JONES GRANATYR

DESCOBERTA DE REGRAS DE CLASSIFICAÇÃO UTILIZANDO ANÁLISE FORMAL DE CONCEITOS

Dissertação de Mestrado apresentado ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná como requisito para obtenção do título de mestre em Informática.

Curitiba

Maio/2011

JONES GRANATYR

DESCOBERTA DE REGRAS DE CLASSIFICAÇÃO UTILIZANDO ANÁLISE FORMAL DE CONCEITOS

Dissertação de Mestrado apresentado ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná como requisito para obtenção do título de mestre em Informática.

Área de Concentração: Agentes de Software

Orientador: Prof. Dr. Edson Emílio Scalabrin

Curitiba

Maio/2011

Conteúdo

Lista de Figuras

Lista de Tabelas

Resumo

Este trabalho apresenta um estudo sobre a descoberta de conhecimentos para ajudar um maquinista a conduzir um trem. Tal objetivo passa pela aplicação de técnicas de aprendizagem de máquina e descoberta de conhecimento a partir de conjuntos de dados de viagens de trens. Duas técnicas foram exploradas para a obtenção de classificadores, a saber: indução de regras e análise formal de conceitos. A segunda baseia-se em modelos matemáticos formais para construir suas hierarquias para mineração, sendo fundamentada nos reticulados da teoria dos conjuntos de Georg Cantor. O principal resultado obtido foi um estudo comparativo entre estas duas abordagens. O problema de escalabilidade em mineração de dados foi tratado por meio do uso do método BAGGING. As regras descobertas foram testadas de forma objetiva. O processo consistiu, de um lado, aplicar tais regras para selecionar ações em um simulador de condução de trens, e de outro lado, quantificar a similaridade entre o desempenho do simulador e do maquinista ser humano. A similaridade ficou em torno de 70%.

Palavras-chave: classificação, condução de trens, mineração de dados, análise formal de conceitos.

Abstract

This work presents a study about knowledge discovery to help a machinist to drive a train. It was used a database containing the data of train rides to get this goal. Two techniques were used to obtain the classifiers: rule induction and formal concept analysis. The second one is based on formal mathematical models and on the lattice set theory to build its hierarchy. The main result was a comparative study between these two approaches. It was used the BAGGING method do solve the scalability problem and the rules discovered were tested objectively. The process consisted about applying the rules to select actions in a train driving simulator and quantify the similarity between the performance of the simulator and the human machinist. The similarity was around 70%.

Keywords: classification, drive trains, data mining, formal concept analysis.

1 Introdução

Este trabalho enquadra-se no contexto do Projeto PAI-L (Piloto Automático Inteligente para Locomotivas). Em linhas gerais, o objetivo do mesmo é o desenvolvimento de um software inteligente para condução de locomotivas de carga. O PAI-L é um projeto em desenvolvimento no LAS (Laboratório de Agentes de Software) da PUCPR (Pontifícia Universidade Católica do Paraná), em parceria com empresas que atuam no setor e financiado pela FINEP (Financiadora de Estudos e Projetos). O projeto objeto desta dissertação é um subprojeto do PAI-L, baseado na dissertação de Borges (2009), a qual teve como objetivo "à utilização de técnicas de aprendizagem de máquina para a descoberta de padrões relevantes de condução segura e econômica de trens a partir de bases de dados que incluem os perfis das vias, as características dos trens e os históricos das viagens (dados lidos dos equipamentos de medidas: pressão de freio, velocidade, posição, etc.)".

Por tratar-se de um trabalho de pesquisa de descoberta de padrões, foi utilizada a aplicação de algoritmos de aprendizagem supervisionada (ex: C4.5 e técnicas de análise formal de conceitos), e de um método de aprendizagem baseado na combinação de classificadores (ex: BAGGING).

O presente trabalho tem seu foco principal na aplicação do algoritmo *Find Class* da teoria de análise formal de conceitos, sendo que esta técnica é baseada na teoria dos conjuntos de George Cantor, consistindo principalmente nos reticulados para a construção de suas hierarquias. Optou-se pela utilização desta técnica devido ao fato de que ela baseia-se em um modelo formal promissor e adequado à mineração de dados; e também, pode ser uma alternativa à descoberta de regras por meio de algoritmos de aprendizagem supervisionada tradicionais (ex: C4.5).

A utilização de técnicas de amostragem foi necessária devido à complexidade dos dados em termos de número de exemplos, atributos e valores de cada atributo. O interesse no uso do C4.5, análise formal de conceitos (*Find Class)* e BAGGING portou essencialmente na geração de regras de fácil compreensão. Como em Borges (2009), a escolha das boas regras de condução de trens foi, em um primeiro momento, realizada pelo próprio algoritmo

de aprendizagem de máquina por meio da taxa de erro. Esta última serviu como parâmetro de filtragem de regras aplicáveis à condução dos trens.

Foram utilizados quatro artifícios para a validação dos padrões descobertos, ou seja, (i) validação cruzada; (ii) cálculo do cosseno; (iii) Teste de Friedman; e (iv) teste em um simulador de viagens de trens, também usado em Borges (2009). Tal simulador utiliza o cálculo do cosseno para comparar os pontos de aceleração aplicados pelo maquinista com os pontos de aceleração resultantes da aplicação dos algoritmos.

Os seguintes fatores foram considerados para análise dos resultados: a taxa de acerto, a complexidade das regras geradas e a taxa de aplicabilidade das regras.

1.1 Motivação

Em Borges (2009), constatou-se que a partir de uma análise cuidadosa das regras obtidas em um processo de descoberta de conhecimentos, podem-se elaborar diretrizes de condução potencialmente aplicáveis; testadas em um ambiente computacional simulado. Nesta linha, a motivação é aplicar técnicas de análise formal de conceitos [Carpineto 2004] para contribuir com novos conhecimentos para a definição de políticas de condução.

1.2 Objetivos

O objetivo geral deste trabalho é comparar a eficiência de métodos clássicos de obtenção de regras de classificação com métodos baseados em análise formal de conceitos aplicados em uma base de dados que possui registros sobre viagens de trens. Tais dados são coletados por meio de diferentes sensores instalados em um trem, os quais foram estudados em detalhes por Borges (2009). A abordagem para obter as regras de classificação foi análise formal de conceitos.

Os objetivos específicos são:

- Extrair regras úteis à elaboração de políticas de ações à condução de trens de cargas;
- Comparar a eficiência de métodos clássicos de obtenção de regras de classificação com métodos baseados em análise formal de conceitos;
- Avaliar à aplicabilidade potencial das regras descobertas vis-à-vis a elaboração de uma política realista de ações.

A base de dados usada neste trabalho já foi previamente formatada e enriquecida por Borges (2009),

1.3 Hipótese

Em Borges (2009), a aplicação de algoritmos de aprendizagem supervisionada (ex: C4.5) geraram soluções eficazes para o problema de geração de regras de condução de trens. Espera-se que com as características matemáticas das técnicas de análise formal de conceitos a obtenção de regras de condução de trens seja feita de forma rápida e consistente; e também, que tais regras possam ser testadas em um simulador que imite a condução de trens realizada por um maquinista ser humano.

1.4 Contribuições

As contribuições científicas do presente trabalho são: (i) a obtenção de regras de condução de locomotivas de forma rápida e eficaz; e (ii) a validação da aplicabilidade potencial das regras descobertas vis-à-vis à elaboração de políticas de ações potencialmente realistas, usando para tal o simulador desenvolvido em [BORGES 2009].

2 DESCOBERTA DE CONHECIMENTOS

A descoberta automatizada de conhecimentos, a partir de base de dados, é uma necessidade imperativa vis-à-vis a obtenção de padrões de comportamento dos dados para apoiar processos decisórios em um determinado domínio. Nestes termos, serão examinados alguns métodos de aprendizagem de máquina simbólica, técnicas de validação e análise do desempenho de classificadores. Os temas abordados são:

- aprendizagem de máquina simbólica realizada por meio de algoritmos de indução de regras (ex: C4.5), de análise formal de conceitos e de combinação de classificadores (ex: BAGGING);
- avaliação de classificadores utilizando validação cruzada;
- avaliação dos conhecimentos aplicados na forma de um simulador de condução, que deve imitar o comportamento de um maquinista ser humano [BORGES 2009].

Os classificadores serão testados no simulador de condução de trens para auxiliar na definição das políticas de ações. Lembramos que o principal objetivo deste trabalho é confrontar os resultados gerados por um método clássico de geração de regras por indução e por método formal de análise de conceitos.

2.1 Processo de Descoberta do Conhecimento

O processo de descoberta de conhecimento visa extrair conhecimentos de maneira automática e útil a partir de dados sobre certo domínio de problema [FAYYAD, 1996]. Geralmente são utilizadas etapas como: selecionar, pré-processar, transformar, minerar os dados a partir de um conjunto de fatos e interpretar os padrões extraídos. A Figura 1 ilustra as etapas do processo de descoberta de conhecimento.

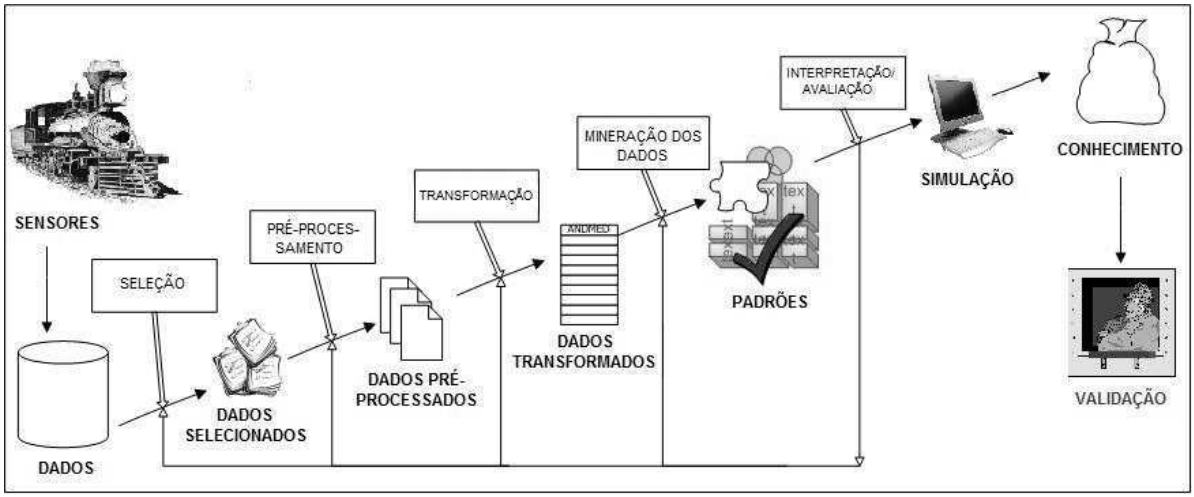

Figura 1. Etapas do processo de Descoberta do Conhecimento [FAYYAD, 1996]

Em nossos experimentos, não foram abordadas todas as suas etapas, visto que se fez uso de dados já preparados no contexto da dissertação de mestrado de Borges (2009). Os registros da base de dados foram obtidos por meio de sensores instalados em locomotivas.

O pré-processamento é dividido em [FAYYAD, et al., 1996]: remover ruídos do conjunto de dados, decidir a forma de tratamento para dados faltantes, selecionar os atributos de modo a reduzir a dimensionalidade dos dados e incluir novas características úteis na descrição dos dados. A transformação responde pela consolidação dos dados vis-à-vis à etapa de mineração. Uma das ações possíveis é normalizar os dados, onde são criadas escalas para os dados com pequenos intervalos, reduzindo assim a discrepância entre os valores [HAN, et al., 2006]. A remoção de ruídos é importante para tornar mais fácil a tarefa de identificar padrões que representem comportamentos nos dados [FAYYAD, et al., 1996]. Um ruído é um erro ou variância aleatória de uma variável cujos valores são conhecidos [HAN, et al., 2006], podendo ser gerado por erro na leitura dos dados ou simplesmente erro na entrada manual dos dados. Por fim, a seleção de atributos diz respeito a localizar o melhor subconjunto de dados a partir de um determinado critério, sendo vista como um processo de filtragem dos dados para assim remover atributos irrelevantes da base de dados e consequentemente aumentar o desempenho do classificador.

2.2 Descoberta Automática de Conhecimentos

A descoberta automatizada de conhecimentos busca encontrar padrões válidos e interessantes a partir do conjunto de dados de um determinado domínio ([CIOS, et al., 2007],[HAN, et al., 2006]). Estes padrões são interessantes se forem de fácil compreensão, potencialmente utilizáveis e novos; e válidos se forem aplicáveis para classificar novos dados/eventos.

O interesse é aplicar os algoritmos de descoberta automática de conhecimentos em bases de dados históricos da área de condução de trens, visando à descoberta de regras de indicação de qual ponto de aceleração deve ser empregado pelo maquinista. O uso da mineração de dados ([FAYYAD, 1996], [FAYYAD et al., 1996]) tem se mostrado eficiente em diversas áreas do conhecimento humano, como na química [KALOS, et al., 2005], classificação de músicas [CHEN, et al., 2009], extração de regras de condução de trens de carga [BORGES, 2009], extração de regras de alteração pós-cirúrgica [BRANCO 2010], entre outras.

Os métodos de aprendizagem de máquina estudados e experimentados neste trabalho geram classificadores simbólicos por meio do método de indução e análise formal de conceitos.

2.3 Aprendizagem de Máquina Indutiva

A aprendizagem de máquina é uma área da inteligência artificial com o objetivo de desenvolver algoritmos e técnicas que permitam ao computador aprender, ou seja, adquirir conhecimento de forma automática. Os sistemas de aprendizagem de máquina tomam decisões baseados em experiências acumuladas por meio de soluções bem sucedidas de eventos anteriores. Dado um conjunto de exemplos *T* rotulados, a aprendizagem de máquina pode ser vista como a inferência automática de conceitos a partir de *T* [MITCHELL, 1997].

Existem algumas diferenças entre a aprendizagem de máquina e a descoberta automática de conhecimentos a partir de base de dados. A descoberta automática de conhecimentos geralmente opera com dados em grande volume e baseados em situações reais [PRATI, 2006]. Para facilitar o aprendizado, no início, os algoritmos de aprendizagem de máquina trabalhavam somente sobre bases de dados bastante pequenas; após, como a descoberta automática de conhecimentos transformou-se em somente uma parte de um processo maior (que vai desde a preparação dos dados até sua utilização na geração de

conhecimentos), foi permitida que a aprendizagem de máquina trabalhasse com conjuntos de dados maiores e mais complexos.

A aplicação prática da aprendizagem de máquina inclui diferentes formas de processamento de aprendizado, de linguagem de descrição e paradigmas; como aprendizagem de máquina: simbólica, que aprende construindo representações simbólicas de um conceito (ex.: árvore de decisão, regras de produção); estática, que representa o aprendizado através de modelos estatísticos (ex.: probabilístico bayesiano); baseada em exemplos, classifica exemplos nunca vistos como similares conhecidos, (ex.: raciocínio baseado em casos); conexionista: aprendizado através da interconexão de unidades simples, (ex.: redes neurais artificiais); evolutiva: onde os elementos fracos são descartados e os fortes predominam.

