Identificação de Fraudes em Licitações:
uma abordagem utilizando agrupamento por interseção

David Pereira Galvão Júnior

CENTRO DE INFORMÁTICA
UNIVERSIDADE FEDERAL DA PARAÍBA

João Pessoa, 2023

David Pereira Galvão Júnior

Identificação de Fraudes em Licitações

Monografia apresentada ao Programa de Pós-Graduação em Informática
do Centro de Informática, da Universidade Federal da Paraíba,
como requisito para a obtenção do título de Mestre em Informática

Orientador: Gilberto Farias de Sousa Filho

Julho de 2023

Catalogação na publicação
Seção de Catalogação e Classificação

G182i Galvão Júnior, David Pereira.
Identificação de fraudes em licitações : uma abordagem utilizando agrupamento por interseção / David Pereira Galvão Júnior. - João Pessoa, 2023.
86 f. : il.

Orientação: Gilberto Farias de Sousa Filho.
Dissertação (Mestrado) - UFPB/CI.

1. Licitações - Fraudes. 2. Aprendizagem de máquina. 3. Análise de grupos. I. Sousa Filho, Gilberto Farias de. II. Título.

UFPB/BC CDU 658.715(043)

Elaborado por Larissa Silva Oliveira de Mesquita - CRB-15/746

UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

Ata da Sessão Pública de Defesa de Dissertação de Mestrado de **DAVID PEREIRA GALVÃO JÚNIOR**, candidato ao título de Mestre em Informática na área de Sistemas de Computação, realizada em 13 de julho de 2023.

Aos treze dias do mês de julho do ano de dois mil e vinte e três, às 10h, no Centro de Informática da Universidade Federal da Paraíba, reuniram-se os membros da Banca Examinadora constituída para julgar o Trabalho Final do discente David Pereira Galvão Júnior, vinculado a esta Universidade sob a matrícula nº 20211023510, candidato ao grau de Mestre em Informática, na área de *"Sistemas de Computação"*, na linha de pesquisa *"Computação Distribuída"*, do Programa de Pós-Graduação em Informática. A comissão examinadora foi composta pelos professores: Gilberto Farias de Sousa Filho, Orientador e Presidente; Bruno Jefferson de Sousa Pessoa, Examinador Interno ao Programa; Fábio Protti, Examinador Externo à Instituição; e Thiago Gouveia da Silva, Examinador Externo à Instituição. Dando início aos trabalhos, o Presidente da Banca cumprimentou os presentes, comunicou a finalidade da reunião e passou a palavra ao candidato para que fizesse a exposição oral do trabalho de dissertação intitulado *"Identificação de Fraudes em Licitações - Uma abordagem utilizando agrupamento por interseção"*. Concluída a exposição, o candidato foi arguido pela Banca Examinadora que emitiu o seguinte parecer: "***aprovado***". Do ocorrido, eu, Fernando Menezes Matos, Coordenador do Programa de Pós-Graduação em Informática, lavrei a presente ata que vai assinada por mim e pelos membros da Banca Examinadora. João Pessoa, 13 de julho de 2023.

gov.br Documento assinado digitalmente
FERNANDO MENEZES MATOS
Data: 21/07/2023 16:31:33-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Fernando Menezes Matos

Prof. Dr. Gilberto Farias de Sousa Filho
Orientador (PPGI-UFPB)

Prof. Dr. Bruno Jefferson de Sousa Pessoa
Examinador Interno (PPGI-UFPB)

Prof. Dr. Fábio Protti
Examinador Externo à Instituição (UFF)

Prof. Dr. Thiago Gouveia da Silva
Examinador Externo à Instituição (IFPB)

**** O cérebro é como um músculo.*
Quando pensamos bem, nos sentimos bem
*****
(Carl Sagan)

AGRADECIMENTOS

Agradeço imensamente à minha família, em especial aos meus pais, sou grato por todo o incentivo durante os momentos de dedicação ao estudo e à pesquisa.

Ao meu orientador Gilberto agradeço pelo seu apoio, que foi fundamental para o desenvolvimento deste trabalho. Sua orientação me guiou em todas as etapas, desde a concepção das ideias até a elaboração final desta dissertação.

Gostaria também de agradecer aos meus colegas de curso, Luyza, Pablo e Victor, que sempre estiveram dispostos a compartilhar conhecimentos, trocar experiências e fornecer apoio mútuo ao longo dessa jornada acadêmica.

Por fim, agradeço a todos os professores e profissionais do Centro de Informática que contribuíram para o meu crescimento acadêmico e pessoal.

Este trabalho não seria possível sem o apoio de todas essas pessoas especiais que fizeram parte dessa trajetória. A todos, meu profundo sentimento de gratidão.

RESUMO

Fraudes em licitações causam prejuízos significativos à sociedade, diminuindo a eficácia de serviços públicos, como saúde e educação. Apesar do fato de que as autoridades buscam fazer uma intensa investigação para mitigar esse problema, a sua identificação não é uma tarefa trivial, uma vez que os agentes fraudadores empregam táticas sofisticadas. Muitos dos trabalhos anteriores procuraram identificar uma licitação fraudulenta por meio da análise de fatores, como por exemplo, os valores financeiros das propostas submetidas pelas empresas participantes e o comportamento dos participantes de uma licitação. Recentemente, surgiram trabalhos que buscam utilizar essa análise como entrada adicional de um algoritmo de aprendizagem de máquina, com fins de detecção automática de fraudes. Neste trabalho, propomos investigar a participação conjunta de empresas em licitações. Para tanto, introduzimos um novo modelo de agrupamento, que busca maximizar o uso de recursos em comum pelos membros do grupo. Desenvolvemos um conjunto de ferramentas para sua resolução: modelo de programação inteira e algoritmo branch-and-bound. Além disso, demonstramos que a versão de particionamento do modelo é um problema NP-Completo e propomos uma adaptação da função silhueta para medir a qualidade dos grupos gerados. Adicionalmente, introduzimos uma variação desse modelo para agrupamentos por cobertura. Para a resolução dessa versão, é proposto um algoritmo enumerativo e um modelo de programação inteira. Nos experimentos realizados, o novo modelo de agrupamento consegue ser superior em comparação a modelos da literatura baseados em distância e edição de arestas. Especificamente, em todos os casos testados, o novo modelo obteve uma soma de interseções igual ou superior. Dos grupos obtidos buscamos medir o quanto da participação em conjunto dos membros em licitações se deu ao acaso ou não. Para tanto, propomos um conjunto de métricas para descrever os grupos gerados. Essas métricas são utilizadas como entrada adicional de um modelo de aprendizagem de máquina. Em dados de licitações de diversos países, os modelos que fazem uso das métricas propostas neste trabalho conseguem superar os modelos que fazem uso das métricas da literatura. Em média, os modelos propostos obtiveram um ganho de aproximadamente 8% na correlação de validação, em comparação com as métricas da literatura.

Palavras-chave: Análise de Grupos, Aprendizagem de Máquina, Identificação de Fraudes em Licitações.

ABSTRACT

Bid rigging in public procurement auctions causes significant harm to the society, reducing the effectiveness of public services such as health and education. Despite being object of intense scrutiny by the authorities to mitigate this problem, its identification is not a trivial task, since fraudsters employ sophisticated tasks. Many of the previous works sought to identify a fraudulent bidding process through the analysis of factors, such as, for example, the financial values of the proposals submitted and the behavior of the participants in a bidding process. Recently, several works have proposed using this analysis as an additional input to a machine learning algorithm, with the purpose of automatic detection of fraudulent bidding. In this work, we propose to investigate the joint participation of companies in bidding processes. With this goal, we introduce a new clustering model, which seeks to maximize the use of common resources by cluster members. We have developed a set of tools to solve it: integer programming model and branch-and-bound algorithm. Furthermore, we demonstrate that the partitioning version of this model is a NP-Complete problem and we propose an adaptation of the silhouette function to measure the quality of the generated clusters. Additionally, we introduce a variation of this model for coverage clustering. To solve this version, we propose an enumerative algorithm and an integer programming model. In the experiments performed, the new clustering model manages to be superior in relation to literature models based on distance and edge editing. Specifically, in all cases tested, the new model obtained an equal or greater sum of intersections. From the obtained clusters we sought to measure how much of the joint participation of the members in bids occurred by chance or not. To do so, we proposed a set of metrics to describe the clusters. These metrics are used as an additional input to a machine learning model. In public tender data from different countries, the models that make use of the metrics proposed in this work manage to outperform the models that make use of the metrics in the literature. On average, the proposed models obtained a gain of approximately 8% in the validation correlation, in comparison with the literature metrics.

Key-words: Cluster Analysis, Machine Learning, Bid Rigging Identification.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

AM - Aprendizagem de Máquina

AUC - Área abaixo da curva (do inglês, Area under the curve)

BEP - Problema de Edição de Biclusters (do inglês, Bicluster Editing Problem)

CEP - Problema de Edição de Grupos (do inglês, Cluster Editing Problem)

CV - Coeficiente de Variação

DIFFP - Diferença entre os mínimos

DT - Árvore de Decisão (do inglês, Decision Tree)

K-NN - K-Vizinhos mais Próximos (do inglês, K-Nearest Neighbors)

KSTEST - Teste de Kolmogorov-Smirnov

KURT - Excesso de Curtose

MPI - Modelo de Programação Inteira

NN - Rede Neural (do inglês, Neural Network)

PA - Problema de Agrupamento

PAA - Problema de Agrupamento Automático

PAI - Problema de Agrupamento por Interseção

PIB - Produto Interno Bruto

PTE - Estimativa Pré-licitação (do inglês, Pre-tender estimate)

RA - Regras de Associação

RD - Distância Relativa (do inglês, Relative Distance)

RF - Random Forest

ROC - Característica de Operação do Receptor (do inglês, Receiver Operating Characteristic)

SKEW - Assimetria

SPD - Amplitude

SVM - Máquina de Vetor de Suporte (do inglês, Support Vector Machine)

Sumário

1 INTRODUÇÃO

1.1 Definição do Problema

Na administração pública, a licitação é um processo adotado para a contratação de bens e serviços. Em um processo licitatório, entidades como empresas e pessoas competem para serem escolhidas como a vencedora do processo. Essa competição visa possibilitar ao poder público acesso a melhores preços nos seus insumos. O Banco Mundial estima que as licitações em 2018 totalizaram 11 trilhões de dólares, ou cerca de 12% do Produto Interno Bruto (PIB) global daquele ano [13]. No Brasil, estima-se que as licitações tenham sido responsáveis por cerca de 20% do PIB em 2018.

Apesar da existência de várias medidas preventivas e das licitações serem objeto de forte escrutínio das autoridades, não é rara a existência desse tipo de fraude em processos licitatórios, acarretando em prejuízos consideráveis aos cofres públicos. Um dos tipos mais comuns de fraude é o conluio entre competidores, que envolve a atuação coordenada entre várias empresas participantes, também chamado de cartel, com fins de aumentar seus lucros. Existe um grande desafio para as autoridades detectarem a formação de um conluio em uma licitação, pois os membros de um cartel tendem a adotar estratégias sofisticadas para dificultar a descoberta dessas práticas [30].

Visando o desenvolvimento de ferramentas para auxiliar na detecção de fraudes em licitações, tais como o conluio entre participantes, diversos autores propuseram métodos que fazem uso de uma análise estrutural [36, 35] e/ou comportamental [52]. Essa análise contempla a investigação de fatores, como por exemplo, condições de mercado, preços das propostas e a distribuição dos valores dessas. Mais recentemente, esses fatores têm sido empregados para a classificação de licitações através de técnicas de aprendizado de máquina [34, 47].

Neste trabalho, procuramos extrair informações de dados de licitações públicas, com o objetivo de detectar conluios. Para isso, é proposto agrupar empresas, que em seu histórico participaram em conjunto em várias licitações, para então quantificar o quão dessa participação conjunta não ocorreu ao acaso. Para tanto, introduzimos um novo modelo de agrupamento, denominado de Problema de Agrupamento por Interseção (PAI), que visa maximizar a soma das interseções de recursos entre membros de um grupo. Esse modelo de agrupamento é utilizado para criar novas características que refinem um classificador. O processo de classificação de licitações proposto se inicia no mapeamento de informações sobre licitações e seus participantes em uma instância do PAI. Em seguida, são extraídas informações dos grupos encontrados que serão utilizados como dados adicionais para a entrada de um algoritmo de aprendizagem de máquina.

1.2 Objetivos

Nesta seção, são apresentados o objetivo geral e os objetivos específicos deste trabalho.

1.2.1 Objetivo geral

Elaborar um modelo para detecção de fraudes em licitações, com auxílio de dados obtidos por meio do agrupamento de empresas participantes com base no histórico de participações em licitação em conjunto.

1.2.2 Objetivos específicos

a) Propor um problema de otimização, denominado de Problema de Agrupamento por Interseção (PAI);

b) Demonstrar que o PAI é um problema NP-Completo;

c) Propor uma métrica baseada na função silhueta para avaliação de agrupamentos do PAI;

d) Formular um modelo de programação inteira para resolução do PAI;

e) Elaborar um algoritmo de branch-and-bound para resolução do PAI;

f) Propor uma versão alternativa do PAI para geração de grupos por cobertura, denominado de PAI_{cov};

g) Formular um modelo de programação inteira para resolução do PAI_{cov};

h) Criar um algoritmo enumerativo para resolução do PAI_{cov};

i) Propor um conjunto de métricas para descrever os grupos gerados;

j) Criar modelos de classificação de licitações fraudulentas com o auxílio de técnicas de aprendizado de máquina. Os modelos fazem uso das métricas obtidas por meio do agrupamento de empresas em uma licitação pelo PAI;

k) Avaliar a eficiência dos métodos propostos, em termos de subsidiar o modelo de detecção de fraudes e de extrair informações sobre participação concomitante entre objetos;

1.3 Justificativa

As licitações são o caminho para entidades públicas comprarem insumos para, entre outros fins, prover serviços essenciais para a população, como por exemplo, saúde e educação. Fraudes em processos licitatórios afetam negativamente a capacidade dessas entidades proverem serviços de forma eficiente, desperdiçando assim o dinheiro do contribuinte. Adicionalmente, em relatório divulgado em 2022 [56], a Transparência Internacional atribuiu uma nota de 38 (na escala entre 0 à 100) ao Brasil em percepção da corrupção, ranqueando o país na 94ª posição entre 180 países estudados, o que demonstra que o país necessita reforçar medidas para combater a corrupção.

Por sua vez, a descoberta dessas fraudes é um processo não trivial, que requer intensa investigação das autoridades competentes. Em trabalhos passados relacionados, as ferramentas criadas para auxílio na detecção de fraudes se baseiam principalmente na análise de fatores como, por exemplo, condições de mercado, preços das propostas e a distribuição dessas.

Neste trabalho, propomos trilhar um caminho diferente para a criação de uma ferramenta de detecção de fraudes em licitações. Conforme mencionado anteriormente, propomos agrupar empresas conforme sua participação em conjunto em licitações para medir o quanto essa participação não foi ao acaso, isto é, se deve a algum tipo de acordo entre um grupo de empresas. Para tanto, é necessário um modelo de agrupamento adequado para esta tarefa.

Inicialmente, considere um modelo básico de agrupamento onde a proximidade entre dois objetos i e j é baseada em uma função de distância $d(i, j)$. Nesse sentido, o primeiro problema que surge é definir qual seria a distância entre duas empresas, uma vez que as ofertas submetidas em uma licitação não se tratam de valores ordinais. Considere um problema de agrupar uísques pelas características similares que possuem. Para tanto, em [39] é utilizada a função de distância de Jaccard

$$d(i,j) = 1 - \frac{|L_i \cap L_j|}{|L_i \cup L_j|},$$

que assume 0 quando a interseção das características do objeto é igual a sua união (ou seja, a similaridade é máxima) e 1 quando a interseção é vazia. No caso das licitações, ao mapear L_i como o conjunto de propostas submetidas pela empresa i, é obtida a matriz de distância de Jaccard. Essa matriz serve como entrada para o modelo p-medianas [23], que busca agrupar objetos de forma que as distâncias destes ao objeto protótipo seja minimizada. No entanto, ao observar a solução formada, os grupos de empresas muitas vezes não possuem uma licitação sequer em comum. Para ilustrar esse fenômeno, considere

as empresas A, B, e C da Figura 1. O grupo azul possui a empresa A como protótipo, e portanto, as empresas B e C possuem uma boa interseção com A. No entanto, esse relacionamento não possui transitividade, uma vez que B e C podem não ter participado de forma conjunta em qualquer licitação, e tal distância não é considerada na função objetivo do modelo.

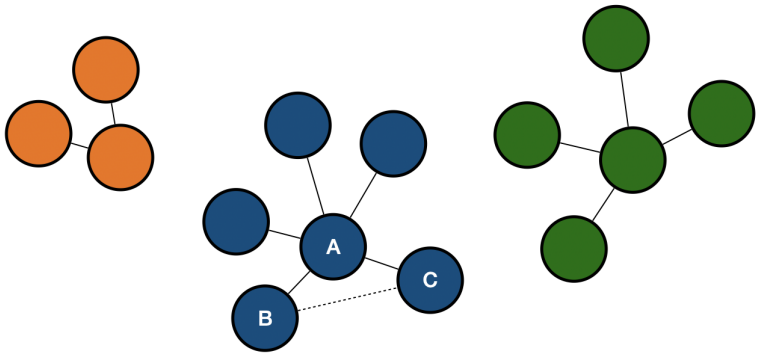


Figura 1: Agrupamento resultante do modelo p-medianas.

Para mitigar a não transitividade da distância de Jaccard, é fornecida a matriz de Jaccard ao modelo do Problema de Edição de Clusters (CEP) [10], que busca particionar o grafo de entrada em cliques disjuntas por meio da modificação mínima de suas arestas (adição ou remoção). Conforme ilustrado na Figura 2, em cada grupo, todas as ligações entre objetos são consideradas na função objetivo. No entanto, mesmo se as empresas A, B e C estiverem no mesmo grupo, não há garantias que as licitações contidas na interseção entre A e B sejam as mesmas da interseção entre B e C. Portanto, a solução continua com poucas interseções.

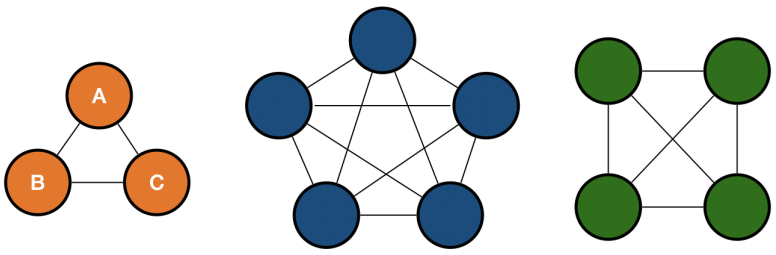


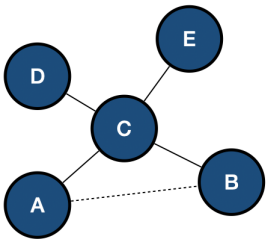
Figura 2: Agrupamento resultante do modelo CEP.

Como aprimoramento, propomos então utilizar uma medida de proximidade entre dois objetos i e j a partir de uma função de similaridade $\delta(i, j)$, que mede o número de características diferentes entre objetos. Em outras palavras, quanto mais características similares os objetos possuírem, mais próximos eles estarão [32]. Esse modelo de agrupamento é utilizado para dados categóricos não ordinais [4].

No contexto deste trabalho, cada oferta submetida por uma empresa seria associada a uma variável binária indicando se tal companhia participou de uma dada licitação, sendo

então mapeada a função $\delta(i, j)$ ao modelo p-medianas, conforme proposto em [33]. No entanto, tal função de similaridade também sofre da não transitividade na suas relações.

Na Figura 3 é possível observar na tabela de similaridades que a empresa C leva as empresas A e B ao seu grupo por possuir apenas duas dissimilaridades. Todavia, se observado o $\delta(A, B)$, tem-se que a dissimilaridade é total, criando assim grupos com interseções pequenas.



	1	2	3	4
A	1	0	1	0
B	0	1	0	1
C	**1**	**1**	**1**	**1**
D	1	0	0	1
E	0	0	1	1

Figura 3: Da esquerda para a direita: agrupamento resultante do modelo p-medianas com função de similaridade, tabela de similaridade das empresas.

O problema persiste mesmo ao abandonar o relacionamento entre empresas par a par. Considere o modelo de bicluster [42], no qual dois conjuntos de entidades distintas se relacionam, com a proibição de relações entre elementos de uma mesma entidade. Observe o grafo bipartido G na parte esquerda da Figura 4. Nele há o conjunto de empresas em uma entidade, representado por círculos, e o de licitações em outra, representado por quadrados, sendo as arestas representações da participação das empresas nas licitações. Esse grafo é fornecido ao modelo do Problema de Edição de Bicluster (BEP) [55] que busca particioná-lo em bicliques disjuntas, de forma que se minimize o número de arestas modificadas.

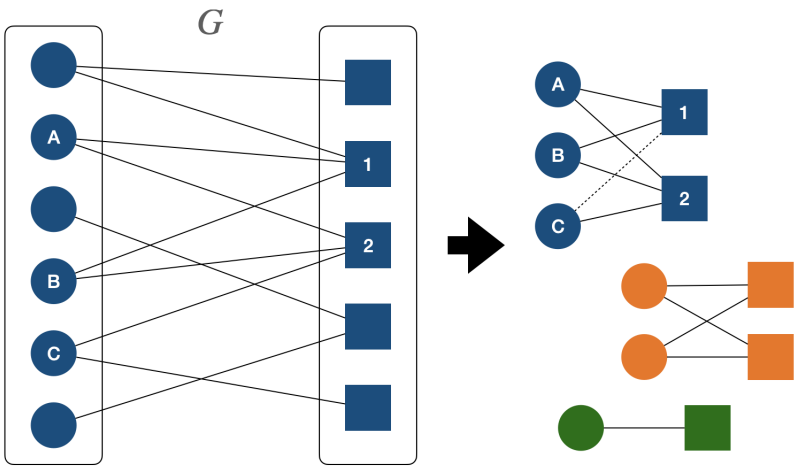
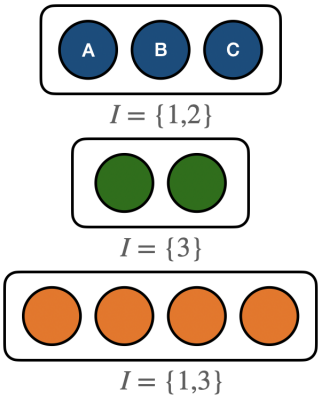


Figura 4: Da esquerda para a direita: grafo bipartido G, agrupamento resultante do modelo BEP.

A utilização do BEP não resolve o problema de obter grupos com interseções pequenas. Revisitando a parte direita da Figura 4, tem-se que o biclique formado pelas empresas A, B, e C possui as licitações 1 e 2. No entanto, a empresa C não participou da licitação 1, cuja relação foi adicionada pelo modelo. Isso significa que, apesar das poucas alterações feitas por este no grafo original, o modelo gerou um grupo que contém apenas a licitação 2 na sua interseção.

Sendo assim, é introduzido por este trabalho o Problema de Agrupamento por Interseção, que não utiliza o relacionamento direto entre dois objetos, mas sim da avaliação do tamanho da interseção dos recursos dos objetos membros de cada grupo. Na Figura 5 é possível visualizar que o grupo formado pelas empresas A, B, e C possui duas licitações na sua interseção, uma vez que todas as empresas participaram das licitações 1 e 2.



	1	2	3	4
A	1	1	0	1
B	1	1	1	0
C	1	1	0	0

Figura 5: Da esquerda para a direita: Agrupamento resultante do modelo de Agrupamento por Interseção, tabela de participação das empresas A, B e C nas licitações 1, 2, 3, 4.

