UNIVERSIDADE FEDERAL DE ALFENAS

MARCELO ANTONIO MENDES BASTOS

UMA TÉCNICA BASEADA EM PROGRAMAÇÃO INTEIRA MISTA PARA SELEÇÃO DE CLASSIFICACORES EM *ENSEMBLES*

Alfenas/MG

2018

MARCELO ANTONIO MENDES BASTOS

UMA TÉCNICA BASEADA EM PROGRAMAÇÃO INTEIRA MISTA PARA SELEÇÃO DE CLASSIFICACORES EM *ENSEMBLES*

Monografia apresentada ao Curso de Bacharelado em Ciência da Computação da Universidade Federal de Alfenas como requisito parcial para obtenção do Título de Bacharel em Ciência da Computação.

Orientador: Prof. Humberto César Brandão de Oliveira

Alfenas, 23 de janeiro de 2018.

MARCELO ANTONIO MENDES BASTOS

UMA TÉCNICA BASEADA EM PROGRAMAÇÃO INTEIRA MISTA PARA SELEÇÃO DE CLASSIFICACORES EM *ENSEMBLES*

A Banca examinadora abaixo-assinada aprova a monografia apresentada como parte dos requisitos para obtenção do título de Bacharel em Ciência da Computação pela Universidade Federal de Alfenas.

Prof. Douglas Donizeti de Castilho Braz
Instituto Federal de Educação, Ciência e Tecnologia do Sul de Minas Gerais

---

Prof. Luiz Eduardo da Silva
Universidade Federal de Alfenas

Prof. Humberto César Brandão de Oliveira (Orientador)
Universidade Federal de Alfenas

Alfenas, 23 de janeiro de 2018.

Dedico este trabalho a Deus, aos meus pais, a minha irmã, e a todos os outros que me apoiaram na realização deste curso de graduação, incluindo meus primos Rafael e Guilherme.

AGRADECIMENTOS

Agradeço primeiramente a Deus, por sempre estar me abençoando e me dando forças em todos os momentos desta caminhada.

Aos meus queridos pais, Antônio e Zelma, e minha irmã Natália, pelo amor e cuidado transmitido, por sempre me apoiarem e incentivarem em todas os momentos difíceis desta jornada.

Aos meus primos Rafael e Guilherme, por me inspirarem a seguir um curso de graduação e a sempre dar o melhor de mim nas horas de dificuldade.

Agradeço ao meu orientador Humberto por ter me apresentado a área de inteligência artificial, e influenciado a participar de inúmeras competições onde pude me interessar mais por essa área e aumentar meu conhecimento. Agradeço também pelos conhecimentos compartilhados, pelas oportunidades e por todo o apoio dado nesta jornada.

A todos aqueles que de alguma forma ajudaram na realização deste trabalho, muito obrigado!

RESUMO

Este trabalho apresenta uma nova técnica para seleção de classificadores em *ensembles* de classificação, que foi baseada nas etapas de um Sistema de Múltiplos Classificadores. Como método de seleção, foi proposto um modelo de Programação Inteira Mista, integrando-se assim as áreas de inteligência artificial e otimização. Para a realização da seleção, o modelo proposto considera também custos para erros e acertos na classificação de padrões, pois como será discutido determinados problemas de classificação podem demandar que diferentes erros ou acertos cometidos tenham diferentes importâncias. Foram propostos experimentos em bases de dados de diferentes áreas do conhecimento, com o propósito de se conseguir melhorar o desempenho dos *ensembles* criados para essas bases usando-se um subconjunto de classificadores selecionados pela técnica. Através dos resultados dos experimentos foi concluído que a técnica se mostrou adequada ao seu propósito, conseguindo melhorar o desempenho de *ensembles* na maior parte das bases de dados testadas.

Palavras-chave: *ensembles*, Sistema de Múltiplos Classificadores, método de seleção, Programação Inteira Mista, inteligência artificial, otimização, custos de classificação.

ABSTRACT

This work presents a new technique for selecting classifiers in classification ensembles, based in the stages of a Multiple Classifiers System. As a selection method, a Mixed-Integer Programming model was proposed, thus integrating fields of artificial intelligence and optimization. In order to perform the selection, the proposed model also considers costs for mistakes and successes in the classification of patterns, because as it will presented certain classification problems can demand that different mistakes or successes have different importance. Experiments were proposed in datasets of different areas of knowledge, in order to improve the performance of the ensembles for those bases using a subset of classifiers selected by the technique. Through the results of the experiments, it was concluded that the technique proved to be suitable for its purpose, improving the performance of ensembles in most of the tested datasets.

Key-words: ensembles, Multiple Classifier System, selection method, Mixed-Integer Programming, artificial intelligence, optimization, classification costs.

LISTA DE FIGURAS

Figura 1 – Etapas de um MCS 16
Figura 2 – Funcionamento da etapa de geração 17
Figura 3 – Matriz de confusão para classificação binária 23
Figura 4 – Exemplo de gráfico ROC para 7 classificadores 25
Figura 5 – Exemplificação da classe positiva 33
Figura 6 - Exemplificação da classe negativa 33
Figura 7 - Fluxo de Experimentação 38
Figura 8 – Gráfico de resultados da métrica AUC nos experimentos na base de dados de retinopatia diabética. 43
Figura 9 – Frequência de seleções de cada classificador na base de dados de retinopatia diabética 44
Figura 10 – Tempo de execução do CPLEX na base de dados de retinopatia diabética para cada *fold*. 45
Figura 11 – Gráfico de resultados da métrica AUC nos experimentos na base de dados de biodegradação de molécula. 46
Figura 12 – Frequência de seleções de cada classificador na base de dados de biodegradação de molécula. 47
Figura 13 – Tempo de execução do CPLEX na base de dados de biodegradação de molécula para cada *fold*. 48
Figura 14 – Gráfico de resultados da métrica AUC nos experimentos na base de dados de *spam*. 49
Figura 15 – Frequência de seleções de cada classificador na base de dados de *spam*. 50
Figura 16 – Tempo de execução do CPLEX na base de dados de *spam* para cada *fold*. 51
Figura 17 – Gráfico de resultados da métrica AUC nos experimentos na base de dados de câncer cervical 53
Figura 18 – Frequência de seleções de cada classificador na base de dados de câncer cervical. 54
Figura 19 – Tempo de execução do CPLEX na base de dados de câncer cervical para cada *fold*. 55
Figura 20 – Gráfico de resultados da métrica AUC nos experimentos na base de dados de solavancos sísmicos. 57

Figura 21 – Tempo de execução do CPLEX na base de dados de solavancos sísmicos para cada *fold*. 58

LISTA DE TABELAS

Tabela 1 - Distribuição de classes na base de dados de retinopatia diabética 35
Tabela 2 - Distribuição de classes na base de dados de biodegradação de molécula 35
Tabela 3- Distribuição de classes na base de dados de spam 36
Tabela 4- Distribuição de classes na base de dados de Câncer Cervical 37
Tabela 5 - Distribuição de classes na base de dados de Solavancos Sísmicos 37
Tabela 6 - Variação dos parâmetros cvp, cfp, cvn e cfn. 39
Tabela 7 - Variação dos parâmetros cvp, cvp, cvn e cfn adicionadas nos experimentos com as bases de dados 4 e 5. 39
Tabela 8 – Resultados da métrica AUC nos experimentos na base de dados de Retinopatia Diabética 42
Tabela 9 – Resultados da métrica AUC nos experimentos na base de dados de biodegradação de molécula. 46
Tabela 10 – Resultados da métrica AUC nos experimentos na base de dados de *spam*. 49
Tabela 11 – Resultados da métrica AUC nos experimentos na base de dados de câncer cervical. 52
Tabela 12 – Resultados da métrica AUC nos experimentos na base de dados de solavancos sísmicos. 56
Tabela 13 – Fatores de influência na escolha dos custos 59
Tabela 14 – Fatores de influência no tempo de execução médio 60
Tabela 15 – Influência dos parâmetros cvp e cvn no experimento de maior tempo de execução. 61

LISTA DE ABREVIAÇÕES

MCS	*Multiple Classifier System*
Bagging	*Boostrap Agregating*
XGBoost	*Extreme Gradient Boosting Tree*
SVM	*Support Vector Machine*
ROC	*Receiver Operating Characteristics*
AUC	*Area Under Curve*
PO	Pesquisa Operacional
MIP	*Mixed-Integer Programing*
UCI	*University of California – Irvine*
QSAR	*Quantitative Structure−Activity Relationship*

SUMÁRIO

1 Introdução 13

1.1 Justificativa e motivação 13

1.2 Problematização 13

1.3 Objetivos 14

1.3.1 Gerais 14

1.3.2 Específicos 14

1.4 Organização da monografia 14

2 Sistema de Múltiplos Classificadores 16

2.1 Geração de Classificadores 16

2.2 Seleção de Classificadores 18

2.3 Agregação de Classificadores 18

3 Referencial Teórico 20

3.1 Aprendizagem de máquina 20

3.1.1 Introdução 20

3.1.2 Técnicas baseadas em *ensemble* 21

3.2 Validação 22

3.2.1 Validação Cruzada 22

3.2.2 Matriz de Confusão 23

3.2.3 Análise ROC 24

3.3 Pesquisa operacional 26

3.3.1 Construção do modelo 27

3.3.2 Implementação 28

4 Materiais e métodos 29

4.1 Geração de classificadores 29

4.2 Modelo de Programação Inteira Mista para seleção de classificadores 30

4.3 Agregação de classificadores 34
4.4 Base de dados 34
4.4.1 Retinopatia Diabética 34
4.4.2 Biodegradação de molécula 35
4.4.3 *Spam* 36
4.4.4 Câncer Cervical 36
4.4.5 Solavancos Sísmicos 37
4.5 Experimentação 37
4.6 Detalhes de implementação 40
5 Resultados e discussões 42
5.1 Análise das bases de dados 42
5.1.1 Retinopatia diabética 42
5.1.2 Biodegradação de molécula 45
5.1.3 *Spam* 48
5.1.4 Câncer cervical 51
5.1.5 Solavancos sísmicos 55
5.2 Análise geral 58
5.2.1 Influência dos custos no desempenho dos experimentos 58
5.2.2 Tempo de execução 60
6 Conclusões e trabalhos futuros 63
7 Referências 65

1 Introdução

1.1 Justificativa e motivação

Nas últimas décadas, pesquisadores tem dado cada vez maior importância na pesquisa de métodos de classificação aplicados em diferentes campos da atividade humana, onde se chegou em uma conclusão de que o uso de um classificador único já não é mais adequeado para se conseguir uma maior eficiência em problemas de classificação de padrões, sendo necessária a combinação de um conjunto destes para se conseguir classificações de maior qualidade, criando-se o que é chamado de *ensemble* de classificadores (KO; SABOURIN; BRITTO, 2008) (BRITTO; SABOURIN; OLIVEIRA, 2014).

Um Sistema de Múltiplos Classificadores (também referenciado pela sigla MCS – *Multiple Classifier System*) consiste, dentre outras tarefas, na seleção de classificadores que serão usados no *ensemble*, tentando melhorar a precisão na classificação de padrões (CRUZ; SABOURIN; CAVALCANTI, 2017) (CRUZ; SABOURIN; CAVALCANTI, 2018).

O estudo de MCS é uma área de pesquisa ativa em aprendizado de máquina e reconhecimento de padrões. De acordo com (CRUZ; SABOURIN; CAVALCANTI, 2018), este é um sistema amplamente usado na resolução de muitos problemas do mundo real, como: Reconhecimento facial (BASHBAGHI *et al*., 2017a), classificação de gênero musical (LISBOA DE ALMEIDA *et al*., 2012), análise de crédito (LESSMANN *et al*., 2015) (XIAO; XIAO; WANG, 2016), desbalanceamento de classes (GALAR *et al*., 2012), sistema de recomendação (PORCEL *et al*., 2012) (JAHRER; TÖSCHER; LEGENSTEIN, 2010), previsão de bug em software (DI NUCCI *et al*., 2017) (PANICHELLA; OLIVETO; DE LUCIA, 2014), detecção de intrusões (GIACINTO; ROLI; DIDACI, 2003) (GIACINTO *et al*., 2008) e para lidar com ambientes em mudança (KRAWCZYK *et al*., 2017) (KUNCHEVA, 2004) (POLIKAR *et al*., 2001).

A possibilidade de aplicação desse método em diferentes problemas enfatiza a importância de desenvolvimento de novas abordagens para melhorar a qualidade na combinação de classificadores. Dessa maneira, este trabalho propõe uma técnica usando Programação Inteira Mista (do inglês *Mixed-Integer Programming* - MIP), para fazer seleção de classificadores em um MCS.

1.2 Problematização

Após a geração de classificadores para a composição em um *ensemble*, existe a possibilidade de, dentre os classificadores gerados, existirem classificadores que caso combinados ao *ensemble*, piorem seu resultado final. Classificadores que tenham baixo

desempenho ou pouca diversidade em relação aos outros são exemplos de classificadores que podem contribuir para que haja uma piora no resultado final. Dessa maneira, a seleção de classificadores pode ser uma abordagem indispensável na tentativa de solução desse problema.

Um outro fator a ser levado em consideração, é que durante uma tarefa de classificação, existem erros que podem ser mais prejudiciais do que outros. Por exemplo, em um problema de classificação de tumores em benigno ou maligno, um tumor ser classificado como benigno mas na verdade ser maligno é mais prejudicial do que ele ser benigno e ser classificado como maligno. A primeira possibilidade representaria um risco de vida a um paciente, enquanto a segunda poderia representar um risco financeiro. Sendo assim, cada uma delas pode ser ponderada por um custo diferente.

A técnica proposta neste trabalho leva isso em consideração, tentando maximizar o acerto com base em custos definidos para as possibilidades de acerto e de erro em uma classificação binária.

Dessa maneira, dado um conjunto inicial de classificadores e custos de acertos e erros em um problema classificação, surge a seguinte questão: Uma técnica baseada em Programação Inteira Mista que considere os custos em questão para a seleção de um subconjunto de classificadores poderia criar um *ensemble* com essa seleção que tenha melhor desempenho do que um *ensemble* composto por todos os classificadores do conjunto inicial?

1.3 Objetivos

1.3.1 Gerais

Este trabalho possui como objetivo geral disponibilizar às pessoas que trabalham na área uma nova técnica de qualidade para criação de *ensembles* de classificação.

1.3.2 Específicos

Os objetivos específicos deste trabalho são:

- Promover a integração das áreas de otimização e aprendizado de máquina.
- Fornecer uma técnica que seja capaz de se adequar a particularidade de diferentes acertos e erros em problemas de classificação.

1.4 Organização da monografia

Esta monografia está organizada da seguinte forma: No Capítulo 2, será definido as etapas para a criação de um Sistema de Múltiplos Classificadores. No Capítulo 3, está o referencial teórico, subdividido em três partes: Apresentação de conceitos e técnicas de aprendizado de máquina, validação de classificadores e definição de conceitos sobre Pesquisa Operacional. O

método proposto será apresentado no Capítulo 4. No Capítulo 5, serão apresentados os resultados deste trabalho e no Capítulo 6, as conclusões e sugestões de trabalhos futuros.

2 Sistema de Múltiplos Classificadores

Um Sistema de Múltiplos Classificadores é composto de três etapas (BRITTO; SABOURIN; OLIVEIRA, 2014): Geração, Seleção e Agregação, representado pela Figura 1. Na primeira etapa, é gerada um conjunto de classificadores; na segunda, é feita a seleção de um ou de um subconjunto desses classificadores; Já na última etapa, é feita a decisão final com o(s) classificador(es) selecionado(s).

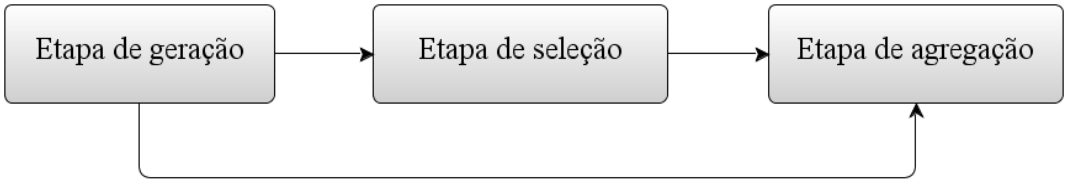

Figura 1 – Etapas de um MCS

Como pode ser observado na Figura 1, a etapa de seleção é opcional, podendo a agregação ser feita com todos os classificadores gerados. Nas seções seguintes será detalhado cada uma das etapas.

2.1 Geração de Classificadores

Como já descrito anteriormente, nesta etapa é feita a geração de um conjunto de classificadores, chamados de classificadores base. Segundo (CRUZ; SABOURIN; CAVALCANTI, 2018), o objetivo é que os classificadores base criados sejam diversos e precisos. A diversidade é um fator importante, pois não há motivos para combiná-los caso eles sempre apresentem o mesmo resultado.

Para a geração de classificadores diversos, existem 6 principais estratégias para a geração de classificadores (DUIN, 2002) (CRUZ; SABOURIN; CAVALCANTI, 2018):

1. Diferentes inicializações: Se o treinamento do classificador for dependente de inicialização, diferentes inicializações podem implicar em classificadores diferentes. Isso inclui Redes Neurais, por exemplo.
2. Diferentes escolhas de parâmetros, como o kernel em uma *Support Vector Machine* ou a quantidade de poda em uma árvore de decisão.
3. Diferentes arquiteturas, como o número de Camadas Ocultas em um Perceptron Multi-Camadas.
4. Diferentes modelos de classificadores: Envolve a combinação de diferentes modelos de classificação (Árvore de Decisão, Redes Neurais e SVM, por exemplo). A diversidade

neste caso vem de propriedades específicas de cada modelo, que mudam a forma que as fronteiras de decisão são geradas. Sistemas que usam diferentes classificadores em geral são chamados de Sistemas Heterogêneos.

5. Diferentes bases de treinamento: Cada classificador base é treinado em diferentes distribuições da base treinamento. Exemplos são o *Bagging* (BREIMAN, 1996) (SKURICHINA; DUIN, 1998) e o *Boosting* (FREUND; SCHAPIRE, 1997) (FENG *et al*., 2012)
6. Diferentes conjuntos de características: Usada em dados que podem ser representados em espaços de caracteríscas distintos. Cada classificador base pode ser treinado com diferentes subconjuntos de características ou com diferentes métodos de extração de características. Alguns exemplos são o *Random Subspace Method* (HO, 1998) (SKURICHINA; DUIN, 2002) e extração de características de imagens faciais (BASHBAGHI *et al*., 2017b).

A Figura 2 apresenta como é o funcionamento desta etapa, considerando que foram gerados 3 classificadores (C1, C2 e C3).

