Uma abordagem baseada em mineração de dados para apoio ao ciclo de vida de projetos de pesquisa e inovação

Louise Guimarães Pedroso, Kate Revoredo, Fernanda Baião

Departamento de Informática Aplicada – Universidade Federal do Estado do Rio de Janeiro (UNIRIO) – Rio de Janeiro, RJ – Brazil

{louise.pedroso, katerevoredo, fernanda.baiao}@uniriotec.br

Abstract. *Monitoring Science, Technology and Innovation projects require control of their lifecycle through reports or management and operational activities. The goal of this work is to build a classification model that automatically discovers the lifecycle stage of a project based on a set of characteristics that describes it. Data Mining was used to achieve this classification model. Moreover, analyses in the models suggest that they can also be used to identify problems in data or failures in the process.*

Resumo. *O acompanhamento de projetos de Ciência, Tecnologia e Inovação requer o controle do ciclo de vida desses, o que normalmente é feito através de relatórios ou em atividades gerenciais e operacionais. O objetivo deste trabalho é construir um classificador que automaticamente descubra o estágio do ciclo de vida de um projeto com base em um conjunto de características que o descrevem. Mineração de Dados foi utilizada para encontrar esse classificador. Análises com o classificador sugerem que ele também pode ser utilizado para identificar problemas nos dados ou mesmo falhas em processos.*

1. Introdução

Ciência e Tecnologia (C&T), no Brasil, é em grande parte desenvolvida por universidades públicas e por institutos de pesquisa. No setor privado, poucas são as empresas brasileiras que investem em desenvolvimento tecnológico, sendo que a maioria delas é pública. Esse panorama reflete a situação de financiamento à pesquisa no Brasil, que vem sendo custeada, preponderantemente, pelo setor público, ao contrário do que ocorre em países desenvolvidos, onde o setor privado representa uma parcela significativa do financiamento. No caso específico do apoio à inovação, onde as empresas são as beneficiárias, o investimento é feito tanto através de concessão de crédito a juros baixos, operados por instituições federais como FINEP (Financiadora de Estudos e Projetos) e BNDES, quanto através de subvenção econômica.

O acompanhamento de projetos desta natureza requer o controle do ciclo de vida dos projetos, seja em relatórios ou em atividades gerenciais e operacionais, de modo a permitir agrupamentos em uma mesma etapa e a facilitar as operações necessárias ao seu desenvolvimento. Essas atividades são particularmente importantes para a FINEP, tendo em vista que a mesma atua como a Agência Brasileira de Inovação.

O tratamento desses projetos, sem um apoio computacional que auxilie a descoberta automática das etapas nas quais cada projeto está, acarreta inúmeros problemas. Por exemplo, em função do grande número de projetos e da dinâmica com que eles evoluem ao longo do tempo, é muito difícil identificar precisamente e a cada momento quais projetos foram contratados, quais devem receber recursos, ou ainda

quais estão em prestação de contas. O problema é ainda maior quando se observa que cada projeto pode passar por diversas áreas da empresa, conforme o tipo de operação realizada. O desembolso de recursos, por exemplo, requer atividades como autorização, empenho, liquidação e pagamento, que em geral são executadas por pessoas diferentes e de unidades distintas. Sem apoio computacional, a identificação destas etapas fica dependente de alguém que monitore todas as operações de todos os projetos e classifique cada um desses projetos, permitindo então o direcionamento destes para as áreas adequadas. Erros nesta classificação podem causar atrasos desnecessários e onerar o projeto e a financiadora, aumentando o risco operacional da empresa.

O objetivo deste trabalho foi chegar a um modelo de classificação que automaticamente descubra o *status* – estágio do ciclo de vida – de um projeto com base em um conjunto de características que o descrevem. Para a construção deste modelo, foi aplicado um processo de descoberta de conhecimento em bases de dados, no qual técnicas de classificação foram avaliadas em um experimento quanto à acurácia, à interpretabilidade do modelo aprendido e ao desempenho computacional.

Este trabalho está estruturado da seguinte forma. Na seção 2 será apresentado o problema do controle do ciclo de vida de projetos de pesquisa e inovação. Na seção 3, o processo de descoberta de conhecimento em bases de dados é descrito. A seção 4 detalha o experimento realizado e a análise dos resultados. A seção 5 traz um apanhado de trabalhos relacionados. Por fim, a seção 6 apresenta a conclusão.

2. Controle do ciclo de vida de projetos de pesquisa e inovação

A FINEP é uma empresa pública ligada ao Ministério da Ciência, Tecnologia e Inovação (MCTI). Como tal, é a principal instituição de fomento a projetos de pesquisa e inovação, atuando desde a seleção até a conclusão dos projetos. Sua atuação se dá por meio de financiamento nas modalidades reembolsável e não reembolsável. Na segunda modalidade atende às principais universidades e centros de pesquisa do país, através de convênios e termos de cooperação, e às empresas, através de programas de subvenção econômica. Na modalidade reembolsável, atua fornecendo crédito às empresas, para o desenvolvimento de projetos de ciência, tecnologia e inovação.

