Universidade Federal de São Carlos
Centro de Ciências Exatas e de Tecnologia
Programa de Pós Graduação em Estatística
Departamento de Estatística

Modelos de Regressão Logística Clássica, Bayesiana e Redes Neurais para *Credit Scoring*

Tiago Silva Mendonça
Orientador: Prof. Dr. Francisco Louzada-Neto

Dissertação apresentada ao Departamento de Estatística da Universidade Federal de São Carlos – DEs/UFSCar, como parte dos requisitos para obtenção do título de Mestre em Estatística

São Carlos
2008

Ficha catalográfica elaborada pelo DePT da Biblioteca Comunitária da UFSCar

M539mr

Mendonça, Tiago Silva.
Modelos de regressão logística clássica, Bayesiana e redes neurais para Credit Scoring / Tiago Silva Mendonça. -- São Carlos : UFSCar, 2009.
177 f.

Dissertação (Mestrado) -- Universidade Federal de São Carlos, 2008.

1. Análise de regressão. 2. Regressão logística. 3. Inferência bayesiana. 4. Redes neurais. I. Título.

CDD: 519.536 (20ª)

Sumário

Resumo

Importantes avanços vêm sendo conquistados na área de concessão de crédito, não obstante, o problema de identificação de bons clientes para a concessão de crédito não apresenta uma solução definitiva. Diversas técnicas foram apresentadas e vêm sendo desenvolvidas, cada uma apresenta suas características, com vantagens e desvantagens no tocante ao seu poder de discriminação, robustez, facilidade de implementação e possibilidade de interpretação.

Este trabalho apresenta três técnicas para a classificação de inadimplência em modelos de Credit Score, Regressão Logística Clássica, Regressão Logística Bayesiana com priori não informativa e Redes Neurais Artificiais com algumas diferentes arquiteturas. O objetivo principal do trabalho é comparar o desempenho destas técnicas na identificação de clientes inadimplentes. Para isto, Foram utilizadas quatro métricas para a comparação dos modelos: Capacidade Preditiva, Curva ROC, Estatística de Kolmogorov Smirnov e a Capacidade de Acerto dos modelos. Dois bancos de dados foram utilizados, um banco artificial e um banco real. O banco de dados artificial foi construído baseado em um artigo de Breiman que gera as variáveis explicativas a partir de uma distribuição normal multivariada e o banco de dados real utilizado trata-se de um problema de Credit Score de uma instituição financeira que atua no mercado varejista brasileiro há mais de vinte anos.

Palavras-chave: Regressão Logística, Inferência Bayesiana, Redes Neurais Artificiais.

Abstract

Important advances have been achieved in the granting of credit, however, the problem of identifying good customers for the granting of credit does not provide a definitive solution. Several techniques were presented and are being developed, each presents its characteristics, advantages and disadvantages as to their discrimination power, robustness, ease of implementation and possibility of interpretation.

This work presents three techniques for the classification of defaults in models of Credit Score, Classical Logistic Regression, Bayesian Logistic Regression with no prior information and Artificial Neural Networks with a few different architectures. The main objective of the study is to compare the performance of these techniques in the identification of customers default. For this, four metrics were used for comparison of models: predictive capacity, ROC Curve, Statistics of Kolmogorov Smirnov and capacity of hit models. Two data bases were used, an artificial bank and a real bank. The database was constructed artificially based on an article by Breiman that generates the explanatory variables from a multivariate normal distribution and the actual database used is a problem with Credit Score of a financial institution that operates in the retail Brazilian market more than twenty years.

Keywords: Logistic Regression, Bayesian Inference, Artificial Neural Networks.

Capítulo 1

Introdução

Nos últimos vinte e cinco anos as decisões relacionadas ao crédito deixaram de ser baseadas apenas na experiência do analista de crédito, atualmente são baseadas em técnicas estatísticas desenvolvidas, suas principais ferramentas de crédito. Antes as correções inflacionárias compensavam as perdas de crédito. Com a estabilidade da moeda houve um re-aquecimento da economia e um grande aumento da demanda por crédito. A falta de uma cultura de crédito forte fez com que as instituições financeiras não conseguissem conciliar a qualidade à demanda, resultando em aprovações equivocadas e/ou na rejeição do crédito rentável. Financiadoras, como bancos e companhias de cartão de crédito, utilizam a análise de *credit score* para avaliar o risco potencial de emprestar dinheiro aos consumidores reduzindo perdas devido aos maus pagadores.

Um *Credit Score* é uma pontuação que representa a confiabilidade de uma pessoa reembolsar o dinheiro que é devido. O *Credit Score* é construído através de informações de crédito obtidas geralmente através de agências de crédito, isto é, companhias que fornecem informações de crédito sobre empréstimos individuais.

Existem várias formas para se determinar o *credit score* de um cliente, o objetivo deste trabalho é apresentar, aplicar e comparar três delas: Regressão Logística Clássica, Regressão Logística Bayesiana e Redes Neurais.

A técnica estatística mais popular utilizada em modelagem de *Credit Score* é a regressão logística, que prediz uma saída binária como, por exemplo, bom e mau pagador. As financiadoras movimentam um grande volume de transações e utilizam *credit score* para determinar se autorizam ou não determinado empréstimo, quando autorizado determina a que taxa de juros e quais serão os limites de crédito. A técnica permite que essas decisões sejam tomadas de forma objetiva, padronizada e imparcial além de permitir a automatização do processo, evitando desperdícios, reduzindo custos e permitindo a maximização dos resultados. A concessão de créditos é atualmente uma das principais fontes de receita de bancos e instituições financeiras. Essas empresas vêm buscando continuamente por modelos cada vez mais aprimorados.

Com o avanço computacional, técnicas de simulação relativamente simples mas extremamente poderosas puderam ser implementadas e a área que muito se beneficiou com este aprimoramento foi a inferência Bayesiana, onde a capacidade de calcular integrais de funções complexas e de muitas dimensões é de extrema importância.

A inferência Bayesiana permite o acréscimo da opinião do especialista na modelagem através de uma informação *a priori*. Em regressão logística o método Bayesiano utilizado consiste em definir uma informação *a priori* para cada um dos parâmetros de interesse no modelo. O modelo é encontrado após a determinação da distribuição *a posteriori* destes parâmetros.

Outro método que pode ser utilizado em modelos de *Credit Score* são modelos de redes neurais, que são modelos computacionais inspirados na estrutura e no funcionamento do cérebro biológico. Através de algoritmos de

aprendizado as ligações entre os neurônios artificiais são ponderadas resultando, após um período de aprendizado, em um modelo de classificação.

Credit Score não é limitado para bancos e instituições financeiras. Outras organizações, como companhias telefônicas e departamentos de governo empregam a mesma técnica.

Primeiramente apresenta-se as três técnicas separadamente nos Capítulos 2, 3 e 4. Nas aplicações realizadas no Capítulo 5 e no Capítulo 6 todas as técnicas foram aplicadas em um banco de dados artificial e um real respectivamente e em seguida comparadas de acordo com quatro métodos: Capacidade Preditiva, Curva ROC, Estatística KS e Capacidade de acerto dos modelos na base de dados reservada para o teste dos modelos.

No Capítulo 2 encontra-se a metodologia de Regressão Logística Clássica desde o cálculo e interpretação dos coeficientes até o teste de significância e intervalo de confiança para os parâmetros estimados.

O Capítulo 3 possui a mesma estrutura e apresenta a metodologia de Regressão Logística Bayesiana. A inclusão de informações externas ao banco de dados não foi realizada na aplicação desta técnica por falta de uma opinião de um especialista na construção das prioris. Como esperado, os resultados obtidos através da metodologia Bayesiana foram bastante próximos aos resultados obtidos pelo método clássico, pois ambos utilizam as informações dos dados através da função de verossimilhança.

O Capítulo 4 contém a técnica de redes neurais com uma breve introdução dos conceitos fundamentais, descrevendo inicialmente as definições de neurônios artificiais, funções de ativação e o conceito de aprendizagem utilizado em inteligência artificial. Foram apresentados ainda, duas arquiteturas de redes neurais: o *perceptron* e o *perceptron* multicamadas. Para cada uma das duas arquiteturas foi apresentado o algoritmo de

aprendizagem mais utilizado, o *back-propagation*, algumas limitações e também uma breve discussão sobre a capacidade de generalização dos modelos e sugestões para os parâmetros do algoritmo de aprendizado do *perceptron* multicamadas.

As técnicas de redes neurais e de regressão logística Bayesiana estão começando a ser testadas na prática em problemas de *Credit Scoring* como alternativa ao método clássico.

Nos capítulos de aplicações, comprovou-se a qualidade das técnicas tanto para os dados reais quanto para os dados artificiais, pois todas tiveram resultados muito próximos a partir dos respectivos conjuntos de dados. Dada que a regressão logística é muito utilizada na modelagem de *Credit Scoring* as técnicas de Redes Neurais e a Regressão Logística Bayesiana foram, para os conjuntos de dados utilizados neste trabalho, igualmente eficazes na modelagem de crédito com uma leve vantagem para o modelo de Redes Neurais MLP com duas camadas ocultas no ajuste para o banco de dados simulado e uma leve vantagem para o modelo de Redes Neurais MLP com uma camada oculta na aplicação dos dados reais.

Capítulo 2

Regressão Logística Clássica

Métodos de regressão são fundamentais em análises de dados onde existe interesse em descrever o relacionamento entre uma variável resposta e uma ou mais variáveis explicativas. Em muitas situações a variável resposta é dicotômica, ou seja, assume dois possíveis valores (ex: sim ou não, doente ou saudável, etc.). Nestes casos a regressão logística é amplamente utilizada, descrevendo assim uma relação entre a ocorrência ou não de um evento de interesse e um conjunto de variáveis aleatórias. Este capítulo é baseado no livro Hosmer e Lemeshow (2000).

Na modelagem de *Credit Score*, a variável resposta representa o desempenho de crédito dos indivíduos durante um determinado período de tempo, normalmente 12 meses. É coletado um conjunto de características dos indivíduos no momento em que eles solicitam o crédito, e em muitos casos observam-se também informações a respeito do próprio produto de crédito a ser utilizado, por exemplo: números de parcelas, finalidade, valor do crédito, etc. Nesse caso, o objetivo é identificar se o proponente é bom ou mau pagador.

Na prática, verifica-se que a regressão logística e a regressão linear apresentam resultados muito semelhantes na discriminação de bons e maus pagadores em um modelo de *Credit Score*. Hand e Henley (1997) atribuem essa semelhança ao fato de uma grande proporção de candidatos ao crédito possuir probabilidades estimadas de serem bons e maus pagadores entre 0,2 e 0,8 onde a curva logística é bem aproximada por uma reta.

Muitas funções de distribuição têm sido propostas para uso em análises de variável resposta dicotômica, uma das principais razões de se escolher a distribuição logística é o fato de que do ponto de vista matemático ela é de fácil aplicação.

Para modelos de resposta binária, a resposta, y, de um indivíduo ou uma unidade experimental pode assumir um de dois possíveis valores, denotados por conveniência por 1 e 0 (por exemplo, y = 1 se o indivíduo é mau pagador, caso contrário y = 0). Suponha $\boldsymbol{x}$ um vetor de variáveis explicativas $(x_1, x_2, \ldots, x_k)$, correspondendo aos valores dos atributos que caracterizam um cliente e $p = Pr(y=1|\boldsymbol{x}) = \pi(\boldsymbol{x})$ a proporção de maus pagadores em função do perfil dos clientes definido e caracterizado pelo vetor $\boldsymbol{x}$ como pode ser visto na equação 2.1.

$$\pi(\boldsymbol{x}) = \frac{e^{\boldsymbol{\beta}'\boldsymbol{x}}}{1 + e^{\boldsymbol{\beta}'\boldsymbol{x}}} \tag{2.1}$$

Pode-se interpretar $\pi(\boldsymbol{x})$ como a probabilidade de um proponente ao crédito ser um mau pagador dado as características representadas pelo vetor $\boldsymbol{x}$. A transformação de $\pi(\boldsymbol{x})$ que é utilizada para a obtenção da forma aditiva é a transformação logito. Esta transformação é definida em termos de $\pi(\boldsymbol{x})$, da seguinte forma:

$$g(\boldsymbol{x}) = ln\left[\frac{\pi(\boldsymbol{x})}{1-\pi(\boldsymbol{x})}\right] = \boldsymbol{\beta}'\boldsymbol{x} = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k \qquad (2.2)$$

A importância desta transformação é que *g(**x**)* possui várias das propriedades desejadas de um modelo de regressão linear. A logito, g(***x***) é linear nos seus parâmetros, pode ser contínua e pode variar de $-\infty$ a $+\infty$, dependendo da variação de ***x***.

Em um modelo de regressão linear assume-se que a variável resposta pode ser expressa como y = E[y|**x**]+ ε. A quantidade ε segue uma distribuição normal com média zero e variância constante. Daí segue que a distribuição condicional da variável resposta dado **x** será normal com média E(y|**x**) e uma variância constante. Isto não ocorre quando temos uma variável resposta dicotômica. Nesta situação nós podemos expressar o valor da variável resposta dado **x** como y = π(**x**) + ε. Aqui a quantidade ε pode assumir um dos dois possíveis valores:

$$\begin{cases} y = 1, & P[\varepsilon = \mathbf{1} - \boldsymbol{\pi}(\mathbf{x})] = \pi(\mathbf{x}) \\ y = 0, & P[\varepsilon = -\boldsymbol{\pi}(\mathbf{x})] = 1 - \pi(\mathbf{x}) \end{cases} \qquad (2.3)$$

Então, ε tem uma distribuição Bernoulli com média **π**(**x**) e variância igual a π(**x**)[1− π(**x**)] . Isto é, a distribuição da variável resposta segue uma distribuição Bernoulli com probabilidade dada pela média π(**x**). Dadas a distribuição da variável resposta e as variáveis explicativas o passo seguinte é a determinação dos coeficientes do modelo.

2.1 Cálculo e interpretação dos coeficientes

Considere uma amostra de n clientes do par (yi,**x**i), i = 1, 2, ..., n, onde Y_i denota o valor da variável resposta dicotômica (0: bom pagador, 1: mau

pagador) e $\mathbf{x}_i$ é o vetor constituído das k variáveis observadas para o i-ésimo indivíduo. Para ajustar um modelo de regressão logística para um conjunto de dados é necessário estimar os parâmetros β's desconhecidos. Os parâmetros são normalmente estimados através do método de máxima verossimilhança. O primeiro passo necessário para encontrar os parâmetros é a construção da função de verossimilhança. Esta função expressa, do ponto de vista clássico, a probabilidade dos dados observados como função dos parâmetros desconhecidos. As estimativas de máxima verossimilhança dos parâmetros são os valores que maximizam essa função, ou seja, maximizam, novamente do ponto de vista clássico, a probabilidade de se obter os dados observados.

Cada linha do banco de dados representa um cliente, cada cliente possui uma probabilidade de ser mau pagador, $\pi(\mathbf{x}_i)$, e o interesse é modelar essa probabilidade. Observe que, Y_i é uma variável aleatória com distribuição Bernoulli, em que:

$$P[Y_i = 1 \mid \mathbf{x}_i] = \pi(\mathbf{x}_i) \text{ e } P[Y_i = 0 \mid \mathbf{x}_i] = 1 - \pi(\mathbf{x}_i) \tag{2.4}$$

Podemos reescrever a equação (2.4) da seguinte forma:

$$P[Y_i = y_i \mid \mathbf{x}_i] = f(y_i \mid \mathbf{x}_i) = [\pi(\mathbf{x}_i)]^{y_i}.[1 - \pi(\mathbf{x}_i)]^{1-y_i},\ y_i = 0,1,\ i = 1,2,\ldots, n. \tag{2.5}$$

Note que $f(1 \mid \mathbf{x}_i) = \pi(\mathbf{x}_i)$ e $f(0 \mid \mathbf{x}_i) = 1 - \pi(\mathbf{x}_i)$.

Considerando os clientes como estatisticamente independentes então a função de verossimilhança é dada pelo produto das distribuições de marginais dadas na equação 2.5.

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n} f(y_i|\boldsymbol{x}_i) = \prod_{i=1}^{n} [\pi(\boldsymbol{x}_i)]^{y_i}[1 - \pi(\boldsymbol{x}_i)]^{1-y_i} \tag{2.6}$$

Os estimadores de máxima verossimilhança são os valores de β que maximizam a função acima, porém é matematicamente mais fácil encontrar os valores que maximizam o logaritmo natural dessa função, ou seja:

$$l(\boldsymbol{\beta}) = ln\big(L(\boldsymbol{\beta})\big) = \sum_{i=1}^{n} ln[f(y_i|\boldsymbol{x}_i)] = \sum_{i=1}^{n} \{y_i ln[\pi(\boldsymbol{x}_i)] + (1-y_i) ln[1-\pi(\boldsymbol{x}_i)]\} \quad (2.7)$$

Deriva-se a função 2.7 em relação a cada um dos β's de interesse para se encontrar os estimadores de máxima verossimilhança. Derivando-se em relação a β_0 tem-se:

$$\frac{\partial L(\boldsymbol{\beta})}{\partial \beta_0} = \sum_{i=1}^{n} \left\{ \frac{y_i}{\pi(\boldsymbol{x}_i)} \, \mathrm{A} - \frac{(1-y_i)}{\big(1-\pi(\boldsymbol{x}_i)\big)} \mathrm{A} \right\} \quad (2.8)$$

onde,

$$\mathrm{A} = \frac{\partial \pi(\boldsymbol{x}_i)}{\partial \beta_0} = \frac{e^{\boldsymbol{x}_i'\boldsymbol{\beta}}\left(1+e^{\boldsymbol{x}'\boldsymbol{\beta}}\right) - e^{\boldsymbol{x}_i'\boldsymbol{\beta}}.e^{\boldsymbol{x}_i'\boldsymbol{\beta}}}{\left(1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}}\right)^2} = \frac{e^{\boldsymbol{x}_i'\boldsymbol{\beta}}\left(1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}}\right)}{\left(1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}}\right)^2} - \frac{e^{\boldsymbol{x}_i'\boldsymbol{\beta}}.e^{\boldsymbol{x}_i'\boldsymbol{\beta}}}{\left(1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}}\right)^2}$$

$$= \pi(\boldsymbol{x}_i) - \pi(\boldsymbol{x}_i)\pi(\boldsymbol{x}_i) = \pi(\boldsymbol{x}_i)\big(1-\pi(\boldsymbol{x}_i)\big) \quad (2.9)$$

logo:

$$\frac{\partial L(\boldsymbol{\beta})}{\partial \beta_0} = \sum_{i=1}^{n} [y_i - \pi(\boldsymbol{x}_i)] \quad (2.10)$$

Derivando em relação aos demais **β**'s temos:

$$\frac{\partial L(\boldsymbol{\beta})}{\partial \beta_k} = \sum_{i=1}^{n} \left\{ \frac{y_i}{ln\big(\pi(\boldsymbol{x}_i)\big)} \, \mathrm{C} - \frac{(1-y_i)}{ln\big(1-\pi(\boldsymbol{x}_i)\big)} \mathrm{C} \right\} \quad (2.11)$$

onde,

$$C = \frac{\partial \pi(\boldsymbol{x}_i)}{\partial \beta_i} = \frac{e^{x_i'\beta} x_i \left(1+e^{x_i'\beta}\right) - e^{x_i'\beta}.e^{x_i'\beta} x_i}{\left(1+e^{x_i'\beta}\right)^2} = x_i \left[\frac{e^{x_i'\beta}\left(1+e^{x_i'\beta}\right)}{\left(1+e^{x'\beta}\right)^2} - \frac{e^{x_i'\beta}.e^{x_i'\beta}}{\left(1+e^{x_i'\beta}\right)^2}\right] =$$

$$= x_i[\pi(\boldsymbol{x}_i) - \pi(\boldsymbol{x}_i)\pi(\boldsymbol{x}_i)] = \boldsymbol{x}_i \pi(\boldsymbol{x}_i)\left(1 - \pi(\boldsymbol{x}_i)\right) \quad (2.12)$$

e

$k = 1, \ldots, p.$

Logo:

$\frac{\partial L(\boldsymbol{\beta})}{\partial \beta_k} = \sum_{i=1}^{n} \boldsymbol{x}_i[y_i - \pi(\boldsymbol{x}_i)]$, para qualquer k

Igualando as derivadas acima a zero, encontram-se as equações para a obtenção dos estimadores de máxima verossimilhança para o vetor desconhecido $\boldsymbol{\beta}$ (Farhat, 2003). Porém pode-se perceber que estas equações são não-lineares, necessitando assim da aplicação de métodos iterativos para a obtenção dos estimadores, como Newton-Raphson ou Escore de Fisher. McCullagh e Nelder (1989) mostram que a solução para as equações acima podem ser obtidas usando o procedimento iterativo de mínimos quadrados ponderados.

Após a obtenção dos estimadores é necessária a interpretação dos valores. O coeficiente de uma variável independente representa a taxa de mudança ou a inclinação da função logito para cada incremento unitário na unidade de medida da variável, considerando-se fixas as demais variáveis do modelo. Por exemplo, se $\beta_1 = 2{,}02$ significa que a cada unidade de medida incrementada em X_1 temos um acréscimo de 2,02 na inclinação da função logito.

Outra métrica utilizada para quantificar a influência das variáveis independentes na variável resposta é a *odds ratio*. A *odds ratio* para uma

variável X_1 é calculada através da divisão dos indivíduos em m grupos ($m > 1$), por exemplo, seja $m = 2$ calcula-se a *odds ratio* para a variável X_1 através da fórmula 2.13.

$$OR(x) = \frac{P[Y = 1|\boldsymbol{x}]}{P[Y = 0|\boldsymbol{x}]} \Big/ \frac{P[Y = 1|\boldsymbol{x}_0]}{P[Y = 0|\boldsymbol{x}_0]} \tag{2.13}$$

Os indivíduos com covariável $\boldsymbol{x}$ (grupo I) são comparados com indivíduos de referência com covariável $\boldsymbol{x}_0$ (grupo II) através da *odds ratio*.

Quando a variável resposta é dicotômica prova-se que a *odds ratio* é calculada através da exponenciação do parâmetro estimado. Uma *odds ratio* igual a um indica que a variável independente não contribui para a explicação da variável resposta, enquanto que uma *odds ratio* igual a dois, por exemplo, indica que a chance de ocorrer o evento é duas vezes maior quando a variável independente pertence ao grupo de referência.

Após o cálculo dos estimadores, é necessário verificar se os parâmetros do modelo ajustado são significantes. Esta verificação é feita a partir de testes estatísticos que serão descritos na seção 2.3.

2.2 Teste de significância para os coeficientes

A formulação e teste de uma hipótese estatística são necessários para determinar se as variáveis no modelo são significativas em relação à variável resposta.

Um teste muito utilizado para a significância do coeficiente de uma variável em qualquer modelo está relacionado com a seguinte questão: O modelo que inclui a variável em questão nos diz mais sobre a variável resposta do que o modelo que não possui a variável? A resposta para esta questão é dada através da comparação dos valores observados na variável resposta para

os dois modelos (com e sem a variável em questão). Se os valores preditos são melhores, ou mais precisos, quando a variável está presente então é um indício de que obtemos uma variável significante.

Em regressão linear usamos a tabela de análise de variância para determinar se a variável é significativa para o modelo. Esta tabela divide a variabilidade total em duas partes: uma referente ao modelo e outra referente aos erros aleatórios. Um grande valor da soma de quadrados do modelo, também chamada de soma de quadrados do tratamento, sugere que a variável independente é importante, enquanto um pequeno valor sugere que a variável independente não auxilia na predição da variável resposta.

Em regressão logística o princípio é análogo, ou seja, comparar valores observados da variável resposta com valores preditos a partir de modelos com e sem a variável em questão. Para isso baseia-se na comparação do log da função de verossimilhança. A comparação dos valores observados com os valores preditos é realizada através da razão de verossimilhança na expressão 2.14:

$$D = -2\,ln\left[\frac{verossimilhança\ do\ modelo\ ajustado}{verossimilhança\ do\ modelo\ saturado}\right] \tag{2.14}$$

Aplicando o log na razão de verossimilhança e multiplicando por menos dois temos uma quantidade com distribuição assintótica conhecida para testar hipóteses. Da equação 2.14 temos:

$$D = -2\sum_{i=1}^{n}\left[y_i ln\left(\frac{\widehat{\pi_i}}{y_i}\right) + (1-y_i) ln\left(\frac{1-\widehat{\pi_i}}{1-y_i}\right)\right] \tag{2.15}$$

Em regressão logística onde a variável resposta é igual a zero ou um, a verossimilhança para o modelo saturado é 1. A verossimilhança de um modelo saturado é dada pela equação 2.16.

$$l(modelo\ saturado) = \prod_{i=1}^{n} y_i{}^{y_i}.(1-y_i)^{(1-y_i)} = 1 \tag{2.16}$$

Sendo assim,

$$D = -2\ ln[verossimilhança\ do\ modelo\ ajustado] \tag{2.17}$$

Para testar a significância da variável independente deve-se comparar o valor de D em 2.17 para o modelo com e sem a variável independente na equação. A mudança em D devido à inclusão da variável independente no modelo é obtida por:

$$G = D(modelo\ sem\ a\ variável) - D(modelo\ com\ a\ variável) \tag{2.18}$$

Como a verossimilhança do modelo saturado é igual para ambos os valores de D temos:

$$G = -2\ ln\left[\frac{verossimlihança\ sem\ a\ variável}{verossimilhança\ com\ a\ variável}\right] \tag{2.19}$$

Sob a hipótese que β_1 é igual a zero, a estatística G dada em 2.19 segue uma distribuição assinótica Qui-quadrado com 1 grau de liberdade.

O teste Wald (Hosmer e Lemeshow, 2000) é estatisticamente equivalente para significância das variáveis do modelo. Este que compara o estimador de máxima verossimilhança para β através de uma aproximação do seu desvio padrão. Sob a hipótese que $\beta_1 = 0$, a razão resultante segue aproximadamente uma distribuição normal padrão.

$$W = \frac{\hat{\beta}_1}{\widehat{SE}(\hat{\beta}_1)} \tag{2.20}$$

O p-valor bilateral, correspondente ao teste de significância, é dado por $P[|z| > W]$, onde z denota uma variável aleatória que segue uma distribuição normal padrão.

Este teste pode ser realizado através de uma estatística Qui-quadrado da seguinte forma:

A hipótese linear para o vetor de parâmetros $\boldsymbol{\beta}$ é expressa na forma matricial como $H_0: \boldsymbol{L\beta} = \boldsymbol{c}$, onde $\boldsymbol{L}$ é uma matriz de coeficientes para a hipótese linear e $\boldsymbol{c}$ é um vetor constante. A estatística Qui-quadrado Wald para testar a hipótese nula é calculado da forma

$$\chi^2_W = \left(\boldsymbol{L\hat{\beta}} - \boldsymbol{c}\right)' \left[\boldsymbol{L}\,\widehat{\boldsymbol{V}}(\boldsymbol{\hat{\beta}})\boldsymbol{L}'\right]^{-1} \left(\boldsymbol{L\hat{\beta}} - \boldsymbol{c}\right), \tag{2.21}$$

onde $\widehat{\boldsymbol{V}}(\boldsymbol{\hat{\beta}})$ é a matriz de variâncias e covariâncias estimada.

Sob a hipótese nula χ^2_W em 2.21 tem uma distribuição assintótica Qui-quadrado com r graus de liberdade, sendo r o rank da matriz $\boldsymbol{L}$.

No capítulo 5 para exemplificar a eficácia do teste Wald utilizamos o pacote estatístico SAS na *procedure logistic* como pode ser observado no capítulo 5 de aplicações.

Resumidamente, o método para o teste da significância do coeficiente de uma variável na regressão logística é similar à aproximação usada na regressão linear.

Os testes de significância estão diretamente ligados com os intervalos de confiança para os parâmetros estimados, como pode ser visto na seção seguinte.

2.3 Intervalo de confiança para os coeficientes

Outra análise importante para o teste de significância dos coeficientes das variáveis do modelo é o cálculo e a interpretação dos intervalos de confiança para os parâmetros de interesse.

A base para a construção dos estimadores intervalares é a mesma teoria estatística usada na formulação dos testes de significância do modelo. Os intervalos de confiança para os parâmetros são baseados na estatística de Wald. Os limites do intervalo de confiança $100(1-\alpha)\%$ para os coeficientes de inclinação são dados por

$$\hat{\beta}_i \pm z_{1-\alpha/2}\, \widehat{DP}(\hat{\beta}_i) \tag{2.22}$$

e para o intercepto temos

$$\hat{\beta}_0 \pm z_{1-\alpha/2}\, \widehat{DP}(\hat{\beta}_0) \tag{2.23}$$

onde $z_{1-\alpha/2}$ é o ponto $100(1-\alpha)\%$ da distribuição normal padrão e $\widehat{DP}(.)$ é o desvio padrão estimado para a estimativa do parâmetro.

As estimativas dos desvios padrão são obtidas através da matriz de segundas derivadas parciais do log da função de verossimilhança, a matriz Hessiana. Estas derivadas parciais têm sua forma geral da seguinte forma:

$$\frac{\partial^2 L(\beta)}{\partial \beta_i^2} = -\sum_{k=1}^{n} x_{ki}^2 \pi_k (1-\pi_k) \tag{2.24}$$

e

$$\frac{\partial^2 L(\beta)}{\partial \beta_i\, \partial \beta_j} = -\sum_{k=1}^{n} x_{ki} x_{kj} \pi_k (1-\pi_k) \tag{2.25}$$

para $i, j = 0,1,2,\ldots,p$. A matriz de dimensão $(p+1)x(p+1)$ contendo o negativo dos termos dados nas equações acima é chamada de $I(\beta)$. A matriz de variância e covariância dos coeficientes estimados é obtida do inverso da

matriz Hessiana e na sua diagonal têm-se as variâncias estimadas e fora da diagonal as covariâncias.

$$\Sigma(\beta) = I^{-1}(\beta) \quad \text{e} \quad \widehat{DP}(\hat{\beta}_j) = \left[\hat{Var}(\hat{\beta}_j)\right]^{1/2} \tag{2.26}$$

A formulação da matriz de informação é dada por:

$$I(\hat{\beta}) = X'VX \tag{2.27}$$

onde,

$$V = \begin{bmatrix} \hat{\pi}_1(1-\hat{\pi}_1) & 0 & \dots & 0 \\ 0 & \hat{\pi}_2(1-\hat{\pi}_2) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \hat{\pi}_n(1-\hat{\pi}_n) \end{bmatrix} \tag{2.28}$$

Se o intervalo de confiança construído para o parâmetro contiver o valor zero, significa que com $100(1-\alpha)\%$ de confiança pode-se afirmar que o zero está contido no intervalo indicando desta forma uma não significância da variável independente.

Capítulo 3

Regressão Logística Bayesiana

Os métodos clássicos tradicionalmente utilizados em modelagem de dados de regressão binomial confiam em inferências assintóticas. Métodos Bayesianos podem ser aplicados através de aproximações ou por meios computacionais simples e se aplicam para qualquer tamanho amostral. O recente desenvolvimento do método de MCMC Cadeias de Markov com Monte Carlo tem ajudado a resolver a maior parte das dificuldades associadas com análise Bayesiana de modelos não lineares, é um método simples e de fácil implementação (Bedrick *et al*, 1997).

Uma das vantagens da metodologia Bayesiana é a capacidade de adicionar ao modelo o conhecimento de um especialista, aproveitando assim um conhecimento externo à base de dados.

Considere os dados de regressão $(Y_i, \boldsymbol{x_i})$, $i = 1, 2, \ldots, n$ onde Y é a variável dicotômica de interesse e $\boldsymbol{x}$ é o vetor conhecido de dimensão k representante das variáveis explanatórias. A probabilidade de um indivíduo ser mau pagador dado o vetor $\boldsymbol{x}_i$ observado é dada por $\boldsymbol{\pi}(\mathbf{x_i'}\boldsymbol{\beta})$, ou seja, $\boldsymbol{\pi}(\mathbf{x_i'}\boldsymbol{\beta}) \equiv p[y = 1|\boldsymbol{x}_i, \boldsymbol{\beta}] =$

p_i, onde $\boldsymbol{\beta}$ é um vetor de coeficientes de regressão desconhecido de tamanho $k+1$. A função $r(.)$ é dada pelo modelo logístico, ou seja:

$$\boldsymbol{\pi}(\mathbf{x_i'\beta}) = \frac{e^{\mathbf{x_i'\beta}}}{1 + e^{\mathbf{x_i'\beta}}} \tag{3.1}$$

Assim, a função de ligação $r^{-1}(p_i)$ é dada por:

$$r^{-1}(p_i) = \ln\left[\frac{\pi(\mathbf{x}_i)}{1-\pi(\mathbf{x}_i)}\right] = g(\mathbf{x}_i) = \mathbf{x_i'\beta} = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k \tag{3.2}$$

A verossimilhança para os dados $Y = (y_1, \ldots, y_n)$ é

$$l(\beta|Y) = \prod_{i=1}^{n} f(y_i|\boldsymbol{x_i}) = \prod_{i=1}^{n} [\pi(\boldsymbol{x}_i)]^{y_i}[1-\pi(\boldsymbol{x}_i)]^{1-y_i} \tag{3.3}$$

A inclusão da informação do especialista no modelo é feita através de uma *priori* e da utilização do teorema de Bayes. Dada uma distribuição *a priori* para β, digamos $p(\beta)$, a obtenção de uma distribuição a posteriori é dada na equação 3.4 (Ehlers, 2005).

$$p(\beta|Y) = \frac{L(\beta|Y)p(\beta)}{\int L(\beta|Y)p(\beta)\,d\beta} \tag{3.4}$$

Integrais envolvendo $p(\beta|Y)$ são, na maioria, intratáveis, o que torna necessário o uso de aproximações. Para o ajuste do modelo de regressão logística Bayesiano utilizou-se o método de Metropolis-Hastings, um dos métodos de Monte Carlo via Cadeias de Markov (MCMC) uma alternativa aos métodos não iterativos em problemas complexos, e aproximações de Laplace que, através da expansão de Taylor aproximam a função *a posteriori* para uma

função de densidade normal. O objetivo é obter uma amostra da distribuição *a posteriori* e calcular estatísticas descritivas desta distribuição.

3.1 Cálculo e interpretação dos coeficientes

Na regressão logística clássica os coeficientes são estimados através da maximização da verossimilhança que é realizada através de métodos iterativos como Newton-Raphson ou Escore de Fisher. Na regressão logística Bayesiana eles são encontrados através da distribuição *a posteriori* do vetor de coeficientes β.

A geração das amostras *a posteriori* pode ser realizada através do método proposto por Metropolis-Hastings em 1953, que consiste na simulação de um passeio aleatório no espaço de β que converge para a distribuição de interesse, $\pi(\beta|Y)$, descrito na seção 3.2.1. Existem diversas formas de obter uma amostra da distribuição *a posteriori*, mas neste trabalho apenas esta maneira será estudada.

A distribuição *a posteriori* contém toda a informação probabilística a respeito dos parâmetros de interesse e os gráficos das suas funções de densidade *a posteriori* são a melhor descrição do processo de inferência (Paulino *et al*, 2003).

A interpretação dos coeficientes estimados em uma análise Bayesiana é a mesma realizada em uma análise clássica, através da *odds ratio* e da taxa de mudança representada pelo coeficiente.

3.1.1 Algoritmo de Metropolis-Hastings

O algoritmo de Metropolis-Hastings foi assim nomeado em referência à Nicholas Metropolis que o criou em 1953 para o caso da distribuição de Boltzmann. Em 1970, W.K. Hastings generalizou o algoritmo de Metropolis.

Tais métodos, denominados MCMC tiveram grande impacto no desenvolvimento e aplicabilidade da metodologia Bayesiana. Esta seção está baseada em Ehlers (2004).

O algoritmo usa o conceito dos métodos de rejeição, isto é, um valor de uma distribuição auxiliar é gerado e aceito com uma dada probabilidade. Este mecanismo de correção garante a convergência da cadeia para a distribuição de equilíbrio, neste caso a distribuição *a posteriori*.

O algoritmo inicia com a geração de um valor inicial θ através da função $q(.|\theta)$, esta geração depende do estado atual da cadeia. Em seguida o valor gerado é aceito com probabilidade $\alpha(\theta, \theta^*)$ que é calculada a partir da equação 3.5.

$$\alpha(\theta, \theta^*) = min\left\{1, \frac{h(\theta^*)q(\theta|\theta^*)}{h(\theta)q(\theta^*|\theta)}\right\}, \tag{3.5}$$

onde $h(.)$ é a distribuição de interesse.

Por fim, os valores aceitos ao longo das iterações formam a amostra com a distribuição de interesse.

Uma característica importante é que a distribuição de interesse pode ser conhecida parcialmente, uma vez que na probabilidade $\alpha(\theta, \theta^*)$ a razão $h(\theta^*)/h(\theta)$ não se altera com a multiplicação da parte constante da distribuição $h(.)$. Isto se torna fundamental em aplicações Bayesianas onde não se conhece completamente as densidades *a posteriori*.

Em termos práticos o algoritmo de Metropolis-Hastings pode ser especificado pelos seguintes passos:

1. Inicialize o contador de iterações $t = 0$ e especifique um valor inicial $\theta^{(0)}$;
2. Gere um novo valor θ^* a partir da distribuição $q(.|\theta)$;
3. Calcule a probabilidade de aceitação $\alpha(\theta, \theta^*)$ e gere $u \sim U(0,1)$;
4. Se $u \leq \alpha$ então aceite o novo valor e faça $\theta^{(t+1)} = \theta^*$, caso contrário rejeite e considere $\theta^{(t+1)} = \theta^{(t)}$;

5. Incremente o contador de t para $t+1$ e volte ao passo 2.

Embora a distribuição proposta possa ser escolhida arbitrariamente na prática deve-se tomar alguns cuidados para garantir a eficiência do algoritmo, em geral $q(.)$ deve ser uma boa aproximação para $h(.)$. Observe que sendo a distribuição *a priori* informativa ou não informativa o método funciona de forma análoga.

Em aplicações Bayesianas a distribuição de interesse é a própria *posteriori*, neste caso a probabilidade de aceitação é dada da seguinte forma:

$$\alpha(\beta,\beta^*) = min\left\{1, \frac{L(\beta^*|Y)p(\beta^*)q(\beta|\beta^*)}{L(\beta|Y)p(\beta)q(\beta^*|\beta)}\right\}. \tag{3.6}$$

Após a implementação do algoritmo é necessário verificar se a cadeia gerada converge para a distribuição de interesse, tal verificação pode ser feita através de diversos diagnósticos de convergência: gráficos e numéricos.

Neste trabalho trabalhou-se com dois gráficos para a verificação da convergência da cadeia de Markov: o primeiro com o valor da variável aleatória $\beta^{(i)}$ *vs* i, onde i representa a i-ésima iteração para mais de uma cadeia. Devem-se visualizar as distribuições dos valores gerados em torno de um valor fixo e o segundo gráfico são as auto-correlações parciais *vs* lags, também para mais de uma cadeia, onde deve-se observar um decaimento rápido na auto-correlação dos valores gerados pela cadeia.

Outro possível diagnóstico de convergência foi proposto por Gelman e Rubin em 1992. Neste caso monitora-se a convergência de duas ou mais cadeias paralelas, cada uma com valores iniciais diferentes e distantes entre si, utilizadas na obtenção da distribuição *a posteriori*. A convergência é diagnosticada através da comparação das distribuições das duas cadeias após certo número de iterações, se as distribuições são indistinguíveis é determinada a convergência. O método de comparação é baseado na

comparação “entre cadeias” e “dentro das cadeias”, como em uma análise de variância.

O diagnóstico de convergência é realizado para uma variável de cada vez, onde valores próximos de um, indicam a convergência das cadeias, para o limite superior do intervalo de confiança do fator calculado.

Caso a convergência da cadeia não seja alcançada, algumas características envolvidas na geração da cadeia podem ser alteradas como o tamanho do *burn-in* e dos saltos. Um *burn-in* de tamanho *k* significa que os primeiros *k* valores gerados da amostra a partir do valor inicial são eliminados da cadeia, eliminando assim os valores gerados antes da convergência do algoritmo e ajudando na convergência. Um salto de valor igual a mil funciona da seguinte forma, depois do período de *burn-in* só será selecionado para a amostra os valores gerados a cada mil iterações, por exemplo, o milésimo valor gerado após o *burn-in* será o primeiro valor da amostra seguido pelo 2.000º valor depois o 3.000º valor e assim sucessivamente. Se o tamanho desejado da amostra da distribuição *a posteriori* for de 5.000 valores com um *burn-in* de tamanho 100 e saltos de tamanho 10 serão necessárias 50.100 iterações para a geração da amostra desejada. Veja Figura 3.1.

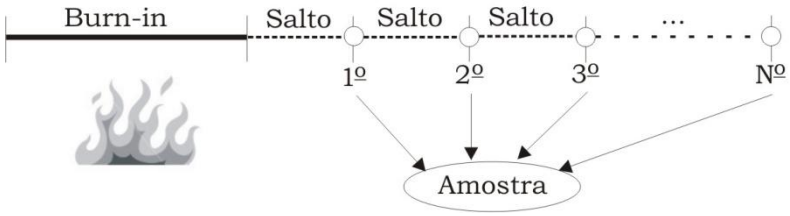


Figura 3.1 – *Burn-in*

O objetivo principal da geração de uma amostra a partir da função densidade de probabilidade *a posteriori* é obter uma amostra da distribuição de cada um dos parâmetros de interesse. Os estimadores dos coeficientes são

obtidos através de medidas descritivas desta distribuição, como por exemplo, média amostral e mediana.

3.2 Teste de significância para os coeficientes:

O objetivo de testar a significância de um determinado coeficiente em um modelo de regressão logística Bayesiano é análogo ao do método clássico, isto é, responder à seguinte questão: O modelo que inclui a variável em questão nos diz mais sobre a variável resposta do que o modelo que não possui a variável?

No modelo de regressão logística clássico a questão é respondida através da utilização do teste da razão de verossimilhança ou do teste Wald, para o modelo Bayesiano através do Fator de Bayes (Paulino e Murteira, 2003) e do Critério de informação Bayesiano (BIC – *Bayesian Information Criterion*) (Schwarz, 1978).

3.2.1 Fator de Bayes e BIC

Dado um problema de seleção de modelos no qual se deve escolher entre dois modelos M_1 e M_2 com base no vetor de dados x. O fator de Bayes FB_{12} é dado por:

$$FB_{12} = \frac{p(\boldsymbol{x}|M_1)}{p(\boldsymbol{x}|M_2)} \tag{3.7}$$

onde $p(\boldsymbol{x}|M_i)$ é chamado de função de verossimilhança marginal para o i-ésimo modelo.

Geralmente os modelos M_1 e M_2 são parametrizados pelos vetores de parâmetros θ_1 e θ_2, respectivamente. Então FB_{12} é dado por:

$$FB_{12} = \frac{p(\boldsymbol{x}|M_1)}{p(\boldsymbol{x}|M_2)} = \frac{\int L(\theta_1|M_1)\pi(\theta_1)d\theta_1}{\int L(\theta_2|M_2)\pi(\theta_2)d\theta_2} \tag{3.8}$$

Pode-se considerar para $p(\boldsymbol{x}|M_i)$ como a esperança da verossimilhança sob a distribuição *a priori* dos parâmetros.

O Fator de Bayes, FB_{12}, é interpretado como a vantagem do modelo M_1 contra M_2, de acordo com os dados observados. O modelo que obtiver o maior valor do fator de Bayes é selecionado, sendo que os modelos são comparados dois a dois.

Através do $\log(FB_{12})$ pode-se obter uma interpretação do fator de Bayes através da Tabela 3.1, onde a hipótese nula é o modelo 1 ser melhor que o modelo 2:

Tabela 3.1 – Interpretação do Fator de Bayes

$2.\log(FB_{ij})$	FB_{ij}	Evidência contra H_0
0 a 2	1 a 3	Inconclusiva
2 a 6	3 a 20	Significante
6 a 10	20 a 150	Forte
> 10	> 150	Decisiva

fonte: Campos (2007)

Uma saída para a obtenção de $p(\boldsymbol{x}|M_i)$ é a utilização de outro método de seleção de modelos, Critério de Informação Bayesiana, *Bayesian Information Criterion* – BIC, que fornece uma medida de evidência em favor de um modelo sobre outro sem a necessidade de utilização da função de densidade *a priori*.

Schwarz (1978) mostra que, para amostras grandes de tamanho n, uma aproximação para $-2.\log BF_{ij}$ é dada por:

$$\Delta BIC = -2\ln\left[\frac{\sup_{M_1} L(\theta_1|M_1)}{\sup_{M_2} L(\theta_2|M_2)}\right] - (p_2 - p_1)\ln(n) \qquad (3.9)$$

onde p_i é o número de parâmetros no modelo M_i, i = 1, 2.

