MODELOS PROBABILÍSTICOS E NÃO PROBABILÍSTICOS DE CLASSIFICAÇÃO BINÁRIA PARA PACIENTES COM OU SEM DEMÊNCIA COMO AUXÍLIO NA PRÁTICA CLÍNICA EM GERIATRIA

Maicon Vinícius Galdino

Tese apresentada à Universidade Estadual Paulista "Júlio de Mesquita Filho", como parte dos requisitos necessários para obtenção do título de Doutor em Biometria
Linha de pesquisa: Bioestatística

Botucatu
São Paulo - Brasil
Fevereiro – 2020

MODELOS PROBABILÍSTICOS E NÃO PROBABILÍSTICOS DE CLASSIFICAÇÃO BINÁRIA PARA PACIENTES COM OU SEM DEMÊNCIA COMO AUXÍLIO NA PRÁTICA CLÍNICA EM GERIATRIA

Maicon Vinícius Galdino

Orientadora: Profa. Dra. Liciana Vaz de Arruda Silveira

Tese apresentada à Universidade Estadual Paulista "Júlio de Mesquita Filho", como parte dos requisitos necessários para obtenção do título de Doutor em Biometria
Linha de pesquisa: Bioestatística

Botucatu
São Paulo - Brasil
Fevereiro – 2020

G149m

Galdino, Maicon Vinícius

Modelos probabilísticos e não probabilísticos de classificação binária para pacientes com ou sem demência como auxílio na prática clínica em geriatria / Maicon Vinícius Galdino. -- Botucatu, 2020

92 p.

Tese (doutorado) - Universidade Estadual Paulista (Unesp), Instituto de Biociências, Botucatu

Orientadora: Liciana Vaz de Arruda Silveira

1. Regressão Logística - Naive Bayes. 2. Árvore de Classificação - Random Forrest. 3. Algoritmo kNN - Redes Neurais Artificiais. 4. Comprometimento cognitivo - Demência. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca do Instituto de Biociências, Botucatu. Dados fornecidos pelo autor(a).

Dedicatória

Aos meus pais Edison Galdino e Ivani Freire Galdino.
À minha esposa Juliana Madureira dos Santos Galdino.

Agradecimentos

Obrigado a todos aqueles que de forma direta ou indireta contribuíram para a realização deste trabalho.

Um agradecimento especial a minha amiga e orientadora Prof(a). Dr(a). Liciana Vaz de Arruda Silveira por todas as sugestões, dicas, conselhos e conversas.

Aos membros da banca Prof. Dr. Rogério Antonio Oliveira, Prof(a). Dr(a). Miriam Harumi Tsunemi, Prof. Dr. Alessandro Ferrari Jacinto e Prof(a) Dr(a) Silvia Helena Modenese Gorla da Silva pelas sugestões feitas com o objetivo de melhorar meu trabalho.

Ao médico geriatra Dr. Alessandro Ferrari Jacinto por ter gentilmente cedido o conjunto de dados utilizados nesta tese.

Ao Prof. Dr. José Raimundo de Souza Passos e à Prof(a). Dr(a). Luzia Aparecida Trinca pelas valiosas sugestões.

À Prof(a). Dr(a). Miriam Rodrigues Silvestre por ter sempre me incentivado, desde os tempos de Graduação, ao estudo dos modelos estatísticos que envolvem classificação.

Agradeço também a todos os professores do Curso de Doutorado por terem contribuído para o meu crescimento pessoal e profissional.

A todos os meus amigos e colegas do Programa de Pós Graduação em Biometria e do Departamento de Bioestatística do Instituto de Biociências de Botucatu.

À Capes, mantenedora do Programa de Pós Graduação em Biometria.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

MODELOS PROBABILÍSTICOS E NÃO PROBABILÍSTICOS DE CLASSIFICAÇÃO BINÁRIA PARA PACIENTES COM OU SEM DEMÊNCIA COMO AUXÍLIO NA PRÁTICA CLÍNICA EM GERIATRIA

Autor: Maicon Vinícius Galdino

Orientadora: Prof(a). Dr(a). Liciana Vaz de Arruda Silveira

RESUMO

Médicos generalistas costumam lidar com pacientes idosos que frequentemente apresentam distúrbios cognitivos. Estes profissionais muitas vezes precisam fazer um diagnóstico e este envolve classificar o paciente como “suspeito” ou “não suspeito” de possuírem alterações cognitivas. Surge então, a partir desta situação, dados de classificação binária.

Para se modelar dados neste contexto, existem várias metodologias. Os métodos que consideram probabilidades (ou distribuições de probabilidades) na estimação da classificação recebem o nome de modelos probabilísticos. Dois deles são bem conhecidos e amplamente utilizados: os Modelos de Regressão Logística e o algoritmo *Naive Bayes*.

Modelos que desconsideram a probabilidade de o evento ocorrer geralmente são chamados de não probabilísticos. Alguns deles são: Árvore de Classificação, Floresta Aleatória, k −vizinhos mais próximos e Redes Neurais Artificiais.

Os objetivos deste trabalho foram apresentar modelos de classificação (Regressão Logística, *Naive Bayes*, Árvores de Classificação, Floresta Aleatória, k −Vizinhos mais próximos e Redes Neurais Artificiais) e a comparação destes utilizando processos de reamostragem em um conjunto de dados da área de geriatria (diagnóstico de demência).

Os dados utilizados neste trabalho foram obtidos por Jacinto (2008). Os 245 pacientes participantes foram submetidos a vários testes clínicos. A partir de dois destes, o idoso era classificado da seguinte forma: “1” como suspeito de possuir alterações cognitivas e “0”, caso contrário.

Para a realização da comparação ou validação dos modelos, foram tomadas 1000 amostras de treinamento e 1000 amostras de testes. Com isso foi possível obter a distribuição empírica das seguintes medidas de desempenho: acurácia, sensibilidade e especificidade.

Num primeiro momento, para a construção dos modelos, foram utilizadas todas as covariáveis coletadas por Jacinto (2008). Estes apresentaram acurácia média e mediana em torno de 85%. Os modelos que apresentaram as maiores medidas em relação a acurácia e sensibilidade foram as Florestas Aleatórias e Árvores de Classificação, respectivamente.

Após a seleção das covariáveis mais significativas, os seis modelos considerados apresentaram acurácia média e mediana em torno de 90% e os modelos de Regressão Logística apresentaram os maiores valores para a acurácia mediana (90,54%) e sensibilidade mediana (93,33%).

Como Jacinto (2008) concluiu que o declínio cognitivo em idosos é pouco detectado por médicos generalistas, os modelos de Regressão Logística podem ser um ótimo instrumento de auxílio a estes profissionais, desde que disponibilizado de um modo prático e funcional.

PROBABILISTIC AND NON-PROBABILISTIC MODELS OF BINARY CLASSIFICATION FOR PATIENTS WITH OR WITHOUT DEMENTIA AS AID IN CLINICAL PRACTICE IN GERIATRY

Author: Maicon Vinícius Galdino

Adviser: Prof(a). Dr(a). Liciana Vaz de Arruda Silveira

SUMMARY

Generalist doctors often deal with elderly patients who often have cognitive disturbances. These professionals often need to make a diagnosis and this involves classifying the patient as "suspected" or "not suspected" of having cognitive alterations. Then appears, from this situation, binary classification data.

To model data in this context, there are various methodologies. Methods that consider probabilities (or probability distributions) in estimating the classification are called probabilistic models. Two of them are well known and widely used: Logistic Regression Models and Naive Bayes algorithm.

Models that disregard the probability of an event will occur are often called non-probabilistic models. Some of them are: Classification Tree, Random Forest, k −Nearest Neighbors and Artificial Neural Networks.

The objectives of this work were to show classification models (Logistic Regression, Naive Bayes, Classification Trees, Random Forest, k −Nearest Neighbors and Artificial Neural Networks) and the comparison of these using resampling processes in a data set in the field of geriatrics (diagnosis of dementia).

The data used in this work were obtained by Jacinto (2008). The 245 participating patients underwent various clinical tests. From two of these, the

elderly person was classified as follows: “1” as suspected of having cognitive alterations and “0” otherwise.

For the comparison or validation of the models, 1000 training samples and 1000 test samples were taken. Thus, it was possible to obtain the empirical distribution of the following performance measures: accuracy, sensitivity and specificity.

At first, for the construction of the models, all the covariables collected by Jacinto (2008) were used. These showed mean and median accuracy around 85%. The models that presented the highest measures in relation to accuracy and sensitivity were Random Forests and Classification Trees, respectively.

After the selection of the most significant covariables, the six models considered showed accuracy around 90% and the Logistic Regression models presented the highest values for median accuracy (90.54%) and median sensitivity (93.33%).

As Jacinto (2008) concluded that cognitive decline in the elderly is little detected by generalist doctors, the Logistic Regression models can be a great instrument to help these professionals, as long as they are made available in a practical and functional way.

1) INTRODUÇÃO

Em muitas pesquisas é comum ser do interesse do pesquisador que dado evento seja classificado em uma ou outra categoria, por exemplo: se ocorreu recidiva ou não de determinada doença em plantas, se determinado cliente é fiel a determinada marca ou não, se determinada criança está alfabetizada ou não depois de exposta a um determinado método de ensino ou se certo cliente deve ser classificado como adimplente ou inadimplente tendo como referência o histórico de pagamentos de faturas do cartão de crédito, dentre outras.

Muitos são os exemplos em pesquisas clínicas em que se tem interesse em classificar em duas categorias, por exemplo, se um paciente veio a óbito ou não por infarto agudo do miocárdio, se um indivíduo desenvolveu determinada doença ou não quando exposto a um agente patológico, se determinado tipo de câncer reincidiu ou não no paciente depois de um tratamento quimioterápico, etc.

Além da classificação, pesquisadores também têm o interesse de saber quais são os fatores que mais contribuíram para que o indivíduo fosse classificado em uma determinada categoria. Esses fatores são chamados de covariáveis, variáveis preditoras, atributos ou variáveis de entrada; e a classificação final recebe o nome de variável resposta, atributo-classe ou variável de saída.

Para exemplificar, Rau et. al (2016) desenvolveram um modelo de previsibilidade do câncer de fígado (variável resposta) para pacientes portadores de diabetes tipo II. Para isso, coletaram as seguintes informações sobre os pacientes (dez variáveis explicativas ou covariáveis): sexo, idade, cirrose alcoólica, cirrose não-alcoólica, hepatite alcoólica, outros tipos de hepatite crônica, doença hepática alcoólica, hepatite viral, outros tipos de doenças hepáticas gordurosas e hiperlipidemia (distúrbio nos níveis de lipídios ou lipoproteínas no sangue).

A distribuição de probabilidade associada ao evento que recebe “0” se não ocorreu e “1” se ocorreu o evento é denominada distribuição Bernoulli. “Analistas de dados normalmente querem que seus modelos sejam com base na

probabilidade do evento, que muitas vezes é codificado como “1” (STOKES, DAVIS, KOCK, 2000).

Os modelos que consideram probabilidades (ou distribuições de probabilidades) na estimação da classificação recebem o nome de modelos probabilísticos. Os modelos que desconsideram a probabilidade de o evento ocorrer e que se baseiam em outras medidas, tais como distâncias, métodos de procura ou baseados em algoritmos de otimização são geralmente denominados não probabilísticos.

Existem diversas metodologias probabilísticas de classificação que têm por finalidade analisar dados binários. Duas delas são: Modelos de Regressão Logística (RL) e *Naive Bayes* (NB).

Sobre os Modelos de Regressão Logística, durante muitos anos eles foram a principal metodologia utilizada para se analisar dados com variável resposta binária e uma ou mais variáveis explicativas (HOSMER, LEMESHOW, STURDIVANT, 2013; GIOLO, 2017).

Agresti (2013) comenta que os Modelos de Regressão Logística primariamente foram usados em estudos biomédicos. Devido ao fato de seus parâmetros serem de fácil interpretação, são frequentemente utilizados em estudos epidemiológicos.

O algoritmo de classificação *Naive Bayes* (Bayesiano Ingênuo) utiliza as probabilidades de cada variável preditora pertencente a cada classe no conjunto de treinamento para prever a classe de novas observações. Esse algoritmo supõe que as variáveis preditoras são independentes entre si (TORGO, 2010).

Há metodologias cujo objetivo também é o de classificar elementos em uma ou outra categoria distinta, mas que não se baseiam em teoria probabilística. Algumas delas são: Árvore de Classificação, Floresta Aleatória (*Random Forest*), k −Vizinhos mais próximos (k-*Nearest Neighbor* ou $k - NN$), Redes Neurais Artificiais (RNA), dentre outras.

Quanto às Árvores de Classificação (AC), estas são compostas de nós e de regras de decisão. O nó inicial corresponde ao atributo mais importante, enquanto os nós-filhos (intermediários ou folhas) são menos relevantes. Assim,

o primeiro nó é dividido em dois nós através de uma regra de decisão. Essa regra é repetida até que chegue ao nó terminal. A finalidade é obter subconjuntos dos dados sendo eles os mais homogêneos possíveis em relação à variável resposta.

A metodologia Floresta Aleatória (*Random Forest* - RF) se baseia em um conjunto de algoritmos que cria várias árvores de decisão e classifica novos dados recebidos em uma classe. O termo “aleatório” vem da seleção de n variáveis preditoras para encontrar o melhor ponto de divisão usando uma função de custo ao construir árvores de decisão.

O algoritmo $k - NN$ é um método de classificação baseado em distâncias. Ele calcula a distância de uma nova observação a todas as observações do conjunto de dados. Desta forma, a nova observação será classificada com o mesmo rótulo que a observação mais próxima dela.

As Redes Neurais Artificiais (RNA) é uma área de estudo que busca reproduzir a forma de funcionamento do cérebro de seres vivos, que tem ampla capacidade de aprendizado. Os neurônios artificias operam em uma rede de camadas, frequentemente uma camada de entrada, uma camada oculta e uma de saída.

O neurônio recebe uma entrada alterada através de um peso, produz uma resposta que é avaliada quanto ao erro encontrado; este erro é então utilizado para ajustar o peso de entrada.

“A RNA classifica corretamente um objeto quando o valor de saída mais elevado produzido pela rede é aquele gerado pelo neurônio de saída que corresponde à classe correta do objeto” (FACELI et al, 2017).

Os profissionais da área da saúde (especialmente médicos) lidam com pacientes idosos que frequentemente apresentam distúrbios cognitivos. Estes profissionais muitas vezes precisam fazer um diagnóstico e este envolve classificar o paciente como “suspeito” ou “não suspeito” de possuir algum tipo de alteração cognitiva.

Dentre os distúrbios mentais na população idosa (acima de 65 anos) estão os processos demenciais. “Demência é uma síndrome caracterizada pelo

declínio da capacidade intelectual, suficientemente grave para interferir nas atividades sociais ou profissionais, que independe de distúrbio do estado de consciência (ou da vigília) e é causada por comprometimento do sistema nervoso central” (FORLENZA, MIGUEL, 2012).

“O diagnóstico precoce pode, em alguns casos, reverter o processo e, em outros, amenizar suas consequências” (FORLENZA, MIGUEL, 2012).

Diante do exposto, este trabalho ficou estruturado no seguinte formato: o Capítulo 2 mostrará os objetivos, justificativa e relevância deste trabalho. O Capítulo 3 abordará os fundamentos teórico-metodológicos dos modelos probabilísticos e não-probabilísticos de classificação (noção intuitiva, aplicações, definições formais, cálculos envolvidos, propriedades, aspectos positivos e negativos). O Capítulo 4 discorrerá sobre medidas de desempenho dos modelos. O Capítulo 5 descreverá o conjunto de dados de pacientes diagnosticados com ou sem demência. O capítulo 6 será destinado para analisar e discutir os resultados que os modelos de classificação tiveram frente a análise do conjunto de dados e o Capítulo 7 será dedicado às considerações finais.

2) OBJETIVOS E JUSTIFICATIVA

Os objetivos deste trabalho foram apresentar modelos de classificação (Regressão Logística, *Naive Bayes*, Árvores de Classificação, *Random Forest*, $k-$Vizinhos mais próximos e Redes Neurais Artificiais) e a comparação destes utilizando processos de reamostragem em um conjunto de dados da área de geriatria (diagnóstico de demência). Analisar as pressuposições de cada metodologia, vantagens, desvantagens e cenários em que cada metodologia pode ser melhor utilizada.

A justificativa e relevância desse projeto se baseiam na importância e na utilidade do tema proposto, visto que a população idosa aumenta em todo o mundo (nos países desenvolvidos e nos em desenvolvimento como o Brasil), os modelos de classificação podem ser úteis aos profissionais médicos, em especial aos médicos generalistas, no diagnóstico de demências, pois “em diversos momentos o diagnóstico não é simples” (FORLENZA, MIGUEL, 2012).

Além disso, “enfatiza-se que a importância desse diagnóstico é enorme para o paciente, a família e a sociedade, já que a diminuição da qualidade de vida e da funcionalidade levam também ao aumento da morbimortalidade, abreviando a vida do indivíduo, bem como o aumento do uso dos sistemas de saúde, o que causa grande impacto econômico” (GORENSTEIN, WANG, HUNGERBÜHLER, 2016).

3) FUNDAMENTAÇÃO TEÓRICA

Para a construção dos modelos de classificação, frequentemente o conjunto de dados é dividido em dois grupos distintos: uma parte é usada para treinar o modelo e outra parte para testá-lo. O primeiro grupo de dados é o que, de fato, vai ser utilizado para construir o modelo. Neste caso este é chamado de amostra de treinamento ou dados de treinamento. Já o segundo grupo é utilizado para verificar o quão eficiente se mostrou o modelo construído. Este segundo grupo é denominado amostra de teste ou conjunto de dados de teste.

Considere que o conjunto de dados tenha n observações independentes, sendo que n_1 observações estão contidas no conjunto de dados de treinamento e n_2 observações no conjunto de dados de teste. Logo, $n = n_1 + n_2$.

Estas definições serão úteis para o entendimento detalhado dos modelos de classificação a seguir.

3.1) MODELOS PROBABILÍSTICOS DE CLASSIFICAÇÃO

3.1.1) Algoritmo *Naive Bayes*

Este algoritmo de classificação é fortemente apoiado na teoria de probabilidades, em especial probabilidades condicionais (HARRINGTON, 2012) e utiliza-se o teorema de Bayes na sua formulação. É frequentemente usado na análise e na classificação de textos e utilizados, por exemplo, em filtros de *spams* (KELLEHER, NAMEE, D'ARCY, 2015).

Considere C como sendo a variável aleatória que indica a classe de uma observação e c_i o valor que esta classe pode assumir ($i = 1,2$). A pergunta que surge ao utilizar esse algoritmo é: "Com base nos dados, qual é a probabilidade de que o resultado seja de uma determinada classe, digamos $C = c_i$"? (BURGER, 2018).

Para o entendimento detalhado desse algoritmo, será necessário discorrer sobre alguns conceitos da teoria de probabilidades. Seja A e B dois

eventos de S (espaço amostral). Podemos definir a probabilidade de B ocorrer dado que A ocorreu (probabilidade condicional de B dado A) da seguinte maneira:

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

em que $P(A) > 0$.

