INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA SUL-RIO-GRANDENSE - IFSUL, *CAMPUS* PASSO FUNDO
CURSO DE TECNOLOGIA EM SISTEMAS PARA INTERNET

VINÍCIUS PIERDONÁ LIMA

UMA ANÁLISE UTILIZANDO MINERAÇÃO DE DADOS PARA INFERÊNCIA ASSOCIADA AOS FENÔMENOS EL NIÑO, LA NIÑA E ANOS NEUTROS

Alexandre Tagliari Lazzaretti

PASSO FUNDO, 2011

VINÍCIUS PIERDONÁ LIMA

UMA ANÁLISE UTILIZANDO MINERAÇÃO DE DADOS PARA INFERÊNCIA ASSOCIADA AOS FENÔMENOS EL NIÑO, LA NIÑA E ANOS NEUTROS

Monografia apresentada ao Curso de Tecnologia em Sistemas para Internet do Instituto Federal Sul-Rio-Grandense, *Campus* Passo Fundo, como requisito parcial para a obtenção do título de Tecnólogo em Sistemas para Internet.

Orientador : Alexandre Tagliari Lazzaretti

PASSO FUNDO, 2011

A minha mãe, minha irmã

e minha namorada

por serem a razão da minha vida.

AGRADECIMENTOS

Devo sinceros agradecimentos primeiramente a minha mãe, Eliana Maria Pierdoná, por ter me ensinado as coisas mais importantes da vida, e por ter feito isso com toda a paciência que somente uma mãe pode dedicar a um filho. A minha irmã, Juliana Pierdoná Lima, por me ensinar sobre responsabilidade e sobre humildade. A minha namorada, Ana Caroline Secco, pela compreensão e pelo amor incondicional. Ao meu pai, Luis Carlos Barriquel Lima, por me ensinar a persistir. Aos meus grandes amigos Guilherme A. Borges, Marcos Vinícius e Matheus Fiebig, por me ensinaram tudo o que somente amigos podem ensinar. Aos meus colegas e grandes amigos, Thomaz C. Xavier e Vinícius C. Reck, por me mostrar que eu não estava sozinho, Norton, Giuseppe, Luiz, Thiago, José Alcebides e Samuel, pela amizade e companheirismo. Ao meu orientador, Alexandre T. Lazzaretti e a professora Anubis G. Rossetto, pela paciência sem fim. E aos meus professores, todos eles, sem eles eu não seria nada.

.

"Feliz de quem pôde conhecer o mistério do mundo!"

Geórgicas, de Virgílio

RESUMO

O presente trabalho apresentara um estudo sobre os algoritmos com maior afinidade na tarefa de mineração de dados em bases climatológicas, além de realizar o processo de mineração de dados a fim de inferir padrões que classifiquem os fenômenos El Niño, La Niña e os anos neutros. Para isso utilizou-se de um conjunto de bases de dados para testes das seguintes características: acurácia prevista, escalabilidade, robustez e velocidade. Após execução e análise das características definidas chegou-se a quatro algoritmos com desempenho igual ou superior a 70%, são eles: *AdaBoost1M*, *OneR*, *DecisionStump* e *SimpleCart*. Então foi feita a inferência dos modelos por eles apresentados e implementação, em linguagem Java, destes modelos para classificação de futuras entradas. Os algoritmos indicam que a radiação solar do mês tem novembro (outono), e a média da temperatura minima do mês de janeiro pode prever o tipo de fenômeno climático. Concluiu-se, também, que os modelos apresentariam maior taxa de assertividade em relação à predição dos fenômenos estudados caso estes abrangessem um maior número de atributos climáticos.

Palavras-chave: mineração de dados; descoberta de conhecimento; classificação.

ABSTRACT

This article will present a study about algorithms with a greater compatibility in the task of data mining in climatological basis, besides performing the data mining process in order to infer patterns that rate the phenomena El Niño, La Niña and neutral years. For this, it was used a group of databases to test the following characteristics: provided accuracy, scalability, hardiness and velocity. After the execution and analysis of the defined characteristics, four algorithms, with a performance equal or superior to 70%, were reached: *AdaBoost1M, OneR, DecisionStump and SimpleCart.* And so, it was done the inference of the models presented by them and the implementation, in Java Language, of these models for the classification on future entries. The algorithms indicate that the solar radiation of November (Fall), and the average of the minimum temperature of January can foresee the type of the climatic phenomenon. It was also concluded that the models would present a greater rate of accuracy in comparison to the precognition of the phenomena studied, if these encompassed a greater number of climatic attributes.

Keywords: data mining; knowledge-discovering; classification.

LISTA DE TABELAS

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

DCBD – Descoberta de Conhecimento em Bancos de Dados, p.14

MD – Mineração de Dados, p.20

WEKA – *Waikato Environment for Knowledge Analysis*, p.22

ARFF – *Atribute-Relation File Format*, p.23

IAPAR – Instituto Agronômico do Paraná, p.27

INPE – Instituição Nacional de Pesquisa Espacial, p.29

SUMÁRIO

1 INTRODUÇÃO

Por caracterizar-se como uma ciência em constante mudança, desde sua criação a produção tecnológica, em especial a tecnologia da informação, sofre mutações evolutivas cada vez mais frequentes, conforme exposto por Han e Kamber, “Desde a década de 1960, os bancos de dados e a tecnologia da informação vêm evoluindo sistematicamente de primitivos sistemas de leitura de arquivo até sofisticados e poderosos sistemas de bancos de dados” (Ha e KAMBER, 2006, p.3).

Juntamente com esta evolução tecnológica veio a consequente baixa dos preços de manutenção e aquisição desta tecnologia, propiciando a acumulação de informações por parte de organização e instituições, informação essa antes de difícil observação.

A oportunidade trazida pela evolução tecnológica modificou os cenários econômicos, criando a necessidade de que gerentes e administradores tomem decisões rápidas, como ressalta Hebarmann

> “O desenvolvimento de uma estratégia é em essência o desenvolvimento de uma fórmula ampla que norteará o modo como uma empresa irá competir, quais serão suas metas a curto, médio e longo prazo, e quais serão as políticas necessárias para o cumprimento destas metas. Para tanto, a informação precisa estar disponível para as pessoas certas, no formato esperado, no momento e local desejados.” (2011, p.1).

Nascidos dessa necessidade, o processo de descoberta de conhecimentos em banco de dados e consequentemente a mineração de dados, tornam, desde 1989, os grandes aglomerados de dados em conhecimento utilizável, tanto para administradores quanto para pesquisadores, já que estas bases servem, diversas vezes, como embasamento ou parte do desenvolvimento de diversas pesquisas cientificas.

Essas mudanças nos cenários provocada pela tecnologia afetou também uma das práticas mais antigas da humanidade, a observação do clima. Para sobreviver, os seres humanos aprenderam desde muito tempo a se adaptarem ao clima e as estações do ano, adaptação essa que ditou costumes e tradições. No entanto o clima e as estações não se comportam de maneira igual todos os anos, algumas vezes as águas tropicais do Pacífico e grandes massas de ar do globo mudam seu comportamento e força uma mudança na rotina dos seres que habitam o globo terrestre (Babkina, 2003).

Dentre os fenômenos, pode-se citar o El Niño e a La Niña que possuem relevância considerável para os povos que vivem banhados pelo Pacífico, em especial na área tropical do planeta, e por isso são monitorados e previstos com antecedência.

Existem diversos algoritmos disponíveis para que seja realizado o processo de mineração de dados, no entanto muitos destes não apresentam bons resultados quando aplicados sobre problemas específicos, como o caso de dados provenientes de bases climatológicas.

Dessa maneira, esta pesquisa vai apresentar um estudo sobre os algoritmos que melhores se encaixam na tarefa de mineração de dados em bases climatológicas, que possuem relação intrínseca com a meteorologia e consequentemente na previsão de fenômenos naturais, e seus resultados para que a escolha destes algoritmos deixe de ser arbitrária e passe a ter um caráter científico.

Além de realizar o processo de mineração de dados sobre um conjunto de dados a fim de inferir padrões que classifiquem os fenômenos El Niño, La Niña e os anos neutros.

O trabalho apresentará em seu segundo capítulo o desenvolvimento geral, separado por tópicos, iniciando na descrição da importância dos dados climatológicos e descrevendo os fenômenos usados neste trabalho, seguido pela descrição do processo de descoberta de conhecimento, tema central do trabalho, suas fases e, em um tópico separado, o processo de mineração de dados e a descrição das tarefas e técnicas deste. Então será apresentada a ferramenta utilizada para gerar os modelos, a interface disponibilizada por ela, bem como o formato utilizado para a entrada dos dados, seguida pela apresentação das bases de testes.

Iniciar-se-á os testes e resultados do trabalho, começando pela decisão da técnica de mineração de dados e sua descrição, as famílias de algoritmos testados e seus resultados de acurácia e desempenho, juntamente com considerações acerca de cada um.

Por fim teremos a descrição detalhada do algoritmo com melhor desempenho, a apresentação dos modelos inferidos e a implementação dos algoritmos que fazem a classificação futura dos dados.

1.1 MOTIVAÇÃO

As bases de dados climatológicas, normalmente alimentadas por instituições de pesquisas com frequência diária, ou ainda maior, tornam-se, progressivamente, grandes aglomerados de dados. Estes aglomerados, apresentados de forma não facilmente compreensível para os seres humanos, devem ser transformados em conhecimento real e

.

utilizável, e para isto, apresentam-se as técnicas de descoberta de conhecimento em bancos de dados (DCBD).

Os dados, após serem submetidos pelos diversos procedimentos inertes ao processo de DCBD, serão apresentados na forma de conhecimento compreensível aos seres humanos que podem ser aproveitados para, por exemplo, a previsão de dados climatológicos futuros ou percepção de frequência de certos fenômenos da natureza, como o El Niño, a La Niña, ou mesmo os anos em que nenhum destes fenômenos se manifeste. Auxiliando produtores e pesquisadores de diversas formas.

1.2 OBJETIVOS

Este trabalho tem como objetivos inferir padrões, ou modelos, que expliquem os fenômenos El Niño, La Niña e anos neutros com base em um banco de dados e selecionar com base em critérios que serão definidos um ou mais algoritmo dentro dos testados como mais propício a esta tarefa.

1.2.1 Objetivo Geral

Inferir perfis e padrões úteis a partir de bases de dados climatológicas utilizando mineração de dados, além de eleger o melhor algoritmo para este processo.

1.2.2 Objetivos específicos

- Selecionar um conjunto de dados sobre os quais a descoberta será realizada;
- Filtrar, se necessário, os dados, utilizando técnicas de redução de dimensionalidade;
- Comparar o desempenho dos algoritmos testados e eleger um ou mais para inferir os modelos
- Selecionar técnicas de extração de conhecimento para as tarefas;
- Analisar padrões de fenômenos naturais, como anos de El Niño, La Niña e anos neutros.

.

2 REFERÊNCIAL TEÓRICO

Nos próximos capítulos segue-se a discussão sobre os temas e assuntos que compõem o conhecimento inicial e necessário para o trabalho.

2.1 DADOS CLIMATOLÓGICOS

O conjunto de dados das variáveis climatológicas são apresentados, historicamente, como constituintes tanto de hipóteses como desenvolvimentos científicos de diversas áreas. Nesta perspectiva, aliado a necessidade recorrente de dados exatos e confiáveis, para que exista embasamento matemático e correspondência na realidade, tão preciosos para o desenvolvimento científico, os dados das variáveis climatológicas demonstram sua necessidade.

Oliveira (2009) discorre diversas vezes sobre a importância apresentada pelos dados climatológicos em um artigo apresentado à revista Caminhos da Geografia. Segundo ele, os dados climatológicos, assim como as questões climáticas, estão presentes em diversos trabalhos realizados nos muitos programas de pesquisa. Oliveira torna claro a inegável necessidade de dados climatológicos para as pesquisas científicas quando diz que:

> "Entendemos que o conjunto de dados das variáveis climáticas é fundamental aos estudos que tratem sobre a temática que envolve o clima e seus atributos, não só na Geografia, mas nas diversas áreas do conhecimento, como a medicina, a agronomia, a arquitetura, entre outras. Apenas a partir de um conjunto de dados diversificados e confiáveis os pesquisadores podem dar sustentação aos seus estudos, que apoiados em um vasto campo teórico-conceitual criam condições para se chegar a construção de novos caminhos metodológicos que contribuam para o desenvolvimento da sociedade." (2009, p.20).

A partir do entendimento de que os dados das variáveis climatológicas são parte constituinte de diversas pesquisas é interessante, também, ressaltar a importância de que estes dados estejam organizados e sejam confiáveis, daí a necessidade de bancos de dados que os armazenem. Ainda segundo Oliveira:

> "Notadamente essas pesquisas devem ser embasadas num minucioso, bem elaborado e completo banco de dados sobre o clima, que contenham as anotações das diversas variáveis climáticas (elementos meteorológicos) em uma escala espaço-temporal definida. Esse banco de dado é, sobretudo, a base de sustentação que vai definir os critérios de cientificidade da pesquisa e gerar condições para uma análise que culminará na elaboração de uma tese sustentada teórico e metodologicamente." (2009, p.10).

.

No entanto, por diversos motivos estes dados podem por vezes, não serem suficientes ou não apresentarem as informações na forma desejada. Neste panorama, talvez o processo de descoberta de conhecimento possa vir em auxílio, retornando, através de algoritmos, dados não triviais e que representem associações, por exemplo, antes de difícil percepção.

2.1.1 Fenômenos Meteorológicos – El Niño e La Niña

Segundo Babkina (2003) o termo El Niño foi adotado por pescadores da costas de Equador e Peru para referirem-se ao aquecimento da águas do Pacífico que ocorria próximo ao fim dos anos. Com o passar do tempo o termo tornou-se sinônimo de um evento que não apenas aquece as águas, mas também causa alterações nas variáveis climáticas locais, ou mesmo, mundiais.

Esse evento normalmente atinge toda a costa da América do Sul, as Ilhas Galápagos e, por vezes, chega a atingir uma faixa de mais de oito mil quilômetros ao longo da linha do Equador. Observando que sua intensidade pode alterar o seu grau de influencia, vide que alguns desses eventos tiveram pouco impacto na vida terrestre, por exemplo, o menor deles acarretou num aumento de apenas 2° Fahrenheit nas águas do Pacífico. No entanto nos anos em que o fenômeno apresenta-se forte, como em 1982 e 1983, não apenas as áreas diretamente afetas sofrem as consequências, mas sim, o globo terrestre inteiro (Babkina, 2003).

O processo inverso do apresentado anteriormente comumente é chamado de La Niña. Caracterizado pelo anormal esfriamento das águas do Pacífico, principalmente na região Equatorial, e a grande diferença entre a pressão atmosférica na superfície terrestre, devido a mudança na direção dos ventos. Esse fenômeno tem efeitos tão relevantes quanto o seu oposto e seu monitoramento é tão importante quanto.

2.2 O PROCESSO DE DESCOBERTA DE CONHECIMENTO

As últimas décadas vivenciaram um aumento progressivamente considerável na capacidade de gerar e armazenar dados. No entanto estes dados não representam necessariamente conhecimento, situação essa conhecida como “rica em dados, mas pobre em informações” (HAN e KAMBER, 2006, p.4), tornando necessário o desenvolvimento de técnicas e ferramentas que facilitem a conversão dos dados brutos, que não representam

.

informações “úteis”, em dados que representem conhecimento. Esta situação tornou a criação do processo de descoberta de conhecimento em banco de dados inevitável.

Segundo Dias (2002), o processo de DCBD surgiu em 1989 com o objetivo de encontrar fatos, conhecimento útil, existente em base de dados e enfatizar o alto nível das aplicações dos métodos de prospecção de dado. A partir de então o processo continuou a evoluir incorporando teorias, métodos e algoritmos de diversas áreas da computação e da matemática, como por exemplo, a inteligência artificial, o aprendizado de máquinas e a estatística (figura 01).