O primeiro termo corresponde exatamente ao teste de razão de verossimilhança enquanto o segundo funciona como uma penalidade que corrige as diferenças de quantidade de parâmetros entre os modelos.

3.3 Intervalo de credibilidade para os coeficientes

Uma estimativa pontual sozinha não possui informação de precisão da estimativa. As medidas de incerteza mais utilizadas são a variância ou o coeficiente de variação para a média *a posteriori* e a distância entre quartis para a mediana *a posteriori*. Em inferência clássica os estimadores intervalares para os parâmetros são chamados de intervalo de confiança, que possuem $(1-\alpha)\%$ de chances de conter o verdadeiro valor do parâmetro, no caso da inferência Bayesiana são chamados de intervalo de credibilidade e possuem uma interpretação direta sobre o parâmetro, ou seja, o parâmetro varia entre o limite inferior e o superior com $(1-\alpha)\%$ de chances.

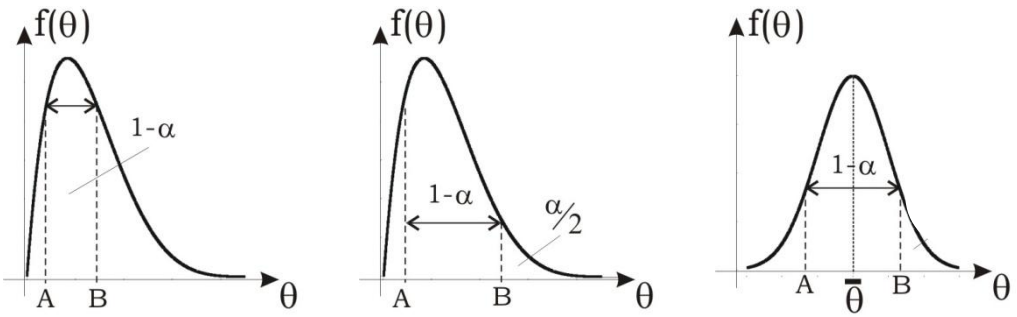
Buscamos intervalos de credibilidade com o menor comprimento possível, indicando assim uma distribuição concentrada em torno do parâmetro estimado.

Da mesma forma como na regressão logística clássica, os intervalos de credibilidade dão indicações sobre a significância dos parâmetros, ou seja, quando o valor zero está dentro do intervalo de credibilidade é um indício para a não significância da respectiva variável no modelo.

A seguir algumas formas de construir um intervalo de credibilidade:

i) Escolher o menor intervalo possível, no caso de uma distribuição unimodal, de forma que maximize a densidade probabilístic, *HPD*;
ii) Escolher o intervalo baseado nas medidas quantílicas, ou seja, a probabilidade de estar abaixo do limite inferior é a mesma de estar acima do limite superior, $\alpha/2$;
iii) Escolher o intervalo a partir da média como ponto central.

A Figura 3.2 fornece uma descrição gráfica de cada um dos itens acima:



i ii iii

Figura 3.2 – Intervalos de credibilidade

Na Figura 3.2 A e B representam, respectivamente, o limite inferior e o limite superior do intervalo de credibilidade, $[A; B]$. No primeiro caso (i), $P[\theta < A] + P[\theta > B] = \alpha$, ou seja, as probabilidades abaixo e acima dos limites do intervalo não precisam ser iguais. No segundo caso (ii), $P[\theta < A] = P[\theta > B] = \alpha/2$, ou seja, as probabilidades são iguais. No terceiro caso (iii), $A = \theta - x$ e $B = \theta + x$, onde $x = constante$. Para o terceiro caso a distribuição *a posteriori* do parâmetro deve ser preferencialmente simétrica para um bom intervalo de credibilidade.

Capítulo 4

Redes Neurais Artificiais

As Redes Neurais Artificiais (RNA) são modelos computacionais inspirados na estrutura e no funcionamento do cérebro biológico. Com base no conhecimento biológico as pesquisas em RNAs buscam desenvolver modelos computacionais capazes de aprender, generalizar, agrupar e organizar dados. Segundo Kandel (1991), a principal semelhança entre Redes Neurais Artificiais e os circuitos formados no cérebro está no extenso processamento paralelo apresentado por ambos e no funcionamento do circuito que não depende de alguns elementos isoladamente, mas sim uma função do conjunto dos elementos.

O cérebro é composto por algo em torno de 10 bilhões de neurônios que se interconectam formando uma verdadeira rede, que é capaz de processar milhões de informações e realizar algo em torno de 60 trilhões de ligações sinápticas, (Braga et al, 2000). As RNAs podem ser vistas como estruturas formadas por um número grande, porém muito menor, de unidades de processamento bastante simplificadas conectadas entre si, os neurônios matemáticos. As Redes Neurais representam uma família de modelos, ao invés de uma simples técnica, cada forma de rede é otimizada para um específico

conjunto de condições, em analogia com a especificidade funcional de diferentes regiões do cérebro. Os neurônios matemáticos estão dispostos em camadas (uma camada de entrada, uma ou várias intermediárias e uma de saída) e são responsáveis pela não linearidade da rede, através do processamento interno de certas funções matemáticas. Um modelo de redes neurais possui como parâmetros os pesos sinápticos, estes pesos são estimados através de algoritmos de aprendizado supervisionados, ou seja, da mesma forma como na metodologia clássica ou Bayesiana o ajuste conta com o conhecimento da variável resposta durante o processo de ajuste do modelo. Modelos de redes neurais com aprendizado não-supervisionado, por exemplo, redes SOM, são utilizadas sem a observação da variável resposta, ou seja, através da análise dos exemplos de treinamento o modelo determina se alguns deles podem ser agrupados de algum modo formando agrupamentos ou clusters.

O objetivo deste capítulo é fornecer uma introdução sobre modelos de Redes Neurais com aprendizado supervisionado, apresentando uma descrição dos diversos componentes das redes, o modelo mais simples de Rede Neural, *perceptron* e uma extensão do *perceptron*, as redes *perceptron* multicamadas, muito aplicada em vários setores.

A seção 4.2 apresenta os fundamentos de redes neurais, iniciando com a seção 4.2.1 onde a primeira versão formal de neurônio artificial, proposta por McCulloch e Pitts (1943) é apresentada, em seguida na seção 4.2.2 apresentam-se as funções de ativação, seguida por uma introdução ao aprendizado na seção 4.2.3. A seção 4.3 apresenta as redes *perceptron* e a seção 4.3.1 apresenta o método delta de aprendizado utilizado neste tipo de rede, a seção 4.3.2 apresenta o problema da capacidade de separação linear através da porta lógica ou-exclusivo, cuja solução é apresentada na seção 4.4 com as redes MLP. Em seguida a seção 4.4.1 contém o algoritmo de aprendizado *back-propagation* e a seção 4.4.2 a utilização da constante

momentum e da taxa de aprendizagem, incluindo sugestões de escolha destes parâmetros. Este capítulo é finalizado com a seção 4.5 que apresenta o problema de *overfitting* e a capacidade de generalização dos modelos de Redes Neurais.

4.1 Fundamentos de Redes Neurais

Nesta seção serão apresentados os fundamentos de Redes Neurais iniciando pelo primeiro neurônio matemático em seguida uma descrição das funções de ativação e definição dos diferentes tipos de aprendizagem.

4.1.1. Neurônio artificial de McCulloch e Pitts

O neurônio biológico representa a unidade elementar de um sistema nervoso biológico. Neurônios biológicos aparecem organizados em estruturas e depois de um período de aprendizado adequado eles trabalham juntos para resolver uma grande quantidade de problemas complexos (Berthold *et al*, 2002).

No neurônio biológico, as fibras neurais de entrada, chamadas dendritos, recebem sinais elétricos dos neurônios conectados via processos bioquímicos através das sinapses. Dependendo da natureza da sinapse química cada junção pode aumentar ou reduzir o sinal transmitido. Se a soma dos sinais elétricos recebidos pelo neurônio atingir um determinado limiar uma ação potencial é disparada pela célula através da fibra de saída, chamada de axônio.

A primeira definição formal de neurônio artificial, ou neurônio matemático, foi proposta por McCulloch e Pitts em 1943. Chamado de neurônio MCP é constituído por diversas entradas (dendritos), ponderadas por pesos (comportamento das sinapses) e uma saída (axônio) que representa

quando o neurônio está ativo ou não, ou seja, a saída é binária: ou está disparando potenciais de ação ou está em repouso.

Os sinais de entrada $x_1(t), x_2(t), \ldots, x_k(t)$ no tempo t provenientes do disparo de outros neurônios são supostamente binários e são transmitidos através das sinapses para o corpo da célula. A entrada $a(t)$ para o neurônio é então equivalente à soma dos sinais de entrada recebidos multiplicadas pelos pesos sinápticos $w_1 x_1(t)$, $w_2\, x_2(t)$,..., $w_k x_k(t)$. Finalmente uma função de ativação, $f(a)$, ajusta a saída, $s(t+1)$ no instante $t+1$. Existem inúmeras funções de ativação possíveis, a função de ativação pode ser qualquer função derivável, ver seção 4.1.2. A função 4.1 é conhecida por função degrau e possui a saída ajustada para um (neurônio ativo) ou zero (neurônio inativo) se o limiar de excitação, w_0, foi ou não ativado.

$$f(a) = \begin{cases} 1 \; se \; a \geq w_0 \\ 0 \; se \; a < w_0 \end{cases} \tag{4.1}$$

A equação final modelada pelo neurônio artificial de McCulloch e Pitts (1943) é dada por:

$$s(t+1) = f\left(\sum_{i=1}^{k} w_i x_i(t)\right) \tag{4.2}$$

Este modelo é muito mais simples quando comparado com um neurônio biológico. Neurônios biológicos não possuem saída binária e em tempos regulares t, possuem saída em intervalo contínuo e em um tempo contínuo, variável e não são necessariamente disparados de forma sincronizada.

Podem-se levantar algumas limitações na descrição do modelo MCP original:

1. Redes MCP com apenas uma camada só conseguem implementar funções linearmente separáveis;

2. Pesos negativos são mais adequados para representar disparos inibidores;
3. O modelo foi proposto com pesos fixos, não ajustáveis.

O neurônio MCP é computacionalmente um mecanismo poderoso para a solução de problemas linearmente separáveis, se os pesos sinápticos e o limiar são devidamente selecionados podem-se aplicar os operadores booleanos básicos de NOT, OR e AND, permitindo a obtenção de uma estrutura de neurônios artificiais computacionalmente poderosa na solução destes problemas. Outra característica importante do neurônio MCP consiste no relacionamento não linear entre entrada e saída, assim como nos neurônios biológicos. Por todas estas razões o neurônio artificial descrito, com melhorias e variações, é adotado como unidade elementar na construção de Redes Neurais Artificiais.

4.1.2. Funções de ativação

A partir do modelo proposto por McCulloch e Pitts (1943), foram derivados vários outros modelos que permitem a produção de uma saída qualquer, não necessariamente zero ou um, e com diferentes funções de ativação. A seguir quatro funções de ativação diferentes: a função logística, a função arctan, a função Elliot e a função tangente hiperbólica que pode ser escrita como função da logística. As funções podem ser observadas na Figura 4.1. São elas:

$$logistic(x) = \frac{1}{1 + e^{-x}} \tag{4.3}$$

$$arctan(x) = \frac{2}{\pi} tan^{-1}(x) \tag{4.4}$$

$$elliot(x) = \frac{x}{1 + |x|} \tag{4.5}$$

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{1 - e^{-2x}}{1 + e^{-2x}} = \frac{2}{1 + e^{-2x}} - 1 = \\ = 2\, logistic(2x) - 1 \tag{4.6}$$

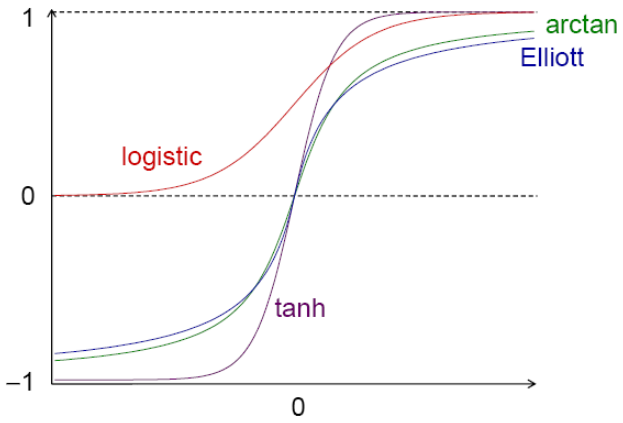


Figura 4.1 – Funções de ativação

A escolha da função de ativação está diretamente ligada com o intervalo da variável resposta a ser explicada pelo modelo, para problemas de *Credit Score* a função de ativação do neurônio na camada de saída utilizada é a logística, por ter intervalo entre zero e um. Para respostas no intervalo $(-\infty, +\infty)$ utiliza-se a função de ativação identidade.

4.1.3. Aprendizado

A aprendizagem é um processo pelo qual os parâmetros livres de uma Rede Neural são adaptados através de um processo de estimulação pelo ambiente

no qual a rede está inserida. O tipo de aprendizagem é determinado pela maneira que a modificação dos parâmetros ocorre (Haykin, 1999)

Os pesos sinápticos devem ser cuidadosamente escolhidos para realizarem a tarefa de interesse. Esta escolha é realizada durante o período de treinamento da Rede Neural no qual ocorre o aprendizado. O processo de aprendizado permite que os pesos sinápticos se adaptem mudando conseqüentemente as funções de entrada e saída, aprendendo assim a implementar automaticamente a tarefa desejada. Suponha que um grande conjunto de dados esteja disponível. As duas estratégias principais mais adotadas são:

- Aprendizado Supervisionado: utilizado quando a saída de interesse, respostas desejadas, da rede é conhecida para todas as linhas do banco de dados de treinamento. É o aprendizado mais comum no treinamento de Redes Neurais, nele a resposta da rede é diretamente comparada com o valor conhecido e desejado da variável resposta para cada linha do banco de treinamento e através da comparação entre o desejado e a saída da rede os pesos sinápticos são corrigidos para a minimização dos erros.
- Aprendizado não-Supervisionado: utilizado quando a saída de interesse, respostas desejadas, é desconhecida. Neste caso a estratégia de aprendizado é ensinar a rede a descobrir por si mesma correlações e similaridades para cada um dos exemplos de treinamento e baseado nisso agrupá-los em diferentes clusters. Neste caso o padrão de treinamento é composto apenas pelas entradas e não possui a saída desejada, portanto a rede não tem uma resposta para utilizar como meta nem informação para re-ponderar os pesos sinápticos.

Ainda pode-se citar o aprendizado por competição como um caso particular do aprendizado não supervisionado e o aprendizado por reforço como um caso

particular do aprendizado supervisionado, para maiores detalhes veja Braga *et al* (2000). Neste trabalho apresentaram-se apenas treinamentos com aprendizado supervisionado, pois é o ajuste indicado na modelagem de *Credit Score*, onde se utiliza informações já conhecidas sobre a adimplência ou inadimplência dos clientes.

4.2. *Perceptron*

O trabalho original de McCulloch e Pitts (1943) enfocou a modelagem de um neurônio biológico e sua capacidade computacional com a apresentação de vários exemplos de topologias de rede com capacidade de execução de funções booleanas. O conceito de aprendizado em RNA só foi introduzido por Frank Rosenblatt em 1958 (Rosenblatt, 1958). O modelo proposto por Rosenblatt, conhecido como *perceptron*, era composto por uma estrutura de rede, tendo como unidades básicas os neurônios matemáticos MCP, e por uma regra de aprendizado. Somente alguns anos depois, em 1962, Rosenblatt (Rosenblatt, 1962) demonstrou o teorema da convergência do *perceptron*, que mostra que um neurônio matemático MCP treinado com o algoritmo de aprendizado do *perceptron* sempre converge caso o problema em questão seja linearmente separável.

A topologia do *perceptron* é composta de uma camada de entrada e uma camada de saída. Embora esta topologia possua dois níveis, ela é conhecida como *perceptron* de uma única camada, já que somente o nível de saída (unidades de resposta) possui propriedades adaptativas e a camada de entrada consiste basicamente em unidades de leitura dos valores de entrada da rede, no caso as variáveis explicativas.

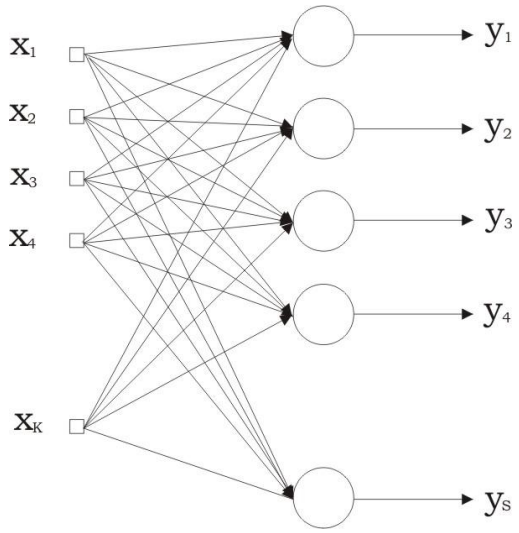


Figura 4.2 – *Perceptron*

A *i*-ésima saída, y_i, é dada da seguinte forma:

$$y_i = f(a_i) = f\left(\sum_{k=0}^{n} w_{ik}x_k\right). \tag{4.7}$$

O valor do limiar, w_0, está inserido na equação.

4.2.1. Regra Delta

Esta seção apresenta a regra de aprendizado do *perceptron*, que permite a adaptação dos pesos sinápticos de forma que a rede execute uma a tarefa de classificação.

Seja T um conjunto contendo n amostras de treinamento associados com um vetor $\boldsymbol{x_i}$ de entrada e uma resposta observada de saída $\boldsymbol{y_i}$. O procedimento de aprendizado deve ajustar os pesos sinápticos da rede

perceptron de forma que a saída da rede s_i para o vetor $\boldsymbol{x_i}$ seja o mais parecido possível com o valor desejado observado y_i. O conjunto de pesos w_{ik} que produz aproximadamente s_i próximo de y_i para qualquer vetor de entrada $\boldsymbol{x_i}$, $i = 1, 2, \ldots, n$ representa a matriz ótima de pesos $\boldsymbol{W}$, chamada de $\boldsymbol{W}^*$. Partindo de valores iniciais aleatórios, a regra de aprendizado precisa atingir o valor $\boldsymbol{W}^*$, se ele existir, em um número finito de passos. Nesta seção a regra de aprendizado é dada através da diferenciação das funções de ativação dos neurônios, $f(a)$.

O primeiro passo necessário é determinar uma medida de erro como função da matriz de pesos $\boldsymbol{W}$ da rede. Esta função deve ser diferenciável com relação aos elementos w_{ik} da matriz de pesos $\boldsymbol{W}$, então uma variação do algoritmo de otimização gradiente pode ser aplicada para encontrar a matriz de pesos ótima no mínimo da função de erro. O melhor método de otimização numérica conhecido é o algoritmo gradiente descendente e a função de erro, φ, mais comumente utilizada é a soma de quadrados dos resíduos, definida como:

$$\varphi(\boldsymbol{W}) = \frac{1}{2}\sum_{q=1}^{m}\sum_{i=1}^{p}\left(s_i^q - y_i^q\right)^2 = \sum_{q=1}^{m}\varphi^q(\boldsymbol{W}) \tag{4.8}$$

$$\varphi^q(\boldsymbol{W}) = \frac{1}{2}\sum_{i=1}^{p}\left(s_i^q - y_i^q\right)^2 \tag{4.9}$$

onde m é o número de exemplos de treinamento e p é o número de unidades de saída da rede.

A idéia do método do gradiente descendente é iniciar a matriz de pesos $\boldsymbol{W}$ com valores aleatórios e mover os pesos na direção oposta do gradiente, ou seja, mudar cada w_{ik} proporcionalmente ao negativo do gradiente de $\varphi(\boldsymbol{W})$ no local presente $\boldsymbol{W}$.

$$w_{ik}(u+1) = w_{ik}(u) + \Delta w_{ik}(u) \tag{4.10}$$

Onde Δw_{ik} corresponde ao tamanho do passo que se dá na direção oposta do gradiente. $\Delta w_{ik}(u) = -\eta \frac{\partial \varphi(u)}{\partial w_{ik}} = -\eta \sum_{i=1}^{n} \frac{\partial \varphi^q(u)}{\partial w_{ik}} = \sum_{i=1}^{n} \Delta w_{ik}^q(u)$ (4.11)

$$\Delta w_{ik}^q(u) = -\eta \frac{\partial \varphi^q(u)}{\partial w_{ik}} \tag{4.12}$$

η é um número positivo pequeno, chamado de taxa de aprendizagem, e u indica o ciclo corrente do procedimento sobre o conjunto de treinamento, pode ser chamado também de época.

Uma possível estratégia para o ajuste dos pesos da rede é escolher os pesos de acordo com a equação 4.11 depois de todos os m indivíduos do banco de dados de treinamento ser percorrido. Outra possível estratégia é ajustar o peso da rede depois de cada exemplo de treinamento da rede ser apresentado a rede, neste caso, a equação 4.9 é utilizada e a atualização dos pesos é realizada de acordo com a equação 4.12. Este procedimento de aprendizado percorre todos os exemplos de treinamento até que um valor de erro satisfatório seja alcançado.

A função de ativação do neurônio f(a) é diferenciável, a derivada parcial para os exemplos de treinamento é dada na equação 4.13.

$$\frac{\partial \varphi^q}{\partial w_{ik}} = \frac{\partial \frac{1}{2} \sum_{i=1}^{p} \left(s_i^q - y_i^q\right)^2}{\partial w_{ik}} = \left(s_i^q - y_i^q\right) \frac{f\left(\sum_{k=0}^{n} w_{ik} x_k^q\right)}{\partial w_{ik}} =$$

$$= \left(s_i^q - y_i^q\right) f'\left(a_i^q\right) x_k^q = \delta_i^q x_k^q \tag{4.13}$$

com $\delta_i^q = \left(s_i^q - y_i^q\right) f'\left(a_i^q\right)$ e $a_i^q = \sum_{k=0}^{n} w_{ik} x_k^q$

então, $$\Delta w_{ik}^q = -\eta \delta_i^q x_k^q. \tag{4.14}$$

A equação 4.14 define uma regra de atualização dos pesos sinápticos da rede *perceptron*, que minimiza a função de erro $\varphi(\boldsymbol{W})$ para a amostra de treinamento. Se a saída da rede estiver correta não ocorre variação no peso, se a saída é menor que o valor da variável resposta cada peso é incrementado de η e decrementado de η se a saída é maior que o valor da variável resposta.

A quantidade δ_i^q, usada para o cálculo do Δw_{ik}^q, é uma função apenas dos parâmetros do i-ésimo neurônio. Esta propriedade facilita a implementação do algoritmo de aprendizado. Se a função de ativação é logística, como em 4.9, então $f'(a) = f(a)(1 - f(a))$ e o cálculo de δ_i^q é especialmente simples.

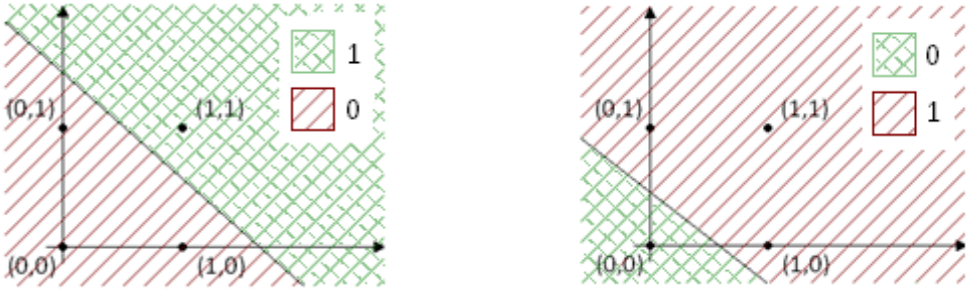
Se a taxa de aprendizado, η, for bastante pequena então os pesos sinápticos vão mudando bem suavemente, tornando o algoritmo mais lento e mais estável. Se por outro lado a taxa de aprendizado for grande os pesos sinápticos vão caminhar com maior velocidade podendo tornar o algoritmo de convergência instável. Costuma-se utilizar valores entre 0 e 1 para a taxa de aprendizado.

4.2.2. Problema da porta lógica ou-exclusivo

Através da regra delta as redes *perceptron* se mostraram capazes de aprender tarefas complexas com base na amostra de treinamento. No entanto, no fim dos anos 60 M. Minsky e S. Papert deixaram claro no livro intitulado *Perceptrons* (Minsk, 1969), do ponto de vista matemático, que a rede *perceptron* não pode resolver uma grande classe de problemas, embora eles possam facilmente resolver problemas similares.

A implementação da função lógica ou-exclusivo (XOR) é um exemplo clássico da incapacidade do *perceptron* na solução de problemas que não são linearmente separáveis, não existe nenhuma reta que separe os pontos cuja

operação XOR seja satisfeita. Na Figura 4.3 estão representadas as operações booleanas AND e OR, respectivamente. Em ambos os casos conseguimos separar as respostas (1's) das respostas (0's) através de uma reta. Nestes casos a rede *perceptron* consegue treinar os pesos e trabalhar com a função da porta lógica.



AND OR

Figura 4.3 – Funções lógicas AND e OR

Para o caso do operador XOR (ou-exclusivo) não existe nenhuma reta que separe os valores 0's dos valores 1's. A Tabela 4.1 representa a tabela verdade para as operações AND, OR e XOR e a Figura 4.4 ilustra a representação de 5 retas mostrando que nenhuma delas consegue separar as duas possíveis respostas.

Tabela 4.1 - Tabelas verdade

X1	X2	AND	OR	XOR
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

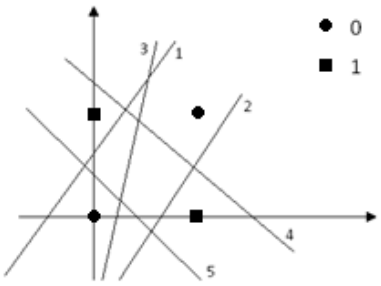


Figura 4.4 – Problema de separação linear Porta XOR

Se a função de ativação, $f(a)$, é uma função não-linear contínua, é impossível separar as duas classes com apenas uma curva do tipo $x_2 = f(x_1)$. Se o problema não é linearmente separável no espaço de saída p-dimensional, pelo menos um dos p neurônios do *perceptron* vai falhar na definição de uma linha discriminante no espaço de entrada. *Perceptrons* não podem resolver problemas não linearmente separáveis.

Uma das possíveis soluções para o problema da função XOR é a inclusão de uma camada intermediária de processadores – *Multilayer Perceptrons.*

4.3. *Multilayer Perceptrons*

Redes Neurais com mais de uma camada de neurônios artificiais, onde as camadas se conectam apenas com as camadas seguintes (rede *feedforward*) desde a camada de entrada até a camada de saída são chamadas de *Multilayer Perceptrons* (MLP). Cada MLP consiste em um conjunto de terminais de entrada, uma camada neural de saída e um número de camadas ocultas intermediárias, como mostra a Figura 4.5.

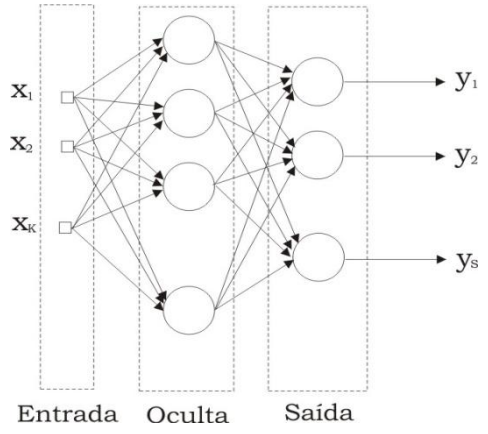


Figura 4.5 - *Multilayer Perceptron*

Em uma rede como a da Figura 4.5 (uma camada oculta), a primeira camada de neurônios realiza um pré-processamento do espaço de entrada e a segunda camada constrói as superfícies de discriminação necessárias para resolver o problema.

Uma rede *feedforward* de duas camadas, como na Figura 4.5, pode implementar qualquer função booleana fornecida se o número de neurônios ocultos for suficientemente grandes. Uma das possíveis soluções para o problema da função XOR é apresentada na rede da Figura 4.6:

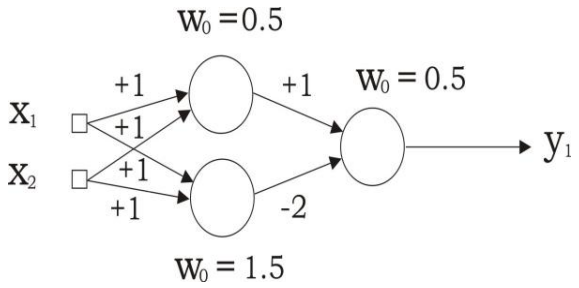


Figura 4.6 – Rede Neural MLP para implementar a função XOR

Para valores de entrada contínuos, uma camada de neurônios artificiais constrói um hiperplano discriminante no espaço de entrada, em um MLP a segunda camada da rede organiza esses hiperplanos de discriminação, de forma que as superfícies de separação convexas arbitrariamente complexas possam ser aproximadas. A terceira camada em uma rede MLP iria permitir também a definição de superfícies de separação disjuntas e não convexas no espaço de entrada. Se a função de ativação for logística, qualquer decisão em fronteira contínua pode ser bem aproximada por uma rede *perceptron* com uma camada intermediária com um número suficiente de neurônios ocultos. Então, redes MLP fornecem funções discriminantes não lineares universais.

As redes MLP não podem ser treinadas através da regra delta, pois não existem valores desejados de resposta para os neurônios das camadas ocultas.

Se um neurônio produz uma saída incorreta para um dado vetor de entrada, é impossível detectar qual dos neurônios das camadas escondidas cometeu um equívoco e conseqüentemente quais pesos necessitam ser ajustados. O algoritmo de aprendizado utilizado neste caso é o algoritmo de retro-propagação (*back-propagation*).

4.3.1. Algoritmo de aprendizado *back-propagation*

O algoritmo *back-propagation* implementa a estratégia do gradiente descendente para redes neurais multicamadas *feedforward*. O algoritmo consiste em dois passos:

1. A propagação desde os neurônios da camada de entrada até os neurônios da camada de saída;
2. A retro-propagação (*back-propagation*) do vetor de erro desde a camada de saída até a camada de entrada da rede.

Para a aplicação desta estratégia de minimização da função de erro deve-se definir uma função de erro arbitrária. A função escolhida neste trabalho é a soma de quadrados dos resíduos $\varphi^q(\boldsymbol{W})$.

O primeiro passo do algoritmo consiste na aplicação dos valores de entrada e no cálculo dos valores de saída através das camadas da rede. A obtenção do valor da função de erro $\varphi^q(\boldsymbol{W})$ na camada de saída conclui o primeiro passo do algoritmo.

Considere um conjunto de treinamento com m exemplos e uma rede MLP com L camadas. Se o erro na camada de saída $\varphi^q(\boldsymbol{W})$ é diferente de zero, é necessária uma atualização $\Delta\boldsymbol{W}^q$ da matriz de pesos. Assim como na regra delta, seguindo a estratégia do gradiente descendente, os pesos são atualizados inicialmente na última camada, penúltima e assim

sucessivamente até a atualização dos pesos na camada de entrada. O peso w_{ik} conecta o neurônio k na camada $c-1$ para a unidade i na camada c.

$$\Delta w_{ik}^{q} = -\eta \frac{\partial \varphi^{q}}{\partial w_{ik}} \tag{4.15}$$

Usando a regra da cadeia, a derivada pode ser expressa como:

$$\frac{\partial \varphi^{q}}{\partial w_{ik}} = \frac{\partial \varphi^{q}}{\partial a_{i}^{q}} \frac{\partial a_{i}^{q}}{\partial w_{ik}} \tag{4.16}$$

lembrando que

$$s_i = f_i(a_i) = f_i\left(\sum_{k=0}^{n(c-1)} w_{ik} s_k\right), \qquad i = 1, \ldots, n(c) \tag{4.17}$$

temos

$$\frac{\partial a_{i}^{q}}{\partial w_{ik}} = s_{i}^{q} \tag{4.18}$$

usando

$$\delta_{i}^{q} = \frac{\partial \varphi^{q}}{\partial a_{i}^{q}} \tag{4.19}$$

encontra-se a derivada

$$\frac{\partial \varphi^{q}}{\partial w_{ik}} = \delta_{i}^{q} s_{i}^{q} \tag{4.20}$$

Então,
$$\Delta w_{ik}^{q} = -\eta \delta_{i}^{q} s_{i}^{q} \tag{4.21}$$

onde
$$i \in camada\ c$$
$$k \in camada\ c-1$$

tendo então a mesma forma da regra delta para a atualização dos pesos em uma rede *perceptron* de uma camada.

Para os neurônios da camada de saída a regra delta é:

$$\delta_i^q = \frac{\partial \varphi^q}{\partial a_i^q} = f'(a_i)\left(s_i^q - y_i^q\right), \quad i \in camada\ saída\ C \tag{4.22}$$

O problema agora é decidir como calcular $\frac{\partial \varphi^q}{\partial a_i^q}$ para as unidades das camadas ocultas. Usando novamente a regra da cadeia para derivadas parciais, pode-se introduzir a contribuição da camada $c+1$ para o cálculo de δ_i^q.

$$\delta_i^q = \frac{\partial \varphi^q}{\partial a_i^q} = \sum_{j=1}^{n(c+1)} \frac{\partial \varphi^q}{\partial a_j^q} \frac{\partial a_j^q}{\partial a_i^q} \tag{4.23}$$

onde

$n(c+1)$ é o número de unidades j na camada $c+1$. O termo $\frac{\partial \varphi^q}{\partial a_j^q}$ é exatamente a quantidade δ_i^q definida para unidades j na camada $c+1$ e, considerando a equação 4.17 tem-se $\frac{\partial a_j^q}{\partial a_i^q} = f'\left(a_i^q\right) w_{ji}$. Então, para a camada oculta i:

$$\delta_i^q = f'\left(a_i^q\right) \sum_{j=1}^{n(c-1)} w_{ji}\, \delta_j^q \tag{4.24}$$

$$i \in camada\ c < C$$
$$j \in camada\ c+1$$

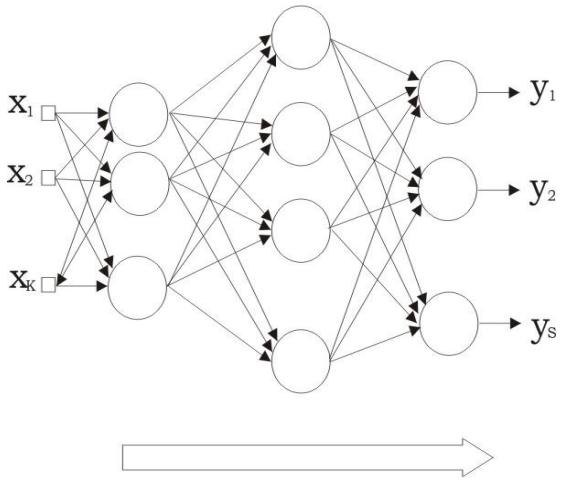


Figura 4.7 – *Feedforward* ilustração da propagação do vetor de entrada para o cálculo do erro

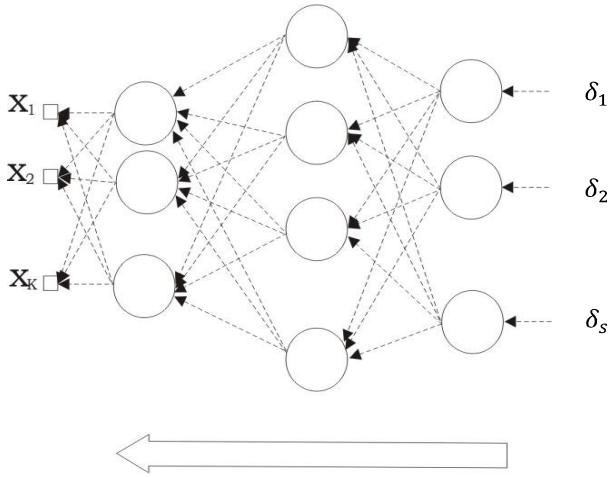


Figura 4.8 – *back-propagation* ilustração da contrária do vetor de erros para atualização da matriz de pesos

Começando com o cálculo dos valores δ na camada de saída C, é possível calcular os valores de δ dos neurônios de uma das camadas ocultas $c < C$ através do uso dos valores δ da camada $c + 1$. Quando todos os pesos estiverem atualizados com Δw_{ik}^{q}, o seguinte exemplo de treinamento é

apresentado e o procedimento re-começa. O critério de parada pode ser uma regra baseada da função de erro ou uma pré-determinação do número de épocas utilizadas no treinamento.

O algoritmo de aprendizado *back-propagation* pode ser resumido da seguinte forma:

1. Inicialize os pesos sinápticos com valores nulos;
2. Apresente a primeira linha do banco de dados, $\boldsymbol{x}^q$, para a camada de entrada;
3. Propague $\boldsymbol{x}^q$ pelas camadas desde a camada de entrada até a camada de saída;
4. Calcule o erro $\varphi^q(\boldsymbol{W})$ na camada de saída;
5. Calcule os valores δ da camada de saída;
6. Calcule os valores δ das camadas anteriores, de acordo com o passo de *back-propagation*;
7. Use $\Delta w_{ik}^q = -\eta \delta_i^q s_i^q$, onde $i \in camada\ c$ e $k \in camada\ c-1$, para todo w_{ik} da rede;
8. $q \rightarrow q+1$ e volte para 2.

O algoritmo *back-propagation* tem sido utilizado com resultados satisfatórios para um grande número de problemas de classificação, predição e aproximação de funções. E é o procedimento de aprendizado mais conhecido para MLP.

4.3.2. Taxa de aprendizagem e constante *momentum*

A escolha da taxa de aprendizagem deve ser feita cuidadosamente no intervalo (0,1). Para valores de η próximos de um a convergência da rede é mais rápida,

porém, para valores próximos de zero a rede pode acabar convergindo para um mínimo local.

Na maioria das vezes a regra de aprendizagem depende do parâmetro da taxa de aprendizagem podendo produzir oscilações ao redor de cada mínimo da função de erro, causando lentidão no processo de convergência ou até mesmo evitando-a. Para anular esta instabilidade costuma-se utilizar outra fórmula na atualização dos pesos sinápticos durante o processo de aprendizagem,

$$\Delta w_{ik}(u) = -\eta \frac{\partial \varphi(u)}{\partial w_{ik}} + \alpha . \Delta w_{ik}(u-1) \tag{4.25}$$

Onde α é uma constante positiva, usualmente escolhida entre 0.1 e 0.8, chamada constante *momentum*.

Quando a derivada $\frac{\partial \varphi(u)}{\partial w_{ik}}$ tem o mesmo sinal algébrico em iterações consecutivas, a série cresce em magnitude e os pesos sinápticos são ajustados em uma quantia grande, acelerando o processo de decida nas regiões de decida da superfície de erro. Em outro caso, quando a derivada $\frac{\partial \varphi(u)}{\partial w_{ik}}$ tem sinais opostos em iterações sucessivas então a série diminui em magnitude e $\Delta w_{ik}(u)$ é atualizado por uma quantia pequena. Desta forma, o termo *momentum* estabiliza a rede nas direções em que o sinal oscila.

Enquanto um valor pequeno para o parâmetro da taxa de aprendizagem resulta uma baixa convergência, ele pode localizar mínimos locais mais fundos na superfície de erro quando comparado com taxas de aprendizado maiores.

Para $\eta \to 0$, o uso de $\alpha \to 1$ resulta no aumento da velocidade de convergência. De outra forma, para $\eta \to 1$, o uso de $\alpha \to 0$ é necessário para garantir a estabilidade do aprendizado.

4.4. Generalização e *Overfitting*

O principal objetivo de um modelo de redes neurais artificiais não é implementar uma função ou memorizar amostras do conjunto de treinamento, mas sim construir um modelo que possa ser generalizado no relacionamento das entradas e saídas através do conjunto de treinamento, podendo ser aplicado para novos conjuntos de dados diferentes do conjunto de dados de treinamento. O principal objetivo de um modelo de redes neurais artificiais é a generalização da relação entre a entrada e a saída aprendida através do conjunto de treinamento.

Desta forma, a escolha do conjunto de treinamento se torna um ponto chave na capacidade de generalização do modelo ajustado. Os exemplos utilizados no conjunto de treino da rede devem representar a população de interesse, portanto devem ter um número significativo limpo e igualmente distribuído nas classes de saída.

Um grande número de parâmetros pode memorizar todos os exemplares do conjunto de treinamento, memorizando inclusive os erros, ruídos e inconsistências fazendo com que a rede perca a capacidade de generalização para novos dados. Este fenômeno é conhecido como super-ajuste (*overfitting*). A Figura 4.9 ilustra os dois extremos da modelagem. Em um extremo o modelo nulo, indicando apenas o que seria em um modelo clássico ou Bayesiano o intercepto, um modelo totalmente insensível a qualquer tendência, e no outro extremo os dados totalmente interpolados, um modelo totalmente sensível ao sinal da tendência e da variação aleatória única, onde a curva do modelo passa através de todas as observações (*overfitting*) com soma de quadrados dos resíduos próxima de zero.

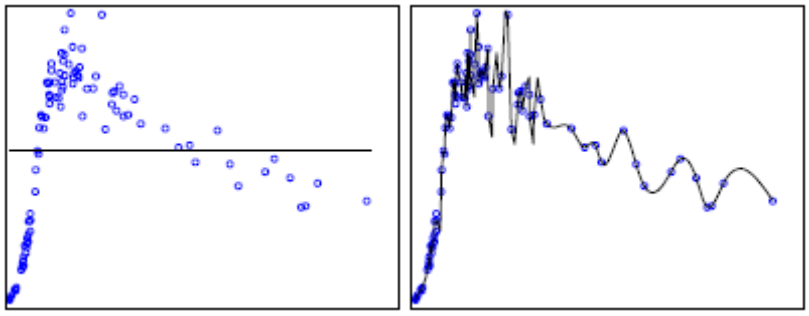


Figura 4.9 – Os dois extremos da modelagem

Nenhum dos casos da Figura 4.9 tem capacidade de generalização para novas observações. Um exemplo de modelo com boa capacidade de generalização é dado na Figura 4.10:

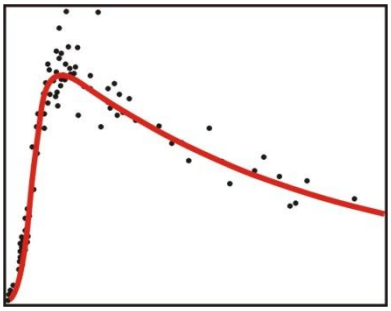


Figura 4.10 – Boa capacidade de generalização

Desta maneira, a topologia da rede deve ser escolhida de forma que a capacidade preditiva do modelo não se altere, ou seja, se mantenha aproximadamente com o mesmo desempenho do banco de dados de ajuste na base de teste.

Capítulo 5

Comparação de modelos

Para a comparação de modelos utilizamos algumas métricas que representam basicamente a capacidade de acerto dos modelos. Através destas métricas podemos comparar diferentes modelos, ajustados através de diferentes técnicas de forma imparcial.

Quatro técnicas serão apresentadas neste capítulo: Capacidade Preditiva, Curva ROC, Estatística de Kolmogorov Smirnov e a Capacidade de acerto dos modelos.

A Capacidade Preditiva é realizada através de re-amostragens via *bootstrap*, que tem como objetivo simular a realização do que seria desejável na prática: repetir a experiência. Através destas re-amostragens são encontrados intervalos de confiança para as estimativas de interesse utilizando amostras, com reposição, provenientes da amostra original, Rud (2001).

A curva ROC é construída através da variação dos pontos de corte, ao longo da amplitude dos escores fornecidos pelos modelos, a fim de se obter

diferentes classificações dos indivíduos e obtendo conseqüentemente os respectivos valores para as medidas de sensibilidade e especificidade para cada ponto de corte estabelecido.

A estatística de Kolmogorov-Smirnov, conhecida pela sigla KS, tem origem no teste de hipóteses não paramétrico onde se deseja testar se duas funções de distribuições associadas a duas populações são idênticas ou não a partir de duas amostras retiradas de populações possivelmente distintas. A estatística KS mede o quanto duas funções de distribuição empíricas dos escores dos grupos de bons e maus pagadores estão separadas. Ver Abreu (2005).

Foram calculadas também as capacidades de acerto total, de bons e de maus pagadores dos modelos para os respectivos pontos de corte selecionados em cada modelo. Estas técnicas foram baseadas em Abreu (2005).