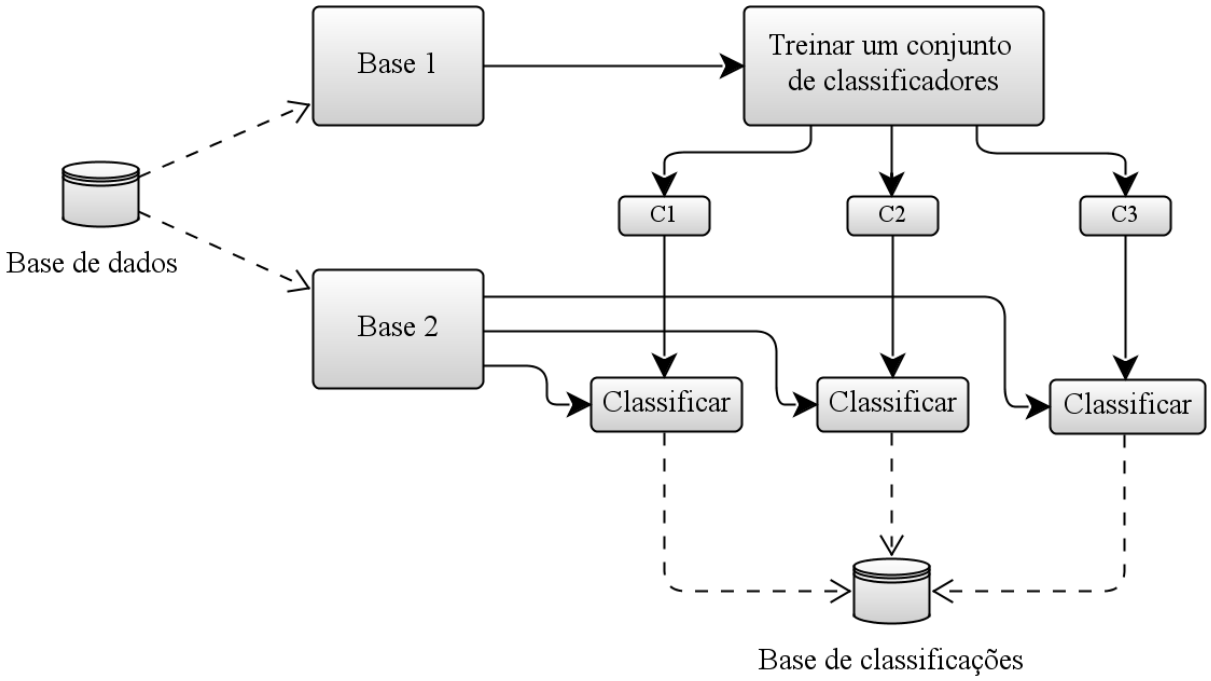

Figura 2 – Funcionamento da etapa de geração

Como apresentado na Figura 2, nesta etapa inicialmente uma base de dados inicial é aleatoriamente separada em duas bases mutuamente exclusivas. São então treinados um

conjunto de classificadores na base 1 (usando as estratégias citadas), e cada um deles são usados para classificar a base 2. As classificações são então salvas em uma base de classificações, onde as linha dessa base representam os padrões da base 2 e as colunas as classificações geradas.

2.2 Seleção de Classificadores

Na etapa de seleção então ocorrerá a seleção de classificadores, ou seja, serão selecionadas colunas da base de classificações gerada na etapa anterior. Existem dois tipos de processos para fazer a seleção: A seleção estática, onde, por exemplo, é selecionado um *ensemble* de classificadores para todos os padrões na base de teste, ou a seleção dinamica, onde é selecionado diferentes *ensembles* de classificadores para cada padrão na base de teste (KO; SABOURIN; BRITTO, 2008).

(KO; SABOURIN; BRITTO, 2008) salientam que o processo de seleção estática pode ser divido em dois passos: "(1) Encontrar uma função objetivo pertinente para selecionar os classificadores; e (2) usar um algoritmo de busca pertinente para aplicar esse critério". Eles enfatizam que um dos elementos mais importantes na seleção de classificadores relevantes é a função objetivo (BROWN *et al*., 2005) (SCHAPIRE, 2003) (RUTA; GABRYS, 2005). Ainda de acordo com (KO; SABOURIN; BRITTO, 2008), como função objetivo, o erro de votação da maioria já foi mostrado como sendo uma das melhores para este fim. Já no caso de algoritmo de busca, o Algoritmo Genético é considerado vantajoso devido à sua abordagem baseada em população, mas ainda não existe um que seja universalmente considerado o melhor.

O raciocíno por trás da seleção dinâmica é que cada classificador base é um especialista em uma região local diferente no espaço de características (ZHU; WU; YANG, 2004). Desta maneira, a seleção dinâmica tem o objetivo de achar o classificador ou subconjunto de classificadores que é mais competente na região local em que um dado padrão de teste está presente (CRUZ; SABOURIN; CAVALCANTI, 2018). Essa região local é comumente chamada de região de competência do classificador, e em geral pode ser definida pelas seguintes técnicas: K-NN (KO; SABOURIN; BRITTO, 2008) (WOODS; KEGELMEYER; BOWYER, 1997), por métodos de clusterização (KUNCHEVA, 2000) (SOARES *et al*., 2006), usando a decisão de classificadores base (CAVALIN; SABOURIN; SUEN, 2013) (GIACINTO; ROLI, 2001) (CAVALIN; SABOURIN; SUEN, 2012) ou mapa de competências que é definido através do uso de uma função potencial (WOLOSZYNSKI; KURZYNSKI, 2011).

2.3 Agregação de Classificadores

Na etapa de agregação, serão combinadas as classificações referentes aos classificadores selecionados, de acordo com uma regra de combinação (CRUZ; SABOURIN; CAVALCANTI,

2018). A combinação dos classificadores base pode ser realizada com relação nos rótulos das classes, como no esquema de votação da maioria, ou usando as probabilidades obtidas pelo classificador base para cada uma das classes do problema de classificação. Existem três estratégias principais para a fase de agregação: não-treinável, treinável e ponderação dinâmica.

Na não-treinável, é usada uma regra de combinação fixa. Alguns exemplos são métodos de Soma, Produto, Máximo, Mínimo, Mediana, e Voto da Maioria (KITTLER; HATER; DUIN, 1996), *Borda count* (GIACINTO; ROLI, 2001), *Behavior Knowledge space* (HUANG; SUEN, 1995), *Decision Templates* (KUNCHEVA; BEZDEK; DUIN, 2001) e *Dempster-Shafer combination* (LU, 1996) (ROGOVA, 1994).

Na estratégia treinável, ao invés de se usar uma regra de combinação fixa, a combinação pode ser adaptada à especificidade do problema de classificação (CRUZ; SABOURIN; CAVALCANTI, 2018). Para isso, as saídas do classificador base são usadas como características de entrada para outro algoritmo de aprendizagem, que aprende a função de agregação com base nos dados de treinamento, abordagem conhecida como *Stacked generalization* (WOLPERT, 1992).

Já a última estratégia, a ponderação dinâmica, é semelhante aos métodos de seleção dinâmica descritos anteriormente. Assim como eles, ela também é baseada na competência local dos classificadores base na região onde padrão de teste está. Contudo, ao invés da seleção de um subconjunto de classificadores, todos os classificadores são agregados para dar a decisão final, de modo que cada classificador recebe um valor de peso de acordo com sua competência, ou seja, o mais competente recebe um peso maior, o menos um menor e assim por diante (CRUZ; SABOURIN; CAVALCANTI, 2018). Ponderação local de classificador por programação quadrática (CEVIKALP; POLIKAR, 2008), integração dinâmica de classificadores (TSYMBAL *et al*., 2008) (JIMENEZ, 1998) e agregação de classificadores dinâmica de fuzzy (ŠTEFKA; HOLEŇA, 2015) são alguns exemplos de ponderação dinâmica.

3 Referencial Teórico

3.1 Aprendizagem de máquina

3.1.1 Introdução

A inteligência artificial é um campo que tenta compreender e construir entidades inteligentes, sendo ela um dos campos mais recentes em ciências e tecnologias (PETER; RUSSELL, 2013). Os subcampos abrangidos por ela são muitos, sendo eles tanto de propósitos gerais, quanto específicos.

A aprendizagem de máquina é um dos subcampos de propósitos gerais da inteligência artificial. Ela é amplamente definida como o uso de modelos computacionais que usam a experiência para melhorar desempenho ou para fazer previsões precisas (MOHRI; ROSTAMIZADEH; TALWALKAR, 2012). Em geral, a experiência é adquirida por meio de dados coletados que ficam disponíveis para análise, onde o tamanho e qualidade são cruciais para uma boa qualidade de previsões.

As tarefas de aprendizado de máquinas em geral são classificadas em três categorias (PETER; RUSSELL, 2013) (MOHRI; ROSTAMIZADEH; TALWALKAR, 2012):

- Aprendizado supervisionado: Os dados recebidos pelo modelo de inteligência artificial são rotulados nos dados de treinamento, e são feitas previsões em novos dados não rotulados. Problemas de classificação e regressão são comuns nessa categoria.
- Aprendizado não supervisionado: O modelo recebe dados não rotulados, e faz previsões em novos dados também não rotulados. Nessa categoria, problemas de agrupamento são muito comuns.
- Aprendizado por reforço: O modelo aprende interagindo dinamicamente com o ambiente, recebendo retornos que podem ser recompensas por ações corretas ou punições por ações incorretas. Essa categoria está muito relacionada a teoria dos jogos, ou seja, modelos que se interagem com ambientes de variados tipos de jogos, tentando ser jogadores autônomos, por exemplo.

Tratando-se mais especificamente do aprendizado supervisionado, de acordo com (PETER; RUSSELL, 2013), ele possui a seguinte tarefa:

Considerando uma base de treinamento de N pares de padrões $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$, sendo cada x_i uma entrada e y_i uma saída gerada por uma função desconhecida $y = f(x)$, o objetivo deste tipo de aprendizado é descobrir uma função h que se aproxime da função f. Essa função h é o que compõe um modelo de aprendizado de máquina.

Uma observação importante é que cada entrada x_i de um padrão é composta por M características relativas ao problema em questão, ou seja, a função h é gerada em relação a essas características. Esse processo de geração é conhecimento como treinamento, pois este é um processo que usa uma técnica de aprendizado de máquina para criar essa função através de uma base de treinamento.

Neste tipo de aprendizado, existem diferentes técnicas para a criação de modelos de classificação e regressão. Ainda de acordo (PETER; RUSSELL, 2013), uma tarefa de aprendizagem é chamada de classificação quando a saída y_i de um padrão possui apenas valores pertencentes a um conjunto finito e discreto de valores, sendo a classificação binária um caso particular, onde esse conjunto possui apenas dois valores. Já a regressão é quando a saída y_i de um padrão possui valores pertencentes a um conjunto não finito e contínuo de valores.

Além do processo de treinamento, na criação de modelos de aprendizado de máquina existem os processos de validação e teste. No processo de validação, é usado uma métrica para validar o desempenho do classificador criado. Mais detalhes desse processo serão dados na Seção 3.2. O processo de teste é quando o classificador criado é aplicado na previsão de novos padrões, não pertencentes a base de treinamento usada na criação do modelo.

A seguir na Subseção 3.1.2 será introduzido o método *ensemble*. Será descrito seu uso apenas para classificação, mas ele também pode ser usado em problemas de regressão.

3.1.2 Técnicas baseadas em *ensemble*

A principal ideia por trás de métodos de *ensemble* é combinar vários classificadores individuais, com o objetivo de criar um novo classificador mais preciso, que supere o desempenho de cada um dos classificadores individuais (ROKACH, 2010) (MOHRI; ROSTAMIZADEH; TALWALKAR, 2012).

De acordo com (OPITZ; MACLIN, 1999), os dois principais métodos para a criação de *ensembles* são o *Bagging* (BREIMAN, 1996) e *Boosting* (SCHAPIRE, 1990) (FREUND; SCHAPIRE, 1996). Esses métodos utilizam reamostragem na base de treinamento, de modo que cada um dos classificadores individuais seja treinado com diferentes partes dessa base. A seguir são apresentados a base por trás de cada um dos métodos.

O *Bagging*, abreviação de *Boostrap Agregating*, é uma das mais antigas técnicas baseadas em *ensemble*. Ela consiste em treinar cada um dos classificadores individuais em diferentes subconjuntos da base total de dados. Cada subconjunto é formado extraindo aleatoriamente padrões da base original, usando reposição. Os classificadores individuais podem ser, por

exemplo, combinados usando voto majoritário, ou seja, a moda de suas previsões (BREIMAN, 1996) (OPITZ; MACLIN, 1999) (POLIKAR, 2006).

Similar ao *Bagging*, o *Boosting* também utiliza a combinação de diferentes classificadores individuais gerados a partir de diferentes subconjuntos, usando o voto majoritário para essa combinação. A diferença é que os classificadores individuais são colocados em série, sendo que classificador N depende do classificador $N-1$. Dessa forma, quando se vai gerar um subconjunto aleatório dos dados para treinar o classificador N, os padrões previstos de forma errada pelo classificador $N-1$ possuem um peso maior, tendo maiores chances de serem escolhidos. Dessa forma, o classificador $N-1$ tende a focar mais em padrões que são difíceis de prever (FREUND; SCHAPIRE, 1996) (OPITZ; MACLIN, 1999) (ROKACH, 2010).

As técnicas mais avançadas baseadas nesses dois métodos possuem utilização dos funcionamentos citados como base de seus algoritmos, porém incrementando essas técnicas com novos conceitos. No *Bagging*, alguns exemplos dessas técnicas são o método *Random Forest* (BREIMAN, 2001) e o método *Extra-Trees* (GEURTS; ERNST; WEHENKEL, 2006). Já no *Boosting*, pode citar-se o *Adaboost* (FREUND; SCHAPIRE, 1996) e o *XGBoost* (CHEN; GUESTRIN, 2016).

3.2 Validação

No aprendizado de máquina, durante o processo de criação de um classificador um ponto importante é a medição de seu desempenho na classificação de padrões. Na Subseção 3.2.1 será descrito o método de validação cruzada para a escolha de uma base de validação, e nas duas seções seguintes serão descritas algumas métricas de avaliação de desempenho usadas no ambiente de classificação binária: Na Subseção 3.2.2 será apresentada a matriz de confusão e na Subseção 3.2.3 será apresentada a análise ROC, sigla de *Receiver Operating Characteristics*.

3.2.1 Validação Cruzada

Um dos métodos de validação cruzada, chamado de *k-fold cross validation*, consiste na divisão da base de treinamento em k subconjuntos aleatórios e mutuamente exclusivos $D_1, D_2, \dots, D_k$ de aproximadamente mesmo tamanho (KOHAVI, 1995). Em seguida são realizadas k etapas de treinamento e de teste. Em cada etapa $e \in \{1,2,\dots,k\}$ o classificador é treinado em todos os conjuntos com exceção de D_e (formando a base de treinamento), e testado no conjunto D_e (base de validação). É escolhida uma métrica, que é sempre aplicada no processo de teste do classificador. No final, é feita uma média do resultado da métrica aplicada nas k etapas, que indicará o desempenho do classificador.

3.2.2 Matriz de Confusão

Em um ambiente de classificação binária, existem duas classes possíveis que um padrão pode pertencer: Classe 0 (Classe negativa) ou classe 1 (Classe positiva), onde temos a matriz demonstrada pela Figura 3:

		Classe predita	
		1	0
Classe Real	1	VP	FN
	0	FP	VN

Figura 3 – Matriz de confusão para classificação binária

A matriz de confusão apresentada na Figura 3 mostra, para cada classe, o número de classificações reais em oposição às classificações preditas. Quando um classificador binário já treinado recebe um padrão desconhecido para classificação, existem quatro possibilidades (FAWCETT, 2006):

- VP (Verdadeiro Positivo): o modelo classifica como classe 1 e o padrão é realmente da classe 1;
- VN (Verdadeiro Negativo): o modelo classifica como classe 0 e o padrão é realmente da classe 0;
- FP (Falso Positivo): o modelo classifica como classe 1, mas na verdade o padrão é da classe 0;
- FN (Falso Negativo): o modelo classifica como classe 0, mas na verdade o padrão é da classe 1.

Vale salientar que a diagonal principal da matriz representa as possiblidades de acerto, enquanto a diagonal secundária representa as possibilidades de erro.

A acurácia, descrita pela fórmula em (1), é uma métrica simples derivada da matriz de confusão, que fornece a taxa de acerto de padrões previstos por um classificador.

$$Acurácia = \frac{VP + VN}{\mathrm{VP} + \mathrm{VN} + \mathrm{FP} + \mathrm{FN}} \quad (1)$$

Contudo, existe um problema com esta métrica: De acordo com (HE; GARCIA, 2009), ela não é adequada para ser aplicada em dados que possuam uma distribuição desbalanceada. Por exemplo, se 95% dos padrões pertencem a classe 1, e 5% pertencem a classe 0, caso um classificador acerte todos os padrões da classe 1 e erre todos da classe 0, ele obterá uma acurácia de 95%. Isso é um problema, pois apesar de todos os padrões da classe majoritária serem classificadores corretamente, todos da classe minoritária estão incorretos, e mesmo assim a acurácia obtida parece ser adequada.

A seguir será apresentada uma métrica que não possui esse problema. Ela foi a métrica escolhida para validar os experimentos deste trabalho.

3.2.3 Análise ROC

(FAWCETT, 2006) conclui que a análise ROC é uma métrica rica em relação a outras métricas como a acurácia. Além dela permitir visualização gráfica, ela possui propriedades que a deixam mais adequada para a avaliação de bases desbalanceadas.

Existem duas análises que podem ser feitas com a ROC: O gráfico ROC e a curva ROC.

O gráfico ROC é formado pela plotagem da taxa de VP (Verdadeiros Positivos) no eixo y e a taxa de FP (Falsos Positivos) no eixo x, derivados da matriz de confusão pelas fórmulas em (2) e (3):

$$Taxa\ de\ VP = \frac{VP}{\mathrm{VP} + \mathrm{FN}} \tag{2}$$

$$Taxa\ de\ FP = \frac{FP}{\mathrm{FP} + \mathrm{VN}} \tag{3}$$

O desempenho de um classificador é dado por um ponto no gráfico, permitindo analisar benefícios (Taxa de VP) e custos (Taxa de FP). A Figura 4 mostra um exemplo de gráfico, para 7 classificadores hipotéticos.