Figura 01

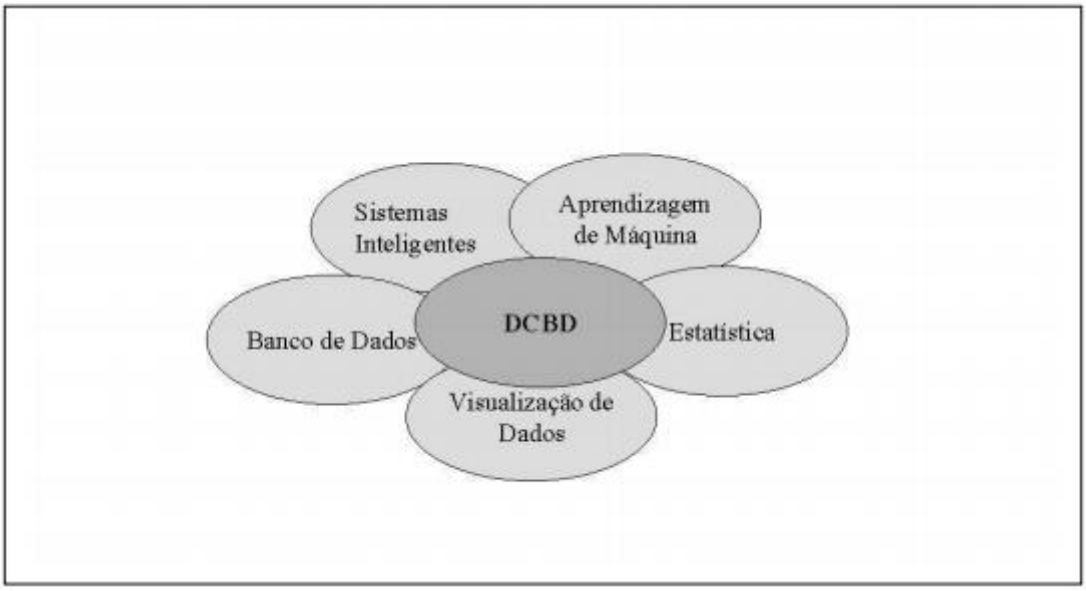

Figura 01 – Áreas convergente do processo de descoberta de conhecimento.
Fonte: Félix, 2008.

Segundo Silva, “DCBD é o processo não-trivial de extração de conhecimento a partir de um grande volume de dados” (2003, p.27). Processo esse, considerado não-trivial, por não ser apresentado de forma explicita ou por não poder ser extraído pelas técnicas de consulta ordinárias, como por exemplo, SQL.

Fayyad, Piatetsky-Shapiro e Smyth apresentam o processo de DCBD da seguinte maneira, “Descoberta de conhecimento em bases de dados é o processo de identificação de padrões válidos, novos, potencialmente úteis e compreensíveis em repositórios de dados”

.

(1996c). A partir desta proposição é possível abstrair algumas características intrínsecas ao processo de DCBD.

Neves (2003) define as características apresentadas como:

- Validade, que refere-se ao grau de certeza que os resultados devem apresentar.

- Novidade, refere-se a necessidade de que os resultados apresentem informações novas, não redundantes, com as já observadas por métodos ordinários.

- A expressão “potencialmente úteis” está relacionada a necessidade de que os resultados tenham, ou representem, utilidade para alguma área de conhecimento.

- A compreensibilidade corresponde ao entendimento do resultado obtido, para que facilite a compreensão do que não é evidenciado nos dados.

O termo processo implica a existência de várias fases constituintes (Dias, 2002). Estas fases são, normalmente, denominadas de seleção, pré-processamento, transformação, mineração de dados e pós-processamento (figura 02), no entanto é interessante ressaltar que outros autores apresentam diferentes nomenclaturas para a divisão de fases do processo de DCBD.

.

Figura 02

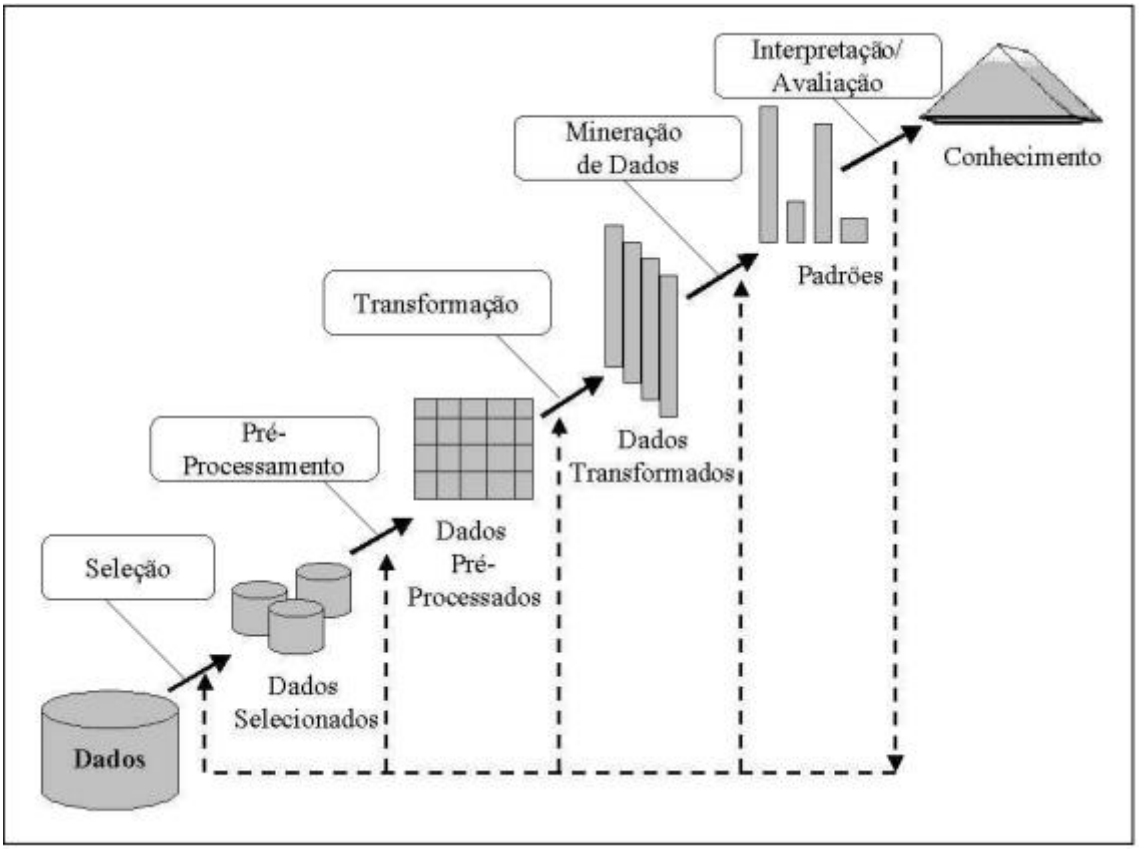

Figura 02 – Processos da descoberta de conhecimento em banco de dados.

Fonte: Fayyad, Piatetsky-Shapiro e Smyth, 1996c.

2.2.1 Fases do Processo de Descoberta de Conhecimento

As fases do processo de DCBD se caracterizam como interativas e iterativas. Isto é, necessitam da interferência de um técnico especialista, interatividade, e se repetem diversas vezes para se chegar a um resultado parcial que será utilizado na próxima repetição, iteratividade.

A fase de seleção é regularmente, definida como a fase onde os itens específicos de uma, ou mais bases de dados são selecionados sobre os quais a descoberta será efetuada, estes itens podem ser tanto um conjunto de dados como um subconjunto de atributos ou instâncias.

A fase de pré-processamento, ou fase de limpeza, segundo Neves (2003), consiste na adequação do conjunto de dados utilizando operações básicas como remoção de dados errôneos e manipulação de atributos ausentes. A importância desta fase para o processo de DCBD é facilmente notada ao se retomar a característica de validade do processo, já que

.

dados errôneos ou atributos faltantes podem alterar o resultado encontrado de modo a invalidar o processo por completo.

A fase de transformação "envolve métodos de transformação para reduzir o número efetivo de atributos relevantes para a representação da dependência em relação ao tipo de problema de MD" (NEVES, 2003, p.28). De maneira geral esta fase objetiva a projeção e redução dos dados sobre os quais a descoberta será realizada.

A fase seguinte é chamada de mineração de dados, definida como uma "etapa essencial do processo consistindo na aplicação de técnicas inteligentes a fim de se extrair os padrões de interesse" (AMO, 2003, p.2). Esta etapa é estudada mais detalhadamente a seguir.

Após a mineração dos dados é realizada a fase de pós-processamento ou interpretação dos resultados que objetiva a avaliação dos dados quanto a sua utilidade, interpretação destes com relação a elementos da realidade, verificação da generalidade dos resultados, se os resultados compreendem conhecimentos novos e por fim a validade destes para os objetivos pretendidos. (Silva, 2003) Além disto, nesta etapa os resultados podem ser convertidos, caso estes se apresentem em uma linguagem ilegível.

2.3 MINERAÇÃO DE DADOS

O termo "Mineração de Dados" é, muitas vezes, confundido com o processo de DCBD completo, no entanto a etapa de mineração de dados é apenas uma parte constituinte de um processo maior, que envolve desde a separação e limpeza dos dados à visualização e organização dos resultados. Fayyad, Piatetsky-Shapiro e Smyth (1996) corroboram isto ao afirmarem que DCBD refere-se ao processo global de descoberta de conhecimento a partir de dados, enquanto a mineração de dados é uma fase desse processo. Meira complementa dizendo que "a mineração de dados refere-se a aplicação de algoritmos específicos para extrair os padrões dos dados. As outras fases do processo são também importantes para se garantir que conhecimento útil seja derivado dos dados" (2008, p. 6).

Ressaltando a diferença entre as tarefas e técnicas em mineração de dados, Vianna define tarefa como sendo a "definição do que se está buscando, quais padrões têm interesse em encontrar ou qual padrão o surpreenderia." (2006, p.19), enquanto que técnica de mineração é descrita por Amo como a "especificação de métodos que nos garantam como descobrir os padrões que nos interessam" (2003, p.3), e Vianna complementa afirmando que pode-se aplicar "várias técnicas em um mesmo problema, ao mesmo tempo, podendo ainda

.

aplicar vários algoritmos da mesma técnica que permite obter um resultado mais preciso." (2006, p.19).

2.3.1 Tarefas e Técnicas de Mineração de Dados

De maneira geral as tarefas de mineração de dados podem ser separadas em duas categorias: descritiva e preditiva (figura 03). As tarefas descritivas se caracterizam pelas propriedades gerais encontradas nos dados, enquanto as tarefas preditivas fazem uso das variáveis já conhecidas do banco de dados para predizer padrões, ainda desconhecidos. Além disso, é interessante notar que várias tarefas podem ser designadas para o mesmo processo de descoberta, já que em algumas situações apenas uma destas tarefas pode não representar todos os resultados pretendidos (Han e Kamber, 2006).

Figura 03

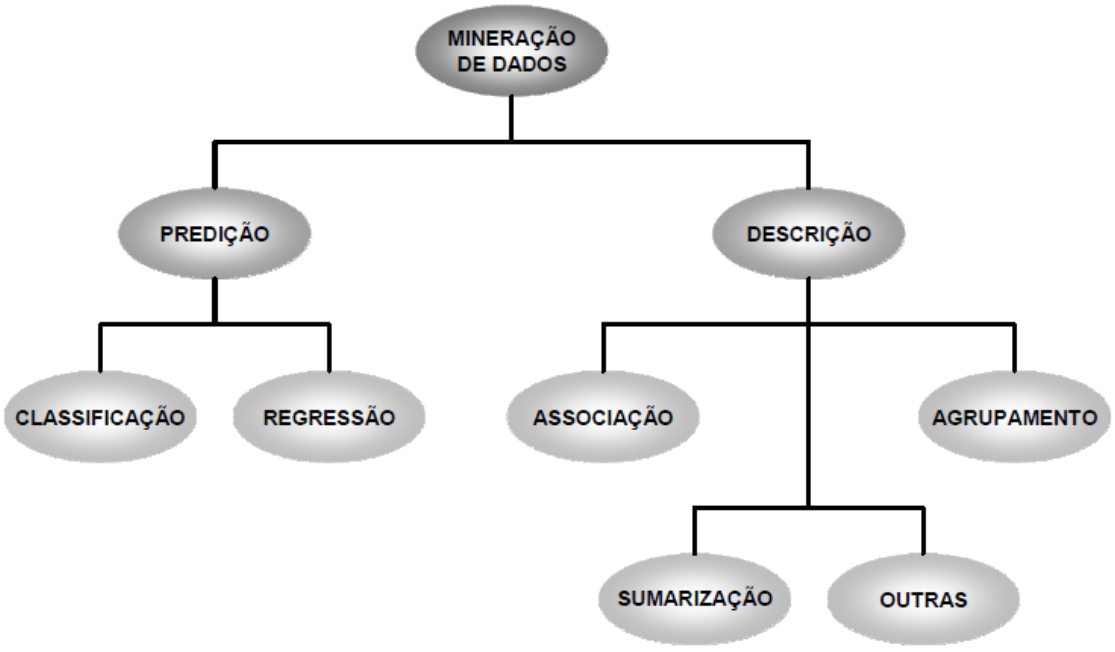

Figura 03 – Técnicas de mineração de dados.

Fonte: Adaptada de Rezende et. al; 2002, apud Meira, 2008.

Conforme visto na figura acima, estas tarefas dividem-se ainda em outras, mais específicas. Estas tarefas específicas estão listadas e descritas abaixo.

.

Classificação: Consiste em encontrar uma função que descreva e mapeie um item ou conjunto de dados novos para uma classe dentre algumas previamente definidas. A variável de predição é discreta ou categórica (Meira, 2008; Han e Kamber, 2006). Esta tarefa pode ser utilizada na classificação de um evento climático a partir de um modelo criado anteriormente.

Regressão: Segundo Meira a tarefa de regressão "consiste em descobrir uma função que mapeie um item de dados para uma variável de predição de valor numérico contínuo" (2008, p.8). Um exemplo comum para a descrição desta tarefa é o processo de colocar preço em uma casa. Basicamente funciona na comparação dos atributos da casa que pretendesse vender em comparação com os atributos das casas vendidas na vizinhança. A tarefa de regressão funciona de maneira similar.

Associação: É a descoberta de regras que indiquem um evento que tende a ocorrer caso um outro evento ocorra. Esta tarefa pode ser descrita na forma de $X \rightarrow Y$, onde X e Y são itens ou registros de dados (Meira, 2008; Vianna, 2006). Está tarefa pode ser descrita utilizando-se o famoso exemplo das cervejas e fraldas, onde o processo de DCBD descobriu que a compra de fraldas por homens levava a compra de cerveja em um determinado mercado.

Agrupamento: Também, comumente, chamada de *Clustering*, esta técnica trabalha agrupando os conjuntos de dados semelhantes, no entanto, diferentemente da tarefa de classificação, esta trabalha na identificação de classes de dados (Meira, 2008; Vianna, 2006). Pode-se utilizar esta tarefa a fim de encontrar aglomerados de clientes com um comportamento específico.

Sumarização: Meira descreve esta tarefa como os "meios de encontrar uma descrição compacta para um subconjunto de dados, como, por exemplo, derivação de regras resumidas ou visualização multivariada" (2008, p.8).

