FACULDADE FIA DE ADMINISTRAÇÃO E NEGÓCIOS

Fundação Instituto de Administração

Elen Caroline do Couto Cosin

ANÁLISE ESTATÍSTICA DOS RISCOS DA DOENÇA RENAL CRÔNICA

São Paulo

2019

Elen Caroline do Couto Cosin

ANÁLISE ESTATÍSTICA DOS RISCOS DA DOENÇA RENAL CRÔNICA

Monografia apresentada à Faculdade FIA de Administração e Negócios mantida pela Fundação Instituto de Administração como requisito para obtenção do certificado de conclusão do curso de Pós-Graduação "Lato Sensu" Especialização em Análise de Big Data.

Orientadores: Profª. Drª. Alessandra de Ávila Montini, Prof. Dr. Adolpho Walter Pimazoni Canton

São Paulo

2019

FOLHA DE APROVAÇÃO

Elen Caroline do Couto Cosin

ANÁLISE ESTATÍSTICA DOS RISCOS DA DOENÇA RENAL CRÔNICA

___/___/___

Banca examinadora:

_____

Profº. Dr. Orientador

_____

Profº. Dr.

_____

Profº. Dr.

Julgamento: _____ Assinatura: _____

RESUMO

A Doença Renal Crônica (DRC) consiste em lesão renal e geralmente perda progressiva e irreversível da função dos rins. A principal função do rim é remover os resíduos e o excesso de água do organismo. A Doença Renal Crônica reduz essa capacidade, por pelo menos três meses, e é classificada em seis estágios, conforme a perda renal. Na maior parte do tempo de sua evolução, é assintomática, fazendo com que o diagnóstico seja feito tardiamente precisando passar pelo procedimento de hemodiálise. Em muitos casos, o diagnóstico precoce e o tratamento da doença nas suas fases iniciais podem ajudar a prevenir que a doença progrida para fases mais avançadas, quando existe a necessidade de tratamento com hemodiálise ou transplante de rim. Neste trabalho será feita uma análise utilizando dados de pacientes que foram coletados através do teste de urina em fita (*urine test strip*), exame de sangue e histórico médico, com objetivo de predizer se um determinado paciente pode ter ou não doença renal crônica.

Palavras-chave: Doença Renal Crônica. Hemodiálise. Transplante de rim. Estatística. *Machine Learning*.

ABSTRACT

Chronic Kidney Disease (CKD) consists of kidney damage and usually progressive and irreversible loss of kidney function. The main function of the kidney is to remove waste and excess water from the body. Chronic Kidney Disease leads to a reduction in this capacity for at least three months and is classified into six stages according to renal loss.

Most of the time of its evolution, it is asymptomatic, causing the diagnosis to be made late needing to undergo the hemodialysis procedure. In many cases, early diagnosis and treatment of the disease in its early stages may help to prevent the disease from progressing to more advanced stages when hemodialysis or kidney transplantation treatment is required. In this study an analysis will be made using patient data that were collected through urine test strip, blood test and medical history, in order to predict whether or not a particular patient may have chronic kidney disease.

Keywords: Chronic Kidney Disease. Hemodialysis. Kidney transplantation. Statistic. Machine learning.

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE GRÁFICOS

1 INTRODUÇÃO

A doença renal crônica consiste em lesão renal e perda progressiva e irreversível da função dos rins. Em sua fase mais avançada (chamada de fase terminal de insuficiência renal crônica), os rins não conseguem mais manter a normalidade do meio interno do paciente (ROMÃO JUNIOR, 2018).

No Brasil, a prevalência de pacientes mantidos em programa crônico de diálise, mais que dobrou nos últimos oito anos. De 24.000 pacientes mantidos em programa dialítico em 1994, em 2004 o número de pacientes aumentou para 59.153. A incidência de novos pacientes cresce cerca de 8% ao ano, tendo sido 18.000 pacientes em 2001. O gasto com o programa de diálise e transplante renal no Brasil situa-se ao redor de 1,4 bilhões de reais ao ano (ROMÃO JUNIOR, 2018).

A detecção precoce da doença renal e condutas terapêuticas apropriadas para o retardamento de sua progressão pode reduzir o sofrimento dos pacientes e os custos financeiros associados à esta doença. Como as duas principais causas de insuficiência renal crônica são a hipertensão arterial e o diabetes, são os médicos clínicos gerais que trabalham na área de atenção básica à saúde que cuidam destes pacientes. Ao mesmo tempo, os portadores de disfunção renal leve apresentam quase sempre evolução progressiva, insidiosa e assintomática, dificultando o diagnóstico precoce da disfunção renal. Assim, a capacitação, a conscientização e vigilância do médico de cuidados primários à saúde são essenciais para o diagnóstico e encaminhamento precoce ao nefrologista e a instituição de diretrizes apropriadas para retardar a progressão da doença renal crônica, prevenir suas complicações, modificar comorbidades presentes e preparo adequado a uma terapia de substituição renal (ROMÃO JUNIOR, 2018).

A importância da identificação da enfermidade não se restringe somente ao acesso à terapia renal substitutiva. O adequado diagnóstico precoce e tratamento permite reduzir complicações e mortalidade cardiovasculares. Tais metas são desafiadoras onde o acesso aos serviços de saúde é limitado, com número reduzido de médicos especializados nas doenças do sistema urinário para o acompanhamento (MARINHO, 2017).

Em países desenvolvidos, o rastreamento estima prevalência de doença renal crônica entre 10 e 13% na população adulta17-19. Nos países em desenvolvimento, dados de prevalência são limitados e heterogêneos. No Brasil, estimativas da prevalência dessa enfermidade são incertas. O conhecimento da prevalência da doença renal crônica entre os brasileiros subsidiaria melhor o planejamento de ações preventivas e assistenciais (MARINHO, 2017).

A doença renal crônica é dividida em seis estágios funcionais, de acordo com o grau de função renal do paciente, são eles: 1) Fase de função renal normal sem lesão renal; 2) fase de lesão com função renal normal; 3) fase de insuficiência renal funcional ou leve; 4) fase de insuficiência renal laboratorial ou moderada; 5) fase de insuficiência renal clínica ou severa e 6) fase terminal de insuficiência renal crônica (ROMÃO JUNIOR, 2018).

1.1 Objetivo

O objetivo desse trabalho é estudar o efeito de fatores relativos à saúde do paciente com doença crônica renal e estudar modelos estatísticos que possam explicar tais relações.

Faz parte dos objetivos deste estudo a aplicação de métodos de *machine learning* com o objetivo de predizer a ocorrência de doença crônica renal, dado as variáveis explicativas presentes na base de dados.

1.2 Organização do trabalho

Nos próximos capítulos serão apresentados estudos estatísticos sobre a doença crônica renal assim como a aplicação de modelos de *machine learning* com o intuito de predizer se um indivíduo apresenta ou não apresenta a doença com base nos dados contidos nos exames de sangue e de urina.

2 DESCRIÇÃO DA BASE DE DADOS E DAS VARIÁVEIS

Os dados foram obtidos através dos pesquisadores de *Alagappa University* da Índia no ano de 2014 (RUBINI 2014). Com dados de 400 pacientes coletados através do teste de urina em fita (*urine test strip*), exame de sangue e histórico médico, a base contém 25 variáveis separadas em qualitativas e quantitativas.

2.1 Variáveis qualitativas

As variáveis qualitativas (ou categóricas) são as características que não possuem valores quantitativos, mas, ao contrário, são definidas por categorias, ou seja, representam uma classificação dos indivíduos (GRADUANDO, 2012). Das 25 variáveis 11 foram identificadas qualitativas e estão descritas abaixo na tabela 1.

Tabela 1: Variáveis Qualitativas

variável	Descrição
rbc	Células vermelhas, normais ou não.
pc	Presença de células de pus, presente ou não.
pcc	Aglomerados de pus, presente ou não.
ba	Bactérias, presente ou não.
htn	Paciente hipertenso, sim ou não.
dm	Paciente diabético, sim ou não.
cad	Doença coronária arterial, sim ou não.
appet	Apetite bom ou ruim.
pe	Presença de edema nos pés, sim ou não.
ane	Anemia, sim ou não.
class	Variável resposta que corresponde ao diagnóstico do paciente em doente crônico dos rins ou não.

Fonte: Desenvolvido pela autora

2.2 Variáveis quantitativas

As variáveis quantitativas são características que podem ser descritas por números (GRADUANDO, 2012). Das 25 variáveis 14 foram identificadas quantitativas e estão descritas abaixo na tabela 2.

Tabela 2: Variáveis Quantitativas

variável	Descrição
age	Idade em anos do indivíduo.
bp	Pressão do sangue em mm/Hg.
sg	Gravidade específica, teste que compara a densidade da urina com a da água. Quanto maior o valor, maior a desidratação. (1.005, 1.010, 1.015, 1.020 ou 1.025).
al	Medida da quantidade de proteína albumina (0,1,2,3,4,5).
su	Medida de quantidade de açúcar na urina (0,1,2,3,4,5).
bgr	Glicose no sangue aleatório em mg/dl.
bu	Ureia no sangue em mg/dl.
sc	Presença de creatinina em mg/dl.
sod	Quantidade de sódio em mEq/L.
pot	Quantidade de potássio em mEq/L.
hemo	Quantidade de hemoglobina em gramas.
pcv	Hematócrito, volume de glóbulos vermelhos.
wbcc	Contagem de glóbulos brancos em células/cumm.
rbcc	Contagem de glóbulos vermelhos em milhões/cmm.

Fonte: Desenvolvido pela autora

Podemos definir variável como a característica que é medida ou avaliada em cada elemento da amostra ou população (GRADUANDO, 2012).

Em resumo, é aquilo que está sendo avaliado no seu ensaio/experimento. Como o próprio nome diz, seus valores variam de elemento para elemento, identificar o tipo das variáveis é uma etapa trivial para a análise de dados.

3 ANÁLISE EXPLORATÓRIA

Através das linguagens de programação R e PYTHON, foram feitas análises descritivas dos dados.

3.1 Medidas de posição

As medidas de posição são as estatísticas que representam uma série de dados orientando-nos quanto à posição da distribuição em relação ao eixo horizontal do gráfico da curva de frequência (TEIXEIRA, 2018). Na Tabela 3 abaixo temos as medidas de posição da base.

Tabela 3: Medidas de posição

variável	quantidade de não nulo	média	min	1° quartil	2° quartil	3° quartil	max
age	391,00	51,48	2,00	42,00	55,00	64,50	90,00
bp	388,00	76,47	50,00	70,00	80,00	80,00	180,00
sg	353,00	1,02	1,01	1,01	1,02	1,02	1,03
al	354,00	1,02	0,00	0,00	0,00	2,00	5,00
su	351,00	0,45	0,00	0,00	0,00	0,00	5,00
bgr	356,00	148,04	22,00	99,00	121,00	163,00	490,00
bu	381,00	57,43	1,50	27,00	42,00	66,00	391,00
sc	383,00	3,07	0,40	0,90	1,30	2,80	76,00
sod	313,00	137,53	4,50	135,00	138,00	142,00	163,00
pot	312,00	4,63	2,50	3,80	4,40	4,90	47,00
hemo	348,00	12,53	3,10	10,30	12,65	15,00	17,80
pcv	329,00	38,88	9,00	32,00	40,00	45,00	54,00
wbcc	294,00	8.406,12	2.200,00	6.500,00	8.000,00	9.800,00	26.400,00
rbcc	269,00	4,71	2,10	3,90	4,80	5,40	8,00

Fonte: Desenvolvido pela autora

3.2 Medidas de dispersão

As medidas de dispersão são amplitude, desvio, variância e desvio padrão e são usadas para determinar o grau de variação dos números de uma lista com relação à média. (TEIXEIRA, 2018). Na Tabela 4 abaixo temos as medidas de dispersão da base.

Tabela 4: Medidas de dispersão

variável	desvio padrão	amplitude	variância	coef. de variação
age	17,17	88,00	294,80	33,35
bp	13,68	130,00	187,24	17,89
sg	0,01	0,02	0,00	0,98
al	1,35	5,00	1,83	132,35
su	1,10	5,00	1,21	244,44
bgr	79,28	468,00	6.285,59	53,55
bu	50,50	389,50	2.550,55	87,93
sc	5,74	75,60	32,96	186,97
sod	10,41	158,50	108,34	7,57
pot	3,19	44,50	10,20	68,9
hemo	2,91	14,70	8,48	23,22
pcv	8,99	45,00	80,82	23,12
wbcc	2.944,47	24.200,00	8.669.928,26	35,03
rbcc	1,03	5,90	1,05	21,87

Fonte: Desenvolvido pela autora

3.3 Tabela de frequência

Tanto os dados qualitativos quanto os quantitativos podem e devem ser agrupados em frequências para construir uma tabela. As frequências associadas aos dados constituem a distribuição de frequência. Uma tabela é constituída por dados organizados em linhas e colunas. A frequência de um dado é o número de ocorrências ou repetições de um dado (PORTAL EDUCAÇÃO, 2017). Na Tabela 5 temos a frequência da base.

Tabela 5: Tabela de frequência

Apresenta doença crônica renal?	Frequência Absoluta	Frequência Relativa
Sim	250	62,50%
Não	150	37,50%

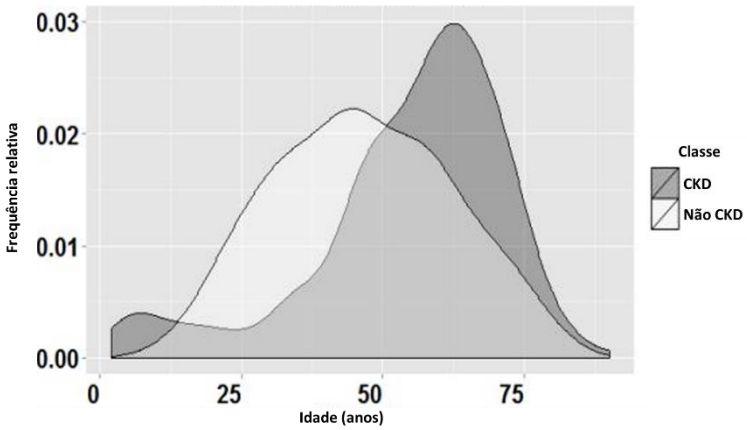
Fonte: Desenvolvido pela autora

3.4 Gráficos de densidade

Nesta parte do estudo, tem-se a apresentação dos gráficos de densidade das variáveis contínuas e gráficos de barras para as discretas, que são úteis para justificar a modelagem na seção seguinte. Nas densidades, as curvas com área esbranquiçada são relativas ao grupo sem doença (Não CKD) e a área acinzentada ao grupo com doença (CKD).

O primeiro gráfico a ser apresentado é o gráfico de densidade de idade que pode ser visualizado no Gráfico 1:

Gráfico 1: Densidade de idade (age)

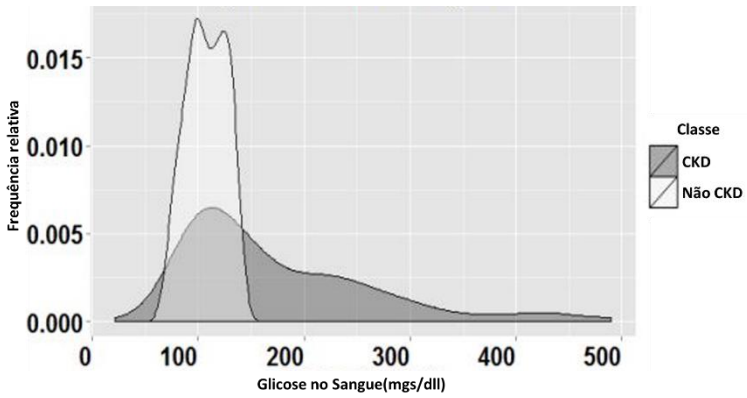


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que a ocorrência de pessoas diagnosticadas com doença crônica renal aumenta de acordo com o aumento da idade do indivíduo.

O segundo gráfico a ser apresentado é o gráfico de densidade de glicose no sangue que pode ser visualizado abaixo no Gráfico 2:

Gráfico 2: Densidade de glicose no sangue (bgr)

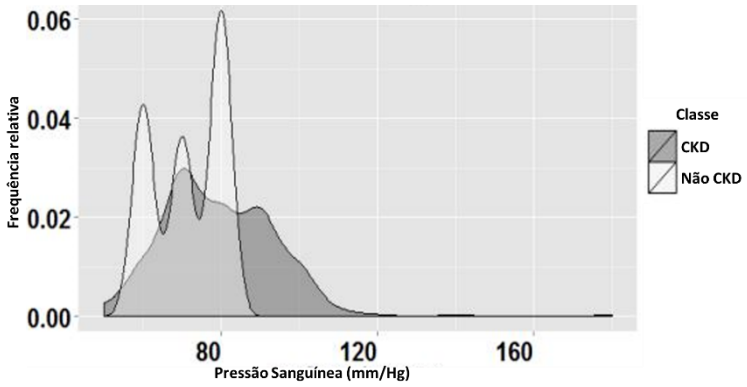


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que a maior parte das pessoas diagnosticadas sem doença crônica renal estão com taxa de glicose no sangue aproximadamente entre 50(mgs/dll) e 150 (mgs/dll), as pessoas diagnosticadas com doença crônica renal estão com uma variação muito maior de taxa de glicose no sangue apresentada pelas saudáveis.

