todos os casos com diferenças nas taxas de acerto maiores ou iguais a 3%.

Os *outliers* foram definidos como pontos da classe Benigno, que passaram a ser rotulados como Maligno. Os *outliers* foram considerados como pontos mais afastados da região de fronteira entre as duas classes.

Para a base de dados WDBC, as diferenças encontradas para as taxas de acerto considerando diferentes valores de *k* é ainda maior.

No caso da função Bisquare com 10% de *outliers*, considerando apenas a alteração do valor de *k*, por exemplo, a taxa de acerto para $kpadrao = 4,685$ é de 85.69%, enquanto que para $kotimo = 3$, o valor da taxa é de 93.61%, levando a uma diferença de 8%, sendo o desvio padrão também menor.

Diferenças em torno de 6% em favor do *k* ótimo podem ser vistas nos casos das funções Talwar com 10% de *outliers* e Bisquare com 20% de *outliers*.

6.10 Resumo do Capítulo

Neste capítulo foram apresentados uma gama abrangente de resultados experimentais realizados com os dois tipos de classificadores lineares discutidos nesta tese, o classificador OLAM e sua versão robusta ROLAM.

Dando continuidade à prova de conceito apresentada no Capítulo 4, foram apresentadas aqui novas provas de conceito para comparar a definição das superfícies de decisão geradas pelos dois classificadores, na presença de diferentes percentuais de outliers, confirmando a maior propensão do classificador OLAM de ser afetado pelos *outliers*.

Os classificadores foram avaliados também em relação ao desempenho na tarefa de classificação de diferentes bases de dados. Neste caso, foram analisados as taxas de acerto e desvios padrões obtidos por cada classificador. Na maioria dos experimentos, é possível constatar a superioridade do classificador robusto em relação ao classificador padrão, mesmo em casos de aumento do percentual de *outliers* presente nos dados.

Para o ROLAM, foi avaliado também o desempenho deste classificador em relação à constante de ajuste *k* definida para cada função robusta, o que possibilitou ver a importância desta constante para o desempenho do classificador, e a própria flexibilidade de um classificador robusto. Al-

	0%	5%	10%	20%
OLAM	95.39±1.87	92.47±2.51	85.76±3.31	74.58±4.45
ROLAM (Andrews)				
k_padrão (1.339)	94.33±1.96	95.24±1.83	86.50±3.27	75.02±3.86
k_ótimo	95.89±1.80 (3.5)	95.40±2.00 (1.5)	**89.75±5.09** (1)	**80.46±4.04** (0.5)
ROLAM (Bisquare)				
k_padrão (4.685)	94.78±1.96	95.17±1.88	85.69±3.38	75.29±4.15
k_ótimo	95.57±1.77 (9.5)	95.31±1.85 (4.5)	**93.61±2.78** (3)	**81.44±4.79** (1.5)
ROLAM (Cauchy)				
k_padrão (2.385)	94.89±1.69	93.59±2.66	86.12±3.51	75.47±3.93
k_ótimo	95.82±1.91 (8.5)	94.09±2.23 (2)	**90.72±2.72** (0.5)	77.53±4.14 (0.1)
ROLAM (Fair)				
k_padrão (1.400)	94.80±1.94	92.55±2.43	85.88±3.19	75.96±3.80
k_ótimo	95.61±1.69 (9)	93.40±2.11 (0.1)	87.17±2.65 (0.1)	76.03±3.89 (4.5)
ROLAM (Huber)				
k_padrão (1.345)	94.07±2.00	93.94±2.27	86.21±3.07	74.61±3.57
k_ótimo	95.87±1.78 (7.5)	93.71±2.33 (1.5)	87.91±3.04 (0.1)	76.61±3.61 (0.5)
ROLAM (Logistic)				
k_padrão (1.205)	94.68±2.12	92.97±2.43	86.23±3.14	75.79±4.10
k_ótimo	95.86±1.60 (6.5)	93.29±2.55 (1)	86.46±3.39 (1.5)	76.16±3.90 (2)
ROLAM (MQO)				
k_padrão (1)	95.57±1.93	91.29±2.50	85.16±3.16	75.54±4.10
k_ótimo	95.87±1.52 (2.5)	91.89±2.13 (9)	86.31±3.17 (1.5)	76.09±4.21 (1)
ROLAM (Talwar)				
k_padrão (2.795)	95.54±1.86	95.02±2.15	85.82±3.31	74.76±3.77
k_ótimo	95.99±1.80 (5)	95.44±1.90 (2.5)	**92.19±2.79** (1.5)	**78.75±3.69** (1)
ROLAM (Welsch)				
k_padrão (2.985)	94.61±1.93	94.89±1.98	85.63±3.25	74.97±3.87
k_ótimo	95.70±1.72 (8)	94.94±1.89 (2.5)	**91.81±3.05** (1.5)	**80.45±4.62** (1)

Tabela 6.8: Desempenho do ROLAM para diferentes valores de *k* (BD: WDBC)

terando apenas o valor de *k* de uma função robusta, é possível obter taxas de acerto maiores, acompanhadas de menores desvios padrões.

No próximo capítulo será apresentada a primeira parte dos resultados experimentais realizados visando comparar o classificador ELM com sua versão robusta, o classificador ROB-ELM.

7 *Classificador Não Linear Robusto: Resultados Experimentais (Parte 1)*

O objetivo deste capítulo é analisar, avaliar e validar a proposta de dois classificadores não lineares robustos, ROB-ELM e ROB-ELM-BIP.

Primeiramente são discutidas as bases de dados utilizadas nos experimentos e o cenário no qual os testes foram realizados.

Em seguida, são apresentados os experimentos executados com os dois classificadores para a classificação de dados na presença de diferentes percentuais de *outliers*.

7.1 Bases de Dados

As bases de dados utilizadas nos testes aqui apresentados são Wisconsin Diagnostic Breast Cancer (apresentada no Capítulo 6) e Ionosphere. Ambas estão publicamente disponíveis para download a partir do repositório UCI Machine Learning (FRANK; ASUNCION, 2010).

7.1.1 Ionosphere

A base de dados Ionosphere possui informações sobre classificação de retorno de radar a partir da Ionosfera. Dois valores são possíveis: Bom, indica que o retorno do radar mostra evidência de algum tipo de estrutura na Ionosfera, e Ruim, indica que o retorno não mostra evidência alguma, pois seu sinal passa através da Ionosfera. São definidas, portanto, duas classes: Bom, com 225 padrões e rótulo -1, e Ruim, com 126 padrões e rótulo $+1$. Todos os padrões possuem 34 atributos.

A Tabela 7.1 resume as principais características de cada base de dados, considerando problemas de classificação binária.

BD	Classe $+1$	Classe -1	Qtd. Padrões Classe $+1$	Qtd. Padrões Classe -1	Qtd. Atributos
WDBC	Maligno	Benigno	212	357	30
Ionosphere	Ruim	Bom	126	225	34

Tabela 7.1: Características das bases de dados utilizadas nos testes (classificação binária).

7.2 Cenários

No Capítulo 6, foram apresentados quatro tipos de cenários de testes, considerando quatro combinações em relação ao posicionamento dos *outliers* (afastados da fronteira ou dispostos em qualquer lugar do espaço) e a classe a ter seu rótulo alterado (classe -1 ou classe +1). Nos testes realizados neste capítulo com os classificadores não-lineares, é adicionado mais um parâmetro à composição destes cenários, relativo ao posicionamento dos pontos de teste.

Na abordagem utilizada, os pontos de teste podem ser de dois tipos: pontos que estão dispostos em qualquer lugar do espaço e pontos que estão próximos à fronteira de decisão. Tradicionalmente, os dados são inicialmente separados, de forma aleatória, entre treinamento e teste. Isso faz com que os pontos de teste possam ser qualquer um dos pontos do conjunto de dados, independente de seu posicionamento no espaço, ou seja, podem estar dispostos em qualquer lugar.

Pontos que encontram-se próximos à fronteira de decisão, no entanto, são os primeiros a serem afetados quando a superfície de decisão se desloca, por exemplo, por influência de *outliers*. Se selecionarmos os pontos de teste nessa região, mais complexa se torna a tarefa de classificação, pois o próprio cenário favorece a diminuição da taxa de acerto. Devido a isso, essa foi a nossa opção na escolha do posicionamento dos pontos de teste.

Os experimentos realizados neste capítulo consideram, portanto, o seguinte cenário: os *outliers* encontram-se dispostos em qualquer lugar, a classe a ter seu rótulo alterado é a classe -1 (classe Benigno, para a base WDBC e classe Bom, para a base Ionospehere), e os pontos de teste encontram-se na fronteira de decisão.

7.3 Classificação de Bases de Dados

Nesta seção é analisado o desempenho de dois tipos de classificadores não lineares robustos, ROB-ELM e ROB-ELM-BIP, que têm como base os classificadores ELM e ELM-BIP, respectivamente.

7.3.1 Classificador ELM Robusto

Nos experimentos realizados com o classificador ROB-ELM, primeiramente é analisada a base de dados Wisconsin Diagnostic Breast Cancer (WDBC) e, em seguida, a base de dados Ionosphere.

Wisconsin Diagnostic Breast Cancer

Nesta seção são apresentados os resultados dos testes realizados para a base de dados Wisconsin Diagnostic Breast Cancer, considerando a adição de 5%, 10% e 20% de *outliers* nos dados originais. Os resultados para cada percentual são apresentados nas Tabelas 7.3.1, 7.4 e 7.3.1, respectivamente.

Com o intuito de facilitar a comparação, a Tabela 7.2 apresenta inicialmente os resultados da classificação desta base de dados, através dos classificadores ELM e ROB-ELM, em um ambiente sem *outliers*.

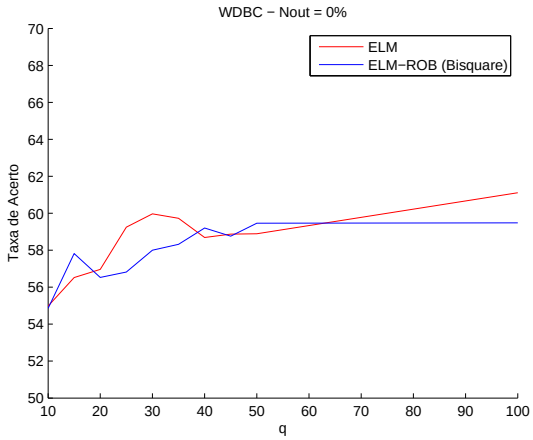
q	ELM ($\lambda = 0.01$)	ROB-ELM (Bisquare)	ROB-ELM (Fair)	ROB-ELM (Huber)	ROB-ELM (Logistic)
10	54.98±12.94	54.86±12.61	55.18±13.13	55.19±13.08	**56.69±11.82**
15	56.52±13.27	**57.82±11.95**	54.97±10.93	55.22±11.97	55.54±12.35
20	56.96±10.05	56.53±11.46	57.11±10.67	54.25±11.29	**58.68±9.42**
25	**59.25±9.42**	56.82±9.41	57.38±9.19	56.46±9.07	55.39±10.61
30	**59.97±7.52**	58.00±8.83	57.71±8.89	58.29±8.67	55.75±9.01
35	**59.73±7.81**	58.32±7.65	58.40±8.12	56.63±6.98	58.89±6.91
40	58.69±6.32	**59.20±7.50**	58.10±6.83	58.01±6.58	57.45±7.02
45	58.87±7.37	58.76±6.82	58.00±5.74	**59.20±6.19**	58.28±6.19
50	58.89±5.60	**59.46±6.33**	57.94±6.19	58.30±6.22	58.45±5.20
100	**61.11±3.71**	59.48±3.42	58.72±2.92	58.59±3.18	57.76±3.45

Tabela 7.2: Comparação de desempenho entre os classificadores ELM e ROB-ELM - BD: WDBC (sem *outliers*).

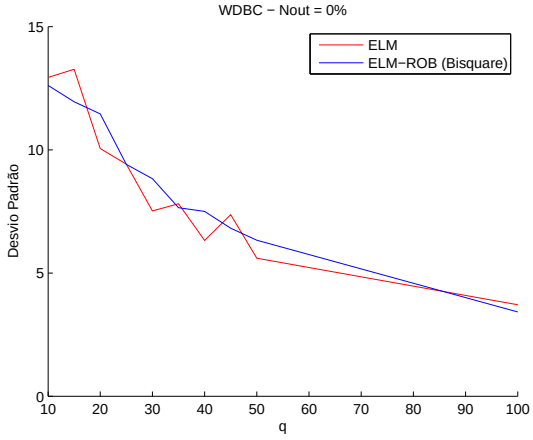
Pela Tabela 7.2, percebe-se que não há uma grande diferença entre os resultados de ELM e ROB-ELM em um ambiente sem *outliers*, o que já era esperado, pois nestas condições, os classificadores comportam-se de forma semelhante, como pode ser comprovado na Figura 7.1, que apresenta uma leve vantagem para o ELM. É importante sempre lembrar, no entanto, que os gráficos consideram apenas uma das funções robustas.

Pela Tabela 7.3.1, percebe-se também que não há uma grande diferença no desempenho entre as situações sem *outliers* e com 5% de *outliers*.

