



UNIVERSIDADE FEDERAL DO CEARÁ

Campus de Sobral

Curso de Engenharia de Computação

Ken Esparta Ccorahua

**APLICAÇÃO DE UM ALGORITMO PARA EXTRAÇÃO DE
REGRAS DE UMA REDE NEURAL ARTIFICIAL NO PROCESSO
DE MINERAÇÃO DE DADOS**

Sobral – CE

2016

Ken Esparta Ccorahua

**APLICAÇÃO DE UM ALGORITMO PARA EXTRAÇÃO DE REGRAS DE UMA REDE
NEURAL ARTIFICIAL NO PROCESSO DE MINERAÇÃO DE DADOS**

Trabalho de Conclusão de Curso apresentado
como requisito parcial para obtenção do grau de
Engenheiro de Computação na Universidade Fe-
deral do Ceará, *Campus Sobral*.

Orientador

Prof. Me. Fernando Rodrigues de Almeida Júnior

Coorientador

Prof. Dr. Márcio André Baima Amora

Sobral – CE, dezembro de 2016

Aos meus queridos pais.

À vida do meu amor e ao amor da minha vida, Diane.

Agradecimentos

Agradeço a Deus, por me permitir existir neste universo.

Agradeço a toda minha família e, em especial aos meus pais que sempre me dizem “Nunca desperdice o tempo”, à minha tia Gloria que foi como minha segunda mãe, ao meu primo Ronald que foi como um irmão para mim e aos meus dois irmãos que sempre estão no meu pensamento.

Agradeço ao professor Fernando, o professor Jarbas e o professor Márcio, que me deram um pouco do seu tempo para me encaminhar na pesquisa e corrigir aqueles errinhos de português.

Agradeço também a todos os professores do curso da Engenharia de Computação e da Engenharia Elétrica pelo tempo e ensinamentos.

Agradecimentos especiais ao professor Iális, por me apoiar quando cheguei neste campus e a professora Jéssica por ser a melhor: *“The best tia ever!”*.

Também agradeço a todos os meus colegas do Brasil e do Peru. Especial agradecimento ao Pedro Mateus que colaborou com a realização deste trabalho.

Por fim, agradeço a Diane, minha querida e amada. Que me brindou o seu apoio moral e carinho durante todos estes anos. Iluminou meu caminho com um amor verdadeiro.

Resumo

Utiliza-se uma rede de Petri para modelar a base teórica utilizada no trabalho.

Abstract

Lista de Figuras

1	Estrutura do modelo de adquisição do conhecimento das ferramentas teóricas e técnicas relacionadas com o presente trabalho.	12
2	Principais componentes da família da SC.	13
3	Desenvolvimento da SC.	14
4	Arquitetura básica de uma rede neural.	18
5	A ficha encontra-se no estado “Aprendizado de máquina”. As fichas vermelhas indicam que somente pode existir uma ficha ou no estado “Usa Lógica Fuzzy” ou no estado “Usa RNA”.	19
6	Processo de classificação de separação linear.	21
7	Processo de classificação de separação não linear.	21
8	A ficha encontra-se no estado “Mineração de dados”. O significado das fichas de cor vermelha são explicados na (FIGURA 5).	23
9	Uma visão geral dos passos que compõem o processo KDD.	23
10	Metodologia SEMMA proposto pela SAS.	24
11	Metodologia CRISP-DM, utilizada no presente trabalho.	25
12	Distribuição em quatro níveis da metodologia CRISP-DM.	25
13	Rede de Petri relacionada com o conhecimento teórico abordado até a árvore de decisão.	27
14	Tela de início do WEKA.	28
15	Exemplo de um arquivo do tipo .arff que contém os dados utilizados para a mineração de dados.	29
16	Na figura mostra-se a árvore de decisão resultante da aplicação do algoritmo J48 no conjunto de dados da planta iris com sua respectiva representação. . . .	30
17	Rede de Petri relacionada com o conhecimento teórico abordado até a extração de regras.	31
18	Os subprocessos da “Compreensão do projeto”.	35
19	Arranjo de sensores utilizado para captar informação do odor do ambiente. . .	35
20	Fluxograma resumido do plano do projeto de mineração de dados.	37
21	Os subprocessos da “Compreensão dos dados”.	38

22	Diagrama relacional para a massa de dados fornecida pelo cliente, disponível em (HUERTA; HUERTA, 2016).	38
23	Função sigmoide da equação (EQUAÇÃO 6), para diferentes valores de α . Essa função será utilizada como função de ativação nos neurônios ocultos da rede neural artificial.	41
24	Histograma de frequências para o sensor R1.	42
25	Histograma de frequências para o sensor R2.	42
26	Histograma de frequências para o sensor R3.	42
27	Histograma de frequências para o sensor R4.	43
28	Histograma de frequências para o sensor R5.	43
29	Histograma de frequências para o sensor R6.	43
30	Histograma de frequências para o sensor R7.	44
31	Histograma de frequências para o sensor R8.	44
32	Os subprocessos da “Preparação dos dados”.	45
33	Diagrama relacional com os atributos necessários para poder realizar a mineração de dados.	46
34	Tabela integrada, resultado da junção das tabelas “Metadados” e “Dados_sensor”.	47
35	Os subprocessos da “Modelagem”.	49
36	Captura de tela do WEKA, utilizando o algoritmo J48.	49
37	Arquitetura da rede neural artificial projetada para a mineração da massa de dados fornecida pelo cliente. É a partir desta rede neural artificial que se utiliza o algoritmo RX.	50

Lista de Tabelas

1	Pesquisa realizada com 200 pessoas sobre qual é a metodologia favorita utilizada nos projetos de mineração de dados, nos anos 2007 e 2014.	26
2	Metadados da massa de dados da planta iris.	29
3	Recursos utilizados na mineração de dados.	36
4	Dez tuplas aleatórias da tabela “Metadados”.	39
5	Doze tuplas aleatórias da tabela “Dados_sensor”, na tabela somente mostram-se os valores de condutância para R1 e R8.	40
6	Dados estatísticos dos valores das condutâncias dos oito resistores utilizados na medição, tal que μ representa o valor da média populacional, σ é o desvio padrão, R_{max} e R_{min} são o valores da condutância máxima e condutância mínima medida por um dos sensores, respectivamente.	40
7	Nove tuplas aleatórias da tabela “Dados_sensor_modificado”, mostrada na (FIGURA 34), para ser utilizada pelo algoritmo J48.	48
8	Nove tuplas aleatórias da tabela “Dados_sensor_modificado”, mostrada na (FIGURA 34), para ser utilizada pelo algoritmo RX.	48

Lista de Algoritmos

1	Estrutura principal de um EP, disponível em (MICHALEWICZ, 1996)	15
2	Pseudocódigo para o algoritmo RX, (LU; SETIONO; LIU, 1996; HRUSCHKA; EBECKEN, 1999).	33
3	<i>Leader Algorithm</i> , disponível em (MICHALEWICZ, 1996).	51

Lista de abreviaturas e siglas

ARFF	Attribute-Relation File Format
CRISP-DM	Cross-Industry Standard Process for Data Mining
EC	Evolutionary Computing
EP	Evolution Program
FK	Foreign Key
FS	Fuzzy Systems
PK	Primary Key
HC	Hard Computing
JOONE	Java Object Oriented Neural Network
KDD	Knowledge Discovery From Data
LM	Levenberg–Marquardt
ML	Machine Learning
MLP	Multilayer Perceptron
NN	Neural Networks
PR	Probabilistic Reasoning
RBF	Radial Basis Function
RNA	Redes Neurais Artificiais
RX	Rule Extraction
SC	Soft Computing
SEMMA	Sample, Explore, Modify, Model and Assess
WEKA	Waikato Environment for Knowledge Analysis

Sumário

1	Introdução	8
1.1	Objetivo	9
1.1.1	Objetivo geral	9
1.1.2	Objetivos específicos	9
2	Justificativa	10
3	Trabalhos relacionados	11
4	Fundamentação teórica	12
4.1	Soft computing	12
4.1.1	Computação Evolucionária	14
4.1.2	Lógica Difusa	15
4.1.3	Raciocínio Probabilístico	16
4.1.4	Rede Neural Artificial	16
4.2	Aprendizado de máquina	19
4.2.1	Modelo de classificação	20
4.2.2	Modelo de regressão	21
4.3	Mineração de dados	22
4.3.1	A metodologia KDD	23
4.3.2	A metodologia SEMMA	24
4.3.3	A metodologia CRISP-DM	24
4.4	Árvore de decisão	26
4.4.1	O software WEKA	28
4.4.2	O algoritmo J48	28
4.5	Extração de regras	29
5	Materiais e métodos	34
5.1	Compreensão do projeto	34
5.1.1	Determinar os objetivos do projeto	35
5.1.2	Avaliar a situação	36

5.1.3	Determinar o objetivo da mineração de dados	36
5.1.4	Realizar um plano do projeto	37
5.2	Compreensão dos dados	37
5.2.1	Coletar dados iniciais	37
5.2.2	Descrever os dados	38
5.2.3	Exploração dos dados	39
5.2.4	Verificação da qualidade dos dados	44
5.3	Preparação dos dados	45
5.3.1	Selecionar os dados	45
5.3.2	Limpar os dados	45
5.3.3	Construir dados	46
5.3.4	Integrar dados	46
5.3.5	Formatar os dados	47
5.4	Modelagem	47
5.4.1	Selecionar a técnica de modelagem	48
5.4.2	Construir o modelo	49
5.4.3	Avaliar o modelo	51
5.5	Avaliação	52
5.6	Desenvolvimento	52
6	Resultados	53
6.1	Utilizando o algoritmo J48	53
6.2	Utilizando o algoritmo RX	53
7	Conclusões e recomendações	55
7.1	Conclusões	55
7.2	Recomendações	56
Referências		57
Anexo A.1		60

1 Introdução

O processo de mineração de dados é a extração do conhecimento a partir de uma massa de dados. Existem diversas metodologias para realizar o processo de mineração de dados, no presente trabalho aborda-se a metodologia CRISP-DM (*Cross-Industry Standard Process for Data Mining*).

A Modelagem é um processo da metodologia CRISP-DM onde se definem as técnicas que auxiliam o processo de mineração de dados. Uma das técnicas utilizadas no presente trabalho está relacionada com as Redes Neurais Artificiais (RNA).

As RNA são consideradas caixas pretas porque se desconhece totalmente o seu funcionamento interno. É por esse motivo que elas não são utilizadas no processo de mineração de dados, porém, existem os algoritmos de extração de regras que permitem obter informação de uma RNA na forma de regras lógicas.

A teoria abordada é representada utilizando uma rede de Petri, de modo que os estados são as áreas da Ciência da Computação a serem atingidas e as transições são as técnicas e o embasamento teórico requeridos para passar ao estado seguinte.

Começa-se pela *Soft Computing* (SC), que é uma área da Ciência da Computação que permite resolver problemas com uma abordagem orientada à tolerância de imprecisões. Neste trabalho, tratam-se duas abordagens da SC: a primeira é o raciocínio probabilístico e a segunda são as RNAs. Segundo a rede de Petri, aborda-se a teoria relacionada com o aprendizado de máquina, que, juntamente com as técnicas de descobrimento de conhecimento, dão origem às técnicas de mineração de dados.

A massa de dados utilizada para este trabalho trata da a simulação de um nariz eletrônico. Oito sensores foram instalados em uma habitação devidamente isolada. Esses sensores mediram os valores de condutância (Ω^{-1}) na presença de um estímulo químico gasoso (odor) de duas diferentes classes: “banana” e “vinho”. Existe mais uma classe chamada “background”, que é a resposta dos sensores quando não existe estímulo químico.

O processo de mineração de dados é realizado utilizando a metodologia CRISP-DM juntamente com a aplicação de um algoritmo que extrai regras de uma RNA devidamente treinada, o algoritmo RX. O treinamento da RNA é realizado utilizando o algoritmo de *Levenberg–Marquardt*, que consiste na junção dos algoritmos de gradiente descendente e o método de *Gauss–Newton*.

É utilizado também um algoritmo que gera uma árvore de decisão diretamente a partir de uma massa de dados sem necessidade de projetar uma rede neural, esse é o algoritmo J48. Este é a implementação *open source* do algoritmo C4.5. Finalmente, comparam-se as regras lógicas obtidas pelo algoritmo RX e o algoritmo J48.

1.1 Objetivo

1.1.1 *Objetivo geral*

Mostrar uma aplicação direta da extração de regras de um perceptron de múltiplas camadas (MLP), com algoritmo de treinamento *Levenberg–Marquardt*, proposto em (LEVENBERG, 1944), no processo de mineração de dados.

1.1.2 *Objetivos específicos*

- * Mostrar a base teórica sobre a metodologia de mineração de dados CRISP-DM, proposta em (CHAPMAN *et al.*, 2000), e sua aplicação em uma massa de dados conhecida.
- * Comparar os métodos de classificação de árvore de decisão com o método de extração de regras chamado de algoritmo RX, proposto em (HRUSCHKA; EBECKEN, 1999), ambos aplicados no contexto da mineração de dados com a metodologia CRISP-DM.

2 Justificativa

Redes neurais artificiais supervisionadas não são amplamente utilizadas em aplicações de mineração de dados devido à dificuldade de interpretação de seus modelos, (HRUSCHKA; EBECKEN, 1999). É sabido também que uma rede neural artificial é considerada uma caixa preta, ou seja, é um sistema onde se conhece as entradas e as saídas, porém não se conhece o seu funcionamento interno.

Desta forma, o principal desafio no uso de redes neurais supervisionadas em aplicações de mineração de dados significa obter conhecimento explícito a partir desses modelos. Neste trabalho, desenvolve-se uma aplicação prática onde se utiliza um algoritmo de extração de regras de uma rede neural artificial. A modelagem dessa rede neural artificial é parte de um dos processos da metodologia CRISP-DM. Esta metodologia é utilizada amplamente para resolver problemas de mineração de dados.

Em resumo, neste trabalho mostra-se uma situação prática onde a extração de regras é aplicada diretamente no processo de mineração de dados. Como produto final do processo, obtém-se regras lógicas que podem ser interpretadas por um especialista. As regras lógicas, resultantes do algoritmo de extração de regras, serão comparadas com as regras lógicas obtidas a partir do algoritmo de árvore de decisão.

3 Trabalhos relacionados

No relatório técnico intitulado “*Mineração de Dados: Conceitos, Tarefas, Métodos e Ferramentas*”, referenciado em (CAMILO; SILVA, 2009), apresentam-se os conceitos chave, tarefas e métodos essenciais a serem aplicados na mineração de dados. Inclusive são mostradas ferramentas desenvolvidas para tornar o processo de mineração de dados menos técnico; entre essas ferramentas, encontra-se o software WEKA, que será utilizado no trabalho.

Em (CHOI *et al.*, 2014), o objetivo do estudo foi descobrir o conhecimento significativo entre duas doenças complexas e sintomas em dados clínicos utilizando árvore de decisão. Foi utilizado o algoritmo C5.0 (que é uma versão melhorada do algoritmo C4.5) para a modelagem de uma árvore de decisão a partir de dados de doenças.

As RNA são consideradas caixas pretas e aproximadores universais. No artigo intitulado “*Are Artificial Neural Networks Black Boxes?*”, referenciado em (BENÍTEZ; CASTRO; REQUENA, 1997), propõem-se métodos para extrair regras a partir uma rede neural artificial. Nesse artigo, apresenta-se uma interpretação das RNA para que elas não sejam mais vistas como caixas pretas. Esta interpretação é construída com base nas regras *fuzzy*, usando um novo operador de lógica *fuzzy*. Além disso, essa interpretação oferece um procedimento automatizado de aquisição de conhecimento.

O sistema proposto em (BEI *et al.*, 2005) é um Sistema de Suporte de Decisões aplicado para o gerenciamento das hospitalizações e terapias. Um dos principais objetivos do projeto é a extração de regras e a identificação das regras adequadas em termos de eficácia, qualidade e redução de custos, especialmente na perspectiva de tecnologias e medicamentos em rápido desenvolvimento.

Segundo (ZHONG; GUO; SONG, 2008), as redes neurais possuem alta acurácia no processo de classificação na descoberta de conhecimento. Utiliza-se o algoritmo RX (Rule Extraction) melhorado para poder extrair as regras de uma rede neural do tipo RBF (Radial Basis Function). O software JOONE (Java Object Oriented Neural Network) também é utilizado, nesse artigo, para otimizar as entradas da rede neural respectiva.

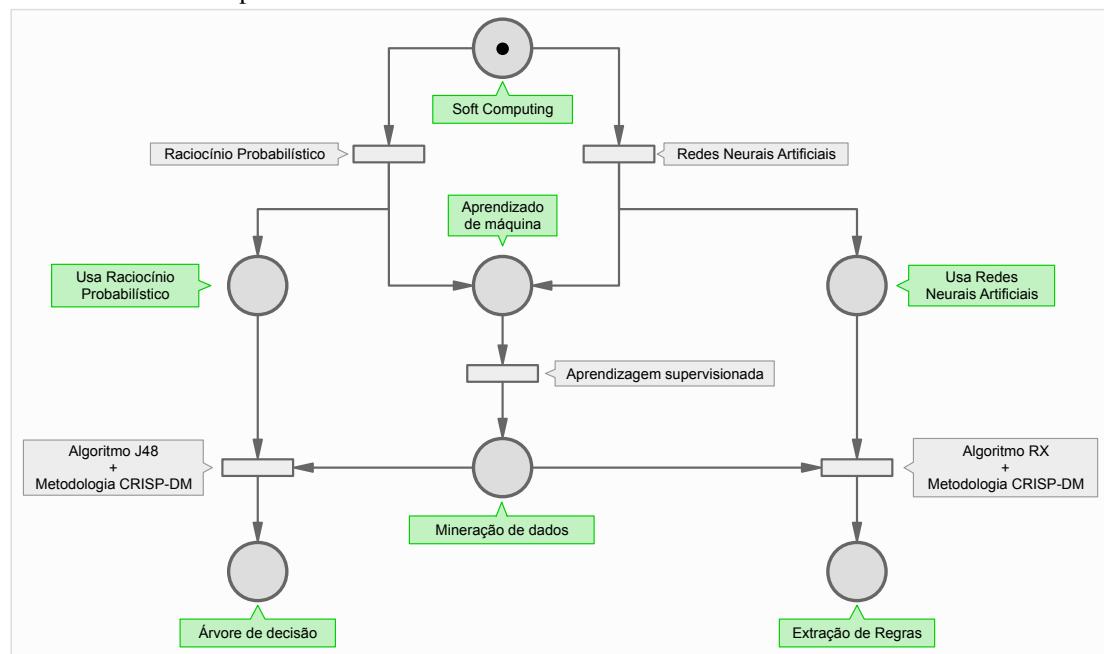
Em (HRUSCHKA; EBECKEN, 1999), afirma-se que a principal limitação da utilização de redes neurais supervisionadas em aplicações de mineração de dados, deve-se à dificuldade de entendimento da representação do conhecimento incorporado por estes modelos. Para superar tal limitação, faz-se uso de técnicas de extração de regras de redes neurais.

4 Fundamentação teórica

A seguir, apresentam-se breves definições sobre os assuntos teóricos relacionados com a monografia, utilizando-se uma abordagem *top-down*, como pode-se observar na (FIGURA 1), exibe-se uma rede de Petri que representa a montagem teórica do presente trabalho, na qual o estado inicial é o mais genérico e o estado final é o mais específico.

