Mayra Santana Siqueira

Classificação binária de padrões: Estudo comparativo entre modelos de Programação Matemática

Campos dos Goytacazes/RJ

2014

Mayra Santana Siqueira

Classificação binária de padrões: Estudo comparativo entre modelos de Programação Matemática

Monografia apresentada ao Curso de Graduação em Ciência da Computação da Universidade Estadual do Norte Fluminense Darcy Ribeiro como requisito para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Fermín Alfredo Tang Montané

UNIVERSIDADE ESTADUAL DO NORTE FLUMINENSE DARCY RIBEIRO

Campos dos Goytacazes/RJ

2014

MAYRA SANTANA SIQUEIRA

Classificação binária de padrões: Estudo comparativo entre modelos de Programação Matemática

Monografia apresentada ao Curso de Graduação em Ciência da Computação da Universidade Estadual do Norte Fluminense Darcy Ribeiro como requisito para obtenção do título de Bacharel em Ciência da Computação.

Aprovada em 27 de janeiro de 2014, Campos dos Goytacazes, RJ.

BANCA EXAMINADORA

Prof. Dr. Fermín Alfredo Tang Montané
Orientador - Universidade Estadual do Norte Fluminense Darcy Ribeiro

---

Prof. Dr. Angel Guillermo Coca Balta
Universidade Estadual do Norte Fluminense Darcy Ribeiro

Prof. Dr. Luis Antonio Rivera Escriba
Universidade Estadual do Norte Fluminense Darcy Ribeiro

“Don’t lose your grip on the dreams of the past

You must fight just to keep them alive”

Eye of the Tiger - Survivor

Dedico este trabalho a minha mãe, Aureni, por priorizar minha educação, me apoiar e incentivar a estudar sempre.

AGRADECIMENTOS

Agradeço a meus amados pais, Aureni e Francisco, primeiramente pelo dom da vida e por seu amor incondicional. A ela por lutar e se dedicar a mim e a minha educação; por seu comprometimento, por sonhar meus sonhos, vibrar com minhas conquistas e dar o suporte necessário para que alcançasse cada uma delas. A ele pelos momentos de dedicação e afeto; por em tão pouco tempo ter conseguido transmitir um amor que ainda hoje permanece vívido em meu coração. A minha irmã Érica por todo incentivo e por ser para mim um exemplo de dedicação, disciplina e esmero. A meu namorado Alexandre Lourenço por sua paciência, companheirismo e amor. A toda minha família e amigos pessoais por todas as palavras de apoio e otimismo.

A todos os professores com quem tive a oportunidade de aprender durante a faculdade. Aos professores Luis Rivera e Annabell Tamariz pela dedicação e amor ao curso. Aos professores, Angel Coca, Fermín Tang e Rodrigo Manhães por sua especial paciência e humildade no ensinar. Registro ainda meu extremo agradecimento ao meu orientador Tang que, apesar de não ter tido a oportunidade de cursar nenhuma de suas matérias, me acolheu como orientada no momento mais crítico em que me encontrei neste curso; por se comprometer com este trabalho e torná-lo possível.

Aos colegas de classe pelos bons momentos vividos em que compartilhamos alegrias e aflições; por todas as dúvidas tiradas, também com eles aprendi. Em especial aos queridos Eduardo Hertz e Thiago Motta pela amizade construída. Às amigas Sânya Carvalho e Camila Morelli, por serem exemplos de perseverança. Nós, "as mulheres" da primeira turma, apesar das dificuldades encontradas no caminho, persistimos e alcançamos, todas, o objetivo final.

Agradeço ainda aos meus cachorros, por me receberem todos os dias com "festinha" e tornarem meus dias mais leves, principalmente aos atuais Nina, Bart e Kenny. Em especial ao Kenny por ser companheiro, estando muitas vezes ao meu lado durante a realização deste trabalho, mesmo que fosse apenas tirando uma soneca rente a minha cadeira.

Resumo

Este trabalho, intitulado *Classificação binária de padrões: Estudo comparativo entre modelos de Programação Matemática*, realiza um estudo comparativo entre modelos de Programação Matemática na classificação de dois conjuntos pontos n-dimensionais linearmente inseparáveis, utilizando cinco conjuntos de dados de aplicações reais. Os conjuntos de dados estudados são divididos em grupos de treinamento e de teste. Os modelos avaliados constroem um hiperplano classificador com auxílio dos dados separados para treino. Através deste hiperplano os dados do grupo de teste são classificados. Os resultados consistem em verificar a taxa de acerto dos classificadores obtidos na fase de treinamento utilizando dos modelos estudados. Foram comparados os resultados de dois modelos de programação linear com os resultados de um modelo de programação não-linear com objetivo de verificar se o melhor ajuste dos classificadores não-lineares nos dados separa melhor os conjuntos que os classificadores lineares.

Sumário

1 Introdução **8**

1.1 Objetivos e Justificativas 9

1.2 Estrutura do trabalho 9

2 Classificação de padrões **11**

2.1 Separabilidade linear entre dois grupos de pontos 11

2.2 Metodologias de classificação de padrões 12

2.2.1 Classificador de Naïve Bayes 12

2.2.2 *Support Vector Machine* 13

2.2.3 Redes Neurais Artificiais 13

2.2.4 Programação Linear na classificação de pontos 14

3 Programação Linear na Otimização **15**

3.1 Pesquisa Operacional 15

3.1.1 Etapas do Estudo da Pesquisa Operacional 16

3.1.2 Modelagem 17

3.1.3 Programação Matemática 18

3.2 Programação Linear 18

3.2.1 Definição do problema 19

3.2.2 Problemas de Programação Linear 21

3.2.3 Propriedades 24

3.2.4 Exemplo numérico do problema de maximização de lucro 25

3.3 Métodos de Solução 26
3.3.1 Representação Gráfica 26
3.3.2 Método Simplex 27

4 Metodologia **32**

4.1 Sobre os dados 33
4.2 Modelos lineares de classificação binária 34
4.2.1 RLP - *Robust Linear Programming* 34
4.2.2 FSLP - *Feature Selection via Linear Programming* 37
4.2.3 PLC - *Piecewise Linear and Convex Separation* 40
4.3 Metodologia de Validação 41
4.3.1 Geração dos Hiperplanos 41
4.3.2 Teste 42

5 Experimento **43**

5.1 Conjuntos de dados testados 43
5.1.1 *Australian Credit Approval* 44
5.1.2 *Boston Housing Data* 44
5.1.3 *Heart Disease Databases* 44
5.1.4 *1984 United States Congressional Voting Records Database* 45
5.1.5 *Wisconsin Breast Cancer Database* 45
5.1.6 Pré-processamento dos dados (eliminando instâncias inconsistentes) . . 45
5.2 Implementação e Validação dos modelos 46
5.2.1 Formatação dos dados para submissão aos modelos 46
5.2.2 Treinamento dos hiperplanos classificadores 47
5.3 Teste 50
5.4 Resultados 50

6 Conclusão **53**

6.1 Considerações Finais 53

6.2 Trabalhos Futuros 54

Referências Bibliográficas **55**

1 Introdução

Classificação é um conceito bastante difundido no meio acadêmico, que consistem em distribuir em grupos, qualificar, determinando as categorias em que se subdivide um grupo (FERREIRA, 2001). Tomando o termo "classe" como grupo, ou divisão que apresenta características semelhantes, numa série ou num conjunto (FERREIRA, 2001). Na ciência, a classificação é utilizada para distinguir um ser dos demais, sejam pessoas, objetos ou conceitos. É aplicada as mais diversas áreas de de estudo, tendo papel fundamental na evolução do conhecimento.

A Programação Linear é uma das mais proeminentes abordagens da programação matemática. Trata-se de uma técnica de otimização com aplicações diversas e amplas no âmbito dos problemas reais. Os problemas de programação linear se referem à distribuição eficiente de recursos limitados entre atividades concorrentes, com a finalidade de atender a um determinado objetivo, mais comumente a maximização de lucro ou a minimização de custos.

Muito reconhecida na área de Engenharia de Produção, a programação linear é amplamente utilizada em vários setores da indústria e logística de empresas. Na computação pode ser aplicada na resolução de problemas de classificação binária de padrões em conjuntos de pontos n-dimensionais linearmente inseparáveis. Em linhas gerais, modelos de classificação via programação linear, capazes de tratar conjuntos de pontos linearmente inseparáveis, buscam minimizar a média de erros de classificação entre os pontos mal posicionados e o hiperplano classificador gerado, balanceando os valores das distâncias entre eles. Uma outra abordagem da programação matemática, a Programação Linear e Inteira Mista busca resolver o problema da classificação de pontos linearmente inseparáveis através da construção de vários hiperplanos obtendo ao final uma superfície classificadora não-linear.

No presente trabalho será realizado um estudo comparativo entre modelos de programação linear *Robust Linear Programming* (RLP) de Bennett e Mangasarian (1991), *Feature Selection via Linear Programming* (FSLP) de Guo e Dyer (2003), segundo metodologia proposta por Ryoo (2006). Os resultados serão comparados a um terceiro modelo de programação multilinear *Piecewise Linear and Convex Separation* (PLC) proposto por Ryoo (2006).

Os modelos serão testados para conjuntos de dados de aplicações do mundo real, cujos dados são subdivididos em dois padrões. As características das instâncias são representadas por pontos no espaço $\mathbb{R}^n$.

1.1 Objetivos e Justificativas

O objetivo deste trabalho é avaliar a eficiência de modelos de classificação binária de padrões linearmente inseparáveis, que utilizam como base um hiperplano separador, quando comparados a uma abordagem que utiliza vários hiperplanos separadores. Com este objetivo, serão testados os modelos de programação linear RLP e FSLP propostos por Bennett e Mangasarian (1991) e Guo e Dyer (2003), respectivamente, e comparados aos resultados do modelo de programação não-linear PLC proposto por Ryoo (2006). Os modelos serão aplicados para cinco conjuntos de dados cujas características são representadas por pontos no $\mathbb{R}^n$. Os conjuntos utilizados no presente trabalho são os mesmos utilizados em Ryoo (2006).

Visando alcançar tais objetivos os conjuntos de dados serão separados em grupos de treino, que juntamente com o modelo irão gerar os hiperplanos separadores; e grupo de teste, que será utilizado para verificar a eficácia do classificador segundo metodologia utilizada em Ryoo (2006). Será verificado se o molelo de programação não-linear PLC proporciona uma melhoria na qualidade da classificação que justifique seu uso, visto sua maior complexidade em relação aos outros dois modelos estudados.

Em situações reais, é difícil encontrar aplicações cujos dados sejam linearmente separáveis. Isso se deve a diversos fatores, como a presença de ruídos nos dados ou à própria natureza do problema que pode ser não-linear. O trabalho se justifica pelo fato de extender o estudo da programação linear no meio da computação, através de um estudo comparativo que coopera na busca de um modelo de programação matemática que tenha melhor eficiência na separação de padrões em que os pontos n-dimensionais que representam as características dos conjuntos são linearmente inseparáveis.

1.2 Estrutura do trabalho

Este trabalho é constituído de seis capítulos. No presente capítulo, consta a introdução ao tema do trabalho, seguida do tópico de objetivos e justificativas.

O Capítulo 2 apresenta uma revisão bibliográfica das principais técnicas e metodologias utilizadas na Programação Linear e na classificação de dados. No Capítulo 3 se encontra o

marco teórico do trabalho, constituído dos principais conceitos sobre Pesquisa Operacional, Programação Linear, separabilidade linear e método Simplex. O Capítulo 4 apresenta os modelos de classificação via programação linear testados e descreve a metodologia de geração dos hiperplanos utilizada. O Capítulo 5 apresenta os conjuntos de dados estudados, descreve o experimento realizado e exibe os resultados estatísticos obtidos. O Capítulo 6 apresenta a conclusão do trabalho, constituída de considerações finais e trabalhos futuros.

2 *Classificação de padrões*

2.1 Separabilidade linear entre dois grupos de pontos

Uma separação binária de padrões consiste na obtenção de critérios para realizar uma distinção entre os elementos de um determinado grupo, dividindo-o em dois subgrupos, tomando o termo "padrão" como uma tendência que se quantifica através de características observáveis de um grupo de indivíduos (sejam objetos ou seres vivos). Para uma separação matemática é necessário que estes padrões sejam representados por pontos no espaço $\mathbb{R}^n$, e uma maneira matemática de obter esta separação é construir um hiperplano, ou alguma outra superfície, de forma que os pontos de padrões diferentes se localizem de lados distintos do hiperplano ou superfície de referência.

Dizer que dois grupos de pontos no $\mathbb{R}^n$ são linearmente separáveis significa não apenas que são disjuntos, mas também que existe um hiperplano linear que os divide os perfeitamente (BENNETT; MANGASARIAN, 1991).

Separabilidade Linear (Definição): Os conjuntos de pontos A e B, representados pela matriz $A \in \mathbb{R}^{m\times n}$ e $B \in \mathbb{R}^{k\times n}$ respectivamente, são linearmente separáveis se e somente se

$$\min_{1\leq i\leq m} A_i\upsilon > \max_{1\leq i\leq k} B_i\upsilon \quad \text{para algum} \quad \upsilon \in \mathbb{R}^n \tag{2.1}$$

que é equivalente a

$$Aw \geq e\gamma + e, \quad e\gamma - e \geq Bw \quad \text{para algum} \quad w \in \mathbb{R}^n,\ \gamma \in \mathbb{R} \tag{2.2}$$

(BENNETT; MANGASARIAN, 1991).

Observa-se ainda que quando os conjuntos A e B são linearmente separáveis como definido na equação acima, o hiperplano $\{x|wx = \gamma\}$ é um hiperplano separador com $Aw > e\gamma$ e $e\gamma > Bw$.