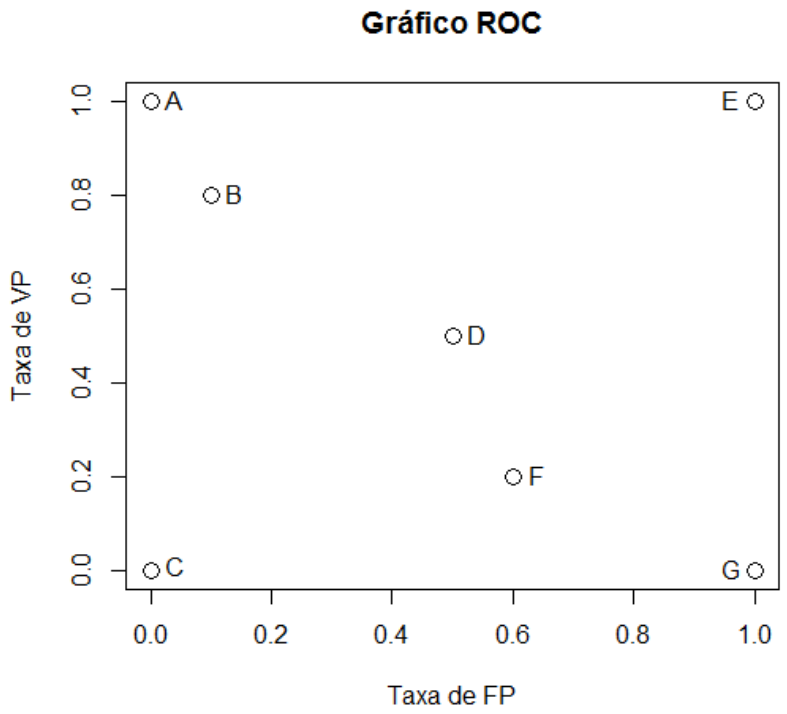

Figura 4 – Exemplo de gráfico ROC para 7 classificadores

O ponto A representa uma classificação ótima, onde todos os padrões da classe positiva e negativa foram acertados (indicados pela Taxa de VP igual a 1 e de FP igual a 0, respectivamente). Um classificador pode ser considerado melhor do que o outro se ele estiver mais próximo desse ponto (por exemplo, o ponto *B* é melhor que o ponto *D*). Classificadores plotados no eixo $x = y$ são classificadores que possuem o desempenho de um palpite aleatório, como os pontos *C*, *D* e *E*. Qualquer classificador que apareça abaixo do eixo $x = y$ possui um desempenho pior que um palpite aleatório, como os pontos *F* e *G*. Finalmente, um classificador no ponto *G* indicaria que ele errou todos os padrões que tentou prever.

Já a curva ROC pode ser usada em classificadores que ao invés de prever a classe que um padrão vai estar, preveem um score ou uma probabilidade de ele pertencer a essa classe. Ao utilizar um limiar, é possível obter uma saída discreta de qual classe esse padrão está. Por exemplo, temos dois padrões, um com 40% e outro com 65% de probabilidade de estar na classe 1. Usando um limiar de 50%, o primeiro padrão seria classificado na classe 0, e o segundo na classe 1.

A curva ROC é gerada justamente com a variação desse limiar. Esse limiar vai variando de 0 a 100%, e em cada variação é plotado um ponto no gráfico ROC que corresponde ao

desempenho do classificador com aquele limiar. A ligação de todos esses pontos tende a uma curva à medida que o número de variações do limiar tende ao infinito.

Uma métrica derivada da ROC é a AUC – *Area Under Curve*, onde o desempenho de um classificador é dado pela área abaixo de sua curva ROC (FAWCETT, 2006). Como a AUC é a área de uma porção de um quadrado, seu valor está sempre entre 0 e 1,0. Quanto mais próximo de 1,0 a AUC de um classificador está, melhor é considerado seu desempenho. Como o palpite aleatório está na linha diagonal do gráfico ROC, ele teria uma área de 0,5, ou seja, uma AUC de 0,5 é equivalente a um palpite aleatório. Dessa maneira, qualquer classificador com uma AUC menor que 0,5 é considerado pior do que um palpite aleatório.

A AUC é a métrica escolhida para validar o desempenho da abordagem proposta neste trabalho.

3.3 Pesquisa operacional

A pesquisa operacional, frequentemente referenciada pela sigla PO, consiste no desenvolvimento de métodos científicos para analisar sistemas complexos e tomar decisões (ARENALES *et al*., 2007). Em geral, a PO tenta encontrar uma melhor solução para um problema, havendo a possibilidade de haver várias soluções consideradas como melhores (HILLIER; LIEBERMAN, 2006). Essa melhor solução é chamada de solução ótima.

As fases usuais de um estudo de PO podem ser resumidas pelos seguintes tópicos: (ARENALES *et al*., 2007) (HILLIER; LIEBERMAN, 2006):

- Definição do problema: ocorre uma observação e formulação cuidadosa do problema, o que envolve também a coleta de dados.
- Construção do modelo: é construído um modelo científico (tipicamente matemático) que tenta abstrair a essência do problema real. Essa fase também é chamada de modelagem do problema. ;
- Solução do modelo: Utiliza métodos de solução e algoritmos conhecidos para resolver o problema;
- Validação do modelo: é verificado se o modelo proposto representa apropriadamente o problema, ou seja, se as soluções obtidas por ele também são validas para o problema real.
- Implementação: É feita a implementação da solução na prática.

Para um melhor entendimento da abordagem desenvolvida neste trabalho, considera-se importante o detalhamento de duas dessas fases: Construção do modelo e implementação. Eles serão feitos nas seções a seguir.

3.3.1 Construção do modelo

Um modelo matemático é definido em termos de símbolos e expressões matemáticas. Ele descreve a essência do problema por meio de um sistema de equações e de expressões matemáticas, que são dependentes de quatro fatores: Variáveis de decisão, função objetivo, restrições e parâmetros (HILLIER; LIEBERMAN, 2006).

As variáveis de decisão representam decisões quantificáveis que devem ser feitas, cujos respectivos valores devem ser determinados. Por exemplo, se existirem n decisões a serem tomadas, elas podem ser representadas pelas variáveis $x_1, x_2, \ldots, x_n$.

A função objetivo é uma medida de desempenho apropriada expressa como uma função matemática das variáveis de decisão (como $Z = 3x_1 + x_2, + \cdots + 9x_n$). Essa função pode ser maximizada ou minimizada dependendo do domínio do problema. Por exemplo, pode ser um problema onde se tem que maximizar a produção de uma empresa, ou minimizar o seu prejuízo.

As restrições são expressões matemática (em geral expressões de desigualdades ou equações) atribuídas as variáveis de decisão que expressam limitações no problema, como a desigualdade $x_1 + 2x_2 > 10$. Por exemplo, a produção de um produto em uma determinada empresa pode ser limitada pela quantidade de um determinado material que está em estoque.

Já os parâmetros são constantes (os coeficientes e os lados direitos) presentes nas restrições e na função objetivo, definidos com base na coleta de dados do problema real.

Dessa maneira, o objetivo do modelo matemático é escolher os valores das variáveis de decisão com base nas restrições e parâmetros, tentando-se maximizar ou minimizar a função objetivo.

Existem diferentes exemplos de modelos matemáticos para diferentes problemas (HILLIER; LIEBERMAN, 2006). Alguns exemplos são: Programação linear, programação inteira, programação inteira mista e programação não-linear.

Um modelo de programação linear é um modelo em que as funções matemáticas que aparecem tanto na função objetivo quanto nas restrições são funções lineares.

Já um modelo de programação inteira (ou programação linear inteira) é usado quando as variáveis de decisão fazem sentido apenas se tiverem valores inteiros. Seu modelo matemático é um modelo de programação linear com restrições adicionais de que as variáveis devem ter valores inteiros.

Caso não seja necessário que todas as variáveis tenham valores inteiros, esse modelo é denominado programação inteira mista (MIP, do inglês *Mixed-Integer Programing)*, tipo de modelo usado neste trabalho, que é uma união dos dois últimos modelos.

Por fim, o modelo de programação não-linear é usado quando é necessário que sejam usadas restrições não-lineares para a modelagem do problema em questão, como restrições quadráticas, por exemplo.

3.3.2 Implementação

Para a solução do modelo, são necessários algoritmos para a solução de certos tipos de problema. Um exemplo essencial é o método Simplex, criado por George Dantzig, que é usado na solução de problemas com programação linear. Mais detalhes sobre ele pode ser encontrados no livro (HILLIER; LIEBERMAN, 2006). Sendo assim, para a solução desses algoritmos por computador, existem pacotes de software, normalmente comerciais, contando com sua implementação já pronta. O livro citado acima cita alguns exemplos, que serão apresentados a seguir.

No programa de planilhas *Microsoft Excel* é possível formular pequenos modelos de PO, em formato de planilha. Ele então usa o *Excel Solver* ou uma versão aperfeiçoada dele, o *Premium Solver for Education* para a solução desses problemas. Além do *Microsoft Excel, o* pacote Lindo e sua linguagem de modelagem Lingo também podem ser usados na solução de problemas não muito complexos.

Para a solução de problemas mais complexos e desafiadores, um pacote de softwares amplamente utilizado é o CPLEX, um produto da IBM, Inc., localizada no Vale do Silício. Desde seu lançamento comercial em 1988, o CPLEX já sofreu uma série de atualizações possibilitando ganhos de eficiência espetaculares, o que vêm ajudando a abrir caminho na solução de problemas de programação linear cada vez maiores.

Devido à complexidade do modelo proposto neste trabalho, o software CPLEX foi escolhido para a solução deste. Foi utilizada uma interface sua com a linguagem de programação Java.

4 Materiais e métodos

Este capítulo apresenta a abordagem proposta neste trabalho, levando em consideração as etapas de criação de um MCS descritas no capítulo 2. Na Seção 4.1 será descrito como houve a geração dos classificadores bases neste trabalho. O método de seleção de classificadores proposto será apresentado na Seção 4.2. Na Seção 4.3 será descrita a etapa de agregação de classificadores. As bases de dados utilizadas serão apresentadas na Seção 4.4. Na Seção 4.5 serão apresentados os experimentos realizados. Por fim, na Seção 4.6, serão dados alguns detalhes sobre a implementação e codificação deste trabalho.

4.1 Geração de classificadores

Na geração de classificadores, foram usadas as seguintes estratégias para geração de classificadores diversos, já descritas na Seção 2.1:

1. Diferentes inicializações: Cada modelo foi iniciado com diferentes sementes.
2. Diferentes escolhas de parâmetros: Cada classificador escolhido foi inicializado com diferentes parâmetros especificos de seu algoritmo.
3. Diferentes Arquiteturas: Os modelos de classificação selecionados neste trabalhos não possuem diferentes arquiteturas.
4. Diferentes modelos de classificadores: Foram usados os seguintes modelos: *XGBoost*, *Random Forest*, *Adaboost* e *Extra-Trees Classifier*. Estes classificadores foram escolhidos por serem baseados nas técnicas de *Bagging* e *Boosting*, como já mostrado na Subseção 3.1.2.
5. Diferentes bases de treinamento: Optou-se por não utilizar essa estratégia, pois a maior parte das bases de dados possuem uma quantidade pequena de padrões.
6. Diferentes conjuntos de características: Também optou-se por não utilizar estratégia, pois a algumas bases de dados selecionadas possuem um quantidade muito pequena de características.

Nesta etapa, foi gerado um número fixo de 20 classificadores. Como descrito na Seção 2.1, a base de dados inicial deveria ser separada em duas bases para se gerar os classificadores, o que implicaria em bases de dados de tamanho menores em relação a base inicial. Para evitar esse problema, foi usado um procedimento baseado no método *k-fold cross validation* descrito na Seção 3.2.1, que é muito utilizado na abordagem de *Stacked generalization* (WOLPERT, 1992). No final deste procedimento, será gerada uma base de classificações do mesmo tamanho da base inicial. A seguir será descrito como ele foi realizado.

Inicialmente, foi definido um número *k=5*, dividindo a base de dados inicial D em 5 subconjuntos aleatórios e mutuamente exclusivos D_1, D_2, D_3, D_4, D_5 de aproximadamente mesmo tamanho, assim como o método de *k-fold cross validation*.

Para cada um dos 20 classificadores, são realizadas 5 etapas de treinamento e de teste. Em cada etapa $e \in \{1,2,3,4,5\}$, o classificador é treinado em todos os conjuntos com exceção de D_e, e testado no conjunto D_e. Sempre que ocorre o teste em um conjunto, é gerado um vetor com as classificações de seus padrões. No final esses 5 vetores serão concatenados na sequência de seus respectivos conjuntos. Vale salientar que as classificações são dadas em probabilidades: Quanto mais próximo de 1 a probabilidade estiver, significa que é mais provável esse padrão pertencer a classe 1, e quanto mais próximo de 0, da classe 0.

No final, serão geradas 20 concatenações, onde cada uma delas representará um classificador gerado. As 20 concatenações são então combinadas, formando uma nova base com as classificações (de mesmo tamanho da base inicial), que será denominada de D'. Observe que a nova base de dados pode ser representada por uma matriz bidimensional, onde cada concatenação é uma coluna que representa um classificador e cada linha representa um padrão da base de dados inicial.

4.2 Modelo de Programação Inteira Mista para seleção de classificadores

O método proposto neste trabalho para a seleção de classificadores é baseado na seleção estática (Seção 2.2). Para a seleção de classificadores, ele usa um modelo de Programação Inteira Mista (Subseção 3.3.1) para achar um subconjunto ótimo de classificadores. O método criado é próprio para problemas de classificação binária.

A seguir serão definidos a função objetivo, variáveis de decisão, parâmetros e restrições que compõem a modelagem proposta. Explicações sobre eles serão dadas logo após as definições.

- Definição da função objetivo:

$$\max\ c_{vp}\sum_{i=1}^{n_1} vp_i + c_{fp}\sum_{j=1}^{n_0} fp_j + c_{vn}\sum_{j=1}^{n_0} vn_j + c_{fn}\sum_{i=1}^{n_1} fn_i \tag{4}$$

- Definição das variáveis de decisão:

$$x_i \in \{0,1\}\ \forall\ i = 1, \dots, m \tag{5}$$

$$limiar \in \{1, \dots, m\} \tag{6}$$

$$vp_i \in \{0,1\}\ \forall\ i = 1, \dots, n_1 \tag{7}$$

$$fn_i \in \{0,1\} \; \forall \; i = 1, \dots, n_1 \tag{8}$$

$$vn_i \in \{0,1\} \; \forall \; i = 1, \dots, n_0 \tag{9}$$

$$fp_i \in \{0,1\} \; \forall \; i = 1, \dots, n_0 \tag{10}$$

- Definição dos parâmetros:

$$c_{vp}, c_{fp}, c_{vn}, c_{fn} \in \mathbb{R} \tag{11}$$

$$n_0, n_1, m \in \mathbb{N}^+ \tag{12}$$

$$\beta_{ij} \in \{0,1\} \; \forall \begin{array}{l} i = 1, \dots, n_1 \\ j = 1, \dots, m \end{array} \tag{13}$$

$$\alpha_{ij} \in \{0,1\} \; \forall \begin{array}{l} i = 1, \dots, n_0 \\ j = 1, \dots, m \end{array} \tag{14}$$

- Definição das restrições:

$$vp_i = \begin{cases} 0, se \sum_{j=1}^{m} \beta_{ij} \, x_j \leq limiar \\ 1, caso\ contrário. \end{cases} \quad \forall \; i = 1, \dots, n_1 \tag{15}$$

$$fn_i + vp_i = 1 \; \forall \; i = 1, \dots, n_1 \tag{16}$$

$$vn_i = \begin{cases} 0, se \sum_{j=1}^{m} \alpha_{ij} \, x_j > limiar \\ 1, caso\ contrário. \end{cases} \quad \forall \; i = 1, \dots, n_0 \tag{17}$$

$$fp_i + vn_i = 1 \; \forall \; i = 1, \dots, n_0 \tag{18}$$

A função objetivo (4) maximiza o número de padrões classificadores corretamente pela combinação do *ensemble*. Já considerando os custos, as variáveis (7), (8), (9) e (10) indicam em qual possibilidade da matriz de confusão o padrão pertencerá após ser feita a combinação do *ensemble*, sendo eles, verdadeiro positivo e falso negativo caso o padrão seja da classe positiva ou verdadeiro negativo e falso positivo caso o padrão pertença a classe negativa, respectivamente de acordo com as variáveis citadas.

Dessa maneira (4) irá ponderar esses padrões pelos seus custos definidos por parâmetros (11). Essa ponderação é útil, pois diferentes problemas de classificação binária demandam que certas possibilidades da matriz de confusão sejam mais importantes do que outras. Técnicas de classificação sensível ao custo usam esse mesmo princípio.

Em (12) são indicados a quantidade de padrões na classe positiva (n_1), na classe negativa (n_0) e o número de classificadores criados na etapa de geração (m). A base de dados D' criada na etapa de geração é usada na definição dos parâmetros α e β, $\alpha \cup \beta = D'$. O parâmetro α em (14) é um subconjunto da base D' contendo os padrões dela que são pertencentes a classe 0 (Negativa). Já o parâmetro β em (13) possui os padrões da classe 1 (positiva). Um detalhe importante é que a base D', que ao ser gerada era composta por probabilidades, foi arredondada para os valores 0 e 1 antes de haver essa separação.

A variável definida em (5) indica se o classificador i foi selecionado (1) ou não (0) para ser integrado ao *ensemble*. Com isso, a multiplicação presente no somatório em (15) ficará zerada caso um classificador não seja selecionado. Para cada padrão presente na classe positiva, (15) possuirá o valor 0 se o somatório for menor que um limiar (6) (Ou seja, não é um verdadeiro positivo), e 1 caso contrário (é um verdadeiro positivo). A equivalência presente em (16) indicará que, caso um padrão na classe positiva não seja um verdadeiro positivo, ele será obrigatoriamente um falso negativo, obedecendo as propriedades da matriz de confusão. O raciocínio é equivalente para (17) e (18), porém para a classe negativa.

Como já comentado na Seção 3.3, foi utilizado o pacote de software CPLEX para a solução deste modelo. Definiu-se um tempo limite de 3600 segundos (1 hora) para o tempo de execução dele, sendo que quando esse tempo fosse atingido, não se teria garantias de que a solução dada por ele seria ótima.

Para deixar mais claro o funcionamento deste modelo, será dada uma exemplificação hipotética. A Figura 5 representará os padrões da classe positiva, e a Figura 6 representará os da classe negativa. No exemplo, considere os parâmetros $m = 7$ classificadores, $n_1 = 7$ padrões na classe positiva, $n_0 = 7$ padrões na classe negativa, β de acordo com a matriz na Figura 5, α de acordo com a matriz na Figura 6 e que em um dado momento da execução da solução as variáveis de decisão estejam com os valores $x_j = \{1,1,0,1,1,1,0\}$, $limiar = 3$. Considere ainda os valores das variáveis de vp_i e fn_i na Figura 5 e vn_i e fp_i na Figura 6.

x_j	1	1	0	1	1	1	0		
β	Padrão	C1	C2	C3	C4	C5	C6	C7	vp_i	fn_i
	1	1	1	1	1	1	0	0	1	0
	2	1	1	0	0	0	1	1	0	1
	3	0	0	1	0	0	1	0	0	1
	4	1	0	0	0	1	1	1	0	1
	5	1	1	0	1	1	1	0	1	0
	6	1	1	0	1	0	1	0	1	0
	7	1	1	1	0	0	0	0	0	1

$$limiar = 3$$

Figura 5 – Exemplificação da classe positiva

x_j	1	1	0	1	1	1	0		
α	Padrão	C1	C2	C3	C4	C5	C6	C7	vn_i	fp_i
	1	0	1	0	1	1	1	1	1	0
	2	0	1	0	0	0	1	1	0	1
	3	0	0	1	0	1	0	1	0	1
	4	1	1	0	1	1	1	1	1	0
	5	1	1	0	0	1	1	0	1	0
	6	1	0	0	1	0	0	0	0	1
	7	1	1	1	1	0	1	0	1	0

$$limiar = 3$$

Figura 6 - Exemplificação da classe negativa

Por exemplo, para a Figura 5, de acordo com a restrição (15) definida no modelo, ocorre o seguinte comportamento: Considerando-se os classificadores selecionados neste momento (variável vp_i), para o padrão 1, temos que o somatório das saídas dos classificadores para a sua linha é $1 + 1 + 1 + 1 + 0 = 4$, que é maior que o $limiar = 3$. Logo temos $vp_{i=1} = 1$ para

este padrão. Para o segundo padrão, temos $vp_{i=2} = 0$, pois o somatório desta linha é menor que o limiar. Observe também que o valor de fn_i é definido de acordo com $fn_i + vp_i = 1$, ou seja, a restrição (16). O raciocínio é equivalente para a Figura 6.