5.1 Capacidade preditiva

A análise da capacidade preditiva utiliza o *bootstrap*, que é uma técnica empírica para encontrar intervalos ao redor de uma estimativa. O *bootstrap* utiliza amostras com reposição do mesmo tamanho do banco de dados. Como a amostra é realizada com reposição é possível que uma linha seja selecionada diversas vezes e outra não seja selecionada nenhuma vez. As amostras com reposição são obtidas a partir de uma amostra previamente selecionada para a validação, chamada de amostra de validação original. Ver Rud (2001).

Para o cálculo do intervalo de confiança bootstrap é necessário calcular primeiramente a estimativa bootstrap através da fórmula 5.1.

$$BS_{est} = 2 * estimativa\ amostral - média(BS_i) \tag{5.1}$$

Na fórmula 5.1 a estimativa amostral é obtida da amostra de validação original e BSi representa a i-ésima estimativa *bootstrap* obtida a partir da i-ésima amostra *bootstrap*.

A idéia do método é obter as estimativas intervalares através dos desvios padrão das estimativas das amostras *bootstraps*, os intervalos de confiança com $(1-\propto)\%$ de confiança são dados pelas fórmulas 5.2 e 5.3.

$$ICI = BS_{est} - \left|Z_{\propto/2}\right| * EP_{BS_i} \tag{5.2}$$

$$ICS = BS_{est} + \left|Z_{\propto/2}\right| * EP_{BS_i} \tag{5.3}$$

Para cada uma das amostras bootstraps obtidas observam-se as freqüências em cada decil para cada uma das medidas de interesse: risco estimado, taxa de inadimplência real observada e a participação da faixa na taxa de inadimplência geral da amostra, bem como seus respectivos limites de confiança.

Os decis são divididos de acordo com a amostra de validação original, no caso a partir da variável que contém a probabilidade de inadimplência ajustada pelo modelo (escore).

Após o ajuste do modelo, foram obtidos os escores para cada indivíduo do banco de validação. Os escores foram colocados em ordem decrescente e divididos em decis, ou seja, o primeiro decil contém o conjunto de indivíduos com maior risco e o décimo e último decil contém o conjunto de indivíduos de menor risco da base de dados.

Para cada decil foram calculadas as estimativas *bootstrap* e os respectivos intervalos de confiança para o risco, a taxa de inadimplência e a participação da faixa na taxa de inadimplência geral da amostra, conforme

dito anteriormente. As estimativas obtidas através do *bootstrap* são dadas de acordo com o algoritmo a seguir:

1. Obtenha da amostra original n amostras independentes e com reposição;
2. Repita o procedimento 100 vezes;
   - O conjunto de 100 amostras forma uma amostra *bootstrap*: $boot(i)$
   neste trabalho foram utilizadas 25 amostras bootstrap.
3. Calcule a estimativa *bootstrap* de cada amostra: $\theta_{boot(i)}$
4. Calcule o desvio padrão: σ_{boot};
5. Encontre a estimativa geral de *bootstrap*:
$$\theta_{boot} = 2\theta_{original} - \frac{\sum \theta_{boot(i)}}{25};$$
6. Encontre os limites do intervalo de confiança:
$$\theta_{superior} = \theta_{boot} + Z\alpha_{/2}\sigma_{boot}$$
$$\theta_{inferior} = \theta_{boot} - Z\alpha_{/2}\sigma_{boot};$$
onde $Z\alpha_{/2}$ é o valor da distribuição normal para a construção do intervalo de confiança $100(1-\alpha)\%$.

Após a execução do algoritmo obtemos uma tabela como a Tabela 5.1 abaixo onde todas as medidas são sumarizadas. A tabela pode ser analisada através de gráficos.

Tabela 5.1 –Exemplos de resultados finais bootstrap

decil	Risco				Taxa de inadimplência				Participação			
	obs	*Bootstrap*			obs	*Bootstrap*			obs	*Bootstrap*		
		est	LI	LS		est	LI	LS		est	LI	LS
0	51.64	52.35	19.24	85.46	50.00%	50.09%	16.98%	83.19%	500	473	131	815
1	19.58	20.15	0.7	51.16	33.33%	33.33%	2.31%	64.35%	333	325	44	606
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
9	0.04	0.03	0.03	0.03	0.00%	0.00%	0.00%	0.00%	0	0	0	0

5.2 Curva ROC

A curva ROC pode ser utilizada na escolha do melhor ponto de corte por envolver as medidas de sensibilidade e especificidade. Estas quantidades são dadas pelas probabilidades em 5.4 e 5.5, onde a sensibilidade é a probabilidade de o cliente ser classificado como mau pagador dado que ele realmente é um mau pagador e a especificidade é a probabilidade de o cliente ser classificado como bom pagador dado que ele realmente é um bom pagador. A estratégia de escolha do ponto de corte adotada neste trabalho foi escolher o ponto que maximizasse a sensibilidade e a especificidade através da intersecção entre as curvas de sensibilidade e especificidade construídas para cada ponto de corte. Assim como na curva ROC esta alternativa indica o ponto de corte que maximiza a sensibilidade e a especificidade.

$$Sensibilidade = P(MAU|MAU) \tag{5.4}$$

$$Especificidade = P(BOM|BOM) \tag{5.5}$$

A curva é construída através da variação dos pontos de corte ao longo da amplitude dos escores fornecidos pelos modelos, a fim de se obter as diferentes classificações dos indivíduos e obtendo, para cada ponto de corte temos uma matriz de confusão e a partir de cada matriz de confusão temos uma medida de sensibilidade e especificidade. O eixo horizontal do gráfico é construído a partir dos valores de (1 – Especificidade) e o eixo vertical é construído a partir dos valores de Sensibilidade. A tabela 5.2 é um exemplo de matriz de confusão.

Tabela 5.2 – Matriz de confusão

		Observado		
		BOM	MAU	
Predito	BOM	b_B	b_M	b
	MAU	m_B	m_M	m
		B	M	

A curva ROC será utilizada aqui no intuito de comparar os diferentes modelos ajustados, a sensibilidade é representada pelo eixo Y e o valor de (1-especificidade) é representado pelo eixo X.

Na comparação de modelos, o melhor modelo é aquele que apresentar a maior área sob a curva ROC.

5.3 Estatística de Kolmogorov-Smirnov (KS) e capacidade de acerto dos modelos

Outra forma de comparação de modelos muito utilizada é a medida da estatística de Kolmogorov-Smirnov (KS). Baseada no teste não-paramétrico de Kolmogorov Smirnov onde se deseja a partir de duas amostras retiradas de populações possivelmente distintas, testar se duas funções distribuições associadas às duas populações são idênticas ou não.

A estatística de Kolmogorov-Smirnov mede o quanto estão separadas as funções distribuições empíricas dos escores dos grupos de bons e maus pagadores. Sendo $F_B(e) = \sum_{x \leq e} F_B(x)$ e $F_M(e) = \sum_{x \leq e} F_M(x)$ a função distribuição empírica dos bons e maus pagadores respectivamente, então a estatística de Kolmogorov-Smirnov é dada por 5.4

$$KS = \max|F_B(e) - F_M(e)| \tag{5.6}$$

onde $F_B(e)$ e $F_M(e)$ corresponde às proporções de clientes bons e maus com escore menor ou igual a e, e a estatística KS é obtida através da distância máxima entre essas duas proporções acumuladas ao longo dos escores obtidos pelos modelos, representada na Figura 5.1

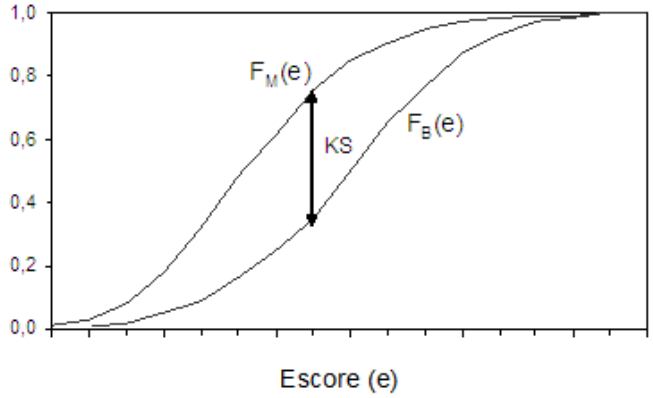


Figura 5.1 – Funções de distribuições empíricas para os bons e maus clientes e a estatística KS

O valor dessa estatística pode variar de 0% a 100%, onde o valor máximo indica uma separação total dos escores dos bons e maus clientes e um valor mínimo de 0% sugere uma sobreposição total das distribuições dos escores dos dois grupos. A representação da interpretação dessa estatística pode ser vista na Figura 5.2.

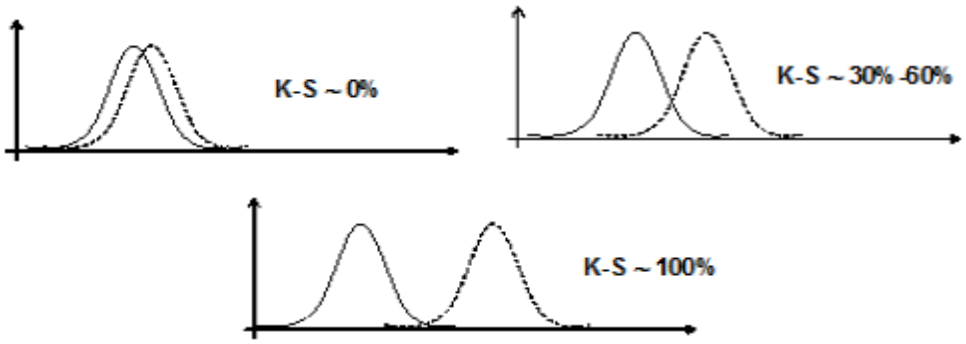


Figura 5.2 – Interpretação da estatística KS

Na Figura 5.2 pode-se notar que o KS maior indica a maior distância entre as distribuições dos maus e dos bons clientes.

A capacidade de acerto do modelo é calculada após a definição de um ponto de corte. É medida sob três perspectivas: Capacidade de Acerto Total (CAT), que é medida entre todas as amostras do banco de teste, a Capacidade de Acerto dos Maus (CAM), que é a medida apenas entre os maus clientes e pode também ser chamada de Especificidade e finalmente a Capacidade de Acerto dos Bons (CAB), que é medida apenas entre os bons clientes, também chamada de Sensibilidade. As fórmulas estão em 5.7:

$$CAT = \frac{b_B + m_M}{n} \qquad CAM = \frac{m_M}{M} \qquad CAB = \frac{b_B}{B} \tag{5.7}$$

onde:

n: número total de clientes na amostra;
b_B: número de bons clientes que foram classificados como bons (acerto);
m_M: número de maus clientes que foram classificados como maus (acerto);
M: total de maus clientes na amostra;
B: total de bons clientes na amostra.

Capítulo 6

Aplicação em dados artificiais

Este capítulo tem como objetivo aplicar e comparar as técnicas apresentadas em um conjunto de dados gerado artificialmente usando distribuição normal multivariada (Breiman, 1988). As técnicas de Regressão Logística Clássica e Bayesiana e de Redes Neurais Artificiais foram comparadas de acordo com a estatística de Kolmogorov-Smirnov (KS), curva ROC, capacidade preditiva e a capacidade de acerto dos modelos.

O banco de dados artificial utilizado neste capítulo consiste originalmente de vinte variáveis explicativas e uma variável resposta dicotômica, com 5.000 linhas usadas no conjunto de treinamento e 2.150 linhas usadas no conjunto de teste, cada um com taxa de inadimplência de aproximadamente 10%.

A variável resposta deste conjunto de dados gerado é nomeada de Default e é igual a um se o cliente (linha do banco de dados) é inadimplente e zero caso contrário. Para cada uma dessas duas classes, as variáveis explicativas são obtidas a partir da geração de uma distribuição normal

multivariada com matriz de covariâncias unitária. O vetor de médias para a classe (Default=1) é igual a $(a, a, \ldots, a)$ e o vetor de médias para a classe (Default=0) é igual a $(-a, -a, -a, \ldots, -a)$, onde $a = 2/\sqrt{20}$.

Os ajustes dos modelos utilizando todas as vinte variáveis disponíveis apresentaram uma taxa de acerto extremamente elevada, modelos de Credit Score não costumam ter uma taxa de predição no acerto muito alta e por este motivo trabalhou-se apenas com as cinco variáveis mais significativas para que a taxa de acerto do modelo fosse mais verossímil.

O programa utilizado para a geração do banco de dados pode ser visto no apêndice.

Cada uma das cinco variáveis trabalhadas foi categorizada através da divisão dos valores em decis e em seguida agrupou-se os decis de acordo com a contribuição na explicação da variável resposta, esta quantificada através da razão de chances e através da taxa de inadimplência das faixas construídas. Esta categorização é também chamada de análise bivariada por conter a variável explicativa em questão e a variável resposta.

Os métodos de seleção de variáveis utilizados foram: *stepwise*, *forward* e *backward*. Todos eles selecionaram as mesmas cinco variáveis entre as variáveis disponibilizadas. As variáveis são não correlacionadas entre si.

6.1 Aplicação no banco de dados de ajuste

Esta seção apresenta os resultados obtidos a partir da amostra de ajuste sendo utilizadas as técnicas de Regressão Logística Clássica, Regressão Logística Bayesiana e Redes Neurais Artificiais. Após a pré-seleção e categorização das variáveis citadas na seção anterior, realizou-se o ajuste de regressão logística clássico de acordo com a estratégia de construção de

modelos proposta por Hosmer e Lemeshow (2000), estimando os coeficientes do vetor β por máxima verossimilhança. O ajuste realizado pela abordagem Bayesiana e de Redes Neurais utilizou as mesmas variáveis do modelo clássico.

6.1.1 Ajuste do modelo de regressão logística clássico

As análises univariadas, bivariadas e de correlação citadas na seção anterior não excluíram nenhuma variáveis. Conforme a estratégia de construção de modelos adotada ajustou-se um modelo de regressão logística univariado para cada uma das variáveis disponibilizadas já com as variáveis categorizadas (CAT_V1, CAT_V2, CAT_V3, CAT_V4, CAT_V5) e observou-se a significância das mesmas de acordo com o teste Wald. Os ajustes obtidos podem ser vistos na Tabela 6.1. Todas as variáveis e seus níveis são significativos.

Através dos procedimentos mecânicos de seleção, como *stepwise*, *forward* e *backward* todas as variáveis submetidas à seleção foram selecionadas. Estes métodos podem ser criticados por serem não plausíveis com o estudo em andamento dependendo da opinião do especialista. As variáveis categorizadas foram transformadas em variáveis *dummy* de acordo com a quantidade de categorias de cada uma, ou seja, a *i*-ésima variável CAT_V*i* deu origem as dummys DUM1_V*i*, DUM2_V*i*, ..., DUMw_V*i*, onde w representa a quantidade de categorias da *i*-ésima variável subtraída de uma unidade. Os estimadores de máxima verossimilhança dos coeficientes do modelo multivariado são dados na Tabela 6.2.

Tabela 6.1 – Estimadores de máxima verossimilhança para ajustes univariados

Ajuste	Variável		GL	Estimativa do parâmetro	Erro padrão	Qui-Quad Wald	Pr > Qui-Quad
1	Intercept		1	2.3095	0.1377	281.4012	<.0001
	CAT_V1	C1	1	-0.8395	0.1502	31.2399	<.0001
	CAT_V1	C3	1	0.5195	0.1846	7.9196	0.0049
	CAT_V1	C4	1	0.9815	0.2531	15.0416	0.0001
	CAT_V1	C5	1	1.6545	0.4345	14.5004	0.0001
	CAT_V1	C6	1	2.7456	0.7222	14.4536	0.0001
2	Intercept		1	2.4218	0.1174	425.5596	<.0001
	CAT_V2	C1	1	-0.989	0.1322	55.9896	<.0001
	CAT_V2	C3	1	0.5114	0.1725	8.7953	0.003
	CAT_V2	C4	1	1.2624	0.279	20.4706	<.0001
	CAT_V2	C5	1	2.2633	0.5917	14.6297	0.0001
3	Intercept		1	2.0725	0.125	274.6758	<.0001
	CAT_V3	C1	1	-0.5471	0.1392	15.4412	<.0001
	CAT_V3	C3	1	0.9513	0.166	32.8425	<.0001
	CAT_V3	C4	1	1.6555	0.2897	32.6673	<.0001
4	Intercept		1	2.5447	0.0968	690.469	<.0001
	CAT_V4	C1	1	-1.0619	0.1141	86.6789	<.0001
	CAT_V4	C3	1	0.4998	0.2133	5.4907	0.0191
	CAT_V4	C4	1	1.1818	0.2787	17.987	<.0001
	CAT_V4	C5	1	2.106	0.5881	12.8237	0.0003
5	Intercept		1	2.4198	0.1182	419.42	<.0001
	CAT_V5	C1	1	-0.979	0.1323	54.7205	<.0001
	CAT_V5	C3	1	0.8914	0.1716	26.9988	<.0001
	CAT_V5	C4	1	1.3899	0.3999	12.0764	0.0005

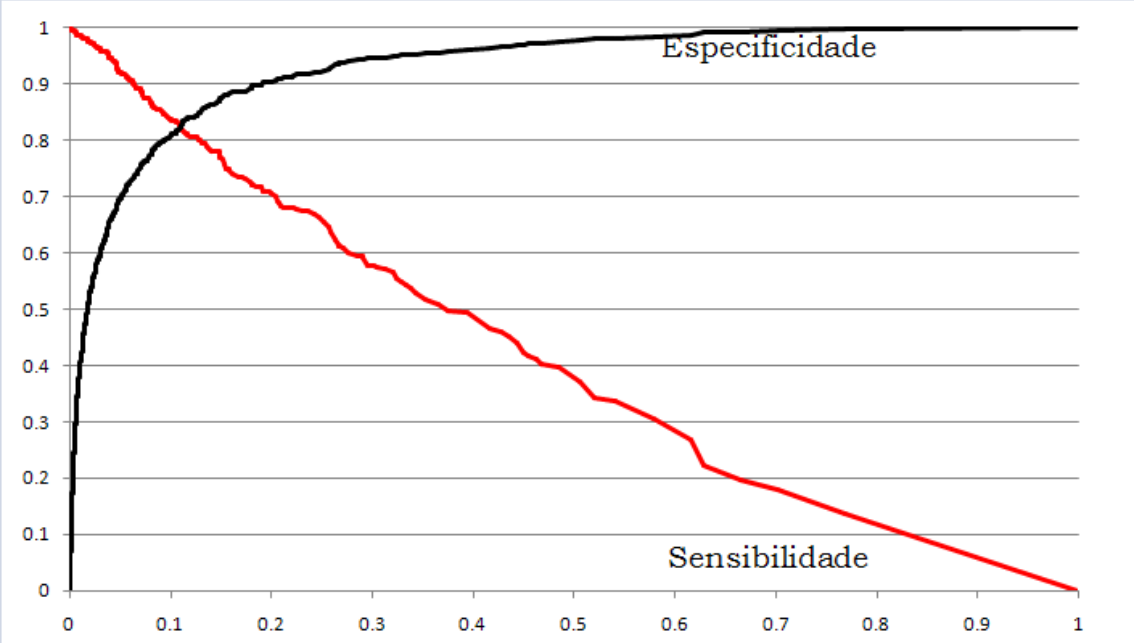
Tabela 6.2 – Estimadores de máxima verossimilhança para o modelo multivariado

Parâmetro	GL	Estimativa	Erro padrão	Qui-Quad Wald	Pr > Qui-Quad
Intercept	1	-3.1141	0.299	108.5233	<.0001
DUM1_V1	1	0.7608	0.178	18.3726	<.0001
DUM2_V1	1	-0.6215	0.211	8.6497	0.0033
DUM3_V1	1	-0.9853	0.28	12.4153	0.0004
DUM4_V1	1	-1.8291	0.459	15.8685	<.0001
DUM5_V1	1	-2.8334	0.74	14.6751	0.0001
DUM1_V2	1	1.1336	0.156	52.7639	<.0001
DUM2_V2	1	-0.4373	0.194	5.1052	0.0239
DUM3_V2	1	-1.0992	0.298	13.5955	0.0002
DUM4_V2	1	-2.1309	0.609	12.253	0.0005
DUM1_V3	1	0.6906	0.17	16.463	<.0001
DUM2_V3	1	-0.8519	0.193	19.4738	<.0001
DUM3_V3	1	-1.6387	0.316	26.9616	<.0001
DUM1_V4	1	1.0835	0.135	64.194	<.0001
DUM2_V4	1	-0.6255	0.236	7.0344	0.008
DUM3_V4	1	-1.1974	0.295	16.497	<.0001
DUM4_V4	1	-2.0739	0.603	11.8278	0.0006
DUM1_V5	1	1.0213	0.155	43.2199	<.0001
DUM2_V5	1	-0.9179	0.191	23.024	<.0001
DUM3_V5	1	-1.551	0.426	13.2413	0.0003

Após o ajuste do modelo é necessário definir um ponto de corte para definir se o cliente é ou não inadimplente. Neste trabalho utilizou-se como critério para a definição do ponto de corte o ponto que maximiza a sensibilidade e a especificidade do modelo. A sensibilidade é a probabilidade de o modelo classificar o cliente como inadimplente dado que ele realmente é inadimplente e a especificidade é a probabilidade de se classificar o cliente como adimplente dado que ele realmente é adimplente.

O gráfico 6.1 indica o melhor ponto de corte para o modelo de regressão logística ajustado, aproximadamente 0,1137.

Gráfico 6.1 – Especificidade e Sensibilidade x Ponto de corte



Após a definição do ponto de corte escolhe-se a regra de decisão para a classificação dos clientes como bons ou maus pagadores. Esta forma de escolha otimiza a sensibilidade e a especificidade do modelo. A aplicação do modelo clássico no banco de dados de teste pode ser vista na seção 6.3 de comparação dos modelos.

6.1.2 Ajuste do modelo de regressão logística Bayesiano

As mesmas variáveis do modelo final clássico foram utilizadas no ajuste Bayesiano. Como as variáveis foram todas categorizadas na análise inicial as

cinco variáveis do ajuste final na análise clássica foram transformadas em variáveis dummy, somando ao todo vinte parâmetros de interesse. Uma das vantagens do ajuste Bayesiano é a possibilidade de inclusão de um conhecimento externo aos dados, normalmente dado por um especialista através de uma distribuição *a priori* para os parâmetros de interesse. Neste trabalho foi ajustado um modelo com priori não informativa (uniforme imprópria), e espera-se que os parâmetros ajustados pela metodologia Bayesiana sejam próximos dos parâmetros estimados pela metodologia clássica já que ambas estão baseadas apenas na informação contida na função de verossimilhança.

O modelo foi ajustado no software WinBUGS que utiliza o algoritmo de Metropolis para a obtenção da distribuição *a posteriori*. As variáveis foram colocadas no programa na seguinte ordem: intercepto (beta00), DUM1_V1 (beta01), DUM2_V1 (beta02), DUM3_V1 (beta03), DUM4_V1 (beta04), DUM5_V1 (beta05), DUM1_V2 (beta06), DUM2_V2 (beta07), DUM3_V2 (beta08), DUM4_V2 (beta09), DUM1_V3 (beta10), DUM2_V3 (beta11), DUM3_V3 (beta12), DUM1_V4 (beta13), DUM2_V4 (beta14), DUM3_V4 (beta15), DUM4_V4 (beta16), DUM1_V5 (beta17), DUM2_V5 (beta18), DUM3_V5 (beta19).

A *priori* não informativa dada para os parâmetros nesta aplicação foi imprópria com distribuição uniforme. A distribuição *a posteriori* neste caso é proporcional à verossimilhança.

$$\pi(\beta|Y) \propto l(\beta|Y)$$

O método utilizado na obtenção da *posteriori* foi o Metropolis Hastings, com um *burn-in* de tamanho 1.000 e em seguida obtendo uma amostra com 5.000 gerações, ambos os casos com saltos de tamanho igual a dez. O tamanho do *burn-in*, da amostra e dos saltos foram escolhidos de forma que as

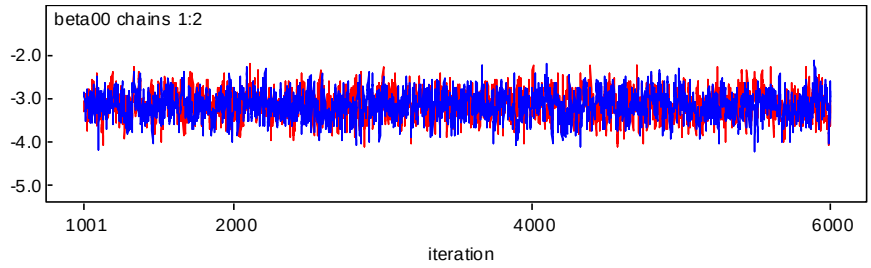
cadeias tivessem todas as suposições satisfeitas, se ajustando conforme a não observação das suposições. Duas cadeias foram geradas e tinham como valores iniciais os parâmetros estimadores de máxima verossimilhança e valores gerados aleatoriamente pelo programa.

Antes de observar as distribuições a posteriori marginais dos parâmetros. São necessárias algumas verificações de convergência e de auto-correlação das cadeias geradas. Essas verificações servem para garantir que as cadeias tenham convergido para a distribuição de interesse.

Inicialmente observa-se o histórico das cadeias com o objetivo de identificar possíveis problemas de convergência, como não aleatoriedade dos valores gerados e presença de tendência. Espera-se que o valor esteja sempre em torno de um valor fixo de forma aleatória. No Gráfico 6.2 pode-se observar o histórico das cadeias geradas para um dos parâmetros de interesse, os demais históricos estão disponíveis no apêndice.

Gráfico 6.2 – Histórico das cadeias de Markov para um dos parâmetros do modelo

Beta 00: (intercepto)



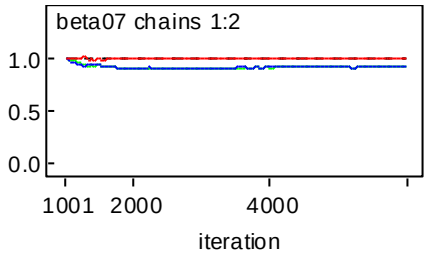
Pôde-se observar em todos os históricos que as cadeias permaneceram aleatórias ao longo das iterações, indicando convergência. Antecipadamente pode-se ter uma idéia da significância dos parâmetros através da distribuição dos pontos gerados. Se os pontos gerados estiverem abrangendo o valor zero é

um indício de que, mesmo a cadeia convergindo, o respectivo parâmetro será não significativo.

Outra forma para a verificar a convergência das cadeias é através da estatística de Gelman Rubin. Um valor próximo de 1, indica convergência da cadeia. O Gráfico 6.3 apresenta o gráfico de Gelman Rubin mais distante do valor um entre todos os parâmetros do modelo, como se pode observar o pior caso ainda é próximo do valor um indicando convergência da cadeia.

Gráfico 6.3 – Gráfico de Gelman Rubin para o pior caso encontrado

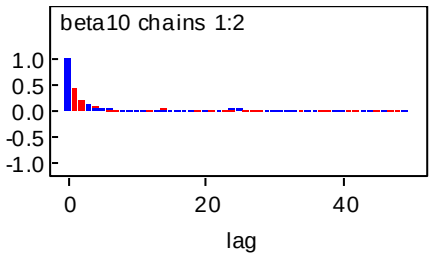
Beta 07: (DUM2_V2)



Em todas as cadeias de parâmetros do modelo observou-se a estatística de Gelman Rubin próxima de 1, reforçando assim a indicação de convergência fornecida pelos históricos das cadeias visto anteriormente.

Os valores da posteriores são gerados a partir das cadeias de Markov. Espera-se que estes valores não sejam auto-correlacionados. Para realizar esta verificação observa-se o gráfico de auto-correlação para cada uma das cadeias geradas. O decaimento da auto-correlação deve ser o mais rápido possível. O Gráfico 6.4 apresenta o gráfico de auto-correlação para o parâmetro beta10, que é o caso mais correlacionado para as variáveis do banco de dados, ainda assim, pouco correlacionado.

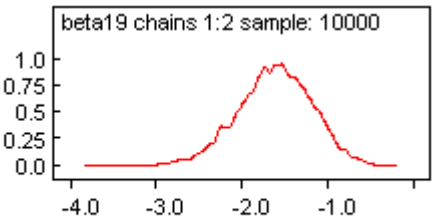
Gráfico 6.4 – Gráfico de auto-correlação das cadeias geradas



Após a verificação das suposições feitas para as cadeias geradas pode-se analisar as distribuições *a posteriori*. No Gráfico 6.5 pode-se observar a aproximação da densidade *a posteriori* para um dos parâmetros de interesse.

Gráfico 6.5 – Densidades a posteriori para os parâmetros de interesse

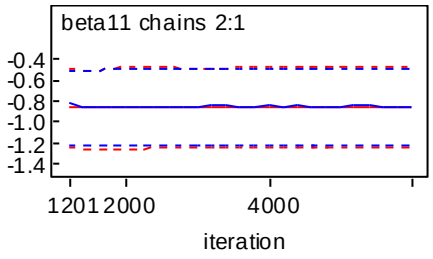
Beta 19: (DUM3_V5)



Todas as distribuições a posteriori são aproximadamente simétricas como pode-se verificar no apêndice, sendo o caso do Gráfico 6.5 o mais assimétrico observado. As médias dos parâmetros são muitos próximos do valor estimado através do método de máxima verossimilhança, como era esperado já que as prioris utilizadas no ajuste eram não informativas. Pode-se notar que as densidades dificilmente incluem o valor zero, ocorrendo em alguns poucos casos de o valor zero estar bem próximo ao limite das caudas da distribuição. Todos os gráficos dos intervalos de credibilidade não incluem o valor zero, indicando assim a significância dos parâmetros ajustados. O caso mais próximo de zero pode ser visto no Gráfico 6.6, os demais gráficos estão no apêndice.

Gráfico 6.6 – Intervalo de credibilidade para os parâmetros ajustados

Beta 11: (DUM2_V3)



A Tabela 6.3 apresenta as estatísticas dos parâmetros Bayesianos estimados. Os parâmetros obtidos são muito próximos dos estimados por máxima verossimilhança, como era esperado. Pode-se observar que todos os parâmetros são significativos através dos intervalos de credibilidade das distribuições *a posteriori*.

Tabela 6.3 – Estimadores Bayesianos

Parâmetro	Média	Erro Padrão	2.5%	Mediana	97.5%
beta00	-3.156	0.2973	-3.152	-3.745	-2.59
beta01	0.7753	0.1774	0.7733	0.434	1.132
beta02	-0.62	0.2116	-0.6177	-1.043	-0.2085
beta03	-1.001	0.2809	-0.9972	-1.558	-0.4608
beta04	-1.901	0.4737	-1.877	-2.897	-1.036
beta05	-3.091	0.82	-3.012	-4.92	-1.749
beta06	1.145	0.1556	1.143	0.8441	1.454
beta07	-0.442	0.1944	-0.4404	-0.824	-0.0556
beta08	-1.123	0.3005	-1.116	-1.721	-0.5577
beta09	-2.312	0.6612	-2.253	-3.771	-1.168
beta10	0.7009	0.1699	0.7001	0.3692	1.036
beta11	-0.854	0.1932	-0.8538	-1.233	-0.4787
beta12	-1.668	0.3196	-1.661	-2.328	-1.074
beta13	1.093	0.1358	1.093	0.8322	1.366
beta14	-0.636	0.2355	-0.6338	-1.106	-0.1818
beta15	-1.23	0.2969	-1.215	-1.848	-0.6826
beta16	-2.238	0.6532	-2.186	-3.662	-1.098
beta17	1.033	0.1555	1.031	0.7363	1.345
beta18	-0.919	0.191	-0.9205	-1.294	-0.5451
beta19	-1.618	0.4366	-1.602	-2.524	-0.8007

O ponto de corte foi escolhido de forma que maximize a sensibilidade e a especificidade do modelo. O Gráfico 6.7 indica o melhor ponto de corte para o modelo de Regressão Logística Bayesiano ajustado, aproximadamente 0.1067.

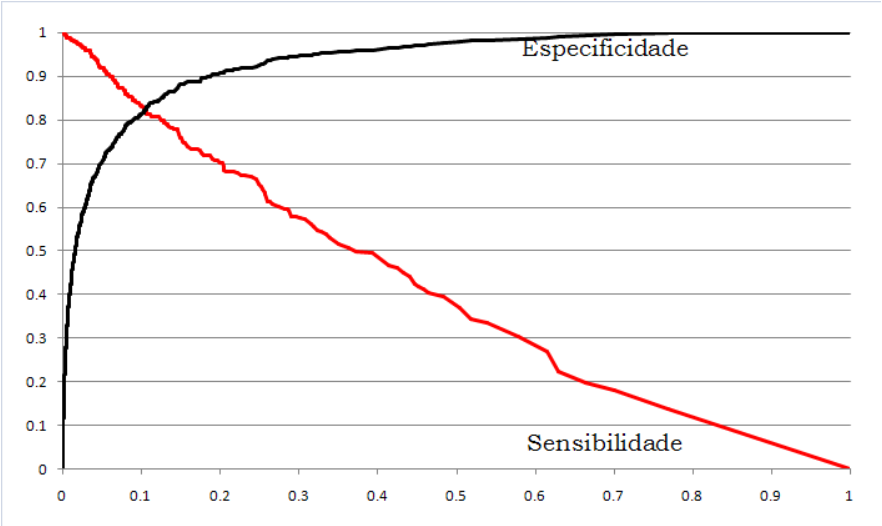


Gráfico 6.7 – Especificidade e Sensibilidade x Ponto de Corte

O ponto de corte define a melhor regra de decisão para a classificação dos clientes como bons ou maus pagadores. Esta forma de escolha otimiza a sensibilidade e a especificidade do modelo. A aplicação do modelo logístico Bayesiano no banco de dados de teste pode ser vista na seção 6.3 de comparação de modelos.

6.1.3 Ajuste do modelo de redes neurais

Um modelo de redes neurais não possui testes de significância para as variáveis. O que se costuma fazer na prática em modelos de *Credit Score* é utilizar variáveis pré-selecionadas por outras técnicas como, por exemplo, regressão logística. As mesmas variáveis *dummy* utilizadas no modelo logístico clássico e Bayesiano foram apresentadas às redes neurais ajustadas neste trabalho.

As redes neurais foram ajustadas no software SAS através do pacote Enterprise Miner. Este pacote utiliza o procedimento *neural* (proc neural) que é

bastante flexível com relação à arquitetura da rede, parâmetros de inicialização e técnicas de treino.

Inicialmente ajustou-se uma rede neural com um único neurônio e em seguida redes *multilayer perceptron* com uma e duas camadas ocultas.

6.1.3.1 Um único neurônio

Um simples neurônio pode ser considerado um modelo linear generalizado, onde a função de ativação da rede se equivale à função de ligação. O modelo de regressão logística é um caso particular dos modelos lineares generalizados.

A arquitetura desta rede é descrita na Figura 6.1.

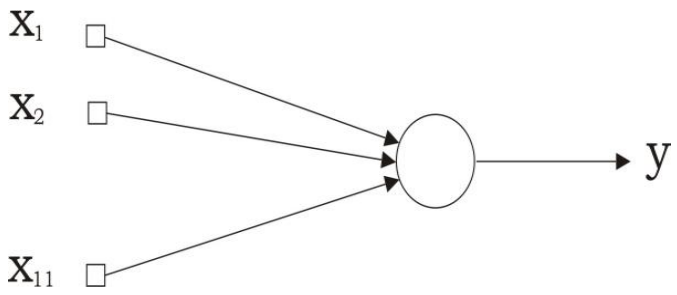


Figura 6.1 – Arquitetura de rede neural com um único neurônio

Através do método de aprendizado da regra delta foram ajustadas várias redes para diferentes combinações entre os parâmetros: taxa de aprendizado e constante *momentum*. As seguintes combinações foram testadas para todas as redes ajustadas neste capítulo: $(\eta = 1; \alpha = 0), (\eta = 0{,}7; \alpha = 0{,}3), (\eta = 0{,}3; \alpha = 0{,}7), (\eta = 0.1; \alpha = 1)$. Os valores $\eta = 1$ e $\alpha = 0$ foram escolhidos após o ajuste de diferentes redes com estas variações de combinações de η e α através dos

valores das estatísticas extraídas do ajuste do modelo no banco de dados de treinamento: erro médio, soma dos quadrados dos resíduos e o número de parâmetros no modelo. A função de ativação logística foi escolhida para a rede.

O número de épocas utilizadas no treinamento da rede foi igual a 29. O gráfico do erro médio ao longo das iterações é dado no Gráfico 6.8.

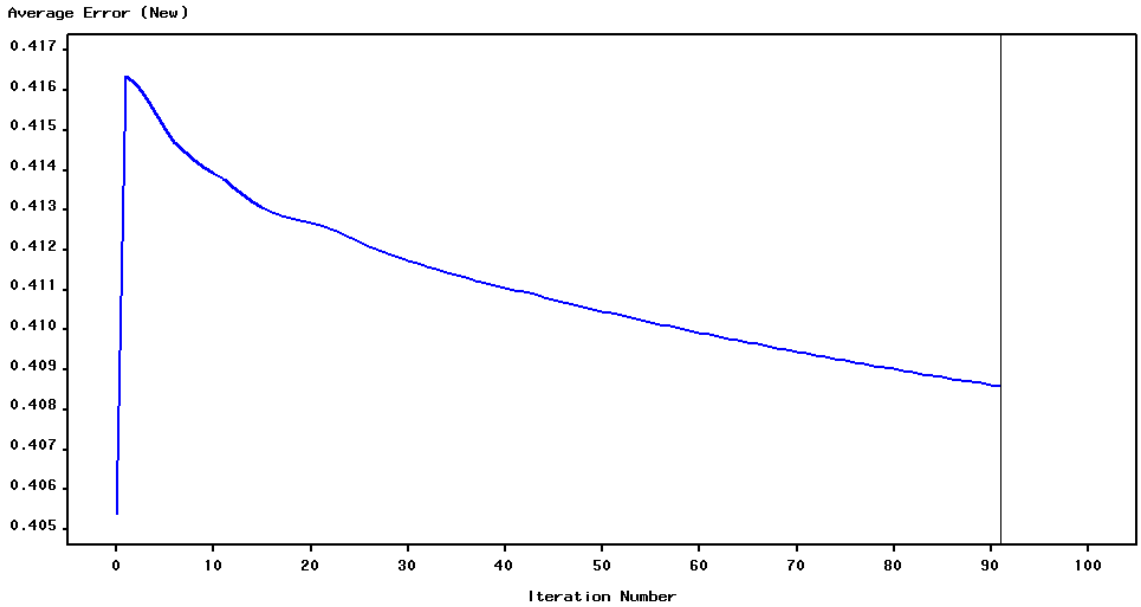


Gráfico 6.8 – Erro médio x Número de iterações

O critério de parada utilizado consiste na seguinte regra: se a alteração no erro médio for menor ou igual a 0,1% então o processo de aprendizagem está terminado.

Os pesos sinápticos ajustados estão na Tabela 6.4

Tabela 6.4 – Pesos sinápticos ajustados para a rede neural com um único neurônio

Variável	Peso sináptico ajustado
BIAS	-6.894567888
DUM1_V3	-0.3964491
DUM1_V4	-0.598896926
DUM1_V5	-0.539799513
DUM1_V1	-0.539848789
DUM1_V2	-0.631025886
DUM2_V3	0.355576234
DUM2_V4	0.183889836
DUM2_V5	0.400127302
DUM2_V1	0.125075365
DUM2_V2	0.122147483
DUM3_V3	0.661777177
DUM3_V4	0.434799371
DUM3_V5	0.509877891
DUM3_V1	0.259473363
DUM3_V2	0.375838188
DUM4_V4	0.569680948
DUM4_V1	0.491984359
DUM4_V2	0.564573859
DUM5_V1	0.693401122

Após o ajuste do modelo definiu-se um ponto de corte que maximizasse a sensibilidade e a especificidade do modelo. O Gráfico 6.9 indica o melhor ponto de corte para o modelo de redes neurais com um único neurônio ajustado, aproximadamente 0,1210.

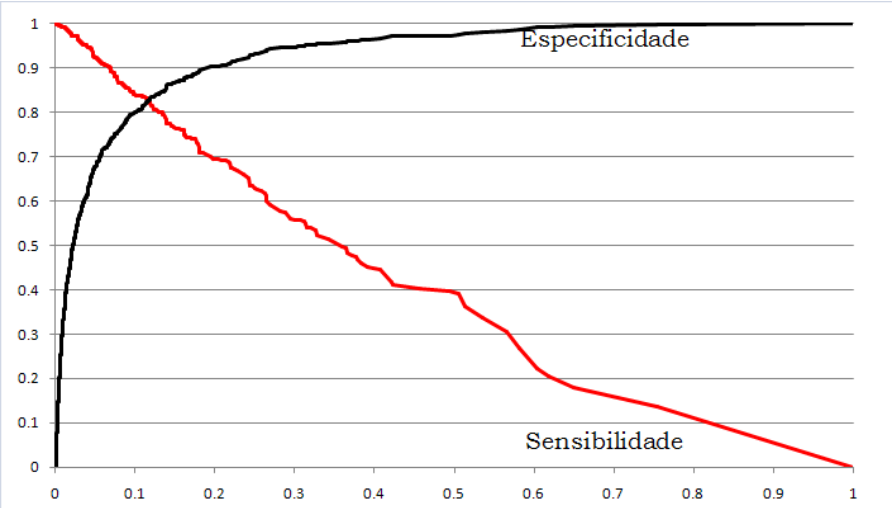


Gráfico 6.9 – Especificidade e Sensibilidade x Ponto de corte

A regra de decisão para a classificação dos clientes como bons ou maus pagadores é dada após a definição do ponto de corte. Esta forma de escolha otimiza a sensibilidade e a especificidade do modelo. As aplicações dos modelos de redes neurais ajustados foram realizadas no banco de dados de teste como pode ser visto na seção 6.2 de comparação dos modelos.

6.1.3.2. MLP com uma camada oculta

Assim como na rede com um único neurônio, várias redes MLP com uma camada oculta foram ajustadas através da técnica de aprendizagem *back-propagation* variando as taxas de aprendizado e a constante *momentum* como no caso anterior e também, neste caso, o número de neurônios na camada oculta. Através dos valores obtidos para o erro médio, a soma dos quadrados dos resíduos e o número de parâmetros no modelo, escolhe-se a rede com os melhores resultados.

A rede ajustada com uma camada intermediária que obteve os melhores resultados contém 15 neurônios na camada intermediária, como pode ser visto na Figura 6.2.

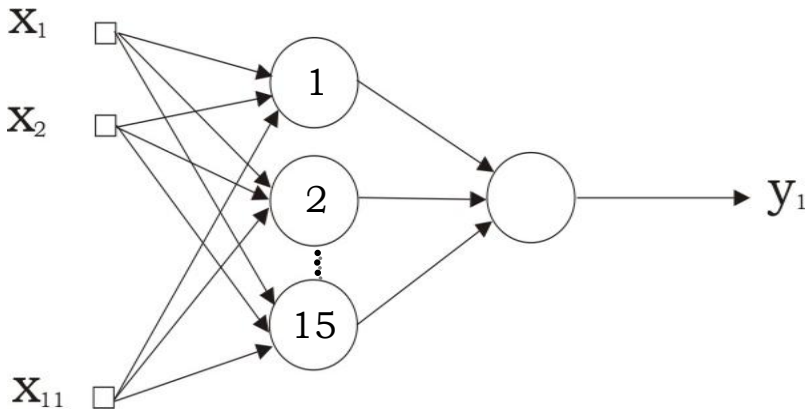


Figura 6.2 – Rede MLP com uma camada intermediária

Foram ajustados então um total de 316 pesos sinápticos ao longo das 110 épocas de treinamento necessárias até que o critério de parada fosse alcançado. O Gráfico 6.10 ilustra o erro médio ao longo das épocas de treinamento.