2.3.1 Aprendizagem Simbólica de Máquina

A aprendizagem simbólica de máquina diz respeito à automatização de um processo de aprendizagem, sendo parte integrante de um sistema de previsão. Tal tipo de sistema tem o objetivo de tomar decisões e aprender através da construção de representações simbólicas de um conceito por meio da análise de exemplos. Comumente, estes exemplos estão armazenados em bases de dados sobre determinado assunto e geralmente possuem um grande número de registros. As representações geradas através deste tipo de sistema, em geral, tomam a forma de árvores de decisão ou regras de produção.

2.3.2 Aprendizagem Indutiva

A aprendizagem indutiva tem a característica de extrair padrões através de um conjunto de exemplos, deduzindo o conhecimento pela observação do seu ambiente; sendo capaz de prever o resultado de ocorrências futuras. A indução também pode ser caracterizada como um raciocínio que parte de um problema específico e o generaliza [MALOOF, et al., 2000].

Existem dois modos principais na aprendizagem indutiva, que são: aprendizagem supervisionada e aprendizagem não supervisionada. O primeiro ocorre quando os conjuntos de exemplos são repassados ao sistema já com suas classes definidas, ou seja, os conjuntos ainda não rotulados devem ser classificados. A segunda estratégia ocorre quando o conjunto de exemplos não está rotulado, ou seja, a busca de cada classe ocorre pelo reconhecimento dos padrões examinando os exemplos.

2.4 Métodos de Classificação

Um método de classificação é um processo de encontrar dependências entre dados com o objetivo de descobrir conhecimentos relevantes a respeito de um determinado problema [VIMIEIRO, 2007]. A obtenção de regras de classificação são relacionamentos entre atributos, em que cada registro contém um conjunto de atributos e um deles é a classe, no qual, o objetivo é encontrar um modelo para o atributo classe como uma função dos valores dos outros atributos. A partir de um conjunto de objetos com suas respectivas classes, deseja-se classificar novos objetos, dos quais não se sabe à que classe pertence. O algoritmo C4.5 é um exemplo de extrator de padrões a partir de bases de dados, o qual será resumidamente analisado a seguir.

2.4.1 Algoritmo C4.5

Este algoritmo realiza a tarefa de aprender de forma supervisionada [MITCHELL, 1997]. Isto é, ele gera, a partir de um conjunto de exemplos previamente rotulados, um classificador no formato de árvore de decisão ou no formato de um conjunto ordenado de regras de produção. Tal classificador tem como objetivo representar os conhecimentos sobre determinado problema ([MITCHELL, 1997], [HAN & KAMBER, 2006]). A aprendizagem baseada em árvore de decisão desperta o interesse por ser robusta quando existem ruídos nos dados [QUINLAN, 1996]. A robusteza do algoritmo caracteriza-se também por operar tanto sobre valores discretos, quanto sobre valores contínuos. Deve-se observar que os valores contínuos precisam ser discretizados para possibilitar a geração de regras de classificação.

2.5 Aprendizagem de Máquina usando AFC

A análise formal de conceitos é uma teoria criada por Rudolf Wille (1982), a qual tem como principal objetivo a identificação de estruturas conceituais de relacionamento entre dados. Esta técnica é aplicada atualmente em várias áreas, tendo importantes contribuições nos setores da psicologia, sociologia, antropologia, medicina, biologia, linguística, ciência da computação, matemática e engenharia industrial.

Análise formal de conceitos inspira-se na teoria dos conjuntos de Georg Cantor, sendo baseado principalmente nos conjuntos ordenados e na teoria dos reticulados para a construção de suas hierarquias de conceitos.

A seguir serão descritos os principais fundamentos matemáticos que estão diretamente envolvidos no domínio desta técnica.

2.5.1 Fundamentos Matemáticos

Como supracitado, os fundamentos matemáticos envolvidos na teoria de AFC estão ligados diretamente aos conjuntos ordenados e aos reticulados. A seguir, tais conceitos serão explanados.

Notações básicas sobre relações de ordem

O primeiro tópico que diz respeito à teoria da AFC envolve os conjuntos ordenados. Como exemplo, a representação de uma ordem pode ser dada por meio do conjunto $P = \{a, b, c, d, e\}$, sendo que $a < c$, $a < d$, $b < c$ e $b < d$. Este conjunto pode ser representado graficamente pelo diagrama mostrado na Figura 2.

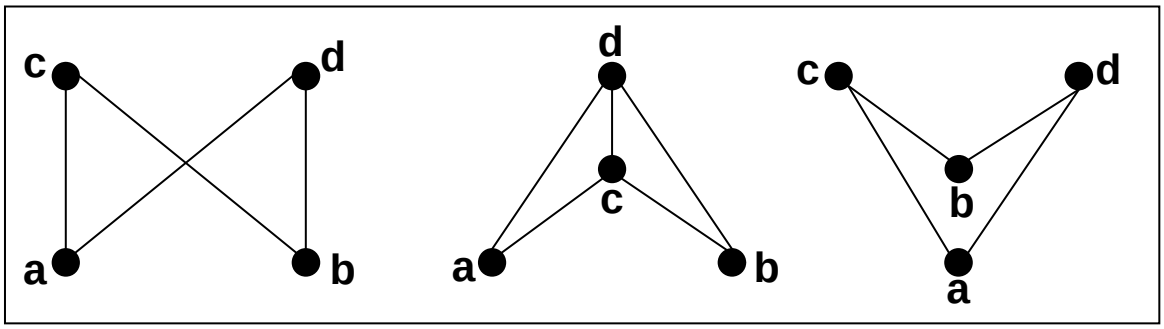

Figura 2. Relações de ordem em um conjunto [ABE, 1996].

Na Figura 2 pode-se observar por meio do diagrama a ordenação que existe no conjunto *P*, sendo que, por exemplo, $a < c$ nos três diagramas porque em todos os três, o elemento *a* está abaixo do elemento *c*.

Na representação gráfica de um conjunto ordenado, existem os elementos máximo, mínimo, maximais e minimais, os quais são mostrados por meio dos diagramas representados na Figura 3.

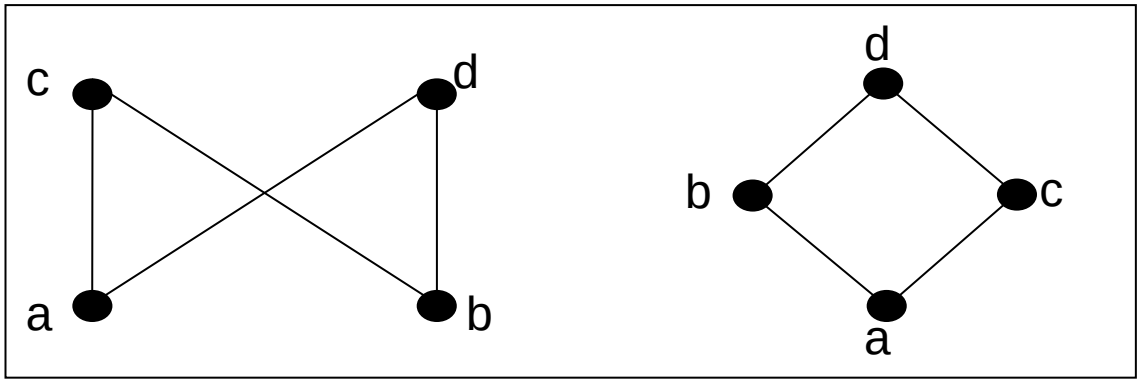

Figura 3. Máximo, mínimo, maximais e minimais [ABE, 1996].

Pode-se observar, na Figura 3, que o diagrama à esquerda não possui elemento máximo tão pouco elemento mínimo, pois analisando a relação de ordem, percebe-se que não existe item exclusivo que está no topo ou na base do diagrama. Por outro lado, existem os elementos maximais e minimais, os quais se encontram no topo e base do diagrama, respectivamente. Em suma, podem-se resumir os elementos do primeiro diagrama (esquerda) como:

Máximo: ∉ Mínimo: ∉
Maximais: {c, d} Minimais: {a, b}

Para o segundo diagrama (direita) da Figura 3, podem-se resumir os elementos como:

Máximo: d Mínimo: a
Maximais: {d} Minimais: {a}

Além disso, existem também em uma representação gráfica de um conjunto ordenado os elementos majorantes, minorantes, supremo e ínfimo, os quais são mostrados por meio do diagrama representado na Figura 4. Nesta figura, pode-se observar a presença de um subconjunto *X* pertencente ao conjunto principal. Um elemento qualquer deste subconjunto é chamado de majorante se este mesmo elemento sucede todo elemento de *X*. Portanto, os majorantes são *a, b* e *c.* Se um majorante do conjunto *X* precede qualquer outro majorante de *X*, então ele é chamado o supremo de *X*.

Por outro lado, um elemento qualquer deste subconjunto é chamado de minorante se este mesmo elemento precede todo elemento de *X*. Portanto, o minorante é *f.* Se um minorante do conjunto *X* precede qualquer outro minorante de *X*, então ele é chamado o ínfimo de *X* [LIPSCHULTZ & LIPSON, 1997].

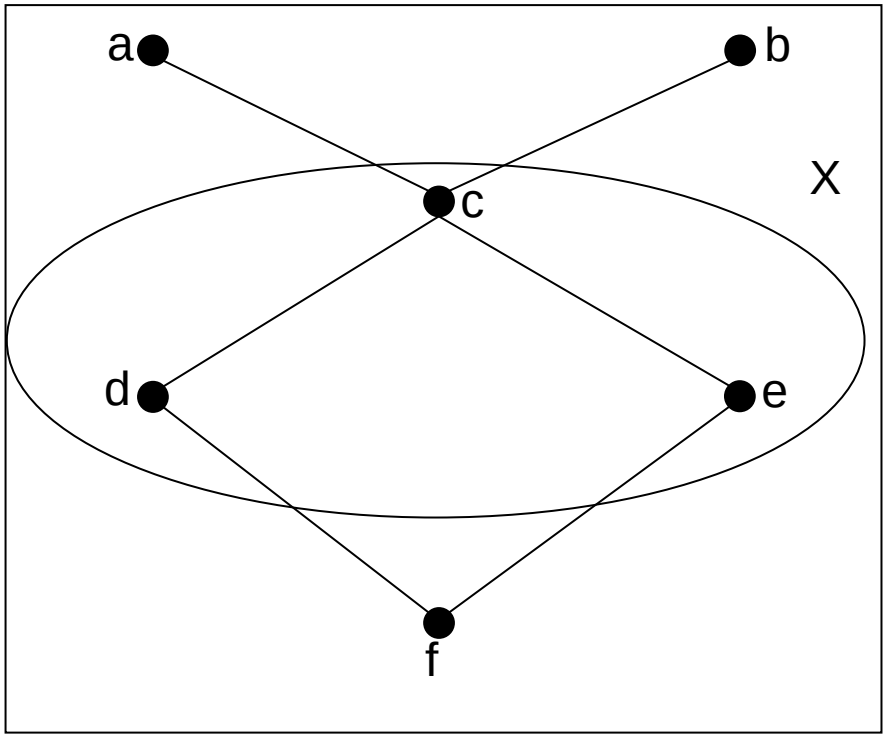

Figura 4. Majorantes, minorantes, supremo e ínfimo [ABE, 1996].

Reticulados

Um reticulado é basicamente um conjunto ordenado que possua algumas das características citadas nos tópicos anteriores. O digrama de linhas da Figura 5 mostra um exemplo de reticulado.

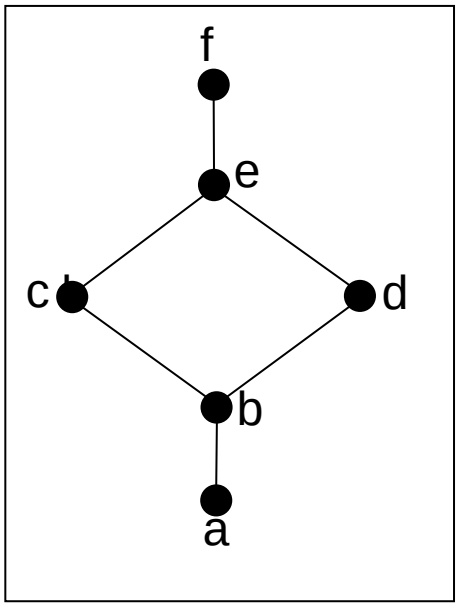

Figura 5. Reticulado [ABE, 1996].

Seja a Figura 5 um conjunto ordenado *P*. Tal conjunto é considerado um reticulado, se para cada par ordenado de elementos *x* e *y* em *P*, o supremo $x \vee y$ e o ínfimo $x \wedge y$ existem. Todo reticulado completo possui um elemento no topo (*f*) e um elemento na base (*a*) [ABE, 1996].

2.5.2 Hierarquia de Conceitos

Para a construção da hierarquia de conceitos por meio dos reticulados, AFC fundamenta-se em três componentes básicos, a saber: contextos formais, conceitos formais e reticulados conceituais. A seguir, esses três elementos serão explanados.

Contexto Formal

Um contexto é uma tripla *(G, M, I)* que consiste em dois conjuntos *G* e *M* e uma relação *I* entre os conjuntos *G* e *M*. Os elementos do conjunto *G* são chamados de objetos, enquanto que os elementos do conjunto *M* são chamados de atributos. Por outro lado, a relação *I* é chamada de relação de incidência de todo o contexto. Para representar tal relação, pode-se escrever *gIm*, indicando que o objeto *g* possui o atributo *m* dentro da relação *I* [CARPINETO & ROMANO, 2004].

A Tabela 1 mostra tal fato por meio da descrição de características de alguns animais, as quais são indicadas por marcações em forma de *X* na tabela de contexto. Se uma célula está vazia, indica que o animal correspondente não possui o atributo em questão. Neste exemplo, o nome dos animais (linhas) são os objetos (*Leão*, *Canário*, *Águia*, *Lebre* e *Avestruz*), as características (colunas) de cada animal (*Predador*, *Voa*, *Ave* e *Mamífero*) são os atributos e as interseções entre as linhas e as colunas indicam se os objetos possuem ou não determinados atributos; sendo a relação de incidência do contexto.

Tabela 1. Contexto Formal (adaptado de [WOLFF, 1993]).

Animais	Predador	Voa	Ave	Mamífero
Leão	X			X
Canário		X	X	
Águia	X	X	X	
Lebre				X
Avestruz			X	

Conceito Formal

Na Tabela 1, observando-se os atributos do objeto *Canário*, verifica-se que o objeto *Águia* também possui os mesmos atributos, ou seja, *Voa* e *Ave*. Desta maneira, obtém-se o conjunto *A* que consiste nesses dois animais. Além disso, este mesmo conjunto *A* é conectado ao conjunto *B*, o qual é composto pelos atributos *Voa* e *Ave*. Por meio desta relação, pode-se concluir que *A* é o conjunto de todos os objetos que tenham seus atributos em *B*, e *B* é o conjunto de todos os atributos que são válidos para os objetos de *A*. Cada par *(A, B)* é chamado de conceito formal dentro de todo o contexto, sendo ainda que o conjunto *A* é chamado de *extensão*, enquanto que o conjunto *B* é chamado de *intenção*. Em suma, para a determinação dos conceitos é necessário que determinados objetos estejam relacionados a certos atributos, e também, que determinados atributos estejam relacionados a certos objetos.