Na abordagem clássica um modelo de programação linear, que busca realizar uma classificação entre dois grupos inseparáveis de pontos, utiliza-se apenas um hiperplano, resultante da minimização

da soma das distâncias dos erros entre os pontos e o hiperplano classificador. Já os modelos de classificação multilineares buscam dividir dois conjuntos de pontos através da geração de vários hiperplanos lineares formando ao final uma superfície não-linear como resultado da união dos hiperplanos por ele gerados. Sendo ilustrados como a seguir:

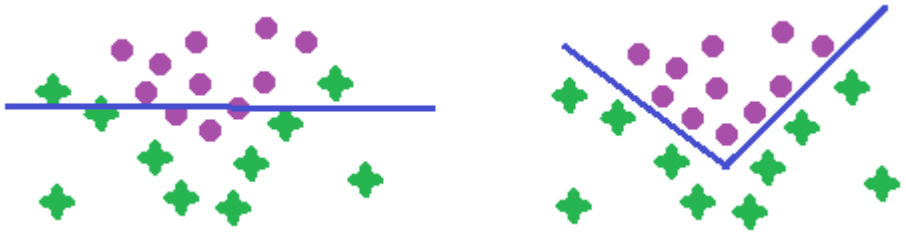


Figura 1: (a) classificador linear (b) classificador multilinear

A Figura 1(a) mostra o provável comportamento do classificador linear na geração de uma reta classificadora única para conjuntos de pontos bilinearmente separáveis no $\mathbb{R}^2$, não conseguindo evitar que alguns pontos sejam posicionados erroneamente em relação à reta. Já a Figura 1(b) mostra o possível comportamento de um modelo de classificação multilinear na classificação dos mesmos pontos, gerando um classificador que separa perfeitamente os dados através da união de duas retas.

2.2 Metodologias de classificação de padrões

Vários ramos da computação têm como meta realizar a classificação de padrões. Neste capítulo serão apresentados brevemente alguns dos principais métodos de classificação de dados.

2.2.1 Classificador de Naïve Bayes

O classificador de Naïve Bayes é também conhecido como classificador bayesiano ingênuo. É uma versão simplificada, uma alternativa ao Classificador Ótimo de Bayes que apresenta alto custo computacional quando o número de hipóteses é alto, ou impossível quando número de hipóteses é infinito, pois considera todas as possíveis hipóteses. Como todo classificador, a função do classificador de Naïve Bayes é definir a qual classe um dado item pertence. Neste caso, isso é feito a partir da probabilidade de um item pertencer a uma das classes (ASTI, 2011). É uma abordagem estatística para reconhecimento de padrões que assume que o problema de decisão é formulado probabilisticamente para todos os valores conhecidos.

Este classificador assume que a probabilidade das características do item são independentes. Em muitos casos, porém, a afirmação de independência não está de acordo com a realidade

e por isso ele é considerado ingênuo. Neste caso não se pode assumir que as probabilidades calculadas estejam corretas. Mesmo assim, este método simples comumente gera um classificador de boa qualidade (ASTI, 2011).

2.2.2 *Support Vector Machine*

As SVMs, máquina de vetores de suporte, do inglês "*support vector machine*", são embasadas pela teoria de aprendizado estatístico, que estabelece uma série de princípios que devem ser seguidos na obtenção de classificadores com boa generalização, definida como capacidade de prever corretamente novos dados do mesmo domínio em que o aprendizado ocorreu (LORENA; CARVALHO, 2007).

SVM é um conceito na ciência da computação relacionado com um conjunto de métodos de aprendizado supervisionado que analisam dados e reconhecem padrões, usado para classificação. A SVM padrão toma como entrada um conjunto de dados e prediz, para cada entrada dada, qual de duas possíveis classes a entrada faz parte. Dados um conjunto de amostras de treinamento, cada uma marcada como pertencente a uma de duas categorias, o algoritmo de treinamento do SVM constrói um modelo que classifica a os pontos de teste como pertencentes a uma categoria ou a outra baseado em qual lado do espaço eles estão localizadas.

Entre as principais limitações das SVMs citadas por Lorena e Carvalho (2007) se encontram a sua sensibilidade a escolhas de valores de parâmetros e a dificuldade de interpretação do modelo gerado, problemas que têm sido abordados em diversos trabalhos da formulação original das SVMs, a qual é capaz de lidar apenas com problemas de classificação binários.

2.2.3 Redes Neurais Artificiais

A Rede Neural é utilizada, basicamente, como um reconhecedor de padrões com capacidade de reconhecer dados que até então não lhe foram apresentados (TAFNER; XERES; FILHO, 1995), ou seja, reconhecer dados que não constituíram o conjunto de treinamento. Considerando essas características, o forte dessas redes é, sem dúvida, o processamento de sinais (TAFNER; XERES; FILHO, 1995).

O processamento de sinais implica, por exemplo, o reconhecimento da fala, da visão, da identificação de comportamentos conforme os diferentes agrupamentos de sinais, dentre outros. Entre as tarefas mais comumente utilizadas pelas redes neurais, estão: predição (meteorologia, oceanografia, economia, bolsa de valores); processamento de sinais complexos (radar, ruído); diagnósticos (máquinas, homens); reconhecimento de voz; reconhecimento de imagens (senso-

riamento remoto); reconhecimento de modelos; identificação geológica; classificação de sinais.

Segundo Jr. e Montgomery (2007) uma desvantagem das redes neurais é o fato delas, normalmente, serem uma "caixa preta". É impossível saber por que uma rede chegou a um resultado ou, no caso de uma rede de múltiplas camadas, saber qual a relevância de um peso sináptico para um determinado resultado ou qual significado físico de um peso sináptico (JR.; MONTGOMERY, 2007). Só é possível saber se a rede funciona corretamente pela análise do erro médio quadrático apresentado ao se introduzir os dados para validação, ou seja, não há como gerar uma prova formal para os resultados obtidos (JR.; MONTGOMERY, 2007).

2.2.4 Programação Linear na classificação de pontos

O problema de Programação Linear é um problema de otimização onde se deseja minimizar (ou maximizar) uma função objetivo linear que possui restrições também lineares, constituidas por um sistema de inequações lineares. Caracteriza-se por utilizar métodos de cálculo baseados na execução repetida de operações relativamente simples, beneficiando-se do advento do computador (PEREIRA; LINS; CALôBA, 2006).

Na classificação de dados a programação linear é utilizada para realizar a busca por um classificador através da maximização das distâncias entre os pontos a serem classificados e o hiperplano gerado, no caso de pontos linearmente separáveis; ou da minimização dos erros de classificação entre os pontos e o hiperplano classificador, no caso de conjuntos de pontos linearmente inseparáveis.

3 *Programação Linear na Otimização*

3.1 Pesquisa Operacional

Durante a Segunda Guerra Mundial, um grupo de cientistas foi convocado na Inglaterra para estudar problemas de estratégia e de tática associados a operações de defesa do país. O objetivo era decidir sobre a utilização mais eficaz de recursos militares limitados. A convocação deste grupo marcou a primeira atividade formal de Pesquisa Operacional.

Os resultados positivos conseguidos pela equipe inglesa motivaram os Estados Unidos a iniciarem atividades semelhantes. A propagação da Pesquisa Operacional deve-se principalmente à equipe de cientistas liderada por George B. Dantzig, dos Estados Unidos, convocada também durante a Segunda Guerra Mundial. Ao resultado deste esforço de pesquisa, concluído em 1947, deu-se o nome de Método Simplex.

Com o fim da guerra, a utilização de técnicas de pesquisa operacional atraiu o interesse de muitas outras áreas. A natureza dos problemas encontrados é bastante abrangente e complexa, exigindo uma abordagem que permita reconhecer os diversos aspectos envolvidos. Uma característica importante da pesquisa operacional, que facilita o processo de análise e de decisão, é a utilização de modelos. Eles permitem a experimentação da solução proposta. Isto significa que uma decisão pode ser mais bem avaliada e testada antes de ser efetivamente implementada. A economia obtida e a experiência adquirida pela experimentação justificam a utilização da Pesquisa Operacional.

Com o aumento da velocidade de processamento e quantidade de memória dos computadores atuais, houve um grande progresso na Pesquisa Operacional. Este progresso é devido também à larga utilização de microcomputadores. Isso faz com que os modelos desenvolvidos pelos profissionais de Pesquisa Operacional sejam mais rápidos e versáteis.

3.1.1 Etapas do Estudo da Pesquisa Operacional

O estudo da Pesquisa Operacional geralmente engloba as seguintes etapas:

Definição do problema

A definição do problema baseia-se em três aspectos principais: a descrição dos objetivos do estudo; a identificação das alternativas de decisão existentes; e o reconhecimento das limitações, restrições e exigências do sistema.

A descrição dos objetivos é uma das atividades mais importantes em todo o processo do estudo, pois a partir dela é que o modelo é concebido. Da mesma forma, é essencial que as alternativas de decisão e as limitações existentes sejam todas explicitadas, para que as soluções obtidas ao final do processo sejam válidas e aceitáveis.

Construção do modelo

A escolha apropriada do modelo é fundamental para a qualidade da solução fornecida. Se o modelo elaborado tem a forma de um modelo conhecido, a solução pode ser obtida através de métodos matemáticos convencionais. Por outro lado, se as relações matemáticas são muito complexas, talvez se faça necessária a utilização de uma combinação de metodologias ou uma nova metodologia.

Solução do modelo

O objetivo desta fase é encontrar uma solução para o modelo proposto. Ao contrário das outras fases, que não possuem regras fixas, a solução do modelo é baseada geralmente em técnicas matemáticas existentes.

No caso de um modelo matemático, a solução é obtida pelo algoritmo mais adequado, em termos de rapidez de processamento e precisão da resposta. A solução obtida, neste caso, é dita "ótima".

Validação do modelo

Nessa altura do processo de solução do problema, é necessário verificar a validade do modelo. Um modelo é válido se, levando-se em conta sua inexatidão em representar o sistema, ele for capaz de fornecer uma previsão aceitável do comportamento do sistema.

Um método comum para testar a validade do sistema é analisar seu desempenho com dados passados do sistema e verificar se ele consegue reproduzir o comportamento que o sistema apresentou.

Implementação da solução

Avaliadas as vantagens e a validação da solução obtida, esta deve ser convertida em regras operacionais. A implementação, por ser uma atividade que altera uma situação existente, é uma das etapas críticas do estudo. É conveniente que seja controlada pela equipe responsável, pois, eventualmente, os valores da nova solução, quando levados à prática, podem demonstrar a necessidade de correções nas relações funcionais do modelo conjunto dos possíveis cursos de ação, exigindo a reformulação do modelo em algumas de suas partes.

Apesar da sequência descrita acima não ser rígida, indica as principais etapas a serem realizadas.

3.1.2 Modelagem

Um modelo é uma representação de um sistema real, que pode já existir ou ser um projeto aguardando execução. No primeiro caso, o modelo pretende reproduzir o funcionamento do sistema, de modo a aumentar sua produtividade. No segundo caso, o modelo é utilizado para definir a estrutura ideal do sistema.

A formulação do modelo depende diretamente do sistema a ser representado. A função objetivo e as funções de restrições podem ser lineares ou não-lineares. As variáveis de decisão podem ser contínuas ou discretas e os parâmetros podem ser determinísticos ou probabilísticos.

A confiabilidade da solução obtida através do modelo depende da validação do modelo na representação do sistema real. A diferença entre a solução real e a solução proposta pelo modelo depende diretamente da precisão do modelo em descrever o comportamento original do sistema. Um problema simples pode ser representado por modelos também simples e de fácil solução. Já problemas mais complexos requerem modelos mais elaborados.

Estrutura dos Modelos Matemáticos

Em um modelo matemático, são incluídos três conjuntos principais de elementos:

1. **Variáveis de decisão e parâmetros:** variáveis de decisão são as incógnitas a serem de-

terminadas pela solução do modelo, parâmetros são valores fixos no problema;

2. **Restrições:** restrições limitam as variáveis de decisão a seus valores possíveis (ou viáveis) de modo a levar em conta as limitações físicas do sistema;

3. **Função objetivo:** é uma função matemática que define a qualidade da solução em função das variáveis de decisão.

3.1.3 Programação Matemática

Um problema de programação matemática é um problema de otimização no qual o objetivo e as restrições são expressos como funções matemáticas e relações funcionais.

Existem diversas técnicas de otimização de modo a resolver cada tipo de modelo existente. Dentre elas se destacam: programação linear, programação inteira, programação dinâmica, programação estocástica e programação não-linear.

Programação linear é utilizada para analisar modelos onde as restrições e a função objetivo são lineares; programação inteira se aplica a modelos que possuem variáveis inteiras (ou discretas); programação dinâmica é utilizada em modelos onde o problema pode ser decomposto em subproblemas; programação estocástica é uma classe especial de modelos onde os parâmetros são descritos por funções de probabilidade; finalmente, programação não-linear é utilizada em modelos contendo funções não-lineares.

Uma característica presente em quase todas as técnicas de programação matemática é que a solução ótima do problema não é encontrada em um único passo, devendo ser obtida iterativamente. É escolhida uma solução inicial (que geralmente não é a solução ótima). Um algoritmo é especificado para determinar, a partir desta solução, uma nova solução, que geralmente é superior à anterior. Este passo é repetido até que a solução ótima seja alcançada (supondo que ela exista). No presente trabalho o algoritmo utilizado para a busca da solução ótima é o algoritmo Simplex.

3.2 Programação Linear

Em um problema de pesquisa operacional a escolha apropriada do modelo é fundamental para a qualidade da solução. A Programação Linear é um método de programação matemática bastante reconhecido pelo seu bom desempenho em aplicações práticas dentro do mundo da otimização.

Dois desenvolvimentos científicos concomitantes, no início do século passado, estimularam o desenvolvimento da Programação Linear. O primeiro iniciou-se em 1928, quando John von Neuman publicou o teorema central da Teoria dos Jogos, que mais tarde foi formulada através da Programação Linear. O segundo desenvolvimento foi a análise insumo-produto (input-output), proposta por Leontief, em um paper de 1936, um modelo matricial linear que posteriormente veio a ser utilizado na forma de um Problema de Programação Linear (PEREIRA; LINS; CALôBA, 2006).

Nas décadas de 30 e 40 alguns problemas foram formulados sob a forma de restrições, e cresceu o interesse em obter um método geral para sua solução. O matemático e econimista soviético Kantorovitch formulou e resolveu um PPL para organização e planejamento da produção em 1939. Em 1941, Hitchcock formulou o PPL do transporte, e em 1945 Strigler formulou o PPL da Dieta. O desenvolvimento de computadores pessoais potentes no início da década de 1980 foi essencial para uma disseminação ainda maior da Programação Linear e da Pesquisa Operacional (PEREIRA; LINS; CALôBA, 2006).

3.2.1 Definição do problema

Como dito anteriormente, a programação linear é utilizada para otimizar (maximizar ou minimizar) uma função linear de variáveis, chamada de função objetivo, sujeita a uma série de equações ou inequações lineares, chamadas restrições. O problema a ser pode ser modelado como o esquema descrito em Lachtermacher (2006).