Como sabemos que $P(B \cap A) = P(A \cap B)$ temos:

$$P(A \cap B) = P(A)P(B|A). \quad (3.2.1)$$

Dividindo toda a expressão (3.2.1) por $P(B) > 0$ temos:

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

Como $P(A|B) = \frac{P(A \cap B)}{P(B)}$ (probabilidade condicional de A dado B) tem-se que:

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

Como a lei da probabilidade total diz que se $A_1, A_2, \dots, A_n$ formam uma partição finita de S, e B é um evento de S então para todo $i = 1,2, \dots, n$, tem-se que:

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

Assim, para o caso em que se tem n partições de S, há a seguinte expressão para o teorema de Bayes (DANTAS, 2008):

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

Usando este teorema, o algoritmo de classificação *Naive Bayes* calcula a probabilidade de cada classe para uma dada observação do conjunto de dados de teste como:

$$P(C = c_i | X_1 = x_1, \dots, X_p = x_p) = \frac{P(C = c_i) P(X_1 = x_1, \dots, X_p = x_p | C = c_i)}{P(X_1 = x_1, \dots, X_p = x_p)} \quad (3.2.2)$$

em que c_i $(i = 1,2)$ é a classe e $x_1, \dots, x_p$ são os valores observados dos preditores $(X_1, \dots, X_p)$ para uma dada observação do conjunto de dados de teste (TORGO, 2015).

Supondo que os valores dos preditores são independentes entre si, daí o termo *Naive* que significa ingênuo pois essa suposição em muitos experimentos do mundo real não ocorre (TORGO, 2010; BEERAVALLI, 2018), o numerador da expressão (3.2.2) pode ser reduzido à seguinte expressão:

$$P(C = c_i) P(X_1 = x_1, \dots, X_p = x_p | C = c_i) = P(C = c_i) \prod_{i=1}^{p} P(X_i = x_i | C = c_i).$$

"A vantagem de assumir que todos os preditores são independentes é que isso reduz drasticamente a complexidade dos cálculos a serem realizados" (BURGER, 2018).

Apesar de se assumir independência condicional entre os preditores, o algoritmo *Naive Bayes* se mostra eficaz na classificação (HARRINGTON, 2012) geralmente fornecendo modelos que possuem boa acurácia (KELLEHER, NAMEE, D'ARCY, 2015).

"O denominador da expressã o (3.2.2) pode ser ignorado, uma vez que é o mesmo para todas as classes, não afetando os valores relativos de suas probabilidades" (FACELI et al, 2017). Assim, a probabilidade de uma observação pertencer à classe $C = c_i$ é proporcional à expressão:

$$P(C = c_i | X_1 = x_1, \dots, X_p = x_p) \propto P(C = c_i) \prod_{i=1}^{p} P(X_i = x_i | C = c_i)$$

No caso de classificação binária, ao calcular $P(C = c_1 | X_1 = x_1, \dots, X_p = x_p)$ se tem a probabilidade de uma determinada observação ser da classe $C = c_1$ dado que possui as características contidas em $X_1, \dots, X_p$. Ao calcular o complementar dessa probabilidade, ou seja, $1 -$

$P(C = c_1 | X_1 = x_1, \dots, X_p = x_p)$ se tem a probabilidade de a observação ser classificada na outra classe $C = c_2$.

O algoritmo *Naive Bayes* classifica a observação na classe que tem a maior probabilidade (KELLEHER, NAMEE, D'ARCY, 2015). Esse método é denominado por estimativa *MAP* (do inglês, *Maximum A Posteriori*). Formalmente, a classe que deve ser associada à observação $\boldsymbol{X} = (X_1 = x_1, X_2 = x_2, \dots, X_p = x_p)$ é dada pela expressão

$$y_{MAP} = \arg max_i \, P(C = c_i | \boldsymbol{X})$$

no qual max_i retorna a classe c_i com maior probabilidade de estar associada a $\boldsymbol{X}$, que é aquela que possui o valor máximo para $P(C = c_i | \boldsymbol{X})$ (FACELI et al, 2017).

O algoritmo *Naive Bayes* possui as seguintes vantagens:

a) "Fácil e rápido para prever a classe do conjunto de dados de teste" (BEERAVALLI, 2018);

b) "Facilmente adaptado para lidar com valores ausentes" (KELLEHER, NAMEE, D'ARCY, 2015);

c) "Pode ser treinado usando um conjunto de dados relativamente pequeno" (KELLEHER, NAMEE, D'ARCY, 2015; HARRINGTON, 2012);

E as seguintes desvantagens:

a) Não tem bom desempenho para variáveis explicativas contínuas (HARRINGTON, 2012);

b) "Se a variável categórica tiver uma categoria (no conjunto de dados de teste) que não foi observada no conjunto de dados de treinamento, o modelo atribuirá uma probabilidade zero e não poderá fazer uma previsão". (BEERAVALLI, 2018);

c) "O impacto das variáveis redundantes deve ser levado em consideração" (FACELLI et al, 2017).

3.1.2) Regressão Logística

O Modelo de Regressão Logística (RL) é um modelo clássico em estatística. A RL permite estimar a probabilidade de um evento ocorrer em função de um conjunto de variáveis explicativas. Além disso, é utilizada na classificação e discriminação de grupos (JOHNSON, WICHERN, 2007). É um dos mais populares e tem aplicações em inúmeras áreas de pesquisa: medicina, seguros, instituições financeiras, epidemiologia, econometria etc.

Considere o seguinte modelo de Regressão Múltipla dado pela expressão:

$$y_i = \boldsymbol{X}_i'\boldsymbol{\beta} + \varepsilon_i$$

em que $i = 1,2, \dots, n$ e seja $\boldsymbol{X}_i' = [1, X_1, X_2, \dots, X_p]$ um vetor que contém p variáveis independentes (covariáveis de escala intervalar ou de razão), $\boldsymbol{\beta}' = [\beta_0, \beta_1, \dots, \beta_p]$ é o vetor de parâmetros, y_i é a variável dependente (ou variável resposta) que pode assumir os valores “0” ou ”1”, conforme já mencionado anteriormente e $\varepsilon_i = 1 - \boldsymbol{X}_i'\boldsymbol{\beta}$ quando $y_i = 1$ e $\varepsilon_i = -\boldsymbol{X}_i'\boldsymbol{\beta}$ quando $y_i = 0$.

Segundo Montgomery, Peck e Vining (2006) a variável y_i é uma variável aleatória de Bernoulli com distribuição de probabilidade dada por: $P(y_i = 1|\boldsymbol{X}_i) = \pi(\boldsymbol{X}_i)$ e $P(y_i = 0|\boldsymbol{X}_i) = 1 - \pi(\boldsymbol{X}_i)$. Sendo assim, a função de ligação $logito = \left(\frac{\pi(\boldsymbol{X}_i)}{1-\pi(\boldsymbol{X}_i)}\right)$ do modelo de regressão logística múltipla é dada pela equação:

$$g(\boldsymbol{X}_i) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p = \boldsymbol{X}_i'\boldsymbol{\beta}$$

e consequentemente o modelo de regressão logística é dado por:

$$\pi(\boldsymbol{X}_i) = \frac{exp(g(\boldsymbol{X}_i))}{1 + exp(g(\boldsymbol{X}_i))} = \frac{exp(\boldsymbol{X}_i'\boldsymbol{\beta})}{1 + exp(\boldsymbol{X}_i'\boldsymbol{\beta})}$$

Se algumas variáveis que compõem o vetor $\boldsymbol{X}_i'$ forem discretas (de escala nominal ou ordinal, como por exemplo, sexo, classe socioeconômica, grau de escolaridade, etc) “então é inapropriado incluí-las no modelo como se fossem de escala intervalar pois os números usados para representar os vários níveis são simplesmente identificadores e não têm significado numérico” (HOSMER, LEMESHOW, STURDIVANT, 2013).

Nesses casos utilizam-se variáveis chamadas indicadoras (*dummy variables*). Para exemplificar, considere a variável “exame de glicose” que tem como objetivo medir a quantidade de açúcar no sangue em jejum. Considere que este exame pode ser categorizado como “normal”, “pré-diabetes” ou “diabetes”.

Neste exemplo, há a necessidade de se criar duas variáveis indicadoras conforme explicado a seguir: se a resposta ao exame for “normal”, as duas variáveis indicadoras, D_1 e D_2 podem receber a seguinte codificação: 0 e 0, respectivamente. Se a resposta for “pré-diabetes” podem receber a seguinte codificação 1 e 0, respectivamente. E se a resposta for “diabetes” podem receber a seguinte codificação: 0 e 1, respectivamente. A Tabela 2 resume essa codificação.

Tabela 1: Exemplo de codificação de uma variável com três categorias

	Variáveis indicadoras	
Resposta ao exame	D_1	D_2
Normal	0	0
Pré-diabetes	1	0
Diabetes	0	1

Hosmer, Lemeshow e Sturdivant (2013) mencionam que “se uma variável de escala nominal tem k possíveis valores, então $k-1$ variáveis indicadoras são necessárias”.

Para obter as estimativas para o vetor $\boldsymbol{\beta}' = (\beta_0, \beta_1, \dots, \beta_p)$ dos parâmetros do modelo, suponhamos que temos uma amostra de n observações independentes $(\boldsymbol{X}_i, y_i),\ i = 1,2, \dots, n$. O método de estimação utilizado será o de máxima verossimilhança. Haverá $(p+1)$ equações de verossimilhança obtidas derivando-se a função log da verossimilhança com respeito a $(p+1)$ coeficientes. As equações de verossimilhança resultantes podem ser expressas como:

$$\sum_{i=1}^{n} [y_i - \pi(\boldsymbol{X}_i)] = 0 \quad (3.5.1)$$

e

$$\sum_{i=1}^{n} x_{ij}[y_i - \pi(\boldsymbol{X}_i)] = 0 \quad (3.5.2)$$

para $j = 1,2, \dots, p$.

Os métodos numéricos são utilizados para encontrar as soluções das equações (3.5.1) e (3.5.2) e encontram-se implementados em *softwares* direcionados à análise de dados.

Seja $\widehat{\boldsymbol{\beta}}' = (\hat{\beta}_1, \hat{\beta}_{2,}, \dots, \hat{\beta}_p)$ o vetor solução (estimado) dos parâmetros das equações (3.5.1) e (3.5.2). Se as suposições do modelo são satisfeitas, então pode-se mostrar que, assintoticamente, $E(\widehat{\boldsymbol{\beta}}) = \boldsymbol{\beta}$ e $Var(\widehat{\boldsymbol{\beta}}) = (\boldsymbol{X'VX})^{-1}$ (em que a matriz $\boldsymbol{V}$ é uma matriz diagonal $n \times n$ contendo a variância estimada de cada observação na diagonal principal) e $\boldsymbol{X}$ é uma matriz $n \times (p + 1)$, ou seja (GIOLO, 2017):

$$\boldsymbol{X} = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{12} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{bmatrix}.$$

O valor estimado de $\boldsymbol{X}_i'\boldsymbol{\beta}$ é $\boldsymbol{X}_i'\widehat{\boldsymbol{\beta}}$ e o valor do modelo de regressão logística ajustado é escrito como:

$$\hat{\pi}(\boldsymbol{X}_i) = \frac{exp(\boldsymbol{X}_i'\widehat{\boldsymbol{\beta}})}{1 + exp(\boldsymbol{X}_i'\widehat{\boldsymbol{\beta}})}$$

Segundo Agresti (2013), o valor predito será $\hat{y} = 1$ quando $\hat{\pi}(\boldsymbol{X}_i) > c$ e $\hat{y} = 0$ quando $\hat{\pi}(\boldsymbol{X}_i) \leq c$, para algum ponto de corte c e que usualmente considera-se $c = 0{,}5$.

Os Modelos de RL possuem as seguintes vantagens:

a) “Computacionalmente barato e fácil de implementar” (HARRINGTON, 2012);

b) “São modelos fáceis de interpretar” (HAIR et al, 2009; HARRINGTON, 2012);

c) Trabalham com covariáveis numéricas e nominais (HARRINGTON, 2012).

E as seguintes desvantagens:

a) Podem ter baixa acurácia quando comparados com outros modelos de classificação (HARRINGTON, 2012);

b) “Não são flexíveis o suficiente para capturar naturalmente relacionamentos mais complexos” (BEERAVALLI, 2018);

c) “Bastante sensível à uma forte colinearidade entre os atributos” (BITTENCOURT, CLARK, 2001).

3.2) MODELOS NÃO PROBABILÍSTICOS DE CLASSIFICAÇÃO

3.2.1) Árvore de Classificação

Os algoritmos de Árvores de Classificação (AC) estão atualmente entre os mais populares e mais utilizados devido ao fato de serem bastante intuitivos para o usuário comum (BURGER, 2018; FACELI et al, 2017). Alguns exemplos de aplicação das AC são: bancos de dados, jogos, sites da internet, e no reconhecimento de poses em dispositivos de detecção em videogames (FLACH, 2012).

As AC são fluxogramas, conforme ilustrado na Figura 1, formados por figuras ovais, retangulares e setas. As figuras ovais são chamadas de nós de decisão (de onde sairá alguma decisão baseada numa regra), os retângulos são chamados de nós-folha ou nó-terminal (em que alguma conclusão foi alcançada) e as setas que saem dos nós são os ramos da árvore; eles podem levar a outros nós de decisão ou nós-folha. (ALPAYDIN, 2010; HARRINGTON, 2012). O primeiro nó oval (de cima para baixo) é também conhecido como nó-raiz (SCHWARTZ, 2014).

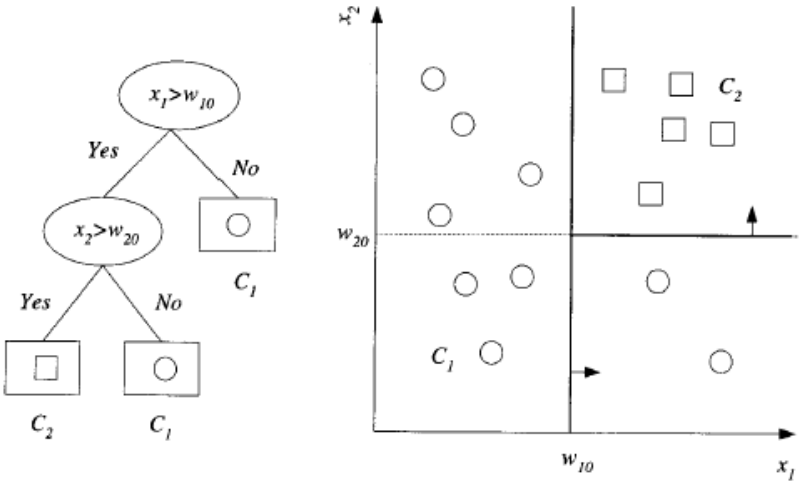


Figura 1: Representação de uma árvore de classificação (à esquerda) e as regiões de decisão no espaço de atributos (à direita). Fonte: Adaptado de Alpaydin (2010).

De maneira intuitiva, o algoritmo de AC funciona da seguinte maneira:

1) Escolhe-se a variável explicativa mais adequada dentre as disponíveis no conjunto de dados. Para determinar isso, realizam-se cálculos de entropia e de ganho de informação para cada covariável e verifica-se qual divisão apresenta os melhores resultados.

2) Com base na variável explicativa escolhida, divide-se o conjunto de dados em subconjuntos através de uma regra de decisão (Figura 1). Os subconjuntos então percorrerão as ramificações do primeiro nó de decisão. Se os dados nos ramos forem da mesma classe, ocorre a classificação e o algoritmo para (dizemos neste caso que o nó é puro). Caso os dados não sejam da mesma classe, repete-se o processo de divisão neste subconjunto.

"A decisão sobre como dividir este subconjunto é feita da mesma forma que o conjunto de dados original e você repete esse processo até classificar todos os dados" (HARRINGTON, 2012).

Algumas perguntas surgem quando estamos construindo uma AC: Por onde começar? Qual árvore é a melhor ou ideal? Qual variável explicativa devemos escolher para fazer parte do nó-raiz?

Para responder a estas perguntas faz-se necessário definirmos algumas medidas utilizadas na construção de uma AC tais como: conceito de informação,

entropia do sistema (ou simplesmente entropia), entropia de cada atributo e ganho de informação.

Informação: ao classificar algo que pode receber dois valores (classificação binária), a informação para a classe $C = c_i$ é definida como:

$$l(C = c_i) = -\log_2 p(C = c_i)$$

em que $p(C = c_i)$ é a probabilidade de escolher a classe $c_i = 1{,}2$ (HARRINGTON, 2012; KUBAT,2015).

Entropia do sistema: é uma medida da heterogeneidade (ou falta de regularidade) de um conjunto de dados. Mede o quanto um conjunto de dados está organizado ou não organizado (KELLEHER, NAMEE, D'ARCY, 2015).

"Para calcular a entropia, é necessário saber o valor de todas as informações de todos os possíveis valores de cada classe" (HARRINGTON, 2012).

A expressão da entropia é dada por (FLACH, 2012):

$$H(T) = -p(C = c_i)\log_2 p(C = c_i) - (1 - p(C = c_i))\log_2(1 - p(C = c_i))$$

em que $p(C = c_i) \in [0{,}1]$ e T representa o conjunto de dados de treinamento.

O conjunto de dados é particionado em subconjuntos através dos resultados de cada atributo.

Entropia de cada atributo: Para os subconjuntos formados das sucessivas divisões no conjunto de dados de treinamento tem-se novos valores de entropia, que denotaremos por T_i (i representa a i-ésima divisão).

A entropia de um determinado atributo at, denotada por $H_{at}(T)$, é calculada como a média ponderada das entropias dos subconjuntos da seguinte forma (KUBAT, 2015):

$$H_{at}(T) = \sum_i P_i \,.\, H(T_i),$$

em que $P_i = \frac{|T_i|}{|T|}$ é a probabilidade de, um exemplo de treinamento, estar em T, $|T_i|$ o número de exemplos em T_i e $|T|$ o número de exemplos em todo o conjunto de dados de treinamento.

Ganho de informação: O ganho de informação quando se leva em consideração um determinado atributo at é dado por:

$$I_{at}(T) = H(T) - H_{at}(T). \quad (3.3.1)$$

O ganho de informação é uma medida da relevância de um atributo em uma escala de 0 como sendo menos útil até 1 como mais útil (BURGER, 2018).

É possível provar que $I_{at}(T)$ não pode ser negativo e que informações só podem ser obtidas, nunca perdidas por considerar um determinado atributo (KUBAT, 2015).

Aplicando a equação (3.3.1) para cada atributo, podemos descobrir qual deles fornece a quantidade máxima de informações e este será escolhido para ser o nó-raiz na construção da AC.

A seguir, segue um exemplo hipotético para ilustrar os cálculos de informação, entropia do sistema, entropia de cada atributo e ganho de informação.

Considere os seguintes dados hipotéticos expostos na Tabela 1 (KUBAT, 2015) referente a preferência de pedaços de tortas em que tamanho da borda, formato e tamanho do recheio são os atributos e a classe (Y) é a variável resposta (positiva se a pessoa gostou e negativa, caso contrário).

Tabela 2: Conjunto de dados hipotético para exemplificar os cálculos envolvidos na construção de AC.