Reticulado Conceitual

Um reticulado conceitual é representado graficamente por um diagrama de linhas, o qual apresenta os conceitos formais juntamente com sua *extensão* e *intenção*. Desta forma, o diagrama da Figura 6 mostra a representação da hierarquia conceitual de todos os conceitos da tabela de animais.

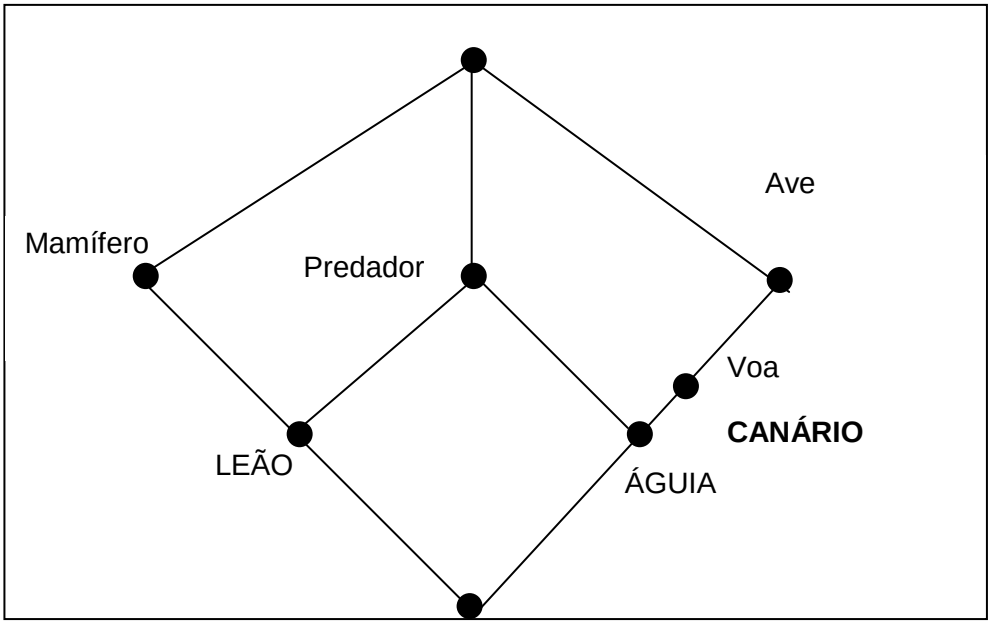

Figura 6. Reticulado Conceitual (adaptado de [WOLFF, 1993]).

O diagrama da Figura 6 consiste em alguns componentes, bem como círculos, linhas e o nome dos objetos e dos atributos do contexto. O círculo representa os conceitos e a sua informação pode ser lida da seguinte forma: um objeto *g* tem um atributo *m* se e somente se existir um caminho do círculo *g* para um círculo *m* [WOLFF, 1993].

De acordo com o diagrama, pode-se notar que os atributos acima do objeto *Canário* são exatamente os mesmos atributos que estão na tabela do contexto. Também se pode notar que o objeto *Canário* possui a *extensão Canário* e *Águia* e a *intenção Voa* e *Ave*. Este fato é representado pelas seguintes funções matemáticas:

$A\uparrow = \{m \in M \mid \forall g \in A\ gIm\}$ $\quad\quad$ $B\downarrow = \{g \in G \mid \forall m \in B\ gIm\}$

A função ↑ retorna o conjunto dos atributos comuns aos objetos de *A*, enquanto que a função ↓ retorna o conjunto de objetos que possui os atributos de *B* em comum [VIMIEIRO & VIEIRA, 2007]. Assim, por meio do reticulado conceitual, podem-se tornar explícitos os relacionamentos entre os atributos, o que possibilita uma base para extração de regras, tanto de associação quanto de classificação.

2.5.3 Transformação de Dados em Reticulados Conceituais

Nesta seção mostrar-se-á de que maneira uma base de dados comum pode ser transformada e convertida na representação de um reticulado conceitual. Para ilustrar, utilizar-se-á os dados da Tabela 2; nesta última, cada coluna representa respectivamente um dos seguintes atributos de uma pessoa: nome, sexo e idade.

Tabela 2. Transformação em reticulado (Adaptado de [WOLFF, 1993]).

Nome	Sexo	Idade
Marcos	M	21
Paula	F	50
Cristiano		66
Janete	F	88
Jéssica	F	17
João	M	
Jorge	M	90
Maria	F	50

Cada registro da Tabela 2 apresenta-se em um contexto univalorado, ou seja, para cada um dos atributos existe apenas um valor possível. Para a correta representação do contexto e a posterior geração do reticulado conceitual, faz-se necessário transformar esta tabela em um contexto multivalorado, obtendo-se a Tabela 3.

Tabela 3. Contexto multivalorado (Adaptado de [WOLFF, 1993]).

	Sexo		Idade				
	M	F	< 18	< 40	<= 65	> 65	>= 80
Marcos	X			X	X		
Paula		X			X		
Cristiano					X		
Janete		X				X	X
Jéssica		X	X	X	X		
João	X						
Jorge	X					X	X
Pedro	X				X		

A Tabela 3 foi transformada em um contexto multivalorado devido ao fato de que os atributos presentes no contexto podem possuir mais de um valor, diferentemente do contexto apresentado na Tabela 2.

Como resultado, tem-se o diagrama da Figura 7 para representar todos os conceitos existentes dentro do contexto.

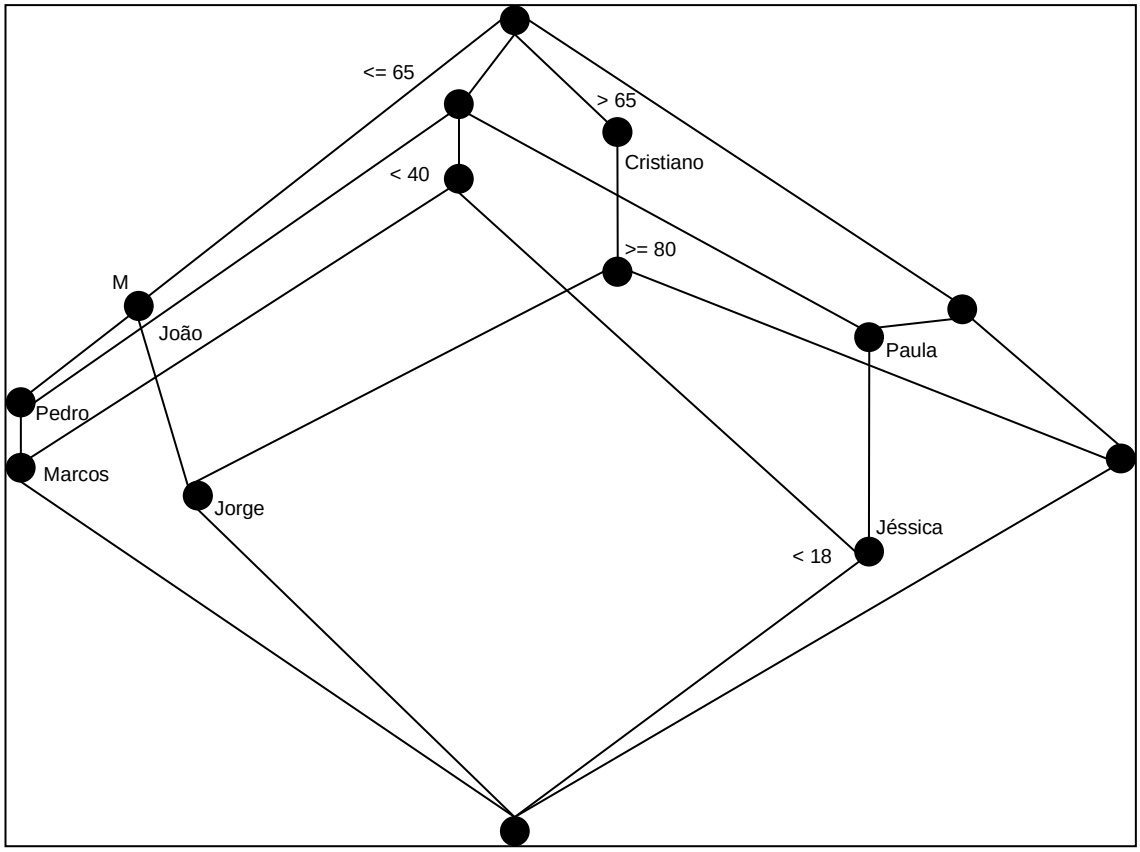

Figura 7. Reticulado conceitual (Adaptado de [WOLFF, 1993]).

O diagrama de linhas mostrado na Figura 7 é obtido por meio da interligação dos conceitos obtidos através do contexto. Como exemplo, analisando o objeto *Pedro*, percebe-se que acima dele, existe um círculo com o atributo *M*, e mais acima outro círculo com o atributo *<= 65*. Desta maneira, pode-se concluir que o objeto *Pedro* é do sexo masculino e possui 65 anos ou menos, conforme tabela do contexto.

2.6 MINERAÇÃO DE REGRAS COM ANÁLISE FORMAL DE CONCEITOS

A seguir são abordados alguns conceitos fundamentais de duas técnicas de mineração de dados, a saber: regras de associação e regras de classificação. Também será mostrado um passo a passo de como gerar regras de classificação de uma base de dados simplificada com dados relativos à condução de trens; a qual é objeto do presente trabalho.

2.6.1 Regras de Associação

Regras de associação são relacionamentos entre atributos em grupos de objetos em uma base de dados. Duas características são consideradas em regras de associação, a saber: o suporte e a confiança.

O suporte revela a probabilidade dos objetos da base de dados possuírem os atributos envolvidos na regra, enquanto que a confiança revela a proporção de objetos que possuem os atributos do antecedente e do consequente. Tanto o suporte quanto a confiança funcionam como uma espécie de filtro para a obtenção das regras e, geralmente, o próprio usuário que os define.

Como exemplo, uma associação $Q \rightarrow R$ é verdadeira se existirem objetos que possuam atributos de ambos os conjuntos *Q* e *R*, e se existem objetos que possuem atributos de *Q* e que podem também possuir atributos de *R*. Neste caso, em um reticulado conceitual o suporte e a confiança podem ser obtidos por meio das seguintes fórmulas:

$$\sup orte(Q \rightarrow R) = \frac{|(Q \cup R)'|}{|G|} \geq \text{suporteminimo}$$

$$confiança(Q \rightarrow R) = \frac{|(Q \cup R)|}{|Q|} \geq \text{confiancaminima}$$

O processo de encontrar o conjunto de regras de associação presentes no contexto é dividido em duas etapas: (i) encontrar os subconjuntos de atributos frequentes e (ii) gerar as regras baseadas no suporte e confiança determinados pelo usuário. Após essas duas etapas, o próximo passo é a extração dos conceitos frequentes do reticulado conceitual. Neste momento, é importante salientar que há diferença entre a extração de conceitos frequentes e a extração de atributos frequentes. Devido à estrutura matemática do reticulado conceitual, a extração utilizando a primeira técnica é mais eficiente que a segunda, pois todos os conceitos já estão presentes na hierarquia do reticulado, possibilitando melhor desempenho e menor quantidade de regras geradas [CARPINETO & ROMANO, 2004].

Para uma melhor compreensão dos procedimentos que compõe a extração de regras de associação é apresentado na Figura 8 o pseudocódigo do algoritmo *Frequent Next Neighbours*.

```
FrequentNextNeighbours
Entrada: Contexto(G, M, I), suporteminimo
Saída: Reticulado (C, E) dos conceitos freqüentes de (G, M, I)

1. C := {(G, G')}
2. E := ∅
3. nivelAtual := {(G, G')}
4. enquanto nivelAtual ≠ ∅
5.    proximoNivel := ∅
6.    para cada (X, Y) ∈ nivelAtual
7.       vizinhosFrequentes := EncontraoVizinhosMaisProximos((X, Y))
8.       para cada (X1, Y1) ∈ vizinhosFrequentes
9.          se (X1, Y1) ∉ C então
10.            C := C ∪ {(X1, Y1)}
11.            proximoNivel := proximoNivel ∪ {(X1, Y1)}
12.         Adiciona nó (X1, Y1) → (X, Y) to E
13.   nivelAtual := proximoNivel

   function EncontraVizinhosMaisProximos((X, Y))
   /* Retorna os vizinhos freqüentes de um conceito */
1. candidatos := ∅
2. candidatosMaximosGerados := ∅
3. para cada m ∈ M\Y
4.    Sup := |{m}' ∪ X|
5.    se Sup ≥ suporteminimo então
6.       X1 := (Y ∪ {m})'
7.       Y1 := X1'
8.       se (X1, Y1) ∉ candidatos então
9.          Adiciona(X1, Y1) to candidatos
10.         contador(X1, Y1) := 1
11.     senão
12.         contador(X1, Y1) := contador(X1, Y1) + 1
13.     se (|Y1| - |Y|) = contador(X1, Y1) então
14.        Adiciona(X1, Y1) to candidatosMaximosGerados
15. retorne candidatosMaximosGerados
```

Figura 8. Algoritmo Frequent Next Neighbours [CARPINETO & ROMANO, 2004].

No algoritmo *Frequent Next Neighbours*, o ponto de início é o elemento topo do reticulado conceitual (dado por *(G, G')*), passando para cada nível um de cada vez, sendo que o próximo nível contém todos os filhos de conceitos presentes no nível atual.

A utilização do parâmetro *suporteminimo* evita a geração de conceitos não frequentes, sendo que primeiramente é feito o teste para saber se determinado conceito encaixa-se no suporte mínimo antes de realizar a geração da regra. Isto é feito buscando a intersecção entre a *extensão* do conceito atual e verificando se sua cardinalidade é maior que o parâmetro *suporteminimo.*

Como resultado, o algoritmo retorna e constrói um reticulado dos conceitos frequentes associados ao contexto em questão.

2.6.2 Regras de Classificação

Em muitos sistemas de aprendizagem indutiva, a busca pelas regras caracteriza-se por meio da utilização de uma heurística, porém, com o reticulado conceitual é possível construir uma espécie de mapa por meio das regras, o que evita o processamento em tentar encaixar os dados de treinamento heuristicamente [CARPINETO & ROMANO, 2004].

A Tabela 4 mostra uma base de dados hipotética. Tal base de dados apresenta as seguintes características: exemplos, atributos e classes, onde *n* é o número de exemplos, *q* é o número de atributos e *m* é o numero de classes, lendo n=14, q= 4 e m=2. Na sequência será mostrado como é feita a geração das regras utilizando análise formal de conceitos, bem como um classificador composto AFC+BAGGING. Este último permite melhorar a taxa de acerto de um classificador por meio da combinação de várias árvores de decisão.

