



UNICAMP

MATEUS CAETANO

MODELOS DE CLASSIFICAÇÃO - APLICAÇÕES NO SETOR
BANCÁRIO

CAMPINAS
2015



UNIVERSIDADE ESTADUAL DE CAMPINAS

Instituto de Matemática, Estatística
e Computação Científica

MATEUS CAETANO

MODELOS DE CLASSIFICAÇÃO - APLICAÇÕES NO SETOR BANCÁRIO

Dissertação apresentada ao Instituto de Matemática, Estatística e Computação Científica da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Mestre em matemática aplicada.

Orientador: Dr. Antonio Carlos Moretti

Coorientadora: Dra. Márcia Aparecida Gomes Ruggiero

ESTE EXEMPLAR CORRESPONDE À VERSÃO FINAL DA DIS-
SERTAÇÃO DEFENDIDA PELO ALUNO MATEUS CAETANO,
E ORIENTADA PELO PROF. DR. ANTONIO CARLOS Mo-
RETTI.

Assinatura do Orientador

Antonio Moretti

Assinatura da Coorientadora

Márcia Gomes Ruggiero

CAMPINAS
2015

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Ana Regina Machado - CRB 8/5467

C116m Caetano, Mateus, 1983-
Modelos de classificação : aplicações no setor bancário / Mateus Caetano. –
Campinas, SP : [s.n.], 2015.

Orientador: Antonio Carlos Moretti.
Coorientador: Márcia Aparecida Gomes Ruggiero.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de
Matemática, Estatística e Computação Científica.

1. Classificação - Modelos matemáticos. 2. Análise de regressão logística. 3. Redes neurais (Computação). 4. Máquina de vetores de suporte. 5. Árvores de decisões. I. Moretti, Antonio Carlos, 1958-. II. Ruggiero, Márcia Aparecida Gomes, 1956-. III. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Computação Científica. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Classification models : applications in banking sector

Palavras-chave em inglês:

Classification - Mathematical models

Logistic regression analysis

Neural networks (Computer science)

Support vectors machine

Decision trees

Área de concentração: Matemática Aplicada

Titulação: Mestre em Matemática Aplicada

Banca examinadora:

Antonio Carlos Moretti [Orientador]

Samara Flamini Kiihl

Washington Alves de Oliveira

Data de defesa: 06-02-2015

Programa de Pós-Graduação: Matemática Aplicada

Dissertação de Mestrado defendida em 06 de fevereiro de 2015 e aprovada

Pela Banca Examinadora composta pelos Profs. Drs.


Prof.(a). Dr(a). ANTONIO CARLOS MORETTI


Prof.(a). Dr(a). SAMARA FLAMINI KIIHL


Prof.(a). Dr(a). WASHINGTON ALVES DE OLIVEIRA

Abstract

Techniques for classification problems have applications in many areas, such as credit risk evaluation, image recognition, SPAM detection, among others. It is an area of intense research, for which many methods were and continue to be developed.

Given that there is not a method whose performance is better across any type of problems, different methods need to be compared in order to select the one that provides the best adjustment for each application in particular.

In this work, we studied six different methods applied to supervised classification problems (when there is a known response for the model training): Logistic Regression, Decision Tree, Naive Bayes, KNN (k -Nearest Neighbors), Neural Networks and Support Vector Machine.

We applied these methods on three data sets related to credit evaluation and customer selection for a banking marketing campaign. We made the data pre-processing to cope with missing data and unbalanced classes. We used data partitioning techniques and several metrics, as accuracy, F1 and ROC curve, in order to evaluate the methods/techniques performances.

We compared, for each problem, the performances of the different methods using the selected metrics. The results obtained for the best models on each application were comparable to other studies that have used the same data sources.

Keywords: Classification, Logistic Regression, Decision Tree, Naive Bayes, KNN, Neural Networks, Support Vector Machine.

Resumo

Técnicas para solucionar problemas de classificação têm aplicações em diversas áreas, como concessão de crédito, reconhecimento de imagens, detecção de SPAM, entre outras. É uma área de intensa pesquisa, para a qual diversos métodos foram e continuam sendo desenvolvidos.

Dado que não há um método que apresente o melhor desempenho para qualquer tipo de aplicação, diferentes métodos precisam ser comparados para que possamos encontrar o melhor ajuste para cada aplicação em particular.

Neste trabalho estudamos seis diferentes métodos aplicados em problemas de classificação supervisionada (onde há uma resposta conhecida para o treinamento do modelo): Regressão Logística, Árvore de Decisão, Naive Bayes, KNN (*k-Nearest Neighbors*), Redes Neurais e *Support Vector Machine*.

Aplicamos os métodos em três conjuntos de dados referentes à problemas de concessão de crédito e seleção de clientes para campanha de marketing bancário. Realizamos o pré-processamento dos dados para lidar com observações faltantes e classes desbalanceadas. Utilizamos técnicas de particionamento do conjunto de dados e diversas métricas, como acurácia, F1 e curva ROC, com o objetivo de avaliar os desempenhos dos métodos/técnicas.

Comparamos, para cada problema, o desempenho dos diferentes métodos considerando as métricas selecionadas. Os resultados obtidos pelos melhores modelos de cada aplicação foram compatíveis com outros estudos que utilizaram os mesmos bancos de dados.

Palavras-chave: Classificação, Regressão Logística, Árvore de Decisão, Naive Bayes, KNN, Redes Neurais, Support Vector Machine.

Sumário

Agradecimentos	xiii
1 Introdução	1
2 Modelos de Classificação	3
2.1 Classificação Supervisionada	3
2.2 Regressão Logística	6
2.2.1 Avaliação de qualidade do ajuste	8
2.2.2 Seleção de Modelos e Método de Akaike	11
2.3 Árvore de Decisão	12
2.4 k -Nearest Neighbors	15
2.5 Naive Bayes	17
2.6 Redes Neurais	18
2.6.1 Descrição do modelo	20
2.6.2 Ajuste da rede neural	21
2.6.3 Método de retropropagação de erro	22
2.6.4 Problemas no treinamento de Redes Neurais	24
2.7 Support Vector Machine	25
2.7.1 Hiperplano Separador	25
2.7.2 Hiperplano Separador Ótimo	26
2.7.3 Support Vector Machine	29
3 Preparação dos dados e medidas de qualidade do modelo	33
3.1 Preparação dos dados	33
3.1.1 Valores Faltantes	34
3.1.2 Técnicas para balanceamento de classes	35
3.1.3 Técnicas para seleção de atributos	35

3.1.4	Transformação de Variáveis	36
3.1.5	Regularização	36
3.2	Métodos para avaliar a qualidade do ajuste	36
3.2.1	<i>Hold-out</i>	37
3.2.2	Validação Cruzada (<i>K-Fold Cross Validation</i>)	37
3.2.3	Treinamento, Validação e Teste	37
3.2.4	Medidas	38
4	Aplicações	41
4.1	Procedimentos	41
4.2	Banco Português	42
4.2.1	Análise Exploratória	43
4.2.2	Regressão Logística	44
4.2.3	SVM	46
4.2.4	ANN	48
4.2.5	KNN	51
4.2.6	Árvore de Decisão	53
4.2.7	Naive Bayes	54
4.2.8	Comparação	56
4.2.9	Conclusão	57
4.3	Banco Alemão	58
4.3.1	Análise Exploratória	59
4.3.2	Regressão Logística	59
4.3.3	SVM	61
4.3.4	ANN	63
4.3.5	KNN	65
4.3.6	Árvore de Decisão	67
4.3.7	Naive Bayes	68
4.3.8	Comparação	69
4.3.9	Conclusão	70
4.4	Aprovação de crédito Australiano	71
4.4.1	Análise Exploratória	71
4.4.2	Regressão Logística	72
4.4.3	SVM	74
4.4.4	ANN	76

4.4.5	KNN	78
4.4.6	Árvore de Decisão	80
4.4.7	Naive Bayes	81
4.4.8	Comparação	82
4.4.9	Conclusão	83
5	Conclusão	85
A	Tabelas de Seleção de Parâmetros	89
A.1	Tabelas - Banco Português	89
A.1.1	SVM	89
A.1.2	ANN	90
A.2	Tabelas - Banco Alemão	91
A.2.1	SVM	91
A.2.2	ANN	92
A.3	Tabelas - Aprovação de Crédito Australiano	93
A.3.1	SVM	93
A.3.2	ANN	94

Agradecimentos

Agradeço em primeiro lugar aos meus pais, Francisco Natalin Caetano e Julia Maria Sgubin Caetano, pelos ensinamentos, carinho e apoio incondicionais durante toda minha vida.

Ao meu orientador Dr. Antonio Carlos Moretti e minha coorientadora Dra. Márcia Aparecida Gomes Ruggiero pelos seus ensinamentos, colaborações e conselhos valiosos e essenciais para este trabalho, além de todo o incentivo e motivação.

Aos docentes e colaboradores do IMECC, em especial ao Dr. Luiz Koodi Hotta, pela orientação e instrução durante o projeto de Iniciação Científica, e à Dra. Samara Flamini Kiihl, por toda a ajuda durante o desenvolvimento dessa dissertação.

Aos meus amigos da universidade e do trabalho pelo incentivo, inspiração e conhecimentos compartilhados.

Ao Instituto de Pesquisas Eldorado, que através de seu programa de capacitação, tornou possível que eu conciliasse o mestrado e as atividades do Instituto.

E ao CNPq, pelo período em que recebi apoio financeiro e por incentivar a pesquisa no país.

Lista de Ilustrações

2.1	Diagrama de modelo de classificação.	3
2.2	Exemplo de fronteira de decisão para problema com dois atributos	5
2.3	Exemplo de Árvore de Decisão	12
2.4	Exemplo de fronteira de decisão do método KNN	16
2.5	Exemplo Rede Neural	19
2.6	Exemplo Hiperplano Separador Ótimo	28
2.7	Exemplo Hiperplano Separador Ótimo - Classes Não Separáveis	29
2.8	Exemplo Support Vector Machine - Kernel RBF	31
3.1	Exemplo Curva Roc	40
4.1	Resíduos vs Alavancagem - Português	45
4.2	Seleção de Modelo - SVM F1 - Português	47
4.3	Seleção de Modelo - SVM - Português	47
4.4	Seleção de Modelo - ANN F1 - Português	49
4.5	Seleção de Modelo - ANN - Português	50
4.6	Seleção de Modelo - KNN - Português	52
4.7	Seleção de Modelo - Árvore - Português	53
4.8	Árvore de Decisão - Português	54
4.9	Seleção de Modelo - NB - Português	55
4.10	Curva Roc - Português	57
4.11	Resíduos vs Alavancagem - Alemão	60
4.12	Seleção de Modelo - SVM F1 - Alemão	62
4.13	Seleção de Modelo - SVM - Alemão	62
4.14	Seleção de Modelo - ANN F1 - Alemão	64
4.15	Seleção de Modelo - ANN - Alemão	64
4.16	Seleção de Modelo - KNN - Alemão	66

4.17 Seleção de Modelo - Árvore - Alemão	67
4.18 Seleção de Modelo - NB - Alemão	68
4.19 Curva Roc - Alemão	70
4.20 Resíduos vs Alavancagem (todas as obs.) - Australiano	73
4.21 Resíduos vs Alavancagem - Australiano	73
4.22 Seleção de Modelo - SVM F1 - Australiano	75
4.23 Seleção de Modelo - SVM - Australiano	75
4.24 Seleção de Modelo - ANN F1 - Australiano	77
4.25 Seleção de Modelo - ANN - Australiano	77
4.26 Seleção de Parâmetros - KNN - Australiano	79
4.27 Seleção de Modelo - Árvore - Australiano	80
4.28 Seleção de Modelo - NB - Australiano	81
4.29 Curva Roc - Australiano	83

Lista de Tabelas

3.1	Tabela de Contingência - Logística	39
4.1	Descrição do Banco de Dados - Banco Português	43
4.2	Fator de inflação da variância - Português	44
4.3	Teste Chi-Quadrado - Português	45
4.4	AIC e Deviance - Logística - Português	45
4.5	Tabela de Contingência - Logística - Português	46
4.6	Tabela de Resultados - Logística - Português	46
4.7	Tabela de Contingência - SVM - Português	48
4.8	Tabela de Resultados - SVM - Português	48
4.9	Tabela de Contingência - ANN - Português	50
4.10	Tabela de Resultados - ANN - Português	51
4.11	Tabela de Contingência - KNN - Português	52
4.12	Tabela de Resultados - KNN - Português	52
4.13	Tabela de Contingência - Árvore de Decisão - Português	54
4.14	Tabela de Resultados - Árvore de Decisão - Português	54
4.15	Tabela de Contingência - Naive Bayes - Português	55
4.16	Tabela de Resultados - Naive Bayes - Português	56
4.17	Tabela de Resultados - Português	56
4.18	Descrição do Banco de Dados - Banco Alemão	58
4.19	Teste Chi-Quadrado - Alemão	59
4.20	AIC e Deviance - Logística - Alemão	59
4.21	Tabela de Contingência - Logística - Alemão	60
4.22	Tabela de Resultados - Logística - Alemão	60
4.23	Tabela de Contingência - SVM - Alemão	63
4.24	Tabela de Resultados - SVM - Alemão	63
4.25	Tabela de Contingência - ANN - Alemão	65

4.26 Tabela de Resultados - ANN - Alemão	65
4.27 Tabela de Contingência - KNN - Alemão	66
4.28 Tabela de Resultados - KNN - Alemão	66
4.29 Tabela de Contingência - Árvore de Decisão - Alemão	67
4.30 Tabela de Resultados - Árvore de Decisão - Alemão	68
4.31 Tabela de Contingência - NB - Alemão	69
4.32 Tabela de Resultados - NB - Alemão	69
4.33 Comparativo de Resultados - Banco Alemão	69
4.34 Descrição do Banco de Dados - Aprovação de Crédito Australiano	71
4.35 Teste Chi-Quadrado - Australiano	72
4.36 AIC e Deviance - Logística - Australiano	72
4.37 Tabela de Contingência - Logística - Australiano	74
4.38 Tabela de Resultados - Logística - Australiano	74
4.39 Tabela de Contingência - SVM - Australiano	76
4.40 Tabela de Resultados - SVM - Australiano	76
4.41 Tabela de Contingência - ANN - Australiano	78
4.42 Tabela de Resultados - ANN - Australiano	78
4.43 Tabela de Contingência - KNN - Australiano	79
4.44 Tabela de Resultados - KNN - Australiano	79
4.45 Tabela de Contingência - Árvore de Decisão - Australiano	80
4.46 Tabela de Resultados - Árvore de Decisão - Australiano	81
4.47 Tabela de Contingência - NB - Australiano	82
4.48 Tabela de Resultados - NB - Australiano	82
4.49 Comparativo de Resultados - Banco Australiano	82
A.1 Seleção de Parâmetros - SVM - Português - 10 melhores em F1	89
A.2 Seleção de Parâmetros - SVM - Português - 20 melhores em Acurácia	90
A.3 Seleção de Parâmetros - ANN - Português - 5 melhores em F1	90
A.4 Seleção de Parâmetros - ANN - Português - 5 melhores em Acurácia	91
A.5 Seleção de Parâmetros - SVM - Alemão - 20 melhores em F1	91
A.6 Seleção de Parâmetros - SVM - Alemão - 20 melhores em Acurácia	92
A.7 Seleção de Parâmetros - ANN - Alemão - 5 melhores em F1	92
A.8 Seleção de Parâmetros - ANN - Alemão - 5 melhores em Acurácia	93
A.9 Seleção de Parâmetros - SVM - Australiano - 5 melhores em F1	93
A.10 Seleção de Parâmetros - SVM - Australiano - 5 melhores em Acurácia	93

A.11 Seleção de Parâmetros - ANN - Australiano - 5 melhores em F1	94
A.12 Seleção de Parâmetros - ANN - Australiano - 5 melhores em Acurácia	94

Capítulo 1

Introdução

Com a crescente quantidade de informações disponíveis e a evolução da capacidade de processamento dos computadores, ao longo das últimas décadas diversas técnicas foram e continuam sendo desenvolvidas para transformar os dados brutos em conhecimento, capaz de guiar tomadas de decisões. Diversas áreas da ciência lidam com esse tipo de tarefa, muitas vezes com o mesmo objetivo mas com diferentes abordagens, como a Estatística, Mineração de Dados, Aprendizado de Máquina e recentemente *Data Science*. Dentro da área de Mineração de Dados é comum dividir as técnicas entre aprendizado supervisionado e não-supervisionado [8]. Nesta dissertação de mestrado é realizado o estudo das principais técnicas de classificação supervisionada (onde há uma resposta correta para treinar o modelo).

Seguindo essa proposta, abordamos as seguintes técnicas: Regressão Logística, Árvore de Decisão, *Naive Bayes* (NB), KNN (*k-Nearest Neighbors*), Redes Neurais Artificiais (ANN) e *Support Vector Machine* (SVM).

Aplicamos os métodos em três conjuntos de dados referentes à problemas de concessão de crédito e seleção de clientes para campanha de marketing bancário. Realizamos o pré-processamento dos dados para lidar com observações faltantes e classes desbalanceadas. Utilizamos técnicas de particionamento do conjunto de dados e diversas métricas, como acurácia, F1 e curva ROC, com o objetivo de avaliar os desempenhos dos métodos.

Os detalhes das técnicas são apresentados no Capítulo 2. No Capítulo 3 são apresentadas técnicas auxiliares necessárias para a construção de um modelo de classificação, como pré-processamento dos dados e técnicas para avaliação da qualidade do ajuste que possam ser aplicadas à todos os métodos. No Capítulo 4 apresentamos as aplicações desenvolvidas no contexto de classificação binária.

Em cada aplicação foi possível analisar o desempenho dos métodos, de acordo com as diversas

métricas utilizadas. Os resultados obtidos pelos melhores modelos de cada aplicação são promissores, e a análise das medidas de desempenho mostra compatibilidade com outros estudos [18] [6] [12] realizados com os mesmos bancos de dados [3].

Neste trabalho adotamos a nomenclatura padrão empregada em trabalhos com tema em Mineração de Dados [8] [10].

Capítulo 2

Modelos de Classificação

2.1 Classificação Supervisionada

A construção de um modelo de classificação envolve um processo de dois estágios, sendo o primeiro o de aprendizado, onde o modelo de classificação é construído (ou treinado) a partir de um conjunto de dados, denominado conjunto de treinamento e, um de classificação, onde o modelo construído é utilizado para classificar novos dados (predição) [8]. O processo está ilustrado na Figura 2.1:

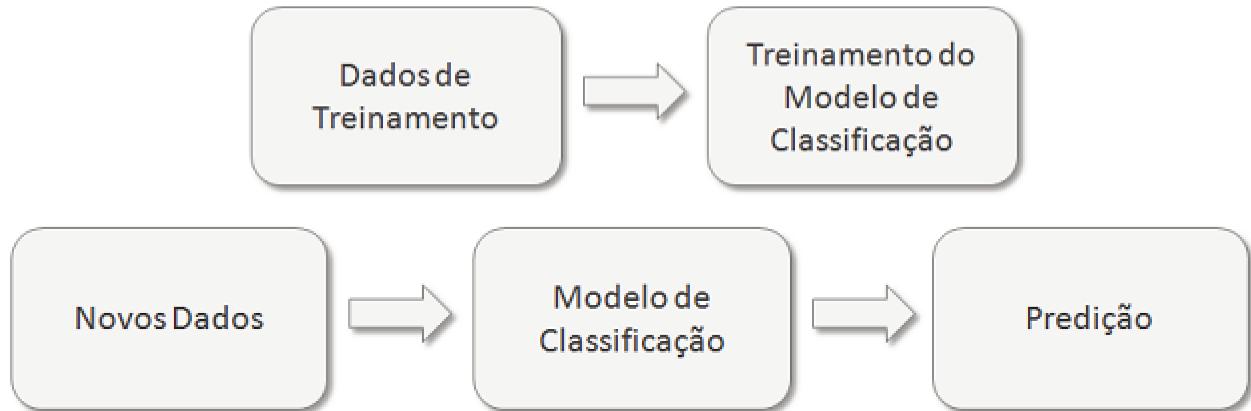


Figura 2.1: Diagrama de modelo de classificação.

Em geral, para a construção do modelo é necessária uma amostra dos dados com todos os atributos que definem cada observação, mais o atributo-classe que representa a resposta conhecida. O nome classificação supervisionada vem justamente do fato de termos esse conjunto de dados onde a resposta já é conhecida. Descrevemos abaixo, a título de exemplo, uma simples aplicação em

crédito bancário.

Suponha que temos um conjunto de dados de clientes que realizaram empréstimos no banco, com as seguintes informações para cada cliente:

- atributos: idade, cidade, renda mensal, estado civil, profissão;
- atributo-classe: pagou ou não pagou o empréstimo no tempo acordado.

Com isso pode-se buscar padrões para classificar um novo cliente como bom ou mal pagador (na prática estima-se a probabilidade do indivíduo pagar ou não), onde sabe-se os valores dos atributos, mas não do atributo-classe.

Os modelos de classificação, buscam portanto encontrar esses padrões e descrevê-los na forma de uma função, denominada regra de classificação, utilizada para separar cada novo exemplar dentro de sua categoria mais provável. Uma regra de classificação pode ser tão simples quanto uma estrutura *if-else*, passando por uma combinação linear dos valores dos atributos, ou uma elaborada função matemática.

Abaixo, segue um exemplo de regra de classificação:

```
if Idade < 25 then
    Classe ← sim
else
    Classe ← não
end if
```

A Figura 2.2 mostra um exemplo de problema de classificação com dois atributos X_1 e X_2 numéricos e um atributo-classe Y binário. O tipo do ponto representa a classe correta e o objetivo é encontrar com esses dados uma regra que possa determinar qual a classe de um novo atributo baseado em seus valores X_1 e X_2 . A curva, que consta nesta figura, representa uma possível fronteira de decisão. Acima da curva o modelo classifica a resposta como 1 e abaixo como 0. Note que algumas observações estão do lado incorreto da fronteira, sendo consideradas como erro do modelo.

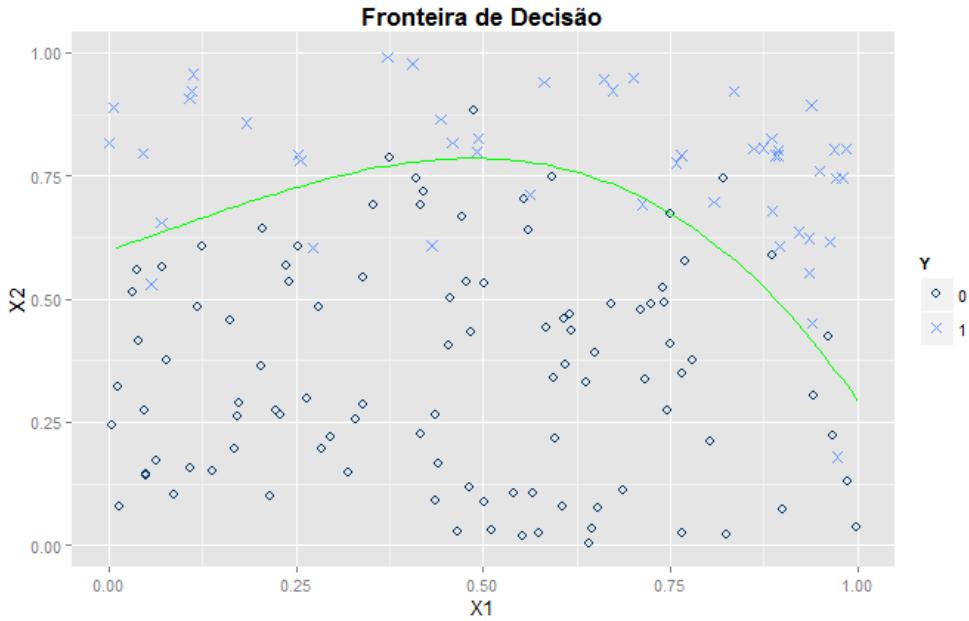


Figura 2.2: Exemplo de fronteira de decisão para problema com dois atributos

Um problema de classificação consiste portanto em encontrar uma relação entre um atributo-classe (atributo categórico ou quantitativo discreto) e demais atributos (categóricos ou quantitativos) através da construção de um modelo matemático ou estatístico, com base em dados contendo os valores dos atributos, e também do atributo-classe (daí o nome aprendizado supervisionado).

Considere um atributo-classe Y assumindo os valores y_1, \dots, y_n e um vetor de atributos X_1, \dots, X_k , busca-se encontrar uma relação funcional da forma:

$$Y = g(X_1, \dots, X_k),$$

ou ainda,

$$P[Y = y_i] = g(X_1, \dots, X_k) \text{ para } i = 1 \dots n,$$

ou seja, busca-se representar o atributo-classe Y , ou sua distribuição de probabilidade $P[Y = y_i]$, como uma função do vetor de atributos X . No caso onde Y assume apenas dois valores, temos um problema de classificação binária, do contrário temos um problema de classificação multi-classes.

Neste trabalho são apresentados e aplicados métodos para resolução de problemas para classes binárias. No entanto, esses algoritmos podem ser estendidos para problemas multi-classes. Uma das formas de fazê-lo é através da utilização do Método one-versus-all [19]. Nesta abordagem, se temos k diferentes classes, são treinados k modelos, sendo que para cada modelo uma classe é

selecionada como classe de interesse e as demais são agrupadas como se fossem uma só, reduzindo o problema a uma classificação binária.

Existem diversos métodos, com abordagens e técnicas distintas para a resolução do problema de classificação, e não há uma regra geral que defina qual método é melhor. Nesse capítulo serão apresentados os seguintes métodos:

- Regressão Logística
- Árvore de Decisão
- *Naive Bayes*
- Redes Neurais
- KNN (*k-Nearest Neighbors*)
- SVM (*Support Vector Machine*)

2.2 Regressão Logística

A regressão logística pertence à família de Modelos Lineares Generalizados, e em sua forma mais simples [25], com atributo-classe (variável resposta) binário, tem as seguintes características: seja $Y \in \{0, 1\}$ o atributo-classe com distribuição Bernoulli $B(p)$ e $X_{(1 \times (m+1))} = (1, X_1, \dots, X_m)$ o vetor de atributos, onde o valor 1 na primeira componente representa o intercepto da regressão.

A distribuição condicional de Y dado X é dada por:

$$P[Y = 1|X] = g(X\beta), \quad (2.2.1)$$

onde:

- $\beta_{((m+1) \times 1)}$ é um vetor de parâmetros que precisam ser estimados a partir dos dados;
- $g(\cdot)$ é a função logística (ou Sigmoid):

$$g(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{1 + e^x}.$$

Para estimar os parâmetros β pode-se utilizar o método de máxima verossimilhança [25]. A função de log-verossimilhança é dada por:

$$\begin{aligned}
l(\beta, y, x) &= \log \left(\prod_{i=1}^n P(Y = 1|x_i)^{y_i} (1 - P(Y = 1|x_i))^{1-y_i} \right) \\
&= \sum_{i=1}^n y_i \log(g(x_i\beta)) + (1 - y_i) \log(1 - g(x_i\beta)).
\end{aligned} \tag{2.2.2}$$

No método da máxima verossimilhança, busca-se o valor dos parâmetros que maximizam a função de verossimilhança, ou equivalentemente a função de log-verossimilhança [9]. E que pode ser escrito como o seguinte problema de otimização:

$$\min_{\beta} \left[\sum_{i=1}^n -y_i \log(g(x_i\beta)) - (1 - y_i) \log(1 - g(x_i\beta)) \right]. \tag{2.2.3}$$

onde:

- n = número de observações;
- y_i = i -ésima resposta;
- x_i = vetor de atributos correspondente à i -ésima observação;
- β = vetor de parâmetros.

A função objetivo em 2.2.3 é convexa e admite um único valor mínimo, que é o mínimo global [16].

Uma das vantagens da regressão logística é que os parâmetros possuem uma interpretação. Considere um exemplo simples, onde Y representa a presença ou não de uma doença, e X representa a presença de um fator de risco:

$$p = P[Y = 1|X = x] = g(\beta_0 + \beta_1 x).$$

A chance de determinado evento é definida como a probabilidade de ocorrência desse evento dividida pela probabilidade de não ocorrência do evento, e é dada por:

$$\begin{aligned}
\text{Chances} &= \frac{p}{1-p} \\
&= e^{\beta_0 + \beta_1 x}.
\end{aligned} \tag{2.2.4}$$

No exemplo da doença, o evento poderia ser definido como a presença da doença.

A razão de chances, representa a mudança na chance na presença do fator de risco X , ou seja, a mudança na chance quando $X = 1$ em relação à $X = 0$, e é dada por:

$$\begin{aligned}\text{Razão de Chances} &= \frac{e^{\beta_0 + \beta_1}}{e^{\beta_0}} \\ &= e^{\beta_1}.\end{aligned}\tag{2.2.5}$$

Interpretação semelhante pode ser obtida quando o atributo X é contínuo, onde a quantidade e^{β_1} , representa a mudança na chance para uma mudança de uma unidade no atributo [25].

Para a utilização de atributos categóricos na regressão logística é necessária a conversão para múltiplos atributos indicadores, sendo necessários $m - 1$ atributos indicadores, onde m é o número de categorias distintas do atributo [25].