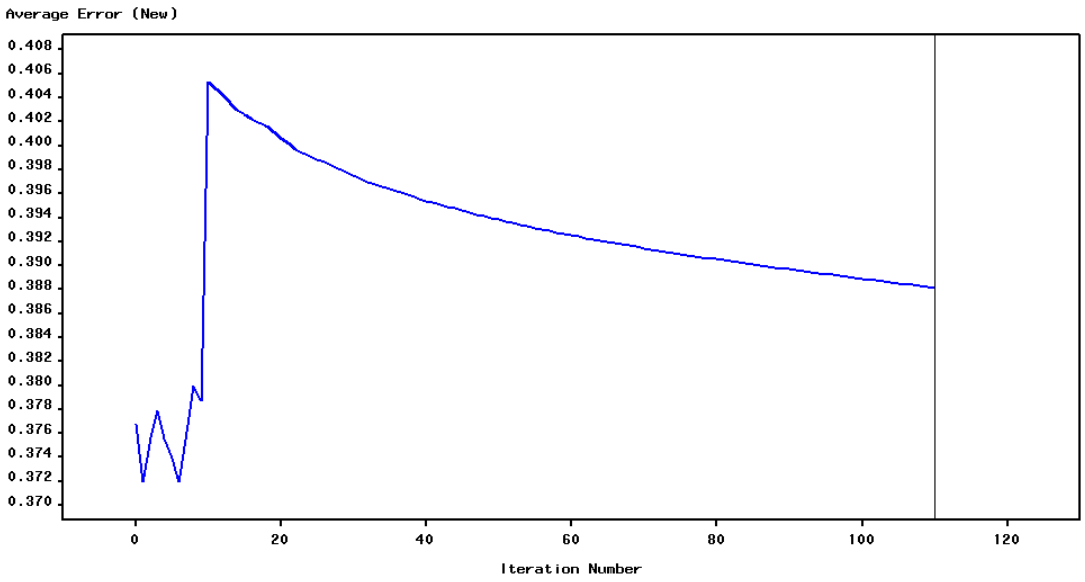


Gráfico 6.10 – Erro médio x Número de iterações

Os pesos sinápticos não serão aqui mostrados, por excesso de parâmetros e falta de interpretação para cada um deles.

A definição do ponto de corte, dada através da maximização da sensibilidade e da especificidade é ilustrada no Gráfico 6.11.

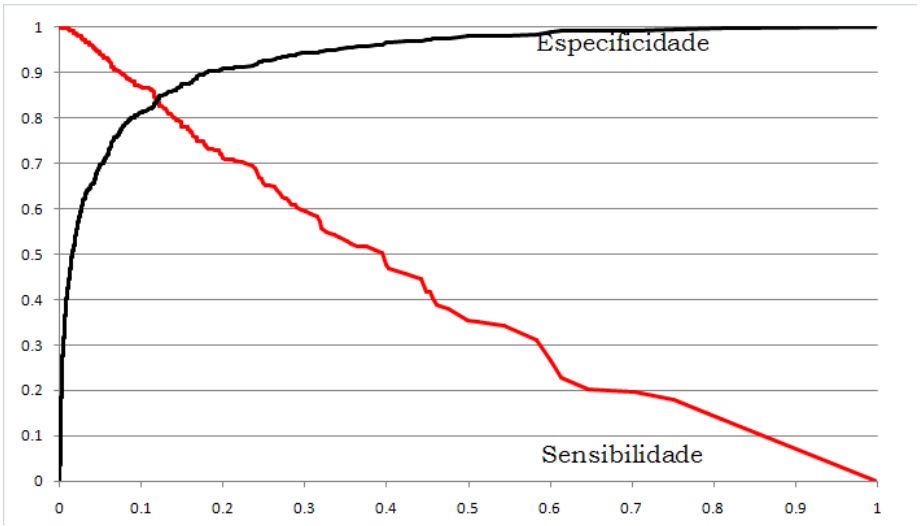


Gráfico 6.11 – Especificidade e Sensibilidade x Ponto de corte

Definido o ponto de corte, 0,1242, através do critério de maximização da especificidade e a sensibilidade constrói-se a regra de decisão para a classificação dos clientes como bons ou maus pagadores.

6.1.3.3 MLP com duas camadas ocultas

Assim como nas redes anteriores, várias redes MLP foram ajustadas, agora com duas camadas ocultas através do algoritmo de aprendizagem *back-propagation* variando também as taxas de aprendizado, a constante *momentum* e o número de neurônios nas duas camadas ocultas. Através dos valores obtidos para o erro médio, a soma dos quadrados dos resíduos e o número de parâmetros no modelo escolheu-se a rede com os melhores resultados.

A rede ajustada com duas camadas intermediárias que obteve os melhores resultados, contém 10 neurônios na primeira camada intermediária e 5 neurônios na segunda camada, como representado na figura 6.3.

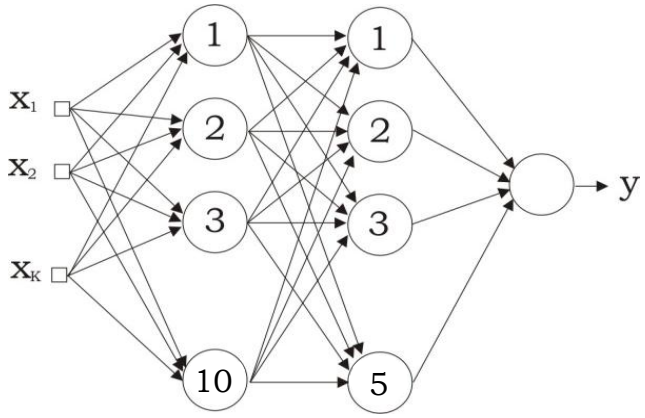


Figura 6.3 – Rede MLP com duas camadas intermediárias

Foram ajustados então um total de 261 pesos sinápticos ao longo de 110 épocas de treinamento, como pode ser visto no Gráfico 6.12.

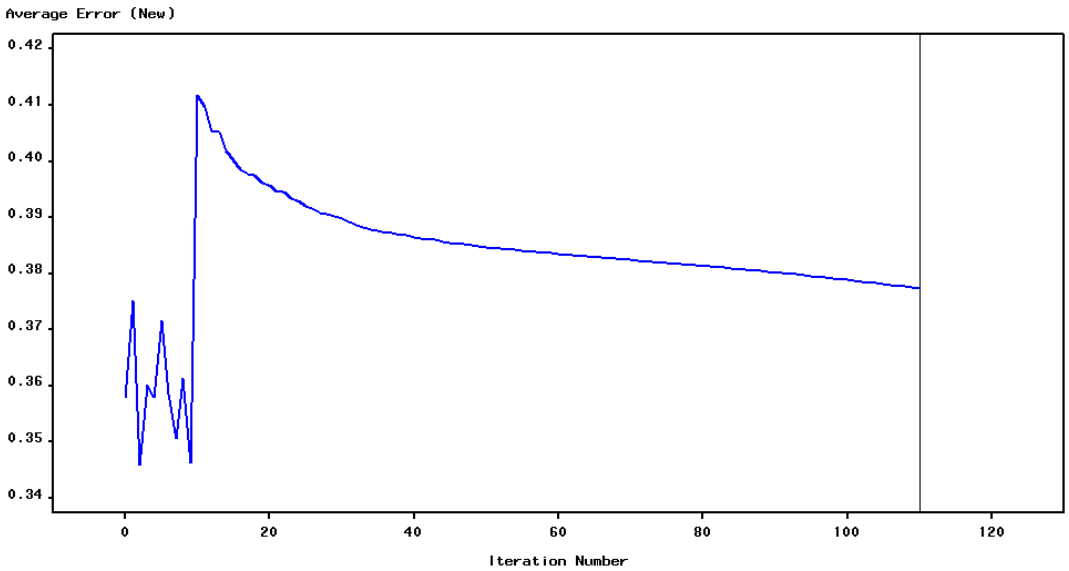


Gráfico 6.12 – Erro médio x Número de iterações

Assim como no ajuste anterior, os pesos sinápticos não serão aqui mostrados, por excesso de parâmetros e falta de interpretação para cada um deles.

A definição do ponto de corte, 0,1342, dada através da maximização da sensibilidade e da especificidade é ilustrada no Gráfico 6.13.

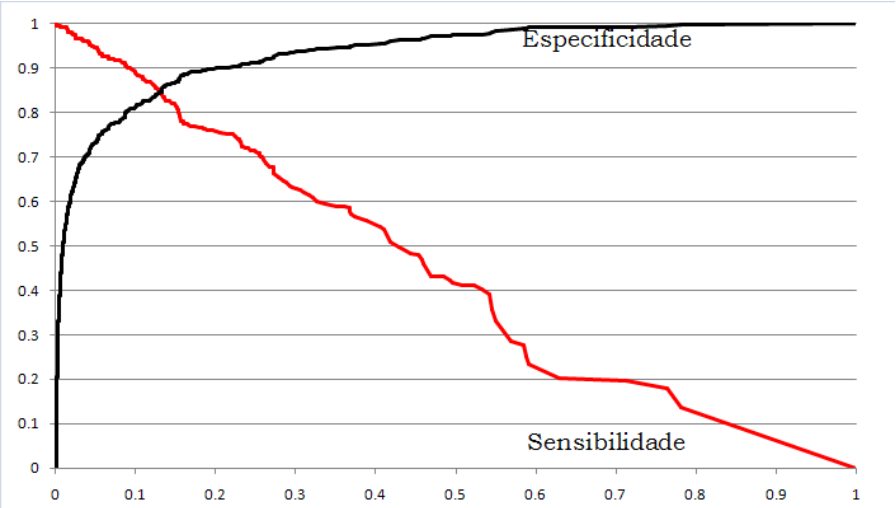


Gráfico 6.13 – Especificidade e Sensibilidade x Ponto de corte

Definido o ponto de corte que maximiza a especificidade e a sensibilidade constrói-se a regra de decisão para a classificação dos clientes como bons ou maus pagadores.

6.2 Comparação dos modelos

A comparação dos modelos ajustados foi realizada neste capítulo através das quatro análises vistas no Capítulo 5: avaliação da capacidade preditiva, curva ROC, estatística de Kolmogorov-Smirnov (KS) e capacidade de acerto do modelo.

6.2.1 Capacidade preditiva

Para o modelo de regressão logística clássico as estimativas obtidas através das re-amostragens *bootstrap* podem ser encontradas na Tabela 6.5.

Tabela 6.5 - Bootstrap (Regressão Logística Clássica)

decil	Risco				Taxa de inadimplência				Participação			
	obs	*Bootstrap*			obs	*Bootstrap*			obs	*Bootstrap*		
		est	LI	LS		est	LI	LS		est	LI	LS
0	0.73	0.73	0.69	0.78	89.72%	88.94%	84.44%	93.45%	299	297	274	320
1	0.44	0.44	0.36	0.52	68.84%	68.56%	60.38%	76.74%	229	229	205	254
2	0.24	0.24	0.16	0.31	55.81%	55.52%	47.95%	63.08%	186	186	162	209
3	0.13	0.13	0.06	0.20	39.54%	39.48%	32.52%	46.44%	132	132	111	153
4	0.06	0.06	0.00	0.13	22.79%	22.62%	16.26%	28.97%	76	76	56	95
5	0.03	0.03	-0.02	0.08	12.56%	12.12%	7.17%	17.07%	42	41	25	56
6	0.02	0.02	-0.03	0.06	6.05%	5.55%	1.34%	9.76%	20	19	5	32
7	0.01	0.01	-0.01	0.03	1.86%	1.66%	0.00%	3.65%	6	6	-	12
8	0.00	0.00	-0.02	0.02	2.79%	3.04%	0.98%	5.09%	9	10	3	17
9	0.00	0.00	-0.01	0.01	0.46%	0.04%	0.00%	1.28%	2	0	-	4

Pode-se observar na Tabela 6.5 que os valores estimados estiveram sempre muito próximos dos valores observados. Os casos que apresentaram

uma taxa muito baixa de inadimplência acabaram tendo um limite inferior do intervalo de confiança negativo. O Gráfico 6.14 permite a avaliação do modelo de acordo com a participação da taxa de inadimplência de cada decil na taxa geral de inadimplência, medida conhecida como *lift*. O resultado esperado para um modelo bem ajustado é uma curva estritamente decrescente monotônica, mostrando que quanto maior o decil, menor a taxa de inadimplência.

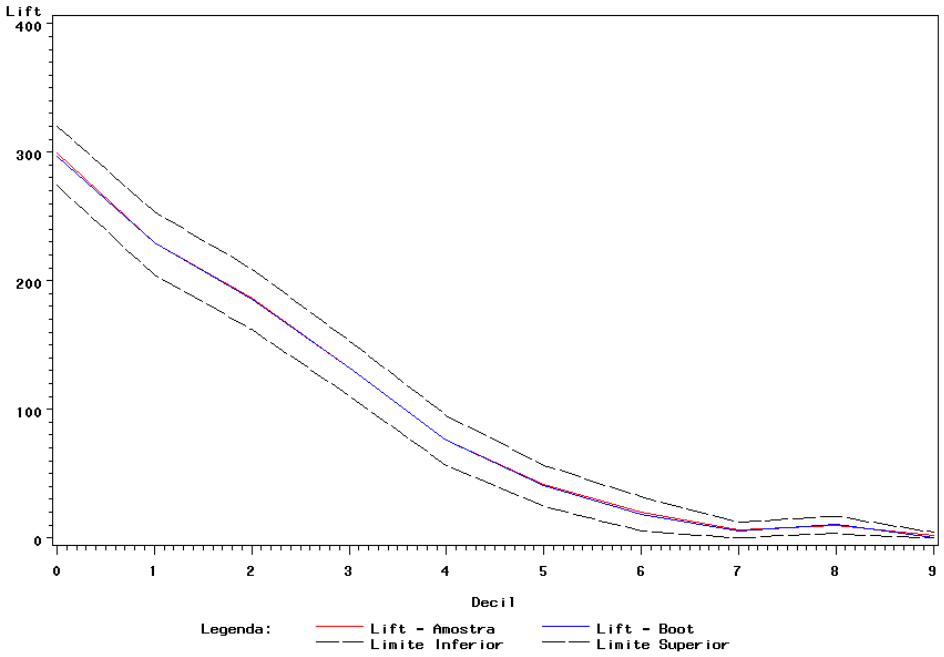


Gráfico 6.14 – Participação da taxa de inadimplência de cada decil na taxa geral de inadimplência

Pode-se observar no Gráfico 6.14 que os valores estimados se comportam de forma estritamente decrescente, indicando um modelo bem ajustado.

Aplicando o modelo ajustado na base de teste pode-se validar o modelo através da análise de decis de acordo com as estimativas obtidas pelo modelo para o risco de inadimplência e as taxas observadas. O Gráfico 6.15 permite a comparação das estimativas do modelo e a verdadeira resposta dos dados para cada decil.

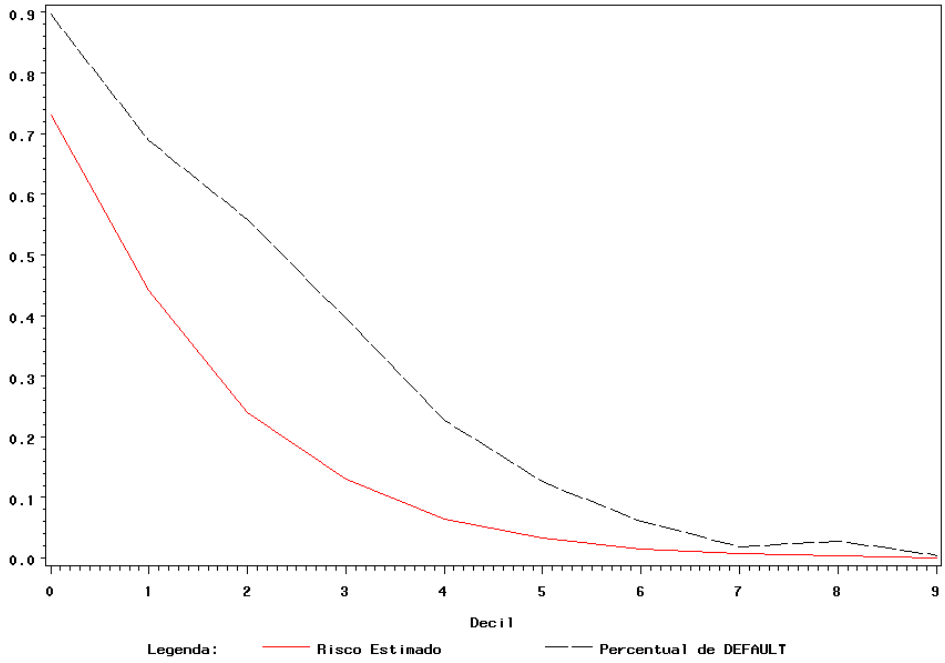


Gráfico 6.15 – Risco estimado x Taxa observada

O modelo está se aproximando de forma satisfatória da verdadeira taxa de inadimplência apenas para os últimos decis. Observe que o risco estimado pelo modelo logístico clássico está subestimando a taxa de inadimplência para os primeiros decis, o que não é muito apropriado em problemas de *Credit Score*. A precisão das estimativas obtidas fica muito boa para os clientes de menor risco, principalmente para os quatro últimos decis.

No modelo de regressão logística Bayesiano com priori não informativa, esta mesma metodologia foi utilizada na avaliação da capacidade preditiva do modelo. Pode-se observar na Tabela 6.6 que, assim como no modelo de regressão logística, os valores observados estão bastante próximos dos valores estimados.

Tabela 6.6 - Bootstrap (Regressão Logística Bayesiana)

decil	Risco				Taxa de inadimplência				Participação			
	obs	*Bootstrap*			obs	*Bootstrap*			obs	*Bootstrap*		
		est	LI	LS		est	LI	LS		est	LI	LS
0	0.73	0.73	0.69	0.77	89.72%	89.18%	85.41%	92.96%	299	297	276	319
1	0.44	0.44	0.37	0.52	69.30%	68.92%	61.38%	76.47%	231	230	208	252
2	0.24	0.24	0.17	0.30	55.35%	55.85%	49.52%	62.18%	184	186	168	204
3	0.13	0.13	0.05	0.20	39.54%	40.19%	32.79%	47.58%	132	134	112	157
4	0.06	0.06	0.00	0.13	22.33%	22.41%	15.66%	29.15%	74	75	54	96
5	0.03	0.03	-0.01	0.07	12.56%	12.22%	8.15%	16.29%	42	41	28	54
6	0.01	0.01	-0.03	0.06	6.05%	5.70%	1.09%	10.31%	20	19	4	34
7	0.01	0.01	-0.01	0.03	2.33%	2.48%	0.57%	4.39%	8	8	2	14
8	0.00	0.00	-0.02	0.03	2.79%	2.83%	0.52%	5.14%	9	9	2	17
9	0.00	0.00	-0.01	0.01	0.46%	0.10%	0.00%	1.51%	2	0	-	5

Os valores da participação para os decis, como podem ser também observados no Gráfico 6.16, indicam que o modelo possui boa estabilidade por apresentar uma curva estritamente decrescente monotônica. O comportamento apresentado é extremamente similar ao modelo clássico, como era esperado, dado que o modelo ajustado através da metodologia Bayesiana foi baseado em uma *priori* não informativa.

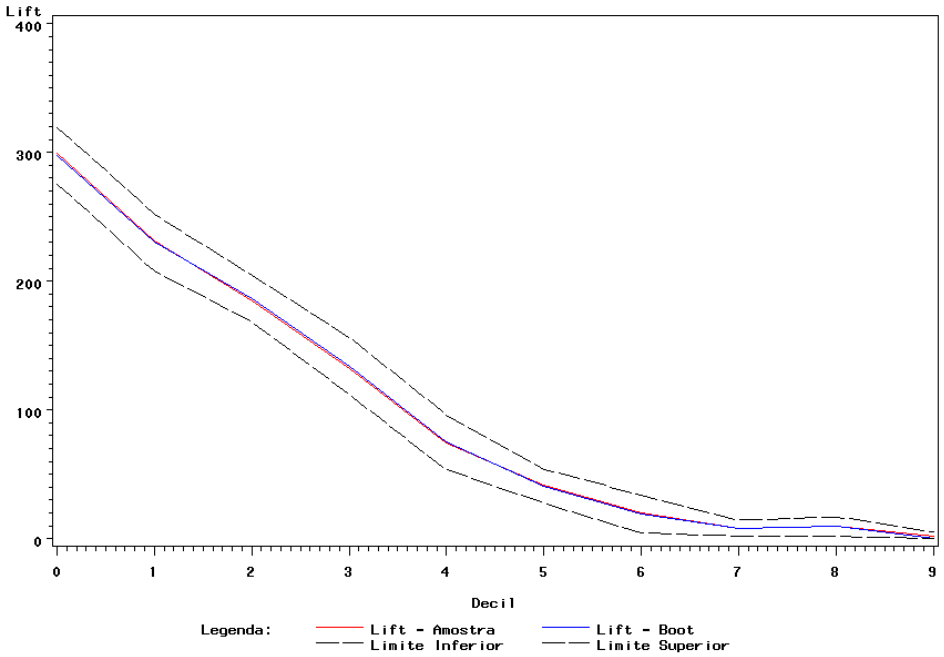


Gráfico 6.16 – Participação na taxa de inadimplência de cada decil na taxa geral de inadimplência

Após a aplicação do modelo ajustado na base de teste pode-se validar o modelo através da comparação entre as estimativas obtidas pelo modelo para o risco de inadimplência e as taxas observadas. O Gráfico 6.17 permite a comparação das estimativas do modelo e a verdadeira resposta dos dados para cada decil.

Como esperado o Gráfico 6.17 ficou muito semelhante ao Gráfico 6.15, já que não houve informação externa aos dados dada através da *priori* (*priori* não informativa). Na maioria dos decis o modelo subestimou as taxas de inadimplência, assim como no modelo clássico.

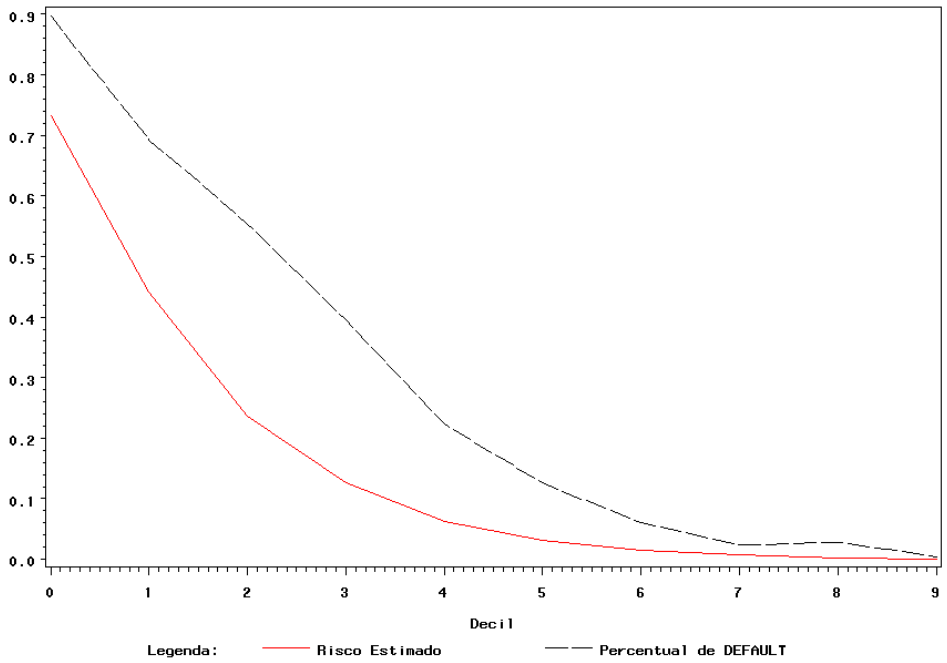


Gráfico 6.17 – Risco estimado x Taxa observada

A análise da capacidade preditiva do modelo de rede neural composto por um único neurônio teve as seguintes estimativas através das re-amostragens *bootstrap*, Tabela 6.7.

Tabela 6.7 - Bootstrap (Redes Neurais com um neurônio)

decil	Risco				Taxa de inadimplência				Participação			
	obs	*Bootstrap*			obs	*Bootstrap*			obs	*Bootstrap*		
		est	LI	LS		est	LI	LS		est	LI	LS
0	0.71	0.71	0.67	0.75	89.72%	88.88%	84.96%	92.80%	299	299	279	319
1	0.43	0.44	0.38	0.49	67.91%	67.84%	62.31%	73.37%	226	228	206	250
2	0.23	0.23	0.17	0.29	55.35%	55.08%	49.38%	60.78%	184	185	166	205
3	0.13	0.13	0.05	0.22	40.00%	38.46%	30.29%	46.63%	133	130	104	155
4	0.07	0.07	0.01	0.13	21.40%	21.41%	15.21%	27.61%	71	72	52	92
5	0.04	0.04	-0.01	0.09	12.09%	12.28%	7.00%	17.57%	40	41	24	58
6	0.02	0.02	-0.03	0.07	8.84%	8.75%	3.38%	14.13%	29	30	12	47
7	0.01	0.01	-0.01	0.03	2.33%	2.50%	0.52%	4.47%	8	8	2	15
8	0.01	0.01	-0.01	0.03	1.86%	1.58%	0.00%	3.55%	6	5	-	12
9	0.00	0.00	-0.01	0.01	0.93%	1.34%	0.33%	2.36%	3	4	1	8

As verdadeiras taxas de inadimplência estão todas dentro dos intervalos de confiança para o risco em cada um dos dez decis do escore. Na Tabela 6.7 e no Gráfico 6.18 pode-se observar também que as taxas de inadimplência e a participação dos escores formam uma curva estritamente decrescente, indicando também um bom ajuste.

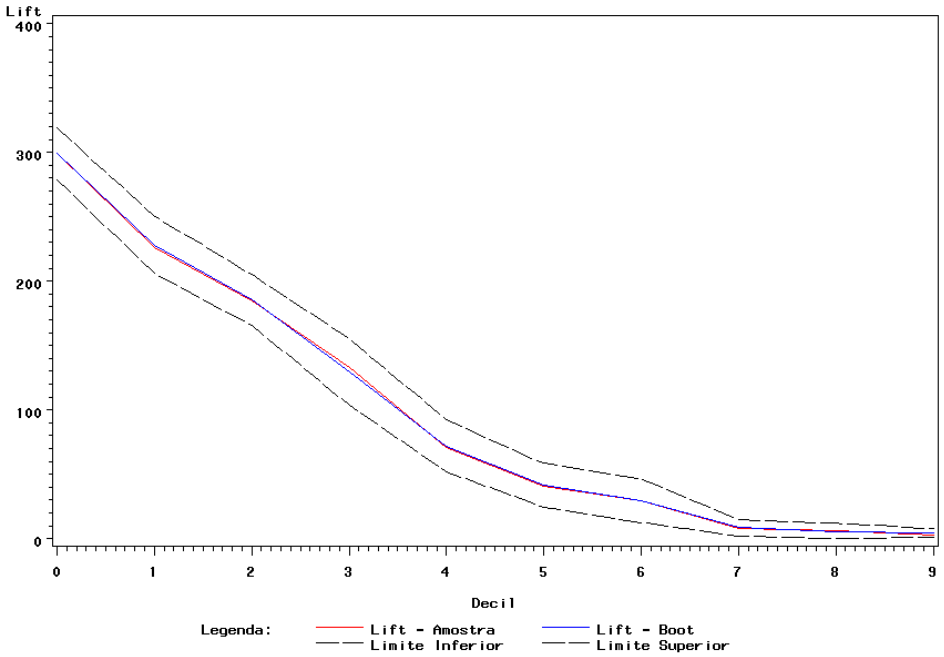


Gráfico 6.18 – Participação na taxa de inadimplência na taxa geral de inadimplência

O Gráfico 6.19 mostra que o modelo de rede neural com um único neurônio também subestima o modelo para a maioria dos decis. De forma geral o modelo se aproximou mais da verdadeira taxa de inadimplência.

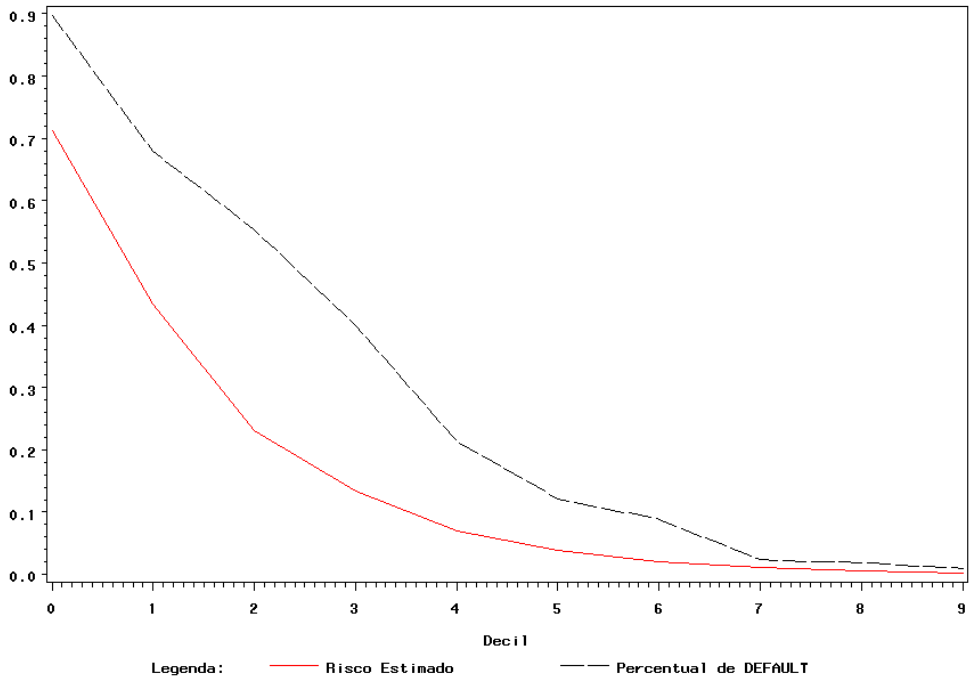


Gráfico 6.19 – Risco estimado x Taxa observada

Com o modelo de rede neural *multilayer perceptron* com uma camada intermediária as seguintes estimativas de re-amostragens obtidas foram encontradas, Tabela 6.8.

Tabela 6.8 - Bootstrap (Redes Neural MLP com uma camada intermediária)

decil	Risco				Taxa de inadimplência				Participação			
	obs	*Bootstrap*			obs	*Bootstrap*			obs	*Bootstrap*		
		Est	LI	LS		est	LI	LS		est	LI	LS
0	0.74	0.75	0.71	0.78	89.72%	89.52%	85.68%	93.36%	299	296	276	315
1	0.43	0.43	0.38	0.49	68.37%	69.22%	63.53%	74.92%	228	229	206	251
2	0.25	0.25	0.17	0.32	53.95%	53.85%	46.26%	61.45%	180	178	153	203
3	0.13	0.13	0.06	0.21	39.07%	39.65%	32.15%	47.14%	130	131	109	153
4	0.06	0.06	0.01	0.12	21.86%	21.69%	16.14%	27.24%	73	72	54	90
5	0.03	0.03	-0.02	0.08	14.42%	14.79%	9.49%	20.08%	48	49	32	66
6	0.02	0.02	-0.02	0.05	7.44%	7.95%	4.01%	11.88%	25	26	13	40
7	0.01	0.01	-0.01	0.03	2.33%	2.46%	0.40%	4.52%	8	8	1	15
8	0.00	0.00	-0.02	0.02	2.79%	3.10%	0.90%	5.31%	9	10	3	18
9	0.00	0.00	-0.01	0.01	0.46%	0.16%	0.00%	1.23%	2	1	-	4

Observe que as taxas de inadimplência observadas pertencem ao intervalo de confiança para o risco em cada decil e a participação obteve uma curva estritamente decrescente, indicando uma capacidade preditiva muito próxima às comparadas aos modelos anteriores.

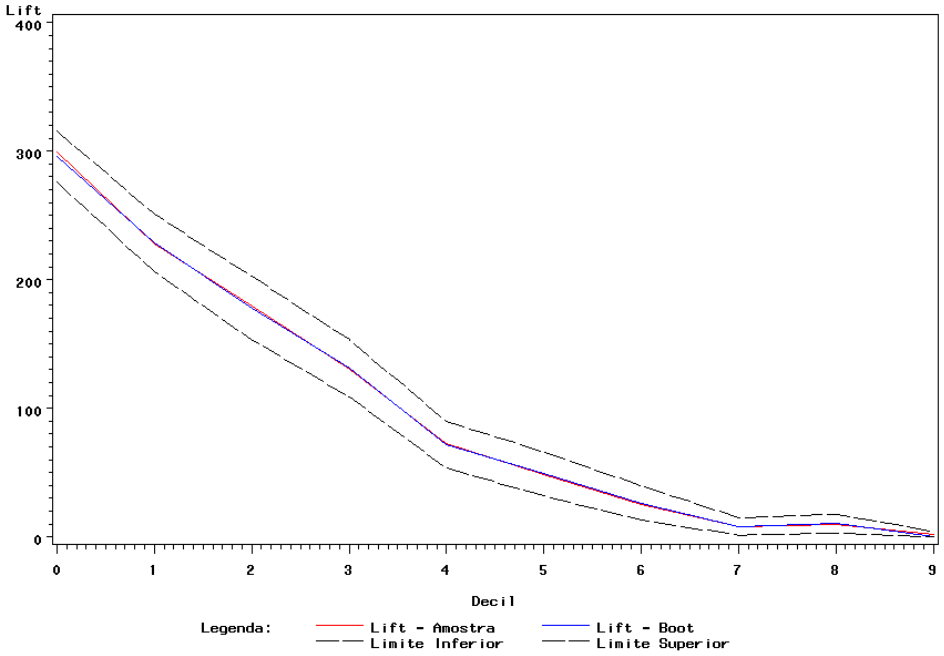


Gráfico 6.20 – Participação na taxa de inadimplência de cada decil na taxa geral de inadimplência

Este modelo é até aqui o que mais se aproximou da verdadeira taxa de inadimplência, como pode ser observado no Gráfico 6.21.

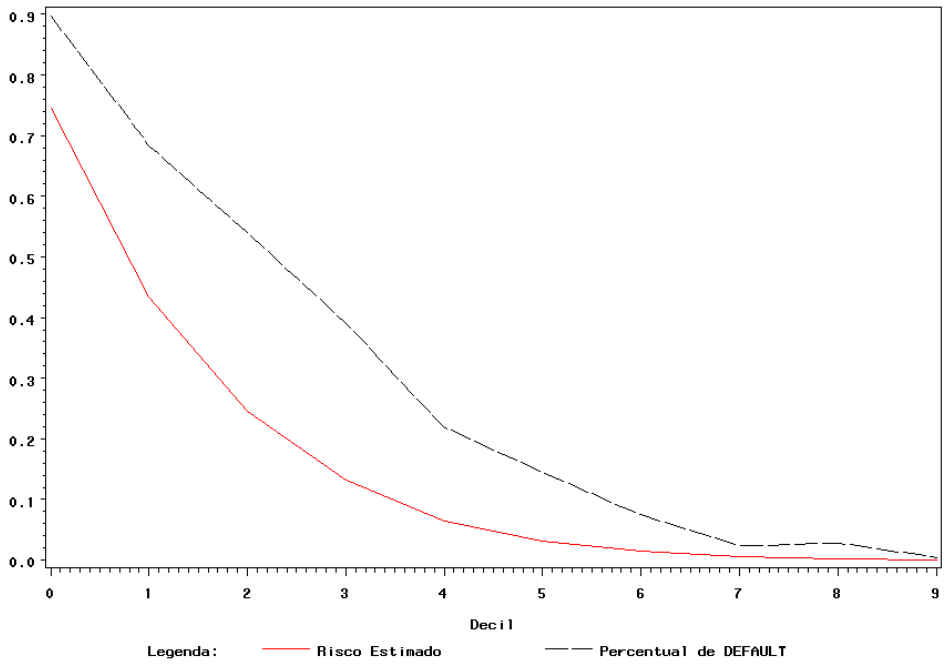


Gráfico 6.21 – Risco estimado x Taxa observada

Para o modelo de redes neurais *multilayer perceptron* com duas camadas intermediárias as estimativas obtidas via *bootstrap* para o risco, a taxa de inadimplência e a participação da taxa de inadimplência de cada decil na taxa geral de inadimplência são dadas na Tabela 6.9.

Tabela 6.9 - Bootstrap (Redes Neural MLP com duas camadas intermediárias)

decil	Risco				Taxa de inadimplência				Participação			
	obs	*Bootstrap*			obs	*Bootstrap*			obs	*Bootstrap*		
		est	LI	LS		est	LI	LS		est	LI	LS
0	0.71	0.72	0.68	0.75	89.72%	89.48%	86.41%	92.54%	299	297	275	319
1	0.47	0.47	0.39	0.55	68.37%	69.08%	61.05%	77.11%	228	230	203	256
2	0.27	0.28	0.20	0.35	51.16%	51.23%	43.93%	58.54%	171	170	147	193
3	0.13	0.13	0.06	0.20	37.21%	37.18%	30.16%	44.20%	124	124	101	146
4	0.06	0.06	-0.01	0.12	20.47%	20.19%	13.98%	26.40%	68	67	47	87
5	0.02	0.02	-0.04	0.08	18.14%	17.93%	11.94%	23.93%	60	60	40	79
6	0.01	0.01	-0.03	0.05	7.91%	7.54%	3.28%	11.80%	26	25	10	40
7	0.00	0.00	-0.02	0.03	3.72%	4.39%	1.97%	6.80%	12	15	7	22
8	0.00	0.00	-0.02	0.02	2.33%	2.07%	0.00%	4.14%	8	7	-	14
9	0.00	0.00	-0.02	0.02	1.39%	1.35%	0.00%	3.00%	5	4	-	10

O Gráfico 6.22 indica um bom ajuste do modelo, por apresentar uma curva estritamente decrescente. O Gráfico 6.23 mostra que o modelo subestima a verdadeira taxa de inadimplência nos primeiros decis.

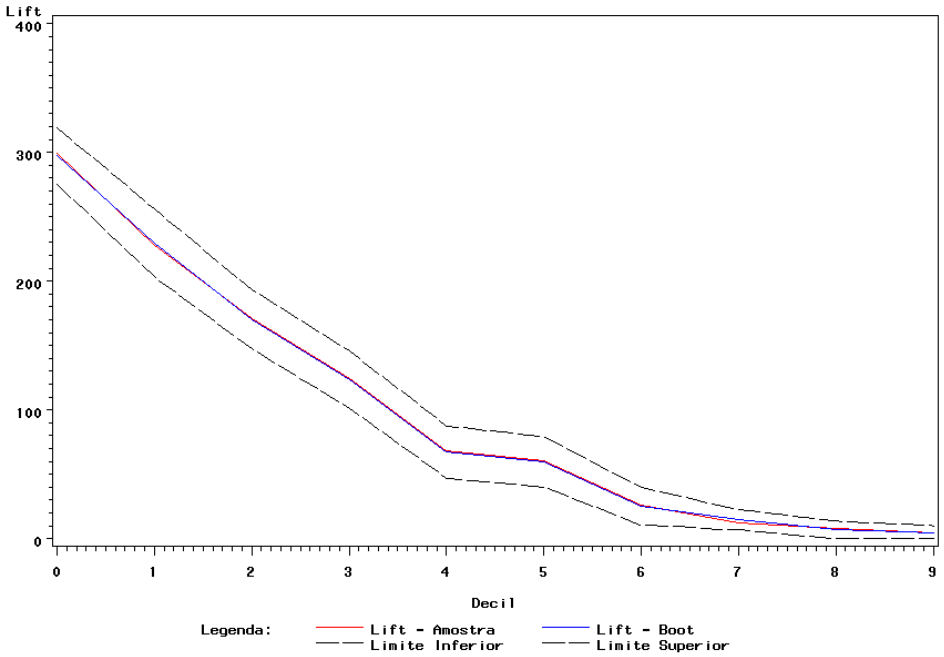


Gráfico 6.22 – Participação da taxa de inadimplência na taxa geral de inadimplência

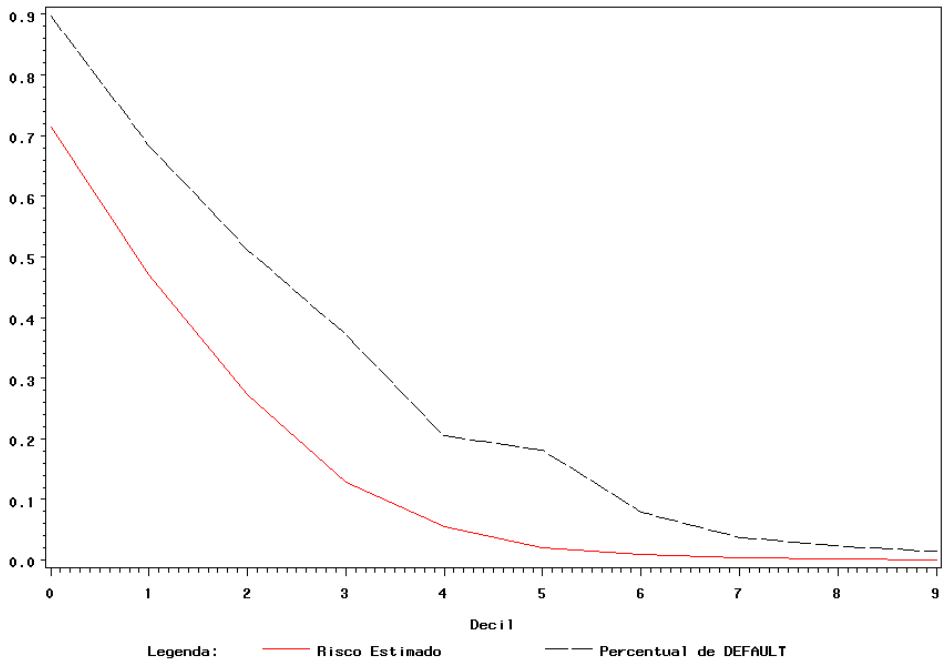


Gráfico 6.23 – Risco estimado x Taxa observada

6.2.2 Curva ROC

A curva ROC pode ser utilizada na escolha do melhor ponto de corte por envolver as medidas de sensibilidade e especificidade. No entanto, a estratégia de escolha do ponto de corte adotada neste trabalho foi escolher o ponto que maximizasse a sensibilidade e a especificidade através da intersecção entre as curvas de sensibilidade e especificidade construídas para cada ponto de corte. A curva ROC será utilizada aqui no intuito de comparar os diferentes modelos ajustados, a sensibilidade é representada pelo eixo Y e o valor de (1-especificidade) é representado pelo eixo X.

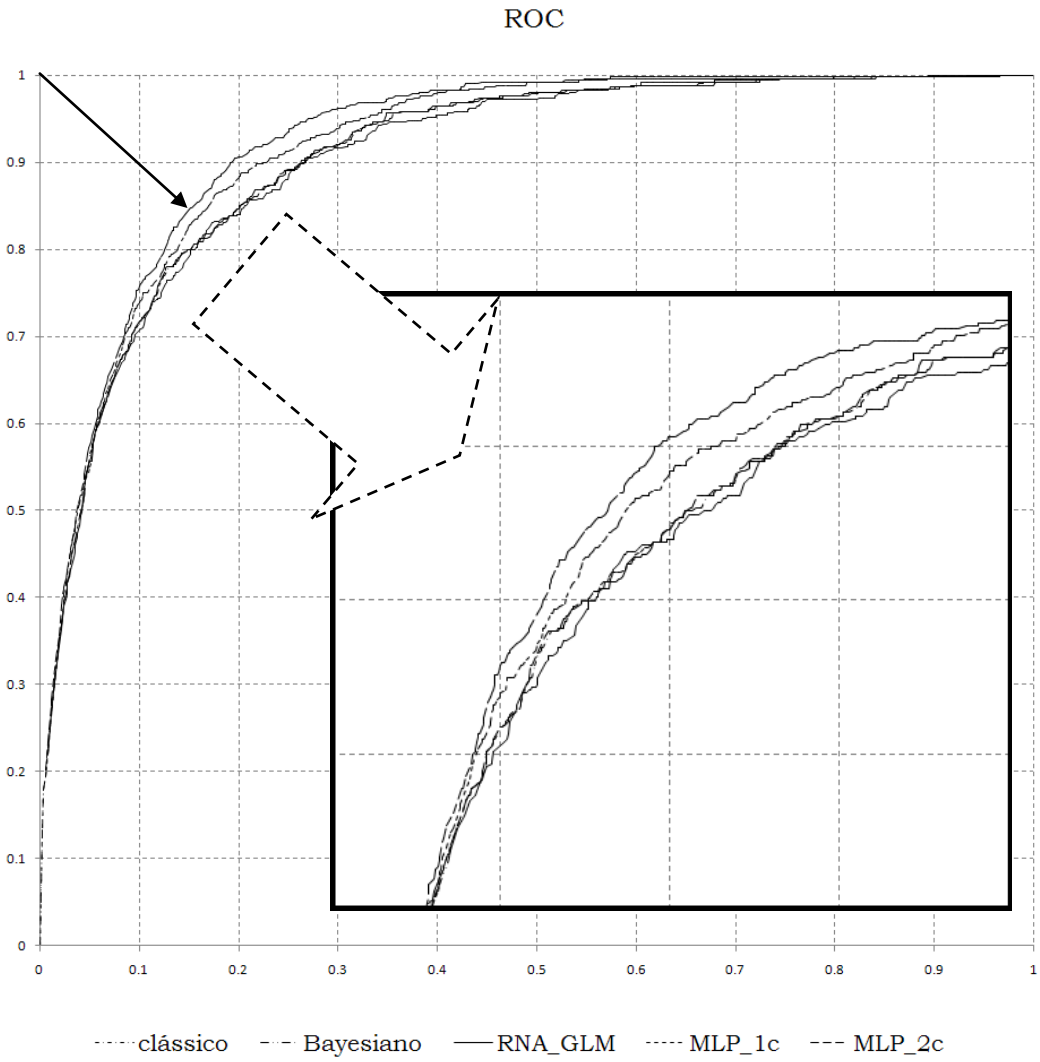


Gráfico 6.24 – Comparação dos modelos ajustados através da curva ROC

Pode-se observar no Gráfico 6.24 que todos os modelos tiveram um comportamento muito semelhante com relação à sensibilidade e à especificidade. Com leve vantagem para o modelo de Redes Neurais com duas camadas intermediárias.

