Meta-aprendizado aplicado a fluxos contínuos de dados

André Luis Debiaso Rossi

Meta-aprendizado aplicado a fluxos contínuos de dados

André Luis Debiaso Rossi

Orientador: **Prof. Dr. André Carlos Ponce de Leon Ferreira de Carvalho**

Coorientador**: Prof. Dr. Carlos Manuel Milheiro de Oliveira Pinto Soares**

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências - Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

USP – São Carlos
Fevereiro de 2014

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

R831m

Rossi, André Luis Debiaso
Meta-aprendizado aplicado a fluxos contínuos de dados / André Luis Debiaso Rossi; orientador André Carlos Ponce de Leon Ferreira de Carvalho; co-orientador Carlos Manuel Milheiro de Oliveira Pinto Soares. -- São Carlos, 2014.
189 p.

Tese (Doutorado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2014.

1. Selec?a?o de algoritmos. 2. Meta-aprendizado. 3. Fluxos contínuos de dados. I. de Carvalho, André Carlos Ponce de Leon Ferreira , orient. II. Soares, Carlos Manuel Milheiro de Oliveira Pinto , co-orient. III. Título.

Agradecimentos

À presença Divina em minha vida, principalmente quando pensei estar só e nos momentos difíceis.

Ao meu orientador Prof. Dr. André Carvalho, pelos ensinamentos, pela confiança depositada em mim, pelos momentos de positivismo e pela sua amizade. Mesmo com o enorme reconhecimento dos seus pares, a humildade e a gentileza com que trata todos a sua volta o torna um exemplo de pessoa e profissional que desejo ser.

Ao meu coorientador Prof. Dr. Carlos Soares, pela sua grande contribuição com este trabalho e pelas longas discussões e *emails*, que contribuíram muito para esta tese e para meu crescimento profissional. O seu entusiasmo pela profissão é contagiante e serviu de combustível para que eu continuasse a trilhar esse caminho. Agradeço também pela amizade e por ter me recebido tão bem no Porto.

Aos professores Ricardo Prudêncio, Renato Tinós e Ricardo Campello, que fizeram parte da banca examinadora da minha qualificação e contribuíram com ideias e direcionamentos para o desenvolvimento deste trabalho.

À Carlos Soares e Jorge Kanda pelos códigos para caracterização dos dados e avaliação de rankings.

À Sociedade de Transportes Colectivos do Porto SA (STCP) pelo fornecimento dos dados e à João Mendes Moreira pela ajuda com esses dados.

Aos meus pais João e Maria, pelas constantes provas de amor incondicional e pelos ensinamentos de honestidade e humildade. As pessoas, assim como as máquinas, aprendem pela experiência que lhes é transmitida, e eu não poderia ter recebido melhores exemplos. Espero que o meu algoritmo de aprendizado seja tão eficiente quanto os seus, para que eu possa generalizar para o resto da minha vida com tanta precisão quanto vocês conseguiram.

Aos meus irmãos Flávio, Regina e Ana Elisa, por todo apoio, pelo carinho recebido e por serem as pessoas com as quais eu sempre poderei contar.

À minha namorada Joyce, pelo seu amor, pelo seu companheirismo e pela sua cumplicidade. Tu me iniciaste nessa jornada e não deixaste de me dar apoio, principalmente

nos momentos difíceis.

Aos meus sobrinhos Ana, Samuel e Diogo. A alegria e o sorriso de vocês me fazem um bem enorme e eu lembro de que não precisamos de muita coisa para sermos felizes. Os momentos de sapequice são inúmeros e as brincadeiras de esconde-esconde com Ana e Samuel sempre exigem muito de mim; eu nunca consigo encontrá-los!

Aos amigos do BIOCOM, pelo acolhimento, pelo apoio e também pelos momentos de descontração. Sem a presença de vocês essa jornada não teria sido tão agradável. Meus agradecimentos especiais aos amigos Bruno, Renatinho, Luís Paulo, Valéria, Ivani e Marina, por terem me ajudado diretamente com esta tese. Aos amigos de outros laboratórios, funcionários e professores do ICMC-USP, pelo incentivo, pelos ensinamentos e pelas conversas sobre futebol.

Aos amigos que fiz durante a minha estadia em Portugal, principalmente aos alunos e professores do INESC-LIAAD. Lugares em que somos bem recebidos, como eu fui nesse laboratório e na cidade do Porto, nos deixa saudades, o que é um grande incentivo para voltar!

Aos amigos de longas datas e aos recentes, aos distantes e aqueles que moram ao lado. Obrigado pelo apoio e pelo carinho, mesmo quando estive ausente por muito tempo.

À máquina de café do BIOCOM e às pessoas que dela cuidavam. Ela foi muito importante e nunca havia falhado enquanto estive no laboratório.

À beleza do nascer e do pôr do sol. O primeiro batia à minha janela (quase) todos os dias enquanto morei em São Carlos, e o segundo me fazia companhia durante as inúmeras viagens à minha cidade natal. – *"Porque a luz do sol vale mais que os pensamentos de todos os filósofos e de todos os poetas."* - Alberto Caeiro - Heterônimo de Fernando Pessoa

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e à Fundação para a Ciência e Tecnologia pelo apoio financeiro durante o período de estágio de doutorado sanduíche no Porto, Portugal (programa CAPES/FCT, projeto 224/09, processo BEX3231/10-0).

À Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) pelo apoio financeiro essencial para a realização deste trabalho (processo número 2008/11569-6).

Resumo

Algoritmos de aprendizado de máquina são amplamente empregados na indução de modelos para descoberta de conhecimento em conjuntos de dados. Como grande parte desses algoritmos assume que os dados são gerados por uma função de distribuição estacionária, um modelo é induzido uma única vez e usado indefinidamente para a predição do rótulo de novos dados. Entretanto, atualmente, diversas aplicações, como gerenciamento de transportes e monitoramento por redes de sensores, geram fluxos contínuos de dados que podem mudar ao longo do tempo. Consequentemente, a eficácia do algoritmo escolhido para esses problemas pode se deteriorar ou outros algoritmos podem se tornar mais apropriados para as características dos novos dados. Nesta tese é proposto um método baseado em meta-aprendizado para gerenciar o processo de aprendizado em ambientes dinâmicos de fluxos contínuos de dados com o objetivo de melhorar o desempenho preditivo do sistema de aprendizado. Esse método, denominado MetaStream, seleciona regularmente o algoritmo mais promissor para os dados que estão chegando, de acordo com as características desses dados e de experiências passadas. O método proposto emprega técnicas de aprendizado de máquina para gerar o meta-conhecimento, que relaciona as características extraídas dos dados em diferentes instantes do tempo ao desempenho preditivo dos algoritmos. Entre as medidas usadas para extrair informação relevante dos dados, estão aquelas comumente empregadas em meta-aprendizado convencional com diferentes conjuntos de dados, que são adaptadas para as especificidades do cenário de fluxos, e de áreas correlatas, que consideram, por exemplo, a ordem de chegada dos dados. O MetaStream é avaliado para três conjuntos de dados reais e seis algoritmos de aprendizado diferentes. Os resultados mostram a aplicabilidade do MetaStream e sua capacidade de melhorar o desempenho preditivo geral do sistema de aprendizado em relação a um método de referência para a maioria dos problemas investigados. Deve ser observado que uma combinação de modelos mostrou-se superior ao MetaStream para dois conjuntos de dados. Assim, foram analisados os principais fatores que podem ter influenciado nos resultados observados e são indicadas possíveis melhorias do método proposto.

Abstract

Machine learning algorithms are widely employed to induce models for knowledge discovery in databases. Since most of these algorithms suppose that the underlying distribution of the data is stationary, a model is induced only once e it is applied to predict the label of new data indefinitely. However, currently, many real applications, such as transportation management systems and monitoring of sensor networks, generate data streams that can change over time. Consequently, the effectiveness of the algorithm chosen for these problems may deteriorate or other algorithms may become more suitable for the new data characteristics. This thesis proposes a metalearning based method for the management of the learning process in dynamic environments of data streams aiming to improve the general predictive performance of the learning system. This method, named MetaStream, regularly selects the most promising algorithm for arriving data according to its characteristics and past experiences. The proposed method employs machine learning techniques to generate metaknowledge, which relates the characteristics extracted from data in different time points to the predictive performance of the algorithms. Among the measures applied to extract relevant information are those commonly used in conventional metalearning for different data sets, which are adapted for the data stream particularities, and from other related areas that consider the order of the data stream. We evaluate MetaStream for three real data stream problems and six different learning algorithms. The results show the applicability of the MetaStream and its capability to improve the general predictive performance of the learning system compared to a baseline method for the majority of the cases investigated. It must be observed that an ensemble of models is usually superior to MetaStream. Thus, we analyzed the main factors that may have influenced the results and indicate possible improvements for the proposed method.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AD	Árvores de Decisão
AM	Aprendizado de Máquina
CVFDT	*Concept-adapting Very Fast Decision Tree*
CART	*Classification and Regression Trees* (Árvores de Classificação e Regressão)
CEP	Controle Estatístico de Processos
EDP	*Electricity Demand Prediction* (Predição da Demanda de Eletricidade)
FCD	Fluxos Contínuos de Dados
VIC	*Victoria*
VN	Verdadeiro Negativo
VP	Verdadeiro Positivo
IBL	*Instance-Based Learning* (Aprendizado Baseado em Instâncias)
ILP	*Inductive Logic Programming* (Programação Lógica Indutiva)
KDD	*Knowledge Discovery in Databases* (Descoberta de Conhecimento em Bases de Dados)
LR	*Linear Regression* (Regressão Linear)
MAE	*Mean Absolute Error* (Erro Absoluto Médio)
MARS	*Multivariate Adaptive Regression Splines*
MD	Mineração de Dados
MDA	*Mean Decrease Accuracy* (Redução Média da Acurácia)
MDInd	Meta-dados Independentes

MDIndDep Meta-dados Independentes e Dependentes

MSE *Mean Squared Error* (Erro Quadrático Médio)

NFL *No Free Lunch*

NMSE *Normalized Mean Squared Error* (Erro Quadrático Médio Normalizado)

NSW *New South Wales*

PHT *Page-Hinkley Test*

PPR *Project Pursuit Regression*

RF *Random Forests*

RMSE *Root Mean Squared Error* (Raiz do Erro Quadrático Médio)

RSE *Relative Squared Error* (Erro Quadrático Relativo)

SVM *Support Vector Machines* (Máquinas de Vetores de Suporte)

TTP *Travel Time Prediction* (Predição do Tempo de Viagem)

VFDT *Very Fast Decision Tree*

VN Verdadeiro Negativo

VP Verdadeiro Positivo

WGK *Weighted Goodman-Kruskal*

Lista de Símbolos

ω	Tamanho de uma janela deslizante (em número de exemplos)
τ	Fator de decaimento
z_i	Exemplo i no nível base
$\mathbf{x}$	Conjunto de p valores dos atributos preditivos $\{x_1, x_2, \ldots, x_p\}$ no nível base
y	Atributo alvo do nível base
$\mathbf{y}$	Conjunto de valores do atributo alvo no nível base
MIN_T	Variável cumulativa de todos os valores observados até o momento (ou exemplo) T
δ_{PHT}	Magnitude das mudanças permitidas no PHT
α_{PHT}	Limiar para sinalizar uma mudança no PHT
α_{DT}	Limiar para sinalizar uma mudança no teste de Kifer et al. (2004)
α_{CEP}	Limiar para sinalizar uma mudança no teste CEP
β_{CEP}	Limiar do estado de alerta no teste CEP
α_{ROT}	Limiar da diferença entre os erros de dois regressores para que eles sejam considerados semelhantes
L_i	Algoritmo de aprendizado i
M_i	Modelo preditivo i
ω_b	Tamanho da janela deslizante de treinamento no nível base (número de exemplos de treinamento)
η_b	Horizonte de predição no nível base (em número de exemplos)
λ_b	Número de exemplos de teste no nível base
e_i	Meta-exemplo i
$\mathbf{a}$	Conjunto de q valores dos meta-atributos preditivos $\{a_1, a_2, \ldots, a_q\}$
c	Meta-atributo alvo
$\mathbf{c}$	Conjunto de valores do meta-atributo alvo
γ	Tamanho do conjunto de seleção (em número de exemplos)
ω_m	Tamanho da janela deslizante de treinamento no nível meta (número de meta-exemplos de treinamento)
η_m	Horizonte de predição no nível meta (em número de exemplos)
λ_m	Número de exemplos de teste no nível meta

Capítulo 1

Introdução

A maioria das pessoas talvez não saibam, mas já utilizam, direta ou indiretamente, técnicas de Aprendizado de Máquina (AM), que estão presentes em diversas aplicações, como na identificação de mensagens eletrônicas não solicitadas (*spams*) e na construção de perfis de usuários em sites de compra na internet (Flach, 2012). Essas técnicas auxiliam na descoberta de regularidades ou padrões que podem estar ocultos nesses dados. AM é comumente definido como o estudo sistemático de sistemas e algoritmos que melhoram automaticamente com a experiência (Mitchell, 1997; Flach, 2012). A experiência pode ser entendida como um conjunto de dados que é usado por um algoritmo de aprendizado para gerar modelos que posteriormente são aplicados para predição de dados futuros ou para outros tipos de tomada de decisão sob incerteza. Em geral, quanto mais dados (experiência) estiverem disponíveis, melhores podem ser os modelos aprendidos (Mitchell, 1997).

O desempenho dos diferentes algoritmos de aprendizado varia de acordo com o domínio e o conjunto de dados sob análise (Gordon e Desjardins, 1995; Pfahringer et al., 2000; Caruana e Niculescu-Mizil, 2006; Statnikov et al., 2008), pois cada algoritmo possui um viés embutido. Sob o ponto de vista de AM, viés pode ser definido como qualquer base ou preferência para a escolha de um modelo sobre outro (Mitchell, 1997). Assim, todo algoritmo de aprendizado executa a busca em um espaço de generalização restrito, definido por uma linguagem de representação e emprega um viés de busca para selecionar uma generalização nesse espaço (Brazdil et al., 2009). As diferenças de desempenho entre os algoritmos observadas nos estudos empíricos supracitados são suportadas pelo teorema *No Free Lunch* (Wolpert, 1996), que afirma a inexistência de um algoritmo universal capaz de obter desempenho preditivo superior aos demais para todos os problemas. Portanto, idealmente, um algoritmo adequado (ou uma combinação deles) deve ser escolhido de acordo com as características do problema sob análise. Para isso, usuários precisam do auxílio de um especialista da área ou podem fazer a seleção por tentativa e erro, esperando que o conjunto de suposições embutidas em um dos algoritmos testados seja apropriado para as características dos dados que estão sendo estudados (Brazdil et al.,

2009). Nenhuma das soluções é totalmente satisfatória para o usuário final, que espera poder ter acesso a essas ferramentas com baixo custo e de forma efetiva (Giraud-Carrier et al., 2004).

Uma possibilidade para a seleção automática de algoritmos de aprendizado é o uso de métodos baseados em meta-aprendizado. Segundo Rendell et al. (1987), meta-aprendizado é aprender por meio da experiência quando diferentes vieses podem ser utilizados para tratar um problema. Meta-aprendizado difere do aprendizado convencional ou base, no escopo do nível de adaptação. O aprendizado base ocorre no nível dos exemplos e o viés é fixado a priori, enquanto em meta-aprendizado é escolhido dinamicamente com base no acúmulo da experiência para diferentes conjuntos de dados. A experiência é adquirida do próprio processo de aprendizado e é denominada meta-dados ou meta-conhecimento (Brazdil et al., 2009). Um algoritmo de AM é aplicado aos meta-dados para induzir um meta-modelo que é capaz de mapear as características dos conjuntos de dados ao desempenho dos modelos. Esse meta-modelo pode então ser usado para determinar quais algoritmos são os mais apropriados para um novo conjunto de dados, sem a necessidade de induzir todos os respectivos modelos (Vilalta e Drissi, 2002; Giraud-Carrier et al., 2004).

Como a maioria dos algoritmos de AM assume que os dados são gerados por uma função de distribuição estacionária, uma única amostra estática dos dados é usada, tipicamente muitas vezes pelos vários algoritmos disponíveis, na indução de um modelo, o qual será usado indefinidamente para novos exemplos (Alpaydin, 2010). Portanto, a seleção do melhor algoritmo para um conjunto de dados é realizada uma única vez. Porém, atualmente, muitos dados são gerados em ambientes dinâmicos e em larga escala por dispositivos digitais, conhecidos como Fluxos Contínuos de Dados[1] (FCD), e estão sujeitos a mudanças ao longo do tempo (Gama, 2010). Esses FCD podem ser encontrados, por exemplo, em aplicações como monitoramento de tráfego de redes de computadores, detecção de fraudes em cartões de crédito e redes de sensores. As alterações que ocorrem nesses dados são conhecidas como mudanças de conceito, e podem ser distinguidas em relação à velocidade com que ocorrem, abrupta (*concept drift*) ou gradual (*concept shift*), e em relação ao tipo, real ou virtual, sendo que esses dois tipos podem ocorrer simultaneamente (Tsymbal, 2004; Beringer e Hüllermeier, 2007; Gama et al., 2014). A mudança real de conceito ocorre na distribuição condicional da variável de saída que se deseja predizer, enquanto a distribuição das variáveis de entrada permanece inalterada. A mudança virtual de conceito ocorre na distribuição das variáveis de entrada, mas o conceito da variável de saída não muda. Dependendo do objetivo do trabalho, a distinção entre mudança real ou virtual não é importante, visto que nos dois casos pode ser necessário atualizar ou substituir o modelo (Tsymbal, 2004).

As mudanças de conceito podem ser tratadas por sistemas de aprendizado explicita-

[1] A expressão fluxo de dados é usada neste trabalho como sinônimo de fluxo contínuo de dados

mente, empregando mecanismos e técnicas para detectá-las (Kifer et al., 2004; Bifet e Gavaldà, 2007; Vallim et al., 2013). Porém, detectar quando os dados em um fluxo estão mudando não é uma tarefa trivial (Aggarwal, 2003). Assim, alguns sistemas tratam as mudanças de conceito implicitamente (Koychev, 2000; Klinkenberg, 2004; Moreira, 2008), trocando regularmente o modelo possivelmente obsoleto por um modelo induzido apenas com os dados mais recentes. Esse procedimento pode ser realizado usando mecanismos de esquecimento, como as janelas deslizantes, que descartam os exemplos mais antigos. Em qualquer dos casos, a atualização ou substituição de modelos usando os dados mais recentes pode não ser suficiente para garantir o melhor modelo preditivo possível, pois o viés embutido no algoritmo de aprendizado pode não ser o mais adequado para as novas características dos dados. A seleção de um algoritmo de aprendizado diferente pode ser benéfica quando o desempenho preditivo do modelo induzido pelo algoritmo atual é inferior ao de um modelo induzido por um outro algoritmo. Isso ocorre quando o algoritmo alternativo se torna mais adequado do que o atual e começa a induzir modelos com maior poder de generalização ou quando há uma degradação no desempenho preditivo do modelo atual.

Uma alternativa para, possivelmente, reduzir o problema de um único viés em FCD com mudanças de conceito, é o uso de *ensembles*, em que as predições individuais de vários modelos são combinadas por meio de um método, como votação, para formar uma predição final (Dietterich, 2000). Apesar do grande poder preditivo, os *ensembles* são de difícil interpretabilidade e são computacionalmente custosos, pois necessitam treinar diversos modelos. Contudo, *ensembles* com custo computacional aceitável para o tratamento de FCD estão sendo desenvolvidos (Kolter e Maloof, 2007; Bifet et al., 2009). Uma outra alternativa, pouco explorada em FCD, é a seleção de algoritmos por meio de meta-aprendizado. Métodos baseados em meta-aprendizado já foram propostos para dados que mudam ao longo do tempo com o objetivo de selecionar modelos quando os conceitos são recorrentes (Gama e Kosina, 2011) ou para a seleção de algoritmos (Klinkenberg, 2005). Entretanto, no primeiro trabalho, o problema do viés inadequado não é tratado, pois um único algoritmo é considerado, enquanto no último, o método é demasiadamente custoso e as características dos dados do nível base não são usadas para auxiliar na seleção do melhor algoritmo.

1.1 Objetivos e hipóteses

As mudanças que comumente ocorrem em dados que são gerados continuamente em ambientes dinâmicos podem afetar o poder de generalização do modelo atual e também o desempenho do algoritmo de aprendizado usado para induzir novos modelos ou atualizá-los, pois o viés embutido nesse algoritmo pode não ser adequado para as novas características dos dados. Essa observação motivou a **hipótese principal** desta tese de

doutorado:

> *Em ambientes em que os dados são gerados continuamente e ocorrem mudanças de conceito, a seleção contínua e dinâmica do algoritmo mais adequado para as características dos dados em cada instante de tempo resulta na melhoria do desempenho preditivo geral do sistema de aprendizado.*

A seleção de algoritmos pode ser promissora em problemas de fluxo de dados, principalmente quando o desempenho médio dos algoritmos são similares para uma grande quantidade de dados, mas diferentes quando analisados sob a perspectiva de uma granularidade menor. Se um algoritmo induz modelos preditivos que são claramente superiores aos demais, a seleção de algoritmos não resultará em grandes melhorias do sistema de aprendizado. A partir do estabelecimento da hipótese principal, definiu-se o **objetivo geral** desta tese de doutorado:

> *Desenvolvimento de um método baseado em meta-aprendizado para gerenciar automaticamente, ao longo do tempo, o processo de sistemas de aprendizado em fluxos de dados com mudanças de conceito.*

Quanto maior for o sucesso da escolha do viés mais adequado para cada instante de tempo, maior poderá ser o poder de generalização do sistema de aprendizado no nível base, principalmente quando: i) o algoritmo atual é inadequado para os novos dados e, portanto, ocorre a degradação do desempenho preditivo do modelo induzido por esse algoritmo; ii) não há degradação do desempenho preditivo do modelo atual, mas existe outro algoritmo mais adequado às características dos dados atuais capaz de induzir um modelo com maior poder de generalização do que o atual.

Para avaliar a veracidade da hipótese principal e atingir o objetivo geral desta tese, investigou-se os princípios de funcionamento de técnicas comumente usadas nas áreas de meta-aprendizado e FCD, o que resultou no método **MetaStream**. A cada instante de tempo, o método MetaStream mapeia as características extraídas dos dados passados com o desempenho obtido pelos modelos, a fim de predizer o melhor algoritmo para os dados que estão chegando no fluxo e que ainda não foram preditos. Esse método consiste de adaptações e extensões das atuais abordagens de meta-aprendizado para as especificidades dos problemas de FCD. O desenvolvimento de meta-atributos, que são as variáveis que caracterizam os dados, permite ilustrar bem uma dessas ideias. Por um lado, é possível desenvolver novos meta-atributos para caracterização contínua de FCD seguindo metodologias comuns em meta-aprendizado, como meta-atributos baseados em modelos (Vilalta et al., 2004). Por outro lado, também é possível usar medidas já conhecidas em FCD e áreas correlatas, mas que nunca foram usadas em meta-aprendizado.

Nesta tese, a seleção de algoritmos é tratada como um problema de classificação (Kalousis, 2002; Lemke e Gabrys, 2010). Portanto, o método de referência para comparação

com o MetaStream consiste em selecionar o melhor algoritmo com base na classe majoritária do conjunto de treinamento, conhecida também como classe padrão, assim como é feito em muitos trabalhos de meta-aprendizado (Kalousis et al., 2004; Prudêncio e Ludermir, 2004). Como o conjunto de treinamento é atualizado ao longo do tempo, com a inclusão dos exemplos mais recentes e o descarte dos exemplos mais antigos, a classe padrão pode mudar a cada instante de tempo. O método de referência será denominado *Default*, por ser um termo em inglês usual em meta-aprendizado.

A seleção do melhor algoritmo é realizada pelos dois métodos continuamente, sem o uso de um mecanismo de detecção de mudanças. A opção por não utilizar tal mecanismo se deve ao fato de que o sucesso ou fracasso da seleção de algoritmos ficaria dependente desse mecanismo, o que dificultaria a avaliação do método proposto. Por exemplo, os mecanismos que monitoram o desempenho dos modelos preditivos para detectar mudanças de conceito (Widmer e Kubat, 1996; Gama et al., 2004; Klinkenberg, 2004) não são capazes de identificar mudanças que não afetam o sistema de aprendizado. Nesse caso, embora o desempenho preditivo do modelo não degrade, modelos induzidos por outros algoritmos podem, em determinados instantes de tempo, se tornar melhores do que os atuais.

A seguir são estabelecidas outras hipóteses específicas derivadas da hipótese principal e dos objetivos gerais.

Hipótese específica 1

A maioria dos estudos de meta-aprendizado para a seleção de algoritmos usa, com sucesso, o mapeamento entre as características dos dados e o desempenho preditivo dos modelos para esses dados como meta-conhecimento para determinar o melhor algoritmo para novos conjuntos de dados (Brazdil et al., 2009). Essa observação e o fato de que o método Default prediz o melhor algoritmo apenas com base no atributo alvo dos dados de treinamento motivaram a seguinte hipótese específica:

> *O método MetaStream pode ser aplicado para a seleção de algoritmos de aprendizado para problemas de fluxo de dados com desempenho preditivo superior ao do método de referência Default.*

Para verificar essa hipótese, supõe-se que os exemplos chegam um a cada instante de tempo ou aos lotes e os métodos MetaStream e Default são aplicados para selecionar o melhor algoritmo para cada lote de exemplos, que pode ter tamanho diferente do número de exemplos que chegam a cada instante. Klinkenberg (2005) também propõe um método para seleção de algoritmos baseado em meta-aprendizado, mas supõe que os exemplos chegam, necessariamente, aos lotes, pois o seu método baseia-se em propriedades do aprendizado desses lotes, como o número de lotes sem mudanças de conceito e o algoritmo com maior sucesso para o último lote. Diferentemente, o método MetaStream

extrai características dos dados do nível base e tenta mapear essas características com o desempenho dos algoritmos para esses dados.

Hipótese específica 2

A ideia de selecionar um algoritmo para cada lote de exemplos surgiu da hipótese de que um determinado conceito permanece inalterado por um período mínimo de tempo (Harries et al., 1998; Klinkenberg, 2005; Gama, 2010). Porém, mesmo considerando lotes com poucos exemplos, não há garantias de que um único algoritmo terá o melhor desempenho preditivo para todos os exemplos desse lote. Em aplicações que possuem sazonalidade diária e semanal, como no problema de transportes coletivos investigado em Moreira (2008), em que todos os exemplos gerados em um dia constituem um único lote, o melhor algoritmo pode variar para dados produzidos em um único dia ou para diferentes dias da semana. Portanto, determinar o número de exemplos em cada lote não é trivial. Devido às limitações da seleção de algoritmos para lotes de exemplos e com base nos trabalhos de Gama e Kosina (2009, 2011), que realizaram a seleção de modelos preditivos para cada exemplo de um fluxo de dados, elaborou-se a seguinte hipótese:

> *O uso do método MetaStream para a seleção de algoritmos para cada exemplo do FCD é viável e resulta em melhor desempenho preditivo do sistema de aprendizado a nível base comparado à seleção para lotes de exemplos.*

Para avaliar essa hipótese, o método MetaStream, que foi inicialmente projetado para a seleção de algoritmos para lotes de exemplos, foi adaptado para que fosse possível selecionar um algoritmo de aprendizado para cada novo exemplo que chega no nível base. A principal adaptação necessária foi na caracterização dos dados. Enquanto na seleção em lote, medidas que caracterizam conjuntos de valores podem ser aplicadas para os dados de treinamento e teste, na seleção unitária há somente um exemplo no conjunto de teste a cada instante de tempo. Portanto, as características que descrevem esse exemplo no nível base também são usadas para caracterizá-lo no nível meta, como em Gama e Kosina (2011). O desempenho preditivo do método MetaStream para a seleção de um algoritmo para cada exemplo é comparado com a seleção de um algoritmo para cada lote de exemplos.

Hipótese específica 3

Em meta-aprendizado convencional com dados estacionários, a seleção de algoritmos é realizada para diferentes conjuntos de dados, que possuem morfologias distintas, ou seja, são descritos por conjuntos de atributos que podem variar em relação à quantidade (de algumas unidades até milhares) e tipo (numérico, nominal, etc.). Portanto, as medidas para a extração de características desses conjuntos são calculadas sobre todos os atributos e depois as características precisam ser sumarizadas em um único valor, para que todos os

conjuntos sejam descritos pelo mesmo número de meta-atributos, o que é uma restrição dos algoritmos de aprendizado proposicionais (Raedt, 2008). Em FCD, a seleção de algoritmos é realizada para o mesmo problema em diferentes instantes de tempo. Portanto, como a morfologia dos dados é a mesma (ou raramente muda) ao longo do tempo, é possível extrair características para cada atributo ou relações entre atributos, as quais podem ser usadas diretamente no nível meta como meta-atributos, sem necessidade de agregação. A perda de informação relevante que pode ocorrer durante o processo de agregação das características motivou a seguinte hipótese:

> *Na utilização de meta-aprendizado para dados com a mesma morfologia é possível usar características específicas dos dados, que contêm informação que contribui para melhorar a predição do melhor algoritmo para esses dados.*

Para verificar essa hipótese, o desempenho do método MetaStream é comparado quando dois conjuntos diferentes de meta-dados são utilizados: o primeiro é composto apenas pelos meta-atributos independentes da morfologia, ou seja, meta-atributos que podem ser usados mesmo que os conjuntos de dados possuam diferentes morfologias, como nas aplicações convencionais de meta-aprendizado; o segundo é composto pelos mesmos meta-atributos independentes com a adição dos meta-atributos dependentes da morfologia, que só podem ser usados se os dados possuem a mesma morfologia, que é o caso de um fluxo de dados.

Hipótese específica 4

Assim como Kalousis (2002), os métodos MetaStream e Default são aplicados na avaliação das hipóteses anteriores para a seleção do melhor algoritmo dentre pares de algoritmos. A hipótese de que o método MetaStream terá desempenho preditivo superior ao do Default na predição do melhor algoritmo para pares de algoritmos (hipótese específica 1) motivou a seguinte hipótese:

> *As predições realizadas pelo método MetaStream podem ser combinadas para a recomendação de algoritmos em forma de rankings com desempenho superior às predições do método Default.*

Para avaliar essa hipótese, as predições realizadas para cada par de algoritmos são combinadas de maneira que a posição de um algoritmo no ranking é determinado pelo número de vezes que esse algoritmo é selecionado dentre todos os pares de algoritmos. A acurácia dos rankings construídos com as predições de cada método é medida pela similaridade com os rankings verdadeiros. A mesma ideia é usada por Kalousis (2002), mas a avaliação é realizada comparando-se apenas o algoritmo que está no topo (melhor) do ranking real com o ranking predito.

1.2 Organização da tese

O restante do texto desta tese de doutorado está organizada como segue.

No **Capítulo 2** são apresentados os principais conceitos sobre mineração convencional de dados, realizada com conjuntos de dados supostamente estacionários. O foco é direcionado para o aprendizado supervisionado de problemas de regressão e classificação. Em seguida, alguns conceitos básicos sobre FCD são introduzidos e os principais trabalhos na área de mineração de FCD encontrados na literatura são discutidos.