O ciclo de vida de um projeto de pesquisa na FINEP engloba desde o envio da proposta, passando pela sua análise e processo decisório, pela contratação e posterior acompanhamento técnico-financeiro até o seu encerramento. A empresa possui um procedimento de classificação dos projetos em *status* com base em determinadas características e em processos de trabalho pelos quais o projeto passou.

A existência de um único modelo de classificação de *status* que atenda a todos os instrumentos operados pela empresa tem se mostrado um desafio. Há ainda um sentimento de que é mais difícil determinar aqueles *status* caracterizados por ocorrerem antes da contratação do projeto. Em geral, os erros de classificação oneram a equipe de Tecnologia da Informação, impedindo-a de tratar de outros projetos que poderiam melhorar os sistemas da empresa. Erros na classificação de alguns *status* podem ainda fazer com que projetos sobre os quais não há mais ação a ser tomada por parte da empresa ainda sejam exibidos na carteira de projetos em andamento, aumentando o trabalho dos responsáveis pelo acompanhamento de projetos.

Outras motivações também consideradas são a identificação dos *status* com regras mais claras, a identificação dos atributos relevantes e dos irrelevantes para a

classificação e a possibilidade de descoberta de novos conhecimentos a partir do resultado dos algoritmos.

3. Mineração de Dados e Aprendizado de Modelos de Classificação

A descoberta de conhecimento em bases de dados (em inglês, *Knowledge Discovery in Databases* – KDD) é o processo de extração de conhecimento útil a partir da identificação de padrões em dados [Fayyad et al., 1996]. Este processo é composto pelas etapas de: (i) pré-processamento, onde os dados de entrada são obtidos e tratados através de técnicas de limpeza, integração, seleção e transformação de dados; (ii) mineração dos dados, em que são extraídos os padrões dos dados tratados na etapa anterior; e (iii) pós-processamento, onde são tratadas as questões de visualização dos resultados e interpretação de padrões.

Dentro da mineração de dados, uma das técnicas é o mapeamento de dados em classes pré-definidas, que se baseia no aprendizado de modelos de classificação e pode ser aplicado em diversas situações amplamente difundidas na literatura, por exemplo em [Jacob and Ramani, 2012] e [Depren et al., 2005]. Os problemas de classificação identificam as características que determinam a que grupo um caso pertence. Esse padrão pode ser utilizado quando se deseja compreender os dados ou para a previsão de novos casos.

Um grupo de algoritmos de aprendizado de máquina é próprio para a tarefa de classificação. Esses algoritmos, mais detalhados na seção 3.2, geram modelos de classificação que podem ser aplicados sobre os atributos de uma instância para prever outro atributo classe, desconhecido ou futuro.

3.1. Técnicas de Pré-Processamento

Durante a fase de pré-processamento do processo de KDD, são tratadas questões de ruído, pontos fora da distribuição, valores faltantes e dados duplicados, por exemplo. São várias as técnicas de pré-processamento que podem ser utilizadas, incluindo redução de dimensionalidade, criação de características e discretização.

Na redução de dimensionalidade, atributos inúteis como os que caracterizam redundância e ruído são removidos, o que, em geral, conduz à melhora da acurácia e da velocidade do aprendizado. Há ainda a estratégia de se basear no modelo gerado por algoritmos de aprendizado para identificar atributos irrelevantes. Neste caso, atributos que não foram utilizados no modelo gerado são fortes candidatos a serem removidos. A criação de características consiste em criar novos atributos que capturem as informações importantes em uma base de maneira muito mais eficiente do que os atributos originais. A discretização é a transformação de atributos numéricos em categóricos, muito usada para gerar intervalos que facilitem a mineração de dados.

3.2. Mineração de Dados

A etapa de Mineração de Dados corresponde à aplicação de algoritmos de aprendizado de máquina que aprendem modelos a partir dos dados. As seções seguintes descrevem algoritmos utilizados para aprender modelos de classificação.

3.2.1. Classificadores baseados em Regras

Um classificador baseado em regra é um modelo composto por um conjunto de regras lógicas com a estrutura SE <premissas> ENTÃO <decisão>, onde o conjunto de premissas é composto por pares atributo/valor [Han et al., 2006]. Considerando por exemplo um cenário de financiamento nas modalidades reembolsável e não reembolsável, uma possível regra é a seguinte:

SE *contratado = nao*, *decisao_de_diretoria = sem_decisao* e *arquivado = nao* ENTÃO *status = Em analise*.

Uma nova instância recebida, como uma que descreve um projeto ainda não contratado, sem decisão de diretoria e que não foi arquivado, receberia o status "Em análise" considerando um classificador com a regra acima.

Para aprender automaticamente um classificador baseado em regras, os algoritmos buscam pelo melhor conjunto de pares (atributo,valor), ou seja as premissas, que determinam a decisão a ser tomada e para isso utilizam a base de dados. Analisando por exemplo a base de dados exibida na Tabela 1, um algoritmo aprenderia que basta o indeferimento da decisão de diretoria para que o status do projeto seja "Indeferido". Ou seja, o par (decisao_de_diretoria, Indeferir) é considerado como determinante para a ação atribuir status, assim a seguinte regra é aprendida:

SE *decisao_de_diretoria = indeferir* ENTÃO *status = Indeferido*.

