DAIANY FRANCISCA LARA

ESTUDOS EMPÍRICOS DOS MÉTODOS DE BALANCEAMENTO PARA A CLASSIFICAÇÃO

CURITIBA

2013

DAIANY FRANCISCA LARA

ESTUDOS EMPÍRICOS DOS MÉTODOS DE BALANCEAMENTO PARA A CLASSIFICAÇÃO

Dissertação apresentada como requisito para à obtenção do grau de Mestre no Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.
Orientadora: Profa. Dra. Aurora Trinidad Ramirez Pozo

CURITIBA

2013

DAIANY FRANCISCA LARA

ESTUDOS EMPÍRICOS DOS MÉTODOS DE BALANCEAMENTO PARA A CLASSIFICAÇÃO

Dissertação aprovada como requisito para à obtenção do grau de Mestre no Programa de Pós-Graduação em Informática da Universidade Federal do Paraná, pela Comissão formada pelos professores:

Orientadora: Profa. Dra. Aurora Trinidad Ramirez Pozo
Departamento de Informática, UFPR

Profa. Dra. Alaine Margarete Guimarães
Universidade Federal de Ponta Grossa, UFPG

Prof. Dr. Andrey Ricardo Pimentel
Departamento de Informática, UFPR

Curitiba, 1 de Julho de 2013

AGRADECIMENTOS

Gostaria de agradecer primeiramente a Deus por me amparar nos momentos difíceis, dando força interior para continuar e superar as dificuldades.

À minha orientadora Aurora Trinidad Ramirez Pozo, pela paciência e compreensão, e por ser a maior incentivadora para a superação dos meus limites, e por ser um exemplo de profissional.

Ao meu namorado Robson, pessoa que sempre me ajudou nos momentos mais difíceis, esteve sempre ao meu lado, demonstrando carinho, amizade e amor.

Aos meus pais Alfeu Picada de Lara e Evanildes Francisca Lara, que mesmo distantes, estiveram sempre comigo, apoiando-me, amando-me incondicionalmente. Eu amo vocês!

Às minhas irmãs Glaucilene e Glaciane, que me deram incentivo para continuar estudando, e me apoiando na vida pessoal e profissional.

Muito Obrigada, a todos que fizeram parte desse momento tão importante na minha vida.

SUMÁRIO

RESUMO

A classificação tem o objetivo de rotular eventos e objetos de acordo com classes pré-estabelecidas. No entanto, alguns algoritmos perdem a capacidade de predição, quando o conjunto de dados possui uma distribuição desbalanceada entre suas classes. Para tentar resolver esse problema diversos métodos têm sido propostos na literatura.

O presente trabalho tem como objetivo analisar e comparar os métodos mais conhecidos que se propõe a resolver o problema de classificação com bases desbalanceadas. Para isto, os métodos foram testados com os classificadores tradicionais como: *Naive Bayes*, *Bayes Net*, *SMO*, *MultilayerPerceptron*, *J48* e *JRip*. As métricas de avaliação consideradas foram RecallP (verdadeiros positivos), RecallN (Verdadeiros negativos) e finalmente a taxa de acurácia. Para realizar esta análise, os testes foram efetuados em 13 bases provenientes do UCI *Machine Learning Repository* e também em dois conjuntos de bases do "mundo real", que são bases construídas com informações sobre defeitos em sistemas de Orientação a Aspectos. O primeiro conjunto são cinco bases do repositório *NASA Metrics Data Project*, sendo elas cm1, jm1, kc1, kc2 e pc1. O segundo conjunto, são três sistemas Orientados a Aspecto que são: Ibatis, HW (HealthWatcher) e MM (MobileMedia).

Os resultados demonstram que é possível melhorar a taxa de classificação, mas é difícil dizer o método que se comporta melhor em bases do mundo real, pois tudo depende de como o classificador generaliza a base, principalmente com a presença de dados ruidosos. As bases do UCI, apresentam melhores resultados em relação às bases de Engenharia de Software. Isto pode ser explicado em função da natureza dos dados reais que costumam conter mais ruídos.

ABSTRACT

The classification aims at labeling objects and events according to pre-established classes. However, some algorithms lose the ability to predict when the data set has an imbalanced distribution between classes. To attack this problem various methods have been proposed in the literature.

This work aims to analyze and compare the most popular methods that attempt to solve the classification problem with imbalanced bases. For this, the methods was tested with the traditional classifiers: Naive Bayes, Bayes Net, SMO, Multilayer Perceptron, J48 and JRip. The evaluation metrics RecallP (True Positive), RecallN (True Negatives) and accuracy rate were considered. To accomplish this analyze, tests were made using 13 databases from UCI Machine Learning Repository and also with two sets of databases of real world, wich are built with information about faulty in Aspect-Oriented System. The first set are five databases from NASA Metrics Data Project, they are: cm1, jm1, kc1, kc2 and pc1. The second set, are three Aspect-Oriented System that are: Ibatis, HW (HeathWatcher) and MM (MobileMedia).

The results show that is harder to determine the method that behave better in databases of real world, because all depends how the classifier generalizes the database mainly in the presence of noisy data. The UCI databases present better results compared to the results obtained with databases Engineering Software. This difference in the results can be explained by the nature of real data that often have more noise.

CAPÍTULO 1

INTRODUÇÃO

Aprendizagem de Máquina (AM) é uma área da Inteligência Artificial (IA) responsável por lidar com os problemas computacionais voltados para a aquisição automática de conhecimento. Um sistema de aprendizado tem como característica a tomada de decisão com base no conhecimento prévio acumulado através da interação com o ambiente. Para tal, são usadas técnicas computacionais para automatizar o aprendizado [16].

Um dos princípios utilizados pelo aprendizado é a inferência por indução. A principal tarefa da indução é generalizar conceitos a partir de fatos observados (exemplos). Para a indução aprender conhecimento novo representativo, os exemplos das classes têm que estar bem definidos e deve haver uma quantidade suficiente deles. O aprendizado indutivo é dividido em dois grupos principais: Aprendizagem não-supervisionada e Aprendizagem supervisionada [16].

A aprendizagem não-supervisionada, não faz uso do atributo de saída, ou seja, não trabalha com um exemplo rotulado [16]. A aprendizagem supervisionada que é empregada neste trabalho, tem como objetivo induzir conceito a partir de exemplos que estão rotulados como uma classe conhecida, ou seja, dado um conjunto de exemplos rotulados, deve-se produzir um modelo que se chama classificador, e este deve ser capaz de predizer precisamente a classe de novos dados com base nos dados anteriormente classificados [16].

As informações geradas pelos processos automatizados tem aumentado de forma considerável e estão dispostas e agrupadas de maneira irregular e desbalanceada, tornando difícil o entendimento dessas informações. De forma a aprimorar a análise dessas informações, algoritmos de aprendizado supervisionado como os classificadores são frequentemente utilizados [22]. Contudo, a utilização de bases com a distribuição desbalanceada entre as classes representa um dos aspectos que tem comprometido significativamente o desempenho dos algoritmos de classificação [4].

Muitos sistemas de aprendizado não estão preparados para induzir bons modelos de classificação na presença de classes desbalanceadas. Quando treinados com esse tipo de base, o desempenho dos classificadores é reduzido significativamente, pois normalmente apresentam uma boa acurácia para a classe majoritária, mas uma acurácia baixa para a classe minoritária. O que pode ser um problema quando a classe de interesse é justamente a classe minoritária [24].

Problemas com classes desbalanceadas estão presentes em inúmeros domínios do mundo real e tem atraído o interesse de muitos pesquisadores nos últimos anos [4], [24], [19], [23]. Uma área bastante comum de se encontrar bases desbalanceadas é a Engenharia de Software, principalmente quando são bases construídas coletando-se métricas específicas para módulos de sistemas Orientados a Aspectos (OA) com o objetivo de determinar a relação dessas métricas com a propensão a defeitos.

Trabalhos encontrados na literatura apresentam métodos para tentar resolver o problema da classificação com classes desbalanceadas, tentando evitar os problemas causados pelos métodos aleatórios *Oversampling e Undersampling* [21][13][4]. Na Maioria dos trabalhos relacionados, os testes dos métodos de balanceamento: *Adasyn, Borderline, Smote* assim como os métodos *Oversampling e Undersampling*, são realizados com no máximo três algoritmos de classificação. Os classificadores mais utilizados são: a árvore de decisão *C4.5*, o classificador *NaiveBayes* e RIPPER. Na maioria das vezes a métrica de avaliação utilizada é apenas a acurácia, o que pode ser um problema, pois nem sempre os métodos conseguem melhorar a classe minoritária, as vezes a melhora acontece apenas na classe majoritária, que consequentemente melhora a acurácia. Os testes desses trabalhos encontrados na literatura foram realizados somente com bases provenientes do repositório UCI *Machine Learning Repository*, o que normalmente são previamente pré-processados, sem muitos dos problemas que podem ser encontrados em dados do "mundo real" [3].

Este trabalho faz uma análise comparativa dos principais métodos encontrados na literatura, com o intuito de apresentar os métodos que realmente melhoram a taxa de acurácia e principalmente os valores dos verdadeiros positivos (Métrica de avaliação RecallP) representada pela classe minoritária, e também melhorar, ou no mínimo manter os

resultados da classe majoritária. Para tanto, pretende-se testar esses métodos com classificadores que neste trabalho são chamados de tradicionais, como: *Naive Bayes*, *Bayes Net*, *SMO*, *MultilayerPerceptron*, *J48* e *JRip*. As métricas de avaliação utilizadas são: RecallP (Verdadeiros positivos), RecallN (Verdadeiros negativos) e Acurácia. Os resultados serão demonstrados de acordo com os valores dessas métricas, pois dessa forma pode-se observar o que realmente acontece em cada classe. Os testes foram realizados com 13 bases provenientes do repositório UCI, e também com dois conjuntos de bases do "mundo real", que foram construídas coletando-se informações sobre defeitos encontrados em sistemas orientados a aspectos. O primeiro conjunto é composto por cinco bases do repositório *NASA Metrics Data Project* (cm1, jm1, kc1, kc2 e pc1). O segundo refere-se aos três sistemas Orientados a Aspectos: Ibatis, HW (HealthWatcher) e MM (MobileMedia).

Os métodos de balanceamento utilizados para a análise na fase de pré-processamento apresentados neste trabalho são: *Adasyn, Borderline, Smote, Oversampling e Undersampling*. A análise é feita também com os métodos conhecidos como Híbridos, pois faz uma junção dos métodos de balanceamento na fase de pré-processamento com a classificação, sendo eles: *BalanceCascade, EasyEnsemble, SMOTEBoost e SMOTEBoosting*.

O objetivo deste trabalho, de acordo com os resultados obtidos nos experimentos, é responder algumas questões que são de grande importância, como:

- A Classificação pode ser melhorada aplicando algum método de balanceamento?
- Existe um método que sempre se comporta bem?
- Os classificadores podem influenciar nos resultados?
- Quais são as métricas de avaliação que realmente apontam a melhora da classe minoritária?

O trabalho está organizado da seguinte forma: No capítulo 2 são apresentados os trabalhos relacionados. No Capítulo 3 são apresentadas as etapas do processo extração de conhecimento e as métricas de avaliação. No capítulo 4 estão relacionados os problemas causados pelo conjunto de dados desbalanceados e os métodos para resolver esses

problemas. As ferramentas e classificadores utilizados neste trabalho, bem como os primeiros experimentos com bases do UCI são apresentados no capitulo 5. Experimentos em dois conjuntos de bases do mundo real são apresentados no Capítulo 6, e finalmente as conclusões deste trabalho são apresentadas no Capítulo 7.

CAPÍTULO 2

TRABALHOS RELACIONADOS

Este capítulo descreve trabalhos relacionados a técnicas utilizados para bases desbalanceadas. A pesquisa dos trabalhos relacionados foi realizada considerando as seguintes bases bibliográficas: IEEE Xplore, ACM Digital Library, Google Scholar e outros, foram utilizadas palavras chaves como: Classificação em bases desbalanceadas, métodos de balanceamento, classes desbalanceadas, mineração de dados e outros. Os trabalhos foram filtrados do ano de 2000 até o presente ano. A seguir descreve-se os finais relevantes.

O trabalho de Prati [35] faz um estudo sistemático, com o objetivo de questionar o real problema de indução do classificador com classes desbalanceadas, ou seja, verificar se esta característica pode ser explorada por outros caminhos. Os autores acreditam que nem sempre as classes desbalanceadas são o problema. Para provar esta hipótese, se utilizou bases que possuem duas classes, uma majoritária e uma minoritária. Os experimentos foram realizados em conjuntos de dados artificiais, cujas características os autores são capazes de controlar, obtendo uma interpretação completa dos resultados. Esses dados artificiais possuem dois parâmetros controladores, o primeiro controle é analisar a distância entre os centroides de dois agrupamentos, um representando a classe majoritária e outro representando a classe minoritária, o objetivo é controlar o nível de dificuldade de classificar corretamente as duas classes. O segundo controle é o grau de desbalanceamento, o intuito é observar o fator do desempenho degradante do classificador. Os resultados mostram que o problema não está diretamente ligado a classes desbalanceadas, mas está relacionado também com o grau de sobreposição entre as classes.

Japkowicz [26] faz um estudo para distinguir dois tipos de desbalanceamento: desbalanceamento entre classes, que corresponde a diferença entre os números representados pela classe positiva e os números de exemplos representados pela classe negativa; Desbalanceamento dentro da classe, que corresponde ao caso de uma única classe ser composta

por vários sub-agrupamentos de diferentes tamanhos. O objetivo era mostrar que ambos os problemas contribuem para o aumento da taxa de classificação incorreta dos classificadores. Os experimentos foram realizados utilizando um conjunto de dados de reconhecimento de letras, e foi utilizado o algoritmo de agrupamento em que cada classe identifica os sub-agrupamentos que a constituem. Também foi utilizado o método resampling, no qual é feita a reamostragem (resampling) em que cada sub-agrupamento da maior classe, atinja o tamanho do maior sub-agrupamento da classe. Em um trabalho realizado posteriormente Japkowicz [27] faz um questionamento se o problema de classes desbalanceadas é realmente o culpado pela perda de desempenho. A proposta foi estender a abordagem anterior, ou seja, tratar os dois problemas simultaneamente utilizando uma estratégia de Rebalanceamento e demonstrar que um método simples para lidar com o problema, pode ser útil no caso de sub-agrupamentos drasticamente desbalanceados. Os experimentos foram realizados apenas nos conjuntos de dados de reconhecimento de letras.

Um dos métodos para tratar o problema em bases desbalanceadas foi apresentado por Chawla et.al. [14]. O método conhecido como SMOTE *(Synthetic Minority Oversampling Technique)*, tem como objetivo criar exemplos sintéticos na classe de interesse, que geralmente é a classe minoritária. O trabalho mostra também que a combinação do método Smote e *Undersampling* (elimina exemplos da classe majoritária), pode alcançar melhor desempenho do classificador do que utilizar somente um deles. Os experimentos foram realizados utilizando o *C4.5*, *Ripper* e um classificador *Naive Bayes*, e avaliados pela medida de avaliação AUC (Área sob a curva de ROC). Um dos pontos fracos do método SMOTE é que todas as instâncias da classe minoritária servem como base para a geração das instâncias sintéticas. No entanto, tal estratégia não leva em conta que nem sempre uma distribuição homogênea de instâncias sintéticas é aplicável a um problema de desbalanceamento, considerando que tal prática pode causar o problema de sobreajustamento e sobreposição de classes [4].

No trabalho de Chawla e Lazarevic [13] apresenta-se uma abordagem que faz uma combinação do algoritmo SMOTE e o processo padrão *Boosting*, denominado SMOTEBoost. O SMOTEBoost cria exemplos sintéticos na classe minoritária a cada iteração

Boosting, mudando os pesos de atualização e de compensação para distribuição heterogênea, ao contrário do Boosting padrão que usa pesos iguais para todos os exemplos classificados incorretamente. O método tem como objetivo melhorar os verdadeiros positivos (TP). Os resultados apresentam uma melhora no desempenho da precisão sobre a classe minoritária.

No trabalho de Han, Hui [21], diz que os exemplos próximos a borda de decisão são mais propensos a serem incorretamente classificados que os exemplos que estão longe dessa borda, e esses são mais importantes para a classificação. Por esse motivo criou-se um método baseado no SMOTE, denominado Borderline-SMOTE. Este método identifica os exemplos próximos a fronteira de decisão formando um conjunto conhecido como *Danger* e assim aplica o método SMOTE somente neste conjunto. O objetivo consiste em não criar instâncias sintéticas próximas a superfície de decisão, para não criar ruídos no treinamento. Este método se comporta melhor em relação ao SMOTE tradicional, pois consegue melhorar a taxa dos verdadeiros positivos.

Haibo He et.al [23] apresentaram um método adaptativo para facilitar a aprendizagem em conjunto de dados desbalanceados, e principalmente, reduzir a tendência de introduzir problemas como ruídos no conjunto de dados, e também adaptativamente deslocar a fronteira de decisão para focar nas amostras que são difíceis de aprender. O método ADASYN também baseado no SMOTE, usa uma distribuição ponderada como critério para decidir o número de amostras sintéticas que devem ser geradas para cada exemplo da classe minoritária. Este método mostra uma eficácia nas métricas de avaliação *Recall*, Precisão e *F-value*.

Em Liu, Xu-Yang et.al [29], dois métodos foram propostos com o objetivo de superar a deficiência do *Undersampling*, que muitas vezes perde informações importantes pertencentes à classe majoritária. Os métodos são: *EasyEnsemble* que é considerado uma abordagem não supervisionada, pois explora o conjunto majoritário através de amostragem aleatória com troca, e *BalanceCascade* que explora o conjunto majoritário de forma supervisionada, e cuja ideia é remover exemplos da classe principal classificados corretamente a partir de uma análise mais aprofundada. Os resultados mostram que ambos

os métodos possuem melhores valores de AUC, além de serem superiores no tempo de treinamento em relação aos outro métodos.

Para compensar o desbalanceamento das classes, Beckmann [5] apresenta um algoritmo genético para Oversampling, que cria instâncias sintéticas minoritárias orientadas por um processo evolutivo. Para balancear o conjunto da base, o algoritmo preenche com instâncias sintéticas positivas as regiões que foram ajustadas e posicionadas dentro dos limites da classe minoritária. O objetivo é não permitir instâncias duplicadas ou similares às existentes, desta forma, evitando a sobreposição de regiões, o que nos dois casos ocorreria o *overfitting*. A função de aptidão consiste em otimizar a medida AUC obtida no processo de classificação. A execução dos experimentos foi realizada com 16 bases provenientes do UCI *machine learning repository* e de acordo com os experimentos do algoritmo genético proposto, foram apresentados melhores resultados no desempenho da classificação em comparação com os resultados obtidos com os métodos ADASYN e Borderline-Smote [21].

Outro trabalho interessante é de Drown e Khoshgoftaar[15], pois utiliza um método de algoritmo genético baseado na amostragem evolutiva, como uma solução para resolver o problema de classe desbalanceada na modelagem de qualidade de software. O método trabalha com a redução dos exemplos da classe majoritária (método de Undersampling). O grande diferencial desse algoritmo genético está na função de aptidão, pois combina duas medidas de avaliação: AUC que indica uma taxa maior de verdadeiros positivos e uma menor taxa de falsos positivos se seu valor for elevado e G-mean que representa a precisão do classificador tanto para a classe positiva como negativa. Esse experimento foi comparado com diversos métodos de amostragem como por exemplo: Undersampling, Wilson's editing, one-sided selection, Oversampling, Cluster baseado em Oversampling, SMOTE e Borderline-SMOTE. As bases utilizadas para os experimentos foram dois sistemas de software: CM1 que contém dados de medidas de software da NASA e PS2 que contém métricas de software e dados de defeitos para o lançamento de sistemas de telecomunicações. Os resultados mostram que a abordagem de amostragem evolutiva de dados renderam desempenhos significativamente melhores em comparação com as demais

técnicas de amostragem de dados.

