UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CURSO DE TECNOLOGIA EM SISTEMAS PARA INTERNET
CÂMPUS GUARAPUAVA

FABIO KENJI OSHIRO TAKATUZI

Ad-DT: ALGORITMO INCREMENTAL PARA APRENDIZAGEM DE ÁRVORES DE DECISÃO ADAPTATIVAS

PROJETO DO TRABALHO DE CONCLUSÃO DE CURSO

GUARAPUAVA
1º Semestre de 2016

FABIO KENJI OSHIRO TAKATUZI

Ad-DT: ALGORITMO INCREMENTAL PARA APRENDIZAGEM DE ÁRVORES DE DECISÃO ADAPTATIVAS

Projeto do Trabalho de Conclusão de Curso de graduação, apresentado à disciplina de Trabalho de Conclusão de Curso 1, do Curso Superior de Tecnologia em Sistemas para Internet – TSI – da Universidade Tecnológica Federal do Paraná – UTFPR – Câmpus Guarapuava, como requisito parcial para obtenção do titulo de Tecnólogo em Sistemas para Internet.

Orientador: Profª. Ma. Renata Luiza Stange Carneiro Gomes

GUARAPUAVA
2016

RESUMO

TAKATUZI, Fabio K. O. Ad-DT: Algoritmo Incremental Para Aprendizagem de Árvores de Decisão Adaptativas. 50 f. Projeto de Trabalho de Conclusão de Curso – Curso Superior de Tecnologia em Sistemas para Internet, Universidade Tecnológica Federal do Paraná – UTFPR. Guarapuava, 2016.

A indução de árvores de decisão é um dos métodos mais utilizados para a resolução de problemas de classificação, tomada de decisão e aprendizagem de máquina. Através deste método é possível criar uma árvore de decisão baseada em um conjunto de dados de treinamento e posteriormente classificar novos dados com base na sua estrutura. A classificação gera um valor de saída, chamado classe, que age como resposta aos valores de entrada. Entretanto, existem muitas situações em que os dados de treinamento se encontram em constante mudança, como nos problemas de aprendizagem incremental, em que processo de aprendizagem considera que o conjunto de treinamento é dinâmico. Para problemas como este, a adaptatividade apresenta uma solução bastante aderente, pois permite que uma estrutura se automodifique em resposta a estímulos externos, incorporando a ela novas informações. Nesse contexto, este projeto propõe implementar um algoritmo para indução de árvores de decisão baseado em adaptatividade. O algoritmo deve ser capaz de alterar dinamicamente sua estrutura hierárquica, durante o processo de classificação. Dessa forma, espera-se que o algoritmo apresente uma alternativa para os métodos de aprendizagem de máquina tradicionais e também represente uma outra solução para problemas de aprendizado incremental.

Palavras-chave: Aprendizagem de Máquina, Adaptatividade, Árvores de decisão.

ABSTRACT

TAKATUZI, Fabio K. O. Ad-DT: Algoritmo Incremental Para Aprendizagem de Árvores de Decisão Adaptativas. 50 f. Projeto de Trabalho de Conclusão de Curso – Curso Superior de Tecnologia em Sistemas para Internet, Universidade Tecnológica Federal do Paraná – UTFPR. Guarapuava, 2016.

The decision tree induction is one of the most used methods to solve classification problems, decision-making and machine learning. By this method it is possible to create a decision tree based on a set of training data and then classify new data based on its structure. The classification generates an output value, called class, which acts in response to input values. However, there are many situations where the training data are constantly changing, as the problems of incremental learning, where learning process considers that the training set is dynamic. For problems such as this, the adaptivity has a very tacky solution because it allows a structure to auto modify in response to external stimuli, incorporating new information to it. In this context, this project proposes an algorithm for induction of decision trees based on adaptivity. The algorithm must be able to dynamically change its hierarchical structure during the classification process. Thus, it is expected that the algorithm presents an alternative for the traditional machine learning methods, and also represents another solution for incremental learning problems.

Key-words: Machine Learning, Adaptive technology, Decision tree.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

LTA – Laboratório de Linguagens e Técnicas Adaptativas

WEKA – Waikato Environment for Knowledge Analysis

TDIDT – Top Down Induction Decision Tree

UCI – UC Irvine

SUMÁRIO

1 INTRODUÇÃO

A resolução de um problema computacional bem especificado pode ser dada através de um algoritmo. Um algoritmo, por sua vez, é uma sequência de instruções computacionais que produzem, a partir de um valor de entrada, ou um conjunto deles, um valor ou conjunto de valores como saída (CORMEN et al., 2009).

No entanto, determinadas tarefas ou problemas existentes que não são passíveis de serem realizadas ou resolvidos utilizando algoritmos tradicionais, como por exemplo o reconhecimento correto de caracteres manuscritos ou qual a sequência ideal de jogadas para ganhar uma partida de damas. Esse tipo de situação necessita que o programa “aprenda” qual a melhor solução para os problemas ou qual a melhor maneira de se executar uma tarefa específica. Nesse contexto, o aprendizado de máquina é uma área da Inteligência Artificial que estuda a capacidade que os sistemas computacionais possuem de aprender e modificar seu comportamento através da experiência adquirida a fim de melhorar seu desempenho em execuções posteriores (ALPAYDIN, 2010).

No campo da aprendizagem de máquina existem diversos métodos de aprendizado, tais como aprendizado Bayesiano (DUDA, HART, 1973), redes neurais artificiais, modelo oculto de Markov e árvores de decisão (Quinlan, 1986). Nesse contexto, as árvores de decisão são um dos métodos mais práticos e utilizados em aprendizagem de máquina (MITCHELL, 1997). Os algoritmos de aprendizagem que implementam as árvores de decisão, em sua grande maioria, utilizam uma abordagem *top-down*, ou seja, constroem a árvore de cima para baixo e utilizam um método guloso, com base nos exemplos que se aplicam à sub-árvore corrente, para escolher qual o atributo de maior importância que será associado a raiz desta sub-árvore (MITCHELL, 1997).

De forma geral, os algoritmos de aprendizagem de máquia existentes classificam-se em incrementais e não-incrementais (MITCHELL, 1997). A grande

maioria desses algoritmos são não-incrementais, ou seja, não possuem a capacidade de incluir novos dados e informações no conjunto de treinamento ao longo de sua execução. A busca por algoritmos baseados em aprendizado incremental no contexto atual é de grande importância, pois existem diversas aplicações em que novas informações vão surgindo ou se alterando ao longo do tempo, formando assim um cenário em que a utilização de algoritmos incrementais é mais conveniente. (PISTORI; NETO, 2002). Alguns exemplos de algoritmos de indução árvores incrementais são o CART incremental (CRAWFORD, 1989), o ID5R (UTGOFF, 1989) e o ITI (UTGOFF; BERKMAN; CLOUSE, 1997).

Considere um sistema de monitoramento climático, o qual realiza leituras ambientais (velocidade do vento, temperatura média do ar, etc.) a todo momento. Suponha então, que o mesmo, por meio das leituras realizadas, precise identificar e aprender quais conjuntos de atributos e valores seriam mais apropriados para que um avião realizasse uma viagem de um lugar para outro de forma segura. Devido ao ambiente ao qual o software está submetido, os dados coletados estão constantemente se alterando, implicando assim em uma frequente mudança no conhecimento a ser aprendido e na recorrência do processo de aprendizagem. Dessa forma, a utilização de métodos não incrementais se torna inapropriada, uma vez que o conjunto instâncias de treinamento do algoritmo não são fixas (UTGOFF, 1989) e a repetição no processo de aprendizagem implica em um custo computacional elevado (YOSHIDA, 2007). Neste caso, surge a necessidade de implementar processos dinâmicos baseados na aprendizagem incremental, de maneira que a base de conhecimento possa ser atualizada sem que o processo de aprendizagem necessite ser repetido.

Um conceito que tem sido adotado na solução de problemas com características dinâmicas é a adaptatividade. De acordo com Neto (2011) a adaptatividade é definida pela capacidade que os sistemas possuem de promover espontaneamente alterações em seu próprio comportamento, de acordo com a necessidade, em função de seu comportamento corrente e dos valores de suas entradas. Para um problema de aprendizagem incremental, a utilização de técnicas

adaptativas pode representar uma solução bastante expressiva (NETO, 2000).

A tecnologia adaptativa consiste na utilização da adaptatividade de maneira prática, e corresponde ao conjunto de métodos, técnicas e ferramentas que visam a resolução de problemas concretos utilizando modelos baseados em dispositivos adaptativos (NETO, 2011). Um dispositivo adaptativo por sua vez é composto por um dispositivo não adaptativo (ex.: árvore de decisão, autômato finito), chamado de dispositivo subjacente, e uma camada adaptativa, que confere a este dispositivo a capacidade de automodificação (NETO, 2011).

Um exemplo de dispositivo adaptativo são as árvores de decisão adaptativas proposta por (PISTORI, 2003; PISTORI, H.; NETO, 2003), que permite que a estrutura hierárquica de uma árvore de decisão possa ser dinamicamente alterada durante o processo de decisão, quando a árvore é percorrida da raiz para as folhas.

1.1 Justificativa

Na corrente pesquisa será proposto um algoritmo incremental de indução de árvores de decisão baseado em técnicas adaptativas, que visa facilitar a representação e especificação de dispositivos que alteram continuamente sua estrutura e conjunto de treinamento. A utilização da tecnologia adaptativa na construção do algoritmo e a existência de poucos algoritmos que induzem árvores de decisão adaptativas, destacam-se como principais justificativas do projeto. Com isso, o algoritmo torna-se uma possível alternativa para a construção de árvores de decisão. Ademais, após a completa construção e implementação do algoritmo, o mesmo será distribuído de maneira *open source* auxiliando assim, a construção de futuros algoritmos para essa finalidade.