Tabela 1: Base de dados histórica sobre classificação para atribuir status do projeto

#	contratado	decisao_de_diretoria	arquivado	Status
1	nao	aprovar	sim	Arquivado
2	sim	aprovar	nao	Em desembolso
3	nao	indeferir	nao	Indeferido
4	nao	arquivar	nao	Arquivado
5	nao	sem_decisao	nao	Em analise

3.2.2. Classificadores baseados em Árvore de Decisão

Uma árvore de decisão é uma estrutura de dados em árvore na qual os nós internos representam ações, os arcos representam os resultados de uma ação e as folhas representam resultados finais. Em uma árvore de decisão utilizada como um classificador, os nós internos representam atributos, os arcos os possíveis valores do atributo correspondente e as folhas a classificação. Considerando um cenário da concessão de visto, a árvore de decisão da Figura 1 indica como a classificação é feita.

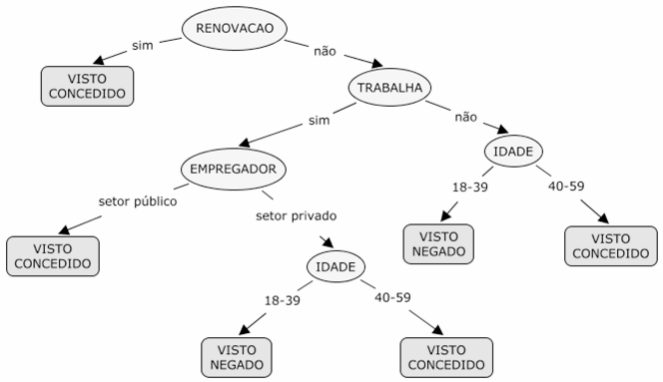

Figura 1: Exemplo de árvore de decisão considerando um cenário da concessão de visto

Dentre as vantagens da árvore de decisão, estão a facilidade de entendimento do modelo, a facilidade de conversão para um conjunto de regras, onde cada ramo da árvore define uma regra [Sonstrod et al., 2011], o fato de não necessitar de suposições sobre a natureza dos dados e a possibilidade de classificação tanto de dados numéricos quanto de dados categorizados, embora o atributo resultante deva ser um dado categorizado [Zhao and Zhang, 2008].

O ponto principal de um algoritmo de aprendizado de árvores de decisão é a seleção dos atributos que serão considerados. A ideia é buscar pelo atributo que trará a pureza mais rapidamente, ou seja, que particionará a base de dados de forma que cada ramo tenha apenas registros relacionados a uma classe. Dessa forma, pequenas variações nos dados utilizados pelo treinamento podem causar seleções de atributos diferentes em cada ponto de escolha da árvore. O efeito pode ser significativo, pois as escolhas de atributos afetam todas as sub-árvores descendentes.

Árvores geradas a partir de um conjunto de dados numéricos podem ser bem complexas, uma vez que a divisão de atributos numéricos se dará de forma binária, o que pode gerar um grande número de nós. A poda é especialmente importante neste caso. Esta técnica reduz o tamanho de árvores de decisão através da remoção de seções da árvore que representem pouca ou nenhuma influência sobre a classificação.

Tanto os classificadores baseados em árvore de decisão quanto os baseados em regras geram modelos com representação simples, sendo de fácil compreensão.

3.2.3. Classificadores baseados em Rede Bayesiana

As redes Bayesianas são grafos direcionados acíclicos onde os nós representam variáveis aleatórias de um domínio específico e os arcos indicam a existência de dependência probabilística entre os nós adjacentes. Essas dependências são expressas por probabilidades condicionais [Matsuura, 2003].

A Figura 2 mostra o grafo de uma rede bayesiana. Os arcos do grafo indicam que a variável *CHUVA* se relaciona com as variáveis *VELOCIDADE BAIXA* e *PISTA ESCORREGADIA* e que as variáveis *VELOCIDADE BAIXA* e *PISTA ESCORREGADIA* se relacionam com a variável *ACIDENTE*. As probabilidades condicionais de cada nó quantificam essas relações.