Procurando obter um bom equilíbrio entre a distribuição da classe e o desempenho do algoritmo, o trabalho de García e Herrera [20] apresenta um conjunto de métodos Undersampling evolutivo que utilizam diferentes funções de fitness. O trabalho faz ainda uma comparação geral entre os modelos propostos e o estado da arte dos métodos Undersampling. A análise é feita com 28 bases oriundas do UCI *machine learning repository*. Os resultados foram contrastados usando procedimentos estatísticos não paramétricos e mostram que o Undersampling evolutivo supera os modelos não evolutivos quando o grau de desbalanceamento é aumentado.

Em consequência do estudo realizado até o momento, notou-se que a maioria dos trabalhos não realizam uma avaliação mais ampla considerando vários métodos de balanceamento existentes, não abordam uma gama maior de classificadores tradicionais, assim como o conjunto de bases costuma ser restrito. Portanto o objetivo deste trabalho é comparar os principais métodos de balanceamento, com bases do mundo real, e poder dizer se os métodos realmente melhoram as classes minoritárias e majoritárias.

CAPÍTULO 3

DESCOBERTA DE CONHECIMENTO EM BASE DE DADOS

O processo de Descoberta de Conhecimento em Base de Dados (Knowledge Discovery in Databases - KDD) tem como intuito extrair conhecimento de grandes bases de dados. Este processo utiliza métodos, algoritmos e técnicas que envolvem diversas áreas como: estatística, matemática, banco de dados, inteligência artificial, visualização de dados e reconhecimento de padrões [17][12].

O KDD abrange 5 (cinco) etapas que podem exigir do usuário capacidade de análise e de tomada de decisão, ou seja, é um processo iterativo podendo retornar a qualquer etapa sem precisar atingir o final do processo, e também um processo interativo, por ter a orientação de um especialista e conhecedor do domínio dos dados. As etapas envolvidas para descoberta de conhecimento estão relacionadas na Figura 3.1 [17].

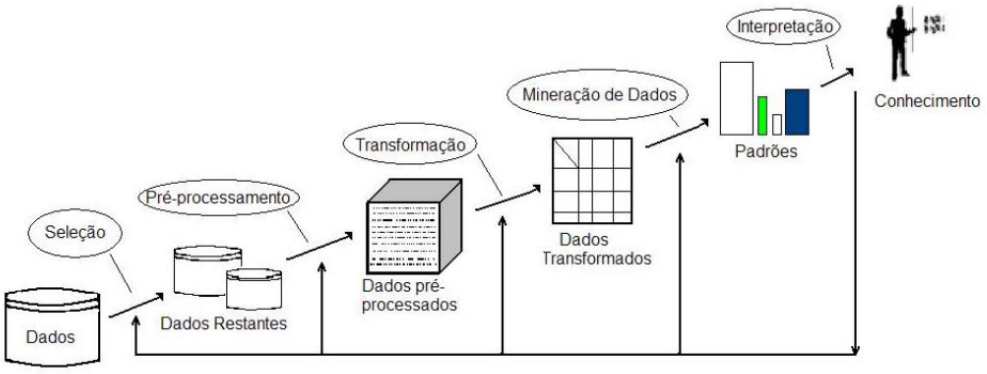

Figura 3.1: Etapas do Processo KDD [17]

- **Seleção -** Etapa que faz uma seleção ou segmentação dos dados apropriados para a análise.

- **Pré-processamento -** Etapa que faz uma limpeza nos dados inválidos, inconsistentes e redundantes, visando adequá-los aos algoritmos que serão utilizados.

- **Transformação -** Para facilitar na utilização de técnicas de mineração de dados,

os dados passam por uma transformação para o armazenamento adequado.

- **Mineração de Dados -** Nesta fase acontece a aplicação dos algoritmos, de acordo com o objetivo do processo KDD: classificação, associação, agrupamento, regressão, entre outros. Fase detalhada na Seção 3.2.

- **Interpretação -** Será gerado um arquivo de descobertas de acordo com o algoritmo utilizado na fase de mineração de dados, e este arquivo deve ser interpretado obtendo as conclusões que fornecem o conhecimento da base de dados estudada.

3.1 Pré-Processamento

A fase de pré-processamento tem o objetivo de aprimorar a qualidade dos dados coletados, visto que frequentemente os dados apresentam diversos problemas, como grande quantidade de valores desconhecidos, ruídos (atributos com valores incorretos), desproporção entre o número de exemplos de cada classe (classes desbalanceadas), dentre outros [37]. Essa melhora pode facilitar o uso de técnicas de Aprendizado de Máquina, facilitando os ajustes de parâmetros do modelo, e principalmente a interpretação do padrões extraídos pelo modelo [16].

A seguir serão mostradas algumas tarefas para selecionar os atributos relevantes da base (Seção 3.1.1). Os métodos de pré-processamento para resolver os problemas de classes desbalanceadas são detalhadas no Capítulo 4.

3.1.1 Seleção de Atributos

A tarefa de seleção de atributos é uma das maneiras utilizadas para lidar com o problema de grande número de atributos na base de dados, pois identifica os atributos mais significativos e quais podem contribuir para a construção de modelos de indução com menor custo computacional.

A seleção de atributos pode ser formulada como um processo de busca que corresponde a um subconjunto ótimo de atributos em um determinado espaço de busca. Este subconjunto é denominado ótimo quando todos os atributos maximizam ou minimizam

um ou mais critérios de importância. A busca realizada por um algoritmo de seleção de atributos pode ser instanciada em relação às questões: sentido de busca, estratégia de busca, definição do critério de parada e de critérios de avaliação de importância de atributos.

As abordagens de avaliação dizem respeito à maneira como os subconjuntos são avaliados, ou seja, dependem da interação com o algoritmo de mineração de dados. As abordagens são [37]:

- **Filtro:** seleciona um subconjunto de atributos na fase de pré-processamento, ou seja, antes do algoritmo de aprendizagem ser aplicado.
- **Wrapper:** utiliza o próprio algoritmo de aprendizagem da mineração de dados para avaliar o subconjunto de atributos;
- **Embedded:** a seleção de atributos é realizada internamente no próprio algoritmo de aprendizagem e avaliada por uma medida independente.

Para medir a qualidade de um atributo é preciso utilizar um tipo de métrica. As principais métricas para essa finalidade são: Consistência, Ganho de informação, Chi-Squared e OneR [33].

3.2 Mineração de Dados

A mineração de dados é umas das etapas mais importantes do processo KDD, diversas definições para esta etapa são encontradas na literatura. Fayyad [17] define a mineração de dados como um processo de identificação de padrões válidos, novos e úteis contidos nos dados. Pode-se dizer que esta é a fase que transforma dados puros em informações úteis [12].

Como diz Fayyad [17], a aplicação cega da mineração de dados pode se tornar uma atividade perigosa e conduzir facilmente para a descoberta de padrões sem sentido. Para escolha da técnica mais adequada é importante saber quais são os atributos mais importantes, quais os relacionamentos possíveis, o que é uma função útil para o usuário, que

padrões já são conhecidos e assim por diante. A mineração consiste na especificação de métodos que nos garantam como descobrir os padrões que nos interessam.

3.2.1 Tarefas de Mineração de Dados

As tarefas de mineração de dados podem extrair diferentes tipos de conhecimento, sendo necessário decidir, já no início do processo de mineração de dados, qual o tipo de conhecimento que o algoritmo deve extrair. Existem dois tipos de padrões e podem estar relacionados tanto as atividades descritivas (não supervisionadas) como atividades preditivas (supervisionadas). As atividades descritivas trabalham com um conjunto de dados que não possuem classe determinada, buscando padrões de comportamento comuns nestes dados. As atividades preditivas buscam identificar tendências futuras (novos dados), ou seja, é descoberto como as coisas acontecem e o que pode estar para acontecer [17].

Este trabalho tem como foco a abordagem supervisionada, mais precisamente a classificação. E para melhor compreensão da tarefa de classificação, é importante descrever algumas definições básicas sobre o conjunto de dados. A maioria dos algoritmos de aprendizagem de máquina, utilizam como entrada uma representação de dados, os dados de entradas são conhecidos como:

- **Exemplos -** caso, dado ou registro;
- **Atributos -** característica de um exemplo;
- **Classes -** rótulo de um atributo especial que descreve o fenômeno de interesse;

A tabela 3.1 apresenta o formato geral de um conjunto de exemplos E, com N exemplos e M atributos.

Tabela 3.1: Exemplo [38]

	A_1	A_2	...	A_M	**Y**
E_1	x_{11}	x_{12}	$\dots$	x_{1M}	y_1
E_2	x_{12}	x_{22}	$\dots$	x_{2M}	y_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
E_N	x_{N1}	x_{N2}	$\dots$	x_{NM}	y_N

Onde as colunas (A_1,...,A_M) representam os diferentes atributos, as linhas (E_1,...,E_N) os diferentes exemplos. O atributo Y é o que assume os valores da classe de cada exemplo E_i. Os E_i são vetores na forma (x_{i1},...,x_{iM}) cujo componentes são valores discretos ou contínuos. Lembrando que o atributo valor (Y) também chamado de classe, somente está presente em conjuntos de dados para o aprendizado supervisionado [38].

A seguir as principais tarefas de mineração de dados tanto supervisionadas como não supervisionadas são descritas de forma sucinta:

- **Classificação -** Função de aprendizado que mapeia dados de entrada, em um número finito de classes, em que cada exemplo pertence a uma classe. O objetivo de um algoritmo de classificação é encontrar uma correlação entre os atributos e uma classe, de modo que o processo de classificação possa usá-la para predizer a classe de um exemplo novo e desconhecido [12]. Um exemplo para a tarefa de classificação é identificar a forma de tratamento mais adequado para um paciente, baseando-se em classes de pacientes que respondem bem ao tratamento médico.

- **Regressão -** A tarefa de regressão tem o conceito similar ao de classificação, a diferença consiste no registro que é identificado por um valor numérico e não um categórico. Desta forma, pode-se estimar o valor de um determinada variável analisando os valores das demais [11].

- **Regras de Associação -** Uma regra de associação caracteriza o quanto a presença de um conjunto de itens nos registros de uma base de dados implica na presença de outro conjunto distinto de itens no mesmo registro, por exemplo, observando os dados de vendas de um supermercado sabe-se que 80% dos clientes que compram um produto Q adquirem na mesma compra o produto W [12].

- **Agrupamento -** Agrupamento tem como objetivo descobrir suas próprias classes, isto é, agrupar os dados e descobrir subconjuntos mais homogêneos possíveis, encontrando descrições de cada um destes subconjuntos. Nesta técnica, os registros são agrupados de acordo com a semelhança [12].

3.3 Métricas de Avaliação

Na classificação, a utilização de alguma métrica para avaliar os resultados obtidos é necessária, e um exemplo para essa avaliação é a matriz de confusão, que tem como intuito mostrar o número de classificações corretas em relação às esperadas por cada classe. A matriz de confusão pode representar o problema de duas classes como também multiclasses, e o número de acertos da classe se encontra na diagonal principal da matriz como mostra a Tabela 3.2.

Tabela 3.2: Matriz de Confusão [32]

	Predição Positiva	**Predição Negativa**
Classe Positiva	Verdadeiro Positivo (True Positive, TP)	Falso Negativo (False Negative, FN)
Classe Negativa	Falso Positivo (False Positive, FP)	Verdadeiro Negativo (True Negative, TN)

Como pode-se perceber geralmente uma classe é denotada como positiva e outra como negativa.

- **TP -** TP ou Verdadeiros Positivos (True Positive), corresponde ao número de exemplos positivos classificados corretamente.
- **TN -** TN ou Verdadeiros Negativos (True Negative), corresponde ao número de exemplos negativos classificados corretamente.
- **FP -** FP ou Falsos Positivos (False Positive), corresponde ao número de exemplos falsos positivos, ou seja, a classe é negativa, mas foi classificada incorretamente como positiva.
- **FN -** FN ou Falsos Negativos (False Negative), corresponde ao número de exemplos falsos negativos, ou seja, a classe é positiva, mas foi classificada incorretamente como negativa. [16]

Além da matriz de confusão, outras métricas são frequentemente utilizadas a partir das informações dessa matriz, que são: a taxa de erro por classe e a acurácia, mas vale lembrar

que somente essas medidas não são apropriadas para problemas de classes desbalanceadas, pois não leva em conta a distribuição das mesmas.

- Taxa dos falsos negativos - são os falsos negativos divididos pelo total de números positivos [4].

$$erro = \frac{FN}{TP+FN} \tag{3.1}$$

- Taxa dos falsos positivos - são os falsos positivos divididos pelo total de números negativos [4].

$$erro = \frac{FP}{FP+TN} \tag{3.2}$$

- Acurácia - o cálculo é feito pela soma dos exemplos classificados corretamente e dividida pela soma de todos os valores dos elementos da matriz [4].

$$acuracia = \frac{TP+TN}{n} \tag{3.3}$$

Essas medidas podem inferir resultados que conseguem ser mal interpretados facilmente, ou até mesmo ser considerados como resultados enganosos. As métricas que são relevantes para os problemas de classes desbalanceadas são: Precisão, Recall e F-measures, que serão detalhadas a seguir, métricas essas que também utilizam as variáveis da matriz de confusão, e são apropriadas quando se está preocupado com o desempenho da classe positiva. Quando o desempenho é importante para ambas as classes o cálculo da área de ROC é aplicado.

3.3.1 Precisão

A precisão é uma medida de exatidão, e indica a porcentagem de exemplos positivos classificados corretamente entre todos aqueles preditos como positivos [4].

$$Precisao = \frac{TP}{TP+FP} \tag{3.4}$$

3.3.2 Recall

Recall também denominada sensitividade, é uma medida que corresponde à taxa de acerto dos exemplos positivos classificados corretamente entre todos os exemplos positivos da base [41].

$$Recall = \frac{TP}{TP+FN} \tag{3.5}$$

3.3.3 F-measures

A utilização do F-measures tem o objetivo de sintetizar as informações das duas últimas métricas, Precisão e Recall, obtendo então uma média entre elas [4].

$$F - measures = \frac{Precisao.Recall}{Precisao+Recall}.2 \tag{3.6}$$

O resultado F-measures é um indicativo de que, quanto mais próximo de 1, melhores são os resultados, caso contrário, os resultados são ruins.

3.3.4 Curva ROC

A curva ROC consiste em um gráfico de duas dimensões, onde o eixo y refere-se a Precisão ou Recall e o eixo x à especificidade como mostra a formula a baixo [4].

$$especificidade = \frac{TN}{FP+TN} \tag{3.7}$$

O desempenho de um classificador pode ser plotado em um espaço denominado espaço ROC.

Cada ponto na curva corresponde a um dos modelos induzidos pelo classificador. Um classificador é denominado melhor que outro, se seu ponto no espaço ROC está localizado acima e à esquerda do ponto correspondente ao segundo classificador, ou seja, classificadores acima da diagonal são os modelos que fazem precisões melhores do que estão abaixo ou próximo a esta diagonal [41].

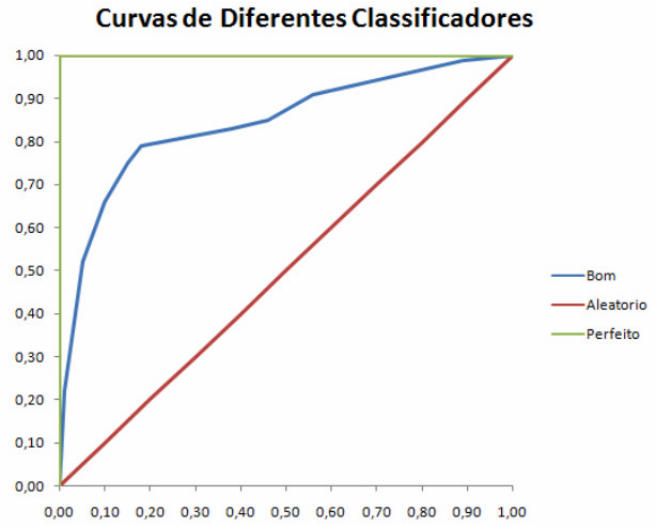

Figura 3.2: Curva ROC [41]

3.3.5 Área de ROC - AUC

A métrica AUC (do inglês, *Area Under Curve*), tem o objetivo de sintetizar a informação representada pela curva ROC, sendo definida como a equação 3.8.

$$AUC = \Phi\left(\frac{\delta}{\sqrt{\phi_{pos}+\phi_{neg}}}\right) \tag{3.8}$$

Onde Φ é a distribuição cumulativa normal padrão, δ é a distância euclidiana entre os centroides das duas classes, e ϕ_{pos} e ϕ_{neg}, são o desvio padrão das classes positiva e negativa respectivamente [4].

CAPÍTULO 4

CLASSIFICAÇÃO EM CONJUNTOS DE DADOS DESBALANCEADOS

As informações do mundo real são distribuídas de maneira desigual, principalmente em situações nas quais determinadas classes são mais difíceis de se obter, causando assim o problema de classes desbalanceadas.

Muitos aspectos podem influenciar o desempenho do modelo criado por um sistema supervisionado. Um desses aspectos está relacionado com a distribuição entre o número de exemplos pertencentes a cada uma das classes, ou seja, uma base de dados é dita desbalanceada, quando o número de exemplos de uma classe é maior que o número de exemplos da outra classe [34]. Tal problema pode prejudicar o desempenho de classificação, pois tendem a classificar exemplos da classe minoritária como sendo da classe majoritária [4].

Um exemplo do problema que os dados desbalanceados podem causar no mundo real, é o diagnóstico de câncer, em que a proporção de pacientes doentes é menor do que os saudáveis, contendo, por exemplo, 260 para a classe positiva (classe minoritária) e 10.923 para a classe negativa (classe majoritária). Neste caso é preciso que um classificador forneça um grau de equilíbrio na precisão, tanto para as classes minoritárias como as majoritárias. Mas se o classificador atingir apenas 10% de precisão para a classe minoritária por exemplo, significa que, 234 amostras dessa classe foram classificadas incorretamente como amostras da classe majoritária, obtendo como consequência 234 pacientes com a doença que não receberão o tratamento a tempo. Na indústria médica, as ramificações desta consequência podem ser extremamente caras [22].

Para tratar o problema de classes desbalanceadas é preciso procurar por uma distribuição equilibrada da classe, para que o desempenho do algoritmo de aprendizagem seja aceitável para a classe minoritária [3].

Neste Capítulo serão mostrados os problemas de classificação em conjuntos de dados

desbalanceados e também os métodos para resolver estes problemas.

4.1 Problemas de Classificação em Conjunto de dados Desbalanceados

De acordo com Weiss [40], nem sempre a raridade de exemplos positivos é a causa principal do baixo desempenho em conjuntos de dados desbalanceados. O autor mostra ainda, algumas causas associadas ao problema de classificação em conjuntos de dados desbalanceados.

4.1.1 Métricas de Avaliação

A métrica de avaliação mais utilizada para calcular o número de exemplos que são classificados corretamente é a acurácia, esta é a mais utilizada para a tarefa de classificação. Mas existe uma falha nesta medida em relação a classes raras, pois tem menos impacto na acurácia que as classes comuns. Por exemplo, um problema de duas classes, que a distribuição das classes fica com 99 exemplos para a classe de negativos e 1 exemplo para a classe positivo, o valor da acurácia é de 99% de acerto na classe negativa, esse valor não faz sentido se analisado em relação à classe positiva [40]. As métricas de avaliação apropriadas para as classes desbalanceadas foram detalhadas na Seção 3.3.

4.1.2 Raridade Absoluta

Com poucos exemplos representando a classe, gera uma dificuldade para o algoritmo generalizar, pois com poucos dados a superfície de decisão ficará muito diferente do que existe no mundo real.

4.1.3 Raridade Relativa

Embora exista muitos exemplos positivos, os mesmos são ofuscados pela grande quantidade de exemplos negativos. Uma frase que ilustra isso é “Como encontrar agulha no

Palheiro". Esta frase é relevante, porque é o grande número de fios de feno no palheiro que torna difícil encontrar a agulha.

4.1.4 Identificação de Ruídos

Várias podem ser as causas da presença de ruídos. Para facilitar a compreensão, será considerado o conjunto de dados da Figura 4.1(a).