Sendo assim, temos que o valor da função objetivo (4) seria: $c_{vp} * (1 + 0 + 0 + 0 + 1 + 1 + 0) + c_{fp} * (0 + 1 + 1 + 0 + 0 + 1 + 0) + c_{vn} * (1 + 0 + 0 + 1 + 1 + 0 + 1) + c_{fn} * (0 + 1 + 1 + 1 + 0 + 0 + 1) = c_{vp} * 3 + c_{fp} * 3 + c_{vn} * 4 + c_{fn} * 4$.

Dados os parâmetros de custo $c_{vp} = 0{,}4$, $c_{fp} = -0{,}3$, $+c_{vn} = 0{,}6$ e $+c_{fn} = -0{,}2$, teríamos $0{,}4 * 3 - 0{,}3 * 3 + 0{,}6 * 4 - 0{,}2 * 4 = 1{,}9$. Caso se chegasse ao fim da execução da solução, e não se conseguisse melhorar esse valor da função objetivo, isso indicaria que a solução ótima seria dada pelas variáveis de decisão considerados nesse cálculo da função objetivo. Caso fosse possível melhorar, a solução ótima teria um valor de função objetivo maior (ela está sendo maximizada), e as variáveis de decisão teriam algum valor diferente destes.

4.3 Agregação de classificadores

Para a etapa de agregação é feita uma média aritmética das saídas dos classificadores selecionados, onde a média representará a combinação dos classificadores.

Baseado na Seção 2.3, essa abordagem de agregação possui propriedades tanto da abordagem da ponderação dinâmica quanto da não-treinável. As propriedades da abordagem de ponderação dinâmica é devido ao fato da variável x_i poder ser definida como pesos arredondados para 0 ou 1 para um dado classificador. Já a não-treinável é porque se usa a média aritmética para a combinação dos classificadores. Optou-se por usar a média, pois a saída final é dada em probabilidades, e não por categoria, onde se poderia usar o voto da maioria para a combinação, por exemplo. A probabilidade é dada de modo que ela é mais próxima de 1 caso seja mais provável que o padrão pertença a classe positiva, e de 0 para a classe negativa.

4.4 Base de dados

Foram escolhidas cinco bases de dados para a realização deste trabalho, adquiridas do repositório UCI (University of California – Irvine) (LICHMAN, 2013). As bases escolhidas possuem cerca de 800 a 5000 padrões e de 19 a 57 características. Para cada base de dados selecionada será inicialmente feita uma breve descrição sobre ela, e em seguida serão realizadas considerações sobre seu grau de desbalanceamento.

4.4.1 Retinopatia Diabética

A primeira base de dados é uma base de retinopatia diabética, complicação da doença diabetes que afeta os olhos. Ela contém características extraídas de uma base de imagens usada para classificar se uma imagem contém sinais de retinopatia diabética ou não. Todas as

características representam uma lesão detectada (microaneurismas, exsudatos), uma característica descritiva de uma parte anatômica (mácula, disco óptico) ou um descritor de nível de imagem (avaliação de qualidade, pré-seleção, AM/FM). O método de análise de imagem e extração de características pode ser encontrado em (ANTAL; HAJDU, 2014).

A base de dados possui um total de 1151 padrões e 20 características. A Tabela 1 mostra a distribuição de padrões em cada classe nesta base de dados. Considere a classe 1 no caso de haver sinais de retinopatia diabética na imagem, e a classe 0 no caso de não haverem sinais.

Tabela 1 - Distribuição de classes na base de dados de retinopatia diabética

Classes	Número de Padrões	Proporção
Classe 0	540	46,92%
Classe 1	611	53,08%

Devido a proporção das duas classes tender a 50%, considerei o grau de desbalancemnto desta base de dados como baixo. Ela pode ser encontrada no seguinte endereço eletrônico:

https://archive.ics.uci.edu/ml/datasets/Diabetic+Retinopathy+Debrecen+Data+Set

4.4.2 Biodegradação de molécula

A segunda base de dados é uma base de biodegradação de molécula. Ela possui características extraidas de relações entre estrutura química e da biodegradação de moléculas, com o objetivo de prever se uma molécula é facilmente biodegradável ou não facilmente biodegradável. Mais detalhes sobre as características podem ser encontrados em (MANSOURI *et al.*, 2013), onde esta base de dados foi utilizada para desenvolver modelos de QSAR - *Quantitative Structure−Activity Relationship* (Relação de Estrutura-Atividade Quantitativa) para a previsão da biodegradabilidade da molécula.

A base de dados possui um total de 1055 padrões e 41 características. A Tabela 2 mostra a distribuição desta base de dados. Considere a classe 0 no caso da molécula ser facilmente biodegradável, e a classe 1 no caso de não ser facilmente biodegradável.

Tabela 2 - Distribuição de classes na base de dados de biodegradação de molécula

Classes	Número de Padrões	Proporção
Classe 0	356	33,74%
Classe 1	699	66,26%

Essa base de dados possui um grau de desbalanceamento maior do que a base de dados de retinopatia diabética, porém ainda não sendo extremo. Por isso será considerado que esta tem um grau médio de desbalanceamento. Ela pode ser encontrada no seguinte endereço eletrônico:

https://archive.ics.uci.edu/ml/datasets/QSAR+biodegradation

4.4.3 *Spam*

A terceira base de dados é uma base de dados de *spam* em mensagens de e-mail. Ela contém características que representam mensagens de *e-mails*, que podem ser indicadoras ou não indicadoras de *spam*. O objetivo desta base é prever se uma mensagem *e-mail* foi marcada como *spam* ou não.

A base de dados possui um total de 4601 padrões e 57 características. A Tabela 3 mostra a distribuição desta base de dados. Considere a classe 1 no caso do e-mail ter sido considerado spam e classe 0 no caso contrário.

Tabela 3- Distribuição de classes na base de dados de spam

Classes	Número de Padrões	Proporção
Classe 0	2788	60,60%
Classe 1	1813	39,40%

A proporção desta base é semelhante a da base anterior, porém com as classes inversas. Assim considerei que esta de base de dados também possui um grau médio. Ela pode ser encontrada no seguinte endereço eletrônico:

https://archive.ics.uci.edu/ml/datasets/spambase

4.4.4 Câncer Cervical

A quarta base de dados é uma base de câncer cervical, tumor maligno do colo do útero. As características desta base de dados envolvem informações demográficas, hábitos e registros médicos históricos de pacientes. O foco dela é a previsão se um diagnóstico de câncer cervical é positivo ou negativo.

A base de dados possui um total de 858 padrões e 36 características. A Tabela 4 mostra a distribuição desta base de dados. Considere a classe 0 no caso do diagnóstico ser negativo e classe 1 no caso de ser positivo.

Tabela 4- Distribuição de classes na base de dados de Câncer Cervical

Classes	Número de Padrões	Proporção
Classe 0	803	93,59%
Classe 1	55	6,41%

Ao contrário das bases de dados anteriores, existe uma grande diferença de proporções entre as duas classes. Dessa maneira, considerei esta base como tendo um alto grau de desbalanceamento. Ela pode ser encontrada no seguinte endereço eletrônico:

https://archive.ics.uci.edu/ml/datasets/Cervical+cancer+(Risk+Factors)

4.4.5 Solavancos Sísmicos

A quinta base de dados é uma base de solavancos sísmicos. Ela possui características extraídas de medições relativas a solavancos sísmicos em minas de carvão (KABIESZ *et al*., 2013). Neste caso um solavanco é definido como um tremor destrutivo de alta energia que resulta na ruptura ou colapso de rochas em uma mina. O objetivo é prever se um solavanco ocorrido é perigoso o suficiente para que seja necessária a evacuação da mina.

A base de dados possui um total de 2584 padrões e 19 características. A Tabela 5 mostra a distribuição desta base de dados. Considere a classe 0 como solavancos não tão perigosos e a classes 1 como solavancos em que é necessária a evacuação da mina.

Tabela 5 - Distribuição de classes na base de dados de Solavancos Sísmicos

Classes	Número de Padrões	Proporção
Classe 0	2414	93,42%
Classe 1	170	6,58%

Esta base de dados possui uma proporção semelhante a base anterior. Dessa maneira, também considerei ela com tendo um alto grau de desbalanceamento. Ela pode ser encontrada no seguinte endereço eletrônico:

https://archive.ics.uci.edu/ml/datasets/seismic-bumps

4.5 Experimentação

Os experimentos foram de acordo com o pseudocódigo apresentado na Figura 7.

```
1   FluxoDeExperimentação
2   Inicio
3       Para cada base de dados D faça:
4           ETAPA DE GERAÇÃO: Gerar m classificadores e usá-los para criar a
5                                 base de dados D';
6           Definir k folds de acordo com a base D';
7           Para cada experimento faça:
8               Para cada fold faça:
9                   Obter as bases de treinamento e validação em relação ao fold atual;
10                  ETAPA DE SELEÇÃO: Aplicar o modelo de Programação Linear Inteira
11                                    na base de treinamento;
12                  ETAPA DE AGREGAÇÃO: Agregar os classificadores selecionados e
13                                      classificar a base de validação;
14                  Calcular o desempenho com a métrica AUC das classificações na
15                  base de validação;
16              Calcular o desempenho médio dos folds fazendo-se uma média das AUC's
17              calculadas;
18  Fim
```

Figura 7 – Pseudocódigo do fluxo de experimentação

Como apresentado na Figura 7, é realizado o mesmo processo para cada base de dados, que será descrito nos parágrafos a seguir.

Inicialmente é realizada a etapa de geração descrita na Seção 4.1 (linhas 4 e 5 do pseudocódigo), onde foi fixado um número de $m = 20$ classificadores a serem gerados.

Para cada experimento que será realizado, será usado o processo de *k-fold cross validation* (Subseção 3.2.1), onde foi fixado k=5 *folds* para todos os experimentos. Após a definição dos *folds* (linha 6), para cada experimento e para cada *fold*, são geradas a base de treinamento e a base de validação em relação ao *fold* atual (linha 9), e então são executadas a etapa de seleção (linhas 10 e 11) e a etapa de agregação (linhas 12 e 13), descritas na Seção 4.2 e 4.3, respectivamente.

O modelo de Programação Linear Inteira é aplicado na base de treinamento, onde após a solução do modelo foi encontrado um valor para as variáveis x_i, simbolizando os classificadores selecionados. Em seguida, com base nesses classificadores, foi realizada a etapa de agregação, onde as classificações na base de validação realizadas pelos classificadores são agregadas (usando a média).

Em seguida, é calculada a métrica AUC para analisar o desempenho do classificador na base de validação (linha 14). Após o processo ser realizado para todos os *folds*, é feita a média aritmética das k=5 AUC's calculadas em cada *fold* (linhas 16 e 17), representando então o desempenho médio do modelo.

Vale salientar que o processo de *k-fold cross validation* agora não foi usado para criar uma nova base de dados como na etapa de geração, mas sim para validar o desempenho do modelo proposto.

Os experimentos nas bases de dados iniciados no *loop* da linha 7 foram definidos da seguinte maneira: Foram executados 9 experimentos para cada base de dados, com diferentes variações dos parâmetros c_{vp}, c_{fp}, c_{vn} e c_{fn}. Para definir os parâmetros, optei por inicialmente o c_{vp} foi variado de 0,1 a 0,9 com um passo de 0,1, ou seja: $c_{vp} = \{0.1, 0.2, \ldots, 0.9\}$; Os outros parâmetros foram calculados em função dele:

- $c_{vn} = 1 - c_{vp}$, ou seja: $\{0.9, 0.8, \ldots, 0.1\}$;
- $c_{fp} = -c_{vn}$, ou seja: $\{-0.9, -0.8, \ldots, -0.1\}$;
- $c_{fn} = -c_{vp}$, ou seja: $\{-0.1, -0.2, \ldots, -0.9\}$.

Os valores de cada variação de parâmetro são apresentados na Tabela 6.

Tabela 6 - Variação dos parâmetros c_{vp}, c_{fp}, c_{vn} e c_{fn}.

Experimento	c_{vp}	c_{fp}	c_{vn}	c_{fn}
1	0,1	-0,9	0,9	-0,1
2	0,2	-0,8	0,8	-0,2
3	0,3	-0,7	0,7	-0,3
4	0,4	-0,6	0,6	-0,4
5	0,5	-0,5	0,5	-0,5
6	0,6	-0,4	0,4	-0,6
7	0,7	-0,3	0,3	-0,7
8	0,8	-0,2	0,2	-0,8
9	0,9	-0,1	0,1	-0,9

Além disso, devido ao grau de desbalanceamento elevado nas bases de dados 4 e 5, adicionaram-se mais duas outras variações de parâmetros, dessa vez com o c_{vp} igual 0.01 e 0.09, como mostrado na Tabela 7.

Tabela 7 - Variação dos parâmetros c_{vp}, c_{vp}, c_{vn} e c_{fn} adicionadas nos experimentos com as bases de dados 4 e 5.

Experimento	c_{vp}	c_{fp}	c_{vn}	c_{fn}
10	0,01	-0,99	0,99	-0,01
11	0,99	-0,01	0,01	-0,99

Para a comparação que será realizada no Capítulo 5, executou-se também um experimento sem a realização da etapa de seleção (possibilidade discutida no Capítulo 2), onde a etapa de agregação foi realizada combinando todos os classificadores. Esse experimento será referenciado como "*ensemble* completo".

O motivo de se ter optado por variar apenas o parâmetro c_{vp} de 0,1 a 0,9, e definir os outros parâmetros em função foi o seguinte: Como mostrado na função objetivo definida na equação (4), os custos são como pesos que irão ponderar os padrões que forem classificados em cada uma das quatro possibilidades da matriz de confusão. Assim, definindo o custo de 0,1 para os VP's, é como se eles tivessem 10% de importância, enquanto os VN's teriam 90%.

Como os custos c_{fn} e c_{fp} estão relacionados a erros de classificação, optei por usá-los com valores negativos, assim cada exemplo que seja classificado incorretamente representará uma penalidade na função objetivo. Para a definição deles, usei o seguinte raciocínio para a definição desses custos: Se for dada uma importância maior para a uma determinada classe, caso se erre um padrão dessa classe, o custo desse erro deve ser proporcional a importância dessa classe. Ou seja, no caso em que o c_{fp} foi definido como $-c_{vn}$, o padrão que é um FP na verdade é um padrão da classe negativa que foi classificado como positivo. Assim, o custo de errar um padrão da classe negativa (c_{fp}) deve ser proporcional a importância de sua classe (c_{vn}). O raciocínio é equivalente para c_{fn}.

4.6 Detalhes de implementação

Para o desenvolvimento deste trabalho, foram adotadas as linguagens de programação Python e Java.

A linguagem Python foi escolhida para a implementação do código de geração de classificadores, de agregação e de validação. Ela foi escolhida para essas tarefas por possuir pacotes com as técnicas de inteligência artificial que serão usadas neste trabalho já implementadas.

Dentre os pacotes que serão utilizadas neste trabalho, está incluído o scikit-learn[1], uma biblioteca que dá suporte a uma grande variedade de técnicas e ferramentas já implementadas, incluindo modelos de classificação e métricas de avaliação. Também foi utilizado um pacote

[1] Detalhes sobre o pacote skicit-learn:
<http://scikit-learn.org/stable/>. Acessado em: 16 jun. 2017.

separado para o modelo *XGBoost*[2] por não haver uma implementação dele no pacote scikit-learn.

Para a implementação do código de seleção de classificadores, foi utilizada a linguagem de programação Java foi usada, integrado ao pacote de software CPLEX[3], como descrito na seções 3.3 e 4.2.

A comunicação entre as duas linguagens foi feita por meio de leitura e escrita em arquivos de texto.

---

[2] Detalhes sobre o pacote *XGBoost*:

<https://xgboost.readthedocs.io/en/latest/>. Acessado em: 16 jun. 2017.

[3] Detalhes sobre o pacote de software CPLEX:

<https://www.ibm.com/analytics/data-science/prescriptive-analytics/cplex-optimizer>. Acessado em: 12 jan. 2018.

5 Resultados e discussões

Neste capítulo serão apresentados e analisados os resultados dos experimentos realizados. Inicialmente na Seção 5.1 será feita uma análise dos resultados para cada base de dados em particular. Será analisado os resultados da métrica AUC nos experimentos, os classificadores mais selecionados, e o tempo de execução do CPLEX para a seleção dos classificadores. Por fim, na Seção 5.2 será feita uma análise geral sobre todas as bases de dados.

Um ponto a ser levantado é que na maior parte das análises feitas serão citados apenas os custos c_{vn} e c_{vp}, visto que o c_{fn} e c_{fp} são derivados destes.

5.1 Análise das bases de dados

5.1.1 Retinopatia diabética

Os resultados da métrica AUC para os 9 experimentos estão dispostos na Tabela 8. A primeira linha após o cabeçalho indica a AUC do *ensemble* completo (composto pela média de todos os classificadores usando validação cruzada com k=5). Na tabela, os experimentos onde a AUC supera a do *ensemble* completo estão em negrito. Além disso, a linha com o melhor resultado de AUC está sublinhada. A Figura 8 apresenta um gráfico referente aos valores da AUC desta tabela para facilitar a visualização.