O terceiro gráfico a ser apresentado é o gráfico de densidade de pressão sanguínea que pode ser visualizado abaixo no Gráfico 3:

Gráfico 3: Densidade de pressão sanguínea

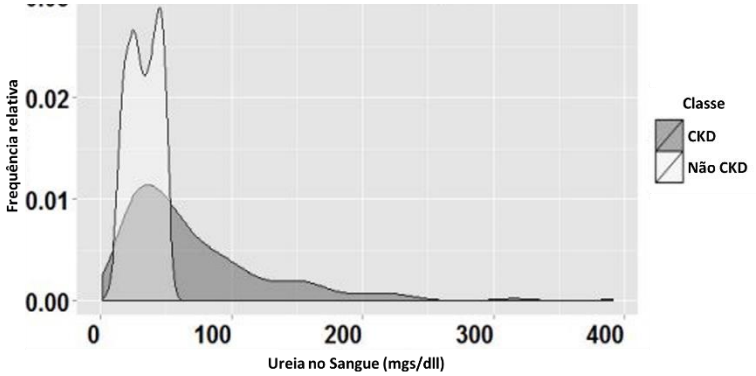


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que a maior parte das pessoas diagnosticadas sem doença crônica renal possuem uma taxa de 80 (mm/Hg) de pressão sanguínea e que as pessoas diagnosticadas com doença crônica renal possuem a taxa de pressão sanguínea menor que 80 (mm/Hg).

O quarto gráfico a ser apresentado é o gráfico de densidade de ureia no sangue que pode ser visualizado abaixo no Gráfico 4:

Gráfico 4: Densidade de ureia no sangue (bu)



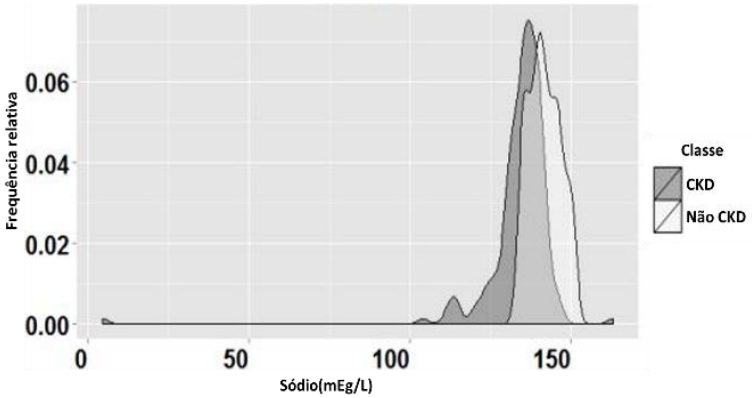
Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que pacientes diagnosticados sem doença crônica renal possuem uma taxa de ureia no sangue entre 20 (mgs/dll) e 50 (mgs/dll) e que pacientes

diagnosticados com doença crônica renal possuem uma taxa de ureia no sangue entre 30 (mgs/dll) e 40 (mgs/dll).

O quinto gráfico a ser apresentado é o gráfico de densidade de Sódio que pode ser visualizado abaixo no Gráfico 5:

Gráfico 5: Densidade de Sódio (sod)

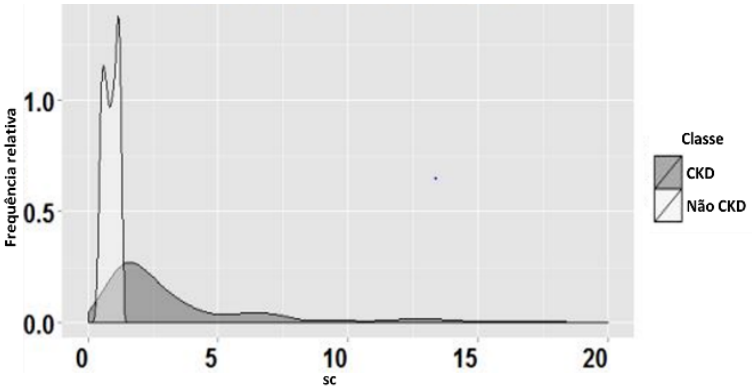


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que pacientes diagnosticados com e sem doença crônica renal possuem a taxa de sódio entre 130 (mEg/L) e 140 (mEg/L).

O sexto gráfico a ser apresentado é o gráfico de densidade de creatinina no sangue que pode ser visualizado abaixo no Gráfico 6:

Gráfico 6: Densidade de creatinina no sangue (sc)

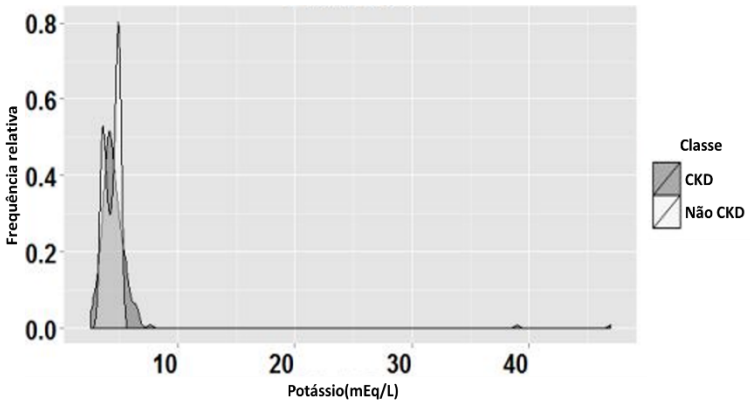


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que pacientes diagnosticados sem doença crônica renal possuem a taxa de creatinina no sangue entre 0,5 e 1,5 e pacientes diagnosticados com doença crônica renal possuem a taxa de creatinina varia entre 0,5 e 2.

O sétimo gráfico a ser apresentado é o gráfico de densidade de Potássio que pode ser visualizado abaixo no Gráfico 7:

Gráfico 7: Densidade de Potássio (bp)

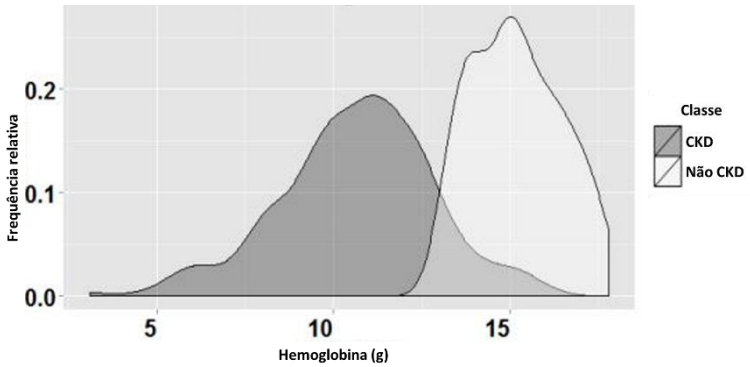


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que pacientes diagnosticados sem doença crônica renal possuem a taxa de potássio entre 4,5 e 6 e pacientes diagnosticados com doença crônica renal possuem a taxa de creatinina varia entre 2 e 3.

O oitavo gráfico a ser apresentado é o gráfico de densidade de Hemoglobina que pode ser visualizado abaixo no Gráfico 8:

Gráfico 8: Densidade de hemoglobina (hemo)

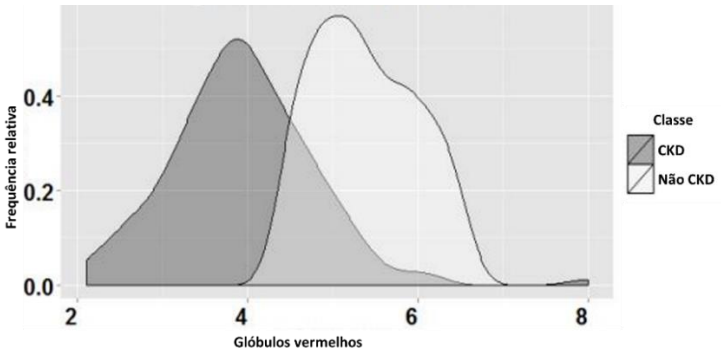


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que pacientes diagnosticados sem doença crônica renal possuem a taxa de hemoglobina entre 11 (g) e 18 (g) e pacientes diagnosticados com doença crônica renal possuem a taxa de hemoglobina entre 5 (g) e 13 (g) tendo um pico maior nos pacientes com taxa de hemoglobina entre 10 (g) e 11,5 (g).

O nono gráfico a ser apresentado é o gráfico de densidade de Glóbulos Vermelhos que pode ser visualizado abaixo no Gráfico 9:

Gráfico 9: Densidade de glóbulos vermelhos (rbcc)

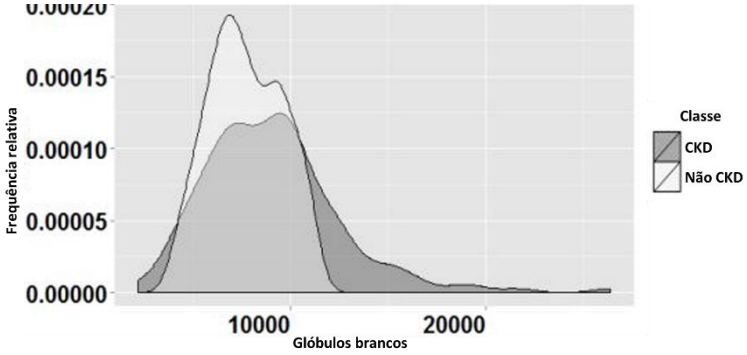


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que a maior parte dos pacientes diagnosticados sem doença crônica renal possuem taxa de glóbulos vermelhos entre 5 e 6, e existe um grande volume de pacientes diagnosticados com doença crônica renal com a taxa de glóbulos vermelhos entre 3,75 e 4,3.

O décimo gráfico a ser apresentado é o gráfico de densidade de Glóbulos brancos que pode ser visualizado abaixo no Gráfico 10:

Gráfico 10: Densidade de glóbulos brancos (wbcc)

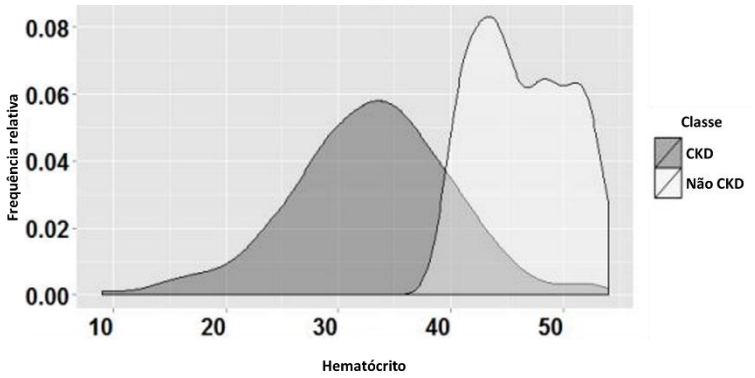


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que a maior parte dos pacientes diagnosticados sem doença crônica renal possuem taxa de glóbulos brancos entre 7000 e 8000 e a maior parte de pacientes diagnosticados com doença crônica renal possuam taxa de glóbulos brancos entre 8000 e 10000.

O décimo primeiro gráfico a ser apresentado é o gráfico de densidade de hematócrito que pode ser visualizado abaixo no Gráfico 11:

Gráfico 11: Densidade de hematócrito

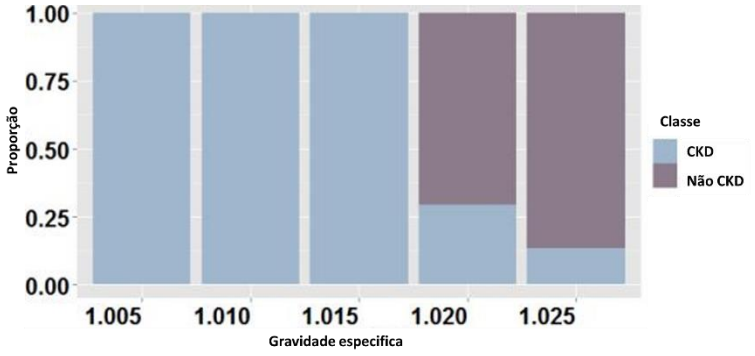


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que a maior parte dos pacientes diagnosticados sem doença crônica renal possuem taxa de hematócrito entre 42 e 43,5 e a maior concentração de pacientes diagnosticados com doença crônica renal possuam taxa de hematócrito entre 33 e 35.

O décimo segundo gráfico a ser apresentado é o gráfico de frequências de gravidade específica que pode ser visualizado abaixo no Gráfico 12:

Gráfico 12: Frequências de Gravidade Específica (sg)

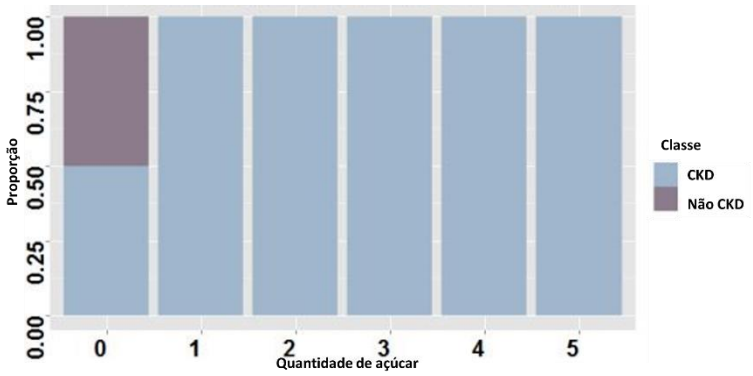


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que a maior concentração de pessoas diagnosticadas com doença crônica renal possui frequências de gravidade específica entre 1.005 e 1.015 e que pessoas diagnosticadas sem doença crônica renal possuem frequências de gravidade específica entre 1.020 e 1.025.

O décimo terceiro gráfico a ser apresentado é o gráfico de frequências de açúcar que pode ser visualizado abaixo no Gráfico 13:

Gráfico 13: Frequências de Açúcar (su)

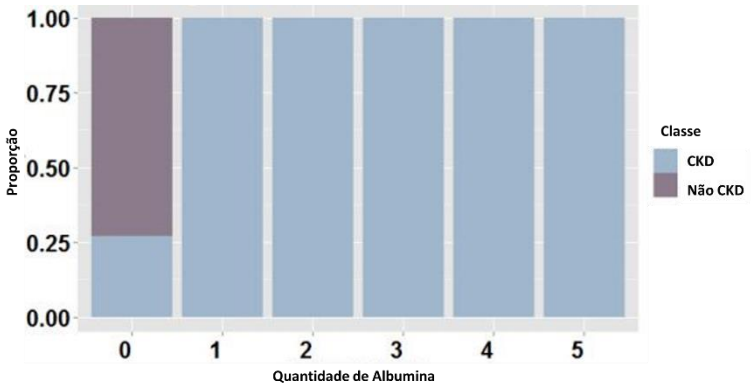


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que a maior concentração de pessoas diagnosticadas com doença crônica renal possui frequências de açúcar maior que 0 e que todas as pessoas diagnosticadas sem doença crônica renal possuem frequências de açúcar igual a 0.

O décimo quarto gráfico a ser apresentado é o gráfico de frequências de albumina que pode ser visualizado abaixo no Gráfico 14:

Gráfico 14: Frequências de Albumina (al)

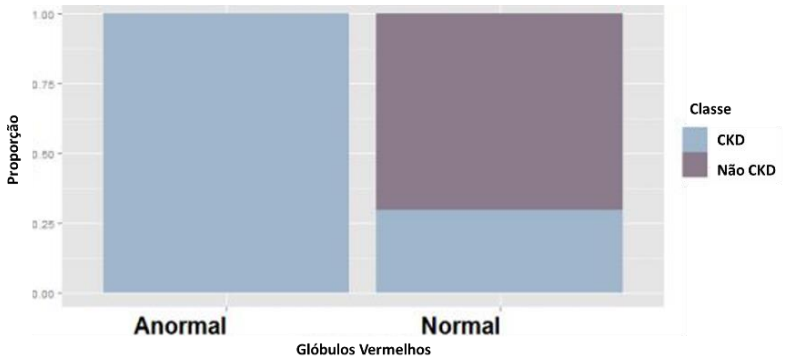


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que a maior concentração de pessoas diagnosticadas com doença crônica renal possui frequências de albumina maior que 0 e que todas as pessoas diagnosticadas sem doença crônica renal possuem frequências de albumina igual a 0.

O décimo quinto gráfico a ser apresentado é o gráfico de frequências de condição dos glóbulos vermelhos que pode ser visualizado abaixo no Gráfico 15:

Gráfico 15: Frequências de condição de glóbulos vermelhos (rbc)

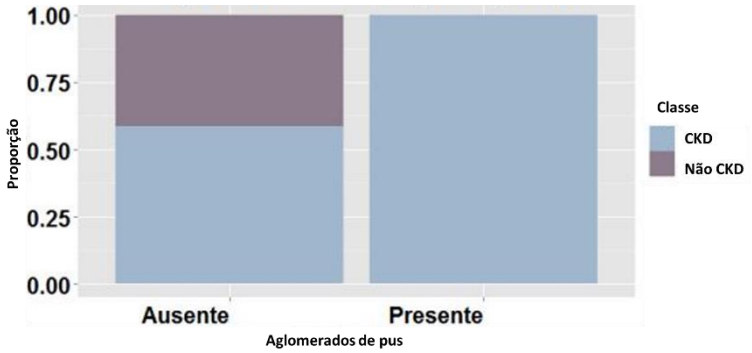


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que a maior concentração de pessoas diagnosticadas com doença crônica renal possui frequências de condição dos glóbulos vermelhos anormal e que todas as pessoas diagnosticadas sem doença crônica renal possuem frequências de condição dos glóbulos vermelhos normal.