Tabela 4. Conjunto de treinamento previamente ordenado (adaptado de QUINLLAN, 1996)

INSTÂNCIAS	ATRIBUTOS PREVISORES			ATRIBUTO META
ID	TEMPO	VELOCIDADE	ACLIVE	PONTO DE ACELERAÇÃO
01	Ensolarado	Media	Nao	Aumentar
02	Ensolarado	Media	Nao	Manter
03	Ensolarado	Media	Sim	Aumentar
04	Ensolarado	Baixa	Nao	Manter
05	Ensolarado	Baixa	Sim	Manter
06	Nublado	Media	Nao	Aumentar
07	Nublado	Media	Sim	Manter
08	Nublado	Baixa	Sim	Aumentar
08	Nublado	Baixa	Nao	Aumentar
10	Chuvoso	Media	Nao	Manter
11	Chuvoso	Media	Nao	Aumentar
12	Chuvoso	Media	Sim	Manter
13	Chuvoso	Baixa	Sim	Manter
14	Chuvoso	Baixa	Nao	Manter

A Tabela 4 apresenta um banco de dados comum que mostra qual o ponto de aceleração deve ser aplicado pelo maquinista humano e/ou classificador de acordo com os atributos previsores. Esta base de dados será utilizada para mostrar o passo a passo de como gerar as regras de classificação. Para isso, conforme Wolff (1993), a primeira etapa para gerar

corretamente um reticulado conceitual é a transformação do contexto *univalorado* em um contexto *multivalorado* com o intuito de obter um desmembramento dos atributos. A Tabela 5 mostra o resultado desta transformação.

Tabela 5. Conjunto de treinamento previamente ordenado transformado em contexto multivalorado.

INSTÂNCIAS	ATRIBUTOS PREVISORES							ATRIBUTO META	
ID	TEMPO			VELOCIDADE		ACLIVE		PONTO DE ACELERAÇÃO	
	Ensolarado	Nublado	Chuvoso	Média	Baixa	Sim	Não	Aumentar	Manter
01	X			X			X	X	
02	X			X			X		X
03	X			X		X		X	
04	X				X		X		X
05	X				X	X			X
06		X		X			X	X	
07		X		X		X			X
08		X			X	X		X	
09		X			X		X	X	
10			X	X			X		X
11			X	X			X	X	
12			X	X		X			X
13			X		X	X			X
14			X		X		X		X

Pode-se notar na Tabela 5 que o atributo TEMPO foi desmembrado em três novos atributos (*Ensolarado*, *Nublado* e *Chuvoso*), o atributo VELOCIDADE em dois novos atributos (*Média* e *Baixa*), ACLIVE em dois novos atributos (*Sim* e *Não*) e finalmente o PONTO DE ACELERAÇÃO nos atributos *Aumentar* e *Manter*.

Depois de o contexto ser transformado em multivalorado, o segundo passo é a construção dos conceitos. Para isso, será utilizado o algoritmo *Next Closure*. O pseudocódigo do referido algoritmo encontra-se na Figura 9.

```
Next Closure
Entrada: Contexto(G, M, I)
Saída: Conjunto C de todos os conceitos do contexto (G, M, I)

1. C := {(M',M)}
2. conjuntoAtual := maximo{g ∈ G)
3. proximoObjeto := Maximo{g ∈ G)
4. enquanto nivelAtual ≠ G
5.     se não existe g ∈ conjuntoAtual então
8.           C := C U {conjuntoAtual", conjuntoAtual'}
9.           proximoObjeto := Maximo{g ∈ G}
10.          conjuntoAtual := conjuntoAtual"
11.    senão
12.        proximoObjeto := Maximo{g ∈ G tal que g <  maximo(conjuntoAtual)}
14. conjuntoAtual := conjuntoAtual U {proximoConjunto}
15. conjuntoAtual := conjuntoAtual tal que {g ∈ conjuntoAtual tal que
    proximoObjeto < g}
```

Figura 9. Algoritmo Next Closure [CARPINETO & ROMANO, 2004].

Este algoritmo tem a característica de fazer a análise somente de alguns subconjuntos pré-determinados, sendo que uma ordem lexográfica é utilizada. A principal característica do *Next Closure* é que o próximo subconjunto é gerado do subconjunto atual, ou seja, adicionando o máximo objeto de *G* e apagando todos os objetos atuais que são maiores que o objeto recém adicionado. A principal vantagem deste algoritmo é que cada conceito é criado somente uma vez.

A Figura 10 mostra o reticulado conceitual parcial gerado através da Tabela 5 e da aplicação do algoritmo *Next Closure*. É apresentado o reticulado parcial somente do atributo TEMPO, pois o reticulado total; devido ao número de atributos, ficaria muito grande e de difícil visualização e compreensão. O gráfico foi gerado pelo software *Lattice Miner*.

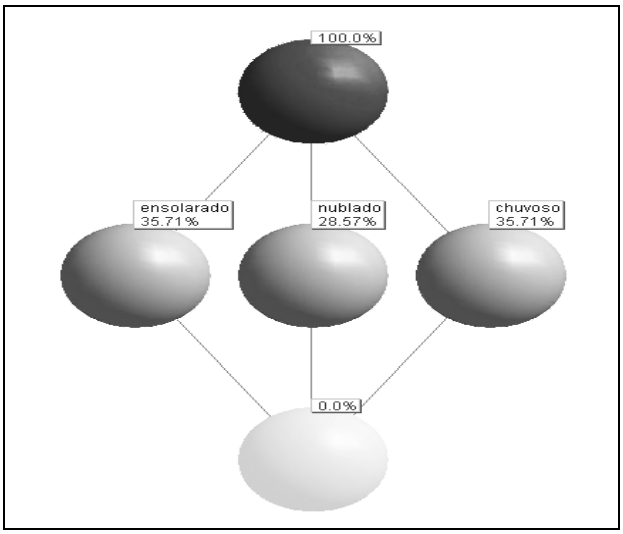

Figura 10. Reticulado parcial do contexto gerado (atributo Tempo)

O terceiro e último passo é a aplicação do algoritmo *Find Class* para efetivamente gerar o conjunto de regras. É apresentado na Figura 11 o pseudocódigo deste algoritmo.

```
Find Class
Entrada:
      um reticulado conceitual L do contexto (G, M, I),
      um conjunto de objetos com funções de classificação C(g) sendo
      que Cg) ∈ [c1, c2] para todo g ∈ G,
      um parâmetro N para tratamento de ruídos,
      um novo objeto x com intenção {x}'
Saída: Classes de objetos x

1. contadorC1 := 0
2. contadorC2 := 0
3. nivelAtual := {(G, G')}
4. enquanto nivelAtual ≠ ∅
5.    proximoNivel := ∅
6.    para cada (X, Y) ∈ nivelAtual
7.       se Y ⊂ {x}' então
8.          se para todo g ∈ Y, C(g) = c1 (com tolerância N) então
9.             contadorC1 := contadorC1 + 1
10.         se para todo g ∈ Y, C(g) = c2 (com tolerância N) então
11.            contadorC2 := contadorC2 + 1
12.         para cada desmarcado(X1, Y1) filho de (X, Y)
13.            Mark(X1, Y1)
14.            proximoNivel := proximoNivel ∪ {(X1, Y1)}
15.   nivelAtual := proximoNivel
16. se contadorC1 > contadorC2 então
17.    retorne c1
18. senão
19.    retorne c2
```

Figura 11. Algoritmo Find Class [CARPINETO & ROMANO, 2004].

No algoritmo *Find Class*, a primeira etapa consiste em construir um reticulado conceitual por meio do conjunto de objetos de treinamento, ignorando o atributo classe. Posteriormente, o objeto atual testado no algoritmo é associado para a classe mais numerosa dentre os conceitos determinados, e após, o teste é realizado para os conceitos que se encontram abaixo do conceito atual examinado.

Este método é eficiente quando implementado em uma abordagem *topdown*, analisando as porções mais relevantes do reticulado conceitual.

A Figura 12 mostra as regras geradas através da aplicação do algoritmo.

```
REGRA 1:
   SE (TEMPO = CHUVOSO) E (ACLIVE = NAO)
   ENTÃO PONTO DE ACELERAÇÃO = AUMENTAR
REGRA 2:
   SE (TEMPO = CHUVOSO) E (ACLIVE = SIM)
   ENTÃO PONTO DE ACELERACAO = MANTER
REGRA 3:
   SE (TEMPO = ENSOLARADO)
   ENTÃO PONTO DE ACELERAÇÃOA = MANTER
REGRA 4:
   SE (TEMPO = NUBLADO) E (ACLIVE = NAO)
   ENTÃO PONTO DE ACELERAÇÃO = AUMENTAR
REGRA 5:
   SE (TEMPO = NUBLADO) E (ACLIVE = SIM) E (VELOCIDADE = BAIXA)
   ENTÃO PONTO DE ACELERAÇÃO = AUMENTAR
REGRA 6:
   SE (TEMPO = NUBLADO) E (ACLIVE = SIM) E (VELOCIDADE = MEDIA)
   ENTÃO PONTO DE ACELERAÇÃO = MANTER
```

Figura 12. Regras geradas pelo algoritmo *Find Class* [CARPINETO & ROMANO, 2004].

Para exemplificar o processo de classificação, as três instâncias da Tabela 6 serão submetidas ao classificador da Figura 12.

Tabela 6. Instâncias a serem classificadas pelo método *Find Class*

Instâncias	ATRIBUTOS			
	PREVISORES			META
	tempo	velocidade	aclive	Ponto de Aceleração
A1	ensolarado	media	não	?
A2	nublado	media	Sim	?
A3	chuvoso	baixa	sim	?

A classificação da instância a_1 resultaria na indicação do valor *aumentar* para o atributo meta *ponto de aceleração*, enquanto que para a instância a_2 resultaria na indicação do valor *manter* para o atributo meta *ponto de aceleração*, e, por final, a instância a_3 resultaria na indicação do valor *manter* para o atributo meta *ponto de aceleração*.

Até aqui, o método apresentado permite a geração de classificadores simbólicos simples a partir de uma base de dados, ou seja, ele não gera vários classificadores para diferentes amostras de uma mesma base de dados.

Teoricamente, a eficiência de classificadores simples poderia melhorar por meio da combinação de diferentes classificadores obtidos a partir de diferentes amostras de uma mesma base de dados. Há dois métodos básicos de combinação de classificadores:

BAGGING e BOOSTING [QUINLAN, 1996]. A combinação de diferentes classificadores visa obter uma melhor taxa de acerto do que a obtida pela aplicação de um único classificador [WITTEN & FRANK, 2000, 2005]. Neste trabalho limitaremos apenas ao método BAGGING.

2.6.3 Método BAGGING

O método BAGGING combina k classificadores a partir de k amostras da base original. As amostras devem ter o mesmo tamanho do conjunto original e para cada uma das k amostras um classificador é obtido [BREIMAN, 1996]. Cabe salientar que estas amostras são sorteadas com reposição e distribuídas de forma uniforme à medida que elas deverão ter o mesmo número de instâncias que o conjunto original.

A Figura 13 mostra o algoritmo adaptado de Breiman, que toma como entrada um conjunto de treinamento T e devolve um conjunto C de classificadores. Cada membro tem igual peso. As linhas cinco e seis correspondem respectivamente a duas chamadas de funções, sendo a primeira para a obtenção de uma amostra A_i, com reposição, de T e a segunda para a obtenção de um classificador a partir da amostra A_i. O processo de amostragem e treinamento repete-se por k vezes. Em outras palavras, cada classificador de C é obtido a partir de uma amostra diferente A_i de T [GRANDVALET, 2004].

```
   Entradas: uma base de dados T com n instâncias
             um número de classificadores k a serem gerados
   Saída:    um conjunto de k classificadores C

1  função f(T, n, k): conjunto de classificadores
2  inicio
3     C ← ∅;                    {C é ajustado inicialmente como vazio}
4     para i de 1 até k faça
5        A_i ← amostrar(T, n);  {obtenção da amostra A_i com reposição}
6        C_i ← treinar(A_i, n); {obtenção do classificador C_i a partir de A_i }
7        C ← C ∪ C_i;           {adição do classificador C_i ao conjunto C}
8     fimpara
9     retorne C;                {retorna o conjunto dos k classificadores}
10 fim.
```

Figura 13. Método BAGGING. Adaptado de BREIMAN (1996).

O método BAGGING é particularmente interessante, quando a técnica de aprendizagem de máquina, aplicado em uma base de dados, possui um comportamento instável; uma pequena mudança na base de dados gera classificadores substancialmente diferentes. Logo, um único

classificador não é capaz de oferecer uma resposta confiável para todas as situações; um classificador composto pode ter maior chance de acerto. Em outras palavras, o método BAGGING permite a obtenção de modelos que melhoram a taxa de acerto se comparado a métodos de obtenção de modelos simples, mas, perde-se em termos de facilidade de interpretação [TAN et al., 2006].

A Figura 14 mostra esquematicamente as etapas do método BAGGING, a saber:

- gerar diferentes amostras a partir da mesma base de dados de treinamento, onde as amostras são geradas com reposição, distribuição uniforme e tamanhos idênticos;
- obter um classificador para cada amostra;
- agrupar classificadores obtidos individualmente na etapa anterior em uma unidade de decisão por consenso; e
- classificar por consenso, onde elege-se a classificação mais popular dentre os classificadores individuais por meio de uma votação simples [BAUER et al., 1999].

É fundamental salientar que o método BAGGING, assim como os demais métodos de combinação de classificadores, gera uma unidade de decisão composta formada por um conjunto de classificadores simples. Nestes termos, o produto final do método BAGGING não é um único classificador, assim como não é a fusão de k árvores de decisão em uma árvore única. E cada nova instância a ser classificada é avaliada pela unidade de decisão composta, cuja classificação da instância é a escolha realizada pela maioria dos k classificadores.

Em termos práticos, o processo de classificação do método BAGGING equivale à situação onde um gestor humano que se cerca de k consultores e toma a decisão baseada na votação feita por estes consultores [WITTEN & FRANK, 2000]. Teoricamente, o resultado de um processo decisório consensual obtido de k consultores ou de k classificadores tende a apresentar uma taxa de acerto maior do que a taxa de acerto de cada um individualmente.

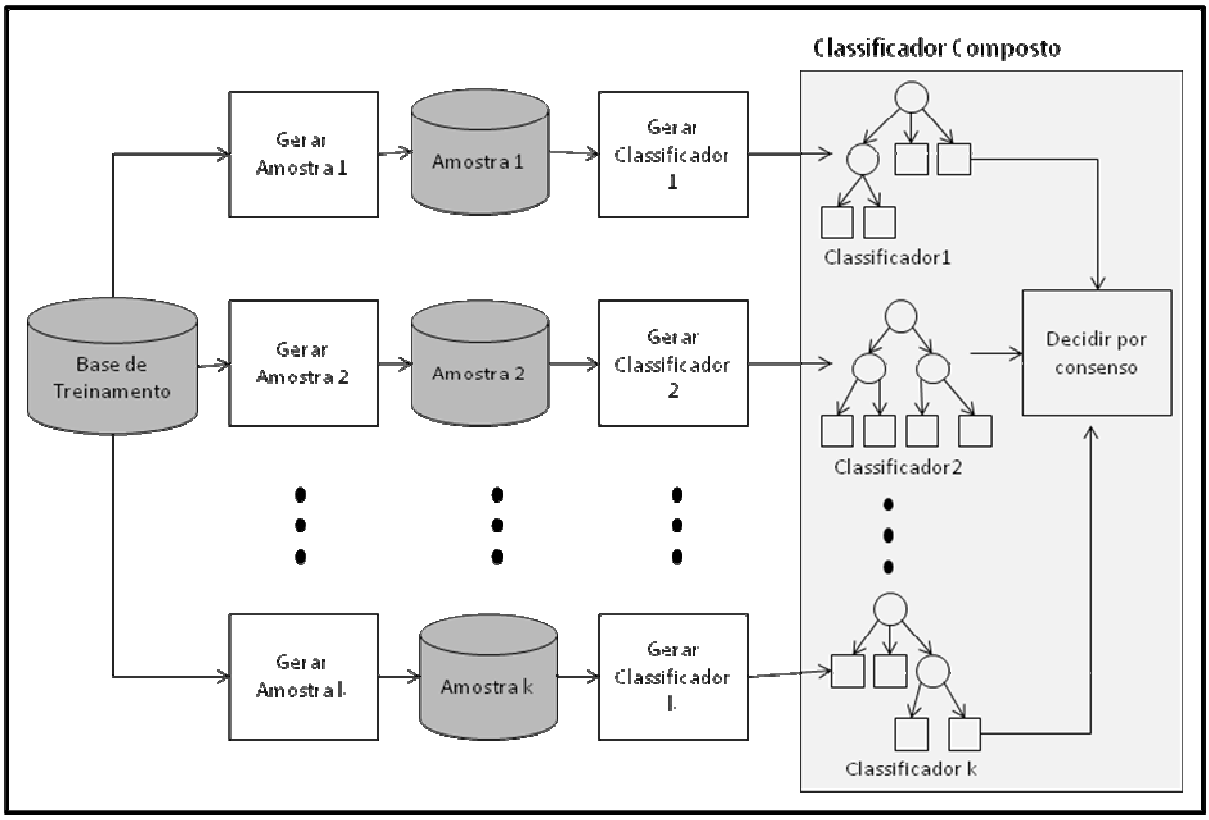

Figura 14. Esquema geral de aplicação do resultado do método BAGGING.