2.2.1 Avaliação de qualidade do ajuste

Resíduos

Dois tipos de resíduos podem ser analisados para um modelo de regressão logística, os resíduos de Pearson e o resíduo Deviance [4].

O resíduo de Pearson é definido como:

$$r_i = \frac{y_i - \hat{\pi}_i}{\sqrt{\hat{\pi}_i(1 - \hat{\pi}_i)}},\tag{2.2.6}$$

e o resíduo Deviance é definido como:

$$r_i = s_i \sqrt{-2 \{y_i \log(\hat{\pi}_i) + (1 - y_i) \log(1 - \hat{\pi}_i)\}},\tag{2.2.7}$$

onde:

- $\hat{\pi}_i = g(x_i \hat{\beta})$;
- $\hat{\beta}$ é o vetor de parâmetros estimados;
- $s_i = 1$ se $y_i = 1$, e $s_i = -1$ se $y_i = 0$.

Alavancagem

No modelo de regressão linear, a alavancagem (*Leverage*) de uma observação y_i , é definida como o componente diagonal h_{ii} da matriz de projeção \mathbf{H} , que tem esse nome pois projeta o vetor de observações no vetor de valores ajustados:

$$\hat{y} = \mathbf{H}y.\tag{2.2.8}$$

O componente diagonal mede a alavancagem que um ponto tem sobre seu próprio valor ajustado [4].

Para o modelo de regressão logística, uma matriz de projeção pode ser definida como:

$$\mathbf{H} = \mathbf{W}^{1/2} \mathbf{X} (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{W}^{1/2} \quad (2.2.9)$$

onde \mathbf{W} é a matriz diagonal com i -ésima entrada $\hat{\pi}_i (1 - \hat{\pi}_i)$.

Embora essa matriz não satisfaça a relação (2.2.8), já que o modelo não produz estimativas lineares, ela ainda tem propriedades similares à matriz de projeção da regressão linear [4] e seus elementos da diagonal podem ser utilizados como uma medida de alavancagem.

Distância de Cook

A distância de Cook [20] é uma estatística utilizada para identificação de valores extremos, e mede a influência da i -ésima observação nos parâmetros estimados. A estatística é definida como:

$$D_i = \frac{\{\hat{\beta} - \hat{\beta}(i)\}^T \mathbf{X}^T \mathbf{W} \mathbf{X} \{\hat{\beta} - \hat{\beta}(i)\}}{ps^2}, \quad (2.2.10)$$

onde:

- \mathbf{X} é a matriz de atributos;
- $\hat{\beta}$ é um vetor de parâmetros estimados;
- $\hat{\beta}(i)$ é um vetor de parâmetros estimados sem a i -ésima observação;
- p é o número de atributos acrescidos de um;
- s^2 é a estimativa da variância obtida no modelo com todas as observações;
- \mathbf{W} é a matriz diagonal com i -ésima entrada $\hat{\pi}_i (1 - \hat{\pi}_i)$.

Um número grande de D_i representa uma ponto influente. Não há uma definição exata do que é um valor grande para D_i , mas como regra prática utiliza-se: se $D_i > 1$ a observação deve provavelmente ser considerada influente; se todos os D_i são menores que 1, um valor maior do que os demais deve ser considerado influente.

Teste Chi-Quadrado

A estatística utilizada para determinar a significância de um modelo logístico é denominada teste de razão de verossimilhança. Este teste compara a verossimilhança do modelo completo (com todos atributos inclusos) com a verossimilhança do modelo nulo (contém apenas o intercepto). A fórmula da estatística do teste é:

$$\begin{aligned} G^2 &= 2 \log \frac{L}{L_0} \\ &= 2(\log L - \log L_0), \end{aligned} \tag{2.2.11}$$

onde L é a verossimilhança do modelo completo e L_0 a verossimilhança do modelo nulo [25].

A estatística do teste tem distribuição aproximada Chi-Quadrado com k graus de liberdade (k é o número de atributos no modelo completo). Nesse caso, a área da curva Chi-quadrado à direita do valor obtido na estatística representa o p -valor, o menor nível de significância a partir do qual rejeitamos a hipótese nula de equivalência dos modelos.

Logo, se significante, essa estatística sugere que se considerados em conjunto, os atributos contribuem significativamente para a predição do atributo-classe.

Duas condições devem ser satisfeitas para a aplicação do teste:

- ambos os modelos devem ser ajustados com as mesmas observações;
- os modelos devem ser aninhados, ou seja, os atributos do modelo mais simples deve estar contido no modelo completo.

O teste pode ser estendido para comparar modelos completos e reduzidos e, neste caso, a estatística do teste é dada por:

$$\begin{aligned} G^2 &= 2 \log \frac{L_c}{L_r} \\ &= 2(\log L_c - \log L_r), \end{aligned} \tag{2.2.12}$$

onde L_c é a verossimilhança do modelo completo e L_r a verossimilhança do modelo reduzido. A distribuição é a mesma Chi-quadrado k .

Deviance

Um indicador de qualidade do ajuste muito utilizado para a regressão logística é a estatística denominada Deviance [25].

A estatística Deviance é uma extensão do teste Chi-quadrado, mas que consiste em comparar o modelo completo com o modelo saturado (modelo com um parâmetro por observação, resultando em ajuste perfeito). A Deviance pode ser interpretada como uma medida da variação inexplicada no modelo [25] e é definida por:

$$\begin{aligned} D &= 2 \log \frac{L_s}{L_c} \\ &= 2(\log L_s - \log L_c) \\ &= -2 \log L_c, \end{aligned} \tag{2.2.13}$$

onde L_c é a verossimilhança do modelo completo e L_s a verossimilhança do modelo saturado. Temos que L_s deve ser 1, já que o modelo saturado ajusta perfeitamente os dados. A distribuição de D é Chi-quadrado com k graus de liberdade.

2.2.2 Seleção de Modelos e Método de Akaike

O método proposto por Akaike [1] é baseado na ideia da seleção de um modelo que seja parcimonioso, ou em outras palavras, que esteja bem ajustado e tenha um número reduzido de parâmetros [21]. O Critério de Informação de Akaike (AIC) é definido como:

$$AIC = -l(\beta) + 2p, \tag{2.2.14}$$

em que p denota o número de parâmetros e $l(\beta)$ é o logaritmo da função de verossimilhança. Como $l(\beta)$ cresce com o aumento do número de parâmetros do modelo, a medida tem um componente de penalização proporcional ao número de parâmetros.

O método de Akaike busca encontrar o modelo com menor valor para a função AIC. Uma implementação deste método pode ser encontrada no R, na função stepAIC (pacote MASS [27]), que consiste de uma busca iterativa nas duas direções (adicionando e removendo variáveis), denominada *stepwise*.

O método parte de um modelo pré-especificado com m atributos, denominado modelo nulo. A cada iteração retira-se um atributo por vez do modelo nulo e calcula-se o AIC para cada modelo com $m - 1$ atributos. Além disso, adiciona-se atributos disponíveis e calcula-se o AIC para cada modelo com $m + 1$ atributos. O modelo com menor AIC é selecionado como sendo o modelo nulo para a próxima iteração até que o menor AIC seja o do modelo nulo, indicando que a adição ou remoção de atributos naquele modelo implica em acréscimo do AIC.

Outra alternativa para seleção de atributos é considerar o p -valor dos parâmetros e remover os que não são estatisticamente significantes.

2.3 Árvore de Decisão

Uma árvore de decisão é uma estrutura em forma de árvore, onde cada nó interno representa um teste em um atributo, cada ramo representa um possível valor do atributo, e cada nó folha (ou nó terminal) representa uma decisão para um valor do atributo-classe [8]. O nó mais ao topo da árvore é denominado nó raiz.

A Figura 2.3 é um exemplo de uma Árvore de Decisão para uma aplicação de concessão de crédito. A árvore é avaliada de cima para baixo, onde o nó raiz é um teste no atributo Idade. Neste caso se o indivíduo tem menos de 25 anos ele é direcionado para o nó-interno à esquerda, que é um teste no atributo Estudante, caso contrário ele é direcionado para o teste à direita, com o atributo Renda Mensal. Dependendo do resultado do segundo teste a decisão é tomada através do valor correspondente ao nó-terminal, concedendo ou não, a aprovação de crédito para o cliente.

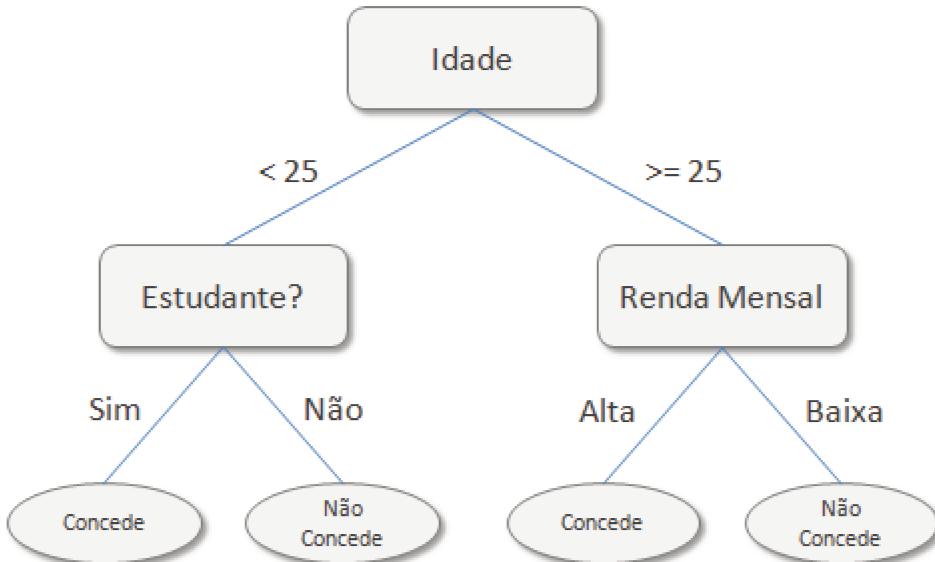


Figura 2.3: Exemplo de Árvore de Decisão

Dada uma nova observação, ao qual a classe é desconhecida, os valores dos atributos são testados na árvore de decisão. Um caminho é traçado do nó raiz até algum dos nós folha, e o valor da folha em questão é atribuída à observação. Nota-se que uma regra de decisão do tipo *if-else* pode ser facilmente construída a partir da árvore.

A construção das árvores não requer conhecimentos específicos e nem detalhamento de parâmetros, o que as torna apropriadas para descoberta de conhecimento exploratório. Árvores de decisão conseguem lidar com dados multidimensionais. Sua representação em forma de árvore é

intuitiva e fácil de assimilar. Os passos de aprendizado e classificação são simples e rápidos e, em geral, árvores de decisão têm uma boa acurácia. [8]

O processo de construção da árvore é denominado Indução da Árvore de Decisão e existem diversos algoritmos com esse propósito, sendo os mais conhecidos o CART, ID3 e o C4.5 [8]. Esses algoritmos selecionam a ordem dos atributos na árvore e a forma de particionamento dos ramos.

Em sua forma mais usual, a árvore é construída do topo para baixo, onde o conjunto de treinamento é recursivamente particionado em subconjuntos menores à medida que a árvore é construída. Um algoritmo básico [8] é descrito a seguir:

Pseudocódigo Gerar_Arvore_de_Decisao

Entradas:

- partição de dados D_i: o conjunto de exemplares de treinamento e suas classes associadas;
- lista_de_atributos: o conjunto de atributos candidatos;
- metodo_de_seleção_de_atributo: procedimento para determinar o critério de divisão que melhor particiona os exemplares em suas classes individuais.

Saída:

- Árvore de decisão

Algoritmo 2.1 Algoritmo básico de Árvore de Decisão [8].

```
N = novo Nó;  
if contagem(valores_distintos( $D$ )) = 1 then  
    classe( $N$ )  $\leftarrow$  valores_distintos( $D$ );  
    return( $N$ );  
end if  
if lista_de_atributos is empty then  
    Retornar  $N$  como nó folha, com classe sendo a classe majoritária em  $D$ ; (voto pela maioria)  
    return( $N$ );  
end if  
criterio_de_particao = metodo_de_seleção_de_atributo( $D$ , lista_de_atributos) //busca melhor critério de partição, atributo e divisão;  
Nome( $N$ )  $\leftarrow$  criterio_de_particao.atributo;  
if criterio_de_particao.atributo é discreto then  
    lista_de_atributos = lista_de_atributos - atributo_de_particao  
end if  
for  $j$  in 1:count(criterio_de_particao.divisão) do  
     $D_j$   $\leftarrow$  o conjunto de exemplares em  $D$  satisfazendo a saída  $j$ ; //uma partição  
    if lista_de_atributos é vazio then  
        Criar um nó folha com a classe majoritária em  $D$  para o nó  $N$ ;  
    else  
        Criar um nó com o retorno de Gerar_Arvore_de_Decisao( $D_j$ , lista_de_atributos) para o nó  $N$ ;  
    end if  
end for  
return( $N$ );
```

Diferentes métodos de seleção de atributo são utilizados, onde o objetivo é selecionar um atributo para o nó, de forma a minimizar a impureza nas folhas, ou seja, obter folhas que separem os diferentes valores do atributo-classe da forma mais homogênea possível. Dentre as medidas de impureza, destacamos a Entropia (também denominada Deviance) e o Índice de Gini.

Considere K classes. Seja \hat{p}_{mk} a proporção de observações da classe k no nó m . A entropia [17] é definida como:

$$E = \sum_{k=1}^K \hat{p}_{mk} \log \hat{p}_{mk}, \quad (2.3.1)$$

e o Índice de Gini [17]:

$$\sum_{k=1}^K \hat{p}_{mk} (1 - \hat{p}_{mk}). \quad (2.3.2)$$

Neste trabalho foi utilizado apenas o critério Entropia (Deviance), implementado no pacote tree do R [26].

2.4 *k*-Nearest Neighbors

O método *k*-Nearest Neighbors (KNN) foi descrito inicialmente no início dos anos 50 [7]. Trata-se de um método que requer alto esforço computacional quando aplicado em grandes quantidades de dados, e por esse motivo ganhou popularidade apenas a partir dos anos 60 com o aumento da capacidade dos computadores. É muito usado na área de reconhecimento de padrões [8].

Baseia-se em aprendizado por analogia, ou seja, comparando uma observação de teste com as observações de treinamento similares. O critério de similaridade é definido usualmente através de uma medida de distância no espaço multidimensional das observações. O tipo de medida de distância pode variar, sendo a mais usual a distância Euclidiana para atributos numéricos.

Por essa razão o método KNN é classificado como *lazy learner*, já que não há a construção de um modelo. Quando carregados com o conjunto de treinamento, a informação é apenas armazenada com mínimo processamento. Mas assim que uma observação de teste é apresentada para a classificação, são então calculadas as distâncias entre as observações, para que as observações mais próximas (vizinhas) sejam selecionadas do conjunto de treinamento, e a classe do atributo de interesse possa ser determinada.

Lazy Learners podem ser computacionalmente caros durante a tarefa de classificação, e requerem portanto técnicas eficientes de armazenamento de dados. Oferecem pouca informação a

respeito da estrutura dos dados. Os pontos positivos são a capacidade natural de suportar aprendizado incremental e a habilidade de modelar espaços de decisões complexos contendo fronteiras de decisão não-lineares, que podem não ser facilmente descritas por outros métodos [8].

A Figura 2.4 apresenta um exemplo de fronteira de decisão obtida através do método KNN.

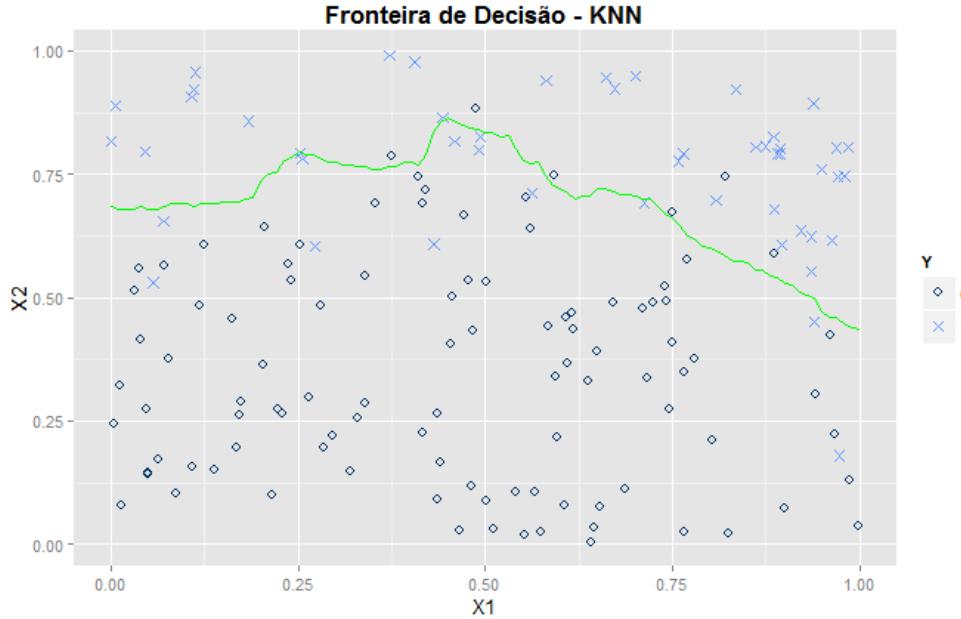


Figura 2.4: Exemplo de fronteira de decisão do método KNN

Definição 2.4.1. Considere um atributo-classe Y assumindo os valores y_1, \dots, y_n e um vetor de atributos X_1, \dots, X_k . Sejam $X_1 = (X_{11}, X_{12}, \dots, X_{1k})$ e $X_2 = (X_{21}, X_{22}, \dots, X_{2k})$ duas observações do vetor de atributos.

Se X_1, \dots, X_k são todos numéricos, um critério de distância possível é a distância Euclidiana definida como:

$$Dist(X_1, X_2) = \sqrt{\sum_{i=1}^k (X_{1i} - X_{2i})^2}. \quad (2.4.1)$$

Outras medidas são bastante utilizadas, como a distância de Manhattan:

$$Dist(X_1, X_2) = \sum_{i=1}^k |X_{1i} - X_{2i}|. \quad (2.4.2)$$

Um problema que pode surgir ao calcular-se a distância diretamente é que se o atributo X_i tem maior variação que o atributo X_j , o primeiro terá um impacto maior na distância. A solução

para essa dificuldade consiste em transformar todos os atributos para uma escala similar. A forma mais utilizada é a normalização min-max, que converte a variável para o intervalo $[0, 1]$, conforme a Equação (2.4.3). No entanto, outros métodos podem ser utilizados, como a normalização por desvio-padrão [8] descrita na Equação (2.4.4).

$$X' = \frac{x - \min(X)}{\max(X) - \min(X)}, \quad (2.4.3)$$

$$X'' = \frac{x - \text{média}(X)}{\sigma(X)}, \quad (2.4.4)$$

onde $\sigma(X)$ é o desvio-padrão de X .

Para atributos categóricos, uma medida possível é definir a distância como 1 se o valor do atributo é idêntico entre as observações e 0 caso contrário. Outros métodos podem adotar uma escala para atributos ordinais, onde é possível estabelecer uma relação de ordem entre os diferentes valores possíveis do atributo.

O algoritmo pode incluir seu próprio tratamento para dados faltantes considerando a distância máxima possível. Portanto se X_1 e X_2 são faltantes, definimos como 1. Se apenas X_1 é faltante, definimos como $\max(|1 - x_1|, |0 - x_1|)$.

Para a classificação pelos k -vizinhos mais próximos, o atributo-classe da nova observação pode ser definido como a moda do atributo-classe para os k -vizinhos selecionados. Para atributos numéricos, a média ou mediana podem ser utilizadas. Variações incluem a utilização de pesos no cálculo das medidas citadas, pesos esses inversamente proporcionais à distância entre as observações de treinamento e a nova observação.

O parâmetro k pode ser definido de forma experimental, buscando a menor taxa de erro através do incremento de k de 1 à n , sendo n o número de observações de treinamento. Em geral, quanto maior o conjunto de treinamento, maior será o valor de k .

O método é computacionalmente caro, pois se n é o número de observações do conjunto de testes e $k = 1$, então $O(n)$ comparações são necessárias para cada observação de teste. No entanto, pode-se reduzir a complexidade para $O(\log(n))$ fazendo uma pré-ordenação adequada e uma organização dos dados em árvores de busca.

2.5 Naive Bayes

O método *Naive Bayes* seleciona, como valor do atributo-classe, a classe com a maior probabilidade de ser observada dado os demais atributos. Essa probabilidade é estimada com base na fórmula de probabilidade condicional de Bayes.

$$P(Y = k|X = x) = \frac{P(Y = k, X = x)}{P(X = x)} = \frac{P(X = x|Y = k)P(Y = k)}{\sum_{i \in K} P(X = x|Y = i)}. \quad (2.5.1)$$

Diversos métodos são construídos com base nessa fórmula, variando as suposições a respeito das distribuições de X e Y . No método Naive Bayes [10], a suposição é que a distribuição condicional de X_i é independente para todo $i \in \{1 \dots p\}$, ou seja:

$$P(X = x|Y = k) = P(X_1 = x_1, \dots, X_n = x_n|Y = k) = \prod_{i=1}^n P(X_i = x_i|Y = k). \quad (2.5.2)$$

E no caso de X contínua, obtemos:

$$f_{X|Y=k}(x) = \prod_{i=1}^n f_{X_i|Y=k}(x_i), \quad (2.5.3)$$

onde $f_{X|Y=k}(x)$ é a distribuição condicional de X dado Y .

O nome do método, Naive Bayes, vem dessa suposição, que em geral não é verdadeira mas simplifica a estimação dramaticamente [10]:

- as densidades marginais condicionais $f_{X_i|Y=k}$ podem ser estimadas separadamente. Usualmente a distribuição é assumida como Gaussiana, embora outras distribuições possam ser utilizadas;
- se um componente X_i é discreto, então um histograma pode ser utilizado para aproximar a distribuição, e dessa forma misturar variáveis de diferentes tipos.

Apesar dessas fortes e otimistas suposições, o método Naive Bayes frequentemente provê resultados melhores do que modelos mais sofisticados [10].

2.6 Redes Neurais

O campo de redes neurais foi desenvolvido inicialmente por neuro-biologistas e psicólogos, como uma forma de desenvolver e testar modelos análogos ao neurônios.

Uma rede neural é um conjunto de unidades de entrada e saída em que cada conexão tem um peso associado. Os pesos são calculados durante a fase de treinamento da rede, com o objetivo de minimizar o erro do atributo-classe (variável resposta).

Há muitos tipos de redes neurais e algoritmos de redes neurais. O algoritmo mais popular é o retropropagação de erro (*backpropagation*). Esse algoritmo é aplicado em uma rede do tipo múltiplas camadas alimentadas adiante (*multilayer feed-forward*), que consiste de um modelo com

uma camada de entrada, algumas camadas escondidas, e uma camada de saída [8]. A camada de entrada representa os atributos e a camada de saída é utilizada, no caso do problema de classificação, para representar o valor do atributo-classe.

Além disso, as conexões são construídas em um único sentido, ou seja, não pode haver um peso da camada escondida retroalimentando a camada de entrada, ou ainda da camada escondida 2 para a camada escondida 1, e assim por diante, daí a nomenclatura alimentação adiante. A rede pode ser ainda completamente conectada, quando cada unidade provê uma entrada para todas as unidades da camada seguinte.

A Figura 2.5 mostra a estrutura típica de uma rede neural:

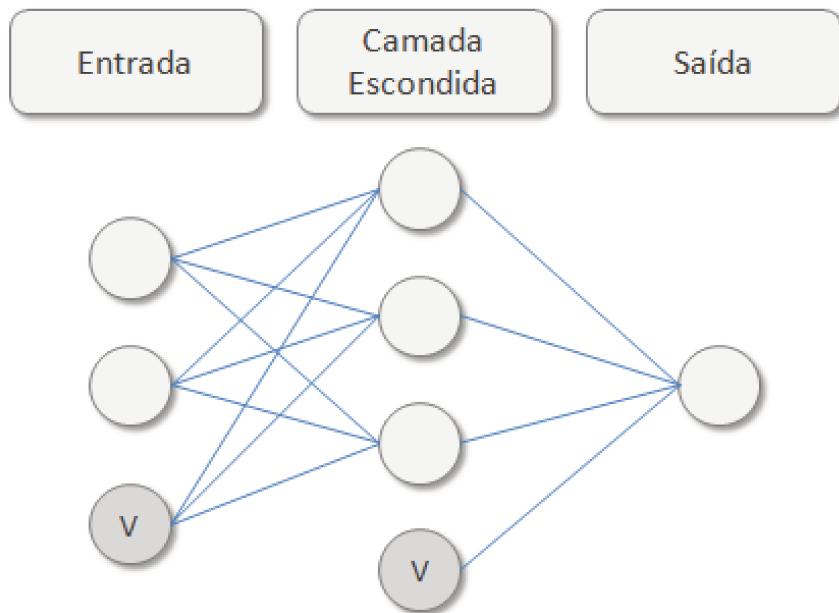


Figura 2.5: Exemplo Rede Neural

Usualmente os valores da camada de entrada são transformados para o intervalo $[0,1]$, assim como a saída obtida pertence a esse intervalo. Nas camadas internas são realizadas somas ponderadas dos valores da camada anterior, com a opção de uma entrada unitária adicional denominada viés, representada na Figura 2.5 pela letra V, que funciona de forma equivalente ao intercepto em uma regressão.

Os pesos são estimados através de algoritmos como o retropropagação. Uma função de ativação transforma o valor dessa soma para o intervalo $[0,1]$ para ser utilizado na camada seguinte.

Em termos estatísticos, o modelo de redes neurais é classificado como um modelo de regressão não-linear, já que faz combinações não-lineares dos atributos de entrada. Além disso, há um

resultado teórico que garante que dadas unidades escondidas e amostras de treinamento suficientes, uma rede neural pode aproximar qualquer função [8].

Não há regra prática sobre o número ótimo de camadas internas. Além disso, o modelo de redes neurais é considerado um modelo “caixa-preta” devido à dificuldade, senão impossibilidade, de obter uma interpretação do resultado.

Não há uma regra que defina o número ideal de camadas escondidas e o número de nós em cada camada. Essas configurações são normalmente determinadas empiricamente, levando em conta as características específicas da aplicação em questão [8]. As técnicas de validação cruzada para estimação de acurácia descritas no Capítulo 3 podem ser utilizadas para a escolha de uma configuração adequada.

O treinamento da rede neural pode exigir muito em termos computacionais, o que dependendo do número de atributos e observações, pode resultar em tempos de execução elevados.

O algoritmo de retropropagação processa iterativamente um conjunto de dados de treinamento, comparando o valor previsto com o valor correto, que pode ser um valor contínuo ou uma categoria, dependendo do tipo de problema.

O algoritmo modifica os pesos de forma a minimizar o erro quadrático médio das previsões. As modificações são feitas na direção para trás, alterando primeiramente os pesos ligados à camadas de saída, depois as camadas internas e finalmente a camada de entrada (daí o nome retropropagação de erro).

Não há garantia que esse processo iterativo de minimização de erros converja para o mínimo global, embora em geral encontre resultados bons o suficiente [8].

2.6.1 Descrição do modelo

A descrição do modelo segue conforme apresentada em Hastie et. al [10].

Para o caso de classificação multi-classes, há K nós terminais, com a k -ésima unidade modelando a probabilidade da classe k . Temos K medidas alvo Y_k , $k = 1, \dots, K$, cada uma codificada como 0 ou 1 para a k -ésima classe. Os atributos derivados Z_m são combinações lineares dos atributos de entrada, e os alvos Y_k são modelados como uma função das combinações lineares de Z_m ,

$$\begin{aligned} Z_m &= h(\alpha_{0m} + \alpha_m^T X), \quad m = 1, \dots, M, \\ T_k &= \beta_{0k} + \beta_k^T Z, \quad k = 1, \dots, K, \\ f_k(X) &= g_k(T), \quad k = 1, \dots, K, \end{aligned} \tag{2.6.1}$$

onde $X = (X_1, X_2, \dots, X_L)$, $Z = (Z_1, Z_2, \dots, Z_M)$, $T = (T_1, T_2, \dots, T_K)$, $\alpha_m \in \Re^L$ e $\beta_k \in \Re^K$.

A função $h(v)$ é denominada função de ativação, e pode assumir as seguintes formas:

- sigmoid $h(v) = \frac{1}{1+e^{-v}}$;
- função de base radial Gaussiana (RBF) - *Gaussian radial basis function*;
- tangente hiperbólica.

Normalmente há a adição de uma unidade adicional em cada camada denominada *bias* (viés), que é na prática uma entrada com uma constante 1, e captura o interceptos α_{0m} e β_{0k} nos modelos (2.6.1).