O desafio na classificação com *outliers* aumenta à medida que aumenta o percentual destes pontos nos dados. No cenário com 5% de *outliers*, não existem grandes diferenças entre os dois classificadores. Com 10% de *outliers*, já é possível ver uma diferença em torno de 5% no uso da função Bisquare com 30 neurônios ocultos. No caso com 20%, no entanto, é possível



(a) Taxa de Acerto.



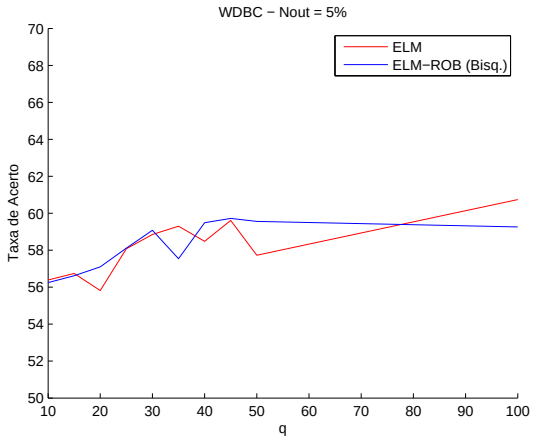
(b) Desvio Padrão.

Figura 7.1: WDBC - Nout = 0%. (a) Taxa de Acerto. (b) Desvio Padrão.

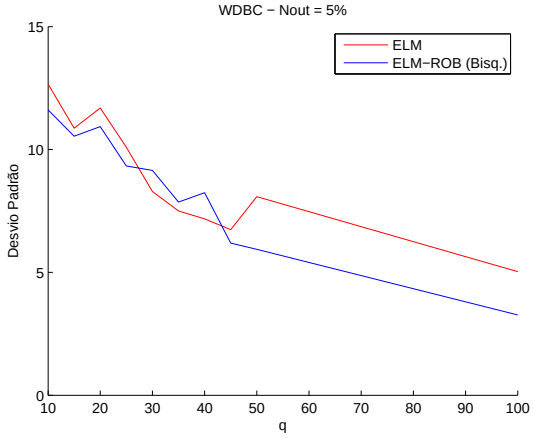
q	ELM ($\lambda = 0.01$)	ROB-ELM (Bisquare)	ROB-ELM (Fair)	ROB-ELM (Huber)	ROB-ELM (Logistic)
10	**56.39±12.66**	56.25±11.61	53.02±12.76	53.35±11.45	52.31±13.53
15	**56.74±10.87**	56.62±10.54	55.20±11.31	56.56±10.97	55.29±12.28
20	55.82±11.69	**57.10±10.93**	53.96±10.03	55.05±11.04	56.07±10.23
25	58.08±10.09	**58.11±9.33**	55.19±10.48	56.25±9.59	55.01±10.00
30	58.85±8.29	**59.08±9.15**	57.79±9.10	58.92±8.68	56.90±8.60
35	**59.30±7.50**	57.54±7.86	57.20±7.58	57.63±8.01	56.31±7.19
40	58.48±7.18	**59.49±8.24**	58.45±6.66	58.04±6.53	56.75±7.92
45	59.61±6.74	**59.72±6.19**	57.69±5.98	57.98±6.17	57.13±6.16
50	57.73±8.08	**59.56±5.94**	58.18±5.27	57.53±5.55	57.84±5.97
100	**60.74±5.03**	59.26±3.27	58.00±2.90	57.93±2.85	57.74±3.23

Tabela 7.3: Comparação de desempenho entre os classificadores ELM e ROB-ELM - BD: WDBC ($P_{out} = 5\%$).

observar as diferenças maiores nas taxas de acerto entre os dois classificadores, chegando a uma diferença de 7% em favor do classificador robusto, também com a função Bisquare, mas agora



(a) Taxa de Acerto.



(b) Desvio Padrão.

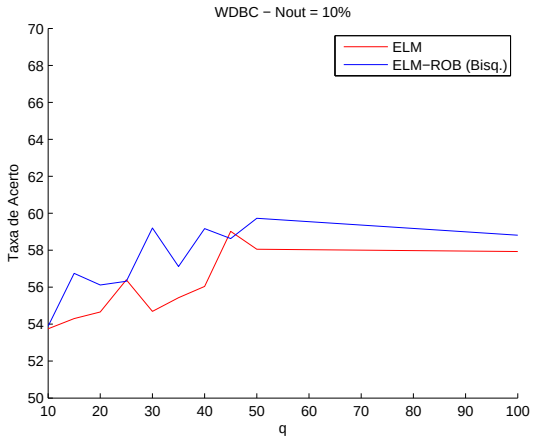
Figura 7.2: WDBC - Nout = 5%. (a) Taxa de Acerto. (b) Desvio Padrão.

q	ELM ($\lambda = 0.01$)	ROB-ELM (Bisquare)	ROB-ELM (Fair)	ROB-ELM (Huber)	ROB-ELM (Logistic)
10	53.75±13.39	53.89±11.62	52.37±11.95	**54.96±11.80**	53.81±11.62
15	54.30±11.08	**56.75±10.15**	54.36±11.02	54.46±11.97	54.99±11.96
20	54.66±11.52	**56.12±11.06**	53.70±9.94	54.15±10.69	54.79±11.55
25	56.40±9.61	56.32±9.52	55.01±9.27	54.17±9.74	**56.68±8.28**
30	54.69±9.60	**59.20±7.67**	54.08±8.70	56.75±8.75	57.50±7.89
35	55.43±9.01	57.12±8.13	57.25±7.88	56.47±7.83	**57.80±7.91**
40	56.04±8.30	**59.17±6.78**	56.74±7.25	56.55±6.43	57.51±7.91
45	**59.02±6.53**	58.63±7.20	57.53±5.85	56.53±7.02	57.61±6.61
50	58.05±7.04	**59.73±5.71**	56.49±5.67	57.28±7.37	57.31±6.25
100	57.93±5.58	**58.81±4.03**	57.64±3.92	57.82±3.95	57.31±3.67

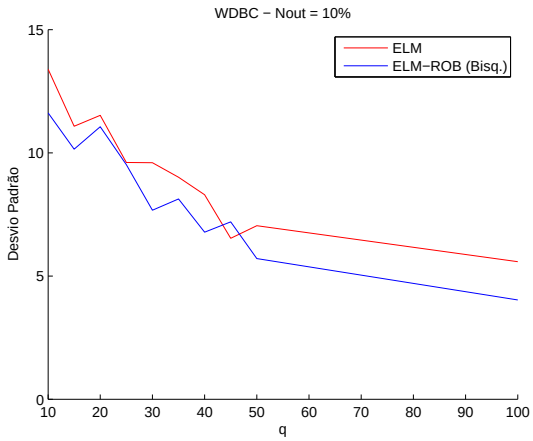
Tabela 7.4: Comparação de desempenho entre os classificadores ELM e ROB-ELM - BD: WDBC ($P_{out} = 10\%$).

com 40 neurônios ocultos. As Figuras 7.2, 7.3 e 7.4 mostram graficamente estas diferenças.

Em todos os casos em que há aumento da taxa de acerto, é possível perceber também a



(a) Taxa de Acerto.



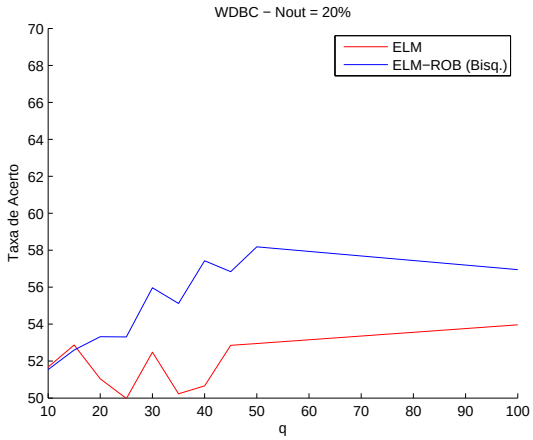
(b) Desvio Padrão.

Figura 7.3: WDBC - Nout = 10%. (a) Taxa de Acerto. (b) Desvio Padrão.

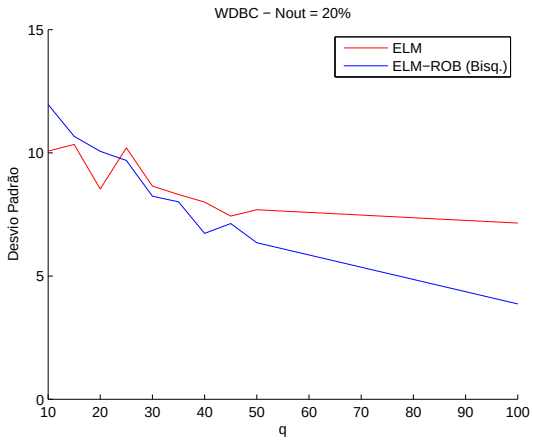
q	ELM ($\lambda = 0.01$)	ROB-ELM (Bisquare)	ROB-ELM (Fair)	ROB-ELM (Huber)	ROB-ELM (Logistic)
10	51.68±10.07	51.54±11.96	52.32±10.61	49.73±10.34	**54.18±10.21**
15	**52.87±10.34**	52.60±10.67	51.32±9.71	52.59±9.69	50.43±8.97
20	51.04±8.53	**53.32±10.06**	51.26±9.01	52.29±9.16	52.83±9.87
25	49.98±10.20	**53.31±9.69**	50.39±8.86	51.16±9.27	52.87±9.54
30	52.48±8.65	**55.97±8.24**	51.14±8.26	51.82±9.19	51.41±8.38
35	50.23±8.31	**55.12±8.01**	51.73±7.20	53.75±9.32	50.97±7.87
40	50.66±8.00	**57.43±6.73**	50.61±8.50	53.52±7.84	52.35±8.45
45	52.85±7.43	**56.84±7.13**	51.75±7.40	53.61±7.07	53.83±7.67
50	52.95±7.69	**58.18±6.35**	52.36±6.67	54.44±7.36	55.09±7.46
100	53.96±7.15	**56.95±3.87**	52.43±7.74	52.50±7.06	53.85±7.28

Tabela 7.5: Comparação de desempenho entre os classificadores ELM e ROB-ELM - BD: WDBC ($P_{out} = 20\%$).

diminuição do desvio padrão.



(a) Taxa de Acerto.



(b) Desvio Padrão.

Figura 7.4: WDBC - Nout = 20%. (a) Taxa de Acerto. (b) Desvio Padrão.

7.3.2 Ionosphere

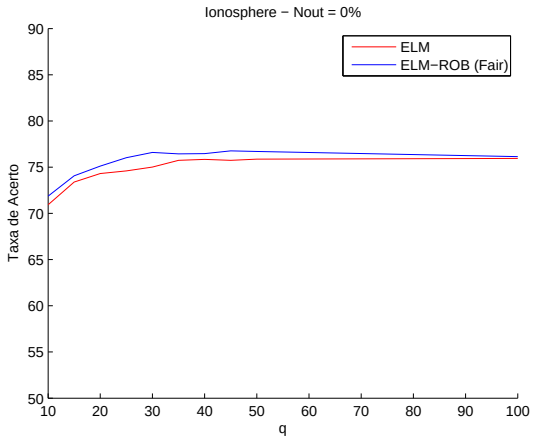
São apresentados agora os resultados dos testes realizados para a base de dados Ionosphere, considerando a adição de 5%, 10% e 20% de *outliers* aos dados originais. Os resultados para cada percentual são apresentados nas Tabelas 7.7, 7.8 e 7.9, respectivamente. A Tabela 7.6 apresenta inicialmente os resultados para um ambiente de classificação sem *outliers*.

Um fato interessante que pode ser observado na Tabela 7.6, é o baixo desempenho do classificador ROB-ELM baseado na função robusta Bisquare, em relação aos outros classificadores ROB-ELM e em relação ao próprio ELM. Suas taxas de acerto, no entanto, tornam-se melhores no ambiente com 5% de *outliers*, ou seja, em um ambiente mais complexo, o ROB-ELM com

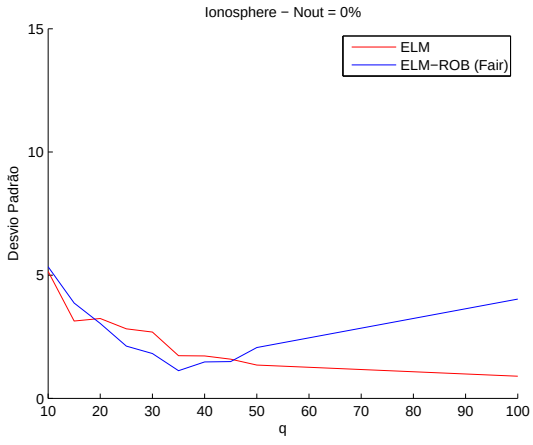
q	ELM ($\lambda = 0.01$)	ROB-ELM (Bisquare)	ROB-ELM (Fair)	ROB-ELM (Huber)	ROB-ELM (Logistic)
10	70.93±5.15	64.03±7.43	**71.87±5.34**	71.13±5.61	71.26±5.92
15	73.39±3.14	69.69±6.54	**74.07±3.87**	73.39±4.02	73.90±4.12
20	74.31±3.24	69.14±6.36	75.11±3.04	**75.19±3.48**	75.00±3.49
25	74.59±2.82	69.30±6.31	**76.03±2.12**	75.87±2.51	75.50±2.65
30	75.01±2.69	69.59±5.78	**76.60±1.82**	76.47±1.88	76.30±1.99
35	75.74±1.73	69.76±5.30	**76.44±1.12**	76.19±0.95	76.34±1.04
40	75.84±1.72	70.34±5.39	76.47±1.48	76.47±1.43	**76.84±1.45**
45	75.74±1.59	70.97±5.48	**76.76±1.49**	76.67±1.63	76.61±1.59
50	75.87±1.35	70.89±5.88	76.70±2.06	**76.80±1.68**	76.57±2.15
100	75.94±0.90	68.80±5.86	76.14±4.03	75.93±4.69	**76.36±4.13**

Tabela 7.6: Comparação de desempenho entre os classificadores ELM e ROB-ELM - BD: Ionosphere (sem *outliers*).

função Bisquare aumenta o seu poder de classificação.