6.2.3 Estatística de Kolmogorov-Smirnov (KS) e capacidade de acerto dos modelos

Outra forma de comparação de modelos muito utilizada é a medida da estatística de Kolmogorov-Smirnov (KS). A Tabela 6.10 apresenta a estatística de KS juntamente com as medidas de capacidade de acerto total, acerto de clientes inadimplentes e acerto dos clientes adimplentes para cada um dos modelos ajustados.

Tabela 6.10 – Estatística KS e Capacidade de Acerto dos modelos nos bancos de teste (Total, Maus e Bons)

%	Clássico	Bayesiano	RNA_GLM	MLP_1c	MLP_2c
KS	72,59	72,59	71,11	69,93	73,33
CAT	82,67	82,00	90,00	84,70	85,30
CAM	80,00	80,00	80,00	73,30	73,30
CAB	82,96	82,22	91,11	85,90	86,70

A estatística KS não diferiu significativamente entre os diferentes modelos ajustados, a capacidade de acertos total, capacidade de acerto dos bons clientes e dos maus clientes indicam o modelo de redes neurais com um único neurônio como o que tem maior capacidade de acerto. Na análise da curva ROC o modelo que mais se destacou foi o modelo de redes neurais com duas camadas intermediárias, assim como na análise da capacidade preditiva.

Capítulo 7

Aplicação em dados reais

Após a aplicação das técnicas estudadas em um banco de dados artificial aplicou-se as mesmas em um conjunto de dados real para um problema de *Credit Score* de uma instituição financeira que atua no mercado varejista brasileiro há mais de vinte anos. As técnicas de Regressão Logística Clássica e Bayesiana e de Redes Neurais Artificiais foram comparadas de acordo com a estatística de Kolmogorov-Smirnov (KS), curva ROC e a capacidade preditiva.

O banco de foi dividido em duas partes, uma parte utilizada para o ajuste do modelo e uma segunda parte utilizada para a validação, contendo respectivamente 70% e 30% da base de dados. Análises de correlação, univariadas e bivariadas foram realizadas com o objetivo de selecionar variáveis significativas e não correlacionadas como candidatas ao modelo. As variáveis foram categorizadas de acordo com a razão de chances na análise bivariada e transformadas em variáveis dummy.

Uma descrição do problema pode ser vista na seção 7.1 e as aplicações das técnicas de regressão logística clássica, Bayesiana e os ajustes de redes

neurais podem ser vistos na seção 7.2. A seção 7.3 realiza uma comparação dos modelos ajustados através da capacidade preditiva, curva ROC, KS e de uma análise da capacidade de acerto dos modelos, além de uma avaliação do desempenho da combinação dos modelos trabalhando conjuntamente através de um comitê.

7.1 Descrição do Problema

A mensuração da confiabilidade de uma pessoa reembolsar o dinheiro que é devido é o intuito de uma análise de *Credit Score*. Tanto na aplicação da técnica clássica quanto a Bayesiana utiliza-se como base informações de crédito, obtidas geralmente de companhias que fornecem informações de crédito sobre empréstimos individuais.

O banco de dados utilizado neste trabalho é constituído de informações de uma instituição financeira onde os clientes adquiriram um produto de crédito. A partir deste conjunto de dados esta instituição tem como objetivo medir o risco de inadimplência de possíveis clientes que possam adquirir o produto. As variáveis disponíveis na base de dados foram coletadas no momento do pedido de empréstimo e uma variável indicadora do desempenho de pagamento nos 12 meses seguintes ao da concessão do crédito. O banco possui 7320 clientes sendo que em aproximadamente 30% dos casos foi observada a inadimplência. O banco de dados de ajuste possui 5124 clientes e o banco de validação possui 2196, ambos com aproximadamente a mesma proporção de inadimplentes (Abreu, 2005). As variáveis estão descritas na Tabela 7.1.

Tabela 7.1 – Descrição das variáveis

Variável	Descrição
TPCLIENT	Indica se é cliente há mais de um ano (dicotômica)
TPEMPREG	Tempo que o cliente está no atual emprego (em meses)
SEXO	Sexo
ESTCIVIL	Estado civil
SITRESID	Condição da residência: própria ou alugada
LIMITE	Valor do crédito concedido (em reais)
TEMPORES	Tempo de residência na casa atual (em anos)
CEP	Os dois primeiros dígitos do CEP
TEMPO	Tempo entre o pedido de empréstimo até a indadimplência
AGE_02	Idade do cliente em anos completos no dia 31/12/2002
MAU	Variável resposta: 0 – bom pagador, 1 – mau pagador

A variável tempo não foi disponibilizada para o ajuste, pois o tempo até a inadimplência não auxilia na predição da mesma. Esta variável pode ser utilizada em um ajuste de um modelo de sobrevivência que prevê o tempo esperado até a ocorrência do evento, diferentemente da regressão logística que tem o objetivo de estimar a probabilidade de ocorrência do evento.

Nenhuma das variáveis foi considerada correlacionada sendo então disponibilizado para o ajuste do modelo um total de nove variáveis.

Estas informações são de clientes para os quais já foram observados os desempenhos de pagamento do crédito adquirido e servirão para a construção dos modelos preditivos a partir das metodologias estudadas, Regressão Logística Clássica, Regressão Logística Bayesiana e modelos de Redes Neurais para diferentes arquiteturas. Tais modelos serão aplicados em futuros potenciais clientes, permitindo que esses possam ser ordenados segundo uma

probabilidade de inadimplência e a partir da qual as políticas de crédito das instituições possam ser definidas.

7.2 Aplicação no banco de dados de ajuste

Esta seção apresenta os resultados obtidos a partir da amostra de ajuste sendo utilizadas as metodologias de Regressão Logística Clássica, Bayesiana e de Redes Neurais.

Após a pré-seleção e categorização das variáveis, realizada através das análises exploratórias citadas na seção anterior, realizou-se o ajuste de regressão logística clássico de acordo com a estratégia adotada para os dados simulados do Capítulo anterior. Os ajustes realizados pela abordagem Bayesiana e de Redes Neurais utilizaram as mesmas variáveis do modelo clássico.

7.2.1 Ajuste do modelo de regressão logística clássico

As análises univariadas, bivariadas e de correlação não excluíram nenhuma variável, como dito anteriormente. Conforme a estratégia de construção de modelos adotada ajustou-se um modelo de regressão logística univariado para cada uma das variáveis disponibilizadas e observou-se a significância das mesmas de acordo com o teste Wald. Os ajustes obtidos podem ser vistos na Tabela 7.2.

Tabela 7.2 – Análise dos estimadores de máxima verossimilhança para os ajustes univariados.

Ajuste	Parâmetro		GL	Estimativa	Erro padrão	Qui-Quad Wald	Pr > Qui-Quad
1	Intercepto		1	-1.0285	0.0398	667.8239	<.0001
	CAT_TPCLIENT	C1	1	0.4828	0.0623	59.9735	<.0001
2	Intercepto		1	-0.9742	0.0413	557.3342	<.0001
	CAT_SEXO	C1	1	0.3017	0.0613	24.1993	<.0001
3	Intercepto		1	-1.0096	0.0391	666.3301	<.0001
	CAT_ESTCIVIL	C1	1	0.4555	0.0628	52.5427	<.0001
4	Intercepto		1	-0.8848	0.0322	755.3547	<.0001
	CAT_SITRESID	C1	1	0.3917	0.1054	13.8203	0.0002
	CAT_SITRESID	C2	1	1.0462	0.3314	9.9647	0.0016
5	Intercepto		1	-0.7988	0.0372	459.9555	<.0001
	CAT_CEP	C1	1	-0.4130	0.0785	27.6964	<.0001
	CAT_CEP	C2	1	0.3788	0.0934	16.4599	<.0001
6	Intercepto		1	-0.8381	0.0432	375.6096	<.0001
	CAT_TPEMPREG	C1	1	0.5697	0.0749	57.7907	<.0001
	CAT_TPEMPREG	C2	1	-0.5234	0.0773	45.8374	<.0001
7	Intercept		1	-0.9994	0.0388	665.1616	<.0001
	CAT_LIMITE	C1	1	0.4399	0.0632	48.5129	<.0001
8	Intercept		1	-0.8216	0.0681	145.3559	<.0001
	CAT_TEMPORES	C1	1	0.1500	0.0863	3.0232	0.0821
	CAT_TEMPORES	C2	1	-0.1430	0.0815	3.0841	0.0791
9	Intercept		1	-0.9915	0.0580	291.8484	<.0001
	CAT_AGE_02	C1	1	0.9033	0.1008	80.3039	<.0001
	CAT_AGE_02	C2	1	0.7606	0.1055	51.9612	<.0001
	CAT_AGE_02	C3	1	0.3430	0.0883	15.0886	0.0001
	CAT_AGE_02	C4	1	-0.4896	0.0881	30.8583	<.0001

Todas as variáveis se mostraram significativas com exceção da variável correspondente ao tempo de residência (CAT_TEMPORES) do cliente na casa atual. Indicando assim que esta variável não contribui significativamente na explicação da variável resposta.

Foram aplicados os mesmos procedimentos de seleção de variáveis, como *stepwise*, *forward* e *backward*. Aplicando qualquer um dos procedimentos, disponibilizando todas as variáveis que foram significativas no ajuste univariado, obteve-se a seguinte seleção de variáveis: AGE_02, TPCLIENT,

TPEMPREG, CEP, LIMITE, SEXO E SITRES. No entanto a variável SITRES possui p-valor maior que 5% para ambas as categorias e foi a última variável a entrar no modelo segundo os métodos *stepwise* e *forward*, mesmo sendo significante no nível de 5% na análise dos efeitos pode-se retirar a mesma do modelo final de acordo com o teste da razão de verossimilhança. Os estimadores dos coeficientes do modelo final, com as demais variáveis, são dados na Tabela 7.3.

Tabela 7.3 – Análise dos estimadores de máxima verossimilhança

Parâmetro		gl	Estimador	Erro Padrão	Qui-Quad Wald	Pr > Qui-Quad
Intercepto		1	-1.1487	0.0816	198.2305	<.0001
CAT_TPCLIENT	C1	1	0.2701	0.0688	15.4060	<.0001
CAT_SEXO	C1	1	0.1507	0.0643	5.4896	0.0191
CAT_CEP	C1	1	-0.2667	0.0814	10.7278	0.0011
CAT_CEP	C2	1	0.2928	0.0968	9.1376	0.0025
CAT_TPEMPREG	C1	1	0.2357	0.0809	8.4967	0.0036
CAT_TPEMPREG	C2	1	-0.2745	0.0826	11.0473	0.0009
CAT_LIMITE	C1	1	0.2302	0.0684	11.3314	0.0008
CAT_AGE_02	C1	1	0.6152	0.1077	32.6151	<.0001
CAT_AGE_02	C2	1	0.5868	0.1096	28.6537	<.0001
CAT_AGE_02	C3	1	0.2453	0.0908	7.2972	0.0069
CAT_AGE_02	C4	1	-0.4074	0.0898	20.5951	<.0001

Através do teste de razão de verossimilhança a retirada da variável não foi significativa estatisticamente e não houve inversão do sinal dos coeficientes para nenhuma das variáveis entre o ajuste univariado e o multivariado e nem para a categorização de acordo com a razão de chances.

Após o ajuste do modelo é necessário definir um ponto de corte que maximize a sensibilidade e a especificidade do modelo. O Gráfico 7.1 indica o melhor ponto de corte para o modelo de regressão logística ajustado, aproximadamente 0.3.

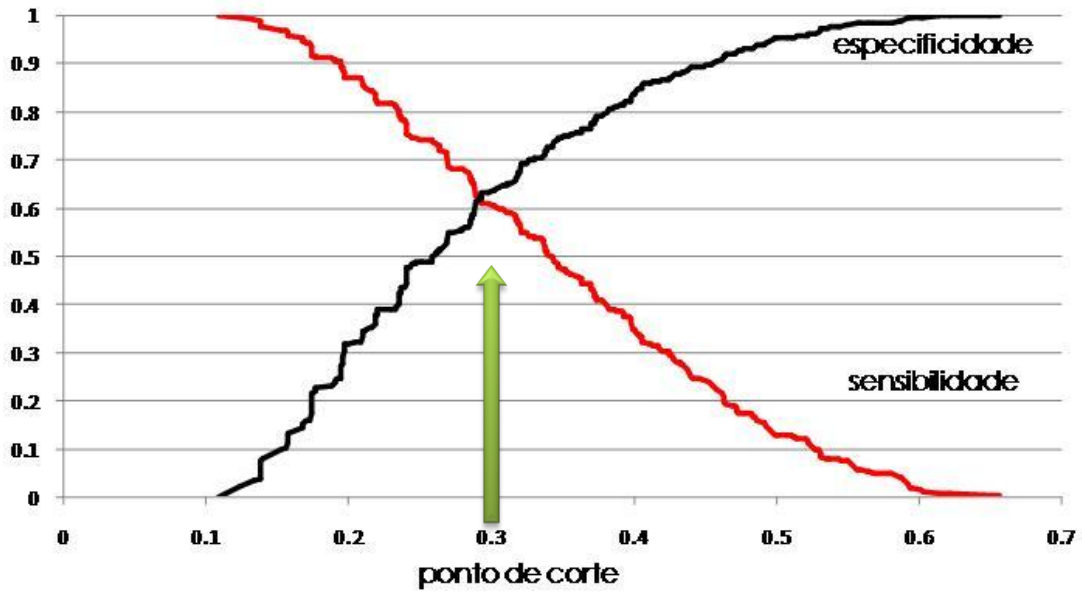


Gráfico 7.1 – Especificidade e Sensibilidade x Ponto de corte

Após a definição do ponto de corte escolhe-se a regra de decisão para a classificação dos clientes como bons ou maus pagadores. Esta forma de escolha otimiza a sensibilidade e a especificidade do modelo. A aplicação do modelo clássico no banco de dados de teste pode ser vista na seção 7.3 de comparação dos modelos.

7.2.2 Ajuste do modelo de regressão logística Bayesiano

As mesmas variáveis do modelo final clássico foram utilizadas no ajuste Bayesiano. Como as variáveis foram todas categorizadas na análise inicial as

seis variáveis que permaneceram no ajuste final da análise clássica foram transformadas em variáveis dummy, somando ao todo doze parâmetros de interesse. Neste trabalho foi ajustado um modelo com priori não informativa, onde se espera que os parâmetros ajustados sejam próximos dos parâmetros estimados pela metodologia clássica, ou seja, apenas com a informação contida na função de verossimilhança.

O modelos foi ajustado no software WinBUGS que utiliza o algoritmo de Metropolis para a obtenção da distribuição *a posteriori*. As variáveis foram colocadas no programa na seguinte ordem: INTERCEPTO (beta0), D_TPCLIENT (beta1), D_SEXO (beta2), D_LIMITE (beta3), D1_CEP (beta4), D2_CEP (beta5), D1_TPEMPREG (beta6), D2_TPEMPREG (beta7), D1_AGE (beta8), D2_AGE (beta9), D3_AGE (beta10) e D4_AGE (beta11).

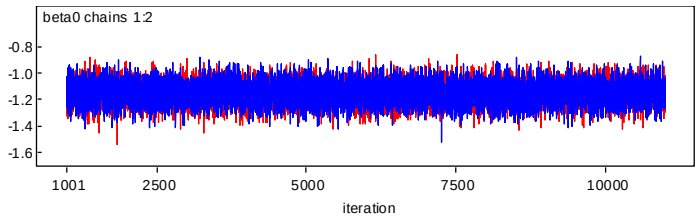
A *priori* não informativa dada para os parâmetros foi imprópria com distribuição uniforme. Utilizou-se um *burn-in* de tamanho 1.000 e uma amostra com 10.000 gerações, ambos os casos com saltos de tamanho igual a dez. Duas cadeias foram geradas e tinham como valores iniciais os parâmetros estimadores de máxima verossimilhança e valores gerados aleatoriamente pelo programa.

Antes de observar as distribuições *a posteriori* marginais dos parâmetros são necessárias algumas verificações de convergência e de auto-correlação das cadeias geradas para garantir que as cadeias tenham convergido para a distribuição de interesse.

Observou-se o histórico das cadeias com o objetivo de identificar possíveis problemas de convergência, este gráfico está entre o valor da variável e o número de iterações e espera-se que este gráfico tenha um comportamento aleatório, isto é, não deve ficar preso em nenhuma área ao longo das iterações.

No Gráfico 7.2 observa-se o histórico das cadeias para cada um dos parâmetros de interesse:

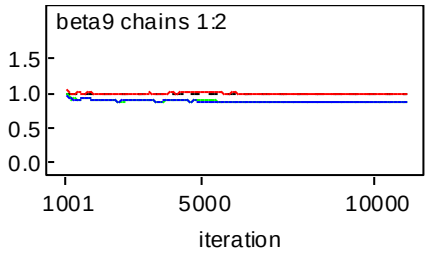
Gráfico 7.2 – Histórico das cadeias de Markov para os parâmetros do modelo Beta 0: (Intercepto)



Pode-se observar no Gráfico 7.2 que o histórico das cadeias permanecem aleatório ao longo das iterações, indicando convergência. Os históricos dos demais parâmetros estão no Apêndice.

A estatística de Gelman Rubin indicou convergência das cadeias para todos os casos. A seguir, Gráfico 7.3, encontra-se um dos piores casos do gráfico de Gelman Rubin entre todos os parâmetros ao longo das iterações.

Gráfico 7.3 – Gráficos de Gelman Rubin para os parâmetros do modelo Beta 9: (D2_AGE)

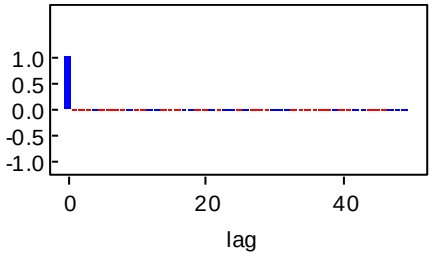


O Gráfico 7.3 reforça a indicação de convergência fornecida pelo histórico da cadeia visto anteriormente, Assim como este, todos os demais gráficos de Gelman Rubin indicam a convergência das cadeias.

No Gráfico 7.4 se observa como exemplo o gráfico de auto-correlação para o parâmetro beta11, todas as demais variáveis obtiveram gráficos de

auto-correlação parciais muito semelhantes, todos indicando que as cadeias não estão auto-correlacionadas.

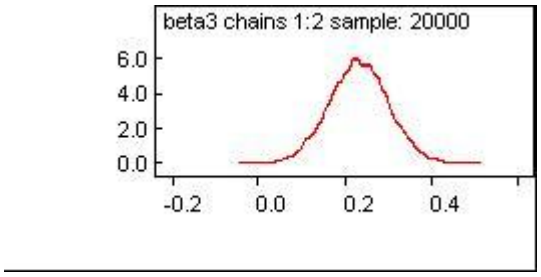
Gráfico 7.4 – Auto de auto-correlação das cadeias geradas



Após a verificação das suposições feitas para a convergência das cadeias pode-se analisar as distribuições *a posteriori*. No Gráfico 7.5 estão dispostas as aproximações das densidades *a posteriori* para os parâmetros de interesse.

Gráfico 7.5 – Densidades *a posteriori* para os parâmetros de interesse

Beta 3: (D_LIMITE)

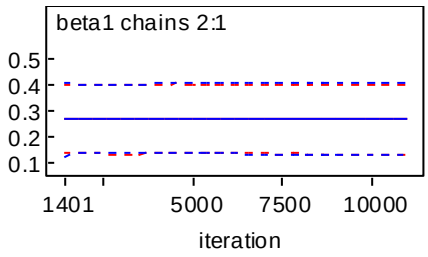


Todas as distribuições *a posteriori* são aproximadamente simétricas e as médias dos parâmetros são muito próximas do valor estimado através do método de máxima verossimilhança, como era esperado já que as prioris utilizadas no ajuste eram não informativas. Pode-se notar que as densidades dificilmente incluem o valor zero, ocorrendo em alguns poucos casos de o valor zero estar bem próximo ao limite das caudas da distribuição. O Gráfico 7.6

apresenta o parâmetro com o respectivo intervalo de credibilidade, espera-se que os mesmos não sejam muito amplos e que não incluam o valor zero.

Gráfico 7.6 – Intervalos de credibilidade para os parâmetros ajustados

Beta 1: (D_TPCLIENT)



Os gráficos dos demais parâmetros se comportam de forma muito parecida, ou seja, nenhum dos intervalos conteve o valor zero, indicando a significância de todos os parâmetros do ajuste.

Na Tabela 7.4 apresentam-se as estatísticas obtidas no ajuste. Os parâmetros obtidos são muito próximos dos estimados por máxima verossimilhança, tanto a média quanto a mediana. Através dos intervalos de credibilidade nenhum dos parâmetros pode ser considerado não significativo. As estimativas dos desvios padrão estão muito próximas dos obtidos no ajuste clássico.

Tabela 7.4 – Análise dos estimadores Bayesianos

Parâmetro	média	Erro Padrão	Mediana	2.5%	97.5%
Intercepto	-1,151	0,08115	-1.31	-1.151	-0.9921
D_TPCLIENT	0,271	0,06874	0.1358	0.2706	0.4046
D_SEXO	0,1512	0,06442	0.02506	0.1509	0.278
D_LIMITE	0,23	0,06825	0.09706	0.2305	0.3649
D1_CEP	-0,2677	0,08142	-0.4294	-0.2676	-0.1092
D2_CEP	0,2931	0,09666	0.1023	0.2931	0.4815
D1_TPEMPREG	0,2358	0,08061	0.07797	0.236	0.3942
D2_TPEMPREG	-0,2751	0,08223	-0.4374	-0.2755	-0.1142
D1_AGE	0,6166	0,1069	0.4074	0.6167	0.8258
D2_AGE	0,5869	0,1095	0.3718	0.5872	0.8024
D3_AGE	0,2449	0,09038	0.06836	0.2454	0.4232
D4_AGE	-0,4091	0,08965	-0.584	-0.4076	-0.233

Assim como no exemplo clássico, após o ajuste do modelo é necessário definir um ponto de corte que maximize a sensibilidade e a especificidade do modelo. O Gráfico 7.7 indica o melhor ponto de corte para o modelo de regressão logística ajustado, aproximadamente 0.3.

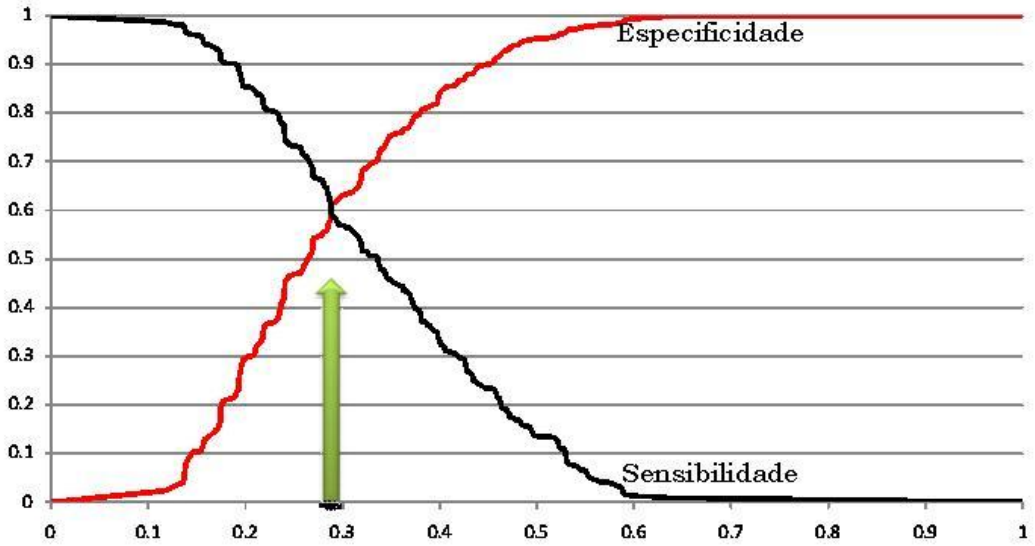


Gráfico 7.7 – Especificidade e Sensibilidade x Ponto de corte

Após a definição do ponto de corte escolhe-se a regra de decisão para a classificação dos clientes como bons ou maus pagadores. Esta forma de escolha otimiza a sensibilidade e a especificidade do modelo. A aplicação do modelo Bayesiano no banco de dados de teste pode ser vista na seção 7.4 de comparação dos modelos.

7.2.3 Ajuste do modelo de redes neurais

As mesmas variáveis *dummy* utilizadas no modelo logístico clássico e Bayesiano foram apresentadas às redes neurais ajustadas neste trabalho.

Inicialmente ajustou-se uma rede neural com um único neurônio e em seguida redes *multilayer perceptron* com uma e duas camadas ocultas.

7.2.3.1 Um único neurônio

Um simples neurônio pode ser considerado um modelo linear generalizado, onde a função de ativação da rede se equivale à função de ligação. O modelo de regressão logística é um caso particular dos modelos lineares generalizados.

A arquitetura desta rede é descrita na Figura 7.1.

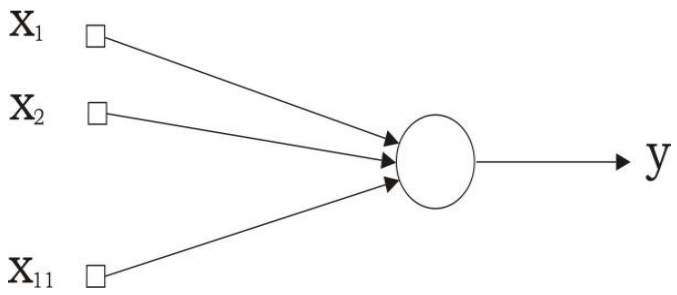


Figura 7.1 – Arquitetura de rede neural com um único neurônio

Através do método de aprendizado da regra delta foram ajustadas várias redes para diferentes combinações entre os parâmetros: taxa de aprendizado e constante *momentum*. Os valores $\eta = 0.1$ e $\alpha = 0.5$ foram escolhidos após diferentes redes através dos valores das estatísticas extraídas do ajuste do modelo no banco de dados de treinamento: erro médio, soma dos quadrados dos resíduos e o número de parâmetros no modelo. A função de ativação logística foi escolhida para a rede.

O número de épocas utilizadas no treinamento da rede foi igual a 29. O gráfico do erro médio ao longo das iterações é dado no Gráfico 7.8.

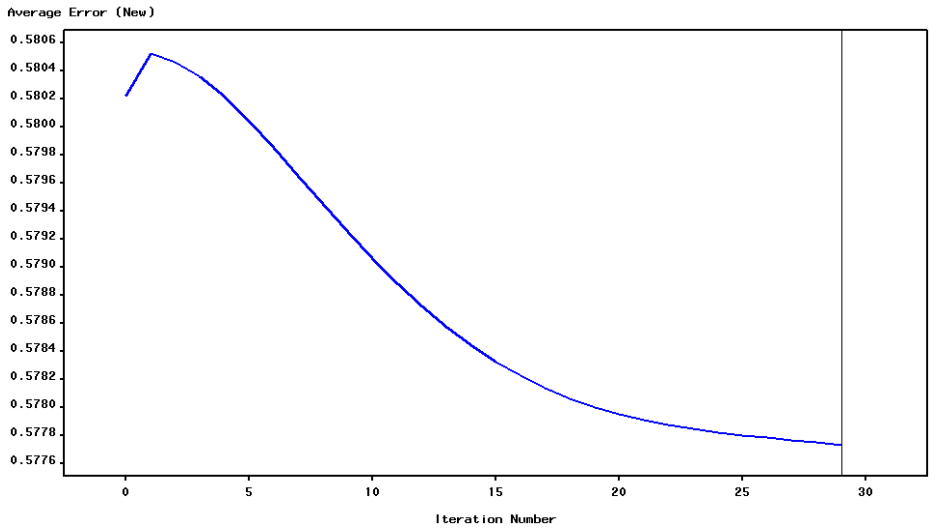


Gráfico 7.8 – Erro médio x Número de iterações

Os pesos sinápticos ajustados foram:

Tabela 7.5 – Pesos sinápticos ajustados para a rede neural com um único neurônio

Parâmetro	Peso sináptico ajustado
BIAS	- 0.7164
D_TPCLIENT	- 0.1246
D_SEXO	- 0.0751
D_LIMITE	- 0.1114
D1_CEP	0.1692
D2_CEP	- 0.0775
D1_TPEMPREG	- 0.1099
D2_TPEMPREG	0.1788
D1_AGE	- 0.1717
D2_AGE	- 0.1625
D3_AGE	- 0.0252
D4_AGE	0.2651

Após o ajuste do modelo é necessário definir um ponto de corte que maximize a sensibilidade e a especificidade do modelo. O Gráfico 7.9 indica o melhor ponto de corte para o modelo de redes neurais com um único neurônio ajustado, aproximadamente 0.31.

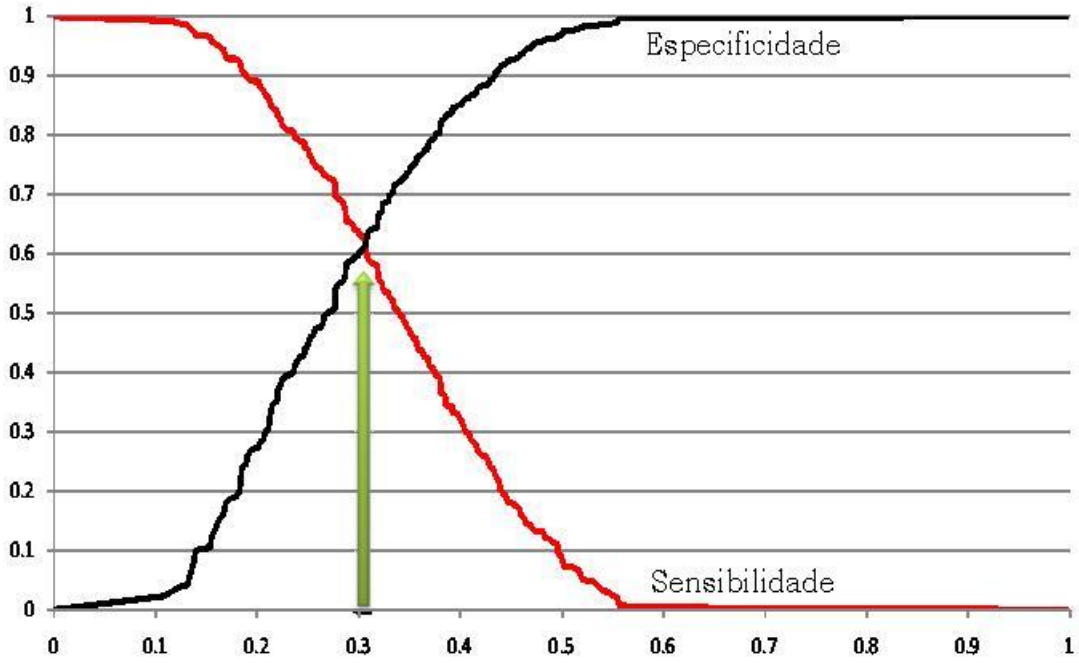


Gráfico 7.9 – Especificidade e Sensibilidade x Ponto de corte

Após a definição do ponto de corte pode-se definir a regra de decisão para a classificação dos clientes como bons ou maus pagadores. Esta forma de escolha otimiza a sensibilidade e a especificidade do modelo. As aplicações dos modelos de redes neurais ajustados foram realizadas no banco de dados de teste como pode ser visto na seção 6.4 de comparação dos modelos.

7.2.3.2. MLP com uma camada oculta

Várias redes MLP com uma camada oculta foram ajustadas através da técnica de aprendizagem *back-propagation* variando as taxas de aprendizado, a constante *momentum* e o número de neurônios na camada oculta. Através dos valores obtidos para o erro médio, a soma dos quadrados dos resíduos e o número de parâmetros no modelo escolheu-se a rede com os melhores resultados.

Um total de três neurônios na camada oculta resultou em 40 pesos sinápticos ajustados. Com onze variáveis de entrada no modelo, cada entrada conectada com cada um dos três neurônios na camada oculta resulta em 33 pesos sinápticos, com mais três pesos sinápticos conectando a camada oculta com a camada de saída tem-se 33 pesos, como cada um dos quatro neurônios da rede possui o termo bias o total de pesos sinápticos da rede é igual a 40. A topologia da rede está representada na Figura 7.2.

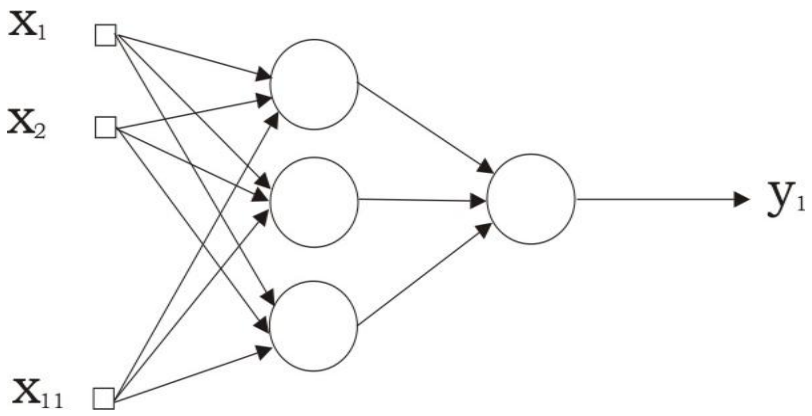


Figura 7.2 – Rede MLP com uma camada intermediária

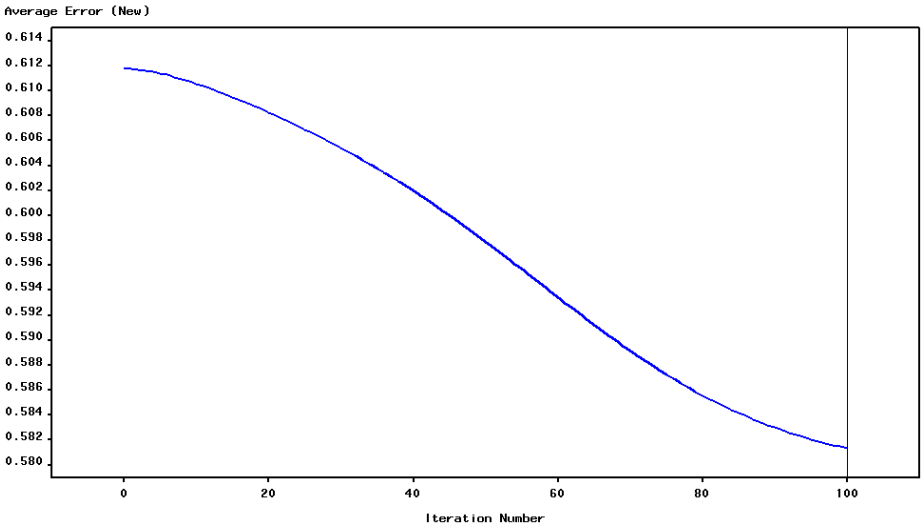


Gráfico 7.10 – Erro médio x Número de iterações

O número de épocas utilizadas no treinamento da rede foi igual a 100. O Gráfico 7.10 ilustra o erro médio ao longo das iterações. Os 40 pesos sinápticos ajustados são dados na Tabela 7.6, onde H_{ij} corresponde ao j-ésimo neurônio da i-ésima camada oculta e MAU1 é o neurônio da camada de saída.

Tabela 7.6 – Pesos sinápticos ajustados para a rede MLP com uma camada intermediária

De	Para	Peso sináptico
D1_AGE0_	D1_AGE0 -> H11	-0.211706186
D1_CEP0_	D1_CEP0 -> H11	0.176670838
D1_TPEMP	D1_TPEMPREG0 -> H11	-0.217717406
D2_AGE0_	D2_AGE0 -> H11	-0.180144167
D2_CEP0_	D2_CEP0 -> H11	-0.055822406
D2_TPEMP	D2_TPEMPREG0 -> H11	0.395455756
D3_AGE0_	D3_AGE0 -> H11	-0.431361048
D4_AGE0_	D4_AGE0 -> H11	0.557477558
D_LIMITE	D_LIMITE0 -> H11	0.005624007
D_SEXO0_	D_SEXO0 -> H11	-0.134078701
D_TPCLIE	D_TPCLIENT0 -> H11	-0.515320111
D1_AGE0_	D1_AGE0 -> H12	0.267541942
D1_CEP0_	D1_CEP0 -> H12	-0.310801417
D1_TPEMP	D1_TPEMPREG0 -> H12	0.618155961
D2_AGE0_	D2_AGE0 -> H12	0.094295164
D2_CEP0_	D2_CEP0 -> H12	0.349623413
D2_TPEMP	D2_TPEMPREG0 -> H12	0.012345419
D3_AGE0_	D3_AGE0 -> H12	0.108970715
D4_AGE0_	D4_AGE0 -> H12	0.019288316
D_LIMITE	D_LIMITE0 -> H12	0.331220858
D_SEXO0_	D_SEXO0 -> H12	0.176449289
D_TPCLIE	D_TPCLIENT0 -> H12	0.265605659
D1_AGE0_	D1_AGE0 -> H13	-0.109729441
D1_CEP0_	D1_CEP0 -> H13	0.047173736
D1_TPEMP	D1_TPEMPREG0 -> H13	0.578921959
D2_AGE0_	D2_AGE0 -> H13	0.646780263

Tabela 7.6 – Pesos sinápticos ajustados para a rede MLP com uma camada intermediária (Cont.)

De	Para	Peso sináptico
D2_CEP0_	D2_CEP0 -> H13	-0.200647331
D2_TPEMP	D2_TPEMPREG0 -> H13	-0.600005724
D3_AGE0_	D3_AGE0 -> H13	-0.330046962
D4_AGE0_	D4_AGE0 -> H13	-0.529257377
D_LIMITE	D_LIMITE0 -> H13	0.257470199
D_SEXO0_	D_SEXO0 -> H13	-0.197889319
D_TPCLIE	D_TPCLIENT0 -> H13	0.268867535
BIAS_H11	BIAS -> H11	0.036814019
BIAS_H12	BIAS -> H12	-0.484232762
BIAS_H13	BIAS -> H13	-0.72339963
H11_MAU1	H11 -> MAU1	0.973522713
H12_MAU1	H12 -> MAU1	-0.64691739
H13_MAU1	H13 -> MAU1	-0.67960849
BIAS_MAU	BIAS -> MAU1	-0.644524633

A definição do ponto de corte, dada através da maximização da sensibilidade e da especificidade é ilustrada abaixo. O Gráfico 7.11 indica o melhor ponto de corte para o modelo de redes neurais com uma camada intermediária constituída de 3 neurônios, um neurônio na camada de saída e 11 neurônios na camada de entrada, o ponto de corte definido é aproximadamente 0.30.

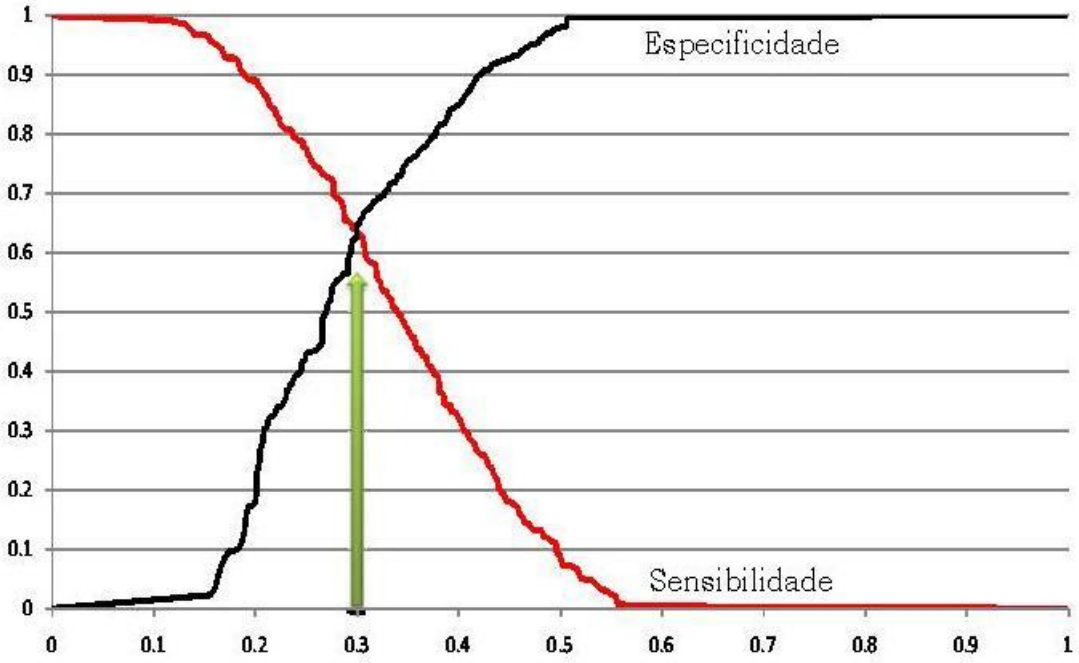


Gráfico 7.11 – Especificidade e Sensibilidade x Ponto de corte

Definido o melhor ponto de corte, sob o critério da máxima especificidade e sensibilidade pode-se definir a regra de decisão para a classificação dos clientes como bons ou maus pagadores.

7.2.3.3 MLP com duas camadas ocultas

Assim como no ajuste da rede neural com uma camada intermediária, várias redes com duas camadas intermediárias foram ajustadas através da técnica de aprendizagem *back-propagation*. Os mesmos parâmetros foram variados, como, taxa de aprendizado, constante *momentum* e o número de neurônios em ambas as camadas. A rede foi escolhida com base na soma de quadrados dos resíduos, erro médio e o número de parâmetros no modelo.

Com um total de 20 neurônios na primeira camada oculta e 10 neurônios na segunda obteve-se 472 pesos sinápticos ajustados, 300 pesos a menos do que considerando 28 neurônios na primeira camada oculta e 14 neurônios na segunda. Optou-se pela primeira arquitetura por causa do número de parâmetros e pela pouca diferença obtida entre a soma de quadrados dos resíduos e o erro médio. A topologia da rede está representada na Figura 7.3.

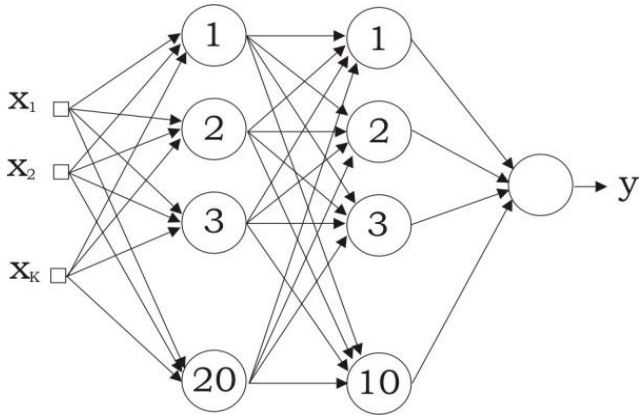


Figura 7.3 – Rede MLP com duas camadas intermediárias

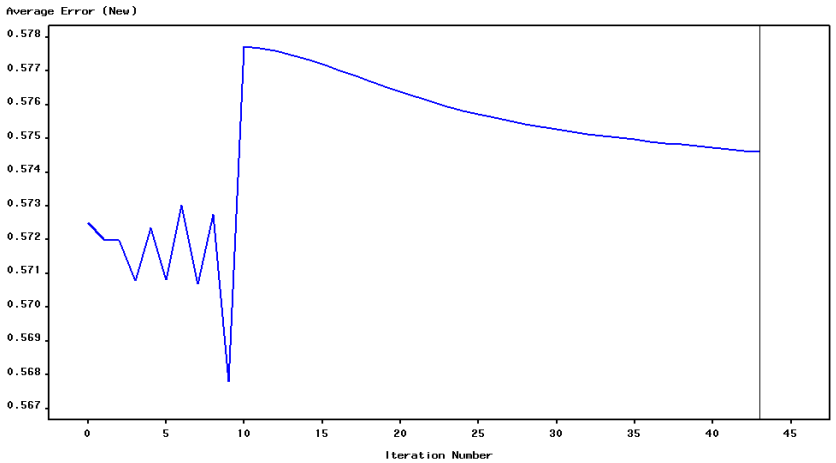


Gráfico 7.12 – Erro médio x Número de iterações

O número de épocas utilizadas no treinamento da rede foi igual a 43. O Gráfico 7.12 representa o erro médio ao longo das iterações. Os 472 pesos sinápticos ajustados não serão exibidos como no caso anterior por não haver interpretação.