A função de saída $g_k(T)$ permite uma transformação final do vetor de saídas T . Para a regressão a função identidade pode ser utilizada, enquanto para classificação, a função *softmax* é mais utilizada:

$$g_k(T) = \frac{e^{T_k}}{\sum_{l=1}^K e^{T_l}}. \quad (2.6.2)$$

As unidades intermediárias Z_m formam a camada escondida, e seus valores não são diretamente observados. Podemos pensar em Z_m como uma expansão de base das entradas originais X .

2.6.2 Ajuste da rede neural

O modelo de rede neural tem parâmetros desconhecidos, denominados pesos, e nosso objetivo é estimar seus valores de forma que o modelo se ajuste adequadamente ao conjunto de treinamento. Denotamos o conjunto de pesos como θ , consistindo de:

$$\begin{aligned} \{\alpha_{0m}, \alpha_m; m = 1, 2, \dots, M\} &\quad M(L + 1) \text{ pesos} \\ \{\beta_{0k}, \beta_k; k = 1, 2, \dots, K\} &\quad K(M + 1) \text{ pesos} \end{aligned} \quad (2.6.3)$$

Para classificação, podemos utilizar como função de erro, tanto a função de erro quadrado:

$$R(\theta) = \sum_{k=1}^K \sum_{i=1}^N (y_{ik} - f_k(x_i))^2, \quad (2.6.4)$$

quanto a entropia cruzada (deviance):

$$R(\theta) = - \sum_{k=1}^K \sum_{i=1}^N (y_{ik} \log f_k(x_i) + (1 - y_{ik}) \log (1 - f_k(x_i))), \quad (2.6.5)$$

e o classificador correspondente é

$$G(x) = \underset{k}{\operatorname{argmax}} f_k(x). \quad (2.6.6)$$

Com a função de ativação sigmoid e entropia cruzada como erro, a rede neural equivale a uma regressão logística linear nas unidades escondidas, e todos os parâmetros são estimados por máxima verossimilhança [10].

Para o caso geral, o objetivo é minimizar a função $R(\theta)$, e por tratar-se de um problema de otimização podemos empregar diferentes métodos para sua resolução [5]. Uma das opções é o método denominado retropropagação de erro [10], que utiliza o método de máxima descida para a minimização.

A obtenção do mínimo global de $R(\theta)$ frequentemente resulta em um problema de sobre-ajuste (*overfitting*). Para evitar isso, podemos adicionar um termo de regularização ou controlar o critério de parada.

2.6.3 Método de retropropagação de erro

Conforme apresentado em [10], de (2.6.1), sejam $z_{mi} = h(\alpha_{0m} + \alpha_m^T x_i)$ e $z_i = (z_{1i}, z_{2i}, \dots, z_{Mi})$, $i = 1 \dots N$, onde N é o número de observações, então:

$$R(\theta) \equiv \sum_{i=1}^N R_i = \sum_{k=1}^K \sum_{i=1}^N (y_{ik} - f_k(x_i))^2, \quad (2.6.7)$$

com derivadas

$$\begin{aligned} \frac{\partial R_i}{\partial \beta_{km}} &= -2(y_{ik} - f_k(x_i))g'_k(\beta_k^T z_i)z_{mi}, \\ \frac{\partial R_i}{\partial \alpha_{ml}} &= -\sum_{k=1}^K 2(y_{ik} - f_k(x_i))g'_k(\beta_k^T z_i)\beta_{km}h'(\alpha_m^T x_i)x_{il}. \end{aligned} \quad (2.6.8)$$

onde β_{km} é o m -ésimo elemento de β_k , $m = 1, \dots, M$, e α_{ml} é o l -ésimo elemento de α_m , $l = 1, \dots, L$.

Dadas essas derivadas, uma atualização do método de máxima descida da $(r+1)$ -ésima iteração

tem a forma:

$$\begin{aligned}\beta_{km}^{r+1} &= \beta_{km}^r - \gamma_r \sum_{i=1}^N \frac{\partial R_i}{\partial \beta_{km}^{(r)}}, \\ \alpha_{ml}^{r+1} &= \alpha_{ml}^r - \gamma_r \sum_{i=1}^N \frac{\partial R_i}{\partial \alpha_{ml}^{(r)}},\end{aligned}\tag{2.6.9}$$

onde γ_r é a taxa de aprendizado.

Agora escreva (2.6.8) como

$$\begin{aligned}\frac{\partial R_i}{\partial \beta_{km}} &= \delta_{ki} z_{mi}, \\ \frac{\partial R_i}{\partial \alpha_{ml}} &= s_{mi} x_{il}.\end{aligned}\tag{2.6.10}$$

As quantidades δ_{ki} e s_{mi} são erros do modelo nas camadas de saída e escondida, respectivamente. De suas definições, esses erros satisfazem:

$$s_{mi} = h'(\alpha_m^T x_i) \sum_{k=1}^K \beta_{km} \gamma_{ki}.\tag{2.6.11}$$

conhecidas como as equações de retropropagação.

Dessa forma, as atualizações em (2.6.9) podem ser implementadas com um algoritmo de dois estágios. No estágio para frente (*forward*), os pesos são fixados e os valores preditos $\hat{f}_k(x_i)$ são calculados da fórmula (2.6.1). Na etapa para trás, os erros δ_{ki} são calculados e então propagados para trás através de (2.6.11) para obtermos os erros s_{mi} . Ambos os conjuntos de erros são então utilizados para calcular os gradientes para as atualizações em (2.6.9) através de (2.6.10).

Os componentes computacionais para a entropia cruzada tem a mesma forma que os da função de erros quadrados.

As atualizações em (2.6.9) são feitas como uma soma sobre todo o conjunto de treinamento, mas modificações podem ser feitas com o objetivo de atualizar os parâmetros a cada observação, passando por todo o conjunto de treinamento múltiplas vezes, possibilitando o treinamento com um número maior de observações, além da atualização de parâmetros quando novas observações são adicionadas ao conjunto de treinamento [10].

A taxa de aprendizado γ_r é usualmente considerada como uma constante, e pode também ser otimizada de forma a minimizar a função de erro.

As vantagens do método de retropropagação são sua simplicidade e natureza local, onde cada

unidade escondida passa e recebe informação apenas para as unidades que compartilham conexões, podendo dessa forma ser implementado de forma a utilizar computação paralela. Já sua principal desvantagem é a velocidade, que dependendo da aplicação pode ser muito lenta, e por este motivo outros algoritmos de minimização podem também ser utilizados, como gradientes conjugados, entre outros [10].

2.6.4 Problemas no treinamento de Redes Neurais

Alguns pontos devem ser observados ao treinar uma rede neural, já que geralmente o modelo tem muitos parâmetros e o problema de otimização resultante não é convexo. Os pontos a seguir contribuem para um bom ajuste da rede:

- como o problema de otimização não tem solução única, diferentes escolhas de valores iniciais podem levar a soluções locais. Em geral a inicialização dos pesos deve ser feita com números aleatórios pequenos (ex.: $[-0.5, 0.5]$). Além disso é uma boa prática testar modelos com diferentes parâmetros aleatórios iniciais;
- devido a grande quantidade de parâmetros que podem ser obtidos através da adição de camadas e unidades escondidas, devemos tomar precauções para evitar o sobre-ajuste, dessa forma é comum a adição de um termo de regularização λ , que penaliza valores (absolutos) grandes nos parâmetros e/ou muitos parâmetros com valores diferentes de zero.
- não há uma regra prática para a escolha do número ótimo de camadas e unidades escondidas, logo um dos desafios do ajuste do modelo de redes neurais é testar e encontrar uma configuração adequada para o problema em questão.

De forma resumida as vantagens das redes neurais são alta tolerância para dados com ruído, habilidade de classificar padrões aos quais não foi treinada, possibilidade de aprendizado contínuo (aprendendo aos poucos) e capacidade de paralelização do algoritmo.

Suas desvantagens são a dificuldade de interpretação, falta de garantia de convergência para o mínimo global e o fato de ser computacionalmente caro (exigindo muitas vezes longos tempos de treinamento). Além disso, há muitas possibilidades de configuração, sendo que alguns parâmetros devem ser determinados empiricamente, como a topologia da rede.

2.7 Support Vector Machine

O método *Support Vector Machine* (SVM) é uma extensão de dois outros métodos denominados hiperplano separador e hiperplano separador ótimo, descritos a seguir.

2.7.1 Hiperplano Separador

Definição 2.7.1. Hiperplano Separador [10]

Esse método consiste em construir uma fronteira de decisão linear para separar as observações de diferentes classes da melhor forma possível. Para o caso de duas classes (codificadas aqui como 1/-1), essa fronteira pode ser expressa como uma equação da seguinte forma:

$$\beta_0 + x^T \beta = 0, \quad (2.7.1)$$

onde x e $\beta \in \mathbb{R}^L$.

Em algumas aplicações essa fronteira linear pode separar perfeitamente os dados de diferentes classes, a qual denominamos linearmente separável, onde comumente temos infinitas fronteiras possíveis.

No entanto, muitas (ou a maioria) das aplicações não são separáveis, e ao criarmos uma fronteira linear inevitavelmente algumas observações serão mal classificadas. Com base nisso, um critério possível para ajustar o hiperplano consiste em minimizar a distância dos pontos mal classificados à fronteira. (Rosenblatt's perceptron learning algorithm)

A descrição dos modelos seguem conforme apresentada no texto Hastie et. al [10]. Se definirmos que para $x_i^T \beta + \beta_0 \geq 0$ a resposta será classificada como 1 e para $x_i^T \beta + \beta_0 < 0$ como -1, com $x_i \in \mathbb{R}^L$ para $i = 1, \dots, N$, temos que para respostas mal classificadas vale a relação:

$$\begin{cases} y_i = 1 \rightarrow x_i^T \beta + \beta_0 < 0 \\ y_i = -1 \rightarrow x_i^T \beta + \beta_0 \geq 0. \end{cases} \quad (2.7.2a)$$

O objetivo portanto pode ser escrito como minimizar a função:

$$D(\beta, \beta_0) = - \sum_{i \in B} y_i (x_i^T \beta + \beta_0), \quad (2.7.3)$$

onde B indexa o conjunto dos pontos mal classificados. O algoritmo de *stochastic gradient descent* é utilizado para esse problema, e se as classes são separáveis, pode ser demonstrado que o algoritmo converge em um número finito de passos [10].

Problemas com essa abordagem incluem:

- quando as classes são separáveis, existem múltiplas soluções, que são dependentes dos valores iniciais,

- o número finito de passos pode ser muito grande,
- quando as classes não são separáveis, o algoritmo não converge, e ciclos se desenvolvem.

2.7.2 Hiperplano Separador Ótimo

O Hiperplano Separador Ótimo é uma extensão do método anterior, e consiste em criar uma margem em torno da fronteira, e separar as duas classes maximizando a distância ao ponto mais próximo da margem de ambas as classes.

Considere o problema de otimização:

$$\begin{aligned} & \max_{\beta, \beta_0, \|\beta\|=1} M \\ & \text{sujeito à } y_i(x_i^T \beta + \beta_0) \geq M, \quad i = 1, \dots, N. \end{aligned} \tag{2.7.4}$$

O conjunto de condições garantem que os pontos estão à uma distância mínima M da fronteira de decisão definida por β e β_0 , e buscamos os parâmetros que resultam na maior margem possível.

Definindo $\|\beta\| = 1/M$, temos que o problema pode ser reescrito como:

$$\begin{aligned} & \min_{\beta, \beta_0} \frac{1}{2} \|\beta\|^2 \\ & \text{sujeito à } y_i(x_i^T \beta + \beta_0) \geq 1, \quad i = 1, \dots, N. \end{aligned} \tag{2.7.5}$$

Essa restrição define uma margem em torno da fronteira de decisão linear de tamanho $1/\|\beta\|$. Portanto minimizando $\|\beta\|$ estamos maximizando o tamanho da margem.

No entanto, para resolver o problema de classes não-separáveis, é necessário uma modificação adicional, que permite que alguns pontos estejam do lado errado da margem. Através da adição de variáveis de folga (*slack*) $\xi = (\xi_1, \xi_2, \dots, \xi_N)$, podemos modificar as restrições da seguinte forma:

$$y_i(x_i^T \beta + \beta_0) \geq M(1 - \xi_i), \quad \forall i, \quad \xi_i \geq 0, \quad \sum_{i=1}^N \xi_i \leq c. \tag{2.7.6}$$

O valor de ξ_i na restrição $y_i(x_i^T \beta + \beta_0) \geq M(1 - \xi_i)$ representa a quantidade proporcional de quanto o ponto x_i está do lado errado da margem.

Com esse método, a solução para o problema de encontrar um hiperplano separador é única. Colocando um limite superior c em $\sum_{i=1}^N \xi_i$, limitamos a quantidade de pontos do lado errado da margem e as classificações incorretas (quando $\xi_i > 1$).

O problema de minimização é quadrático com restrições lineares, e portanto é um problema de otimização convexo. Abaixo, temos uma solução utilizando multiplicadores de Lagrange [10]. É

conveniente computacionalmente reescrever o problema como:

$$\begin{aligned} \min \frac{1}{2} \|\beta\|^2 + C \sum_{i=1}^N \xi_i \\ \text{sujeito à } y_i(x_i^T \beta + \beta_0) \geq 1 - \xi_i, \forall i, \xi_i \geq 0. \end{aligned} \quad (2.7.7)$$

onde o parâmetro C representa o custo.

A função de Lagrange (primal) tem a forma:

$$L_P = \frac{1}{2} \|\beta\|^2 + C \sum_{i=1}^N \xi_i - \sum_{i=1}^N \alpha_i [y_i (x_i^T \beta + \beta_0) - (1 - \xi_i)] - \sum_{i=1}^N \mu_i \xi_i. \quad (2.7.8)$$

que minimizamos com respeito à β , β_0 , e ξ_i . Igualando as derivadas à 0, temos:

$$\beta = \sum_{i=1}^N \alpha_i y_i x_i; \quad (2.7.9)$$

$$0 = \sum_{i=1}^N \alpha_i y_i; \quad (2.7.10)$$

$$\alpha_i = C - \mu_i, \forall i \quad (2.7.11)$$

com, α_i , μ_i e $\xi_i \geq 0$, $\forall i$.

A função objetivo dual Lagrangiana (Wolfe) é dada por:

$$L_D = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{i'=1}^N \alpha_i \alpha_{i'} y_i y_{i'} x_i^T x_{i'}. \quad (2.7.12)$$

Maximizamos L_D sujeito à $0 \leq \alpha_i \leq C$ e $\sum_{i=1}^N \alpha_i y_i = 0$.

As condições KKT incluem (2.7.9) – (2.7.11), além de:

$$\alpha_i [y_i (x_i^T \beta + \beta_0) - (1 - \xi_i)] = 0,$$

$$\mu_i \xi_i = 0,$$

$$y_i (x_i^T \beta + \beta_0) - (1 - \xi_i) \geq 0,$$

para $i = 1, \dots, N$. Juntas, as equações (2.7.9) – (2.7.12) caracterizam de forma única a solução do problema dual e primal.

De (2.7.9) temos que a solução para β tem a forma:

$$\hat{\beta} = \sum_{i=1}^N \hat{\alpha}_i y_i x_i;$$

E de (2.7.2) e (2.7.2)

- se $\hat{\alpha}_i > 0$ então $y_i (x_i^T \beta + \beta_0) - (1 - \xi_i) = 0$, ou seja, x_i está na borda ou do lado errado da margem;
- se $y_i (x_i^T \beta + \beta_0) - (1 - \xi_i) > 0$, x_i não está na borda da margem, e $\hat{\alpha}_i = 0$.

Logo $\hat{\beta}$ é representado em função dos pontos x_i na borda ou do lado errado da margem, esses pontos são denominados vetores de suporte (ou pontos de suporte). Podemos obter uma regra de classificação a partir desse método, definida como:

$$\hat{G}(x) = \text{sinal} [\hat{f}(x)] = \text{sinal} [x^T \hat{\beta} + \hat{\beta}_0]$$

A Figura 2.6 ilustra a aplicação do método em um problema linearmente separável. Os pontos preenchidos representam os vetores de suporte, a linha sólida representa a fronteira de decisão e as linhas tracejadas correspondem às margens.

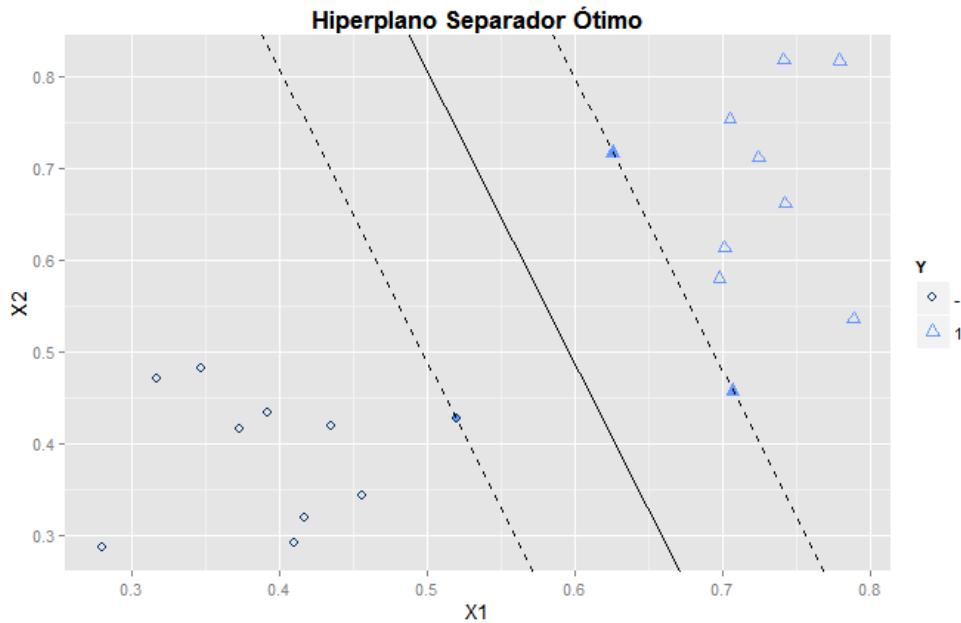


Figura 2.6: Exemplo Hiperplano Separador Ótimo

Já a Figura 2.7 apresenta o mesmo método em um problema não separável linearmente. Note que alguns pontos ficam dentro da margem e outros ficam também do lado errado da fronteira de

decisão. Neste caso, a fronteira linear não é suficiente pra encontrar uma regra de decisão livre de erros.

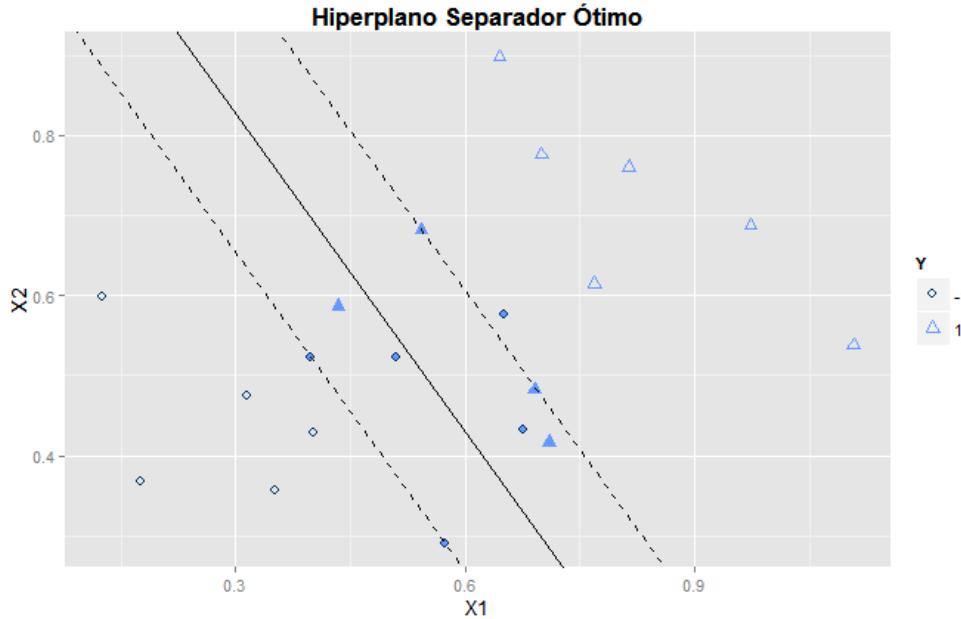


Figura 2.7: Exemplo Hiperplano Separador Ótimo - Classes Não Separáveis

2.7.3 Support Vector Machine

O método Support Vector Machine, consiste em uma modificação no método do Hiperplano Separador Ótimo, onde os atributos de entrada originais x_i são substituídos por transformações da forma $h(x_i)$, onde a função $h(.)$ é escolhida de forma que os produtos internos possam ser calculados facilmente. Com a substituição, a função dual de Lagrange tem a forma:

$$L_D = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{i'=1}^N \alpha_i \alpha_{i'} y_i y_{i'} \langle h(x_i), h(x_{i'}) \rangle.$$

E a função de solução $f(x)$ pode ser escrita como:

$$\begin{aligned} f(x) &= h(x)^T \beta + \beta_0 \\ &= \sum_{i=1}^N \alpha_i y_i \langle h(x), h(x_i) \rangle + \beta_0. \end{aligned}$$

Tanto (2.7.3) quanto (2.7.3) envolvem $h(x)$ apenas através de seu produto interno e, de fato, não precisamos nem especificar a transformação $h(x)$, mas apenas sua função *kernel*:

$$k(x, x') = \langle h(x), h(x') \rangle,$$

Podemos interpretar a técnica SVM considerando que ela consiste em encontrar um hiperplano separador para as classes em um subespaço de dimensão maior que o espaço original [2].

Exemplos de função kernel são:

- kernel linear

$$k(x, x') = \langle x, x' \rangle, \quad (2.7.13)$$

- RBF - *Radial Basis Function*

$$k(x, x') = \exp(-\gamma \|x - x'\|^2), \quad (2.7.14)$$

- kernel polinomial

$$k(x, x') = (s \langle x, x' \rangle + o)^d, \quad (2.7.15)$$

- kernel tangente hiperbólica (Sigmoid)

$$k(x, x') = \tanh(s \langle x, x' \rangle + o). \quad (2.7.16)$$

onde γ , s , o e d são hiper-parâmetros que devem ser selecionados empiricamente, conforme pode ser visto no Capítulo 4.

A Figura 2.8 mostra a aplicação do método SVM com o kernel RBF para os mesmos dados exibidos na Figura 2.7. Os pontos pretos representam os vetores de suporte, e as cores do gráfico de contorno representam os valores da regra de decisão, sendo que a fronteira é representada pela cor branca ($\hat{f}(x) = 0$).

Com a utilização deste kernel as fronteiras passaram a ser não-lineares no espaço original, e os erros de classificação foram eliminados.

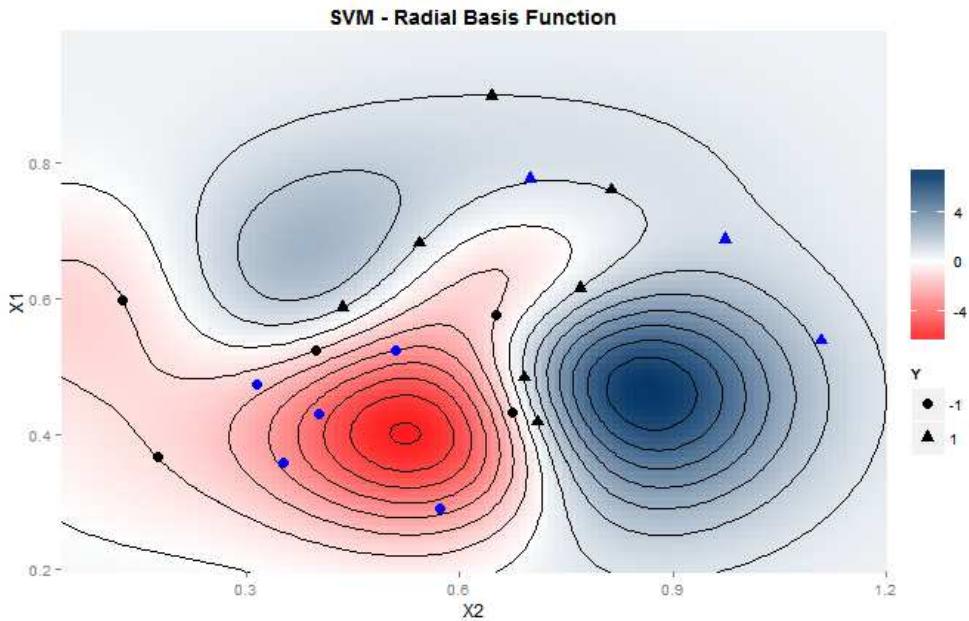


Figura 2.8: Exemplo Support Vector Machine - Kernel RBF

O método SVM é capaz de modelar uma grande variedade de problemas e é, em geral, menos sensível ao sobre-ajuste do que outros métodos [8], embora não elimine totalmente o problema [10]. Dependendo da escolha do kernel é possível obter uma interpretação do modelo, como por exemplo, na utilização do kernel linear, ou polinomial. Outra vantagem do método SVM, ao contrário das redes neurais, é que para uma determinada função kernel e parâmetro de custo fixos, ele sempre encontra uma solução global [8].

Problemas incluem o alto tempo de processamento e a grande variedade de combinações de hiper-parâmetros (como a função kernel e seus parâmetros) que podem ser testadas, não havendo uma regra ótima para a seleção dos mesmos [8].

Capítulo 3

Preparação dos dados e medidas de qualidade do modelo

3.1 Preparação dos dados

A etapa inicial de uma análise de dados consiste em seu pré-processamento, com o objetivo de obter dados com qualidade para as etapas seguintes (como o ajuste do modelo de predição). Diversos fatores englobam o conceito de qualidade de dados como consistência, integridade, completude, periodicidade e interpretabilidade [8].

- Consistência e integridade: relacionados à qualidade da coleta e armazenamento dos dados, sendo que algumas dificuldades decorrem de diferentes unidades de medida utilizadas em um mesmo atributo e erros humanos de entrada de valores.
- Completude: ausência de dados faltantes.
- Periodicidade: frequência dos dados (diária, mensal, etc).
- Interpretabilidade: reflete o quanto simples de entender são os dados.

Como dados reais tendem a ser incompletos, ruidosos e inconsistentes, uma limpeza é usualmente necessária, com a finalidade de completar valores faltantes, suavizar ruídos e identificar *outliers* (valores extremos).

A seguir serão apresentadas as técnicas de preparação de dados utilizadas neste trabalho.

3.1.1 Valores Faltantes

Existem diversas técnicas para o tratamento de dados faltantes, dentre as quais podem ser citadas [8]:

1. ignorar a observação: é feito usualmente quando o atributo-classe é faltante. Não é um método muito efetivo, pois perde-se a informação dos demais atributos, que poderiam ser utilizados de alguma forma para a análise;
2. completar o campo manualmente: é uma abordagem que consome tempo, além disso, dependo da aplicação, nem mesmo um especialista pode conseguir estimar o valor de forma correta;
3. usar uma constante global: substituir por um valor que represente a ausência como “Ausente”. É um método simples, mas os métodos das análises subsequentes podem interpretar este valor como uma característica comum de interesse às observações faltantes, o que pode ser interessante, caso a presença de valor faltante não seja independente (ex.: para indivíduos com determinada doença não foi possível realizar determinado teste físico), mas também pode levar a conclusões errôneas onde essa associação não é válida;
4. usar medidas de tendência central: usar média, mediana ou moda para preencher os valores faltantes;
5. usar medidas de tendência central para todas as observações de uma mesma classe: usar média, mediana ou moda para preencher os valores faltantes, mas segmentando os dados de acordo com o atributo-classe;
6. usar o valor mais provável para preencher o valor faltante: o valor faltante é determinado através de algum modelo, como o de regressão ou árvore de decisão.

Os métodos de 3 a 6 adicionam algum viés aos dados, já que os dados preenchidos podem estar incorretos.

Neste trabalho, para os atributos que contêm dados faltantes, utilizamos o método de substituição pela medida central: média para atributos numéricos contínuos, mediana para numéricos discretos, e moda para atributos nominais.

Esse método foi escolhido para que pudéssemos utilizar todas as observações disponíveis para o ajuste e avaliação dos modelos e por sua facilidade de aplicação. Os métodos que utilizam de alguma forma a informação do atributo-classe na obtenção do valor a ser preenchido (métodos 5 e

6) foram descartados, pois como temos também que preencher informações faltantes no conjunto de testes, sua utilização acarretaria em uma avaliação otimista das métricas de qualidade de ajuste.

3.1.2 Técnicas para balanceamento de classes

Em muitas aplicações, o atributo-classe é desbalanceado, o que significa que há um número de observações maior em determinada categoria. Como exemplo, considere um problema de classificação de SPAM, onde 99% dos e-mails são SPAM. Um algoritmo que busca minimizar o erro de ajuste, poderia facilmente encontrar uma regra com apenas 1% de erro, consistindo em classificar todos os e-mails como SPAM, no entanto o modelo resultante seria inútil.

Para contornar este problema, existem técnicas para rebalanceamento de classes através de amostragem. Dentre as quais destacamos:

- realização de amostragem com reposição, e com seleção de $k \in N$ elementos de cada uma das classes. Obtendo-se desta forma o mesmo número de observações para cada classe;
- realização de amostragem sem reposição, mas reduzindo o número de observações na classe majoritária.