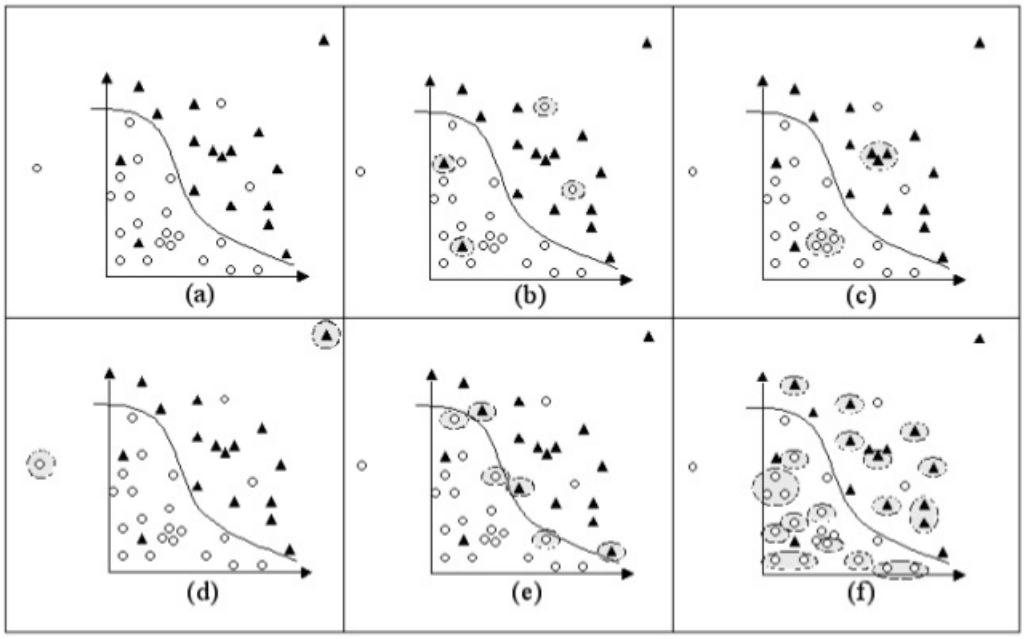

Figura 4.1: Conjunto de Dados [30].

Nesse conjunto pode-se apresentar 5 categorias como:

- **Dados com rótulo incorreto:** são dados classificados incorretamente, esses são os verdadeiros ruídos, ou seja, são exemplos que estão do lado errado da borda de decisão. Este exemplo é mostrado na Figura 4.1(b).

- **Dados redundantes:** são dados que possuem grande semelhança com outro objeto do mesmo conjunto de dados, ou seja, os valores dos atributos são muito semelhantes. Em caso extremo os atributos possuem os mesmos valores [16]. Se a redundância não for eliminada, o algoritmo de aprendizagem de máquina pode atribuir aos exemplos repetidos uma importância maior que os demais exemplos. Alguns exemplos são

apresentados na Figura 4.1(c). A redução desses dados pode ser vantajosa no balanceamento do número de exemplos pertencentes a diferentes classes em conjuntos de dados desbalanceados [30].

- **Outliers:** exemplos muito distintos dos demais, podem ser um indicador da presença de ruídos, pois são valores que estão além dos limites aceitáveis. A influência desses na indução da fronteira de decisão deve ser minimizada. Exemplo especificado na Figura 4.1(d).

- **Dados próximos da borda de decisão:** esses dados são considerados inseguros, pois podem mover a fronteira de decisão para o lado errado. A sua remoção pode suavizar a borda de decisão, evitando o overfitting, ou seja, generalização excessiva. Exemplos na Figura 4.1(e).

- **Dados seguros:** A Figura 4.1(f) apresenta os dados que devem ser mantidos no treinamento, pois são os dados mais típicos de cada classe.

O objetivo da identificação dos ruídos, é poder detectar e eliminar os ruídos presentes nos conjuntos de dados, procurando manter os dados seguros.

4.2 Métodos para Resolver problemas de Classificação em Conjuntos de dados Desbalanceados

Existem métodos que consistem em efetuar ajustes no conjunto de dados uniformizando uma distribuição de exemplos entre as classes utilizando amostragem (em inglês, *sampling*). Dentre os métodos existentes na literatura, seja para adicionar ou remover instâncias das classes, os mais conhecidos são: Oversampling (*Sobreamostragem*) aleatória, e Undersampling (*Subamostragem*) aleatória, métodos que fazem parte do pré-processamento de dados fazendo com que o algoritmo de classificação trabalhe com o conjunto de dados balanceados [6].

- **Undersampling Aleatória-** tem como objetivo balancear o conjunto de dados pela eliminação de exemplos da classe majoritária [3].

- **Oversampling Aleatória-** replica exemplos da classe minoritária com o objetivo de obter uma distribuição mais balanceada [3].

São métodos equivalentes no que diz respeito ao balanceamento artificial das classes, mas com um conjunto próprio de problemas consequentes que podem atrapalhar o aprendizado. O problema causado pelo *undersampling* acontece na eliminação de exemplos da classe majoritária fazendo com que o classificador perca informações importantes pertencentes a essa classe. No caso do *oversampling* aumenta a probabilidade de ocorrer *overfitting*, ou seja, prejudica o poder de generalização para a classe de interesse já que o método faz cópias exatas dos exemplos da classe minoritária [19].

4.2.1 Oversampling Orientado

Nesta seção são apresentadas técnicas que utilizam alguma heurística para balancear a base adicionando exemplos na classe minoritária na fase de pré-processamento.

Uma das técnicas mais conhecidas que tem se mostrado como um poderoso método em várias aplicações é o SMOTE (*Synthetic Minority Oversampling Technique*). Este método cria dados artificiais com base nas semelhanças existentes no espaço de característica entre os exemplos da classe minoritária. O algoritmo funciona criando exemplos sintéticos baseados em exemplos minoritários e seus k vizinhos mais próximos. Os exemplos sintéticos são criados com base nos exemplos x_i e $\hat{x}_i$, sendo x_i instância da classe minoritária, e $\hat{x}_i$ selecionada pelos k vizinhos de x_i e δ um número aleatório entre 0 e 1 [9].

$$x_{new} = x_i + (\hat{x}_i - x_i)\delta \tag{4.1}$$

Esse método pode aumentar a taxa de acurácia, mas ocorre o efeito indesejado de modelos muito específicos para estes casos replicados, prejudicando a generalização para a classe de interesse, fato conhecido como overfitting. Para resolver esse problema vários métodos adaptativos como Borderline-SMOTE e ADASYN (Adaptative Synthetic Sampling) têm sido propostos [22], para que esses novos dados sejam gerados na vizinhança de cada caso da classe minoritária, de forma a fazer crescer a região de decisão e, assim,

aumentar o poder de generalização dos classificadores [31].

- **Borderline-SMOTE** - O algoritmo Borderline-SMOTE tem o objetivo de identificar as amostras próximas à superfície de decisão, essas amostras são identificadas como conjunto DANGER. É preciso determinar o conjunto de vizinhos mais próximos para cada x_i e depois identificar o número de vizinhos mais próximos que pertencem a classe majoritária, formando assim o conjunto DANGER, ou seja, instâncias que pertencem a estes conjuntos são aquelas que, possuem mais vizinhos da classe majoritária do que da classe minoritária. Ao contrário do SMOTE que gera instâncias sintéticas a partir de todos os exemplos do conjunto P de instâncias minoritárias, o Borderline-SMOTE irá aplicar o SMOTE somente às instâncias do conjunto DANGER [4][22]. A Figura 4.2 mostra a ocorrência de um ruído e nenhum exemplo sintético gerado por ele, como mostra a instância C [22].

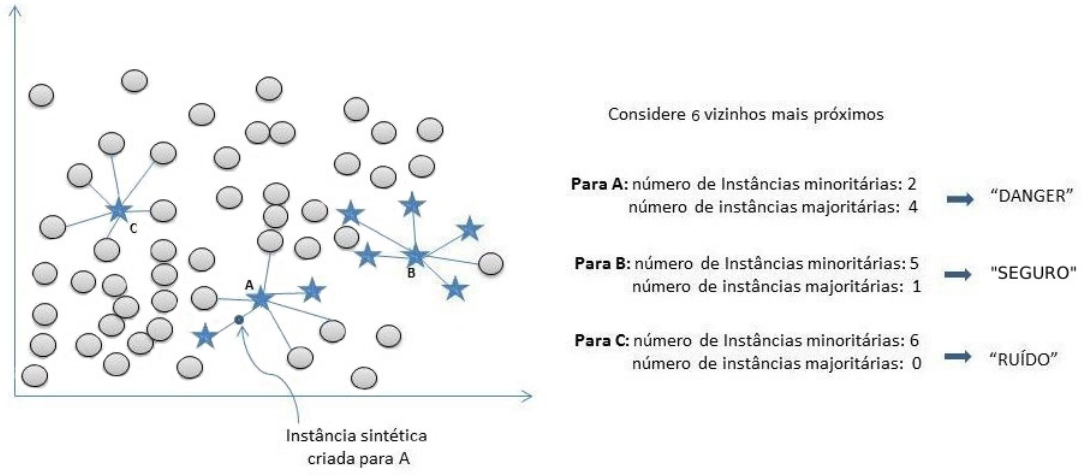

Figura 4.2: Geração de dados baseados no método Borderline [22]

- **ADASYN** - O algoritmo ADASYN é baseado no algoritmo SMOTE, o objetivo é reduzir o viés introduzido pela distribuição desbalanceada, e principalmente deslocar adaptativamente a fronteira de decisão para focar nos exemplos mais difíceis de aprender. Primeiramente é obtido o grau de desbalanceamento da classe pela Equação 4.2.

$$d = \frac{m_p}{m_n} \tag{4.2}$$

Onde m_p é o número de exemplos da classe minoritária e m_n é o número de exemplos da classe majoritária, e $d \in (0, 1)$. Se $d < d_{th}$ então o cálculo do número de exemplos sintéticos que precisam ser gerados, é feito pela Equação 4.3.

$$G = (m_n - m_p).\beta \tag{4.3}$$

Onde $\beta \in (0, 1)$ e é um parâmetro usado para especificar o nível de balanceamento desejado depois da geração dos dados sintéticos. Para cada exemplo x_i no conjunto de instâncias positivas P, sendo i = 1,...,P, são encontrados os K vizinhos mais próximos baseados na distância euclidiana no espaço dimensional n, e então calcular a proporção de D_i (Equação 4.4).

$$D_i = \frac{\Delta_i}{k} \tag{4.4}$$

Onde Δ_i é o número de exemplos dos k vizinhos mais próximos de x_i que pertence a classe majoritária, portanto $D_i \in (0, 1)$. Uma normalização é feita de D_i para $\hat{D}_i = D_i / \sum_{i=1}^{mp} D_i$, de modo que $\hat{D}_i$ é a distribuição de densidade ($\sum_i \hat{D}_i = 1$).

Então o número de exemplos sintéticos que precisam ser gerados para cada exemplo minoritário x_i é dado pela Equação 4.5.

$$g_i = \hat{D}_i \times G \tag{4.5}$$

Onde G é o total de números de exemplos minoritários que precisam ser gerados para a classe minoritária definida na Equação 4.3. A ideia principal do ADASYN é usar a distribuição de densidade $\hat{D}_i$ como um critério para decidir automaticamente o número de exemplos sintéticos que precisam ser gerados para cada exemplo da classe minoritária. De acordo com esses cálculos é possível perceber que quanto maior o número de instâncias da classe majoritária, mais se está próximo a superfície de decisão, e mais instâncias sintéticas serão criadas.

- **Cluster baseado em Oversampling** - Algoritmos de Cluster baseados em oversampling têm como objetivo minimizar o problema de pequenos disjuntos que são conceitos aprendidos que cobrem poucos casos do conjunto de treinamento, podendo ser aplicável a conjuntos de dados multiclasses. Utiliza técnica de agrupamento k-means, ou seja, leva um conjunto aleatório K exemplos de cada cluster e calcula a média do vetor de características desses exemplos, chamados como centros do cluster. Em seguida, o restante dos exemplos de treinamento é apresentado um de cada vez, e para cada exemplo, é feito o cálculo da distância euclidiana entre ele e cada centro do cluster. Então cada exemplo de treinamento é atribuído ao cluster que apresenta a menor magnitude do vetor de distância. Todos os meios de cluster são atualizados e o processo é repetido até que todos os exemplos sejam esgotados [22], [31].

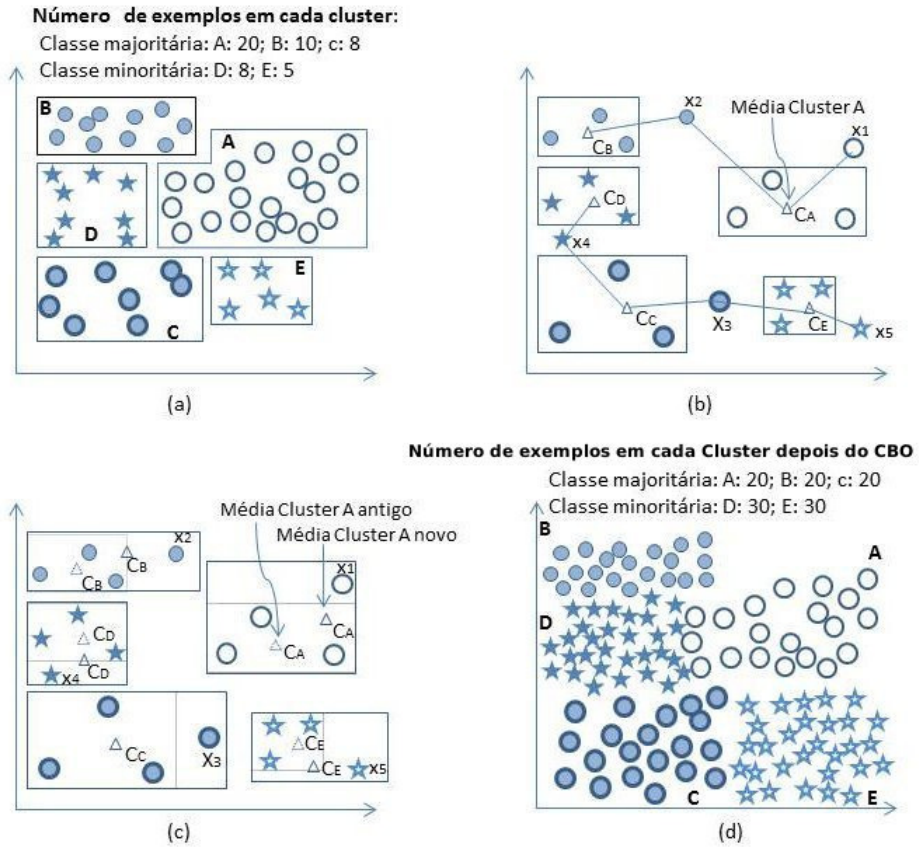

Figura 4.3: Etapas do Algoritmo cluster baseado em oversampling [22]

A Figura 4.3 ilustra as etapas do algoritmo, a Figura 4.3a mostra a distribuição do

conjunto de dados original, nesta etapa a classe majoritária tem 3 clusters (A, B, C) contendo cada um 20, 10, e 8 exemplos respectivamente. A classe minoritária tem 2 clusters, D com 8 exemplos e E com 5. A Figura 4.3b apresenta a média do cluster para 3 exemplos aleatórios de cada cluster (ou seja, k=3), que estão representados pelos triângulos, e mostra também o vetor de distância para os 5 exemplos introduzidos individualmente x_1, x_2, x_3, x_4 e x_5. a Figura 4.3c são as médias de cluster atualizadas e as fronteiras dos clusters como resultado dos 5 exemplos introduzidos. Uma vez que todos os exemplos tiverem sido esgotados, o algoritmo CBO preenche todos os outros clusters da classe minoritária até que fique do mesmo tamanho do cluster maior A (ou seja, o cluster B e C terão 20 exemplos cada). É denotado que o número de exemplos da classe majoritária depois do procedimento oversampling é igual a 60. Então o procedimento é aplicado nos clusters minoritários até que cada cluster minoritário D e E tenha o número total de $60/2 = 30$ exemplos. A Figura 4.3d mostra o conjunto de dados depois de aplicado o método de cluster baseado em oversampling [22].

4.2.2 Undersampling Orientado

Nesta seção são apresentadas técnicas que utilizam alguma heurística para remover exemplos da classe majoritária. Essas técnicas são conhecidas também como métodos híbridos, pois pertencem a um grupo de métodos que juntam os algoritmos de pré-processamento com classificação.

Xu-Ying Liu [29] apresenta dois exemplos que têm o objetivo de reduzir a perda de informações causadas pelo método tradicional de undersampling aleatória. Os algoritmos apresentados são: EasyEnsemble e BalanceCascade.

- EasyEnsemble - Aprendizagem considerada não supervisionada, pois explora o conjunto N (instâncias da classe majoritária) utilizando uma amostragem aleatória com substituição. A implementação do EasyEnsemble é considerada simples, pois dado um conjunto de treinamento minoritário P, e o conjunto de treinamento majoritário N, criam-se t subconjuntos $N_i', i = 1, ..., i = t$, compostos de exemplos retirados de

N, onde $N_i' < |N|$, e utiliza-se comumente $|N_i'| = |P|$. Os subconjuntos $N_i', ..., N_t'$ são criados e treinados separadamente, onde um classificador H_i é treinado usando N_i' e todos os exemplos de P (O classificador H_i utilizado foi o *Adaboost*). Todos os classificadores gerados são combinados para uma decisão final [4] [29].

- BalanceCascade - Considerada uma abordagem supervisionada por explorar exemplos da classe majoritária N de um conjunto de treinamento. A ideia é depois que o H_1 é treinado, se um exemplo $x_1 \in N$ for corretamente classificado por H_1, é aceitável pressupor que x_1 é um pouco redundante em N, dado que H_1 já aprendeu a classificar esse exemplo. Desta forma pode-se remover alguns exemplos classificados corretamente do conjunto N. Este método é chamado BalanceCascade por ser similar ao classificador em cascata, diferenciando apenas no final da classificação, pois o classificador cascata $H(x)$ prediz positivo se e somente se cada $H_i(x)(i = 1, 2, ..., T)$ prediz positivo. Enquanto no BalanceCascade, a dependência sequencial entre classificadores é principalmente explorada para reduzir a informação redundante na classe principal. Assim a estratégia de amostragem leva para um espaço de amostra restrita para o seguinte processo undersampling, na esperança de explorar o máximo de informação útil possível.

4.2.3 Integração de Amostragem e Boosting

Chawla [9] propôs um método que faz uma combinação da técnica SMOTE com o procedimento de Boosting. O SMOTE é utilizado para melhorar a predição das classes minoritárias, e o Boosting para não sacrificar a precisão do conjunto de dados, ou seja, é um método iterativo de combinação de classificadores, onde a cada iteração um classificador é induzido a partir de uma amostra de instâncias do conjunto de treinamento [36].

A ideia dessa junção de técnicas é melhorar o modelo da classe minoritária no conjunto de dados, e fornecer não só instâncias classificadas incorretamente da classe minoritária, mas também uma representação mais ampla dessas instâncias [9].

Outra abordagem integrada é o DataBoost-IM, que combina a técnica de geração de dados introduzidas com AdaBoost.M1 para atingir alta predição de previsão para a classe minoritária sem sacrificar a precisão da classe majoritária [4][22].

CAPÍTULO 5

ESTUDOS EMPÍRICOS

Neste capítulo descreve-se um estudo empírico realizado com diferentes bases de dados e métodos de balanceamento com o objetivo de entender mais profundamente os efeitos de cada um dos métodos. Para a realização dos experimentos foram utilizadas algumas ferramentas que estão descritas na Seção 5.1. Os algoritmos utilizados para o processo de classificação estão relacionados na Seção 5.1.1.

5.1 Ferramentas

Os algoritmos aqui utilizados são provenientes do pacote de software Weka (*Waikato Environment for Knowledge Analysis*) que foi desenvolvido pela Universidade de Waikato na Nova Zelândia, uma das principais características é pelo fato de ser uma ferramenta desenvolvida em Java. Compreende uma coleção de algoritmos de aprendizagem para tarefas de mineração de dados, como o pré-processamento, classificação, regressão, regras de associações e visualizações de dados, que possibilitam a obtenção de resultados estatísticos [42]. Também foi utilizado o Software Keel (*Knowledge Extraction based on Evolutionary Learning*) [1], que igualmente ao WEKA é uma ferramenta de código aberto e implementada pela linguagem Java, foi desenvolvida pelos projetos nacionais Espanhóis, tem objetivo de proporcionar diversas técnicas para avaliar o comportamento de aprendizagem evolutiva, diferentes tipos de algoritmos evolucionários e de problemas de mineração de dados. O software Keel foi utilizado para aplicar os métodos de balanceamento.