1.4 Estrutura da monografia

O restante deste trabalho se encontra dividido em 6 capítulos. O Capítulo 2 aborda fundamentos teóricos necessários para a elaboração deste trabalho. O Capítulo 3 se dedica aos trabalhos relacionados sobre identificação de fraudes em licitações. O Capítulo 4 trata do PAI, sendo descrito formalmente o problema, e apresentando um conjunto de ferramentas para a resolução do problema. O Capítulo 5 descreve as etapas para utilização dos grupos obtidos pelo PAI como entrada para o método proposto de identificação de fraudes, incluindo as métricas adotadas para descrição dos grupos. O Capítulo 6 apresenta os resultados do algoritmo proposto para resolução do PAI, bem como o desempenho do modelo proposto para a classificação das licitações. Por fim, o Capítulo 7 realiza as considerações finais deste trabalho e aborda propostas de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Aprendizagem de Máquina

A área de *Aprendizagem de Máquina* procura elaborar algoritmos que aprendem com base em um conjunto de dados. A maior parte das tarefas da aprendizagem de máquina se dividem em três categorias:

- **Aprendizagem supervisionada:** O algoritmo recebe um conjunto de dados com os valores das variáveis de entrada e saída. Para esse conjunto, supõe-se que exista uma função f, desconhecida, que define o padrão dos dados. O objetivo do modelo de aprendizagem de máquina é fornecer uma hipótese g que se aproxime ao máximo de f. A depender do domínio da variável de saída, é possível dividir as tarefas da aprendizagem supervisionada entre problemas de regressão e classificação. Há uma vasta gama de algoritmos para resolução de uma tarefa dessa natureza. Entre eles, é possível citar Redes Neurais, Máquinas de Vetor de Suporte (SVM), e Árvores de Decisão. Ainda, existem algoritmos que fazem uso de múltiplos classificadores, como é o caso do algoritmo Random Forest. As Seções 2.2.1 à 2.2.5 abordam alguns desses algoritmos em detalhes;

- **Aprendizagem não supervisionada:** Neste caso, o algoritmo desconhece os valores das variáveis de saída. Uma tarefa desse tipo consiste em encontrar padrões entre as variáveis de entrada e inclui, por exemplo, a redução de dimensionalidade e o agrupamento de objetos. A Seção 2.4 trata de forma mais aprofundada problemas de agrupamento;

- **Aprendizagem por reforço:** Se caracteriza pelo algoritmo interagir com o seu ambiente com base em uma política. A ação escolhida pelo algoritmo faz com que ele receba uma recompensa do ambiente. O objetivo nesse caso é de selecionar ações de forma que a recompensa futura seja maximizada. Algoritmos para esse tipo de tarefa incluem, entre outros, o método de Monte Carlo, e Q-learning.

2.2 Algoritmos de Aprendizagem supervisionada

2.2.1 Árvore de Decisão

A *Árvore de Decisão* é um algoritmo bastante utilizado para resolução de tarefas de aprendizagem supervisionada. Uma das suas grandes vantagens é a facilidade de interpretação, uma vez que o modelo gerado por esta é visualizável. Na árvore, cada nó interno representa um teste a ser feito com uma das variáveis de entrada. Sendo criados

ramos para os valores (ou uma faixa de valores) do atributo. Cada nó folha representa o valor da variável de saída a ser retornado.

Basicamente, para cada objeto a ser avaliado, o modelo de uma árvore de decisão realiza uma sequência de testes, que conduz este até um nó folha. No nó folha, é atribuído o valor do atributo que se deseja prever [49].

Idealmente, uma Árvore de Decisão deve ser consistente com os dados utilizados para inferir as regras. Além disso, é necessário que ela seja equilibrada em relação ao seu tamanho. Isso deve ao fato que árvores grandes tendem a criar hipóteses complexas, e assim, são mais suscetíveis a um sobre-ajuste dos dados.

Considerando um conjunto de dados qualquer com n variáveis booleanas, existem 2^{2^n} árvores de decisão distintas. Essa quantidade torna inviável uma busca exaustiva pela melhor árvore.

No entanto, existem heurísticas para a obtenção de boas aproximações para a árvore de decisão ótima [19]. Um dos métodos mais conhecidos emprega uma estratégia gulosa de divisão e conquista: escolhe a variável mais importante primeiro para ser utilizada como teste, e então, utiliza esse teste para dividir o problema em sub-problemas que podem ser resolvidos recursivamente, comumente. Comumente, a variável mais importante é definida por meio do conceito de ganho de informação.

Alternativamente, Árvores de Decisão construídas por essa heurística possuem outros fins, como por exemplo, ajudar a selecionar variáveis que são importantes para a classificação [27].

2.2.2 Random Forest

O algoritmo *Random Forest* consiste em utilizar múltiplas árvores de decisão para construção de sua função hipótese. Comumente, cada árvore é treinada com base em uma amostra dos dados. Para um dado a ser previsto, o valor da variável de saída é definido, em um problema de classificação como o rótulo da maioria das árvores e em um problema de regressão como a média dos valores previstos pelas árvores [15].

A combinação de múltiplas árvores permite que a hipótese gerada por este algoritmo seja menos suscetível a um sobre-ajuste dos dados. Além disso, o modelo é mais robusto em relação a um modelo gerado por uma única árvore. Por outro lado, essa combinação adiciona um grau de complexidade ao modelo. Por consequência, se perde poder de interpretação em comparação a árvore de decisão.

2.2.3 K-Vizinhos mais Próximos

O algoritmo de *K-Vizinhos mais Próximos* (K-NN) se baseia no princípio de que objetos semelhantes tendem a possuir rótulos semelhantes. A ideia central é prever o valor da variável de saída de um objeto x com base nos K vizinhos mais próximos a x [22]. Por exemplo, a definição de $K = 3$ fará com que o algoritmo demarcar como rótulo de x o rótulo mais comum dentre os 3 objetos mais próximos a x. Em problema de regressão, comumente é utilizada a média entre os vizinhos para definir a saída de x. Esse algoritmo é considerado preguiçoso, uma vez que o processo de treinamento consiste em memorizar o conjunto de dados de treinamento invés de aprender uma função discriminativa dos dados.

A proximidade entre os objetos i e j é definida conforme alguma função de distância $d(i, j)$. Comumente, é empregada a distância Euclidiana. No entanto, outras funções também são utilizadas, a depender do tipo de problema e da natureza dos dados.

Existem algumas características importantes do K-NN que necessitam ser consideradas para se obter um modelo com bom poder preditivo. Entre elas, é necessário definir um valor adequado para K. Valores baixos tornarão o modelo suscetível a ruídos, enquanto que valores altos podem causar uma perda de informações discriminativas. Além disso, é recomendável um pré-processamento dos dados, uma vez que, características que são apresentadas em escalas diferentes causam um impacto desproporcional na distância computada.

2.2.4 Máquina de Vetor de Suporte

O algoritmo *Máquina de Vetor de Suporte* (SVM) busca encontrar um hiperplano no espaço de características que proporcione a melhor separação entre diferentes classes [12]. Em um contexto de um problema de classificação binária, o hiperplano faz uma divisão do espaço em duas regiões, uma para cada classe.

Durante o treinamento, o SVM busca encontrar o hiperplano que maximiza a margem entre os objetos mais próximos de cada classe. Esses objetos são chamados de vetores de suporte. A margem é definida como a distância entre o hiperplano e os vetores de suporte. Ao maximizar a margem, o SVM procura diminuir a chance de ocorrência de um sobre-ajuste.

Na prática, nem sempre é possível encontrar um hiperplano que faça uma separação perfeita entre as classes. Por conta disso, é introduzido o conceito de margem suave [21]. Basicamente, a margem suave permite que alguns objetos estejam dentro da margem ou até mesmo no lado errado do hiperplano. Observe que, com a margem suave, embora o modelo possa cometer mais erros em relação aos dados de treinamento, ele se torna

mais robusto em relação a objetos que são muito discrepantes de seus pares da mesma classe. O grau de liberdade da margem suave é definido pelo parâmetro de regularização C, que determina a compensação entre a maximização da margem e minimização dos erros. Valores altos para C resultam em uma margem mais rígida, isto é, mais rigorosa em relação aos objetos que a violam.

Uma das principais vantagens do SVM é a sua capacidade, em comparação com outros algoritmos, de lidar com dados de alta dimensionalidade, tornando-o adequado para problemas com muitas características. Por outro lado, em grandes conjuntos de dados, o algoritmo possui um alto custo computacional, o que pode tornar a sua utilização inviável.

2.2.5 Rede Neural

A *Rede Neural* é uma técnica de aprendizagem de máquina inspirada no funcionamento do cérebro humano. Basicamente, consiste em vários nós de processamento interconectados, distribuídos em camadas. Uma rede neural é composta por uma camada de entrada, uma ou mais camadas ocultas e uma camada de saída.

A unidade fundamental de uma rede neural é o nó de processamento. O nó é responsável por receber entradas, realizar um processamento interno e gerar uma saída. Um nó pode estar conectado a vários outros nós na camada anterior, da qual ele recebe informações, e também a vários nós na camada posterior, para o qual ele transmite informações. Para toda conexão da rede neural é atribuído um peso. Quando ativo, um nó da rede neural recebe um diferente valor dos dados em cada conexão de entrada. Este valor é multiplicado pelo respectivo peso da conexão. É realizada uma soma dos produtos, e caso o resultado dessa operação seja maior que um limiar, o nó transmite informações para a camada posterior.

Durante o treinamento, são ajustados os valores dos limiares e dos pesos das conexões para prever os dados de entrada de forma consistente. Em todo caso, os dados percorrem todas as camadas da rede, sendo submetidos a várias transformações, até chegar à camada de saída, que realiza a previsão da variável desejada.

As redes neurais são capazes de aprender e extrair características relevantes dos dados por conta própria, através do ajuste dos pesos. Isso permite a captura de relações complexas e não lineares nos dados, o que favorece o seu uso para criar modelos para problemas envolvendo dados com alta dimensionalidade e não linearidade. Por outro lado, o treinamento de uma rede neural, a depender da arquitetura utilizada e do tamanho da entrada, pode ser computacionalmente custoso. Além disso, em muitas situações, compreender a linha de decisão do modelo não é uma tarefa trivial, o que levanta questões éticas e/ou legais a depender da sua aplicação.

2.3 Funções de Avaliação de Modelos de Aprendizagem Supervisionada

Para um modelo de aprendizagem supervisionada, funções de avaliação buscam medir o desempenho e a sua qualidade. No contexto deste trabalho, é proposto um modelo que classifique licitações de acordo com a presença de fraudes, se tratando de um problema de classificação. Nesse sentido, as métricas apresentadas nesta seção se limitam aquelas utilizadas para problemas desse tipo.

2.3.1 Acurácia

A *Acurácia* mede a proporção de amostras classificadas corretamente pelo modelo. Sejam $\hat{y}_i$ e y_i respectivamente o rótulo previsto pelo modelo e o verdadeiro rótulo do i-ésimo objeto do conjunto de dados consistido por n objetos, a acurácia é dada por:

$$\text{Acurácia} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i = y_i) \tag{1}$$

Vale salientar que em conjuntos de dados onde os rótulos estão distribuídos de forma desbalanceada, um valor alto para esta métrica pode ser enganoso. Uma vez que um modelo que classifique todos os objetos como sendo da classe majoritária irá receber um valor considerado alto para essa métrica.

2.3.2 F-Score

Representa uma média harmônica entre a precisão e a revocação (recall) do modelo, onde Precisão $= \frac{VP}{VP+FP}$ e Recall $= \frac{VP}{VP+FN}$. Em ambos os casos, VP é o número de verdadeiros positivos, FP é o número de falsos positivos e FN é o número de falsos negativos. Sendo assim, o F-Score é definido como [50]:

$$\text{F-Score} = \frac{2 \times (\text{Precisão} \times \text{Recall})}{\text{Precisão} \times \text{Recall}} \tag{2}$$

Essa métrica é bastante utilizada em situações onde há um desbalanceamento entre as classes ou quando se deseja encontrar um equilíbrio entre a precisão e a revocação. Uma vez que é considerada tanto a capacidade do modelo em evitar falsos positivos quanto a capacidade de identificar corretamente os verdadeiros positivos.

2.3.3 Área abaixo do curva ROC (AUC)

A *curva ROC* denota a taxa de verdadeiros positivos em função da taxa de falsos positivos para diferentes limiares de classificação.

Por sua vez, a *Área abaixo da curva ROC* é um valor numérico que mede probabilidade de um exemplo positivo aleatório receber uma pontuação maior que um exemplo negativo aleatório [25]. Valores acima de 0,5 indicam que o modelo possui um desempenho melhor que um classificador aleatório. Sendo que quanto mais próximo é o valor desta métrica de 1, melhor é a capacidade do classificador de ter uma alta taxa de verdadeiros positivos e uma baixa taxa de falsos positivos. É considerada uma métrica bastante robusta em casos onde a distribuição entre as classes é desigual.

2.4 Agrupamento

Em termos gerais, *Agrupamento* é um processo que visa unir objetos similares em um mesmo grupo, de forma que cada grupo formado sejam constituídos por objetos similares entre si e dissimilares com os que não pertencem ao grupo. Essa técnica é empregada para extrair informações relevantes sobre dados não rotulados, possuindo um grande leque de aplicações, como por exemplo, a identificação de padrões em diferentes tipos de transporte, análise na tendência de comportamento de clientes na indústria têxtil, e o entendimento de padrões em locais de assistência média [43].

Em um problema de agrupamento, é possível conhecer previamente o número de grupos ou não. No primeiro caso, o problema é denominado de *Problema de Agrupamento* (PA), enquanto que o segundo caso é conhecido como *Problema de Agrupamento Automático* (PAA) [24].

2.4.1 Tipos de Grupos

Seja O o conjunto de objetos a serem agrupados. Existem diferentes tipos de grupos comumente utilizados:

- **Subconjunto:** Define um grupo $C \subset O$;
- **Particionamento:** Divide O em um conjunto P contendo N grupos, sendo $P = \{C_1, C_2, ..., C_N\}$, de forma que sejam satisfeitas as seguintes condições:
  1. $C_j \neq \emptyset$, para $j = 1, 2, ..., N$;
  2. $C_i \cap C_j = \emptyset$, para $i, j = 1, 2, ..., N$ e $i \neq j$;
  3. $\bigcup_{i=0}^{N} C_i = O$;
- **Empacotamento:** Define um conjunto P contendo N grupos, que satisfazem as condições (1) e (2) do Particionamento;

- **Cobertura:** De forma similar aos dois tipos de grupos mencionados anteriormente, define-se um conjunto P contendo N grupos, que cumprem as condições (1) e (3) do Particionamento.

2.4.2 Etapas de um Problema de Agrupamento

Apesar do principal objetivo de um algoritmo de agrupamento ser a demarcação de objetos em diferentes classes, não existe uma técnica em particular considerada a mais apropriada para qualquer problema dessa natureza. Isso implica em uma análise por parte do usuário considerando, entre diversos fatores, a natureza dos dados e o tipo de saída desejado. Em [29] os autores descrevem, de forma geral, um Problema de Agrupamento como um processo envolvendo as seguintes etapas:

1. **Amostragem:** Seleciona um conjunto de n objetos $X = \{x_1, x_2, ..., x_n\}$ nos quais os grupos devem ser encontrados;
2. **Dados:** Observa ou mede as p características dos objetos de X, gerando uma matriz de dados $M_{n \times p}$;
3. **Dissimilaridades:** Computa da matriz M uma matriz de dissimilaridades $D_{n \times n} = (d_{kl})$, que satisfaz as seguintes propriedades: $d_{kl} \geq 0, d_{kk} = 0, d_{kl} = d_{lk}$ para $k, l = 1, 2, ..., n$;
4. **Restrições:** Especificação do tipo de agrupamento desejado;
5. **Critério:** Escolha de um ou mais critérios para expressar a homogeneidade e/ou separação dos grupos a serem obtidos;
6. **Algoritmo:** Projetar um algoritmo para o problema definido;
7. **Computação:** Aplicar o algoritmo à matriz D, obtendo os grupos;
8. **Interpretação:** Aplicar testes para selecionar os melhores grupos obtidos anteriormente e descrever os grupos.

2.5 Modelos de Agrupamento

Em [43], os autores propõem uma classificação de algoritmos de agrupamento com base na medida de proximidade, os dividindo em matriz de distância e matriz de similaridade. Neste trabalho, será utilizada essa classificação para separar modelos de agrupamento que, dentro do nosso conhecimento, podem ser adaptados para encontrar grupos com elementos em sua interseção. Adicionalmente, apresentaremos modelos de agrupamento baseados na edição de arestas.

2.5.1 Matriz de Distância

Modelos que utilizam uma matriz de distância agrupam os objetos de acordo com uma certa função de distância $d(i, j)$, geralmente aplicada a características contínuas e ordinais c. Considerando m como o número de características que descrevem os objetos, a função de distância mais comum é a Euclideana:

$$d(i, j) = \sqrt{\sum_{c=1}^{m}(i_c - j_c)^2}.$$

Entre os algoritmos deste tipo, o *K-Means*, proposto em [41], é um dos mais utilizados. Seu funcionamento consiste em colocar cada objeto $x \in X$ em um dos k grupos a serem criados, de forma que este grupo contenha o centroide mais próximo a x, minimizando a soma das distâncias dos objetos dos grupos aos seus centroides [53]. Um *centroide* é um ponto, real ou não, localizado exatamente no centro de um grupo. Em linhas gerais, o algoritmo segue o seguinte processo:

1. Escolha k objetos de X para serem os centroides iniciais dos grupos;
2. Para os objetos não selecionados anteriormente, os atribua ao grupo com centroide mais próximo;
3. Compute os novos centroides dos grupos;
4. Repita os passos 2 e 3 enquanto pelo menos um objeto mudar de grupo;

Observe que não necessariamente o processo descrito acima garante que a soma das distâncias dos objetos dos grupos aos seus respectivos centroides é a melhor possível. Além disso, o conjunto de grupos resultante é sensível a forma em que os centroides são inicialmente definidos.

Por outro lado, o problema de *p-medianas* busca identificar p locais de origem e assinalá-los a n destinos, de forma que seja minimizada a soma das distâncias entre os destinos e suas respectivas origens [28]. Modelos de programação inteira foram propostos em [23], além do problema possuir diversas meta-heurísticas adaptadas para sua resolução, como por exemplo, Algoritmo Genético [2] e Busca Tabu [9]. Na área de mineração de dados, a heurística mais utilizada para o p-medianas é chamada de *k-medoides* [44], que foi inspirado pelo algoritmo k-means, nele, a partição de X é construída através da escolha de um objeto pertencente a um grupo como seu centroide.

2.5.2 Matriz de Dissimilaridade

Dados categóricos são definidos por serem originados da observação de variáveis categóricas, isto é, variáveis que possuem um conjunto finito e não ordenado de valores possíveis. Formalmente, seja $X = \{x_1, x_2, ..., x_n\}$ um conjunto de n objetos e $A = \{a_1, a_2, ..., a_m\}$ um conjunto de m atributos usados para descrever os objetos de X. O domínio do atributo a_j é denotado por D_{a_j}, onde $D_{a_j} = \{a_j^{(1)}, a_j^{(2)}, ..., a_j^{(n_j)}\}$ e n_j é o número de categorias possíveis para o atributo a_j. Tem-se que para qualquer $1 \leq p \leq q \leq n_j$, ou $a_j^{(p)} = a_j^{(q)}$ ou $a_j^{(p)} \neq a_j^{(q)}$. Adicionalmente, todo objeto $x_i \in X$ pode ser representado como um vetor $[x_{i1}, x_{i2}, ..., x_{i3}]$ [32]. Para medir a distância entre dois objetos $i, j \in X$ é empregada uma função de similaridade

$$\delta(i, j) = \sum_{p=1}^{m} 1[a_p^{(i)} \neq a_p^{(j)}],$$

que mede o número de atributos que são diferentes entre i e j, ou seja, quanto mais atributos idênticos existirem entre os dois objetos, mais próximos eles serão. Para o agrupamento deste tipo de dado, um dos algoritmos mais utilizados é o *K-modes*, proposto em [20], que diverge do K-Means ao utilizar uma função de similaridade para definição do centroide mais próximo.

2.5.3 Modelos de Edição de Grafos

Modelos de Edição de Grafos trabalham com a formação de cliques em suas soluções. Um clique em um grafo é um subconjunto de vértices no qual cada par de vértices distintos é adjacentes. O Problema de Edição de Grupos (CEP) consiste em dado um grafo $G = (V, E)$, transformar G em uma união disjunta de cliques por meio do menor número possível de operações de inserção e remoção de arestas de G. Em [10] é introduzido o problema e elaborado um modelo de programação inteira para o CEP. Mais recentemente, em [8] são apresentados melhorias aos métodos exatos, além de propor das meta-heurísticas para resolução do CEP.

Por sua vez, uma versão não automática do CEP chamada de p-CEP, constrói um particionamento de p cliques, minimizando edições de arestas. Visando a resolução do p-CEP, em [16] é proposta uma abordagem utilizando branch-and-bound. Além disso, em [17] é proposta uma abordagem por meio de branch-and-price além de uma heurística.

Por fim, o problema de edição de biclusters (BEP) consiste em editar um número mínimo de arestas com fins de transformar um grafo bipartido $G = (V_1, V_2, E)$ em uma união de subgrafos bipartidos completos, em outras palavras, na solução cada componente conectado é um biclique. Em [55] são propostas soluções exatas para o problema, enquanto

que em[54] é proposta a utilização da meta-heurística GRASP.

2.6 Funções de Validação de Agrupamentos

A validação de agrupamentos consiste em avaliar a qualidade do particionamento e selecionar o particionamento que melhor se ajusta aos dados fornecidos. Técnicas de validação podem ser divididas em funções de validação externa e interna, que divergem no uso de informações externas, no caso, rótulos de classes [5]. Nas Seções 2.6.1 à 2.6.3 são apresentadas algumas funções para validação de agrupamentos.

2.6.1 Função Silhueta

A *Função Silhueta* mede o quão similar é um objeto em relação aos membros de seu grupo, em comparação a outros grupos [48]. Ela busca um equilíbrio entre homogeneidade e separação dos dados. Considere um $i \in X$ e l_i como o identificador do grupo que contém o objeto i, a função silhueta é definida como

$$s(i) = \frac{b(i) - a(i)}{max\{b(i), a(i)\}} \in [-1, +1], \tag{3}$$

onde $a(i) = \frac{1}{n_{l_i}-1} \sum_{j \in C_{l_i}} d(i,j)$, que consiste na média da distância do objeto i aos outros objetos do grupo. Além disso, $b(i) = \min_{h \neq l_i} \frac{1}{n_h} \sum_{j \in C_h} d(i,j)$, mede a menor distância média de i a todos os objetos em qualquer outro grupo do qual i não faz parte.

Observe que, por esta métrica, $s(i) \approx 1$ indica que o objeto está adequadamente agrupado, ou seja ele se assemelha mais aos membros do seu próprio grupo em comparação aos membros de outros grupos. Por outro lado, $s(i) \approx -1$ aponta que o objeto i está no grupo errado, sendo i mais similar aos elementos de outros grupos. Um valor para a silhueta $s(i) \approx 0$ mostra que o objeto está na fronteira de dois grupos, e possui uma similaridade parecida com objetos de seu grupo e de outros grupos.

2.6.2 Função de Utilidade Categórica

Busca quantificar a probabilidade de dois objetos em um mesmo grupo obtenham os mesmos valores para os atributos. Proposta em [26], a função de utilidade categórica é definida como:

$$UC = \sum_{l=1}^{k} \frac{|c_l|}{n} \sum_{j=1}^{m} \sum_{q=1}^{n_j} [P(a_j^{(q)}|c_l)^2 - P(a_j^{(q)})^2] \tag{4}$$

Onde $P(a_j^{(q)}|c_l) = \frac{|c_{ljq}|}{|c_l|}$, sendo que $|c_{ljq}| = \sum_{i=1,x_{ij}=a_j^{(q)}}^{n} w_{li}$ e $|c_l| = \sum_{i=1}^{n} w_{li}$. Sendo $P(a_j^{(q)}) = \frac{|a_j^{(q)}|}{n}$, onde $|a_j^{(q)}| = |\{x_i|x_{ij} = a_j^{(q)}, x_i \in X\}|$.

O termo dentro dos colchetes mede o aumento no número esperado de valores de atributos que podem ser previstos, levando em consideração a presença do grupo c_l em comparação com a sua ausência. Por sua vez, $\frac{|c_l|}{n}$ faz uma ponderação dos grupos de acordo com seus respectivos tamanhos.

2.6.3 Função de Entropia da Informação

Com base em princípios da teoria da informação e na noção de entropia como medida de resultados de um agrupamento, a *Função de Entropia da Informação* pressupõe que grupos de objetos similares possuem uma entropia menor do que aqueles com objetos dissimilares. A Equação 5 define esta função [7].

$$E = -\sum_{l=1}^{k} |c_l| \sum_{j=1}^{m} \sum_{q=1}^{n_j} P(a_j^{(q)}|c_l) \cdot \log P(a_j^{(q)}|c_l) \tag{5}$$

2.7 Otimização Combinatória

Um *problema combinatório* envolve encontrar agrupamentos, designações, ou ordenações de um conjunto discreto de objetos, de forma que satisfaça certas condições [31]. Adicionalmente, é possível caracterizar diversos problemas combinatórios como problemas de decisão, onde a solução para instância pode ser descrita por um conjunto de condições lógicas.