Na rede de Petri proposta, observa-se também que cada estado representa uma área das Ciências da Computação a qual possui ferramentas teóricas que são adotadas para que a ficha possa ativar as transições. Cada transição representa uma ferramenta que será adquirida do estado anterior antes de passar ao próximo. Em outras palavras, a ficha pode passar somente ao estado seguinte quando uma ferramenta específica do estado anterior é definida para ser utilizada.

Figura 1: Estrutura do modelo de adquisição do conhecimento das ferramentas teóricas e técnicas relacionadas com o presente trabalho.



Fonte: Autoria própria.

4.1 Soft computing

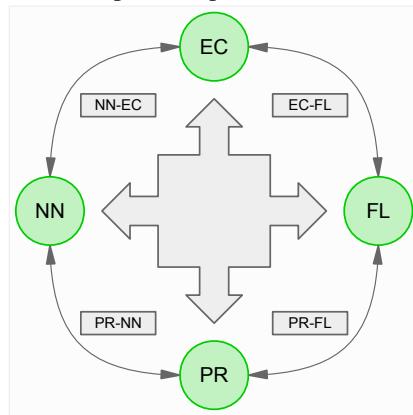
Observando a posição da ficha na (FIGURA 1), pode-se perceber que a primeira ferramenta teórica é a relacionada com a *Soft Computing*. Existem dois tipos de paradigmas computacionais, a *Soft Computing* (SC) e a *Hard Computing* (HC), ambos os termos foram estabelecidos pela primeira vez pelo professor L. A. Zadeh no ano 1996. O HC trata sobre modelos precisos em que as soluções são atingidas imediatamente. Por outro lado, a SC lida

com modelos de aproximações e dá soluções a problemas complexos (SIVANANDAM; DE-EPA, 2004). Como pode-se perceber, a HC é basicamente a computação convencional de modo que a solução dos problemas baseia-se nos princípios da precisão. Em contraste, o paradigma da SC trata sobre solucionar problemas utilizando modelos imprecisos que possuem certa porcentagem de aproximação.

Cataloga-se a SC em (MAIMON; ROKACH, 2007) como uma coleção de novas técnicas em inteligência artificial que exploram a tolerância para a imprecisão, incerteza, verdade parcial e manipulação de não linearidades para poder alcançar rastreabilidade, robustez e soluções de menor custo comparado com os métodos da HC, ou seja, apresenta-se uma coleção de ferramentas aptas para minerar a Web porque esta encaixa-se nas definições de imprecisão, incerteza e veracidade duvidosa.

A SC é composta por técnicas como as Redes Neurais (*Neural Networks*, NN), Computação Evolucionária (*Evolutionary Computing*, EC), Sistemas Difusos (*Fuzzy Systems*, FS) e Raciocínio Probabilístico (*Probabilistic Reasoning*, PR), como mostrado na (FIGURA 2).

Figura 2: Principais componentes da família da SC.

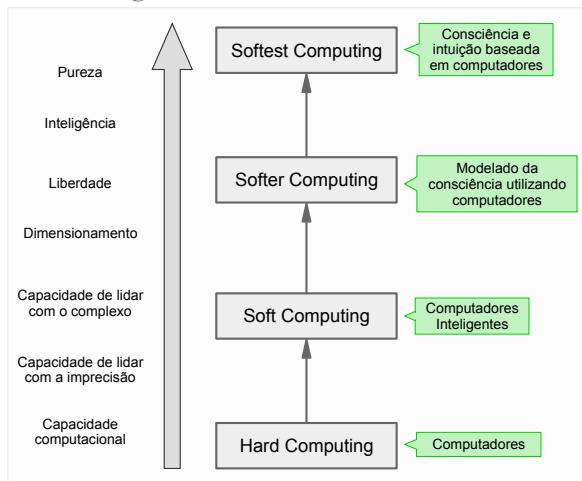


Fonte: Adaptado de (MANKAD, 2013).

As técnicas mostradas na (FIGURA 2), nem sempre estão isoladas. Elas podem ser combinadas para resolver um mesmo problema, e isso denomina-se hibridação.

Em (ZADEH, 1996) define-se a SC como uma nova abordagem da computação a qual é análoga à habilidade marcante da mente humana para pensar e aprender sobre um entorno incerto e impreciso. A SC, segundo (CHATURVEDI, 2008), é uma vertente da computação na qual pretende-se construir máquinas inteligentes e sábias. Pode-se observar na (FIGURA 3) o desenvolvimento da SC, começando pela computação convencional, até chegar ao extremo de pensamento puro por parte de um computador, ou seja, o objetivo final da SC é projetar e desenvolver um computador que possa agir de forma semelhante aos seres humanos.

Figura 3: Desenvolvimento da SC.



Fonte: Adaptado de (CHATURVEDI, 2008).

4.1.1 Computação Evolucionária

O domínio da computação evolutiva envolve o estudo das fundações e das aplicações de técnicas computacionais baseadas nos princípios da evolução natural. De um modo geral, as técnicas evolutivas podem ser vistas como métodos de busca ou como técnicas de otimização, (MANKAD, 2013). Uma tarefa abstrata pode ser resolvida de tal forma otimizada que a melhor solução pode ser encontrada como resultado da busca através de um espaço de soluções potenciais, (MICHALEWICZ, 1996).

A computação evolucionária está ligada estreitamente a um Programa Evolutivo (*Evolution Program*, EP). O termo EP utiliza-se para tudo que é relacionado com sistemas baseados na teoria da evolução natural, ou seja, que cada unidade do sistema possua as propriedades de reprodução, mutação e seleção.

Em (MICHALEWICZ, 1996), um EP é um algoritmo de probabilidade o qual mantém uma população de indivíduos, $P(t) = \{x_1^t, \dots, x_n^t\}$ para cada iteração t . Cada indivíduo representa uma solução potencial ao problema em questão e, em qualquer EP, é implementado como uma estrutura de dados S . Cada solução $x_i^t \in P(t)$ é avaliada para poder obter alguma medição da sua “capacidade”. Então, a nova população (iteração $t + 1$) é formada pela seleção dos indivíduos mais capazes. Os membros da nova população sofrem transformações por meio dos operadores “genéticos” para formar novas gerações. Existem transformações unárias (como a mutação $m_i : S \rightarrow S$), a qual criam novos indivíduos por uma mudança pequena em um único indivíduo. Existem outros tipos e transformações de alta ordem (como o *crossover* $c_j : S \times \dots \times S \rightarrow S$), cuja função é gerar novos indivíduos pela combinação das partes de dois

ou mais indivíduos de uma geração. A estrutura básica de um EP é observada no (ALGORITMO 1).

Algoritmo 1 Estrutura principal de um EP, disponível em (MICHALEWICZ, 1996).

```

1: procedure EVOLUTION PROGRAM
2:    $t \leftarrow 0$ 
3:   inicializar  $P(t)$ 
4:   avaliar  $P(t)$ 
5:   while (not condição de terminação) do
6:      $t \leftarrow t + 1$ 
7:     selecionar  $P(t)$  de  $P(t - 1)$ 
8:     alterar  $P(t)$ 
9:     avaliar  $P(t)$ 
10:   end while
11: end procedure
```

4.1.2 Lógica Difusa

Segundo (MANKAD, 2013), a lógica difusa (ou lógica *fuzzy*) é uma lógica de múltiplos valores introduzida por L. Zadeh em 1965. Os sistemas difusos são baseados na Lógica Difusa, uma generalização da lógica convencional que foi estendida para lidar com o conceito de verdade parcial e valores verdade entre "completamente verdadeiro" e "completamente falso". A lógica difusa fornece um mecanismo de inferência que permite que as capacidades aproximadas de raciocínio humano sejam aplicadas a sistemas baseados no conhecimento (AKERKAR; SAJJA, 2010).

Segundo (ZADEH, 1996), algumas das características principais da lógica difusa são as seguintes:

- * Extrair conhecimento é enxergado como um caso limitativo de conhecimento aproximado.
- * O conhecimento é interpretado como uma coleção de restrições *fuzzy* em uma coleção de variáveis.
- * A inferência é enxergada como um processo de propagação de restrições adaptáveis.
- * Qualquer sistema lógico pode ser tornado em um sistema *fuzzy* (*fuzzificação*).

Segundo (HANS-JIIRGEN, 2001), um conjunto *fuzzy* é assim definido: Sendo X uma coleção de objetos denotados por x , então um conjunto *fuzzy* (\tilde{A}) em X é o conjunto de

pares ordenados:

$$\tilde{A} = \{(x, \mu_\lambda(x) | x \in X)\}$$

tal que $\mu_\lambda(x)$ é chamado de função de associação ou grau de adesão de x em \tilde{A} que mapeia o conjunto X para o espaço de associação M . A imagem da função de associação é um subconjunto dos números reais não negativos cujo supremo é finito.

4.1.3 Raciocínio Probabilístico

O objetivo da lógica probabilística (também lógica de probabilidade e raciocínio probabilístico) é combinar a capacidade da teoria de probabilidade de lidar com a incerteza, com a capacidade da lógica dedutiva (MANKAD, 2013). O raciocínio probabilístico pode ser visto como uma maneira análoga ao raciocínio *fuzzy*, considerando a incerteza em lugar da *fuzzificação* como o conceito de aproximação que é aplicável. A abordagem Bayesiana é comumente utilizada.

Os axiomas de probabilidade, combinados com o teorema de Bayes, dão origem a um sistema de raciocínio completo, que inclui a lógica dedutiva tradicional como um caso especial. Mostra-se o teorema de Naive Bayes na (EQUAÇÃO 1):

$$P(B|A) = \frac{P(A|B) \cdot P(B)}{P(A)} \quad (1)$$

de modo que $P(B|A)$ lê-se como “a probabilidade de B , dado A ”. $P(B)$ é chamada de probabilidade *a priori* de B e $P(B|A)$ é chamada probabilidade *a posteriori* de B (COPIN, 2004).

O paradigma central do raciocínio probabilístico é identificar todas as variáveis relevantes do entorno e realizar um modelo probabilístico da interação dessas variáveis. O raciocínio (inferência) é então realizado através da introdução de evidências que são atribuídas às variáveis (deixando-as em estados conhecidos) e em seguida, são computadas as probabilidades de interesse que estão condicionadas por aquelas evidências, (BARBER, 2010).

4.1.4 Rede Neural Artificial

Uma rede neural artificial (ou simplesmente rede neural) é um modelo matemático simplificado de uma rede de neurônios biológicos. A unidade fundamental de uma rede neural é o neurônio que está interconectado com outros. Essas redes executam em paralelo uma tarefa global comum e possuem a habilidade de aprendizagem. Explica-se em (MANKAD, 2013) que

uma consequência da aprendizagem dos neurônios de uma rede neural é a aquisição de conhecimento de modo a torná-lo disponível para o uso. As características básicas de uma rede neural são o paralelismo inerente, acesso à informação local, a semelhança entre suas componentes e o aprendizado incremental.

A rede neural utilizada no trabalho é do tipo perceptron de múltiplas camadas (MLP), com algoritmo de treinamento de *Levenberg–Marquardt* (LM), que representa uma variação do algoritmo de propagação para atrás.

Perceptron de múltiplas camadas Consiste em um conjunto de unidades sensoriais que constituem a camada de entrada, uma ou mais camadas ocultas de nós computacionais e, por fim, uma camada de saída. O sinal de entrada se propaga através da rede camada por camada. Segundo (HAYKIN, 2008), uma MLP possui três características importantes

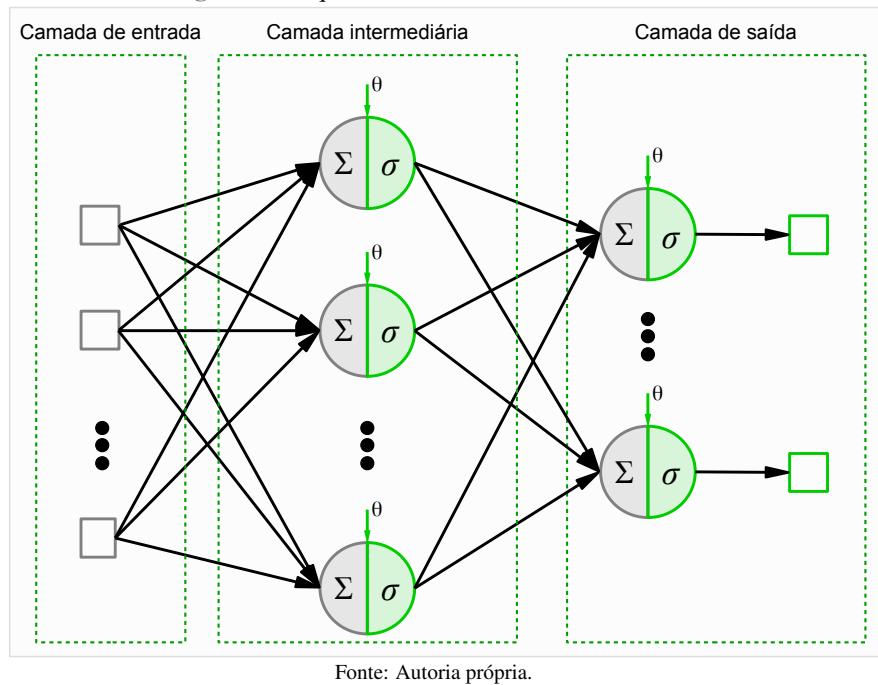
1. Alto grau de conectividade, isto que dizer que uma modificação na conectividade da rede, requer uma mudança na população das conexões sinápticas o dos seus valores de pesos correspondentes.
2. A rede possui uma ou mais camadas de neurônios ocultos, na sua camada intermediária, os quais capacitam o aprendizado da rede.
3. O modelo de cada neurônio da camada intermediária inclui uma função de ativação não linear e diferenciável em todo o seu domínio. Essa função de ativação comumente é a função sigmoide definida na (EQUAÇÃO 2):

$$y_j = \frac{1}{1 + e^{-v_j}} \quad (2)$$

de modo que v_j é a soma ponderada de todas as entradas sinápticas mais o bias do neurônio j e y_j é a saída do neurônio. Na (FIGURA 4) mostra-se a arquitetura básica de uma rede neural artificial, de forma que o símbolo Σ indica o somatório ponderado dos pesos com as entradas da rede neural (ou as saídas de cada neurônio), o σ é a função de ativação de cada neurônio e θ é o valor de bias.

Algoritmo de treinamento Levenberg-Marquardt O algoritmo de Levenberg-Marquardt (LM) é o algoritmo de otimização mais utilizado. Ele supera o método gradiente descendente simples e outros métodos de gradiente conjugado em uma ampla variedade de problemas,

Figura 4: Arquitetura básica de uma rede neural.



Fonte: Autoria própria.

(RANGANATHAN, 2004). O algoritmo de LM é uma técnica iterativa que localiza o mínimo de uma função que é expressa como a soma de quadrados de funções não-lineares, (LOURAKIS, 2005).

O algoritmo de LM é rápido e tem convergência estável. No campo das RNA, este algoritmo é adequado para o treinamento de pequenos e médios problemas, (YU; WILAMOWSK, 2011).

Como se afirma em (GAVIN, 2016), os métodos de mínimos quadrados não lineares envolvem uma melhoria iterativa dos valores dos parâmetros, a fim de reduzir a soma dos quadrados dos erros entre a função e os pontos de dados medidos. O método de ajuste de curva de LM é uma combinação de dois métodos de minimização: o método de gradiente descendente e o método de Gauss-Newton.

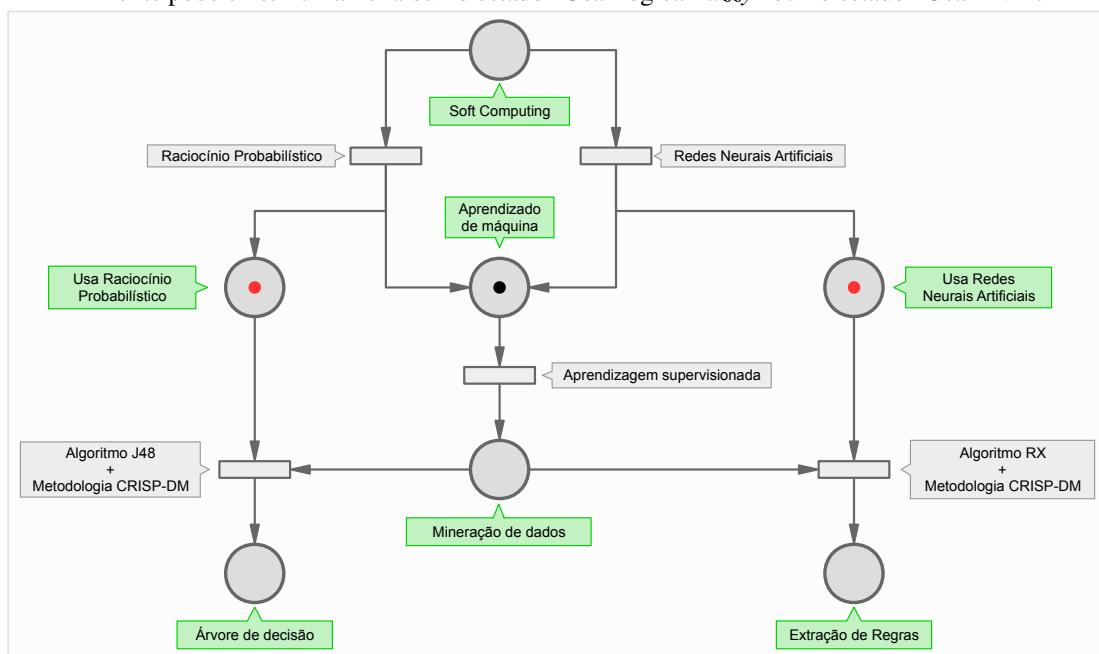
1. *Método gradiente descendente:* É um método de minimização geral que atualiza os valores dos parâmetros no sentido "descendente": direção oposta ao gradiente da função objetivo. Este método converge bem para problemas com funções objetivas simples porém, para problemas com vários parâmetros não é um método viável.
2. *Método de Gauss-Newton:* é um método para minimizar uma função objetivo de soma de quadrados. Presume-se que a função objetivo é aproximadamente quadrática nos parâmetros próximos à solução ótima. Para problemas de tamanho moderado, o método de

Gauss-Newton converge tipicamente muito mais rapidamente do que o método de gradiente descendente.

4.2 Aprendizado de máquina

Nesta seção trata-se sobre as técnicas que são adotadas do aprendizado de máquina (*Machine Learning*, ou ML). Utilizam-se duas abordagens para a comparação: árvore de decisão e extração de regras. A posição da ficha vermelha na (FIGURA 5) pode estar em dois possíveis estados mutuamente exclusivos.

Figura 5: A ficha encontra-se no estado “Aprendizado de máquina”. As fichas vermelhas indicam que sómente pode existir uma ficha ou no estado “Usa Lógica Fuzzy” ou no estado “Usa RNA”.



Fonte: Autoria própria.

O aprendizado automático ou aprendizado de máquina é um programa de computador que pode “aprender” de um conjunto de entradas disponíveis. Define-se, em (MURPHY, 2012), que o aprendizado de máquina é um conjunto de métodos que podem detectar automaticamente padrões em um determinado conjunto de dados e, posteriormente, utilizar esses padrões descobertos para predizer dados futuros. Aprendizado é, grosseiramente falando, o processo de converter experiência em habilidade ou conhecimento (SHALEV-SHWARTZ; BEN-DAVID, 2014). Aprender a partir de dados é o conceito fundamental do ML.