Obs.	Tamanho da borda	Formato	Tamanho do recheio	Classe (Y)
1	grande	círculo	pequeno	positiva
2	pequena	círculo	pequeno	positiva
3	grande	quadrado	pequeno	negativa
4	grande	triângulo	pequeno	negativa
5	grande	quadrado	grande	positiva
6	pequena	quadrado	pequeno	negativa
7	pequena	quadrado	grande	positiva
8	grande	círculo	grande	positiva

Fonte: Kubat (2015)

As divisões do conjunto de dados obedecerão aos possíveis resultados de um atributo. Por exemplo, o atributo formato possui os resultados: quadrado, círculo e triângulo. Assim, o conjunto de dados será dividido conforme essas categorias, formando os subconjuntos T_1, T_2 e T_3 respectivamente.

Primariamente, calcula-se a entropia do sistema (em que $C = c_1$ é a classe positiva e $C = c_2$ é a classe negativa):

$$H(T) = -p(C = c_1)log_2 p(C = c_1) - p(C = c_2)log_2 p(C = c_2)$$

$$= -(5/8)log_2(5/8) - (3/8)log_2(3/8) = 0{,}954$$

Em seguida, calcula-se a entropia das divisões definida pelos valores do atributo formato:

$$H_{formato}(T_1) = -(2/4)log_2(2/4) - (2/4)log_2(2/4) = 1$$

$$H_{formato}(T_2) = -(3/3)log_2(3/3) - (0/3)log_2(0/3) = 0$$

$$H_{formato}(T_3) = -(0/1)log_2(0/1) - (1/1)log_2(1/1) = 0$$

A partir destes cálculos obtêm-se a entropia de cada atributo em que os rótulos das classes e o valor do atributo formato são conhecidos:

$$H_{formato}(T) = \sum_i P_i H(T_i) = (4/8) * 1 + (3/8) * 0 + (1/8) * 0 = 0{,}5$$

Repetindo o mesmo procedimento para os atributos borda e recheio, têm-se os seguintes resultados:

$$H_{borda}(T) = 0{,}951$$

$$H_{recheio}(T) = 0{,}607$$

Estes valores nos dão os seguintes valores de ganhos da informação:

$$I_{formato}(T) = H(T) - H_{formato}(T) = 0{,}954 - 0{,}5 = 0{,}454$$

$$I_{borda}(T) = H(T) - H_{borda}(T) = 0{,}954 - 0{,}951 = 0{,}003$$

$$I_{recheio}(T) = H(T) - H_{recheio}(T) = 0{,}954 - 0{,}607 = 0{,}347$$

Assim, concluímos que o atributo formato é o que mais traz informação e, portanto, o que será usado no nó-raiz da AC para o conjunto de dados da Tabela 1.

A teoria discorrida anteriormente considera os atributos como sendo discretos. Para atributos contínuos, veja Kubat (2015), Kelleher, Namee e D'Arcy (2015) e Faceli et al (2011).

Na escolha entre árvores maiores (com muitos ramos e folhas) e árvores menores é aconselhado escolher as menores pelas diversas vantagens que elas têm em relação às maiores: fácil interpretação, remoção de atributos irrelevantes e redundantes e redução de risco de *overfitting* (superajustamento aos dados de treinamento). O conceito de poda (*pruning*) nos ajuda na escolha de qual árvore devemos escolher como classificador.

"A poda consiste em substituir uma ou mais subárvores por folhas, cada uma rotulada pela classe mais comum entre os exemplos de treinamento". (KUBAT, 2015).

Por exemplo, a Figura 2(a) mostra uma árvore formada por todos os atributos $t_1, \ldots, t_6$ de um determinado conjunto de dados de treinamento. A Figura 2(b) mostra uma árvore podada e esta utilizou somente os atributos t_1, t_2 e t_5. A árvore da Figura 2(a) foi podada nas subárvores com nós-raiz em t_3 e t_6 sendo substituídas pelos nós-folha rotulados como $(-)$ e $(+)$ respectivamente (Figura 2b).

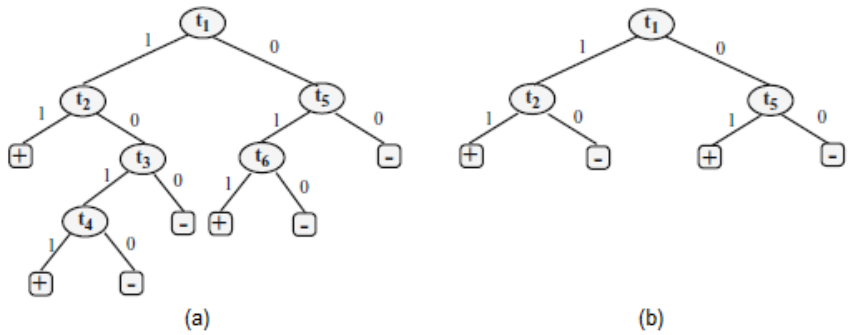


Figura 2: Árvores de classificação: Árvore utilizando todos os atributos (a) e árvore podada (b). Fonte: Kubat (2015).

Para mais detalhes sobre o funcionamento dos algoritmos de poda em AC veja Kubat (2015), Schwartz (2014) e Faceli et al (2017).

As AC possuem as seguintes vantagens:

a) O classificador é muito simples de entender, fácil de interpretar e de visualizar. (SCHWARTZ, 2014; MOHRI, ROSTAMIZADEH, TALWALKAR, 2018; BURGER, 2018; FLACH, 2012);

b) São flexíveis, pois não assumem nenhuma distribuição para os dados (FACELI et al, 2017);

c) “O processo de construção de AC seleciona os atributos a usar no modelo de decisão. Essa seleção de atributos produz modelos que tendem a ser bastante robustos contra a adição de atributos irrelevantes e redundantes”. (FACELI et al, 2017; KUBAT, 2015);

d) Não requer recursos computacionais caros (HARRINGTON, 2012) e são rápidos computacionalmente (BURGER, 2018);

E as seguintes desvantagens:

a) “Pode criar árvores supercomplexas dos dados de treinamento que não classificam bem os dados de teste” (BEERAVALLI, 2018);

b) Podem ser tendenciosas na classificação caso o número de observações em cada classe for desbalanceado (BEERAVALLI, 2018);

c) “Podem ser muito sensíveis a condições iniciais ou variações nos dados, podendo resultar na geração de uma árvore completamente diferente” (BURGER, 2018; BEERAVALLI, 2018);

d) “Apresentam dificuldades na presença de atributos contínuos” (FACELI et al, 2017).

3.2.2) Random Forest

O algoritmo de *Random Forest* (Floresta Aleatória), RF, são uma coleção de diferentes Árvores de Decisão (BURGER, 2018; SCHWARTZ, 2014). Elas são um exemplo de conjunto de modelos (*ensembles models*), isto é, um modelo que é formado por um conjunto de modelos mais simples.

As RF têm inúmeras aplicações; desde previsibilidade de inadimplência (classificação de possíveis clientes de uma instituição financeira como adimplentes ou inadimplentes) até na identificação de doenças, através da análise de registros médicos de pacientes (BEERAVALLI, 2018).

O termo *random* (aleatório) se refere a amostra aleatória que se faz nos atributos necessários para a construção de cada árvore de decisão do conjunto de árvores geradas. A esse conjunto dá-se o nome de Floresta (*Forest*).

“Cada árvore é aprendida usando uma amostra *bootstrap* do conjunto de dados de treinamento. Com cada uma das amostras geradas, uma árvore é obtida” (TORGO, 2015).

A Figura 3 ilustra o funcionamento do algoritmo de RF.

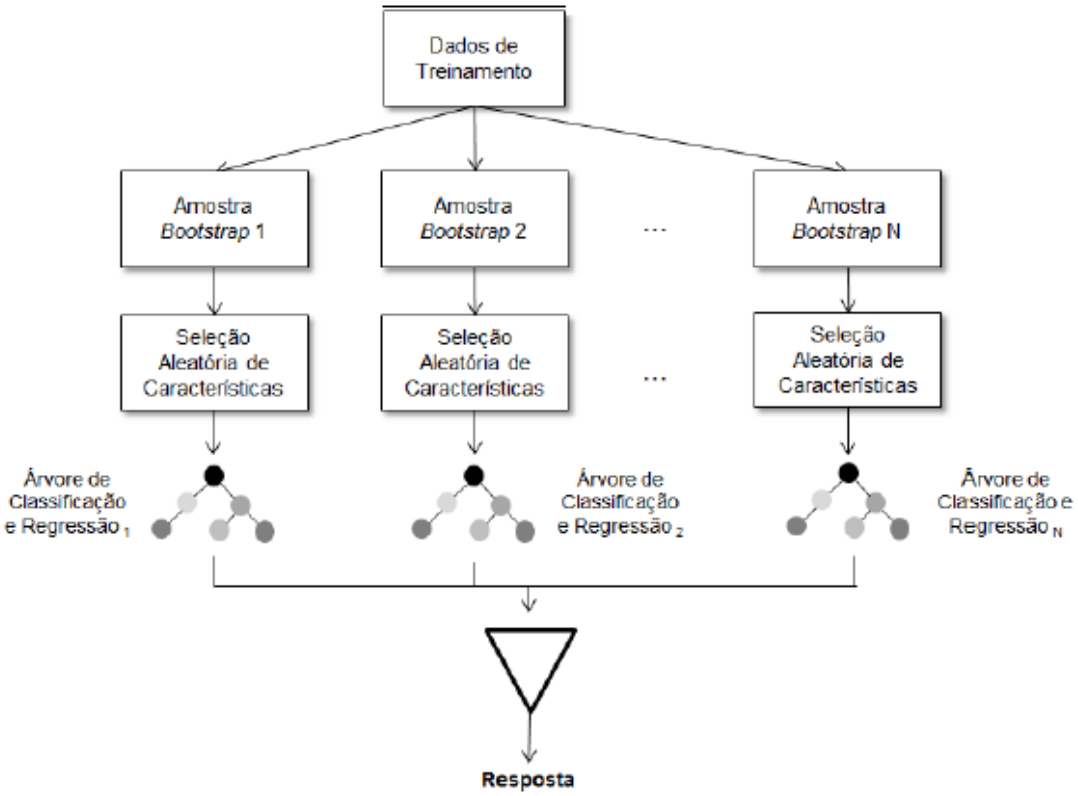


Figura 3: Esquema ilustrativo do algoritmo RF com amostras *bootstrap*. Fonte: Borges Jr. (2016)

Para cada amostra *bootstrap* gerada, ocorre uma amostragem aleatória de características (variáveis explicativas ou atributos) que serão utilizadas na criação de cada árvore (Figura 3). Portanto, diferentes árvores terão diferentes atributos na geração dos nós de decisão.

A criação de cada árvore se dá de maneira análoga ao algoritmo de AC incluindo todos os cálculos de entropia e ganho de informação envolvidos.

O processo de classificação de novas observações se dá da seguinte forma: cada árvore da RF classifica as observações do conjunto de teste em uma ou outra categoria, ou seja, caso a RF tenha N árvores, uma dada observação terá N votos (Figura 3). A variável resposta que receber mais votos em todas as árvores é a previsão do conjunto (TORGO, 2015).

As RF possuem as seguintes vantagens:

a) Elas são uma alternativa para reduzir o perigo de ajustamento demasiado aos dados de treinamento quando se utiliza uma única AC na tomada de decisão (BURGER, 2018; REBALA, RAVI, CHURIWALA, 2019; SCHWARTZ, 2014);

b) “São capazes de trabalhar com um número muito grande de covariáveis”. (ROKACH, MAIMON, 2015);

c) “Possuem boa acurácia e são relativamente robustas em relação a valores discrepantes” (BREIMAN, 2001 apud BORGES. Jr, 2016);

d) “As RF combinam a simplicidade das AC com flexibilidade, resultando em grande melhoria na acurácia” (TORGO, 2015).

E as seguintes desvantagens:

a) “Podem tornar o algoritmo lento e ineficaz para previsões em tempo real” (BEERAVALLI, 2018);

b) “O aumento da acurácia (das RF comparadas com as AC) vem em um custo de alguma perda de ajuste interpretável dos dados”. (REBALA, RAVI, CHURIWALA, 2019);

c) “Complexo, difícil de implementar e computacionalmente custoso” (BEERAVALLI, 2018);

d) Quando muitas variáveis forem correlacionadas e se estas têm efeito significativo nos resultados, as RF terão dificuldades em identificar a importância relativa dessas variáveis (REBALA, RAVI, CHURIWALA, 2019);

e) “É difícil de visualizar e de entender o porquê o algoritmo previu algo” (BEERAVALLI, 2018).

3.2.3) Algoritmo $k - NN$

O algoritmo $k - NN$ (k −*Nearest Neighbor* ou k −Vizinhos mais próximos) é um dos mais antigos, mais utilizados e mais simples algoritmos de classificação. Sua aplicabilidade está nas mais diversas áreas tais como: finanças, saúde, ciência política, reconhecimento de imagens e de vídeos (LUZ, 2019).

Seja R^p o espaço p-dimensional de um conjunto de dados com p atributos (ou p variáveis explicativas), ou seja, R^p é o espaço formado por todos os atributos de um conjunto de dados de treinamento. Estes atributos podem ser de escala nominal, ordinal, intervalar ou de razão.

Sabe-se que cada observação deste conjunto de dados tem uma variável resposta (classe) que é rotulada como: “0” se ocorre fracasso e “1” se ocorre sucesso. Considere $\boldsymbol{X} = (X_1, X_2, \dots, X_p)$ e $\boldsymbol{Y} = (Y_1, Y_2, \dots, Y_p)$ ambos pontos de R^p.

O algoritmo $k - NN$ tem por hipótese: dados similares tendem a estar mais próximos e dados não similares tendem a estar mais distantes entre si.

Por “mais próximos” e “mais distantes”, entende-se que há a necessidade de se definir uma medida (métrica) que calcule a distância entre os pontos de R^p.

Há muitas maneiras diferentes de se definir a distância entre dois pontos em R^p. A mais intuitiva e frequentemente utilizada é a distância Euclidiana (quando se tem somente atributos contínuos) definida como (TORGO, 2015; BURGER, 2018):

$$d_E(\boldsymbol{X}, \boldsymbol{Y}) = \sqrt{\sum_{i=1}^{p} (X_i - Y_i)^2},$$

Quando se têm atributos contínuos e discretos, a distância mais adequada é definida como (KUBAT, 2015):

$$d_M(\boldsymbol{X}, \boldsymbol{Y}) = \sqrt{\sum_{i=1}^{p} d(X_i, Y_i)},$$

em que $d(X_i, Y_i) = (X_i - Y_i)^2$ para atributos contínuos e, para atributos discretos, $d(X_i, Y_i) = 0$ se $X_i = Y_i$ e $d(X_i, Y_i) = 1$ se $X_i \neq Y_i$. O índice M indica a mistura de atributos discretos e contínuos no cálculo da distância.

Para classificar uma observação do conjunto de teste, o algoritmo calcula a distância desta observação a todas as observações do conjunto de treinamento. Esta observação receberá o rótulo da observação mais próxima a ela (FACELI et al, 2017). Neste caso o algoritmo é $1 - NN$.

Muitas vezes a taxa de erro de classificação utilizando o algoritmo $1 - NN$ é alta (KUBAT, 2015). Por isso se considera k vizinhos mais próximos (k inteiro maior que 1), se calcula a distância de todas as observações do conjunto de treinamento à esta observação do conjunto de teste, e se verifica as k menores distâncias. Se observa quais vizinhos estão associados às menores distâncias e então se verifica a classe de cada um (KELLEHER, NAMEE, D'ARCY, 2015). Cada vizinho vota em uma classe. A observação é classificada na classe mais votada (FACELI et al, 2017; ALPAYDIN, 2010).

"A escolha do valor de k pode não ser trivial. O valor de k é definido pelo usuário. Frequentemente, o valor de k é pequeno e ímpar: $k = 3,5, \ldots$. Em problemas de classificação, não é usual utilizar k sendo par, para evitar empates" (FACELI et al, 2017).

O algoritmo $k - NN$ possui alguns aspectos positivos e negativos. Os aspectos positivos são:

a) O algoritmo de treinamento é simples (FACELI et al, 2017; BEERAVALLI, 2018), aplicável mesmo em problemas complexos (FACELI et al, 2017) e fácil de interpretar (KELLEHER, NAMEE, D'ARCY, 2015);

b) Quando novos exemplos de treinamento estão disponíveis basta armazená-los na memória (FACELI et al, 2017). Desta forma, o $k - NN$ pode ser facilmente

atualizado, o que o torna relativamente robusto (KELLEHER, NAMEE, D'ARCY, 2015);

c) "É robusto a dados que possuem valores muito diferentes do esperado ou que possuem erros de medida quando k é definido apropriadamente" (KELLEHER, NAMEE, D'ARCY, 2015);

d) São eficientes ao lidar com diferentes tipos de covariáveis explicativas discretas, contínuas ou binárias (KELLEHER, NAMEE, D'ARCY, 2015);

Os aspectos negativos do $k - NN$ são:

a) Predição pode ser custosa (KELLEHER, NAMEE, D'ARCY, 2015) devido ao fato de, para conjuntos de treinamento grandes, esse processo pode ser demorado (FACELI et al, 2017);

b) "O desempenho do algoritmo é afetado pela presença de atributos redundantes (quando duas ou mais observações têm os mesmos valores para todas as covariáveis explicativas ou duas ou mais covariáveis explicativas têm os mesmos valores para duas ou mais observações) e atributos irrelevantes" (FACELI et al, 2017);

c) O desempenho do algoritmo tende a diminuir com dados esparsos (KUBAT, 2015);

d) É intensivo em memória e apresenta desempenho ruim para dados de alta dimensão (BEERAVALLI, 2018; KUBAT, 2015).

O algoritmo $k - NN$, por se basear no cálculo de distâncias entre observações, muitas vezes é afetado por variáveis explicativas que são medidas em escalas com amplitudes muito diferentes.

Por exemplo, considere que um conjunto de dados tenha as seguintes variáveis explicativas: Salário Anual e Idade. O menor valor de Salário Anual é \$45000 e o maior valor é \$75000. Assim, temos uma amplitude nos valores observados de \$30000. Por outro lado, o menor valor observado da Idade é 25 anos e o maior valor é 60 anos. Logo, a amplitude é de 35 anos (KELLEHER, NAMEE, D'ARCY, 2015).

Ao se aplicar o algoritmo $k - NN$, verifica-se que a variável Salário Anual contribui mais para o cálculo da distância do que a variável Idade.

Para contornar esse problema, faz-se necessário normalizar as escalas das variáveis explicativas conforme se segue (KUBAT, 2015; BURGER, 2018):

$$X_{norm} = \frac{X - min(X)}{max - min}$$

“A normalização reduz a taxa de erro em muitas aplicações práticas, especialmente se as escalas dos atributos originais variarem significativamente” (KUBAT, 2015).

3.2.4) Redes Neurais Artificiais

As Redes Neurais Artificiais (RNA) têm sua fundamentação, conforme o próprio nome diz, em como as redes biológicas presentes no sistema nervoso funcionam, em especial o cérebro humano (ALPAYDIN, 2010; REBALA, 2019; SHWARTZ, 2014) sendo sua unidade básica de processamento o neurônio (KUBAT, 2015; MONTGOMERY, PECK, VINING, 2006).

Segundo Braga, Carvalho e Ludermir (2011) as RNA são sistemas paralelos distribuídos compostos por unidades de processamento simples (nós ou neurônios artificiais) que computam determinadas funções matemáticas frequentemente não lineares.

