UNIVERSIDADE DO VALE DO TAQUARI – UNIVATES
CURSO DE ENGENHARIA DA COMPUTAÇÃO

APLICAÇÃO DE TÉCNICAS DE MINERAÇÃO DE DADOS PARA DETECÇÃO DE PERDAS COMERCIAIS NA DISTRIBUIÇÃO DE ENERGIA ELÉTRICA

Tiago Jasper Zuege

Lajeado, julho de 2018

Tiago Jasper Zuege

APLICAÇÃO DE TÉCNICAS DE MINERAÇÃO DE DADOS PARA DETECÇÃO DE PERDAS COMERCIAIS NA DISTRIBUIÇÃO DE ENERGIA ELÉTRICA

Monografia apresentada ao Centro de Ciências Exatas e Tecnológicas da Universidade do Vale do Taquari – UNIVATES, como parte da exigência para a obtenção do título de bacharel em Engenharia da Computação.

Orientador: Prof. Ms. Alexandre Stürmer Wolf

Lajeado, julho de 2018

Tiago Jasper Zuege

APLICAÇÃO DE TÉCNICAS DE MINERAÇÃO DE DADOS PARA DETECÇÃO DE PERDAS COMERCIAS NA DISTRIBUIÇÃO DE ENERGIA ELÉTRICA

A Banca Examinadora abaixo aprova a Monografia apresentada ao Centro de Ciências Exatas e Tecnológicas, da Universidade do Vale do Taquari UNIVATES, como parte da exigência para obtenção do grau de Bacharel em Engenharia da Computação:

Prof. Me. Alexandre Stürmer Wolf
Universidade do Vale do Taquari UNIVATES

Prof. Me. Evandro Franzen
Universidade do Vale do Taquari UNIVATES

Prof. Dr. Marcelo de Gomensoro Malheiros
Universidade do Vale do Taquari UNIVATES

Lajeado, julho de 2018

AGRADECIMENTOS

Aos meus pais, Vandir Zuege (*In memoriam*) e Janice Jasper, pelo amor e incentivo, os quais me tornaram na pessoa que sou hoje.

A minha querida esposa, Danieli Zuege, e minha filha, Ágatha Zuege, pela paciência, pelo carinho e principalmente pelo amor, fazendo com que este sonho fosse realizado.

Ao professor Alexandre Stürmer Wolf, pela orientação e dedicação, os quais tornaram este trabalho possível.

RESUMO

As perdas comerciais na distribuição de energia elétrica causam prejuízos no faturamento das concessionárias e inclusive aos próprios consumidores, sendo um desafio detectar tais eventos de maneira eficiente. Para identificar possíveis perdas o mais rápido possível, facilitando o processo de inspeção das unidades consumidoras, este trabalho propõe como solução, o processo de mineração de dados sobre informações de uma empresa de distribuição de energia do Vale do Taquari. Para alcançar este objetivo, o presente trabalho traz um embasamento teórico onde são tratados os principais algoritmos e técnicas utilizadas no contexto de mineração de dados. Para a execução deste processo, foram utilizadas ferramentas de mineração de dados como o WEKA e o módulo scikit-learn. Os algoritmos executados foram avaliados através de métricas de desempenho, sendo que o método *random forests* atingiu uma acuracidade de 82%, conseguindo classificar corretamente 97% dos exemplos negativos de perdas comerciais, enquanto que a técnica de árvore de decisão obteve uma taxa de 76% de acuracidade, com aproximadamente 35% dos casos positivos de perdas comerciais classificados corretamente.

Palavras-chave: Mineração de dados. Perdas comerciais. Aprendizado de máquina.

ABSTRACT

Commercial losses in the distribution of electric power cause losses in the revenues of the concessionaires and even the consumers themselves, being a challenge detecting such events in an efficient way. In order to identify this type of loss as fast as possible, simplifying the inspection process of the consumer units, this paper proposes as a solution, the data mining process about the information of a energy distribution company at Vale do Taquari. To reach this objective, the present work presents a theoretical basis where the main algorithms and techniques used in the data mining context are treated. For the execution of this process, data mining tools like WEKA and the module scikit-learn where used. The algorithms performed were evaluated through evaluation metrics, where the random forests method reached an accuracy of 82% and was able to correctly classify 97% of negative examples of commercial losses, while the decision tree technique obtained a rate of 76% of accuracy, with approximately 35% of positive cases of correctly classified commercial losses.

Keywords: D**ata mining. Comercial loss. Machine learning.**

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

1R	Uma Regra
ANEEL	Agência Nacional de Energia Elétrica
API	Application Programming Interface – Interface de Programação de Aplicações
ARFF	Attribute Relation File Format – Formato de Arquivo de Relação de Atributos
CART	Classification and Regression Trees – Árvores de Classificação e Regressão
CLI	Command Line Interface – Interface de Linha de Comando
CSV	Comma Separated Values – Valores Separados por Vírgula
GUI	Graphic User Interface – Interface Gráfica do Usuário
JSON	JavaScript Object Notation – Notação de Objeto JavaScript
KWh	Kilowatt Hora
KDD	Knowledge Discovery in Databases – Descoberta de Conhecimento em Bancos de Dados
kNN	k Nearest Neighbors – k Vizinhos Próximos
PC	Perda Comercial
PRODIST	Procedimentos de Distribuição de Energia Elétrica no Sistema Elétrico Nacional
RNA	Redes Neurais Artificiais
SQL	Structured Query Language – Linguagem de Consulta Estruturada
SSE	Square Sum Error – Soma dos Erros Quadrados
SGD	Stochastic Gradient Descent – Gradiente Descendente Estocástico
SVM	Support Vector Machines – Máquinas de Vetores de Suporte
TFP	Taxa de Falsos Positivos
TVP	Taxa de Verdadeiros Positivos
UC	Unidade Consumidora
WEKA	Waikato Environment for Knowledge Analysis – Ambiente para Análise de Conhecimento Waikato

SUMÁRIO

1 INTRODUÇÃO

A evolução de tecnologias em persistência e recuperação de dados, bem como a diminuição nos custos de armazenamento e a comodidade de simplesmente manter os dados ao invés de descartá-los, proporcionam um aumento crescente de informações geradas e mantidas no mundo todo. Estima-se que a quantidade de informações presentes em bancos de dados dobram a cada 20 meses. Neste contexto, surgem novos desafios, como a capacidade de analisar e obter conhecimento a partir destes recursos, utilizando técnicas como a mineração de dados como método para resolver estes problemas, trazendo vantagens competitivas e boas oportunidades de negócios para as empresas que utilizam este processo (WITTEN; FRANK, 2005).

Devido a questões legais, algumas instituições governamentais exigem que empresas de determinado segmento armazenem informações por longos períodos para fins de fiscalização. Um exemplo disto, são as empresas de distribuição de energia elétrica que necessitam obrigatoriamente manter determinados registros de seus clientes em um banco de dados. O módulo cinco do PRODIST[1] da Agência Nacional de Energia Elétrica (ANEEL), entidade vinculada ao Ministério de Minas e Energia que regula e fiscaliza os serviços de energia elétrica no Brasil, determina que os dados relacionados as medições e faturamento dos clientes referentes aos últimos 5 anos devem estar armazenados de maneira que possam ser disponibilizados facilmente tanto para os próprios consumidores, quanto para a reguladora. Portanto, além das demandas legais, estes dados proporcionam à distribuidora oportunidades como a disponibilidade de tais informações para o apoio à tomada de decisão. Neste sentido, com a quantidade de informações armazenadas, torna-se necessária a utilização de uma técnica mais avançada como a mineração de dados para transformar estas informações em conhecimento útil, no intuito de auxiliar a organização no ponto de vista estratégico e operacional.

1 Procedimentos de Distribuição de Energia Elétrica no Sistema Elétrico Nacional (PRODIST), são documentos produzidos pela ANEEL que contém normas e padrões das atividades técnicas relativas ao funcionamento e desempenho dos sistemas de distribuição de energia elétrica.

Um exemplo prático da aplicação destes referidos recursos é a descoberta de perdas comerciais de energia, assunto que será discutido no Seção 2.1, que neste conceito é o conhecimento útil que deseja-se obter com base nas informações extraídas de um banco de dados mantido pela organização. Este conhecimento é relevante, pois as perdas comerciais representam um grande problema para as distribuidoras de energia. Segundo o Instituto Acende Brasil, estima-se que em 2015 as perdas comerciais das 59 principais distribuidoras do país foram na ordem de 5%, o que corresponde a mais de 15 milhões de megawatts-hora não tarifados, o equivalente ao consumo do estado de Santa Catarina. Estes números representam, ainda segundo a instituição, a perda de receita de mais de 8 bilhões de reais. Neste contexto, as empresas distribuidoras propõem soluções para minimizar o impacto destas perdas, como o uso de campanhas educativas, inspeções nos medidores de consumidores suspeitos, utilização de medidores eletrônicos, entre outras. Porém, não há como determinar de uma maneira totalmente confiável, quais unidades consumidoras são suspeitas de estarem gerando perdas comerciais sem empregar uma inspeção *in loco*, o que gera altos custos. Por este motivo, inspecionar todos os consumidores é praticamente inviável, todavia, uma das soluções para este problema é analisar os registros de consumo e medição destes consumidores a fim de encontrar automaticamente o conhecimento necessário para executar uma ação corretiva ou preventiva (CASTRO; FERRARI, 2016).

Entretanto, muitas vezes não é possível obter resultados interessantes utilizando métodos tradicionais de análise de dados, mas sim, combinando a estes métodos algoritmos mais sofisticados, chegando assim no processo de mineração de dados (TAN; STEINBACH; KUMAR, 2009).

1.1 Objetivos

Nesta seção serão apresentados o objetivo geral e os objetivos específicos do trabalho, além da metodologia utilizada para o desenvolvimento do presente estudo.

1.1.1 Objetivo Geral

O presente trabalho tem como principal objetivo aplicar técnicas de mineração de dados sobre os registros extraídos de uma base de dados de uma empresa de distribuição de energia elétrica, no sentido de detectar possíveis perdas comerciais de energia de maneira mais eficiente e de uma forma automatizada, de modo que este processo possa ser utilizado como ferramenta de apoio a decisão, tornando assim mais eficiente as tarefas de inspeção das unidades consumidoras.

1.1.2 Objetivos específicos

- Pesquisar trabalhos relacionados ao tema e compará-los de forma a obter informações que possam contribuir para este projeto;

- Selecionar atributos e registros da base de dados que podem potencialmente sugerir ou não uma perda comercial, com ajuda de um especialista na área de inspeção de medições;

- Empregar diferentes técnicas de mineração de dados sobre os dados com os atributos selecionados, utilizando e comparando as ferramentas pesquisadas neste trabalho;

- Verificar e validar a técnica que consegue determinar com maior acuracidade um caso positivo ou negativo de perda comercial através de métricas de desempenho dos algoritmos, como a precisão, a revocação e a medida F, utilizadas para medir a precisão um modelo preditivo, comparando os métodos utilizados e avaliando os seus resultados.

1.2 Metodologia

Inicialmente foi realizado um estudo sobre as áreas de conhecimento que compõem a Mineração de Dados, que fomentaram o embasamento teórico a este trabalho. Também foi executada uma pesquisa de ferramentas de software que foram utilizadas para a aplicação da mineração de dados e suas tarefas associadas, além de um estudo comparativo entre trabalhos selecionados pelo autor que são relacionados ao tema. Posteriormente, com a ajuda de um especialista da área de inspeção de medições, foram definidos alguns atributos do conjunto de dados que podem estar associados as perdas comerciais. Estes atributos associados a outras características definidas pelo autor, foram utilizados como dados de entrada dos algoritmos de mineração de dados. Estes dados de entrada foram extraídos da base e então pré-processados para prepará-los para uma determinada técnica. O conjunto de dados continha informações sobre perdas comerciais e também objetos em que este não era o caso. Este conhecimento prévio, com os objetos já classificados em ambos os casos, foi utilizado como conjunto de treinamento do algoritmo de mineração de dados. Os resultados foram avaliados sobre um conjunto de teste, que continha registros diferentes do conjunto de treino. Com base nos resultados obtidos e nas medidas de desempenho dos algoritmos executados, foi feita uma avaliação para verificar qual a melhor técnica a ser utilizada, apresentando os resultados a um especialista, onde foi verificado se o processo pode ser útil na detecção de perdas comerciais.

1.3 Estrutura do Trabalho

Este trabalho está organizado em seis capítulos. Após o presente capítulo de introdução, encontra-se o capítulo dois, que traz o embasamento teórico utilizado para solucionar o problema proposto neste trabalho, relacionando aspectos sobre as perdas comerciais e definindo os principais conceitos ligados ao assunto de mineração de dados. No terceiro capítulo, foram analisados alguns dos trabalhos relacionados a área de pesquisa, comparando as suas metodologias e resultados, como forma de apoiar a resolução do problema proposto.

Já no quarto capítulo, estão relacionadas as tecnologias que foram utilizadas no desenvolvimento deste trabalho, como bibliotecas e ferramentas de *software* para análise de dados e aplicação de algoritmos de classificação. No quinto capítulo, o desenvolvimento do trabalho foi descrito com base na metodologia proposta, demonstrando como os objetivos foram alcançados, discutindo e apresentando os resultados obtidos. Por fim, no sexto capítulo, foram descritas as considerações finais.

2 REFERENCIAL TEÓRICO

Este capítulo tem como objetivo definir caracterização da perda comercial e como funciona o processo de detecção de perdas comerciais na empresa objeto do presente estudo, além de descrever as principais técnicas e metodologias que compõem o campo de Mineração de Dados, definindo todas as tarefas que fazem parte deste processo e o atual estado da arte. Nas seções que descrevem as técnicas de mineração de dados, será abordado um ou mais algoritmos onde estas são aplicadas, além da descrição do funcionamento dos mesmos.

2.1 Perdas comerciais

O conceito de perda, tratando-se de distribuição de energia elétrica pode ser definido conforme Gedra, Borelli e Barros (2014):

> As perdas em uma rede de distribuição podem ser divididas entre perdas técnicas e comerciais. As perdas técnicas são aquelas associadas ao aquecimento dos condutores e equipamentos por onde circula a energia elétrica. As perdas comerciais estão relacionadas às ligações irregulares e às fraudes nos centros de medição dos consumidores (GEDRA; BORELLI; BARROS, 2014, p. 36).

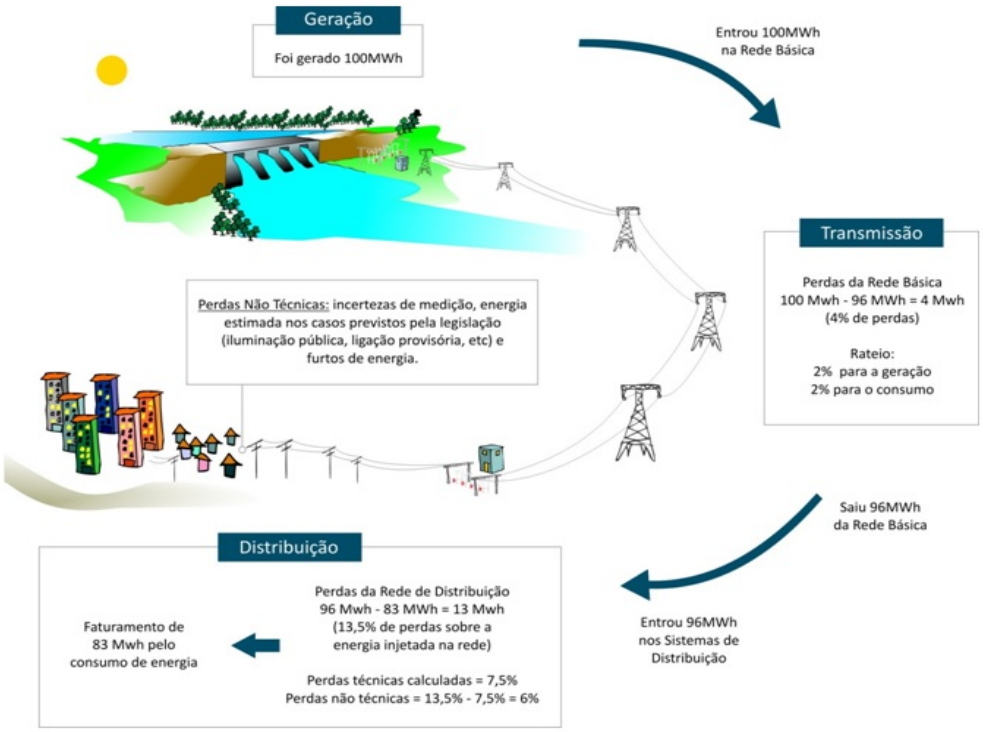
Segundo a Associação Brasileira de Distribuidores de Energia Elétrica (ABRADEE) (2017), a energia decorrente da perda comercial não é faturada pela distribuidora, apesar de ser injetada no sistema de distribuição. A perda comercial pode ser dividida em duas categorias principais (ABRADEE, 2017):

- Furto: pode ser definido como o desvio de energia elétrica do sistema de distribuição da concessionária de energia, realizado de forma ilegal para atender um consumidor ilícito.

- Fraude: caracteriza-se como uma alteração dos condutores de energia elétrica a fim de beneficiar o consumidor, legalmente registrado na distribuidora, para que este pague um valor menor do que o seu consumo.

Cabe ressaltar que as perdas comerciais podem ser representadas não somente por furto ou fraude, mas também por erros na medição do consumo de energia e por falhas no faturamento. Outra questão a ser observada é que os valores destas perdas são regulados pela ANEEL, portanto as perdas comerciais interferem no valor regulatório que é repassado para a conta de energia. Deste modo, um consumidor, mesmo que não se utilize de uma prática ilegal, acaba arcando com as consequências destas perdas. A figura 1 demonstra como as perdas ocorrem no sistema elétrico (ANEEL, 2015).

Figura 1 – Perdas no sistema elétrico



Fonte: ANEEL (2017).

Para combater as perdas comerciais, as empresas distribuidoras de energia elétrica tem realizado algumas ações que vão desde a divulgação e propaganda de materiais educativos ao

uso de medidores eletrônicos, além das inspeções periódicas nas unidades consumidoras. Porém este processo tem um custo elevado, já que requer o deslocamento de uma equipe especializada até o local (CASTRO; FERRARI, 2016).

Na empresa em que este projeto foi aplicado, existe um setor específico que visa detectar as perdas comerciais, no sentido de recuperar a receita perdida, seja por questões de fraude, problemas no faturamento, erros de leitura ou falhas no medidor. Uma equipe especializada analisa os dados de consumo do usuário de energia para verificar possíveis irregularidades, além do próprio profissional que está a campo realizando a leitura estar atento a anormalidades. Por vezes, a detecção de uma perda se dá através de denúncias anônimas, no caso das fraudes ou desvios, ou por meio de vistorias rotineiras da unidade consumidora.

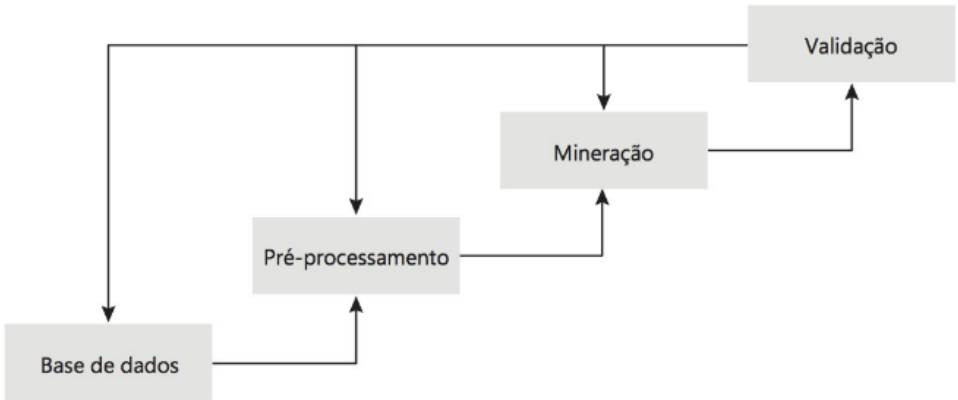
As análises dos dados são realizadas por meio de um sistema gerencial, que aplica uma regra de verificação sobre os dados de consumo, onde são filtrados os consumidores que se aplicam a esta condição. A condição em questão é a variação de aproximadamente 30% do consumo de energia. Consumidores que se aplicam a esta regra, são analisados pela equipe técnica e podem ser posteriormente candidatos a uma vistoria de inspeção. No caso de irregularidade, são aplicadas algumas regras e cálculos para reaver o consumo não faturado causado pela perda comercial. Em alguns casos, é necessário aguardar mais de um ciclo de leitura do consumo para confirmar a anormalidade, descartando assim períodos de férias, por exemplo, onde o consumidor encontra-se ausente. Em casos de fraude, a mesma só será confirmada por meio de inspeção, ficando a empresa impossibilitada de realizar a cobrança da perda sem a confirmação *in loco* da irregularidade. Uma maneira de auxiliar o especialista na detecção das perdas comerciais é através técnicas de mineração de dados, oferecendo uma maior confiabilidade na análise dos dados.

