UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

RODRIGO AUGUSTO SCHELLER BOOS

Funções de Escolha Social para Elaboração de Consenso em Aprendizado de Máquina Descentralizado: um Estudo em Problemas de Classificação Multiclasse

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação

Orientadora: Profa. Dra. Mariana Recamonde Mendoza

Porto Alegre
2017

AGRADECIMENTOS

Agradeço primeiramente a meus pais, por todo incentivo, amor e apoio incondicional.

Também agradeço aos professores da Universidade Federal do Rio Grande do Sul, que de alguma forma colaboraram para a minha formação. Em especial a minha professora orientadora, Mariana Recamonde Mendoza, que desempenhou muito bem o papel de nortear as etapas deste trabalho.

À CAPES / CNPQ, instituições que financiaram diversos projetos dos quais fiz parte durante a graduação.

Deixo meu muito obrigado à todos!

RESUMO

Em alguns cenários envolvendo aprendizado de máquina, os dados a serem analisados podem estar sendo adquiridos e analisados de forma distribuída, tal que o conjunto de atributos de cada instância pode não estar completamente disponível em uma localização central, seja por motivos de confidencialidade ou de custos computacionais envolvidos na comunicação de grandes volumes de dados. Neste contexto, surge o problema de como realizar a classificação de novas instâncias de forma descentralizada, utilizando informações contidas em diferentes sítios ou bases de dados a fim de viabilizar a tarefa de classificação sem o comprometimento do seu desempenho. O objetivo principal deste trabalho é analisar diferentes abordagens baseadas na Teoria da Escolha Social para extrair um consenso a partir de um conjunto de modelos de classificação treinados em sítios distintos, utilizando apenas o subconjunto de atributos disponível localmente, focando especificamente em problemas de classificação multiclasse. Seguindo esta direção, a classificação de dados descentralizados ocorre em duas etapas: primeiramente os modelos treinados localmente são aplicados aos novos dados apresentados para predizer as classes correspondentes, e em um segundo momento estas predições são centralizadas e agregadas através de funções de escolha social a fim de se obter um resultado global. Os resultados dos experimentos realizados demonstram bom desempenho do método para diversos casos de teste obtidos do UCI *Machine Learning Repository*, tendo como principal conclusão que as funções de escolha social são boas agregadoras para conjuntos de dados com classes balanceadas, e que para cenários caracterizados por desbalanceamento de classes o método da pluralidade é mais promissor.

Palavras-chave: Aprendizado de máquina. aprendizado supervisionado. classificação multiclasse. classificação distribuída. funções de escolha social.

ABSTRACT

In some scenarios involving machine learning, the data to be analyzed may be acquired and analyzed in a distributed fashion, such that the set of attributes of each instance may not be completely available in a central facility, either due to confidentiality reasons or computational costs involved in the communication of large volumes of data. In this context, the problem that arises is how to perform the classification of new instances in a decentralized way, using information contained in different sites or databases in order to enable the classification task without compromising its performance. The main goal of this work is to analyze different approaches based on the Social Choice Theory to extract a consensus from a set of classification models trained in distinct sites, using only the subset of features available locally, focusing specifically on multiclass classification problems. Following this direction, the classification of decentralized data takes place in two stages: locally trained models are applied to the new data presented, in order to predict their corresponding classes; and in a second moment these predictions are centralized and aggregated through social choice functions in order to obtain a global result. Our results demonstrate good performance of the proposed method for several test cases from the UCI Machine Learning Repository, and the main conclusion of this work is that the social choice functions are good aggregators for datasets with balanced classes, while plurality is the most promising aggregation method for datasets characterized by large class imbalance.

Keywords: machine learning, multiclass classification, distributed classification, social choice functions, supervised learning.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AVA	All-Versus-All
AM	Aprendizado de Máquina
ECOC	Error Correcting Output Code
KNN	K-Nearest Neighbours
OVA	One-Versus-All
SCF	Social Choice Functions
SVM	Support Vector Machine
UCI	Universidade California Irvine

SUMÁRIO

1 INTRODUÇÃO

O crescente volume de dados disponível sobre os mais diversos contextos tem feito há alguns anos com que a comunidade científica e industrial aumente o interesse no estudo de algoritmos de aprendizado de máquina (AM). Uma das áreas de AM estuda a classificação de dados, e é amplamente utilizada em diversos setores. Detecção de spam, categorização de imagens, detecção de fraudes, definição de diagnóstico médico, dentre outros, são exemplos no mundo real que podem se beneficiar de bons algoritmos de classificação para suas tarefas. Para tal, existe uma gama de algoritmos bem estabelecidos na literatura. Contudo, estes algoritmos clássicos assumem centralização dos dados de treinamento, e o grau de sucesso da tarefa geralmente está atrelado a relevância e completude dos dados (MODI; KIM, 2005).

Por outro lado, pode-se observar que em diversas aplicações os dados vêm sendo gerados de forma distribuída (por exemplo em estações metereológicas, dados financeiros, dados clínicos, etc), e com volume crescente. A presença dos atributos em localidades diferentes pode ser justificada por fatores como confidencialidade ou custo de comunicação entre as entidades responsáveis pelos diferentes dados. Nota-se uma certa carência de métodos capazes de lidar de forma robusta com tarefas de classificação nestes contextos, exigindo algoritmos que lidem com o problema de forma descentralizada.

Peteiro-Barral and Guijarro-Berdiñas (2013) apontam que novos problemas surgem ao lidar com a aprendizagem distribuída, como por exemplo, a influência da heterogeneidade de dados entre as partições na acurácia da predição. Portanto, isso é uma linha aberta de pesquisa que precisa enfrentar esses novos desafios. Trabalhos neste tipo de contexto, como o abordado por Moghadam and Ravanmehr (2017), envolvendo análise de dados gerados por estações metereológicas, distribuídas fisicamente em lugares distintos, buscam resolver este tipo de problema. Ainda, existem trabalhos como o de Govada and Sahay (2016), que utilizam o particionamento de dados para tarefas de AM com dados astronômicos, visto que estes possuem muitas dimensões e representam um grande volume de informações. Com a distribuição da tarefa, pode-se reduzir a dimensionalidade das entradas, e consequentemente ter de lidar com formas mais simples do problema.

A tarefa de classificação em cenários distribuídos, em que os dados encontram-se descentralizados entre múltiplas partições, é um problema que pode ser tratado de diversas formas, e o presente trabalho busca analisar algumas alternativas para tal. De uma forma geral, a estratégia utilizada para lidar com este problema resume-se a treinar modelos de

classificação locais a partir das partições de dados às quais têm acesso, e agregar os modelos ou as predições realizadas por eles através de algum método pré-definido a fim de se elaborar um resultado global. Visto que a combinação de modelos propriamente ditos possui diversas limitações e desafios, como a definição de uma representação uniforme de conhecimento entre múltiplos modelos, a agregação de saídas dos classificadores tem sido mais adotada na literatura (PETEIRO-BARRAL; GUIJARRO-BERDIÑAS, 2013). Neste cenário, faz-se necessária a escolha de uma forma de obter a agregação dos resultados computados pelos diferentes modelos locais. A proposta mais comum para elaborar este tipo de consenso entre múltiplos modelos é a votação por pluralidade. No entanto, conforme discutido por trabalhos relacionados (RECAMONDE-MENDOZA; BAZZAN, 2016), existem métodos alternativos e potencialmente melhores a serem adotados ao invés de uma decisão por votação simples.

Este trabalho configura-se como uma extensão do estudo realizado por Recamonde-Mendoza and Bazzan (2016), no qual foi constatado o potencial de funções de escolha social na agregação de modelos de classificação binária treinados de forma descentralizada. No artigo, as autoras analisam um cenário distribuído com particionamento vertical de dados, isto é, quando os atributos utilizados para descrever as instâncias estão distribuídos entre múltiplos sítios. Assim, o intuito principal deste estudo é propor uma abordagem similar, mas focando em conjuntos de dados com múltiplas classes. A hipótese adotada é de que similarmente ao que foi observado anteriormente para classificação descentralizada em problemas binários, funções de escolha social podem representar estratégias úteis e eficientes para lidar com problemas multiclasse em cenários nos quais os dados estão fisicamente distribuídos e que apresentam restrições que impedem sua centralização. Mais especificamente, deseja-se verificar se algumas funções de escolha social funcionam bem em ambiente distribuídos com múltiplas classes, assim como avaliar quais destas são mais eficientes neste tipo de cenário a partir de uma análise da qualidade de suas predições.

O trabalho está organizado como segue. Os Capítulos 2 e 3 trazem uma revisão da literatura acerca dos principais assuntos envolvidos neste tópico, elencando métodos propostos por trabalhos anteriores e seus respectivos resultados para os problemas de classificação multiclasse e classificação descentralizada. No Capítulo 4, apresenta-se a abordagem proposta neste trabalho, descrevendo a metodologia adotada e os domínios de dados explorados para validação da mesma. Por fim, no Capítulo 5 traz-se um estudo a respeito dos resultados experimentais obtidos em variados conjuntos de dados, e no Capítulo 6 discute-se as conclusões e propostas de trabalhos futuros.

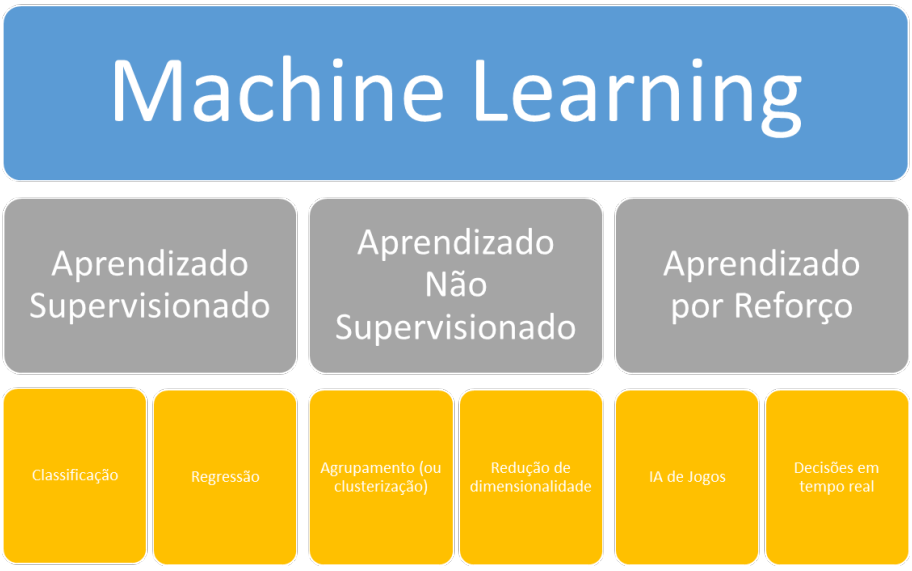
2 APRENDIZADO DE MÁQUINA E TAREFAS DE CLASSIFICAÇÃO

2.1 Aprendizado de Máquina

O Aprendizado de Máquina (AM), ou *machine learning*, é uma das subáreas da Ciência da Computação, e foi definido por Arthur Samuel em 1959 como um campo de estudo que confere aos computadores a habilidade de aprender sem serem explicitamente programados (MUNOZ, 2011). Na prática, AM pode ser visto como uma combinação de otimizações matemáticas e estatísticas realizadas com o objetivo de prever comportamentos para variadas entradas de dados (MITCHELL, 1997).

Há diferentes tipos de cenários em que o AM pode ser adotado, e em cada um dos mesmos pode-se aplicar um conjunto diferente de algoritmos.

Figura 2.1: *Machine Learning*, suas principais classes de algoritmos e tarefas relacionadas



Fonte: Autor, adaptado de Mitchell (1997)

Como visto na figura 2.1, existem três classes principais de algoritmos aplicados na área de AM, sendo elas:

- Aprendizado supervisionado: o algoritmo utiliza uma série de exemplos (entradas), rotulados com suas respectivas classes (saídas), para aprender padrões dentre estes dados; e com base nisto formular regras para classificação de novas entradas;
- Aprendizado não supervisionado: os algoritmos exploram dados de entrada não rotulados a fim de obter melhor percepção e uma descrição acerca de sua estrutura, detectando agrupamentos implícitos ou características especialmente úteis para a

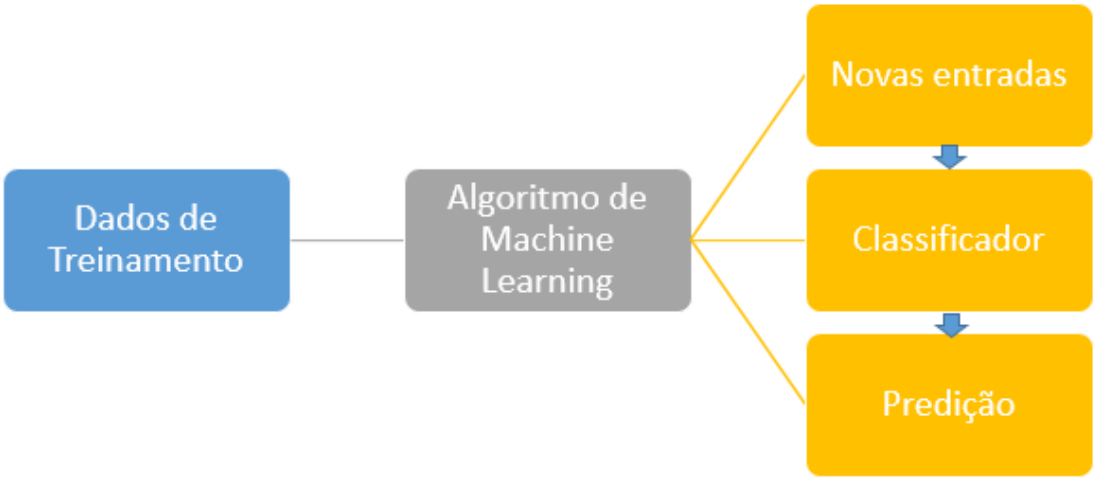
sua categorização;

- Aprendizado por reforço: os algoritmos aprendem políticas para agir através da interação repetida com o ambiente, o qual provê feedback positivo ou negativo para o algoritmo, assim guiando-o em decisões ou passos subsequentes.

Este trabalho foca em problemas de classificação, que são uma forma de aprendizado supervisionado. Neste tipo de tarefa, o objetivo é elaborar um modelo a partir de um conjunto de dados de treinamento, o qual seja capaz de generalizar as relações entre os atributos e classes destes dados tal que possa ser posteriormente aplicado para predizer a classe de novos dados, isto é, dados cujas classes são deconhecidas. Portanto, os dados de treinamento são compostos por um conjunto de instâncias ou exemplos, que possuem uma classe conhecida, pré-definida por um especialista. Define-se uma instância como um vetor de características quantitativas ou qualitativas que a descrevem no contexto do problema, sendo cada uma destas características denominada atributo ou *feature*. Os atributos e as respectivas classes de um conjunto de instâncias conhecidas formam os exemplos de treinamento, os quais são utilizados na indução do modelo de classificação (ou classificador) através de variadas regras estatísticas. O modelo gerado pode ser então utilizado para a predição da classe de instâncias desconhecidas.

O funcionamento do algoritmo compreende, assim, as etapas a seguir:

Figura 2.2: Abordagem para classificação através de Aprendizado de Máquina



Fonte: Autor

A abordagem mais comum deste tipo de problema é quando possuímos apenas duas classes (resultado da predição é sim ou não, por exemplo), como em filtros de e-mails, nos quais deseja-se saber se uma dada mensagem é spam ou não, cenário conhecido

como problema de classificação binária. Para estes problemas existem uma ampla variedade de métodos capazes de treinar modelos de classificação a fim de realizar a predição da classe para novos dados apresentados, e com bom desempenho (MITCHELL, 1997).

No entanto, diversos problemas do mundo real são caracterizados pela existência de múltiplas classes, sendo conhecidos como problemas de classificação multiclasse. Nestes contextos, deseja-se treinar um modelo para predizer a qual dentre N classes possíveis uma dada instância pertence a partir do valor de seus atributos. Visto que neste trabalho estamos interessados em classificação multiclasse, a próxima seção é dedicada à revisão da literatura relacionada a este tipo de problema de classificação, apresentando as principais abordagens existentes para tratar este caso específico.

2.2 Classificação em Problemas Multiclasse

A tarefa de classificação em AM é caracterizada por possuir instâncias de treinamento pertencentes a uma dentre N classes existentes no contexto de determinado problema. A partir deste conjunto de treinamento, busca-se aprender um modelo capaz de indicar corretamente a qual das N classes uma nova instância pertence, com base nos seus valores de atributos. Mais especificamente, temos uma classificação multiclasse quando o número de categorias, ou rótulos, possíveis para cada entrada é superior a dois ($N > 2$).

Formalmente, tem-se um conjunto de treinamento na forma ($\mathbf{x}_\mathrm{i}$, y_i), onde $\mathbf{x}_\mathrm{i} \in \mathbb{R}^n$ é o i-ésimo elemento do conjunto de treinamento (representado pelo seu vetor de atributos), e $y_i \in (C_1, C_2, ..., C_N)$, é sua respectiva categoria ou classe (ALY, 2005). Em um problema de classificação, busca-se encontrar um modelo H, tal que $H(\mathbf{z}) = y$, onde $\mathbf{z}$ é o vetor de atributos de uma entrada não vista no conjunto de treinamento, e y refere-se à classe com a qual $\mathbf{z}$ deveria estar associada.

Diversos algoritmos foram desenvolvidos e extensivamente explorados para classificação binária (apenas duas classes), e este trabalho aponta diversas formas de combinar estes métodos para gerar classificações multiclasse. Alguns algoritmos são naturalmente extensíveis para múltiplas classes, enquanto que em outros são necessárias agregações de diversos pares de classificação binária para atingir o resultado esperado (ALY, 2005; LORENA; CARVALHO, 2008).

2.2.1 Decomposição Binária

Lorena and Carvalho (2008) mencionam que a extensão direta de um algoritmo binário a uma versão multiclasse nem sempre é possível ou fácil de realizar. Por este motivo, as formas mais exploradas pela comunidade científica até hoje para classificação multiclasse são baseadas na derivação de métodos de classificação binária, e algumas destas são analisadas nesta seção (ALY, 2005; RIFKIN, 2008).

2.2.1.1 One-Versus-All

A estratégia *One-Versus-All* (OVA), também conhecida como "um-contra-todos", consiste em construir N classificadores binários distintos, onde N representa o número de classes do problema. Cada um destes classificadores será treinado de forma a aprender a distinguir exemplos de uma determinada classe das demais $N - 1$ classes, tomando as instâncias pertencentes à classe de interesse como exemplos positivos e todas as demais como exemplos negativos.

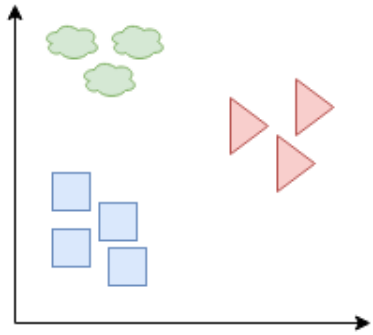
Segundo Rifkin (2008), tendo um classificador binário para cada classe, pode-se dizer que o i-ésimo classificador representa um modelo que calcula a probabilidade de uma instância pertencer ou não àquela classe. Define-se $f_i(\mathbf{x})$ como o resultado dado pelo modelo referente à classe i, significando uma probabilidade entre 0 e 1 de pertencer à i-ésima classe, com relação a uma instância **x**. A partir disso, verifica-se qual a classe com maior probabilidade dada como resultado, logo

$$f(\mathbf{x}) = \underset{i}{\operatorname{argmax}}(f_i(\mathbf{x})) \tag{2.1}$$

Pode-se visualizar um exemplo da abordagem OVA nas figuras 2.3 e 2.4, representando um problema composto por 3 classes. Neste caso, realiza-se a decomposição do problema original (2.3) em 3 classificações binárias distintas, sendo triângulos ou não, quadrados ou não, e nuvens ou não. A figura 2.4 representa os três problemas distintos de classificação criados com base na decomposição por OVA.

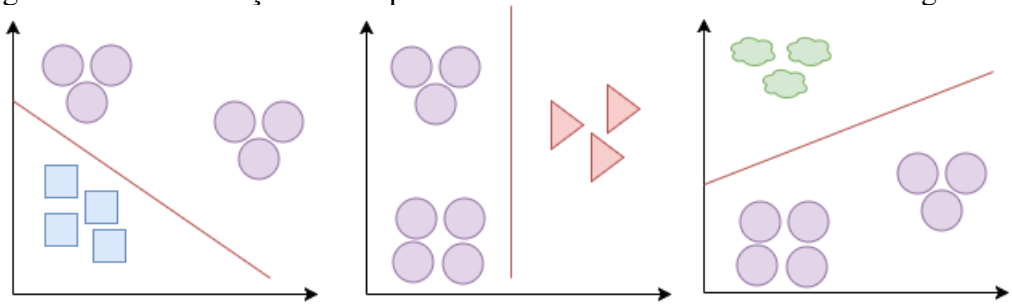
Esta técnica é amplamente utilizada, e diversos autores apontam como a forma mais simples e menos custosa computacionalmente para realizar a classificação multiclasse (RIFKIN; KLAUTAU, 2004; ALLWEIN; SCHAPIRE; SINGER, 2000; HSU; LIN, 2002). Rifkin and Klautau (2004) discutem que esta abordagem é tão precisa quanto qualquer outra utilizada, em um cenário centralizado utilizando um tipo específico de al-

Figura 2.3: Representação visual dos dados de treinamento para um problema multiclasse composto por 3 classes



Fonte: Autor

Figura 2.4: Classificação de um problema multiclasse com base na abordagem OVA



Fonte: Autor

goritmo supervisionado denominado *Support Vector Machines* (SVMs), ou máquina de vetores de suporte.

No estudo conduzido por Rifkin and Klautau (2004), compara-se a precisão da técnica OVA com algumas outras, como *All-Versus-All* (AVA) e Código de Correção de Erros (ECOC), as quais serão explicadas em detalhes nas seções seguintes. Para isto, analisou-se o desempenho das diferentes técnicas em 10 conjuntos de dados multiclasse do UCI *Machine Learning Repository*, doravante referido apenas como UCI (LICHMAN, 2013). Podemos citar alguns exemplos de conjuntos de dados utilizados, bem como suas características e desempenho observado na classificação:

Tabela 2.1: Resultados do estudo de Rifkin and Klautau (2004). O melhor resultado dentre os 3 conjuntos de dados comparados está destacado em negrito.

Dataset	Número de Instâncias	Número de Atributos	Número de Classes	Taxa de erro OVA (%)	Taxa de erro AVA (%)	Taxa de erro ECOC (%)
Soybean	307	35	19	5.85	6.38	**5.58**
Letter	20000	16	26	**2.75**	3.85	2.95
Sat-image	5435	36	7	7.80	8.15	**7.65**

Analisando os números relativos a taxa de erro na tabela acima, pode-se concluir que os testes realizados para estes conjuntos indicam que a acurácia do método OVA

realmente pode equiparar-se aos demais métodos para decomposição de problemas multiclasse. Os outros sete conjuntos testados pelos autores também indicaram desempenho comparável entre o OVA, AVA e ECOC.

Ng, Tse and Tsui (2014) ainda propõem um estudo que aplica o OVA no diagnóstico de falhas em rolamentos industriais. As classes neste caso são referentes a qual tipo de defeito o rolamento apresenta, sendo 4 possíveis. Além disso, são utilizados 18 atributos para representar os dados. Aplicando OVA com classificadores binários SVM os autores conseguiram uma taxa de 66,7% de acertos na predição das classes.

Existem diversos outros trabalhos que exploram o OVA de variadas maneiras, e ainda vale citar especificamente o projeto desenvolvido por Hong et al. (2008), no qual os autores conseguem atingir um desempenho de 90% de acertos para identificação de impressões digitais divididas em 4 classes. Neste problema, as classes correspondem a diferentes tipos de formato nas curvas presentes nas impressões digitais e os atributos correspondem a setores de cada imagem, totalizando 192 para cada instância. Esta classificação é útil pois reduz o espaço de busca quando deseja-se verificar se há um *match* de uma digital em um banco de dados.

2.2.1.2 All-Versus-All

All-Versus-All (AVA), ou "todos-contra-todos", é outra forma simples de decomposição de problemas multiclasse em classificações binárias (RIFKIN, 2008). Nesta abordagem, treina-se um classificador para cada par de classes existentes, e no processo de classificação de novas instâncias cada um dos classificadores gerados indica qual dentre as duas classes contidas nos seus dados de treinamento é a mais provável para uma dada entrada. Para inferir a classe de novos exemplos, verifica-se qual das classes possui maior soma das probabilidades entre todos os classificadores que a contém.