No **Capítulo 3** é apresentada uma visão geral da área de meta-aprendizado e as principais etapas comumente encontradas nos métodos para seleção de algoritmos, como a caracterização dos dados, a indução do meta-modelo e a sua aplicação para a predição de algoritmos. Nesse capítulo também são descritos e discutidos os trabalhos que utilizaram meta-aprendizado para a seleção de algoritmos ou modelos em FCD, os quais motivaram algumas ideias do método proposto nesta tese.

No **Capítulo 4** é apresentado o método proposto neste trabalho para a seleção de algoritmos. Como o MetaStream é baseado em conceitos de meta-aprendizado com adaptações para se adequar às especificidades de fluxos de dados que podem mudar ao longo do tempo, a apresentação do funcionamento geral do método foi dividida em: i) nível base: os algoritmos usam os dados mais recentes para induzir modelos que são aplicados para predizer o atributo alvo de interesse; ii) nível meta: as características dos dados do nível base e o desempenho preditivo dos algoritmos para esses dados são usados para induzir um meta-modelo, que então é aplicado para predizer o melhor algoritmo para os dados que estão chegando. Além da estrutura geral do método, são discutidas diversas possibilidades para a caracterização de FCD e as diferenças em relação à caracterização de conjuntos de dados estacionários. Por último, a questão da complexidade do método é brevemente discutida.

No **Capítulo 5** é apresentado o planejamento de experimentos, incluindo todos os materiais e métodos envolvidos nesse processo: conjuntos de dados, algoritmos e parâmetros usados nos níveis base e meta, as medidas e o processo de extração de características dos dados, as abordagens de rotulação dos meta-dados e os métodos de avaliação nos níveis base e meta. Adicionalmente, são apresentados os experimentos preliminares para ajustar os valores de alguns parâmetros.

No **Capítulo 6** são apresentados e discutidos os resultados experimentais realizados para avaliar o método MetaStream e verificar a veracidade das hipóteses estabelecidas nesta tese. A avaliação é realizada para os dois níveis de aprendizado (meta e base) para todos os experimentos, com exceção da recomendação de rankings de algoritmos, que é avaliada somente no nível meta, pois os rankings podem ser usados no nível base para diferentes propósitos, como a seleção de um único algoritmo ou uma combinação qualquer de algoritmos.

No **Capítulo 7** são apresentadas as considerações finais desta tese, suas principais contribuições, as limitações do método proposto e da avaliação experimental empregada, além de possíveis direcionamentos para trabalhos futuros. Por último, são listadas as publicações resultantes desta tese.

Capítulo 2

Mineração de Fluxos Contínuos de Dados

O rápido crescimento da capacidade de gerar e coletar dados possibilitou que muitas empresas e instituições armazenem enormes volumes de dados. Porém, a obtenção de conhecimento a partir desses dados ainda é um grande desafio. Mineração de dados (MD) é uma área que estuda como extrair informações não-triviais de grandes quantidades de dados com o objetivo de descobrir padrões desconhecidos, tendências inesperadas ou outras relações presentes nesses dados (Han e Kamber, 2006). De maneira simplificada, MD refere-se à extração de conhecimento ou "mineração" de grandes quantidades de dados e pode ser entendida como o resultado natural da evolução da tecnologia da informação (Han e Kamber, 2006; Tan et al., 2005). A MD é uma área multidisciplinar, incluindo aprendizado de máquina, estatística, tecnologias de bancos de dados, visualização de informações, reconhecimento de padrões e computação de alto desempenho. Este capítulo trata especificamente do uso de técnicas de aprendizado de máquina (AM) para a MD.

Nas últimas duas décadas, pesquisas em AM têm focado no aprendizado em lote (*batch*), geralmente com pequenos conjuntos de dados. No aprendizado em lote, o conjunto de treinamento completo está disponível para o algoritmo, que gera um modelo de decisão depois de processar todos os dados, eventualmente, diversas vezes. A razão dessa prática é que os exemplos são gerados aleatoriamente de acordo com uma função de probabilidade estacionária, embora desconhecida (Gama e Rodrigues, 2009).

Atualmente, é grande o número de sistemas computacionais que geram dados automaticamente, sem a interferência humana, conhecidos como Fluxos Contínuos de Dados (FCD) (Gama, 2010). Esses dados são gerados em aplicações como monitoramento de tráfego de redes de computadores, gerenciamento de dados de telecomunicações, monitoramento de redes de sensores e detecção de fraudes em cartões de crédito. Nessas aplicações, os dados são gerados ilimitadamente (final aberto), eventualmente em alta velocidade e os conceitos contidos nos dados podem mudar ao longo do tempo. Algumas dessas características podem variar entre diferentes problemas de FCD ou diferentes ins-

tantes de tempo, como a taxa com que os dados são gerados e a velocidade de mudança dos conceitos. Considerar todas essas características durante o aprendizado envolve uma adaptação das atuais técnicas de AM e o desenvolvimento de novos algoritmos que sejam adequados para a obtenção de conhecimento desses dados.

As próximas seções deste capítulo estão organizadas como segue. Na Seção 2.1, é apresentada uma visão geral das áreas de MD e AM e como os problemas de regressão e classificação são tratados sob a perspectiva dessas áreas. Em seguida, na Seção 2.2 são abordados os principais conceitos sobre mineração de FCD, incluindo os mecanismos de esquecimento e o tratamento das mudanças de conceito, além de citar os principais trabalhos de mineração de FCD e os métodos de avaliação. Por último são feitas algumas considerações finais sobre este capítulo na Seção 2.3.

2.1 Mineração de dados e aprendizado de máquina

A ubiquidade dos sistemas computacionais e a possibilidade de transmissão de dados por redes de computadores estão provocando uma inundação de dados. Com o avanço das tecnologias de armazenamento, empresas especializadas conseguem manter uma grande quantidade desses dados por um curto período de tempo ou indeterminadamente, dependendo da quantidade dados gerados. Por outro lado, a capacidade de analisar essa gigantesca quantidade de dados com o objetivo de encontrar informações úteis ainda é pequena. Nesse cenário, a área de mineração de dados recebe cada vez mais atenção. Mineração de dados pode ser definida como o processo automático de descoberta de padrões que representam o conhecimento implícito em grandes volumes de dados. Em alguns meios, como na indústria, o termo mineração de dados é usado como sinônimo do processo de descoberta de conhecimento de bases de dados (KDD, em inglês, *Knowledge Discovery in Databases*). Porém, o mais comum é entender MD como uma etapa essencial do KDD, que envolve (Fayyad et al., 1996): i) coleta de dados; ii) pré-processamento de dados; iii) transformação de dados; iv) mineração de dados; v) avaliação e interpretação dos resultados.

A área de aprendizado de máquina (AM) provê ferramentas que podem ser usadas no processo de MD e têm alcançado grande sucesso em tarefas como reconhecimento da fala e de padrões. Aprendizado de máquina pode ser definido como o estudo de algoritmos e sistemas que melhoram automaticamente com a experiência (Mitchell, 1997). A "experiência" pode ter diferentes formas, mas para o propósito deste trabalho é entendida como um conjunto de dados que são usados para gerar modelos que posteriormente podem ser empregados para predição de dados futuros ou para outros tipos de tomada de decisão sob incerteza ou para a explicação ou compreensão dos padrões ou regularidades que foram detectados (Flach, 2012; Witten e Frank, 2005).

O aprendizado que é utilizado para a criação do modelo a partir de dados previ-

amente observados é o aprendizado indutivo. Portanto, um algoritmo de aprendizado induz um modelo usando um conjunto de dados disponível. O aprendizado indutivo pode ser dividido em aprendizado supervisionado, não-supervisionado e semisupervisionado. No aprendizado supervisionado, cada exemplo do conjunto de dados possui um rótulo, informado por um supervisor. Assim, o algoritmo de aprendizado utiliza essa informação para aprender a associação entre as características dos exemplos e seus rótulos. Se os rótulos forem discretos, o problema é conhecido como classificação; se forem contínuos, como regressão. Os dois problemas possuem muitas similaridades e também são conhecidos como aproximação de funções (Hastie et al., 2009). Nos métodos não-supervisionados, não há um supervisor para informar os rótulos de cada exemplo. Nesse caso, o algoritmo utiliza apenas os dados de entrada e precisa encontrar padrões ou regularidades nesses dados. No aprendizado semisupervisionado, exemplos em que os rótulos são conhecidos e também exemplos em que os rótulos não são conhecidos são apresentados ao algoritmo. A seguir, são introduzidos os conceitos do aprendizado supervisionado para regressão e classificação necessários para o entendimento do restante desta tese.

2.1.1 Regressão

Regressão é um conceito espalhado em diferentes áreas do conhecimento. Nesta tese, esse conceito está restrito à área de aprendizado indutivo. Em problemas de regressão, dado um exemplo $z = (\mathbf{x}, y)$, em que $\mathbf{x}$ é um vetor de p variáveis preditoras, $\mathbf{x} = (x_1, x_2, \ldots, x_p)$, também denominadas de variáveis independentes ou atributos preditivos, o objetivo é predizer o valor da variável de resposta numérica ou atributo alvo y.

Técnicas de regressão paramétricas assumem que a forma do relacionamento funcional entre as variáveis preditoras e a variável de resposta é conhecida, reduzindo o problema para a estimação de um conjunto de parâmetros. Por exemplo, a regressão linear pode ser usada quando a relação entre os atributos e a variável de resposta pode ser aproximada por uma reta. No caso de uma única variável independente x, é fácil visualizar como o valor de y pode ser estimado por meio de uma reta, como mostrado na Figura 2.1. Nesse caso, um modelo de regressão linear pode ser expresso como: $y = \phi_1 + x\phi_2 + \varepsilon$, em que ϕ_2 é a inclinação da reta e ϕ_1 é o ponto onde x cruza o eixo y ($x = 0$). Assim, o processo básico da regressão linear para um conjunto de dados é estimar os parâmetros (coeficientes) ϕ_i de cada atributo preditivo x_i, que representa o efeito que esse atributo tem sobre a variável de resposta y (Nisbet et al., 2009). Essa estimativa pode ser feita, por exemplo, usando o método dos mínimos quadrados, que minimiza a distância entre a reta real que separa os dados e a reta estimada. Depois de encontrar os valores desses parâmetros, o modelo é aplicado para novos dados para os quais não se conhece o valor de y. Por outro lado, técnicas não-paramétricas, como árvores de decisão (Breiman et al.,

1984), fazem apenas algumas suposições gerais sobre a forma da função, que será estimada a partir dos dados (Uysal e Güvenir, 1999).

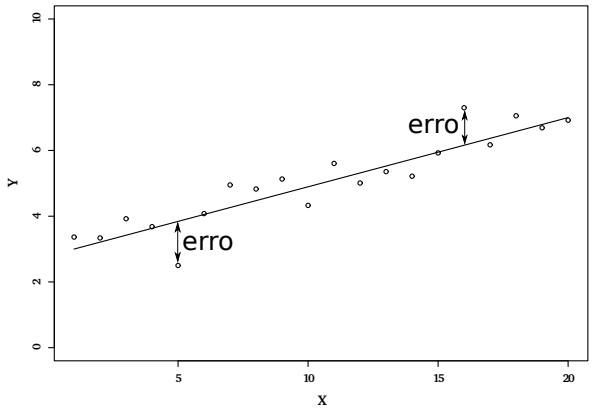


Figura 2.1: Exemplo de um problema em que a relação entre x e y é estimada por uma reta.

O processo de aprendizado para problemas de regressão consiste em induzir uma função $\hat{f} : \mathbf{X} \rightarrow \mathbb{R}$, em que $\hat{f}$ é denominada regressor, modelo ou hipótese. Idealmente, $\hat{f}(\mathbf{x}) = f(\mathbf{x}), \forall \mathbf{x} \in \mathbf{X}$, em que f representa a função real desconhecida. Porém, na prática, $\hat{f}$ não é idêntica a f, mas uma função de aproximação que minimiza a diferença entre as duas. A função real f pode ser decomposta como mostrado na Equação 2.1, em que $g(\mathbf{x})$ é uma função determinística e ε é o erro ou ruído (Cherkassky e Mulier, 1998). Se a função aprendida $\hat{f}(\mathbf{x}) = f(\mathbf{x}), \forall \mathbf{x} \in \mathbf{X1}$, em que $\mathbf{X1}$ é uma amostra da população $\mathbf{X}$, significa que $\hat{f}$ está aprendendo também o ruído ε, e, portanto, ocorreu *overfitting*. O problema contrário, *underfitting*, também pode acontecer, ou seja, $\hat{f}$ não aprender $g(\mathbf{x})$. Em ambos os casos o poder de generalização da função $\hat{f}$ será afetado (Moreira, 2008; Mendes-Moreira et al., 2012).

$$f(\mathbf{x}) = g(\mathbf{x}) + \varepsilon \tag{2.1}$$

Avaliação

O erro de generalização dos modelos construídos para problemas de regressão pode ser calculado usando diferentes medidas. Na prática, deve ser escolhida uma ou mais medidas que sejam adequadas para o problema tratado. Em geral, as medidas mais comumente usadas são simples de serem calculadas. Algumas dessas medidas são apresentadas na Tabela 2.1 (Witten e Frank, 2005), em que $\hat{\mathbf{y}} = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n\}$ são os valores preditos e $\mathbf{y} = \{y_1, y_2, \ldots, y_n\}$ são os valores reais.

A primeira medida dessa tabela é o erro quadrático médio (MSE, do inglês, *mean squared error*), que é uma das mais usadas na literatura. Muitas vezes, a raiz quadrada é utilizada para manter a mesma unidade dos valores preditos, denominada de raiz do erro

quadrático médio (RMSE, do inglês, *root mean squared error*). O erro absoluto médio (MAE, do inglês, *mean absolute error*) é uma medida alternativa que calcula uma média da magnitude dos erros individuais, sem considerar seus sinais. A MSE tende a aumentar o efeito de valores mais discrepantes presentes nos dados, quando alguns erros de predição são maiores do que outros. A MAE não tem esse efeito, pois todos os erros são tratados igualmente de acordo com suas magnitudes.

Em muitas aplicações, não é o erro absoluto que é importante, mas sim o erro relativo. O erro quadrático relativo (RSE, do inglês, *relative squared error*) refere-se a uma medida diferente das anteriores, pois é obtida pela razão do erro quadrático do modelo induzido pelo algoritmo de regressão e do erro quadrático de um preditor simples, que sempre utiliza a média do atributo alvo do conjunto de treinamento para predizer o atributo alvo dos exemplos de teste. Assim, um algoritmo de regressão deve ter RSE entre 0 e 1 (próximo de zero é melhor) para ser considerado melhor do que esse preditor simples. A principal vantagem é que sua interpretação é independente da unidade de medida usada para expressar o atributo alvo. Na área de meta-aprendizado, a medida RSE também é conhecida como erro quadrático médio normalizado (NMSE, do inglês, *normalized mean squared errror*) (Brazdil et al., 2009). O erro absoluto relativo (RAE, do inglês, *relative absolute error*) é simplesmente o erro absoluto total, ao invés do erro quadrático, mas com o mesmo tipo de normalização do RSE (Armstrong e Collopy, 1992).

Tabela 2.1: Medidas de avaliação do desempenho preditivo de modelos para problemas de regressão.

Medida de desempenho	**Fórmula**
Erro Quadrático Médio	$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$
Raiz do Erro Quadrático Médio	$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$
Erro Absoluto Médio	$MAE = \frac{\lvert p_1 - a_1\rvert + \cdots + \lvert p_n - a_n\rvert}{n}$
Erro Quadrático Relativo	$RSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \mu_{\mathbf{y}})^2}$, em que $\mu_{\mathbf{y}} = \frac{1}{n}\sum_{i}^{n} y_i$
Erro Absoluto Relativo	$RAE = \frac{\sum_{i=1}^{n}\lvert y_i - \hat{y}_i\rvert}{\sum_{i=1}^{n}\lvert y_i - \mu_{\mathbf{y}}\rvert}$,
Coeficiente de Correlação	$r = \frac{\sum_{i=1}^{n}(y_i - \mu_{\mathbf{y}})(\hat{y}_i - \mu_{\hat{\mathbf{y}}})}{\sqrt{\sum_{i=1}^{n}(y_i - \mu_{\mathbf{y}})^2 \sum_{i=1}^{n}(\hat{y}_i - \mu_{\hat{\mathbf{y}}})^2}}$, em que $\mu_{\hat{\mathbf{y}}} = \frac{1}{n}\sum_{i}^{n}\hat{y}_i$

A última medida da Tabela 2.1 é o coeficiente de correlação, que mede o grau de associação entre as variáveis. O coeficiente de correlação de Pearson (r), ou simplesmente correlação de Pearson, é a medida mais comum de correlação e reflete o grau de associação

linear entre duas variáveis. O coeficiente r pode assumir valores entre -1 e 1. Quando $r = 1$, significa que há uma correlação perfeita positiva entre as duas variáveis, ou seja, as duas possuem a mesma tendência de crescimento ou decrescimento. Quando $r = -1$, significa que há uma correlação perfeita negativa entre as duas variáveis, ou seja, os valores de uma variável estão crescendo enquanto os da outra variável estão decrescendo. Quando $r = 0$, significa que as duas variáveis são absolutamente independentes. Porém, como a correlação de Pearson avalia apenas a associação linear entre as variáveis, pode existir uma dependência não linear e outras medidas devem ser investigadas. A correlação é diferente das outras medidas porque possui uma escala independente. Além dessa diferença, na correlação, o maior valor possível (1) é o que indica o melhor desempenho, ao contrário das medidas de erro, em que os menores valores são os que indicam o melhor desempenho.

2.1.2 Classificação

Em problemas de classificação, dado um exemplo $z = (\mathbf{x}, y)$, em que $\mathbf{x}$ é um vetor de p valores dos atributos preditivos $\mathbf{x} = (x_1, x_2, \ldots, x_p)$, o objetivo é predizer o rótulo categórico da variável de resposta ou atributo alvo y. Esse processo se resume basicamente em construir um modelo (também denominado de classificador ou hipótese) a partir dos dados observados que seja capaz de classificar exemplos com rótulos desconhecidos. Em alguns casos, como no aprendizado baseado em exemplos (ou memória) (IBL, do inglês, *instance-based learning*) (Aha et al., 1991) não há a construção de um modelo e todos os exemplos de treinamento são usados para a predição da classe de novos casos. Na Figura 2.2 é ilustrado um diagrama do processo de criação e uso de um modelo. Primeiramente, o conjunto de dados de treinamento, no qual os rótulos das classes dos exemplos são conhecidos, é utilizado por um algoritmo de aprendizado para construir um modelo. Após a construção, esse classificador pode ser aplicado para predizer as classes dos exemplos do conjunto de teste, cujas classes reais são desconhecidas.

Uma característica importante dos modelos é que eles possuam boa capacidade de generalização, ou seja, consigam predizer com alta taxa de acerto rótulos das classes para exemplos que não foram apresentados anteriormente (Tan et al., 2005). Tais modelos podem ser induzidos por diversos algoritmos de AM e não existe um específico que gere melhor resultado do que os demais em todos os problemas, dado que cada um apresenta um viés indutivo diferente (Wolpert, 1996).

Outra característica dos classificadores a ser observada é em relação a interpretabilidade proporcionada ao ser humano. Os classificadores do tipo caixa-preta são aqueles que possuem uma representação interna que, geralmente, não se consegue interpretar facilmente, ou seja, é difícil conhecer as características do problema que o levaram a uma determinada dedução. Um exemplo de classificador desse tipo são as redes neurais artificiais (Haykin, 1999). Os classificadores orientados a conhecimento, como as árvo-

res de decisão (Breiman et al., 1984), criam estruturas simbólicas que normalmente são mais compreensíveis do que os classificadores do tipo caixa-preta (Monard e Baranauskas, 2003).

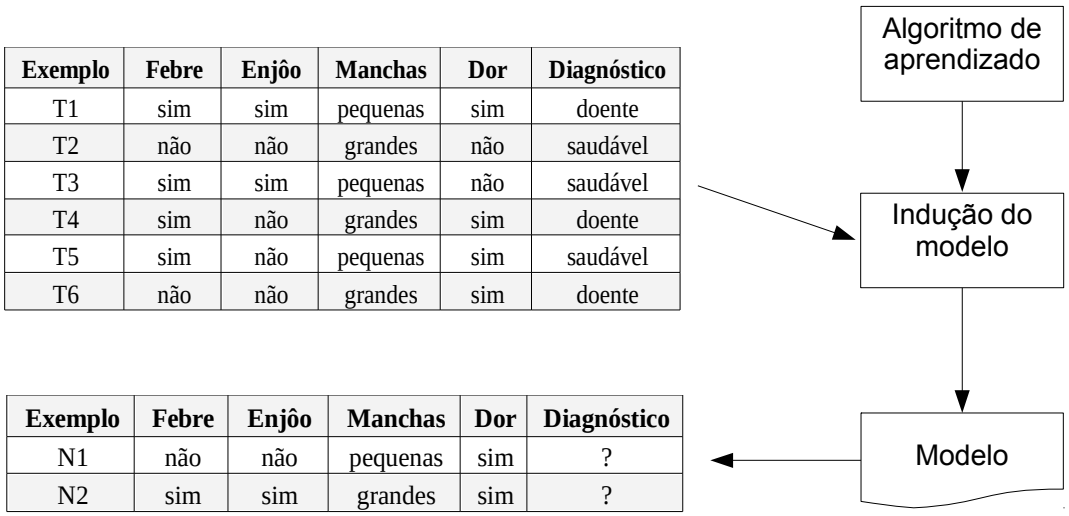


Figura 2.2: Diagrama do processo de indução de um classificador e sua utilização na dedução de novos exemplos.

Avaliação

Em AM, diferentes medidas têm sido utilizadas para avaliar o desempenho de modelos preditivos em problemas de classificação. A afirmação de qual é o melhor modelo pode variar com a medida utilizada na avaliação. Diversos trabalhos já se dedicaram a investigar e descrever as medidas comumente usadas pela comunidade de AM, como Ferri et al. (2009) e Japkowicz e Shah (2011). Porém, a medida mais adequada depende principalmente das características do problema sob análise e do objetivo final da avaliação.

Em problemas de classificação, a taxa de erro de um modelo para um conjunto de teste com m exemplos é definida pela Equação 2.2, em que $\hat{y}_i$ é a classe predita, y_i é a classe observada e L é uma função de perda ou função de custo que, geralmente, é definida pela Equação 2.3. Outras medidas para avaliar o desempenho podem ser calculadas a partir de uma tabela de contingência com frequências absolutas, conhecida como matriz de confusão, exibida na Tabela 2.2 para um problema de duas classes: positiva e negativa. Nessa tabela as linhas indicam a classe verdadeira e as colunas indicam a classe predita. As siglas VP, VN, FP e FN correspondem a:

- Verdadeiro Positivo (VP): total de exemplos classificados como positivos e que realmente são positivos;
- Verdadeiro Negativo (VN): total de exemplos classificados como negativos e que realmente são negativos;

- Falso Positivo (FP): total de exemplos classificados como positivos mas que na verdade são negativos;
- Falso Negativo (FN): total de exemplos classificados como negativos mas que na verdade são positivos.

$$\text{taxa de erro} = \frac{1}{m}\sum_{i=1}^{m} L(y_i, \hat{y}_i) \tag{2.2}$$

$$L(\hat{y}, y) = \begin{cases} 0, & \text{se } \hat{y}_i = y_i \\ 1, & \text{se } \hat{y}_i \neq y_i \end{cases} \tag{2.3}$$

Tabela 2.2: Matriz de confusão para um problema de duas classes: positiva e negativa.

		classe predita		
		Positiva	Negativa	Total
classe verdadeira	Positiva	VP	FN	$Y_P = VP + FN$
	Negativa	FP	VN	$Y_N = FP + VN$
	Total	$\hat{Y}_P = VP + FP$	$\hat{Y}_N = FN + VN$	m

O total de exemplos verdeiros das classes positiva e negativa são dados por Y_P e Y_N, respectivamente, enquanto o total de exemplos preditos como positivos e negativos são dados por $\hat{Y}_P$ e $\hat{Y}_N$. A soma dos valores da diagonal principal da Tabela 2.2 representa o número total de exemplos classificados corretamente, enquanto a soma dos valores da diagonal secundária representa o número total de exemplos classificados incorretamente. A taxa de acerto ou acurácia é definida pela Equação 2.4 a partir dessa matriz. A medida de precisão, dada pela equação 2.5, estima a probabilidade da predição positiva estar correta. A sensibilidade ou *recall* estima a probabilidade de um exemplo pertencente à classe positiva ser predito como positivo e é definida pela Equação 2.6. A medida de especificidade estima a probabilidade de um exemplo pertencente à classe negativa ser predito corretamente como negativo e é definida pela Equação 2.7.

$$\text{taxa de acerto} = \text{acurácia} = \frac{VP + VN}{m} \tag{2.4}$$

$$\text{precisão} = \frac{VP}{\hat{Y}_P} \tag{2.5}$$

$$\text{sensibilidade} = \frac{VP}{Y_P} \tag{2.6}$$

$$\text{especificidade} = \frac{VN}{Y_N} \tag{2.7}$$

A estatística *kappa* é outra medida em que o problema de desbalanceamento das classes, ou seja, quando há uma considerável diferença no número de exemplos pertencentes a cada classe, é minimizado, pois ela considera a distribuição de classes no seu cálculo, assim como as medidas de sensibilidade e especificidade. Essa medida é definida segundo a Equação 2.8, em que p_0 é a acurácia do classificador e p_c é a probabilidade devido à chance, que pode ser obtida a partir da matriz de confusão. Sendo m o número total de exemplos, p_c é definido pela Equação 2.9. O valor de κ varia de -1 a 1, em que -1 significa total discordância, 0 um classificador aleatório e 1 total concordância.

$$\kappa = \frac{p_0 - p_c}{1 - p_c} \tag{2.8}$$

$$p_c = \frac{Y_P}{m} \times \frac{\hat{Y}_P}{m} + \frac{Y_N}{m} \times \frac{\hat{Y}_N}{m} \tag{2.9}$$

2.2 Mineração de fluxos contínuos de dados

Ao contrário dos conjuntos de dados convencionais, FCD entram e saem de sistemas de computação continuamente sem nenhuma intervenção humana e com várias taxas de atualização. Com o crescente interesse na mineração desses fluxos, um grande número de estudos tem sido realizado e muitos algoritmos têm sido desenvolvidos (vide Gaber et al. (2005), Žliobaite et al. (2012), Gama et al. (2014) e referências ali contidas). Para extrair conhecimento ou descobrir padrões em FCD é necessário desenvolver métodos de análise de simples varredura usando uma capacidade limitada de processamento e armazenamento (Han e Kamber, 2006; Gama, 2010). Uma outra questão importante na mineração desses dados é a possibilidade de mudanças. A maioria dos algoritmos de aprendizado convencionais assume que os exemplos de um determinado problema ou conjunto de dados são gerados aleatoriamente de acordo com uma distribuição estacionária. Portanto, uma vez que um modelo é induzido com uma quantidade suficiente de dados, não há necessidade de atualizá-lo no futuro. Entretanto, em FCD gerados em ambientes dinâmicos, é muito provável que ocorram mudanças nos dados ao longo do tempo. Assim, uma abordagem natural para o aprendizado é utilizar sistemas capazes de se adaptar, ou seja, que considerem as mudanças de conceito (Gama e Rodrigues, 2009).

Embora muitos trabalhos em FCD desenvolvam métodos e algoritmos para tratar dados que são gerados em grandes quantidades e com altas taxas de velocidade, a avaliação, geralmente, é realizada com problemas com milhares de exemplos, principalmente quando se tratam de dados reais (Bifet, 2009). Para muitos desses casos, a afirmação que os dados são transitórios e não podem ser mantidos permanentemente não é verdadeira dada a capacidade das atuais tecnologias de armazenamento.

Nesta tese, são analisados problemas que geram dezenas ou centenas de dados diari-

amente, ou seja, a velocidade do fluxo não é tão intensa (Han e Kamber, 2006), e que foram usados em outros estudos de FCD, inclusive com algoritmos de aprendizado em lote (Seção 5.1). Independentemente da abordagem utilizada, muitos conceitos de FCD são úteis para o aprendizado dos conjuntos de dados investigados neste trabalho.

A seguir, são apresentados os conceitos necessários sobre FCD para o estudo realizado nesta tese. Na Seção 2.2.1 são descritos os mecanismos de esquecimento janelas deslizantes e fatores de decaimento. Em seguida, na Seção 2.2.2 são apresentados os tipos de mudanças de conceitos existentes e as principais técnicas para detecção e tratamento dessas mudanças. Na Seção 2.2.3 é descrito o funcionamento dos principais algoritmos e técnicas para mineração de FCD (os trabalhos de meta-aprendizado aplicados a FCD são apresentados no Capítulo 3, depois da apresentação dos principais conceitos sobre meta-aprendizado). Por último, na Seção 2.2.4 são apresentados os métodos para avaliação do desempenho preditivo de modelos em problemas de FCD.

2.2.1 Mecanismos de esquecimento

A abordagem mais comum para lidar com dados que mudam ao longo do tempo é o esquecimento das observações antigas, que podem representar um conceito diferente daquele dos dados mais recentes. Em FCD, a janela deslizante é o mecanismo mais comumente empregado para esse propósito. Apenas os exemplos que estão dentro dessa janela são usados na indução de modelos. Babcock et al. (2002) definem dois tipos básicos de janelas deslizantes: janelas baseadas em sequência e janelas baseadas em tempo. O tamanho da janela baseada em sequência é definido pelo número de exemplos. Dois modelos diferentes são as janelas deslizantes de tamanho fixo ω (Figura 2.3) e as janelas de pontos relevantes (Figura 2.4), que armazenam todos os dados após um determinado instante de tempo. Nos dois modelos, a janela desliza a cada λ exemplos, conhecido como passo da janela. O tamanho das janelas baseadas em tempo é definido por um tempo de duração ao invés de um número de exemplos. Assim, uma janela de tamanho ω consiste de todos os exemplos que chegaram dentro de um intervalo de tempo ω. As janelas de tempo comprimidas representam um modelo em que os dados mais recentes são armazenados com maiores detalhes e os mais antigos são sumarizados (Figuras 2.5 e 2.6). A seguir esses modelos de janelas são descritos com mais detalhes.

Janelas de pontos relevantes

Nesse modelo, janelas sucessivas compartilham pontos e são crescentes em relação ao tamanho (Figura 2.4). Todos os dados recebidos após um instante de tempo considerado relevante são armazenados até o próximo ponto relevante, que pode ser, por exemplo, quando uma mudança é detectada. Nesse caso, o tamanho da janela cresce enquanto não forem detectadas mudanças. Se ocorrer uma mudança, a janela atual termina e uma nova

janela vazia começa depois do ponto de mudança.