1.2 Objetivos

Nesta Seção serão apresentados os objetivos geral e específico do projeto.

1.2.1 Objetivo Geral

Avaliar a utilização da tecnologia adaptativa no processo de aprendizado de máquina aplicado às árvores de decisão.

1.2.2 Objetivos Específicos

- Desenvolver um algoritmo de indução de árvores de decisão adaptativas com desempenho comparável aos algoritmos de árvores adaptativas já existentes;
- Realizar a Implementação e os testes do algoritmo desenvolvido;
- Comparar o algoritmo proposto com os demais algoritmos de indução de árvores de decisão adaptativas e alguns algoritmos de indução de árvores tradicionais.

1.3 Estrutura do Projeto

Esta Seção tem por objetivo apresentar a divisão do conteúdo abordado pelo projeto. O Capítulo 2 corresponde a Resenha Literária utilizada no trabalho. Os procedimentos metodológicos previstos para o projeto são apresentados no Capítulo 3. No Capítulo 4 é apresentado o desenvolvimento do projeto, onde são abordados a estrutura e funcionamento do Ad-DT. O Capítulo 5 corresponde ao cronograma do projeto e para concluir, as Considerações Finais são abordadas no Capítulo 6.

2 RESENHA LITERÁRIA

Neste capítulo serão abordados os trabalhos correlatos e o embasamento teórico necessário para a estruturação e concepção do presente projeto.

2.1 ESTADO DA ARTE

A seguir será apresentado um breve histórico das pesquisas em adaptatividade, além de seu conceito aplicado a árvores de decisão.

2.1.1 Uma pequena história das pesquisas em Adaptatividade

O LTA – Laboratório de Linguagens e Técnicas Adaptativas[1] – nasceu em 1985, sendo impulsionado pelo interesse pedagógico no desenvolvimento de novas tecnologias e pela criação de linguagens de alto nível. Em 1993 todas as pesquisas realizadas até então se consolidaram e foram expandidas pelo novo conceito de autômato adaptativo, internacionalmente publicado pela primeira vez em Neto (1994). Esse formalismo representou um marco importantíssimo nessa área de pesquisa devido ao fato de ser equivalente, em poder computacional, às máquinas de Turing (ROCHA, 2001). A partir disso, o LTA de início a uma gama de pesquisas em adaptatividade.

Dispositivos baseados nos autômatos adaptativos possuem a característica de terem seu comportamento determinado por conjuntos variáveis de regras (NETO, 2011). Hoje, diversos são os dispositivos automodificáveis em uso, podemos citar as

1 Site do LTA - http://www.pcs.usp.br/~lta

Redes de Markov (Bassato, 2000), Redes de Petri, Tabelas de Decisão (TCHEMRA, 2009), Autômatos Diversos, sendo que da formulação e estudo a cerca desses conceitos produziram-se diversas teses, dissertações, aplicações em software, ferramentas e diversas outras publicações.

Com a criação de tantos trabalhos baseados na adaptatividade, surgiram diversas motivações para pesquisas na área, acarretando na criação de novos e ricos assuntos de pesquisa, temas para dissertações, além de aplicações interdisciplinares. Abaixo são apresentados alguns trabalhos que utilizam-se da tecnologia adaptativa como conceito fundamental.

Há muito tempo têm-se claro a ideia de que a comunicação (verbal e não verbal) é um dos principais meios de entendimento entre os seres humanos. Diversos avanços na área do reconhecimento de voz têm sido realizados permitindo que através de uma simples frase ou palavra, possam ser realizados um universo de ações e eventos.

Alfenas e Pereira (2012) apresentam uma arquitetura de referência para a construção de sistema de conversação para robôs sociáveis com a utilização de adaptatividade no gerenciamento de diálogo. O trabalho proposto propicia uma conversação sobre assuntos banais, permitindo que os robôs sociáveis possam interagir, de maneira natural, com determinada pessoa. O conceito da adaptatividade está atrelado às Máquinas de Markov Adaptativas que permitem que uma máquina de nível superior (Condutor) possa definir a condução do diálogo, estendendo-o de uma saudação para diversos rumos, até que a conversa evolua para uma despedida. No trabalho apresentado, o uso da tecnologia adaptativa viabilizou a utilização de uma técnica para construção de comportamentos considerados criativos, ou até mesmo humanos de maneira aceitável.

Outro trabalho interessante a ser citado é o de Cereda e Neto (2015), no qual são apresentados conceitos e técnicas para a escrita de código adaptativo em linguagens de programação orientadas a objetos convencionais. O trabalho baseou-se na utilização de duas técnicas para incorporar conceitos da adaptatividade nas

linguagens de programação. A primeira técnica abordada foi a reflexão, característica que permite a análise e modificação de classes em tempo de execução. O trabalho utilizou o conceito de herança do paradigma da orientação a objetos em conjunto com a reflexão para que a estrutura de uma classe fosse alterada. Dessa maneira, uma classe base é instanciada e uma nova classe é estendida da mesma, no entanto tendo sua estrutura modificada em tempo de execução.

A segunda técnica abordada no trabalho foi a utilização de blocos de código (*threading)*. Blocos de código podem conter definições aninhadas de outros blocos de código em suas especificações. Dessa forma, novos blocos podem ser criados e novas ligações dinâmicas podem ser estabelecidas. O trabalho de Cereda e Neto demonstra a viabilização da implementação de programas adaptativos, com a inserção de trechos de códigos com características de automodificação, utilizando técnicas tais como a reflexão ou particionamento por blocos de código.

2.1.2 Adaptatividade em Árvores de Decisão

O desenvolvimento de um algoritmo para indução de árvores de decisão que utiliza-se de técnicas adaptativas para sua concepção foi proposto anteriormente por Pistori e Neto (2002). O trabalho descreve um algoritmo denominado *Adaptree* que, utilizando conceitos da teoria dos autômatos e da tecnologia adaptativa, constrói uma árvore de decisão. O desempenho obtido apresenta resultados comparáveis aos principais algoritmos para aprendizagem de máquina, tornando-se o primeiro algoritmo de aprendizagem baseado em árvores de decisão adaptativas.

O algoritmo baseia-se na utilização e combinação de técnicas sintáticas e estatísticas, além de ser uma proposta de um algoritmo incremental, ou seja,

durante o processo de classificação de dados, novos exemplos podem ser incorporados a ele. A utilização de tecnologias adaptativas no trabalho, facilita a representação de dispositivos que sofrem constante alteração em resposta a estímulos externos. Os testes do Adaptree foram realizados sobre os conjuntos de dados da ferramenta WEKA (Waikato Environment for Knowledge Analysis) e comparados a outros algoritmos tradicionais não incrementais (ID3, Naive Bayes, C4.5, 5NN e Redes Neurais com *Backpropagation*).

Catae e Rocha (2011) através do uso de dispositivos adaptativos aplicados a árvores de decisão apresentam a possibilidade de uma modelagem adequada em operações de paridade, ou-exclusivo (XOR) e multiplexação, abordagens que não apresentavam bons resultados em estruturas de árvores de decisão tradicionais. O algoritmo propõe que a árvore seja capaz de associar novos atributos com base em atributos já existentes. A utilização de dispositivos adaptativos possibilita o funcionamento de um nível independente do módulo de construção principal da árvore, possibilitando que a mesma seja construída, no entanto, sem a realização de mudanças em sua estrutura principal. A modelagem do algoritmo baseia-se, fundamentalmente, no Princípio da Navalha de Occam e em conceitos da Complexidade Algorítmica (complexidade de Kolmogorov), que optam pela simplicidade como atributo de maior importância.

Silva et al. (2016) apresentou um trabalho que tem por objetivo concluir, a partir de sentimentos exibidos nos *tweets* da rede social Twitter, se uma população achou um determinado fato positivo ou negativo, além de realizar uma classificação das emoções expressadas pelas postagens. No trabalho, foi feito uso das árvores de decisão adaptativas para a determinação das palavras contidas nos textos a serem mineradas. Além disso, também foi utilizada na diferenciação e reconhecimento de palavras, bem como a sua devida classificação e categorização como sentimento. O trabalho proposto, através do uso de funções adaptativas e a utilização de dispositivos adaptativos tais como as árvores de decisão adaptativas obtiveram resultados bastantes expressivos com relação a tempo de processamento, tomada de decisão e no tamanho do caminho que o sistema irá percorrer para inferir,

corretamente, uma resposta.

2.1.3 Contribuições para o projeto

A tecnologia adaptativa e o aprendizado incremental são conceitos muito importantes quando a tarefa de aprendizagem precisa lidar com conjuntos de dados dinâmicos. O trabalho de Pistori e Neto, além de ser uma proposta de aprendizagem incremental, é o primeiro algoritmo a utilizar a estrutura das árvores de decisão e técnicas adaptativas para seu funcionamento, dando uma base fundamental para a elaboração do trabalho proposto. Como os testes do A*daptree* foram realizados apenas em comparação com algoritmos não incrementais, existe a possibilidade de testes futuros serem realizados apenas em comparação a ele, sem a necessidade de se realizar comparações com os algoritmos de indução de árvores tradicionais. Analogamente, o trabalho de Catae e Rocha também utilizou técnicas adaptativas para a construção de árvores de decisão, no entanto, partindo do pressuposto de que a simplicidade é um dos conceitos mais importantes. Nesse contexto, os princípios utilizados também podem contribuir para o trabalho proposto, uma vez que o mesmo também necessita de métricas em comum para a elaboração e determinação de melhores modelos. Além disso, seguindo a ideia da simplicidade, o algoritmo proposto pode apresentar resultados mais expressivos. É importante ressaltar que grande das partes das pesquisas em adaptatividade que utilizam árvores de decisões adaptativas baseiam-se, de maneira geral, em seus conceitos e aplicações, sem considerar sua implementação e formulação. O presente projeto têm seu foco principalmente nesses quesitos, que além de explorar os conceitos apresentados pelas árvores de decisão adaptativas, especifica sua implementação, formulação e especificação.

