Pós-Graduação em Ciência da Computação

EDSON LEITE ARAÚJO

UM CLASSIFICADOR BASEADO EM PERTURBAÇÕES

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE
2017

Edson Leite Araújo

Um Classificador Baseado em Perturbações

Este trabalho foi apresentado à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Doutor em Ciência da Computação.

ORIENTADOR: **Prof. Dr. George D. C. Cavalcanti**
CO-ORIENTADOR: **Prof. Dr. Tsang Ing Ren**

RECIFE
2017

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

A662c Araújo, Edson Leite
Um classificador baseado em perturbações / Edson Leite Araújo. – 2017.
103 f.:il, fig., tab.

Orientador: George Darmiton da Cunha Cavalcanti.
Tese (Doutorado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2017.
Inclui referências.

1. Inteligência artificial. 2. Reconhecimento de padrão. I. Cavalcanti, George Darmiton da Cunha (orientador). II. Título.

006.3 CDD (23. ed.) UFPE- MEI 2017-162

Edson Leite Araújo

Um Classificador Baseado em Perturbações

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutora em Ciência da Computação

Aprovado em: 10/04/2017.

_____

Orientador: Prof. Dr. George Darmiton da Cunha Cavalcanti

BANCA EXAMINADORA

_____

Profa. Dra. Renata Maria Cardoso Rodrigues de Souza
Centro de Informática / UFPE

_____

Prof. Dr. Germano Crispim Vasconcelos
Centro de Informática / UFPE

_____

Prof. Dr. Leandro Maciel Almeida
Centro de Informática / UFPE

_____

Profa. Dra. Anne Magaly de Paula Canuto
Departamento de Informática e Matemática Aplicada / UFRN

_____

Profa. Dra. Roberta Andrade de Araújo Fagundes
Escola Politécnica de Pernambuco / UPE

Dedico esta tese à minha mãe, D. Neci (Dona Moça). Sua fé inabalável neste filho me fizeram ser o que sou. Esta conquista é sua, mãe!

Agradecimentos

Em fim, após longos anos de muito esforço e dedicação, é chegada a hora de agradecer àqueles que me ajudaram nesta jornada. Começo meus agradecimentos pelo meu orientador, o professor George Darmiton. Sua serenidade, tranquilidade, apoio e acima de tudo, a sua crença na minha capacidade de trabalho em uma área que não é exatamente a minha, permitiu-me chegar até aqui. Uma ideia que aparentemente não teria importância nenhuma, sob a luz de sua ótica e maturidade, tornou-se esta tese. O professor Tsang Ing Ren, com seu olhar clínico e suas críticas sempre úteis e construtivas, foram essenciais à conclusão deste trabalho. Muito obrigado.

Agradeço aos meus familiares, em especial à minha esposa Sônia Cristina e meus filhos Thalles e Isaac, que toleraram meus maus momentos, sempre me devotando este imenso amor que é a fortaleza sobre a qual me apoio em todos os momentos. À minha mãe, meus irmãos, por sempre estarem ao meu lado não apenas nesta, mas em todas as minhas lutas, acreditando em mim mais do que eu mesmo.

Agradeço aos meus colegas do colegiado de engenharia civil, que concordaram com meu afastamento, o que possibilitou a escrita deste trabalho. Em especial, agradeço ao meu amigo Prof. Sérgio Motta, por seu ombro amigo e sua paciência em ouvir minhas inseguranças tantas vezes e à amiga Silvia Coelho que, com sua serenidade me ajudou a encontrar equilíbrio emocional e psicológico em momentos difíceis.

Agradeço ao amigo e hoje compadre, Prof. Lino Marcos. Nossas longas conversas, aliviaram por tantas vezes a pressão de tantas cobranças. Sua calma e seu conselho foram de grande valia.

Ao final deste doutorado, tive a honra de fazer uma nova amizade com o irmão de orientação, o amigo Renê Gadelha, sempre disposto a ajudar em tudo que foi preciso. Agradeço muito, por sua amizade e por sua generosidade.

Não poderia esquecer de você, amiga Profª Michelle Christini. Não fosse por suas instruções, minha inscrição neste DINTER não teria acontecido. Falando em DINTER, agradeço também ao amigo Prof. Max Rolemberg que tornou real este programa de pós-graduação interinstitucional, entre a instituição em que trabalho, a UNIVASF e a UFPE que proporcionou o curso. Agradeço a ambas instituições pela oportunidade.

Agradeço à FACEPE pelo apoio financeiro durante os 12 meses que permaneci na cidade de Recife/PE.

Por fim, agradeço Àquele que não me deixou sucumbir. Após tantas idas e voltas, dificuldades, sobressaltos e cobranças, a força para manter-se firme e convicto na busca deste sonho, sem nunca desistir, veio de Ti meu Pai. Agradeço a Ti, Jesus.

Let the future tell the truth and evaluate each one according to their work and accomplishments. The present is theirs. The future, for which I really worked, is mine.

—NIKOLA TESLA

Resumo

Muitos algoritmos de reconhecimento de padrões são probabilísticos em sua construção e como tal, usam a inferência estatística para determinar o melhor rótulo para uma dada instância a ser classificada. A inferência estatística baseia-se em geral, na *teoria de Bayes* que por sua vez, utiliza fortemente dos vetores médios, $\boldsymbol{\mu}_i$, e matrizes de covariância, Σ_i, de classes existentes nos dados de treinamento. Estes parâmetros são desconhecidos e estimativas são realizadas seguindo vários algoritmos. Entretanto, as estimativas feitas exclusivamente a partir dos dados de treinamento são ainda as mais utilizadas. Por se tratarem de estimativas, os parâmetros $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ sofrem perturbações quando se insere um novo vetor na classe à qual pertencem. Avaliando as perturbações ocorridas em todas as classes simulando uma possível inserção da instância a ser classificada nas mesmas, definimos neste trabalho uma nova regra de decisão a qual atribui a instância de teste à classe em que ocorrer a menor perturbação nos parâmetros $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ ou numa combinação de ambos. Nesta área, várias abordagens são possíveis, entre elas merecem destaque as árvores de decisão, as redes neurais, o aprendizado baseado em instâncias e a máquina de vetores de suporte(*SVM*). Entretanto, até o momento da escrita deste texto, não foi encontrado na literatura, abordagens que utilizem as perturbações de parâmetros para a classificação de padrões. Em testes realizados inicialmente em dados sintéticos e posteriormente em 21 bancos de dados reais disponíveis no *UCI Repository Learning*, verificou-se que o classificador baseado em perturbações, o qual foi denominado *PerC* (**Per***turbation* **C***lassifier*), apresentou performance significativamente superior às versões do *SVM* com *kernels* polinomiais de graus 2 e 3, e praticamente equivalente aos *k-Nearest Neighboor* com k=3 e k=5, *Naïve Bayes*, *SVM* com *kernel* gaussiano, *CART* e as redes neurais *MLP*, tendo o *PerC* o maior ranking segundo o teste estatístico de Friedman. Os resultados demonstraram que a abordagem baseada em perturbações são, portanto, úteis para a classificação de padrões.

Palavras-chave: Reconhecimento de padrões. Perturbações. Matriz de Covariância e Vetor Médio. Classificador de Bayes.

Abstract

Many pattern recognition algorithms are probabilistic in their structure and as such, they use statistical inference to determine the best label for a given instance to be classified. The statistical inference is based generally on Bayes theory which strongly uses the average vectors, $\boldsymbol{\mu}_i$, and covariance matrices, Σ_i, of existing classes in the training data. These parameters are unknown and estimates are made by following various algorithms. However, the estimates made exclusively from the training data are still the most used. Because they are estimates, the parameters $\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ and $\hat{\Sigma}_i$ are perturbed when a new vector is inserted into the class which they belong to. Evaluating the perturbations that occurred in all classes simulating a possible inclusion of the instance to be classified in the same one, we defined in this work a new decision rule which assigns the test instance to the class in which occurs the slightest perturbation in $\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ and $\hat{\Sigma}_i$ parameters or the combination of both. In this area, several approaches are possible, it's worth mentioning the decision trees, neural networks, instance-based learning and the *support vector machine* (*SVM*). However, until the moment of the writing of this text, was not found in the literature, approaches that use parameters perturbations to pattern's classification. In tests performed initially on synthetic data and later on 21 real databases available in the *UCI Repository Learning*, was verified that perturbation-based classifier, which was denominated *PerC* (Perturbation Classifier), presented performance significantly superior to the versions of the *SVM* with polinomial kernels of degrees 2 and 3 and roughly equivalent to *k-Nearest Neighboor* with k = 3 and k = 5, *Naïve Bayes*, *SVM* with Gaussian kernel, *CART* and *MLP* neural networks, having the *PerC* the highest ranking according to the Friedman statistical test. The results demonstrated that the perturbation-based approach is therefore useful to pattern classification.

Keywords: Patterns Recognition. Perturbations. Covariance Matrice and Mean Vector. Bayes Classifier.

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos

Lista de Símbolos

C	Número de classes
i ou j	Índices da classe
N	Número de amostras (vetores) do banco de treinamento
N_i	Número de amostras (vetores) da classe i
$\boldsymbol{x}$	Vetor
$\boldsymbol{x}_{i,j}$	j-ésima amostra (vetor) da classe i
$p(\pi_i)$	Probabilidade *a priori* de um vetor estar na classe i
$\boldsymbol{p}_i(\boldsymbol{x})$	Probabilidade *a posteriori* de uma vetor $\boldsymbol{x}$ estar na classe i
$d_i(\boldsymbol{x})$	Estimativa de $\boldsymbol{p}_i(\boldsymbol{x})$
$\mathbb{R}^n$	Espaço vetorial n-dimensional
$\boldsymbol{\mu}_i$	Vetor médio da classe i
$\hat{\boldsymbol{\mu}}_i$	Estimativa do vetor médio da classe i
$\hat{\boldsymbol{\mu}}'_i$	Estimativa do vetor médio da classe i alterada
$\Delta\hat{\boldsymbol{\mu}}_i$	Variação em $\hat{\boldsymbol{\mu}}_i$
Σ_i	Matriz de covariância da classe i
$\hat{\Sigma}_i$	Estimativa da matriz de covariância da classe i
$\hat{\Sigma}'_i$	Estimativa da matriz de covariância da classe i alterada
$\Delta\hat{\Sigma}_i$	Variação em $\hat{\Sigma}_i$

Sumário

1 INTRODUÇÃO

For every complex problem there is an answer that is clear, simple, and wrong.

—H. L. MENCKEN

1.1 Motivação

O processo de reconhecimento de padrões consiste em duas tarefas fundamentais: descrição e classificação. Dado um objeto a ser analisado, um sistema de reconhecimento de padrões inicialmente produz uma descrição dele, o *padrão*, e então classifica-o de acordo com esta descrição, o *reconhecimento*. O problema essencial do reconhecimento de padrões é identificar um objeto como pertencendo a um grupo em particular, assumindo que objetos de um mesmo grupo compartilham atributos comuns mais do que com quaisquer objetos de outros grupos (DUDA; HART; STORK, 2012; FUKUNAGA, 1972). Entre as várias abordagens utilizadas destacam-se o *modelo estatístico*, o *aprendizado baseado em lógica*, as *redes neurais* e a *máquina de vetores de suporte* (em inglês, *Support Vector Machine* (*SVM*)).

As aplicações do reconhecimento de padrões, incluem desde bioinformática, classificação de documentos, análise de imagens, mineração de dados, automação industrial, reconhecimento biométrico, sensoriamento remoto, análise de texto manuscrito, o diagnóstico médico e reconhecimento de fala (WEBB; COPSEY, 2011).

Os algoritmos de reconhecimento de padrões podem ser agrupados de acordo com o paradigma de aprendizado utilizado. No aprendizado *supervisionado*, assume-se que é fornecido um conjunto de dados de treinamento cujas instâncias foram adequadamente rotuladas. Por sua vez, o aprendizado *não supervisionado*, também conhecido como *clustering*, admite que os dados de treinamento não tenham sido rotulados e tenta encontrar padrões entre os dados e definir a partir destes, categorias e o agrupamento dos dados. A combinação destas duas técnicas, conhecida como aprendizado *semi-supervisionado*, tem sido explorada e usa, em geral, uma pequena quantidade de dados rotulados e uma grande quantidade de dados não rotulados

(CHAPELLE; SCHÖLKOPF; ZIEN, 2006).

O aprendizado por reforço (*Reinforcement Learning*) foi inspirado no comportamento psicológico humano e lida com ações a serem tomadas de modo a maximizar alguma noção de recompensa acumulativa. Difere das outras abordagens para o aprendizado de máquinas (segundo Bishop (2006), o reconhecimento de padrões e o aprendizado de máquinas podem ser vistos como faces de um mesmo campo de pesquisa), no sentido em que, instâncias corretamente classificadas nunca são fornecidas e ações incorretas não são explicitamente corrigidas. O aprendizado por reforço é definido não pelo método de aprendizagem utilizado, mas por caracterizar o problema a ser aprendido. Deste modo, qualquer método que seja adequado a resolver este problema é considerado um reforço de aprendizagem. (SUTTON; BARTO, 1998).

Os dados de treinamento podem ser disponibilizados de diferentes maneiras: uma instância por vez, algumas instâncias ou todas juntas. De acordo com esta disponibilidade, os algoritmos de aprendizado podem ainda ser subagrupados em *online* e *incremental*. *Incremental Learning* é um paradigma de aprendizado de máquina, no qual o processo de aprendizagem ocorre sempre que novas instâncias surgem e incorporando-as aos dados de treinamento atualiza-se o aprendizado prévio a estas novas instâncias. Desta forma, o *incremental learning* não assume que os dados disponíveis para o treinamento sejam suficientes para o aprendizado (ADE; DESHMUKH, 2013). De modo semelhante, o *online learning* é utilizado quando as instâncias dos dados de treinamento tornam-se disponíveis uma de cada vez e deseja-se que o algoritmo de aprendizagem produza uma sequência de hipotéses $(f_1, \ldots, f_n)$ em que f_1 é uma hipótese inicial arbitrária e f_i é a hipótese escolhida após o método receber a $(i-1)$-ésima instância. Estas hipóteses são escolhidas de modo a minimizar o custo de realizar uma previsão incorreta. (KIVINEN; SMOLA; WILLIAMSON, 2004).

Há ainda o que se conhece como *aprendizado por transferência* (*Transfer learning*). Esta técnica tem por objetivo melhorar o aprendizado de máquina tradicional, transferindo o conhecimento adquirido em uma ou mais tarefas prévias para o aprendizado em uma nova tarefa alvo relacionada (MITCHELL, 1997).

Entre as várias abordagens tradicionalmente formuladas para o reconhecimento de padrões, o modelo estatístico tem sido um dos mais intensivamente estudado e utilizado na prática (WEBB; COPSEY, 2011). Outras abordagens, como as *redes neurais* e o *SVM* por exemplo, acabam por também utilizar o modelo estatístico em suas regras de decisões.

Dado um conjunto de padrões de treinamento, o objetivo dos métodos baseados na abordagem estatística é estabelecer fronteiras de decisão no espaço de características que separem os padrões pertencentes a diferentes classes. As fronteiras de decisão são construídas a partir de distribuições de probabilidades dos padrões pertencentes à cada classe. Tais distribuições devem ser especificadas ou aprendidas (DEVROYE; GYÖRFI; LUGOSI, 1996; DUDA; HART et al., 1973).

1.2 O Problema

A normalidade dos dados de treinamento é em geral, uma suposição necessária para as técnicas que utilizam o modelo estatístico. Dados com esta distribuição são caracterizados por seu vetor médio, $\boldsymbol{\mu}$, e sua matriz de covariância, Σ.

A **teoria de Bayes** estabelece os princípios da inferência bayesiana (ou aprendizado bayesiano) que servem como base para o *classificador de Bayes*. Desta forma, os métodos desenvolvidos a partir da inferência bayesiana ou derivados do classificador de Bayes, possuem forte dependência dos vetores médios, $\boldsymbol{\mu_i}$, e das matrizes de covariância, Σ_i, de cada uma das classes (grupos) existentes nos dados de treinamento (FUKUNAGA, 1972; MITCHELL, 1997; JAIN; DUIN; MAO, 2000; DUDA; HART; STORK, 2012). Em geral, estes parâmetros são desconhecidos e em seus lugares, estimativas são utilizadas.

Embora existam várias maneiras de realizar aproximações de $\boldsymbol{\mu_i}$ e Σ_i (PERRON, 1992; HOFFBECK; LANDGREBE, 1996; TADJUDIN; LANDGREBE, 1999; KUO; LANDGREBE, 2002; LEDOIT; WOLF, 2004; WU; XIAO, 2012), as estimativas de máxima verossimilhança, baseadas exclusivamente nas amostras disponíveis para o treinamento, denotadas por $\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ e $\hat{\Sigma}_i$ (JOHNSON; WICHERN, 2007), são as mais utilizadas.

Por se tratar de estimativas, a inserção de uma nova instância, o padrão de teste ou de consulta, na classe ω_i, fará com que seus parâmetros $\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ e $\hat{\Sigma}_i$, sejam alterados. Estas alterações (ou perturbações) são denotadas por $\Delta\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ e $\Delta\hat{\Sigma}_i$, respectivamente.

Simulando a inserção do padrão de consulta na classe ω_i é possível avaliar as perturbações $\Delta\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ e $\Delta\hat{\Sigma}_i$ para cada uma das classes presentes no banco de treinamento e distinguir qual classe sofrerá maior ou menor perturbação.

Com base nas perturbações obtidas, algumas questões podem ser colocadas:

- Seria correto afirmar que o padrão de consulta deve ser atribuido à classe que sofreu as menores perturbações em seu vetor médio e sua matriz de covariância?
- Ou seja, seria possível criar um classificador baseado nas perturbações $\Delta\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ e $\Delta\hat{\Sigma}_i$?
- Qual destas perturbações, $\Delta\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ ou $\Delta\hat{\Sigma}_i$, seria mais útil para a classificação? Ou ambas teriam a mesma importância?
- Seria possível usar o poder de discriminação de ambas as perturbações em um único classificador?

1.3 Objetivo

Neste trabalho é proposta uma nova abordagem para a classificação de padrões baseada nas perturbações dos parâmetros $\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ e $\hat{\Sigma}_i$, causadas pela inserção de um padrão de consulta nas classes ω_i, presentes nos dados de treinamento.

Tomando por base as questões colocadas na seção anterior, propõe-se como objetivo geral deste trabalho:

- Apresentar uma nova abordagem para a classificação de padrões, baseada nas perturbações de $\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ e $\hat{\Sigma}_i$.

Como objetivos específicos, propõe-se:

- Apontar meios para o cálculo eficiente de $\Delta\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ e $\Delta\hat{\Sigma}_i$.
- Desenvolver dois classificadores, um baseado nas perturbações do vetor médio e o outro baseado nas perturbações da matriz de covariância e avaliá-los a partir de dados sintéticos comparando-os com classificadores, bastante conhecidos nesta área: o *k-Nearest Neighboor* (*kNN*) (COVER; HART, 1967), em suas versões para *k=1,3* e *5*, o *Naïve Bayes* (DOMINGOS; PAZZANI, 1997; HAND; YU, 2001), o *SVM*, em suas versões para *kernels polinomiais de graus 1, 2, 3* e o *kernel gaussiano*, as *Árvores de Classificação e Regressão* (do inglês, *Classification And Regression Trees* (*CART*)) (BREIMAN et al., 1984) e as *Redes Neurais Multicamadas* (em inglês, *Multilayers Perceptrons* (*MLP*)) (BISHOP, 1994).
- Desenvolver um classificador que reúna o poder de discriminação de ambas as perturbações, avaliá-lo a partir de dados reais obtidos do *UCI Repository Learning* (BACHE; LICHMAN, 2013), e comparar sua performance com a dos classificadores *kNN*, em suas versões para *k=1, 3* e *5*, o *Naïve Bayes*, o *SVM*, em suas versões para *kernels polinomiais de graus 1, 2, 3* e o *kernel gaussiano*, o *CART* e o *MLP*.

1.4 Visão geral da proposta

No Capítulo 2, será realizada uma revisão das principais técnicas utilizadas em algoritmos de reconhecimento de padrões. Até o momento da escrita deste texto, não foi encontrado na literatura, trabalhos que utilizem as perturbações em $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ ou quaisquer outras espécies de perturbações, para o desenvolvimento de algoritmos de classificação. Espera-se portanto, expor nesse capítulo, os principais aspectos de cada abordagem que possam demonstrar as semelhanças ou diferenças com o algoritmo proposto.

A *teoria de Bayes*, sobre a qual está fundamentado este trabalho, encontra-se de forma resumida, com seus principais conceitos e resultados utilizados, exposta no Capítulo 3.

No Capítulo 4 são apresentados os classificadores com base nas perturbações de $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$. A validade destes classificadores é verificada a partir de experimentos realizados inicialmente sobre dados sintéticos. No Capítulo 5, os experimentos são ampliados para dados sintéticos gaussianos com mais classes, não gaussianos e finalmente para de dados reais retirados do *UCI Repository Learning*. Os resultados são expostos e é realizada uma análise da performance do

classificador proposto em relação aos classificadores *kNN*, em suas versões para *k=1, 3* e *5*, o *Naïve Bayes*, o *SVM*, em suas versões para *kernels polinomiais de graus 1, 2, 3* e o *kernel gaussiano*, o *CART* e o *MLP*. No Capítulo 6 são apresentadas as conclusões.

2

RECONHECIMENTO DE PADRÕES

Learn from yesterday, live for today, hope for tomorrow. The important thing is not to stop questioning.

—A. EINSTEIN

A riqueza da literatura nas décadas de 1960, 1970 e 1980 estabeleceu as bases para o reconhecimento de padrões moderno (DEVIJVER; KITTLER, 1982; DUDA; HART et al., 1973; FU, 1982; FUKUNAGA, 1972; MINSKY; PAPERT, 1969; NILSSON, 1962; PATRICK, 1972; ROSENBLATT, 1962; SEBESTYEN, 1962; TOU; GONZALEZ, 1974). Diante de desafios gerados a partir de problemas da vida real, sofisticadas e elegantes teorias coexistem com idéias *ad hoc*, intuição e até mesmo adivinhação (KUNCHEVA, 2004).

O reconhecimento de padrões é uma área de pesquisa multidisciplinar com influência e conceitos em matemática, computação, inteligência artificial, estatística, biologia, psicologia, economia e teoria de controle entre outras. É um ramo da *aprendizagem de máquina* que foca na detecção de padrões e regularidades existentes em dados obtidos de problemas diversos. Em alguns casos, o reconhecimento de padrões e o aprendizado de máquina são considerados quase sinônimos (BISHOP, 2006).

Em termos gerais, o reconhecimento de padrões lida com programas computacionais que, através da experiência, ou seja, por meio dos exemplos fornecidos nos dados, realiza previsões. Em outras palavras, com programas que são capazes de aprender.

Abrange vários problemas de processamento de informações de grande siginificado prático, desde o reconhecimento da fala, classificação de caracteres manuscritos, detecção de falhas em máquinas e em diagnósticos médicos, classificação de documentos, previsão financeira, organização e recuperação de dados multimídia, biometria (reconhecimento facial e de digitais). Estes problemas, são frequentemente, resolvidos por seres humanos de um modo aparentemente simples. Entretanto, computacionalmente, tal solução tem em muitos casos, mostrado ser imensamente difícil.

A área de aprendizado de máquina tem contribuído com numerosos resultados teóricos,

paradigmas de aprendizado, algoritmos e aplicações. Entre as diferentes aplicações desenvolvidas estão, por exemplo, veículos autônomos (POMERLEAU, 1989; TEICHMAN; THRUN, 2012) , classificação de requerentes de seguro de vida (TAN; ZHANG, 2005), detecção de vazamentos em dutos (CHEN et al., 2004) e etc.

2.1 Conceitos Básicos

Geralmente, existem três aspectos que afetam o desenvolvimento de programas computacionais que lidam com o aprendizado de máquina: como a solução deve ser representada, a entrada e o conhecimento prévio disponível a respeito dos dados (LAVESSON, 2006).

2.1.1 Representação

Dependendo do tipo de saída considerada para um problema de aprendizado, contínua ou discreta, a solução é representada por uma uma *função de regressão* ou um *classificador*. Uma função de regressão é definida por um número de atributos (entradas) e coeficientes, e retorna uma saída contínua. Durante a fase de aprendizado, os coeficientes são ajustados para produzir a saída correta dada uma entrada em particular. Um *classificador* por sua vez, é uma função que retorna saídas discretas, ou seja, atribui um valor discreto, frequentemente chamado de *rótulo* ou *classe*, para uma determinada entrada (LAVESSON, 2006).

2.1.2 Dados de Entrada

Usualmente, a entrada para um programa de aprendizado consiste em uma amostra de instâncias de dados. Neste contexto, uma instância é formalmente descrita por um vetor de características, que constitue uma representação de todas as características conhecidas daquela instância. Estas características podem ser entendidas como eixos num espaço n-dimensional, conhecido como *espaço de características*.

Selecionando-se as características (algumas vezes também denominadas de *atributos*) adequadas, instâncias podem descrever quaisquer coisas, desde veículos até caracteres escritos à mão.

2.1.3 Conhecimento Prévio

É possível distinguir entre quatro tipos básicos de conhecimento prévio para o reconhecimento de padrões:

- Supervisionado,
- Não-supervisionado,

- Semi-supervisionado,
- Por reforço.

O *aprendizado supervisionado*, também conhecido como *análise de discriminantes* (JAIN; DUIN; MAO, 2000), possui acesso a um supervisor ou professor, que lhe fornece respostas corretas (saídas) para um número limitado de questões (entradas). A este número limitado de questões corretas dá-se o nome de *conjunto de treinamento* e às respostas corretas, chamamos de *rótulos* ou *classes*.

Algumas vezes, ao analisarmos problemas reais, pode ser relevante assumir que este supervisor nem sempre possa fornecer respostas corretas, ou seja, admitir a possível existência de ruído. O objetivo é aprender como generalizar a partir do que tem sido aprendido e dar respostas corretas para questões previamente desconhecidas.

O *aprendizado não-supervisionado*, também chamado de *agrupamento (clustering)* (JAIN; DUIN; MAO, 2000; THEODORIDIS; KOUTROUMBAS, 2008) não possui acesso prévio a qualquer resposta sendo, portanto, seu objetivo encontrar padrões existentes nos dados de entrada, sem a ajuda de um supervisor. Desta forma, o conjunto de treinamento de um aprendiz não supervisionado não possui quaisquer rótulos conhecidos, tendo que criar tais rótulos a partir dos dados de treinamento. Um aprendiz puramente não-supervisionado pode portanto, não aprender o que fazer numa determinada situação pois a ele nunca foi dado quaisquer informações sobre quando ou não uma ação é correta (RUSSELL; NORVIG, 2003).

O aprendizado *semi-supervisionado* é um conjunto de tarefas de aprendizado supervisionado e técnicas que também fazem uso de dados não rotulados para o treinamento - tipicamente, uma pequena quantidade de dados rotulados e uma grande quantidade de dados não rotulados. Este tipo de aprendizado está entre o aprendizado supervisionado e o não-supervisionado. Pesquisadores tem descoberto que dados não rotulados, quando utilizados em conjunto com uma pequena quantidade de dados rotulados, podem produzir uma consideravel melhoria na precisão do aprendizado. A aquisição de dados rotulados para um problema de aprendizado frequentemente requer o envolvimento de um agente humano ou um experimento físico. O custo associado com este processo de rotulação pode ser expansivo e nestes casos, o aprendizado semi-supervisionado pode ser de grande valor prático. Seu valor teórico também tem sido de interesse no aprendizado de máquina como modelo para aprendizado humano (CHAPELLE; SCHÖLKOPF; ZIEN, 2006).

O aprendizado por *reinforcement* obtém o conhecimento prévio por meio de recompensas ocasionais pelas ações ou decisões que recomendam. Este tipo de aprendizado é frequentemente usado quando o mapeamento entre entradas e saídas envolvem vários passos. É um modelo de aprendizado interativo, diferindo portanto do aprendizado supervisionado no sentido em que este aprende através de exemplos fornecidos por um supervisor externo enquanto, o aprendizado por *reinforcement* aprende através de interações, geradas dentro do próprio problema. Por exemplo, aprender a pousar uma aeronave envolve centenas, ou talvez milhares de passos. Em vez de

receber um positivo ou negativo (sucesso ou falha) após ter tentado pousar uma aeronave talvez seja mais sábio usar um sistema de recompensa que dá pequenas recompensas para cada etapa ou cadeia de etapas subsequentes, executadas corretamente (SUTTON; BARTO, 1998).

O *aprendizado incremental* é um paradigma de aprendizagem de máquina em que o processo de aprendizagem ocorre incrementalmente, autoajustando-se sempre que novas instâncias são adicionadas aos dados de treinamento. A principal diferença em relação ao aprendizado de máquina tradicional é que a aprendizagem incremental não assume que os dados disponíveis para treinamento sejam suficientes para o aprendizado, sendo este atualizado à medida que surgem novos exemplos (GENG; SMITH-MILES, 2009).

Ao lidar-se com processos incrementais, vários termos diferentes e com significados semelhantes, tem sido utilizados para referir-se ao *aprendizado incremental* (Incremental Learning). Alguns pesquisadores (BLUM, 1998; KIVINEN; SMOLA; WILLIAMSON, 2004; CHENG et al., 2007), nomearam algoritmos que aprendem de modo incremental como algoritmos de *aprendizado online* (Online Learning). Os algoritmos que tentam resolver o *problema de drift* são também conhecidos por *aprendizado adaptativo* (Adaptive Learning) (HUO; LEE, 1997), e ainda há outros chamados de *aprendizado por transferência* (Transfer Learning) (PAN; KWOK; YANG, 2008).

2.2 Principais Abordagens

Entre as abordagens utilizadas no reconhecimento de padrões destacam-se:

- árvores de decisão;
- aprendizado por regras;
- redes neurais;
- aprendizado baseado em instâncias;
- máquina de suporte a vetores;
- aprendizado estatístico;

Estes modelos não são necessariamente independentes e algumas vezes um mesmo algoritmo existe em mais de uma abordagem com diferentes interpretações. Várias pesquisas têm sido realizadas buscando desenvolver sistemas híbridos envolvendo múltiplos modelos (RUMELHART; HINTON; WILLIAMS, 1986).

Nas próximas seções é feita uma breve descrição sobre cada uma destas abordagens e seus principais algoritmos.

2.2.1 Árvores de Decisão

O aprendizado através do uso de árvores de decisão tem sido um dos mais amplamente utilizados para a inferência indutiva. É um método para aproximar funções discretas em que a função aprendida é representada por uma árvore de decisão (MITCHELL, 1997).

As árvores de aprendizagem classificam as instâncias ordenando-as ao percorrer a árvore em sentido descendente a partir do nó raíz até alguma folha. Cada nó na árvore especifica um teste em algum atributo da instância e cada ramo abaixo deste nó corresponde a um possível valor para este atributo. Uma instância é classificada iniciando-se um percurso de decisões no nó raíz da árvore, testando o atributo especificado neste nó e então movendo-se para o ramo correspondente ao valor deste atributo. Este processo é então repetido recursivamente para cada subárvore abaixo do nó (MURTHY, 1998).

O principal problema em se construir uma árvore de decisão é determinar uma árvore binária ótima em relação aos dados de treinamento. A construção de árvores binárias ótimas é um problema NP-completo e por isto, pesquisadores tem buscado eurísticas eficientes para construções aproximadamente ótimas (KOTSIANTIS, 2007).

O algoritmo C4.5 (QUINLAN, 1993) talvez seja o mais conhecido algoritmo para a construção de árvores de decisão. Um estudo comparando árvores de decisão a outros algoritmos de aprendizado (LIM; LOH; SHIH, 2000) mostrou que o C4.5 possui uma boa relação entre taxa de acerto e velocidade de execução.

O C4.5 assume que o sistema possui memória suficiente para os dados de treinamento e em Gehrke, Ramakrishnan e Ganti (2000) é proposto o *Rainforest*, para o desenvolvimento de algoritmos rápidos e escaláveis que constroem árvores de decisão que encaixam-se perfeitamente à memória disponível.

O C4.5 usa uma abordagem dividir para conquistar equivalente a um particionamento recursivo, para as decisões de crescimento da árvore. Seleciona o teste que maximiza o ganho de informação, que é medido através da entropia definida pela *teoria da informação de Shannon* (SHANNON, 1948).

O *CART* (BREIMAN et al., 1984) utiliza um método particionamento recursivo que pode ser usado para classificação e regressão. A árvore é construída subdividindo os dados considerando todas as variáveis preditoras. A melhor variável preditora é escolhida usando uma medida de impureza conhecida com *índice de Gini*. O objetivo é produzir subconjuntos de dados que são homogêneos em relação à variável alvo.

Usualmente, as árvores de decisão são univariadas, dado que as ramificações a partir de cada nó são criadas de acordo com um único atributo. Entretanto, existem métodos (ZHENG, 1998; GAMA; BRAZDIL, 1999) que constroem árvores multivariadas com melhor precisão na classificação, através de operadores lógicos tais como a conjunção, a negação e a disjunção.

Árvores de decisão podem ser traduzidas em um conjunto de regras, criando uma regra específica para cada caminho a partir da sua raíz até as folhas. Porém, regras podem ser induzidas

diretamente dos dados de treinamento usando uma extensa variedade de algoritmos baseados em regras (FÜRNKRANZ, 1997; FÜRNKRANZ, 1999; FÜRNKRANZ, 2001; FÜRNKRANZ; FLACH, 2005).

Neste caso, o objetivo consiste em obter o menor conjunto de regras, consistentes com os dados de treinamento. Um grande número de regras, em geral significa que o algoritmo está tentando lembrar-se dos dados ao invés de descobrir as hipóteses que o governam.

Um algoritmo busca por uma regra que explique uma parte das instâncias de treinamento, separa estas instâncias e, recursivamente busca por novas regras entre as instâncias remanescentes, até que não restem mais instâncias.

É importante que um algoritmo baseado em indução por regras, gere regras de decisão que possuam alto poder de previsão e confiabilidade. Estas propriedades são freqüentemente avaliadas através de uma *função de medida de qualidade da regra* (AN; CERCONE, 2000). Ao usar conjuntos de regras desordenados, conflitos podem surgir entre tais regras, ocasionando por exemplo, que duas ou mais regras possam prever classes diferentes para uma mesma instância (LINDGREN, 2004).

O algoritmo *RIPPER* (COHEN, 1995) cria regras através de um processo conhecido como *crescimento e poda*, aplicado repetidamente. Durante o processo de *crescimento* as regras são criadas de maneira mais restritiva, de modo a atingir o maior número possível de instâncias entre os dados de treinamento. Na etapa de *poda* as regras tornam-se menos restritivas para se evitar os conflitos.

De um modo diferente, o algoritmo *PART* (FRANK; WITTEN, 1998) infere regras de decisão gerando repetidamente, árvores de decisão parciais e usando o algoritmo dividir para conquistar. A cada árvore de decisão parcial construída, uma única regra é extraída e o processo repete-se.

Embora alguns classificadores baseados em regras possam lidar com atributos numéricos, este não é o caso em geral. Alguns pesquisadores propõem que, nestes casos, os atributos sejam discretizados antes do processo de indução, para que se reduza o tempo de treinamento e aumente a precisão da classificação (AN; CERCONE, 1999).

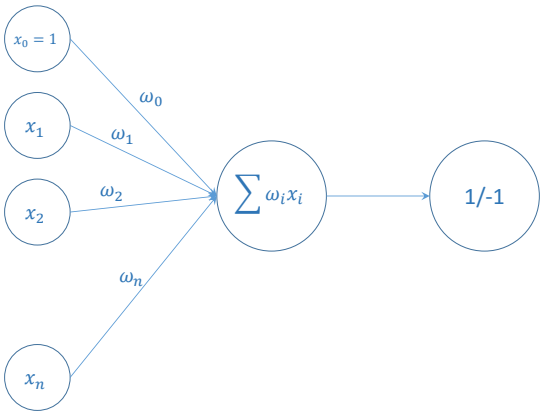
2.2.2 Redes Neurais

As *redes neurais artificiais* (GOLDBERG, 1989; BISHOP, 1995) também chamadas com freqüência de *redes neurais*, é uma abordagem que baseia-se em uma versão abstrata muito simples de como funcionam as redes neurais biológicas. O cérebro humano consiste de um bilhão de neurônios, conectados através de sinapses em uma rede muito complexa.

Uma representação abstrata simples de um neurônio, chamada *perceptron* (Figura 2.1) foi concebida originalmente por Rosenblatt por volta de 1950 (ROSENBLATT, 1957).

Um *perceptron* toma como entrada um vetor de valores reais e calcula uma combinação linear destas entradas, devolvendo 1 como saída se o resultado desta combinação é maior que

Figura 2.1: Representação de um *perceptron* com entrada $x_1, ..., x_n$ e saída 1 ou -1



algum limiar pré-estabelecido e -1 em caso contrário. O processo de aprendizagem envolve a atualização de pesos entre as conexões, usados no cálculo da combinação linear, de modo que a rede possa eficientemente realizar uma tarefa de classificação específica.

As redes neurais emergiram como uma importante ferramenta para classificação. Amplas pesquisas em classificação neural tem estabelecido que esta é uma alternativa promissora aos vários métodos de classificação convencional (ZHANG, 2000).