O foco do ML é a modelagem do aprendizado e a adaptação (atividades de animais e humanos) num computador. Os métodos do ML são denominados “sub-simbólicos” porque não existem símbolos ou manipulação deles envolvidos, em contraste com a Inteligência Ar-

tificial, em que o computador manipula símbolos que refletem o entorno (processo simbólico) (MARSLAND, 2015).

Tomando como ponto de partida que as máquinas aprendem a partir de dados, então, em (MARSLAND, 2015) a ML trata sobre como fazer computadores modificarem ou adaptarem as suas ações de modo que estas se tornem mais precisas. A precisão é medida por quão bem a escolha de ações refletem as escolhas corretas.

Saber em que momento precisa-se do ML é fundamental porque justifica a utilização dos conceitos de ML no trabalho. Em (SHALEV-SHWARTZ; BEN-DAVID, 2014), sugere-se que os conceitos de ML devem ser utilizados em tarefas realizadas por humanos ou animais. De acordo com isso, precisa-se do ML para poder modelar os algoritmos deste trabalho, de modo que utilizam-se dados gerados pela interação humana com a Web a partir de buscas.

De acordo com (MURPHY, 2012), o aprendizado de máquina divide-se, usualmente, em aprendizado supervisionado, aprendizado não supervisionado e aprendizado por reforço. Na abordagem supervisionada ou preditiva do ML, tem-se como objetivo o aprendizado mapeando as saídas a partir das entradas, dado um conjunto de treinamento. A segunda abordagem de ML, a não supervisionada ou descritiva, tem como objetivo encontrar padrões e correlações entre os dados. Finalmente, a abordagem de aprendizado por reforço, trata-se do modo de agir ou se comportar diante de uma situação de recompensa ou de punição.

A atividade de extrair conhecimento a partir da massa de dados utilizada neste trabalho, disponível em (HUERTA; HUERTA, 2016), justifica a escolha da utilização das ferramentas oferecidas pela abordagem supervisionada do aprendizado de máquina.

Até este ponto, foi escolhida a abordagem de uma rede neural artificial de treinamento supervisionado. Em (AMORA, 2013), afirma-se que o modelo de treinamento supervisionado pode ser dividido em dois grupos: o modelo de classificação e o modelo de regressão.

4.2.1 *Modelo de classificação*

Segundo (TAN; STEINBACH; KUMAR, 2006), é a tarefa de aprendizado de uma função objetivo (f) que mapeia cada conjunto de atributos (x) para cada uma das classes predefinidas (y). A função objetivo é chamada também como “modelo de classificação”.

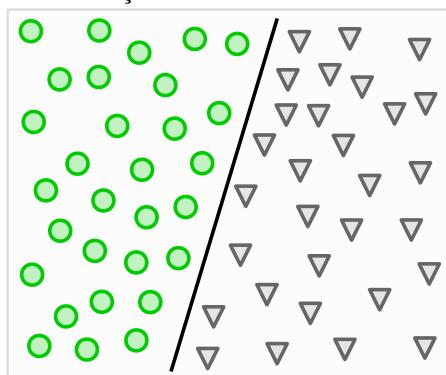
A classificação tem dois significados distintos. Podemos receber um conjunto de observações com o objetivo de estabelecer a existência de classes ou *clusters* nos dados, conhecido como aprendizado não supervisionado. Ou pode-se ter certeza de que há um número

definido de classes, e o objetivo é estabelecer uma regra segundo a qual possamos classificar uma nova observação em uma das classes existentes. Este método é conhecido como aprendizado supervisionado. Na Estatística, a aprendizagem supervisionada geralmente é referida como discriminação, isto é, o estabelecimento da regra de classificação a partir de dados corretamente classificados (MICHIE; SPIEGELHALTER; TAYLOR, 1994). Neste trabalho, quando se usa o termo classificação, refere-se à aprendizagem supervisionada.

O processo de classificação é a separação de dados por meio de uma fronteira do tipo linear ou não linear, como se argumenta em:

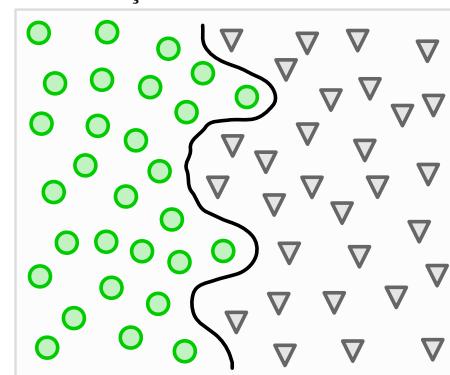
Durante o processo de classificação, os métodos podem realizar a separação dos dados de entrada através da determinação de uma fronteira representada por uma reta, situação onde os dados serão considerados linearmente separáveis, veja a (FIGURA 6). Caso a fronteira não possa ser representada por uma reta, os dados serão considerados como não linearmente separáveis e a fronteira será representada por curvas na dimensão normal do problema, veja a (FIGURA 7). (AMORA, 2013).

Figura 6: Processo de classificação de separação linear.



Fonte: Adaptado de (AMORA, 2013).

Figura 7: Processo de classificação de separação não linear.



Fonte: Adaptado de (AMORA, 2013).

4.2.2 *Modelo de regressão*

Em (CAMPBELL; CAMPBELL, 2008), afirma-se que a regressão é uma técnica estatística para determinar a relação linear entre duas ou mais variáveis. A regressão é usada principalmente para predição e inferência causal. Em consequência, nos modelos de regressão mostra-se como a variação em uma variável acontece em relação com a variação de outra variável. Segundo (AMORA, 2013), os modelos de regressão promovem o mapeamento do espaço de entrada em um domínio de valor real.

Em (DAVIDSON; MACKINNON, 2003), se afirma que o modelo de regressão mais elementar é o modelo de regressão linear simples, que pode ser expresso pela (EQUAÇÃO 3):

$$y_t = \beta_1 + \beta_2 X_t + \mu_t \quad (3)$$

de modo que o subíndice t indica as observações de uma amostra. O total de observações é denotado por n , então para uma amostra de tamanho n , o valor de t estará no intervalo $[1, n]$. Cada observação é definida através de uma variável dependente, denotada como y_t e a observação para uma única variável independente (ou variável explicativa) é denotada por X_t , para uma observação t . Os valores dos parâmetros β_1 e β_2 são chamadas de constantes da regressão e o valor do termo μ_t é chamado de erro não observado. Então, das cinco quantidades observadas na (EQUAÇÃO 3), duas são observadas (y_t e X_t) e três não são (β_1 , β_2 e μ_t). Os valores dos parâmetros que possuem subíndice t são próprios de cada observação, entretanto, os valores dos parâmetros que não possuem os subíndices são comuns a todas as n observações.

Afirma-se em (CAMPBELL; CAMPBELL, 2008), que é importante reconhecer que a análise de regressão é fundamentalmente diferente da constatação das correlações entre diferentes variáveis. A correlação determina a força da relação entre as variáveis, enquanto a regressão tenta descrever essa relação entre essas variáveis com mais detalhes.

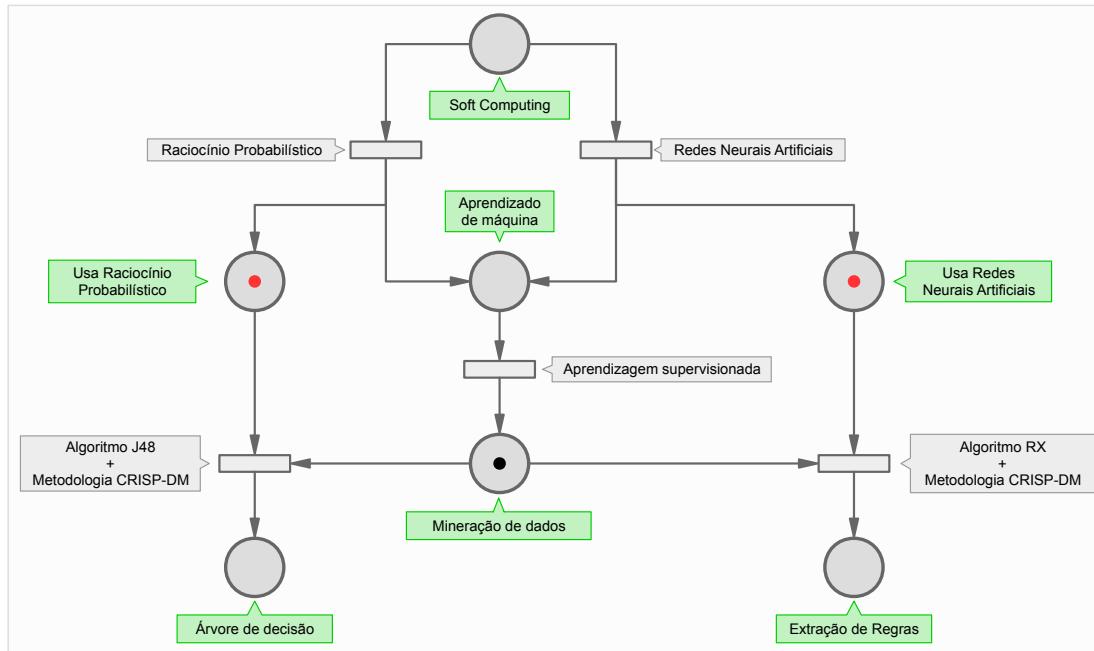
4.3 Mineração de dados

A mineração de dados é definida, segundo (ZAKI; JR., 2014), como o processo de descoberta intuitiva de novos padrões de interesse, assim como modelos preditivos, descritivos e compreensíveis. Até este ponto, tem-se as ferramentas teóricas ou da rede neural artificial ou raciocínio probabilístico, juntamente com a abordagem de aprendizado supervisionado, como se mostra na (FIGURA 8).

Em (RAJARAMAN; LESKOVEC; ULLMAN, 2012) descreve-se a mineração de dados como a descoberta de modelos a partir de dados. Existem duas abordagens: a estatística e a do aprendizado de máquina. Para o presente trabalho utiliza-se a abordagem de aprendizado de máquina, esta decisão justifica-se na (SUBSEÇÃO 4.2).

Uma analogia da mineração de dados faz-se em (HAN; PEI, 2012), com a obtenção de ouro puro a partir da extração de toneladas de rochas em uma mina. Ademais, enfatiza-se que o nome apropriado para mineração de dados é “conhecimento minerado a partir de dados”.

Figura 8: A ficha encontra-se no estado “Mineração de dados”. O significado das fichas de cor vermelha são explicados na (FIGURA 5).



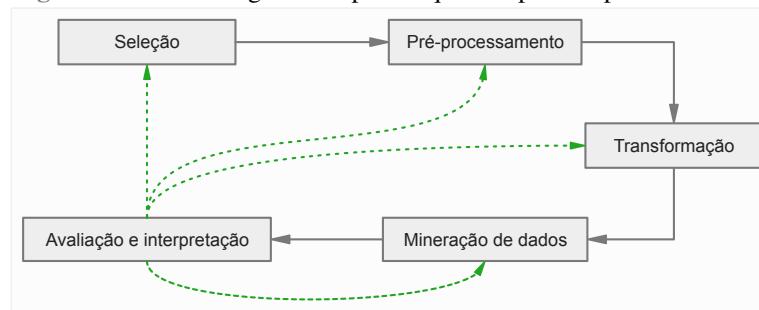
Fonte: Autoria própria.

Segundo (HAN; PEI, 2012), a descoberta de conhecimento a partir de dados (ou KDD, do inglês Knowledge Discovery from Data) é sinônimo de mineração de dados, não obstante em (SHAFIQUE; QAISER, 2014) coloca-se o KDD, juntamente com o CRISP-DM e o SEMMA (Sample, Explore, Modify, Model and Assess), como um tipo de processo relacionado com a mineração de dados. A seguir, expõe-se uma breve descrição sobre eles:

4.3.1 A metodologia KDD

É um modelo de processo que consiste na extração de conhecimentos escondidos a partir de um banco de dados. Deve ser um processo interativo e iterativo, e possui cinco passos de desenvolvimento que se podem observar na (FIGURA 9).

Figura 9: Uma visão geral dos passos que compõem o processo KDD.



Fonte: Adaptado de (FAYYAD; PIATECKI-SHAPIRO; SMYTH, 1996).

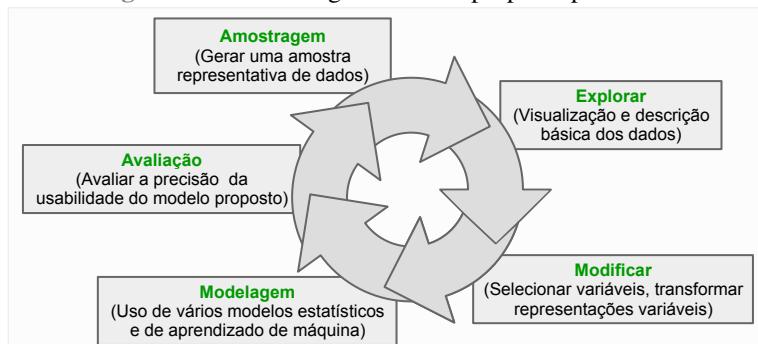
Em (ZAPATA; GIL, 2011), menciona-se que a mineração de dados existia como

uma fase do processo KDD. A mineração de dados só compreendia a aplicação de algoritmos de otimização e classificação, mas com o tempo evoluiu e atualmente refere-se ao processo completo de KDD.

4.3.2 A metodologia SEMMA

Do acrônimo Sample, Explore, Modify, Model and Assess; que foi desenvolvido pelo instituto SAS¹. Foca-se basicamente no desenvolvimento e manutenção de projetos de mineração de dados. Consiste em um ciclo altamente iterativo de cinco passos como pode-se verificar na (FIGURA 10).

Figura 10: Metodologia SEMMA proposto pela SAS.



Fonte: Adaptado de (BINUS, 2014).

4.3.3 A metodologia CRISP-DM

É um processo de mineração de dados que significa Cross-Industry Standard Process for Data Mining. Segundo (CHAPMAN *et al.*, 2000), o processo foi concebido no ano 1996 pelas empresas DaimlerChrysler², SPSS³ e NCR⁴. É uma ferramenta que está constituída por seis processos cílicos como se mostra na (FIGURA 11). Segundo a pesquisa de (PIATETSKY, 2014) no site KDnuggets, pode-se observar na (TABELA 1) que o CRISP-DM é o método mais utilizado nos anos 2007 e 2014.

Como se mostra na (FIGURA 12), a metodologia CRISP-DM é descrita em termos de um modelo de processo hierárquico, consistindo de conjuntos de tarefas descritas em quatro

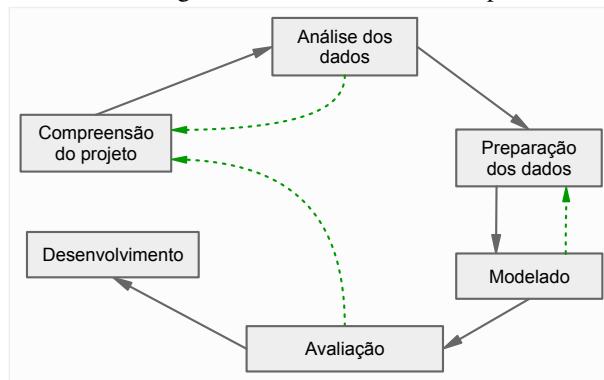
¹Statistical Analysis System, é o nome de uma empresa pioneira em Business intelligence. Disponível em <<http://www.sas.com>>

²É um fabricante de automóveis de passageiros e veículos comerciais. Disponível em <<https://www.daimler.com>>

³É um software para análise estatística de dados, disponível em <<http://www.ibm.com/analytics/us/en/technology/spss/>>

⁴É uma empresa de tecnologia especializada em produtos para o varejo e setores financeiros, disponível em <<http://www.ncr.com>>

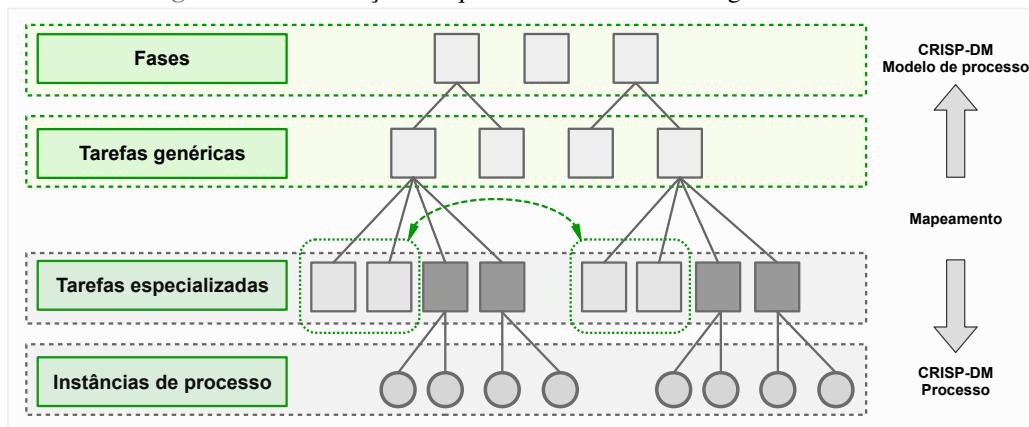
Figura 11: Metodologia CRISP-DM, utilizada no presente trabalho.



Fonte: Adaptado de (OLSON; DELEN, 2008).

níveis de abstração (de geral a específico): fases, tarefas genéricas, tarefas especializadas e instâncias de processos.

Figura 12: Distribuição em quatro níveis da metodologia CRISP-DM.



Fonte: Adaptado de (CHAPMAN *et al.*, 2000).

Tendo como referência a (FIGURA 12), no nível superior, o processo de mineração de dados é organizado em um número definido de fases; cada fase consiste em várias tarefas genéricas de segundo nível. Este segundo nível é chamado de genérico porque é destinado para cobrir todas as possíveis situações de mineração de dados. As tarefas genéricas devem ser o mais completas e estáveis possível. Completo significa abranger o processo de mineração de dados e suas possíveis aplicações. Estável significa que o modelo deve ser válido para novos desenvolvimentos que não foram previstos, (CHAPMAN *et al.*, 2000).

O nível das tarefas especializadas, na (FIGURA 12), é o lugar para descrever como as ações nas tarefas genéricas devem ser realizadas em determinadas situações específicas. Segundo (CHAPMAN *et al.*, 2000), no terceiro nível se descreve como essas tarefas se diferenciam umas com outras na forma de serem executadas em diversos contextos.

Finalmente, ainda na (FIGURA 12), tem-se no quarto nível as instâncias de pro-

cesso. Estas são registros das ações, decisões e resultados de um projeto de mineração de dados real. Uma instância de processo é organizada de acordo com as tarefas definidas nos níveis mais altos e representa, de forma específica, o que realmente acontece no projeto, (CHAPMAN *et al.*, 2000).

The fourth level, the process instance, is a record of the actions, decisions, and results of an actual data mining engagement. A process instance is organized according to the tasks defined at the higher levels, but represents what actually happened in a particular engagement, rather than what happens in general.