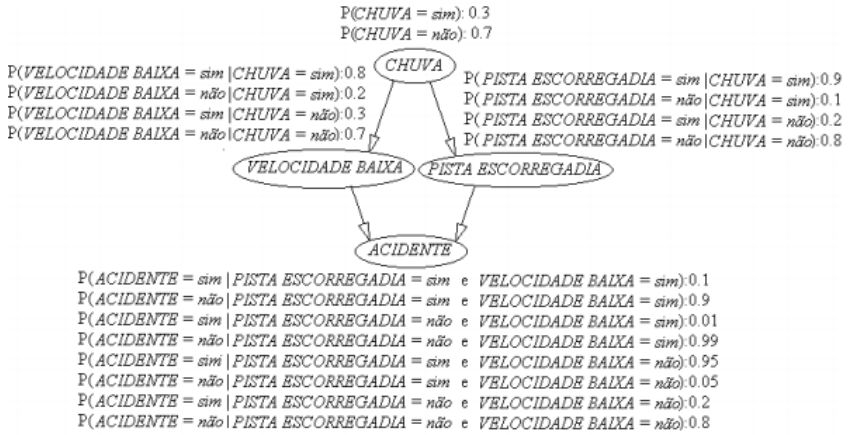

Figura 2: Exemplo de Rede Bayesiana [Matsuura, 2003]

Uma rede bayesiana, muito utilizada em tarefas de classificação, é o Naive Bayes. Neste caso, um nó representa a variável a ser classificada e existem arcos da variável classificada para todos os outros nós, chamados de atributos da classe. Uma característica do Naive Bayes é que os atributos da classe são independentes, ou seja, não possuem arcos entre si [Matsuura, 2003]. Aprender uma rede Naive Bayes significa aprender as probabilidades condicionais, já que a estrutura do grafo é conhecida. A Figura 3 ilustra uma rede Naive Bayes.

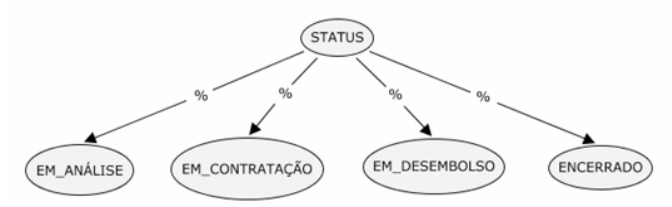

Figura 3: Exemplo de grafo gerado pelo algoritmo NaiveBayes

Por sua característica de inferência sob condições de incerteza, os algoritmos bayesianos são especialmente úteis em problemas de detecção de fraude e na classificação de avaliação de risco. De acordo com [Goldenberg and Moore, 2004], eles também têm sido muito utilizados em sistemas de recomendação, campo que tem recebido muitas atenções nos últimos anos. Sistemas online, como a Amazon e diversos outros sites de comércio eletrônico, utilizam redes bayesianas para prover sugestões aos usuários com base nas suas preferências [Goldenberg and Moore, 2004].

4. Avaliação Experimental

Neste artigo, argumentamos que um modelo para classificação pode ser utilizado na tarefa de identificação do estágio do ciclo de vida de avaliação de um projeto de pesquisa e inovação. Alem disso, propomos a utilização de um processo de descoberta de conhecimento em bases de dados para gerar automaticamente esse modelo de classificação a partir de dados históricos de projetos. Para avaliar a proposta foi executado um estudo de caso utilizando um cenário real na FINEP, descrito a seguir.

O conjunto de dados obtidos para este experimento refere-se a um conjunto de 23.147 propostas de projetos de pesquisa e inovação submetidas à FINEP desde 2003, com o balanceamento indicado na tabela 2. Inicialmente, foram extraídos 16 atributos, incluindo o *status*, listados na Tabela 3. Esses atributos foram selecionados a partir de relatos de especialistas da FINEP, que indicaram que eles poderiam ter participação na determinação do status do projeto. O conjunto foi, então, extraído da base de dados da empresa, com o pré-processamento indicado a seguir.

Dentre os atributos, a modalidade indica se um projeto é não reembolsável, quando assume valor 1, ou reembolsável, quando assume valor 2. Os atributos eh_subvencao, eh_contrato, eh_audiovisual e eh_convenio indicam o tipo de instrumento contratual que o projeto dará origem, o que poderia influenciar no ciclo de vida do projeto.

Tabela 2: Balanceamento do conjunto de dados

Valor	Quantidade	Valor	Quantidade	Valor	Quantidade
EM_ANALISE	971	EM_CONTRATACAO	523	EM_AMORTIZACAO	64
INDEFERIDO	12869	EM_DESEMBOLSO	1761	ENCERRADO	1154
ARQUIVADO	1109	EM_PRESTACAO_DE_CONTAS	4682	RESCINDIDO	14

Tabela 3: Atributos utilizados na 1ª iteração

Atributo	Conjunto de Valores	Atributo	Conjunto de Valores
existe_saldo_a_liberar	sim, nao	modalidade	0, 1, 2
existe_registro_de_arquivamento	sim, nao	eh_convenio	sim, não
contratado	sim, nao	decisao_de_diretoria	aprovar, aprovar_recom, indeferir, arquivar, sem_decisao
possui_data_execucao_expirada	sim, nao		
possui_data_prestacao_contas_expirada	sim, nao		
existe_registro_de_rescisao	sim, nao	Status	EM_ANALISE, INDEFERIDO, ARQUIVADO, EM_CONTRATACAO, EM_DESEMBOLSO, EM_PRESTACAO_DE_CONTAS, EM_AMORTIZACAO, ENCERRADO, RESCINDIDO
existe_registro_de_encerramento	sim, nao		
possui_data_assinatura_maisde5anos	sim, nao		
eh_contratoreembolsavel	sim, nao		
eh_audiovisual	sim, nao		
possui_prestacao_de_contas	sim, nao		

4.1. Pré-processamento

Os dados extraídos foram pré-processados de forma a permitir ou facilitar este experimento. Neste processo, as informações de identificação do projeto foram removidas.