As vantagens das redes neurais são seus aspectos teóricos entre os quais destacam-se o auto-ajuste aos dados sem qualquer especificação explícita, as aproximações universais, no sentido em que podem aproximar quaisquer funções com precisão arbitrária (CYBENKO, 1989; HORNIK, 1991; HORNIK; STINCHCOMBE; WHITE, 1989), além disso são modelos não lineares, o que as tornam flexíveis em modelar relações complexas do mundo real. Por fim, são capazes de realizar estimativas de probabilidades *a posteriori*, o que fornece a base para estabelecer regras de classificação e análises estatísticas (RICHARD; LIPPMANN, 1991).

A família de redes neurais mais comumente utilizadas em reconhecimento de padrões é a *rede feed-forward* (JAIN; MAO; MOHIUDDIN, 1996), às quais incluem as *redes de perceptrons multicamadas* ou redes *MLP* e as *funções de base radial* (BISHOP, 1994). Outra rede popular é o *Mapeamento auto-organizável* ou *Rede Kohonen* (KOHONEN, 1998), que é utilizada principalmente em tarefas de agrupamento ou extração de características.

2.2.3 Máquina de Vetores de Suporte

Os fundamentos sobre *Máquinas de Vetores de Suporte (SVM)*, foram desenvolvidos principalmente por Vapnik (VAPNIK, 1995; VAPNIK, 1998) e os dispositivos de suporte a vetores correspondentes estão ganhando popularidade devido às suas muitas características atraentes e uma promissora performance empírica.

Podem ser entendidas como uma técnica de treinamento alternativa às *redes multicamadas de perceptrons* e às *funções de base radial*, na qual os pesos da rede são

encontrados resolvendo um problema de programação quadrático com desigualdade linear e restrições de igualdade, ao invés de resolver um problema de minimização não convexo e irrestrito, como nas técnicas tradicionais de treinamento usando redes neurais (OSUNA; FREUND; GIROSI, 1997).

A ideia básica em se estimar uma *SVM* consiste em realizar um mapeamento $\Phi : \mathbb{R}^p \to \mathcal{F}$, do espaço de características $\mathbb{R}^p$ em um espaço $\mathcal{F}$ de dimensão maior (possivelmente infinita), e então determinar uma função linear cuja forma deve ser

$$f(\boldsymbol{x}) = w \cdot \Phi(\boldsymbol{x}) + b$$

sendo $w \in \mathcal{F}$, com $(\cdot)$ denotando o produto interno em $\mathcal{F}$ e b uma constante. As coordenadas do vetor w são chamadas de pesos para cada $\Phi(\boldsymbol{x})$.

Para um problema com duas classes apenas, se os dados mapeados $\Phi(\boldsymbol{x}_i)$, $i = 1, \cdots, N$ são linearmente separáveis, existe um par (w, b) tal que

$$\begin{aligned} f(\boldsymbol{x}_i) &\geq +1, \ \forall \boldsymbol{x}_i \in Classe\,1 \\ f(\boldsymbol{x}_i) &\leq -1, \ \forall \boldsymbol{x}_i \in Classe\,2 \end{aligned} \tag{2.1}$$

Considere os pontos $\Phi(\boldsymbol{x}_i) \in \mathcal{F}$ para os quais as desigualdades em (2.1) são válidas. Estes pontos moram em dois hiperplanos

$$\begin{aligned} H_1 &: w \cdot \Phi(\boldsymbol{x}_i) + b \geq +1 \\ H_2 &: w \cdot \Phi(\boldsymbol{x}_i) + b \leq -1 \end{aligned}$$

Estes hiperplanos são paralelos e não há instâncias de treinamento projetadas entre eles. A distância entre estes hiperplanos é $\dfrac{2}{\|w\|}$, de modo que, encontrar um par de hiperplanos com distância máxima entre eles pode ser feito minimizando $\|w\|$ restrito às desigualdes (2.1) (BURGES, 1998). Escrito como um problema de otimização quadrática convexa, torna-se:

$$\begin{aligned} \underset{w,b}{\boldsymbol{min}} \quad & \frac{1}{2}\|w\|^2 \\ \boldsymbol{restrito} \quad & y_i\left[w \cdot \Phi(\boldsymbol{x}_i) + b - 1\right] \geq 0, \forall i \end{aligned} \tag{2.2}$$

sendo a primeira função conhecida como objetivo primário em problemas de otimização convexa e a segunda função corresponde às restrições. As constantes y_i são as classes existentes nos dados.

As funções $k : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ tais que

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \Phi(\boldsymbol{x}_i) \cdot \Phi(\boldsymbol{x}_j)$$

são conhecidas como *funções kernel*. Tais funções, quando existem, permitem que os produtos internos em $\mathcal{F}$ sejam calculados diretamente a partir dos dados no espaço de caracerísticas, não havendo portanto, a necessidade de se realizar o mapeamento descrito anteriormente (SCHÖLKOPF; BURGES; SMOLA, 1999).

Uma vez que um hiperplano tenha sido criado, a função kernel é usada para mapear novas instâncias do espaço de características projetado, para posterior classificação.

A escolha de uma função kernel apropriada é importante pois, ela define o espaço de atributos transformado no qual o conjunto de novas instâncias serão classificadas. Genton (GENTON, 2002) descreve várias classes de kernels, entretanto não endereça o problema de determinar qual escolha é mais adequada para um determinado problema.

Por fim, o problema de otimização necessário ao treinamento através de *SVM*s sempre atinge um mínimo global, algo que pode não ocorrer em outros algoritmos de busca tais como as redes neurais. Por outro lado, as *SVM*s são classificadores binários e no caso de problemas multiclasses é preciso reduzí-los a múltiplos problemas de classificação entre duas classes (DUAN; KEERTHI, 2005). Dados discretos representam outro problema, embora com um reescalamento adequado bons resultados podem ser obtidos.

2.2.4 Aprendizado Baseado em Instâncias

Nesta categoria de aprendizado encontram-se os algoritmos conhecidos como de "lento aprendizado" (*lazy learning*), por haver nestes a espera pelo processo de generalização ou indução para que, após seja realizada a classificação.

Algoritmos de aprendizado lento requerem menos tempo de computação durante a fase de treinamento quando comparados aos algoritmos de aprendizado rápido. Entretanto, possuem maior tempo de computação durante o processo de classificação (AHA, 2013).

Entre os algoritmos de aprendizado lento, destaca-se o *kNN*, que baseia-se no princípio de que as instâncias de um banco de dados estarão mais próximas àquelas que possuem características semelhantes (COVER; HART, 1967). O rótulo de uma instância não classificada pode então ser determinado observando a classe mais freqüente entre seus vizinhos mais próximos.

Em geral as instâncias são consideradas como pontos no espaço $\mathbb{R}^d$, no qual cada uma das dimensões correspondem às características usadas para descrevê-las. A distância entre as instâncias é calculada usando-se uma métrica e a escolha desta métrica afeta o comportamento do classificador resultante (WANG; SUN, 2014). Para resultados mais precisos, vários algoritmos usam esquemas de ponderação e a influência de cada uma das instâncias na vizinhança para a determinação do rótulo (WETTSCHERECK; AHA; MOHRI, 1997).

O poder do *kNN* tem sido comprovado em um grande número de domínios reais. Entretanto, existem algumas limitações em sua utilidade. Destacam-se a grande exigência de armazenamento, uma vez que todos o dados disponíveis para o treinamento devem ser

utilizados, a sensibilidade à escolha da métrica utilizada para medir as distâncias e a falta de um princípio que determine a escolha do k.

O tempo de classificação dos algoritmos de lento aprendizado está fortemente relacionado ao número de instâncias disponíveis na etapa de treinamento e à quantidade de atributos que as descrevem. Para a redução do número de instâncias, algoritmos de filtragem tem sido propostos (KUBAT; JR, 2001). Os algoritmos ICF (BRIGHTON; MELLISH, 2002) e RT3 (WILSON; MARTINEZ, 2000) alcançaram altos índices de redução do número de instâncias, enquanto ainda mantêm a precisão da classificação.

Muitas técnicas também têm sido desenvolvidas para se determinar quais características devem ser utilizadas para o aprendizado, através da seleção de propriedades (YU; LIU, 2004) e redução de dimensões (FODOR, 2002).

2.2.5 Aprendizado Estatístico

A abordagem estatística caracteriza-se por uma fundamentação explicita no modelo probabilístico, em que se calcula a probabilidade de que uma instância pertença a uma certa classe em vez de simplesmente classificá-la.

Dado um vetor de teste $\boldsymbol{x}$ a ser classificado entre uma de c classes $\omega_1, \omega_2, \ldots, \omega_c$, assume-se que as instâncias do problema em questão, possuam uma densidade de probabilidade (uma *função de distribuição de probabilidade,* em geral denotada por *fdp* apenas). Com isto, um padrão $\boldsymbol{x}$ pertencendo a uma classe ω_i é entendido como uma observação aleatória de uma *fdp* classe-condicional $p(\boldsymbol{x}|\omega_i)$.

Existem várias de regras de decisão bem conhecidas e entre estas a *regra de decisão de Bayes* e a regra de máxima verossimilhança (da qual a regra de Bayes pode ser vista como um caso particular), estão entre as mais usadas para definir as fronteiras de decisão no espaço de propriedades. No capítulo 3, veremos detalhadamente a *Teoria de Bayes* onde estão definidos tais conceitos e sobre a qual estabelecem-se os principais fundamentos utilizados em nosso trabalho.

As *redes bayesianas* estão entre os mais conhecidos representantes dos algoritmos de aprendizado estatístico (JENSEN, 1996). São conhecidas como redes de crenças ou redes de Bayes simplesmente, e pertencem ao grupo dos modelos probabilísticos gráficos. Sua estrutura básica consiste de um grafo acíclico direcionado em que cada nó corresponde a uma dependência probabilística entre o respectivo atributo que representa e os seus ancestrais e descendentes. Desta forma, as *redes bayesianas* envolvem conceitos da teoria de grafos, além de conceitos de probabilidade e estatística.

As *redes bayesianas ingênuas* são redes de Bayes muito simples compostas de grafos com um único nó pai, correspondente a instância a ser classificada, e vários nós filhos com a forte hipótese de independência dos nós filhos em relação ao nó pai (GOOD, 1950). Quase sempre, a hipótese de independência está errada e por esta razão espera-se que classificadores baseados em redes bayesianas ingênuas sejam menos precisos que aqueles baseados em algoritmos de

aprendizados mais sofisticados. Entretanto, alguns estudos (DOMINGOS; PAZZANI, 1997; HAND; YU, 2001) tem mostrado que algumas vezes, sua performance supera outros esquemas de aprendizado mesmo com a presença substancial de dependência entre os atributos.

O modelo básico de independência entre os atributos tem sido modificado de várias maneiras na tentativa de melhorar a performance. A ideia principal nestas pesquisas baseia-se em adicionar ao grafo, arestas extras que incluam algumas dependências entre os atributos (FRIEDMAN; GEIGER; GOLDSZMIDT, 1997). *Classificadores bayesianos semi-ingênuos* também tentam superar a hipótese de independência, particionando os atributos em grupos e assumindo que duas características são condicionalmente independentes se, e somente se, estão em grupos diferentes (KONONENKO, 1994).

A principal vantagem dos classificadores bayesianos ingênuos é o seu curto tempo computacional para o treinamento. Além disso, como a estrutura básica deste modelo está sobre o cálculo de um produtório de probabilidades, transformá-lo na soma de vários logaritmos se traduz em significativas melhorias computacionais. Se os atributos são contínuos, é comum discretizá-los durante a etapa de pré-processamento (YANG; WEBB, 2003), entretanto distribuições normais podem ser utilizadas para o cálculo de probabilidades (BOUCKAERT, 2005).

2.3 Considerações

Neste capítulo foi apresentado um apanhado geral sobre as principais abordagens e algoritmos usados no reconhecimento de padrões. Dentre estas abordagens, destaca-se o modelo estatístico e a Teoria de Bayes, sobre os quais se fundamenta este trabalho. No capítulo 3, será apresentado o modelo de inferência bayesiana e demais conceitos utilizados nesta tese.

3

MODELO ESTATíSTICO

Learn from yesterday, live for today, hope for tomorrow. The important thing is not to stop questioning.

—A. EINSTEIN

3.1 Teoria de Bayes

A *teoria bayesiana* utiliza probabilidades para expressar informações e realizar previsões a respeito de quantidades desconhecidas. Em um sentido mais preciso, pode ser mostrado matematicamente que há um relacionamento entre a probabilidade e informações observadas, podendo inclusive, tal relacionamento ser representado através de um conjunto racional de possibilidades (HOFF, 2009).

De um modo geral, os métodos baseados na inferência bayesiana

- estimam parâmetros com boas propriedades estatísticas;
- realizam descrições seguras dos dados observados;
- fornecem estimativas de dados ausentes e previsões de dados futuros;
- constituem uma estrutura computacional para a estimativa, seleção e validação do modelo.

Nas próximas seções descrevemos os conceitos básicos da *teoria da probabilidade* (KALLENBERG, 2002; JAYNES, 2003) sobre os quais se estabelece a *teoria de Bayes*.

3.1.1 Vetores Randômicos

O conceito de *vetor randômico* é uma generalização de *variável aleatória* (MONTGOMERY; RUNGER, 2010). Suponha a condução de um experimento probabilístico

em que os possíveis resultados sejam descritos pelo espaço amostral Ω. Um vetor randômico é um vetor cujas coordenadas dependem dos resultados do experimento, conforme enunciado na seguinte definição

Definição 3.1. Um vetor randômico é dado por um mapeamento

$$\boldsymbol{x} : \Omega \to X \subseteq \mathbb{R}^k$$

que associa um vetor $\boldsymbol{x}(\omega)$ de k números reais a cada resultado elementar $\omega \in \Omega$, sendo Ω o espaço amostral.

De um modo mais rigoroso, a *teoria da probabilidade* exige ainda que a função $\boldsymbol{x}$ seja mensurável (KALLENBERG, 2002). O vetor real $\boldsymbol{x}(\omega)$ associado à amostra $\omega \in \Omega$ é chamado uma *realização* do vetor randômico. O conjunto de todas as possíveis realizações de $\boldsymbol{x}$ é denominado *suporte* e é denotado por $R_{\boldsymbol{x}}$.

Denota-se por $P(A)$ a probabilidade de ocorrer um evento $A \subseteq \Omega$. As seguintes convenções também são utilizadas:

- Se $A \subseteq \mathbb{R}^k$, escreve-se $P(X \subseteq A)$ com significado

  $$P(\boldsymbol{x} \in A) = P\left(\{\omega \in \Omega | \, \boldsymbol{x}(\omega) \in A\}\right)$$

  ou seja, a probabilidade de que ocorra um evento que esteja em A;

- Novamente, considerando $A \subseteq \mathbb{R}^k$, é comum usar-se a notação $P_{\boldsymbol{x}}(A)$ tendo o significado

  $$P_{\boldsymbol{x}}(A) = P(\boldsymbol{x} \in A);$$

- Em geral, escreve-se apenas $\boldsymbol{x}$ em vez de $\boldsymbol{x}(\omega)$, omitindo-se portanto a dependência de ω.

Suponha que após atribuirmos $P(A)$ aos eventos $A \subseteq \Omega$, recebamos informação sobre novos eventos que irão ocorrer. Em particular, suponha que tais eventos pertençam ao conjunto $B \subseteq \Omega$. Denota-se por $P\left(A | \, B\right)$ a probabilidade revisada de que um evento $A \subseteq \Omega$ ocorra após sabermos que tenha ocorrido o evento B. $P\left(A | \, B\right)$ é chamada *probabilidade condicional* de A dado B.

Definição 3.2. A probabilidade condicional de que um evento A ocorra dado que um evento B ocorreu, denotada por $P\left(A | \, B\right)$ é dada por

$$P\left(A | \, B\right) = \frac{P\left(A \cap B\right)}{P(B)}$$

desde que $P(B) > 0$.

De acordo com esta definição, observa-se que

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

e

$$P(B|A) = \frac{P(B \cap A)}{P(A)}$$

Disto segue-se que

$$P(B|A)\,P(A) = P(A|B)\,P(B)$$

Ou seja, como consequência direta da definição 3.2 tem-se o importante teorema:

Teorema 3.1 (de Bayes)**.** *Seja* $\{A_1, A_2, \ldots, A_n\}$ *um conjunto de eventos mutuamente exclusivos que, juntos, formam o espaço amostral* S*. Seja* B *um evento arbitrário de* S *tal que* $P(B) > 0$*. Então*

$$P(A_k|B) = \frac{P(B|A_k)\,P(A_k)}{P(B)} \tag{3.1}$$

sendo

$$P(B) = \sum_{k=1}^{n} P(B|A_k)\,P(A_k)$$

3.1.2 Aprendizado Bayesiano

No reconhecimento de padrões, lida-se com vetores randômicos extraídos de diferentes classes (grupos ou categorias), cada uma tendo sua própria *função de densidade de probabilidade* (*fdp*). Esta função de densidade também é chamada de *densidade da classe* ω_i ou *densidade condicional* da classe ω_i e é expressa por

$$p(\boldsymbol{x}|\omega_i) \text{ ou } p_i(\boldsymbol{x}),\ i = 1, \ldots, C$$

em que ω_i indica a classe e C o número de classes.

A função de *densidade não condicional* de $\boldsymbol{x}$ por sua vez é dada por

$$p(\boldsymbol{x}) = \sum_{i=1}^{C} p(\boldsymbol{x}|\omega_i)\,P(\omega_i)$$

sendo $P(\omega_i)$ a probabilidade *a priori* da classe ω_i (THEODORIDIS; KOUTROUMBAS, 2008; FUKUNAGA, 1972).

A probabilidade *a posteriori* de ocorrer ω_i dado $\boldsymbol{x}$, ou seja, a probabilidade de que $\boldsymbol{x}$ esteja na classe ω_i, denotada por $P(\omega_i|\boldsymbol{x})$ pode ser calculada usando o *teorema de Bayes*, da seguinte forma

$$P(\omega_i|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\omega_i)\,P(\omega_i)}{p(\boldsymbol{x})} \tag{3.2}$$

De modo que, a *regra de decisão de Bayes* atribui a classe ω_i a um padrão de teste $\boldsymbol{x}$ se

$$P(\omega_i|\boldsymbol{x}) > P(\omega_j|\boldsymbol{x}), \forall j \neq i \tag{3.3}$$

Os casos em que houver igualdade, o padrão pode ser atribuido a qualquer uma das classes envolvidas.

Em geral, assume-se que as probabilidades *a priori* $P(\omega_i)$ são iguais, ou seja, há igual probabilidade de que ocorra qualquer uma das classes e neste caso, usando-se a Eq. (3.2), a regra de decisão definida pela Eq. (3.3) pode ser reescrita da seguinte forma: atribua a classe ω_i a um padrão de teste $\boldsymbol{x}$ se

$$p(\boldsymbol{x}|\omega_i) > p(\boldsymbol{x}|\omega_j), \forall j \neq i$$

Ou seja, a *classificação de Bayes* consiste da busca por um máximo entre as *funções densidade de probabilidade* condicionais calculadas em $\boldsymbol{x}$.

3.1.3 Erro de Classificação

Considere P_c sendo a probabilidade de ocorrer uma classificação correta. Esta probabilidade é dada por

$$P_c = \sum_{i=1}^{C} P(\boldsymbol{x} \in R_i, \omega_i) \tag{3.4}$$

sendo $\boldsymbol{x}$ um vetor de teste que está em R_i, que é a região do espaço $\mathbb{R}^k$ onde estão as instâncias (vetores) da classe ω_i. Além disso, o termo $P(\boldsymbol{x} \in R_i, \omega_i)$ denota a *probabilidade da interseção*, ou seja, avalia a possibilidade de ocorrerem simultaneamente $\boldsymbol{x} \in R_i$ e ω_i, e pode ser calculada, (FUKUNAGA, 1972), da seguinte forma

$$\begin{aligned} P(\boldsymbol{x} \in R_i, \omega_i) &= P(\boldsymbol{x} \in R_i|\omega_i)\, P(\omega_i) \\ &= \left(\int_{R_i} p(\boldsymbol{x}|\omega_i)\, d\boldsymbol{x} \right) P(\omega_i) \end{aligned} \tag{3.5}$$

Assim, a Eq. (3.4) pode ser reescrita como

$$P_c = \sum_{i=1}^{C} \left(\int_{R_i} p(\boldsymbol{x}|\omega_i)\, d\boldsymbol{x} \right) P(\omega_i) \Leftrightarrow$$

$$P_c = \sum_{i=1}^{C} \int_{R_i} p(\boldsymbol{x}|\omega_i)\, P(\omega_i) d\boldsymbol{x} \tag{3.6}$$

Como esta é a soma de C diferentes termos não negativos, para que tenha-se valor máximo para P_c. Ou seja, para se maximizar a probabilidade de acerto ou, por outro lado, minimizar a probabilidade de erro, é suficiente atribuir R_i como a região do espaço onde $\boldsymbol{x}$ estará, o que é

equivalente a atribuir $\boldsymbol{x}$ à classe ω_i, se

$$p(\boldsymbol{x}|\omega_i)\,P(\omega_i) > p(\boldsymbol{x}|\omega_j)\,P(\omega_j), \forall j \neq i \tag{3.7}$$

Dividindo ambos os lados desta desigualdade, por $p(\boldsymbol{x})$ e usando o *Teorema de Bayes*, tem-se

$$P(\omega_i|\,\boldsymbol{x}) > P(\omega_j|\,\boldsymbol{x}), \forall j \neq i \tag{3.8}$$

Ou seja, ao usar-se a *regra de decisão de Bayes*, incorre-se em menor probabilidade de erro.

3.1.4 Risco Médio

A minimização da probabilidade de erro nem sempre é o melhor critério a ser adotado em processos de classificação. Isto porquê é dada a mesma importância a todos os erros. Entretanto, existem situações em que algumas decisões erradas podem ter implicações mais sérias que outras.

O caso em que um médico tenha que diagnosticar um câncer, por exemplo. Será muito mais grave classificá-lo como benigno, quando o correto seria maligno. Enquanto que, diagnosticar como maligno um tumor que de fato seja benigno, poderá posteriormente ser diagnosticado corretamente em exames futuros. Por outro lado, a decisão errada, tomada na primeira situação pode ser fatal. Em tais casos é mais apropriado associar um termo de penalidade para ponderar cada erro.

Voltando ao problema com C classes, considere novamente R_i, $i = 1, \ldots, C$ sendo as regiões do espaço com características relativas à classe ω_i. Suponha que o vetor de teste $\boldsymbol{x}$ pertença à classe ω_k e tenha sido erroneamente designado à região R_i com $i \neq k$. Ou seja, houve um erro de classificação.

Um termo de penalidade λ_{ki}, conhecido como *perda* (ou *custo*) é associado com esta decisão errada. A matriz que tem em sua entrada (k, i) o termo λ_{ki} correspondente, é denominada *matriz de perda* (ou *matriz de custo*). Na diagonal desta matriz estão os elementos λ_{ii} correspondentes às decisões corretas. Na prática, estes termos são em geral iguais a zero (já que não há *perda* ou *custo* ao se tomar a decisão correta) e são considerados apenas para efeito de generalização.

O *risco* ou *perda* (ou *custo*) associado com a classe ω_k é definido como

$$r_k = \sum_{i=1}^{C} \lambda_{ki} \left(\int_{R_i} p(\boldsymbol{x}|\omega_k)\, d\boldsymbol{x} \right) \tag{3.9}$$

ou seja, o risco é a soma de todas as possibilidades de vetores das classes ω_k, serem atribuidos à classe ω_i, multiplicadas pelas suas respectivas penalidades.

O objetivo agora é determinar a qual região R_i deve ser atribuido o vetor de teste $\boldsymbol{x}$, de

modo que o *risco médio*, dado por

$$\begin{aligned} r &= \sum_{k=1}^{C} r_k \, P(\omega_k) \\ &= \sum_{k=1}^{C} \left(\sum_{i=1}^{C} \lambda_{ki} \left(\int_{R_i} p\left(\boldsymbol{x}|\omega_k\right) d\boldsymbol{x} \right) \right) P(\omega_k) \\ &= \sum_{i=1}^{C} \int_{R_i} \left(\sum_{k=1}^{C} \lambda_{ki} p\left(\boldsymbol{x}|\omega_k\right) P(\omega_k) \right) d\boldsymbol{x} \end{aligned} \tag{3.10}$$

seja minimizado.

Usando mais uma vez o *teorema de Bayes*, observa-se que

$$p\left(\boldsymbol{x}|\omega_k\right) P(\omega_k) = P\left(\omega_k|\boldsymbol{x}\right) p(\boldsymbol{x}) \tag{3.11}$$

e disto segue-se que o *risco médio* (Eq. (3.10)) pode ser reescrito como

$$\begin{aligned} r &= \sum_{i=1}^{C} \int_{R_i} \left(\sum_{k=1}^{C} \lambda_{ki} P\left(\omega_k|\boldsymbol{x}\right) \right) p(\boldsymbol{x}) \, d\boldsymbol{x} \\ &= \sum_{i=1}^{C} \int_{R_i} l_i \, p(\boldsymbol{x}) \, d\boldsymbol{x} \end{aligned} \tag{3.12}$$

sendo

$$l_i = \sum_{k=1}^{C} \lambda_{ki} P\left(\omega_k|\boldsymbol{x}\right) \tag{3.13}$$

Observando-se que (3.12) consiste do somatório de C termos não-negativos, uma vez que $l_i \geq 0$ e $p(\boldsymbol{x}) \geq 0$, e que $p(\boldsymbol{x})$ é o mesmo em todos os termos, a minimização desta expressão, será alcançada atribuindo-se o vetor de teste $\boldsymbol{x}$ à região R_i se

$$l_i < l_j, \forall j \neq i.$$

Por outro lado, se usarmos a *regra de decisão de Bayes*, ou seja, atribuirmos $\boldsymbol{x}$ à classe ω_i caso

$$P\left(\omega_i|\boldsymbol{x}\right) > P\left(\omega_j|\boldsymbol{x}\right), \forall j \neq i \tag{3.14}$$

teremos

$$l_i - l_j = \sum_{k=1}^{C} \left(\lambda_{ki} - \lambda_{kj}\right) P\left(\omega_k|\boldsymbol{x}\right)$$

Lembrando que $\lambda_{ii} = 0$ e admitindo que $\lambda_{ki} < \lambda_{kj}$, ou seja , a penalidade em se atribuir um vetor da classe ω_k à classe ω_i (diagnosticar como maligno um tumor benigno, por exemplo), seja

menor que quaisquer outras penalidades. Teremos, portanto

$$\begin{aligned} l_i - l_j = &(\lambda_{1i} - \lambda_{1j})\, P(\omega_1|\boldsymbol{x}) + \cdots + (\lambda_{ii} - \lambda_{ij})\, P(\omega_i|\boldsymbol{x}) + \cdots + \\ &+ (\lambda_{ji} - \lambda_{jj})\, P(\omega_j|\boldsymbol{x}) + \cdots + (\lambda_{ci} - \lambda_{cj})\, P(\omega_c|\boldsymbol{x}) \\ \\ = &(\lambda_{1i} - \lambda_{1j})\, P(\omega_1|\boldsymbol{x}) + \cdots - \lambda_{ij} P(\omega_i|\boldsymbol{x}) + \cdots + \\ &+ \lambda_{ji} P(\omega_j|\boldsymbol{x}) + \cdots + (\lambda_{ci} - \lambda_{cj})\, P(\omega_c|\boldsymbol{x}) \\ \\ < &-\lambda_{ij} P(\omega_i|\boldsymbol{x}) + \lambda_{ji} P(\omega_j|\boldsymbol{x}) \end{aligned}$$

e pela suposição que foi feita em (3.14), segue-se que

$$\begin{aligned} l_i - l_j &< -\lambda_{ij} P(\omega_i|\boldsymbol{x}) + \lambda_{ji} P(\omega_j|\boldsymbol{x}) \\ &< -\lambda_{ij} P(\omega_i|\boldsymbol{x}) + \lambda_{ji} P(\omega_i|\boldsymbol{x}) \\ &= (\lambda_{ji} - \lambda_{ij})\, P(\omega_i|\boldsymbol{x}) \end{aligned}$$

e como $\lambda_{ji} < \lambda_{ij}$, tem-se

$$\begin{gathered} l_i - l_j < 0,\ \forall j \neq i \\ \Leftrightarrow \\ l_i < l_j,\ \forall j \neq i \end{gathered}$$

Ou seja, ao utilizar-se a *regra de decisão de Bayes* produz-se também o menor *risco médio*, para quaisquer perdas associadas aos erros cometidos.

3.2 Distribuições Gaussianas

3.2.1 A Função de densidade de probabilidade Gaussiana

Em geral denominada apenas de *função de densidade normal* (também por *distribuição normal* ou *distribuição Gaussiana*) esta é uma das *fdp*'s mais comumente encontrada na prática. As razões para esta popularidade são a sua tratabilidade computacional e o fato de que modela adequadamente um grande número de casos reais.

Um dos mais célebres teoremas em estatística, o *teorema Central do Limite*, enuncia que se uma variável randômica é o resultado de um somatório de outras variáveis também randômicas e independentes, sua *fdp* se aproxima de uma *fdp Gaussiana* à medida que o número de termos neste somátorio tende ao infinito. Na prática, é comum assumir que a soma de variáveis randômicas está distribuída de acordo com uma *fdp Gaussiana* para um número suficientemente

grande de termos somados.

A *fdp Gaussiana* em um espaço n-dimensional é dada por

$$p(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{x}-\boldsymbol{\mu})}, \tag{3.15}$$

sendo $\boldsymbol{\mu} = E[\boldsymbol{x}]$ o valor médio de $\boldsymbol{x}$ e Σ a matriz de covariância, dada por

$$\Sigma = E\left[(\boldsymbol{x}-\boldsymbol{\mu})^T (\boldsymbol{x}-\boldsymbol{\mu})\right] \tag{3.16}$$

e $|\Sigma|$ denota o determinante de Σ. É comum utilizar-se a notação

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$$

significando que a variável randômica $\boldsymbol{x}$ está *distribuída normalmente* ou que admite uma *fdp gaussiana* com vetor médio $\boldsymbol{\mu}$ e matriz de covariância Σ.

O termo $(\boldsymbol{x}-\boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{x}-\boldsymbol{\mu})$ presente no expoente da expressão dada em (3.15) é o quadrado da *distância generalizada* de $\boldsymbol{x}$ a $\boldsymbol{\mu}$, ou como é mais conhecida, *distância de Mahalanobis*,

$$\Delta^2 = (\boldsymbol{x}-\boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{x}-\boldsymbol{\mu}). \tag{3.17}$$

Uma coleção de propriedades a respeito das *distribuições gaussianas* pode ser verificada e para isso recomenda-se a leitura de Rencher (2003), Johnson e Wichern (2007).

3.2.2 Classificador de Bayes em Distribuições Gaussianas

Nesta seção usaremos a *regra de decisão de Bayes* (Eq. (3.3)), para o desenvolvimento do *Classificador de Bayes* para dados cujas classes possuem *fdp*'s $p(\boldsymbol{x}|\omega_i), i = 1, \ldots, C$ normalmente distribuídas. Ou seja, cada uma das classes ω_i obedece a uma distribuição gaussiana $\mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i)$, sendo $\boldsymbol{\mu}_i$ e Σ_i os respectivos vetor médio e matriz de covariância da classe ω_i.

De acordo com o *teorema de Bayes* (Eq. (3.1)), temos

$$P(\omega_i|\boldsymbol{x}) = \frac{P(\omega_i) p(\boldsymbol{x}|\omega_i)}{p(\boldsymbol{x})}. \tag{3.18}$$

Como a classe ω_i está distribuída normalmente, é válido que

$$p(\boldsymbol{x}|\omega_i) = \frac{1}{\sqrt{(2\pi)^n |\Sigma_i|}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T \Sigma_i^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_i)}. \tag{3.19}$$

Assim, a *probabilidade a posteriori* de que o vetor de teste $\boldsymbol{x}$ esteja na classe ω_i é dada por

$$P(\omega_i|\boldsymbol{x}) = \frac{P(\omega_i)}{p(\boldsymbol{x})\sqrt{(2\pi)^n |\Sigma_i|}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T \Sigma_i^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_i)} \tag{3.20}$$

Como $p(\boldsymbol{x})$ não depende da classe, o *classificador de Bayes* para dados gaussianos, atribui o vetor de teste $\boldsymbol{x}$ à classe ω_i caso

$$p_i(\boldsymbol{x}) = \max_{1 \leq j \leq C} p_j(\boldsymbol{x}) \tag{3.21}$$

sendo

$$p_i(\boldsymbol{x}) = P(\omega_i) \frac{e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T \Sigma_i^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_i)}}{\sqrt{(2\pi)^n |\Sigma_i|}}. \tag{3.22}$$

Em virtude da complexidade em se calcular $p_i(\boldsymbol{x})$, é comum buscar alternativas mais simples e de algum modo estimar a *distância* de um dado vetor de teste $\boldsymbol{x}$, à cada uma das classes e designá-lo àquela que esteja mais próxima. Uma forma simplificada de se realizar esta estimativa é feita aplicando-se o logaritmo natural à Eq. (3.22) donde obtêm-se

$$\begin{aligned} d_j(\boldsymbol{x}) &= \ln p_j(\boldsymbol{x}) \\ &= \ln P(\omega_j) + \ln \frac{e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_j)^T \Sigma_j^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_j)}}{\sqrt{(2\pi)^n |\Sigma_j|}} \\ &= \ln P(\omega_j) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_j)^T \Sigma_j^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_j) - \frac{1}{2} \ln (2\pi)^n |\Sigma_j| \\ &= \ln P(\omega_j) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_j)^T \Sigma_j^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_j) - \frac{n}{2} \ln 2\pi - \frac{1}{2} \ln |\Sigma_j| \,. \end{aligned} \tag{3.23}$$

Reorganizando a expressão obtida na Eq. (3.23) e observando que o termo $\frac{n}{2} \ln 2\pi$ é o mesmo, independente da classe, obtemos

$$d'_j(\boldsymbol{x}) = \ln |\Sigma_j| + (\boldsymbol{x}-\boldsymbol{\mu}_j)^T \Sigma_j^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_j) - 2 \ln P(\omega_j) \tag{3.24}$$

e atribui-se a classe ω_i ao vetor $\boldsymbol{x}$ se

$$d'_i(\boldsymbol{x}) = \min_{1 \leq j \leq C} d'_j(\boldsymbol{x}) \tag{3.25}$$

Esta é uma versão mais simples e eficiente do *classificador de Bayes* para dados gaussianos.

3.3 Vetor Médio e Matriz de Covariância

Embora assuma-se que a *fdp* de cada uma das classes sejam conhecidas, não é este o caso em geral. No caso geral, estas *fdp*'s devem ser estimadas a partir dos dados disponíveis.

Quando uma distribuição é supostamente *normal*, as estimativas de seus parâmetros, $\boldsymbol{\mu}$ e Σ, são frequentemente encontrados através do método conhecido como *máxima verossimilhança*.

Esta técnica consiste em, dados os vetores $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$ observados, determinarmos os valores de $\boldsymbol{\mu}$ e Σ que maximizam a *função de densidade conjunta*, conhecida como *função de*

verossimilhança.

Definição 3.3. Sejam $\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_N}$ amostras obtidas de uma *fdp* $p(\boldsymbol{x}|\theta)$. Supondo a indepêndencia estatística entre as amostras define-se a *função de verosimilhança* de θ com respeito a $X = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_N}\}$ como

$$p(X|\theta) = \prod_{i=1}^{N} p(\boldsymbol{x_i}|\theta). \tag{3.26}$$

O *método da máxima verossimilhanaça* estima o parâmetro θ de modo que a função de verossimilhança alcance seu valor máximo, ou seja

$$\hat{\theta} = \arg\max_{\theta} \prod_{i=1}^{N} p(\boldsymbol{x}_i|\theta).$$

Teorema 3.2. *Sejam $\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_N}$ vetores randômicos obtidos de uma distribuição normal com vetor médio $\boldsymbol{\mu}$ e matriz de covariância Σ. Então*

$$\hat{\boldsymbol{\mu}} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{x_i}, \tag{3.27}$$

$$\hat{\Sigma} = \frac{1}{N-1} \sum_{i=1}^{N} (\boldsymbol{x_i} - \hat{\boldsymbol{\mu}})^T (\boldsymbol{x_i} - \hat{\boldsymbol{\mu}}) \tag{3.28}$$

são as estimativas de máxima verossimilhança de $\boldsymbol{\mu}$ e Σ, respectivamente.
*Demonstração.: ver Johnson e Wichern (2007), **Result 4.11**, pg. 171.*

3.4 Máxima Entropia

O *princípio da máxima entropia* (JAYNES, 1968; GOOD, 1963) enuncia que ao buscar uma distribuição de probabilidade que satisfaça algumas restrições, o correto é escolher aquela que maximize a incerteza, ou *entropia*, sujeita a estas restrições.

Sendo $p(\boldsymbol{x})$ a *fdp* de uma variável aleatória, a *entropia* desta distribuição define-se, segundo a *teoria da informação de Shannon* (SHANNON, 1948), como

$$h(p(\boldsymbol{x})) = -\int_{\boldsymbol{x}} p(\boldsymbol{x}) \ln(p(\boldsymbol{x})) d\boldsymbol{x}. \tag{3.29}$$

Teorema 3.3. *Seja $q(\boldsymbol{x})$ uma fdp qualquer da variável randômica $\boldsymbol{x}$, com vetor médio $\boldsymbol{\mu}$ e matriz*

de covariância Σ. Considere ainda $p(\boldsymbol{x})$ a fdp gaussiana tal que $p(\boldsymbol{x}) \sim N(\boldsymbol{\mu}, \Sigma)$. Então

$$h(q(\boldsymbol{x})) \leq h(p(\boldsymbol{x})).$$

Demonstração.: Ver Murphy (2012).