Tabela 1: Pesquisa realizada com 200 pessoas sobre qual é a metodologia favorita utilizada nos projetos de mineração de dados, nos anos 2007 e 2014.

Metodologia	Ano 2007 (%)	Ano 2014 (%)
CRISP-DM	42.0	43.0
Criada por mim	19.0	27.5
SEMMA	13.0	8.5
Outra	4.0	8.0
Processo KDD	7.3	7.5
Criada pela organização onde trabalho	5.3	3.5
Específica de domínio	4.7	2.0
Nenhuma	4.7	0.0

Fonte: Adaptado de (PIATETSKY, 2014).

Para passar ao seguinte estado da rede de Petri estabelecida, de acordo com a (FIGURA 8), é necessário adotar um tipo de processo de mineração de dados. Como é mencionado em (OLSON; DELEN, 2008), nas três metodologias que foram consideradas, não há obrigação de seguir de forma rígida seus respectivos passos. Afinal, optou-se por seguir a metodologia CRISP-DM.

4.4 Árvore de decisão

Como resultado da construção teórica, chegou-se ao último estado. Existem duas fichas porque o conhecimento é adquirido tanto do raciocínio probabilístico como da mineração de dados, isto se pode verificar na (FIGURA 13).

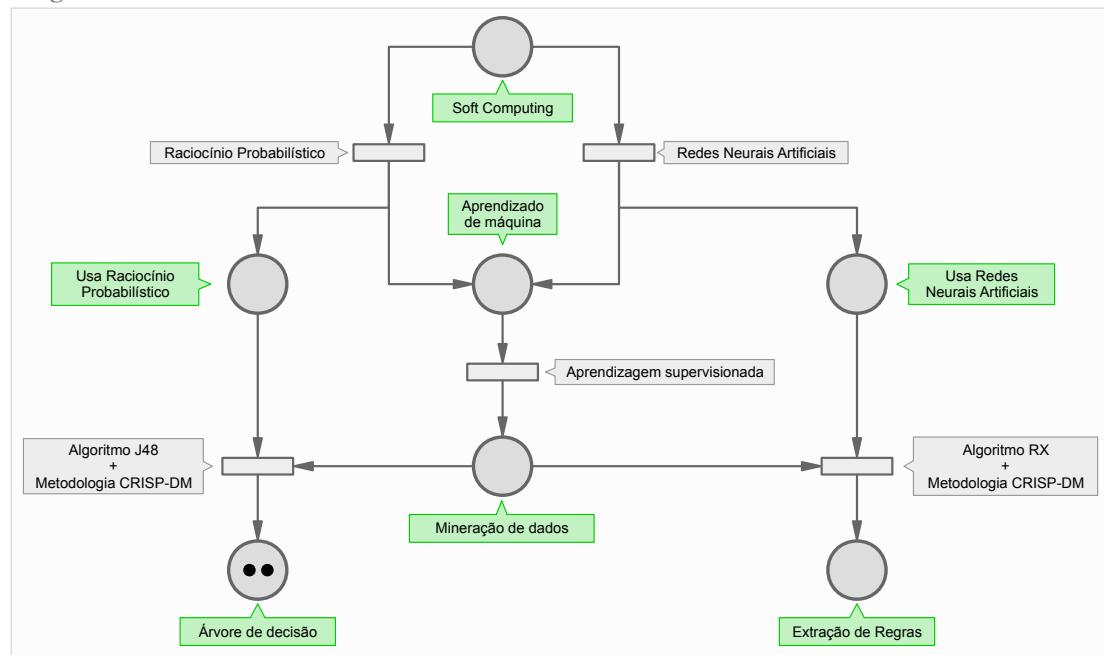
Segundo (ALPAYDM, 2004), uma árvore de decisão é uma estrutura de dados implementando a estratégia “dividir e conquistar”. É um método paramétrico eficiente o qual

pode ser utilizado tanto para classificação como para regressão. Quando uma árvore de decisão é usada para tarefas de classificação, é mais comumente referida como uma árvore de classificação. Quando é usado para tarefas de regressão, é chamado uma árvore de regressão. No presente trabalho se utilizará uma árvore de classificação.

Em (SAS, 2006), afirma-se que as árvores de decisão são uma forma simples, mas poderosa, de análise de variáveis múltiplas. Eles oferecem capacidades únicas para suplementar, complementar ou substituir

- * formas estatísticas tradicionais de análise (como regressão linear múltipla);
- * uma variedade de ferramentas e técnicas de mineração de dados (como redes neurais);
- * formas multidimensionais de relatórios e análises encontradas no campo da inteligência de negócios.

Figura 13: Rede de Petri relacionada com o conhecimento teórico abordado até a árvore de decisão.



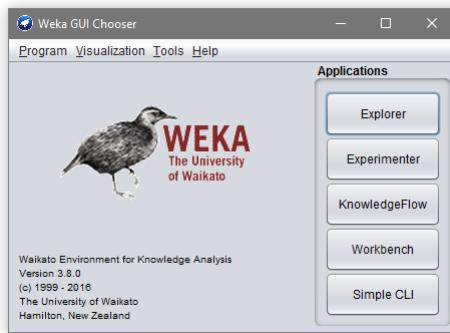
Árvores de decisão são modelos estatísticos que utilizam um treinamento supervisionado para a classificação e previsão de dados. Em outras palavras, em sua construção é utilizado um conjunto de treinamento formado por entradas e saídas, (PUC-RIO, 2005).

Uma árvore de decisão realiza a classificação de uma dada amostra de dados através de vários níveis de decisões para ajudar a chegar a uma decisão final. Essa sequência de decisões é representada em uma estrutura em árvore. A estrutura em árvore é usada na classificação de registros de dados desconhecidos.

4.4.1 O software WEKA

WEKA, disponível em <<http://www.cs.waikato.ac.nz/ml/weka/index.html>>, foi desenvolvido na Universidade de *Waikato*, na Nova Zelândia. O nome significa *Waikato Environment for Knowledge Analysis*. O sistema é escrito em Java e distribuído sob os termos da GNU General Public License. Na (FIGURA 14) mostra-se a tela de início do programa.

Figura 14: Tela de início do WEKA.



O WEKA é executado em praticamente qualquer plataforma e foi testado em sistemas operacionais Linux, Windows e Macintosh. Segundo (WITTEN; FRANK; HALL, 2011), WEKA fornece implementações de algoritmos de aprendizagem que você pode facilmente aplicar ao seu conjunto de dados. Também inclui uma variedade de ferramentas para transformar conjuntos de dados.

Neste trabalho, utilizou-se o formato de arquivo `.arff` que significa *Attribute-Relation File Format* (Formato de Arquivo de Relação de Atributos) disponível em <<http://www.cs.waikato.ac.nz/ml/weka/arff.html>>. O tipo ARFF foi desenvolvido pelo Projeto de Aprendizado de Máquina no Departamento de Ciência da Computação da Universidade de *Waikato* para uso com o software WEKA. Na (FIGURA 15) pode-se observar um exemplo desse formato de arquivo para a massa de dados utilizada no projeto de mineração de dados.

4.4.2 O algoritmo J48

O algoritmo J48 é uma implementação, de código aberto e escrito na linguagem de programação Java, do algoritmo C4.5 na ferramenta de mineração de dados WEKA. O algoritmo C4.5 é um programa que cria uma árvore de decisão baseada em um conjunto de dados de entrada rotulados. Este algoritmo foi desenvolvido por Ross Quinlan. As árvores de decisão geradas pelo C4.5 podem ser usadas para classificação, e por esta razão, C4.5 é muitas vezes referido como um classificador estatístico, (GHOLAP, 2012).

Figura 15: Exemplo de um arquivo do tipo .arff que contém os dados utilizados para a mineração de dados.

```

@RELATION dados_preparados

@ATTRIBUTE class {banana,vinho,background}
@ATTRIBUTE r1 REAL
@ATTRIBUTE r2 REAL
@ATTRIBUTE r3 REAL
@ATTRIBUTE r4 REAL
@ATTRIBUTE r5 REAL
@ATTRIBUTE r6 REAL
@ATTRIBUTE r7 REAL
@ATTRIBUTE r8 REAL

@DATA
banana,0.896340,0.864204,0.876274,0.864266,0.032686,0.106250,0.042064,0.045797
banana,0.878452,0.912125,0.925358,0.913614,0.033378,0.148516,0.024761,0.024682
banana,0.649617,0.415504,0.439591,0.411746,0.037033,0.199481,0.030146,0.045352
banana,0.792921,0.687938,0.698283,0.682982,0.024951,0.091974,0.034176,0.036763
banana,0.922375,0.965182,0.887113,0.893211,0.030109,0.123912,0.055095,0.010277
banana,0.926835,0.894647,0.905198,0.897277,0.033707,0.110228,0.043057,0.047215
banana,0.887890,0.861837,0.874734,0.848216,0.031280,0.101607,0.041829,0.045110
banana,0.850332,0.832996,0.848095,0.828702,0.030799,0.146340,0.042350,0.057940
banana,0.503819,0.589086,0.562953,0.588308,0.020583,0.182004,0.019000,0.021501

```

Fonte: Autoria própria.

O algoritmo J48 utiliza uma tecnologia *greedy* para induzir árvores de decisão para posterior classificação. O modelo de árvore de decisão é construído pela análise dos dados de treino e o modelo utilizado para classificar dados ainda não classificados. O J48 gera árvores de decisão, em que cada nó da árvore avalia a existência ou significância de cada atributo individual, (MARTINS; MARQUES; COSTA, 2009). A árvore de decisão gerada, pelo software WEKA, para o conjunto de dados da planta iris se mostra na (FIGURA 16). Segundo (FISHER; MARSHALL, 1988), o conjunto de dados da planta iris possui 150 tuplas, quatro atributos reais e um atributo de classe, como se mostra na (TABELA 2) .

Tabela 2: Metadados da massa de dados da planta iris.

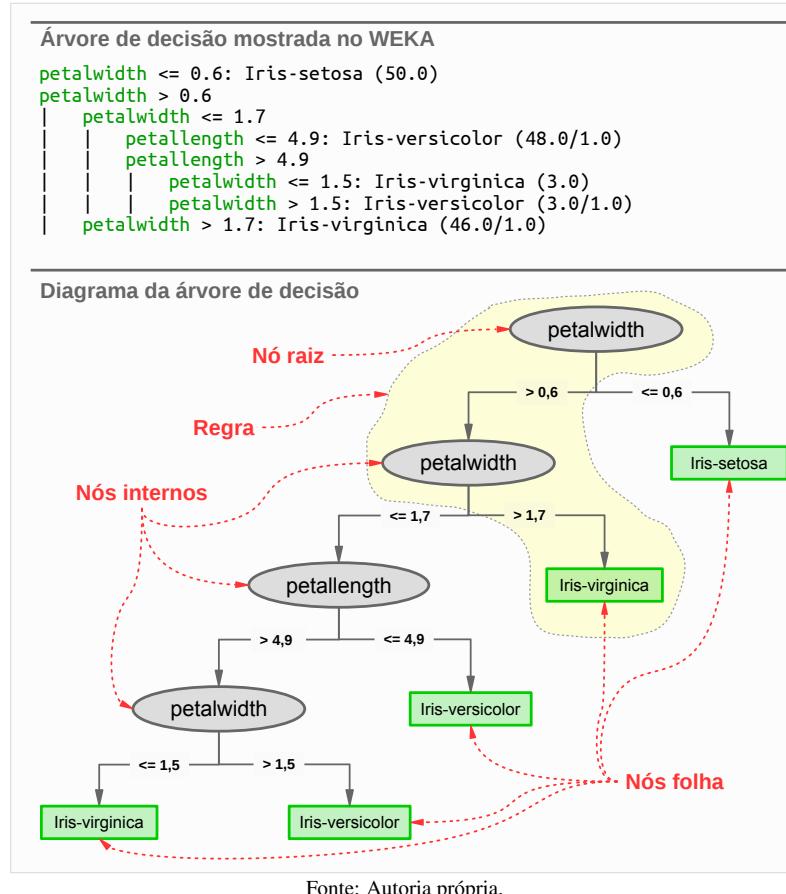
Atributo	Tipo
sepallength	REAL
sepalwidth	REAL
petallength	REAL
petalwidth	REAL
class	{Iris-setosa, Iris-versicolor, Iris-virginica}

Fonte: Adaptado de (FISHER; MARSHALL, 1988).

4.5 Extração de regras

Como resultado da construção teórica, chegou-se ao último estado. Existem duas fichas porque o conhecimento é adquirido tanto da rede neural artificial como da mineração de dados, isto se pode verificar na (FIGURA 17).

Figura 16: Na figura mostra-se a árvore de decisão resultante da aplicação do algoritmo J48 no conjunto de dados da planta iris com sua respectiva representação.



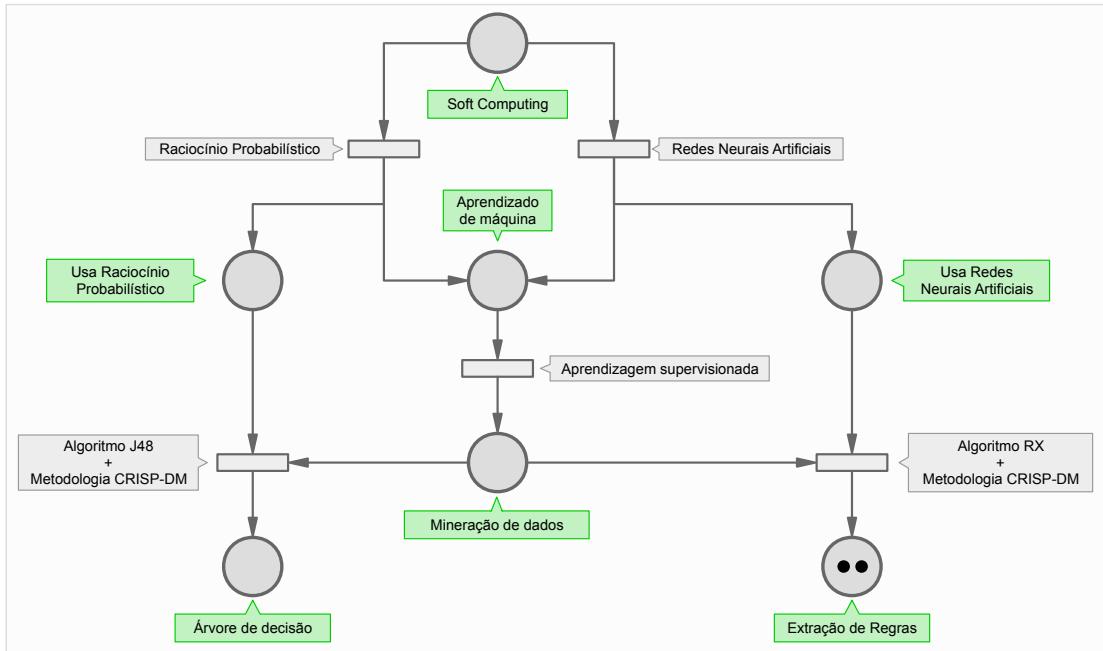
Em (AMORA *et al.*, 2007), afirma-se que as RNA representam modelos computacionais eficientes que são amplamente utilizados, porém, a maior dificuldade associada à utilização de RNA se deve ao fato das mesmas serem consideradas “caixas-pretas”. Isto significa o total desconhecimento sobre o comportamento interno da rede neural artificial.

Segundo (HRUSCHKA; EBECKEN, 1999), o processo de extração de regras é resumido da seguinte forma: dada uma rede neural treinada, o conjunto de exemplos de treinamento e um nível de fidelidade entre o conjunto de regras e o modelo de rede neural; deseja-se obter um conjunto de regras que modela a rede neural para um dado nível de fidelidade.

Segundo (AMORA *et al.*, 2007), sistemas baseados em Regras Fuzzy (SRFs) representam uma ferramenta poderosa para resolver problemas de controle (controlador *fuzzy*) e para modelar conhecimento. Em seguida, apresentam-se algumas formas de representação do conhecimento de regras extraídas de uma RNA:

1. *Regras convencionais:* Representam uma expressão em duas partes, os antecedentes e a

Figura 17: Rede de Petri relacionada com o conhecimento teórico abordado até a extração de regras.



Fonte: Autoria própria.

consequência. A forma dessas regras é:

$$\text{SE } \langle x_1 \leq t_1 \text{ E } x_2 \leq t_2 \dots \text{ E } x_n \leq t_n \rangle \text{ ENTÃO } \langle C \rangle$$

de modo que x_i e t_i são variáveis e C é uma classe associada a um conceito.

2. *Regras M de N*: São semelhantes às regras convencionais. Para que a premissa seja verdadeira, um número M de um total de N regras devem ser verdadeiras. A forma está dada por:

$$\text{SE } M \text{ DE } \langle x_1 \leq t_1 \text{ E } x_2 \leq t_2 \dots \text{ E } x_N \leq t_N \rangle \text{ SÃO VERDADEIROS, ENTÃO } \langle C \rangle$$

de modo que x_i e t_i são variáveis que representam as N regras lógicas e C é uma classe associada a um conceito.

3. *Regras oblíquas*: É um conjunto de regras tal que cada condição é dada na forma da desigualdade linear do tipo $\mathcal{R} : \sum k_j x_j \leq \eta$. A expressão $\sum k_j x_j$ representa um hiperplano, k_j é um coeficiente, x_j é o valor do atributo j e η é um valor de limiar. Para uma desigualdade linear \mathcal{R}_i , uma regra oblíqua está definida por:

$$\text{SE } \langle \mathcal{R}_1 \text{ E } \mathcal{R}_2 \dots \text{ E } \mathcal{R}_n \rangle \text{ ENTÃO } \langle C \rangle$$

Segundo (SETIONO; LIU, 1997), o hiperplano oblíquo é mais geral e pode reduzir substancialmente o número de condições necessárias para descrever a região de decisão. No presente trabalho, utilizando o algoritmo RX, se geram regras lógicas deste tipo.

4. *Regras fuzzy:* Essas regras utilizam funções de pertinência para lidar com verdades parciais, as regras *fuzzy* apresentam a forma:

$$\mathbf{SE} \left\langle \left(x_1 \text{ É } f_1 \right) \text{ E } \left(x_2 \text{ É } f_2 \right) \dots \text{ E } \left(x_n \text{ É } f_n \right) \right\rangle \mathbf{ENTÃO} \langle C \rangle$$

tal que f_i e C estão relacionados a conjuntos *fuzzy*.

5. *Autômato de estado finito:* Um autômato de estado finito forma parte do conjunto das Linguagens Regulares juntamente com as Gramáticas Regulares. Segundo (BOLOGNA, 2003), as gramáticas são formalismos que geram linguagens. Elas fornecem um conjunto de regras para permitir a formação de frases corretas. As RNA recorrentes têm sido usadas para lidar com problemas em gramáticas regulares. Em (OMLIN; GILES, 1995) se desenvolveu um algoritmo para extrair conhecimento na forma de autômatos de estados finitos, esse método foi usado para extraer regras gramaticais de uma linguagem regular.

Existem diversos algoritmos de extração de regras, porém, neste trabalho é abordado somente o algoritmo RX. Segundo (HRUSCHKA; EBECKEN, 1999), o algoritmo RX é baseado nas ativações das unidades ocultas. Primeiramente, realiza-se o treinamento da rede neural, de modo que se obtenha a taxa de classificação correta desejada. Eliminam-se, então, as conexões redundantes, seguindo-se uma análise dos valores das ativações para a obtenção de regras baseadas nestes valores. Basicamente, pode-se resumir o algoritmo RX nos passos do (ALGORITMO 2).