De acordo com Manning and Raghavan (2008), formalmente, para N classes, o algoritmo executa N*(N-1)/2 classificadores, sendo i e j os indíces das classes contidas em um dado classificador, no formato:

$$f_{ij}, \forall i, j \mid i \neq j. \tag{2.2}$$

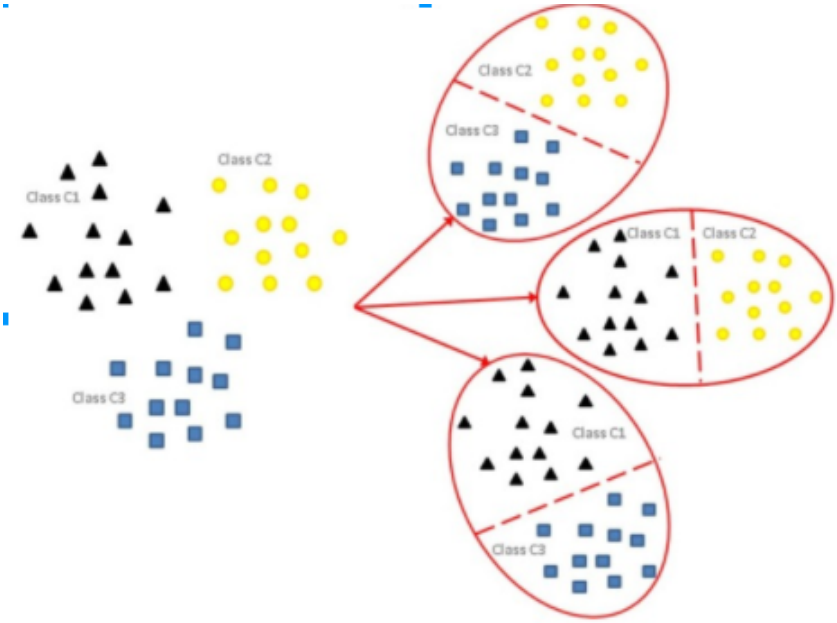
Com isso, a função que retorna a classe predita para uma nova entrada é definida como

$$f(\mathbf{x}) = \underset{i}{\operatorname{argmax}} \sum (f_{ij}(\mathbf{x}))\, \forall j \tag{2.3}$$

Nota-se que o número de classificadores exigidos nesta abordagem é muito maior que na OVA, porém, como é explicado por Hsu and Lin (2002), em geral estes são mais simples que os classificadores treinados com a estratégia OVA, tal que o maior número de classificadores não implica necessariamente em um maior custo computacional. No entanto, é importante ressaltar que nenhuma informação é fornecida pelo preditor quando o exemplo não pertence às classes i ou j por ele englobadas (LORENA; CARVALHO, 2008).

Pode-se utilizar como exemplo o mesmo caso explicado na seção anterior, em um cenário de classificação com 3 classes. Como pode ser visualizado na figura 2.5, cada par de classes, i.e., (C1,C2), (C1,C3) e (C2,C3), gera um classificador. Verifica-se qual a classe correspondente à uma entrada desconhecida definindo qual dentre as três classes possui maior grau de aceitação entre todas as comparações efetuadas.

Figura 2.5: Classificação de um problema com 3 classes através de decomposição pela estratégia AVA



Fonte: (NG; TSE; TSUI, 2014)

Hsu and Lin (2002) propuseram utilizar classificadores AVA para a tarefa de identificação de letras do alfabeto. Neste caso o número de classes é 26 e a base de dados utilizada é a presente no UCI (LICHMAN, 2013), com 20000 instâncias. Neste projeto os pesquisadores conseguiram alcançar um percentual de 3.22 para a taxa de erro, correspondente ao já observado para o mesmo caso utilizando OVA, com a diferença de que o tempo de convergência geral do algoritmo foi um pouco menor para AVA.

Um estudo mais extenso utilizando esta técnica foi proposto por Allwein, Schapire

and Singer (2000), no qual uma série de conjuntos de dados do UCI (LICHMAN, 2013) foi utilizada como teste. Podemos visualizar que os resultados obtidos com AVA superam o método OVA para todos os casos testados na Tabela 2.2.

Tabela 2.2: Resultados dos experimentos realizados por Allwein, Schapire and Singer (2000). Os melhores desempenhos para cada conjunto de dados estão destacados em negrito

Dataset	Número de Instâncias	Número de Atributos	Número de Classes	Taxa de erro OVA (%)	Taxa de erro AVA (%)
Dermatology	366	33	6	5.0	**3.1**
Glass	214	10	7	31.0	**28.6**
Sat-image	5435	36	7	14.9	**11.7**
Ecoli	336	8	8	19.1	**17.6**

2.2.1.3 Código de Correção de Erros

A abordagem *Error Correcting Output Code* (ECOC), ou Código de Correção de Erros, entende a classificação como uma tarefa de comunicação (LORENA; CARVALHO, 2008). Ao invés de simplesmente realizar comparações para cada classe como na estratégia OVA, faz-se um número de comparações maior que o número de classes contidas no problema a fim de introduzir redundância, com o intuito de corrigir possíveis erros nas predições de cada classificador (DIETTERICH; BAKIRI, 1995). A cada classificador é assimilado um *array* de valores binários, sendo que cada posição i do *array* corresponde a i-ésima classe. Na tabela formada pela junção destes *arrays*, cada linha representa uma classe, e cada coluna o *array* de um classificador, conforme exemplifica a Tabela 2.3. A notação f_i representa o i-ésimo classificador binário treinado.

Tabela 2.3: Exemplo de regras para classificador ECOC. Adaptado de Aly (2005)

	f_1	f_2	f_3	f_4	f_5	f_6	f_7
Classe 1	0	0	0	0	0	0	0
Classe 2	0	1	1	0	0	1	1
Classe 3	0	1	1	1	1	0	0
Classe 4	1	0	1	1	0	1	0
Classe 5	1	1	0	1	0	0	1

Na tabela 2.3, é apresentado um modelo com sete classificadores, todos sendo classificadores binários, treinados para distinguir elementos pertencentes às classes marcadas com 1 das classes marcadas com 0 em suas respectivas colunas. Um novo elemento é avaliado como positivo (ou "1") caso a probabilidade de pertencer a uma das classes marcadas com "1" seja maior que 50%. Caso contrário, é avaliado como "0".

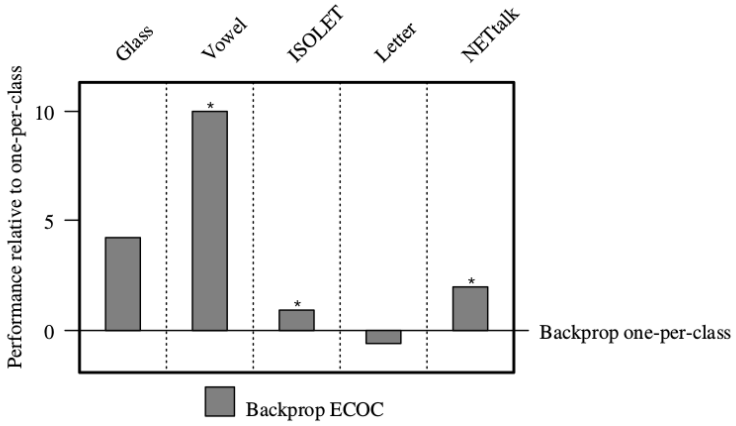
Define-se o vetor solução V(**x**) de um exemplo de entrada **x** como o resultado dado por cada um dos classificadores. Define-se a distância $d(y, \mathbf{x})$ de uma classe $y \in (C_1, C_2, ..., C_N)$ e um exemplo **x** como a distância de Hamming entre o *array* de y com o vetor solução V(**x**). Com isso, a classe de saída gerada por um modelo ECOC, para um exemplo V(**x**), pode ser dada por

$$f(\mathbf{x}) = \underset{y}{\operatorname{argmin}} \; d(y, \mathbf{x}) \; \forall y. \tag{2.4}$$

Ainda, Dietterich and Bakiri (1995) apontam que é importante para o funcionamento do algoritmo a escolha de *arrays* para cada classificador de forma a maximizar a distância de Hamming entre eles. Isto decorre do fato de que para se avaliar um novo exemplo, aplica-se todos os classificadores para predição da classe e o *array* binário originado pela classificação dos mesmos é comparado ao *array* original de cada classe (i.e., linhas na Tabela 2.3), sendo selecionada como predição final a classe cujo *array* possui a menor distância de Hamming com o *array* binário gerado na classificação.

Dietterich and Bakiri (1995) argumentam em seus estudos que o desempenho deste método é superior aos demais, dando como exemplo alguns conjuntos de dados explorados e seus resultados. Porém como a diferença de desempenho é pequena, é difícil saber se realmente são melhores ou a diferença foi simplesmente fruto de escolha dos algoritmos de classificação binária utilizados.Os conjuntos utilizados nos testes realizados pelos autores são os mesmos já vistos em outras seções, e com exceção dos experimentos com os dados Vowel, os resultados mostram leve vantagem deste método com relação ao OVA (Figura 2.6).

Figura 2.6: Desempenho do método ECOC relativo ao OVA. Asterisco indica que a diferença é significativa em um nível de significância de 0,05.



Fonte: (DIETTERICH; BAKIRI, 1995)

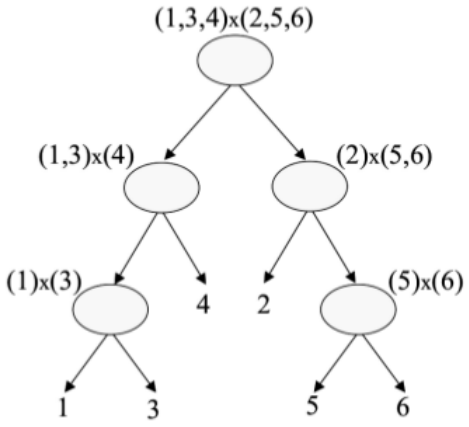
2.2.1.4 Divisão Hierárquica

Segundo Aly (2005), podemos utilizar uma divisão hierárquica do espaço de saída, com as N classes possíveis dispostas em uma árvore. Esta estrutura é criada de tal forma que as classes em cada nodo pai são divididas entre dois nodos filho, tal que metade das classes de um nó é assimilada ao filho da direita, outra metade para o filho à esquerda. Este processo continua até as folhas, sendo que cada uma delas vai obrigatoriamente conter apenas uma classe. Em cada nodo da árvore, existe um classificador binário, que verifica se a entrada pertence a uma das classes do filho à esquerda ou do filho à direita.

A ideia por trás dos algoritmos de divisão hierárquica é o princípio da divisão e conquista para chegarmos à classe que um exemplo pertence. Para isso, treina-se diversos classificadores, cada um responsável por cortar na metade a busca pela classe correspondente de uma instância. Ainda pode-se entender o funcionamento destes algoritmos como uma busca binária em uma árvore, na qual o valor procurado é a classe de saída. Com isso, teremos número de classificadores na ordem de 2*N para a árvore completa.

Todos estes classificadores passam pela etapa de modelagem a partir do conjunto de treinamento. Porém quando deve-se avaliar um novo indivíduo, avalia-se apenas $logN$ classificadores (princípio da busca binária), pois a cada classificador analisado, metade das classes restantes é excluída do espaço de possibilidades. Quando chega-se em uma folha, esta representa uma classe propriamente dita, e logo é o resultado da classificação. Pode-se ilustrar o funcionamento do algoritmo através do exemplo na figura 2.7:

Figura 2.7: Exemplo de divisão hierárquica para classificação em um problema multi-classe.



Fonte: (LORENA; CARVALHO, 2008)

Neste caso, o problema de classificação é composto por 6 classes. Quando o algoritmo inicia, classifica-se a entrada em um dos dois conjuntos (1,3,4) ou (2,5,6). Esta

divisão das classes em subconjuntos é feita sucessivamente, até que o algoritmo atinge um nó folha, representando uma classe específica.

Em comparação com a abordagem OVA, os algoritmos hierárquicos geralmente apresentam classificadores binários mais uniformes, isto é, o número de elementos na duas classes tende a ser mais balanceado. Este fator pode ser positivo para classificadores que funcionam utilizando populações proporcionais com relação a distribuição entre classes. A partição de um conjunto de classes em dois subconjuntos não é feita de forma arbitrária, mas sim visando privilegiar que classes semelhantes mantenham-se em conjunto. Para tanto, é aplicado um processo de *simulated annealing* determinístico, como explicado por Kumar, Ghosh and Crawford (2002).

Alguns estudos foram dirigidos a avaliar o desempenho da abordagem hierárquica, dentre os quais Kumar, Ghosh and Crawford (2002) propõem a classificação de imagens hiperespectrais, isto é, dados que possuem mais espectros do que os visíveis pelo olho humano. Os dados utilizados são imagens da terra vistas da órbita obtidas através do *Kennedy Space Center*. O objetivo da classificação é de analisar qual tipo de vegetação uma imagem apresenta, existindo doze classes possíveis. Foram utilizadas cerca de 350 instâncias para cada classe. As imagens obtidas possuem 224 espectros, mas foram utilizados 180 para elaboração dos atributos do problema. Utilizando a técnica hierárquica, os pesquisadores conseguiram atingir 94,7% de acurácia, i.e. taxa de predições corretas, enquanto que a classificação baseada em Redes Neurais com Perceptron de Múltiplas Camadas atingiu uma taxa de acertos de apenas 74,5%.

Outro estudo, conduzido por Rajan and Ghosh (2004), analisou a performance de algoritmos hierárquicos em vários conjuntos de dados do UCI (LICHMAN, 2013). A relação dos conjuntos utilizados, suas características e as taxas de acerto obtidas são descritos na Tabela 2.4.

Tabela 2.4: Resultados do estudo de Rajan and Ghosh (2004) para classificação multiclasse com divisão hierárquica. Os melhores resultados para cada conjunto de dados estão destacados em negrito.

Dataset	Nº de classes	Nº de atributos	Nº de instâncias	% Acerto Hierárquico	% Acerto ECOC
Glass	7	9	214	**66.59**	63.7
Yeast	11	9	1484	**56.91**	53.81
Sat-image	6	36	4435	**91.45**	91.4
Segment	7	16	2310	**89.94**	89.17
Page blocks	5	10	5473	94.5	**94.8**
Pen Digits	10	16	7494	97.9	**98.45**
Opt Digits	10	64	3823	96.6	**96.99**
Letter	26	16	16000	96.23	**96.83**
Vowel	11	12	528	**61.03**	51.29

Observa-se que para cerca de metade dos casos, esta abordagem se mostrou levemente mais eficiente que a comparada (ECOC). No entanto, as diferenças observadas em alguns casos são muito pequenas, tal que é difícil afirmar que o ganho de desempenho proporcionado por estratégias hierárquicas em relação ao método ECOC é significativo, podendo estar associada também a características dos próprios conjuntos de treinamento.

2.2.2 Extensão Direta

Ainda existem alguns trabalhos dedicados a aplicação de classificação multiclasse a partir de um modelo único, utilizando algoritmos naturalmente extensíveis para tal tarefa (OU; MURPHEY, 2007; RAJAN; GHOSH, 2004; BAY, 1998). Vale mencionar que utilizar este tipo de algoritmo pode deixar o problema bem mais custoso computacionalmente que em decomposições binárias, pois as funções de otimização tendem a se tornarem mais complexas (ALY, 2005). Apresentamos aqui alguns dos algoritmos de classificação que possuem extensão direta para problemas multiclasse.

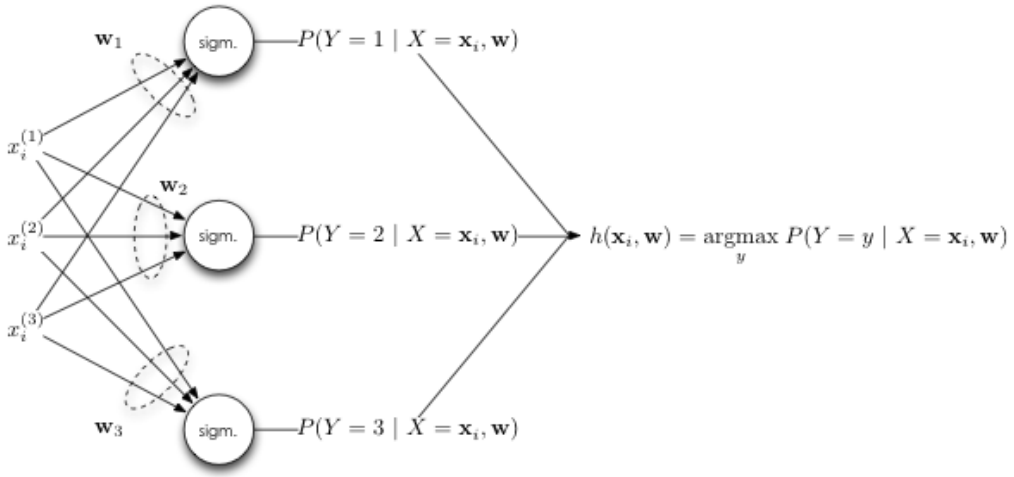
2.2.2.1 Redes Neurais

Redes neurais, conforme Ou and Murphey (2007), são naturalmente extensíveis para classificação multiclasse. A modificação necessária é que é preciso incluir neurônios e funções de ativação para cada par de atributo e classe. Com isso, obtem-se N saídas, cada uma correspondendo a uma dentre as N classes existentes. Cada uma destas N saídas calcula a probabilidade de o indivíduo pertencer à classe correspondente, e a predição é definida de acordo com a classe associada com a maior probabilidade. A Figura 2.8 representa o modelo de uma rede neural para três atributos e três classes.

No exemplo, tem-se uma rede neural em que cada uma das classes tem seus neurônios representados pelos círculos em tracejado. Os mesmos podem ser um neurônio simples, ou uma estrutura mais complexa, como uma rede com múltiplas camadas. Esta abordagem também pode ser entendida como uma variante do método OVA, pois incorporam-se um ou vários neurônios para cada classe independentemente das demais.

Alguns estudos demonstram o uso de redes neurais para a tarefa de classificação multiclasse, e o trabalho realizado por Ou and Murphey (2007) é um dos mais citados. Ele utiliza diversos conjuntos de dados do UCI (LICHMAN, 2013), e mostra que uma única rede neural gerou resultados melhores do que várias com OVA (Tabela 2.5). No entanto,

Figura 2.8: Redes Neurais adaptadas para classificação em problemas multiclasse



Fonte: (DOLHANSKI, 2013)

como visto nas outras técnicas já mencionadas, o resultado do trabalho demonstra leve melhora para alguns conjuntos de dados, mas sem grande significância.

Tabela 2.5: Resultados de Ou and Murphey (2007). Os melhores resultados para cada conjunto de dados estão destacados em negrito.

Dataset	Nº de classes	Nº de Atributos	Nº de instâncias	% Acerto com Única Rede	% Acerto várias redes com OVA
Glass	7	9	214	70.56	**71.04**
Shuttle	7	4	23	**99.63**	99.63
Digit	10	16	10992	**97.39**	96.50
Letter	26	16	20000	**87.38**	86.34
Iris	3	4	150	98	98

2.2.2.2 K-Vizinhos Mais Próximos

O algoritmo de *K-Nearest Neigbours* (KNN), ou K-Vizinhos Mais Próximos, define a escolha da saída partindo das instâncias presentes no treinamento, calculando quais as mais próximas ou similares com relação ao novo exemplo, e utilizando as mesmas para atribuir a classe ao exemplo que se deseja classificar. A ideia geral é escolher k vizinhos, e realizar uma votação entre os mesmos para decidir qual será a classe da nova instância (ou uma soma das probabilidades).

Definindo de outra forma, para classificarmos um exemplo desconhecido, calcula-se a distância do mesmo para todos os indivíduos do conjunto de treinamento (ALY, 2005). As k menores distâncias são identificadas, e a classe mais representada dentre essas k instâncias é considerada a classe de saída do algoritmo. A distância mencionada pode ser calculada utilizando diversas funções, como por exemplo a distância Euclidiana

entre os valores dos atributos para valores contínuos, e a distância de Hamming para valores simbólicos. Uma das vantagens deste tipo de classificador é que em geral atua bem sobre conjuntos pequenos de treinamento (BAY, 1998).

A técnica proposta por Bay (1998) para abordar problemas multiclasse com KNN, é de combinar vários 1-NN classificadores para apenas um subconjunto dos atributos. O número de classificadores utilizado foi obtido empiricamente realizando testes sobre números aleatórios entre 2 e o número de atributos de cada conjunto de dados. A função de agregação entre os mesmos é a votação simples, isto é, a classe mais votada dentre a maioria dos classificadores vence.

Foi realizada a comparação do algoritmo proposto com relação aos classificadores 1-NN (baseado em um vizinho único) e KNN, com k sendo definido através de experimentos com diversos valores possíveis. Os conjuntos de dados utilizados também foram obtidos no UCI (LICHMAN, 2013). A Tabela 2.6 sumariza alguns resultados obtidos no estudo de Bay (1998).

Tabela 2.6: Resultados obtidos por Bay (1998). Os melhores resultados para cada conjunto de dados estão destacados em negrito.

Dataset	Nº de classes	Nº de atributos	Nº de instâncias	% Acerto Multi NN (algoritmo proposto)	% Acerto KNN	% Acerto 1-NN
Glass	7	9	214	**76.1**	66.8	67.9
Hepatites	2	19	155	**82.7**	80.4	79.2
Sonar	2	60	208	82.3	**85.1**	85
Iris	3	4	150	92.7	**95.1**	94.3

Intuitivamente, combinar modelos que não possuem todas as informações dos indivíduos de treinamento pode não parecer uma boa estratégia. Porém, como demonstram os resultados, o desempenho deste algoritmo é comparável aos modelos com apenas um classificador com acesso a todos os atributos.

Analisando o cenário proposto, pode-se notar que de certa forma este método é naturalmente extensível para o cenário distribuído, que é o interesse deste trabalho. A diferença é que em Bay (1998), os diferentes classificadores podem compartilhar atributos, enquanto que na abordagem que pretende-se explorar ao longo deste trabalho os atributos são mutuamente exclusivas.

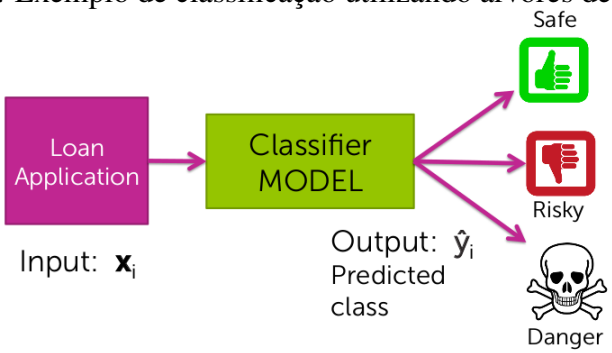
2.2.2.3 Árvores de decisão

Árvores de decisão são uma forma de realizar classificação, independente do número de classes (ALY, 2005). Pode-se entender como uma série de perguntas, as quais buscam eliminar o maior número possível das possibilidades de resposta a cada vez. De

acordo com os dados do conjunto de treinamento, escolhem-se duas faixas de valores para os atributos, de modo a separar possíveis entradas em duas partições. Após uma série de iterações neste formato, resta apenas um valor de saída possível.

No caso da classificação multiclasse, a ideia é realizar uma série de verificações nos valores dos atributos, de forma que a cada uma destas verifcações o número de classes possíveis para saída é reduzido. Um exemplo prático pode ser visualizado no diagrama da Figura 2.9.

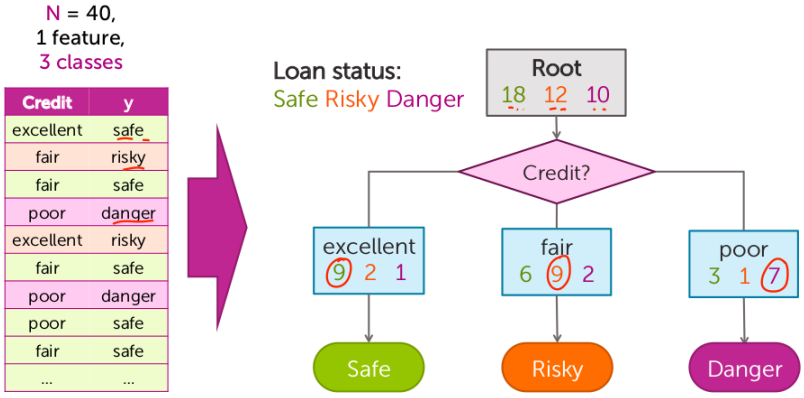
Figura 2.9: Exemplo de classificação utilizando árvores de decisão.



Fonte: (GUESTRIN; FOX, 2014)

Neste exemplo, deseja-se desenvolver um algoritmo que classifica uma aplicação de empréstimo entre segura, arriscada, e perigosa (GUESTRIN; FOX, 2014). Para efeito de simplificação, assume-se que os dados possuem apenas um atributo: o nível de crédito do indivíduo (excelente, bom, ruim, etc). Com isso, tem-se uma árvore de decisão com a estrutura dada pela Figura 2.10.

Figura 2.10: Estudo de caso para classificação multiclasse usando árvores de decisão.



Fonte: (GUESTRIN; FOX, 2014)

Clientes com histórico de crédito excelente, serão provavelmente associados à classe *safe*, pois a maior porcentagem de *safe* refere-se aos mesmos. A mesma linha

de raciocínio vale para os outros ramos da árvore.

Um estudo realizado por Hoens et al. (2012) utiliza árvores de decisão para lidar com classificação multiclasse, e os mesmos apontaram diversos resultados interessantes para conjuntos de dados do UCI (Tabela 2.7). Para a maioria dos conjuntos de dados, a porcentagem de acertos é maior ou equivalente aos algoritmos já explorados nas seções anteriores. Inclusive em alguns a diferença é significativa, indicando uma possível superioridade deste algoritmo em relação aos demais.

Tabela 2.7: Resultado do estudo de classificação com árvores de decisão obtido por Hoens et al. (2012).