Ao igualarmos o número de observações em cada classe, os modelos que buscam minimizar o erro de ajuste, como a regressão logística e redes neurais, terão que discriminar bem as duas classes para atingir um baixo erro no conjunto de treinamento.

Já para um modelo que funciona por votos das observações mais próximas, como o KNN, sem a amostragem a maioria dos vizinhos seria, em geral, da classe majoritária. Ao equilibrarmos o número de observações entre as classes, nós contornaríamos esse problema.

Além disso, como visto no exemplo anterior, especial atenção deve ser dada na escolha do método de avaliação da qualidade do ajuste ao lidar com classes desbalanceadas, pois se considerarmos apenas a taxa de erro geral, podemos selecionar um modelo que não discrimina as classes de forma satisfatória.

3.1.3 Técnicas para seleção de atributos

A seleção de atributos pode ser realizada de duas formas [8]:

- na fase de pré-processamento dos dados, utilizando técnicas como Chi-Quadrado, Análise de Componentes Principais, decomposição SVD;
- utilizando os critérios específicos para o algoritmo de classificação aplicado. (Ex.: teste de significância de parâmetros na Regressão Logística, poda em Árvores de Decisão, etc).

3.1.4 Transformação de Variáveis

Algumas das técnicas como KNN (depende da medida de distância) requerem que as variáveis estejam na mesma escala. Para isso algumas transformações podem ser aplicadas:

- normalização em um intervalo especificado (ex.: $[0, 1]$);
- normalização através da distribuição normal $Z_i = (X_i - \mu)/\sigma$, onde μ é a média e σ o desvio-padrão de X ;
- outras transformações podem ser úteis quando há valores muito grandes: $Z_i = \log(X_i)$;
- transformações em potência podem ser úteis quando para modelar comportamentos não lineares: $Z_i = X_i^p$;

Dentre as transformações de potência mais utilizadas, temos as transformações de Box-Cox (para valores estritamente positivos) e Yeo-Johnson [28], que buscam encontrar a potência p que aproxime a distribuição dos dados da normalidade.

- multiplicação de variáveis podem modelar a interação entre as mesmas: $Z_i = X_i * Y_i$.

3.1.5 Regularização

A regularização tem como objetivo prevenir o sobre-ajuste [10], penalizando modelos com muitos parâmetros e/ou com pesos dos parâmetros muito grandes, dando prioridade para um modelo equilibrado (parcimonioso).

3.2 Métodos para avaliar a qualidade do ajuste

Para problemas de classificação é natural medir o desempenho do classificador em termos da taxa de erro. O classificador prediz a classe de cada observação, se a predição é correta é considerado um Sucesso, caso contrário um Erro. A taxa de erro é a proporção de erros sobre o tamanho do conjunto de observações onde foram feitas as predições, e mede o desempenho médio do classificador [29].

Em geral, estamos interessados no desempenho do modelo em um conjunto de novos dados, que não foi utilizado durante o treinamento do modelo, simulando a aplicação prática, onde o modelo é ajustado e depois predições são feitas para dados novos, onde não há uma resposta conhecida. Se utilizarmos as mesmas observações para treinar o modelo e avaliá-lo, podemos incorrer no chamado

erro de resubstituição. Essa estimativa do erro é uma estimativa otimista da verdadeira taxa de erro, já que não utiliza nenhuma observação que não foi utilizada para treinar o classificador [8].

Para obtermos estimativas confiáveis para a taxa de erro podemos utilizar técnicas de particionamento do conjunto de dados, onde retira-se parte do conjunto de dados, reservando-o apenas para o teste do desempenho do classificador [29]. Há diversas formas de realizar essa divisão do conjunto de dados, entre elas:

3.2.1 ***Hold-out***

O método *hold-out* consiste em partitionar o conjunto de dados em dois subconjuntos independentes, denominados conjunto de treinamento e conjunto de testes. Tipicamente dois terços das observações são alocadas como o conjunto de treinamento, e o restante como conjunto de testes. O modelo é construído com base no conjunto de treinamento e em seguida as medidas de desempenho são calculadas no conjunto de testes [8]

3.2.2 **Validação Cruzada (*K-Fold Cross Validation*)**

Na validação cruzada *K-Fold*, o conjunto de dados inicial é partitionado aleatoriamente em k subconjuntos ou *folds*, D_1, D_2, \dots, D_k mutuamente exclusivos de tamanho aproximadamente igual. Treinamento e teste são realizados k vezes, e para cada iteração i , o subconjunto D_i é utilizado como teste, e os demais subconjuntos são utilizados para o treinamento do modelo [8].

Embora não haja um consenso absoluto na escolha do parâmetro k , diversos estudos práticos tem obtido as melhores estimativas da taxa de erro com a utilização de $k = 10$ [29].

3.2.3 **Treinamento, Validação e Teste**

Uma variação do método *hold-out* consiste em dividir o conjunto de dados em três subconjuntos: conjunto de treinamento, conjunto de validação e conjunto de testes. É útil quando queremos comparar diversas configurações de um mesmo método (como variações em valores de parâmetros), pois ajusta-se os diversos modelos no conjunto de treinamento. Em seguida, o melhor modelo é selecionado de acordo com seu desempenho no conjunto de validação, e o último passo é a estimação das medidas no conjunto de teste, não utilizado para a escolha de nenhum parâmetro ou ajuste do modelo, evitando dessa forma a superestimação do desempenho [19]. Percentuais usualmente utilizados para a divisão são 60 – 20 – 20.

Outra variação possível é a combinação deste conceito e da validação cruzada, reservando uma parte dos dados para o teste final, e aplicando *K-Fold* no restante das observações para a escolha

dos parâmetros e configurações do modelo [19].

3.2.4 Medidas

A seguir listamos as medidas utilizadas para avaliação dos métodos [29]. Uma observação é que a classe escolhida como resultado positivo deve ser a classe com o menor número de observações para o caso de classes desbalanceadas [19].

- Verdadeiro Positivo (VP): contagem de observações corretamente classificadas como positivo;
- Falso Positivo (FP): contagem de observações classificadas como positivo, mas que são negativos;
- Verdadeiro Negativo (VN): contagem de observações corretamente classificadas como negativo;
- Falso Negativo (FN): contagem de observações classificadas como negativo mas que são, na verdade, positivas;
- Acurácia: Mede o desempenho médio do classificador

$$\text{Acurácia} = \frac{VP + VN}{N};$$

onde N é o número de observações classificadas

- Precisão: mede a porcentagem de acertos entre as observações classificadas como positivas (pode ser aplicada para a classe negativa também):

$$\text{Precisão} = \frac{VP}{VP + FP};$$

- Sensibilidade: também chamado de *Recall* ou taxa de verdadeiro positivo, mede a porcentagem das observações positivas que foram corretamente classificadas:

$$\text{Sensibilidade} = \frac{VP}{VP + FN};$$

quando aplicada para a classe negativa essa métrica tem o nome de Especificidade.

- F1 score: essa medida busca um equilíbrio entre a sensibilidade e a precisão.

$$F1 = \frac{2 * Precisão * Sensibilidade}{Precisão + Sensibilidade}.$$

- Kappa

$$\text{Kappa} = \frac{Acurácia - Acurácia_e}{1 - Acurácia_e}.$$

onde $Acurácia_e$ é a acurácia observada em um classificador aleatório.

Tabela de Contingência

A tabela de contingência fornece uma ferramenta de comparação visual do desempenho do modelo:

		Tabela 3.1: Tabela de Contingência - Logística	
		Classificado como Positivo	Classificado como Negativo
Positivo	VP		FN
	FP		VN

Curva ROC

A Curva ROC é uma alternativa de avaliação do classificador, que consiste na utilização de um gráfico bi-dimensional, onde o eixo vertical representa a taxa de verdadeiros positivos e o eixo horizontal representa a taxa de falsos positivos obtida pelo classificador [22].

A interpretação é que quanto mais próximo do ponto (0, 1) melhor é o modelo. Modelos no ponto (0, 0) não classificam nenhuma observação como positivo, e modelos próximos do ponto (1, 1) classificam tudo como positivo.

Uma variação é a curva ROC que consiste em avaliar o desempenho do modelo para diferentes critérios de classificação, variando a probabilidade estimada a partir da qual uma informação é classificada como positiva.

A Figura 3.1 ilustra uma curva ROC. Quanto maior a área abaixo da curva, melhor o desempenho médio do modelo.

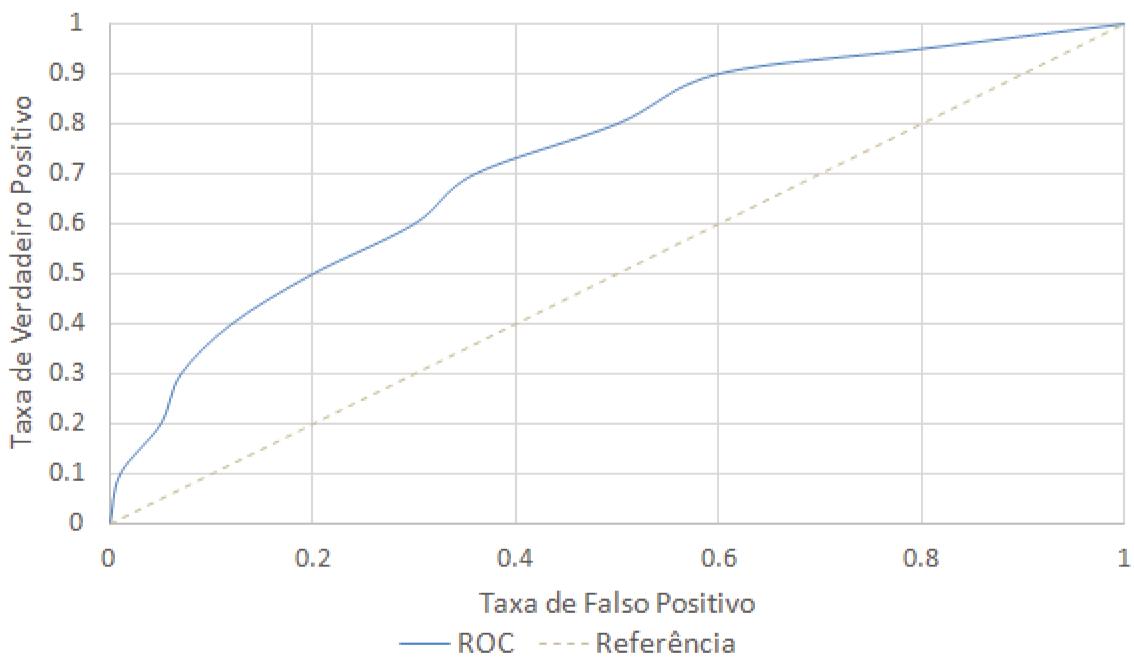


Figura 3.1: Exemplo Curva Roc

Interpretação dos valores das medidas de qualidade de ajuste

Uma observação importante para as medidas citadas, é que o nível para o qual podemos considerar um valor como bom depende do problema modelado.

Problemas mais simples possibilitam um melhor ajuste. Logo um modelo com, por exemplo, área ROC de 0.9 poderia ser ainda superado. No entanto em problemas mais complexos, com um maior ruído inerente nos dados ou em que atributos importantes estejam faltando, pode ser que não seja possível encontrar modelos com valores muito bons nas medidas de qualidade.

A determinação dos níveis aceitáveis para o modelo, deve ser feita caso a caso, baseada no custo de uma classificação errônea.

Capítulo 4

Aplicações

Neste capítulo são apresentadas três aplicações, todas envolvendo a aplicação de modelos de classificação binária à problemas encontrados em instituições financeiras. Os softwares utilizados para as análises foram o R [26] e RStudio [13].

O problema número um, cujo objetivo envolve a seleção de clientes de um banco de Portugal para uma campanha de marketing direto, será utilizado para apresentar os detalhes dos procedimentos utilizados. Considerando que os demais problemas seguem a mesma metodologia, o foco para ambos será nos resultados obtidos.

Todos os dados foram obtidos do repositório da Universidade da Califórnia, Irvine (UCI) [3].

4.1 Procedimentos

Na lista abaixo estão elencados os passos utilizados em todos os problemas para pré-processamento dos dados e separação dos conjuntos para treinamento, validação e teste.

1. Análise exploratória:

- reconhecimento da base de dados e identificação de problemas.

2. Separar Dados de Teste:

- 80% dos dados para treinamento;
- 20% para teste.

3. Preparação dos dados:

- dados faltantes: substituição dos dados faltantes pela média ou mediana do conjunto de treinamento. Os dados faltantes no conjunto de testes também foram preenchidos com a média obtida no conjunto de treinamento, de forma a não adicionar viés na avaliação dos resultados de teste;
- atributos discretos: alguns algoritmos requerem dados já convertidos em formato de indicadores, outros lidam internamente com a conversão;
- balanceamento de classes.

4. Seleção dos parâmetros e variáveis do modelo utilizando o conjunto de treinamento:

- para a regressão logística, o método adotado consiste em selecionar a combinação de atributos que minimizam o AIC, utilizando o conjunto de treinamento balanceado;
- para os demais métodos, foi utilizado o método *K-fold* no conjunto de treinamento desbalanceado, com 10 subconjuntos (*folds*). Para cada iteração, o subconjunto de validação foi mantido inalterado, e os demais subconjuntos de treinamento foram balanceados. As medidas analisadas para a seleção dos parâmetros/atributos foram Acurácia e *F1*.

5. Validação e Comparação dos resultados de todos os métodos no conjunto de testes.

4.2 Banco Português

Os dados foram obtidos do repositório de dados da UCI, e são provenientes de campanhas de marketing promovidas por uma instituição financeira portuguesa entre maio/2008 e novembro/2010 [18], com o objetivo de vender uma aplicação bancária de longo-prazo com boa taxa de remuneração (depósito a termo). A base de dados, chamada The Bank Marketing Data Set, contém 45211 observações e 17 atributos, incluindo o atributo-classe que indica se o cliente comprou ou não o produto. A forma de contato foi via telefone. Na Tabela 4.1 temos a descrição detalhada de todos os atributos. Note que há atributos relacionados às características do cliente (atributos 1 a 8), à campanha atual (atributos 9 a 13), e ao histórico de transações com o banco (atributos 14 a 16):

Tabela 4.1: Descrição do Banco de Dados - Banco Português

Num	Variável	Tipo	Descrição	Dados Faltantes
1	X_1	Numérico	Idade	Não
2	X_2	Nominal	Profissão	Sim (1%)
3	X_3	Nominal	Estado Civil	Não
4	X_4	Ordinal	Nível educacional	Sim (4%)
5	X_5	Binário	Tem empréstimo em atraso	Não
6	X_6	Numérico	Balanço Médio Anual	Não
7	X_7	Binário	Tem financiamento imobiliário	Não
8	X_8	Binário	Tem empréstimo pessoal	Não
9	X_9	Nominal	Meio de comunicação da campanha	Sim (29%)
10	X_{10}	Numérico	Dia do último contato	Não
11	X_{11}	Nominal	Mês do último contato	Não
12	X_{12}	Numérico	Duração do contato (segundos)	Não
13	X_{13}	Numérico	Número de contatos dessa campanha	Não
14	X_{14}	Numérico	Número de dias passados depois que o cliente foi contatado para outra campanha. (-1 significa que não foi contatado)	Não
15	X_{15}	Numérico	Número de contatos feitos antes dessa campanha para esse cliente.	Não
16	X_{16}	Nominal	Resultado da última campanha	Não
17	Y	Binário	Contratou o produto (depósito a termo)	Não

4.2.1 Análise Exploratória

O atributo classe é binário e desbalanceado, contendo 11.7% de respostas positivas.

Para os atributos numéricos, a maior correlação linear ocorre entre os atributos X_{14} e X_{15} , com valor de 0.45 e a segunda maior entre os atributos X_{10} e X_{13} , com valor de 0.16. Como os valores são baixos, optamos por manter todas as variáveis.

Para os atributos numéricos, foi aplicada a transformação de potência Yeo-Johnson, com o objetivo de aproximar a distribuição dos atributos numéricos de uma distribuição normal. Os atributos com essa transformação serão indicados através do índice sobrescrito YJ (ex.: X^{YJ}). Estes atributos extras transformados serão utilizados apenas na regressão logística e Naive Bayes. O Naive Bayes baseia-se na hipótese de normalidade das variáveis numéricas, logo a transformação pode melhorar o desempenho do mesmo. Para a regressão logística, a adição da variável transformada pode ser útil já que o algoritmo não lida naturalmente com fronteiras não lineares (a priori

não é possível saber se a transformação é necessária).

Os dados foram divididos em dois conjuntos de forma aleatória: Treinamento (80%) e Teste (20%).

4.2.2 Regressão Logística

Para a regressão Logística o método de seleção utilizado foi da minimização do AIC, conforme visto na Subseção 2.2.2, utilizando step-wise (stepAIC do pacote MASS [27]) do R aplicado no conjunto de treinamento balanceado.

Além disso após o seleção do modelo foi avaliado o VIF [14] (Fator de Inflação da Variância), para diagnóstico de multi-colinearidade, resultando na remoção dos atributos com VIF maior que 5, listados na Tabela 4.2:

Tabela 4.2: Fator de inflação da variância - Português

Variável	VIF
X_1^{YJ}	42.36
X_{14}^{YJ}	1459.41
X_{15}^{YJ}	1130.79
X_6^{YJ}	7.44
X_{12}^{YJ}	8.64

O modelo final selecionado pode ser descrito como:

$$P[Y = 1] = f(X_1, X_2, X_3, X_4, X_6, X_7, X_8, X_9, X_{11}, X_{12}, X_{16}, X_{13}^{YJ}). \quad (4.2.1)$$

onde $f(\cdot) = g(X\beta)$

Através da análise dos resíduos foram removidas 33 observações consideradas extremas. A Figura 4.1 mostra os resíduos, alavancagem (*leverage*) e a distância de Cook (Subseção 2.2.1), para o modelo construído após a remoção das 33 observações, e não indica a presença de outros pontos influentes.

A Tabela 4.3 mostra o teste Chi-Quadrado com H_0 sendo o modelo completo, versus H_1 o modelo selecionado. A coluna Dif. G.l. representa a diferença nos graus de liberdade entre H_1 e H_0 e a coluna Dif. Deviance representa a diferença na Deviance. Podemos rejeitar H_0 com nível de significância 1%, o que indica que embora o modelo ajustado tenha o melhor AIC, ela perde alguma informação segundo o critério da Deviance, que não penaliza a complexidade do modelo. No entanto, optamos por manter o modelo de menor AIC e sem as variáveis que contribuem para a presença de multicolinearidade.

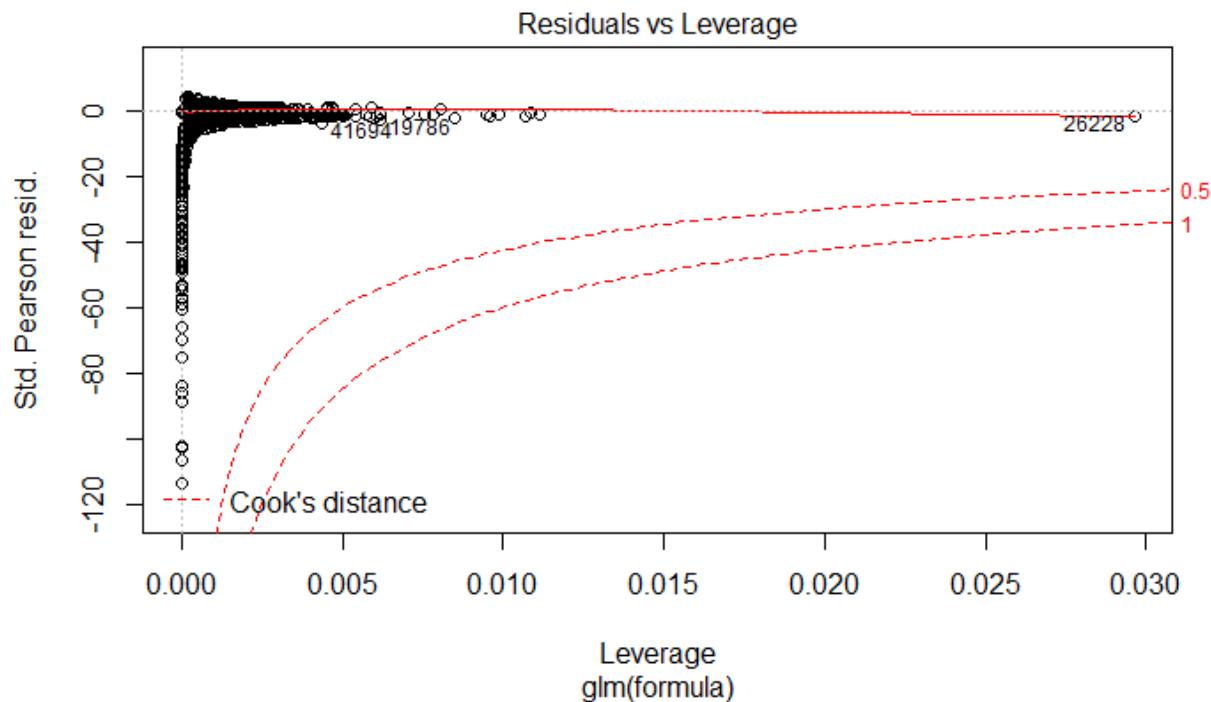


Figura 4.1: Resíduos vs Alavancagem - Português

Tabela 4.3: Teste Chi-Quadrado - Português

Modelo	Graus de Liberdade	Deviance	Dif. G.l.	Dif. Deviance	P-valor
H_0	63795	48884			
H_1	63806	50977	-11	-2092.9	< 2.2e-16

A Tabela 4.4 apresenta o AIC e Deviance do modelo nulo e ajustado:

Tabela 4.4: AIC e Deviance - Logística - Português

Deviance Modelo Nulo:	88504 com 63841 graus de liberdade (g.l.)
Deviance:	50977 com 63806 g.l.
AIC:	51049

Conforme esperado, podemos observar que houve uma grande redução da Deviance, comparado a um modelo apenas com o termo constante.

Após ajustado o modelo, o aplicamos no conjunto de testes, de forma a avaliar seu desempenho nos dados novos.

A Tabela 4.5, de contingência, mostra em termos absolutos quantas unidades de teste foram classificadas em cada categoria e se de forma correta ou não:

Tabela 4.5: Tabela de Contingência - Logística - Português

	Classificado como Positivo	Classificado como Negativo
Positivo	843	214
Negativo	1244	6741

Na Tabela 4.6 estão apresentadas as medidas de qualidade de ajuste para o modelo em questão.

Tabela 4.6: Tabela de Resultados - Logística - Português

Tipo	Acurácia	Área ROC	Kappa
Teste	0.838	0.9024109	0.4510674

Os resultados obtidos pela regressão logística foram razoáveis, no entanto, foram inferiores aos obtidos por outros métodos.

4.2.3 SVM

Para o SVM, foi utilizada a versão do pacote e1071 [15] do R. Foi realizada uma busca em grade (*grid-search*) preliminar para selecionar um intervalo de busca dos parâmetros e o tipo de *kernel*. A busca foi realizada usando o método *10-fold*.

Devido a quantidade de dados de treinamento, e ao tempo necessário para cada iteração do SVM, a busca foi realizada com uma amostra aleatória de tamanho 5000 do conjunto de treinamentos e apenas para o *kernel* Radial, que apresentou os melhores resultados em testes preliminares.

As configurações avaliadas na busca em grade foram as seguintes:

- kernel: Radial;
- *gamma*: sequência no intervalo [0.01, 2], com passos de tamanho 0.25;
- *custo*: sequência no intervalo [0.01, 3.01], com passos de tamanho 0.5.

Os gráficos de contorno para as métricas acurácia e F1, em função dos parâmetros *gamma* e *custo* podem ser observados nas Figuras 4.2 e 4.3.

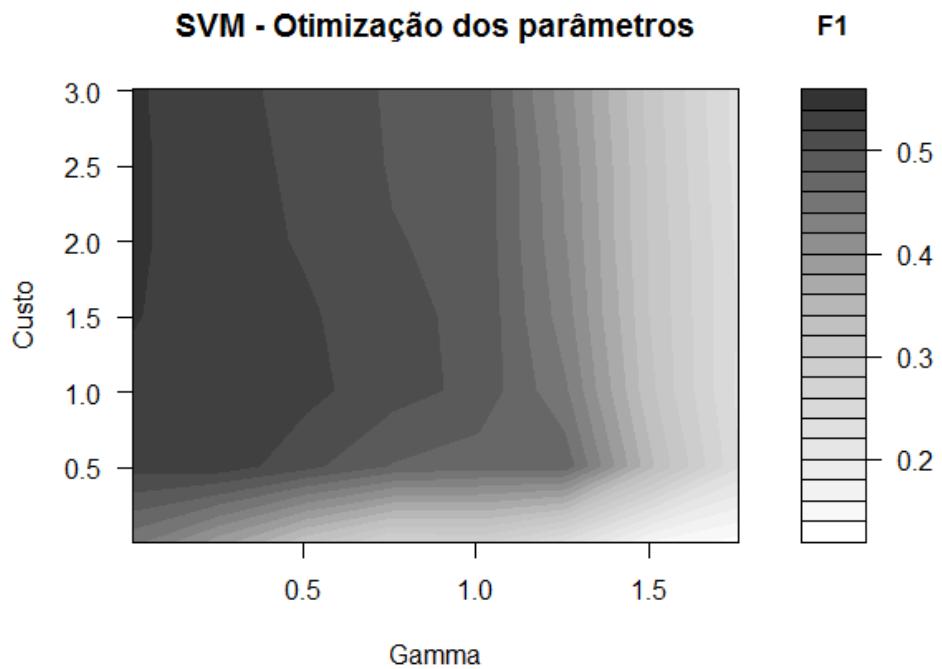


Figura 4.2: Seleção de Modelo - SVM F1 - Português

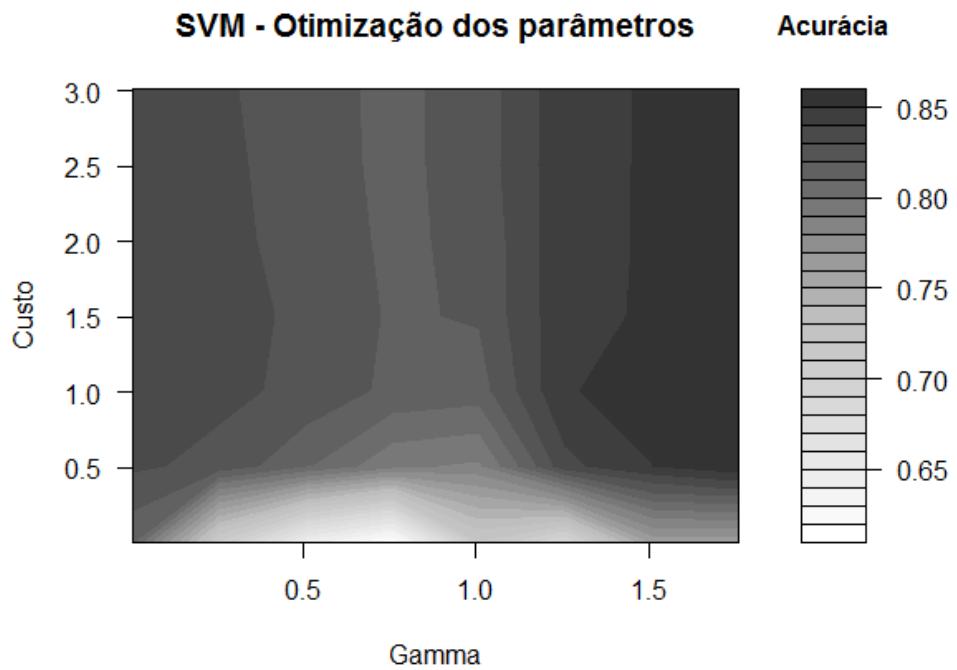


Figura 4.3: Seleção de Modelo - SVM - Português

Através da inspeção do gráfico e das tabelas de resultados (ver Tabelas A.1 e A.2 no Apêndice A), foi possível selecionar um ponto de equilíbrio entre acurácia e F1, utilizando $\gamma = 0.01$ e $custo = 2.51$.

O próximo passo foi o reajuste do modelo com todos os dados de treinamento e predição para os dados de teste. As métricas área ROC e Kappa foram calculadas apenas para o conjunto de testes, para todos os modelos ajustados.

O resultado está nas Tabelas 4.7 e 4.8:

Tabela 4.7: Tabela de Contingência - SVM - Português

	Classificado como Positivo	Classificado como Negativo
Positivo	891	166
Negativo	1241	6744

Tabela 4.8: Tabela de Resultados - SVM - Português

Tipo	Acurácia	F1 (+)	Área ROC	Kappa
Validação (K-Fold)	0.8379180	0.5423520	-	-
Teste	0.8443928	0.5587959	0.9167499	0.4770572

Podemos observar que os resultados para o conjunto de treinamento de testes são próximos para as medidas consideradas, além disso os resultados são superiores aos do modelo de regressão logística.