É interessante ressaltar que uma tarefa de mineração de dados pode possuir mais de uma técnica aplicada a ela e que não existe uma técnica melhor que as demais genericamente, a escolha da técnica envolve uma análise profunda do problema proposto. As técnicas mais populares são: árvores de decisão, regras de classificação, redes neurais, análise de cesta de mercado, vizinhos mais próximos, regressão linear ou não linear e algoritmos genéticos (Meira, 2008; Han e Kamber, 2006).

2.4 WEKA (WAIKATO ENVIRONMENT FOR KNOWDLEDGE ANALISES)

Desenvolvida pela Universidade de Waikato na Nova Zelândia, escrito em Java e distribuída sobre a licença GNU, a *Waikato Environment for Knowledge Analysis* (WEKA)

.

tem como objetivo prover uma maneira de implementar as principais técnicas de mineração de dados, compreendendo seus mais conhecidos algoritmos (Figura 04).

Figura 04

Figura 04 – Tela inicial da ferramenta WEKA.

Além disso, a ferramenta disponibiliza uma interface gráfica, chamada de *Explorer*, que prove acesso a todas as suas principais funcionalidades, como por exemplo, a fácil leitura dos arquivos no formato ARFF (*Attribute-Relation File Format*) e a construção de árvores de decisão sobre os resultados dos algoritmos rodados (Witten, Frank e Hall, 2011).

No entanto a interface gráfica do *Explorer* apresenta uma seria desvantagem, já que mantém todas as suas atividades na memória principal, impossibilitando assim sua utilização para a resolução de problemas de grande porte (Witten, Frank e Hall, 2011).

A ferramenta ainda apresenta outras duas interfaces gráficas orientadas a facilitar a resolução de problemas mais específicos. A *Knowledge Flow* disponibiliza caixas de dados e algoritmos que podem ser arrastados, conectados e arranjados, e a *Experimenter*, desenvolvida para auxiliar a decisão de qual método ou parâmetros usar em algoritmos de classificação e regressão. Por fim a ferramenta ainda apresenta uma forma textual, com acesso a todos os recursos do sistema (Witten, Frank e Hall, 2011; Waikato, 2008).

Por ser mais simples e com larga documentação, será utilizada neste trabalho a interface *Explorer*.

.

2.4.1 Execução dos algoritmos através do WEKA

A *Explorer* oferece acesso gráfico a todas as principais ferramentas do sistema e possibilita um fácil acesso a maior parte das técnicas de mineração de dados disponibilizadas pela Weka em um menu superior (Figura 05) (Witten, Frank e Hall, 2011; Waikato, 2008).

Figura 05

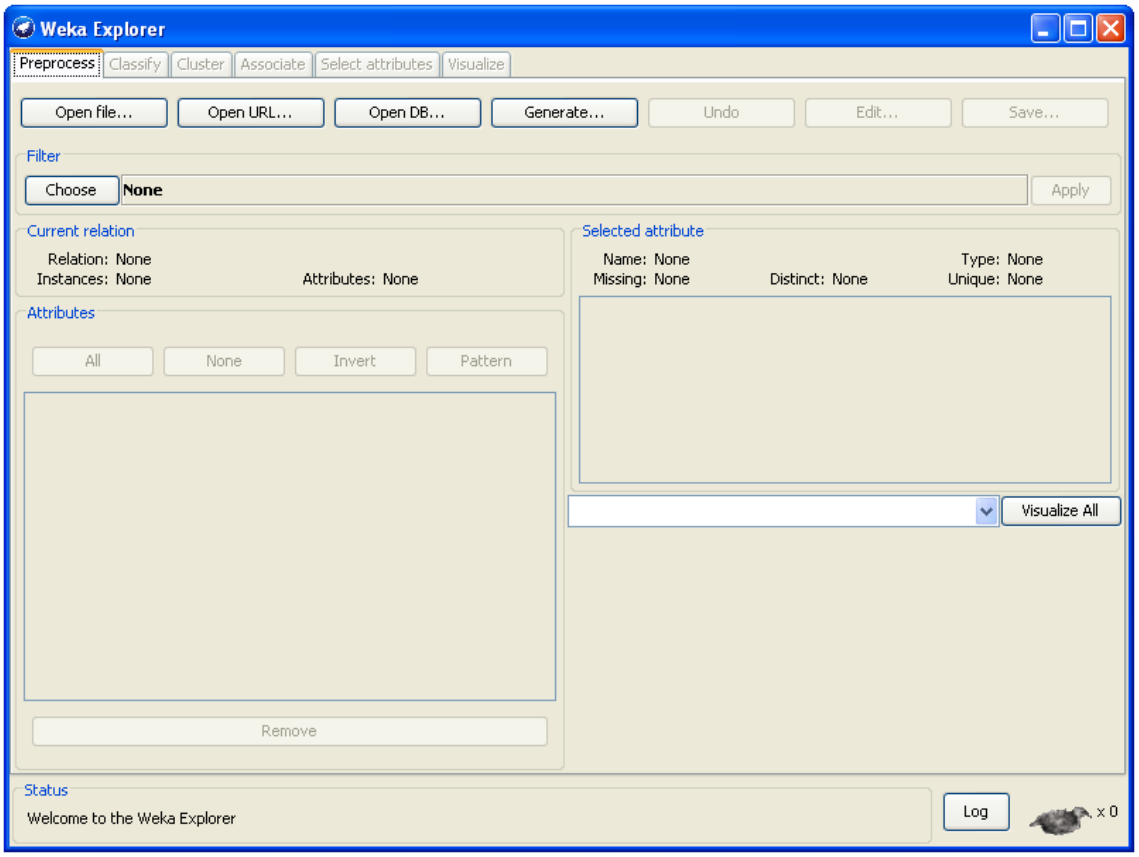

Figura 05 – Tela inicial da interface gráfica do *explorer*.

Inicialmente, em *Preprocess*, deve ser feita a escolha do conjunto de dados, prioritariamente em formato ARFF apresentado mais adiante, podendo nesta aba, ainda, ser feita a escolha dos atributos que farão parte do processo. Então na aba *Classify* (figura 06) deve ser feita a escolha do algoritmo através do botão *Choose*, observando que os algoritmos estarão separados em suas respectivas famílias e que a opção *Cross-validation* deve ser mantida marcada. Em seguida deve-se ser feita a escolha da variável *class label*[1]. Para finalizar o processo basta clicar em *Start*, o resultado deve sair na área *Classifier output*.

.

Figura 06

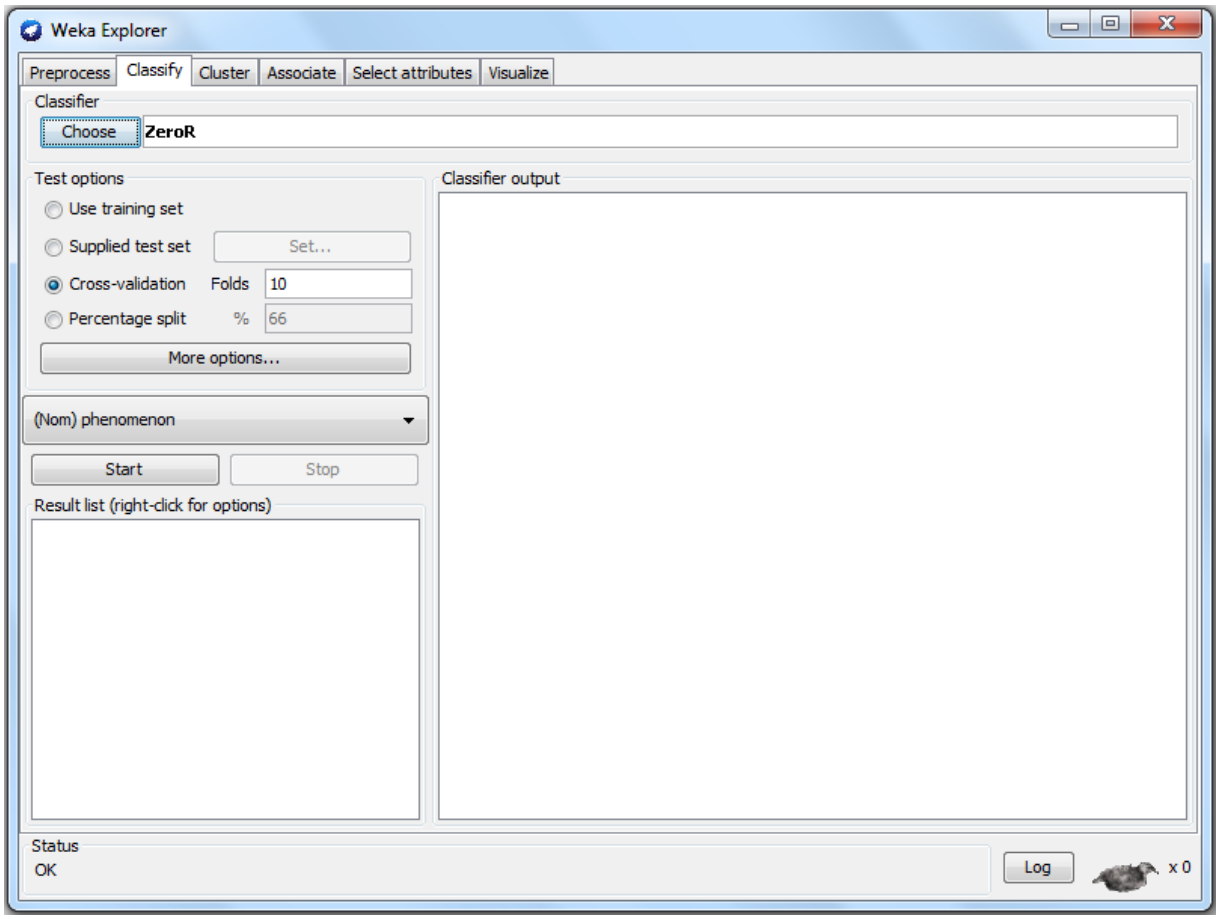

Figura 06 – Aba *Classify da ferramenta WEKA.*

2.4.2 O Formato Attribute-Relation

O *Attribute-Relation File Format* (ARFF) é um arquivo de texto que descreve e relaciona dados aos atributos. Foi desenvolvido pelo projeto *Machine Learning* do Departamento de Ciências da Computação da Universidade de Waikato especialmente para ser utilizado em conjunto com a ferramenta (Waikato, 2008).

É constituído por duas partes distintas, pelo cabeçalho seguido pelos dados. O cabeçalho é constituído, principalmente, pelo nome da relação dos dados, pelos atributos ou colunas e pelos seus tipos (Figura 07). A segunda parte é composta pelos dados, organizados em linha e separados por vírgula (Figura 08) (Waikato, 2008).

.

Figura 07

```
@RELATION nome_da_relacao

@ATTRIBUTE valor1   NUMERIC
@ATTRIBUTE valor2   NUMERIC
@ATTRIBUTE valor3   NUMERIC
@ATTRIBUTE valor4   NUMERIC
@ATTRIBUTE class    {valor5,valor6,valor7}
```

Figura 07 – Exemplo de cabeçalho de um arquivo ARFF.

Figura 08

```
@DATA
5.1,3.5,1.4,0.2,valor5
4.9,3.0,1.4,0.2,valor5
4.7,3.2,1.3,0.2,valor5
4.6,3.1,1.5,0.2,valor6
5.0,3.6,1.4,0.2,valor6
5.4,3.9,1.7,0.4,valor6
4.6,3.4,1.4,0.3,valor6
5.0,3.4,1.5,0.2,valor6
4.4,2.9,1.4,0.2,valor7
4.9,3.1,1.5,0.1,valor7
```

Figura 08 – Exemplo de dados em um arquivo ARFF.

O nome da relação de dados é apresentado na primeira linha e marcado por @RELATION, devendo ser uma *string* circundada por parênteses caso existam espaços nela. Cada um dos atributos deve iniciar pelo marcador @ATTRIBUTE seguido pelo nome do atributo e o tipo. A ordem em que os atributos são declarados deve ser respeitada quando são relacionados os dados e o WEKA espera que todos os dados estejam presentes na relação (Waikato, 2008).

3 DESENVOLVIMENTO

A seguir tem-se o desenvolvimento do trabalho, as etapas de sua construção, os resultados e a discussão destes.

.

3.1

3.2 MATERIAIS E MÉTODOS

Para que este trabalho pudesse ser realizado foi, inicialmente, necessário gerar uma base de dados para testes. Esta base foi gerada e alimentada pelo Instituto Agronômico do Paraná (IAPAR) e contem uma série histórica de dados observados validados, ou seja, sem erros, diários de 1980 a 2009. A partir desta base foi gerada uma segunda e uma terceira base de dados a fim de criar a possibilidade de teste de todas as características desejadas, de forma que os dados testados fossem os mesmo, mas apresentados de formas diferentes.

Foi necessário adotar um método que pudesse ser aplicado na comparação dos algoritmos selecionados para teste. O método escolhido é uma adaptação do descrito por Han e Kamber (2006), que apresentam cinco características bem definidas para a comparação entre algoritmos de classificação, são elas:

- Acurácia Prevista: Talvez o ponto mais importante na comparação entre os algoritmos, a acurácia é definida como a habilidade do algoritmo de classificar assertivamente novos dados quando aplicados ao modelo gerado.
- Velocidade: Referente ao tempo que o algoritmo leva para apresentar os resultados. De uma maneira geral esse aspecto não teve grande importância neste trabalho, já que a grande maioria dos algoritmos não apresentaram tempo de execução considerável.
- Robustez: Refere-se a assertividade do modelo quanto aos dados incorretos, inexistentes ou com ruído.
- Escalabilidade: Referente ao desempenho do algoritmo, ou o número de recursos computacionais usados, quando aplicados a grandes conjuntos de dados.
- Interpretabilidade: Refere-se ao nível de compreensibilidade apresentado pelo modelo gerado. Respeitando as características da fase de pós-processamento, essa interpretação do modelo gerado deve ser analisada por alguém com conhecimento na área de aplicação deste.

Neste trabalho foi desconsiderada a característica da interpretabilidade, já que seria inviável levar a analise de um especialista todos os testes feitos, em decorrência do grande número destes. No entanto esta característica pode ser explorada em trabalhos futuros.

Após as bases de dados terem sido criadas e o método de comparação de algoritmos ter sido escolhido foi necessário eleger a tarefa de mineração de dados que mais fosse propícia. Devido ao potencial de utilidade das bases de dados climáticas para a área de

.

meteorologia e observando que a previsão de eventos climáticos é um característica marcante desta, foi analisada uma tarefa de mineração de dados que pudesse classificar fenômenos a partir de dados fornecido por estações meteorológicas, de modo que os algoritmos de classificação mostraram-se mais adequados.

Então, passou-se ao processo de execução e análise dos algoritmos de classificação. Foram testados 64 algoritmos de seis famílias diferentes e seus resultados foram registrados e comparados para que se pudesse eleger os que obtivessem maiores probabilidades de retornar modelos de classificação úteis.

Os resultados de acurácia foram organizados em tabelas para facilitar a compreensão e comparação.

Depois de feita a eleição dos algoritmos, fez-se a descrição dos modelos de classificação inferidos por eles para explicar os fenômenos El Niño, La Niña e de anos neutros, bem como foram executados testes em bases de dados associadas a região norte e região sul do estado. Pois essas regiões possuem características distintas de clima, com possibilidade de alteração no comportamento dos algoritmos testados.

Por fim, ainda, foi feita uma implementação destes modelos em linguagem Java para a classificação de novas entradas, tornando, assim, os modelos utilizáveis.

3.3 RESULTADOS E DISCUSSÕES

Nos próximos tópicos far-se-á a apresentação detalhada dos resultados obtidos e a discussões sobre estes, além de descrever de forma geral o funcionamento dos algoritmos através de suas famílias.