A definição do ponto de corte é realizada através dos mesmos critérios que o ajuste anterior. O Gráfico 7.13 indica o melhor ponto de corte para o modelos de redes neurais com duas camadas intermediárias descrito acima. O ponto definido é de aproximadamente 0.31.

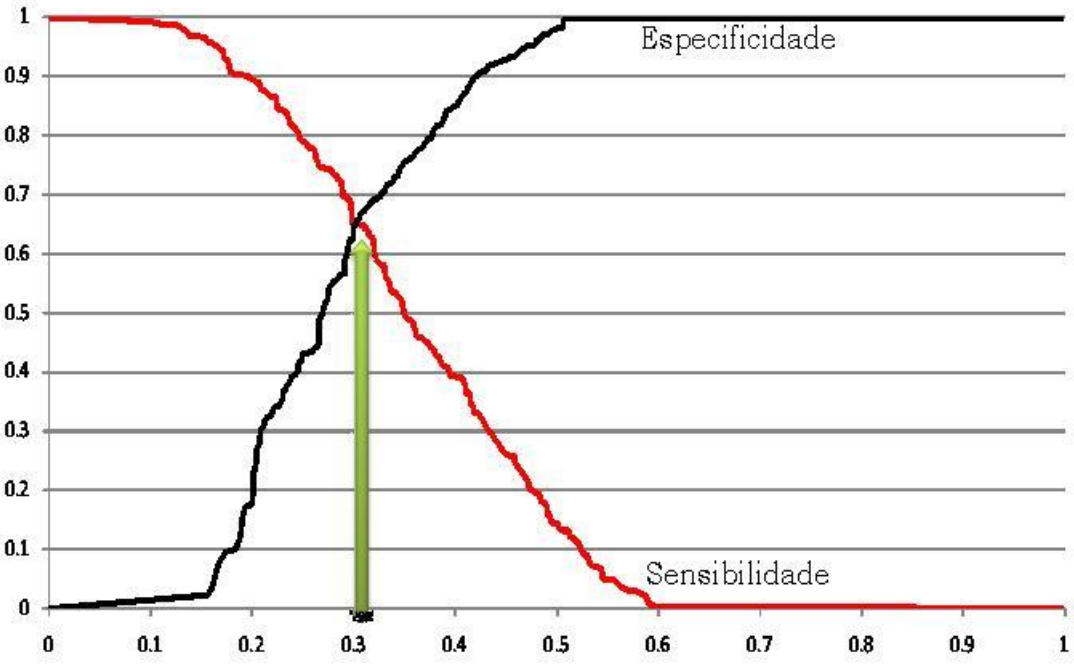


Gráfico 7.13 – Especificidade e Sensibilidade x Ponto de corte

Definido o melhor ponto de corte, sob o critério da máxima especificidade e especificidade pode-se definir a regra de decisão para a classificação dos clientes como bons ou maus pagadores.

A seção a seguir compara todos os modelos ajustados.

7.3 Comparação dos modelos

Assim como no capítulo anterior, a comparação dos modelos ajustados foi realizada neste capítulo através de três análises: avaliação da capacidade preditiva, curva ROC, estatística de Kolmogorov-Smirnov (KS) e capacidade de acerto do modelo.

7.3.1 Capacidade preditiva

Para o modelo de regressão logística clássico as estimativas obtidas através das re-amostragens *bootstrap* podem ser encontradas na Tabela 7.7.

Tabela 7.7 – *Bootstrap* (Modelo de *Credit Score* clássico)

decil	Risco				Taxa de inadimplência				Participação			
	Obs.	*Bootstrap*			Obs.	*Bootstrap*			Obs.	*Bootstrap*		
		Est.	LI	LS		Est.	LI	LS		Est.	LI	LS
0	53.96	54.04	48.24	59.83	54.34	53.74	47.94	59.54	174	173	154	192
1	44.24	44.24	37.67	50.81	43.64	44.44	37.87	51.01	140	143	124	162
2	38.50	38.55	31.45	45.66	40.18	40.57	33.47	47.67	129	131	108	153
3	33.67	33.75	27.25	40.25	28.64	27.42	20.92	33.92	92	88	67	110
4	29.68	29.75	24.00	35.49	32.4200	32.55	26.80	38.29	104	105	88	122
5	26.72	26.83	20.25	33.41	30.9090	30.83	24.25	37.41	99	99	80	119
6	23.56	23.62	16.82	30.42	24.5454	23.93	17.13	30.73	79	77	58	96
7	20.45	20.54	14.05	27.02	23.2876	25.14	18.65	31.62	75	81	62	100
8	17.58	17.76	11.65	23.87	17.7272	16.90	10.79	23.01	57	54	35	74
9	13.38	13.45	7.70	19.20	15.9090	16.34	10.59	22.10	51	53	35	71
Total	30.17	30.25	23.91	36.60	31.1591	31.19	24.84	37.53	100	101	81	120

Pode-se observar na Tabela 7.7 que em todos os decis a taxa de inadimplência está contida no intervalo de confiança para o risco de crédito predito pelo modelo. O Gráfico 7.14 permite a avaliação do modelo de acordo com a participação da taxa de inadimplência de cada decil na taxa geral de inadimplência, esta medida é conhecida como *lift*, o resultado esperado para um modelo bem ajustado é uma curva estritamente decrescente monotônica, mostrando que quanto maior o decil, menor a taxa de inadimplência.

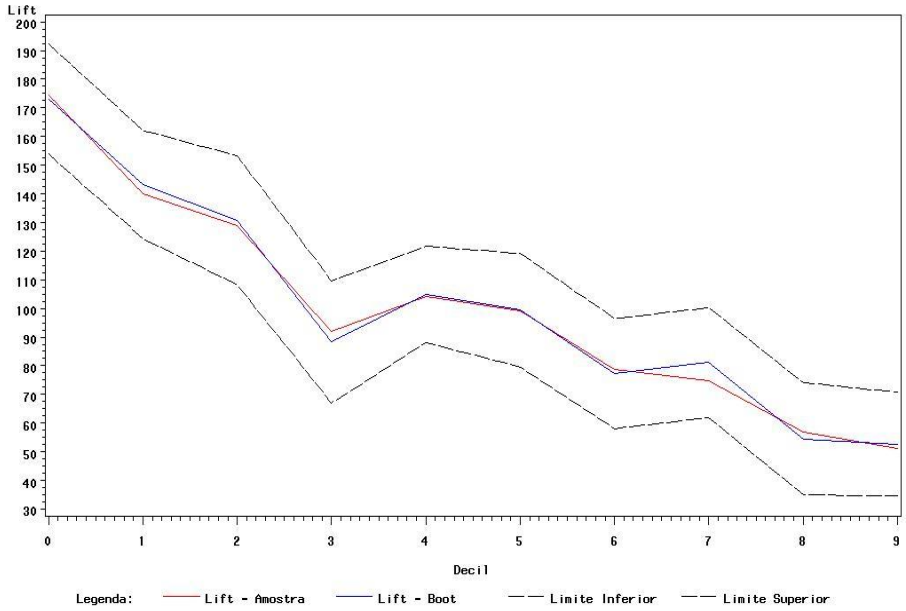


Gráfico 7.14 – Participação da taxa de inadimplência na taxa geral de inadimplência

Pode-se observar no Gráfico 7.14 que os valores estimados não se comportam de forma estritamente decrescente, mostrando uma fragilidade do modelo. Na prática costuma-se agrupar os decis de forma que a curva fique estritamente decrescente e a partir destas novas faixas de escore definir o melhor ponto de corte para a concessão de crédito.

Aplicando o modelo ajustado na base de teste podemos validar o modelo através da análise de decis de acordo com as estimativas obtidas pelo modelo para o risco de inadimplência e as taxas observadas. O Gráfico 7.15 permite a comparação das estimativas do modelo e a verdadeira resposta dos dados para cada decil.

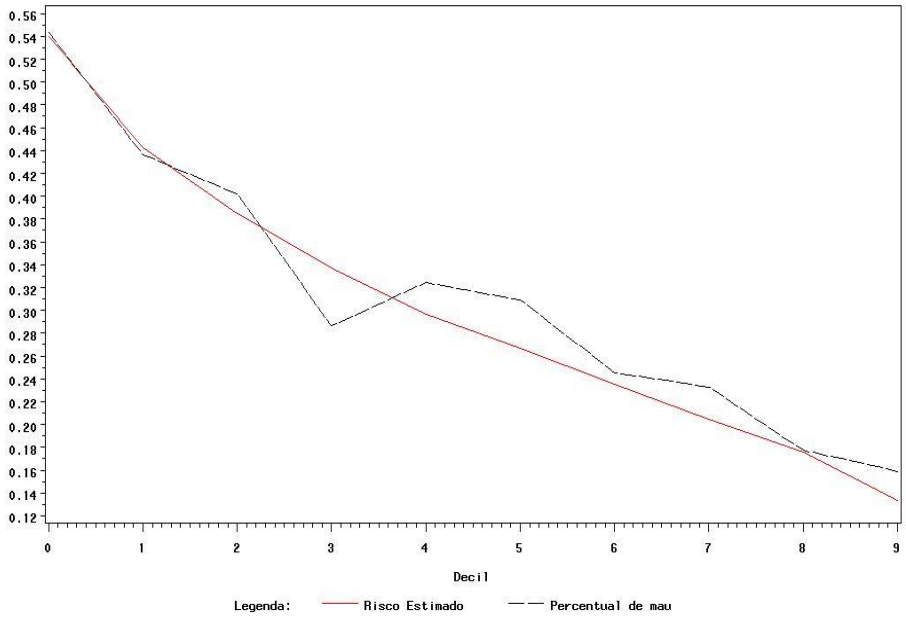


Gráfico 7.15 – Risco estimado x Taxa observada

Na maioria dos decis o modelo está se aproximando de forma satisfatória da verdadeira taxa de inadimplência. Observe que o risco estimado pelo modelo logístico clássico está subestimando a taxa de inadimplência na maioria dos decis, o que não é muito apropriado em problemas de *Credit Score*. A precisão das estimativas obtidas fica muito boa para os clientes de maior risco, principalmente para os três primeiros decis, e para os clientes de menor risco os quatro últimos decis, ficando menor para o quarto decil. No

geral pode-se considerar um modelo bem ajustado, pois não há grandes erros para clientes de alto risco, embora o modelo esteja subestimando a inadimplência do modelo.

No modelo de regressão logística Bayesiano com priori não informativa esta mesma metodologia foi utilizada na avaliação da capacidade preditiva do modelo. Pode-se observar na Tabela 7.8 que, assim como no modelo de regressão logística, em todos os decis o intervalo de confiança para o risco inclui a verdadeira taxa de inadimplência.

Tabela 7.8 – *Bootstrap* (Modelo de *Credit Score* Bayesiano)

decis	Risco				Taxa de inadimplência				Participação			
	obs	*Bootstrap*			obs	*Bootstrap*			obs	*Bootstrap*		
		est	LI	LS		est	LI	LS		est	LI	LS
0	53.95	53.82	47.60	60.04	54.34	54.55	48.33	60.76	174	176	159	193
1	44.21	44.21	36.86	51.55	43.64	43.83	36.49	51.18	140	141	117	166
2	38.46	38.48	32.58	44.38	40.18	39.24	33.34	45.13	129	127	109	144
3	33.63	33.72	26.35	41.10	28.64	28.69	21.32	36.07	92	93	70	115
4	29.64	29.69	22.64	36.74	32.42	32.82	25.77	39.87	104	106	84	128
5	26.69	26.78	20.97	32.59	31.36	31.03	25.22	36.84	101	100	81	119
6	23.53	23.54	17.86	29.23	24.09	22.69	17.01	28.37	77	73	56	91
7	20.44	20.47	13.13	27.81	23.29	24.27	16.93	31.60	75	78	58	99
8	17.58	17.75	12.79	22.71	17.73	17.35	12.39	22.31	57	56	42	71
9	13.37	13.46	7.99	18.92	15.91	15.48	10.02	20.95	51	50	33	67
Total	30.15	30.19	23.88	36.51	31.16	30.99	24.68	37.31	100	100	81	119

Os valores da participação para os decis, como podem ser também observados no Gráfico 7.16 indicam que o modelo possui certa fragilidade por não apresentar uma curva estritamente decrescente monotônica. O comportamento apresentado é extremamente similar ao modelo clássico, como era esperado, dado que o modelo ajustado através da metodologia Bayesiana foi baseado em uma *priori* não informativa.

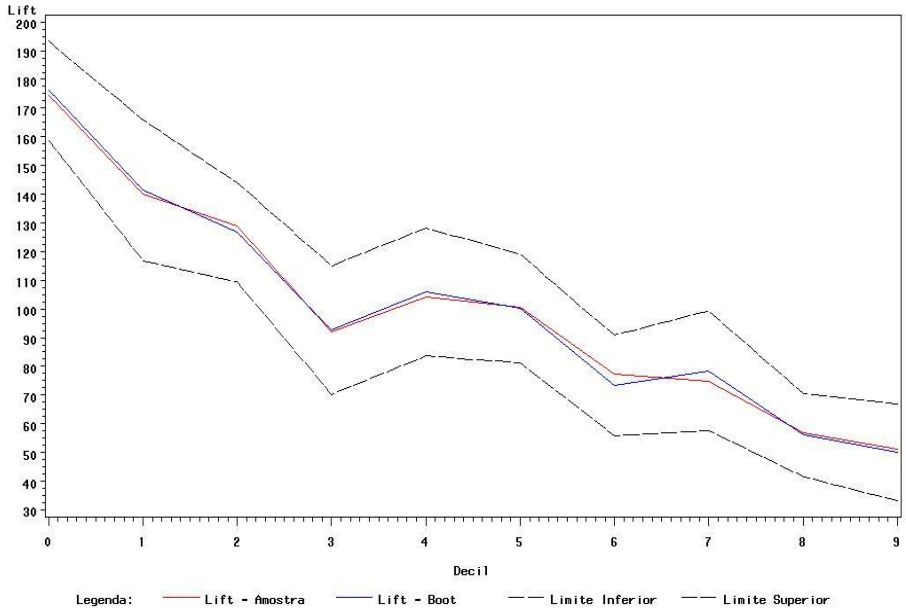


Gráfico 7.16 – Participação na taxa de inadimplência na taxa geral de inadimplência

Após a aplicação do modelo ajustado na base de teste pode-se validar o modelo através da comparação entre as estimativas obtidas pelo modelo para o risco de inadimplência e as taxas observadas. O Gráfico 7.17 permite a comparação das estimativas do modelo e a verdadeira resposta dos dados para cada decil.

Como esperado o Gráfico 7.17 ficou muito semelhante ao Gráfico 7.15, já que não houve informação externa aos dados dada através da *priori* (*priori* não informativa). Na maioria dos decis o modelo subestimou as taxas de inadimplência, assim como no modelo clássico.

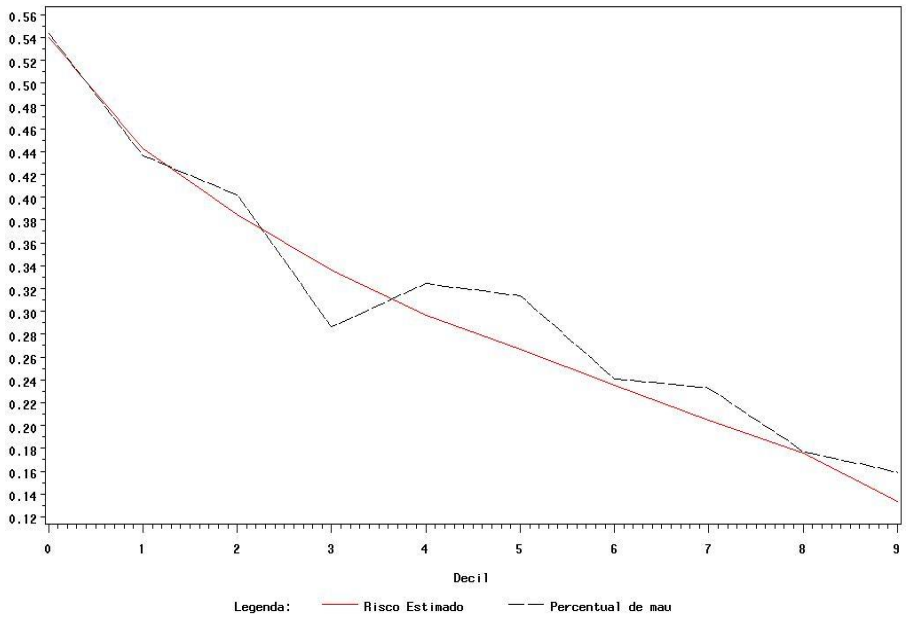


Gráfico 7.17 – Risco estimado x Taxa observada

A análise da capacidade preditiva do modelo de rede neural composto por um único neurônio teve as seguintes estimativas através das re-amostragens *bootstrap*, Tabela 7.9.

Tabela 7.9 – *Bootstrap* (Modelo de Rede Neural com um único neurônio)

decis	Risco				Taxa de inadimplência				Participação			
	obs	*Bootstrap*			obs	*Bootstrap*			obs	*Bootstrap*		
		est	LI	LS		est	LI	LS		est	LI	LS
0	52.14	52.15	46.80	57.50	54.79	54.99	49.63	60.34	176	176	162	190
1	43.06	43.05	37.57	48.52	41.82	40.90	35.43	46.38	134	131	115	147
2	38.00	37.96	31.30	44.62	40.64	40.75	34.09	47.41	130	131	110	152
3	33.87	33.84	26.61	41.06	30.45	29.45	22.23	36.67	98	94	73	116
4	30.09	30.07	25.12	35.02	31.05	32.53	27.58	37.48	100	104	88	121
5	27.29	27.29	20.94	33.63	30.45	30.13	23.78	36.47	98	97	76	117
6	24.13	24.08	18.36	29.81	25.45	25.33	19.60	31.05	82	81	66	97
7	20.93	20.93	15.02	26.84	23.29	24.93	19.02	30.83	75	80	61	99
8	17.48	17.40	12.50	22.30	17.73	16.81	11.91	21.71	57	54	39	69
9	12.94	12.90	8.64	17.16	15.91	16.20	11.95	20.46	51	52	38	66
Total	29.99	29.97	24.29	35.65	31.16	31.20	25.52	36.88	100	100	83	117

As verdadeiras taxas de inadimplência estão todas dentro dos intervalos de confiança para o risco em cada um dos dez decis do escore. Na Tabela 7.9 e no Gráfico 7.18 a seguir pode-se observar também que as taxas de inadimplência e a participação dos escores não formam uma curva estritamente decrescente, embora a alteração da taxa observada de inadimplência no terceiro decil tenha amenizado um pouco o não comportamento estritamente decrescente da curva.

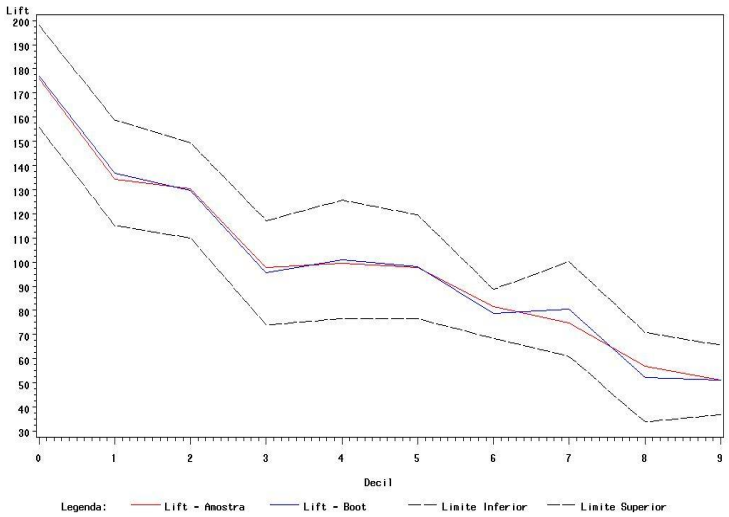


Gráfico 7.18 – Participação na taxa de inadimplência na taxa geral de inadimplência

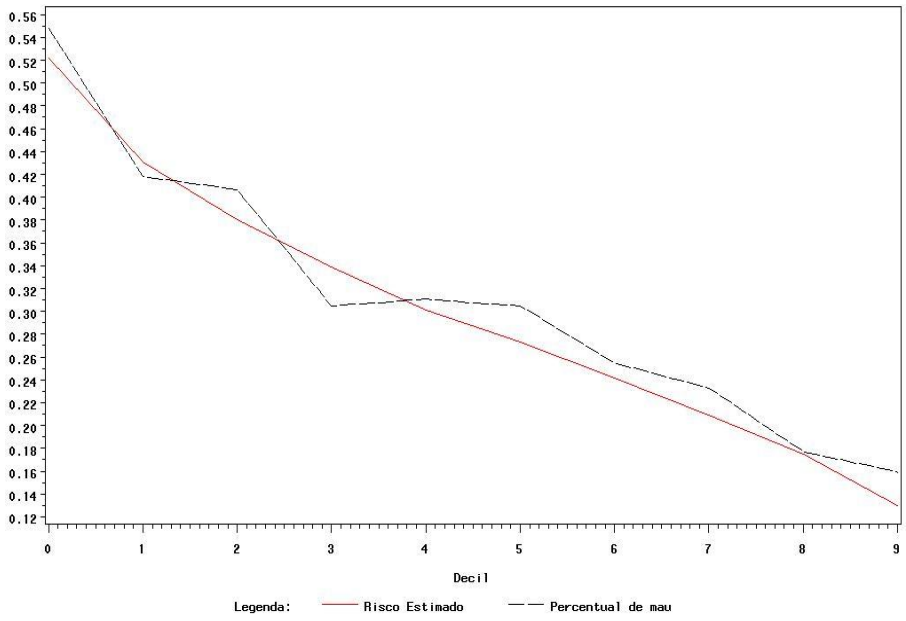


Gráfico 7.19 – Risco estimado x Taxa observada

O Gráfico 7.19 mostra que o modelo de rede neural com um único neurônio também subestima o modelo. Observa-se que a precisão dos dois primeiros decis, clientes com maior risco, foi levemente reduzida, mas de forma geral o modelo se aproximou mais da verdadeira taxa de inadimplência.

Com o modelo de rede neural *multilayer perceptron* com uma camada intermediária as seguintes estimativas de re-amostragens obtidas são dadas na Tabela 7.10.

Tabela 7.10 – *Bootstrap* (Modelo de Rede Neural MLP com uma camada intermediária)

decis	Risco				Taxa de inadimplência				Participação			
	obs	*Bootstrap*			obs	*Bootstrap*			obs	*Bootstrap*		
		Est	LI	LS		est	LI	LS		est	LI	LS
0	48.35	48.30	40.84	55.76	52.05	53.13	45.67	60.59	167	171	151	190
1	41.82	41.76	34.25	49.27	45.00	45.78	38.27	53.28	144	147	125	169
2	37.42	37.40	28.05	46.75	38.81	37.04	27.69	46.39	125	119	91	147
3	33.06	33.04	25.64	40.45	32.73	32.34	24.94	39.74	105	104	80	127
4	29.83	29.80	23.64	35.97	34.25	35.42	29.25	41.58	110	114	92	135
5	27.73	27.70	22.59	32.80	26.36	25.49	20.38	30.59	85	82	67	97
6	25.41	25.37	19.81	30.92	24.09	23.65	18.09	29.20	77	76	58	93
7	21.43	21.33	15.25	27.40	22.83	22.68	16.61	28.75	73	73	53	92
8	19.67	19.65	14.29	25.00	18.64	18.34	12.99	23.70	60	59	42	76
9	16.66	16.62	12.32	20.93	16.82	17.07	12.77	21.38	54	55	41	68
Total	**30.14**	**30.10**	**23.67**	**36.52**	**31.16**	**31.09**	**24.67**	**37.52**	**100**	**100**	**80**	**119**

Observe que as taxas de inadimplência observadas pertencem ao intervalo de confiança para o risco em cada decil e a participação não obteve uma curva estritamente decrescente somente por causa do quarto decil, indicando uma boa capacidade preditiva.

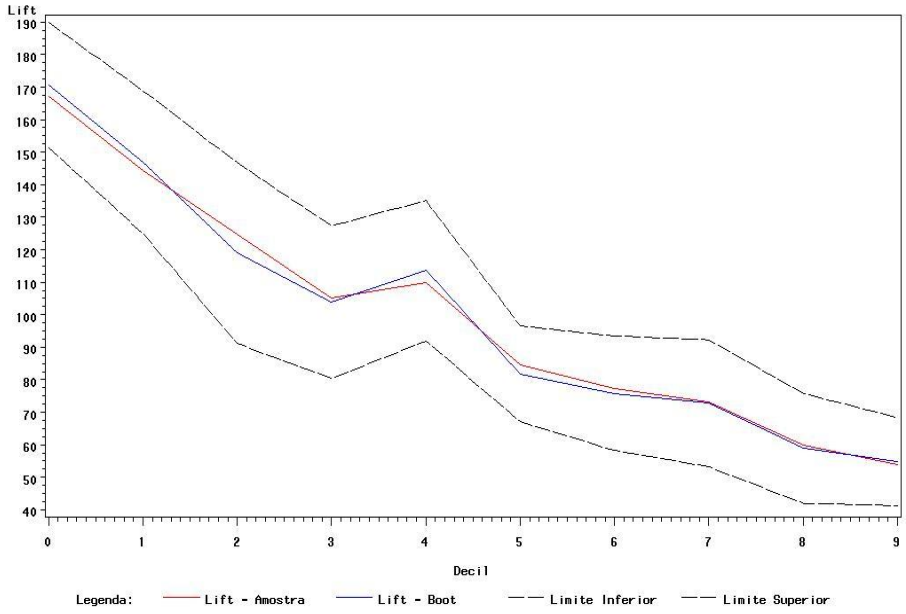


Gráfico 7.20 – Participação na taxa de inadimplência na taxa geral de inadimplência

Este modelo subestima a inadimplência para metade das faixas de escore, porém para o terceiro, o quinto, o sexto, o oitavo e o nono decil o modelo está superestimando a verdadeira taxa de inadimplência, o que é bom em modelos de *Credit Score*. Porém, a qualidade das estimativas para os clientes de alto risco foi reduzida aumentando a diferença entre a taxa de inadimplência estimada e a observada.

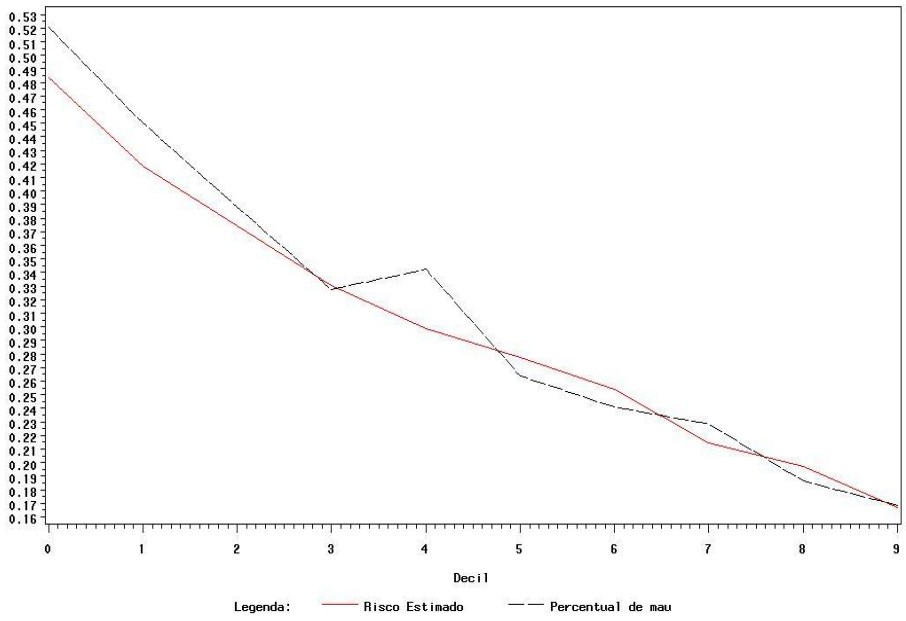


Gráfico 7.21 – Risco estimado x Taxa observada

De forma geral, comparando o risco estimado com as verdadeiras taxas de inadimplência, embora a diferença entre essas medidas tenham aumentado para os clientes de maior risco, o modelo de redes neurais com uma camada melhorou o ajuste para os clientes de baixo risco.

Para o modelo de redes neurais *multilayer perceptron* com duas camadas intermediárias as estimativas obtidas via *bootstrap* para o risco, a taxa de inadimplência e a participação da taxa de inadimplência de cada decil na taxa geral de inadimplência são dadas na Tabela 7.11.

Tabela 7.11 – *Bootstrap* (Modelo de Rede Neural MLP com duas camadas intermediárias)

decis	Risco				Taxa de inadimplência				Participação			
	obs	*Bootstrap*			obs	*Bootstrap*			obs	*Bootstrap*		
		est	LI	LS		est	LI	LS		est	LI	LS
0	51.53	51.52	44.79	58.24	47.03	47.12	40.40	53.84	151	151	132	171
1	45.37	45.34	37.98	52.71	48.18	47.45	40.09	54.81	155	152	128	177
2	39.29	39.23	32.90	45.55	42.92	41.94	35.62	48.26	138	135	116	154
3	34.40	34.35	28.03	40.67	28.64	28.10	21.79	34.42	92	90	70	111
4	31.03	31.01	25.48	36.53	30.59	30.85	25.33	36.38	98	99	80	118
5	27.59	27.53	21.64	33.43	27.73	27.17	21.28	33.07	89	87	69	105
6	24.38	24.36	17.02	31.69	27.27	28.91	21.57	36.25	88	93	72	114
7	20.27	20.23	12.93	27.52	25.11	25.00	17.70	32.30	81	81	60	101
8	16.75	16.75	11.45	22.05	19.09	19.73	14.42	25.03	61	63	48	79
9	11.33	11.23	6.17	16.29	15.00	14.36	9.30	19.42	48	46	31	62
Total	**30.20**	**30.15**	**23.84**	**36.47**	**31.16**	**31.06**	**24.75**	**37.38**	**100**	**100**	**81**	**119**

Neste caso a taxa real de inadimplência não pertence ao intervalo de confiança do risco estimado para a primeira faixa de escore, ou seja, para os clientes de maior risco.

O Gráfico 7.22 indica uma fragilidade do modelo, por não apresentar uma curva estritamente decrescente, começa crescente do primeiro para o segundo decil e tem duas inflexões: no terceiro e no quinto decil.

O Gráfico 7.23 mostra que o modelo subestima a verdadeira taxa de inadimplência em 60% das faixas.

De forma geral, comparando o risco estimado com as verdadeiras taxas de inadimplência, o modelo de rede neural com duas camadas apresentou melhores resultados nas faixas centrais de escores, quarto e quinto decis.

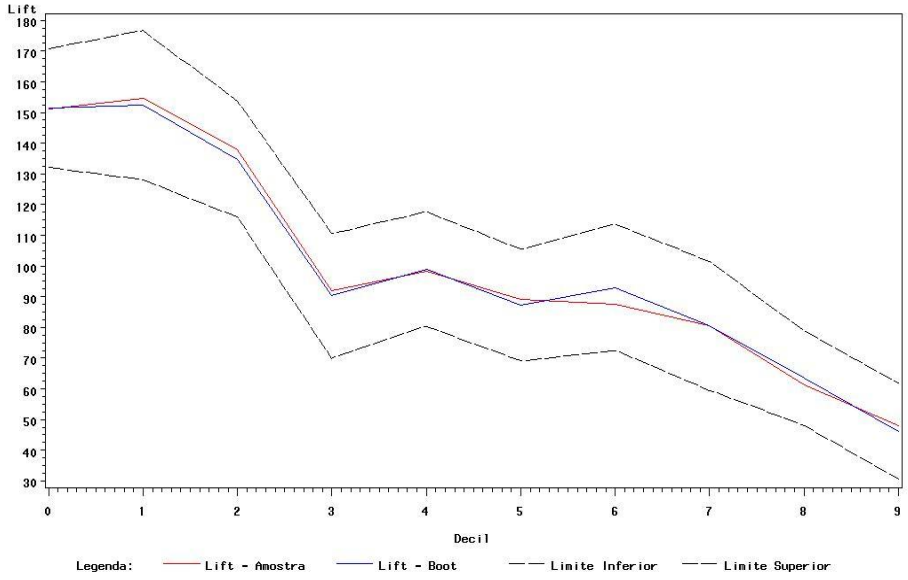


Gráfico 7.22 – Participação na taxa de inadimplência na taxa geral de inadimplência

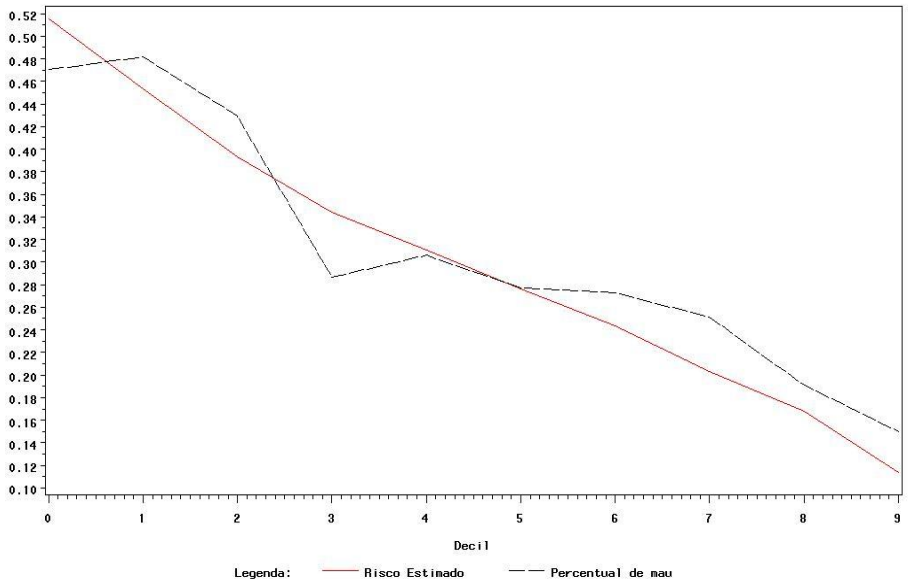


Gráfico 7.23 – Risco estimado x Taxa observada

7.3.2 Curva ROC

A curva ROC, assim como para o conjunto de dados reais, foi utilizada para comparar os diferentes modelos ajustados, a sensibilidade é representada pelo eixo Y e o valor de (1-especificidade) é representado pelo eixo X.

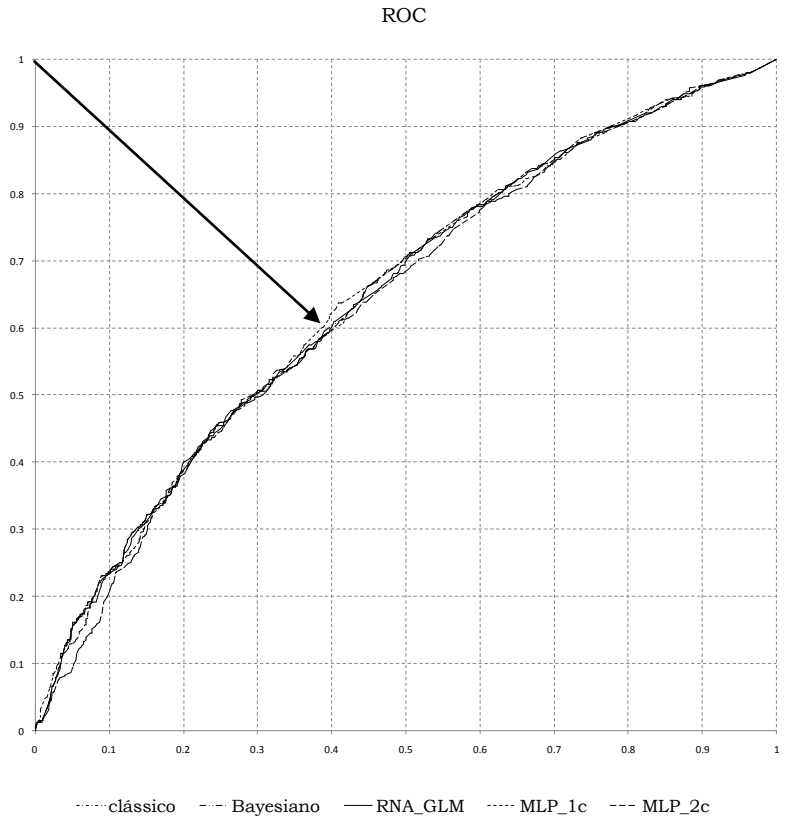


Gráfico 7.24 – Comparação dos modelos ajustados através da curva ROC

Pode-se observar no Gráfico 7.24 que todos os modelos tiveram um comportamento muito semelhante com relação à sensibilidade e à especificidade.

Buscando o ponto da reta com menor distância para o ponto (0,1) do gráfico, ou seja, especificidade e sensibilidade igual a um, encontra-se com

pequena vantagem a curva ROC do modelo de redes neurais *perceptron* com uma camada intermediária, como indicado pela seta no gráfico.

7.3.3 Estatística de Kolmogorov-Smirnov (KS) e capacidade de acerto dos modelos

A medida da estatística de Kolmogorov-Smirnov (KS) é apresentada na Tabela 7.12 juntamente com as medidas de capacidade de acerto total, acerto de clientes inadimplentes e acerto dos clientes adimplentes para cada um dos modelos ajustados.

Tabela 7.12 – Estatística KS e Capacidade de Acerto dos modelos nos bancos de teste (Total, Maus e Bons)

%	Clássico	Bayesiano	RNA_GLM	MLP_1c	MLP_2c
KS	21,37	21,37	21,27	22,87	20,99
CAT	60,56	60,56	61,34	61,84	60,38
CAM	58,63	59,50	57,31	55,99	58,77
CAB	61,38	61,04	63,16	64,48	61,11

Através da estatística KS, da capacidade de acertos Total e capacidade de acerto dos bons clientes o modelo de redes neurais *multilayer perceptron* com uma camada possui uma pequena vantagem em relação aos demais modelos, assim como na análise da curva ROC. Na análise da capacidade preditiva os modelos que apresentaram melhor desempenho para os clientes bons, ou seja, clientes de baixo risco foram os modelos: Redes Neurais *Multilayer Perceptron* com uma camada intermediária seguido pelo modelo de Regressão Logística clássico.

7.3.4 Combinação dos modelos

As medidas CAT, CAM e CAB só podem ser obtidas após a definição do ponto de corte para cada modelo ajustado e, assim como a curva ROC, está associada às medidas de sensibilidade e especificidade. A análise da estatística do KS, no entanto, não está atrelada à definição do ponto de corte, medindo a distância entre as distribuições empíricas dos escores dos grupos de clientes adimplentes e inadimplentes. A análise da capacidade preditiva leva em conta não apenas dois grupos: acima e abaixo do ponto de corte, mas o desempenho do modelo para cada uma das dez faixas de escore.

Pode-se observar que, embora a diferença seja pequena, cada modelo teve uma melhor capacidade preditiva para determinados decis. Os modelos, clássico e Bayesiano, tiveram um melhor desempenho na predição de inadimplência para os clientes de maior risco, três primeiros decis, a quinta e a sexta faixa de escore foram mais bem ajustadas para a rede neural de duas camadas e a terceira e as quatro últimas tiveram melhor desempenho para a rede *perceptron* com uma camada intermediária. De forma geral, o modelo clássico e o Bayesiano foram melhores para os clientes de maior risco, o modelo de redes neurais *perceptron* com uma camada intermediária obteve melhor desempenho na predição de inadimplência dos clientes de baixo risco e as faixas intermediárias foram mais bem explicadas pelo modelo de redes *perceptron* com duas camadas intermediárias.

Após definidas as regras de decisão para a concessão de crédito para cada modelo, pode-se trabalhar com regras lógicas a partir da decisão de cada modelo, montando assim uma espécie de comitê, fazendo com que a decisão final de dar ou não crédito ao cliente seja baseada em dois ou mais modelos ao invés de um só.

Para isso foram medidos os percentuais de discordância entre as decisões dos modelos, comparados dois a dois, como pode ser visto na Tabela 7.13.

Tabela 7.13 – Discordância entre as decisões dos modelos

Comparação dois a dois	Discordância
Clássico x Bayesiano	0.50%
Clássico x RNA com um neurônio	3.87%
Clássico x MLP com uma camada	9.84%
Clássico x MLP com duas camadas	10.93%
Bayesiano x RNA com um neurônio	3.37%
Bayesiano x MLP com uma camada	10.34%
Bayesiano x MLP com duas camadas	11.43%
RNA com um neurônio x MLP com uma camada	10.79%
RNA com um neurônio x MLP com duas camadas	11.43%
MLP com uma camada x MLP com duas camadas	11.38%

A discordância entre os modelos combinados dois a dois tem discordância máxima de 11,43%. Neste aspecto o julgamento do modelo clássico é muito semelhante ao julgamento do modelo Bayesiano, com apenas 0,5% de discordância no julgamento entre os dois. Para definir quais modelos serão considerados com aproximadamente o mesmo julgamento considerou-se semelhantes os modelos com discordância menor que 5%. Ficando com o julgamento dos modelos Clássico, Rede *Perceptron* com uma camada e com duas camadas. Os valores calculados para a capacidade de acerto são dados na Tabela 7.14.

Tabela 7.14 – comitê com três modelos

%	Maioria (≥ 2)	Unanimidade (= 3)
CAT	61.43	63.89
CAM	58.19	50.29
CAB	62.90	70.04

Como a maior parte do banco de dados é formada por clientes adimplentes e a regra de decisão tomada através do comitê com três modelos reduz a quantidade de predições com clientes inadimplentes o aumento na capacidade de acerto total e a capacidade de acertos dos adimplentes é esperada, não significando um aumento da capacidade preditiva do comitê formado pelos três modelos. Na modelagem de *Credit Score* a identificação dos clientes inadimplentes pelo modelo é o principal objetivo, pois um cliente inadimplente gera um custo maior para a instituição financeira do que a ausência de vários clientes adimplentes.

Capítulo 8

Comentários Finais

As três técnicas apresentadas neste trabalho: Regressão Logística Clássica, Regressão Logística Bayesiana e Redes Neurais Artificiais obtiveram resultados muito semelhantes para os dois bancos de dados utilizados, artificial e real.

Como esperado, os modelos clássicos ficaram extremamente semelhantes aos modelos Bayesianos pela utilização de uma priori não informativa para os parâmetros do modelo. Os modelos de Redes Neurais também apresentaram resultados parecidos.

Todas as técnicas apresentadas neste trabalho se mostraram igualmente capazes de classificar os clientes como inadimplentes para os bancos de dados aqui trabalhados. É importante ressaltar que todas as afirmações deste trabalho estão limitadas aos bancos de dados aqui trabalhados. Não são conclusões generalizadas.

Para o banco de dados artificial, observou-se que o modelo que apresentou a melhor capacidade de classificação foi o modelo de Redes

Neurais com duas camadas, com uma leve vantagem sobre as demais técnicas.

Para o banco de dados real, observou-se uma proximidade ainda maior das classificações fornecidas pelos modelos. Todos os modelos foram igualmente capazes de classificar a inadimplência dos modelos.

Bibliografia

[1] Abreu, H.J. (2005) *Aplicação da Análise de Sobrevivência em um problema de Credit Scoring e comparação com a Regressão Logística*. Dissertação de mestrado. DEs – UFSCar.

[2] Bedrick, E. J; Christensen, R; Johnson, W. (1997). *Bayesian Binomial Regression: Predicting Survival at a Trauma Center*. The American Statistician, August, Vol. 51, No. 3.

[3] Berthold, M; Hand, D. J. (2002). *Intelligent Data Analysis*. Springer, Berlin.