2.2 FUNDAMENTAÇÃO TEÓRICA

Esta Seção apresenta os principais conceitos utilizados no trabalho, sendo estes resultados da análise bibliográfica relacionada ao Aprendizado de Máquina e Tecnologia Adaptativa.

2.2.1 Aprendizagem de Máquina

O termo “aprendizagem de máquina” é um conceito da Inteligência Artificial que se refere à capacidade que os sistemas computacionais possuem de aprenderem e modificarem seu comportamento em resposta a estímulos e mudanças inferidas externamente ou também através da experiência adquirida ao longo de sua execução (ALPAYDIN, 2010). De acordo com Mitchell (1997) um programa aprende, a partir de uma de uma experiência *E* em respeito à uma classe de tarefas *T* e uma medida de desempenho *P,* se seu desempenho nas tarefas *T*, segundo a medida *P*, melhora com a experiência *E.*

Existem diversas situações nas quais a aprendizagem de máquina pode apresentar uma solução bastante expressiva. De maneira geral, um problema computacional bem especificado pode ser solucionado com a utilização de um algoritmo, contudo nem todos os problemas computacionais são passíveis de serem resolvidos com o uso de métodos tradicionais de programação. Por exemplo, imagine um sistema que fosse capaz de identificar quais e-mails, em um conjunto deles, são mal intencionados. Todos os dias, diversos e-mails de diferentes assuntos e nichos chegam à nossa caixa de entrada, entretanto cabe apenas a nós verificar quais deles são legítimos e quais não nos tem serventia. Imagine um outro sistema que possuísse a capacidade de realizar corretamente identificações faciais. Todos os dias ao encontrarmos com pessoas conhecidas como parentes próximos ou amigos, conseguimos, de maneira rápida e precisa, identificar tais pessoas. Entretanto, embora existam características que nos permitam realizar tais identificações, isto acontece de maneira inconsciente e dificilmente conseguimos explicar como

realizamos tais feitos. A realização das tarefas apresentadas, devido ao fato de não poderem ser explicadas, não podem ser representadas por um algoritmo computacional (ALPAYDIN, 2010). Neste âmbito, a utilização de conceitos da aprendizagem de máquina permitem que a prática de tais conjuntos de tarefas possam ser realizadas computacionalmente e de maneira eficiente.

2.2.1.1 Inferência Indutiva

A inferência indutiva ou indução é um conceito lógico que permite se obter conclusões genéricas a partir de um conjunto particular de exemplos. Na indução, um conceito sobre determinado assunto é aprendido através da inferência indutiva aplicada ao conjunto de exemplos disponíveis, de maneira que o raciocínio sobre tal conceito específico possa ser generalizado. O objetivo principal de um algoritmo indutivo é construir um classificador que seja capaz de determinar corretamente a classe para um determinado dado ainda não classificado, ou seja, que ainda não possui uma rotulação de classe (REZENDE, 2005). Os estudos que norteiam o aprendizado indutivo podem ser divididos basicamente em dois grupos: supervisionado e não-supervisionado.

Aprendizado supervisionado: É fornecido ao algoritmo de aprendizagem um conjunto de dados de treinamento nos quais os rótulos de classe dos dados associados a eles já são previamente conhecidos pelo fornecedor. Para as classes que possuírem valores de dados discretos, o problema é dito de classificação. Por outro lado, se as classes possuírem valores contínuos, o problema é categorizado como regressão. O objetivo desta forma de aprendizado é induzir conceitos que já estão previamente definidos, ou seja, que já possuem uma rotulação de classe.

Aprendizado não-supervisionado: Neste tipo de abordagem, a rotulação de classe não é conhecida e existe a incerteza sobre a saída de dados esperada. O algoritmo indutor analisa os dados disponíveis e tenta, utilizando medidas probabilísticas,

realizar um agrupamento dos dados. Após realizado esse agrupamento, o indutor verifica o que cada um dos grupos de dados formados significa no contexto do problema analisado.

2.2.1.2 Indução de Árvores de Decisão[2]

O campo da aprendizagem de máquina conta com diversos métodos de aprendizagem que visam aplicar os conceitos principais da área. Uma das técnicas de aprendizado mais utilizadas são as árvores de decisão, sendo elas um dos métodos mais simples e práticos para tratar a inferência indutiva.

Os algoritmos de aprendizagem de árvores de decisão utilizam a estratégia dividir para conquistar, de maneira que um problema mais complexo é decomposto em subproblemas mais simples e, dessa maneira, recursivamente, a mesma estratégia é aplicada para cada subproblema estruturando assim a árvore. Os atributos de maior importância vão sendo inseridos na árvore de decisão desde a raiz até os nós folha, conforme alguma medida estatística (ex. ganho de informação). Diversos algoritmos desenvolvidos para o aprendizado de árvores de decisão baseiam-se neste paradigma *top down,* que constroem a árvore escolhendo o atributo raiz das sub-árvores (MITCHELL, 1997). Alguns dos exemplos mais conhecidos são o ID3 (Quinlan, 1986) e o seu sucessor C4.5 (Quinlan, 1993).

O primeiro nó é chamado *raiz,* e por convenção, se localiza no topo da árvore. A partir do nó raiz, se estendem ramificações sucessivas denominadas *ramos,* que por sua vez, conectam os demais nós. Os últimos nós de uma árvore são denominados *folhas* e, neste contexto, representam uma decisão a ser tomada. Na estrutura dessa abordagem, cada nó da árvore especifica testes que devem ser realizados sobre algum atributo da instância de dados, e cada ramo descendente a partir desse nó corresponde a um dos possíveis valores que esse atributo pode assumir (Quinlan 1986).

O método para indução de árvores *Top Down Induction Decision Tree*

2 Seção elaborada a partir de (Mitchell, 1997).

(TDIDT) define um processo recursivo para a construção de árvores em que, dado um conjunto de treinamento vinculado a um nó, define o nó como folha, ou especifica outra propriedade para dividir o conjunto em um novo subconjunto estruturando a árvore. O ponto principal nos algoritmos de indução de árvores está em seu critério de escolha do melhor atributo que particionará o conjunto de exemplos, a fim de se obter subconjuntos mais homogêneos a cada iteração do algoritmo. De acordo com Mitchel (1997), para obter árvores compactas e com maior capacidade de tomar decisões corretamente, deve-se escolher o teste sobre algum atributo que gere nós mais "puros". Um nó é chamado "puro" quando possui todos seus exemplos de treinamento pertencentes a uma mesma classe. Nós que possuem essa característica também são denominados nós com grau de impureza nulo. Paralelamente a esse contexto, um nó é denominado "impuro" se possui seus exemplos de treinamento pertencentes a classes distintas.

Para a escolha dos atributos que melhor separem os exemplos de treinamento, é necessário a utilização de alguma medida estatística. Dessa forma é definida uma propriedade estatística denominada **ganho de informação** que mensura como um determinado atributo separa os exemplos de treinamento de acordo com sua classificação. Para os algoritmos de indução de árvores tradicionais, ID3 (Quinlan, 1986) e C4.5 (Quinlan, 1993), o ganho de informação é utilizado para definir, entre os candidatos, quais serão os atributos serão utilizados em cada iteração, durante a construção da árvore.

A definição do ganho de informação depende, primeiramente, do estabelecimento de uma medida chamada **entropia,** que determina o grau de impureza de um determinado conjunto arbitrário de exemplos. A entropia no contexto da indução de árvores, indica a homogeneidade dos exemplos do conjunto de treinamento. Dado um conjunto S de dados contendo exemplos positivos e negativos, a entropia relativa à este conjunto é definida por:

$$Entropia = -p\oplus \log_2 \oplus - p\ominus \log_2 \ominus$$

Onde $p\oplus$ é a proporção de exemplos positivos em S e $p\ominus$ é a proporção de exemplos negativos em S. Para o cálculo de entropia, toda expressão 0 log 0 é definida como sendo 0. Para exemplificar, suponha um conjunto S de

dados contendo 14 exemplos, sendo 9 exemplos positivos e 5 exemplos negativos (adotar a notação[9+, 5-]). Temos que para essa representação a entropia é dada por:

$$Entropia\ ([9+, 5-]) = -\ (9/14)\ log_2(9/14) - (5/14)\ log_2(5/14) = \mathbf{0.940}$$

A entropia será nula se todos os exemplos de dados contidos em S pertencerem à uma mesma classe. Dessa forma, se todos os membros de S são positivos ($p\oplus$ = 1), então por consequência $p\ominus$ = 0, implicando em Entropia (S) = -1 log2(1) - 0 log20 = -1. 0 - 0 log20 = **0.**

Após definida a medida de impureza (entropia), pode-se então, definir o ganho de informação para medir a efetividade de classificação de um determinado atributo. O ganho de informação de um atributo A, relativo a uma coleção de dados S é definida como:

$$Ganho(S, A) = Entropia(S) - \sum_{v \in Valores(A)} |S_v|/|S|\, Entropia(S_v)$$