Foram criados novos atributos com base nas informações referentes às datas. Os novos atributos *possui_data_prestacao_contas_expirada* e *possui_data_execucao_expirada* e passaram a contar com um número finito de valores, no caso "sim" ou "nao". Além disso, por se referirem a informações disponíveis apenas no caso de projetos contratados, esses atributos apresentam valores faltantes em 67% dos casos e optou-se por manter esses dados faltantes deixando que os algoritmos utilizados na fase de mineração de dados lidassem com essa questão.

Outros dois atributos, *decisao_de_diretoria* e *modalidade*, também tinham dados faltantes. Nesse caso optou-se por preenchê-los com um novo valor "sem_decisao" e "0" respectivamente, expandindo assim o domínio desses atributos. Já o atributo *contratado* foi obtido a partir da data de assinatura do contrato.

Uma vez selecionados e transformados os atributos, foi gerado um arquivo com todos os 16 atributos, compondo a base de dados no formato utilizado pelo WEKA, ferramenta utilizada no experimento.

4.2. Mineração dos Dados

Foram selecionados algoritmos disponíveis no WEKA, que lidassem com dados faltantes e que levassem no máximo 5 segundos para a geração do modelo. Este último requisito foi um desejo dos especialistas, que arbitraram esse valor. Com isso, foram selecionados cinco algoritmos: dois baseados em regras (JRip e PART), dois baseados em árvore de decisão (J48 e REPTree) e um baseado em redes bayesianas (NaiveBayes). Os parâmetros utilizados para cada um deles com seus valores padrão estão descritos na Tabela 4.

Foram realizadas 4 iterações do experimento:

- 1ª iteração: foram utilizados todos os atributos selecionados inicialmente, com os parâmetros padrões dos algoritmos.

- 2ª iteração: foram excluídos os atributos considerados irrelevantes na 1ª iteração, ou seja, aqueles que não foram utilizados em pelo menos 3 dos 5 algoritmos.
- 3ª iteração: somente com o algoritmo JRip sobre todos os atributos selecionados inicialmente, alterando o valor do parâmetro *optimizations* (número de execuções otimizadas) para 0 e o valor do parâmetro *folds* (quantidade de dados utilizada na poda) para 2. A utilização da poda é comum em algoritmos voltados para o aprendizado de modelos de classificação, e é apresentada nos parâmetros como *usePruning*, *unpruned* e *noPruning*.
- 4ª iteração: somente com o algoritmo JRip, os mesmos parâmetros da 3ª iteração foram aplicados aos atributos utilizados na 2ª iteração.

Tabela 4: Parâmetros utilizados no experimento

JRip	PART	J48	REPTree	NaiveBayes
checkErrorRate = True debug = False folds = 3 minNo = 2.0 optimizations = 2 seed = 1 usePruning = True	binarySplits = False confidenceFactor = 0.25 debug = False minNumObj = 2 numFolds = 3 reducedErrorPruning = False seed = 1 unpruned = False useMDLcorrection = True	binarySplits = False collapseTree = True confidenceFactor = 0.25 debug = False minNumObj = 2 numFolds = 3 reducedErrorPruning = False saveInstanceData = False seed = 1 subtreeRaising = True unpruned = False useLaplace = False useMDLcorrection = True	debug = False initialCount = 0.0 maxDepth = -1 minNum = 2.0 minVarianceProp = 0.0010 noPruning = False numFolds = 3 seed = 1 spreadInicialCount = False	debug = False displayModelInOldFormat = False useKernelEstimator = False useSupervisedDiscretization = False

4.3. Resultados Obtidos

No experimento, utilizou-se o método de validação *10-fold cross-validation* para dividir a base de dados em treinamento e teste. Neste método, os dados são divididos em um número k de *folds*. A cada iteração, um *fold* é apresentado com dados de teste, enquanto os demais k-1 são usados como dados de treinamento. Esse procedimento é executado k vezes, de forma que em cada iteração um dos *folds* possa atuar como teste. O desempenho dos classificadores é calculado como a média obtida nas k iterações (Depren et al., 2005). No experimento, foi utilizado k=10.

Os resultados obtidos foram avaliados quanto à acurácia (que indica o percentual de instâncias classificadas corretamente), a precisão (que indica o percentual de padrões que foram corretamente classificados em uma categoria), a cobertura (que indica o percentual de padrões que foram recuperados) e a *f-measure* (média harmônica da precisão e da cobertura, utilizada para mensurar a exatidão do classificador). Além disso, foram observadas a velocidade de construção do modelo (representando o custo computacional do aprendizado) e a interpretabilidade (clareza e facilidade de interpretação do modelo aprendido pelos usuários finais).