2.2 Definição de mineração de dados

O volume crescente de dados disponível se deve principalmente a informatização da sociedade e da evolução de tecnologias de armazenamento e coleta de dados. Redes de telecomunicações, buscas na web, redes de lojas mundialmente distribuídas, experimentos científicos, redes sociais, entre outros, geram uma enorme quantidade de dados. A necessidade de se obter conhecimento sobre estes dados, originou a área de mineração de dados (HAN, KAMBER e PEI, 2011).

Análogo ao processo de mineração, que consiste em extrair minerais valiosos a partir de uma mina, o termo mineração de dados equivale ao processo de explorar uma base de dados por meio de ferramentas que auxiliam a obter um conhecimento. Conforme proposto na primeira conferência sobre *Knowledge Discovery in Databases* (KDD) realizada em Montreal no Canadá, no ano de 1995, esta é a etapa de descoberta de um macroprocesso conhecido como Descoberta de Conhecimento em Bancos de Dados. Este processo está divido em etapas que, além da própria mineração de dados, compõem a base de dados, o pré-processamento e a validação ou avaliação do conhecimento, conforme a figura 2 (CASTRO; FERRARI, 2016).

Figura 2 – Processo de descoberta de conhecimento



Fonte: Castro e Ferrari (2016, pág. 6).

Tan, Steinbach e Kumar (2009) definem a mineração de dados como um processo automático para descoberta de informações úteis, sendo aplicadas sobre grandes massas de

dados no sentido de identificar padrões que poderiam passar desapercebidos. Os autores também salientam a multidisciplinaridade da área, que se utiliza de ideias como inteligência artificial, reconhecimento de padrões, aprendizado de máquina, estatística e banco de dados.

2.3 Dados de entrada

Castro e Ferrari (2016), e Tan; Steinbach e Kumar (2009), afirmam que é importante conhecer os dados e prepará-los de forma adequada para o pré-processamento. Neste sentido, existe um nível de qualidade requerida para os dados de entrada, pois é de se esperar que os mesmos apresentem alguns problemas quanto a sua coleta.

Han, Kamber e Pei (2011), ressaltam que conhecimentos de informações estatísticas acerca dos dados, como as medidas de tendência central por exemplo, podem auxiliar a preencher dados ausentes, ou detectar anomalias nos objetos. Ferramentas de visualização de dados também são úteis para conhecer o conjunto de dados e prepará-lo para o pré-processamento.

O conjunto de dados pode ser representado por uma coleção de objetos que possuem determinados atributos. Estes objetos muitas vezes são conhecidos como registros, entidades, vetores, padrões, entre outros. Já os atributos dos dados são propriedades que variam no tempo ou entre diferentes objetos e também possuem diferentes tipos e características, o que permite utilizar uma determinada operação estatística que esteja mais apta para cada tipo (TAN; STEINBACH; KUMAR, 2009). Um atributo pode ter valores do tipo numérico, também chamado de atributo contínuo, podendo este ser composto por números inteiros ou reais, ou nominal, muitas vezes chamado de atributo categórico, pois este assume valores de um conjunto predefinido de possibilidades (WITTEN; FRANK, 2005).

Os dados brutos extraídos de uma base podem apresentar problemas que necessitam ser tratados antes da execução dos processos de mineração de dados. Os principais problemas que

podem ser encontrados em um conjunto de dados são os relacionados a seguir (CASTRO; FERRARI, 2016):

- Dados incompletos: podem ser a falta de um determinado atributo ou a falta de um objeto inteiro em um determinado conjunto. Porém, às vezes a falta pode não ser detectada, a não ser por um especialista no domínio dos dados. Este problema não é incomum, visto que alguns objetos não possuem atributos obrigatórios, e assim, podem vir a estarem faltando. Algumas estratégias para lidar com dados incompletos consistem em eliminar objetos inteiros que contenham informações faltantes, estimar o valor do atributo ausente por meio de técnicas estatísticas ou simplesmente ignorá-los;

- Dados Inconsistentes: um determinado dado pode vir a ser inconsistente se este conter um atributo incompatível ou tiver um desvio de sua característica em relação aos outros dados do conjunto. Casos comuns de inconsistência podem ocorrer em atributos que utilizam unidades de medida. Para corrigir um problema de inconsistência, muitas vezes são necessários dados redundantes ou alguma informação auxiliar;

- Ruídos: geralmente estão associados a dados que variam no tempo. Um ruído nos dados pode compreender um valor distorcido ou objetos adulterados adicionados ao conjunto, sendo que este problema leva a inconsistência dos dados. Mesmo assim, alguns algoritmos de mineração aceitam certos níveis de ruídos nos objetos.

2.4 Pré-processamento dos dados

A etapa de pré-processamento dos dados, conforme Castro e Ferrari (2016), também chamada de preparação da base de dados, é crucial para atingir bons resultados na mineração. Segundo os autores, este processo consiste em detectar problemas nos dados, conhecê-los e também prepará-los para que a aplicação das técnicas de mineração de dados obtenha resultados válidos. Uma falha no pré-processamento pode fazer com que este processo seja ineficiente ou até mesmo inutilizá-lo.

Tan, Steinbach e Kumar (2009), ressaltam que o pré-processamento dos dados é fundamental para ajustá-los a um tipo de técnica específica de mineração. Neste sentido, pode ser necessário a execução de algumas tarefas, como discretizar dados contínuos ou reduzir a quantidade de atributos.

Como as tarefas de pré-processamento são tão importantes quanto a aplicação das técnicas de mineração de dados, é importante entender com clareza o funcionamento de cada método que compõe esta etapa. Algumas tarefas para a preparação da base de dados são relacionadas nas próximas seções.

2.4.1.1 Limpeza

Este método trata de uma série de etapas responsáveis por resolver os problemas relacionados a qualidade dos dados, conforme citado na Seção 2.3 deste capítulo. Conforme Castro e Ferrari (2016), o processo de limpeza dos dados atua no sentido de resolver os seguintes problemas:

- Valores ausentes: são considerados um problema para certo tipo de algoritmos de mineração, podendo ser necessário substituí-los manualmente no conjunto por um valor estimado, porém, o valor incluso não deve distorcer a base. Este processo é chamado de imputação. O dado a ser imputado pode ser definido empiricamente, desde que obedeça o domínio do atributo. É possível também inserir um valor constante para todos os atributos ausentes do conjunto, ou um valor aleatório que seja similar aos demais. Ainda pode-se simplesmente ignorar objetos com atributos faltantes, ou utilizar ferramentas estatísticas como a média para valores numéricos e a moda para valores nominais, onde os resultados podem ser imputados em atributos incompletos;

- Dados ruidosos: um problema de qualidade dos dados que também devem ser tratados no processo de limpeza da base. Ruídos não detém um padrão para facilitar a sua identificação e podem resultar de erros acumulados ou erros de entrada dos dados na base. Neste sentido, algumas técnicas podem ser utilizadas no intuito de suavizar os dados e reduzir o impacto dos ruídos no processo de mineração, como o encaixotamento[2]. Ainda é possível utilizar algoritmos de agrupamento, onde os objetos são agrupados automaticamente baseados em valores médios daquele objeto, sendo que estes valores centrais substituirão valores com ruído. Outra maneira de suavizar dados ruidosos é por meio de aproximações, onde os atributos resultantes de funções de aproximação são substituídos pelos atributos verdadeiros;

- Dados inconsistentes: valores com inconsistência devem ser tratados muitas vezes através da ajuda de um especialista. Existem técnicas para determinar se um atributo corresponde a determinado domínio ou não, sendo uma

2 Método que consiste em definir valores que representam os dados ruidosos, que podem ser computados através da média ou moda.

delas, o algoritmo *Apriori*[3]. Por outro lado, a utilização de gráficos para cada atributo também facilitam a identificação de inconsistências na base.

2.4.1.2 Integração dos dados

O passo de integração consiste em agregar conjuntos de dados em uma única base. Porém, este processo pode ocasionar alguns problemas, segundo os autores. O primeiro deles é a redundância de dados, que na área de mineração, possui um conceito diferente da duplicidade, ou seja, dados redundantes são, neste contexto, atributos ou características que podem ser obtidos de outros atributos do conjunto. Uma das maneiras de detectar-se a redundância é por meio de uma análise de correlação.

Outro problema que pode vir a surgir durante a integração dos dados é a duplicidade de atributos. Estes podem aparecer de forma repetitiva no conjunto, e pode ser resolvida por meio da normalização dos dados, por exemplo.

Já os conflitos de dados que, segundo os autores, também podem ocorrer nesta fase do pré-processamento, ocorrem quando um mesmo objeto que, por exemplo, esteja em duas bases diferentes, possui escala ou unidade de medida diferentes em cada conjunto em que este objeto pertence.

3 O algoritmo *Apriori* é um método utilizado para encontrar relação entre itens através de regras em um conjunto de dados.

2.4.1.3 Redução dos dados

A tarefa de redução consiste em reduzir o número de objetos ou atributos do conjunto. Alguns algoritmos de mineração podem exigir um esforço computacional muito grande, ainda mais sobre uma grande quantidade de registros. Neste sentido, esta tarefa busca selecionar os atributos desejáveis e descartar os irrelevantes, bem como comprimi-los, reduzir o próprio número de objetos ou discretizar atributos, para tornar o processo de mineração mais eficiente.

A compressão, que é uma maneira de reduzir os dados, equivale a obter uma representação para os atributos, diminuindo a dimensionalidade dos dados. Através desta representação é possível chegar aos dados originais sem perda, ou com perda de informações. Um procedimento estatístico que visa chegar neste objetivo é chamado de análise de componentes principais, que segundo Tan, Steinbach e Kumar (2009) é uma estratégia da álgebra linear utilizada em atributos contínuos para obter novos atributos, que são chamados de componentes principais. Estas novas características são combinações dos atributos originais e possuem o máximo de variações dos dados.

Para reduzir o número dos dados no conjunto, Castro e Ferrari (2016) propõe alguns métodos para selecionar os objetos que serão mantidos e outros descartados do processo de mineração. Um dos métodos é a amostragem, que resume-se a selecionar um subconjunto de objetos que simboliza toda a coleção. Pode-se selecionar uma amostra selecionando registros distintos de forma aleatória, de forma sistemática, por grupos ou estratificada. Outro método proposto é o de utilizar modelos de aproximação para seleção dos dados, que podem ser divididos em paramétricos e não-paramétricos. Em uma aproximação paramétrica, é possível obter um modelo que represente os dados através de uma função de aproximação, como uma função polinomial por exemplo. Já em uma aproximação não-paramétrica, pode ser utilizada uma técnica de agrupamento, onde os dados podem ser representados pelos elementos centrais dos grupos encontrados.

2.4.1.4 Transformação dos dados

A tarefa de transformação baseia-se em padronizar, normalizar e discretizar o conjunto de dados, ou seja, preparar a coleção de objetos em formatos propícios para o processo de mineração. Neste sentido, a padronização resolve questões como capitalização dos dados nominais, tratamento de caracteres especiais, ajustes de formatos de dados, como datas e a padronização de unidades de medida (CASTRO; FERRARI, 2016).

Já Han, Kamber e Pei (2011), definem que a normalização dos dados é um processo análogo a padronização, que basicamente busca definir um padrão para os dados, colocando-os dentro de um intervalo em comum. Deste modo, o objetivo da normalização é dar um peso igual a todos os atributos. Uma forma de normalizar os dados é através da normalização *Max-Min* que executa uma transformação linear nos dados originais, conforme equação (1), onde *A* é um atributo do conjunto, v_i é um valor de *A*, e v'_i é o mapeamento de um valor v_i no intervalo $[novomin_A, novomax_A]$.

$$v'_i = \frac{v_i - min_A}{max_A - min_A}\left(novomax_A - novomin_A\right) + novomin_A \quad (1)$$

2.4.1.5 Discretização

A tarefa de discretização é basicamente a conversão de atributos contínuos em atributos categóricos. Este procedimento é importante para certo tipo de técnica de mineração, como a classificação, por exemplo. Para discretizar atributos contínuos, é necessário definir o número de categorias a serem utilizadas e de que forma eles devem ser associados a estas classes (TAN; STEINBACH; KUMAR, 2009).

Para Castro e Ferrari (2016), a discretização pode ser realizada através de métodos de agrupamento, por meio de utilização de faixas de um histograma ou ainda através do método

de encaixotamento. Segundo os autores, também existe uma forma mais trivial, que é dividir os atributos em intervalos iguais.

2.5 Aprendizagem de máquina em mineração de dados

A mineração de dados possui um vasto campo para suas aplicações, incorporando diversas técnicas de outras áreas de estudo, como por exemplo o aprendizado de máquina. O processo de mineração de dados tem tido êxito devido a esta qualidade (HAN; KAMBER; PEI, 2011).

O aprendizado de máquina pode ser definido como um processo de descoberta de uma hipótese ou uma aproximação de função através de experiências passadas, dando assim a capacidade de computadores aprenderem e tirarem conclusões com base em exemplos (CARVALHO et. al, 2011).

Conforme Luger (2013) o desempenho do aprendizado deve melhorar não só na execução repetitiva da mesma tarefa, mas sim generalizar a solução de novos problemas a partir da visualização de parte de todos os exemplos possíveis.

Russel e Norvig (2013) determinam que uma das justificativas de fazer um agente aprender está na questão de que não é possível prever todas as situações em que ele possa se envolver. Há também a questão de que não se pode prever todas as ocorrências de um determinado problema, já que estas podem sofrer mudanças com o passar do tempo. Já em relação a aprendizagem, os autores identificam três principais formas de um agente aprender:

- Aprendizagem supervisionada: nesta forma de aprendizagem, o agente aprende uma função que relaciona as entradas com as saídas, com base no estudo de exemplos de outros pares de entradas e saídas;

- Aprendizagem não supervisionada: nesta abordagem, o agente identifica padrões na entrada que foi fornecida, porém não existem exemplos de relação entre entradas e saídas;

- Aprendizagem por reforço: o agente aprende através de um reforço, ou seja, através de recompensas ou punições, conforme avança para a direção correta, ficando responsável por decidir qual ação posterior ao reforço proporcionou um melhor resultado.

2.6 Técnicas de mineração de dados

As técnicas de mineração de dados estão associadas a um conjunto de tarefas que possuem objetivos específicos, dependendo do tipo de informação que se deseja obter a partir dos dados. Estas tarefas estão agrupadas em dois tipos: descritivas e preditivas. Tarefas descritivas são utilizadas para encontrar características nas propriedades dos dados, enquanto que as tarefas preditivas têm como objetivo obter conclusões a partir dos dados, realizando predições (CASTRO; FERRARI, 2016).

Nas seções a seguir, são descritas técnicas que compõem as tarefas da mineração de dados, relacionando alguns dos principais algoritmos para cada técnica.

2.6.1 Técnicas de classificação

Em alguns casos, para a solução de determinados problemas em que há um histórico de dados, pode ser necessário relacionar um valor de saída para cada objeto, com base nesses valores passados. Esta abordagem implica em construir um modelo que possa, com base em

registros históricos, predizer um valor de saída para cada novo registro no conjunto de dados. Valores de saída também são chamados de rótulos de classe (CASTRO; FERRARI, 2016).

Deste modo, técnicas de classificação, propostas principalmente por pesquisadores na área de aprendizagem de máquina, estatística e reconhecimento de padrões, são métodos utilizados para analisar dados através de modelos, chamados de classificadores, que realizam a tarefa de predizer o rótulo de classe para cada objeto. Este rótulo de saída é um valor discreto, não-ordenado. Os algoritmos de classificação têm um vasto número de aplicações, sendo algumas na área de detecção de fraudes, marketing, diagnósticos médicos, entre outros (HAN; KAMBER e PEI, 2011).

Para a construção deste modelo preditivo, existem duas etapas principais, segundo Castro e Ferrari (2016), sendo elas:

- Etapa de Treinamento: nesta etapa, um conjunto de dados com os valores de saída já conhecidos para cada objeto é utilizado para construir o modelo preditivo;

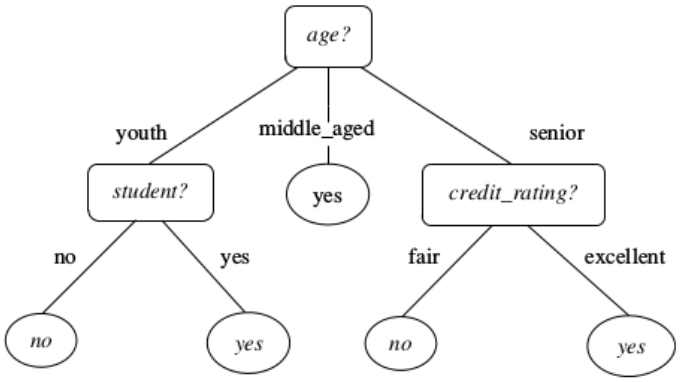
- Etapa de Teste: consistem em avaliar o desempenho do modelo gerado na etapa anterior, com dados que não foram utilizados no treinamento, exibindo uma estimativa da sua capacidade de classificação.

2.6.1.1 Árvores de decisão

Árvores de decisão são uma espécie de fluxograma, sendo que os nós internos são os testes feitos sobre cada atributo, as arestas são a saída destes testes e os nós folhas são o rótulo da classe, conforme ilustrado na Figura 3. Quando uma classe desconhecida precisa ser classificada, cada objeto do conjunto de dados é testado na árvore. Então, um caminho é percorrido na estrutura, desde a raiz até o nó folha, que possui o rótulo da classe para o objeto.

Deste modo, árvores de decisão são estruturas simples, que não dependem de parâmetros adicionais. São utilizadas para uma análise exploratória dos dados, sendo que é um método bastante popular, devido a sua rapidez e simplicidade, porém, o seu êxito depende muito dos dados utilizados (HAN; KAMBER e PEI, 2011).

Figura 3 – Árvore de decisão



Fonte: Han, Kamber e Pei (2011, p. 331).

A estratégia utilizada para construir uma árvore decisão, conforme Castro e Ferrari (2016), é dada de maneira recursiva, através de um algoritmo guloso[4], que de uma maneira geral, segue os seguintes passos:

- O primeiro nó é a representação do conjunto de dados;
- Se os objetos pertencem a uma mesma classe, o nó é rotulado com esta classe, tornando-se um nó folha. Caso contrário, é utilizado um atributo de teste para dividir a base;
- Este processo é executado recursivamente, de forma que ele pode ser interrompido em algumas condições: a não existência de objetos para o atributo de teste, se todos os registros de determinado nó pertencerem a

4 Algoritmos gulosos tentam encontrar a melhor solução das escolhas disponíveis no momento, ou seja, uma solução ótima local, buscando encontrar uma solução ótima global.

mesma classe ou se não há mais atributos para particionar os objetos. Neste último caso é utilizado um método de votação pela maioria para definir a classe.

Após a construção de uma árvore de decisão, muitos ramos da árvore podem refletir irregularidades dos dados de treinamento, devido a ruídos ou anomalias dos dados. Por este motivo, existem métodos de *prunning* ou poda da árvore, que consistem em remover ramos não confiáveis. Neste sentido, existem duas abordagens: a pré-poda, em que a construção da árvore é parada mais cedo, e a pós-poda, quando ramos da árvore são removidos após ela ter sido construída por inteiro (HAN; KAMBER e PEI, 2011).