Onde Valores(A) é o conjunto de todos os valores possíveis para o atributo A, S_v é o subconjunto de S no qual o atributo A possui valor v. Exemplificando, suponha que S é uma conjunto de treinamento de dias descrita por atributos, incluindo Vento, que pode ter os valores Fraco ou Forte. Análogo aos exemplos anteriores, assuma que S é uma coleção que contém 14 exemplos, [9+, 5 -]. Suponha que dos 14 exemplos, 6 positivos e 2 negativos possuem o atributo Vento = Fraco, e os demais exemplos possuam Vento = Forte. O ganho de informação adquirido classificando-se os 14 exemplos originais do atributo Vento pode ser calculado da seguinte maneira:

Atributos = Vento

Valores(Vento) = Fraco, Forte

S = [9+, 5-]

$S_{forte} \leftarrow$ *[6+, 2-]*

$S_{forte} \leftarrow$ *[3+, 3-]*

$$Ganho(S, Vento) = Entropia(S) - \sum_{v \in (Fraco, Forte)} |S_v|/|S|\, Entropia(S_v)$$

$= Entropia(S) – (8/14)\ Entropia(S_{Fraco}) - (6/l4)\ Entropia(S_{Forte})$

$= 0.940 - (8/14)0.811 - (6/14)1.00$

$= \mathbf{\underline{0,048}}$

2.2.1.3 Algoritmo C4.5

O C4.5 (Quinlan, 1993) é um exemplo clássico de algoritmo para indução de árvores de decisão considerado como sucessor do algoritmo ID3 (MITCHELL, 1997). O C4.5 além de utilizar o método TDIDT, utiliza também a medida de ganho de informação para selecionar o melhor atributo a cada passo da construção da árvore. A seguir são apresentados os passos utilizados pelo algoritmo para a indução de árvores de decisão:

1. Iniciamos com o conjunto de dados rotulados D e uma árvore vazia T.
2. Se D for composto de dados da mesma classe ou outro critério de parada for atingido, interrompa o algoritmo (criando uma folha contendo D).
3. Para cada atributo a existente em D calcule o ganho de informação para divisão do conjunto de dados utilizando o atributo A.
4. Selecione o melhor ganho de informação. Crie dois ou mais subconjuntos de D baseados na separação usando A.
5. Para cada subconjunto Da crie uma árvore vazia T_A e execute o algoritmo recursivamente e anexe a árvore T_A a T.
6. Retorne a árvore T com todas as sub-árvores concatenadas.

Este algoritmo pode utilizar diversos tipos atributos para a divisão dos dados. Dentre eles podemos citar atributos binários, que dividem o conjunto D em dois subconjuntos; atributos categóricos, que permitem a criação de subconjuntos proporcional ao número de atributos existentes; e atributos numéricos cuja divisão acontece de acordo com limiar que maximiza o ganho de informação dos grupos criados. Árvores de decisão criadas com o algoritmo C4.5 possuem a propriedade de poda, que permite a substituição de sub-árvores inteiras por uma folha, caso não haja aumento na taxa de erros do classificador com tão substituição.

2.2.2 Adaptatividade

A concepção de programas automodificáveis é bastante antiga, sendo ela advinda da programação dos primeiros computadores e perdurando até os dias atuais. A ideia da adaptatividade, no entanto, é um conceito bastante novo, sendo apresentado internacionalmente pela primeira vez em (NETO, 1994). Sua definição pode ser sumarizada à capacidade que os sistemas computacionais possuem de realizar alterações em sua estrutura interna em resposta a estímulos de entrada, situação e comportamento correntes.

Um sistema adaptativo têm seu comportamento determinado por um conjunto de regras que, quando exposto a novos estímulos de entrada em uma dada situação, podem alterar, também, seu comportamento subsequente realizando eventuais remoções e inserções dessas regras (NETO, 2011). A seguir serão abordados alguns conceitos que ajudam a materializar a ideia da adaptatividade.

2.2.2.1 Dispositivos Adaptativos

O conceito de dispositivos guiados por regras é definido, fundamentalmente, pela característica que os mesmos compartilham de possuírem sua representação e especificação guiada por um conjunto finito e variável de regras. Tais regras ditam o funcionamento do dispositivo, bem como o mesmo será estruturado ao longo de sua execução, levando em consideração cada estímulo de entrada e a geração de uma determinada saída. O funcionamento de um dispositivo guiado por regras inicia-se em uma dada configuração. Então se aplica, sucessivamente, cada uma das regras do conjunto especificado até o ponto em que não existam mais regras ou estímulos externos, ou até o momento em que os mesmos não possam mais ser aplicados (PISTORI, 2003; NETO, 2011).

Um dispositivo adaptativo ou, dispositivo guiado por regras adaptativo, passa a ter seu conjunto de regras variável durante sua leitura, todavia essa capacidade é

conferida a uma outra camada, que funciona sobre o conjunto original modificando-o através de inserções e remoções de regras. Dessa maneira, temos uma forma de operação do dispositivo adaptativo composto por duas camadas. A primeira delas, denominada *camada subjacente,* é composta por um dispositivo guiado por regras não adaptativo (árvores de decisão, autômato de pilha estruturado etc.). A segunda camada é chamada *adaptativa,* a qual possui características adaptativas que conferem à camada subjacente a propriedade automodificadora, sendo esta capaz de transformar um dispositivo não adaptativo em um que seja capaz de realizar mudanças em sua estrutura durante a operação (PISTORI, 2003).

Um dispositivo adaptativo pode ser formulado a partir de uma dupla DA = (CS, CA) em que CS representa um dispositivo guiado por regras, não adaptativo (camada subjacente) e CA representa a camada adaptativa a ele acoplada (PISTORI, 2003). De acordo com Neto (2009), a descrição de um dispositivo, não adaptativo e guiado por regras, pode ser dada por meio de uma quíntupla de seguinte formulação:

$$CS = (C, R, S, c_0, A)$$

onde:

C é o conjunto de todas as possíveis configurações para o dispositivo.
R é uma relação de mudança de configuração para CS: $R \subseteq C \times (S \cup \{ \varepsilon \}) \times C$.
S corresponde ao conjunto fixo e finito de eventos dados como estímulos válidos de entrada para CS.
$\mathbf{C_0 \in C}$ representa a configuração inicial do dispositivo.
$\mathbf{A \subseteq C}$ representa o conjunto de todas as configurações de aceitação do dispositivo.

Ademais, a camada adaptativa (CA) é definida por:

$$CA = (B, Z)$$

na qual:

B é um conjunto de ações adaptativas, que contém também o valor nulo, o qual

representa as ações que não causam modificações na camada subjacente.
Z:R → B² é a representação de uma função que mapeia cada regra da camada subjacente em um par ordenado de ações adaptativas. Ambas ações deste par devem ser acionadas, respectivamente, antes e depois de cada regra à qual o par está associado.

Esta camada adaptativa é representada por um conjunto de funções que são acopladas às regras da camada não adaptativa. Tais funções são capazes de realizar eventuais alterações no conjunto de regras do dispositivo subjacente implicando em mudanças no comportamento do dispositivo adaptativo (STANGE, 2011). Dessa maneira, uma formulação para tal dispositivo pode ser descrita por uma sêxtupla (NETO, 2009) da seguinte forma:

$$AD = (C, RA, S, AA, C_0, RA_0, A),$$ onde:

C é o conjunto de todas as possíveis configurações para o dispositivo adaptativo.
RA é o conjunto de regras adaptativas de AD.
S corresponde ao conjunto de estímulos válidos de entrada para AD.
$C_0 \in C$ representa a configuração inicial e única do dispositivo adaptativo.
RA é o conjunto de regras iniciais, fixas e adaptativas de AD.
$A \subseteq C$ é o conjunto de todas as configurações de aceitação de AD.
AA é o conjunto de funções adaptativas[3] de AD.

2.2.2.2 Funções adaptativas

O objetivo das funções adaptativas é definir quais mudanças serão realizadas na camada subjacente quando uma determinada ação adaptativa for acionada. Dessa maneira, pode-se dizer que uma ação adaptativa é definida pela chamada de uma função adaptativa. Uma função adaptativa possui como característica principal uma lista de ações adaptativas sendo elas: ações de consulta, inserção e remoção

3 Uma função adaptativa possui estrutura similar às funções das linguagens de programação. Sua formulação segue na Seção 2.2.2.2.

de regras do conjunto que permeia a camada subjacente (PISTORI, 2003). Formalmente, é possível definir uma função adaptativa através de um 9-upla da seguinte maneira:

FA = (F, P, V, G, C, R, I, A, B), na qual:

F representa o nome da função adaptativa.

P é a ênupla ($p_1, p_2, p_3, \ldots, p_p$) de parâmetros formais.

V é a lista ($v_1, v_2, v_3, \ldots, v_v$) de variáveis. Tais variáveis têm seus valores preenchidos uma única vez através das ações adaptativas de consulta.

G é a lista ($g_1, g_2, g_3, \ldots, g_g$) de geradores. Os geradores têm seus valores preenchidos a cada nova chamada da função adaptativa.

C é a lista das ações de consulta.

R representa a lista de ações de remoção.

I representa as ações de inserção.

A é uma função adaptativa inicial que pode, opcionalmente ser executada antes de F.

B é uma função adaptativa final que pode, opcionalmente ser executada depois de F.

2.2.2.3 Autômatos adaptativos

Um autômato adaptativo pode ser definido por meio de um dispositivo adaptativo guiado por regras, no qual a camada adaptativa está acoplada a um mecanismo subjacente representado por um autômato de pilha estruturado e as ações dessa camada são implementadas através das funções adaptativas.