Algoritmo 2 Pseudocódigo para o algoritmo RX, (LU; SETIONO; LIU, 1996; HRUSCHKA; EBECKEN, 1999).

- 1: Processar os eventos pertencentes ao conjunto de treinamento, computando-se os valores de ativação das unidades ocultas da rede neural.
 - 2: Aplicar um algoritmo de agrupamento aos valores de ativação das unidades ocultas.
 - 3: Enumerar os valores de ativação discretizados e computar os respectivos valores de saída da rede. Gerar regras que descrevam as saídas da rede em termos dos valores de ativação discretizados em (2:).
 - 4: Para cada unidade oculta, enumerar os valores de entrada que conduzem aos respectivos valores de ativação das unidades ocultas, gerando regras que descrevam os valores discretizados das unidades ocultas em termos das entradas da rede neural.
 - 5: Combinar as regras obtidas em (3:) e (4:) para obter as regras que relacionam as variáveis dependentes às variáveis independentes.
-

5 Materiais e métodos

Se situando no contexto deste trabalho, o autor cumpre dois papéis diferentes: um de cliente e outro de analista de dados. Cumprindo o papel de cliente, dispõe-se de uma massa de dados relacionada com o artigo referenciado em (HUERTA *et al.*, 2016) e disponível no site da UCI⁵. Em contrapartida, cumprindo o papel de analista de dados⁶, aplica-se a metodologia CRISP-DM à massa de dados disponibilizada pelo cliente.

Tendo em conta o exposto no parágrafo prévio, apresenta-se a seguinte situação: o cliente deseja obter conhecimento relevante dos dados que possui⁷, então ele pede a um analista de dados para que possa realizar o trabalho relacionado com obtenção de conhecimento relevante a partir dos dados.

Depois de gerar o contexto no qual a ação de minerar dados é realizada, aprofundam-se nas etapas do processo da metodologia CRISP-DM para que a mineração dos dados do cliente seja satisfatória. Em (CHAPMAN *et al.*, 2000) mostra-se, de forma detalhada, o modelo de referência da metodologia CRISP-DM. Este modelo possui seis etapas que são aplicadas no processo de mineração do conjunto de dados proporcionados pelo cliente.

5.1 Compreensão do projeto

Segundo (OLSON; DELEN, 2008), o elemento-chave de um estudo de mineração de dados é saber para que serve o estudo. Isso começa com uma necessidade gerencial de novos conhecimentos e uma expressão do objetivo do projeto com relação ao estudo a ser realizado.

O primeiro objetivo do analista de dados é compreender o que o cliente realmente deseja fazer com os dados que dispõe. O objetivo do analista é descobrir, no início, fatores importantes que possam influenciar o resultado do projeto.

Esta fase inicial centra-se na compreensão dos objetivos e os requerimentos a partir de uma perspectiva de negócio, convertendo depois este conhecimento em uma definição de problema de mineração de dados e um plano preliminar concebido para atingir os objetivos. Na (FIGURA 18) mostram-se os quatro subprocessos relacionados à compreensão do projeto por

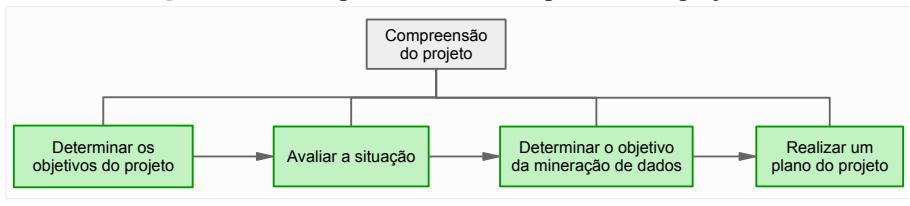
⁵É uma organização que dispõe, de forma gratuita, massas de dados com suas respectivas referências, disponível em <<http://archive.ics.uci.edu/ml/datasets.html>>

⁶Um analista de dados é um profissional que coleta, processa e executa análises estatísticas de dados. O do analista de dados é descobrir como os dados podem ser usados para responder a perguntas e resolver problemas, disponível em <<http://www.mastersindatascience.org/careers/data-analyst/>>.

⁷Ressaltando que o cliente não é quem gerou os dados. A massa de dados foi obtida de <<http://archive.ics.uci.edu/ml/datasets/Gas+sensors+for+home+activity+monitoring>>

parte do analista de dados.

Figura 18: Os subprocessos da “Compreensão do projeto”.



Fonte: Adaptado de (CHAPMAN *et al.*, 2000).

5.1.1 Determinar os objetivos do projeto

Neste trabalho, o cliente possui uma massa de dados relacionada com sensores químicos que podem detectar diferentes tipos de odor. Esses sensores são oito e formam um “nariz eletrônico” como mostrado na (FIGURA 19). Utilizando o nariz eletrônico, coletaram-se dados durante 537 dias em um ambiente fechado como mencionado em (HUERTA *et al.*, 2016), esses dados representam a variedade de eventos presentes em cenários de monitoramento de uma residência.

Figura 19: Arranjo de sensores utilizado para captar informação do odor do ambiente.



Fonte: Adaptado de (HUERTA *et al.*, 2016).

Neste caso particular, o cliente deseja saber se existe algum tipo de regra lógica que relate os valores da condutância (em Ω^{-1}) dos oito sensores e os tipos de odor. Os tipos de odores são dois: de banana e de vinho, ademais existe outro tipo chamado de “*background*” que é inodoro. Na (SUBSEÇÃO 5.2) são apresentados, de forma detalhada, os metadados e os dados referentes ao banco de dados fornecido pelo cliente.

5.1.2 Avaliar a situação

Esta tarefa envolve uma verificação mais detalhada de todos os recursos, restrições, suposições e outros fatores que devem ser considerados na determinação da meta de análise de dados e do plano do projeto. Na (TABELA 3) mostram-se os recursos utilizados para poder realizar a mineração de dados do banco de dados fornecido pelo cliente.

Tabela 3: Recursos utilizados na mineração de dados.

Recurso	Descrição
Dados	Massa de informação que foi fornecida pelo cliente do experimento de um nariz eletrônico, disponível em (HUERTA; HUERTA, 2016).
Matlab	É utilizado na mineração de dados para treinar a rede neural e executar o algoritmo RX.
WEKA	É um programa utilizado na mineração de dados para gerar uma árvore de decisão a partir da massa de dados fornecida pelo cliente.

Fonte: Autoria própria.

A principal restrição da mineração de dados é que não será realizada em tempo real, devido a que o experimento de onde se extraíram os dados já foi finalizado.

5.1.3 Determinar o objetivo da mineração de dados

O objetivo principal da mineração de dados é mostrar a aplicação prática do algoritmo RX (que é um método de extração de regras) na mineração.

Outro objetivo é obter diversas regras lógicas que relacionem os oito sensores químicos utilizados na medição com as respectivas saídas: odor de banana, odor de vinho e inodoro (*background*). As regras lógicas que são obtidas são da forma oblíqua:

SE ⟨expressão lógica⟩ **ENTÃO** ⟨conclusão⟩

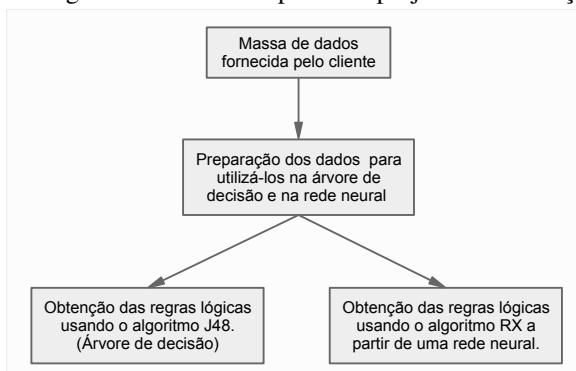
Depois de obter as regras lógicas utilizando o algoritmo RX, estas são comparadas com as obtidas pelo algoritmo J48.

5.1.4 Realizar um plano do projeto

O plano do projeto baseia-se no pedido do cliente. Se extrai conhecimento relevante a partir da massa de dados obtida da medição dos sensores, como exposto na (SUBSEÇÃO 5.1). Na (FIGURA 20) se mostra o fluxograma das tarefas realizadas no projeto.

A obtenção das regras lógicas será utilizando uma rede neural do tipo MLP e, em seguida, será aplicado o algoritmo RX modificado, disponível em (HRUSCHKA; EBECKEN, 1999).

Figura 20: Fluxograma resumido do plano do projeto de mineração de dados.



Fonte: Autoria própria.

5.2 Compreensão dos dados

A massa de dados na qual vai ser aplicada a metodologia CRISP-DM está relacionada com sensores químicos que podem detectar diferentes tipos de odor. No caso deste projeto, esses tipos de odores são dois: o de banana e o de vinho. Existe também o tipo inodoro que é chamado de “*background*”. Na (FIGURA 21) mostram-se os quatro subprocessos relacionados com compreensão dos dados.

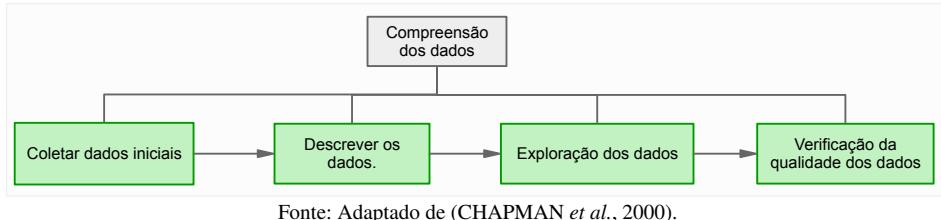
5.2.1 Coletar dados iniciais

O processo da coleta de dados já foi realizado pelos autores, em (HUERTA *et al.*, 2016). A coleta foi feita durante 537 dias (desde 17 de fevereiro do ano 2013 até 5 de junho de 2015) em um ambiente fechado. Logo após foi aplicada uma subamostragem para uma amostra por minuto e por sensor. Com a subamostragem obtiveram-se 919 438 tuplas, que representam os valores medidos pelos sensores em diferentes intervalos de tempo para os três tipos de casos: odor de banana, odor de vinho e inodoro.

No caso deste trabalho, não se realiza a mineração de dados em tempo real, ou seja,

a massa de dados obtida pelo cliente é estática e não será alimentada posteriormente com novos dados.

Figura 21: Os subprocessos da “Compreensão dos dados”.

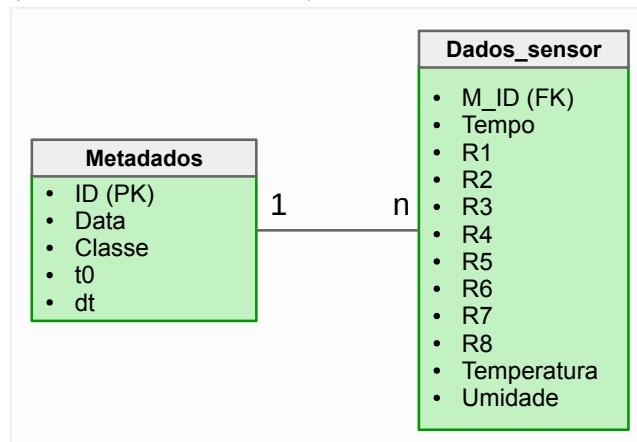


Fonte: Adaptado de (CHAPMAN *et al.*, 2000).

5.2.2 Descrever os dados

O autor, em (HUERTA *et al.*, 2016), disponibiliza duas fontes de dados: os metadados e os dados em si. Para poder visualizar as duas fontes de dados é necessário elaborar um diagrama objeto relacional, como se mostra na (FIGURA 22).

Figura 22: Diagrama relacional para a massa de dados fornecida pelo cliente, disponível em (HUERTA; HUERTA, 2016).



Fonte: Autoria própria.

O atributo “ID” da tabela “Metadados” é a chave primaria (PK) e são inteiros que representam números de 0 a 99, representando as cem diferentes induções dos dois tipos de gases (banana e vinho) mais o inodoro. Na (TABELA 4), mostram-se as dez primeiras tuplas da tabela Metadados. O atributo “Data” representa a data na qual foi feita o experimento para uma classe determinada. O atributo “Classe” representa as três classes consideradas no experimento: banana, vinho e “background”. O atributo “t0” significa o tempo em horas em que o experimento começou. O atributo “dt” é o intervalo que o respectivo experimento durou.

O atributo “M_ID” da tabela “Dados_sensor” é a chave estrangeira (FK) e representa as *n* ocorrências de cada ID da tabela Metadados. O atributo “Tempo” representa o início do

Tabela 4: Dez tuplas aleatórias da tabela “Metadados”.

ID	Data	Classe	t0 (h)	dt (h)
0	07-04-15	banana	13.49	1.64
1	07-05-15	wine	19.61	0.54
2	07-06-15	wine	19.99	0.66
3	07-09-15	banana	6.49	0.72
4	07-09-15	wine	20.07	0.53
5	07-10-15	banana	16.38	0.80
6	07-11-15	wine	11.19	0.95
7	07-12-15	banana	8.17	0.49
:	:	:	:	:
69	08-03-15	background	15.42	0.98
70	08-04-15	background	11.27	0.40
:	:	:	:	:

Fonte: Adaptado de (HUERTA; HUERTA, 2016).

experimento, em horas; o tempo negativo significa que o experimento começou antes do tempo “t0” estabelecido na tabela “Metadados”. Os valores de “R1” até “R8” representam a variação da condutância (em Ω^{-1}) dos sensores⁸. Os atributos “Temperatura” e “Umidade” representam, respectivamente, o valor da temperatura em Celsius e a porcentagem de umidade medidas no instante⁹. Na (TABELA 5), mostram-se as dez primeiras tuplas da tabela “Dados_sensor”.

5.2.3 Exploração dos dados

Para poder preparar os dados apropriadamente, eliminam-se alguns atributos das tabelas “Metadados” e “Dados_sensor”. Para analisar a correlação entre os valores da condutância medida pelos sensores, considera-se que não são necessários os atributos “Temperatura”, “Umidade” e “Tempo” da tabela “Dados_sensor” e os atributos “Data”, “t0” e “dt” da tabela “Metadados”. Em seguida, juntam-se os dados das duas tabelas mantendo as classes “banana”, “wine” e “background”, o procedimento é detalhado na (SUBSEÇÃO 5.3).

Utilizando o software WEKA, (HALL *et al.*, 2009), determina-se o valor mínimo,

⁸Os sensores utilizados foram do tipo MOX, disponibilizados por Figaro, disponível em (HUERTA; HUERTA, 2016).

⁹A temperatura e umidade foram medidos usando sensor Sensirion SHT75, disponível em (HUERTA; HUERTA, 2016).

Tabela 5: Doze tuplas aleatórias da tabela “Dados_sensor”, na tabela somente mostram-se os valores de condutância para R1 e R8.

M_ID	Tempo (h)	R1(Ω^{-1})	...	R8(Ω^{-1})	Temperatura (C°)	Umidade (%)
0	-0.999750	12.862100	...	8.739010	26.225700	59.052800
0	-0.999472	12.861700	...	8.739080	26.230800	59.029900
0	-0.999194	12.860700	...	8.739150	26.236500	59.009300
59	-0.482428	12.681200	...	3.745130	26.940100	60.291100
59	-0.482150	12.681500	...	3.745240	26.940100	60.291500
59	-0.481872	12.681800	...	3.745340	26.940100	60.291900
90	-0.995389	12.963200	...	7.543210	27.896500	49.630500
90	-0.995103	12.963500	...	7.542990	27.896900	49.631100
90	-0.994825	12.963400	...	7.542920	27.896200	49.634200
99	1.672666	12.620200	...	7.059780	27.732100	54.127700
99	1.672944	12.620000	...	7.059660	27.730900	54.134200
99	1.673222	12.619800	...	7.059670	27.729900	54.140100

Fonte: Adaptado de (HUERTA; HUERTA, 2016).

o valor máximo, a média e o desvio padrão dos valores da condutância para os atributos R1 ao R8 da tabela “Dados_sensor”, conforme a (TABELA 6).

Tabela 6: Dados estatísticos dos valores das condutâncias dos oito resistores utilizados na medição, tal que μ representa o valor da média populacional, σ é o desvio padrão, R_{max} e R_{min} são o valores da condutância máxima e condutância mínima medida por um dos sensores, respectivamente.

	R1	R2	R3	R4	R5	R6	R7	R8
μ	12.185	8.958	8.945	10.13	15.154	16.052	5.391	5.913
σ	0.868	1.558	1.749	1.712	18.392	3.303	2.889	3.304
R_{min}	5.431	1.821	1.627	2.283	1.901	5.588	1.22	1.431
R_{max}	13.733	11.316	11.374	12.755	378.75	73.818	102.575	99.888

Fonte: Autoria própria.

Nos histogramas apresentados, da (FIGURA 24) até a (FIGURA 31), mostra-se a distribuição dos dados da tabela “Dados_sensor” dividido em 50 classes para cada atributo (de R1 a R8). A informação dos histogramas assim como os dados da tabela (TABELA 6) são importantes para a mineração de dados devido a função de ativação dos neurônios da rede neural ser do tipo sigmoide. A função sigmoide tem dois pontos de saturação e para que a rede

neural possa aprender sem convergir rapidamente, é necessário normalizar a massa de dados. A normalização pode ser efetuada utilizando o desvio padrão (σ) e a média amostral (\bar{x}), como se mostra na (EQUAÇÃO 4):

$$N_d(r) = \frac{r - \bar{x}}{\sigma} \quad (4)$$

ou também pode ser efetuada utilizando os valores da condutância máximo (R_{\max}) e mínimo (R_{\min}) medidos para cada sensor, como se mostra na (EQUAÇÃO 5):

$$N_m(r) = \frac{r - R_{\min}}{R_{\max} - R_{\min}} \quad (5)$$

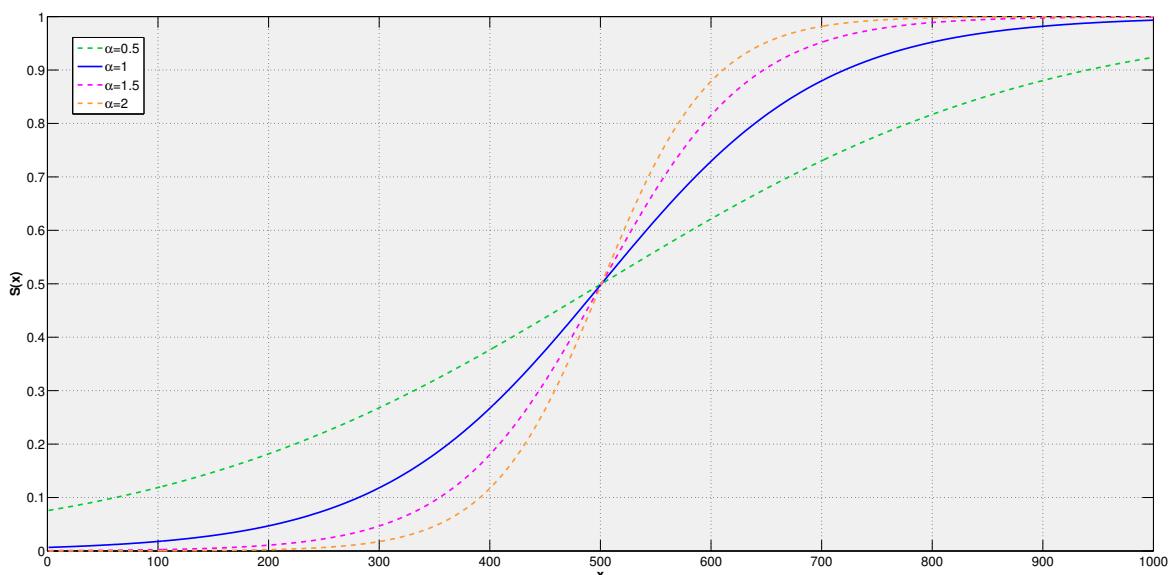
de modo que r é o valor da condutância medida para cada sensor.