O décimo sexto gráfico a ser apresentado é o gráfico de frequências de aglomerados de pus que pode ser visualizado abaixo no Gráfico 16:

Gráfico 16: Frequências de aglomerados de pus (pcc)

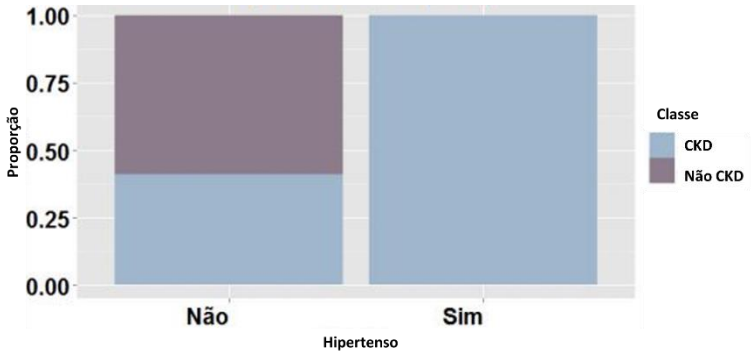


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que pessoas diagnosticadas com doença crônica renal possui frequências de aglomerados de pus presente ou ausente e que todas as pessoas diagnosticadas sem doença crônica renal possuem frequências de aglomerados de pus ausente.

O décimo sétimo gráfico a ser apresentado é o gráfico de frequência de hipertensão que pode ser visualizado abaixo no Gráfico 17:

Gráfico 17: Frequência de hipertensão (htn)

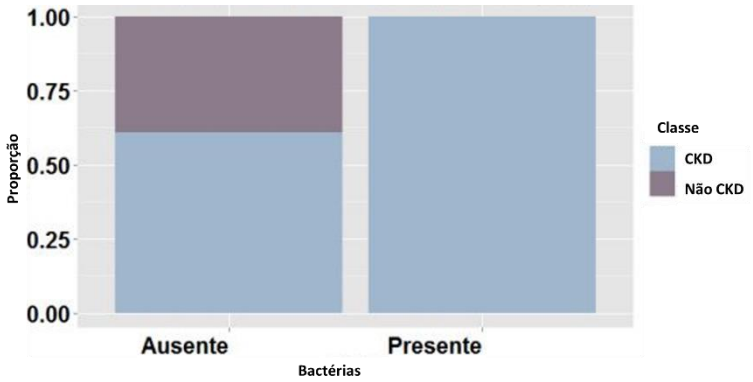


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que a maior concentração de pessoas diagnosticadas com doença crônica renal possui frequência de hipertensão igual a sim e que todas as pessoas diagnosticadas sem doença crônica renal possuem frequência de hipertensão igual a não.

O décimo oitavo gráfico a ser apresentado é o gráfico de frequências de bactérias que pode ser visualizado abaixo no Gráfico 18:

Gráfico 18: Frequências de bactérias (ba)

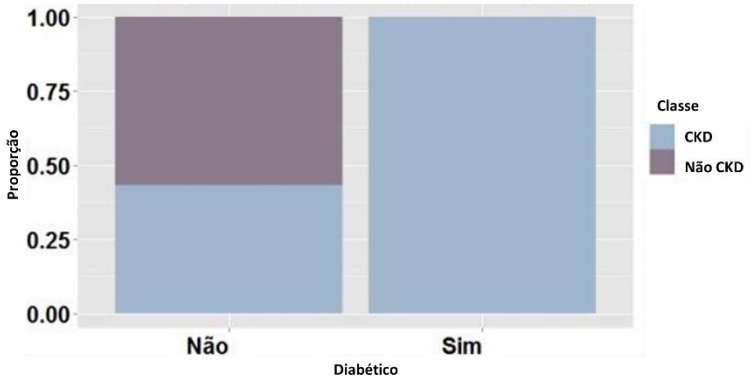


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que pessoas diagnosticadas com doença crônica renal possui frequências de bactérias igual a ausente ou presente e que todas as pessoas diagnosticadas sem doença crônica renal possuem frequências de bactérias igual a ausente.

O décimo nono gráfico a ser apresentado é o gráfico de frequências de diabetes que pode ser visualizado abaixo no Gráfico 19:

Gráfico 19: Frequências de diabetes (dm)

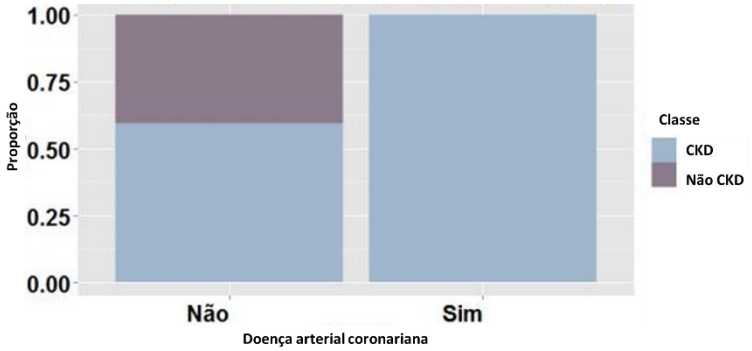


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que a maior concentração de pessoas diagnosticadas com doença crônica renal possui frequências de diabetes igual a sim e que todas as pessoas diagnosticadas sem doença crônica renal possuem frequências de diabetes igual a não.

O vigésimo gráfico a ser apresentado é o gráfico de frequências de doença arterial coronariana que pode ser visualizado abaixo no Gráfico 20:

Gráfico 20: Frequências de doença arterial coronariana (cad)

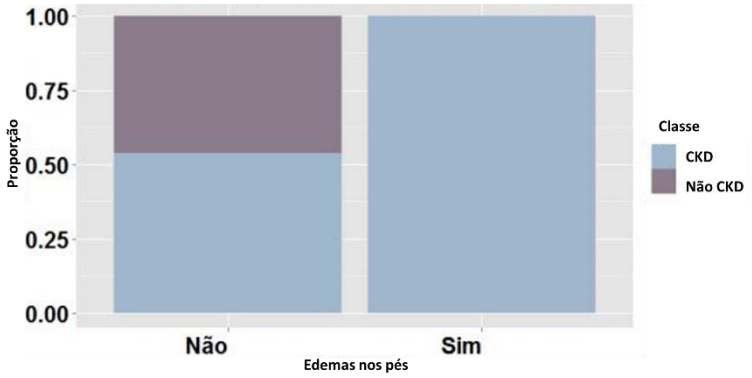


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que pessoas diagnosticadas com doença crônica renal possui frequências de doença arterial coronariana igual a sim ou não e que todas as pessoas diagnosticadas sem doença crônica renal possuem frequências de doença arterial coronariana igual a não.

O vigésimo primeiro gráfico a ser apresentado é o gráfico de frequências de edemas nos pés que pode ser visualizado abaixo no Gráfico 21:

Gráfico 21: Frequências de edemas nos pés (su)

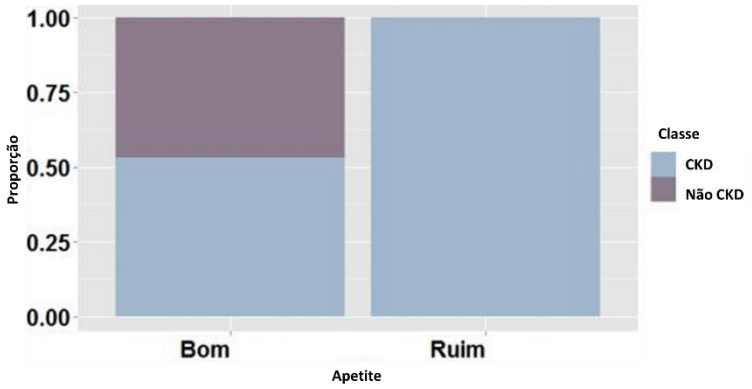


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que pessoas diagnosticadas com doença crônica renal possui frequências de edemas nos pés igual a sim ou não e que todas as pessoas diagnosticadas sem doença crônica renal possuem frequências de edemas nos pés igual a não.

O vigésimo segundo gráfico a ser apresentado é o gráfico de frequências de apetite que pode ser visualizado abaixo no Gráfico 22:

Gráfico 22: Frequências de apetite (appet)

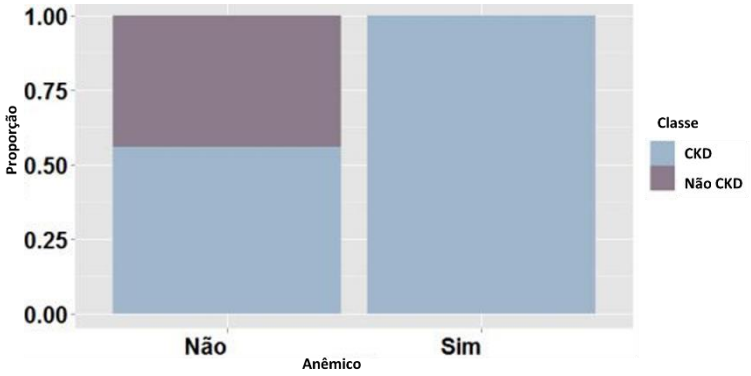


Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que pessoas diagnosticadas com doença crônica renal possui frequências de apetite igual a bom ou ruim e que todas as pessoas diagnosticadas sem doença crônica renal possuem frequências de apetite igual a bom.

O vigésimo terceiro gráfico a ser apresentado é o gráfico de frequências de anemia que pode ser visualizado abaixo no Gráfico 23:

Gráfico 23: Frequências de anemia (ane)



Fonte: Desenvolvido pela autora

Neste gráfico é possível observar que pessoas diagnosticadas com doença crônica renal possui frequências de anemia igual a não ou sim e que todas as pessoas diagnosticadas sem doença crônica renal possuem frequências de apetite igual a não.

3.5 Correlação entre as variáveis

Nesta parte do estudo, tem-se a apresentação da correlação de Pearson entre as variáveis contínuas. Em probabilidade e estatística, correlação, dependência ou associação é qualquer relação estatística (causal ou não causal) entre duas variáveis e correlação é qualquer relação dentro de uma ampla classe de relações estatísticas que envolva dependência entre duas variáveis (BUSSAB, 2010).

O coeficiente de correlação de Pearson (r) ou coeficiente de correlação produto-momento ou o r de Pearson mede o grau da correlação linear entre duas variáveis quantitativas. É um índice adimensional com valores situados ente -1,0 e 1.0 inclusive, que reflete a intensidade de uma relação linear entre dois conjuntos de dados (CARMO, 2010).

A figura 1 apresenta o quadro de correlação entre as variáveis, que foi gerada através da função CORR da biblioteca pandas da linguagem de programação Python, combinada com a biblioteca *matplotlib* para gerar uma visualização mais adequada, o código utilizado pode ser visualizado no apêndice F.

Figura 1: Quadro de correlação entre as variáveis

	age	bp	sg	al	su	bgr	bu	sc	sod	pot	hemo	pcv	wbcc	rbcc
age	1.0	0.16	-0.19	0.12	0.22	0.24	0.2	0.13	-0.1	0.058	-0.19	-0.24	0.12	-0.27
bp	0.16	1.0	-0.22	0.16	0.22	0.16	0.19	0.15	-0.12	0.075	-0.31	-0.33	0.03	-0.26
sg	-0.19	-0.22	1.0	-0.47	-0.3	-0.37	-0.31	-0.36	0.41	-0.073	0.6	0.6	-0.24	0.58
al	0.12	0.16	-0.47	1.0	0.27	0.38	0.45	0.4	-0.46	0.13	-0.63	-0.61	0.23	-0.57
su	0.22	0.22	-0.3	0.27	1.0	0.72	0.17	0.22	-0.13	0.22	-0.22	-0.24	0.18	-0.24
bgr	0.24	0.16	-0.37	0.38	0.72	1.0	0.14	0.11	-0.27	0.067	-0.31	-0.3	0.15	-0.28
bu	0.2	0.19	-0.31	0.45	0.17	0.14	1.0	0.59	-0.32	0.36	-0.61	-0.61	0.05	-0.58
sc	0.13	0.15	-0.36	0.4	0.22	0.11	0.59	1.0	-0.69	0.33	-0.4	-0.4	-0.0064	-0.4
sod	-0.1	-0.12	0.41	-0.46	-0.13	-0.27	-0.32	-0.69	1.0	0.098	0.37	0.38	0.0073	0.34
pot	0.058	0.075	-0.073	0.13	0.22	0.067	0.36	0.33	0.098	1.0	-0.13	-0.16	-0.11	-0.16
hemo	-0.19	-0.31	0.6	-0.63	-0.22	-0.31	-0.61	-0.4	0.37	-0.13	1.0	0.9	-0.17	0.8
pcv	-0.24	-0.33	0.6	-0.61	-0.24	-0.3	-0.61	-0.4	0.38	-0.16	0.9	1.0	-0.2	0.79
wbcc	0.12	0.03	-0.24	0.23	0.18	0.15	0.05	-0.0064	0.0073	-0.11	-0.17	-0.2	1.0	-0.16
rbcc	-0.27	-0.26	0.58	-0.57	-0.24	-0.28	-0.58	-0.4	0.34	-0.16	0.8	0.79	-0.16	1.0

Fonte: Desenvolvido pela autora

Neste quadro é possível identificar as variáveis que apresentam correlações mais fortes e mais fracas, quando mais próximo a 1 for a correlação entre duas variáveis, mais próximo ao vermelho a representação gráfica esta e quanto mais distante de 1 for a correlação entre duas variáveis, mais próximo ao azul a representação gráfica está. Desta forma pode-se ler que as cores quentes representam uma correlação maior e as cores frias representam a falta de correlação.

4 ANÁLISE ESTATÍSTICA

Através da regressão logística, veremos o efeito de cada fator levando em conta outras variáveis e escolher as variáveis que melhor ajustam o modelo.

4.1 Regressão logística

A regressão logística é uma técnica estatística que tem como objetivo produzir, a partir de um conjunto de observações, um modelo que permita a predição de valores tomados por uma variável categórica, frequentemente binária, a partir de uma série de variáveis explicativas contínuas e/ou binárias. O êxito da regressão logística assenta sobretudo nas numerosas ferramentas que permitem interpretar de modo aprofundado os resultados obtidos.

Em comparação com as técnicas conhecidas em regressão, em especial a regressão linear, a regressão logística distingue-se essencialmente pelo fato de a variável resposta ser categórica. Enquanto método de predição para variáveis categóricas, a regressão logística é comparável às técnicas supervisionadas propostas em aprendizagem automática (árvores de decisão, redes neurais etc.), ou ainda a análise discriminante preditiva em estatística exploratória. É possível de as colocar em concorrência para escolha do modelo mais adaptado para um certo problema preditivo a resolver.

A regressão logística tem o objetivo de projetar a probabilidade de ocorrer um evento de interesse. Por exemplo projetar as probabilidades:

• de um cliente realizar pagamentos por internet *banking*;

• de um cliente pagar o crédito que tomou emprestado;

É uma técnica utilizada para situações em que a variável resposta (dependente) é de natureza dicotômica ou binária. Quanto às independentes, tanto podem ser categóricas ou não.

A regressão logística é um recurso que permite estimar a probabilidade associada à ocorrência de determinado evento em combinação com um conjunto de variáveis explicativas.

Busca estimar a probabilidade de a variável dependente assumir um determinado valor em função dos valores conhecidos de outras variáveis. Os resultados da análise ficam contidos no intervalo de zero a um, exemplo:

Suponha que um indivíduo emprestou dinheiro para uma pessoa. Existem dois eventos possíveis desse empréstimo: ele ser pago ou não. Se tal pessoa tiver um bom salário, um emprego estável e um histórico impecável, é provável que ela pague. Agora, se ela estiver desempregada e com dívidas, a probabilidade de não pagar é maior. É isso que a regressão

logística proporciona. A variável resposta aqui é se a pessoa paga ou não o crédito, é binária: sim ou não. Na regressão linear temos uma variável resposta contínua (e.g.: o valor da dívida), na regressão logística a variável resposta é binária, 0 ou 1 (MENARD, 1995).

4.2 Máxima verossimilhança

Em estatística, a estimativa por máxima verossimilhança (*maximum-likelihood estimation- MLE*) é um método para estimar os parâmetros de um modelo estatístico. Assim, a partir de um conjunto de dados e dado um modelo estatístico, a estimativa por máxima verossimilhança estima valores para os diferentes parâmetros do modelo.

Por exemplo, alguém pode estar interessado na altura de girafas fêmeas adultas, mas devido a restrições de custo ou tempo, medir a altura de todas essas girafas de uma população pode ser impossível. Podemos assumir que as alturas são normalmente distribuídas (modelo estatístico), mas desconhecemos a média e variância (parâmetros do modelo) dessa distribuição. Esses parâmetros da distribuição podem então ser estimados por máxima verossimilhança (*maximum-likelihood estimation- MLE*) a partir da medição de uma amostra da população.

O método busca aqueles valores para os parâmetros de maneira a maximizar a probabilidade dos dados amostrados, dados o modelo assumido (no caso, distribuição normal). De maneira geral, posto um conjunto de dados e um modelo estatístico, o método de máxima verossimilhança estima os valores dos diferentes parâmetros do modelo estatístico de maneira a maximizar a probabilidade dos dados observados (isto é, busca parâmetros que maximizem a função de verossimilhança).

O método de máxima verossimilhança apresenta-se como um método geral para estimação de parâmetros, principalmente no caso de distribuições normais (MCFADDEN, 1994).