2.2.2.4 Árvores de Decisão Não-Determinísticas

A abordagem da teoria dos dispositivos adaptativos, que visa a conjunção de dispositivos guiados por regras não adaptativos com a tecnologia adaptativa, apresenta um novo enfoque para algoritmos de indução árvores de decisão. Este enfoque apresenta um dispositivo adaptativo cujo mecanismo subjacente é uma árvore de decisão, resultando em um dispositivo denominado árvore de decisão adaptativa.

Pistori (2003), apresenta uma abordagem para um dispositivo adaptativo, o qual utiliza como camada subjacente uma estrutura denominada *árvore de decisão não-determinística (árvore-DND).* Tal estrutura, ao contrário das árvores de decisão tradicionais que só obtêm uma decisão a partir do momento em que é estabelecido um caminho completo da raiz para alguma de suas folhas, permite que mesmo na ausência ou inconsistência de valores para determinados atributos, a busca por um caminho da raiz até alguma de suas folhas permaneça ininterrupta, utilizando-se de todas as arestas advindas do nó correspondente ao valor ausente ou inconsistente.

A Figura 1 apresenta uma árvore de decisão com objetivo de determinar uma ação (*sair* ou *não-sair*) para um determinado casal baseado no tempo (*chuvoso* ou *ensolarado*) e no clima (frio, quente, ameno ou úmido) de um determinado dia da semana.

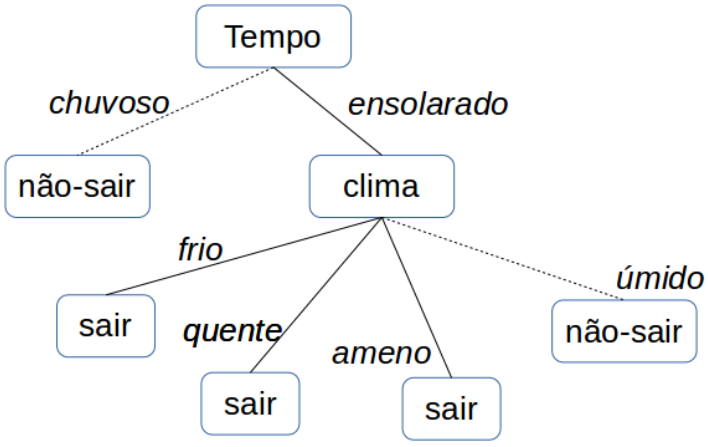

Figura 1: Exemplo de árvore de decisão

Fonte: Autoria própria.

Na árvore apresentada, se o **tempo** para um determinado dia fosse desconhecido e o **clima** fosse úmido, com a utilização de árvores de decisão o processo de decisão não seria praticável, pois existiria a ausência de uma aresta que ligasse a folha à raiz. No entanto, com a utilização da árvore de decisão não-determinística, existiriam duas arestas que ligariam a raiz às folhas, ambas apontando para a ação não-sair. Já em uma outra situação, na qual um dia contasse com o tempo desconhecido mas com o clima ameno, a árvore de decisão não-determinística retornaria como saída uma probabilidade de 0,5 para sair e para não-sair, indicando a possibilidade de ocorrência de chuvoso ou ensolarado para o atributo tempo.

A formulação de uma árvore-DND é dada através de uma 7-upla T = (*I, Σ, Γ, f, c, A, S*) na qual:

I, Σ, Γ são conjuntos não-vazios representando, respectivamente, *exemplos, atributos* e *valores.* Σ e Γ são conjuntos finitos ao contrário de I que é contavelmente infinito. O conjunto Γ possui o valor "**?**" (valor ausente) que tem como objetivo representar valores ausentes, desconhecidos ou inconsistentes.

$\boldsymbol{f : \Sigma \rightarrow 2^{\Gamma}}$ é uma função que mapeia cada atributo contido em Σ em um conjunto de valores possíveis para esse atributo, representando assim o domínio de cada atributo.

$\boldsymbol{c \in \Sigma}$ representa o atributo classe o qual têm seus valores mostrados nas folhas de uma árvore de decisão.

$\boldsymbol{A : I \times \Sigma \rightarrow \Gamma}$ é uma função binária que têm como objetivo descrever cada elemento do conjunto de exemplos, criando uma associação entre exemplos, atributos e valores.

S é uma estrutura hierárquica finita, denominada sub-árvore, e definida recursivamente como:

> **Folha:** uma dupla (*id, v*), onde $v \in$ f(c) representa um valor para o atributo classe, e *id* é um identificador único.
>
> **Não Folha:** Uma (n+2)-upla (*id*, a, (v_1, S_1), ..., (v_n , S_n)), na qual *id* é um identificador, a $\in \Sigma - \{c\}$ é um atributo, $v_i \in$ f(a), $1 \leq i \leq n$ são valores e todos os elementos S_i , $1 \leq i \leq n$ são sub-árvores.

2.2.2.4.1 Exemplo de árvore-DND

A árvore de decisão apresentada na Figura 1 pode ser formalmente representada por T = (*I, Σ, Γ, f, c, A, S*) onde:

- *I* é um conjunto de exemplos de dias da semana e decisões relacionadas a dias, por exemplo: {*segunda, terça, decisão1, decisão2}.*
- *Σ* = {*tempo, clima, ação*}
- *Γ* = {*chuvoso, ensolarado, frio, quente, ameno, úmido, sair, não-sair,* **?**}
- *f* é definido através da tabela:

Σ	**f**
tempo	{*chuvoso, ensolarado,* **?**}
clima	{*frio, quente, ameno, úmido,* **?**}
ação	{*sair, não-sair,* **?**}

- *c* = {*ação*}
- *A* é definida pela seguinte tabela:

I	*tempo*	*clima*	*ação*
segunda	*chuvoso*	*frio*	*não-sair*
terça	*ensolarado*	*quente*	*sair*
decisão1	**?**	*ameno*	**?**
decisão2	ensolarado	úmido	**?**

- *S* é dada por:
  - (S_0, *tempo, (chuvoso,* S_1*), (ensolarado,* S_2*))*
  - *(*S_1, *não-sair)*
  - *(*S_2, *clima, (frio,* S_3*), (quente,* S_4*), (ameno,* S_5*), (úmido, (*S_6*, não-sair)))*
  - *(*S_3*, sair)*
  - *(*S_4*, sair)*
  - *(*S_5*, sair)*

Através do identificador (*id*) é possível referenciar toda a sub-árvore. No exemplo acima, a sub-árvore S_6 foi alocada dentro de R_2, demonstrando que é possível descrever uma sub-árvore dentro de outra.

2.2.2.5 Árvore-DND adaptativa

Com a aplicação da tecnologia adaptativa na estrutura da árvore de decisão apresentada anteriormente, obtêm-se um dispositivo guiado por regras adaptativo denominado árvore de decisão não-determinística adaptativa, ou árvore-DND adaptativa. A camada adaptativa do dispositivo permite agora que a estrutura S da árvore seja modificada antes ou depois de cada modificação ocorrida na camada subjacente, sendo que todas as alterações realizadas são obtidas a partir de um conjunto de ações adaptativas[4]. Todas as alterações realizadas pela camada

4 Disponível na Seção 2.2.2.2.

adaptativa por meio das ações adaptativas podem ser aplicadas a qualquer sub-árvore gerada (PISTORI, 2003).

No caso das árvores-DND adaptativas, tais ações adaptativas pode ser representadas através de sub-árvores genéricas, citadas na Seção anterior, cercadas de parênteses e precedidas pelos símbolos que correspondem ao tipo da ação elementar: “+” para inserção, “-” para remoção e “?” para consulta. Para completar e facilitar a especificação da camada adaptativa, também utiliza-se o símbolo τ para denotar o exemplo corrente no conjunto de exemplos de treinamento. Dessa forma, um exemplo de uma ação adaptativa de consulta pode ser dado da seguinte maneira: ?[((τ, ?*atributo*), ?*valor)*]. Substituindo-se ?*atributo* pelo nome do atributo, obtêm-se os valores para o atributo especificado (PISTORI, 2003).

De maneira geral, a formulação de uma árvore de decisão adaptativa é dada pela mesma representação do dispositivo adaptativo apresentado na Seção 2.2.2.1, através de uma dupla ADT = (*CS, CA*). Entretanto, o mecanismo subjacente neste caso é representado por uma árvore-DND CS = (*I, Σ, Γ, f, c, A, S*) possuindo uma alteração em sua estrutura, que agora aceita a vinculação de ações adaptativas em cada uma das sub-árvores de S. Analogamente, a camada adaptativa *CA* é constituída pelo conjunto de ações adaptativas citadas acima.

A operação de uma árvore de decisão não-determinística adaptativa segue de igual maneira ao mecanismo subjacente (Seção 2.2.2.4), até o momento em que haja a chamada de uma função adaptativa. Da mesma forma, a estrutura de uma função adaptativa segue a definição geral das funções adaptativas definidas para os dispositivos adaptativos, descritos na Seção 2.2.2.1, no entanto, com as alterações propostas acima.

2.2.2.5.1 Exemplo de Árvore-DND adaptativa

Para exemplificar a estrutura de uma árvore de decisão não-determinística adaptativa, utilizaremos a árvore-DND apresentada na Seção 2.2.2.4, que determina se um casal deve ou não sair de casa em um determinado dia da semana. Este

exemplo estenderá a árvore anteriormente apresentada através de uma função adaptativa que acrescentará a ela uma nova regra: para dias da semana em que o tempo for chuvoso e o clima ameno, a ação para o casal neste caso será de sair, ao contrário do que a árvore sugeria originalmente.

