CEFET/RJ
DIRETORIA DE PESQUISA E PÓS-GRADUAÇÃO

APLICAÇÃO DE WORKFLOWS CIENTÍFICOS PARA IMPUTAÇÃO DE DADOS AUSENTES EM BIG DATA

Rodrigo Tavares de Souza

Exame de Qualificação para Dissertação de Mestrado do Programa de Pós-Graduação em Ciência da Computação do Centro Federal de Educação Tecnológica Celso Suckow da Fonseca, CEFET/RJ.

Orientador:
Jorge de Abreu Soares

Rio de Janeiro,

Novembro 2017

Aplicação de Workflows Científicos para Imputação de Dados Ausentes em Big Data

Exame de Qualificação para Dissertação de Mestrado do Programa de Pós-Graduação em Ciência da Computação do Centro Federal de Educação Tecnológica Celso Suckow da Fonseca, CEFET/RJ.

Rodrigo Tavares de Souza

Aprovada por:

Prof. Jorge de Abreu Soares, D.Sc. (orientador)

Prof. Eduardo Soares Ogasawara, D.Sc.

Prof. Ronaldo Ribeiro Goldschmidt, D.Sc.

Rio de Janeiro,

Novembro 2017

RESUMO

Aplicação de Workflows Científicos para Imputação de Dados Ausentes em Big Data

Rodrigo Tavares de Souza

Orientador:
Jorge de Abreu Soares

Resumo do Exame de Qualificação submetido ao Programa de Pós-Graduação em Ciência da Computação do Centro Federal de Educação Tecnológica Celso Suckow da Fonseca CEFET/RJ como parte dos requisitos necessários à obtenção do título de mestre.

É cada vez maior a quantidade de dados gerados por sistemas informatizados. O termo *Big Data* nunca foi tão relevante. O campo de pesquisa científica tem como um de seus pilares, o processamento de modelos matemáticos complexos sobre enormes massas de dados. Sistemas apoiados em técnicas de inteligência artificial ganham cada vez mais campo e são capazes de inferir conhecimento em uma escala jamais vista antes. O aumento no volume de dados por sua vez maximiza o problema de inconsistências de dados, em especial a ausência de dados, que pode comprometer seriamente a qualidade das análises produzidas.

Uma alternativa para a complementação de dados ausentes é a imputação, tema central deste trabalho. Essa dissertação apresenta um sistema de *workflow* científico para experimentação de técnicas de imputação em ambientes *Big Data*. O sistema propõe as técnicas de imputação por média, pelo algoritmo K-NN e por redes neurais *Back Propagation*, além da imputação composta, que consiste na composição de técnicas de agrupamento e seleção com os algoritmos K-Means e Principal Components Analysis (PCA), para a criação de planos de imputação. O Sistema apresenta ainda uma solução para a modelagem gráfica de planos de imputação.

Palavras-chave:
Imputação; Workflow Científico; *Big Data*.

Rio de Janeiro,

Novembro 2017

ABSTRACT

Application of Scientific Workflows for Missing Data Imputation in Big Data

Rodrigo Tavares de Souza

Advisors:
Jorge de Abreu Soares

Abstract of the qualification exam submitted to Programa de Pós-graduação em Ciência da Computação - Centro Federal de Educação Tecnológica Celso Suckow da Fonseca CEFET/RJ as partial fulfillment of the requirements for the degree of master.

The amount of data generated by computerized systems is increasing. The term *Big Data* has never been so relevant and the field of scientific research has as one of its pillars the processing of complex mathematical models on huge masses of data. Systems based on artificial intelligence techniques gain more and more ground and are able to infer knowledge on a scale never seen before. The increase in the volume of data in turn maximizes the problem of data inconsistencies, especially the lack of data, which can seriously compromise the quality of the analyzes produced.

An alternative to the complementation of missing data is imputation, the central theme of this work. This dissertation presents a scientific workflow system for experimenting imputation techniques in Big Data environments. The system proposes the imputation techniques by means, by the K-NN algorithm and by Back Propagation neural networks, in addition to the composite imputation, which consists in the composition of grouping and selection techniques with K-Means and PCA algorithms for the creation of imputation plans. The System also presents a solution for graphical modeling of imputation plans.

Key-words:
Imputation; Scientific Workflow; Big Data.

Rio de Janeiro,

November 2017

Tavares, Rodrigo.

Aplicação de Workflows Científicos para Imputação de Dados Ausentes em Big Data / Rodrigo Tavares de Souza – 2017.

x, 38 f; enc.

Exame de Qualificação para Dissertação (Mestrado), Centro Federal de Educação Tecnológica Celso Suckow da Fonseca, 2017.

Bibliografia: f, 34–38

1. Imputacão 2. Workflow Científico 3. Big Data I. Título

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

Capítulo I Introdução

É cada vez maior a quantidade de dados gerados, processados, gerenciados e armazenados por sistemas informatizados. Dispositivos de armazenamento tornaram-se mais acessíveis, apresentam melhor desempenho e maior capacidade. Tais fatos contribuíram para que o volume de dados armazenados e, consequentemente, processados, aumentasse massivamente nos últimos anos Khan et al. [2014]. Soma-se a isso a significativa e crescente melhora da infraestrutura (redes e internet) e a crescente utilização de sistemas distribuídos. A figura I.1 mostra o aumento da produção de HDs entre os anos de 1976 e 2013.

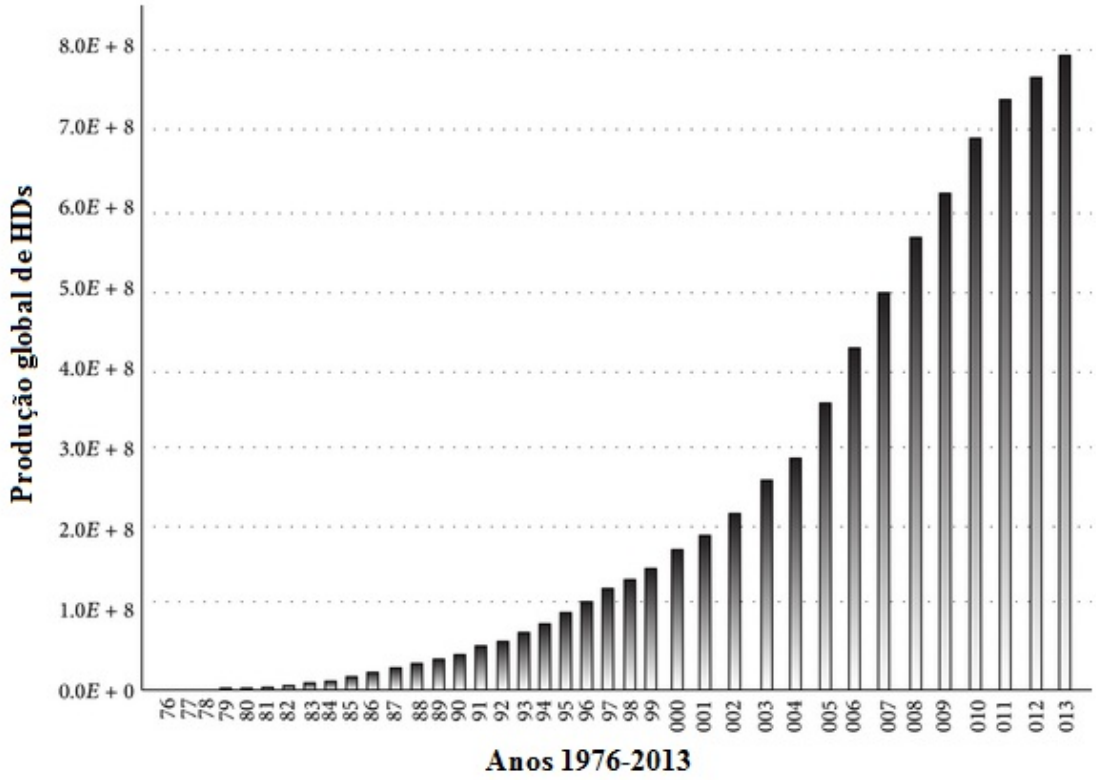

Figura I.1: Produção global de HDs. Fonte: adaptado de Khan et al. [2014].

Novos sensores e transmissores são capazes de prover diversos tipos de informação. Permitem que até mesmo eletrodomésticos sejam capazes de gerar dados. A *Internet of Things (IoT)* tende a transformar os campos físico e digital em um só [Atzori et al., 2010]. A integração entre dispositivos, o acesso a internet por parte destes e, consequentemente, a grandes data centers e nuvens, contribuem para o aumento da quantidade de dados armazenados.

O termo *Big Data* representa essa nova escala de volume de dados. Decisões anteriormente baseadas em suposições, ou sobre modelos de realidade meticulosamente trabalhada, são agora feitas através de modelos matemáticos [Jagadish et al., 2014]. A descoberta de informações nesse nível demanda uma estrutura de processamento de dados apoiada em arquiteturas de computação distribuída [Chen et al., 2014]. Devem compor uma plataforma de alto desempenho, processando a maior quantidade de dados possível em tempo adequado.

Grandes volumes de dados podem esconder informações valiosas capazes de definir o rumo estratégico das organizações. É possível criar um processo de descoberta de conhecimento em base de dados [M. Fayyad et al., 1996]. O termo KDD remete a este processo. O comportamento de clientes, tendências, características de negócio, entre outros, são muitas as informações que podem ser descobertas. Entretanto para que dados sejam transformados em conhecimento existe um longo caminho a ser percorrido. A área de KDD procura descobrir e analisar as relações intrínsecas existentes em um conjunto de dados, gerando conhecimento [Fayyad et al., 1996].

O aumento no volume de dados por sua vez maximiza um problema conhecido dos administradores de dados: as inconsistências de dados. Essas inconsistências são fruto primário dos diversos momentos onde bases de dados são integradas, com vistas ao aumento da qualidade das análises produzidas. Essas bases vêm de diversas fontes, que nem sempre recebem o devido cuidado. Podem também ocorrer por outros motivos, tais como falhas de equipamentos, na transmissão da mensagem, erros de preenchimento não tratados, falhas em rotinas de carga, entre outros [Han et al., 2011].

Essas ausências, dependendo de sua natureza e incidência, podem prejudicar sobremaneira a análise de dados por qualquer técnica produtora de informação, tais como a própria KDD, os armazéns de dados *(Data Warehouse)*, ou similares, e comprometer seus resultados [Little and Rubin, 1986]. Para isso, existe um campo de estudo denominado “Imputação”, que busca resolver esses problemas, complementando dados ausentes fundamentalmente com base em métodos estatísticos e de inteligência computacional [Little, 2015]. Esses métodos podem ter abordagens simples, com resultados razoáveis, ou um pouco mais complexas, de acordo com o mecanismo de ausência apresentado [Gelman and Hill, 2007].

A computação hoje é uma das forças motrizes da ciência. A evolução dos supercomputadores, soluções de processamento paralelo e distribuído, técnicas de inteligência artificial, são áreas que impulsionam a capacidade de processamento computacional e, consequentemente, o avanço científico. Entretanto, é necessário que o trabalho, as diversas etapas e repetições de tarefas dentro de um experimento, sejam gerenciadas de forma organizada [Deelman et al., 2009]. Garantir a gravação dos resultados, acompanhar a evolução do processo como um todo e garantir a reprodutibilidade dos experimentos são características de sistemas de *workflow* ci-

entífico [Davidson and Freire, 2008]. Sistemas de workflow científico apoiam a experimentação, possibilitando que cientistas analisem diversos tipos de dados, técnicas, algoritmos, criando um ambiente agradável e, preferencialmente, de fácil utilização, sendo este um dos objetivos principais do campo de estudo denominado *e-science* [Deelman et al., 2009].

Este trabalho propõe a criação de um sistema de workflow científico que permita a experimentação de diversas técnicas de imputação sobre bases de dados *Big Data*. Possibilitará também a composição gráfica de experimentos, organizados em planos de imputação [Soares, 2007]. Garantirá a proveniência e reprodutibilidade dos experimentos. Além disso, permitirá a utilização de diferentes algoritmos em cada etapa dos planos criados, propiciando um ambiente agradável ao cientista de dados.

Primeiramente são abordados os conceitos necessários ao entendimento deste trabalho. Em um segundo momento, serão explicados os trabalhos relacionados, algoritmos e soluções referentes ao tema imputação. Por fim, será detalhada a arquitetura do sistema proposto e apresentados os resultados dos experimentos iniciais.

Capítulo II Preliminares

Neste capítulo serão introduzidos os conceitos de imputação, o posicionamento da técnica no processo de *KDD* e as tarefas de mineração de dados relevantes para o sistema proposto. Além disso, serão abordados os conceitos de *Big Data* e de sistemas de workflow científico.

II.1 Descoberta de Conhecimento em Bases de Dados

A necessidade de ampliar as capacidades de análise humana para inferir conhecimento sobre grandes volumes de dados possui motivações econômicas e científicas. Empresas utilizam dados para ganhar vantagem competitiva, aumentar a eficiência e fornecer serviços mais valiosos para os clientes. A ciência, por sua vez, busca interpretar dados que geram informações sobre o meio ambiente, o espaço, fenômenos naturais, entre outros. Essas informações ajudam na formulação de teorias e modelos que apoiam o desenvolvimento científico [Han et al., 2011].

A descoberta de conhecimento em base de dados tem origem no termo *KDD*, que consiste num processo de várias etapas, não trivial, interativo e iterativo, para identificação de padrões compreensíveis, válidos, novos e potencialmente úteis, a partir de grandes conjuntos de dados [M. Fayyad et al., 1996]. As aplicações de *KDD* são diversas e buscam desenvolver métodos e técnicas que criem sentido para os dados.

O processo possui dois objetivos: verificação e descoberta [Fayyad et al., 1996]. No de verificação, o sistema limita-se a avaliar hipóteses pré-estabelecidas. No de descoberta, o sistema é capaz de identificar automaticamente padrões ocultos nos dados. A descoberta pode ser subdividida em preditiva – onde o sistema descobre padrões que indicam um comportamento ou acontecimento futuro – e descritiva, que encontra padrões a serem apresentados de forma inteligível.

O processo de descoberta de conhecimento em bases de dados é composto pelas seguintes etapas: seleção, pré-processamento, transformação, mineração de dados, avaliação de padrões e apresentação do conhecimento [Han et al., 2011]. A figura II.1 mostra o caminho percorrido desde os dados originais, passando pelas tarefas de *KDD*, até a geração do conhecimento.