A utilização do resultado do método BAGGING sobre a base de treinamento representada pela Tabela 5 dar-se-á no seguinte cenário:

- a base de treinamento T possui 14 exemplos;
- três amostras geradas com reposição, conforme esquema Tabela 5; e
- a unidade de decisão composta possui 3 classificadores.

Para facilitar a visualização das amostras da Tabela 7, as instâncias foram ordenadas; a coluna P indica a posição da instância na tabela de treinamento original. Como na base original, o atributo meta corresponde à coluna mais a direita de cada da amostra, neste caso o atributo meta é o ponto de aceleração. A geração destas amostras corresponde à execução da linha cinco do algoritmo da Figura 13.

Tabela 7. Amostras com reposição e exemplos com pesos idênticos.

AMOSTRA I				
P	**TP**	**VE**	**AC**	**PC**
1	E	M	N	A
2	E	M	N	M
3	E	M	S	A
4	E	B	N	M
5	E	B	S	M
6	N	M	N	A
6	N	M	N	A
7	N	M	S	M
7	N	M	S	M
9	N	B	N	A
10	C	M	N	M
10	C	M	N	M
10	C	M	N	M
14	C	B	N	M

AMOSTRA II				
P	**TP**	**VE**	**AC**	**PC**
1	E	M	N	A
3	E	M	S	A
3	E	M	S	A
6	N	M	N	A
6	N	M	N	A
7	N	M	S	M
8	N	B	S	A
9	N	B	N	A
10	C	M	N	M
10	C	M	N	M
11	C	M	N	A
13	C	B	S	M
13	C	B	S	M
14	C	B	N	M

AMOSTRA III				
P	**TP**	**VE**	**AC**	**PC**
2	E	M	N	M
2	E	M	N	M
2	E	M	N	M
3	E	M	S	A
3	E	M	S	A
3	E	M	S	A
4	E	B	N	M
5	E	B	S	M
6	N	M	N	A
8	N	B	S	A
12	C	M	S	M
14	C	B	N	M
14	C	B	N	M
14	C	B	N	M

LEGENDA:
TP:Tempo {E:Ensolarado, N:Nublado, C:Chuvoso}
VE:Velocidade {M:Media, B:Baixa}
AC:Aclive {Não, Sim}
PC:Ponto de Aceleração {M:Manter, A:Aumentar}

A Figura 15 mostra um conjunto de três classificadores, sendo que os classificadores C1, C2 e C3 foram obtidos, respectivamente, a partir da Amostra I, II e III da Tabela 7, utilizando o algoritmo *Find Class*. Para facilitar a leitura, os classificadores C1, C2 e C3 gerados foram escritos na forma de três conjuntos de regras ordenadas.

```
CLASSIFICADOR C1
   SE TEMPO = CHUVOSO ENTÃO PC = MANTER
   SE TEMPO = ENSOLARADO ENTÃO PC = MANTER
   SE TEMPO = NUBLADO E ACLIVE = NÃO ENTÃO PC = AUMENTAR
   SE TEMPO = NUBLADO E ACLIVE = SIM ENTÃO PC = MANTER
PESO DE C1: 1

CLASSIFICADOR C2
   SE TEMPO = CHUVOSO E ACLIVE = NÃO ENTÃO PC = AUMENTAR
   SE TEMPO = NUBLADO E ACLIVE = NÃO ENTÃO PC = AUMENTAR
   SE TEMPO = CHUVOSO E VELOCIDADE = BAIXA E ACLIVE = SIM ENTÃO PC = MANTER
  SE TEMPO = NUBLADO E VELOCIDADE = BAIXA E ACLIVE = SIM ENTÃO PC =
   AUMENTAR
  SE ACLIVE = SIM E VELOCIDADE = MEDIA ENTÃO PC = MANTER
  PC = MANTER
PESO DE C2: 1

CLASSIFICADOR C3
   SE TEMPO = CHUVOSO E VELOCIDADE = MÉDIA ENTÃO PC = MANTER
   SE TEMPO = ENSOLARADO E VELOCIDADE = MÉDIA ENTÃO PC = AUMENTAR
   SE TEMPO = NUBLADO E VELOCIDADE = MÉDIA NTÃO PC = AUMENTAR
PESO DE C3: 1
```

Figura 15. Classificador composto – exemplo BAGGING.

A Tabela 8 possui uma instância cujas predições são desconhecidas. Para conhecê-las, o processo consiste em submetê-las aos três classificadores da Figura 15. Cada classificador

informa como resposta um valor de predição. O valor de predição mais frequente é a classificação.

Tabela 8. Instâncias a serem classificadas pelo método BAGGING.

Instâncias	ATRIBUTOS			
	PREVISORES			META
	tempo	velocidade	aclive	Ponto de Aceleração
a1	ensolarado	Media	não	?

A classificação da instância a_1 resulta na resposta **AUMENTAR**, para o Classificador *C1*; **MANTER**, para o Classificador *C2*; e **AUMENTAR**, para o Classificador *C3*. Consequentemente, a resposta **AUMENTAR** será considerada, visto que a mesma obteve dois votos contra um.

Teoricamente, a eficiência dos classificadores, em termos de taxa de acerto, segue a seguinte ordem: AFC, BAGGING+AFC. A taxa de acerto está ligada a forma de validação de cada classificador, seja ele simples ou composto. Nesta dissertação, a técnica de validação escolhida foi à cruzada por produzir bons resultados e por ser uma das formas mais utilizadas.

2.7 Validação Cruzada

Em aprendizagem de máquina, a obtenção do conhecimento, por exemplo, na forma de uma árvore de decisão é uma primeira tarefa fundamental. A segunda tarefa é estimar a taxa de acerto do conhecimento descoberto. Nestes termos, a escolha de uma boa técnica de validação é essencial, assim sendo, a técnica de validação cruzada é uma das formas mais completas e utilizadas para estimar a taxa de acerto de classificadores ([DIAMANTIDIS et al. 2000], [KOHAVI, 1995]).

É importante salientar que a técnica de validação empregada deva ser simples e não tendenciosa. Em termos de validação simples, uma opção seria verificar a taxa de acerto de um classificador *C* obtido por meio de um método qualquer *M* e de uma base de dados de treinamento *T* com *n* instâncias, e gerar o classificador *C* utilizando as n instâncias de *T* como conjunto de treinamento e testar o classificador C utilizando a mesma base de dados *T*. Esta técnica é simples por que ela gera um valor de taxa de acerto a partir de uma única taxa de acerto; ou seja, a taxa de acerto final não é uma composição/média de várias outras. Todavia, tal valor de taxa de acerto pode ser tendencioso. Isto pode ocorrer por que o mesmo conjunto

de instâncias foi utilizado para gerar o classificador e para testá-lo. Uma alternativa é fazer com que a técnica combine vários valores de taxa de acerto.

O princípio da técnica de validação cruzada é simples. Aqui, a base de dados *T* com *n* instâncias é subdividida em *f* amostras geradas aleatoriamente. O tamanho de cada amostra é igual ao número de instâncias *n* dividido pelo número de amostras *f*. Tal fração pode gerar um valor aproximado para o caso *n* de uma divisão não exata [TAN et al., 2006].

A operacionalização do método de validação cruzada é a seguinte: repete *f* vezes um processo de treinamento e teste. Sendo que, cada interação utiliza uma das *f* amostras para ser o conjunto de teste/validação *tf* e as demais *f-1* amostras para ser o conjunto de treinamento *tt*. Cada uma destas *f* amostras de validação retorna uma taxa de acerto tx_i. A taxa de acerto final *txf* é expressa pela média aritmética simples das tx_i, conforme a fórmula a seguir:

$$txf = \frac{1}{f}\sum_{i=1}^{f} tx_i$$

Ilustração do Método Validação Cruzada

Seja T uma base de dados com *n* instâncias, onde *n* é igual a duzentos. Seja o número de *folds* *f* igual quatro. Para estes valores têm-se as seguintes amostras: A_1, A_2, A_3, A_4, A_5. Cada amostra A_i tem quarenta instâncias. As configurações para as interações são as seguintes:

Interação 1:
Entradas: conjunto de treinamento T é {A_2, A_3, A_4 }
conjunto de teste *tt* é {A_1}
Saídas: classificador C_1
taxa de acerto tx_1 é 92%

Interação 2:
Entradas: conjunto de treinamento T é {A_1, A_3, A_4 }
conjunto de teste *tt* é {A_2}
Saídas: classificador C_2
taxa de acerto tx_2 é 92%

Interação 3:
Entradas: conjunto de treinamento T é {A_1, A_2, A_4 }
conjunto de teste *tt* é {A_3}
Saídas: classificador C_3
taxa de acerto tx_3 é 98%

Interação 4:
Entradas: conjunto de treinamento T é {A_1, A_2, A_3 }
conjunto de teste *tt* é {A_4}
Saídas: classificador C_4
taxa de acerto tx_4 é 97%

Taxa de acerto final é 94,7%

Como descrito anteriormente, a técnica de validação cruzada pode ser aplicada a qualquer método de aprendizagem de máquina.

2.8 Trabalhos Relacionados

Diversas técnicas de aprendizagem de máquina com a utilização de análise formal de conceitos vêm sendo utilizadas como opção na resolução de determinados problemas e nas mais variadas áreas.

Silva et al., (2007) fizeram um estudo envolvendo a descrição e aplicação de alguns algoritmos de classificação que utilizam análise formal de conceitos, bem como *Grand*, *Rulearner*, *Legal*, *Galois*, *Similares 1* e *Similares 2*. Foi utilizada neste trabalho uma base de dados para aplicação e comparação dos resultados destes algoritmos com alguns algoritmos clássicos existentes em mineração de dados. Foi realizada também, uma avaliação na estrutura de tais algoritmos juntamente com os resultados de desempenho obtidos em cada um.

Ainda na área de mineração de dados, Belohlavek (et al, 2008) propôs um novo método para geração de regras por meio da indução de árvores de decisão utilizando análise formal de conceitos, o qual teve a preocupação de explorar as técnicas de análise formal de conceitos na implementação de árvores de decisão. Além disso, Poelmans (et al, 2010) fez um estudo da aplicabilidade desta técnica na resolução de certos problemas em mineração de dados, tais como associação e classificação de bases de dados, mineração na web, aplicações em biologia e medicina e também aplicação em lógica *fuzzy*.

Por outro lado, alguns autores desenvolveram pesquisas também na área de engenharia de software, como Buchli (et al, 2003), que desenvolveu uma ferramenta para detectar padrões de projeto em códigos orientados a objetos. O objetivo da ferramenta é, por meio da análise de determinado código orientado a objetos, detectar quais são os padrões de projeto existentes e também fazer a indicação de qual padrão poderia estar sendo utilizado na codificação.

Seguindo a mesma linha, Arévalo (et al, 2004) também desenvolveu uma ferramenta para realizar engenharia reversa em códigos orientados a objetos com o intuito de detectar dependências nesses códigos, bem como conexões entre classes, atributos e métodos.

2.9 Considerações Finais

Ao longo deste capítulo, foram examinados dois métodos de geração de classificadores utilizados nos experimentos, a saber: C4.5, BAGGING, *Find Class* e também uma técnica de validação, conhecida como validação cruzada.

Sobre os métodos examinados, o C4.5, além de possuir a capacidade de gerar modelos de classificação a partir de exemplos com valores faltantes, apresenta bons resultados em bases de dados ruidosas. Frente a estas características, o C4.5 tende a apresentar resultados bastante satisfatórios, principalmente quando combinado com método BAGGING. Apresentaram-se também as principais definições matemáticas que dizem respeito à teoria sobre a AFC, sendo discutidos principalmente os conceitos sobre relações de ordem e reticulados; bem como seus elementos. Também foram abordados os princípios fundamentais de AFC, que são: contexto, conceito e reticulado conceitual. O interesse particular aqui é o algoritmo *Find Class* para geração de regras de classificação, o qual foi mostrado passo a passo os procedimentos desde a obtenção do contexto até a geração dos classificadores.

Na literatura, o método BAGGING mostrou-se eficiente para reduzir a taxa de erro de classificadores quando aplicado em bases de dados com características diferentes [BREIMAN, 1996], bem como mostrou eficiência satisfatória em amostras com uma quantidade pequena de instâncias [LOPES, 2007].

A validação cruzada é particularmente interessante por reduzir a possibilidade de taxas de acerto tendenciosas. A técnica também tem méritos pela simplicidade de realização e compreensão.

Nos próximos capítulos usaremos o termo AFC para representar o algoritmo de extração de regras *Find Class*, que usa análise forma de conceitos.

3 METODOLOGIA

O presente trabalho tem o objetivo de descobrir padrões, a partir de dados coletados por meio de diferentes sensores instalados em um trem de carga; para ajudar no planejamento e execução de uma boa política de condução. Além disso, é interesse fazer a comparação dos padrões descobertos com algoritmos clássicos de mineração (ex: C4.5) com algoritmo de análise formal de conceitos. A consecução desta meta inclui as seguintes tarefas:

- a implementação do algoritmo *Find Class* da análise formal de conceitos;
- a execução de testes afim de extrair padrões para o processo de condução;
- a análise para validação das regras (taxas de acerto dos classificadores e similaridade da condução); e
- a comparação dos resultados obtidos com o algoritmo C4.5 e AFC

Deve-se salientar que o estudo detalhado dos dados coletados a partir de diferentes sensores instalados em um trem já foi realizado por Borges (2009).

Em Borges (2009), são enumerados os objetivos que devem nortear a aplicação dos conhecimentos descobertos, no contexto de uma boa condução de um trem de carga segundo a ALL (2008), que são:

- elevar a economia de combustível;
- reduzir os esforços internos dos veículos e destes sobre as vias;
- reduzir os danos ao equipamento; e
- reduzir/eliminar danos à carga.