4.3 Máxima verossimilhança penalizada

Visando resolver o problema de existência dos estimadores de máxima verossimilhança na presença de separação, Heinze e Schemper (2002) sugerem a modificação da função escore para a estimação dos coeficientes do modelo de regressão logística. Originalmente, essa proposta foi desenvolvida por Firth (1993) buscando reduzir o vício das estimativas de máxima verossimilhança em modelos lineares generalizados. Ela produz estimativas finitas para os parâmetros do modelo através da estimação por máxima verossimilhança penalizada. As estimativas de máxima verossimilhança dos parâmetros da regressão são encontradas

solucionando o sistema de equações do vetor escore. No entanto, Firth (1993) sugere a estimação baseada nas equações escore modificadas dadas por:

$$U_j(\beta)^* \equiv U_j(\beta) + \frac{1}{2} traço\left[I(\beta)^{-1} \left\{ \frac{\partial I(\beta)}{\partial \beta_j} \right\} \right] = 0, \qquad j = 1,..., p+1.,$$

onde $I(\beta)^{-1}$ é a inversa da matriz de informação de Fisher avaliada em β. A função escore modificada $U_j(\beta)^*$ é relacionada à função de log-verossimilhança penalizada: $l(\beta)^* = l(\beta) + \frac{1}{2}\ln|I(\beta)|$, e à função de verossimilhança penalizada: $L(\beta)^* = L(\beta)|I(\beta)|^{\frac{1}{2}}$.

A função de penalização $|I(\beta)|^{\frac{1}{2}}$ tem influência, assintoticamente, desprezível. Utilizando esta modificação, Firth (1993) mostrou que o vício das estimativas de máxima verossimilhança é removido (GONÇALVES, 2008).

4.4 Método stepwise

O método *Stepwise* para a seleção de variáveis é muito usado em regressão linear. Qualquer procedimento para seleção ou exclusão de variáveis de um modelo é baseado em um algoritmo que checa a importância das variáveis, incluindo ou excluindo-as do modelo se baseando em uma regra de decisão. A importância da variável é definida em termos de uma medida de significância estatística do coeficiente associado à variável para o modelo. Essa estatística depende das suposições do modelo.

No *Stepwise* da regressão linear um teste F é usado desde que os erros tenham distribuição normal. Na regressão logística os erros seguem distribuição binomial e a significância é assegurada via Teste da Razão de Verossimilhança. Assim, em cada passo do procedimento a variável mais importante, em termos estatísticos, é aquela que produz a maior mudança no logaritmo da verossimilhança em relação ao modelo que não contém a variável (TEIXEIRA, 2010).

4.5 Quantil-quantil

Em estatística, um gráfico Q-Q ("Q" significa quantil) é um gráfico de probabilidades, que é um método gráfico para comparar duas distribuições de probabilidade, traçando seus quantis uns contra os outros. Primeiro, o conjunto de intervalos para os quantis é escolhido. Um

ponto (x, y) no gráfico corresponde a um dos quantis da segunda distribuição (coordenada y) plotadas contra o mesmo, mesmo quantil da primeira distribuição de (coordenada x). Portanto, a linha é uma curva paramétrica com o parâmetro que é o (número do) intervalo para quantil. Se as duas distribuições que estão sendo comparadas são semelhantes, os pontos no gráfico Q-Q vão repousar na linha y = x, aproximadamente. Se as distribuições são linearmente relacionadas, os pontos no gráfico Q-Q irão repousar em uma linha, aproximadamente, mas não necessariamente na linha y = x. Gráficos Q-Q também podem ser usados como meio gráfico de estimativa de parâmetros de dispersão e tendência central em uma família de distribuições.

Um gráfico Q-Q é usado para comparar as formas de distribuições, fornecendo uma exibição gráfica de como as propriedades, tais como medidas de tendência central, dispersão e assimetria são semelhantes ou diferentes nas duas distribuições. Gráficos Q-Q podem ser usados para comparar conjuntos de dados ou distribuições teóricas. O uso de gráficos Q-Q para comparação de duas amostras de dados pode ser visto como uma abordagem não-paramétrica para comparação de suas distribuições subjacentes (WILK, 1968).

4.6 Critério de Informação de Akaike (AIC)

O critério de informação de Akaike (AIC) é baseado no máximo da função de verossimilhança (MFV). Este critério admite a existência de um modelo “real” que descreve os dados que é desconhecido, e tenta escolher dentre um grupo de modelos avaliados. O modelo com menor valor e AIC é considerado o modelo de melhor ajuste (BOZDANGAN, 1987; WOLFINGER, 1993; LITTELL, 2002).

4.7 Método de jeffreys

Em estatística, o método de Jeffreys, regra de Jeffreys ou a priori de Jeffreys, nomeado em homenagem a Sir Harold Jeffreys é uma probabilidade a priori não informativa para um espaço de parâmetros definida como:

$$\pi_\theta(\vec{\theta}) \propto \sqrt{\det \mathcal{I}\left(\vec{\theta}\right)},$$

Onde $\mathcal{I}\left(\vec{\theta}\right)$ é a matriz de informação de Fisher e $\det$ é a função determinante.

Isto é, a priori $\pi_\theta(\vec{\theta})$ de Jeffreys é proporcional $(\propto)$ a raiz quadrada do determinante da matriz de informação de Fisher (EHLERS, 2002).

4.8 Fenômeno de separação parcial

Quando falamos em regressão logística, e principalmente quando a abordamos no domínio dos eventos raros, é fundamental falarmos do fenômeno de separabilidade, bem como do problema a ele associado.

Na regressão logística podem surgir situações em que o algoritmo para a determinação da verossimilhança convirja, mas a estimativa de um parâmetro tenda para $\pm\infty$. Normalmente, designamos este fenômeno por separação. Vulgarmente, ao nível da literatura atual, é também possível ver este fenômeno a ser referido como ”probabilidade monótona”.

De um modo geral, o fenômeno de separação está associado ao facto de a ocorrência do evento de interesse e a sua não ocorrência serem facilmente separados por um único fator de risco ou por uma combinação linear, não trivial, de fatores de risco.

A separação ocorre principalmente em amostras pequenas e/ou com várias variáveis desequilibradas (HEINZE; SCHEMPER, 2002). Entenda-se aqui variáveis desequilibradas como sendo variáveis categóricas, que quando confrontadas com a variável resposta, apresentam diferenças muito consideráveis entre as várias proporções.

Como nos mostram Heinze e Schemper no artigo *A solution to the problem of separation in logistic regression* de 2002 (HEINZE; SCHEMPER, 2002), o problema da separação não é insignificante. São vários os fatores que podem levar à existência de separabilidade nos dados. Ao nível da literatura encontram-se documentados um conjunto de fatores que conduzem à ocorrência deste fenômeno, dos quais se destacam: o tamanho da amostra, o número de variáveis binárias e a magnitude dos *odds* associados a tais variáveis.

Como é de fácil percepção, num estudo sobre eventos raros, uma das principais causas para o aparecimento do fenômeno de separabilidade está relacionado com o facto de existir um elevado desequilibro nas várias categorias das variáveis. Tais desequilíbrios conduzem ao aparecimento de *odds* com valores indesejados.

No artigo já referenciado, os autores deixam-nos ainda como proposta, para ultrapassar parte dos problemas relacionados com a separabilidade dos dados, a aplicação ao do modelo de regressão logística de Firth. Este modelo de regressão foi proposto num artigo da autoria de David Firth em 1993 (HEINZE & SCHEMPER, 2002, FIRTH, 1993).

4.9 Aplicação e testes

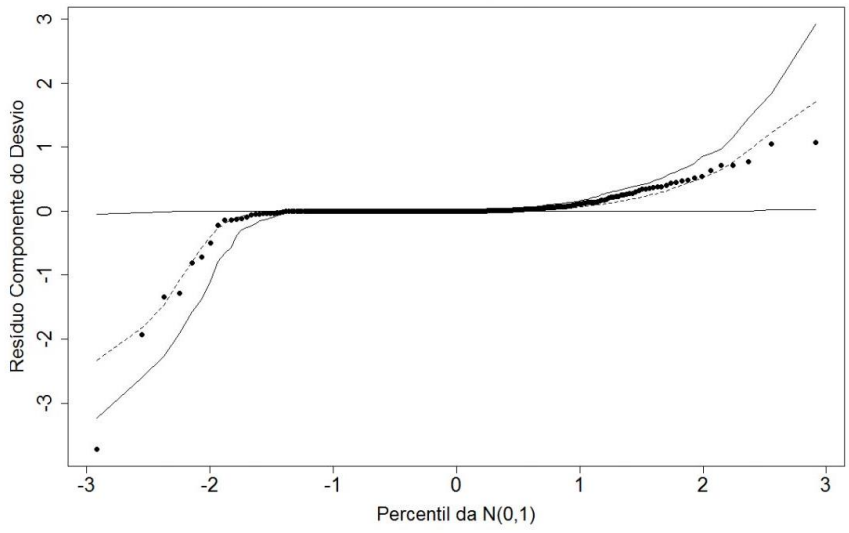
Dando andamento na análise estatística realizada, é possível observar na seção 3 que os dados apresentam o fenômeno de separação parcial. Como o método de ajuste usado na regressão logística é o Método de Máxima Verossimilhança, para o primeiro modelo ajustado foram retiradas todas as variáveis que apresentavam separação, já que esse fenômeno acarreta estimadores viesados.

As variáveis restantes e suas interações foram submetidas ao modo de seleção *stepwise* e assim foi obtido o Modelo 1, que considera as variáveis sobre a glicose no sangue (bgr), quantidade de hemoglobina (hemo) e gravidade específica (sg nos níveis 1020 e 1025).

Porém, apenas a variável quantidade de hemoglobina (hemo) foi estatisticamente significativa no modelo porque apenas o valor desta variável foi menor que o nível de significância estabelecido em 0.05.

O critério de informação de Akaike (AIC) obtido é igual a 37,17, abaixo é possível visualizar o gráfico de envelope, Gráfico 24.

Gráfico 24: Quantis-Quantis do resíduo componente do desvio do Modelo 1

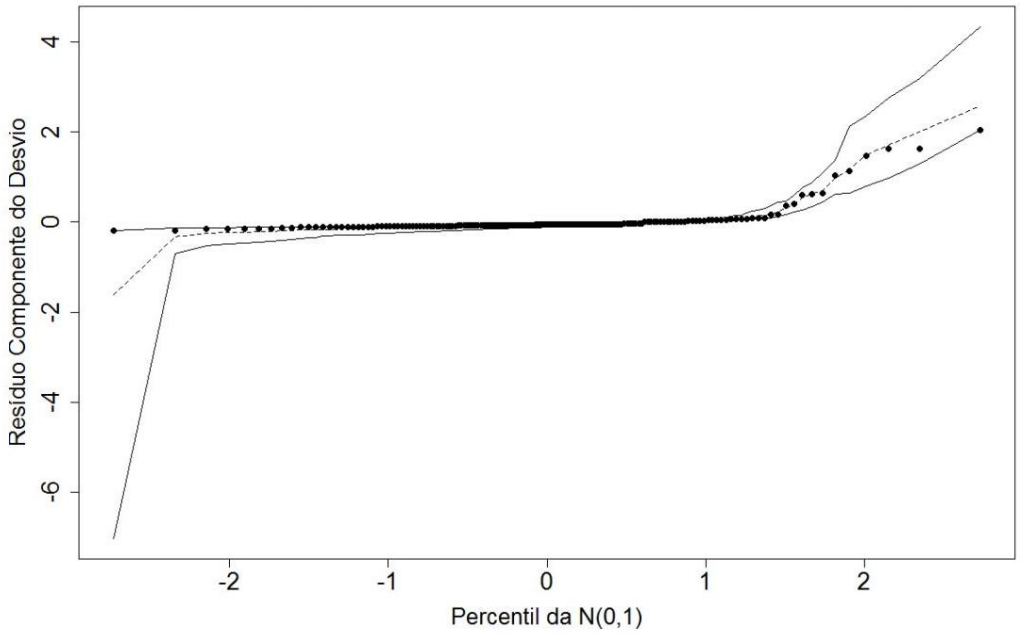


Fonte: Desenvolvido pela autora

No gráfico 24 é possível visualizar que não existe nenhuma evidência que indica a não normalidade dos resíduos componentes do desvio. Por causa da grande perda de informação devido ao fenômeno de separação, que se encontram na maior parte das variáveis, foi usado o método da máxima verossimilhança penalizada, em que a priori de Jeffreys é usada para fazer a verossimilhança. A partir deste ponto, todos os modelos serão ajustados através do método da

verossimilhança penalizada. E assim, fazendo um ajuste com todas as variáveis, no Modelo 2, é possível visualizar que nenhuma das variáveis são significativas estatisticamente a um nível de 5%. O AIC obtido foi igual à 55,48, abaixo é possível visualizar o gráfico de envelope, Gráfico 25.

Gráfico 25: Quantis-Quantis do resíduo componente do desvio do Modelo 2

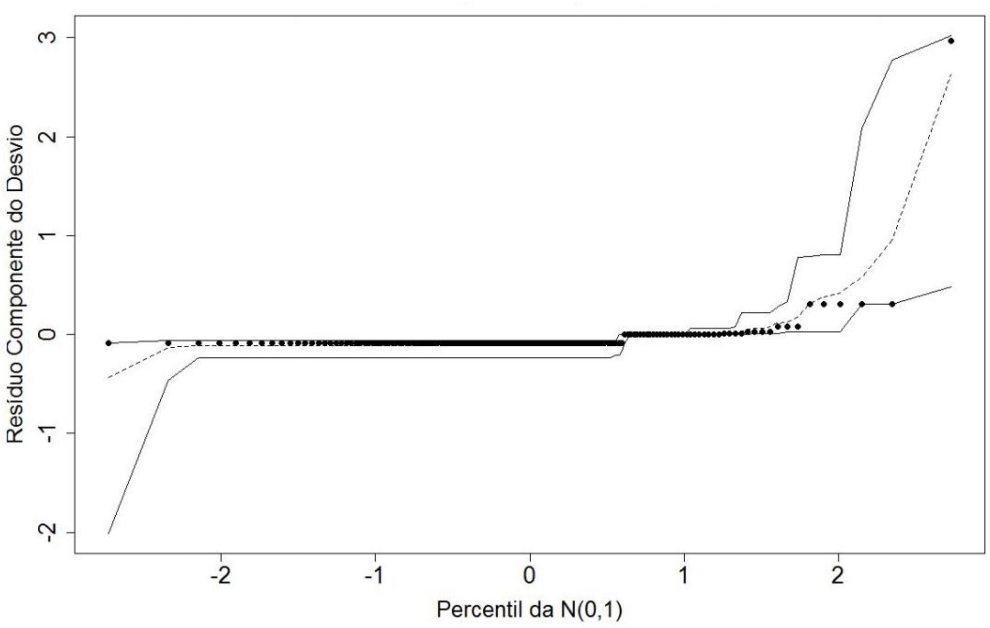


Fonte: Desenvolvido pela autora

O gráfico mostra que alguns resíduos componentes do desvio, estão quase fora do limite das bandas.

Usando a seleção *stepwise*, tem-se o Modelo 3 que considera as variáveis sobre a quantidade de albumina (al), e diabetes (dm). Abaixo é possível visualizar o gráfico de envelope, Gráfico 26.

Gráfico 26: Quantis-Quantis do resíduo componente do desvio do Modelo 3



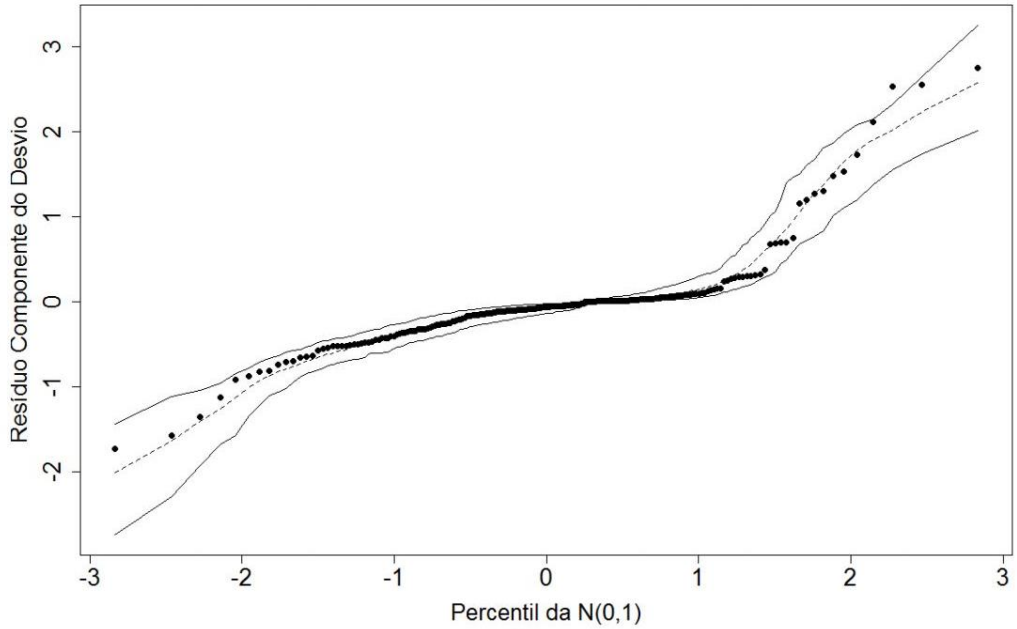
Fonte: Desenvolvido pela autora

Verifica-se o gráfico Quantis- Quantis tem os resíduos componentes do desvio no limite da banda superior. O critério de informação de Akaike é igual a 7.77. Porém, o Modelo 3 contém variáveis que podem apresentar um modelo de interpretação pouco conclusiva, já que leva em consideração a quantidade de albumina e diabetes, que não estão necessariamente associadas com a insuficiência renal.

Por motivos de interpretação, o modelo 4 é baseado em características diferentes que estão associadas à insuficiência renal.

O Modelo 4 ajusta as variáveis idade (age), pressão sanguínea (bp), glicose no sangue (bgr), uréia no sangue (bu), quantidade de sódio (sod). Abaixo é possível visualizar o gráfico de envelope, Gráfico 27.