2.6.1.2 *Random forests*

Random forests ou florestas aleatórias é uma técnica que pertence aos chamados métodos de grupo. Um método de grupo é um conjunto de classificadores onde cada um realiza um voto e uma determinada classe predita é retornada pelo método por meio da coleção de votos praticados por classificador. Métodos de conjunto tendem a ter uma acuracidade maior em relação aos classificadores que o compõem (HAN; KAMBER e PEI, 2011).

Segundo Han, Kamber e Pei (2011), *random forests* é um conjunto de árvores de decisão, formando assim uma floresta. Cada árvore de decisão é gerada baseada em um atributo de escolha aleatória que determina a divisão em cada nodo.

Random forests suportam tanto variáveis numéricas quanto atributos categóricos para dados de entrada. Em problemas em que há muitas variáveis de entrada, como no caso de diagnósticos médicos, a técnica de árvores aleatórias pode aumentar a acuracidade, porém ela depende da força de cada árvore que compõe a floresta, bem como da dependência de cada

uma. *random forests* também são mais robustas no que diz respeito a ruídos (BREIMAN, 2001).

2.6.1.3 Classificadores bayesianos

Conforme Tan, Steinbach e Kumar (2009), os classificadores bayesianos, tem a capacidade de, através da probabilidade, prever se um registro pertence a uma determinada classe. Eles possuem uma taxa de acurácia e velocidade de execução altos quando aplicados a grandes bases de dados. Estes tipos de classificadores são baseados no teorema de Bayes, que é dado pela equação 2:

$$P(X,Y)=P(Y|X).P(X)=P(X|Y).P(Y) \quad (2)$$

Sendo que P(X,Y) é a função de probabilidade do par de variáveis aleatórias X e Y, onde P(X|Y) e P(Y|X) são as probabilidades condicionais em que as variáveis são dependentes. Portanto, após a reorganização da relação, é possível determinar o teorema de Bayes (conforme equação (3)).

$$P(Y/X)=\frac{P(X/Y)}{P(X)} \quad (3)$$

Para exemplificar, a classificação determina a probabilidade de um objeto X pertencer a uma determinada classe, quando uma hipótese H for satisfeita, com base no valor X. Portanto, a finalidade do teorema de Bayes é calcular a probabilidade *a posteriori*, definida como *P(H|x)*, a partir de *P(H)*, *P(X)* e *P(x|H)* (CASTRO; FERRARI, 2016).

Naive[5] Bayes é um exemplo de um algoritmo de classificação bayesiano, que parte do pressuposto que o efeito de um valor de determinado atributo em uma determinada classe é independente dos valores de outros atributos (HAN; KAMBER e PEI, 2011).

5 A palavra naive significa ingênuo.

Segundo Castro e Ferrari (2016), este princípio, também chamado de independência condicional de classe, é utilizado para tornar os cálculos mais simples.

2.6.1.4 Classificadores de k vizinhos mais próximos

A técnica de classificação de *k* vizinhos mais próximos, amplamente utilizada na área de reconhecimento de padrões, é baseada na comparação de um objeto do conjunto de teste com os demais objetos similares a ele. Os objetos são representados por *n* atributos, onde cada objeto é um ponto em um espaço de padrões de *n* dimensões, no qual todos os exemplos do conjunto são inseridos. Neste sentido, dado um objeto desconhecido, o classificador de *k* vizinhos mais próximos procura no espaço de padrões pelos *k* objetos de treinamento que são mais próximos do objeto desconhecido através de medidas de proximidade (HAN; KAMBER e PEI, 2011).

Conforme Tan, Steinbach e Kumar (2009), pelo fato de classificadores de *k* vizinhos mais próximos utilizarem medidas de distância para as predições, erros podem ocorrer devido a variação na escala de atributos, como no caso de atributos que representam o peso e altura de uma pessoa. Neste caso a medida de proximidade do algoritmo pode ser dominada por atributos de escala maior, como é o caso do atributo de peso. Outra questão é que o algoritmo é bastante suscetível a ruídos na base.

2.6.1.5 Classificação baseada em regras

Um modelo de classificação baseada em regras consiste em um conjunto de expressões *SE-ENTÃO* ligados por *E* lógicos que estabelecem regras para classificar os dados. Uma regra pode ser dividia em antecedente e consequente, onde o antecedente é um teste

realizado sobre um ou mais atributos conectados por operadores lógicos *E*, e o consequente, é a predição da classe. Se uma determinada regra possui o seu antecedente verdadeiro, a regra foi satisfeita. Ainda, cada regra pode ter medidas de cobertura e precisão, onde a cobertura é a razão entre o número de tuplas[6] cobertas por uma determinada regra e o número total de tuplas do conjunto. Já a precisão é a razão entre o número de tuplas classificadas corretamente por determinada regra e o número de tuplas cobertas por esta regra. O cálculo das medidas é apresentado nas equações (4) e (5), onde $n_{cobertas}$ é o número de tuplas cobertas pela regra *R*, $n_{corretas}$ o número de tuplas classificadas corretamente por *R*; e D_{tuplas} o número de tuplas no conjunto *D* (HAN; KAMBER e PEI, 2011).

$$cobertura(R) = \frac{n_{cobertas}}{D_{tuplas}} \tag{4}$$

$$precisão(R) = \frac{n_{correta}}{n_{cobertas}} \tag{5}$$

Um exemplo de algoritmo classificador é o “uma-regra” (*1R*), que é capaz de predizer a classe utilizando apenas um atributo. Por ser um algoritmo simples, possui uma baixa complexidade, sendo capaz de descobrir boas regras. De maneira geral, o funcionamento do *1R* se dá da seguinte forma: o atributo no qual as regras são criadas é dividido em ramos que possuem valores diferentes para este atributo, sendo que no conjunto de treinamento, a classe que ocorre com maior regularidade é o que determina qual o melhor ramo. Portanto, um conjunto de regras é determinado para cada valor do atributo, sendo que a regra com menor taxa de erros é avaliada como a melhor regra (CASTRO; FERRARI, 2016).

2.6.1.6 Máquinas de Vetores de Suporte

A técnica de Máquinas de Vetores de Suporte, ou *Support Vector Machines* (SVM), possui uma abordagem estatística de aprendizagem, trazendo resultados com base na experiência e na observação dos dados, funcionando bem com dados com uma

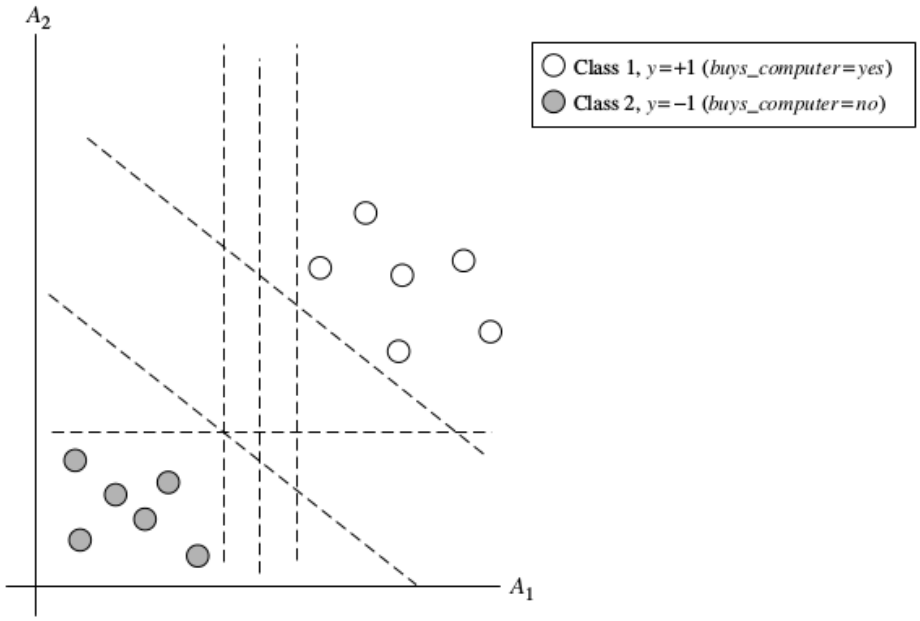
6 Uma tupla pode ser definida como um conjunto finito de elementos ordenados.

dimensionalidade alta. Basicamente o algoritmo traz um conceito de hiperplano, onde este é definido como um limite de decisão linear para a classificação. O algoritmo tenta então detectar o hiperplano de margem máxima, que é o limite de decisão com maior margem de separação dos dados, visando ter erros de generalização melhores do que aqueles com pequenas margens. (TAN; STEINBACH; KUMAR, 2009).

A aplicabilidade do SVM consiste em, de acordo com Han, Kamber e Pei (2011), resolver problemas tanto de classificação como de estimação. O tempo de treinamento do modelo pode vir a ser bastante lento, porém, esta técnica possui uma alta acurácia, sendo ela utilizada em aplicações como o reconhecimento de objetos e de fala. Outra questão relacionada ao algoritmo é que este assume que, para o caso de classificação, os dados possuem apenas duas classes.

Han, Kamber e Pei (2011) ressaltam que existem dois casos em que os dados podem ser separados. O caso em que os dados podem ser separados linearmente e os que não podem ser separados linearmente. No primeiro caso, traçando um gráfico de um conjunto de dados onde os atributos de treinamento estão associados com uma classe, conforme figura 4, é possível verificar que os registros podem ser separados por uma linha reta, de acordo com a sua categoria. O SVM então, procura neste espaço, o hiperplano de margem máxima. No caso em que os objetos não podem ser separados de forma linear, tem de ser feita uma transformação para um espaço de maior dimensionalidade, realizando um mapeamento não linear dos dados para o novo espaço. Em seguida, neste novo espaço é encontrado um hiperplano, o que leva a resolução do problema utilizando a abordagem do primeiro caso.

Figura 4 – Dados rotulados em duas classes separados linearmente



Fonte: Han, Kamber e Pei (2011, p. 409).

2.6.2 Avaliação de desempenho de classificadores

Uma forma de avaliar o desempenho de um classificador, ou seja, determinar o quanto ele é preciso, é medindo a sua habilidade preditiva através de algumas métricas de desempenho. Para isso, é recomendado verificar estas métricas sobre um conjunto de dados de teste, que consiste em exemplos que não foram vistos pelo classificador durante a fase de treinamento, para evitar um problema chamado de *overfitting* dos dados, ou seja, quando um modelo preditivo não consegue generalizar dados não vistos devido a anomalias ou ruídos incorporados do conjunto de treinamento (HAN; KAMBER; PEI, 2011).

Em se tratando de uma classificação binária, ou seja, um problema em que deseja-se prever duas classes, como no caso de fraudes de cartão de crédito, a classe positiva indica a ocorrência de uma fraude, enquanto que a classe negativa indica a ocorrência de uma situação normal, neste contexto. Para representar a classificação correta e incorreta dos objetos

pertencentes a cada classe, utiliza-se a matriz de confusão, representada na tabela 1 (TAN; STEINBACH; KUMAR, 2009).

Tabela 1 – Matriz de confusão

		Classe Prevista	
		0	1
Classe Atual	0	Verdadeiros Negativos	Falsos Positivos
	1	Falsos Negativos	Verdadeiros Positivos

Fonte: do autor, adaptado de Tan, Steinbach e Kumar (2009, p. 351).

Segundo Tan, Steinbach e Kumar (2009), os termos utilizados para compor uma matriz de confusão são os seguintes:

- Verdadeiro positivo (VP): número de exemplos positivos classificados corretamente;
- Verdadeiro negativo (VN): número de exemplos negativos classificados corretamente;
- Falso positivo (FP): número de exemplos negativos que foram classificados incorretamente como positivos;
- Falso negativo (FN): número de exemplos positivos classificados incorretamente como negativos.

Outras medidas de avaliação de um classificador podem ser resumidas na tabela 2.

Tabela 2 – Medidas de avaliação de desempenho de um classificador

Medida	**Fórmula**
Acuracidade	$\frac{VP+VN}{P+N}$
Taxa de erro	$\frac{FP+FN}{P+N}$
Revocação, taxa de verdadeiros positivos (TVP)	$\frac{VP}{P}$
Especificidade, taxa de verdadeiros negativos (TVN)	$\frac{VN}{N}$
Precisão	$\frac{VP}{VP+FP}$
Medida F, F1, média harmônica de precisão e revocação	$\frac{2\ x\ Precisão\ x\ Revocação}{Precisão+Revocação}$

Fonte: do autor, adaptado de Han, Kamber e Pei (2011, p.365).

As medidas listadas na tabela anterior tem a seguinte definição (HAN; KAMBER; PEI, 2011):

- Acuracidade: é a taxa de objetos do conjunto de testes classificados corretamente, também conhecida como taxa de reconhecimento do classificador;

- Taxa de erro: é representada pelo percentual de objetos classificados incorretamente pelo classificador;

- Revocação: medida também conhecida como taxa de verdadeiros positivos ou medida de completude, representa a proporção de objetos positivos classificados como tal;

- Especificidade: medida que por vezes é chamada de taxa de verdadeiros negativos, indica a proporção de objetos negativos classificados como tal;

- Precisão: medida de exatidão, que indica o percentual de objetos rotulados como positivo e realmente o são;

- Medida F: também chamada de F1, é a combinação das medidas de precisão e revocação através da média harmônica das mesmas, onde ambas recebem o mesmo peso.

2.6.3 Técnicas de Agrupamento

As técnicas de agrupamento consistem em separar o conjunto de dados em grupos menores, com base na sua similaridade, que pode ser determinada com base nos atributos de cada objeto ou através de medidas de distância. Diferentes técnicas de agrupamento geram diferentes grupos utilizando o mesmo conjunto de dados. As tarefas de agrupamento podem ser utilizadas para obter um conhecimento prévio do conjunto de dados, e em alguns casos, é utilizada na etapa de pré-processamento. Neste caso, após o agrupamento, outras técnicas podem ser aplicadas sobre os grupos resultantes, como, por exemplo, um algoritmo de classificação (HAN; KAMBER; PEI, 2011).

Para que os algoritmos de agrupamento possam definir grupos de objetos semelhantes entre si, necessariamente é preciso uma medida de similaridade ou distância. A maior parte dos métodos assume uma matriz $n \times m$ onde n representa os objetos e m representa os atributos de cada objeto como princípio. Outras medidas de similaridade são descritas nos tópicos a seguir (CASTRO; FERRARI, 2016):

- Matriz de distância: é uma medida comumente utilizada, onde cada elemento da matriz é representado por uma medida de proximidade;

- Medidas de dados binários: para este tipo de dados, uma medida que pode ser utilizada é a distância de *Hamming*, representada por $H = \sum_{l=1}^{m} \delta_l$, onde $\delta_l = 1$ se $x_{il} \neq x_{jl}$ e $\delta_l = 0$ para outros casos;

- Medidas para dados nominais: uma das maneiras de medir a distância entre objetos nominais é através de uma comparação simples de atributo a atributo, chegando ao cálculo desta distância por $d_{ij} = (m - M) \vee m$, onde *M* é o número de atributos em que *i* e *j* possuem o mesmo valor e *m* é o total de atributos do conjunto;

- Medidas para atributos contínuos: existem várias formas de se obter uma medida de distância ou similaridade entre os atributos contínuos ou numéricos, dentre elas, a distância euclidiana, que é uma das mais utilizadas, representada pela equação (6).

$$d_{ij} = \left(\sum_{k=1}^{m} \left(x_{ik} - x_{jk} \right)^2 \right)^{1/2} \quad (6)$$

Segundo Han, Kamber e Pei (2011), existem vários algoritmos de agrupamento disponíveis, porém, de modo geral, os métodos de grupos podem ser divididos em algumas categorias principais, conforme Tabela 1.

Tabela 3 – Categorias principais de métodos de agrupamento

Método	Características
Particionamento	Encontrar agrupamentos de formato esférico mutuamente exclusivos; Baseado em distância; Pode utilizar a média ou medoide[7] para representar o centro de um agrupamento; Eficaz para conjuntos de dados pequenos ou médios.
Hierárquico	Agrupamento é uma decomposição de vários níveis; Não consegue corrigir combinações ou divisões erradas; Pode incorporar outras técnicas como o microagrupamento.
Baseado em densidade	Pode achar formas de agrupamento arbitrários; Agrupamentos são regiões densas de objetos no espaço, separadas por regiões de baixa densidade; Agrupamento de densidade: Cada ponto deve ter um número mínimo de pontos dentro de sua “vizinhança”; Pode filtrar anomalias.
Baseado em grade	Utiliza uma estrutura de dados em forma de grade multidimensional; Tempo de processamento rápido, normalmente não depende da quantidade de objetos, mas sim do tamanho da grade.

Fonte: do autor, adaptado de Han, Kamber e Pei (2011, p. 450).

Um dos algoritmos de agrupamento mais conhecidos é o k-Médias. Este método pertence a categoria de técnicas de agrupamentos que utilizam métodos de particionamento. Os métodos de particionamento são simples e consistem em dividir um conjunto em vários agrupamentos, onde o número de grupos é informado como parâmetro inicial, representados por *k* (HAN; KAMBER; PEI, 2011).

Conforme Tan, Steinbach e Kumar (2009), o algoritmo k-Médias parte da definição do usuário do parâmetro *k*, onde este é o número de grupos desejado, ou centróides. Cada objeto do conjunto de dados é atribuído a um centróide, sendo que os objetos que ficam próximos a ele formam um grupo. Os centróides são recalculados e o processo é iterativo, se repetindo até que este não sofra mais alterações. Para a atribuição de um objeto a um centróide, uma medida de proximidade é necessária, como por exemplo a distância Euclidiana, baseada num espaço Euclidiano.

7 Em métodos de agrupamento, um medoide é um objeto que representa os demais em um grupo, ou seja, é um objeto central do *cluster*.

Castro e Ferrari (2016), explicam que o algoritmo *k*-Médias determina quando deve parar de iterar através de uma função de custo ou também conhecida como função objetivo. Outro meio de determinar o critério de parada é através de um número máximo de iterações, quando a função objetivo apresenta apenas pequenas mudanças. A função objetivo que determina a qualidade de um grupo num espaço de dados euclidiano, é feita através do cálculo da Soma do Erro Quadrado, ou SSE, representado pela equação (7).

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist\left(c_i, x\right)^2 \tag{7}$$

Onde x é um objeto, C_i é o grupo de índice i, c_i é o centróide do grupo C_i , e K é o número de grupos. Portanto o objetivo é minimizar o SSE a cada iteração do algoritmo. Para casos em que o espaço dos dados não seja Euclidiano, existem outros métodos de determinar a função objetivo, centróide e proximidade, e cada método depende de cada caso (TAN; STEINBACH; KUMAR, 2009).

2.6.4 Detecção de anomalias

A tarefa de detecção de anomalias em um conjunto de dados consiste em identificar características que diferem significativamente do padrão dos dados. Os atributos destes objetos anômalos possuem um desvio em relação aos atributos esperados em um determinado grupo. Tais anomalias também são chamadas de fatores estranhos (TAN; STEINBACH, KUMAR, 2009).

Segundo Castro e Ferrari (2016), anomalias ou valores discrepantes nos dados se referem a objetos com comportamentos diferentes do seu grupo ou amostra, podendo ser detectados de várias maneiras, como através de métodos estatísticos ou medidas de distância, de modo que o afastamento de um objeto de um agrupamento indica que este seja um desvio, um valor atípico, mas não necessariamente um ruído.

Para Han, Kamber e Pei (2011), as tarefas de detecção de anomalias e agrupamento estão relacionadas entre si, sendo que enquanto o agrupamento encontra padrões em um conjunto de dados e os organiza em grupos, a detecção de anomalias tenta encontrar desvios nestes padrões encontrados utilizando diferentes abordagens, como métodos baseados em estatística, classificação e proximidade, por exemplo.

2.6.4.1 Métodos baseados em estatística

Métodos estatísticos são baseados em modelos probabilísticos dos dados, ou seja, eles determinam por meio de probabilidade se um objeto pertence a determinado modelo estatístico ou não. Estes métodos são apresentados nas seções 2.6.4.1.1 e 2.6.4.1.2 (CASTRO; FERRARI, 2016).