4.4. Análise dos Resultados

Os resultados da 1ª e 2ª iterações do experimento são exibidos na Tabela 5, onde a decisão sobre os atributos irrelevantes removidos na 2ª iteração foi tomada considerando a Tabela 6. Ao todo, 6 atributos foram considerados dispensáveis e foram removidos, sendo que 3 deles (*possui_data_execucao_expirada*, *modalidade*, *possui_data_assinatura_maisde5anos*) não foram utilizados em quatro algoritmos.

A acurácia ficou em torno de 95%, com pouca variação entre os algoritmos utilizados. Dois algoritmos de tipos distintos (PART e J48) dividiram a maior proporção de acertos (95,43%) na 1ª iteração. Com a pior acurácia, embora não tão atrás dos demais algoritmos, ficou o bayesiano NaiveBayes, com 94,75% na 1ª iteração. O valor alto e a pequena diferença entre as acurácias obtidas indica que qualquer um dos algoritmos avaliados poderia ser utilizado na prática.

Na avaliação de velocidade, NaiveBayes foi claramente superior, com o resultado quase instantâneo de 0,02 e 0,01 segundos nas 2 iterações. Também é possível observar que a redução de dimensionalidade (16 para 10 atributos) entre a 1ª e a 2ª iterações trouxe um ganho computacional significativo em todos os algoritmos.

Tabela 5: Resultados da 1ª e da 2ª Iterações

Algoritmo	Acurácia		Velocidade (seg.)		Regras	
	1ª Iter.	2ª Iter.	1ª Iter.	2ª Iter.	1ª Iter.	2ª Iter.
rules.JRip	95,27 %	95,27 %	2,3	1,61	11	10
rules.PART	95,43 %	95,37 %	0,29	0,18	14	13
tree.J48	95,43 %	95,37 %	0,21	0,14	16	14
tree.REPTree	95,41 %	95,42 %	0,3	0,17	17	17
bayes.NaiveBayes	94,75 %	95,37 %	0,02	0,01	-	-

Tabela 6: Atributos que não foram utilizados em cada algoritmo

Atributos	JRip	PART	J48	REPTree
possui_data_execucao_expirada	X	X	X	X
possui_data_prestacao_contas_expirada		X	X	X
eh_subvencao	X	X		X
eh_audiovisual	X		X	X
Modalidade	X	X	X	X
possui_data_assinatura_maisde5anos	X	X	X	X

A análise dos critérios avaliados indica que, mantidas as relações de acurácia e velocidade, a redução de atributos em bases muito grandes pode ser vantajosa em decorrência da diminuição no custo computacional.

O algoritmo de maior custo computacional no experimento foi o JRip, baseado em regras. Mesmo mais lento, a acurácia obtida por este algoritmo não foi boa, ficando em 4º lugar nos testes. Em uma tentativa de melhorar o desempenho desse algoritmo, foram efetuadas algumas modificações nos parâmetros de execução e duas novas iterações foram realizadas, repetindo os atributos das duas primeiras iterações. Com isso, na 4ª iteração (sobre os mesmos atributos da 2ª iteração), a acurácia passou de 95,27% para 95,23% e a velocidade foi reduzida de 1,61 para 0,35 segundos. Ainda que o algoritmo continuasse a ser superado nesses dois quesitos, a compreensão e o controle dos parâmetros provou ser importante. Com o ajuste dos parâmetros, houve um aumento de apenas uma regra, tanto na comparação entre a 3ª e a 1ª iteração quanto entre a 4ª e 2ª iteração, não representando muita diferença na interpretação do modelo.

Na questão da interpretabilidade, os algoritmos baseados em árvore de decisão possuem a vantagem de expressar o modelo gráfica ou textualmente, através da indução de árvores de decisão, como mostra a Figura 4, ou da simples conversão para regras. A análise dos resultados do experimento sugere que a diferença entre o número de regras geradas pelos algoritmos de regras e árvore de decisão não fez tanta diferença na acurácia do modelo gerado, o que poderia levar à adoção do modelo com menor número de regras sem grandes prejuízos. Vale salientar que esta foi uma característica