Dataset	Nº de classes	Nº de atributos	Nº de instâncias	% Acerto
Glass	6	9	214	88.34
Sat	6	36	6435	99.87
Yeast	9	9	1484	80.87
Dermatology	6	35	366	99.68

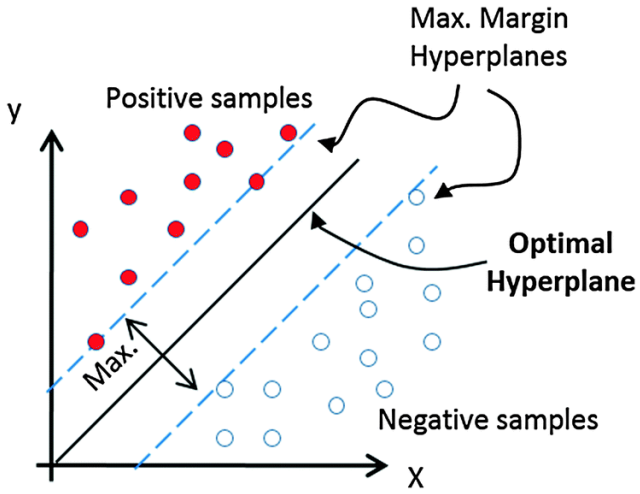
2.2.2.4 Máquinas de Vetores de Suporte

De acordo com Aly (2005), as máquinas de vetores de suporte, referenciadas em inglês como *Support Vector Machines* (SVM), estão entre os melhores algoritmos para classificação, sendo originalmente implementadas para o caso de problemas binários. A ideia básica nos modelos SVM é a formulação de uma linha (ou plano) no espaço definido pelos exemplos dados como entrada, que separe de forma clara e mais próxima ao ideal as duas classes envolvidas no processo. Esta entidade responsável por separar indivíduos das duas classes, que é um hiperplano, pode ser encontrada de forma ótima para classes linearmente separáveis. O exemplo da figura 2.11 demonstra essa situação. Nesta figura, os elementos que tocam as linhas tracejadas são os mais próximos a uma possível linha de separação. O algoritmo SVM ainda garante que a distância entre eles seja máxima.

Em um problema real de classificação, essa premissa de classe linearmente separável normalmente é falsa. Mesmo assim, classificadores SVM ainda conseguem encontrar bons hiperplanos, transformando o espaço original de alguns exemplos (através de um mapeamento não linear).

Diversos autores, tais como Weston and Watkins (1998), Lee, Lin and Wahba (2004), Crammer and Singer (2001), propõem diferentes formulações para adaptar o algoritmo SVM ao caso multiclasse, e um dos que demonstra resultados mais promissores é o método apontado por Crammer and Singer (2001). O processo proposto é complexo de desenvolver, e está explicado em detalhes no artigo. Diversas otimizações foram reali-

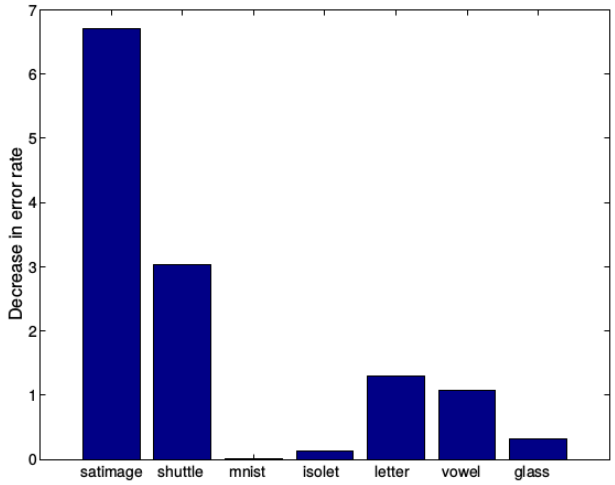
Figura 2.11: Exemplo de classes linearmente separáveis, analisadas com SVM.



Fonte: (FERNANDEZ-LOZANO et al., 2014)

zadas na implementação devido ao fato de o algoritmo ser muito custoso.

Figura 2.12: Diminuição da taxa de erro do método proposto por Crammer and Singer (2001) com relação à estratégia OVA



Fonte: (CRAMMER; SINGER, 2001)

O gráfico da Figura 2.12 representa o percentual de melhora na taxa de erro do algoritmo proposto com relação a abordagem OVA em sete conjuntos de dados do UCI (LICHMAN, 2013). Para um dos conjuntos de dados, satimage, a redução na taxa de erro se aproxima a 7%, enquanto para os demais esta redução varia entre 0% e 3%, apontando que o maior custo e complexidade poderiam ser justificados pela perspectiva de maior acurácia na classificação. De acordo com os autores, melhoras significativas foram observadas para os conjuntos de dados que apresentaçam maior razão entre número de exemplos por classe (CRAMMER; SINGER, 2001) .

3 CLASSIFICAÇÃO DE DADOS DESCENTRALIZADOS

Em grande parte dos estudos realizados a respeito de tarefas de classificação com algoritmos de AM, assume-se que o classificador possui acesso irrestrito a todos os atributos e exemplos do conjunto de dados de treinamento. No capítulo anterior, discutiu-se métodos e trabalhos relacionados para este tipo de classificação, dita centralizada, com ênfase em classificação com múltiplas classes. O artigo elaborado por Aly (2005) também apresenta uma pesquisa elencando os principais tipos de classificadores existentes para aplicação neste cenário.

No entanto, conforme Recamonde-Mendoza and Bazzan (2016) mencionam, existem situações em que não é possível executar a tarefa de classificação de forma centralizada, isto é, com todos os dados dos conjuntos de treinamento disponíveis em um único sítio ou base de dados. Há vários contextos em que depara-se com este tipo de limitação, seja por grande quantidade de dados, que inviabiliza a computação da classificação com apenas um modelo central, ou em situações em que a privacidade de diferentes dados é exigida. Alguns cenários práticos onde este tipo de restrição pode ser observado são em pesquisas biomédicas e detecção de fraudes bancárias, onde motivos éticos e/ou legais impedem o compartilhamento de diferentes tipos de dados sobre o mesmo indivíduo entre instituições distintas (MCCONNELL; SKILLICORN, 2004) .

Adicionalmente, a descentralização também pode ocorrer por motivos de distribuição geográfica das instituições ou fontes geradoras dos dados, as quais em algumas situações podem não conseguir transmitir rapidamente suas informações para outros locais, principalmente quando os dados são gerados de forma contínua e em grande volume. Um domínio característico deste tipo de cenário é a análise de dados meteorológicos, a qual se dá a partir de dados gerados por estações meteorológicas que estão naturalmente distribuídas geograficamento, adquirindo dados com uma alta frequência e de forma regular (MOGHADAM; RAVANMEHR, 2017). Nestes casos, a análise descentralizada dos dados é uma alternativa interessante visto que elimina a necessidade de transferir os dados brutos para uma base central, o que por sua vez demandaria maiores recursos de comunicação e processamento.

Para a definição do problema abordado neste trabalho, assume-se que classificadores locais podem ser treinados em cada sítio ou fonte, baseados em suas respectivas visões parciais do conjunto de dados. Estes classificadores podem ser obtidos com quaisquer dos algoritmos e estratégias explorados no capítulo anterior. Uma limitação intrínseca a este

cenário é que em algumas situações os atributos disponíveis em uma dada fonte não são descritivos o suficiente para a predição correta a respeito da classe de possíveis exemplos de entrada. A partir do conjunto de classificadores locais, cada qual treinado com uma partição dos dados, adota-se uma estratégia para combinação dos resultados de cada classificador a fim de agregar o conhecimento representado por cada modelo sem requisitar acesso aos dados utilizados para treiná-los. Este trabalho avalia algumas estratégias para combinação de modelos de classificação treinados de forma descentralizada para problemas com múltiplas classes.

Este capítulo irá revisar os principais conceitos relacionados a classificação descentralizada de dados, discutindo desde os tipos de particionamento de dados existentes até possíveis abordagens computacionais para lidar com esta característica do domínio. Ao final, irá revisar trabalhos relacionados à tarefa de classificação descentralizada, com foco em problemas multiclasse.

3.1 Particionamento dos dados

Em problemas de classificação abordando cenários com informações fisicamente descentralizadas, existem duas maneiras em que a distribuição dos dados entre os locais pode ocorrer. A primeira refere-se a fontes de informação homogêneos, isto é, cada sítio tem acesso a exatamente os mesmos atributos, porém a respeito de instâncias diferentes. Fontes homogêneas correspondem a partições horizontais dos dados.

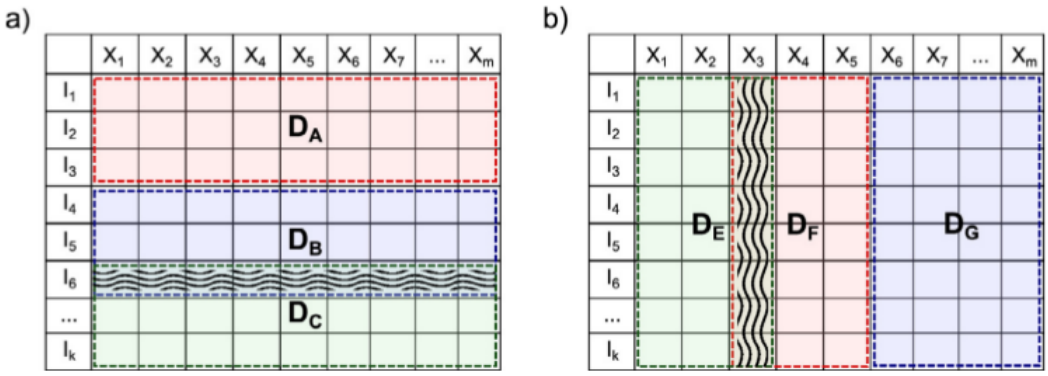
Um exemplo de situação de partição horizontal é quando deseja-se realizar a classificação de e-mails localizados em diferentes servidores. Neste caso temos a informação completa de cada instância (e-mail) armazenada em um dado servidor, porém existem múltiplos servidores e cada qual com um conjunto de instâncias distintas, isto é, nem todos os e-mails estão centralizados em um único local. Desta forma, classificadores locais poderiam ser treinados com um conjunto igual e mais abrangente de atributos, mas com dados referentes a diferentes instâncias.

Por outro lado, existe a partição vertical dos dados, que corresponde a diferentes fontes com uma visão parcial sobre o mesmo conjunto de instâncias, isto é, cada fonte conhece apenas um subconjunto das atributos descritivos das instâncias no conjunto de dados. O particionamento vertical também é conhecido como um ambiente de fontes heterogêneas. Pode-se observar este tipo de situação em sistemas de detecção de fraudes, nos quais cada instituição financeira armazena um conjunto de características (atributos)

sobre os seus clientes e não deseja compartilhar os dados dos seus clientes diretamente, ainda que dados de um mesmo cliente possam ser coletados por múltiplas instituicões. Neste caso, um modelo de classificação pode ser treinado localmente a partir de um conjunto abrangente de instâncias, mas considerando apenas o subconjunto de seus atributos ao qual a instituição tem acesso.

Genericamente, os dois tipos de particionamento podem ser entendidos conforme mostra a Figura 3.1. A Figura 3.1-a) representa o caso do particionamento homogêneo. Cada uma das fontes D_A, D_B e D_C possui todas os atributos (colunas) de um subconjunto de instâncias (linhas). Já a Figura 3.1-b) representa o cenário heterogêneo, com partição vertical dos dados, pois cada local D_E, D_F e D_G possui informações sobre todas as instâncias, mas apenas de um subconjunto das atributos que as descrevem.

Figura 3.1: Tipos de particionamento de dados em classificação descentralizada. a) Particionamento horizontal ou homogêneo. b) Particionamento vertical ou heterogêneo.



Fonte: (RECAMONDE-MENDOZA; BAZZAN, 2016)

3.2 Abordagens para classificação descentralizada de dados

Como mencionado nos estudos de Modi and Kim (2005) e Zeng et al. (2012), o problema da classificação em um cenário de partição vertical dos dados não possui muitos estudos relacionados. A literatura existente refere-se principalmente a métodos de *ensemble*, uma estratégia já bem estabelecida para melhorar a performance de algoritmos de AM (POLIKAR, 2006) e que tem sido a base da grande maioria dos trabalhos relacionados a AM distribuída (PETEIRO-BARRAL; GUIJARRO-BERDIÑAS, 2013). Além disso, apresentam-se propostas de arquiteturas baseadas em agentes para cenários específicos que se caracterizam por dados distribuídos, que poderiam ser generalizadas

para outros contextos em AM descentralizada (DEVI, 2014; ZENG et al., 2012; CAO; WEISS; YU, 2012). Nesta seção, são apresentadas as duas classes principais de abordagens encontradas na literatura para o problema de classificação descentralizada, baseadas em aprendizado *ensemble* e sistemas multiagentes (DEVI, 2014).

3.2.1 Arquitetura baseada em aprendizado *ensemble*

O objetivo dos métodos de aprendizado *ensemble* é combinar as predições de vários classificadores com o intuito de melhorar a qualidade da classificação, baseando-se no princípio de que esta combinação pode generalizar de forma mais precisa para diferentes exemplos de entrada. Agrupando vários métodos, a ideia é que o algoritmo global consiga utilizar as melhores características de cada modelo, o que não só tem o potencial de melhorar o desempenho geral do sistema, mas também evita a escolha de um único algoritmo para o qual não há garantia de bom desempenho no domínio abordado (POLIKAR, 2006).

De acordo com Polikar (2006), a estratégia utilizada em sistemas *ensemble* é portanto criar múltiplos classificadores a partir de variações nos dados ou no algoritmo de treinamento, e combinar a saída destes modelos a fim de melhorar o desempenho em relação a classificadores individuais. Rokach (2010) elenca uma série de estratégias para realizar aprendizado a partir de um *ensemble* de algoritmos, discutindo métodos de agregação, alguns dos quais serão abordados na Seção 3.3. Embora esta abordagem seja originalmente utilizado com outro propósito, isto é exatamente o que deseja-se para a agregação em cenários distribuídos, tal que aprendizado *ensemble* têm sido a estratégia mais recorrente na literatura relacionada a AM descentralizada (DEVI, 2014).

Tipicamente, estes múltiplos modelos do sistema *ensemble* são treinados a partir de fontes homogêneos de dados, no entanto, conforme discutido na literatura é possível adaptar esta estratégia para fontes heterogêneos de dados, ainda que este seja um cenário mais desafiador visto que cada modelo é inferido a partir de um conhecimento incompleto a respeito do conjunto de dados, representado pelo subconjunto de atributos que está disponível no local (DEVI, 2014). Naturalmente, pode-se estender estes métodos para o cenário aqui estudado no qual os dados estão fisicamente distribuídos, assumindo que diferentes modelos podem ser gerados em cada sítio ou fonte de dados e posteriormente agregados utilizando estratégias de combinação de modelos em sistemas *ensemble*. Partindo desta ideia, pode-se imaginar que possivelmente o desempenho obtido pelo sis-

tema *ensemble* pode ser interessante, mesmo não possuindo todos os atributos, como é observado em trabalhos que adotam aprendizado *ensemble* para tarefas clássicas de AM centralizada.

Alguns trabalhos de métodos de agregação utilizando aprendizado *ensemble* foram propostos na literatura, como por exemplo o estudo de Bay (1998), já abordado na seção sobre KNN para problemas multiclasse. Neste trabalho, o autor utiliza votação simples para a decisão da classe de saída entre múltiplos classificadores treinados com o algoritmo KNN, cada qual sobre um subconjunto dos atributos disponíveis. Outros trabalhos que abordam especificamente o problema de AM descentralizada com sistemas *ensemble* foram propostos, conforme discutido em Peteiro-Barral and Guijarro-Berdiñas (2013) e Devi (2014), alguns dos quais serão revisados na Seção 3.4 de acordo com a similaridade com o problema abordado neste trabalho, de classificacão descentralizada em cenários com múltiplas classes.

3.2.2 Arquitetura baseada em sistemas multiagentes

Sistemas multiagentes são compostos de múltiplos elementos computacionais, denominados agentes (WOOLDRIDGE, 2009). Cada agente possui a habilidade de: i) realizar ações de forma autônoma, ou em outras palavras, decidir por si próprio o que é necessário fazer para satisfazer seus objetivos, e ii) interagir com os demais agentes.

Partindo desta definição, pode-se entender o problema de classificação descentralizada como um sistema multiagente, com a premissa de que cada classificador local satisfaz as propriedades necessárias para ser denominado agente. Isto é, conseguem decidir a saída, isto é, a classe, de um exemplo por si próprio e conseguem comunicar esta decisão aos demais classificadores do sistema.

O contexto da literatura sobre o assunto refere-se principalmente a tarefas de mineração de dados distribuída, em que a arquitetura utilizada pode ser estendida para algoritmos de AM que resolvem a classificação. Cheung et al. (1996) mencionam que, de uma forma geral, uma tarefa de mineração de dado distribuída consiste de duas etapas principais: computação de um modelo local em cada sítio, e posterior fusão ou agregação dos resultados observados em cada localidade.

Devi (2014) aponta que sistemas multiagente adequam-se perfeitamente para este tipo de situação, envolvendo a resolução de uma tarefa (neste caso, classificação) de forma descentralizada, visto que sistemas multiagente foram originalmente propostos como uma

estratégia de solução de problemas em ambientes distibuídos. As computações locais são realizadas por agentes distintos, e então um agente agregador executa as funções de agregação dos resultados. Alguns trabalhos relacionados tratam do problema de classificação descentralizada utilizando este tipo de arquitetura, conforme revisado por Devi (2014) e Recamonde-Mendoza and Bazzan (2016). Na Seção 3.4 serão detalhadas trabalhos que propõem abordagens baseadas em sistemas multiagentes para o problema de classificação com particionalmente vertical de dados, em problemas binários ou multiclasse.

3.3 Métodos de agregação

Para executar as técnicas descritas acima (aprendizado *ensemble* ou sistemas multiagente), faz-se necessária uma forma de agregar os resultados observados em cada classificador treinado localmente, a fim de gerar uma predição global. Esta seção aborda os principais métodos de agregação adotados na literatura, os quais foram agrupados em dois conjuntos conforme as estratégias aplicadas: métodos algébricos ou métodos baseados em funções de escolha social.

3.3.1 Métodos algébricos

Polikar (2006) cita diversas formas possíveis para realização de agregação em sistemas de aprendizado *ensemble*. Uma das formas mais aplicadas para elaboração desta tarefa é a realização de cálculos simples envolvendo as saídas de cada modelo, sendo estas referentes às probabilidades conferidas à cada classe. Algumas das opções estudadas na literatura são as seguintes:

- Média aritmética simples: Computa-se a média das probabilidades observadas em cada modelo para todas as classes. Seleciona-se a classe que retorna o maior valor.
- Média Ponderada: Consiste em realizar uma média entre os resultados, porém conferindo pesos para cada um dos modelos, e opcionalmente também para cada classe dentro de um modelo. Usando informações como qual dos modelos é mais eficiente para determinadas classes, os pesos podem ser ajustados de forma a obter desempenho global superior.
- Regra do Produto: Dado um problema com N classes e K modelos, e as probabilidades $p(i, k)$ para cada par da i-ésima classe com o k-ésimo modelo, define-se:

$$v(i) = \prod \left(p(i,k)\right) \forall\ k \tag{3.1}$$

A classe de saída é definida como aquela que possui maior valor $v(i)$.

- Regra do Máximo: para todas as classes, verifica-se qual o classificador que atribui o maior valor para o exemplo, e confere-se este valor para a respectiva classe. A saída é a classe com maior valor observado.
- Regra do Mínimo: nos mesmos moldes da Regra do Máximo, porém verifica o menor valor atribuído para cada classe sobre todos os classificadores, e define como saída a classe com o maior valor dentre estes observados.

3.3.2 Funções de escolha social

Funções de escolha social, ou *Social Choice Functions* (SCF), têm o objetivo de agregar a preferência de vários indivíduos em uma única, de forma a estabelecer uma decisão do grupo (ARROW; MASKIN, 2012). Elas podem ser vistas como um conjunto de regras que visa definir o resultado de uma eleição. No caso aqui estudado, os eleitores são os agentes ou classificadores locais, os quais dependendo da arquitetura do sistema, definem uma classe de preferência para cada exemplo de entrada ou uma ordem de preferência entre os exemplos a respeito de cada classe.

Em uma analogia com relação a uma eleição política, as classes podem ser entendidas como cargos os quais cada candidato (exemplo de entrada) pode assumir. A eleição refere-se a escolher um candidato vencedor para cada cargo a partir de consulta a um conjunto de votantes (modelos de classificação). Para isso, utilizam-se as Funções de Escolha Social, realizando um sistema de votação entre a opinião de todos os votantes, os quais indicam sua ordem de preferência em função dos candidatos analisadas. Com uma ordem de preferência acordada entre os votantes, pode-se definir uma classe ganhadora para cada candidato ou exemplo de entrada (definido formalmente abaixo).

Para realizar a agregação dos modelos, podem-se utilizar diversas funções estudadas na literatura de sistemas eleitorais (ARROW; MASKIN, 2012; SCIENCE, 2012; BRANDT et al., 2016), tais como:

- Pluralidade: votação simples para cada exemplo de entrada. Cada modelo retorna qual classe acredita ser o melhor rótulo para determinada instância, e a classe com mais votos é atribuída à mesma. Para problemas de classificação binária, com ape-

nas duas classes possíveis, é também chamada de votação majoritária.

- Votação majoritária ponderada: Uma variação da votação majoritária na qual os votos de cada classificador são ponderados pelo seu desempenho geral, tal que classificadores mais qualificados (isto é, com predições mais precisas) tendem a ter maior peso na decisão final.
- Voto transferível simples: são computados os votos de todos os exemplos apenas se estão em primeiro em cada ranking. Após isso, remove-se o exemplo com menos votos, e os seus votos são transferidos para a próxima opção de cada eleitor que votou no mesmo. Este processo é repetido até que reste apenas um candidato (exemplo), que será o vencedor.
- Borda Count: cada instância recebe uma pontuação de acordo com a quantidade de candidatos rankeados abaixo da mesmo. Com N instâncias, uma instância I tem uma pontuação $P(i) = N - L_i$, L_i sendo a posição da instância em um ranking. A pontuação final de uma instância é dada pela soma de todas as P(i), isto é, da pontuação desta instância em todos os rankings.
- Função de Dowdall: é uma modificação do método de Borda Count, em que a pontuação $P(i)$ é dada por $(N - L_i)/L_i$. Ela pode ser vista como uma forma de dar pesos maiores para as posições de cima dos rankings.
- Função de Copeland: realiza uma "eleição"entre todos os pares de indivíduos. A pontuação final de um indivíduo é dada pelo número de vitórias menos o número de derrotas dentre todas as eleições em que participou.

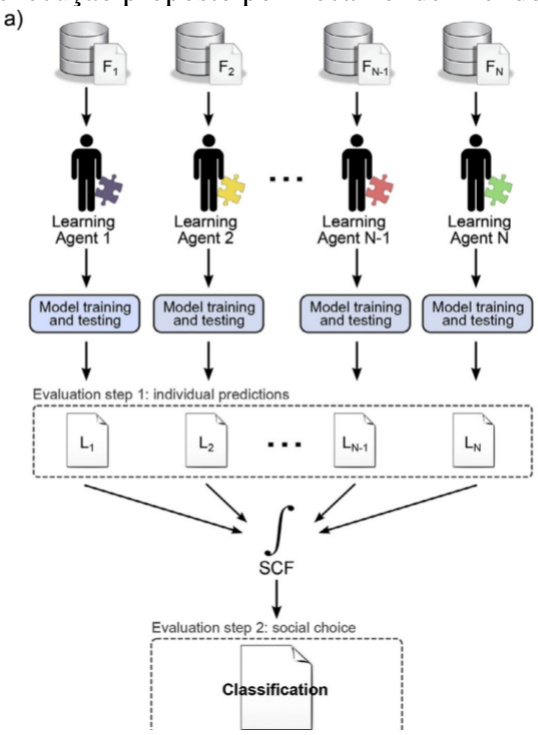
Muitos trabalhos da literatura relacionada que utilizam mecanismos de votação para agregação de modelos ou classificadores adotam o método da votação simples ou da votação majoritária ponderada entre as entidades. Uma votação simples para extrair um consenso entre múltiplos classificadores a respeito da classe predita para uma dada instância é a função de escolha mais simples e direta possível para se obter uma solução neste cenário. Porém, pode-se explorar mecanismos mais sofisticados que este, sem comprometer o desempenho e restrições de comunicação entre modelos presentes na aplicação conforme discutido em trabalhos anteriores (RECAMONDE-MENDOZA; BAZZAN, 2016).

3.4 Trabalhos relacionados

Recamonde-Mendoza and Bazzan (2016) propõem a utilização de um sistema multiagente para lidar com a tarefa de classificação binária com particionamento vertical dos dados. No sistema proposto, há dois tipos de entidades: agentes autônomos, representando cada classificador local; e um agente facilitador, que calcula as funções de agregação dos resultados observados nos agentes anteriores.

No artigo, propõe-se a computação de um modelo em cada local, e posteriormente a agregação dos mesmos utilizando funções mais sofisticadas de escolha social, ao invés da forma mais simples e intuitiva, que é a votação simples entre as soluções. Dentre as funções utilziadas, estão a Borda Count e Função de Copeland, conforme revisado na seção anterior. Os agentes autônomos podem utilizar diferentes algoritmos de classificação para computação de suas soluções locais. Após a etapa de construção e computação dos resultados de cada modelo, o agente facilitador é utilizado. Sua função é analisar os valores das probabilidades calculados por cada agente, as quais quanto mais próxima de 1.0 indicam maior probabilidade de um exemplo pertencer à classe alvo (lembrando que trata-se de um problema e classificação binária), e então decidir a partir de alguma regra qual será a saída.