Janelas deslizantes

Ao invés de utilizar todos os exemplos recebidos desde o início do fluxo, pode ser mais interessante analisar apenas os dados mais recentes. As janelas deslizantes de tamanho fixo representam uma abordagem simples para esse fim (Figura 2.3). Esse modelo de janela é similar a uma estrutura de dados do tipo fila: quando um ou mais exemplos recentes são inseridos na janela, a mesma quantidade dos exemplos mais antigos, que provavelmente pertencem a um conceito obsoleto, são descartados (esquecidos);

Janelas comprimidas

Os modelos de janelas anteriores de pontos relevantes e deslizantes usam um método de esquecimento definitivo, ou seja, qualquer exemplo passado está ou não dentro da janela. No modelo de janelas comprimidas, os dados mais recentes são armazenados com detalhes dentro da janela, enquanto os dados antigos são sumarizados ou armazena-se apenas as informações mais importantes, sendo o nível de detalhamento (granularidade) dependente da aplicação. Janelas de tempo comprimidas podem ser concebidas de diferentes maneiras. Duas variantes possíveis são (Han e Kamber, 2006): janelas comprimidas naturais e janelas comprimidas logarítmicas. Na primeira, os dados são armazenados com a granularidade definida por uma taxonomia do tempo natural: última hora com granularidade de 15 minutos (4 pontos), último dia com granularidade em horas (24 pontos), último mês com granularidade em dias (30 pontos) e último ano com granularidade em meses (12 pontos) (Figura 2.5). Na segunda variante, a granularidade decresce de forma logarítmica com a idade dos dados (Figura 2.6). Com o passar do tempo, a janela armazena os dois últimos períodos de tempo t e consecutivamente agrega os demais com menor granularidade (2 períodos, 4 períodos, 8 períodos e assim por diante).

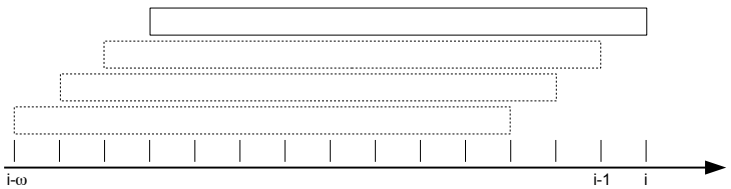


Figura 2.3: Janela deslizante.

Um outro mecanismo para o esquecimento de dados antigos são os fatores de decaimento. Esse mecanismo é baseado na ideia simples de que o esquecimento deve ocorrer gradualmente e não abruptamente como nas janelas deslizantes e de pontos relevantes, ou seja, a importância de um dado diminui com o passar do tempo. Para isso, cada exemplo é associado a um peso ou fator de decaimento τ, $0 < \tau < 1$, que reflete a sua idade. Quanto mais recente for um exemplo, maior será a sua importância e quanto mais antigo,

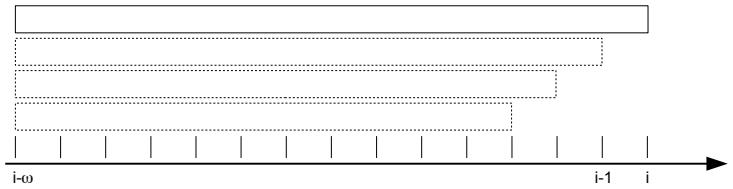


Figura 2.4: Janela de pontos relevantes.

Figura 2.5: Janela comprimida natural.

menor ela será. Portanto, nesse modelo, todos os exemplos são considerados, com maior ou menor importância. Esse fator τ pode, por exemplo, ter um decaimento exponencial, assim, um exemplo terá peso τ^n após n exemplos terem sido processados após sua chegada. A desvantagem desse mecanismo é que ele precisa estar embutido no processo de aprendizado, ao contrário das janelas deslizantes (Koychev, 2000; Klinkenberg, 2004; Chen et al., 2009).

As janelas deslizantes e os fatores de decaimento são usados também para avaliar o desempenho dos modelos induzidos (Gama et al., 2013), baseado na intuição de que o interesse do especialista é maior no comportamento dos modelos para os dados mais recentes. Outros exemplos e referências para mecanismos de esquecimento podem ser encontradas em Widmer e Kubat (1996), Klinkenberg e Renz (1998), Maloof e Michalski (2000), Chen et al. (2009) e referências ali contidas.

2.2.2 Mudanças de conceito

Os dados que são produzidos continuamente em ambientes dinâmicos e em larga escala estão sujeitos a mudanças ao longo do tempo. Essas alterações são chamadas de mudanças de conceito e podem ser reais ou virtuais. A mudança real de conceito ocorre na distribuição condicional da variável de saída a ser predita, $p(y|\mathbf{x})$, enquanto a distribuição das variáveis de entrada permanece inalterada. Um exemplo de aplicação em que isso ocorre é a filtragem de informação, ou seja, a classificação adaptativa de documentos com respeito ao interesse particular de um pessoa (Klinkenberg, 2005). Por exemplo, um usuário de um portal da internet que estava interessado em comprar um carro inicialmente classificava todas as notícias relacionadas a esse assunto como relevantes. Ao passar do tempo, começou a descartar notícias sobre modelos que não lhe interessavam, até que comprou o carro desejado e todas as notícias relacionadas a esse assunto passaram a ser classificadas como irrelevantes por esse usuário. Uma outra aplicação é a de monitoramento

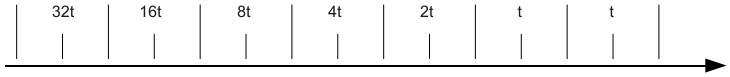


Figura 2.6: Janela comprimida logarítmica.

de redes de computadores, pois a natureza dos ataques mudam em resposta à melhoria das estratégias de prevenção. Por outro lado, a mudança virtual de conceito ocorre na distribuição dos dados, ou seja, $p(\mathbf{x})$ muda e leva a mudanças na fronteira de decisão. A mudança de conceito virtual pode ocorrer, por exemplo, na categorização de *spams*. Enquanto o entendimento do que é um *spam* possa permanecer inalterado relativamente por um longo período, a frequência dos diferentes tipos de *spams* pode mudar drasticamente com o tempo. Dependendo do objetivo do estudo, a distinção entre mudança real ou virtual não é importante, pois nos dois casos pode ser necessário atualizar ou substituir o modelo (Tsymbal, 2004).

As mudanças também podem ser distinguidas em relação à velocidade: gradual ou abrupta. Nos exemplos anteriores, o leitor que queria comprar um carro passou a descartar gradualmente notícias que não lhe interessavam, enquanto que na categorização de *spams*, um domínio que antes era classificado automaticamente como *spam* passa a ser, abruptamente, considerado como um *email* relevante, se o usuário alterar manualmente a categoria dos *emails* recebidos daquele domínio. As mudanças graduais, conhecidas como *concept drift*, normalmente são mais difíceis de serem detectadas do que as mudanças abruptas, conhecidas como *concept shift*, pois as perturbações inicias podem ser entendidas como ruídos. Quando essa dúvida acontece, o sistema aguarda a chegada de novos casos para tentar distinguir se está ocorrendo uma mudança ou se são apenas interferências de dados ruidosos. Em um caso extremo, as alterações podem ocorrer tão lentamente que podem ser confundidas com estacionariedade ou podem mudar tão rapidamente que não é possível aprender nenhum conceito (Gama et al., 2014).

As causas de uma mudança também é outro fator importante para a sua detecção. Segundo Gama (2010), duas causas podem ser reconhecidas: as mudanças no contexto do aprendizado e nas variáveis observadas. Os algoritmos aprendem a partir de observações que são descritas por um conjunto finito de atributos. Em problemas reais, podem existir propriedades importantes do domínio estudado que não estão sendo descritas pelos atributos, conhecidas como variáveis ocultas (Harries et al., 1998), e que podem influenciar no aprendizado desses dados. Portanto, o comportamento dos modelos pode variar de acordo com o contexto que não está sendo utilizado para descrever o problema (contexto oculto). Ao contrário das mudanças que ocorrem nas variáveis observadas, essa mudança, portanto, não pode ser detectada apenas pela análise dos atributos preditivos dos dados.

Klinkenberg (2005) afirma que o aprendizado de mudanças de conceito é inviável se nenhuma restrição é imposta, mas pode ser bem sucedido se a taxa ou a extensão das

mudanças for limitada de diferentes maneiras. Helmbold e Long (1994) assumem a possibilidade permanente, mas devagar, de mudanças de conceito e definem extensão de uma mudança como a probabilidade de que dois conceitos subsequentes sejam diferentes para um exemplo escolhido aleatoriamente. Os resultados reportados pelos autores incluem um limitante superior para a tolerância máxima de mudança aceita por qualquer algoritmo de aprendizado. Os mesmos autores propuseram também algoritmos que são capazes de aprender conceitos que não mudam mais do que uma certa constante e mostraram que é suficiente para um algoritmo de aprendizado utilizar um número fixo dos exemplos mais recentes. Assim, usando uma janela de tempo com um tamanho mínimo previamente definido, é possível aprender conceitos para o qual a extensão de mudança é apropriadamente limitada (Klinkenberg, 2005).

Na prática, porém, normalmente não é possível garantir que uma certa aplicação obedeça a essas restrições. Por exemplo, um leitor de notícias pode mudar seus interesses frequentemente e quase que arbitrariamente. Nesse caso, janelas de tempo com uma grande quantidade de exemplos, para os quais os resultados teóricos funcionam, seriam impraticáveis. Assim, muitas abordagens precisam usar janelas de tamanho fixo menores ou heurísticas que ajustam o tamanho das janelas automaticamente. Essas técnicas, normalmente, funcionam melhor do que janelas de tamanho fixo ou maiores (Widmer e Kubat, 1996; Klinkenberg e Renz, 1998). Enquanto essas heurísticas são intuitivas e funcionam bem em aplicações de domínios particulares, elas normalmente requerem ajuste de parâmetros, não são freqüentemente transferíveis para outros domínios e ainda sofrem da falta de uma fundamentação teórica (Klinkenberg, 2005).

Em geral, abordagens que lidam com mudanças de conceito podem ser classificadas em duas categorias (Gama et al., 2004): i) na abordagem implícita, o sistema de aprendizado é adaptado em intervalos regulares, sem verificar se de fato ocorreram mudanças; ii) na abordagem explícita são usados mecanismos para detectar mudanças e em seguida o sistema é adaptado de acordo com as mudanças ocorridas. A seguir são apresentados alguns exemplos dessas abordagens.

Adaptação regular do sistema de aprendizado

Quando os dados são gerados por uma função de distribuição estacionária, quanto mais dados estiverem disponíveis para treinar um modelo, melhor esse modelo poderá ser. Porém, quando esses dados mudam ao longo do tempo, o mais interessante pode ser utilizar abordagens que adaptem ou induzam novos modelos em intervalos (de tempo ou exemplos) regulares. As janelas deslizantes e os fatores de decaimento são as abordagens mais usadas para esse fim. Quando uma janela deslizante é usada, o algoritmo de aprendizado é atualizado ou induzido apenas com os exemplos que estão incluídos nessa janela cada vez que o seu conteúdo mudar. Entretanto, uma dificuldade dessa abordagem é selecionar o tamanho apropriado da janela. Janelas pequenas podem assegurar uma rápida

adaptação em fases com mudanças de conceito. Uma janela maior, por outro lado, produziria bons resultados em fases estáveis, mas poderia não reagir rapidamente a mudanças de conceito (Gama et al., 2004). Ao contrário das janelas de tempo, quando fatores de decaimento são usados, os dados mais recentes possuem, de fato, maior importância (Chen et al., 2009).

Os sistemas de aprendizado *online* processam um exemplo por vez e, futuramente, não têm acesso aos exemplos anteriores. Esses sistemas, como o WINNOW (Littlestone, 1988) e o VFDT (Domingos e Hulten, 2000) podem ser vistos como naturalmente adaptativos às mudanças que podem ocorrer. Isso porque eles atualizam continuamente o modelo com os dados mais recentes e, consequentemente, os dados mais antigos perdem cada vez mais a influência sobre o comportamento do modelo. A principal limitação é que essa adaptação ocorre lentamente, o que pode não ser adequado em casos de mudanças abruptas, dependendo da sensibilidade desses modelos à atualização com novos exemplos (Gama et al., 2014).

Detecção de mudanças

Detectar mudanças de conceito não é uma tarefa trivial (Aggarwal, 2003), pois, como mencionado anteriormente, elas podem ser reais ou virtuais, podem ocorrer abrupta ou gradualmente e podem ter diferentes causas. Um aspecto interessante é a possibilidade de prover descrições significativas sobre as mudanças, como indicar os pontos em que ocorreram e quantificá-las (Gama e Rodrigues, 2007). Uma estratégia muito comum para detectar mudanças de conceito é monitorar alguns indicadores, como medidas de desempenho dos algoritmos de aprendizado e propriedades dos dados. Se uma mudança é detectada, algumas ações devem ser tomadas para adaptar o algoritmo para essas mudanças. Uma alternativa é utilizar janelas de tamanho adaptativo. Heurísticas podem ser usadas para ajustar automaticamente o tamanho da janela de acordo com a extensão da mudança (Klinkenberg e Renz, 1998). De maneira geral, se uma mudança é detectada, o tamanho da janela diminui, e, caso contrário, o tamanho aumenta. Exemplos de trabalhos que utilizam janelas de tamanho adaptativo são a família de algoritmos FLORA (Widmer e Kubat, 1996) e o método proposto por Klinkenberg e Joachims (2000), que utilizou máquinas de vetores de suporte (Cristianini e Shawe-Taylor, 2000) para reconhecer e tratar mudanças de conceitos. A ideia principal é selecionar um tamanho de janela em que o erro de generalização estimado para novos exemplos é minimizado.

Um dos algoritmos mais usados para detecção de mudanças é o *Page-Hinkley Test* (PHT) (Page, 1954), uma técnica de análise sequencial tipicamente usada para detectar mudanças em processamento de sinais (Gama, 2010). O PHT monitora a evolução da média de um sinal Gaussiano e utiliza um limiar constante para reportar a ocorrência de mudanças. O PHT considera uma variável cumulativa m_T, que é definida como a diferença entre os valores observados e a média desses valores até o momento atual, conforme

a Equação 2.10. A variável δ_{PHT} corresponde à magnitude das mudanças que são permitidas. Além disso, o teste PHT armazena o valor mínimo de m_t: $MIN_T = min(m_t)$, em que $t = \{1, \ldots, T\}$. Assim, o teste verifica a diferença entre MIN_T e m_t. Se essa diferença for maior do que um dado limiar α_{PHT}, o teste sinaliza que uma mudança foi detecta. O limiar α_{PHT} influencia diretamente na taxa de alarmes falsos positivos. Aumentando o valor de α_{PHT} o número de falsos positivos é reduzido, mas pode fazer com que o teste detecte um número menor de mudanças ou pode ocorrer atrasos na detecção. Nos experimentos realizados em Gama et al. (2009), o PHT foi usado juntamente com fatores de decaimento para monitorar o erro de um classificador a fim de detectar mudanças. Segundo os autores, o uso desse mecanismo de esquecimento acelerou as taxas de detecção de mudanças e manteve a capacidade de ser flexível a alarmes falsos quando não há mudanças.

$$m_T = \sum_{t=1}^{T} x_t - \bar{x}_T - \delta_{PHT} \tag{2.10}$$

$$\bar{x}_T = \frac{1}{T} \sum_{t=1}^{T} x_t \tag{2.11}$$

Um outro exemplo para detecção de mudanças é o método de Controle Estatístico de Processos (CEP), que realiza o monitoramento da variabilidade da qualidade de um processo. Em aprendizado de FCD, esse método pode ser utilizado para monitorar o desempenho do sistema de aprendizado com o objetivo de detectar mudanças (Gama et al., 2004; Gomes et al., 2011). São estabelecidos três estados para identificar o desempenho do aprendizado: "sob controle", "em alerta" e "fora de controle". Quando o erro do sistema é considerado estável, o estado é dito "sob controle". Quando o sistema de aprendizado atinge um erro maior do que um determinado limiar β_{CEP}, o estado muda para "em alerta". Esse estado significa que algo está errado no aprendizado, mas pode não ser por causa de uma mudança de conceito e sim por outros fatores, como ruído nos dados. Portanto, é necessário processar mais dados para tentar detectar o real motivo dessas variações. Se o erro continuar aumentando até ultrapassar o limiar α_{CEP}, em que $\alpha_{CEP} > \beta_{CEP}$, o estado muda para "fora de controle" e afirma-se, com uma determinada probabilidade, que uma mudança de conceito ocorreu recentemente ou que ainda está em processo, sendo necessário executar ações para o restabelecimento do estado "sob controle", como o descarte dos dados antigos e a atualização ou substituição do modelo usando apenas os dados mais recentes.

Kifer et al. (2004) propuseram um novo método para detecção e quantificação de mudanças na distribuição dos dados apenas com base nos atributos preditivos, ou seja, independentemente do algoritmo de aprendizado. Dado dois conjuntos de dados S_1 e S_2 que foram gerados por duas distribuições de probabilidade P_1 e P_2, a questão que os

autores tentam responder é: *é possível inferir a partir de S_1 e S_2 se eles foram gerados pela mesma distribuição, $P_1 = P_2$, ou se $P_1 \neq P_2$?* Dados reais raramente são bem-comportados, não seguindo uma distribuição paramétrica (Kifer et al., 2004). O meta-algoritmo proposto utiliza duas janelas: a janela de referência X contém os m primeiros elementos do fluxo de dados logo após uma mudança ter sido detectada e a janela Y é uma janela deslizante que contém os n elementos mais recentes do fluxo de dados. A janela Y desliza sempre que um novo elemento está disponível, e a cada atualização o meta-algoritmo checa se a distância entre as distribuições dos dois conjuntos de dados é maior do que um limiar α_{DT}, $dist(X, Y) > \alpha_{DT}$. Se a distância é maior, uma mudança de conceito é reportada e o processo é repetido, agora com X contendo os m primeiros elementos logo após a mudança. O ponto chave dessa abordagem é a escolha adequada da função de distância e da constante α_{DT}. Os autores desenvolveram uma medida, denominada discrepância relativa, que além de ser sensível à medição da discrepância entre distribuições possui garantias estatísticas que as medidas avaliadas são detectáveis a partir de amostras de tamanhos delimitados. Essa medida obteve melhores resultados do que os testes estatísticos Kolmogorov-Smirnov e Wilcoxon (Corder e Foreman, 2009) para algumas distribuições de dados gerados artificialmente. Outros métodos que mensuram a diferença entre dois conjuntos de dados com o objetivo de determinar se eles foram gerados a partir da mesma distribuição são o FOCUS (Ganti et al., 1999) e o PANDA (Bartolini et al., 2009), que permitem a descrição qualitativa e quantitativa das diferenças entre dois conjuntos de dados ou entre dois modelos derivados desses dados (Böttcher et al., 2008).

Tao e Ozsu (2009) analisaram diferentes problemas de FCD e descobriram que muitos deles apresentam periodicidade na distribuição dos dados, ou seja, uma distribuição que ocorreu no passado reaparece depois de um período de tempo. Com isso, os autores propuseram um *framework* para a mineração de FCD que possuem periodicidade nas mudanças entre diferentes distribuições. Esse *framework* inclui uma nova técnica para detecção de mudanças de distribuições baseada em Kifer et al. (2004). A maior diferença é que a janela de referência não contém os exemplos imediatamente seguintes após uma mudança, sob o argumento de que esses dados ainda podem ser instáveis, principalmente se a mudança ocorre gradualmente. Assim, a janela de referência contém os exemplos que chegam depois, quando a nova distribuição está, teoricamente, totalmente estabelecida. Além disso, o *framework* inclui uma nova técnica para identificar distribuições que já ocorreram anteriormente e reusar o conhecimento adquirido em minerações passadas. Resultados experimentais com dados sintéticos e um conjunto de dados real (Pacific Marine Environmental Laboratory, 2010) mostraram que essa técnica foi capaz de detectar mudanças de distribuição dos dados com maior precisão e sensibilidade do que as abordagens propostas por Aggarwal (2003) e Kifer et al. (2004). Para mostrar que o *framework* pode aumentar a eficiência do sistema de aprendizado quando há mudanças periódicas de distribuição nos dados, os autores utilizaram o *framework* proposto com o algoritmo

VFDT (Domingos e Hulten, 2000). Os resultados obtidos apontaram que houve uma redução de mais de 30% no tempo necessário para processar todos os exemplos quando os modelos que tinham sido aprendidos com os conceitos periódicos foram reusados.

Segundo Böttcher et al. (2008), o desafio não está apenas em adaptar os modelos para um ambiente de mudanças, mas também analisar quando e como eles mudam. Os autores propuseram um novo paradigma, denominado mineração de mudanças. O objetivo desse paradigma é a descoberta, a modelagem do monitoramento e a interpretação de mudanças em modelos que descrevem uma população que evolui. Os autores sugerem quatro tarefas genéricas que constituem um processo metodológico para mineração de mudanças: 1) Determinar os objetivos da mineração de mudanças; 2) Especificar um modelo em relação ao tempo; 3) Determinar os objetivos da mudança; 4) Projetar um mecanismo de monitoramento.

2.2.3 Algoritmos de aprendizado

Os algoritmos de aprendizado convencionais, denominados de algoritmos de aprendizado em lote, precisam retreinar um modelo desde o início sempre que um novo exemplo precisa ser incluído. Por outro lado, os algoritmos de aprendizado *online* e incrementais processam exemplos a medida que eles se tornam disponíveis, sem a necessidade de retreinar um modelo, e, portanto, possuem menor custo computacional (Laskov et al., 2006). Essas características os tornam mais adequados para o processamento de FCD, principalmente quando há uma alta taxa de fluxo. Segundo Gama et al. (2014), há uma diferença entre algoritmos *online* e incrementais. Quando os FCD são enormes e os dados não podem ser armazenados em memória principal, ou seja, são transitórios, apenas algoritmos online são adequados. Os algoritmos incrementais são menos restritivos, pois processam os exemplos um a um ou de lote em lote e atualizam o modelo após cada exemplo. Esses algoritmos podem ter acesso aleatório a exemplos passados ou a uma seleção ou representação desses exemplos. Esses algoritmos são chamados de algoritmos incrementais com memória parcial (Maloof e Michalski, 2004). Idealmente, os sistemas de KDD também deveriam funcionar continuamente e indefinidamente, incorporando exemplos assim que eles chegam e nunca perder informações potencialmente valiosas.

Domingos e Hulten (2000) desenvolveram um algoritmo de indução de árvores de decisão *online* para conjuntos de dados extremamente grandes (potencialmente infinitos). Esse algoritmo, denominado VFDT (do inglês, *Very Fast Decision Tree*) é um dos mais conhecidos em FCD e foi a base para muitos outros algoritmos. O VFDT requer que cada exemplo seja lido apenas uma vez e com tempo constante para processá-los, o que torna possível processar dados sem a necessidade de armazená-los. Os autores se basearam em Catlett (1991) para mostrar que o melhor atributo para testar em um nó da árvore pode

ser encontrado considerando apenas um pequeno subconjunto de exemplos de treinamento que passam por aquele nó. Assim, dado um fluxo de exemplos, os primeiros seriam usados para escolher o nó raiz e os seguintes seguiriam para os nós folha para que os atributos de teste fossem escolhidos e assim sucessivamente. Domingos e Hulten (2000) usaram um resultado estatístico, conhecido como limite de Hoeffding (Hoeffding, 1963), para decidir exatamente quantos exemplos são necessários em cada nó. O objetivo é garantir, com uma alta probabilidade, que o atributo teste escolhido com n exemplos (em que n é tão pequeno quanto possível) é o mesmo que seria escolhido usando um número infinito de exemplos. A propriedade chave dos algoritmos que usam o limite de Hoeffding é a garantia de que as árvores produzidas são assintóticas às construídas por um algoritmo de aprendizado em lote, ou seja, um algoritmo que usa todos os exemplos para escolher o atributo teste para cada nó (Domingos e Hulten, 2000). Em outras palavras, a natureza incremental do algoritmo não afeta significativamente a qualidade das árvores produzidas.

O VFDT permite usar as medidas de ganho de informação ou de índice de Gini em problemas de FCD para decidir qual atributo de teste deve ser usado para cada nó. Em uma comparação com dados sintéticos, o algoritmo C4.5 (Quinlan, 1993) obteve uma acurácia maior que o VFDT para os 25k primeiros exemplos. Entre 25k a 100k exemplos, os sistemas foram similares, e depois desse número de exemplos o VFDT melhorou sua acurácia para 88.7% contra 76.5% do C4.5. O número de nós encontrados pelo algoritmo VFDT também foi muito menor do que o C4.5 conforme o número de exemplos aumenta. Isso significa que as árvores induzidas pelo VFDT são mais compreensíveis e tendem a ser mais robustas em relação a dados com ruído do que o C4.5.

Uma extensão do algoritmo VFDT, chamado de CVFDT (*Concept-adapting Very Fast Decision Tree*), foi desenvolvida em Hulten et al. (2001) para possibilitar a utilização da base do VFDT em FCD com mudanças de conceito, pois esse algoritmo faz a suposição de que os dados de treinamento são amostras aleatórias de dados que originam de distribuições estacionárias. O CVFDT periodicamente percorre a árvore procurando por nós que antes tinham passado pelo teste de Hoeffding e que, atualmente, devido à mudança de conceito ou simplesmente por um erro na escolha do atributo, podem não passar mais, porque um outro atributo agora tem um ganho maior ou muito próximo (Hulten et al., 2001). Quando isso acontece, o CVFDT começa a criar uma subárvore alternativa com o atributo que agora possui o maior ganho de informação e que passa no teste de Hoeffding como a raiz dessa subárvore. A subárvore antiga é substituída pela alternativa quando essa última tiver maior acurácia do que a antiga (Hulten et al., 2001). Esse processo possibilita uma mudança suave, com finos ajustes quando a mudança de conceito ocorre, sem a necessidade de começar uma nova árvore a partir do início (Hulten et al., 2001).

Hulten et al. (2001) realizaram um estudo empírico para comparar o CVFDT com o VFDT. A primeira série de experimentos com dados artificiais comparou a capacidade dos algoritmos em lidar com grandes quantidades de dados com mudanças de conceito.

A cada 10 000 exemplos, a acurácia dos modelos era medida e a média desses resultados era reportada. Ao longo dos exemplos processados, pode-se observar que o CFVDT conseguiu responder rapidamente às mudanças de conceito, enquanto a taxa de erro do VFDT aumenta drasticamente antes de reagir à mudança. Além disso, o CVFDT produziu árvores consideravelmente menores do que o VFDT. Segundo Hulten et al. (2001) essa vantagem do CVFDT deriva do fato de que as árvores deste algoritmo são construídas com os 100 000 exemplos mais relevantes, enquanto o VFDT constrói suas árvores com milhares de exemplos desatualizados. Nos experimentos com um conjunto de dados real de requisições de páginas da internet, o CFVDT conseguiu uma acurácia média de 72.3%, enquanto o VFDT conseguiu 72.7%. A acurácia do CVFDT foi maior nos primeiros 70% exemplos, mas no final houve uma queda. De acordo com os autores, essa redução da acurácia no final do fluxo pode ter ocorrido porque o tamanho da janela era muito pequeno para aprender um modelo detalhado dos dados, ao contrário do VFDT, que usou todos os dados.

Segundo Law e Zaniolo (2005), as propriedades teóricas da abordagem utilizada pelo CVFDT são interessantes, mas o tempo necessário para atualização da árvore pode ser significativo, e um grande número de exemplos pode ser necessário para construir um classificador com eficácia razoável. Outra limitação do CVFDT é que ele não detecta automaticamente a ocorrência de mudanças de conceito. Por isso, o algoritmo precisa varrer a árvore periodicamente, o que aumenta ainda mais o custo computacional. Liu et al. (2009) acredita que as subárvores alternativas poderiam ser melhor aproveitadas no CVFDT, pois elas são apenas um conjunto de candidatas a substituírem subárvores da árvore principal. A substituição ocorrerá apenas se a acurácia das subárvores alternativas for maior do que a acurácia das subárvores atuais. Porém, durante esse período, as subárvores alternativas podem ser mais adequadas para fazer algumas predições e poderiam ser usadas para esse propósito. Além disso, se conceitos antigos ocorrem novamente após a substituição da subárvore antiga pela nova, é necessário reaprendê-los a partir do início (Liu et al., 2009).

Ikonomovska e Gama (2008) desenvolveram um outro algoritmo para aprendizado de modelos de árvores para problemas de regressão capazes de aprender em ambientes com FCD. O algoritmo, denominado FIMT (do inglês, *Fast and Incremental Model Tree*), assume que as distribuições são estacionárias e utiliza o limite de Chernoff (Hagerup e Rüb, 1990) como suporte estatístico para tomar uma decisão estável de quando dividir um nó. Em cada nó folha é utilizada uma rede neural do tipo perceptron, que realiza um ajuste fino no espaço de busca correspondente. O FIMT foi comparado com o algoritmo de aprendizado em lote CART (Breiman et al., 1984) e os resultados mostraram que o FIMT obteve os menores erros para grandes conjuntos de dados e desempenho semelhante ao CART em conjuntos de dados de tamanho médio.

Uma extensão do FIMT foi desenvolvido por Ikonomovska et al. (2009, 2011) para

incluir mecanismos de detecção de mudanças de conceitos. Esse mecanismo monitora a evolução do erro em cada região do espaço de busca e informa o algoritmo caso mudanças significativas possam ter afetado o aprendizado local ou global. Se a mudança é local, o algoritmo atualizará apenas as partes relacionadas com as regiões afetadas. A adaptação para a mudança é feita de maneira semelhante ao CVFDT, ou seja, uma árvore alternativa é induzida para substituir uma subárvore da árvore principal. O algoritmo proposto apresentou uma melhora significativa em comparação a outros métodos para conjuntos de dados artificiais e um conjunto de dados real.

Ensembles

Ensembles consistem na combinação das predições individuais de vários modelos para formar uma única predição final. Uma condição necessária e suficiente para que um *ensemble* de classificadores tenha uma maior acurácia do que um único modelo é que ele seja constituído por classificadores eficazes e diversos (Dietterich, 2000). Existem muitas abordagens para combinação de modelos (Mendes-Moreira et al., 2012; Kuncheva, 2004), como *boosting* (Freund e Schapire, 1996), *bagging* (Breiman, 1996), *stacked generalization* (Wolpert, 1992), entre outras. *Ensembles* têm sido empregados em problemas de FCD com sucesso, embora sejam computacionalmente mais custosos do que um único algoritmo, pois é necessário treinar vários modelos.