2.6.4.1.1 Métodos paramétricos

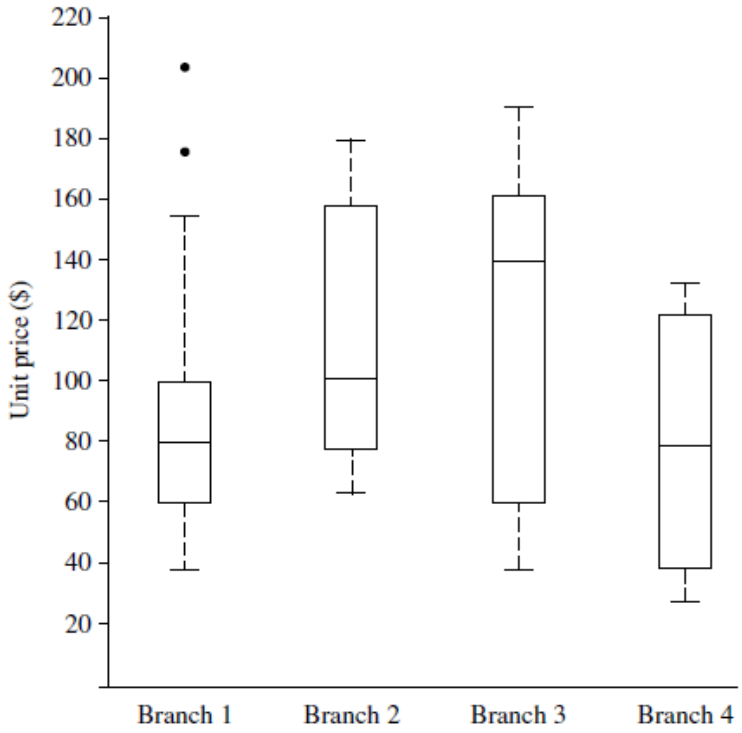
Os métodos paramétricos partem do princípio de que os dados foram gerados a partir de distribuição previamente conhecida, sendo utilizados para grandes conjuntos de dados, pois sua complexidade não está associada a quantidade de objetos contidos na base.

Um exemplo de técnica simples utilizada neste contexto é o diagrama de caixa, que é uma técnica não supervisionada, que basicamente consiste em imprimir um diagrama em forma de caixa, onde as anomalias aparecem como valores que ultrapassam os limites superiores e inferiores.

Os limites do diagrama são calculados com base na distância entre os quartis ou o desvio da média. Considerando que Φ é o limite superior e ϕ o limite inferior, pode-se

calculá-los da seguinte maneira: $\phi = Q_1 - (k \,.\, RI)$; $\Phi = Q_3 + (k \,.\, RI)$, onde Q_1 e Q_3 são os quartis, RI é o *range* interquartil e k é uma constante definida pelo usuário. A expressão $k \,.\, RI$ pode ser substituída pelo símbolo σ. Se determinado valor $x < \phi$ ou $x > \Phi$, x é considerado uma anomalia. O diagrama é exibido na Figura 5.

Figura 5 – Diagrama de caixa



Fonte: Han; Kamber e Pei(2011, p. 50).

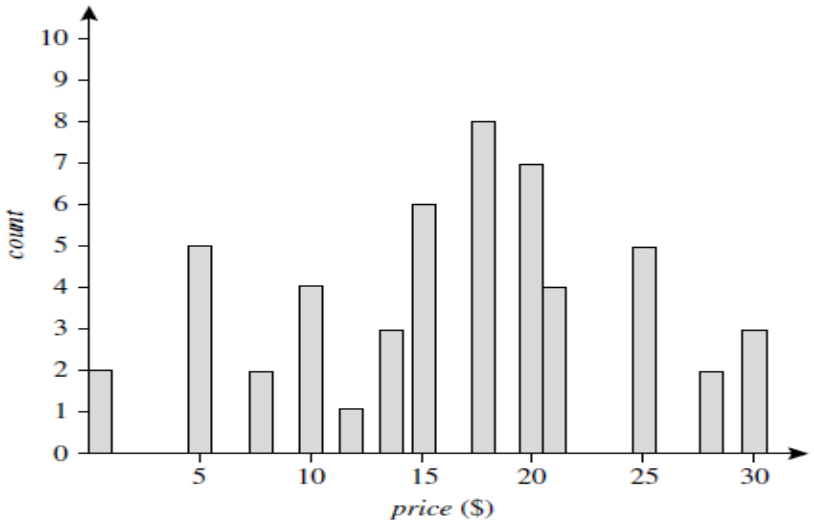
2.6.4.1.2 Métodos não paramétricos

Métodos não paramétricos assumem que não há uma distribuição previamente conhecida dos dados e que também não há um modelo que se ajuste a eles.

Um histograma é um método não paramétrico amplamente utilizado. Baseado na frequência em que determinados valores aparecem no conjunto de dados, um histograma pode

estimar a probabilidade de um objeto ocorrer. Os histogramas podem ser construídos de forma supervisionada ou não-supervisionada. Desta forma, cada atributo do conjunto de dados é utilizado para criar um histograma diferente. Para atributos categóricos, é calculada a frequência relativa, já para atributos numéricos, os valores são inseridos dentro de *k* caixas, onde a frequência dos objetos que ocorre em cada caixa é utilizada para determinar sua altura. Uma anomalia, portanto, são objetos que aparecem nas caixas em uma frequência menor do que nas demais, conforme a Figura 6. No caso de uma abordagem supervisionada, são construídos histogramas das classes pré-definidas.

Figura 6 – Histograma



Fonte: Han; Kamber e Pei(2011, p. 107).

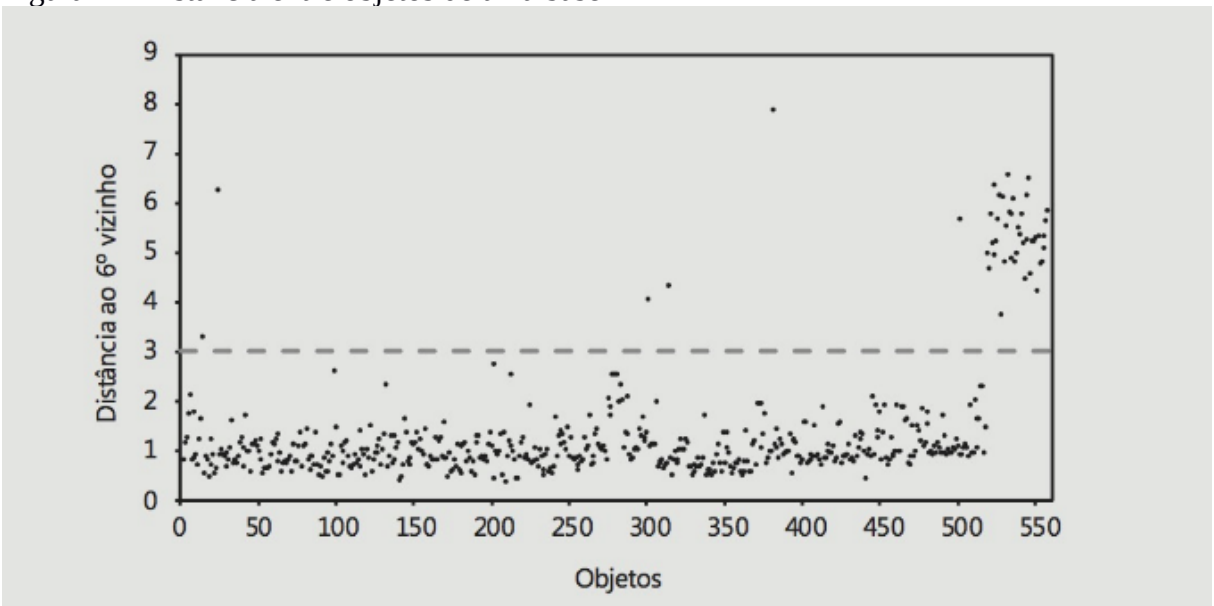
2.6.4.2 Métodos baseados em proximidade

De acordo com Castro e Ferrari (2016), Os métodos baseados em proximidade são fundamentados sobre medidas de proximidade ou similaridade entre pares de objetos do conjunto de dados. São fáceis de implementar, porém tem um alto custo computacional, pois dependem do número de objetos da base e de sua dimensionalidade.

Em uma definição simples, Tan, Steinbach e Kumar (2009), expressam que nos métodos de proximidade, as anomalias são objetos da base que estão distantes da maioria. Portanto, a partir desta abordagem, é mais fácil definir a proximidade entre objetos do que determinar uma distribuição estatística, como nos métodos estatísticos.

Um dos métodos é o algoritmo de *k* vizinhos mais próximos, ou *k-NN*, que através de uma medida de distância, como por exemplo a distância *euclidiana*, verifica a proximidade entre os objetos do conjunto. Determinando um critério, os objetos anômalos, podem ser identificados baseados no cálculo da matriz de distâncias. A Figura 7 representa um gráfico com as distâncias entre objetos de uma base, determinando as anomalias através de um critério. (CASTRO; FERRARI, 2016).

Figura 7 – Distância entre objetos de uma base



Fonte: Castro e Ferrari (2016, p. 289).

2.6.4.3 Métodos baseados em agrupamento

A detecção de anomalias está relacionada com as técnicas de agrupamento da mineração de dados. Isto pode ser determinado pela ideia de que se um objeto estiver longe de

um agrupamento, este pode ser considerado uma anomalia. O mesmo vale para agrupamentos esparsos, onde este grupo de objetos são considerados anomalias. Outras situações sobre métodos de detecção de anomalias baseadas em agrupamento são indicadas a seguir (HAN; KAMBER; PEI, 2011):

- Objetos que não pertencem a nenhum agrupamento: nas situações em que objetos estão em determinado agrupamento, valores que se encontram fora destes grupos podem ser considerados anomalias. Para esta situação, um dos métodos que podem ser utilizados é o *DBSCAN*;

- Distância do objeto em relação ao agrupamento mais próximo: através de métodos de agrupamento como o *k-means*, é possível separar os dados em agrupamentos. Os objetos que estiverem mais distantes do centro de cada agrupamento podem ser considerados anomalias.

2.6.4.4 Métodos baseados em classificação

Técnicas de classificação podem ser utilizadas para detecção de anomalias em um conjunto de dados. De modo geral, a ideia é treinar um modelo que possa classificar os objetos em dados normais ou anômalos. Porém, geralmente os métodos de classificação empregados na detecção de anomalias utilizam um modelo de apenas uma classe, ou seja, um objeto é classificado como anomalia se não pertencem a classe “normal” (HAN; KAMBER; PEI, 2011).

Segundo Castro e Ferrari (2016), métodos de classificação como árvores de decisão e classificação baseada em regras podem ser utilizadas, assumindo que a informação sobre o rótulo da classe exista previamente, definindo os objetos como normais ou anômalos.

3 TRABALHOS RELACIONADOS

Neste capítulo serão descritos alguns trabalhos relacionados com o atual problema de pesquisa, onde a metodologia proposta pelos autores será analisada, além de seus resultados, a fim de compará-los e extrair conhecimento que possa ser de alguma forma utilizado para a realização deste trabalho.

Os trabalhos que são citados neste capítulo possuem a característica principal a identificação de perdas comerciais na distribuição de energia elétrica através de processos de mineração de dados, o que serviu como base para a seleção destes estudos.

É necessário salientar que, apesar de o termo fraude ser empregado nos trabalhos que são mencionados neste capítulo, esta é apenas uma das características da perda comercial, conforme descrito na Seção 2.1. Porém, como a fraude é um requisito para a ocorrência deste tipo de perda, a pesquisa sobre trabalhos com esta abordagem se torna relevante para agregar conhecimentos no presente estudo.

Nas próximas seções deste capítulo, serão abordados os objetivos dos trabalhos selecionados, a descrição sobre a metodologia utilizada pelos autores, além de seus resultados obtidos. Por fim, será realizado um comparativo entre os trabalhos.

3.1 Descrição e objetivos

Silva e Scarpel (2007) propõem em seu artigo a utilização do algoritmo de Máquina de Vetores de Suporte (SVM), para a detecção de fraudes na distribuição de energia elétrica, baseado nos dados fornecidos por uma empresa do ramo. O objetivo é criar um modelo de classificação que possa rotular os clientes desta empresa, classificando os consumidores como

honestos ou fraudulentos. Para isso, foi utilizado um conjunto de dados contendo 596 registros divididos em duas partes iguais: uma contendo clientes fraudadores e outra com clientes que não cometeram fraude. Outra questão proposta pelos autores foi a comparação entre o SVM e a Análise Discriminante Linear, que segundo eles, é o método quantitativo mais empregado para detectar práticas fraudulentas, baseando-se na classificação de observações.

Todesco et al. (2007), propõem o desenvolvimento de uma aplicação para identificar fraudadores de energia elétrica, no sentido de melhorar o processo de inspeção de consumidores suspeitos. Para isso, os autores utilizam a mineração de dados sobre uma base fornecida por uma empresa distribuidora de energia. O processo foi empregado sobre dados de consumidores residenciais e comerciais, sendo que, para determinar se um consumidor executa atividade fraudulenta, os autores propuseram o uso de duas variáveis obtidas através de cálculos baseados no consumo de energia: *score* e *score* acumulado. Com base nestas informações, os autores desenvolveram um protótipo de uma aplicação para permitir que o usuário possa interagir com os dados e verificar por meio de uma interface gráfica, quais os consumidores são candidatos a inspeção.

Ferreira (2008), apresenta em sua tese a utilização de técnicas de aprendizado de máquina para identificar perdas comerciais sobre os dados de inspeções de uma concessionária de energia elétrica. O autor aplicou quatro tipos diferentes de técnicas de aprendizado de máquina, sobre diferentes conjuntos de dados, além de compará-las para verificar qual estratégia gerou o melhor resultado. Desta forma, as técnicas utilizadas são métodos baseados em classificação, sendo eles o SVM, Redes Neurais Artificiais, o algoritmo de árvore de decisão *C4.5* e o *Näive Bayes.* Segundo o autor, o objetivo é utilizar estas ferramentas para avaliar qual a melhor técnica a ser utilizada para detectar as perdas comerciais, melhorando consequentemente o processo de inspeção da empresa.

3.2 Metodologia e resultados

Silva e Scarpel (2007), utilizaram os atributos do conjunto de dados relacionados na Tabela 2 para a aplicação do método proposto. A técnica de SVM utilizada foi avaliada de acordo com o uso de dois tipos de funções: linear e polinomial de segundo grau. Ainda, conforme citam os autores, foi incluída uma constante de custo adicional *C* às funções, para que o modelo estivesse tolerância a ruídos e erros oriundos da base, no entanto, a etapa de pré-processamento dos dados não foi executada.

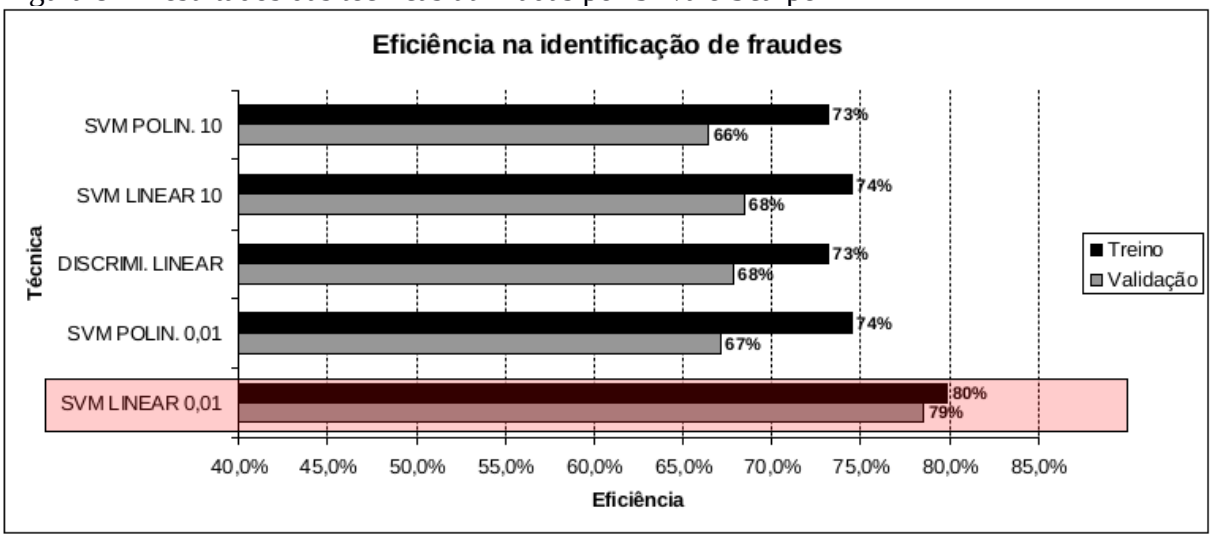
Tabela 4 – Atributos selecionados para aplicação dos algoritmos

Atributo	Descrição
Tempo de residência	Quantidade de anos que o cliente reside no local
Valor última conta	Valor em reais, da última conta do cliente
Valor conta média	Valor médio da conta do cliente, em relação aos últimos seis meses
Região	Descrição da região em que o cliente reside
Forma pagamento	Descreve se cliente optou por pagamento por meio de débito em conta
Variação valor última conta	Razão entre valor da conta do mês atual e valor da conta do mês anterior

Fonte: do autor, adaptado de Silva e Scarpel (2007).

A base de dados de Silva e Scarpel (2007), foi dividida aleatoriamente em um conjunto de treinamento e outro de teste. Para a verificação dos resultados, os autores utilizaram uma matriz de confusão para avaliar os métodos empregados, ajustando o valor da constante de custo entre 0,1 e 10, tanto no SVM de função linear quanto na função polinomial de segundo grau. Para o método de Análise Discriminante Linear, os autores estabeleceram um limiar de 0,5 para classificar o cliente em fraudador ou honesto. Na Figura 8, os autores demonstram os resultados de cada técnica, concluindo que o SVM de função linear, utilizando constante de custo de 0,1 obteve os melhores resultados para classificar os clientes fraudadores, chegando a 80 % de registros classificados corretamente no conjunto de treinamento.

Figura 8 – Resultados das técnicas utilizadas por Silva e Scarpel



Fonte: Silva e Scarpel (2007, p. 11).

Para a construção do modelo de detecção de fraudes, conforme artigo elaborado por Todesco et al. (2007), foi utilizada a medida de *score*, que mantém a característica da sazonalidade do consumo de energia, que pode variar de acordo com o período em que a leitura está sendo executada. Outra variável utilizada foi o *score* acumulado, que é o somatório do *score* no período de doze meses para o mês atual, apresentado nas equações (8) e (9).

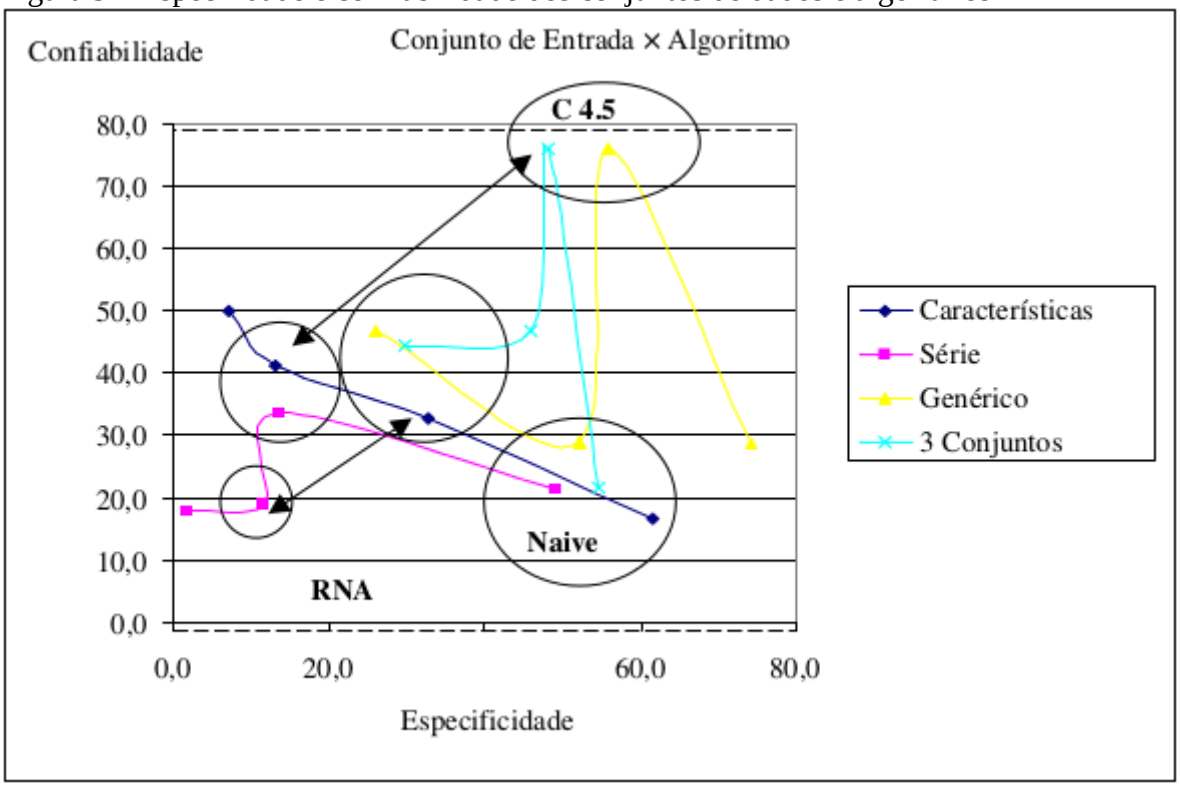
$$score_{mensal} = inteiro\left(\left(\frac{leitura_{mêsanoanterior} - leitura_{mêsatual}}{leitura_{mêsanoanterior} + 1}\right) x\, 10\right) \tag{8}$$

$$score_{acumulado} = \sum_{i=0}^{11} score_i \tag{9}$$

Conforme Todesco et al. (2007), quanto maior o *score* acumulado, maior é o indicativo de ocorrência de fraude. Os consumidores então são divididos em grupos de não suspeitos, indefinidos e suspeitos de fraude de acordo com os intervalos definidos para cada grupo. Estes intervalos baseiam-se no valor de *score* acumulado. Por fim, os resultados puderam ser visualizados no protótipo desenvolvido. Os autores estimam uma taxa de acerto de 63,64% para consumidores residenciais e 80,42%, considerando um *score* acumulado acima de 18.