(a) Taxa de Acerto.



(b) Desvio Padrão.

Figura 7.5: Ionosphere - Nout = 0%. (a) Taxa de Acerto. (b) Desvio Padrão.

Na base de dados Ionosphere, é possível perceber, em relação à base WDBC, uma maior

q	ELM ($\lambda = 0.01$)	ROB-ELM (Bisquare)	ROB-ELM (Fair)	ROB-ELM (Huber)	ROB-ELM (Logistic)
10	70.94±4.95	69.77±6.72	72.21±4.66	**72.30±4.27**	72.23±4.05
15	70.67±4.53	70.76±5.54	73.41±3.34	73.11±3.40	**73.56±3.75**
20	70.93±3.90	72.29±5.87	74.00±3.49	74.07±3.39	**74.44±3.40**
25	71.24±3.33	72.89±4.86	73.94±2.94	**74.51±3.35**	74.41±2.87
30	71.90±3.16	72.06±4.54	75.50±2.56	**75.81±2.19**	75.30±2.41
35	71.81±3.01	73.90±3.49	**76.36±1.75**	76.13±1.51	76.27±1.62
40	72.73±2.53	73.87±3.35	76.20±2.10	**76.69±1.96**	76.60±2.03
45	73.34±2.25	73.46±4.05	76.69±1.83	76.83±1.86	**76.93±1.89**
50	73.27±2.33	73.71±4.36	**77.21±2.02**	76.80±2.07	76.76±2.18
100	73.97±2.03	71.49±5.42	**77.03±3.68**	76.46±4.32	77.03±4.17

Tabela 7.7: Comparação de desempenho entre os classificadores ELM e ROB-ELM - BD: Ionosphere ($P_{out} = 5\%$).

diferença nas taxas de acerto obtidas pelo ELM, ao comparamos a situação sem *outliers* e com 5% de *outliers*. No ELM, a redução da taxa de acerto no caso sem *outliers* em relação ao caso com 5% de *outliers*, chega a 4% com 35 neurônios ocultos. Já no caso do ROB-ELM, não se percebe esta queda no desempenho. Pelo contrário, em alguns casos, o desempenho do ROB-ELM chega a melhorar visivelmente no ambiente com 5% de *outliers*, como por exemplo, no caso com 35 neurônios ocultos e função robusta Bisquare, onde a taxa de acerto do ROB-ELM chega a ser 4% maior no ambiente com 5% de *outliers* em relação ao ambiente sem *outliers*.

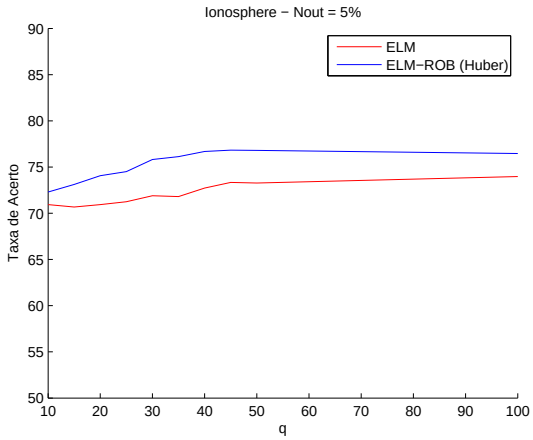
q	ELM ($\lambda = 0.01$)	ROB-ELM (Bisquare)	ROB-ELM (Fair)	ROB-ELM (Huber)	ROB-ELM (Logistic)
10	67.03±5.12	69.77±4.80	**71.06±4.52**	70.57±4.77	70.47±4.03
15	66.39±5.35	70.61±4.94	**70.73±4.79**	68.79±5.03	70.63±4.73
20	65.57±5.71	**72.04±4.77**	70.86±4.06	70.56±4.76	70.90±3.98
25	65.93±4.50	**73.37±4.25**	72.03±3.19	71.64±3.81	71.49±4.08
30	65.83±4.07	72.17±3.69	**73.17±3.09**	72.40±3.48	73.04±3.08
35	66.51±3.82	73.46±3.26	**73.89±2.79**	73.17±2.71	73.81±2.68
40	67.17±2.78	74.56±3.01	74.86±2.94	**75.07±2.78**	74.60±2.90
45	66.96±3.00	75.23±3.73	75.23±3.22	74.67±2.95	**75.63±2.72**
50	67.27±2.57	75.33±3.66	75.67±2.79	75.66±3.14	**75.97±2.74**
100	68.81±2.37	72.99±6.02	**77.07±4.41**	76.90±4.19	77.01±4.50

Tabela 7.8: Comparação de desempenho entre os classificadores ELM e ROB-ELM - BD: Ionosphere ($P_{out} = 10\%$).

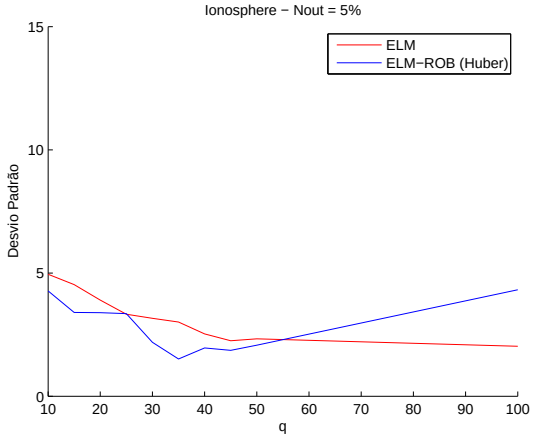
q	ELM ($\lambda = 0.01$)	ROB-ELM (Bisquare)	ROB-ELM (Fair)	ROB-ELM (Huber)	ROB-ELM (Logistic)
10	50.03±5.19	50.93±5.22	**55.16±6.95**	50.51±5.82	53.20±6.93
15	50.07±4.49	51.43±5.53	**55.20±4.86**	50.79±4.64	53.87±4.99
20	48.63±3.63	52.39±4.99	**55.99±6.00**	50.51±4.33	54.74±5.05
25	49.83±4.32	52.31±4.25	**56.46±5.64**	50.90±4.41	54.11±4.49
30	49.49±3.35	53.21±3.47	**56.93±4.38**	51.26±4.00	55.26±3.94
35	49.90±3.22	54.76±4.39	**58.71±4.90**	52.84±4.37	56.69±4.62
40	49.24±2.79	57.53±5.80	**61.11±6.28**	56.37±5.44	59.03±6.09
45	49.93±3.03	58.41±6.83	**62.76±6.92**	58.69±6.73	61.04±5.87
50	50.16±3.02	61.93±6.68	**64.51±7.83**	60.76±7.40	63.61±6.82
100	51.43±3.18	68.94±7.89	**69.91±6.97**	67.03±9.39	68.39±7.67

Tabela 7.9: Comparação de desempenho entre os classificadores ELM e ROB-ELM - BD: Ionosphere ($P_{out} = 20\%$).

Com a base de dados Ionosphere, o classificador robusto apresenta uma diferença em torno de 8% na taxa de acerto já no cenário com 10% de *outliers*, podendo essa diferença ser obser-



(a) Taxa de Acerto.

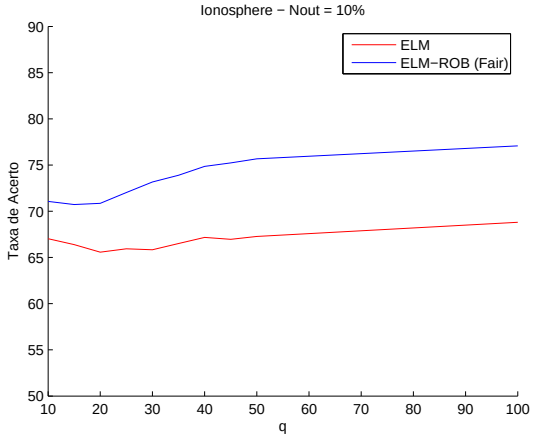


(b) Desvio Padrão.

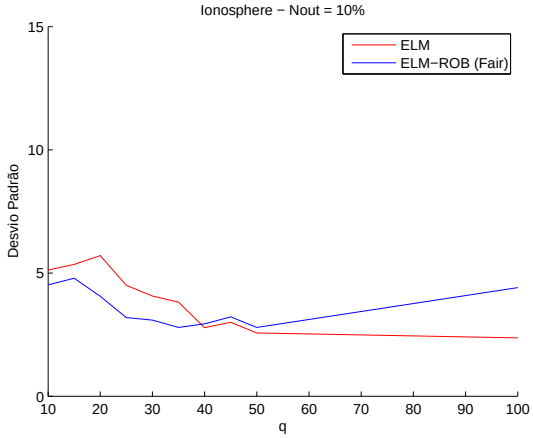
Figura 7.6: Ionosphere - Nout = 5%. (a) Taxa de Acerto. (b) Desvio Padrão.

vada para diferentes funções robustas e quantidades de neurônios ocultos. Com 20% de *outliers*, a diferença na taxa de acerto chega a quase 19% para 100 neurônios ocultos. Observa-se neste experimento que o aumento da quantidade de neurônios ocultos favorece o desempenho do classificador ROB-ELM, ou seja, neste caso, este é um recurso a mais que pode ser utilizado pelo classificador, enquanto que o classificador ELM, na mesma situação, não conseguiu tirar proveito desta possibilidade.

Mais uma vez, as diferenças entre os classificadores ELM e ROB-ELM podem ser vistas graficamente, nas Figuras 7.5, 7.6, 7.7 e 7.8.



(a) Taxa de Acerto.



(b) Desvio Padrão.

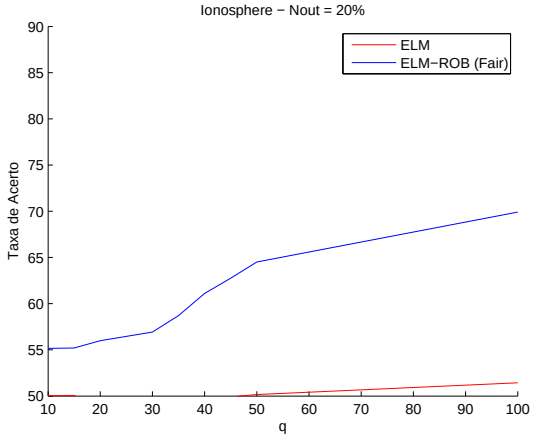
Figura 7.7: Ionosphere - Nout = 10%. (a) Taxa de Acerto. (b) Desvio Padrão.

7.3.3 Classificador ELM-BIP Robusto

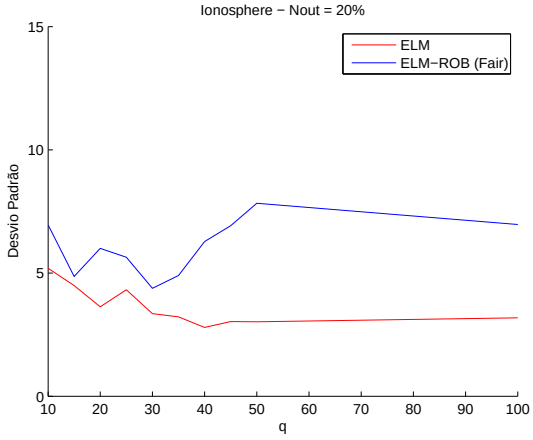
Além da análise de um classificador robusto baseado no ELM (ROB-ELM), foi analisado também neste trabalho, o desempenho de um classificador robusto baseado em uma extensão do ELM, o ELM-BIP, definido como ROB-ELM-BIP.

Se o ELM-BIP é uma opção elegante e recente de melhoria do ELM, o ROB-ELM-BIP vai além, se tornando uma opção de aperfeiçoamento desta melhoria em relação ao ELM original.

Nesta seção é avaliado o desempenho do classificador ROB-ELM-BIP considerando a base de dados Ionosphere. Os resultados apresentados consideram os casos de adição de 5%, 10% e 20% de *outliers* aos dados originais. Os resultados para cada percentual são apresentados nas



(a) Taxa de Acerto.



(b) Desvio Padrão.

Figura 7.8: Ionosphere - Nout = 20%. (a) Taxa de Acerto. (b) Desvio Padrão.

Tabelas 7.3.3, 7.3.3 e 7.3.3, respectivamente.