Segundo Wang et al. (2003), o problema fundamental em FCD com mudanças de conceitos é como identificar na hora certa os dados que não são mais consistentes com os conceitos atuais e, por isso, devem ser descartados. Um exemplo usado pelos autores ilustra bem o problema de seleção de dados. Dado que um FCD de duas dimensões é particionado em blocos, com base na ordem de chegada desses dados, e seja S_i os dados que chegam entre o tempo t_i e t_{i+1}, na Figura 2.7 é apresentado a distribuição dos dados e a fronteira de decisão ótima durante três intervalos de tempo. O problema é decidir no tempo t_3, após a chegada de S_2, quais dados influenciarão no modelo atual, para que os dados que chegarão depois de t_3 possam ser classificados com maior acurácia. Por um lado, apenas os dados mais recentes poderia ser escolhidos para compor o conjunto de treinamento, ou seja, apenas S_2 (S_0 e S_1 seriam descartados). Porém, como é possível observar na Figura 2.7, o modelo aprendido poderia ter uma alta variância, já que S_2 tem poucos dados e isso pode levar ao *overfitting* do modelo. Por outro lado, usar os blocos $S_1 \cup S_2$ pode diminuir a acurácia, como mostrado no primeiro bloco da Figura 2.8. Caso seja incluído ainda o bloco S_0, a acurácia não melhorará, como pode ser visto no bloco do meio da Figura 2.8. No último bloco da Figura 2.8 é mostrado que o conjunto de treinamento formado por $S_0 \cup S_2$ cria um classificador com menos *overfitting* e conflito entre conceitos. Porém, encontrar os dados de treinamento que obtém o classificador com a melhor acurácia não é trivial.

Assim, o objetivo em Wang et al. (2003) foi construir um *ensemble* sobre um conjunto

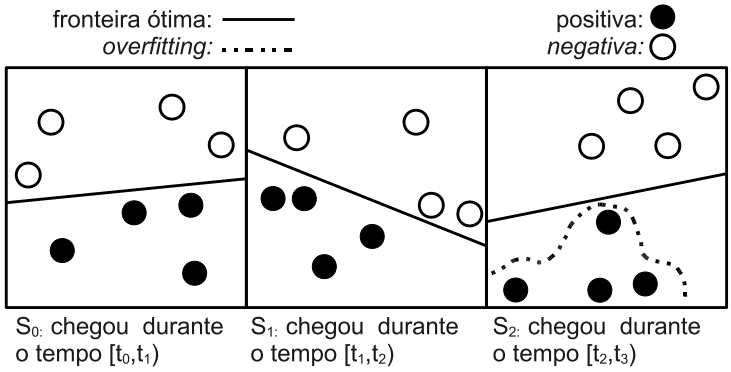


Figura 2.7: Distribuição dos dados e fronteiras de decisão ótimas. Adaptado de Wang et al. (2003).

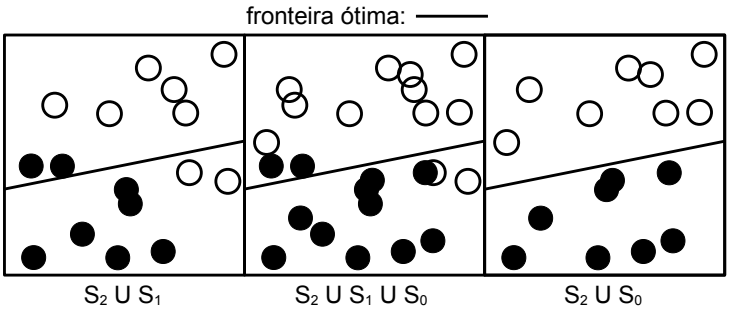


Figura 2.8: Escolha do conjunto de treinamento a ser usado. Adaptado de Wang et al. (2003).

de dados que produza um classificador melhor do que se todos os dados ou apenas o último bloco de dados forem utilizados. Os autores mostram que comparado com um único classificador, o qual é induzido por meio de todos os dados em uma janela com k lotes de exemplos, a abordagem por *ensemble* é capaz de reduzir o erro de classificação por meio de um esquema onde o peso de um classificador é inversamente proporcional ao erro esperado desse classificador. O peso do classificador é estimado pelo erro das predições feitas por ele para os exemplos de teste. Para validar o método proposto, Wang et al. (2003) realizaram experimentos com dados artificiais e reais de fraude de cartões de crédito. O *ensemble* é constituído por diferentes quantidades de modelos base dos algoritmos C4.5, RIPPER (Cohen, 1995) ou Naive Bayes (Murphy, 2012). O *ensemble* é comparado com um classificador que usa apenas os dados mais recentes, chamado de G_k, em que k é o número de lotes de exemplos em uma janela. Além disso, usa-se um classificador G_0, que é induzido com todos os dados, ou seja, os dados do início do fluxo até o momento. Segundo Wang et al. (2003), o algoritmo VFDT é um classificador G_0 e CVFDT é um classificador G_k. Porém, deve-se observar que o classificador G_k não possui mecanismos para tratamento de mudanças de conceito e outras técnicas implementadas no CVFDT. Os *ensembles* formados por classificadores Naive Bayes obtiveram as menores

taxas de erro para diferentes tamanhos de lotes de exemplos. Porém, a diferença entre um único classificador Naive Bayes e o *ensemble* é menor que 1.5% para três diferentes tamanhos de lotes, do total de quatro testados.

A falta de um critério mais rigoroso de quais dados antigos devem ser usados foi criticada por (Fan, 2004). O autor argumenta que os dados antigos podem contribuir com a acurácia do sistema de aprendizado desde que não tenha ocorrido mudanças de conceito. Portanto, a melhora da acurácia em alguns sistemas pode ser creditada à seleção aleatória de dados antigos. O autor também afirma que detectar mudanças de conceito e a quantidade de dados suficientes para treinar um modelo é difícil e não se pode quantificar. Como alternativa, ele propõe um algoritmo eficiente baseado em *ensemble* de árvores de decisão não correlacionadas que induzem modelos usando dados antigos indiscriminadamente: uma combinação de novos e antigos dados, apenas novos dados, modelos induzidos com dados antigos e atualizados com novos dados, etc. Cada árvore de decisão é construída a partir de atributos selecionados aleatoriamente dentre aqueles com maior ganho de informação. Para classificar um exemplo, calcula-se a média da saída (probabilidade condicional de cada classe) de cada árvore. O erro de classificação reportado é consideravelmente menor do que outras abordagens que usam dados antigos indiscriminadamente.

O *ensemble* proposto por Kolter e Maloof (2007), denominado de DWM (do inglês, *dynamic weighted majority*), mantém uma lista ponderada de modelos, que são atualizados sempre que novos dados estão disponíveis. Inicialmente, há um único modelo e o DMW adiciona ou remove modelos dessa lista com base no desempenho global do DWM. Sempre que um erro é cometido, o DWM adiciona um novo modelo com peso inicial igual a 1. Se o modelo comete um erro, o seu peso é reduzido. Ao longo do tempo, se um modelo possui um peso muito pequeno, ou seja, cometeu muitos erros, o DWM o remove do *ensemble*. O DWM usa a predição de cada modelo e o seu peso para calcular o peso ponderado de cada classe; aquela com o maior peso será a classe global predita. O método proposto pode ser utilizado, em princípio, com qualquer algoritmo de aprendizado incremental para induzir e atualizar os modelos da lista. Há também a possibilidade de diferentes algoritmos serem usados, mas, nesse caso, é necessário implementar políticas de controle para determinar quais algoritmos adicionar ao *ensemble*. A avaliação experimental do DWM foi realizada usando os algoritmos de aprendizado Naive Bayes (Witten e Frank, 2005) e uma árvore de decisão incremental (Utgoff et al., 1997) com conjuntos de dados artificiais e reais com mudanças de conceito. Os autores reportaram que o método proposto manteve um número de modelos similar a outros *ensembles*, mas obteve maiores acurácias preditivas e convergiu mais rapidamente para esses valores.

2.2.4 Avaliação

Em contraste com o crescente número de algoritmos para aprendizado de FCD, métodos de avaliação experimental e métricas para estimar o desempenho dos modelos aprendidos ainda é um assunto que não está bem estabelecido. Um dos primeiros trabalhos publicados com esse enfoque foi o de Gama et al. (2009). Posteriormente, esse trabalho foi estendido em Gama et al. (2013) e um outro artigo foi publicado também no mesmo ano por Bifet et al. (2013). As discussões e ideias desta Seção são, portanto, baseadas nesses trabalhos.

Segundo Gama et al. (2009, 2013), as principais dificuldades para avaliar algoritmos de aprendizado para FCD são:

- um fluxo de dados ao invés de um número fixo de exemplos;
- modelos de decisão que evoluem com o tempo ao invés de modelos estáticos;
- os dados são gerados por uma distribuição não-estacionária ao invés de uma amostra estacionária.

Em Gama et al. (2009, 2013) são abordados assuntos referentes à avaliação de algoritmos de aprendizado para FCD tentando responder a seguinte pergunta: *uma estratégia de amostragem é viável para o cenário de FCD?* A amostragem é sugerida por (Dietterich, 1998) como uma maneira de avaliar algoritmos de aprendizado quando uma grande quantidade de dados está disponível. Entretanto, a resposta encontrada para essa pergunta é negativa, pois uma análise sequencial é mais indicada nesse caso. A análise sequencial se refere a teorias e métodos estatísticos em que o tamanho da amostra não é fixado a priori, ou seja, os dados são avaliados assim que se tornam disponíveis (Ghosh e Sen, 1991). Como em domínios de FCD o fluxo é potencialmente infinito e a distribuição que gera os exemplos evolui ao longo do tempo, assim como os modelos utilizados nesse domínio, as estratégias de amostragem mais usadas em AM, como a validação cruzada, não são aplicáveis em FCD.

As alternativas viáveis apresentadas na literatura, segundo Gama et al. (2009, 2013) são *holdout* e *prequential* (do inglês, *predictive sequential*) (Dawid e Vovk, 1999). Apesar da importância em avaliar o tempo de processamento e a memória utilizada pelos algoritmos, os autores focam apenas na avaliação do poder de generalização dos modelos. No método *holdout* o modelo atual é aplicado a um conjunto de dados de teste em intervalos de tempo (ou exemplos) regulares. No método *prequential*, também conhecido como testar-então-treinar[1] o erro de um modelo é computado de uma sequência de exemplos, em que cada exemplo é usado para testar e, posteriormente, quando o rótulo verdadeiro estiver disponível, é usado para treinar o modelo. Assim, o modelo é sempre testado

[1]Tradução própria. Em inglês, esse termo é referido como *test-then-train*.

com exemplos que não tinham sido usados anteriormente. O erro é computado como a soma acumulada de uma função de perda entre a predição e os valores observados. Esse método de avaliação resulta em uma curva da evolução do aprendizado similar à obtida com o método *holdout*, quando o modelo atual é aplicado a conjuntos de dados em intervalos de tempo regulares. Os autores também realizaram experimentos para estudar as propriedades de convergência do método *prequential* usando janelas deslizantes. Os resultados mostraram que, apesar dos diferentes tamanhos de janela, o erro *prequential* convergiu rapidamente para a estimativa do erro usando *holdout*. Em outro experimento, as janelas deslizantes foram substituídas por fatores de decaimento e o comportamento foi semelhante àquele com janelas.

Para comparar o desempenho de dois algoritmos em FCD, Gama et al. (2009, 2013) sugerem o uso de uma estatística que pode ser usada com quase qualquer função de perda. Sejam A e B dois algoritmos e S_i^A e S_i^B a sequência da perda acumulada para cada algoritmo, respectivamente, a estatística Q_i é definida pela Equação 2.12. O sinal de Q_i indica o desempenho relativo dos dois modelos e o seu valor indica a força das diferenças entre eles. Um experimento comparando duas topologias diferentes de redes neurais artificiais usando dados reais mostrou que o sinal de Q_i se manteve sempre negativo, indicando que uma rede neural foi superior a outra para todos os exemplos apresentados.

$$Q_i(A, B) = log\left(\frac{S_i^A}{S_i^B}\right) \tag{2.12}$$

Bifet et al. (2013) compararam a acurácia de 16 algoritmos adaptativos com um classificador simples, denominado *No-Change*, que usa a classe do último exemplo rotulado do conjunto de treinamento para predizer a classe do exemplo de teste. Nessa comparação, que utilizou um problema real de FCD de classificação, apenas 6 dos 16 classificadores testados tiveram acurácia maior do que o classificador *No-Change*. Esse resultado é surpreendente, visto que esse classificador ignora completamente os atributos dos dados e sua predição é baseada apenas na classe do último exemplo. Os autores concluíram que esses resultados ocorreram porque os dados não são independentemente distribuídos. Portanto, é necessário adaptar os métodos de avaliação, que supõem a independência entre os exemplos. Para isso, Bifet et al. (2013) propõem uma nova medida denominada *Kappa Plus* ou simplesmente κ^+, que subtrai a acurácia do classificador proposto p_0 da acurácia do classificador *No-Change* p_e, conforme a Equação 2.13.

$$\kappa^+ = \frac{p_0 - p'_e}{1 - p'_e} \tag{2.13}$$

Testes estatísticos para FCD

Testes estatísticos para algoritmos em FCD ainda é um assunto em aberto. Alguns autores, como Bifet (2009), argumentam que devido ao grande número de exemplos de

teste usados para computar o desempenho dos algoritmos, todas as diferenças podem ser consideradas estatisticamente significativas.

Um dos fatores que devem ser considerados para a escolha do teste estatístico é o tempo de processamento desses teste, principalmente em FCD, em que esse fator é ainda mais crítico (Dietterich, 1998). Para algoritmos que podem ser executados uma única vez, Dietterich (1998) afirma que o teste de McNemar é interessante. Além de possuir um baixo erro do Tipo I (dificilmente detecta uma diferença significativa quando não há), esse teste é incremental e é menos poderoso do que o teste de validação cruzada 5 X 2. Isso significa que se o teste de McNemar for capaz de detectar diferenças significativas, outros testes, como o de validação cruzadas 5 X 2, provavelmente, apontariam uma diferença ainda maior.

Para aplicar o teste de McNemar, é necessário apenas computar o número de exemplos classificados incorretamente pelo algoritmo A e não pelo B ($n_{0,1}$) e os exemplos classificados incorretamente pelo algoritmo B e não pelo A ($n_{1,0}$). Esses valores podem ser atualizados rapidamente, o que é uma propriedade desejável em mineração de FCD de alta velocidade. Esse teste pode ser mais preciso se uma correção para continuidade é utilizada. A correção é necessária porque um distribuição contínua (chi-quadrado) é usado para aproximar uma distribuição discreta (Siegel e Jr., 1988). A estatística da Equação 2.14 tem uma distribuição χ^2 com um grau de liberdade. Para um nível de confiança de 95%, a hipótese nula é rejeitada se a estatística é maior do que 3.841. Nos experimentos realizados em Gama et al. (2009), esse teste obteve diferentes resultados sobre a significância estatística quando diferentes fatores de decaimento ou tamanhos de janela deslizante foram usados. Isso acontece porque quanto maior o número de exemplos considerados, maiores são as chances de que pequenas diferenças sejam significativas.

$$M = sign(n_{0,1} - n_{1,0}) \frac{|(n_{0,1} - n_{1,0}| - 1)^2}{n_{0,1} + n_{1,0}} \tag{2.14}$$

2.3 Considerações finais

Os principais desafios em mineração de fluxos de dados incluem amostragem, aprendizado incremental e esquecimento, viés e gerenciamento do custo computacional e desempenho preditivo (Gama e Rodrigues, 2009). Em problemas em que ocorrem mudanças de conceitos, um dos principais desafios é como incorporar mecanismos de detecção de mudanças em algoritmos de aprendizado para diferentes paradigmas. Outro aspecto relevante são os critérios de avaliação das hipóteses e as métricas a serem utilizadas. A maioria dos métodos e métricas de avaliação foram desenvolvidos para casos estáticos e fornecem uma medida simples da qualidade das hipóteses, enquanto que no contexto de FCD, é mais interessante saber como a avaliação das métricas evolui ao longo do tempo (Gama e Rodrigues, 2009). Como a área de mineração de FCD é relativamente nova, as práticas de

avaliação não estão bem estabelecidas como nas pesquisas com algoritmos de aprendizado convencionais e a maioria dos experimentos usam menos do que um milhão de exemplos. Para Bifet (2009), essa prática não fornece evidências suficientes de que os sistemas de aprendizado avaliados sob tais condições serão capazes de lidar com problemas que geram dados abundantemente. Outros desafios para a mineração de FCD podem ser encontrados em Žliobaite et al. (2012) e referências ali contidas.

Capítulo 3

Meta-aprendizado

A utilização de técnicas de Aprendizado de Máquina (AM) (Mitchell, 1997) e Mineração de Dados (MD) (Han e Kamber, 2006) tem se expandido além dos laboratórios. Para que tais técnicas possam corresponder às necessidades dos usuários, elas devem ser criteriosamente selecionadas. De acordo com o teorema *No Free Lunch* (Wolpert, 1996), qualquer desempenho positivo apresentado por um algoritmo em um dado contexto de aprendizado é compensado por um desempenho negativo de mesmo grau quando de sua aplicação em um outro contexto. Assim, se todos os cenários forem equiprováveis, então os algoritmos apresentarão, em média, desempenho preditivo semelhante. Portanto, a definição de que algoritmo empregar para cada problema deve ser realizada de maneira específica.

Tal escolha pode ser feita, por exemplo, com o auxílio de um especialista do domínio do problema ou por meio de um processo de investigação empírica baseado em tentativa e erro. Nenhuma das soluções é totalmente satisfatória para o usuário final, que espera poder ter acesso a essas ferramentas com baixo custo e de forma efetiva (Giraud-Carrier et al., 2004). Sendo assim, usualmente, a análise de dados é realizada de maneira *ad hoc*, considerando apenas a disponibilidade dos algoritmos. Nesse caso, espera-se que o conjunto de suposições embutidas em um desses algoritmos funcione para as características dos dados que estão sendo estudados. Entretanto, em muitas aplicações reais, essa expectativa pode não ser atendida, especialmente em casos nos quais a distribuição dos dados pode mudar com o tempo. Assim, essa abordagem simples pode ser insatisfatória.

Uma alternativa para auxiliar o usuário na tarefa de selecionar os algoritmos mais apropriados para o seu problema são os sistemas de recomendação de algoritmos (Brazdil et al., 2009). Eles operam primeiro avaliando os dados em estudo e depois sugerindo os algoritmos de AM mais adequados para a realização das análises devidas. Com isso, pretende-se reduzir o número de algoritmos testados, de modo a minimizar o tempo de experimentação, com perda reduzida na qualidade dos resultados obtidos.

Uma possibilidade para desenvolver os sistemas de recomendação de algoritmos é por meio da utilização de conceitos de meta-aprendizado (Pratt e Thrun, 1997; Thrun e Pratt,

1998; Vilalta e Drissi, 2002; Giraud-Carrier et al., 2004). Segundo Rendell et al. (1987), o termo meta-aprendizado designa o aprendizado por meio da experiência quando diferentes vieses podem ser utilizados para tratar um problema. O seu objetivo, portanto, é descobrir como buscar dinamicamente a melhor estratégica de aprendizado conforme o número de tarefas aumenta (Thrun e Pratt, 1998). Dessa maneira, meta-aprendizado explora o conhecimento acumulado sobre diversos problemas e aplica esse conhecimento para a resolução de tarefas similares (Giraud-Carrier et al., 2004). Na prática, métodos de meta-aprendizado funcionam como os algoritmos de AM convencionais: adaptando seus parâmetros a um ambiente (Mitchell, 1997). Entretanto, ao contrário daqueles, que trabalham sobre um conjunto de dados por vez, o aprendizado no nível meta é baseada em várias aplicações de um algoritmo de AM para diferentes conjuntos de dados. Com isso, é possível entender sob quais condições cada algoritmo é mais apropriado, possibilitando sugestões de uso mais adequadas.

Para que sistemas de recomendação baseados em meta-aprendizado possam ser bem aplicados, quatro critérios básicos devem ser observados quando de seu desenvolvimento (Kalousis, 2002): i) caracterização do domínio; ii) medidas de avaliação; iii) formas de sugestão e; iv) métodos de construção de sugestão. Neste capítulo, uma sucinta explanação sobre cada um desses tópicos é fornecida.

O restante deste capítulo é organizado como segue. Na Seção 3.1, o problema de seleção de algoritmos é definido e sua relação com meta-aprendizado é discutida. Na Seção 3.2, as abordagens comumente empregadas para a caracterização dos conjuntos de dados são apresentadas. Na Seção 3.3, as medidas de desempenho mais utilizadas para avaliar sistemas de meta-aprendizado são comentadas. Na Seção 3.4, as formas comumente usadas para a apresentação da sugestão ao usuário são discutidas. Na Seção 3.5, são descritas alternativas para a construção de sugestões. Na Seção 3.6, são apresentados os trabalhos de meta-aprendizado para FCD de que se tem conhecimento. Por fim, na Seção 3.7, são feitas algumas considerações finais sobre este capítulo.

3.1 Recomendação de algoritmos de AM

A recomendação de algoritmos de AM apropriados a uma determinada tarefa pode ser entendida como um problema de busca. O espaço de soluções para esse problema consiste dos algoritmos e o objetivo da busca é identificar um ou mais algoritmos que melhor se adequam às análises devidas (Brazdil et al., 2009). Métodos de meta-aprendizado lidam com isso construindo regras capazes de relacionar o desempenho dos algoritmos às propriedades dos problemas. Assim, é possível fornecer sugestões de maneira sistemática ao usuário. Embora a utilização de meta-aprendizado para essa tarefa seja recente (vide referências em (Smith-Miles, 2008)), a necessidade de relacionar o viés indutivo de cada algoritmo às propriedades dos dados é conhecida desde o trabalho seminal de Rice

(1976). Segundo Smith-Miles (2008) os seguintes pré-requisitos são necessários para tratar o problema de seleção de algoritmos usando uma abordagem de meta-aprendizado:

- disponibilidade de grandes coleções de instâncias de problemas de várias complexidades: espaço de problemas P;
- existência de um grande número de algoritmos diversos para tratar as instâncias dos problemas: espaço dos algoritmos A;
- métricas de desempenho para avaliar os algoritmos: espaço de desempenho Y;
- existência de características apropriadas para descrever as propriedades dos dados: espaço de características F.

Dado esses requisitos, o problema de seleção de algoritmos pode ser formalizado, de acordo com Smith-Miles (2008), como segue:

> Para uma instância de um problema $x \in P$, com características $f(x) \in F$, encontre o mapeamento da seleção $S(f(x))$ no espaço de algoritmos A, em que o algoritmo selecionado $\alpha \in A$ maximize o mapeamento de desempenho $y(\alpha(x)) \in Y$.

Associar as características dos problemas às métricas de desempenho dos algoritmos por meio de um grande número de instâncias resolvidas por diferentes algoritmos cria um conjunto abrangente de meta-conhecimento sobre desempenho de algoritmos. Segundo Brazdil et al. (2009), meta-conhecimento é o conhecimento extraído do processo de aprendizado. Tal extração pode ser feita automaticamente com o uso de algoritmos de AM. Essa estratégia originou o termo aprender a aprender, empregado muitas vezes como sinônimo de meta-aprendizado (Thrun e Pratt, 1998), e é a base para o processo de recomendação de algoritmos proposto em Brazdil et al. (2009) e ilustrado na Figura 3.1.

O processo tem início com a construção do repositório P, que armazena conjuntos de dados referentes a instâncias de problemas diversos. Cada um desses problemas é investigado quanto a suas propriedades e a caracterização dos mesmos é realizada segunda as métricas em F. Aos problemas em P também são aplicados os algoritmos em A e o desempenho preditivo destes é avaliado. As informações sobre a caracterização dos dados (ou meta-atributos preditivos) e o desempenho dos algoritmos (ou meta-atributo alvo) são agrupadas em meta-exemplos, um para cada problema. Coletivamente, os meta-exemplos são chamados de meta-dados. Esse conjunto é, então, utilizado por um algoritmo de aprendizado, chamado de meta-aprendiz, para induzir um modelo, conhecido como meta-modelo, que deve ser capaz de relacionar os meta-atributos preditivos ao meta-atributo alvo, gerando a relação S da definição de seleção de algoritmos. Por meio desse esquema, é possível sugerir, por fim, os algoritmos mais adequados para um problema desconhecido.

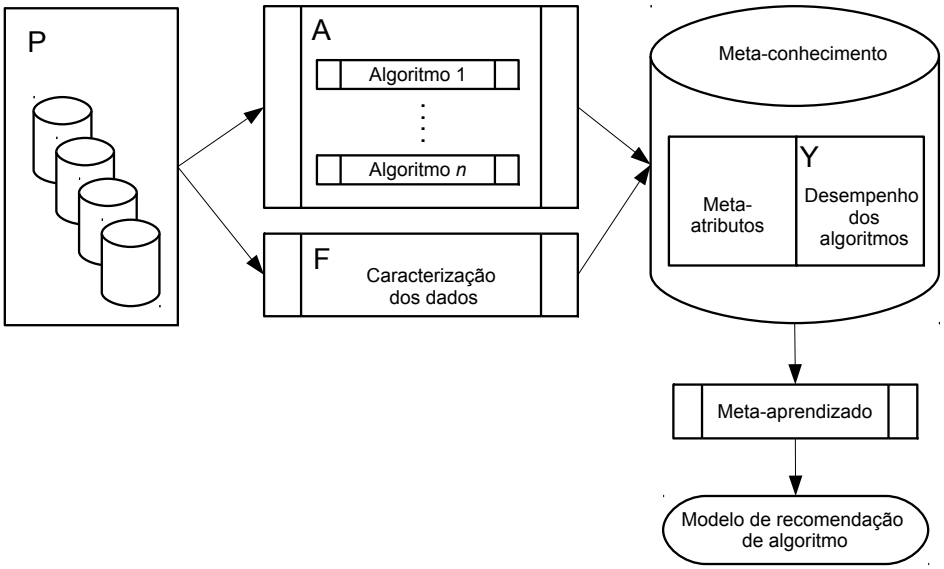


Figura 3.1: Processo de recomendação de algoritmos usando meta-aprendizado. Adaptado de Brazdil et al. (2009)

A seguir, os principais aspectos necessários para a utilização de meta-aprendizado para a recomendação de algoritmos são discutidos.

3.2 Caracterização do domínio

Todo sistema de meta-aprendizado precisa extrair as características relevantes sobre a tarefa ou o domínio que está sendo estudado. Essas características, que são usadas como meta-atributos de entrada, devem enfatizar os aspectos importantes do domínio, provendo conhecimento útil capaz de diferenciar o desempenho preditivo de um dado conjunto de algoritmos de AM. Dessa forma, a qualidade das recomendações realizadas é dependente da caracterização dos dados e, por isso, esse componente é essencial para o sucesso dos métodos baseados em meta-aprendizado.

Apesar da sua importância, ainda poucos trabalhos têm focado em entender a conexão entre as características de um domínio em análise e o desempenho preditivos de algoritmos de AM (Brazdil et al., 2009). O projeto STATLOG (Michie et al., 1994) foi o primeiro grande projeto a tentar encontrar essa relação. Mais recentemente, o projeto METAL (em inglês, *A Meta-Learning Assistant for Providing User Support in Machine Learning and Data Mining*) pesquisou o desenvolvimento de ferramentas baseadas em meta-aprendizado para auxiliar usuários na seleção de algoritmos de AM e MD.

Segundo Vilalta et al. (2004), atualmente há três abordagen para realizar a caracterização dos dados: i) direta; ii) via modelos e; iii) baseada em *landmarking*; . A seguir, cada uma delas é brevemente descrita.

3.2.1 Caracterização direta

Os meta-atributos dessa abordagem podem ser obtidos por três grupos de medidas: simples, estatísticas e baseadas na teoria da informação. As medidas simples incluem informações gerais sobre os conjuntos de dados, como número de atributos, número de exemplos, taxa de exemplos por atributos, ganho de informação, etc. As medidas estatísticas são aplicadas sobre atributos numéricos para calcular grandezas estatísticas, como assimetria, curtose, variância, etc. Por fim, as medidas baseadas na teoria da informação, como entropia e a informação mútua, são empregadas para caracterizar os atributos nominais (Brazdil et al., 2009). As medidas de caracterização direta mais comumente empregadas são apresentadas na Tabela 3.1. Elas foram utilizadas nos projetos STATLOG e METAL. Aquelas informadas em itálico foram consideradas apenas no METAL. Desenvolvimentos posteriores importantes podem ser encontrados em (Sohn, 1999; Lindner e Studer, 1999; Kalousis, 2002; Soares, 2004).

Tabela 3.1: Medidas para caracterização de dados utilizadas nos projetos STATLOG e METAL. As medidas em itálico foram consideradas apenas no METAL.

Tipo	**Descrição**
Simples	Número de exemplos
	Número de atributos
	Número de classes
	Número de atributos binários
	Número de atributos nominais
	Número de atributos numéricos
Estatísticas	Razão média entre desvio padrão dos atributos
	Correlação média absoluta entre atributos, por classe
	Primeira correlação canônica
	Proporção de variância explicada pelo 1o discriminante canônico
	Assimetria média absoluta dos atributos
	Curtose média dos atributos
	Número de atributos com outliers
	Estatística M de Box
	Graus de liberdade da Estatística M
	Valor de Lambda de Wilk
	Estatística V de Barlett
Informação	Entropia normalizada das classes
	Entropia média dos atributos
	Informação mútua média entre classe e atributos
	Razão sinal/ruído
	Entropia conjunta de classe e atributos

Dependendo do domínio de aplicação de métodos de meta-aprendizado, medidas específicas podem ser empregadas para extrair informação dos dados. Por exemplo, Kanda et al. (2010) usam características específicas para descrever diversos problemas do caixeiro viajante (Applegate et al., 2007) com o objetivo de selecionar a melhor técnica de otimização para instâncias específicas desses problemas. Algumas das características utilizadas

nesse estudo são: número de vértices, número de arestas e valor da maior e da menor aresta. Medidas específicas também têm sido propostas para o domínio de análise de séries temporais com o propósito final de selecionar a técnica mais adequada para cada série, como em Adya et al. (2001), Prudêncio e Ludermir (2004), Wang et al. (2009) e Lemke e Gabrys (2010). Alguns exemplos de propriedades que são obtidas de séries temporais pelas medidas específicas propostas nesses estudos são: autocorrelação, sazonalidade e tendência da série.

3.2.2 Caracterização via modelos

Nessa abordagem, utilizam-se propriedades do modelo induzido por uma algoritmo de AM para realizar a caracterização dos dados (Bensusan, 1998; Bensusan et al., 2000; Peng et al., 2002). De acordo com Vilalta et al. (2005), há duas vantagens importantes nessa perspectiva. A primeira é que o conjunto de dados é comprimido em uma estrutura que contém informações sobre a complexidade e o desempenho do modelo. Sendo assim, a caracterização não está restrita à distribuição dos exemplos. A segunda consiste no fato de que a representação dos dados nessa forma pode servir de base para explicar o desempenho do algoritmo. Dentre os algoritmos de AM, as árvores de decisão são as mais frequentemente usadas na indução dos modelos, basicamente porque apresentam comportamento determinístico e geram uma estrutura que permite a compreensão das propriedades dos dados analisados (Bensusan et al., 2000; Peng et al., 2002). Algumas características que podem ser obtidas a partir das árvores geradas são: o número de nós, a profundidade máxima da árvore e o grau de balanceamento.