Ferreira (2008), utilizou duas bases de dados para aplicar as técnicas de aprendizado de máquina: uma base com atributos de consumidores e outra com resultados das inspeções. O autor destaca que os conjuntos de dados de entrada foram divididos em três tipos, sendo eles o tipo série, o tipo características e o tipo genérico. Conforme o autor exemplifica, as séries são séries de tempo referente ao consumo em kWh dos clientes dos últimos 59 meses. Já as caraterísticas são um conjunto de dezessete atributos extraídos da série temporal de consumo de energia. O conjunto de dados genérico contém demais atributos como localização do cliente, classe de consumo, situação, média diária de consumo, entre outros. Para a saída dos dados, o autor definiu duas classes: normal e fraude. Para comparar as ferramentas de classificação utilizadas sobre os conjuntos de dados mencionados, o autor utilizou o princípio de dominância de Pareto, que envolve duas medidas diferentes: especificidade e confiabilidade. Conforme a figura 9, o método de classificação que obteve uma melhor taxa de confiabilidade e especificidade foi o algoritmo C4.5, e o conjunto de dados em que foram obtidos os melhores resultado foi o genérico.

Figura 9 – Especificidade e confiabilidade dos conjuntos de dados e algoritmos



Fonte: Ferreira (2008, p. 71).

3.3 Comparativo

Os trabalhos mencionados neste capítulo apresentaram resultados satisfatórios, comprovando que algumas técnicas de mineração de dados podem ser utilizadas para identificar perdas comerciais. Como pode-se observar, diferentes metodologias podem ser utilizadas para resolver o mesmo problema, sendo que o campo de mineração de dados possui várias técnicas e abordagens, as quais muitas delas não foram exploradas nos estudos destacados neste capítulo.

Uma característica que pode ser observada nos referidos trabalhos, é a de que não foram mencionadas tecnologias ou ferramentas de software utilizadas para a aplicação das técnicas de mineração de dados. Outra questão que pode ser considerada, é a de que diferentes abordagens geraram bons resultados, conforme destacado nos trabalhos de Silva e Scarpel (2007), e Ferreira (2008), em que foram utilizados os algoritmos de classificação SVM e C4.5 respectivamente. No caso de Todesco et al. (2007), um modelo de classificação de autoria própria também obteve resultados satisfatórios. Na tabela 3, é possível verificar as diferentes abordagens utilizadas pelos autores.

Tabela 5 – Comparativo entre metodologias, objetivos e técnicas dos autores

Autor(es)	Silva e Scarpel	Todesco et al.	Ferreira
Objetivos	- Detectar fraudes na distribuição de energia através de mineração de dados; - Comparar duas técnicas de classificação	- Desenvolvimento de uma aplicação para identificar fraudes na distribuição de energia; - Utilizar por métodos de mineração de dados quais consumidores serão candidatos a inspeção	- Determinar, dentro de um conjunto de técnicas, qual a melhor para identificar perdas comerciais; - Melhorar o processo de inspeção
Técnicas	- SVM	- Classificação de fraudes baseado em um valor de *score*.	- C4.5; - RNA; - Näive Bayes; - SVM
Metodologia	- Base de dados dividida em conjuntos de treinamento e teste; - Aplicação de SVM com função linear e polinomial	- Cálculo de valor de *score* acumulado, baseado em dados de consumo de clientes comerciais e residenciais	- Utilização de dois conjuntos de dados diferentes; - Comparação entre as técnicas através de dominância de Pareto

Autor(es)	Silva e Scarpel	Todesco et al.	Ferreira
Resultados	80% dos registros classificados corretamente como fraudadores	63,64% de acerto para clientes residenciais e 80,42% para clientes comerciais, considerando o valor 18 como limiar para o *score* acumulado	Algoritmo C4.5 obtém melhores resultados, com índice de confiabilidade de 75,8 e especifidade de 56,0

Fonte: do autor.

No capítulo seguinte serão descritas as ferramentas e bibliotecas utilizadas para executar o desenvolvimento do projeto, como o pré-processamento e a execução dos algoritmos de mineração de dados. Uma breve descrição das ferramentas pesquisadas será realizada, além de seus recursos disponíveis para alcançar o objetivo deste trabalho.

4 TECNOLOGIAS E FERRAMENTAS UTILIZADAS

Este capítulo visa descrever as principais tecnologias e ferramentas utilizadas para o desenvolvimento deste trabalho, ou seja, para a aplicação dos algoritmos de mineração sobre os dados e também para a análise dos mesmos.

Nas seguintes seções serão descritas as ferramentas utilizadas para aplicação dos algoritmos de mineração, sendo elas o WEKA e o scikit-learn, bem como os métodos utilizados para a extração do conjunto de dados. Também será feito o detalhamento da linguagem de programação Python, muito utilizada no contexto de análise de dados, possuindo um vasto número de bibliotecas para este fim, como por exemplo, a biblioteca Scipy, muito útil para realizar o pré-processamento e a análise descritiva dos dados.

4.1 WEKA

O WEKA, sigla para *Waikato Environment for Knowledge Analysis*, é um software multiplataforma desenvolvida pela Universidade de Waikato, Nova Zelândia. Seu objetivo é proporcionar um conjunto de ferramentas para aplicação de técnicas de aprendizado de máquina e pré-processamento de dados, garantindo assim todo o ferramental necessário para a execução do processo de mineração de dados, incluindo os métodos de classificação, regressão, agrupamento e associação de regras (WITTEN; FRANK, 2005).

Conforme descrevem Witten e Frank (2005), existem diversas interfaces disponibilizadas para os usuários do WEKA, entre elas, três interfaces gráficas e uma interface de linha de comando. O software disponibiliza também a sua API em Java, caso o usuário queira desenvolver a sua própria aplicação de mineração de dados.

4.1.1 Formato ARFF

O formato ARFF é o formato de arquivo utilizado para representar o conjunto de dados e seus atributos, nativo do WEKA. Usualmente, os valores extraídos de uma base de dados estão em formato CSV, porém existem várias ferramentas disponíveis para conversão para este formato específico. No formato ARFF, existem *tags* que são utilizados como marcação para o processamento do arquivo, utilizando o símbolo @ com uma determinada palavra-chave. Para a declaração dos atributos, existem quatro tipos de dados, sendo eles o tipo nominal, o tipo numérico, o tipo *string* e o tipo data, sendo precedidos pela *tag @attribute*. O tipo de dados, no caso de *string*, data e numérico deve ser explicitamente declarado no arquivo. Dados do tipo nominal são declarados entre chaves. Para a representação de valores ausentes, é utilizado um ponto de interrogação. Após a declaração dos atributos, encontra-se a *tag @data*, que indica o início dos objetos ou instâncias da base. A *tag @relation* pode ser encontrada no início do arquivo, indicando o nome da relação utilizada. Um exemplo do formato ARFF pode ser visto na Figura 10 (WITTEN; FRANK, 2005).

Figura 10 – O Formato ARFF

```
% ARFF file for the weather data with some numeric features
%
@relation weather

@attribute outlook { sunny, overcast, rainy }
@attribute temperature numeric
@attribute humidity numeric
@attribute windy { true, false }
@attribute play? { yes, no }

@data
%
% 14 instances
%
sunny, 85, 85, false, no
sunny, 80, 90, true, no
overcast, 83, 86, false, yes
rainy, 70, 96, false, yes
rainy, 68, 80, false, yes
rainy, 65, 70, true, no
overcast, 64, 65, true, yes
sunny, 72, 95, false, no
sunny, 69, 70, false, yes
rainy, 75, 80, false, yes
sunny, 75, 70, true, yes
overcast, 72, 90, true, yes
overcast, 81, 75, false, yes
rainy, 71, 91, true, no
```

Fonte: Witten e Frank (2005, p. 54).

4.1.2 A interface *Explorer*

Uma das principais interfaces gráficas do WEKA é o *Explorer.* Nele é possível selecionar diferentes tarefas de mineração de dados divididas em seis abas no topo do painel. Na aba que corresponde a tarefa de pré-processamento, é possível carregar o arquivo ARFF para a ferramenta, que exibirá informações sobre os dados em relação ao conjunto carregado, como histogramas e frequências dos atributos do arquivo. Além disso, é permitido fazer a edição e remoção de atributos. Na aba correspondente a tarefa de classificação, é possível selecionar diferentes técnicas e algoritmos, assim como nas abas relacionadas a tarefas de agrupamento e associação. Uma vez que o algoritmo selecionado foi executado, o WEKA exibe uma tela de saída referente a tarefa selecionada. A Tabela 4 exibe os algoritmos disponíveis no WEKA separados por tarefa de mineração de dados.

Tabela 6 – Algoritmos disponíveis no WEKA separados por tarefa

Tarefa	Algoritmo	Descrição
Classificação	AODE	Estimadores de dependência única
	BayesNet	Aprendizado de redes bayesianas
	ComplementNaiveBayes	Classificador Complementar Naive Bayes
	NäiveBayes	Classificador Naive Bayes probabilístico padrão
	NäiveBayesMultinomial	Versão multinomial do Naive Bayes
	NäiveBayesSimple	Implementação simples do Naive Bayes
	NäiveBayesUpdateable	Naive Bayes incremental que aprende uma instância por vez
	ADTree	Árvores de decisão alternadas
	DecisionStump	Árvores de decisão de um nível
	ID3	Algoritmo básico de árvore de decisão de divisão e conquista
	J48	Árvore de decisão de aprendizado C4.5 (implementa C4.5 versão 8)
	LMT	Árvores de modelo de logístico
	M5P	Modelo de árvore de aprendizado M5’
	NBTree	Árvore de decisão com classificador Naive Bayes nas folhas
	RandomForest	Florestas aleatórias
	RandomTree	Árvore que considera um determinado número de recursos aleatórios em cada nó
	REPTree	Árvore de aprendizado rápido que utiliza poda com erro reduzido
	UserClassifier	Permite que usuários construam suas próprias árvores
	ConjunctiveRule	Aprendizado de regras conjuntivas simples
	DecisionTable	Tabela de decisão simples com classificação majoritária
	Jrip	Algoritmo *RIPPER* para indução de regras rápidas e efetivas
	M5Rules	Obter regras de modelos de árvores construídas utilizando M5’
	Nnge	Método de vizinhos mais próximos para gerar regras utilizando exemplos generalizados não aninhados
	OneR	Classificador 1R
	Part	Obtém regras de árvores de decisão parciais construídas utilizando J4.8
	Prism	Algoritmo de cobertura simples para regras
	Ridor	Aprendizado com regra *Ripple-down*
	ZeroR	Prediz a classe majoritária (se nominal) ou o valor médio (se numérico)
	LeastMedSq	Regressão robusta que utiliza a mediana ao invés da média
	LinearRegression	Regressão linear padrão
	Logistic	Modelo de regressão logística linear
	MultilayerPerceptron	Rede neural com *Backpropagation*
	PaceRegression	Modelo de regressão linear usando regressão de ritmo
	RBFNetwork	Implementa uma rede de função de base radial
	SimpleLinearRegression	Modelo de regressão linear baseado em um atributo único
	SimpleLogistic	Modelo de regressão logística linear com seleção de atributos embutido
	SMO	Algoritmo de otimização mínimo sequencial para suporte classificação vetorial
Tarefa	**Algoritmo**	**Descrição**
	SMOreg	Algoritmo de otimização mínimo seqüencial para suporte regressão vetorial
	VotedPerceptron	Algoritmo de *Perceptron* votado
	Winnow	Perceptron orientado a erros com atualizações multiplicativas
	IB1	Vizinho próximo com aprendizado baseado em instâncias
	Ibk	Classificador *kNN*
	Kstar	Vizinho próximo com função de distância generalizada
	LBR	Classificador bayesiano de regras com abordagem *lazy learner*

	LWL	Algoritmo geral para aprendizagem ponderada localmente
	Hyperpipes	Aprendizado rápido, extremamente simples baseado em hipervolumes em espaço de instâncias
	VFI	Métodos de intervalos de função de votação, simples e rápidos
Agrupamento	EM	Agrupamento que usa maximização de expecativa
	CobWeb	Implementa os algoritmos CobWeb e Classit
	FarthestFirst	Usa o algoritmo de primeiro percurso mais longo
	MakeDensityBasedClusterer	Um agrupador que retorna a distribuição e densidade
	SimpleKMeans	Utiliza o método K-médias
Associação	Apriori	Encontra regras de associação usando o Apriori
	PredictiveApriori	Encontra regras ordenadas por precisão de predição
	Tertius	Descoberta de regras por confirmação

Fonte: do autor, adaptado de Witten e Frank (2005, p. 404-405, 419).

O painel de visualização na interface *Explorer* ajuda o usuário na visualização de dados referente ao próprio conjunto selecionado, exibindo uma matriz de diagramas de dispersão de duas dimensões, para cada par de atributos do conjunto de dados (WITTEN; FRANK, 2005).

4.1.3 A interface *Knowledge Flow*

A interface *Knowledge Flow* é uma alternativa ao *Explorer*, permitindo uma visão de como ocorre o fluxo de dados no WEKA. O usuário pode selecionar os componentes que deseja construindo um grafo dirigido que processa e analisa os dados. O *Knowledge Flow* também permite processar *streaming* de dados, sendo que esta funcionalidade não é permitida no painel *Explorer* (WITTEN; FRANK e HALL, 2016).

Os componentes presentes na interface *Knowledge Flow* são semelhantes aos utilizados na interface *Explorer*, como classificadores, agrupadores e associadores. Cada componente é executado em *threads* separadas nesta interface, com exceção em que os dados são processados de forma incremental (WITTEN; FRANK e HALL, 2016).

4.1.4 A interface *Experimenter*

Esta GUI implementa uma interface avançada para os usuários do WEKA. Neste ambiente é possível executar experimentos em larga escala, permitindo armazenar as estatísticas de performance da execução em arquivos ARFF. Também é possível exibir os resultados em formato CSV (WITTEN; FRANK e HALL, 2016).

A interface *Experimenter* possui três painéis onde é possível realizar a configuração do experimento, executar o experimento e analisar o resultado dele. Ao executar o experimento, o mesmo gera um arquivo de saída com os resultados, incluindo o número de instâncias utilizadas para treinamento e testes, o percentual de instâncias classificadas corretamente e incorretamente, média de erro absoluto, entre outros. Para analisar e comparar experimentos, o painel de análise pode ser utilizado.

O *Experimenter* também permite executar tarefas distribuídas em diferentes *hosts*, utilizando uma *engine* baseada em Java, permitindo enviar os resultados para uma base de dados centralizada, utilizando um driver JDBC para conexão com a base.

4.1.5 A interface de linha de comando

A CLI oferece ao usuário uma interface simples para utilizar as funcionalidades básicas do WEKA, permitindo invocar comandos disponibilizados pela ferramenta, como algoritmos de mineração de dados por exemplo, através dos pacotes de classes Java disponíveis. É possível executar o próprio WEKA pela interface de linha de comando do sistema operacional, porém, a ferramenta de comandos do WEKA facilita a chamada de funções para o usuário através de comandos reduzidos, não sendo necessário especificar o nome de classe totalmente qualificada (WITTEN; FRANK e HALL, 2016).

4.2 Python

Python é uma linguagem de programação interpretada, orientada a objetos, multiplataforma, de propósito geral, gratuita e de código aberto. Foi criada no Instituto Nacional de Pesquisa em Matemática e Ciência da Computação, por Guido van Rossum, na Holanda, baseada em uma linguagem chamada ABC, que tem como base BASIC e Pascal. O Python foi concebido para ser uma linguagem de programação altamente legível e por sua simplicidade, é de fácil aprendizado (TELLES, 2008).

Por ser uma linguagem interpretada, o Python pode ser executado de forma mais lenta do que programas feitos por linguagens compiladas, como por exemplo Java e C++. Em alguns casos de aplicações em que o sistema deve ter uma resposta em tempo real, ou com baixa latência, linguagens compiladas são mais performáticas. Porém, o Python ganha vantagem em produtividade, fazendo com que os programadores optem por esta linguagem a outras de mais baixo nível, mesmo que em alguns casos se tenha um tempo de execução mais alto (MCKINNEY, 2012).

Beazley (2009), descreve algumas características da linguagem Python, como o uso de identação em vez de chaves para indicar diferentes blocos de código, como o corpo de funções, laços de repetição, classes, entre outros. Outra característica importante é que todos os dados armazenados em um programa escrito em Python utilizam o conceito de objeto, porém, o usuário pode criar os próprios objetos na forma de classes. O paradigma de programação funcional também está presente fortemente na linguagem.

McKinney (2012), cita que a partir do início dos anos 2000, houve um aumento crescente na adoção da linguagem Python para propósitos de computação científica, tanto nos meios acadêmicos como na indústria. Nos últimos anos, Python tornou-se uma alternativa para tarefas de análise de dados, principalmente devido a seu crescente número de bibliotecas para fins científicos, como o Pandas e NumPy.

4.2.1 Pandas

A biblioteca Pandas transforma o Python em um ambiente rico para análise de dados. Ela oferece funções e estruturas de dados valiosas no intuito de facilitar o trabalho com dados estruturados, disponibilizando métodos para junção de dados, limpeza e transformação, por exemplo. Outra facilidade que a biblioteca provém é a capacidade de manipular dados de uma forma flexível como em planilhas eletrônicas ou bancos de dados relacionais (MCKINNEY, 2012).

Existem dois principais tipos de estruturas de dados no Pandas, conforme descreve McKinney (2012), que são a base da maioria das tarefas na biblioteca:

- *Series: e*sta estrutura de dados é um objeto parecido com um vetor de apenas uma dimensão que contém um vetor de dados e outro vetor chamado de índice. A estrutura *Series* também pode ser vista como um dicionário de tamanho fixo. Por padrão, o vetor de índice do objeto mostra o número do índice de cada posição do vetor de dados. Porém, é possível informar uma *label* para o índice e também acessar as posições do vetor de dados através desta *label* especificada pelo usuário;

- *DataFrame: u*m *DataFrame* é uma estrutura que se assemelha a uma planilha eletrônica, de forma tabular, contendo uma coleção ordenada de colunas que podem assumir diversos tipos de dados, nos quais possuem um índice para cada linha e coluna. Uma forma de criar um *DataFrame* pode ser declarando um dicionário com listas de igual tamanho. Também é possível importar um arquivo CSV dentro de um *DataFrame*.