É comum observar o processo de *KDD* sendo referido simplesmente como Mineração de Dados, já que esta é, por diversas vezes, considerada a etapa mais importante do trabalho. Neste

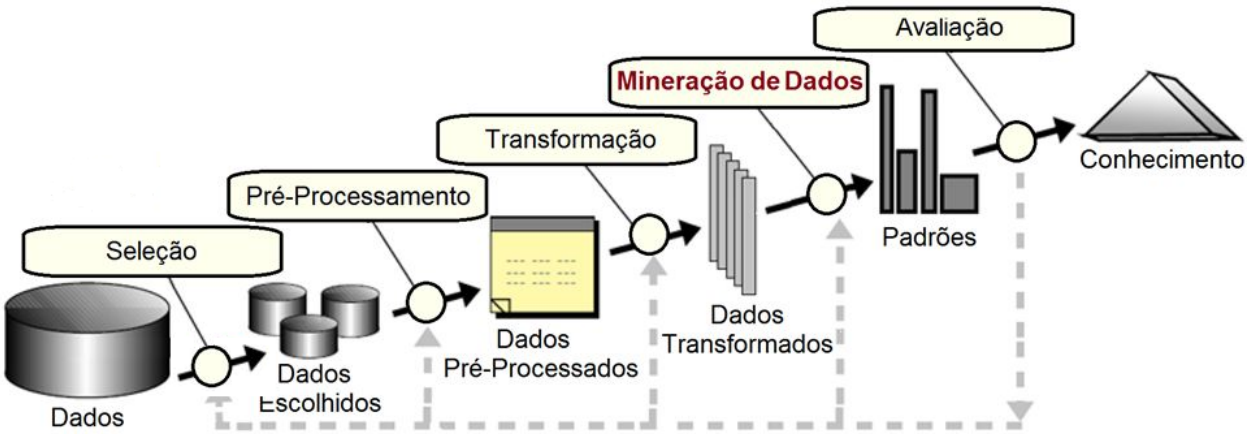

Figura II.1: O processo de KDD. Fonte: adaptado de Han et al. [2011].

trabalho serão abordadas especificamente as tarefas de Pré-Processamento e de Mineração de Dados, que são onde se empregam as técnicas de imputação e os algoritmos utilizados no sistema proposto.

II.2 Processo de Mineração de Dados

Mineração de dados é o processo de descoberta de padrões e geração de conhecimento sobre grandes volumes de dados [Han et al., 2011]. As fontes de dados podem incluir bancos de dados, armazéns de dados, a Web, repositórios de informação, entre outros. É durante esta etapa que as relações entre os elementos da base de dados são descobertas.

Hand et al. [2001] apresenta um conceito interessante, sob a ótica da representatividade dos resultados gerados na etapa de mineração de dados:

"A análise de (quase sempre grandes) conjuntos dados observados para descobrir relações escondidas e para consolidar os dados de uma forma tal que sejam inteligíveis e úteis aos seus donos".

Tornar os dados inteligíveis significa apresentá-los de forma que informações relevantes possam ser obtidas sobre os mesmos. Informações essas que gerem conhecimento e agreguem valor científico ou estratégico. Para tal, uma série de técnicas podem ser utilizadas, dependo do tipo de análise que se deseja realizar. As tarefas do processo de mineração de dados mais utilizadas são [Han et al., 2011]:

- Regras de associação: baseia-se nos conceitos de suporte e confiança para inferir associações entre os atributos de um conjuntos de dados. Identifica objetos como antecedentes e consequentes, quando é observado o acontecimento de um fenômeno em consequência de outro.
- Classificação: extrai modelos que descrevem classes de dados. Tais modelos, chamados classificadores, predizem rótulos de classe para novos valores inseridos na amostra.

• Agrupamento: tem como objetivo reunir em um mesmo grupo objetos que mantenham algum grau de afinidade. Busca maximizar a similaridade de objetos do mesmo grupo e minimizar a similaridade entre elementos de grupos distintos.

• Previsão de Séries Temporais: uma série temporal é um conjunto de observações de um fenômeno ordenadas no tempo. Sua análise consiste no processo de identificação das características, padrões e propriedades, utilizados para descrever o fenômeno gerador. Tem como principal objetivo a geração de modelos voltados à previsão de valores futuros.

• Detecção de *Outliers*: um outlier é um objeto cujo valor desvia significativamente dos demais na amostra, como se fosse gerado por um mecanismo diferente do usual.

Dados podem possuir diferentes formas e tipos. A sessão II.2.1 aborda os tipos de atributos existentes em conjuntos de dados.

II.2.1 Natureza dos dados

Os atributos de uma tabela podem ser divididos em duas classes: numéricos e categóricos. Atributos numéricos são quantitativos, apresentam ou não limites inferior e superior. Subdividem-se em atributos contínuos e discretos. Atributos contínuos são aqueles que podem ser medidos em uma escala contínua, como peso e temperatura, por exemplo. Atributos discretos são os que representam um número finito ou uma quantidade enumerável, como o número de membros de uma equipe, por exemplo.

Atributos categóricos ou nominais indicam o conceito ao qual o objeto se enquadra, revelando a categoria em que o mesmo se encontra. Neste tipo de atributo não existe a noção de ordem entre os valores. Podem ser representados por tipos alfanuméricos.

Para que tarefas de mineração de dados gerem resultados completos, é necessário que os dados estejam íntegros. Dados com erros, incompletos, redundantes ou desnivelados, podem comprometer a qualidade da análise. Esses problemas são tratados na etapa de pré-processamento.

II.2.2 Pré-Processamento

Dados possuem qualidade se satisfazem os requisitos do uso pretendido. Há muitos fatores que compõem a qualidade dos dados, entre eles: precisão, integridade, consistência, pontualidade, credibilidade e interpretabilidade [Han et al., 2011].

A etapa de pré-processamento possui como principais atividades [Goldschmidt et al., 2015]:

- Agrupamento de dados
- Coleta e Integração
- Codificação
- Construção de Atributos

- Correção de Prevalência
- Discretização
- Enriquecimento
- Limpeza dos Dados
- Normalização de Dados
- Partição dos Dados
- Redução de Dados
- Seleção de Dados

A figura II.2 ilustra algumas atividades da etapa de pré-processamento de dados.

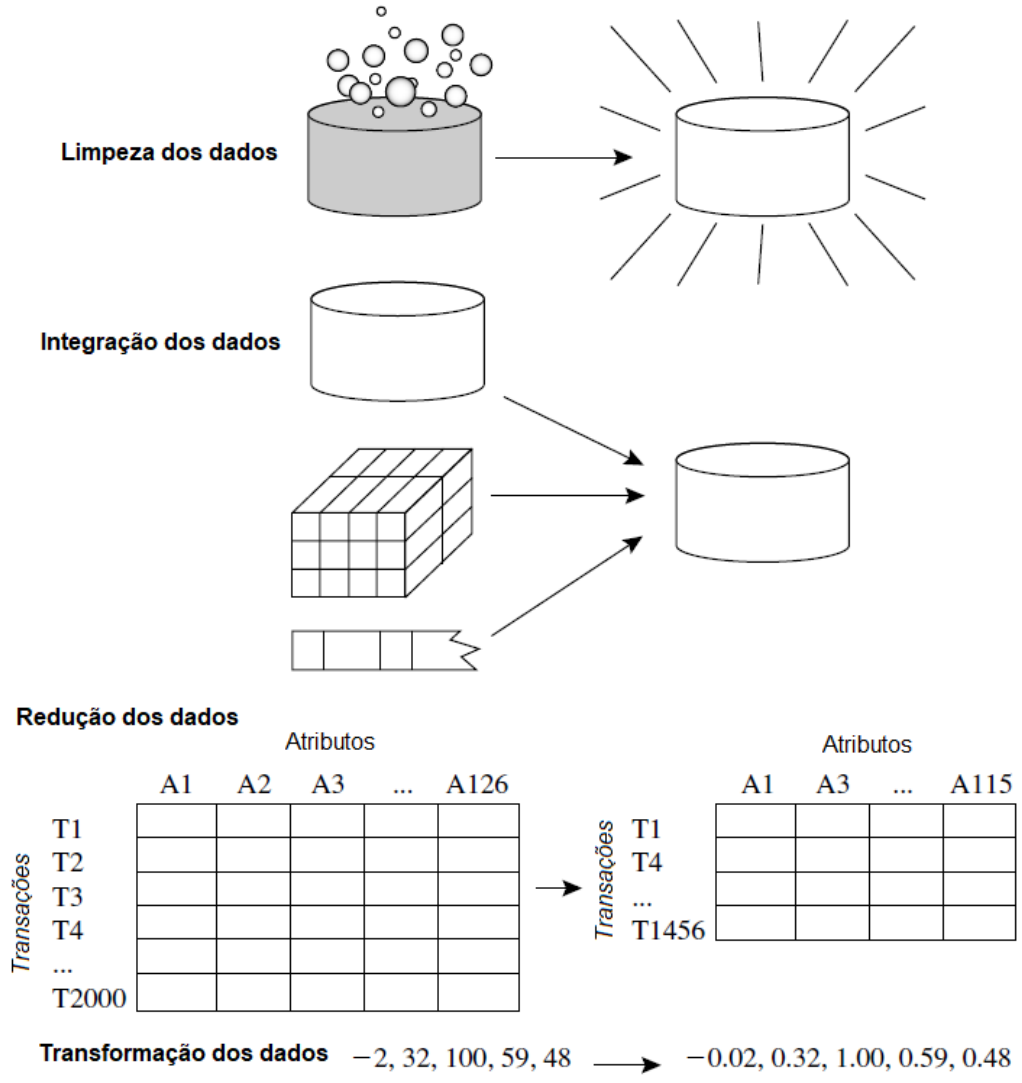

Figura II.2: A etapa de pré-processamento de dados. Fonte: adaptado de Han et al. [2011].

A tarefa de limpeza de dados tem como objetivo [Han et al., 2011]:

- Complementar dados ausentes.
- Suavizar ruídos através da identificação de *outliers*.
- Corrigir possíveis inconsistências nos dados.

A tarefa de complementação de dados ausentes será abordada com mais detalhes, pois é a que compreende a subtarefa de imputação, tema central deste trabalho.

II.2.3 Complementação de Dados Ausentes

Tanto em tarefas de treinamento como em testes de modelos de KDD, o significado da análise depende de duas premissas: a precisão do banco de dados e a qualidade dos dados da amostra. De acordo com Brown and Kros [2003], a ausência de dados deve ser corretamente tratada, pois ignorar esse problema pode introduzir uma polarização nos modelos avaliados, levando a conclusões incorretas no processo de mineração de dados.

Existem métodos mais avançados para o tratamento de ausências de dados, como a imputação, explicada em mais detalhes na sessão II.3. Entretanto, existem métodos convencionais, menos eficientes, que tratam a ausência de dados simplesmente removendo-a da amostra. Magnani [2004] classificou-os como:

a) Remoção completa de casos *(Listwise Deletion)* - É a maneira mais fácil de obter um conjunto de dados completo. Somente registros completos são mantidos. Este método é de fácil implementação, mas possui a desvantagem de que, em situações reais, muitos dados podem ser perdidos.

b) Remoção em pares *(Pairwise Deletion)* - É uma variante da remoção completa de casos. Utiliza um registro incompleto somente quando a variável analisada não é ausente. A vantagem desta técnica é poder utilizar todos os dados quando os valores analisados não apresentam dependência de valores ausentes. Como desvantagem, sua implementação é mais complexa, pois demanda a análise da relação entre os atributos do conjunto de dados.

c) Remoção de colunas com valores ausentes - Qualquer atributo que apresente valores ausentes é inteiramente removido. Sua aplicação na maioria das vezes não é recomendada, já que a perda de informação com a remoção de atributos é significativa.

II.2.4 Mecanismos de ausência de dados

Ausências de dados normalmente seguem um mecanismo de distribuição, mas também podem ocorrer de forma intermitente ou simplesmente ao acaso [Little and Rubin, 1986]. Os motivos de ocorrência são vários: integrações entre bases de dados, falhas de equipamentos, na transmissão de mensagens, erros de preenchimento não tratados, falhas em rotinas de carga, entre outros.

Trabalhos envolvendo a complementação de dados ausentes levam inevitavelmente em conta o mecanismo que causou a ausência dos dados. Little and Rubin [1986] classificaram esses mecanismos em três tipos: completamente aleatório - Missing Completely at Random (MCAR),

aleatório - Missing at Random (MAR) e não aleatório - Not Missing at Random (NMAR).

MCAR: é o nível mais alto de aleatoriedade. Ocorre quando a probabilidade de uma instância possuir valores ausentes, para um determinado atributo, não depende dos valores conhecidos e nem dos ausentes. Nesse nível de aleatoriedade, qualquer tratamento pode ser aplicado sem risco de enviesar os dados.

MAR: ocorre quando a probabilidade de uma instância possuir valores ausentes depende de valores conhecidos, ou seja, de atributos diferentes do que possui a ausência. Por exemplo, um atributo que registra os custos de reparos realizados em uma oficina, não possui alguns valores quando o carro é da marca Ford (outro atributo).

NMAR: nesse caso a ausência é relacionada ao próprio atributo que possui valor ausente. Por exemplo, um atributo que registra a velocidade máxima dos carros em uma via não possui registros para velocidades maiores que 180 km/h.

As características da ausência de dados devem ser analisadas para a escolha do tratamento adequado. Dados com alto nível de ausência, ou cujos valores ausentes não sigam uma aleatoriedade, devem ser tratados por métodos robustos que sigam um modelo preditivo [Batista and Monard, 2001]. Em um modelo preditivo, o atributo com valores ausentes é utilizado como classe (atributo de saída) e os demais atributos como entrada, para a composição de valores mais precisos para a substituição da ausência.

II.3 Imputação

Existem diversos métodos para tratar a ausência de dados, ou seja, substituir valores ausentes por valores reais [Rubin, 1988]. A tarefa de imputação tem como objetivo recuperar valores ausentes de maneira mais precisa, através de técnicas que variam desde a média simples, regressão linear, modelos preditivos específicos, até a utilização de algoritmos de aprendizado de máquina. O método de imputação utilizado depende primordialmente da natureza do atributo imputado.

As relações entre os atributos de amostra também podem revelar informações valiosas para o processo de imputação. O experimento de Agrawal et al. [1993], por exemplo, mostrou que numa padaria, noventa por cento das vendas que envolviam pão e manteiga também envolviam leite.