A função sigmoide permite uma transição gradual e não linear entre dois estados, como se mostra na (EQUAÇÃO 6)

$$S(x) = \frac{1}{1 + e^{-\alpha x}} \quad (6)$$

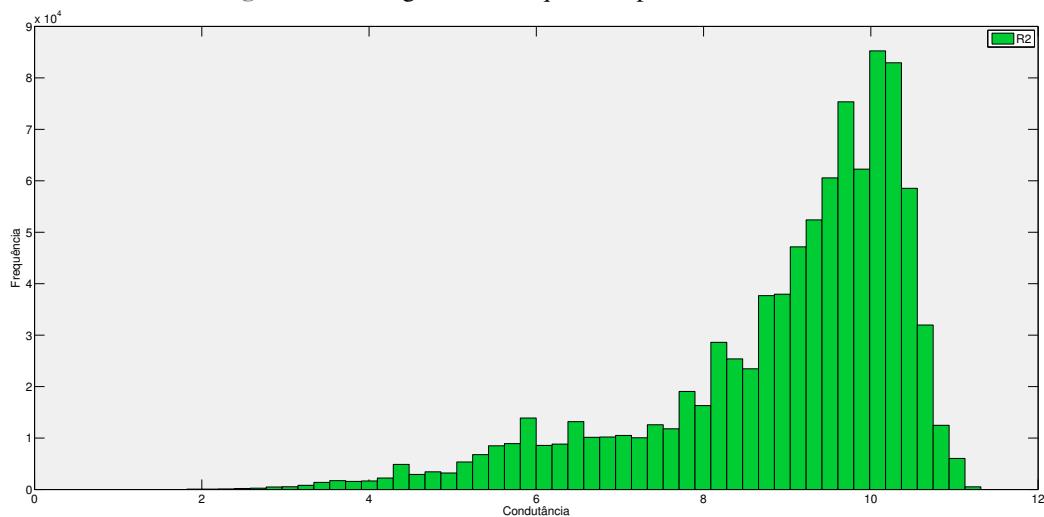
tal que α é um número real positivo. Quanto maior o valor de α , mais detalhada será a transição de um estado a outro, (CARDON; MÜLLER, 1994). Para a atividade de mineração de dados neste trabalho, utiliza-se uma rede neural com função de transferência do tipo sigmoide. Os estados de saturação são entre 0 e 1, é dizer, a função possui assintotas horizontais em 0 e 1, como se pode visualizar na (FIGURA 23).

Figura 23: Função sigmoide da equação (EQUAÇÃO 6), para diferentes valores de α . Essa função será utilizada como função de ativação nos neurônios ocultos da rede neural artificial.



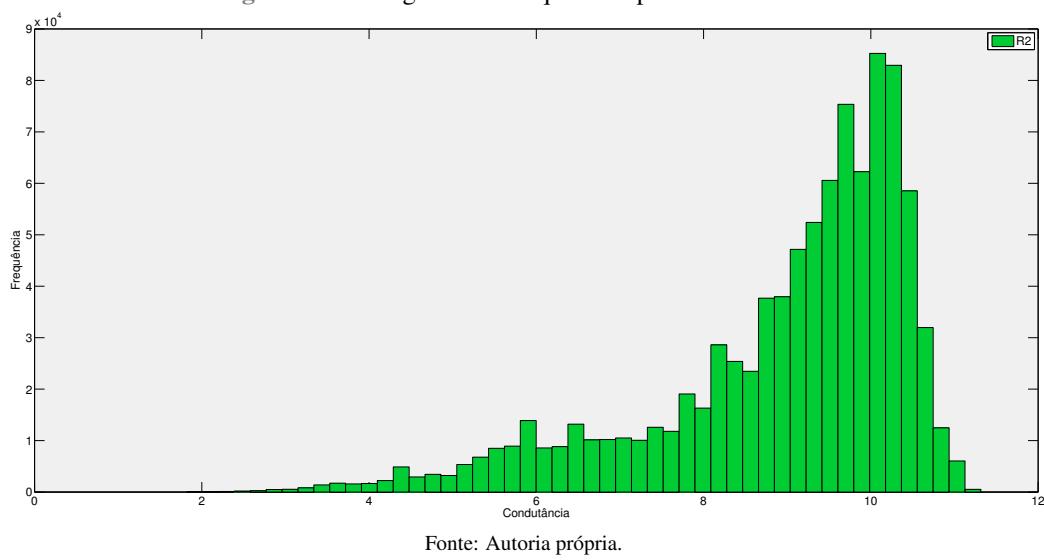
Fonte: Autoria própria.

Figura 24: Histograma de frequências para o sensor R1.



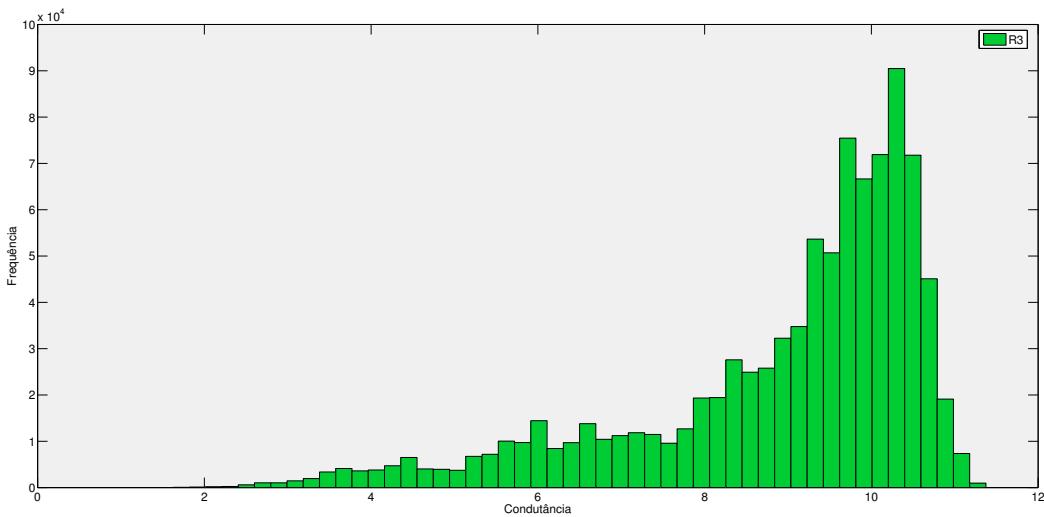
Fonte: Autoria própria.

Figura 25: Histograma de frequências para o sensor R2.



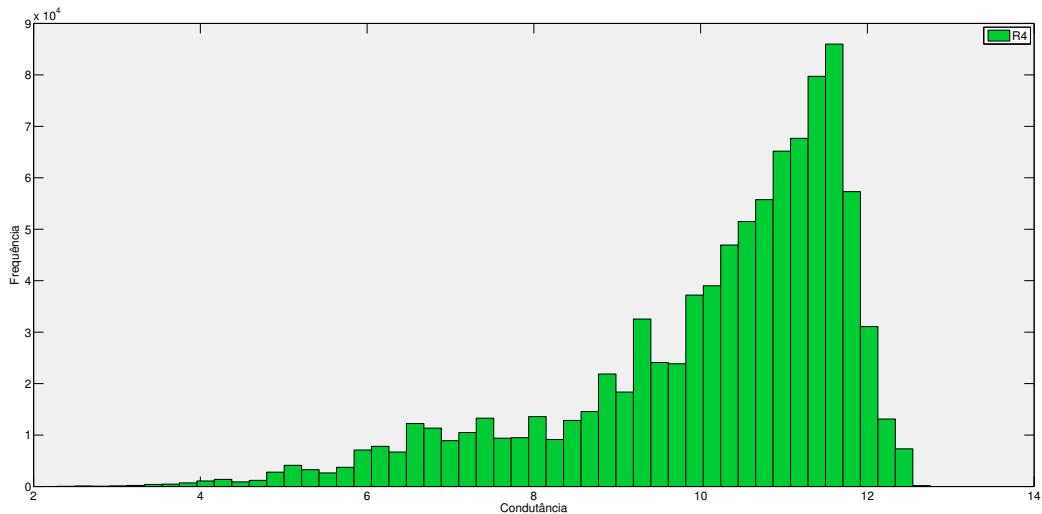
Fonte: Autoria própria.

Figura 26: Histograma de frequências para o sensor R3.



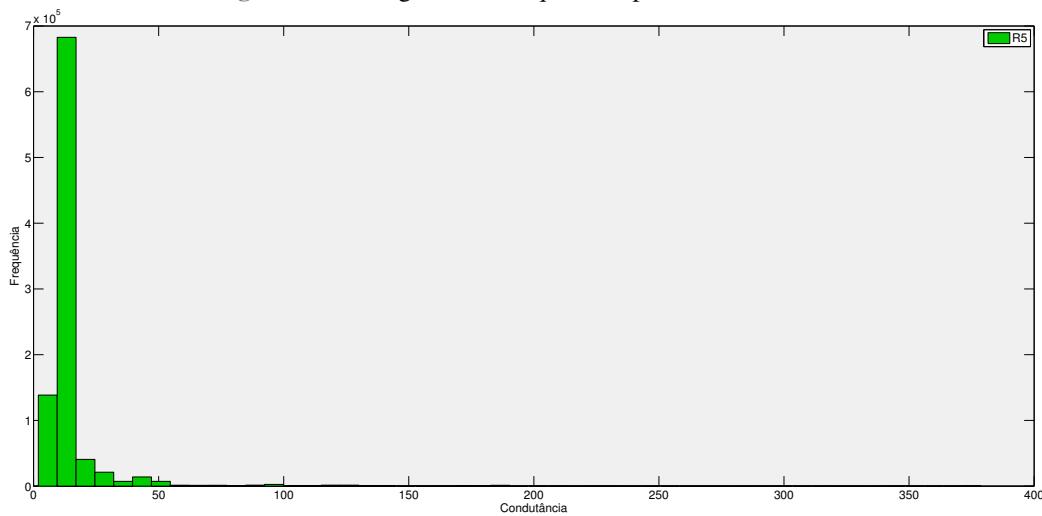
Fonte: Autoria própria.

Figura 27: Histograma de frequências para o sensor R4.



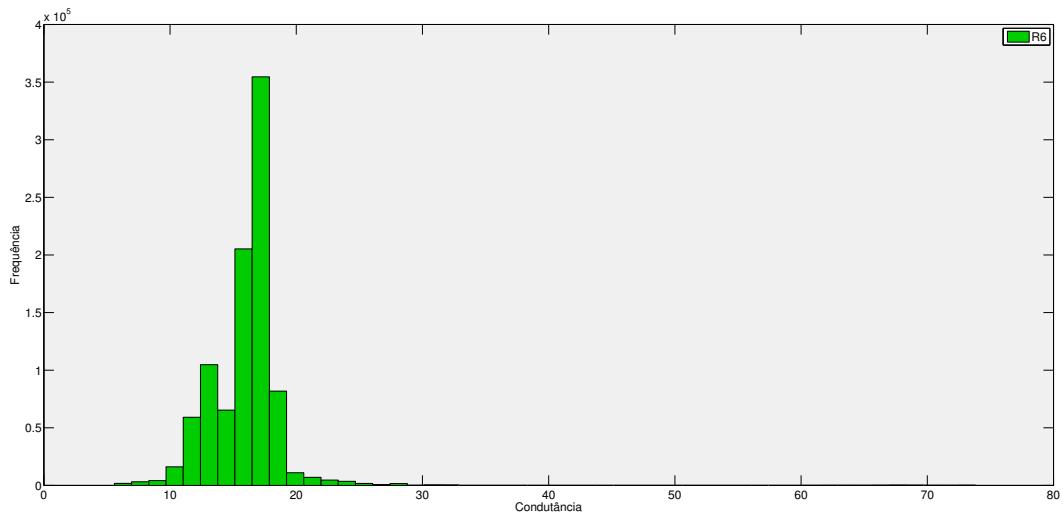
Fonte: Autoria própria.

Figura 28: Histograma de frequências para o sensor R5.



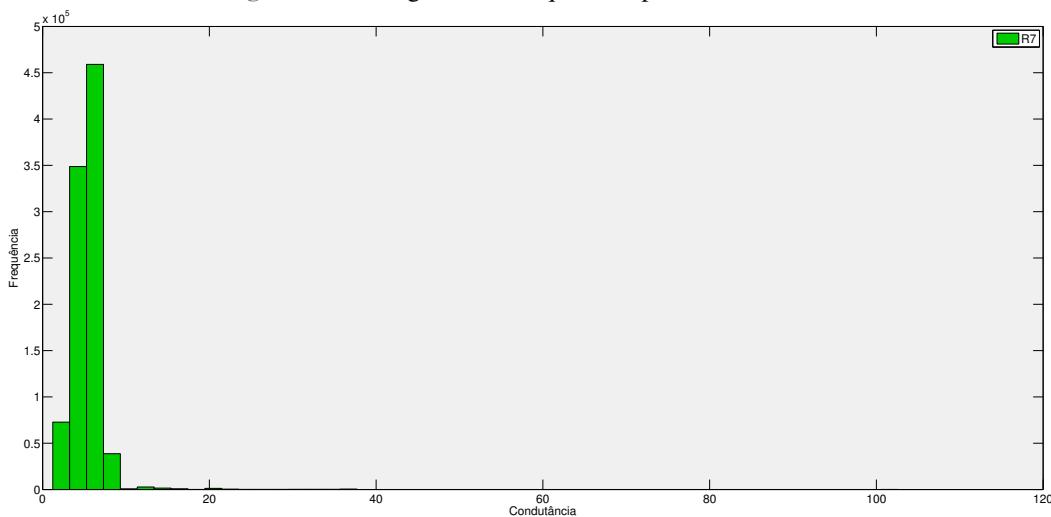
Fonte: Autoria própria.

Figura 29: Histograma de frequências para o sensor R6.



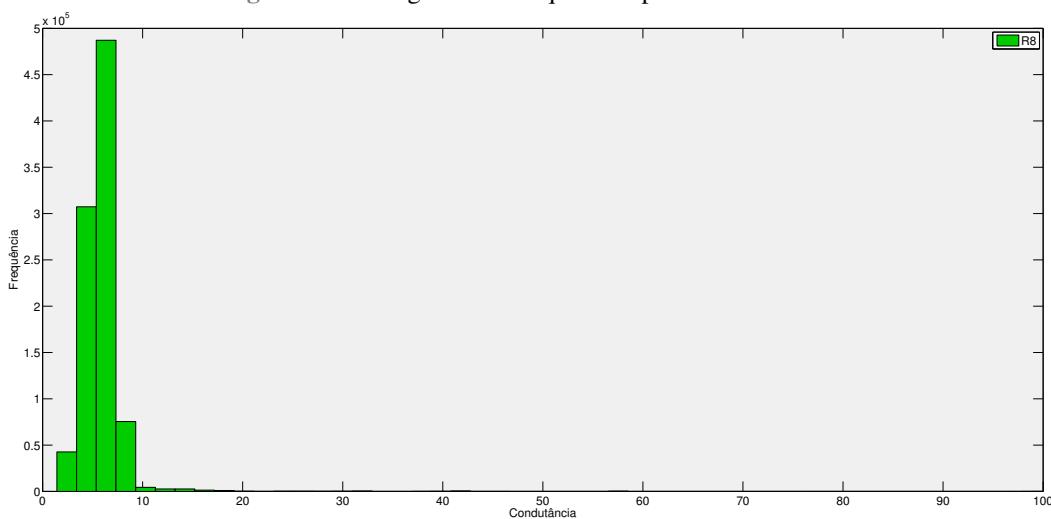
Fonte: Autoria própria.

Figura 30: Histograma de frequências para o sensor R7.



Fonte: Autoria própria.

Figura 31: Histograma de frequências para o sensor R8.



Fonte: Autoria própria.

5.2.4 Verificação da qualidade dos dados

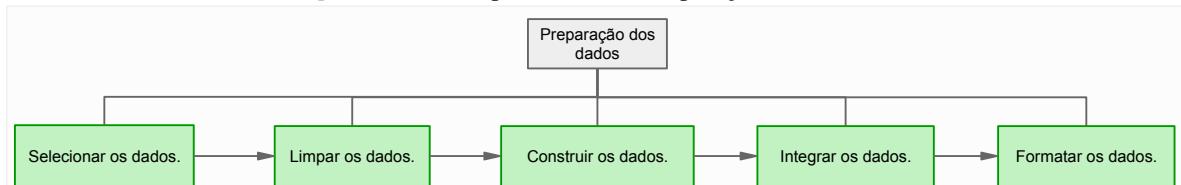
Em (HUERTA *et al.*, 2016), fonte da massa de dados, os autores propõem um modelo matemático novo de descorrelação dos sinais dos sensores químicos com os efeitos ambientais da umidade e a temperatura. Para poder chegar ao modelo que eles propõem, utilizam técnicas de aprendizado de máquina (ML) para poder discriminar as amostras. Devido ao mencionado anteriormente, considera-se que a massa de dados resultante possui uma qualidade aceitável.

5.3 Preparação dos dados

O objetivo do pré-processamento de dados é limpar os dados selecionados para uma melhor qualidade. Alguns dados selecionados podem ter formatos diferentes porque são escollidos de fontes de dados diferentes. Por exemplo, se os dados selecionados forem de arquivos planos, mensagem de voz e texto da Web, eles devem ser convertidos em um formato eletrônico consistente. Em geral, a limpeza de dados significa filtrar, agregar e preencher valores ausentes, (OLSON; DELEN, 2008).

Segundo (CHAPMAN *et al.*, 2000), nesta fase preparam-se os dados para cobrir todas as atividades necessárias para construir o conjunto final de dados a partir dos dados brutos iniciais. As tarefas incluem seleção de tabela, registro e atributo, bem como transformação e limpeza de dados para serem utilizados pelas ferramentas de modelagem, veja a (FIGURA 32).

Figura 32: Os subprocessos da “Preparação dos dados”.



Fonte: Adaptado de (CHAPMAN *et al.*, 2000).

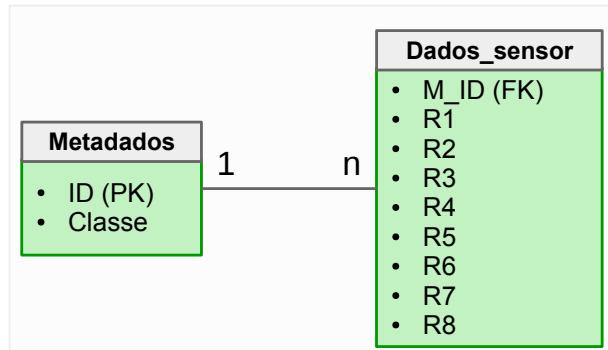
5.3.1 Selecionar os dados

Antes de selecionar os dados da tabela “Dados_sensor”, eliminam-se os atributos que não vão ser utilizados. Na tabela “Metadados”, os atributos “Data”, “t0” e “dt” não serão necessários porque o cliente somente deseja ver a relação entre os valores da condutância de cada um dos oito sensores (atributos “R1” ate “R8”), com as saídas respectivas (atributo “Classe”). Como resultado, tem-se o diagrama relacional da (FIGURA 33).