5.1.1 Classificadores

Os classificadores são modelos baseados em um conjunto de dados de treinamento, e esse modelo é usado para classificar outras instâncias em um conjunto de testes. A utilização é feita quando se quer conhecer qual classe de valores o algoritmo de aprendizagem prediz

para cada registro. Neste trabalho foram utilizados cinco algoritmos que são implementados no Software Weka (Waikato Environment for Knowledge Analysis) que são detalhados a seguir:

- **Naive Bayes** é um algoritmo de classificação ingênuo probabilístico, baseado na aplicação do teorema de Bayes, foi projetado para uso em tarefas de indução supervisionada, e tem como objetivo determinar a classe de maior probabilidade para cada nova instância a ser classificada [28]. Para classificar uma nova instância, o algoritmo determina a classe mais provável.

- **Bayes Net** constrói uma rede Bayesiana sobre um conjunto de dados, gerando um grafo e um conjunto de tabelas de probabilidade, esta rede Bayesiana representa a distribuição de probabilidades dos dados. A classificação é feita encontrando uma rede Bayesiana apropriada a um conjunto de dados. Todos os algoritmos de redes Bayesianas implementados no WEKA assumem para os conjuntos de dados: (a) todas as variáveis são discretas e finitas, e (b) não existem instâncias com valores ausentes. Caso estas hipóteses não sejam satisfeitas, a base é automaticamente filtrada e um aviso é mostrado [8].

- **SMO** (*Sequential Minimal Optimization*) é uma extensão sobre a decomposição de problemas de otimização quadrática, em que este, é quebrado em problemas menores com apenas dois multiplicadores de Lagrange, a condição principal para o SMO é $\sum_{i=1}^{l} \alpha_i y_i = 0$. Ou seja, essa condição obriga que quando um Multiplicador é atualizado o outro deve ser ajustado para manter a condição verdadeira. A escolha dos dois pontos é feita a partir de uma heurística, já a atualização dos valores é feita de forma analítica [2].

- **J48** - É baseado nos algoritmos de árvores de decisão, que tem o objetivo de formar a árvore mais adequada sobre o conjunto de dados. Uma árvore de decisão utiliza uma abordagem dividir para conquistar, pois os dados a serem analisados possuem atributos diferentes, não possuindo uma forma lógica para representação. Desta forma os resultados são representados na forma de árvore [16].

- **JRip** - O algoritmo RIPPER (Repeated Incremental Pruning to Produce Error Reduction), chamado de JRip por ser implementado na linguagem Java para o software WEKA. Ele constrói um conjunto de regras que modelam um conjunto de dados. O algoritmo RIPPER gera o primeiro modelo adicionando uma regra por vez. O passo seguinte consiste em remover todos os exemplos classificados pela regra do conjunto base de geração das regras, processo chamado de pruning. O processo é repetido até que não existam mais exemplos a serem classificados positivamente, ou até que uma regra gerada possua uma taxa de erro alta [7].

5.2 Base UCI

Nos primeiros experimentos deste trabalho, foram utilizadas bases do repositório UCI *Machine Learning Repository*. O repositório UCI, abrange bases relacionadas a diversos problemas, e estão disponíveis para todos que precisam trabalhar com aprendizagem de máquina. As bases utilizadas nos experimentos e suas principais características estão relacionadas na Tabela 5.1.

Tabela 5.1: Caracteristicas da Base UCI

Base	**Instâncias**	**Atributos**	**Classe**		**%**
			Positivo	**Negativo**	
Lettera	19999	17	789	19210	4%
Glass	214	10	17	197	9%
Flag	194	29	17	177	10%
Satimage	6435	37	626	5809	11%
Ecoli	336	8	35	301	12%
New-thyroid	215	6	35	180	19%
Vehicle	846	19	199	647	31%
Haberman	306	4	81	225	36%
Breast	683	10	239	444	54%
Ionosphere	351	34	126	225	56%
Bupa	345	7	145	200	73%
Heart	270	14	120	150	80%
Pima	768	9	500	268	54%

A tabela está ordenada pela base que possui maior nível de desbalanceamento.

5.2.1 Resultados - Bases UCI

Primeiramente, foram aplicados os classificadores tradicionais nas bases originais com o objetivo de poder observar o comportamento destes, com as bases que possuem um alto nível de desbalanceamento. As métricas de avaliação utilizadas foram RecallP, RecallN e Acurácia, pois dessa forma é possível perceber o método de balanceamento que realmente consegue evitar os problemas indesejados como os ruídos, pois quando acontece a melhora dos valores da classe minoritária e os valores da classe majoritária são mantidos, consequentemente acontece a melhora da acurácia.

A Tabela 5.2 apresenta os resultados das bases retiradas do repositório UCI, classificadas com os algoritmos tradicionais. É possível perceber que os classificadores nem sempre conseguem predizer corretamente algumas bases, principalmente para a classe positiva (RecallP), pois provavelmente estas bases possuem um número menor de exemplos que determinam essa classe, ou também, podem possuir os problemas como ruídos, dados redundantes ou outliers, dificultando assim o poder de generalização do classificador.

Com o intuito de melhorar os resultados obtidos anteriormente, foram aplicados os métodos de balanceamento (Adasyn, Borderline, Smote, Oversampling e Undersampling). Os resultados para esse experimento são apresentados na Tabela 5.3.

Tabela 5.2: Classificação com a base original

Base	Class. Tradicionais	RecallP	RecallN	AUC	Acurácia
breast	NaiveBayes	0,975	0,957	0,985	96,340
	SMO	0,962	0,973	0,968	96,924
	MultilayerP	0,941	0,969	0,989	95,900
	J48	0,937	0,953	0,954	94,729
	JRip	0,958	0,953	0,964	95,460
	BayesNet	0,979	0,971	0,993	97,364
bupa	NaiveBayes	0,772	0,395	0,650	55,362
	SMO	0,007	0,995	0,501	57,971
	MultilayerP	0,634	0,770	0,745	71,304
	J48	0,634	0,690	0,673	66,667
	JRip	0,490	0,785	0,643	66,087
	BayesNet	0,172	0,870	0,521	57,681
Ecoli	NaiveBayes	0,800	0,894	0,928	88,402
	SMO	0,029	1,000	0,514	89,877
	MultilayerP	0,714	0,963	0,931	93,766
	J48	0,486	0,973	0,772	92,270
	JRip	0,571	0,960	0,771	91,971
	BayesNet	0,886	0,860	0,918	86,308
Flag	NaiveBayes	0,500	0,808	0,637	77,908
	SMO	0,067	0,960	0,513	88,138
	MultilayerP	0,117	0,949	0,574	87,625
	J48	0,000	1,000	0,500	91,242
	JRip	0,100	0,960	0,530	88,650
	BayesNet	0,300	0,904	0,731	85,088
Glass	NaiveBayes	0,783	0,457	0,758	48,084
	SMO	0,000	1,000	0,500	92,060
	MultilayerP	0,183	0,970	0,830	90,642
	J48	0,283	0,964	0,727	91,118
	JRip	0,067	0,980	0,523	90,664
	BayesNet	0,000	1,000	0,528	92,060
haberman	NaiveBayes	0,199	0,942	0,662	74,521
	SMO	0,000	1,000	0,500	73,533
	MultilayerP	0,286	0,902	0,685	73,882
	J48	0,470	0,809	0,624	71,893
	JRip	0,345	0,867	0,606	72,871
	BayesNet	0,257	0,893	0,644	72,544
heart	NaiveBayes	0,792	0,853	0,910	82,593
	SMO	0,775	0,907	0,841	84,815
	MultilayerP	0,733	0,820	0,859	78,148
	J48	0,717	0,780	0,771	75,185
	JRip	0,708	0,833	0,795	77,778
	BayesNet	0,808	0,860	0,903	83,704
ionosphere	NaiveBayes	0,865	0,809	0,937	82,893
	SMO	0,714	0,960	0,837	87,167
	MultilayerP	0,810	0,964	0,937	90,881
	J48	0,818	0,911	0,853	87,738
	JRip	0,826	0,929	0,891	89,175
	BayesNet	0,810	0,929	0,949	88,600
Lettera	NaiveBayes	0,858	0,992	0,957	98,650
	SMO	0,845	0,998	0,921	99,160
	MultilayerP	0,888	1,000	0,960	99,540
	J48	0,958	0,999	0,991	99,715
	JRip	0,949	0,998	0,974	99,635
	BayesNet	0,848	0,992	0,971	98,605
New-thyroid	NaiveBayes	1,000	0,972	1,000	97,674
	SMO	0,543	1,000	0,771	92,558
	MultilayerP	0,943	0,989	0,999	98,140
	J48	0,829	0,989	0,894	96,279
	JRip	0,857	0,978	0,917	95,814
	BayesNet	0,886	0,994	0,998	97,674
pima	NaiveBayes	0,848	0,608	0,812	76,433
	SMO	0,896	0,534	0,715	76,956
	MultilayerP	0,838	0,563	0,795	74,219
	J48	0,816	0,567	0,723	72,916
	JRip	0,836	0,605	0,730	75,525
	BayesNet	0,828	0,630	0,815	75,910
Satimage	NaiveBayes	0,866	0,815	0,920	82,020
	SMO	0,000	1,000	0,500	90,272
	MultilayerP	0,660	0,970	0,938	94,017
	J48	0,545	0,957	0,756	91,671
	JRip	0,514	0,968	0,743	92,432
	BayesNet	0,835	0,863	0,925	86,030
Vehicle	NaiveBayes	0,889	0,592	0,815	66,192
	SMO	0,909	0,975	0,942	95,981
	MultilayerP	0,939	0,983	0,995	97,279
	J48	0,864	0,957	0,920	93,503
	JRip	0,859	0,944	0,910	92,436
	BayesNet	0,945	0,683	0,906	74,467

Tabela 5.3: Métodos de Pré-processamento

Base	Métodos	NaiveBayes			SMO			MultilayerP			J48			JRip			BayesNet		
		RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia
breast	ADASYN	0,987	0,944	95,901	0,996	0,955	96,926	0,979	0,946	95,754	0,950	0,939	94,292	0,971	0,948	95,606	0,992	0,968	97,656
	Boderline	0,987	0,948	96,193	0,992	0,962	97,219	0,975	0,959	96,487	0,941	0,959	95,317	0,975	0,957	96,339	0,983	0,966	97,217
	Smote	0,983	0,955	96,487	0,983	0,971	97,510	0,967	0,957	96,048	0,950	0,948	94,873	0,966	0,953	95,754	0,979	0,968	97,217
	Oversampling	0,979	0,957	96,487	0,979	0,966	97,072	0,958	0,962	96,048	0,950	0,953	95,165	0,954	0,966	96,193	0,992	0,971	97,804
	Undersampling	0,979	0,946	95,756	0,975	0,971	97,217	0,992	0,960	97,073	0,966	0,948	95,460	0,967	0,944	95,172	0,983	0,971	97,511
Bupa	ADASYN	0,834	0,275	51,014	0,745	0,525	61,739	0,786	0,570	66,087	0,621	0,610	61,449	0,710	0,610	65,217	0,883	0,125	44,348
	Boderline	0,807	0,300	51,304	0,717	0,490	58,551	0,724	0,665	68,986	0,517	0,685	61,449	0,648	0,650	64,928	0,669	0,430	53,043
	Smote	0,828	0,295	51,884	0,745	0,480	59,130	0,703	0,640	66,667	0,614	0,645	63,188	0,634	0,610	62,029	0,717	0,330	49,275
	Oversampling	0,821	0,260	49,565	0,724	0,495	59,130	0,703	0,660	67,826	0,607	0,670	64,348	0,641	0,650	64,638	0,614	0,515	55,652
	Undersampling	0,834	0,230	48,406	0,786	0,430	57,971	0,703	0,565	62,319	0,545	0,680	62,319	0,669	0,605	63,188	0,883	0,125	44,348
Ecoli	ADASYN	0,943	0,840	85,114	0,857	0,807	81,247	0,800	0,897	88,696	0,800	0,874	86,611	0,829	0,870	86,611	0,914	0,844	85,119
	Boderline	0,886	0,867	86,901	0,886	0,834	83,924	0,743	0,937	91,677	0,714	0,927	90,478	0,600	0,934	89,890	0,914	0,857	86,308
	Smote	0,943	0,844	85,404	0,886	0,824	83,029	0,886	0,900	89,881	0,743	0,914	89,587	0,743	0,910	89,289	0,914	0,851	85,720
	Oversampling	0,914	0,844	85,105	0,857	0,830	83,332	0,800	0,937	92,270	0,657	0,937	90,777	0,629	0,907	87,801	0,800	0,864	85,711
	Undersampling	0,971	0,840	85,404	0,971	0,787	80,632	0,800	0,837	83,332	0,800	0,784	78,595	0,914	0,837	84,522	0,914	0,807	81,831
Flag	ADASYN	0,617	0,514	52,105	0,400	0,740	71,134	0,283	0,791	74,750	0,217	0,769	72,186	0,250	0,893	83,522	0,367	0,791	75,263
	Boderline	0,433	0,825	78,934	0,267	0,893	83,522	0,183	0,927	86,100	0,200	0,905	84,049	0,067	0,943	86,599	0,067	0,994	91,242
	Smote	0,550	0,429	43,900	0,417	0,752	72,173	0,533	0,769	74,764	0,617	0,875	85,020	0,317	0,870	81,957	0,367	0,849	80,459
	Oversampling	0,550	0,746	72,753	0,050	0,893	81,943	0,133	0,870	80,391	0,283	0,898	84,521	0,167	0,909	84,521	0,350	0,893	84,575
	Undersampling	0,550	0,493	49,649	0,700	0,696	69,663	0,700	0,662	66,545	0,683	0,566	57,800	0,550	0,482	48,596	0,650	0,700	69,568
Glass	ADASYN	0,850	0,431	46,213	1,000	0,314	36,866	0,633	0,822	80,864	0,583	0,888	86,390	0,650	0,797	78,505	0,533	0,777	75,703
	Boderline	0,800	0,523	54,142	1,000	0,319	37,298	0,433	0,924	88,317	0,433	0,954	91,096	0,300	0,934	88,283	0,433	0,888	85,039
	Smote	0,783	0,431	45,759	0,867	0,380	42,027	0,533	0,878	85,028	0,650	0,832	81,761	0,550	0,878	84,994	0,417	0,832	79,867
	Oversampling	0,733	0,461	48,084	0,883	0,390	42,924	0,483	0,939	90,188	0,417	0,964	92,049	0,233	0,939	88,328	0,233	0,934	87,829
	Undersampling	0,833	0,406	43,865	1,000	0,263	32,193	0,883	0,461	49,446	0,800	0,542	56,489	0,800	0,415	44,762	0,933	0,411	45,293
Haberman	ADASYN	0,371	0,898	75,833	0,368	0,884	74,870	0,297	0,884	72,898	0,629	0,693	67,652	0,667	0,667	66,653	0,704	0,622	64,363
	Boderline	0,334	0,898	74,849	0,346	0,902	75,505	0,471	0,804	71,581	0,580	0,778	72,559	0,582	0,738	69,614	0,556	0,800	73,538
	Smote	0,358	0,893	75,177	0,346	0,902	75,510	0,349	0,858	72,242	0,555	0,733	68,646	0,642	0,693	67,980	0,506	0,764	69,603
	Oversampling	0,346	0,898	75,177	0,382	0,889	75,516	0,455	0,836	73,548	0,531	0,680	64,056	0,568	0,684	65,341	0,457	0,809	71,565
	Undersampling	0,396	0,858	73,543	0,454	0,747	67,023	0,692	0,627	64,379	0,485	0,782	70,270	0,592	0,747	70,608	0,663	0,587	60,894
Heart	ADASYN	0,842	0,847	84,444	0,833	0,827	82,963	0,750	0,787	77,037	0,758	0,760	75,926	0,700	0,773	74,074	0,825	0,813	81,852
	Boderline	0,808	0,867	84,074	0,842	0,860	85,185	0,733	0,760	74,815	0,767	0,787	77,778	0,717	0,813	77,037	0,808	0,840	82,593
	Smote	0,808	0,840	82,593	0,783	0,853	82,222	0,800	0,793	79,630	0,783	0,780	78,148	0,642	0,813	73,704	0,825	0,827	82,593
	Oversampling	0,792	0,853	82,593	0,817	0,847	83,333	0,650	0,787	72,593	0,667	0,753	71,481	0,742	0,780	76,296	0,817	0,827	82,222
	Undersampling	0,817	0,853	83,704	0,842	0,853	84,815	0,767	0,793	78,148	0,767	0,787	77,778	0,742	0,787	76,667	0,825	0,840	83,333
Ionosphere	ADASYN	0,873	0,844	85,453	0,738	0,920	85,473	0,905	0,907	90,588	0,888	0,880	88,306	0,841	0,822	82,893	0,905	0,871	88,314
	Boderline	0,865	0,849	85,445	0,746	0,920	85,755	0,794	0,947	89,163	0,849	0,956	91,726	0,833	0,920	88,889	0,858	0,893	88,028
	Smote	0,865	0,858	86,016	0,738	0,929	86,040	0,818	0,933	89,163	0,818	0,929	88,889	0,850	0,849	84,926	0,873	0,907	89,449
	Oversampling	0,865	0,787	81,469	0,714	0,933	85,461	0,834	0,933	89,734	0,873	0,920	90,314	0,779	0,911	86,342	0,802	0,942	89,171
	Undersampling	0,865	0,787	81,473	0,738	0,929	86,036	0,786	0,929	87,738	0,841	0,951	91,163	0,873	0,880	87,751	0,834	0,933	89,738
Lettera	ADASYN	0,350	0,794	77,614	0,989	0,860	86,534	0,944	0,991	98,880	0,953	0,999	99,680	0,913	0,997	99,340	0,757	0,998	98,875
	Boderline	0,247	0,964	93,565	0,437	0,824	80,899	0,973	0,904	90,634	0,971	0,996	99,490	0,956	0,997	99,535	0,900	0,998	99,370
	Smote	0,173	0,995	96,270	0,896	0,976	97,280	0,951	0,971	97,000	0,978	0,999	99,805	0,957	0,998	99,595	0,937	0,994	99,190
	Oversampling	0,782	0,993	98,490	0,896	0,974	97,050	0,949	0,972	97,105	0,987	0,975	97,505	0,972	0,970	96,990	0,947	0,968	96,745
	Undersampling	0,868	0,983	97,855	0,890	0,973	97,010	0,949	0,969	96,820	1,000	0,928	93,953	1,000	0,922	93,488	0,971	0,911	92,093
NewThyroid	ADASYN	0,902	0,923	92,969	1,000	0,822	85,116	1,000	0,950	95,814	0,943	0,989	98,140	0,914	0,983	97,209	0,943	0,978	97,209
	Boderline	1,000	0,967	97,209	1,000	0,933	94,419	1,000	0,978	98,140	1,000	0,967	97,209	0,914	0,950	94,419	0,914	0,961	95,349
	Smote	0,971	0,950	95,349	1,000	0,928	93,953	1,000	0,967	97,209	0,943	0,983	97,674	0,886	0,978	96,279	0,886	0,983	96,744
	Oversampling	1,000	0,983	98,605	1,000	0,939	94,884	1,000	0,989	99,070	0,971	0,989	98,605	0,886	0,961	94,884	0,886	0,944	93,488
	Undersampling	0,943	0,989	98,140	1,000	0,939	94,884	0,914	0,994	98,140	0,688	0,742	70,705	0,660	0,758	69,399	0,696	0,769	72,140
Pima	ADASYN	0,921	0,909	91,755	0,750	0,720	73,959	0,738	0,765	74,747	0,724	0,732	72,659	0,740	0,743	74,086	0,768	0,706	74,613
	Boderline	0,706	0,750	72,142	0,774	0,679	74,090	0,752	0,731	74,483	0,690	0,750	71,107	0,736	0,728	73,318	0,750	0,672	72,272
	Smote	0,730	0,732	73,054	0,772	0,683	74,093	0,744	0,720	73,574	0,752	0,735	74,607	0,748	0,649	71,358	0,730	0,739	73,299
	Oversampling	0,722	0,724	72,275	0,774	0,675	73,960	0,752	0,705	73,574	0,702	0,739	71,481	0,714	0,739	72,266	0,690	0,732	70,445
	Undersampling	0,724	0,694	71,361	0,774	0,694	74,613	0,772	0,713	75,134	0,797	0,884	87,584	0,673	0,919	89,495	0,574	0,931	89,650
satimage	ADASYN	0,795	0,712	74,173	0,925	0,678	70,194	0,940	0,573	60,839	0,701	0,947	92,354	0,620	0,938	90,707	0,552	0,952	91,329
	Boderline	0,186	0,957	88,174	0,879	0,744	75,695	0,885	0,614	63,994	0,818	0,900	89,215	0,695	0,918	89,666	0,689	0,921	89,868
	Smote	0,395	0,951	89,728	0,872	0,756	76,721	0,928	0,588	62,067	0,706	0,945	92,199	0,594	0,945	91,096	0,707	0,915	89,510
	Oversampling	0,557	0,938	90,070	0,891	0,743	75,742	0,911	0,594	62,471	0,895	0,842	84,755	0,827	0,822	82,284	0,871	0,826	83,015
	Undersampling	0,850	0,838	83,916	0,891	0,741	75,524	0,947	0,570	60,699	0,949	0,943	94,448	0,909	0,920	91,728	0,939	0,870	88,669
Vehicle	ADASYN	0,848	0,790	80,107	0,834	0,660	70,065	0,995	0,913	93,258	0,924	0,974	96,222	0,889	0,944	93,144	0,945	0,909	91,738
	Boderline	0,915	0,677	73,293	0,889	0,653	70,892	0,990	0,940	95,151	0,919	0,937	93,265	0,895	0,930	92,198	0,900	0,892	89,378
	Smote	0,924	0,708	75,896	0,940	0,643	71,248	0,995	0,921	93,850	0,935	0,972	96,339	0,884	0,961	94,327	0,919	0,900	90,437
	Oversampling	0,944	0,705	76,131	0,955	0,644	71,719	0,990	0,943	95,386	0,965	0,943	94,803	0,935	0,923	92,555	0,945	0,843	86,663
	Undersampling	0,940	0,699	75,542	0,945	0,644	71,483	0,995	0,910	93,021	0,950	0,939	94,292	0,971	0,948	95,606	0,992	0,968	97,656

A seguir, é feito uma análise por base, considerando os diferentes métodos de balanceamento com os diferentes classificadores (Tabela 5.3). Uma comparação é feita com os resultados obtidos com a base original (Tabela 5.2), o objetivo é poder dizer os métodos que se destacam.