Imagine uma situação hipotética, um conjunto de dados que possui atributos com tuplas completas para as compras de pão e manteiga, mas alguns registros de compras de leite estão ausentes, para essas mesmas tuplas. A probabilidade dos registros ausentes confirmarem a compra de leite é, no mínimo maior, do que a de não confirmarem. Imputar dados que confirmem essa afirmação seria a ação mais correta.

A existência de relações entre os atributos de uma amostra nem sempre pode ser assumida

como verdade absoluta para o processo de imputação. Presumir que os valores ausentes de um atributo estão relacionados a todos os demais atributos pode levar a um modelo de imputação mal especificado [Austin and Escobar, 2005]. É necessário inferir um nível maior de confiança a respeito dessas relações. Soares [2007] obteve bons resultados utilizando o algoritmo PCA para analisar a correlação de atributos na etapa de seleção do processo de imputação composta, que será explicado com mais detalhes na sessão III.5.

Métodos de imputação são classificados em dois grupos Soares [2007]: métodos simples e métodos híbridos. Métodos simples consideram apenas um valor proposto para cada valor ausente. Métodos híbridos combinam dois ou mais métodos de imputação simples, com o objetivo de melhorar a qualidade do processo de imputação como um todo [Rubin, 1988].

Como exemplo de método híbrido, Raghunathan [2004] propõe o método de imputação múltipla, onde o conjunto de valores ausentes é substituído por mais de um conjunto plausível de valores. A criação de mais de um, no caso *n* plausíveis conjuntos de substituição para valores ausentes. Consequentemente gerando *n* conjuntos completos. A variação em todos os conjuntos completos, medida através do desvio padrão, reflete a incerteza da imputação. C Yuan [2005] propõe uma abordagem parecida, entretanto após a identificação do método com melhor desempenho, novas inferências estatísticas são realizadas de forma a ajustar os hiperparâmetros dos métodos utilizados. Esse processo minimiza o erro da imputação.

Métodos de imputação normalmente pressupõem que a distribuição da ausência seja MAR. Allison [1999] aborda uma técnica de imputação múltipla em que essa premissa não é necessária. O método identifica relações de linearidade entre os atributos e utiliza um modelo de regressão linear para gerar o valor imputado. Entretanto, essa técnica não obteve um nível alto de precisão para todas bases de dados testadas.

Inspirado no modelo de imputação composta [Soares, 2007], este trabalho propõe um sistema que permita a experimentação de diversos métodos de imputação, onde tarefas de seleção, agrupamento e regressão possam ser organizadas em estratégias chamadas planos de imputação. O sistema permitirá também que outras tarefas sejam criadas ao longo do processo de imputação, modeladas de acordo com a necessidade do cientista de dados. Os algoritmos utilizados serão explicados com mais detalhes no capítulo III.

Este trabalho abordará a imputação de dados numéricos. Inicialmente a imputação de dados categóricos não faz parte do escopo do sistema.

II.4 Big Data

Nos últimos vinte anos, o volume de dados gerados por sistemas informatizados cresceu em larga escala. De acordo com um informativo do grupo *International Data Corporation (IDC)*, em

2011 o volume de dados gerados no mundo cresceu quase nove vezes em relação aos anos anteriores [Chen et al., 2014]. A projeção é de que o volume de dados dobrará a cada dois anos em um futuro próximo. O planejado telescópio SKA (Square Kilometre Array), por exemplo, irá produzir até um milhão de terabytes de dados brutos por dia a partir de 2020 [Jagadish et al., 2014].

Vultuosos volumes de dados podem ser gerados tanto por sistemas de business convencional, como os encontrados em sistemas Enterprise Resource Planning (ERP), quanto por sistemas mais complexos, como os biomoleculares, georreferenciados ou espaciais, que demandam grandes esforços de análise e processamento. Existem ainda sistemas que se destacam pelo alcance, como as redes sociais, com o uso combinado em desktops e aplicativos de smartphone, entre outros, capazes de gerar enormes quantidades de dados diariamente.

A chamada IoT tem como idéia básica a presença generalizada de uma variedade de coisas ou objetos, como tags, sensores, atuadores, telemóveis, entre outros, que são capazes de interagir uns com os outros para atingir objetivos comuns [Atzori et al., 2010]. Praticamente qualquer dispositivo é capaz de gerar dados, que posteriormente serão transferidos para grandes *data centers* e nuvens.

Esses imensos volumes de dados remetem ao termo *Big Data*. O modelo dos cinco Vs [Chen et al., 2014] é amplamente utilizado para a caracterização de Big Data: volume (grandes volumes), variedade (diversas modalidades), velocidade (geração rápida), veracidade (dados verdadeiros) e valor (valor enorme, porém de baixa densidade) são características de bases big data.

Métodos para descobrir conhecimento em *big data* são fundamentalmente diferentes dos utilizados na análise estatística tradicional ou em pequenas amostras. Dados da escala *big data* normalmente apresentam ruído, são dinâmicos, heterogêneos e inter-relacionados. Contudo, possuem maior valor significativo, pois a mineração de dados nesses ambientes ignora flutuações individuais, identificando padrões ocultos e gerando conhecimento de maneira mais confiável [Jagadish et al., 2014].

A descoberta de conhecimento nesse nível demanda uma estrutura de processamento de dados apoiada em arquiteturas de computação distribuída [Chen et al., 2014]. Devem compor uma plataforma de alto desempenho, capaz de processar grandes volumes de dados em tempo satisfatório.

II.5 Workflow Cientifíco

Atualmente a computação tem se estabelecido como uma das bases do avanço científico. Supercomputadores são capazes de simular modelos complexos envolvendo diversos passos computacionais. A execução de experimentos envolve tarefas repetitivas, com movimentos cíclicos

dos dados. São gerados resultados de entrada e saída durante a realização de tarefas isoladas e ao longo de toda a execução da repetição. Esse processo se repete até a obtenção dos resultados finais. Sistemas de Workflow Científico automatizam tarefas cíclicas, de forma que os cientistas possam se concentrar somente nas pesquisas e não no gerenciamento computacional [Deelman et al., 2009].

A área de *e-Science* remete a utilização de métodos de obtenção de resultados científicos através da utilização de computação intensiva. De acordo com [Deelman et al., 2009], sistemas de workflows de *e-Science* tem como principal objetivo oferecer ambientes de programação especializados, que simplifiquem o esforço de programação despendido por cientistas para orquestrar experimentos científicos em ambientes computacionais.

A composição é uma etapa importante na elaboração de workflows. Nessa etapa o cientista especifica os passos e as dependências envolvidas desde o início da execução até o final da realização do experimento. Uma forma de simplificar a elaboração de workflows é fornecer ferramentas gráfica para a composição dos experimentos. De acordo com Deelman et al. [2009], em um nível abstrato, workflows são compostos por séries de unidades funcionais, sejam componentes, tarefas, serviços ou dependências, executados em uma ordem definida. Podem ser representados por grafos Directed Acyclic Graph (DAG), redes de *petri*, *Business Process Model and Notation* (BPMN), Unified Modelling Language (UML), entre outros.

Sistemas de workflow são classificados de duas formas:

a) *Data centric* - o foco é a análise e transformação dos dados.

b) *Process centric* - o foco é a definição do fluxo de negócio.

Cientistas tendem a ter uma visão centrada nos dados para suas análises [McPhillips et al., 2009]. Passos computacionais são importantes, mas a obtenção, análise e criação de informação através dos dados tendem a ser mais relevantes.

Uma questão fundamental na temática de workflows é a capacidade de proveniência do sistema. Proveniência permite interpretar e entender resultados, através da análise da sequência de passos executados desde o início do processo até o resultado final [Freire et al., 2008]. Dessa forma, é possível obter informações sobre o encadeamento das tarefas, avaliando se estas foram executadas seguindo os procedimentos esperados. Outro ponto fundamental a ser garantido pela proveniência é a capacidade de reprodutibilidade dos experimentos [Davidson and Freire, 2008]. Sistemas de workflows devem ser capazes de salvar os dados da execução para que possam ser reproduzidos posteriormente.

Quando se trata de tarefas computacionais, existem duas formas de proveniência [Freire et al., 2008]: prospectiva e retrospectiva. A proveniência prospectiva captura a especificação de uma tarefa computacional (seja um script ou um workflow) e corresponde às etapas que devem ser

seguidas para gerar um determinado resultado. A proveniência retrospectiva capta as etapas executadas, bem como informações sobre o ambiente usado para derivar o resultado - em outras palavras, é uma espécie de *log* detalhado da execução das tarefas computacionais.

Capítulo III Trabalhos Relacionados

Neste capítulo serão abordados trabalhos relacionados a imputação. Primeiramente, serão explicados os algoritmos relacionados ao tema. Posteriormente, será detalhada a técnica de imputação composta. Por fim, serão abordadas soluções de computação distribuída e processamento em larga escala.

III.1 Seleção de Atributos com o Algoritmo PCA

O algoritmo PCA é um método de redução de dimensionalidade. Suponha que em uma amostra, os dados a serem reduzidos consistem em tuplas ou vetores de dados descritos por *n* atributos ou dimensões. O algoritmo busca por vetores ortogonais *n*-dimensionais que possam melhor representar os dados, onde [k <= *n*] [Han et al., 2011]. Os dados originais são projetados em um espaço menor, resultando em redução de dimensionalidade. É um algoritmo muito utilizado para compressão de imagens.

Além da redução de dimensionalidade, o algoritmo PCA é uma maneira de identificar padrões em dados e expressá-los de forma a destacar suas semelhanças e diferenças [Smith, 2005]. Neste trabalho, será utilizado para inferir a correlação entre os valores dos atributos em um conjunto de dados. Dessa forma, permite selecionar os mais representativos para utilização no processo de imputação.

Funcionamento do algoritmo:

1) Os dados de entrada são normalizados, para evitar que valores de grandezas maiores enviesem os cálculos.

2) São computados *k* vetores ortonormais, que servem como base para os dados de entrada normalizados. Vetores unitários são gerados, de forma que cada ponto em uma direção é perpendicular aos demais. Esses vetores são denominados componentes principais.

3) Os componentes principais são ordenados em ordem decrescente de significância ou força. Servem como um novo grupo de eixos representando os dados, provendo informação sobre a variância. A matriz de componentes possui os eixos ordenados da esquerda para a direita em ordem de representatividade.

4) Como os componentes são classificados em ordem decrescente de significado, o tamanho

dos dados pode ser reduzido eliminando os componentes menos representativos.

Trabalhos recentes buscam otimizar o funcionamento do algoritmo para ambientes *Big Data*, com foco na melhora de desempenho. Elgamal et al. [2015] implementou mudanças no algoritmo PCA em Spark. Através da alavancagem da esparsidade de matrizes, foi observado ganho de desempenho em operações matemáticas.

[Wu et al., 2016] apresenta uma versão do algoritmo PCA otimizada para computação paralela e distribuída, de forma a ajustar a utilização da matriz de correlação ao modelo *map-reduce*. O algoritmo foi executado em Spark e apresentou ganhos significativos de *speedup* no experimento realizado.

III.2 Agrupamento de Dados com o Algoritmo K-Means

O K-Means é um algoritmo apresentado por MacQueen [1967], que busca particionar os dados em *k* grupos cujas instâncias apresentem semelhanças entre si. É um algoritmo consolidado e com ampla utilização em tarefas de agrupamento.

O algoritmo funciona da seguinte forma:

1) São passados como entrada o conjunto de dados D e o número *k* de grupos que se deseja encontrar.

2) Arbitrariamente, são selecionados *k* objetos de D, denominados centroides.

3) Para cada objeto de D, é verificada a distância do mesmo em relação aos centroides. O objeto é agrupado junto ao centróide mais próximo.

4) Ao fim da verificação de todos os objetos de D, é calculada a média simples entre os objetos de cada grupo, definindo os valores dos novos centroides.

5) O processo reinicia no passo 3, até que não ocorra mais mudança de grupo entre os objetos ou um número máximo definido de iterações seja alcançado.

A figura III.1 denota o funcionamento do algoritmo:

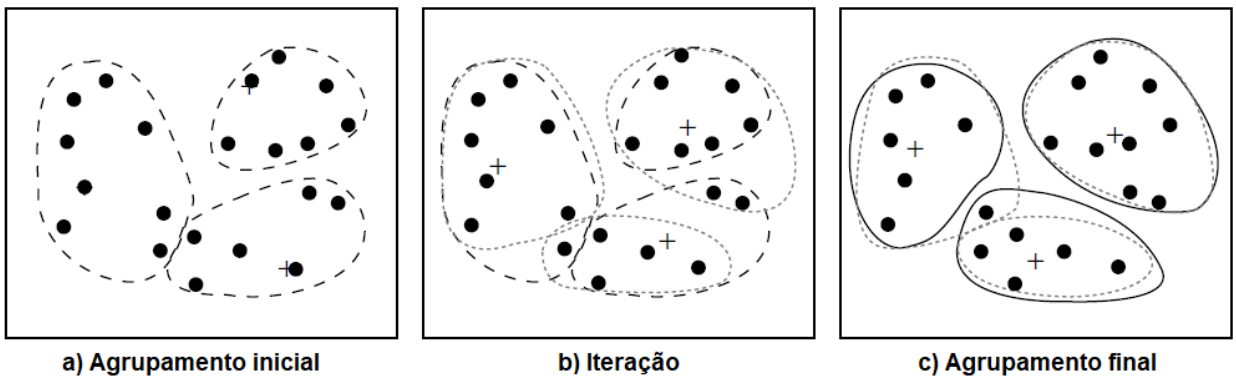

Figura III.1: O algoritmo K-Means. Fonte: adaptado de Han et al. [2011].

O algoritmo K-Means pressupõe que cada tupla seja agrupada de forma que pertença a um único grupo. Entretanto, em casos reais, é possível observar situações em que um objeto pode pertencer a mais de um grupo. Existe um campo de estudo chamado *soft computing*, que busca gerenciar imprecisões, incertezas e soluções inexatas. Baseado nessa idéia, Li et al. [2004] propõe a utilização do algoritmo Fuzzy K-Means como tarefa de agrupamento predecessora a de imputação. Nessa versão do algoritmo, o centroide é calculado como a média de todos os pontos, ponderada pelo grau de inclusão do objeto em cada grupo.

III.3 Imputação com o Algoritmo K-NN (K-Nearest-Neighbor)

O algoritmo *K-Nearest-Neighbor* (K-NN) tem como base o aprendizado por analogia, isto é, compara tuplas que sejam similares a tupla analisada [Han et al., 2011]. Em uma amostra com n atributos, cada tupla representa um ponto em um espaço n-dimensional. As tuplas são armazenadas em um espaço de padrão de tamanho n. Dada uma tupla desconhecida, o algoritmo busca o espaço de padrão para as k tuplas mais próximas da tupla desconhecida. Essas k tuplas mais próximas são os vizinhos mais próximos da tupla desconhecida.