Gráfico 27: Quantis-Quantis do resíduo componente do desvio do Modelo 4



Fonte: Desenvolvido pela autora

Observa-se o gráfico de Quantis-Quantis dos resíduos componentes do desvio, eles mostram-se bons se comparados à uma distribuição normal, tendo apenas um ponto fora das bandas, pertencente à cauda direita. O critério de informação de Akaike é dado por 69,73. Em resumo, vemos que o modelo que não usa a verossimilhança penalizada apresenta grande perda de informação; todos os gráficos de envelope para todos os modelos não têm fortes indicações de inadequação dos modelos ajustados para os dados. O critério de escolha foi principalmente a interpretabilidade do modelo. Portanto, o Modelo 4 é o ajuste escolhido. Os parâmetros ajustados estão na Tabela 6 abaixo:

Tabela 6: Parâmetros estimados pelo Modelo 4

	Parâmetro estimado	Erro padrão	Nível descritivo
Intercepto	20,267	10,982	0,06
Idade (age)	0,028	0,02	0,16
Pressão sanguínea (bp)	0,084	0,033	0,01
Glicose no sangue (bgr)	0,019	0,008	0,02
Uréia no sangue (bu)	0,046	0,015	0
Sódio (sod)	-0,182	0,074	0,01

Contagem de glóbulos brancos (wbcc)	0,0003	0,0001	0,01
Contagem de glóbulos vermelhos (rbcc)	-2,202	0,474	0

Fonte: Desenvolvido pela autora

Através dos parâmetros ajustados visto na tabela 6 tem-se a exponencial dos parâmetros para concluir sobre as razões de chances controlando por todas as outras variáveis. Na tabela 7 abaixo é possível visualizar esses valores e a interpretação dessas variáveis:

Tabela 7: Interpretação dos parâmetros estimados pelo Modelo 4

Variável	Razão de chances	Interpretação
age	1,028	a cada ano acrescentado na idade, a chance de ter a doença é aproximadamente 3% maior.
bp	1,087	a cada uma unidade de incremento de pressão sanguínea, aumenta em quase 9%a chance de ter a doença.
bgr	1,218	a cada ml/dl a mais de glicose encontrada no sangue, a chance de ter a doença aumenta em 22% aproximadamente.
bu	1,048	a cada mg/dl de ureia aumentada na urina, aumenta-se a chance de ter a doença em 5%.
sod	0,832	cada incremento em mEq/L, a chance de ter a doença diminui em 17%.
wbcc	1.000	aparentemente uma maior ou menor contagem de glóbulos brancos, não altera a chance de ter a doença.
rbcc	0,110	quanto maior a taxa de glóbulos vermelhos, a chance de ter a doença diminui em 89%.

Fonte: Desenvolvido pela autora

Pode-se perceber que dentre as 24 variáveis explicativas, muitas delas estavam associadas às mesmas características e não era necessário todas para explicar o efeito sobre a saúde do doente.

O modelo 4, considera variáveis que revelam que a função de filtragem renal está falha ou então outras doenças que estão associadas, como o diabetes ou a hipertensão. Sendo então, um modelo satisfatório para verificar o efeito de fatores relativos à saúde do paciente com Doença Crônica do Rins.

5 MACHINE LEARNING

Neste capítulo serão apresentadas e aplicadas algumas técnicas de *machine learning*, utilizando a linguagem Python, com o objetivo apresentar modelos que possam prever a ocorrência de doença crônica renal dado os dados de entrada.

No capítulo anterior, foi utilizada a regressão logística como método estatístico para se ajustar aos dados e identificar a importância de cada variável, neste capítulo serão utilizados outros modelos estatísticos, com o intuito de agregar mais valor a este estudo, no sentido de apresentar abordagens adicionais a vista anteriormente.

Serão apresentados os resultados de cada modelo de *machine learning*, assim como o comparativo entre eles, com o objetivo de identificar o modelo que melhor se ajustou aos dados.

5.1 Preparação dos dados

Antes de submeter os dados para serem treinados por modelos estatísticos, foram realizados alguns tratamentos nos dados, identificando tipos de dados e ocorrência de falta de valores (*missing values*).

Os procedimentos apresentados neste subcapítulo gerarão o dado tratado para posteriormente serem ajustados aos modelos apresentados nos próximos subcapítulos, gerando um *dataframe* preparado para ser processado pelos modelos de *machine learning*, desta forma os próximos subcapítulos serão focados no entendimento teórico de cada modelo e suas respectivas aplicações.

Todas as variáveis apresentaram valores nulos, as variáveis contínuas que apresentaram valores nulos, foram atualizadas com a média de seu respectivo atributo, ou seja, receberam o valor médio encontrado dos registros que apresentavam valor não nulo.

Em contrapartida, os registros que apresentaram variáveis categóricas que com valores nulos, foram atualizados com a palavra ‘NULL’, estratégia esta, utilizada para posteriormente ser criado um atributo ao realizar o procedimento de geração de *dummies*.

O código responsável pela leitura dos dados e geração de um *dataframe* inicial, definição de campo *target* e criação de um segundo *dataframe* com os dados tratados conforme estratégia apresentada no parágrafo anterior, está disponível no apêndice A.

Após a execução do código disponível no apêndice A, o *dataframe* ***df2*** contém uma estrutura sem dados nulos e com os campos categóricos separados em atributo do tipo *dummy*,

ou seja, com valores 0 ou 1 para cada combinação de atributos e valores. Na figura 2 pode ser visualizado a estrutura final e uma prévia dos valores calculados:

Figura 2: Visualização de dados após tratamento

```
df2.head()
```

	ane_NULL	ane_no	ane_yes	pe_NULL	pe_no	pe_yes	appet_NULL	appet_good	appet_poor	cad_NULL	...	wbcc	pc
0	0	1	0	0	1	0	0	1	0	0	...	7800.0	44.
1	0	1	0	0	1	0	0	1	0	0	...	6000.0	38.
2	0	0	1	0	1	0	0	0	1	0	...	7500.0	31.
3	0	0	1	0	0	1	0	0	1	0	...	6700.0	32.
4	0	1	0	0	1	0	0	1	0	0	...	7300.0	35.

5 rows × 73 columns

Fonte: Desenvolvido pela autora

Como pode ser visto na figura 2, o novo *dataframe* apresenta uma estrutura com os atributos que originalmente eram de característica categórica em atributos numéricos, com valores *dummy*, ou seja, 0 ou 1. A figura 3 apresenta a estrutura final do *dataframe* que será utilizado pelos modelos de *machine learning*, onde não existem mais valores nulos para nenhum atributo e todas as variáveis categóricas foram transformadas em *dummies*.

Figura 3: Visualização estrutura de dados após tratamento (sem nulos e com dummies)

```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 73 columns):
ane_NULL          400 non-null uint8
ane_no            400 non-null uint8
ane_yes           400 non-null uint8
pe_NULL           400 non-null uint8
pe_no             400 non-null uint8
pe_yes            400 non-null uint8
appet_NULL        400 non-null uint8
appet_good        400 non-null uint8
appet_poor        400 non-null uint8
cad_NULL          400 non-null uint8
cad_no            400 non-null uint8
cad_yes           400 non-null uint8
dm_ yes           400 non-null uint8
dm_NULL           400 non-null uint8
dm_no             400 non-null uint8
dm_yes            400 non-null uint8
htn_NULL          400 non-null uint8
htn_no            400 non-null uint8
htn_yes           400 non-null uint8
ba_NULL           400 non-null uint8
ba_notpresent     400 non-null uint8
ba_present        400 non-null uint8
pcc_NULL          400 non-null uint8
pcc_notpresent    400 non-null uint8
pcc_present       400 non-null uint8
pc_NULL           400 non-null uint8
pc_abnormal       400 non-null uint8
pc_normal         400 non-null uint8
rbc_NULL          400 non-null uint8
rbc_abnormal      400 non-null uint8
rbc_normal        400 non-null uint8
su_0.0            400 non-null uint8
su_1.0            400 non-null uint8
su_2.0            400 non-null uint8
su_3.0            400 non-null uint8
su_4.0            400 non-null uint8
su_5.0            400 non-null uint8
su_NULL           400 non-null uint8
al_0.0            400 non-null uint8
al_1.0            400 non-null uint8
al_2.0            400 non-null uint8
al_3.0            400 non-null uint8
al_4.0            400 non-null uint8
al_5.0            400 non-null uint8
al_NULL           400 non-null uint8
sg_1.005          400 non-null uint8
sg_1.01           400 non-null uint8
sg_1.015          400 non-null uint8
sg_1.02           400 non-null uint8
sg_1.025          400 non-null uint8
sg_NULL           400 non-null uint8
bp_50.0           400 non-null uint8
bp_60.0           400 non-null uint8
bp_70.0           400 non-null uint8
bp_80.0           400 non-null uint8
bp_90.0           400 non-null uint8
bp_100.0          400 non-null uint8
bp_110.0          400 non-null uint8
bp_120.0          400 non-null uint8
bp_140.0          400 non-null uint8
bp_180.0          400 non-null uint8
bp_NULL           400 non-null uint8
rbcc              400 non-null float64
wbcc              400 non-null float64
pcv               400 non-null float64
hemo              400 non-null float64
pot               400 non-null float64
sod               400 non-null float64
sc                400 non-null float64
bu                400 non-null float64
bgr               400 non-null float64
age               400 non-null float64
result            400 non-null object
dtypes: float64(10), object(1), uint8(62)
memory usage: 58.7+ KB
```

Fonte: Desenvolvido pela autora

A estratégia de transformação das variáveis categóricas em variáveis *dummies* (como visto na figura 3) é importante para alguns modelos de *machine learning*, muitos modelos são algébricos, como por exemplo SVM (*support vector machine*), e aceitam apenas valores numéricos como entrada. Embora alguns modelos já façam a conversão nativamente de categórico para numéricos, nem todos apresentam este mecanismo, por este motivo foi utilizado a abordagem de utilização de *dummies* neste trabalho (KUHN; JOHNSON, 2013).

Posteriormente os dados devem ser separados em bases de teste e treino, para assim ser possível medir o *score* dos modelos, diminuindo o risco de hiper-ajustar (*overfit*) o modelo aos dados. A codificação responsável pela divisão das bases pode ser vista no apêndice B.

Após todos os procedimentos acima, todos os modelos apresentados neste estudo serão ajustados sobre os dados contidos nos *dataframes X_train* e *X_test*, que apresentam todas as variáveis explicativas, com respectivamente 75% e 25% dos dados do estudo e os *dataframes Y_train* e *Y_test*, que contém apenas a variável resposta, respectivamente 75% e 25% dos dados do estudo.

5.2 Métricas de score

Com o intuito de mensurar qual modelo está melhor ajustado aos dados, ou seja, consegue fazer predições mais assertivas, é necessário utilizar métricas de *score*. Existem diversas métricas de *score* utilizadas por modelos de *machine learning*.

Neste trabalho serão utilizadas duas métricas de *score*, a primeira delas é a acurácia e a segunda métrica é a área sob a curva roc. Porém, primeiramente é importante ter o entendimento de um método comum para descrever o desempenho de problemas de classificação e que é a base para as duas métricas de score que serão utilizadas neste trabalho, este método é a matriz de confusão (KUHN; JOHNSON, 2013).

A matriz de confusão é uma tabulação cruzada entre as classes observadas e previstas dos dados conforme um modelo aplicado, um exemplo de matriz de confusão pode ser observado na tabela 8.

Tabela 8: Matriz de confusão

n = 400 (número de amostras)	**Predição: ckd**	**Predição: not ckd**
Observado: ckd	210	40
Observado: not ckd	20	130

Fonte: Adaptado de MISHRA, 2018

Através da matriz de confusão, apresentada pela tabela 8, é possível derivar quatro termos que serão utilizados para calcular as métricas de *score*: acurácia e área sob a curva roc. Os quatro termos que serão utilizados, representam cada um dos quadrantes de combinações que ocorrem na matriz de confusão e estão compreendidos na tabela 9 (KUHN; JOHNSON, 2013).

Tabela 9: Derivação da matriz de confusão

Termo	**Sigla**	**Descrição**	**Quantidade de ocorrências encontradas na tabela 8**
True Positives	*TP*	Acerto: Modelo previu "CKD" e dado observado é "CKD"	210
True Negatives	*TN*	Acerto: Modelo previu "NOT CKD" e dado observado é "NOT CKD"	130
False Positives	*FP*	Erro: Modelo previu "CKD" e dado observado é "NOT CKD"	20
False Negatives	*FN*	Erro: Modelo previu "NOT CKD" e dado observado é "CKD"	40

Fonte: Adaptado de MISHRA, 2018

Todas as combinações possíveis entre os acertos e erros de predição em relação a variável resposta observada nos dados, são representadas pelos quatro termos descritas na tabela 9, este entendimento será necessário para o cálculo correto das métricas que serão utilizadas.

A métrica de *score* acurácia (*accuracy*), é calculada através da proporção das predições acertadas em relação a todas as predições realizadas, ou seja, este cálculo considera a apenas a proporção de acertos realizados pelo modelo. Esta estratégia pode gerar um erro de análise por parte do cientista de dados, pois este resultado será confiável apenas nos casos em que a base de dados esteja balanceada em relação as classes da variável resposta. Por exemplo, em um cenário onde a base de estudo contém transações de cartão de crédito e é desejado prever possíveis fraudes, porém a proporção de registros que apresentem a variável resposta "FRAUDE" sejam de 0,1% da base, um modelo de *machine learning* que classifique todos os registros como "NÃO FRAUDE", conseguiria atingir facilmente uma acurácia acima de 99%, o que não representa um modelo realmente eficiente (MISHRA, 2018).

Matematicamente o *score* utiliza os quatro termos apresentamos pela tabela 9, o cálculo realizado utilizando os valores também presentes na tabela 9, podem ser observados logo abaixo.

$$Acurácia = \frac{TP + TN}{TP + TN + FP + FN} = \frac{210 + 130}{210 + 130 + 20 + 40} = 0{,}85$$

Com o intuito de minimizar o risco de realizar uma análise comparativa equivocada entre os modelos de *machine learning*, este estudo irá utilizar também a métrica de *score* denominada área sob a curva roc (*roc_auc*), abrangendo assim a limitação de cálculo apresentado pela métrica de acurácia descrita anteriormente.

Inicialmente é necessário entender como a curva roc é formada. A curva roc é resultante do cruzamento de duas métricas, a taxa de *True Positives* e a taxa de *False Positives*, estas duas métricas também são nomeadas, respectivamente por, sensibilidade e especificidade (MISHRA, 2018). A representação matemáticas de ambas as métricas são:

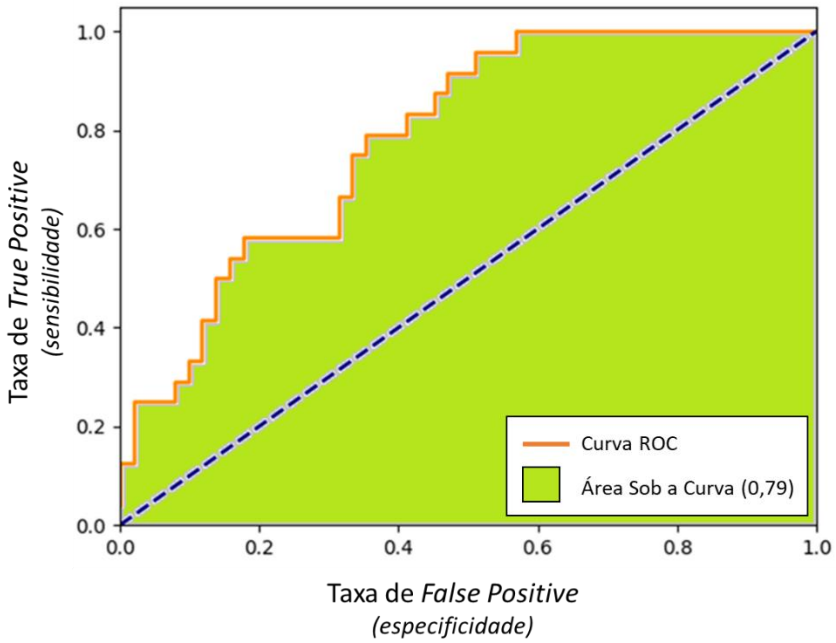
$$Taxa\ de\ True\ Positive\ (\text{sensibilidade}) = \frac{TP}{TP + FN}$$

$$Taxa\ de\ False\ Positive\ (\text{especificidade}) = \frac{FP}{FP + TN}$$

Diferentemente do calculo realizado pela métrica acurácia, que considera apenas a proporção de acertos do modelo em relação a quantidade total de predições realizadas, a curva roc, ao utilizar a métrica sensibilidade, considera a proporção de dados classificados corretamente como positivos em relação a todos os dados que deveriam ter sido classificados como positivos e ao utilizar a métrica especificidade, considera a proporção de dados classificados corretamente como negativos em relação a todos os dados que deveriam ter sido classificados como negativos. Utilizando a combinação das duas métrica para o cálculo da curva roc, é possível calcular um *score* que contempla um cenário completo, penalizando os casos de falhas e não considerando apenas os casos de acertos (MISHRA, 2018).

Ambas as métricas, *taxa de True Positive* e *taxa de False Positive*, por se tratar de proporções, variam entre 0 e 1, desta forma o curva roc é representada dentro deste *range*, conforme figura 4.

Figura 4: Área sob a curva ROC



Fonte: Adaptado de MISHRA, 2018.

A curva roc é representada pela linha laranja na figura 4, para ser concluído o cálculo da métrica área sob a curva roc, é realizado o cálculo da área destacado em verde na figura 4, que representa justamente a área que está abaixo da curva roc. Dado os limites máximos de 1 das métricas utilizadas nos eixos do gráfico, a área sob a curva roc irá variar entre 0, representa que o modelo errou todas as predições e 1, representa acerto de 100% das predições (MISHRA, 2018).