São diversas as aplicações das RNA, por exemplo, em processamento de imagens, reconhecimento de caracteres usados na detecção de fraudes (fraude bancária), previsões em decisões comerciais (BEERAVALLI, 2018), previsões financeiras (TORGO, 2015), ecologia, meio ambiente (FACELI et al, 2017), indústria (BURGER, 2018), meteorologia (COUTINHO, SILVA, DELGADO, 2016) dentre outras.

A seguir será exposto alguns conceitos básicos para o melhor entendimento das RNA e o seu equivalente nas redes neurais biológicas.

O neurônio artificial possui terminais de entrada $(x_1, x_2, \dots, x_n)$ que têm a função análoga ao dos dendritos do neurônio biológico. Os dendritos são estruturas que recebem estímulos nervosos do ambiente (ou de outros neurônios) e transmitem esses estímulos para o corpo do neurônio.

Possui um somador central mais uma função de ativação que podem ser comparados ao corpo celular do neurônio biológico em que o estímulo vindo dos dendritos é processado). A Figura 4 mostra a correspondência entre um neurônio biológico simplificado (à esquerda) e um neurônio artificial (à direita).

O neurônio artificial também possui um terminal de saída (y) semelhante ao axônio (do neurônio biológico) que se trata de uma estrutura que transmite um impulso elétrico que parte do corpo neuronal.

Assim, em comparação com as outras técnicas descritas, $(x_1, x_2, \dots, x_n)$ são as variáveis explicativas que serão processadas através de funções matemáticas e que nos ajudarão a explicar (classificar) a variável resposta y da RNA.

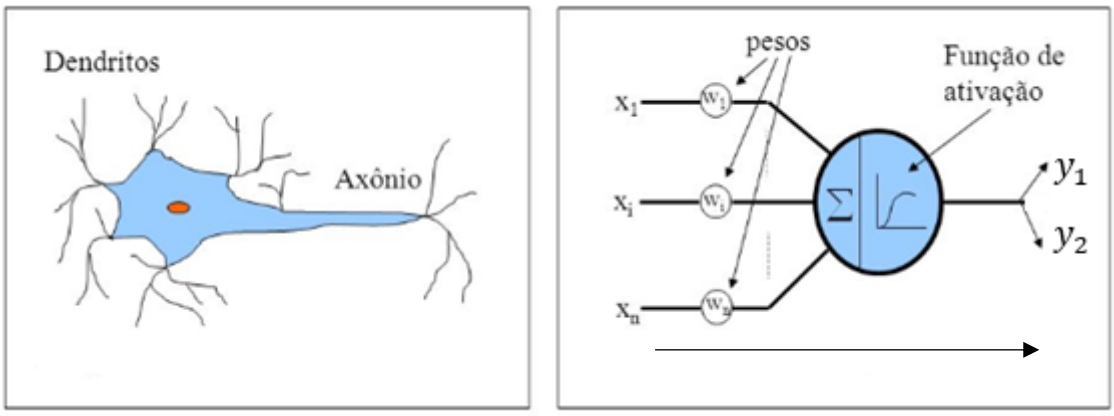


Figura 4: Neurônio biológico simplificado (à esquerda) e neurônio artificial (à direita).
Fonte:Gudwin (2019).

Além disso, no neurônio artificial, encontram-se também as seguintes características:

a) os terminais de entrada do neurônio têm pesos $(w_1, w_2, \dots, w_n)$ que "podem ser positivos ou negativos dependendo de o comportamento ser excitatório ou inibitório, respectivamente" (FACELI et al, 2017);

b) uma soma ponderada dos diferentes produtos $x_i w_i$ dada por $s = \sum_{i=1}^{n} x_i w_i$;

c) uma função de ativação que é aplicada à soma ponderada s e que, dependendo do seu valor, ativa ou não a saída. A saída da função de ativação é o resultado da RNA.

Existem diversas funções de ativação que podem ser utilizadas ao treinar RNA. Algumas delas são:

a) Linear:

$$y = f(s) = a.s$$

em que a é um número real (inclinação da reta).

b) Limiar:

Dois tipos são frequentemente utilizados:

b.1)

$$y = f(s) = \begin{cases} 1 \; se \; s \geq 0 \\ 0 \; se \; s < 0 \end{cases}$$

ou

b.2)

$$y = f(s) = \begin{cases} 1 \; se \; s \geq 0 \\ -1 \; se \; s < 0 \end{cases}$$

c) Sigmóide:

$$y = f(s) = \frac{1}{1 + e^{-a.s}}$$

O parâmetro a da função sigmóide modifica a inclinação da função. Quando o valor do parâmetro tende a infinito, a função sigmóide se torna uma função limiar (b.1).

A função de ativação sigmóide é a mais frequentemente utilizada porque possui a característica de ser contínua, diferenciável e não decrescente (BURGER, 2018; KUBAT, 2015; KULKARNI, 2011).

Sobre a arquitetura das redes, a Figura 5 mostra um exemplo de uma rede neural artificial e sua estrutura:

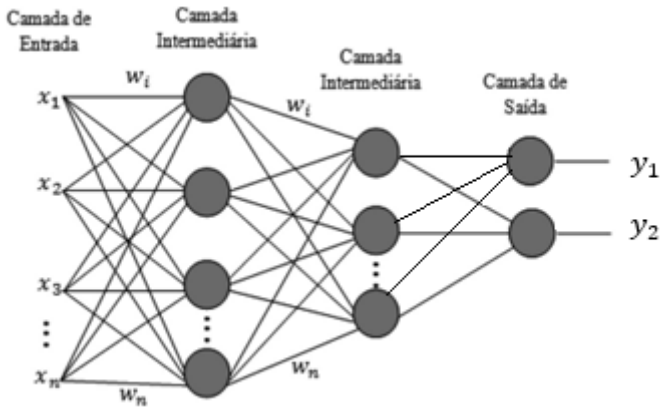


Figura 5: Exemplo de uma estrutura de rede neural artificial. Fonte: Coutinho, Silva, Delgado (2016).

Seus neurônios estão organizados em camadas. A primeira camada contém os neurônios de entrada da rede. As observações de treinamento são apresentadas à rede por meio dessas entradas. A camada final (ou de saída) contém as previsões da RNA para qualquer caso apresentado em seus neurônios de entrada. No meio, geralmente temos uma ou mais camadas denominadas ocultas, intermediárias ou escondidas (BURGER, 2018).

Existem diversas estruturas de RNA. As utilizadas neste trabalho são as do tipo “alimentadas para frente” (*feedforward*) (KULKARNI, 2011; SHWARTZ, 2014), indicada pelo sentido da seta na Figura 4 à direita. Além disso, qualquer neurônio de uma camada comunica-se com todos os neurônios da camada seguinte, mas neurônios da mesma camada não se comunicam, ou seja, a comunicação é unidirecional; da camada de entrada para a camada de saída.

Quando a rede tem somente a camada de entrada $(x_1, x_2, \dots, x_n)$ e a camada de saída y (Figura 4 à direita) esta é denominada *perceptron* de uma camada. Caso a RNA tenha camadas de neurônios ocultas, esta é denominada *perceptron* multicamada (*multilayer perceptron*, MLP), como mostra a Figura 5.

As redes *perceptron* de uma camada resolvem apenas problemas linearmente separáveis. As observações de duas classes são linearmente separáveis se houver um hiperplano que separa os dados em duas classes (FACELI et al, 2017). Porém, inúmeros problemas de classificação são não linearmente separáveis. Neste caso, deve-se utilizar as redes neurais artificiais MLP.

A utilização de camadas ocultas pode ajudar a aumentar a precisão da RNA. Em contrapartida, mais camadas aumenta o risco de ajustar demais o modelo, provocando assim *overfitting*, ou seja, superajustamento dos dados de treinamento (BURGER, 2018).

Construir uma RNA consiste em estabelecer (escolher) uma arquitetura para a rede. Isso envolve escolher o número de camadas ocultas, número de neurônios em cada camada e qual função de ativação utilizar. Em seguida usar um algoritmo para encontrar os pesos das conexões entre os neurônios (TORGO, 2015; MONTGOMERY, PECK, VINING, 2006).

Portanto, a tarefa de se construir uma RNA é encontrar a maneira mais adequada de ajustar os pesos para que se tenha um bom desempenho nos dados de treinamento (KULKARNI, 2011).

Este trabalho se utiliza das RNA *perceptron* multicamadas (MLP) para o seu treinamento. Existem diversos algoritmos de treinamento das redes neurais MLP. O mais simples e frequentemente utilizado é a retropropagação do erro (*backpropagation*).

Nesse algoritmo, a determinação do sinal de erro δ é um processo recursivo que se inicia nos neurônios da camada de saída e vai até os neurônios da primeira camada intermediária.

O algoritmo *backpropagation* é constituído de duas fases:

1) “Para frente” (*forward*): um padrão é apresentado à camada de entrada da rede e propagado em direção à camada de saída. A saída obtida é comparada com a saída desejada para esse padrão particular. Se esta não estiver correta, o erro é calculado;

2) “Para trás” (*backward*): o erro é definido como sendo a diferença entre os valores de saída produzidos e desejados para cada neurônio da camada de saída, e é propagado a partir da camada de saída até a camada de entrada. Os pesos das conexões dos neurônios das camadas ocultas vão sendo modificados conforme o erro é retropropagado.

A equação (3.6.1) mostra como é feito o ajuste dos pesos de uma rede MLP pelo algoritmo *backpropagation*:

$$w_{jl}^{(t+1)} = w_{jl}^{t}(t) + \eta x^{j} \delta_{l} \quad (3.6.1)$$

em que:

- $w_{jl}^{(t)}(t)$ representa o peso entre um neurônio l e o j-ésimo atributo de entrada ou a saída do j-ésimo neurônio da camada anterior;
- δ_{l} indica o erro associado ao l-ésimo neurônio;
- x^{j} indica a entrada recebida por esse neurônio (o j-ésimo atributo de entrada ou a saída do j-ésimo neurônio da camada anterior);
- η é o valor da taxa de aprendizado da rede e;
- t é a interação.

Para os neurônios das camadas intermediárias, em que não existem saídas desejadas, o sinal do erro δ é determinado recursivamente em termos dos sinais dos erros dos j neurônios diretamente conectados a eles e dos pesos destas conexões. Para mais detalhes sobre a estimação dos erros dos neurônios das camadas ocultas, veja Faceli et al (2017).

“Os ciclos de apresentação dos dados de treinamento e eventuais ajustes de pesos no *backpropagation* são iterados até que seja atingido um critério de parada. Diferentes critérios de parada podem ser utilizados, como, por exemplo, um número máximo de ciclos ou uma taxa máxima de erro” (FACELI et al, 2017).

“O classificador escolhe a classe cujo neurônio de saída retornou o valor mais alto” (KUBAT, 2015).

As Redes Neurais Artificiais MLP possui as seguintes vantagens:

a) “São consideravelmente tolerantes a ruídos” (FACELI et al, 2017);

b) “São capazes de aprender com a experiência e de generalizar, não existe a necessidade de programar cada porção explícita de informação” (GARDNER, KORNHABER, WAKE, 1998)

c) “Apresentam bom desempenho para dados complexos, de alta dimensão e não lineares” (MONTGOMERY, PECK, VINING, 2006).

As Redes Neurais Artificiais MLP possui as seguintes desvantagens:

a) Dificuldade do usuário em entender como e porquê a RNA apresentou determinada saída (BURGER, 2018; FACELI et al, 2017);

b) “Tempo de treinamento é muito alto; requer alto poder computacional” (BEERAVALLI, 2019);

c) “Computacionalmente caro” (KUBAT, 2015) e “de difícil implementação” (SHWARTZ, 2014).

4) MEDIDAS DE DESEMPENHO DOS MODELOS

Para medir o desempenho dos modelos, aplica-se a amostra de teste para que se faça a classificação destes dados. Visto que os dados da amostra de teste já estão previamente classificados, os modelos irão classificar sem o conhecimento prévio de qual classe cada observação é rotulada. Terminada esta etapa, analisa-se o desempenho do modelo, comparando os valores da classe dos dados de teste com os valores preditos.

A Tabela de Classificação, também denominada Matriz de Confusão, (Tabela 3) resume os resultados do ajuste dos modelos ou algoritmos de classificação. Usualmente uma classe é denotada positiva $(+)$ como possuidora da característica estudada e a outra é denominada negativa $(-)$, caso contrário. Cada elemento da matriz de confusão possui as seguintes características (FACELI et al, 2017):

a) VP corresponde ao número de verdadeiros positivos, ou seja, o número de observações da classe positiva classificadas como positivas (classificação correta);

b) VN corresponde ao número de verdadeiros negativos, ou seja, o número de observações da classe negativa classificadas como negativas (classificação correta);

c) FP corresponde ao número de falsos positivos, ou seja, o número de observações cuja classe verdadeira é negativa mas que foram classificadas como pertencendo à classe positiva (classificação incorreta);

d) FN corresponde ao número de falsos negativos, ou seja, o número de observações cuja classe verdadeira é positiva mas que foram classificadas como pertencendo à classe negativa (classificação incorreta);

Tabela 3: Matriz de confusão para um problema de classificação binária

		Classe predita	
		$+$	$-$
Classe verdadeira	$+$	VP	FN
	$-$	FP	VN

Da matriz de confusão (Tabela 3), derivam-se medidas de desempenho. Dentre essas medidas três serão destacadas neste trabalho: a acurácia, a sensibilidade e a especificidade. Elas são definidas como (FACELI et al, 2017; GORDIS, 2010):

a) Acurácia: definida como a taxa de acerto total, ou seja, $ac = \frac{VP+VN}{n}$ (em que n é o tamanho amostral);

b) Sensibilidade (capacidade de classificar corretamente aqueles que possuem a característica estudada): definida como a taxa de acerto na classe positiva, ou seja, $sens = \frac{VP}{VP+FN}$;

c) Especificidade (capacidade de identificar corretamente aqueles que não apresentam a doença): definida como a taxa de acerto na classe negativa, ou seja, $esp = \frac{VN}{VN+FP}$.

“A Tabela 3 é apropriada quando a classificação é o objetivo geral da análise, senão deve ser usada como um suplemento do método da avaliação do ajuste de modelos de regressão logística” (HOSMER, LEMESHOW, STURDIVANT, 2013).

5) DESCRIÇÃO DOS DADOS

Os dados utilizados neste trabalho foram obtidos por Jacinto (2008) e publicados em sua tese de doutorado do Programa de Pós-Graduação em Medicina (Neurologia) da Faculdade de Medicina da Universidade de São Paulo (FMUSP).

Dos 248 pacientes avaliados, 115 eram oriundos do Ambulatório Geral e Didático (AGD) e 133 do Ambulatório de Clínica Médica Geral (ACMG). Ambos os ambulatórios estão associados ao Departamento de Clínica Médica do Hospital das Clínicas da Faculdade de Medicina da Universidade de São Paulo (HCFMUSP).

Três pacientes foram retirados do estudo, pois dois deles não aceitaram retornar para a avaliação e um mudou-se da cidade de São Paulo.

Para os 245 pacientes participantes da pesquisa foi feita uma avaliação constituída pela pergunta: “Como está a memória?”, análise do histórico clínico (diagnósticos prévios e intercorrências recentes), anotações das medicações em uso, aplicação de dois testes cognitivos: Mini Exame do Estado Mental (MEEM) e Bateria Cognitiva Breve, aplicação de um questionário para medir depressão geriátrica (Geriatric Depression Scale, GDS), questionário de atividades funcionais de Pfeffer (Pfeffer) e um instrumento com 16 questões chamado Short-IQCode (Informant Questionnaire on Cognitive Decline in the Elderly) (JACINTO, 2008).

Também foram coletadas as seguintes informações dos pacientes: Sexo (0=feminino; 1=masculino), Escolaridade (anos de estudo) e Idade (em anos). Os critérios de inclusão dos pacientes para a participação do estudo foram: ter idade igual ou superior a 65 anos (ser idoso) e possuir telefone para contato (JACINTO, 2008). Desta forma, o estudo de Jacinto (2008) se caracteriza por um estudo observacional (GORDIS, 2010).

A pergunta “Como está a memória? ” foi feita com a intenção de verificar a presença ou não de queixa subjetiva de memória. Se a resposta dada fosse “ruim”, era feita outra pergunta: “Há quanto tempo? ”. Para nosso estudo vamos

nos referir a esta pergunta como sendo a covariável “Queixa” (0=não tem queixa;1=sim tem queixa).

O MEEM é um teste que “foi desenvolvido como uma avaliação clínica prática de mudança do estado cognitivo para pacientes idosos” (GORENSTEIN, WANG, HUNGERBÜHLER, 2016). Ele “avalia sete domínios da cognição: orientação espacial, orientação temporal, registro de três palavras faladas, atenção e cálculo, evocação de três palavras, linguagem e construção visual” (JACINTO, 2008). A partir deste teste era gerado um escore de, no máximo, 30 pontos (Anexo A).

A Bateria Cognitiva Breve inclui: extensão de dígitos na ordem direta (OD) e na ordem inversa (OI), percepção visual e nomeação (Nom) de 10 itens apresentados como figuras simples, memória incidental dos 10 itens (MI), memória imediata dos 10 itens (Mim), aprendizado dos 10 itens (Aprend), teste de fluência verbal, categoria semântica “animais” (TFV), teste do desenho do relógio (TR), memória tardia (recordação dos 10 itens) denominada por MT e reconhecimento dos 10 itens dentre os 20 apresentados como figuras simples (Rec) (Anexo B).

Quanto a GDS, “é um dos instrumentos mais frequentemente utilizados para a detecção de depressão em idosos. Diversos estudos já demonstraram que a GDS oferece medidas válidas e confiáveis para a avaliação de transtornos depressivos” (ALMEIDA e ALMEIDA, 1999).

Dentre as diversas versões abreviadas da GDS, Jacinto (2008) utilizou uma que continha 15 itens “que avaliam o humor do paciente e foi aplicada por que estados depressivos podem interferir no desempenho dos indivíduos na avaliação cognitiva e funcional” (JACINTO, 2008), “mascarando assim declínios cognitivos globais (demências)” (GORENSTEIN, WANG, HUNGERBÜHLER, 2016).

“A GDS com 15 itens pode ser utilizada com relativa confiabilidade na prática clínica” (ALMEIDA e ALMEIDA, 1999). É composta pelas seguintes perguntas (Tabela 4) e, ao final de sua aplicação, era gerada uma pontuação.

Tabela 4: Perguntas que compõem a GDS-15.

Itens da GDS-15	Resposta
1) Satisfeito com sua vida?	(não=1) (sim = 0)
2) Interrompeu muitas de suas atividades?	(sim=1) (não = 0)
3) Acha sua vida vazia?	(sim=1) (não = 0)
4) Aborrece-se com frequência?	(sim=1) (não = 0)
5) Sente-se de bem com a vida na maior parte do tempo?	(não=1) (sim = 0)
6) Teme que alguma coisa ruim lhe aconteça?	(sim=1) (não = 0)
7) Sente-se alegre a maior parte do tempo?	(não=1) (sim = 0)
8)Sente-se desamparado com frequência?	(sim=1) (não = 0)
9) Prefere ficar em casa a sair e fazer coisas novas?	(sim=1) (não = 0)
10 Acha que tem mais problemas de memória que outras pessoas?	(sim=1) (não = 0)
11) Acha que é maravilhoso estar vivo agora?	(não=1) (sim = 0)
12) Vale a pena viver como vive agora?	(não=1) (sim = 0)
13) Sente-se cheio de energia?	(não=1) (sim = 0)
14) Acha que sua situação tem solução?	(não=1) (sim = 0)
15) Acha que tem muita gente em situação melhor?	(sim=1) (não = 0)

Fonte: Jacinto (2008).