O método de imputação baseado no algoritmo K-NN é amplamente utilizado e pode apresentar bons resultados para valores desconhecidos que sigam distribuição MCAR [Batista, 2003]. Valores imputados com o algoritmo K-NN são geralmente mais bem comportados, uma vez que são relacionados a valores de outros atributos. Normalmente, quanto mais atributos com valores desconhecidos, e quanto maior a quantidade de valores desconhecidos, mais simples são os classificadores induzidos [Batista and Monard, 2003b]. Dessa forma, a imputação deve ser cuidadosamente aplicada, sob o risco de simplificar demasiadamente o problema em estudo.

Descobrir o melhor valor k para imputação não é uma tarefa objetiva. Uma estratégia que apresenta bons resultados, é considerar a substituição de valores ausentes pelo correspondente valor de atributo da instância completa mais semelhante no conjunto de dados [Hruschka et al., 2005]. Em outras palavras, utilizar o algoritmo K-NN com [$k = 1$] e uma função de distância, como a euclidiana, para identificar os vizinhos mais próximos. Entretanto, Jonsson and Wohlin [2004] sugerem que, para alguns casos, como em dados de pesquisas do tipo *likert data*, utilizar k como a raiz quadrada do número de casos completos (arredondado para o número inteiro mais próximo) pode apresentar melhores resultados. Logo, no contexto de imputação, o valor ótimo de k pode variar dependendo das características dos dados.

A principal desvantagem do algoritmo K-NN é que, sempre que os vizinhos mais semelhantes são procurados, o algoritmo pesquisa todas as tuplas do conjuntos de dados. Isso pode ser um ponto crítico para a execução em grandes volumes de dados [Batista and Monard, 2003a].

Otimizações do algoritmo K-NN para aplicações de imputação são vastas na literatura. Huang

and Lee [2004] propõe uma modificação no algoritmo K-NN que utiliza *Grey Relational Analysis (GRA)* como métrica de distância para determinar os vizinhos mais próximos de instâncias com valores ausentes.

García-Laencina et al. [2009] propõe uma variação do algoritmo K-NN para classificação e imputação simultânea de valores ausentes. Utiliza o conceito de informação mútua, que é uma medida natural para calcular distâncias entre variáveis aleatórias. As distância entres os atributos que possuem valores ausentes é considerada para criação de classificadores, melhorando o valor imputado.

Existem adaptações do algoritmo K-NN que apresentam resultados melhores para dados com características específicas. Por exemplo, para séries temporais *Big Data*, Talavera-Llames et al. [2016] propõe um novo método de previsão com base na técnica *Weighted Nearest Neighbours (WNN)*, que utiliza vetores correlacionados para identificar vizinhos mais próximos.

O método K-NN-IS apresenta melhorias para execução do algoritmo K-NN em ambientes *Big Data* com Spark [Maillo et al., 2017]. A tarefa de *map* computa a distância dos vizinhos mais próximos em diferentes particionamentos de dados. Múltiplos *reducers* processam os vizinhos da lista gerada na tarefa de *map*.

III.4 Imputação com Redes Neurais Back Propagation

A área de Aprendizado de Máquina (AM) é uma subárea da Inteligência Artificial, que tem como objetivo o desenvolvimento de métodos e técnicas com a finalidade de encontrar padrões, regularidades ou conceitos em conjuntos de dados [Goldschmidt et al., 2015]. Pode ser divida em dois grupos Han et al. [2011]: aprendizado supervisionado e aprendizado não supervisionado. Aprendizado supervisionado é basicamente um sinônimo de classificação. A supervisão no aprendizado do algoritmo vem dos exemplos rotulados no conjunto de dados de treinamento. Aprendizado não supervisionado é essencialmente um sinônimo de agrupamento. O processo de aprendizagem é não supervisionado pois os exemplos de entrada não são rotulados pela classe, e sim aprendidos pelo algoritmo.

Rede neurais com aprendizado supervisionado por retropropagação de erros, ou redes *back propagation* [Rumelhart et al., 1986], utilizam uma arquitetura de redes neurais artificiais do tipo *feedforward*. A idéia fundamental é gerar uma rede capaz de ser continuamente treinada, a partir de um conjunto de dados de entrada e saída. As saídas produzidas pela rede são comparadas com as desejadas, ajustando os pesos das conexões sinápticas dos neurônios da rede em função da diferença (erro) entre a previsão e o valor original. Modelos de predição de valores baseados em redes neurais tendem a superar os de regressão linear quando relações não-lineares estão presentes nos dados [Bansal et al., 1993].

Métodos de imputação utilizando redes *back propagation* tem se tornado populares dado o advento da área de aprendizado de máquina. Jerez et al. [2010] por exemplo, utilizou técnicas baseadas em métodos estatísticos (média, *hot-deck* e imputação múltipla) e de aprendizado de máquina (*multi-layer perceptron*, *self-organising maps* e K-NN) para imputação em uma base de dados de pacientes com câncer de mama. As redes *back propagation* apresentaram melhores resultados.

Para aplicações em escala *big data*, algumas otimizações tem sido propostas no sentido de melhorar o desempenho do método. Liu et al. [2017] apresenta uma variação das redes back propagation com foco na paralelização de execução da tarefa de construção de classificadores. Utiliza a técnicas de *bootstrapping* para compartimentar o conjunto de dados original e *majority voting* para construir classificadores fortes baseados em classificadores fracos.

III.5 Imputação Composta

A imputação composta é uma técnica proposta por Soares [2007], baseada no conceito de estratégias e planos de imputação de dados. Combina uma ou mais tarefas usadas na etapa de mineração de dados, como por exemplo, agrupamento e seleção de colunas, para gerar valores imputados mais precisos.

O processo de imputação composta baseia-se na definição dos seguintes elementos:

• T_i: uma tarefa do processo de Descoberta de Conhecimento em Bases de Dados (KDD). Exemplos: T_1 = seleção de atributos, T_2 = agrupamento, T_3 = criação de regras de associação, T_4 = imputação, entre outros.

• $\rightarrow$: operador que define uma ordem de precedência de tarefas de KDD. A expressão $\rightarrow$ significa que a tarefa precede a tarefa . Exemplo: agrupamento $\rightarrow$ imputação significa que a tarefa de agrupamento precederá a de imputação.

• E(*v*,B): estratégia utilizada no processo de imputação de um atributo *v* de uma base de dados B. E(*v*,B) é representada por $T_1 \rightarrow T_2 \rightarrow ... \rightarrow T_m$, onde T_m é necessariamente uma tarefa de imputação.

• A_i : algoritmo utilizado no processo de imputação. Exemplos: A_1 = média, A_2 = algoritmo dos k vizinhos mais próximos.

• $\Rightarrow$: operador que define uma ordem de precedência de aplicação de algoritmos. A expressão $A_i \Rightarrow A_j$ significa que o algoritmo A_i é aplicado antes do algoritmo A_j.

• P(*v*,B): plano de imputação utilizado no processo de imputação de um atributo de uma base de dados B. P(*v*,B) é representada por $A_1 \Rightarrow A_2 \Rightarrow ... \Rightarrow A_p$, onde A_p é necessariamente um algoritmo de imputação. Exemplo: $A_1 \Rightarrow A_2 \Rightarrow A_3$ representa a aplicação sequenciada dos algoritmos A_1 = algoritmo dos K centroides, A_2 = análise de componentes principais e A_3 =

algoritmo dos k vizinhos mais próximos.

• Ψ_i: instância da aplicação de um algoritmo A_i, segundo parâmetros Θ_i = { Θ_{i1}, Θ_{i2}, ... , Θ_{iq} }. Ψ_i = f(A$_1$, Θ_i)

• I(v,B): instância de um plano de imputação de um atributo de uma base de dados , representada por uma sequência ordenada de q instâncias de aplicações de algoritmos. $\Psi_1 \Rightarrow \Psi_2 \Rightarrow$ .... Ψ_q, onde Ψ_q é necessariamente uma instância de aplicação de algoritmo de imputação.

• ε(I(v,B)): medida do erro na execução de uma instância de um plano de imputação do atributo .

O conjunto de valores assumidos por um plano de imputação será composto pelos valores da instância que apresentar o menor erro médio de todas as instâncias do referido plano.

III.6 Computação Distribuída e Processamento em Larga Escala com Apache Spark

Um dos requisitos principais de sistemas de workflow é conseguir processar tarefas de alto custo computacional de forma eficiente, através da utilização de recursos distribuídos [McPhillips et al., 2009]. Apache Spark é uma ferramenta unificada para processamento de dados distribuídos [Zaharia et al., 2016]. Possui um modelo de programação similar ao consagrado *MapReduce*, otimizado por uma abstração de compartilhamento de dados chamada *Resilient Distributed Datasets (RDD)*.

Através dos RDDs Spark consegue atuar em uma ampla faixa de processamento, de forma que processamentos que anteriormente necessitariam de *engines* separadas, como SQL, *streaming*, *machine learning*, entre outros, possam ser executados de maneira otimizada. Spark atualmente é uma das ferramentas mais utilizadas para processamento *Big Data*.

Spark SQL também é uma ferramenta relevante em processamento distribuído. Apoiada em Spark, possui uma abordagem mais próxima a bancos de dados relacionais. Integra o processamento relacional e procedural através de uma *interface* declarativa chamada *DataFrame* [Armbrust et al., 2015]. Além disso, possui um otimizador extensível chamado *Catalyst*, criado na linguagem *Scala*, que facilita a composição de regras, controle de geração de código e definição de pontos de extensão. Spark SQL utiliza a *engine* Spark de forma nativa. Entretanto, as otimizações foram capazes de gerar ganhos significativos em comparação com o próprio Spark, como pode ser visto no experimento de Armbrust et al. [2015].

O experimento de Gopalani and Arora [2015] é um bom exemplo do desempenho de algoritmos executados em Spark em comparação com outras soluções. O Apache Mahout é uma ferramenta otimizada para execução de algoritmos de aprendizado de máquina. As tabelas III.1 e III.2 apresentam, respectivamente, os resultados da execução do algoritmo K-Means, no mesmo ambiente e com a mesma quantidade de dados, utilizando Spark e Mahout.

Tabela III.1: Spark (pacote MLib). Fonte: adaptado de Gopalani and Arora [2015].

Tamanho dos dados	Número de *cores*	Tempo (s)
62MB	1	18
1240MB	1	149
1240MB	2	85

Tabela III.2: Apache Mahout. Fonte: adaptado de Gopalani and Arora [2015].

Tamanho dos dados	Número de *cores*	Tempo (s)
62MB	1	44
1240MB	1	291
1240MB	2	163

Como exemplo de utilização de Spark para tarefas de imputação, Qu et al. [2016] apresenta uma modificação no algoritmo Apriori para imputação baseada em regras de associação. O algoritmo Apriori original é aplicado a dados sob uma abordagem boleana e não se aplica a dados multidimensionais. Logo os dados foram convertidos para o formato binário-simétrico, para possibilitar a utilização de regras de associação.

Capítulo IV Proposta

Nesse capítulo será detalhada a arquitetura do sistema proposto, seus objetivos, características, funcionalidades e aplicações futuras.

IV.1 Appraisal 2.0

O sistema Appraisal foi inicialmente implementado por Soares [2007], para materialização da técnica de imputação composta. Todas as abstrações da técnica, detalhada na sessão III.5, como a implementação de planos de imputação, a medição da qualidade do dado imputado, entre outras, estão refletidas no sistema Appraisal. É composto basicamente por quatro módulos, ilustrados na figura IV.1:

1) O módulo *Crowner*, que executa os planos de imputação;

2) O módulo *Committee*, que representa o comitê de complementação de dados ausentes. Gera os valores de imputação, baseados nos demais valores sugeridos pelas estratégias e em outros atributos do conjunto de dados;

3) O módulo *Reviewer*, que verifica a qualidade das sugestões de imputação produzidas pelos módulos *Crowner* e *Committee*;

4) O módulo *Eraser*, que simula valores ausentes em base de dados, seguindo o mecanismo e percentual de ausência definidos pelo usuário.

O sistema Appraisal foi desenvolvido em Java, com implementações dos algoritmos K-Means, K-NN, redes neurais Back Propagation, PCA e AVG. Os quatro módulos descritos anteriormente foram organizados em *beans* utilizando o *framework Spring*. Para implementação das redes BP foi utilizado o *framework JOONE (Java Object Oriented Neural Engine)*.

O banco de dados utilizado foi o MySQL, na versão *portable*. Os experimentos realizados no Appraisal por Soares [2007] envolveram as bases de dados *iris plants*, *pima indians* e *breast cancer*, bastante conhecidas na literatura para experimentos de KDD.

Este trabalho propõe o Appraisal 2.0, um sistema de workflow científico capaz de permitir a experimentação de diversas técnicas de imputação, norteadas pelo modelo de imputação composta, para bases de dados de natureza *big data*. Os requisitos do sistema serão detalhados na próxima sessão.