4.2.4 ANN

Para a rede neural foi utilizada a função nnet do pacote nnet [27] do R. A busca em grade para a rede neural seguiu o mesmo formato adotado no método SVM, consistindo em uma amostragem aleatória de 5000 observações para o treinamento. As seguintes combinações de parâmetros foram utilizadas:

- $tamanho$ = sequência no intervalo [2, 14], com passos de tamanho 2;
- $decay$ = sequência no intervalo [0.1, 3.1], com passos de tamanho 0.5.

Os pesos foram inicializados com valores aleatórios no intervalo $[-0.1, 0.1]$, o número máximo de iterações foi definido como 10000, e o método de minimização foi o de entropia (máxima verossimilhança condicional) [27].

O parâmetro *tamanho* controla o número de unidades da camada escondida (apenas modelos com uma camada escondida foram testados), já o parâmetro *decay* é utilizado em cada iteração para reduzir o valor dos parâmetros proporcionalmente ao seu tamanho [27], com o objetivo de penalizar pesos grandes e reduzir o sobre-ajuste.

As Figuras 4.4 e 4.5 apresentam os resultados da busca em grade, na forma de gráficos de contorno para a Acurácia e F1 em função dos valores dos parâmetros.

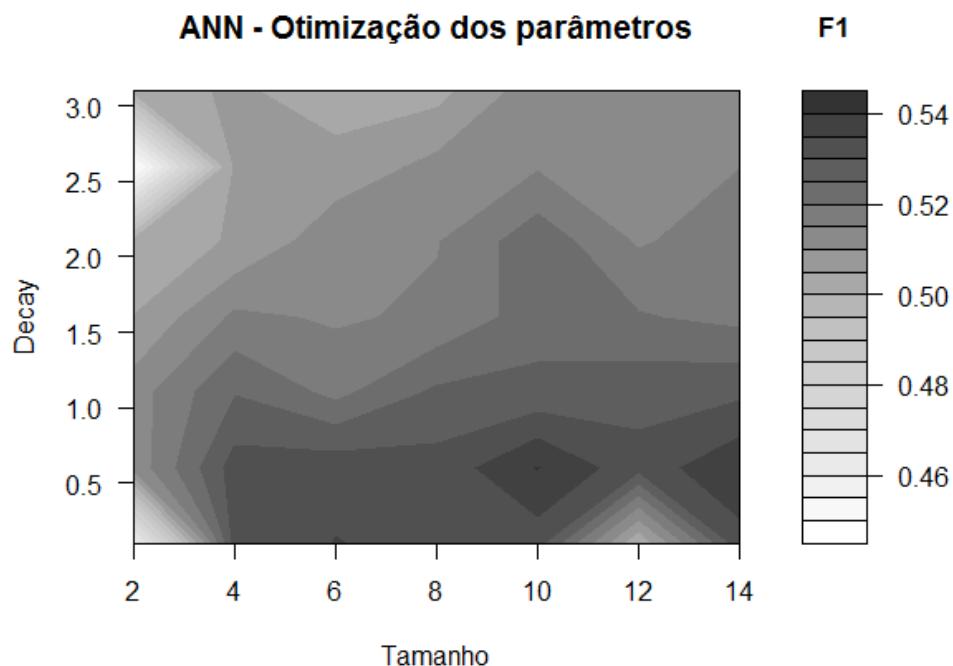


Figura 4.4: Seleção de Modelo - ANN F1 - Português

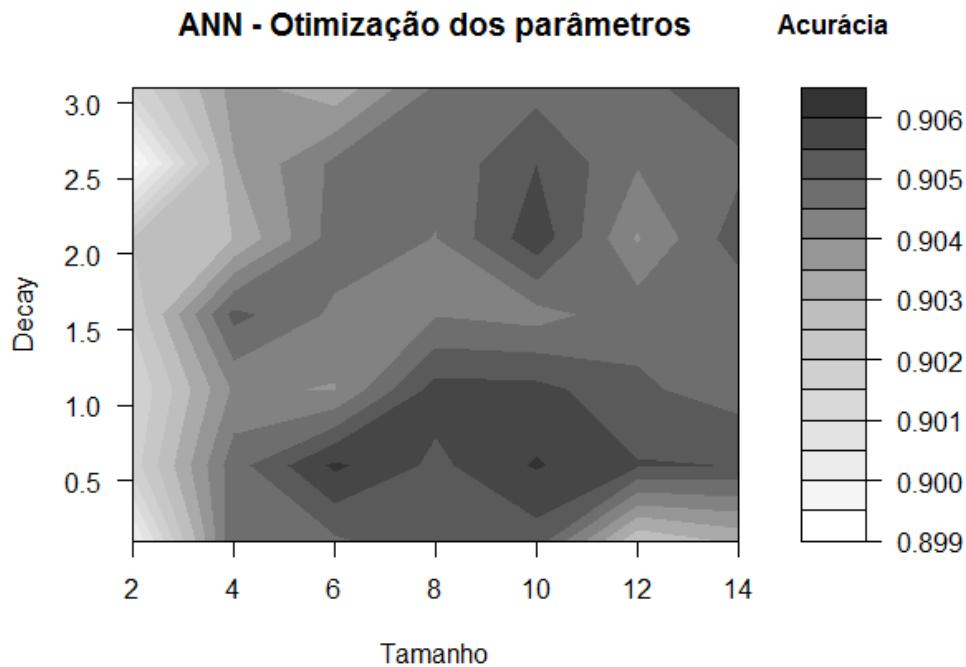


Figura 4.5: Seleção de Modelo - ANN - Português

Através da análise dos gráficos e das tabelas de resultados (ver Tabelas A.3 e A.4 no Apêndice A), e buscando um equilíbrio entre acurácia e F1, os parâmetros selecionados foram $tamanho=10$ e $decay=0.6$.

A estrutura selecionada da rede foi portanto de 40 nós de entrada, dez nós na camada escondida e 1 nó de saída, que junto com os termos constantes das camadas de entrada e interna resultam em 421 pesos para serem estimados.

Os resultados do modelo final no conjunto de testes estão nas Tabelas 4.9 e 4.10:

Tabela 4.9: Tabela de Contingência - ANN - Português

	Classificado como Positivo	Classificado como Negativo
Positivo	914	143
Negativo	1352	6633

Tabela 4.10: Tabela de Resultados - ANN - Português

Tipo	Acurácia	F1 (+)	Área ROC	Kappa
Validação (K-Fold)	0.9060514	0.5403172	-	-
Teste	0.8346605	0.5501053	0.9176864	0.4647743

Os resultados de acurácia no conjunto de validação foram os melhores até aqui, no entanto o desempenho no conjunto de testes teve uma significante perda, indicando que houve um sobre-ajuste, mesmo assim o resultado no conjunto de testes é similar aos modelos anteriores.

4.2.5 KNN

Para o modelo KNN foi utilizado o pacote RWeka [11] do R, pois o mesmo suporta atributos nominais. O pacote também lida com a normalização dos atributos numéricos.

Assim como nos métodos anteriores, devido ao tempo de ajuste, a busca em grade para seleção da configuração foi realizada com 5000 observações de treinamento. Além disso, não foi possível realizar uma busca exaustiva em todas as combinações de parâmetros, logo alguns testes preliminares foram realizados, e a configuração inicial selecionada foi:

- peso igual a 1/distância;
- $k = 90$.

Com essa configuração fixa, foram selecionados os parâmetros de forma *step-forward*, ou seja, partindo de um modelo com um atributo, e adicionando um novo atributo a cada passo, sendo selecionado a cada passo aquele atributo que contribuiu para o melhor valor de F1, na média dos 10 subconjuntos. O modelo resultante foi:

$$Y = f(X_{12}, X_{16}, X_{11}, X_{10}, X_6) \quad (4.2.2)$$

Em seguida, fixando os atributos foi realizada uma nova busca para escolha do k .

Através da inspeção da Figura 4.6 e das tabelas de resultados. O valor de k escolhido foi 1410. Em seguida foi construído o modelo com a configuração escolhida e com todos os dados de treinamento. Os resultados estão nas Tabelas 4.11 e 4.12:

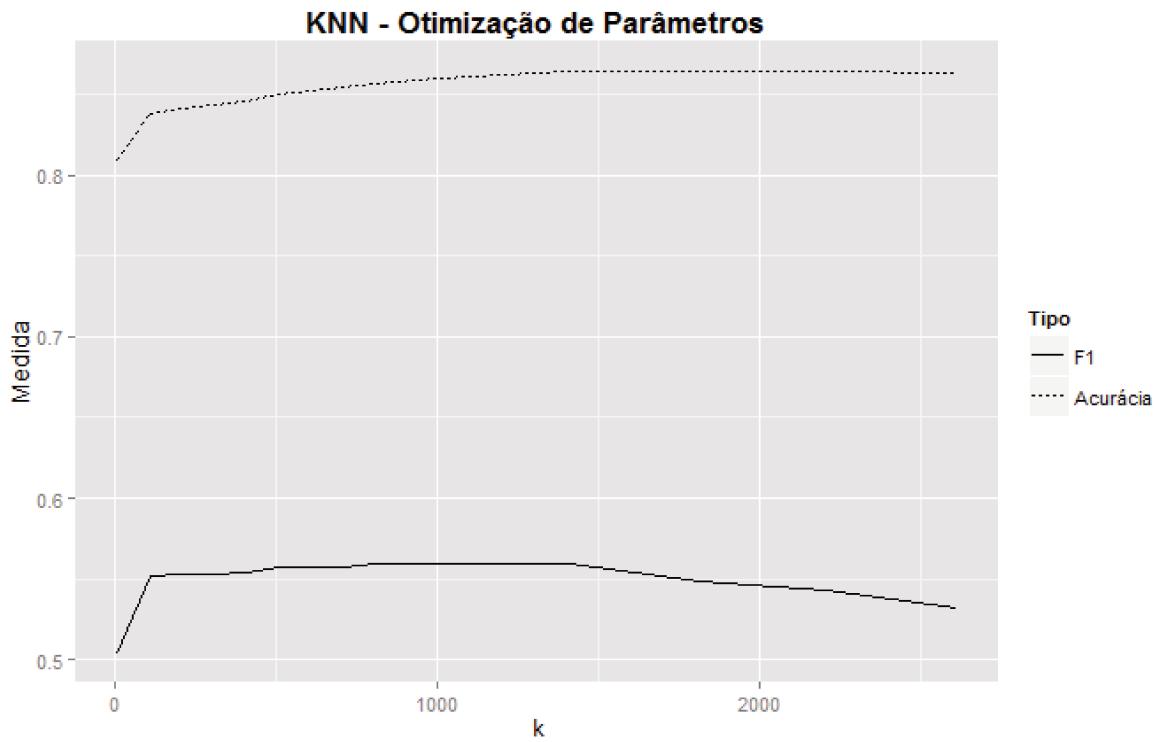


Figura 4.6: Seleção de Modelo - KNN - Português

Tabela 4.11: Tabela de Contingência - KNN - Português

	Classificado como Positivo	Classificado como Negativo
Positivo	779	278
Negativo	927	7058

Tabela 4.12: Tabela de Resultados - KNN - Português

Tipo	Acurácia	F1 (+)	Área ROC	Kappa
Validação (K-Fold)	0.8647237	0.5598669	-	-
Teste	0.866733	0.5638798	0.9065523	0.4903011

Observamos que os resultados do KNN são melhores que os dos modelos anteriores, além disso o desempenho no conjunto de validação foi próxima da obtida no conjunto de testes, indicando uma boa capacidade de extração do modelo.

4.2.6 Árvore de Decisão

Para o modelo de árvore de decisão foi utilizado o pacote tree [23] do R, com os métodos tree para o modelo e prune.tree para realizar a poda com o número indicado de folhas.

A árvore do R é criada a partir de partições binárias, usando a resposta na fórmula especificada e escolhendo divisões (quebras) nos atributos. Variáveis numéricas são divididas em $X < a$ e $X > a$. Os níveis de variáveis categóricas são divididos em dois grupos não vazios. A separação é escolhida de forma a maximizar a redução na impureza. O processo é repetido até que os nós terminais sejam muito pequenos ou muito poucos para dividir. O critério de impureza utilizado em todos os modelos foi o de Entropia.

Para a seleção do tamanho de nós terminais, de forma similar aos métodos anteriores, foi utilizado o método de *K-Fold* ($k = 10$), otimizando a árvore em relação às estatísticas Acurácia e F1.

A Figura 4.7 mostra o desempenho para diferentes tamanhos de árvore:

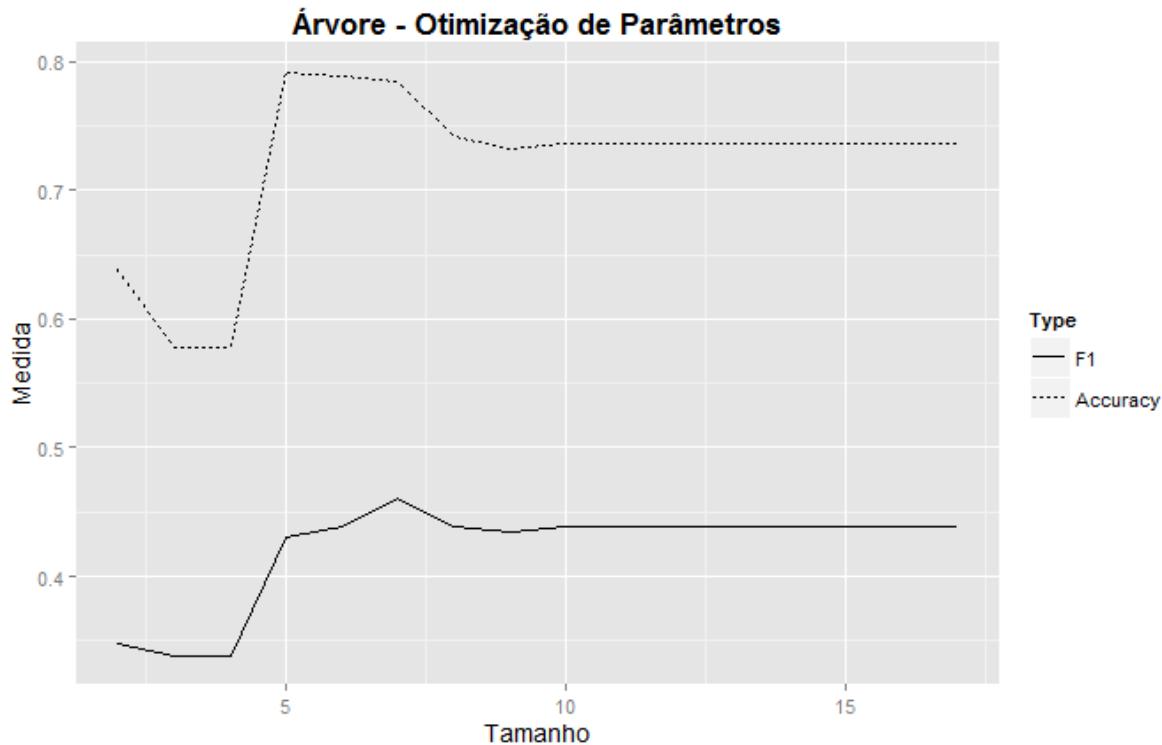


Figura 4.7: Seleção de Modelo - Árvore - Português

O modelo selecionado foi com 7 folhas, como pode ser visto na Figura 4.8:

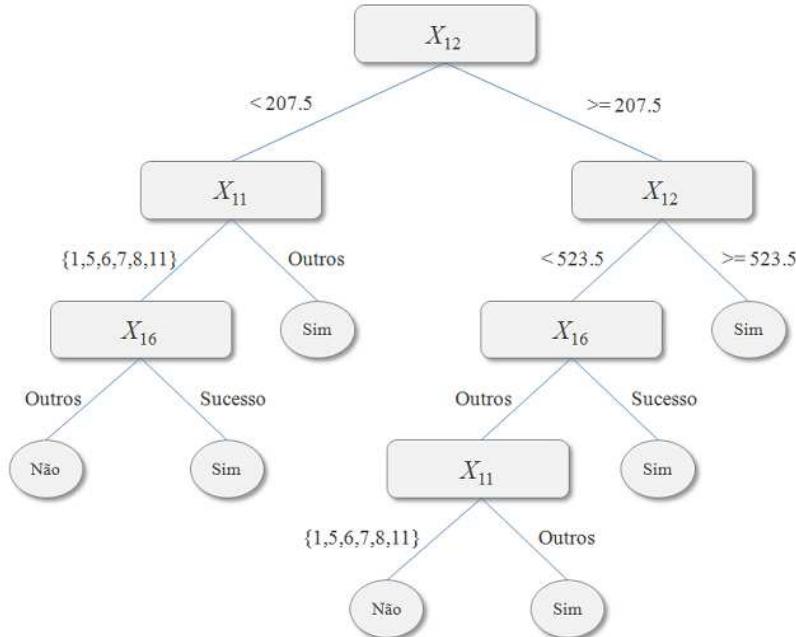


Figura 4.8: Árvore de Decisão - Português

Com o modelo selecionado, o desempenho do mesmo foi avaliado no conjunto de testes:

Tabela 4.13: Tabela de Contingência - Árvore de Decisão - Português

	Classificado como Positivo	Classificado como Negativo
Positivo	805	252
Negativo	1646	6339

Tabela 4.14: Tabela de Resultados - Árvore de Decisão - Português

Tipo	Acurácia	F1 (+)	Área ROC	Kappa
Validação (K-Fold)	0.7836175	0.4384781	-	-
Teste	0.7900907	0.458951	0.848587	0.3533135

Como pode ser observado, o modelo de árvore de decisão não obteve um bom desempenho, quando comparado com os demais modelos.

4.2.7 Naive Bayes

Para o método Naive Bayes (NB), foi utilizada a função naiveBayes do pacote e1071 [15] do R.

Foi realizada uma seleção de variáveis utilizando o método *K-Fold* ($k = 10$), e selecionando de forma *step-forward* as variáveis com maior valor da estatística F1.

A Figura 4.9 mostra o resultado em relação ao número de variáveis incluídas:

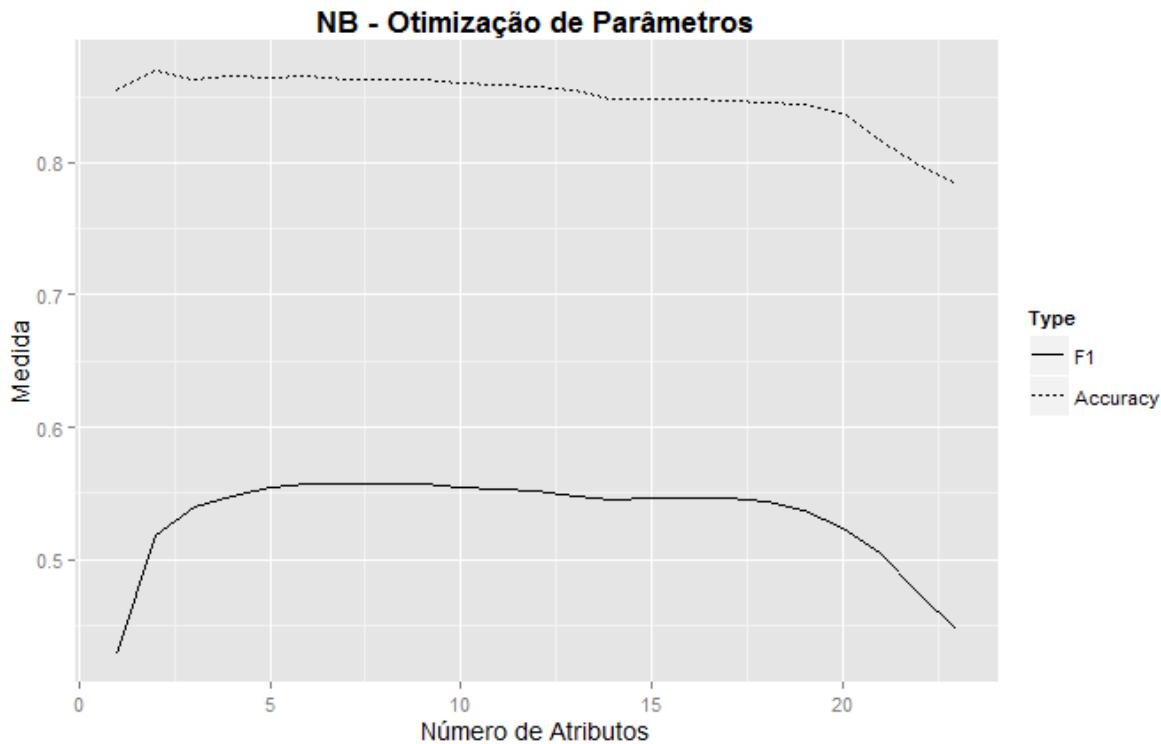


Figura 4.9: Seleção de Modelo - NB - Português

O modelo selecionado foi o com 7 variáveis:

$$Y = f(X_{12}, X_{16}, X_{11}, X_1^{YJ}, X_7, X_8, X_3) \quad (4.2.3)$$

Os resultados são:

Tabela 4.15: Tabela de Contingência - Naive Bayes - Português

	Classificado como Positivo	Classificado como Negativo
Positivo	765	292
Negativo	945	7040

Tabela 4.16: Tabela de Resultados - Naive Bayes - Português

Tipo	Acurácia	F1 (+)	Área ROC	Kappa
Validação (K-Fold)	0.8651096	0.5573801	-	-
Teste	0.863194	0.5529454	0.8893594	0.477443

Os resultados obtidos foram muito próximos aos do modelo KNN, apresentando um bom desempenho tanto no conjunto de treinamento como de testes.

4.2.8 Comparação

A Tabela 4.17 apresenta os resultados para os métodos utilizados.

As medidas são as apresentadas no Capítulo 3. Para as que são dependentes da classe, como a Precisão, temos que *C.Neg.* representa que a medida foi calculada com base na classe negativa (resposta 'Não' nessa aplicação). *Dp. Acurácia* representa o desvio padrão estimado da Acurácia, obtida através da variância da distribuição Binomial, e nos fornece uma medida de dispersão para a Acurácia do modelo.

Além disso, apresentamos os tempos de ajuste e predição para o modelo final selecionado, através das medidas *Tempo Ajuste* e *Tempo Predição*.

Tabela 4.17: Tabela de Resultados - Português

Medida	Aleatório	KNN	ANN	SVM	Árvore	Logística	NB
Acurácia	0.788	0.867	0.835	0.844	0.790	0.839	0.863
Dp. Acurácia	0.0043	0.0036	0.0039	0.0038	0.0043	0.0039	0.0036
Precisão	0.100	0.457	0.403	0.418	0.328	0.404	0.447
Recall	0.101	0.737	0.865	0.843	0.762	0.798	0.724
F1 Score	0.101	0.564	0.550	0.559	0.459	0.536	0.553
Taxa FP	0.121	0.116	0.169	0.155	0.206	0.156	0.118
Taxa TP	0.101	0.737	0.865	0.843	0.762	0.798	0.724
Kappa	-0.019	0.490	0.465	0.477	0.353	0.451	0.477
Precisão C.Neg	0.881	0.962	0.979	0.976	0.962	0.969	0.960
Recall C.Neg	0.879	0.884	0.831	0.846	0.794	0.844	0.882
F1 C.Neg	0.880	0.921	0.899	0.906	0.870	0.902	0.919
Área ROC	0.490	0.907	0.918	0.917	0.849	0.902	0.889
Tempo Ajuste	0 m	20.253 m	9.621 m	132.161 m	0.014 m	0.042 m	0.002 m
Tempo Predição	0 m	2.72 m	0.001 m	1.929 m	0.001 m	0.001 m	0.148 m

A Figura 4.10 mostra as Curvas ROC para os modelos testados.

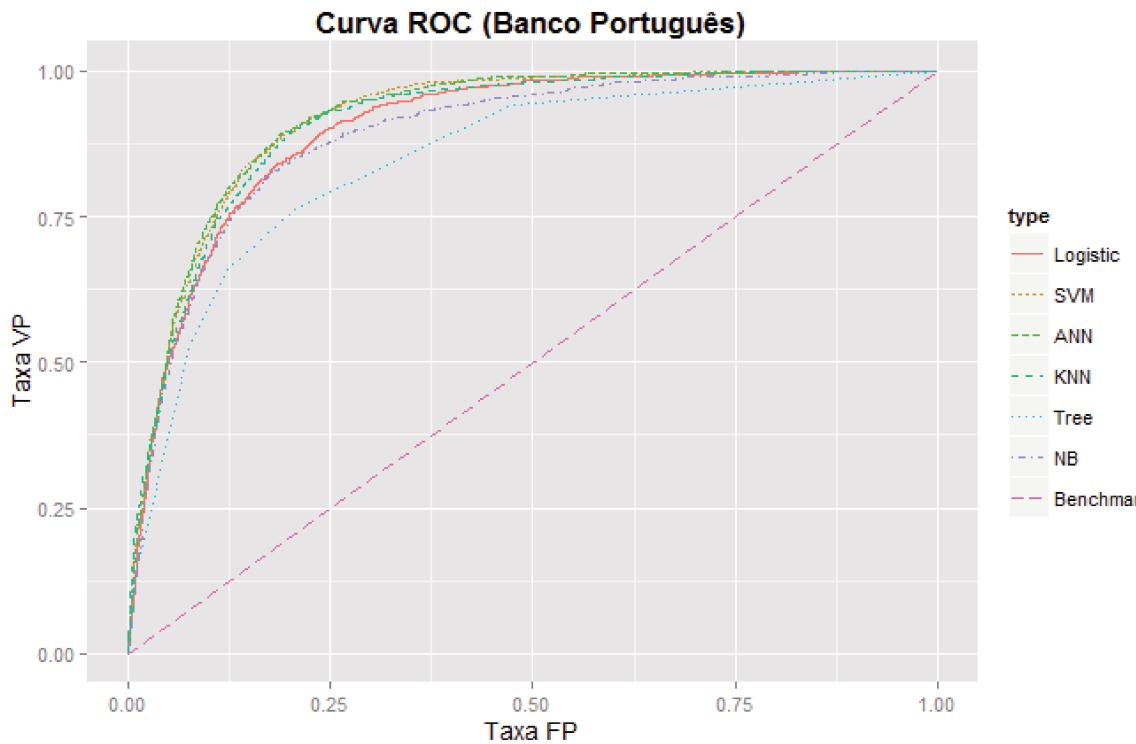


Figura 4.10: Curva Roc - Português

4.2.9 Conclusão

Não há uma resposta definitiva sobre qual modelo obteve o melhor desempenho, já que podemos observar que alguns modelos foram melhores em algumas medidas e piores em outras. No entanto, os modelos de árvore de decisão e regressão logística não obtiveram nenhum destaque positivo.

Do ponto de vista de acurácia e precisão, KNN e NB obtiveram os melhores resultados, no entanto na métrica *recall* os modelos ANN e SVM foram superiores, estando estes quatro próximos no valor da métrica F1, que pondera a precisão e o *recall*. Logo, vemos que embora essa métrica possa ser utilizada para encontrar modelos com bom desempenho geral, ainda há espaço para variações nas características dos mesmos. Utilizando o critério da curva ROC, os melhores são ANN e SVM.

Considerando a aplicação, que consiste em vender produtos para clientes, não selecionar um cliente comprador representa uma perda de venda, enquanto selecionar um cliente não comprador a mais resulta em um aumento de um contato no esforço de venda.

Supondo que o custo de perder o cliente é maior, o modelo escolhido neste caso seria a rede neural, já que tem o maior *recall*, e mesmo assim sem perder muito na precisão.

Como comparação, Moro et al. [18] obtiveram o melhor desempenho utilizando o SVM, com uma área abaixo da curva ROC de 0.938, ligeiramente maior do que o obtido aqui pelo ANN de 0.918. O método de divisão do conjunto de dados utilizado foi distinto, com 2/3 dos dados utilizados para teste.

4.3 Banco Alemão

Este conjunto de dados foi retirado do repositório da UCI Machine Learning [3]. A base contém 1000 observações de 21 atributos, com valores codificados e é referente a uma avaliação de crédito de clientes de um banco da Alemanha.

Tabela 4.18: Descrição do Banco de Dados - Banco Alemão

Num	Variável	Tipo	Descrição
1	X_1	Nominal	Estado da conta
2	X_2	Numérico	Duração em meses
3	X_3	Nominal	Histórico de crédito
4	X_4	Nominal	Finalidade do empréstimo
5	X_5	Numérico	Valor do empréstimo
6	X_6	Nominal	Poupança/Investimentos
7	X_7	Nominal	Tempo no Trabalho atual
8	X_8	Numérico	Valor percentual das parcelas em relação à renda
9	X_9	Nominal	Estado civil e sexo
10	X_{10}	Nominal	Outro aplicante/garantidor
11	X_{11}	Numérico	Tempo na residência atual
12	X_{12}	Nominal	Propriedades
13	X_{13}	Numérico	Idade
14	X_{14}	Nominal	Outros empréstimos
15	X_{15}	Nominal	Moradia
16	X_{16}	Numérico	Número de créditos neste banco
17	X_{17}	Nominal	Trabalho
18	X_{18}	Numérico	Quantidade de Dependentes
19	X_{19}	Nominal	Tem telefone
20	X_{20}	Nominal	Trabalhador estrangeiro
21	Y	Binário	Classificação do Cliente (Mau pagador=1)

4.3.1 Análise Exploratória

O atributo classe é binário e desbalanceado, contendo 30% de respostas positivas.