Figura 3.2: Fluxo de execução proposto por Recamonde-Mendoza and Bazzan (2016)

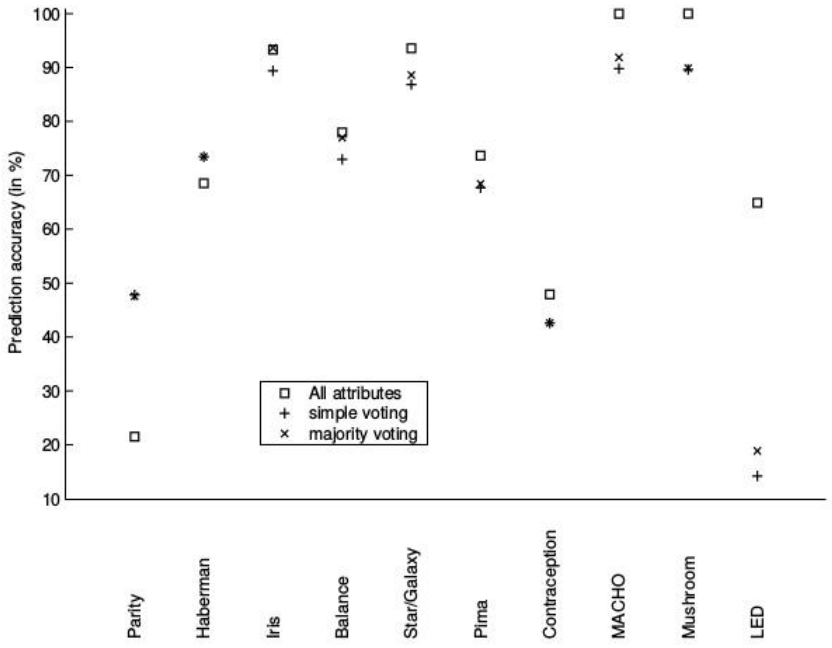


Fonte: (RECAMONDE-MENDOZA; BAZZAN, 2016)

A Figura 3.2 demonstra o processo de classificação adotado por Recamonde-Mendoza and Bazzan (2016), sendo $F_1 ... F_N$ os modelos de classificação embutidos em cada agente e $L_1 ... L_N$ o resultado observado, representado por um vetor de probabilidades. As autoras analisaram a classificação descentralizada de dados em um domínio da Bioinformática e constataram que o método proposto gerou bons resultados em comparação com abordagens de classificação centralizada ou agregação de modelos treinados de forma descentralizada pelo método de votação simples.

McConnell and Skillicorn (2004) desenvolveram um trabalho que utiliza um sistema similar ao sistema multiagente descrito neste trabalho. Assim como em Recamonde-Mendoza and Bazzan (2016), primeiramente os modelos locais são computados, sendo neste caso treinados com o algoritmo de Árvores de Decisão, que se adapta a casos multiclasse. Após este primeiro passo, a agregação é feita com votação simples entre os modelos. Para avaliar os resultados, os autores executaram o método com alguns conjuntos de dados obtidos do UCI (LICHMAN, 2013), conforme mostra a Figura 3.3. Como podemos observar, para a maioria dos conjuntos de dados analisados, a taxa de acertos da abordagem descentralizada proposta é bastante próxima dos resultados obtidos com classificação centralizada de dados.

Figura 3.3: Resultados obtidos por McConnell and Skillicorn (2004) para conjuntos de dados do UCI. O eixo y representa a porcentagem de acerto para cada conjunto de dados.



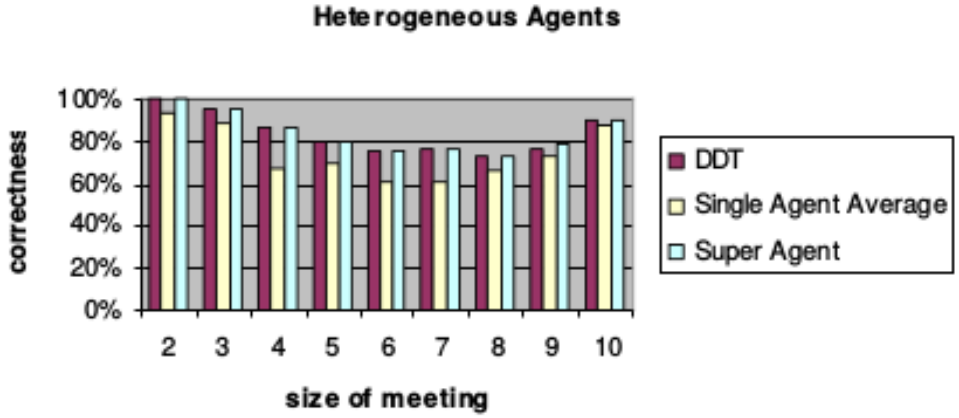
Fonte: (MCCONNELL; SKILLICORN, 2004)

Moghadam and Ravanmehr (2017) propuseram uma abordagem de mineração de

dados distribuída com a motivação de analisar dados que são gerados em diferentes locais, distribuídos geograficamente. Assim como os dois trabalhos anteriores, o sistema proposto utiliza uma arquitetura baseada em agentes, computando soluções locais e então agregando-as através de um método de combinação pré-definido. O problema de pesquisa descrito é de classificação binária e envolve o uso das informações de estações metereológicas localizadas no Irã, as quais na totalidade possuem mais de 20 milhões de registros particionados verticalmente entre múltiplos sítios. Neste cenário, a comunicação de todas as informações entre os diferentes locais é inviável, assim surgindo o contexto para utilização de sistemas descentralizados. Como algoritmo os autores utilizam uma adaptação de árvores de decisão, e diferentemente das abordagens anteriores a agregação entre modelos é realizada em termos de compartilhamento dos melhores valores para divisão de nós no processo de treinamento das árvores de decisão, ao invés de entre saídas fornecidas por estes modelos. A avaliação dos resultados foi feita baseada no tempo de execução da tarefa de mineração de dados, apontando que a abordagem proposta apresenta desempenho melhor que outras opções vistas na literatura.

Modi and Kim (2005) realizaram um trabalho que trata o problema da classificação em casos que apresentam particionamento vertical de dados. A estratégia proposta é uma adaptação de árvores de decisão, na qual calcula-se o ganho de informação para cada um dos atributos independentemente dos demais. Para a agregação, cada agente comunica aos outros o caminho observado localmente na árvore de decisão, e posteriormente verificam se o seu caminho pode ser melhorado pelas informações contidas nos outros.

Figura 3.4: Resultados obtidos por Modi and Kim (2005) para um cenário de classificação binária com particionamento vertical dos dados.



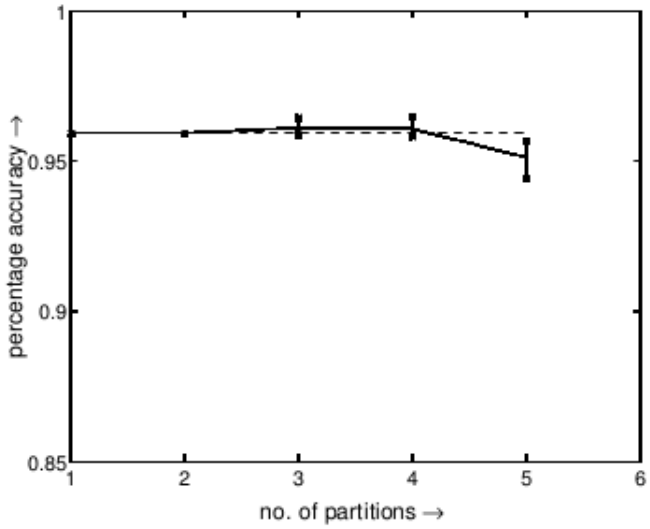
Fonte: (MODI; KIM, 2005)

Com a abordagem descrita, os pesquisadores conseguiram chegar em ótima taxas

de predição, muito próximas ou iguais às observadas em um cenário com dados centralizados, conforme sumariza a Figura 3.4. No gráfico, as colunas roxas representam o resultado do algoritmo proposto para classificação descentralizada com particionamento vertical dos dados. As barras amarelas representam a média observada entre todos os classificadores locais, com visão parcial dos atributos do conjunto de treinamento. Por fim, fez-se a comparação com um agente que teoricamente tivesse acesso ao conjunto completo de dados de treinamento (colunas azuis), representando um cenário de classificação centralizada.

Basak and Kothari (2004) tratam exatamente do mesmo problema para o qual este trabalho busca propor uma solução: tarefa de classificação multiclasse em um cenário distribuído, com particionamento vertical dos dados. A abordagem descrita no artigo compreende a utilização das probabilidades de cada indivíduo pertencer a determinada classe, dada por cada um dos modelos locais. O método proposto consiste em combinar as probabilidades observadas em cada modelo para uma dada entrada utilizando funções de aproximação definidas através de probabilidades posteriores. Em resumo, verifica-se, por exemplo, qual a probabilidade de uma instância **x** estar na classe i, e os pressupostos são as probabilidades estimadas por cada modelo.

Figura 3.5: Resultados obtidos por Basak and Kothari (2004) para o conjunto de dados Optical Digits.



Fonte: (BASAK; KOTHARI, 2004)

Os autores utilizaram como classificadores nos modelos locais os algoritmos KNN, Árvores de Decisão e Perceptron de Múltiplas Camadas, revisados na Seção 2.2.2.1, e realizaram a avaliação de desempenho da abordagem proposta em diversos conjuntos de

dados obtidos do UCI (LICHMAN, 2013). Seus resultados são interessantes pois apresentam uma perda de desempenho muito pequena com relação aos resultados de algoritmos centralizados. A Figura 3.5 representa os resultados para um dos conjuntos de dados analisados, Optical Digits, na qual o eixo x representa a quantidade de partições, e consequentemente de modelos, presentes na classificação descentralizada.

Em um contexto diferente, porém relacionado, temos o artigo publicado por Yu, Vaidya and Jiang (2006). O problema analisado pelos autores consiste na classificação de dados com particionamento vertical, nos quais o motivo do particionamento é especificamente a segurança. Resumidamente, propõe-se uma espécie de codificação dos valores dos atributos em uma matriz de cada modelo local, e então as matrizes são combinadas, de forma que não se consiga reverter para os valores originais. A partir disso, faz-se uma classificação global dos dados.

4 ABORDAGEM PROPOSTA

Devi (2014) menciona que o processo clássico de elaboração de algoritmos para ambientes distribuídos requer a coleta dos dados em uma central, para posterior aplicação de um algoritmo. Contudo, esta abordagem não é aplicável em diversas situações, por motivos como o uso de dados sensíveis, custos de comunicação de grandes volumes de dados, entre outros.

Como visto nas seções anteriores, poucos trabalhos lidam com a classificação multiclasse em ambientes distribuídos. Além disso, os poucos que utilizam algum método de consenso, se limitam a regras simples de agregação como votação por pluralidade, sem explorar diversas formas. E para o trabalho que analisa diversas funções de agregação, baseadas em funções de escolha social, as soluções propostas são válidas apenas para o caso de classificação binária (RECAMONDE-MENDOZA; BAZZAN, 2016).

Assim, o método aqui descrito analisa uma configuração que diverge em alguns pontos sobre cada estudo já realizado, sendo especificamente um cenário de classificação descentralizada em problemas multiclasse. Nas seguintes seções, aborda-se em detalhes as estratégias e a metodologia envolvidas na construção desta abordagem.

4.1 Descrição da abordagem

No cenário analisado neste trabalho, define-se um agente classificador como um modelo local de classificação. Cada um destes agentes treina o seu modelo local e realiza a classificação de novos exemplos baseando-se nos atributos disponíveis em seus respectivos domínios. Do ponto de vista da arquitetura, a abordagem proposta segue a estrutura utilizada por Recamonde-Mendoza and Bazzan (2016), com múltiplos agentes autônomos treinando modelos de classificação locais a partir de uma partição vertical do conjunto de treinamento. No entanto, diferente do trabalho anterior, a abordagem aqui proposta está voltada à classificação multiclasse.

Definindo, $f(\mathbf{x}, A)$ é a função resultante da análise de um agente classificador A, para uma instância de entrada $\mathbf{x}$. Sendo N o número de classes envolvidas no problema, a função $f(\mathbf{x}, A)$ tem como retorno um vetor de probabilidades com tamanho N, no qual cada posição i se refere a probabilidade da entrada pertencer à respectiva classe C_i. Assim, a classe predita por um determinado modelo para uma instância de entrada é definida

como

$$\underset{i}{\operatorname{argmax}} \left(f(\mathbf{x}, A)\right) \tag{4.1}$$

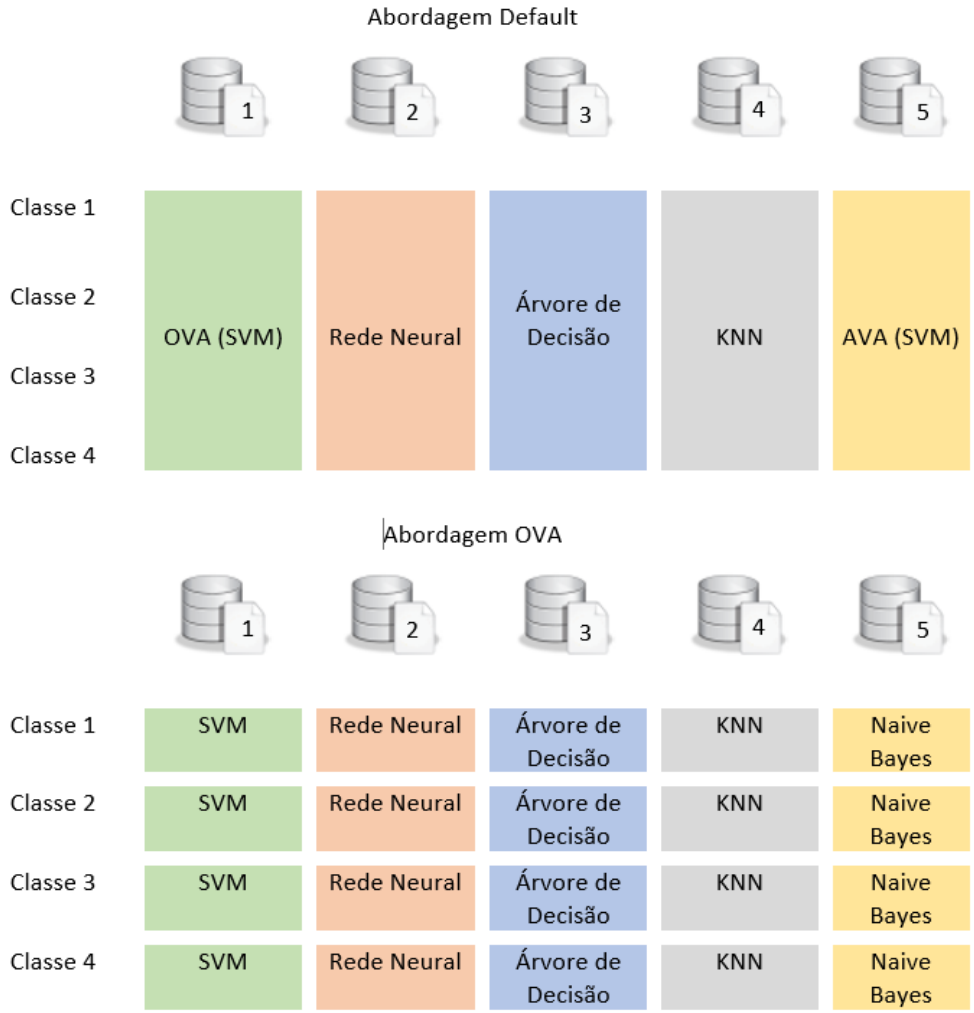
sendo i o índice da posição com probabilidade máxima (referente a uma classe $C_i, ..., C_N$ dentre as N possíveis). Em alguns tipos de modelos, como árvores de decisão, que geram diretamente uma classe resultado, $f(\mathbf{x}, A)$ apresenta o valor "1" para a classe predita pelo modelo, e "0" para as demais.

Os agentes classificadores podem utilizar diferentes algoritmos ou estratégias de classificação a fim de treinar os modelos para computação de suas soluções locais. No presente trabalho, analisamos duas formas diferentes de realizar a tarefa de classificação descentralizada em problemas multiclasse, as quais denominamos de abordagem ***default*** e abordagem ***OVA***:

- **Abordagem "*default*"**: cada agente encapsula um dos métodos estudados no Capítulo 2 para classificação multiclasse, baseados em decomposição binária ou em extensão direta, a fim de realizar o treinamento de seu modelo local. Por motivos de simplicidade e disponibilidade de implementação, foram escolhidos os seguintes métodos para execução dos experimentos realizados neste trabalho, sendo cada qual executado por um agente classificador:
  - OVA (utilizando SVMs em cada sub classificador);
  - AVA (utilizando SVMs em cada sub classificador);
  - Árvores de decisão;
  - KNN (com k=5);
  - Redes Neurais.
- **Abordagem "*OVA*"**: cada agente realiza a classificação multiclasse pelo método de decomposição binária, adotando a estratégia OVA para tanto. Assim, N classificadores são treinados em cada agente classificador ou partição, um para cada classe possível. A abordagem OVA também pode ser entendida como executar K classificadores OVA, sendo K o número de agentes classificadores, mas com algoritmos de classificação binária diferentes em cada um. Neste trabalho são utilizados como classificadores binários, em cada partição de dados, os seguintes métodos:
  - Árvores de Decisão;
  - KNN (com k=5);
  - Naive Bayes;
  - Redes Neurais;

- SVM.

Figura 4.1: Comparação da arquitetura do sistema para as abordagens implementadas: abordagem "*default*" e abordagem "*OVA*. Cada caixa colorida representa um classificador treinado com sua respectiva partição vertical dos dados.



Fonte: Autor

A Figura 4.1 exemplifica e compara as duas abordagens para um cenário contendo 4 classes. Nos experimentos realizados, o número de classes varia conforme o conjunto de dados utilizado (mais detalhes sobre características dos dados explorados serão dados na Seção 4.3). Nesta figura, cada coluna representa um sítio ou base de dados com sua respectiva partição vertical dos dados originais, e cada caixa colorida representa um classificador treinado com sua respectiva visão parcial dos atributos. Como é possível observar, na abordagem "*default*" cada sítio de dados possui apenas um modelo treinado, apto a classificar as instâncias em uma dentre as N classes do problema (para este exemplo, $N = 4$). Já na abordagem "*OVA*", cada sítio possui N classificadores, onde cada

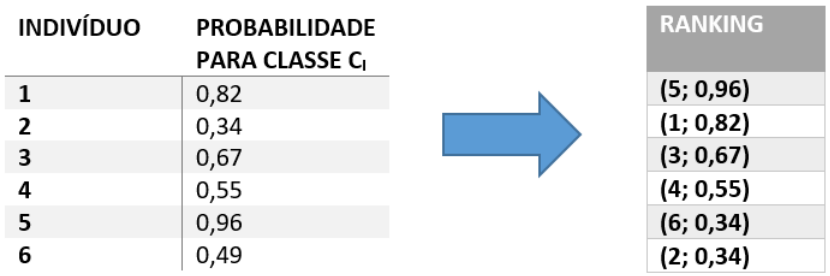
classificador é treinado para distinguir instâncias de uma classe i das demais $N-1$ classes.

Cabe salientar que neste trabalho, o estudo a respeito da melhora de performance em cada classificador específico não foi realizado, visto que não se trata de nosso objetivo. Assim, todos os treinamentos dos modelos utilizam os valores padrões para cada parâmetro dos algoritmos mencionados, conforme definido nas implementações adotadas.

Após a etapa de treinamento dos modelos locais com base na visão parcial de cada agente classificador, e a posterior aplicação destes modelos para predição das classes de novas instâncias, um agente facilitador é utilizado. Sua função é analisar os valores das probabilidades de cada classe conforme calculados por cada agente, e então decidir a partir de um método de agregação qual será a classe de saída para os exemplos de entrada. Em uma tarefa de classificação, define-se uma série de exemplos, ou instâncias, de entrada O, cada qual descrito pelo seu vetor de atributos, os quais deseja-se classificar.

A fim de viabilizar a análise pelo agente facilitador, em cada agente classificador realiza-se um processamento como definido a seguir. Para cada classe C_i, calcula-se um ranking entre as instâncias O, de acordo com a probabilidade de cada uma deles ser da classe C_i. Em outras palavras, este ranking reflete uma ordem de preferência do agente em função da associação das instâncias de entrada com uma classe C_i. Como por exemplo, caso uma instância I_1 tenha probabilidade 0.9 de pertencer à classe C_i (de acordo com o modelo do agente), e uma outra instância I_2 tenha probabilidade 0.85 de pertencer a mesma classe, a instância I_1 estaria acima de I_2 neste ranking. Na Figura 4.2, demonstra-se a criação de um ranking em um cenário com 6 indivíduos (exemplos).

Figura 4.2: Elaboração do ranking por um agente para uma dada classe C_i.



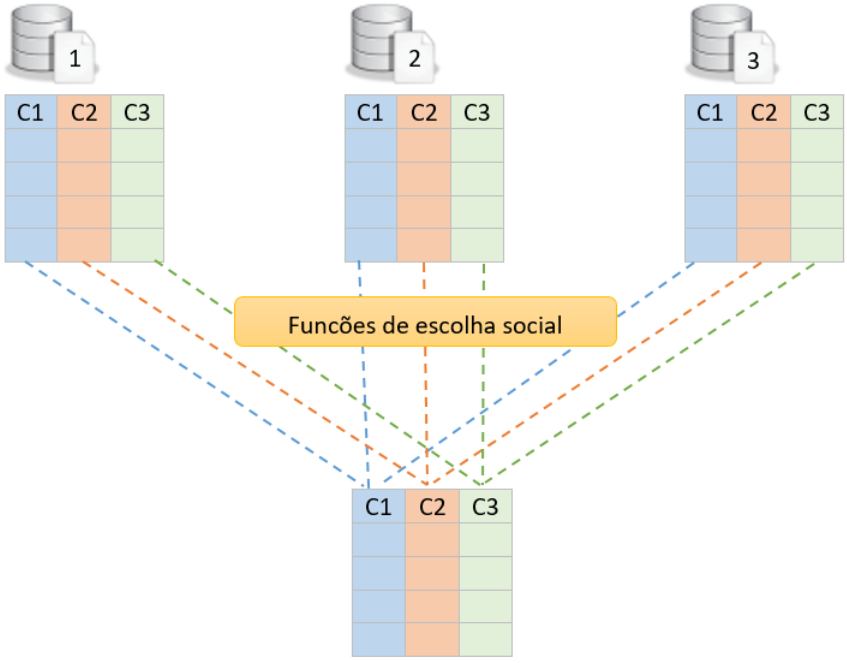
Fonte: Autor

Partindo da descrição acima, após a execução dos agentes classificadores obtém-se $N \times K$ rankings, sendo K o número de agentes classificadores no sistema de classificação descentralizada e N o número de classes envolvidas no problema abordado. É importante

notar que a probabilidade obtida para cada classe varia entre os múltiplos agentes classificadores em razão das diferenças no treinamento de seus respectivos modelos referentes aos dados de treinamento e algoritmo de classificação adotados. Adicionalmente, para a abordagem "*default*" as probabilidades são obtidas para cada classe por modelos que suportam a análise simultânea de múltiplas classes, enquanto que para a abordagem "*OVA*" as probabilidades calculadas para cada classe refletem a probabilidade de uma instância pertencer àquela classe em comparação com as demais $N - 1$ classes tomadas em conjunto. No entanto, em ambos os casos o agente facilitador realiza a agregação dos rankings de todos os modelos, combinando-os por classe através de Funções de Escolha Social. Neste trabalho, exploraram-se as funções Borda Count, Copeland, Dowdall e Pluralidade, revisadas na Seção 3.3.2. A partir disto, cada instância recebe uma pontuação "consenso" para cada classe, formando novamente N rankings entre todas as instâncias (um ranking para cada classe). Por fim, para cada um destas instâncias, a classe cujo ranking apresenta esta instância com maior pontuação será a classe dada como predição final do sistema.

Na Figura 4.3 tem-se uma representação da forma com que os rankings gerados por múltiplos agentes classificadores são agregados pelo agente facilitador para o cenário de classificação multiclasse.

Figura 4.3: Agregação de rankings formados a partir das probabilidades preditas na abordagem proposta.



Fonte: Autor

4.2 Metodologia adotada

Nesta seção aborda-se de forma mais específica os passos envolvidos na metodologia adotada para implementação da abordagem proposta, detalhando o conjunto de algoritmos utilizado neste trabalho e o que cada etapa realiza em termos de computações.