Tabela 8 – Resultados da métrica AUC nos experimentos na base de dados de Retinopatia Diabética

Experimento	**Custos**				**AUC**	**Diferença com *ensemble* completo**
	c_{vp}	c_{fp}	c_{vn}	c_{fn}		
Ensemble completo	-	-	-	-	0,7702	-
1	0,1	-0,9	0,9	-0,1	0,7702	0,0000
2	0,2	-0,8	0,8	-0,2	0,7675	-0,0028
3	**0,3**	**-0,7**	**0,7**	**-0,3**	**0,7725**	**0,0022**
4	**0,4**	**-0,6**	**0,6**	**-0,4**	**0,7714**	**0,0012**
5	**0,5**	**-0,5**	**0,5**	**-0,5**	**0,7769**	**0,0066**
<u>6</u>	**<u>0,6</u>**	**<u>-0,4</u>**	**<u>0,4</u>**	**<u>-0,6</u>**	**<u>0,7801</u>**	**<u>0,0098</u>**
7	**0,7**	**-0,3**	**0,3**	**-0,7**	**0,7754**	**0,0051**
8	**0,8**	**-0,2**	**0,2**	**-0,8**	**0,7713**	**0,0011**
9	**0,9**	**-0,1**	**0,1**	**-0,9**	**0,7717**	**0,0014**

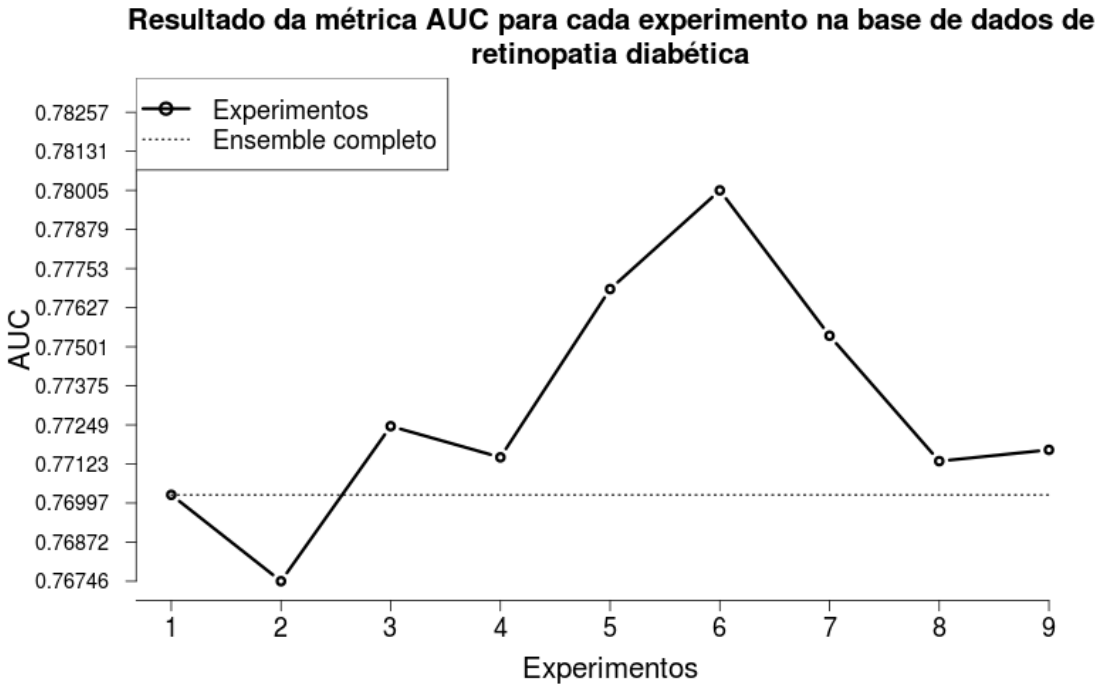

Figura 8 – Gráfico de resultados da métrica AUC nos experimentos na base de dados de retinopatia diabética.

A abordagem proposta mostrou resultados satisfatórios para esta base de dados, onde 7 dos 9 experimentos conseguiram superar o resultado de um *ensemble* completo. Um comportamento interessante que ocorreu foi que os experimentos com um c_{vp} menor que 0,3 foram os que não conseguiram resultados superiores ao *ensemble* completo. Além disso, pude perceber que existe um comportamento crescente no gráfico, até se chegar em um ponto de máximo (experimento 6), onde em seguida há um comportamento decrescente. O ponto de máximo é onde tem-se um $c_{vp} = 0{,}6$ e um $c_{vn} = 0{,}4$. Isso indica que a medida que os parâmetros tendem a esse valor, o desempenho da classificação melhora.

A Figura 9 mostra um gráfico da frequência dos classificadores mais selecionados, considerando apenas os experimentos onde a AUC superou a do *ensemble* completo.

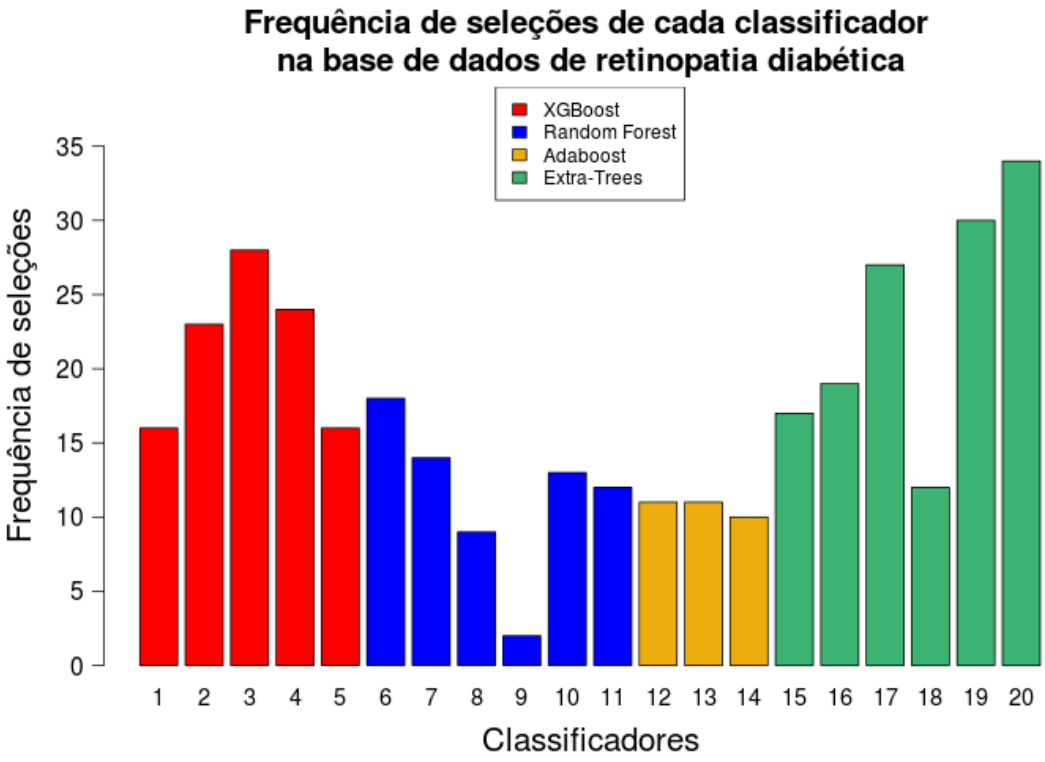

Figura 9 – Frequência de seleções de cada classificador na base de dados de retinopatia diabética

A Figura 9 mostra que nesta base de dados os 5 classificadores mais selecionados foram respectivamente um *Extra-Trees*, *Extra-Trees*, *XGBoost*, *Extra-Trees* e *XGBoost*. Os classificadores *Random Forest* e *Adaboost* não ficaram entre os 5 mais selecionados.

A seguir, na Figura 10, está um gráfico do tempo de execução do CPLEX gasto na solução do modelo proposto.

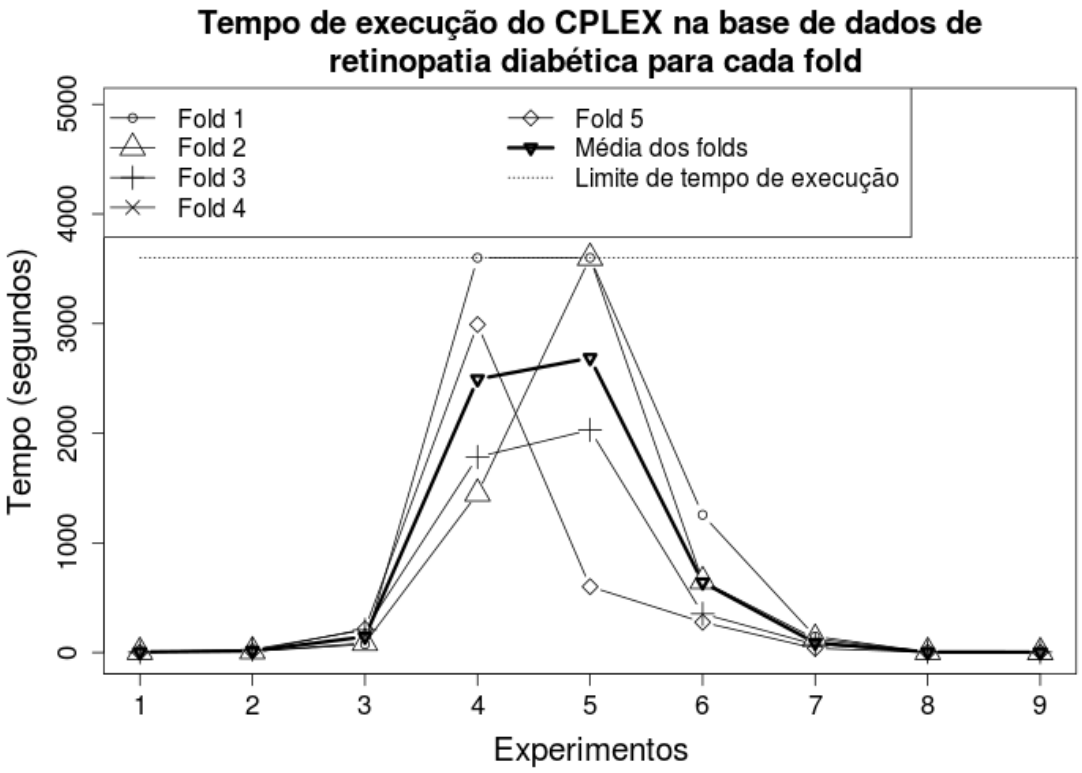

Figura 10 – Tempo de execução do CPLEX na base de dados de retinopatia diabética para cada *fold*.

De acordo com o gráfico da Figura 10, houve uma grande variação de tempo entre os experimentos: variou de poucos segundos até 3600 segundos (1 hora). Os experimentos que atingiram 3600 segundos são os que atingiram o limite máximo de tempo de execução, conseguindo uma solução que não se tem garantia de otimalidade.

Analisando se a linha do gráfico que representa a média dos *folds*, um detalhe que percebi foi que a partir do experimento 3 começa a haver um comportamento crescente (afetando todos os *folds*). Após chegar-se em um pico com o tempo máximo (experimento 5), começa a haver um comportamento decrescente até chegar no experimento 7, onde o gráfico volta a ter um comportamento estável. Reparei então que o pico de tempo máximo está justamente onde os custos c_{vp} e c_{vn} são iguais (com valor 0.5). Isso mostra indícios que para essa base de dados à medida que os custos tendem a um mesmo valor, o CPLEX encontra mais dificuldade para achar a solução ótima, aumentando o tempo de execução.

5.1.2 Biodegradação de molécula

A Tabela 9 apresenta os resultados da métrica AUC para os 9 experimentos. A Figura 11 apresenta um gráfico referente aos valores da AUC desta tabela.

Tabela 9 – Resultados da métrica AUC nos experimentos na base de dados de biodegradação de molécula.

Experimento	Custos				AUC	Diferença com *ensemble* completo
	c_{vp}	c_{fp}	c_{vn}	c_{fn}		
Ensemble completo	-	-	-	-	0,9326	-
1	**0,1**	**-0,9**	**0,9**	**-0,1**	**0,9331**	**0,0005**
2	**0,2**	**-0,8**	**0,8**	**-0,2**	**0,9349**	**0,0023**
3	**0,3**	**-0,7**	**0,7**	**-0,3**	**0,9354**	**0,0028**
4	**0,4**	**-0,6**	**0,6**	**-0,4**	**0,9348**	**0,0022**
5	**0,5**	**-0,5**	**0,5**	**-0,5**	**0,9329**	**0,0003**
6	**0,6**	**-0,4**	**0,4**	**-0,6**	**0,9334**	**0,0008**
7	**0,7**	**-0,3**	**0,3**	**-0,7**	**0,9354**	**0,0028**
8	**0,8**	**-0,2**	**0,2**	**-0,8**	**0,9338**	**0,0012**
9	0,9	-0,1	0,1	-0,9	0,9316	-0,0010

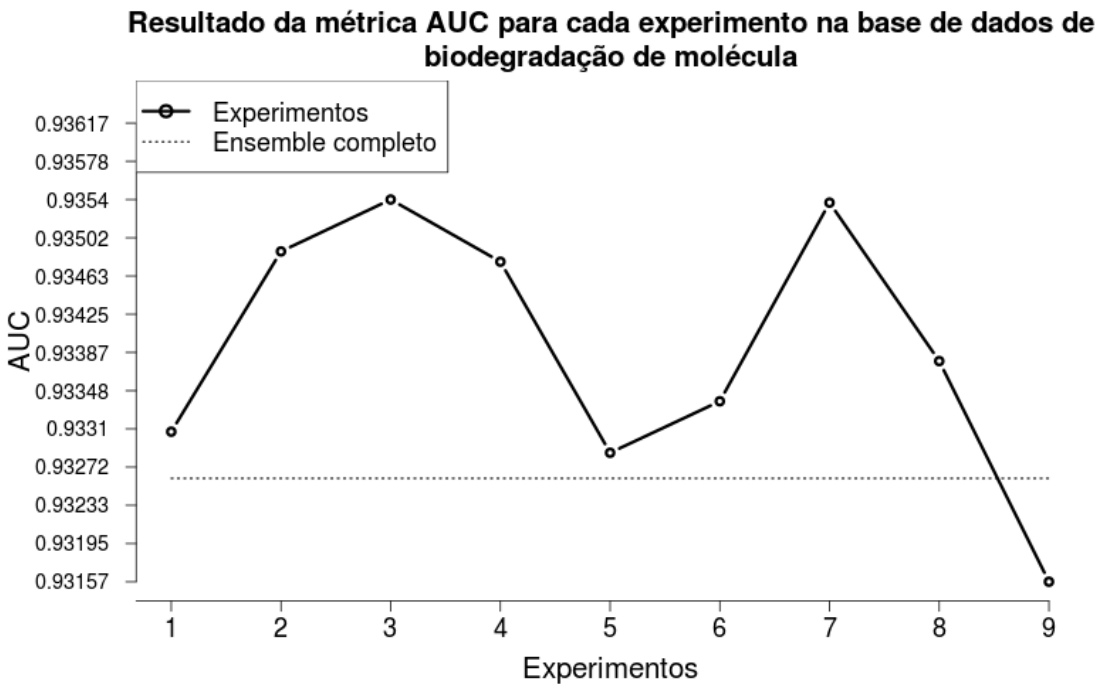

Figura 11 – Gráfico de resultados da métrica AUC nos experimentos na base de dados de biodegradação de molécula.

A maior parte dos experimentos conseguiram superar a AUC do *ensemble* completo (8 dos 9), mostrando que abordagem proposta também obteve resultados satisfatórios para esta base de dados. No gráfico podemos ver 2 picos de máximo (experimentos 3 e 7), onde cada pico segue de um comportamento crescente e é seguido de um comportamento decrescente, indicando uma certa relação com os parâmetros e o desempenho dos resultados. O melhor resultado esteve no experimento 7, com $c_{vp} = 0{,}3$ e $c_{vn} = 0{,}7$, porém sua diferença é muito pouca com o outro pico para se ter uma confiança que esta é a melhor configuração de parâmetros.

A Figura 12 apresenta um gráfico da frequência dos classificadores mais selecionados para esta base de dados, considerando apenas os experimentos onde a AUC superou a do *ensemble* completo.

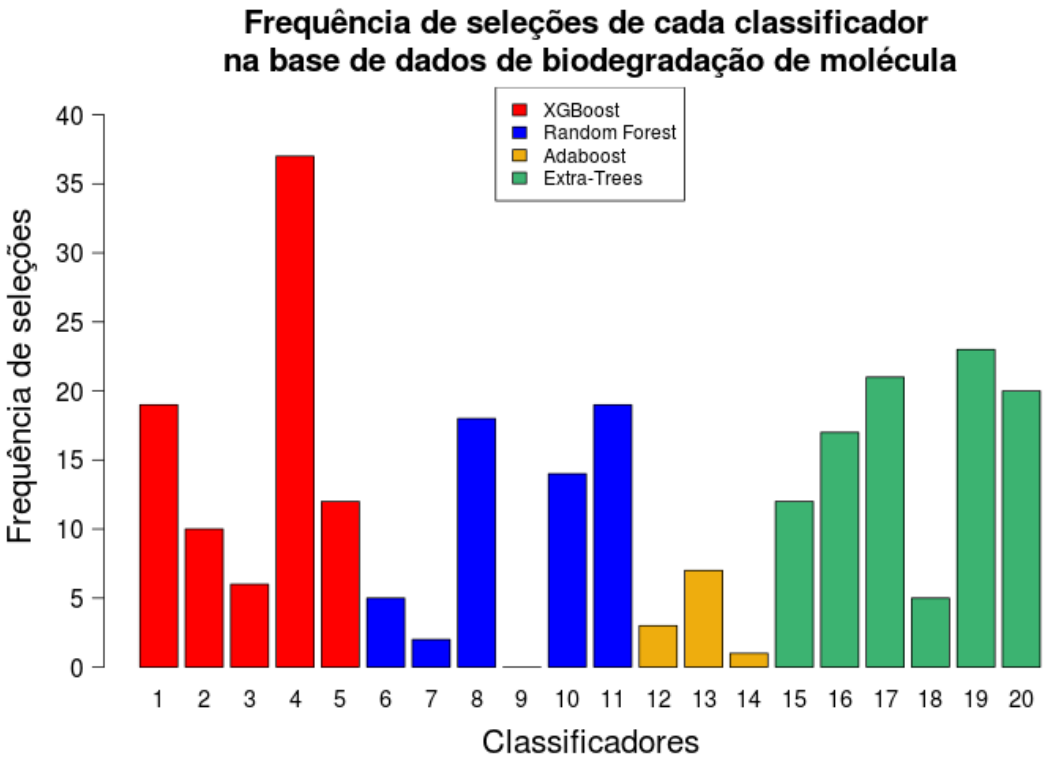

Figura 12 – Frequência de seleções de cada classificador na base de dados de biodegradação de molécula.

De acordo com a Figura 12, nesta base de dados os 5 classificadores mais selecionados foram respectivamente um *XGBoost*, *Extra-Trees*, *Extra-Trees*, *Extra-Trees* e *XGBoost* (junto com *Random Forest*, que obteve a mesma frequência), sendo que o classificador *Adaboost* não ficou entre eles.

A seguir a Figura 13 apresenta um gráfico do tempo de execução do CPLEX.