Por sua vez, um *problema de otimização* pode ser visto como uma generalização de um problema de decisão. Nesse caso, as soluções são avaliadas adicionalmente por uma função objetivo, com fins de encontrar uma solução que possua um valor ótima para a função objetivo dada. Matematicamente, um problema de otimização pode ser descrito por um conjunto de equações da seguinte forma:

$$\text{Minimize} \quad f(x) \tag{6}$$

$$\text{s.a.} \quad g_i(x) \leq b_i, i = 1, 2, ..., n \tag{7}$$

$$l_j \leq x_j \leq u_j, j = 1, 2, ..., m. \tag{8}$$

Onde a Equação 6 é a função objetivo do problema, no qual se procura otimizar. No caso acima, o objetivo é minimizar a função. No entanto, vale salientar que a multiplicação

da função de objetivo por -1 permite transformar um problema de minimização em um problema de maximização, e vice-versa. A Equação 7 representa o conjunto de restrições do problema. Por fim, a Equação 8 define o domínio das variáveis do problema.

Quando a função objetivo e o conjunto de restrições são todas lineares, o problema é chamado de problema de programação linear. Nesse caso, o conjunto de restrições pode ser expresso como um sistema de equações lineares [40].

2.7.1 Branch and bound

Para resolução de problemas de otimização combinatória, um dos métodos mais utilizados é o *Branch and bound* (BB) [38]. Sua ideia principal é empregar uma estratégia de ramificação para explorar o espaço de busca dos possíveis valores das variáveis.

Esse método consiste em uma enumeração de todos os candidatos a solução, chamado de conjunto S. Sendo realizada uma busca pela solução $s \in S$ que maximiza ou minimiza uma dada função objetivo $f(x)$. Para tanto, o algoritmo de forma recursiva divide S em subconjuntos menores, e então otimiza $f(x)$ nesses subconjuntos. Esse processo é chamado de *branching* (ramificação).

Em seguida, é aplicada uma função de limitação que calcula o limite superior da função objetivo. Isso serve para decidir entre a continuação ou não da busca. Esse processo é chamado de *bounding* (limitação) e serve para eliminar subconjuntos em que é possível afirmar que a solução ótima não está contida. Vale salientar que, apesar do processo de limitação acelerar consideravelmente o modelo em comparação com uma busca exaustiva, sua eficácia depende da formulação do problema.

3 REVISÃO DA LITERATURA

Neste Capítulo são apresentados alguns dos trabalhos existentes na literatura que buscaram desenvolver ferramentas para identificação de fraudes em licitações, estando incluídas entre estas fraudes, o conluio entre empresas participantes. Em suma, trabalhos anteriores empregaram técnicas envolvendo a análise estrutural e/ou análise comportamental das licitações. Ao final deste Capítulo, na Seção 3.6, são discutidos os aspectos principais das contribuições de outros autores e suas implicações na elaboração deste trabalho.

A *análise estrutural* serve como um filtro, identificando licitações (ou também grupo de licitações) que são mais suscetíveis à fraudes. Nela, são analisadas questões referentes à características específicas da licitação e do mercado. Por outro lado, a *análise comportamental* cuida de questões sobre o comportamento dos participantes de uma licitação, e visa observar certos aspectos que podem indicar conluio entre os participantes desta.

3.1 Análise do comportamento conjunto de participantes de uma licitação

Em [52], os autores propõem um método probabilístico, baseado na análise do comportamento conjunto de participantes que agem juntos com fins de fraudar uma licitação. Nesse trabalho, foram estudadas as licitações envolvendo a Petrobras, estatal brasileira do setor petrolífero, investigadas no âmbito da Operação Lava Jato.

Para cada licitação, é elaborada por parte da companhia uma estimativa pré-licitação (PTE). Essa métrica estabelece um norte para o preço das propostas submetidas pelos participantes de uma licitação. Em uma proposta honesta, se assume que o licitante faz sua própria estimativa de preço e aplica uma margem que reflete fatores, como por exemplo, estratégia da empresa e condições de mercado. Esses fatores são considerados como difíceis de prever. No entanto, é apontado que um conjunto de propostas de licitantes diferentes pode se comportar de maneira previsível.

Em suma, os autores consideram que dada a natureza aleatória de cada proposta, as diferenças entre as propostas e a estimativa pré-licitação seguem uma distribuição normal com média $\mu = 0$ e um desvio padrão, que teoricamente é difícil de mensurar. No entanto, é assumido arbitrariamente que 90% das propostas honestas ficam a uma distância de até 20% da estimativa pré-licitação. Isso resulta em um desvio padrão $\sigma = 0.12$. Sendo assim, é possível identificar casos suspeitos, pois estes tendem a fugir significativamente da média. O modelo então consiste nos seguintes passos:

1. Computar as probabilidades individuais de uma proposta aleatória possuir valores

inferiores à cada proposta observada i. Essa probabilidade é denominada de $P(i)$ e obtida por meio da função de distribuição acumulada;

2. Calcular a probabilidade conjunta de um conjunto aleatório $B_1, B_2, ..., B_n$ de n propostas serem observados ao acaso. Essa probabilidade é denominada de $P(x)$, obtida plea multiplicação das probabilidades individuais segundo a equação:

$$P(x) = \prod_{i=1}^{n} P(i) \tag{9}$$

3. Comparar se para um conjunto de n propostas e um dado intervalo de confiança $P(x)$ é superior aos valores limites. Os valores limites são computados utilizando a função Gama incompleta:

$$\pi_n(x) = \frac{\Gamma(n, -ln(x))}{(n-1)!}, 0 \leq x \leq 1 \tag{10}$$

4. Se $P(x) > \pi_n(x)$, houve conluio na licitação.

3.2 Estatística descritiva para detecção de conluios

Por meio de estatística descritiva, é possível fazer a análise de variáveis estratégicas para definir se uma ou mais empresas fogem de um comportamento competitivo. No contexto de uma licitação, essas variáveis incluem, preços de propostas e participação de mercado dos competidores. Na literatura, diversos trabalhos se dedicam a propor fórmulas para definir características adicionais em uma licitação. O objetivo é fornecer padrões de conluios com fins de auxiliar o modelo de detecção de conluio. Seja t uma licitação, são definidas as seguintes características [36]:

- *Coeficiente de Variação (CV)*: Obtido pela razão entre o desvio padrão e a média dos valores das ofertas em uma licitação, dados respectivamente por σ_t e μ_t. Essa métrica parte do pressuposto que em licitações fraudulentas, os participantes atuam de forma coordenada, submetendo propostas com fins de aumentar o valor da licitação. Como não é possível exagerar nos preços das ofertas, os fraudadores acabam por limitar a distribuição dos valores das ofertas, reduzindo assim o coeficiente de variação;

$$CV(t) = \frac{\sigma_t}{\mu_t} \tag{11}$$

- *Distância Relativa (RD)*: Basicamente é a razão entre a diferença do valor da menor ($b_{min,t}$) e da segunda menor ($b_{2,t}$) proposta pelo desvio padrão das propostas

perdedoras ($\sigma_{l,t}$) de uma licitação. Possui como finalidade medir assimetrias na distribuição dos valores das propostas. Essa métrica parte do pressuposto que em uma licitação fraudulenta a diferença entre as duas ofertas de menor valor aumenta e a diferença entre as ofertas perdedoras diminui;

$$RD(t) = \frac{b_{2,t} - b_{min,t}}{\sigma_{l,t}} \tag{12}$$

- *Amplitude (SPD)*: Mede a razão entre a diferença da oferta de maior e menor valor pela oferta de menor valor;

$$SPD(t) = \frac{b_{max,t} - b_{min,t}}{b_{min,t}} \tag{13}$$

- *Diferença entre os mínimos (DIFFP):* Similar ao SPD, mede a razão entre a diferença das duas ofertas de menor valor pela oferta de menor valor;

$$DIFFP(t) = \frac{b_{2,t} - b_{min,t}}{b_{min,t}} \tag{14}$$

- *Excesso de Curtose (KURT):* Faz a mensuração de quão juntos estão os valores das propostas próximos da média. No entanto, também é possível utilizar essa métrica pra medir grupos de propostas longe da média. Para seu cálculo é necessário que uma licitação tenha pelo menos quatro propostas submetidas. Sejam n_t o número de propostas submetidas à licitação t e b_{it} o valor da i-ésima proposta submetida, o excesso de curtose é dado pela Equação 15;

$$KURT(t) = \frac{n_t^2 + n_t}{(n_t - 1)(n_t - 2)(n_t - 3)} \sum_{i=1}^{n_t} (\frac{b_{it} - \mu_t}{\sigma_t})^4 - \frac{3(n_t - 1)^3}{(n_t - 2)(n_t - 3)} \tag{15}$$

- *Assimetria (SKEW):* Busca identificar possíveis assimetrias na distribuição dos valores das ofertas;

$$SKEW(t) = \frac{n_t}{(n_t - 1)(n_t - 2)} \sum_{i=1}^{n_t} (\frac{b_{it} - \mu_t}{\sigma_t})^3 \tag{16}$$

- *Teste de Kolmogorov-Smirnov (KSTEST):* Mede a similaridade da distribuição dos valores das ofertas em comparação com outra distribuição, no caso, é feita a comparação com uma distribuição uniforme.

Utilizando dados referentes à licitações na Suíça em [35] são empregadas as características CV e RD para separar dos dados licitações e empresas que apresentam um

padrão suspeito. Para tanto, são selecionadas licitações que apresentam valores baixos para o CV e altos para o RD. Em outro ponto, são estudados se existem empresas regularmente submetendo ofertas para as mesmas licitações. Nesse sentido, é apontado pelos autores que a análise de agrupamento pode ser utilizada para tal tarefa. Todavia, foi optado pela não utilização uma vez que os dados sob disposição pelos autores não se adéquam a esse problema.

Adicionalmente, em [34] e [47], os autores buscam empregar essas características com fins de auxiliar algoritmos de aprendizagem de máquina a processar dados de licitações. Em ambos os casos é constatado que a incorporação dessas características aos dados originais resulta em ganhos na precisão do modelo quanto a detecção de conluios. Mais especificamente em [47] é realizado um estudo envolvendo 11 algoritmos diferentes de aprendizagem de máquina, sendo treinados em 6 conjuntos de dados, cada um de um país diferente, envolvendo cerca de 10 mil licitações e 65 mil ofertas. Nesse trabalho, foram observados resultados superiores para detecção de conluios quando o modelo é treinado com os dados originais em conjunto com as características descritas anteriormente.

3.3 Análise de agrupamentos de licitações para detecção de conluios

Na inexistência de dados passados sobre a existência de carteis, uma abordagem envolvendo agrupamentos é uma alternativa para criação de um modelo de detecção de conluio. Em [18], os autores partem do pressuposto que existem apenas duas formas de propostas, isto é, proposta com e sem conluio. Além disso, assumem que as propostas fraudulentas se diferem bastante das legítimas. Com isso, os autores propõem um método que busca criar dois grupos, um para cada tipo de oferta, o processo proposto envolve quatro etapas:

1. Realizar uma análise de agrupamentos, por meio da divisão dos dados em dois grupos, seguindo a razão entre o preço da oferta vencedora e da estimativa original (PTE). Um dos grupos seria formado por licitações com uma razão alta, possivelmente indicando conluio, e outro com uma razão baixa;

2. Aplicar teste não-paramétricos, que servem para checar se os grupos são significativamente diferentes;

3. Testes de normalidade e simetria, por meio do teste de Komogorov-Smirnov, comparando a distribuição das ofertas dos dois grupos com a distribuição normal. No caso, para o grupo com licitações fraudulentas é esperada uma distribuição assimétrica;

4. Teste de hipótese, que checa se existem diferenças significativas entre diferentes tipos de dados, no caso do artigo, se há diferenças entre licitações de anos consecutivos.

3.4 Testes de Conluios

Uma outra linha seguida por trabalhos anteriores consiste em realizar uma série de testes econométricos para identificação de conluios. Esses testes são baseados em informações sobre as licitações e os participantes destas.

Entre estes, no trabalho de [6] é proposto um método de detecção de conluios em licitações que parte de um modelo de licitação genérico. Esse modelo descreve uma disputa competitiva entre os participantes desta e assume que os licitantes são assimétricos, isto é, seus custos diferem devido a fatores como localização, porte da empresa, ou familiaridade com a legislação vigente.

Além disso, é elencado um conjunto de condições que são suficientes para a distribuição de propostas em uma licitação ser explicada pelo modelo genérico. Sendo assim, é apontado que em situações de conluio, se espera encontrar ofertas correlacionadas onde os membros de um cartel submetem propostas fantasmas, com fins de dar uma aparência de competitividade ao processo. Adicionalmente, pressupõe-se que os custos por si só são suficientes para determinar como as empresas submetem propostas e que as identidades dos competidores de uma empresa não deve mudar a forma que esta faz suas ofertas. Para tanto, os autores argumentam que, em situações de conluio é esperado que os membros do cartel não façam propostas agressivas contra os outros membros deste, ao passo que são agressivos com os não membros do cartel.

Ainda nesse trabalho, com a ajuda de especialistas da área, os autores obtém uma distribuição prévia dos parâmetros do custo estrutural de uma empresa. Com essas informações, é empregado o Teorema de Bayes e as leis da probabilidade condicional para comparar os modelos de licitação com conluio e sem conluio.

Em outro ponto, em [3] são realizados dois testes para detecção de fraudes que buscam explorar uma característica típica de um conluio, que é a diferença nas margens de lucro. O primeiro teste consiste em selecionar participantes cuja atuação na licitação não segue um padrão competitivo. Em seguida, o segundo teste visa estimar o custo para os participantes de uma licitação em cenários com e sem conluio. A detecção de conluio proposta pelos autores se baseia na premissa que em situações de conluio a margem de lucro do vencedor é maior. Portanto, a identificação do conluio se resume a testar a dominância estocástica de primeira ordem, sendo empregados os testes de Kolmogorov-Smirnov e Wilcoxon–Mann–Whitney.

3.5 Identificação de licitações suspeitas por meio da descoberta de regras de associação

Regras de associação (RA) visam encontrar padrões de relacionamento entre objetos que ocorrem em comum em um conjunto de dados. Em [46] por meio da descoberta de RA entre empresas participantes de licitações, é proposto um modelo para identificar licitações suspeitas.

O modelo proposto faz uso do algoritmo Apriori [1]. Sendo que, o processo para sua criação consiste em, gerar a cada iteração i um conjunto de todos os i subconjuntos que atendem ao suporte mínimo da regra. Seja x uma empresa, L o conjunto de licitações e L_x o conjunto de licitações em que x participa, tem-se que $suporte(x) = \frac{|L_x|}{|L|}$. Em seguida, os conjuntos da iteração i são combinados para a próxima iteração $i+1$. Nessa transição os conjuntos não frequentes são descartados. O processo se repete até que não seja possível gerar novos conjuntos de dados. Dado as RA que atendem ao suporte mínimo, com fins de selecionar as regras mais fortes, é aplicada a seguinte função de avaliação à cada RA:

$$F(RA) = I(RA) \cdot M(RA) \tag{17}$$

$I(RA)$ é uma função indicadora para filtragem de regras onde os membros não são proponentes frequentes, assumindo 1 caso o número de licitações que cada empresa de RA tenha participado seja inferior ao quantil 0,95 da distribuição de participação anual de fornecedores e 0 caso contrário. A função indicadora serve para eliminar padrões de associações que podem surgir por conta do envolvimento de grandes fornecedores. Por outro lado, $M(RA)$ mapeia zonas de risco, sendo uma ponderação da chance de vitória de ao menos uma empresa pertencente à RA quando há a atuação conjunta do grupo desta e pela presença de outras concorrentes das empresas da regra. Tem-se então que licitações com alto risco de conluio são caracterizadas por RA contendo membros que registram elevadas probabilidades de vencer a licitação e possui mais concorrentes que a média geral.

Em outro ponto, é proposto um sistema de ranqueamento de empresas envolvidas nas RA selecionadas, denominado de Indicador de Suspeição da Empresa. O ranqueamento é realizado por meio de três componentes:

1. Pontuação pela recorrência da empresa em grupos com alta associação de membros e tendenciosidade de vitória;

2. Recorrência da empresa em grupos com indícios de falsa concorrência;

3. Pontuação pela recorrência da empresa em grupos com atuação concentrada em alguns municípios;

3.6 Discussão

Em suma, a vasta maioria dos trabalhos anteriores na identificação de fraudes se baseiam na análise dos preços das ofertas submetidas pelos participantes de uma licitação.

Abordagens que fazem uso de fórmulas de estatística descritiva buscam identificar por meio de equações, discrepâncias na distribuição dos valores das ofertas de uma licitação. Nessa linha, há os trabalhos que fazem um uso direto dessas métricas [36, 35]. Mais recentemente, vários autores utilizam essas métricas como características adicionais de um modelo de aprendizagem de máquina [34, 47]. Ainda, existe abordagens que faz uso dessas fórmulas como uma etapa em uma análise de agrupamentos de licitações [18] ou para a aplicação de testes econométricos [3, 6].

Em outro ponto, apesar de se basear nos preços, o trabalho de [52] propõe um método probabilístico. Esse método serve para quantificar a chance das propostas tem ocorrido, tanto individualmente quanto em grupo.

Por fim, em relação aos trabalhos anteriores, o caminho tomado em [46] explora as relações entre a ocorrência em comum de empresas em licitações. Para tanto, são utilizadas regras de associação.

Sendo assim, o presente trabalho trilha um caminho novo ao propor o agrupamento de empresas pela participação simultânea em licitações. No entanto, o que está sendo proposto não se trata do primeiro estudo sobre a simultaneidade das participações de empresas em licitações [46]. No caso, a diferença reside no uso de métricas para quantificar essa participação e a utilização dessas como entrada adicional de um modelo de identificação automática de fraudes. Uma vez que, as métricas utilizadas anteriormente nessas situações consistiam na análise dos valores envolvidos nas propostas submetidas a uma licitação.

4 AGRUPAMENTO POR INTERSEÇÃO

Neste capítulo definimos formalmente o Problema de Agrupamento por Interseção (PAI), sendo apresentadas duas variações do problema, que divergem no tipo dos grupos gerados, no caso, são propostas versões para particionamento e cobertura. Para cada versão, é proposto um modelo de programação matemática e um algoritmo para sua resolução. Adicionalmente, é demonstrado que a versão de particionamento do PAI é NP-completo.

Comumente, um modelo de agrupamento consiste em particionar um conjunto de objetos de forma que objetos similares sejam designados a um mesmo grupo, sendo utilizada uma função de distância para determinar a proximidade dos objetos. O modelo de agrupamento proposto neste trabalho diverge desse conceito tradicional no sentido de que: (1) não há uma medida de proximidade entre pares de objetos, uma vez que se mede a proximidade sobre o conjunto completo de objetos pertencentes a um grupo invés de uma composição de métricas; (2) o agrupamento é guiado pela quantidade de recursos que todos os objetos em um grupo possuem simultaneamente. Por esta razão, é proposta uma medida para medir a homogeneidade das soluções encontradas.

4.1 Problema de Agrupamento por Interseção (PAI)

Seja $X = \{1, 2, .., n\}$ um conjunto de n objetos, L o conjunto de recursos compartilhados entre os objetos, $L_i \subseteq L$ o conjunto de recursos utilizados pelos objetos $i \in X$, $P = \{C_1, C_2, .., C_m\}$ um particionamento de X,

$$L_{C_k} = \begin{cases} \bigcap_{i \in C_k} L_i & \text{, se } |C_k| > 1 \\ \emptyset & \text{, caso contrário} \end{cases}$$

o conjunto interseção dos recursos utilizados pelos objetos contidos em uma parte C_k. No PAI deseja-se encontrar um particionamento P que

$$\text{Maximize} \quad \sum_{k=1}^{m} |L_{C_k}| \tag{18}$$

$$\text{s.a.} \quad \bigcup_{k=1}^{m} C_k = X, \tag{19}$$

$$C_k \neq \emptyset, \qquad \forall k \in \{1, 2, ..., m\}, \tag{20}$$

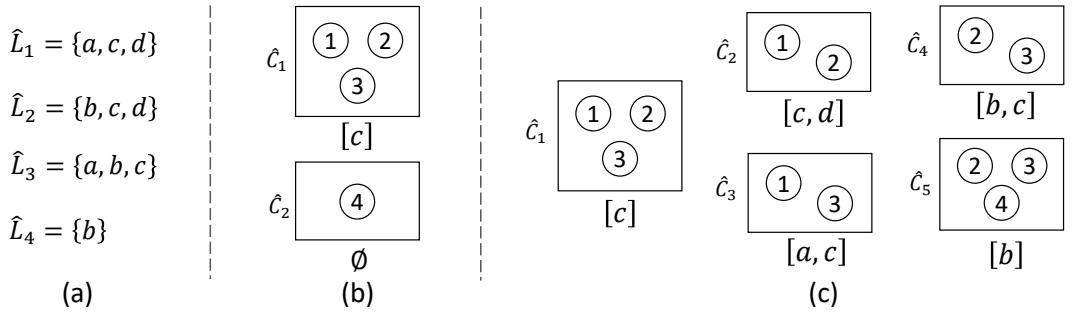
$$C_k \cap C_l = \emptyset, \qquad \forall k, l \in \{1, 2, ..., m\}, k \neq l. \tag{21}$$

$$L_{C_l} \neq L_{C_k}, \qquad \forall k, l \in \{1, 2, ..., m\}, k \neq l. \tag{22}$$

Neste modelo o agrupamento resultante é um particionamento de X obtido pelas restrições (19), (20), (21), que garantem que um objeto pertença a somente um grupo e (22) que proíbe a existência de dois grupos distintos com interseção idêntica.

A Figura 6(a) ilustra uma instância exemplo $\hat{X} = \{1, 2, 3, 4\}$ para o PAI através dos conjuntos de recursos $\hat{L}_i$. Já a Figura 6(b) ilustra um particionamento $\hat{P} = \{\hat{C}_1, \hat{C}_2\}$ de $\hat{X}$. Para essa última, observe que $\hat{C}_1 = \{1, 2, 3\}$ possui apenas o recurso c em sua interseção, e o grupo $\hat{C}_2 = \{4\}$, por ser unitário, não possui recursos em sua interseção. Sendo assim, este particionamento contém apenas um recurso em suas interseções. No entanto, é possível perceber que este não é o particionamento ótimo, pois movendo o objeto 3 de $\hat{C}_1$ para $\hat{C}_2$ obtemos $\hat{L}_{C_1} = \{c, d\}$ e $\hat{L}_{C_2} = \{b\}$ que é uma solução ótima para $\hat{X}$ com três recursos em suas interseções.

Figura 6: Soluções para o PAI. (a) Recursos dos objetos da instância $\hat{X} = \{1, 2, 3, 4\}$, (b) particionamento de $\hat{X}$, (c) cobertura de $\hat{X}$.



Uma variante do PAI para agrupamento por cobertura, denominada de PAI$_{cov}$, propõe a enumeração de todos os agrupamentos de X que atendam as restrições (19) e (20) garantindo que cada objeto pertence a pelo menos um agrupamento de P, e as restrições

$$|L_{C_k}| > 0, \qquad \forall k \in \{1, 2, ..., m\}, |C_k| > 1, \text{ e} \tag{23}$$

$$L_{C_l} \neq L_{C_k} \neq \emptyset, \qquad \forall k, l \in \{1, 2, ..., m\}, C_k \subset C_l, \tag{24}$$

que evitam que o agrupamento formado seja uma simples combinação dos objetos de X. Para o conjunto de restrições (23), cada grupo C_k não unitário ($|C_k| > 1$) deve ter recursos em sua interseção ($|L_{C_k}| > 0$). E para todo grupo $C_l \in P$, o conjunto de restrições (24) garantem que todo grupo $C_k \subset C_l$ só está em P se $L_{C_l} \neq L_{C_k}$ e $L_{C_l} \neq \emptyset$.

A Figura 6(c) ilustra a cobertura $\hat{P} = \{\hat{C}_1, \hat{C}_2, \hat{C}_3, \hat{C}_4, \hat{C}_5\}$ de $\hat{X} = \{1, 2, 3, 4\}$. Todos os objetos de $\hat{X}$ estão em pelo menos um grupo de $\hat{P}$. Note que os grupos $\hat{C}_2$, $\hat{C}_3$ e $\hat{C}_4$ que estão contidos em $\hat{C}_1$ fazem parte de $\hat{P}$, pois suas interseções são distintas de $\hat{L}_{C_1}$. Observe também que o grupo $\{2, 4\} \subset \hat{C}_5$ não está na cobertura $\hat{P}$, pois $\hat{L}_{\{2,4\}} = \hat{L}_{C_5} = \{b\}$.