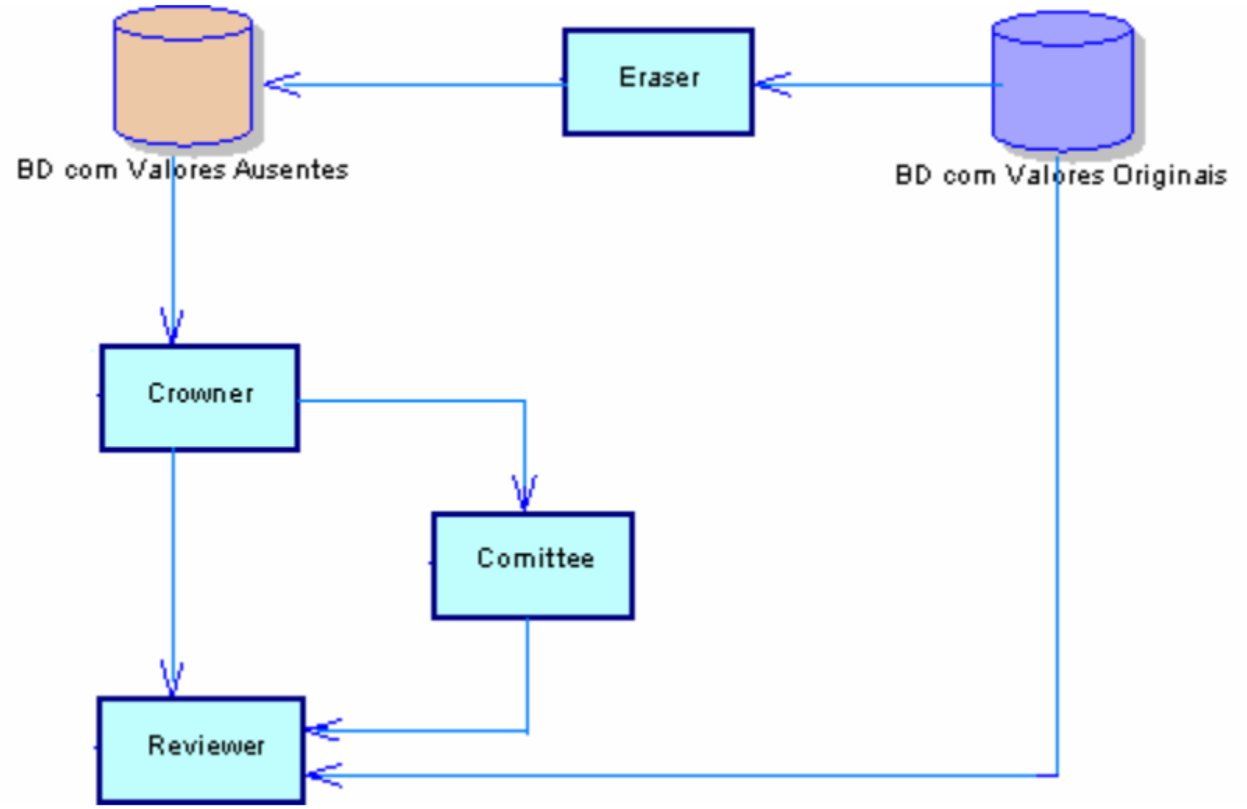

Figura IV.1: Módulos do sistema Appraisal. Fonte: adaptado de Soares [2007].

IV.1.1 Requisitos do sistema

O sistema Appraisal 2.0 possui os seguintes requisitos:

A) Requisitos não-funcionais

- Suportar grandes volumes de dado, *Big Data*.
- Alto desempenho, com processamento de dados paralelo e distribuído.
- Deve garantir a proveniência e reprodutibilidade dos experimentos realizados.

B) Requisitos funcionais:

- Executar planos de imputação seguindo o modelo de imputação composta.
- Possibilitar a composição de planos de imputação através de ferramenta gráfica.
- Possibilitar a utilização de algoritmos de KDD, nativos da solução, nas diversas tarefas dos planos de imputação.
- Possibilitar a utilização de algoritmos personalizados nas diversas tarefas dos planos de imputação.
- Possuir um módulo web para execução e acompanhamento dos experimentos realizados.
- Possibilitar a integração com outros sistemas através da utilização de micro-serviços.

IV.1.2 Arquitetura

A arquitetura do Appraisal 2.0 utilizará a estrutura em módulos da primeira versão do sistema, cerne do modelo de imputação composta. Apenas o módulo *Committee* não será utilizado, pois apresentou comportamento irregular nos experimentos realizados por Soares [2007].

O módulo *Eraser* será uma funcionalidade da interface web do sistema, permitindo a geração de ausências seguindo os mecanismos MCAR, MAR e NMAR. A figura IV.2 ilustra o funcionamento do módulo:

Figura IV.2: Módulo Eraser.

O módulo *Crowner* gerenciará a execução dos planos de imputação. A figura IV.3 denota as atividades do mesmo.

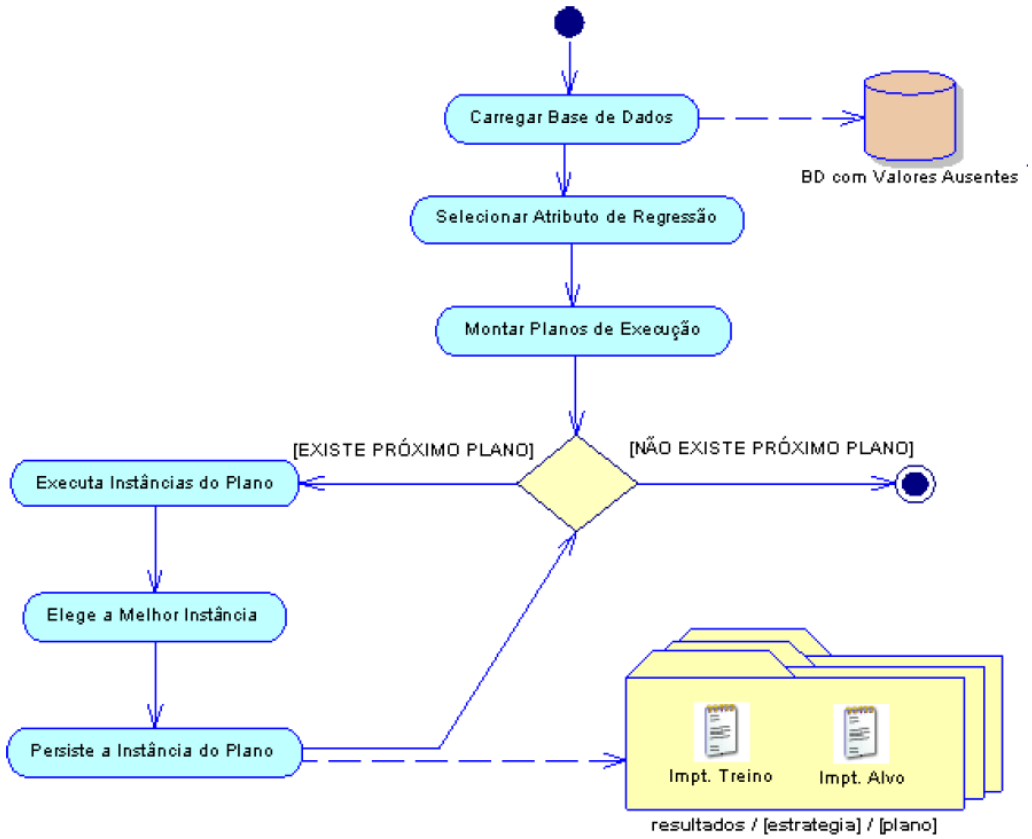

Figura IV.3: Diagrama de atividades do módulo Crowner. Fonte: adaptado de Soares [2007].

O módulo *Reviewer* avaliará os resultados das imputações sugeridas pelo módulo *Crowner*. A figura IV.4 ilustra as atividades do mesmo.

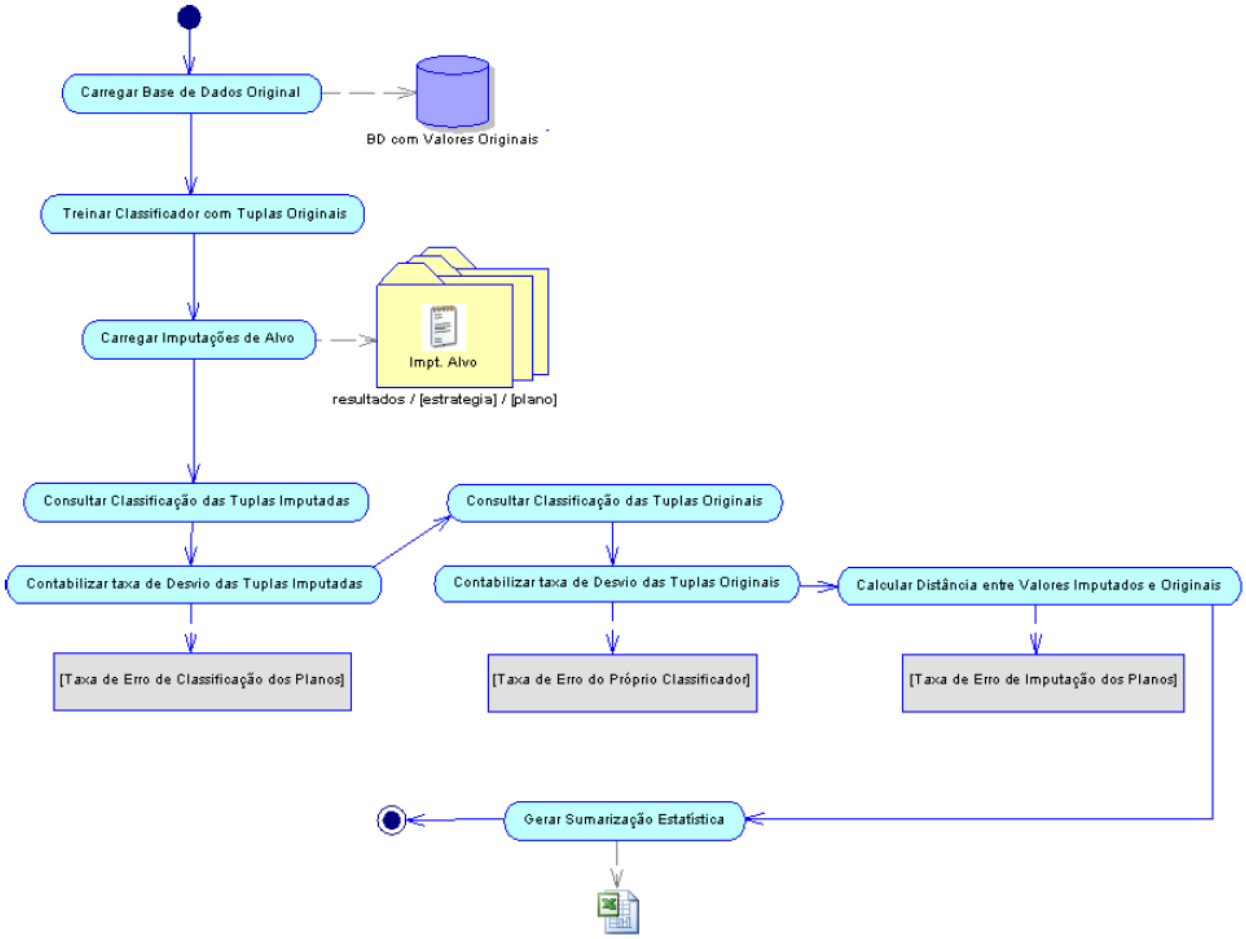

Figura IV.4: Diagrama de atividades do módulo Reviwer. Fonte: adaptado de Soares [2007].

IV.1.3 Interface Web

A interface Web será responsável por gerenciar as execuções dos experimentos e pelo acompanhamento dos resultados parciais e finais. Proverá as seguintes funcionalidades:

- Executar experimentos e, consequentemente, os planos de imputação configurados para o mesmo.
- Exibir a representação gráfica do experimento, com sua atual completude, resultados parciais e finais.
- Possibilitar a implantação de algoritmos personalizados, para que possam ser utilizados em qualquer tarefa do plano de imputação.

IV.1.4 Módulo de Composição de Planos de Imputação

Apesar de ser um sistema de workflow científico, com abordagem *data centric*, a capacidade de modelar e compor planos de imputação, sem que o cientista de dados conheça linguagens de programação, é um requisito do sistema. Dessa forma, a organização por Business Process Management (BPM) será utilizada. A interface será denominada Módulo de Composição de Planos de Imputação (MCPI).

O sistema possibilitará inicialmente a utilização dos cinco planos de imputação, e suas respectivas tarefas e algoritmos, propostos por Soares [2007] no modelo de imputação composta. São eles:

- Seleção ⇒ Agrupamento ⇒ Regressão

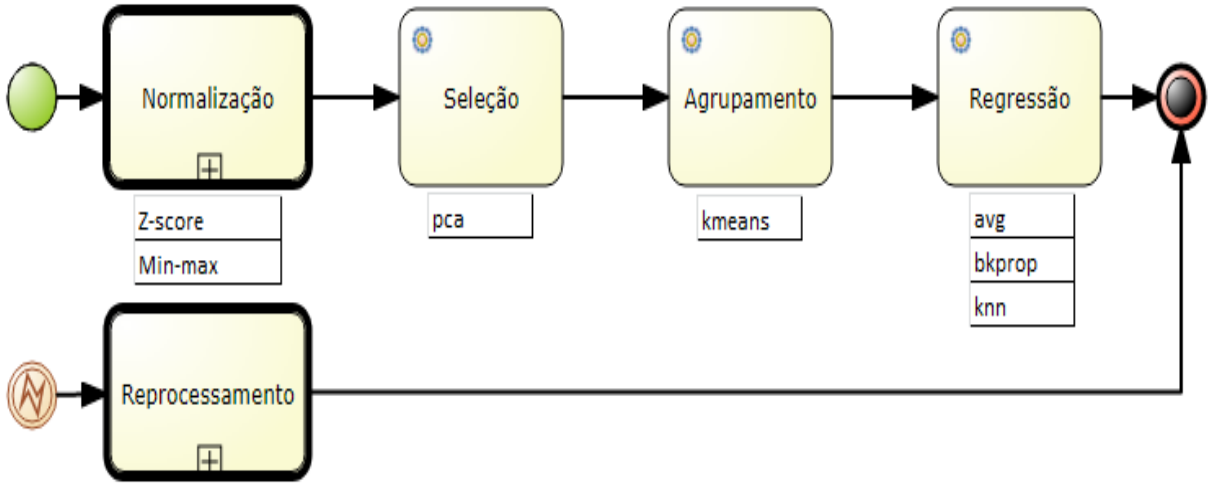

Figura IV.5: BPM Seleção / Agrupamento / Regressão.

- Agrupamento ⇒ Seleção ⇒ Regressão

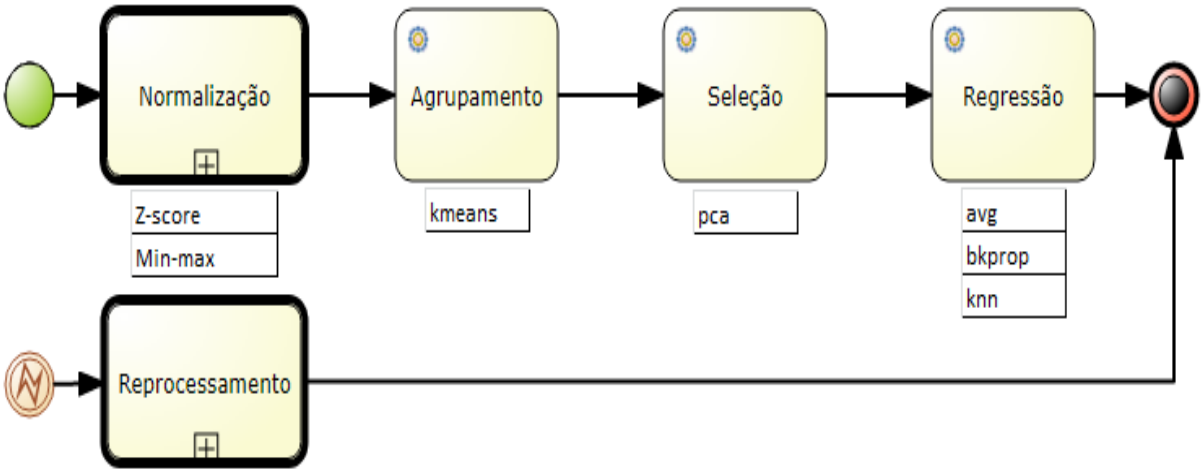

Figura IV.6: BPM Agrupamento / Seleção / Regressão.