Para os atributos numéricos a maior correlação linear ocorre entre os atributos X_2 e X_5 , com valor de 0.62 e a segunda maior entre os atributos X_5 e X_8 , com valor de -0.27. Optamos por manter todas essas variáveis.

Para os atributos X_2 , X_5 e X_{13} foi aplicada uma transformação de Yeo-Johnson, para aproximar as observações de uma distribuição normal, estes atributos extras transformados serão utilizados apenas na regressão logística e Naive Bayes. Os dados foram divididos em dois conjuntos de forma aleatória: Treinamento (80%) e Teste (20%).

4.3.2 Regressão Logística

Para a regressão Logística o método de seleção utilizado foi da minimização do AIC utilizando *step-wise* do R aplicado no conjunto de treinamento balanceado. Após a seleção do modelo foi avaliado o VIF, para diagnóstico de multi-colinearidade, resultando na remoção do atributo X_5^{YJ} ($VIF = 7.72$).

O Modelo final é:

$$P[Y = 1] = f(X_1, X_3, X_4, X_5, X_6, X_7, X_8, X_9, X_{10}, X_{14}, X_{15}, X_{16}, X_{19}, X_{20}, X_2^{YJ}). \quad (4.3.1)$$

O teste Chi-Quadrado com H_0 sendo o modelo completo (com as variáveis transformadas inclusive), versus H_1 o modelo selecionado resultou em:

Tabela 4.19: Teste Chi-Quadrado - Alemão

Modelo	Graus de Liberdade	Deviance	Dif. G.l.	Dif. Deviance	P-valor
H_0	1080	1034.1			
H_1	1092	1059.3	-12	-25.22	0.01381

Podemos rejeitar H_0 portanto com nível de significância 5%, indicando que há alguma perda no Deviance ao se reduzir o número de variáveis, no entanto, o modelo com o menor AIC foi mantido.

Comparando o AIC e Deviance do modelo selecionado em relação ao modelo nulo temos:

Tabela 4.20: AIC e Deviance - Logística - Alemão

Deviance Modelo Nulo:	1569.3 com 1131 g.l.
Deviance:	1059.3 com 1092 g.l.
AIC:	1139.1

O que indica uma boa redução na Deviance ao utilizar o modelo de regressão logística.

A Figura 4.11 não aponta pontos influentes, logo consideramos que os resíduos indicam um bom ajuste do modelo.

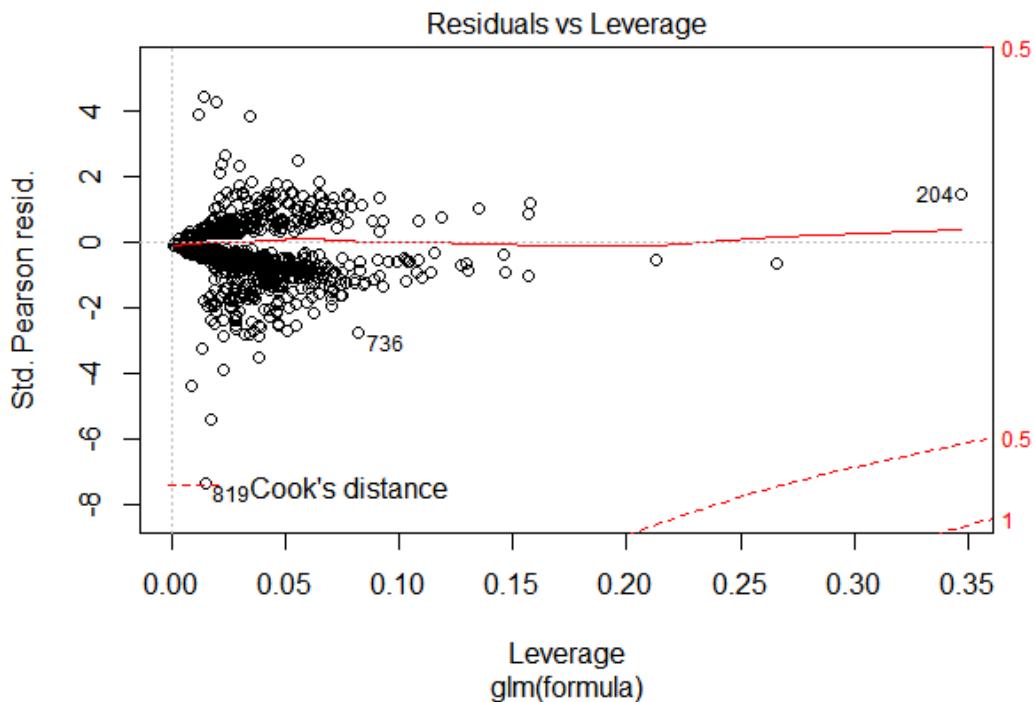


Figura 4.11: Resíduos vs Alavancagem - Alemão

Os resultados do modelo aplicado no conjunto de testes:

Tabela 4.21: Tabela de Contingência - Logística - Alemão

	Classificado como Positivo	Classificado como Negativo
Positivo	47	20
Negativo	37	96

Tabela 4.22: Tabela de Resultados - Logística - Alemão

Tipo	Acurácia	Área ROC	Kappa
Teste	0.715	0.7689373	0.3982264

A regressão logística apresentou um resultado razoável no conjunto de testes, sendo o terceiro melhor modelo em termos de acurácia, kappa e área ROC.

4.3.3 SVM

Foi realizada uma busca em grade preliminar para selecionar um intervalo de busca dos parâmetros e o tipo de *kernel*. A busca foi realizada usando o método 10-*Fold* no conjunto de treinamento.

As configurações testadas foram as seguintes:

- kernel = Radial

$$\gamma = \{2^{-8}, \dots, 2^0\};$$

$$custo = \{2^{-1}, \dots, 2^2\};$$

- kernel = Linear

$$custo = \{2^{-1}, \dots, 2^2\};$$

- kernel = Polynomial

$$\gamma = \{2^{-3}, \dots, 2^2\};$$

$$custo = \{2^0, 2^1\};$$

$$coef0 = \{0, 1\};$$

$$degree = \{1, 2\};$$

- kernel = Sigmoid

$$\gamma = \{2^{-3}, \dots, 2^1\};$$

$$custo = \{2^0, 2^1\};$$

$$coef0 = \{0, 1\}.$$

Com o melhor resultado do kernel tipo radial, foi feita uma segunda busca com os parâmetros:

- kernel = Radial

$$\gamma = \text{sequência no intervalo } [0, 0.25], \text{ com passos de tamanho } 0.02;$$

$$custo = \text{sequência no intervalo } [0.01, 4], \text{ com passos de tamanho } 0.25.$$

O resultado pode ser observado nos gráficos de contorno das Figuras 4.12 e 4.13.

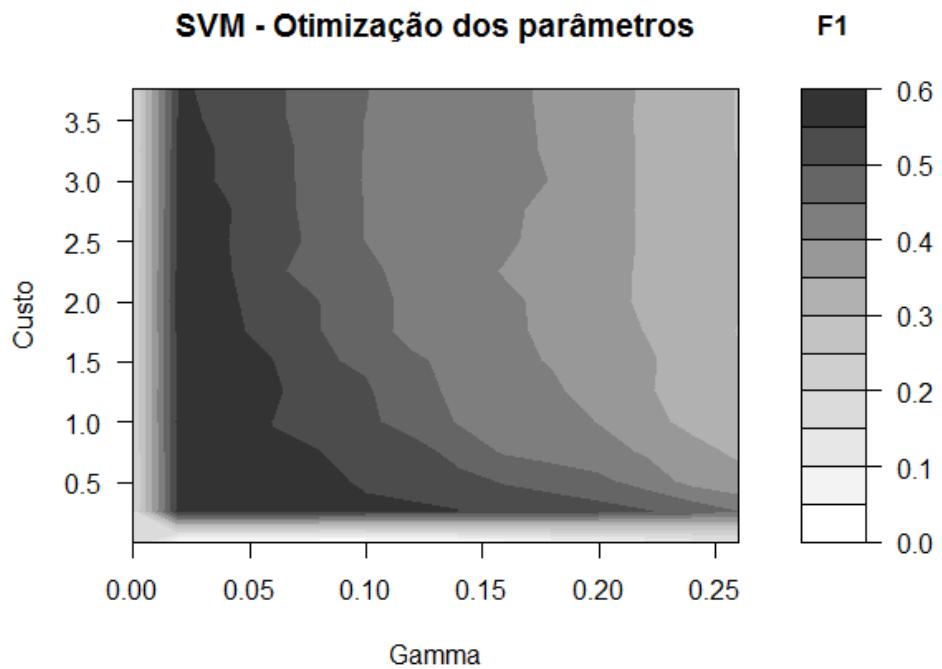


Figura 4.12: Seleção de Modelo - SVM F1 - Alemão

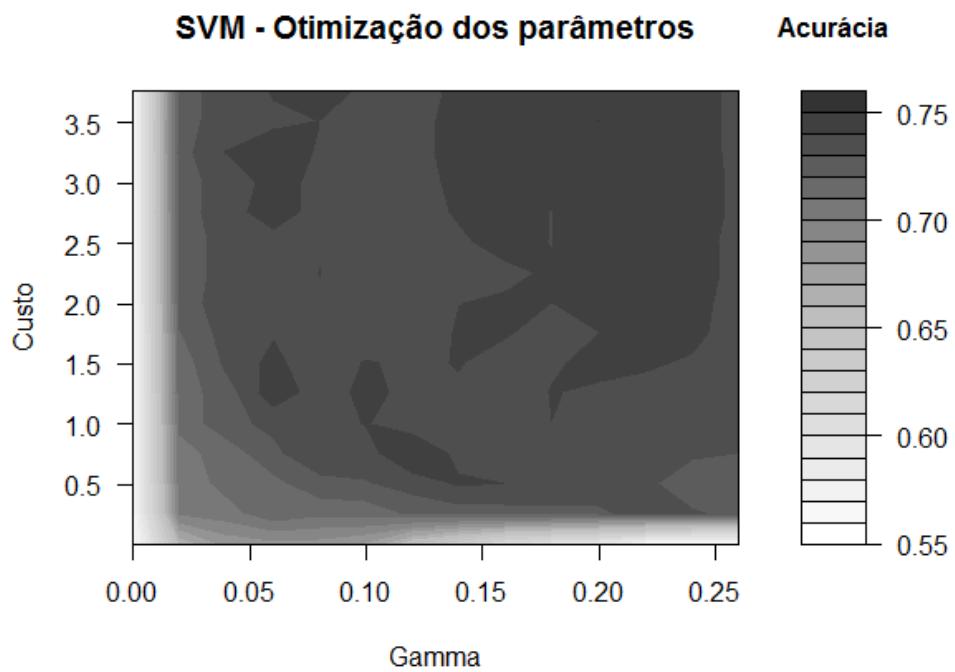


Figura 4.13: Seleção de Modelo - SVM - Alemão

Através da inspeção do gráfico e das tabelas de resultados (ver Tabelas A.5 e A.6 no Apêndice A), foi possível selecionar um ponto de equilíbrio entre acurácia e F1, utilizando $\gamma = 0.06$ e $custo = 1.26$

O Resultado para o melhor modelo está abaixo:

Tabela 4.23: Tabela de Contingência - SVM - Alemão

	Classificado como Positivo	Classificado como Negativo
Positivo	41	26
Negativo	24	109

Tabela 4.24: Tabela de Resultados - SVM - Alemão

Tipo	Acurácia	F1 (+)	Área ROC	Kappa
Treino (K-Fold)	0.7465147	0.5626183	-	-
Teste	0.75	0.6212121	0.7809449	0.4347089

Nota-se que o resultado do conjunto de testes é próximo do obtido no *K-Fold*, um indicativo que não houve sobre-ajuste ao selecionar os parâmetros.

4.3.4 ANN

Para a Rede Neural, a busca foi com os parâmetros realizada na seguinte configuração:

- $tamanho$ = sequência no intervalo [2, 14], com passos de tamanho 2;
- $decay$ = sequência no intervalo [0.1, 1], com passos de tamanho 0.1.

Os resultados estão nos gráficos de contorno das Figuras 4.14 e 4.15.

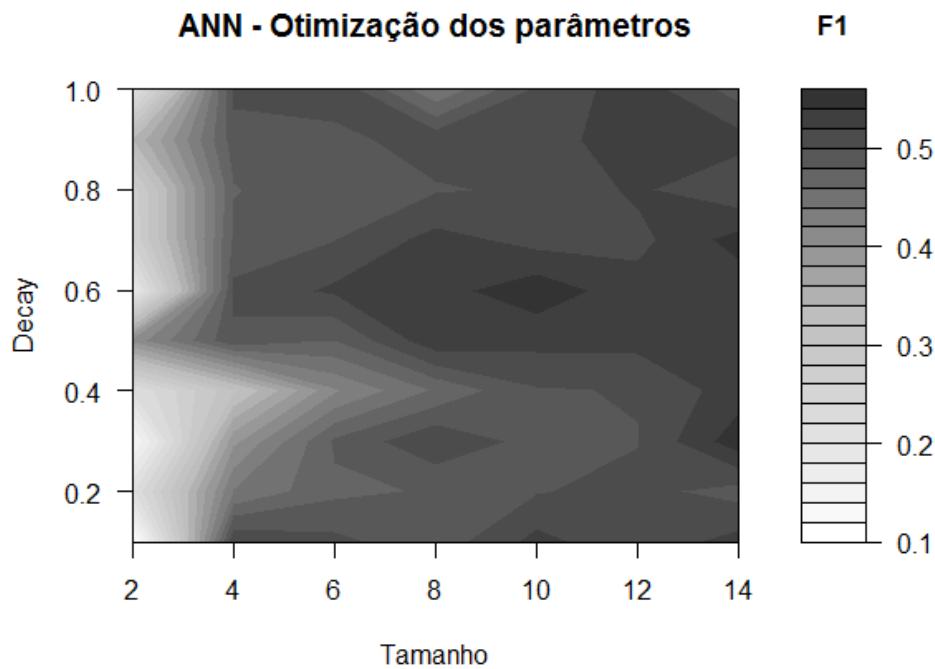


Figura 4.14: Seleção de Modelo - ANN F1 - Alemão

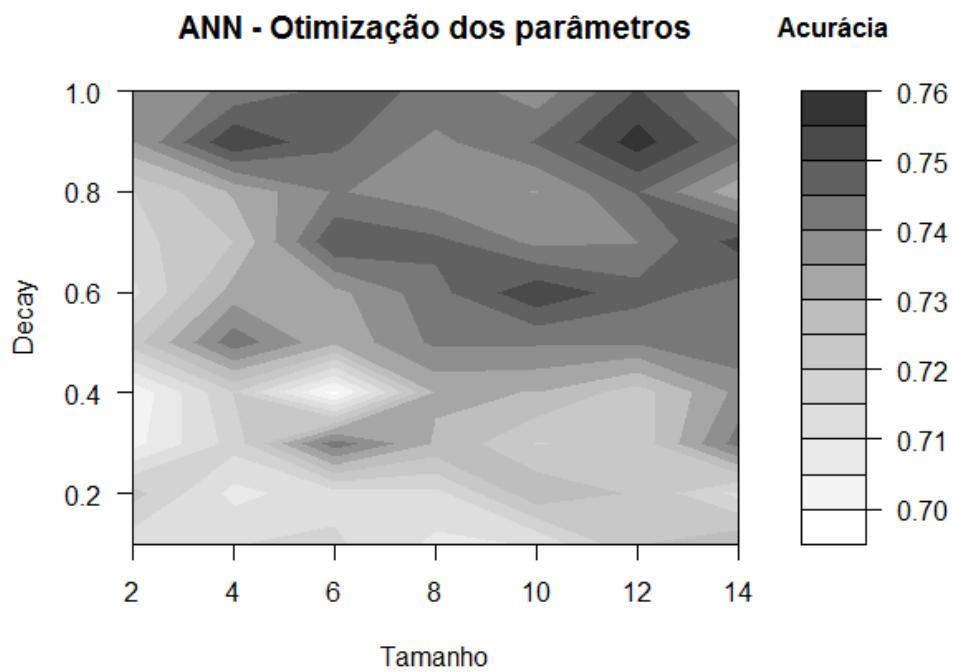


Figura 4.15: Seleção de Modelo - ANN - Alemão

Através da análise dos gráficos e tabelas de resultados (ver Tabelas A.7 e A.8 no Apêndice A), e buscando um equilíbrio entre acurácia e F1, os parâmetros selecionados foram $tamanho = 10$ e $decay = 0.6$.

A estrutura selecionada da rede foi portanto de 49 nós de entrada, dez nós na camada escondida e 1 nó de saída, que junto com os termos constantes das camadas de entrada e interna resultam em 511 pesos para serem estimados.

Os resultados do modelo final no conjunto de teste estão nas Tabelas 4.25 e 4.26:

Tabela 4.25: Tabela de Contingência - ANN - Alemão

	Classificado como Positivo	Classificado como Negativo
Positivo	40	27
Negativo	38	95

Tabela 4.26: Tabela de Resultados - ANN - Alemão

Tipo	Acurácia	F1 (+)	Área ROC	Kappa
Treino (K-Fold)	0.7542457	0.5518137	-	-
Teste	0.675	0.5517241	0.7338121	0.2991158

Os resultados de acurácia no conjunto de testes são inferiores ao do conjunto de treino indicando sobre-ajuste. Além disso os resultado foram bem inferiores aos obtidos com o SVM.

4.3.5 KNN

Para o modelo KNN, foi feita uma busca em grade no parâmetro k (número de vizinhos) e peso (pesos iguais versus 1/distância), combinando com *step-forward* para seleção de atributos, adicionando para cada k as variáveis em ordem de desempenho na medida F1.

A melhor combinação de atributos selecionada foi:

$$Y = f(X_1, X_{12}, X_{20}, X_6, X_{10}, X_{18}) \quad (4.3.2)$$

Com peso = 1/distância e $k = 8$. A escolha de k pode ser observada na Figura 4.16:

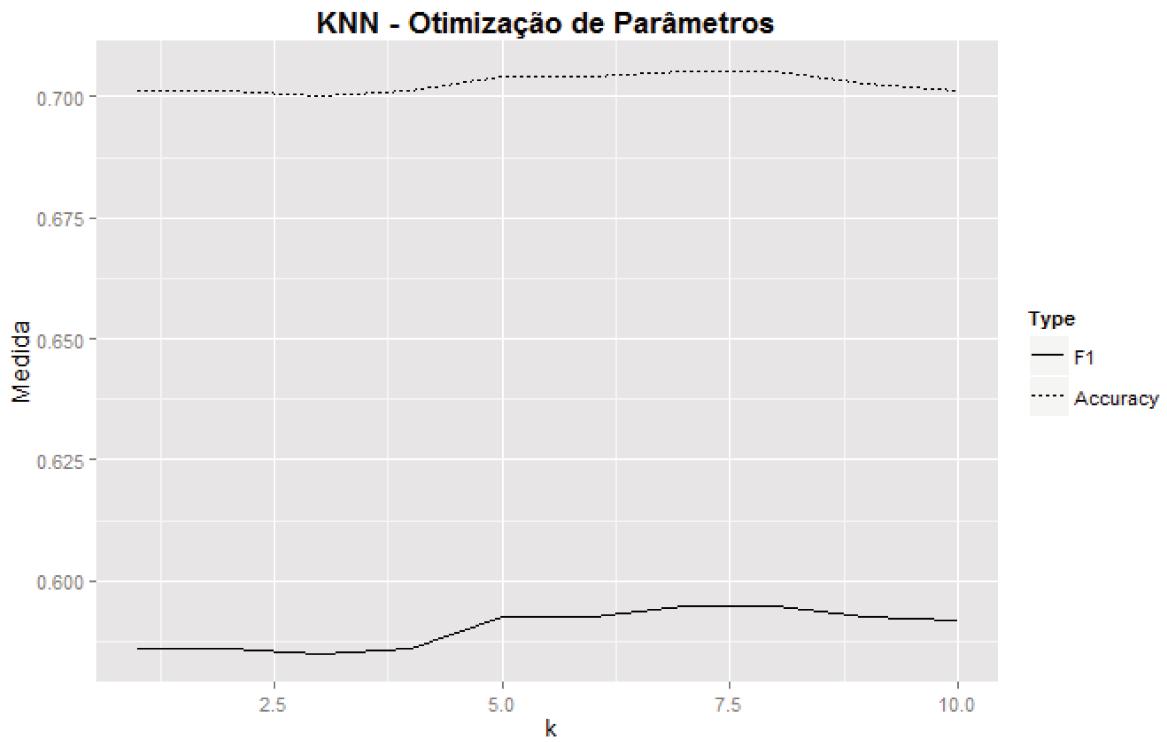


Figura 4.16: Seleção de Modelo - KNN - Alemão

Os resultados do modelo selecionado são:

Tabela 4.27: Tabela de Contingência - KNN - Alemão

	Classificado como Positivo	Classificado como Negativo
Positivo	44	23
Negativo	49	84

Tabela 4.28: Tabela de Resultados - KNN - Alemão

Tipo	Acurácia	F1 (+)	Área ROC	Kappa
Treino (K-Fold)	0.7053954	0.5949279	-	-
Teste	0.64	0.55	0.6515543	0.2629747

O modelo KNN não proporcionou um bom ajuste e o desempenho no conjunto de testes não foi bom.

4.3.6 Árvore de Decisão

Para o modelo de árvore de decisão, o número de folhas foi otimizado em relação às estatísticas Acurácia e F1, utilizando o método de $K - Fold$ ($k = 10$).

A Figura 4.17 mostra o resultado:

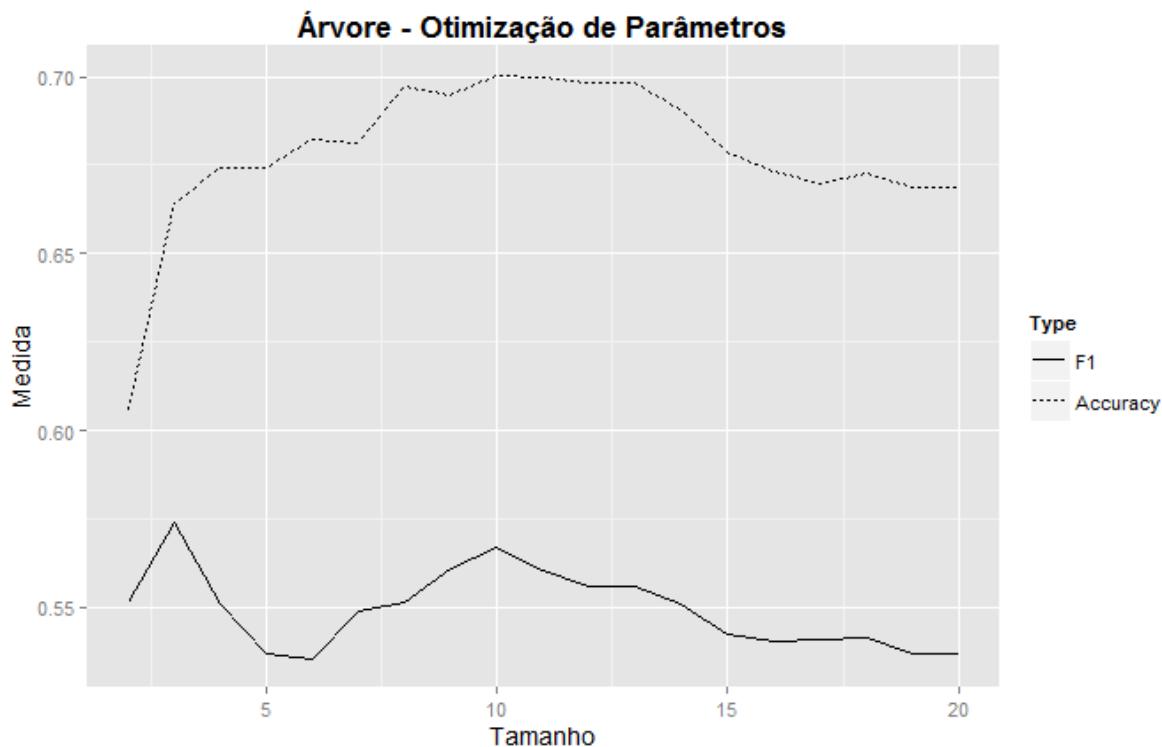


Figura 4.17: Seleção de Modelo - Árvore - Alemão

O modelo selecionado foi com 10 folhas, e os resultados estão na Tabela 4.29:

Tabela 4.29: Tabela de Contingência - Árvore de Decisão - Alemão

	Classificado como Positivo	Classificado como Negativo
Positivo	40	27
Negativo	42	91

Tabela 4.30: Tabela de Resultados - Árvore de Decisão - Alemão

Tipo	Acurácia	F1 (+)	Área ROC	Kappa
Treino (K-Fold)	0.7004195	0.5671808	-	-
Teste	0.655	0.5369128	0.6820783	0.2664257

Os resultados da árvore foram similares aos obtidos com o método KNN.

4.3.7 Naive Bayes

Para o método Naive Bayes, foi realizada uma seleção de variáveis utilizando o método *K – Fold* ($k = 10$), e selecionando de forma *step-forward* as variáveis com maior valor da estatística F1.

A Figura 4.18 mostra o resultado em relação ao número de variáveis incluídas:

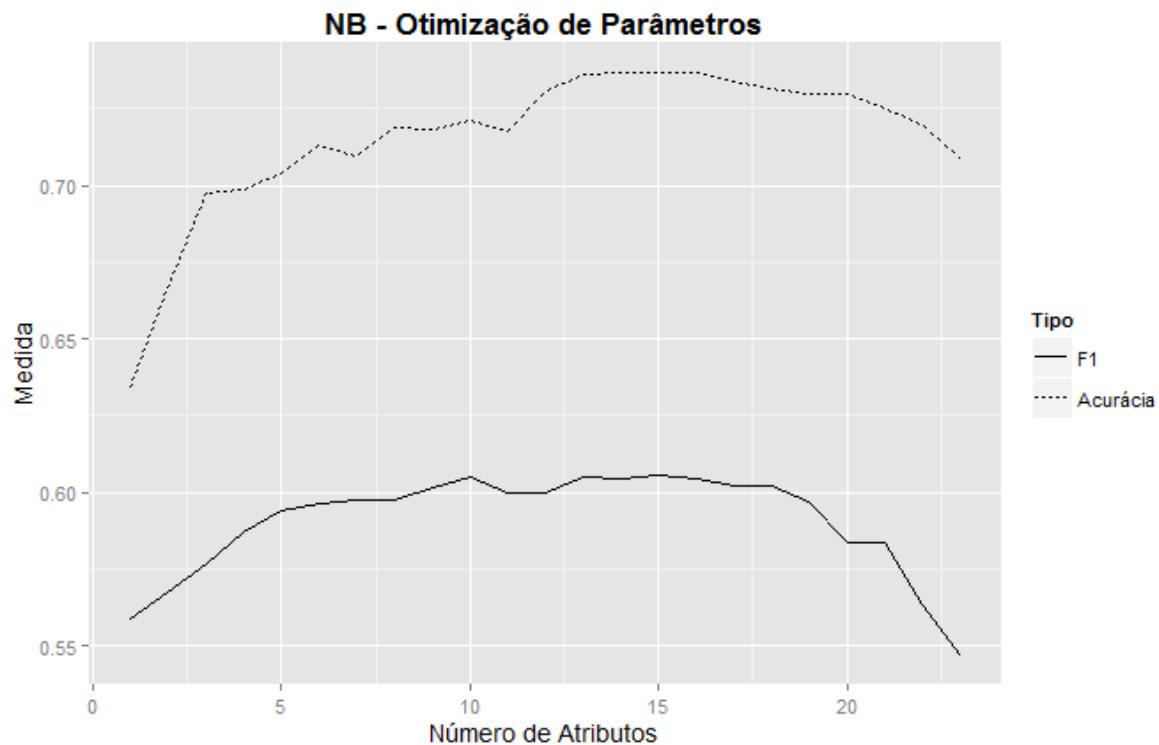


Figura 4.18: Seleção de Modelo - NB - Alemão

O modelo selecionado foi o com 15 variáveis:

$$Y = f(X_1, X_2^{YJ}, X_{12}, X_6, X_{19}, X_9, X_{13}^{YJ}, X_3, X_8, X_{13}, X_{10}, X_5, X_7, X_{11}, X_{20}) \quad (4.3.3)$$

Os resultados são:

Tabela 4.31: Tabela de Contingência - NB - Alemão

	Classificado como Positivo	Classificado como Negativo
Positivo	47	20
Negativo	35	98

Tabela 4.32: Tabela de Resultados - NB - Alemão

Tipo	Acurácia	F1 (+)	Área ROC	Kappa
Treino (K-Fold)	0.7368013	0.6057541	-	-
Teste	0.725	0.630	0.7776905	0.4152669

Os resultados foram muito bons para o modelo Naive Bayes, comparáveis aos do SVM.