Assume-se aqui também como foco principal neste trabalho a geração de regras de condução que auxiliem a decisão de qual ponto de aceleração utilizar. A aceitação de uma regra de auxílio a condução deve inexoravelmente aderir aos princípios de boa condução.

3.1 Modelagem dos Dados

Para entendimento da solução proposta foi desenvolvido o modelo de domínio parcial do problema, o qual é mostrado na Figura 16 e representa uma viagem de trem. A viagem é decorada por meio de vários recursos: operador (es) que pode ser um ser humano e/ou agente de condução virtual, um trem decorado por meio de vários vagões e locomotivas — cada locomotiva é decorada por um conjunto de nove pontos de aceleração definidos em termos de potência e consumo, cada trem, vagão e locomotiva possui também uma tara e um peso, uma via férrea pode ser decorada por meio de conjunto de seguimentos, cruzamentos, desvios e pontos de medidas.

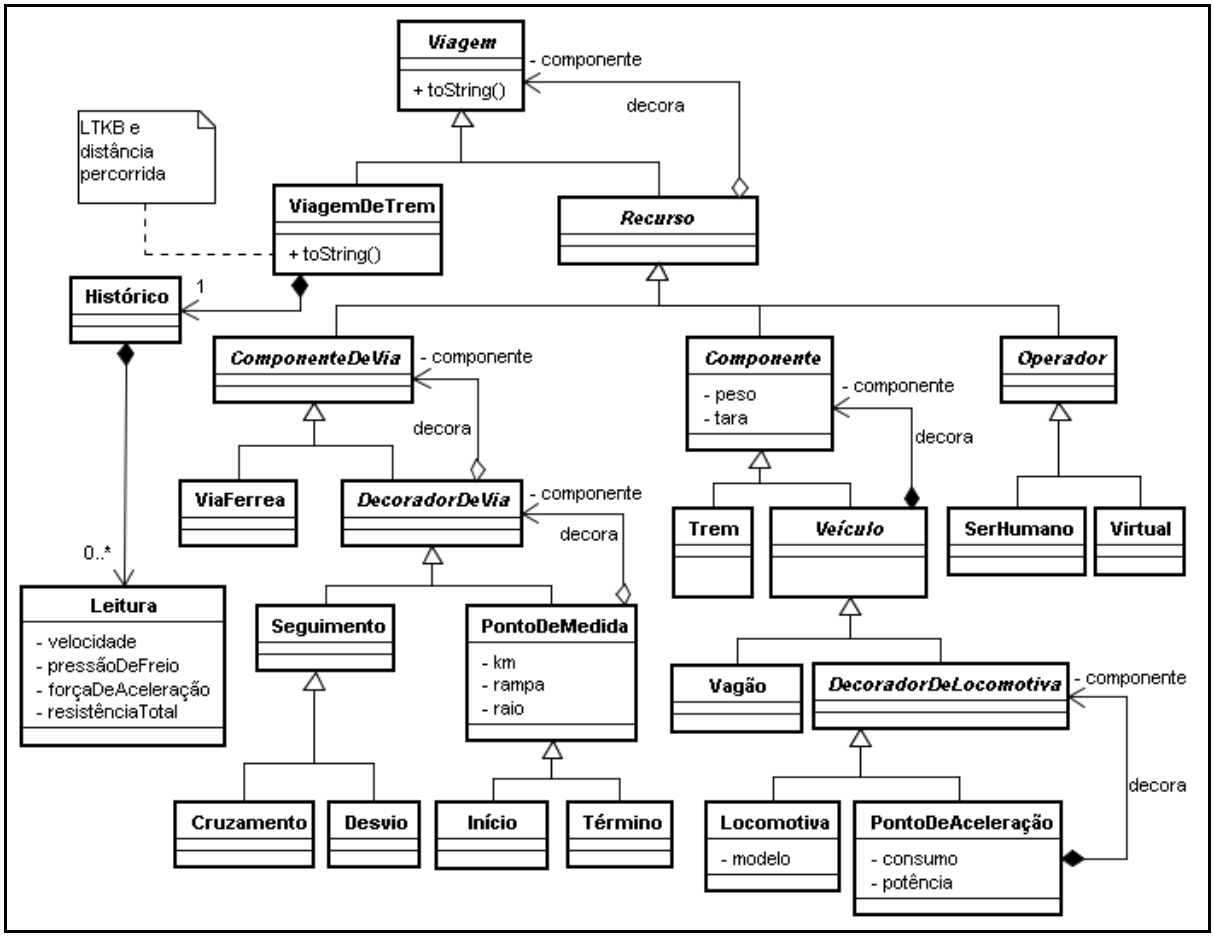

Figura 16. Modelo conceitual parcial (adaptado de BORGES, 2009).

Cada seguimento, cruzamento e desvio podem ter cada um deles o seu conjunto de pontos de medidas. Pode-se ter ponto de medida específico definido para marcar o início ou término de um seguimento, cruzamento ou desvio, bem como de início e término de uma viagem.

Finalmente, uma viagem de trem tem um histórico definido por um conjunto de leituras de valores dos sensores embarcados e de cálculos de resistência e força de aceleração. Tais valores podem ser: velocidade, pressão de freios, resistência total e força de aceleração (Borges, 2009).

3.2 Origem e Formato dos Dados

Os dados utilizados para a geração da base de dados inicial são originários de quatro fontes diferentes [BORGES 2009], que são:

- FONTE I: leitura de sensores de movimento: quilômetro, velocidade, velocidade máxima, pressão de freios, tempo (hh:mm:ss), ponto de aceleração, posição da manopla de reverso de movimento do trem (para frente/para trás), indicação de intervenção do maquinista, freio dinâmico (ligado/desligado), sequencial de identificação de evento, indicação de patinagem e relê terra.
- FONTE II: informa as datas de início e fim de uma viagem, bem como as identificações das subestações de início e término da viagem. Cada subestação tem um tipo associado (principal/não principal).
- FONTE III: informa o peso do trem (em toneladas), distância percorrida (em metros), consumo final (em toneladas transportadas por quilômetro), identificação do trem, identificação do operador.
- FONTE IV: informa apenas as identificações das locomotivas da viagem.

A Tabela 9 mostra um resumo dos dados de oito viagens de trem realizadas. As características dos dados são:

- Trecho que compreende as cidades de Londrina (PR) e Paiçandu (PR)
- Número total de registros da base de dados: aproximadamente 17.164
- Número de registros usados nos experimentos: aproximadamente 14.258
- Número de registros em cada viagem: 1.600 a 2.800
- Trecho total percorrido: 69.500 metros (um registro para cada 33 metros)

Tabela 9. Resumo de dados de diferentes viagens (BORGES, 2009)

ID DA VIAGEM	QUILÔMETRO		NÚMERO DE REGISTROS (NR)		METROS PERCORRIDOS (NP)	NP ÷ NR	TEMPO DE VIAGEM (MIN)
	INICIAL	FINAL	ORIGINAL	FILTRADO			
1	339495	268014	2154	1999	71481	33	184
2	335693	268837	2350	1695	66856	28	134
3	336858	269919	1763	1694	66939	38	144
5	340627	268443	1817	1666	72184	39	150
6	339779	269160	2061	1962	70619	34	172
7	335874	268370	1699	1566	67504	39	176
8	341434	268108	2514	1541	73326	29	208
9	339980	268289	2806	2135	71691	25	197

Na sequência será mostrado apenas um resumo de todo o processo do tratamento e enriquecimento dos dados. Os detalhes destes processos são encontrados em Borges (2009).

3.2.1 Tratamento dos Dados

Para o tratamento dos dados foram utilizadas as seguintes técnicas: remoção de ruídos, inclusão de novos atributos, seleção de atributos e transformação dos dados. Por primeiro, a remoção de ruídos permitiu a obtenção de dados e cálculos consistentes, sendo utilizado um conjunto de regras empíricas para a remoção. O segundo passo foi à inclusão de novos atributos, bem como: peso médio dos vagões, peso médio das locomotivas, números de vagões do trem, área frontal do vagão, área frontal da locomotiva, comprimento do vagão, comprimento da locomotiva, número de eixos do vagão, número de eixos da locomotiva, tamanho da bitola e coeficiente de aderência. Além desses, foram adicionados dados relativos aos perfis das vias; o que possibilitou uma melhor descrição e enriquecimento dos dados.

Com relação à seleção de atributos, foi realizada a redução por meio da utilização dos métodos *CfsSubSetEval* com o algoritmo *BestFirst* e o *GainRatio* usando o algoritmo *Ranker*, os quais reduziram em 95% e 12% respectivamente. Por fim, foi realizada a discretização do atributo contínuo LTKB (que representa dados sobre o consumo) a fim de utilizá-lo como atributo meta, visto que este atributo representa um dos objetivos de uma boa condução [BORGES, 2009]. Importante lembrar que neste trabalho somente foi utilizado o atributo Ponto de Aceleração para a geração dos classificadores.

3.3 Mineração de Dados

Após a finalização do tratamento dos dados, a mineração foi iniciada por meio da aplicação do JRIP e do AFC; e na sequência, também foi aplicado método de combinação de classificadores BAGGING.

O próximo passo foi a aplicação do algoritmo *Resample*[1] para definir uma amostragem da base de dados. Este algoritmo permite definir o tamanho da amostra que deseja obter em relação ao tamanho da base de dados original. Os tamanhos das amostras geradas variaram de 10% a 30%, sendo que não foram utilizadas amostras com tamanhos maiores devido à falta de recursos computacionais para gerar os classificadores.

Foram gerados conjuntos de dados para treinamento e para testes, sendo que o primeiro foi utilizado para aplicação dos algoritmos durante a geração dos classificadores. O segundo conjunto foi usado para realizar testes na eficiência do classificador.

O próximo passo foi à aplicação dos algoritmos de mineração (JRIP e AFC), sendo utilizado o atributo meta PONTO DE ACELERAÇÃO. Este atributo é usado para indicar como classe-alvo são capazes de indicar ao condutor qual ponto aplicar para o deslocamento do trem e também para atravessar um conjunto de resistências (BORGES, 2009).

A utilização dos classificadores somente é possível com um software capaz de calcular todas as variáveis envolvidas na viagem, simulando assim o desenrolar de uma viagem real. Tal software foi desenvolvido por Borges (2009) e foi usado ao longo deste trabalho. Os detalhes de tal software estão descritos na referência aqui citada.

Para avaliar a qualidade dos resultados gerados, foram utilizadas técnicas de validação cruzada e distância do cosseno, sendo que este último serviu para medir a similaridade entre as ações tomadas pelo maquinista e pelo sistema automático de condução. Esta medida foi usada por traduzir o grau de “imitação” na forma de condução entre o agente automático de condução e o maquinista ser humano.

3.4 Considerações Finais

A metodologia apresentada utiliza as etapas clássicas do processo de mineração de dados e descoberta de conhecimentos. Um ponto importante foram as etapas de tratamento e

[1] Este algoritmo é uma implementação constante no pacote de software WEKA.

enriquecimento dos dados e também, a definição de uma métrica para medir a similaridade da condução de um maquinista humano e do agente de condução automática. Esses dois itens foram realizados por Borges (2009); sendo que o esforço principal deste trabalho foi à aplicação e comparação das regras geradas pelas técnicas de análise formal de conceitos com técnicas clássicas de mineração de dados (ex: C4.5).

4 RESULTADOS

Os resultados mostrados neste capítulo referem-se aos experimentos realizados utilizando classificadores gerados a partir de uma base de dados de viagens reais enriquecidas, contendo 14.258 registros armazenados e 119 atributos. Doravante tal base de dados será identificada por *BD*.

Os classificadores gerados pelos métodos JRIP, AFC, JRIP+BAGGING e AFC+BAGGING usados nos experimentos foram obtidos a partir de amostras de 10%, 20% e 30% da base *BD*. Tais classificadores gerados possuem algumas diferenças, no que se refere ao conjunto de treinamento, podendo este ser de quatro tipos:

- CN: base de dados *BD* com conjunto de dados contendo leituras de paradas intermediárias (C) e com o número padrão de atributos-meta *pontos de aceleração* (N);
- C4: base de dados *BD* com informações de paradas intermediárias (C) e apenas quatro atributos-meta (4);
- SN: base de dados *BD* sem informações de paradas intermediárias (S) e com o número padrão atributos-meta *pontos de aceleração* (N); e
- S4: base de dados *BD* sem informações de paradas intermediárias (S) e com número reduzido atributos-meta *pontos de aceleração* (4);

Sobre o conjunto de treinamento duas formas de avaliação foram utilizadas: a primeira utilizando 30% do total de dados do conjunto de treinamento para testes (TT) e a segunda utilizando o método de validação cruzada (VC).

Pôde-se observar em BD que as classes 7 e 8, referentes aos pontos de aceleração, tinham valores bem superiores às demais classes, o que caracterizou um desbalanceamento entre classes. O procedimento realizado para reduzir tal desbalanceamento foi rotular todos os registros das classes de 1 a 6 para a classe 3. Restando então quatro classes, a saber: -1, 3, 7 e 8. Este abordagem foi a mesma usada em Borges (2009).

As análises subsequentes, sobre as quatro bases de dados supracitadas, devem responder questões como: (i) qual o melhor classificador obtido, (ii) qual o grau de

similaridade entre a condução realizada pelo maquinista e pelas regras de classificação geradas; e, por fim, (iii) qual o comparativo entre a condução realizada pelas regras AFC e JRIP.

4.1 Geração e Avaliação dos Classificadores

Os dados sobre os classificadores foram estruturados em quatro tabelas. Elas apresentam respectivamente os dados dos classificadores: JRIP, JRIP+BAGGING, AFC, AFC+BAGGING. Os valores da coluna “Identificador” foram utilizados ao longo deste capítulo para identificar a configuração do experimento realizado, assim como os classificadores gerados a partir de um método específico. Este último foi referenciado apenas como “classificador <nome do método>”.

Cada tabela apresenta as taxas de acertos dos respectivos classificadores. Aqueles que apresentaram as melhores taxas de acertos foram obtidos por meio do método JRIP+BAGGING, chegando a taxas de 96% de acerto em alguns casos (JRBA_30_S4_VC). O classificador com a menor taxa de acerto foi gerado pelo método AFC, quando utilizado apenas 10% da amostra de dados (AFC_10_CN_TT). Os classificadores gerados a partir do método BAGGING tiveram uma taxa de acerto média acima de 77% na configuração AFC+BAGGING e uma taxa de acerto média de 83% para a configuração JRIP+BAGGING. Houve apenas um caso onde um classificador JRIP gerou uma taxa de acerto superior a configuração AFC+BAGGING (JR_20_C4_TT). Esta última configuração gerou apenas três classificadores com taxas de acerto superiores a configuração JRIP+BAGGING (AFCBA_10_SN_TT, AFCBA_10_S4_TT e AFCBA_20_SN_TT). Dentre os classificadores BAGGING a configuração JRIP+BAGGING foi superior em 87% dos casos. Analogamente, dentre os classificadores simples (JRIP e AFC), o JRIP foi superior em 95% dos casos.