- **Breast -** É possível perceber que o método Oversampling utilizando os classificadores NaiveBayes, J48 e BayesNet, conseguem melhorar os valores da classe positiva mantendo a classe negativa. Outro método, que se destaca nesta base melhorando ambas as classes é o Borderline classificado pelo J48 e JRip.

- **Bupa -** Nesta base todos os métodos conseguem melhorar a classe positiva, mas não conseguem manter os valores da classe negativa, isso acontece com todos os classificadores.

- **Ecoli -** Os métodos também não conseguem melhores valores em ambas as classes nesta base, um exemplo para esclarecer o que está sendo dito, pode ser observado com o classificador SMO utilizando o método Undersampling, acontece uma melhora na classe positiva, mas uma diminuição drástica na classe negativa.

- **Flag e Haberman -** Essas bases conseguem uma melhora na classe positiva mas diminui a classe negativa, e principalmente a taxa de classificação. Vale lembrar que as bases Flag e Ecoli possuem um nível de desbalanceamento razoável em relação a Bupa e Haberman, o que não justifica a não melhora, este fato pode estar relacionado a problemas de generalização, ruídos ou outros problemas já mencionados (Seção 4.1.1).

- **Heart -** O método Undersampling é o mais frequente nesta base, pois consegue melhorar os valores da classe positiva e manter da classe negativa com o classificador NaiveBayes e melhora os valores de ambas as classes com o J48. O classificador J48 também apresenta melhoras com os métodos Borderline e Smote.

- **Ionosphere -** O método Undersampling melhora as classes positivas e negativas, consequentemente melhorando a taxa de classificação, isto é possível com a utilização

dos classificadores J48 e BayesNet. Outro método que consegue uma significativa melhora nesta base é o Borderline com os classificadores NaiveBayes e J48.

- **Lettera -** O método Smote consegue melhorar a classe positiva e manter os valores da classe negativa com os classificadores J48 e JRip. O classificador BayesNet melhora ambas as classes com os métodos Borderline e Smote.

- **New-Thyroid -** Os classificadores NaiveBayes, MultilayerP e J48 conseguem melhorar a classe positiva e manter a classe negativa com o método Oversampling. O método Adasyn consegue também melhorar e manter com o classificador J48 e JRip. Acontece o mesmo com o método Smote e classificado pelo JRip.

- **Pima -** Nesta base, o único método que consegue melhorar significativamente o valores das classes e taxa de classificação é o Adasyn, classificado pelo algoritmo NaiveBayes.

- **Satimage -** É possível perceber que a melhora na classe positiva acontece na maioria dos métodos e classificadores, mas o método que consegue melhorar ambas as classes nesta base é o Oversampling com o classificador BayesNet.

- **Vehicle -** Nesta base a maioria dos métodos consegue uma melhora na maioria dos classificadores. O método Adasyn aplicado e classificado pelo J48 e JRip melhora os valores das classes positivas e negativas e consequentemente a Acurácia. O método Smote melhora todos os valores com os classificadores NaiveBayes, J48 e JRip. O método Oversampling com os classificadores NaiveBayes e BayesNet e o método Undersampling com os classificadores NaiveBayes, JRip e BayesNet.

Considerando os diferentes métodos de balanceamento com os diferentes classificadores, é possível perceber uma frequência de alguns métodos em cada base, ou seja, alguns métodos conseguem melhorar os valores de RecallP e RecallN com diversos classificadores. De acordo com essa frequência uma análise foi feita entre os classificadores, com o intuito de comparar esse resultado com o resultado da base original utilizando o mesmo classi-

ficador. A Tabela 5.4 apresenta a porcentagem de melhoria do método que se destaca com a base original.

Tabela 5.4: % de Melhoria dos métodos com a base original

Base	Métodos	Classificador	RecallP	RecallN	Acurácia
breast	Original	Jrip	0,958	0,953	95,460
	Borderline	Jrip	0,975	0,957	96,339
	% de Melhoria		**1,77%**	**0,42%**	**0,92%**
Glass	Original	J48	0,283	0,964	91,118
	Oversampling	J48	0,417	0,964	92,049
	% de Melhoria		**47,35%**	**0,00%**	**1,02%**
Heart	Original	J48	0,717	0,780	75,185
	Undersampling	J48	0,767	0,787	77,778
	% de Melhoria		**6,97%**	**0,90%**	**3,45%**
Ionosphere	Original	J48	0,818	0,911	87,738
	Borderline	J48	0,849	0,956	91,726
	% de Melhoria		**3,79%**	**4,94%**	**4,55%**
Lettera	Original	BayesNet	0,848	0,992	98,605
	Smote	BayesNet	0,937	0,994	99,19
	% de Melhoria		**10,50%**	**0,20%**	**0,59%**
NewThyroid	Original	Jrip	0,857	0,978	95,814
	Adasyn	Jrip	0,914	0,983	97,209
	% de Melhoria		**6,67%**	**0,57%**	**1,46%**
Pima	Original	NaiveBayes	0,848	0,608	76,433
	Adasyn	NaiveBayes	0,921	0,909	91,755
	% de Melhoria		**8,61%**	**49,48%**	**20,05%**
Vehicle	Original	BayesNet	0,945	0,683	74,467
	Undersampling	BayesNet	0,992	0,968	97,656
	% de Melhoria		**4,94%**	**41,78%**	**31,14%**

As bases Bupa, Ecoli, Flag, Haberman e Satimage não aparecem na Tabela 5.4 por não ter um método que melhore ambas as classes em nenhum dos classificadores. As bases Pima e Vehicle por exemplo, se destacam entre as bases, pois conseguem um aumento na taxa de classificação, mas evidentemente com métodos e classificadores diferentes. Para melhor visualizar, a Figura 5.1 apresenta um gráfico com os melhores resultados apresentados na Tabela 5.4.

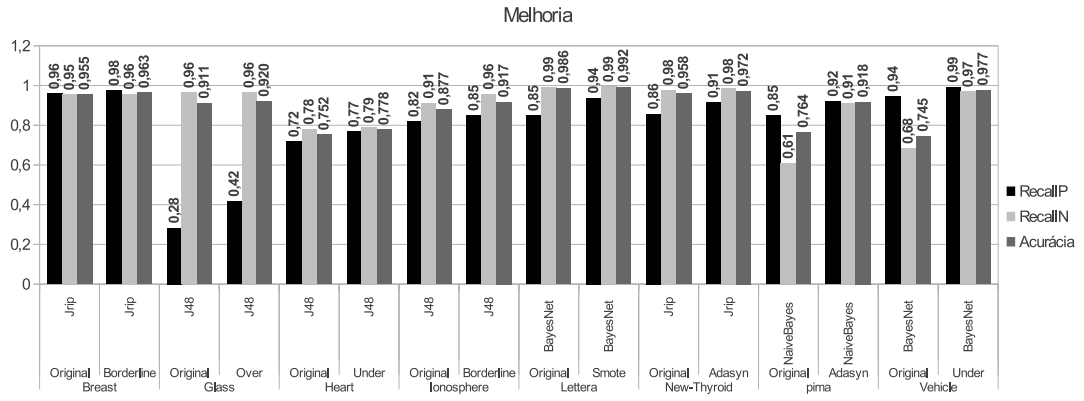

Figura 5.1: Base original e base com os métodos de balanceamento aplicados.

O destaque da base Pima é o método Adasyn com o classificador NaiveBayes, e da base Vehicle, o método Undersampling com o classificador BayesNet. Métodos esses que trabalham de forma totalmente diferente, o Adasyn por exemplo, tem a função de

aumentar a classe minoritária acrescentando exemplos sintéticos próximos a superfície de decisão, e Undersampling, tem a função de diminuir os exemplos da classe majoritária.

Pode-se observar na Figura 5.1 que não é possível dizer que um método é melhor que o outro, pois os métodos possuem comportamentos diferentes em cada base, é importante ressaltar também, que o classificador pode influenciar nos resultados. Comprovando este fato, pode ser visto nos resultados obtidos no trabalho de Haibo He et.al. [23], os resultados para a base Pima com o Algoritmo Adasyn aplicado e comparado com o algoritmo Smote e ambos classificados por um algoritmo de árvore de decisão. Os resultados do trabalho de Haibo He et.al. [23] demonstram que o Adasyn obtém o melhor resultado em relação a taxa de acurácia 68% e 60% de Recall, e pode ser observado nos resultados obtidos nos experimentos deste trabalho que o classificador que se destaca com resultados melhores é o NaiveBayes, tendo 92% de RecallP e 91,77% de acurácia.

A Tabela 5.5 apresenta os resultados das bases originais utilizando os métodos híbridos. Esses métodos são chamados híbridos por terem uma combinação entre métodos de pré-processamento e classificadores.

Observando os resultados com os métodos híbridos (Tabela 5.5), é possível notar resultados razoáveis para a classe minoritária, sem diminuir drasticamente os valores da classe majoritária, se comparado com os resultados obtidos na Tabela 5.2. Os métodos que se destacam são: *BalanceCascade, EasyEnsemble e SmoteBoost*, é importante observar também, que esses métodos conseguem valores razoáveis da classe positiva, principalmente nas bases que o nível de desbalanceamento é grande, como por exemplo, Lettera, Glass, Flag, Satimage, Ecoli, New-thyroid e Vehicle.

A Tabela 5.6 apresenta os resultados com os métodos de balanceamento classificados com os métodos híbridos. É notório que os resultados para a classe positiva (RecallP) são melhores na maioria das bases, no entanto, os resultados são piores para a classe negativa.

Tabela 5.5: Métodos Híbridos

Base	**Híbridos**	**RecallP**	**RecallN**	**AUC**	**Acurácia**
breast	BalanceCascade	0,980	0,940	0,966	96,00
	EasyEnsemble	0,960	0,960	0,964	96,00
	SMOTEBagging	0,940	0,960	0,957	95,00
	SMOTEBoost	0,940	0,960	0,955	95,00
bupa	BalanceCascade	0,910	0,240	0,578	52,00
	EasyEnsemble	0,710	0,640	0,678	67,00
	SMOTEBagging	0,510	0,760	0,641	66,00
	SMOTEBoost	0,600	0,720	0,666	67,00
Ecoli	BalanceCascade	0,940	0,830	0,888	84,00
	EasyEnsemble	0,880	0,790	0,841	80,00
	SMOTEBagging	0,820	0,880	0,858	88,00
	SMOTEBoost	0,800	0,940	0,872	92,00
Flag	BalanceCascade	0,820	0,610	0,725	63,00
	EasyEnsemble	0,820	0,550	0,694	57,00
	SMOTEBagging	0,470	0,840	0,665	81,00
	SMOTEBoost	0,290	0,900	0,596	85,00
Glass	BalanceCascade	0,880	0,540	0,707	57,00
	EasyEnsemble	0,760	0,650	0,713	66,00
	SMOTEBagging	0,580	0,850	0,726	83,00
	SMOTEBoost	0,580	0,900	0,752	87,00
haberman	BalanceCascade	0,660	0,610	0,643	63,00
	EasyEnsemble	0,560	0,760	0,665	70,00
	SMOTEBagging	0,550	0,750	0,655	70,00
	SMOTEBoost	0,590	0,720	0,659	68,00
heart	BalanceCascade	0,920	0,360	0,643	61,00
	EasyEnsemble	0,830	0,740	0,787	78,00
	SMOTEBagging	0,760	0,860	0,813	81,00
	SMOTEBoost	0,750	0,780	0,765	76,00
ionosphere	BalanceCascade	0,890	0,840	0,869	86,00
	EasyEnsemble	0,860	0,910	0,890	89,00
	SMOTEBagging	0,840	0,940	0,894	90,00
	SMOTEBoost	0,840	0,950	0,903	91,00
Lettera	BalanceCascade	0,980	0,970	0,981	97,00
	EasyEnsemble	0,980	0,970	0,979	97,00
	SMOTEBagging	1,00	0,000	0,500	3,00
	SMOTEBoost	1,00	0,000	0,500	3,00
New-thyroid	BalanceCascade	0,970	0,930	0,952	93,00
	EasyEnsemble	0,850	0,960	0,912	94,00
	SMOTEBagging	0,910	0,950	0,935	94,00
	SMOTEBoost	0,970	0,960	0,969	96,00
pima	BalanceCascade	0,540	0,860	0,706	65,00
	EasyEnsemble	0,670	0,760	0,722	70,00
	SMOTEBagging	0,770	0,690	0,733	74,00
	SMOTEBoost	0,740	0,710	0,731	73,00
Satimage	BalanceCascade	0,870	0,830	0,856	83,00
	EasyEnsemble	0,860	0,820	0,847	83,00
	SMOTEBagging	0,720	0,940	0,833	92,00
	SMOTEBoost	0,710	0,950	0,836	93,00
Vehicle	BalanceCascade	0,980	0,890	0,941	91,00
	EasyEnsemble	0,960	0,910	0,939	92,00
	SMOTEBagging	0,950	0,930	0,947	94,00
	SMOTEBoost	0,950	0,950	0,959	95,00

Tabela 5.6: Pré-processamento com os Métodos Híbridos

Base	Métodos	BalanceCascade			EasyEnsemble			SmoteBagging			SmoteBoost		
		RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia
breast	ADASYN	0,920	0,950	94,000	0,960	0,950	96,000	0,990	0,920	94,000	0,950	0,960	96,000
	Boderline	0,770	0,970	90,000	0,950	0,960	96,000	0,970	0,940	95,000	0,960	0,960	96,000
	Smote	0,880	0,970	94,000	0,970	0,960	96,000	0,990	0,940	96,000	0,940	0,960	95,000
	Oversampling	0,910	0,970	95,000	0,950	0,960	96,000	0,980	0,940	95,000	0,950	0,960	96,000
	Undersampling	0,910	0,970	95,000	0,950	0,940	95,000	0,980	0,940	95,000	0,950	0,950	95,000
Bupa	ADASYN	0,280	0,840	61,000	0,520	0,740	64,000	0,810	0,470	61,000	0,600	0,620	61,000
	Boderline	0,240	0,920	64,000	0,570	0,770	68,000	0,760	0,540	63,000	0,640	0,650	64,000
	Smote	0,070	0,950	58,000	0,500	0,810	68,000	0,800	0,560	66,000	0,600	0,710	66,000
	Oversampling	0,270	0,930	65,000	0,460	0,750	63,000	0,780	0,570	66,000	0,560	0,710	65,000
	Undersampling	0,170	0,950	62,000	0,560	0,730	66,000	0,750	0,600	66,000	0,680	0,670	67,000
Ecoli	ADASYN	0,570	0,930	89,000	0,680	0,900	87,000	0,910	0,850	85,000	0,710	0,900	88,000
	Boderline	0,510	0,960	91,000	0,680	0,940	91,000	0,850	0,890	88,000	0,680	0,940	91,000
	Smote	0,480	0,980	92,000	0,600	0,940	90,000	0,800	0,890	88,000	0,570	0,960	92,000
	Oversampling	0,510	0,970	92,000	0,710	0,940	91,000	0,880	0,880	88,000	0,710	0,940	91,000
	Undersampling	0,480	0,950	90,000	0,800	0,910	90,000	0,850	0,860	86,000	0,740	0,850	83,000
Flag	ADASYN	0,110	0,940	87,000	0,110	0,870	80,000	0,410	0,720	69,000	0,170	0,920	85,000
	Boderline	0,110	0,960	88,000	0,050	0,930	85,000	0,170	0,900	84,000	0,170	0,950	88,000
	Smote	0,290	0,930	88,000	0,290	0,870	82,000	0,580	0,880	85,000	0,410	0,960	91,000
	Oversampling	0,110	0,970	89,000	0,640	0,880	86,000	0,640	0,750	74,000	0,290	0,940	89,000
	Undersampling	0,230	0,720	68,000	0,880	0,640	67,000	0,700	0,670	67,000	0,880	0,650	67,000
Glass	ADASYN	0,230	0,970	92,000	0,470	0,930	89,000	0,760	0,790	78,000	0,520	0,920	89,000
	Boderline	0,290	0,960	91,000	0,350	0,950	91,000	0,410	0,910	87,000	0,410	0,960	92,000
	Smote	0,290	0,990	93,000	0,410	0,950	91,000	0,470	0,900	87,000	0,350	0,950	91,000
	Oversampling	0,350	0,950	90,000	0,410	0,910	87,000	0,820	0,780	78,000	0,410	0,910	87,000
	Undersampling	0,230	0,900	85,000	0,640	0,630	63,000	0,520	0,590	59,000	0,640	0,620	62,000
Haberman	ADASYN	0,140	0,910	71,000	0,580	0,690	66,000	0,640	0,700	68,000	0,600	0,710	68,000
	Boderline	0,280	0,870	71,000	0,530	0,740	68,000	0,610	0,730	70,000	0,500	0,780	70,000
	Smote	0,160	0,900	70,000	0,500	0,730	67,000	0,640	0,660	66,000	0,510	0,690	65,000
	Oversampling	0,250	0,860	70,000	0,550	0,740	69,000	0,580	0,720	68,000	0,590	0,680	66,000
	Undersampling	0,120	0,920	70,000	0,640	0,680	67,000	0,610	0,730	70,000	0,580	0,700	67,000
Heart	ADASYN	0,310	0,960	67,000	0,740	0,810	78,000	0,810	0,740	77,000	0,770	0,800	78,000
	Boderline	0,150	0,980	61,000	0,680	0,850	77,000	0,740	0,750	74,000	0,760	0,770	77,000
	Smote	0,200	0,950	61,000	0,680	0,840	77,000	0,800	0,770	78,000	0,730	0,790	76,000
	Oversampling	0,190	0,980	62,000	0,700	0,820	77,000	0,800	0,760	77,000	0,790	0,800	80,000
	Undersampling	0,100	0,960	57,000	0,680	0,790	74,000	0,800	0,760	77,000	0,800	0,740	77,000
Ionosphere	ADASYN	0,700	0,960	86,000	0,840	0,950	91,000	0,920	0,830	86,000	0,920	0,920	92,000
	Boderline	0,690	0,980	88,000	0,820	0,960	91,000	0,890	0,890	89,000	0,870	0,940	92,000
	Smote	0,690	0,980	88,000	0,800	0,960	90,000	0,900	0,880	89,000	0,840	0,960	92,000
	Oversampling	0,730	0,980	89,000	0,800	0,920	88,000	0,890	0,880	89,000	0,860	0,940	91,000
	Undersampling	0,750	0,930	86,000	0,810	0,900	87,000	0,880	0,890	89,000	0,840	0,960	92,000
Letter-a	ADASYN	1,00	0,000	3,00	1,00	0,000	3,00	1,00	0,000	3,00	1,00	0,000	3,00
	Boderline	1,00	0,000	3,00	1,00	0,000	3,00	1,00	0,000	3,00	1,00	0,000	3,00
	Smote	1,00	0,000	3,00	1,00	0,000	3,00	1,00	0,000	3,00	1,00	0,000	3,00
	Oversampling	1,00	0,000	3,00	1,00	0,000	3,00	1,00	0,000	3,00	1,00	0,000	3,00
	Undersampling	0,860	0,990	98,000	0,970	0,970	97,000	0,990	0,920	92,000	0,980	0,980	98,000
New-Thyroid	ADASYN	0,820	0,980	96,000	0,940	0,950	94,000	0,970	0,930	94,000	1,00	0,960	96,000
	Boderline	0,880	1,00	98,000	0,880	0,980	96,000	0,910	0,980	97,000	0,880	0,970	96,000
	Smote	0,850	0,980	96,000	0,880	0,980	96,000	0,910	0,970	96,000	0,850	0,980	96,000
	Oversampling	0,850	0,970	95,000	0,850	0,960	94,000	0,940	0,930	93,000	0,910	0,940	93,000
	Undersampling	0,940	0,920	93,000	0,940	0,890	90,000	0,910	0,910	91,000	0,910	0,930	93,000
Pima-New	ADASYN	0,240	0,950	49,000	0,640	0,760	68,000	0,770	0,620	72,000	0,660	0,790	70,000
	Boderline	0,390	0,920	57,000	0,710	0,760	73,000	0,830	0,570	74,000	0,730	0,720	72,000
	Smote	0,490	0,880	62,000	0,730	0,710	72,000	0,800	0,620	74,000	0,800	0,640	74,000
	Oversampling	0,360	0,940	56,000	0,680	0,730	70,000	0,810	0,600	74,000	0,690	0,750	71,000
	Undersampling	0,250	0,960	50,000	0,650	0,760	69,000	0,780	0,650	73,000	0,750	0,720	74,000
satimage	ADASYN	0,460	0,970	92,000	0,650	0,940	91,000	0,980	0,630	67,000	0,690	0,950	93,000
	Boderline	0,540	0,970	92,000	0,610	0,950	92,000	0,930	0,730	75,000	0,690	0,960	94,000
	Smote	0,490	0,970	92,000	0,600	0,960	92,000	0,960	0,680	71,000	0,610	0,970	94,000
	Oversampling	0,510	0,960	92,000	0,670	0,940	92,000	0,970	0,670	70,000	0,700	0,960	93,000
	Undersampling	0,470	0,950	91,000	0,800	0,870	86,000	0,930	0,730	75,000	0,900	0,840	84,000
Vehicle	ADASYN	0,840	0,960	93,000	0,920	0,940	94,000	0,980	0,840	88,000	0,960	0,950	95,000
	Boderline	0,550	0,980	88,000	0,900	0,960	94,000	0,970	0,890	91,000	0,950	0,970	97,000
	Smote	0,730	0,970	92,000	0,870	0,960	94,000	0,980	0,870	90,000	0,930	0,970	96,000
	Oversampling	0,820	0,960	93,000	0,910	0,940	93,000	0,990	0,820	86,000	0,950	0,960	96,000
	Undersampling	0,640	0,950	88,000	0,940	0,900	91,000	0,970	0,870	90,000	0,960	0,910	92,000