4.3.8 Comparação

A Tabela 4.33 summariza os principais resultados, e a Figura 4.19 apresenta a curva ROC para os modelos utilizados.

Tabela 4.33: Comparativo de Resultados - Banco Alemão

Medida	Aleatório	KNN	ANN	SVM	Árvore	Logística	NB
Acurácia	0.570	0.640	0.675	0.750	0.655	0.715	0.725
Dp. Acurácia	0.035	0.034	0.033	0.031	0.034	0.032	0.032
Precisão	0.344	0.473	0.513	0.631	0.488	0.560	0.573
Recall	0.313	0.657	0.597	0.612	0.597	0.701	0.701
F1 Score	0.328	0.550	0.552	0.621	0.537	0.623	0.631
Taxa FP	0.301	0.368	0.286	0.180	0.316	0.278	0.263
Taxa TP	0.313	0.657	0.597	0.612	0.597	0.701	0.701
Kappa	0.013	0.263	0.299	0.435	0.266	0.398	0.415
Precisão C.Neg	0.669	0.785	0.779	0.807	0.771	0.828	0.831
Recall C.Neg	0.699	0.632	0.714	0.820	0.684	0.722	0.737
F1 C.Neg	0.684	0.700	0.745	0.813	0.725	0.771	0.781
Área ROC	0.506	0.652	0.734	0.781	0.682	0.769	0.778
Tempo Ajuste	0 s	0.422 s	7.533 s	1.754 s	0.031 s	0.031 s	0.012 s
Tempo Predição	0 s	0.085 s	0.003 s	0.047 s	0 s	0.008 s	0.263 s

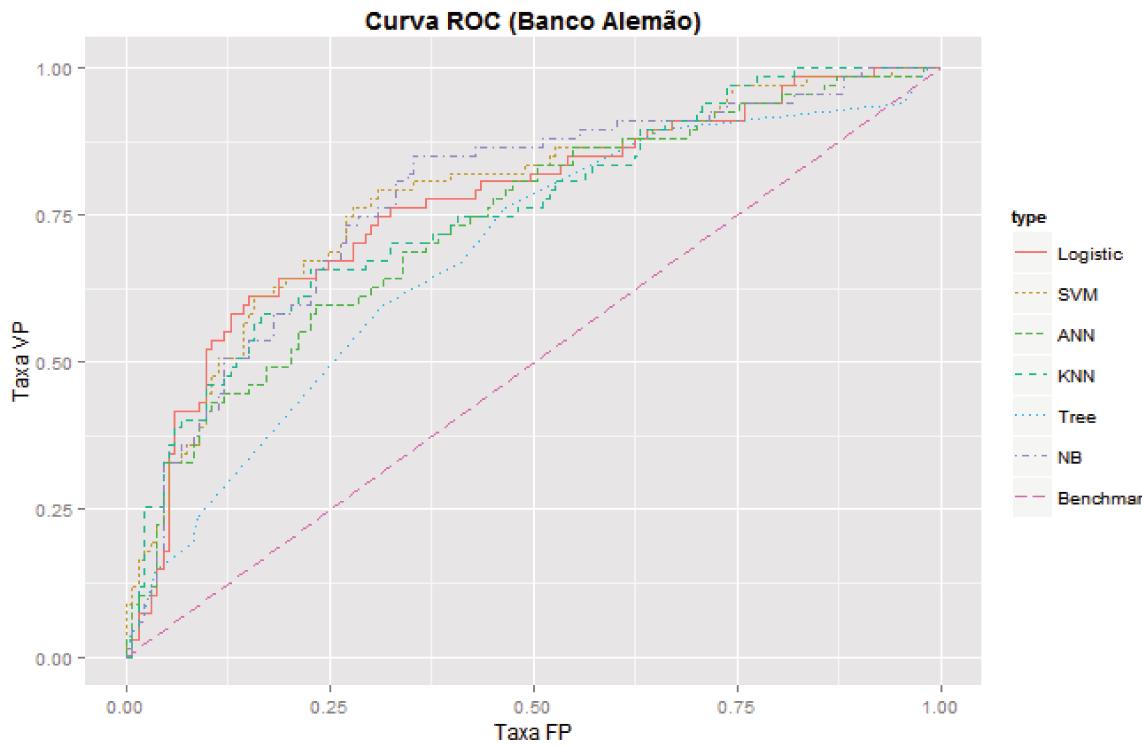


Figura 4.19: Curva Roc - Alemão

4.3.9 Conclusão

Para esta aplicação, não foi possível selecionar um modelo específico como o melhor. No entanto, três obtiveram destaque: NB, Regressão Logística e SVM, sendo que este último apresentou maior acurácia, mas perdeu na taxa de verdadeiro positivo. Nos critérios que consideram o equilíbrio do modelo em relação às classes, o melhor modelo considerando F1 foi Naive Bayes e considerando a área ROC, foi o SVM.

Usualmente em aplicações de crédito, o custo é maior ao classificarmos um cliente ruim como bom, já que isto representa um empréstimo onde o pagamento não será realizado. Neste conjunto de dados, a classe denominada como positivo (por ser a classe minoritária) representa os clientes considerados ruins, logo, devemos analisar com cautela a métrica *recall* (Taxa de Verdadeiro Positivo), que nos indica o percentual de clientes ruins identificados pelo modelo.

Apesar da Regressão Logística e Naive Bayes terem obtido um *recall* maior que o do SVM, através da Curva ROC percebe-se que se flexibilizarmos o critério de classificação para o SVM, permitindo um valor de Falso Positivo similar ao dos outros dois métodos, obteríamos um valor similar de *recall* para o SVM, o que é esperado, considerando que o mesmo possui a maior área

ROC.

Como comparação, Eggermont et al. [6] obtiveram o melhor desempenho utilizando um método de Programação Genética baseada em árvore, utilizando 10-fold, e obtiveram um resultado médio de 72.9% de acurácia, com o melhor fold obtendo 75.7% de acurácia e o pior 71.5%. Neste trabalho obtivemos uma acurácia de 75% no conjunto de testes para o modelo SVM.

4.4 Aprovação de crédito Australiano

Este conjunto de dados foi retirado do repositório da UCI Machine Learning [3]. A base contém 690 observações de 16 atributos, com nomes dos atributos e valores codificados. É referente a uma avaliação de crédito para clientes de uma empresa Australiana.

Tabela 4.34: Descrição do Banco de Dados - Aprovação de Crédito Australiano

Num	Variável	Tipo	Dados Faltantes
1	X_1	Nominal	Sim (1.7%)
2	X_2	Numérico	Sim (1.7%)
3	X_3	Numérico	Não
4	X_4	Nominal	Sim (1%)
5	X_5	Nominal	Sim (1%)
6	X_6	Nominal	Sim (1.3%)
7	X_7	Nominal	Sim (1.3%)
8	X_8	Numérico	Não
9	X_9	Nominal	Não
10	X_{10}	Nominal	Não
11	X_{11}	Numérico	Não
12	X_{12}	Nominal	Não
13	X_{13}	Nominal	Não
14	X_{14}	Numérico	Sim (1.9%)
15	X_{15}	Numérico	Não
16	Y	Binário	Não

4.4.1 Análise Exploratória

O atributo classe é binário e contém 44% de respostas positivas. Mesmo com a pequena diferença, optamos pelo balanceamento de classes como nos demais problemas.

Para os atributos numéricos foi aplicada uma transformação de Yeo-Johnson, para aproximar as observações de uma distribuição normal. Assim como nas aplicações anteriores, estes atributos transformados serão utilizados apenas na regressão logística e Naive Bayes e indicados por YJ .

A maior correlação linear ocorre entre os atributos X_2 e X_8 , com valor de 0.395. Optamos por manter todas as variáveis.

Os dados foram divididos em dois conjuntos de forma aleatória: Treinamento (80%) e Teste (20%).

4.4.2 Regressão Logística

O Modelo foi selecionado pelo método *stepwise*, e os atributos X_2 (VIF = 13.97) e X_3 (VIF = 7.66) foram posteriormente removidos devido ao valor do VIF, resultando no modelo final:

$$Y = f(X_4, X_7, X_9, X_{10}, X_{11}, X_{13}, X_{15}, X_2^{YJ}, X_3^{YJ}, X_8^{YJ}, X_{14}^{YJ}) \quad (4.4.1)$$

O teste Chi-Quadrado com H_0 sendo o modelo completo (com as variáveis transformadas inclusive), versus H_1 o modelo selecionado resultou em:

Tabela 4.35: Teste Chi-Quadrado - Australiano

Modelo	Graus de Liberdade	Deviance	Dif. G.l.	Dif. Deviance	P-valor
H_0	573	352.58			
H_1	594	382.31	-21	-29.724	0.09767

O alto p-valor sugere poucas evidências contra H_0 , e optamos portanto, por ficar com o modelo de H_1 pois além disso contém menos variáveis e tem o menor AIC (dado que o método de seleção minimiza o AIC).

AIC e Deviance do modelo:

Tabela 4.36: AIC e Deviance - Logística - Australiano

Deviance Modelo Nulo:	852.57 com 614 g.l.
Deviance:	382.31 com 594 g.l.
AIC:	424.31

A Figura 4.20 indica que a observação 322 é muito influente, ela foi portanto removida da análise (apenas para a regressão logística) e o novo modelo não mostrou observações influentes, como pode ser observado na Figura 4.21

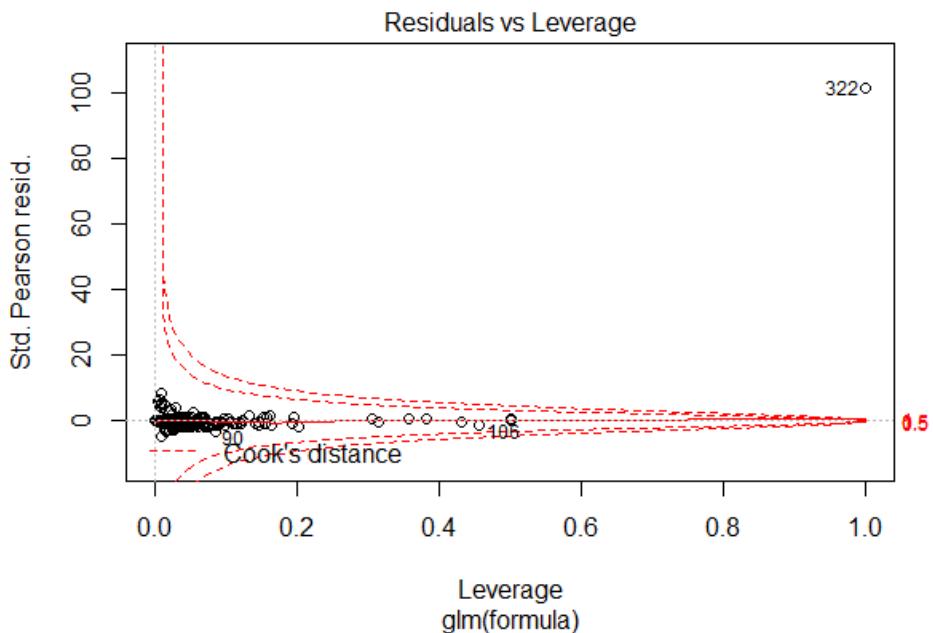


Figura 4.20: Resíduos vs Alavancagem (todas as obs.) - Australiano

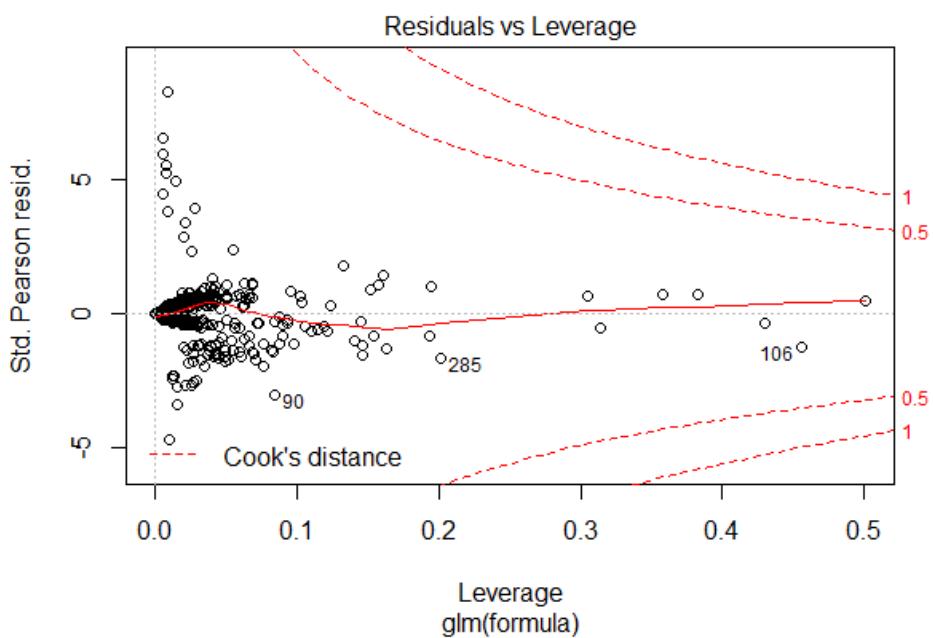


Figura 4.21: Resíduos vs Alavancagem - Australiano

Os resultados do modelo aplicado no conjunto de testes:

Tabela 4.37: Tabela de Contingência - Logística - Australiano

	Classificado como Positivo	Classificado como Negativo
Positivo	57	6
Negativo	14	61

Tabela 4.38: Tabela de Resultados - Logística - Australiano

	Tipo	Acurácia	Área ROC	Kappa
Teste	0.8550725	0.9303704	0.7108737	

Já neste primeiro método nota-se a grande diferença nas medidas de ajuste em comparação ao problema anterior (Banco Alemão), indicando que alguns problemas são mais fáceis de modelar e construir um modelo preditivo.

4.4.3 SVM

Para o SVM, foram testadas combinações de parâmetro semelhantes aos da aplicação anterior, e os melhores resultados foram obtidos com o kernel do tipo Radial. Uma segunda busca foi feita, em um sub-set reduzido, mas mais detalhado dos parâmetros:

- kernel = Radial;
- $gamma$ = sequência no intervalo $[0.005, 0.2]$ com passos de tamanho 0.005;
- $custo$ = sequência no intervalo $[0.03, 3]$ com passos de tamanho 0.1.

O resultado pode ser observado nas Figuras 4.22 e 4.23.

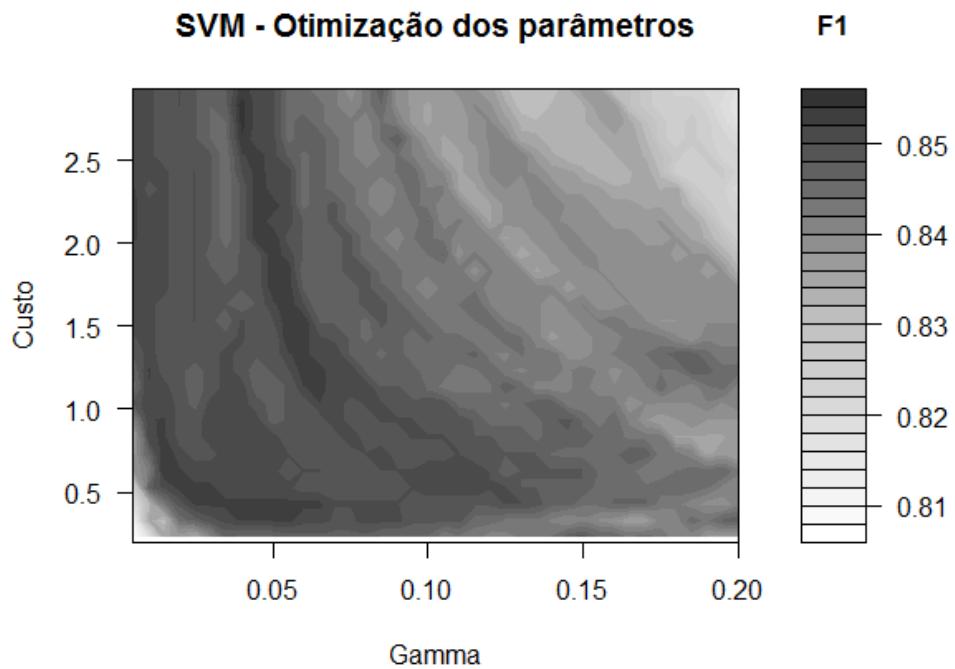


Figura 4.22: Seleção de Modelo - SVM F1 - Australiano

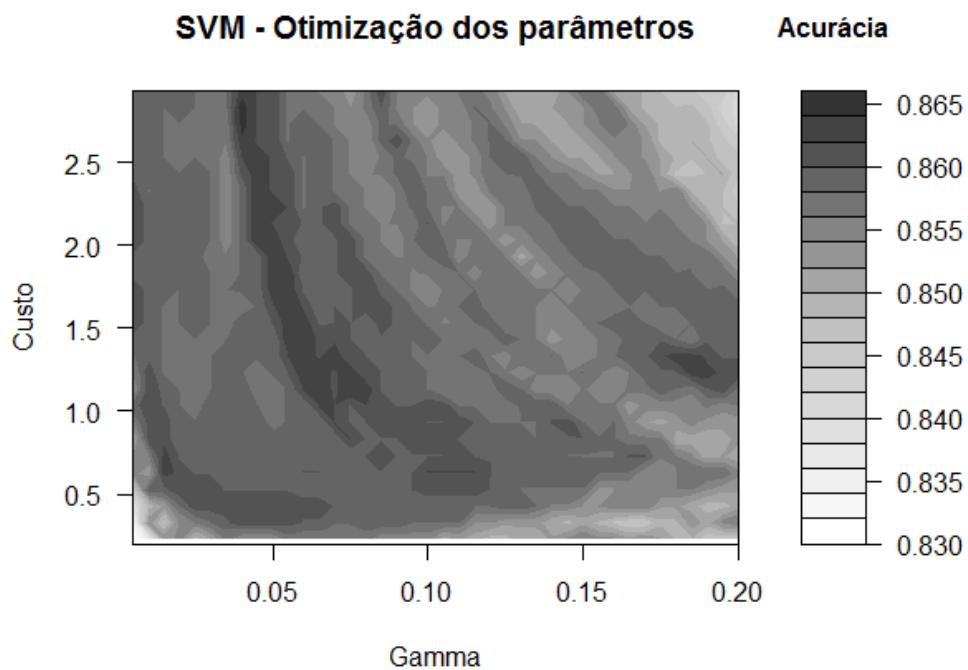


Figura 4.23: Seleção de Modelo - SVM - Australiano

Através da inspeção do gráfico e das tabelas de resultados (ver Tabelas A.9 e A.10 no Apêndice A), foi possível selecionar um ponto de equilíbrio entre acurácia e F1, utilizando $\gamma = 0.04$ e $C = 2.73$.

O Resultado para o melhor modelo está nas Tabelas 4.39 e 4.40.

Tabela 4.39: Tabela de Contingência - SVM - Australiano

	Classificado como Positivo	Classificado como Negativo
Positivo	58	5
Negativo	16	59

Tabela 4.40: Tabela de Resultados - SVM - Australiano

Tipo	Acurácia	F1 (+)	Área ROC	Kappa
Treino (K-Fold)	0.8652859	0.8550979	-	-
Teste	0.8478261	0.8467153	0.9140741	0.6975579

Nota-se que o resultado do conjunto de testes é próximo do obtido no *K-Fold*, um indicativo que não houve sobre-ajuste ao selecionar os parâmetros.

4.4.4 ANN

Para a Rede Neural, a busca foi com os parâmetros realizada na seguinte configuração:

- $tamanho$ = sequência no intervalo $[2, 14]$ com passos de tamanho 2;
- $decay$ = sequência no intervalo $[0.1, 3]$ com passos de tamanho 0.2.

As figuras 4.24 e 4.25 mostram os resultados da busca em grade.

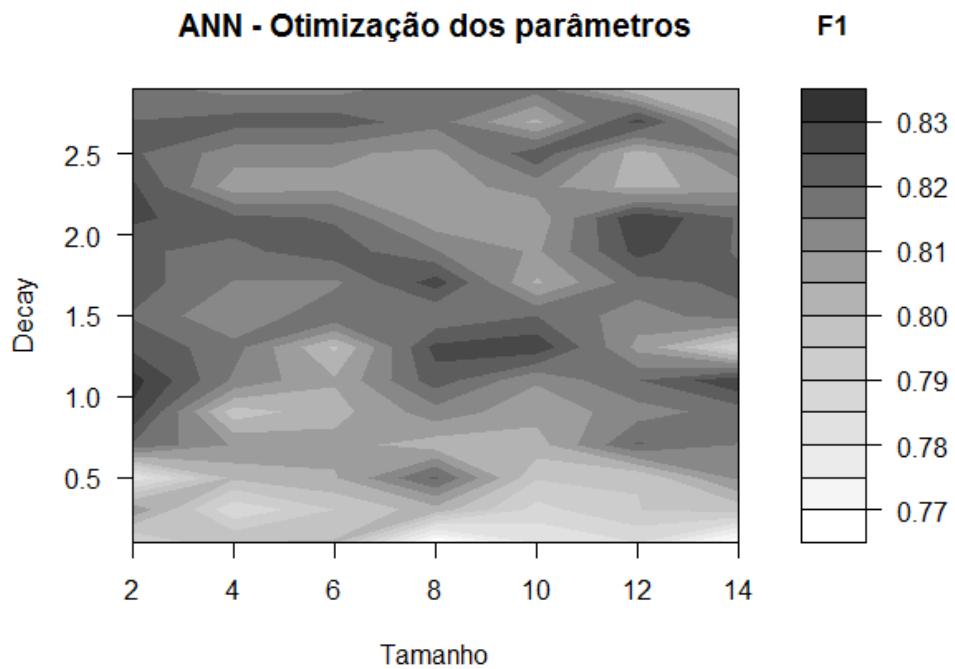


Figura 4.24: Seleção de Modelo - ANN F1 - Australiano

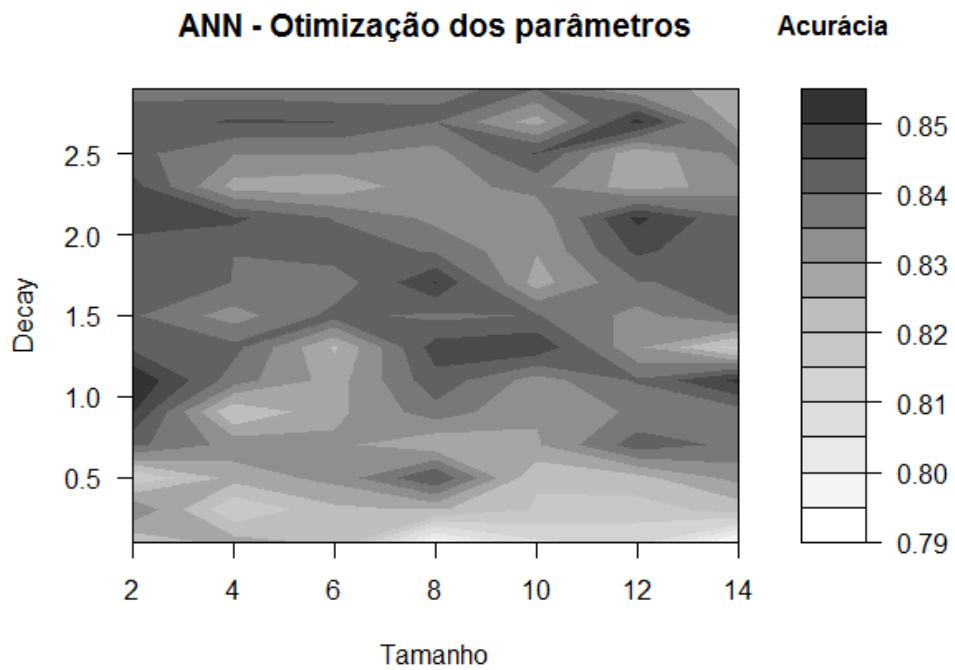


Figura 4.25: Seleção de Modelo - ANN - Australiano

Através da análise dos gráficos e tabelas de resultados (ver Tabelas A.11 e A.12 no Apêndice A), e buscando um equilíbrio entre acurácia e F1, os parâmetros selecionados foram $tamanho = 2$ e $decay = 1.1$.

A estrutura selecionada da rede foi portanto de 38 nós de entrada, 2 nós na camada escondida e 1 nó de saída, que junto com os termos constantes das camadas de entrada e interna resultam em 81 pesos para serem estimados.

Os resultados do modelo final no conjunto de teste estão nas Tabelas 4.41 e 4.42:

Tabela 4.41: Tabela de Contingência - ANN - Australiano

	Classificado como Positivo	Classificado como Negativo
Positivo	56	7
Negativo	9	66

Tabela 4.42: Tabela de Resultados - ANN - Australiano

Tipo	Acurácia	F1 (+)	Área ROC	Kappa
Treino (K-Fold)	0.8543336	0.8320544	-	-
Teste	0.884058	0.875	0.9470899	0.7669411

A Rede Neural foi o método com o melhor desempenho no conjunto de testes, considerando as quatro métricas apresentadas. Ao contrário das aplicações anteriores, não há indicação de sobre-ajuste, já que o desempenho no conjunto de testes foi inclusive superior ao do obtido no conjunto de treinamento.

4.4.5 KNN

Para o KNN foi feita a busca com k no conjunto $\{1, 5, 10, 20, 30, 50\}$, e buscando a melhor combinação de atributos.

A melhor combinação de atributos selecionada foi:

$$Y = f(X_9, X_1, X_{13}, X_{14}, X_8, X_{15}) \quad (4.4.2)$$

Com peso = $1/distância$ e $k = 20$.

Em seguida a escolha de k foi refinada como pode ser observado na Figura 4.26, e o k selecionado foi 21.

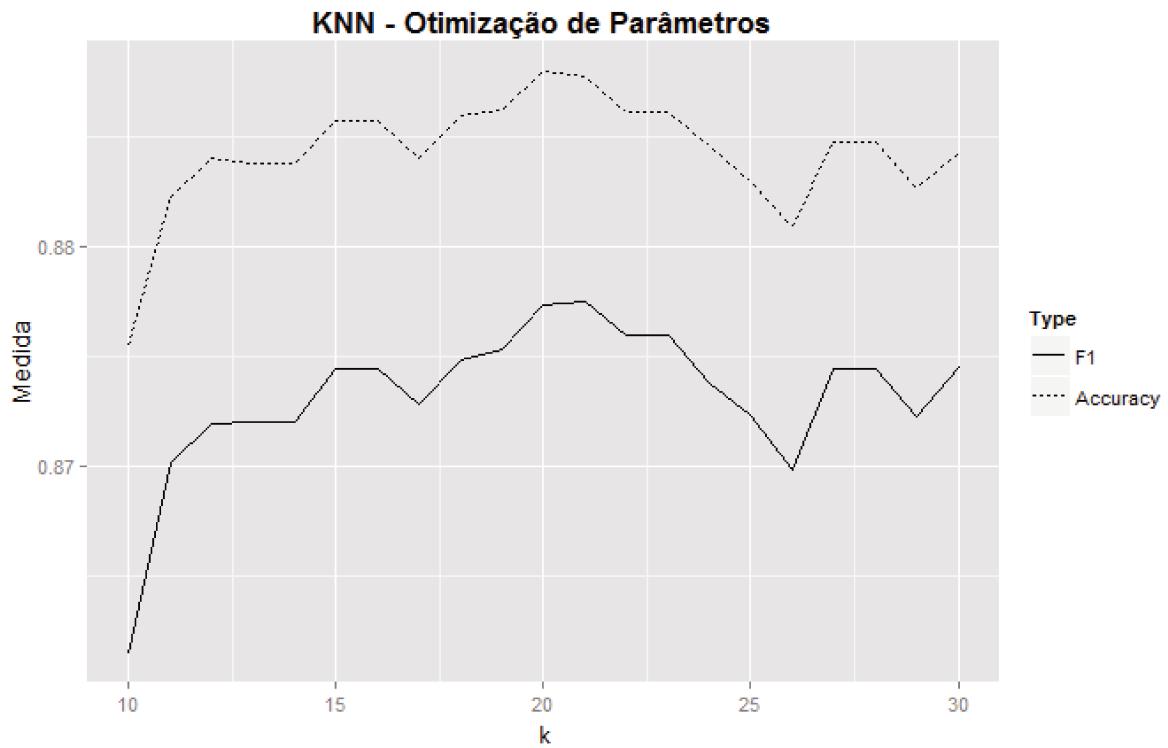


Figura 4.26: Seleção de Parâmetros - KNN - Australiano

Os resultados do modelo selecionado são:

Tabela 4.43: Tabela de Contingência - KNN - Australiano

	Classificado como Positivo	Classificado como Negativo
Positivo	55	8
Negativo	15	60

Tabela 4.44: Tabela de Resultados - KNN - Australiano

	Tipo	Acurácia	F1 (+)	Área ROC	Kappa
Treino (K-Fold)		0.8877704	0.8774637	-	-
Teste		0.8333333	0.8270677	0.8956614	0.6670862

O KNN obteve o pior resultado para ambas as métricas Acurácia, F1 e Kappa, embora a diferença não tenha sido tão grande em relação aos outros métodos.