5.3.2 Limpar os dados

A partir da massa de dados fornecida pelo usuário, é selecionada uma amostra de 90 000 tuplas da tabela “Dados_sensor”, que representam 9,79 % do total dos dados. A amostra selecionada contém 30 000 tuplas de cada um dos três tipos: banana, vinho e *background*. Seleciona-se o mesmo número de tuplas para cada classe para que a rede neural artificial não seja tendenciosa.

Figura 33: Diagrama relacional com os atributos necessários para poder realizar a mineração de dados.



Fonte: Autoria própria.

5.3.3 Construir dados

Essa tarefa inclui operações construtivas de preparação de dados, como a produção de atributos derivados ou novos registros inteiros, ou valores transformados para atributos existentes. Neste trabalho não se utiliza nenhuma das operações mencionadas.

5.3.4 Integrar dados

Se pode observar na (TABELA 4), que para um elemento do atributo “classe” existem vários “ID” correspondentes. Por exemplo, para a classe “banana” existem os “ID” correspondentes: 0, 3, 5, 7, … . Como foram eliminados os atributos que não são utilizados, veja a (FIGURA 33), não faz sentido manter a tabela “Metadados” separada da tabela “Dados_sensor”. Podem-se juntar ambas tabelas e dessa forma facilitar o processo de extração de conhecimento da massa de dados. O processo de junção acontece do modo seguinte:

1. Agrupar pelo atributo “Classe” os dados da tabela “Metadados”. Não se descarta totalmente a informação do atributo “ID”, para cada uma das tuplas.
2. Procuram-se os ID correspondentes a cada classe na tabela “Dados_sensor” e substituem-se pelo nome da classe respectiva, em consequência, atributo “M_ID” da tabela “Dados_sensor” é modificada.
3. A tabela “Dados_sensor_modificado” é criada a partir da modificação da tabela “Dados_sensor”. Esta nova tabela tem como atributos os valores da condutância dos sensores R1 até R8 e classe (tipo de odor), como se pode observar na (FIGURA 34).

Figura 34: Tabela integrada, resultado da junção das tabelas “Metadados” e “Dados_sensor”.

Dados_sensor_modificado
• Classe
• R1
• R2
• R3
• R4
• R5
• R6
• R7
• R8

Fonte: Autoria própria.

5.3.5 Formatar os dados

A formatação dos dados serão feitas de dois tipos: uma delas será utilizada pelo algoritmo J48 e a outra, pelo algoritmo RX. Os dois tipos de formatação estão normalizadas utilizando a equação (EQUAÇÃO 6).

Formatação para o algoritmo J48 As tuplas do atributo “Classe” serão substituídas pelos nomes “banana”, “vinho” e “background”, ou seja serão dados do tipo *String*. Um exemplo desse tipo de formatação se pode ver na (TABELA 7).

Formatação para o algoritmo RX As tuplas do atributo “Classe” serão substituídas pelos vetores [1 0 0] para banana, [0 0 1] para vinho e [0 1 0] para *background*, ou seja os dados serão do tipo *Array*. Um exemplo desse tipo de formatação se pode ver na (TABELA 8).

5.4 Modelagem

Nesta fase, várias técnicas de modelagem são selecionadas e aplicadas e seus parâmetros são calibrados para valores ótimos. Normalmente, existem várias técnicas para o mesmo tipo de problema de mineração de dados. Algumas técnicas têm requisitos específicos sobre a forma dos dados. Portanto, voltar à fase de preparação de dados é muitas vezes necessário. Na (FIGURA 35) mostram-se os subprocessos da modelagem.

Com a intenção de ver qual das técnicas de modelagem tem uma maior qualidade de regras, aplicam-se duas técnicas diferentes: o algoritmo J48 e o algoritmo RX. Ambos algoritmos são aplicados à massa de dados devidamente preparada (tuplas que pertencem à tabela “Dados_sensor_modificado”). A massa de dados fornecida pelo cliente foi preparada seguindo

os subprocessos abordados na (SUBSEÇÃO 5.3).

Tabela 7: Nove tuplas aleatórias da tabela “Dados_sensor_modificado”, mostrada na (FIGURA 34), para ser utilizada pelo algoritmo J48.

Classe	R1(Ω^{-1})	R2 (Ω^{-1})	...	R6 (Ω^{-1})	R7 (Ω^{-1})	R8 (Ω^{-1})
“banana”	0,896340	0,864204	...	0,106250	0,042064	0,045797
“banana”	0,878452	0,912125	...	0,148516	0,024761	0,024682
“banana”	0,649617	0,415504	...	0,199481	0,030146	0,045352
“vinho”	0,593568	0,314689	...	0,108995	0,008386	0,008141
“vinho”	0,757291	0,565973	...	0,152908	0,004955	0,010631
“vinho”	0,855587	0,867998	...	0,172288	0,038788	0,041255
“background”	0,786858	0,742969	...	0,167254	0,038589	0,039787
“background”	0,758460	0,795414	...	0,138567	0,040692	0,050395
“background”	0,838362	0,890991	...	0,165994	0,053878	0,052771

Fonte: Autoria própria.

Tabela 8: Nove tuplas aleatórias da tabela “Dados_sensor_modificado”, mostrada na (FIGURA 34), para ser utilizada pelo algoritmo RX.

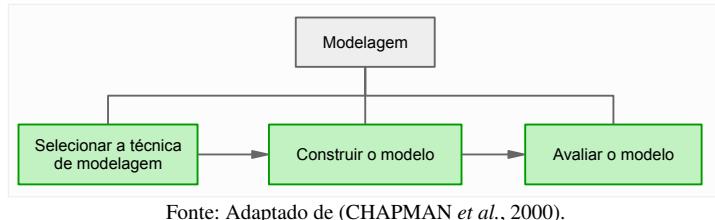
Classe	R1(Ω^{-1})	R2 (Ω^{-1})	...	R6 (Ω^{-1})	R7 (Ω^{-1})	R8 (Ω^{-1})
[1 00]	0,896340	0,864204	...	0,106250	0,042064	0,045797
[1 00]	0,878452	0,912125	...	0,148516	0,024761	0,024682
[1 00]	0,649617	0,415504	...	0,199481	0,030146	0,045352
[0 01]	0,593568	0,314689	...	0,108995	0,008386	0,008141
[0 01]	0,757291	0,565973	...	0,152908	0,004955	0,010631
[0 01]	0,855587	0,867998	...	0,172288	0,038788	0,041255
[0 1 0]	0,786858	0,742969	...	0,167254	0,038589	0,039787
[0 1 0]	0,758460	0,795414	...	0,138567	0,040692	0,050395
[0 1 0]	0,838362	0,890991	...	0,165994	0,053878	0,052771

Fonte: Autoria própria.

5.4.1 Selecionar a técnica de modelagem

Como primeiro passo na modelagem, seleciona-se a técnica de modelagem que será usada. Neste trabalho utilizam-se duas técnicas de modelagem: o algoritmo J48 e o algoritmo RX, para depois comparar a qualidade das regras lógicas obtidas de cada uma dessas técnicas.

Figura 35: Os subprocessos da “Modelagem”.



Fonte: Adaptado de (CHAPMAN *et al.*, 2000).

O algoritmo J48 é baseado em uma árvore de decisão, ver (SUBSEÇÃO 4.4.2). O algoritmo RX é um algoritmo de extração de regras que utiliza os valores dos pesos e dos bias dos neurônios da camada oculta de uma rede neural artificial, ver (SUBSEÇÃO 4.5).

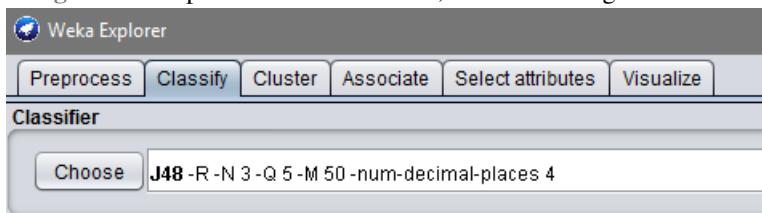
5.4.2 Construir o modelo

Modelagem do algoritmo J48 O algoritmo J48 será aplicado diretamente à massa de dados da (TABELA 7), sem utilizar uma rede neural artificial, utilizando o comando:

```
weka.classifiers.trees.J48 -R -N 3 -Q 5 -M 50 -num-decimal-places 4
```

no software WEKA, como se pode verificar na (FIGURA 36). O argumento “-R” significa usar uma redução de erro na poda da árvore, “-N 3” define o número de dobrões para reduzir a poda de erros, “-Q 5” semente para aleatorização dos dados, “-M 50” define o número mínimo de instâncias por folha e “-num-decimal-places 4” define o número de casas decimais.

Figura 36: Captura de tela do WEKA, utilizando o algoritmo J48.



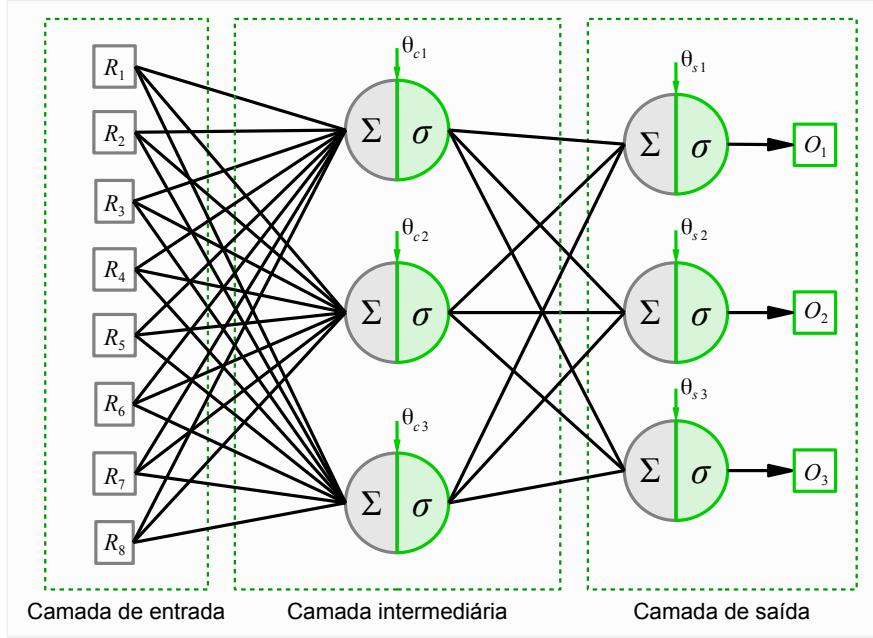
Fonte: Autoria própria.

Modelagem do algoritmo RX O algoritmo RX foi utilizado tendo em conta o artigo (HRUSCHKA; EBECKEN, 1999). Primeiramente projetou-se uma rede neural artificial de oito entradas (uma para cada valor de condutância), uma camada escondida de três neurônios e uma camada de saída de três neurônios. Na (TABELA 8) mostra-se um exemplo da amostra de dados coletada para o treinamento da rede neural artificial. Para treinar a rede neural utilizou-se o algoritmo LM, na caixa de ferramentas de Redes Neurais do software Matlab.

Na (FIGURA 37) mostra-se a rede neural projetada, tal que a função de ativação (σ) é a função sigmoide da (EQUAÇÃO 6), com $\alpha = 1$; os valores de θ_{ci} e θ_{si} são os valores de bias

para a camada oculta e de saída respectivamente e os valores O_i (com $i \in \{1, 2, 3\}$) são as saídas no formato de Array, como se pode verificar na coluna Classe da (TABELA 8).

Figura 37: Arquitetura da rede neural artificial projetada para a mineração da massa de dados fornecida pelo cliente. É a partir desta rede neural artificial que se utiliza o algoritmo RX.



Fonte: Autoria própria.

Depois de projetada e treinada a rede neural artificial, obtém-se os valores de pesos e bias para poder introduzi-los nas entradas do algoritmo RX. O algoritmo RX foi implementado no software Matlab pelos autores do artigo (HRUSCHKA; EBECKEN, 1999). Para poder utilizar o algoritmo seguem-se os passos:

1. Inserir o fator multiplicador do desvio padrão da massa de dados e o número de classes.
Os autores do artigo (HRUSCHKA; EBECKEN, 1999) recomendam colocar o valor do fator multiplicador do desvio igual a 2. Fixa-se o número de classes igual a 3.
2. Em seguida, se insere o número de tuplas relacionadas com as três classes relacionadas: “banana”, “vinho” e “background”. Cada classe possui 30 000 tuplas.
3. Por fim, escolhe-se a porcentagem mínima de elementos que conterão o *cluster*, como se indica em (HRUSCHKA; EBECKEN, 1999). O algoritmo de agrupamento (*clustering*) que se utiliza em (HRUSCHKA; EBECKEN, 1999) é o *Leader Algorithm*. Neste algoritmo se constrói uma partição a partir de *clusters*, com um diâmetro especificado no espaço e no tempo, dependendo da ordem de apresentação dos elementos que vão ser agrupados, (HARTIGAN, 1973). No (ALGORITMO 3), explica-se de forma resumida

o *Leader Algorithm*. Em (HRUSCHKA; EBECKEN, 1999), recomenda-se utilizar 80 % dos elementos do *cluster*.

Algoritmo 3 *Leader Algorithm*, disponível em (MICHALEWICZ, 1996).

```

1: procedure LEADER ALGORITHM
2:    $I \leftarrow 1$                                  $\triangleright I$  indica a posição do objeto
3:    $L(1) \leftarrow 1$                              $\triangleright L(I)$  é o objeto líder
4:   while ( $I < N$ ) do                       $\triangleright N$  indica o número de elementos do conjunto
5:      $I \leftarrow I + 1$                            $\triangleright d(I, J)$  indica a distância entre  $I$  e o possível líder  $J$ 
6:     Procure o objeto  $J$  entre  $L(1), \dots, L(I - 1)$ , para que cada  $d(I, J)$  seja mínima.
7:     if ( $d(I, J) \leq D$ ) then                   $\triangleright D$  é o limiar de distância
8:        $L(I) \leftarrow J$ 
9:     end if
10:    if ( $d(I, J) > D$ ) then
11:       $L(I) \leftarrow I$ 
12:    end if
13:  end while
14: end procedure
  
```

5.4.3 Avaliar o modelo

Neste item, resumem-se os resultados, listam-se as qualidades dos modelos gerados (por exemplo, em termos de precisão) e classificam-se a sua qualidade em relação uns aos outros.

Utilizando o algoritmo J48 do WEKA, disponível em (HALL *et al.*, 2009), a partir da massa de dados fornecida pelo cliente, obtém-se o seguinte resultado:

Número de folhas da árvore	: 217
Tamanho da árvore	: 432
Instâncias corretamente classificadas	: 84 057 (93,40 %)
Instâncias incorretamente classificadas	: 5 943 (6,60 %)

Utilizando o algoritmo RX proposto em (HRUSCHKA; EBECKEN, 1999), a partir da rede neural artificial treinada com a massa de dados normalizada, obtém-se o seguinte resultado:

Instâncias corretamente classificadas	: 80 802 (89,78 %)
Instâncias incorretamente classificadas	: 9 198 (10,22 %)

A correta avaliação da classificação das instâncias (tuplas) são feitas por procedimentos internos a cada algoritmo (J48 e RX). As regras lógicas extraídas e a árvore de decisão são mostradas na (SEÇÃO 6).

5.5 Avaliação

Segundo (CHAPMAN *et al.*, 2000), antes de proceder à implantação final do modelo, é importante avaliá-lo cuidadosamente e rever as etapas executadas para criá-lo, para ter certeza de que o modelo atinge adequadamente os objetivos. Um dos principais objetivos é determinar se existe alguma questão comercial importante que não tenha sido suficientemente considerada. No final desta fase, uma decisão sobre o uso dos resultados de mineração de dados deve ser alcançada.

Segundo (OLSON; DELEN, 2008), a fase de interpretação dos dados é muito crítica. Duas questões são essenciais. Uma delas é como reconhecer o valor comercial dos padrões de conhecimento descobertos no estágio de mineração de dados. Outro problema é qual ferramenta de visualização deve ser usada para mostrar os resultados de mineração de dados.

No presente trabalho não é desenvolvida uma ferramenta para visualizar o resultado da mineração de dados, porém, na (SEÇÃO 6), mostram-se a árvore de decisão resultante e as regras lógicas extraídas.

5.6 Desenvolvimento

Em (CHAPMAN *et al.*, 2000), afirma-se que a criação do modelo geralmente não é o fim do projeto. Mesmo se o propósito do modelo é aumentar o conhecimento a partir da geração de novos dados, o conhecimento adquirido terá que ser organizado e apresentado de forma que o cliente possa usá-lo.

Segundo (OLSON; DELEN, 2008), os modelos projetados precisam ser monitorados para detectar as mudanças na fase de produção. Se ocorrerem mudanças significativas, o modelo deve ser refeito. Também é aconselhável registrar os resultados dos projetos de mineração de dados, de forma que evidências documentadas estejam disponíveis para futuros estudos.

O modelo proposto não foi implementado no presente trabalho devido a que os dados coletados são estáticos, ou seja, não serão gerados novos dados em tempo real porque o experimento, realizado em (HUERTA *et al.*, 2016), já foi concluído.

6 Resultados

Os resultados são mostrados como resultado do processo de modelagem da metodologia CRISP-DM, ver (SUBSEÇÃO 5.4). Uma árvore de decisão e um conjunto de regras lógicas são construídos a partir da massa de dados fornecida pelo cliente, como se explica na (SEÇÃO 5).

6.1 Utilizando o algoritmo J48

Aplicando o algoritmo J48 na massa de dados, utilizando o software WEKA versão 3.8.0, obteve-se que 93,40 % de tuplas foram classificadas corretamente. Criou-se uma árvore de decisão de tamanho 432 e 217 folhas, ver ANEXO A.1. Relembrando que os valores que se mostram na árvore de decisão estão normalizados.

6.2 Utilizando o algoritmo RX

Foi aplicado o algoritmo RX na rede neural artificial projetada, ver (FIGURA 37). Conforme os procedimentos realizados em (HRUSCHKA; EBECKEN, 1999), obtiveram-se as seguintes equações de ativação das unidades ocultas (neurônios da camada escondida) da rede neural artificial:

$$\begin{aligned} N_1 &= 3.212 - 1.101R_1 + 0.154R_2 - 0.933R_3 + 1.886R_4 + \dots \\ &\quad \dots + 1.395R_5 + 1.610R_6 - 0.315R_7 - 0.135R_8 \\ N_2 &= 0.000 - 0.132R_1 - 1.859R_2 - 1.156R_3 + 1.183R_4 + \dots \\ &\quad \dots + 0.796R_5 - 0.489R_6 - 1.723R_7 + 0.521R_8 \\ N_3 &= -3.212 - 0.793R_1 + 1.120x2 + 0.802R_3 - 1.665R_4 + \dots \\ &\quad \dots + 0.883R_5 - 1.646R_6 - 0.771R_7 + 0.969R_8 \end{aligned}$$

de tal forma que R_i , para $i = \{1, 2, 3, 4, 5, 6, 7, 8\}$ são os valores da condutância dos sensores normalizados. Estes valores são inseridos na camada de entrada da rede neural artificial. Os valores de N_j , para $j = \{1, 2, 3\}$ são os valores relacionados aos três neurônios da camada escondida da rede neural artificial, ver (FIGURA 37).