Podemos resumir a abordagem implementada com o pseudocódigo dado no Algoritmo 1, o qual retorna a predição da classe das instâncias contidas em um conjunto de dados de entrada. Fornecendo como entrada o conjunto de dados a ser utilizado para classificação e uma função de escolha social, realizam-se os seguintes passos:

Algorithm 1 Pseudocódigo da abordagem proposta para classificação multiclasse descentralizada

Input: *Dataset*: conjunto de dados; *SCF*: função de escolha social
Output: Classe predita para cada instância de entrada

1: $D \leftarrow$ *parseDataset*$(Dataset)$
2: $P \leftarrow$ *ParticionarDados*(D) ▷ *P* é uma lista de partições, com subconjuntos dos atributos que os modelos utilizam
3: $rankings \leftarrow \emptyset$ ▷ *rankings* armazenará todos os rankings locais gerados
4: **para cada** p **em** P :
5: $vetoresDeProbabilidades \leftarrow$ *Classificacao*(p)
6: $rankingLocal \leftarrow$ *CriaRanking*($vetoresDeProbabilidades$)
7: $rankings \leftarrow$ *AdicionaRanking* ($rankings, rankingLocal$)
8: $rankingsAgregados \leftarrow$ *AgregaRankings*($SCF, rankings$)
9: **para cada** $instancia$:
10: $Print$ *getClasseVencedora* ($rankingsAgregados$)

A etapa de *parseDataset* (linha 1) compreende eventuais transformações nos conjuntos de dados para serem portados à abordagem aqui desenvolvida. Após isso, os dados são particionados, de forma a conferir um subconjunto dos atributos disponíveis em cada partição (linha 2), as quais são então passadas para agentes classificadores que realizam a classificação de instâncias a partir de seus modelos locais (linha 5). Uma vez que as classificações são desempenhadas, ocorre a criação dos rankings (linhas 6 e 7) e em seguida é realizada a aplicação da função de escolha social para agregação dos rankings gerados por todos os classificadores (linha 8). Por fim, para cada instância analisa-se as probabilidades preditas para cada classe pela abordagem baseada em funções de escolha social a fim de definir a saída do sistema (linhas 9 e 10). Assim, as operações descritas nas linhas 4 a 7 são realizadas por agentes classificadores, enquanto as operações realizadas nas linhas 8 a 10, bem como toda a preparação e particionamento dos dados, são realizadas pelo agente facilitador.

A próxima seção contém explicações mais detalhadas sobre cada parte específica deste processo.

4.2.1 Fluxo de Execução

Dado que um conjunto de dados possua F atributos para cada instância, define-se o número de partições ou de agentes classificadores K a partir da seguinte fórmula:

$$K = \min((F/2), 5) \tag{4.2}$$

Idealmente, obtem-se K = 5, então cada um algoritmos de classificação mencionadas na seção anterior pode ser utilizado. Em casos nos quais os dados possuem poucos atributos, colocar a disposição de cada classificador apenas um atributo não é interessante, então cada um fica com metade dos atributos devido ao limite $F/2$ estabelecido. No caso em que K < 5, utilizam-se os K primeiros classificadores da figura 4.1, da esquerda para a direita.

Cada um dos conjuntos de dados foi utilizado em diversos experimentos, visando comparar os resultados de diferentes funções de escolha social na agregação de modelos. O diagrama a seguir demonstra o fluxo das tarefas envolvidas na construção deste processo, as quais serão detalhadas na seções seguintes.

Figura 4.4: Fluxo de execução

Fonte: Autor

4.2.1.1 Preparação dos Dados

Esta etapa compreende a transformação dos conjuntos de dados em um formato que os classificadores possam tratar. Instâncias com valores nulos em algum de seus atri-

butos são descartadas, e a formatação é transferida para um arquivo no formato CSV. Além disso, o *label encoding* dos identificadores das classes é realizado, de forma a transformar os nomes das mesmas em números, de forma a possibilitar otimizações na performance dos classificadores.

4.2.1.2 Simulação de ambiente distribuído

Ainda que tarefas de classificação descentralizada sejam cada vez mais recorrentes nos dias de hoje, existe um acesso bastante limitado a dados gerados sob esta condição. Os dados disponíveis publicamente para treinamento e teste de algoritmos de classificação são gerados predominantemente em cenários centralizados, ou disponibilizados como tal. Assim, uma vez que tem-se todos os dados disponíveis em um único conjunto de treinamento, é preciso separar os atributos em partições distintas (cinco partições ou menos, dependendo do número de atributos, como explicado anteriormente) para simular o cenário descentralizado aqui abordado. Esta separação é feita de forma aleatória, de maneira que cada agente classificador (sítio de dados) obtenha o mesmo número de atributos. Esta estratégia de simulação de ambiente distribuído tem sido adotada por trabalhos anteriores, os quais realizam um particionamento "artificial" do conjunto de dados de treinamento (MCCONNELL; SKILLICORN, 2004; RECAMONDE-MENDOZA; BAZZAN, 2016).

4.2.1.3 Tarefas de classificação

Nesta etapa, cada agente classificador recebe como entrada os dados para treinamento, que consistem de um subconjunto dos atributos ao qual o agente tem acesso, conforme explicado acima. Relembrando, por se tratar de um particionamento vertical dos dados, cada partição engloba o mesmo conjunto de instâncias, mas com visões parciais e distintas a respeito das mesmas (isto é, de seus atributos). O treinamento dos modelos e a classificação de instâncias é então realizada seguindo uma das formas descritas na seção anterior: abordagem "*default*" ou abordagem "*OVA*". No entanto, independente da abordagem utilizada, a saída de um agente classificador será dada por N vetores de probabilidades, cada um referente a uma classe do problema. Cada um destes vetores possui o mesmo número de posições, referentes a cada instância analisada.

4.2.1.4 Elaboração dos Rankings

A partir dos vetores de probabilidades gerados no processo de classificação, cada agente classificador elabora N rankings, um para cada classe, de acordo com a probabilidade estimada para as instâncias no conjunto O pelo seu modelo de classificação embutido.

4.2.1.5 Agregação dos modelos

Para cada classe, seus K rankings devem ser agregados, formando scores únicos para cada instância analisada. Estes scores representam um consenso obtido pelos múltiplos agentes classificadores quanto à probabilidade de uma dada instância pertencer a cada uma das classes possíveis. Esta agregação é realizada pelo agente facilitador e adota métodos já apresentados na Seção 3.3.2. Neste trabalho, os seguintes métodos de agregação foram utilizados e comparados:

- Borda count
- Função de Copeland
- Função de Dowdall
- Pluralidade

Vale notar que neste fluxo de execução, como temos a criação de rankings, para que os mesmos tenham significância ou sejam viáveis em primeira instância, é necessário um conjunto de exemplos sendo analisados em "stream", isto é, durante a mesma tarefa de classificação. Neste caso, decidiu-se utilizar todas as instâncias em um conjunto de dados de uma só vez, tal que elas compõem o conjunto O de instâncias definidos previamente.

4.2.1.6 Definição da classe de saída

A etapa final do sistema implementado consiste em definir a classe predita para uma dada instância a partir dos N rankings gerados após agregação dos múltiplos modelos. Para cada instância, sua classe de saída é definida a partir do score consenso final calculado para cada uma dentre as N classes. Especificamente, a classe na qual a instância obtiver o maior valor de score, será a classe de saída.

4.2.2 Validação dos testes

Algumas características dos algoritmos implementados e dos conjuntos de dados podem fazer com que diferentes execuções apresentem resultados muito diferentes. Também faz-se necessário a seleção adequada de conjuntos de treinamento e teste independentes, buscando garantir a generalidade do modelo. Por isso, utilizaram-se algumas técnicas descritas a seguir para avaliar o desempenho do sistema na tarefa de classificação.

4.2.2.1 Validação cruzada

A validação cruzada, ou *cross-validation*, é uma técnica de validação de modelos que avalia a sua capacidade de generalização, isto é, visa avaliar seu desempenho com diferentes conjuntos de dados para estimar da forma mais precisa possível como o modelo iria responder a dados novos (desconhecidos). É util, por exemplo, para verificar se o modelo não possui *overfitting* com relação aos conjunto de dados de treinamento utilizado.

Mais especificamente, utiliza-se neste trabalho o método de k-fold cross-validation, um tipo de validação cruzada. Configurando $k = 10$, o método funciona da seguinte forma: dentro do conjunto de dados, divide-se as instâncias em 10 partes iguais (partições), tentando manter em cada uma destas a mesma proporção de classes visualizada no conjunto original. A partir daí, faz-se 10 execuções do algoritmos de classificação, onde cada uma delas utiliza como conjunto de teste uma das partições, e todas as outras como conjunto de treinamento. Assim, treina-se um total de k modelos com diferentes subconjuntos de instâncias como dados de treinamento, e cada uma das partições será utilizada como conjunto de teste independente exatamente uma vez. Por fim, o desempenho do algoritmo de classificação é estimado a partir do desempenho médio entre os k modelos treinados no processo de k-fold cross-validation. A Figura 4.5 representa um exemplo de k-fold cross-validation, com $k = 4$.

Figura 4.5: K-fold Cross-Validation

ITERAÇÃO	INDIVÍDUOS (4 partições)			
1	Teste	Treinamento	Treinamento	Treinamento
2	Treinamento	Teste	Treinamento	Treinamento
3	Treinamento	Treinamento	Teste	Treinamento
4	Treinamento	Treinamento	Treinamento	Teste

Fonte: Autor

No contexto deste trabalho, pode-se perceber que os atributos do conjunto de dados estão distribuídos entre os agentes classificadores de forma aleatória, conforme exposto anteriormente na Seção 4.2.1.2. Logo, existe a possibilidade de que conforme essa distribuição ocorra, alguns agentes classificadores fiquem com um subconjunto de atributos melhor do que outros, isto é, mais descritivos em relação às classes, ou que simplesmente se obtenha uma combinação particularmente vantajosa entre subconjunto de atributos e algoritmos de classificação embutido por um agente.

Com o objetivo de reduzir ao máximo a influência deste fator nos resultados, os modelos foram validados com base em 10 repetições do 10-*fold cross-validation* para proporcionar uma variação da distribuição de atributos entre os diferentes sítios de dados, e consequentemente pelo conjunto de treinamento utilizada por cada agente classificador. Após as 10 repetições do 10-*fold cross-validation*, o desempenho dos modelos é estimado a partir da média aritmética entre as medidas de desempenho calculadas para cada uma das repetições.

4.2.3 Medidas de desempenho

Em tarefas de classificação, existem diversas formas de avaliar a performance de um método sobre um determinado conjunto de dados. De acordo com Sokolova, Japkowicz and Szpakowicz (2006), as métricas de desempenho existentes avaliam diferentes características dos algoritmos de AM. Por isso, pode ser importante verificar o desempenho de um método com o auxílio de variadas métricas. No entanto, em todos os casos o cálculo destas métricas adota como base uma matriz de confusão obtida a partir da comparação entre a classe real e a classe predita para todos os exemplos analisados, definindo assim o conjunto de verdadeiros positivos (VP) e verdadeiros negativos (VN) como o conjunto de exemplos corretamente classificados como positivos ou negativos, respectivamente, e falsos positivos (FP) e falsos negativos (FN) como o conjunto de exemplos que foi incorretamente classificado em relação à sua classe real. A Figura **4.6** exemplifica esta matriz de confusão para um cenário com duas classes, Positivo e Negativo.

A partir da matriz de confusão, a alternativa mais simples e intuitiva para avaliação de modelos é a taxa de acertos média, ou acurácia, que consiste em calcular o número de exemplos previstos com sucesso, dividido pelo número total de exemplos analisados:

$$acuracia = \frac{VP + VN}{VP + VN + FP + FN} \tag{4.3}$$

Figura 4.6: Matriz de confusão para classificação binária

		Classe Prevista	
		Positivo	Negativo
Classe Real	Positivo	Verdadeiro Positivo	Falso Negativo
	Negativo	Falso Positivo	Verdadeiro Negativo

Fonte: Autor

Porém, conforme Sokolova, Japkowicz and Szpakowicz (2006) mencionam, a acurácia pode não representar bem o desempenho de um classificador, como por exemplo em casos de desbalanceamento de classes. Um exemplo facilmente visualizado é o caso de uma classificação binária, com 90% das instâncias sendo positivas, e 10% negativas. Caso um classificador erre todas as previsões para as negativas, ainda é possível obter uma acurácia satisfatória de 90%.

Assim, com o intuito de melhor avaliar o desempenho dos métodos sem risco de enviesamento pelas classes majoritárias, aplicou-se uma métrica denominada *F1 Score*. O cálculo desta métrica é realizado a partir de duas outras medidas de desempenho comumente adotadas em AM: precisão e revocação (*recall*). Precisão refere-se ao número de previsões positivas acertadas pelo algoritmo, isto é, à proporção dos elementos verdadeiramente positivos entre todos os elementos previstos como tal. Em outras palavras, define-se precisão como o total de exemplos VP dividido pela soma entre VP e FP.

Por sua vez, *recall* se refere à porção dos elementos pertencente positivos que o método foi capaz de prever como tal, sendo portanto calculada como o número de exemplos VP dividido pela soma entre VP e FN.

Intuitivamente, a precisão demonstra a habilidade do algoritmo de não classificar instâncias positivas como negativas, enquanto que o *recall* refere-se a habilidade do algoritmo de identificar todas as instâncias positivas (PEDREGOSA et al., 2011).

A partir disso, a fórmula do F1 é calculada como segue:

$$F1 = 2 \times \frac{precisao \times recall}{precisao + recall} \tag{4.4}$$

Em cenários de classificação multiclasse, pode-se realizar o cálculo do F1 Score de três formas distintas:

- F1 Macro: calcula o F1 Score para cada uma das classes separadamente, e então faz a média simples entre os valores obtidos.
- F1 Micro: calcula o F1 Score partindo do número total de VP, FP e FN obtidos considerando todas as classes tomadas em conjunto. No cenário analisado, será igual a taxa de acerto média, visto que cada instância pode pertencer a apenas uma classe.
- F1 Ponderado: calcula o F1 Score para cada uma das classes separadamente, e então calcula uma média ponderada entre estes valores, com os pesos atribuídos à cada classe de acordo com o número de instâncias pertencentes à classe em questão.

4.3 Dados de teste

Como pode-se observar na literatura analisada, não existe consenso sobre qual algoritmo de classificação funciona melhor em problemas centralizados, porque provavelmente não existe algum vencedor absoluto (alguns algoritmos funcionam melhor para determinados casos). Por este motivo, buscou-se utilizar conjuntos de dados variados para a realização dos experimentos, com o intuito de avaliar possíveis tendências e generalidade no desempenho das abordagens para classificação e métodos de agregação propostos.

Inicialmente, para realizar uma avaliação geral do desempenho de cada um dos métodos analisados, optou-se pela escolha de diferentes conjuntos de dados disponíveis no repositório UCI (LICHMAN, 2013), dado que os mesmos são bastante utilizados como *benchmarks* na área de AM. A maioria dos conjuntos de dados selecionados referem-se a dados já analisados em outros trabalhos relacionados, até mesmo para fins de comparação com as soluções já propostas. Outro motivo para a escolha de conjuntos desta fonte é que os mesmos são exemplos reais de dados, e cobrem diversos domínios (biologia, imagens espaciais, reconhecimento de dígitos, entre outros).

Além disso, priorizaram-se algumas características dos conjuntos de dados propriamente ditos nesta escolha, tais como:

- Quantidade de classes;
- Quantidade de atributos disponível.
- Tamanho do conjunto de treinamento, i.e., número de instâncias.

Buscou-se selecionar conjuntos de dados que variem entre si conforme as caracte-

rísticas acima, com o intuito de obter resultados que indiquem estimativas de desempenho para um conjunto variado de dados de teste, e assim permitam uma avaliação e comparação dos métodos em diferentes cenários. Com isso, os seguintes conjuntos de dados foram selecionados para avaliação da abordagem proposta:

Tabela 4.1: Conjuntos de dados analisados nos experimentos realizados

Dataset	Nº de classes	Nº de atributos	Nº de instâncias
Iris	3	4	150
Page blocks	5	10	5473
Dermatology	6	33	366
Sat-image	6	36	4435
Glass	7	9	214
Segmentation	7	16	2310
Pen Digits	10	16	7494
Opt Digits	10	64	3823
Yeast	11	9	1484
Vowel	11	12	528
Soybean	19	35	307

Todos os conjuntos de dados foram pré-processados para o formato desejado nos experimentos, e encontram-se disponíveis na seguinte URL: <https://github.com/rsboos/DistributedClassifier/tree/master/datasets>.

4.4 Recursos e código-fonte

A implementação dos classificadores foi realizada através do uso da biblioteca SCI Kit (PEDREGOSA et al., 2011). Algumas funções de escolha social foram obtidas deste repositório: <https://github.com/qpwo/pySCF>. Os demais processos e algoritmos foram desenvolvidos pelo autor, e o código fonte está disponível no seguinte repositório: <https://github.com/rsboos/DistributedClassifier>.

5 EXPERIMENTOS E RESULTADOS

Com base nos conjuntos de dados selecionados para avaliação, elaborou-se uma sequência de execuções do algoritmo definido acima, visando comparar o desempenho das diferentes técnicas analisadas. Como métricas, utilizou-se o F1 Macro, F1 Ponderado e a Acurácia. O F1 Micro não foi incluído pois neste cenário de classificação multiclasse em que cada instância pode pertencer a apenas uma classe esta métrica é equivalente à acurácia.

Para avaliar a efetividade dos métodos adotados (funções de escolha social e algoritmos de classificação) no cenário de classificação descentralizada, compara-se o desempenho dos mesmos com dois tipos de classificadores:

- **Modelos globais:** estes classificadores possuem acesso a todos os atributos do conjunto de treinamento, representando um cenário de dados centralizados. Buscam-se técnicas de agregação de modelos treinados de forma descentralizada que gerem resultados o mais próximo possível dos resultados observados para estes modelos globais. Foram treinados modelos globais utilizando-se os algoritmos OVA, ECOC, e KNN, conforme revisado na Seção 2.2. Estes modelos são identificados nos gráficos apresentandos neste capítulo pelo prefixo *Global* seguido pelo algoritmo específico adotado.
- **Modelos locais:** na classificação descentralizada, cada um dos K modelos locais de classificação, treinados individualmente pelos agentes classificadores distribuídos entre os múltiplos sítios, possui seu respectivo desempenho. Este desempenho reflete o quão bem o modelo generaliza as regras de classificação através de um subconjunto de atributos, os quais podem variar entre múltiplas repetições do 10-*fold cross-validation*. Assim, o intuito desta comparação é verificar o possível ganho de desempenho dos modelos obtidos a partir da agregação de resultados com funções de escolha social em relação aos modelos locais. Estes modelos são identificados nos gráficos apresentados a seguir pelo prefixo *Local*, seguido pelo algoritmo específico adotado. Como algoritmos utilizou-se os métodos bases da abordagem "*default*" proposta anteriormente, sendo estes AVA, OVA, Árvores de Decisão, KNN e Redes Neurais.

Como mencionado no capítulo anterior, nestes experimentos foram executadas 10 repetições do 10-*fold cross-validation* para melhor avaliar o poder de generalização dos

modelos e, principalmente, para que as diferenças nos resultados entre diferentes distribuições de atributos entre os agentes classificadores sejam amortizadas. Com o intuito de demonstrar a significância dos resultados apresentados nas próximas seções, apresentamos para cada conjunto de dados analisado a média das métricas de desempenho utilizadas, bem como a distribuição de valores do F1 Ponderado entre as 10 repetições do 10-*fold cross-validation* (isto é, 100 valores de F1 Ponderado, visto que executamos 10 vezes o treinamento de 10 modelos pela estratégia de *k-fold cross-validation*). Nestas distribuições, podemos observar não só o desempenho médio, mas também o desvio padrão entre os múltiplos modelos, o qual quando pequeno indica significância dos resultados obtidos. Foi selecionado o F1 Ponderado visto que considera em seu cálculo o efeito do desbalanceamento de classes, se houver.

5.1 Domínios do UCI

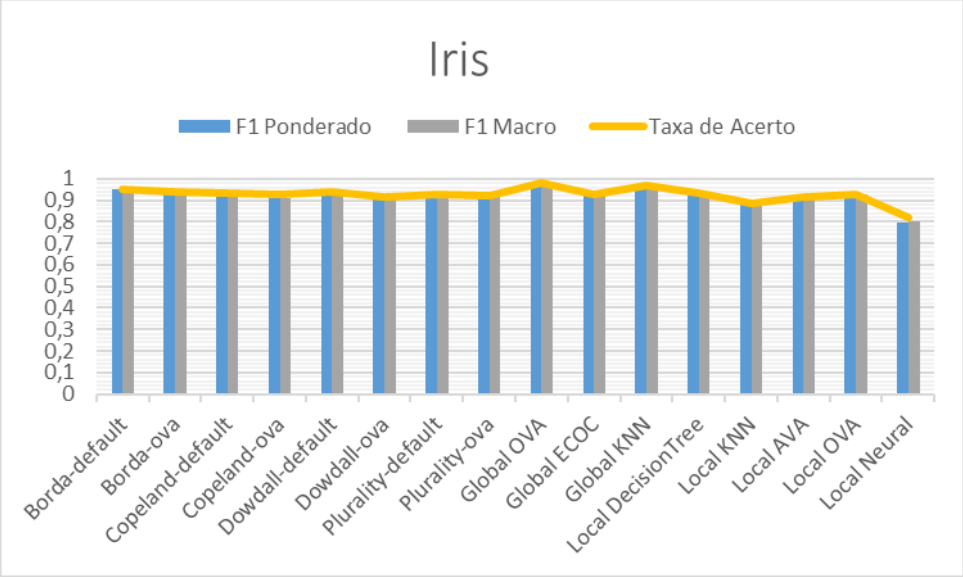
Esta seção compreende ideias específicas a respeito dos resultados de cada conjunto de dados, bem como ao final conclusões gerais sobre a tarefa analisada.

5.1.1 Iris

Iris é o conjunto de dados com menor número de classes (i.e., 3 classes) e apresenta classes perfeitamente balanceadas. A Figura 5.1 mostra o desempenho médio entre as duas abordagens propostas, "*default*" e "*OVA*", cada qual associada com uma dentre as quatro funções de escolha social adotadas (Borda Count, Copeland, Dowdall e Pluralidade), bem como para os modelos globais e locais treinados. Podemos observar que não houve diferença importante entre os modelos treinados com as duas abordagens para uma mesma função de escolha social, embora a abordagem "*default*" apresente um leve aumento no desempenho em relação à abordagem "*OVA*". Adicionalmente, nota-se que o desempenho das funções Borda Count, Copeland e Dowdall é um pouco melhor que o desempenho da pluralidade, ainda que esta diferença não possa ser considerada significativa. Observamos também que existe uma melhora no desempenho dos modelos descentralizados em relação aos modelos locais, e que este desempenho está bem próximo do que foi observado para os modelos globais. A estratégia OVA (com classificadores SVM) foi a que obteve o melhor desempenho tanto entre os modelos globais, como entre todos os

demais modelos treinados.

Figura 5.1: Desempenho para o conjunto de dados iris. Os números apresentados referem-se às médias calculadas para 10 repetições de 10-*fold cross-validation*.

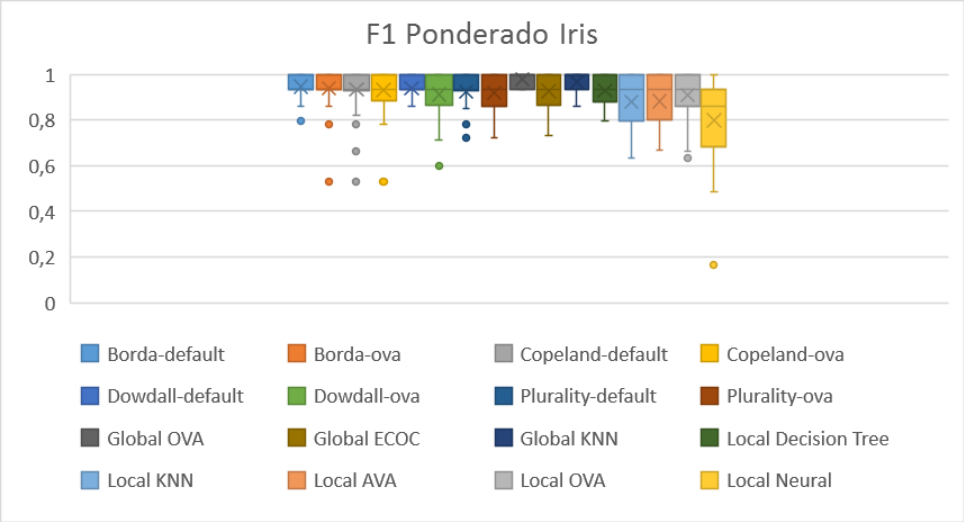


Fonte: Autor

Assim, de uma forma geral observamos que as médias das métricas de desempenho para o conjunto iris são bastante comparáveis entre os diferente modelos, com exceção dos modelos locais que possuem um desempenho um pouco inferior, especialmente o modelo treinado com redes neurais. Este comportamento foi bastante particular para este conjunto de dados, como poderá ser observado nas discussões a seguir. Isso ocorre, provavelmente, devido ao fato de que o conjunto de dados possui poucos atributos, então o número total de modelos é apenas 2 na classificação distribuída a fim de atender à restrição discutida na Seção 4.2.1, e os classificadores locais consequentemente conseguem acessar uma boa porcentagem dos atributos.

Podemos observar a distribuição dos valores de F1 Ponderado na Figura 5.2, os quais indicam que no geral os modelos treinados de forma descentralizada com a abordagem "*default*"possuem menor dispersão de valores em relação à abordagem "*OVA*", indicando uma possível maior consistência no desempenho. Estes boxplots também sugerem uma variação maior nos scores F1 ponderados calculados ao longo das 10 repetições de 10-*fold cross-validation*, suportando a hipótese de que a classificação pode ser aprimorada em ambientes com particionamento vertical dos dados através de métodos de agregação, obtendo resultados bem próximos aos cenários centralizados.