3.2.3 *Landmarking*

Nessa abordagem, informação acerca do desempenho de algoritmos de AM quando aplicados a conjuntos de dados é empregada para produzir meta-atributos relevantes (Bensusan e Giraud-Carrier, 2000a; Pfahringer et al., 2000). O pressuposto utilizado é que cada algoritmo atua de maneira satisfatória em uma área de competência específica, definida pelas propriedades dos dados. Assim, pela utilização de algoritmos simples mas com vieses indutivos bastante diversos, conhecidos como *landmarkers*, seria possível extrair informação importante acerca dos problemas em estudo. Dessa maneira, um conjunto de dados poderia ser descrito pela coleção de áreas de competência às quais ela pertence. Nesse contexto, o *landmarking* é utilizado para determinar a proximidade de um conjunto de dados em relação a outros, por meio da similaridade de desempenho dos *landmarkers*. Com isso, espera-se que conjuntos de dados de natureza semelhante compartilhem as mesmas áreas de competência e possam ser analisados eficazmente pelos mesmos algoritmos de AM.

3.3 Medidas de avaliação

Em um processo de recomendação de algoritmos por meta-aprendizado, é necessário estabelecer uma métrica de desempenho adequada para avaliar cada técnica candidata. Em um estudo abrangente, Caruana e Niculescu-Mizil (2006) compararam várias abordagens de AM considerando oito critérios de desempenho, divididos nos seguintes grupos: baseados em limiar, baseados em ordenação e baseados em probabilidade. Exemplos representativos das três categorias incluem a acurácia, a área sob as curvas ROC e o erro quadrático médio, respectivamente. Como ponderação relevante, os autores ressaltaram que dependendo da métrica utilizada, um aspecto ou outro do comportamento do algoritmo de AM é ressaltado, sendo, portanto, interessante o emprego de múltiplas medidas.

Alinhadas a esse perspectiva, alguns trabalhos em meta-aprendizado têm proposto a utilização de metodologias de avaliação multiobjetivas, em que duas ou mais medidas são combinadas. Elas não focam necessariamente apenas no comportamento preditivo dos algoritmos, já que, em muitas aplicações, pode ser interessantes mensurar outros aspectos, como, o tempo requerido para o algoritmo de AM construir um classificador, o tempo requerido para o classificador rotular um exemplo, a quantidade de memória requerida pelo algoritmo e a simplicidade e interpretabilidade dos classificadores construídos, entre outras. No caso de (Soares e Brazdil, 2000), por exemplo, considerou-se uma medida chamada *Adjusted Ratio of Ratios*, que combina acurácia e tempo de execução. Assim, o usuário tem a opção de considerar em suas análises um algoritmo mais eficiente computacionalmente ao custo de um discreto decréscimo de desempenho.

Apesar da maior flexibilidade de soluções envolvendo várias medidas ao mesmo tempo, a utilização de apenas um critério de avaliação ainda é a prática mais comum em meta-aprendizado.

3.4 Formas de sugestão

Em um sistema de recomendação de algoritmos de AM, a forma como a sugestão é fornecida deve ser determinada de acordo com as necessidades do usuário. Segundo Kalousis (2002), há três formas predominantes de apresentação dos resultados: i) melhor algoritmo; ii) subconjunto de algoritmos e; iii) *ranking* de algoritmos. A seguir, cada uma delas é brevemente comentada.

Na primeira abordagem, o sistema indica, dentre os algoritmos disponíveis, apenas o mais promissor, segundo algum critério. Nesse caso, o problema de meta-aprendizado é visto como uma tarefa de classificação, possivelmente multiclasses, e apresenta a vantagem de poder ser tratado empregando diversos algoritmos de classificação ordinários de AM (Brazdil et al., 2009). A desvantagem de focar em uma única opção, é que se esta mostrar-se inadequada a uma determinada situação, o usuário não dispõe de auxílio para

selecionar outro algoritmo.

Na segunda abordagem, esse cenário indesejável é evitado fornecendo ao usuário mais de uma sugestão. Assim, inclui-se na recomendação outros algoritmos potencialmente adequados. Para estabelecer a composição desse conjunto, pode-se considerar o valor esperado de desempenho de cada candidato e compará-lo com o do algoritmo mais promissor. Para tanto, Brazdil et al. (2009) citam duas possibilidades. A primeira, é definir uma margem de desempenho em torno do melhor caso e todos os algoritmos com desempenho estimado dentro dessa margem serão incluídos na sugestão. A segunda possibilidade é utilizar testes estatísticos para aferir a similaridade preditiva entre os algoritmos e considerar aqueles que não são significantemente diferentes do melhor algoritmo (Kalousis, 2002). Embora apresentar múltiplos algoritmos como sugestão seja conveniente, o fato deles não estarem dispostos em uma ordem pode dificultar sua utilização.

Na terceira abordagem, os algoritmos são fornecidos em ordem de preferência com relação a um determinado problema. O critério de ordenação pode ser, por exemplo, uma estimativa de desempenho dos algoritmos disponíveis, conforme discutido na Seção 3.3. Assim, o usuário tem a seu dispor uma quantidade maior de informação para fazer bom uso das sugestões fornecidas, já que ele pode simplesmente seguir a ordem de algoritmos sugerida.

3.5 Construção de sugestão

Uma vez definida como a recomendação será fornecida ao usuário, deve-se decidir a abordagem adequada para sua construção. Basicamente, tal escolha depende do tipo de meta-atributo alvo considerado. Nesse contexto, três abordagens de meta-aprendizado são comumente estudadas: i) meta-aprendizado por classificação; ii) meta-aprendizado por regressão e; iii) meta-aprendizado por *rankings*.

A primeira abordagem é a mais comum, tendo sido empregada desde o projeto STATLOG. Nela, o fenômeno de interesse corresponde a classes, como os nomes dos algoritmos que se quer sugerir ou então se um dado algoritmo é aplicável ou não a um determinado problema. Assim, um algoritmo de classificação é utilizado para mapear os meta-atributos de entrada ao meta-atributo de saída. Exemplos de meta-aprendizado por classificação podem ser encontrados em Kalousis (2002) e Brazdil et al. (2009).

No lugar de classes discretas, pode-se tentar predizer diretamente o desempenho dos algoritmos de AM e, posteriormente, ordenar tais predições para prover a sugestão para um novo problema. Esse é o fundamento de meta-aprendizado por regressão, que, para realizar as predições, utiliza algoritmos de regressão para induzir os chamados meta-regressores. Com isso, a tarefa de recomendação é dividida em n subproblemas de predição de desempenho, um para cada algoritmo candidato. O uso de meta-regressão pode ser encontrada, por exemplo, em Köpf et al. (2000) e Prudêncio et al. (2008).

Em meta-aprendizado por rankings, uma função de preferência sobre um conjunto de itens é aprendida. De maneira simples, essa tarefa pode ser realizada por meio do algoritmo k-vizinhos mais próximos (k-NN, do inglês, *k-nearest neighbor*) (Aha et al., 1991), que foi adaptado com sucesso para fornecer rankings como saída em Soares e Brazdil (2000). Nessa técnica, o aprendizado consiste simplesmente no armazenamento dos exemplos de treinamento e a predição do ranking para um exemplo de teste qualquer ocorre agregando-se os rankings de seus k vizinhos mais próximos.

3.6 Meta-aprendizado para FCD

Em métodos convencionais de aprendizado, como citado nas seções anteriores, a estratégia para obtenção do meta-conhecimento pode se dar de diferentes maneiras para vários conjuntos de dados. Para grandes conjuntos de dados, uma possibilidade é utilizar aprendizado ativo (Weiss et al., 2004), em que os exemplos são processados em lotes e o modelo é construído utilizando-se o primeiro lote. Após a criação desse modelo, é realizada a seleção dos exemplos informativos dos próximos lotes e os demais exemplos são descartados (Brazdil et al., 2009). Trabalhos com dados possivelmente infinitos, como fluxos contínuos de dados (FCD), necessitam de um mecanismo de controle que possibilita a seleção de diferentes tipos de sistemas de aprendizado conforme mais dados se tornam disponíveis. O meta-conhecimento que é obtido dos dados que chegam a todo momento pode ser utilizado para determinar se o sistema deve continuar utilizando o modelo atual, se deve substituí-lo por outro modelo ou apenas estendê-lo (Brazdil et al., 2009).

O trabalho de Widmer (1997) é um dos pioneiros a empregar meta-aprendizado em problemas de FCD. Nesse trabalho, assume-se que os conceitos dos dados dependem do contexto atual, como na previsão do clima, que muda radicalmente de acordo com a estação do ano (Tsymbal, 2004). Conceitos de meta-aprendizado foram usados para desenvolver dois métodos que são capazes de detectar indícios de mudanças nesses contextos a partir dos atributos preditivos dos dados. Esses métodos consistem, basicamente, de um algoritmo que realiza o aprendizado *on-line* no nível-base e um meta-modelo que procura identificar atributos e valores que podem prover sinais de contexto. Os dois métodos, denominados Metal(B) e Metal(IB) diferem em relação ao algoritmo de aprendizado utilizado no nível-base e na maneira com que as informações sobre os sinais de contexto são usadas. Enquanto o método Metal(B) utiliza um classificador bayesiano, o Metal(IB) emprega um classificador baseado em instâncias (Aha et al., 1991). No primeiro, os dados de entrada que provêm sinais de contexto são identificados via meta-aprendizado e são sinalizados como relevantes para o classificador bayesiano. Quando um novo exemplo chega, o classificador utiliza como base para predição apenas os exemplos de treinamento que possuem o mesmo contexto do novo exemplo. Quando não há exemplos do mesmo contexto, todos aqueles que pertencem à janela deslizante atual são usados para treinamento. No

método Metal(IB), há variações que consistem basicamente em utilizar pesos para cada um dos exemplos ou atributos pertencentes à janela (de tamanho fixo) que está sendo utilizada como base para predição, de acordo com os contextos identificados. Os resultados experimentais com dados reais e artificiais mostraram que o uso do meta-aprendizado proporcionou uma rápida adaptação a mudanças e, geralmente, um aumento da acurácia em domínios que mudam dinamicamente. A identificação de contextos, segundo Widmer (1997), é o nível em que o poder do meta-aprendizado se torna aparente. O autor ainda menciona que há uma conexão interessante entre o modelo de aprendizado desenvolvido nesse trabalho e as noções de transferência de aprendizado e algoritmos de aprendizado com vida-longa, como proposto por Thrun e Mitchell (1995).

Conceitos de meta-aprendizado também foram utilizados por Prodromidis et al. (2000) para tratar problemas de mineração de fluxos de dados distribuídos. Em um ambiente com vários conjuntos de dados interconectados, o objetivo era prover meios para que cada local que possui um conjunto utilize seus próprios dados e, ao mesmo tempo, se beneficie dos dados disponíveis em outros locais, sem que seja necessário transferir ou acessar diretamente todos eles. Prodromidis et al. (2000) realizaram um estudo de caso para predizer fraude em transações com cartões de crédito usando meta-aprendizado para combinar e integrar classificadores induzidos separadamente com dados de diferentes instituições financeiras. A ideia consistia em gerar classificadores locais que, posteriormente, poderiam ser transferidos entre os demais locais (instituições), eliminando a necessidade de transferência dos dados. Os resultados experimentais sugeriram que o meta-aprendizado, junto com outras técnicas e métodos, constitui uma abordagem efetiva e escalável para mineração de conjuntos de dados distribuídos.

Prudêncio e Ludermir (2004), propuseram duas diferentes abordagens de meta-aprendizado para seleção de modelos para previsão de séries temporais, que é uma área que possui semelhanças com FCD. Essas abordagens foram testadas com dois estudos de caso. Primeiramente, foi utilizado um algoritmo de AM para selecionar entre dois modelos para previsão de séries temporais estacionárias. Posteriormente, foi utilizado o sistema NOEMON (Kalousis e Theoharis, 1999) para ordenar, em termos de desempenho, três modelos usados para previsão de séries temporais da *M3-Competition* (Makridakis e Hibon, 2000). O NOEMON é um sistema baseado em meta-aprendizado que provê um ranking de modelos combinando diferentes algoritmos de AM. Os experimentos realizados para os dois estudos de caso revelaram resultados encorajadores em termos de acurácia na tarefa de seleção dos modelos e no desempenho preditivo dos modelos selecionados. Segundo os autores, uma questão importante para o sucesso de abordagens similares é a definição de um conjunto de características adequadas para o domínio de séries temporais. Trabalhos posteriores, como os de Wang et al. (2009) e Lemke e Gabrys (2010) se concentraram nessa questão.

Klinkenberg (2005) investigou duas abordagens para tratar o problema de mudanças

de conceito em FCD. A primeira abordagem mantém um janela deslizante adaptativa automática sobre o conjunto de dados de treinamento (Klinkenberg e Joachims, 2000) e seleciona apenas os exemplos mais adequados ou associa um peso para cada exemplo. Além dos exemplos, o algoritmo base (e os valores para seus parâmetros) que será empregado para o aprendizado desses dados é selecionado de maneira exaustiva, testando vários algoritmos a partir de uma lista pré-definida. A ideia chave é encontrar o conjunto de dados e o algoritmo de aprendizado que minimizam o erro de generalização estimado para esses exemplos. Essa abordagem superou outros métodos estáticos, que ignoram a mudança de conceito, em experimentos com diferentes tipos de simulações de mudanças de conceito em dados reais de texto.

A segunda abordagem proposta por Klinkenberg (2005) é baseada em meta-aprendizado e permite selecionar automaticamente um algoritmo de aprendizado base (e seus parâmetros) a partir de um conjunto de algoritmos (e seus parâmetros) definido a priori. A abordagem proposta nesse trabalho não assume a existência de variáveis que indicam contexto, como em Widmer (1997), e utiliza informações sobre a seleção dos dados realizada pela primeira abordagem e do desempenho preditivo dos modelos para selecionar um algoritmo. Os exemplos supostamente chegam aos lotes e a abordagem utiliza um algoritmo de aprendizado para induzir um meta-modelo, que prediz o algoritmo base mais apropriado para os dados do lote mais recente. Cada meta-exemplo usado na indução do meta-modelo é composto por características que descrevem o processo de aprendizado do nível base, como o número de lotes de exemplos usado para treinamento, o algoritmo que obteve o melhor desempenho preditivo para o último lote e o melhor algoritmo considerando todos os lotes já processados. Como os resultados desses experimentos foram publicados em um relatório técnico da Universidade de Dortmund, na Alemanha, e não está disponível pela internet, não foi possível verificar o desempenho do método proposto.

Gama e Kosina (2009, 2011) também desenvolveram um método baseado em meta-aprendizado para mineração de fluxos de dados com mudanças de conceito. Esse método é capaz de detectar a recorrência de um conceito e recuperar modelos de decisão aprendidos anteriormente para esse mesmo conceito. Assim, quando uma mudança ocorre, não é necessário induzir um modelo para os dados mais recentes a partir do início se o conceito já ocorreu para dados passados. O método desenvolvido pode selecionar modelos históricos apropriados para os dados mais novos, caso haja um, sem o conhecimento da classe verdadeira desses dados.

Esse método é baseado em duas camadas de aprendizado, base e meta, sendo que cada camada treina seu próprio modelo e recebe seus próprios dados. A primeira camada utiliza os exemplos rotulados para treinar um classificador que, em seguida, é aplicado para predizer o rótulo da classe de cada novo exemplo que chega. Os exemplos dos quais se conhece a classe verdadeira são usados para calcular a perda entre a classe predita e a classe real e atualizar o classificador atual. O mesmo exemplo usado para treinar

o classificador na primeira camada é usado para treinar um meta-modelo na segunda camada, sendo que a única diferença é o atributo alvo, que ao invés de indicar a classe real, indica se o classificador do nível base classificou o exemplo corretamente ou não. Assim, cada exemplo possui os mesmos valores para os atributos preditivos nos dois níveis de aprendizado, com exceção do valor do atributo alvo. Com isso, o meta-aprendiz aprende a região do espaço dos exemplos onde o classificador tem um bom desempenho. Sempre que uma mudança de conceito é detectada, o meta-modelo busca na lista de modelos já utilizados se um deles é apropriado para o conceito dos novos dados. Se esse modelo existir, ou seja, se o conceito dos dados é recorrente, ele é recuperado e utilizado para classificar os novos dados; caso contrário, ou seja, se o conceito dos dados é inédito, um novo modelo é induzido. Nos experimentos realizados foi usado o classificador Naive Bayes e apesar de não ter ocorrido uma grande melhora na acurácia geral, o método foi capaz de detectar a recorrência de conceitos e reusar modelos aprendidos no passado.

3.7 Considerações finais

Neste capítulo, foram apresentados os principais conceitos de meta-aprendizado, com ênfase para aqueles relacionados à recomendação de algoritmos. A seleção de algoritmos por métodos baseados em meta-aprendizado consiste, basicamente, no mapeamento entre as características que descrevem propriedades relevantes dos dados e o desempenho preditivo dos algoritmos. Nesse contexto, foram abordadas as principais questões envolvidas no desenvolvimento de sistemas de recomendação de algoritmos de AM, o que inclui a caracterização dos dados, as medidas de avaliação, as formas de sugestão e a construção de sugestão.

Métodos convencionais baseados em meta-aprendizado selecionam um algoritmo uma única vez para cada conjunto de dados a partir do pressuposto de que o algoritmo mais apropriado para uma amostra suficiente de dados para um determinado problema também o será para novas instâncias. Essa prática é suportada pela hipótese de que esses dados são gerados por uma função de distribuição estacionária. Entretanto, dados que são produzidos continuamente em ambientes dinâmicos estão propensos a mudanças e, consequente, o algoritmo que era inicialmente adequado pode se tornar obsoleto ao longo do tempo. Na seção anterior foram apresentados alguns trabalhos que usaram meta-aprendizado para a seleção de algoritmos ou modelos nesse cenário. Esses trabalhos serviram como base para o método proposto nesta tese, que é apresentado no capítulo seguinte.

Capítulo 4

MetaStream

Neste capítulo, é apresentado o método proposto nesta tese de doutorado, denominado **MetaStream**, e as abordagens para caracterização de fluxos de dados. O MetaStream é um método baseado em meta-aprendizado para seleção de algoritmos em ambientes dinâmicos de fluxos contínuos de dados com o objetivo de melhorar o desempenho preditivo do sistema de aprendizado. A partir de um conjunto pré-definido de algoritmos, o MetaStream prediz qual deles é o mais adequado para os dados que estão chegando, de acordo com as suas características. O processo de seleção de algoritmos pelo MetaStream pode ser dividido em dois níveis de aprendizado: nível base e nível meta. No nível base, diferentes algoritmos usam os dados rotulados mais recentes para induzir um modelo, o qual pode ser aplicado para predizer o atributo alvo dos novos exemplos. No nível meta, um algoritmo de AM, conhecido como meta-aprendiz, é utilizado para relacionar as características dos dados do nível base com o desempenho dos modelos para esses dados. O meta-modelo produzido pelo meta-aprendiz deve ser capaz de generalizar para novos casos, ou seja, deve ser capaz de selecionar o algoritmo mais apropriado para os dados que estão chegando de acordo com suas características. Como podem ocorrer mudanças quase que arbitrariamente ao longo do tempo, é preciso que os modelos nos dois níveis estejam constantemente atualizados. Isso pode implicar na sua adaptação ou reconstrução periódica, sempre utilizando os dados mais recentes.

A seleção de algoritmos pode ser promissora em problemas de fluxo de dados, principalmente quando os desempenhos médios dos algoritmos são similares para uma grande quantidade de dados, mas diferentes quando analisados sob a perspectiva de uma granularidade menor, isto é, considerando subconjuntos menores desse fluxo. Se for possível relacionar as propriedades que podem ser obtidas a partir dos dados com as variações de desempenho preditivo dos modelos em cada um desses subconjuntos, então é possível predizer o melhor algoritmo para cada instante de tempo e, assim, melhorar o desempenho geral do sistema de aprendizado.

O restante deste capítulo está organizado como segue. Na Seção 4.1, é apresentado o processo de indução e avaliação de algoritmos de aprendizado no nível base. A seguir, na

Seção 4.2, é descrito o funcionamento do método MetaStream no nível meta, dividido nas três principais etapas de um sistema de meta-aprendizado: a geração dos meta-dados, a indução de um meta-modelo e a aplicação do meta-modelo para selecionar o algoritmo mais promissor para o nível base. Na Seção 4.3, é apresentado um estudo sobre quais dados e medidas podem ser utilizadas na caracterização de fluxos de dados. Na Seção 4.4, é apresentada a abordagem para a seleção de algoritmos para cada exemplo do nível base, assim como as diferenças de caracterização dos dados para essa abordagem. Na Seção 4.5, discute-se brevemente a questão da complexidade dos métodos investigados. Por último, na Seção 4.6, são feitas algumas considerações finais sobre este capítulo e o método proposto.

4.1 Nível base

No nível base, m algoritmos de aprendizado $L_1, L_2, \ldots L_m$ recebem exemplos de um fluxo contínuo de dados do problema sob análise. Cada exemplo $z = (\mathbf{x}, y)$ consiste de uma tupla formada por p valores de atributos preditivos $\mathbf{x} = (x_1, \ldots, x_p)$ e um valor de atributo alvo y, em que y é categórico para problemas de classificação ou numérico para problemas de regressão. Esses exemplos chegam em lotes ou um a cada instante de tempo e são utilizados pelos algoritmos na indução e avaliação de modelos por meio do método testar-então-treinar intercalado com uma janela deslizante baseada em sequência, como ilustrado na Figura 4.1 e explicado em seguida. No estado atual do fluxo de dados mostrado nessa figura, a janela deslizante contém os ω_b exemplos de treinamento, de índices $[i-\omega_b+1, \ldots, i]$, que são os exemplos mais recentes dos quais se conhece os valores reais do atributo alvo. Cada algoritmo de aprendizado L_j, $j = \{1, \ldots, m\}$, utiliza esses dados para induzir ou atualizar um modelo M_j, que, em seguida, prediz o valor do atributo alvo para os exemplos de índices $[i + \eta_b + 1, \ldots, i + \eta_b + \lambda_b]$, em que $\eta_b \geq 0$ representa o horizonte de predição (em número de exemplos) e $\lambda_b \geq 1$ é o número de exemplos de teste. Os exemplos de índices $[i+1, \ldots, i+\eta_b]$ já foram preditos anteriormente e, portanto, seus atributos preditivos são conhecidos no instante de tempo atual, mas os valores reais do atributo alvo ainda são desconhecidos devido ao atraso que pode ocorrer entre fazer uma predição e observar o valor real de saída. Esse atraso, como pode ser notado, depende do horizonte de predição, pois quanto mais próximo for o horizonte de predição, menor será o atraso. A janela desliza a cada λ_b exemplos, o que é conhecido como passo da janela.

Algoritmos incrementais ou em lote podem ser utilizados nesse nível de aprendizado, desde que sejam adequados para o fluxo de dados sob análise e eficientes o suficiente para lidar com a velocidade de chegada dos exemplos. Se forem utilizados algoritmos incrementais, cada modelo é, normalmente, induzido uma única vez e atualizado ao longo do fluxo de dados a cada λ_b exemplos. Caso contrário, se forem utilizados algoritmos de aprendizado em lote, um novo modelo é induzido por cada algoritmo a cada λ_b exemplos

utilizando os ω_b exemplos de treinamento. Portanto, além dos algoritmos, o tamanho da janela de treinamento ω_b e o número de exemplos de teste λ_b também interferem diretamente nos requisitos computacionais necessários. Quanto maior o valor de ω_b e menor o valor de λ_b, mais eficientes precisam ser os algoritmos, pois maior será o número de exemplos de treinamento e o modelo será induzido mais frequentemente.

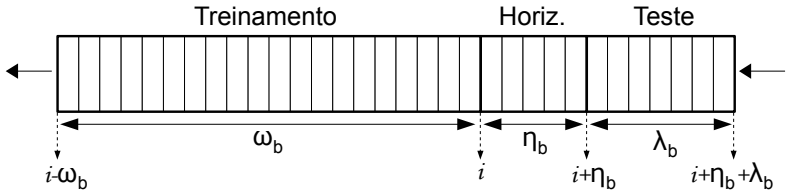


Figura 4.1: Fluxo de dados no nível base usando uma janela deslizante. Os ω_b exemplos de treinamento são utilizados por algoritmos de aprendizado para a indução de modelos, que são empregados para predizer os valores do atributo alvo dos λ_b exemplos de teste.

4.2 Nível meta

No nível meta, o método MetaStream pode ser descrito em três etapas principais, assim como a maioria dos métodos baseados em meta-aprendizado: i) a geração dos meta-dados; ii) a indução de um meta-modelo; iii) a aplicação do meta-modelo para a seleção de algoritmos. Essas etapas podem ser visualizadas no diagrama da Figura 4.2. Nesse diagrama, o processamento do fluxo de dados que chega no nível base (parte superior esquerda da figura) foi considerado com uma janela deslizante de $\omega_b = 100$ exemplos, um conjunto de teste de $\lambda_b = 10$ exemplos e nenhum atraso para observar o valor real do atributo alvo, $\eta_b = 0$. A janela deslizante atual contém os exemplos de 501 a 600, indicando que os primeiros 500 exemplos já foram processados e descartados. Quando $\eta_b = 0$, significa que o valor real do atributo alvo do exemplo de teste atual é observado antes da predição do atributo alvo de um exemplo do próximo conjunto de teste. O restante do diagrama é abordado a seguir, na descrição das etapas do nível meta.

4.2.1 Geração dos meta-dados

Em um cenário típico de meta-aprendizado, os meta-dados são extraídos de diferentes conjuntos de dados, em que cada conjunto resulta em um único meta-exemplo. Como esses dados são supostamente estacionários, esse processo é realizado para um conjunto de dados com número de exemplos fixo uma única vez. No contexto de fluxo de dados, é necessário que os meta-dados sejam gerados ou atualizados continuamente, pois os dados não são estacionários, criando-se um fluxo de meta-dados. As características extraídas dos dados do nível base devem conter informação que descreva com precisão o fenômeno em cada momento. Para os objetivos deste trabalho, é importante que essas características

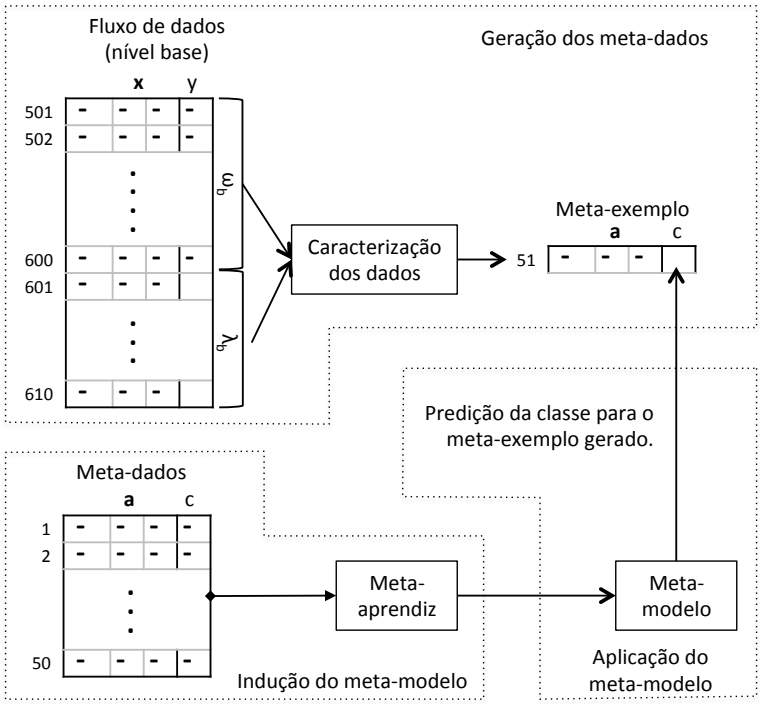


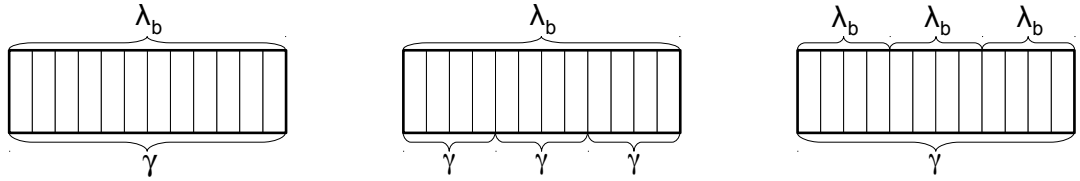
Figura 4.2: Ilustração do processo de geração de um meta-exemplo de teste a partir dos dados do nível base e a predição de sua classe usando o meta-modelo induzido a partir dos meta-dados disponíveis. Cada meta-exemplo usado na indução do meta-modelo (ou seja, cada linha nos meta-dados) é gerado durante a etapa de geração dos meta-dados, adicionando depois o valor do meta-atributo alvo. Esse valor identifica o algoritmo que teve o melhor desempenho ao nível base no conjunto de dados respectivo.

contenham informação que reflita o desempenho dos algoritmos. Como consequência das mudanças que ocorrem nos dados, as características também devem variar com o tempo, em particular, quando o desempenho dos algoritmos varia significativamente. A taxa de mudança dessas características depende das mudanças ocorridas nos dados, do passo da janela deslizante no nível base λ_b e de alguns parâmetros do nível meta, que serão discutidos mais adiante nesta seção.

No método MetaStream, um meta-exemplo $e = (\mathbf{a}, c)$ é uma tupla de q valores de meta-atributos preditivos $\mathbf{a} = (a_1, \ldots, a_q)$ e um valor de meta-atributo alvo c, que indica o melhor algoritmo para os dados de teste caracterizados por esse meta-exemplo. Os atributos preditivos são obtidos a partir da extração de características dos exemplos do nível base e do próprio processo de aprendizado no nível meta (Seção 4.3). O valor do meta-atributo alvo c só será conhecido posteriormente, depois que os valores do atributo alvo y dos respectivos exemplos de teste do nível base forem observados, como é descrito a seguir nesta seção.

Um meta-exemplo é criado para cada γ exemplos de teste do nível base, em que γ representa o número de exemplos para os quais se deseja selecionar um algoritmo. Esse conjunto de exemplos é doravante denominado de conjunto de seleção de algoritmos, ou

simplesmente, **conjunto de seleção**. O tamanho desse conjunto pode ser igual ao do conjunto de teste do nível base, λ_b, o que significa que um algoritmo é selecionado para cada conjunto de teste, como na Figura 4.3(a), mas também pode ser menor ou maior que λ_b, como mostrado nas Figuras 4.3(b) e 4.3(c), respectivamente. Por questão de simplicidade, neste trabalho, assume-se que γ é divisor de λ_b, se $\gamma < \lambda_b$, ou múltiplo de λ_b, se $\gamma > \lambda_b$. Se $\gamma < \lambda_b$, um algoritmo diferente pode ser selecionado para cada subconjunto dos exemplos de teste. Por exemplo, se $\lambda_b = 12$ e $\gamma = 4$ (Fig. 4.3(b)), o conjunto de teste é dividido em três conjuntos de seleção, com γ exemplos cada. Assim, um algoritmo diferente pode ser selecionado para cada subconjunto. Por outro lado, se $\gamma > \lambda_b$, um único algoritmo será selecionado para vários conjuntos de teste do nível base, pois supõe-se que o melhor algoritmo será o mesmo para todos esses dados. Por exemplo, se $\lambda_b = 4$ e $\gamma = 12$, como na Figura 4.3(c), então, um único meta-exemplo será criado a partir de três conjuntos de teste, que constituem o conjunto de seleção, e um único algoritmo será selecionado para esse conjunto. Em todos os casos, os dados de treinamento e horizonte de predição são os mesmos que foram utilizados em cada instante de tempo no nível base, independentemente do valor de γ.