Otimizar: $f(x_1, x_2, .., x_n)$

Sujeito à:

$$\left.\begin{array}{l} g_1(x_1, x_2, .., x_n) \\ g_2(x_1, x_2, .., x_n) \\ g_3(x_1, x_2, .., x_n) \\ \quad ... \\ g_n(x_1, x_2, .., x_n) \end{array}\right\} \begin{array}{c} \leq \\ \\ = \\ \\ \geq \end{array} \left\{\begin{array}{l} b_1 \\ b_2 \\ b_3 \\ ... \\ b_n \end{array}\right.$$

$$x_1, x_2 \ ..., \ x_n \geq 0$$

Deve ser definido primeiramente o objetivo do problema, ou seja, a otimização a ser alcançada (maximização de lucros ou de desempenhos; minimização de custos, de perdas ou de tempo), que será representado pela função objetivo, a ser maximizada ou minimizada.

$$f(x_1, x_2, ..., x_n) = c_1x_1 + c_2x_2 + ... + c_nx_n$$

Para que a função objetivo seja matematicamente especificada, devem ser definidas as variáveis de decisão envolvidas (número de máquinas, a área a ser explorada, as classes de investimento à disposição). Normalmente, assume-se que todas estas variáveis possam assumir somente valores positivos.

$$x_1, x_2 \ ..., \ x_n \geq 0$$

Estas variáveis estão sujeitas a uma série de restrições, geralmente representadas por inequações (quantidade de equipamento disponível, tamanho da área a ser explorada, exigências nutricionais para determinada dieta) dadas por:

$$g_i(x_1, x_2, ..., x_n) = a_{i1}x_1 + a_{i2}x_2 + ... \ a_{in}x_m$$

Todas essas expressões devem estar de acordo com a hipótese principal da programação linear, ou seja, todas as relações entre as variáveis devem ser lineares. A característica de linearidade pode ser importante para a simplificação da estrutura matemática envolvida. O problema geral de programação linear pode então ser definido como descrito em Hillier e Lieberman (2009):

Maximizar (ou Minimizar) $\quad Z = c_1x_1 + c_2x_2 + ... + c_nx_n$

sujeito à:

$$a_{11}x_1 + a_{12}x_2 + ... \ a_{1n}x_m \leq b_1 (ou \geq, ou =)$$

$$a_{21}x_1 + a_{22}x_2 + ... \ a_{2n}x_m \leq b_2 (ou \geq, ou =)$$

$$...$$

$$a_{m1}x_1 + a_{m2}x_2 + ... \ a_{mn}x_m \leq b_m (ou \geq, ou =)$$

$$x_1, x_2 \ ..., \ x_n \geq 0$$

onde:

- **Z** é a função objetivo a ser maximizada (ou minimizada);
- $\mathbf{c_j}$ é o coeficiente de lucro (ou custo), para i = 1, ..., n;
- $\mathbf{a_{ij}}$ é o coeficiente da variável, para i = 1, ..., n e j = 1, ..., m;
- $\mathbf{b_i}$ é o valor limite da restrição i, para i = 1, ..., n;
- **m** é o número de restrições impostas;
- **n** é o número de variáveis do problema.

3.2.2 Problemas de Programação Linear

A Programação Linear possui vários ramos de aplicação que incluem administração de produção, análise de investimentos, alocação de recursos limitados, logística entre outros. Na sequência é apresentada uma breve descrição dos problemas clássicos da Programação Linear cujas definições e aplicações foram retiradas de Araújo (2013).

Problema de Análise de Atividades (Maximização de lucro)

Esse problema consiste em encontrar $x_1, x_2, ..., x_n$ que maximize a função linear (função objetiva):

$$f(x_1, x_2, ..., x_n) = c_1x_1 + c_2x_2 + ... + c_nx_n$$

sabendo que $x_1, x_2, ..., x_n$ devem satisfazer ao seguinte sistema de inequações lineares (as restrições):

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + ... \; a_{1n}x_m \leq b_1 \\ a_{21}x_1 + a_{22}x_2 + ... \; a_{2n}x_m \leq b_2 \\ ... \\ a_{m1}x_1 + a_{m2}x_2 + ... \; a_{mn}x_m \leq b_m \end{cases}$$

$$x_1, x_2 ...., x_n \geq 0$$

É possível representar este problema de uma forma mais compacta, como mostrado a seguir:

$$\text{Max} \quad f(x_1, x_2, ..., x_n) = \sum_{j=1}^{n} c_j x_j$$

$$\text{sujeito a} \quad \sum_{j=1}^{n} a_{ij} x_j \leq b_i; \quad i = 1, 2, ..., m$$

e

$$x_j \geq 0; \quad j = 1, 2, ..., n$$

Este modelo pode ser associado a uma empresa que tem m recursos disponíveis para a realização de n atividades que representem a fabricação de produtos. Para $i = 1, 2, ..., m$ e $j = 1, 2, ..., n$, tem-se que:

- $\mathbf{a_{ij}}$ é a quantidade de recurso i consumida na produção de uma unidade do produto j;
- $\mathbf{b_i}$ é a quantidade de recurso i disponível para as n atividades $(b_i \geq 0)$;
- $\mathbf{c_j}$ é o lucro unitário do produto j.

- $\mathbf{x_j}$ é o nível de produção da atividade j (são as incógnitas do problema);

Verifica-se então que a função objetivo a ser maximizada representa o lucro total da empresa nas n atividades, as m restrições b_i informam que o total gasto do recurso i nas n atividades tem de ser menor ou no máximo igual a quantidade b_i disponível daquele recurso e as restrições $x_j \geq 0$ eliminaram a possibilidade de valores negativos para as atividades.

Problema da Dieta (Minimização de custo)

Este problema consiste em achar $x_1, x_2, ..., x_n$ que minimize a função objetiva:

$$f(x_1, x_2, ..., x_n) = c_1x_1 + c_2x_2 + ... + c_nx_n$$

sabendo que $x_1, x_2, ..., x_n$ devem satisfazer às restrições:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + ... \; a_{1n}x_m \geq b_1 \\ a_{21}x_1 + a_{22}x_2 + ... \; a_{2n}x_m \geq b_2 \\ ... \\ a_{m1}x_1 + a_{m2}x_2 + ... \; a_{mn}x_m \geq b_m \end{cases}$$

$$x_1, x_2 ...., x_n \geq 0$$

É possível representar este problema como:

$$\text{Min} \quad f(x_1, x_2, ..., x_n) = \sum_{j=1}^{n} c_j x_j$$

$$\text{sujeito a} \quad \sum_{j=1}^{n} a_{ij} x_j \geq b_i; \quad i = 1, 2, ..., m$$

e

$$x_j \geq 0; \quad j = 1, 2, ..., n$$

Este modelo pode ser associado a uma pessoa que deseja minimizar o custo da sua dieta diária. As atividades apresentam o consumo dos alimentos que poderão entrar na dieta e os recursos são as vitaminas que não podem deixar de ser consumidas na dieta. Para $i = 1, 2, ..., m$ e $j = 1, 2, ..., n$, tem-se que:

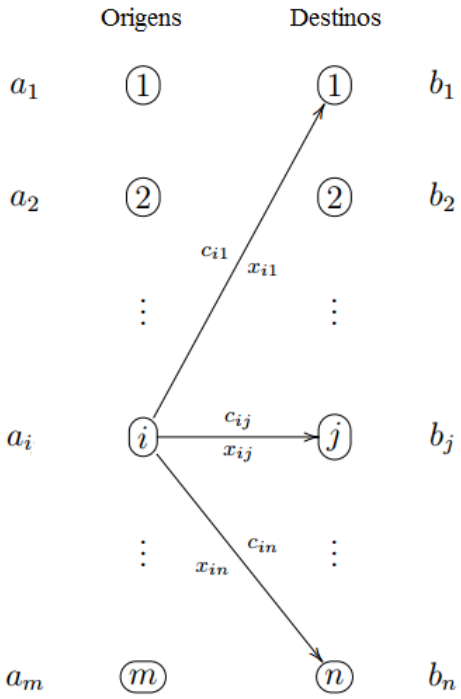
- $\mathbf{a_{ij}}$ é a quantidade de vitamina i fornecida por uma unidade do alimento j;
- $\mathbf{b_i}$ é a quantidade mínima de vitamina i que deve ser obtida nos n alimentos $(b_i \geq 0)$;

- $\mathbf{c_j}$ é o custo unitário do alimento j;
- $\mathbf{x_j}$ é quantidade de alimento j na dieta (são as incógnitas do problema).

Verifica-se então que a função objetivo a ser minimizada representa o custo total da dieta com os n alimentos, as m restrições b_i indicam que o total de vitamina i obtida nos n alimentos tem de ser maior ou igual a quantidade mínima b_i daquela vitamina.

Problema do Transporte (Minimização de custo)

O Problema do Transporte tem como objetivo minimizar o custo total do transporte necessário para abastecer n centros consumidores (demanda), chamados destinos, a partir de m centros fornecedores (oferta), chamados origens. Este problema pode ser esquematizado como mostrado seguir:



Para $i = 1,2,...,m$ e $j = 1,2,...,n$, tem-se que:

- $\mathbf{a_i}$ é a quantidade disponível na origem i;
- $\mathbf{b_j}$ é a quantidade requerida no destino j;
- $\mathbf{c_{ij}}$ é o custo unitário de transporte da origem i para o destino j;
- $\mathbf{x_{ij}}$ é quantidade a ser transportada da origem i para o destino j (incógnitas do problema);

O problema consiste em encontrar os valores de x_{ij} que minimize o custo total do transporte

$$f = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}$$

fazendo x_{ij} satisfazerem às seguintes restrições de oferta e demanda:

$$\sum_{i=1}^{m} x_{ij} = a_i, \quad i = 1,2,...,m$$

$$\sum_{j=1}^{n} x_{ij} = b_j; \quad i = 1,2,...,n$$

e

$$x \geq 0.$$

3.2.3 Propriedades

As limitações no uso de problemas de programação linear para representar sistemas reais devem-se aos pressupostos que somos obrigados a assumir de forma a operacionaliza-los (PEREIRA; LINS; CALôBA, 2006). Segundo Bazaraa, Sherali e Jarvis (1990), estes pressupostos são:

1. **Proporcionalidade:** Os níveis de aditividade das alternativas são multiplicados pelos coeficientes na função objetivo e nas restrições: $c_j x_j$ e $a_{ij} x_j$ - isto implica em que não se considera economias de escala e nem custos iniciais para implantação da alternativa j, no caso da função objetivo.

2. **Aditividade:** Não existe interação entre as alternativas de atividades, tanto na função objetivo quanto nas restrições.

3. **Divisibilidade:** As variáveis de decisão podem ser fracionadas. A importância desse pressuposto depende do porte do problema. Se as quantidades se tratarem da ordem de milhares, a aproximação para valores inteiros não implicará em erros importantes, mas se lidamos com valores da ordem de poucas unidades, é possível incorrer em erros significativos. Neste caso, é preciso utilizar a Programação Inteira, outro ramo da Pesquisa Operacional, a qual faz também uso da Programação Linear.

4. **Determinismo:** c_j, a_{ij} e b_j não incorporam a natureza probabilística de custos, demandas, preços e disponibilidade de recursos. Uma alternativa para superar esta limitação é efetuar uma análise de pós-otimização ou sensibilidade.

A observância desses pressupostos implica admitir simplificações que não invalidam o resultado a ser obtido através do modelo de Programação Linear (PEREIRA; LINS; CALôBA, 2006).

3.2.4 Exemplo numérico do problema de maximização de lucro

A seguir um exemplo retirado de Lisboa (2002)

"Uma empresa de comida canina produz dois tipos de rações: Tobi e Rex. Para a manufatura das rações são utilizados cereais e carne. Sabe-se que:

- a ração Tobi utiliza 5 kg de cereais e 1 kg de carne, e a ração Rex utiliza 4 kg de carne e 2 kg de cereais;
- o pacote de ração Tobi custa $20 e o pacote de ração Rex custa $30;
- o kg de carne custa $4 e o kg de cereais custa $ 1;
- estão disponíveis por mês 10 000 kg de carne e 30 000 kg de cereais.

Deseja-se saber qual a quantidade de cada ração a produzir de modo a maximizar o lucro."

A Tabela a seguir apresenta o cálculo do lucro unitário de cada ração.

	Ração Tobi	Ração Rex
Custo de carne	1kg x $4 = $4	4kg x $4 = $16
Custo de cereais	5kg x $1 = $5	2kg x $1 = $2
Custo total	$9	$18
Preço	$20	$30
Lucro	$11	$12

Tabela 1: Cálculo do lucro unitário de cada ração

O modelo deseja maximizar o lucro (Z) a partir da quantidade de ração Tobi x_1 e de ração Rex x_2. A função objetivo pode ser escrita como:

$$\begin{aligned} \text{maximizar} \quad & Z = 11x_1 + 12x_2 \\ \text{sujeito à} \quad & 1x_1 + 4x_2 \leq 10.000 \\ & 5x_1 + 2x_2 \leq 30.000 \\ & x_1,\ x_2 \geq 0 \end{aligned}$$

3.3 Métodos de Solução

3.3.1 Representação Gráfica

É possível resolver problemas de Programação Linear através da solução gráfica. Quando se tratam de problemas com até duas variáveis de controle, $Z = f(x_1, x_2)$, traça-se um gráfico com os eixos sendo as duas variáveis x_1 e x_2. Depois, são traçadas as retas referentes às restrições do problema e delimita-se a região viável. Encontrada a região viável, deve-se traçar uma reta com a inclinação da função objetivo. Após isso são traçadas diversas retas paralelas no sentido de Z crescente (no caso de maximização da função). O ponto ótimo é o ponto onde a reta de maior valor possível cortar a região viável (normalmente num vértice).

Para o problema da empresa de comida canina descrito na Seção 3.2.4 a região viável é delimitada como no gráfico abaixo.