4.1.1 Silhueta por Interseção

Para identificar a homogeneidade de uma solução P para o PAI, é proposta a mensuração da relação de similaridade entre um objeto $i \in X$ e um grupo $C_k \in P$ por meio da função de distância:

$$d(i, C_k) = \left| L_i \cup L_{C_k} \setminus L_i \cap L_{C_k} \right| \tag{25}$$

Que mede o número de recursos que faltam para que L_i seja igual a L_{C_k}. Com isso, é proposta uma adaptação da função silhueta, proposta em [48], para ser utilizada considerando as características dos grupos gerados pelo PAI. Para tanto, considere $s(i)$ o valor da silhueta de cada objeto $i \in X$ definido como

$$s(i) = \frac{b(i) - a(i)}{max\{b(i), a(i)\}} \tag{26}$$

Onde $a(i) = d(i, C_w)$, sendo a distância do objeto i ao grupo $C_w \subset P$ no qual i está contido em ($i \in C_w$). Além disso, $b(i) = min\{d(i, C_t) \mid C_t \in P, C_t \neq C_w\}$, é a menor distância entre i e todos os grupos C_t no qual i não está contido. O denominador $max\{b(i), a(i)\}$ de $s(i)$ garante que o valor da função se situe dentro do intervalo $[-1, +1]$. Os valores dessa função podem ser interpretados da seguinte forma: os objetos foram perfeitamente agrupados segundo a interseção quando $s(i) = +1$ e inadequadamente agrupados quando $s(i) = -1$. Em um caso excepcional, se $a(i) = b(i) = 0$, então $s(i) = 0$.

Dado um particionamento P, ao computar a média dos valores das silhuetas dos objetos $i \in X$ se obtém a *Silhueta por Interseção Média*:

$$I_{asi}(P) = \frac{1}{n} \cdot \sum_{i \in X} s(i) \tag{27}$$

Analisando a partição P ilustrada pela Figura 6 (b), tem-se que $s(1) = s(2) = s(3) = \frac{1}{3}$ e $s(4) = \frac{1}{2}$, com isso, $I_{asi}(P) = \frac{1}{4} \cdot \frac{3}{2} = +0.375$. Vale salientar que P não é o particionamento ótimo para a instância da Figura 6 (a). Considerando a solução ótima P^* para essa instância, obtida ao mover o objeto 3 de $\hat{C}_1$ para $\hat{C}_2$ é obtido $\hat{L}_{C_1} = \{c, d\}$ e $\hat{L}_{C_2} = \{b\}$, com isso, $s(1) = \frac{3}{4}$, $s(2) = \frac{1}{2}$, $s(3) = \frac{1}{3}$ e $s(4) = 1$, resultado em $I_{asi}(P^*) = \frac{1}{4} \cdot \frac{31}{12} = +0.649$.

4.1.2 Aspectos Teóricos

Proposição 1. *O Problema de Agrupamento por Interseção é NP-Completo*

Demonstração. Para provar a NP-completude do problema, considere uma versão de decisão do PAI, na qual um inteiro positivo t é adicionado à entrada. A pergunta a se responder é: X admite uma partição P tal qual $\sum_{k=1}^{m} |L_{C_k}| \geq t$?

Primeiramente, demonstremos que o PAI $\in$ NP. É trivial observar que dada uma coleção $P = \{C_1, C_2, ..., C_m\}$ de subconjuntos de X, é preciso verificar se P é uma partição de X. Em caso positivo, é necessário verificar se $\sum_{k=1}^{m} |L_{C_k}| \geq t$. Todos esse passos podem ser facilmente resolvidos em tempo polinomial.

A prova de que o problema é NP-Difícil é realizada por meio de uma redução do Problema da Cobertura Exata por 3-conjuntos (X3C) ao PAI. O X3C consiste em, dada uma coleção S, formada por três elementos de um conjunto X_{cov} de tamanho $3q$, para um dado inteiro q, decidir se existe uma sub-coleção S^* de S que contenha q conjuntos tais que cada elemento de X_{cov} apareça em exatamente um conjunto de S^*.

Considere X_{cov} e S, como respectivamente, o conjunto de objetos e a coleção de subconjuntos de X, que constituem a entrada do X3C. A partir disso, mapeamos as entradas a três variáveis: X, L, e L_x, que são, respectivamente, o conjunto de objetos, recursos, e um mapeamento do conjunto de objetos ao conjunto de recursos. Essas três variáveis serão utilizadas como entrada do PAI. O processo de conversão consiste em, para cada subconjunto $A \in S$, criar um recurso r_A e adicioná-lo em L. Então, para cada objeto $x \in A$, é guardada em L_x a informação de que x faz uso de r_A. Adicionalmente, é definido $t = q$. Observe que esta construção leva no máximo $|S| \cdot |X_{cov}|$ passos, e portanto, pode ser realizada em tempo polinomial.

Para concluir a prova, considere uma versão restritiva do PAI no qual cada conjunto C_k da partição P de X necessita conter pelo menos 3 elementos. Se o X3C possui uma resposta SIM, então, claramente existe uma partição P de $X = X_{cov}$ em conjuntos $C_1, \ldots, C_q$ com três elementos cada. Adicionalmente, os elementos em cada conjunto C_k possuem exatamente um recurso em comum, sendo assim $\sum_{k=1}^{m} |L_{C_k}| \geq t = q$.

Reciprocamente, seja $P = \{C_1, \ldots, C_m\}$ a partição de $X = X_{cov}$ de tal forma que: $\sum_{k=1}^{m} |L_{C_k}| \geq t = q$, $|C_k| \geq 3$ para cada k, e $L_{C_i} \neq L_{C_j}$, $i \neq j$. Observe que cada r_A é compartilhado por exatamente três elementos de X_{cov}. Portanto, $|L_{C_k}| \leq 1$ se $|C_k| = 3$ e $|L_{C_k}| = 0$ se $|C_k| > 3$. Todavia, como $\sum_{k=1}^{m} |L_{C_k}| \geq k = q$, são necessários pelo menos q conjuntos C_k com $|L_{C_k}| = 1$. Dado o fato que P é uma partição, pode-se concluir que $m = q$ e todos os conjuntos C_k compartilham exatamente um recurso, o que implica que C_k é um conjunto A da coleção S. Sendo assim, o X3C possui SIM como resposta.

□

Proposição 2. *Existe uma solução ótima P^* para o PAI com $\lceil \frac{n}{2} \rceil$ grupos*

Demonstração. Por definição, todo grupo unitário C_i possui interseção vazia. Além disso, é trivial observar que para toda parte C_k, com $|C_k| \geq 2$, a adição de um novo objeto não aumenta sua interseção, podendo permanecer a mesma ou perder recursos. Por outro lado, para toda parte C_k, com $|C_k| \geq 3$, a remoção de um objeto não diminui sua interseção, podendo permanecer a mesma ou ganhar recursos.

Sem perda de generalidade, considere $P^* = \{C_1^*, C_2^*, ..., C_k^*\}$ uma partição ótima qualquer de X cujas partes estão ordenadas de forma não crescente pelos seus tamanhos, ou seja, $|C_1^*| \geq |C_2^*| \geq .. \geq |C_k^*|$. Seja u o menor índice onde $|C_u^*| = 1$. Em seguida, para todo $i \in \{u, .., k\}$ remova um objeto x de qualquer parte $C_j^* \in \{C_1^*, .., C_{u-1}^*\}$, caso exista $|C_j^*| \geq 3$, e adicione em C_i^*. A saída de x de C_j^* não aumenta sua interseção, pois bastaria criar uma parte só com x para termos uma solução melhor que P^*, o que é impossível pois P^* é ótimo. Do mesmo modo, o objeto x não possui interseção com o único objeto de C_i^*, caso contrário, a parte $C_i^* \cup x$ já estaria em P^*. Logo, a adição de x nas partes unitárias C_i^* não altera suas interseções.

Seja $z = k + 1$, para todo $i \in \{z, ..., \frac{n}{2}\}$ remova um objeto x de qualquer parte $C_j^* \in \{C_1^*, ..., C_{u-1}^*\}$, caso exista $|C_j^*| \geq 3$. Em seguida, crie uma parte unitária C_i^* com $L_{C_i^*} = \emptyset$ e adicione-a em P^*. Agora, repita o procedimento anterior com as partes unitárias $\{C_z^*, ..., C_{\frac{n}{2}}^*\}$ e obtenha um novo particionamento P^*. Esse particionamento conterá partes com no máximo dois objetos cada, e com a mesma interseção do particionamento ótimo original. □

4.2 Soluções Exatas para o PAI e PAI$_{cov}$

O restante deste capítulo se dedica a apresentar os métodos propostos para obtenção de soluções exatas para o problema. As Seções 4.2.1 e 4.2.2 abordam os métodos para o PAI. Por outro lado, as Seções 4.2.3 e 4.2.4 tratam dos métodos para a versão de cobertura do PAI, denominada de PAI$_{cov}$.

4.2.1 Modelo de Programação Inteira para o PAI

Inicialmente, propomos um modelo de programação linear inteira que a cada passo garanta as seguintes decisões discretas para os grupos formados:

- Contabiliza a quantidade de recursos contidos na interseção de cada grupo;
- Garante que o conjunto interseção de um grupo unitário seja vazio;
- Todo grupo seja único na solução;

- Cada objeto só pode pertencer a um grupo.

Primeiro, definimos o conjunto $K = \{1, ..., m\}$ como os índices dos possíveis grupos de P. Pela Proposição 2, o número máximo de grupos é $m = \lceil \frac{n}{2} \rceil$. Definimos a variável binária $l_{rk} = 1$ para indicar que o recurso r pertence a interseção de recursos do grupo k ($r \in L_{C_k}$), e igual a 0 em caso contrário. Sendo assim, a função objetivo (18) pode ser reescrita como

$$\text{Maximize} \sum_{r \in L} \sum_{k \in K} l_{rk}. \tag{28}$$

Adotamos a notação $\bar{L}_i = L \setminus L_i$ como o conjunto de recursos que o objeto i não utiliza. Além disso, definimos as variáveis de decisão: $x_{ik} = 1$ indicando que o objeto i está contido no grupo k, sendo igual a 0 em caso contrário e $s_k = 1$ indicando que o grupo k possui pelo menos dois objetos, sendo igual a 0 em caso contrário.

Os recursos r só podem pertencer a interseção de um grupo k se $s_k = 1$ ($|C_k| > 1$) e quando todos os objetos i contidos no grupo k ($x_{ik} = 1$) utilizem o recurso r ($r \in L_i$). Para contabilizar apenas estes recursos propomos o conjunto de inequações

$$l_{rk} + x_{ik} \leq s_k, \qquad \forall k \in K, \forall i \in X, \forall r \in \bar{L}_i. \tag{29}$$

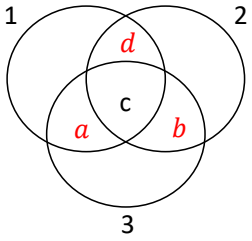
Como a função objetivo de maximização (28) procura colocar toda variável $l_{rk} = 1$, as inequações (29) fazem $l_{rk} = 0$ para todo recurso caso $s_k = 0$, pois $l_{rk} + x_{ik} \leq 0$. Já quando $s_k = 1$ e algum objeto i, contido no grupo k ($x_{ik} = 1$), não possua o recurso r ($r \in \bar{L}_i$) então temos $l_{rk} + 1 \leq 1$, logo, $l_{rk} = 0$.

Retornando à instância $\hat{X} = \{1, 2, 3, 4\}$ da Figura 6(a) extraímos o grupo $\hat{C}_k = \{1, 2, 3\}$ cujos conjuntos de recursos de seus objetos estão ilustrados no diagrama de Venn da Figura 7(a), neste diagrama podemos observar que apenas o recurso c está na interseção de $\hat{C}_k$. Como $s_k = 1$ então as inequações (29) ilustradas na Figura 7(b) foram geradas para proibir que os recursos a, b e d pertençam à interseção de $\hat{C}_k$, por exemplo, como $d \in \bar{L}_3$ a inequação $l_{dk} + x_{3k} \leq 1$ é gerada, e como $x_{3k} = 1$ então $l_{dk} = 0$.

Considerando o grupo unitário $\hat{C}_k = \{4\}$ então por definição $\hat{L}_{C_k} = \emptyset$, entretanto, as inequações (29) não proíbem que os recursos $r \in \hat{L}_4$ façam parte de $\hat{L}_{C_k}$. Sendo assim, para todo grupo $k \in K$ temos

$$|L| \cdot s_k \geq \sum_{r \in L} l_{rk}, \tag{30}$$

Figura 7: Inequações que permitem apenas o recurso c na interseção de $\hat{C}_k = \{1, 2, 3\}$. (a) Diagrama de Venn dos recursos de $\hat{C}_k$, (b) Inequações (29) que proíbem os recursos a, b e d de pertencerem à interseção de $\hat{C}_k$.



(a)

$$l_{dk} + x_{3k} \leq 1$$
$$l_{ak} + x_{2k} \leq 1$$
$$l_{bk} + x_{1k} \leq 1$$

(b)

logo, se $|C_k| \leq 1$ então $s_k = 0$ e assim temos a inequação $0 \geq \sum_{r \in L} l_{rk}$, logo, nenhum recurso $r \in L$ estará contido em L_{C_k}.

Para garantir a consistência na formação do conjunto L_{C_k} de cada grupo, adicionamos a notação $\bar{X}_r = \{i \in X \mid r \notin L_i\}$, que representa o conjunto de objetos que não usam o recurso r, e as inequações

$$l_{rk} \geq s_k - \sum_{i \in \bar{X}_r} x_{ik}, \qquad \forall k \in K, \forall r \in L, \tag{31}$$

que garantem que todo recurso r estará na interseção do grupo k se $s_k = 1$ e nenhum objeto $i \in \bar{X}_r$ esteja no grupo k ($\sum_{i \in \bar{X}_r} x_{ik} = 0$).

Como a função objetivo 28 é de maximização, temos que forçar $s_k = 0$ quando o grupo k tiver $|C_k| \leq 1$. Para isso, introduzimos a variável de decisão $c_k = 1$ indicando que o grupo k não está vazio, sendo igual a 0 em caso contrário. Com isso, propomos para cada grupo $k \in K$ as inequações

$$c_k \leq \sum_{i \in X} x_{ik}, \tag{32}$$
$$n \cdot c_k \geq \sum_{i \in X} x_{ik}, \tag{33}$$
$$s_k \leq \sum_{i \in X} x_{ik} - c_k. \tag{34}$$

Os conjuntos de inequações (32) e (33) garantem que se nenhum objeto i estiver no grupo k então $c_k = 0$, e se pelo menos um objeto i estiver no grupo k então $c_k = 1$. O conjunto de inequações (34) garante que se não houver objetos no grupo k ($\sum_{i \in X} x_{ik} = 0$) e como $c_k = 0$ então $s_k \leq 0$, e se houver apenas um objeto no grupo k ($\sum_{i \in X} x_{ik} = 1$) e como $c_k = 1$ então $s_k \leq 1 - 1$, entretanto, se $\sum_{i \in X} x_{ik} > 1$ e como $c_k = 1$ então s_k fica livre.

Propomos para cada objeto $i \in X$ que

$$\sum_{k \in K} x_{ik} \leq 1, \tag{35}$$

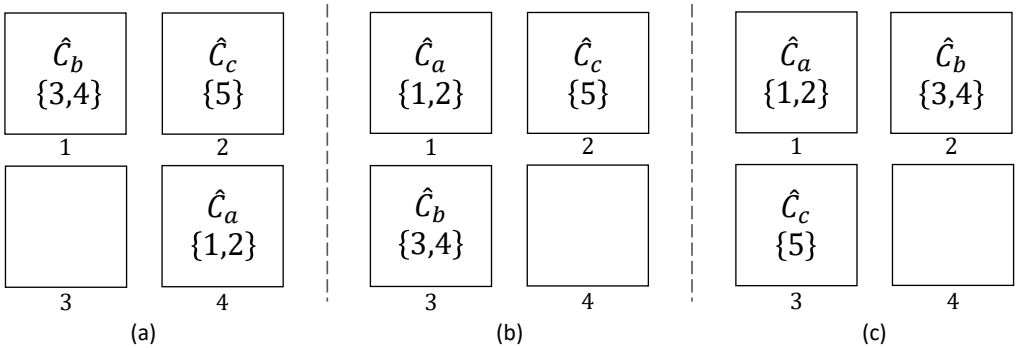
ou seja, o objeto i está em no máximo um grupo de K.

A Figura 8 apresenta uma solução exemplo $\hat{P} = \{\hat{C}_1, \hat{C}_2, \hat{C}_3\}$, sendo $L_{\hat{C}_1} = \{1, 2\}$, $L_{\hat{C}_2} = \{3, 4\}$ e $L_{\hat{C}_3} = \{5\}\}$, ilustrada em três representações válidas e distintas da variável l_{rk}. Por exemplo, na Figura 8(a) temos $\hat{P}$ representada por $l_{3,1} = l_{4,1} = 1$, $l_{5,2} = 1$ e $l_{1,4} = l_{2,4} = 1$, já na Figura 8(b) temos $\hat{P}$ representada por $l_{1,1} = l_{2,1} = 1$, $l_{5,2} = 1$ e $l_{3,3} = l_{4,3} = 1$. Isto é ocasionado pela simetria de soluções do modelo e para evitar este fenômeno propomos o conjunto de inequações

$$\sum_{r \in L} \frac{1}{r} \cdot l_{rk_1} \geq \epsilon \cdot s_{k_1} + \sum_{r \in L} \frac{1}{r} \cdot l_{rk_2}, \qquad \forall k_1 \in \{1, ..., m-1\}, k_2 = k_1 + 1, \tag{36}$$

que procura atribuir ao grupo k_1 uma sequência de índices de recursos contidos em sua interseção cuja soma de suas frações correspondentes seja maior que a soma das frações correspondentes à sequência de índices dos recursos do grupo $k_2 = k_1 + 1$, e por transitividade garantir que toda sequencia de índices dos recursos nas interseções de cada grupo sejam diferentes. Atribuímos um valor $\epsilon = \frac{1}{10 \cdot |L|}$ pequeno o suficiente para ser menor que qualquer fração $\frac{1}{r}$ associada a um recurso $r \in L$.

Figura 8: Três representações simétricas da solução $\hat{P} = \{\hat{C}_a, \hat{C}_b, \hat{C}_c\}$.



No exemplo da Figura 8(a), considerando $k_1 = 3$, temos $l_{1,4} = 1$, $l_{2,4} = 1$ e $s_{k_1} = 3$, logo, do conjunto de inequações (36) temos a formação da inequação $0 \geq \frac{1}{1} \cdot l_{1,4} + \frac{1}{2} \cdot l_{2,4}$, obtendo $0 \geq \frac{3}{2}$, logo, uma solução inviável.

Na solução da Figura 8(b), considerando $k_1 = 2$, $k_2 = 3$ e que $s_{k_1} = 1$, temos a formação da inequação $\frac{1}{5} \cdot l_{5,2} \geq \epsilon \cdot s_2 + \frac{1}{3} \cdot l_{3,3} + \frac{1}{4} \cdot l_{4,3}$, obtendo $\frac{1}{5} \geq \frac{7}{12}$, logo, uma solução inviável. Sendo assim, a solução ilustrada na Figura 8(c) que atende completamente ao conjunto de restrições (36) é a única representação válida para a partição $\hat{P}$.

Por fim, com o objetivo de aumentar a eficiência do modelo, adicionamos as inequações que forçam o grupo $k_2 = k_1 + 1$ ficar vazio caso $c_{k_1} = 0$:

$$c_{k_1} \geq c_{k_2}, \qquad \forall k_1 \in \{1, ..., m-1\}, k_2 = k_1 + 1, \tag{37}$$

Colocando tudo junto temos a seguinte formulação para o problema:

$$\begin{aligned}
PAI = \text{Maximize} & \sum_{r \in L} \sum_{k \in K} l_{rk} & \\
\text{s.a.} \quad l_{rk} + x_{ik} &\leq s_k, & \forall k \in K, \forall i \in X, \forall r \in \bar{L}_i, \\
|L| \cdot s_k &\geq \sum_{r \in L} l_{rk}, & k \in K, \\
l_{rk} &\geq s_k - \sum_{i \in \bar{X}_r} x_{ik}, & \forall k \in K, \forall r \in L, \\
c_k &\leq \sum_{i \in X} x_{ik}, & k \in K, \\
n \cdot c_k &\geq \sum_{i \in X} x_{ik} & k \in K, \\
s_k &\leq \sum_{i \in X} x_{ik} - c_k, & k \in K, \\
\sum_{k \in K} x_{ik} &\leq 1, & i \in X, \\
\sum_{r \in L} \frac{1}{r} \cdot l_{rk_1} &\geq \epsilon \cdot s_{k_1} + \sum_{r \in L} \frac{1}{r} \cdot l_{rk_2}, & \forall k_1 \in \{1, ..., m-1\}, k_2 = k_1 + 1, \\
c_{k_1} &\geq c_{k_2}, & \forall k_1 \in \{1, ..., m-1\}, k_2 = k_1 + 1.
\end{aligned}$$

É possível que ao final da execução deste modelo seja formado um agrupamento solução P que possua um subconjunto de objetos $S \subset X$ que não estejam em nenhum grupo k por não possuírem interseção entre si. Para garantir que a solução P seja um particionamento de X, fazemos $P = P \cup \{i \mid i \in S\}$, ou seja, adicionamos em P um novo grupo de interseção vazia com todos os objetos de S.

4.2.2 Algoritmo Branch and Bound para o PAI

Da Proposição 2 definimos um algoritmo *branch and bound* para resolução do PAI chamado *PartitionTree*. A ideia principal dele é selecionar o subconjunto com a maior soma de interseções de um conjunto de partes candidatas de forma que este subconjunto seja uma solução do PAI.

Sejam $I(P) = \sum_{c \in P} |L_c|$ o valor da solução do particionamento P, $|P|$ a quantidade de partes contidas em P e $C_{cand} = \{\{i, j\} \mid \forall i, j \in X, i \neq j \text{ tal que } L_i \cap L_j \neq \emptyset\}$ a lista

de partes candidatas que contém dois objetos com interseção não vazia. C_{cand} é ordenado de forma não crescente pelo tamanho de suas interseções e fornecida para a execução $P^* = PartitionTree(C_{cand}, 0, \{\}, \{\})$. Caso existam objetos em X que não estão em P^*, então $P^* = P^* \cup \{x \mid x \in X \text{ e } x \notin P^*\}$, tornado assim P^* uma partição ótima de X.

Algoritmo 1: PartitionTree(C_{cand}, i, $\hat{P}$, P^*)

```
1  se I(P̂) > I(P*) então
2  |  P* ← P̂
3  fim
4  enquanto i ≤ |C_cand| faça
5  |  c ← C_cand[i]
6  |  se I(P̂) + |L_c| · (⌈n/2⌉ − |P̂|) ≤ I(P*) então
7  |  |  retorna
8  |  fim
9  |  se os objetos e a interseção de c ∉ P̂ então
10 |  |  P* ← PartitionTree(C_cand, i + 1, P̂ ∪ c, P*)
11 |  fim
12 |  i ← i + 1
13 fim
14 retorna P*
```

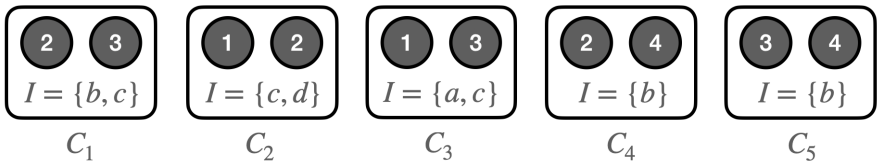
O algoritmo *PartitionTree* é recursivo e, nas iterações das linhas 4-13, visita todos os possíveis agrupamentos formados pelas partes de C_{cand} numa abordagem de busca em profundidade. O teste da linha 1 garante que P^* retorne o melhor agrupamento para o valor de $I(.)$.

Observe que como C_{cand} está ordenado de forma não crescente pelo tamanho de suas interseções, se fôssemos construir uma solução a partir do i-ésimo elemento dessa lista seria possível obter no máximo uma solução com valor $\lceil \frac{n}{2} \rceil \cdot |L_{C_{cand}[i]}|$. Com base nisso, o teste da linha 6 estima um valor máximo a ser alcançado para o $\hat{P}$ corrente considerando o tamanho da interseção do melhor candidato atual e o número de grupos que ainda podem ser adicionados. Se a estimativa for pior do que o valor atual de P^*, então esta solução $\hat{P}$ é descartada, uma vez que ela não é capaz de melhorar a melhor solução encontrada até o momento. Por fim, o teste da linha 9 garante a propriedade de particionamento na construção de $\hat{P}$ corrente.

Para ilustrar o funcionamento do *PartitionTree*, considere a instância da Figura 6 (a). Com base nas relações entre empresas e licitações é a construída a lista de candidatos C_{cand} dessa instância, ilustrada na Figura 9, onde cada grupo é representado por seus membros e sua respectiva interseção.