3.3.1 Bases de dados para testes

As bases de dados utilizadas nos testes executados sobre os algoritmos são, na verdade formas de visualização de uma mesma base, ou seja, todas possuem as mesmas informações, mas dispostas de maneira diferente, assim pode-se analisar o comportamento dos algoritmos sem correr o risco das bases possuírem algum tipo de especificidade diferente da testada que alterasse os resultados.

.

A base de dados que originou as demais é alimentada por 28 estações espalhadas pelo estado do Paraná e foram disponibilizadas pelo IAPAR através do projeto SIMCAFE[1], de forma que os dados são verificados e não possuem erros. De modo que representam 30 anos (1980 a 2009) de dados climatológicos diários coletados. Observando que, através de dados fornecidos pela Instituição Nacional de Pesquisa Espacial (INPE), foi possível relacionar cada ano da base de dados a fenômeno meteorológico (El Niño, La Niña ou ano neutro), conforme pode ser visto na tabela 01 abaixo.

Tabela 01 – Relação ano/fenômeno

Ano	Fenômeno
1980	Neutro
1981	Neutro
1982	El Niño
1983	La Niña
1984	La Niña
1985	Neutro
1986	El Niño
1987	Neutro
1988	La Niña
1989	Neutro
1990	El Niño
1991	Neutro
1992	Neutro
1993	Neutro
1994	El Niño
1995	La Niña
1996	Neutro
1997	El Niño
1998	La Niña
1999	Neutro
2000	Neutro
2001	Neutro
2002	El Niño
2003	Neutro

[1] Projeto que envolve instituições do sul do Brasil, fincanciado pelo FINEP.

.

2004	El Niño
2005	Neutro
2006	El Niño
2007	La Niña
2008	Neutro
2009	El Niño

Cada estação apresenta, diariamente, as seguintes variáveis climatológicas: temperatura mínima, temperatura máxima, umidade, radiação solar e precipitação (figura 09), de forma que as bases de dados apresentam diversos registros, facilitando o processo de descoberta de conhecimento.

Para que seja possível testar a relação e o comportamento dos algoritmos quanto ao sua assertividade, foi criada uma segunda base de testes composta pelos dados da média dos valores dos atributos (temperatura máxima, mínima, umidade...) por ano e separados pelos meses (figura 10). Por exemplo, na primeira tupla, a segunda coluna, nomeada por tmaxjan, apresenta os dados da média da temperatura máxima do mês de janeiro no ano de 1980.

Figura 10

	year integer	tmaxjan double precision	tmaxfev double precision	tmaxmar double precision	tmaxabr double precision	tmaxmai double precision
1	1980	28.7461981566	28.5868226600	30.3308755760	27.3796428571	24.4539170506
2	1981	29.0956221198	30.1784438775	29.1070276497	26.5351190476	25.8846774193
3	1982	29.2387096774	29.1542091836	28.1783410138	26.4539285714	23.5936635944
4	1983	29.7644009216	29.3235969387	27.5023041474	25.8539285714	23.0953917050
5	1984	30.6467741935	32.0852216748	28.6802995391	25.2733333333	25.3603686635
6	1985	29.7326036866	29.8980867346	28.5592165898	26.5560714285	24.2366359447
7	1986	30.9380184331	28.8900510204	28.4989631336	27.26	24.0169354838
8	1987	30.2512672811	27.9607142857	29.4580645161	27.0808333333	20.8021889400
9	1988	30.9518433179	27.7513546798	30.3473502304	26.0782142857	21.2118663594
10	1989	27.4394009216	28.2511479591	28.5536866359	27.1246428571	23.2291474654
11	1990	28.2686635944	29.9396683673	29.6647465437	27.8586904761	22.6055299539
12	1991	29.8993087557	29.6700255102	28.1802995391	26.8221428571	24.3429723502
13	1992	30.5798387096	29.7162561576	27.4799539170	25.4986904761	22.8744239631
14	1993	29.8455069124	27.3446428571	29.0964285714	27.7103571428	23.7304147465
15	1994	28.8319124423	29.7742346938	28.025	26.5782142857	24.5891705069
16	1995	29.3482718894	28.5710459183	28.3505760368	25.9655952380	23.4911290322

Figura 10 – Amostra da base de dados 01.

.

Essa base de dados é denominada de base 01 em referências futuras durante este trabalho.

Foram feitos testes de acurácia, ainda, com os atributos da base 01 (temperatura máxima, mínima, umidade...) separadamente, para visualizar o comportamento dos algoritmos caso o número de colunas seja menor. Esses testes foram repetidos com todos os atributos para excluir a possibilidade de uma alteração no comportamento do algoritmo baseado em alguma especificidade da coluna, ou seja, que o resultado de acurácia fosse alterado por alguma característica única apresentada pelo atributo.

Já para testar a característica de robustez dos algoritmos foi gerada uma terceira base de dados. Esta base é composta pelas mesmas informações da base 01, no entanto, possui valores faltantes e alguns outros errôneos conforme exemplificado na figura 11. Essa base de dados é denominada de base 02 quando referenciada adiante neste trabalho.

Figura 11

	year [PK] integer	tmaxjan double precis	tmaxfev double precis	tmaxmar double precis	tmaxabr double precis	tmaxmai double precis
1	1980		28.58682266	30.33087557	27.37964285	24.45391705
2	1981	29.09562211	30.17844387	29.10702764	26.53511904	25.88467741
3	1982	29.23870967	29.15420918		26.45392857	23.59366359
4	1983	29.76440092	29.32359693	27.50230414	25.85392857	23.09539170
5	1984	30.64677419	32.08522167	28.68029953	25.27333333	25.36036866
6	1985	29.73260368	29.89808673	28.55921658	26.55607142	24.23663594
7	1986	30.93801843	28.89005102	28.49896313	27.26	24.01693548
8	1987	30.25126728	27.96071428	29.45806451	27.08083333	20.80218894
9	1988	30.95184331	27.75135467	30.34735023	26.07821428	21.21186635
10	1989	27.43940092	28.25114795	28.55368663	27.12464285	23.22914746
11	1990	28.26866359	29.93966836	29.66474654	27.85869047	22.60552995
12	1991	29.89930875	29.67002551		26.82214285	24.34297235
13	1992	30.57983870	29.71625615	27.47995391	25.49869047	22.87442396
14	1993	29.84550691	27.34464285	29.09642857	27.71035714	23.73041474
15	1994	28.83191244	29.77423469	28.025	26.57821428	24.58917050
16	1995	29.34827188	28.57104591	28.35057603	25.96559523	23.49112903

Figura 11 – Amostra da base de dados 02.

Por fim, após a comparação e escolha dos algoritmos com melhor desempenho, fez-se testes utilizando duas outras bases com características diferentes. Estas bases de dados comportam dados de estações do norte (Ibipora, Joaquim Tavora, Bandeirantes, Londrina,

.

Apucarana, Bela Vista Do Paraiso, Cianorte e Umuarama) e de estações do sul (Guarapuava e Palmas) do estado do Paraná.

3.3.2 Método de classificação

Na meteorologia, área que mantem estreita relação com o uso de bases climatológicas e as aplicações de DCBD sobre estas, a previsão de eventos futuros de forma semi-automatizada e com antecedência apresenta importância notável (LIMA et. al., 2010).

Através de dados de prognósticos disponíveis em grandes bases de dados, é possível conhecer, ou prever, o comportamento do clima, de forma razoavelmente precisa, através dos anos. Estes prognósticos, que contém dados de anos inteiros, são potencialmente mais úteis na previsão de grandes fenômenos climáticos que os dados diários isolados. Então pode-se pensar que ao analisar-se os padrões apresentados por anos passados de El Niño, La Niña, ou anos neutros, teríamos o com classificar os anos futuros nos mesmo moldes.

Mostram-se, assim, mais interessantes as técnicas de mineração de dados que visem à criação de modelos que descrevam os fenômenos e que, posteriormente, façam a predição dos dados sobre estes, já que, como foi explicitado, essa é a principal e mais óbvia aplicação das bases climatológicas. Neste escopo, e considerando as características da base de dados utilizadas nos testes, os algoritmos de classificação apresentam-se como mais adequados à problemática.

De maneira geral a classificação de dados é uma técnica constituída por duas fases distintas. A primeira fase compreende a criação do modelo através da análise do atributo, ou coluna, da tabela que posteriormente será definida como classe e que será relacionada aos dados de treinamento para que seja gerado o modelo que explica o fenômeno (figura 12). Essa variável é chamada de *class label* e caracteriza a técnica como de aprendizado supervisionado (Han e Kamber, 2006). Neste trabalho, por exemplo, a *class label* é a coluna de fenômenos.

.

Figura 12

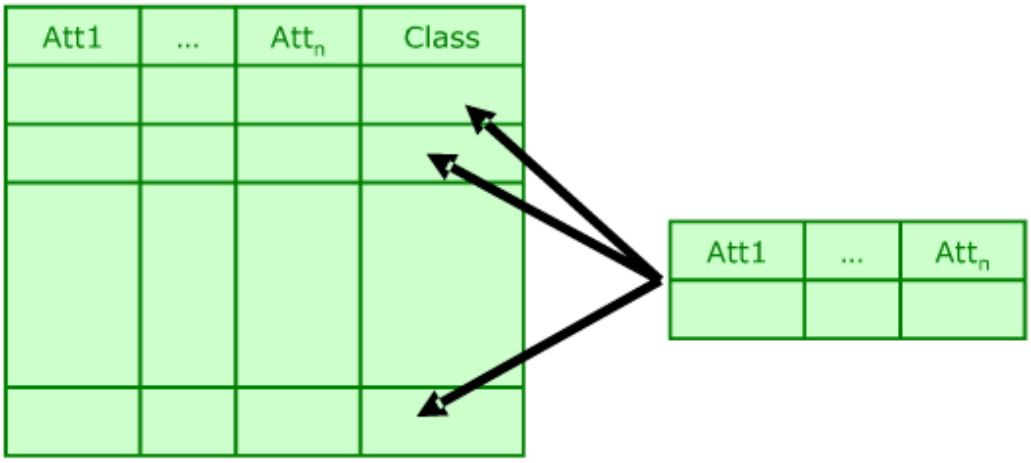

Figura 12 – Processo de classificação.
Fonte: LANZI, 2007.

Antes que se tenha início a segunda fase do processo de classificação é feita a previsão da acurácia do modelo, e só depois, caso a acurácia seja satisfatória é que o modelo pode ser associado a novos dados para que sejam estes classificados concluindo, assim, o processo de mineração de dados (Han e Kamber, 2006).

Neste trabalho a estimativa da acurácia foi feita utilizando um método oferecido pela ferramenta WEKA chamado de *cross-validation*. Este método de teste apresenta um maior índice de confiabilidade já que baseia o seu resultado em um conjunto de dez execuções do algoritmo selecionado, sendo que em cada uma destas execuções 90% dos dados oferecidos são utilizados como grupo de treinamento e os outros 10% restantes são aplicados ao modelo gerado excluindo-se a *class label*, garantindo assim a impossibilidade do algoritmo fazer uso da simples relação trivial já presente na tupla. Lembrando que, esses dados escolhidos para o grupo de treinamento são aleatórios diminuindo assim o risco de exceções. A medida da acurácia do modelo é a porcentagem dos testes que foram corretamente classificados pelo algoritmo.

3.3.3 Algoritmos testados

Foram testados 64 algoritmos de seis famílias, disponibilizadas pela ferramenta WEKA.

.

Nós tópicos a seguir são relatadas as peculiaridades de cada família, bem como os resultados apresentados pelos algoritmos destas.

Bayes

Frequentemente a informação, de um modo geral, é apresentada em termos de probabilidades condicionais, que fornecem a probabilidade de um evento dada uma condição. No entanto existe, naturalmente, o interesse na descoberta da probabilidade de uma condição estar presente dado um resultado, de modo a realizar o processo de modo inverso (Montgomery e Runger, 2006).

Em 1700, Thomas Bayes a fim de tratar essa questão desenvolveu o teorema de Bayes, no qual se baseia essa família de algoritmos, que pode ser descrito partindo-se da definição de probabilidade condicional (Montgomery e Runger, 2006),

$$P(A \cap B) = P(A|B)P(B) = P(B \cap A) = P(B|A)P(A) \quad (1)$$

Ainda, de modo a nos permitir resolver P(A|B) em termos de P(B|A), podemos considerar o segundo e o último termo na definição apresentada anteriormente, então temos (Montgomery e Runger, 2006),

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \text{ para P(B) >0} \quad (2)$$

Desse modo obtemos o seguinte resultado geral, que é conhecido como teorema de Bayes, "Se E_1,E_2, ..., E_k forem eventos mutuamente excludentes e exaustivos e B for qualquer evento, então," (MONTGOMERY E RUNGER, p.34, 2006)

$$P(E_1|B) = \frac{P(B|E_1)P(E_1)}{\sum_{j=l}^{n} P(A_j)P(A|A_j)} \quad (3)$$

Baseada no teorema acima explicitado, também conhecido por classificador de Naïve Bayes, esta família de algoritmos usa-se de probabilidade para classificar os dados em uma classe específica.

Estudos anteriores de comparação entre algoritmos de classificação encontraram um simples algoritmo dessa família que apresenta um desempenho comparável com os algoritmos baseados em árvores de decisão e redes neurais, chamado de *naiveBayes*, que ainda compartilha o bom desempenho e acurácia quando aplicados a grandes bases de dados apresentados pelos demais algoritmos dessa família (Han e Kamber, 2006). Desse modo foi dada uma atenção maior na observação dos resultados exprimidos por esse algoritmo.

.

Foram realizados testes sobre oito algoritmos dessa família, *BayesNet, ComplementNaiveBayes, DMNBtext, NaiveBayes, NaiveBayesMultinominal, NaiveBayesMultinominalUpdateable, NaiveBayesSimple* e *NaiveBayesUpdateable*.

Resultados e conclusões

Na tabela de resultados a seguir estão relacionados os resultados dos testes de acurácia de cada algoritmo com a base de testes sobre qual foi rodado. Observando que também foi testada a acurácia dos algoritmos sobre cada atributo (temperatura máxima, mínima, precipitação...) isoladamente.

Grande parte dos algoritmos dessa família (tabela 02) apresentaram comportamento bastante semelhantes, com resultados de acurácia baixos e pouco variantes quanto a quantidade de atributos testada, deixando as diferenças por conta dos testes rodados sobre a base 02, base esta que tem por objetivo mensurar a robustez dos algoritmos.

Tabela 02 - Resultados de acurácia dos algoritmos da família de Bayes.

	Base 01	Base 02	Precipitação	Umidade	Radiação	T. Máxima	T. Miníma
BayesNet	56,67%	66,67%	46,67%	50%	46,67%	50%	63,33%
ComplementNaiveBayes	40%		40%	50%	16,67%	36,67%	36,67%
DMNBtext	50%		50%	50%	50%	50%	50%
NaiveBayes	46,67%	33,33%	53,33%	33,33%	36,67%	23,33%	40%
NaiveBayesMultinomial	43,33%		43,33%	50%	50%	50%	50%
NaiveBayesMultinomialUpdateable	56,67%		50%	50%	50%	50%	50%
NaiveBayesSimple	43,33%	37,04%	60%	33,33%	40%	20%	40%
NaiveBayesUpdateable	46,67%	33,33%	53,33%	33,33%	36,67%	23,33%	40%

.

Enquanto 50% dos algoritmos (*ComplementNaiveBayes*, *DMNBtext*, *NaiveBayesMultinomial*, *NaiveBayesMultinomialUpdateable*) não puderam rodar quando a base apresenta valores inexistentes, 37,5% deles (*NaiveBayes*, *NaiveBayesSimple*, *NaiveBayesUpdateable*) apresentaram valores de acurácia baixos e menor que os apresentados quanto a base 01 e ao contrário dos demais, o algoritmo BayesNet apresentou uma taxa de acurácia melhorada em 10% quando comparado com a base 01.