- Agrupamento ⇒ Regressão

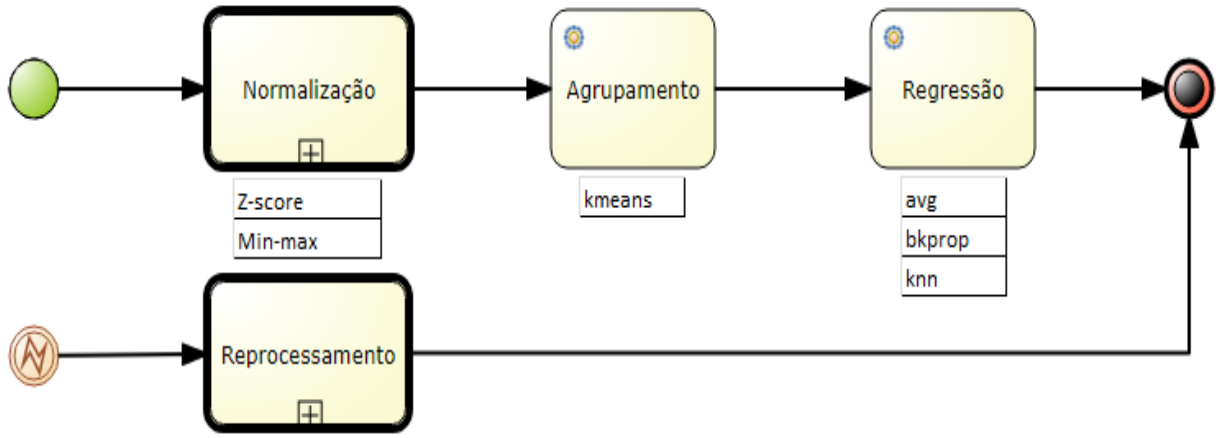

Figura IV.7: BPM Agrupamento / Regressão.

- Seleção ⇒ Regressão

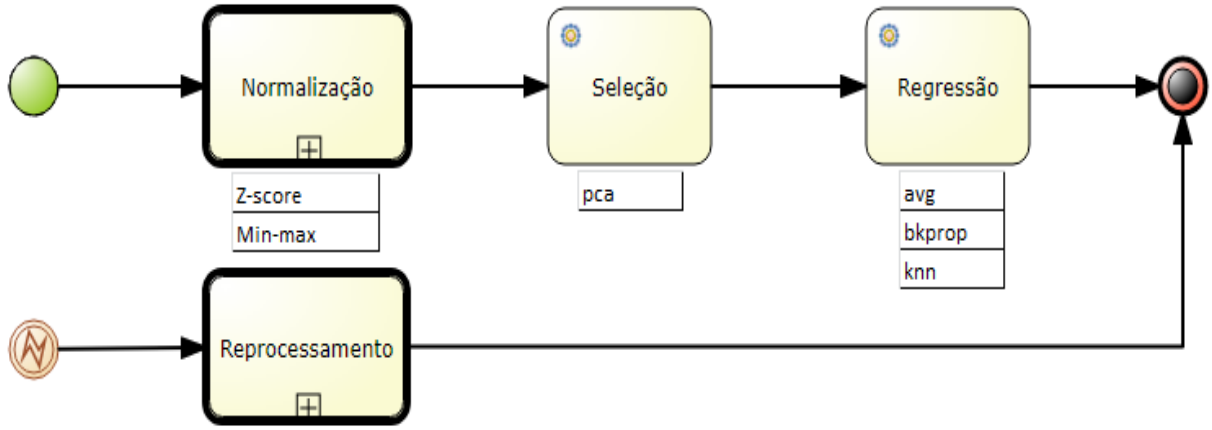

Figura IV.8: BPM Seleção / Regressão.

- Regressão

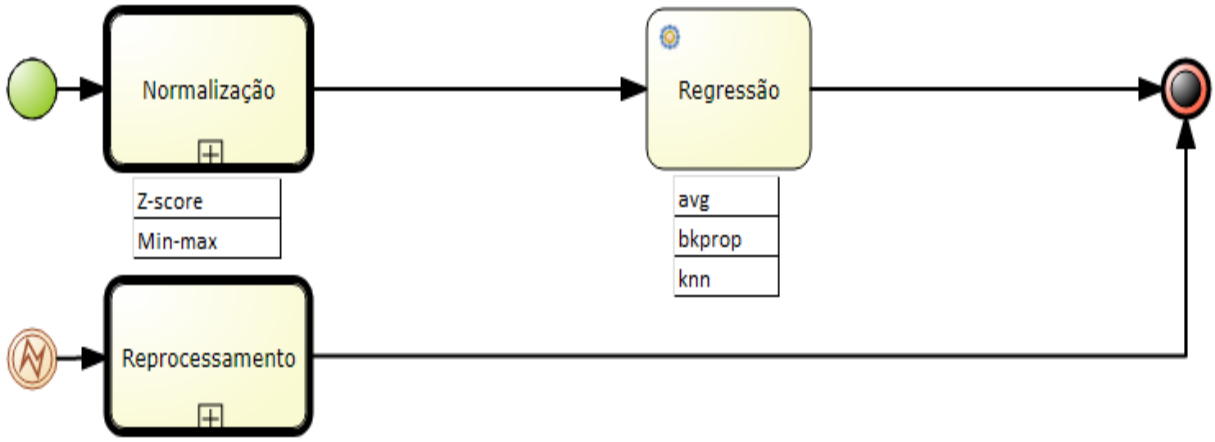

Figura IV.9: BPM Regressão.

Conforme mencionado anteriormente, apesar de disponibilizar cinco planos de imputação iniciais, o cientista de dados poderá modelar novos planos de acordo com a necessidade dos experimentos.

O sistema possibilitará nativamente a utilização dos algoritmos do pacote MLlib (Spark). Dessa forma, algoritmos do pacote poderão ser utilizados ao longo das tarefas de imputação. Será possível também utilizar algoritmos personalizados, implantando-os na unidade Spark de forma que também estejam disponíveis para utilização. Essas funcionalidades são explicadas com maior detalhe na sessão IV.1.6.

IV.1.5 Pool de Serviços

Para favorecer o baixo acoplamento e maximizar a reutilização de componentes, as funcionalidades do sistema estarão dispostas internamente sob a forma de *Enterprise JavaBeans* (EJBs) e externamente sob a forma de *Web services*. *Engines* BPM ofercem a capacidade de acessar *Web services* através de tarefas do tipo *Web Service task*.

Os algoritmos implementados em Apache Spark também serão disponibilizados sob a forma de *Web services*. O modelo e estrutura desses algoritmos serão explicados na próxima sessão. A figura IV.10 mostra a interação entre os componentes na arquitetura.

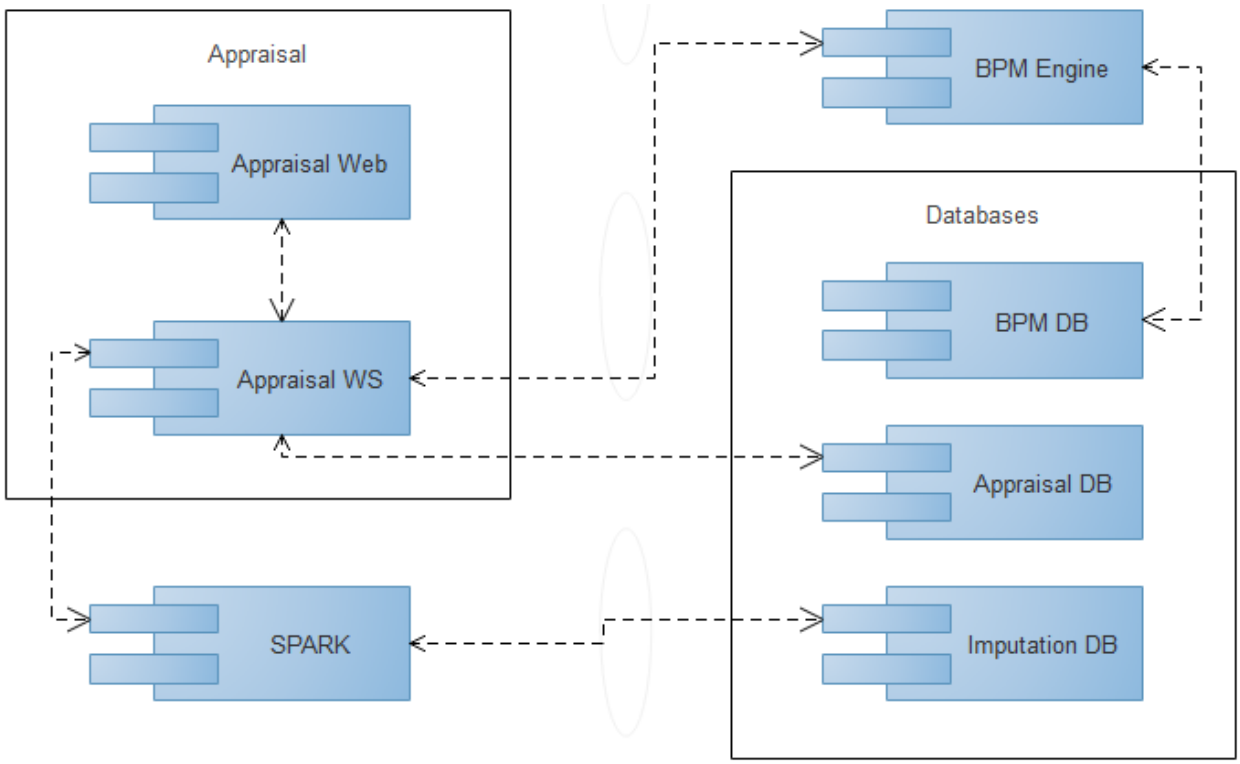

Figura IV.10: Diagrama de componentes: Appraisal, BPM engine, Serviços e Spark.

Toda a comunicação entre os componentes *Web service* serão realizadas por meio do protocolo *Hyper Text Transfer Protocol Secure (HTTPS)*, por motivos de segurança. Pelo mesmo

motivo, a comunicação EJB utilizará o protocolo *Secure Socket Layer (SSL)*.

IV.1.6 Algoritmos Spark

Os algoritmos utilizados nos planos de imputação serão implementados em Spark na linguagem Scala. Serão integrados nativamente ao pacote MLlib, que oferece algoritmos Spark para *machine learning (ML)* [Zaharia et al., 2016]. O pacote possui mais de 50 algoritmos para treinamento de modelos distribuídos. Tem como objetivo tornar escalável e fácil a utilização de algoritmos de ML.

Possui ferramentas como [Meng et al., 2016]: algoritmos de ML (classificação, regressão, agrupamento e filtragem colaborativa), *featurization* (extração de características, transformação, redução de dimensionalidade e seleção), *pipelines* (construção, avaliação e *tuning* de *pipelines*), persistência (salvar e carregar algoritmos, modelos e *pipelines*) e utilidades (álgebra linear, estatística, gerenciamento de dados, etc).

Os algoritmos serão executados através da API de *reflection* da linguagem Scala [Miller et al., 2016]. Serão criadas *interfaces* do sistema Appraisal para os algoritmos de KDD utilizados nas tarefas dos planos de imputação. Dessa forma, os algoritmos serão instanciados e executados por reflexão. Por exemplo, os algoritmos utilizados no modelo de imputação composta utilizarão as interfaces descritas no diagrama de classes IV.11

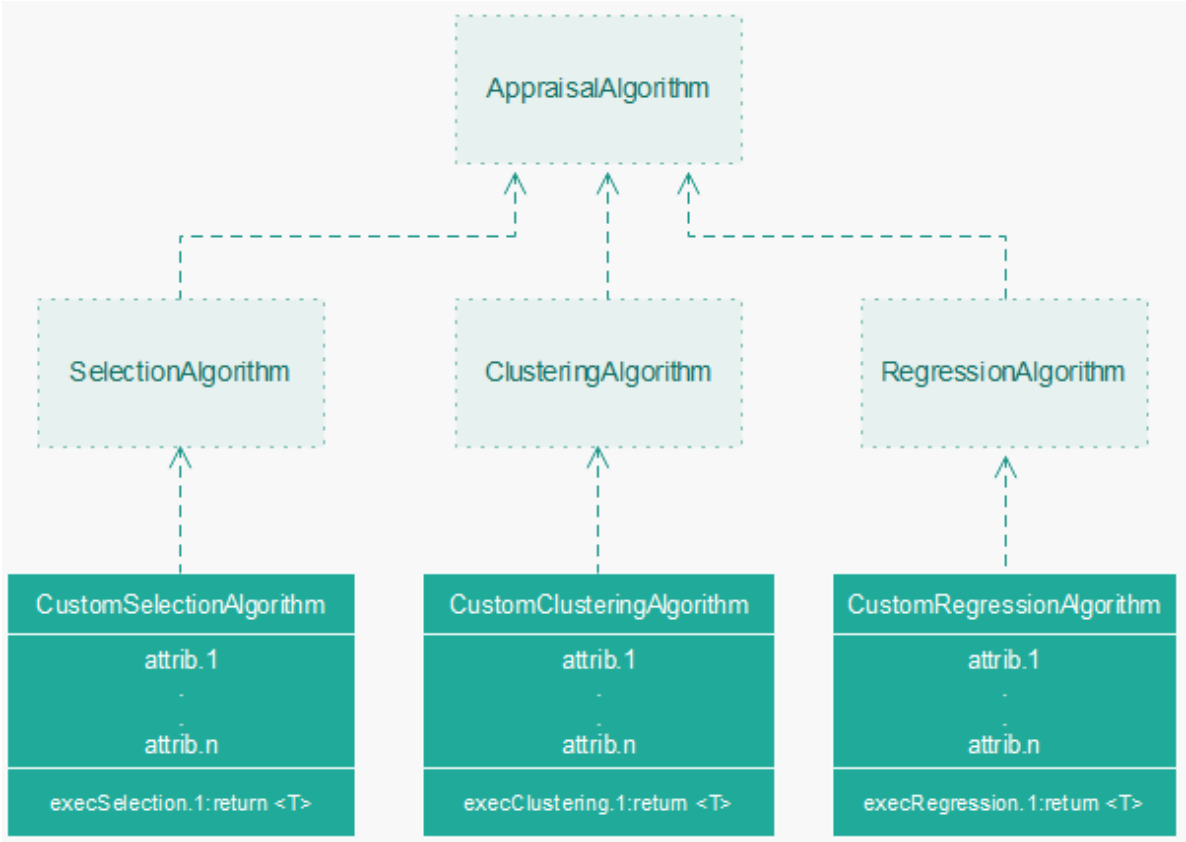

Figura IV.11: Diagrama de classes: interfaces para algoritmos personalizados.