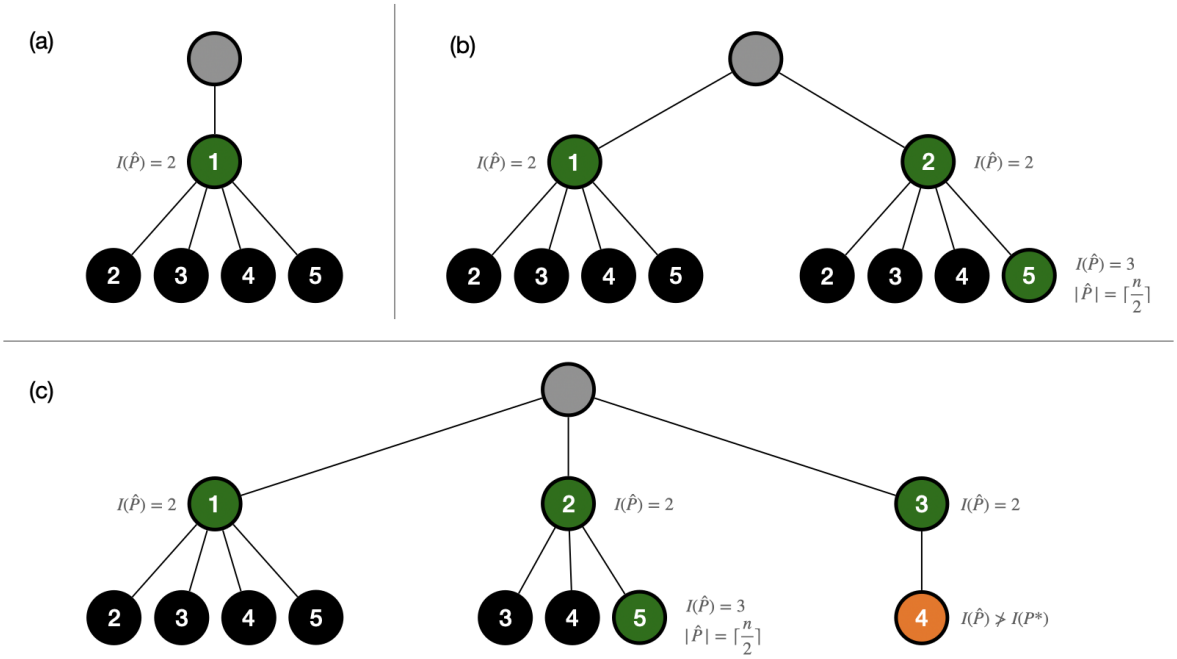
Como está sendo tratada uma instância com 4 empresas, devem ser selecionados pelo algoritmo até 2 grupos de C_{cand}. A Figura 10 ilustra alguns dos passos percorridos

Figura 9: Conjunto de partes candidatas da instância da Figura 6 (a).



pelo algoritmo para construção da solução da instância representada na Figura 6 (b). Para efeitos didáticos, o processo é ilustrado como uma árvore. Cada nó da árvore é denotado pelo índice do grupo em C_{cand} considerado pelo nó. O caminho da raiz ao nó é uma possível solução. Nós pretos e laranjas indicam soluções inviáveis, por violarem, respectivamente, a condição da linha 9 e 6.

Figura 10: (a-c) Etapas da construção da solução do algoritmo branch and bound sobre a instância da Figura 6 (a).



O algoritmo inicia na Figura 10 (a) pela raiz. Nesse ponto, as partições P^* e $\hat{P}$ não contém um grupo sequer. O primeiro grupo avaliado de C_{cand} é o C_1, que possui dois recursos como interseção. A solução $\hat{P} = \{C_1\}$ atende a condição das linhas 6 e 9, o que permite a criação de um nó, denotado pelo nó verde com número 1. O algoritmo então desce na árvore para avaliar novos grupos a serem adicionados. No entanto, nenhum grupo de C_{cand} atende a condição da linha 9. Isso faz o algoritmo encontrar sua primeira solução $P^* = \{C_1\}$. Em seguida, ele sobe a árvore em direção a raiz. O próximo candidato a se avaliar é C_2. Como a partir deste ponto é possível obter uma solução melhor que o P^* atual, o algoritmo passa a avaliar um nó onde $\hat{P} = \{C_2\}$, ilustrado pela Figura 10 (b). A seguir, serão avaliados os outros grupos de C_{cand}. A adição de C_3 e C_4 geram soluções inviáveis. Por outro lado, é possível adicionar C_5 a solução. A adição de C_5 a $\hat{P}$ cria

um solução com o mesmo número de grupos máximo para a instância. Logo, o algoritmo volta a percorrer o caminho até a raiz, desta vez, tem-se que $P^* = \{C_2, C_5\}$ e $I(P^*) = 3$. Nesse momento, $\hat{P} = \{\}$ e o próximo grupo que pode ser considerado é o C_3, que contém 2 recursos na sua interseção. Como a melhor solução possível a partir de C_3 é 4, o algoritmo adiciona esse grupo ao conjunto $\hat{P}$ na Figura 10 (c) e avança na lista de candidatos. No entanto, o próximo elemento de C_{cand} possui apenas 1 recurso em sua interseção, o que torna o prosseguimento do algoritmo desnecessário. Isso ocorre porque, como C_{cand} está ordenado de forma não crescente, não é possível após esse ponto encontrar um grupo em C_{cand} após C_4 cuja interseção seja maior que 1 e, assim, permita $I(\hat{P}) > I(P^*)$. Sendo assim, o algoritmo encerra com a solução $P^* = \{C_2, C_5\}$, que possui 3 recursos na soma das interseções.

4.2.3 Modelo de Programação Inteira para o PAI$_{cov}$

O modelo de programação linear inteira proposto para o PAI$_{cov}$ procura listar todos os agrupamentos da cobertura P tomando as seguintes decisões discretas:

- Grupos não unitários possuem recursos em sua interseção;
- Cada objeto está em pelo menos um grupo;
- Todas as interseções dos grupos são diferentes.

Adotando todos os parâmetros e variáveis de decisão do modelo de particionamento temos o seguinte modelo para criar uma cobertura:

$$PAI_{cov} = \text{Maximize} \sum_{r \in L} \sum_{k \in K} l_{rk}$$

$$\text{s.a.}, (29), (30), (31), (32), (33), (34), (36), (37),$$

$$\sum_{r \in L} l_{rk} \geq s_k, \qquad \forall k \in K, \qquad (38)$$

$$\sum_{k \in K} x_{ik} \geq 1, \qquad \forall i \in X, \qquad (39)$$

$$l_{rk}, x_{i,k}, c_k, s_k \in \{0, 1\}, \qquad \forall k \in K, \forall r \in L, \forall i \in X.$$

O conjunto de restrições (38) garantem que se o grupo k tiver mais de um objeto ($s_k = 1$) então sua interseção terá pelo menos um recurso ($\sum_{r \in L} l_{rk} \geq 1$). Já as restrições (39), que substituíram as restrições de particionamento (35), garantem que cada objeto esteja em pelo menos um grupo da solução. Por fim, o conjunto de restrições (36) garantem que o conjunto interseção de cada grupo seja único.

Limite superior para a dimensão m

Para o modelo PAI_{cov}, a quantidade máxima possível de grupos considerando todos os subconjuntos de X menos o conjunto vazio e os conjuntos unitários, cujos objetos já estão contidos nos outros conjuntos, é $m = 2^n - (n+1)$, o que torna este modelo bastante ineficiente. Entretanto, é possível encontrar um limite superior menor, para isto, considere o grafo $G = (V, E)$, onde $V = X$ e $E = \{i, j \in V \mid L_i \cap L_j \neq \emptyset\}$, o conjunto Q formado pelas cliques maximais do grafo G, e $n(q)$ a quantidade de vértices contidos na clique $q \in Q$. Logo, temos que

$$m = \sum_{q \in Q} 2^{n(q)} - (n(q) + 1).$$

4.2.4 Algoritmo Enumerativo para o PAI_{cov}

É proposto um algoritmo enumerativo, denominado *IntersecForest*, para resolução do PAI_{cov}. O algoritmo emprega uma estratégia *bottom-up*, isto é, inicia-se com grupos unitários e com o auxílio do algoritmo recursivo $IntersecTree$ busca adicionar membros a esse grupo enquanto sua interseção formada não se torne vazia ou já não tenha aparecido na cobertura. Ao fim da execução do algoritmo *IntersecForest* o resultado é uma floresta contendo para cada nó de cada árvore uma 3-upla (i, C, I), que são respectivamente, o objeto i visitado no nó, o grupo de objetos C formado pela adição de i e a sua interseção de recursos I associada.

Para todo nó folha (f, C_f, I_f) de cada árvore o grupo C_f é adicionado a cobertura P. Além disso, para todo nó não folha (t, C_t, I_t) o grupo C_t é adicionado em P se $C_t \neq \emptyset$ e para todo nó filho (j, C_j, I_j) temos $C_t \neq C_j$. Para esse último caso, o nó t é denominado de *folha de interseção*.

O algoritmo $IntersecForest$, descrito pelo Algoritmo 3, pretende enumerar toda a cobertura P da instância (X, L). Para tanto, é construída uma árvore de interseção enraizada em cada objeto $i \in X$. A construção é realizada por meio do algoritmo $IntersecTree$, que adiciona à cobertura P os grupos ainda não descobertos que contém i como membro e cuja interseção não está contida em I_{cov}. Observe que, como na raiz o grupo é formado por apenas um objeto, então sua interseção é vazia. Sendo assim, o nó raiz passado como parâmetro é inicializado como $root \leftarrow (i, \{i\}, \emptyset)$.

Por outro lado, o algoritmo *IntersecTree*, descrito pelo Algoritmo 3, recebe os parâmetros: (X, L) que é a instância a ser resolvida, *no* representa a 3-upla do nó visitado numa busca em profundidade, P é o conjunto de grupos que formarão a cobertura a ser construída e I_{cov} contém a lista de todas as interseções formadas pelos grupos de P. Observe que I_{cov} é uma estrutura empregada para evitar a criação de grupos repetidos em P.

Algoritmo 2: IntersecForest(X, L)

```
1  P ← ∅
2  I_cov ← ∅
3  para cada i ∈ X faça
4  |  root ← (i, {i}, ∅)
5  |  IntersecTree(X, L, root, P, I_cov)
6  fim
7  retorna P
```

Algoritmo 3: IntersecTree(X, L, no, P, I_{cov})

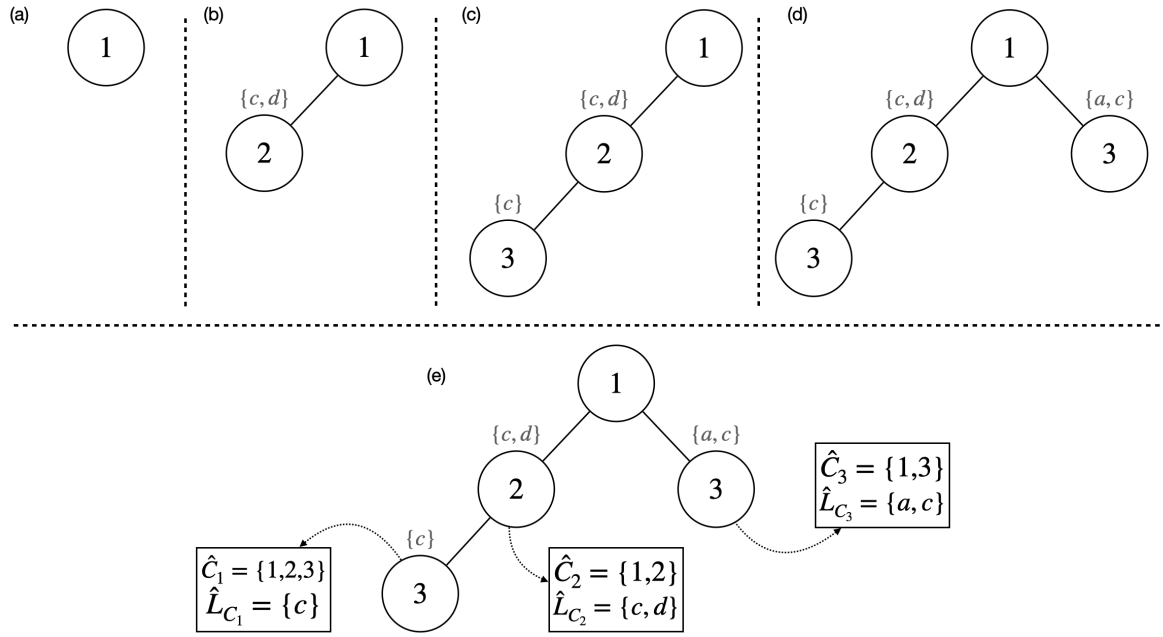
```
1  nFilhos ← 0
2  folhaIntersecao ← verdadeiro
3  para cada j ∈ X, j > no.i faça
4  |  se |no.C| > 1 então  I_filho ← no.I ∩ L_j
5  |  senão  I_filho ← L_no.i ∩ L_j
6  |  se I_filho ≠ ∅ e I_filho ∉ I_cov então
7  |  |  no_filho ← (j, no.C ∪ j, I_filho)
8  |  |  IntersecTree(X, L, no_filho, P, I_cov)
9  |  |  nFilhos ← nFilhos + 1
10 |  |  se I_filho = no.I então
11 |  |  |  folhaIntersecao ← falso
12 |  |  fim
13 |  fim
14 fim
15 se no.I ≠ ∅ e (nFilhos = 0 ou folhaIntersecao = verdadeiro) então
16 |  P ← P ∪ no.C
17 |  I_cov ← I_cov ∪ no.I
18 fim
```

As linhas 1-2 inicializam variáveis auxiliares. Em seguida, o laço das linhas 3-14 realiza a criação dos filhos do nó corrente de forma recursiva. Para a criação de um filho são apenas considerados os objetos que possuem um índice maior ao objeto corrente e cuja interseção com o grupo corrente não seja vazia e não esteja contida em I_{cov}. Por fim, nas linhas 15-17, caso o nó corrente seja um nó folha ou folha de interseção, o seu grupo é adicionado em P e sua interseção em I_{cov}.

A Figura 11 ilustra o funcionamento do algoritmo *IntersecTree* ao construir a árvore enraizada pelo objeto 1 da instância ilustrada na Figura 6(a). Para efeitos de simplificação, cada nó é enumerado pelo índice do objeto visitado. Acima de cada nó é apresentada a interseção do seu grupo. Os membros do grupo do nó correspondem aos

objetos visitados entre o menor caminho do nó em questão até a raiz da árvore.

Figura 11: (a - d) Etapas de construção da árvore para a instância exemplo da Figura 6 enraizada sobre o objeto 1, (e) árvore resultante com os respectivos grupos obtidos.



Na Figura, são ilustrados, respectivamente, os seguintes passos:

(a) A construção é iniciada chamando o algoritmo sobre o objeto 1;

(b) Ao iterar entre os objetos de índice maior que 1, o primeiro filho criado pra este é o nó 2, com interseção $\{c, d\}$. Observe que, por realizar uma construção por profundidade, o algoritmo passa a avaliar a criação de filhos para o nó 2;

(c) É criado o nó 3 (filho de 2), que forma o grupo $\{1, 2, 3\}$ com interseção $\{c\}$. Nesse ponto, não é possível expandir o grupo. Logo, esse nó é uma folha e define o grupo $\hat{C}_1$. O algoritmo retorna a recursão ao nó 2. Entretanto, como nesse nó não há como criar novos filhos com interseção não vazia e todos os filhos possuem interseção diferente, o nó 2 é uma folha de interseção e define o grupo $\hat{C}_2 = \{1, 2\}$. Em seguida, o algoritmo retorna a raiz;

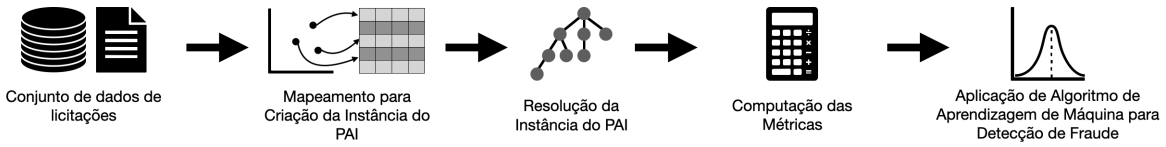
(d) Na raiz, é possível criar mais um filho, justamente o nó 3 (filho de 1). Ao descer para esse nó, tem-se que não é possível criar mais filhos. Logo, ele é um nó folha que define o grupo $\hat{C}_3 = \{1, 3\}$. O algoritmo então volta a subir na árvore. Ao chegar na raiz é constatada a impossibilidade de criar novos filhos, o que leva ao seu encerramento;

(e) Os grupos encontrados pela árvore e retornados pelo algoritmo são ilustrados com seus respectivos objetos e interseções.

5 CLASSIFICAÇÃO DE FRAUDES EM LICITAÇÕES PÚBLICAS

Neste capítulo é apresentada a metodologia empregada para criação de um modelo para classificação de fraudes em licitações públicas. Propomos agrupar as empresas com base na participação concomitante em licitações fraudulentas. Com essa finalidade, utilizamos o PAI, e em seguida, computamos métricas para descrever os grupos gerados que servem como novas características de entrada do classificador. A Figura 12 ilustra as etapas envolvidas no sistema proposto, sendo cada etapa detalhadas com maior ênfase nas Seções seguintes.

Figura 12: Etapas para classificação de fraudes em licitações



Dado um conjunto de dados de licitações como entrada, a primeira etapa (Seção 5.1) do processo consiste no mapeamento desses dados em estruturas específicas do PAI, o que fornece um conjunto de grupos P, compostos por empresas que possuem licitações em comum. Em seguida, a instância criada é resolvida pelo PAI. A terceira etapa (Seção 5.2) consiste na computação de métricas associados aos grupos formados com fins de quantificar a chance desta participação conjunta ser superior ao acaso indicando uma possível formação de conluio. A quarta etapa (Seção 5.3) utiliza essas métricas como entrada adicional para a construção de um classificador de licitações fraudulentas a partir de algoritmos de aprendizado de máquina.

5.1 Mapeamento

As bases de dados das licitações públicas utilizadas neste trabalho possuem formato tabular, onde cada linha corresponde a uma proposta de uma empresa em uma licitação, contendo entre diversas informações, os identificadores da empresa e da licitação associada. Extraímos a lista de empresas que deram lances e mapeamos no conjunto de objetos X. A lista de licitações é mapeada no conjunto de recursos L. Em cada linha encontramos uma relação entre uma empresa e e uma licitação l, que é mapeada na lista L_i dos recursos utilizados pelo objeto i, logo, fazemos $L_e = L_e \cup l$ para cada relação da base. Com a entrada (X, L, L_i) formada, podemos utilizar uma das ferramentas de resolução do PAI e PAI_{cov} para nos fornecer o conjunto de grupos P necessário para a análise dos grupos formados.

5.2 Métricas

5.2.1 Métricas para Particionamento

O Teorema de Bayes foi demonstrado por diversos autores ser bastante eficaz em problemas de classificação. Neste trabalho, propomos seu uso para computar a probabilidade condicionada de uma licitação l ser fraudulenta caso a lista de empresas E participe desta licitação.

Do Teorema de Bayes e da lei da probabilidade total, a probabilidade de uma licitação l, cujas empresas $E = \{e_1, ..., e_m\}$ lançaram proposta, pertencer a categoria c é:

$$Pr(l = c \mid E) = \frac{Pr(E \mid l = c) \cdot Pr(l = c)}{\sum_{k \in \{fraude, legit\}} Pr(E \mid l = k) \cdot Pr(l = k)}. \tag{40}$$

Por causa da dificuldade de computar $Pr(E \mid l = c)$ o classificador *Naive Bayes* assume de forma simplista que o evento de participação individual de cada empresa $E = e_i$ é independente das outras, logo, temos que

$$Pr(E \mid l = c) = \prod_{i=1}^{m} Pr(E = e_i \mid l = c).$$

Já para computar as probabilidades a priori básicas contidas nos dados, temos que

$$Pr(l = c) = \frac{|L^c|}{|L|} \text{ e}$$
$$Pr(E = e_i \mid l = c) = \frac{k + |L^c_{e_i}|}{2k + |L^c|},$$

sendo $k = 0,001$ uma constante necessária para evitar erros de classificação quando a ocorrência do evento $E = e_i$ for rara na base de dados.

Como citamos acima, a suposição de independência dos eventos $E = e_i$ é simplista, e supomos que exista a formação de conluios entre empresas para fraudar licitações. Para quantificar estas relações contidas nos dados, propomos a construção do particionamento P^c das empresas $E = \{e_1, ..., e_m\}$ através do algoritmo $PartitionTree$, sendo cada grupo de empresas $S \in P^c$ uma relação de dependência medida pela suas ocorrências concomitantes em outras licitações. Logo, redefinimos a probabilidade condicionada do evento de participação das empresas E em uma licitação de categoria c como

$$Pr(E \mid l = c) = \prod_{S \in P^c} Pr(E = S \mid l = c).$$

E por fim, a probabilidade a priori do evento $E = S$ pode ser computada por

$$Pr(E = S \mid l = c) = \frac{k + |L_S|}{2k + |L^c|},$$

aplicando a mesma constante k com o objetivo de tratamento de eventos raros.

Para o entendimento do efeito da identificação de dependências entre eventos pelo PAI para a classificação de licitações, propomos computar a probabilidade condicionada $Pr(\hat{l} = fraude \mid \hat{E})$ da licitação exemplo $\hat{l}$ cuja lista de empresas concorrentes $\hat{E} = \{e_1, e_2, e_3, e_4\}$ tem seu histórico de participação em licitações ilustradas na Figura 13(a).

Figura 13: Licitação exemplo $\hat{l}$. (a) Histórico das licitações que as empresas de $\hat{E}$ concorreram e (b) probabilidades a priori contidas no histórico de $\hat{E}$.

	licitações
e_1	l_1 l_3 l_6
e_2	l_1 l_2 l_4 l_5
e_3	l_2 l_3 l_4 l_5
e_4	l_1 l_2 l_5 l_6

(a)

E	$Pr(E\|fraude)$
e_1	1/3
e_2	3/3
e_3	2/3
e_4	2/3

E	$Pr(E\|legit)$
e_1	2/3
e_2	1/3
e_3	2/3
e_4	2/3

(b)

$\hat{L}^{legit} = \{l_3, l_5, l_6\}$

$\hat{L}^{fraude} = \{l_1, l_2, l_4\}$

$Pr(l = fraude) = 1/2$

$Pr(l = legit) = 1/2$

A Figura 13(b) apresenta as probabilidades a priori extraídas do histórico de licitações das empresas de $\hat{E}$. Por exemplo, a probabilidade da empresa e_3 participar de uma licitação caso ela seja fraudulenta é $Pr(E = e_2 \mid l = fraude) = 2/3$, pois das 3 empresas rotuladas como fraudulentas, e_3 participou de 2 (l_2 e l_4). A chance de uma licitação l qualquer ser fraudulenta é $Pr(l = fraude) = 1/2$. A categorização destas licitações do histórico está ilustrada nas listas $\hat{L}^{fraude}$ e $\hat{L}^{legit}$. Nesse exemplo ilustrativo consideramos a constante de eventos raros $k = 0$ para facilitar o seu entendimento.

Em seguida computamos $Pr(\hat{E} \mid l = fraude)$ e $Pr(\hat{E} \mid l = legit)$. Inicialmente, para efeito de comparação, assumiremos que os eventos são independente, logo,

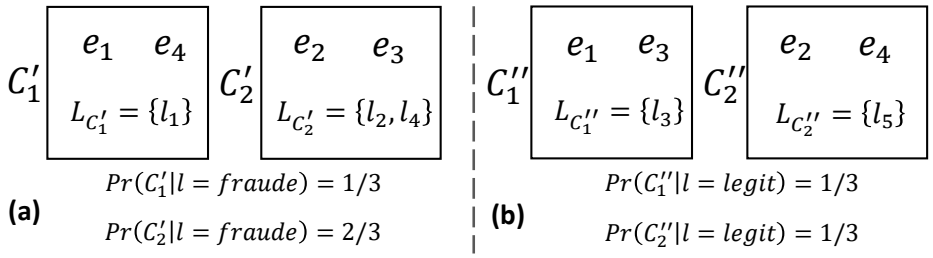
$$Pr(\hat{E} \mid l = fraude) = \frac{1}{3} \cdot 1 \cdot \frac{2}{3} \cdot \frac{2}{3} = \frac{4}{27},$$

de forma análoga computamos $Pr(\hat{E} \mid l = legit) = 8/81$, e aplicando a equação (40) temos que $Pr(\hat{l} = fraude \mid \hat{E}) = 3/5$.