Nenhum desses algoritmos apresentou grande potencial quando executados sobre as bases de testes, portanto nenhum mostrou-se interessante para a inferência de modelos em bases de dados climatológicas.

Functions

Essa família é composta, principalmente, por algoritmos que implementam os conceitos de regressão linear simples ou logística.

Segundo Berenson et. al. (2008) a relação mais simples entre duas variáveis consiste em uma relação linear simples e direta, relação essa que é intrinsicamente ligada a regressão linear (gráfico 01).

Gráfico 01

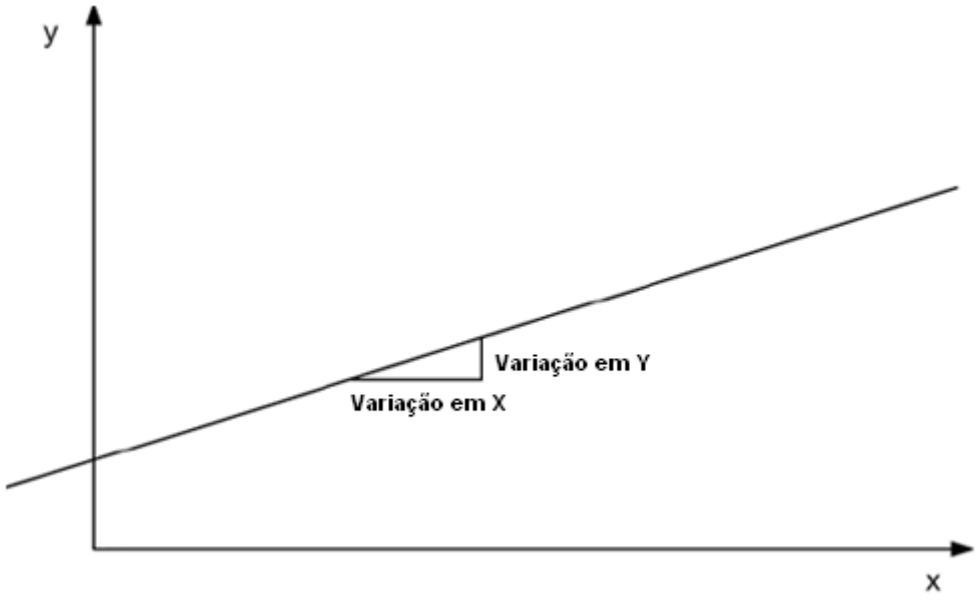

Gráfico 01 – Modelo linear.

Fonte: Adaptado de Berenson et. al; p. 448, 2008.

.

Regressão linear (gráfico 02) é um modelo matemático no qual, basicamente, “uma única variável independente numérica, X, é utilizada para prever a variável dependente numérica, Y” (Berenson et. al., 2008), valendo lembrar que nesse modelo a variável dependente é também contínua.

Gráfico 02

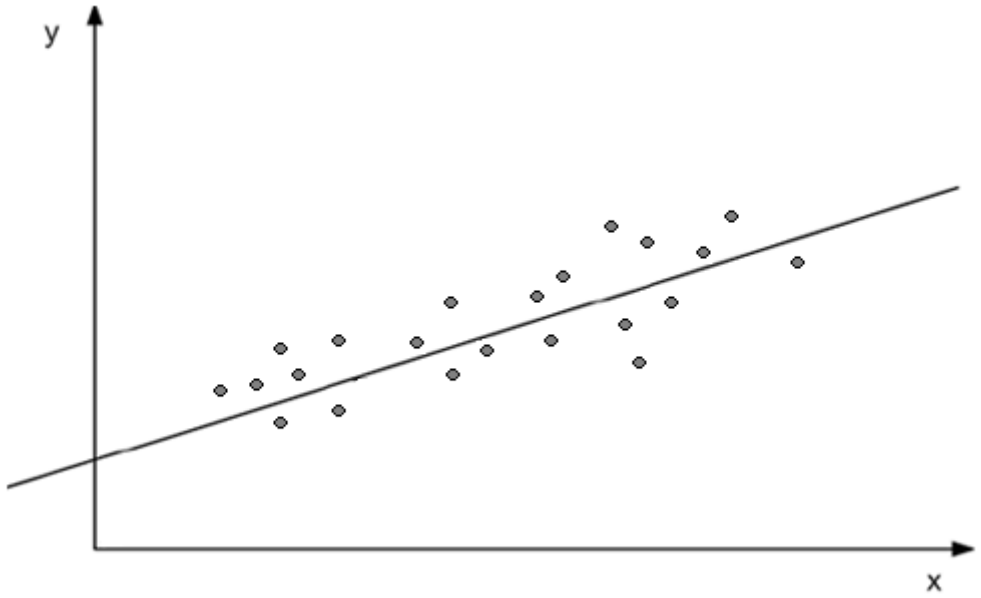

Gráfico 02 – Relação linear simples.
Fonte: Adaptado de Berenson et. al; p. 449, 2008.

Esse modelo de regressão é uma escolha bastante lógica a se considerar quando a *class label* e os atributos da base de dados são todos numéricos e apresentam uma interdependência linear (Witten, Frank e Hall, 2011).

No entanto, em alguns casos a variável dependente Y pode se apresentar como qualitativa e ser expressa por duas ou mais categorias. Nesse caso o uso de modelos de regressão linear podem apresentar resultados imprecisos. Para isso, o modelo de regressão logística (Gráfico 03) pode apresentar resultados mais interessantes (Figueira, 2006).

.

Gráfico 03

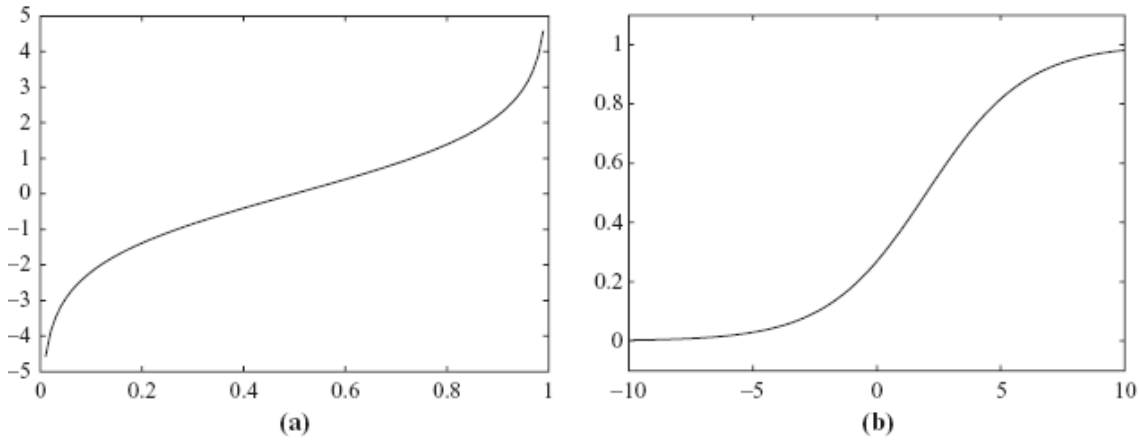

Gráfico 03 – Exemplos de modelos de regressão logística.

Fonte: Witten, Fran e Hall, 2011.

A maior parte dos algoritmos que fazem uso de regressão logística o fazem através de um especificidade desse modelo, a regressão logística binária. Neste modelo a saída do algoritmo possui apenas duas categorias, natureza binária e dicotômica.

Foram realizados testes sobre cinco algoritmos dessa família, *Logistic*, *MultilayerPerceptron*, *RBFNetwork*, *SimpleLogistic* e *SMO*.

Resultados e conclusões

Os algoritmos dessa família apresentaram uma divergência interessante entre si, conforme visto na tabela 03. Apesar de os resultados quanto a base 01 ficarem em torno dos 40% e nenhum dos algoritmos apresentarem um valor de acurácia significativo, cada algoritmo reagiu de forma diferente às diferentes características testadas.

.

Tabela 03 - Resultados de acurácia dos algoritmos da família de Functions.

	Base 01	Base 02	Precipitação	Umidade	Radiação	T. Máxima	T. Miníma
Logistic	40%	40%	26,67%	36,67%	33,33%	26,67%	40%
MultilayerPerceptron	40%	40%	33,33%	30%	37%	33,33%	26,67%
RBFNetwork	36,67%	46,67%	30%	30%	33,33%	23,33%	40%
SimpleLogistic	43,33%	46,67%	50%	40%	43,33%	50%	56,67%
SMO	33,33%	43,33%	43%	50%	50%	46,67%	40%

Enquanto os algoritmos *Logistic* e *MultilayerPerceptron* mantiveram sua acurácia estável quando aplicados a base 02, revelando uma indiferença quanto a possíveis erros ou dados inexistentes, outros tiveram uma aumento de certa de 10% quando aplicados a mesma base de dados.

Além disso, ao analisar os testes feitos sobre os atributos independentemente, percebemos que os algoritmos *SimpleLogistic* e o *SMO* mostraram resultados de acurácia superior quando executados sobre estes, de forma que podemos concluir que seu resultados poderiam obter maior taxa de assertividade caso as bases possuíssem uma quantidade menor de atributos.

Lazy

"Em termos computacionais os sistemas de aprendizado são divididos em duas fases distintas, treinamento e consulta" (Webb e Sammut, p. 572, 2011). A fase de treinamento consiste na inferência de padrões no conjunto de dados de treinamento enquanto a fase de consulta consiste na utilização destes padrões previamente inferidos (Webb e Sammut, 2011).

Os algoritmos dessa família baseiam-se no paradigma computacional de *lazy learning* que se refere a qualquer processo de aprendizado de máquinas que priorize a fase de consulta, a generalização da requisição ocorre quando esta é enviada, em contraste ao *eager learning*, processo que prioriza a fase de treinamento, onde a generalização das requisições é previa (Webb e Sammut, 2011; Brownlee, 2007).

As características de *lazy learning* fazem com que os algoritmos baseados em seus conceitos tenham bons desempenhos com grupos de testes menores ou dados que sofram

.

atualizações constantes, que é o caso dos dados provenientes da web. Outra característica inerte a esses algoritmos é o menor tempo na fase de treinamento compensada pela demora na predição dos dados novos (Webb e Sammut, 2011; Lanzi, 2007).

Foram realizados testes sobre quatro algoritmos dessa família, *IB1*, *IBk*, *KStar* e *LWL*.

Resultados e conclusões

Os algoritmos dessa família apresentaram, de maneira geral, resultados com baixa acurácia, menores que 50% e alguma variação quanto à robustez (tabela 04).

Tabela 04 - Resultados de acurácia dos algoritmos da família de Lazy.

	Base 01	Base 02	Precipitação	Umidade	Radiação	T. Máxima	T. Miníma
IB1	30%	40%	36,67%	36,67%	30%	26,67%	46,67%
IBk	30%	10%	36,67%	36,67%	30%	26,67%	46,67%
KStar	40%	36,67%	50%	36,67%	36,67%	36,67%	33,33%
LWL	63,33%	63,33%	50%	16,67%	40%	63,33%	70%

Ao analisar a tabela acima tem-se dois casos peculiares. O primeiro é apresentado pelo algoritmo *IBk*, que tem sua acurácia reduzida a 10% quando executado sobre dados errôneos ou inexistentes, denotando assim uma baixíssima capacidade deste em obter resultados razoáveis caso a base de dados não esteja devidamente organizada.

O segundo caso é representado pela considerável alta taxa de acurácia apresentada pelo algoritmo *LWL*, que o difere drasticamente dos outros da mesma família. Outro ponto interessante a notar-se sobre esse algoritmo é que o resultado manteve-se igual quando executado sobre a base 02, mostrando uma considerável robustez, especialmente interessante caso o banco de dados alvo apresente dados errôneos e/ou inexistentes.

.

Meta

Essa família de algoritmos baseia-se nos conceitos de *meta learnin*g, que pode ser descrito de maneira básica como técnicas que implementam outras técnicas de mineração de dados. Os algoritmos dessa família se separam em três grupos, o primeiro faz uso de diversos algoritmos de uma só vez de modo que o resultado de um algoritmo represente a entrada de outro, técnica chamada de *Stacking*. O segundo assemelha-se ao primeiro em face de utilizar diversos algoritmos, no entanto esses são executados independentemente, e simultaneamente em diversos casos, sobre diferentes partes do grupo de dados, técnica essa chamada de *Bagging*, e o terceiro grupo de algoritmos são executados de forma recursiva até que se obtenha um modelo consistente, essa técnica é chamada de *Boosting* (Spohn, 2006).

Os algoritmos dessa família apresentam boa acurácia, no entanto, além de consumirem bastantes recursos computacionais, apresentam uma complexidade na sua utilização indesejável.

Foram realizados testes sobre vinte e seis algoritmos dessa família, *AdaBoostM1, AttributeSelectedClassifier, Bagging, ClassificationViaClustering, ClassificationViaRegression, CVParameterSelection, Dagging, Decorate, END, FilteredClassifier, Grading, LogitBoost, MultiBoostAB, MultiClassClassifier, MultiScheme, ClassBalancedND, DataNearBalancedND, ND, OrdinalClassClassifier, RacedIncrementalLogitBoost, RandomCommittee, RandomSubSpace, RotationForest, Stacking, StackingC e Vote*.

Resultados e conclusões

Essa família, além de possuir o maior número de algoritmos, é a que possui as maiores taxas de acurácia e robustez (tabela 05). Cerca de 80% dos algoritmos tiveram assertividade maior ou igual a 50% quando executados sobre a base 01, e cerca de 75% deles tiveram resultados de robustez sobre a base 02 maiores ou iguais do que a taxa de acurácia atingida quando executados sobre a base 01.

.

Tabela 05 - Resultados de acurácia dos algoritmos da família de Meta.

	Base 01	Base 02	Precipitação	Umidade	Radiação	T. Máxima	T. Miníma
AdaBoostM1	76,67%	60%	36,67%	43,33%	43,33%	63,33%	53,33%
AttributeSelectedClassifier	56,67%	66,67%	50%	50%	46,67%	46,67%	60%
Bagging	66,67%	53,33%	53,33%	40%	53,33%	46,67%	56,67%
ClassificationViaClustering	46,67%	46,67%	50%	30%	43,33%	33,33%	36,67%
ClassificationViaRegression	43,33%	43,33%	50%	43,33%	46,67%	36,67%	36,67%
CVParameterSelection	50%	50%	50%	50%	50%	50%	50%
Dagging	50%	50%	53,33%	40%	46,67%	46,67%	46,67%
Decorate	33,33%	53,33%	36,67%	36,67%	36,67%	36,67%	53,33%
END	60%	56,67%	26,67%	40%	63,33%	43,33%	70%
FilteredClassifier	60%	70%	46,67%	50%	46,67%	50%	63,33%
Grading	50%	50%	50%	50%	50%	50%	50%
LogitBoost	66,67%	46,67%	40%	40%	50%	33,33%	60%
MultiBoostAB	56,67%	60%	36,67%	50%	56,67%	56,67%	63,33%
MultiClassClassifier	33,33%	43,33%	26,67%	33,33%	36,67%	20%	43,33%
MultiScheme	50%	50%	50%	50%	50%	50%	50%
ClassBalancedND	56,67%	63,33%	30%	40%	46,67%	40%	60%
DataNearBalancedND	56,67%	56,67%	33,33%	33,33%	56,67%	40%	70%
ND	40%	53,33%	93,33%	43,33%	50%	40%	60%
OrdinalClassClassifier	50%	46,67%	36,67%	33,33%	53,33%	43,33%	60%
RacedIncrementalLogitBoost	50%	50%	50%	50%	50%	50%	50%
RandomCommittee	60%	56,67%	33,33%	40%	46,67%	36,67%	63,33%
RandomSubSpace	66,67%	70%	53,33%	46,67%	46,67%	46,67%	70%
RotationForest	53,33%	70%	46,67%	46,67%	40%	30%	36,67%
Stacking	50%	50%	50%	50%	50%	50%	50%
StackingC	50%	50%	50%	50%	50%	50%	50%
Vote	50%	50%	50%	50%	50%	50%	50%

Pode-se notar, ao analisar a tabela acima, um estranho padrão que alguns algoritmos apresentam, de apresentar resultados de acurácia igual a 50% quando executados sobre qualquer base de dados.