identificada para esta base de dados e pode não se repetir em outras situações. Também é importante ressaltar que o algoritmo NaiveBayes representa uma ótima opção quando o usuário precisa de um modelo que indique o grau de crença na classificação, já que ele indica a probabilidade para cada uma das classes.

```
contratado = sim
|   aliberar = sim
|   |   encerrado = sim : ENCERRADO (23/0) [10/0]
|   |   encerrado = nao : EM_DESEMBOLSO (1182/9) [585/0]
|   aliberar = nao
|   |   encerrado = sim : ENCERRADO (747/0) [374/0]
|   |   encerrado = nao
|   |   |   pcf = sim
|   |   |   |   contratoreembolsavel = sim : EM_AMORTIZACAO (41/0) [22/0]
|   |   |   |   contratoreembolsavel = nao : EM_PRESTACAO_DE_CONTAS (9/0) [11/0]
|   |   |   pcf = nao
|   |   |   |   rescindido = sim : RESCINDIDO (5/0) [5/0]
|   |   |   |   rescindido = nao : EM_PRESTACAO_DE_CONTAS (3109/1) [1552/2]
contratado = nao
|   decdir = aprovar
|   |   arquivado = sim : ARQUIVADO (144/0) [75/0]
|   |   arquivado = nao : EM_CONTRATACAO (225/0) [113/0]
|   decdir = aprovar_recom
|   |   arquivado = sim : ARQUIVADO (101/0) [52/0]
|   |   arquivado = nao : EM_CONTRATACAO (26/0) [11/0]
|   decdir = indeferir
|   |   arquivado = sim : ARQUIVADO (15/0) [5/0]
|   |   arquivado = nao : INDEFERIDO (5862/16) [2973/11]
|   decdir = arquivar : ARQUIVADO (76/0) [45/0]
|   decdir = sem_decisao
|   |   arquivado = sim : ARQUIVADO (404/0) [191/0]
|   |   arquivado = nao
|   |   |   convenio = sim : INDEFERIDO (3238/590) [1569/293]
|   |   |   convenio = nao : EM_ANALISE (224/85) [123/53]
```

Figura 4: Modelo gerado pelo algoritmo REPTree, na 1ª iteração, utilizando o WEKA.

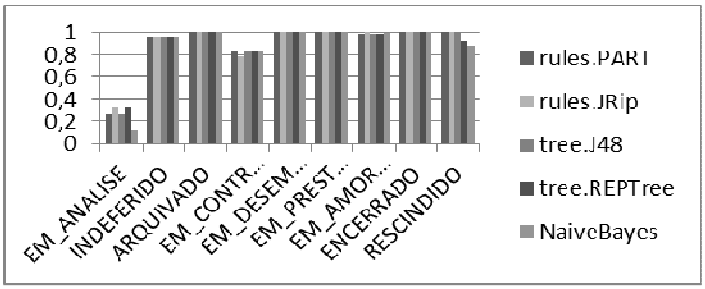

Figura 5: Medida *f-measure* obtida a partir da 2ª iteração

Tendo em vista que, em todas as iterações, os valores obtidos na comparação dos algoritmos foram muito próximos e altos, representando um bom resultado, e que o tempo de execução não teve impacto significativo, todos os algoritmos poderiam ser considerados para a classificação de *status* de projetos.

Algumas outras conclusões e padrões foram observados durante o experimento. Através da análise da precisão, da cobertura e do *f-measure*, foi possível identificar em que situações houve maior dificuldade de classificação. Os baixos valores obtidos no *status* EM_ANALISE, e que podem ser observados na Figura 5, indicam que este é o mais problemático dentre os valores possíveis. Outro valor que requer atenção é o *status* EM_CONTRATAÇÃO, que apesar de ter tido alta precisão, apresentou a segunda pior cobertura. As medidas obtidas pelos dois *status* são exibidas na Tabela 7. Todos os dois representam estágios anteriores à efetiva contratação do projeto, o que corrobora a percepção dos quatro analistas da empresa consultados em entrevista informal. O grupo consultado incluía um representante da equipe de banco de dados, um analista de negócio, um desenvolvedor e um representante das áreas finalísticas da empresa.

Além disso, a análise das regras geradas pode, em alguns casos, sugerir problemas nos dados ou mesmo falhas em processos, como a ausência de algum registro

que deveria ter sido efetuado. No caso relatado, o experimento permitiu identificar alguns poucos convênios que, embora já tivessem sua prestação de contas final concluída, não haviam sido encerrados, situação que não deveria ocorrer. Posteriormente, parte do experimento foi reproduzido na empresa com atributo adicional que permitisse a identificação desses projetos, que foram então encaminhados para verificação pelos responsáveis pelo acompanhamento dos projetos. O processo em questão também foi revisado. Isso reforça a análise dos dados por especialistas no domínio e ilustra como esta técnica pode ser utilizada inclusive para melhorar a qualidade de dados e dos processos.

Tabela 7: Medidas de precisão, cobertura e *f-measure* dos status EM_ANALISE e EM_CONTRATACAO obtidas durante a 2ª iteração

	EM_ANALISE		
	Precisão	**Cobertura**	**F-Measure**
rules.PART	0,60	0,17	0,27
rules.JRip	0,60	0,22	0,32
tree.J48	0,60	0,17	0,27
tree.REPTree	0,60	0,22	0,32
NaiveBayes	0,98	0,07	0,12

	EM_CONTRATACAO		
	Precisão	**Cobertura**	**F-Measure**
rules.PART	1,00	0,72	0,84
rules.JRip	1,00	0,65	0,79
tree.J48	1,00	0,72	0,84
tree.REPTree	1,00	0,72	0,84
NaiveBayes	1,00	0,72	0,83

5. Trabalhos Relacionados

Ainda que em um campo diferente do aplicado neste trabalho, [Piggott, 2013] também enfrentou problema semelhante no contexto de projetos de software aberto. Em seu artigo, ele também aplica mineração de dados, explorando o uso de árvores de decisão para obter um modelo de classificação com o objetivo de determinar o estágio em que o projeto se encontra com base em suas métricas, restrições e fatos relacionados.