O questionário de atividades funcionais de Pfeffer foi aplicado ao informante (pessoa que acompanhava o paciente na consulta, cuidador ou familiar) e abrange dez questões que avaliam atividades gerais da vida diária. As dez perguntas estão na Tabela 5.

Tabela 5: Questionário de atividades funcionais de Pfeffer.

01) Ele manuseia seu próprio dinheiro?
02) Ele é capaz de comprar roupas, comida, coisas para casa sozinho?
03) Ele é capaz de esquentar a água para o café e apagar o fogo?
04) Ele é capaz de preparar uma comida?
05) Ele é capaz de manter-se em dia com as atualidades, com os acontecimentos da comunidade ou da vizinhança?
06) Ele é capaz de prestar atenção, entender e discutir um programa de rádio ou televisão, um jornal ou uma revista?
07) Ele é capaz de lembrar-se de compromissos, acontecimentos familiares, feriados?
08) Ele é capaz de manusear seus próprios remédios?
09) Ele é capaz de passear pela vizinhança e encontrar o caminho para casa?
10) Ele pode ser deixado em casa sozinho de forma segura?

Fonte: Jacinto (2008).

As respostas para todas as perguntas do questionário Pfeffer foram codificadas como mostra a Tabela 6. Ao final da aplicação do questionário de Pfeffer também foi gerado um escore.

Tabela 6: Codificação das respostas do questionário de atividades funcionais de Pfeffer.

Codificação	Descrição da resposta
0	Normal ou Nunca o fez mas poderia fazê-lo agora
1	Faz com dificuldade ou Nunca o fez e agora teria dificuldade
2	Necessita de ajuda
3	Não é capaz

Fonte: Jacinto (2008).

O IQCode é um instrumento com 16 questões por meio do qual o informante avalia o desempenho atual do paciente em diferentes situações da vida diária, comparando com o desempenho observado há 10 anos (GORENSTEIN, WANG, HUNGERBÜHLER, 2016). As perguntas que formam o IQCode encontram-se na Tabela 7 e, ao final, era gerada uma pontuação.

Tabela 7: Perguntas que compõem o IQCode.

Comparada há 10 anos, como está a pessoa nas situações:	Muito melhor	Um pouco melhor	Não muito alterado	Um pouco pior	Muito pior
1) Lembrar-se de coisas sobre a família e amigos, por exemplo ocupações, aniversários, endereços					
2) Lembrar-se de coisas que aconteceram recentemente					
3) Lembrar-se do que conversou nos últimos dias					
4) Lembrar-se de seu endereço e telefone					
5) Lembrar-se do dia e mês correntes					
6) Lembrar-se onde as coisas são guardadas usualmente					
7) Lembrar-se onde foram guardadas coisas que foram colocadas em locais diferentes do usual					
8) Saber como os aparelhos da casa funcionam					
9) Aprender como usar novos aparelhos da casa					
10) Aprender coisas novas em geral					
11) Acompanhar uma história em um livro ou na televisão					
12) Tomar decisões em problemas do dia-a-dia					
13) Manusear dinheiro para as compras					
14) Lidar com problemas financeiros, como pensão, coisas de banco					
15) Lidar com outros problemas matemáticos do dia-a-dia, como saber quanta comida comprar, quanto tempo transcorreu entre as visitas de familiares e amigos					
16) Usar sua inteligência para entender qual o sentido das coisas					

Fonte: Jacinto (2008).

Dos 245 pacientes participantes da pesquisa, 49 foram considerados e classificados como suspeitos de possuírem alterações cognitivas (uma ou mais) e 196 como não suspeitos. Para tal classificação foram utilizados, por Jacinto (2008), os escores alterados do MEEM ou do IQCode.

Dentre os objetivos de Jacinto (2008), um deles era verificar quais instrumentos podem ser sugeridos para a utilização por médicos generalistas para o diagnóstico de declínio cognitivo. Defina-se médico generalista como “aquele que resolve os problemas mais básicos de saúde, possui um nível diagnóstico aceitável, trata a maioria das doenças e tem sapiência em encaminhar ao especialista casos mais complexos” (SATO, 2019).

6) ANÁLISE E RESULTADOS

Para a composição dos Modelos de Classificação construídos neste trabalho, a variável “Suspeitos” é a variável resposta tomando como valores: “1” se o paciente era suspeito de possuir alterações cognitivas e “0”, caso contrário (distribuição Bernoulli). Nesta situação pode supor independência entre os pacientes.

As covariáveis utilizadas na construção dos modelos de classificação, estão expostas na Tabela 8.

Tabela 8: Características das covariáveis utilizadas nos modelos de classificação

Covariável	Codificação ou intervalo de variação	Tipo da covariável
Idade	(65-94)	contínua
Sexo	0: feminino 1: masculino	binária
Escolaridade (anos de estudo)	(0-18)	contínua
Queixa	0: não 1: sim	binária
Ordem direta (OD)	(0-06)	discreta
Ordem inversa (OI)	(0-06)	discreta
Nomeação (Nom)	(0-10)	discreta
Memória incidental (MI)	(0-10)	discreta
Memória imediata (Mim)	(0-10)	discreta
Aprendizado (Aprend)	(0-10)	discreta
Teste de fluência verbal (TFV)	(0-24)	discreta
Teste do desenho do relógio (TDR)	(1-10)	discreta
Memória tardia (MT)	(0-10)	discreta
Recordação (Rec)	(0-10)	discreta
GDS*	(0-15)	discreta
Pfeffer**	(0-30)	discreta

*: escala de depressão em idosos.
**: questionário de atividades funcionais de Pfeffer.

A seguir, será realizada a análise descritiva dos dados obtidos por Jacinto (2008) e, logo após, a construção e a comparação dos modelos de classificação para as diferentes metodologias descritas na Seção 3.

6.1) Análise Descritiva

Dos 245 pacientes participantes do estudo de Jacinto (2008), 186 (75,91%) eram mulheres e 59 (24,09%) eram homens. Do total de mulheres, 143 (76,88%) foram classificadas como “não suspeitas” e 43 (23,12%) como

“suspeitas” de possuírem demência. Do total de homens, 53 (86,83%) foram classificados como “não suspeitos” e 06 (13,17%) como “suspeitos”.

A Tabela 9 mostra algumas medidas-resumo para as covariáveis, OD, OI, Nom, MI, Mim, Aprend, TFV, TR, MT, Rec, GDS e Pfeffer. A Tabela 10 exibe algumas estatísticas descritivas das covariáveis Idade e Escolaridade.

Tabela 9: Estatísticas descritivas das covariáveis utilizadas na análise dos dados

Covariável	mínimo	1º quartil	mediana	média	3º quartil	máximo
OD	0	04	05	4,849	06	06
OI	0	01	02	2,302	03	06
Nom	0	10	10	9,743	10	10
MI	0	05	06	5,780	07	09
Mim	0	07	07	7,041	08	10
Aprend	0	07	08	7,727	09	10
TFV	0	09	11	11,760	15	24
TR	0	05	09	7,486	10	10
MT	0	06	07	6,976	08	10
Rec	0	09	10	9,253	10	10
GDS	0	01	03	3,771	06	13
Pfeffer	0	00	00	1,743	01	30

Tabela 10: Estatísticas descritivas das covariáveis Idade e Escolaridade

Covariável	Mín.	1º quartil	Mediana	3º quartil	Máx.	Média	Desvio-padrão
Idade	65	67	70	74	94	71,25	5,13
Escolaridade	0	02	04	08	18	4,78	4,02

Pela Tabela 10, verifica-se que 50% da amostra tem 4 anos ou menos de escolaridade (com média de 4,78 anos) e 50% tem idade acima de 70 anos (com média de 71,25 anos).

A Tabela 11 mostra as tabulações cruzadas dos seguintes pares de variáveis: Sexo vs. Suspeitos e Queixa vs. Suspeitos e seus respectivos p-valores para o teste de associação qui-quadrado.

Tabela 11: Frequências absolutas e relativas (em parênteses) das covariáveis Sexo vs. Suspeitos e Queixa vs. Suspeitos e p-valor do teste de associação qui-quadrado.

		Suspeitos		
		Não	Sim	p-valor
Sexo	Feminino	143(0,58)	43(0,17)	0,04773
	Masculino	53(0,22)	06(0,03)	
Queixa	Não	95(0,39)	05(0,02)	$2{,}453\times10^{-6}$
	Sim	101(0,41)	44(0,18)	

A Tabela 11 indica que há associação entre a variável Suspeitos com as variáveis Sexo e Queixa com p-valores 0,04773 e 2,453 x 10^{-6}, respectivamente.

6.2) Ajuste dos Modelos de Classificação

Das 245 observações do conjunto de dados, 70% (171 observações) foram reservadas para construir os modelos e 30% (74 observações) para validá-los (BURGER, 2018; AMARAL, 2016). Considerou-se a variável “Suspeitos” para tal divisão. Para isso foi usada amostragem aleatória simples sem reposição, utilizando o comando ‘*SplitRatio*’ da linguagem R. Com exceção das covariáveis Sexo e Queixa, todas as outras foram normalizadas (seção 3.2.3) para todas as metodologias utilizadas.

Em ambas as partições do conjunto de dados, há observações contendo valores “0” e “1” para a variável resposta, pois nos dois subconjuntos devem haver os dois valores para que o desempenho dos classificadores não seja comprometido.

Para a realização da comparação dos diferentes modelos de classificação foram construídas 1000 amostras de treinamento e 1000 amostras de teste (conforme mencionado). Neste processo, os diferentes métodos de classificação receberam as mesmas amostras.

Com a geração destas amostras e com a construção dos modelos de classificação, obtivemos 1000 estimativas para cada uma das seguintes medidas de desempenho: acurácia, sensibilidade e especificidade.

Com isso foi possível obter a “distribuição empírica” das medidas mencionadas para os diferentes modelos de classificação: *Naive Bayes*, regressão logística, Árvore de Classificação, *Random Forest*, $k - NN$ e Redes Neurais Artificiais apresentados no Capítulo 3.

Optou-se pela utilização do *software* estatístico R versão 3.5.3 (de domínio público) para a análise dos dados. Há funções disponíveis que foram necessárias para a modelagem dos dados: ‘*e1071*’ usada na abordagem *Naive*

Bayes, '*rpart*' utilizada na construção das Árvores de Classificação, '*class*' para o $k-NN$, '*glm*' para os modelos de Regressão Logística, '*randomForest*' para as RF e '*h2o*' para as Redes Neurais Artificiais (R CORE TEAM, 2017). Todos os códigos dos programas estão em Anexo.

Para o algoritmo $k-NN$, a Tabela 12 indica que o número de vizinhos mais adequado é $k=5$ pois foi o que apresentou a maior acurácia mediana quando comparado com outros valores de k. Além disso, é um valor pequeno e ímpar, frequentemente usado em problemas de classificação (FACELI et al, 2017). A distância utilizada foi a euclidiana.

Tabela 12: Estatísticas descritivas da distribuição da acurácia para o algoritmo $\boldsymbol{k-NN}$ considerando diferentes valores de $\boldsymbol{k}$.

Valores de k	Mediana	Média	Desvio-padrão
1	0,7973	0,7967	0,0361
3	0,8378	0,8396	0,0310
5	0,8518	0,8513	0,0283
7	0,8517	0,8512	0,0270
9	0,8514	0,8542	0,0263

Para os modelos utilizando a metodologia *Random Forest*, as florestas com o número de árvores iguais a 20, 30 e 40 apresentaram o mesmo valor para a acurácia mediana (Tabela 13). Porém escolheu-se o número de árvores igual a 20 pois é o menor valor e, consequentemente, um modelo *Random Forest* mais simples que os demais.

Tabela 13: Estatísticas descritivas da distribuição da acurácia para os modelos de *Random Forest* considerando diferentes número de árvores.

Número de árvores	Mediana	Média	Desvio-padrão
10	0,8919	0,8919	0,0291
20	0,9054	0,8996	0,0272
30	0,9054	0,9023	0,0271
40	0,9054	0,9036	0,0273

Considerando as Redes Neurais Artificiais, optou-se pela utilização da que possui uma camada oculta e mil iterações do algoritmo *backpropagation* pois apresentou uma acurácia mediana de 0,8243 (Tabela 14). Apesar de ter apresentado acurácia mediana igual a outras RNA com outros parâmetros, optou-se pela mais simples, ou seja, a que possui menos camadas ocultas e

menos iterações. Adotou-se nove como sendo o número de neurônio em cada camada oculta para todas as RNA da Tabela 14.

Tabela 14: Estatísticas descritivas da distribuição da acurácia para as Redes Neurais Artificiais considerando diferentes parâmetros.

Número de camadas ocultas	Número de iterações	Mediana	Média	Desvio-padrão
1	1000	0,8243	0,8234	0,0511
2	10000	0,8243	0,8241	0,0416
1	1000	0,7838	0,7857	0,0491
2	10000	0,8243	0,8181	0,0435

Partindo do pressuposto de que o modelo mais adequado será aquele que possui a maior medida (mediana) de desempenho (JAMES et al, 2013).

A Tabela 15 mostra algumas estatísticas descritivas da distribuição da acurácia para os seis modelos de classificação utilizados. Nestes modelos foram consideradas todas as covariáveis descritas na Tabela 8.

Tabela 15: Estatísticas descritivas da distribuição da acurácia para os diferentes modelos de classificação construídos considerando todas as covariáveis.

Classificador	Média	Mediana	Desvio-padrão	Coef. de variação (%)
RF	0,9005	0,9054	0,0274	3,0445
AC	0,8827	0,8784	0,0273	3,1032
$k-NN$	0,8513	0,8518	0,0288	3,3816
RL	0,8735	0,8784	0,0315	3,6095
NB	0,8315	0,8378	0,0353	4,2531
RNA	0,8667	0,8679	0,0311	3,5923

A Figura 6 mostra o gráfico *box plot* da distribuição empírica da acurácia dos modelos de classificação utilizados:

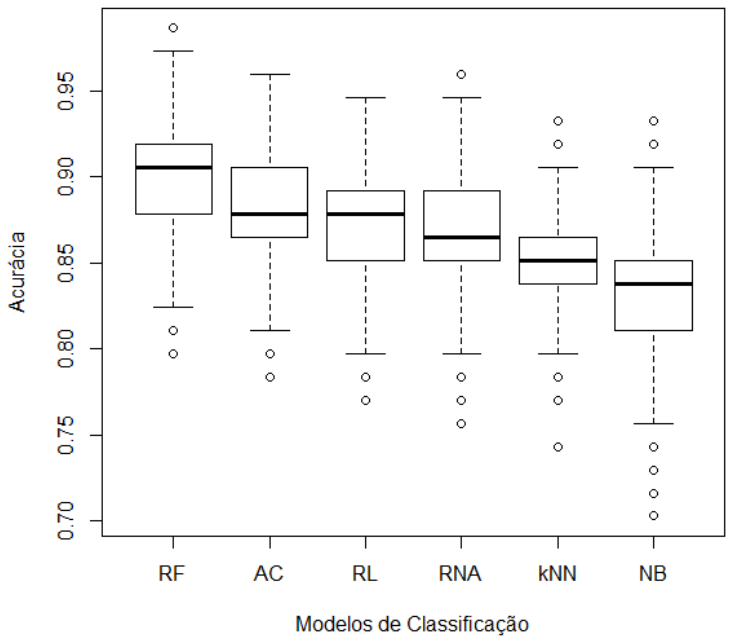


Figura 6: Box plot da distribuição da acurácia para os diferentes modelos de classificação construídos considerando todas as covariáveis.

Assim, de acordo com a Tabela 15 e a Figura 6, nota-se que o método de classificação que mostrou a maior acurácia média (0,9005) e mediana (0,9054) foi a RF seguido da RL, AC, RNA e $k-NN$, respectivamente. Além disso, a RF foi a que apresentou o menor valor para o coeficiente de variação (3,04%), mostrando-se o classificador mais homogêneo. O algoritmo *Naive Bayes* foi o que demonstrou pior desempenho (Figura 6) e o menos homogêneo, tendo como coeficiente de variação 4,25%.

A Tabela 16 exibe algumas estatísticas descritivas da distribuição da sensibilidade para os seis modelos de classificação utilizados.

Tabela 16: Estatísticas descritivas da distribuição da sensibilidade para os diferentes modelos de classificação construídos considerando todas as covariáveis.

Classificador	Média	Mediana	Desvio-padrão	Coef. de variação (%)
RF	0,9203	0,9194	0,0264	2,8718
AC	0,9208	0,9194	0,0285	3,1046
$k-NN$	0,8732	0,8730	0,0241	2,7623
RL	0,9072	0,9048	0,0244	2,6996
NB	0,8787	0,8750	0,0299	3,4028
RNA	0,8973	0,8983	0,0244	2,7295

A Figura 7 mostra o gráfico *box plot* da distribuição empírica da sensibilidade para cada um dos modelos de classificação utilizados.

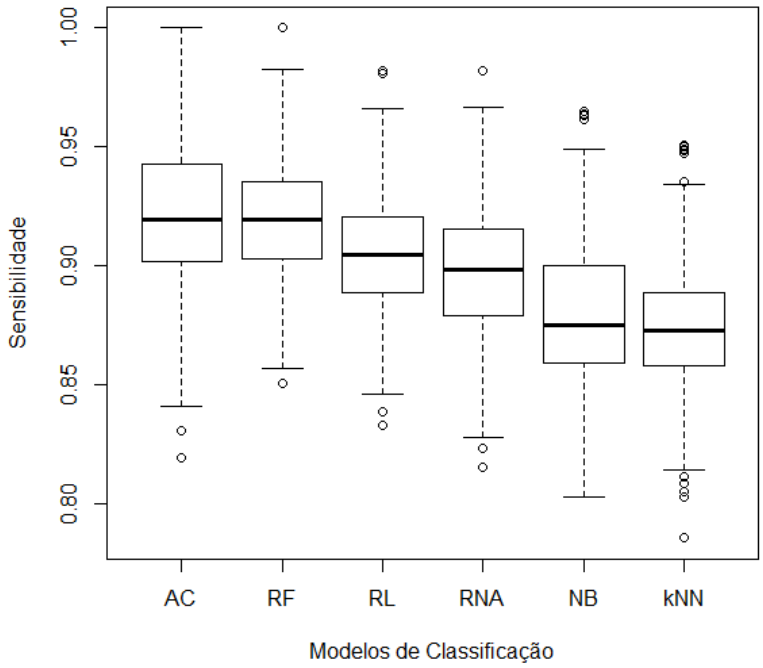


Figura 7: Box plot da distribuição da sensibilidade para os diferentes modelos de classificação construídos considerando todas as covariáveis.

Conforme a Tabela 16 e a Figura 7, nota-se que o método de classificação que mostrou a maior sensibilidade média (0,9208) e mediana (0,9194) foi o modelo de Árvore de Classificação seguido dos modelos: RF, RL, RNA e NB, respectivamente. O algoritmo $k - NN$ foi o que demonstrou pior desempenho (sensibilidade média de 0,8732).