O algoritmo *AdaBoostM1* além de ter apresentado o melhor resultado de acurácia durante este trabalho, aparece como escolha de diversos outras pesquisas, o que sugere um

.

bom desempenho com outras bases de dados, e por isso será discutido e analisado mais a frente.

Rules

Os algoritmos dessa família trabalham com regras de classificação semelhantes as que podem ser extraídas nos processos baseados em indução de árvores de decisão, ou seja, baseiam-se em modelos IF-THEN. Em alguns casos as duas técnicas podem, até mesmo, comportarem-se de maneira semelhante (Han e Kamber, 2006).

A maior diferença entre as técnicas de árvores de decisão e as regras de classificação é a clareza dos modelos apresentados. As árvores de decisão podem apresentar regras muito extensas quando a base sobre qual rodam são muito grandes. Além disso no caso de haver mais de uma classe, as regras de classificação farão o tratamento individual de cada uma sem levar em conta as demais (Han e Kamber, 2006).

Foram realizados testes sobre nove algoritmos dessa família, *ConjunctiveRule, DecisionTable, DTNB, JRip, NNge, OneR, PART, Rido* e *ZeroR*.

Resultados e conclusões

Conforme observado na tabela 06, a seguir, os algoritmos dessa família apresentaram resultados de acurácia mais interessantes quando comparados aos resultados obtidos pela maioria das outras famílias, destacando que nenhum teve um resultado inferior a 50%.

.

Tabela 06 - Resultados de acurácia dos algoritmos da família de Rules.

	Base 01	Base 02	Precipitação	Umidade	Radiação	T. Máxima	T. Miníma
ConjunctiveRule	56,67%	63,33%	46,67%	40%	40%	50%	66,67%
DecisionTable	60%	66,67%	46,67%	50%	46,67%	50%	63,33%
DTNB	60%	66,67%	46,67%	50%	46,67%	50%	63,33%
JRip	63,33%	70%	43,33%	46,67%	53,33%	53,33%	50%
NNge	53,33%	40%	50%	46,67%	46,67%	40%	60%
OneR	70%	70%	50%	40%	70%	30%	46,67%
PART	60%	70%	30%	46,67%	46,67%	40%	66,67%
Ridor	60%	50%	40%	36,67%	50%	33,33%	66,67%
ZeroR	50%	50%	50%	50%	50%	50%	50%

Nota-se a potencialidade do algoritmo *JRip*, que além de obter uma boa média de acurácia quando executado sobre a base 01 e um aumento de aproximadamente 7% nela quando executado sobre a base 02, teve sua taxa de assertividade melhorada quando houve um aumento no número de atributos testados.

Trees

Os algoritmos dessa família baseiam-se nos conceitos de árvores de decisão, que segundo Preiss (2000) é uma das estruturas não-lineares mais importantes. Muito frequentemente uma árvore representa uma hierarquia, devido ao fato que o relacionamento entre os itens lembra os galhos e folhas de uma árvore real (Preiss, 2000).

Uma árvore A pode ser descrita como um conjunto finito, não-vazio de nós, na forma de (Preiss, 2000),

$$A = \{raiz\} \cup A_1 \cup A_2 \cup A_3 \ldots \cup A_n \quad (4)$$

O termo *raiz* indicado na expressão refere-se ao item inicial da árvore, os demais se relacionam a este ou a outra árvore, notando que a definição assim é recursiva já que temos uma árvore definida em termos dela própria.

Os termos finais, ou de grau zero, não possuem subárvores, também chamado de *filhos*, esses termos são chamados de folhas e, nos algoritmos de mineração de dados

.

baseados em árvores de decisão, representam as classes finais, ou resultados (Han e Kamber, 2006; Preiss, 2000).

Tratando-se dos algoritmos dessa família a árvore de decisão é organizada de modo que cada nodo interno represente um teste sobre algum atributo, cada galho represente os resultados deste teste e cada folha represente uma classe ou a distribuição do atributo sobre uma classe (Han e Kamber, 2006).

É criada uma regra para cada caminho, da raiz à folha, onde cada valor e sua relação com o nodo anterior compõem a cláusula IF e cada folha compõem a cláusula THEN. Esta é uma grande vantagem dos algoritmos dessa família, já que modelos do tipo IF-THEN são facilmente interpretáveis por seres humanos, principalmente caso a árvore seja muito grande. Desse modo os modelos gerados por esse processo podem ser facilmente convertidos a regras de classificação em um modelo IF-THEN (Han e Kamber, 2006).

Foram realizados testes sobre doze algoritmos dessa família, *BFTree*, *DecisionStump*, *FT*, *J48*, *J48graft*, *LADTree*, *LMT*, *NBTree*, *RandomForest*, *RandomTree*, *REPTree* e *SimpleCart*.

Resultados e conclusões

Assim como os algoritmos baseados em regras de classificação e *meta learning*, os algoritmos de indução de árvores de decisão (tabela 07) apresentaram boas taxas de acurácia, em geral maiores que 50%.

.

Tabela 07 - Resultados de acurácia dos algoritmos da família de Trees.

	Base 01	Base 02	Precipitação	Umidade	Radiação	T. Máxima	T. Miníma
BFTree	66,6667%	76,6667%	46,6667%	43,3333%	50%	46,6667%	66,6667%
DecisionStump	70%	70%	33,3333%	46,6667%	43,3333%	63,3333%	70%
FT	50%	50%	50%	50%	50%	50%	50%
J48	60%	83,3333%	40%	43,3333%	53,3333%	50%	80%
J48graft	63,3333%	83,3333%	40%	43,3333%	53,3333%	43,3333%	80%
LADTree	63,3333%	70%	26,6667%	36,6667%	53,3333%	40%	66,6667%
LMT	50%	60%	43,3333%	40%	43,3333%	50%	56,6667%
NBTree	56,6667%	66,6667%	46,6667%	50%	46,6667%	50%	63,3333%
RandomForest	53,3333%	36,6667%	46,6667%	43,3333%	33,3333%	36,6667%	60%
RandomForest	43,3333%	46,6667%	36,6667%	50%	40%	33,3333%	56,6667%
REPTree	60%	73,3333%	50%	46,6667%	46,6667%	46,6667%	70%
SimpleCart	70%	60%	46,6667%	50%	46,6667%	40%	73,3333%

Nota-se, no entanto, um fenômeno não observado em tamanha intensidade nos algoritmos anteriores. Alguns algoritmos apresentaram uma taxa de acurácia muito elevada (83,34%), quando executados sobre a base 02, esse fenômeno que já foi notado em outras famílias de algoritmos mas, em menor intensidade revela-se um tanto ilógico, já que é de se pensar que seja mais provável um erro na etapa de classificação de uma tupla sem a *class label* quando algumas das tuplas apresentarem dados errôneos e/ou inexistentes.

Por fim, vale ressaltar, que os algoritmos dessa família possuem a interessante capacidade de mostrar graficamente árvores de decisão. Estas tornam a aplicação desses algoritmos, bem como a observação dos modelos, um processo consideravelmente mais fácil.

3.3.4 MODELOS INFERIDOS

Considerando algoritmos com as maiores taxas de acurácias, igual ou maior que 70%, temos um conjunto de modelos que descrevem com relativo grau de assertividade os fenômenos de El Niño, La Niña e de anos neutros.

.

A análise e comparação desses modelos pode contribuir de forma interessante ao conhecimento que se tem desses eventos climáticos, bem como mostrar a capacidade das técnicas de mineração de dados quando aplicados à fins meteorológicos.

A seguir tem-se a tabela 08 para comparação dos algoritmos que apresentaram os melhores resultados.

Tabela 08 – Comparação dos algoritmos com melhor desempenho.

Algoritmos	Acurácia	Robustez	Atributos Utilizados
AdaBoostM1	76,67%	60%	tminjan
OneR	70%	70%	rnov
DecisionStump	70%	70%	tminjan
SimpleCart	70%	70%	tminjan e rnov

Sendo que tminjan representa o atributo da temperatura mínima do mês de janeiro e rnov a radiação do mês de novembro.

Dessa forma, a seguir relata-se os modelos inferidos sobre os algoritmos *AdaBoostM1*, *OneR*, *DecisionStumb* e *SimpleCart*.

AdaBoostM1

Dos 64 algoritmos testados, o que obteve melhor taxa de acurácia foi o *AdaBoostM1* (76,67%). Esse algoritmo pertence a uma classe que faz uso da técnica de *Boosting* que, como já foi citado anteriormente neste trabalho, procura melhorar a performance de um algoritmo fraco, ou instável através de recursividade, e utiliza um método de pesos que, diferentemente do método de *Bagging*, sofre influência dos modelos gerados anteriormente (Spohn, 2006; Rezende, 2009).

Dessa forma, inicialmente, é feita a atribuição de peso igual a todas as instâncias do algoritmo. De acordo com as saídas de cada instância, ou seja, conforme o algoritmo termina a execução em uma das instâncias, o peso das demais é atualizado (figura 13), de forma que as saída corretas tem seu peso diminuído enquanto as saídas classificadas erroneamente têm seus pesos mantidos iguais, aumentando sua diferença quanto o conjunto de saídas corretas (Rezende, 2009).

.

Figura 13

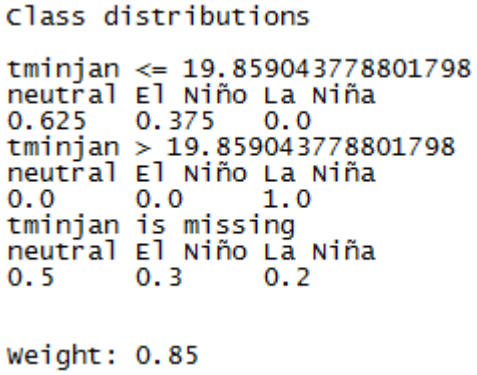

Figura 13 – Exemplo de uma saída de instância do algoritmo *AdaBoost.1M*. Detalhe ao peso do modelo no fim do trecho.

De acordo com Rezende (2009), as atualizações geradas por cada instância respeita uma equação descrita por

$$peso = peso * \frac{e}{(1-e)} \quad (4)$$

Onde *e* é o número de erro do erro, que pode ser calculado quando o algoritmo acaba sua execução sobre a instância que desencadeou o processo de atualização dos pesos.

Observando que, esta fórmula somente é aplicada às instâncias corretas, já que as incorretas mantem seus pesos iguais.

Na figura 14, a seguir, é feita a descrição do algoritmo *AdaBoostM1*.

.

Figura 14

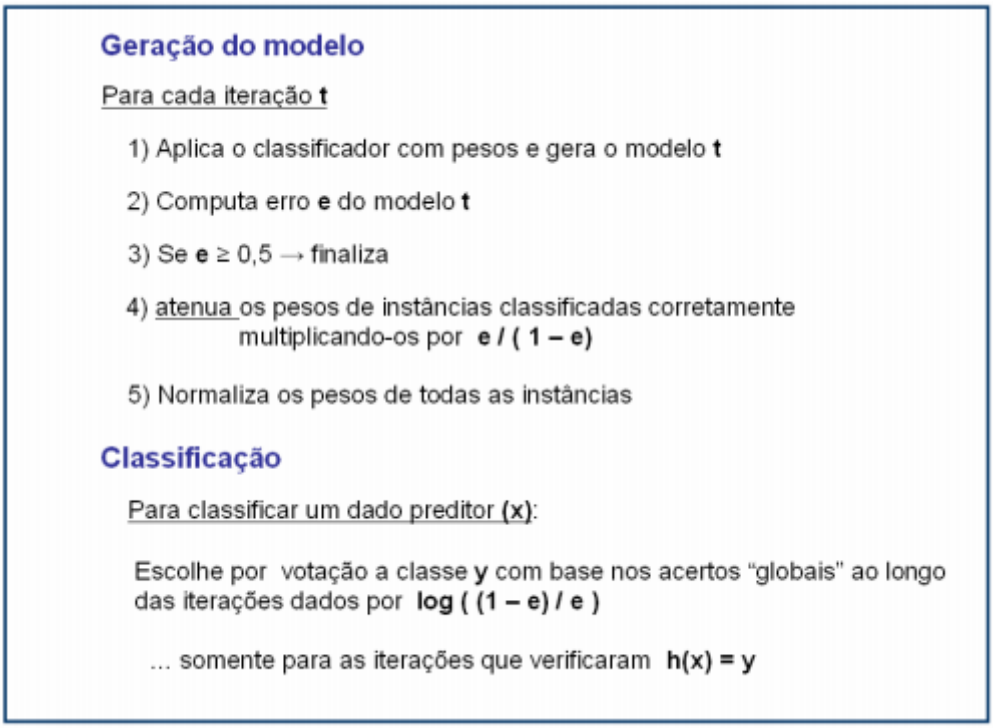

Figura 14 – Descrição do algoritmos *AdaBoost.1M*.

Fonte: Rezende, 2009

Na figura 15 pode-se observar o modelo inferido pelo algoritmo *AdaBoost1M*. Nota-se nele que apenas o atributo temperatura mínima de janeiro é que teve relevância na construção do modelo de classificação.

.