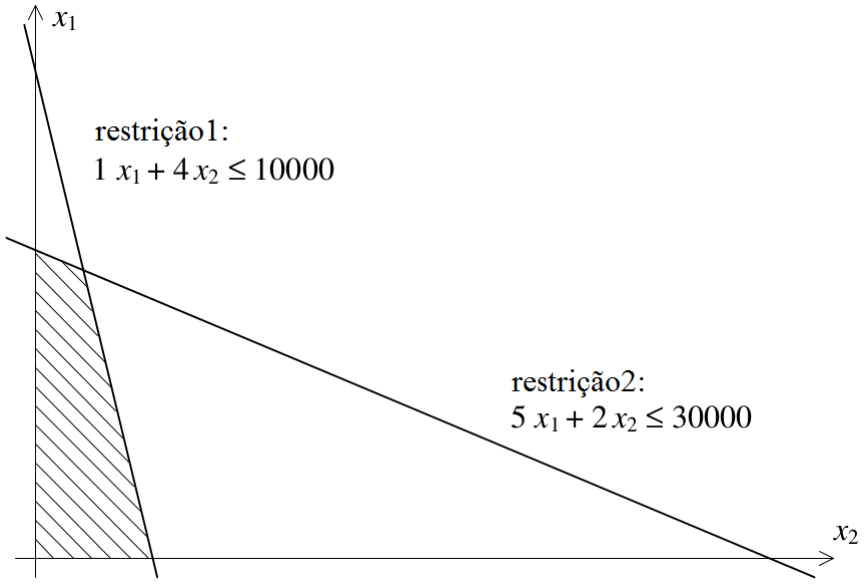


Figura 2: Região viável para o problema das rações

As retas traçadas pela função objetivo fazendo Z crescente e sua interseção com o vértice contendo a solução ótima em $x_1 = 5555{,}6$ e $x_2 = 1111{,}1$ são mostradas no gráfico a seguir:

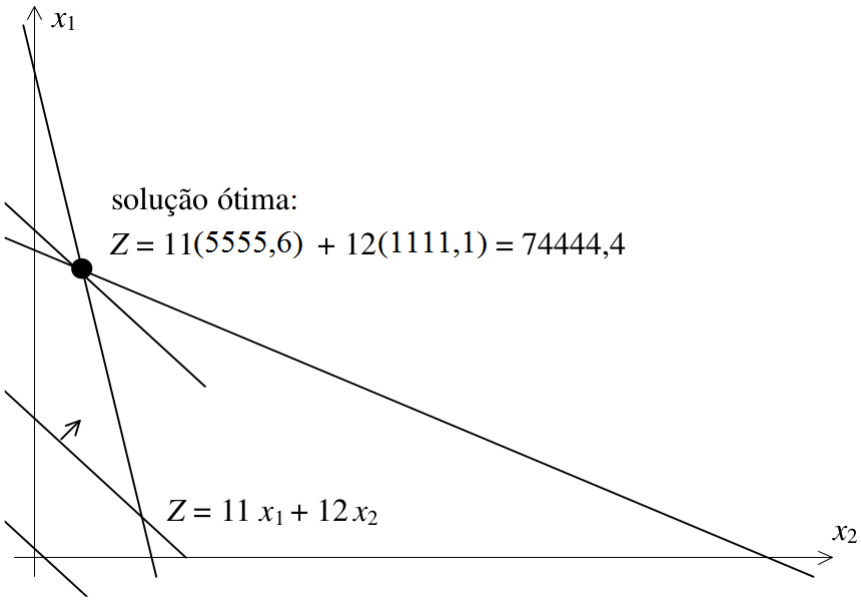


Figura 3: Busca da solução ótima para o problema das rações

A maior parte dos problemas práticos envolvem mais variáveis, o que faz com que seja necessária a utilização de outros métodos para solucionar o problema.

3.3.2 Método Simplex

O método Simplex é uma técnica que procura encontrar algebricamente, a solução ótima de um modelo de programação linear. É um procedimento iterativo que fornece a solução exata de qualquer problema de programação linear em um número finito de iterações. É também capaz de indicar se o problema tem solução ilimitada, se não tem solução ou se possui infinitas soluções.

Estas características permitiram sua codificação em programas rápidos e eficientes, possibilitando a solução de sistemas com centenas de variáveis. Extensões posteriores, como o simplex revisado e o princípio da decomposição, aumentaram sua capacidade para dezenas de milhares e, finalmente, centenas de milhares de variáveis. Além disso, possui uma interpretação geométrica bastante simples como é possível observar na ilustração da Figura 4 para o problema da empresa canina resolvido na Seção 3.3.1.

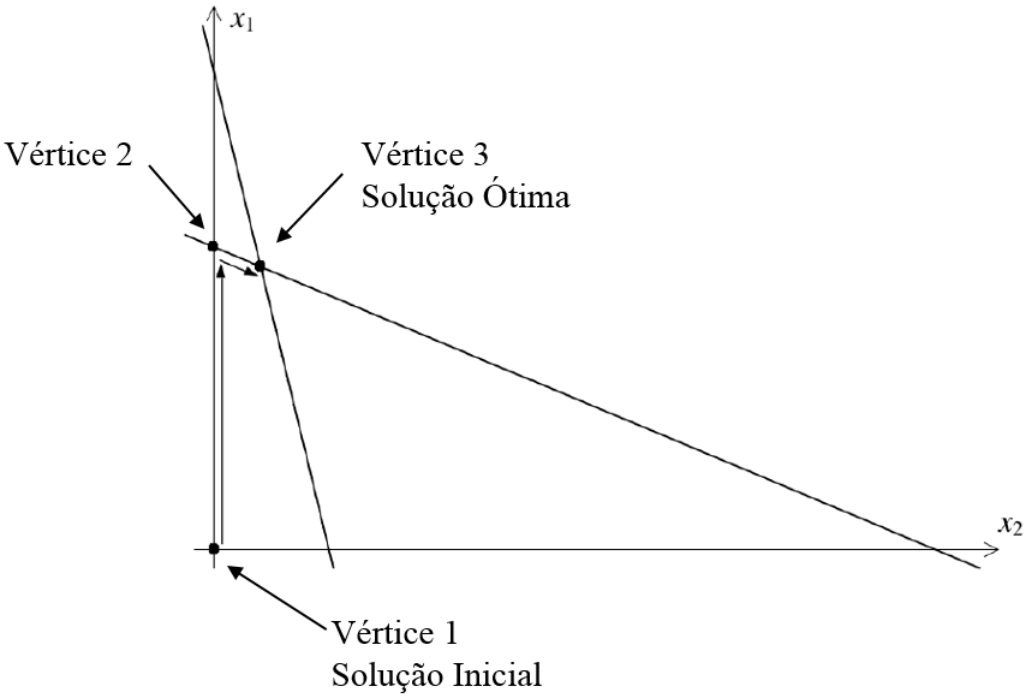


Figura 4: Percurso do método Simplex na busca da solução ótima para o problema das rações

A idéia básica do método Simplex consiste em resolver repetidas vezes um sistema de equações lineares para obter uma sucessão de soluções básicas adjacentes, uma sempre melhor do que a anterior, até chegar a uma solução ótima. Cada nova solução é obtida a partir da anterior, substituindo uma variável básica por uma variável não básica.

O modelo de Programação Linear pode ser resolvido por um método de solução de equações lineares. Este problema requer a resolução de sistemas e a comparação de soluções de muitas equações lineares. Para problemas reais de Programação Linear, esta solução se torna inviável. Desta forma, é necessário uma sistemática que forneça:

- qual o sistema de equações será resolvido;
- que o próximo sistema a ser resolvido fornecerá uma solução melhor que os anteriores;
- como identificar uma solução ótima, quando ela for encontrada.

As regras descritas acima compõem a sistemática do método Simplex, que é um algoritmo criado para obter uma solução de forma algébrica. O método caminha pelos vértices da região viável até encontrar uma solução que não possua soluções vizinhas melhores que ela. A solução ótima pode não existir quando: não há nenhuma solução viável para o problema (devido a restrições incompatíveis); ou quando não há máximo ou mínimo (quando uma ou mais variáveis tenderem ao infinito), o que fornece valor sem limites para a função objetivo.

Algoritmo Simplex

A seguir serão listados os passos do algoritmo Simplex descritos por Lisboa (2002).

1. Introduzir as variáveis de folga, uma para cada desigualdade.

2. Montar um quadro para os cálculos, colocando os coeficientes de todas as variáveis com os respectivos sinais e, na última linha, incluir os coeficientes da função objetivo transformada.

3. Estabelecer uma solução básica inicial, usualmente atribuindo valor zero às variáveis originais e encontrando valores positivos para as variáveis de folga.

4. Como próxima variável a entrar na base, escolher a variável não básica que oferece, na última linha, a maior contribuição para o aumento da função objetivo (ou seja, tem o maior valor negativo). Se todas as variáveis que estão fora da base tiverem coeficientes nulos ou positivos nesta linha, a solução atual é ótima. Se alguma dessas variáveis tiver coeficiente nulo, isto significa que ela pode ser introduzida na base sem aumentar o valor da função objetivo. Isso quer dizer que foi encontrada uma solução ótima, com o mesmo valor da função objetivo.

5. Para escolher a variável que deve deixar a base, deve-se realizar o seguinte procedimento:

   a) Dividir os elementos da última coluna pelos correspondentes elementos positivos da coluna da variável que vai entrar na base. Caso não haja elemento algum positivo nesta coluna, o processo deve parar, já que a solução seria ilimitada.

   b) O menor quociente indica a equação cuja respectiva variável básica deverá ser anulada, tornando-se variável não básica.

6. Usando operações válidas com as linhas da matriz, transformar o quadro de cálculos de forma a encontrar a nova solução básica. A coluna da nova variável básica deverá se tornar um vetor identidade, onde o elemento 1 aparece na linha correspondente à variável que está sendo anulada.

7. Retornar ao passo 4 para iniciar outra iteração.

Exemplo

O problema das rações proposto na Seção 3.2.4 será resolvido através do algoritmo Simplex especificado acima. O modelo é relembrado a seguir:

maximizar $Z = 11x_1 + 12x_2$

sujeito à $1x_1 + 4x_2 \leq 10.000$

$5x_1 + 2x_2 \leq 30.000$

$x_1,\ x_2 \geq 0$

Inclusão das variáveis de folga:

Para a resolução do modelo, o sistema é escrito na forma canônica, na qual se observa a inclusão das variáveis x_3 e x_4, para que as inequações se tornem equações sem perder a validade matemática. Essas variáveis adicionais são as variáveis de folga. A cada equação de restrição é adicionada uma variável de folga. E cada variável de folga adicionada exige uma equação de não-negatividade correspondente.

$$\begin{aligned} Z - 11x_1 - 12x_2 \qquad\quad &= 0 \\ 1x_1 + 4x_2 + x_3 \qquad &= 10.000 \\ 5x_1 + 2x_2 \qquad + x_4 &= 30.000 \\ x_1,\ x_2,\ x_3,\ x_4 &\geq 0 \end{aligned}$$

Solução inicial:

Base	x_1	x_2	x_3	x_4	b
x_3	1	4	1	0	10.000
x_4	5	2	0	1	30.000
Z	-11	-12	0	0	0

Primeira iteração:

Variável a entrar na base: x_2 (coluna de maior valor negativo na última linha)

Variável a sair da base: x_3 (com 10.000/4 sendo o menor quociente entre a última coluna e a coluna da variável x_2 que entrará na base)

$L_1 \leftarrow L_1/4$

$L_2 \leftarrow L_2 - 2L_1$

$L_3 \leftarrow L_3 + 12L_1$

Segunda iteração:

Variável a entrar na base: x_1 (coluna de maior valor negativo na última linha)

Variável a sair da base: x_4 (com 25.000/4,5 sendo o menor quociente entre a última coluna e a

Base	x_1	x_2	x_3	x_4	b
x_3	1/4	1	1/4	0	2.500
x_4	4,5	0	-1/2	1	25.000
Z	-8	0	3	0	30.000

coluna da variável x_1 que entrará na base)

$L_2 \leftarrow L_1/4,5$

$L_1 \leftarrow L_1 - L_2$

$L_3 \leftarrow L_3 + 8L_2$

Base	x_1	x_2	x_3	x_4	b
x_2	0	1	0,2778	-0,0556	1111,11
x_1	1	0	-0,1111	0,2222	5555,56
Z	0	0	2,1111	1,7778	74444,44

Solução ótima encontrada:

Como todos os valores da última linha são positivos ou nulos, conclui-se que a solução encontrada é ótima, ou seja:

$x_1 = 5555,56$

$x_2 = 1111,11$

$Z = 74444,44$

4 *Metodologia*

Neste trabalho foram avaliados dois modelos de programação linear para classificação de conjuntos inseparáveis, o *Robust Linear Programming* (RLP) de Bennett e Mangasarian (1991) e o *Feature Selection via Linear Programming* (FSLP) de Guo e Dyer (2003), com objetivo de comparar sua eficiência à do modelo de classificação multilinear *Piecewise Linear and Convex Separation* (PLC) de Ryoo (2006). Os modelos avaliados são testados para conjuntos de dados reais de diferentes tipos de aplicação no intuito de comprovar sua eficácia na classificação binária de conjuntos inseparáveis através da determinação de um hiperplano classificador único.

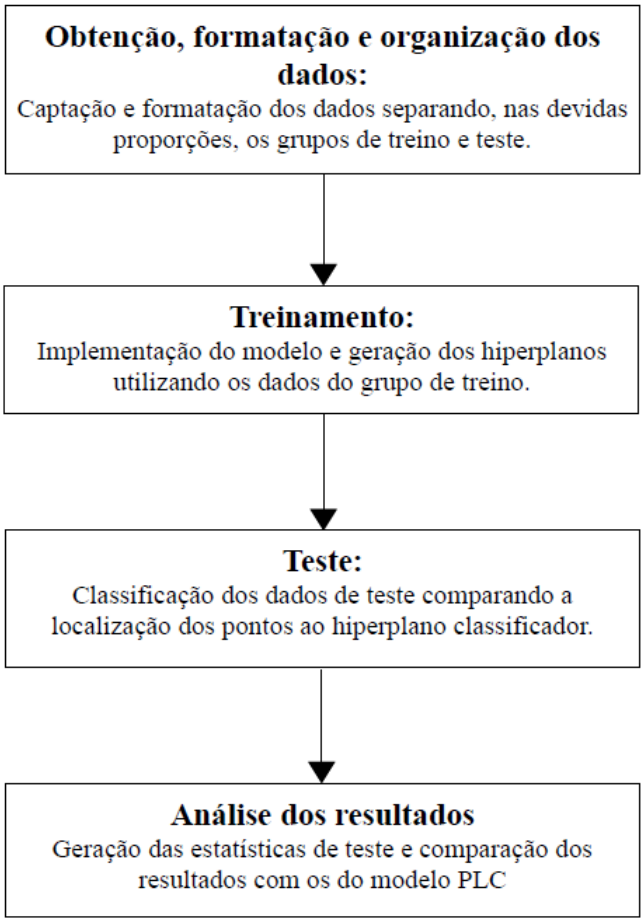


Figura 5: Etapas da avaliação de cada modelo estudado

Visando alcançar os referidos objetivos, foram realizadas as etapas de obtenção, formatação e organização dos dados, treinamento dos classificadores, teste dos dados e cálculo das estatísticas de resultado. A Figura 5 ilustra as etapas citadas, que foram seguidas para cada um dos cinco conjuntos de dados avaliados.