A Tabela 17 exibe algumas estatísticas descritivas da distribuição da especificidade para os modelos de classificação utilizados.

Tabela 17: Estatísticas descritivas da distribuição da especificidade para os diferentes modelos de classificação construídos considerando todas as covariáveis.

Classificador	Média	Mediana	Desvio-padrão	Coef. de variação (%)
RF	0,8208	0,8182	0,0994	12,1143
AC	0,7410	0,7333	0,0891	12,0300
$k - NN$	0,7201	0,7143	0,1230	17,0829
RL	0,7324	0,7273	0,1032	14,0967
NB	0,6119	0,6111	0,1206	19,7084
RNA	0,7241	0,7273	0,1083	14,9579

A Figura 8 mostra o gráfico *box-plot* da distribuição empírica da especificidade para os cinco modelos de classificação utilizados.

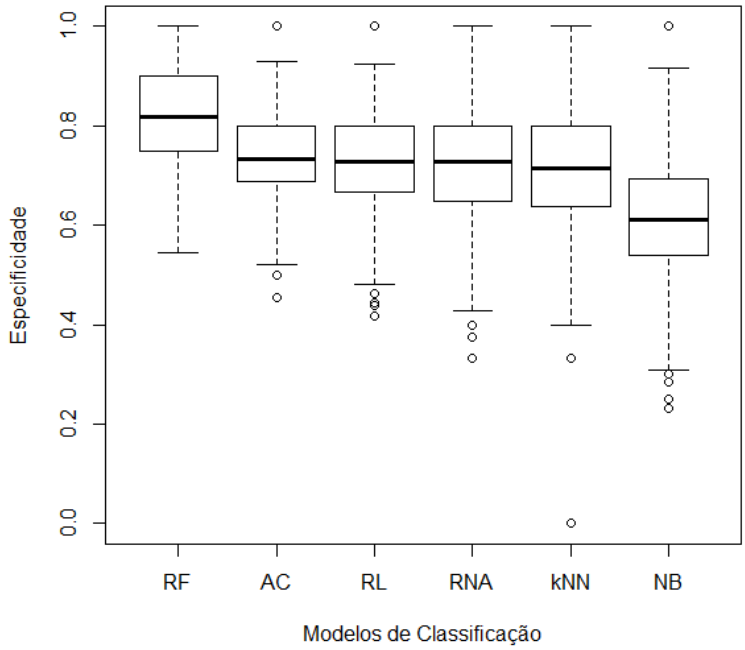


Figura 8: Box plot da distribuição da especificidade para os diferentes modelos de classificação construídos considerando todas as covariáveis.

Conforme a Tabela 17 e a Figura 8, o método de classificação que obteve maior especificidade média (0,8208) foi os modelos de Random Forest; na sequência aparecem AC, RL, RNA e $k - NN$. O algoritmo *Naive Bayes* foi o que demonstrou pior desempenho médio (0,6119) e o menos homogêneo, com coeficiente de variação 19,70%.

Tomando como referência o valor da acurácia mediana da RF (Tabela 15), das 1000 amostras geradas, 31 apresentaram acurácia mediana maior ou igual a 0,9054 para quatro ou mais modelos de classificação utilizados. Desta forma, para estas amostras foram gerados modelos de Regressão Logística como método de seleção das covariáveis mais significativas para explicar a variável resposta Suspeitos.

Para isso, foram analisadas a distribuição dos p-valores de todas as covariáveis da Tabela 8. Os resultados encontram-se na Tabela 18.

Tabela 18: Estatísticas descritivas da distribuição dos p-valores considerando todas as covariáveis.

Covariável	Mediana	Média	Desvio-padrão
Idade	0,1325	0,1485	0,1125
Sexo	0,1115	0,1697	0,2209
Escolaridade (anos de estudo)	0,1618	0,2089	0,1641
Queixa	0,0183	0,0430	0,0568
Ordem direta (OD)	0,6919	0,6670	0,2435
Ordem inversa (OI)	0,4151	0,4854	0,2312
Nomeação (Nom)	0,1064	0,2187	0,2698
Memória incidental (MI)	0,7271	0,7070	0,1651
Memória imediata (Mim)	0,8063	0,7677	0,1446
Aprendizado (Aprend)	0,5308	0,5521	0,2142
Teste de fluência verbal (TFV)	0,5827	0,5945	0,2042
Teste do desenho do relógio (TR)	0,0138	0,0368	0,0534
Memória tardia (MT)	0,7270	0,7031	0,2043
Recordação (Rec)	0,3514	0,3681	0,2705
GDS*	0,4485	0,4808	0,2613
Pfeffer**	0,0008	0,0013	0,0013

*: escala de depressão em idosos.
**: questionário de atividades funcionais de Pfeffer.

Optou-se por utilizar um valor médio do p-valor menor que 0,05. Assim, as covariáveis Queixa, Teste do desenho do relógio (TR) e Pfeffer foram significativas (Tabela 18).

Conforme a Tabela 18, a covariável Idade não mostrou-se estatisticamente significante ao nível de 0,05 (p-valor médio igual a 0,1485 e desvio padrão de 0,1125). Entretanto, visto que a idade é uma covariável de extrema importância no diagnóstico de demência, sendo o principal fator de risco para o desenvolvimento de demências, em especial o Alzheimer (NITRINI, CARAMELLI, HERRERA, 2004), esta foi incluída para compor o modelo final.

As estatísticas descritivas da distribuição dos p-valores do modelo composto pelas variáveis Idade, Queixa, Teste do desenho do relógio e Pfeffer encontram-se na Tabela 19.

Tabela 19: Estatísticas descritivas da distribuição dos p-valores das covariáveis Idade, Queixa, Teste do desenho do relógio e Pfeffer.

Covariável	Mediana	Média	Desvio-padrão
Idade	0,1335	0,1675	0,1283
Queixa	0,0071	0,0170	0,0196
Teste do desenho do relógio	0,0024	0,0060	0,0082
Pfeffer**	0,0001	0,0001	0,0001

**: questionário de atividades funcionais de Pfeffer.

Foram testados novamente diferentes valores de k para o algoritmo $k-NN$, número de árvores nos modelos de *Random Forest* e números de camadas ocultas e número de iterações do algoritmo *backpropagation* nas Redes Neurais Artificiais com o objetivo de verificar se haveria necessidade de mudar esses parâmetros visto que o número de covariáveis diminuiu de 16 para 04.

A Tabela 20 mostra estatísticas descritivas da distribuição da acurácia para o algoritmo $k-NN$ para diferentes valores de k, a Tabela 21 se refere as estatísticas descritivas dos modelos de *Random Forest* e a Tabela 22 para as RNA.

Tabela 20: Estatísticas descritivas da distribuição da acurácia para o algoritmo $\boldsymbol{k-NN}$ considerando diferentes valores de $\boldsymbol{k}$ para modelos contendo as covariáveis Idade, Queixa, Teste do relógio e Pfeffer.

Valores de k	Mediana	Média	Desvio-padrão
1	0,8919	0,8949	0,0296
3	0,8919	0,8926	0,0274
5	0,8919	0,8912	0,0251
7	0,9054	0,8989	0,0254
9	0,9054	0,8997	0,0260

Tabela 21: Estatísticas descritivas da distribuição da acurácia para os modelos de *Random Forest* considerando diferentes número de árvores para modelos contendo as covariáveis Idade, Queixa, Teste do relógio e Pfeffer.

Número de árvores	Mediana	Média	Desvio-padrão
10	0,9054	0,8978	0,0295
12	0,9054	0,8987	0,0290
14	0,9054	0,8992	0,0289
16	0,9054	0,9004	0,0290
18	0,9054	0,9010	0,0284
20	0,9054	0,9006	0,0290

Tabela 22: Estatísticas descritivas da distribuição da acurácia para as Redes Neurais Artificiais considerando diferentes parâmetros para modelos contendo as covariáveis Idade, Queixa, Teste do relógio e Pfeffer.

Nº de camadas ocultas (*)	Número de iterações	Mediana	Média	Desvio-padrão
1	1000	0,9054	0,9088	0,0291
2	10000	0,9054	0,9091	0,0289
1	1000	0,9012	0,8869	0,0292
2	10000	0,9054	0,9091	0,0291

(*): três neurônios em cada camada oculta.

Optou-se por utilizar o valor de $k=7$ para o algoritmo $k-NN$ pois apresentou acurácia mediana de 0,9054 (Tabela 20) e, comparado com $k=9$, é

menor, ou seja, torna o algoritmo mais simples e, consequentemente, mais adequado.

Para os modelos de RF, pela Tabela 21, todos os números de árvores considerados apresentaram a mesma acurácia mediana (0,9054). Então, optou-se por utilizar um modelo de RF com um número de árvores igual a 10 pois contém menos árvores e, consequentemente, mais simples que os demais. Quanto as RNA, escolheu-se a rede com uma camada oculta e 1000 iterações do algoritmo *backpropagation* (Tabela 22).

A seguir, foram calculadas estatísticas descritivas da distribuição da acurácia para os seis modelos de classificação descritos para as 1000 amostras de treinamento e de teste, considerando somente as covariáveis que aparecem na Tabela 19, ou seja, Idade, Queixa, Teste do desenho do relógio (TR) e Pfeffer (Tabela 23).

Tabela 23: Estatísticas descritivas da distribuição da acurácia para os diferentes modelos de classificação construídos considerando as covariáveis Idade, Queixa, TR e Pfeffer.

Classificador	Média	Mediana	Desvio-padrão	Coef. de variação (%)
RF	0,8978	0,9054	0,0295	3,2924
AC	0,8946	0,8919	0,0278	3,1105
$k-NN$	0,8989	0,9054	0,0254	2,8354
RL	0,9093	0,9054	0,0280	3,0801
NB	0,8905	0,8919	0,0259	2,9169
RNA	0,9088	0,9054	0,0281	3,0941

A Figura 9 mostra o gráfico *box plot* da distribuição empírica da acurácia dos modelos de classificação utilizados considerando somente as covariáveis da Tabela 19:

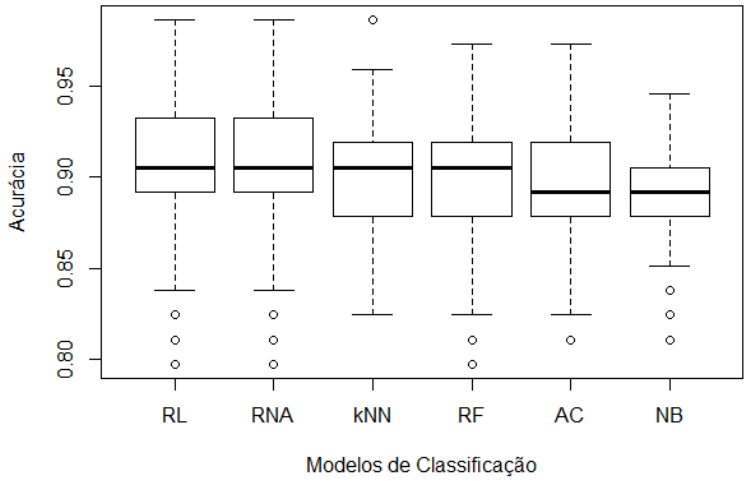


Figura 9: Box plot da distribuição da acurácia para os diferentes modelos de classificação construídos considerando as covariáveis Idade, Queixa, TR e Pfeffer.

De acordo com a Tabela 23 e a Figura 9, o modelo de Regressão Logística, as RNA, o algoritmo $k-NN$ e os modelos de *Random Forest* apresentaram acurácia mediana de 0,9054, porém o modelo de Regressão Logística apresentou o maior valor para a acurácia média (0,9093).

A Tabela 24 e a Figura 10 apresentam informações sobre a distribuição da sensibilidade considerando somente as covariáveis Idade, Queixa, Teste do desenho do relógio (TR) e Pfeffer.

Tabela 24: Estatísticas descritivas da distribuição da sensibilidade para os diferentes modelos de classificação construídos considerando as covariáveis Idade, Queixa, TR e Pfeffer.

Classificador	Média	Mediana	Desvio-padrão	Coef. de variação (%)
RF	0,9274	0,9310	0,0261	2,8207
AC	0,9291	0,9310	0,0274	2,9495
$k-NN$	0,9126	0,9167	0,0230	2,5254
RL	0,9303	0,9333	0,0237	2,5511
NB	0,9109	0,9077	0,0252	2,7763
RNA	0,9299	0,9333	0,0237	2,5590

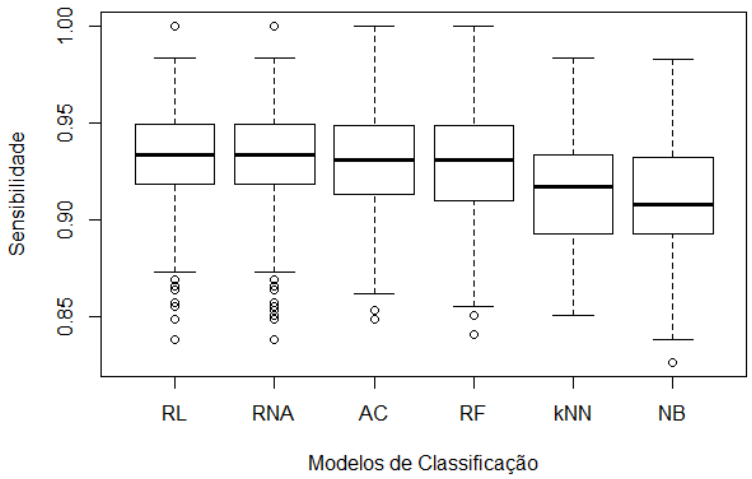


Figura 10: Box plot da distribuição da sensibilidade para os diferentes modelos de classificação construídos considerando as covariáveis Idade, Queixa, TR e Pfeffer.

Conforme a Tabela 24 e a Figura 10, os modelos de Regressão Logística e as Redes Neurais Artificiais possuem a mesma sensibilidade mediana (0,9333). Todavia, os valores da sensibilidade média diferem: 0,9303 para a RL e 0,9299 para as RNA.

A seguir, a Tabela 25 e a Figura 11 exibem estatísticas descritivas da distribuição da especificidade para diferentes modelos de classificação, considerando as covariáveis Idade, Queixa, Teste do desenho do relógio (TR) e Pfeffer.

Tabela 25: Estatísticas descritivas da distribuição da especificidade para os diferentes modelos de classificação construídos considerando as covariáveis Idade, Queixa, TR e Pfeffer.

Classificador	Média	Mediana	Desvio-padrão	Coef. de variação (%)
RF	0,7832	0,7854	0,0990	12,6527
AC	0,7670	0,7647	0,0959	12,5061
$k-NN$	0,8327	0,8333	0,0844	10,1403
RL	0,8226	0,8333	0,0922	11,2191
NB	0,7970	0,8000	0,0962	12,0743
RNA	0,8220	0,8333	0,0918	11,1755

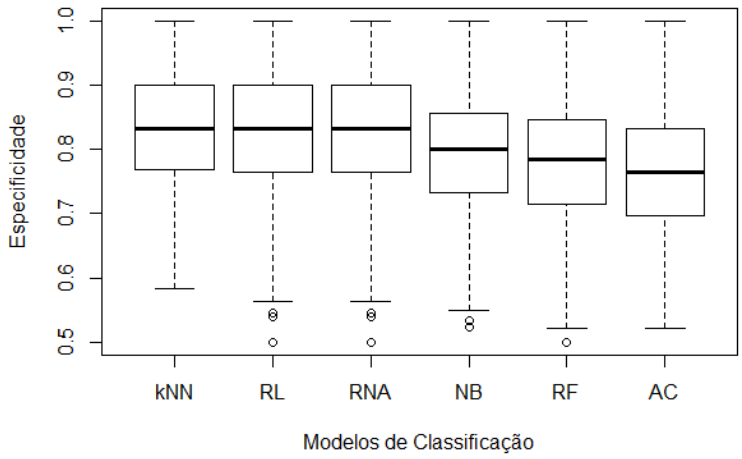


Figura 11: Box plot da distribuição da especificidade para os diferentes modelos de classificação construídos considerando as covariáveis Idade, Queixa, TR e Pfeffer.

Conforme a Tabela 25 e a Figura 11, as metodologias que apresentaram os maiores valores médios para a especificidade foram: o algoritmo $k-NN$, os modelos de Regressão Logística e as RNA respectivamente. Os modelos que apresentaram os valores mais baixos para a especificidade foram as RF e as AC. Para todos os seis modelos de classificação construídos, o coeficiente de variação apresentou valores entre 10,14% e 12,65%, sendo o $k-NN$ o mais homogêneo e as RF o menos homogêneo.

Quanto aos modelos de Regressão Logística, após a seleção de covariáveis, foi o que apresentou as melhores medidas de desempenho tanto para a acurácia quanto para a sensibilidade (Figuras 9 e 10). Assim os modelos de Regressão Logística foram escolhidos como os mais adequados para classificar pacientes como sendo “suspeitos” ou “não suspeitos” de possuírem alterações cognitivas.

Como exemplo, foi utilizada a amostra 566 para a geração de um modelo de Regressão Logística. A escolha dessa amostra se deu porque apresentou os seguintes valores para a acurácia: 0,9054 para os modelos de RF, AC, RNA e $k-NN$; 0,9324 para a RL e 0,8243 para o NB.

A Tabela 26 mostra um modelo de Regressão Logística aplicado aos dados da amostra 566.

Tabela 26: Estatísticas e estimativas dos parâmetros do modelo de Regressão Logística para a amostra 566

Covariável	Estimativa	Erro-Padrão	Estatística z	p-valor
Intercepto	1,72326	4,41997	0,390	0,69662
Idade	-0,04977	0,06011	-0,828	0,40773
Queixa	1,64429	0,79502	2,068	0,03862
TR	-0,24509	0,09306	-2,634	0,00844
Pfeffer	0,43380	0,10590	4,096	$4{,}2\times10^{-5}$

A expressão seguinte nos fornece a probabilidade de um paciente ser classificado como “suspeito” ou “não suspeito” de possuir demência em função das covariáveis Idade, Queixa, Teste do desenho do relógio (TR) e Pfeffer:

$$\hat{\pi} = \frac{exp(1{,}72326 - 0{,}04977 * Idade + 1{,}64429 * Queixa - 0{,}24509 * TR + 0{,}43380 * Pfeffer)}{1 + exp(1{,}72326 - 0{,}04977 * Idade + 1{,}64429 * Queixa - 0{,}24509 * TR + 0{,}43380 * Pfeffer)} \quad (7.1)$$

A Figura 12 mostra o gráfico do envelope simulado para o modelo de Regressão Logística (Tabela 26).

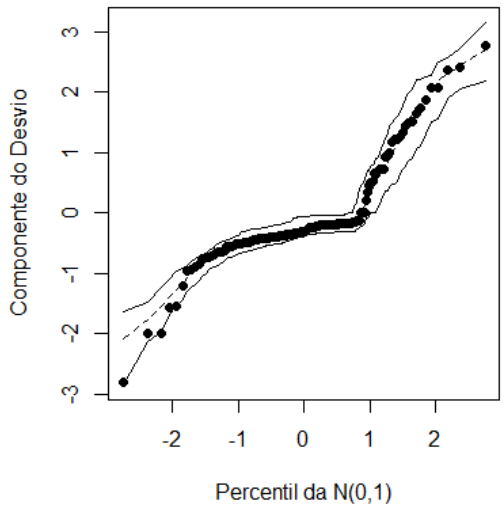


Figura 12: Envelope simulado para o modelo de Regressão Logística da Tabela 26.