A árvore-DND adaptativa que denota o caso exposto acima é dada por ADT = (*CS, CA*) em que o dispositivo subjacente *CS* é representado por T, visto na Seção 2.2.2.4.1. No entanto, será acoplada uma ação adaptativa A_1 à sub-árvore S_1 de *S* e a mesma passará a ser: [A_1] (*S_1, não-sair*). A função adaptativa A_1 tem por objetivo substituir a sub-árvore S_1 que determina a ação não-sair, por uma outra sub-árvore um pouco mais complexa que, mesmo possuindo o atributo tempo com valor chuvoso, também leva em consideração o clima, alterando a ação para sair, se o mesmo possuir valor ameno. A função adaptativa A_1, descrita na Tabela 1, possui também uma função adaptativa A_2 que têm como finalidade remover a nova sub-árvore corrente e chamar a função A_3, que por sua vez, retorna a árvore à sua situação inicial. As Figuras 2 e 3 apresentam, respectivamente a árvore de decisão após a realização da função A_1 e após a realização das funções A_2 e A_3.

1	A_1	Cabeçalho da função
2	+ [(*S_7, clima*, (*ameno*, (*S_8*, *sair*) [A_2]), (*frio*, (*S_9*, *não-sair*) [A_2]), (*quente*, (*S_{10}*, *não-sair*) [A_2]), (*úmido*, (*S_{11}*, *não-sair*) [A_2]))]	Ação elementar de inserção
1	A_2	Cabeçalho da função
2	- [(*S_7*, *clima*), [A_3]]	Ação elementar de remoção
1	A_3	Cabeçalho da função
2	+ [[A_1] (*S_1*, *não-sair*)]	Ação elementar de inserção

Tabela 1: Funções adaptativas do exemplo de Árvore-DND adaptativa

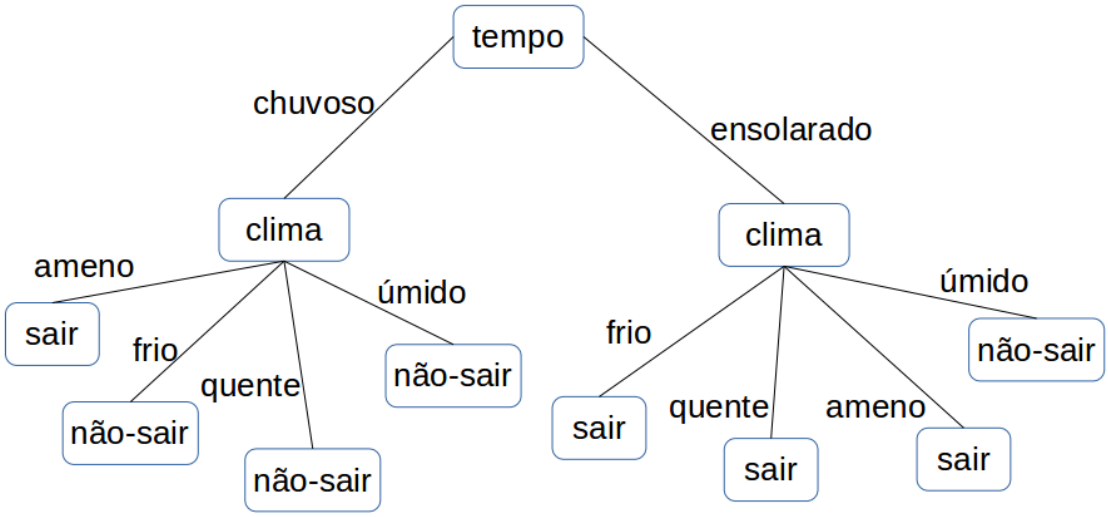

Figura 2: Árvore após a execução da função [A1]

Fonte: Autoria própria.

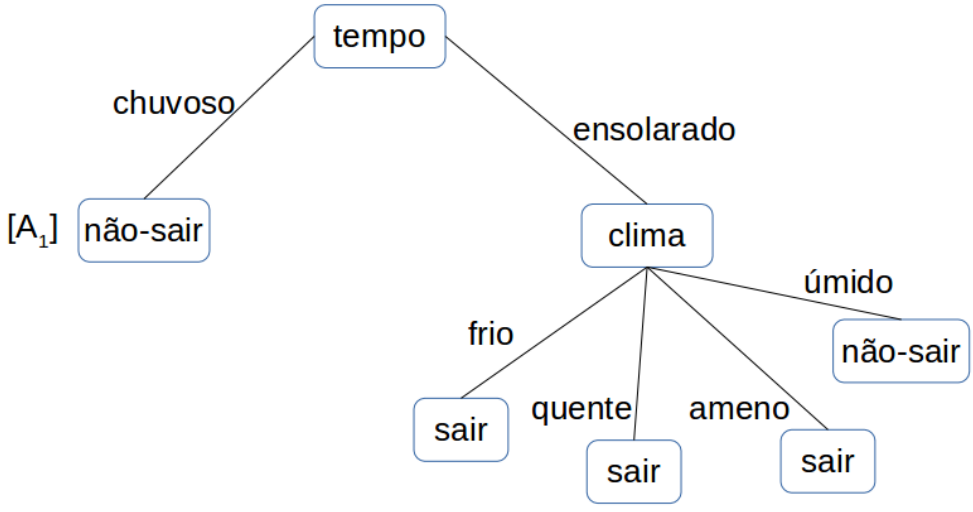

Figura 3: Árvore após a execução das funções [A2] e [A3]

Fonte: Autoria própria.

2.2.3 Ferramenta WEKA

Waikato Environment for Knowledge Analysis – WEKA[5] (Waikato, 2004; Witten; Frank, 2000) é uma ferramenta *open source software* desenvolvida na Universidade de Waikato (Nova Zelândia), que contempla uma série de algoritmos utilizados para realizar tarefas de aprendizagem de máquina e mineração de dados.

O software foi escrito em linguagem de programação JAVA além de possuir uma abrangente documentação e uma interface gráfica que facilita a sua utilização. Dentre suas funcionalidades, permite o acesso a relatórios com dados analíticos e estatísticos, tornando-se um grande atrativo para diversos tipos de aplicações (SILVA, 2004).

A WEKA disponibiliza uma interface gráfica chamada Explorer que contém grande parte de suas funcionalidades. A Figura 4 apresenta em 5 abas as principais tarefas suportadas pela ferramenta (BOUCKAERT et al., 2016).

(1) Opções disponíveis: tarefas relacionadas à aprendizagem de máquina e mineração de dados.

***Preprocess*:** permite escolher o conjunto de dados de treinamento e modificá-los aplicando diferentes tipos de filtros

***Classify*:** possibilita treinar e testar sistemas de aprendizagem que realizam classificação ou regressão.

***Cluster*:** para aprendizado não supervisionado através de algoritmos de clusterização.

5 Disponível em www.cs.waikato.ac.nz/ml/weka/.

***Associate*:** permite o aprendizado de regras de associação para um conjunto de dados.

***Select attributes*:** disponibiliza algoritmos para seleção de atributos mais relevantes aplicados a um conjunto de dados.

***Visualize*:** visualização de um gráfico 2D interativo dos dados.

(2) Open file, Open URL, Open DB, Generate: permitem, respectivamente, o carregamento de bases de dados a partir de pastas locais, bases de dados remotos (Web), bancos de dados e a geração artificial de conjuntos de dados a partir de DataGenerators.

(3) Filter: nesta opção é possível efetuar sucessivas filtragens de atributos e instâncias na base de dados previamente carregada (seleção, discretização, normalização, amostragem, dentre outros).

(4) Attributes: mostra os atributos existentes no conjunto de dados especificado.

(5) Selected attribute: apresenta informações estatísticas e quantitativas acerca dos atributos selecionados em (4).