A obtenção dos dados consistiu em utilizar grupos de dados cujas classes fossem previamente definidas. Esses dados foram formatados e organizados em subgrupos de treino e teste de modo a se adequarem à metodologia apresentada em Ryoo (2006). Depois de separar os subgrupos de treino e teste, ainda nesta etapa é determinado a qual grupo pertence cada ponto, através do atributo de classe. A etapa de treinamento trata da implementação e execução dos modelos de programação linear nos pontos do grupo separado para treino, objetivando gerar os hiperplanos classificadores. Na etapa de teste são avaliados os pontos do grupo de teste com relação ao hiperplano classificador gerado na etapa de treino. Finalmente a obtenção dos resultados é feita através do cálculo das médias de acerto e respectivos desvios-padrão de cada conjunto de dados utilizado.

4.1 Sobre os dados

O presente trabalho utiliza modelos que demandam que o conjunto de pontos analisados se apresentem na forma de vetores de características n-dimensionais sendo agrupados em dois padrões definidos pelo atributo de classe. Os vetores de uma classe são atribuídos ao grupo A e os outros ao grupo B.

A organização dos dados se deu através da metodologia de validação de dados apresentada em Ryoo (2006), que avalia os conjuntos de dados com duas abordagens para cada um deles. A divisão é realizada de maneira aleatória mediante permutação dos dados. Foram realizadas 30 repetições para cada abordagem.

Uma abordagem divide os dados em 50% de treino e 50% de teste, já a outra abordagem separa 80% para treino e consequentemente 20% para teste. Depois, tando no subgrupo de treino quanto no de teste, os pontos são mais uma vez separados em Classe A e Classe B através do atributo de classe previamente conhecido. O processo é realizado para cada uma das abordagens (50% e 80% de dados para treino) 30 vezes, sendo as instâncias (vetores) permutadas antes de cada divisão de porcentagens (treino e teste).

4.2 Modelos lineares de classificação binária

Um hiperplano classificador é avaliado como gerador de uma separação ótima se classifica um conjunto de pontos de forma a distingui-los ou, no caso de conjuntos inseparáveis, se minimiza a soma dos erros de classificação. Quando um modelo de programação linear busca realizar a classificação binária de grupos de pontos linearmente inseparáveis, seu objetivo é encontrar uma solução que irá errar menos, dado que a separação perfeita nunca será atingida.

Para este trabalho foram implementados dois modelos de classificação via programação linear, o RLP e o FSLP. As duas seções a seguir descrevem como cada um se apresenta no objetivo de gerar um hiperplano classificador ótimo para a separação binária dos conjuntos estudados. Na sequência encontra-se também a explicação do modelo multilinear PLC.

4.2.1 RLP - *Robust Linear Programming*

O modelo *Robust Linear Programming* - RLP consiste numa técnica de programação linear que busca classificar binariamente conjuntos de pontos n-dimensionais. Para atingir esse objetivo, o modelo busca encontrar um hiperplano linear que faça distinção entre esses dois grupos de pontos, mesmo que estes pontos sejam linearmente inseparáveis.

Para dois conjuntos de pontos A e B em $\mathbb{R}^n$, procura-se uma função linear tal que $f(x) > 0$ se $x \in A$, e $f(x) \leq 0$ se $x \in B$, que é dada por $f(x) = w'x–\gamma$, e determina um plano $P : w'x = \gamma$, com normal $w \in \mathbb{R}^n$, que busca separar os pontos A dos de B.

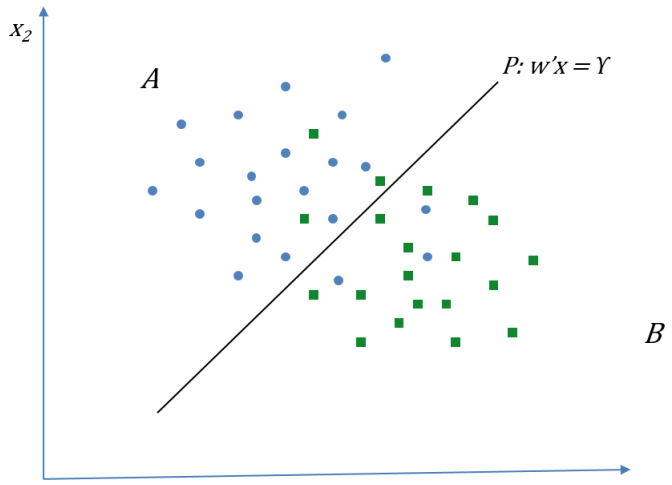


Figura 6: Hiperplano P: $w'x = \gamma$ sobre os conjuntos de pontos inseparáveis A representado por círculos) e B (representado por quadrados)

O conjunto de m pontos, A, é representado por uma matriz $A \in \mathbb{R}^{m \times n}$ e o conjunto de k pontos, B, é representado pela matriz $B \in \mathbb{R}^{k \times n}$, onde se deseja satisfazer

$$Aw \geq e\gamma + e, \quad Bw \leq e\gamma - e \tag{4.1}$$

em que e é um vetor de 1s com dimensão associada à dimensão do grupo a que pertence. Sendo assim o vetor e referente ao grupo A deve ser de dimensão m e o e referente ao grupo B possuir dimensão k.

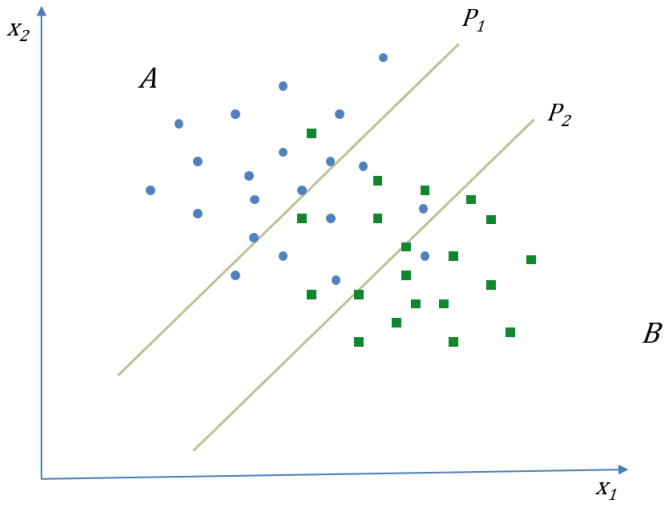


Figura 7: Hiperplanos auxiliares $P_1 : w'x = \gamma + 1$ e $P_2 : w'x = \gamma - 1$ sobre os conjuntos de pontos inseparáveis A e B

Na prática, por causa da sobreposição entre as duas classes, deve-se minimizar parte da média dos erros da norma-1 em (4.1) (BENNETT; MANGASARIAN, 1991), como apresentado a seguir:

$$\min_{w,\gamma} f(w,\gamma) = \min_{w,\gamma} \frac{1}{m}\|(-Aw + e\gamma + e)_+\|_1 + \frac{1}{k}\|(Bw - e\gamma + e)_+\|_1 \tag{4.2}$$

onde x_+ indica o vetor com componentes $\max\{0,\ x_i\}$. Há duas razões principais para a escolha da norma-1 na Eq. (4.2): i) É fácil formular como um programa linear, com propriedades teóricas que o tornam computacionalmente eficiente (BENNETT; MANGASARIAN, 1991), e ii) A norma-1 é menos sensível a valores extremos e discrepâncias.

A equação (4.2) pode então ser modelada como o *Robust Linear Programming* de Bennett e Mangasarian (1991), que minimiza a soma média dos erros de classificação dos pontos com relação aos dois hiperplanos delimitadores, $x'w = \gamma + 1$ e $x'w = \gamma - 1$. O modelo é descrito a seguir:

$$\begin{aligned} \min_{w,\gamma,y,z} \quad & \frac{e'y}{m} + \frac{e'z}{k} \\ \textit{sujeito à} \quad & -Aw + e\gamma + e \leq y, \\ & Bw - e\gamma + e \leq z, \\ & y \geq 0,\ z \geq 0. \end{aligned} \tag{4.3}$$

As variáveis do modelo são descritas abaixo:

- w - vetor que contém os coeficientes angulares do hiperplano classificador;
- γ - coeficiente linear do hiperplano classificador;
- y - vetor que contém os valores dos erros de cada ponto do Grupo A;
- z - vetor que contém os valores dos erros de cada ponto do Grupo B;

A ilustração a seguir representa os pontos da ilustração 8 corretamete localizados em relação aos hiperplanos auxiliares P1 e P2.

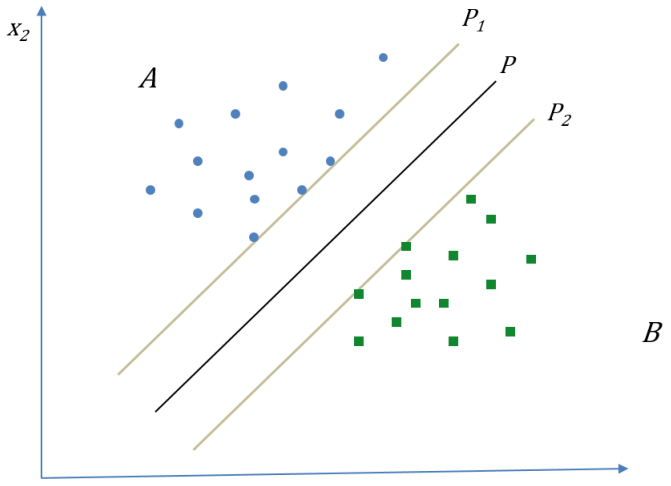


Figura 8: Pontos que não violam as fronteiras P_1 e P_2

A ilustração a seguir representa os pontos da ilustração 9 que violam as fronteiras dos hiperplanos auxiliares e contribuem na piora da função objetivo.

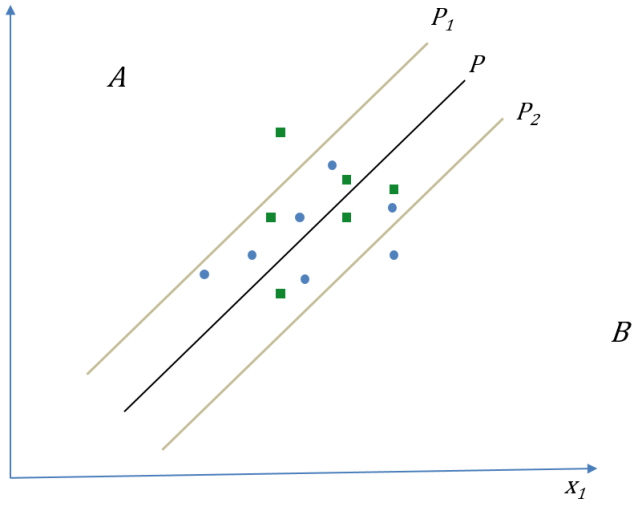


Figura 9: Pontos que violam as fronteiras P_1 e P_2

Devido ao fato da função objetivo ser calculada a partir dos hiperplanos auxiliares (P_1 e P_2), os pontos ilustrados acima são considerados como violações, mesmo que alguns estejam localozados do lado correto em relação ao hiperplano classificador P.

4.2.2 FSLP - *Feature Selection via Linear Programming*

O objetivo da seleção de características em problemas de reconhecimento de padrões é obter um pequeno conjunto de propriedades, mantendo apenas as características mais significativas para a classificação. Mas os benefícios da seleção características não são apenas reduzir o tempo de reconhecimento através da redução da quantidade de dados que tem de serem analisados, mas também, em muitos casos, produzir uma melhor precisão da classificação (GUO; DYER, 2003).

O modelo RLP (4.3) resolve o problema da classificação, sem considerar o problema da seleção de características. Em Bradley e Mangasarian (apud GUO; DYER, 2003) uma estratégia de seleção de características foi integrada na função objetivo, afim de, simultaneamente, selecionar um subconjunto das características. A seleção de características é definida através da supressão de tantos componentes do vetor w normal ao plano P de separação quanto forem necessários para obter uma discriminação aceitável entre os conjuntos A e B, ou seja, reduzir a dimensão dos vetores de características do conjunto de dados que auxiliará a geração do hiperplano classificador. Para realizar isto, um termo extra é adicionado à função objetivo de (4.3), a reformulando como

$$\begin{aligned} \min_{w,\gamma,y,z} \quad & (1-\lambda)\left(\frac{e'y}{m}+\frac{e'z}{k}\right)+\lambda e'|w|_* \\ \textit{sujeito à} \quad & -Aw+e\gamma+e \leq y, \\ & Bw-e\gamma+e \leq z, \\ & y \geq 0,\ z \geq 0. \end{aligned} \tag{4.4}$$

onde $|w|_* \in \mathbb{R}^n$ tem componentes iguais a 1 se os componentes correspondentes de w são diferentes de zero, e tem componentes iguais a 0 se os componentes correspondentes de w são 0. Então $e'|w|_*$, é na verdade um contador dos elementos diferentes de zero no vetor w. Esta é a chave para integrar a seleção de características com o processo de treinamento do classificador. Como resultado, o problema (4.4) equilibra o erro na discriminação entre os dois conjuntos A e B, $\frac{e'\gamma}{m}+\frac{e'z}{k}$, e o número de elementos diferentes de zero de w, $e'|w|_*$. Além disso, se um elemento de w é 0, a característica correspondente é removida. Assim, apenas os recursos correspondentes aos componentes diferentes de zero na normal w são selecionados após a otimização realizada pelo modelo.

Em Bradley e Mangasarian (apud GUO; DYER, 2003) um método chamado *Feature Selection via Concave Minimization* (FSV), que se traduz "Seleção de Características via Minimização Côncava" foi desenvolvido para tratar o último termo na função objetivo de (4.4). No FSLP, alternativamente, ao invés de calcular a aproximação exponencial côncava, utiliza-se um termo simples $e's$ com apenas um parâmetro μ. Isso produz a formulação a seguir:

$$\begin{aligned} \min_{w,\gamma,y,z} \quad & \left(\frac{e'y}{m}+\frac{e'z}{k}\right)+\mu e's \\ \textit{sujeito à} \quad & -Aw+e\gamma-y \leq -e, \\ & Bw-e\gamma-z \leq -e, \\ & -s \leq w \leq s, \\ & y,z \geq 0. \end{aligned} \tag{4.5}$$

As características que correspondem aos componentes 0 no vetor normal podem ser descartados, e somente aqueles componentes diferentes de zero são usados. Como resultado, nenhum parâmetro especificado pelo usuário é requerido para informar ao sistema quantas características usar (GUO; DYER, 2003). Quando $\mu = 0$ o último termo na função objetivo de (4.5) desaparece.