A análise da Tabela 5.6 é comparada com os métodos híbridos aplicados à base original (Tabela 5.5).

- **Breast -** O método Smote melhora a classe positiva mantendo a classe negativa se classificada pelo método hibrido EasyEnsemble. O mesmo acontece com os métodos Adasyn, Borderline e Oversampling classificados pelo SmoteBoost.

- **Bupa -** Pode-se observar a melhora significativa da classe negativa com o método BalanceCascade com todos os métodos de pré-processamento, mas é notório também os péssimos valores da classe de interesse (positiva). Nesta base um método que pode ser destacado é SmoteBoost com o método Smote, pois quase mantem os valores das duas classes.

- **Ecoli -** O método híbrido SmoteBagging consegue melhorar a classe positiva e manter a classe negativa com o método de pré-processamento Oversampling. O destaca vai para o método de pré-processamento Borderline, que consegue melhorar ambas as classes com este mesmo classificador.

- **Flag -** Nesta base os resultados são satisfatórios, pois o métodos de pré-processamento Smote consegue aumentar os valores de ambas as classes, com os classificadores SmoteBagging e SmoteBoost, Oversampling com o SmoteBoost e o método Undersampling com o EasyEnsemble.

- **Glass, Haberman e Satimage -** Os métodos, tanto para pré-processamento quanto os híbridos, não conseguem melhorar a classe positiva sem diminuir drasticamente os valores da classe negativa.

- **Heart -** O método híbrido SmoteBoost, utilizando o método de pré-processamento Adasyn e Oversampling, consegue melhorar ambas as classes nesta base.

- **Ionosphere -** SmoteBoost com o método Smote e Undersampling melhora a classe negativa mantendo o valor da classe positiva, consequentemente melhorando a taxa de classificação.

- **Lettera -** O único método de pré-processamento que consegue predizer a classe negativa é o Undersampling. O restante dos métodos não consegue predizer nenhum exemplo da classe positiva, e acerta 100% da classe positiva.

- **New-Thyroid -** Os métodos Borderline e Smote conseguem melhoras nas classes positiva e negativa, classificados pelos métodos híbridos EasyEnsemble. O método Oversampling mantem uma das classes com o EasyEnsemble, o mesmo acontece com Borderline com SmoteBagging e Adasyn com SmoteBoost.

- **Pima -** O Método híbrido SmoteBoost aumenta razoavelmente os valores das classes com o método de pré-processamento Undersampling, enquanto que o Borderline com o EasyEnsemble aumenta apenas a classe positiva.

- **Vehicle -** Nesta base, o método híbrido SmoteBoost consegue aumentar os valores de uma das classes mantendo os valores da outra, melhorando assim a taxa de classificação, os métodos em destaque são Adasyn, Borderline e Oversampling.

A Figura 5.2 apresenta um gráfico com os resultados das bases que os métodos conseguiram algum resultado.

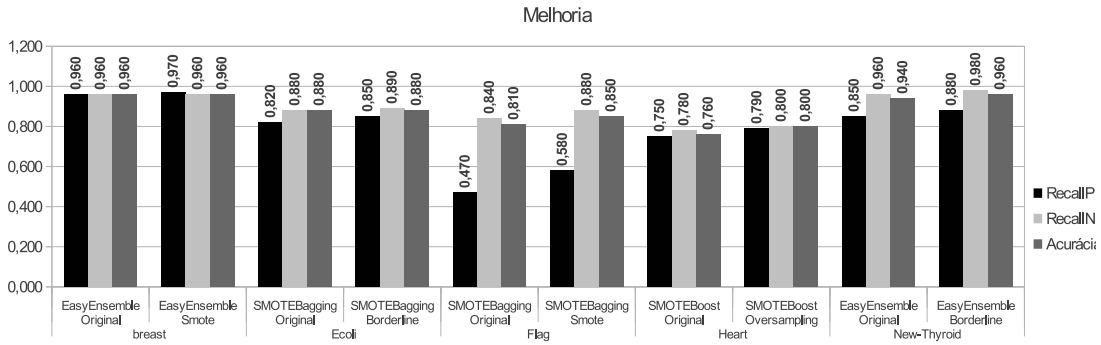

Figura 5.2: Base original e base com os métodos de balanceamento aplicados

Os resultados afirmam que os métodos de balanceamento melhoram a capacidade de predição dependendo de como o classificador generaliza a base, e principalmente o nível de desbalanceamento. A piora em algumas bases com alguns métodos de balanceamento pode ter ocorrido por terem movido a fronteira de decisão, dificultando ainda mais o poder de generalização dos métodos híbridos.

CAPÍTULO 6

ESTUDO DE CASO

Nesta Seção, será apresentado um estudo sobre métodos para resolver problemas de classificação em bases desbalanceadas utilizando bases do mundo real. As bases utilizadas foram as bases da Engenharia de software. Vale ressaltar que foram utilizadas para esses experimentos as mesmas métricas de avaliação: RecallP, RecallN e Acurácia. Os resultados são apresentados a seguir.

6.1 Classificação - Base *NASA*

Foram utilizados 5 conjuntos de dados do repositório da *NASA Metrics Data Project*. Foram desenvolvidos na linguagem C as bases PC1, CM1 e JM1, e as outras duas bases (KC1 e KC2) foram desenvolvidos em C++.

As principais características estão relacionadas na Tabela 6.1.

Tabela 6.1: Caracteristicas das Bases

Base	**Instâncias**	**Atributos**	**Classe**		**%**
			True	**False**	
PC1	1109	22	77	1032	7%
CM1	498	22	49	449	11%
KC1	2109	22	326	1783	18%
JM1	10885	22	2106	8779	24%
KC2	522	22	107	415	26%

A Tabela 6.1 está ordenada pela base que possui maior nível de desbalanceamento, ou seja, é possível perceber que a classe *True* possui uma quantidade menor de instâncias em relação a classe *False*, dificultando assim o poder de predição do classificador.

Sabendo o nível de desbalanceamento entre as classes, foram aplicados nessas bases, os classificadores tradicionais. O objetivo é observar o poder de predição dos classificadores nesse tipo de base. A Tabela 6.2 apresenta os resultados da base original *NASA Metrics data Project*, utilizando os classificadores tradicionais.

Tabela 6.2: Classificação Base Original - NASA

Base	Classificador	RecallT	RecallF	Acurácia
cm1	NaiveBayes	0,286	0,904	84,34
	SMO	0,000	0,996	89,76
	MultilayerP	0,041	0,978	88,55
	J48	0,082	0,969	88,15
	Jrip	0,061	0,973	88,35
	BayesNet	0,694	0,659	66,27
jm1	NaiveBayes	0,206	0,948	80,45
	SMO	0,004	1,0	80,73
	MultilayerP	0,073	0,989	81,15
	J48	0,246	0,938	80,45
	Jrip	0,162	0,965	80,93
	BayesNet	0,585	0,703	68,05
kc1	NaiveBayes	0,377	0,905	82,36
	SMO	0,021	0,996	84,50
	MultilayerP	0,267	0,966	85,78
	J48	0,291	0,934	83,50
	Jrip	0,248	0,950	84,12
	BayesNet	0,733	0,696	70,18
kc2	NaiveBayes	0,411	0,947	83,72
	SMO	0,224	0,990	83,33
	MultilayerP	0,467	0,942	84,48
	J48	0,523	0,884	81,03
	Jrip	0,477	0,916	82,57
	BayesNet	0,738	0,810	79,50
pc1	NaiveBayes	0,299	0,932	88,82
	SMO	0,000	1,0	93,06
	MultilayerP	0,156	0,990	93,24
	J48	0,195	0,986	93,15
	Jrip	0,234	0,987	93,51
	BayesNet	0,506	0,764	74,57

De acordo com os resultados, podemos perceber que a taxa de classificação é baixo, e se a análise for feita entre as classes, os resultados são piores ainda para a classe minoritária (RecallT), explicando assim os valores obtidos na acurácia. Pode-se observar que cada classificador se comporta de uma maneira, e se apontarmos o classificador que tem a maior taxa de classificação em cada base, seria o classificador J48. Mas não significa que o J48 seria o melhor classificador, pois se a comparação for feita entre as classes, é evidente os valores significamente baixos para a classe minoritária (RecallT), e este fato acontece em todas as Bases.

Para tentar resolver esse problema de classificação, ou seja, resolver os problemas dos baixos valores para a classe minoritária, foram aplicados então os métodos de balanceamento.

Os resultados com os métodos de balanceamento aplicados e classificados pelos algo-

ritmos tradicionais são apresentados na Tabela 6.3.

Tabela 6.3: Métodos de Pré-processamento

Base	Método	Naive Bayes			SMO			MultilayerP			J48			Jrip			BayesNet		
		RecallT	RecallF	Acurácia	RecallT	RecallF	Acurácia	RecallT	RecallF	Acurácia	RecallT	RecallF	Acurácia	RecallT	RecallF	Acurácia	RecallT	RecallF	Acurácia
cm1	Adasyn	0,264	0,917	85,33	0,360	0,909	85,53	0,549	0,617	61,06	0,409	0,886	83,93	0,367	0,864	81,53	0,409	0,826	78,50
	Borderline	0,364	0,846	79,91	0,633	0,799	78,31	0,471	0,820	78,53	0,289	0,880	82,13	0,307	0,853	79,90	0,556	0,784	76,09
	Smote	0,364	0,880	82,93	0,591	0,764	74,68	0,660	0,660	65,87	0,433	0,818	77,91	0,633	0,800	78,31	0,651	0,697	69,28
	Oversampling	0,324	0,882	82,73	0,653	0,788	77,50	0,598	0,777	75,90	0,304	0,902	84,33	0,311	0,895	83,74	0,691	0,699	69,87
	Undersampling	0,364	0,877	82,73	0,447	0,857	81,72	0,609	0,788	77,10	0,776	0,523	54,79	0,691	0,670	67,28	0,693	0,704	70,30
jm1	Adasyn	0,245	0,934	80,08	0,370	0,870	77,32	0,690	0,597	61,47	0,413	0,815	73,72	0,366	0,871	77,32	0,477	0,791	73,04
	Borderline	0,216	0,943	80,26	0,375	0,874	77,77	0,642	0,652	65,00	0,328	0,866	76,20	0,310	0,884	77,27	0,408	0,841	75,76
	Smote	0,224	0,940	80,17	0,357	0,879	77,78	0,672	0,612	62,35	0,336	0,876	77,12	0,319	0,882	77,27	0,386	0,865	77,23
	Oversampling	0,217	0,944	80,37	0,360	0,877	77,67	0,628	0,655	64,98	0,420	0,800	72,62	0,596	0,653	64,20	0,607	0,663	65,20
	Undersampling	0,223	0,942	80,30	0,338	0,883	77,78	0,532	0,735	69,53	0,612	0,649	64,21	0,585	0,692	67,16	0,556	0,718	68,65
kc1	Adasyn	0,420	0,887	81,51	0,696	0,732	72,69	0,791	0,545	58,32	0,417	0,900	82,50	0,371	0,896	81,51	0,767	0,669	68,38
	Borderline	0,395	0,898	82,08	0,653	0,756	74,02	0,659	0,744	73,07	0,463	0,887	82,12	0,438	0,903	83,12	0,555	0,835	79,14
	Smote	0,392	0,898	82,03	0,714	0,708	70,94	0,828	0,597	63,25	0,494	0,874	81,51	0,451	0,860	79,71	0,582	0,810	77,48
	Oversampling	0,395	0,900	82,17	0,705	0,718	71,60	0,758	0,623	64,39	0,436	0,879	81,08	0,533	0,826	78,10	0,733	0,691	69,80
	Undersampling	0,396	0,899	82,12	0,693	0,734	72,74	0,807	0,625	65,29	0,712	0,693	69,56	0,822	0,616	64,77	0,794	0,653	67,52
kc2	Adasyn	0,411	0,925	81,99	0,578	0,892	82,76	0,439	0,945	84,10	0,589	0,843	79,11	0,590	0,812	76,62	0,626	0,875	82,38
	Borderline	0,476	0,918	82,75	0,756	0,800	79,12	0,626	0,855	80,84	0,543	0,834	77,41	0,627	0,846	80,07	0,729	0,817	79,89
	Smote	0,448	0,928	82,95	0,775	0,786	78,35	0,682	0,841	80,83	0,626	0,805	76,82	0,718	0,812	79,31	0,775	0,783	78,16
	Oversampling	0,402	0,937	82,76	0,756	0,805	79,50	0,634	0,839	79,70	0,559	0,877	81,22	0,654	0,800	76,99	0,766	0,790	78,54
	Undersampling	0,439	0,935	83,34	0,748	0,795	78,54	0,673	0,831	79,88	0,755	0,764	76,24	0,709	0,795	77,77	0,747	0,783	77,58
pc1	Adasyn	0,297	0,926	88,28	0,478	0,839	81,42	0,740	0,710	71,22	0,531	0,918	89,09	0,530	0,881	85,66	0,674	0,733	72,86
	Borderline	0,335	0,909	86,93	0,452	0,875	84,58	0,608	0,833	81,79	0,311	0,953	90,80	0,300	0,954	90,89	0,363	0,912	87,37
	Smote	0,310	0,918	87,56	0,440	0,857	82,78	0,817	0,675	68,52	0,533	0,920	89,27	0,504	0,894	86,74	0,453	0,867	83,86
	Oversampling	0,297	0,931	88,73	0,425	0,875	84,40	0,703	0,801	79,43	0,402	0,937	89,99	0,388	0,928	89,09	0,467	0,873	84,49
	Undersampling	0,297	0,916	87,29	0,336	0,921	88,01	0,530	0,828	80,71	0,871	0,626	64,30	0,807	0,647	65,82	0,699	0,665	66,73

Como pode ser observado na tabela 6.3, em algumas bases nenhum dos métodos consegue aumentar os valores das duas classes. Na maioria das vezes acontece a melhora na classe minoritária mas uma piora nos valores da classe majoritária. A seguir estão descritos detalhadamente os acontecimentos em cada base.

- **Cm1 -** Nesta base, fica evidente a melhora da classe minoritária (True) na maioria dos métodos e na maioria dos classificadores, e uma diminuição drástica na classe majoritária. Nos classificadores SMO, MultilayerP e J48 todos os métodos conseguem melhorar a classe verdadeira (RecallT), mas diminui drasticamente a classe falsa (RecallF). Os Métodos Oversampling e Undersampling se destacam com o classificador BayesNet, pois melhoram a classe negativa e quase mantem a classe positiva e consequentemente melhora a acurácia.

- **Jm1 -** O classificador Naive Bayes, consegue melhorar a classe positiva com todos os métodos, mas o único método que quase mantem a classe negativa é o método Oversampling. O BayesNet melhora a taxa de acurácia com quase todos os métodos, melhorando na maioria das vezes apenas a classe negativa.

- **Kc1 -** O método Oversampling consegue melhorar a classe positiva e quase manter o valor da classe negativa com o classificador NaiveBayes, como mostra a Figura 6.1. O Oversampling com o classificador BayesNet também se destaca, pois mantem a classe positiva e melhora a classe negativa.

- **Kc2 -** O método Undersampling consegue melhorar a classe positiva, mas com uma leve diminuição da classe negativa utilizando o classificador Naive Bayes. Com o classificador SMO, J48 e JRip todos os métodos conseguem melhorar a classe minoritária, mas diminui a classe majoritária. Com o classificador MultilayerP, o único método que não consegue melhorar a classe positiva é o Adasyn.

- **Pc1 -** Todos os métodos conseguem melhorar a classe positiva com os classificadores SMO, MultilayerP, J48 e JRip, mas diminui os valores da classe negativa. O método Borderline se destaca nesta base por não diminuir drasticamente os valores da classe negativa nos classificadores MultilayerP, J48 e JRip.

A Figura 6.1 apresenta um gráfico com os classificadores e métodos de balanceamento aplicados que conseguiram algum resultado em cada base, facilitando a visualização dos acontecimentos em cada classe.