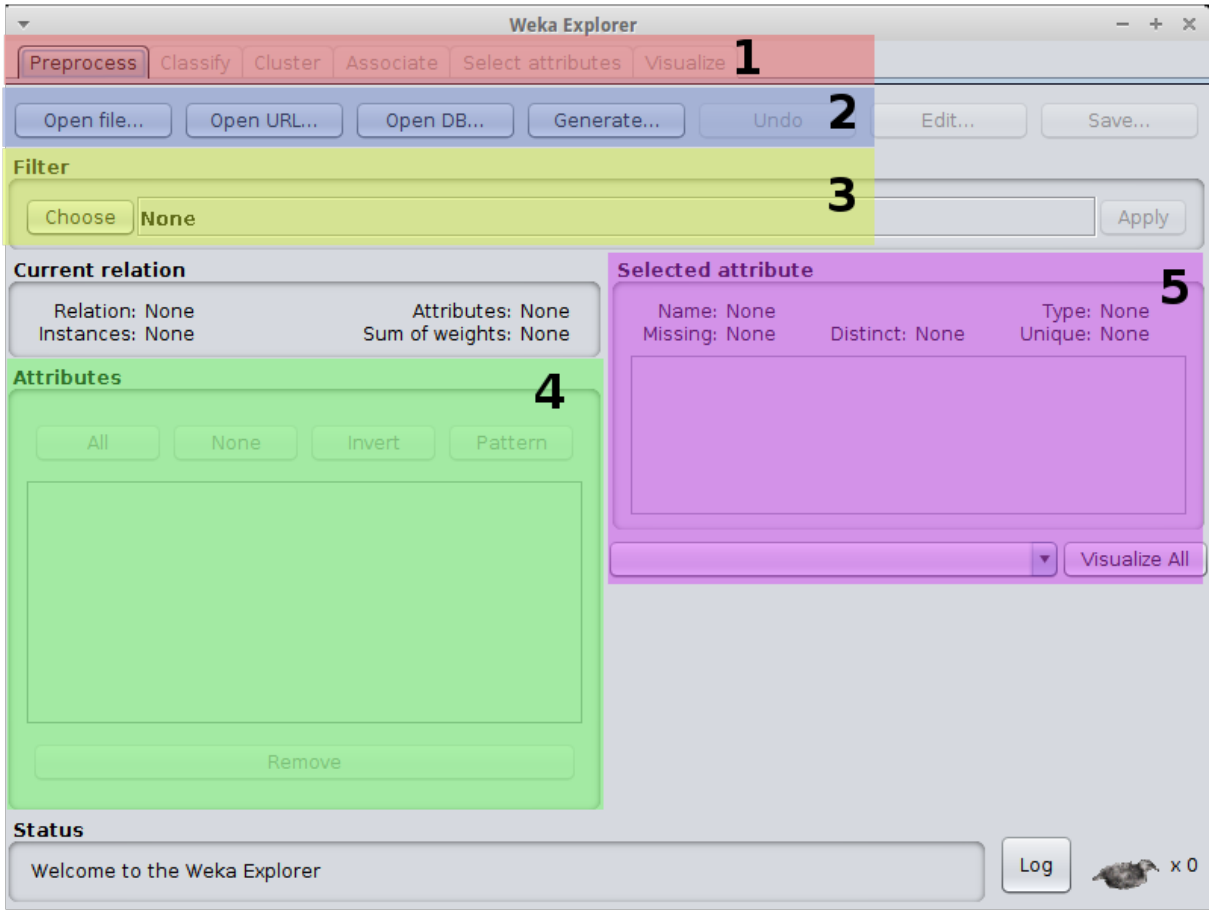

Figura 4: Interface Explorer do WEKA (pré-processamento)

Fonte: Autoria própria.

3 METODOLOGIA

O presente capítulo apresenta os passos metodológicos para o desenvolvimento do projeto, organizados da seguinte maneira:

1. Estudar os conceitos gerais de aprendizagem de máquina e adaptatividade.

    - Estudar o comportamento de dispositivos adaptativos aplicados a árvores de decisão.
    - Estudar o funcionamento dos algoritmos de aprendizagem de máquina clássicos (ex. ID3, C4.5, ID5R, Adaptree).

2. Propor um algoritmo para árvores de decisão adaptativas baseado no conceito da adaptatividade e aprendizagem incremental.

    - A estrutura do algoritmo será baseada no algoritmo Adaptree.

3. Implementar e testar o algoritmo de criação de árvores de decisão adaptativas.

    - O algoritmo implementado será integrado com a ferramenta WEKA[6] (*Waikato Environment for Knowledge Analysis*), que é um ambiente para execução de algoritmos de aprendizagem de máquina.
    - Os testes do algoritmo serão realizados utilizando medidas de precisão e

6 Disponível na Seção 2.2.3.

taxas de erro que serão calculados pela própria ferramenta.

4. Comparar o algoritmo proposto com alguns algoritmos de indução de árvores de decisão incrementais e não incrementais.

   - Serão aplicados diferentes algoritmos de aprendizagem de máquina em conjuntos de dados disponíveis no UCI[7], que é um repositório de *data sets* para testes de aprendizagem de máquina.

7 Disponível em http://archive.ics.uci.edu/ml/.

4 DESENVOLVIMENTO DO PROJETO

O desenvolvimento do projeto baseia-se na criação de um algoritmo para indução de árvores de decisão utilizando técnicas adaptativas. Para isso, será utilizado o conceito de dispositivo adaptativo, o qual é constituído de uma camada subjacente que, para o Ad-DT, será utilizada a árvore de decisão não-determinística, e uma camada adaptativa. Esta última será composta basicamente de ações adaptativas elementares, conferindo assim à camada subjacente a capacidade automodificadora. Este capítulo têm como objetivo descrever a estrutura do algoritmo Ad-DT, bem como seu funcionamento.

4.1 Algoritmo Ad-DT

A ideia geral do algoritmo é transformar cada exemplo do conjunto de treinamento em uma cadeia de dados e armazená-los. Dessa maneira, a estrutura da árvore se torna similar a uma árvore de prefixos, na qual cada nível da árvore representa um atributo do conjunto de dados. O Ad-DT utilizará a abordagem incremental para dados discretos, possibilitando que exemplos de treinamento possam ser fornecidos separadamente, além de ser possível intercalar os mesmos com exemplos de teste.

Uma vez que a árvore for construída, seu aprendizado terá sido baseado apenas no conjunto de treinamento fornecido para o algoritmo. Entretanto, a natureza não-determinística advinda do dispositivo subjacente permite que a árvore possa generalizar além dos exemplos concedidos. Ademais, o conceito de não-determinismo em conjunto com medidas probabilísticas também permite ao Ad-DT o tratamento de valores ausentes ou inconscientes.

O Ad-DT contém um conjunto de funções adaptativas iniciais as quais impõe uma ordenação pré-determinada para o conjunto de atributos Σ em que o atributo classe, por convenção é posicionado por último. Essa ordenação determina em quais níveis da árvore cada um dos atributos será tratado. Inicialmente, a estrutura S

da árvore inicia-se com a raiz possuindo um valor desconhecido "**?**" e uma ação adaptativa. Conforme os exemplos de treinamento vão sendo lidos, a estrutura vai crescendo até o tamanho máximo igual ao total de atributos (Σ), uma vez que cada nível representa um atributo do conjunto de dados de treinamento. Dessa forma, o crescimento da árvore se concentrará mais em largura do que em altura.

A árvore-DND adaptativa do Ad-DT pode ser formalmente definida por uma dupla ADT = ((*I, Σ, Γ, f, c, A, S), Φ*), onde o conjunto de atributos *Σ* possui uma ordenação arbitrária podendo ser representado por uma sequência $a_1, \ldots, a_j$, na qual j = total de atributos (*Σ*), $a_j = c$ representa o atributo classe, *S* = $[A_1]$ $(S_1, ?)$ e a camada adaptativa *Φ* é composta por um conjunto de funções adaptativas $A_1, \ldots, A_j$. Todas as funções adaptativas A_k, uma para cada atributo, possuem uma única ação elementar, de inserção, com a seguinte configuração:

1	A_k	Cabeçalho da função
2	+ [($*r_0$, a_i, ($f_1(a_i)$, $[A_{i+1}](*r_1, ?)$), ($f_2(a_i)$, $[A_{i+1}](*r_2, ?)$), …, ($f_v(a_i)$, $[A_{i+1}](*r_v, ?)$))]	Ação elementar de inserção

,

na qual $*r_v$ são geradores, a_i é o i-ésimo atributo de *Σ*, $f_c(a_i)$ representa o c-ésimo valor possível para o atributo a_i, para $1 \leq c \leq v$, sendo *v* o total de valores possíveis para a_i. A última função adaptativa A_j, apresentada abaixo, tem por objetivo criar um nó folha contendo a classe do exemplo de treinamento corrente

1	A_j	Cabeçalho da função
2	? [((τ, ?*atributo_classe*), ?*valor)*]	Ação elementar que obtém o valor do atributo classe
3	+ [(*r, ?*valor*)]	Adiciona a folha contendo o valor obtido em 2.

4.2 Exemplo Ilustrativo

Novamente pegando o exemplo sobre dias da semana, tempo e clima, a árvore de decisão adaptativa a ser gerada pelo Ad-DT pode ser formalizada da seguinte maneira: ADDT = ((*I, Σ, Γ, f, c, A, S), Φ*) onde:

- *I, Σ, Γ, f, c, A* são idênticos ao exemplo em 2.2.3.4.1;
- *S* é uma estrutura formada pela sub-árvore [A_1] (S_1);
- *Φ* é formada pelo conjunto de 3 funções adaptativas mostradas na Tabela 2;

1	A_1	Cabeçalho da função
2	+ [(*r_0, *tempo*, (*chuvoso*, [A_2] (*r_1, ?)), (*ensolarado*, [A_2] (*r_2, ?)))]	Ação elementar que insere a sub-árvore para o atributo *tempo*
1	A_2	Cabeçalho da função
2	+ [(*r_0, *clima*, (*frio*, [A_3] (*r_1, ?)), (*quente*, [A_3] (*r_2, ?)), (*ameno*, [A_3] (*r_3, ?)), (*úmido*, [A_3] (*r_4, ?)))]	Ação elementar que insere a sub-árvore para o atributo *clima*
1	A_3	Cabeçalho da função
2	? [((τ, *ação*), ?valor)]	Ação elementar de consulta que obtém o valor para o atributo classe *ação*
3	+ [(*r, ?valor)]	Ação elementar que insere o nó folha contendo o valor para *ação*

Tabela 2: Funções adaptativas do Exemplo Ilustrativo

Fonte: Auditoria Própria

4.3 Funcionamento do Ad-DT

Para facilitar a compreensão do algoritmo Ad-DT, o funcionamento do mesmo será divido em 3 etapas: **Entrada, Processamento e Saída**.