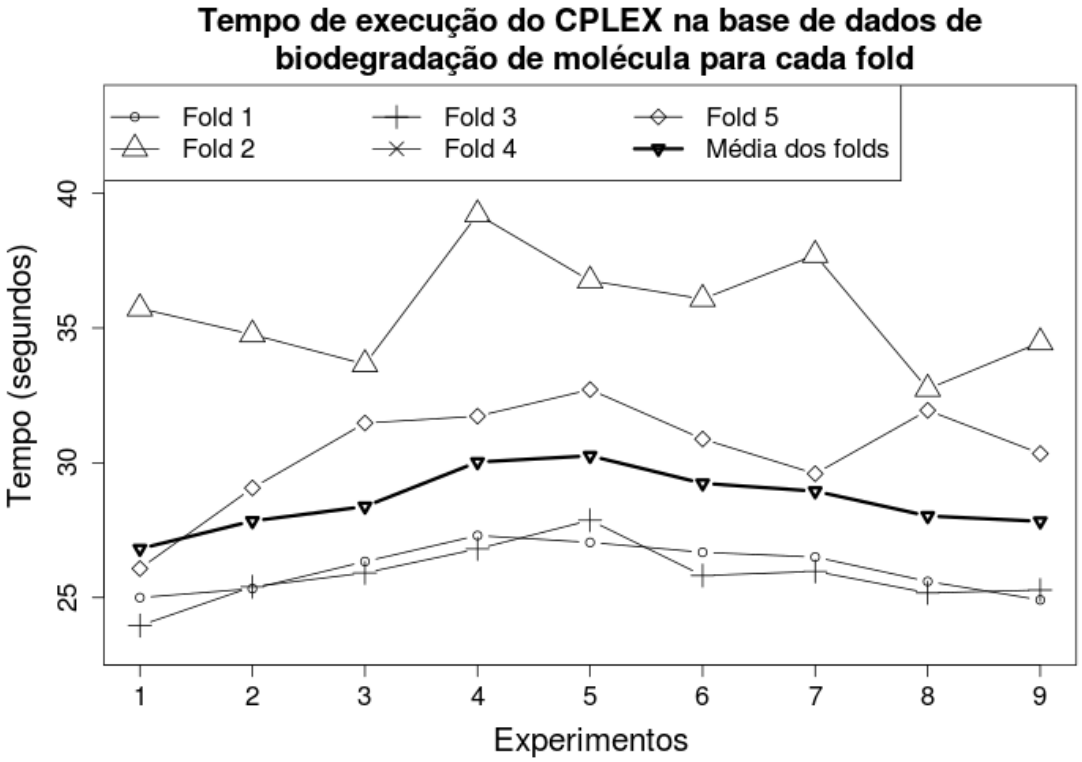

Figura 13 – Tempo de execução do CPLEX na base de dados de biodegradação de molécula para cada *fold*.

De acordo com gráfico na Figura 13, observei que o tempo de execução do CPLEX para essa base foi menor em relação a base anterior. Ele gastou entre cerca de 20 a 40 segundos, enquanto na última alguns experimentos chegaram a 1 hora, implicando que nesta nenhum experimento atingiu o limite de tempo de execução. Vale salientar que nesta base o tempo de cada *fold* nos 9 experimentos permaneceu estável, outra diferença com a última base de dados.

Analisando-se a linha da média dos *folds*, percebi que o ponto de tempo de execução máximo foi no experimento 5, onde os custos c_{vp} e c_{vn} são iguais, sugerindo que nesta base também ocorre o comportamento de que a medida que os custos tendem a um mesmo valor aumenta o tempo de execução do CPLEX, contudo neste caso a variação sendo pequena. Na Seção 5.2 será dada uma possível explicação para este comportamento.

5.1.3 *Spam*

A Tabela 10 apresenta os resultados da métrica AUC para os 9 experimentos e a Figura 14 apresenta um gráfico referente aos valores da AUC desta tabela.

Tabela 10 – Resultados da métrica AUC nos experimentos na base de dados de *spam*.

Experimento	Custos				AUC	Diferença com *ensemble* completo
	c_{vp}	c_{fp}	c_{vn}	c_{fn}		
Ensemble completo	-	-	-	-	0,9885	-
1	**0,1**	**-0,9**	**0,9**	**-0,1**	**0,9897**	**0,0011**
2	**0,2**	**-0,8**	**0,8**	**-0,2**	**0,9898**	**0,0013**
3	**0,3**	**-0,7**	**0,7**	**-0,3**	**0,9900**	**0,0015**
4	**0,4**	**-0,6**	**0,6**	**-0,4**	**0,9898**	**0,0013**
5	**0,5**	**-0,5**	**0,5**	**-0,5**	**0,9896**	**0,0011**
6	**0,6**	**-0,4**	**0,4**	**-0,6**	**0,9896**	**0,0011**
7	**0,7**	**-0,3**	**0,3**	**-0,7**	**0,9896**	**0,0011**
8	**0,8**	**-0,2**	**0,2**	**-0,8**	**0,9898**	**0,0012**
9	**0,9**	**-0,1**	**0,1**	**-0,9**	**0,9898**	**0,0012**

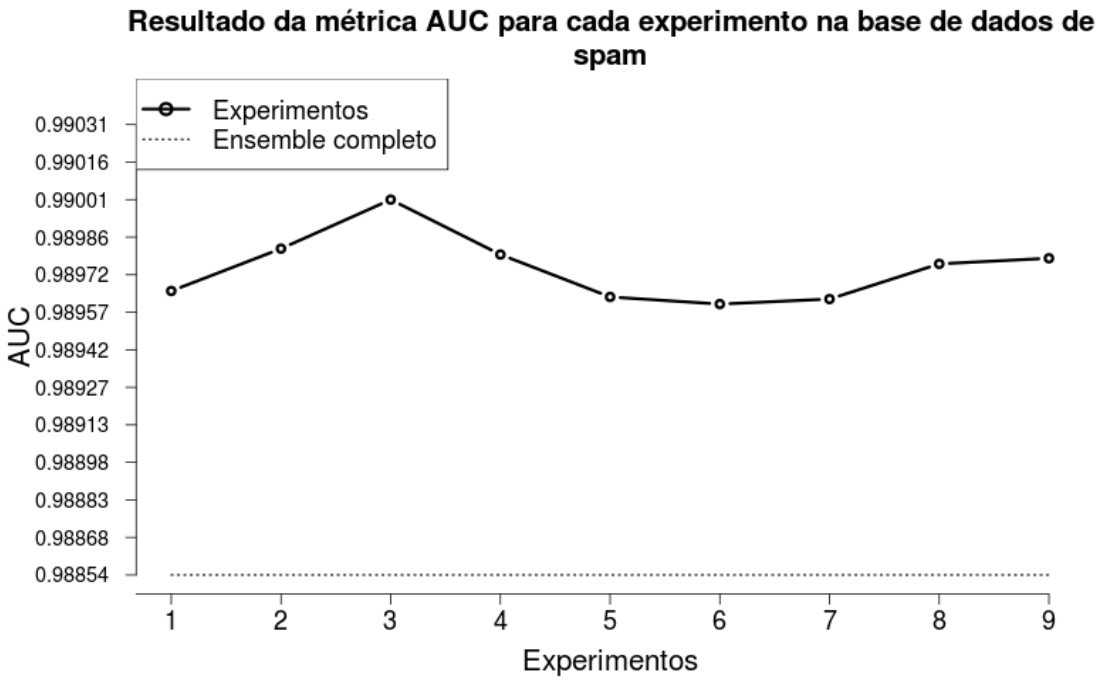

Figura 14 – Gráfico de resultados da métrica AUC nos experimentos na base de dados de *spam*.

Nesta base de dados, todos os experimentos conseguiram superar a AUC do *ensemble* completo. No gráfico da Figura 14 podemos ver 1 pico de máximo no experimento 3, com comportamentos crescentes e decrescentes respectivamente antes e depois dos picos. O experimento 3 teve $c_{vp} = 0{,}3$ e $c_{vn} = 0{,}7$.

A Figura 15 apresenta um gráfico da frequência dos classificadores mais selecionados para esta base de dados, considerando apenas os experimentos onde a AUC superou a do *ensemble* completo.

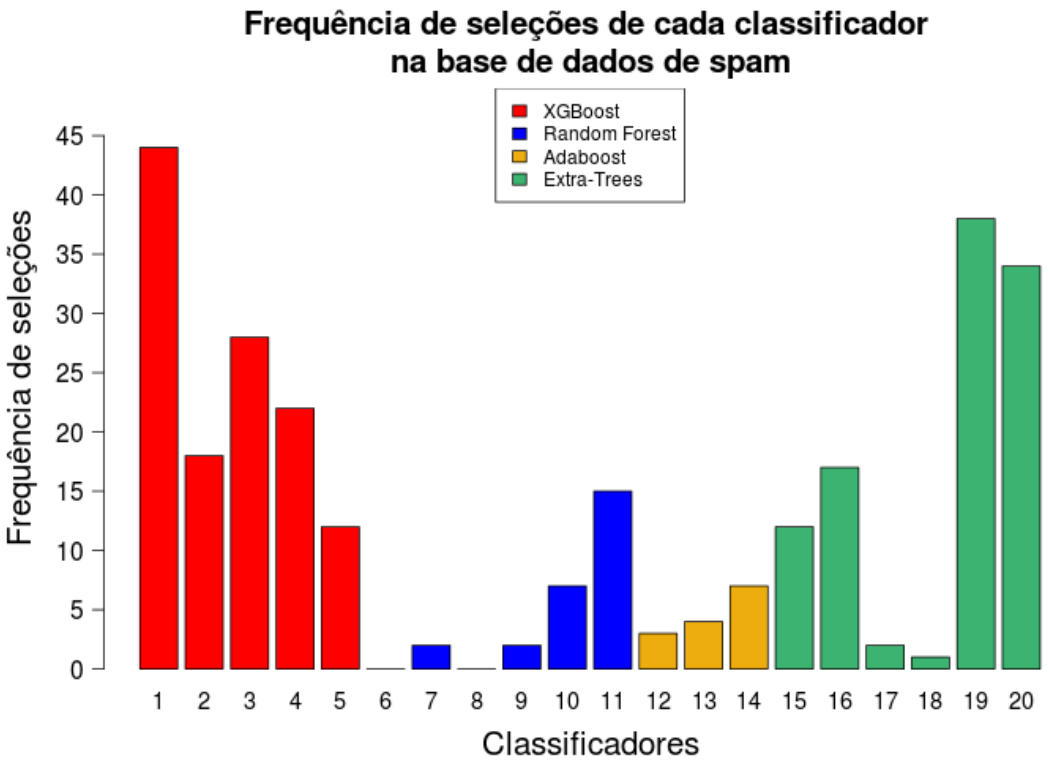

Figura 15 – Frequência de seleções de cada classificador na base de dados de *spam*.

Conforme a Figura 15, os 5 classificadores mais selecionados nesta base de dados foram respectivamente um *XGBoost, Extra-Trees, Extra-Trees, XGBoost e XGBoost.* Os classificadores *Random Forest, Adaboost* não ficaram entre os 5 mais selecionados.

A seguir a Figura 16 apresenta um gráfico do tempo de execução do CPLEX.

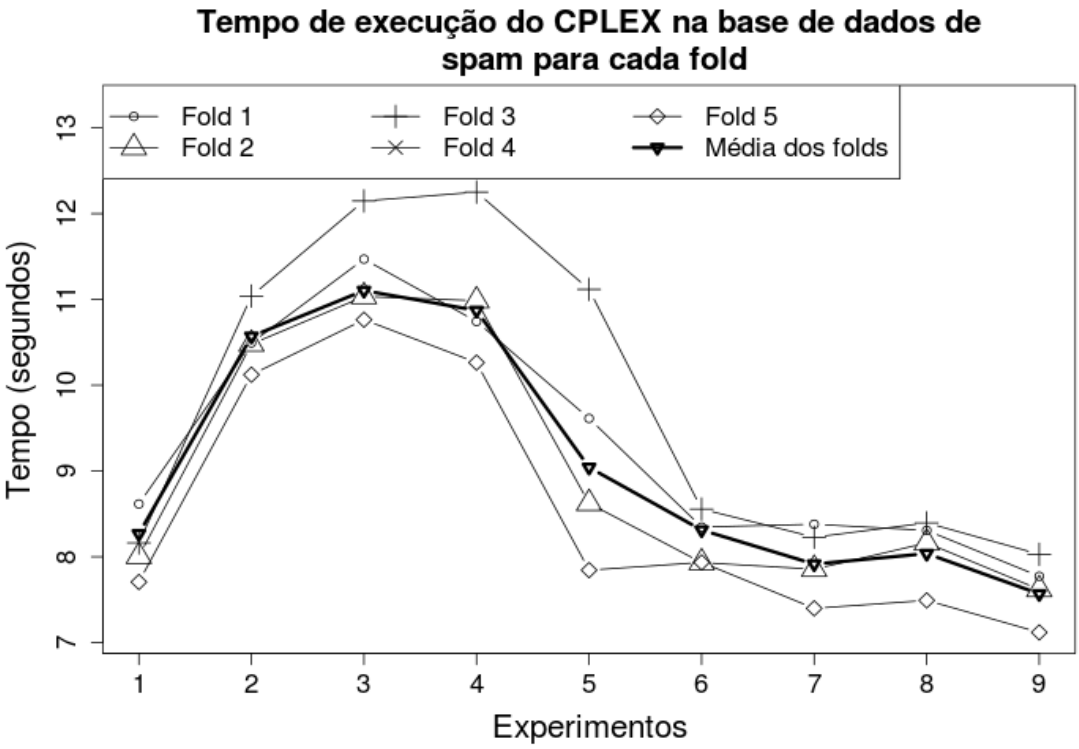

Figura 16 – Tempo de execução do CPLEX na base de dados de *spam* para cada *fold*.

De acordo com gráfico na Figura 16, notei que em média o tempo de execução do CPLEX para essa base foi menor do que o da base anterior, porém ainda continuando na casa dos segundos. Ele gastou entre cerca de 7 a 12 segundos, também não havendo nenhum experimento que atingiu o limite de tempo de execução.

Analisando a linha da média dos *folds*, observei que esta base de dados não apresentou o comportamento das anteriores, onde a medida que os valores dos custos c_{vp} e c_{vn} tendiam a um mesmo valor o tempo de execução aumentava. Nela o pico de tempo de execução foi no experimento 3 ($c_{vp} = 0{,}3$ e $c_{vn} = 0{,}7$), que concidentemente foi o mesmo experimento que conseguiu o melhor resultado de AUC.

5.1.4 Câncer cervical

A Tabela 11 apresenta os resultados da métrica AUC para os 11 experimentos e a Figura 17 apresenta um gráfico referente aos valores da AUC desta tabela.

Tabela 11 – Resultados da métrica AUC nos experimentos na base de dados de câncer cervical.

Experimento	**Custos**				**AUC**	**Diferença com *ensemble* completo**
	c_{vp}	c_{fp}	c_{vn}	c_{fn}		
Ensemble completo	-	-	-	-	0,9597	-
1	0,1	-0,9	0,9	-0,1	0,9479	-0,0118
2	0,2	-0,8	0,8	-0,2	0,9564	-0,0033
3	0,3	-0,7	0,7	-0,3	0,9582	-0,0015
4	0,4	-0,6	0,6	-0,4	0,9590	-0,0007
5	0,5	-0,5	0,5	-0,5	0,9591	-0,0006
6	0,6	-0,4	0,4	-0,6	0,9590	-0,0007
7	0,7	-0,3	0,3	-0,7	0,9555	-0,0042
8	**0,8**	**-0,2**	**0,2**	**-0,8**	**0,9668**	**0,0072**
9	**0,9**	**-0,1**	**0,1**	**-0,9**	**0,9717**	**0,0120**
10	0,01	-0,99	0,99	-0,01	0,9579	-0,0018
11	**0,99**	**-0,01**	**0,01**	**-0,99**	**0,9723**	**0,0126**

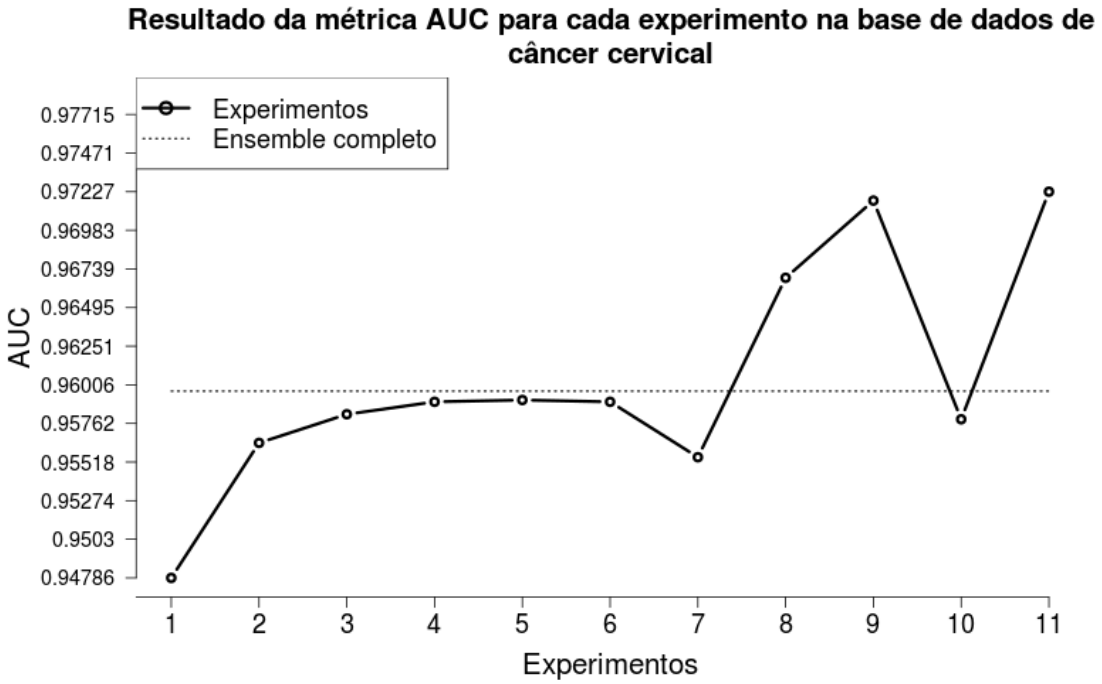

Figura 17 – Gráfico de resultados da métrica AUC nos experimentos na base de dados de câncer cervical.

Nesta base de dados os experimentos obtiveram um desempenho pior do que os anteriores: apenas 3 experimentos conseguiram superar a AUC do *ensemble* completo. Isso provavelmente se deve ao grau elevado de desbalanceamento desta base, que fez apenas com que experimentos com um c_{vp} alto (0.8,0.9,0.99) conseguissem superar a AUC do *ensemble* completo. Observe que esses custos tendem ao oposto da distribuição das classes, ou seja, um c_{vp} alto enquanto a distribuição de classe 1 (Verdadeira) é baixa (6,58%). O raciocínio é equivalente para a classe 0 (Negativa).

A Figura 18 apresenta um gráfico da frequência dos classificadores mais selecionados para esta base de dados, considerando apenas os experimentos onde a AUC superou a do *ensemble* completo.