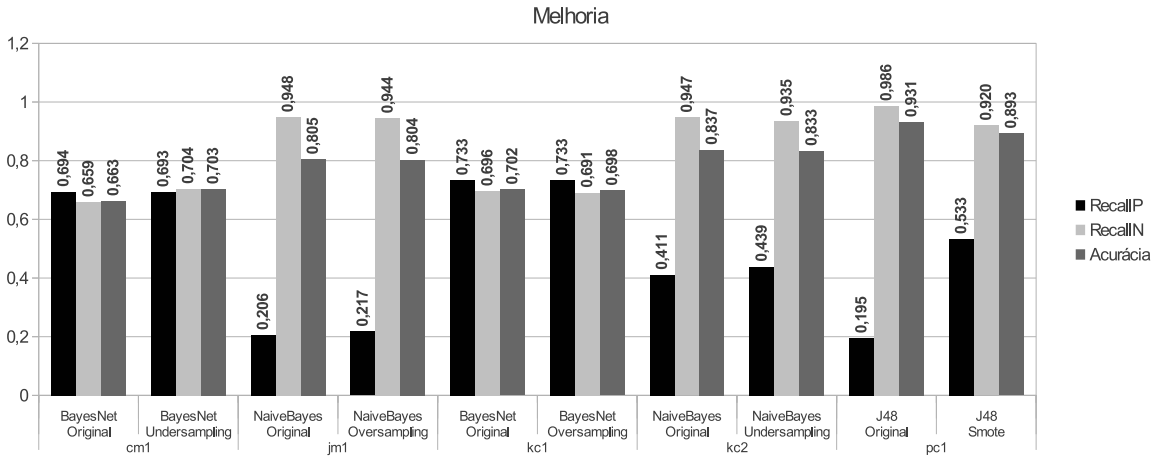

Figura 6.1: Base original e base com os métodos de balanceamento aplicados.

De acordo com os resultados, pode-se observar que a melhora não depende apenas do método de balanceamento, mas sim do tipo da base e também do classificador usado. Neste experimento, os métodos Oversampling e Undersampling são destaque na maioria das vezes, pois conseguem melhorar a classe positiva sem diminuir drasticamente a classe negativa.

A Tabela 6.4 apresenta os resultados com os métodos híbridos. Vale lembrar que o método híbrido é uma junção da fase de pré-processamento com os classificadores.

É perceptível que os resultados com os métodos híbridos são mais equilibrados em relação as classes, pois na maioria dos métodos, a diferença entre as classes não é muito grande se comparados com os resultados da base original classificados pelos algoritmos tradicionais.

- **Cm1 -** Nesta base o método híbrido que se destaca para a classe verdadeira (RecallT) é o SmoteBoost, obtendo 77% de RecallT e 65% de RecallF. Pode-se observar um certo equilíbrio entre as classes.

- **Jm1 -** Observando a taxa de classificação desta base, é possível notar que o método que se destaca é o SmoteBagging com 80% de acerto, mas se a análise for feita

Tabela 6.4: Métodos Híbridos - NASA

Base	Classificador	RecallT	RecallF	Acurácia
cm1	BalanceCascade	0,690	0,480	50,00
	EasyEnsemble	0,570	0,660	65,00
	SMOTEBagging	0,810	0,470	51,00
	SMOTEBoost	0,770	0,650	66,00
jm1	BalanceCascade	0,660	0,620	63,00
	EasyEnsemble	0,680	0,610	62,00
	SMOTEBagging	0,140	0,960	80,00
	SMOTEBoost	0,290	0,910	79,00
kc1	BalanceCascade	0,920	0,520	58,00
	EasyEnsemble	0,920	0,510	57,00
	SMOTEBagging	0,800	0,600	63,00
	SMOTEBoost	0,740	0,720	72,00
kc2	BalanceCascade	0,780	0,780	78,00
	EasyEnsemble	0,770	0,770	77,00
	SMOTEBagging	0,760	0,790	78,00
	SMOTEBoost	0,710	0,810	79,00
pc1	BalanceCascade	0,810	0,730	74,00
	EasyEnsemble	0,800	0,710	71,00
	SMOTEBagging	0,610	0,880	86,00
	SMOTEBoost	0,490	0,940	91,00

pelos valores das classes, fica evidente os baixos valores da classe minoritária. Portanto os métodos que mantém um equilíbrio entre as classes são BalanceCascade e EasyEnsemble.

- **Kc1 -** Para esta base os métodos BalanceCascade e EasyEnsemble apresenta ótimos resultados para a classe minoritária (RecallF), mas possui péssimos resultados para a classe majoritária, consequentemente baixos valores de acurácia. O método que sobressai aos demais em relação ao equilíbrio entre as classes e ainda possui um bom valor de acurácia é o SmoteBoost.

- **Kc2 -** Todos os métodos conseguem manter um certo equilíbrio entre as classes.

- **Pc1 -** O método SmoteBoost consegue um valor razoável para a classe minoritária, mas um péssimo valor da classe majoritária.

Se for observado apenas o resultado da acurácia de todas as bases, o método que sobressai aos outros é o SmoteBoost, mantendo na maioria das vezes um certo equilíbrio entre as classes. Foram aplicados então, os métodos de balanceamento classificados pelos métodos híbridos, o intuito é analisar o poder de balanceamento dos métodos de balance-

amento na fase de pré-processamento e o comportamento dos métodos híbridos nas bases com o nível de balanceamento adequado.

A Tabela 6.5 apresenta os resultados dos métodos de balanceamento na fase de pré-processamento classificados pelos métodos híbridos.

- **Cm1 -** Os métodos Adasyn e Borderline se destacam com o classificador híbrido BalanceCascade, pois melhoram ambas as classes, consequentemente melhorando a acurácia. Outro método é o Oversampling com o SmoteBagging, que também melhora todos os valores.

- **Jm1 -** Os classificadores SmoteBagging e SmoteBoost melhoram e muito a classe positiva, mas observando a classe negativa é visível os péssimos resultados. O BalanceCascade não consegue melhorar a classe positiva com nenhum dos métodos de balanceamento.

- **Kc1 -** Nesta base o método Híbrido que se destaca é o SmoteBagging, pois é o único que conseguem melhores valores da classe positiva com todos os métodos de balanceamento, dando o destaque para o Borderline, o mesmo mantem os valores de ambas as classes.

- **Kc2 -** Nesta base o método Undersampling melhora os valores das classes com o classificador EasyEnsemble, e Oversampling com o SmoteBoost.

- **Pc1 -** Apenas o classificador SmoteBagging consegue bons resultados com a maioria dos métodos.

Para melhor visualizar, a Figura 6.2 apresenta os melhores resultados para cada base, mostrando o antes com a base original e depois com os métodos de balanceamento aplicados, utilizando o mesmo método híbrido.

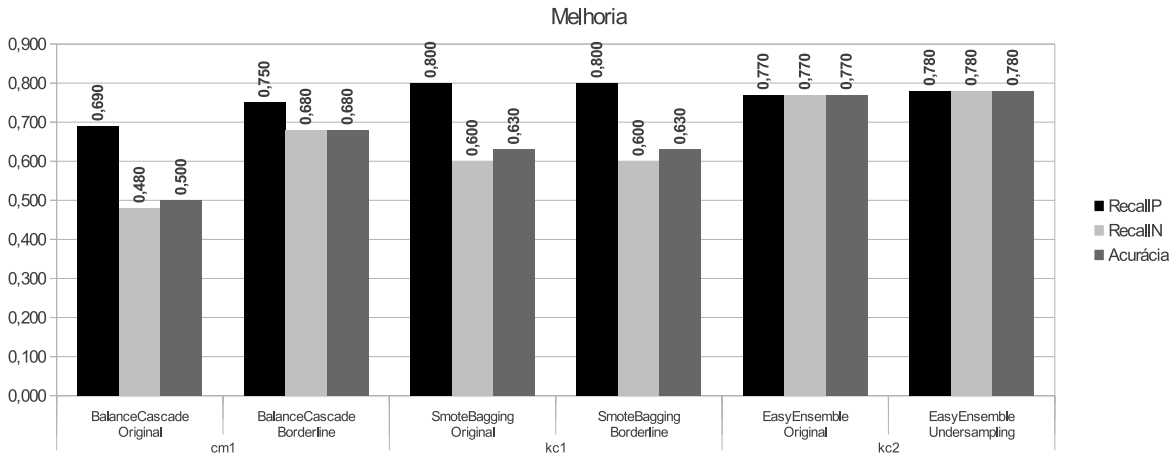

Figura 6.2: Base original e base com os métodos de balanceamento aplicados

Pode-se observar que a melhora não é muito significativa, mas é o suficiente para poder dizer que não existe um método melhor que o outro, mas que depende muito da base e do poder de generalização do classificador.

Tabela 6.5: Pré-processamento com os Métodos Híbridos

Base	Métodos	BalanceCascade			EasyEnsemble			SmoteBagging			SmoteBoost		
		RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia
cm1	Adasyn	0,730	0,600	61,00	0,890	0,460	50,00	0,830	0,330	38,00	0,040	0,990	90,00
	Borderline	0,750	0,680	68,00	0,790	0,590	61,00	0,530	0,720	70,00	0,280	0,870	81,00
	Oversampling	0,510	0,800	77,00	0,830	0,520	55,00	0,750	0,630	64,00	0,060	0,970	88,00
	Smote	0,260	0,880	82,00	0,770	0,630	65,00	0,830	0,520	55,00	0,750	0,630	65,00
	Undersampling	0,480	0,810	78,00	0,910	0,330	39,00	0,400	0,790	75,00	0,460	0,630	62,00
jm1	Adasyn	0,210	0,930	79,00	0,330	0,870	76,00	0,990	0,030	22,00	0,430	0,830	75,00
	Borderline	0,240	0,850	36,00	0,820	0,420	74,00	0,990	0,050	80,00	0,870	0,350	77,00
	Oversampling	0,430	0,740	49,00	0,780	0,430	71,00	0,990	0,040	81,00	0,850	0,350	76,00
	Smote	0,210	0,880	34,00	0,830	0,400	74,00	0,990	0,020	80,00	0,870	0,350	77,00
	Undersampling	0,120	0,940	28,00	0,530	0,700	57,00	0,940	0,190	80,00	0,650	0,620	64,00
kc1	Adasyn	0,880	0,660	70,00	0,900	0,660	70,00	0,930	0,440	52,00	0,070	0,980	84,00
	Borderline	0,670	0,780	76,00	0,730	0,690	70,00	0,800	0,600	63,00	0,680	0,760	75,00
	Oversampling	0,800	0,700	71,00	0,880	0,660	70,00	0,930	0,420	50,00	0,420	0,800	74,00
	Smote	0,750	0,720	72,00	0,780	0,600	63,00	0,920	0,500	56,00	0,710	0,730	73,00
	Undersampling	0,730	0,700	70,00	0,850	0,560	61,00	0,840	0,580	62,00	0,920	0,510	58,00
kc2	Adasyn	0,560	0,750	71,00	0,420	0,930	83,00	0,630	0,810	77,00	0,440	0,900	81,00
	Borderline	0,670	0,820	79,00	0,680	0,830	80,00	0,720	0,800	78,00	0,710	0,820	79,00
	Oversampling	0,670	0,840	80,00	0,750	0,810	80,00	0,760	0,770	77,00	0,720	0,820	80,00
	Smote	0,770	0,680	70,00	0,710	0,810	79,00	0,730	0,790	77,00	0,730	0,780	77,00
	Undersampling	0,730	0,770	76,00	0,780	0,780	78,00	0,770	0,780	78,00	0,650	0,860	82,00
pc1	Adasyn	0,330	0,960	92,00	0,460	0,930	90,00	0,880	0,690	71,00	0,480	0,930	90,00
	Borderline	0,180	0,970	92,00	0,320	0,960	92,00	0,530	0,850	83,00	0,360	0,960	92,00
	Oversampling	0,160	0,980	92,00	0,440	0,950	92,00	0,750	0,780	78,00	0,370	0,960	91,00
	Smote	0,370	0,950	91,00	0,510	0,930	90,00	0,900	0,580	60,00	0,460	0,930	90,00
	Undersampling	0,280	0,930	88,00	0,610	0,790	77,00	0,890	0,700	71,00	0,760	0,730	73,00

6.2 Classificação Base Engenharia de Software

As bases da Engenharia de Software utilizadas, foram bases com defeitos encontrados em três sistemas de Orientação a Aspectos de diferentes domínios de aplicação. O primeiro sistema é o IBATIS [25], que é um *framework* de código aberto, para o mapeamento de dados objeto-relacionais. A segunda aplicação é o *HealthWatcher* (HW) [39], um sistema de informações típico, baseado na web. HW foi lançado em 2002, com versões tanto em Java quanto em AspectJ. O terceiro sistema é uma linha de produtos de software para dispositivos móveis chamada *MobileMedia* (MM) [18]. MM foi desenvolvida originalmente em 2005 para permitir a manipulação de arquivos de imagem em dispositivos móveis pelos usuários.

Para a obtenção da base de dados foram coletadas e documentadas informações sobre defeitos em diversos módulos desses sistemas. O objetivo é predizer dadas as características de um módulo, sua probabilidade de conter defeitos. Os módulos são caracterizados através de métricas de acoplamento específicas para a programação Orientada a Aspectos. Mais informações a respeito dos sistemas e da metodologia de obtenção dos dados podem ser encontradas em [10].

As principais características das Bases para a realização dos experimentos estão relacionadas na Tabela 6.6.

Tabela 6.6: Caracteristicas das Bases

Base	**Instâncias**	**Atributos**	**Classe**		**%**
			N	**Y**	
HW	159	12	149	10	7%
Ibatis	285	12	262	23	9%
MM	70	12	64	6	9%

Pode-se observar o grande desequilíbrio entre as classes em todas as Bases.

A Tabela 6.7 mostra os resultados das bases que possuem defeitos encontrados em três sistemas de Orientação a Aspecto. Foram aplicados nestas bases os mesmos classificadores dito anteriormente, com o objetivo de observar o comportamento destes com bases do mundo real.

Pode-se observar na Tabela 6.7 que os resultados da base original com os classifi-

Tabela 6.7: Classificadores Tradicionais

Base	Classificador	RecallY	RecallN	Acurácia
HW	NaiveBayes	0,800	0,658	66,67
	SMO	0,000	0,993	93,08
	MultilayerP	0,500	0,980	94,97
	J48	0,300	0,980	93,71
	JRip	0,300	0,980	93,71
	BayesNet	0,100	0,980	92,45
Ibatis	NaiveBayes	0,130	0,931	86,67
	SMO	0,087	1,000	92,63
	MultilayerP	0,000	0,985	90,53
	J48	0,000	1,000	91,93
	JRip	0,000	0,985	90,53
	BayesNet	0,000	1,000	91,93
MM	NaiveBayes	0,833	0,703	71,43
	SMO	0,000	1,000	91,43
	MultilayerP	0,500	0,969	92,86
	J48	0,500	0,969	92,86
	JRip	0,500	0,984	94,29
	BayesNet	0,000	1,000	91,43

cadores tradicionais não são bons, principalmente para a classe minoritária. Nenhuma das bases consegue resultados razoáveis com o nível de balanceamento desequilibrado. O classificador NaiveBayes é o único que consegue resultados razoáveis para a classe minoritária, mas resultados ruins para a classe majoritária e também acurácia, isto acontece em todas as bases.

A Tabela 6.8 apresenta os resultados dos métodos de balanceamento aplicados e classificados pelos classificadores tradicionais.

A seguir são descritos detalhes dos resultados de cada base, para o experimento com os métodos de balanceamento com os classificadores tradicionais.

- **HW -** O método Borderline consegue melhorar a classe majoritária mantendo a classe minoritária utilizando o classificador NaiveBayes e o método Undersampling consegue manter a classe minoritária e melhorar a classe majoritária. Nos outros classificadores, todos os métodos conseguem melhores resultados na classe positiva, no entanto, diminuem a classe negativa. O método que se destaca nesta base na maioria dos classificadores é o Borderline, pois aumenta os valores da classe positiva e quase mantem os valores da classe negativa.

- **Ibatis -** Nesta base, nenhum dos métodos conseguem aumentar os valores de ambas

as classes. Pode-se observar que na maioria das vezes, os métodos que conseguem resultados bons na classe minoritária, diminui drasticamente os valores da classe majoritária. Um exemplo, é o classificador BayesNet com o método de balanceamento Undersampling, que consegue 80% de acerto na classe minoritária, mas um péssimo resultado para classe majoritária, se comparado com os resultados da base original que não conseguiu predizer nenhum exemplo da classe minoritária e 100% da classe majoritária.

- **MM -** O método que se destaca nesta base é o Undersampling classificado pelo NaiveBayes, pois obteve 100% de acerto para a classe minoritária e manteve o valor da classe majoritária. Na maioria dos classificadores este método consegue 100% de acerto na classe minoritária, mas diminui a outra classe.

Para facilitar a compreensão dos resultados obtidos, a Figura 6.3 apresenta um gráfico com o método e o classificador que obteve melhor resultado em cada base.

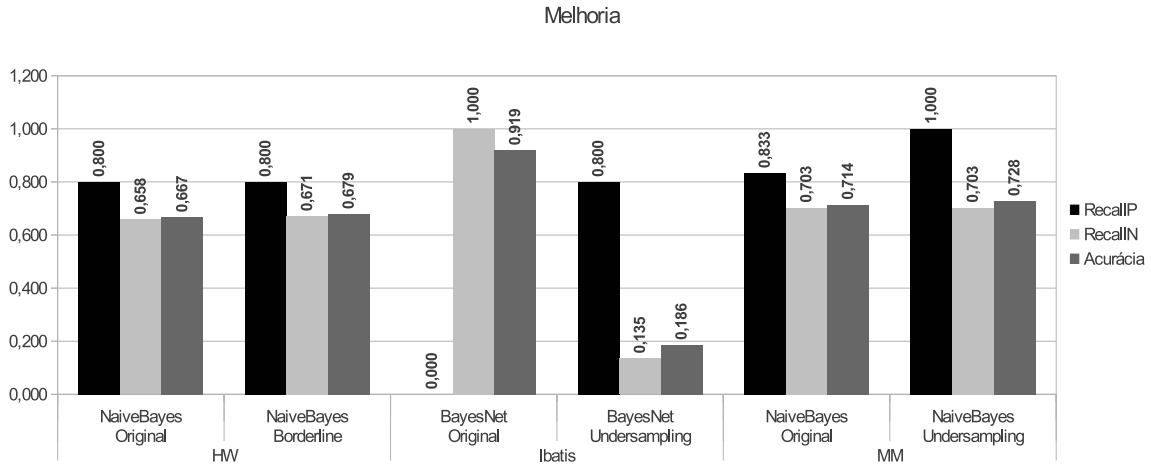

Figura 6.3: Base original e base com os métodos de balanceamento aplicados.