Assim, a *distribuição gaussiana* é aquela de máxima desordem (máxima entropia) entre todas as distribuições com vetor médio $\boldsymbol{\mu}$ e covariância Σ. Ao supor-se a normalidade dos dados em problemas de reconhecimento de padrões, evita-se portanto, introduzir expectativas espúrias sobre os dados, mantendo o problema o mais genérico possível.

3.5 Considerações

Apresentou-se neste capítulo, os fundamentos do aprendizado bayesiano. O classificador de Bayes, construído a partir desta forma de aprendizagem, possui entre suas qualidades, a menor probabilidade de erro e menor risco médio para quaisquer formas de se penalizar os erros cometidos.

Entretanto, este classificador possui apenas valor teórico, haja visto não ser possível conhecer *a priori* a função de distribuição de probabilidade dos dados envolvidos em problemas reais. Como alternativa para suprir esta necessidade, supõe-se a normalidade dos dados. Neste caso, tal hipótese possui impacto mínimo, visto que a distribuição gaussiana, entre todas as distribuições possíveis, é aquela que possui maior entropia.

Porém, o classificador de Bayes, mesmo com esta construção, permanece teórico, dado que o vetor médio e a matriz de covariância de dados reais são igualmente desconhecidos *a priori*. Assim, o classificador construído a parti do aprendizado bayesiano, supondo a normalidade dos dados e usando as estimativas de máxima verossimilhança para o vetor médio e matriz de covariância destes dados, trata-se na verdade, de uma aproximação do classificador de Bayes.

No capítulo 4 será mostrado que perturbações nos parâmetros $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ ocorrem quando uma nova amostra é inserida na classe ω_i. Estas perturbações quando avaliadas e comparadas possuem poder discriminatório e a partir delas uma nova abordagem para a classificação é construída.

4

CLASSIFICAÇÃO DE PADRÕES BASEADA EM PERTURBAÇÕES

Science never solves a problem without creating ten more.

—GEORGE BERNARD SHAW

4.1 Introdução

O *classifcador de Bayes*, para dados que obedecem uma distribuição gaussiana, possui uma dependência intrínseca dos vetores médios $\boldsymbol{\mu}_i$ e das matrizes de covariância Σ_i das classes presentes nos dados, conforme a teoria apresentada no capítulo anterior.

Em geral, estes parâmetros são desconhecidos e apesar de existirem trabalhos sobre estimativas para o vetor médio (IOSIFIDIS; TEFAS; PITAS, 2013) e para a matriz de covariância (HOFFBECK; LANDGREBE, 1996; KUO; LANDGREBE, 2002; LEDOIT; WOLF, 2004; PERRON, 1992; TADJUDIN; LANDGREBE, 1999), as estimativas de máxima verossimilhança $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ (Eqs. (3.27) e (3.28)) obtidas a partir dos dados disponíveis são ainda as mais utilizadas para este tipo de classificador.

Segundo Theodoridis e Koutroumbas (2008), tais estimativas são assintoticamente *imparciais*, uma vez que, na média convergem para os valores reais, *consistentes*, no sentido em que quanto maior for o número N_i de amostras disponíveis, tais estimativas estarão arbitrariamente próximas de seus reais valores, e *eficientes* por apresentarem menor variância que *quaisquer outras* estimativas. Entretanto, todas estas boas propriedades são válidas apenas para grandes valores de N_i.

Admitindo as estimativas de máxima verossimilhança como escolha ideal para os *classificadores de Bayes* em dados supostamente gaussianos, a função utilizada pela *regra*

de decisão de Bayes (Eq. 3.22) é aproximada pela função

$$\hat{\boldsymbol{p}}_i(\boldsymbol{x}) = P(\omega_i)\frac{e^{-\frac{1}{2}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)^T\hat{\Sigma}_i^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)}}{\sqrt{(2\pi)^n\,|\hat{\Sigma}_i|}} \tag{4.1}$$

sendo

$$\hat{\boldsymbol{\mu}}_i = \frac{1}{N_i}\sum_{j=1}^{N_i}\boldsymbol{x_{i,j}}\,, \tag{4.2}$$

$$\hat{\Sigma}_i = \frac{1}{N_i-1}\sum_{j=1}^{N_i}(\boldsymbol{x_{i,j}}-\hat{\boldsymbol{\mu}}_i)(\boldsymbol{x_{i,j}}-\hat{\boldsymbol{\mu}}_i)^T \tag{4.3}$$

com $\boldsymbol{x_{i,j}}$ sendo o j*-ésimo* vetor da classe ω_i e N_i o número de vetores da classe ω_i.

A função dada em (Eq. 4.1) representará melhor um *classificador de Bayes*, quanto mais próximas estiverem as estimativas $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ de seus valores reais $\boldsymbol{\mu}_i$ e Σ_i.

Suponha portanto que a função $\hat{\boldsymbol{p}}_i(\boldsymbol{x})$ (Eq. 4.1) seja contínua e que dependa não apenas de $\boldsymbol{x}$ mas também dos parâmetros $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$. Considere $\boldsymbol{x}$ como um vetor de teste a ser classificado e suponha ainda que $\boldsymbol{x}$ tenha sido atribuído à classe ω_i. Ao inserirmos $\boldsymbol{x}$ nesta classe, os valores de $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ serão alterados, causando assim *perturbações* em ambos. Denotemos por $\hat{\boldsymbol{\mu}}_i'$ e $\hat{\Sigma}_i'$ os novos valores de $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$.

Tem-se então,

$$\hat{\boldsymbol{\mu}}_i' = \hat{\boldsymbol{\mu}}_i + \Delta\hat{\boldsymbol{\mu}}_i\,, \tag{4.4}$$

$$\hat{\Sigma}_i' = \hat{\Sigma}_i + \Delta\hat{\Sigma}_i \tag{4.5}$$

sendo $\Delta\hat{\boldsymbol{\mu}}_i$ e $\Delta\hat{\Sigma}_i$ as alterações (perturbações) ocorridas em $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$, respectivamente, após a inclusão de $\boldsymbol{x}$ em ω_i. A classe ω_i possui agora uma nova *fdp*, dada por

$$\hat{\boldsymbol{p}}_i'(\boldsymbol{x}) = P(\omega_i)\frac{e^{-\frac{1}{2}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i')^T(\hat{\Sigma}_i')^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i')}}{\sqrt{(2\pi)^n\,|\hat{\Sigma}_i'|}}\,. \tag{4.6}$$

De acordo com a *teoria da aproximação* (ACHIESER, 2013), a função $\hat{\boldsymbol{p}}_i'(\boldsymbol{x})$ (Eq. 4.6) será uma boa aproximação da função $\hat{\boldsymbol{p}}_i(\boldsymbol{x})$ (Eq. 4.1), se

$$\left|\hat{\boldsymbol{p}}_i(\boldsymbol{x}) - \hat{\boldsymbol{p}}_i'(\boldsymbol{x})\right| \to 0$$

à medida[1] que $\|\Delta\hat{\boldsymbol{\mu}}_i\| \to 0$, $\left\|\Delta\hat{\Sigma}_i\right\| \to 0$. Em outras palavras, a nova função $\hat{\boldsymbol{p}}'_i(\boldsymbol{x})$ dada em (Eq. (4.6)) será uma melhor aproximação de $\hat{\boldsymbol{p}}_i(\boldsymbol{x})$ (Eq. (4.1)), quanto menores forem as perturbações em $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$.

Neste trabalho propõe-se que tais perturbações ou uma combinação de ambas possam ser utilizadas para propósitos de classificação e para isto sugere-se como *regra de decisão*, que um dado vetor de teste $\boldsymbol{x}$ seja atribuido à classe ω_i se as perturbações em $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$, ou uma combinação destas, forem as menores entre todas as perturbações das classes, ou seja

$$\boldsymbol{x} \in \omega_i \quad \text{se} \quad \Delta f_i < \Delta f_j, \forall j \neq i$$

sendo f_i uma função responsável por tentar descrever a distribuição dos dados da classe ω_i e para isto utiliza os parâmetros $\hat{\boldsymbol{\mu}}_i$ e/ou $\hat{\Sigma}_i$, ou seja

$$f_i = f(\hat{\boldsymbol{\mu}}_i),$$

$$f_i = g(\hat{\Sigma}_i)$$

ou

$$f_i = h(\hat{\boldsymbol{\mu}}_i, \hat{\Sigma}_i).$$

No primeiro caso, $f_i = f(\hat{\boldsymbol{\mu}}_i)$, a distribuição dos dados seria descrita através do parâmetro $\hat{\boldsymbol{\mu}}_i$ apenas. Se, por outro lado, é possível descrever a distribuição dos dados usando-se apenas o parâmetro $\hat{\Sigma}_i$, tem-se o caso em que $f_i = g(\hat{\Sigma}_i)$. Para os dados cuja distribuição forem necessários os parâmetros $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ para descrevê-la, o terceiro caso, $f_i = h(\hat{\boldsymbol{\mu}}_i, \hat{\Sigma}_i)$ deve ser utilizado.

4.1.1 Perturbações em $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ e Classificação

Nesta seção avalia-se o quanto podem ser úteis para a classificação de padrões, as perturbações dos vetores médios $\hat{\boldsymbol{\mu}}_i$ e das matrizes de covariância $\hat{\Sigma}_i$, $\Delta\hat{\boldsymbol{\mu}}_i$ e $\Delta\hat{\Sigma}_i$ respectivamente, obtidas a partir dos dados de treinamento e do vetor de teste $\boldsymbol{x}$.

Especificamente, a inserção do vetor de teste $\boldsymbol{x}$ na classe ω_i, implica alterações em $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$. Como mencionado anteriormente, suspeita-se que estas alterações, as diferenças entre os valores dos vetores médios e das matrizes de covariâcia, antes e depois da inserção do vetor de teste,

$$\Delta\hat{\boldsymbol{\mu}}_i = \hat{\boldsymbol{\mu}}'_i - \hat{\boldsymbol{\mu}}_i\,,$$

$$\Delta\hat{\Sigma}_i = \hat{\Sigma}'_i - \hat{\Sigma}_i$$

sejam relevantes e úteis para a classificação de $\boldsymbol{x}$.

[1] Sendo mais rigoroso deve-se mencionar também que $\|\Delta\boldsymbol{x}\| \to 0$. Contudo, isto não foi feito, por não haver alterações no vetor de teste $\boldsymbol{x}$.

Para exemplificar esta suspeita, considere um banco de dados com *distribuição Gaussiana* (Figura 4.1a), 2 dimensões, com 2 classes e 80 instâncias, 40 em cada classe, geradas artificialmente. Para gerar estes dados, usou-se

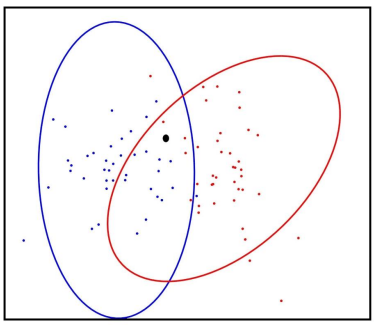
$$\hat{\boldsymbol{\mu}}_1 = (-0,4222; -0,0497)\,,$$
$$\hat{\boldsymbol{\mu}}_2 = (\ \ 0,3671;\ \ 0,0124)$$

e

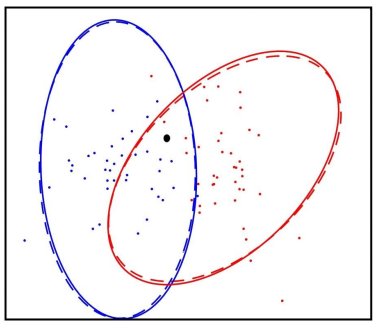
$$\hat{\Sigma}_1 = \begin{bmatrix} 0,1062 & 0,0404 \\ 0,0404 & 0,0943 \end{bmatrix}, \hat{\Sigma}_2 = \begin{bmatrix} 0,0525 & -0,0089 \\ -0,0089 & 0,0744 \end{bmatrix}$$

Figura 4.1: Calculando alterações e verificando sua influência

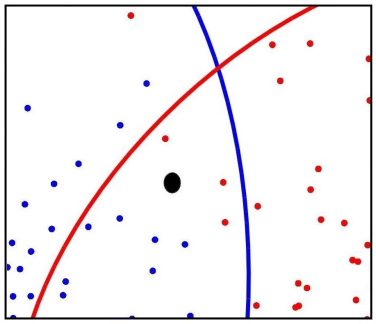
(a) Dados antes da inserção do vetor de teste. Elipses delimitam a região de confiança com 99% de certeza.



(b) Dados após a inserção do vetor de testes em cada uma das classes. Elipses tracejadas indicam a região de confiança antes da inserção.



(c) Usando o *INN* a distância do vetor de teste à classe 2 ($d_2 = 0,000823$), é menor que a distância á classe 1 ($d_1 = 0,001812$), classificando erroneamente, o vetor de teste como sendo da classe 2.



O vetor de teste $\boldsymbol{x} = (-0,3306;\ \ 0,0655)$ (ponto em preto) é inserido simultaneamente nas classes ω_1 (pontos em azul) e ω_2 (pontos em vermelho) e sabe-se previamente que $\boldsymbol{x}$ pertence

à classe ω_1 (Figura 4.1a). Após esta inserção (Figura 4.1b), foram recalculadas as estimativas para os vetores médios e matrizes de covariância de ambas as classes e obtemos,

$$\hat{\boldsymbol{\mu}}_1' = (-0,4200; -0,0469)\,,$$
$$\hat{\boldsymbol{\mu}}_2' = (\quad 0,3501; \quad 0;0137)$$

e

$$\hat{\Sigma}_1' = \begin{bmatrix} 0,1040 & 0,0399 \\ 0,0399 & 0,0924 \end{bmatrix}, \quad \hat{\Sigma}_2' = \begin{bmatrix} 0,0631 & 0,0033 \\ 0,0033 & 0,0845 \end{bmatrix}.$$

Utilizando a norma euclideana para vetores e a norma de *Frobenius* para matrizes, a diferença obtida é

$$\|\Delta\hat{\boldsymbol{\mu}}_1\| = 0,003590\,, \quad \|\Delta\hat{\boldsymbol{\mu}}_2\| = 0,017065$$

e

$$\left\|\Delta\hat{\Sigma}_1\right\| = 0,002884\,, \quad \left\|\Delta\hat{\Sigma}_2\right\| = 0,022592.$$

Ou seja, as alterações no vetor médio e na matriz de covariância da classe ω_1, a qual o vetor $\boldsymbol{x}$ pertence, foram menores que as alterações ocorridas nestes parâmetros da classe ω_2. Observe porém que, ao usar-se o *kNN (k=1)* (COVER; HART, 1967) nos mesmos padrões de treinamento, o vetor de teste $\boldsymbol{x}$ é classificado incorretamente, pois

$$d_1 = 0,001812 \text{ e } d_2 = 0,000823\,,$$

ou seja $d_2 < d_1$, sendo d_1 e d_2 as distâncias dos vetores das classes ω_1 e ω_2, respectivamente, mais próximos ao vetor de teste $\boldsymbol{x}$ (Figura 4.1c).

Com o intuito de confirmar estas suspeitas, replicou-se o experimento esboçado através do exemplo dado (Figura 4.1), para 1000 novas instâncias de teste, todas da classe ω_1 (o mesmo comportamento foi obtido ao analisar-se instâncias da classe ω_2) e avaliaram-se as alterações ocorridas em $\hat{\boldsymbol{\mu}}_1$, $\hat{\boldsymbol{\mu}}_2$, $\hat{\Sigma}_1$ e $\hat{\Sigma}_2$.

Figura 4.2: Histogramas das alterações ocorridas em $\hat{\boldsymbol{\mu}}_1$ e $\hat{\boldsymbol{\mu}}_2$.

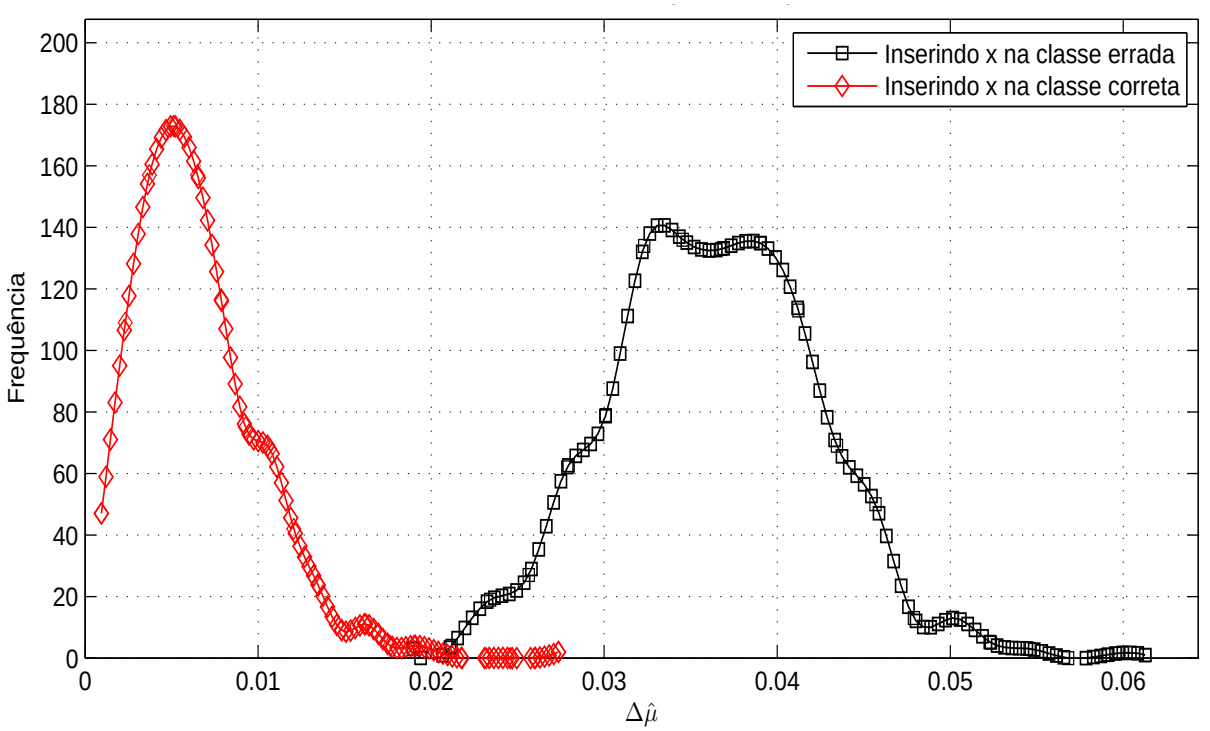
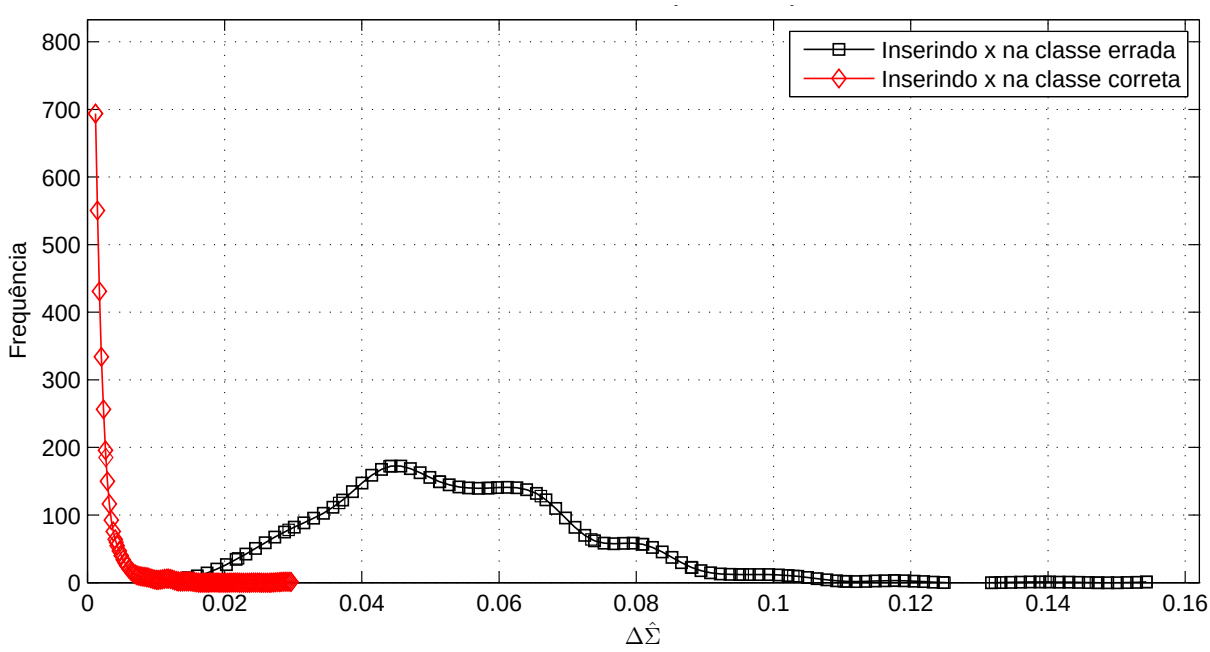


Figura 4.3: Histogramas das alterações ocorridas em $\hat{\Sigma}_1$ e $\hat{\Sigma}_2$.



Durante os experimentos executados coletou-se os valores obtidos para $\Delta\hat{\boldsymbol{\mu}}_1$, $\Delta\hat{\boldsymbol{\mu}}_2$, $\Delta\hat{\Sigma}_1$ e $\Delta\hat{\Sigma}_1$. A partir dos resultados obtidos, construiu-se seus histogramas (Figuras 4.2 e 4.3). Constatou-se nestes histogramas, que em geral, as alterações em $\hat{\boldsymbol{\mu}}_1$ e $\hat{\Sigma}_1$ (inserção na classe correta) concentram-se em torno de valores menores, o que leva a crer que, de fato as alterações em $\hat{\boldsymbol{\mu}}_1$, $\hat{\boldsymbol{\mu}}_2$, $\hat{\Sigma}_1$, e $\hat{\Sigma}_2$, podem ser úteis para propósitos de classificação.

4.2 *Per*turbações usadas para *C*lassificação

Diante do exposto na Subseção 4.1.1, a tarefa de classificar um dado vetor de teste $\boldsymbol{x} \in \mathbb{R}^d$ como pertencente à classe ω_i está, de algum modo, relacionada às perturbações do vetor médio $\hat{\boldsymbol{\mu}}_i$ e da matriz de covariância $\hat{\Sigma}_i$, ocorridas em decorrência da inserção de $\boldsymbol{x}$ nesta classe, quando comparadas com perturbações ocorridas com sua inserção nas demais classes.

Propomos que $\boldsymbol{x}$ seja designado à classe ω_j se

$$\Delta f_j = \min_{1 \leq i \leq C} \Delta f_i$$

sendo f_i uma função que é perturbada quando o vetor de teste $\boldsymbol{x}$ é inserido da classe ω_i. Especificamente, a função f_i pode ter uma das seguintes formas:

- $f_i = f(\hat{\boldsymbol{\mu}}_i)$: a distribuição da classe ω_i descreve-se como dependente apenas do vetor médio $\hat{\boldsymbol{\mu}}_i$;
- $f_i = g(\hat{\Sigma}_i)$: a distribuição da classe ω_i descreve-se como depende apenas da matriz de covariância $\hat{\Sigma}_i$ ou
- $f_i = h(\hat{\boldsymbol{\mu}}_i, \hat{\Sigma}_i)$: quando a distribuição da classe ω_i depende de ambos parâmetros $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$.

No primeiro caso, $f_i = f(\hat{\boldsymbol{\mu}}_i)$, as perturbações $\Delta\hat{\boldsymbol{\mu}}_i$ influem diretamente nas perturbações de f_i enquanto, $\Delta\hat{\Sigma}_i$ afetam as perturbações em $f_i = g(\hat{\Sigma}_i)$, no segundo caso. No último caso, ambas as perturbações $\Delta\hat{\boldsymbol{\mu}}_i$ e $\Delta\hat{\Sigma}_i$ atuam de forma combinada para a perturbação em f_i.

Algorithm 1 PerC Training

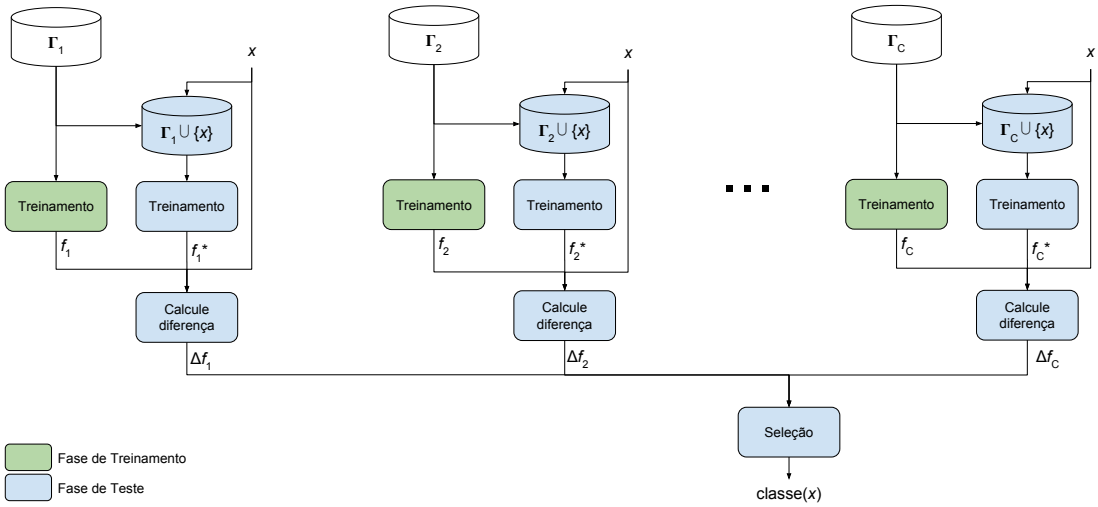
```
1: procedure PERC-TRAIN(Γ)
2:     for all Γ_i ⊂ Γ do                ▷ (All instances from ω_i class)
3:         μ̂_i ← Mean(Γ_i)
4:         Σ̂_i ← CovarianceMatrix(Γ_i)
5:         f_i ← EvaluateFrom(μ̂_i, Σ̂_i)
6:     end for
7: return f_i
8: end procedure
```

Algorithm 2 PerC Test

1: **procedure** PERC-TEST($\boldsymbol{x}, \Gamma, f_i$)
2: **for all** $\Gamma_i \subset \Gamma$ **do**
3: $\hat{\boldsymbol{\mu}}_i^* \leftarrow Mean(\Gamma_i \cup \{\boldsymbol{x}\})$
4: $\hat{\Sigma}_i^* \leftarrow CovarianceMatrix(\Gamma_i \cup \{\boldsymbol{x}\})$
5: $f_i^* \leftarrow EvaluateFrom(\hat{\boldsymbol{\mu}}_i^*, \hat{\Sigma}_i^*)$
6: $\Delta f_i \leftarrow CalculateDifference(f_i, f_i^*)$
7: **end for**
8: $Class(\boldsymbol{x}) = \arg\min_i(\Delta f_i)$ ▷ (Selection Module)
9: **return** $Class(\boldsymbol{x})$
10: **end procedure**

A Figura 4.4 exibe o diagrama de blocos para o classificador proposto: **Per**turbation based **C**lassifier (*PerC*). Recebe como entrada um padrão de teste $\boldsymbol{x} \in \mathbb{R}^d$ e os dados de treinamento $\Gamma \in \mathbb{R}^{d \times N}$ ($\Gamma = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_C\}$) sendo d o número de atributos, N o número de amostras de treinamento e C o número de classes. A saída do *PerC* é a classe do padrão de teste $\boldsymbol{x}$. O classificador proposto é composto por duas fases: treinamento e teste. Estas fases são também descritas nos Algoritmos 1 e 2, respectivamente.

Figura 4.4: Diagrama de Blocos - Algoritmo *PerC*



Na fase de treinamento, cada classificador f_i é treinado usando os dados de treinamento Γ_i, de modo que um total de C classificadores são treinados, um para cada classe. Seguindo o classificador de Bayes, o treinamento de f_i consiste em calcular a média $\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ e a matriz de covariância $\hat{\Sigma}_i$ (Linhas 3-5 no Algoritmo 1).

Na fase de teste, a função f_i^* é aprendida usando o novo banco de dados que é formado pela adição do padrão de teste $\boldsymbol{x}$ ao conjunto Γ_i. Assim, cada f_i^* é dado por um par $(\hat{\boldsymbol{\mu}}_i^*, \hat{\Sigma}_i^*)$ que é calculado usando os dados $\Gamma_i \cup \{\boldsymbol{x}\}$ (Linhas 3-5 no Algoritmo 2). Após o módulo "Calcule

diferença" calcular Δf_i que representa o quanto a classe ω_i foi perturbada pela inserção do padrão de teste $\boldsymbol{x}$ na classe ω_i(Linha 6 no Algoritmo 2). A classe do padrão de teste $\boldsymbol{x}$ é determinada pelo módulo "Seleção" que seleciona a classe ω_i que sofre a menor perturbação sofrida, em outras palavras, o menor Δf_i (Linha 8 no Algoritmo 2).

A Subsection 4.2.1 exibe um modo eficiente de calcular as perturbações que não requer armazenar o conjunto de treinamento Γ (observe que Γ é uma das entradas para o Algoritmo 2). Deduzimos equações que calculam $\Delta\hat{\boldsymbol{\mu}}_i$ e $\Delta\hat{\Sigma}_i$ sem calcular explicitamente $\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}^*$ e $\hat{\Sigma}_i^*$ (Eqs. (4.11) e (4.20)).

A sexta linha no Algoritmo 2 calcula a diferença entre f_i e f_i^*. Esta diferença pode ser calculada usando-se apenas a perturbação da média $\Delta\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$, apenas a perturbação da matriz de covariância $\Delta\hat{\Sigma}_i$ ou uma combinação de ambas. A Subseção 4.2.3 mostra um método que combina $\Delta\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ e $\Delta\hat{\Sigma}_i$.

É importante destacar que numa classificação baseada nesta abordagem, apesar da semelhança, não está sendo realizado, o que se conhece como *incremental* ou *online learning* (ADE; DESHMUKH, 2013; KIVINEN; SMOLA; WILLIAMSON, 2004; LÜTZ; RODNER; DENZLER, 2013; LÜTZ; RODNER; DENZLER, 2011), uma vez que apenas simula-se o que ocorreria com cada uma das classes caso o vetor de teste $\boldsymbol{x}$ fosse inserido nela, para com isso calcular-se as perturbações $\Delta\hat{\boldsymbol{\mu}}_i$ e $\Delta\hat{\Sigma}_i$ e finalmente Δf_i. Não há portanto, alteração dos valores de $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ em nenhuma das classes durante ou após o processo de classificação.

4.2.1 Calculando $\Delta\hat{\boldsymbol{\mu}}_i$ e $\Delta\hat{\Sigma}_i$

A inserção do vetor $\boldsymbol{x}$ na classe ω_i, faz com que o vetor médio desta classe torne-se

$$\hat{\boldsymbol{\mu}}_i^* = \frac{\boldsymbol{x}_1 + \ldots + \boldsymbol{x}_{N_i} + \boldsymbol{x}}{N_i + 1} \tag{4.7}$$

Observe porém, que

$$\hat{\boldsymbol{\mu}}_i = \frac{\boldsymbol{x}_1 + \ldots + \boldsymbol{x}_{N_i}}{N_i} \tag{4.8}$$

Ou seja,

$$\boldsymbol{x}_1 + \ldots + \boldsymbol{x}_{N_i} = \hat{\boldsymbol{\mu}}_i N_i \tag{4.9}$$

e, substituindo a Eq. (4.9) na Eq. (4.7), obtêm-se

$$\hat{\boldsymbol{\mu}}_i^* = \frac{\hat{\boldsymbol{\mu}}_i N_i + \boldsymbol{x}}{N_i + 1} \tag{4.10}$$

e a alteração ocorrida neste parâmetro, será portanto

$$\begin{aligned}\Delta\hat{\boldsymbol{\mu}}_i &= \hat{\boldsymbol{\mu}}_i^* - \hat{\boldsymbol{\mu}}_i \\ &= \frac{\hat{\boldsymbol{\mu}}_i N_i + \boldsymbol{x}}{N_i+1} - \hat{\boldsymbol{\mu}}_i \\ &= \frac{\hat{\boldsymbol{\mu}}_i N_i + \boldsymbol{x} - (N_i+1)\hat{\boldsymbol{\mu}}_i}{N_i+1} \\ &= \frac{\hat{\boldsymbol{\mu}}_i N_i + \boldsymbol{x} - \hat{\boldsymbol{\mu}}_i N_i - \hat{\boldsymbol{\mu}}_i}{N_i+1}\end{aligned}$$

$$\Delta\hat{\boldsymbol{\mu}}_i = \frac{1}{N_i+1}\left(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i\right) \tag{4.11}$$

Do mesmo modo, a matriz de covariância $\hat{\Sigma}_i$, também sofrerá alterações e tornar-se-á

$$\hat{\Sigma}_i^* = \frac{1}{N_i}\left[\sum_{j=1}^{N_i}(\boldsymbol{x}_{i,j} - \hat{\boldsymbol{\mu}}_i^*)(\boldsymbol{x}_{i,j} - \hat{\boldsymbol{\mu}}_i^*)^T + (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^*)(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^*)^T\right] \tag{4.12}$$

Para que observe-se apenas as alterações ocorridas, perceba que

$$\begin{aligned}\hat{\Sigma}_i^* &= \frac{1}{N_i}\left[\sum_{j=1}^{N_i}\left[(\boldsymbol{x}_{i,j} - \hat{\boldsymbol{\mu}}_i) - \left(\hat{\boldsymbol{\mu}}_i^* - \hat{\boldsymbol{\mu}}_i\right)\right]\cdot\left[(\boldsymbol{x}_{i,j} - \hat{\boldsymbol{\mu}}_i) - \left(\hat{\boldsymbol{\mu}}_i^* - \hat{\boldsymbol{\mu}}_i\right)\right]^T + (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^*)(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^*)^T\right] \\ &= \left[\frac{1}{N_i}\sum_{j=1}^{N_i}\left[(\boldsymbol{x}_{i,j} - \hat{\boldsymbol{\mu}}_i) - \left(\hat{\boldsymbol{\mu}}_i^* - \hat{\boldsymbol{\mu}}_i\right)\right]\cdot\left[(\boldsymbol{x}_{i,j} - \hat{\boldsymbol{\mu}}_i) - \left(\hat{\boldsymbol{\mu}}_i^* - \hat{\boldsymbol{\mu}}_i\right)\right]^T\right] + \frac{1}{N_i}(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^*)(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^*)^T \\ &= \frac{1}{N_i}\sum_{j=1}^{N_i}\Big[(\boldsymbol{x}_{i,j} - \hat{\boldsymbol{\mu}}_i)(\boldsymbol{x}_{i,j} - \hat{\boldsymbol{\mu}}_i)^T - (\boldsymbol{x}_{i,j} - \hat{\boldsymbol{\mu}}_i)\left(\hat{\boldsymbol{\mu}}_i^* - \hat{\boldsymbol{\mu}}_i\right)^T - \left(\hat{\boldsymbol{\mu}}_i^* - \hat{\boldsymbol{\mu}}_i\right)(\boldsymbol{x}_{i,j} - \hat{\boldsymbol{\mu}}_i)^T + \\ &\quad + \left(\hat{\boldsymbol{\mu}}_i^* - \hat{\boldsymbol{\mu}}_i\right)\left(\hat{\boldsymbol{\mu}}_i^* - \hat{\boldsymbol{\mu}}_i\right)^T\Big] + \frac{1}{N_i}(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i)(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^*)^T \\ &= \frac{1}{N_i}\sum_{j=1}^{N_i}(\boldsymbol{x}_{i,j} - \hat{\boldsymbol{\mu}}_i)(\boldsymbol{x}_{i,j} - \hat{\boldsymbol{\mu}}_i)^T - \frac{1}{N_i}\sum_{j=1}^{N_i}(\boldsymbol{x}_{i,j} - \hat{\boldsymbol{\mu}}_i)\left(\hat{\boldsymbol{\mu}}_i^* - \hat{\boldsymbol{\mu}}_i\right)^T - \\ &\quad - \frac{1}{N_i}\sum_{j=1}^{N_i}\left(\hat{\boldsymbol{\mu}}_i^* - \hat{\boldsymbol{\mu}}_i\right)(\boldsymbol{x}_{i,j} - \hat{\boldsymbol{\mu}}_i)^T + \frac{1}{N_i}\sum_{j=1}^{N_i}\left(\hat{\boldsymbol{\mu}}_i^* - \hat{\boldsymbol{\mu}}_i\right)\left(\hat{\boldsymbol{\mu}}_i^* - \hat{\boldsymbol{\mu}}_i\right)^T + \frac{1}{N_i}(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^*)(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^*)^T\end{aligned}$$

$$\hat{\Sigma}_i^* = \frac{N_i-1}{N_i}\hat{\Sigma}_i - \frac{1}{N_i}\sum_{j=1}^{N_i}(\boldsymbol{x}_{i,j}-\hat{\boldsymbol{\mu}}_i)\left(\hat{\boldsymbol{\mu}}_i^*-\hat{\boldsymbol{\mu}}_i\right)^T - \frac{1}{N_i}\sum_{j=1}^{N_i}\left(\hat{\boldsymbol{\mu}}_i^*-\hat{\boldsymbol{\mu}}_i\right)(\boldsymbol{x}_{i,j}-\hat{\boldsymbol{\mu}}_i)^T +$$
$$+\left(\hat{\boldsymbol{\mu}}_i^*-\hat{\boldsymbol{\mu}}_i\right)\left(\hat{\boldsymbol{\mu}}_i^*-\hat{\boldsymbol{\mu}}_i\right)^T + \frac{1}{N_i}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i^*)(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i^*)^T \tag{4.13}$$

Usando a Eq. (4.10), deduz-se

$$\hat{\boldsymbol{\mu}}_i = \frac{(N_i+1)\hat{\boldsymbol{\mu}}_i^* - \boldsymbol{x}}{N_i} \tag{4.14}$$

e disto segue-se que

$$\left(\hat{\boldsymbol{\mu}}_i^*-\hat{\boldsymbol{\mu}}_i\right)\left(\hat{\boldsymbol{\mu}}_i^*-\hat{\boldsymbol{\mu}}_i\right)^T = \left(\frac{\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i^*}{N_i}\right)\left(\frac{\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i^*}{N_i}\right)^T$$
$$= \frac{1}{N_i^2}\left(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i^*\right)\left(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i^*\right)^T \tag{4.15}$$

Tem-se também, que

$$\sum_{j=1}^{N_i}(\boldsymbol{x}_{i,j}-\hat{\boldsymbol{\mu}}_i)\left(\hat{\boldsymbol{\mu}}_i^*-\hat{\boldsymbol{\mu}}_i\right)^T + \left(\hat{\boldsymbol{\mu}}_i^*-\hat{\boldsymbol{\mu}}_i\right)(\boldsymbol{x}_{i,j}-\hat{\boldsymbol{\mu}}_i)^T = \tag{4.16}$$

$$\left(\hat{\boldsymbol{\mu}}_i^*-\hat{\boldsymbol{\mu}}_i\right)^T\underbrace{\sum_{j=1}^{N_i}(\boldsymbol{x}_{i,j}-\hat{\boldsymbol{\mu}}_i)}_{0} + \left(\hat{\boldsymbol{\mu}}_i^*-\hat{\boldsymbol{\mu}}_i\right)\underbrace{\sum_{j=1}^{N_i}(\boldsymbol{x}_{i,j}-\hat{\boldsymbol{\mu}}_i)^T}_{0} = 0$$

Substituindo os resultados obtidos nas Eqs. (4.15) e (4.16) na expressão obtida para $\hat{\Sigma}_i^*$, Eq. (4.13), tem-se

$$\hat{\Sigma}_i^* = \frac{N_i-1}{N_i}\hat{\Sigma}_i + \frac{1}{N_i^2}\left(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i^*\right)\left(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i^*\right)^T + \frac{1}{N_i}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i^*)(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i^*)^T$$

$$= \frac{N_i-1}{N_i}\hat{\Sigma}_i + \frac{N_i+1}{N_i^2}\left(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i^*\right)\left(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i^*\right)^T \tag{4.17}$$

e em consequência,

$$\begin{aligned}\Delta\hat{\Sigma}_i &= \hat{\Sigma}_i^* - \hat{\Sigma}_i \\ &= \frac{N_i - 1}{N_i}\hat{\Sigma}_i + \frac{N_i + 1}{N_i^2}\left(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^*\right)\left(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^*\right)^T - \hat{\Sigma}_i \\ &= -\frac{1}{N_i}\hat{\Sigma}_i + \frac{N_i + 1}{N_i^2}\left(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^*\right)\left(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^*\right)^T \end{aligned} \tag{4.18}$$

porém,

$$\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^* = \boldsymbol{x} - \frac{\hat{\boldsymbol{\mu}}_i N_i + \boldsymbol{x}}{N_i + 1} = \frac{N_i}{N_i + 1}\left(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i\right) \tag{4.19}$$

ou seja

$$\begin{aligned}\Delta\hat{\Sigma}_i &= \hat{\Sigma}_i^* - \hat{\Sigma}_i \\ &= -\frac{1}{N_i}\hat{\Sigma}_i + \frac{N_i + 1}{N_i^2}\left(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^*\right)\left(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i^*\right)^T \\ &= -\frac{1}{N_i}\hat{\Sigma}_i + \frac{1}{N_i + 1}\left(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i\right)\left(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i\right)^T \end{aligned} \tag{4.20}$$

Observe que as Eqs. (4.11) e (4.20) apresentam-se como meios eficientes de se calcular as alterações $\Delta\hat{\boldsymbol{\mu}}_i$ e $\Delta\hat{\Sigma}_i$, uma vez que podem ser calculadas usando-se apenas os valores de $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$, além do próprio vetor de teste $\boldsymbol{x}$. Tornando desnecessário, portanto, o cálculo de $\hat{\boldsymbol{\mu}}_i^*$ e $\hat{\Sigma}_i^*$.