Figura 15

```
=== Classifier model (full training set) ===

AdaBoostM1: Base classifiers and their weights:

Decision Stump

Classifications

tminjan <= 19.859043778801798 : neutral
tminjan > 19.859043778801798 : La Niña
tminjan is missing : neutral

Class distributions

tminjan <= 19.859043778801798
neutral El Niño La Niña
0.625   0.375   0.0
tminjan > 19.859043778801798
neutral El Niño La Niña
0.0     0.0     1.0
tminjan is missing
neutral El Niño La Niña
0.5     0.3     0.2


Weight: 0.85

   -
Decision Stump

Classifications

tminjan <= 19.859043778801798 : El Niño
tminjan > 19.859043778801798 : La Niña
tminjan is missing : El Niño

Class distributions

tminjan <= 19.859043778801798
neutral El Niño La Niña
0.4166666666666667      0.5833333333333334      0.0
tminjan > 19.859043778801798
neutral El Niño La Niña
-3.626728547108846E-16  -1.0362081563168133E-16 1.0000000000000004
tminjan is missing
neutral El Niño La Niña
0.35714285714285715     0.5000000000000001      0.14285714285714288


Weight: 0.59

Decision Stump

Classifications

tminjan <= 19.859043778801798 : neutral
tminjan > 19.859043778801798 : La Niña
tminjan is missing : neutral

Class distributions

tminjan <= 19.859043778801798
neutral El Niño La Niña
0.5625000000000001      0.43749999999999994     0.0
tminjan > 19.859043778801798
neutral El Niño La Niña
0.0     0.0     1.0
tminjan is missing
neutral El Niño La Niña
0.5000000000000001      0.38888888888888884     0.11111111111111112


Weight: 0.45
```

```
Decision Stump

Classifications

tminjan <= 19.859043778801798 : El Niño
tminjan > 19.859043778801798 : La Niña
tminjan is missing : El Niño

Class distributions

tminjan <= 19.859043778801798
neutral El Niño La Niña
0.46176652972637794     0.5382334702736221      0.0
tminjan > 19.859043778801798
neutral El Niño La Niña
1.5724047578394803E-16  -2.358607136759221E-16  1.0
tminjan is missing
neutral El Niño La Niña
0.3748230408657629      0.4368924403057853      0.18828451882845182


Weight: 0.51


Decision Stump

Classifications

tminjan <= 19.859043778801798 : neutral
tminjan > 19.859043778801798 : La Niña
tminjan is missing : neutral

Class distributions

tminjan <= 19.859043778801798
neutral El Niño La Niña
0.588640402843602       0.4113595971563981      0.0
tminjan > 19.859043778801798
neutral El Niño La Niña
-4.9151561251714636E-17 0.0     1.0
tminjan is missing
neutral El Niño La Niña
0.5000000000000001      0.34941502075732794     0.15058497924267195


   Weight: 0.62

Decision Stump

Classifications

tminjan <= 19.859043778801798 : El Niño
tminjan > 19.859043778801798 : La Niña
tminjan is missing : El Niño

Class distributions

tminjan <= 19.859043778801798
neutral El Niño La Niña
0.43456155696479326     0.5654384430352066      0.0
tminjan > 19.859043778801798
neutral El Niño La Niña
-2.5581813965353345E-16 3.837272094803002E-16   0.9999999999999999
tminjan is missing
neutral El Niño La Niña
0.38426955428792425     0.5000000000000001      0.1157304457120757


Weight: 0.47
```

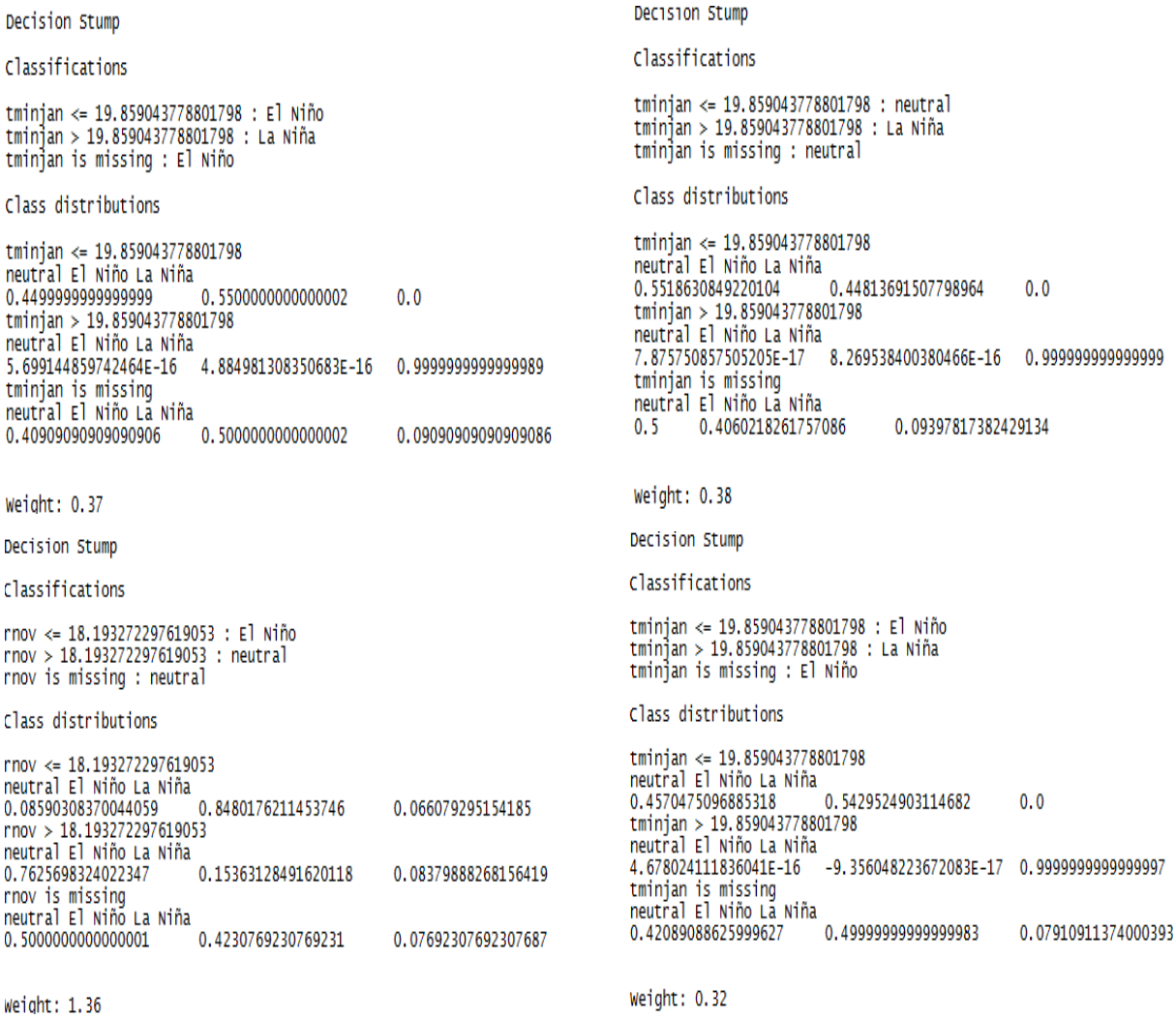

Figura 15 – Modelo gerado pelo *AdaBoost.1M*.

É interessante notar que o único modelo em que o fenômeno La Niña não aparece tem o maior peso, ou seja, foi o que teve menor capacidade de classificação. Com exceção deste, todos os outros modelos apresentam o fenômeno La Niña, enquanto o El Niño e os anos neutros intercalam-se aparecendo em um ou outro fenômeno, mas nunca juntos. Isso reflete-se diretamente na matriz de confusão (figura 16), mostrando claramente uma maior facilidade na classificação do fenômeno La Niña, já que todos foram corretamente classificados, e uma dificuldade na criação de um modelo que classifique El Niño ou anos neutros.

Esse problema provavelmente poderia ter sido evitado caso o algoritmo tivesse incluído outros atributos (temperatura máxima, precipitação, umidade...) em seus modelos.

.

Figura 16

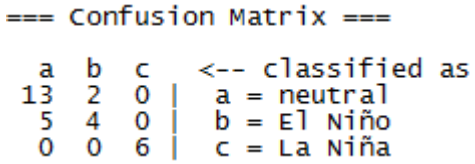

Figura 16 – Matriz de confusão do algoritmo *AdaBoost.M1*.

OneR

Este algoritmo gera regras em um padrão IF-THE, como é caracteristico dos algoritmos de regras de classificação, para cada atributo do banco de dados testado. No entanto é tomado como resultado apenas a regra que obter maior taxa de acurácia. Percebe-se então uma desvantagem na utilização deste algoritmo para classificação de fenômenos climaticos, já que estes, normalmente, estão relacionados a mais de uma variavel (Martins, Marques e Costa, 2009).

A seguir (figura 17) pode-se analisar o modelo inferido pelo algoritmos *OneR* para explicar os fenômenos. Observa-se nesse algoritmo que, diferentemente do *AdaBoost1M*, ele criou seu modelo baseado no atributo radiação do mês de novembro, quando analisado o algoritmo *SimpleCart* percebe-se que ambos (temperatura mínima do mês de janeiro e radiação solar do mês de novembro) os atributos foram utilizados.

Figura 17

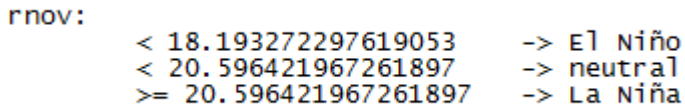

Figura 17 – Modelo gerado pelo *OneR*.

DeciosionStump

Este algoritmo é uma forma de árvore de decisão simplificada, de modo que possui apenas uma raiz. Dessa forma, normalmente é utilizado em conjunto com outros algoritmos ou na forma de *Boosting* (Benjio, 2006).

.

Na figura 18 tem-se o modelo resultado do algoritmo de indução de árvore de decisão *DecisionStump*.

Figura 18

```
=== Classifier model (full training set) ===

Decision Stump

Classifications

tminjan <= 19.859043778801798 : neutral
tminjan > 19.859043778801798 : La Niña
tminjan is missing : neutral

Class distributions

tminjan <= 19.859043778801798
neutral El Niño La Niña
0.625   0.375   0.0
tminjan > 19.859043778801798
neutral El Niño La Niña
0.0     0.0     1.0
tminjan is missing
neutral El Niño La Niña
0.5     0.3     0.2
```

Figura 18 – Modelo gerado pelo *DecisionStump*.

Como pode-se notar, esse algoritmo apresenta uma característica que prejudica a sua escolha para classificação baseada em bases climatológicas. O seu modelo não contemplou todas as possibilidade para o atributo *class label*, ou seja, não fez a classificação de todos os fenômenos, já que não foi criada regra para o fenômeno El Niño. Isso fica mais explicito ao analisar-se a matriz de confusão (figura 19).

Figura 19

```
=== Confusion Matrix ===

  a  b  c   <-- classified as
 15  0  0 |  a = neutral
  9  0  0 |  b = El Niño
  0  0  6 |  c = La Niña
```

Figura 19 – Matriz de confusão do *DecisionStump*.

.

É notável que todas as tuplas correspondentes ao fenômeno El Niño foram classificadas como anos neutros. Além disso, é interessante notar que o conjunto IF-THEN referente ao fenômeno La Niña é semelhante a inferida pelo algoritmo *AdaBoost1M*, e que ambos conseguiram classificar todas as ocorrências desses eventos corretamente.

SimpleCart

Este algoritmo é uma variante do algoritmo de indução de árvores decisão CART (Classification and Regression Trees), que tem como principal característica a "capacidade de gerar árvores de reduzidas dimensões, de elevado desempenho, possuindo grande capacidade de generalização" (GARCIA, p.51, 2003).

O resultado do processo é uma árvore binária, de modo que todos os nodos dividam-se em exatamente dois conjuntos sempre como resposta a questões simples, do tipo sim-não (Garcia, 2003).

A seguir, na figura 20 tem-se o modelo gerado pelo algoritmo *SimpleCart*. Observa-se nesse algoritmo algumas características interessantes, primeiramente ele foi o único dos algoritmos que fez uso de dois atributos na construção do modelo, e também o único registro que foi erroneamente classificado do fenômeno La Niña (figura 21), causado pelo sinal de igualdade na regra gerada para a classificação deste.

Figura 20

```
=== Classifier model (full training set) ===

CART Decision Tree

tminjan < 19.859043778801798
|  rnov < 18.193272297619053: El Niño(7.0/1.0)
|  rnov >= 18.193272297619053: neutral(14.0/2.0)
tminjan >= 19.859043778801798: La Niña(6.0/0.0)
```

Figura 20 - Modelo gerado pelo *SimpleCart*.

.

Figura 21

```
=== Confusion Matrix ===

  a  b  c   <-- classified as
 11  4  0 |  a = neutral
  4  5  0 |  b = El Niño
  1  0  5 |  c = La Niña
```

Figura 21 - Matriz de confusão do algoritmo *SimpleCart*.

3.3.5 IMPLEMENTAÇÃO DOS MODELOS PARA CLASSIFICAÇÃO DE NOVOS DADOS

Com a finalidade de apresentar uma forma de implementação dos modelos para que seja possível a utilização destes na classificação de novos dados climatológicos, foram desenvolvidos três algoritmos em linguagem Java.

Os algoritmos são bastante simples, já que sua funcionalidade limita-se a tarefa de classificação de novos dados a partir de um modelo já criados.

Na construção deles foram aplicados os modelos extraídos do algoritmo *OneR*, *DecisionStump* e *SimpleCart* e podem ser vistos no anexo A.

Não foi implementado um algoritmo para classificação de novos dados baseado no modelo *AdaBoost1M*, já que este iria apresentar uma complexidade maior e a classificação de novos dados não é um objetivo deste trabalho.

Os algoritmos foram construídos utilizando uma conexão ao banco de dados por meio do *driver JDBC* e um conjunto simples de condições, que representam os padrões demonstrados pelos algoritmos de classificação.

3.3.6 RESULTADOS COM BASES REGIONALIZADAS

A fim de obter-se resultados diferenciados, ou que pudessem validar os modelos inferidos, foi escolhido dois algoritmos, *OneR* e *SimpleCart*, para serem executados sobre bases de dados de estações do norte e sul do Paraná.

Os resultados de acurácia (tabela 08) obtiveram taxas inferiores aos apresentados pelos modelos inferidos sobre as outras bases de dados testadas durante este trabalho, bem como fizeram uso de atributos diferentes (média da temperatura máxima de abril e de fevereiro)

.

além da, já vista nos modelos apresentados anteriormente, radiação solar do mês de novembro.

Tabela 09 – Resultados de acurácia sobre as bases regionalizadas.

Algoritmos	Estações do Sul	Estações do Norte
OneR	30%	60%
SimpleCart	50%	63,34%

Estes resultados mostram que mesmo as bases climatológicas apresentam características diferentes entre si, e a escolha do algoritmo deve ser feita cuidadosamente, para que se possa obter melhores resultados.

A seguir tem-se os modelos apresentados (figura 22) para as estações do sul, seguido pelos modelos apresentados (figura 23) para as estações do norte.

Figura 22

OneR

```
=== Classifier model (full training set) ===

tmaxfev:
        < 25.959482758620652    -> neutral
        < 26.472321428571398    -> El NiÃ±o
        < 28.398121921182252    -> neutral
        >= 28.398121921182252   -> La NiÃ±a
```

SimpleCart

```
=== Classifier model (full training set) ===

CART Decision Tree
: neutral(15.0/15.0)
```

Figura 22 – Resultados para as bases das estações do sul.

.

Figura 23

OneR

```
=== Classifier model (full training set) ===

rnov:
        < 20.271510072916698    -> El NiÃ±o
        >= 20.271510072916698   -> neutral
```

SimpleCart

```
=== Classifier model (full training set) ===

CART Decision Tree

rnov < 20.55717044791665: El NiÃ±o(7.0/2.0)
rnov >= 20.55717044791665
|   tmaxabr < 27.680625: La NiÃ±a(5.0/1.0)
|   tmaxabr >= 27.680625: neutral(13.0/2.0)
```

Figura 23 - Resultados para as bases das estações do norte.

Lembrando-se que é necessária a validação destes últimos modelos por técnicas especializados.

.

4 CONSIDERAÇÕES FINAIS

As diversas facetas testadas revelaram, durante este trabalho, que os algoritmos disponíveis para classificação em mineração de dados comportam-se de maneiras peculiares conforme o conjunto de dados sobre os quais se aplicam. E sua escolha deve basear-se no objetivo que se pretende atingir. Por isso, antes de iniciar-se o processo de descoberta de conhecimento deve-se ter em mente o resultado pretendido.

Há alguns fatos a notar-se quanto aos resultados de acurácia dos algoritmos. Primeiramente nota-se que mesmo as melhores taxa de assertividade apresentadas podem ser consideradas insuficientes para uma classificação com baixa margem de erro.

Pode-se notar também que diversos algoritmos apresentaram maiores taxas de acurácia quando executados sobre uma base com dados inconsistentes. Esse é um evento não previsto e que pode dar margem a pesquisas futuras.

Além disso, pode-se notar que alguns algoritmos tem resultados de acurácia iguais a 50% não importando o base sobre qual são executados. A primeira vista estes algoritmos parecem apresentar um resultado incorreto, mas é necessária uma pesquisa mais profunda sobre este evento para que se tenha uma melhor compreensão sobre o mesmo.

É interessante também fazer testes quanto a intensidade dos fenômenos, variável com notável possibilidade de contribuir com o conhecimento acadêmico. Essa intensidade classifica o fenômeno El Niño ou La Nina em três categorias diferentes, *low*, *moderate* e *high*. No entanto devido aos poucos registros, os modelos gerados seriam bastante imprecisos, já que alguns fenômenos com determinada intensidade repetem-se apenas duas vezes. Então, em vista de não fornecer dados incorretos à comunidade científica essa análise não foi realizada.