4.4.6 Árvore de Decisão

Para a árvore de decisão o intervalo inicial de tamanho de árvore foi de 2 a 30, mas após constatação que as árvores com menos folha apresentaram melhor desempenho, foi realizado um refinamento de tamanho=2, 3, ..., 10.

A Figura 4.27 mostra o resultado por tamanho de árvore:

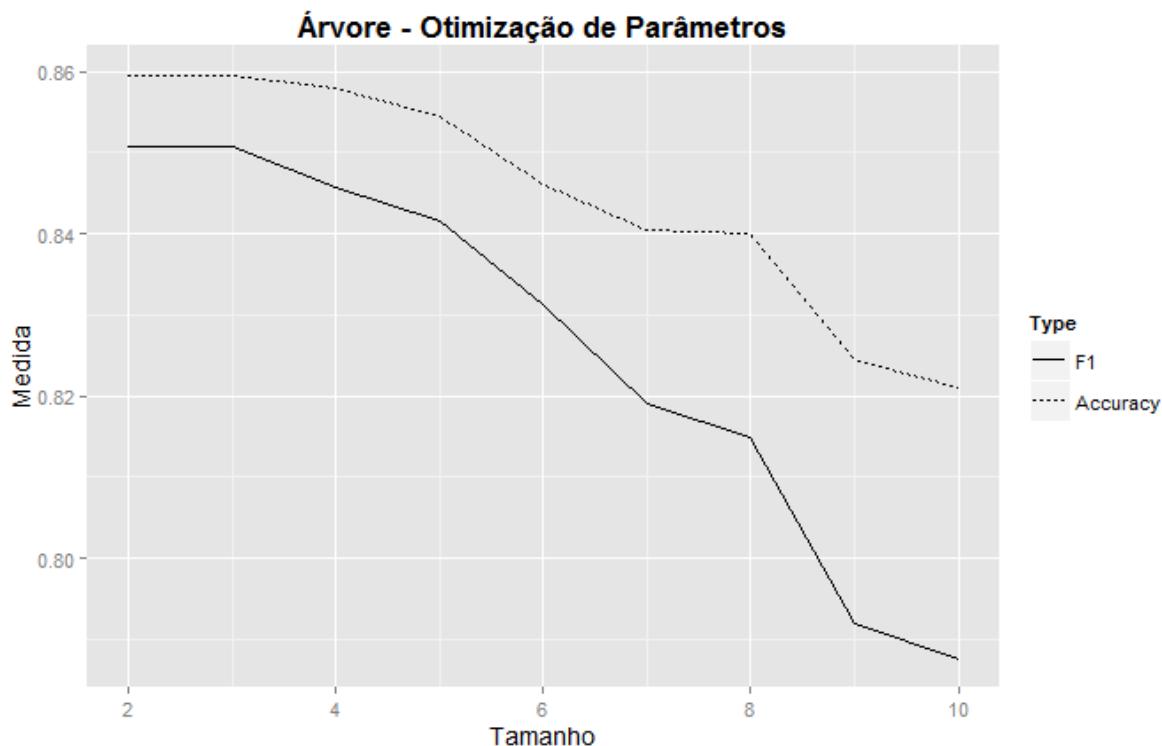


Figura 4.27: Seleção de Modelo - Árvore - Australiano

O modelo selecionado foi com apenas 3 folhas, e os resultados estão na Tabela 4.45:

Tabela 4.45: Tabela de Contingência - Árvore de Decisão - Australiano

Classificado como Positivo Classificado como Negativo

Positivo	58	5
Negativo	18	57

Tabela 4.46: Tabela de Resultados - Árvore de Decisão - Australiano

Tipo	Acurácia	F1 (+)	Área ROC	Kappa
Treino (K-Fold)	0.8595645	0.8507845	-	-
Teste	0.8333333	0.8345324	0.8913228	0.6695815

A Árvore de decisão teve desempenho similar ao KNN, empatando no critério Acurácia e sendo ligeiramente superior no critério F1.

4.4.7 Naive Bayes

Para o método Naive Bayes, foi utilizada a função `naiveBayes` do pacote `e1071` [15] do R.

Foi realizada uma seleção de variáveis utilizando o método *K-Fold* ($k=10$), e selecionando de forma *step-forward* as variáveis com maior valor da estatística F1.

A Figura 4.28 mostra o resultado em relação ao número de variáveis incluídas:

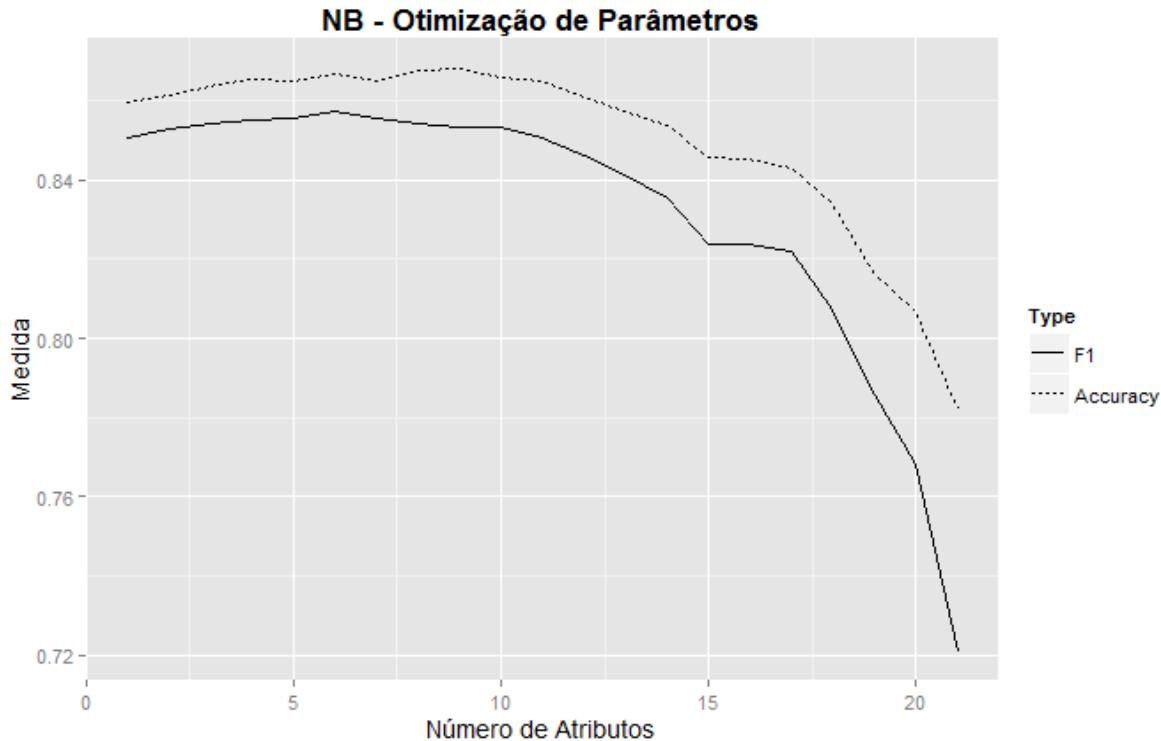


Figura 4.28: Seleção de Modelo - NB - Australiano

O modelo selecionado foi o com 6 variáveis:

$$Y = f(X_9, X_{14}, X_{10}, X_7, X_2, X_{12}) \quad (4.4.3)$$

Os resultados são:

Tabela 4.47: Tabela de Contingência - NB - Australiano

	Classificado como Positivo	Classificado como Negativo
Positivo	57	6
Negativo	16	59

Tabela 4.48: Tabela de Resultados - NB - Australiano

	Tipo	Acurácia	F1 (+)	Área ROC	Kappa
Treino (K-Fold)	0.8666922	0.8571033	-	-	-
Teste	0.8405797	0.8382353	0.8973545	0.6827586	

O método NB teve resultados ligeiramente superiores aos da Árvore de Decisão.

4.4.8 Comparação

A Tabela 4.49 a seguir sumariza os principais resultados para os métodos utilizados:

Tabela 4.49: Comparativo de Resultados - Banco Australiano

Medida	Aleatório	KNN	ANN	SVM	Árvore	Logística	NB
Acurácia	0.507	0.833	0.884	0.848	0.833	0.855	0.841
Dp. Acurácia	0.043	0.032	0.031	0.031	0.032	0.030	0.031
Precisão	0.464	0.786	0.862	0.784	0.763	0.803	0.781
Recall	0.508	0.873	0.889	0.921	0.921	0.905	0.905
F1 Score	0.485	0.827	0.875	0.847	0.835	0.851	0.838
Taxa FP	0.493	0.200	0.120	0.213	0.240	0.187	0.213
Taxa TP	0.508	0.873	0.889	0.921	0.921	0.905	0.905
Kappa	0.014	0.667	0.767	0.698	0.670	0.711	0.683
Precisão C.Neg	0.551	0.882	0.904	0.922	0.919	0.910	0.908
Recall C.Neg	0.507	0.800	0.880	0.787	0.760	0.813	0.787
F1 C.Neg	0.528	0.839	0.892	0.849	0.832	0.859	0.843
Área ROC	0.507	0.896	0.947	0.914	0.891	0.930	0.897
Tempo Ajuste	0.001 s	0.191 s	0.322 s	0.351 s	0.079 s	0.056 s	0.01 s
Tempo Predição	0.001 s	0.043 s	0.009 s	0.02 s	0.005 s	0.011 s	0.172 s

A Figura 4.29 apresenta a curva ROC para os modelos ajustados.

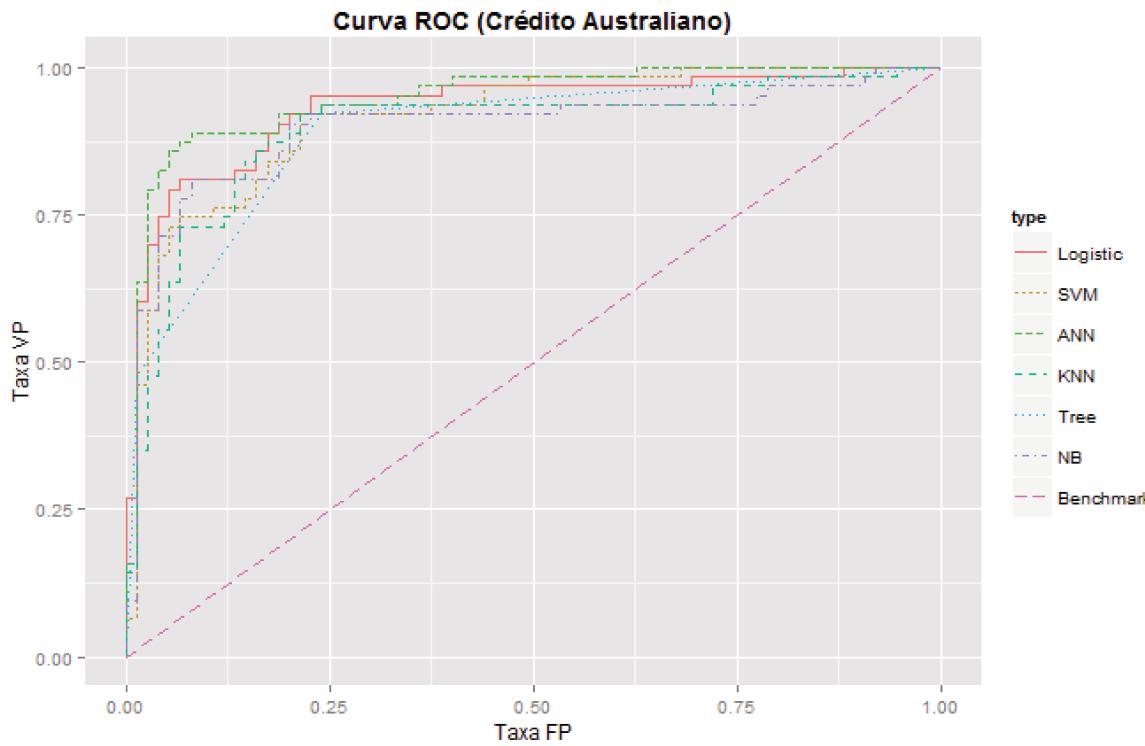


Figura 4.29: Curva Roc - Australiano

4.4.9 Conclusão

Nesta aplicação, o método Redes Neurais obteve o melhor desempenho na maioria das métricas (considerando Classe Positiva como referência), ganhando na Acurácia, Precisão, F1, Kappa e Área ROC, e perdendo apenas em relação ao *recall*. Isso indica que o critério da Rede Neural erra menos para a classe positiva, porém o número absoluto de acertos na classe positiva é menor do que em métodos como SVM e Árvore de Decisão.

Ainda assim, devido a superioridade nas outras métricas, consideramos a Rede Neural como o melhor método para essa aplicação.

Como comparação, Huo et al. [12], obtiveram a melhor acurácia no conjunto de teste de 85.16% utilizando uma árvore de decisão com poda Custo-Complexidade, denominada CCP. Neste trabalho, o melhor resultado para a acurácia foi de 88.4%, utilizando a Rede Neural.

Capítulo 5

Conclusão

Nessa dissertação de mestrado estudamos alguns métodos para resolver o problema de classificação binária, assim como técnicas para pré-processamento de dados e avaliação e comparação dos resultados obtidos pelos modelos. Os métodos foram aplicados em três conjuntos de dados, e para cada aplicação, os melhores modelos foram selecionados de acordo com as diversas métricas analisadas.

Na etapa de pré-processamento de dados, notamos a importância da limpeza dos dados e do tratamento dos dados faltantes, possibilitando a utilização da totalidade de observações disponíveis através do preenchimento dos dados. A técnica de balanceamento de classes se mostrou muito importante ao lidar com classes com poucas ocorrências e foi crucial para o correto ajuste dos modelos.

Dos métodos estudados, a maioria requereu algum tipo de seleção de hiper-parâmetros, e as buscas utilizadas, em sua maior parte baseadas em busca em grade, se mostraram bastante custosas computacionalmente, e para trabalhos futuros, seria interessante explorar alternativas como os Algoritmos Genéticos.

Nas aplicações ficou evidente que cada conjunto de dados tem suas peculiaridades, e não foi possível definir um método como o melhor de forma genérica, sendo necessária uma análise cuidadosa caso a caso. Além disso, a seleção do método depende das métricas escolhidas como mais importantes, métricas estas que devem ser selecionadas de acordo com os objetivos de cada estudo.

Observamos que uma atenção especial deve ser dada ao problema de sobre-ajuste, especialmente ao trabalharmos com redes neurais, onde o número de parâmetros pode crescer rapidamente dependendo da topologia da rede.

Finalmente, os resultados obtidos pelos melhores modelos de cada aplicação foram similares ao de outros trabalhos que utilizaram os mesmos conjuntos de dados.

Referências Bibliográficas

- [1] H. Akaike. A new look at the statistical model identification. *IEEE Trans. Automatic Control* 19(6), page 716–723, 1974.
- [2] K. Alexandros, M. David, and H. Kurt. Support vector machines in R. *Journal of Statistical Software, Volume 15, Issue 9.*, April 2006.
- [3] K. Bache and M. Lichman. UCI machine learning repository, 2013.
- [4] P. Breheny. Advanced regression (notes), April 2012. <http://web.as.uky.edu/statistics/users/pbreheny/760/S11/notes.html>.
- [5] J. W. Chinneck. *Practical optimization: a gentle introduction*. Carleton University, 2012.
- [6] J. Eggermont, J. N. Kok, and W. A. Kosters. Genetic programming for data classification: Partitioning the search space. In *Proceedings of the 2004 Symposium on applied computing (ACM SAC'04)*, pages 1001–1005. ACM, 2004.
- [7] E. Fix and J. L. Hodges. Discriminatory analysis, nonparametric discrimination: Consistency properties. *Technical Report 4, USAF School of Aviation Medicine, Randolph Field, Texas*, 1951.
- [8] J. Han, M. Kamber, and J. Pei. *Data Mining: Concepts and Techniques - Third Edition*. Morgan Kaufmann, Waltham–MA, 2012.
- [9] J. W. Harris and H. Stocker. *Handbook of Mathematics and Computational Science*. Springer-Verlag, New York, 1998.
- [10] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2013.
- [11] K. Hornik, C. Buchta, and A. Zeileis. Open-source machine learning: R meets Weka. *Computational Statistics*, 24(2):225–232, 2009.

- [12] X. Huo. FBP: A frontier-based tree-pruning algorithm, Oct 2002. School of Industrial and Systems Engineering, Georgia Institute of Technology.
- [13] RStudio Inc. *RStudio: Integrated development environment for R (Version 0.98.994) [Computer software]*. Boston, MA, 2014.
- [14] M. H. Kutner, C. J. Nachtsheim, and J. Neter. *Applied Linear Regression Models (4th ed.)*. McGraw-Hill Irwin, 2004.
- [15] D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, and F. Leisch. *e1071: Misc Functions of the Department of Statistics (e1071)*, TU Wien, 2014. R package version 1.6-4.
- [16] T. P. Minka. Algorithms for maximum-likelihood logistic regression, September 2003.
- [17] G. G. Moisen. Classification and regression trees. *Sven Erik Jørgensen and Brian D. Fath (Eds.), Ecological Informatics. Encyclopedia of Ecology, Volume 1: 582-588.*, 2008.
- [18] S. Moro, R. M. S. Laureano, and P. Cortez. Using data mining for bank direct marketing: An application of the crisp-dm methodology. In *P. Novais et al. (Eds.), Proceedings of the European Simulation and Modelling Conference - ESM2011*, pages 117–121, 2011.
- [19] A. Ng. Machine learning, coursera.org, June 2013.
- [20] A. A. M. Nurunnabi and Mohammed Nasser. Outlier diagnostics in logistic regression: A supervised learning technique. *2009 International Conference on Machine Learning and Computing IPCSIT vol.3 (2011), IACSIT Press, Singapore*, 2011.
- [21] G. A. Paula. Modelos de regressão com apoio computacional, 2013.
- [22] R. C. Prati, G. E. A. P. A. Batista, and M. C. Monard. Curvas ROC para avaliação de classificadores. *Revista IEEE América Latina 6 (2), 215-222*, 2008.
- [23] B. Ripley. *tree: Classification and regression trees*, 2014. R package version 1.0-35.
- [24] R. Silva, T. Macedo, J. Soares, et al. Modelo de dissertação/tese do instituto de matemática, estatística e computação científica (IMECC) da universidade estadual de campinas (UNICAMP), 2013.
- [25] M. Stevenson. An introduction to logistic regression. Technical report, 2008.
- [26] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2014.

- [27] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. ISBN 0-387-95457-0.
- [28] S. Weisberg. Yeo-johnson power transformations, 2001.
- [29] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco, 2nd edition, 2005.

Apêndice A

Tabelas de Seleção de Parâmetros

Apresentamos neste apêndice as tabelas de resultados utilizadas para a seleção de parâmetros dos métodos SVM e Redes Neurais.

Para cada combinação de conjunto de dados e método, analisamos duas tabelas, a primeira com os valores ordenados pela métrica F1 e a segunda pela métrica Acurácia. O objetivo é de, em conjunto com a análise gráfica, encontrar combinações de parâmetros que estejam entre as melhores em relação a ambas as métricas.

A linha destacada em cada tabela indica a combinação de parâmetros selecionada.

A.1 Tabelas - Banco Português

A.1.1 SVM

Id	Kernel	Gamma	Custo	Acurácia	F1	Precisão	Recall
41	radial	0.01	2.51	0.838	0.542	0.405	0.821
49	radial	0.01	3.01	0.838	0.542	0.405	0.821
33	radial	0.01	2.01	0.838	0.542	0.404	0.821
25	radial	0.01	1.51	0.837	0.541	0.404	0.818
17	radial	0.01	1.01	0.836	0.538	0.401	0.817
18	radial	0.26	1.01	0.832	0.538	0.397	0.833
26	radial	0.26	1.51	0.833	0.536	0.397	0.824
34	radial	0.26	2.01	0.833	0.534	0.396	0.820
10	radial	0.26	0.51	0.827	0.532	0.389	0.842
42	radial	0.26	2.51	0.833	0.532	0.395	0.815

Tabela A.1: Seleção de Parâmetros - SVM - Português - 10 melhores em F1

Id	Kernel	Gamma	Custo	Acurácia	F1	Precisão	Recall
16	radial	1.76	0.51	0.858	0.242	0.327	0.197
24	radial	1.76	1.01	0.856	0.235	0.310	0.190
32	radial	1.76	1.51	0.854	0.238	0.306	0.195
48	radial	1.76	2.51	0.854	0.233	0.302	0.190
40	radial	1.76	2.01	0.854	0.235	0.302	0.193
54	radial	1.76	3.01	0.854	0.230	0.299	0.188
23	radial	1.51	1.01	0.853	0.323	0.351	0.302
47	radial	1.51	2.51	0.851	0.312	0.339	0.291
39	radial	1.51	2.01	0.851	0.315	0.340	0.295
31	radial	1.51	1.51	0.851	0.317	0.341	0.298
15	radial	1.51	0.51	0.850	0.342	0.364	0.348
22	radial	1.26	1.01	0.849	0.443	0.390	0.520
30	radial	1.26	1.51	0.848	0.430	0.383	0.494
46	radial	1.26	2.51	0.847	0.416	0.376	0.470
38	radial	1.26	2.01	0.847	0.420	0.377	0.479
41	radial	0.01	2.51	0.838	0.542	0.405	0.821
49	radial	0.01	3.01	0.838	0.542	0.405	0.821
33	radial	0.01	2.01	0.838	0.542	0.404	0.821
25	radial	0.01	1.51	0.837	0.541	0.404	0.818
14	radial	1.26	0.51	0.837	0.474	0.387	0.636

Tabela A.2: Seleção de Parâmetros - SVM - Português - 20 melhores em Acurácia

A.1.2 ANN

Id	Tamanho	Decay	Acurácia	F1	Precisão	Recall
12	10	0.6	0.906	0.540	0.632	0.472
14	14	0.6	0.905	0.539	0.627	0.474
3	6	0.1	0.905	0.535	0.625	0.469
4	8	0.1	0.905	0.534	0.631	0.463
10	6	0.6	0.906	0.533	0.636	0.460

Tabela A.3: Seleção de Parâmetros - ANN - Português - 5 melhores em F1

Id	Tamanho	Decay	Acurácia	F1	Precisão	Recall
10	6	0.6	0.906	0.533	0.636	0.460
12	10	0.6	0.906	0.540	0.632	0.472
33	10	2.1	0.906	0.523	0.643	0.442
18	8	1.1	0.906	0.526	0.639	0.447
19	10	1.1	0.906	0.527	0.638	0.448

Tabela A.4: Seleção de Parâmetros - ANN - Português - 5 melhores em Acurácia

A.2 Tabelas - Banco Alemão

A.2.1 SVM

Id	Kernel	Gamma	Custo	Acurácia	F1	Precisão	Recall
106	radial	0.02	2.01	0.723	0.581	0.521	0.672
93	radial	0.02	1.76	0.720	0.580	0.517	0.676
17	radial	0.06	0.26	0.717	0.579	0.514	0.679
54	radial	0.02	1.01	0.712	0.577	0.504	0.691
119	radial	0.02	2.26	0.723	0.577	0.520	0.663
15	radial	0.02	0.26	0.702	0.575	0.494	0.705
67	radial	0.02	1.26	0.715	0.575	0.510	0.675
41	radial	0.02	0.76	0.707	0.573	0.500	0.688
16	radial	0.04	0.26	0.706	0.572	0.501	0.683
145	radial	0.02	2.76	0.724	0.571	0.523	0.645
132	radial	0.02	2.51	0.722	0.571	0.521	0.650
29	radial	0.04	0.51	0.713	0.571	0.509	0.667
80	radial	0.02	1.51	0.715	0.570	0.510	0.662
171	radial	0.02	3.26	0.726	0.568	0.526	0.634
55	radial	0.04	1.01	0.727	0.566	0.530	0.625
158	radial	0.02	3.01	0.723	0.566	0.522	0.634
28	radial	0.02	0.51	0.704	0.566	0.496	0.674
18	radial	0.08	0.26	0.713	0.565	0.507	0.654
107	radial	0.04	2.01	0.738	0.563	0.547	0.592
69	radial	0.06	1.26	0.747	0.563	0.564	0.574

Tabela A.5: Seleção de Parâmetros - SVM - Alemão - 20 melhores em F1

Id	Kernel	Gamma	Custo	Acurácia	F1	Precisão	Recall
193	radial	0.20	3.51	0.750	0.387	0.683	0.279
46	radial	0.12	0.76	0.749	0.524	0.580	0.493
206	radial	0.20	3.76	0.749	0.382	0.679	0.274
141	radial	0.20	2.51	0.749	0.377	0.688	0.271
154	radial	0.20	2.76	0.749	0.375	0.683	0.270
167	radial	0.20	3.01	0.749	0.381	0.677	0.274
180	radial	0.20	3.26	0.749	0.381	0.677	0.274
128	radial	0.20	2.26	0.747	0.382	0.673	0.275
69	radial	0.06	1.26	0.747	0.563	0.564	0.574
166	radial	0.18	3.01	0.746	0.399	0.656	0.299
179	radial	0.18	3.26	0.746	0.394	0.656	0.294
204	radial	0.16	3.76	0.745	0.413	0.637	0.316
152	radial	0.16	2.76	0.745	0.414	0.638	0.317
165	radial	0.16	3.01	0.745	0.414	0.638	0.317
178	radial	0.16	3.26	0.745	0.414	0.638	0.317
191	radial	0.16	3.51	0.745	0.417	0.638	0.321
182	radial	0.24	3.26	0.745	0.337	0.696	0.229
195	radial	0.24	3.51	0.745	0.337	0.696	0.229
208	radial	0.24	3.76	0.745	0.337	0.696	0.229
192	radial	0.18	3.51	0.745	0.389	0.654	0.289

Tabela A.6: Seleção de Parâmetros - SVM - Alemão - 20 melhores em Acurácia

A.2.2 ANN

Id	Tamanho	Decay	Acurácia	F1	Precisão	Recall
21	14	0.3	0.743	0.552	0.559	0.559
40	10	0.6	0.754	0.552	0.603	0.527
49	14	0.7	0.752	0.548	0.600	0.530
62	12	0.9	0.757	0.537	0.615	0.495
32	8	0.5	0.741	0.534	0.574	0.518

Tabela A.7: Seleção de Parâmetros - ANN - Alemão - 5 melhores em F1

Id	Tamanho	Decay	Acurácia	F1	Precisão	Recall
62	12	0.9	0.757	0.537	0.615	0.495
58	4	0.9	0.755	0.489	0.554	0.449
40	10	0.6	0.754	0.552	0.603	0.527
49	14	0.7	0.752	0.548	0.600	0.530
69	12	1.0	0.750	0.532	0.588	0.496

Tabela A.8: Seleção de Parâmetros - ANN - Alemão - 5 melhores em Acurácia

A.3 Tabelas - Aprovação de Crédito Australiano

A.3.1 SVM

Id	Kernel	Gamma	Custo	Acurácia	F1	Precisão	Recall
1,088	radial	0.04	2.73	0.865	0.855	0.805	0.917
1,128	radial	0.04	2.83	0.865	0.855	0.805	0.917
243	radial	0.02	0.63	0.863	0.854	0.796	0.925
283	radial	0.02	0.73	0.863	0.854	0.796	0.925
1,048	radial	0.04	2.63	0.864	0.854	0.802	0.917

Tabela A.9: Seleção de Parâmetros - SVM - Australiano - 5 melhores em F1

Id	Kernel	Gamma	Custo	Acurácia	F1	Precisão	Recall
1,088	radial	0.04	2.73	0.865	0.855	0.805	0.917
1,128	radial	0.04	2.83	0.865	0.855	0.805	0.917
453	radial	0.07	1.13	0.864	0.853	0.801	0.917
492	radial	0.06	1.23	0.864	0.853	0.801	0.917
493	radial	0.07	1.23	0.864	0.853	0.801	0.917

Tabela A.10: Seleção de Parâmetros - SVM - Australiano - 5 melhores em Acurácia

A.3.2 ANN

Id	Tamanho	Decay	Acurácia	F1	Precisão	Recall
36	2	1.1	0.854	0.832	0.821	0.852
76	12	2.1	0.851	0.829	0.827	0.836
42	14	1.1	0.851	0.829	0.826	0.836
29	2	0.9	0.851	0.829	0.822	0.844
47	10	1.3	0.849	0.828	0.820	0.843

Tabela A.11: Seleção de Parâmetros - ANN - Australiano - 5 melhores em F1

Id	Tamanho	Decay	Acurácia	F1	Precisão	Recall
36	2	1.1	0.854	0.832	0.821	0.852
97	12	2.7	0.852	0.827	0.833	0.828
76	12	2.1	0.851	0.829	0.827	0.836
42	14	1.1	0.851	0.829	0.826	0.836
29	2	0.9	0.851	0.829	0.822	0.844

Tabela A.12: Seleção de Parâmetros - ANN - Australiano - 5 melhores em Acurácia