Agora iremos considerar a dependência dos eventos entre as empresas de $\hat{E}$ e utilizando o PAI construímos os particionamentos P^{fraude} e P^{legit} que estão ilustrados na Figura 14. Temos também as probabilidades a priori dos grupos de empresas das partições, como por exemplo, da partição $P^{fraude} = \{C'_1, C'_2\}$, ilustrada na Figura 14(a), podemos observar que a probabilidade $Pr(E = C'_2 \mid l = fraude) = 2/3$ pois sua interseção

$L_{C_2'}$ possui duas (l_2 e l_4) das três licitações rotuladas como fraude.

Figura 14: Particionamentos de $\hat{E}$ para as listas de empresas $\hat{L}^{fraude}$ e $\hat{L}^{legit}$. (a) Particionamento P^{fraude} e (b) particionamento P^{legit}.



De posse das probabilidades a priori das partições, podemos recalcular

$$Pr(\hat{E} \mid l = fraude) = \frac{1}{3} \cdot \frac{2}{3} = \frac{2}{9},$$

$Pr(\hat{E} \mid l = legit) = 1/9$, e aplicando a equação (40) temos que $Pr(\hat{l} = fraude \mid \hat{E}) = 2/3$ que é superior à probabilidade de $3/5$ computada anteriormente assumindo independência dos eventos.

5.2.2 Métricas para Cobertura

É proposto um conjunto de 3 métricas base para avaliação dos grupos contidos em uma cobertura P. Dessas métricas, duas são baseadas em operações envolvendo probabilidades e uma na medição da similaridade dos membros dos grupos.

Inicialmente, para melhor entender o uso da teoria das probabilidades nas métricas dos grupos propomos o seguinte cenário fictício: considere um supermercado cuja base de dados é composta pelo registro de $n = 100$ compras. Foram registrados $n_f = 12$ compras de fralda, $n_c = 8$ compras de cerveja e $n_{f \cap c} = 4$ compras de fralda e cerveja simultâneas. Logo, podemos afirmar que a probabilidade a priori de um cliente qualquer comprar fralda é $p(f) = 12\%$ e comprar cerveja é $p(c) = 8\%$. Se dois eventos são considerados independentes então $p_{ind}(A \cap B) = p(A) \cdot p(B)$. Sendo assim, a chance do mesmo cliente comprar simultaneamente fralda e cerveja ao acaso é $p_{ind}(f \cap c) = 0,08 \cdot 0,12 = 0,96\%$. Entretanto, podemos computar a probabilidade a priori de um cliente qualquer desta loja comprar os dois produtos simultaneamente usando a fórmula $p_{priori}(f \cap c) = \frac{n_{f \cap c}}{n} = 4\%$.

Considerando que um grupo pode conter duas ou mais empresas, é extrapolado o conceito para um grupo C com a equação

$$p_{ind}(C) = \prod_{i \in C} \frac{|L_i|}{|L|},$$

que mede a probabilidade independente das empresas $i \in C$ serem observadas juntas. Já a probabilidade a priori de um grupo C é definida pela equação

$$p_{priori}(C) = \frac{|L_C|}{|L|},$$

que mede a razão entre o tamanho da interseção L_C pelo número de licitações existentes no conjunto de dados.

Retornando ao exemplo do supermercado, muitas vezes queremos conhecer quanto a compra de dois produtos possuem dependência. Para isto utilizamos, por exemplo, a função $lift(f, c) = \frac{p_{priori}(f \cap c)}{p_{ind}(f \cap c)} = 4,17$ que indica quantas vezes a compra de fralda e cerveja simultânea neste supermercado é maior que o acaso. Isso permite que sejam tomadas decisões sobre promoções de venda.

No contexto de análise dos grupos é desejável ter uma métrica para medir a surpresa de sua formação, isto é, quão provável é a ocorrência destas empresas em grupo em comparação ao acaso. Comumente, em análise de associações de itens, são utilizadas duas métricas com esse fim: *lift* e *alavancagem* [45]. Propomos para a métrica de grupos as equações $lift(C) = \frac{p_{priori}(C)}{p_{ind}(C)}$ e $alavancagem(C) = p_{priori}(C) - p_{ind}(C)$. O *lift* serve para medir quantas vezes os membros de um grupo aparecerem juntos comparados com o acaso. Enquanto que, a *alavancagem* mensura quão mais (ou menos) provável é do dado grupo existir em comparação à probabilidade ao acaso.

Considerando P_e o conjunto de grupos da cobertura P que possuem a empresa e, então são propostas as métricas *lift* e *alavancagem* de cada empresa $e \in X$ como $lift(e) = \frac{\sum_{C \in P_e} lift(C)}{|P_e|}$ e $alavancagem(e) = \frac{\sum_{C \in P_e} alavancagem(C)}{|P_e|}$, sendo a média da respectiva métricas de todos os grupos contidos em P_e.

Entretanto, as métricas a serem utilizadas na fase de aprendizado supervisionado se referem às licitações e não às empresas ou grupos. Logo, considerando X_l como o conjunto de empresas que participam da licitação l, temos as métricas

$$lift(l) = \frac{1}{|X_l|} \cdot \sum_{e \in X_l} lift(e) \text{ e} \tag{41}$$

$$alavancagem(l) = \frac{1}{|X_l|} \cdot \sum_{e \in X_l} alavancagem(e), \tag{42}$$

associadas às licitações $l \in L$.

Por fim, considere m_C como o número de pares distintos de empresas participantes

de um grupo C. É proposta uma métrica para medição da similaridade de C pela equação

$$similaridade(C) = \frac{1}{m_C} \cdot \sum_{(i,j) \in C} \frac{|L_i \cap L_j|}{|L_i \cup L_j|},$$

que consiste na média aritmética do coeficiente de Jaccard [37] de cada par distinto de empresas em C.

De forma análoga às métricas de $lift$ e *alavancagem* propomos para a métrica de *similaridade* uma versão para cada empresa e. Esse medida é a média aritmética das similaridades de cada grupo de P_e. Para cada licitação $l \in L$ temos a seguinte equação

$$similaridade(l) = \frac{1}{|X_l|} \cdot \sum_{e \in X_l} similaridade(e). \tag{43}$$

5.3 Algoritmo de Aprendizagem de Máquina

Em todos os conjuntos de dados utilizados, cada licitação possui um rótulo binário indicando a presença ou não de conluio nesta, se tratando portanto, de um problema de classificação. Adotamos as métricas computadas na etapa anterior como novos parâmetros das amostras a serem fornecidas como entrada para as técnicas de aprendizado supervisionado. Para a resolução do problema de classificação das licitações, é proposta a utilização de alguns dos algoritmos de Aprendizagem de Máquina mais utilizados, no caso:

- Máquina de Vetor de Suporte (SVM);
- K-Vizinhos mais Próximos (K-NN);
- Rede Neural (NN);
- Random Forest (RF);
- Árvore de Decisão (DT).

Em alguns algoritmos, se pressupõe uma distribuição normal dos dados ou que as características estejam na mesma escala. Caso contrário, algumas das características apresentadas ao algoritmo podem fazer com que este seja incapaz de aprender com os dados apresentados. Sendo assim, com fins de mitigação desse problema, cada variável x dos dados passa por uma padronização seguindo a equação

$$z = \frac{x - \mu}{\sigma}$$

onde μ é sua média aritmética e σ seu desvio padrão.

Para a avaliação dos modelos de aprendizagem de máquina, são definidas partições de treino e teste, sendo feita uma divisão dos objetos do conjunto de dados entre estas partições. Considerando que uma licitação pode ter mais de um participante, a divisão entre partições é realizada por licitação.

Para evitar dar informações sobre o teste para a função hipótese treinada (bisbilhotagem de dados) propomos as seguinte ações:

- A operação de normalização dos dados é feita sobre a partição de treinamento e utilizando os valores μ e σ computados no treino efetuamos a normalização da partição teste;
- A partição de treino não considera o conjunto de licitações da partição teste para a construção da solução do PAI e posterior computação das métricas sobre as licitações. Por outro lado, na computação das métricas das licitações do teste é considerado todo o conjunto de licitações original.

6 APRESENTAÇÃO E ANÁLISE DOS RESULTADOS

Os algoritmos propostos na Seção 4.2 foram implementados em linguagem C++, utilizando o compilador g++ versão 5.4.0 (opção de compilação -O2). Em relação aos algoritmos de Aprendizagem de Máquina da Seção 5.3, foi utilizada a linguagem de programação Python versão 3.9.6 e a biblioteca *scikit-learn* [11]. Todos os modelos de programação inteira foram solucionados com auxílio do resolvedor matemático IBM CPLEX versão 12.9. Todos os experimentos computacionais deste trabalho foram executados em um computador com processador Intel Xeon E5-2650 v4 2.20 GHz, com 128GB de memória RAM, e sistema operacional Linux Ubuntu 16.04.

Estudos inicias em instâncias artificiais aleatoriamente geradas foram conduzidos. Para estas, $XL(n, m, d)$ denota uma instância com n objetos em X, m recursos em L, tal que cada relacionamento entre um objeto e um recurso pode existir de maneira independente com probabilidade p. Primeiro, os subconjuntos X e L são inicializados com, respectivamente, n e m elementos. Em seguida, para cada objeto $i \in X$, é feita uma decisão com probabilidade p sobre adicionar ou não o recurso $r \in L$ em L_i. Observe que a densidade de XL é definida como $d = \frac{\sum_{i \in X} |L_i|}{n \cdot m}$.

Foram utilizados quatro conjuntos de dados envolvendo licitações no Brasil, Itália, Japão e Estados Unidos, disponibilizados pelos autores do artigo *"Collusion detection in public procurement auctions with machine learning algorithms"* [47]. A Tabela 1 descreve as principais características dos conjuntos de dados. Na tabela, a densidade da instância do PAI é dada por Densidade $= \frac{\#\text{ propostas}}{\#\text{ licitações} \times \#\text{ empresas}}$. Todas as bases contém as características: quantidade de participantes e valor da proposta vencedora. As licitações do Brasil, Japão e Itália incluem a estimativa de valor pré-licitação (PTE), e a razão entre o valor da proposta vencedora e o PTE.

Tabela 1: Descrição dos conjuntos de dados utilizados

País de origem dos dados	**Brasil**	**Japão**	**Itália**	**Estados Unidos**
Escopo	Licitações de infraestrutura da Petrobras	Construção de edificações e engenharia civil	Construção de estradas	Fornecimento de leite para escolas
# licitações	101	1080	278	3754
# empresas	272	1665	821	120
# propostas	683	13515	20286	7004
# licitações fraudulentas	33	123	143	487
Densidade (PAI)	2,48%	0,75%	8,88%	1,55%

6.1 PAI contra modelos da literatura

Inicialmente, para avaliar as estratégias para resolução do PAI em comparação à modelos de agrupamento da literatura, serão utilizadas as instâncias XL. Para os métodos propostos para o PAI, as instâncias XL não necessitam de um pré-processamento. Por

outro lado, para utilização dessas instâncias por parte dos modelos CEP, p-CEP, e p-medianas$_{jac}$, é necessário prover uma matriz de distância. No caso, a matriz é computada por meio da função de distância de Jaccard $d(i,j) = 1 - \frac{|L_i \cap L_j|}{|L_i \cup L_j|}$. Para o modelo do BEP, no qual se espera como entrada um grafo bipartido $G = (V_1, V_2, E)$, é realizado um mapeamento $V_1 = X$, $V_2 = L$, e $E = (i,j) \mid i \in X$ e $j \in L_i$. Para o modelo p-medianas$_{dis}$, é provida uma matriz de similaridade entre os objetos de X computada pela função de similaridade $\delta(i,j)$, que mede o número de características dissimilares entre i e j ($|L_i \setminus L_j|$).

A Tabela 2 apresenta as métricas para os modelos da literatura e os algoritmos para o PAI nas instâncias XL onde $10 \leq |X| \leq 20$. O grupo de colunas Inst. descrevem as dimensões de cada instâncias com as colunas $|X|$, $|L|$, e d, que representam, respectivamente, o número de objetos, recursos e a densidade da instância. Cada algoritmo testado possui seu próprio grupo de colunas. Em cada grupo, a coluna I_{asi} se refere ao valor médio da Silhueta por Interseção do particionamento obtido, definida na Seção 4.1.1). A coluna Obj representa o valor da função objetivo (28). Por sua vez, a coluna t(s) é o tempo gasto, em segundos, para resolução da instância. No conjunto de colunas PAI-B&B, CEP e BEP, K indica o número de grupos na solução. Em todas as instâncias, o valor de K foi idêntico entre os modelos PAI-B&B e PAI-MPI. Sendo assim, a coluna K foi omitida para este último modelo. Considerando que os modelos p-CEP, p-medianas$_{jac}$, e p-medianas$_{dis}$ são modelos de agrupamento não-automático, ou seja, o número de grupos necessita ser fornecido como parâmetro de entrada, o valor de K obtido pelo PAI-B&B foi utilizado para esses modelos. Para todos os modelos, foi definido um tempo limite de 1 hora para a execução de cada instância. As instâncias onde o tempo limite foi excedido estão marcadas com TL na coluna t(s). Valores em negrito indicam os melhores valores das colunas I_{asi}, Obj, e t(s) em uma instância entre todos os modelos.

Tabela 2: Métricas de diferentes modelos de agrupamento em instâncias artificiais selecionadas

Inst.			PAI-MPI			PAI-B&B				CEP				BEP				p-CEP			p-medianas$_{dis}$			p-medianas$_{jac}$		
$\|X\|$	$\|L\|$	d	I_{asi}	Obj	t(s)	I_{asi}	Obj	K	t(s)	I_{asi}	Obj	K	t(s)	I_{asi}	Obj	K	t(s)	I_{asi}	Obj	t(s)	I_{asi}	Obj	t(s)	I_{asi}	Obj	t(s)
10	16	0.3	0.11	**12**	1.11	0.11	**12**	5	**0.0**	0.0	0	2	**0.0**	0.0	0	5	0.06	**0.18**	**12**	0.14	0.0	2	0.01	0.0	1	0.01
10	16	0.5	0.12	**27**	2.2	0.13	**27**	5	**0.0**	0.0	0	1	**0.0**	0.0	0	2	0.47	**0.18**	**27**	0.09	0.0	5	0.01	0.0	6	0.01
10	16	0.6	0.15	**34**	2.06	0.15	**34**	5	**0.0**	0.0	0	1	**0.0**	0.0	6	4	1.92	**0.16**	**34**	0.09	0.0	16	0.01	0.0	10	0.01
10	16	0.7	**0.16**	**45**	1.75	**0.16**	**45**	5	**0.0**	0.0	1	1	**0.0**	0.0	1	2	4.32	**0.16**	**45**	0.08	0.0	13	0.01	0.0	13	0.01
10	16	0.8	**0.12**	**54**	0.86	**0.12**	**54**	5	**0.0**	0.0	1	1	**0.0**	0.09	11	2	0.05	0.03	43	0.1	0.0	7	0.01	0.0	7	0.01
10	21	0.27	0.06	**12**	1.28	**0.07**	**12**	5	**0.0**	0.0	0	3	0.01	0.0	0	6	0.04	0.06	**12**	0.15	0.0	11	0.01	0.0	6	0.01
10	21	0.33	**0.31**	**15**	0.82	**0.31**	**15**	3	**0.0**	0.0	13	6	**0.0**	0.0	1	3	0.01	0.0	12	0.07	0.2	13	0.01	0.2	13	0.01
16	30	0.3	**0.15**	**34**	2661.43	0.14	**34**	8	**0.0**	0.0	0	1	0.01	0.0	2	9	205.6	0.12	**34**	TL	0.0	12	0.02	0.0	10	0.01
16	30	0.4	0.17	53	TL	**0.19**	**55**	8	**0.0**	0.0	0	1	0.02	0.0	6	7	TL	**0.19**	**55**	TL	0.0	26	0.01	0.0	17	0.02
16	30	0.5	0.15	**76**	TL	0.15	**76**	8	0.01	0.0	0	1	**0.0**	0.0	7	6	TL	**0.17**	**76**	TL	0.0	43	0.01	0.0	28	0.36
16	30	0.6	**0.15**	**104**	TL	**0.15**	**104**	8	0.02	0.0	0	1	**0.0**	0.05	4	2	TL	**0.15**	**104**	TL	0.0	55	0.02	0.0	40	0.22
16	30	0.7	**0.12**	**131**	1263.08	**0.12**	**131**	8	0.07	0.0	0	1	**0.0**	0.1	9	2	TL	**0.12**	**131**	TL	0.0	56	0.02	0.0	38	0.02
16	30	0.8	0.11	**164**	421.05	0.11	**164**	8	0.02	0.0	1	1	**0.0**	0.0	1	2	TL	**0.12**	**164**	TL	0.0	51	0.01	0.0	27	0.01
16	30	0.9	**0.06**	**200**	12.45	**0.06**	**200**	8	0.08	0.0	5	1	**0.0**	0.0	5	1	1709.83	0.01	174	TL	0.0	14	0.01	0.0	14	0.01
19	23	0.36	**0.04**	**62**	344.18	**0.04**	**62**	10	0.23	0.0	1	3	0.03	0.0	0	7	2.14	**0.04**	**62**	TL	0.0	30	**0.01**	0.0	18	**0.01**
19	23	0.42	**0.01**	52	TL	**0.01**	**53**	10	0.66	0.0	0	1	0.02	0.0	5	9	TL	**0.01**	**53**	TL	0.0	16	**0.01**	0.0	14	**0.01**
19	33	0.18	0.13	**32**	13.58	**0.14**	**32**	8	**0.0**	0.0	2	7	0.03	0.0	0	5	0.35	0.0	19	TL	0.0	18	0.01	0.0	2	0.01
19	33	0.37	**0.2**	**68**	TL	0.19	**68**	8	5.02	0.0	0	5	0.02	0.0	4	4	TL	0.0	45	TL	0.0	30	**0.01**	0.0	8	**0.01**
19	35	0.37	**0.03**	**96**	50.18	**0.03**	**96**	9	13.93	0.0	3	4	0.02	0.0	0	2	892.55	0.0	74	TL	0.0	51	**0.01**	0.0	43	**0.01**
19	35	0.42	**0.01**	74	TL	**0.01**	**78**	10	0.81	0.0	0	1	**0.0**	0.0	19	6	TL	**0.01**	**78**	TL	0.0	27	0.01	0.0	23	0.01
19	35	0.95	**0.01**	**263**	135.76	**0.01**	**263**	10	TL	0.0	0	1	**0.0**	0.0	0	1	TL	0.0	207	TL	0.0	49	0.01	0.0	49	0.01
19	39	0.53	**0.17**	**153**	10.93	**0.17**	**153**	8	0.82	0.0	12	5	0.02	0.0	14	3	36.67	0.0	93	TL	0.0	122	**0.01**	0.0	41	**0.01**
19	39	0.68	**0.11**	**197**	TL	**0.11**	**197**	9	323.9	0.0	0	3	0.02	0.0	0	4	TL	0.0	157	TL	0.0	79	**0.01**	0.0	2	**0.01**
20	23	0.2	0.12	**24**	2679.88	0.13	**24**	10	**0.0**	0.0	2	4	0.13	0.0	1	9	0.45	**0.16**	**24**	TL	0.0	7	0.01	0.0	7	0.01
20	23	0.3	0.14	36	TL	0.15	**37**	10	**0.01**	0.0	0	1	0.03	0.0	7	10	906.97	**0.17**	**37**	TL	0.0	9	0.02	0.0	11	**0.01**
20	23	0.5	0.11	74	TL	**0.12**	**75**	10	0.28	0.0	0	1	**0.0**	0.0	5	6	TL	0.11	**75**	TL	0.0	28	0.01	0.0	21	0.01
20	23	0.6	**0.11**	98	TL	**0.11**	**101**	10	0.26	0.0	0	1	**0.0**	0.0	6	4	TL	**0.11**	**101**	TL	0.0	54	0.01	0.0	37	0.01
20	23	0.7	**0.12**	**133**	TL	0.11	**133**	10	2.89	0.0	0	1	**0.0**	0.01	1	2	TL	**0.12**	**133**	TL	0.0	71	0.01	0.0	18	0.01
20	23	0.8	**0.09**	**161**	TL	**0.09**	**161**	10	1.02	0.0	0	1	**0.0**	0.0	0	1	TL	0.01	144	TL	0.0	21	0.01	0.0	20	0.02
20	35	0.2	0.12	26	TL	0.09	**27**	10	0.03	0.0	0	2	0.07	0.0	2	7	621.58	**0.15**	**27**	TL	0.0	5	**0.01**	0.0	9	**0.01**
20	35	0.3	0.11	50	TL	0.08	**52**	10	0.02	0.0	0	1	0.03	0.0	8	7	TL	**0.13**	**52**	TL	0.0	13	**0.01**	0.0	17	**0.01**
20	35	0.5	0.13	108	TL	0.14	**112**	10	0.05	0.0	0	1	**0.0**	0.0	9	6	TL	**0.15**	**112**	TL	0.0	74	0.01	0.0	31	0.01
20	35	0.6	0.14	142	TL	0.15	**146**	10	0.24	0.0	0	1	**0.0**	0.0	0	3	TL	**0.16**	**146**	TL	0.0	72	0.01	0.0	38	0.01
20	35	0.7	**0.1**	189	TL	0.09	**193**	10	57.05	0.0	0	1	**0.0**	0.0	2	3	TL	0.03	182	TL	0.0	91	0.02	0.0	94	0.01
20	35	0.8	**0.11**	**239**	TL	**0.11**	**239**	10	6.72	0.0	1	1	**0.0**	0.0	1	1	TL	0.01	214	TL	0.0	8	0.01	0.0	9	0.01
20	39	0.28	**0.06**	**72**	15.98	**0.06**	**72**	9	0.68	0.0	4	6	**0.02**	0.0	0	7	27.29	0.0	52	TL	0.0	26	**0.02**	0.0	1	0.49
20	39	0.34	**0.11**	**89**	17.48	**0.11**	**89**	9	0.18	0.0	17	6	0.02	0.0	0	5	0.23	0.0	61	1.58	0.0	44	0.02	0.0	16	**0.01**

Considerando os valores obtidos pelos modelos na coluna Obj, é possível ver que os modelos CEP e BEP, que são modelos de agrupamento automático por meio de edição de arestas, alcançaram os piores valores de recursos nas interseções dos grupos, sendo estes valores, em média, 1,84% e 4% dos melhores valores conhecidos, respectivamente. Por sua vez, os modelos p-medianas$_{jac}$ e p-medianas$_{dis}$, que agrupam objetos com base na proximidade de um objeto ao centroide de cada grupo, obtiveram interseções maiores em comparação aos modelos de edição de arestas. No entanto, a marca é somente 32,15% e 37,05% dos melhores valores conhecidos, respectivamente. Vale mencionar que na vasta maioria dessas instâncias, esses quatro modelos obtiveram $I_{asi} = 0.0$ em suas soluções.

O modelo de programação inteira (MPI) para o PAI foi capaz de solucionar 19 das 36 instâncias da tabela dentro do tempo limite. Ainda, o valor da solução ótima foi alcançado em 6 outras instâncias, apesar que para essas a otimalidade não foi garantida dentro do tempo limite. Por outro lado, o algoritmo branch and bound (B&B) resolveu 35 das 36 instâncias dentro do tempo limite. Para a instância cujo tempo foi excedido pelo B&B, $XL(19, 35, 0.95)$, a sua solução intermediária foi provada como ótima pelo MPI.

Por sua vez, o modelo de agrupamento não automático p-CEP foi apenas capaz de resolver 8 instâncias dentro do tempo limite de 1 hora. No entanto, ao considerar suas soluções intermediárias, tem-se que o modelo foi capaz de encontrar 22 soluções ótimas para o PAI. Nas 14 instâncias onde o p-CEP não alcançou o valor ótimo, houve uma diferença relativa média para o melhor valor conhecido de 88.89%.

Analisando o valor de I_{asi}, é possível visualizar que o algoritmo B&B encontrou o melhor valor em apenas 22 instâncias. Isso ocorre apesar dele ter encontrado a solução ótima em todos os casos. Nas 14 instâncias, tem-se que o melhor valor o I_{asi} foi encontrado pelo MPI ou pelo p-CEP. Todavia, nesses casos, modelos diferentes resultaram em um mesmo valor para a função objetivo. Isso mostra que existe múltiplas soluções ótimas com um mesmo valor para a função objetivo do PAI, mas com valores distintos para o I_{asi}, que considera a distância de cada objeto ao grupo mais próximo que este não pertence.