A utilização de técnicas de mineração de dados para auxiliar na detecção, quantificação, explicação e correção de deficiências na qualidade dos dados é defendida por [Hipp et al., 2001] como um campo promissor sob o ponto de vista corporativo.

[E. Januzaj and V. Januzaj, 2009] apresenta uma abordagem que facilita a identificação de problemas nos dados armazenados em data warehouse com base em técnicas de mineração de dados, como agrupamento e classificação. Segundo os autores, a mineração de dados facilita a identificação de problemas de qualidade de dados existentes sem que seja necessário saber se o problema de fato existe.

[Espinosa et al., 2011] analisa as técnica de mineração de dados para entender o comportamento de diversos critérios de qualidade de dados nas origens e como eles afetam os resultados dos algoritmos de mineração de dados. O artigo lida com os problemas de qualidade, e não na melhoria da qualidade dos dados de origem.

6. Conclusão

O presente trabalho apresentou uma abordagem para construção automática de um classificador para determinar o estágio do ciclo de vida de um projeto. A abordagem segue o processo de KDD e foi experimentada em um cenário real, onde os resultados foram analisados quanto à acurácia, interpretabilidade do modelo aprendido e desempenho computacional.

O experimento foi conduzido sobre a base de dados de projetos da FINEP e avaliou o desempenho de cinco algoritmos de aprendizado de classificadores, dois deles

baseados em árvore de decisão, dois baseados em regras e um baseado em redes bayesianas. No quesito acurácia, todos os algoritmos apresentaram percentuais na casa dos 95%, o que indica que todos poderiam ser considerados na classificação do *status* dos projetos. Em geral, foram utilizadas as mesmas opções em cada um dos algoritmos, embora tenha sido efetuado um pequeno teste com o algoritmo baseado em regras JRip, que apresentava tempos bem superiores aos demais. Foram executados cenários variando-se pontualmente os valores de alguns parâmetros, o que resultou em uma redução sensível do seu tempo de processamento, embora ainda tenha permanecido superior ao tempo dos demais algoritmos.

Foi possível confirmar algo que era apenas um sentimento dos analistas da empresa, identificar os atributos irrelevantes, e ainda identificar potenciais problemas em projetos e em processos de trabalho a partir da análise do modelo gerado. Assim, a utilização da técnica e a análise dos resultados por especialistas no domínio podem ajudar a sanar problemas de qualidade de dados e falhas em processos.

No domínio da empresa, a análise dos classificadores gerados e de métricas como precisão, cobertura e *f-measure* pode ser aprofundada, permitindo a identificação de outras questões que possam ser tratadas ou que possam vir a melhorar os processos internos. Também seria interessante tentar aplicar a técnica na análise de crédito de contratos reembolsáveis, situação em que a avaliação é mais subjetiva e os aspectos avaliados não costumam ser determinísticos.

Referências

Depren, O., Topallar, M., Anarim, E., Ciliz, M.K., 2005. An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks. Expert Systems with Applications 29, 713–722.

Espinosa, R., Zubcoff, J., Mazón, J.-N., 2011. A Set of Experiments to Consider Data Quality Criteria in Classification Techniques for Data Mining.

Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., 1996. From Data Mining to Knowledge Discovery in Databases. AI Magazine 17, 37.

Goldenberg, A., Moore, A., 2004. Tractable learning of large Bayes net structures from sparse data, in: Proceedings of the Twenty-first International Conference on Machine Learning, ICML'04. ACM, New York, NY, USA, p. 44–.

Han, J., Kamber, M., Pei, J., 2006. Data Mining, Second Edition: Concepts and Techniques. Morgan Kaufmann.

Hipp, J., Güntzer, U., Grimmer, U., 2001. Data Quality Mining – Making a Virtue of Necessity, in: In Proceedings of the 6th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD 2001). pp. 52–57.

Jacob, S.G., Ramani, R.G., 2012. Mining of classification patterns in clinical data through data mining algorithms, in: Proceedings of the International Conference on Advances in Computing, Communications and Informatics, ICACCI'12. ACM, New York, pp. 997–1003.

Januzaj, E., Januzaj, V., 2009. An application of data mining to identify data quality problems, in: 3rd International Conference on Advanced Engineering Computing and Applications in Sciences, ADVCOMP 2009. pp. 17–22.

Matsuura, J.P., 2003. Discretização para aprendizagem bayesiana: aplicação no auxílio à validação de dados em proteção ao vôo. Instituto Tecnológico de Aeronáutica, SP.

Piggott, J.J.H., 2013. Open Source Software Attributes as Success Indicators. Univ. of Twente.

Sonstrod, C., Johansson, U., Konig, R., 2011. Evolving accurate and comprehensible classification rules, in: IEEE Congress on Evolutionary Computation (CEC)., pp. 1436-1443.

Zhao, Y., Zhang, Y., 2008. Comparison of decision tree methods for finding active objects. Advances in Space Research 41, 1955–1959.