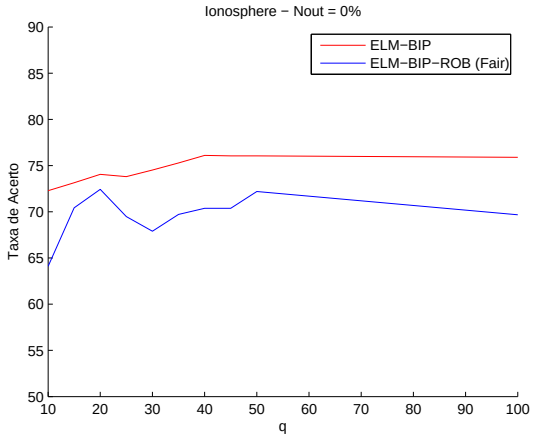
Com o intuito de facilitar a comparação, a Tabela 7.10 apresenta inicialmente os resultados da classificação desta base de dados, através dos classificadores ELM-BIP e ROB-ELM-BIP, em um ambiente sem *outliers*.

Assim como aconteceu entre ROB-ELM e ELM para a base de dados Ionosphere sem *outliers* (Tabela 7.6), o ROB-ELM-BIP com função Bisquare também apresenta um desempenho pior para o caso sem *outliers* em relação ao ELM-BIP. Aqui no caso, no entanto, as outras funções robustas também oferecem taxas de acerto menores que as do ELM-BIP.

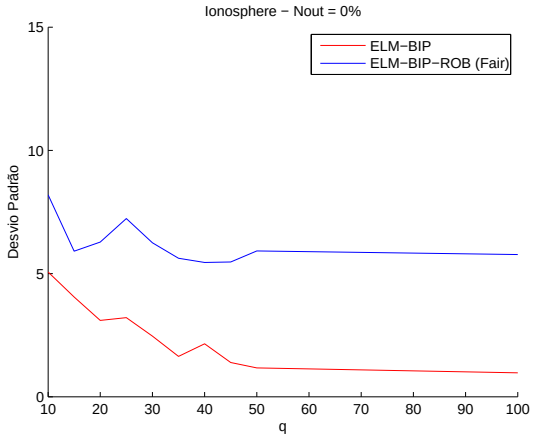
Como era de se esperar devido aos resultados anteriores, entre ELM e ROB-ELM, a partir

q	ELM-BIP($\lambda = 0.01$)	ROB-ELM-BIP(Bisq.)	ROB-ELM-BIP(Fair)	ROB-ELM-BIP(Huber)	ROB-ELM-BIP(Log.)
10	**72.29±5.06**	67.00±7.95	64.10±8.19	66.14±7.75	64.00±8.54
15	**73.14±4.05**	68.38±7.55	70.43±5.91	63.81±8.00	69.29±6.79
20	**74.05±3.10**	69.67±7.10	72.43±6.28	70.29±7.16	71.52±4.79
25	**73.81±3.21**	70.14±5.99	69.48±7.23	68.90±7.02	69.43±6.78
30	**74.52±2.46**	70.71±6.94	67.90±6.25	69.10±5.98	67.62±6.13
35	**75.29±1.64**	70.95±6.27	69.71±5.62	69.81±5.74	69.67±5.65
40	**76.10±2.15**	69.86±5.27	70.38±5.45	71.48±5.42	71.62±5.14
45	**76.05±1.39**	71.05±5.68	70.38±5.47	70.90±6.27	70.81±5.18
50	**76.05±1.17**	70.90±5.84	72.19±5.92	69.90±5.94	70.29±6.59
100	**75.90±0.97**	69.05±6.19	69.67±5.77	70.90±6.84	68.33±6.71

Tabela 7.10: Comparação de desempenho entre os classificadores ELM-BIP e ROB-ELM-BIP - BD: Ionosphere (sem *outliers*).



(a) Taxa de Acerto.



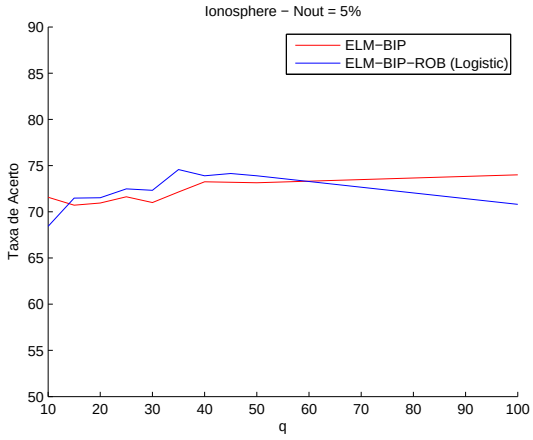
(b) Desvio Padrão.

Figura 7.9: Ionosphere - Nout = 0%. (a) Taxa de Acerto. (b) Desvio Padrão.

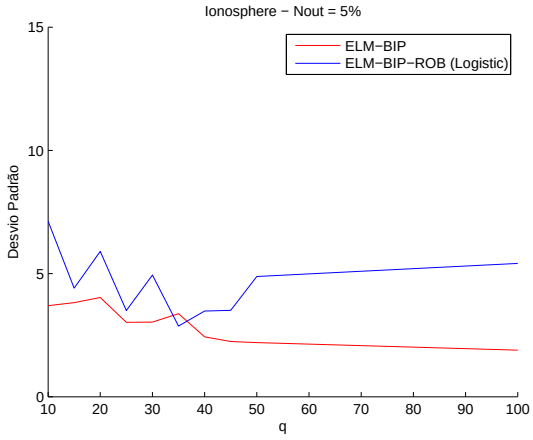
do momento que *outliers* são adicionados aos dados, com 5% e 10%, por exemplo, cai o desempenho do classificador ELM-BIP e melhora o desempenho do classificador ROB-ELM-BIP.

q	ELM-BIP($\lambda = 0.01$)	ROB-ELM-BIP(Bisq.)	ROB-ELM-BIP(Fair)	ROB-ELM-BIP(Huber)	ROB-ELM-BIP(Log.)
10	**71.57±3.70**	69.57±5.48	69.90±6.89	69.81±6.54	68.43±7.12
15	70.71±3.82	71.48±5.79	71.43±6.04	70.57±5.76	**71.48±4.41**
20	70.95±4.03	**72.48±5.22**	71.33±5.47	71.86±5.72	71.52±5.90
25	71.62±3.02	71.86±6.05	72.52±5.72	**73.10±4.88**	72.48±3.50
30	71.00±3.03	71.14±4.97	72.05±5.08	70.95±5.26	**72.33±4.94**
35	72.14±3.37	74.52±3.12	74.48±2.18	73.38±3.47	**74.57±2.87**
40	73.24±2.43	73.10±4.69	71.76±4.89	73.71±3.39	**73.90±3.48**
45	73.19±2.24	73.00±4.40	**74.33±3.47**	73.38±4.68	74.14±3.51
50	73.14±2.20	**74.05±4.02**	72.86±5.33	73.52±4.51	73.90±4.88
100	**74.00±1.89**	72.33±5.39	72.29±6.60	70.86±6.45	70.81±5.41

Tabela 7.11: Comparação de desempenho entre os classificadores ELM-BIP e ROB-ELM-BIP - BD: Ionosphere ($P_{out} = 5\%$).



(a) Taxa de Acerto.



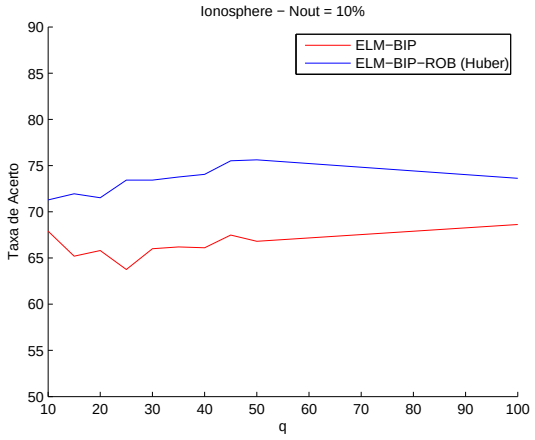
(b) Desvio Padrão.

Figura 7.10: Ionosphere - Nout = 5%. (a) Taxa de Acerto. (b) Desvio Padrão.

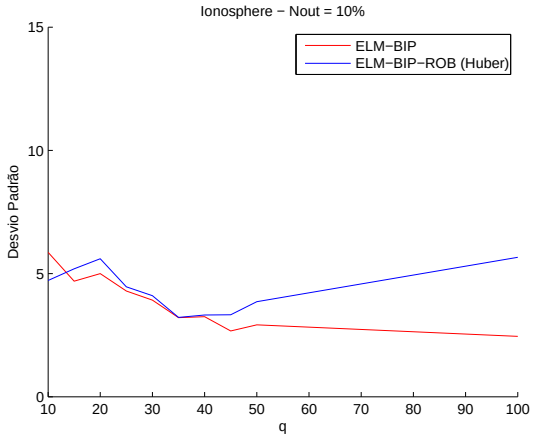
Mais uma vez, a diferença entre os classificadores fica mais visível à medida que aumenta a quantidade de *outliers*, como pode ser comprovado também pelas Figuras 7.9, 7.10, 7.11 e

q	ELM-BIP($\lambda = 0.01$)	ROB-ELM-BIP(Bisq.)	ROB-ELM-BIP(Fair)	ROB-ELM-BIP(Huber)	ROB-ELM-BIP(Log.)
10	67.90±5.86	71.24±3.89	69.95±4.88	**71.29±4.72**	69.48±4.57
15	65.19±4.69	71.71±3.80	69.33±5.04	**71.95±5.19**	70.00±5.37
20	65.81±5.00	71.10±4.71	**73.19±4.19**	71.52±5.60	72.29±4.22
25	63.76±4.29	72.95±4.49	72.19±4.12	**73.43±4.46**	71.86±3.90
30	66.00±3.92	**73.48±3.15**	71.52±3.56	73.43±4.10	72.71±3.64
35	66.19±3.21	73.19±2.95	73.38±2.72	**73.76±3.22**	72.81±1.86
40	66.10±3.25	73.81±4.46	74.29±3.83	74.05±3.32	**74.86±2.99**
45	67.48±2.67	75.00±3.37	74.76±4.23	**75.52±3.33**	74.71±2.63
50	66.81±2.92	75.14±4.15	74.52±3.54	**75.62±3.86**	75.52±3.04
100	68.62±2.45	73.00±5.83	**74.38±6.43**	73.62±5.66	72.90±6.23

Tabela 7.12: Comparação de desempenho entre os classificadores ELM-BIP e ROB-ELM-BIP - BD: Ionosphere ($P_{out} = 10\%$).



(a) Taxa de Acerto.



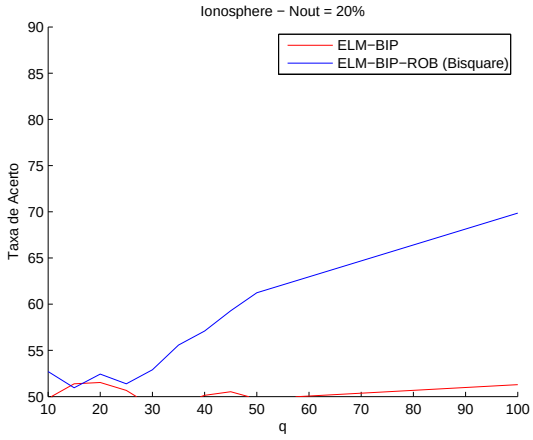
(b) Desvio Padrão.

Figura 7.11: Ionosphere - Nout = 10%. (a) Taxa de Acerto. (b) Desvio Padrão.

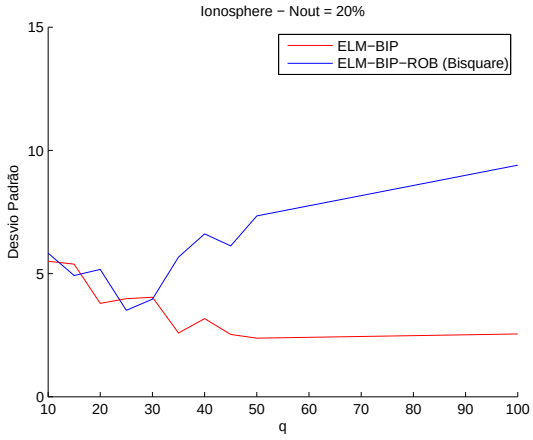
7.12. Com 5% de *outliers*, as taxas de acerto são praticamente similares. Com 10% de *outliers*, a diferença entre as taxas de acerto chega a 10% no caso da função robusta Huber com 25

q	ELM-BIP ($\lambda = 0.01$)	ROB-ELM-BIP (Bisquare)	ROB-ELM-BIP (Fair)	ROB-ELM-BIP (Huber)	ROB-ELM-BIP (Logistic)
10	49.81±5.50	**52.71±5.83**	49.33±5.86	51.43±5.64	51.86±5.31
15	51.38±5.38	50.95±4.92	51.76±4.57	49.52±5.27	**52.67±5.11**
20	51.52±3.79	**52.43±5.17**	51.62±4.05	52.14±4.79	52.19±3.95
25	50.67±3.98	51.38±3.51	50.48±3.57	51.90±4.17	**52.19±5.50**
30	48.95±4.04	52.90±3.96	**53.43±3.82**	52.19±3.53	52.90±3.43
35	49.19±2.59	**55.57±5.67**	55.38±4.76	54.67±4.14	55.43±5.71
40	50.14±3.17	57.10±6.61	**58.57±6.28**	57.10±6.39	56.57±5.68
45	50.52±2.53	59.29±6.12	59.62±7.93	57.76±7.32	**62.38±8.48**
50	49.76±2.38	**61.24±7.34**	60.10±6.25	60.05±6.07	60.81±9.58
100	51.29±2.55	69.86±9.40	**71.43±7.80**	68.95±7.29	69.19±6.90

Tabela 7.13: Comparação de desempenho entre os classificadores ELM-BIP e ROB-ELM-BIP - BD: Ionosphere ($P_{out} = 20\%$).