Tabela 10. Identificadores dos experimentos realizados com o método JRIP (BORGES, 2009)

Método	Tamanho da Amostra	Conjunto de Treinamento	Avaliação	Identificador	Taxa de Acerto
JRip	10	CN	TT	JR_10_CN_TT	46,9%
			VC	JR_10_CN_VC	76,3%
		C4	TT	JR_10_C4_TT	68,2%
			VC	JR_10_C4_VC	83,0%
		SN	TT	JR_10_SN_TT	48,4%
			VC	JR_10_SN_VC	80,8%
		S4	TT	JR_10_S4_TT	65,9%
			VC	JR_10_S4_VC	82,3%
	20%	CN	TT	JR_20_CN_TT	56,4%
			VC	JR_20_CN_VC	78,4%
		C4	TT	JR_20_C4_TT	73,6%
			VC	JR_20_C4_VC	87,6%
		SN	TT	JR_20_SN_TT	56,9%
			VC	JR_20_SN_VC	83,1%
		S4	TT	JR_20_S4_TT	73,1%
			VC	JR_20_S4_VC	89,1%
	30%	CN	TT	JR_30_CN_TT	58,5%
			VC	JR_30_CN_VC	77,2%
		C4	TT	JR_30_C4_TT	78,6%
			VC	JR_30_C4_VC	87,6%
		SN	TT	JR_30_SN_TT	60,6%
			VC	JR_30_SN_VC	79,9%
		S4	TT	JR_30_S4_TT	76,9%
			VC	JR_30_S4_VC	91,2%

Tabela 11. Identificadores dos experimentos realizados com o método JRIP+BAGGING (BORGES, 2009)

Método	Tamanho da Amostra	Conjunto de Treinamento	Avaliação	Identificador	Taxa de Acerto
JRip + BAGGING	10	CN	TT	JRBA_10_CN_TT	58,9%
			VC	JRBA_10_CN_VC	93,0%
		C4	TT	JRBA_10_C4_TT	76,1%
			VC	JRBA_10_C4_VC	94,7%
		SN	TT	JRBA_10_SN_TT	58,9%
			VC	JRBA_10_SN_VC	95,0%
		S4	TT	JRBA_10_S4_TT	72,5%
			VC	JRBA_10_S4_VC	94,8%
	20%	CN	TT	JRBA_20_CN_TT	64,4%
			VC	JRBA_20_CN_VC	93,5%
		C4	TT	JRBA_20_C4_TT	79,7%
			VC	JRBA_20_C4_VC	94,3%
		SN	TT	JRBA_20_SN_TT	64,8%
			VC	JRBA_20_SN_VC	93,5%
		S4	TT	JRBA_20_S4_TT	79,9%
			VC	JRBA_20_S4_VC	95,0%
	30%	CN	TT	JRBA_30_CN_TT	68,6%
			VC	JRBA_30_CN_VC	92,4%
		C4	TT	JRBA_30_C4_TT	84,2%
			VC	JRBA_30_C4_VC	95,7%
		SN	TT	JRBA_30_SN_TT	69,4%
			VC	JRBA_30_SN_VC	93,9%
		S4	TT	JRBA_30_S4_TT	84,7%
			VC	JRBA_30_S4_VC	96,2%

Tabela 12. Identificadores dos experimentos realizados com o método AFC.

Método	Tamanho da Amostra	Conjunto de Treinamento	Avaliação	Identificador	Taxa de Acerto
AFC	10	CN	TT	AFC_10_CN_TT	38,7%
			VC	AFC_10_CN_VC	62,1%
		C4	TT	AFC_10_C4_TT	57,2%
			VC	AFC_10_C4_VC	70,4%
		SN	TT	AFC_10_SN_TT	41,5%
			VC	AFC_10_SN_VC	69,7%
		S4	TT	AFC_10_S4_TT	58,1%
			VC	AFC_10_S4_VC	73,3%
	20%	CN	TT	AFC_20_CN_TT	50,4%
			VC	AFC_20_CN_VC	69,5%
		C4	TT	AFC_20_C4_TT	68,6%
			VC	AFC_20_C4_VC	77,7%
		SN	TT	AFC_20_SN_TT	52,4%
			VC	AFC_20_SN_VC	76,4%
		S4	TT	AFC_20_S4_TT	69,7%
			VC	AFC_20_S4_VC	71,1%
	30%	CN	TT	AFC_30_CN_TT	52,7%
			VC	AFC_30_CN_VC	71,7%
		C4	TT	AFC_30_C4_TT	74,7%
			VC	AFC_30_C4_VC	76,8%
		SN	TT	AFC_30_SN_TT	57,6%
			VC	AFC_30_SN_VC	69,4%
		S4	TT	AFC_30_S4_TT	70,7%
			VC	AFC_30_S4_VC	81,2%

Tabela 13. Identificadores dos experimentos realizados com o método AFC+BAGGING.

Método	Tamanho da Amostra	Conjunto de Treinamento	Avaliação	Identificador	Taxa de Acerto
AFC + BAGGING	10	CN	TT	AFCBA_10_CN_TT	50,9%
			VC	AFCBA_10_CN_VC	81,2%
		C4	TT	AFCBA_10_C4_TT	72,1%
			VC	AFCBA_10_C4_VC	83,7%
		SN	TT	AFCBA_10_SN_TT	59,9%
			VC	AFCBA_10_SN_VC	88,3%
		S4	TT	AFCBA_10_S4_TT	74,5%
			VC	AFCBA_10_S4_VC	83,1%
	20%	CN	TT	AFCBA_20_CN_TT	57,8%
			VC	AFCBA_20_CN_VC	89,2%
		C4	TT	AFCBA_20_C4_TT	72,9%
			VC	AFCBA_20_C4_VC	87,4%
		SN	TT	AFCBA_20_SN_TT	66,9%
			VC	AFCBA_20_SN_VC	90,2%
		S4	TT	AFCBA_20_S4_TT	78,4%
			VC	AFCBA_20_S4_VC	86,7%
	30%	CN	TT	AFCBA_30_CN_TT	67,1%
			VC	AFCBA_30_CN_VC	87,4%
		C4	TT	AFCBA_30_C4_TT	79,8%
			VC	AFCBA_30_C4_VC	90,1%
		SN	TT	AFCBA_30_SN_TT	67,8%
			VC	AFCBA_30_SN_VC	89,2%
		S4	TT	AFCBA_30_S4_TT	78,7%
			VC	AFCBA_30_S4_VC	82,6%

4.2 Análise Comparativa dos Classificadores: Teste de Friedman

A análise comparativa dos classificadores limitou-se a aplicação do teste de Friedman, o qual se trata de um teste não-paramétrico, ou seja, não requer o conhecimento da distribuição da variável da população. Este teste permite ranquear os algoritmos (JRIP, JRIP+BAGGING, AFC, AFC+BAGGING), permitindo obter a resposta de qual teve o melhor comportamento.

O principal objetivo do teste de Friedman é conferir se os classificadores gerados possuem diferenças expressivas, sendo caracterizada hipótese nula; ou seja, quando não existem diferenças entre os algoritmos. Para verificar se existe ou não correlação entre os dados, deve-se fazer o somatório das variâncias dos ranques, para então, através deste somatório calcular a probabilidade do valor ser superior ou igual à variância obtida utilizando a distribuição *qui-quadrada* com *k-1* graus de liberdade. O resultado numérico final deste teste apresenta um nível de significância (p-valor). Quando tal valor for menor que 0.05, então é aconselhado rejeitar a hipótese nula (BORGES, 2009).

O valor de *qui-quadrado* do conjunto de valores foi de 3.6 e *p-valor* de 0.11161, sendo que este valor representa a semelhança entre os classificadores. A Tabela 14 apresenta o *p-valor* de cada configuração para o teste de Friedman.

Tabela 14. p-valores obtidos no teste de Friedman.

Tamanho da Amostra	Configuração	p-valor
10%	70% treinamento e 30% teste	0.11161
10%	validação cruzada	0.30802
20%	70% treinamento e 30% teste	0.11161
20%	validação cruzada	0.39163
30%	70% treinamento e 30% teste	0.11161
30%	validação cruzada	0.61494

Para um valor de $p < 0.05$ é possível concluir que não há diferença significativa entre as diferentes configurações, ou seja, é não rejeitada a hipótese nula, visto que todos os testes obtiveram resultados superiores a 0.05.

Para medir o desempenho dos classificadores, foi utilizado o simulador desenvolvido por Borges (2009), o qual mede a similaridade entre as ações de um maquinista ser humano e de um simulador de condução. As decisões de tal simulador são baseadas nos classificadores descritos anteriormente. O esperado é que o grau de similaridade seja o mais próximo de 1.

Este valor significaria em tese que o simulador conseguiu reproduzir o comportamento do maquinista ser humano.

Os demais resultados mostrados a seguir referem-se aos experimentos realizados utilizando o software descrito em [BORGES 2009]. Este software permite colocar em prática os classificadores, ou seja, gerar as políticas de ações a serem confrontadas às políticas de ações dos maquinistas seres humanas.

4.3 Aplicação dos Classificadores e Análise da Similaridade de Condução

O esforço principal neste trabalho foi a obtenção de classificadores, usando AFC, a partir de históricos de viagens de trens e aplicá-los de modo a sugerir o melhor ponto de aceleração a ser aplicado, visando à definição de uma boa política de condução.

Uma forma de avaliar a eficiência de um classificador, na definição de uma política de ação, é medir a similaridade entre a sugestão da ação sugerida pelo classificador e a ação efetivamente aplicada [BORGES 2009]. A métrica foi operacionalizada por meio do cálculo do cosseno [SWOKOWSKI 1983]. O cálculo é realizado pela Equação 01, na qual o vetor $\vec{x}$ representa os pontos de aceleração usados na ação e o vetor e $\vec{y}$ representa os pontos de aceleração que foram sugeridos pelo classificador (*i* é o número total de ações realizadas). Os resultados da equação variam entre zero e um; e, quanto mais próximo de um melhor é o conhecimento obtido, ou seja, as ações tomadas pelo classificador foram mais similares às ações realizadas pelo maquinista humano.

$$\cos\ \theta = \cos(\vec{x}, \vec{y}) = \frac{\vec{x} \times \vec{y}}{\sqrt{\sum x_i^2 \times \sum y_i^2}} \qquad (01)$$

As taxas de similaridades obtidas são diretamente confrontadas com os resultados reais. Nestes termos, foram produzidos os resultados experimentais para as seguintes configurações:

- Viagem Simulada I: realizada aplicando os padrões descobertos a partir de conjunto enriquecido de dados e com dez classes
- Viagem Simulada II: os classificadores obtidos para a simulação desta viagem

baseou-se em um conjunto de dados da Viagem Simulada I. A particularidade reside na redução do número de classes de 10 para 4 (classes: -1, 3, 7 e 8)

- Viagem Simulada III: os classificadores obtidos para a simulação desta viagem baseou-se no conjunto de dados original. A particularidade reside na remoção dos registros relacionados às paradas intermediárias do trem
- Viagem Simulada IV: do mesmo modo que optamos por gerar classificadores com apenas quatro classes, conforme realizado na Viagem Simulada II reproduziu-se o experimento para o conjunto de dados sem paradas.

Em vários trechos, o ponto sugerido pelo classificador foi o mesmo ponto de aceleração aplicado. Três diferentes situações são possíveis:

- quando o valor é zero, tem-se que o ponto de aceleração sugerido e aplicado é mesmo;
- quando o valor for maior que zero, tem-se que o ponto de aceleração sugerido é maior que o aplicado; e
- quando o valor for menor que zero, tem-se que o ponto de aceleração sugerido é menor que o aplicado.

Viagem Simulada I

Esta viagem foi realizada aplicando os padrões descobertos a partir de um conjunto de dados enriquecido e com dez classes (dez diferentes pontos de aceleração). A Tabela 15 resume os melhores resultados. Pode-se observar que o classificador com melhor resultado é o AFC+BAGGING. Ele foi melhor dentre as diferentes viagens com taxa de similaridade superior a 79%. As piores taxas de similaridade foram às resultantes da aplicação do AFC. Em outras palavras, quanto mais próximo de 100% melhor foi a aplicação do classificador na definição de uma política de ação.

Tabela 15. Resultados usando classificadores obtidos a partir do conjunto de treinamento CN.

Classificador	Viagem número	Tamanho da Amostra (%)	Similaridade
AFC+BAGGING	1	10	70%
AFC	2	10	60%
AFC+BAGGING	3	20	75%
AFC	4	20	62%
AFC+BAGGING	5	30	79%
AFC	6	30	63%

Viagem Simulada II

Os classificadores obtidos para a simulação desta viagem baseou-se em um conjunto de dados anterior. A particularidade reside na redução do número de classes de 10 para 4 (classes: -1, 3, 7 e 8). A Tabela 16 mostra os melhores resultados obtidos. Comparando os resultados com os valores Tabela 15, nota-se que não houve uma queda significativa no desempenho dos classificadores. Deve-se destacar que os resultados envolvendo AFC foram ligeiramente melhores e os resultados envolvendo AFC+BAGGING foram ligeiramente piores.

Tabela 16. Resultados usando classificadores obtidos a partir do conjunto de treinamento C4.

Classificador	Viagem Número	Tamanho da Amostra (%)	Similaridade
AFC+BAGGING	1	10	70%
AFC	2	10	68%
AFC+BAGGING	3	20	73%
AFC	5	20	63%
AFC+BAGGING	6	30	74%
AFC	7	30	65%

Apesar de apresentarem resultados próximos aos obtidos nas viagens com dados de paradas e com número original de classes, os valores de similaridade não indicam melhorias significativas.

Viagem Simulada III

A Tabela 17 apresenta os melhores resultados obtidos em cada viagem. Assim como nos experimentos realizados com paradas e com apenas 4 classes, os classificadores gerados pelo método AFC+BAGGING se sobrepõem significativamente em relação aos demais.

Tabela 17. Resultados usando classificadores obtidos a partir do conjunto de treinamento SN.

Classificador	Viagem Número	Tamanho da Amostra (%)	Similaridade
AFC+BAGGING	1	10	70%
AFC	2	10	60%
AFC+BAGGING	3	20	76%
AFC	5	20	61%
AFC+BAGGING	6	30	79%
AFC	7	30	66%

Viagem Simulada IV

Do mesmo modo que optamos por gerar classificadores com apenas quatro classes, substituindo os valores (1, 2, 3, 4, 5 e 6) dos pontos de aceleração pelo valor médio (3) no conjunto de dados de viagens com paradas, reproduzimos o experimento considerando o conjunto de dados sem paradas. A Tabela 18 apresenta os resultados obtidos pelos melhores classificadores em cada viagem. Diferente do que aconteceu nos experimentos "com paradas intermediárias e com apenas quatro classes" e também nos experimentos "sem paradas intermediárias", os resultados mostram que os classificadores BAGGING foram melhores em todas as viagens (viagens 1, 3 e 6) . As taxas de similaridade permaneceram muito próximas das obtidas em experimentos anteriores.

Tabela 18. Resultados usando classificadores obtidos a partir do conjunto de treinamento S4.

Classificador	Viagem Número	Tamanho da Amostra (%)	Similaridade
AFC+BAGGING	1	10	78%
AFC	2	10	70%
AFC+BAGGING	3	20	74%
AFC	5	20	64%
AFC+BAGGING	6	30	71%
AFC	7	30	68%

4.4 Comparativo AFC e JRIP

Após as quatro viagens terem sido executadas, foram selecionados os classificadores que tiveram as melhores taxas de similaridades de condução e montou-se a Tabela 19, que apresenta um comparativo entre as viagens simuladas com os classificadores AFC e com os

classificadores JRIP. Lembrando que maiores detalhes sobre as viagens executadas com os classificadores JRIP encontram-se em Borges (2009).