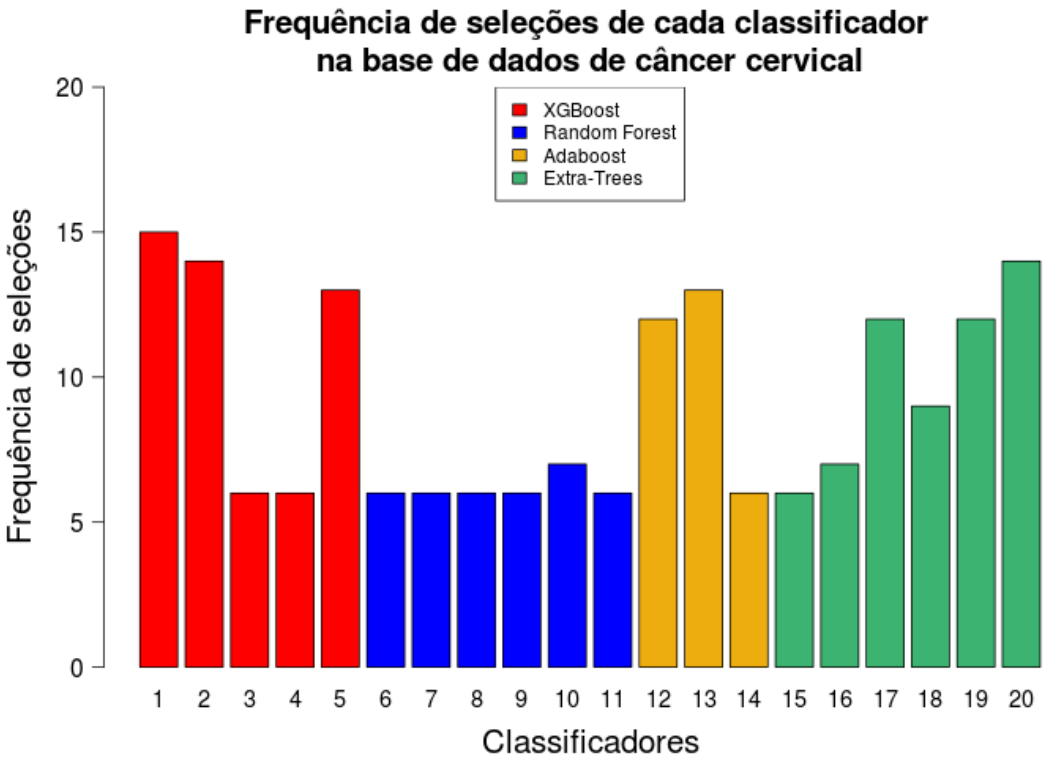

Figura 18 – Frequência de seleções de cada classificador na base de dados de câncer cervical.

De acordo com a Figura 18, os 5 classificadores mais selecionados nesta base de dados foram respectivamente um *XGBoost*, *XGBoost*, *Extra-Trees*, *XGBoost* e *Random Forest.* Mais uma vez o classificador *Adaboost* não ficou entre os 5 mais selecionados.

A seguir a Figura 19 apresenta um gráfico do tempo de execução do CPLEX.

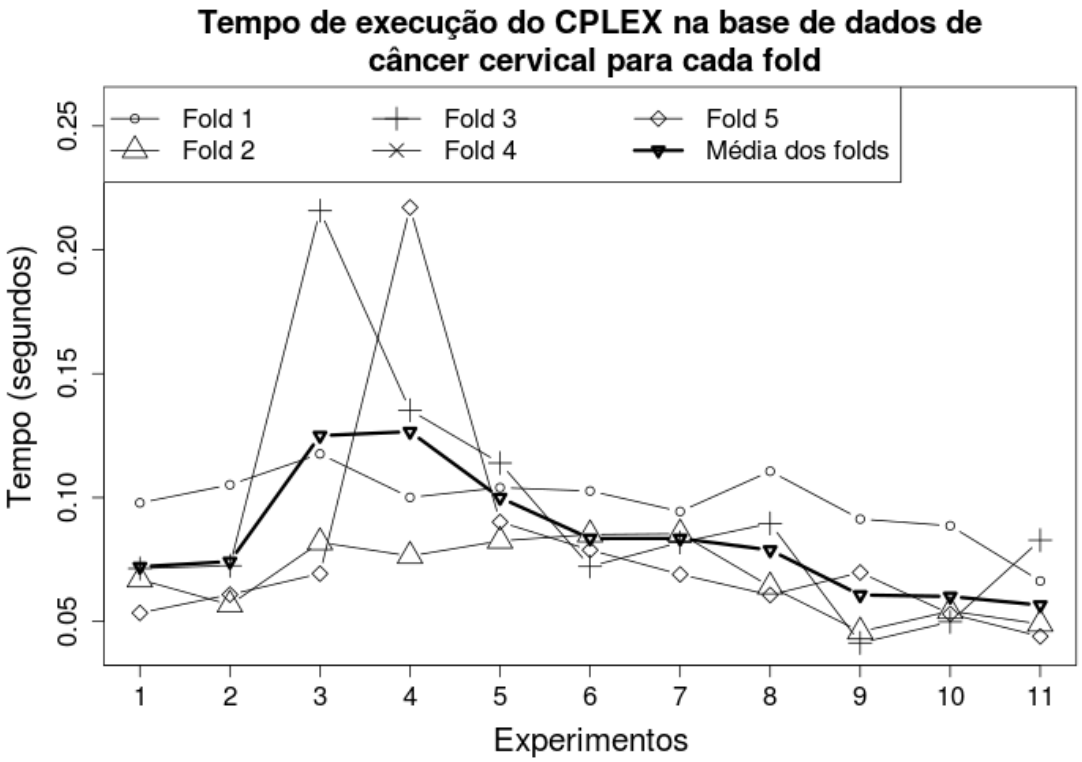

Figura 19 – Tempo de execução do CPLEX na base de dados de câncer cervical para cada *fold*.

Analisando o gráfico na Figura 19, pude perceber que o tempo de execução nesta base de dados foi menor do que as anteriores, sendo sempre menor que 1 segundo. Ele varia de cerca de 0.03 a 0.23 segundos.

De acordo com a linha da média dos *folds*, percebi que esta base também não obteve o tempo máximo onde o c_{vp} e c_{vn} tendiam a um mesmo valor, sendo o ponto de tempo máximo no experimento 4 ($c_{vp} = 0{,}4$ e $c_{vn} = 0{,}6$). Ele obteve uma pequena diferença do experimento 3 ($c_{vp} = 0{,}3$ e $c_{vn} = 0{,}7$).

5.1.5 Solavancos sísmicos

A Tabela 12 apresenta os resultados da métrica AUC para os 11 experimentos, onde o experimento com maior AUC está sublinhado. A Figura 20 apresenta um gráfico referente aos valores da AUC desta tabela.

Tabela 12 – Resultados da métrica AUC nos experimentos na base de dados de solavancos sísmicos.

Experimento	Custos				AUC	Diferença com *ensemble* completo
	c_{vp}	c_{fp}	c_{vn}	c_{fn}		
Ensemble completo	-	-	-	-	0,7668	-
1	0,1	-0,9	0,9	-0,1	0,7511	-0,0157
2	0,2	-0,8	0,8	-0,2	0,7529	-0,0139
3	0,3	-0,7	0,7	-0,3	0,7539	-0,0129
4	0,4	-0,6	0,6	-0,4	0,7572	-0,0096
5	0,5	-0,5	0,5	-0,5	0,7514	-0,0155
6	0,6	-0,4	0,4	-0,6	0,7521	-0,0147
7	0,7	-0,3	0,3	-0,7	0,7534	-0,0134
8	0,8	-0,2	0,2	-0,8	0,7461	-0,0207
9	0,9	-0,1	0,1	-0,9	0,7498	-0,0170
10	0,01	-0,99	0,99	-0,01	0,7546	-0,0122
11	0,99	-0,01	0,01	-0,99	0,7469	-0,0199

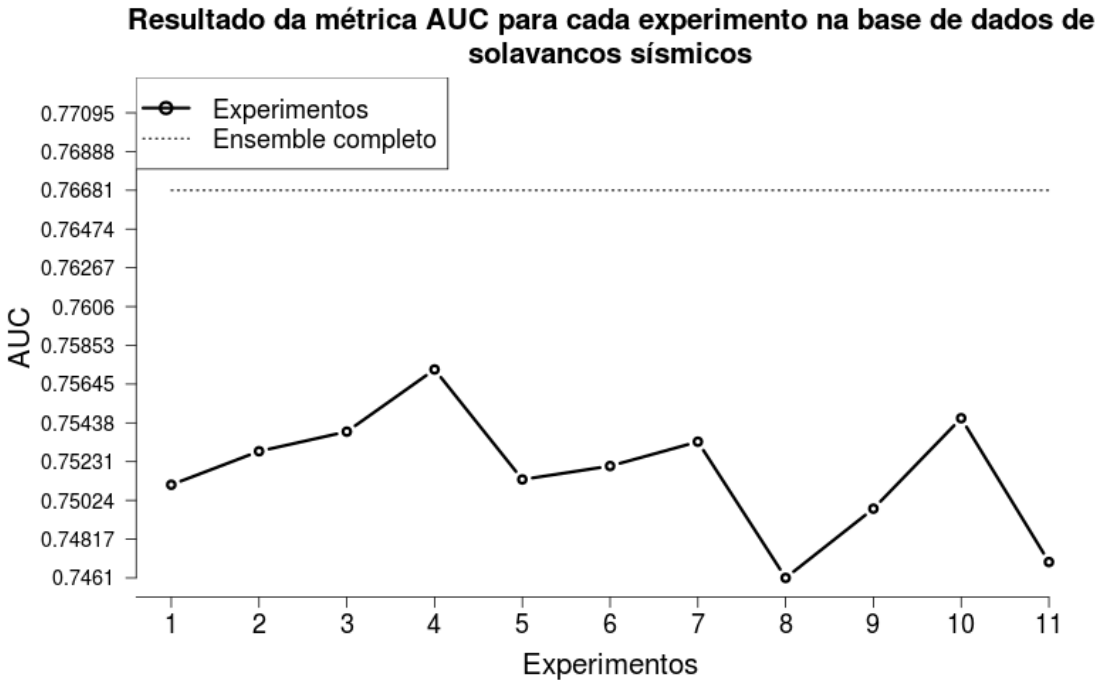

Figura 20 – Gráfico de resultados da métrica AUC nos experimentos na base de dados de solavancos sísmicos.

Esta foi a base de dados onde os experimentos obtiveram o pior desempenho: todos os 11 experimentos conseguiram resultados piores que a AUC do *ensemble* completo, sendo que nem os custos mais extremos (experimentos 10 e 11) conseguiram melhorar os resultados. Um dos fatores que provavelmente contribuiu para isso acontecer foi o grau elevado de desbalanceamento da base, que já havia mostrado indícios de afetar a última base.

Esta base teve 3 picos de máximo, onde o melhor deles esteve no experimento 4 ($c_{vp} = 0{,}4$ e $c_{vn} = 0{,}6$). Mesmo sendo o melhor experimento, ainda ficou com uma diferença considerável do *ensemble* completo (-0,0096).

Pelo fato de nenhum experimento ter sido melhor que o *ensemble* completo, não será apresentado o gráfico de frequência de seleções de classificadores.

A seguir a Figura 21 apresenta um gráfico do tempo de execução do CPLEX.

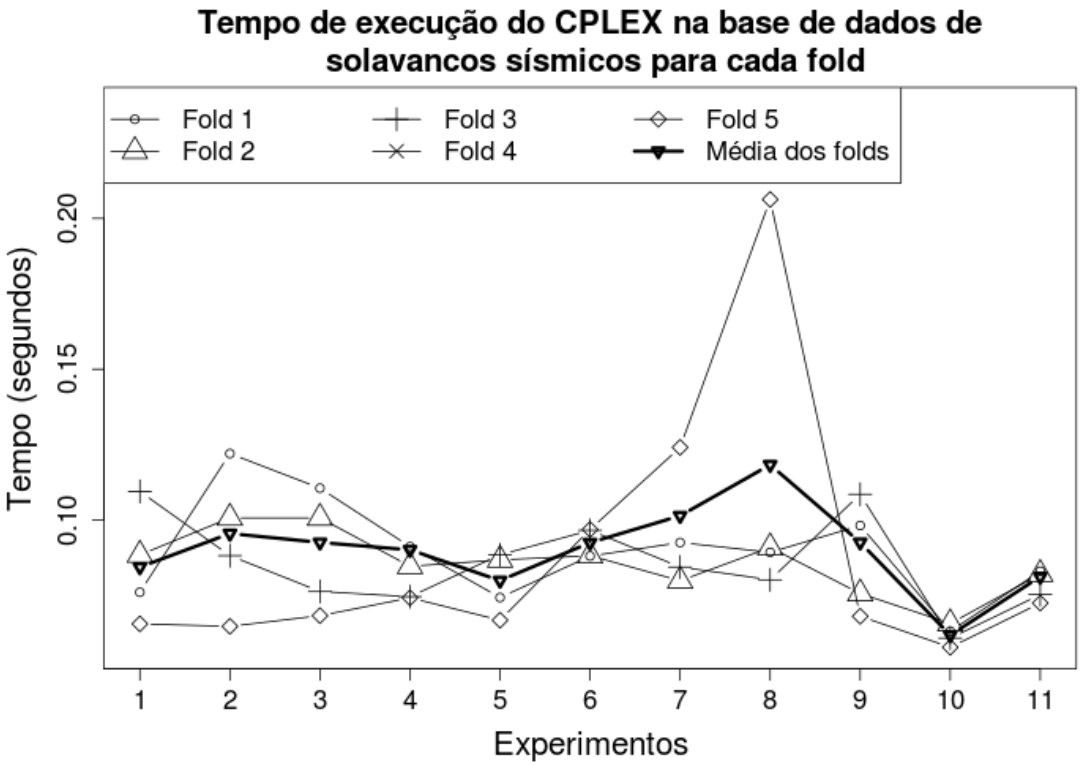

Figura 21 – Tempo de execução do CPLEX na base de dados de solavancos sísmicos para cada *fold*.

De acordo com o gráfico da Figura 21, pude perceber que o tempo de execução nesta base foi semelhante a base de dados anterior, variando de cerca de 0.05 a 0.20 segundos enquanto da anterior variou em cerca de 0.03 a 0.23 segundos.

De acordo com a linha da média dos *folds*, o ponto de tempo máximo esteve no experimento 8 ($c_{vp} = 0{,}8$ e $c_{vn} = 0{,}2$).

5.2 Análise geral

Nesta Seção será feita uma análise geral sobre todas as bases. Na Subseção 5.2.1 a análise será sobre a relação da escolha dos parâmetros e o desempenho da abordagem proposta e na 5.2.2 será feita uma análise sobre o tempo de execução do CPLEX nas bases de dados.

Vale salientar que o número de bases testadas foi pequeno, logo para se ter um maior grau de confiança nas análises feitas seria necessário testar com um número maior de bases de dados.

5.2.1 Influência dos custos no desempenho dos experimentos

A Tabela 13 apresenta os custos de c_{vn} e c_{vp} do experimento que obteve o melhor desempenho de AUC para as 5 bases de dados, além de alguns fatores em que se acredita ter alguma relação com eles.

Tabela 13 – Fatores de influência na escolha dos custos

Base de dados	Grau de desbalanceamento	AUC média de todos os experimentos	Distribuição das classes		Parâmetros de custo do experimento de melhor desempenho	
			Classe 0	Classe 1	c_{vn}	c_{vp}
1	Baixo	0,7730	46,92%	53,08%	0,4	0,6
2	Médio	0,9339	33,74%	66,26%	0,7	0,3
3	Médio	0,9897	60,60%	39,40%	0,7	0,3
4	Alto	0,9603	93,59%	6,41%	0,01	0,99
5	Alto	0,7518	93,42%	6,58%	0,6	0,4

De acordo com a análise realizada, percebi que diferentes comportamentos ocorrem para diferentes graus de desbalanceamento. Por esse motivo a análise será feita inicialmente para as bases com grau de desbalanceamento baixo e médio (bases 1,2 e 3) e em seguida para as bases de grau alto (4 e 5).

Nas bases 1, 2 e 3 notei que os custos c_{vn} e c_{vp} de certa forma são relativos a distribuição de suas respectivas classes: Na Base 1, ao se truncar a distribuição da classe 0 (46,92% truncado para 40%), observei um valor proporcional ao $c_{vn} = 0{,}4$. Consequentemente o mesmo acontece com a classe 1.

Na base 2, têm-se o de certa forma um comportamento oposto. O custo do c_{vn} é proporcional ao de sua classe oposta, a classe negativa. Contudo ao se analisar a Figura 11 da Subseção 5.1.2, observei que o segundo experimento de melhor valor de AUC também forma um pico no gráfico, e ele possui uma diferença muito pequena do melhor experimento. Já este segundo experimento possui comportamento equivalente ao da base 1, ou seja, do custo ser proporcional a distribuição da classe (e não o oposto dele).

Já na base 3, o comportamento volta a ser semelhante ao da base 1, porém ao invés de se truncar a classe 0 teria que arredondar (60,60% arredondado para 70%).

Na base 4 os custos também tendem a ser próximos ao oposto da distribuição da classe, semelhante a base 2: $c_{vp} = 0{,}99$ tende a ser próximo de 93,59% e $c_{vn} = 0{,}01$ tende a ser próximo de 6,41%.

Por último, mais uma vez a base 5 se mostrou diferente das outras classes, não mostrando nenhum comportamento de proporcionalidade, direto ou oposto, com a distribuição das classes.

Com base no que foi exposto nesta Seção, cheguei a conclusão de que como parâmetros mais prováveis de se maximizar o desempenho deste modelo, o mais indicado seria tentar valores proporcionalmente próximos da distribuição da classe, ou proporcionalmente opostos a ela. No capítulo 6 será proposto como trabalho futuro uma abordagem para a previsão dos parâmetros de custos mais prováveis a maximizar a AUC de uma dada base de dados.

5.2.2 Tempo de execução

Inicialmente, pensei que o número de classificadores gerados na etapa de geração, o número de padrões da base e o valor dos custos seriam os parâmetros mais impactantes no tempo de execução do CPLEX. Porém após a execução dos experimentos, observei indícios de dois fatores que, pelo menos para as bases de dados estudadas, contribuíram mais para o aumento desse tempo: O grau de desbalanceamento da base e a AUC média dela. Observe a Tabela 14.

Tabela 14 – Fatores de influência no tempo de execução médio

Base de dados	Número de padrões	Grau de desbalanceamento	AUC média de todos os experimentos	Tempo de execução médio de todos os experimentos (segundos)
1	1151	Baixo	0,7730	2686,2520
2	1055	Médio	0,9339	30,2592
3	4601	Médio	0,9897	11,1037
4	858	Alto	0,9603	0,1266
5	2584	Alto	0,7518	0,1183

A AUC média foi calculada fazendo-se a média da AUC de todos os experimentos de uma base de dados. Já para o tempo de execução médio inicialmente foi calculada a média de todos os *folds* em cada experimento, e em seguida foi calculada a média dessa média de todos experimentos. No caso do tempo, existe um desvio padrão grande ao calcular essas médias, na análise feita a seguir será comentado sobre isso.