[4] Braga, A. P; Carvalho A. C. P. L. F; Ludemir, T. B. (2000) *Redes Neurais Artificiais Teoria e aplicações*. LTC.

[5] Campos, P. S. S. (2007) *Estimação Bayesiana em modelos de regressão logística*. Dissertação de mestrado. UFPA.

[6] Ehlers, R.S. (2004). *Métodos Computacionalmente Intensivos em Estatística*. Notas de aula. UFPR.

[7] Ehlers, R.S. (2005). *Introdução à inferência Bayesiana*. Notas de aula. UFPR.

[8] Farhat, C. A. V. (2003) *Análise de diagnóstico em regressão logística*. Dissertação de mestrado. IME-USP.

[9] Hand, D. J; Henley, W. E. (1997). *Statistical classification methods in consumer credit.* Journal Royal Statistical Society, A 160, 523-541.

[10] Haykin, S. (1999). *Neural Networks a comprehensive foundation* Prentice-Hall, Inc. New Jersey. 2ed.

[11] Hosmer, D.W; Lemeshow, S. (2000). *Applied Logistic Regression*. John Wiley & Sons, Inc. New York. 2ed.

[12] Kandel, E. R; Schwartz, J. H. (1992). *Principles of Neural Science*, 3ed, New York, Elsevier.

[13] McCullagh, P; Nelder, J. A. (1989). *Generalized Linear Models*, 2ed, Chapman & Hall, London.

[14] McCulloch, W. S; Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics, 5, 115-133.

[15] Minsky, M; Papert, S. (1969). *Perceptrons*, MIT Press, Cambridge, M. A.

[16] Paulino, C. D; Turkman, A; Murteira, B. (2003). *Estatística Bayesiana*. Fundação Calouste Gulbenkian, Lisboa.

[17] Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. Psychol. Rev., 65, 386-408.

[18] Rosenblatt, F. (1962). *Principles of Neurodynamics: Perceptrons and the theory of brain mechanisms*. Spartan Books, New York.

[19] Rud, O. P. (2001). *Data Mining Cookbook*, John Wiley & Sons, Inc.

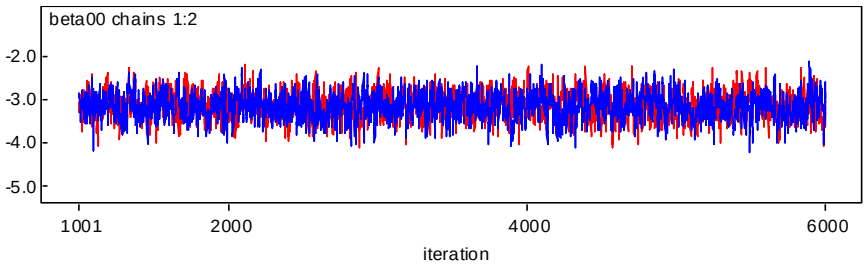
[20] Schwarz, G. (1978). *Estimating the dimension of a model*, *Annals of Statistics* Vol.6, 461-464.

Apêndice

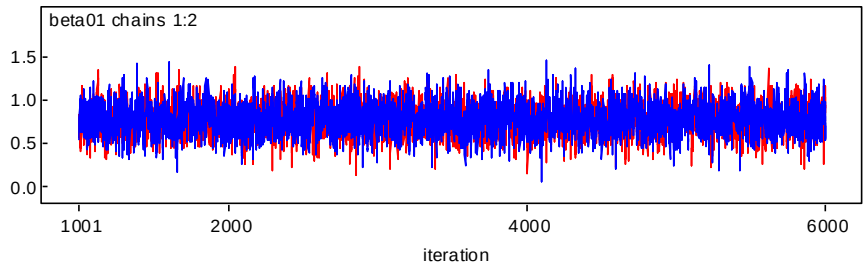
Histórico das cadeias geradas para os dados do capítulo 6:

Histórico das cadeias de Markov para os parâmetros do modelo

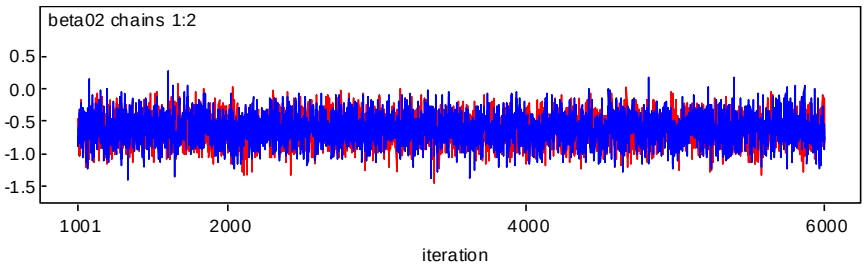
Beta 00: (intercepto)



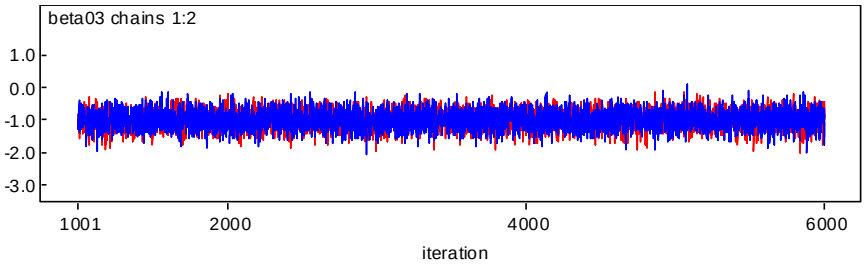
Beta 01: (DUM1_V1)



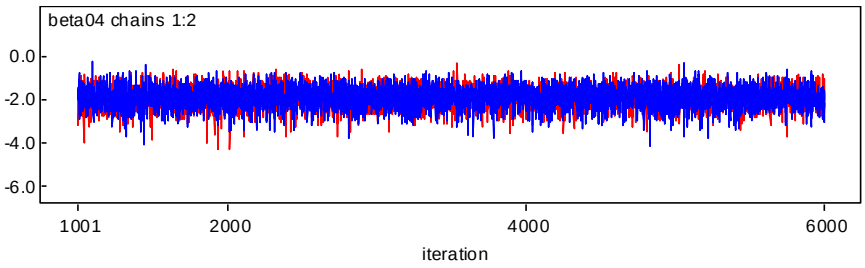
Beta 02: (DUM2_V1)



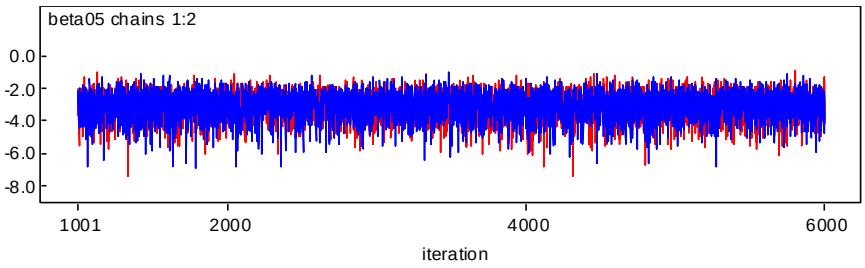
Beta 03: (DUM3_V1)



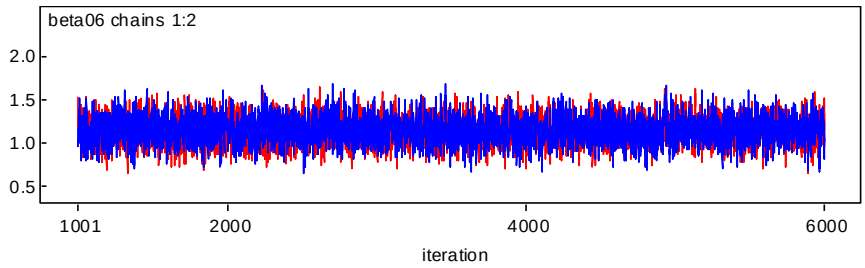
Beta 04: (DUM4_V1)



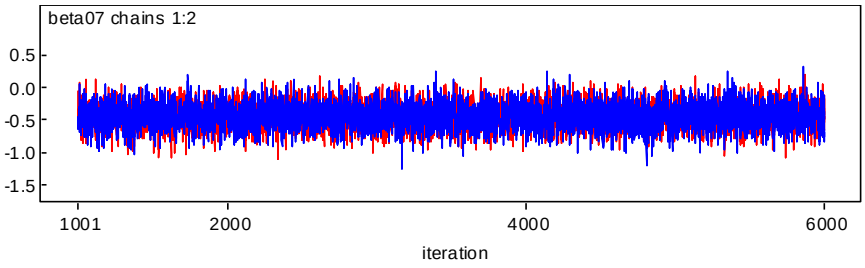
Beta 05: (DUM5_V1)



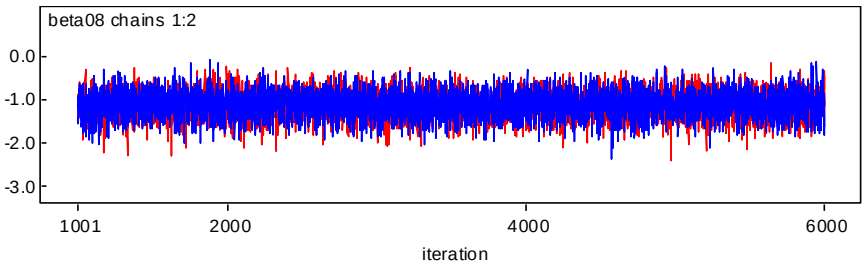
Beta 06: (DUM1_V2)



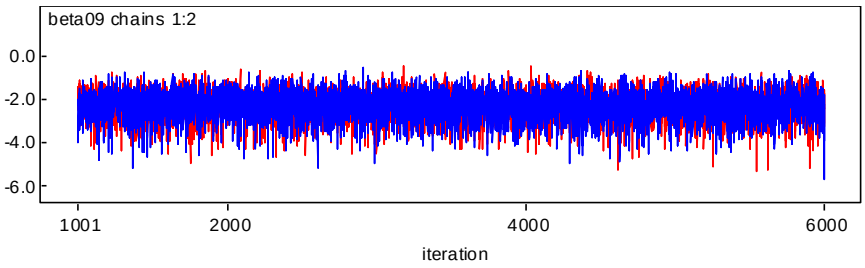
Beta 07: (DUM2_V2)



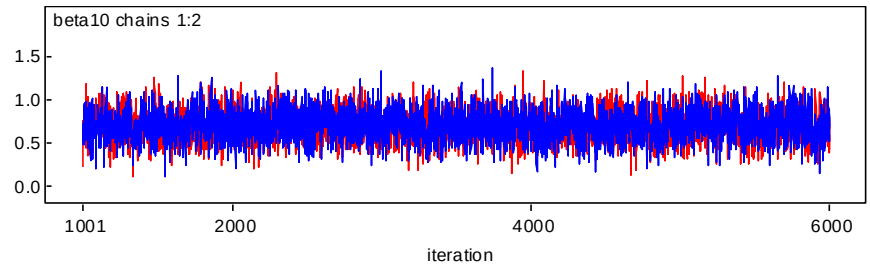
Beta 08: (DUM3_V2)



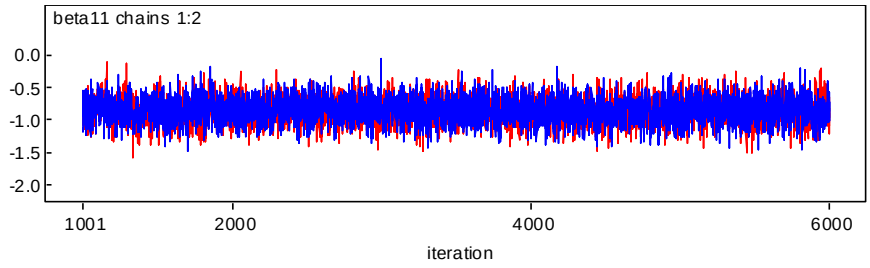
Beta 09: (DUM4_V2)



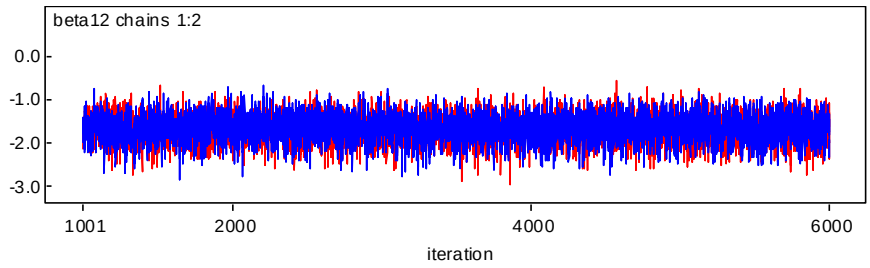
Beta 10: (DUM1_V3)



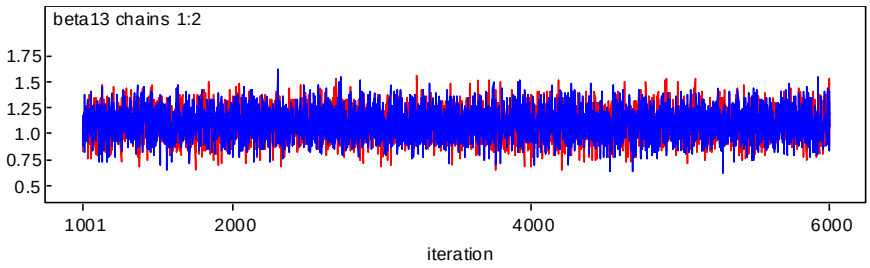
Beta 11: (DUM2_V3)



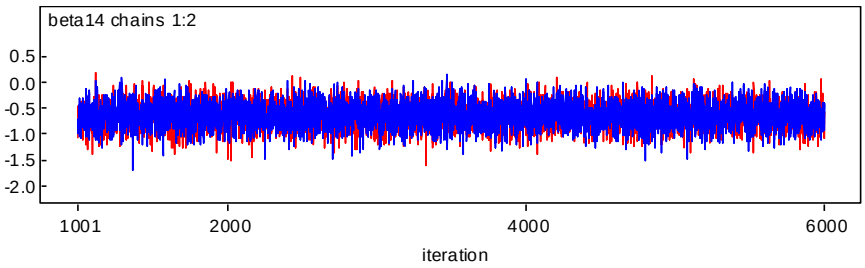
Beta 12: (DUM3_V3)



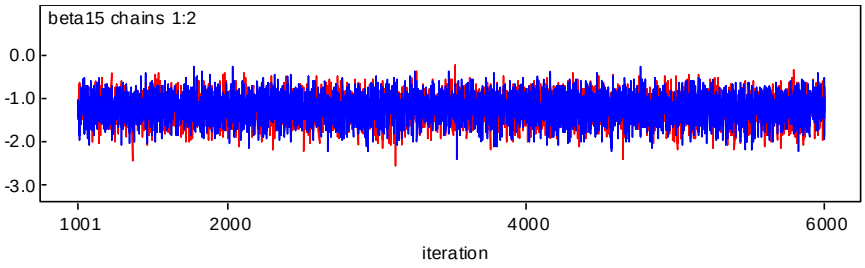
Beta 13: (DUM1_V4)



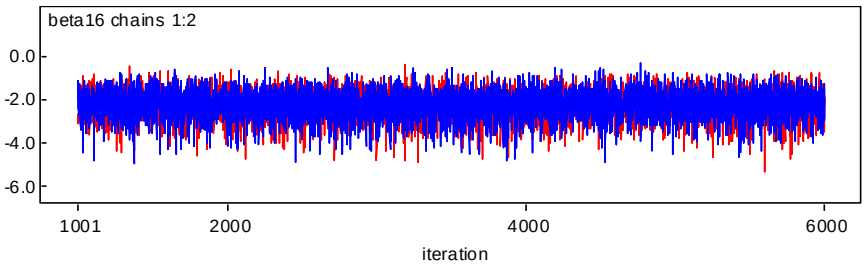
Beta 14: (DUM2_V4)



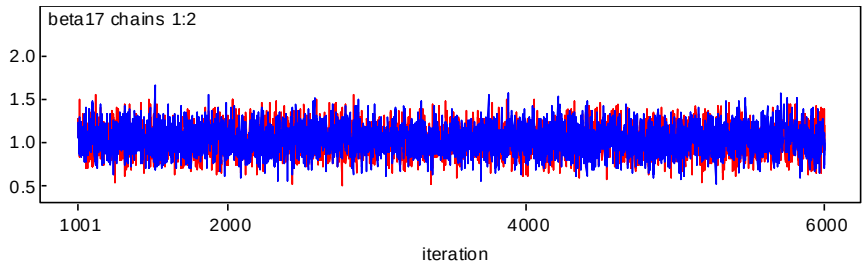
Beta 15: (DUM3_V4)



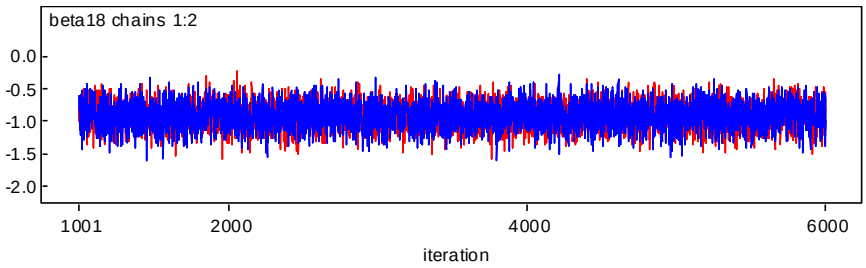
Beta 16: (DUM4_V4)



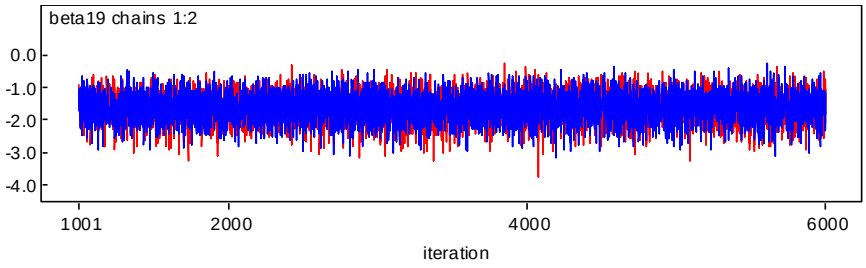
Beta 17: (DUM1_V5)



Beta 18: (DUM2_V5)

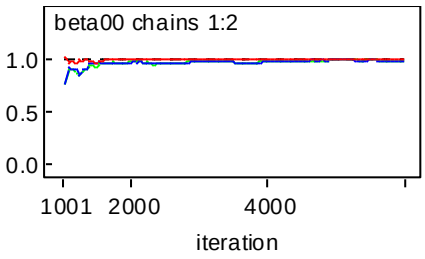


Beta 19: (DUM3_V5)

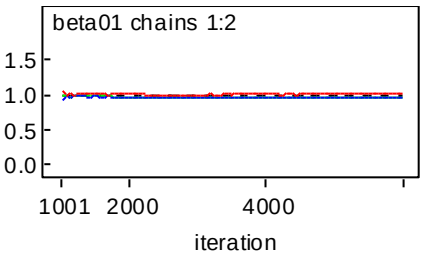


Gráficos de Gelman Rubin para os parâmetros do modelo Bayesiano (capítulo 6)

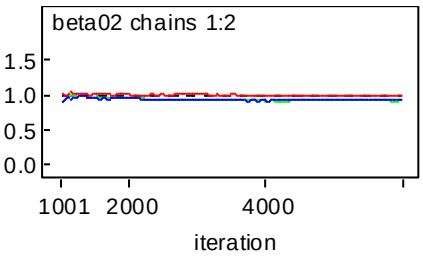
Beta 00: (intercepto)



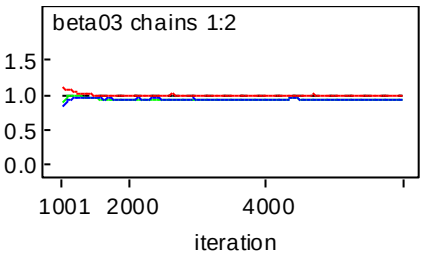
Beta 01: (DUM1_V1)



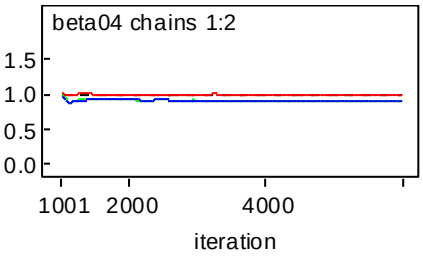
Beta 02: (DUM2_V1)



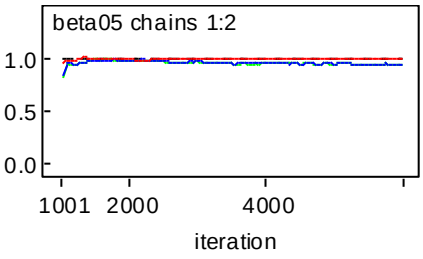
Beta 03: (DUM3_V1)



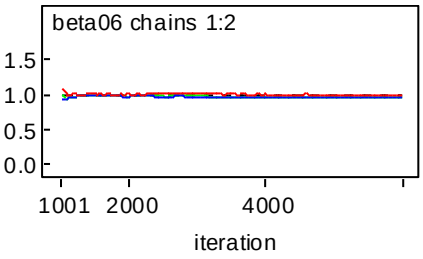
Beta 04: (DUM4_V1)



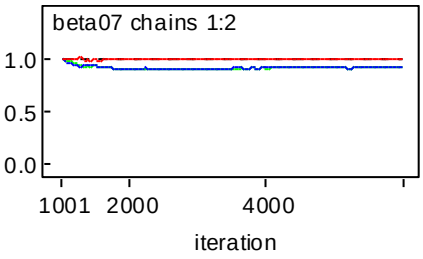
Beta 05: (DUM5_V1)



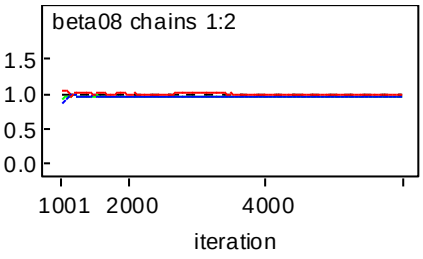
Beta 06: (DUM1_V2)



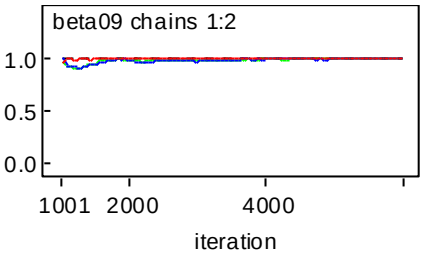
Beta 07: (DUM2_V2)



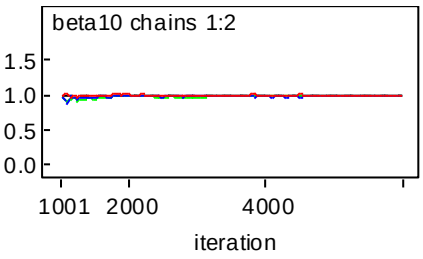
Beta 08: (DUM3_V2)



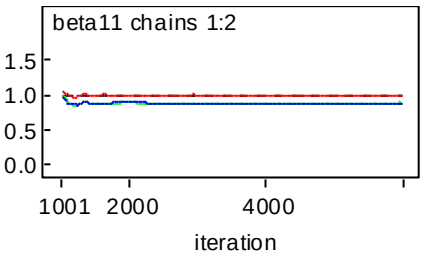
Beta 09: (DUM4_V2)



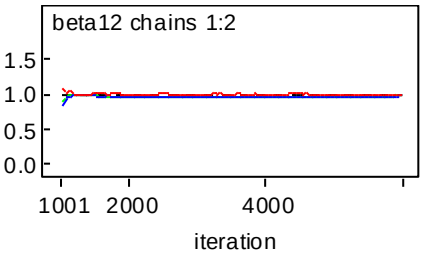
Beta 10: (DUM1_V3)



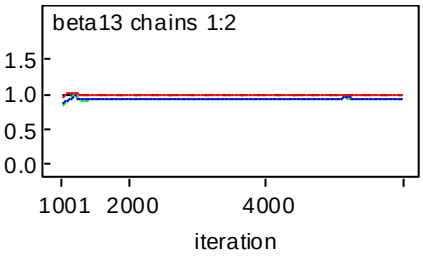
Beta 11: (DUM2_V3)



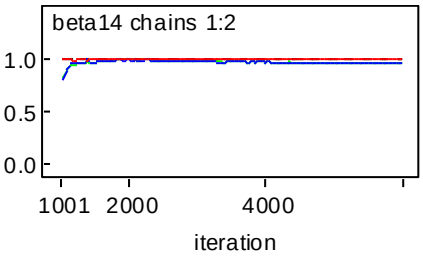
Beta 12: (DUM2_V3)



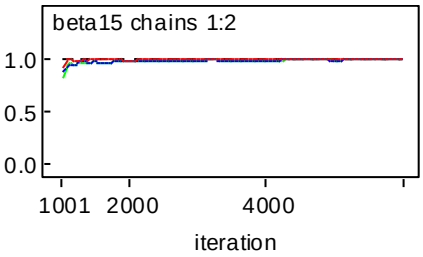
Beta 13: (DUM1_V4)



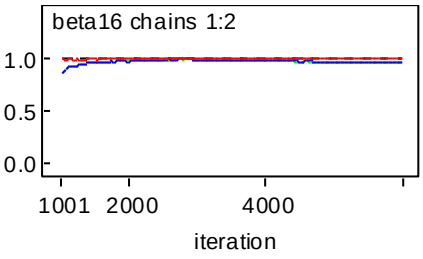
Beta 14: (DUM2_V4)



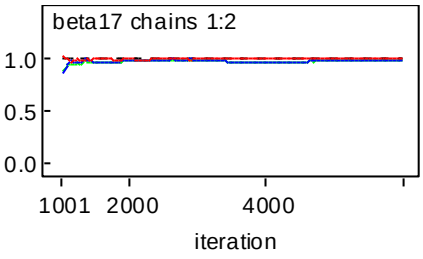
Beta 15: (DUM3_V4)



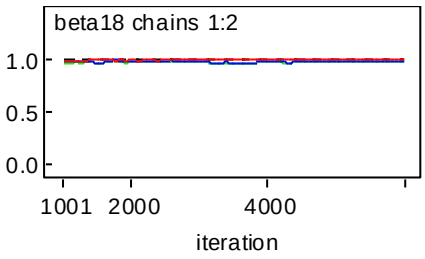
Beta 16: (DUM4_V4)



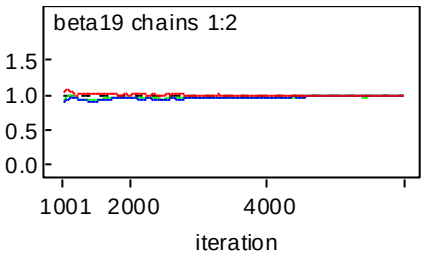
Beta 17: (DUM1_V5)



Beta 18: (DUM2_V5)

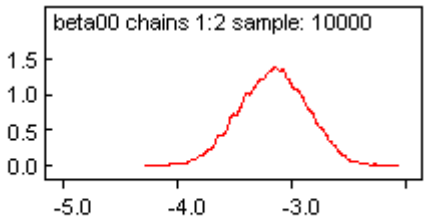


Beta 19: (DUM3_V5)

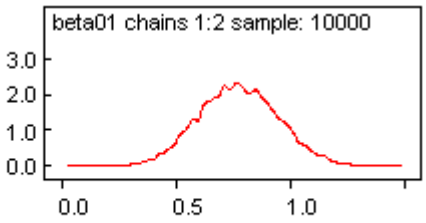


Densidades a posteriori para os parâmetros de interesse do modelo Bayesiano (capítulo 6)

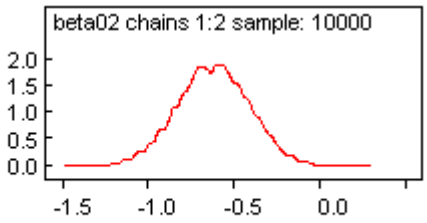
Beta 00: (intercepto)



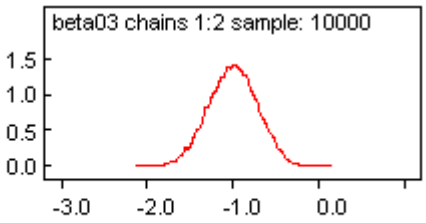
Beta 01: (DUM1_V1)



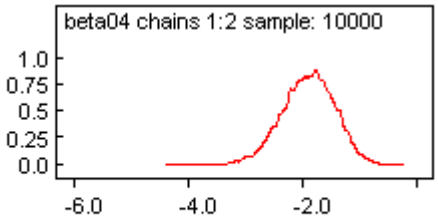
Beta 02: (DUM2_V1)



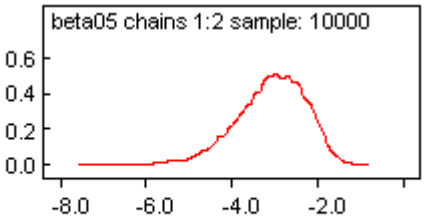
Beta 03: (DUM3_V1)



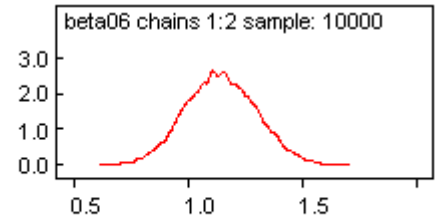
Beta 04: (DUM4_V1)



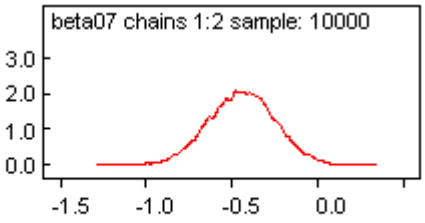
Beta 05: (DUM5_V1)



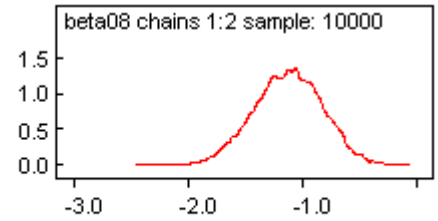
Beta 06: (DUM1_V2)



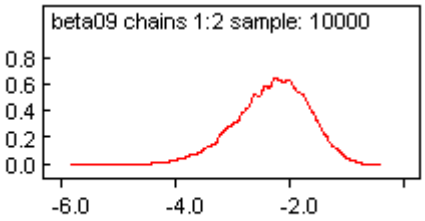
Beta 07: (DUM2_V2)



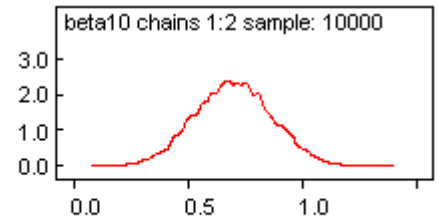
Beta 08: (DUM3_V2)



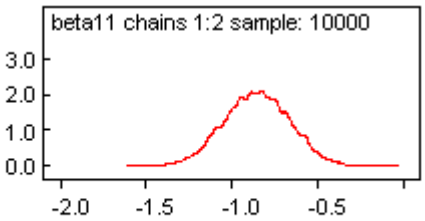
Beta 09: (DUM4_V2)



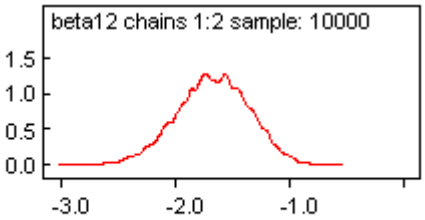
Beta 10: (DUM1_V3)



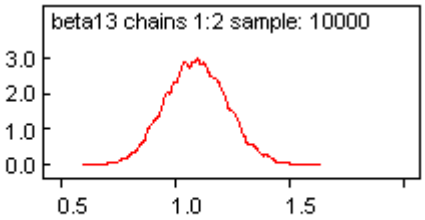
Beta 11: (DUM2_V3)



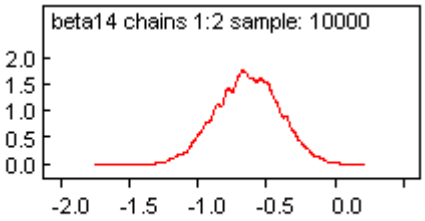
Beta 12: (DUM3_V3)



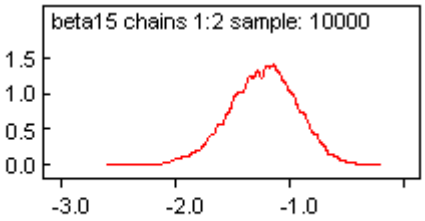
Beta 13: (DUM1_V4)



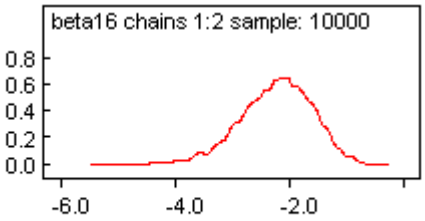
Beta 14: (DUM2_V4)



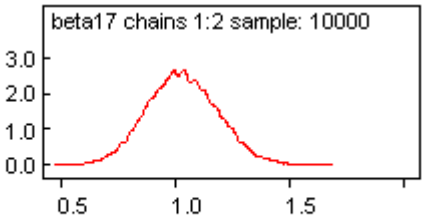
Beta 15: (DUM3_V4)



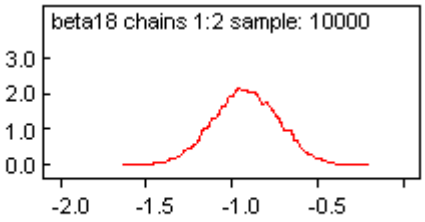
Beta 16: (DUM4_V4)



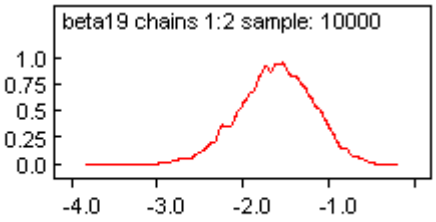
Beta 17: (DUM1_V5)



Beta 18: (DUM2_V5)

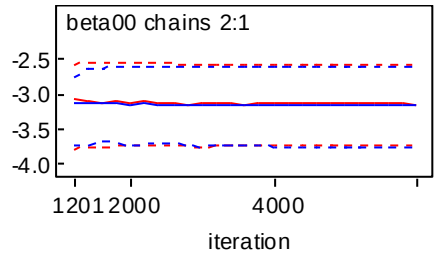


Beta 19: (DUM3_V5)

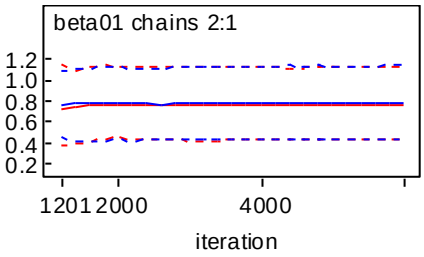


Intervalo de credibilidade para os parâmetros ajustados (capítulo 6)

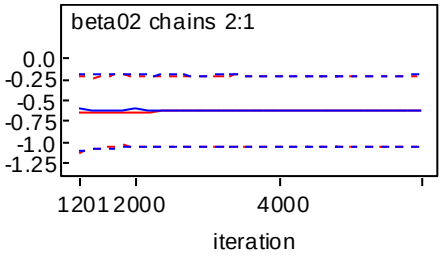
Beta 00: (intercepto)



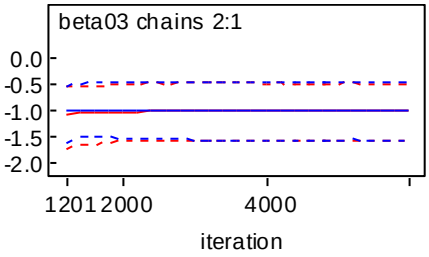
Beta 01: (DUM1_V1)



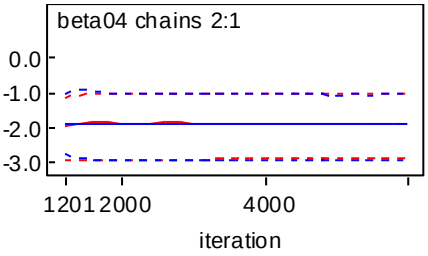
Beta 02: (DUM2_V1)



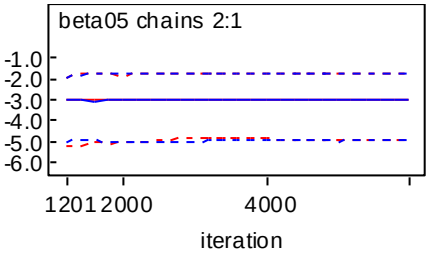
Beta 03: (DUM3_V1)



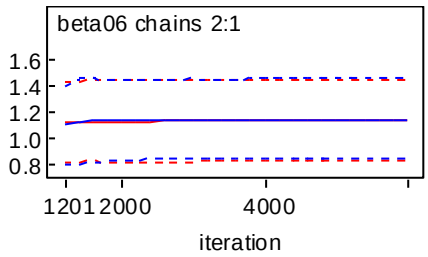
Beta 04: (DUM4_V1)



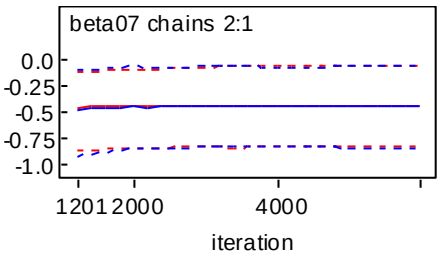
Beta 05: (DUM5_V1)



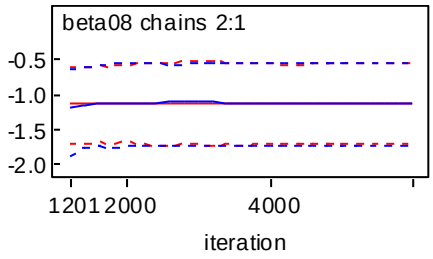
Beta 06: (DUM1_V2)



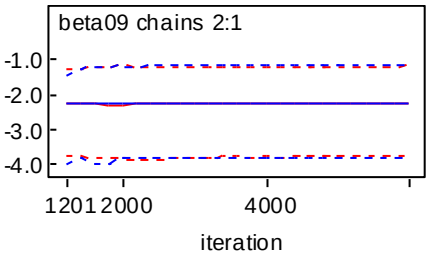
Beta 07: (DUM2_V2)



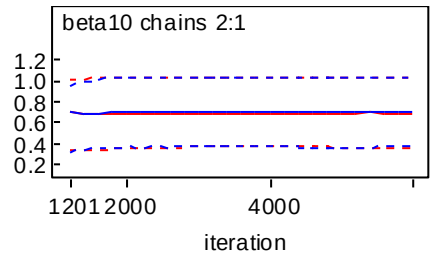
Beta 08: (DUM3_V2)



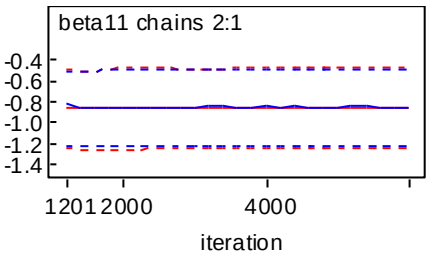
Beta 09: (DUM4_V2)



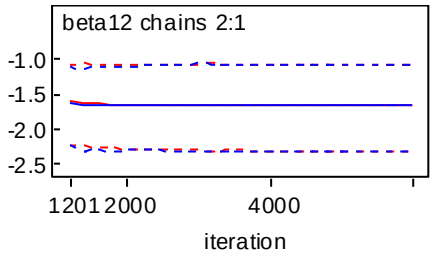
Beta 10: (DUM1_V3)



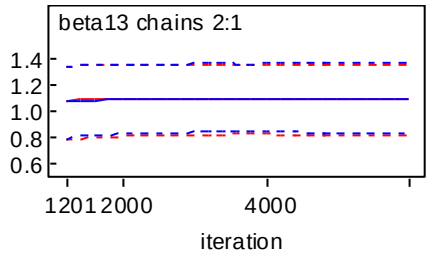
Beta 11: (DUM2_V3)



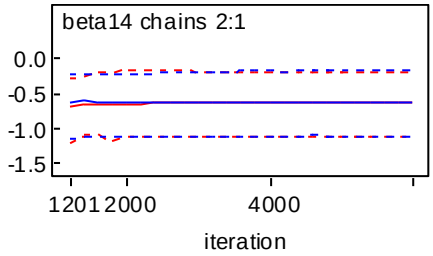
Beta 12: (DUM3_V3)



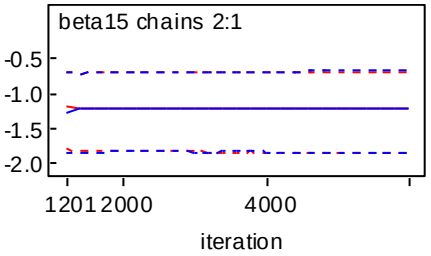
Beta 13: (DUM1_V4)



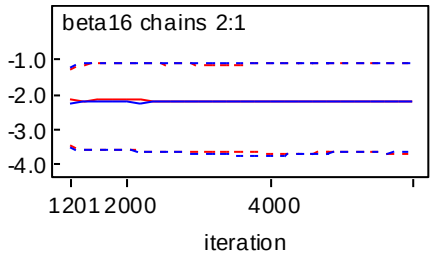
Beta 14: (DUM2_V4)



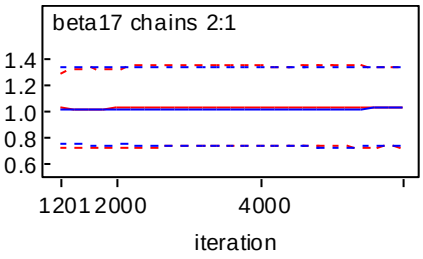
Beta 15: (DUM3_V4)



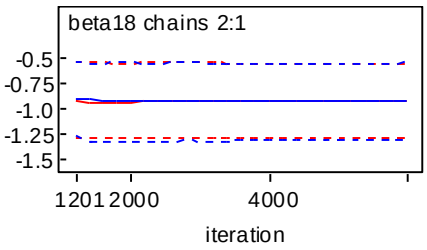
Beta 16: (DUM4_V4)



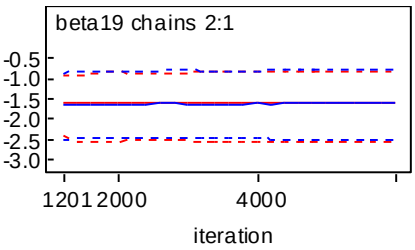
Beta 17: (DUM1_V5)



Beta 18: (DUM2_V5)

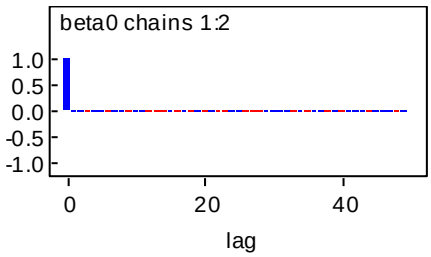


Beta 19: (DUM3_V5)

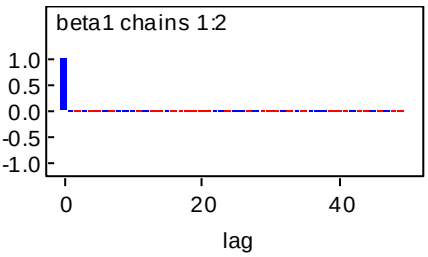


Gráficos de auto-correlação (capítulo 6)

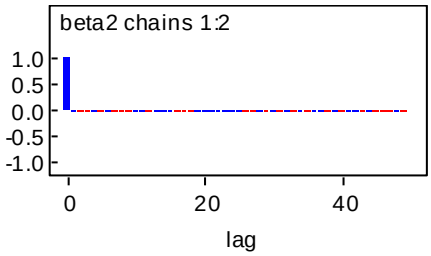
Beta 00: (intercepto)



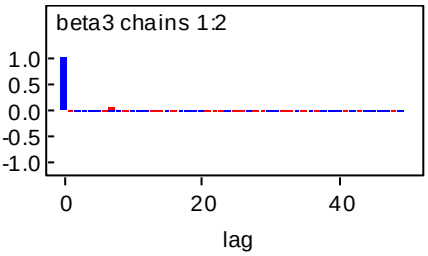
Beta 01: (DUM1_V1)



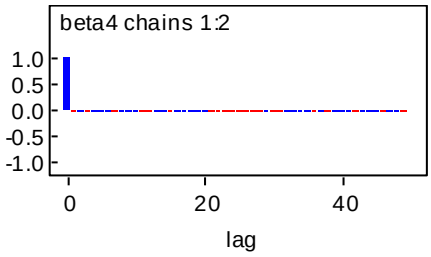
Beta 02: (DUM2_V1)