(a) Taxa de Acerto.



(b) Desvio Padrão.

Figura 7.12: Ionosphere - Nout = 20%. (a) Taxa de Acerto. (b) Desvio Padrão.

neurônios ocultos. Com 20% de *outliers*, esta diferença chega a 20% com a função Fair e 100 neurônios ocultos.

7.4 Resumo do Capítulo

Neste capítulo, inicialmente foram apresentadas as bases de dados utilizadas nos experimentos e o cenário de testes utilizado. Foram então apresentados os resultados para os experimentos realizados com classificadores não lineares, sendo mostrada a superioridade dos classificadores robustos, ROB-ELM e ROB-ELM-BIP, em relação aos classificadores ELM e ELM-BIP, respectivamente.

Foi possível perceber que, em algumas situações, mesmo com um pequeno percentual de *outliers*, os classificadores robustos já se mostram superiores, mas essa superioridade se torna ainda mais visível à medida que aumenta o percentual de *outliers* nos dados.

No próximo capítulo será apresentada a segunda parte dos resultados experimentais realizados com os classificadores ELM e ROB-ELM. O objetivo destes experimentos é utilizar uma metaheurística de otimização baseada em inteligência de enxame para determinar parâmetros ótimos para os classificadores supracitados.

8 *Classificador Não Linear Robusto: Resultados - Parte 2*

Neste capítulo, o classificador não linear robusto, ROB-ELM, é analisado sob o ponto de vista do método Otimização por Enxame de Partículas (*Particle Swarm Optimization*, PSO) (KENNEDY; EBERHART, 1995). O objetivo é utilizar a metaheurística PSO na busca da melhor configuração para o classificador ROB-ELM em cenários com presença de *outliers*, no caso de bases de dados, ou de presença de ruído em bases de imagens.

Em relação às bases de dados, são analisadas as bases Ionosphere, Coluna Vertebral e Wisconsin Diagnostic Breast Cancer, já discutidas nos capítulos anteriores. O PSO, neste caso, é utilizado para buscar os melhores parâmetros de configuração do classificador robusto na presença de 5% e 10% de *outliers* nestas bases.

Em relação às imagens, são apresentadas duas bases de imagens de faces, YALE A e SUSSEX, e a forma como as imagens são preparadas para serem apresentadas ao classificador, incluindo o método de redução de dimensionalidade aplicado e a forma de adição de ruído a estas imagens. Por fim, são apresentados os resultados do PSO em relação à seleção dos parâmetros do classificador ROB-ELM.

Em primeiro lugar, é discutido o método PSO, apresentando as configurações do mesmo para os experimentos realizados neste trabalho.

8.1 Otimização por Enxame de Partículas

A metaheurística PSO é inspirada no comportamento social e auto-organização de revoadas de pássaros e cardumes de peixes. O comportamento social, demonstrado pela troca de informação entre os elementos da população, gera exploração para melhores soluções, enquanto que a aprendizagem individual corresponde ao componente de investigação. Tais características fornecem uma combinação balanceada de busca local e global para o algoritmo (BRATTON; KENNEDY, 2007; PEDERSEN; CHIPPERFIELD, 2010).

Sejam $\mathbf{x}_i \in \mathbb{R}^d$ e $\mathbf{v}_i \in \mathbb{R}^d$, respectivamente, o vetor posição e o vetor velocidade do i-ésimo elemento em um enxame de partículas d-dimensionais, onde $d = N+1$ é o número de variáveis do problema. Sejam também $\mathbf{p}_i \in \mathbb{R}^d$ e $\mathbf{pl}_k \in \mathbb{R}^d$, respectivamente, os vetores de melhor posição individual histórica da i-ésima partícula e a melhor posição histórica da k-ésima vizinhança. Esses vetores são definidos como segue:

$$\begin{aligned}
\mathbf{x}_i &= [x_{i,1}, x_{i,2}, \cdots, x_{i,d}]^T, \\
\mathbf{v}_i &= [v_{i,1}, v_{i,2}, \cdots, v_{i,d}]^T, \\
\mathbf{p}_i &= [p_{i,1}, p_{i,2}, \cdots, p_{i,d}]^T, \\
\mathbf{pl}_k &= [pl_{k,1}, pl_{k,2}, \cdots, pl_{k,d}]^T.
\end{aligned}$$

O algoritmo PSO é implementado de acordo com os seguintes passos para um total de L_{PSO} gerações:

Passo 1 Seja $m = 1$ e faça

$$\begin{aligned}
\mathbf{x}_i(0) &= \mathbf{x}_{min} + (\mathbf{x}_{max} - \mathbf{x}_{min})\mathbf{U}, \\
\mathbf{v}_i(0) &= (\mathbf{x}_{max} - \mathbf{x}_{min})\mathbf{U} - \mathbf{x}_i(0), \\
\mathbf{p}_i(0) &= \mathbf{0}, \\
\mathbf{pl}_k(0) &= \mathbf{0},
\end{aligned}$$

onde $\mathbf{U}$ é um vetor aleatório d-dimensional cujos componentes são uniformemente distribuídos no intervalo $[0,1]$, $\mathbf{0}$ é um vetor nulo d-dimensional e $\mathbf{x}_{max}$ e $\mathbf{x}_{min}$ são, respectivamente, os valores máximo e mínimo permitidos para as variáveis de uma solução.

Passo 2 Avalie todas as partículas do enxame. O vetor $\mathbf{p}_i$ recebe a posição atual da i-ésima partícula, bem como o valor da função fitness correspondente. Seja $\mathcal{N}_k$ o conjunto de todas as partículas dentro da k-ésima vizinhança. Então, o vetor $\mathbf{pl}_k$ e seu valor de fitness recebem a melhor posição e o melhor valor de função fitness dentre as partículas em $\mathcal{N}_k$.

Passo 3 Sejam $\mathbf{x}_i(m)$ e $\mathbf{v}_i(m)$, respectivamente, a posição e a velocidade da i-ésima partícula na iteração m. Então, execute as seguintes equações de atualização:

$$\begin{aligned}
\mathbf{v}_i(m+1) &= \chi\{\mathbf{v}_i(m) + c_1\gamma_1[\mathbf{p}_i(m) - \mathbf{x}_i(m)] + \\
&\qquad + c_2\gamma_2[\mathbf{pl}_k(m) - \mathbf{x}_i(m)]\}, \qquad (8.1) \\
\mathbf{x}_i(m+1) &= \mathbf{x}_i(m) + \mathbf{v}_i(m+1), \qquad (8.2)
\end{aligned}$$

onde χ é o fator de constrição, c_1 e c_2 são constantes positivas chamadas coeficientes de aceleração, enquanto γ_1 e γ_2 são variáveis aleatórias independentes uniformemente distribuídas no intervalo $[0,1]$.

Passo 4 Avalie todas as partículas do enxame.

Passo 5 Para a i-ésima partícula, se $f(\mathbf{x}_i(m+1)) > f(\mathbf{p}_i(m))$, então $\mathbf{p}_i(m) \leftarrow \mathbf{x}_i(m+1)$.

Passo 6 Calcule $\mathbf{p}_{k_{max}}(m) = \arg\max(f(\mathbf{p}_i(m))), \forall i \in \mathcal{N}_k$. Se $f(\mathbf{p}_{k_{max}}(m)) > f(\mathbf{pl}_k(m))$, então $\mathbf{pl}_k(m) \leftarrow \mathbf{p}_{k_{max}}(m)$.

Passo 7 Selecione aleatoriamente uma fração das vizinhanças e realize o passo de busca local baseado em SA nas soluções $\mathbf{pl}_k$ para cada valor selecionado de k.

Passo 8 Seja $m = m+1$. Se $m > L_{PSO}$, pare e apresente o melhor $\mathbf{pl}_k$ de todas as vizinhanças como a melhor solução encontrada. Caso contrário, vá para o Passo 3.

As configurações definidas na implementação do PSO para este trabalho são apresentadas a seguir.

8.1.1 Configurações do PSO

Para o tamanho do enxame, foi adotada a quantidade de 50 partículas que, segundo Bratton & Kennedy (2007), obteve bom desempenho em vários problemas analisados. A dimensão do espaço de busca é um parâmetro que varia de problema para problema, pois está associada à quantidade variáveis envolvidas no problema a ser otimizado. Neste trabalho, a dimensão do espaço foi definida com tamanho igual a 3, no caso do classificador ROB-ELM, e igual a 1, para o caso classificador ELM. A partícula para o ELM-ROB foi formada pela função robusta, seu parâmetro de configuração (k), e a quantidade de neurônios ocultos do classificador. No caso do classificador ELM, o único parâmetro variável foi a quantidade de neurônios ocultos. A partícula neste caso foi formada apenas por este parâmetro.

Para os fatores de aprendizado $\varphi 1$ (cognitivo) e $\varphi 2$ (social), foram utilizados valores iguais a 2,05, como sugeridos também em (BRATTON; KENNEDY, 2007). Como condição de parada para o algoritmo, foi definida uma quantidade máxima de iterações, neste caso 500 iterações ou 10% do máximo de iterações sem que haja melhoria da solução.

A função objetivo foi definida com base na média da taxa de acerto e no desvio padrão do conjunto de resultados de trinta execuções da rede neural. Esta função foi modelada para encontrar soluções que impliquem em uma maior taxa de acerto para o classificador, através da

maximização da média da taxa de acerto da rede, e em soluções estáveis, através da minimização do desvio padrão da série de execuções. Além da média e do desvio padrão, foi considerada uma terceira variável, a quantidade de neurônios ocultos. Esta variável foi inserida para atuar como critério de desempate no caso de duas configurações apresentarem valores semelhantes para a média e o desvio padrão. Desta forma, dentre as configurações empatadas, será considerada melhor, a que possuir a menor quantidade de neurônios ocultos, contribuindo para uma menor complexidade do classificador.

O problema a ser otimizado poderia ser interpretado como multiobjetivo, por tratar-se da maximização da média e da minimização do desvio padrão e da quantidade de neurônios ocultos. Porém, foi utilizada uma formulação monoobjetivo. Nesta formulação, a abordagem multiobjetivo foi substituída por uma única função, onde a média é subtraída pelo desvio padrão e por 1% da quantidade de neurônios ocultos, como mostra a Equação 8.3.

$$fitness = media - (desviopadrao - (0.01 * q)) \tag{8.3}$$

em que q é a quantidade de neurônios ocultos.

Nas próximas seções são apresentados os testes realizados com as bases de dados e as bases de imagens.

8.2 Classificação de Bases de Dados

Um dos pontos fortes dos métodos robustos propostos nesta tese é a possibilidade de escolha de mais de uma função robusta na busca da solução do problema de regressão linear. Além disso, cada função ainda oferece a possibilidade de definição de um parâmetro de configuração (k) responsável por orientar a sua execução em relação aos pesos de cada ponto no conjunto de dados. Desta forma, o conjunto de parâmetros configuráveis em uma ROB-ELM contém três valores, sendo eles, quantidade de neurônios ocultos, função robusta e k, o que torna sua configuração bem mais flexível que a de uma ELM padrão, onde pode ser variada apenas a quantidade de neurônios ocultos.

Para comprovar esta questão, foram realizados testes com as redes ELM e ROB-ELM utilizando o método PSO. Para tal experimento foi utilizada a bases de dados Ionosphere com 5% e 10% de *outliers*. Os resultados podem ser vistos na Tabela 8.1.

Um primeiro ponto a considerar nesta análise é a quantidade necessária de neurônios ocultos na definição dos classificadores. No caso do ELM, ocorre uma explosão da quantidade de

neurônios ocultos. O que se percebe é que o ELM sofre uma estagnação no seu desempenho ao atingir uma taxa de acerto por volta de 74%, para o caso de 5% de *outliers*, e 69%, para o caso de 10% de *outliers*. Estas taxas, no entanto, já são obtidas com uma quantidade de 100 neurônios ocultos (resultados apresentados nas Tabelas 7.6 e 7.7 do capítulo anterior). Assim, na tentativa de melhorar a taxa de desempenho e contando com a possibilidade de ajustar apenas a quantidade de neurônios ocultos, o PSO termina por explodir esta quantidade na tentativa de um ganho mínimo da taxa resultante.

Além da diferença significativa da quantidade de neurônios definida para cada classificador, as taxas de acerto alcançadas pelo ROB-ELM são maiores que as do ELM, sendo a diferença de 5% no caso com 5% de *outliers* e de 7% no caso com 10% de *outliers*.

É possível perceber também que, com o aumento da quantidade de *outliers*, ou seja, com o aumento da complexidade dos dados a serem classificados, a função robusta escolhida pelo PSO para a rede ROB-ELM é diferente da função MQO ou é a função MQO com parâmetro k bem maior que 1, o que faz com que esta função trabalhe com a ideia de pesos adicionados aos padrões, descaracterizando o método padrão dos mínimos quadrados. Esta observação pode ser confirmada na Tabela 8.1.