Figura 5.2: Boxplot para as execuções com o conjunto de dados iris. Os números apresentados referem-se a probabilidade média.



Fonte: Autor

5.1.2 Page Blocks

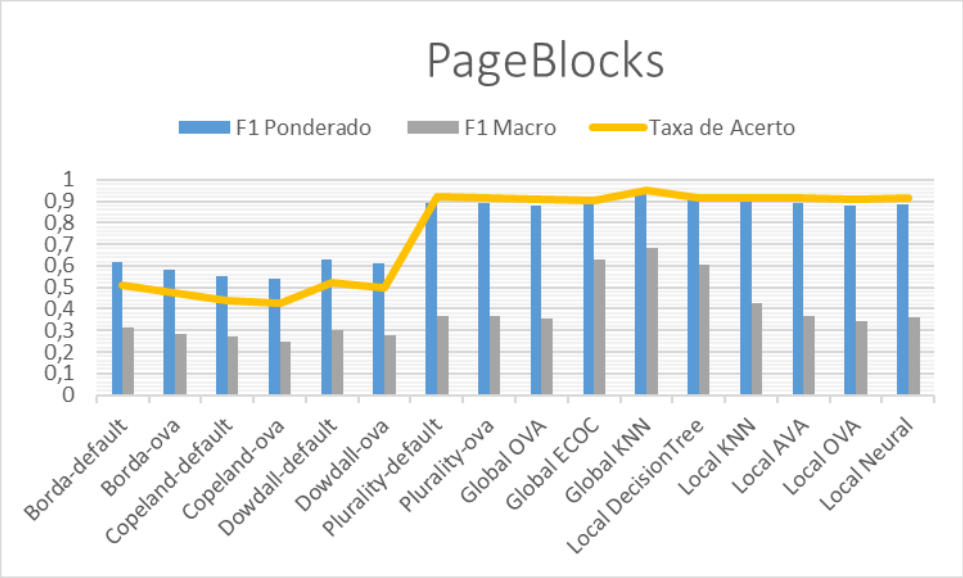
O conjunto de dados Page Blocks apresenta um desbalanceamento de classes bem acentuado. A classse majoritária possui 90% das instâncias, enquanto que a minoritária possui apenas 0,5%. Talvez como consequência deste desbalanceamento, podemos observar na Figura 5.3 que com exceção do método da pluralidade, as funções de escolha social não apresentam bons resultados para estes experimentos. Já os modelos locais e globais apresentam desempenho comparável quando avaliado pela métrica F1 Ponderado.

Adicionalmente, é possível observar uma grande diferença entre os scores F1 Ponderados, Taxa de Acerto e F1 Macro para todos os modelos reportados. Neste caso especificamente, nota-se uma substancial redução dos valores de F1 Macro em relação à Taxa de Acerto (a qual é equivalente à F1 Micro), sugerindo que os modelos tendem a classificar corretamente as instâncias das classes maiores, mas erram bastante a classificação de instâncias das classes menores. Isto reforça a necessidade de se avaliar diferentes métricas de desempenho em contextos como este.

O pior desempenho dos modelos treinados de forma descentralizada, agregados por funções de escolha social, também é evidente no boxplot da Figura 5.4. A pluralidade alcançou bons resultados, comparáveis com os modelos globais tanto em questão de medianas como de desvio padrão, o que não foi observado para as demais funções, sugerindo que as funções Borda, Copeland e Dowdall são mais sensíveis ao efeito do desbalanceamento de classes. Comparando-se as abordagens "*default*" e "*OVA*"para os

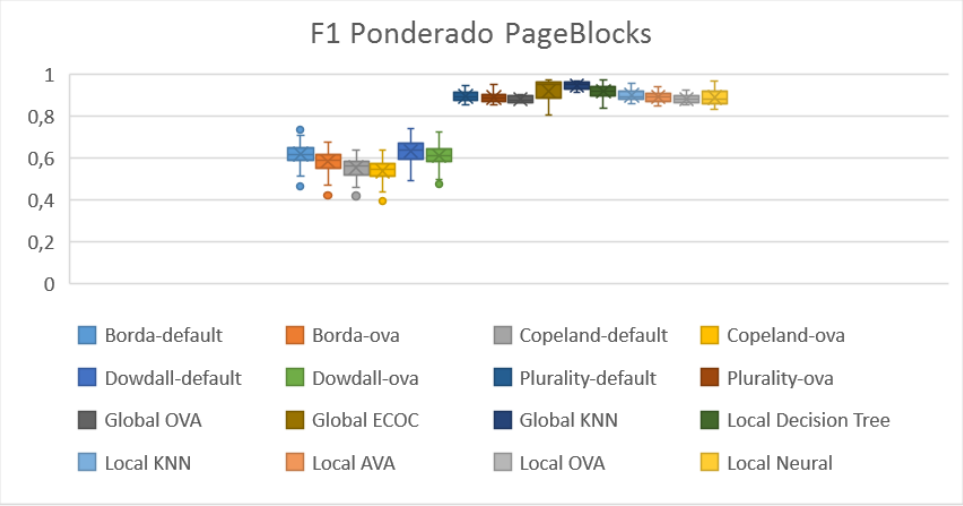
cenários descentralizados, percebe-se que a primeira obteve um desempenho um pouco melhor que a segunda, principalmente no caso das funções Borda, Copeland e Dowdall. Por fim, o melhor resultado absoluto foi obtido com o modelo "*Global-KNN*", o qual alcançou as médias mais altas, acompanhadas de um pequeno desvio padrão.

Figura 5.3: Desempenho para o conjunto de dados page blocks. Os números apresentados referem-se às médias calculadas para 10 repetições de 10-*fold cross-validation*.



Fonte: Autor

Figura 5.4: Boxplot para as execuções com o conjunto de dados page blocks. Os números apresentados referem-se a probabilidade média.



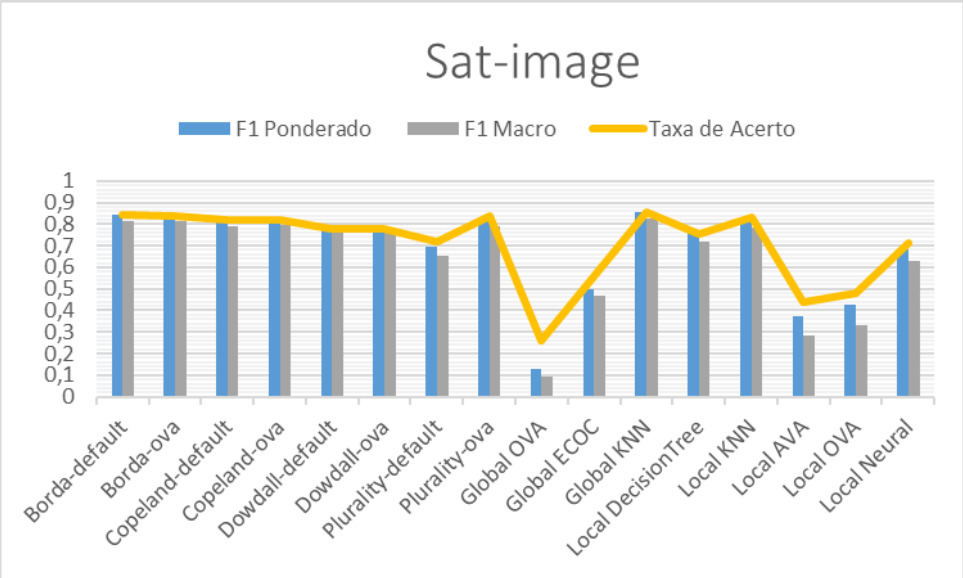
Fonte: Autor

5.1.3 Sat Image

Neste conjunto de dados, podemos observar que todas as estratégias de agregação fizeram com que o desempenho melhorasse de forma significativa com relação às soluções locais obtidas com os algoritmos AVA, OVA e Redes Neurais (Figura 5.5). Já os modelos locais treinados com árvores de decisão e KNN atingiram bons resultados, comparáveis aos resultados dos modelos construídos com as abordagens "*default*" e "*OVA*". Ainda, percebemos que neste cenário a pluralidade teve um desempenho inferior às demais funções de agregação, principalmente quando combinada à abordagem "*default*".

Porém, vale notar que para este conjunto de dados as soluções obtidas com modelos centralizados utilizando os algoritmos ECOC e OVA, apresentaram-se piores até mesmo que resultados relacionados aos modelos locais. Este é um resultado inesperado, e pode indicar que os atributos disponíveis para treinamento não foram bem selecionados por quem construiu o conjunto de dados, apresentando pouco poder preditivo quando tomados em conjunto. Neste caso, o processo de seleção de atributos adotado usualmente como pré-processamento de dados em tarefas de classificação poderia excluir atributos irrelevantes a fim de melhorar o desempenho do sistema, possivelmente justificando o fato de alguns modelos locais alcançarem desempenho satisfatório.

Figura 5.5: Desempenho para o conjunto de dados sat-image. Os números apresentados referem-se às médias calculadas para 10 repetições de 10-*fold cross-validation*.

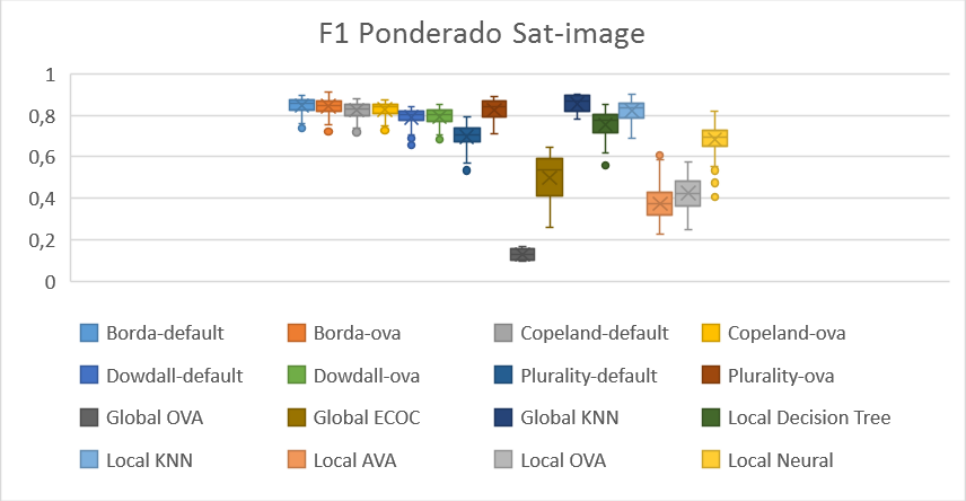


Fonte: Autor

O pior desempenho dos algoritmos centralizados OVA e ECOC também é observado no boxplot da Figura 5.6. Além disso, verifica-se que os modelos treinados de

forma descentralizada possuem menor dispersão de valores em relação aos locais, assim como com relação ao ECOC global; sugerindo uma possível melhora na consistência do desempenho com a utilização de classificação distribuída.

Figura 5.6: Boxplot para as execuções com o conjunto de dados sat. Os números apresentados referem-se a probabilidade média.



Fonte: Autor

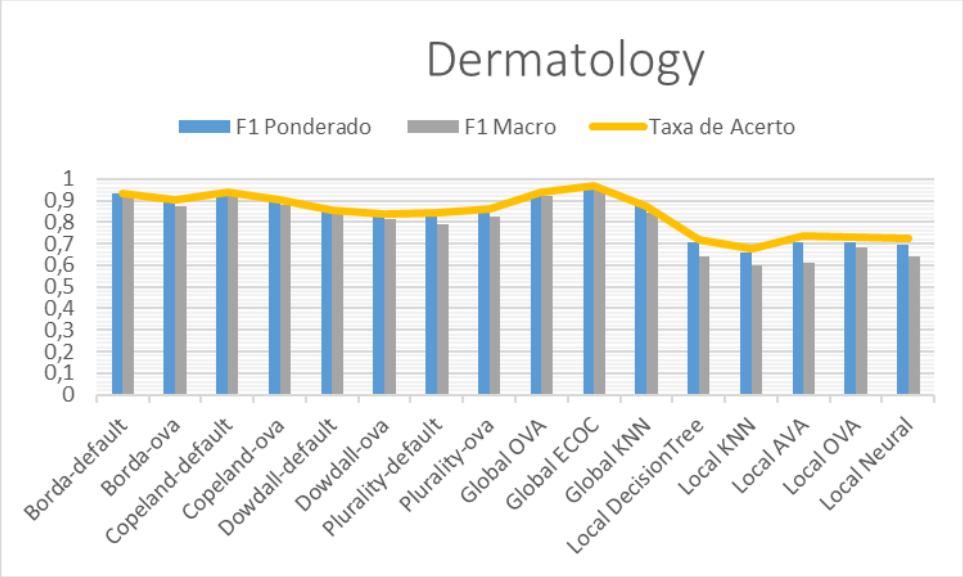
5.1.4 Dermatology

Para o conjunto de dadoa Dermatology, notamos que os modelos treinados com classificadores distribuídos alcançaram bons resultados, apresentando melhor desempenho que todos os modelos locais (Figura 5.7). Além disso, observa-se que os modelos agregados por funções de escolha social se aproximaram bastante dos modelos globais em termos das diferentes métricas de avaliação adotadas. Ainda, é possível perceber que Borda e Copeland têm o melhor desempenho entre os métodos de agregação, ultrapassando o desempenho obtido com agregação por pluralidade.

No cenário descentralizado, os melhores resultados foram novamente obtidos com os modelos treinados com a abordagem "*default*", tanto quando consideramos os valores médios de desempenho, como a distribuição do F1 Ponderado (Figura 5.8). Adicionalmente, é possível perceber que existe uma distribuição um pouco mais dispersa para a pluralidade em relação aos métodos Borda, Copeland e Dowdall, e que a variação no desempenho é especialmente grande para os modelos locais, treinados com subconjuntos de atributos derivados do particionamento vertical dos dados. Assim, em cenários com descentralização de dados, as funções de escolha social apresentam-se novamente como

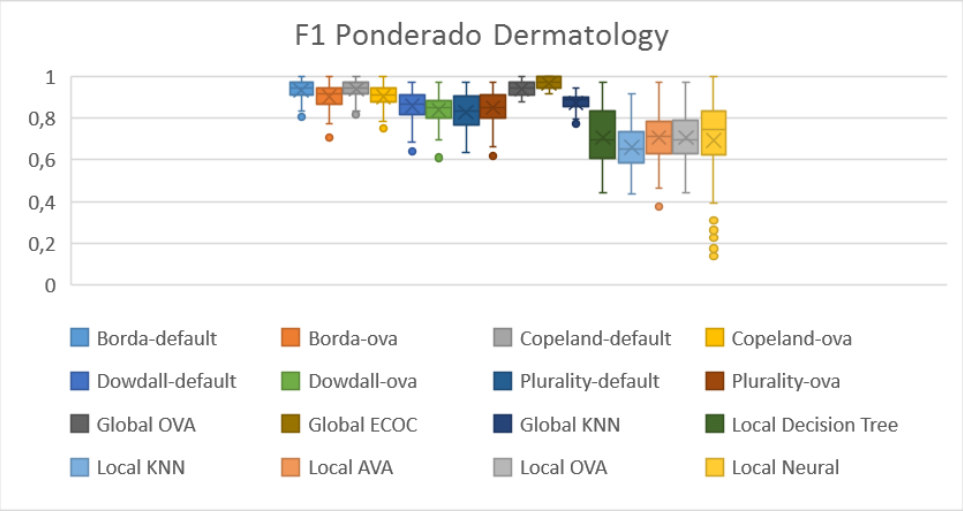
uma boa alternativa para lidar com este particionamento dos atributos.

Figura 5.7: Desempenho para o conjunto de dados dermatology. Os números apresentados referem-se às médias calculadas para 10 repetições de 10-*fold cross-validation*.



Fonte: Autor

Figura 5.8: Boxplot para as execuções com o conjunto de dados dermatology. Os números apresentados referem-se a probabilidade média.



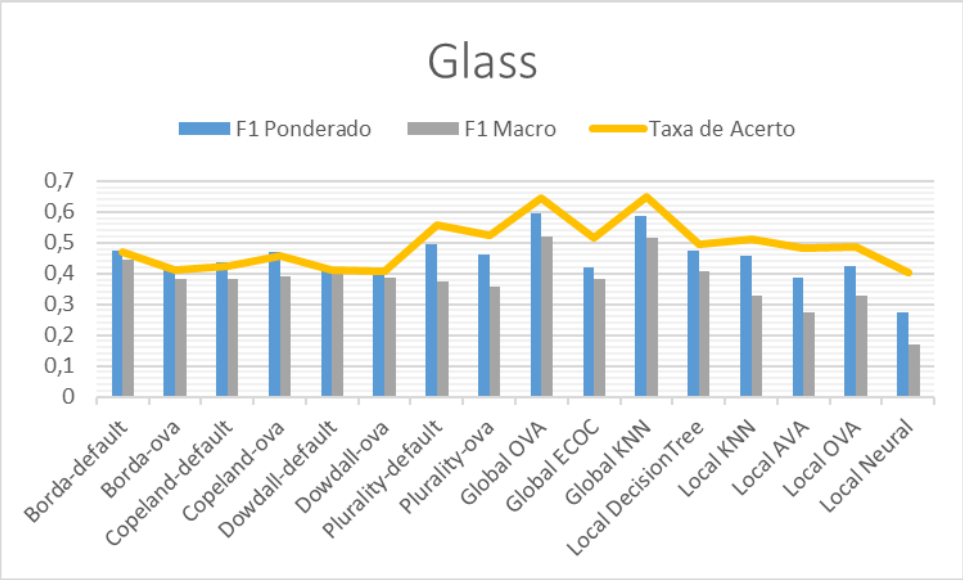
Fonte: Autor

5.1.5 Glass

Assim como no conjunto de dados Page Blocks, os dados Glass apresentam alto desbalanceamento de classes, mas de forma um pouco menos acentuada, contendo 35%

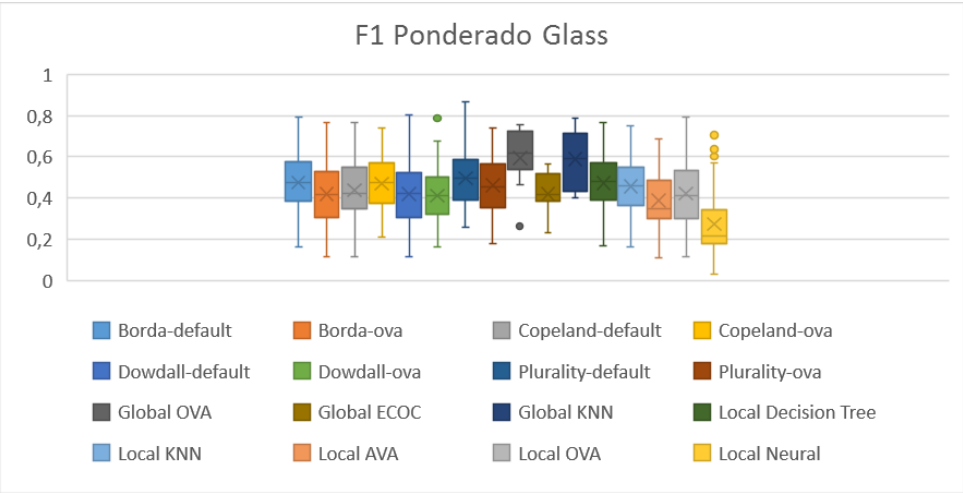
das instâncias na classe majoritária e 4% na minoritária. Os desempenhos médios para F1 Ponderado, F1 Macro e Taxa de Acerto são sumarizados na Figura 5.9. Novamente, podemos observar uma piora no desempenho dos modelos descentralizados em relação aos modelos globais, repetindo-se o fenômeno de maior impacto desta característica sobre o desempenho das funções Borda, Copeland e Dowdall, tal que a pluralidade associada à abordagem "*default*"apresentou-se como o melhor método de agregação neste caso.

Figura 5.9: Desempenho para o conjunto de dados glass. Os números apresentados referem-se às médias calculadas para 10 repetições de 10-*fold cross-validation*.



Fonte: Autor

Figura 5.10: Boxplot para as execuções com o conjunto de dados glass. Os números apresentados referem-se a probabilidade média.



Fonte: Autor

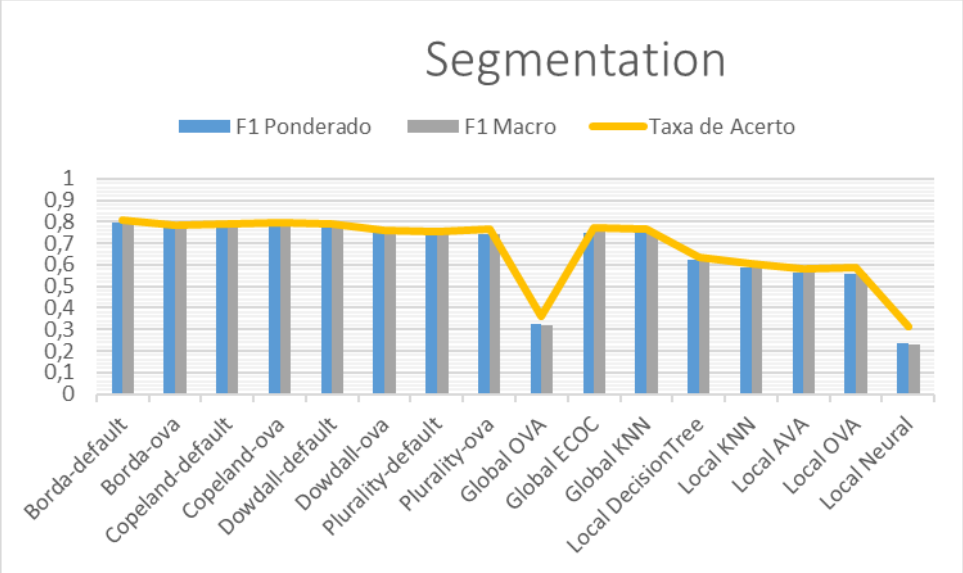
Dentre todos os modelos treinados, os modelos globais com OVA e KNN foram

os que apresentaram melhores desempenhos médios. No entanto, os boxplots da Figura 5.10 mostram que analisando as distribuições de valores F1 Ponderado para ambos os modelos, o algoritmo OVA obteve menor dispersão do que o algoritmo KNN. Todos os demais modelos, tanto descentralizados como locais, apresentaram variações de desempenho maiores e comparáveis entre si.

5.1.6 Segmentation

Assim como no conjunto Sat Image, os classificadores distribuídos tiveram desempenho melhor do que as soluções centralizadas, porém a discrepância não é tão grande quanto no conjunto anterior. Apesar desta característica, visualiza-se que os modelos globais geram resultados melhores que os modelos locais (com exceção do OVA). Por isso, neste caso não é possível concluir que os atributos não estão bem selecionados. Partindo destas informações, é possível notar que a agregação de diversos tipos de classificadores é bastante efetivo neste conjunto de dados, visto seu ótimo desempenho mesmo utilizando apenas subconjuntos dos atributos de cada instância.

Figura 5.11: Desempenho para o conjunto de dados segmentation. Os números apresentados referem-se às médias calculadas para 10 repetições de 10-*fold cross-validation*.

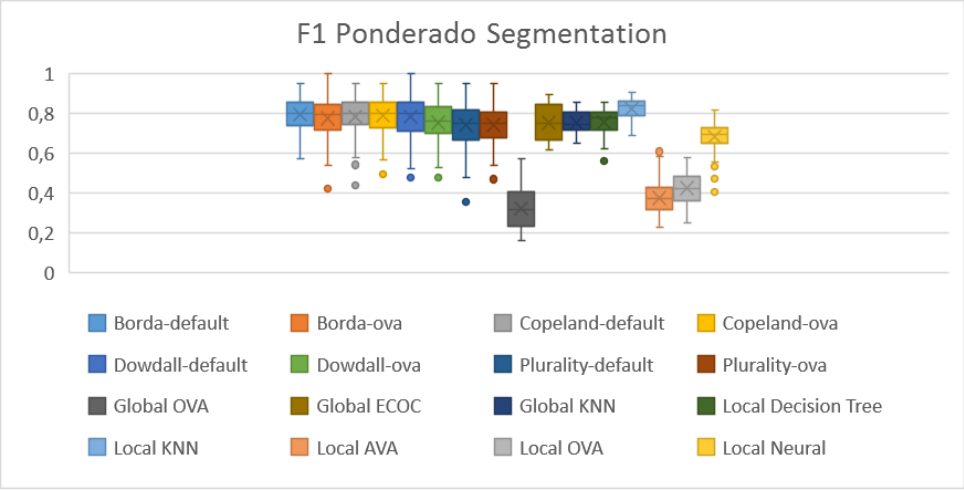


Fonte: Autor

Outras conclusões já mencionadas em outros conjuntos de dados também são válidas, como a melhora das funções Borda, Copeland e Dowdall com relação a pluralidade. Além disso, verificam-se desempenhos muito similares entre as diferentes funções de es-

colha social, inclusive em termos de dispersão dos valores (Figura 5.12). Por fim, ainda vale mencionar que a variação de desempenho observada entre os múltiplos modelos globais e locais treinados não é consistente, com o modelo global ECOC apresentando não só um desempenho médio inferior em relação aos demais modelos globais, como também distribuição mais dispersa, e os modelos locais baseados em KNN e árvores de decisão possuindo menos variação que os demais nessa categoria.