(a) $\gamma = \lambda_b$ (b) $\gamma < \lambda_b$ (c) $\gamma > \lambda_b$

Figura 4.3: Na figura à esquerda, o número de exemplos do conjunto de seleção é igual ao do conjunto de teste, $\gamma = \lambda_b$. Na figura central, $\gamma < \lambda_b$ e o conjunto de teste é dividido, formando vários conjuntos de seleção de algoritmos. Na figura à direita, $\gamma > \lambda_b$ e os vários conjuntos de teste são unidos, formando um único conjunto de seleção.

O tamanho do conjunto de dados de seleção γ é um fator importante no processo de caracterização e seleção de algoritmos em fluxos de dados. O método MetaStream não possui nenhuma restrição em relação à quantidade de exemplos que chega a cada instante de tempo, como acontece, por exemplo, em (Klinkenberg, 2005), mas é necessário definir previamente se o método será utilizado para selecionar algoritmos para lotes de exemplos ($\gamma > 1$) ou para cada exemplo individual ($\gamma = 1$). Quando o método MetaStream é utilizado para a seleção de algoritmos para lotes de exemplos, o valor de γ pode variar ao longo do tempo. Por exemplo, o método MetaStream pode ser aplicado para selecionar um algoritmo para os dados que são produzidos diariamente por uma aplicação que gera um número variável de dados por dia. Entretanto, quando $\gamma = 1$, o método MetaStream sempre selecionará um algoritmo para cada exemplo do nível base e não pode variar ao longo do tempo. Essa restrição se deve principalmente ao processo de caracterização de dados, que será discutido detalhadamente na Seção 4.3, e da abordagem para a seleção

de um algoritmo para cada exemplo, discutida na Seção 4.4.

Na parte superior do diagrama da Figura 4.2, é exibido o processo de criação de um meta-exemplo não rotulado a partir da caracterização dos dados do nível base utilizando $\gamma = \lambda_b$. Como $\gamma = \lambda_b = 10$, um meta-exemplo será criado para cada conjunto de teste do nível base.

Posteriormente, quando os valores do atributo alvo de todos os exemplos base da janela de seleção forem observados, o meta-exemplo atual é rotulado de acordo com o desempenho preditivo dos modelos para esses exemplos, produzindo $e = (\mathbf{a}, c)$, conforme ilustrado na Fig. 4.4. Dependendo da estratégia de rotulação e do desempenho preditivo dos modelos para os γ exemplos de teste do nível base, o rótulo c do meta-exemplo pode indicar, por exemplo: i) um único algoritmo; ii) uma combinação de algoritmos; iii) ranking de algoritmos. As estratégias de rotulação investigadas nesta tese são descritas detalhadamente na Seção 5.3.2, mas outras estratégias poderiam ser utilizadas.

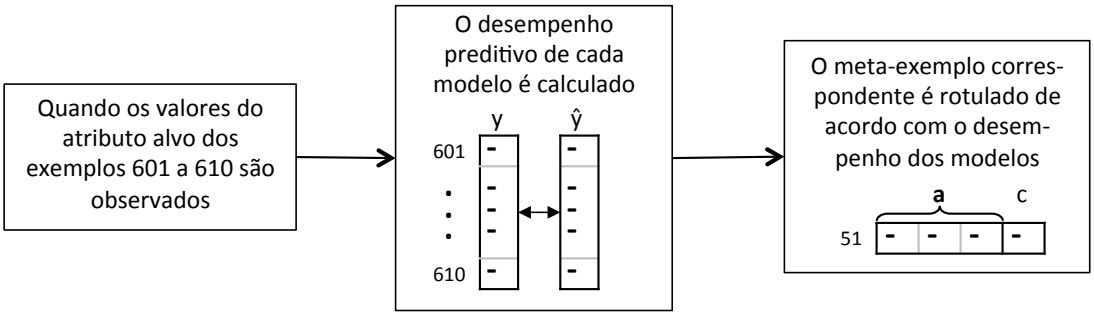


Figura 4.4: Rotulação do meta-exemplo 51 de acordo com o desempenho dos algoritmos para os exemplos do nível base do conjunto de seleção, de índices 601 a 610. O rótulo do meta-exemplo depende da estratégia de rotulação utilizada e pode indicar, por exemplo, o melhor algoritmo, uma combinação de algoritmos ou um ranking de algoritmos.

A qualidade das características extraídas dos exemplos do nível base depende da qualidade desses dados e da quantidade de exemplos sendo caracterizados. Quanto menor o número de exemplos, maior pode ser a variância das características extraídas. Por exemplo, se um regressor realiza uma predição discrepante para um único exemplo que pertence a um pequeno conjunto de seleção, isso pode fazer com que a qualidade das características baseadas no desempenho preditivo desse algoritmo não seja uma boa estimativa para os próximos conjuntos de seleção. Por outro lado, quanto maior o valor de γ, possivelmente, maior será a qualidade das características para o conjunto de seleção. Entretanto, a diferença de desempenho preditivo entre os algoritmos de aprendizado tende a diminuir conforme o tamanho do conjunto de seleção aumenta, embora eles possam ser claramente distintos quando analisados para pequenos subconjuntos desses dados.

Duas questões importantes ainda precisam ser respondidas sobre a caracterização dos dados: i) quais são os dados disponíveis que podem ser caracterizados a cada instante de tempo? ii) quais medidas são capazes de extrair informações relevantes para a predição

do melhor algoritmo? Em relação à primeira questão, é possível, por exemplo, extrair informações sobre a relação entre os atributos preditivos e o atributo alvo dos dados da janela de treinamento, pois os valores do atributo alvo já foram observados. Por outro lado, essas informações não estão disponíveis para os dados de seleção. A segunda questão refere-se a qual medida será utilizada para mensurar essa relação entre os atributos. Essas questões são discutidas na Seção 4.3.

4.2.2 Indução do meta-modelo

Depois da geração de um número mínimo suficiente de meta-exemplos rotulados, de acordo com a descrição da seção anterior, pode-se prosseguir para a etapa de indução do meta-modelo. Essa etapa está ilustrada na parte inferior esquerda da Figura 4.2. O método MetaStream trata a questão da seleção de algoritmos como um problema de aprendizado supervisionado. Assim, os meta-dados gerados anteriormente são utilizados por um algoritmo de aprendizado para mapear as características que descrevem os dados e o processo de aprendizado com o desempenho dos modelos induzidos no nível base, resultando em um meta-modelo. Dependendo de como o problema de seleção de algoritmos é tratado no nível meta, o algoritmo de aprendizado utilizado para induzir esse meta-modelo pode ser de classificação, regressão ou sugestão de rankings (Capítulo 3). Nos experimentos realizados nesta tese, a seleção de algoritmos é tratada como um problema de classificação (Seção 5.3).

Para induzir ou atualizar o meta-modelo ao longo do tempo sempre com os dados mais recentes, usa-se o método testar-então-treinar intercalado com um uma janela deslizante de tamanho ω_m, idêntico ao mecanismo utilizado para induzir os modelos no nível base. Assim, o tamanho mínimo de meta-exemplos supracitado refere-se ao tamanho da janela de treinamento ω_m.

Observando novamente a Figura 4.2, é possível visualizar 50 meta-exemplos no conjunto de meta-dados (parte inferior esquerda). Considerando uma janela de treinamento no nível meta de $\omega_m = 50$, todos os meta-exemplos que já estão rotulados são utilizados na indução do meta-modelo. O número de meta-exemplos gerados é definido pelo número de exemplos já processados pelo nível base e pelos valores de parâmetros utilizados nessa ilustração. Assim, no exemplo da figura, dos 600 exemplos dos quais já se conhece os valores do atributo alvo, os primeiros 100 foram utilizados apenas para treinamento, enquanto os outros 500 exemplos foram utilizados para teste e depois para treinamento. Como o tamanho do conjunto de seleção $\gamma = 10$, um meta-exemplo é gerado para cada 10 exemplos do nível base, resultando nos 50 meta-exemplos gerados até o momento ilustrado na figura.

4.2.3 Seleção de algoritmos de aprendizado

Nesta etapa, o meta-modelo induzido na etapa anterior é empregado para predizer o rótulo do meta-atributo alvo do próximo meta-exemplo para o qual é preciso selecionar um algoritmo. É importante ressaltar que o rótulo predito representa o algoritmo (ou algoritmos) que será utilizado no nível base, o qual induzirá o modelo que predirá os valores do atributo alvo para os exemplos de teste que originaram o respectivo meta-exemplo. Na Figura 4.2, o meta-modelo prediz o rótulo do meta-exemplo 51. O algoritmo predito será utilizado para induzir um modelo com os ω_b exemplos base da janela de treinamento. Em seguida, esse modelo será aplicado para predizer o atributo alvo dos exemplos de teste 601 a 610.

4.3 Meta-atributos

A caracterização dos dados é uma etapa crucial dos sistemas de meta-aprendizado. Assim, as medidas utilizadas para esse fim devem ser capazes de extrair dos dados e do processo de aprendizado características que influenciam o desempenho dos modelos no nível base (Kalousis, 2002). Uma das principais limitações das abordagens convencionais de meta-aprendizado é o fato de que a maioria dessas medidas são calculadas sobre todos os atributos e depois precisam ser sumarizadas em um único valor, pois diferentes conjuntos de dados possuem diferentes morfologias e, consequentemente, geram diferentes características. A morfologia de um conjunto de dados refere-se aos atributos que descrevem esses dados. Dois conjuntos de dados são morfologicamente diferentes se eles são caracterizados por atributos diferentes. Essa diferença pode ser em relação à quantidade, ao tipo (numérico, nominal, categórico, etc) ou à semântica dos atributos. Portanto, mesmo que todos os conjuntos de dados a serem caracterizados sejam descritos pela mesma quantidade e tipo de atributos, a semântica dos atributos devem ser iguais, para que as características extraídas de cada atributo ou das relações entre eles possam ser usadas diretamente como meta-atributos, sem a necessidade de agregação. A imposição de ter que descrever um conjunto de dados usando os mesmos atributos é uma limitação dos algoritmos de aprendizado proposicionais (Raedt, 2008). Um exemplo de medida que extrai diferentes características dependendo da morfologia dos dados é a correlação entre atributos numéricos, que é calculada para cada par de atributos. Quanto maior o número de atributos numéricos, maior o número de características geradas. Assim, para que diferentes conjuntos de dados sejam descritos pelas mesmas características, as correlações entre todos os pares de atributos numéricos precisam ser expressas por um único valor, o que pode ser feito, por exemplo, calculando-se a média de todas as correlações.

Em fluxos de dados, o objetivo é selecionar o melhor algoritmo para o mesmo problema em diferentes instantes de tempo. Como esse problema é descrito pelo mesmo conjunto

de atributos ao longo do tempo, é possível aplicar algoritmos proposicionais e gerar meta-modelos com base em características para cada atributo ou para cada relação entre os atributos. Assim, a correlação entre cada par de atributos, por exemplo, pode gerar um meta-atributo, sem a necessidade de agregação.

Medidas que são comumente usadas em estudos de meta-aprendizado para extrair características de diferentes conjuntos de dados, como em Kalousis (2002), Kuba et al. (2002), Soares (2004) e Amasyali e Ersoy (2009), também podem ser empregadas neste estudo para a caracterização de fluxos de dados. Para isso, um mecanismo de esquecimento, como as janelas deslizantes, é usado para realizar a seleção dos dados que serão caracterizados a cada instante de tempo. Geralmente, há uma sobreposição dos dados em janelas subsequentes, o que depende do número de exemplos antigos que são descartados para a adição de novos exemplos. Consequentemente, se os dados mudam ao longo do tempo, as características tendem a mudar gradualmente juntamente com os dados.

Para aplicações que geram FCD, a ordem de chegada dos exemplos pode interferir no desempenho preditivo dos modelos. Portanto, características que consideram a ordem dos dados podem ser relevantes para a seleção de algoritmos para fluxos de dados. Tais características não podem ser obtidas com as medidas utilizadas nos estudos supracitados, que consideram que os dados são independentes e identicamente distribuídos, mas podem ser extraídas por medidas comumente usadas na caracterização de séries temporais, como a correlação serial e outras medidas apresentadas em Adya et al. (2001), Prudêncio e Ludermir (2004), Wang et al. (2009) e Lemke e Gabrys (2010). A dependência entre exemplos sucessivos, geralmente, não é tão forte em fluxos de dados como na análise de séries temporais, principalmente para horizontes de predição distantes, mas pode ser verificada para alguns problemas, como no fluxo de dados real investigado em Bifet et al. (2013). Algumas aplicações geram exemplos que são totalmente independentes daqueles gerados anteriormente, ou seja, possuem uma distribuição de Poisson (Gama, 2010). Consequentemente, nesses casos, a ordem dos exemplos não possui nenhuma relevância para a seleção de algoritmos.

Nas próximas seções são discutidas quais medidas de caracterização podem ser aplicadas e quais dados do nível base e do nível meta podem ser utilizados na obtenção de informações relevantes.

4.3.1 Caracterização de dados do nível base

Nesta seção, é apresentada uma proposta para a extração sistemática de características de fluxos de dados. Na Figura 4.5, é mostrado um esquema que serve de base à apresentação, em que são representados os dados de treinamento (*trein.*), horizonte de predição (*horiz.*) e de seleção de algoritmos (*sel.*), divididos em atributos preditivos $\mathbf{X}$,

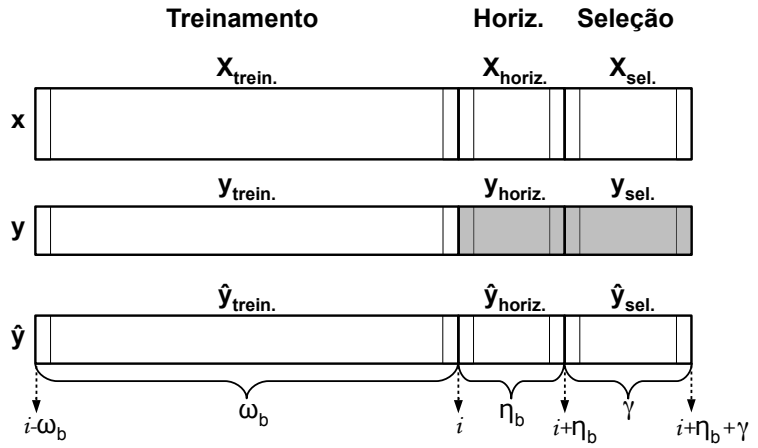


Figura 4.5: Fluxo de dados no nível base separado entre atributos preditivos, atributo alvo e predições dos modelos para os dados de treinamento, horizonte de predição e seleção.

atributo alvo $\mathbf{y}$ e as predições do modelo $\hat{\mathbf{y}}$ a fim de facilitar a compreensão e distinção das características que podem ser extraídas desses dados. Assim, nove conjuntos de dados, doravante denominados de blocos de dados, podem ser visualizados nessa figura: $\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$, $\mathbf{y_{trein.}}$, $\mathbf{y_{horiz.}}$, $\mathbf{y_{sel.}}$, $\hat{\mathbf{y}}_{\mathbf{trein.}}$, $\hat{\mathbf{y}}_{\mathbf{horiz.}}$ e $\hat{\mathbf{y}}_{\mathbf{sel.}}$. No instante de tempo mostrado nessa figura, o MetaStream será aplicado para selecionar um algoritmo para os exemplos de índices $[i + \eta_b + 1, \ldots, i + \eta_b + \gamma]$, sendo que os blocos de dados em branco estão disponíveis para caracterização, enquanto os dados dos blocos em cinza ainda não foram observados.

As medidas de caracterização dos dados podem ser aplicadas para extrair características de cada atributo separadamente ou de relações entre os próprios atributos ou entre os atributos e as predições. Algumas medidas também podem caracterizar cada bloco de dados separadamente ou relações entre diferentes blocos de dados. Alguns símbolos são usados para facilitar a representação de vários blocos de dados ou da relação entre eles. O símbolo $_{\odot}$ é usado quando uma medida pode ser aplicada para um ou mais atributos dos dados de treinamento, horizonte de predição e seleção, como $\mathbf{X_{trein.}}$ representa $\mathbf{X_{horiz.}}$ e $\mathbf{X_{sel.}}$. A caracterização de relações entre atributos é representada por uma seta unilateral $\rightarrow$, quando se calcula a influência de atributos do bloco da esquerda sobre atributos do bloco da direita, ou uma seta bilateral $\leftrightarrow$, quando a medida se refere à relação entre atributos dos dois blocos. Por exemplo, uma medida que calcula a relação entre os valores reais do atributo alvo e as predições dos modelos para os dados de treinamento é representada por $\mathbf{y_{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{trein.}}$. Quando as relações podem ser calculadas para os dados de treinamento, horizonte de predição e seleção, usa-se o símbolo $_{\odot}$ para facilitar sua representação. Assim, $\mathbf{X_{trein.}} \leftrightarrow \mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}} \leftrightarrow \mathbf{X_{horiz.}}$ e $\mathbf{X_{sel.}} \leftrightarrow \mathbf{X_{sel.}}$ pode ser representado apenas por $\mathbf{X}_{\odot} \leftrightarrow \mathbf{X}_{\odot}$. Também é possível calcular relações entre os dados de treinamento, horizonte de predição e seleção, como $\mathbf{X}_{trein.} \leftrightarrow \mathbf{X}_{sel.}$. Quando as medidas podem ser calculadas para todas as combinações, o símbolo $_{\bowtie}$ é usado. Por

Tabela 4.1: Representação estendida e concisa dos dados para os quais pode-se aplicar medidas de caracterização.

$\mathbf{X_{trein.}}, \mathbf{X_{horiz.}}, \mathbf{X_{sel.}}$:	$\mathbf{X}_{\odot}$
$\mathbf{X_{trein.}} \leftrightarrow \mathbf{X_{trein.}}, \mathbf{X_{horiz.}} \leftrightarrow \mathbf{X_{horiz.}}, \mathbf{X_{sel.}} \leftrightarrow \mathbf{X_{sel.}}$:	$\mathbf{X}_{\odot} \leftrightarrow \mathbf{X}_{\odot}$
$\mathbf{X_{trein.}} \leftrightarrow \mathbf{X_{sel.}}, \mathbf{X_{trein.}} \leftrightarrow \mathbf{X_{horiz.}}, \mathbf{X_{horiz.}} \leftrightarrow \mathbf{X_{sel.}}$:	$\mathbf{X}_{\bowtie} \leftrightarrow \mathbf{X}_{\bowtie}$
$\hat{\mathbf{y}}_{\mathbf{trein.}}, \hat{\mathbf{y}}_{\mathbf{horiz.}}, \hat{\mathbf{y}}_{\mathbf{sel.}}$:	$\hat{\mathbf{y}}_{\odot}$
$\hat{\mathbf{y}}_{\mathbf{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{horiz.}}, \hat{\mathbf{y}}_{\mathbf{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{sel.}}, \hat{\mathbf{y}}_{\mathbf{horiz.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{sel.}}$:	$\hat{\mathbf{y}}_{\bowtie} \leftrightarrow \hat{\mathbf{y}}_{\bowtie}$

exemplo, as relações entre todos os blocos de dados para os atributos preditivos $\mathbf{X}$ podem ser representadas apenas por $\mathbf{X}_{\bowtie} \leftrightarrow \mathbf{X}_{\bowtie}$, ao invés de $\mathbf{X_{trein.}} \leftrightarrow \mathbf{X_{sel.}}$, $\mathbf{X_{trein.}} \leftrightarrow \mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}} \leftrightarrow \mathbf{X_{horiz.}}$. Na Tabela 4.1 é mostrado um resumo dessas representações. A seguir são descritos quais dados do nível base são usados para a criação dos meta-dados e sua relevância para a seleção de algoritmos no cenário de fluxo de dados.

$\mathbf{X_{trein.}}$, $\mathbf{X_{trein.}} \leftrightarrow \mathbf{X_{trein.}}$

$\mathbf{X_{trein.}}$ é a tabela dos valores dos atributos preditivos para os dados de treinamento. Cada linha i corresponde a um exemplo z_i e cada coluna j corresponde ao atributo preditivo x_j.

As características obtidas dos dados $\mathbf{X_{trein.}}$ podem ser informativas sobre o comportamento do modelo, pois $\mathbf{X_{trein.}}$ contém todos os valores dos atributos preditivos usados na indução do modelo. Por exemplo, a correlação entre atributos pode ser uma característica útil para a seleção de algoritmos. Se os atributos são correlacionados, algoritmos que não têm o pressuposto da condição de independência são mais promissores do que aqueles que têm esse pressuposto, como o classificador Naive Bayes (Witten e Frank, 2005). Medidas comumente usadas em meta-aprendizado, tais como aquelas usadas em Michie et al. (1994), Kalousis (2002) e Soares (2004), e medidas que consideram a ordem de chegada dos exemplos, tais como em Adya et al. (2001), Prudêncio e Ludermir (2004), Wang et al. (2009) e Lemke e Gabrys (2010), podem ser aplicadas para caracterizar $\mathbf{X_{trein.}}$.

$\mathbf{X_{horiz.}}$, $\mathbf{X_{horiz.}} \leftrightarrow \mathbf{X_{horiz.}}$

$\mathbf{X_{horiz.}}$ é a tabela dos valores dos atributos preditivos para os dados do conjunto horizonte de predição. Novamente cada linha i corresponde a um exemplo z_i e cada coluna j correspondente ao atributo preditivo x_j.

As medidas aplicadas para extrair características de $\mathbf{X_{trein.}}$ também podem ser empregadas para $\mathbf{X_{horiz.}}$. Apesar de $\mathbf{X_{horiz.}}$ não ser usado para induzir o modelo, esses dados podem ser importantes na predição do melhor algoritmo, pois $\mathbf{X_{horiz.}}$ contém informações mais recentes dos meta-atributos preditivos do que $\mathbf{X_{trein.}}$. Uma situação hipotética para salientar a importância dos dados do horizonte de predição é quando o melhor algoritmo para o conjunto de seleção diverge do melhor algoritmo para o conjunto de treinamento

após uma mudança abrupta nos dados. Nesse caso, os dados $\mathbf{X_{horiz.}}$ podem conter informações sobre essa mudança e ajudar na seleção correta do melhor algoritmo, pois são mais semelhantes aos dados do conjunto de seleção.

$\mathbf{X_{sel.}}$, $\mathbf{X_{sel.}} \leftrightarrow \mathbf{X_{sel.}}$

$\mathbf{X_{sel.}}$ é a tabela dos valores dos atributos preditivos para o conjunto de seleção de algoritmos. Assim como para $\mathbf{X_{trein.}}$ e $\mathbf{X_{horiz.}}$, cada linha i corresponde a um exemplo z_i e cada coluna j correspondente ao atributo preditivo x_j.

As características obtidas de $\mathbf{X_{sel.}}$ podem ser comparadas com aquelas de $\mathbf{X_{trein.}}$ para determinar se o algoritmo mais apropriado para os dados de treinamento será o mesmo para os dados de seleção. Suponha que as medidas de caracterização tenham detectado a presença de *outliers* nos dados de seleção mas não nos de treinamento. Portanto, se o melhor algoritmo para os dados de treinamento for sensível a *outliers*, esse algoritmo pode não ser a melhor escolha para os dados de seleção. Como os dados de treinamento, horizonte de predição e de seleção são descritos pelo mesmo conjunto de atributos, as medidas empregadas para $\mathbf{X_{trein.}}$ e $\mathbf{X_{horiz.}}$ também podem ser aplicadas para $\mathbf{X_{sel.}}$.

$\mathbf{y_{trein.}}$

$\mathbf{y_{trein.}}$ é o vetor dos valores do atributo alvo y dos dados de treinamento.

As características extraídas de $\mathbf{y_{trein.}}$ podem ser importantes para guiar o processo de aprendizado no nível meta, pois, dado que pode existir alguma dependência temporal entre os dados, é de se esperar que a natureza dos valores de $\mathbf{y_{sel.}}$ dependam dos valores de $\mathbf{y_{trein.}}$, que os antecedem. Além disso, se os dados não mudam com muita frequência, a distribuição de $\mathbf{y_{sel.}}$ pode ser a mesma dos dados de treinamento $\mathbf{y_{trein.}}$.

Em problemas de regressão, medidas que caracterizam dados numéricos podem ser aplicadas para $\mathbf{y_{trein.}}$. Apesar da dependência entre os valores de $\mathbf{y_{trein.}}$ possivelmente não ser tão forte quanto em séries temporais, o teste de correlação serial e outras medidas para caracterização de séries temporais podem ser relevantes para descrever $\mathbf{y_{trein.}}$ (Lemke e Gabrys, 2010; Wang et al., 2009; Prudêncio e Ludermir, 2004; Adya et al., 2001). Em problemas de classificação, medidas para dados categóricos podem ser utilizadas, como uma medida de impureza ou de informação mútua.

$\hat{\mathbf{y}}_{\odot}$

$\hat{\mathbf{y}}_{\odot}$, que representa $\hat{\mathbf{y}}_{\mathbf{trein.}}$, $\hat{\mathbf{y}}_{\mathbf{horiz.}}$ e $\hat{\mathbf{y}}_{\mathbf{sel.}}$, são os vetores das predições obtidas por um modelo para os dados de treinamento, horizonte de predição e seleção, respectivamente.

As predições para os exemplos dos conjuntos de treinamento e horizonte de predição, $\hat{\mathbf{y}}_{\mathbf{trein.}}$ e $\hat{\mathbf{y}}_{\mathbf{horiz.}}$, respectivamente, foram feitas quando esses exemplos pertenciam ao conjunto de seleção de algoritmos e, portanto, são apenas recuperadas. Por outro lado, para

os exemplos atuais do conjunto de seleção, as predições são feitas no mesmo instante de tempo em que um algoritmo precisa ser escolhido para esses dados. Se houver restrição de tempo que impeça a predição do atributo alvo por todos os algoritmos, então as predições para os exemplos do conjunto de seleção, $\hat{\mathbf{y}}_{\mathbf{sel}}$, não poderão ser caracterizadas no momento da seleção do algoritmo.

As predições $\hat{\mathbf{y}}_{\odot}$ podem prover, principalmente, informações sobre o comportamento dos modelos. Por exemplo, a presença de *outliers* ou uma alta variância das predições pode indicar que escolher o algoritmo que induziu esse modelo é mais arriscado do que escolher algum outro com uma menor variância ou que não fez predições discrepantes.

$\mathbf{X}_{\bowtie} \leftrightarrow \mathbf{X}_{\bowtie}$

Como mencionado anteriormente, o conjunto de atributos que descrevem os dados não mudam ao longo do tempo. Portanto, é possível calcular medidas de relação entre os valores de cada atributo entre os dados de treinamento, horizonte de predição e seleção. A diferença entre a média de um atributo numérico do conjunto de treinamento e do conjunto de seleção, por exemplo, pode indicar alguma mudança nesse atributo. Essas relações entre os atributos não pode ser explorada em aplicações de meta-aprendizado que lidam com diferentes conjuntos de dados, pois tais conjuntos são morfologicamente diferentes.

Ao invés de calcular a relação entre cada atributo, pode ser mais relevante medir a distância entre as distribuições dos dados de treinamento, horizonte e seleção. Por exemplo, a distância entre $\mathbf{X}_{\mathbf{trein.}}$ e $\mathbf{X}_{\mathbf{sel.}}$ pode indicar se ocorreu alguma mudança na distribuição entre os dados de treinamento e teste e ajudar na decisão do melhor algoritmo para os dados de seleção. A distância entre os dados pode ser calculada usando diferentes medidas (Tao e Ozsu, 2009), como a discrepância relativa (Kifer et al., 2004) e a Kullback-Leibler ou entropia relativa (Dasu et al., 2009; Sebastião et al., 2009).

$\hat{\mathbf{y}}_{\bowtie} \leftrightarrow \hat{\mathbf{y}}_{\bowtie}$

A caracterização das relações entre os valores preditos por um modelo para os dados de treinamento, horizonte de predição e seleção segue a mesma ideia da caracterização dos valores dos atributos preditivos. As informações obtidas das relações entre os dados de predição pode revelar mudanças ocorridas no comportamento dos modelos induzidos ao longo do tempo. Essas alterações podem indicar, por exemplo, que a distribuição aprendida para o instante de tempo atual é diferente daquela aprendida anteriormente. Nesse caso, espera-se que os modelos apresentem desempenhos diferentes. Durante o processo de meta-aprendizado, essas características podem complementar aquelas obtidas das relações entre os valores dos atributos preditivos.

$\mathbf{X_{trein.}} \leftrightarrow \mathbf{y_{trein.}}$

Medidas que caracterizam a relação entre os valores dos atributos preditivos e do atributo alvo têm sido comumente usadas na literatura de meta-aprendizado (Kalousis, 2002; Soares, 2004). A caracterização de tal relação pode ser importante para a predição do algoritmo mais promissor, pois supõe-se que os algoritmos de aprendizado sejam capazes de induzir modelos que fazem o mapeamento entre $\mathbf{X_{trein.}}$ e $\mathbf{y_{trein.}}$. Portanto, informações sobre a influência de $\mathbf{X_{trein.}}$ sobre $\mathbf{y_{trein.}}$ podem ser relevante para a tarefa de seleção de algoritmos. A caracterização dessa relação também pode ser realizada por meio de *landmarkers*, que são algoritmos de aprendizado simples que podem ser usados para obter informações sobre os dados para os quais eles são aplicados. Em problemas de regressão, a correlação entre atributos numéricos e o atributo alvo é um exemplo de característica que pode ser obtida a partir desses dados. Se o problema sob análise no nível base é de classificação, a informação mútua pode ser empregada para caracterizar atributos preditivos nominais e o atributo alvo.

$\mathbf{y_{trein.}} \leftrightarrow \mathbf{\hat{y}_{trein.}}$

Medidas de relação entre os valores reais do atributo alvo e as predições realizadas pelos modelos são relevantes para a escolha do melhor algoritmo, pois elas caracterizam o comportamento dos modelos. O método proposto neste trabalho para a seleção de algoritmos supõe que o desempenho preditivo dos modelos induzidos pelos algoritmos de aprendizado para dados passados seja preditivo do desempenho de modelos que venham a ser induzidos e aplicados para dados mais recentes. Assim, é possível predizer o melhor algoritmo para os novos dados com base no comportamento dos modelos para dados antigos. Essa conjectura será falsa caso ocorra uma mudança de conceito real entre os dados de treinamento e seleção (Seção 2.2.2). Qualquer medida de avaliação, como o erro quadrático médio para regressão e a acurácia para classificação, pode ser utilizada para medir o desempenho preditivo dos modelos induzidos.

Relações de relações

Além das relações entre atributos e atributos e predições dos modelos, caracterizar a relação entre duas relações também pode fornecer informação útil para o aprendizado do meta-modelo. O número dessas possíveis relações pode aumentar muito rapidamente com o número de atributos. Portanto, o seu uso deve ser restringido a alguns atributos ou a problemas em que tais características podem, de fato, ser relevantes.