- **Entrada:**

  - ***Leia*** o conjunto I_1 de dados de treinamento;
  - ***Executa*** o conjunto de funções adaptativas iniciais para obter os atributos *a* com uma ordenação imposta sendo o último, o atributo classe;
  - ***Obtenha*** os valores possíveis para cada atributo;

- **Processamento:**

  - ***Inicia*** a estrutura *S* da árvore contendo apenas o nó raiz com um atributo desconhecido **?** e uma função adaptativa A_1;
  - ***Executa*** uma função F_1 que recebe como parâmetro uma instância Y_z do conjunto I_1 de dados de treinamento para $1 \leq z \leq$ total de instâncias de treinamento;
    - ***para*** k = 1 até j, onde j = total de atributos ***faça***
      - ***Executa*** a função adaptativa A_k:

        $+ [(*r_0, a_i,$

        $(f_1(a_i), [A_{i+1}](*r_1, ?)),$

        $(f_2(a_i), [A_{i+1}](*r_2, ?)), \ldots, (f_v(a_i), [A_{i+1}](*r_v, ?)))]$

      - ***Consulta*** o valor do atributo a_k do exemplo corrente Y_z e atualiza o próximo nó a ser processado correspondente ao valor obtido para a_k;
    - ***fim para***
  - ***Retorna*** todas as sub-árvores de *S* concatenadas;

- **Saída:**

    - Árvore de decisão adaptativa;

4.4 Conclusões

O funcionamento do algoritmo Ad-DT, além da utilização da tecnologia adaptativa como ponto principal, também utiliza o conceito de não-determinismo para o aprendizado e classificação de conjuntos de dados. Futuramente, para que a construção do algoritmo possa ser completamente finalizada, serão definidas e especificadas algumas informações ausentes no presente projeto. A maioria dessas informações circunvizinham os seguintes pontos:

- Tratamento de valores ausentes, desconhecidos ou inconscientes;
- Situações em que a utilização do ganho de informação é mais apropriada;
- Medidas a serem tomadas quando o algoritmo não for capaz de classificar determinada instância do conjunto de treinamento;
- Melhoria na capacidade de generalização do algoritmo.

Após esclarecidos os pontos citados acima, espera-se que o algoritmo seja plenamente habilitado a gerar uma árvore de decisão adaptativa. Além disto, a árvore deve ser capaz de classificar corretamente quaisquer exemplos de treinamento, se não de maneira mais eficiente, comparável aos algoritmos de indução de árvores de decisão adaptativas já existentes.

5 CRONOGRAMA

Atividades	TCC 1						TCC 2			
	Mar	Abr	Mai	Jun	Jul	Ago	Set	Out	Nov	Dez
1. Revisão bibliográfica										
2. Escrita da proposta										
3. Escrita do projeto										
4. Elaboração do algoritmo										
5. Implementação do algoritmo e integração com a ferramenta										
6. Realização dos testes e comparação dos resultados										
7. Escrita da Monografia de TCC										
8. Elaboração da apresentação final e defesa.										

Representa as atividades em andamento.

Representa as atividades já realizadas.

Representa as tarefas ainda não realizadas.

6 CONSIDERAÇÕES FINAIS

A estrutura das árvores de decisão é hoje um dos métodos mais utilizados para a resolução de problemas de tomada de decisão, classificação e aprendizagem de máquina. Utilizando conceitos advindos da tecnologia adaptativa, as árvores de decisão podem se tornar ferramentas ainda mais eficazes para a inferência indutiva, podendo também serem utilizadas para a solução de problemas de aprendizagem incremental. O projeto propôs um algoritmo de indução de árvores de decisão adaptativas denominado Ad-DT. O Ad-DT, baseado nos conceitos e funcionamento do primeiro algoritmo de indução de árvores adaptativas de nome Adaptree, deve ser capaz de gerar, a partir de um conjunto de dados de treinamento, uma árvore de decisão como saída. Tal árvore deve possuir a capacidade de classificar, corretamente, determinada instância de dados. O presente projeto também descreveu uma série de conceitos a respeito da aprendizagem de máquina, tecnologia adaptativa e indução de árvores de decisão, sendo estes fundamentais para a compreensão do funcionamento, estrutura e formulação do algoritmo proposto.

Devido ao fato da adaptatividade ser um conceito relativamente novo e bastante amplo, encontrou-se certa dificuldade em sua compreensão como um todo, bem como suas implicações no meio em que é utilizada. Contudo, o uso da tecnologia adaptativa em conjunto com técnicas de aprendizagem de máquina, pode transformar completamente o cenário da Inteligência Artificial, ocasionando métodos de tomada de decisão e classificação mais rápidos e eficientes, inclusive os que utilizam as árvores de decisão. Após o desenvolvimento completo do algoritmo, pretende-se distribuí-lo de maneira *open source,* uma vez que os algoritmos existentes para essa finalidade não o fazem. Desse modo, espera-se que o algoritmo possa auxiliar a construção de projetos futuros nessa área de conhecimento, além de ser utilizado em tarefas de aprendizado incremental, classificação e tomada de decisão de maneira eficiente.

REFERÊNCIAS

ALPAYDIN, Ethem. **Introduction to Machine Learning**. 2ª Edição. MIT Press, 2010. ISBN 978-0-262-01243-0.

ALFENAS, D. A.; BARRETTO, M.R.P.. Adaptatividade em robôs sociáveis, uma proposta de um gerenciador de diálogos. In: Sexto Workshop de Tecnologia Adaptativa - WTA 2012. EPUSP, 2012.

BASSETO, Bruno A. Um sistema de composição musical automatizada, baseado em gramáticas sensíveis ao contexto, implementado com formalismos adaptativos. Dissertação de Mestrado, Escola Politécnica da USP, São Paulo, 2000.

CATAE, Fabrício S.; ROCHA, Ricardo L. A. Introdução a Árvores de Decisão Adaptativas. In: WORKSHOP DE TECNOLOGIA ADAPTATIVA, 5. São Paulo. **Anais eletrônicos…** São Paulo: 2011. Disponível em: <http://lta.poli.usp.br/lta/publicacoes/artigos/2011/catae-e-rocha-2011-introducao-a-arvores-de-decisao-adaptativas/view>. Acesso em: 01 abr. 2011.

CEREDA, Paulo R. M.; NETO, João J. Utilizando linguagens de programação orientadas a objetos para codificar programas adaptativos. Memórias do IX Workshop de Tecnologia Adaptativa - WTA 2015. EPUSP, São Paulo, ISBN: 978-85-86686-82-5, pp. 2-9. Janeiro, 2015.

CORMEN, Thomas H.; LEISERSON, Charles E.; RIVEST, Ronald L.; STEIN, Clifford. **Introduction to Algorithms.** 3ª Edição. MIT Press, 2009. ISBN 978-0-262-03384-8.

DUDA, R. O.; HART, P. E. Pattern Classi cation and Scene Analysis [S.l.]: John Wiley Sons Inc, 1973. Hardcover.

MITCHELL, Tom M. **Machine Learning.** 1ª Edição. McGraw-Hill, 1997. ISBN: 0070428077.

NETO, J. J.: **Solving complex problems with Adaptive Automata**. Lecture Notes in Computer Science. S. Yu, A. Paun (Eds.): Implementation and Application of Automata 5th International Conference, CIAA 2000, Vol.2088, London, Canada,

Springer-Verlag, pp.340, 2000.

NETO, João J. Um levantamento da pesquisa em técnicas adaptativas na epusp. **Sistemas e Computação,** Salvador, v.1, n.1, p. 23-47, jan./jun. 2011.

NETO, João. J. Adaptive Automata for Context-Sensitive Languages. SIGPLAN NOTICES, Vol. 29, n. 9, pp. 115-124, September, 1994

PISTORI, Hemerson; NETO, João J. **Adaptree – Proposta de um Algoritmo para Indução de Árvores de Decisão Baseado em Técnicas Adaptativas**. 2002. Anais Conferência Latino Americana de Informática - CLEI. Montevideo, Uruguai, 2002.

PISTORI, Hemerson. **Tecnologia Adaptativa Em Engenharia De Computação: Estado Da Arte e Aplicações.** 2003. 174p. Tese - Escola Politécnica da Universidade de São Paulo, São Paulo, 2003.

RESENDE, Solange O. **Sistemas Inteligentes: fundamentos e aplicações.** Barueri, São Paulo: Manole, 2005. ISBN: 85-204-1683-7.

ROCHA, Ricardo. L. A.; NETO, João. J. Autômato adaptativo, limites e complexidade em comparação com máquina de Turing. In: Proceedings of the second Congress of Logic Applied to Technology - LAPTEC 2000. São Paulo: Faculdade SENAC de Ciências Exatas e Tecnologia, p. 33-48, 2001.

SILVA, Marcelino P. S. Mineração de Dados - Conceitos, Aplicações e Experimentos com Weka. 2004. Belo Horizonte, UFMG. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/erirjes/2004/004.pdf>

SILVA, A. M. ;ROCHA, R. L. A.; NETO, J.J.. Análise semântica de sentimentos utilizando árvores de decisão adaptativas. Em: Memórias do X Workshop de Tecnologia Adaptativa - WTA 2016. EPUSP, São Paulo. ISBN: 978-85-86686-86-3, pp. 37-45. 28 e 29 de Janeiro, 2016

STANGE, Renata L. **Adaptatividade em aprendizagem de máquina: conceitos e estudo de caso**. 2011. 98p. Dissertação - Escola Politécnica da Universidade de São Paulo, São Paulo, 2011.

TCHEMRA, Angela H. Tabela de Decisão Adaptativa na Tomada de Decisão Multicritério. Tese de Doutorado, EPUSP, São Paulo, 2009.

UTGOFF, P. E.: *Incremental Induction of Decision Trees*. Machine Learning, Machine Learning, 4, 161-186, 1989.

YOSHIDA, Murilo L. **Aprendizado supervisionado incremental de Redes Bayesianas para mineração de dados.** 2007. 132p. Dissertação - São Carlos : UFSCar, 2007.