Figura 5.12: Boxplot para as execuções com o conjunto de dados segmentation. Os números apresentados referem-se a probabilidade média.



Fonte: Autor

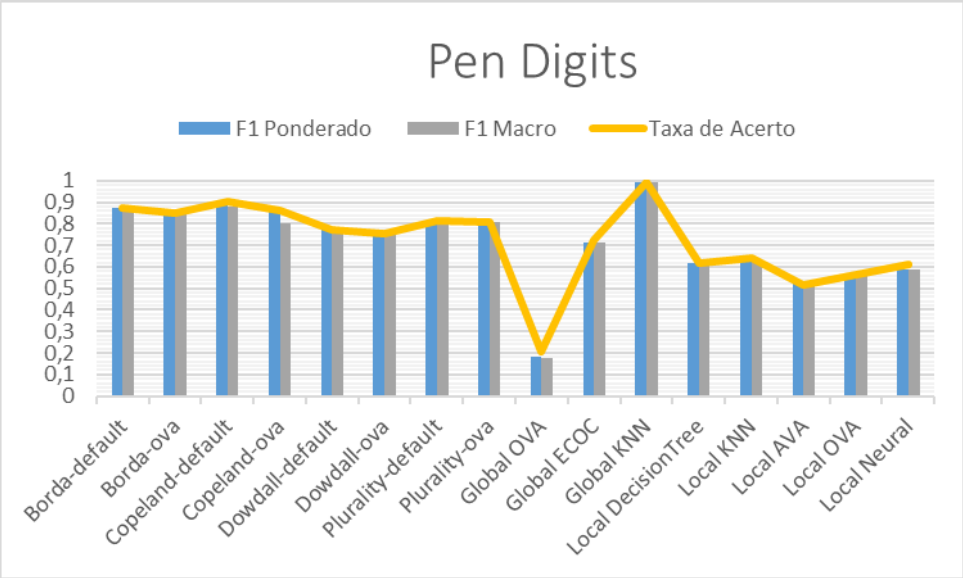
5.1.7 Pen Digits

Neste conjunto de dados, foi observado que o classificador central OVA tem desempenho muito abaixo do que até mesmo o classificador local OVA (5.13). Esta situação também ocorreu em outros testes já reportados, e o problema provavelmente segue a linha do que foi discutido para o conjunto de dados Segmentation. O modelo global KNN foi o que obteve melhor desempenho entre todos os modelos treinados nas 3 categorias (agregados, globais e locais)

Em relação aos métodos de agregação, observamos que no geral estes de mostraram como uma boa estratégia, fornecendo soluções relativamente próximas a ideal, e com alto índice de ganho de desempenho com relação aos modelos locais. Com exceção do modelo global KNN, que como discutivo teve excelente desempenho para este conjunto de dados, os modelos treinados de forma descentralizada também apresentaria melhoria mem relação aos modelos globais. Como diferença no desempenho das funções de es-

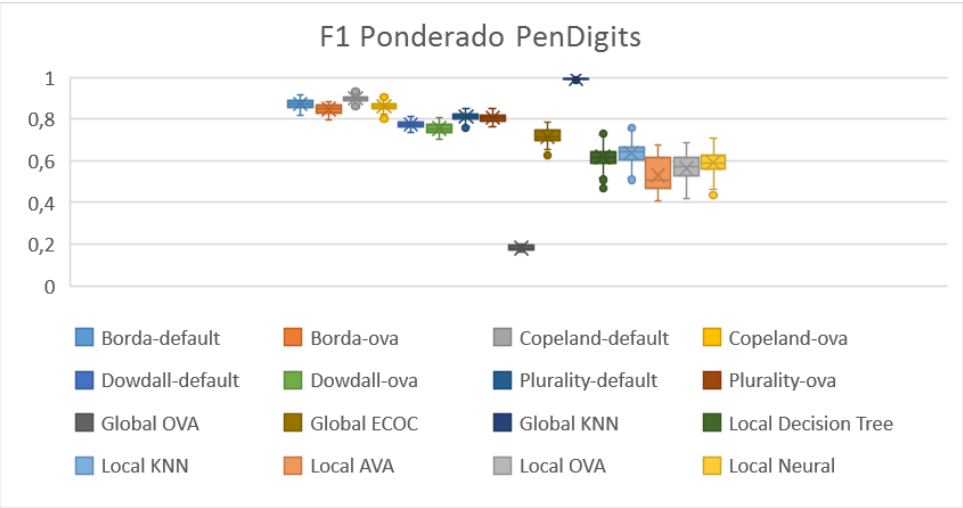
colha social, verifica-se que a função de Copeland gera os melhores resultados para este cenário quando associada com a abordagem "*default*", porém sem diferenças significativas com relação a Borda. Estas duas, por sua vez, apresentam melhor desempenho que aos métodos baseados na pluralidade e Dowdall.

Figura 5.13: Desempenho para o conjunto de dados pen digits. Os números apresentados referem-se às médias calculadas para 10 repetições de 10-*fold cross-validation*.



Fonte: Autor

Figura 5.14: Boxplot para as execuções com o conjunto de dados pen digits. Os números apresentados referem-se a probabilidade média.



Fonte: Autor

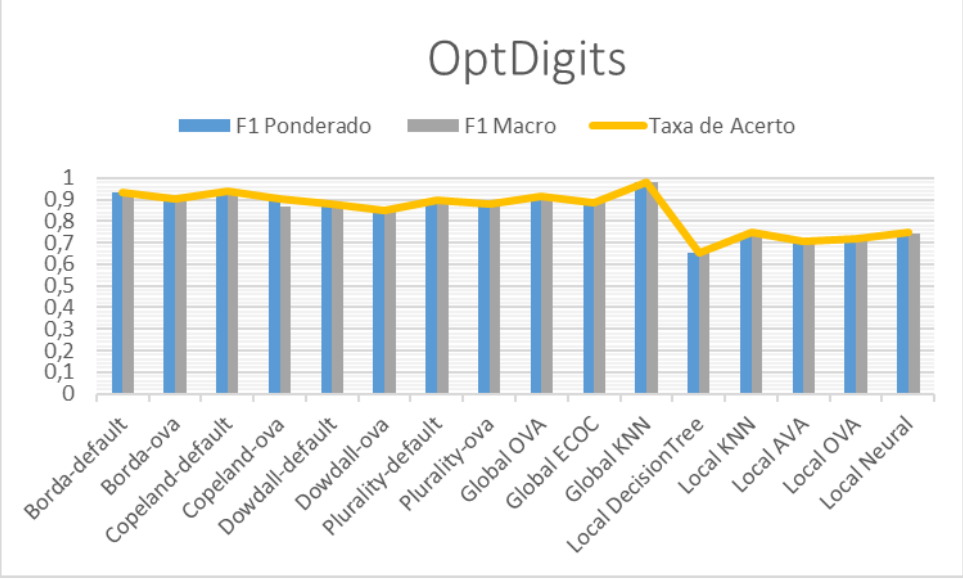
Como principal indicativo da Figura 5.14, constatamos a baixa dispersão nas distribuições de F1 Ponderado entre as execuções, especalmente para os modelos globais e

agregados via funções de escolha social. No entanto, mesmo nos modelos locais a dispersão foi menor do que observada para outros testes. Isto indica alta consistência no desempenho, destacando de forma precisa os valores das métricas vistos na Figura 5.13.

5.1.8 Opt Digits

Neste conjunto de dados, as funções de escolha social fizeram com que o desempenho dos modelos agregados aumentasse bastante com relação aos modelos locais, conforme podemos constatar na Figura 5.15. As mesmas apresentaram desempenhos bastante similares, sendo que apenas a função de Dowdall ficou com performancce um pouco abaixo das demais, no entanto com uma diferneça que não ser significativa. Além disso, os classificadores distribuídos seguindo a abordagem "*default*" e agregados com as funções Borda e Copeland apresentaram resultados melhores que os métodos globais OVA e ECOC, indicando que neste caso a agregação não só foi eficiente em lidar com a descentralização de dados, mas possibilitou melhora na solução em relação ao cenário centralizado.

Figura 5.15: Desempenho para o conjunto de dados opt digits.Os números apresentados referem-se às médias calculadas para 10 repetições de 10-*fold cross-validation*.

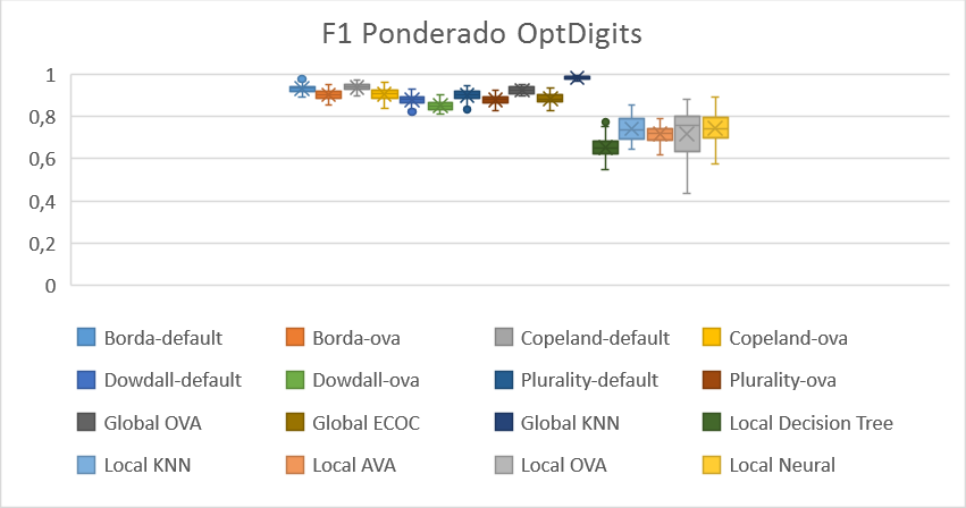


Fonte: Autor

Ainda, visualiza-se que a variação de resultados entre as execuções é pequena para todos os modelos globais e agregados via funções de escolha social. Assim, como discutido nos outros conjuntos de dados, também configura-se como uma forma de indicar alta consistência no desempenho dos algoritmos e abordagens adotadas, algo que não foi

observado com a mesma significância em relação aos modelos locais. Isto é, a agregação de modelos descentralizados seguindo as abordagens propostas é capaz de mitigar os efeitos de variações no poder preditivo dos modelos individuais, causadas pela seleção aleatória de subconjuntos de atributos que de acordo com sua composição pode gerar impacto positivo ou negativo no desempenho do modelo. Ainda, esse efeito da agregação nos leva a uma possível evidência de que para este caso, os atributos no conjunto de treinamento são independentes entre si e possuem boa complementariedade para a tarefa de classificação, levando a resultados bem semelhantes e consistentes nos casos em que estão centralizados ou agregados por funções de escolha social.

Figura 5.16: Boxplot para as execuções com o conjunto de dados opt digits. Os números apresentados referem-se a probabilidade média.



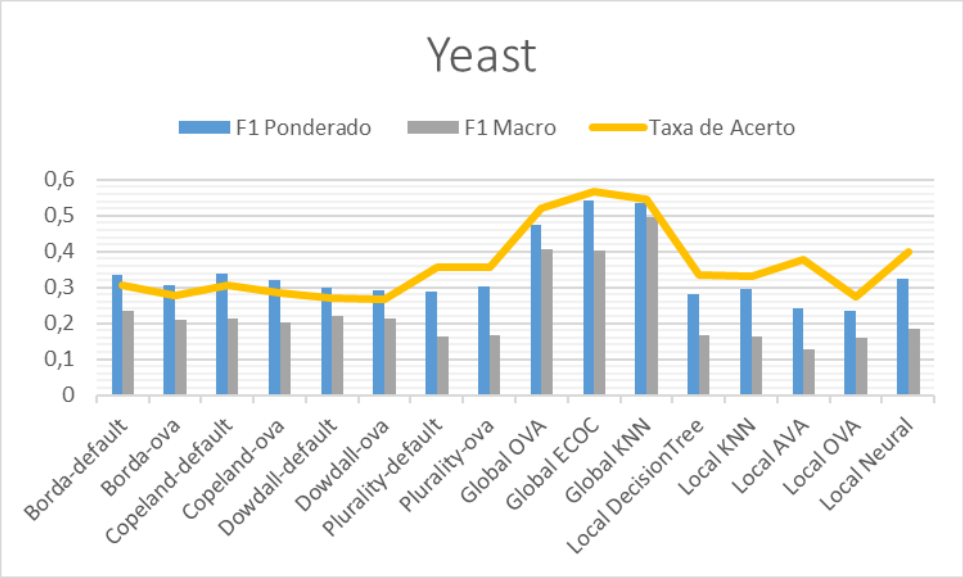
Fonte: Autor

5.1.9 Yeast

Assim como os conjuntos de dados Glass e Page Blocks, estes dados possuem alto índice de desbalanceamento de classes. A classe majoritária representa 32% das instâncias, ao passo que a menos recorrente possui apenas 1,3% das instância. Por isso, as conclusões são similares as apontadas nos outros dois conjuntos citados. No entanto, vale ressaltar que os modelos globais alcançaram desempenho significativamente maior que os demais tipos de modelo, e que o KNN no cenário centralizado foi o algoritmo que melhor lidou com o desbalanceamento de classes mencionado (Figura 5.17).

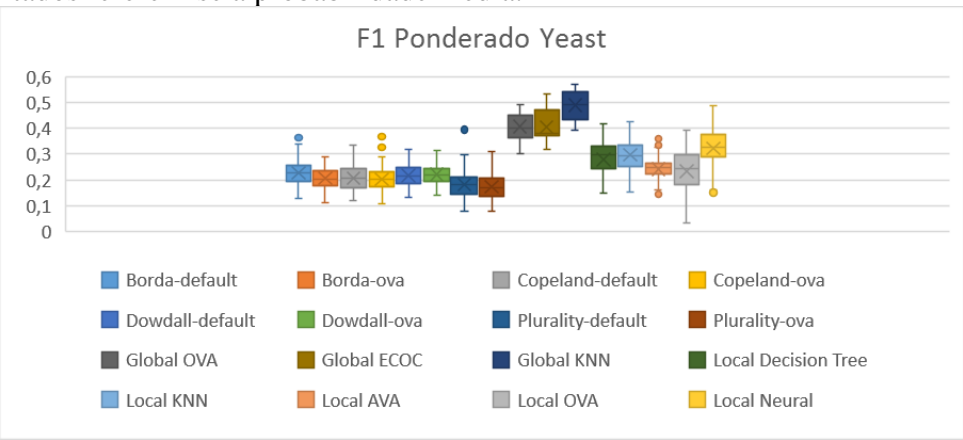
Visualiza-se claramente na Figura 5.17 que o F1 Macro gera resultados piores que as demais métricas, principalmente em relação à taxa de acerto. Isso ocorre quando há um

Figura 5.17: Desempenho para o conjunto de dados yeast. Os números apresentados referem-se às médias calculadas para 10 repetições de 10-*fold cross-validation*.



Fonte: Autor

Figura 5.18: Boxplot para as execuções com o conjunto de dados yeast. Os números apresentados referem-se a probabilidade média.



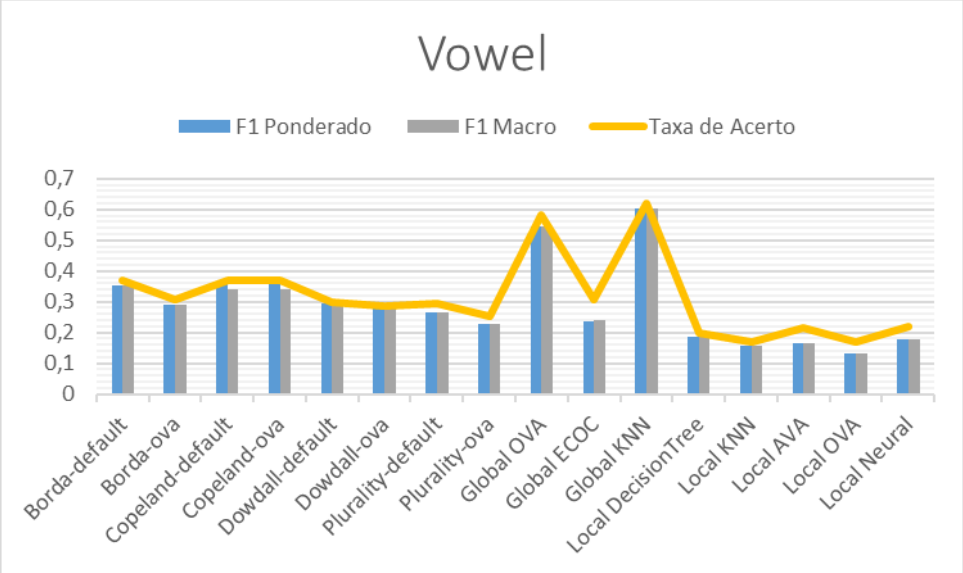
Fonte: Autor

alto índice de erro de predição neste cenário desbalanceado, e indica que as instâncias das classes menos representativas não estão sendo previstas de forma correta. Neste sentido, percebe-se que a diferença entre as métricas F1 Ponderado e taxa de acerto é menor para os modelos agregados com as funções Boda, Copeland e Dowdall em relação à agregação com pluralidade, bem como em comparação com todos os modelos locais. Isto é, ainda que a abordagem proposta, baseada nos métodos de agregação mais sofisticados, não tenha sido efetiva em introduzir grandes melhoras no desempenho absoluto, foi capaz de diminuir o impacto do desbalanceamento de classes.Podemos observar na Figura 5.18 que esta hipótese é suportada, também, pela menor dispersão para a distribuição relacionada às abordagens "*default*" e "*OVA*", do que em relação aos modelos locais.

5.1.10 Vowel

Para estes testes, o gráfico representado na Figura 5.19 demonstra que houve ganho de desempenho com a agregação dos modelos locais, porém a solução final ainda ficou longe da obtida com modelos centrais. Novamente, as funções Borda e Copeland tiveram melhor desempenho que a pluralidade. Além disso, nota-se que a abordagem "*default*" gera melhores soluções que a abordagem "*OVA*", conforme observado em testes anteriores.

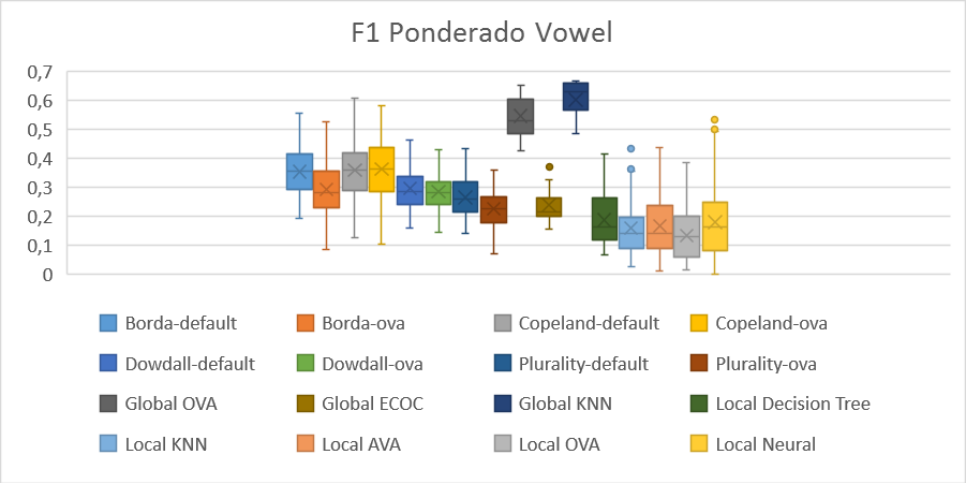
Figura 5.19: Desempenho para o conjunto de dados vowel. Os números apresentados referem-se às médias calculadas para 10 repetições de 10-*fold cross-validation*.



Fonte: Autor

Como particularidade dos resultados para este conjunto de dados, vale citar o baixo desempenho do modelo global ECOC. Este tipo de classificador se provou bastante ineficiente para este conjunto de dados, alcançando resultados comparáveis com os modelos locais treinados a partir de subconjuntos de atributos. Novamente, observamos grandes variações de desempenho para os modelos agregados e locais (Figura 5.20. No entanto, é interessante ressalter que para este cenário, as funções Dowdall e Pluralidade apresentaram menor dispersão do que as funções Borda e Copeland para os cenários de classificação descentralizada.

Figura 5.20: Boxplot para as execuções com o conjunto de dados vowel. Os números apresentados referem-se a probabilidade média.



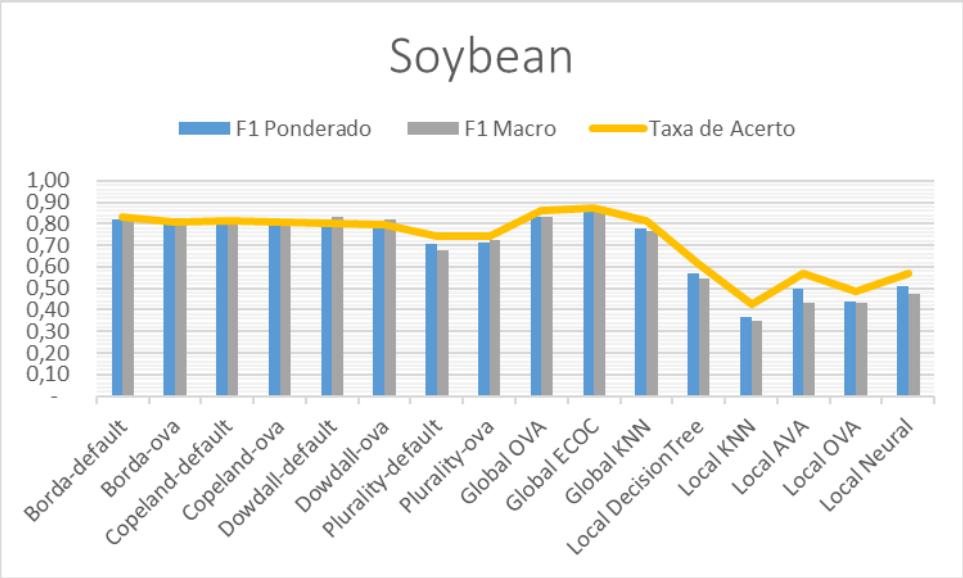
Fonte: Autor

5.1.11 Soybean

Como podemos analisar na Figura 5.21, as agregações via funções de escolha social demonstram um bom desempenho, estando muito próximas dos resultados obitodos com os classificadores treinados em cenários centralizados (isto é, os modelos globais). Além disso, percebe-se que o ganho de informação partindo das soluções locais é alto, logo conclui-se que o método cumpre bem seu papel de mitigar os efeitos da descentralização de dados para este conjunto de dados.

Adicionalmente, as funções de escolha social mais sofisticadas (Borda, Copeland e Dowdall) demonstram um desempenho médio superior à pluralidade, indicando uma possível melhora na agregação com a utilização das mesmas. Quando analisamos a dispersão dos modelos (Figura 5.22), podemos observar que dentre os modelos agregados, a

Figura 5.21: Desempenho para o conjunto de dados soybean. Os números apresentados referem-se às médias calculadas para 10 repetições de 10-*fold cross-validation*.



Fonte: Autor

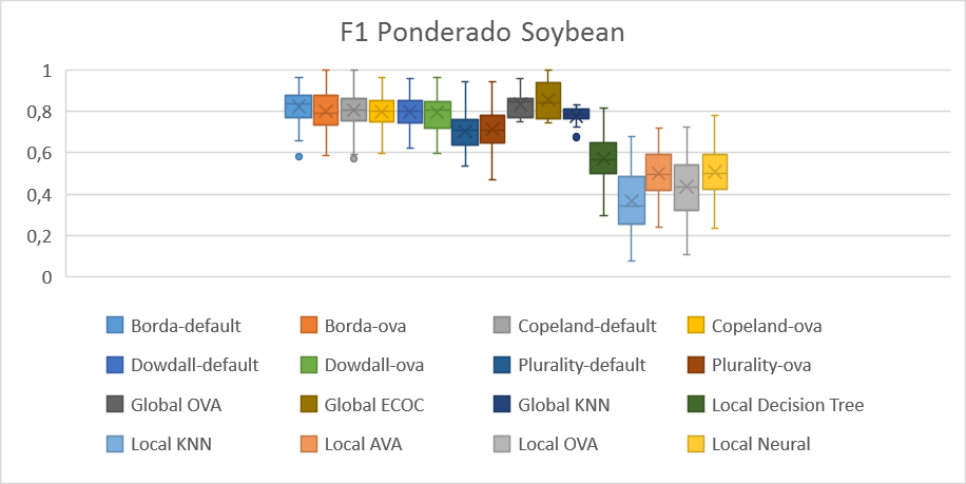
pluralidade combinada à abordagem "*OVA*"possui a maior variação de desempenho. Por outro lado, o modelo global KNN é o mais consistente entre todos os modelos comparados. Por fim, pode-se observar que tanto os modelos treinados com as abordagens "*default*" e "*OVA*" no cenário descentralizado como os modelos globais apresentam melhor estabilidade no desempenho do que os cinco modelos locais comparados. Este resultado é esperado, visto que diferentes seleções de atributos podem impactar de forma significativa no resultado final, e em apenas um classificador essa diferença tende a ser notada de forma mais marcante.