Em problemas de regressão, por exemplo, é possível calcular, para os dados de treinamento, a correlação entre os valores de um atributo preditivo numérico e os valores reais do atributo alvo ($\mathbf{X_{trein.}} \leftrightarrow \mathbf{y_{trein.}}$). Uma alta correlação indica que o atributo preditivo é útil para a predição do atributo alvo. Essa medida também pode ser apli-

cada para mensurar a correlação entre o mesmo atributo preditivo e as predições feitas por um modelo: ($\mathbf{X_{treino.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{treino.}}$). As duas características obtidas dessas relações podem ser relevantes para a seleção de algoritmos, como mencionado anteriormente. Porém, uma outra possibilidade é caracterizar a relação entre essas duas relações, ou seja, ($\mathbf{X_{treino.}} \leftrightarrow \mathbf{y_{treino.}}$) $\leftrightarrow$ ($\mathbf{X_{treino.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{treino.}}$). Essa relação pode, por exemplo, fornecer informação sobre o comportamento do modelo especificamente para um atributo. Se a diferença entre essas duas relações for pequena, pode ser um indício de que o algoritmo de aprendizado reconheceu a utilidade do atributo preditivo. Por outro lado, se a diferença for grande, significa que o algoritmo, embora possa ser eficaz, não considerou a utilidade do respectivo atributo preditivo na indução do modelo.

A relação entre relações de diferentes blocos de dados também pode ser informativa. A diferença da média de um atributo preditivo numérico em $\mathbf{X_{treino.}}$ e da média do mesmo atributo em $\mathbf{X_{sel.}}$, por exemplo, pode indicar se esse atributo está mudando ao longo do tempo. Para ter uma ideia melhor do grau dessa mudança e quão rápido ela está ocorrendo, é possível calcular a diferença entre a média desse atributo entre os dados de treinamento, horizonte e seleção: ($\mathbf{X_{treino.}} \leftrightarrow \mathbf{X_{horiz.}}$) $\leftrightarrow$ ($\mathbf{X_{treino.}} \leftrightarrow \mathbf{X_{sel.}}$). Se a diferença for grande, significa que esse atributo está mudando rapidamente; caso contrário, a mudança está ocorrendo gradualmente.

Dados de diferentes instantes de tempo

Além de aplicar medidas de relação entre dados do mesmo instante de tempo, é possível calcular a relação entre dados para diferentes instantes de tempo, o que pode contribuir para a caracterização das mudanças que ocorrem nos dados. Por exemplo, a relação entre os dados $\hat{\mathbf{y}}_{\mathbf{sel.}}$ no instante de tempo atual, t, e os dados $\hat{\mathbf{y}}_{\mathbf{sel.}}$ do instante anterior $t-1$, como mostrado na Figura 4.6, pode fornecer indícios sobre a mudança de comportamento dos modelos. Entretanto, deve-se notar que os exemplos que pertenciam ao conjunto de seleção no instante de tempo $t-1$ pertencem ao conjunto de horizonte de predição e à janela de treinamento no instante atual. Esse fato pode gerar meta-atributos redundantes, pois pode haver muita sobreposição dos dados, dependendo do tamanho do conjunto de seleção γ.

Assim como na caracterização de relações de relações, considerar dados de diferentes instantes de tempo abre inúmeras possibilidades para a geração de novas características. Portanto, é necessário definir um critério para selecionar quais características podem, de fato, ser úteis para a seleção de algoritmos, evitando gerar uma grande quantidade de meta-atributos irrelevantes.

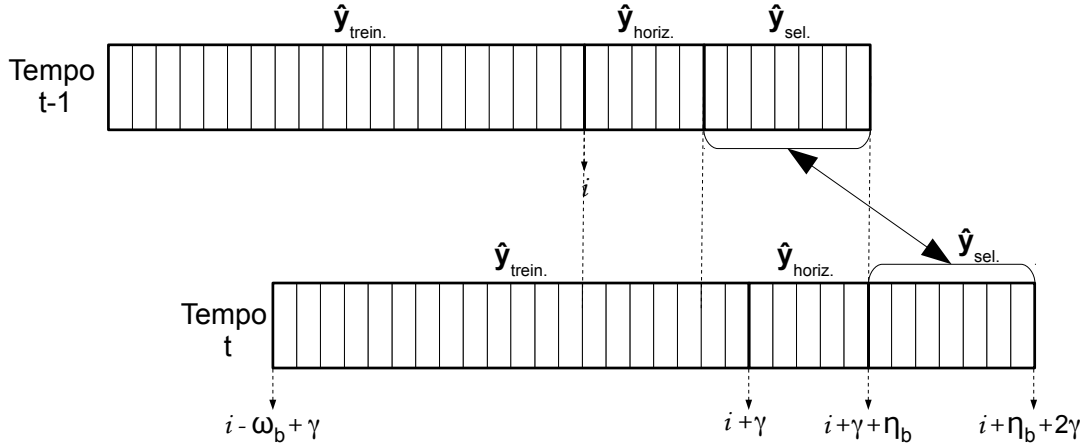


Figura 4.6: Caracterização da relação entre os dados $\hat{\mathbf{y}}_{sel.}$ do instante de tempo atual, t, e os dados $\hat{\mathbf{y}}_{sel.}$ do instante de tempo anterior, $t-1$.

4.3.2 Características independentes e dependentes da morfologia dos dados

Em estudos convencionais de meta-aprendizado para recomendação de algoritmos, os meta-dados são gerados a partir da extração de características de vários conjuntos de dados, que, muito provavelmente, possuem morfologias distintas. Isso implica em um número variável de características, pois algumas medidas produzem um valor de saída para cada atributo. Por exemplo, o cálculo da correlação entre os atributos preditivos numéricos e o atributo alvo do conjunto de treinamento, $\mathbf{X}_{trein.}$ e $\mathbf{y}_{trein.}$, respectivamente, gera uma saída para cada atributo preditivo numérico. Consequentemente, se o algoritmo de aprendizado no nível meta for proposicional (Raedt, 2008), é necessário agregar os valores obtidos para todos os atributos em um único valor, utilizando, por exemplo, a média, para que todos os conjuntos de dados sejam descritos pelos mesmos meta-atributos. Um problema dessa abordagem, usada no projeto STATLOG e em vários trabalhos subsequentes (Soares et al., 2004; Souza et al., 2008; Lemke e Gabrys, 2010; Gomes et al., 2012), é que características similares podem ser extraídas a partir de conjuntos de dados completamente diferentes. Esse problema ocorre, por exemplo, se um conjunto de dados possui atributos que são altamente correlacionados positiva (próximo de 1) e negativamente (próximo de -1) enquanto um outro possui atributos sem nenhuma relação: em ambos a correlação média será próxima de zero, embora os dados sejam completamente distintos (Kalousis, 2002; Vanschoren, 2010).

Algumas alternativas para contornar esse problema foram propostas na literatura. Kalousis e Theoharis (1999) usaram um histograma de valores para representar cada medida que produzia mais de um meta-atributo. Os valores para cada medida são normalizados pelo intervalo teórico para essa medida e são divididos em dez *bins* de tamanhos iguais. Assim, a correlação entre todos os atributos numéricos, por exemplo, pode ser

representada como um vetor de características. Todorovski et al. (2000) incluíram, além da média, o valor máximo e mínimo para cada medida que produzia várias saídas. A fim de reduzir o grande número de meta-atributos gerados, os autores usaram técnicas de seleção de atributos (Liu et al., 2010). Uma alternativa substancialmente distinta das demais foi proposta por Todorovski e Džeroski (1999), que utilizaram programação em lógica indutiva (ILP, do inglês, *inductive logic programming*) (Raedt, 2008), permitindo que os meta-atributos fossem armazenados em uma representação relacional e gerando regras de primeira ordem sobre o comportamento dos algoritmos de aprendizado. Um trabalho similar foi proposto por Kalousis e Hilario (2003), em que um sistema de raciocínio baseado em casos (Aamodt e Plaza, 1994) foi usado.

No cenário de fluxo de dados, o processo de caracterização é realizado continuamente com exemplos que são descritos pelo mesmo conjunto de atributos, que raramente muda ao longo do tempo. Isso significa que as características obtidas para cada atributo ou da relação entre dois ou mais atributos podem ser usadas diretamente no nível meta, sem a necessidade de agregação. Nesta tese, é proposta uma nova categorização das características que podem ser extraídas dos dados. As características **dependentes** do domínio só podem ser representadas sem agregação no nível meta se a morfologia dos dados for idêntica para todos os conjuntos de dados, ou seja, dependem da morfologia dos dados. Por exemplo, a medida de correlação entre os atributos preditivos do conjunto de treinamento ($\mathbf{X}_{trein.}$) gera um valor de saída para cada atributo, que só podem ser utilizados diretamente no nível meta se os dados forem descritos pelos mesmos atributos. Por outro lado, as características **independentes** do domínio podem ser obtidas para conjuntos de dados de diferentes morfologias, com exceção do atributo alvo, que deve ser do mesmo tipo, numérico (regressão) ou categórico (classificação). Em estudos com conjuntos de dados de diferentes morfologias, o número de atributos numéricos e nominais ou o número de classes, para problemas de classificação, são exemplos de medidas independentes. Essas medidas, especificamente, não são relevantes para caracterizar um fluxo de dados, pois este sempre possui o mesmo número de atributos e classes, ou raramente muda. Exemplos de medidas independentes que podem ser relevantes em fluxos de dados são aquelas que caracterizam os valores reais ($\mathbf{y}$) e as predições ($\hat{\mathbf{y}}$) do atributo alvo, como a medida de assimetria (regressão) ou entropia (classificação), ou as relações entre eles, $\mathbf{y}_{trein.} \leftrightarrow \hat{\mathbf{y}}_{trein.}$.

4.3.3 Caracterização do nível meta

Nas seções anteriores, foi apresentada uma abordagem para caracterização dos dados do nível base. A cada janela de seleção de algoritmos, um meta-exemplo é gerado, formando um fluxo de meta-dados ao longo do tempo. Esse fluxo de meta-dados e o próprio processo de aprendizado no nível meta também podem fornecer informações importan-

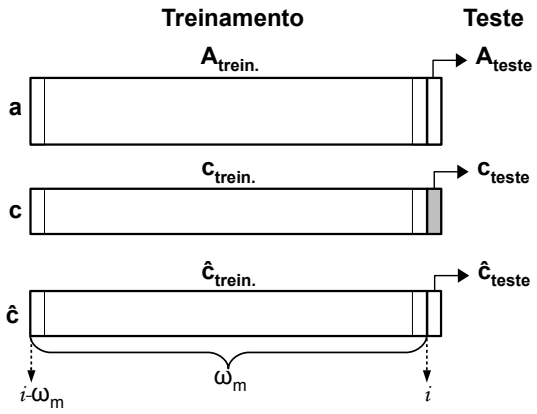


Figura 4.7: Fluxo de meta-dados com uma janela deslizante de tamanho ω_m. Os dados estão separados em atributos preditivos ($\mathbf{a}$), atributo alvo ($\mathbf{c}$) e predições do meta-modelo ($\hat{\mathbf{c}}$) para os meta-dados de treinamento e teste.

tes para a predição do melhor algoritmo. Essa ideia está relacionada com o trabalho de Klinkenberg (2005), que usou apenas características relativas ao processo de aprendizado para predizer o algoritmo mais apropriado para dados futuros. Nesta tese, diferentemente de Klinkenberg (2005), as características obtidas do nível meta, seja a partir dos meta-atributos ou do processo de aprendizado no nível meta, são apenas complementares àquelas obtidas do nível base.

No nível meta, o método testar-então-treinar intercalado também é usado para induzir e avaliar o meta-modelo, conforme apresentado na Seção 4.2. Assim, na Figura 4.7, pode-se visualizar uma janela deslizante com ω_m meta-exemplos de treinamento e um meta-exemplo de teste. Os blocos de dados em branco estão disponíveis no momento da predição da classe (algoritmo) do meta-exemplo de teste, enquanto o valor real do meta-atributo alvo, em cinza, só será conhecido depois que os valores para o atributo alvo dos exemplos do nível base, $\mathbf{y}_{sel.}$, correspondente a esse meta-exemplo, forem observados.

A partir da Figura 4.7, é possível delinear alguns meta-atributos que podem ser obtidos a partir dos meta-dados disponíveis, como: i) a distribuição de classes dos meta-exemplos; ii) a taxa de erro para cada classe; iii) um valor nominal indicando se o meta-exemplo foi predito corretamente ou não; iv) última classe predita. Esses meta-atributos são computados sempre para os dados do instante de tempo anterior. Assim, o valor do meta-atributo que mede a distribuição de classes no nível meta para o meta-exemplo $i+1$, é calculado para os meta-exemplos de $i-\omega_m$ até i. O mesmo deve ocorrer para as outras características que podem ser extraídas dos meta-dados.

4.4 Seleção de um algoritmo para cada exemplo

No cenário de FCD, os dados podem mudar ao longo do tempo quase que arbitrariamente, mas é necessário que um conceito permaneça por um período mínimo de tempo

Tabela 4.2: Erros preditivos dos algoritmos hipotéticos A e B para cada exemplo do nível base, erro médio considerando todos os exemplos e o menor erro teórico (selecionando sempre o melhor algoritmo) para cada exemplo.

	Ex.1	Ex.2	Ex.3	Ex.4	Ex.5	Ex.6	Erro médio
Erro do alg. A	0.20	0.50	0.20	0.15	0.60	0.40	0.34
Erro do alg. B	0.40	0.10	0.40	0.30	0.20	0.25	0.28
Menor erro	0.20	0.10	0.20	0.15	0.20	0.25	0.18

para que seja possível aprender algo sobre ele (Harries et al., 1998; Klinkenberg, 2005). Assim, espera-se que o melhor algoritmo também não mude com tanta frequência. Entretanto, outros fatores podem interferir na determinação do melhor algoritmo, como as mudanças virtuais de conceito, o nível de ruído nos dados e o grau de dependência entre os exemplos, como discutido nos capítulos anteriores. Devido a esses e outros fatores, o algoritmo selecionado para um lote de exemplos dificilmente será o melhor algoritmo para cada exemplo desse lote.

Nesta seção, é proposta uma abordagem para a seleção de algoritmos para cada exemplo do nível base. Essa abordagem é denominada de **SelUnit** no restante desta tese. Conjectura-se que, com as devidas adaptações necessárias, o método MetaStream seja capaz de selecionar o melhor algoritmo para cada exemplo do nível base. Se essa hipótese for confirmada, o ganho preditivo que pode ser obtido com a seleção de algoritmos usando a abordagem SelUnit é maior do que em comparação com a seleção de algoritmos para lotes de exemplos, doravante denominada de **SelLote**. Obviamente, essa melhora depende do desempenho preditivo do método MetaStream em cada abordagem. Isso pode ser ilustrado com a Tabela 4.2, em que é mostrado o erro preditivo dos modelos induzidos por dois algoritmos de aprendizado hipotéticos, A e B, para cada um dos seis exemplos do nível base (Ex.1, ..., Ex.6) e o erro preditivo médio para esses exemplos. Adicionalmente, na última linha, é mostrado o menor erro para cada exemplo de teste se o melhor algoritmo for escolhido. Por essa tabela, é evidente que se o MetaStream sempre predisser o melhor algoritmo corretamente nas duas abordagens, o erro com SelUnit (0.18) será menor do que com SelLote (0.28, selecionando o algoritmo B).

Uma das principais adaptações necessárias no método MetaStream para a abordagem SelUnit é o processo de caracterização dos dados, que é o principal aspecto que a diferencia da abordagem SelLote. A abordagem de caracterização conforme apresentada na seção anterior, parte do pressuposto da existência de vários exemplos na janela de treinamento e nos conjuntos horizonte de predição e de seleção de algoritmos. Em métodos baseados em meta-aprendizado convencional, a caracterização é feita sempre para um conjunto de dados. Em FCD, a suposição de que há vários exemplos para cada bloco de dados é sempre verdadeira para a janela de treinamento, pois é necessário uma quantidade mínima de dados para induzir um modelo. Porém, o conjunto de seleção na abordagem SelUnit

contém um único exemplo, $\gamma = 1$, o que significa que não é possível seguir o processo de caracterização descrito anteriormente.

Para caracterizar o único exemplo do conjunto de seleção, os próprios valores dos atributos desse exemplo são usados para descrevê-lo no nível meta. Assim, os meta-atributos são constituídos pelas características extraídas dos dados de treinamento e horizonte de predição e dos valores dos atributos do único exemplo do conjunto de seleção de algoritmos. Os valores dos atributos preditivos do nível base também são utilizados como meta-atributos preditivos em Gama e Kosina (2011), mas, nesse trabalho, apenas essas características são usadas para a seleção de modelos. A mesma ideia pode ser utilizada se o conjunto horizonte de predição tiver um único exemplo, $\eta_b = 1$, acrescentando os valores dos atributos desse exemplo aos meta-atributos.

Uma ilustração do processo de geração de meta-atributos para SelLote e SelUnit considerando apenas os atributos preditivos base é mostrada na Figura 4.8. Na caracterização para SelLote, considerando que o tamanho do conjunto de seleção é igual ao do tamanho do conjunto de teste, $\gamma = \lambda_b = 8$, um único meta-exemplo será criado para cada conjunto de teste. Para cada meta-exemplo, o vetor de meta-atributos preditivos $\mathbf{a}$ é formado por k características extraídas dos conjuntos de treinamento e horizonte de predição e l características extraídas do conjunto de seleção, $\mathbf{a} = (a_1, \ldots, a_k, a_{k+1}, \ldots, a_{k+l})$. Na caracterização para SelUnit, um meta-exemplo é criado para cada exemplo de teste, pois um conjunto de seleção corresponde a um único exemplo de teste, ou seja, $\gamma = 1$. Para cada meta-exemplo, o vetor de meta-atributos preditivos $\mathbf{a}$ é formado pelos mesmos valores dos primeiros k meta-atributos de SelLote e pelos valores dos m atributos preditivos do respectivo exemplo de teste, $\mathbf{a} = (a_1, \ldots, a_k, a_{k+1}, \ldots, a_{k+m})$. Os valores dos primeiros k meta-atributos são os mesmos para os 8 meta-exemplos de SelUnit. Porém, os valores dos meta-atributos de a_{k+1} até a_{k+m} são diferentes para cada meta-exemplo.

A caracterização dos dados para as abordagens SelLote e SelUnit não difere apenas para os atributos preditivos do conjunto de seleção, como na ilustração anterior, mas para todos os dados de seleção (Figura 4.5). Entretanto, como os valores reais do atributo alvo não são conhecidos, resta apena as predição realizada pelo modelo, $\hat{\mathbf{y}}_{\mathbf{sel}.}$. Portanto, o valor dessa predição também pode ser utilizado diretamente como um meta-atributo, assim como foi feito com os atributos preditivos.

4.5 Complexidade

Geralmente, um dos fatores críticos em aplicações de meta-aprendizado e mineração de fluxos de dados é o custo computacional. Em meta-aprendizado, esse custo está relacionado principalmente com o cálculo dos meta-atributos, enquanto em fluxo de dados, os algoritmos devem ser rápidos o suficiente para processar os exemplos, pelo menos, com a mesma taxa com que eles chegam.

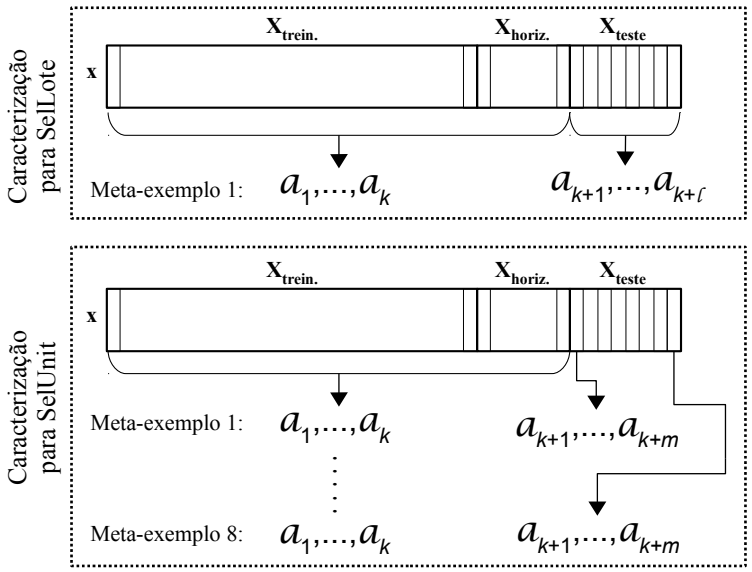


Figura 4.8: Geração dos meta-atributos a partir dos atributos preditivos do nível base para a seleção de um algoritmo para um lote de exemplos (SelLote) e para cada exemplo do nível base (SelUnit).

Em ambientes onde os dados são supostamente estacionários, o melhor algoritmo para um determinado conjunto de dados pode ser conhecido avaliando-se todos os algoritmos para uma amostra desses dados. Nesse cenário, a utilização de um sistema baseado em meta-aprendizado somente é viável se o tempo necessário para extrair as características dos dados e induzir um meta-modelo for menor do que avaliar todos os algoritmos de aprendizado para a amostra disponível dos dados. De acordo com Brazdil et al. (2009), a redução do custo computacional é um dos principais objetivos da recomendação de algoritmos. Essa redução, segundo os autores, é possível diminuindo o número de algoritmos avaliados em um dado problema, com a perda mínima de qualidade dos resultados obtidos quando comparado com os melhores algoritmos possíveis. Para isso, Pfahringer et al. (2000) argumentam que o processo de extração de características dos dados deve ter um baixo custo computacional, preferencialmente $O(n \log n)$, em que n é o número de exemplos. A maioria das medidas utilizadas nesta tese (Capítulo 5) também foram utilizadas em Kalousis (2002), e, segundo o autor, possuem complexidade $O(n)$.

Ao contrário do cenário descrito no parágrafo anterior, em FCD os dados podem mudar ao longo do tempo. Portanto, o desempenho preditivo de um modelo que é induzido e avaliado com uma amostra inicial desses dados, não pode ser considerado como uma estimativa confiável do poder de generalização desse modelo, pois os dados podem mudar quase que arbitrariamente e afetar o desempenho desse modelo. Nesse contexto, o custo computacional do sistema de meta-aprendizado não deve ser limitado ao tempo necessário para executar todos os algoritmos em uma amostra de dados, pois pois não há como estimar confiavelmente o poder de generalização dos modelos. Por outro lado, existe a restrição em relação à taxa de chegada de novos exemplos. Portanto, é necessário que

o cálculo as características, a indução do meta-modelo e a predição do algoritmo de aprendizado seja feito antes que o próximo exemplo ou lote de exemplos chegue.

Muitos trabalhos em fluxos de dados lidam com problemas em que os exemplos são gerados continuamente e em alta taxa (Gomes et al., 2011; Rajaraman e Ullman, 2011; Bifet et al., 2009; Domingos e Hulten, 2000). Nesta tese, são tratados problemas de fluxos de dados que podem mudar ao longo do tempo, mas que não são gerados em uma quantidade massiva, como ocorre nos dados investigados nos trabalhos supracitados. Portanto, até mesmo os algoritmos tradicionais são capazes de processar esses dados em tempo hábil, como investigado também por Moreira (2008) e Klinkenberg (2005). No entanto, para o método MetaStream, é indiferente o uso de algoritmos incrementais ou em lote nos níveis base ou meta. Esses algoritmos devem apenas ser capazes de processar o exemplo atual antes da chegada do próximo.

4.6 Considerações finais

Neste capítulo, foi apresentado o método proposto nesta tese de doutorado para a seleção de algoritmos para FCD, denominado MetaStream. Esse método foi proposto inicialmente para a seleção de algoritmos para lotes de exemplos (SelLote). Posteriormente, uma extensão possibilitou também a sua aplicabilidade na seleção de algoritmos para cada exemplo do nível base (SelUnit). O principal aspecto que diferencia as abordagens SelLote e SelUnit refere-se à caracterização dos dados. Na primeira abordagem, medidas que extraem características de conjuntos de valores podem ser aplicadas para o conjunto de dados de seleção, que contém os exemplos para os quais um algoritmo será selecionado. Na segunda abordagem, entretanto, há um único exemplo no conjunto de seleção, inviabilizando o uso desses medidas. Nesse caso, os valores dos atributos desse único exemplo são usados como meta-atributos.

O método MetaStream é flexível em relação ao problema tratado no nível base, podendo ser aplicado para problemas de regressão ou classificação. A maior diferença da utilização do método MetaStream para problemas de regressão ou classificação é o conjunto de medidas que podem ser utilizadas para caracterizar o atributo alvo e suas relações. Para problemas de regressão, os valores do atributo alvo podem ser caracterizados com medidas que se aplicam a valores numéricos, enquanto que para classificação, podem ser usadas medidas que se aplicam a valores categóricos. Medidas que extraem informação de relações entre dois atributos numéricos ou entre um atributo numérico e um categórico são apropriadas para problemas de regressão, enquanto que para problemas de classificação deve-se usar medidas que caracterizam a relação entre dois atributos categóricos ou entre um atributo categórico e um numérico.

O MetaStream também é independente dos algoritmos utilizados nos níveis base e meta, sendo que a utilização de algoritmos de aprendizado incrementais ou em lote de-

pende da taxa de chegada dos exemplos e da disponibilidade e adequação desses algoritmos com o problema a ser tratado.

Além do método MetaStream, também foram apresentadas neste capítulo propostas para a caracterização de fluxos de dados. Algumas medidas originadas a partir dessas ideias são usadas nos experimentos realizados nesta tese, como será descrito no capítulo seguinte. Essas e outras medidas são definidas e descritas no Apêndice A.

Capítulo 5

Planejamento de Experimentos

Neste capítulo, é apresentado o planejamento experimental empregado na avaliação do método MetaStream a fim de confirmar as hipóteses estabelecidas neste trabalho. Para a realização e posterior análise dos experimentos, é necessário definir o escopo de aplicação do método proposto, as medidas e abordagens para a geração dos meta-dados e outros parâmetros envolvidos no processo de seleção de algoritmos pelo método MetaStream. A avaliação experimental realizada nesta tese contempla problemas de fluxos de dados em que o atributo alvo é numérico. Portanto, o planejamento dos experimentos apresentado no restante deste capítulo, principalmente na geração dos meta-dados, é adequado para problemas de regressão. Entretanto, o MetaStream pode ser facilmente adaptado para problemas de classificação. As principais modificações são em relação às medidas de avaliação dos algoritmos no nível base e às medidas de caracterização do atributo alvo.

O restante do presente capítulo está organizado como segue. Primeiramente, na Seção 5.1, são descritos os três conjuntos de dados reais usados na indução e avaliação de modelos de regressão no nível base. Esses dados já foram investigados por outros estudos da área de fluxo de dados e suas características indicam a ocorrência de mudanças ao longo do tempo. Na Seção 5.2, são apresentados os algoritmos de aprendizado empregados na indução de modelos de regressão, as técnicas para amostragem dos dados e as medidas de avaliação dos modelos induzidos. Em seguida, na Seção 5.3, são apresentados os algoritmos e métodos envolvidos nas principais etapas do nível meta: caracterização dos dados, rotulação dos meta-exemplos e indução dos meta-modelos. Adicionalmente, são descritos os procedimentos para as abordagens de seleção de algoritmos SelLote e SelUnit e a avaliação experimental no nível meta. Na Seção 5.4, são realizados experimentos preliminares para o ajuste de alguns parâmetros envolvidos na avaliação experimental. Por último, na Seção 5.5, são feitas algumas considerações finais sobre este capítulo.

5.1 Conjuntos de dados

Muitos trabalhos na área de fluxo de dados desenvolvem e avaliam seus algoritmos e métodos utilizando dados gerados artificialmente (Street e Kim, 2001; Hulten et al., 2001; Widmer, 1997; Schlimmer e Granger, 1986). Como a geração desses dados é controlada, é possível saber previamente se há mudanças de conceito e quando elas ocorrem e desenvolver algoritmos específicos para tratarem esses dados. Porém, dados reais, normalmente, não são bem comportados como os que são gerados artificialmente e, consequentemente, são mais complexos. Assim, dados artificiais são, geralmente, utilizados durante o desenvolvimento de um novo método, que posteriormente é avaliado também em conjuntos de dados reais (Fidalgo-Merino e Nunez, 2011; Gomes et al., 2011; Ikonomovska et al., 2011; Bifet et al., 2009).

Nesta tese, inicialmente foi testado um conjunto de dados gerado artificialmente, proposto por Friedman (1991) e usado em Ikonomovska et al. (2011). Para esse conjunto de dados, os desempenhos dos algoritmos de regressão testados neste trabalho eram muito estáveis, mesmo quando ocorriam mudanças de conceito. Se o algoritmo que consegue os melhores resultados ao longo de todo o fluxo é sempre o mesmo, não há motivos para a seleção de algoritmos. Com base nesses resultados preliminares, decidiu-se investigar problemas reais de regressão, que são gerados em ambientes dinâmicos e são conhecidos por apresentarem mudanças ao longo do tempo. Essas características provavelmente justificariam o uso de um método de seleção de algoritmos. Sendo assim, os seguintes conjuntos de dados são investigados neste trabalho: predição do tempo de duração de viagens de ônibus (TTP, do inglês, *Travel Time Prediction*), predição de demanda de eletricidade (EDP, do inglês, *Electricity Demand Prediction*) e predição do tempo de atraso de vôos (*Airline*). O conjunto de dados TTP, usado também por Moreira (2008), foi obtido a partir de uma parceria com a Universidade do Porto e a empresa que coletou os dados, enquanto os outros dois conjuntos de dados são de domínio público e foram usados em muitos trabalhos, como em Fidalgo-Merino e Nunez (2011) e Ikonomovska et al. (2011), respectivamente. Na Tabela 5.1 é apresentado o número de exemplos, o número de atributos e o período a que os dados utilizados se referem. A seguir, esses conjuntos de dados são descritos com mais detalhes.

Tabela 5.1: Principais características dos conjuntos de dados investigados.

	#Exemplos	#Atributos	Período
TTP	24 974	6	01/2007 a 03/2008
EDP	27 552	6	05/1997 a 12/1998
Airline	20 285	7	01/2007 a 12/2008

5.1.1 TTP

A predição do tempo de duração de viagens em transportes públicos pode ser útil para diferentes tarefas, como na definição da tripulação, ajudando usuários a decidirem a melhor rota e horário de partida para chegar sem atrasos no destino, e para ajustes em tempo real dos horários planejados (Bajwa et al., 2005). Normalmente, a predição do tempo de duração de viagens precisa ser realizada com alguns dias de antecedência para a primeira tarefa, alguns minutos ou horas para a segunda e em tempo real para a última. Quanto maior o horizonte de predição, maior é a dificuldade de previsão, dado que há muitos fatores importantes, como meteorologia e acidentes, que são mais difíceis de antever quanto maior a antecedência da previsão (Kisgyorgy e Rilett, 2002). Em redes de transportes urbanos, a duração das viagens é estocástico, pois atrasos em intersecções e o tempo gasto nos pontos de parada, por exemplo, oscila espacial e temporariamente (Bin et al., 2006).