Algoritmos personalizados também poderão ser utilizados, desde que implementem as interfaces descritas e sigam o modelo citado. Uma funcionalidade de implantação de códigos personalizados será disponibilizada na interface Web. Essa funcionalidade utilizará a ferramenta *Simple Build Tool (SBT)* para implantar algoritmos no ambiente Spark. Dessa forma, algoritmos personalizados serão utilizados por reflexão da mesma forma que os algoritmos nativos.

IV.1.7 Infraestrutura

Inicialmente o projeto conta com quatro servidores no laboratório de pesquisa, divididos nas camadas de Aplicação, Bancos de Dados e Spark. No servidor 1 será instalado um servidor de aplicação, onde serão implantados o Appraisal 2.0 (Web + Serviços WS) e a *engine* BPM. Os bancos de dados da *engine* BPM, do Appraisal e as bases de imputação serão criadas no servidor 2. Os servidores 3 e 4 irão compor o *cluster* Spark. O diagrama de implantação IV.12 detalha a utilização dos servidores.

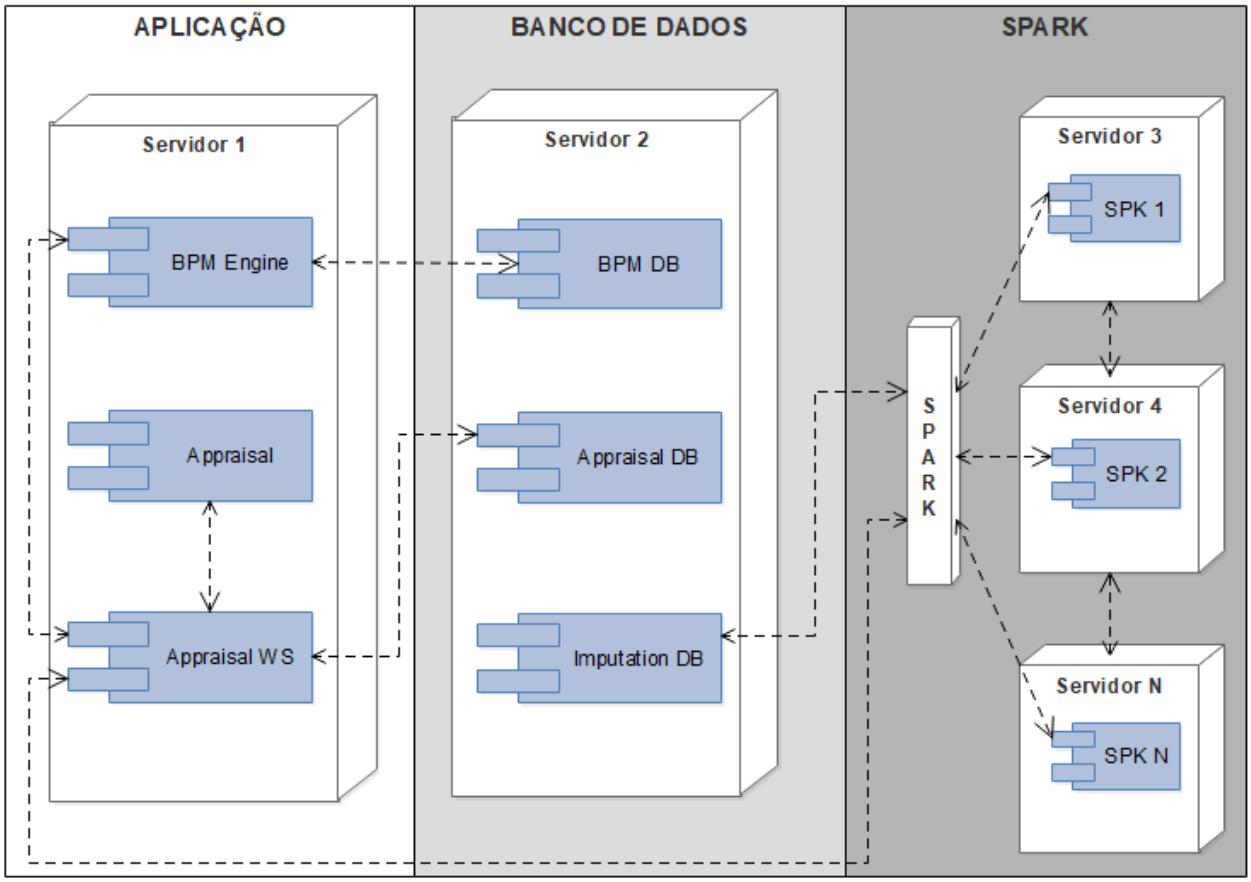

Figura IV.12: Diagrama de Implantação.

Outros servidores podem ser adicionados na camada Spark para otimização de desempenho.

IV.2 Resultados Obtidos

Foi criada uma nova versão do sistema, o Appraisal 1.1. Nessa versão, a camada de replicação de bases de dados foi reimplementada. Além disso, foi substituída a versão *portable* do banco de dados MySql pela versão 5.7. Posteriormente, foram reproduzidos os experimentos de imputação composta realizados por Soares [2007].

Os planos de imputação foram executados nas bases de dados *iris plants*, *pima indians* e *breast cancer*. Os resultado dos experimentos foram consolidados, com exceção da base de dados *breast cancer*, que apresentou valores anormais para o algoritmo *back propagation*. Provavelmente esse valores ocorreram devido a configuração de hiper-parâmetros da rede. Esse experimento será repetido em momento oportuno.

Foram simuladas ausências de dados de dez, vinte, trinta, quarenta e cinquenta por cento para cada um dos atributos das bases de dados. Foi calculado o erro médio dos planos de imputação para cada percentual de ausência.

A tabela IV.13 apresenta os resultados para a base *iris plants*:

Iris plants						
Plano de Imputação	**Algoritmos**	**Erro médio (%)**				
		10%	**20%**	**30%**	**40%**	**50%**
agrupamento.regressão	kmeans.avg	7,28	8,13	8,07	8,33	8,77
	kmeans.bkprop	6,03	6,72	6,74	7,31	7,02
	kmeans.knn	4,87	7,18	6,00	6,84	6,91
agrupamento.seleção.regressão	kmeans.pca.avg	7,28	8,13	8,07	8,33	8,77
	kmeans.pca.bkprop	5,98	6,71	6,73	7,29	7,02
	kmeans.pca.knn	4,87	7,19	6,00	6,84	6,91
regressão	avg	29,45	28,72	27,47	31,23	29,27
	bkprop	6,93	8,16	7,47	7,95	7,71
	knn	4,88	7,65	6,21	7,15	6,96
seleção.agrupamento.regressão	pca.kmeans.avg	10,03	8,72	8,13	8,40	8,81
	pca.kmeans.bkprop	6,62	6,79	6,76	7,28	7,00
	pca.kmeans.knn	5,76	7,26	6,03	6,84	6,93
seleção.regressão	pca.bkprop	6,34	7,74	7,55	8,20	7,71
	pca.knn	4,88	7,66	6,21	7,15	6,97

Figura IV.13: Erro médio dos planos de imputação na base *iris plants*.

O gráfico IV.14 compara o desempenho dos planos de imputação na base *iris plants*:

A tabela IV.15 apresenta o erro médio dos planos de imputação na base *pima indians*:

O gráfico IV.16 compara o desempenho dos planos de imputação na base *pima indians*:

O plano de imputação que compreende apenas a tarefa de regressão com média simples foi retirado das análises de desempenho, pois apresentou erro elevado que distorcia a análise

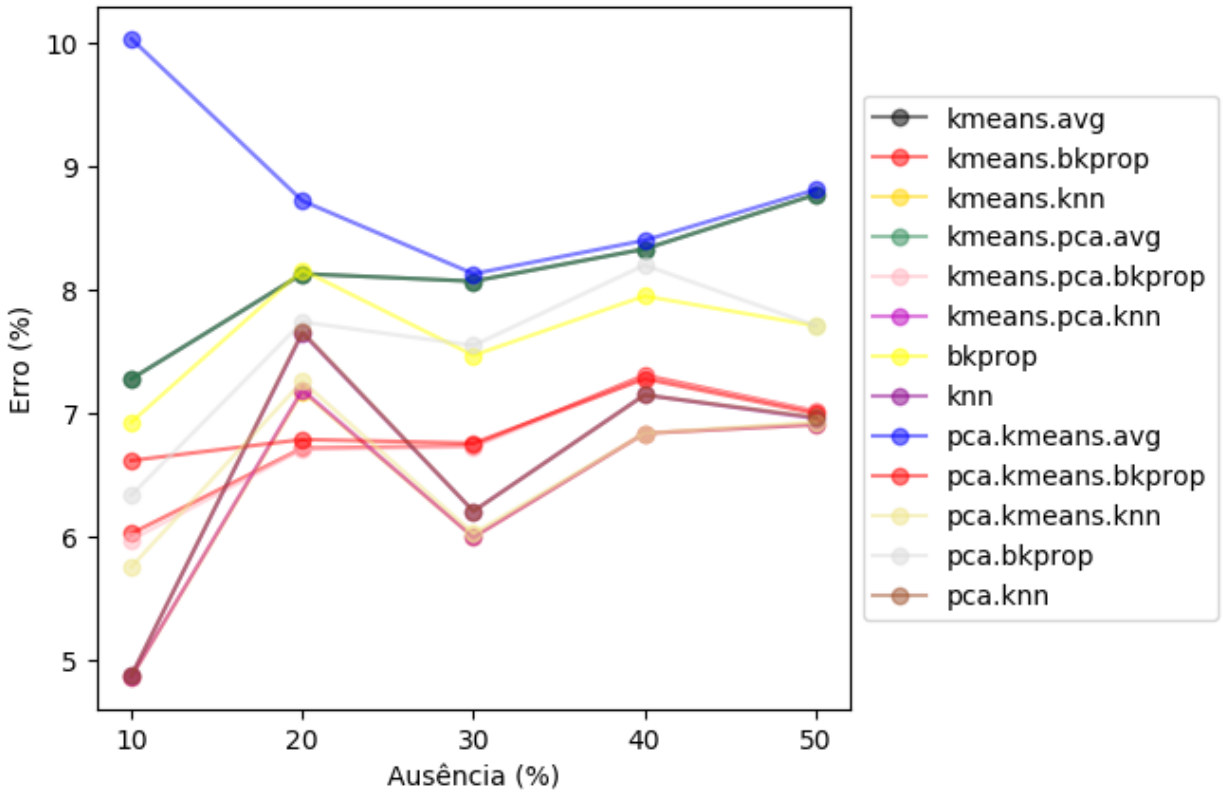

Figura IV.14: Desempenho dos planos de imputação na base *iris plants*.

Pima indians						
Plano de Imputação	**Algoritmos**	**Erro médio (%)**				
		10%	**20%**	**30%**	**40%**	**50%**
agrupamento.regressão	kmeans.avg	35,13	39,14	40,28	38,68	39,42
	kmeans.bkprop	35,59	39,24	40,32	38,88	39,26
	kmeans.knn	31,94	37,08	36,84	36,70	36,53
agrupamento.seleção.regressão	kmeans.pca.avg	35,13	39,14	40,28	38,68	39,42
	kmeans.pca.bkprop	35,53	39,22	40,29	38,83	39,23
	kmeans.pca.knn	31,94	37,08	36,84	36,70	36,53
regressão	avg	47,91	48,70	48,12	46,76	46,89
	bkprop	39,76	41,79	42,07	41,14	41,74
	knn	33,48	38,08	37,60	37,06	37,03
seleção.agrupamento.regressão	pca.kmeans.avg	38,84	40,66	40,62	39,21	39,55
	pca.kmeans.bkprop	38,22	40,20	40,42	39,28	39,31
	pca.kmeans.knn	32,69	37,29	36,84	36,74	36,53
seleção.regressão	pca.bkprop	39,36	41,35	41,89	40,81	41,46
	pca.knn	33,48	38,08	37,60	37,06	37,03

Figura IV.15: Erro médio dos planos de imputação na base *pima indians*.

gráfica.