A sequência de figuras a seguir ilustra o comportamento dos hiperplanos auxiliares para μ em diferentes circunstâncias. Quando $\mu = 0$ (Figura 10) a margem permanece com distância 1, assim como ocorre em RLP; fazento $0 < \mu < 1$ (Figura 11) a margem é estreitada; e quando $\mu \geq 1$ (Figura 12) a margem é aumentada.

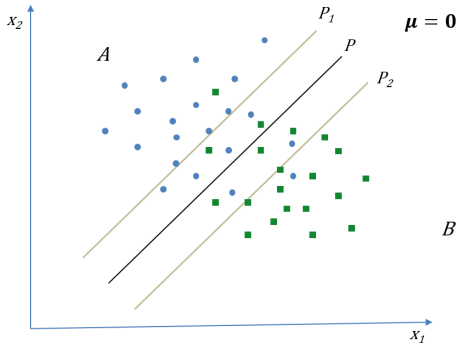


Figura 10: Exemplo iliustrativo de FSLP para $\mu = 0$

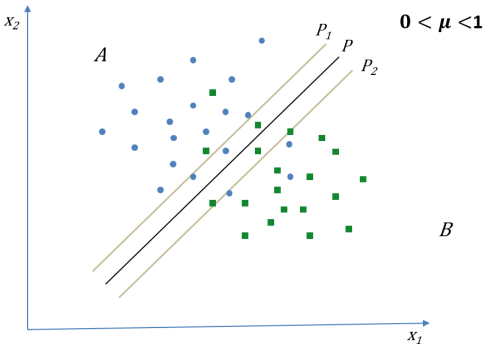


Figura 11: Exemplo iliustrativo de FSLP para $0 < \mu < 1$

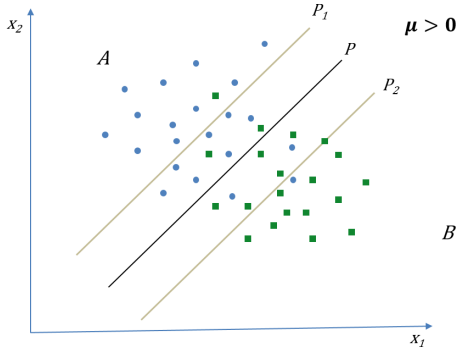


Figura 12: Exemplo iliustrativo de FSLP para $\mu \geq 1$

Dentre as propriedades que o FSLP possui, como as citadas em Guo e Dyer (2003) as que se destacam considerando o propósito do presente trabalho, são: (1) FSLP pode determinar quantas características usar automaticamente sem nenhuma interação com o usuário; (2) FSLP

promove uma alta performance em reconhecimento; e (3) seleção de características FSLP é rápida de computar.

4.2.3 PLC - *Piecewise Linear and Convex Separation*

Esta é uma abordagem de aprendizado baseada em um modelo de MILP (programação linear e inteira mista) para a separação convexa e linear por partes para dois conjuntos de pontos no $\mathbb{R}^n$. Este modelo constrói concomitantemente k hiperplanos que coletivamente constituem uma superfície de decisão não-linear.

Aqui, considera-se um classificador binário como uma superfície linearmente secionada, que pode ser formada por um número finito, diga-se k, de hiperplanos $\omega_i x = \gamma_i, i = 1,...,k$, de modo que uma combinação da metade dos espaços com o melhor associado a eles separa um tipo de dado do outro. Onde, 'melhor' se refere à separação robusta dos dois conjuntos no sentido introduzido em Bennett e Mangasarian (1991), já mencionado na seção 4.2.1.

No caso das k separações secionadas linearmente, a forma do modelo de aprendizado é exclusivamente determinada pelo valor de k. Os parâmetros desconhecidos da teoria de decisão são as normais e localização dos k hiperplanos que colocam conjuntamente um tipo de dado acima e o outro tipo de dado abaixo, de forma que os valores totais dos erros de classificação, sejam minimizados. Depois da solução, se a taxa de erro nos dados de treinamento não for aceitável ou se a taxa de predição num conjunto de dado de teste é ruim, uma nova forma é postulada no modelo, e todo o processo é repetido.

O modelo realiza a separação convexa e por partes lineares k de dois conjuntos e determina superfícies de decisão convexas e lineares por partes do modelo $\max_{i=1,...,k} \{\omega_i x \geq \gamma_i\}$ (RYOO, 2006) como ilustrado na figura a seguir.

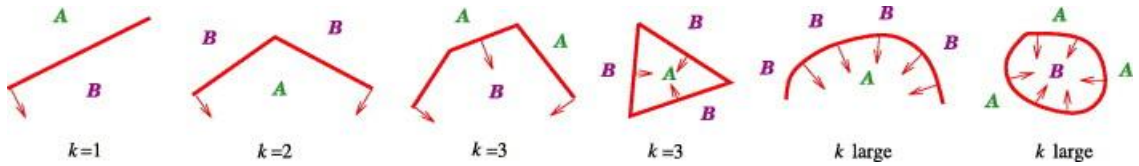


Figura 13: Exemplos de classificadores convexos e lineares por partes

O modelo considera tanto a sinergia como a execução individual dos hiperplanos k na construção da superfície de decisão.

Seguem aqui algumas observações a respeito da notação: $\bullet_i$ denotam dados do tipo $\bullet$, onde $\bullet \in \{A, B\}$ e $\overline{\bullet}$ denota o complemento do tipo $\bullet$ em relação ao conjunto $\{A, B\}$; $m_\bullet$ representa m

pontos do tipo $\bullet$ de dado, $\bullet \in \{A,B\}$ no conjunto de treinamento; K denota o conjunto de índice dos hiperplanos k cujos parâmetros devem ser determinados.

O ponto A_i é mal classificado se o dado esta localizado acima, com relação à direção das setas indicadas, os dois hiperplanos enquanto a observação B_j é mal classificada se está abaixo de qualquer um dos dois hiperplanos. Utilizando a métrica da norma-1, o total de má classificação associada com os dados A_i e B_j são mensurados por $y_{ik} = max_k\{\omega_k A_i - \gamma_k, 0\}$ e $z_{jk} = max_k\{-\omega_k B_j + \gamma_k, 0\}$, respectivamente, e um classificador robusto pode ser obtido por minimizar o total de erros de classificação.

$$\sum_{i=1}^{m_{\mathscr{A}}} \prod_{k\in K} y_{ik} + \sum_{j=1}^{m_{\mathscr{B}}} \sum_{k\in K} z_{jk}.$$

Isso gera a formulação da programação não-linear a seguir para a separação convexa e por partes k dos dois conjuntos.

$$\min_{\omega,\gamma,\mathbf{y},\mathbf{z}} \sum_{i=1}^{m_\bullet} \prod_{k\in K} y_{ik} + \sum_{j=1}^{m_{\bar{\bullet}}} \sum_{k\in K} z_{jk}$$

$$\begin{aligned} \text{s.t. } & \bullet_i\omega_k - \gamma_k + y_{ik} \geqslant \varepsilon, \qquad i = 1,\ldots,m_\bullet, k \in K \\ & \bar{\bullet}_j\omega_k - \gamma_k - z_{jk} \leqslant -\varepsilon, \qquad j = 1,\ldots,m_{\bar{\bullet}}, k \in K \\ & \mathbf{y} \in \mathbb{R}_+^{m_\bullet \times |K|}, \mathbf{z} \in \mathbb{R}_+^{m_{\bar{\bullet}} \times |K|}, \end{aligned}$$

onde $\bullet \in \{A,B\}$. Acima, ε é um número positivo introduzido para garantir uma separação forte dos dois conjuntos e pode ser substituído pela unidade mediante normalização (RYOO, 2006 apud BENNETT; MANGASARIAN, 1994). Um ponto A_i é classificado corretamente se qualquer y_{ik} se reduz a zero, isto é, o ponto é mal classificado se o mínimo dos y_{ik} forem estritamente maiores que 0.

4.3 Metodologia de Validação

4.3.1 Geração dos Hiperplanos

Seguindo a metodologia apresentada em Ryoo (2006), na fase de treinamento cada modelo é implementado e utilizado juntamente com os pontos do grupo de treino para gerar os hiperplanos classificadores. Para cada conjunto a ser avaliado o modelo é executado 30 vezes, para os trinta grupos de treino, gerando assim 30 hiperplanos classificadores que serão utilizados na classificação dos 30 grupos de dados selecionados para teste.

4.3.2 Teste

A fase de testes consiste em lançar na função classificadora $f(x) = w'x - \gamma$ os valores das características de cada um dos pontos do subconjunto de teste, cuja classe é previamente conhecida. O atributo de classe, porém, não é utilizado diretamente na realização dos cálculos que resultam na classificação do ponto, sendo conhecido apenas para fins da avaliação do modelo e geração das estatísticas qualitativas.

Lançados os pontos de teste, é definido então se os valores resultantes da função classificadora é maior que zero ou se é menor ou igual a zero, ou seja, se pertencente ao grupo A, ou se pertencente ao grupo B, respectivamente. Em aplicações reais esta informação (de que "lado" se encontra o ponto) é a incógnita que se busca encontrar.

É calculada a média de acerto dos hiperplanos classificadores utilizando os valores dos pontos dos 30 grupos de teste, bem como os desvios-padrão.

5 *Experimento*

Para avaliação extensiva dos modelos RLP e FSLP analisados neste trabalho foram realizadas as etapas de obtenção, formatação e organização dos dados, geração dos hiperplanos através da execução do modelo nos dados de treino e obtenção dos resultados pela comparação dos pontos de teste com o hiperplano classificador. Todo o experimento foi executado com os seguintes equipamentos e programas: notebook de configuração Intel CORE i3, 1.40 Ghz, RAM 4GB; Sistema Operacional Windows 7 Home Basic; linguagem Ruby versão 1.9.3 (para implementação dos programas de formatação e manipulação dos arquivos dos dados de treino e teste); IBM ILOG CPLEX Optimization Studio versão: 12.5 (para implementação e execução dos modelos na fase de treino); Microsoft Excel Starter 2010 (para comparação dos pontos ao hiperplano e cálculo das estatísticas dos resultados).

Foram avaliados cinco conjuntos de dados, obtidos em *UCI Repository of Machine Learning Databases and Domain Theories* (BACHE; M.LICHMAN, 2013). Os conjuntos foram testados para os modelos de PL para classificação RLP e FSLP, segundo metodologia apresentada em Ryoo (2006) e seus resultados experimentais foram comparados aos do modelo PCL obtidos também em Ryoo (2006).

5.1 Conjuntos de dados testados

Para este experimento, os cinco conjuntos de dados obtidos em Bache e M.Lichman (2013), foram *Australian Credit Card* de Quinlan (1999), *Boston Housing* de Jr. e Rubinfeld (1978), *Cleveland Heart Disease* de Detrano e Janosi (1989), *Congressional Voting* de Schimmer (1985) e *Wisconsin Breast Cancer* de Wolberg e Mangasarian (1990).

Os dados se apresentam originalmente em arquivos contendo um vetor em cada uma de suas linhas, contendo os atributos de classificação e o atributo de classe. Sendo representados genericamente como [$atributo_1$, $atributo_2$, ..., $atributo_n$, $classe$]. A seguir, breve descrição com informações mais relevantes e especificidades de cada um deles.

5.1.1 *Australian Credit Approval*

Este conjunto de dados diz respeito a aplicações de aprovação de cartão de crédito. Possui originalmente 690 instâncias, sendo 307 aprovações e 383 negações. As instâncias são constituídas de 14 atributos (mais atributo referente à classe). No caso deste grupo todos os nomes e valores foram modificados por emblemas sem significado para proteger a confidencialidade dos dados.

O conjunto é interessante porque possui uma boa variedade de atributos (continuos, discretos com poucos número de valores e discretos com maior número de valores). Dos 14 atributos que possui 6 são contínuos e 8 são discretos. As legendas de alguns atributos também foram modificadas para a conveniência de algoritmos estatísticos. O atributo 4, por exemplo, possuía as legendas p, g e gg, que foram trocadas por 1, 2 e 3. Segundo informações disponíveis no repositório em que se encontrava este conjunto, o mesmo possuía originalmente 37 instâncias com um ou mais valores ausentes, que foram substituídos pela Moda do atributo ausente (no caso dos discretos) e pela Média (no caso dos contínuos).

5.1.2 *Boston Housing Data*

Os dados de *Boston Housing Data* dizem respeito a valores habitacionais nos subúrbios de Boston como taxa de criminalidade per capita, média de números de cômodos por domicílio e índice de acessibilidade às rodovias radiais, por exemplo. Possui 506 instâncias constituídas de 14 atributos, sendo 13 contínuos (incluindo o atributo de classe) e um atributo binário.

O atributo de classe MEDV - *Median value of owner-occupied homes in $1000's* representa o valor da médio (em $1.000's) das casas ocupadas por seus proprietários. Por ser contínuo é separado em valores a partir de $21.000 (grupo A) e abaixo desse valor (grupo B). O conjunto não possui instâncias com dados imcompletas.

5.1.3 *Heart Disease Databases*

Este conjunto é constituído de dados que visam diagnosticar a doença cardíaca angina. Contém originalmente 76 atributos, mas os experimentos já realizados com este grupo anteriormente referem-se ao uso de um subconjunto de 14 deles. O atributo de classe refere-se à presença de doença do coração no paciente e se apresenta na forma de um inteiro avaliado entre 0 a 4. Experimentos com os dados *Cleveland Heart Disease* tem se concentrado em simplesmente distinguir presença (valores 1, 2, 3, 4) de ausência (valor 0).

Com um total de 303 instâncias, o grupo possui 164 ausências e 139 ocorrências da doença.

Nomes e números de identidade dos pacientes foram removidos do conjunto de dados, substituídos por valores fictícios.

5.1.4 *1984 United States Congressional Voting Records Database*

Este conjunto de dados contém informações sobre votos de cada um dos Congressistas da Câmara dos Deputados dos EUA. São enumerados nove tipos diferentes de votos: votou, pareados por, e anunciou para (estes três simplificado para sim); votou contra, emparelhado contra, e anunciou contra (estes três simplificado para não); votou presente, votou presente para evitar conflito de interesses, e não votaram ou não fazer uma posição conhecida (estes três simplificada para um tratamento desconhecido), estes últimos não serão considerados para o experimento.