5.2 Decision tree (árvore de decisão)

Árvores de decisão são modelos estatísticos supervisionados para classificação de um conjunto de dados, ou seja, é um modelo estatístico que recebe dados de entrada, também chamadas de variáveis explicativas, e a variável de saída, também conhecida como variável resposta. A variável resposta é a classe que o modelo deverá prever, com base nos dados de entrada, no estudo em questão existes duas classes possíveis, paciente apresenta ou não apresenta doença renal crônica (MITCHELL, 1997). Na área de ciência de dados existem diversos modelos utilizados para responder problemas de classificação, o modelo de árvore de decisão é um dos mais utilizados (YIU, 2019).

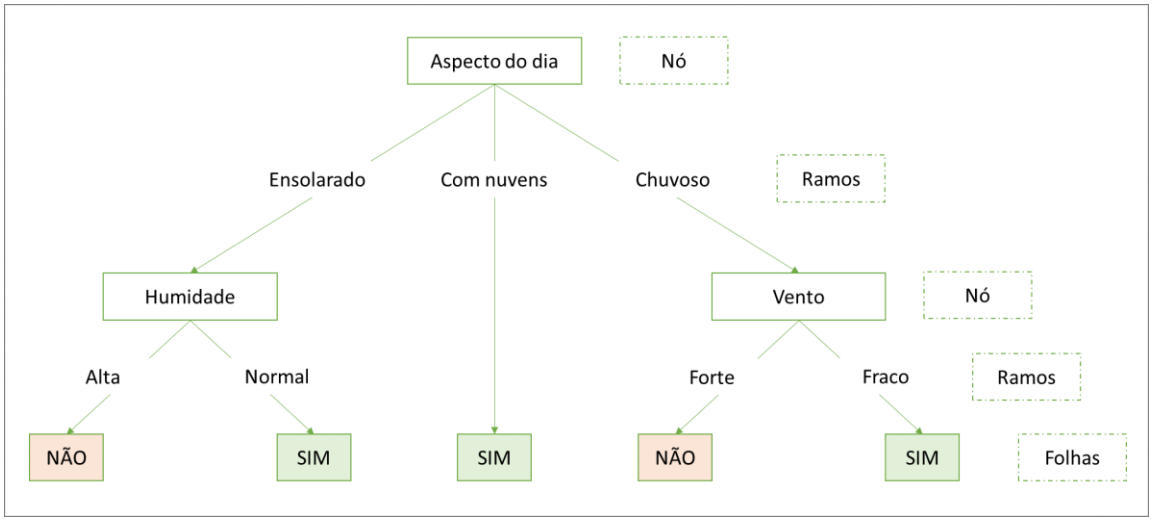
A estratégia utilizada por modelos do tipo árvore de decisão é a divisão de um problema complexo em vários problemas menores, fazendo a decomposição deste problema mais

complexo em subproblemas de menor complexidade. O método de árvore de decisão é muito popular pois pode ser facilmente interpretado, permitindo a identificação dos motivos que geraram uma determinada classificação (Gama, 2002). A representação de uma árvore de decisão é realizada utilizando os termos:

- Nós: representa um teste de valores em um determinado atributo, exemplo: Idade.
- Ramos: corresponde aos valores possíveis após ramificação do nó, exemplo: Idade > 45.
- Folhas: cada folha representa uma classe, exemplo: Apresenta doença Renal.

A figura 5 apresenta um exemplo de representação de uma árvore de decisão, neste exemplo a árvore tem como objetivo prever se um dia específico está propício para se jogar tênis.

Figura 5: Representação gráfica de uma árvore de decisão



Fonte: Adaptado de MITCHELL, 1997.

No exemplo apresentado pela figura 5, a árvore de decisão utiliza apenas 3 atributos para decidir se um dia está propício para a prática de tênis, o primeiro teste ocorre no nó referente ao aspecto do dia, em caso de estar com o valor “Com nuvens” o modelo irá classificar este dia específico como propício para prática de tênis, pois esta resposta direcionou para uma folha de classe “SIM”, caso o aspecto do dia esteja “Ensolarado” a árvore de decisão realiza um segundo teste, referente ao valor do atributo humidade, estando com valor de humidade “Alta” a árvore classificará o dia como não propício para prática de tênis, pois a combinação das duas respostas direcionaram a árvore para uma folha de classe “NÃO”, o mesmo mecanismo se repete para o restante da árvore.

O método de *machine learning* realiza a construção de diversas árvores de decisão variando as quebras, atributos, profundidade da árvore, dentre outros parâmetros, buscando sempre a árvore que proporcione o menor erro na classificação. O conceito mais utilizado para representar a diminuição de erro ao decorrer da construção da árvore, é a entropia, que representa o quão aleatório os dados estão distribuídos, desta forma é possível mensurar a entropia antes e após a divisão da árvore em um próximo nó, selecionando assim o melhor atributo e valor a ser testado (MITCHELL, 1997).

5.2.1 Aplicação de decision tree (árvore de decisão)

Neste capítulo será implementado o algoritmo de árvore de decisão sobre os dados dos pacientes estudados no presente trabalho, apresentando os melhores resultados atingidos através do ajuste de hiper-parâmetros. Como visto no capítulo anterior, o método de árvore de decisão possibilita uma fácil interpretação dos dados, sendo assim será apresentado também a árvore de decisão que melhor se ajustou aos dados.

Foi utilizado o método *GridSearchCV* para realizar os testes dos modelos com diferentes combinações de hiper-parâmetros, com o intuito de identificar a melhor combinação de hiper-parâmetros. A implementação do código pode ser observada no apêndice C, inicialmente foi ajustado o melhor modelo de árvore de decisão considerando a acurácia como métrica de *score*, na sequência foi ajustado o melhor modelo considerando a roc_auc como métrica de *score*. Os resultados obtidos pelo modelo de *machine learning DecisionTreeClassifier*, combinados com ambas as métricas de *score* propostas, podem ser observados na tabela 10.

Tabela 10: Resultados do modelo DecisionTreeClassifier

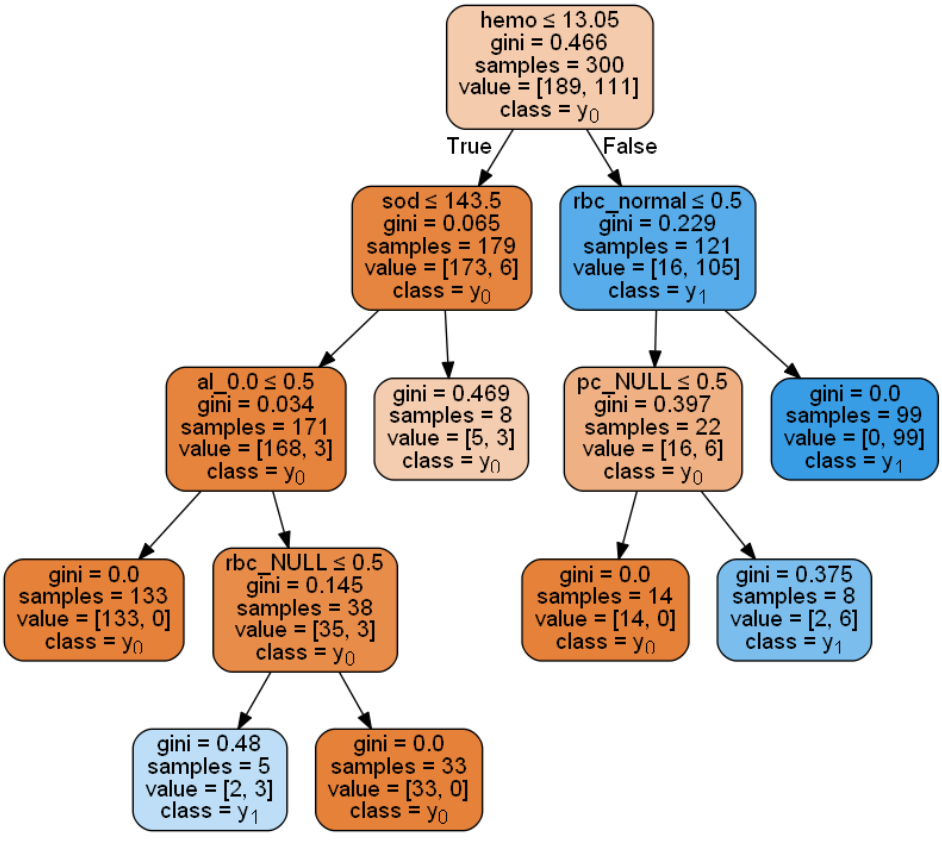
Modelo de *Machine Learning*	**Métrica de *Score***	**Melhores hiper-parâmetros**	***Score* Apurado**
DecisionTreeClassifier	accuracy	max_depth: 4 max_features: 54 min_samples_leaf: 5 min_samples_split: 10	score_treino: 0.977 score_teste: 0.980
DecisionTreeClassifier	roc_auc	max_depth: 16 max_features: 18 min_samples_leaf: 5 min_samples_split: 0.1	score_treino: 0.963 score_teste: 0.936

Fonte: Desenvolvido pela autora

Analisando os resultados obtidos pelo modelo de árvore de decisão, como era esperado, a utilização da métrica acurácia apresentou um valor de *score* mais elevado, porém ao utilizar a métrica *roc_auc*, que conforme visto anteriormente, apresenta um valor ponderado e consequentemente mais seguro de *score*, apresentou um valor positivo, atingindo 0,936 na base de teste.

Com o objetivo de visualizar a arvore de decisão criada pelo modelo, foi implementado o código disponível no apêndice C, logo abaixo da linha de comentário “Visualizando a árvore de decisão”. Este trecho de código gera uma imagem da árvore de decisão criada, esta representação gráfica pode ser observada na figura 6.

Figura 6: Diagrama de Árvore de Decisão



Fonte: Desenvolvido pela autora

O diagrama observado na figura 6 explica qual foi é a lógica de classificação utilizada pela árvore de decisão, podemos salientar os itens abaixo:

- Apenas seis variáveis foram utilizadas pelo modelo.
- É apresentado o índice Gini, que mede a impureza em um determinado nó ou folha.

- Toda ramificação para esquerda representa que a condição no nó anterior foi satisfeita e toda ramificação direita significa que a condição no nó anterior não foi satisfeita.
- A classe y0 representa ter doença crônica renal e y1 representa não ter doença crônica renal.
- Para as variáveis categóricas a variável dummy é testada utilizando o valor 0.5, por exemplo, rbc diferente de “normal” é o mesmo que rbc_normal <= 5, se enquadra no valor zero para a variável dummy rbc_normal.
- Exemplo de Classificação:
  hemo <= 13.05 E sod <= 143.5 E al = 0 E rbc = NULL ➔ Tem doença crônica renal.

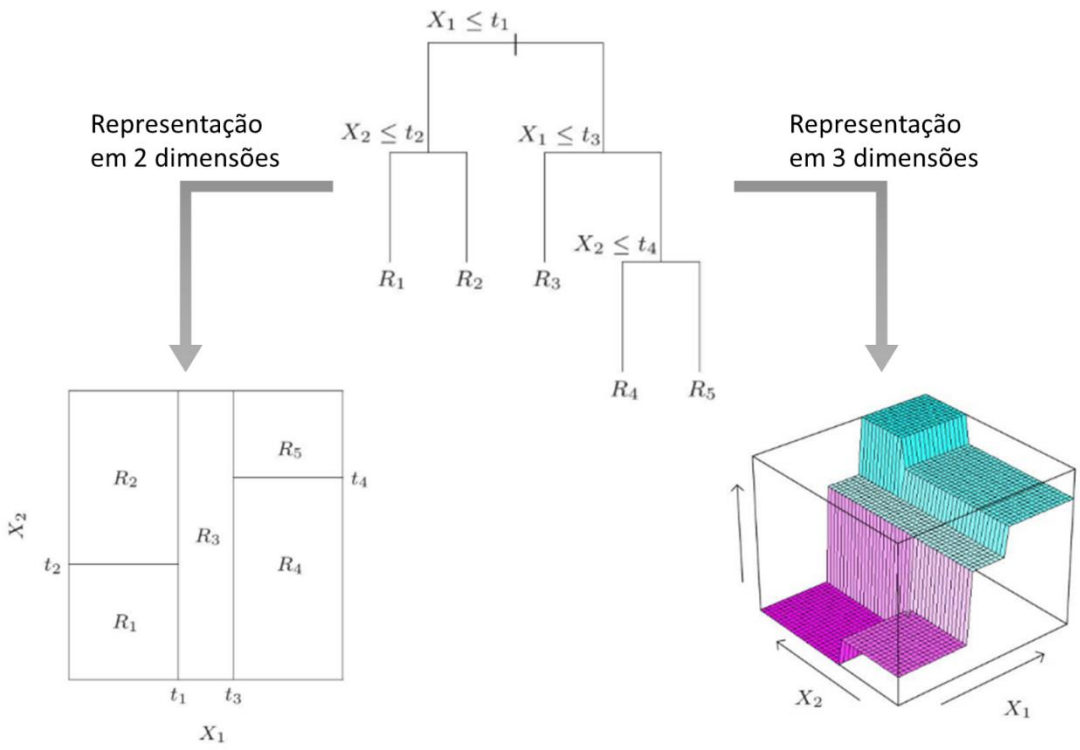
O modelo de *machine learning* de árvore de decisão apresentou resultados satisfatórios, ao analisar o *score* atingido pelo estudo; outro ponto importante deste modelo, é a vantagem sobre os demais modelos por possibilitar a visualização da árvore de decisão criada, o que possibilita ter o entendimento completo da lógica necessárias para a classificação, podendo ser replicada em outras linguagens ou soluções necessárias, exemplo: triagem em *call center*, cenário em que dependendo das respontas obtidas pelo cliente, o sistema pode indicar uma possível fraude, perda de cliente etc.

5.3 Random forest

Como visto anteriormente, o modelo estatístico de árvore de decisão é bastante eficiente, mesmo tendo um funcionamento relativamente simples, porém existem algumas características do modelo de árvore de decisão que foram aprimorados com a abordagem do modelo de *Random Forest*.

Uma característica importante dos modelos de árvore de decisão é a alta, ou seja, existe um comportamento de “degrau” na classificação, ao se variar muito pouco o valor de um dos atributos. Por exemplo, na figura 6, caso o valor de hemo seja 0.01 maior ocorrerá um desvio na sequência de classificação (ZHANG, 2018). Na figura 7 é possível visualizar um suposto cenário em que se tenham duas variáveis explicativas e cinco possíveis valores para que a variável resposta seja classificada.

Figura 7: Representação visual do funcionamento de árvores de decisão



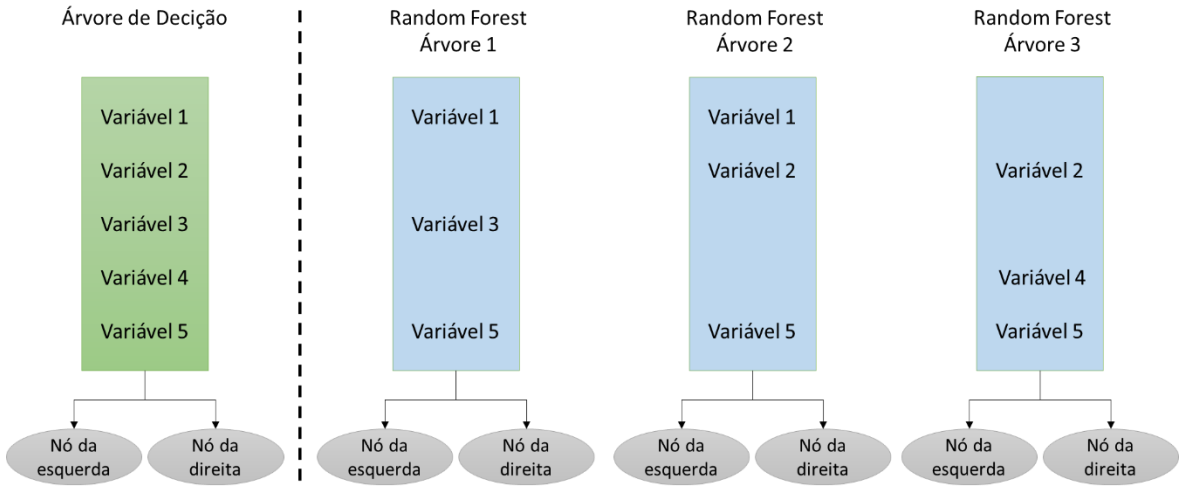
Fonte: Adaptado de: ZHANG, 2018

Conforme ilustrado pela figura 7, a depender da combinação de valores das variáveis X1 e X2, o modelo de árvore de decisão irá realizar a classificação da variável resposta como R1, R2, R3, R4 ou R5. A característica de alta variância, simplificado no parágrafo anterior como comportamento de “degrau”, pode ser visualizado pela representação em três dimensões da figura 6, supondo que o limite t4 tenha o valor de 1000 e o limite t3 tenha o valor de 500, no cenário em que X1 seja qualquer valor acima de 500 e X2 = 1000, o modelo classificaria a variável resposta como R4, em um segundo cenário, em que X1 seja qualquer valor acima de 500 e X2 = 1001, o modelo classificaria a variável resposta como R5, ou seja, com uma pequena alteração em uma das variáveis explicativas, o resultado da predição pode ser bem diferente (KUHN; JOHNSON, 2013).

O modelo *Random Forest*, apresentado neste capítulo, utiliza a combinação de diversas árvores de decisão para resolver problemas de classificação de forma mais assertiva. O princípio deste modelo é a geração de várias árvores de decisão que não sejam correlacionadas, utilizando diferentes atributos para cada árvore e/ou substituição de registros originais da amostra por registros duplicados, para assim gerar árvores diferentes. Este método utilizado para geração

randômica das diferentes árvores de decisão recebe o nome de *Bagging* e está representado pela figura 8 (*Bootstrap Aggregation*) (YIU, 2019).