De forma geral, apesar de não ter sido possível a classificação da intensidade dos fenômenos, pelos motivos apresentados anteriormente, a inferência de modelos para classificação dos fenômenos El Niño, La Niña e de anos neutros foi feita com uma margem de aproximadamente 30% de erro e foi possível fazer a análise de comportamento dos algoritmos. De forma que a escolha destes em futuros trabalhos deixe de ser arbitrária e ganhe um caráter científico.

Quanto aos modelos inferidos pelos algoritmos pode-se observar que eles fizeram uso de apenas dois atributos (média da temperatura mínima do mês de janeiro e a radiação solar do mês de novembro). Estes atributos podem ter sido suficientes para classificar corretamente

.

os anos de La Niña, no entanto sempre apresentaram problemas na classificação dos fenômenos El Niño e de anos neutros. Estes problemas poderiam ser resolvidos caso os demais atributos fossem considerados.

Tendo em vista que as pesquisas meteorológicas fazem uso de, normalmente, espaços de três meses de informações para fazer inferências quanto às características dos fenômenos climatológicos, pode-se pensar, como trabalho futuro, a possibilidade de utilização de técnicas de mineração de dados para inferir modelos nestes espaços intrasazonais. Desse modo, pode-se, por exemplo, encontrar modelos de comportamento destes fenômenos dentro intervalos de dias, ao invés de meses.

.

REFERÊNCIAS

AMO, Sandra de. *Técnicas de Mineração de Dados*. Disponível em: <http://www.deamo.prof.ufu.br/arquivos/JAI-cap5.pdf>. Acesso em 24 Jun 2011.

BABKINA, A. M. *El Niño:* Overview and Bibliography. Nova York, 2003.

BENJIO, Samy. *Statical machine learning from data*: Decision Trees. Martigny, 2006.

BERENSON, Mark L; et. al. *Estatística:* Teoria e Aplicação. Rio de Janeiro, 2008.

BERRY, Michael; LINOFF, Gordon. *Mastering Data Mining: The Art and Science of Customer Relationship Management.* Wiley, 2000.

BROWNLEE, Jason. *Lazy and Competitive Learning*. Melbourne, 2007.

CHAPMAN, Pete; CLINTON, Julian; KERBER, Randy; KHABAZA, Thomas; REINARTZ, Thomas; SHEARER, Colin; WIRTH, Rüdiger. *CRISP-DM 1.0:* step-by-step data mining guide. Illinois: SPSS, 2000.

DIAS, Cristiano Araujo. *Descoberta de Conhecimento em Banco de Dados para Apoio á Tomada de Decisão.* Guaratinguetá, 2002.

ELMASRI, Ramez; NAVATHE, Shamkant B. *Sistemas de Banco de Dados*. 4ª ed. São Paulo: Pearson Addison Wesley, 2005.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. *From data mining to knowledge Discovery in database. Al Magazine*, v. 17, n. 3, 1996a.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. *Knowledge Discovery and data mining:* towards a unifying framework. Em: INTERNATIONAL CONFERENCE ON

.

KNOWLEDGE DISCOVERY AND DATA MINING, 2., 1996. *Proceedings...* Menlo Park: AAAI Press, 1996b.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. Data Mining to *Knowledge Discovery: an overview*. Em: FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P.; UTHURUSAMY, R. (Ed.). *Advances in knowledge Discovery and data mining*. Menlo Park: AAAI Press, 1996c.

FÉLIX, L. C. M. *Data Miningno Processo de Extração de Conhecimento em Bases de Dados.* São Carlos, 1998.

FIGUEIRA, Cleonis Viater. *Modelos de Regressão Logísitca*. Porto Alegre, 2006.

GARCIA, Simone Carboni. *O uso de árvores de decisão na descoberta de conhecimento na área da saúde*. Porto Alegre, 2003.

GONÇALVEZ, Lóren Pinto Ferreira. *Avaliação de ferramentas de mineração de dados como fonte de dados relevantes para a tomada de decisão*: aplicação na rede Unidão de supermercados. São Leopoldo-RS [Dissertação]. Porto Alegre, 2011.

HAN, Jiawei; KAMBER, Micheline. *Data Mining*: Concepts and Techniques. 2ª ed. San Francisco: Morgan Kaufmann, 2006.

HEBARMANN, Humberto. *A importância do Business Intelligence no Planejamento Estratégico de pequenas e médias empresas*. Disponível em: <http://www.datawarehouse.inf.br/Artigos/BI%20Humberto.pdf>. Acesso em 15 Jun 2011.

LANZI, Pier Luca. *Machine Learning and Data Mining:* 13 Nearest Neighbor and Bayesian Classifiers. Milão, 2007. 52 slides.

LIMA, Glauston Roberto Teixeira de; et. al. *Mineração de Dados Meteorológicos para Previsão de Eventos Severos pela Abordagem de Similaridade de Vetores.* São José dos Campos, 2010.

.

MARTINS, António Cardoso; MARQUES, João Miguel, COSTA, Paulo Dias; *Estudo comparativo de três algoritmos de machine learning na classificação de dados eletrocardiográficos*. Porto, 2009

MEIRA, Carlos Alberto Alves. *Processo de Descoberta de Conhecimento em Bases de Dados para Análise e Aletra de Doenças de Culturas Agrícolas e sua Aplicação na Ferrugem do Cafeeiro.* Campinas, 2008.

MONTGOMERY, Douglas C; RUNGER, George C. *Estatística aplicada e probabilidade para engenheiros.* Rio de Janeiro, 2009.

NAVEGA, Sérgio. *Princípios essenciais do Data Mining*. São Paulo, 2002.

NEVES, Rita de Cássia David das. *Pré-Processamento no Processo de Descoberta de Conhecimento em Banco de Dados.* Porto Alegre, 2003.

OLIVEIRA, Aristeu Giovani de. *A importância dos dados das variáveis climáticas na pesquisas em Geografia: um estudo de caso empregando a precipitação pluviométrica*. Disponível em: <http://www.seer.ufu.br/index.php/caminhosdegeografia/article/view/10763>. Acesso em 21 Jun 2011.

PETTA, Nicolina Luiza de; OJEDA, Eduardo Aparicio Baez. *História*: uma abordagem integrada. 1ª ed. São Paulo: Moderna, 1999.

PREISS, Bruno R. *Estruturas de dados e algoritmos:* Padrões de projetos orientados a objeto com Java. Rio de Janeiro, 2000.

WAIKATO. Attribute-Relation File Format. Disponível em: <http://www.cs.waikato.ac.nz/ml/weka/arff.html>. Waikato, 2008. Acesso em 01 Set 2011.

WEBB, Geoffrey I; SAMMUT, Claude. *Encyclopedia of Machine Learning.* Nova York, 2011.

.

WITTEN, Ian H; FRANK, Eibe; HALL, Mark A. *Data Mining:* Practical Machine Learning Tools and Techniques. Burlington, 2011.

SILBERSCHATZ, Abraham; KORTH, Henry F.; SUDARSHAN, S. *Sistema de Banco de Dados*. 5ª ed. Rio de Janeiro: Elsevier, 2006.

SILVA, Carolina Martins Soares. *Utilizando o Processo de Descoberta de Conhecimento em Banco de Dados para Identificar Candidatos a Padrões de Análise para Bancos de Dados Geográficos.* Porto Alegre, 2003.

SILVA, Glauco Carlos. *Mineração de Regras de Associação Aplicada a Dados da Secretaria Municipal de Saúde de Londrina – PR*. Porto Alegre, 2004.

SPOHN, Daniel. *Meta Learning:* For Classification. Youngstown, 2006. 14 slides.

REZENDE, Luiz Felipe Campos de. *Mineração de dados aplicada à análise e predição de cintilação ionosférica.* São José dos Campos, 2009.

VIANNA, Glaciene Lago. *Aplicação da Técnica de Árvore de Decisão Utilizando Algoritmo J48 para Analisar Ocorrência de Sinistralidade em uma O peradora de Seguro Saúde.* Salvador, 2006.

WIVES, Leandro Krug. *Tecnologia de Descoberta de Conhecimento em Textos Aplicada à Inteligencia Competitiva*. Porto Alegre, 2002.

.

APÊNDICES

APENDICE A – Implementação dos modelos para classificação de novos dados.

Classe de Conexão ao banco de dados

```
package util;

import java.sql.Connection;
import java.sql.DatabaseMetaData;
import java.sql.DriverManager;
import java.sql.SQLException;

/**
 *
 * @author vinícius
 */
public class AcessoBD {
    private Connection conn;
    private String stringDeConexao;
    private String enderecoServidor;
    private String nomeBancoDeDados;
    private String nomeUsuario;
    private String senhaUsuario;

    // Construtor
    public AcessoBD(){
        // Carrega o driver de conexÃ£o com o banco de dados
        // se nÃ£o achar o driver lanÃ§a uma exceÃ§Ã£o.
        try{
            Class.forName("org.postgresql.Driver");
        }
        catch (ClassNotFoundException cnfe){
            System.out.println(cnfe.getMessage());
```

.

```
        }
    }
    // Construtor com parÃ¢metros
    public AcessoBD(String servidorBD, String nomeBD, String usuarioBD, String
senhaBD){
        this();
        this.setStringConexao(servidorBD, nomeBD, usuarioBD, senhaBD);
    }

    // Estabelece conexÃ£o com banco de dados.
    public void conectar(){
        // Tenta conectar na base de dados.
        try{
            conn = DriverManager.getConnection(getStringDeConexao());
            System.out.println("ConexÃ£o com BD estabelecida com sucesso!");
        }
        catch (SQLException sqlex){
            System.out.println(sqlex.getMessage());
        }
    }
    // Desconecta do banco de dados.
    public void desconectar(){
        if (conn!=null){
            try{
                conn.close();
                System.out.println("ConexÃ£o encerrada com sucesso!");
            }
            catch (Exception e){
                System.out.println("desconectar: erro ao fechar a conexÃ£o.");
            }
        }
    }
    // Retorna um objeto Connection
```

.

```
    public Connection getConexao(){
        return conn;
    }
    // Retorna string de conexÃ£o com o banco
    public String getStringDeConexao(){
        return stringDeConexao;
    }
    // Configura a string de conexÃ£o
    public void setStringConexao(String servidorBD, String nomeBD, String usuarioBD, String senhaBD){
        this.setEnderecoServidor(servidorBD);
        this.setNomeBancoDeDados(nomeBD);
        this.setNomeUsuario(usuarioBD);
        this.setSenhaUsuario(senhaBD);
        // Monta a string de conexÃ£o com banco de dados.
        // Essa string serve apenas para banco Postgres
        // Exemplo de string de conexÃ£o:
        // "jdbc:postgresql://localhost/teste?user=postgres&password=ifsul"
        stringDeConexao = "jdbc:postgresql://";
        stringDeConexao += enderecoServidor+"/";
        stringDeConexao += nomeBancoDeDados+"?";
        stringDeConexao += "user="+nomeUsuario+"&";
        stringDeConexao += "password="+senhaUsuario;
    }

    // Metado que retorna os metadados do banco de dados
    public DatabaseMetaData getMetaData() throws SQLException{
        DatabaseMetaData metadado = conn.getMetaData();
        return metadado;
    }

    public String getEnderecoServidor() {
        return enderecoServidor;
```

.

```
    }
    public void setEnderecoServidor(String enderecoServidor) {
        this.enderecoServidor = enderecoServidor;
    }
    public String getNomeBancoDeDados() {
        return nomeBancoDeDados;
    }
    public void setNomeBancoDeDados(String nomeBancoDeDados) {
        this.nomeBancoDeDados = nomeBancoDeDados;
    }
    public String getNomeUsuario() {
        return nomeUsuario;
    }
    public void setNomeUsuario(String nomeUsuario) {
        this.nomeUsuario = nomeUsuario;
    }
    public String getSenhaUsuario() {
        return senhaUsuario;
    }
    public void setSenhaUsuario(String senhaUsuario) {
        this.senhaUsuario = senhaUsuario;
    }
}
```

Implementação do modelo apresentado por OneR

```
package viniciuspierdona;

import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import util.AcessoBD;
```

.

```
/**
 *
 * @author Pierdona
 */
public class OneR {

   public static void main(String[] args) {

       AcessoBD bd;
       PreparedStatement pstmt;
       ResultSet rs;

      bd = new AcessoBD("localhost", "mining", "postgres", "postgres");

       String sql = "SELECT * FROM mining.data1 ORDER BY year;";
       bd.conectar();
       try {
          Float rnov;
          Float year;
          pstmt = bd.getConexao().prepareStatement(sql);
          rs = pstmt.executeQuery();
          while (rs.next()) {
             rnov = rs.getFloat("rnov");
             year = rs.getFloat("year");
             if(rnov < 18.193272297619053){
                System.out.println("Ano "+year+" - El Niño");
             }else if (rnov < 20.596421967261897){
                System.out.println("Ano "+year+" - Ano neutro");
             }else if(rnov >= 20.596421967261897){
                System.out.println("Ano "+year+" - La Niña");
             }
          }
          rs.close();
```

.

```
            pstmt.close();
        } catch (SQLException ex) {
            System.out.println("Erro ao Buscar: " + ex);
        }



    }
}
```

Implementação do modelo apresentado por DecisionStump

```
package viniciuspierdona;

import util.AcessoBD;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

/**
 *
 * @author Vinícius Pierdona
 */
public class DecisionStump {

    public static void main(String[] args) {

        AcessoBD bd;
        PreparedStatement pstmt;
        ResultSet rs;

        bd = new AcessoBD("localhost", "mining", "postgres", "postgres");

        String sql = "SELECT * FROM mining.data1 ORDER BY year;";
```

.

```
        bd.conectar();
        try {
            Float tmin;
            Float year;
            pstmt = bd.getConexao().prepareStatement(sql);
            rs = pstmt.executeQuery();
            while (rs.next()) {
                tmin = rs.getFloat("tminjan");
                year = rs.getFloat("year");
                if(tmin<= 19.859043778801798){
                    System.out.println("Ano "+year+" - Ano neutro");
                }else if (tmin > 19.859043778801798){
                    System.out.println("Ano "+year+" - La Niña");
                }else{
                    System.out.println("Ano "+year+" - Ano neutro");
                }
            }
            rs.close();
            pstmt.close();
        } catch (SQLException ex) {
            System.out.println("Erro ao Buscar: " + ex);
        }


    }
}
```

Implementação do modelo apresentados por SimpleCart

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package viniciuspierdona;
```

.

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import util.AcessoBD;

/**
 *
 * @author Pierdona
 */
public class SimpleCart {

    public static void main(String[] args) {

        AcessoBD bd;
        PreparedStatement pstmt;
        ResultSet rs;

        bd = new AcessoBD("localhost", "mining", "postgres", "postgres");

        String sql = "SELECT * FROM mining.data1 ORDER BY year;";
        bd.conectar();
        try {
            Float tmin;
            Float rnov;
            Float year;
            pstmt = bd.getConexao().prepareStatement(sql);
            rs = pstmt.executeQuery();
            while (rs.next()) {
                tmin = rs.getFloat("tminjan");
                rnov = rs.getFloat("rnov");
                year = rs.getFloat("year");
                if (tmin < 19.859043778801798) {
```

.

```
                if (rnov < 18.193272297619053) {
                    System.out.println("Ano " + year + " - El Niño");
                } else if (rnov >= 18.193272297619053) {
                    System.out.println("Ano " + year + " - Ano neutro");
                }
            } else {
                System.out.println("Ano " + year + " - La Niña");
            }
        }
        rs.close();
        pstmt.close();
    } catch (SQLException ex) {
        System.out.println("Erro ao Buscar: " + ex);
    }


  }
}
```

.