Ao analisar os dados da Tabela 14, inicialmente percebi que não existe correlação entre o número de padrões e o tempo de execução, pois a base de dados 3 possui 4601 padrões e um tempo médio de 11 segundos, enquanto a base 1 possui 1151 padrões e um tempo médio de cerca de 2686 segundos, que é consideravelmente maior. Contudo, observei a seguinte relação: Para as bases que tem um grau de desbalanceamento baixo e médio (1,2 e 3), existe 100% de

correlação de sua AUC média com o tempo médio de execução: quanto menor sua AUC média, mais tempo o CPLEX levou para achar a solução ótima.

Para essas três primeiras bases, considerou-se que o desvio padrão alto da média do tempo de execução não afetou a análise, pois existe uma diferença considerável de tempo de execução entre cada uma delas.

Considerando agora as bases com grau de desbalanceamento alto (bases 4 e 5), notei que o tempo de execução médio delas foi bem baixo. A conclusão que cheguei aqui é que quando o grau de desbalanceamento é alto, o tempo de execução tende a ser menor, pois como existem menos padrões de uma classe, será mais fácil para o CPLEX decidir se o padrão pertencerá a essa classe ou a outra classe.

Outro detalhe que percebi, ainda nas bases 4 e 5, foi que aparentemente a AUC média não se mostrou correlacionada com o tempo de execução, pois apesar de haver uma diferença considerável de AUC entre as duas bases, o tempo médio de execução das duas se mostrou bem próximo.

A seguir será feita uma breve análise sobre a relação do valores dos parâmetros c_{vp} e c_{vn} com o tempo médio de execução. A análise será feita com base na Tabela 15.

Tabela 15 – Influência dos parâmetros c_{vp} e c_{vn} no experimento de maior tempo de execução.

Base de dados	Grau de desbalanceamento	AUC média de todos os experimentos	Parâmetros de custo do experimento de maior de tempo	
			c_{vp}	c_{vn}
1	Baixo	0,7730	0,5	0,5
2	Médio	0,9339	0,5	0,5
3	Médio	0,9897	0,3	0,7
4	Alto	0,9603	0,99	0,01
5	Alto	0,7518	0,8	0,2

Nesta análise, observei apenas uma relação dos parâmetros com o tempo de execução nas duas primeiras bases: Como já comentado nas seções 4.4.1 e 4.4.2, para essas duas bases de dados, a medida que o custo tendeu a ser igual (0,5), o tempo de execução tendeu a aumentar.

É possível que esse comportamento não tenha se repetido para a base 3 devido a sua AUC média (que já se mostrou correlacionada com o tempo de execução) alta, e nas bases 4 e 5 por elas possuírem um grau de desbalanceamento alto.

6 Conclusões e trabalhos futuros

Este trabalho apresentou uma técnica baseada em MCS's, que usa um modelo de programação inteira mista para a seleção de classificadores em um *ensemble*. Ela conseguiu um melhor desempenho em relação ao *ensemble* sem o uso de seleção em 4 de 5 bases testadas, mostrando que com esta técnica é possível criar *ensembles* eficientes na classificação de padrões. Com isso, o objetivo central deste trabalho foi realizado.

Com as análises realizadas no capítulo 5, concluiu-se que o desempenho do modelo foi afetado pelos parâmetros de custo definidos em cada base de dados: Os parâmetros que maximizaram o desempenho do modelo mostraram ter relação direta ou oposta com a distribuição das classes, dependendo da base em questão.

Essa relação indica a possibilidade de um trabalho futuro: a criação de um modelo de aprendizado de máquina para a previsão de quais seriam os custos mais prováveis a maximizar o desempenho em uma determinada base de dados. Para a criação deste modelo, seria necessário inicialmente rodar experimentos com diversas bases de dados, salvando informações como distribuição das classes, grau de desbalanceamento e AUC média. Essas informações seriam as características presentes nos padrões usados nesse modelo.

Um outro trabalho interessante a ser proposto seria em relação a granularidade dos parâmetros de custos: Ao invés de se utilizar um passo de 0,1 entre eles, usar um passo em uma granularidade menor para se ter uma ideia do desempenho do modelo nesses pontos onde os custos não foram testados. Além disso, outras formas para a definição dos custos poderiam ser testadas, não se limitando a maneira proposta neste trabalho para a definição deles.

Além dos parâmetros de custo, o grau de desbalanceamento mostrou afetar o desempenho geral do modelo nas bases de dados: A maior parte dos experimentos feitos em bases com graus baixo e médio de desbalanceamento (bases 1, 2 e 3) mostraram superar o desempenho do *ensemble* completo, enquanto os experimentos feitos em bases com grau elevado de desbalanceamento mostraram dificuldade em melhorar esse resultado (base 4) ou realmente não conseguiram melhorar ele (base 5).

Em busca de melhorar o desempenho da abordagem criada, além da previsão dos custos ideais, poderia se desenvolver os seguintes trabalhos futuros:

- Modificar a abordagem criada para, ao invés dela usar modelo proposto em uma estratégia de seleção estática, ser usada em uma estratégia de seleção dinâmica.
- Desenvolver variações do modelo proposto em busca de um que apresente melhores resultados.

Se tratando agora do tempo de execução, ele se mostrou ter relação principalmente com AUC média dos experimentos e o grau de desbalanceamento das bases. Para as bases de baixo e médio desbalanceamento, a AUC média se mostrou 100% correlacionada ao tempo de execução, sendo que quanto menor o valor dela maior é o tempo de execução. Já para bases de grau elevado de desbalanceamento, concluiu-se que independente da AUC média elas possuem um tempo de execução baixo.

A relação citada acima indica a possibilidade do uso de aprendizado de máquina para estimar o tempo de execução médio do modelo em uma determinada base de dados. Essa estimativa poderia ser criada com uma modificação do modelo de aprendizado de máquina de previsão de custos citado acima como trabalho futuro para, ao invés de prever custos, prever o tempo de execução médio.

Com o objetivo de diminuir o tempo de execução, que afetou principalmente os experimentos na base de dados 1, propõe-se dois trabalhos futuros:

- Definição de uma heurística para determinar uma solução inicial para o CPLEX, que seja mais próxima da solução ótima do que a definida por ele. Além de definir a solução inicial, a heurística criada também poderia ser executada durante a execução do CPLEX, tentando melhorar a solução encontrada por ele em um dado momento de sua execução.
- Definição de uma heurística para aumentar a velocidade em que o CPLEX prova que uma solução é ótima, pois notou-se em alguns experimentos que mesmo quando a solução ótima é encontrada, ele ainda acaba demorando um tempo para provar que ela realmente é ótima.

Vale salientar que uma tarefa importante a ser realizada nos trabalhos futuros propostos seria o desenvolvimento de testes com uma maior quantidade de bases de dados, de diferentes números de padrões, números de classificadores gerados no *ensemble*, graus de desbalanceamento e valores médios de AUC, com o objetivo de se tirar conclusões com uma maior confiabilidade estatística.

Pretende-se desenvolver os trabalhos futuros citados neste capítulo em um futuro mestrado.

7 Referências

ANTAL, B.; HAJDU, A. An ensemble-based system for automatic screening of diabetic retinopathy. **Knowledge-Based Systems**, v. 60, p. 20–27, 2014.

ARENALES, M. et al. **Pesquisa Operacional para Cursos de Engenharia**. [s.l: s.n.].

BASHBAGHI, S. et al. **Dynamic Selection of Exemplar-SVMs for Watch-list Screening through Domain Adaptation**. Proceedings of the 6th International Conference on Pattern Recognition Applications and Methods. **Anais**...SCITEPRESS - Science and Technology Publications, 2017a

BASHBAGHI, S. et al. Robust watch-list screening using dynamic ensembles of SVMs based on multiple face representations. **Machine Vision and Applications**, v. 28, n. 1–2, p. 219–241, 2017b.

BREIMAN, L. Bagging predictors. **Machine Learning**, v. 24, n. 2, p. 123–140, 1996.

BREIMAN, L. Random Forests. **Machine Learning**, v. 45, n. 1, p. 5–32, 2001.

BRITTO, A. S.; SABOURIN, R.; OLIVEIRA, L. E. S. Dynamic selection of classifiers - A comprehensive review. **Pattern Recognition**, v. 47, n. 11, p. 3665–3680, 2014.

BROWN, G. et al. Diversity creation methods: A survey and categorisation. **Information Fusion**, v. 6, n. 1, p. 5–20, 2005.

CAVALIN, P. R.; SABOURIN, R.; SUEN, C. Y. **LoGID: An adaptive framework combining local and global incremental learning for dynamic selection of ensembles of HMMs**. Pattern Recognition. **Anais**...2012

CAVALIN, P. R.; SABOURIN, R.; SUEN, C. Y. Dynamic selection approaches for multiple classifier systems. **Neural Computing and Applications**, v. 22, n. 3–4, p. 673–688, 2013.

CEVIKALP, H.; POLIKAR, R. Local classifier weighting by quadratic programming. **IEEE Transactions on Neural Networks**, v. 19, n. 10, p. 1832–1838, 2008.

CHEN, T.; GUESTRIN, C. **XGBoost**. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD ’16. **Anais**...2016Disponível em: <http://dl.acm.org/citation.cfm?doid=2939672.2939785>

CRUZ, R. M. O.; SABOURIN, R.; CAVALCANTI, G. D. C. Analyzing different prototype selection techniques for dynamic classifier and ensemble selection. **Proceedings of the International Joint Conference on Neural Networks**, v. 2017–May, p. 3959–3966, 2017.

CRUZ, R. M. O.; SABOURIN, R.; CAVALCANTI, G. D. C. Dynamic classifier selection: Recent advances and perspectives. **Information Fusion**, v. 41, n. May, p. 195–216, 2018.

DI NUCCI, D. et al. Dynamic Selection of Classifiers in Bug Prediction: An Adaptive Method. **IEEE Transactions on Emerging Topics in Computational Intelligence**, v. 1, n. 3, p. 202–212, jun. 2017.

DUIN, R. P. W. The combining classifier: to train or not to train? **Object recognition**

supported by user interaction for service robots, v. 2, n. c, p. 765–770, 2002.

FAWCETT, T. An introduction to ROC analysis. **Pattern Recognition Letters**, v. 27, n. 8, p. 861–874, 2006.

FENG, J. et al. Boosting and margin theory. **Frontiers of Electrical and Electronic Engineering in China**, v. 7, n. 1, p. 127–133, 2012.

FREUND, Y.; SCHAPIRE, R. E. Experiments with a New Boosting Algorithm. **Proceedings of International Conference on Machine Learning**, p. 148–156, 1996.

FREUND, Y.; SCHAPIRE, R. E. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. **Journal of Computer and System Sciences**, v. 55, n. 1, p. 119–139, 1997.

GALAR, M. et al. A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. **IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews**, v. 42, n. 4, p. 463–484, 2012.

GEURTS, P.; ERNST, D.; WEHENKEL, L. Extremely randomized trees. **Machine Learning**, v. 63, n. 1, p. 3–42, 2006.

GIACINTO, G. et al. Intrusion detection in computer networks by a modular ensemble of one-class classifiers. **Information Fusion**, v. 9, n. 1, p. 69–82, 2008.

GIACINTO, G.; ROLI, F. Dynamic classifier selection based on multiple classifier behaviour. **Pattern Recognition**, v. 34, n. 9, p. 1879–1881, 2001.

GIACINTO, G.; ROLI, F.; DIDACI, L. Fusion of multiple classifiers for intrusion detection in computer networks. **Pattern Recognition Letters**, v. 24, n. 12, p. 1795–1803, 2003.

HE, H.; GARCIA, E. A. Learning from imbalanced data. **IEEE Transactions on Knowledge and Data Engineering**, v. 21, n. 9, p. 1263–1284, 2009.

HILLIER, F. S.; LIEBERMAN, G. J. **Introdução a Pesquisa Operacional**. 8. ed. Porto Alegre: McGraw-Hill, 2006.

HO, T. K. The Random Subspace Method for Constructing Decision Forest. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 20, n. 8, p. 832–844, 1998.

HUANG, Y. S.; SUEN, C. Y. A Method of Combining Multiple Experts for the Recognition of Unconstrained Handwritten Numerals. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 17, n. 1, p. 90–94, 1995.

JAHRER, M.; TÖSCHER, A.; LEGENSTEIN, R. **Combining predictions for accurate recommender systems**. Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD ’10. **Anais**...2010Disponível em: <http://dl.acm.org/citation.cfm?doid=1835804.1835893>

JIMENEZ, D. **Dynamically weighted ensemble neural networks for classification**. 1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98CH36227). **Anais**...1998Disponível em:

<http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=682375&contentType=Conference+Publications&matchBoolean=true&rowsPerPage=30&searchField=Search_All&queryText=(p_Title:%22Dynamically+Weighted+Ensemble+Neural+Networks+for+Cla%5Cnhttp://ieeexpl>

KABIESZ, J. et al. Application of rule-based models for seismic hazard prediction in coal mines. **Acta Montanistica Slovaca**, v. 18, n. 4, p. 262–277, 2013.

KITTLER, J.; HATER, M.; DUIN, R. P. W. On Combining classifiers. **Proceedings - International Conference on Pattern Recognition**, v. 2, n. 3, p. 897–901, 1996.

KO, A. H. R.; SABOURIN, R.; BRITTO, A. S. From dynamic classifier selection to dynamic ensemble selection. **Pattern Recognition**, v. 41, n. 5, p. 1735–1748, 2008.

KOHAVI, R. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. **Appears in the International Joint Conference on Articial Intelligence (IJCAI)**, v. 5, p. 1–7, 1995.

KRAWCZYK, B. et al. Ensemble learning for data stream analysis: A survey. **Information Fusion**, v. 37, p. 132–156, 2017.

KUNCHEVA, L. I. Clustering-and-selection model for classifier combination. **KES'2000. Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies. Proceedings (Cat. No.00TH8516)**, v. 1, p. 185–188, 2000.

KUNCHEVA, L. I. Classifier Ensembles for Changing Environments. In: **Multiple Classifier Systems**. [s.l: s.n.]. p. 1–15.

KUNCHEVA, L. I.; BEZDEK, J. C.; DUIN, R. P. W. Decision templates for multiple classifier fusion: an experimental comparison. **Pattern Recognition**, v. 34, n. 2, p. 299–314, 2001.

LESSMANN, S. et al. Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. **European Journal of Operational Research**, v. 247, n. 1, p. 124–136, 2015.

LICHMAN, M. **{UCI} Machine Learning Repository**. Disponível em: <http://archive.ics.uci.edu/ml>. Acesso em: 12 jan. 2018.

LISBOA DE ALMEIDA, P. R. et al. Music genre classification using dynamic selection of ensemble of classifiers. **Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on**, p. 2700–2705, 2012.

LU, Y. Knowledge integration in a multiple classifier system. **Applied Intelligence**, v. 6, n. 2, p. 75–86, 1996.

MANSOURI, K. et al. Quantitative structure-activity relationship models for ready biodegradability of chemicals. **Journal of Chemical Information and Modeling**, v. 53, n. 4, p. 867–878, 2013.

MOHRI, M.; ROSTAMIZADEH, A.; TALWALKAR, A. **Foundations of Machine Learning**. London, England: The MIT Press, 2012.

OPITZ, D.; MACLIN, R. Popular Ensemble Methods: An Empirical Study. **Journal of Artificial Intelligent Research**, v. 11, p. 169–198, 1999.

PANICHELLA, A.; OLIVETO, R.; DE LUCIA, A. **Cross-project defect prediction models: L'Union fait la force**. 2014 Software Evolution Week - IEEE Conference on Software Maintenance, Reengineering, and Reverse Engineering, CSMR-WCRE 2014 - Proceedings. **Anais**...2014

PETER, N.; RUSSELL, S. **Inteligência Artificial**. 3. ed. [s.l.] Prentice Hall, 2013.

POLIKAR, R. et al. Learn++: An incremental learning algorithm for supervised neural networks. **IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews**, v. 31, n. 4, p. 497–508, 2001.

POLIKAR, R. Ensemble based systems in decision making. **Circuits and Systems Magazine, IEEE**, v. 6, n. 3, p. 21–45, 2006.

PORCEL, C. et al. A hybrid recommender system for the selective dissemination of research resources in a technology transfer office. **Information Sciences**, v. 184, n. 1, p. 1–19, 2012.

ROGOVA, G. Combining the results of several neural network classifiers. **Neural Networks**, v. 7, n. 5, p. 777–781, jan. 1994.

ROKACH, L. Ensemble-based classifiers. **Artificial Intelligence Review**, v. 33, n. 1–2, p. 1–39, 2010.

RUTA, D.; GABRYS, B. Classifier selection for majority voting. **Information Fusion**, v. 6, n. 1, p. 63–81, mar. 2005.

SCHAPIRE, R. E. The strength of weak learnability. **Machine Learning**, v. 5, n. 2, p. 197–227, jun. 1990.

SCHAPIRE, R. E. Measures of Diversity in Classifier Ensembles. **Machine Learning**, v. 51, n. 2, p. 181–207, 2003.

SKURICHINA, M.; DUIN, R. P. W. Bagging for linear classifiers. **Pattern Recognition**, v. 31, n. 7, p. 909–930, 1998.

SKURICHINA, M.; DUIN, R. P. W. Bagging, boosting and the random subspace method for linear classifiers. **Pattern Analysis and Applications**, v. 5, n. 2, p. 121–135, 2002.

SOARES, R. G. F. et al. Using Accuracy and Diversity to Select Classifiers to Build Ensembles. **The 2006 IEEE International Joint Conference on Neural Network Proceedings**, 2006.

ŠTEFKA, D.; HOLEŇA, M. Dynamic classifier aggregation using interaction-sensitive fuzzy measures. **Fuzzy Sets and Systems**, v. 270, p. 25–52, 2015.

TSYMBAL, A. et al. Dynamic integration of classifiers for handling concept drift. **Information Fusion**, v. 9, n. 1, p. 56–68, 2008.

WOLOSZYNSKI, T.; KURZYNSKI, M. A probabilistic model of classifier competence for dynamic ensemble selection. **Pattern Recognition**, v. 44, n. 10–11, p. 2656–2668, 2011.

WOLPERT, D. H. Stacked generalization. **Neural Networks**, v. 5, n. 2, p. 241–259, 1992.

WOODS, K.; KEGELMEYER, W. P.; BOWYER, K. Combination of multiple classifiers using local accuracy estimates. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 19, n. 4, p. 405–410, 1997.

XIAO, H.; XIAO, Z.; WANG, Y. Ensemble classification based on supervised clustering for credit scoring. **Applied Soft Computing Journal**, v. 43, p. 73–86, 2016.

ZHU, X. Z. X.; WU, X. W. X.; YANG, Y. Y. Y. Dynamic classifier selection for effective mining from noisy data streams. **Fourth IEEE International Conference on Data Mining (ICDM'04)**, p. 305–312, 2004.