6.2 Resultados das abordagens para resolução do PAI$_{cov}$

A Tabela 3 apresenta métricas da soluções para o PAI$_{cov}$ do MPI e do algoritmo enumerativo IntersecForest em instâncias artificiais onde $10 \leq |X| \leq 20$. Cada algoritmo possui um conjunto específico de colunas, onde a coluna K mostra o número de grupos formado na cobertura, a coluna $\Sigma|L_C|$ representa a soma dos tamanhos dos conjuntos de interseções de recursos formados em cada grupo e t(s) é o tempo gasto, em segundos, para resolver a instância. Em todas as instâncias, foi adotado um tempo limite de 1 hora para execução, sendo os casos onde a execução ultrapassou esse limite marcados como TL na coluna t(s). Para as colunas K, $\Sigma|L_C|$, e t(s), o melhor valor encontrado está assinalado

em negrito. O parâmetro m do MPI define o número máximo de grupos enumerados na solução, sendo para cada instância definido como o valor de K obtido pela solução do algoritmo IntersecForest.

Tabela 3: Métricas dos métodos para resolução do PAI$_{cov}$ em instâncias artificiais selecionadas

Inst.			IntersecForest			MPI			Inst.			IntersecForest			MPI		
$\|X\|$	$\|L\|$	d	K	$\Sigma\|L_C\|$	t(s)	K	$\Sigma\|L_C\|$	t(s)	$\|X\|$	$\|L\|$	d	K	$\Sigma\|L_C\|$	t(s)	K	$\Sigma\|L_C\|$	t(s)
10	16	0.3	**21**	**47**	**0.0**	**21**	**47**	218.44	19	35	0.42	**752**	**2982**	**0.01**	14	86	TL
10	16	0.5	**71**	**260**	**0.0**	35	142	TL	19	35	0.95	**40210**	**592384**	**5.61**	1	0	TL
10	16	0.6	**117**	**499**	**0.0**	40	201	TL	19	39	0.53	**1618**	**28502**	**0.06**	16	353	TL
10	16	0.7	**174**	**985**	**0.0**	55	338	TL	19	39	0.68	**20834**	**188689**	**1.88**	1	0	TL
10	16	0.8	**146**	**1054**	**0.0**	57	479	TL	20	23	0.2	**53**	**111**	**0.0**	34	83	TL
10	21	0.27	**29**	**55**	**0.0**	25	49	TL	20	23	0.3	**138**	**341**	**0.0**	29	75	TL
16	30	0.3	**143**	**377**	**0.0**	31	97	TL	20	23	0.5	**742**	**2996**	**0.01**	27	123	TL
16	30	0.4	**372**	**1252**	**0.0**	18	80	TL	20	23	0.6	**1670**	**8274**	**0.02**	19	158	TL
16	30	0.5	**753**	**3145**	**0.01**	39	197	TL	20	23	0.7	**3328**	**24187**	**0.13**	1	0	TL
16	30	0.6	**1849**	**10024**	**0.02**	11	132	TL	20	23	0.8	**6986**	**53724**	**0.42**	1	0	TL
16	30	0.7	**3466**	**24002**	**0.13**	1	0	TL	20	35	0.2	**103**	**218**	**0.0**	19	45	TL
16	30	0.8	**8923**	**85720**	**0.59**	1	1	TL	20	35	0.3	**287**	**846**	**0.0**	43	166	TL
16	30	0.9	**9643**	**144546**	**0.58**	1	5	TL	20	35	0.5	**1646**	**7639**	**0.02**	15	120	TL
19	23	0.36	**341**	**1599**	**0.0**	36	220	TL	20	35	0.6	**5023**	**27138**	**0.26**	1	0	TL
19	23	0.42	**357**	**1195**	**0.0**	40	165	TL	20	35	0.7	**16884**	**133322**	**2.11**	1	0	TL
19	33	0.18	**52**	**200**	**0.0**	42	170	TL	20	35	0.8	**23384**	**237132**	**3.78**	1	1	TL
19	33	0.37	**581**	**2475**	**0.0**	40	199	TL	20	39	0.28	**996**	**6529**	**0.01**	7	75	TL
19	35	0.37	**1179**	**9644**	**0.02**	18	241	TL	20	39	0.34	**151**	**1612**	**0.0**	25	282	TL

O modelo de programação inteira foi capaz de resolver apenas uma instância dentro do tempo limite estipulado, ao passo que o algoritmo IntersecForest resolveu todas as 36 instâncias com um tempo médio de 0,44 segundos. É possível observar uma relação direta entre a densidade d do grafo e os valores de K e $\Sigma|L_C|$. Por exemplo, considerando as instâncias com $|X| = 16$ e $|L| = 30$ e densidade variável em $d \in \{0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$, é possível visualizar que, conforme o valor de d aumenta, tanto K quanto $\Sigma|L_C|$ aumentam, o que leva a um maior tempo gasto para computação da solução. Ou seja, aumentar a densidade implica em aumentar a dificuldade de resolução da instância.

Adicionalmente, o algoritmo IntersecForest foi aplicado nos conjuntos de dados de licitações públicas, sendo os resultados apresentados na Tabela 4.

Tabela 4: Métricas do algoritmo IntersecForest nos conjuntos de dados de licitações públicas

Conjunto de dados				IntersectForest		
País	$\|X\|$	$\|L\|$	d	K	Obj	t(s)
Brasil	272	101	0.02	332	879	0.1
Japão	1665	1080	0.01	18144	69953	22.8
Itália	821	278	**0.09**	1339632	4.89e+07	**33144.4**
Estados Unidos	120	3754	0.02	388	2987	0.2

Em 3 dos 4 conjuntos de dados testados, o algoritmo IntersecForest resolveu a instância em menos de 30 segundos. No entanto, a exceção foi o conjunto de dados de

licitações da Itália, onde a densidade da instância é $d = 9\%$. Essa densidade maior faz com que o algoritmo enumere um número maior de grupos, resultando em um maior tempo de execução.

Nesse sentido, se percebe uma fraqueza do algoritmo em situações onde há uma grande quantidade de licitações e empresas, e estas se relacionam de forma que a instância seja densa. Nessas situações, o tempo gasto pelo algoritmo pode tornar a sua utilização inviável. Portanto, é importante considerar essa limitação ao aplicar o algoritmo nesse tipo de instância.

6.3 Resultados do Modelo de Aprendizagem de Máquina

Para a avaliação das características propostas neste trabalho em comparação com trabalhos da literatura e as características já existentes nos dados tratados, os algoritmos escolhidos na Seção 5.3 para a geração de um modelo de detecção de conluios foram treinados em 5 configurações de conjuntos de dados, onde cada versão difere das características apresentadas ao classificador. Em todas as configurações são incluídas as cinco características em comum dos conjuntos de dados testados, no caso, a estimativa de valor pré-licitação (PTE), valor médio das propostas, razão entre o valor da proposta vencedora e o PTE, e quantidade de propostas. Sendo assim, são definidas as seguintes configurações:

- *Configuração 1*: Acrescenta ao conjunto de características originais as 7 características criadas por meio de estatística descritiva (CV, SPD, DIFFP, RD, KURT, SKEW, KTEST), utilizadas em [47] e abordadas na Seção 3.2;
- *Configuração 2*: incorpora a característica da probabilidade $P(x)$ de uma licitação, proposta em [52], e descrita pela equação 9 na Seção 3.1;
- *Configuração 3*: inclui ao conjunto original as 3 métricas de uma licitação dos grupos obtidos de uma cobertura, conforme descrito na Seção 5.2.2 pelas equações 42, 41 e 43. Adicionalmente, para cada licitação l, é considerado o lift e a alavancagem de C_l, onde C_l é o maior grupo de P no qual l aparece como membro da interseção deste. Além disso, todas as 5 novas características passam por uma transformação $x = log_{10}(x)$;
- *Configuração 4*: adiciona a probabilidade $Pr(l = fraude \mid E)$ definida na Seção 5.2.1;
- *Configuração 5*: contempla apenas as características originais do conjunto de dados;
- *Configuração 6*: considera todas as características diferentes utilizadas nas configurações anteriores;

Cada um dos conjuntos de dados foi dividido em partições de treino e teste, sendo utilizadas 80% das licitações para treinamento do modelo, e 20% para o teste da performance dos modelos, sendo os objetos das partições idênticos para toda combinação entre algoritmo e configuração do conjunto de dados utilizado. As características dos grupos são adicionadas após a definição das partições, sendo que a instância de treinamento do PAI não inclui as licitações que foram selecionadas para o teste, ao passo de que a instância de teste inclui todas as licitações do conjunto de dados.

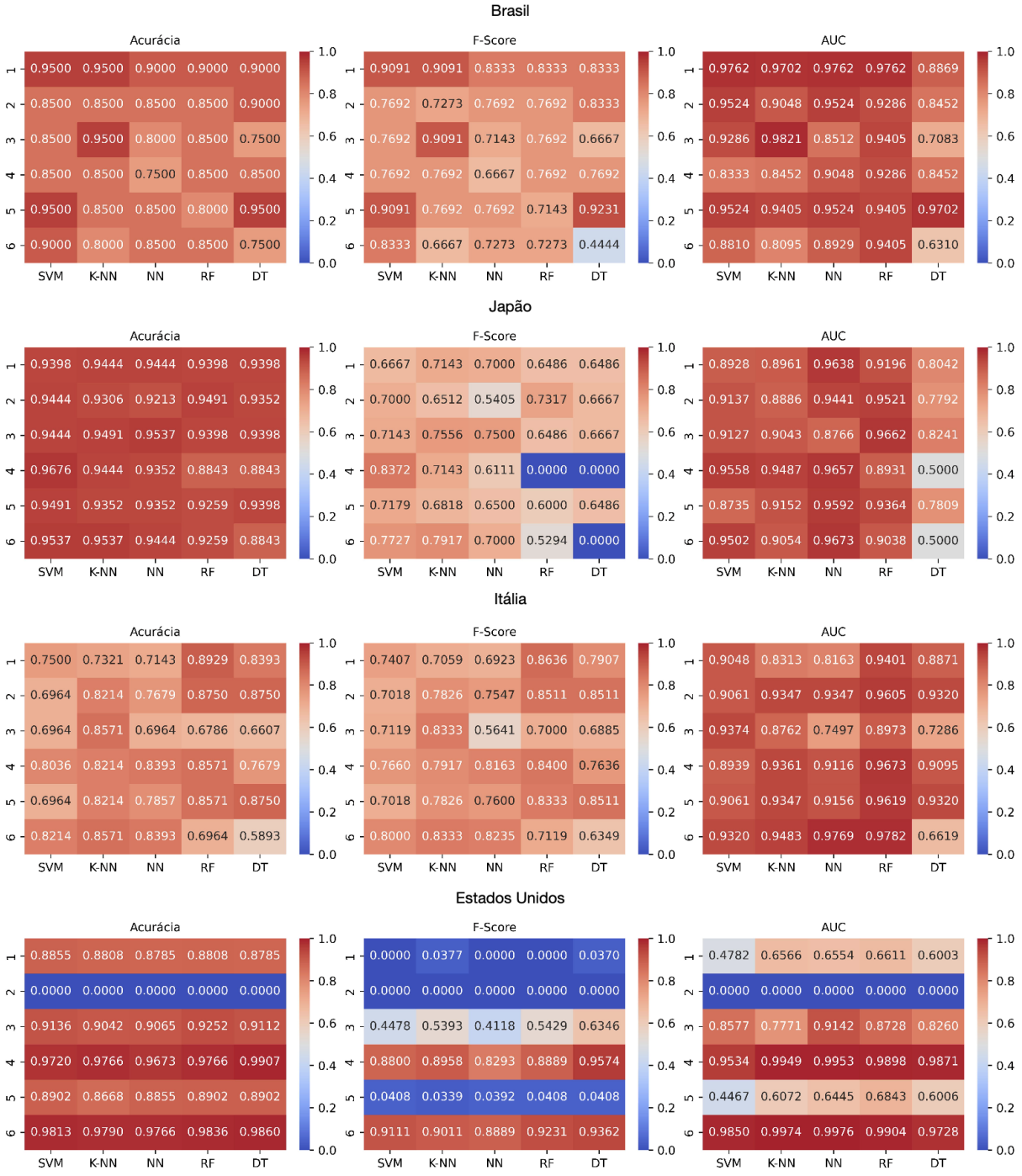
Para todas as combinações entre conjunto de dados e configuração das características testadas, é realizada uma calibração dos hiperparâmetros dos algoritmos utilizados por meio de uma validação cruzada K-fold ($K = 5$), a seguir são apresentados os valores testados na validação:

- SVM: flexibilidade da função fronteira $C \in \{0.001, 0.1, 1, 2\}$ e coeficiente do kernel $\gamma \in \{\frac{1}{|F|}, \frac{1}{|F| \cdot var(X)}\}$, onde $|F|$ é o número de características de entrada, X é a entrada do algoritmo, e $var(X)$ é uma função que retorna a variância de X;
- K-NN: número de vizinhos $k \in \{1, 3, 5, 7\}$, ordem da função de distância de Minkowski $p \in \{1, 2, 3, 4\}$, e peso dado aos objetos $w \in \{\text{uniforme}, \text{proporcional}\}$;
- NN: taxa de aprendizagem inicial $\theta \in \{0.001, 0.01, 0.1\}$, taxa de aprendizagem $\in \{\text{constante}, \text{adaptativa}\}$, número de neurônios nas camadas ocultas $\in \{(100, 1), (24, 1), (8, 4, 2, 1)\}$, parâmetro de regularização $\alpha \in \{0.0001, 0.001, 0.1\}$;
- RF: número de classificadores $\in \{29, 99, 199\}$, profundidade máxima da árvore $\in \{\text{ilimitado}, 3, 4, 5, 7\}$, número mínimo de objetos em um nó folha $\in \{1, 2, 3\}$;
- DT: profundidade máxima da árvore, realizada com auxílio da heurística de poda por complexidade de custos (α) [14];

Quando aplicável, foi utilizada como semente aleatória o número 42. Além disso, os hiperparâmetros não envolvidos no processo de calibração foram definidos conforme o padrão da biblioteca *scikit-learn*. O Apêndice 7 apresenta os valores dos hiperparâmetros selecionados. Os modelos obtidos foram avaliados utilizando as métricas abordadas na Seção 2.3, no caso, Acurácia, F-Score e a Área abaixo da curva ROC (AUC).

A Figura 15 ilustra, por meio de um mapa de calor, os valores obtidos para essas métricas na partição de teste nos quatro conjuntos de dados utilizados para os algoritmos de aprendizagem de máquina utilizados. O eixo X representa um algoritmo de aprendizagem de máquina, ao passo que a o eixo Y representa a configuração do conjunto de dados, todas as métricas assumem valores entre 0 e 1, sendo preferível em todos os casos um valor mais alto. Vale ressaltar que para os dados dos Estados Unidos, a Configuração 2 não é aplicável, logo, os valores das métricas para esse caso estão todos zerados.

Figura 15: Mapa de calor das métricas na partição de teste dos modelos de Aprendizagem de Máquina para diferentes configurações de conjunto de dados



Em relação a acurácia, vale salientar que em dois dos três conjuntos de dados, licitações em que não houveram conluio constituem mais de 65% dos dados. Isso reforça que é necessária certa cautela com os números apresentados por essa métrica. De toda forma, as métricas apontam que os melhores modelos treinados com a adição somente das características propostas neste trabalho (Configurações 3 e 4), para os dados do Brasil, conseguem empatar com a melhor acurácia e F-Score das outras configurações. Ao mesmo tempo, o modelo K-NN da Configuração 3 possui o melhor AUC entre todos para esse conjunto de dados. No entanto, o que se nota é que a Configuração 1 é a mais consistente para todos os algoritmos, superando ou igualando a acurácia, F-Score e AUC dos mesmos algoritmos treinados em outras configurações em 4 dos 5 algoritmos, sendo a única exceção o modelo da árvore de decisão.

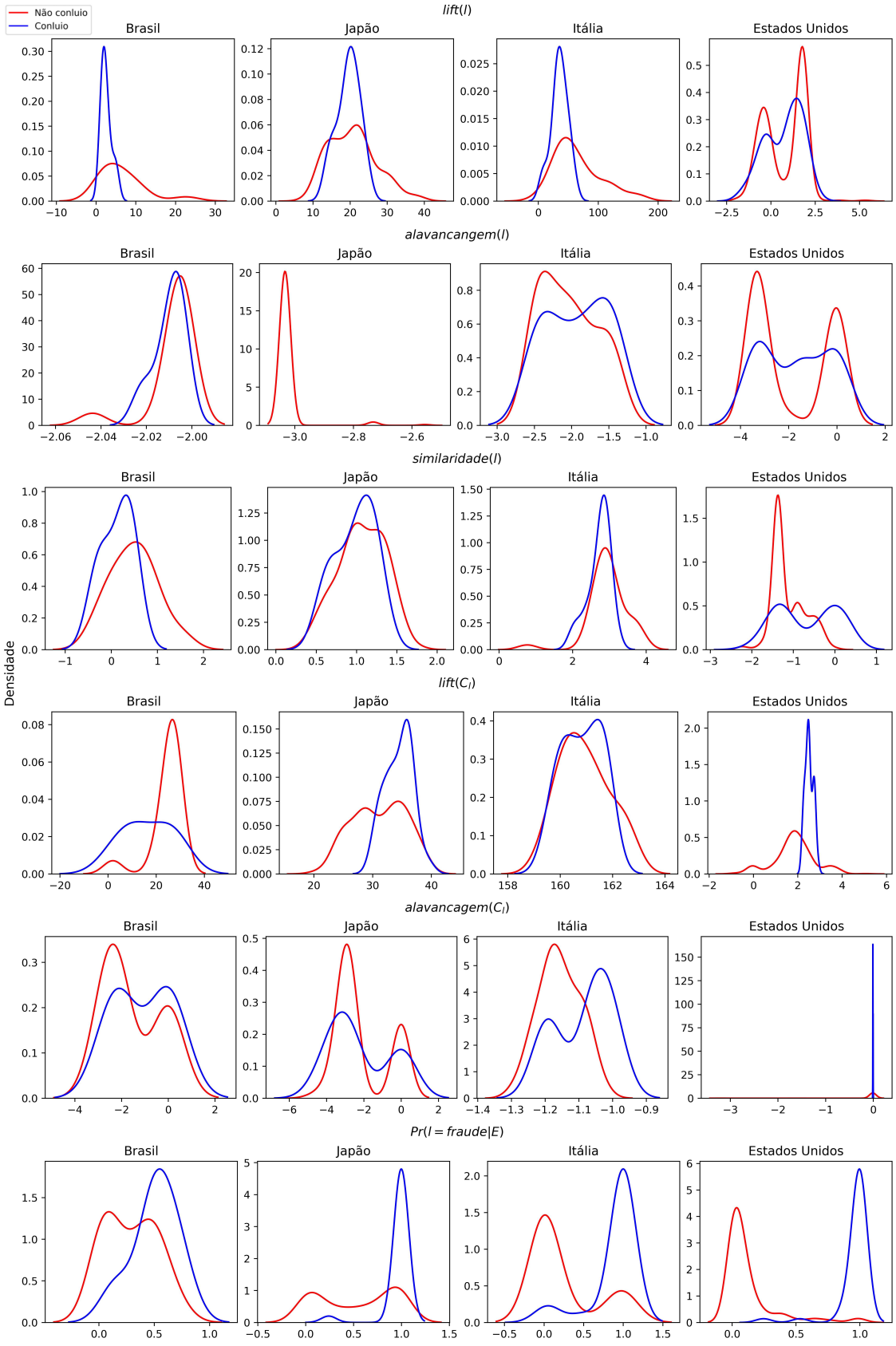
Por sua vez, nos dados do Japão, o melhor valor para a acurácia e o F-Score é obtida pela Configuração 4 utilizando o SVM. Em outro ponto, o melhor AUC é obtido pelo o modelo que incorpora todas as características. Nesse mesmo conjunto de dados, percebe-se que algoritmos que fazem uso de árvores de decisão (RF e DT) obtiveram a pior métrica com a Configuração 4, o que indica um possível sobreajuste do modelo aos dados.

Ao passo de que, no conjunto de dados da Itália, as melhores métricas são obtidas quando se incorpora todas as características utilizadas (Configuração 6) em 3 dos 5 algoritmos para a acurácia e F-Score e 4 dos 5 para o AUC. A Configuração 4 se comparada as características propostas na literatura (Configurações 1 e 2) é melhor ou igual na acurácia e F-Score em 3 dos 5 algoritmos e no AUC em 2 dos 3.

Por fim, as características propostas neste trabalho conseguem melhor refinar o modelo nos dados dos Estados Unidos, superando em todos os casos a Configuração 1. Além disso, o modelo que incorpora todas as características é o melhor de todos em 4 dos 5 algoritmos. Em suma, se comparado ao modelo treinado somente com as características base, a adição das características propostas conseguem melhorar as métricas do modelo na maior parte dos casos testados.

A Figura 16 ilustra a distribuição dos valores das cinco características acrescentadas pela Configuração 3 transformadas por $x = log_{10}(x)$ e da característica acrescentada pela Configuração 4 em relação a presença de conluio em uma licitação na partição de teste dos quatro conjuntos de dados. No gráfico, a linha vermelha se refere à distribuição dos valores para licitações sem conluio, ao passo de que a linha azul trata de licitações com conluio, o eixo vertical trata da densidade de licitações, enquanto que o eixo horizontal se refere ao valor da característica.

Figura 16: Gráfico da estimativa de densidade por kernel para as características dos grupos no conjunto de teste



Pelo gráfico, é possível observar que para o lift da licitação ($lift(l)$) e do grupo da licitação ($lift(C_l)$), a distribuição dos valores desta métrica para os dois tipos de licitações possui uma diferença em 3 dos 4 conjuntos de dados. No caso do $lift(l)$, licitações não fraudulentas possuem uma média superior e também há uma cauda longa. Para o $lift(C_l)$, é observável uma média diferente para diferentes tipos de licitação. Por outro lado, a alavancagem da licitação ($alavancagem(l)$) e do grupo da licitação ($alavancagem(C_l)$ além da similaridade da licitação ($similaridade(l)$) assumem distribuições não muito distintas entre os tipos de licitações em 3 dos 4 conjuntos de dados, com exceção da Itália. Para esse último conjunto de dados, as licitações não fraudulentas possuem valores menores para os dois tipos de alavancagem. De qualquer forma, para essas características, uma possível associação de valores com conluios fica a cargo do processo de treinamento do algoritmo de aprendizagem de máquina.

Por sua vez, para as probabilidades condicionadas, que são adicionadas pela Configuração 4, é possível observar que para todos os conjuntos há uma nítida diferença entre os valores das probabilidades em licitações fraudulentas e não fraudulentas. No caso, as licitações fraudulentas assumem valores superiores em comparação com as licitações não fraudulentas. No entanto, para as licitações do Brasil, há muita sobreposição dos valores, sendo um gerador de ruídos para o classificador. Vale salientar que esta base é a menor de todas, possuindo no total 101 licitações com uma predominância de licitações não fraudulentas. Além disso, existe um grupo de empresas que participaram de um mesmo conjunto de licitações fraudulentas e não fraudulentas, sendo denominado de "Clube dos 16,"conforme descrito em [51].

7 CONSIDERAÇÕES FINAIS

Este trabalho se dedicou a propor um método para detecção de fraudes em licitações públicas, uma prática ilícita que causa prejuízos aos cofres públicos e é objeto de intenso escrutínio das autoridades. No decorrer deste trabalho, procuramos agrupar empresas que em seu histórico participaram em conjunto em várias licitações para detectar possíveis conluios em licitações. Por conta da dificuldade dos modelos de agrupamento clássicos criarem grupos com uma boa interseção, foi introduzido um novo modelo de agrupamento, denominado de Problema de Agrupamento por Interseção (PAI). Em outro ponto, foram elaboradas medidas para descrever os grupos gerados. Em seguida, as métricas propostas são utilizadas em conjunto aos dados originais das licitações como entrada de um algoritmo de aprendizagem de máquina com fins de classificar licitações pela ocorrência de fraudes.

Os experimentos demonstraram que o modelo do PAI, se comparado aos modelos de agrupamento clássicos da literatura, é capaz de particionar os objetos de acordo com a interseção de recursos destes de forma melhor, alcançando soluções que são até 90% melhores sob este ponto de vista. Ao mesmo tempo, se considerado os métodos propostos para resolução do PAI e PAI_{cov}, tem-se que o algoritmo branch and bound, para o primeiro, e o IntersecForest, para o segundo, apresentaram um desempenho superior em questão de tempo em comparação à formulação do problema como um MPI.

Com os grupos em mãos, são propostas diversas métricas para descrevê-los, a depender do tipo de agrupamento gerado. Nesse sentido, ao observar o gráfico da estimativa de densidade por kernel para as métricas propostas, tem-se que a quantificação de probabilidades com base no particionamento de empresas em uma licitação utilizando o PAI resulta em uma discrepância visível nas distribuições entre diferentes tipos de licitações para os conjuntos de dados testados. Todavia, essa observação não se repete em todo caso para as métricas derivadas dos grupos do PAI_{cov}.