4.2.2 NumPy

NumPy, que significa *Numerical Python* ou Python Numérico, é a base da computação científica para o ambiente Python. Algumas características da ferramenta são (MCKINNEY, 2012):

- O objeto *ndarray* que é um vetor multidimensional rápido e eficiente;
- Funções para executar operações matemáticas entre vetores;
- Suporte a Transformações de Fourier, Álgebra Linear e geração de números aleatórios;
- Ferramentas de integração de código C, C++ e Fortran ao Python.

4.2.3 Matplotlib

Matplotlib é uma biblioteca para visualização de dados 2D, similar ao MAT-LAB, suportando vários tipos de plotagens, tais como por exemplo, um gráfico de duas dimensões. As visualizações geradas pela biblioteca são interativas, permitindo que o usuário possa manipulá-las e interagir com elas. Em geral, é um ambiente interativo para exploração, utilizado principalmente no processo de análise exploratória dos dados (MCKINNEY, 2012).

4.2.4 IPython

A ferramenta *IPython* é um projeto que surgiu em 2001, com o intuito de melhorar o interpretador Python e torná-lo mais interativo. Atualmente, é uma das mais importantes bibliotecas voltadas para computação científica do ecossistema Python. *IPython* foi projetado para aumentar a produtividade tanto da computação interativa como do desenvolvimento de software. A biblioteca traz uma abordagem exploratória, o que facilita o processo de análise de dados, já que a maior parte deste processo este comportamento, com métodos de tentativa e erro (MCKINNEY, 2012).

McKinney (2012), destaca que o ambiente interativo do *IPython* pode ser integrado com a biblioteca M*atplotlib* e outras ferramentas de interface gráfica. A vantagem de utilizar a biblioteca *Matplotlib* dentro do *IPython* é que ele não bloqueia a sessão interativa enquanto a janela de plotagem está ativa, permitindo mais interatividade por parte do usuário.

4.2.5 Scikit-learn

Scikit-learn é um módulo baseado em Python, de código aberto, que implementa vários algoritmos de aprendizado de máquina conhecidos, desenvolvido para resolver problemas utilizando abordagens de aprendizado supervisionado e não supervisionado. O pacote *scikit-learn* tem como objetivo proporcionar uma interface de alto nível de algoritmos de aprendizado de máquina para usuários que não são especialistas na área. Além disto, necessita de poucas dependências é licenciado sobre a licença BSD simplificada, que encoraja o seu uso tanto para fins acadêmicos quanto para propósitos comerciais (PEDREGOSA et al., 2011).

Segundo Pedregosa et al. (2011), o pacote *scikit-learn* é baseado no ecossistema científico da linguagem Python, portanto, ele pode ser integrado com facilidade a outros pacotes que não estão no escopo da área de análise de dados. Uma das tecnologias utilizadas

pelo pacote de aprendizado de máquina é o *NumPy* descrito na seção 4.2.2. Outra tecnologia utilizada pelo pacote é o *Scipy*, que proporciona algoritmos para representação de matrizes, álgebra linear, funções estatísticas, entre outros. Além do *Scipy*, o módulo *Cython* é utilizado para combinar a linguagem Python com a linguagem C.

No capítulo seguinte, serão descritos os procedimentos realizados durante o desenvolvimento do projeto utilizando as ferramentas pesquisadas, visando alcançar os objetivos deste trabalho.

5 DESENVOLVIMENTO DO PROJETO

Neste capítulo são descritos os procedimentos realizados durante o desenvolvimento do projeto que visaram atingir os objetivos propostos. Nas próximas seções são apresentados os dados de entrada e seus respectivos atributos, além dos processos realizados inerentes a qualidade dos mesmos, com a etapa de mineração de dados, onde os algoritmos foram executados, além das ferramentas utilizadas para este fim.

Neste capítulo também é apresentado o comparativo entre as ferramentas WEKA e scikit-learn, que foram apresentadas no Capítulo 4. Os resultados são discutidos de modo a verificar qual algoritmo obteve o melhor desempenho, além de descrever de que forma este desempenho foi avaliado, de acordo com cada conjunto de dados utilizado.

5.1 Conjuntos de dados de entrada

A empresa de distribuição de energia elétrica que forneceu os dados para o presente trabalho, possui uma grande quantidade de registros armazenados em sua base de dados. A seleção do melhor conjunto de dados de entrada para a execução dos algoritmos de mineração de dados pode ser uma tarefa extremamente complexa, tendo em vista a variedade dos dados existentes na base, além do fato de que existem diversas combinações de atributos que podem ser realizadas, obtendo-se resultados diferentes, como discutido na seção 5.4.

Foi consultado um especialista da área de inspeção de medições, setor da empresa responsável por verificar supostas irregularidades no consumo de energia com o objetivo de detectar possíveis perdas comerciais. Com a ajuda deste profissional, foram definidos alguns atributos que foram utilizados no conjunto de dados de entrada, como por exemplo, os registros contendo o histórico de consumo dos consumidores de energia elétrica.

É importante ressaltar que segundo o especialista, as perdas comerciais ocorrem em sua grande maioria entre consumidores de baixa tensão, que representam a maior parte dos clientes da empresa. Como a ocorrência de irregularidades em consumidores de alta tensão é quase nula, os dados utilizados limitaram-se a consumidores de baixa tensão, também conhecidos como consumidores do grupo B. Isto porque as UCs desta modalidade de consumo são monitoradas em tempo real através de medidores inteligentes, tornando difícil a ocorrência de uma irregularidade. Portanto, as principais características selecionadas para compor o conjunto de dados são:

- Histórico do consumo: conjunto de 24 leituras realizadas na unidade consumidora, medidas em KWh;

- Região geográfica: código da região onde está localizada a unidade consumidora, classificada por município;

- Classe: classe ou categoria de consumo da unidade consumidora;

- Registro da perda comercial: indicador de ocorrência ou não da perda comercial, registrada em data posterior aos registros de histórico de consumo selecionados.

Para complementar o conjunto de características dos dados, alguns atributos foram selecionados de forma empírica pelo autor, que correspondem principalmente a informações relativas ao próprio consumidor, além dos mencionados anteriormente. A tabela 7 contém a descrição das características utilizadas nos primeiros experimentos, totalizando 34 atributos.

Os dados brutos foram extraídos de uma base relacional através de comandos de consulta SQL, porém o atributo que indica a ocorrência de uma perda comercial foi inserido no conjunto com base em relatórios gerados pelo sistema gerencial da empresa. Estes dados foram exportados para arquivos de formato CSV, sendo que posteriormente passaram por tarefas de pré-processamento. É necessário enfatizar que o atributo que corresponde a perda

comercial representa a ocorrência da mesma num período de 24 meses. As demais características atribuídas aos dados do consumidor são referentes ao mesmo período em que as medições do consumo foram coletadas.

Tabela 7 – Conjunto de atributos selecionados para o conjunto de dados de entrada

Nome do atributo	Descrição	Tipo de dado
L1 - L24	Colunas contendo as leituras efetuadas na unidade consumidora medidas em KWh, realizadas em um período de 24 meses, onde cada leitura representa uma coluna no conjunto de dados	Quantitativo - Numérico
IDREGIAO	Região geográfica representada pelo município onde está localizada a unidade consumidora	Qualitativo - Nominal
IDCLASSE	Classe de consumo na qual o consumidor está associado, podendo ser do tipo residencial, comercial, rural, industrial ou poder público	Qualitativo - Nominal
DATA_CADASTRO	Representa o período de data em que o consumidor foi cadastrado	Qualitativo - Ordinal
FONE1	Atributo que indica se o consumidor forneceu um número de telefone válido no cadastro	Qualitativo - Nominal
ESTADO_CIVIL	Representa o estado civil do consumidor	Qualitativo - Nominal
SEXO	Representa o sexo do consumidor	Qualitativo - Nominal
DATA_NASCIMENTO	Atributo que indica o ano de nascimento do consumidor dentro de um intervalo de anos	Qualitativo - Ordinal
PREF_DEB_CONTA	Atributo que indica se o consumidor optou por efetuar o pagamento de sua fatura de energia pela modalidade de débito em conta	Qualitativo - Nominal
MULTA_ATRASO	Atributo que indica se o consumidor já pagou multa por atraso no pagamento de sua fatura de energia, nos últimos 24 meses	Qualitativo - Nominal
PC	Atributo alvo que indica ou não a ocorrência de uma perda comercial, na qual o algoritmo de mineração de dados deve predizer	Qualitativo - Nominal

Fonte: do autor.

Para a execução dos experimentos, uma pequena amostra dos dados foi utilizada devido a questões de performance, contudo é importante destacar que o custo computacional e a escalabilidade dos algoritmos não foram analisados neste trabalho, tendo em vista que o objetivo foi verificar as métricas de acuracidade das técnicas de mineração, que são discutidas na seção 2.6.2. A amostra utilizada contém um total de 513 registros, sendo que entre eles 113

objetos representam a ocorrência de perda comercial, enquanto que o restante dos dados da amostra não representam uma perda comercial.

O conjunto de características selecionadas originalmente foi dividido em mais outros quatro conjuntos, com o objetivo de verificar a possibilidade de encontrar melhores resultados combinando atributos diferentes, além de reduzir a dimensionalidade dos dados. Para cada novo subconjunto de características, foi gerado um novo arquivo com o mesmo número de registros do conjunto original, também divididos em um conjunto de treinamento e teste. Os atributos de cada subconjunto são os que seguem:

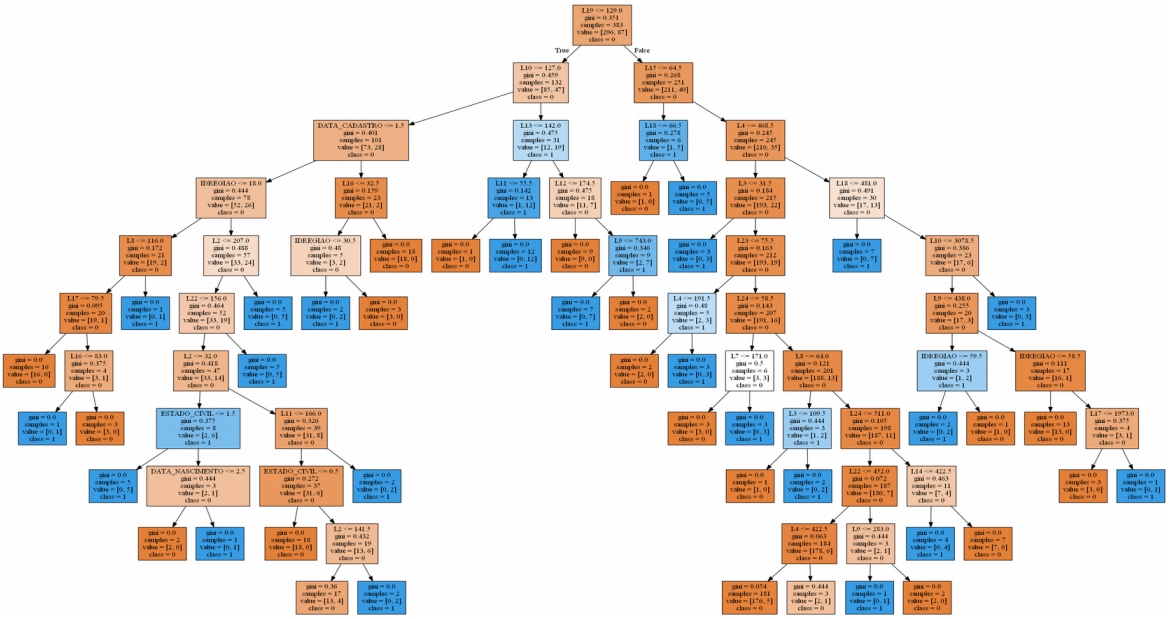
- Primeiro subconjunto: arquivo CSV contendo os dados com todos os atributos relacionados na tabela 7;

- Segundo subconjunto: arquivo CSV com os atributos IDREGIAO, IDCLASSE, DATA_CADASTRO, FONE1, ESTADO_CIVIL, SEXO, DATA_NASCIMENTO, PREFERENCIA_DEB_CONTA, MULTA_ATRASO e PC, utilizando como critério de escolha os atributos qualitativos;

- Terceiro subconjunto: arquivo CSV contendo os atributos L1 até L24, IDREGIAO, ESTADO_CIVIL, DATA_CADASTRO, DATA_NASCIMENTO e PC, definidos utilizando um algoritmo de árvore de decisão, conforme ilustra a figura 11 (APÊNDICE A);

- Quarto subconjunto: arquivo CSV contendo os atributos L1 até L24, além da classe PC, tendo como critério de escolha apenas os atributos numéricos.

Figura 11 – Árvore de decisão gerada através da biblioteca scikit-learn



Fonte: do autor. Imagem disponível em uma maior resolução no Apêndice A.

Na seção 5.2 são apresentados os procedimentos relativos ao pré-processamento realizados nos dados, tratando dos processos executados para cada atributo do conjunto de características, no sentido de melhorar a qualidade dos dados de entrada, além de preparar os dados para sua posterior utilização nas ferramentas de mineração de dados.

5.2 Pré-processamento dos dados de entrada

Os dados brutos extraídos da base continham vários problemas quanto a sua qualidade, como valores ausentes ou inconsistentes. Para resolver estas questões, fez-se necessário a aplicação de tarefas de pré-processamento. Além disso, nesta etapa o conjunto de dados de entrada foi preparado para ser compatível com as ferramentas WEKA e scikit-learn. Este processo também foi útil para aumentar a confiabilidade dos resultados dos algoritmos, visto que problemas na qualidade dos dados de entrada podem interferir nos resultados do processo de mineração.

Os atributos L1 à L24, que são dados numéricos contendo as últimas 24 coletas de leitura da unidade consumidora, medidos em KWh, continham problemas como, por exemplo, valores ausentes, além de alguns valores estarem negativos. Para estes atributos, as seguintes tarefas foram realizadas:

- Dados dos atributos das leituras do consumo de valor negativo, foram considerados inconsistências e portanto, foram tratados para não causar distorções nos resultados dos algoritmos. Estes dados foram removidos e em seu lugar foram imputados valores de acordo com a média aritmética dos dados de cada objeto. O cálculo foi feito para cada registro separadamente, considerando os valores anteriores ou posteriores ao dado inconsistente. Não foi utilizada uma média global para estes dados;

- Dados ausentes foram tradados da mesma forma que os dados inconsistentes. Porém, objetos que continham mais de 5 atributos ausentes, foram simplesmente descartados, tendo em vista que o fato de imputar valores com base na média também poderia causar alguma distorção nos resultados.

A imputação de valores e a remoção de dados ausentes foi executada com as ferramentas pandas e scikit-learn. Esta última possui uma vasta biblioteca para executar diferentes técnicas de pré-processamento. Já o pandas é um complemento, onde os dados são importados e podem ser processados de forma tabular, semelhante a planilhas eletrônicas, facilitando visualização e a seleção dos dados irregulares para posteriormente serem tratados pelo scikit-learn.

Os atributos SEXO e ESTADO_CIVIL, ambos atributos nominais, estavam aptos a serem utilizados na ferramenta WEKA. Porém para a aplicação dos algoritmos no scikit-learn, os dados de entrada precisam estar em um formato de matriz de dados do tipo numérico. Além disso, o atributo SEXO continha valores ausentes. Desta forma, os seguintes procedimentos foram executados:

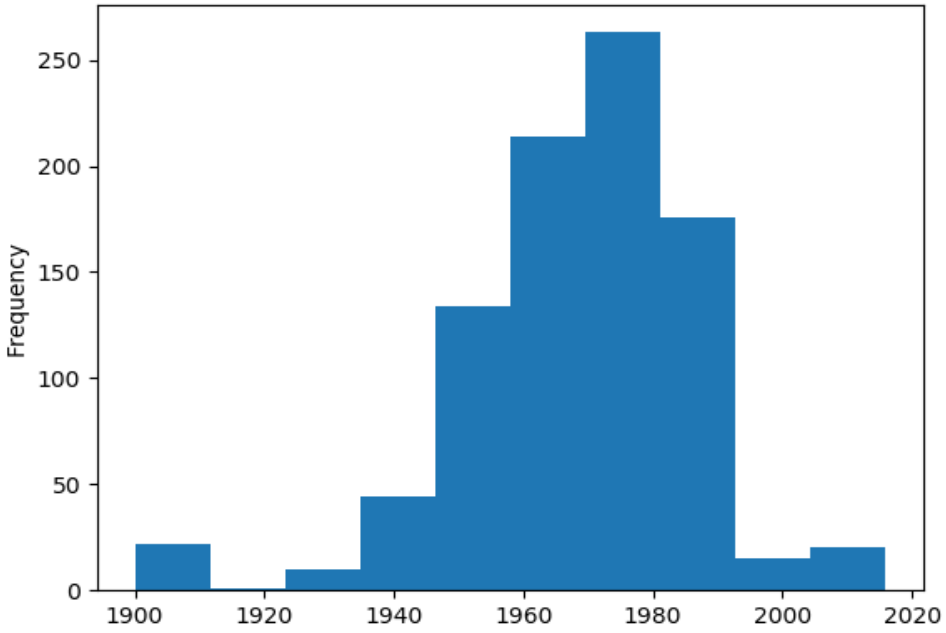
- Valores ausentes do atributo SEXO foram substituídos pelo valor estatístico da moda, ou seja, o valor “M”;

- Os valores dos atributos SEXO e ESTADO_CIVIL foram convertidos para valores numéricos para representá-los como, por exemplo, no atributo SEXO, o valor “M” foi convertido para 1 e o valor “F” para 0.

Dados que representam datas, como, por exemplo, os atributos DATA_NASCIMENTO e DATA_CADASTRO foram convertidos para números inteiros ordinais, levando-se em consideração apenas o ano de nascimento do consumidor, para que fossem compatíveis com o módulo scikit-learn, além do fato de que o WEKA também não suporta um atributo do tipo data como entrada em certos tipos de algoritmos.

Para a conversão destes atributos, foram utilizadas faixas de intervalo entre datas através da visualização de um histograma. Desta forma, cada intervalo de data recebeu um número discreto. A figura 12 mostra um exemplo de histograma utilizado para dividir as faixas de intervalo. Nota-se que existem valores que podem ser considerados uma anomalia, devido a data de nascimento ser muito inferior, como por exemplo os consumidores nascidos no ano de 1910. Estes foram substituídos pelo valor de maior frequência, ou seja, 1980.

Figura 12 – Histograma criado para identificar faixas de intervalo de datas no atributo DATA_NASCIMENTO



Fonte: do autor.

Para os dados referentes aos atributos IDREGIAO e IDCLASSE, não foi necessário efetuar nenhuma tarefa de pré-processamento. Estes dados não continham valores ausentes e já estavam originalmente organizados em dados de tipo numérico, permitindo assim a sua utilização nas ferramentas de mineração.

Já os atributos restantes, PREFERENCIA_DEBITO_CONTA, MULTA_ATRASO e PC foram representados por dados binários, onde 1 significa uma ocorrência positiva, como por exemplo, a indicação de que os registros representam uma perda comercial, ou a indicação positiva de que o consumidor adotou o pagamento de sua fatura de energia por meio de débito em conta. Já o valor 0 é um indicador negativo do atributo, como no caso da coluna MULTA_ATRASO, em que neste sentido o valor 0 indica que o consumidor não pagou sua fatura em atraso nos últimos 24 meses.

Com os ajustes necessários executados, os dados de entrada foram convertidos no formato ARFF para serem utilizados pelo WEKA e mantidos em CSV para alimentar os algoritmos no scikit-learn. A figura 13 mostra o conjunto de dados de entrada de treinamento no formato ARFF.