Nota-se, pela Figura 12, que não há pontos (resíduos padronizados) fora das bandas de confiança, o que é um bom indicativo da qualidade de ajuste, ou seja, o modelo se ajustou bem aos dados.

7) CONSIDERAÇÕES FINAIS

Considerando os modelos com todas as covariáveis do estudo de Jacinto (2008), com exceção do algoritmo *Naive Bayes* (Tabela 15 e Figura 6), todos os outros modelos de classificação apresentaram acurácia média e mediana acima de 0,85, ou seja, tiveram uma taxa de acerto total na classificação de 85% o que demonstra um desempenho satisfatório dos classificadores utilizados neste trabalho.

Ainda em relação aos modelos de classificação que consideraram todas as covariáveis, em relação à sensibilidade (capacidade de identificar corretamente aqueles que apresentam a doença), os modelos de Árvore de Classificação e os modelos de *Random Forest* apresentaram uma sensibilidade mediana de 91,94% (Tabela 16 e Figura 7), porém a sensibilidade média dos modelos de Árvore de Classificação foi ligeiramente maior (92,08%).

Após a aplicação de um método de seleção de variáveis, verificou-se que as covariáveis Queixa subjetiva de memória (Queixa), Teste do desenho do relógio (TR) e o questionário de atividades funcionais de Pfeffer apresentaram significância estatística com p-valores médios menores que 0,05. Com exceção da covariável Queixa, as outras duas foram as que mais contribuíram para o diagnóstico de declínio cognitivo no estudo de Jacinto (2008).

Para compor o modelo final, a covariável Idade, mesmo não apresentando significância estatística, foi incluída por ser extremamente importante no diagnóstico de demências na população idosa.

Em relação aos modelos de classificação somente com as covariáveis Idade, Queixa, Teste do desenho do relógio e Pfeffer, houve um aumento da acurácia mediana (Tabela 23) quando comparada com a acurácia mediana do modelo que considerou todas as covariáveis (Tabela 15) para todos os modelos de classificação, com exceção dos modelos de *Random Forest,* que apresentaram o mesmo valor.

O menor valor observado para a acurácia mediana (Tabela 23 e Figura 9) foi o do algoritmo *Naive Bayes* (0,8919) mostrando que a seleção de covariáveis

foi útil para o aumento de desempenho dos modelos considerados. Os modelos com acurácia em torno de 90% demonstram bom desempenho dos classificadores.

Houve um aumento na sensibilidade mediana após a seleção de variáveis para os seis modelos considerados. Para a especificidade mediana, houve uma queda de desempenho dos modelos de *Random Forest*, de 0,8182 para 0,7857 mas um aumento para as demais técnicas (Tabela 17 e Tabela 22).

Considerando as medidas de desempenho acurácia e sensibilidade, de todos os modelos de classificação, o algoritmo *Naive Bayes* foi a metodologia que apresentou desempenho inferior, tanto antes como depois do processo de seleção de covariáveis e, portanto, foi descartado de um possível uso prático no auxílio do diagnóstico de demência ou declínio cognitivo.

Houve uma melhora no desempenho das Redes Neurais Artificiais após a seleção de covariáveis. Entretanto, visto que sua implementação foi trabalhosa e o tempo de treinamento foi alto, é preferível escolher outros modelos de classificação mais simples e que não necessitem de alto poder computacional.

Os modelos de Árvore de Classificação apresentaram bons valores para a acurácia e sensibilidade antes do processo de seleção de covariáveis. Porém, após a seleção, ficaram atrás de outros modelos de classificação (Figura 6 e Figura 9). O mesmo ocorreu com os modelos de *Random Forest*.

Visto que o estudo de Jacinto (2008) concluiu “que o declínio cognitivo em idosos é pouco detectado por médicos generalistas”, o modelo de Regressão Logística (Tabela 26 e expressão 7.1) poderia ser um ótimo instrumento de auxílio a estes profissionais.

Uma maneira de tornar isso prático e funcional para os médicos generalistas seria, por exemplo, a criação de um *website* onde poderiam ser inseridos as covariáveis Idade, Queixa, Teste do desenho do relógio e o questionário de atividades funcionais de Pfeffer e o modelo de Regressão Logística estando de modo embutido no sistema, como, por exemplo a expressão (7.1). Este sistema daria a probabilidade de um paciente estar doente e sua classificação.

Desta forma, o resultado desta tese seria de proveito a estes profissionais no diagnóstico e na tomada de decisão como, por exemplo, encaminhando os pacientes a médicos especialistas (geriatras, neurologistas ou psiquiatras) ou, se fosse o caso, a profissionais ligados a reabilitação (por exemplo, fisioterapeutas, terapeutas ocupacionais ou fonoaudiólogos) melhorando assim a qualidade de vida destes pacientes e de seus cuidadores.

REFERÊNCIAS BIBLIOGRÁFICAS

AGRESTI, A. **Categorical data analysis**. 3 ed. Wiley, 2013.

ALMEIDA, O. P.; ALMEIDA S. A. **Confiabilidade da versão brasileira da escala de depressão em geriatria (GDS) versão reduzida.** Arquivos de Neuro-Psiquiatria, v. 57, 1999.

ALPAYDIN, E. **Introduction to machine learning**. 2 ed. Cambridge: The MIT Press (Adaptative Computation and Machine Learning), 2010.

AMARAL, F. **Introdução à ciência de dados: mineração de dados e Big Data**. Rio de Janeiro: Alta Books, 2016.

BEERAVALLI, V. **Apresenta informações sobre comparação de modelos de classificação de machine learning**. Disponível em: https://medium.com/@vijaya.beeravalli/comparison-of-machine-learning-classification-models-for-credit-card-default-data-c3cf805c9a5a. Acesso em: 01 abr. 2019.

BITTENCOURT, H. R.; CLARKE, R.T. **Um classificador baseado na discriminação logística: vantagens e desvantagens**. ANAIS X SBSR, Foz do Iguaçu, INPE, p. 1217-1223, 2001.

BORGES, Jr. S. R. **SEnsembles – uma abordagem para melhorar a qualidade das correspondências de instâncias disjuntas em estudos observacionais explorando características idênticas e ensembles de regressores**. Tese (doutorado), Universidade Federal de São Carlos. 190 f. 2016.

BRAGA, A. P.; CARVALHO, A. P. L. F.; LUDERMIR, T. B. **Redes Neurais Artificiais – Teoria e Aplicações**. 2 ed. Rio de Janeiro. LTC, 2011.

BURGER, S. V. **Introduction to machine learning with R: rigorous mathematical analysis**. O'Reilly, 2018.

COUTINHO, E. R.; SILVA, R. M.; DELGADO, A. R. S. **Utilização de técnicas computacionais na predição de dados meteorológicos**. Revista Brasileira de Meteorologia, v. 31, 2016.

DANTAS, C. A. B. **Probabilidade: um curso introdutório**. 3 ed. São Paulo: EDUSP, 2008.

FACELI, K.; LORENA, A. C.; GAMA, J.; CARVALHO, A. C. P. L. F. **Inteligência artificial: uma abordagem de aprendizado de máquina**. Rio de Janeiro: LTC, 2017.

FLACH, P. **Machine Learning: the art and Science of algorithms that make sense of data.** New York: Cambridge University Press, 2012.

FORLENZA, O.V.; MIGUEL E. C. **Compêndio de clínica psiquiátrica**. Barueri: Manole, 2012.

GARDNER, H.; KORNHABER, M. L.; WAKE, W. K. **Inteligência: múltiplas perspectivas.** Porto Alegre: Artmed, 1998.

GIOLO, S. R. **Introdução à análise de dados categóricos com aplicações.** São Paulo: Blucher, 2017

GORDIS, L. **Epidemiologia**. 4 ed. Rio de Janeiro: Revinter, 2010.

GORENSTEIN, C.; WANG, Y-P.; HUNGERBÜHLER, I. **Instrumentos de avaliação em saúde mental**. Porto Alegre: Artmed, 2016.

GUDWIN, R. R. **Apresenta informações sobre sistemas inteligentes**. Disponível em: https://slideplayer.com.br/slide/5631904/. Acesso em 19 dez. 2019.

HAIR, J. F.; BLACK, W. C.; BABIN, B. J.; ANDERSON, R. E. **Multivariate data analysis**. 7 ed. Pearson Prentice Hall, 2009.

HARRINGTON, P. **Machine learning in action**. New York: Manning, 2012.

HOSMER, D.W.; LEMESHOW, S.; STURDIVANT. R. X. **Applied logistic regression.** 3 ed. Hoboken: Wiley, 2013.

JACINTO, A. F. **Alterações cognitivas em pacientes idosos atendidos em ambulatório geral de clínica médica**. Tese (doutorado), Universidade de São Paulo, Faculdade de Medicina. 86 f. 2008.

JAMES, G.; WITTEN, D.; HASTIE, T.; TIBSHIRANI, R. **An introduction to statistical learning**. New York: Springer, 2013.

JOHNSON, R. A.; WICHERN, D.W. **Applied multivariate statistical analysis**. Delhi: Pearson Prentice Hall, 2007.

KELLEHER, J.D.; NAMEE, B.M.; D'ARCY. A. **Fundamentals of machine learning for predictive data analysis**. Cambridge: The MIT Press, 2015.

KUBAT, M. **An introduction to machine learning**. New York: Springer, 2015.

KULKARNI, S.; HARMAN, G. **An elementary introduction to statistical learning theory**. Hoboken: Wiley, 2011.

LUZ, F. **Apresenta informações sobre o algoritmo k-NN para classificação**. Disponível em: https://inferir.com.br/artigos/algoritimo-knn-para-classificacao/. Acesso em: 02 abr. 2019.

MOHRI, M.; ROSTAMIZADEH, A.; TALWALKAR, A. **Foundation of machine learning**. 2 ed. Cambridge: The MIT Press, 2018.

MONTGOMERY, D. C.; PECK, E. A.; VINING, G. G. **Introduction to linear regression analysis**. 4 ed. Hoboken: Wiley, 2006.

NITRINI, R.; CARAMELLI, P.; HERRERA, J.R. **Incidence of dementia in a community-dwelling Brazilian population.** Alzheimer Dis. Assoc. Disord., v. 18, n. 4, 2004, 241-246.

PAULA, G. A. **Apresenta informações sobre modelos de regressão com apoio computacional.** Disponível em: https://www.ime.usp.br/~giapaula/textoregressao.htm. Acesso em: 01 dez. 2019.

R CORE TEAM. R: **A language and environment for statistical computing**. R Foundation for Statistical Computing. Vienna, Austria, 2017.

RAU, H.H.; HSU, C.Y.; LIN, Y. A.; ATIQUE, S.; FUAD, A.; WEI, L.M.; HSU, M.H. **Development of a web-based liver câncer prediction model for type II diabetes patients by using an artificial neural network.** Elsevier, 2016; 58-65.

REBALA, G.; RAV, A.; CHURIWALA, S. **An introduction to machine learning**. New York: Springer, 2019.

ROKACH, L,; MAIMON, O. **Data mining with decision trees: theory and applications**. Singapore. World Scientific: series in machine learning perception and artificial intelligence (vol. 81), 2015.

SATO, H. H. **Apresenta informações e reflexões sobre a profissão de médico generalista.** Disponível em: https://academiamedica.com.br/blog/apertem-os-cintos-o-medico-generalista-sumiu. Acesso em: 19 jun. 2019.

SHWARTZ, S.S.; DAVID, S.B. **Understanding machine learning: from theory to algorithms.** New York: Cambridge University Press, 2014.

STOKES, M.E.; DAVIS, C.S.; KOCH, G. G. **Categorical data analysis using the SAS system**. 6 ed. C, N.C.: SAS Institute, 2000.

TORGO, L. **Data mining with R: learning with case studies**. Boca Raton: Springer, 2015.

ANEXOS

A) MINI EXAME DO ESTADO MENTAL (MEEM)
(Folstein et al, 1975; Brucki et al, 2003)

ORIENTAÇÃO
- Dia da semana:
- Dia do mês:
- Mês:
- Ano:
- Hora aproximada:
- Local específico (aposento ou setor):
- Instituição (hospital, residência, clínica):
- Bairro ou rua próxima:
- Cidade:
- Estado:

MEMÓRIA IMEDIATA
- Vaso, carro, tijolo:

ATENÇÃO E CÁLCULO
- 100-7 sucessivos

EVOCAÇÃO
- Recordar as 3 palavras:

LINGUAGEM
- Nomear um relógio e uma caneta:
- Repetir: ”Nem aqui nem ali nem lá”:
- Comando: “Pegue este papel com sua mão direita, dobre ao meio e coloque no chão”:
- Ler e obedecer: “Feche os olhos”:
- Escreva uma frase:
- Copiar um desenho:

ESCORE:__ / 30

SOLETRAR
- Soletrar a palavra “mundo” de trás para frente:

ESCORE:__ / 35

B) BATERIA COGNITIVA BREVE
(Nitrini et al, 1994)

PERCEPÇÃO VISUAL E NOMEAÇÃO
Mostre a folha (adiante) contendo as 10 figuras e pergunte: “Que figuras são estas?”
Percepção correta: _____
Nomeação correta: _____

MEMÓRIA INCIDENTAL
Esconda as figuras e pergunte: “Que figuras eu acabei de lhe mostrar?”
(Tempo máximo de evocação: 60 segundos)
Escore (número de acertos): ___

MEMÓRIA IMEDIATA 1
Mostre as figuras novamente durante 30 segundos dizendo: “Olhe bem e procure memorizar estas figuras” (Se houver déficit visual importante, peça que memorize as palavras que você vai dizer; diga os nomes dos objetos lentamente, um nome/segundo; fale a série toda duas vezes). Esconda as figuras e pergunte: “Que figuras eu acabei de lhe mostrar?” (Tempo máximo de evocação: 60 segundos)
Escore (número de acertos): ___

MEMÓRIA IMEDIATA 2
Mostre as figuras novamente durante 30 segundos dizendo: “Olhe bem e procure memorizar estas figuras” (Se houver déficit visual importante, peça que memorize as palavras que você vai dizer; diga os nomes dos objetos lentamente, um nome/segundo; fale a série toda duas vezes). Esconda as figuras e pergunte: “Que figuras eu acabei de lhe mostrar?” (Tempo máximo de evocação: 60 segundos)
Escore (número de acertos): ___

TESTE DE FLUÊNCIA VERBAL
“Você deve falar todos os nomes de animais (qualquer bicho) que se lembrar, no menor tempo possível. Pode começar”.
Anote o número de animais lembrados em 1 minuto: _____

_____
_____
_____
_____
_____

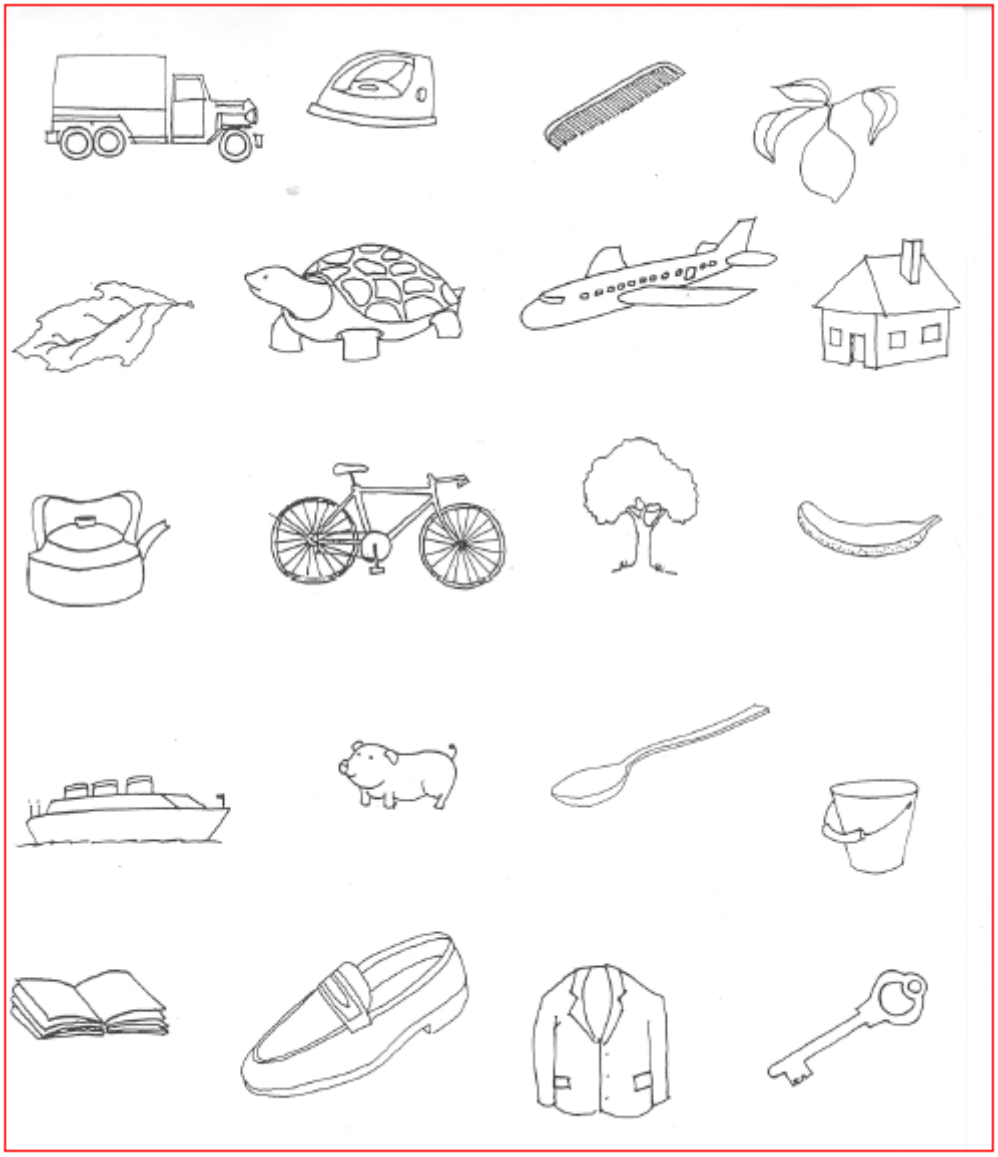
DESENHO DO RELÓGIO (Sunderland et al, 1989)
Dê uma folha de papel em branco e diga: “Desenhe um relógio com todos os números. Coloque ponteiros marcando 2h45” (guarde o desenho com a ficha).
Avaliação: 10-6: relógio e números estão corretos
10 – hora certa
09 – leve distúrbio nos ponteiros (p. ex.: ponteiro das horas sobre o 2)
08 – distúrbios mais intensos nos ponteiros (p. ex.: anotando 2:20)
07 – ponteiros completamente errados

06 – uso inapropriado (p. ex.: uso de código digital ou de círculos envolvendo números)

Avaliação 5-1: desenhos do relógio e dos números incorretos
5 – números em ordem inversa ou concentrados em alguma parte do relógio
4 – números faltando ou situados fora dos limites do relógio
3 – números e relógios não mais conectados. Ausência de ponteiros
2 – alguma evidência de ter entendido as instruções mas com vaga semelhança com um relógio
1 – não tentou ou não conseguiu representar um relógio