Por outro lado, as métricas comumente utilizadas na literatura para quantificar uma possível conduta ilícita entre os participantes são um tanto triviais para serem computadas por uma máquina, enquanto que resolver o PAI implica em resolver um problema NP-Completo. Para poder criar os grupos do PAI é imperativo que os dados possuam alguma forma de identificar as empresas participantes. Além disso, é preciso ressaltar que as licitações envolvidas precisam ser de certa forma relacionadas, como por exemplo, envolvendo a mesma entidade como contratante. Caso contrário, pode-se ter distorções nos valores encontrados, uma vez que é de se imaginar que a maior parte das empresas possui uma área de atuação mais limitada. Essa situação é mitigada neste trabalho, uma vez que em cada conjunto de dados utilizado, as licitações são da mesma área e/ou contratante.

Em relação a detecção de fraudes, foi possível constatar que as características obtidas por meio das métricas dos grupos do PAI foram capazes de melhorar o modelo obtido em comparação ao modelo treinado somente com as características originais. Além disso, os modelos treinados com as características propostas neste trabalho apresentaram melhores resultados se comparados aos modelos treinados com características adicionais propostas na literatura, apresentando um melhor AUC em 3 dos 4 conjuntos de dados testados.

Trabalhos futuros incluem a proposição de métodos heurísticos para resolução do PAI, uma vez que estes, apesar de não possuírem a garantia da otimalidade, são capazes de obterem soluções de boa qualidade com um menor esforço computacional se comparado aos métodos exatos, o que viabiliza a utilização do problema em situações onde a instância é muito grande. Em outra vertente, criar novas métricas para descrever os grupos do PAI_{cov} é um caminho interessante, pois as métricas propostas se mostraram menos eficientes em comparação a probabilidade obtida por meio do particionamento. Por fim, apesar deste trabalho ter se dedicado à aplicação do PAI em licitações, é possível enxergar que esse modelo de agrupamento pode ter uma gama muito maior de aplicações, o que pode ser objeto de um estudo futuro.

Referências

[1] Rakesh Agrawal e Ramakrishnan Srikant. "Fast Algorithms for Mining Association Rules in Large Databases". Em: *Proceedings of the 20th International Conference on Very Large Data Bases*. VLDB '94. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1994, 487–499. ISBN: 1558601538.

[2] Osman Alp, Erhan Erkut e Zvi Drezner. "An Efficient Genetic Algorithm for the p-Median Problem". Em: *Annals of Operations Research* 122 (2003), pp. 21–42. ISSN: 1572-9338. DOI: https://doi.org/10.1023/A:1026130003508.

[3] Gaurab Aryal e Maria F. Gabrielli. "Testing for collusion in asymmetric first-price auctions". Em: *International Journal of Industrial Organization* 31.1 (2013), pp. 26–35. ISSN: 0167-7187. DOI: https://doi.org/10.1016/j.ijindorg.2012.10.002.

[4] Liang Bai e Jiye Liang. "Cluster validity functions for categorical data: a solution-space perspective". Em: *Data Mining and Knowledge Discovery* 29 (nov. de 2015). DOI: 10.1007/s10618-014-0387-5.

[5] Liang Bai e Jiye Liang. "Cluster validity functions for categorical data: a solution-space perspective". Em: *Data Mining and Knowledge Discovery* 29 (6 2015), pp. 1560–1597.

[6] Patrick Bajari e Lixin Ye. "Deciding between Competition and Collusion". Em: *The Review of Economics and Statistics* 85.4 (2003), pp. 971–989. ISSN: 00346535, 15309142. URL: http://www.jstor.org/stable/3211820 (acesso em 08/03/2023).

[7] Daniel Barbará, Yi Li e Julia Couto. "COOLCAT: An Entropy-Based Algorithm for Categorical Clustering". Em: *Proceedings of the Eleventh International Conference on Information and Knowledge Management*. Association for Computing Machinery, 2002, 582–589. ISBN: 1581134924.

[8] Lucas Bastos et al. "Efficient algorithms for cluster editing". Em: *Journal of Combinatorial Optimization* 31 (2016), pp. 347–371. ISSN: 1573-2886. DOI: 10.1007/s10878-014-9756-7. URL: https://doi.org/10.1007/s10878-014-9756-7.

[9] María Beatríz Bernábe Loranca, Rogelio González Velázquez e Martín Analco. "The P-Median Problem: A Tabu Search Approximation Proposal Applied to Districts". Em: *Journal of Mathematics and System Science* 5 (mar. de 2015). DOI: 10.17265/2159-5291/2015.03.002.

[10] Sebastian Böcker e Jan Baumbach. "Cluster Editing". Em: *The Nature of Computation. Logic, Algorithms, Applications*. Ed. por Paola Bonizzoni, Vasco Brattka e Benedikt Löwe. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 33–44. ISBN: 978-3-642-39053-1.

[11] Jérémie du Boisberranger et al. *scikit-learn: machine learning in Python*. URL: https://scikit-learn.org/stable/ (acesso em 22/12/2022).

[12] Bernhard E Boser, Isabelle M Guyon e Vladimir N Vapnik. "A training algorithm for optimal margin classifiers". Em: *Proceedings of the fifth annual workshop on Computational learning theory*. 1992, pp. 144–152.

[13] Erica Bosio e Simeon Djankov. *How large is public procurement?* 2020. URL: https://blogs.worldbank.org/developmenttalk/how-large-public-procurement (acesso em 30/12/2022).

[14] L. Breiman et al. *Classification and Regression Trees*. Taylor & Francis, 1984. ISBN: 9780412048418.

[15] Leo Breiman. "Machine Learning, Volume 45, Number 1 - SpringerLink". Em: *Machine Learning* 45 (out. de 2001), pp. 5–32. DOI: 10.1023/A:1010933404324.

[16] Teobaldo Bulhões et al. "Branch-and-cut approaches for p-Cluster Editing". Em: *Discrete Applied Mathematics* 219 (2017a), pp. 51–64. ISSN: 0166-218X. DOI: https://doi.org/10.1016/j.dam.2016.10.026.

[17] Teobaldo Bulhões et al. "Branch-and-price for p-cluster editing". Em: *Computational Optimization and Applications* 67 (2017b), pp. 293–316. DOI: 10.1007/s10589-017-9893-x.

[18] Mihail Busu e Cristian Busu. "Detecting Bid-Rigging in Public Procurement. A Cluster Analysis Approach". Em: *Administrative Sciences* 11 (2021). ISSN: 2076-3387. DOI: 10.3390/admsci11010013.

[19] Bahzad Charbuty e Adnan Abdulazeez. "Classification based on decision tree algorithm for machine learning". Em: *Journal of Applied Science and Technology Trends* 2.01 (2021), pp. 20–28.

[20] Anil Chaturvedi, Paul E Green e J Douglas Caroll. "K-modes clustering". Em: *Journal of classification* 18 (2001), pp. 35–55.

[21] Corinna Cortes e Vladimir Vapnik. "Support-Vector Networks". Em: *Machine Language* 20.3 (1995), 273–297. ISSN: 0885-6125. DOI: 10.1023/A:1022627411411. URL: https://doi.org/10.1023/A:1022627411411.

[22] T. Cover e P. Hart. "Nearest neighbor pattern classification". Em: *IEEE Transactions on Information Theory* 13.1 (1967), pp. 21–27. DOI: 10.1109/TIT.1967.1053964.

[23] Mark S Daskin, Kayse Lee Maass et al. "Chapter 2-The p-Median Problem". Em: *Locationscience. Springer.* (2015). DOI: https://doi.org/10.1007/978-3-319-13111-5_2.

[24] Marcelo Dib Cruz. "O Problema de Clustrização Automática". Tese de dout. Universidade Federal do Rio de Janeiro, jul. de 2010.

[25] Tom Fawcett. "An introduction to ROC analysis". Em: *Pattern Recognition Letters* 27.8 (2006). ROC Analysis in Pattern Recognition, pp. 861–874. ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2005.10.010.

[26] M. A. Gluck e J. E. Corter. "Information Uncertainty and the Utility of Categories". Em: *Proceedings of the Seventh Annual Conference of Cognitive Science Society*. 1985, pp. 283–287.

[27] K. Grabczewski e N. Jankowski. "Feature selection with decision tree criterion". Em: *Fifth International Conference on Hybrid Intelligent Systems (HIS'05)*. 2005. DOI: 10.1109/ICHIS.2005.43.

[28] Harsha Gwalani, Chetan Tiwari e Armin R. Mikler. "Evaluation of heuristics for the p-median problem: Scale and spatial demand distribution". Em: *Computers, Environment and Urban Systems* 88 (2021), p. 101656. ISSN: 0198-9715. DOI: https://doi.org/10.1016/j.compenvurbsys.2021.101656.

[29] Pierre Hansen e Brigitte Jaumard. "Cluster Analysis and Mathematical Programming". Em: *Math. Program.* 79 (out. de 1997). DOI: 10.1007/BF02614317.

[30] Alberto Heimler. "CARTELS IN PUBLIC PROCUREMENT". Em: *Journal of Competition Law & Economics* 8.4 (dez. de 2012), pp. 849–862. ISSN: 1744-6414. DOI: 10.1093/joclec/nhs028.

[31] Holger H. Hoos e Thomas Stützle. *Stochastic Local Search : Foundations & Applications*. Morgan Kaufmann, 2004. ISBN: 1558608729.

[32] Joshua Zhexue Huang. "A Fast Clustering Algorithm to Cluster Very Large Categorical Data Sets in Data Mining". Em: *Workshop on Research Issues on Data Mining and Knowledge Discovery*. 1997.

[33] Zhexue Huang. "Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values". Em: *Data Mining and Knowledge Discovery* 2.3 (1998), pp. 283–304. DOI: 10.1023/a:1009769707641. URL: https://doi.org/10.1023\%2Fa\%3A1009769707641.

[34] Martin Huber e David Imhof. "Machine learning with screens for detecting bid-rigging cartels". Em: *International Journal of Industrial Organization* 65 (2019), pp. 277–301.

[35] David Imhof, Yavuz Karagök e Samuel Rutz. "SCREENING FOR BID RIGGING DOES IT WORK?" Em: *Journal of Competition Law & Economics* 14.2 (jul. de 2018), pp. 235–261. ISSN: 1744-6414.

[36] David Imhof et al. *Simple Statistical Screens to Detect Bid Rigging Acknowledgement*. 2017.

[37] Paul Jaccard. "Etude de la distribution florale dans une portion des Alpes et du Jura". Em: *Bulletin de la Societe Vaudoise des Sciences Naturelles* 37 (jan. de 1901), pp. 547–579. DOI: 10.5169/seals-266450.

[38] A. H. Land e A. G. Doig. "An Automatic Method of Solving Discrete Programming Problems". Em: *Econometrica* 28 (1960), pp. 497–520.

[39] François-Joseph Lapointe e Pierre Legendre. "A Classification of Pure Malt Scotch Whiskies". Em: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 43.1 (1994), pp. 237–257. DOI: https://doi.org/10.2307/2986124.

[40] David G. Luenberger e Yinyu Ye. *Linear and Nonlinear Programming*. Springer Publishing Company, Incorporated, 2015. ISBN: 3319188410.

[41] J. B. MacQueen. "Some Methods for Classification and Analysis of MultiVariate Observations". Em: *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*. Ed. por L. M. Le Cam e J. Neyman. Vol. 1. University of California Press, 1967, pp. 281–297.

[42] Mohamed Nadif e Gerard Govaert. *Co-clustering: models, algorithms and applications.* John Wiley & Sons, fev. de 2013. ISBN: 978-1-848-21473-6.

[43] Gbeminiyi John Oyewole e George Alex Thopil. "Data clustering: application and trends". Em: *Artificial Intelligence Review* (2022). ISSN: 15737462. DOI: 10.1007/s10462-022-10325-y.

[44] Hae-Sang Park e Chi-Hyuck Jun. "A simple and fast algorithm for K-medoids clustering". Em: *Expert Systems with Applications* 36.2, Part 2 (2009), pp. 3336–3341. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2008.01.039.

[45] Foster Provost e Tom Fawcett. *Data Science for Business*. USA: O'Reilly Media, Inc., 2013. ISBN: 9781449361327.

[46] Hilton Martins Brito Ramalho, Aléssio Tony Cavalcanti de Almeida e Alcimar Alves Fraga. "Detecção de Casos Suspeitos de Conluio em Licitações Públicas: uma Aplicação do Algoritmo A priori de Aprendizado de Máquina para o Estado da Paraíba". Em: *Teoria e Prática em Administração* 10 (2 ago. de 2020), pp. 5–22. DOI: 10.21714/2238-104x2020v10i2-51526.

[47] Manuel J. García Rodríguez et al. "Collusion detection in public procurement auctions with machine learning algorithms". Em: *Automation in Construction* 133 (jan. de 2022). ISSN: 09265805. DOI: 10.1016/j.autcon.2021.104047.

[48] Peter J. Rousseeuw. “Silhouettes: A graphical aid to the interpretation and validation of cluster analysis”. Em: *Journal of Computational and Applied Mathematics* 20 (1987), pp. 53–65. ISSN: 0377-0427. DOI: https://doi.org/10.1016/0377-0427(87)90125-7.

[49] Stuart Russell e Peter Norvig. *Artificial Intelligence: A Modern Approach*. 3rd. USA: Prentice Hall Press, 2009. ISBN: 0136042597.

[50] Yutaka Sasaki. “The truth of the F-measure”. Em: (jan. de 2007).

[51] Regis Signor, Peter E. D. Love e Lavagnon A. Ika. “White Collar Crime: Unearthing Collusion in the Procurement of Infrastructure Projects”. Em: *IEEE Transactions on Engineering Management* 69.5 (2022), pp. 1932–1943. DOI: 10.1109/TEM.2020.2994636.

[52] Regis Signor et al. “Detection of Collusive Tenders in Infrastructure Projects: Learning from Operation Car Wash”. Em: *Journal of Construction Engineering and Management* 146 (1 jan. de 2020). ISSN: 0733-9364. DOI: 10.1061/(asce)co.1943-7862.0001737.

[53] Kristina P. Sinaga e Miin-Shen Yang. “Unsupervised K-Means Clustering Algorithm”. Em: *IEEE access* 8 (2020), pp. 80716–80727. ISSN: 2169-3536.

[54] Gilberto F. Sousa Filho et al. “New heuristics for the Bicluster Editing Problem”. Em: *Annals of Operations Research* 258 (2017), pp. 781–814. DOI: 10.1007/s10479-016-2261-x.

[55] Gilberto F. Sousa Filho et al. “The biclique partitioning polytope”. Em: *Discrete Applied Mathematics* 301 (2021), pp. 118–130. ISSN: 0166-218X. DOI: https://doi.org/10.1016/j.dam.2021.05.023.

[56] *Índice de Percepção da Corrupção*. 2022. URL: https://transparenciainternacional.org.br/ipc/ (acesso em 11/03/2023).

APÊNDICE 1 - HIPERPARÂMETROS

A seguir são apresentados são valores selecionados para os hiperparâmetros, para cada combinação entre conjunto de dados, algoritmo de aprendizagem de máquina e configuração do conjunto de dados. Este apêndice se divide da seguinte forma: a Seção 7.1 descreve os valores para os dados do Brasil, 7.2 os dados do Japão, 7.3 os da Itália, e 7.4 os Estados Unidos.

7.1 Brasil

7.1.1 Configuração 1

- SVM: $C = 1$, $\gamma = \frac{1}{|F|}$;
- K-NN: $k = 3$, $p = 1$, $w =$ uniforme;
- NN: $\theta = 0.001$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas $= (24, 1)$, $\alpha = 0.0001$;
- RF: número de classificadores = 199, profundidade máxima da árvore = 3, número mínimo de objetos em um nó folha = 1;
- DT: $\alpha = 0.04376$;

7.1.2 Configuração 2

- SVM: $C = 2$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 7$, $p = 3$, $w =$ uniforme;
- NN: $\theta = 0.001$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas $= (24, 1)$, $\alpha = 0.1$;
- RF: número de classificadores = 29, profundidade máxima da árvore = 4, número mínimo de objetos em um nó folha = 1;
- DT: $\alpha = 0.01537$;

7.1.3 Configuração 3

- SVM: $C = 2$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 7$, $p = 1$, $w =$ uniforme;

- NN: $\theta = 0.1$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(8, 4, 2, 1)$, $\alpha = 0.1$;
- RF: número de classificadores = 29, profundidade máxima da árvore = 3, número mínimo de objetos em um nó folha = 2;
- DT: $\alpha = 0.02066$;

7.1.4 Configuração 4

- SVM: $C = 1$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 1$, $p = 2$, $w =$ uniforme;
- NN: $\theta = 0.01$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(100, 1)$, $\alpha = 0.0001$;
- RF: número de classificadores = 99, profundidade máxima da árvore = ilimitada, número mínimo de objetos em um nó folha = 1;
- DT: $\alpha = 0.0$;

7.1.5 Configuração 5

- SVM: $C = 1$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 3$, $p = 4$, $w =$ proporcional;
- NN: $\theta = 0.01$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(100, 1)$, $\alpha = 0.0001$;
- RF: número de classificadores = 29, profundidade máxima da árvore = 5, número mínimo de objetos em um nó folha = 1;
- DT: $\alpha = 0.01537$;

7.1.6 Configuração 6

- SVM: $C = 1$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 3$, $p = 1$, $w =$ uniforme;
- NN: $\theta = 0.01$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(100, 1)$, $\alpha = 0.0001$;

- RF: número de classificadores = 199, profundidade máxima da árvore = 3, número mínimo de objetos em um nó folha = 1;
- DT: $\alpha = 0.06538$;

7.2 Japão

7.2.1 Configuração 1

- SVM: $C = 2$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 7$, $p = 1$, $w =$ proporcional;
- NN: $\theta = 0.001$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(100, 1)$, $\alpha = 0.0001$;
- RF: número de classificadores = 199, profundidade máxima da árvore = 3, número mínimo de objetos em um nó folha = 3;
- DT: $\alpha = 0.00672$;

7.2.2 Configuração 2

- SVM: $C = 2$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 7$, $p = 1$, $w =$ uniforme;
- NN: $\theta = 0.1$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(24, 1)$, $\alpha = 0.001$;
- RF: número de classificadores = 29, profundidade máxima da árvore = 5, número mínimo de objetos em um nó folha = 1;
- DT: $\alpha = 0.00551$;

7.2.3 Configuração 3

- SVM: $C = 1$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 5$, $p = 2$, $w =$ uniforme;
- NN: $\theta = 0.001$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(8, 4, 2, 1)$, $\alpha = 0.1$;
- RF: número de classificadores = 29, profundidade máxima da árvore = 5, número mínimo de objetos em um nó folha = 3;
- DT: $\alpha = 0.00406$;

7.2.4 Configuração 4

- SVM: $C = 2$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 5$, $p = 1$, $w =$ proporcional;
- NN: $\theta = 0.01$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(100, 1)$, $\alpha = 0.0001$;
- RF: número de classificadores = 29, profundidade máxima da árvore = ilimitada, número mínimo de objetos em um nó folha = 3;
- DT: $\alpha = 0.00225$;

7.2.5 Configuração 5

- SVM: $C = 1$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 7$, $p = 1$, $w =$ uniforme;
- NN: $\theta = 0.1$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(100, 1)$, $\alpha = 0.001$;
- RF: número de classificadores = 199, profundidade máxima da árvore = 7, número mínimo de objetos em um nó folha = 1;
- DT: $\alpha = 0.00551$;

7.2.6 Configuração 6

- SVM: $C = 2$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 3$, $p = 1$, $w =$ uniforme;
- NN: $\theta = 0.01$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(24, 1)$, $\alpha = 0.1$;
- RF: número de classificadores = 29, profundidade máxima da árvore = ilimitada, número mínimo de objetos em um nó folha = 1;
- DT: $\alpha = 0.00347$;

7.3 Itália

7.3.1 Configuração 1

- SVM: $C = 2$, $\gamma = \frac{1}{|F|}$;
- K-NN: $k = 5$, $p = 3$, $w =$ uniforme;
- NN: $\theta = 0.01$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(8, 4, 2, 1)$, $\alpha = 0.001$;
- RF: número de classificadores = 29, profundidade máxima da árvore = ilimitada, número mínimo de objetos em um nó folha = 3;
- DT: $\alpha = 0.00601$;

7.3.2 Configuração 2

- SVM: $C = 2$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 5$, $p = 1$, $w =$ proporcional;
- NN: $\theta = 0.01$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(8, 4, 2, 1)$, $\alpha = 0.1$;
- RF: número de classificadores = 29, profundidade máxima da árvore = ilimitada, número mínimo de objetos em um nó folha = 2;
- DT: $\alpha = 0.00686$;

7.3.3 Configuração 3

- SVM: $C = 2$, $\gamma = \frac{1}{|F|}$;
- K-NN: $k = 1$, $p = 2$, $w =$ uniforme;
- NN: $\theta = 0.01$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(8, 4, 2, 1)$, $\alpha = 0.0001$;
- RF: número de classificadores = 29, profundidade máxima da árvore = ilimitada, número mínimo de objetos em um nó folha = 1;
- DT: $\alpha = 0.0$;

7.3.4 Configuração 4

- SVM: $C = 0.1$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 7$, $p = 1$, $w =$ uniforme;
- NN: $\theta = 0.01$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(8, 4, 2, 1)$, $\alpha = 0.0001$;
- RF: número de classificadores = 199, profundidade máxima da árvore = 4, número mínimo de objetos em um nó folha = 3;
- DT: $\alpha = 0.02317$;

7.3.5 Configuração 5

- SVM: $C = 2$, $\gamma = \frac{1}{|F|}$;
- K-NN: $k = 5$, $p = 1$, $w =$ proporcional;
- NN: $\theta = 0.01$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(24, 1)$, $\alpha = 0.0001$;
- RF: número de classificadores = 99, profundidade máxima da árvore = ilimitada, número mínimo de objetos em um nó folha = 1;
- DT: $\alpha = 0.00686$;

7.3.6 Configuração 6

- SVM: $C = 1$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 7$, $p = 1$, $w =$ proporcional;
- NN: $\theta = 0.01$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(24, 1)$, $\alpha = 0.1$;
- RF: número de classificadores = 199, profundidade máxima da árvore = ilimitada, número mínimo de objetos em um nó folha = 1;
- DT: $\alpha = 0.0$;

7.4 Estados Unidos

7.4.1 Configuração 1

- SVM: $C = 2$, $\gamma = \frac{1}{|F| \cdot var(x)}$;
- K-NN: $k = 7$, $p = 3$, $w =$ uniforme;
- NN: $\theta = 0.1$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas $= (100, 1)$, $\alpha = 0.1$;
- RF: número de classificadores = 29, profundidade máxima da árvore = 7, número mínimo de objetos em um nó folha = 3;
- DT: $\alpha = 0.00328$;

7.4.2 Configuração 3

- SVM: $C = 2$, $\gamma = \frac{1}{|F|}$;
- K-NN: $k = 3$, $p = 1$, $w =$ proporcional;
- NN: $\theta = 0.01$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas $= (100, 1)$, $\alpha = 0.0001$;
- RF: número de classificadores = 99, profundidade máxima da árvore = ilimitada, número mínimo de objetos em um nó folha = 1;
- DT: $\alpha = 0.0$;

7.4.3 Configuração 4

- SVM: $C = 1$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 3$, $p = 1$, $w =$ proporcional;
- NN: $\theta = 0.1$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas $= (100, 1)$, $\alpha = 0.0001$;
- RF: número de classificadores = 99, profundidade máxima da árvore = 4, número mínimo de objetos em um nó folha = 1;
- DT: $\alpha = 0.00061$;

7.4.4 Configuração 5

- SVM: $C = 2$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 7$, $p = 1$, $w =$ uniforme;
- NN: $\theta = 0.01$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(100, 1)$, $\alpha = 0.1$;
- RF: número de classificadores = 29, profundidade máxima da árvore = 4, número mínimo de objetos em um nó folha = 1;
- DT: $\alpha = 0.00151$;

7.4.5 Configuração 6

- SVM: $C = 2$, $\gamma = \frac{1}{|F| \cdot var(X)}$;
- K-NN: $k = 7$, $p = 1$, $w =$ proporcional;
- NN: $\theta = 0.01$, taxa de aprendizagem = adaptativa, número de neurônios nas camadas ocultas = $(100, 1)$, $\alpha = 0.1$;
- RF: número de classificadores = 29, profundidade máxima da árvore = ilimitada, número mínimo de objetos em um nó folha = 1;
- DT: $\alpha = 0.00231$;