Tabela 19. Comparativo AFC e JRIP

Base	Tamanho da amostra (%)	Classificador	Similaridade	Classificador	Similaridade
CN	10	AFC + BAGGING	69%	JRIP + BAGGING	81%
CN	10	AFC	59%	JRIP	84%
CN	20	AFC + BABBING	74%	JRIP + BAGGING	84%
C4	10	AFC	68%	JRIP	83%
C4	30	AFC + BAGGING	73%	JRIP + BAGGING	83%
SN	20	AFC	60%	JRIP	87%
SN	20	AFC + BAGGING	75%	JRIP + BAGGING	84%
S4	30	AFC + BAGGING	68%	JRIP	84%
S4	10	AFC + BAGGING	78%	JRIP + BAGGING	82%

A Tabela 19 apresenta diversas configurações, que vão desde o tipo da base de dados (CN, C4, SN e S4), o tamanho da amostra utilizada (10%, 20% e 30%) e as taxas de similaridade entre as duas técnicas. Pode-se observar que o grau de similaridade na condução com os classificadores JRIP foi melhor em todos os casos, tendo uma média de 83,5% contra 69,3% da técnica de AFC.

4.5 Considerações Finais

Todos os classificadores usados nos experimentos foram obtidos a partir de uma mesma base de dados, a qual teve todo o tratamento e enriquecimento dos dados realizado por Borges (2009). Existiram algumas diferenças nas bases de dados no que diz respeito ao número de classes utilizadas, sendo definidos quatro diferentes conjuntos de treinamentos, que foram:

a) T1: base de dados original enriquecida;

b) T2: base de dados T1 com apenas quatro classes;

c) T3: base de dados T1 removidos os registros das paradas intermediárias; e

d) T4: base de dados T2 removidos os registros das paradas intermediárias.

Sobre estes quatro conjuntos de treinamentos foram gerados diferentes classificadores usando as seguintes configurações: JRIP, AFC, JRIP+BAGGING e AFC+BAGGING. Em resumo, foram gerados 96 classificadores. Pode-se concluir para o experimento que os classificadores

gerados a partir do método BAGGING produziram as melhores taxas de acertos e similaridade quando comparadas as viagens executadas pelo simulador com as viagens realizadas pelo maquinista humano.

Em termos de descoberta de conhecimentos e validação objetiva dos mesmos, pode-se dizer que o resultado é bom, considerando que se obteve um grau de similaridade próximo de 70% de conhecimentos aplicáveis. Deve-se enfatizar que os dados de campo usados para comparação referem-se às melhores políticas de condução da prática atual.

5 Conclusões

O trabalho realizado apresentou um estudo sobre a descoberta de padrões para ajudar um maquinista na condução de um trem. Os esforços foram na aplicação de técnicas de aprendizagem de máquina e descoberta de conhecimento a partir de conjuntos de dados de viagens de trens de cargas até aplicação propriamente destes conhecimentos por meio de um simulador de condução desenvolvido por Borges (2009). Além disso, o resultado da aplicação dos conhecimentos foi medido objetivamente.

Descoberta de padrões com o JRIP e a AFC:

As técnicas exploradas, para a obtenção de classificadores, foram à indução de regras e a análise formal de conceitos. A indução de regras foi baseada no algoritmo clássico C4.5. Uma das particularidades interessantes do C4.5 é que ele gera modelos simbólicos de fácil leitura. A extração de regras a partir da análise formal de conceitos foi baseada no algoritmo *Find Class*, o qual também gera regras simbólicas. Além dos modelos simbólicos, buscou-se também obter modelos com boa taxa de acerto. Para tal, se utilizou método de combinação de classificadores BAGGING. Em termos teóricos, o método BAGGING produziu resultados melhores que os demais. Foram gerados 96 classificadores, sendo 24 classificadores para cada uma das seguintes configurações: JRIP, JRIP+BAGGING, AFC, AFC+BAGGING.

A configuração JRIP+BAGGING apresentou as melhores taxas de acertos, chegando a taxas de 96% de acerto em alguns casos (JRBA_30_S4_VC). O classificador com a menor taxa de acerto foi gerado pelo método AFC, quando utilizado apenas 10% da amostra de dados (AFC_10_CN_TT). Os classificadores gerados a partir do método BAGGING tiveram uma taxa de acerto média acima de 77% na configuração AFC+BAGGING e uma taxa de acerto média de 83% para a configuração JRIP+BAGGING. Houve apenas um caso onde um classificador JRIP gerou uma taxa de acerto superior a configuração AFC+BAGGING (JR_20_C4_TT). Esta última configuração gerou apenas três classificadores com taxas de acerto superiores a configuração JRIP+BAGGING (AFCBA_10_SN_TT, AFCBA_10_S4_TT e AFCBA_20_SN_TT). Dentre os classificadores BAGGING a

configuração JRIP+BAGGING foi superior em 87% dos casos. Analogamente, dentre os classificadores simples (JRIP e AFC), o JRIP foi superior em 95% dos casos.

5.1 Trabalhos Futuros

Apesar dos resultados com análise formal de conceitos não terem sido superior à indução de regras por meio do ganho de informação, acredita-se que os resultados podem ser melhorados de forma significativa mudando a forma de preparação dos conjuntos de treinamento. Alguns testes já foram feitos e observou-se que para certos conjuntos de treinamentos a abordagem baseada na análise formal de conceitos resultou em classificador muito bom e (para o mesmo conjunto de dados) a abordagem baseada no ganho de informação resultou em um classificador pífio.

Além disso, podem-se obter melhores resultados por meio da utilização de algoritmos mais sofisticados de análise formal de conceitos, tais como *Grand*, *Rulearner*, *Legal*, *Galois*, *Similares 1* e *Similares 2*.

6 Referências Bibliográficas

ABE, J. M. 1992. *Teoria Intuitiva dos Conjuntos*. São Paulo: Makron Books, 1992.

ARÉVALO, G. B. 2004. *High Level Views in Object-Oriented Systems using Formal Concept Analysis.*2004. Universidade de Bern, Suiça.

BAUER, ERIC e KOHAVI, RON. 1999. *An Empirical Comparison of Voting Classification Algorithms: BAGGING, BOOSTING, and Variants*. Machine Learning, 36(1/2). 1999, pp. 105-139.

BELOHLAVEK, R., BAETS, B. e OUTRATA J. *Inducing decision trees via concept lattices.* 2008. Watson School of Engineering and Applied Science, Estados Unidos.

BORGES, ANDRÉ PINZ. 2009. *Descoberta de Regras de Condução de Trens de Carga.* Curitiba : s.n., 2009. Dissertação de Mestrado em Informática Aplicada – Pontifícia Universidade Católica do Paraná.

BRANCO, PATRÍCIA NASCIMENTO MANFRÉ. 2010. *Concepção de um Sistema de Informação Integrado Inteligente para apoio ao Profissional Fisioterapeuta: Construído sobre uma Base de Dados Simulados.* Dissertação de Mestrado em Tecnologia em Saúde – Pontifícia Universidade Católica do Paraná.

BREIMAN, L. 1996. *BAGGING predictors*. Machine learning, 24(2). 1996, pp. 123-140.

BUCHLI, F. 2003. *Detecting Software Patterns using Formal Concept Analysi.*2003. Universidade de Bern, Suiça.

CARPINETO, C., ROMANO, G. 2004. *Concept Data Analysis: Theory and Applications*. England: John Wiley & Sons, 2004.

CHEN, L., WRIGHT, P. e NEJDL, W. 2009. *Improving music genre classification using collaborative tagging data*. Proceedings of the Second ACM International Conference on Web Search and Data Mining. 2009, pp. 84-93.

CIMIANO, P., HOTHO, A. e STAAB S. *Learning Concept Hierarchies from Text Corpora ussing Formal Concept Analysis.* 2005. Universidade de Kassel Wilhelmshoher Allee, Alemanha.

CIOS, K. J., et al. 2007. *Data Mining: A Knowledge Discovery Approach.* s.l. : Springer, 2007. p. 606. ISBN: 978-0-387-33333-5.

DIAMANTIDIS, N.A., KARLIS, D. e GIANKOUMAKIS, E.A. 2000. *Unsupervised stratification of cross-validation for accuracy estimation.* Artificial Intelligence, 116. 2000, pp. 1-16.

FAYYAD, U., PIATETSKI-SHAPIRO, G. e PADHRAIC, P. 1996. The KDD Process for Extracting Useful Knowledge from Volumes of Data. *Communications of the ACM.* 1996, pp. 27-34.

FAYYAD, U. M. 1996. Data mining and knowledge discovery: making sense out of data. *IEEE Expert.* 5, 1996, Vol. 11, pp. 20-25.

GRANDVALET, Y. Bagging Equalizes Influence**.** *Machine Learning*, 55(3): 251-270, 2004.

GRAY, R.M. Entropy and information theory. *Springer Verlag*, New York, 1990.

HALL, M. A. 2000. Correlation-based feature selection for discrete and numeric class machine learning. *Proc. of the 17th Int. Conf. on Machine Learning.* 2000, pp. 359-366.

HAN, JIAWEI e KAMBER, MICHELINE. 2006. *Data Mining: Concepts and Techniques.* Second Edition. San Franciso, CA : Morgan Kaufmann, 2006. p. 772.

HUAN, LIU e MOTODA, HIROSHI. 1998. *Feature Selection for Knowledge Discovery and Data Mining.* s.l. : Kluwer Academic Publishers, 1998.

KALOS, A. e REY, T. 2005. *Data mining in the chemical industry.* Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining. 2005, pp. 763-769.

KOHAVI, R. 1995. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection**.** *International Joint Conference on Artificial Intelligence (IJCAI) p. 1137-1145.*

KOHAVI, R. e JOHN, G. 1996. Wrappers for Feature Subset Selection. *AIK special issue on relevance.* 1996.

KOLLER, D. e SAHAMI, M. 1996. Toward optimal feature selection. *Proc. of the 13th Int. Conf. on Machine Learning.* 1996, pp. 284-292.

LEE, H. D., MONARD, M. C. e BARANAUSKAS, J. A. 1999. *Empirical comparison of wrapper and filter approaches for feature subset selection.* São Carlos : ICMC-USP, 1999. Disponível em: ftp://ftp.icmc.usp.br/pub/BIBLIOTECA/rel_tec/RT_094.pdf.

LIPSCHULTZ, S., *LIPSON***, M. 1997.** *Matemática Discreta*. 2 ed. São Paulo: Bookman, 1997.

LIU, H.; MOTODA, H. 1998. Feature selection for knowledge discovery and data mining. *Kluwer*, 1998.

LIU, HUAN e YU, LEI. 2005. Toward Integration Feature Selection Algorithms for Classification and Clustering. *IEEE Transactions on Knowledge and Data Engineering.* 2005, pp. 491-502.

LOPES, LUCELENE. 2007. *Aprendizagem de máquina baseada na combinação de classificadores em bases de dados da área de saúde.* Curitiba : s.n., 2007. Dissertação de Mestrado em Tecnologia em Saúde – Pontifícia Universidade Católica do Paraná.

MALOOF, MARCUS A. e MICHALSKI, RYSZARD S. 2000. *Selecting Examples for Partial Memory Learning.* s.l. : Machine Learning Journal, 2000. pp. 27-52. Vol. 41, citeseer.ist.psu.edu/maloof00selecting.html.

MITCHELL, T. 1997. *Machine Learning.* New York : McGraw-Hill, 1997.

NILSSON, N.J. 1996. *Introduction to machine learning.* Stanford : Stanford university, 1996.

OCKHAM, W. 1999. *Prólogo da Exposição dos Oitos Livros da Física.* São Paulo : Nova Cultural, 1999.

PRATI, R. C. 2006. Novas abordagens em aprendizado de máquina para a geração de regras, classes desbalanceadas e ordenação de casos. São Carlos : s.n., 2006. p. 191. Tese de doutorado.

POELMANS, J., ELZINGA, P., VIAENE, S. e DEDENE, G. *Formal Concept Analysis in Knowledge Discovery: a Survey.* 2010. Faculty of Business and Economics, Bélgica.

QUINLAN, J. R. 1993. *C4.5: Programs for machine learning.* San Francisco : Morgan Kaufman, 1993.

QUINLAN, J. R. 1986. *Induction of decision trees.* Machine Learning, 1(1): 81-106, 1986.

QUINLAN, J. R. 1987. *Generation Production Rules from Decision Trees.* s.l. : In Proc. of IJCAI 87, 1987. pp. 304-307.

QUINLAN, J. R. 1996. Improved Use of Continuous Attributes in C4.5. *Journal of Artificial Intelligence Research.* 1996, Vol. IV, pp. 77-90.

QUINLAN, L. R. 1996. Bagging, Boosting and C4.5. *Proceedings of the Thirteenth National Conference on Artificial Intelligence and Eighth Innovative Applications of Artificial Intelligence Conference*, AAAI 96, IAAI 96, August 4-8, 1996, Portland, Oregon - Volume 1. AAAI Press / The MIT Press, 1996.

ROMÃO, W. 2002. Descoberta de Conhecimento Relevante em Banco de Dados sobre Ciência e Tecnologia; Tese – Doutorado – Engenharia da Produção – Universidade Federal de Santa Catarina, 2002.

SEBBAN, M.; NOCK,R. e LALLICH, S., 2001. *Boosting Neighborhood- Based Classifiers.* Proceedings of the Eighteenth International Conference on Machine Learning, San Francisco: Morgan Kaufmann, 505-512.

SILVA, J. P. D. 2007. *Agoritmos de Classificação baseados em Análise Formal de Conceitos.* Belo Horizonte : s.n., 2007. Dissertação de Mestrado em Ciência da Computação – Universidade Federal de Minas Gerais.

TAN, M., STEINBACH, V. e KUMAR, A.W. 2006. *Introduction to Data Mining.* Minnesota : Addison Wesley, 2006.

VALTCHEV, P., MISSAOUI R. e GODIN R. *Formal Concept Analysis for Knowledge Discovery and Data Mining: The New Chalenges*. 2003. Universidade de Montreal, Canada.

VIMIEIRO, R., VIEIRA, N. J. 2007. Uma análise de algoritmos para extração de regras de assosiação usando Análise Formal de Conceitos. *III Workshop em Algoritmos e Aplicações de Mineração de Dados da Universidade de Federal de Minas Gerais*. 2007.

WEKA. 2008. Data Mining with Open Source Machine Learning Software in JAVA. *WEKA.* [Online] 2008. http://www.cs.waikato.ac.nz/ml/weka/ Acesso em: 15/01/2008.

WILLE R. 1982. Restructuring Lattice Theory, RIVAL I., Ed., Symposium on Ordered Sets, University of Calgary, Boston, 1982, p. 445-470.

WITTEN, I.H e FRANK, E. 2005. *Data Mining: Practical machine learning tools and techniques.* 2 ed. San Francisco : Morgan Kaufmann, 2005.

WITTEN, I.H; FRANK, E. 2000. Data mining: practical machine learning tools and techniques with Java implementations, *Morgan Kaufmann*, 2000.

WOLFF, K. E. 1993. A first course in Formal Concept Analysis. *Esnst Schröder Zentrum für Begriffliche Wissensverarbeitung*. 1993.