Contem 435 instâncias, sendo 267 ocorrências da classe "democratas" e outas 168 de "republicanos". Possui 16 atributos, além do atributo de classe, todos booleanos. Os valores ausentes são denotados por "?" e os vetores que contém este valor serão retirados do conjunto antes da submissão aos modelos.

5.1.5 *Wisconsin Breast Cancer Database*

Este conjunto diz respeito a diagnósticos de câncer de mama e seus atributos são avaliados para determinar se o câncer é maligno ou benigno. Essas informações são provenientes de imagens digitalizadas de uma *fine needle aspirate* (FNA), traduzido como aspiração de agulha fina, de uma massa mamária e descrevem as características do núcleo da célula presente na imagem.

Totalizando 9 atributos, mais o atributo de classe, originalmente possui 699 instâncias, sendo 458 com diagnóstico Benigno e 241 com diagnóstico Maligno. Valores indisponíveis contidos no grupo são denotados por "?" e fazem com que 16 instâncias sejam excluídas antes da submissão dos dados aos modelos.

5.1.6 Pré-processamento dos dados (eliminando instâncias inconsistentes)

Como já mencionado em tópicos anteriores, os conjuntos de dados *Breast Cancer*, *Cleveland Heart Disease* e *Congress Voting*, apresentam problemas como inconsistência de valores nos atributos de algumas de suas instâncias. Esses dados foram tratados através de programas, escritos na linguagem Ruby. Para estes conjuntos foi realizada a subtração das instâncias problemáticas.

Como cada instância se encontra em uma linha diferente dos arquivos originais, o programa

realiza a abertura e leitura dos arquivos de dados (de extensão .txt), armazenando as informações em um vetor contendo as linhas presentes no arquivo. Depois é criado um outro arquivo para inserir os dados válidos, e através da utilização de expressões regulares, são identificas as linhas que contém inconsistência, descartando-as, inserindo apenas as instâncias válidas no arquivo de destino.

Os dados válidos de cada conjunto utilizado no experimento são resumidos em termos do número de atributos e observações na Tabela a seguir.

Conjunto de dados	Nº atributos	Nº Pontos	Classe *A*	Classe *B*
Australian Credit Approval	14	690	307 (Aprovação)	383 (Negação)
Boston Housing	13	506	260 ($\geq$ \$21.000)	246 ($<$ \$21.000)
Cleveland Heart	13	297	137 (Doente)	160 (Sem doença)
Congressional Voting	16	232	124 (Democrata)	108 (Republicano)
Winsconsing Breast Cancer	9	683	239 (Benigno)	444 (Maligno)

Tabela 2: Descrição dos conjuntos de dados

5.2 Implementação e Validação dos modelos

As precisões de teste dos classificadores foram calculadas como no passo a passo descrito em Ryoo (2006). A sequência é mostrada a seguir:

1. dividir aleatoriamente o conjunto de dados em duas partes com *P%* de dados no conjunto de treinamento e o restante (100 - *P*) dos dados no conjunto de teste;
2. treinar um classificador no conjunto de dados de treinamento e mensurar a precisão de treinamento do classificador;
3. mensurar a precisão do classificador gerado no conjunto de teste;
4. repetir, passos 1-3, 30 vezes para calcular as precisões médias de acerto e respectivos desvios padrão.

5.2.1 Formatação dos dados para submissão aos modelos

Os vetores de cada grupo são permutados e submetidos ao modelo classificador 30 vezes, fazendo com que sejam obtidos 30 grupos de treino gerando 30 hiperplanos classificadores para cada um dos 10 grupos. Os hiperplanos gerados são utilizados para classificar os 30 conjuntos

de teste correspondentes a cada grupo. A partir dos 30 resultados obtidos na avaliação dos erros e acertos do grupo de teste com relação aos seus respectivos hiperplanos classificadores é que se obtém a média final de desempeno do classificador naquele grupo.

Visando alcançar este objetivo foram escritos programas, também na linguagem Ruby, para cada um dos cinco conjuntos de dados analisados. Foram geradas as pastas e arquivos separando os dados a serem utilizados no treinamento do classificador e seus respectivos conjuntos de teste, como exemplificado na Figura 14, que mostra a estrutura de pastas e arquivos gerada para o conjunto *Wisconsin Breast Cancer*.

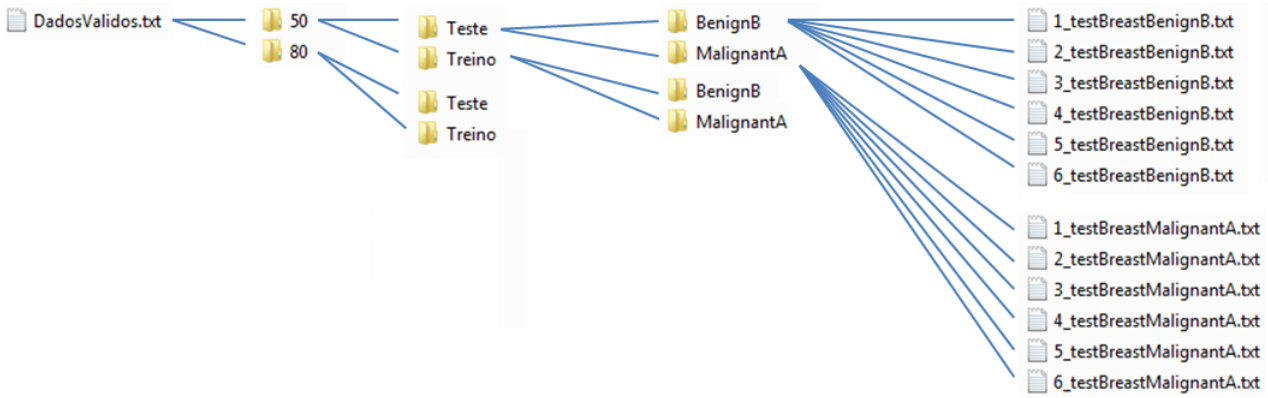


Figura 14: Ilustração da estrutura de pastas e arquivos para *Wisconsing Breast Cancer*

O programa começa realizando as seguintes tarefas: leitura do arquivo de dados e armazenamento das suas linhas em um vetor; definição da porcentagem de treino e de teste (no caso deste experimento os grupos de treino continham 50% e 80% dos dados válidos e, consequentemente, os de teste continham 50% e 20%, respectivamente.); permutação o vetor que contém as linhas do arquivo.

Para os arquivos de treino o programa primeiro cria os arquivos, já com a extensão .dat (extensão utilizada nos projetos CPlex para arquivos de dados); depois formata as linhas, através de expressões regulares, para que se adequem a sintaxe do CPlex e finalmente insere as linhas na quantidade previamente definida no arquivo de treino. Na sequência o programa cria os arquivos de teste, realiza ajustes e insere as linhas restantes no arquivo teste de destino.

5.2.2 Treinamento dos hiperplanos classificadores

Cada conjunto de dados foi avaliado duas vezes, porém com proporções de treino e teste distintas. Uma abordagem trabalha com 50% dos dados sendo usados para treino e 50% para teste, já a outra trabalha com 80% para treino e 20% para teste. Assim sendo, com cinco grupos de

dados e duas abordagens sobre cada um deles resultam em 10 grupos a serem submetidos aos, treino e teste, dos modelos.

Os modelos são implementados e incluídos em um projeto criado no CPlex para cada conjunto avaliado. Juntamente com o modelo são adicionadas as pastas contendo os vetores de treino. A Figura 15 mostra o projeto do conjunto *Wisconsin Breast Cancer* para 80% de treino.

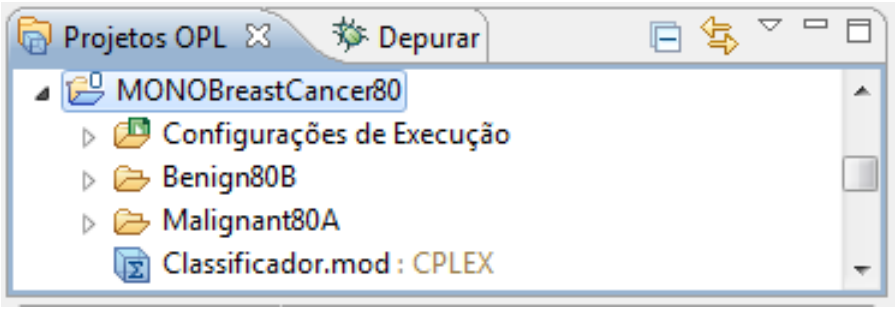


Figura 15: Implementação

Para a execução dos modelos implementados foram criadas as 30 configurações de execução contendo modelo e dados de treino correspondentes para então rodar os modelos e finalmente obter os valores da funções que geram os hiperplanos classificadores. A Figura 16 a seguir ilustra como se apresentam as configurações no software CPlex.

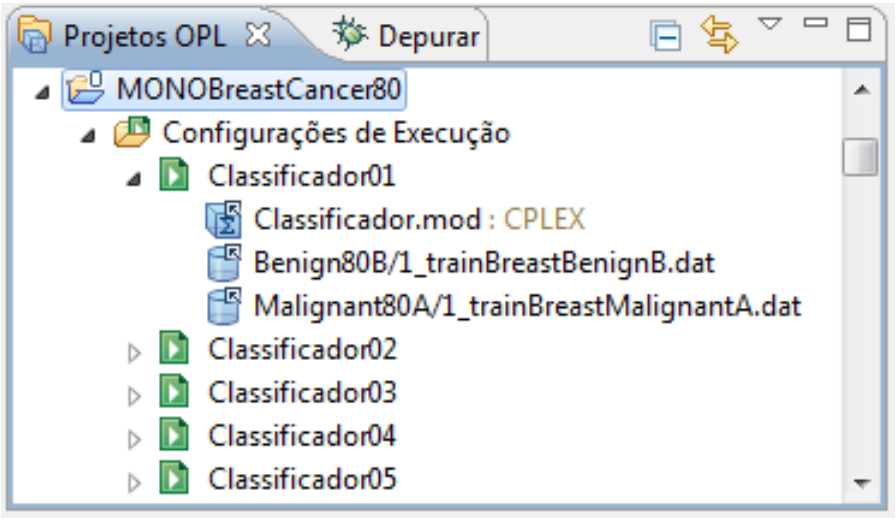


Figura 16: Treinamento

A Figura 17 mostra os resultados da execução do modelo RLP para *Wisconsing Breast Cancer* no grupo de treino número 1.

Para o modelo FSLP foi necessário ainda realizar o ajuste do μ. Este parâmetro regula a supressão ou manutenção das características a serem utilizadas na determinação do hiperplano

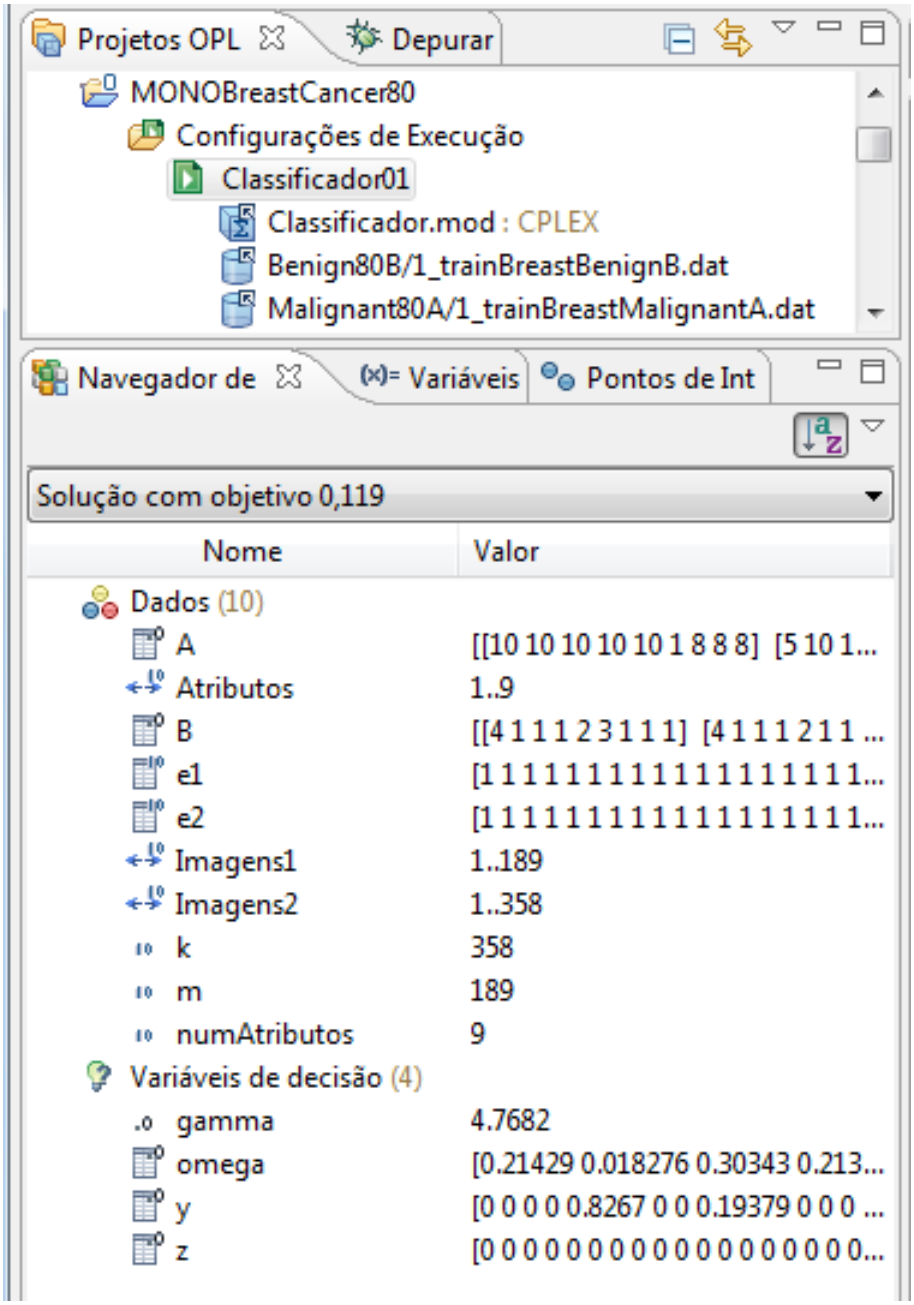


Figura 17: Execução RLP para *Wisconsing Breast Cancer* no grupo de treino nº 1

classificador. O μ é determinado experimentalmente de acordo com a melhora nos resultados da classificação nos pontos de teste.