4.2.2 Classificadores Simples baseados em $\Delta\hat{\boldsymbol{\mu}}_i$ ou $\Delta\hat{\Sigma}_i$

A partir do método proposto, foram construídos dois classificadores, um que depende exclusivamente de $\Delta\hat{\boldsymbol{\mu}}_i$ e outro de $\Delta\hat{\Sigma}_i$. Ou seja, tomou-se

$$f_i = f(\hat{\boldsymbol{\mu}}_i) = \hat{\boldsymbol{\mu}}_i \tag{4.21}$$

$$f_i = g(\hat{\Sigma}_i) = \hat{\Sigma}_i \tag{4.22}$$

De agora em diante, será chamado de *PerC(Mean)* o classificador baseado em (4.21) de *PerC(Cov)* o classificador baseado em (4.22). Disto segue-se que, a Linha 6 do Algoritmo 2, torna-se

$$\Delta f_i = \|\Delta\hat{\boldsymbol{\mu}}_i\|$$

para o *PerC(Mean)*, e

$$\Delta f_i = \left\|\Delta\hat{\Sigma}_i\right\|$$

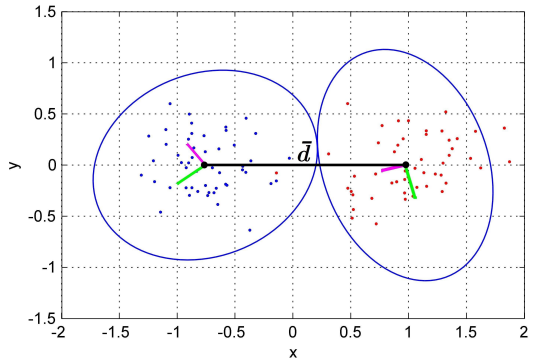
para o *PerC(Cov)*.

Como experimento inicial, utilizou-se bancos de dados bidimensionais, com *distribuição Gaussiana* e 2 classes, gerados artificialmente a partir de matrizes de covariância Σ_1 e Σ_2 e de vetores médios $\boldsymbol{\mu}_1$ e $\boldsymbol{\mu}_2$, para cada uma das classes.

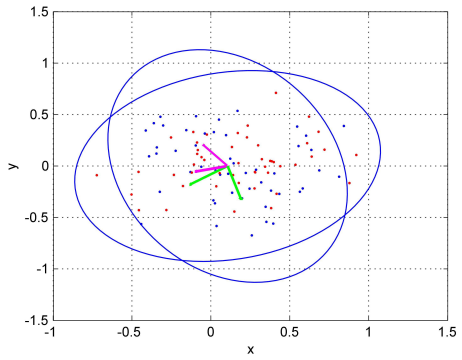
As matrizes Σ_1 e Σ_2 são positivas definidas e geradas aleatoriamente. Suas elipses de confiança são determinadas com 99% de certeza, usando o método apresentado em (WU; XIAO, 2012), e para estas elipses estimou-se uma distância mínima $\bar{d}$ entre os vetores médios $\boldsymbol{\mu}_1$ e $\boldsymbol{\mu}_2$, de tal modo que os dados não possuam interseção entre si (Fig. 4.5a).

Figura 4.5: Banco de dados gaussiano, 2D com 2 classes

(a) distância $\bar{d}$ entre $\boldsymbol{\mu}_1$ e $\boldsymbol{\mu}_2$.



(b) distância 0 entre $\boldsymbol{\mu}_1$ e $\boldsymbol{\mu}_2$.



Em seguida, gradativamente reduziu-se a distância $\bar{d}$, colocando os vetores médios em

$$\boldsymbol{\mu}_1 = (\bar{d}_1, 0),\ \boldsymbol{\mu}_2 = (\bar{d}_2, 0)$$

sendo que

$$\bar{d}_2 - \bar{d}_1 = \bar{d}$$

Assim, à medida que $\bar{d} \to 0$, os valores de $\boldsymbol{\mu}_1$ e $\boldsymbol{\mu}_2$ aproximam-se até que coincidem (Fig. 4.5b). A cada alteração de $\bar{d}$, um novo banco de dados é gerado a partir de $(\Sigma_1, \boldsymbol{\mu}_1)$ e $(\Sigma_2, \boldsymbol{\mu}_2)$ e a classificação é realizada 100 vezes, através dos classificadores propostos, *PerC(Mean)*, Eq. (4.21) e *PerC(Cov)*, Eq. (4.22)), do *kNN* (COVER; HART, 1967), em suas versões para *k=1,3* e *5*, do *Naïve Bayes* (DOMINGOS; PAZZANI, 1997; HAND; YU, 2001), do *SVM*, em suas versões para *kernels polinomiais de graus 1, 2, 3* e o *kernel gaussiano*, das *Árvores de Classificação e Regressão* (do inglês, *CART*) (BREIMAN et al., 1984) e das *Redes Neurais Multicamadas* (em inglês, *MLP*) (BISHOP, 1994), utilizando o *10-fold cross-validation*. Os mesmos experimentos foram repetidos para diferentes quantidades de instâncias no banco de dados (Figuras 4.6, 4.7, 4.8, 4.9 e 4.10). Em relação ao *kNN* e *SVM*, estão expostos nas figuras, apenas as suas versões com melhores resultados em relação aos demais classificadores.

Figura 4.6: Distância × Taxa de Acerto, Com 20 pontos em cada classe

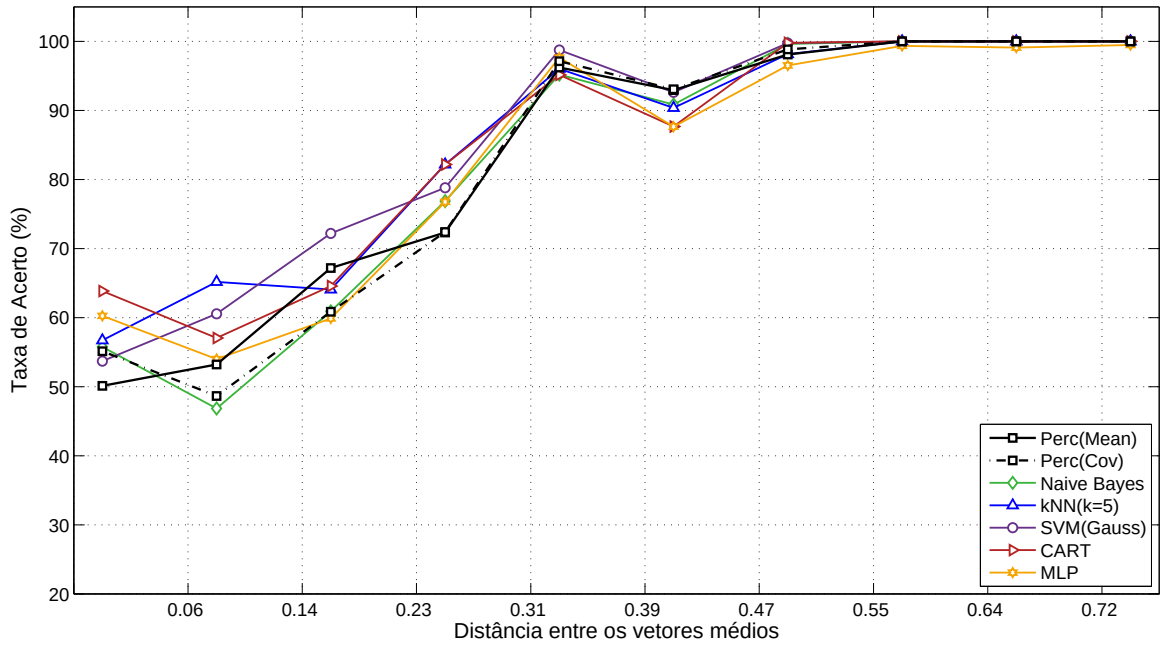


Figura 4.7: Distância × Taxa de Acerto, Com 100 pontos em cada classe

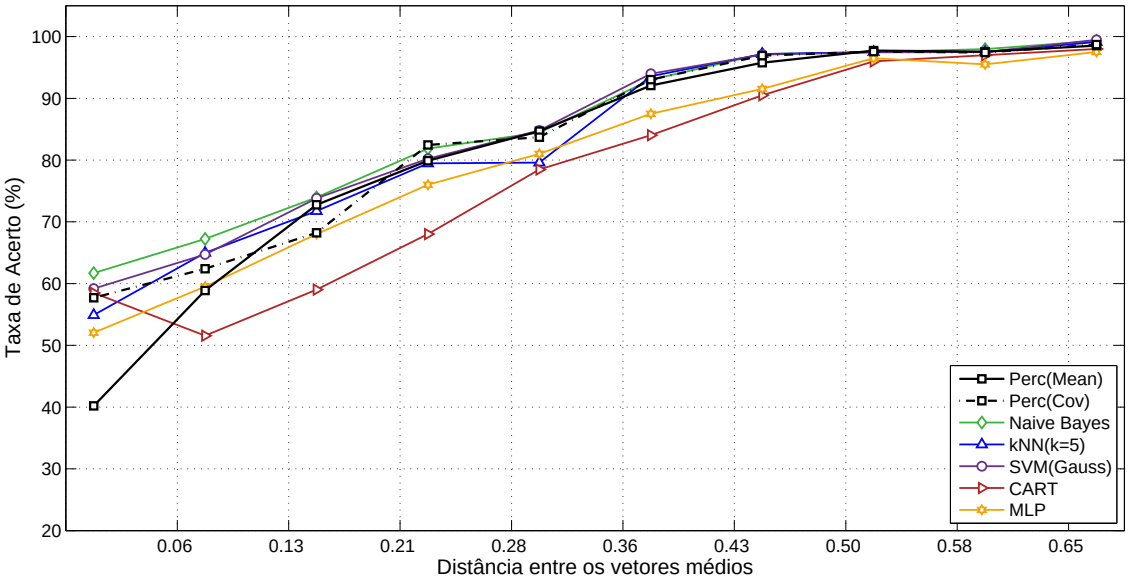


Figura 4.8: Distância × Taxa de Acerto, Com 300 pontos em cada classe

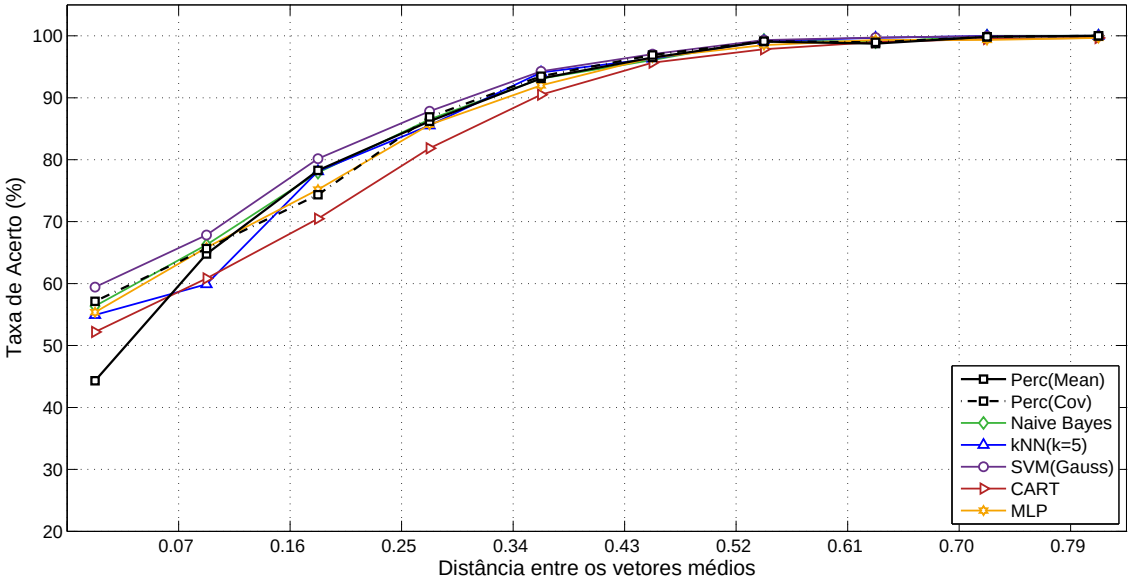


Figura 4.9: Distância × Taxa de Acerto, Com 500 pontos em cada classe

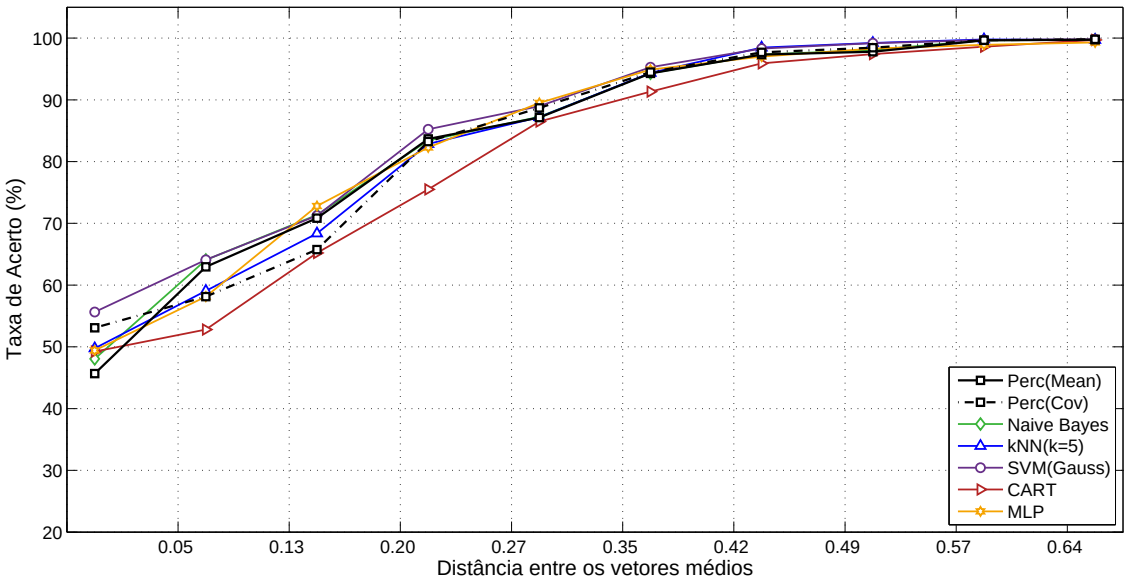
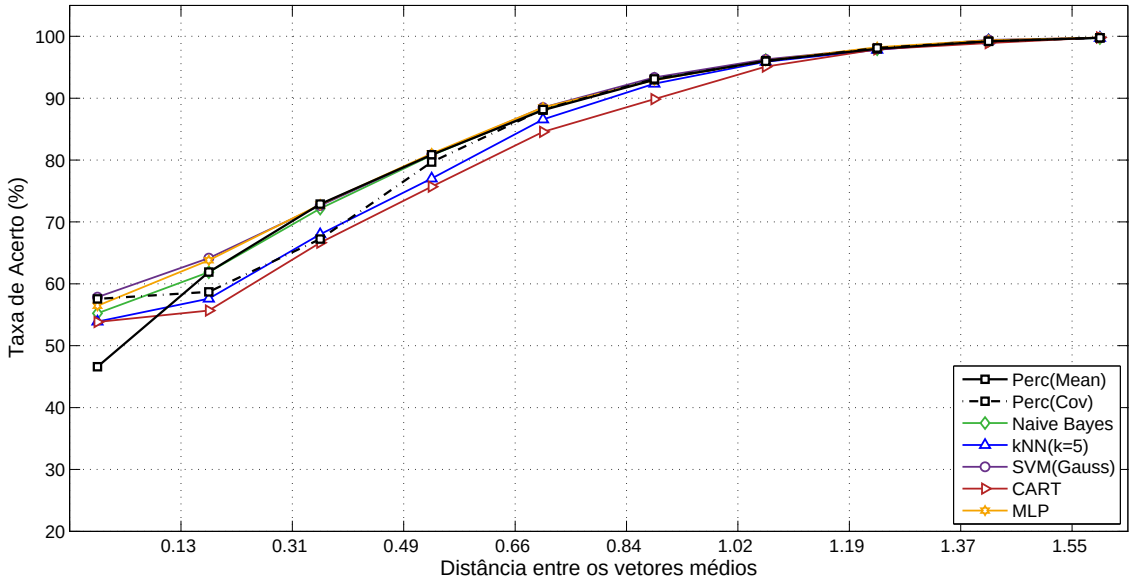


Figura 4.10: Distância × Taxa de Acerto, Com 1000 pontos em cada classe



De acordo com os resultados obtidos, os classificadores propostos, *PerC(Mean)* e *PerC(Cov)*, baseados exclusivamente nas alterações em $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$, respectivamente, apresentam comportamento semelhante, em relação às taxas de acerto médio, aos dos demais classificadores. Entretanto, encontram-se entre os de maior taxa de acerto na maioria dos casos, sendo exceção apenas as situações em que há poucas amostas de treinamento além de uma separabilidade difícil entre as classes (distância pequena entre os vetores médios $\hat{\boldsymbol{\mu}}_\mathbf{1}$ e $\hat{\boldsymbol{\mu}}_\mathbf{2}$) (Figuras 4.6 e 4.7). Observa-se que nestas situações o classificador *PerC(Cov)* possui performance sempre superior ao *PerC(Mean)*, indicando que as perturbações na matriz de covariância possuem maior poder de discriminação que as perturbações do vetor médio, nestes casos. Quando há uma maior quantidade de dados de treinamento, a semelhança com os classificadores comparados, torna-se ainda mais evidente (Figuras 4.8, 4.9 e 4.10).

4.2.3 Classificador Combinado

Como visto anteriormente, as alterações em $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ influenciam significativamente na classificação do vetor de teste $\boldsymbol{x}$. Nesta seção investiga-se a possibilidade de combinar tais alterações em um único e mais poderoso classificador. Para isto, admita que a função d'_i (Eq. 3.24) dependa também de $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$, ou seja

$$d'_i(\boldsymbol{x}, \hat{\boldsymbol{\mu}}_i, \hat{\Sigma}_i) = \ln\left|\hat{\Sigma}_i\right| + (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i)^T \hat{\Sigma}_i^{-1} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i) - 2\ln P(\omega_i) \qquad (4.23)$$

Pequenas alterações em $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ acarretarão pequenas alterações em d'_i. Do cálculo diferencial, tem-se que

$$\partial d'_i = \frac{\partial d'_i}{\partial \boldsymbol{x}} \cdot d\boldsymbol{x} + \frac{\partial d'_i}{\partial \hat{\boldsymbol{\mu}}_i} \cdot d\hat{\boldsymbol{\mu}}_i + \frac{\partial d'_i}{\partial \hat{\Sigma}_i} \cdot d\hat{\Sigma}_i$$

e disto segue-se que

$$\Delta d'_i \cong \frac{\partial d'_i}{\partial \boldsymbol{x}} \cdot \Delta \boldsymbol{x} + \frac{\partial d'_i}{\partial \hat{\boldsymbol{\mu}}_i} \cdot \Delta \hat{\boldsymbol{\mu}}_i + \frac{\partial d'_i}{\partial \hat{\Sigma}_i} \cdot \Delta \hat{\Sigma}_i$$

Neste caso, como o vetor de teste é o mesmo, $\Delta \boldsymbol{x} = 0$, ou seja

$$\Delta d'_i \cong \frac{\partial d'_i}{\partial \hat{\boldsymbol{\mu}}_i} \cdot \Delta \hat{\boldsymbol{\mu}}_i + \frac{\partial d'_i}{\partial \hat{\Sigma}_i} \cdot \Delta \hat{\Sigma}_i \tag{4.24}$$

e de acordo com (FUKUNAGA, 1972) (Eq. A.39, pg. 569), esta expressão pode ser reescrita como

$$\Delta d'_i \cong \left(\frac{\partial d'_i}{\partial \hat{\boldsymbol{\mu}}_i} \right)^T \Delta \hat{\boldsymbol{\mu}}_i + \mathrm{tr} \left\{ \frac{\partial d'^*_i}{\partial \hat{\Sigma}_i} \Delta \hat{\Sigma}_i \right\} \tag{4.25}$$

sendo

$$\frac{\partial d'^*_i}{\partial \hat{\Sigma}_i} = \frac{1}{2} \left\{ \frac{\partial d'_i}{\partial \hat{\Sigma}_i} + \mathrm{diag} \left[\frac{\partial d'_i}{\partial \hat{\Sigma}_i} \right] \right\} \tag{4.26}$$

Observe, portanto que

$$\begin{aligned}
\frac{\partial d'_i}{\partial \hat{\boldsymbol{\mu}}_i} &= \frac{\partial}{\partial \hat{\boldsymbol{\mu}}_i} \left[\ln \left| \hat{\Sigma}_i \right| + (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i)^T \hat{\Sigma}_i^{-1} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i) - 2 \ln P(\omega_i) \right] \\
&= \frac{\partial}{\partial \hat{\boldsymbol{\mu}}_i} \ln \left| \hat{\Sigma}_i \right| + \frac{\partial}{\partial \hat{\boldsymbol{\mu}}_i} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i)^T \hat{\Sigma}_i^{-1} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i) - 2 \frac{\partial}{\partial \hat{\boldsymbol{\mu}}_i} \ln P(\omega_i) \\
&= \frac{\partial}{\partial \hat{\boldsymbol{\mu}}_i} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i)^T \hat{\Sigma}_i^{-1} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i) \\
&= -2 \hat{\Sigma}_i^{-1} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i)
\end{aligned} \tag{4.27}$$

Ver Searle (1982), para a última passagem nesta dedução. Ainda segundo Fukunaga (1972) (Eqs. A.43 e A.44, pg. 570),

$$\begin{aligned}
\frac{\partial d'_i}{\partial \hat{\Sigma}_i} &= \frac{\partial}{\partial \hat{\Sigma}_i} \left[\ln \left| \hat{\Sigma}_i \right| + (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i)^T \hat{\Sigma}_i^{-1} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i) - 2 \ln P(\omega_i) \right] \\
&= \frac{\partial}{\partial \hat{\Sigma}_i} \ln \left| \hat{\Sigma}_i \right| + \frac{\partial}{\partial \hat{\Sigma}_i} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i)^T \hat{\Sigma}_i^{-1} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i) - 2 \frac{\partial}{\partial \hat{\Sigma}_i} \ln P(\omega_i)
\end{aligned}$$

$$\begin{aligned}
\frac{\partial d'_i}{\partial \hat{\Sigma}_i} = 2 \hat{\Sigma}_i^{-1} - \mathrm{diag} \left(\Sigma_i^{-1} \right) - \Big\{ & 2 \hat{\Sigma}_i^{-1} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i)(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i)^T \hat{\Sigma}_i^{-1} - \\
& - \mathrm{diag} \left[\hat{\Sigma}_i^{-1} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i)(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i)^T \hat{\Sigma}_i^{-1} \right] \Big\}
\end{aligned} \tag{4.28}$$

e

$$\begin{aligned}\operatorname{diag}\left(\frac{\partial d_i'}{\partial \hat{\Sigma}_i}\right) &= \operatorname{diag}\Bigg(2\hat{\Sigma}_i^{-1} - \operatorname{diag}\left(\Sigma_i^{-1}\right) - \Big\{2\hat{\Sigma}_i^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)^T\hat{\Sigma}_i^{-1} - \\ &\quad -\operatorname{diag}\left[\hat{\Sigma}_i^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)^T\hat{\Sigma}_i^{-1}\right]\Big\}\Bigg) \\ &= 2\operatorname{diag}\left(\Sigma_i^{-1}\right) - \operatorname{diag}\left(\Sigma_i^{-1}\right) - 2\operatorname{diag}\left[\hat{\Sigma}_i^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)^T\hat{\Sigma}_i^{-1}\right] + \\ &\quad + \operatorname{diag}\left[\hat{\Sigma}_i^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)^T\hat{\Sigma}_i^{-1}\right] \\ &= \operatorname{diag}\left(\Sigma_i^{-1}\right) - \operatorname{diag}\left[\hat{\Sigma}_i^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)^T\hat{\Sigma}_i^{-1}\right] \qquad (4.29)\end{aligned}$$

Substituindo as Eqs. (4.28) e (4.29) na Eq. (4.26) tems-se,

$$\begin{aligned}\frac{\partial d_i'^*}{\partial \hat{\Sigma}_i} &= \frac{1}{2}\Bigg\{2\hat{\Sigma}_i^{-1} - \operatorname{diag}\left(\Sigma_i^{-1}\right) - 2\hat{\Sigma}_i^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)^T\hat{\Sigma}_i^{-1} + \\ &\quad + \operatorname{diag}\left[\hat{\Sigma}_i^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)^T\hat{\Sigma}_i^{-1}\right] + \operatorname{diag}\left(\Sigma_i^{-1}\right) - \\ &\quad -\operatorname{diag}\left[\hat{\Sigma}_i^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)^T\hat{\Sigma}_i^{-1}\right]\Bigg\} \\ &= \frac{1}{2}\left\{2\hat{\Sigma}_i^{-1} - 2\hat{\Sigma}_i^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)^T\hat{\Sigma}_i^{-1}\right\} \\ &= \hat{\Sigma}_i^{-1} - \hat{\Sigma}_i^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)^T\hat{\Sigma}_i^{-1} \qquad (4.30)\end{aligned}$$

Assim, usando os resultados obtidos nas Eqs. (4.27) e (4.30), é possível reescrever a Eq. (4.25) da seguinte forma

$$\Delta d_i' \cong \left[-2\hat{\Sigma}_i^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)\right]^T \Delta\hat{\boldsymbol{\mu}}_i + \operatorname{tr}\left\{\left[\hat{\Sigma}_i^{-1} - \hat{\Sigma}_i^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_i)^T\hat{\Sigma}_i^{-1}\right]\Delta\hat{\Sigma}_i\right\} \qquad (4.31)$$

Perceba que esta equação depende fortemente de $\hat{\Sigma}_i^{-1}$ que, em alguns casos não existe. Para esta nova versão do algoritmo, utilizou-se a pseudo-inversa da matriz $\hat{\Sigma}_i$ que funciona como uma aproximação numérica de $\hat{\Sigma}_i^{-1}$. Apesar disto, a Eq. (4.31) apresenta a vantagem de não depender de $\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}'$ e $\hat{\Sigma}_i'$ e, sendo assim, passível de ser calculada eficientemente.

Fazendo uso da Eq. (4.31) e retornando ao Algoritmo 1, escolha

$$f_i = d_i'(\boldsymbol{x}, \hat{\boldsymbol{\mu}}_i, \hat{\Sigma}_i) \qquad (4.32)$$

donde segue-se que, a Linha 6 do Algoritmo 2, torna-se

$$\Delta f_i = \Delta d'_i$$

de modo que o vetor de teste $\boldsymbol{x}$ será designado à classe ω_j se

$$\left\|\Delta d'_j\right\| = \min_{1\leq i\leq C} \left\|\Delta d'_i\right\| \tag{4.33}$$

A partir de agora, o classificador baseado na escolha da função f_i conforme (4.32), será denominado *PerC(Comb)* ou simplesmente *PerC* quando não houver possibilidade de confusão.

Realizando novamente os experimentos executados na Subseção 4.2.2, percebe-se, diante dos resultados obtidos (Figuras 4.11, 4.12, 4.13, 4.14 e 4.15), que o *PerC(Comb)*, apresenta em geral, melhor performance que as versões baseadas apenas nas perturbações do vetor médio ou da matriz de covariância. Contudo, essa performance agora apresenta taxas de acerto entre as maiores quando comparadas aos classificadores avaliados. O *PerC(Comb)* obteve seu pior resultado, no mais extremo dos cenários avaliados, 20 pontos e distância zero entre os vetores médios das classes (Figura 4.11), sendo superior apenas ao *PerC(Mean)*. Nas situações em que houve maior quantidade de amostras ou maior distanciamento entre as classes, o *PerC(Comb)* sempre esteve entre os classificadores com maior taxa de acerto (Figuras 4.12, 4.13, 4.14 e 4.15).