Tabela 8.1: Comparação de desempenho, com base em configurações do PSO, para a classificação da base Ionosphere pelos classificadores ELM e ROB-ELM.

Método	q	Função Robusta	k	Taxa de acerto	Desvio padrão
P_out = 5%					
ELM	1845	–	–	74.40	1.43
ROB-ELM	85	MQO	5.21	79.17	2.51
P_out = 10%					
ELM	1423	–	–	69.71	1.75
ROB-ELM	34	Logistic	0.1	77.28	1.12

Para a situação com 5% de *outliers*, onde é escolhida a função MQO, o valor de k é de 5,21. Com 10% de *outliers*, a própria função robusta passa a ser diferente, sendo definida a função Logistic.

Para comprovar a redução da quantidade de neurônios ocultos necessárias ao ROB-ELM em relação ao ELM, mais uma comparação foi realizada entre os classificadores. Desta vez, foi utilizado um outro cenário, com *outliers* afastados da fronteira e pontos de testes espalhados em qualquer lugar. Os resultados podem ser vistos na Tabela 8.2. Além da redução da quantidade de neurônios ocultos, comprova-se também, um aumento na taxa de acerto e um decréscimo (exceto em no último caso) no desvio padrão, em favor do ROB-ELM.

Um outro experimento, agora não mais para comparar os classificadores, foi realizado apenas para analisar a escolha da função robusta no casos de classificação das bases de dados

Tabela 8.2: Comparação de desempenho, com base em configurações do PSO, para a classificação da base Coluna Vertebral e Ionosphere pelos classificadores ELM e ROB-ELM.

BD	Método	q	Função Robusta	k	Taxa de acerto	Desvio padrão
N_out = 5%						
Coluna	ELM	15	–	–	94.44	2.71
	ROB-ELM	10	Welsch	2.41	95.32	2.72
Ionosphere	ELM	72	–	–	87.71	2.88
	ROB-ELM	30	Fair	8.77	87.57	2.71
N_out = 10%						
Coluna	ELM	13	–	–	91.40	3.59
	ROB-ELM	10	Andrews	0.87	94.56	2.99
Ionosphere	ELM	95	–	–	78.97	3.85
	ROB-ELM	75	Andrews	3.04	81.51	3.98

Coluna Vertebral e Wisconsin Diagnostic Breast Cancer na presença de *outliers*.

A Tabela 8.3 apresenta os resultados. Para a base Coluna Vertebral, com 5% de *outliers*, é escolhida a função MQO, mas com valor de k igual a 6,64, bem maior que 1, e para todos os outros casos, a função robusta escolhida pelo PSO é diferente da MQO, sendo definidas as funções Andrews e Welsch.

Tabela 8.3: Análise das funções robustas selecionadas pelo PSO para a classificação das bases Coluna Vertebral e WDBC através do ROB-ELM.

BD	q	Função Robusta	k	Taxa de acerto	Desvio padrão
P_out = 5%					
Coluna	16	MQO	6.64	83.96	3.09
WDBC	128	Welsch	2.52	59.66	2.38
P_out = 10%					
Coluna	17	Welsch	10	82.88	4.27
WDBC	175	Andrews	1.23	59.75	2.40

Além de testes com bases de dados, foram realizados também testes com bases de imagens, que são discutidos a seguir.

8.3 Classificação de Bases de Imagens

Esta seção trata da tarefa de classificação, utilizando o ROB-ELM, de dois bancos de imagens de faces. São apresentadas as imagens, o método de redução de dimensionalidade aplicado a elas, a preparação das imagens antes da apresentação ao classificador e, por fim, são discutidos os resultados dos testes do classificador com base no PSO.

8.4 Bases de Imagens

Para os experimentos com imagens, foram selecionadas duas bases de imagens de faces bastante conhecidas na literatura: YALE A e SUSSEX. As caraterísticas destas bases são apre-

sentadas a seguir.

8.4.1 YALE A

O banco de imagens YALE A (BELLHUMER et al., 1997) contém 165 imagens em escala de cinza, de tamanho 243 $\times$ 320 pixels, no formato GIF. As imagens estão divididas entre 15 indivíduos, cada um contendo igualmente 11 imagens. Cada uma das onze imagens de cada indivíduo corresponde a diferentes expressões faciais: tristeza, sonolência, surpresa, piscar de olhos e alegria; e de diferentes configurações: iluminação frontal, expressão normal sem óculos, expressão normal com óculos, iluminação lateral à esquerda, iluminação lateral à direita. A Figura 8.1 apresenta os 15 indivíduos do banco de imagens e a Figura 8.2 apresenta as 11 variações de um mesmo indivíduo.

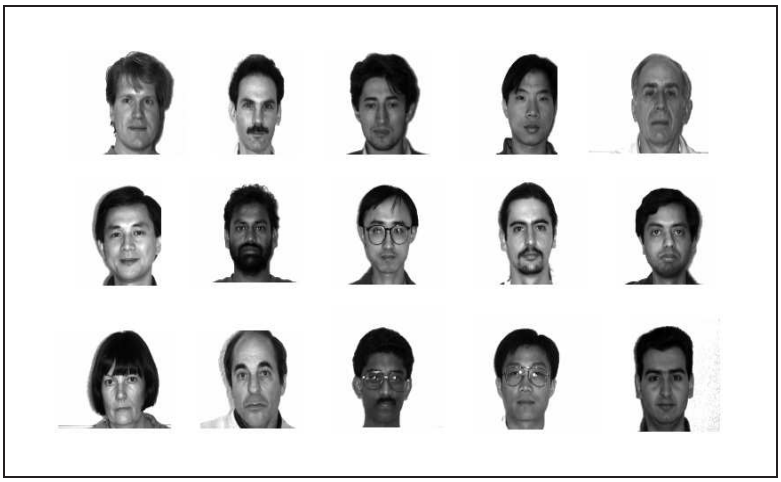


Figura 8.1: Indivíduos que compõem o banco de imagens YALE A.

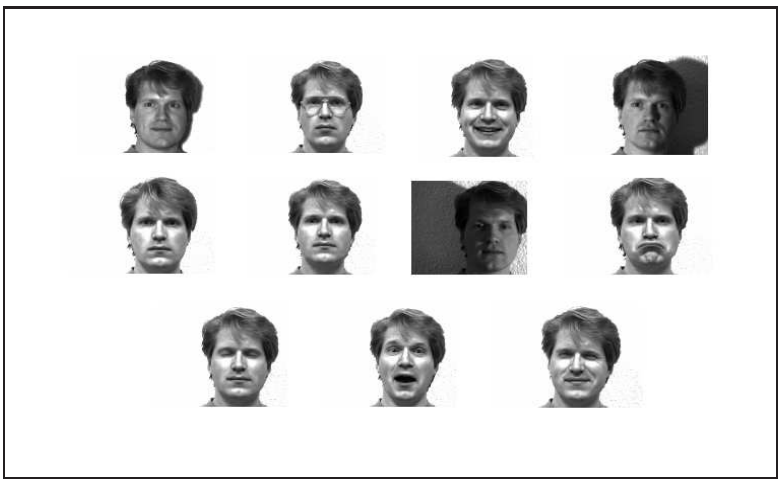


Figura 8.2: Amostra das 11 variações faciais presentes no banco de imagens YALE A.

8.4.2 SUSSEX

O banco de imagens SUSSEX foi organizado por Howell (September 1997). Esse banco contém 100 imagens em escala de cinza, de tamanho 384×287 pixels, no formato Sun Standard Rasterfile. As imagens estão divididas entre 10 indivíduos, cada um contendo igualmente 10 imagens. Cada uma das dez imagens corresponde a uma posição da câmera que varia de 0° a 90° com um passo de 10° entre as imagens. Para cada indivíduo existe uma vista frontal de sua face, além das variações de pose. O incremento angular para cada pose é de 10° em relação à posição frontal. A Figura 8.3 apresenta os 10 indivíduos do banco de imagens e a Figura 8.4 apresenta as 10 variações de um mesmo indivíduo.

Figura 8.3: Indivíduos que compõem o banco de imagens SUSSEX.

Figura 8.4: Amostra das 10 variações faciais presentes no banco de imagens SUSSEX.

A Tabela 8.4 resume as principais características apresentadas para cada banco de imagens.

A manipulação de imagens introduz alguns desafios que não são encontrados na manipulação de dados. Um deles é a necessidade de tratamento de uma grande quantidade de infor-

BD	Qtd. Indivíduos	Qtd. Imagens/Indiv.	Cor	Tamanho (pixels)	Formato
YALE A	15	11	Escala de cinza	243×320	GIF
SUSSEX	10	10	Escala de cinza	384×287	Sun Standard Rasterfile

Tabela 8.4: Características dos bancos de imagens utilizados nos testes.

mações. Para tanto, muitas vezes, são aplicadas às imagens métodos de redução de dimensionalidade. O método utilizado neste trabalho foi o de Fatoração de Matrizes Não-negativas (Non-negative Matrix Factorization, NMF).

8.5 Redução de Dimensionalidade

Métodos de redução de dimensionalidade têm duas vantagens principais. Primeiro, reduzem o tamanho da informação, característica importante para lidar com imagens. Segundo, possibilitam a projeção da informação em um outro espaço, diferente do espaço de entrada. No novo espaço projetado, existe a possibilidade de serem privilegiadas caracteríticas importantes para a diferenciação dos padrões, realizando, portanto, uma forma de análise das características dos dados. A seguir é discutido o método de redução de dimensionalidade utilizado neste trabalho, o NMF.

8.5.1 Fatoração de Matrizes Não-Negativas

Fatoração em Matrizes Não-Negativas (*Non-Negative Matrix Factorization*, NMF), é um método para aprender partes de faces e características semânticas de textos. Seu propósito é diferente de outros métodos que representam a informação de forma holística (LEE; SEUNG, 1999), como por exemplo, Análise de Componentes Principais, que usa *eigenfaces* como imagens base, as quais representam faces inteiras.

Um propriedade importante de NMF é o uso de restrições de não negatividade. Isso faz com que não seja possível usar valores negativos para representar a informação e usar combinações subtrativas da informação. Apenas combinações aditivas são permitidas, o que leva a uma representação baseada em partes da informação sendo tratada. Imagens base em NMF correspondem a características localizadas, tais como olhos, boca, nariz, e assim por diante. Essa é uma forma interessante de representar faces, pois ela lida com a noção intuitiva de que uma face é composta de partes.

NMF usa fatoração baseada em matriz para representar as imagens, uma para as imagens base e uma para os coeficientes da combinação linear. A fatoração é da forma $V \approx WH$ (LEE;

SEUNG, 1999), onde

$$V_{i\mu} \approx (WH)_{i\mu} = \sum_{a=1}^{r} W_{ia} H_{a\mu} \tag{8.4}$$

A matriz V é uma matriz $n \times m$ que contém m imagens de faces com n pixels não negativos em cada uma. As r colunas de W representam as imagens base, e as m colunas de H representam codificaçoes de imagens base. Cada codificação representa uma imagem em V. Geralmente, r é escolhido tal que $(n+m)r < n \cdot m$. Dessa forma, o produto WH representa uma versão comprimida da matriz original V (LEE; SEUNG, 2000).

NMF trabalha apenas com operações de adição. Por possuir restrições de não negatividade, ele não permite elementos negativos nas matrizes fatores W e H, ou seja, todos os elementos devem ser iguais ou maiores que zero. Isso leva a imagens base que contêm informação não-global, tal como tipos diferentes de olhos, bocas e sobrancelhas. A imagem final é composta por uma combinação linear dessas partes. Uma boa aproximação, no entanto, pode ser obtida apenas se os vetores base descobrirem a estrutura latente nos dados (LEE; SEUNG, 2000).

A técnica NMF pode ser implementada por diferentes algoritmos. Lee & Seung (2000) propõem duas formas de atualizar W e H. Em cada iteração do algoritmo, os novos valores de W ou H são encontrados através da multiplicação do valor atual por algum fator que depende da qualidade da aproximação 8.4. Um dos algoritmos usa o quadrado da distância Euclidiana entre duas matrizes não negativas como uma função custo que quantifica a qualidade da aproximação. Portanto, o problema é minimizar $(V - WH)^2$ em relação a W e H, sujeito às restrições $W; H \geq 0$.

A distância Euclidiana é não-crescente sob as seguintes regras de atualização (LEE; SEUNG, 2000).

$$H_{a\mu} \leftarrow H_{a\mu} \frac{(W^T V)_{a\mu}}{(W^T WH)_{a\mu}} \tag{8.5}$$

$$W_{ia} \leftarrow W_{ia} \frac{(VH^T)_{ia}}{(WHH^T)_{ia}} \tag{8.6}$$

NMF é usada neste trabalho para pré-processar imagens de faces, decompondo-as em fatores e representando as imagens originais por uma combinação linear destes fatores antes de apresentá-las ao classificador.

8.6 Preparação das Imagens

O objetivo dos experimentos realizados com imagens é o de avaliar a escolha, pelo PSO, de parâmetros para o classificador ROB-ELM, em situações onde as imagens não apresentam qualquer pertubação e em situações onde as mesmas imagens recebam a adição de ruído, tornando a tarefa de classificação mais complexa.