Figura 8: Criação de árvores não correlacionadas

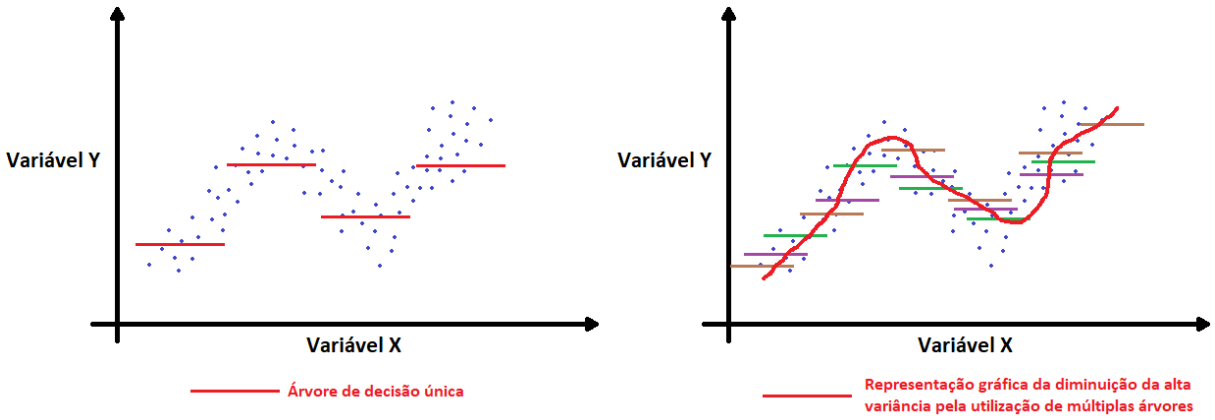


Fonte: Adaptado de YIU, 2018

Desta forma existiram diversas árvores para classificar um registro, a classificação se dá pela maioria das classificações, por exemplo, caso sejam geradas 13 árvores de decisão em um modelo de *Random Forest*, ao ser classificado um determinado registro por cada uma das 13 árvores de decisão foram obtidos os resultados: 9 árvores classificaram o registro como “apresenta doença crônica renal” e 4 árvores classificaram o registro como “não apresenta doença crônica renal”, ou seja, a classificação do modelo *Random Forest* será “apresenta doença crônica renal”.

Este mecanismo é muito poderoso, pois caso algumas árvores não estejam classificando de forma correta em determinadas situações, as demais árvores corrigirão a classificação final. Na Figura 9 é apresentada uma representação gráfica do efeito de diminuição da alta variância que ocorre nos modelos de árvore de decisão.

Figura 9: Diminuição da alta variância utilizando *Random Forest*



Fonte: Adaptado de KUHN; JOHNSON, 2013

Após a utilização de várias árvores de decisão pelo modelo de *Random Forest* (cada árvore representada por uma cor), é possível utilizar o equivalente à média das árvores para determinado ponto, trazendo assim um resultado de classificação não linear e com baixa variância, conforme pode ser observado no figura 9.

5.3.1 Aplicação de random forest

De forma similar ao realizado para o modelo de árvore de decisão, neste capítulo será implementado o algoritmo de *Random Forest* sobre os dados, assim como a utilização do método *GridSearchCV* para realizar os testes dos modelos com diferentes combinações de hiper-parâmetros, com o intuito de identificar a combinação de hiper-parâmetros que melhor explica dos dados.

Como visto no capítulo anterior, o modelo estatístico de *Random Forest* utiliza a combinação de várias árvores de decisão para aprimorar o resultado da classificação, porém diferentemente do modelo de árvore de decisão, não é possível visualizar a árvore com as regras utilizadas para classificação, pois para o modelo de *Random Forest* não existe apenas uma árvore, com apenas um caminho possível para cada folha (ou classificação).

A implementação do código pode ser observada no apêndice D, inicialmente foi ajustado o melhor modelo de *Random Forest* considerando a acurácia como métrica de *score*, na sequência foi ajustado o melhor modelo considerando a roc_auc como métrica de *score*. Os resultados obtidos pelo modelo de *machine learning RandomForestClassifier*, combinados com ambas as métricas de *score* propostas, podem ser observados na tabela 11.

Tabela 11: Resultados do modelo *RandomForestClassifier*

Modelo de *Machine Learning*	**Métrica de *Score***	**Melhores hiper-parâmetros**	***Score* Apurado**
RandomForestClassifier	accuracy	bootstrap: True max_depth: 8 max_features: 18 min_samples_leaf: 5 min_samples_split: 0.1 n_estimators: 100	score_treino: 0.983 score_teste: 0.970
RandomForestClassifier	roc_auc	bootstrap: False max_depth: 8 max_features: 18 min_samples_leaf: 5 min_samples_split: 0.1 n_estimators: 100	score_treino: 0.977 score_teste: 0.962

Fonte: Desenvolvido pela autora

O modelo de *Random Forest* apresentou valores de *scores* superiores aos valores de *scores* atingidos pelo modelo de árvore de decisão, apresentados no capítulo anterior. Conforme abordado neste capítulo, o modelo de *machine learning Random Forest* faz uso de diferentes árvores de decisão combinadas, com o intuito de melhorar os resultados de classificação, quando comparados a uma árvore de decisão isolada, esta melhoria no resultado foi atingida, conforme dados da tabela 11.

5.4 Gradient boosting (GBM) e XGBoost

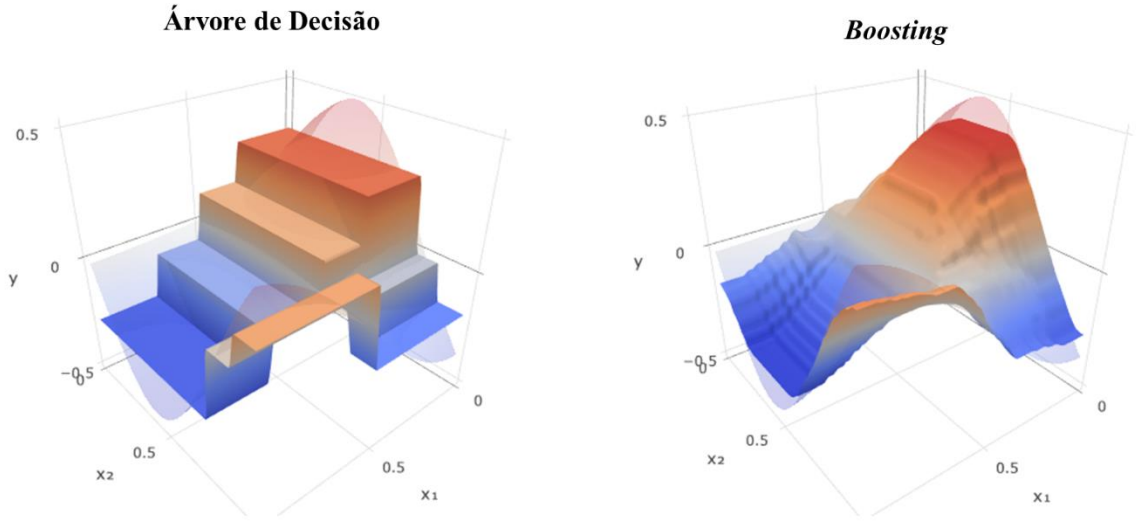
Neste capítulo serão apresentados os modelos de *machine learning* GBM e *XGBoost*, ambos os algoritmos utilizam o conceito de *boosting*, que tem como princípio utilizar diversas regras fracas para gerar uma aprendizagem forte. O mecanismo por trás do funcionamento do *boosting* utiliza diversas iterações de modelos de predições mais simples de *machine learning*, gerando diversas regras fracas. Ao realizar a combinação destas regras fracas é gerada uma regra única forte, que tende a realizar predições mais assertivas (2016, BYLIPUDI).

Algoritmos de *boosting* são genéricos, podem ser aplicados a diferentes modelos, porém é muito utilizado a árvores de decisão. Outra característica do mecanismo de *boosting* é a utilização de pesos. Primeiramente é definindo peso igual para toda a amostra de dados, após a primeira etapa de classificação, todas as observações que foram classificadas de forma

equivocada recebem um aumento de peso, tendo uma relevância maior na próxima etapa de classificação (KUHN; JOHNSON, 2013).

Após fazer a combinação das predições fracas gerando assim uma predição forte, normalmente é observado um ganho nas predições, por este motivo este mecanismo recebe o nome de *boosting*, impulsionar traduzindo para o português. De forma similar ao modelo de *machine* de *Random Forest*, os algoritmos de *boosting* também reduzem o comportamento de alta variância, a figura 10 ilustra o comparativo entre um modelo de árvore de decisão e um modelo de *boosting*.

Figura 10: Comparativo visual de modelos de árvore de decisão e boosting



Fonte: Adaptado de ROGOZHNIKOV, 2016.

A representação gráfica do modelo de *boosting* apresentada na figura 10, representa a combinação de 100 árvores de decisão e como pode ser observado, o efeito de alta variância é minimizado, gerando um modelo que explica melhor aos dados, quando comparado a uma árvore de decisão isoladamente (ROGOZHNIKOV, 2016).

Neste estudo serão apresentados dois algoritmos muito utilizados de *boosting*, são eles, *Gradient Boosting* (GBM) e *XGBoost*. Ambos os algoritmos podem apresentar bons resultados, idealmente podem ser ajustados ambos os modelos para aferir qual deles obteve melhores resultados sobre os dados analisados, porém o modelo *XGBoost* normalmente apresenta um melhor resultado devido a algumas vantagens em seu algoritmo (BYLIPUDI, 2016).

O algoritmo do *XGBoost* realiza a regularização, mecanismo importante para diminuir a chance de sobre ajuste (*overfiting*), tal mecanismo não existe na implementação padrão do *Gradient Boosting*; o algoritmo do *XGBoost* apresenta processamento paralelo, incluindo a

implementação em ecossistema *Hadoop*, sendo muito mais rápido do que o *Gradient Boosting*; o algoritmo do *XGBoost* consegue tratar valores faltantes automaticamente (*missing*), não sendo necessário um tratamento prévio dos dados, de forma automática o algoritmo adota abordagens diferentes para os valores faltantes e identifica a melhor estratégia algoritmo. Os resultados apresentados pelos modelos de *boosting* em bases com muitas variáveis e algumas delas com baixa qualidade, tendem a ser melhores do que o modelo de *Random Forest* (BYLIPUDI, 2016).

5.4.1 Aplicação de gradient boosting (GBM) e XGBoost

De forma similar ao realizado para os modelos abordados anteriormente, neste capítulo serão implementados os modelos de *gradiente boosting* e *xgboost*, utilizando o método *GridSearchCV* para realizar os testes dos modelos com diferentes combinações de hiper-parâmetros, com o intuito de identificar a combinação de hiper-parâmetros apresente melhores resultados aos modelos.

A implementação do código aplicando os modelos de *gradiente boosting* e *xgboost*, podem ser encontrados no apêndice E, de forma similar aos modelos anteriores, serão calculados os *scores* para as métricas de acurácia e roc_auc. Os resultados da combinação entre os dois modelos abordados e as duas métricas de *score* utilizadas, podem ser observados na tabela 12.

Tabela 12: Resultados dos modelos *GradientBoostingClassifier e XGBoost*

Modelo de *Machine Learning*	**Métrica de *Score***	**Melhores hiper-parâmetros**	***Score* Apurado**
GradientBoostingClassifier	accuracy	learning_rate: 1 max_depth: 2 max_features: 18 min_samples_leaf: 0.1 min_samples_split: 10 n_estimators: 16	score_treino: 1.000 score_teste: 0.980
GradientBoostingClassifier	roc_auc	learning_rate: 1 max_depth: 2 max_features: 18 min_samples_leaf: 0.1 min_samples_split: 10 n_estimators: 100	score_treino: 1.000 score_teste: 1.000
XGBClassifier	accuracy	colsample_bytree: 0.7 learning_rate: 0.1 max_depth: 4 min_child_weight: 12 n_estimators: 20 nthread: 4 objective: binary:logistic silent: 1 subsample: 0.5	score_treino: 0.953 score_teste: 0.960
XGBClassifier	roc_auc	colsample_bytree: 0.8 learning_rate: 0.05 max_depth: 4 min_child_weight: 12 n_estimators: 10 nthread: 4 objective: binary:logistic silent: 1 subsample: 0.5	score_treino: 0.936 score_teste: 0.921

Fonte: Desenvolvido pela autora

Analisando os resultados disponíveis na tabela 12, o modelo de *Gradient Boosting* apresentou resultados perfeitos, mesmo ao se utilizar a métrica de *score* roc_auc, o *score* foi de 100% de acerto em ambas as bases, este resultado indica um possível sobre ajuste (*overfitting*), conforme visto anteriormente, este modelo não provê o mecanismo de regularização, que penaliza o modelo a medida em que ele se especializa, desta forma existe um risco aumentado de ocorrer sobre ajuste.

Por outro lado, o modelo de *machine learning XGBoost*, apresentou *scores* superiores a 0.92, sendo assim, não tão próximos a 100% como os resultados atingidos pelo *Gradient Boosting*. Desta forma é possível concluir ser mais seguro a utilização do modelos *XGBoost*,

pois ao utilizar o mecanismo de regularização, minimiza a chance de ocorrer o sobre ajuste (*overfitting*) e ainda apresenta um *score* relativamente alto.

6 CONCLUSÃO

Este trabalho ajudou a entender a viabilidade da aplicação de estudos estatísticos e modelos de *machine learning*, em predizer a ocorrência de doenças renais crônicas em indivíduos dados algumas informações observadas em exames de sangue e exames de urina.

O estudo estatístico realizado sobre os dados ajudou a entender quais as variáveis mais significativas ao se utilizar a regressão logística, assim como as características das variáveis e as relações entre elas.

Dentre todos os modelos e métricas de *score* utilizadas neste trabalho, o que apresentou melhores resultados foi o modelo de *machine learning Random Forest*, atingindo o *score* roc_auc de 0.96. O modelo *Gradient Boosting* apresentou valores de score superiores, porém existe uma grande chance de ter ocorrido sobre ajuste (*overfitting*) para este caso.

Para todos os modelos ajustados, foram atingidos *scores* relativamente altos, acima de 0.90, o que indica que a base de dados tem poucos problemas de qualidade de dados e as variáveis explicativas, de fato explicam a relação dos dados e a classificação referente a se ter ou não a doença crônica renal.

O modelo de árvore de decisão, embora não tenha apresentado os melhores resultados preditivos, quando comparados aos demais modelos, ainda assim apresentou um alto poder explicativo, com roc_auc de 0.936. A utilização deste modelo traz uma vantagem no entendimento da classificação, pois apresenta o diagrama de decisões para tal classificação. A utilização deste diagrama poderia ser apresentada como ferramenta de auxílio de diagnóstico prévio, sendo facilmente implementado em sistemas que não sejam necessariamente de *machine learning*.

Para trabalhos futuros, caberia o estudo de viabilidade da utilização do diagrama resultante da árvore de decisão em consultório médico, de forma a apoiar o corpo clínico no prévio diagnóstico da doença crônica renal. Cabe também a análise de novas variáveis e uma amostragem maior de dados, possibilitando assim o aumento da segurança do estudo, diminuindo o risco de ocorrência de sobre ajuste e considerando novos atributos, provenientes de novos exames.

REFERÊNCIAS

As variáveis quantitativas e qualitativas e os testes estatísticos. Pós-Graduando, 2012. Disponível em: <https://posgraduando.com/as-variaveis-quantitativas-e-qualitativas-e-os-testes-estatisticos/>. Acessado em: 29 de out. de 2019.

BUSSAB, Wilton de O; MORETTIN, Pedro A. **Estatística Básica**. São Paulo: Saraiva, 2010. 557p.

CARMO, V. Correlação Pearson Spearman Kendall. 2010. Disponível em: <http://www.inf.ufsc.br/~vera.carmo/Correlacao/Correlacao_Pearson_Spearman_Kendall.pdf >. Acessado em: 02 de nov. de 2019.

EHLERS, Ricardo S. **Inferência Estatística (PDF)**. USP, 2002. Disponível em: <http://conteudo.icmc.usp.br/pessoas/ehlers/bayes/bayes.pdf>. Acessado em: 04 de nov. de 2019.

FIRTH, D. **Bias reduction of maximum likelihood estimates**. 1993. Biometrika Volume 80, Número 1: 27 – 38.1993

GALVÃO, Taís Freire; MARINHO, Ana Wanda Guerra Barreto; PENHA, Anderson da Paz; SILVA, Marcus Tolentino. Prevalência de doença renal crônica em adultos no Brasil: revisão sistemática da literatura. Scientific Electronic Library Online, 2017. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S1414-462X2017000300379>. Acessado em: 28 de out. de 2019.

GONÇALVES, Joria Martinho. **SOLUÇÕES PARA O PROBLEMA DE SEPARAÇÃO QUASE-COMPLETA EM REGRESSÃO LOGÍSTICA**. 2008. 55f. Trabalho de Conclusão de Curso (Especialização) - Universidade Federal de Minas Gerais, Belo Horizonte, 2008

HEINZE, George; SCHEMPER, Michael (2002) **A solution to the problem of separation in logistic regression. Statistics in Medicine.** Capítulo 21: 2409 – 2419.2002

KUHN, Max.; JOHNSON, Kjell. **Applied Predictive Modeling**. New York: Springer, 2013. 615 p.

LITTELL, R. C.; MILLIKEN, G. A.; STROUP, W. W.; WOLFINGER, R. D; SCHABENBERGER, O. **SAS System for Mixed Models. Cary: Statistical Analysis System Institute**, 2002. 633p.