MEMÓRIA TARDIA (5 minutos)
"Que figuras eu lhe mostrei há alguns minutos?" Se necessário, reforce, dizendo figuras desenhadas numa folha de papel plastificada (60 segundos).
Escore: _____ (Instruções: _____)

RECONHECIMENTO
Mostre a folha contendo as 20 figuras e diga: "Aqui estão as figuras que eu lhe mostrei hoje e outras figuras novas. Quero que você me diga quais você já tinha visto há alguns minutos".
Escore: _____ (Instruções: _____)



C) PROGRAMAÇÃO EM R

```
#analise descritiva dos dados

getwd()
setwd("C:/Users/maicon.galdino/Desktop")

dadosbrutos <- read.table('tesemaicon.csv',header=T,dec=',',sep=';')
attach(dadosbrutos)

Sexo <- as.factor(Sexo)
Queixa <- as.factor(Queixa)
Suspeitos <- as.factor(Suspeitos)

#Sexotb <- table(Sexo)
#prop.table(Sexotb)
#Queixatb <- table(Queixa)
#prop.table(Queixatb)
#Suspeitostb <- table(Suspeitos)
#prop.table(Suspeitostb)


OD <- dadosbrutos$OD
OI <- dadosbrutos$OI
Nom <- dadosbrutos$Nom
MI <- dadosbrutos$MI
Mim <- dadosbrutos$Mim
Aprend <- dadosbrutos$Aprend
TFV <- dadosbrutos$TFV
TR <-dadosbrutos$TR
MT <- dadosbrutos$MT
Rec <- dadosbrutos$Rec
GDS <- dadosbrutos$GDS
Pfeffer <- dadosbrutos$Pfeffer


summary(OD)
summary(OI)
summary(Nom)
summary(MI)
summary(Mim)
summary(Aprend)
summary(TFV)
summary(TR)
summary(MT)
summary(Rec)
summary(GDS)
summary(Pfeffer)
Idade <- dadosbrutos$Idade
sd(Idade)

Escol <- dadosbrutos$Escol
sd(Escol)

summary(Idade)
summary(Escol)


freqsus_sex=table(Sexo,Suspeitos)
freqsus_sex
```

```
chisq.test(Sexo,Suspeitos)

freqsus_queixa=table(Queixa,Suspeitos)
freqsus_queixa
chisq.test(Queixa,Suspeitos)


#modelagem dos dados

getwd()
setwd("C:/Users/maicon.galdino/Desktop/tese")

#install.packages("caret",dependencies = c("Depends","Suggests"))

#leitura dos dados----------------------------------------------
dados <- read.table('tesemaicon.csv',header=T,dec=',',sep=';')

# processamento dos dados-----------------------------------
dados$Obs=NULL
dados$Ambul=NULL
dados$Sexo = factor(dados$Sexo,levels = c(0,1))
dados$Queixa = factor(dados$Queixa,levels = c(0,1))

#escalonamento dos dados---------------------------------
dados[,1]=scale(dados[,1])
dados[,3]=scale(dados[,3])
dados[,5:16]=scale(dados[,5:16])

#codificacao da classe-----------------------
dados$Suspeitos = factor(dados$Suspeitos,levels = c(0,1))

#instalação de pacotes----------------------------------

#install.packages('e1071')
library(e1071)
#install.packages('rpart')
library(rpart)
#install.packages('randomForest')
library(randomForest)
#install.packages('class')
library(class)
#install.packages('caTools')
library(caTools)
#install.packages("caret")
library(caret)

#1naiveBayes-----------------------------------------
resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  classificador1=naiveBayes(x=dados_treino[-17],y=dados_treino$Suspeitos)
  previsoes1=predict(classificador1,newdata=dados_teste[-17])
  matriz_confusao1 = table(dados_teste[,17],previsoes1)
  t=confusionMatrix(matriz_confusao1)
  MC=t$table
  a11=MC[1,1]
```

```
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
  acuracia=(a11+a22)/74
  print(acuracia)
  resultados1000=c(resultados1000,acuracia)
}

for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}
#-----------------------------------------------------------------
resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  classificador1=naiveBayes(x=dados_treino[-17],y=dados_treino$Suspeitos)
  previsoes1=predict(classificador1,newdata=dados_teste[-17])
  matriz_confusao1 = table(dados_teste[,17],previsoes1)
  t=confusionMatrix(matriz_confusao1)
  MC=t$table
  a11=MC[1,1]
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
  sens=a11/(a11+a21)
            print(sens)
  resultados1000=c(resultados1000,sens)
}

for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}
#-----------------------------------------------------------------------------
resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  classificador1=naiveBayes(x=dados_treino[-17],y=dados_treino$Suspeitos)
  previsoes1=predict(classificador1,newdata=dados_teste[-17])
  matriz_confusao1 = table(dados_teste[,17],previsoes1)
  t=confusionMatrix(matriz_confusao1)
  MC=t$table
  a11=MC[1,1]
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
  especif=a22/(a12+a22)
  print(especif)
  resultados1000=c(resultados1000,especif)
```

```
}
for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}

#2arvore de decisao------------------
resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  classificador2=rpart(formula = Suspeitos~.,data=dados_treino)
  previsoes2 = predict(classificador2, newdata = dados_teste[-17], type = 'class')
  matriz_confusao2 = table(dados_teste[, 17], previsoes2)
  t=confusionMatrix(matriz_confusao2)
  MC=t$table
  a11=MC[1,1]
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
acuracia=(a11+a22)/74
  print(acuracia)
  resultados1000=c(resultados1000,acuracia)
}

for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}
#-------------------------------------------------------------------------------------------
resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  classificador2=rpart(formula = Suspeitos~.,data=dados_treino)
  previsoes2 = predict(classificador2, newdata = dados_teste[-17], type = 'class')
  matriz_confusao2 = table(dados_teste[,17],previsoes2)
  t=confusionMatrix(matriz_confusao2)
  MC=t$table
  a11=MC[1,1]
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
  sens=a11/(a11+a21)
  print(sens)
  resultados1000=c(resultados1000,sens)
}

for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
```

```
}
#-------------------------------------------------------------------------------

resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  classificador2=rpart(formula = Suspeitos~.,data=dados_treino)
  previsoes2 = predict(classificador2, newdata = dados_teste[-17], type = 'class')
  matriz_confusao2 = table(dados_teste[,17],previsoes2)
  t=confusionMatrix(matriz_confusao2)
  MC=t$table
  a11=MC[1,1]
  a12=MC[1,2]
a21=MC[2,1]
  a22=MC[2,2]
  especif=a22/(a12+a22)
  print(especif)
  resultados1000=c(resultados1000,especif)
}
for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}


#3randomForest------------------
resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  classificador3 = randomForest(x = dados_treino[-17], y = dados_treino$Suspeitos, ntree = 20)
  previsoes3 = predict(classificador3, newdata = dados_teste[-17])
  matriz_confusao3 = table(dados_teste[,17], previsoes3)
  t=confusionMatrix(matriz_confusao3)
  MC=t$table
  a11=MC[1,1]
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
  acuracia=(a11+a22)/74
  print(acuracia)
  resultados1000=c(resultados1000,acuracia)
}

for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}
#-------------------------------------------------------------------------------

resultados1000=c( )
```

```
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  classificador3 = randomForest(x = dados_treino[-17], y = dados_treino$Suspeitos, ntree = 20)
            previsoes3 = predict(classificador3, newdata = dados_teste[-17])
matriz_confusao3 = table(dados_teste[,17],previsoes3)
  t=confusionMatrix(matriz_confusao3)
  MC=t$table
  a11=MC[1,1]
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
  sens=a11/(a11+a21)
  print(sens)
  resultados1000=c(resultados1000,sens)
}

for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}

#-------------------------------------------------------------------------------------
resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  classificador3 = randomForest(x = dados_treino[-17], y = dados_treino$Suspeitos, ntree = 20)
  previsoes3 = predict(classificador3, newdata = dados_teste[-17])
  matriz_confusao3 = table(dados_teste[,17],previsoes3)
  t=confusionMatrix(matriz_confusao3)
  MC=t$table
  a11=MC[1,1]
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
  especif=a22/(a12+a22)
  print(especif)
  resultados1000=c(resultados1000,especif)
}
for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}
#4knn------------------

resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
```

```
dados_teste=subset(dados,divisao==FALSE)
  previsoes4 = knn(train = dados_treino[, -17], test = dados_teste[, -17],cl = dados_treino[, 17], k
= 3)
  matriz_confusao4 = table(dados_teste[,17], previsoes4)
  t=confusionMatrix(matriz_confusao4)
  MC=t$table
  a11=MC[1,1]
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
  acuracia=(a11+a22)/74
  print(acuracia)
  resultados1000=c(resultados1000,acuracia)
}
for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}

#------------------------------------------------------------------------
resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  previsoes4 = knn(train = dados_treino[, -17], test = dados_teste[, -17],cl = dados_treino[, 17], k
= 3)
  matriz_confusao4 = table(dados_teste[,17], previsoes4)
  t=confusionMatrix(matriz_confusao4)
  MC=t$table
  a11=MC[1,1]
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
  sens=a11/(a11+a21)
  print(sens)
  resultados1000=c(resultados1000,sens)
}
for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}

#------------------------------------------------------------------------
resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  previsoes4 = knn(train = dados_treino[, -17], test = dados_teste[, -17],cl = dados_treino[, 17], k
= 3)
            matriz_confusao4 = table(dados_teste[,17], previsoes4)
  t=confusionMatrix(matriz_confusao4)
  MC=t$table
```

```
  a11=MC[1,1]
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
  especif=a22/(a12+a22)
  print(especif)
  resultados1000=c(resultados1000,especif)
}
for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}

#5regressao logística----------
resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  classificador5 = glm(formula = Suspeitos ~ ., family = binomial, data = dados_treino)
  probabilidades = predict(classificador5, type = 'response', newdata = dados_teste[-17])
  previsoes5 = ifelse(probabilidades > 0.5, 1, 0)
  matriz_confusao5 = table(dados_teste[,17], previsoes5)
  t=confusionMatrix(matriz_confusao5)
  MC=t$table
  a11=MC[1,1]
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
  acuracia=(a11+a22)/74
  print(acuracia)
  resultados1000=c(resultados1000,acuracia)
}
for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}

#------------------------------------------------------------------------
resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
            classificador5 = glm(formula = Suspeitos ~ ., family = binomial, data = dados_treino)
probabilidades = predict(classificador5, type = 'response', newdata = dados_teste[-17])
  previsoes5 = ifelse(probabilidades > 0.5, 1, 0)
  matriz_confusao5 = table(dados_teste[,17], previsoes5)
  t=confusionMatrix(matriz_confusao5)
  MC=t$table
  a11=MC[1,1]
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
```

```
  sens=a11/(a11+a21)
  print(sens)
  resultados1000=c(resultados1000,sens)
}
for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}

#--------------------------------------------------------------------------------
resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  classificador5 = glm(formula = Suspeitos ~ ., family = binomial, data = dados_treino)
  probabilidades = predict(classificador5, type = 'response', newdata = dados_teste[-17])
  previsoes5 = ifelse(probabilidades > 0.5, 1, 0)
  matriz_confusao5 = table(dados_teste[,17], previsoes5)
  t=confusionMatrix(matriz_confusao5)
  MC=t$table
  a11=MC[1,1]
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
  especif=a22/(a12+a22)
  print(especif)
  resultados1000=c(resultados1000,especif)
}
for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}



#RNA-----------------------------------------

install.packages('h2o')
library(h2o)
resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  classificador7=h2o.deeplearning(y='Suspeitos',training_frame = as.h2o(dados_treino),
                                  activation='Rectifier',
                                  hidden=c(9),
                                  epochs=1000)
  previsoes7=h2o.predict(classificador7,newdata=as.h2o(dados_teste[-17]))
  previsoes7=(previsoes7>0.5)
  previsoes7=as.vector(previsoes7)
  matriz_confusao7 = table(dados_teste[,17],previsoes7)
  t=confusionMatrix(matriz_confusao7)
```

```
  MC=t$table
  a11=MC[1,1]
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
  acuracia=(a11+a22)/74
  print(acuracia)
  resultados1000=c(resultados1000,acuracia)
}

for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}

#-----------------------------------------------------------------
resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  classificador7=h2o.deeplearning(y='Suspeitos',training_frame = as.h2o(dados_treino),
                                  activation='Rectifier',
                                  hidden=c(9),
                                  epochs=1000)
  previsoes7=h2o.predict(classificador7,newdata=as.h2o(dados_teste[-17]))
  previsoes7=(previsoes7>0.5)
  previsoes7=as.vector(previsoes7)
  matriz_confusao7 = table(dados_teste[,17],previsoes7)
  t=confusionMatrix(matriz_confusao7)
  MC=t$table
  a11=MC[1,1]
              a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
  sens=a11/(a11+a21)
  print(sens)
  resultados1000=c(resultados1000,sens)
}

for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}

#-------------------------------------------------------------------------------
resultados1000=c( )
for(i in 1:1000)
{
  set.seed(i)
  divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
  dados_treino=subset(dados,divisao==TRUE)
  dados_teste=subset(dados,divisao==FALSE)
  classificador7=h2o.deeplearning(y='Suspeitos',training_frame = as.h2o(dados_treino),
                                  activation='Rectifier',
                                  hidden=c(9),
```

```
                              epochs=1000)
  previsoes7=h2o.predict(classificador7,newdata=as.h2o(dados_teste[-17]))
  previsoes7=(previsoes7>0.5)
  previsoes7=as.vector(previsoes7)
  matriz_confusao7 = table(dados_teste[,17],previsoes7)
  t=confusionMatrix(matriz_confusao7)
  MC=t$table
  a11=MC[1,1]
  a12=MC[1,2]
  a21=MC[2,1]
  a22=MC[2,2]
  especifc=a22/(a12+a22)
  print(especific)
  resultados1000=c(resultados1000,especific)
}

for(i in 1:1000)
{
  cat(resultados1000[i])
  cat('\n')
}


#-------------------analise das medidas de desempenho-------------------

getwd()
setwd("C:/Users/maicon.galdino/Desktop")
dados <- read.table('acuracia.csv',header=T,dec='.',sep=';')
attach(dados)
RF <- dados$RF
AC <- dados$AC
knn <- dados$kNN
RL <- dados$RL
NB <- dados$NB
RNA <- dados$RNA
boxplot(dados,xlab="modelos de classificação",ylab="acuracia")
summary(dados)
sd(RF)
sd(AC)
sd(knn)
sd(RL)
sd(NB)
sd(RNA)

CV = 100*sd(RF)/mean(RF)
CV
CV = 100*sd(AC)/mean(AC)
CV
CV = 100*sd(knn)/mean(knn)
CV
CV = 100*sd(RL)/mean(RL)
CV
CV = 100*sd(NB)/mean(NB)
CV
CV=100*sd(SVM)/mean(RNA)
CV

#comparacao de diferentes numeros de arvores na RF
dados2 <- read.table('random_forest.csv',header=T,dec='.',sep=';')
```

```
attach(dados2)
RF10 <- dados2$RF10
RF20 <- dados2$RF20
RF30 <- dados2$RF30
RF40 <- dados2$RF40

boxplot(dados2,xlab="RF com diferentes número de árvores",ylab="acurácia")

summary(dados2)
sd(RF10)
sd(RF20)
sd(RF30)
sd(RF40)

#comparacao de diferentes valores de k no kNN
dados3 <- read.table('dados3.csv',header=T,dec='.',sep=';')

attach(dados3)
knn1 <- dados3$knn1
knn3 <- dados3$knn3
knn5 <- dados3$knn5
knn7 <- dados3$knn7
knn9 <- dados3$knn9

boxplot(dados3,xlab="diferentes valores de k",ylab="acurácia")

summary(dados3)
sd(knn1)
sd(knn3)
sd(knn5)
sd(knn7)
sd(knn9)

#-------------------------------------------------------------------------------------------
#Ajustando o modelo de regressão logistica para a amostra 566

getwd()
setwd("C:/Users/maicon.galdino/Desktop")

#install.packages("caret",dependencies = c("Depends","Suggests"))
library(caTools)
#leitura dos dados----------------------------------------------
dados <- read.table('tesemaicon_1.csv',header=T,dec=',',sep=';')

# processamento dos dados-----------------------------------
dados$Queixa = factor(dados$Queixa,levels = c(0,1))

#codificacao da classe------------------------
dados$Suspeitos = factor(dados$Suspeitos,levels = c(0,1))

set.seed(566)
divisao=sample.split(dados$Suspeitos,SplitRatio=0.70)
dados_treino=subset(dados,divisao==TRUE)
dados_teste=subset(dados,divisao==FALSE)
fit.model = glm(formula = Suspeitos ~ ., family = binomial, data = dados_treino)
summary (fit.model)



probabilidades = predict(fit.model, type = 'response', newdata = dados_teste[-5])
```

```
previsoes = ifelse(probabilidades > 0.5, 1, 0)
matriz_confusao = table(dados_teste[,5], previsoes)
print(matriz_confusao)
MC=matriz_confusao
MC
a11=MC[1,1]
a12=MC[1,2]
a21=MC[2,1]
a22=MC[2,2]
acuracia=(a11+a22)/74
print(acuracia)
sens=a11/(a11+a21)
print(sens)
especif=a22/(a12+a22)
print(especif)


#---------------------------------------------------------------------------

par(mfrow=c(1,1))
X <- model.matrix(fit.model)
n <- nrow(X)
p <- ncol(X)
w <- fit.model$weights
W <- diag(w)
H <- solve(t(X)%*%W%*%X)
H <- sqrt(W)%*%X%*%H%*%t(X)%*%sqrt(W)
h <- diag(H)
td <- resid(fit.model,type="deviance")/sqrt(1-h)
e <- matrix(0,n,100)
#
for(i in 1:100){
  dif <- runif(n) - fitted(fit.model)
  dif[dif >= 0 ] <- 0
  dif[dif<0] <- 1
  nresp <- dif
  fit <- glm(nresp ~ X, family=binomial)
  w <- fit$weights
  W <- diag(w)
  H <- solve(t(X)%*%W%*%X)
  H <- sqrt(W)%*%X%*%H%*%t(X)%*%sqrt(W)
  h <- diag(H)
  e[,i] <- sort(resid(fit,type="deviance")/sqrt(1-h))}
#
e1 <- numeric(n)
e2 <- numeric(n)
#
for(i in 1:n){
  eo <- sort(e[i,])
  e1[i] <- (eo[2]+eo[3])/2
  e2[i] <- (eo[97]+eo[98])/2}
#
med <- apply(e,1,mean)
faixa <- range(td,e1,e2)
par(pty="s")
qqnorm(td,xlab="Percentil da N(0,1)",
       ylab="Componente do Desvio", ylim=faixa, pch=16, main="")
#
par(new=TRUE)
#
```

```
qqnorm(e1,axes=F,xlab="",ylab="",type="l",ylim=faixa,lty=1, main="")
par(new=TRUE)
qqnorm(e2,axes=F,xlab="",ylab="", type="l",ylim=faixa,lty=1, main="")
par(new=TRUE)
qqnorm(med,axes=F,xlab="", ylab="", type="l",ylim=faixa,lty=2, main="")
#---------------------------------------------------------#
```