Figura 4.11: Distância $\times$ Taxa de Acerto, Com 20 pontos em cada classe

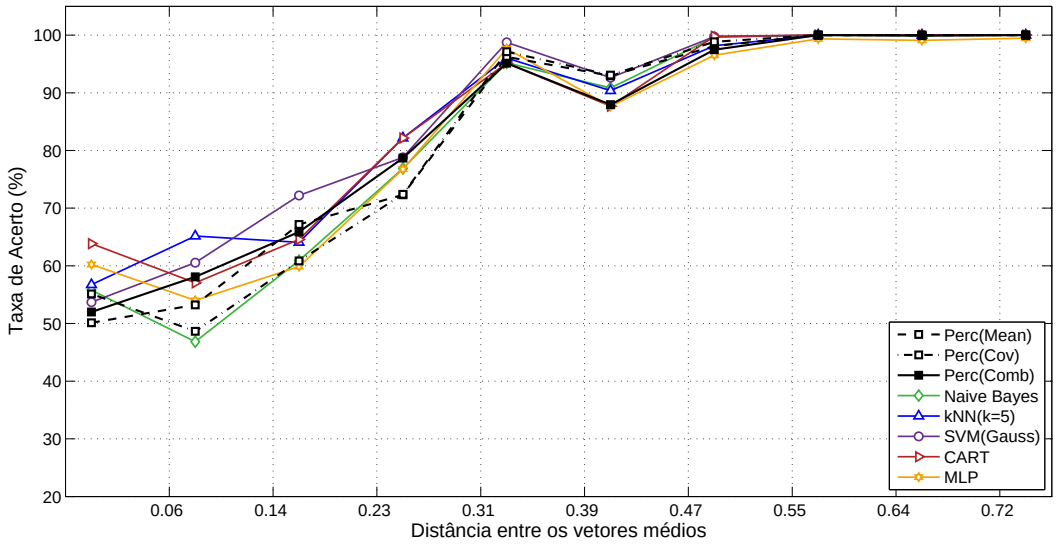


Figura 4.12: Distância × Taxa de Acerto, Com 100 pontos em cada classe

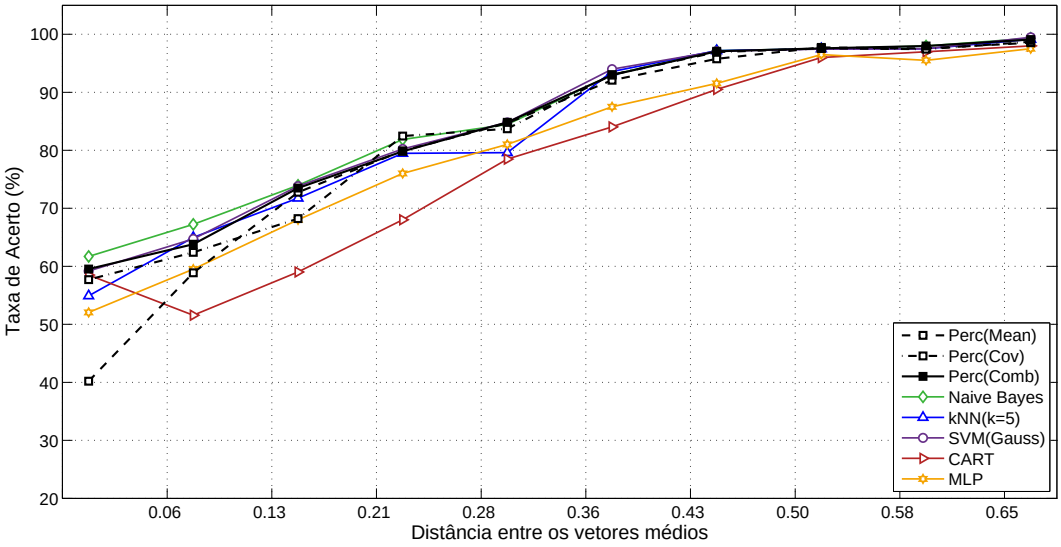


Figura 4.13: Distância × Taxa de Acerto, Com 300 pontos em cada classe

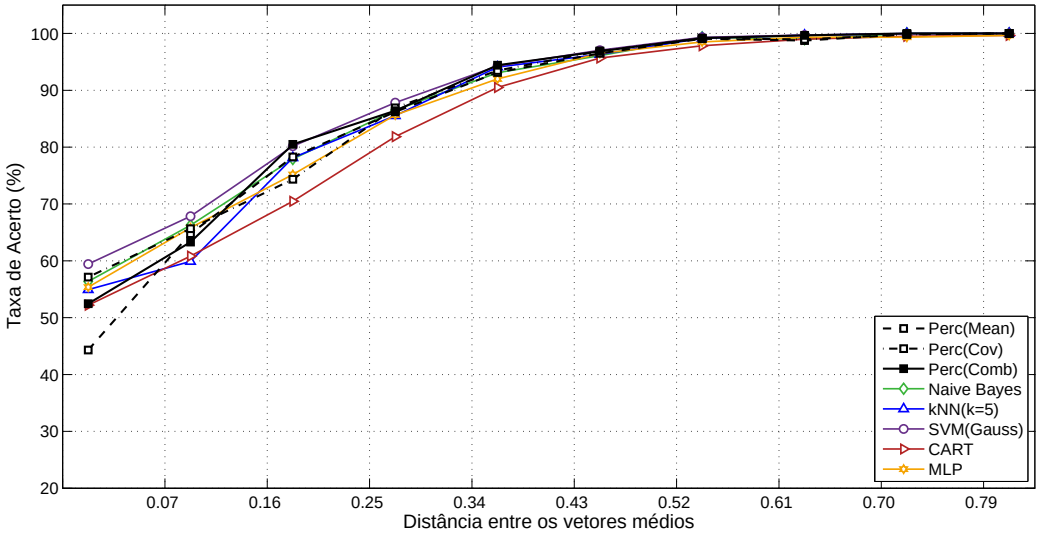


Figura 4.14: Distância × Taxa de Acerto, Com 500 pontos em cada classe

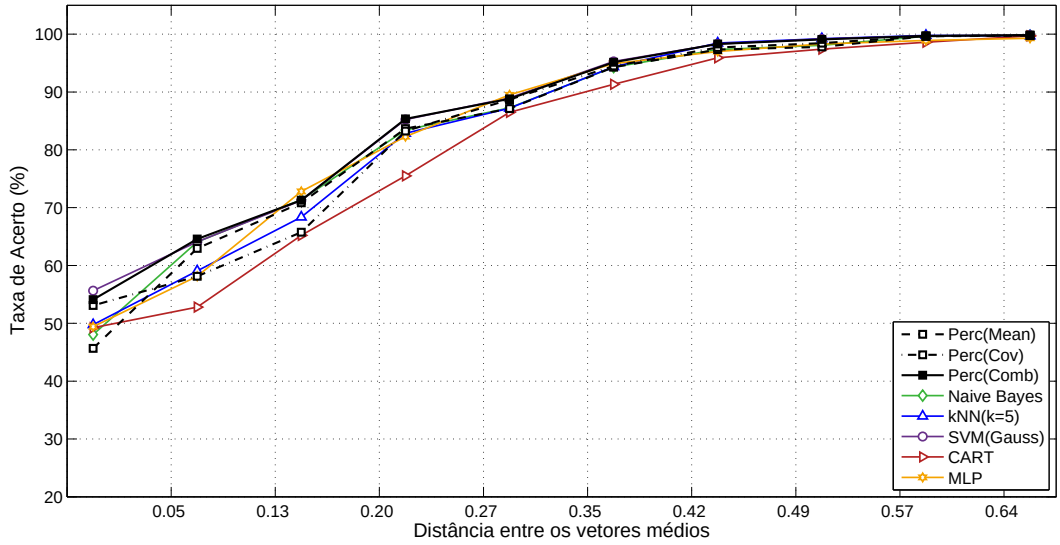
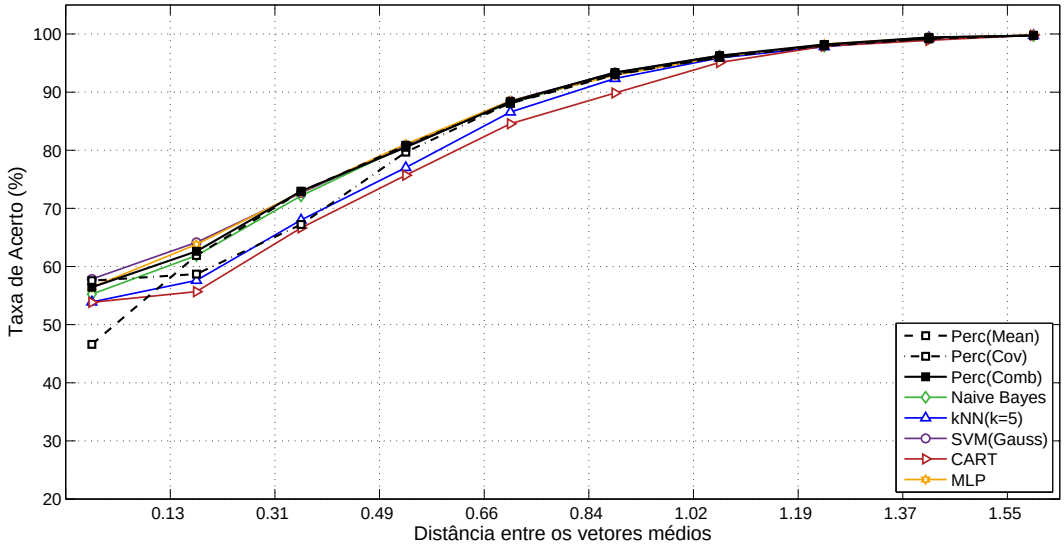


Figura 4.15: Distância × Taxa de Acerto, Com 1000 pontos em cada classe



4.2.4 Análise de Complexidade

Supondo que se tenham n padrões de treinamento com d dimensões em um problema de classificação com C classes e admitindo o *PerC(Comb)* como classificador analisado, segue-se de acordo com os Algoritmos 1 e 2, que para o cálculo dos vetores médios e das matrizes de covariância para cada uma das classes, Linhas 3 e 4 do Algoritmo 1, possuem complexidade de tempo $O(n)$ e $O(d^2n)$, respectivamente.

As perturbações em $\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ e $\hat{\Sigma}_i$, calculadas de acordo com as Equações (4.11) e (4.20), possuem complexidade $O(1)$ e $O(d^2)$, uma vez que utilizam os valores de $\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ e $\hat{\Sigma}_i$ calculados previamente nas Linhas 3 e 4, além do próprio vetor $\boldsymbol{x}$ a ser classificado.

A Linha 6, de acordo com a Equação (4.31), para seu cálculo, além da multiplicação de

matrizes, $O(d^2)$, do traço de matriz, O(d), necessita ainda a inversão da matriz de covariância estimada $\hat{\Sigma}_i$. Usando a pseudo inversa, que possui complexidade $O(d^3)$, esta linha tem $O(d^3)$ como complexidade final.

Assumindo que o algoritmo de ordenação utilizado na Linha 8 do Algoritmo 2 possua complexidade linear, segue-se que o algoritmo como um todo, etapas de treinamento e teste, possui complexidade de tempo $O(d^2n + d^3)$, ou seja, é linear em relação a quantidade de padrões de treinamento e cúbico em relação a dimensão dos padrões.

Em relação à complexidade de armazenamento, é possível calcular os vetores médios e matrizes de covariância sem a necessidade de armazenar todo os padrões de treinamento na memória a máquina. Deste modo o maior armazenamento necessário para a execução do *PerC(Comb)* são as C matrizes de covariância. Portanto a complexidade de armazenamento é da ordem de $O(d^2)$. Caso seja necessário armazenar todos os padrões na memória, cuja complexidade de armazenamento é $O(n)$. O algoritmo resulta em complexidade de armazenamento $O(n + d^2)$. Sendo portando, linear quanto ao número de padrões de treinamento e quadrático em relação à dimensão.

4.3 Considerações

Neste capítulo, propõe-se as perturbações $\Delta\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ e $\Delta\hat{\Sigma}_i$ como parâmetros úteis para a classificação supervisionada. Inicialmente avaliou-se a validade desta proposição de forma simples por meio de um exemplo e posteriormente através da extensão do mesmo para uma escala maior. Os resultados foram comparados com classificadores reconhecidos na área e com isto confirmou-se a validade da abordagem proposta. Um classificador mais eficiente, denominado *PerC(Comb)*, explorando o poder de discriminação de ambas perturbações, foi desenvolvido e comparado com os mesmos classificadores anteriormente utilizados. Por fim, avaliou-se a complexidade de tempo e armazenamento do algoritmo *PerC(Comb)* proposto.

No capítulo 5, com o intuito de avaliar a influência do número de classes na performance do *PerC(Comb)*, será ampliado o espectro do experimento para dados sintéticos gaussianos com maior número de classes. Em seguida, para avaliar a importância da distribuição dos dados, experimentos serão realizados com dados sintéticos porém, não gaussianos. Encerrando os experimentos, 21 bancos de dados reais obtidos do *UCI Repository Learning* (BACHE; LICHMAN, 2013) serão usados e os resultados comparados com os classificadores utilizados até aqui.

5

RESULTADOS

Science never solves a problem without creating ten more,

—GEORGE BERNARD SHAW

5.1 Introdução

Neste capítulo, a partir da Seção 5.2 amplia-se o escopo dos experimentos realizados nas Seções 4.2.2 e 4.2.3, incluindo agora dados sintéticos gaussianos com 3, 4 e 5 classes. Usando-se basicamente a mesma metodologia, explora-se qual a influência da quantidade de classes presentes nos dados, na abordagem proposta. Na Seção 5.3 investiga-se qual a importância da normalidade dos dados, através de experimentos seguindo a mesma metodologia sobre dados sintéticos não gaussianos.

Na Seção 5.4 são realizados experimentos em bases de dados reais disponibilizados no *UCI Repository Learning* (BACHE; LICHMAN, 2013). Novamente a classificação foi repetida por 100 vezes e sua validade foi verificada usando-se o *10-fold cross validation*. A partir dos resultados obtidos, um comparativo entre os classificadores *kNN(1,3,5)*, *Naïve Bayes*, *SVM* com *kernels* polinomiais de graus 1, 2 e 3 e *kernel* gaussiano, *CART* e *MLP*, é estabelecido através do teste estatístico de *Friedman*.

5.2 Experimentos em Dados Sintéticos Gaussianos

Na Seção 4.2.2 verificou-se que as perturbações dos parâmetros $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ são menores quando os vetores de teste são inseridos na classe correta. Nesta seção amplia-se o espectro desta análise para dados ainda sintéticos e com distribuição gaussiana, mas com 3, 4 e 5 classes.

Em cada caso, os experimentos realizados são idênticos aos que foram feitos para os dados com duas classes: são geradas sinteticamente 80 amostras de cada classe (Figuras 5.1, 5.4 e 5.7), sendo estas determinadas a partir de parâmetros $\boldsymbol{\mu}_i$ e Σ_i escolhidos de modo que possuam interseção entre as mesmas mas, que tenham também fronteiras bem estabelecidas

(separabilidade). Em seguida uma das classes é aleatoriamente escolhida para que a partir dela, sejam gerados 1000 amostras de teste. As amostras de teste são inseridas simultaneamente em todas as classes e as perturbações em $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ avaliadas. Os resultados obtidos são coletados e expostos na forma de histogramas nos quais pretende-se examinar a frequência destas perturbações.

5.2.1 Dados com 3 Classes

Para o caso em que os dados possuem 3 classes (Figura 5.1), usou-se para a sua geração:

$$\hat{\boldsymbol{\mu}}_1 = (0,00; 0,81), \quad \hat{\Sigma}_1 = \begin{bmatrix} 0,0780 & -0,0004 \\ -0,0004 & 0,0711 \end{bmatrix},$$

$$\hat{\boldsymbol{\mu}}_2 = (0,79; -0,40), \quad \hat{\Sigma}_2 = \begin{bmatrix} 0,0425 & -0,0030 \\ -0,0030 & 0,0463 \end{bmatrix},$$

$$\hat{\boldsymbol{\mu}}_3 = (-0,70; 0,40), \quad \hat{\Sigma}_3 = \begin{bmatrix} 0,0802 & -0,0311 \\ -0,0311 & 0,1009 \end{bmatrix}.$$

Figura 5.1: Dados Gaussianos com 3 classes, $\omega_1 \sim N(\boldsymbol{\mu}_1, \Sigma_1)$, $\omega_2 \sim N(\boldsymbol{\mu}_2, \Sigma_2)$ e $\omega_3 \sim N(\boldsymbol{\mu}_3, \Sigma_3)$

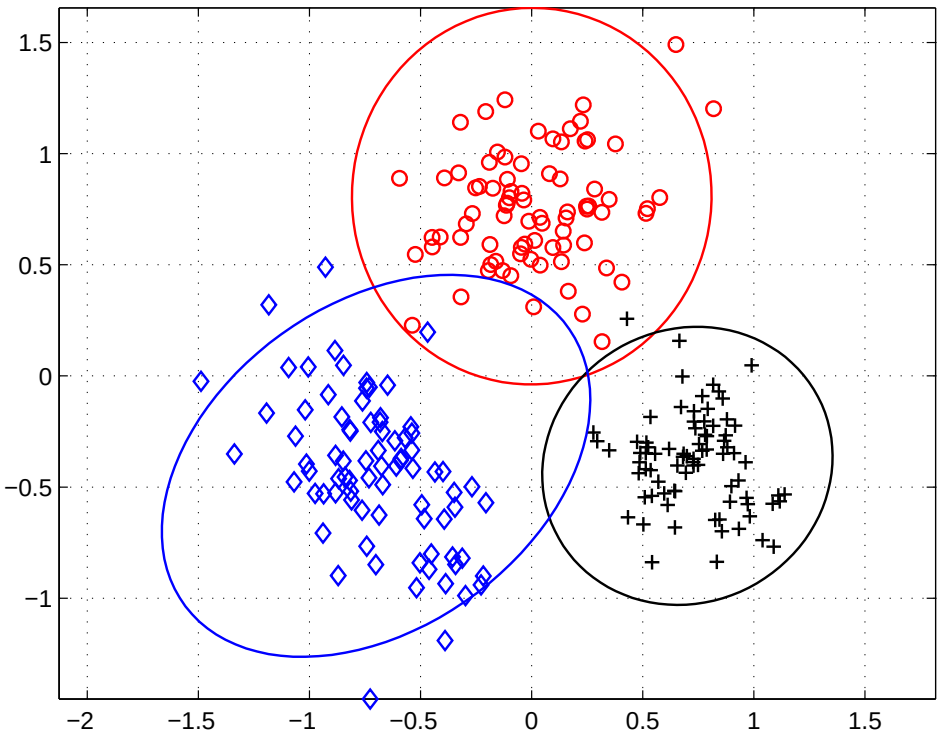


Figura 5.2: Histogramas das alterações ocorridas em $\hat{\boldsymbol{\mu}}_1, \hat{\boldsymbol{\mu}}_2$ e $\hat{\boldsymbol{\mu}}_3$.

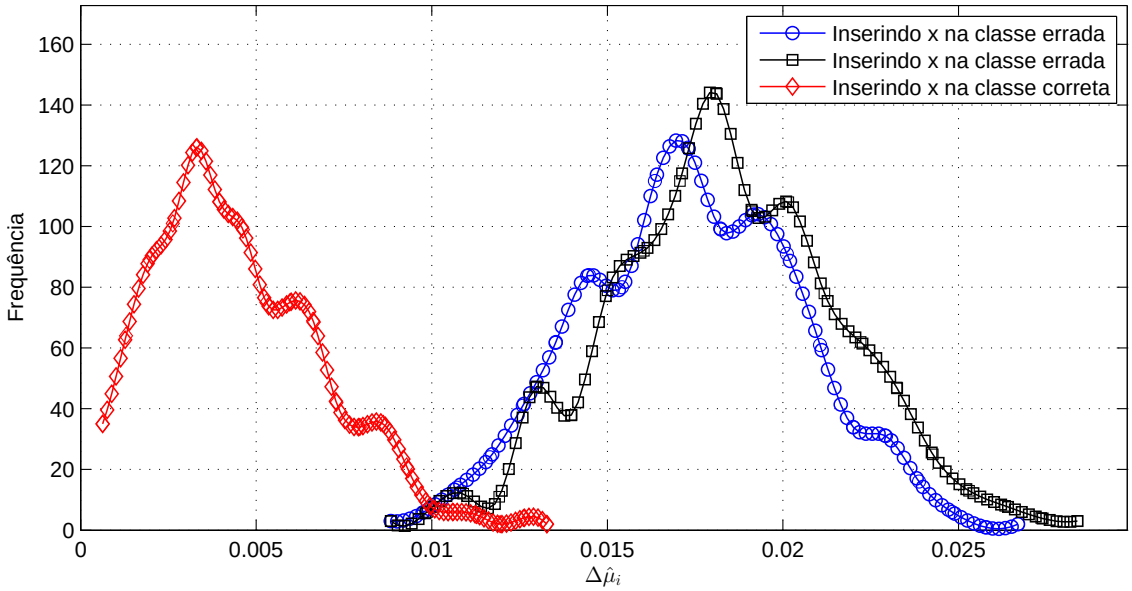
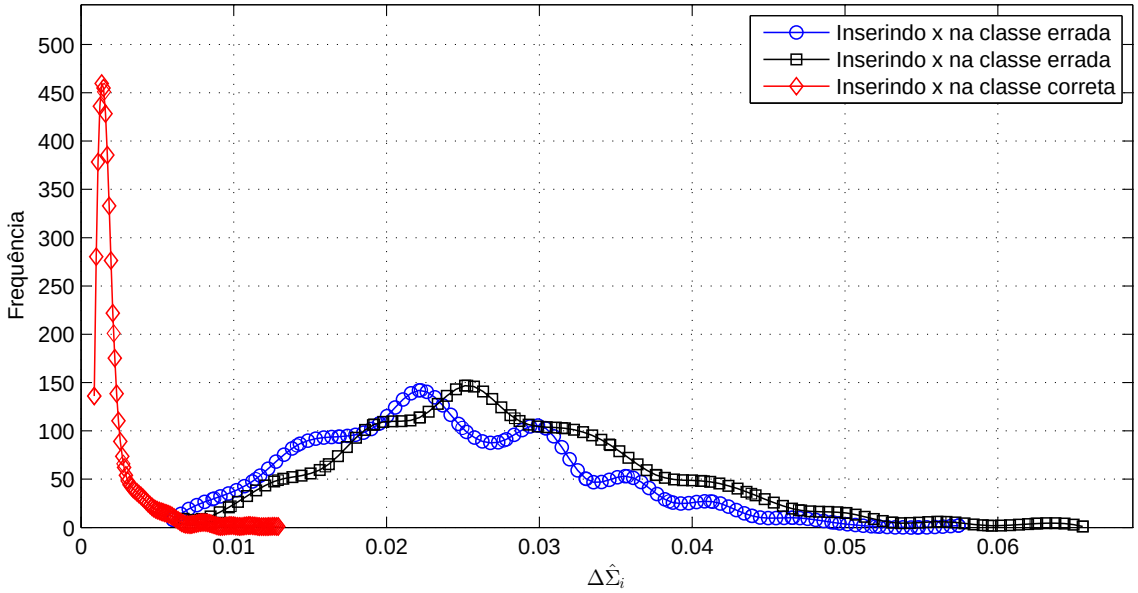


Figura 5.3: Histogramas das alterações ocorridas em $\hat{\Sigma}_1, \hat{\Sigma}_2$ e $\hat{\Sigma}_3$.



Novamente (Figuras 5.2, 5.3), as perturbações, tanto em $\hat{\boldsymbol{\mu}}_i$ quanto em $\hat{\Sigma}_i$, são menores quando os vetores de teste são inseridos na classe a qual pertencem.

Em seguida realizou-se a classificação destes dados e a validação dos resultados foi feita através do *10-fold cross-validation*. Esta classificação foi repetida 100 vezes e coletou-se a média e desvio padrão da taxa de acerto da classificação feita através das três versões do *PerC* (Tabela 5.1), além dos classificadores *5-NN*, *Naïve Bayes*, *SVM* com *kernel gaussiano*, *CART* e *MLP*. Percebe-se que as perturbações em $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ continuam proporcionando altas taxas de acerto, mesmo para dados com 3 classes. Além disto, a versão combinada *PerC(Comb)* teve uma performance melhor, embora não muito, em relação às outras versões do *PerC*. Comparando

com os demais classificadores obteve a terceira maior taxa de acerto.

Tabela 5.1: Classificação - Dados Gaussianos com 3 Classes

Classificador	Taxa de acerto(%)
PerC(Mean)	$97,26 \pm 0,32$
PerC(Cov)	$97,51 \pm 0,29$
PerC(Comb)	$97,59 \pm 0,41$
Naïve Bayes	$97,62 \pm 0,28$
5-NN	$97,02 \pm 0,41$
SVM(Gauss)	$\mathbf{97,76 \pm 0,35}$
CART	$94,78 \pm 0,67$
MLP	$95,13 \pm 1,93$

5.2.2 Dados com 4 Classes

Para o caso em que os dados possuem 4 classes (Figura 5.4), usou-se para a sua geração:

$$\hat{\boldsymbol{\mu}}_1 = (0,00; 0,83), \quad \hat{\Sigma}_1 = \begin{bmatrix} 0,0811 & -0,0178 \\ -0,0178 & 0,0591 \end{bmatrix},$$

$$\hat{\boldsymbol{\mu}}_2 = (0,83; 0,00), \quad \hat{\Sigma}_2 = \begin{bmatrix} 0,0743 & 0,0143 \\ 0,0143 & 0,0603 \end{bmatrix},$$

$$\hat{\boldsymbol{\mu}}_3 = (0,00; -0,83), \quad \hat{\Sigma}_3 = \begin{bmatrix} 0,0879 & 0,0130 \\ 0,0130 & 0,0891 \end{bmatrix},$$

$$\hat{\boldsymbol{\mu}}_4 = (-0,83; 0,00), \quad \hat{\Sigma}_4 = \begin{bmatrix} 0,0524 & 0,0118 \\ 0,0118 & 0,1098 \end{bmatrix}.$$

Figura 5.4: Dados Gaussianos com 4 classes, $\omega_1 \sim N(\boldsymbol{\mu}_1, \Sigma_1)$, $\omega_2 \sim N(\boldsymbol{\mu}_2, \Sigma_2)$, $\omega_3 \sim N(\boldsymbol{\mu}_3, \Sigma_3)$ e $\omega_4 \sim N(\boldsymbol{\mu}_4, \Sigma_4)$

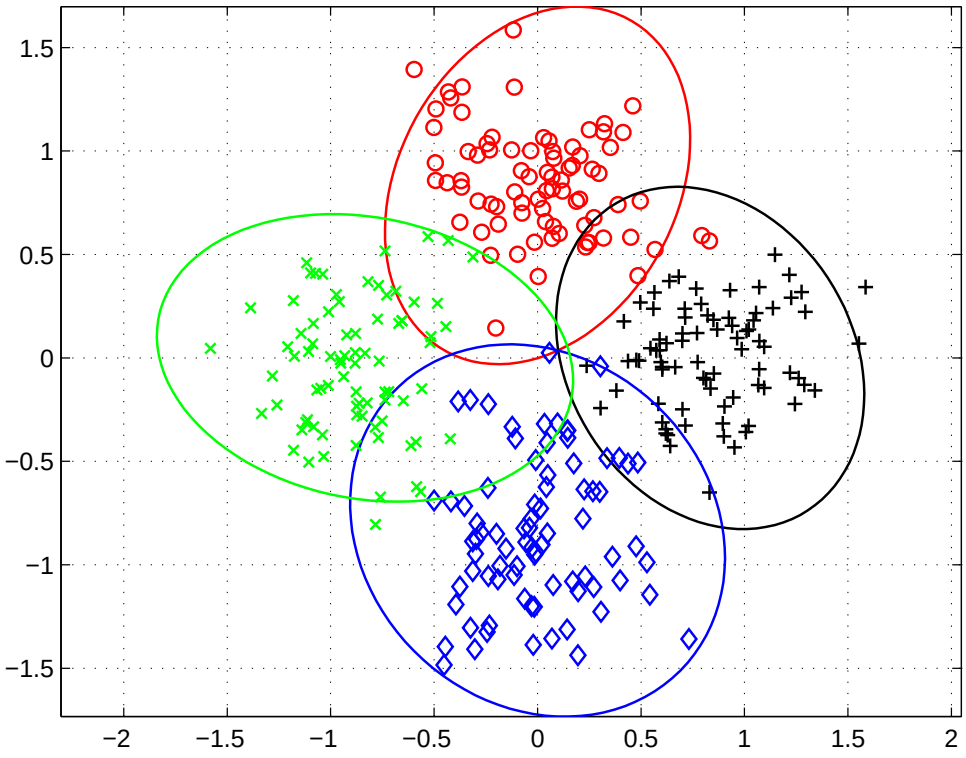


Figura 5.5: Histogramas das alterações ocorridas em $\hat{\boldsymbol{\mu}}_1, \hat{\boldsymbol{\mu}}_2, \hat{\boldsymbol{\mu}}_3$ e $\hat{\boldsymbol{\mu}}_4$.

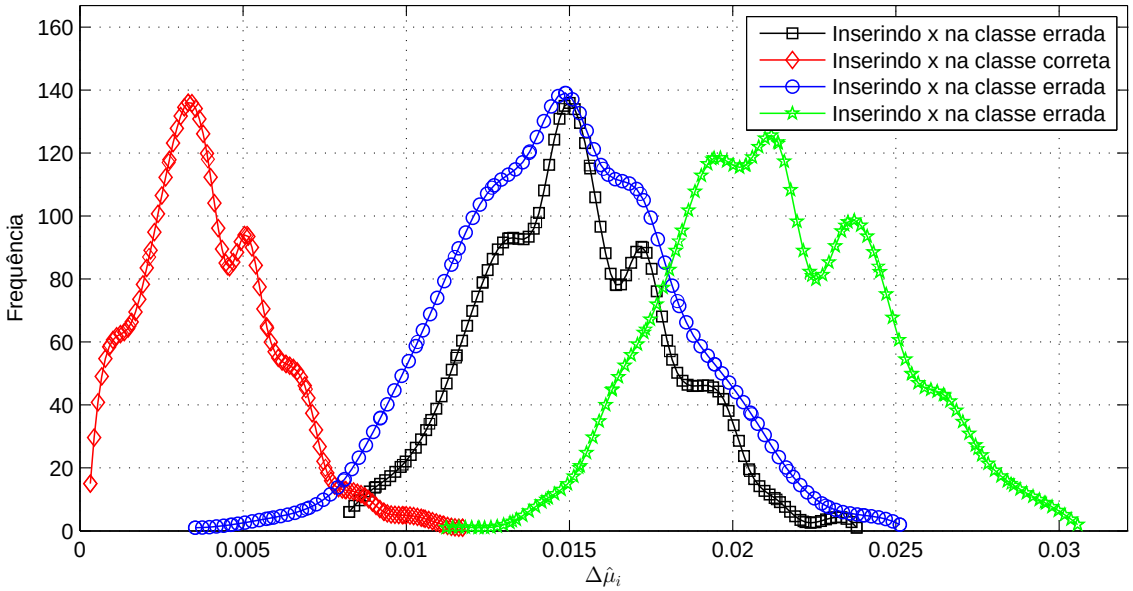
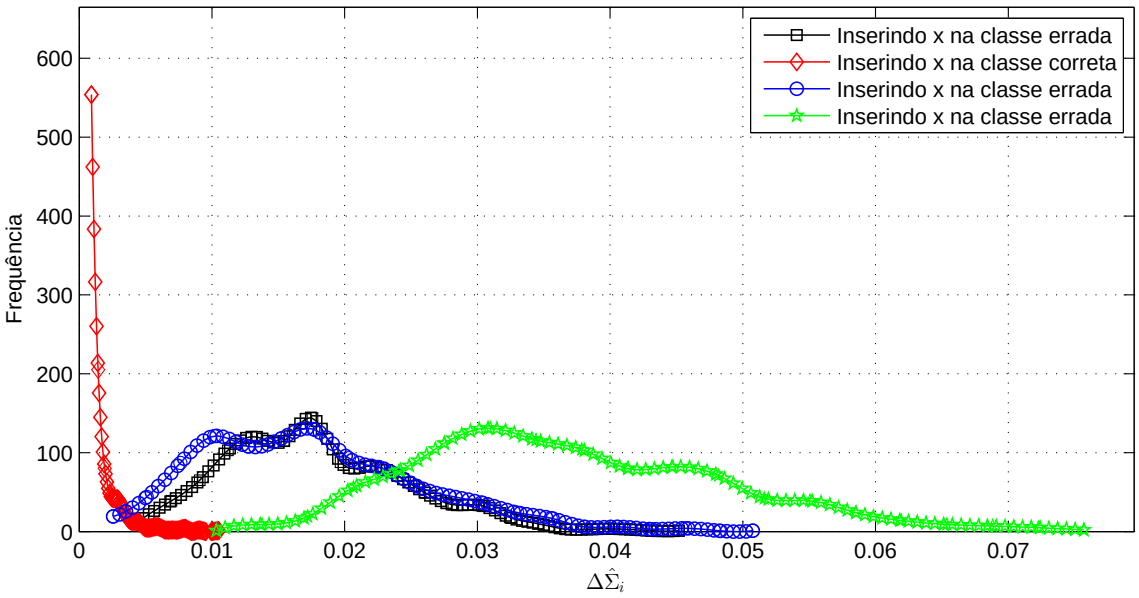


Figura 5.6: Histogramas das alterações ocorridas em $\hat{\Sigma}_1, \hat{\Sigma}_2, \hat{\Sigma}_3$ e $\hat{\Sigma}_4$.



Observando agora, a situação em que 4 classes estão envolvidas, embora exista um grau de confusão maior (Figuras 5.5 e 5.6), percebe-se ainda que há o acúmulo das perturbações de $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ em torno de valores menores quando os vetores de teste são inseridos na classe correta.

Tabela 5.2: Classificação - Dados Gaussianos com 4 Classes

Classificador	Taxa de acerto(%)
PerC(Mean)	$93,95 \pm 0,34$
PerC(Cov)	$94,38 \pm 0,11$
PerC(Comb)	$93,89 \pm 0,35$
Naïve Bayes	$93,68 \pm 0,37$
5-NN	$93,48 \pm 0,52$
SVM(Gauss)	$\mathbf{95,52 \pm 0,40}$
CART	$91,39 \pm 0,82$
MLP	$92,50 \pm 2,79$

Procedendo do mesmo modo feito para os dados gaussianos com 3 classes, a classificação dos dados e validação através do *10-fold cross validation* foi repetida 100 vezes e coletou-se a média e desvio padrão da taxa de acerto obtidas através das três versões do *PerC*, além dos classificadores utilizados anteriormente (Tabela 5.2). Novamente, percebe-se que as perturbações em $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$ continuam proporcionando altas taxas de acerto, embora menores que aquelas obtidas para dados com 3 classes. Esta redução sugere que a quantidade de classes nos dados reduz a eficiência da abordagem proposta. Embora a versão combinada, *PerC(Comb)*, não tenha obtido a melhor performance em relação às outras versões do *PerC*, obteve ainda assim a segunda melhor taxa de acerto quando comparado aos demais classificadores.

5.2.3 Dados com 5 Classes

Usou-se para a geração dos dados sintéticos gaussianos com 5 classes (Figura 5.7), os seguintes parâmetros:

$$\hat{\boldsymbol{\mu}}_1 = (0,00; 0,81), \qquad \hat{\Sigma}_1 = \begin{bmatrix} 0,1145 & 0,0178 \\ 0,0178 & 0,0584 \end{bmatrix},$$

$$\hat{\boldsymbol{\mu}}_2 = (0,78; 0,25), \qquad \hat{\Sigma}_2 = \begin{bmatrix} 0,1042 & -0,0115 \\ -0,0115 & 0,0770 \end{bmatrix},$$

$$\hat{\boldsymbol{\mu}}_3 = (0,48; -0,66), \qquad \hat{\Sigma}_3 = \begin{bmatrix} 0,0724 & 0,0132 \\ 0,0132 & 0,0489 \end{bmatrix},$$

$$\hat{\boldsymbol{\mu}}_4 = (-0,48; -0,66), \qquad \hat{\Sigma}_4 = \begin{bmatrix} 0,0536 & -0,0154 \\ -0,0154 & 0,0887 \end{bmatrix},$$

$$\hat{\boldsymbol{\mu}}_5 = (-0,78; 0,25), \qquad \hat{\Sigma}_5 = \begin{bmatrix} 0,0935 & -0,0071 \\ -0,0071 & 0,0608 \end{bmatrix}.$$

Figura 5.7: Dados Gaussianos com 5 classes, $\omega_1 \sim N(\boldsymbol{\mu}_1, \Sigma_1), \ldots, \omega_5 \sim N(\boldsymbol{\mu}_5, \Sigma_5)$

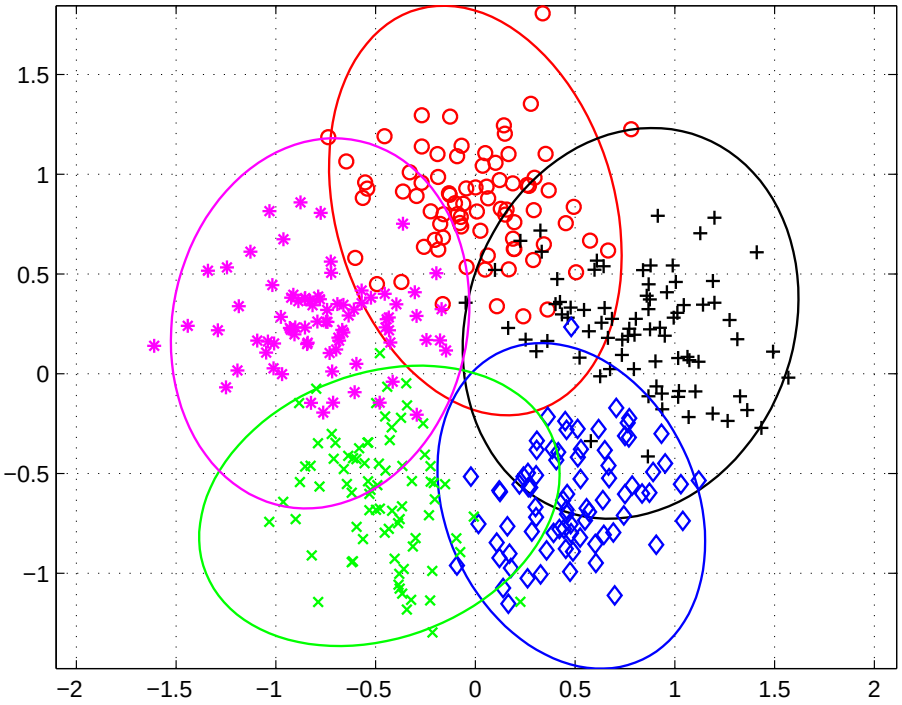


Figura 5.8: Histogramas das alterações ocorridas em $\hat{\boldsymbol{\mu}}_1, ..., \hat{\boldsymbol{\mu}}_5$

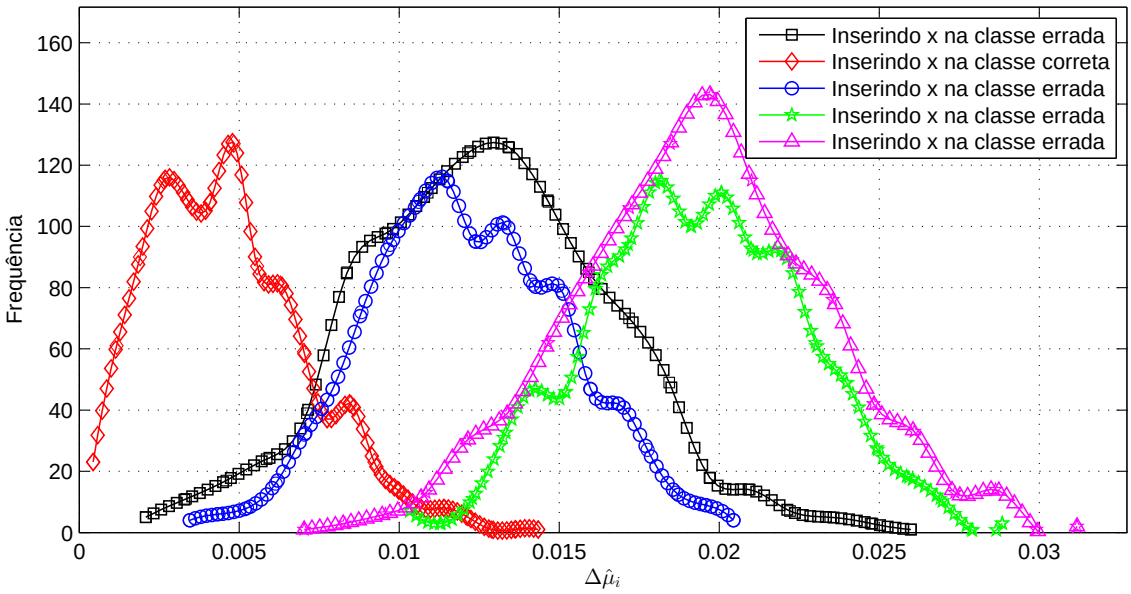
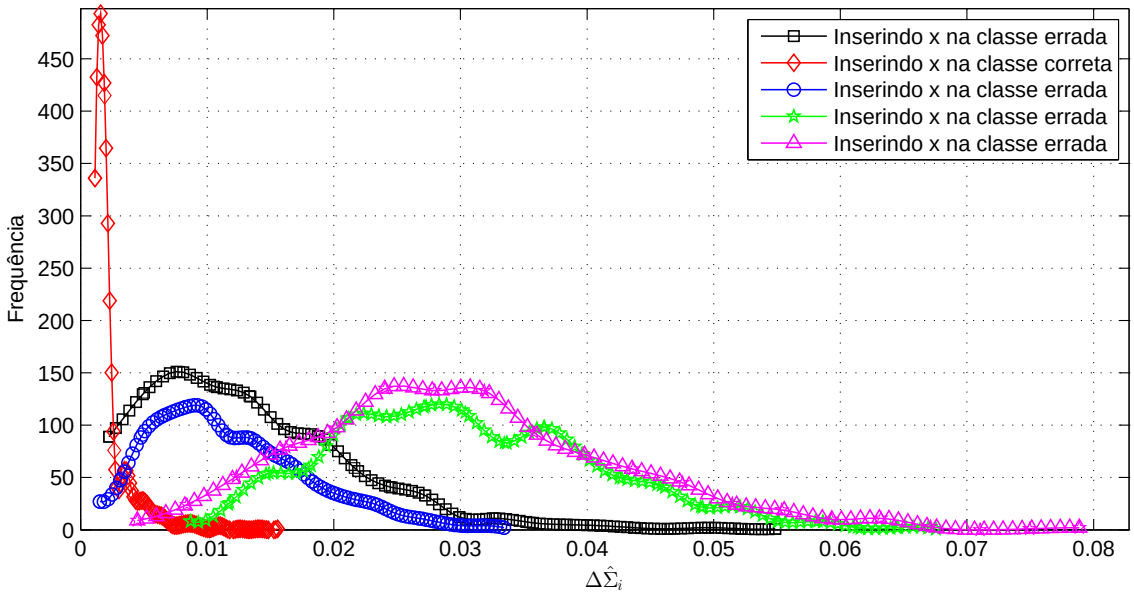


Figura 5.9: Histogramas das alterações ocorridas em $\hat{\Sigma}_1, ...\hat{\Sigma}_5$



O mesmo comportamento verifica-se para 5 classes envolvidas no experimento (Figuras 5.8 e 5.9), sendo a quantidade de confusão ainda maior que nos casos anteriores. Observa-se também, para esta situação que as perturbações em $\hat{\Sigma}_i$ apresentam um acúmulo mais evidente em torno de valores menores para as inserções corretas dos vetores de teste.