5.3 Teste

A eficácia da classificação foi aferida através da comparação da localização do ponto de teste de um lado ou outro do hiperplano classificador $w'x = \gamma$ gerado pelo modelo com auxílio dos pontos de treino. Na etapa de teste os valores dos pontos de teste são lançados na função classificadora $f(x) = w'x - \gamma$. Se o valor da função no ponto de teste satisfizer às condições $f(x) > 0$ se $x \in A$ e $f(x) \leq 0$ se $x \in B$ significa que o ponto foi classificado corretamente, senão será considerado mal classificado.

Através dos reultados desta comparação nas 30 execuções do modelo em cada conjunto é que foram geradas as estatísticas finais.

5.4 Resultados

Nesta seção são apresentadas as estatísticas de resultados obtidas com os modelos RLP e FSLP para cada um dos cinco conjuntos e comparadas às obtidas para o modelo PLC em Ryoo (2006). As tabelas a seguir resumem as taxas (em porcentagem de acerto) e respectivos desvios padrão.

A primeira linha de cada tabela informa o conjunto de dados testado. A primeira coluna informa o modelo avaliado; a segunda e terceira colunas informam as porcentagens de precisão de treino e teste, respectivamente, para a abordagem de 50% de treino e 50% de teste; as terceira e quarta colunas informam precisão de treino e teste para a abordagem de 80% de treino e 20% de teste. As estatísticas de treino correspondem a média de acerto com respectivos desvios padrão e são apresentadas a seguir:

***Congress Voting* (% de acerto)**								
Modelo	50% Treino		50% Teste		80% Treino		20% Teste	
RLP	99,6%	$\pm 2,1$	**94,3%**	$\pm 2,2$	99,2%	$\pm 3,7$	**93,6%**	$\pm 3,6$
FSLP (μ = 0,01)	96,9%	$\pm 5,6$	**94,8%**	$\pm 5,5$	96,1%	$\pm 5,6$	**95,6%**	$\pm 5,3$
PLC (k = 1)	100%	± 0	**94,5%**	$\pm 1,8$	100%	± 0	**94,5%**	$\pm 1,8$

Tabela 3: Performance dos modelos para *Congress Voting*

***Australian Credit Approval* (% de acerto)**								
Modelo	50% Treino		50% Teste		80% Treino		20% Teste	
RLP	86,2%	$\pm 1,7$	**85,4%**	$\pm 1,4$	85,8%	$\pm 0,7$	**85,0%**	$\pm 2,7$
FSLP (μ = 0,01)	85,8%	$\pm 1,4$	**85,5%**	$\pm 1,3$	85,7%	$\pm 0,7$	**85,4%**	$\pm 3,0$
PLC (k = 1)	86,5%	$\pm 1,3$	**86,4%**	$\pm 1,3$	86,8%	$\pm 0,6$	**85,4%**	$\pm 2,5$

Tabela 4: Performance dos modelos para *Australian Credit Card*

***Wisconsin Breast Cancer* (% de acerto)**								
Modelo	50% Treino		50% Teste		80% Treino		20% Teste	
RLP	97,7%	$\pm 0,7$	**96,7%**	$\pm 0,8$	97,5%	$\pm 0,3$	**97,0%**	$\pm 1,4$
PLC (k = 1)	97,6%	$\pm 0,6$	**96,3%**	$\pm 0,8$	97,5%	$\pm 0,4$	**95,8%**	$\pm 1,4$

Tabela 5: Performance dos modelos para *Wisconsin Breast Cancer*

***Boston Housing* (% de acerto)**								
Modelo	50% Treino		50% Teste		80% Treino		20% Teste	
RLP	89,4%	$\pm 1,9$	**85,5%**	$\pm 1,9$	88,3%	$\pm 0,9$	**85,3%**	$\pm 3,0$
PLC (k = 1)	88,9%	$\pm 1,8$	**85,9%**	$\pm 1,5$	88,2%	$\pm 0,9$	**85,3%**	$\pm 3,3$

Tabela 6: Performance dos modelos para *Boston Housing*

***Cleveland Heart Disease* (% de acerto)**								
Modelo	50% Treino		50% Teste		80% Treino		20% Teste	
RLP	86,9%	$\pm 3,0$	**81,1%**	$\pm 3,4$	85,8%	$\pm 1,4$	**81,0%**	$\pm 5,5$
PLC (k = 1)	87,2%	$\pm 2,2$	**81,8%**	$\pm 2,2$	85,3%	$\pm 0,9$	**83,4%**	$\pm 3,3$

Tabela 7: Performance dos modelos para *Cleveland Heart Disease*

Os resultados apresentados para o modelo RLP mostram que as precisões de acerto nos pontos de treino são bastante similares às obtidas nos conjuntos de teste. Isso torna possível avaliar o desempenho do modelo em uma aplicação sem que seja necessário testar o classificador nos demais pontos. Esta informação é bastante importante em aplicações do mundo real em que os dados a serem testados possuírem classe realmente desconhecida.

As taxas de acerto se mostraram satisfatórias (acima de 80%), demonstrando consistência nas comparações entre os modelos para cada grupo de dados. Mas quando se comparam os resultados entre os conjuntos as taxas de acerto variam mais amplamente. As melhores marcas foram as dos conjuntos *Congress Votin* e *Wisconsing Breast Cancer* que obtiveram taxas de acerto acima de 90%.

O modelo RLP obteve taxas semelhantes aos dos outros modelos estudados, embora seus resultados inferiores em 3 dos conjuntos de dados numa margem de 2%. Na avaliação do modelo FSLP, para três dos conjuntos de dados estudados (*Boston Housuing*, *Cleveland Heart Disease* e *Wisconsing Breast Cancer*) não foi encontrado $\mu \neq 0$ que contribuísse na melhora da classificação. Isso ocorre provavelmente pelo fato desses conjuntos já apresentarem reduzida quantidade de atributos 13, 13 e 9, respectivamente, fazendo com que a tentativa de supressão de qualquer um deles fosse prejudicial à classificação. Para os outros dois (*Australian Credit Approval* e *Congress Voting*) houve melhora (para $\mu = 0{,}01$), porém bastante discreta. Também pelo fato de, apesar de apresentarem quantidade de atributos um pouco maior que os demais, 15 e 16 respectivamente, o número de atributos ainda ser reduzido.

6 *Conclusão*

6.1 Considerações Finais

Para os cinco conjuntos de dados estudados, foi avaliada a eficiência dos modelos de classificação via programação linear RLP e FSLP, que buscam gerar um hiplerplano classificador único, sobre os resultados obtidos pelo modelo PLC em Ryoo (2006), que realiza a classificação de dados através de uma superfície constituída pela união de um conjunto de hiperplanos.

Este objetivo se concretizou no que tange as estatísticas de resultados bastante expressivas e comparáveis às do modelo PLC para $k = 1$, que por sua vez são superiores aos resultados de PLC para $k = 2$ e $k = 3$ apresentados em Ryoo (2006). Sendo o experimento bem sucedido para todos os conjuntos de dados estudados, o conjunto *Wisconsing Breast Cancer* o modelo RLP foi inclusive superior ao PLC (para $k = 1$), atingindo a marca de 96,7% de acerto contra 96,3% (na abordagem 50% Treino) e 97,0% contra 95,8% (na abordagem com 80% de Treino).

O presente estudo corrobora com a idéia de que um hiperplano único pode ser um separador mais robusto para a classificação de dados linearmente inseparáveis citada por Ryoo (apud HOLTE, 1993). Visto que a precisão média dos classificadores PLC, para os mesmos conjuntos de dados utilizados neste trabalho, eram em sua maioria menores a medida que aumentavam o número de hiperplanos gerados em Ryoo (2006) na superfície final de classificação. A princípio supõe-se que uma superfície que tenha mais capacidade de ajuste sobre dados linearmente inseparáveis performe melhor classificação, mas para Ryoo (2006) uma explicação plausível é a ocorrência de um superajuste aos dados de treino, fazendo o classificador não performar tão bem no conjunto de teste.

Visto que os modelos classificadores estudados, que utilizam um hiperplano único, se mostraram tão robustos quanto o modelo de classificação multilinear, no que diz respeito à qualidade da classificação, os modelos lineares tem a vantagem de serem computacionalmente mais leves, dado que programas multilineares são computacionalmente de difícil solução (RYOO, 2006).

Para o modelo FSLP houve a dificuldade em encontrar um $\mu \neq 0$ que interferisse satisfato-

riamente na classificação dos pontos em relação ao modelo PLC. Mas dado que fazendo $\mu = 0$, o FSLP é "reduzido" ao modelo ao RLP, seus resultados atingem no mínimo os resuldados obtidos por RLP, considera-se então que seus resultados são igualmente satisfatórios.

Foi encontrado experimentalmente $\mu = 0,01$ em ambos os casos testados para FSLP (*Australian Credit Approval e Congressional Voting*). Houve melhora, porém muitíssimo discreta com menos de 0,5% de aumento da média de precisão em quase todos os testes, tendo sua melhor marca no conjunto *Congressional Voting* para 80% de Treino, atingindo uma melhora de 2,0% na média dos resultados. A maior vantagem do modelo FSLP para o conjunto *Congressional Voting* tanto na abordagem de 50% quanto na de 80% foi a redução do número de características usadas reduzindo para 3 (das 14 originais), o que é bastante significativo.

Os resultados para os diferentes grupos foi bastante heterogênea, porém a capacidade classificadora do hiperplano é limitada pela natureza da aplicação. Se o grau de inseparabilidade dos pontos for muito alto, mesmo o hiperplano linear ótimo não performará satisfatoriamente no conjunto.

6.2 Trabalhos Futuros

Para melhor aproveitamento do modelo seletor de características FSLP sugere-se que o mesmo seja utilizado em aplicações cujos conjuntos de dados possuam um maior número de atributos. Fazendo com que o refinamento na quantidade de características usadas seja mais bem aproveitado.

O estudo da automatização da busca pelo melhor valor para o parâmetro μ é uma proposta importante. Visto que, apesar do modelo FSLP não requerer interação com o usuário na seleção de quais características selecionar, a escolha de μ é feita experimentalmente, não havendo portando garantias de que seja a ótima.

Em alguns conjuntos de dados houve ainda a ocorrência de piora da performance para o conjunto de 80% treino em relação ao de 50%, o que não era desejável ou esperado. Apesar da diferença não ser muita, uma hipótese a se considerar é que a forma de validação dos dados não seja a ideal. Para fins de futuras verificações seria propícia a repetição do experimento através da utilização de outra metodologia de validação.

Referências Bibliográficas

ARAúJO, P. F. da S. *Programação Linear e suas Aplicações: Definição e Métodos de Solução*. Universidade Federal de Goiás - UFG, 2013. Disponível em: <http://bit.profmat-sbm.org.br/xmlui/handle/123456789/541>.

ASTI, P. L. *Anotador Mosfossintático para o português-twitter*. Rio de Janeiro: PUC-Rio - Departamento de informática, Abril 2011.

BACHE, K.; M.LICHMAN. *UCI Machine Learning Repository*. 2013. Disponível em: <http://archive.ics.uci.edu/ml/datasets.html>. Acesso em: 01/11/2013.

BAZARAA, M. S.; SHERALI, H. D.; JARVIS, J. J. *Linear programming and network flows*. 2ª. ed. Nova York: Wiley, 1990.

BENNETT, K. P.; MANGASARIAN, O. *Robust Linear Programming of Two Linearly Inseparable Sets*. [S.l.], Outubro 1991.

BENNETT, K. P.; MANGASARIAN, O. L. Bilinear separation of two sets in n-space. *Computational Optimization and Applications*, v. 2, 1994.

BRADLEY, P.; MANGASARIAN, O. L. Feature selection via concave minimization and support vector machines. In: *Machine Learning Proceedings of the Fifteenth International Conference (ICML '98)*. Califórnia: Morgan Kaufmann, 1998. p. 82–90.

DETRANO, R.; JANOSI, A. *International application of a new probability algorithm for the diagnosis of coronary artery disease*. [S.l.]: American Journal of Cardiology, 1989.

FERREIRA, A. B. de H. *Miniaurélio Século XXI Escolar: O minidicionário da lingua portuguesa*. 2ª. ed. Rio de Janeiro: Nova Fronteira, 2001.

GUO, G.; DYER, C. R. Simultaneous features selection and classifier training via linar programming: A case study for face expression recognition. 2003.

HILLIER, F. S.; LIEBERMAN, G. J. *Introdução à Pesquisa Operacional*. 4ª. ed. São Paulo: McGraw-Hill, 2009.

HOLTE, R. C. Very simple classification rules perform well on most commonly used datasets. In: *Machine Learning*. [S.l.: s.n.], 1993. p. 63–91.

JR., D. H.; RUBINFELD, D. L. *Hedonic prices and the demand for clean air*. 1978.

JR., O. L.; MONTGOMERY, E. *Redes Neurais - Fundamentos e aplicações com programas em C*. Rio de Janeiro: Ciência Modrena, 2007.

LACHTERMACHER, G. *Pesquisa Operacional na tomada de decisões*. 8ª. ed. São Paulo: Pearson - Prentice Hall, 2006.

LISBOA, E. F. A. *Pesquisa Operacional*. Rio de Janeiro: [s.n.], Fevereiro 2002. Disponível em: <http://www.ericolisboa.eng.br>. Acesso em: 17/12/2013.

LORENA, A. C.; CARVALHO, A. C. P. L. F. de. *Uma introdução às Suport Vector Machines*. [S.l.]: RITA - Revista de Informática Técnica e Aplicada, 2007.

PEREIRA, M.; LINS, E.; CALôBA, G. M. *Programação Linear com aplicações em teoria dos jogos e avaliação de desempenho*. Rio de Janeiro: Interciência, 2006.

QUINLAN, J. R. Simplifying decision trees. *Int. J. Hum.-Comput. Stud*, v. 51, p. 497, 1999.

RYOO, H. S. Pattern classification by concurrently determined piecewise linear and convex discriminat functions. ScienceDirect - Computers & Industrial Engineering, Agosto 2006.

SCHIMMER, J. *Congressional Quartely Almanac*. Washington, D.C.: [s.n.], 1985.

TAFNER, M. A.; XERES, M. de; FILHO, I. W. R. *Redes Neurais Artificiais - Introdução e Princípios de Neurocomputação*. Blumenau: EKO, 1995.

WOLBERG, W. H.; MANGASARIAN, O. L. *Multisurface method of pattern separation for medical diagnosis applied to breast cytology*. [S.l.]: Proceedings of the National Academy of Sciences, Dezembro 1990.