MCFADDEN, Daniel. *Handbook of Econometrics, Volume IV*. 1994. Disponível em: <https://www.ssc.wisc.edu/~xshi/econ715/chap36neweymacfadden.pdf> Acessado em: 02 de nov. de 2019.

Medidas de dispersão. Portal Action, 2018. Disponível em: <http://www.portalaction.com.br/estatistica-basica/22-medidas-de-dispersao >. Acessado em: 02 de nov. de 2019.

Medidas de posição. Portal Action, 2018. Disponível em: <http://www.portalaction.com.br/estatistica-basica/21-medidas-de-posicao>. Acessado em: 02 de nov. de 2019.

MENARD, Scott W. **Applied logistic regression analysis**. Thousands Oaks, Calif: Sage

MISHRA, Aditya. Metrics to Evaluate your Machine Learning Algorithm. Towards Data Science, 2018. Disponível em: <https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234>. Acesso em: 07 de nov. de 2019.

MITCHELL, Tom M. **Machine Learning**. New York: McGraw-Hill Science, 1997. 432 p.

Publications, n. 7, 1995

ROGOZHNIKOV, A. Gradient Boosting explained. Git Hub, 2016. Disponível em: <https://arogozhnikov.github.io/2016/06/24/gradient_boosting_explained.html>. Acesso em: 07 de nov. de 2019.

ROMÃO JUNIOR, João Egidio. Doença Renal Crônica: Definição, Epidemiologia e Classificação. Brazilian journal of nephrology, 2018. Disponível em: <http://www.bjn.org.br/details/1183/pt-BR/doenca-renal-cronica--definicao--epidemiologia-e-classificacao>. Acessado em: 28 de out. de 2019.

RUBINI, L. Jerlin. Chronic Kidney Disease Data Set. UCI, 2015. Disponível em: <https://archive.ics.uci.edu/ml/datasets/chronic_kidney_disease>. Acessado em: 28 de out. de 2019.

Seleção stepwise. Portal Action, 2018. Disponível em: <http://www.portalaction.com.br/analise-de-regressao/4251-selecao-stepwise>. Acessado em: 02 de nov. de 2019.

Tabela de Frequência. Portal Educação, 2017. Disponível em: <https://www.portaleducacao.com.br/conteudo/artigos/administracao/tabela-de-frequencia/30575>. Acessado em: 02 de nov. de 2019.

WILK, M.B.; GNANADESIKAN, R. (1968). **Probability plotting methods for the analysis of data**. Biometrika Trust. 55(1): 1–17.

WOLFINGER, R. D. **Covariance estruture selection in general mixed models. Comunications in Statistic**s. V.22. p1079-1106. 1993.

ZHANG, W. Decision Trees. Git Hub, 2018. Disponível em: <https://wei2624.github.io/MachineLearning/sv_trees/>. Acesso em: 07 de nov. de 2019.

APÊNDICE A - Código para leitura e tratamento de dados

Código disponível na íntegra em: https://github.com/eccoutos/TCC_FIA.git

```
# Atribuindo parametros de entrada
target = 'result'

# Importando os dados de entrada
df = pd.read_csv('C:/Users/elen/Desktop/FIA/TCC/Rim/chronic_kidney_disease_full.csv', sep =
',', decimal = '.')


# Adicionando coluna de Target
df2 = pd.concat([df['result']])

# Criando Dataframe auxiliar
df_temp = pd.DataFrame()


# Adicionando colunas continuas com nulos
df_temp = df.age.fillna( df.age.mean() )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.bgr.fillna( df.bgr.mean() )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.bu.fillna( df.bu.mean() )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.sc.fillna( df.sc.mean() )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.sod.fillna( df.sod.mean() )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.pot.fillna( df.pot.mean() )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.hemo.fillna( df.hemo.mean() )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.pcv.fillna( df.pcv.mean() )
df2 = pd.concat([df temp, df2], axis=1)
df_temp = df.wbcc.fillna( df.wbcc.mean() )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.rbcc.fillna( df.rbcc.mean() )
df2 = pd.concat([df_temp, df2], axis=1)


# Adicionando e preparando colunas com dummies com tratamento de NULL
df_temp = df.bp.fillna( 'NULL' )
df_temp = pd.get_dummies( df_temp, prefix='bp' )
df2 = pd.concat([df_temp, df2], axis=1)
df temp = df.sg.fillna( 'NULL' )
df_temp = pd.get_dummies( df_temp, prefix='sg' )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.al.fillna( 'NULL' )
df_temp = pd.get_dummies( df_temp, prefix='al' )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.su.fillna( 'NULL' )
df_temp = pd.get_dummies( df_temp, prefix='su' )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.rbc.fillna( 'NULL' )
df_temp = pd.get_dummies( df_temp, prefix='rbc' )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.pc.fillna( 'NULL' )
df_temp = pd.get_dummies( df_temp, prefix='pc' )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.pcc.fillna( 'NULL' )
df_temp = pd.get_dummies( df_temp, prefix='pcc' )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.ba.fillna( 'NULL' )
df_temp = pd.get_dummies( df_temp, prefix='ba' )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.htn.fillna( 'NULL' )
df_temp = pd.get_dummies( df_temp, prefix='htn' )
df2 = pd.concat([df_temp, df2], axis=1)
```

```
df_temp = df.dm.fillna( 'NULL' )
df_temp = pd.get_dummies( df_temp, prefix='dm' )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.cad.fillna( 'NULL' )
df_temp = pd.get_dummies( df_temp, prefix='cad' )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.appet.fillna( 'NULL' )
df_temp = pd.get_dummies( df_temp, prefix='appet' )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.pe.fillna( 'NULL' )
df_temp = pd.get_dummies( df_temp, prefix='pe' )
df2 = pd.concat([df_temp, df2], axis=1)
df_temp = df.ane.fillna( 'NULL' )
df_temp = pd.get_dummies( df_temp, prefix='ane' )
df2 = pd.concat([df_temp, df2], axis=1)
```

APÊNDICE B - Divisão das bases de treino e teste

Código disponível na íntegra em: https://github.com/eccoutos/TCC_FIA.git

```
# Dividindo a base entre variaveis explicativas e variavel resposta
X = df2.loc[:, df2.columns !='result']
Y = df2.result

# Separando a Base entre Teste e Treino
X_train, X_test, Y_train, Y_test = model_selection.train_test_split(X, Y, train_size=0.75,
test_size=0.25, random_state=7)
```

APÊNDICE C - Aplicação modelo decision tree classifier

Código disponível na íntegra em: https://github.com/eccoutos/TCC_FIA.git

```
# ## Modelo: DecisionTreeClassifier com Score: Acurácia
modelo = DecisionTreeClassifier()

# Tunning:
parametros = {'max_depth': [2, 4, 8, 12, 16], 'min_samples_split': [0.1, 1.0, 10],
'min_samples_leaf': [0.1, 0.5, 5], 'max_features': [18, 36, 54, 72]}
mod = GridSearchCV(modelo, parametros, n_jobs=4,  cv=4, scoring='accuracy', verbose=10,
refit=True)
mod.fit(X_train, Y_train)

# Aplicando o Modelo
model = DecisionTreeClassifier(**mod.best_params_)
model.fit(X_train, Y_train)
predict_train = model.predict(X_train)
predict_test = model.predict(X_test)

# Calculando o Score
print('métrica: acurácia')
print('score_treino: ' + str(round(accuracy_score(Y_train, predict_train),6) ))
print('score_teste: ' + str(round(accuracy_score(Y_test, predict_test),6) ))
print('melhores hiperparametros: ' + str(mod.best_params_))



# ## Modelo: DecisionTreeClassifier com Score: roc_auc
modelo = DecisionTreeClassifier()

# Tunning:
parametros = {'max_depth': [2, 4, 8, 12, 16], 'min_samples_split': [0.1, 1.0, 10],
'min_samples_leaf': [0.1, 0.5, 5], 'max_features': [18, 36, 54, 72]}
mod = GridSearchCV(modelo, parametros, n_jobs=4,  cv=4, scoring='roc_auc', verbose=10,
refit=True)
mod.fit(X_train, Y_train)

# Aplicando o Modelo
model = DecisionTreeClassifier(**mod.best_params_)
model.fit(X_train, Y_train)
predict_train = model.predict(X_train)
predict_test = model.predict(X_test)

# Calculando o Score
print('métrica: acurácia')
print('score_treino: ' + str(round(roc_auc_score(Y_train, predict_train),6) ))
print('score_teste: ' + str(round(roc_auc_score(Y_test, predict_test),6) ))
print('melhores hiperparametros: ' + str(mod.best_params_))


# Visualizando a árvore de decisão:
from sklearn.externals.six import StringIO
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus

dot data = StringIO()
export_graphviz(model, out_file=dot_data,
                filled=True, rounded=True,
                special_characters=True,
                class_names=True,
                feature_names = X_train.columns.tolist())

graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```

APÊNDICE D - Aplicação modelo random forest classifier

Código disponível na íntegra em: https://github.com/eccoutos/TCC_FIA.git

```
# ## Modelo: RandomForestClassifier com Score: Acurácia
modelo = RandomForestClassifier()

# Tunning:
parametros = {'n_estimators': [10, 50, 100], 'max_depth': [2, 8, 16], 'min_samples_split':
[0.1, 1.0, 10], 'min_samples_leaf': [0.1, 0.5, 5], 'bootstrap': [True, False], 'max_features':
[18, 36, 54, 72]}
mod = GridSearchCV(modelo, parametros, n_jobs=4,  cv=4, scoring='accuracy', verbose=10,
refit=True)
mod.fit(X_train, Y_train)

# Melhores Parametros: {'bootstrap': True, 'max_depth': 8, 'max_features': 18,
'min_samples_leaf': 5, 'min_samples_split': 10, 'n_estimators': 100}

# Aplicando o Modelo
model = RandomForestClassifier(**mod.best_params_)
model.fit(X_train, Y_train)
predict_train = model.predict(X_train)
predict_test = model.predict(X_test)

# Calculando o Score
print('métrica: acurácia')
print('score_treino: ' + str(round(accuracy_score(Y_train, predict_train),6) ))
print('score_teste: ' + str(round(accuracy_score(Y_test, predict_test),6) ))
print('melhores hiperparametros: ' + str(mod.best_params_))



# ## Modelo: RandomForestClassifier com Score: roc_auc
modelo = RandomForestClassifier()

# Tunning:
parametros = {'n_estimators': [10, 50, 100], 'max_depth': [2, 8, 16], 'min_samples_split':
[0.1, 1.0, 10], 'min_samples_leaf': [0.1, 0.5, 5], 'bootstrap': [True, False], 'max_features':
[18, 36, 54, 72]}
mod = GridSearchCV(modelo, parametros, n_jobs=4,  cv=4, scoring='roc_auc', verbose=10,
refit=True)
mod.fit(X_train, Y_train)

# Melhores Parametros: {'bootstrap': False, 'max_depth': 8, 'max_features': 18,
'min_samples_leaf': 5, 'min_samples_split': 10, 'n_estimators': 10}

# Aplicando o Modelo
model = RandomForestClassifier(**mod.best_params_)
model.fit(X_train, Y_train)
predict_train = model.predict(X_train)
predict_test = model.predict(X_test)

# Calculando o Score
print('métrica: roc_auc')
print('score_treino: ' + str(round(roc_auc_score(Y_train, predict_train),6) ))
print('score_teste: ' + str(round(roc_auc_score(Y_test, predict_test),6) ))
print('melhores hiperparametros: ' + str(mod.best_params_))
```

APÊNDICE E - Aplicação modelo gradient boosting e xgboost

Código disponível na íntegra em: https://github.com/eccoutos/TCC_FIA.git

```
# ## Modelo: GradientBoostingClassifier com Score: accuracy
modelo = GradientBoostingClassifier()

# Tunning:
parametros = {'learning_rate': [1, 0.25, 0.05, 0.01], 'n_estimators': [4, 16, 100],
'max_depth': [2, 8, 16], 'min_samples_split': [0.1, 1.0, 10], 'min_samples_leaf': [0.1, 0.5,
5], 'max_features': [18, 36, 54, 72]}
mod = GridSearchCV(modelo, parametros, n_jobs=4,  cv=4, scoring='accuracy', verbose=10,
refit=True)
mod.fit(X_train, Y_train)

# Melhores Parametros: {'learning_rate': 1, 'max_depth': 2, 'max_features': 18,
'min_samples_leaf': 0.1, 'min_samples_split': 0.1, 'n_estimators': 100}

# Aplicando o Modelo
model = GradientBoostingClassifier(**mod.best_params_)
model.fit(X_train, Y_train)
predict_train = model.predict(X_train)
predict_test = model.predict(X_test)

# Calculando o Score
print('métrica: acurácia')
print('score_treino: ' + str(round(accuracy_score(Y_train, predict_train),6) ))
print('score_teste: ' + str(round(accuracy_score(Y_test, predict_test),6) ))
print('melhores hiperparametros: ' + str(mod.best_params_))


# ## Modelo: GradientBoostingClassifier com Score: roc_auc
modelo = GradientBoostingClassifier()

# Tunning:
parametros = {'learning_rate': [1, 0.25, 0.05, 0.01], 'n_estimators': [4, 16, 100],
'max_depth': [2, 8, 16], 'min_samples_split': [0.1, 1.0, 10], 'min_samples_leaf': [0.1, 0.5,
5], 'max_features': [18, 36, 54, 72]}
mod = GridSearchCV(modelo, parametros, n_jobs=4,  cv=4, scoring='roc_auc', verbose=10,
refit=True)
mod.fit(X_train, Y_train)

# Melhores Parametros: {'learning_rate': 1, 'max_depth': 2, 'max_features': 18,
'min_samples_leaf': 5, 'min_samples_split': 10, 'n_estimators': 16}

# Aplicando o Modelo
model = GradientBoostingClassifier(**mod.best_params_)
model.fit(X_train, Y_train)
predict_train = model.predict(X_train)
predict_test = model.predict(X_test)

# Calculando o Score
print('métrica: roc_auc')
print('score_treino: ' + str(round(roc_auc_score(Y_train, predict_train),6) ))
print('score_teste: ' + str(round(roc_auc_score(Y_test, predict_test),6) ))
print('melhores hiperparametros: ' + str(mod.best_params_))


# ## Modelo: XGBClassifier com Score: accuracy
modelo = XGBClassifier()

# Tunning:
parametros = {'nthread': [4], 'objective': ['binary:logistic'], 'learning_rate': [0.05, 0.1,
0.15], 'max depth': [4, 8, 9, 10], 'min child weight': [12, 20, 30], 'silent': [1],
'subsample': [0.5, 0.6, 0.7], 'colsample_bytree': [0.6, 0.7, 0.8], 'n_estimators': [5, 10, 20,
100]}
mod = GridSearchCV(modelo, parametros, n_jobs=4,  cv=4, scoring='accuracy', verbose=10,
refit=True)
mod.fit(X_train, Y_train)
```

```
# Melhores Parametros: {'colsample_bytree': 0.7, 'learning_rate': 0.05, 'max_depth': 8,
'min_child_weight': 12, 'n_estimators': 20, 'nthread': 4, 'objective': 'binary:logistic',
'silent': 1, 'subsample': 0.5}

# Aplicando o Modelo
model = XGBClassifier(**mod.best_params_)
model.fit(X_train, Y_train)
predict_train = model.predict(X_train)
predict_test = model.predict(X_test)

# Calculando o Score
print('métrica: acurácia')
print('score_treino: ' + str(round(accuracy_score(Y_train, predict_train),6) ))
print('score_teste: ' + str(round(accuracy_score(Y_test, predict_test),6) ))
print('melhores hiperparametros: ' + str(mod.best_params_))


# ## Modelo: XGBClassifier com Score: roc_auc
modelo = XGBClassifier()

# Tunning:
parametros = {'nthread': [4], 'objective': ['binary:logistic'], 'learning_rate': [0.05, 0.1,
0.15], 'max_depth': [4, 8, 9, 10], 'min_child_weight': [12, 20, 30], 'silent': [1],
'subsample': [0.5, 0.6, 0.7], 'colsample_bytree': [0.6, 0.7, 0.8], 'n_estimators': [5, 10, 20,
100]}
mod = GridSearchCV(modelo, parametros, n_jobs=4,  cv=4, scoring='roc_auc', verbose=10,
refit=True)
mod.fit(X_train, Y_train)

# Melhores Parametros: {'colsample_bytree': 0.8, 'learning_rate': 0.05, 'max_depth': 8,
'min_child_weight': 12, 'n_estimators': 10, 'nthread': 4, 'objective': 'binary:logistic',
'silent': 1, 'subsample': 0.5}

# Aplicando o Modelo
model = XGBClassifier(**mod.best_params_)
model.fit(X_train, Y_train)
predict_train = model.predict(X_train)
predict_test = model.predict(X_test)

# Calculando o Score
print('métrica: roc_auc')
print('score_treino: ' + str(round(roc_auc_score(Y_train, predict_train),6) ))
print('score_teste: ' + str(round(roc_auc_score(Y_test, predict_test),6) ))
print('melhores hiperparametros: ' + str(mod.best_params_))
```

APÊNDICE F - Código para visualizar correlação entre variáveis

Código disponível na íntegra em: https://github.com/eccoutos/TCC_FIA.git

```
# Importando Modulos e Pacotes
import numpy as np
import pandas as pd

# Atribuindo parametros de entrada
target = 'result'

# Importando os dados de entrada
df = pd.read_csv('C:/Users/rcosin/Desktop/FIA/TCC/Rim/chronic_kidney_disease_full.csv', sep =
',', decimal = '.')

import matplotlib.pyplot as plt

df.corr().style.format("{:.2}").background_gradient(cmap=plt.get_cmap('coolwarm'), axis=1)
```