5.2 Discussão

Analisando o conjunto dos resultados apresentados na seção anterior, podemos generalizar algumas informações visualizadas na maioria dos conjuntos de dados testados, conforme determinadas características dos mesmos:

- Os modelos locais têm desempenho muito inferior aos modelos globais, como é esperado que aconteça. Isto ocorre em função da limitada quantidade de atributos acessados durante o treinamento, sendo que a maior quantidade de informações disponíveis geralmente acarretou em melhores resultados;
- Salvo algumas situações específicas, que serão comentadas a seguir, a agregação de

Figura 5.22: Boxplot para as execuções com o conjunto de dados soybean. Os números apresentados referem-se a probabilidade média.



Fonte: Autor

modelos via funções de escolha social apresentou-se como uma estratégia interessante para o cenário descentralizado em problemas de classificação multiclasse, corroborando resultados de trabalhos anteriores para o cenário centralizado (RECAMONDE-MENDOZA; BAZZAN, 2016);

- A abordagem "*default*" é em geral melhor do que a "*OVA*": nos 12 conjuntos de dados analisados, apenas em 2 casos específicos com funções Copeland a abordagem "*OVA*"gerou resultados melhores. No entanto, vale lembrar que a diferença absoluta entre os desempenhos médios geralmente não foi muito alta;
- Dowdall não se apresenta como uma alternativa interessante à agregação por Borda: nenhum exemplo indicou melhoras expressivas da função Dowdall com relação à função Borda;
- Funções de escolha social mais sofisticadas (como Borda) são melhores do que pluralidade quando o conjunto de dados não apresenta classes altamente desbalanceadas: este assunto é tratado na próxima seção.

5.3 Análise do impacto do desbalanceamento de classes

A fim de quantificar e comparar o desbalanceamento entre classes para os diferentes conjuntos de dados testados, utilizou-se uma métrica para definir o quão balanceados os conjuntos de dados estão com relação a presença das classes, uma vez que foram notadas características particulares dos conjuntos desbalanceados. O indicador aqui calculado

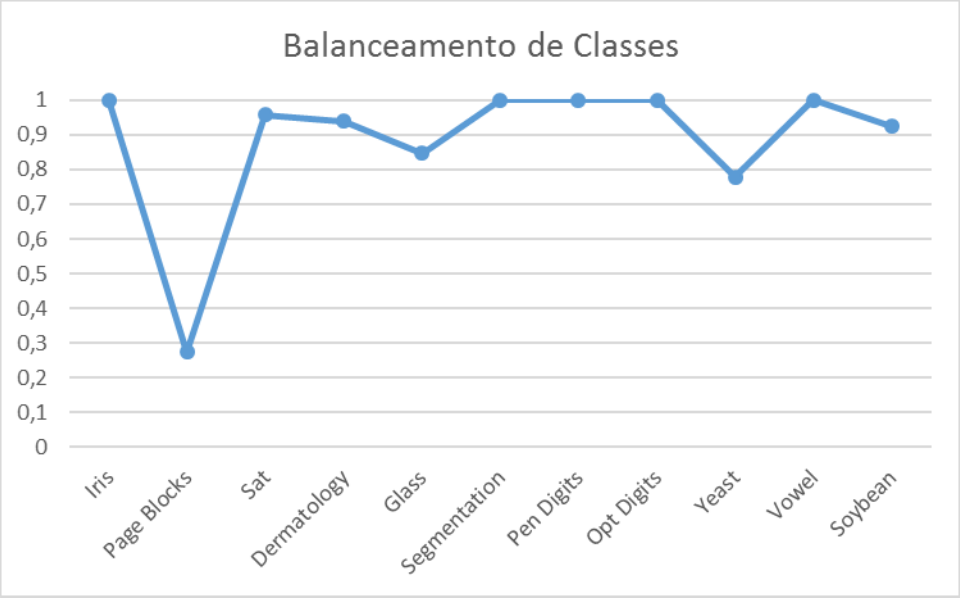
é baseado na entropia de Shannon, a qual fornece uma medida de equilíbrio de informação, e através da razão utilizada obtem-se valores variando entre 0 a 1, onde 1 representa um conjunto de dados altamente balanceado, e 0 um conjunto de dados completamente desbalanceado.

Assumindo um conjunto de dados com m instâncias, tendo N classes distintas, $C_{\mathrm{i}} \in \{C_1, C_2, ..., C_N)\}$, de tamanhos $|C_{\mathrm{i}}|$ (número de instâncias), a medida de balanceamento é dada por:

$$\text{Balanceamento} = \frac{H}{\log N} = \frac{-\sum_{i=1}^{N} \frac{|C_i|}{m} \log \frac{|C_i|}{m}.}{\log N} \tag{5.1}$$

Utilizando esta fórmula, os seguintes valores são obtidos para os conjuntos de dados analisados, conforme a Figura 5.23.

Figura 5.23: Índice de balanceamento de classes para conjuntos de dados do UCI



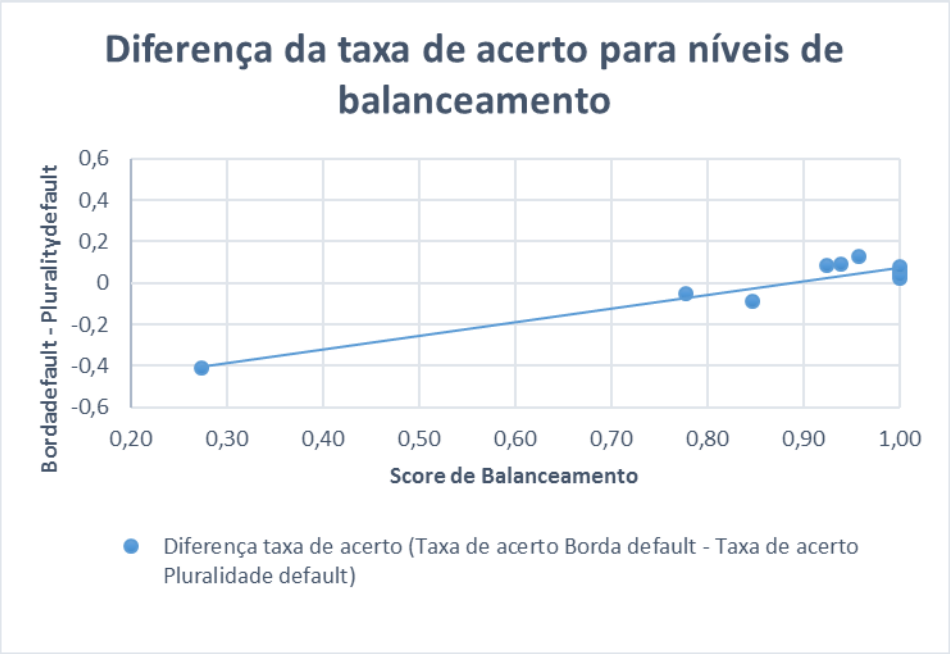
Fonte: Autor

Como pode ser visto, os três conjuntos de dados menos balanceados, são o Page Blocks, Yeast e Glass. Interessante observar que foram exatamente esses três conjuntos de dados os únicos para os quais obtivemos melhor resultado utilizando a agregação por pluralidade do que pela função Borda.

Com base nesta constatação, e visando analisar o ganho de desempenho destas duas funções de escolha social conforme o nível de balanceamento de classe, construiu-se um gráfico correlacionando o índice de balanceamento com os valores de desempenho médio obtido para os modelos treinados de forma descentralizada seguindo a abordagem

"*default*", agregados com os métodos Borda e pluralidade. Especificamente, a medida de sempenho reportada no eixo y das figuras a seguir é obtida a partir do cálculo da diferença no desempenho entre o modelo "*Borda-default*" e o modelo "*Pluralidade-default*".

Figura 5.24: Relação do índice de balanceamento de classes com a diferença entre as execuções "*Borda-default*" e "*Pluralidade-default*", para taxa de acerto.

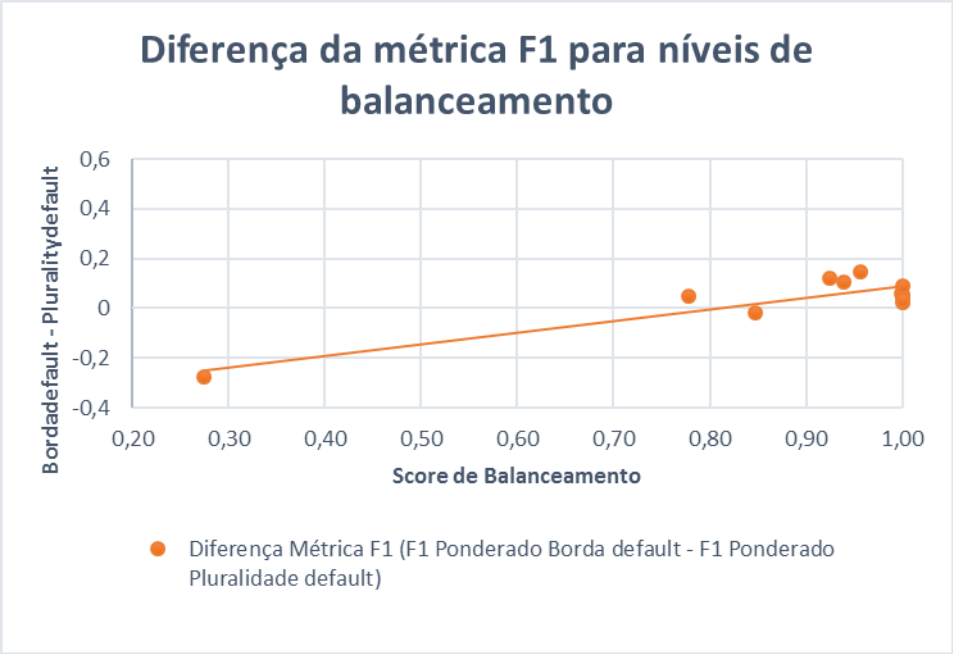


Fonte: Autor

No gráfico 5.24, representando no eixo y os valores referentes à diferença nas taxas de acerto entre os dois modelos citados, nota-se claramente uma tendência de melhor desempenho da pluralidade com relação à função Borda conforme o conjunto de dados é menos balanceado, isto é, tem-se valores negativos para a diferença calculada entre ambos os modelos quando o score de balanceamento está mais próximo de 0. Assim, de forma geral, observa-se uma correlação positiva entre o score de balanceamento dos conjuntos de dados e o ganho de desempenho do método Borda em relação ao método pluralidade ($r = 0.946$, $p < 0.001$). A Figura 5.25 mostra os resultados de uma análise similar, porém realizada com base nas diferenças entre os respectivos valores de F1 ponderado. Observou-se novamente uma correlação positiva e significativa entre as diferenças de desempenho e o score de balanceamento ($r = 0.884$, $p < 0.005$).

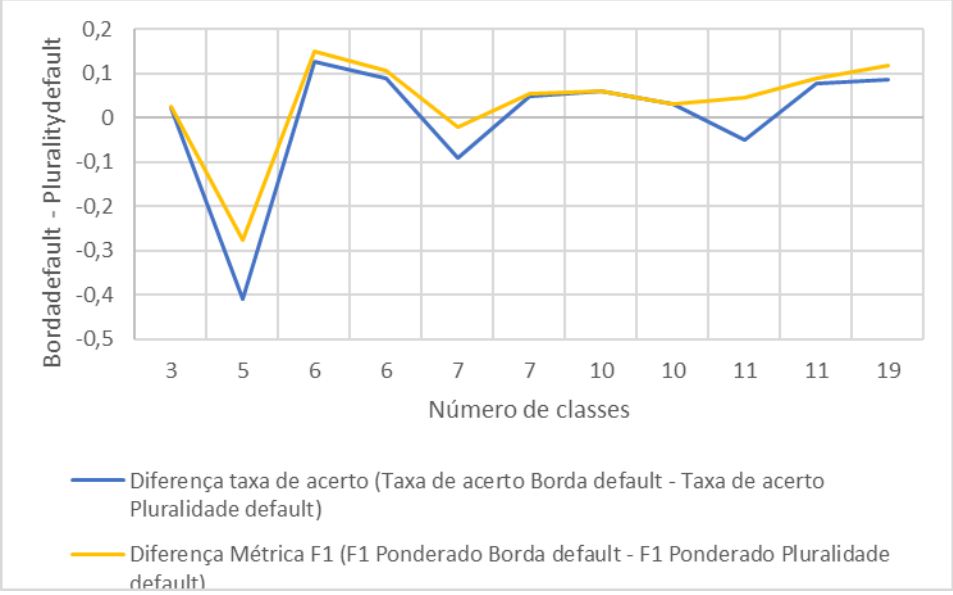
Ainda, tentamos verificar se existe alguma relação entre o número de classes e o desempenho dos modelos agregados pela função Borda ou pela Pluralidade. Como pode-se observar na Figura 5.26, esta relação não pode ser inferida a partir dos conjuntos de dados utilizados.

Figura 5.25: Relação do índice de balanceamento de classes com a diferença entre as execuções Borda default e Pluralidade default, para F1 Ponderado.



Fonte: Autor

Figura 5.26: Relação do número de classes com a diferença entre as execuções "*Borda-default*" e "*Pluralidade-default*", para taxa de acerto e F1 Ponderado.



Fonte: Autor

6 CONCLUSÃO

Em diversos contextos da atualidade, os dados vem sendo distribuídos em diferentes locais, e para extrair informações relevantes deste conjunto de locais, alguns desafios surgem em determinados tipos de tarefa (MOGHADAM; RAVANMEHR, 2017). Uma delas é a classificação, realizada por algoritmos de AM. Uma série de algoritmos bem estabelecidos existe com o intuito de realizar esta tarefa, porém,como visto nas primeiras seções, estes assumem centralização dos dados de treinamento, e o grau de sucesso da tarefa geralmente está atrelado a relevância e completude dos dados.

Como mencionado ao longo do trabalho, existem algumas propostas para realizar a classificação de forma descentralizada, usualmente baseadas em computar a agregação de diversos modelos locais através de métodos algébricos, ou votação simples (MODI; KIM, 2005; BASAK; KOTHARI, 2004). Ainda, há um trabalho relacionado que estuda esta etapa de agregação utilizando funções de escolha social em um cenário de classificação binária (RECAMONDE-MENDOZA; BAZZAN, 2016), discutindo a adequabilidade desta estratégia em lidar com classificação em um cenário descentralizado, sem exigir centralização dos dados brutos e sem comprometer o desempenho do sistema. Portanto, o objetivo deste trabalho foi de propor uma adaptação da agregação por funções de escolha social para o contexto analisado, sendo considerado o cenário de dados com particionamento vertical, apresentando múltiplas classes.

Em suma, o método aqui proposto consiste em computar a classificação multiclasse para cada local dentre os existentes partindo dos algoritmos estudados no Capítulo 2, os quais estão embutidos em agentes classificadores que possuem acesso a um subconjunto de atributos, e em seguida realizando a computação de ordens de preferência para instâncias e classes a partir de cada modelo treinado localmente. Por fim, a agregação dos rankings é realizada por um agente facilitador através de uma função de escolha social pré-selecionada. Neste trabalho, estudamos as funções Borda Count, Copeland, Dowdall e Pluralidade. Como forma de simular o cenário distribuído, os conjuntos de dados utilizados foram particionados conforme o número de agentes classificadores, assimilando atributos aleatórios para cada um deles nas execuções de validação dos resultados.

Para os experimentos, utilizaram-se diversos conjuntos de dados presentes no UCI (LICHMAN, 2013). Buscou-se executar o método com conjuntos diferentes entre si quanto a algumas características: número de classes do problema, número de atributos, número total de instância, e balanceamento de classes. Comparou-se os resultados visua-

lizados pelas abordagens de agregação aqui propostas, com aqueles gerados por modelos centrais (que possuem todos os atributos disponíveis) e modelos locais (um dos agentes classificadores no cenário descentralizado). Esta comparação visa demonstrar o ganho de desempenho dos algoritmos distribuídos com relação aos modelos locais, assim como uma possível similaridade de resultados obtidos com os modelos globais.

Através dos experimentos realizados, verificou-se que para a maior parte dos conjuntos, as funções de escolha social melhoram os resultados de forma significativa com relação aos modelos locais. Isto foi observado tanto em função do desempenho médio, como em função da menor dispersão na distribuição de valores para as métricas de desempenho analisadas. Este resultado não só corrobora achados de estudos anteriores (RECAMONDE-MENDOZA; BAZZAN, 2016), como também evidencia que a aplicação de funções de escolha social não está limitada ao cenário de classificação binária, podendo ser alternativas eficientes mesmo quando os conjuntos de dados são caracterizados por um número maior de classes.

Além disso, descobriu-se uma relação interessante entre o balanceamento de classes e o desempenho de algumas funções a partir dos resultados analisados. Em suma, quanto mais desbalanceadas as classes para um conjunto de dados, maiores as chances de os métodos Borda count, Copeland e Dowdall não apresentarem resultados satisfatórios. Assim, para cenários com desbalanceamento de classe, percebe-se que a pluralidade é a melhor opção para agregação de modelos de classificação em ambientes distribuídos.

Existem algumas possíveis extensões para este trabalho, e uma das possibilidades seria incluir a aplicação da abordagem proposta em dados naturalmente distribuídos, como dados meteorológicos, bem como para análise de dados mais volumosos, como dados genômicos. Além disso, poderia-se estender as análises do capítulo 5 para outras funções de escolha social.

REFERÊNCIAS

ALLWEIN, E. L.; SCHAPIRE, R. E.; SINGER, Y. Reducing multiclass to binary: A unifying approach for margin classifiers. **Journal of machine learning research**, v. 1, n. Dec, p. 113–141, 2000.

ALY, M. Survey on multiclass classification methods. **Neural Netw**, p. 1–9, 2005.

ARROW, K. J.; MASKIN, E. S. **Social Choice and Individual Values**. Yale University Press, 2012. ISBN 9780300179316. Available from Internet: <http://www.jstor.org/stable/j.ctt1nqb90>.

BASAK, J.; KOTHARI, R. A classification paradigm for distributed vertically partitioned data. **Neural computation**, MIT Press, v. 16, n. 7, p. 1525–1544, 2004.

BAY, S. D. Combining nearest neighbor classifiers through multiple feature subsets. In: . [S.l.: s.n.], 1998.

BRANDT, F. et al. **Handbook of Computational Social Choice**. Cambridge University Press, 2016. ISBN 9781107060432. Available from Internet: <https://books.google.com.br/books?id=nMHgCwAAQBAJ>.

CAO, L.; WEISS, G.; YU, P. S. A brief introduction to agent mining. **Autonomous Agents and Multi-Agent Systems**, Springer, p. 1–6, 2012.

CHEUNG, D. W. et al. Efficient mining of association rules in distributed databases. **IEEE transactions on Knowledge and Data Engineering**, IEEE, v. 8, n. 6, p. 911–922, 1996.

CRAMMER, K.; SINGER, Y. On the algorithmic implementation of multiclass kernel-based vector machines. **Journal of machine learning research**, v. 2, n. Dec, p. 265–292, 2001.

DEVI, S. A survey on distributed data mining and its trends. **International Journal of Research in Engineering & Technology (IMPACT: IJRET)**, v. 2, n. 3, p. 107–120, 2014.

DIETTERICH, T. G.; BAKIRI, G. Solving multiclass learning problems via error-correcting output codes. **Journal of artificial intelligence research**, v. 2, p. 263–286, 1995.

DOLHANSKI, B. **Artificial Neural Networks: Linear Multiclass Classification**. 2013. Available from Internet: <http://briandolhansky.com/blog/2013/9/23/artificial-neural-nets-linear-multiclass-part-3>.

FERNANDEZ-LOZANO, C. et al. Improving enzyme regulatory protein classification by means of SVM-RFE feature selection. **Mol. BioSyst.**, The Royal Society of Chemistry, v. 10, p. 1063–1071, 2014. Available from Internet: <http://dx.doi.org/10.1039/C3MB70489K>.

GOVADA, A.; SAHAY, S. K. A communication efficient and scalable distributed data mining for the astronomical data. **Astronomy and Computing**, Elsevier, v. 16, p. 166–173, 2016.

GUESTRIN, C.; FOX, E. **Lecture 66 - Multiclass classification with decision trees**. 2014. Available from Internet: <Lecture66-Multiclassclassificationwithdecisiontrees>.

HOENS, T. et al. Building decision trees for the multi-class imbalance problem. **Advances in knowledge discovery and data mining**, Springer, p. 122–134, 2012.

HONG, J.-H. et al. Fingerprint classification using one-vs-all Support Vector Machines dynamically ordered with naı¨ ve bayes classifiers. **Pattern Recognition**, Elsevier, v. 41, n. 2, p. 662–671, 2008.

HSU, C.-W.; LIN, C.-J. A comparison of methods for multiclass support vector machines. **IEEE transactions on Neural Networks**, IEEE, v. 13, n. 2, p. 415–425, 2002.

KUMAR, S.; GHOSH, J.; CRAWFORD, M. M. Hierarchical fusion of multiple classifiers for hyperspectral data analysis. **Pattern Analysis & Applications**, Springer, v. 5, n. 2, p. 210–220, 2002.

LEE, Y.; LIN, Y.; WAHBA, G. Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. **Journal of the American Statistical Association**, Taylor & Francis, v. 99, n. 465, p. 67–81, 2004.

LICHMAN, M. **UCI Machine Learning Repository**. 2013. Available from Internet: <http://archive.ics.uci.edu/ml>.

LORENA, A. C.; CARVALHO, A. C. de. Estratégias para a combinaçao de classificadores binários em soluçoes multiclasses. **Revista de Informática Teórica e Aplicada**, v. 15, n. 2, p. 65–86, 2008.

MANNING, C. D.; RAGHAVAN, P. **Introduction to information retrieval**. [S.l.: s.n.], 2008.

MCCONNELL, S.; SKILLICORN, D. B. Building predictors from vertically distributed data. In: IBM PRESS. **Proceedings of the 2004 conference of the Centre for Advanced Studies on Collaborative research**. [S.l.], 2004. p. 150–162.

MITCHELL, T. **Machine Learning**. McGraw-Hill, 1997. (McGraw-Hill International Editions). ISBN 9780071154673. Available from Internet: <https://books.google.com.br/books?id=EoYBngEACAAJ>.

MODI, P. J.; KIM, P. W. T. Classification of examples by multiple agents with private features. In: IEEE. **Intelligent Agent Technology, IEEE/WIC/ACM International Conference on**. [S.l.], 2005. p. 223–229.

MOGHADAM, A. N.; RAVANMEHR, R. Multi-agent distributed data mining approach for classifying meteorology data: case study on iran's synoptic weather stations. **International Journal of Environmental Science and Technology**, Springer, p. 1–10, 2017.

MUNOZ, A. Machine learning. 2011. Available from Internet: <https://www.cims.nyu.edu/~munoz/files/ml_optimization.pdf>.

NG, S. S.; TSE, P. W.; TSUI, K. L. A one-versus-all class binarization strategy for bearing diagnostics of concurrent defects. **Sensors**, Multidisciplinary Digital Publishing Institute, v. 14, n. 1, p. 1295–1321, 2014.

OU, G.; MURPHEY, Y. L. Multi-class pattern classification using neural networks. **Pattern Recognition**, Elsevier, v. 40, n. 1, p. 4–18, 2007.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

PETEIRO-BARRAL, D.; GUIJARRO-BERDIÑAS, B. A survey of methods for distributed machine learning. **Progress in Artificial Intelligence**, Springer, p. 1–11, 2013.

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

RAJAN, S.; GHOSH, J. An empirical comparison of hierarchical vs. two-level approaches to multiclass problems. In: SPRINGER. **International Workshop on Multiple Classifier Systems**. [S.l.], 2004. p. 283–292.

RECAMONDE-MENDOZA, M.; BAZZAN, A. L. Social choice in distributed classification tasks: dealing with vertically partitioned data. **Information Sciences**, Elsevier, v. 332, p. 56–71, 2016.

RIFKIN, R. Multiclass classification. 2008. Available from Internet: <https://pdfs.semanticscholar.org/d79a/df69c0d2fa96ebb98767208ae6a4f1185d0f.pdf>.

RIFKIN, R.; KLAUTAU, A. In defense of one-vs-all classification. **Journal of machine learning research**, v. 5, n. Jan, p. 101–141, 2004.

ROKACH, L. Ensemble-based classifiers. **Artificial Intelligence Review**, v. 33, n. 1, p. 1–39, Feb 2010. ISSN 1573-7462. Available from Internet: <http://dx.doi.org/10.1007/s10462-009-9124-7>.

SCIENCE, T. C. for E. **Interview with Dr. Kenneth Arrow**. 2012. Available from Internet: <https://electology.org/podcasts/2012-10-06_kenneth_arrow>.

SOKOLOVA, M.; JAPKOWICZ, N.; SZPAKOWICZ, S. Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation. In: **Australian conference on artificial intelligence**. [S.l.: s.n.], 2006. v. 4304, p. 1015–1021.

WESTON, J.; WATKINS, C. **Multi-class support vector machines**. [S.l.], 1998.

WOOLDRIDGE, M. **An introduction to multiagent systems**. [S.l.]: John Wiley & Sons, 2009.

YU, H.; VAIDYA, J.; JIANG, X. Privacy-preserving SVM classification on vertically partitioned data. In: SPRINGER. **PAKDD**. [S.l.], 2006. v. 3918, p. 647–656.

ZENG, L. et al. Distributed data mining: a survey. **Information Technology and Management**, Springer, v. 13, n. 4, p. 403–409, 2012.