Figura 13 – Arquivo ARFF utilizado no WEKA gerado a partir de um dos conjuntos de dados de entrada

```
@relation train5

@attribute L1 numeric
@attribute L2 numeric
@attribute L3 numeric
@attribute L4 numeric
@attribute L5 numeric
@attribute L6 numeric
@attribute L7 numeric
@attribute L8 numeric
@attribute L9 numeric
@attribute L10 numeric
@attribute L11 numeric
@attribute L12 numeric
@attribute L13 numeric
@attribute L14 numeric
@attribute L15 numeric
@attribute L16 numeric
@attribute L17 numeric
@attribute L18 numeric
@attribute L19 numeric
@attribute L20 numeric
@attribute L21 numeric
@attribute L22 numeric
@attribute L23 numeric
@attribute L24 numeric
@attribute IDREGIAO {1,2,5,6,8,9,11,14,17,18,22,25,28,29,32,35,37,39,40,41,43,47,51,54,57,59,60,62,64,65,66}
@attribute DATA_CADASTRO {0,1,2}
@attribute ESTADO_CIVIL {C,D,S,O,V}
@attribute DATA_NASCIMENTO {0,1,2,3}
@attribute PC {0,1}

@data
454,343,427,438,316,306,313,320,257,211,164,343,434,561,381,356,321,143,163,72,55,60,161,116,29,1,C,3,0
327,370,302,310,280,322,255,300,282,236,245,274,416,367,329,314,284,317,298,305,326,335,314,312,5,1,C,2,0
212,226,198,230,155,108,99,114,120,108,123,122,169,187,210,142,154,105,105,97,120,115,110,144,22,2,C,1,0
146,134,82,70,70,103,102,75,147,98,125,157,165,163,140,177,150,34,32,139,152,147,144,153,60,1,S,3,0
232,199,210,180,139,151,121,150,133,126,151,161,226,235,219,176,148,125,124,125,147,133,173,166,32,1,C,1,0
```

Fonte: o autor.

Depois dos procedimentos realizados nesta seção, os algoritmos de mineração de dados foram executados utilizando as ferramentas WEKA e scikit-learn, cujo o comparativo entre elas e os seus respectivos resultados serão apresentados na seção seguinte, os quais serão discutidos e explicados na seção 5.4.

5.3 Comparativo entre as ferramentas WEKA e scikit-learn

Os primeiros experimentos foram executados utilizando o pacote WEKA, permitindo a execução de diversos algoritmos de mineração de dados. Através do WEKA, foi possível executar diversos testes para validar os conjuntos de dados, além dos algoritmos e técnicas de mineração. Neste sentido, a ferramenta foi utilizada como um instrumento de análise exploratória, na qual foram utilizados todos os subconjuntos de características dos dados de entrada, descritos na seção 5.1.

Os algoritmos utilizados pertencem à tarefa de classificação e foram empregados utilizando uma abordagem de aprendizagem supervisionada. A seleção de cada algoritmo ocorreu da seguinte forma: a sua utilização em alguns dos trabalhos estudados no Capítulo 3, a utilização de classificadores baseados em métodos diferentes, como os baseados em função e proximidade, além da própria observação dos resultados de diferentes técnicas aplicadas.

Os conjuntos de dados de entrada foram divididos em duas partes: conjunto de treinamento e teste. Deste modo, os algoritmos foram ajustados a um conjunto de dados de treinamento, gerando um modelo preditivo que posteriormente foi avaliado sobre um conjunto de teste, contendo dados que não foram vistos anteriormente pelo modelo, observando as métricas de desempenho para cada algoritmo.

A avaliação do modelo sobre o conjunto de testes é uma abordagem necessária, visto que o modelo tende a ter um resultado otimista quando avaliado sobre o conjunto de treinamento, resultando em uma avaliação equivocada do modelo preditivo, devido ao fato da sua extrema especialização (HAN; KAMBER e PEI, 2011).

A tabela 8 relaciona os conjuntos de dados, algoritmos e suas métricas de desempenho, avaliadas sobre os dados de teste de cada conjunto, utilizando a ferramenta WEKA.

Tabela 8 – Relação entre os resultados dos algoritmos executados no WEKA e cada conjunto de dados

ALGORITMO	CONJUNTO DE DADOS	PRECISÃO	REVOCAÇÃO	MEDIDA F	TVP PC = 1	TFP PC = 1
Árvore de Decisão	Conjunto 1	0,748	0,777	0,758	0,269	0,096
	Conjunto 2	?	0,800	?	0,000	0,000
	Conjunto 3	0,754	0,785	0,764	0,269	0,087
	Conjunto 4	0,733	0,785	0,744	0,154	0,058
Random Forests	Conjunto 1	0,855	0,823	0,762	0,115	0,000
	Conjunto 2	0,686	0,762	0,713	0,077	0,067
	Conjunto 3	0,845	0,808	0,729	0,038	0,000
	Conjunto 4	0,813	0,831	0,799	0,269	0,029
SGD (SVM)	Conjunto 1	0,744	0,800	0,725	0,038	0,010
	Conjunto 2	0,845	0,808	0,729	0,038	0,000
	Conjunto 3	0,744	0,800	0,725	0,038	0,010
	Conjunto 4	0,845	0,808	0,729	0,038	0,000
Simple Logistic	Conjunto 1	0,845	0,808	0,729	0,038	0,000
	Conjunto 2	0,845	0,808	0,729	0,038	0,000
	Conjunto 3	0,845	0,808	0,729	0,038	0,000
	Conjunto 4	?	0,800	?	0,000	0,000
Logistic Regression	Conjunto 1	0,710	0,746	0,725	0,192	0,115
	Conjunto 2	0,709	0,792	0,720	0,038	0,019
	Conjunto 3	0,706	0,738	0,720	0,192	0,125
	Conjunto 4	0,733	0,785	0,744	0,154	0,058
k-NN	Conjunto 1	0,691	0,731	0,708	0,154	0,125
	Conjunto 2	0,672	0,769	0,707	0,038	0,048
	Conjunto 3	0,706	0,738	0,720	0,192	0,125
	Conjunto 4	0,781	0,808	0,787	0,308	0,067

Fonte: do autor.

Os conjuntos de dados foram testados seguindo a mesma abordagem no módulo scikit-learn, porém utilizando apenas os algoritmos com melhores resultados que foram encontrados nos primeiros experimentos com o WEKA. A tabela 9 relaciona os conjuntos de dados, medidas de desempenho e algoritmos com os seus respectivos resultados utilizando o scikit-learn.

Tabela 9 – Relação entre os resultados dos algoritmos executados no scikit-learn e cada conjunto de dados

ALGORITMO	CONJUNTO DE DADOS	PRECISÃO	REVOCAÇÃO	MEDIDA F	TVP PC = 1	TFP PC = 1
Árvore de Decisão	Conjunto 1	0,750	0,760	0,760	0,346	0,135
	Conjunto 2	0,660	0,720	0,690	0,077	0,115
	Conjunto 3	0,730	0,770	0,750	0,231	0,096
	Conjunto 4	0,720	0,750	0,730	0,231	0,125
Random Forests	Conjunto 1	0,770	0,810	0,780	0,231	0,048
	Conjunto 2	0,750	0,800	0,750	0,115	0,029
	Conjunto 3	0,750	0,790	0,760	0,192	0,058
	Conjunto 4	0,800	0,820	0,790	0,231	0,029
k-NN	Conjunto 1	0,780	0,810	0,780	0,269	0,058
	Conjunto 2	0,690	0,690	0,690	0,231	0,192
	Conjunto 3	0,780	0,810	0,780	0,269	0,058
	Conjunto 4	0,790	0,820	0,790	0,269	0,048

Fonte: do autor.

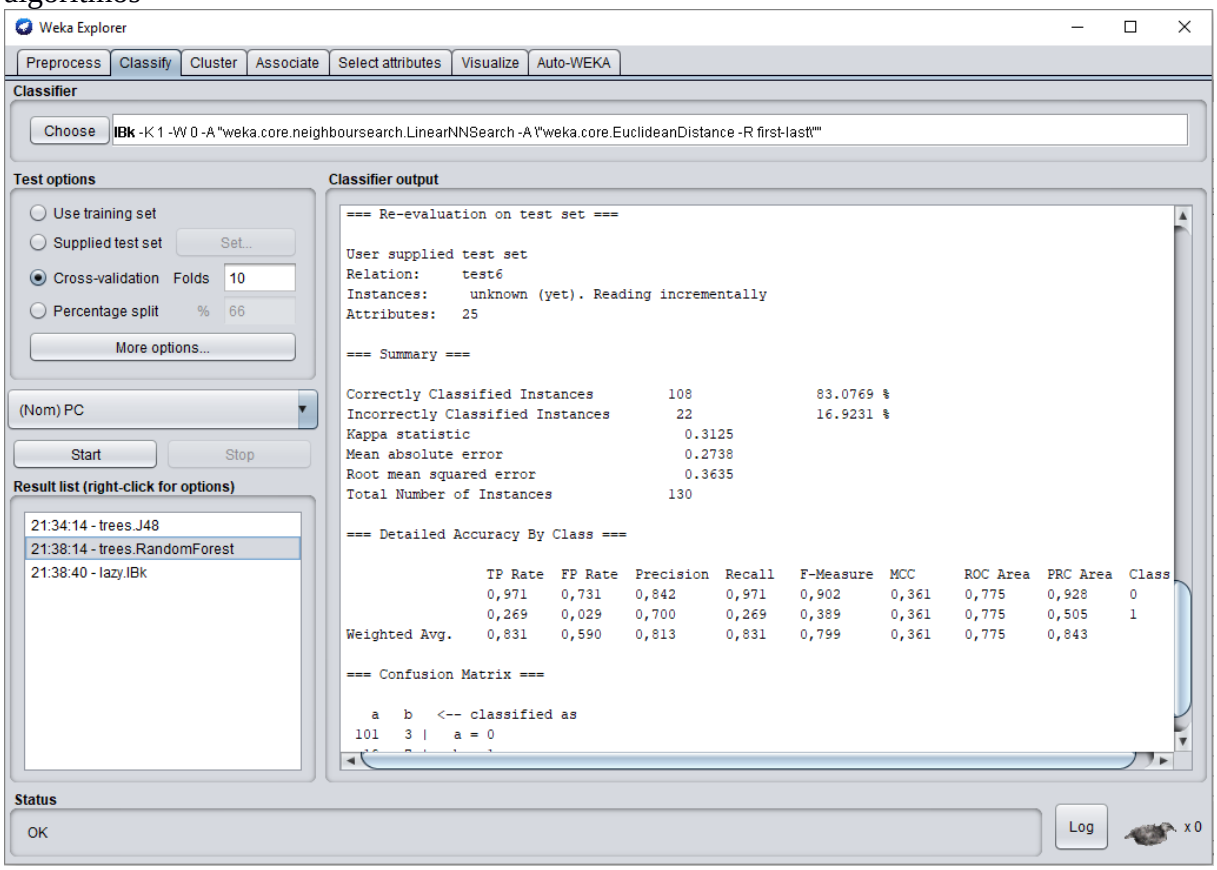
Algumas técnicas não possuem os mesmos parâmetros de ajuste, ou não suportam alguns mecanismos, como no caso das árvores de decisão, onde a característica de poda não está disponível no scikit-learn, além do fato de sua implementação ser com base no algoritmo CART, diferentemente do WEKA que utiliza C4.5 (SCIKIT-LEARN, 2018). Por estes motivos, algumas técnicas obtiveram resultados um pouco diferentes nas duas ferramentas de mineração de dados.

Porém, na maioria dos casos, os resultados mantiveram-se próximos, trazendo deste modo, mais confiança na avaliação da saída dos algoritmos, apesar de haver algumas diferenças em alguns parâmetros e na implementação, como já foi discutido. Sobretudo, ambos possuem vantagens e desvantagens.

A principal vantagem do WEKA é a de que sua interface gráfica e intuitiva facilita o processo de mineração de dados, podendo ser utilizada para uma análise exploratória, quando deseja-se ter rapidamente resultados e testar modelos diferentes, no sentido de verificar se o conjunto de dados ajusta-se bem ao modelo. Outra vantagem é a de que o WEKA pode

executar automaticamente algumas tarefas de pré-processamento dos dados, agilizando ainda mais o processo para o usuário que deseja explorar diferentes algoritmos e conjuntos de dados. A figura 14 exibe uma das interfaces gráficas do usuário durante a execução de um algoritmo.

Figura 14 – Interface Gráfica Explorer do ambiente WEKA com resultados de execução de algoritmos



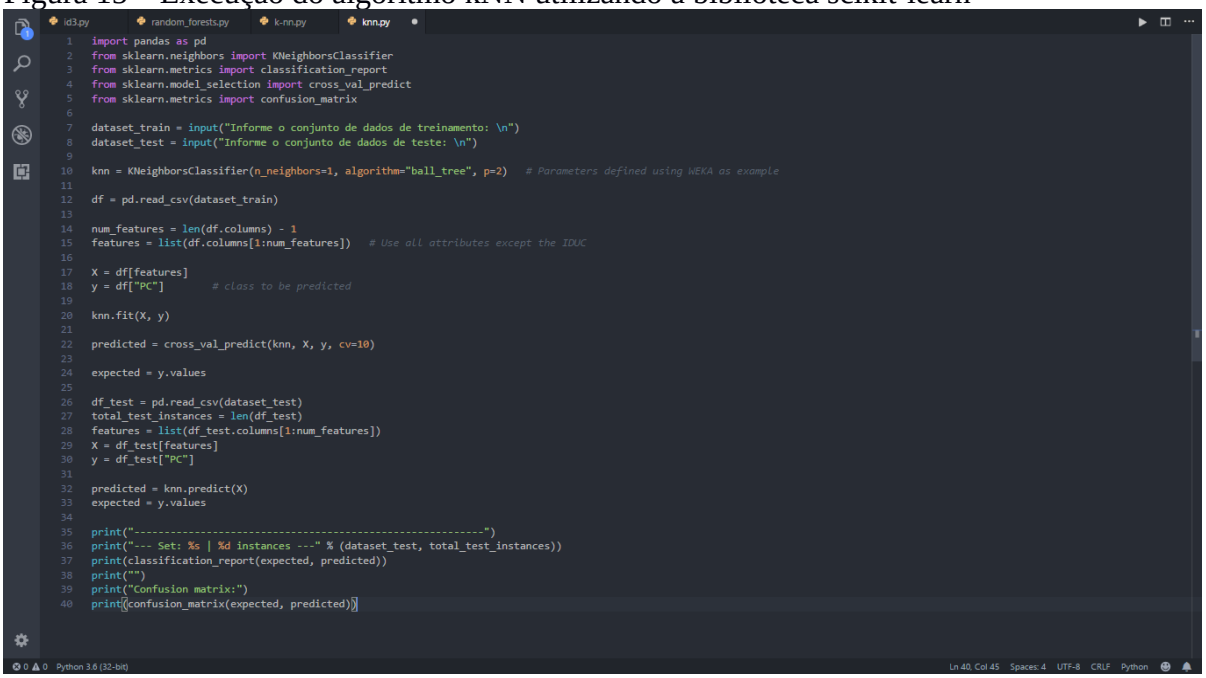
Fonte: do autor.

Uma das desvantagens do ambiente WEKA é o fato de que o conjunto de dados de entrada deve ser convertido para um formato específico, no caso do arquivo ARFF, sendo que a conversão para este padrão causa alguns erros no arquivo, fazendo com que o usuário tenha que editá-lo manualmente. Outra desvantagem que possui relevância é a visualização e plotagem de gráficos, entre outras tarefas descritivas dos dados, que são um tanto quanto pobres graficamente.

A biblioteca de mineração de dados scikit-learn possui como principal vantagem a fácil integração com diversos módulos e bibliotecas disponíveis em Python, sendo esta uma linguagem de programação de alto nível com constante aperfeiçoamento em computação científica, segundo Pedregosa et al. (2011), possuindo diversas bibliotecas que auxiliam na experiência com mineração de dados, como por exemplo, o módulo de plotagem de gráficos matplotlib e a ferramente utilizada para manipulação de dados pandas que também foram utilizadas nos experimentos deste trabalho. Além disso, os dados de entrada não necessitam ser convertidos para um formato específico, podendo este ser CSV, planilhas eletrônicas, bases relacionais, JSON, etc. Outro ponto que merece destaque é a vasta documentação da ferramenta.

Uma das desvantagens do scikit-learn é a questão de que é necessário um conhecimento prévio na área de aprendizado de máquina, não sendo tão intuitivo quanto o ambiente WEKA. Além disso, a biblioteca scikit-learn não disponibiliza uma interface gráfica nativa, mas sim, expõe sua API, necessitando de conhecimentos em programação, especificamente da linguagem Python. A figura 15 demonstra a utilização da ferramenta na aplicação de algoritmos de mineração de dados.

Figura 15 – Execução do algoritmo kNN utilizando a biblioteca scikit-learn



```
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import confusion_matrix

dataset_train = input("Informe o conjunto de dados de treinamento: \n")
dataset_test = input("Informe o conjunto de dados de teste: \n")

knn = KNeighborsClassifier(n_neighbors=1, algorithm="ball_tree", p=2)   # Parameters defined using WEKA as example

df = pd.read_csv(dataset_train)

num_features = len(df.columns) - 1
features = list(df.columns[1:num_features])   # Use all attributes except the IDUC

X = df[features]
y = df["PC"]        # class to be predicted

knn.fit(X, y)

predicted = cross_val_predict(knn, X, y, cv=10)

expected = y.values

df_test = pd.read_csv(dataset_test)
total_test_instances = len(df_test)
features = list(df_test.columns[1:num_features])
X = df_test[features]
y = df_test["PC"]

predicted = knn.predict(X)
expected = y.values

print("-------------------------------------------------------")
print("--- Set: %s | %d instances ---" % (dataset_test, total_test_instances))
print(classification_report(expected, predicted))
print("")
print("Confusion matrix:")
print(confusion_matrix(expected, predicted))
```

Fonte: do autor.

As ferramentas utilizadas para o desenvolvimento do trabalho mostraram-se satisfatórias, sendo o WEKA utilizado para executar uma análise exploratória dos algoritmos e validar os conjuntos de dados de entrada, enquanto que o módulo scikit-learn foi utilizado para executar e validar os algoritmos de classificação que tiveram os melhores resultados no WEKA, sendo possível por meio destas alcançar o objetivo do trabalho.

Na seção seguinte, os resultados aqui demonstrados serão discutidos e interpretados, utilizando como base a ferramenta scikit-learn, descrevendo como foram definidas as medidas de desempenho e de que forma elas podem ser interpretadas para avaliar os resultados, além dos procedimentos realizados para alcançá-los.

5.4 Discussão dos resultados obtidos e procedimentos realizados

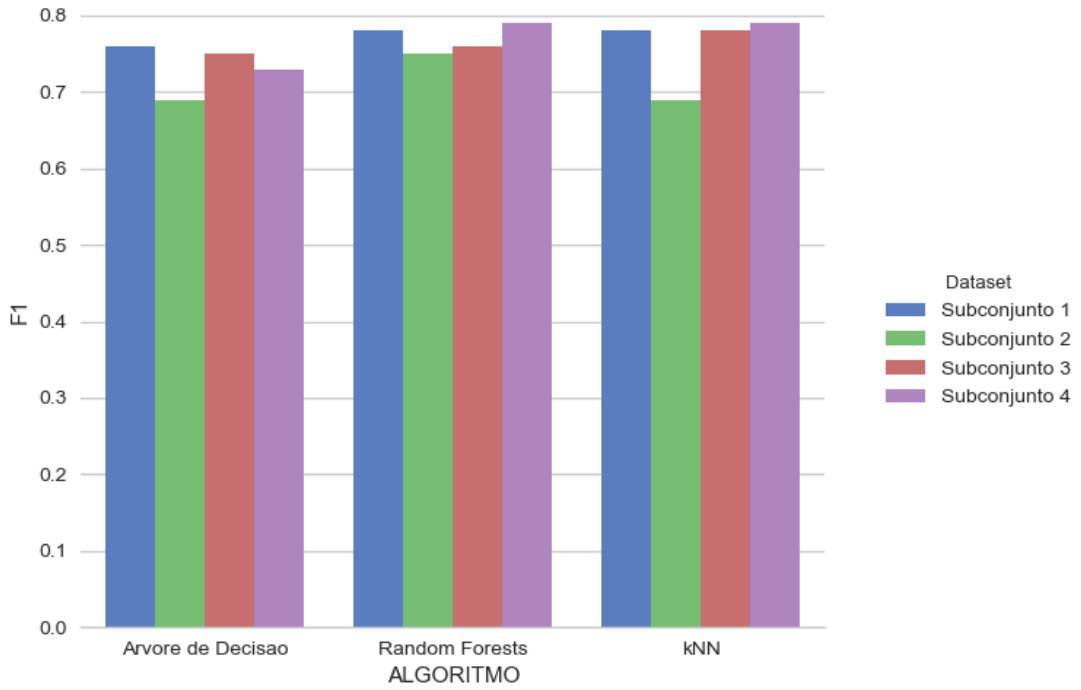
A descoberta de perdas comerciais foi ajustada a abordagem de classificação, dentro do tema de mineração de dados. Desta forma, os algoritmos de classificação selecionados tinham a tarefa de predizer se um determinado conjunto de dados de entrada deve ser classificado como perda comercial ou não, de acordo com os atributos destes dados. Neste sentido, o problema pode ser considerado um problema binário de classificação, onde o modelo classifica como 0 um objeto que não representa uma perda comercial, ou seja, um caso negativo, e 1 para um caso positivo de perda comercial. Dito isto, para avaliar a capacidade preditiva do classificador, foram utilizadas algumas medidas de avaliação, sendo então possível verificar a acuracidade do algoritmo utilizado, ou seja, a capacidade do algoritmo classificar um objeto corretamente, sendo este um caso positivo ou não.