Após 100 repetições deste experimento, nos moldes realizados anteriormente para os dados com 3 e 4 classes, constata-se que a redução na eficiência dos três classificadores permanece, embora as taxas de acerto continuem altas tendo em vista a quantidade de embaralhamento entre as 5 classes presentes nos dados (Tabela 5.3). O *PerC(Comb)* foi novamente superior ao *PerC(Mean)* e *PerC(Cov)*, podendo ser considerada portanto, a melhor

Tabela 5.3: Classificação - Dados Gaussianos com 5 Classes

Classificador	Taxa de acerto(%)
PerC(Mean)	$91,25 \pm 0,42$
PerC(Cov)	$91,44 \pm 0,38$
PerC(Comb)	$92,07 \pm 0,33$
Naïve Bayes	$90,84 \pm 0,33$
5-NN	$89,43 \pm 0,50$
SVM(Gauss)	$\mathbf{92,75 \pm 0,42}$
CART	$88,97 \pm 0,76$
MLP	$89,08 \pm 2,72$

entre as três versões propostas deste classificador. Quando comparado aos demais classificadores, o *PerC(Comb)* mais um vez apresentou a segunda maior taxa de acerto médio, sendo inferior apenas ao *SVM(Gauss)*.

Conclui-se portanto que o poder discriminatório das perturbações $\Delta\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ e $\Delta\hat{\Sigma}_i$ permanece válido para dados gaussianos com 3, 4 e 5 classes. Sua eficiência reduz-se gradativamente com o aumento do número de classes, porém não mais que os classificadores comparados.

5.3 Experimentos em Dados Sintéticos Não-Gaussianos

Pretende-se nesta seção, avaliar o poder de discriminação das perturbações em $\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}$ e $\hat{\Sigma}_i$ para dados sintéticos que não possuem distribuição gaussiana.

Os experimentos realizados seguem a seguinte metodologia:

- Os bancos de dados utilizados foram: Banana set (Figura 5.10), P2 Problem (Figura 5.13), Two Spirals (Figura 5.16), Cluster in Cluster (Figura 5.19), Corners (Figura 5.22), Crescent and Full Moon (Figura 5.25), Half-kernel (Figura 5.28) e Outlier (Figura 5.31).

- Uma das classes presentes no banco de dados é escolhida aleatoriamente e a partir desta classe, são geradas 1000 novas amostras de teste.

- As amostras de teste são inseridas simultaneamente em cada uma das classes do banco de treinamento: na classe correta, à qual pertencem, e nas demais. Em seguida são calculadas as perturbacões em $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma}_i$.

- Os resultados obtidos, são exibidos na forma de histogramas, nos quais pretende-se comparar a incidência dos valores obtidos para as inserções corretas e as incorretas.

- Realizou-se também a classificação destes bancos de dados através das três versões do *PerC* e os classificadores *5-NN*, *Naïve Bayes*, *SVM* com *kernel gaussiano*, *CART*

e *MLP*, usando o *10-fold cross validation* para validação dos resultados. Este experimento foi repetido 100 vezes coletando-se a média e desvio padrão das taxas de acerto obtidas.

5.3.1 Experimentos: *Banana Set*

Figura 5.10: *Banana Set*: 2 classes com 500 amostras em cada.

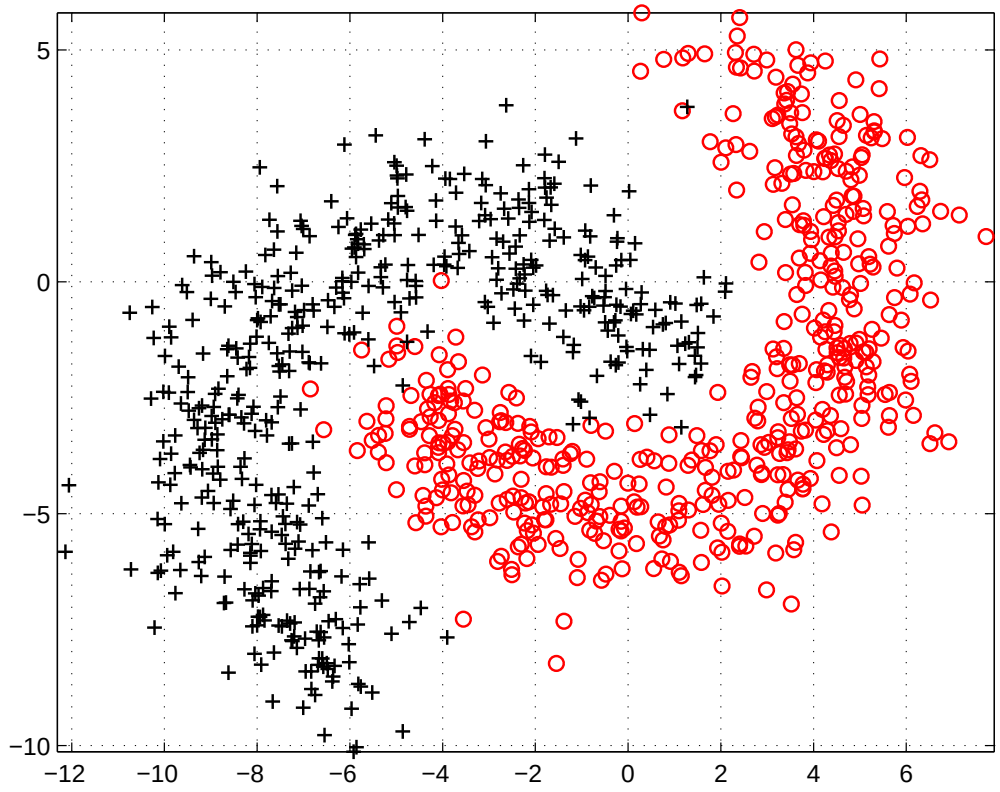


Figura 5.11: *Banana Set*: Histograma para as perturbações em $\hat{\boldsymbol{\mu}}_i$, $i = 1, 2$

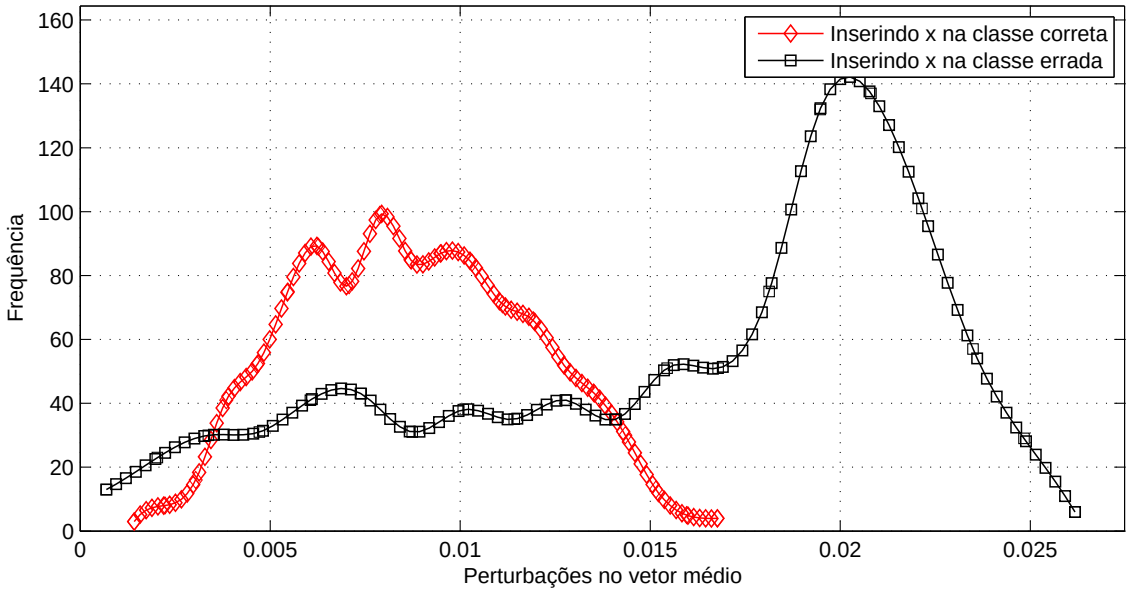
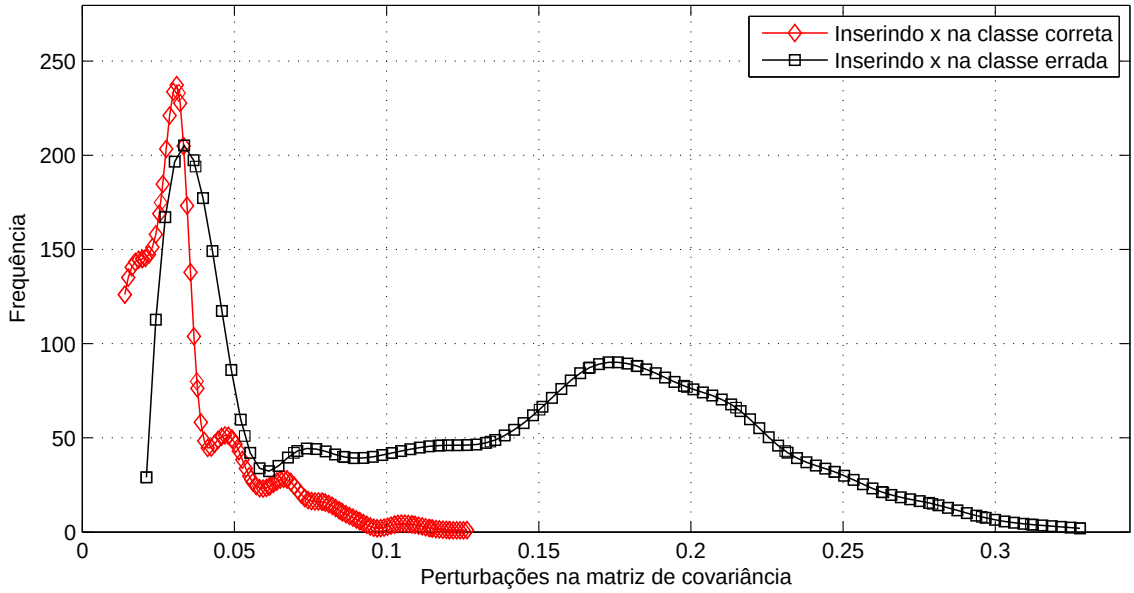


Figura 5.12: *Banana Set*: Histograma para as perturbações em $\hat{\Sigma}_i$, $i = 1, 2$



De acordo com os resultados (Figuras 5.11 e 5.12), apesar de existir um certo embaralhamento entre as classes, ambas as perturbações, $\Delta\hat{\Sigma}_i$ e $\Delta\hat{\boldsymbol{\mu}}_i$, se acumulam em torno de valores menores quando as inserções são realizadas na classe correta. De acordo com a metodologia proposta e verificar-se o poder de discriminação de tais perturbações, realizou-se classificação deste banco de dados por 100 vezes e validando os resultados através do *10-fold cross validation*. As taxas de acerto médio e os respectivos desvios-padrão foram coletados e estão expostos na Tabela 5.4.

Tabela 5.4: *Banana Set*: Classificação

Classificador	Taxa de acerto(%)
PerC(Mean)	$78,49 \pm 0,15$
PerC(Cov)	$85,87 \pm 0,16$
PerC(Comb)	$84,15 \pm 0,12$
Naïve Bayes	$78,37 \pm 0,12$
5-NN	$97,29 \pm 0,14$
SVM(Gauss)	$\mathbf{98,10 \pm 0,16}$
CART	$96,69 \pm 0,37$
MLP	$97,66 \pm 0,21$

Observa-se taxas de acerto razoáveis para as três versões do *PerC*, indicando que apesar do embaralhamento apresentado nos histogramas, o poder de discriminação se confirma (Tabela 5.4). Entretanto, no comparativo com os demais classificadores ficou entre os de menores taxas de acerto médio.

Comparando apenas as três versões do *PerC*, percebe-se que a versão baseada nas alterações da matriz de covariância possui maior taxa de acerto médio, 85,87%, que a versão baseada nas perturbações do vetor médio, 78,48%. Observando a distribuição dos dados (Figura

5.10) é possível notar que a covariância apresenta razoável distinção entre as classes ao passo que o vetor médio de ambas aparentam se próximos, o que pode justificar a performance superior do *PerC(Cov)*.

5.3.2 Experimentos: *P2 Dataset*

Figura 5.13: *P2 Dataset*: 2 classes com 500 amostras em cada.

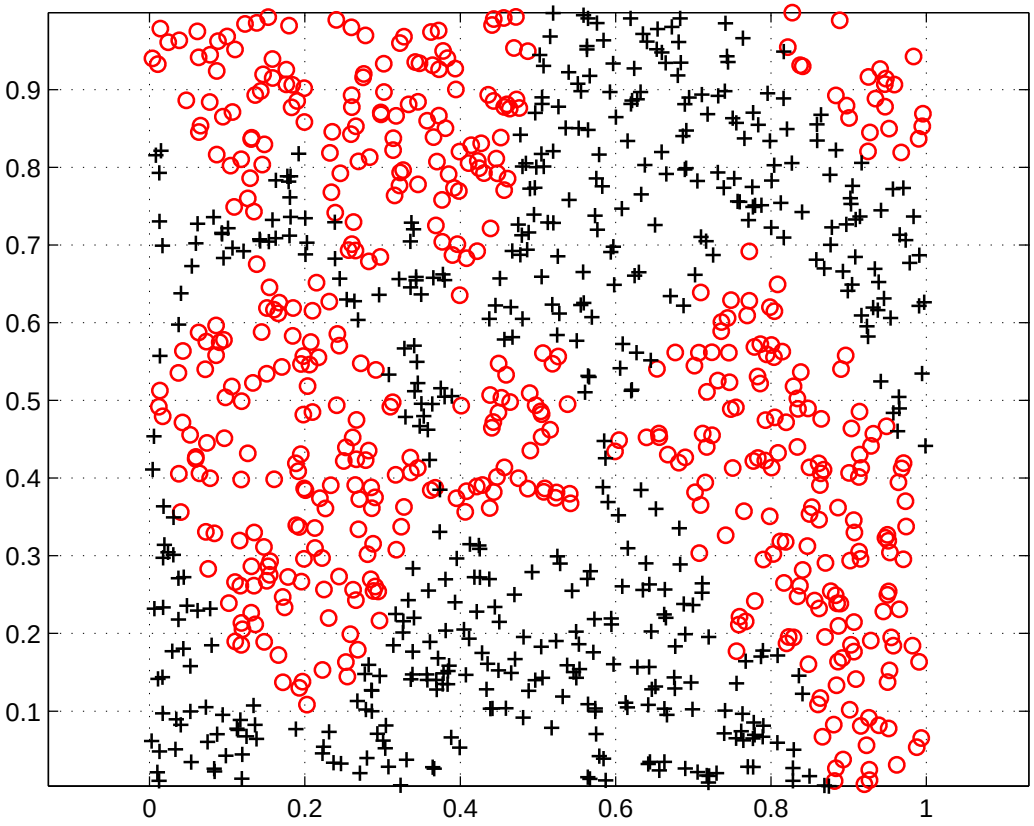


Figura 5.14: *P2 Dataset*: Histograma para as perturbações em $\hat{\boldsymbol{\mu}}_i$, $i = 1, 2$

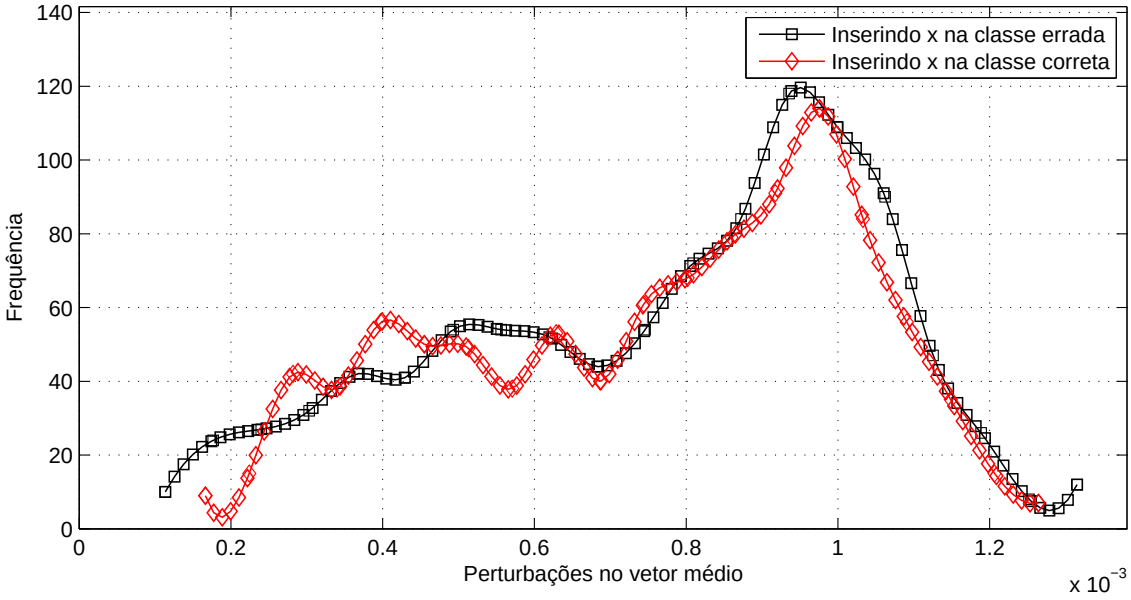
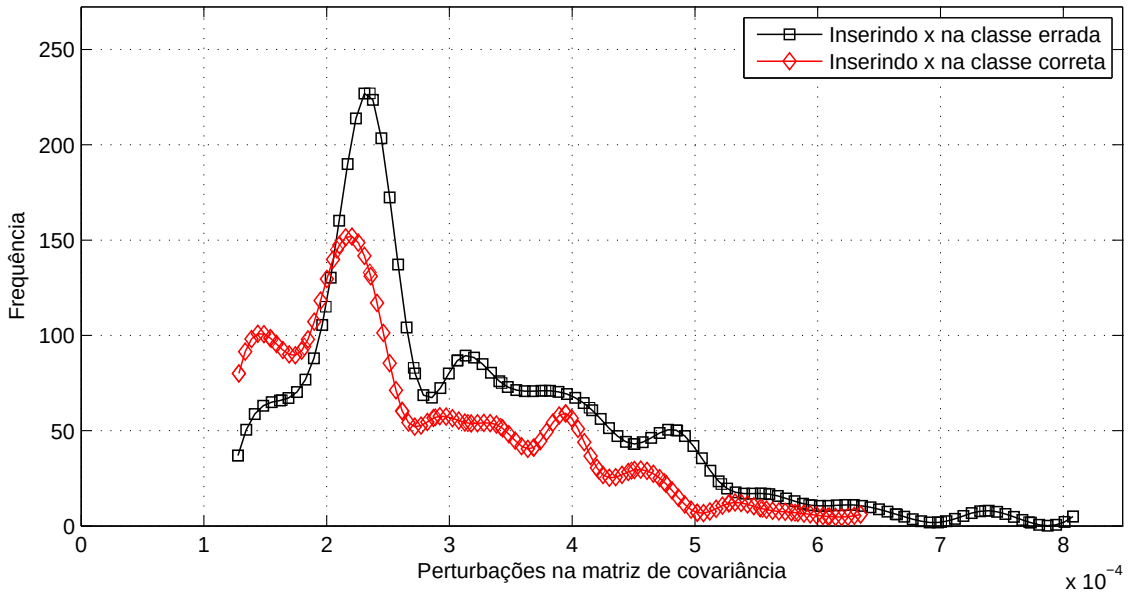


Figura 5.15: *P2 Dataset*: Histograma para as perturbações em $\hat{\Sigma}_i$, $i = 1, 2$



Neste caso, não há uma distinção clara entre os histogramas obtidos para as perturbações em $\hat{\boldsymbol{\mu}}_i$ ou em $\hat{\Sigma}_i$ (Figuras 5.14 e 5.15). Entretanto, após realizar os experimentos propostos, verificou-se que apesar da difícil separabilidade entre as classes (Figura 5.13), o *PerC(Cov)* e *PerC(Comb)* apresentaram taxas de acerto razoáveis (Tabela 5.5). Entretanto, este desempenho encontra-se bem abaixo da média apresentada pelos demais classificadores.

Novamente, é possível perceber nestes dados que os vetores médios das classes presentes são de difícil distinção, bem como suas distribuições (a covariância). De modo que, a performance ruim do *PerC*, seja possivelmente consequência destas características.

Tabela 5.5: *P2 Dataset*: Classificação

Classificador	Taxa de acerto(%)
PerC(Mean)	$49,64 \pm 0,47$
PerC(Cov)	$67,20 \pm 0,57$
PerC(Comb)	$66,44 \pm 0,48$
Naïve Bayes	$55,22 \pm 0,43$
5-NN	$\mathbf{93,05 \pm 0,36}$
SVM(Gauss)	$80,11 \pm 0,34$
CART	$89,40 \pm 0,69$
MLP	$86,95 \pm 1,78$

5.3.3 Experimentos: *Two Spirals Dataset*

Figura 5.16: *Two Spirals Dataset*: 2 classes com 500 amostras em cada.

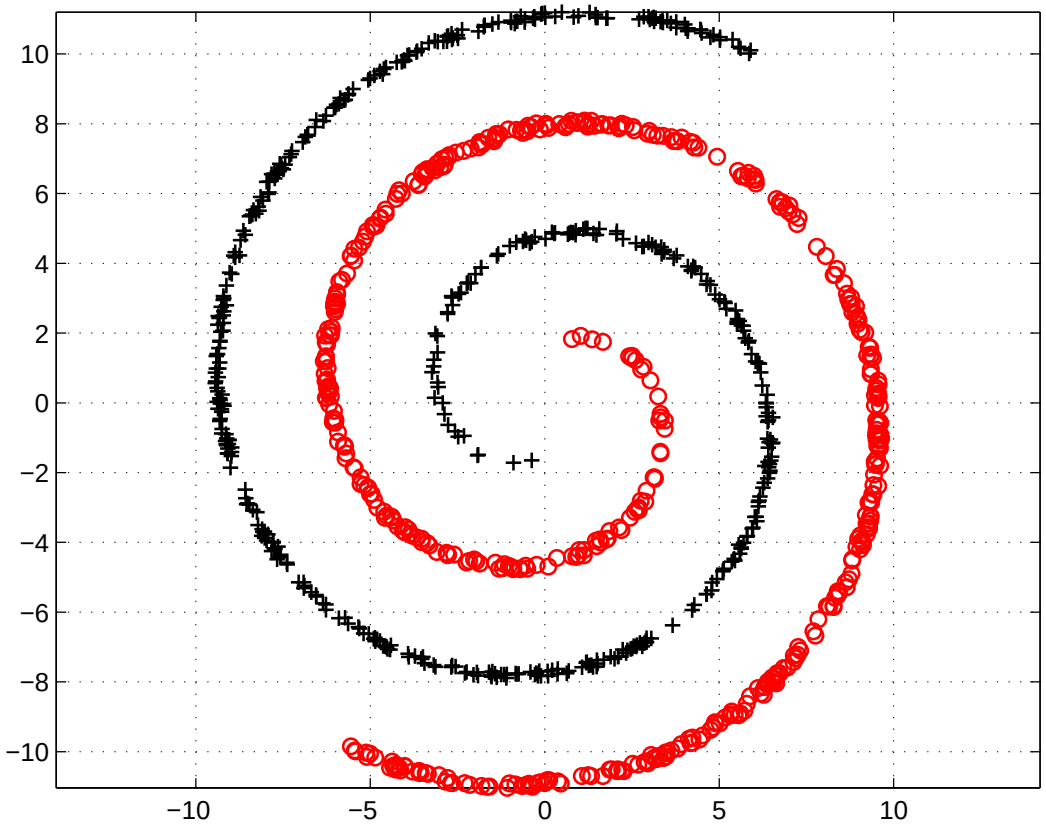


Figura 5.17: *Two Spirals Dataset*: Histograma para as perturbações em $\hat{\boldsymbol{\mu}}_i, i = 1, 2$

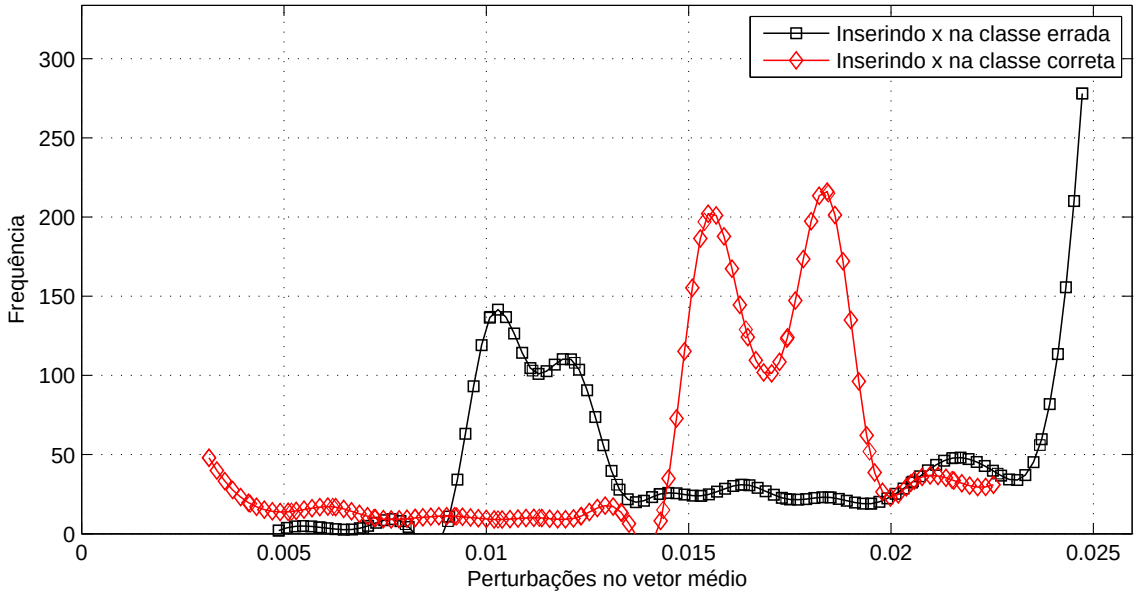
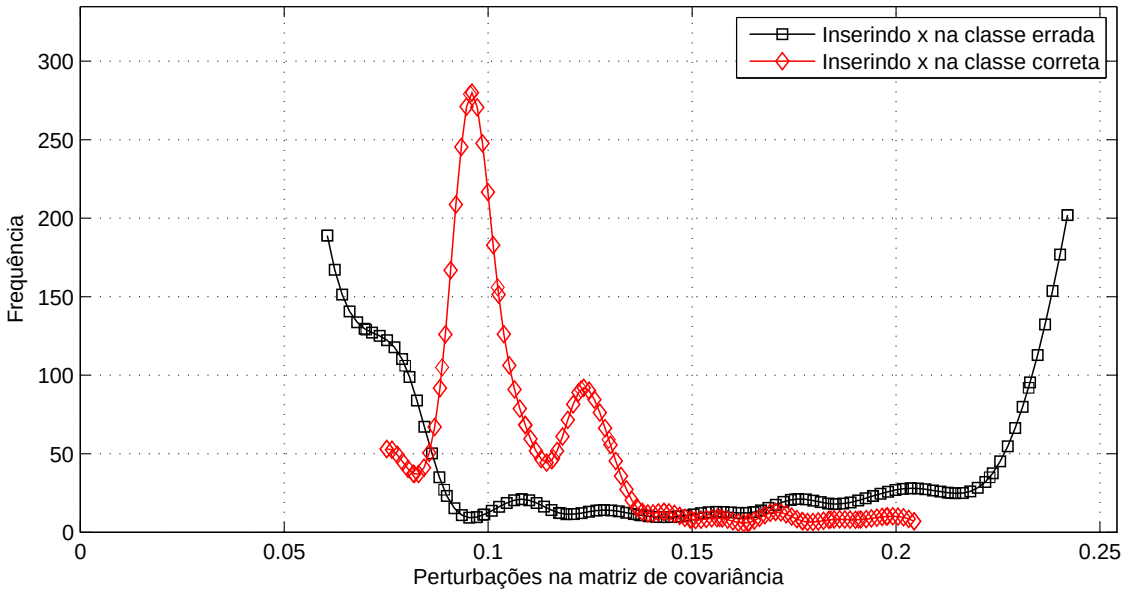


Figura 5.18: *Two Spirals Dataset*: Histograma para as perturbações em $\hat{\Sigma}_i, i = 1, 2$



Para o Two Spirals Dataset ocorre o contrário do que é proposto neste trabalho: as inserções na classe incorreta acumulam-se em torno de valores menores que os encontrados para as inserções na classe correta (Figuras 5.17 e 5.18).

A estrutura de covariância em ambas as classes são semelhantes, o que justifica a baixa taxa de acerto do *PerC(Cov)*. Do mesmo modo, o vetores médios encontram-se a pequena distância um do outro e por isto o poder discriminatório do *PerC(Mean)* também apresenta-se reduzido. Apesar disto, a versão combinada conseguiu extrair o máximo das duas perturbações.

A classificação conforme a metodologia proposta, demonstra para este banco de dados que o *PerC* encontra-se abaixo da performance média encontrada para os demais classificadores

(Tabela 5.6).

Tabela 5.6: *Two Spirals Dataset*: Classificação

Classificador	Taxa de acerto(%)
PerC(Mean)	$66,05 \pm 0,28$
PerC(Cov)	$53,79 \pm 0,19$
PerC(Comb)	$66,85 \pm 0,21$
Naïve Bayes	$66,94 \pm 0,20$
5-NN	$\mathbf{100,000 \pm 0,00}$
SVM(Gauss)	$\mathbf{100,000 \pm 0,00}$
CART	$99,23 \pm 0,18$
MLP	$96,08 \pm 3,13$

Apesar das classes neste banco de dados serem bem distintas, uma curva que defina a fronteira entre elas seria de difícil traçado, levando a crer que seja este o motivo do baixo percentual de acerto do *PerC* (Tabela 5.6).

5.3.4 Experimentos: *Cluster in Cluster Dataset*

Figura 5.19: *Cluster in Cluster Dataset*: 2 Classes com 500 amostras em cada.

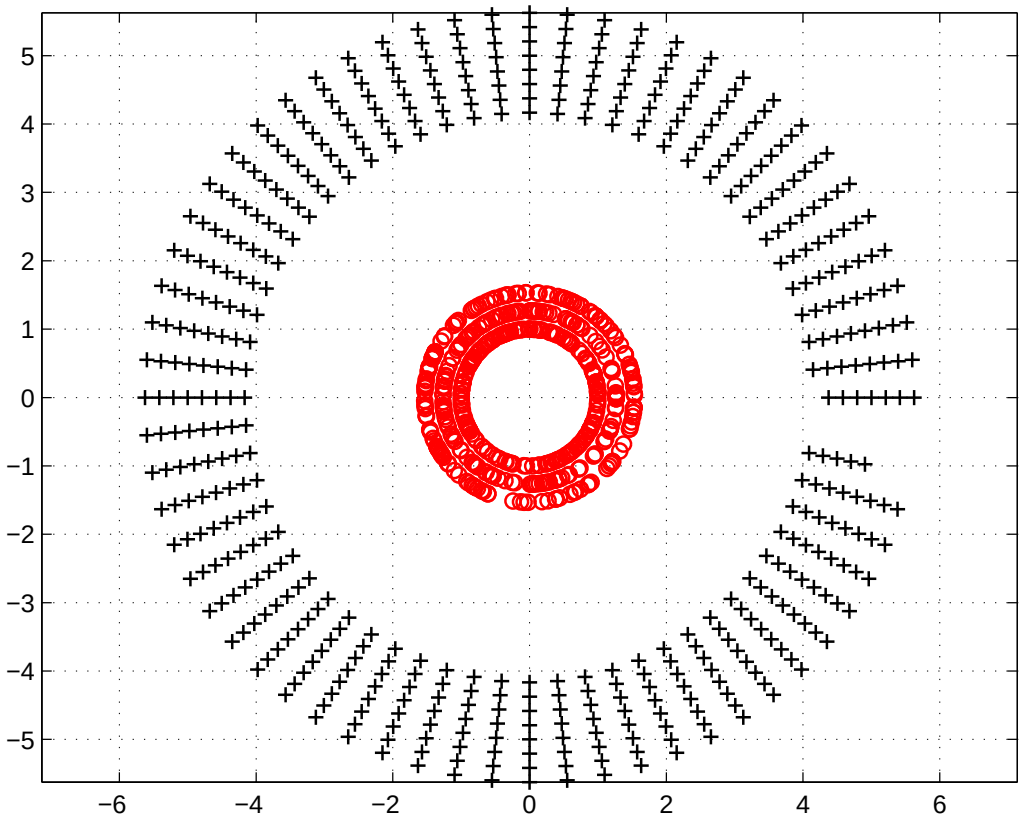


Figura 5.20: *Cluster in Cluster Dataset*: Histograma para as perturbações em $\hat{\boldsymbol{\mu}}_i$, $i = 1, 2$

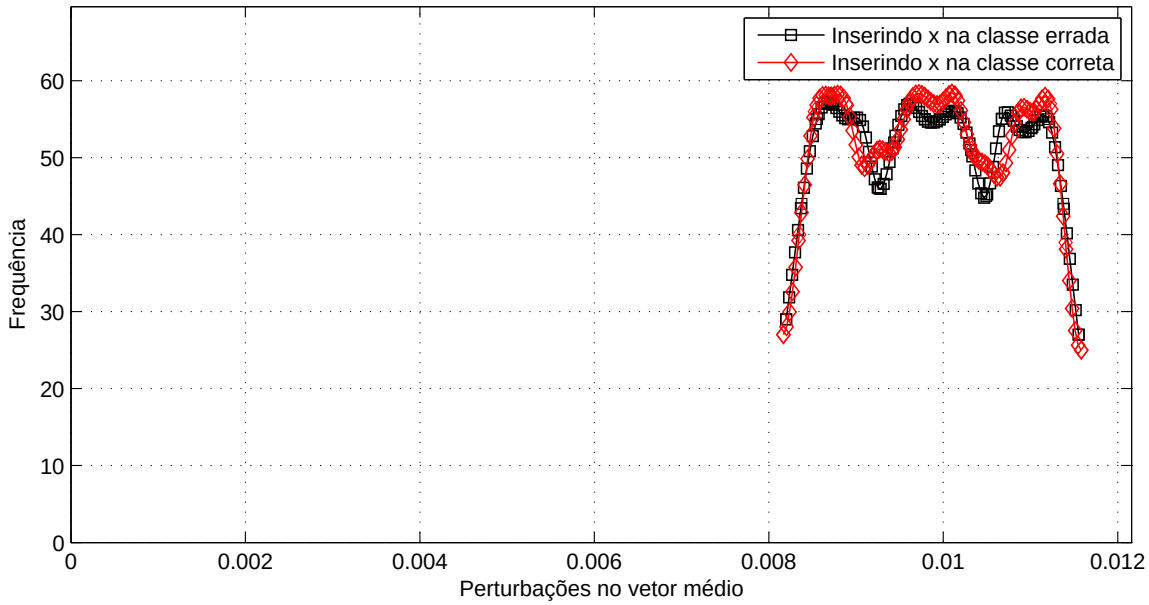
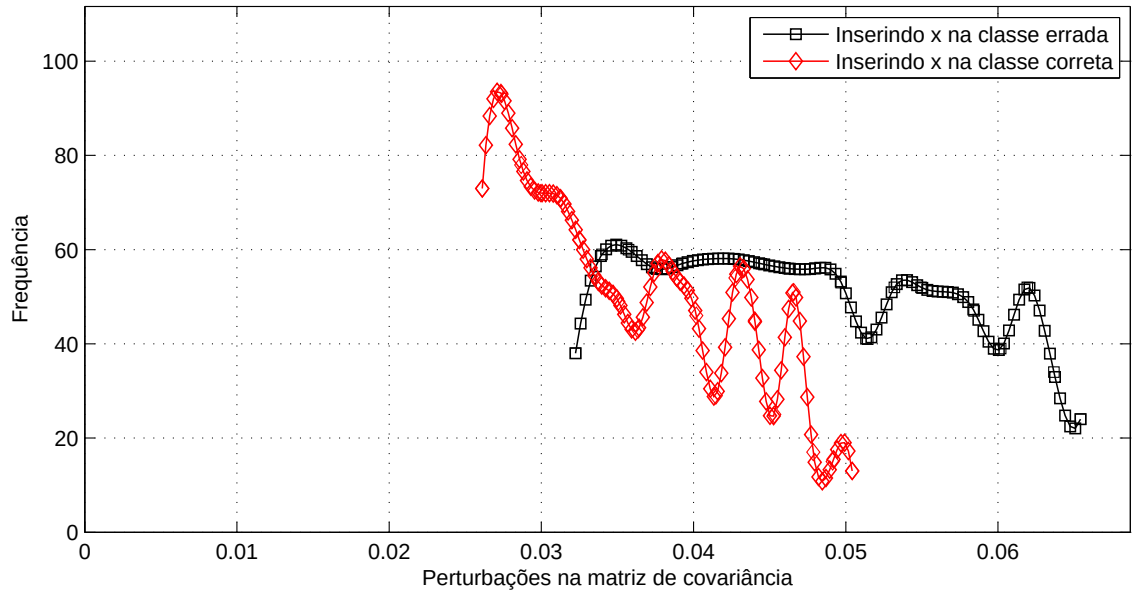


Figura 5.21: *Cluster in Cluster Dataset*: Histograma para as perturbações em $\hat{\Sigma}_i$, $i = 1, 2$



Novamente, não há distinção entre os histogramas encontrados para as inserções na classe correta ou incorreta, para ambas as perturbações(Figuras 5.20 e 5.21).