Inicialmente, o tamanho das imagens foi reduzido para 50×50 pixels. Em seguida, o ruído foi adicionado às imagens através do comando *imnoise* do Matlab, sendo escolhido o tipo de ruído Sal e Pimenta. O comando recebe um parâmetro de entrada D que define a densidade do ruído, ou seja, aproximadamente, a quantidade de pixels afetados pela adição do ruído é igual a $D * quantidadetotaldepixelsdaimagem$. O valor *default* de D é igual a 0,05, assim, foram definidas para os testes, imagens com nível densidade de ruído igual a 0,05.

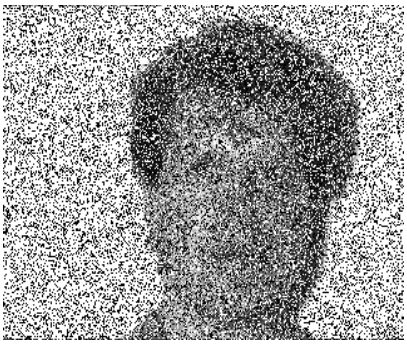
Após a adição de ruído, e antes do processo de classificação propriamente dito, as imagens passaram pela fase de pré-processamento de redução de dimensionalidade. O método escolhido para tanto foi o NMF com a definição de 50 fatores. Acredita-se que este formato foi o que forneceu melhor representação das imagens no espaço de projeção em termos de extração de informação de variabilidade e, consequentemente, de informação significativa para o processo de classificacão.

A Figura 8.5 apresenta uma imagem sem ruído da base YALE A, seguida da mesma imagem com ruído Sal e Pimenta adicionado. Os resultados dos testes com esta base encontram-se na Tabela 8.5. Da mesma forma, a Figura 8.6 apresenta uma imagem sem ruído da base SUSSEX, seguida da mesma imagem com ruído Sal e Pimenta adicionado. Os resultados dos testes para esta base encontram-se na Tabela 8.6.

Como discutido anteriormente, o objetivo deste experimento era analisar o comportamento do classificador ROB-ELM ao analisar dados na presença de ruído. O PSO deveria, portanto, definir, para cada situação (imagens sem e com ruído), qual a melhor função robusta, o melhor parâmetro de configuração k desta função, e a melhor quantidade de neurônios ocultos para um classificador ROB-ELM.

A conclusão interessante à qual foi possível chegar, foi a de que, nos dados sem ruído, para as duas bases de imagens, a função robusta definida pelo PSO foi a função MQO, que tem como base o método dos mínimos quadrados, comportando-se de forma exatamente igual a esse método quando $k = 1$. Nos dados com ruído, no entanto, a função robusta definida foi a função Fair. Desta forma, a perturbação do ruído nos dados fez com que a função MQO não fosse mais a ideal para a classificação das imagens ruidosas.

(a) Face sem ruído.



(b) Face com ruído Sal e Pimenta (parâmetro=0.5).

Figura 8.5: Face da base de dados YALE A. (a) Face sem ruído. (b) Face com ruído Sal e Pimenta (parâmetro=0.5).

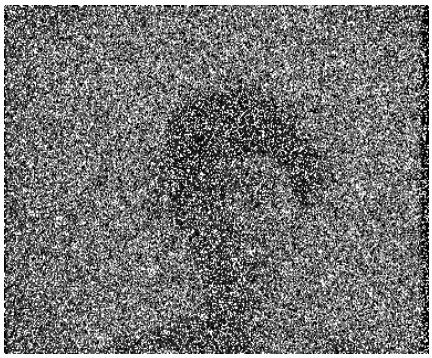
Tabela 8.5: Comparação da função robusta escolhida pelo PSO na classificação da base de faces YALE A com e sem ruído.

Dados	q	Função robusta	k	Taxa de acerto	Desvio padrão
Sem ruído	35	MQO	0.69	98.88%	1.48
Com ruído	52	Fair	3.12	95.35%	3.15

Tabela 8.6: Comparação da função robusta escolhida pelo PSO na classificação da base de faces SUSSEX com e sem ruído.

Dados	q	Função robusta	k	Taxa de acerto	Desvio padrão
Sem ruído	46	MQO	0.16	98.5%	2.67
Com ruído	51	Fair	0.26	89.83%	4.25

(a) Face sem ruído.



(b) Face com ruído Sal e Pimenta (parâmetro=0.5).

Figura 8.6: Face da base de dados SUSSEX. (a) Face sem ruído. (b) Face com ruído Sal e Pimenta (parâmetro=0.5).

8.7 Resumo do Capítulo

O capítulo apresentou formas de utilizar a metaheurística PSO para analisar a configuração do classificador ROB-ELM. Para tanto, foram utilizadas bases de dados e base de imagens de faces. Em alguns casos, através de resultados obtidos pelo PSO, foi feita também a comparação entre o desempenho do ELM e do ROB-ELM.

Com os experimentos realizados, foi possível perceber que, com o aumento da complexidade nos dados a serem classificados, seja por adição de *outliers* às bases de dados, ou por adição de ruído às imagens, a tendência do PSO é escolher uma função robusta diferente da

função MQO. Este resultado é uma confirmação interessante das idéias apresentadas nesta tese, de que o arcabouço de estimadores-*M* é de fato útil no projeto de classificação de padrões robustos a *outliers*.

9 *Conclusão*

Esta tese apresenta uma revisão sobre regressão linear e aborda a questão da estimação de parâmetros neste tipo de problema. É discutido o conceito de amostras discrepantes (*outliers*) e a influência destes pontos em problemas de análise de regressão.

Mínimos quadrados ordinário (MQO) é um método de estimação de parâmetros fundamentado no cálculo da pseudoinversa (ou inversa generalizada de Moore-Penrose). Este é um método comumente usado no projeto de classificadores. No entanto, ele possui diversas restrições, sendo uma das principais, sua sensibilidade à presença de *outliers* nos dados. Como uma alternativa ao MQO, introduzimos neste trabalho um arcabouço teórico, conhecido como estimação-M, que tem como base os estimadores-*M*.

O conceito de estimação-M (onde M significa 'do tipo Máxima Verossimilhança') foi introduzido por Huber (1964), como um processo de esitmação de parâmetros, e sugere que esta estimação tenha como base a minimização de uma outra função dos erros, em vez da soma de seus quadrados.

Em problemas de regressão, o método MQO trata da mesma forma todos os padrões de um determinado conjunto de dados. Os estimadores-M, por outro lado, atribuem pesos diferentes aos padrões, dependendo do resíduo gerado por cada um e, assim, diminuem a influência de pontos com resíduos muito grandes, como é o caso das amostras discrepantes, ou *outliers*. Desta forma, os estimadores-M se mostram muito mais eficientes no tratamento de dados que contenham tais amostras.

O uso do arcabouço de estimação-M para problemas de regressão já é conhecido. Este trabalho, no entanto, propôs a aplicação deste arcabouço a problemas de classificação. São apresentadas propostas de classificadores robustos, através do uso de estimadores-M, tanto lineares quanto não-lineares, juntamente com experimentações que validam o uso da técnica robusta e comprovam sua eficiência na classificação de conjuntos de dados envolvendo *outliers*.

O uso de otimização por enxame de partículas (*particle swarm optimization*) para otimizar os parâmetros do classificador ELM robusto, foi um recurso a mais para comprovar a eficiência

deste classificador, principalmente em casos de classificação mais complexos, envolvendo a presença de *outliers* em bases de dados ou de ruído em bases de imagens.

Através dos experimentos realizados foi possível constatar, em relação a problemas de classificação de padrões envolvendo *outliers*, a superioridade dos classificadores robustos em comparação com os classificadores padrões baseados no MQO. No caso de classificadores ELM, foi mostrado também que, neste tipo de problema, um classificador ELM robusto, em comparação com um classificador ELM padrão, requer uma menor quantidade de neurônios ocultos na definição de sua arquitetura.

9.1 Resumo das Contribuições da Tese

As principais constriubições desta tese são listadas a seguir.

Proposta de Estimação-M para Classificação Robusta - Os estimadores-M são bastante utilizados em problemas de regressão. No entanto, sua aplicação a problemas de classificação supervisionada de padrões é uma ideia nova. Esta tese propõe, portanto, a combinação entre o uso de estimação-M e classificadores de padrões, neurais ou estatísticos. Foram avaliados neste trabalho modelos de redes neurais em problemas de classificação de padrões na presença de *outliers*. Até onde foi possível investigar, esta é a primeira vez que o desempenho de classificadores OLAM e ELM está sendo avaliado sob a presença de *outliers*, analisando a robustez destes modelos em problemas de classificação de padrões na presença destes pontos considerados discrepantes.

Discussão Prática dos Conceitos - É discutida a aplicação prática, de forma didática, dos conceitos abordados através da apresentação de códigos e análises de implementações em ambientes de computação científica, tais como Matlab e Octave.

Proposta do Classificador OLAM Robusto - É proposto um classificador OLAM (*Optimal Linear Associative Memory*) robusto, denominado OLAM Robusto (ROLAM, do inglês *Robust OLAM*), que tem como base os conceitos de regressão linear robusta, através da aplicação dos estimadores-M, com o intuito de definir um classificador linear menos sensível a *outliers*. A idéia básica desta proposta é substituir o método de estimação da matriz de pesos da camada de saída de uma rede neural, baseado no MQO, por um método baseado no uso de estimação-M.

Proposta do Classificador ELM Robusto - É proposto um classificador ELM (*Extreme Learning Machine*) robusto, denominado ELM Robusto (ROB-ELM, do inglês *Robust ELM*).

Este é um classificador não-linear que, como o classificador linear robusto, tem como base os conceitos de regressão linear robusta, através da aplicação dos estimadores-M. O método MQO é substituído pela proposta do arcabouço de estimação-M e este classificador mostra uma maior robustez a *outliers* quando comparado ao classificador ELM padrão.

Proposta do Classificador ELM-BIP Robusto -É proposto um classificador ELM-BIP (*Extreme Learning Machine*/*Batch Intrinsic Plasticity*) robusto, denominado ELM-BIP Robusto (ROB-ELM-BIP, do inglês *Robust ELM-BIP*). O método BIP é uma proposta de melhoria de treinamento para uma rede ELM, e a adição de robustez a este classificador é uma evolução desta melhoria, levando ao projeto de um classificador do tipo ELM ainda mais poderoso.

Aplicação do PSO a Classificadores Robustos - É proposta uma forma de avaliar o classificador ELM robusto e demonstrar sua flexibilidade através do uso da metaheurística de otimização por enxame de partículas. Esta flexibilidade se dá em decorrência dos recursos oferecidos pelos estimadores-M, como diversas funções objetivo bem como diversos valores para seus parâmetros de configuração. Em comparação com o ELM padrão, o classificador ELM robusto apresenta uma maior capacidade de se adequar a problemas de classificação mais complexos, tais como os que envolvem *outliers* em bases de dados e ruído em bases de imagens. Além disso, um fato significativo é que o classificador ELM robusto requer uma menor quantidade de neurônios ocultos, na construção de sua arquitetura, que o classificador padrão.

9.2 Propostas para Trabalhos Futuros

Esta tese possui desdobramentos que podem ser investigados como uma continuação do trabalho de pesquisa aqui apresentado. São listados a seguir algumas potenciais linhas de pesquisa que podem ser desenvolvidas com o objetivo de colocar em prática esta continuidade:

- Testar os classificadores propostos em problemas de classificação multiclasses.
- Estender a abordagem proposta nesta tese para problemas de classificação espaço-temporal via redes de estados de ecos, tais como classificação de anomalias cardíacas a partir do sinal de eletrocardiograma e reconhecimento da fala.
- Avaliar a abordagem proposta nesta tese em problemas de detecção de novidades, a fim

de desenvolver métricas que possam identificar um outlier como uma amostra útil ou descartável.

Referências Bibliográficas

ANDERSON, J. A simple neural network generating an interactive memory. *Mathematical Biosciences*, v. 14, n. 3-4, p. 197–220, 1972.

ANDREWS, D. F. A robust method for multiple linear regression. *Technometrics*, v. 16, n. 4, p. 523–531, 1974.

AUGUSTEIJN, M. F.; FOLKERT, B. A. Neural network classification and novelty detection. *International Journal of Remote Sensing*, v. 23, n. 14, p. 2891–2902, 2002.

BADDELEY, R. et al. Responses of neurons in primary and inferior temporal visual cortices to natural scenes. In: *Proc. R. Soc. Lond.* [S.l.: s.n.], 2005. p. 1775–1783.

BAEK, D.; OH, S.-Y. Improving optimal linear associative memory using data partitioning. In: *Proceedings of the 2006 IEEE International Conference on Systems, Man, and Cybernetics (SMC'06)*. [S.l.: s.n.], 2006. v. 3, p. 2251–2256.

BAI, Z. D.; WU, Y. General M-estimation. *Journal of Multivariate Analysis*, v. 63, p. 119–135, 1997.

BARNETT, V.; LEWIS, T. *Outliers in Statistical Data*. 3a.. ed. [S.l.]: John Wiley & Sons, 1994.

BARRETO, G. A. *Redes Neurais Não-Supervisionadas para Processamento de Seqüências Temporais*. Dissertação (Mestrado) — Departamento de Engenharia Elétrica, Universidade de São Paulo, São Carlos, SP, 1998.