Conclusões sobre os resultados na base *iris plants*:

- O plano de imputação que utiliza a tarefa de agrupamento com K-Means e regressão

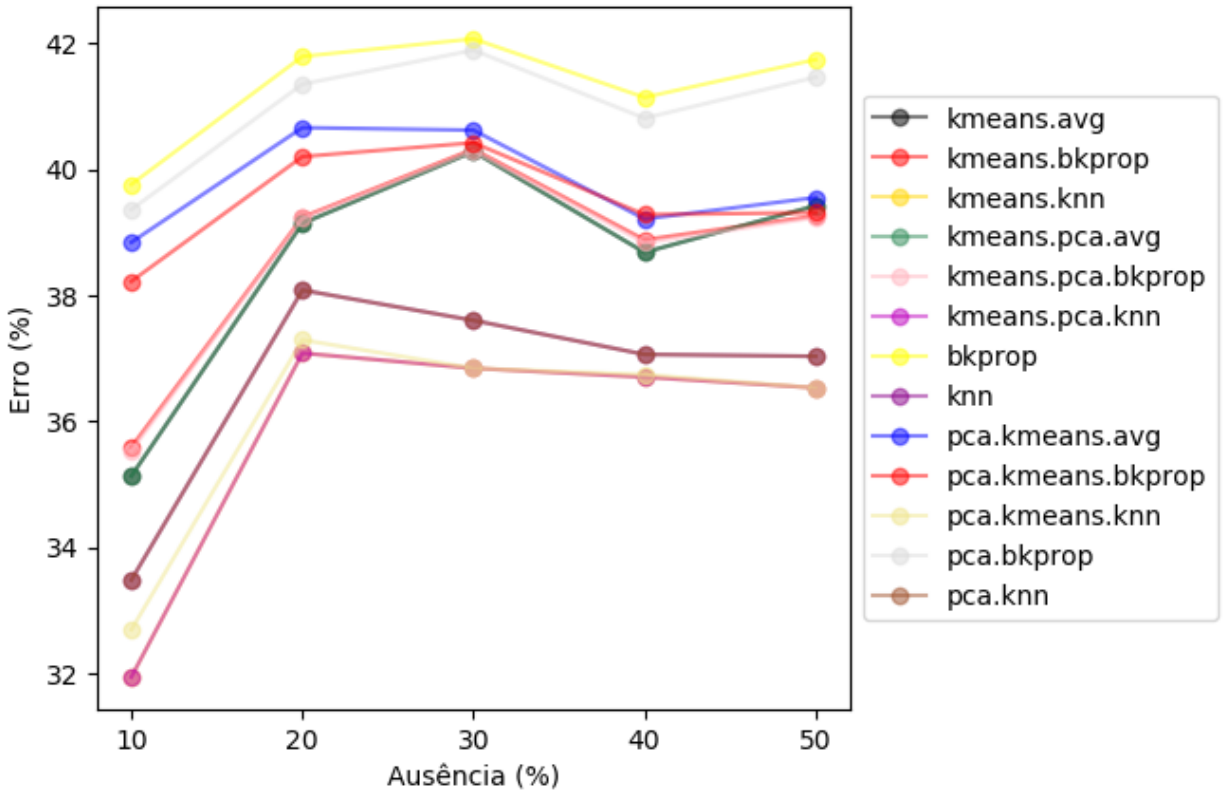

Figura IV.16: Desempenho dos planos de imputação na base *pima indians*.

com K-NN foi o que apresentou o melhor resultado.

• É possível observar que o plano de imputação com a tarefa de regressão por média simples, precedida da tarefa de agrupamento com K-Means, apresentou os mesmos resultados do plano que utiliza, além destas, a tarefa de seleção com PCA.

• O plano de imputação com a tarefa de regressão com K-NN, precedido da tarefa de agrupamento com K-Means, obteve os mesmos resultados do plano que utiliza, além destas, a tarefa de seleção com PCA.

• O plano de imputação que utiliza apenas a tarefa de regressão com K-NN apresentou os mesmos resultados do plano que utiliza, além desta, a tarefa de seleção com PCA.

Portanto, é possível concluir que a tarefa de seleção com PCA não foi relevante na base *iris plants*, muito provavelmente pela correlação baixa, ou talvez até nula, entre os atributos da mesma.

Conclusões sobre os resultados na base *pima indians*:

• O plano de imputação que utiliza a tarefa de agrupamento com K-Means e regressão com K-NN foi novamente o que apresentou o melhor resultado.

• A tarefa de seleção com PCA teve o mesmo comportamento nos planos citados nas conclusões da base *iris plants*.

IV.3 Cronograma

A tabela IV.17 apresenta o cronograma com o planejamento das atividades de construção do sistema Appraisal 2.0, com a escrita do artigo científico e a elaboração da dissertação sobre o tema.

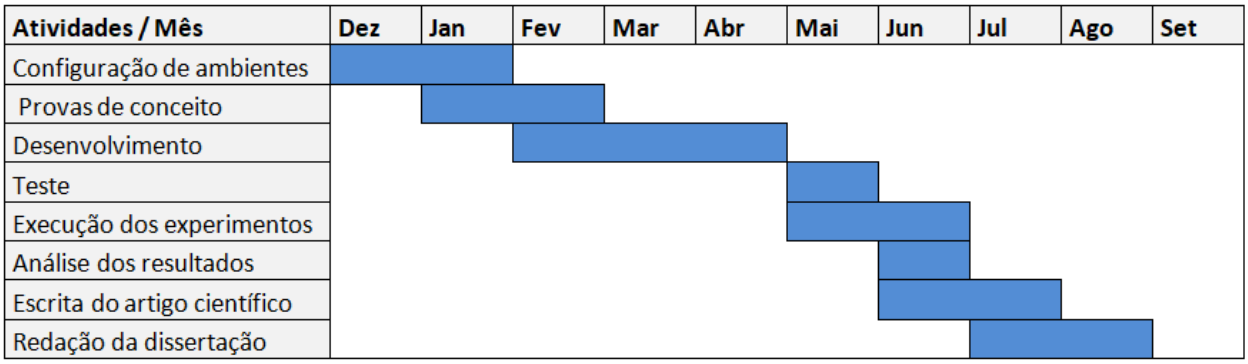

Figura IV.17: Crograma de atividades.

Capítulo V Conclusão

Com base nos resultados promissores apresentados por Soares [2007] na experimentação da técnica de imputação composta, e na reprodução dos mesmos com o Appraisal 1.1, é possível imaginar que em ambientes *big data* novos conhecimentos sejam descobertos acerca do tema imputação. Em grandes volumes de dados, flutuações individuais são ignoradas dada a magnitude do número de exemplos de treinamento [Jagadish et al., 2014]. Portanto, algoritmos de aprendizado de máquina podem apresentar resultados melhores que os atuais.

A capacidade de modelar diferentes planos de imputação, além dos cinco propostos por Soares [2007], pode permitir que novas técnicas de imputação sejam descobertas.

Após vasta pesquisa na literatura, foi encontrado somente um experimento com foco em imputação para ambientes *big data*, o de Qu et al. [2016], que utiliza uma versão personalizada do algoritmo Apriori em Spark.

Ainda que o foco do sistema seja imputação, a arquitetura proposta permite modelar tarefas de KDD que não tenham necessariamente esse objetivo, usufruindo da capacidade de processamento paralelo e distribuído do sistema.

O Appraisal 2.0 é uma ferramenta capaz de impulsionar a pesquisa científica na área de imputação, além de possibilitar a execução de tarefas de KDD gerenciadas por um sistema de *workflow*.

Referências Bibliográficas

Agrawal, R., Imieliński, T., and Swami, A. (1993). Mining association rules between sets of items in large databases. *SIGMOD Rec.*, 22(2):207–216.

Allison, P. (1999). Multiple imputation for missing data: A cautionary tale. 28.

Armbrust, M., Xin, R. S., Lian, C., Huai, Y., Liu, D., Bradley, J. K., Meng, X., Kaftan, T., Franklin, M. J., Ghodsi, A., and Zaharia, M. (2015). Spark sql: Relational data processing in spark. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, SIGMOD '15, pages 1383–1394, New York, NY, USA. ACM.

Atzori, L., Iera, A., and Morabito, G. (2010). The internet of things: A survey. *Computer Networks*, 54(15):2787 – 2805.

Austin, P. C. and Escobar, M. D. (2005). Bayesian modeling of missing data in clinical research. *Computational Statistics Data Analysis*, 49(3):821 – 836.

Bansal, A., Kauffman, R. J., and Weitz, R. R. (1993). Comparing the modeling performance of regression and neural networks as data quality varies: A business value approach. *Journal of Management Information Systems*, 10(1):11–32.

Batista, G. (2003). *Pré-Processamento de Dados em Aprendizado de Máquina Supervisionado*. PhD thesis, USP.

Batista, G. and Monard, M.-C. (2001). A study of k-nearest neighbour as a model-based method to treat missing data.

Batista, G. and Monard, M.-C. (2003a). An analysis of four missing data treatment methods for supervised learning. 17.

Batista, G. and Monard, M.-C. (2003b). Um estudo sobre a efetividade do método de imputação baseado no algoritmo k-vizinhos mais próximos. In *Proceedings of IV Workshop on Advances Trends in AI for Problem Solving*.

Brown, M. L. and Kros, J. F. (2003). Data mining. chapter The Impact of Missing Data on Data Mining, pages 174–198. IGI Global, Hershey, PA, USA.

C Yuan, Y. (2005). Multiple imputation for missing data: Concepts and new development.

Chen, M., Mao, S., and Liu, Y. (2014). Big data: A survey. *Mobile Networks and Applications*, 19(2):171–209.

Davidson, S. B. and Freire, J. (2008). Provenance and scientific workflows: Challenges and opportunities. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1345–1350, New York, NY, USA. ACM.

Deelman, E., Gannon, D., Shields, M., and Taylor, I. (2009). Workflows and e-science: An overview of workflow system features and capabilities. *Future Generation Computer Systems*, 25(5):528 – 540.

Elgamal, T., Yabandeh, M., Aboulnaga, A., Mustafa, W., and Hefeeda, M. (2015). spca: Scalable principal component analysis for big data on distributed platforms. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, SIGMOD '15, pages 79–91, New York, NY, USA. ACM.

Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). The kdd process for extracting useful knowledge from volumes of data. *Commun. ACM*, 39(11):27–34.

Freire, J., Koop, D., Santos, E., and Silva, C. T. (2008). Provenance for computational tasks: A survey. *Computing in Science Engineering*, 10(3):11–21.

García-Laencina, P. J., Sancho-Gómez, J.-L., Figueiras-Vidal, A. R., and Verleysen, M. (2009). K nearest neighbours with mutual information for simultaneous classification and missing data imputation. *Neurocomputing*, 72(7):1483 – 1493. Advances in Machine Learning and Computational Intelligence.

Gelman, A. and Hill, J. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Analytical Methods for Social Research. Cambridge University Press.

Goldschmidt, R., Passos, E., and Bezerra, E. (2015). *Data Mining: Conceitos, técnicas, algoritmos, orientações e aplicações*.

Gopalani, S. and Arora, R. (2015). Comparing Apache Spark and Map Reduce with Performance Analysis using K-Means. *International Journal of Computer Applications*, 113(1):8–11.

Han, J., Kamber, M., and Pei, J. (2011). *Data Mining: Concepts and Techniques, Third Edition*. Morgan Kaufmann, Waltham, Mass., 3 edition edition. 00000.

Hand, D. J., Smyth, P., and Mannila, H. (2001). *Principles of Data Mining*. MIT Press, Cambridge, MA, USA.

Hruschka, E. R., Hruschka, E. R., and Ebecken, N. F. F. (2005). *Missing Values Imputation for a Clustering Genetic Algorithm*, pages 245–254. Springer Berlin Heidelberg, Berlin, Heidelberg.

Huang, C.-C. and Lee, H.-M. (2004). A grey-based nearest neighbor approach for missing attribute value prediction. *Applied Intelligence*, 20(3):239–252.

Jagadish, H. V., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J. M., Ramakrishnan, R., and Shahabi, C. (2014). Big data and its technical challenges. *Commun. ACM*, 57(7):86–94.

Jerez, J. M., Molina, I., García-Laencina, P. J., Alba, E., Ribelles, N., Martín, M., and Franco, L. (2010). Missing data imputation using statistical and machine learning methods in a real breast cancer problem. *Artif. Intell. Med.*, 50(2):105–115.

Jonsson, P. and Wohlin, C. (2004). An evaluation of k-nearest neighbour imputation using likert data. In *10th International Symposium on Software Metrics, 2004. Proceedings.*, pages 108–118.

Khan, N., Yaqoob, I., Hashem, I., Inayat, Z., Kamaleldin, W., Alam, M., Shiraz, M., and Gani, A. (2014). Big data: Survey, technologies, opportunities, and challenges. 2014:18.

Li, D., Deogun, J., Spaulding, W., and Shuart, B. (2004). *Towards Missing Data Imputation: A Study of Fuzzy K-means Clustering Method*, pages 573–579. Springer Berlin Heidelberg, Berlin, Heidelberg.

Little, R. J. (2015). *Missing Data/Imputation*. John Wiley Sons, Inc.

Little, R. J. A. and Rubin, D. B. (1986). *Statistical Analysis with Missing Data*. John Wiley & Sons, Inc., New York, NY, USA.

Liu, Y., Xu, L., and Li, M. (2017). The parallelization of back propagation neural network in mapreduce and spark. *International Journal of Parallel Programming*, 45(4):760–779.

M. Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). From data mining to knowledge discovery in databases. 17:37–54.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif. University of California Press.

Magnani, M. (2004). Techniques for dealing with missing data in knowledge discovery tasks.

Maillo, J., Ramírez, S., Triguero, I., and Herrera, F. (2017). knn-is: An iterative spark-based design of the k-nearest neighbors classifier for big data. *Knowledge-Based Systems*, 117(Supplement C):3 – 15. Volume, Variety and Velocity in Data Science.

McPhillips, T., Bowers, S., Zinn, D., and Ludäscher, B. (2009). Scientific workflow design for mere mortals. *Future Generation Computer Systems*, 25(5):541 – 551.

Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., Freeman, J., Tsai, D., Amde, M., Owen, S., Xin, D., Xin, R., Franklin, M. J., Zadeh, R., Zaharia, M., and Talwalkar, A. (2016). Mllib: Machine learning in apache spark. *J. Mach. Learn. Res.*, 17(1):1235–1241.

Miller, H., Burmako, E., and Haller, P. (2016). *Reflection overview*. scala-lang, The address of the publisher.

Qu, Z., Yan, J., and Yin, S. (2016). Association-rules-based data imputation with spark. In *2016 4th International Conference on Cloud Computing and Intelligence Systems (CCIS)*, pages 145–149.

Raghunathan, T. E. (2004). What do we do with missing data? some options for analysis of incomplete data. *Annual Review of Public Health*, 25(1):99–117. PMID: 15015914.

Rubin, D. B. (1988). An overview of multiple imputation. In *In Proceedings of the Survey Research Section, American Statistical Association*, pages 79–84.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. chapter Learning Internal Representations by Error Propagation, pages 318–362. MIT Press, Cambridge, MA, USA.

Smith, L. I. (2005). A Tutorial on Principal Component Analysis.

Soares, J. (2007). *Pré-processamento em Mineração de Dados: um Estudo Comparativo em Complementação*. PhD thesis, COPPE/UFRJ.

Talavera-Llames, R. L., Pérez-Chacón, R., Martínez-Ballesteros, M., Troncoso, A., and Martínez-Álvarez, F. (2016). *A Nearest Neighbours-Based Algorithm for Big Time Series Data Forecasting*, pages 174–185. Springer International Publishing, Cham.

Wu, Z., Li, Y., Plaza, A., Li, J., Xiao, F., and Wei, Z. (2016). Parallel and distributed dimensionality reduction of hyperspectral data on cloud computing architectures. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(6):2270–2278.

Zaharia, M., Xin, R. S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M. J., Ghodsi, A., Gonzalez, J., Shenker, S., and Stoica, I. (2016). Apache spark: A unified engine for big data processing. *Commun. ACM*, 59(11):56–65.