Tabela 6.8: Métodos de Pré-processamento com Classificadores Tradicionais

Base	Métodos	NaiveBayes			SMO			MultilayerP			J48			JRip			BayesNet		
		RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia
HW	Adasyn	0,900	0,644	66,030	1,000	0,624	64,760	0,900	0,879	88,020	0,700	0,920	90,560	0,700	0,919	90,560	0,600	0,906	88,630
	Borderline	0,800	0,671	67,900	0,800	0,853	84,960	0,700	0,940	92,460	0,600	0,960	93,710	0,500	0,960	93,060	0,500	0,932	90,500
	Oversampling	1,000	0,549	57,780	0,900	0,839	59,060	0,900	0,893	79,210	0,700	0,893	85,180	0,700	0,900	85,080	0,900	0,899	88,850
	Smote	0,800	0,651	74,540	0,900	0,732	72,190	0,800	0,920	87,560	0,900	0,926	71,260	0,600	0,940	86,110	0,800	0,912	90,330
	Undersampling	0,800	0,697	80,460	1,000	0,685	72,080	1,000	0,839	79,920	0,800	0,819	77,740	1,000	0,812	80,450	1,000	0,544	67,820
Ibatis	Adasyn	0,220	0,927	87,020	0,490	0,531	52,630	0,270	0,706	67,020	0,390	0,832	79,650	0,340	0,817	77,890	0,310	0,840	79,650
	Borderline	0,480	0,550	54,390	0,690	0,336	36,490	0,530	0,642	63,160	0,100	0,943	87,370	0,090	0,954	88,420	0,100	0,943	87,370
	Oversampling	0,170	0,927	86,670	0,390	0,595	57,890	0,540	0,573	56,840	0,210	0,828	77,890	0,400	0,889	84,910	0,270	0,771	72,980
	Smote	0,220	0,916	85,960	0,390	0,549	53,680	0,650	0,519	52,980	0,170	0,870	81,400	0,350	0,802	76,490	0,390	0,709	68,420
	Undersampling	0,310	0,765	72,630	0,880	0,188	24,210	0,830	0,412	44,560	0,420	0,588	57,190	0,360	0,679	65,260	0,800	0,135	18,600
MM	Adasyn	1,000	0,641	67,140	0,800	0,718	72,860	0,800	0,737	74,290	0,800	0,829	82,860	0,600	0,876	85,710	0,600	0,877	85,710
	Borderline	1,000	0,656	68,570	0,600	0,923	90,000	0,600	0,860	84,290	0,600	0,938	91,430	0,600	0,922	90,000	0,600	0,908	88,570
	Oversampling	1,000	0,624	65,710	0,600	0,826	81,430	0,600	0,797	78,570	0,600	0,877	85,710	0,600	0,906	88,570	0,600	0,877	85,710
	Smote	1,000	0,656	68,570	0,800	0,846	84,290	0,600	0,778	77,140	0,600	0,954	92,860	0,600	0,923	90,000	0,600	0,938	91,430
	Undersampling	1,000	0,703	72,860	0,800	0,624	64,290	1,000	0,747	77,140	1,000	0,578	61,430	1,000	0,594	62,860	1,000	0,462	51,430

A Tabela 6.9 mostra os resultados das bases originais com os métodos híbridos.

Tabela 6.9: Métodos Híbridos

Base	Híbridos	RecallP	RecallN	AUC	Acurácia
HW	BalanceCascade	0,900	0,790	0,849	80,00
	EasyEnsemble	1,00	0,770	0,886	78,00
	SMOTEBagging	0,700	0,890	0,800	88,00
	SMOTEBoost	0,400	0,970	0,687	93,00
Ibatis	BalanceCascade	0,470	0,510	0,499	51,00
	EasyEnsemble	0,470	0,570	0,532	56,00
	SMOTEBagging	0,170	0,850	0,517	79,00
	SMOTEBoost	0,260	0,900	0,587	85,00
MM	BalanceCascade	1,00	0,600	0,804	64,00
	EasyEnsemble	0,830	0,480	0,643	51,00
	SMOTEBagging	0,660	0,820	0,715	81,00
	SMOTEBoost	0,660	0,950	0,777	92,00

Para cada base um método se comporta de uma maneira diferente, por exemplo o método EasyEnsemble na base HW, consegue 100% de acerto na classe minoritária, mas se observar a base Ibatis, este método não consegue chegar aos 50% de acerto. O mesmo método na base MM, tem um valor razoável para a classe minoritária, mas um valor ruim para a classe majoritária.

A Tabela 6.10 apresenta os resultados com os métodos de balanceamento classificados pelos métodos híbridos. Analisando esta tabela com a Tabela 6.9, pode-se observar que melhoras acontecem com alguns métodos. A seguir estão descritos detalhes dos resultados de cada base.

- **HW -** O método híbrido SmoteBagging consegue melhorar os resultados das duas classes com o método de balanceamento Oversampling, o método Adasyn consegue melhor a classe positiva e manter a classe negativa. Já para o SmoteBoost, o método que consegue melhor resultado da classe positiva mantendo a classe negativa é o Smote. Os métodos híbridos BalanceCascade e EasyEnsemble não consegue melhoras em nenhum dos métodos de balanceamento.

- **Ibatis -** Os classificadores que conseguem melhores resultados apenas para a classe positiva são: SmoteBagging e SmoteBoost utilizando os métodos de balanceamento Oversampling, Smote e Undersampling.

- **MM -** O classificador EasyEnsemble melhora apenas os resultados da classe majoritária mantendo a classe minoritária com o método Borderline. Para o classificador SmoteBagging os métodos que aumentam ambas as classes são Adasyn e Borderline. Os métodos de balanceamento que mantem a classe positiva mantendo a classe negativa são Oversampling e Smote.

A Figura 6.4 mostra um gráfico com os métodos e classificadores que conseguiram melhores resultados para cada base.

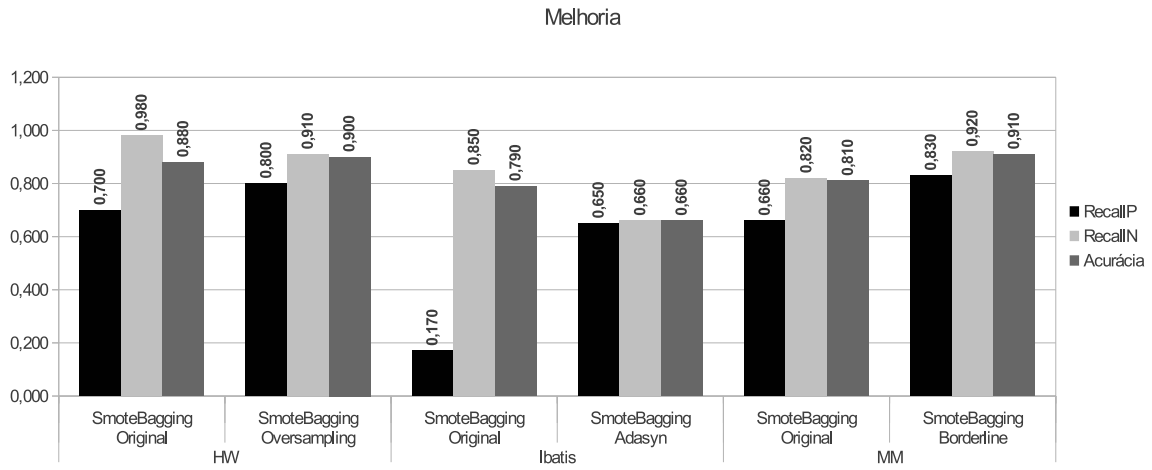

Figura 6.4: Base original e base com os métodos de balanceamento aplicados

É notável que para esse conjunto de bases os resultados são mais interessantes. Os métodos que pode ser destacado é o Undersampling e Borderline, pois conseguem resultados razoáveis para a classe minoritária.

Tabela 6.10: Pré-processamento com os Métodos Híbridos

Base	Métodos	BalanceCascade			EasyEnsemble			SmoteBagging			SmoteBoost		
		RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia	RecallP	RecallN	Acurácia
HW	Adasyn	0,600	0,930	91,00	0,600	0,900	88,00	0,800	0,890	89,00	0,600	0,940	92,00
	Borderline	0,500	0,930	91,00	0,500	0,920	89,00	0,600	0,930	91,00	0,600	0,960	94,00
	Oversampling	0,700	0,930	91,00	0,800	0,910	91,00	0,800	0,910	90,00	0,800	0,930	92,00
	Smote	0,400	0,970	94,00	0,400	0,950	91,00	0,900	0,870	87,00	0,600	0,970	95,00
	Undersampling	0,400	0,950	91,00	0,800	0,700	71,00	1,000	0,440	48,00	0,800	0,690	70,00
Ibatis	Adasyn	0,210	0,900	85,00	0,300	0,890	84,00	0,650	0,660	66,00	0,340	0,810	77,00
	Borderline	0,080	0,960	89,00	0,130	0,950	88,00	0,080	0,920	85,00	0,130	0,940	87,00
	Oversampling	0,210	0,910	85,00	0,260	0,880	83,00	0,300	0,700	67,00	0,300	0,760	72,00
	Smote	0,170	0,920	86,00	0,170	0,910	85,00	0,470	0,690	68,00	0,260	0,810	76,00
	Undersampling	0,130	0,760	71,00	0,210	0,660	63,00	0,430	0,510	50,00	0,560	0,460	47,00
MM	Adasyn	0,660	0,950	92,00	0,660	0,840	82,00	0,830	0,840	84,00	0,660	0,930	91,00
	Borderline	0,830	0,950	94,00	0,830	0,930	92,00	0,830	0,920	91,00	0,830	0,920	91,00
	Oversampling	0,660	0,950	92,00	0,660	0,950	92,00	0,660	0,920	90,00	0,660	0,950	92,00
	Smote	0,660	0,930	91,00	0,660	0,930	91,00	0,660	0,850	84,00	0,660	0,950	92,00
	Undersampling	0,830	0,650	67,00	0,830	0,590	61,00	1,000	0,340	40,00	0,830	0,530	55,00

CAPÍTULO 7

CONCLUSÃO

Neste trabalho foram apresentados métodos propostos na literatura para tratar o problema de classificação em bases desbalanceadas. Estes métodos foram aplicados em treze (13) bases de dados provenientes do UCI *Machine Learning Repository* e em 2 conjuntos de bases da Engenharia de Software, bases essas que foram construídas com defeitos encontrados em sistemas Orientados a Aspectos. O primeiro conjuntos são 5 bases do repositório NASA Metrics Data Project (cm1, jm1, kc1, kc2 e pc1) e o segundo conjunto são 3 sistemas de orientação a Aspectos (HW, Ibatis e MM).

Primeiramente, foram efetuados os experimentos aplicando os 5 classificadores implementados no software Weka que foram chamados de classificadores Tradicionais (NaiveBayes, SMO, MultilayerP, J48, JRip e BayesNet). Com o intuito de observar o comportamento dos métodos de balanceamento foram aplicados na base original os métodos de balanceamento na fase de pré-processamento: Adasyn, Borderline, Oversampling, Smote e Undersampling e novamente classificados pelos algoritmos tradicionais. Com o mesmo objetivo os métodos híbridos *BalanceCascade, EasyEnsemble, SMOTEBagging e SMOTEBoost* foram aplicados na base original e depois aplicados com a base já balanceada com o métodos de pré-processamento.

Para os 3 conjuntos de bases, cada método apresenta resultados diferentes para cada classificador. De acordo com os resultados obtidos nos experimentos deste trabalho é possível responder as questões realizadas ao assunto, que são:

- A Classificação pode ser melhorada aplicando algum método de balanceamento?

  A melhora da classificação é possível, mas para tal, é preciso da combinação correta do método aplicado com o classificador. Observar utilizando uma métrica de avaliação que realmente indique a melhoria nos valores da classe de interesse (classe minoritária). Para isso, vários testes deveriam ser realizados para descobrir o método

que resulta melhores valores para uma base específica.

- Existe um método que sempre se comporta bem?

  Com os resultados obtidos é possível dizer que ainda não existe um método que resolva todos os problemas da classificação desbalanceada, pois como pode ser visto, os métodos possuem comportamentos diferentes, dependendo da base e do classificador. Geralmente, as bases utilizados para testar a eficácia dos métodos são bases provenientes do UCI *machine learning repository*, bases que são pré-processadas e que não possuem os problemas como ruídos, que normalmente são mais frequentes em bases do mundo real.

- Os classificadores podem influenciar nos resultados?

  Cada classificador é capaz de construir um modelo a partir do conjunto de dados de treinamento, por este motivo, o classificador pode sim influenciar nos resultados, como mostra os resultados das bases provenientes do UCI *machine learning repository*. Ou seja, a mesma base com o mesmo método de balanceamento aplicado, utilizando classificadores diferentes possuem resultados diferentes. Pode-se concluir, que é de suma importância a utilização de vários classificadores para observar melhor o comportamento dos métodos aplicados.

- Quais são as métricas de avaliação que realmente apontam a melhora da classe minoritária?

  Um dos fatores importantes observado neste trabalho é o tipo de métrica de avaliação utilizada como critério de avaliação, pois a maioria dos trabalhos relacionados citados neste trabalho utilizam a acurácia e o F-Measure como o critério de avaliação, o que não significa que essas métricas mostram a melhora na classe minoritária, pois na maioria das vezes a melhora acontece apenas na classe majoritária, mascarando o verdadeiro objetivo dos métodos de balanceamento. Desta forma, é muito importante observar os resultados obtidos entre as classes separadamente. Uma maneira interessante é utilizar a métrica de avaliação Recall, fazendo os cálculos dos verdadeiros positivos e dos verdadeiros negativos.

BIBLIOGRAFIA

[1] J. Alcalá, A. Fernández, J. Derrac J. Luengo, S. García, L. Sánchez, e F. Herrera. Keel Data-Mining Software Tool: Data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-*, 17:255–287, 2010.

[2] Vanessa Terezinha Ales. O Algoritmo Sequential Minimal Optimisation para Resolução do problema de Support Vector Machine: Uma Técnica para Reconhecimento de Padrões. Dissertação de mestrado, Universidade Federal do Paraná, 2008.

[3] Gustavo E. A. P. Alves Batista. *Pré-processamento de Dados em Aprendizado de Máquina Supervisionado*. Tese de doutorado, USP - São Carlos, 2003.

[4] Marcelo Beckmann. Algoritmos Genéticos como estratégia de Pré-Processamento em conjunto de dados Desbalanceados. Dissertação de mestrado, Universidade Federal do Rio de Janeiro, 2010.

[5] Marcelo Beckmann, Beatriz Souza L.P. de Lima, e Nelson F.F. Ebecken. Genetic algorithms as a pre processing strategy for imbalanced datasets. *Proceedings of the 13th annual conference companion on Genetic and evolutionary computation*, GECCO '11, páginas 131–132, New York, NY, USA, 2011. ACM.

[6] Valnaide Gomes Bittencourt. Aplicação de Técnicas de Aprendizado de Máquina no Reconhecimento de Classes Estruturais de Proteínas. Dissertação de mestrado, Universidade Federal do Rio Grande do Norte, 2005.

[7] André Pinz Borges. *Descoberta de regras de condução de trens de carga*. Dissertação de mestrado, Universidade Católica do Paraná, 2009.

[8] Remco R. Bouckaert. Bayesian network classifiers in weka for version 3-5-7. *Artificial Intelligence Tools*, 2008.

[9] KW Bowyer, NV Chawla, L.O. Hall, e W.P. Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *Arxiv preprint arXiv:1106.1813*, 16:321–357, 2011.

[10] Rachel Burrows, Fabiano C. Ferrari, Otavio A.L. Lemos, Alessandro Garcia, e Francois Taiani. The impact of coupling on the fault-proneness of aspect-oriented programs: An empirical study. *International Symposium on Software Reliability Engineering*, 0:329–338, 2010.

[11] C.O. Camilo e J.C. da Silva. Mineração de Dados: Conceitos, Tarefas, Métodos e Ferramentas. *inf.ufg.br*, 2009.

[12] L.G. Castanheira. Aplicação de Técnicas de Mineração de Dados em Problemas de Classificação de Padrões. Dissertação de mestrado, Universidade Federal de Minas Gerais, 2008.

[13] N Chawla e Aleksandar Lazarevic. SMOTEBoost: Improving prediction of the minority class in boosting. *Knowledge Discovery*, páginas 107–119, 2003.

[14] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, e W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[15] Dennis J. Drown e Taghi Khoshgoftaar. Evolutionary sampling and software quality modeling of high-assurance systems. *Systems*, setembro de 2009.

[16] Katti Faceli, Ana Carolina Lorena, João Gama, e André C. P. L. Ferreira Carvalho. *Inteligência Artificial: Uma Abordagem de Aprendizagem de Máquina*. LTC, 2011.

[17] Usama Fayyad e G Piatetsky-Shapiro. From data mining to knowledge discovery in databases. *AI magazine*, páginas 37–54, 1996.

[18] Eduardo Figueiredo, Nelio Cacho, Claudio Sant'Anna, Mario Monteiro, Uira Kulesza, Alessandro Garcia, Sérgio Soares, Fabiano Ferrari, Safoora Khan, Fernando Castor Filho, e Francisco Dantas. Evolving software product lines with aspects: an

empirical study on design stability. *Proceedings of the 30th International Conference on Software Engineering*, ICSE '08, páginas 261–270. ACM, 2008.

[19] Mikel Galar, A. Fernández, Edurne Barrenechea, Humberto Bustince, e Francisco Herrera. A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches. *sci2s.ugr.es*, páginas 1–22, 2011.

[20] Salvador García e Francisco Herrera. Evolutionary undersampling for classification with imbalanced datasets: proposals and taxonomy. *Evolutionary computation*, 17(3):275–306, janeiro de 2009.

[21] Hui Han. Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. *Advances in Intelligent Computing*, páginas 878–887, 2005.

[22] Haibo He. Learning from imbalanced data. *and Data Engineering, IEEE Transactions*, 21(9):1263–1284, 2009.

[23] Haibo He, Yang Bai, Edwardo A Garcia, e Shutao Li. ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning. *International Joint Conference on Neural Networks*, (3):1322–1328, 2008.

[24] Jason Van Hulse e TM Khoshgoftaar. Experimental perspectives on learning from imbalanced data. *on Machine learning*, 2007.

[25] iBATIS. http://ibatis.apache.org/, 2010.

[26] Nathalie Japkowicz. Concept-learning in the presence of between-class and within-class imbalances. *Advances in Artificial Intelligence*, páginas 67–77, 2001.

[27] Nathalie Japkowicz. Class Imbalances: Are we Focusing on the Right Issue. *Workshop on Learning from Imbalanced Data Sets II*, 2003.

[28] George John e Pat Langley. Estimating continuous distributions in bayesian classifiers. *In Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, páginas 338–345. Morgan Kaufmann, 1995.

[29] Xu-Ying Liu, Jianxin Wu, e Zhi-Hua Zhou. Exploratory undersampling for class-imbalance learning. *IEEE transactions on systems, man, and cybernetics.*, 39(2):539–50, abril de 2009.

[30] Ana Carolina Lorena e André C. P. L. Ferreira Carvalho. Idendificação de ruídos em dados de bioinformática. 2007.

[31] E.L. Machado. Um estudo de limpeza em base de dados desbalanceada e com sobreposição de classes. Dissertação de mestrado, Universidade de Brasília, 2009.

[32] Jairo Lucas D E Moraes. Controle de acesso baseado em biometria facial. 2010.

[33] R.B. Pereira. Seleção lazy de atributos para a tarefa de classificação. Dissertação de mestrado, Universidade Federal Fluminense, 2009.

[34] Ronaldo C. Prati, Gustavo Batista, e Maria C. Monard. Uma experiência no Balanceamento Artificial de Conjuntos de Dados para Aprendizado com Classes Desbalanceadas utilizando Análise ROC. *In Proceedings of IV Workshop on Advances Trends in AI for Problem Solving,*, (2):1–6, 2003.

[35] Ronaldo C Prati, Gustavo E A P A Batista, e Maria C Monard. Class imbalances versus class overlapping: an analysis of a learning system behavior. *MICAI 2004: Advances in Artificial.*

[36] J.P. Rodrigues, R.B. Prudêncio, e F.A. Barros. B-Boost: Uma Extensão do Método de Boosting para Conjuntos de Treinamento Desbalanceados. *lbd.dcc.ufmg.br*, páginas 1039–1048, 2004.

[37] D.M. Santoro. Sobre o Processo de Seleção de Subconjuntos de Atributos - As Abordagens Filtro e Wrapper. Dissertação de mestrado, Universidade Federal de São Carlos, 2005.

[38] Matheus Victor Brum Soares. *Aprendizado de máquina parcialmente supervisionado multidescrição para realimentação de relevância em recuperação de informação na WEB.* Dissertação de mestrado, USP, São Carlos, 2009.

[39] Sergio Soares, Eduardo Laureano, e Paulo Borba. Implementing distribution and persistence aspects with aspectj. *SIGPLAN Not.*, 37:174–190, November de 2002.

[40] GM Weiss. Mining with rarity: a unifying framework. *Sigkdd Explorations*, 6(1):7–19, 2004.

[41] Ian H. Witten e Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Series in Data Management Sys. Morgan Kaufmann, second edition, June de 2005.

[42] Ian H. Witten e Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

DAIANY FRANCISCA LARA

ESTUDOS EMPÍRICOS DOS MÉTODOS DE BALANCEAMENTO PARA A CLASSIFICAÇÃO

CURITIBA

2013