Entretanto, a classificação baseada nas perturbações ocorridas em $\hat{\Sigma}_i$, *PerC(Cov)* apresentam notáveis 100% de acerto (Tabela 5.7). Observe que neste banco de dados, há também uma clara distinção entre as classes (Figura 5.19) porém, uma curva que represente a fronteira entre elas seria de fácil traçado, ao contrário do que ocorre com bancos P2 Problem (Figura 5.13) e Two Spirals Dataset (Figura 5.16). Note também que as duas classes deste banco de dados possuem praticamente a mesma média ($\hat{\boldsymbol{\mu}}_1 = (-0,04; 0,02)$ e $\hat{\boldsymbol{\mu}}_2 = (-0,11; 0,01)$, com $\text{dist}(\hat{\boldsymbol{\mu}}_1, \hat{\boldsymbol{\mu}}_2) = 0,07$) (Figura 5.19), ou seja, as perturbações em $\hat{\boldsymbol{\mu}}_1$ e $\hat{\boldsymbol{\mu}}_2$ após a inserção de um

Tabela 5.7: *Cluster in Cluster Dataset*: Classificação

Classificador	Taxa de acerto(%)
PerC(Mean)	$49,51 \pm 1,52$
PerC(Cov)	$\mathbf{100,00 \pm 0,00}$
PerC(Comb)	$50,00 \pm 0,00$
Naïve Bayes	$\mathbf{100,00 \pm 0,00}$
5-NN	$\mathbf{100,00 \pm 0,00}$
SVM(Gauss)	$\mathbf{100,00 \pm 0,00}$
CART	$\mathbf{100,00 \pm 0,00}$
MLP	$99,96 \pm 0,36$

vetor de teste, são quase iguais, o que torna $\Delta\hat{\boldsymbol{\mu}}_1$ e $\Delta\hat{\boldsymbol{\mu}}_2$ sem utilidade para classificação. Este fato explica a baixa eficiência obtida para o *PerC(Mean)* neste banco de dados.

É importante destacar também que apesar de ter melhor performance que o *PerC(Mean)*, o *PerC(Comb)* não conseguiu usar todo o poder de discriminação demonstrado pelo *PerC(Cov)*.

5.3.5 Experimentos: *Corners Dataset*

Figura 5.22: *Corners Dataset*: 4 classes com 500 amostras em cada

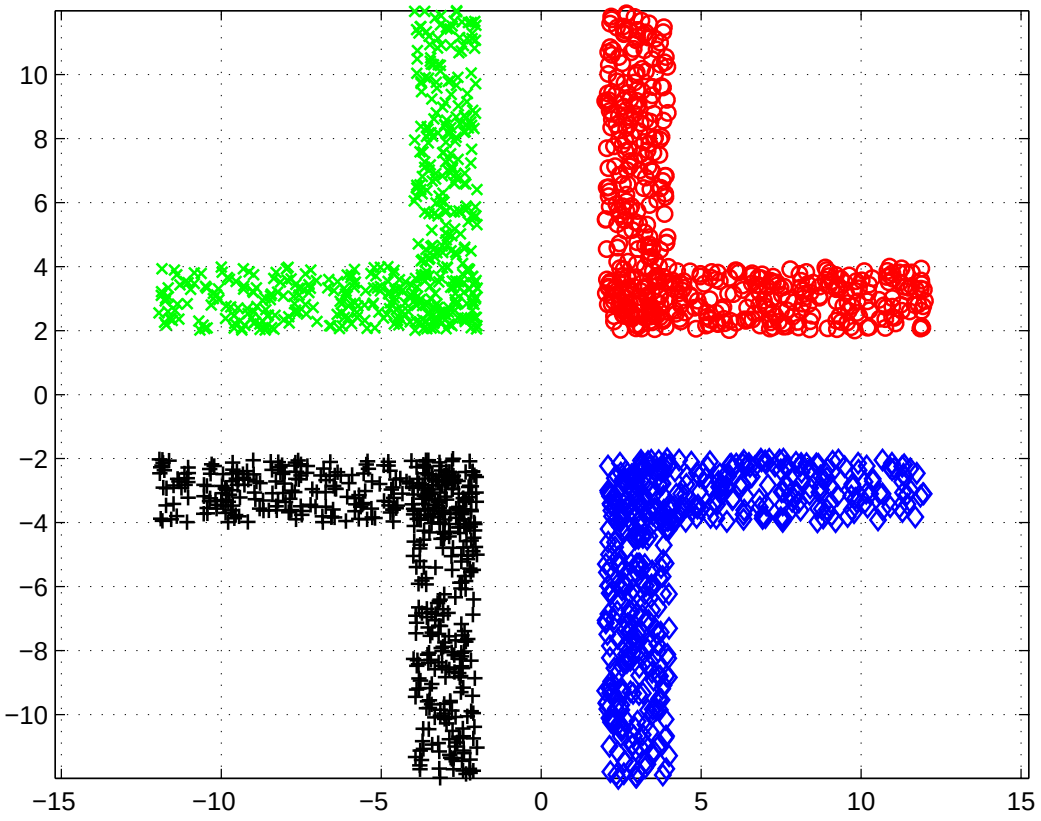


Figura 5.23: *Corners Dataset*: Histograma para as perturbações em $\hat{\boldsymbol{\mu}}_i$, $i = 1, 2, 3, 4$

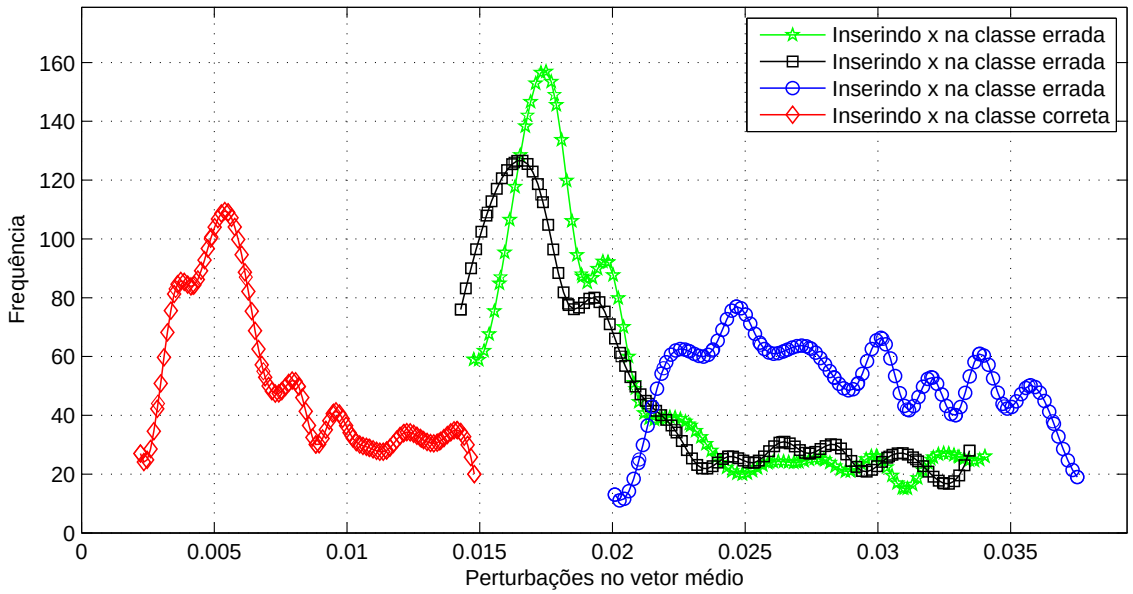
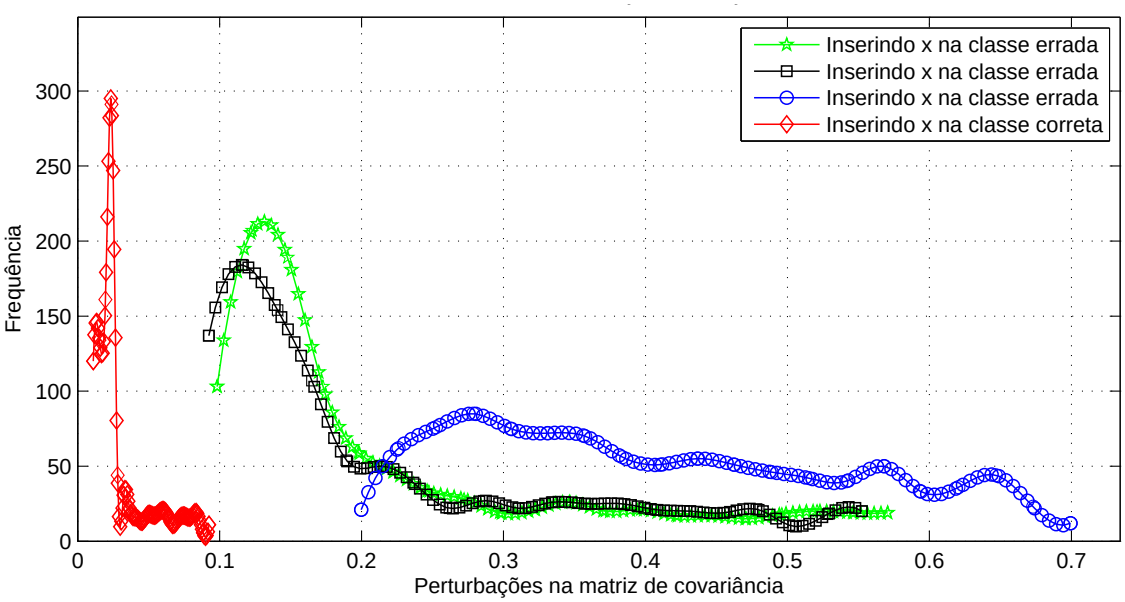


Figura 5.24: *Corners Dataset*: Histograma para as perturbações em $\hat{\Sigma}_i$, $i = 1, 2, 3, 4$



Para o banco de dados Corners (Figura 5.22) foram encontrados histogramas com distinções bastante claras em ambas as perturbações (Figuras 5.23 e 5.24) e surpreendentes 100% de acerto para os três classificadores (Tabela 5.8).

Tabela 5.8: *Corners Dataset*: Classificação

Classificador	Taxa de acerto(%)
PerC(Mean)	$\mathbf{100,00 \pm 0,00}$
PerC(Cov)	$\mathbf{100,00 \pm 0,00}$
PerC(Comb)	$\mathbf{100,00 \pm 0,00}$
Naïve Bayes	$\mathbf{100,00 \pm 0,00}$
5-NN	$\mathbf{100,00 \pm 0,00}$
SVM(Gauss)	$\mathbf{100,00 \pm 0,00}$
CART	$\mathbf{100,00 \pm 0,00}$
MLP	$99,43 \pm 0,36$

5.3.6 Experimentos: *Crescent & Full Moon Dataset*

Figura 5.25: *Crescent & Full Moon Dataset*: 2 classes com 500 amostras em cada.

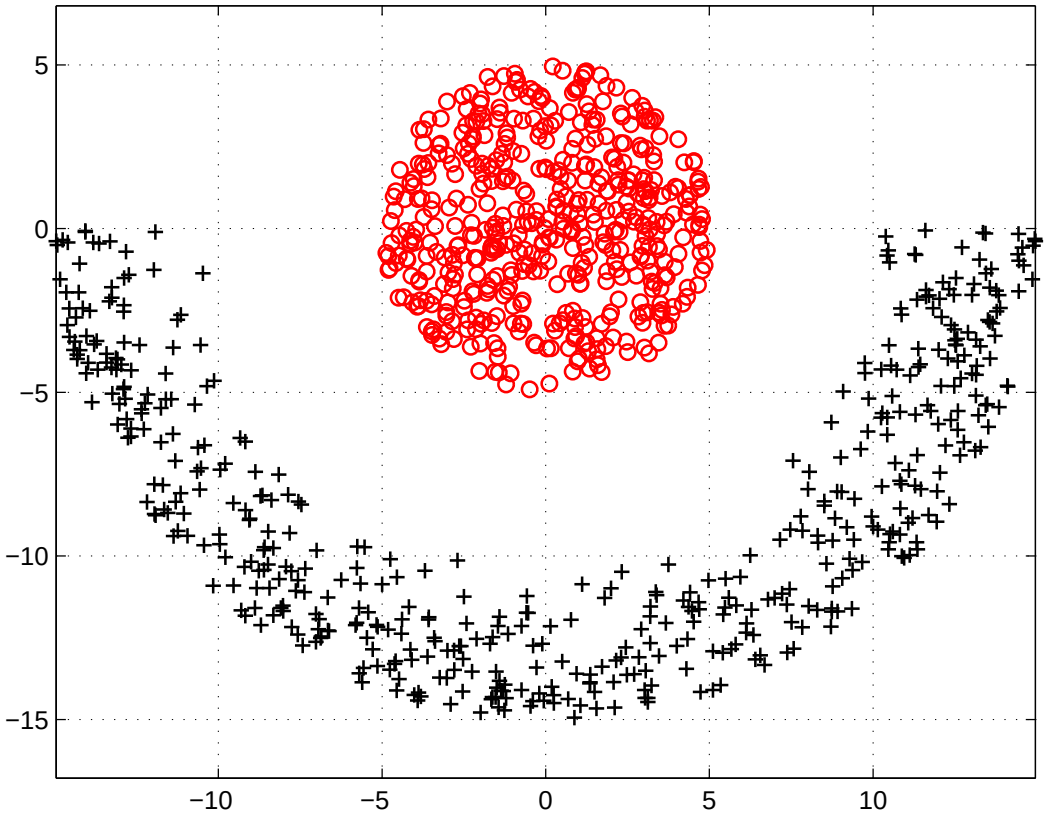


Figura 5.26: *Crescent & Full Moon Dataset*: Histograma para as perturbações em $\hat{\boldsymbol{\mu}}_i, i = 1, 2$

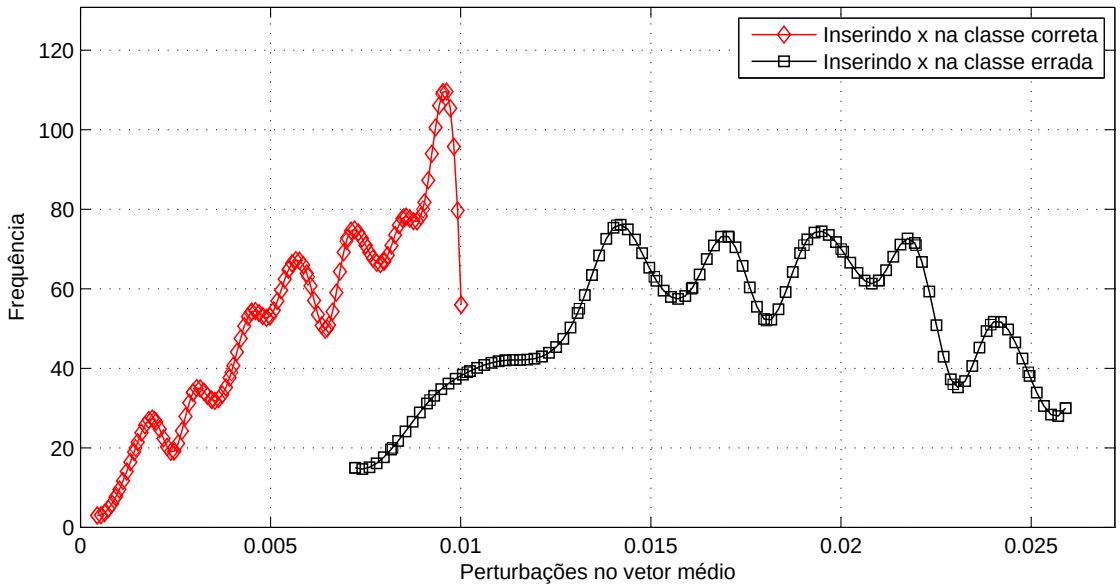


Figura 5.27: *Crescent & Full Moon Dataset*: Histograma para as perturbações em $\hat{\Sigma}_i, i = 1, 2$

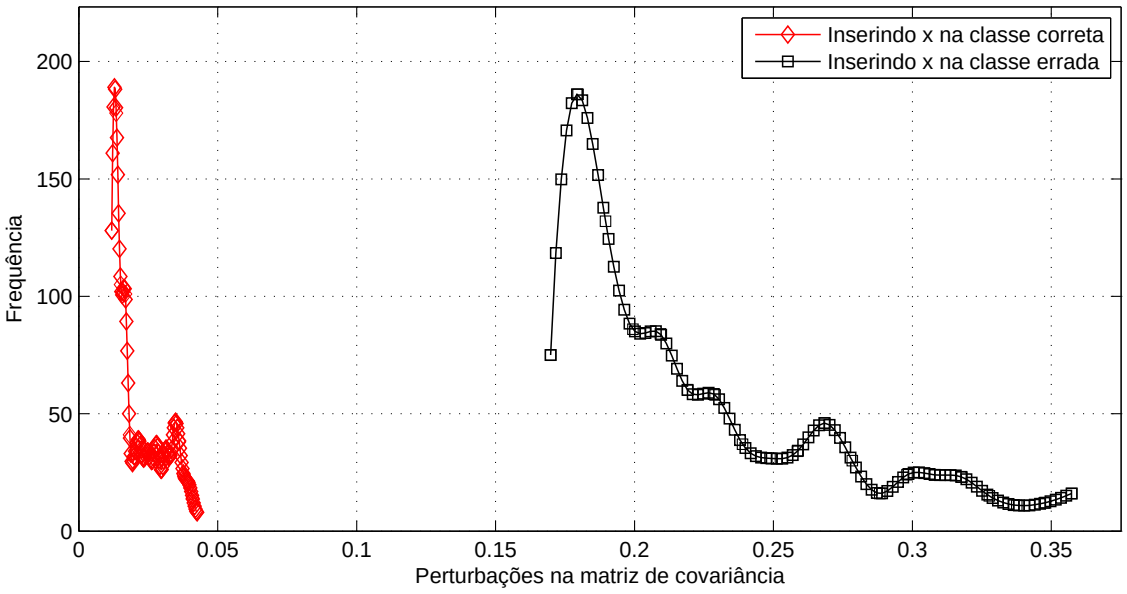


Tabela 5.9: *Crescent & Full Moon Dataset*: Classificação

Classificador	Taxa de acerto(%)
PerC(Mean)	$85,55 \pm 0,29$
PerC(Cov)	$98,16 \pm 0,18$
PerC(Comb)	$87,75 \pm 0,17$
Naïve Bayes	$\mathbf{100,00 \pm 0,00}$
5-NN	$\mathbf{100,00 \pm 0,00}$
SVM(Gauss)	$\mathbf{100,00 \pm 0,00}$
CART	$99,68 \pm 0,10$
MLP	$99,96 \pm 0,30$

5.3.7 Experimentos: *Half-kernel Dataset*

Figura 5.28: *Half-kernel Dataset*: 2 Classes com 500 amostras em cada

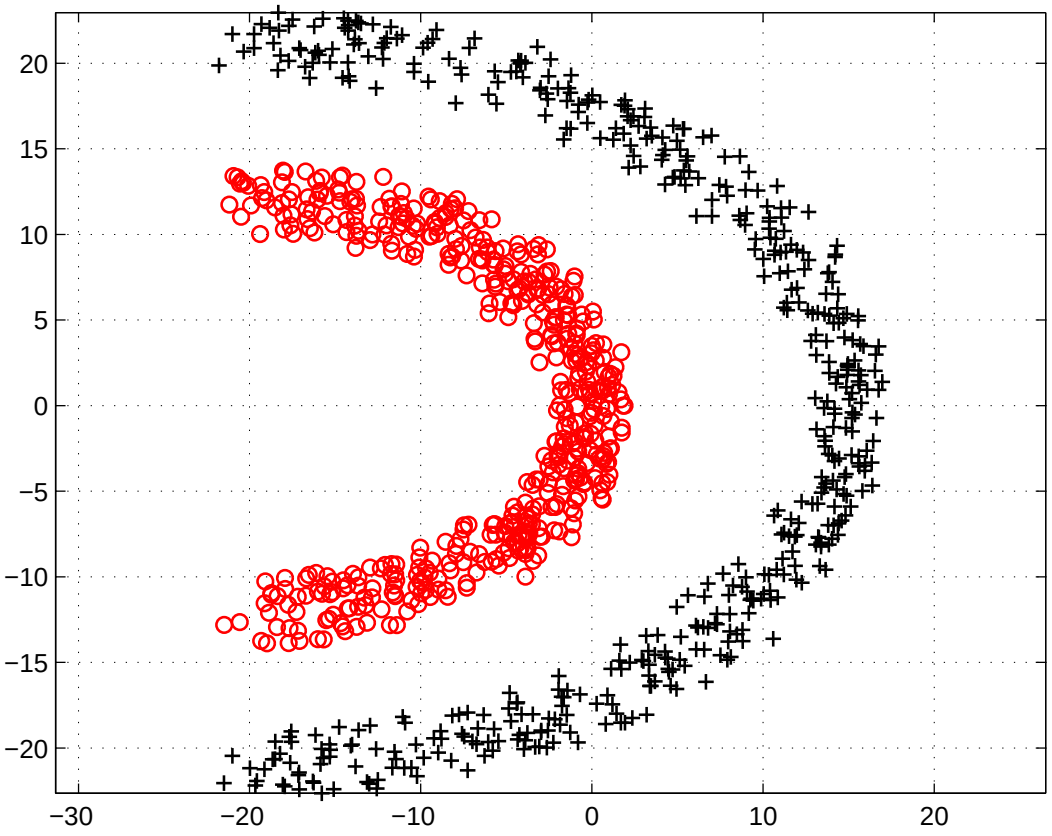


Figura 5.29: *Half-kernel Dataset*: Histograma para as perturbações em $\hat{\boldsymbol{\mu}}_i$, $i = 1, 2$

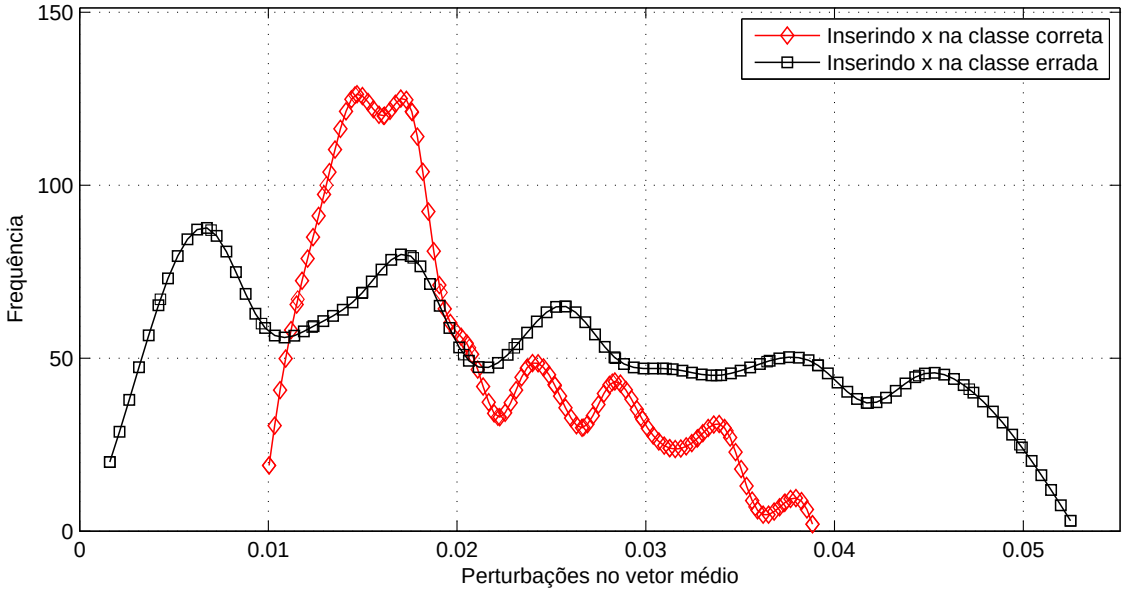


Figura 5.30: *Half-kernel Dataset*: Histograma para as perturbações em $\hat{\Sigma}_i$, $i = 1, 2$

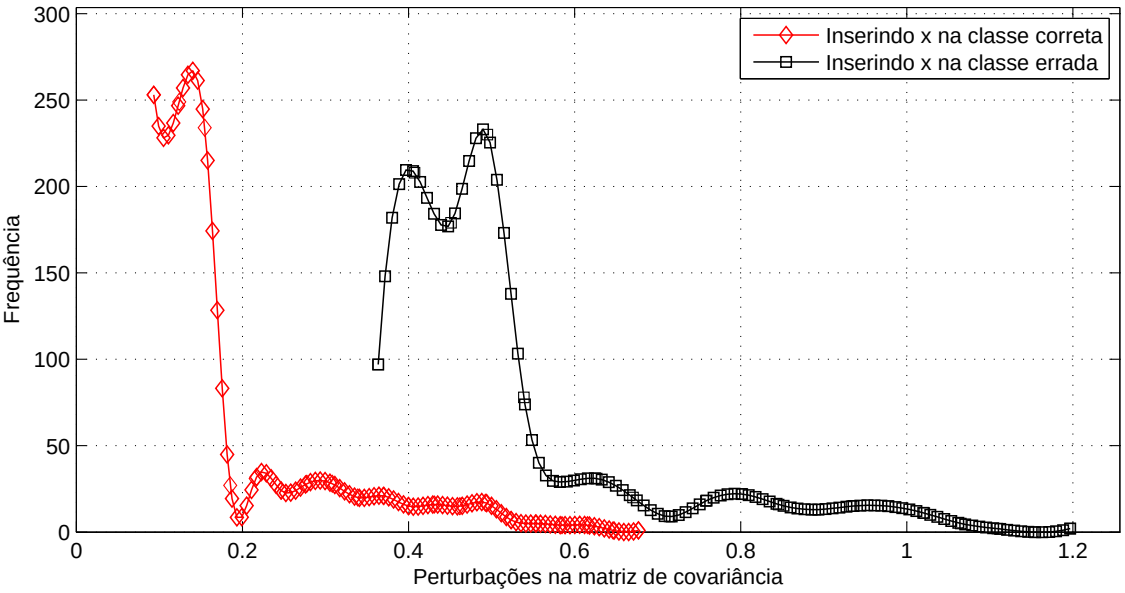


Tabela 5.10: *Half-kernel Dataset*: Classificação

Classificador	Taxa de acerto(%)
PerC(Mean)	$67,42 \pm 0,27$
PerC(Cov)	$88,83 \pm 0,18$
PerC(Comb)	$61,03 \pm 0,42$
Naïve Bayes	$98,58 \pm 0,17$
5-NN	$\mathbf{100,00 \pm 0,00}$
SVM(Gauss)	$\mathbf{100,00 \pm 0,00}$
CART	$99,98 \pm 0,07$
MLP	$99,99 \pm 0,06$

5.3.8 Experimentos: *Outliers Dataset*

Figura 5.31: *Outliers Dataset*: 4 classes com 500 amostras em cada.

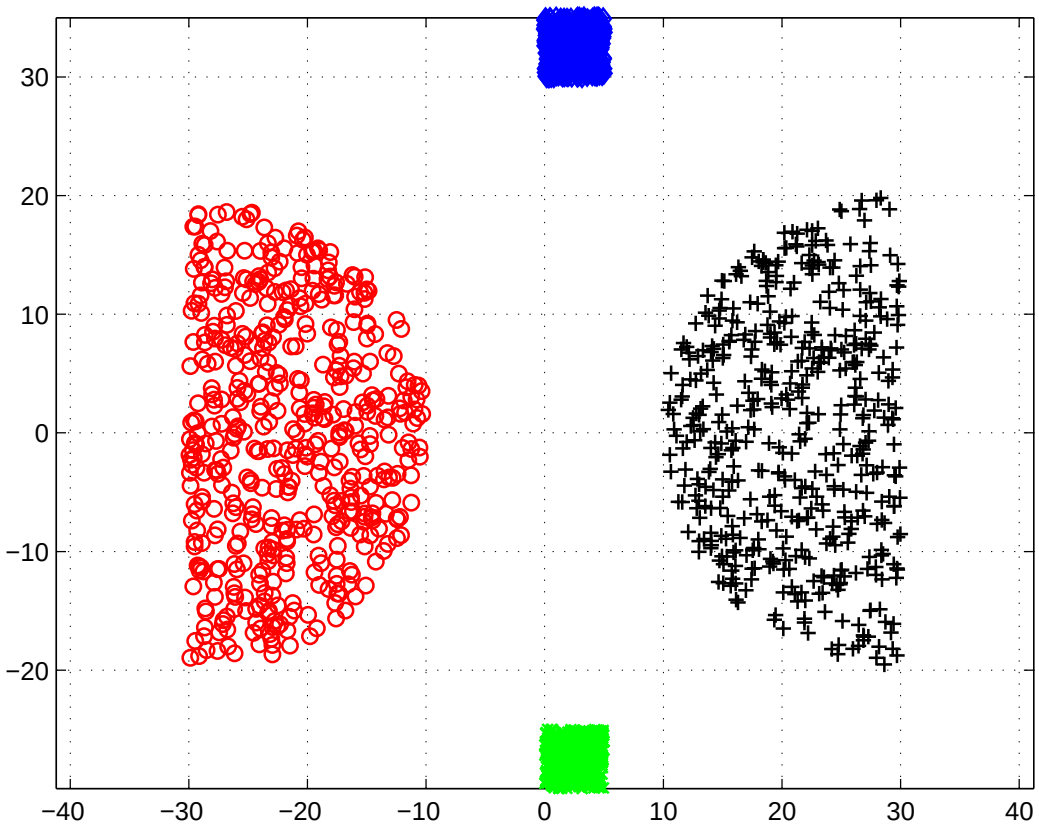


Figura 5.32: *Outliers Dataset*: Histograma para as perturbações em $\hat{\boldsymbol{\mu}}_i$, $i = 1, 2, 3, 4$

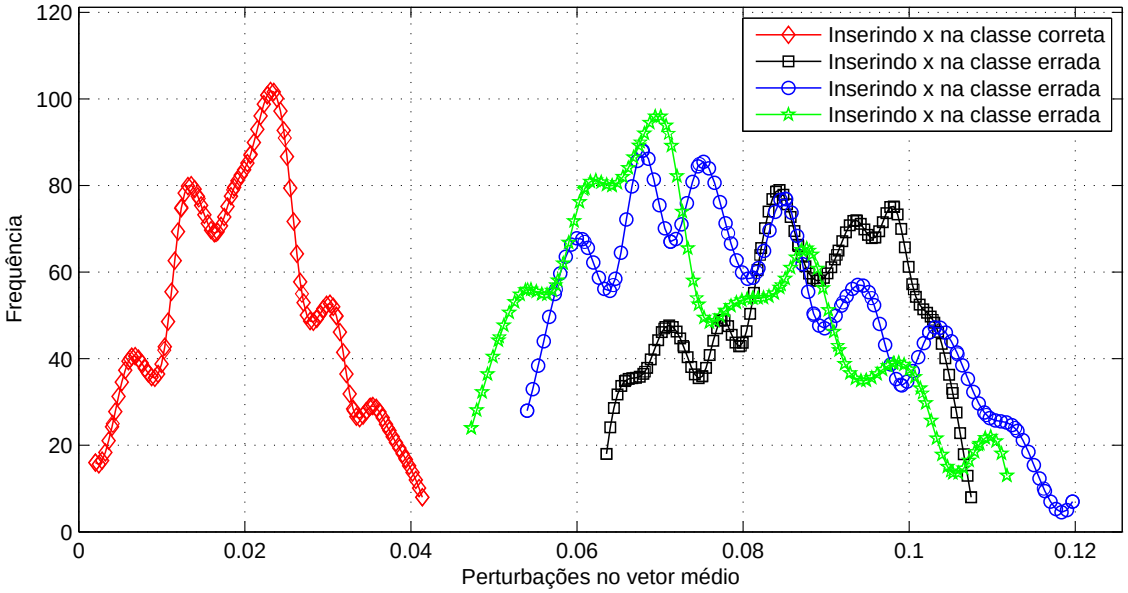
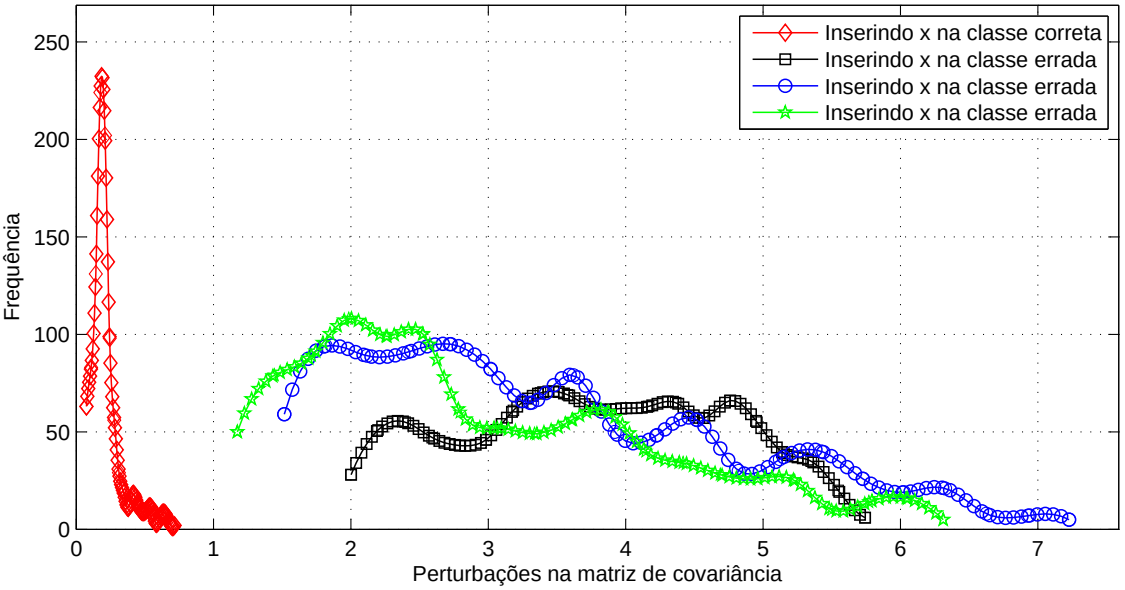


Figura 5.33: *Outliers Dataset*: Histograma para as perturbações em $\hat{\Sigma}_i$, $i = 1, 2$



Nos bancos de dados Crescent & Full Moon, Half-kernel e Outliers (Figuras 5.25, 5.28 e 5.31) são nítidas as distinções entre os histogramas obtidos, tanto para as perturbações em $\hat{\boldsymbol{\mu}}_i$ (Figuras 5.26, 5.29 e 5.32) quanto para as perturbações em $\hat{\Sigma}_i$ (Figuras 5.27, 5.30 e 5.33). Como previsto, as classificações baseadas nestas perturbações confirmam seu poder de discriminação (Tabelas 5.9, 5.10 e 5.11). Entretanto, o *PerC(Comb)* teve sua performance sempre mais próxima do *PerC(Mean)* quando a separabilidade entre as classes dos bancos era simples(*Two Spirals, Cluster in Cluster, Crescent and Full Moon* e *Half Kernel*) e mais próxima ao *PerC(Cov)* quando houve separabilidade complexa (*Banana Set* e *P2 Problem*), indicando a influência das perturbações no poder de discriminação do *PerC(Comb)* em cada uma das situações.