Para medir a acuracidade dos classificadores, foram utilizadas três medidas principais (seção 2.6.2): precisão, revocação e medida F. Estas medidas foram selecionadas pois consideram os falsos positivos e falsos negativos, no caso da precisão e revocação respectivamente, enquanto que a medida F representa uma média ponderada das mesmas.

Entretanto, para verificar a capacidade de acerto de verdadeiros positivos do modelo, ou seja, o caso em que uma ocorrência de perda comercial é positiva, foi verificado a taxa de verdadeiros positivos (TVP) e a taxa de falsos positivos (TFP), para o caso em que a classe de saída era igual a 1 (PC = 1). Estas taxas foram incluídas devido ao fato de que a medida F exibe somente a média ponderada da precisão e da revocação, sendo útil como uma avaliação geral do modelo, porém não deixa explícito se o algoritmo está classificando corretamente uma perda comercial.

A figura 16 apresenta a avaliação da acuracidade dos modelos, utilizando como parâmetro a medida F, considerando os resultados obtidos através da ferramenta scikit-learn.

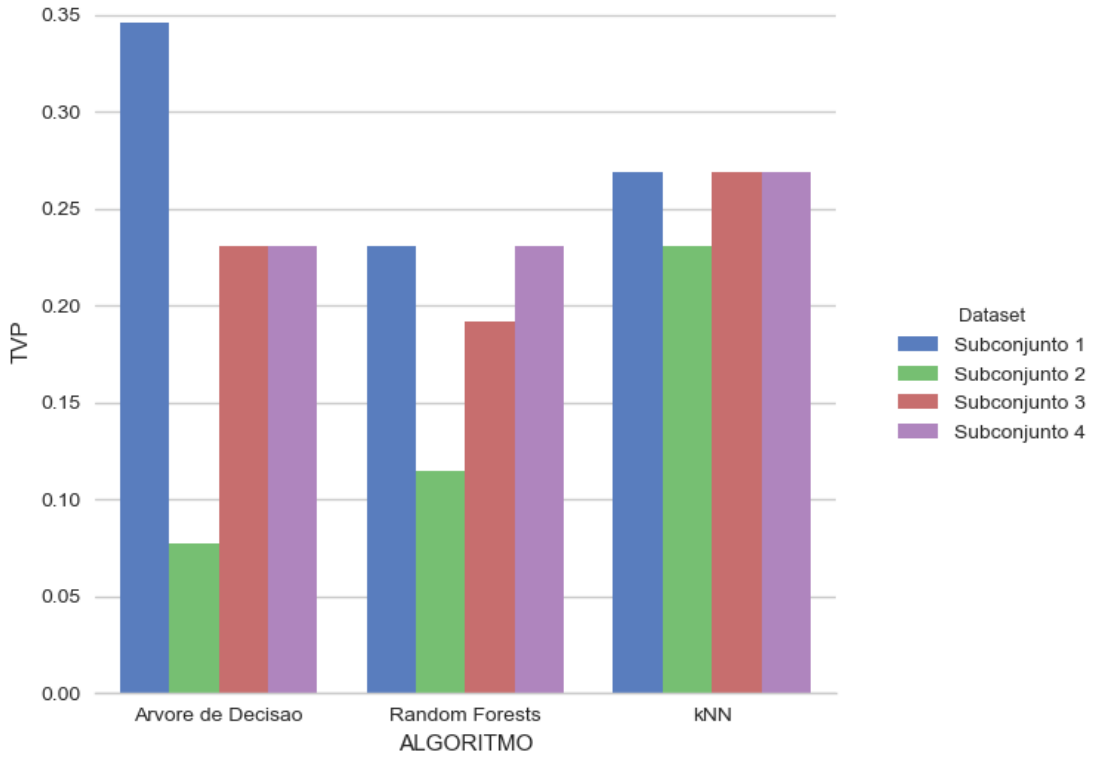
Figura 16 – Gráfico representando a medida F dos algoritmos aplicados



Fonte: do autor.

É possível confirmar que os algoritmos que tiveram os melhores resultados são o r*andom forests* e o *kNN*, atingindo uma pontuação de 0,8 na medida F, considerado um bom desempenho, visto que 1,0 seria a melhor pontuação possível. Isto refere-se ao quarto subconjunto de dados de entrada, que contém apenas os atributos numéricos, ou seja, os dados de leitura em KWh dos consumidores. Já a figura 17 demonstra os resultados obtidos considerando apenas a taxa de verdadeiros positivos para os registros classificados como perda comercial.

Figura 17 – Gráfico representando a taxa de verdadeiros positivos dos algoritmos aplicados



Fonte: do autor.

Desta forma, fica evidenciado que o algoritmo que conseguiu uma melhor taxa de acerto dos objetos que são uma ocorrência de perda comercial foi o de árvore de decisão, utilizando o primeiro subconjunto de dados, atingindo aproximadamente 35% de objetos classificados como verdadeiros positivos. Neste sentido, fica claro que a medida F apenas não é suficiente para avaliar se os modelos estão detectando corretamente as perdas comerciais. Além destas métricas, a acuracidade também foi verificada através das matrizes de confusão (seção 2.6.2), conforme demonstram as tabelas 10, 11 e 12, onde o valor 1 corresponde a uma perda comercial e o valor 0 não considera uma perda.

Tabela 10 – Matriz de confusão para algoritmo Árvore de Decisão utilizando o subconjunto 1

	Classe predita	
Classe atual	**0**	**1**
0	90	14
1	17	9

Fonte: do autor.

Tabela 11 – Matriz de confusão para algoritmo r*andom forests* utilizando o subconjunto 4

	Classe predita	
Classe atual	**0**	**1**
0	101	3
1	20	6

Fonte: do autor.

Tabela 12 – Matriz de confusão para algoritmo *kNN* utilizando o subconjunto 4

	Classe predita	
Classe atual	**0**	**1**
0	99	5
1	19	7

Fonte: do autor.

Através da matriz de confusão foi possível verificar que os modelos preditivos têm uma boa acuracidade na predição da classe 0, ou seja, exemplos que não tem ocorrência de perda comercial. Por outro lado, o modelo possui uma habilidade inferior para rotular a classe 1, que representa a ocorrência de perda comercial. Porém esta classe possui um número bem menor de exemplos, sendo que a amostra do conjunto de dados reflete a base de dados de origem, que contém um número bem pequeno de exemplos em que a perda comercial é positiva. Isto pode ser considerado um problema de classe não balanceada, onde a classe de principal interesse é representada por poucos objetos da base, enquanto que a classe negativa é representada pela maioria dos objetos (HAN; KAMBER e PEI, 2011). Contudo, deve ser observado que a classificação correta dos casos negativos também é importante, tendo em vista que os casos positivos requerem uma inspeção *in loco* para deflagar a perda comercial no caso de fraude, causando assim um gasto desnecessário nas inspeções de falsos positivos.

Deste modo, é possível afirmar que o algoritmo que melhor classificou os casos negativos foi o r*andom forests*, enquanto que a técnica de árvore de decisão conseguiu melhores resultados para reconhecer casos positivos de perdas comerciais, sendo esta última um bom método para extrair regras para reconhecer quando uma perda ocorre, já que a árvore gerada não é difícil de ser interpretada. Ainda assim, outra técnica que teve bons resultados foi o kNN que, apesar de ter uma boa taxa de acuracidade, não obteve resultados superiores a árvore de decisão que teve uma taxa superior de verdadeiros positivos. Na tabela 13 está a relação de cada algoritmo com a sua acuracidade.

Tabela 13 – Taxa de acuracidade dos algoritmos

Algoritmos	**Taxa de Verdadeiros Positivos**	**Taxa de Verdadeiros Negativos**	**Acuracidade**
Random Forests	23%	97%	82%
Árvore de Decisão	35%	86%	76%
kNN	27%	95%	81%

Fonte: do autor.

Para validar os resultados, as predições realizadas pelos algoritmos foram exibidas para um especialista na área de identificação de perdas comerciais. Foi verificado que houve alguns casos de falso positivo onde é possível identificar uma diferença no consumo, o que pode ter afetado a decisão do algoritmo, porém não são fraudes ou problemas no medidor e sim, uma oscilação bem significativa devido ao consumo irregular, podendo citar como exemplo um caso de consumidor do poder público, que não possui um consumo habitual.

Também houve um caso de falso positivo em que o medidor estava com defeito, gerando vários meses de consumo mínimo, ou seja, a demanda que é cobrada pela distribuidora mesmo sem uma utilização significativa de energia. Após a substituição do medidor, o consumo subiu, gerando uma variação que pode ter influenciado o modelo preditivo. Uma sugestão indicada pelo especialista seria a de atribuir pesos para determinados atributos, sendo que a região e a classe podem determinar com mais segurança a ocorrência ou não de uma perda comercial, em caso de esta não ser detectada pelo apenas pelo consumo. Isto pode ser comprovado pelo fato de que uma determinada classe de consumo, como por exemplo o poder público, não teve nenhum caso de fraude até o momento. Porém o modelo

preditivo criado utilizando árvore de decisão não considerou este atributo, mas poderia utilizá-lo como sendo o primeiro nodo da árvore e a partir deste, gerar os demais ramos. Já na questão da região, esta poderia receber um peso maior de acordo com o local onde há incidência maior de fraudes de energia, por exemplo.

É necessário salientar que do ponto de vista da distribuidora de energia, é mais admissível um caso de falso positivo do que um falso negativo, pelo fato de que os dados de falsos positivos ainda podem ser posteriormente analisados por especialistas antes de enviar uma equipe de inspeção até o local, ao contrário de um falso negativo que poderia passar desapercebido e ser simplesmente ignorado. Por outro lado, o consumidor poderia ser encorajado a cometer uma fraude caso a inspeção de um falso positivo fosse realizada, pois do ponto de vista do usuário da energia, este acreditaria não ter uma nova visita de inspeção. Além disso existem casos em que só os dados não identificam uma perda comercial e a única maneira de detectá-la é através de uma inspeção na UC.

6 CONSIDERAÇÕES FINAIS

Com base no estudo realizado e nos resultados obtidos, ficou comprovado que a abordagem de mineração de dados pode ser aplicada na área de detecção de perdas comerciais utilizando os dados obtidos da empresa de distribuição de energia, servindo como um processo de apoio a decisão, onde o usuário final poderá utilizar o conhecimento obtido para direcionar corretamente os recursos de inspeção na unidade consumidora, estando de acordo com a proposta deste trabalho.

Diferentes técnicas foram aplicadas variando o conjunto de características dos dados de entrada. Neste sentido, o algoritmo *random forests* foi o que obteve os melhores resultados para classificar corretamente casos negativos de perdas comerciais, com uma acuracidade de 82% e TVN de 97% utilizando o conjunto de dados contendo valores numéricos. Já o método de classificação por árvore de decisão obteve uma maior taxa de acerto para casos positivos de perda, com uma acuracidade de 76% e TVP de aproximadamente 35%.

As técnicas utilizadas não reconheceram corretamente todos os casos de perda comercial, o que pode ser devido ao fato de existirem poucos registros de perdas na base para treinar o classificador ou ao conjunto de dados que não se ajustou bem ao modelo, porém, a abordagem de classificação demonstrou ser uma técnica útil para ser aplicada, cruzando informações do consumidor e também do seu hábito de consumo para encontrar uma possível ocorrência de perda comercial, como fraude, desvio, ou problemas no medidor.

As ferramentas utilizadas para executar as tarefas que compõem a mineração de dados, o WEKA e a biblioteca scikit-learn, mostraram-se satisfatórias, com destaque a esta última, que contém uma vasta coleção de algoritmos, recursos estatísticos e ferramentas de pré-processamento dos dados, integrando-se facilmente com diversas bibliotecas da comunidade científica e de análise de dados, podendo ser utilizada para aplicar o projeto em produção e automatizar o processo de mineração.

Quanto a pesquisa dos trabalhos relacionados ao tema, constatou-se que diferentes métodos podem ser utilizados para alcançar os objetivos, servindo de contribuição para a desenvolvimento deste trabalho, que tratou de demonstrar e aprofundar quais as ferramentas e algoritmos utilizados, obtendo-se mais uma opção para identificar perdas comerciais.

Para utilizar este projeto em ambiente de produção, são propostas algumas ideias no sentido de melhorar os resultados obtidos, com o objetivo de diminuir os casos de falsos positivos de falsos negativos, para melhorar a acuracidade do modelo, na seção 6.1.

6.1 Trabalhos futuros

Com o objetivo de obter melhores resultados na predição de perdas comerciais, é proposto algumas sugestões como melhoria e também como projetos futuros:

- Treinar os modelos preditivos utilizando mais exemplos de casos positivos de perda comercial, com base em novos exemplos obtidos ou dados simulados;

- Utilizar outros atributos no conjunto de dados de entrada, como, por exemplo, um atributo que indicará a variação mensal de consumo;

- Identificar e remover atributos desnecessários ou que não influenciam no resultado final;

- Atribuir pesos aos atributos que caracterizam a região e a classe de consumo com maior chance de ocorrer uma perda comercial;

- Desenvolver uma aplicação que implementará todo o processo de mineração de dados descrito neste trabalho, exibindo os resultados das unidades consumidoras que podem estar gerando uma possível perda comercial, para que esta possa ser utilizada pelo setor que realiza as inspeções.

Os testes realizados com os algoritmos de classificação foram executados utilizando dados extraídos da base da empresa de distribuição de energia em formato de arquivos CSV. Estes testes foram executados offline e depois validados com um especialista. Outra proposta para este projeto é a de utilizar um servidor com uma aplicação web para o usuário final, de modo que este possa fornecer dados novos para os modelos preditivos, fornecendo uma interface de alto nível para o usuário, de modo que ele possa tomar as decisões com base nos resultados fornecidos pela aplicação, sem que seja necessário possuir um conhecimento na área de mineração de dados. Além disso, a aplicação poderá exibir outras informações com dados em tempo real, como relatórios, valores recuperados, gráficos, dados de consumo, entre outros.

REFERÊNCIAS

ABRADEE, Associação Brasileira de Distribuidores de Energia Elétrica. **Furto e Fraude de Energia**. Disponível em: <http://www.abradee.com.br/setor-de-distribuicao/perdas/furto-e-fraude-de-energia>. Acesso em: 12 de outubro de 2017.

ANEEL, Agência Nacional de Energia Elétrica. **Perdas de Energia**. Disponível em: <http://www.aneel.gov.br/metodologia-distribuicao/-/asset_publisher/e2INtBH4EC4e/content/perdas/654800?inheritRedirect=false>. Acesso em: 12 de outubro de 2017.

ANEEL, Agência Nacional de Energia Elétrica. **Procedimentos de Distribuição de Energia Elétrica no Sistema Elétrico Nacional**. Disponível em: <http://www.aneel.gov.br/prodist>. Acesso em 23 de outubro de 2017.

ANEEL, Agência Nacional de Energia Elétrica. **Procedimentos de Distribuição de Energia Elétrica no Sistema Elétrico Brasileiro. Módulo 5**. Disponível em: <http://www2.aneel.gov.br/arquivos/pdf/modulo5_f.pdf>. Acesso em 19 de outubro de 2017.

BEAZLEY, David M. **Python Essential Reference**. 4ª edição. Editora Addison-Wesley Professional, 2009.

BREIMAN, Leo. **Random Forests**. Machine Learning. Berkley, n. 45, p. 5-32, 2001. Disponível em <https://link.springer.com/content/pdf/10.1023/A:1010933404324.pdf>. Acesso em 18 de Maio de 2018.

CARVALHO, André Carlos Ponce de Leon de; FACELI, Katti; LORENA, Ana Carolina; GAMA, João. **Inteligência Artificial:** uma abordagem de aprendizado de máquina. 1ª Edição. Rio de Janeiro: Editora LTC, 2011.

CASTRO de, Leandro Nunes; FERRARI, Daniel Gomes. **Introdução à Mineração de Dados: c**onceitos básicos, algoritmos e aplicações. 1ª Edição. São Paulo: Saraiva, 2016.

FERREIRA, Hamilton Melo. **Uso de Ferramentas de Aprendizado de Máquina para Prospecção de Perdas Comerciais em Distribuição de Energia Elétrica**. Universidade Estadual de Campinas. 2008.
HAN, Jiawei; KAMBER, Micheline; PEI, Jian. **Data mining:** concepts and techniques. 3ª edição. Whaltman: Editora Morgan Kaufmann, 2011.

GEDRA, Ricardo Luis; BORELLI, Reinaldo; BARROS, Benjamim Ferreira. **Geração, Transmissão, Distribuição e Consumo de Energia Elétrica**. 1ª Edição. São Paulo: Editora Érica, 2014.

Instituto Acende Brasil. **Perdas Comerciais e Inadimplência no Setor Elétrico**. Disponível em: <http://www.acendebrasil.com.br/media/estudos/2017_WhitePaperAcendeBrasil_18_PerdasInadimplencias.pdf >. Acesso em 13 de Agosto de 2017.

LUGER, Jorge F. **Inteligência artificial**. 6ª Edição. São Paulo: Editora Pearson, 2013.

MCKINNEY, Wes. **Python for Data Analysis**. 1ª edição. Sebastopol: O'Reilly Media, 2012.

PEDREGOSA, Fabian; VAROQUAUX, Gaël; GRAMFORT, Alexandre; MICHEL, Vincent; THIRION, Bertrand; GRISEL, Olivier; BLONDEL, Mathieu; PRETTENHOFER, Peter; WEISS, Ron; DUBOURG, Vincent; VANDERPLAS, Jake; PASSOS, Alexandre; COURNAPEAU, David; BRUCHER, Matthieu; PERROT, Matthieu; DUCHESNAY, Édouard. **Scikit-learn:** machine learning in Python. Journal of Machine Learning Research, n. 12, p. 2825-2830, 2011. Disponível em <http://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf>. Acesso em 5 de Novembro de 2017.

RUSSEL, Stuart; NORVIG, Peter. **Inteligência Artificial**. 3ª Edição. Rio de Janeiro: Elsevier Editora, 2013.

SCIKIT-LEARN. **Scikit-learn.** Disponível em: <http://scikit learn.org/stable/modules/tree.html>. Acesso em 2 de Maio de 2018.

SILVA, Vinícius Dornela; SCARPEL, Rodrigo Arnaldo. **Detecção de Fraudes na Distribuição de Energia Elétrica Utilizando Support Vector Machine**. Instituto Tecnológico de Aeronáutica. 2007.

TAN, Pang – Ning; STEINBACH, Michael; KUMAR, Vipin. **Introdução ao Datamining:** mineração de dados. 1ª Edição. Rio de Janeiro: Editora Ciência Moderna, 2009.

TELLES, Matt. **Python Power!:** the comprehensive guide. 1ª Edição. Boston: Editora Cengage Learning PTR, 2008.

TODESCO, J. L.; MORALES, A. B. T.; RAUTENBERG, S.; GARBELOTTO, L. A.; ATHAYDE, E. D. **Aplicação de Técnicas de Mineração de Dados para detecção de Fraudes de Energia**. Universidade Federal de Santa Catarina. 2007.

WITTEN, Ian H.; FRANK, Eibe. **Data Minining:** practical machine learning tools and techniques. 2ª Edição. San Francisco: Editora Morgan Kaufmann, 2005.

WITTEN, Ian H.; FRANK, Eibe; HALL, Mark A. **The WEKA Workbench**. Disponível em: <https://www.cs.waikato.ac.nz/ml/weka/Witten_et_al_2016_appendix.pdf>. Acesso em 11 de Outubro de 2017.

APÊNDICES

APÊNDICE A – Árvore de decisão gerada através da biblioteca scikit-learn