BARRETO, G. A.; FROTA, R. A. A unifying methodology for the evaluation of neural network models on novelty detection tasks. *Pattern Analysis and Applications*, v. 16, n. 1, p. 83–972, 2013.

BELLHUMER, P. N.; HESPANHA, J.; KRIEGMAN, D. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence, Special Issue on Face Recognition*, v. 17, n. 7, p. 711–720, 1997.

BEN-GAL, I. Outlier detection. In: MAIMON, O.; ROKACH, L. (Ed.). *Data Mining and Knowledge Discovery Handbook*. [S.l.]: Springer, 2005. p. 131–146.

BLATNÁ, D. Outliers in regression. In: *Proceedings of the 9th International Scientific Conference on Applications on Mathematics and Statistics in Economy (AMSE'2006)*. [S.l.: s.n.], 2006. p. 1–6.

BOCCATO, L. *Novas Propostas e Aplicaçõs de Redes Neurais com Estados de Eco*. Tese (Doutorado) — Faculdade de Engenharia Elétrica e de Computação (FEEC), Universidade de Campinas, São Paulo, 2013.

BOULESTEIX, A.-L. PLS dimension reduction for classification with microarray data. *Statistical Applications in Genetics and Molecular Biology*, v. 3, n. 1, p. 1–32, 2004.

BRANHAM, R. L. Alternatives to least squares. *The Astronomical Journal*, v. 87, n. 6, p. 928–937, 1982.

BRATTON, D.; KENNEDY, J. Defining a standard for particle swarm optimization. In: *Proceedings of the IEEE Swarm Intelligence Symposium*. Honolulu, Hawaii: [s.n.], 2007. p. 120–127.

CHATTERJEE, S.; MÄCHLER, M. Robust regression: a weighted least squares approach. *Communications in Statistics - Theory and Methods*, v. 26, n. 6, p. 1381–1394, 1997.

CHERKASSKY, V.; FASSETT, K.; VASSILAS, N. Linear algebra approach to neural associative memories and noise performance of neural classifiers. *IEEE Transactions on Computers*, v. 40, n. 12, p. 1429–1435, 1991.

CUDMORE, R. H.; DESAI, N. S. Intrinsic plasticity. *Scholarpedia*, v. 3, n. 2, p. 1363, 2008.

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.

DUDA, R. O.; HART, P. E.; STORK, D. G. *Pattern Classification*. 2nd. ed. [S.l.]: John Wiley & Sons, 2006.

EICHMANN, G.; KASPARIS, T. Pattern classification using a linear associative memory. *Pattern Recognition*, v. 22, n. 6, p. 733–740, 1989.

EMMERICH, C.; REINHART, F.; STEIL, J. Recurrence enhances the spatial encoding of static inputs in reservoir networks. In: *Proceedings of the 20th International Conference on Artificial Neural Networks*. [S.l.]: Springer, 2010. LNCS 6353, p. 148–153.

FAWZYA, A.; MOKHTAR, H. M.; HEGAZY, O. Outliers detection and classification in wireless sensor networks. *Egyptian Informatics Journal*, 2013.

FOX, J. *Applied Regression Analysis, Linear Models, and Related Methods*. [S.l.]: Sage Publications, 1997.

FOX, J. *Robust Regression: Appendix to An R and S-PLUS Companion to Applied Regression*. [S.l.], 2002.

FRANK, A.; ASUNCION, A. *UCI Machine Learning Repository*. 2010. Disponível em: <http://archive.ics.uci.edu/ml>.

FREEDMAN, D.; PISANI, R.; PURVES, R. *Statistics*. 4a.. ed. [S.l.]: W. W. Norton & Company, 2007.

GAUSS, C. F. *Theoria Motus Corporum Coelestium in Sectionibus Conicis Solem Ambientium*. [S.l.]: Perthes et I. H. Besser, Hamburgi, 1809.

GLAVIN, F. G.; MADDEN, M. G. Analysis of the effect of unexpected outliers in the classification of spectroscopy data. In: *Proceedings of the 20th Irish Conference on Artificial Intelligence and Cognitive Science (AICS'09)*. [S.l.: s.n.], 2010. p. 124–133.

GOLUB, G. H.; VAN LOAN, C. F. *Matrix Computations*. 3a.. ed. [S.l.]: The Johns Hopkins University Press, 1996.

HAMPEL, F. Robust statistics: a brief introduction and overview. In: *Robust Statistics and Fuzzy Techniques in Geodesy and GIS*. [S.l.: s.n.], 2001. p. 1–6.

HEBB, D. *The Organization of Behavior*. [S.l.]: New York: Wiley, 1949.

HILL, R. W.; HOLLAND, P. W. Two robust alternatives to least-squares regression. *Journal of the American Statistical Association*, v. 72, n. 360, p. 828–833, 1977.

HINES, W. W. et al. *Probabilidade e Estatística na Engenharia*. Quarta. [S.l.]: LTC, 2006.

HODGE, V. J.; AUSTIN, J. A survey of outlier detection methodologies. *Artificial Intelligence Review*, v. 22, n. 2, p. 85–126, 2004.

HOERL, A. E.; KENNARD, R. W. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, v. 42, n. 1, p. 80–86, 1970.

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

HOWELL, A. J. *Automatic Face Recognition using Radial Basis Function Networks*. Tese (Doutorado) — University of Sussex, Brighton, UK, September 1997.

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.

HUANG, G.-B.; ZHU, Q.-Y.; SIEW, C.-K. Extreme learning machine: Theory and applications. *Neurocomputing*, v. 70, p. 489–501, 2006.

HUBER, P. J. Robust estimation of a location parameter. *Annals of Mathematical Statistics*, v. 35, n. 1, p. 73–101, 1964.

HUNT, B. et al. Synthesis of a nonrecurrent associative memory model based on a nonlinear transformation in the spectral domain. *IEEE Transactions on Neural Networks*, v. 4, n. 5, p. 873–878, 1993.

JOHNSON, W.; GEISSER, S. A predictive view of the detection and characterization of influential observations in regression analysis. *Journal of the American Statistical Association*, v. 78, p. 137–144, 1983.

KENNEDY, J.; EBERHART, R. C. Particle swarm optimization. In: *Proceedings of the IEEE International Conference on Neural Networks*. Piscataway, NJ, USA: [s.n.], 1995. v. 4, p. 1942–1948.

KIM, H.-C.; GHAHRAMANI, Z. Outlier robust gaussian process classification. In: *Proceedings of the 2008 Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition (SSPR)'08*. [S.l.: s.n.], 2008. p. 896–905.

KOHONEN, T. Correlation matrix memory. *IEEE Transactions on Computers*, C-21, n. 4, p. 353–359, 1972.

KOHONEN, T.; OJA, E. Fast adaptive formation of orthogonalizing filters and associative memory in recurrent networks of neuron-like elements. *Biological Cybernetics*, v. 25, p. 85–95, 1976.

KOHONEN, T.; RUOHONEN, M. Representation of associated data by matrix operators. *IEEE Transactions on Computers*, v. 22, n. 7, p. 701–702, 1973.

LEE, C.-C. et al. Noisy time series prediction using *m*-estimator based robust radial basis function neural networks with growing and pruning techniques. *Expert Systems and Applications*, v. 36, n. 3, p. 4717–4724, 2009.

LEE, C.-C. et al. Robust radial basis function neural networks. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, v. 29, n. 6, p. 674–685, 1999.

LEE, D. D.; SEUNG, H. S. Learning the parts of objects by non-negative matrix factorization. *Nature*, v. 401, p. 788–791, 1999.

LEE, D. D.; SEUNG, H. S. Algorithms for non-negative matrix factorization. In: PRESS, M. (Ed.). *Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference*. [S.l.: s.n.], 2000. p. 556–562.

LEGENDRE, A. M. *Nouvelles Méthodes pour la Détermination des Orbites des Comètes*. [S.l.]: Courcier, Paris, 1805.

LI, C. A model of neuronal intrinsic plasticity. *IEEE Transactions on Autonomous Mental Development*, v. 3, n. 4, p. 277–284, 2011.

LI, D.; HAN, M.; WANG, J. Chaotic time series prediction based on a novel robust echo state network. *IEEE Transactions on Neural Networks and Learning Systems*, v. 23, n. 5, p. 787–799, 2012.

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

MARONNA, R. A.; MARTIN, D. R.; YOHAI, V. J. *Robust Statistics: Theory and Methods*. 1a.. ed. [S.l.]: John Wiley & Sons, 2006.

MESQUITA, M. E. R. V. Introdução às memórias associativas lineares, morfológicas e fuzzy. In: *Anais do 2o. Colóquio de Matemática da Região Sul (ColMatSul'2012*. [S.l.: s.n.], 2012.

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

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

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.

NAKANO, K. Associatron: A model of associative memory. *IEEE Transactions on on Systems, Man, Cybernetics*, SMC-2, n. 3, p. 380–388, 1972.

NETO, A. R. R.; BARRETO, G. A. On the application of ensembles of classifiers to the diagnosis of pathologies of the vertebral column: A comparative analysis. *IEEE Latin America Transactions*, v. 7, n. 4, p. 487–496, 2009.

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

OES, R. S. B.; LIMA, V. M. C. Comparação de estimadores de regressão. In: *Anais do XIX Simpósio Nacional de Probabilidade e Estatística (SINAPE'2010)*. [S.l.: s.n.], 2010. p. 1–6.

PEDERSEN, M. E. H.; CHIPPERFIELD, A. J. Simplifying particle swarm optimization. *Applied Soft Computing*, v. 10, n. 2, p. 618–628, 2010.

POGGIO, T.; GIROSI, F. Networks for approximation and learning. *Proceedings of the IEEE*, v. 78, n. 9, p. 1481–1497, 1990.

PRASAD, B. et al. A study on associative neural memories. *International Journal of Advanced Computer Science and Applications*, v. 1, n. 6, p. 124–133, 2010.

RAMALHO, G. L. B.; MEDEIROS, F. N. S. Using boosting to improve oil spill detection in sar images. In: *Proceedings of the 18th International Conference on Pattern Recognition (ICPR'2006)*. [S.l.: s.n.], 2006. v. 2, p. 1066–1069.

RAO, C. R.; TOUTENBURG, H. *Linear Models: Least Squares and Alternatives*. 2nd. ed. [S.l.]: Springer, 1999.

RIPLEY, B. D. *Robust Statistics*. [S.l.], 2004.

RITTER, G.; GALLEGOS, M. T. Outliers in statistical pattern recognition and an application to automatic chromosome classification. *Pattern Recognition Letters*, v. 18, n. 6, p. 525–539, 1997.

RONCHETTI, E. The historical development of robust statistics. In: *Proceedings of the 7th International Conference on Teaching Statistics (ICOTS-7)*. [S.l.: s.n.], 2006. p. 1–4.

ROUSSEEUW, P. J.; LEROY, A. M. *Robust Regression and Outlier Detection*. 1a.. ed. [S.l.]: John Wiley & Sons, 1987.

SCHONHOFF, T.; GIORDANO, A. *Detection and Estimation Theory*. 1a.. ed. [S.l.]: Prentice Hall, 2006.

SINGH, S.; MARKOU, M. An approach to novelty detection applied to the classification of image regions. *IEEE Transactions on Knowledge and Data Engineering*, v. 16, n. 4, p. 396–407, 2004.

SMITH, M. R.; MARTINEZ, T. Improving classification accuracy by identifying and removing instances that should be misclassified. In: *Proceedings of the 2011 International Joint Conference on Neural Networks (IJCNN'2011)*. [S.l.: s.n.], 2011. p. 2690–2697.

STANIMIROVA, I.; WALCZAK, B. Classification of data with missing elements and outliers. *Talanta*, v. 76, n. 3, p. 602–609, 2008.

STEVENS, J. P. Outliers and influential data points in regression analysis. *Psychological Bulletin*, v. 95, n. 2, p. 334–344, 1984.

STILES, G.; DENQ, D.-L. A quantitative comparison of the performance of three discrete distributed associative memory models. *IEEE Transactions on Computers*, v. 36, n. 3, p. 257–263, 1987.

STILES, G. S.; DENQ, D. On the effect of noise on the Moore-Penrose generalized inverse associative memory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 7, n. 3, p. 358–360, 1985.

TONG, T. T. *Diagnostics for Outliers and Influential Points*. [S.l.], 2010.

TRIESCH, J. A gradient rule for the plasticity of a neuron's intrinsic excitability. In: *Proc. of the Int. Conf. on Artificial Neural Networks (ICANN)*. [S.l.: s.n.], 2005.

TUKEY, J. W. The future of data analysis. *Annals of Mathematical Statistics*, v. 33, n. 1, p. 1–67, 1964.

VASCONCELOS, G. C.; FAIRHURST, M. C.; BISSET, D. L. Investigating feedforward neural networks with respect to the rejection of spurious patterns. *Pattern Recognition Letters*, v. 16, p. 207–212, 1995.

VIEIRA, A. F. C.; DAL BELLO, L. H. A. Experimentos com mistura para otimizaççção de processos: Uma aplicação com respostas nρrocessos: Uma aplicaÇÃo com respostas nÃo normais. *Pesquisa Operacional*, v. 26, n. 3, p. 605–623, 2006.

WEBB, A. *Statistical Pattern Recognition*. 2. ed. [S.l.]: John Wiley & Sons, LTD, 2002.

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