Tabela 5.11: *Outliers Dataset*: Classificação

Classificador	Taxa de acerto(%)
PerC(Mean)	$\mathbf{100,00 \pm 0,00}$
PerC(Cov)	$\mathbf{100,00 \pm 0,00}$
PerC(Comb)	$\mathbf{100,00 \pm 0,00}$
Naïve Bayes	$\mathbf{100,00 \pm 0,00}$
5-NN	$\mathbf{100,00 \pm 0,00}$
SVM(Gauss)	$\mathbf{100,00 \pm 0,00}$
CART	$\mathbf{100,00 \pm 0,00}$
MLP	$98,74 \pm 0,56$

5.4 Experimentos em Dados Reais

Nos experimentos com dados reais, foram utilizados 21 bancos de dados disponíveis no *UCI Repository Learning* (BACHE; LICHMAN, 2013) (Tabela 5.12). A metodologia utilizada foi a mesma realizada nas Subseções 4.2.2 e 4.2.3, em que usou-se o *10-fold cross-validation* e 100 repetições para cada experimento, ao final a média e o desvio padrão das taxas de acerto foram coletadas. Os resultados obtidos para o método proposto, *PerC* (versão combinada)[1], são comparadas com o *kNN* em suas versões para *k=1,3* e *5*, o *Naïve Bayes*, o *SVM* em suas versões para *kernels* polinomiais de graus 1, 2, 3 e *kernel* gaussiano, as *Árvores de Classificação e Regressão* (*CART*) e as *Redes Neurais Multicamadas* (*MLP*) (Tabelas 5.13).

Em 12 das 21 bases avaliadas o *PerC* foi **superior** ou esteve na segunda posição em relação aos métodos comparados. Além disso, quando comparado individualmente com cada um dos classificadores, constata-se que em 13 bases foi superior ao *Naïve Bayes*, em 16 bases (não as mesmas) foi superior ao *3NN* e ao *5NN*, em 17 bases foi superior ao *SVM(d=2)* e em outras 17 bases, superior ao *SVM(d=3)*, em 12 bases foi superior ao *SVM* com *kernel gaussiano*, em 16 bases superior ao *CART* e em outras 13 foi superior ao *MLP*. Estes dados são apresentados na última linha da Tabela 5.13.

O teste de Friedman (DEMSAR, 2006) foi utilizado para verificar a *hipótese nula* de que os classificadores avaliados possuem a mesma performance ou que as diferenças observadas são meramente randômicas. Após organizar-se os resultados numa tabela de *rankings*, chegou-se aos *rankings* médios $R_{nb} = 4,57$ para o *Naïve Bayes*, $R_{3nn} = 5,74$ para o *3NN*, $R_{5nn} = 5,52$ para o *5NN*, $R_{svm2} = 6,14$ para o *SVM(d=2)*, $R_{svm3} = 6,02$ para o *SVM(d=3)*, $R_{svmgauss} = 4,57$ para o *SVM* com *kernel* gaussiano, $R_{cart} = 5,29$ para o *CART*, $R_{mlp} = 3,81$ para o *MLP* e $R_{perc} = 3,33$ para o *PerC*. O *SVM* com *kernel* polinomial de grau 1 e o *kNN(k=1)* foram deixados de fora desta análise por terem sido as versões de pior performance para estes classificadores.

O teste estatístico de Friedman com $k = 9$ e $N = 21$ (9 classificadores e 21 bancos

[1] Os experimentos em dados reais demonstraram que esta versão do *PerC* se sobressai em relação ao *PerC(Mean)* e *PerC(Cov)*. Por este motivo, apenas os resultados obtidos com o *PerC(Comb)* são exibidos.

Tabela 5.12: UCI Datasets Features

Data	dimensão	n^o amostras	n^o classes
Australian	14	690	2
Balance	4	625	3
Banknote	4	1372	2
Climate Model	18	540	2
German	24	1000	2
Haberman	3	306	2
Heart	13	270	2
Indians	8	768	2
Iris	4	150	3
Letter	16	20000	26
Page Blocks	10	5473	5
Relax	12	182	2
Sat	36	6435	6
Spect	22	267	2
Spectf	44	267	2
Tic-Tac-Toe	9	958	2
Transfusion	4	748	2
Vehicle	18	846	4
Vertebral2C	6	310	2
Waveform	40	5000	3
Wine	13	178	3

avaliados) atesta que o valor

$$\chi_F^2 = \frac{12N}{k(k+1)}\left(\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4}\right) = 21,89$$

obtido para os dados dispostos na Tabela 5.13, está distribuído de acordo com uma distribuição χ^2 com $k-1=8$ graus de liberdade. Uma versão melhorada deste teste (IMAN.; DAVENPORT, 1979) estabelece uma correção menos conservadora para χ_F^2 dada por

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2} = 3,00$$

distribuída de acordo com uma distribuição F com $(k-1)=8$ e $(k-1)\times(N-1)=160$ graus de liberdade. O valor crítico para uma distribuição F nestas condições, ou seja $F(8,160)$ com $\alpha = 0,05$ é $1,9967$, ou seja F_F está acima deste valor e tem-se portanto, rejeitada a hipótese nula de que todos classificadores possuam a mesma performance.

Prosseguindo com o teste de Nemenyi para a comparação dois a dois, entre os

Tabela 5.13: Comparando os resultados. Em negrito estão destacados, para cada banco de dados, o melhor resultado obtido. A linha *upper/lower* indica quantas vezes o *PerC(Comb)* obteve performance inferior e superior, em relação a cada um dos demais classificadores, nas bases de dados utilizadas.

	Taxa de Acerto (%)								
Base de Dados	**PerC**	**N. Bayes**	**3NN**	**5NN**	**SVM(d=2)**	**SVM(d=3)**	**SVM(Gauss)**	**CART**	**MLP**
Australian	**84,6 ± 0,3**	80,2 ± 0,3	65,3 ± 0,8	65,3 ± 0,9	61,6 ± 2,8	58,5 ± 2,7	55,5 ± 0,1	83,5 ± 0,9	83,4 ± 0,9
Balance	90,4 ± 0,3	90,5 ± 0,3	79,0 ± 0,5	79,0 ± 0,5	**100,0 ± 0,0**	99,1 ± 0,3	97,8 ± 0,5	77,8 ± 0,8	95,5 ± 2,1
Banknote	98,9 ± 0,1	83,9 ± 0,1	99,9 ± 0,0	99,9 ± 0,0	99,9 ± 0,1	99,8 ± 0,1	**100,0 ± 0,0**	98,3 ± 0,2	99,9 ± 0,5
Climate Model	91,5 ± 0,0	91,0 ± 0,5	88,0 ± 0,6	88,0 ± 0,5	90,5 ± 0,7	90,5 ± 0,6	**93,5 ± 0,4**	88,5 ± 0,9	90,5 ± 0,9
German	71,8 ± 0,3	**72,6 ± 0,6**	66,6 ± 0,5	66,6 ± 0,6	63,2 ± 2,2	58,2 ± 2,4	70,1 ± 0,1	70,3 ± 1,1	71,6 ± 1,2
Haberman	**75,0 ± 0,5**	74,6 ± 0,4	67,9 ± 1,2	68,0 ± 1,1	58,6 ± 5,2	58,5 ± 5,5	70,0 ± 0,6	68,4 ± 1,6	71,8 ± 1,4
Heart	82,3 ± 0,9	**84,0 ± 0,6**	57,7 ± 1,3	57,9 ± 1,3	62,4 ± 2,8	61,0 ± 3,3	55,8 ± 0,2	75,8 ± 1,8	78,9 ± 1,7
Indians	74,4 ± 0,6	**75,6 ± 0,4**	67,9 ± 0,7	67,9 ± 0,7	60,9 ± 3,1	60,6 ± 2,4	65,2 ± 0,1	70,9 ± 1,2	74,7 ± 1,4
Iris	**97,1 ± 0,4**	95,3 ± 0,4	95,9 ± 0,3	95,9 ± 0,3	75,1 ± 5,6	83,5 ± 3,8	93,4 ± 0,9	94,4 ± 1,0	93,5 ± 4,7
Letter	87,7 ± 0,1	64,3 ± 0,1	96,0 ± 0,1	96,0 ± 0,1	92,5 ± 0,1	93,1 ± 0,1	**97,7 ± 0,1**	86,4 ± 0,2	75,4 ± 1,8
Page Blocks	95,4 ± 0,1	90,1 ± 0,2	95,7 ± 0,1	95,7 ± 0,1	76,9 ± 4,7	74,8 ± 5,9	91,4 ± 0,1	**96,5 ± 0,1**	94,9 ± 0,7
Relax	70,6 ± 0,7	65,5 ± 1,8	63,6 ± 1,5	63,5 ± 1,6	62,5 ± 1,9	56,1 ± 2,6	**71,8 ± 0,5**	57,9 ± 3,4	57,8 ± 4,4
Sat	83,7 ± 0,1	79,6 ± 0,0	90,7 ± 0,1	90,7 ± 0,1	67,3 ± 1,4	**90,7 ± 0,8**	23,4 ± 0,4 [a]	85,9 ± 0,3	88,5 ± 2,1
Spect	65,6 ± 1,6	67,8 ± 0,7	60,0 ± 1,4	59,7 ± 1,4	63,9 ± 1,7	65,2 ± 1,5	**70,4 ± 1,6**	66,6 ± 1,8	65,9 ± 2,3
Spectf	79,4 ± 0,0	67,9 ± 0,9	74,3 ± 1,0	74,4 ± 1,1	76,3 ± 1,5	76,9 ± 1,6	**79,4 ± 0,0**	73,7 ± 1,9	76,8 ± 1,7
Tic-Tac-Toe	67,8 ± 0,3	72,6 ± 0,5	63,8 ± 0,3	63,7 ± 0,2	97,2 ± 1,4	97,2 ± 0,4	**98,0 ± 0,4**	92,7 ± 0,7	82,2 ± 3,0
Transfusion	77,9 ± 0,2	75,1 ± 0,2	68,7 ± 0,8	68,7 ± 0,6	60,4 ± 6,3	63,7 ± 7,1	74,7 ± 0,3	74,4 ± 0,8	**78,7 ± 0,6**
Vehicle	**84,3 ± 0,5**	45,7 ± 0,7	64,9 ± 0,6	65,1 ± 0,6	75,3 ± 1,3	75,7 ± 1,3	22,3 ± 1,0 [a]	70,8 ± 1,2	82,3 ± 1,1
Vertebral2C	71,3 ± 0,7	77,7 ± 0,3	83,5 ± 0,9	**83,7 ± 0,9**	52,3 ± 5,0	51,6 ± 5,4	67,4 ± 0,3	80,1 ± 1,4	83,2 ± 3,6
Waveform	83,6 ± 0,2	80,0 ± 0,1	76,8 ± 0,2	76,8 ± 0,3	81,0 ± 0,3	81,9 ± 0,3	**84,8 ± 0,2**	74,7 ± 0,5	83,5 ± 2,0
Wine	96,8 ± 0,7	**97,3 ± 0,6**	76,2 ± 1,2	76,3 ± 1,4	91,7 ± 3,1	90,3 ± 3,4	44,7 ± 0,8 [a]	90,5 ± 1,3	96,2 ± 3,9
Mean	**82,4 ± 0,4**	77,7 ± 0,4	76,3 ± 0,7	76,3 ± 0,7	74,7 ± 2,4	75,6 ± 0,5	72,7 ± 0,4	79,9 ± 1,1	82,1 ± 2,0
Friedman Rank	**3.33**	4.57	5.74	5.52	6.14	6.02	4.57	5.29	3.81
upper/lower	–	8/13	5/16	5/16	4/17	5/16	9/12	5/16	8/13

[a] Para confirmar estas taxas obtidas para o *SVM(Gauss)*, os experimentos para os bancos de dados *Sat*, *Vehicle* e *Wine*, foram realizados duas vezes, usando as bibliotecas diferentes.

classificadores, o valor crítico para se realizar os testes é dado por

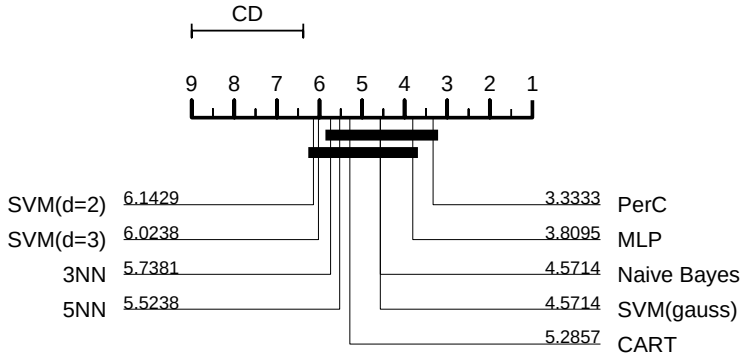
$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}} = 2,62$$

sendo $q_\alpha = 3,102$ o valor crítico estabelecido para $k = 9$. Calculando as diferenças entre os rankings obtidos anteriormente para os classificadores comparados, tem-se

$$\begin{aligned} R_{nb} - R_{perc} &= 1,24 < CD \\ R_{3nn} - R_{perc} &= 2,41 < CD \\ R_{5nn} - R_{perc} &= 2,19 < CD \\ \boldsymbol{R_{svm2} - R_{perc} = 2,81 > CD} \\ \boldsymbol{R_{svm3} - R_{perc} = 2,69 > CD} \\ R_{svmgauss} - R_{perc} &= 1,24 < CD \\ R_{cart} - R_{perc} &= 1,96 < CD \\ R_{mlp} - R_{perc} &= 0,48 < CD \end{aligned}$$

constatando-se, portanto, que o classificador *PerC* possui performance significativamente superior às versões do *SVM* com *kernels* polinomiais de graus 2 e 3, e praticamente equivalente aos demais classificadores, tendo o *PerC* o menor *ranking* (Fig. 5.34).

Figura 5.34: Comparativo entre os classificadores, segundo o Teste de Friedman. $CD = 2,62$.



Destaca-se ainda que a taxa média de acerto, apresentada pelo *PerC*, 82,4%, é sensivelmente superior aos valores encontrados para esta mesma medida para o *Naïve Bayes*, 77,7%, *kNN(k=3)*, 76,3%, *kNN(k=5)*, 76,3%, *SVM* com *kernel* polinomial de grau 2, 74,7%, *SVM* com *kernel* polinomial de grau 3, 75,6%, *SVM* com *kernel* gaussiano, 72,7 e o *CART*, 79,9% (Tabela 5.13).

6

CONSIDERAÇÕES FINAIS

A classificação é em geral a última etapa no processo de reconhecimento de padrões. Muitas abordagens têm sido utilizadas e entre estas destaca-se o **Classificador de Bayes** que, apesar de teórico apresenta-se como modelo para a construção de classificadores de uso prático e eficiente.

O classificador de Bayes fundamenta-se no conhecimento prévio dos vetores médios e das matrizes de covariância de cada uma das classes existentes nos dados de treinamento. Em geral, estes parâmetros são desconhecidos e as estimativas de máxima verossimilhança são as alternativas mais comumente utilizadas.

Quando um novo vetor de teste é inserido numa das classes, o vetor médio e matriz de covariância estimados para esta classe, $\hat{\boldsymbol{\mu}}_i$ e $\hat{\Sigma} i$, serão perturbados.

Neste trabalho apresentou-se, Capítulo 4, uma nova abordagem baseada nas perturbações das estimativas dos vetores médio e matrizes de covariância, $\Delta\hat{\boldsymbol{\mu}}_i$ e $\Delta\hat{\Sigma}_i$, de todas as classes existentes nos dados de treinamento, ocasionadas pela possível inserção do vetor de teste nas mesmas. Utilizando-se de uma forma eficiente de calcular tais perturbações (Equações 4.11 e 4.20), inicialmente certificou-se a validade desta abordagem através de um exemplo simples.

Ainda na Subseção 4.1.1, foram gerados bancos de dados sintéticos, com duas classes e distribuição gaussiana e a partir de um banco de teste contendo 1000 amostras de uma única classe, escolhida aleatoriamente, foram comparados o histograma das alterações ocorridas quando estes vetores de teste são inseridos na classe a qual pertencem, a *classe correta*, com o histograma obtido das alterações ocorridas na inserção destes mesmos vetores na outra classe, a *classe incorreta*. Novamente constatou-se que as inserções na classe correta se acumulam em torno de valores menores que aqueles gerados para as inserções na classe incorreta.

Ampliando o espectro dos testes para dados gaussianos com 2, 3, 4 e 5 classes e seguindo a mesma metodologia, Subseção 5.2, outra vez confirmou-se que as inserções na classe correta em geral se acumulam em torno de valores menores do que os obtidos nas inserções nas classes erradas, sugerindo que a quantidade de classes não tem influência sobre o poder da classificação baseada em perturbações.

Continuando os testes preliminares em dados sintéticos, Subseção 5.3, resultados

semelhantes foram obtidos para dados que não possuem distribuição gaussiana. Os resultados obtidos nos bancos de dados *Banana set*, *Cluster in cluster*, *Corners*, *Crescent & full moon*, *Half-kernel* e *Outliers*, evidenciam que a normalidade dos dados também não possui influência no poder de classificação baseada na abordagem proposta. Entretanto, os resultados obtidos sobre os dados *P2 Problem* e *Two Spirals*, sugerem que a separabilidade entre as classes, influem de algum modo no processo de classificação baseado na nossa abordagem.

Com base na abordagem proposta, desenvolvemos dois classificadores simples, *PerC(Mean)*, Eq. (4.21) e *PerC(Cov)*, Eq. (4.22), que utilizam exclusivamente as perturbações no vetor médio e na matriz de covariância, respectivamente. Comparamos em vários cenários, a performance destes classificadores com os classificadores, *kNN*, n=1,3 e 5, *Naïve Bayes* e *SVM* com kernels polinomiais de grau 1, 2 e 3 e kernel gaussiano, as *Árvores de classificação e regressão CART* e as *Redes neurais multicamadas MLP*. Nestes testes, utilizamos dados sintéticos bidimensionais com distribuições gaussianas e validação a partir do *10-fold cross validation*. Após repetir o teste por 100 vezes e coletar a média e o desvio padrão da taxa de acerto, verificou-se que a performance do *PerC(Mean)* e *PerC(Cov)* é semelhante aos demais classificadores.

Nos cenários em que há separabilidade complexa entre as classes (distância zero entre os vetores médios das classes) constatou-se que apesar da baixa performance do *PerC(Mean)* e *PerC(Cov)*, resultados idênticos foram encontrados para os demais classificadores. (Figuras 4.6, 4.7, 4.8, 4.9 e 4.10).

Combinando o poder de classificação de ambas as perturbações foi desenvolvido um terceiro classificador, denominado *PerC(Comb)* e realizando os mesmos testes, verificou-se performance superior ao *PerC(Mean)* e *PerC(Cov)* e novamente, performance semelhante aos demais classificadores.

Na Seção 5.4, os mesmos testes foram realizados sobre 21 bancos de dados do *UCI Repository Learning* e novamente comparou-se o *PerC(Comb)* com os mesmos classificadores. Através do método estatístico de *Friedman*, foi comprovada a validade da abordagem proposta, além de ter-se confirmado performance do *PerC(Comb)*, significativamente superior às versões do *SVM* com kernels polinomiais de graus 2 e 3, e praticamente equivalente aos demais classificador e, tendo o *PerC(Comb)* o maior ranking segundo o mesmo teste.

Face ao exposto, este trabalho apresenta como contribuições à área em que se insere, a abordagem baseada em perturbações e seu foco voltado para a classificação bem como o classificador *PerC* construído a partir dela.

6.1 Trabalhos Futuros

- Nesta proposta, utilizou-se a norma euclidiana para se avaliar as alterações ocorridas na matriz de covariância. Esta matriz em geral é semi-definida positiva, chegando a ser, em alguns casos, positiva definida. Existem métricas especificas para matrizes

positiva definidas. Adaptar uma métrica para este tipo de matriz e usá-la para o cálculo mais preciso das pertubações necessárias para o *PerC*.

- Os autovalores e autovetores em geral são utilizados como parâmetros para avaliar comportamentos da matriz a qual pertencem. A inserção de um vetor de teste numa determinada classe, altera a matriz de covariância e consequentemente seus autovalores e autovetores. As perturbações geradas em tais autovetores e autovalores, podem ser exploradas como medidas de perturbação da classe e a partir delas, uma nova regra de decisão ser criada.

- Usar a abordagem baseada em perturbações para a construção de um classificador voltado para *One-class classification*(KHAN; MADDEN, 2004). Utilizando o *leave-one-out* para estimar o vetor médio e matriz de covariância da classe e partir destes, avaliar perturbações geradas para a possível inserção de vetores de teste, adaptar o *PerC* para este tipo de classificação.

- Explorar outras combinações entre as perturbações utlizadas.

- Criar um metaclassificador baseado nos resultados obtidos e extrair uma regra através do classificador C4.5 que explique a eficiência e ineficiência do *PerC*.

REFERÊNCIAS

ACHIESER, N. I. *Theory of approximation*. [S.l.]: Courier Corporation, 2013.

ADE, R. R.; DESHMUKH, P. R. Methods for incremental learning: A survey. *International Journal of Data Mining & Knowledge Management Process*, v. 3, n. 4, p. 119–125, July 2013.

AHA, D. *Lazy learning*. [S.l.]: Springer Science & Business Media, 2013.

AN, A.; CERCONE, N. Discretization of continuous attributes for learning classification rules. In: *Methodologies for Knowledge Discovery and Data Mining*. [S.l.]: Springer, 1999. p. 509–514.

AN, A.; CERCONE, N. Rule quality measures improve the accuracy of rule induction: An experimental approach. *Lecture notes in computer science*, Springer, v. 1932, p. 119–129, 2000.

BACHE, K.; LICHMAN, M. *UCI Machine Learning Repository*. 2013.

BISHOP, C. M. Neural networks and their applications. *Review of scientific instruments*, AIP Publishing, v. 65, n. 6, p. 1803–1832, 1994.

BISHOP, C. M. *Neural networks for pattern recognition*. [S.l.]: Oxford university press, 1995.

BISHOP, C. M. *Pattern recognition and machine learning*. New York, NY: Springer, 2006.

BLUM, A. *On-line algorithms in machine learning*. [S.l.]: Springer, 1998.

BOUCKAERT, R. R. Naive bayes classifiers that perform well with continuous variables. In: *AI 2004: Advances in Artificial Intelligence*. [S.l.]: Springer, 2005. p. 1089–1094.

BREIMAN, L. et al. *Classification and Regression Trees*. Boca Raton: CRC Press, 1984.

BRIGHTON, H.; MELLISH, C. Advances in instance selection for instance-based learning algorithms. *Data mining and knowledge discovery*, Springer, v. 6, n. 2, p. 153–172, 2002.

BURGES, C. J. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, Springer, v. 2, n. 2, p. 955–974, 1998.

CHAPELLE, O.; SCHÖLKOPF, B.; ZIEN, A. *Semi-supervised learning*. Cambridge, Massachusetts: MIT Press Cambridge, 2006.

CHEN, H. et al. Application of support vector machine learning to leak detection and location in pipelines. In: *IEEE Conference on Instrumentation and Measurement Technology*. [S.l.: s.n.], 2004. v. 3, p. 2273–2277.

CHENG, L. et al. Implicit online learning with kernels. *Advances in neural information processing systems*, v. 19, p. 249–256, 2007.

COHEN, W. W. Fast effective rule induction. In: *International Conference on Machine Learning*. [S.l.: s.n.], 1995. p. 115–123.

COVER, T. M.; HART, P. E. Nearest neighbor pattern calssification. *IEEE Transactions on Information Theory*, I, p. 21–27, 1967.

CYBENKO, G. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, Springer, v. 2, n. 4, p. 303–314, 1989.

DEMSAR, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, v. 07, p. 1–30, December 2006.

DEVIJVER, P. A.; KITTLER, J. *Pattern recognition: A statistical approach*. Englewood Cliffs: Prentice-Hall London, 1982. v. 761.

DEVROYE, L.; GYÖRFI, L.; LUGOSI, G. *A Probabilistic Theory of Pattern Recognition*. Berlin: Springer Science & Business Media, 1996.

DOMINGOS, P.; PAZZANI, M. On the optimality of the simple bayesian classifier under zero-one loss. *Machine learning*, Springer, v. 29, n. 2-3, p. 103–130, 1997.

DUAN, K.-B.; KEERTHI, S. S. Which is the best multiclass svm method? an empirical study. In: ____. *Multiple Classifier Systems: International Workshop, Seaside, CA, USA, June 13-15, 2005. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p. 278–285.

DUDA, R. O.; HART, P. E. et al. *Pattern classification and scene analysis*. New York: Wiley New York, 1973.

DUDA, R. O.; HART, P. E.; STORK, D. G. *Pattern classification*. New York: John Wiley & Sons, 2012.

FODOR, I. K. *A survey of dimension reduction techniques*. [S.l.]: Technical Report UCRL-ID-148494, Lawrence Livermore National Laboratory, 2002.

FRANK, E.; WITTEN, I. H. Generating accurate rule sets without global optimization. In: *International Conference on Machine Learning*. San Francisco, CA: [s.n.], 1998.

FRIEDMAN, N.; GEIGER, D.; GOLDSZMIDT, M. Bayesian network classifiers. *Machine Learning*, v. 29, n. 2-3, p. 131–163, 1997.

FU, K. S. *Syntactic Pattern Recognition and Applications*. Englewood Cliffs, NJ: Prentice Hall, 1982.

FUKUNAGA, K. *Introduction to Statistical Pattern Recognition*. 1. ed. Orlando, FL: Academic Press, 1972.

FÜRNKRANZ, J. Pruning algorithms for rule learning. *Machine Learning*, Springer, v. 27, n. 2, p. 139–172, 1997.

FÜRNKRANZ, J. Separate-and-conquer rule learning. *Artificial Intelligence Review*, Springer, v. 13, n. 1, p. 3–54, 1999.

FÜRNKRANZ, J. Round robin rule learning. In: CITESEER. *Proceedings of the 18th International Conference on Machine Learning: 146–153*. [S.l.], 2001.

FÜRNKRANZ, J.; FLACH, P. A. Roc 'n' rule learning - towards a better understanding of covering algorithms. *Machine Learning*, Springer, v. 58, n. 1, p. 39–77, 2005.

GAMA, J.; BRAZDIL, P. Linear tree. *Intelligent Data Analysis*, Elsevier, v. 3, n. 1, p. 1–22, 1999.

GEHRKE, J.; RAMAKRISHNAN, R.; GANTI, V. Rainforest a framework for fast decision tree construction of large datasets. *Data Mining and Knowledge Discovery*, Springer, v. 4, n. 2-3, p. 127–162, 2000.

GENG, X.; SMITH-MILES, K. Incremental learning. In: LI, S.; JAIN, A. (Ed.). *Encyclopedia of Biometrics*. [S.l.]: Springer US, 2009. p. 731–735.

GENTON, M. G. Classes of kernels for machine learning: a statistics perspective. *The Journal of Machine Learning Research*, v. 2, p. 299–312, 2002.

GOLDBERG, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*. 1st. ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989.

GOOD, I. J. *Probability and the Weighing of Evidence*. London: Charles Griffin, 1950.

GOOD, I. J. Maximum entropy for hypothesis formulation, especially for multidimensional contingency tables. *The Annals of Mathematical Statistics*, v. 34, p. 911–934, 1963.

HAND, D. J.; YU, K. Idiot's bayes: Not so stupid after all? *International Statistical Review*, v. 69, n. 3, p. 385–398, Dec. 2001.

HOFF, P. D. *A first Course in Bayesian Statistical Methods*. Seattle, USA: Springer Science & Business Media, 2009.

HOFFBECK, J. P.; LANDGREBE, D. A. Covariance matrix estimation and classification with limited training data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Washington, DC, USA, v. 18, n. 7, p. 763–767, jul. 1996.

HORNIK, K. Approximation capabilities of multilayer feedforward networks. *Neural networks*, v. 4, n. 2, p. 251–257, 1991.

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

HUO, Q.; LEE, C.-H. On-line adaptive learning of the continuous density hidden markov model based on approximate recursive bayes estimate. *IEEE transactions on Speech and audio processing*, v. 5, n. 2, p. 161–172, 1997.

IMAN., R.; DAVENPORT, J. *Approximations of the Critical Region of the Friedman Statistic*. Albuquerque, NM, USA, 1979.

IOSIFIDIS, A.; TEFAS, A.; PITAS, I. On the optimal class representation in linear discriminant analysis. *IEEE Transactions on Neural Networks and Learning Systems*, v. 24, n. 9, p. 1491–1497, 2013.

JAIN, A. K.; DUIN, R. P. W.; MAO, J. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 22, n. 1, p. 4–37, 2000.

JAIN, A. K.; MAO, J.; MOHIUDDIN, K. M. Artificial neural networks: A tutorial. *Computer*, n. 3, p. 31–44, 1996.

JAYNES, E. T. Prior probabilities. *IEEE Transactions on Systems Science and Cybernetics*, IEEE, v. 4, n. 3, p. 227–241, September 1968.

JAYNES, E. T. *Probability Theory: The Logic of Science*. Cambridge: Cambridge university press, 2003.

JENSEN, F. V. *An introduction to Bayesian networks*. 1st. ed. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1996.

JOHNSON, R. A.; WICHERN, D. W. *Applied multivariate stastistical analysis*. 6th. ed. EUA: Prentice-Hall of India Private Limited, 2007.

KALLENBERG, O. *Foundations of modern probability*. Berlin: Springer Science & Business Media, 2002.

KHAN, S. S.; MADDEN, M. G. One-class classification: Taxonomy of study ans review of techniques. *The Knowledge Engineering Review*, v. 1, n. 4, p. 1–35, July 2004.

KIVINEN, J.; SMOLA, A. J.; WILLIAMSON, R. C. Online learning with kernels. *IEEE Transactions on Signal Processing*, v. 52, n. 8, p. 2165–2176, August 2004.

KOHONEN, T. The self-organizing map. *Neurocomputing*, Elsevier, v. 21, n. 1, p. 1–6, 1998.

KONONENKO, I. Estimating attributes: analysis and extensions of relief. In: SPRINGER. *European conference on machine learning*. [S.l.], 1994. p. 171–182.

KOTSIANTIS, S. B. Supervised machine learning: A review of classification techniques. *Informatica*, v. 31, p. 249–268, 2007.

KUBAT, M.; JR, M. C. A reduction technique for nearest-neighbor classification: Small groups of examples. *Intelligent Data Analysis*, IOS Press, v. 5, n. 6, p. 463–476, 2001.

KUNCHEVA, L. I. *Combining pattern classifiers: methods and algorithms*. Hoboken, New Jersey: John Wiley & Sons, 2004.

KUO, B.-C.; LANDGREBE, D. A. A Covariance Estimator for Small Sample Size Classification Problems and Its Application to Feature Extraction. *IEEE Transactions on Geoscience and Remote Sensing*, v. 40, n. 4, p. 814–819, 2002.

LAVESSON, N. *Evaluation and Analysis of Supervised Learning Algorithms and Classifiers*. Sweden: Blekinge Institute of Technology, 2006.

LEDOIT, O.; WOLF, M. A well-conditioned estimator for large-dimensional covariance matrices. *Jornal of Multivariate Analysis*, v. 88, p. 365–411, 2004.

LIM, T.-S.; LOH, W.-Y.; SHIH, Y.-S. A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine learning*, Springer, v. 40, n. 3, p. 203–228, 2000.

LINDGREN, T. Methods for rule conflict resolution. *Lecture Notes in Computer Science*, Berlin:Springer, v. 3201, p. 262–237, 2004.

LÜTZ, A.; RODNER, E.; DENZLER, J. Efficient multi-class incremental learning using gaussian processes. In: *Open German-Russian Workshop on Pattern Recognition and Image Understanding*. [S.l.: s.n.], 2011. p. 182–185.

LÜTZ, A.; RODNER, E.; DENZLER, J. I want to know more - efficient multi-class incremental learning using gaussian processes. *Pattern Recognition and Image Analysis*, v. 23, n. 3, p. 402–407, 2013.

MINSKY, M.; PAPERT, S. *Perceptrons: An Introduction to Computational Geometry*. Cambridge, MA: MIT Press, Cambridge, MA, 1969.

MITCHELL, T. M. *Machine Learning*. [S.l.]: McGraw-Hill Boston, 1997. ISBN 0070428077.

MONTGOMERY, D. C.; RUNGER, G. C. *Applied statistics and probability for engineers*. [S.l.]: John Wiley & Sons, 2010.

MURPHY, K. P. *Machine learning: a probabilistic perspective*. [S.l.]: MIT press, 2012.

MURTHY, S. K. Automatic construction of decision trees from data: A multi-disciplinary survey. *Data mining and knowledge discovery*, Kluwer Academic Publishers, v. 2, n. 4, p. 345–389, 1998.

NILSSON, N. J. *Learning Machines.* New York: McGraw-Hill, 1962.

OSUNA, E.; FREUND, R.; GIROSI, F. An improved training algorithm for support vector machines. In: *IEEE Workshop on Neural Networks for Signal Processing VII*. [S.l.: s.n.], 1997. p. 276–285.

PAN, S. J.; KWOK, J. T.; YANG, Q. Transfer learning via dimensionality reduction. In: *AAAI Conference on Artificial Intelligence*. Chicago, IL: [s.n.], 2008. v. 8, p. 677–682.

PATRICK, E. A. *Fundamentals of pattern recognition*. Englewood Cliffs, New Jersey: Prentice-Hall, 1972.

PERRON, F. Minimax estimators of a covariance matrix. *Journal of Multivariate Analysis*, v. 43, n. 1, p. 16–28, out. 1992. ISSN 0047259X.

POMERLEAU, D. A. *Alvinn: An autonomous land vehicle in a neural network*. Pittsburgh, PA, USA, 1989.

QUINLAN, J. R. *C4.5: Programs for Machine Learning*. [S.l.]: Morgan Kaufmann Publishers Inc., 1993. v. 1.

RENCHER, A. C. *Methods of multivariate analysis*. EUA: John Wiley & Sons, 2003.

RICHARD, M. D.; LIPPMANN, R. P. Neural network classifiers estimate bayesian a posteriori probabilities. *Neural computation*, MIT Press, v. 3, n. 4, p. 461–483, 1991.

ROSENBLATT, F. *The Perceptron–a perceiving and recognizing automaton*. [S.l.], 1957.

ROSENBLATT, F. *Principles of neurodynamics. Perceptrons and the theory of brain mechanisms*. Washington DC: Spartan Books, 1962.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning internal representations by error propagation. *Parallel Distributed Processing*, n. 1, p. 318–362, 1986.

RUSSELL, S.; NORVIG, P. *Artificial intelligence: a modern approach*. 2. ed. EUA: Prentice Hall Series in Artificial Intelligence, 2003.

SCHÖLKOPF, B.; BURGES, C. J. C.; SMOLA, A. J. *Advances in kernel methods: support vector learning*. [S.l.]: MIT press, 1999.

SEARLE, S. R. *Matrix Algebra Useful for Statistics*. [S.l.]: John Wiley & Sons, 1982.

SEBESTYEN, G. S. *Decision-making processes in pattern recognition*. New York: Macmillan Publishing Co., Inc., 1962.

SHANNON, C. E. *A mathematical theory of communication*. [S.l.: s.n.], 1948.

SUTTON, R. S.; BARTO, A. G. *Reinforcement learning: An introduction*. Cambridge, Massachusetts: MIT press Cambridge, 1998. v. 1.

TADJUDIN, S.; LANDGREBE, D. A. Covariance Estimation with Limited Training Samples. *IEEE Transactions on Geoscience and Remote Sensing Sensing*, v. 37, n. 4, p. 2113–2118, 1999.

TAN, Y.; ZHANG, G. The application of machine learning algorithm in underwriting process. *International Conference on Machine Learning and Cybernetics*, IEEE, v. 6, p. 3523–3527, 2005.

TEICHMAN, A.; THRUN, S. Tracking-based semi-supervised learning. *The International Journal of Robotics Research*, SAGE Publications, v. 31, n. 7, p. 804–818, 2012.

THEODORIDIS, S.; KOUTROUMBAS, K. *Pattern Recognition*. 4th. ed. California: Academic Press, 2008.

TOU, J. T.; GONZALEZ, R. C. *Pattern recognition principles*. Reading, MA: Addison-Wesley, 1974.

VAPNIK, V. *The nature of statistical learning theory*. New York, EUA: Springer Science & Business Media, 1995.

VAPNIK, V. *Statistical learning theory*. New York, EUA: Wiley New York, 1998. v. 1.

WANG, F.; SUN, J. Survey on distance metric learning and dimensionality reduction in data mining. *Data Mining and Knowledge Discovery*, Springer, v. 29, n. 2, p. 534–564, 2014.

WEBB, A. R.; COPSEY, K. D. *Statistical pattern recognition*. [S.l.]: John Wiley & Sons, 2011.

WETTSCHERECK, D.; AHA, D. W.; MOHRI, T. A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. *Artificial Intelligence Review*, Springer, v. 11, n. 1-5, p. 273–314, 1997.

WILSON, D. R.; MARTINEZ, T. R. Reduction techniques for instance-based learning algorithms. *Machine Learning*, Springer, v. 38, n. 3, p. 257–286, 2000.

WU, W. B.; XIAO, H. Covariance matrix estimation in time series. In: RAO, S. S. R. T. S.; RAO, C. (Ed.). *Time Series Analysis: Methods and Applications*. [S.l.]: Elsevier, 2012, (Handbook of Statistics, v. 30). p. 187 – 209.

YANG, Y.; WEBB, G. I. On why discretization works for naive-bayes classifiers. In: *Advances in Artificial Intelligence*. [S.l.]: Springer, 2003. p. 440–452.

YU, L.; LIU, H. Efficient feature selection via analysis of relevance and redundancy. *The Journal of Machine Learning Research*, JMLR. org, v. 5, p. 1205–1224, 2004.

ZHANG, G. P. Neural networks for classification: a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, IEEE, v. 30, n. 4, p. 451–462, 2000.

ZHENG, Z. Constructing conjunctions using systematic search on decision trees. *Knowledge-Based Systems Journal*, Elsevier, v. 10, n. 7, p. 421–430, 1998.