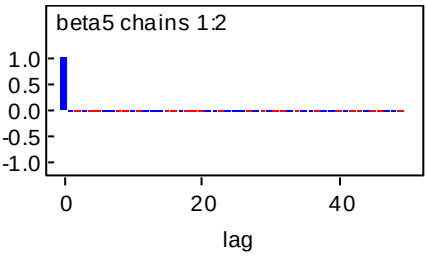
Beta 03: (DUM3_V1)



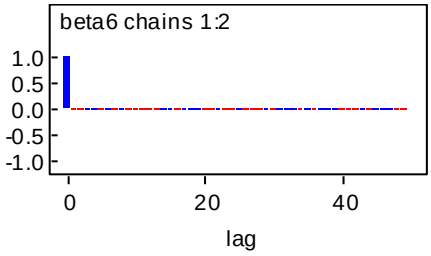
Beta 04: (DUM4_V1)



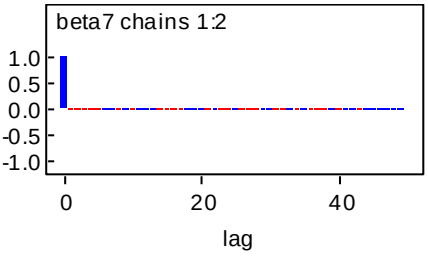
Beta 05: (DUM5_V1)



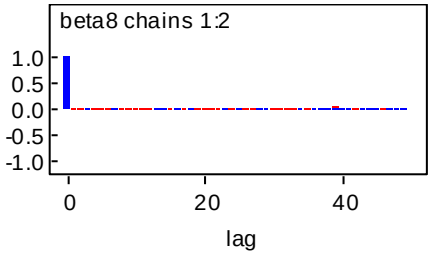
Beta 06: (DUM1_V2)



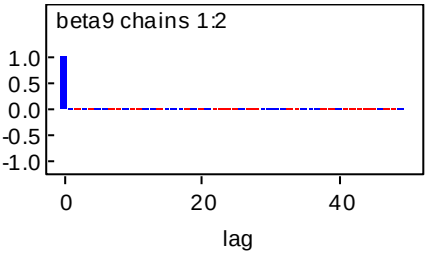
Beta 07: (DUM2_V2)



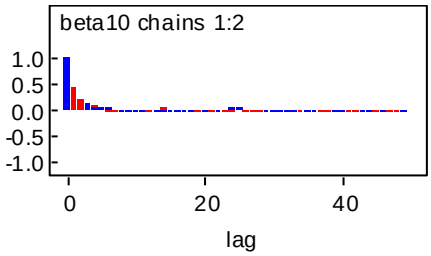
Beta 08: (DUM3_V2)



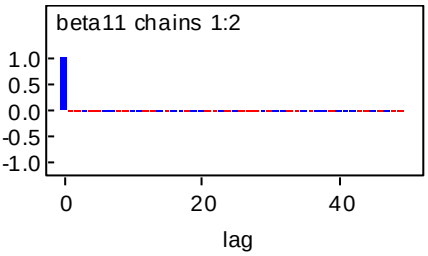
Beta 09: (DUM4_V2)



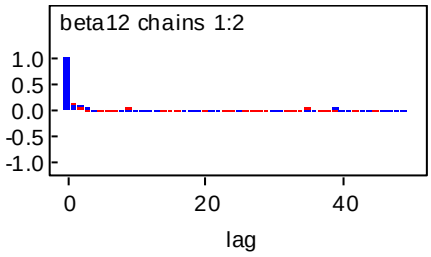
Beta 10: (DUM1_V3)



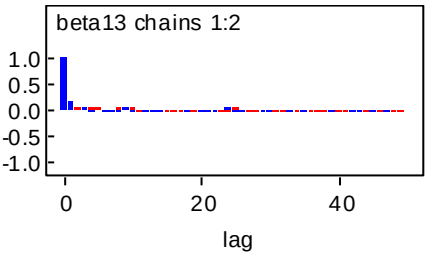
Beta 11: (DUM2_V3)



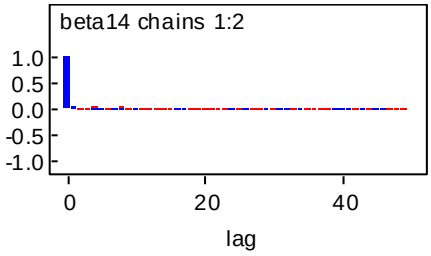
Beta 12: (DUM3_V3)



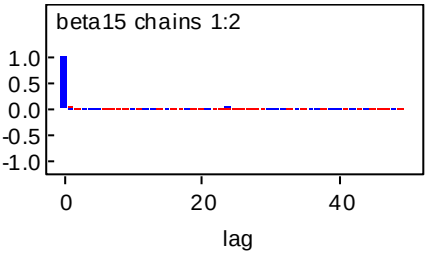
Beta 13: (DUM1_V4)



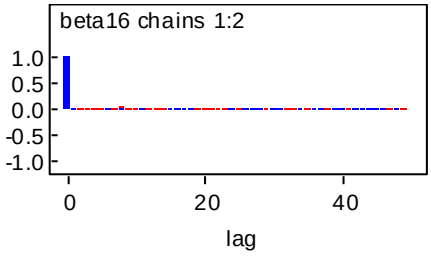
Beta 14: (DUM2_V4)



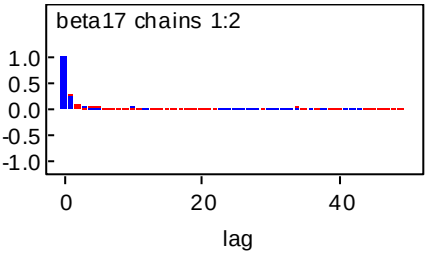
Beta 15: (DUM3_V4)



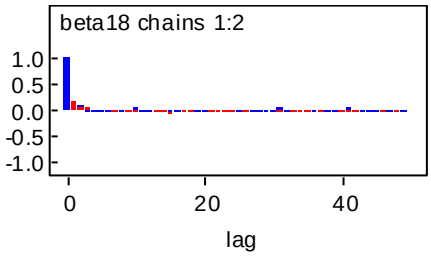
Beta 16: (DUM4_V4)



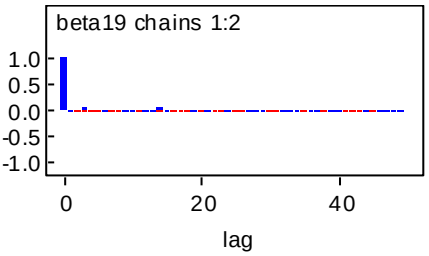
Beta 17: (DUM1_V5)



Beta 18: (DUM2_V5)



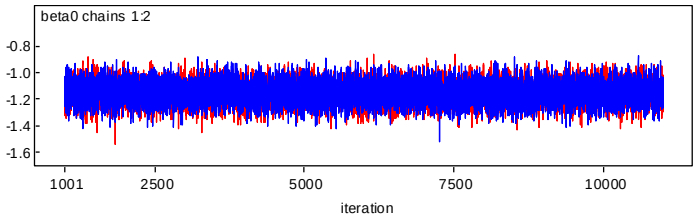
Beta 19: (DUM3_V5)



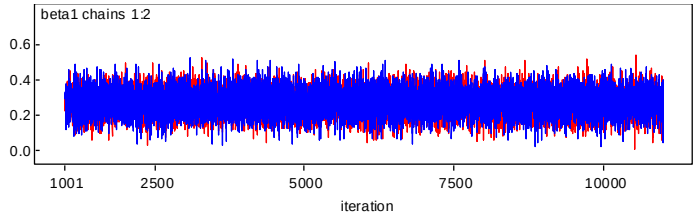
Histórico das cadeias geradas para os dados do capítulo 7:

Histórico das cadeias de Markov para os parâmetros do modelo Bayesiano

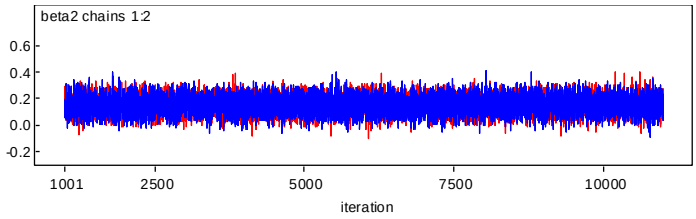
Beta 0: (Intercepto)



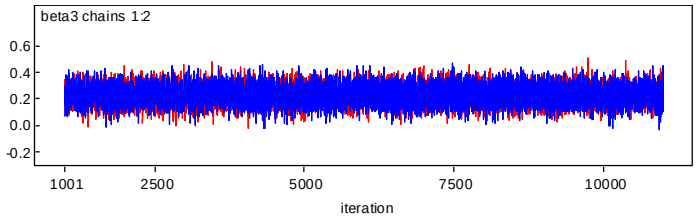
Beta 1: (D_TPCLIENT)



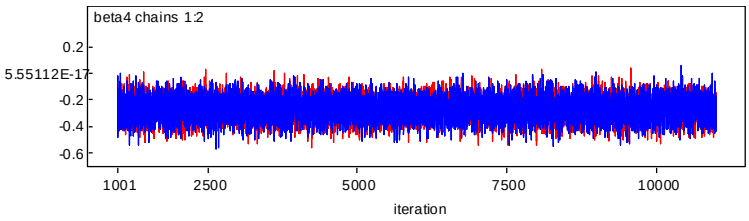
Beta 2: (D_SEXO)



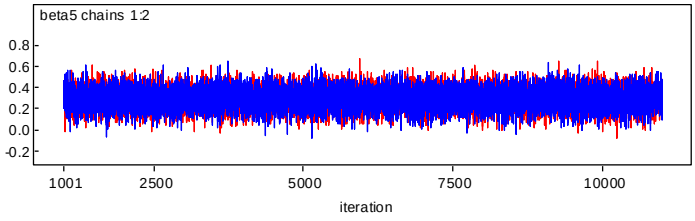
Beta 3: (D_LIMITE)



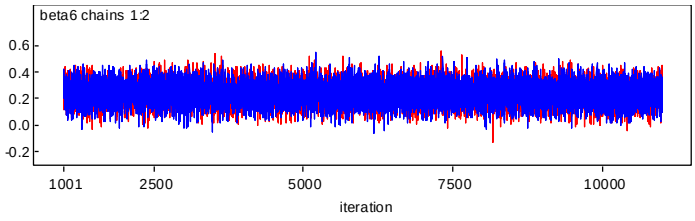
Beta 4: (D1_CEP)



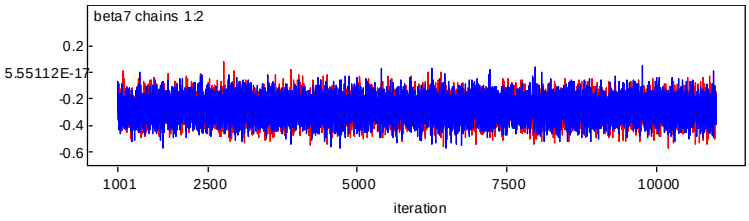
Beta 5: (D2_CEP)



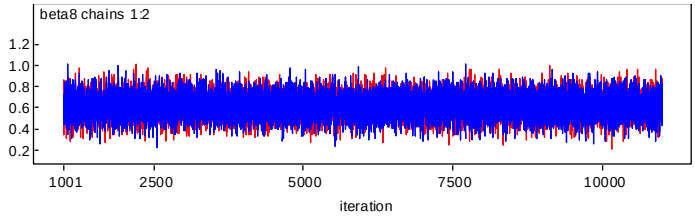
Beta 6: (D1_TPEMPREG)



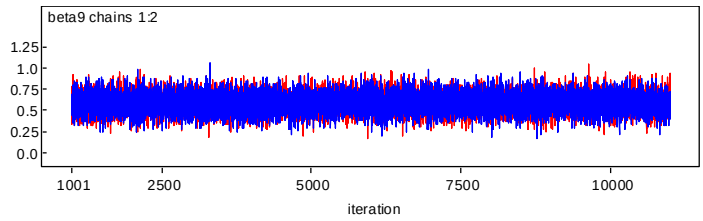
Beta 7: (D2_TPEMPREG)



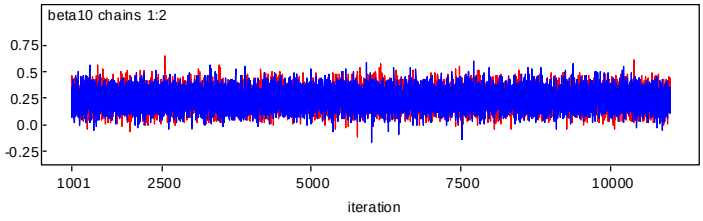
Beta 8: (D1_AGE)



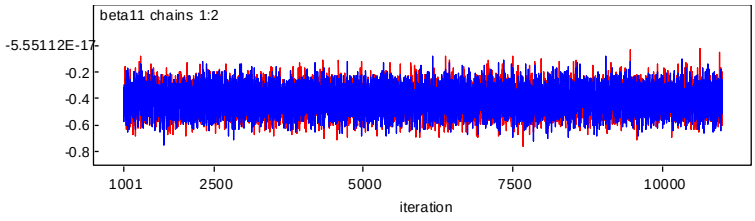
Beta 9: (D2_AGE)



Beta 10: (D3_AGE)

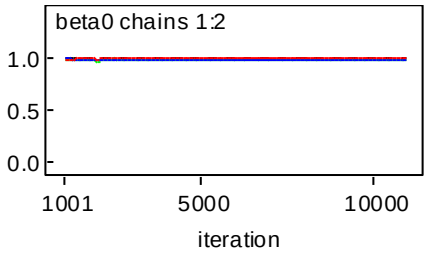


Beta 11: (D4_AGE)

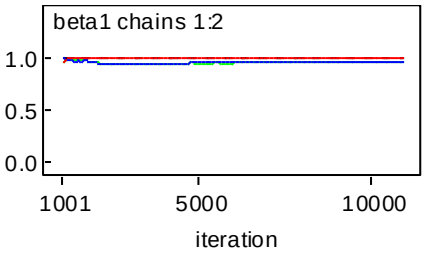


Gráficos de Gelman Rubin para os parâmetros do modelo Bayesiano

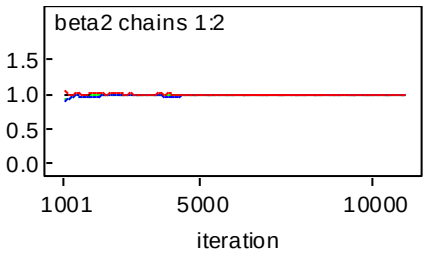
Beta 0: (Intercepto)



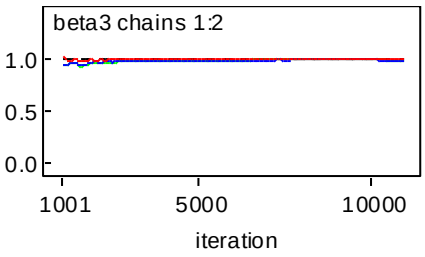
Beta 1: (D_TPCLIENT)



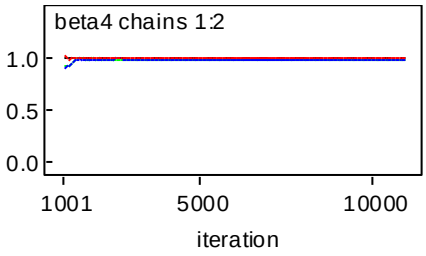
Beta 2: (D_SEXO)



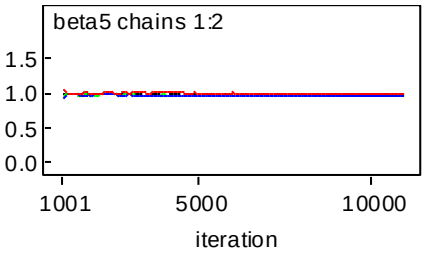
Beta 3: (D_LIMITE)



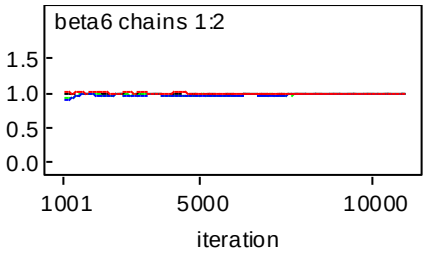
Beta 4: (D1_CEP)



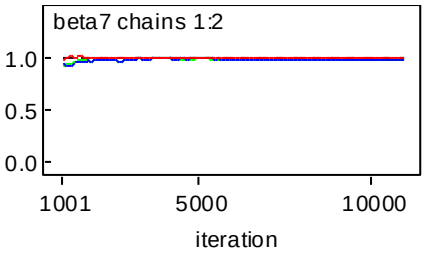
Beta 5: (D2_CEP)



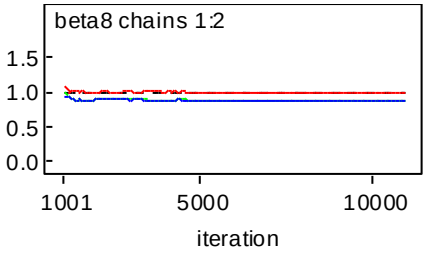
Beta 6: (D1_TPEMPREG)



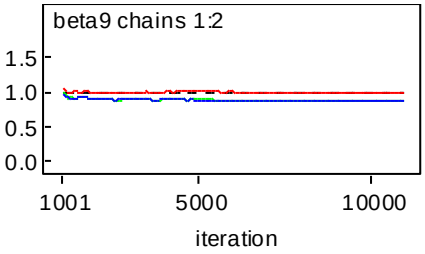
Beta 7: (D2_TPEMPREG)



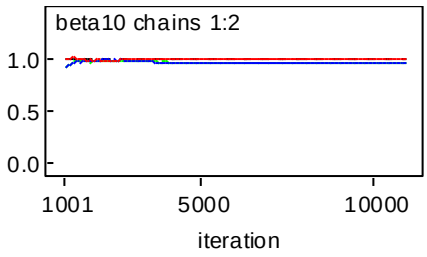
Beta 8: (D1_AGE)



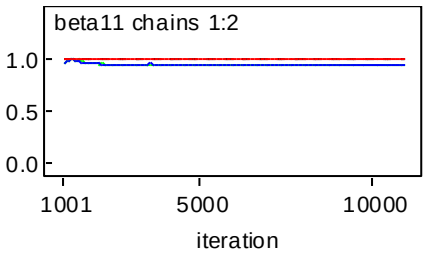
Beta 9: (D2_AGE)



Beta 10: (D3_AGE)

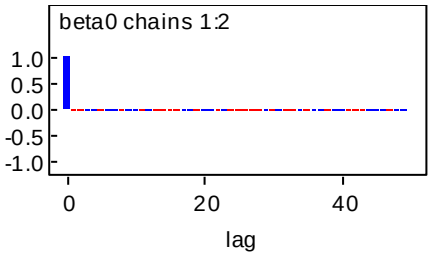


Beta 11: (D4_AGE)

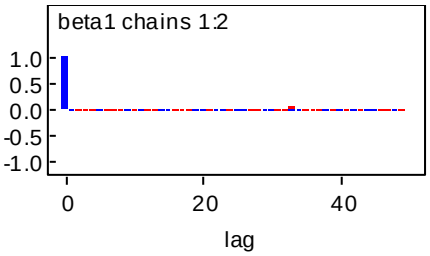


Autocorrelação para as cadeias dos parâmetros de interesse para o modelo Bayesiano

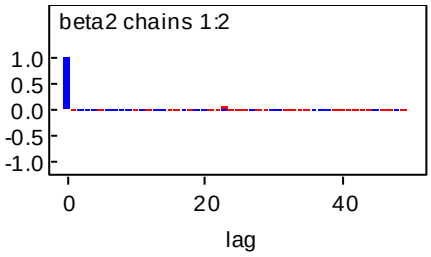
Beta 0: (Intercepto)



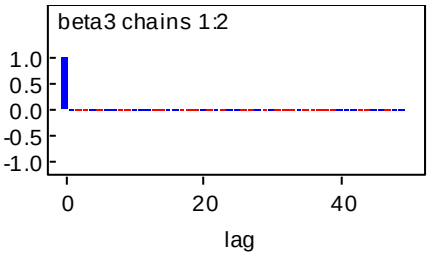
Beta 1: (D_TPCLIENT)



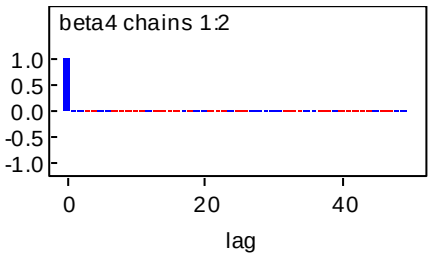
Beta 2: (D_SEXO)



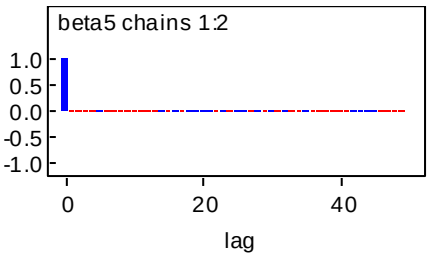
Beta 3: (D_LIMITE)



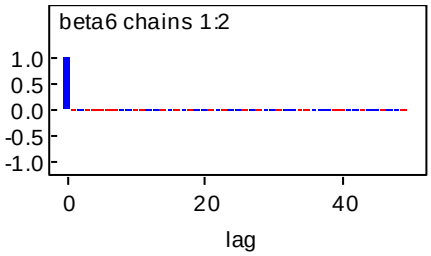
Beta 4: (D1_CEP)



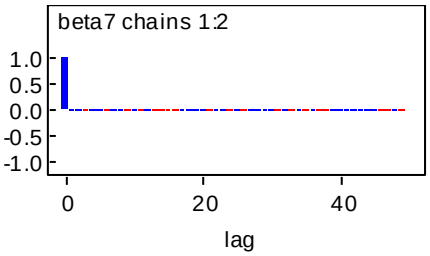
Beta 5: (D2_CEP)



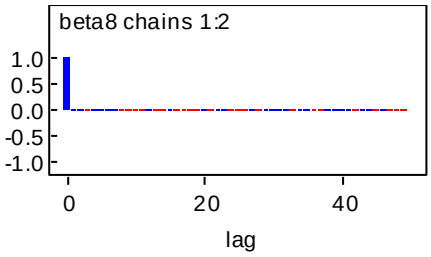
Beta 6: (D1_TPEMPREG)



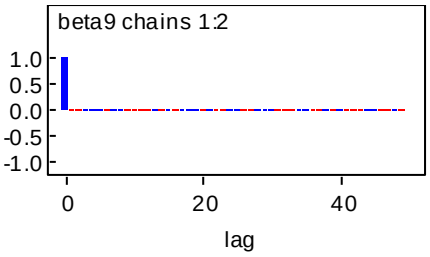
Beta 7: (D2_TPEMPREG)



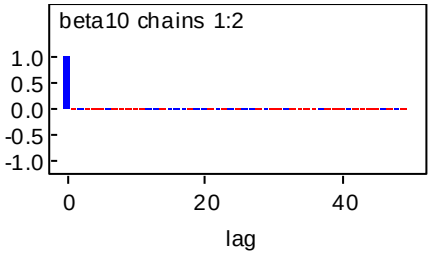
Beta 8: (D1_AGE)



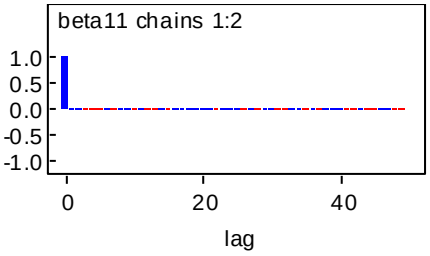
Beta 9: (D2_AGE)



Beta 10: (D3_AGE)

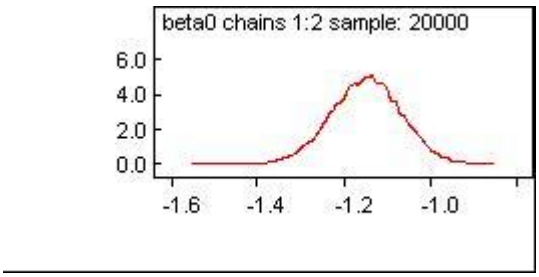


Beta 11: (D4_AGE)

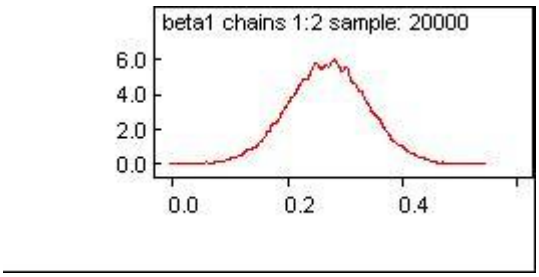


Densidades *a posteriori* para os parâmetros de interesse para o modelo Bayesiano

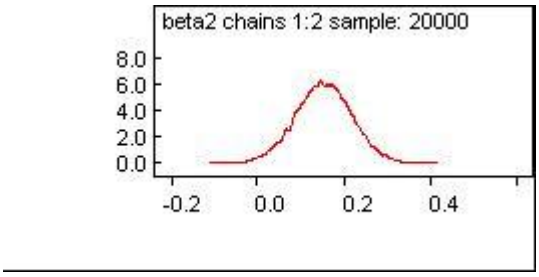
Beta 0: (Intercepto)



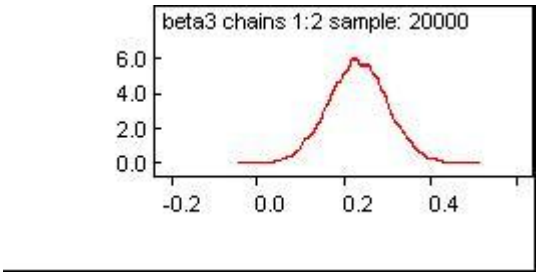
Beta 1: (D_TPCLIENT)



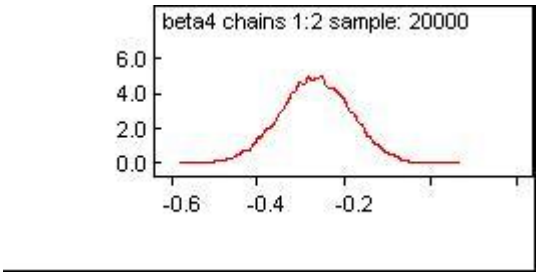
Beta 2: (D_SEXO)



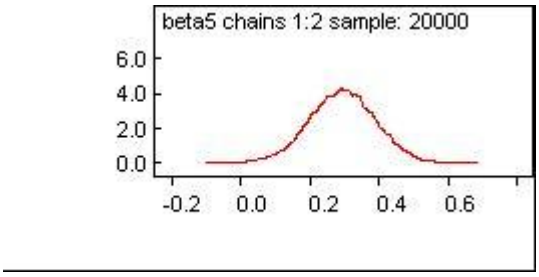
Beta 3: (D_LIMITE)



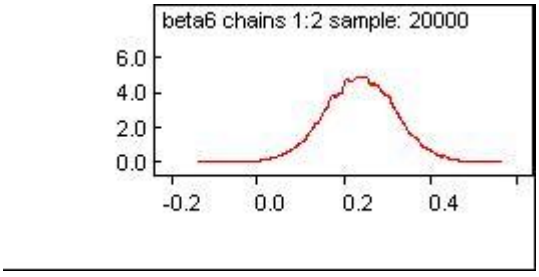
Beta 4: (D1_CEP)



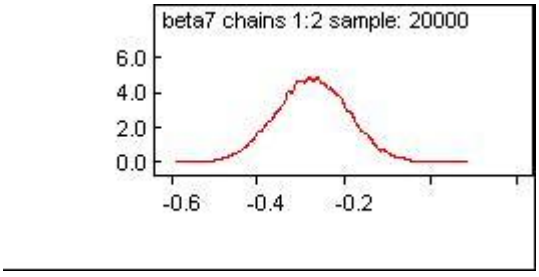
Beta 5: (D2_CEP)



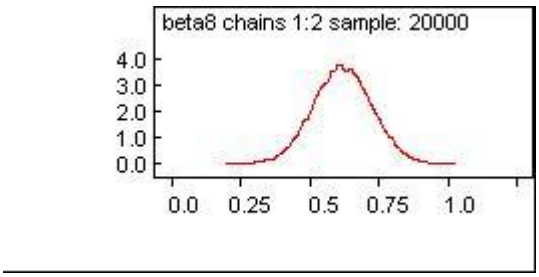
Beta 6: (D1_TPEMPREG)



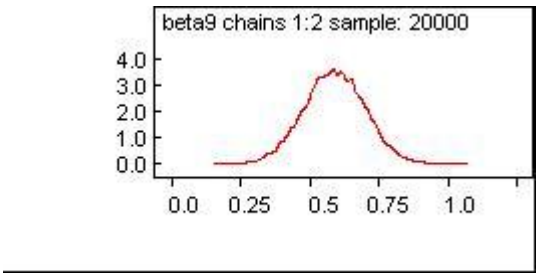
Beta 7: (D2_TPEMPREG)



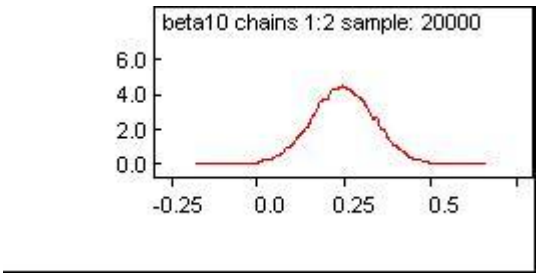
Beta 8: (D1_AGE)



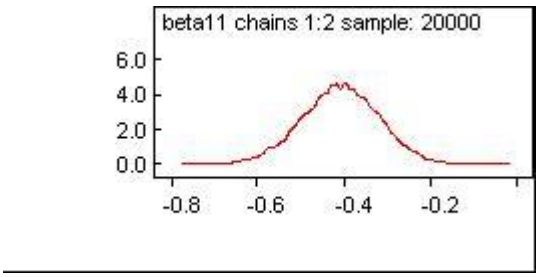
Beta 9: (D2_AGE)



Beta 10: (D3_AGE)



Beta 11: (D4_AGE)



Códigos utilizados:

Regressão Logística Clássica – SAS

Ajuste do modelo:

```
/* Um modelo de regressão logística por variável disponível */

%let x1 = CAT_TPCLIENT;
%let x2 = CAT_SEXO;
%let x3 = CAT_ESTCIVIL;
%let x4 = CAT_SITRESID;
%let x5 = CAT_CEP;
%let x6 = CAT_TPEMPREG;
%let x7 = CAT_LIMITE;
%let x8 = CAT_TEMPORES; * Retirada após o ajuste univariado;
%let x9 = CAT_AGE_02;

PROC LOGISTIC DATA=FINA.AJUSTE_70 DESCENDING;
CLASS &x1 (REF="C0") /PARAM=REF;
MODEL MAU = &x1;
RUN;
PROC LOGISTIC DATA=FINA.AJUSTE_70 DESCENDING;
CLASS &x2 (REF="C0") /PARAM=REF;
MODEL MAU = &x2;
RUN;
PROC LOGISTIC DATA=FINA.AJUSTE_70 DESCENDING;
CLASS &x3 (REF="C0") /PARAM=REF;
MODEL MAU = &x3;
RUN;
PROC LOGISTIC DATA=FINA.AJUSTE_70 DESCENDING;
CLASS &x4 (REF="C0") /PARAM=REF;
MODEL MAU = &x4;RUN;
PROC LOGISTIC DATA=FINA.AJUSTE_70 DESCENDING;
CLASS &x5 (REF="C0") /PARAM=REF;
MODEL MAU = &x5;
RUN;
PROC LOGISTIC DATA=FINA.AJUSTE_70 DESCENDING;
CLASS &x6 (REF="C0") /PARAM=REF;
MODEL MAU = &x6;
RUN;
PROC LOGISTIC DATA=FINA.AJUSTE_70 DESCENDING;
CLASS &x7 (REF="C0") /PARAM=REF;
MODEL MAU = &x7;
RUN;
PROC LOGISTIC DATA=FINA.AJUSTE_70 DESCENDING;
CLASS &x8 (REF="C0") /PARAM=REF;
MODEL MAU = &x8;
```

```
RUN;
PROC LOGISTIC DATA=FINA.AJUSTE_70 DESCENDING;
CLASS &x9 (REF="C0") /PARAM=REF;
MODEL MAU = &x9;
RUN;

/* Métodos de seleção */

%MACRO METODO_SELECAO(METODO);
PROC LOGISTIC DATA=FINA.AJUSTE_70 DESCENDING;
CLASS
CAT_TPCLIENT  (REF="C0")
CAT_SEXO      (REF="C0")
CAT_ESTCIVIL  (REF="C0")
CAT_SITRESID  (REF="C0")
CAT_CEP       (REF="C0")
CAT_TPEMPREG  (REF="C0")
CAT_LIMITE    (REF="C0")
CAT_AGE_02    (REF="C0")
/PARAM=REF;
MODEL MAU = CAT_TPCLIENT CAT_SEXO CAT_ESTCIVIL CAT_SITRESID CAT_CEP
            CAT_TPEMPREG CAT_LIMITE CAT_AGE_02
      /OUTROC=DADOS_ROC SELECTION=&METODO sls=0.05 sle=0.05 LACKFIT;
OUTPUT OUT=AJUSTE_MOD_&METODO PREDICTED=PROB_EV;RUN;

* ANÁLISE DO KS ;
PROC NPAR1WAY DATA=AJUSTE_MOD_&METODO EDF;
      CLASS MAU;
      VAR PROB_EV;
RUN;
%MEND;
%METODO_SELECAO(STEPWISE);
%METODO_SELECAO(FORWARD);
%METODO_SELECAO(BACKWARD);
/* Ajuste manual */
PROC LOGISTIC DATA=FINA.AJUSTE_70 DESCENDING;
CLASS
  CAT_TPCLIENT  (REF="C0")
  CAT_SEXO      (REF="C0")
/*CAT_ESTCIVIL  (REF="C0")*/
/*CAT_SITRESID  (REF="C0")*/
  CAT_CEP       (REF="C0")
  CAT_TPEMPREG  (REF="C0")
  CAT_LIMITE    (REF="C0")
  CAT_AGE_02    (REF="C0") /PARAM=REF;
MODEL MAU = CAT_TPCLIENT
            CAT_SEXO
/*          CAT_ESTCIVIL*/
/*          CAT_SITRESID*/
```

```
            CAT_CEP
            CAT_TPEMPREG
            CAT_LIMITE
            CAT_AGE_02
                        /OUTROC=DADOS_ROC LACKFIT;
OUTPUT OUT=AJUSTE_MOD PREDICTED=PROB_EV;
RUN;

* ANÁLISE DO KS ;
PROC NPAR1WAY DATA=AJUSTE_MOD EDF;
      CLASS MAU;
      VAR PROB_EV;
RUN;

* Curva ROC;
goptions ftext=SWISS ctext=BLACK htext=1 cells gunit=pct htitle=6;
         symbol color=BLUE i=join v=SQUARE height=1 width=1;
         axis1 order=(0 to 1 by .1) width=1;
         axis2 order=(0 to 1 by .1) width=1;

proc gplot data=DADOS_ROC;
   plot _sensit_ * _1mspec_ / vminor=0 hminor=0
   vaxis=axis1 haxis=axis2 cframe=CXF7E1C2 caxis=BLACK
   name='roc' description='ROC curve';
run;
quit;

goptions reset=symbol ftext= ctext= htext=;
goptions reset=all device=WIN;
```

Regressão Logística Bayesiana – WinBUGS

```
model{
  for(i in 1:N){
     y[i]~dbern(p[i])
        logit(p[i])<-beta0+beta1*x1[i]+beta2*x2[i]+beta3*x3[i]+
                     beta4*x4[i]+beta5*x5[i]+beta6*x6[i]+beta7*x7[i]+
                     beta8*x8[i]+beta9*x9[i]+beta10*x10[i]+
                     beta11*x11[i]
              }
          beta0   ~ dflat()
          beta1   ~ dflat()
          beta2   ~ dflat()
          beta3   ~ dflat()
          beta4   ~ dflat()
```

```
          beta5   ~ dflat()
          beta6   ~ dflat()
          beta7   ~ dflat()
          beta8   ~ dflat()
          beta9   ~ dflat()
          beta10  ~ dflat()
          beta11  ~ dflat()
    }
list(y=c(DADOS),
    x1=c(DADOS),
    x2=c(DADOS),
    x3=c(DADOS),
    x4=c(DADOS),
    x5=c(DADOS),
    x6=c(DADOS),
    x7=c(DADOS),
    x8=c(DADOS),
    x9=c(DADOS),
   x10=c(DADOS),
   x11=c(DADOS),
         N=5124)


list(beta0=0, beta1=0, beta2=0, beta3=0,  beta4=0,  beta5=0,
     beta6=0, beta7=0, beta8=0, beta9=0, beta10=0, beta11=0)
list(beta0=8, beta1=8, beta2=8, beta3=8,  beta4=8,  beta5=8,
     beta6=8, beta7=8, beta8=8, beta9=8, beta10=8, beta11=8)
```

Redes Neurais Artificiais – SAS

a) Um único neurônio:

```
proc neural data=EMDATA.DMDB6PG7 dmdbcat=EMPROJ.DMDB6PG7
network=EMPROJ.NNS_GXVJ.NETWORK
predata=EMDATA.NNDQD2JF;
netoptions plot;
decision decisiondata= EMPROJ.MAU2_ decvars=_DEC1;
*;
initial inest= EMPROJ.NNXFZQH2 BYLABEL;
*;
```

```
prelim 1
outest= EMPROJ.NNP6QZSP;
*;
train
outest= EMPROJ.NNEB7LUZ estiter=1
outfit= EMPROJ.NNFH91PU;
*;
code noerror nores metabase=EMPROJ.NNS_GXVJ.DATASTEP.SOURCE;
*;
RUN;
```

b) MLP com uma camada intermediária:

```
proc neural data=EMDATA.DMDBVCR2 dmdbcat=EMPROJ.DMDBVCR2
network=EMPROJ.NNS_ESI3.NETWORK;
netoptions plot;
decision decisiondata= EMPROJ.MAU3_ decvars=
_DEC1;
*;
train
outest= EMPROJ.NNEJPQEK estiter=1
outfit= EMPROJ.NNFGKISJ;
*;
code noerror nores metabase=EMPROJ.NNS_ESI3.DATASTEP.SOURCE;
*;
RUN;
```

c) MLP com duas camadas intermediárias:

```
proc neural data=EMDATA.DMDB4JO6 dmdbcat=EMPROJ.DMDB4JO6
network=EMPROJ.NNS_2WW7.NETWORK
predata=EMDATA.NND9SGBP;
netoptions plot;
decision decisiondata= EMPROJ.MAU6_ decvars=_DEC1;
*;
prelim 10
outest= EMPROJ.NNP07Y9I;
*;
train
outest= EMPROJ.NNEXBNE4 estiter=1
outfit= EMPROJ.NNFVYFG3;
*;
code noerror nores metabase=EMPROJ.NNS_2WW7.DATASTEP.SOURCE;
*;
RUN;
```

Geração do banco de dados artificial – SAS

```
%macro mvn(version,
           varcov=,
           means=,
           n=,
           seed=0,
           sample=);


%if &version ne %then %put MVN macro Version 1.0;

data _null_;
   if &seed le 0 then do;
      seed = int(time());   /* get clock time in integer seconds */
      put seed=;
      call symput('seed',seed);   /* store seed as macro variable */
   end;
run;


proc iml worksize=100;
   use &varcov;            /* read variance-covariance matrix */
   read all into cov;
   use &means;             /* read means */
   read all into mu;
   v=nrow(cov);            /* calculate number of variables */
   n=&n;
   seed = &seed;
   l=t(root(cov));         /* calculate cholesky root of cov matrix */
   z=normal(j(v,&n,&seed));/* generate nvars*samplesize normals */
   x=l*z;                  /* premultiply by cholesky root */
   x=repeat(mu,1,&n)+x;    /* add in the means */
   tx=t(x);
   create &sample from tx;  /* write out sample data to sas dataset */
   append from tx;
quit;

%mend mvn;

libname results "C:\SAS" ;
libname set1 'C:\SAS\libSAS';

%let repMC=1;
%let ntreino=5000;
%let nteste=150;
%let tx_def=10/100;   * taxa de default;
%let tx_ndef=90/100;  * taxa de não default;
/* Criando matriz de covariância unitária*/
```

```
proc iml;
      varcov=i(20);
      create varcov from varcov;
  append from varcov;
quit;
data varcov;
      set varcov(rename=(col1-col20=m1-m20));
run;

/* Criando vetor de médias*/
/* população default*/

data medias_default;
   do i=1 to 20;
      m1=-2/sqrt(20);
      output;
   end;
drop i;
run;

data medias_naodefault;
   do i=1 to 20;
      m1=2/sqrt(20);
      output;
   end;
 drop i;
run;

/* Gerando dados*/

/*Macro para normal multivariada*/
%inc "'C:\SAS\libSAS\mvn.sas";

/* Amostra de TREINO */

/* treino default */
%let seed=242;
%mvn(seed=&seed,
             varcov=varcov,
     means=medias_default,
     n=%eval(&repMC*&ntreino*&tx_def),
     sample=set1.treino_default);

data x;
      do repMC=1 to &repMC;
            do i=1 to %eval(&ntreino*&tx_def);
            output;
            end;
```

```
      drop i;
      end;
run;

data set1.treino_default;
      retain repMC default;
      set x;
      set set1.treino_default(rename=(COL1-COL20=V1-V20));
      default=1;
run;

/* treino NÃO default*/
options mprint;
%let seed=1655;
%mvn(seed=&seed,
             varcov=varcov,
     means=medias_naodefault,
     n=%eval(&repMC*(&ntreino)*&tx_ndef),
     sample=set1.treino_naodefault);

data x;
      do repMC=1 to &repMC;
            do i=1 to %eval(&ntreino*&tx_ndef);
            output;
            end;
      drop i;
      end;
run;

data set1.treino_naodefault;
      retain repMC default;
      set x;
      set set1.treino_naodefault(rename=(COL1-COL20=V1-V20));
      default=0;
run;

data set1.s1_trn;
      set set1.treino_default set1.treino_naodefault;
run;

proc sort   data=set1.s1_trn;
      by repMC;
run;

/*Marcando ID*/

data ID_treino(drop=repMC);
      do repMC=1 to &repMC;
            do IDtreino=1 to &ntreino;
```

```
            output;
            end;
end;
run;

data set1.s1_trn;
      retain repMC IDtreino default;
      set set1.s1_trn ;
      set ID_treino;
run;

proc freq data=set1.s1_trn;
      table repMC*default/missing nocol nofreq nopercent;
run;

/*          Amostra de TESTE*/

/* teste default*/

%let seed=423;
%mvn(seed=&seed,
             varcov=varcov,
     means=medias_default,
     n=%eval(&repMC*(&nteste)*&tx_def),
     sample=set1.teste_default);

/*proc corr data=set1.teste_default noprob cov nocorr;*/
/*run;*/

data x;
      do repMC=1 to &repMC;
            do i=1 to %eval(&nteste*&tx_def);
            output;
            end;
      drop i;
      end;
run;

data set1.teste_default;
      retain repMC default;
      set x;
      set set1.teste_default(rename=(COL1-COL20=V1-V20));
      default=1;
run;

/*teste NÃO default*/
%let seed=1655;
%mvn(seed=&seed,
             varcov=varcov,
```

```
     means=medias_naodefault,
     n=%eval(&repMC*(&nteste)*&tx_ndef),
     sample=set1.teste_naodefault);

data x;
      do repMC=1 to &repMC;
            do i=1 to %eval(&nteste*&tx_ndef);
            output;
            end;
      drop i;
      end;
run;

data set1.teste_naodefault;
      retain repMC default;
      set x;
      set set1.teste_naodefault(rename=(COL1-COL20=V1-V20));
      default=0;
run;

data set1.s1_teste;
      set set1.teste_default set1.teste_naodefault;
run;

proc sort   data=set1.s1_teste;
      by repMC;
run;

/*Marcando ID*/

data ID_teste(drop=repMC);
      do repMC=1 to &repMC;
            do IDteste=1 to &nteste;
            output;
            end;
end;
run;

data set1.s1_teste;
      retain repMC IDteste default;
      set set1.s1_teste ;
      set ID_teste;
run;

proc freq data=set1.s1_teste;
      table repMC*default/missing nocol nofreq nopercent;
run;
proc delete data=set1.Teste_default set1.Teste_naodefault
set1.treino_default set1.treino_naodefault;run;
```