Uma vez que se têm as equações de ativação, geram-se as regras lógicas para cada classe especificada no treinamento da rede neural artificial projetada. As regras lógicas para a

classe “banana” são

$$\begin{aligned} \mathbf{SE} \quad & 3.07 \leq N_1 \leq 3.74 \quad \mathbf{E} \\ & -2.22 \leq N_2 \leq -0.91 \quad \mathbf{E} \\ & -4.03 \leq N_3 \leq -3.65 \quad \mathbf{ENTÃO É banana} \end{aligned}$$

para a classe “vinho” são

$$\begin{aligned} \mathbf{SE} \quad & 2.98 \leq N_1 \leq 3.83 \quad \mathbf{E} \\ & -2.27 \leq N_2 \leq -0.63 \quad \mathbf{E} \\ & -4.18 \leq N_3 \leq -3.57 \quad \mathbf{ENTÃO É vinho} \end{aligned}$$

e para a classe “background” são

$$\begin{aligned} \mathbf{SE} \quad & 3.33 \leq N_1 \leq 3.60 \quad \mathbf{E} \\ & -1.98 \leq N_2 \leq -1.49 \quad \mathbf{E} \\ & -3.99 \leq N_3 \leq -3.80 \quad \mathbf{ENTÃO É background} \end{aligned}$$

O algoritmo RX proposto em (HRUSCHKA; EBECKEN, 1999) classificou corretamente 89,78 % das tuplas da massa de dados.

7 Conclusões e recomendações

7.1 Conclusões

A metodologia CRISP-DM, abordada neste trabalho, é fundamental na hora de fazer a mineração de dados porque fornece as ferramentas necessárias para poder tratar uma grande massa de dados e extrair conhecimento dela.

A compreensão do projeto é o primeiro passo da metodologia CRISP-DM e ajuda a construir uma base sólida para realizar todo o processo de mineração de dados; a realização das tarefas como planejamento, determinação de objetivos e avaliação de contextos é necessária para o projeto de extração de conhecimento a partir de uma massa de dados.

Compreender os dados fornecidos pelo cliente é o passo seguinte da metodologia CRISP-DM. Conhecer os metadados e as variáveis estatísticas envolvidas é essencial para realizar a modelagem e o treinamento da rede neural artificial. Selecionar, limpar, construir e integrar os dados são os subprocessos da preparação dos dados e são essenciais para que a rede neural projetada convirja no menor tempo possível.

A modelagem da metodologia adotada é a parte mais importante do processo CRISP-DM. Utilizar o algoritmo RX, aplicado a uma rede neural artificial, para o processo de classificação de dados é uma das formas mais diretas de extrair conhecimento.

Os processos de avaliação e de desenvolvimento da metodologia CRISP-DM são dinâmicos, isto é, os métodos abordados (extração de regras e árvore de decisão) têm que ser avaliados com dados que sejam gerados em tempo real e desenvolvidos com métodos que se adaptem aos novos dados gerados. Isto quer dizer que os métodos descritos neste trabalho podem não ser eficientes para novos dados. A eficiência dos modelos se mede justamente no processo de avaliação, onde se descartam alguns métodos e se propõem outros novos.

Fazendo uma comparação dos resultados do método de extração de regras (algoritmo RX) e o método de árvore de decisão (algoritmo J48), pode-se concluir que o algoritmo RX (89,78 % de classificações corretas) tem menos acurácia que o algoritmo J48 (93,40 % de classificações corretas). Conclui-se também que o algoritmo RX possui visivelmente menor número de regras lógicas com relação ao algoritmo J48.

7.2 Recomendações

1. Realizar o processo de CRISP-DM utilizando diferentes algoritmos de extração de regras em diferentes arquiteturas de RNA. Utilizar diferentes tipos de algoritmos para classificação utilizando árvores de decisão.
2. Utilizar uma massa de dados que seja sempre atualizada em tempo real. É interessante porque é onde se dá uma aplicação real ao projeto de mineração de dados utilizando a metodologia CRISP-DM.
3. Desenvolver uma aplicação que mostre o resultado da mineração e implemente as regras lógicas para que o usuário final possa utilizá-las. Essas regras podem ser implementadas usando uma linguagem de programação de paradigma lógico como SWI Prolog, disponível em <<http://www.swi-prolog.org/>>.

Referências

- AKERKAR, R.; SAJJA, P. *Knowledge-Based Systems*. [S.l.]: Jones & Bartlett, 2010.
- ALPAYDM, E. *Introduction to Machine Learning*. [S.l.]: The MIT Press, 2004.
- AMORA, M. A. . B. *et al.* Extração de conhecimento de redes neurais artificiais e aplicação na análise de transformadores. *Sociedade Brasileira de Automática*, 2007.
- AMORA, M. A. B. *Extração de regras interpretáveis para o diagnóstico eficiente de transformadores de potência isolados a óleo a partir do aumento do espaço de atributos*. Tese (Doutorado) — Universidade Federal do Ceará, 2013.
- BARBER, D. *Bayesian Reasoning and Machine Learning*. [S.l.: s.n.], 2010.
- BEI, A. *et al.* Health-mining: a disease management support service based on data mining and rule extraction. *Engineering in Medicine and Biology 27th Annual Conference*, 2005.
- BENÍTEZ, J. M.; CASTRO, J. L.; REQUENA, I. Are artificial neural networks black boxes? *IEEE Transactions On Neural Networks*, 1997.
- BINUS. *Processes in Data Mining*. 2014. Disponível em: <<http://sisbinus.blogspot.com.br/2014/11/processes-in-data-mining.html>>. Acesso em: 10 jun. 2016.
- BOLOGNA, G. A model for single and multiple knowledge based networks. *Artificial Intelligence in Medicine*, 2003.
- CAMILO, C. O.; SILVA, J. C. da. *Mineração de Dados: Conceitos, Tarefas, Métodos e Ferramentas*. [S.l.], 2009.
- CAMPBELL, D.; CAMPBELL, S. *Introduction to Regression and Data Analysis*. [S.l.: s.n.], 2008.
- CARDON, A.; MÜLLER, D. N. Introdução às redes neurais artificiais. 1994.
- CHAPMAN, P. *et al.* *CRISP-DM 1.0: Step-by-step data mining guide*. [S.l.], 2000.
- CHATURVEDI, D. K. *Soft Computing: Techniques and its Applications in Electrical Engineering*. [S.l.]: Springer, 2008.
- CHOI, J.-K. *et al.* Study on datamining techinique for foot disease prediction. *IT Convergence and Security (ICITCS), 2014 International Conference*, 2014.
- COPIN, B. *Inteligência Artificial*. [S.l.]: gen LTC, 2004.
- DAVIDSON, R.; MACKINNON, J. G. *Econometric Theory and Methods*. [S.l.: s.n.], 2003.
- FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From data mining to knowledge discovery in databases. *AI MAGAZINE*, 1996.
- FISHER, R.; MARSHALL, M. *Iris Plants Database*. 1988. Disponível em: <<http://www.cs.mcmaster.ca/~cs4tf3/iris.arff>>. Acesso em: 10 nov. 2016.
- GAVIN, H. P. The levenberg-marquardt method for nonlinear least squares curve-fitting problems. *Department of Civil and Environmental Engineering*, 2016.

- GHOLAP, J. Performance tuning of j48 algorithm for prediction of soil fertility. 2012.
- HALL, M. et al. *The WEKA Data Mining Software: An Update. SIGKDD Explorations, Volume 11, Issue 1*. 2009. Disponível em: <<http://www.cs.waikato.ac.nz/ml/index.html>>. Acesso em: 23 set. 2016.
- HAN, J.; PEI, M. K. J. *Data Mining Concepts and Techniques*. 3rd. ed. [S.l.]: ELSEVIER, 2012.
- HANS-JIIRGEN. *Fuzzy Set Theory-and Its Applications*. [S.l.]: Springer Science, 2001.
- HARTIGAN, J. A. Clustering. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, Vol. 28, No. 1, 1973.
- HAYKIN, S. *Redes Neurais, Princípios e prática*. [S.l.]: Prentice Hall, 2008.
- HRUSCHKA, E. R.; EBECKEN, N. F. F. Extração de regras de redes neurais por meio do algoritmo rx modificado: Um exemplo de aplicação em modelagem de dados meteorológicos. *Proceedings of the IV Brazilian Conference on Neural Networks*, 1999.
- HUERTA, F.; HUERTA, R. *Gas sensors for home activity monitoring Data Set*. 2016. Disponível em: <<http://archive.ics.uci.edu/ml/datasets/Gas+sensors+for+home+activity+monitoring>>. Acesso em: 10 oct. 2016.
- HUERTA, R. et al. Online decorrelation of humidity and temperature in chemical sensors for continuous monitoring. *Chemometrics and Intelligent Laboratory Systems*, 2016.
- LEVENBERG, K. A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 1944.
- LOURAKIS, M. I. A. A brief description of the levenber g-marquardt algorithm implemened. *Foundation for Research and Technology - Hellas*, 2005.
- LU, H.; SETIONO, R.; LIU, H. Effective data mining using neural networks. *IEEE Transactions On Knowledge And Data Engineering*, 1996.
- MAIMON, O.; ROKACH, L. *Soft Computing for Knowledge Discovery and Data Mining*. [S.l.]: Springer, 2007.
- MANKAD, K. B. *A Genetic-Fuzzy Approach to Measure Multiple Intelligence*. Dissertação (Mestrado) — Sardar Patel University, 2013.
- MARSLAND, S. *Machine Learning: An Algorithmic Perspective*. [S.l.]: CRC Press, 2015.
- MARTINS, A. C.; MARQUES, J. ao M.; COSTA, P. D. *Três Algoritmos de Machine Learning na Classificação de Dados Electrocardiográficos*. Dissertação (Mestrado) — Faculdade de Medicina da Universidade do Porto, 2009.
- MICHALEWICZ, Z. *Genetic Algorithms + Data Structures = Evolution Programs*. [S.l.]: Springer, 1996.
- MICHIE, D.; SPIEGELHALTER, D.; TAYLOR, C. *Machine Learning, Neural and Statistical Classification*. [S.l.: s.n.], 1994.

- MURPHY, K. P. *Machine Learning: A Probabilistic Perspective*. [S.l.]: Massachusetts Institute of Technology, 2012.
- OLSON, D. L.; DELEN, D. *Advanced Data Mining Techniques*. [S.l.]: Springer, 2008.
- OMLIN, C. W.; GILES, C. L. Extraction of rules from discrete-time recurrent neural networks. 1995.
- PIATETSKY, G. *CRISP-DM, still the top methodology for analytics, data mining or data science projects*. 2014. Disponível em: <<http://www.kdnuggets.com/2014/10/crisp-dm-top-methodology-analytics-data-mining-data-science-projects.html>>. Acesso em: 10 jun. 2016.
- PUC-RIO. *Árvore de Decisão*. 2005. Disponível em: <http://www.maxwell.vrac.puc-rio.br/7587/7587_4.PDF>.
- RAJARAMAN, A.; LESKOVEC, J.; ULLMAN, J. D. *Mining of Massive Datasets*. [S.l.]: Stanford University, 2012.
- RANGANATHAN, A. The levenberg-marquardt algorithm. 2004.
- SAS. *Decision Trees - What Are They?* 2006. Disponível em: <<http://support.sas.com/publishing/pubcat/chaps/57587.pdf>>. Acesso em: 05 set. 2016.
- SETIONO, R.; LIU, H. Neurolinear: From neural networks to oblique decision rules. *Neurocomputing*, 1997.
- SHAFIQUE, U.; QAISER, H. A comparative study of data mining process models (kdd, crisp-dm and semma). *International Journal of Innovation and Scientific Research*, 2014.
- SHALEV-SHWARTZ, S.; BEN-DAVID, S. *Understanding Machine Learning: From Theory to Algorithms*. [S.l.]: Cambridge University Press, 2014.
- SIVANANDAM, S. N.; DEEPA, S. N. *Principles of Soft Computing*. [S.l.]: Wiley, 2004.
- TAN, P.-N.; STEINBACH, M.; KUMAR, V. *Introduction to Data Mining*. [S.l.: s.n.], 2006.
- WITTEN, I. H.; FRANK, E.; HALL, M. A. *Data Mining Practical Machine Learning Tools and Techniques*. [S.l.]: El Sevier, 2011.
- YU, H.; WILAMOWSK, B. M. *Intelligent systems*. [S.l.]: CRC Press, 2011.
- ZADEH, L. A. The roles of soft computing and fuzzy logic in the conception, design and deployment of intelligent system. *IEEE Intelligent Systems*, 1996.
- ZAKI, M. J.; JR., W. M. *Data Mining and Analysis Fundamental Concepts and Algorithms*. [S.l.]: Cambridge University Press, 2014.
- ZAPATA, C. M.; GIL, N. Incorporation of both pre-conceptual schemas and goal diagrams in crisp-dm. *Computing Congress*, 2011.
- ZHONG, L.; GUO, C.; SONG, H. Research and improvement of splitting rule extraction data mining algorithm based on neural networks. *International Conference on Computer Science and Software Engineering*, 2008.

Anexo A.1: Árvore de decisão resultante da aplicação do algoritmo J48


```
244 | | | | | r5 <= 0.071977
245 | | | | | r2 <= 0.950938
246 | | | | | r8 <= 0.067533
247 | | | | | r3 <= 0.71179
248 | | | | | r2 <= 0.76057
249 | | | | | r8 <= 0.043617
250 | | | | | r8 <= 0.038877: banana (53.0/5.0)
251 | | | | | r8 > 0.038877: background (401.0/49.0)
252 | | | | | r8 > 0.043617
253 | | | | | r7 <= 0.035213: background (98.0/5.0)
254 | | | | | r7 > 0.035213
255 | | | | | r7 <= 0.055828
256 | | | | | r6 <= 0.15332: vinho (253.0)
257 | | | | | r6 > 0.15332
258 | | | | | r6 <= 0.162911: banana (58.0/9.0)
259 | | | | | r6 > 0.162911: vinho (153.0/25.0)
260 | | | | | r7 > 0.055828: background (52.0/8.0)
261 | | | | | r2 > 0.76057: banana (100.0)
262 | | | | | r3 > 0.71179
263 | | | | | r1 <= 0.817655
264 | | | | | r8 <= 0.039521: banana (42.0/8.0)
265 | | | | | r8 > 0.039521
266 | | | | | r1 <= 0.803202: background (3532.0/12.0)
267 | | | | | r1 > 0.803202
268 | | | | | r3 <= 0.782749
269 | | | | | r7 <= 0.036191: background (158.0/3.0)
270 | | | | | r7 > 0.036191
271 | | | | | r6 <= 0.153648: banana (113.0/15.0)
272 | | | | | r6 > 0.153648: background (52.0/15.0)
273 | | | | | r3 > 0.782749: background (1140.0/7.0)
274 | | | | | r1 > 0.817655
275 | | | | | r3 <= 0.782173
276 | | | | | r7 <= 0.040523: background (157.0/28.0)
277 | | | | | r7 > 0.040523
278 | | | | | r8 <= 0.047423: banana (154.0/19.0)
279 | | | | | r8 > 0.047423
280 | | | | | r6 <= 0.159133
281 | | | | | r7 <= 0.045482: vinho (136.0)
282 | | | | | r7 > 0.045482
283 | | | | | r1 <= 0.837784: vinho (52.0/15.0)
284 | | | | | r1 > 0.837784: banana (101.0/18.0)
285 | | | | | r6 > 0.159133
286 | | | | | r1 <= 0.866712
287 | | | | | r5 <= 0.028894: vinho (70.0/24.0)
288 | | | | | r5 > 0.028894: background (50.0/4.0)
289 | | | | | r1 > 0.866712: vinho (185.0)
290 | | | | | r3 > 0.782173
291 | | | | | r7 <= 0.041496
292 | | | | | r5 <= 0.023545
```


386 | | | | | | | | **r5** > 0.02791: banana (154.0)
387 | | | | | | | | **r5** > 0.071977
388 | | | | | | | | **r1** <= 0.809422
389 | | | | | | | | **r6** <= 0.147811: background (170.0)
390 | | | | | | | | **r6** > 0.147811
391 | | | | | | | | **r8** <= 0.040522: vinho (52.0)
392 | | | | | | | | **r8** > 0.040522: banana (54.0)
393 | | | | | | | | **r1** > 0.809422: banana (683.0/10.0)
394 | | | | | | | | **r6** > 0.181565
395 | | | | | | | | **r1** <= 0.935237
396 | | | | | | | | **r8** <= 0.048716
397 | | | | | | | | **r5** <= 0.029728
398 | | | | | | | | **r7** <= 0.039164
399 | | | | | | | | **r6** <= 0.232425
400 | | | | | | | | **r6** <= 0.18561
401 | | | | | | | | | | **r3** <= 0.863258: vinho (61.0)
402 | | | | | | | | | | **r3** > 0.863258: banana (50.0/12.0)
403 | | | | | | | | **r6** > 0.18561: banana (222.0/28.0)
404 | | | | | | | | **r6** > 0.232425: vinho (78.0)
405 | | | | | | | | **r7** > 0.039164: vinho (256.0/19.0)
406 | | | | | | | | **r5** > 0.029728: vinho (682.0/8.0)
407 | | | | | | | | **r8** > 0.048716
408 | | | | | | | | **r8** <= 0.056899
409 | | | | | | | | **r7** <= 0.056015
410 | | | | | | | | **r2** <= 0.835023
411 | | | | | | | | **r7** <= 0.045098
412 | | | | | | | | **r6** <= 0.187783: background (183.0/17.0)
413 | | | | | | | | **r6** > 0.187783: banana (125.0/2.0)
414 | | | | | | | | **r7** > 0.045098: vinho (145.0)
415 | | | | | | | | **r2** > 0.835023
416 | | | | | | | | **r6** <= 0.183348: banana (128.0/25.0)
417 | | | | | | | | **r6** > 0.183348: vinho (257.0/11.0)
418 | | | | | | | | **r7** > 0.056015: background (266.0/4.0)
419 | | | | | | | | **r8** > 0.056899
420 | | | | | | | | **r5** <= 0.026816
421 | | | | | | | | **r6** <= 0.250205
422 | | | | | | | | **r4** <= 0.857513: banana (151.0)
423 | | | | | | | | **r4** > 0.857513
424 | | | | | | | | **r6** <= 0.235971: vinho (53.0/9.0)
425 | | | | | | | | **r6** > 0.235971: banana (105.0/43.0)
426 | | | | | | | | **r6** > 0.250205: vinho (108.0/1.0)
427 | | | | | | | | **r5** > 0.026816
428 | | | | | | | | **r7** <= 0.321445: vinho (559.0/8.0)
429 | | | | | | | | **r7** > 0.321445: banana (53.0/11.0)
430 | | | | | | | | **r1** > 0.935237
431 | | | | | | | | **r8** <= 0.048328: banana (198.0)
432 | | | | | | | | **r8** > 0.048328: vinho (53.0/5.0)