O conjunto de dados TTP possui informações sobre o transporte coletivo na cidade de Porto, Portugal, e foi disponibilizado pela Sociedade de Transportes Colectivos do Porto SA[1] (STCP), que administra o transporte coletivo daquela cidade. Os dados foram coletados em um projeto de parceria da empresa com a Universidade do Porto. Os dados foram obtidos de sistemas de gerenciamento de transportes, que armazenam informações sobre a viagem, como a localização por unidade de tempo, bem como registros relativos ao controle operacional, como informações sobre o ônibus e o motorista. Os dados não são necessariamente gerados em tempos igualmente espaçados e a periodicidade pode mudar arbitrariamente, de acordo com estratégias de negócio. Cada viagem constitui um exemplo no conjunto de dados, que é descrito pelos seguintes atributos preditivos: data da viagem, horário de partida, dia da semana, dia do ano, tipo do dia e duração da viagem. O atributo tipo do dia pode ter um dos seguintes valores: normal, feriado, ponte (de feriado) ou tolerância. A duração da viagem é o atributo alvo, cujo valor deseja-se predizer. Esses atributos foram selecionados dentre os que estavam disponíveis por um especialista da área de transportes durante a realização do trabalho de Moreira (2008).

Moreira (2008) utilizou esses dados para investigar o problema de predição do tempo de duração de viagens com três dias de antecedência. Os dados analisados apresentam sazonalidade diária, semanal e anual e mudam continuamente, o que requer uma abordagem contínua de mineração de dados. Nos experimentos realizados, Moreira (2008) considerou cada linha e rota de ônibus separadamente. Entende-se como linha o trajeto completo realizado por um veículo (ida e retorno) e como rota o trajeto realizado por um veículo de um ponto de partida até um ponto de chegada. Algumas linhas possuem informações referentes às duas rotas, ou seja, ida e retorno. Diferentes linhas e rotas da STCP foram consideradas nesses experimentos. O modelo de janelas deslizantes foi utilizado no

[1] `http://www.stcp.pt/`

processo de treinamento e avaliação dos modelos, sendo cada janela de dados considerada como um conjunto de treinamento. A janela deslizante é usada porque supõe-se que os dados mais recentes contêm informações mais importantes que dados antigos para prever o tempo de duração das viagens.

Nesta tese, algoritmos de aprendizado são utilizados para predizer o tempo de duração de viagens com alguns minutos ou horas de antecedência. Portanto, imediatamente após conhecer o tempo real de duração de uma viagem, o exemplo correspondente é usado para treinamento do próximo modelo. A rota 205-1-1, da linha 205, foi utilizada nos experimentos. Os dados dessa rota compreendem o período de primeiro de janeiro de 2007 a 31 de março de 2008. Como normalmente são realizadas diversas viagens em um único dia, um grande número de exemplos, 24 974, foram coletados durante o período mencionado.

5.1.2 EDP

O segundo conjunto de dados foi disponibilizado pela *Australian New South Wales Electricity Market*, que coletou dados durante o período de 7 de maio de 1996 a 5 de dezembro de 1998. A cada 30 minutos, um novo exemplo era coletado, perfazendo um total de 48 exemplos por dia e 45 312 exemplos para todo o período mencionado. Cada exemplo do conjunto de dados possui seis atributos: o dia da semana, a hora do dia, a demanda de eletricidade do estado de New South Wales, a demanda de eletricidade do estado de Victoria, a transferência de eletricidade programada entre esses estados e o sinal da mudança de preço com base na média móvel das últimas 24 horas (valor nominal indicando se o preço aumentou ou reduziu). O conjunto de dados EDP foi primeiramente descrito por Harries (1999) e ficou conhecido como *Elec2*. Harries (1999) mostra que os preços são afetados pela demanda e pelo fornecimento do mercado, pela sazonalidade e sua sensibilidade em relação a eventos de curto prazo, como flutuações das condições climáticas. Outro fator que influenciou os preços foram as importantes mudanças no mercado de eletricidade, como a inclusão de áreas adjacentes. Essa alteração se refere à ligação entre o mercado de eletricidade do estado de Victoria (VIC) e o mercado de New South Wales (NSW), o que permitiu o melhor gerenciamento do fornecimento de eletricidade.

Esse conjunto de dados tem sido usado por muitos autores da área de FCD, como Žliobaite (2011), Kolter e Maloof (2007) e Gama et al. (2004), com o objetivo de predizer um aumento ou uma redução no preço da eletricidade, de acordo com uma média móvel das últimas 24 horas. Diferentemente, Fidalgo-Merino e Nunez (2011) investigaram o problema de predizer a demanda de eletricidade. Ao invés de um problema de classificação, como definido nos outros trabalhos, tem-se, assim, um problema de regressão. Devido às políticas de mercado, normalmente, quanto antes ocorrer a compra ou a venda de

eletricidade, menor o preço pago por ela. Portanto, nos estudos mencionados, a predição do preço sempre é feita com um dia (48 exemplos) ou uma semana de antecedência (336 exemplos).

Nesta tese, o conjunto de dados EDP é investigado com o mesmo propósito de Fidalgo-Merino e Nunez (2011), ou seja, o de predizer a demanda de eletricidade do estado de NSW para a semana seguinte. Os primeiros 17 760 exemplos desse conjunto de dados, referente ao período até 4 de maio de 1997, foram removidos, pois eles não possuem os valores do atributo para a demanda de eletricidade do estado de VIC e a demanda de transferência de eletricidade entre os estados NSW e VIC, que começou a ocorrer após aquela data. Depois dessa remoção, restaram 27 552 exemplos.

5.1.3 Airline

O último conjunto de dados investigado neste estudo foi disponibilizado pela Associação Americana de Estatística (*American Statistical Association*) (ASA, 2009) para uma competição de gráficos estatísticos em 2009. Esse conjunto de dados possui um grande número de exemplos, em que cada um representa um voo comercial doméstico nos Estados Unidos para o período compreendido entre Outubro de 1987 e Dezembro de 2008, totalizando aproximadamente 120 milhões de voos. Cada exemplo é descrito originalmente por 29 atributos, como local de origem e destino, tempo de atraso para partida e chegada e distância percorrida.

O conjunto de dados Airline foi utilizado em Ikonomovska et al. (2011) para predizer o atraso na chegada dos voos utilizando o algoritmo proposto naquele estudo. Após o pré-processamento dos dados, referente ao período de outubro de 1987 a abril de 2008, 116 milhões de exemplos foram utilizados, sendo que cada exemplo é descrito por 13 atributos: ano, mês, dia do mês, dia da semana, horário de partida previsto, horário de chegada previsto, código da empresa de transportes, número do voo, tempo real do voo[2], origem, destino, distância e se houve mudança de rota do voo. Segundo Ikonomovska et al. (2011), a curva de aprendizado obtida confirma a existência de mudanças de conceito, as quais ocorrem em altas taxas e em intervalos irregulares. Ainda segundo os autores, os atributos que mais influenciaram no aprendizado foram a origem, o destino, o tempo de voo, o dia da semana, o dia do mês e os horários previstos de partida e de chegada.

Em Fidalgo-Merino e Nunez (2011), o conjunto de dados Airline foi usado com o objetivo de predizer o tempo de atraso de partida dos voos que saíam do Aeroporto Internacional O'Hare, Chicago (o aeroporto mais movimentado do mundo até 2004). O período compreendido nos experimentos foi de 1987 a 2000, o que resultou em aproximadamente sete milhões de exemplos após o pré-processamento dos dados. O passo da janela deslizante utilizado foi de 10 000 exemplos. Assim, a cada janela (aproximadamente uma

[2]O fato dos autores usarem o tempo real de voo ao invés da previsão, pode ter sido um erro durante a experimentação ou na escrita do artigo, pois essa variável não deve ser usada no treinamento dos modelos.

semana), os modelos eram atualizados e as estatísticas sobre o erro, consumo de memória e tempo de aprendizado dos modelos eram obtidos usando os próximos 10 000 exemplos.

Nesta tese, são utilizados os dados referentes aos voos para uma única rota, assim como foi feito para os dados TTP (Moreira, 2008), com origem no aeroporto Internacional de O'Hare (ORD), Chicago, e destino no Aeroporto de LaGuardia (LGA), Nova Iorque, nos anos de 2007 e 2008. Após a seleção dessa rota e da limpeza dos dados, um total de 20 285 exemplos constituem o novo conjunto de dados. Esses exemplos foram ordenados pelo horário de partida previsto, como em Fidalgo-Merino e Nunez (2011). Alguns atributos, como origem, destino e distância, foram removidos para adequar os dados à amostragem realizada, restando os seguintes atributos: data, dia da semana, horário de partida previsto, horário de chegada previsto, número do voo, tempo de voo previsto e tempo de atraso para partida, que é o atributo alvo que se deseja predizer.

5.2 Nível base

Algoritmos incrementais ou em lote poderiam ser aplicados no nível base para predizer o valor do atributo alvo dos conjuntos de dados mencionados na seção anterior. Porém, como apontado por Žliobaite et al. (2012), ainda há poucos algoritmos de regressão incrementais disponíveis, tais como Online-RD/RA (Potts e Sammut, 2005), FIMT-DD (Ikonomovska et al., 2011), SAIRT (Fidalgo-Merino e Nunez, 2011) e IBLStreams (Shaker e Hüllermeier, 2012).

Devido à escassez de algoritmos de regressão incrementais, algoritmos de aprendizado em lote foram empregados, mas isso não é uma restrição do método MetaStream, que pode também utilizar algoritmos incrementais. Os algoritmos de regressão em lote usados no nível base são:

- *Random Forests* (RF) (Breiman, 2001);
- Máquinas de Vetores de Suporte (SVM, do inglês, *Support Vector Machines*) (Cristianini e Shawe-Taylor, 2000);
- Árvores de Classificação e Regressão (CART, do inglês, *Classification and Regression Trees*) (Breiman et al., 1984);
- *Project Pursuit Regression* (PPR) (Friedman e Stuetzle, 1981);
- *Multivariate Adaptive Regression Splines* (MARS) (Friedman, 1991);
- Regressor Linear (LR, do inglês, *Linear Regression*) (Witten e Frank, 2005).

Apenas o último algoritmo da lista foi usado a partir do programa Weka (Witten e Frank, 2005), que possui uma coleção de algoritmos de mineração de dados. Todos

os demais algoritmos foram usados a partir de pacotes do projeto R (R Core Team, 2012), respectivamente: *randomForest* (Liaw e Wiener, 2002), *e1071* (Meyer et al., 2012), *rpart* (Therneau et al., 2012), *stats* (R Core Team, 2012) e *earth* (Milborrow, 2012). Nenhum dos parâmetros desses algoritmos foi modificado, ou seja, os valores padrão, sugeridos pelas implementações dos mesmos, foram usados nos experimentos. O único parâmetro que não possui um valor padrão e precisou ser informado foi o número de termos do algoritmo PPR, que recebeu o valor 1, com base em Moreira (2008).

O ajuste de parâmetros desses algoritmos não foi realizado no presente trabalho porque o objetivo desta tese não é obter o melhor resultado absoluto no nível base para comparálo com o estado da arte para esses conjuntos de dados, mas sim avaliar o desempenho de métodos de seleção de algoritmos. Assim, para este trabalho, é mais importante que o melhor algoritmo para um FCD varie no tempo.

Para avaliar esses algoritmos no nível base, janelas deslizantes são utilizadas para a amostragem dos dados. Embora existam heurísticas e algoritmos que ajustem o tamanho das janelas automaticamente (Widmer e Kubat, 1996; Bifet e Gavaldà, 2007), neste trabalho, os tamanhos das janelas são fixados a priori, e foram ajustados com base em experimentos preliminares. O tamanho fixo permite maior controle sobre os dados que serão usados para treinamento e teste e reduz a influência de outros fatores externos aos métodos de seleção de algoritmos. Isso garante também que as características são calculadas para conjuntos de dados com o mesmo tamanho, o que é essencial para poder retirar informação sobre o valor dos meta-atributos. No entanto, o MetaStream é aplicável também no caso do tamanho da janela ser ajustado dinamicamente. Os tamanhos das janelas para os conjuntos de dados TTP e Airline foram definidos como $\omega_b = 1000$ com base em experimentos com 300 e 1000 exemplos. Para esses dados, o horizonte de predição, η_b, foi definido como o menor intervalo (em número de exemplos) entre predizer o valor do atributo alvo de um exemplo e observar o seu valor real. Assim, esse parâmetro foi definido como $\eta_b = 2$ e $\eta_b = 5$ para os conjuntos de dados TTP e Airline, respectivamente. O tamanho do conjunto de teste, que determina também o passo da janela, foi definido como $\lambda_b = 1$. Assim, um modelo é induzido com os exemplos da janela de treinamento e é usado para predizer o atributo alvo do próximo exemplo disponível. Para o conjunto de dados EDP, o tamanho da janela foi definido para $\omega_b = 672$, que representa duas semanas de dados, com base em experimentos com 336, 672 e 1680 exemplos. O horizonte de predição é zero, $\eta_b = 0$ e o conjunto de teste é formado por $\lambda_b = 336$ exemplos, ou uma semana, assim como foi definido em outros estudos que utilizaram esse conjunto de dados, como Fidalgo-Merino e Nunez (2011) e Gama et al. (2004). Em resumo, um modelo é induzido a cada semana utilizando os dados das últimas duas semanas e prediz a demanda de eletricidade para a semana seguinte. Na Tabela 5.2, são apresentados os valores dos parâmetros para o aprendizado no nível base para cada conjunto de dados.

Tabela 5.2: Valores dos parâmetros para o aprendizado no nível base para cada conjunto de dados.

Conjunto de dados	Parâmetro ω_b	η_b	λ_b
TTP	1000	2	1
EDP	672	0	336
Airline	1000	5	1

5.3 Nível meta

Como mencionado no Capítulo 3, a recomendação de algoritmos por sistemas de meta-aprendizado pode ser formulado como um problema de ranking (Soares e Brazdil, 2000), classificação (Bensusan e Giraud-Carrier, 2000b; Brazdil et al., 1994) ou regressão (Köpf et al., 2000). Nas duas primeiras abordagens, geralmente, um único meta-modelo global é induzido. Assim, a caracterização dos dados deve conter informações que sejam relevantes para todos os algoritmos de aprendizado utilizados. Porém, é muito provável que algumas características sejam importantes para a predição do desempenho de alguns algoritmos, mas sejam irrelevantes para outros. Portanto, com o uso de um único meta-modelo, perde-se flexibilidade e, possivelmente, reduz-se o desempenho preditivo (Kalousis, 2002). A formulação da tarefa de recomendação de algoritmos como um problema de regressão geralmente emprega um único meta-modelo para cada algoritmo com o objetivo de predizer o desempenho desse algoritmo. Apesar de flexível, essa formulação pode não ser adequada para prever o desempenho relativo dos algoritmos, o que é necessário para a tarefa de seleção, que é o objetivo principal deste trabalho.

Para lidar com estas questões, nesta tese, utiliza-se a forma de recomendação por classificação, em que cada par de algoritmos constitui um problema de meta-aprendizado de classificação, semelhante ao que foi investigado em Kalousis (2002). Espera-se que, com essa abordagem, o problema de meta-aprendizado seja mais simples do que se todos os algoritmos fossem considerados, pois é possível induzir um meta-modelo específico para cada par de algoritmos. Além disso, o tratamento aos pares permite uma investigação mais detalhada dos fatores que podem ser relevantes para o sucesso ou fracasso dos métodos de seleção de algoritmos. Assim, se m algoritmos de aprendizado forem utilizados no nível base, haverá $\binom{m}{2}$ problemas de meta-aprendizado. Como são utilizados 6 algoritmos de regressão no nível base (Sec. 5.2), 15 problemas de meta-aprendizado são investigados para cada avaliação experimental. A principal desvantagem dessa abordagem é o custo computacional. Se todos algoritmos fossem considerados de uma vez, um único problema de meta-aprendizado, com m classes diferentes, seria investigado e um único meta-modelo poderia ser empregado para esse fim.

Em AM, comumente compara-se o algoritmo que está sendo proposto com estratégias de predição simples, como classificar um exemplo de teste segundo a classe majoritária no

conjunto de treinamento, conhecida também como classe padrão. Uma extensão natural dessa abordagem para o caso de janelas deslizantes é utilizar a classe majoritária dos meta-exemplos presentes nessa janela para cada instante de tempo. Dessa maneira, se a classe majoritária mudar ao longo do tempo, a predição também será automaticamente atualizada. Esse método será denominado **Default**, por ser um termo em inglês usual na área de meta-apredizado (Soares et al., 2004; Kalousis e Hilario, 2003). O Default também foi comparado a outra abordagem simples, denominada *No-Change* em Bifet et al. (2013), que consiste em predizer o valor do novo exemplo apenas com base no último exemplo rotulado. Essa abordagem mostrou-se muito superior ao Default para o conjunto de dados investigado naquele trabalho. Porém, nesta tese, o método Default apresentou melhores resultados em comparação com o *No-Change*. Portanto, a avaliação experimental prosseguiu apenas com o método Default. As diferenças entre este trabalho e o de Bifet et al. (2013) provavelmente foram as causas da divergência entre os resultados. Por exemplo, em Bifet et al. (2013) o problema de classificação ocorre no nível base do fluxo de dados, enquanto que, nesta tese, o Default e o *No-Change* foram empregados no problema de classificação do nível meta, em que ocorre um maior atraso entre classificar um meta-exemplo e observar a sua classe verdadeira.

5.3.1 Caracterização dos dados

As hipóteses estabelecidas neste trabalho podem ser avaliadas empiricamente para diferentes conjuntos de características que descrevem os dados, as quais podem ter grande influência nos resultados obtidos, como mencionado no Capítulo 4. Portanto, é de suma importância a definição das medidas para a caracterização dos dados na avaliação dessas hipóteses. No restante desta seção são descritos os processos e as medidas para a geração dos meta-dados que serão usados na comparação entre os métodos MetaStream e Default, entre as abordagens de seleção de algoritmos SelLote e SelUnit e entre os conjuntos de medidas dependentes e independentes.

Comparação entre MetaStream e Default

A hipótese de que o método MetaStream é superior ao Default para a tarefa de seleção de algoritmos é avaliada empiricamente usando um conjunto de meta-dados que inclui características dependentes e independentes da morfologia dos dados. Quando características dependentes e independentes podem ser obtidas dos mesmos dados, apenas as dependentes são usadas, com base na hipótese de que tais características provêm informações úteis que não são obtidas com as características independentes. Por exemplo, a correlação entre atributos numéricos gera uma saída para cada par de atributos, e, portanto, as características são dependentes da morfologia dos dados. Se essa medida for aplicada para diferentes conjuntos de dados, que não é o caso desta tese, é necessário

agregar essas características em um único valor, calculando, por exemplo, a média de todas as correlações. Nesse caso, a característica resultante é independente da morfologia dos dados. Quando não há características dependentes que podem ser extraídas de determinados dados, as independentes são utilizadas para descrevê-los. Por exemplo, as características obtidas do atributo alvo e do desempenho dos modelos são consideradas independentes, pois geram um único valor de saída, conforme explicado na Seção 4.3.2.

Na Tabela 5.3 são apresentadas todas as medidas empregadas na geração dos meta-dados que serão utilizados nos experimentos de comparação entre os métodos MetaStream e Default, que são apresentados na Seção 6.1. As colunas *Ref.* e *Medida* correspondem às referências e ao nome de cada medida, respectivamente, e a coluna *Dados* refere-se aos dados para os quais cada medida é aplicada. Essas medidas são definidas e descritas detalhadamente no Apêndice A.

É importante mencionar que a caracterização dos dados não tem nenhuma influência sobre o desempenho do Default, pois esse método prediz somente com base na classe majoritária dos meta-exemplos de treinamento e, portanto, apenas as estratégias de rotulação dos meta-exemplos (Seção 5.3.2) é que são relevantes para determinar o seu comportamento.

Comparação das abordagens de seleção de algoritmos SelLote e SelUnit

As maiores diferenças entre as abordagens de seleção de algoritmos SelLote e SelUnit ocorrem no processo de caracterização e rotulação dos dados, o que resulta em diferentes conjuntos de meta-dados. Para explorar essas diferenças da melhor maneira possível, medidas específi cas de cada abordagem foram empregadas. Nos experimentos da abordagem SelLote, são utilizados os mesmos conjuntos de meta-dados gerados para a comparação entre os métodos MetaStream e Default. Na geração dos meta-dados para a abordagem de seleção SelUnit, as medidas da Tabela 5.3 não podem ser aplicadas para caracterizar os dados $\mathbf{X}_{sel.}$, pois há um único exemplo de teste no conjunto seleção de algoritmos para cada instante de tempo. Por outro lado, os valores dos atributos preditivos desse exemplo são usados para caracterizá-lo no nível meta, como discutido na Seção 4.4. Portanto, os valores dos meta-atributos de cada meta-exemplo para a abordagem SelUnit são obtidos com a aplicação das medidas apresentadas na Tabela 5.3 para todos os dados, com exceção de $\mathbf{X}_{sel.}$, e dos valores dos atributos preditivos do exemplo de teste do nível base. Os resultados experimentais e as discussões da comparação entre SelLote e SelUnit são apresentados na Seção 6.2.

Comparação entre medidas independentes e dependentes

A hipótese de que as características dependentes provêm informações que contribuem para melhorar o desempenho preditivo do método MetaStream na seleção de algoritmos

Tabela 5.3: Medidas de caracterização e os dados para os quais elas são aplicadas. A referência e o nome de cada medida, assim como os dados para os quais ela se aplica são apresentados nas colunas *Ref.*, *Medida* e *Dados*, respectivamente.

Ref.	**Medida**	**Dados**
M1	Média aritmética	$\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$, $\mathbf{y_{trein.}}$
M2	Média truncada	$\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$, $\mathbf{y_{trein.}}$
M3	Desvio padrão	$\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$, $\mathbf{y_{trein.}}$
M4	Mediana	$\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$, $\mathbf{y_{trein.}}$
M5	Intervalo interquartil	$\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$, $\mathbf{y_{trein.}}$
M6	Máximo	$\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$, $\mathbf{y_{trein.}}$
M7	Mínimo	$\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$, $\mathbf{y_{trein.}}$
M8	Outliers	$\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$, $\mathbf{y_{trein.}}$
M9	Assimetria	$\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$, $\mathbf{y_{trein.}}$
M10	Curtose	$\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$, $\mathbf{y_{trein.}}$
M11	Coeficiente de variação	$\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$, $\mathbf{y_{trein.}}$
M12	Correlação serial	$\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$, $\mathbf{y_{trein.}}$
M13	Taxa de translação	$\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$, $\mathbf{y_{trein.}}$
M14	Entropia	$\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$
M15	Coeficiente de concentração	$\mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}}$
M16	Correlação	$\mathbf{X_{trein.}} \leftrightarrow \mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}} \leftrightarrow \mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}} \leftrightarrow \mathbf{X_{sel.}}$, $\mathbf{X_{trein.}} \leftrightarrow \mathbf{y_{trein.}}$
M17	p-valor da distribuição F	$\mathbf{X_{trein.}} \leftrightarrow \mathbf{X_{trein.}}$, $\mathbf{X_{horiz.}} \leftrightarrow \mathbf{X_{horiz.}}$, $\mathbf{X_{sel.}} \leftrightarrow \mathbf{X_{sel.}}$, $\mathbf{X_{trein.}} \leftrightarrow \mathbf{y_{trein.}}$
M18	Ganho de dispersão	$\mathbf{X_{trein.}} \leftrightarrow \mathbf{y_{trein.}}$
M19	Intervalo	$\mathbf{y_{trein.}}$
M20	Heterogeneidade	$\mathbf{y_{trein.}}$
M21	Taxa do MSE	$\mathbf{y_{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{trein.}}$
M22	Desvio padrão do SE	$\mathbf{y_{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{trein.}}$
M23	Ranking dos modelos	$\mathbf{y_{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{trein.}}$
M24	Diversidade II	$\mathbf{y_{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{trein.}}$

é investigada empiricamente utilizando dois conjuntos de meta-dados.

O primeiro conjunto, denominado **MDInd** (meta-dados independentes), é formado apenas por meta-atributos independentes da morfologia dos dados. Para gerar esse conjunto, as medidas apresentadas na Tabela 5.3 são aplicadas para os dados apresentados também nessa tabela. Entretanto, diferentemente dos meta-dados gerados para a comparação entre MetaStream e Default, se uma medida gera uma característica para cada atributo ou para cada relação entre atributos, os valores são agregados calculando-se a média. Adicionalmente, os valores máximo e mínimo das características geradas por cada medida também são usados como meta-atributos, conforme proposto por Todorovski et al. (2000). Assim, apenas características que são independentes da morfologia dos dados são usadas como meta-atributos. Por exemplo, a medida de entropia gera n valores, em que n é o número de atributos preditivos categóricos. Portanto, ao invés de n características serem usadas como meta-atributos, apenas três meta-atributos representarão essas características: a média, o valor máximo e o valor mínimo de entropia.

Para verificar se informações úteis são perdidas durante a agregação das características para a geração de MDInd, o segundo conjunto de meta-dados, denominado **MDIndDep** (meta-dados independentes e dependentes), é formado pela adição das características dependentes ao conjunto de meta-dados independentes MDInd. As mesmas medidas e dados da Tabela 5.3 são usadas para a extração das características dependentes, com exceção dos dados $\mathbf{y}$ e $\hat{\mathbf{y}}$ e das relações entre eles, pois, como discutido na Seção 5.3.1, essas características são naturalmente independentes, pois geram um único valor. A outra diferença, é que todas as características extraídas são usadas diretamente como meta-atributos, sem a necessidade de agregação, como foi feito para MDInd. Por exemplo, ao invés de usar a média e os valores máximo e mínimo para representar a entropia de todos os atributos categóricos, a entropia de cada atributo é usada diretamente como um meta-atributo. Como muitas dessas características são obtidas para os mesmos dados, há uma redundância dos meta-atributos no conjunto de dados MDIndDep. Portanto, apenas as características dependentes que possuem correlação linear de Pearson menor que um limiar de 0.9 (definido arbitrariamente) com todos os meta-atributos de MDInd são adicionadas ao conjunto MDIndDep.

5.3.2 Rotulação dos meta-exemplos

Após um meta-exemplo de teste ser predito, ele não é imediatamente rotulado, pois pode ocorrer um atraso na obtenção dos valores reais do atributo alvo dos exemplos do conjunto de seleção. Quando esses valores são observados, o sistema de meta-aprendizado rotula os meta-exemplos gerados a partir desses exemplos base.

A rotulação de cada meta-exemplo é realizada de acordo com o desempenho preditivo obtido pelos regressores no nível base para os γ exemplos do conjunto de seleção

de algoritmos. Esse desempenho pode ser calculado empregando uma ou mais medidas apresentadas na Seção 2.1.1. Nesta tese, quando $\gamma > 1$ (SelLote), a medida NMSE é utilizada para esse fim, pois trata-se de uma medida relativa, o que é uma característica indicada quando há mudanças ao longo do tempo (Armstrong e Collopy, 1992), e é comumente usada pela comunidade de meta-aprendizado (Brazdil et al., 2009). Para avaliar o desempenho dos regressores para um único exemplo de teste, $\gamma = 1$ (SelUnit), o erro absoluto é calculado.

Em aplicações de meta-aprendizado convencional, os meta-exemplos são rotulados de acordo com o erro de generalização dos regressores para o conjunto de dados. Quanto menor o número de exemplos desse conjunto, menor é a confiança na estimativa do desempenho, podendo causar a adição de ruídos durante o processo de rotulação dos meta-exemplos. Entretanto, nesta tese, o desempenho dos regressores sempre é calculado para os exemplos mais recentes que estão chegando. Portanto, o erro usado para rotular um meta-exemplo não é considerado como erro de generalização e não há a adição de ruídos no atributo alvo mesmo quando esse erro é calculado com base em um pequeno número de exemplos.

Três estratégias foram utilizadas neste trabalho para rotular os meta-exemplos, denominadas *Sem-empate*, *Com-empate* e *Combinação*, as quais são descritas a seguir. Estratégias iguais ou similares já foram utilizadas em outros trabalhos de meta-aprendizado (Lemke e Gabrys, 2010; Nascimento et al., 2009; Prudêncio e Ludermir, 2004; Bensusan e Giraud-Carrier, 2000b).

Rotulação Sem-empate

Na estratégia *Sem-empate*, considera-se que sempre existe um regressor com desempenho preditivo superior aos demais. Assim, um meta-exemplo é rotulado como "A" se o erro preditivo do regressor A é menor do que os demais regressores para os exemplos do nível base que são representados pelo meta-exemplo em questão. O erro preditivo é calculado pela medida NMSE na abordagem SelLote e pelo erro absoluto na abordagem SelUnit, pois o erro dos regressores é calculado somente para um exemplo. Se o menor erro for obtido por mais de um regressor, o meta-exemplo será rotulado como um desses regressores aleatoriamente. O empate deverá acontecer raramente, visto que não é utilizado nenhum limiar para desconsiderar pequenas diferenças de desempenho. Essa estratégia foi motivada pela sua simplicidade, por não requerer nenhum parâmetro adicional para rotular os meta-exemplos e porque o desempenho preditivo de cada regressor corresponde ao seu erro real para os exemplos em questão, e não ao erro de generalização, como em uma aplicação convencional de meta-aprendizado.

Rotulação Com-empate

Na estratégia de rotulação *Com-empate*, um meta-exemplo é rotulado com um valor categórico que representa o algoritmo com o menor erro preditivo se e somente se a diferença do erro entre os dois regressores for maior ou igual a um limiar α_{ROT} predefinido. Caso contrário, o meta-exemplo é rotulado como *empate*, indicando que os regressores possuem desempenhos preditivos semelhantes.

O limiar α_{ROT} é um parâmetro que deve ser configurado pelo usuário. Durante o processo de seleção de algoritmos, quando um meta-exemplo é predito como *empate* pelo método MetaStream ou Default, a média das predições dos regressores que possuem desempenhos similares é utilizada no nível base para predizer o valor do atributo alvo. O uso do limiar α_{ROT} para desconsiderar pequenas diferenças entre os desempenhos preditivos pode ser uma vantagem desta estratégia em relação à Sem-empate, pois cria-se uma margem de separação entre as duas classes mais importantes (quando um dos algoritmos claramente é superior ao outro) separando os casos em que essa diferença é menor, e portanto, se espera que seja mais difícil de prever (quando não há um algoritmo claramente melhor que o outro). Para além disso, nesse último caso, a escolha de qualquer um dos algoritmos não afeta consideravelmente o desempenho preditivo no nível base.

A estratégia Com-empate é adequada apenas para a seleção dentre pares de algoritmos, pois sua generalização para um número maior de algoritmos não é trivial. Algum método de classificação multi-rótulos (Madjarov et al., 2012) poderia ser empregado para lidar com mais do que dois algoritmos, mas essa questão está além do escopo desta tese. Apesar do seu potencial, essa estratégia não será analisada no restante desta tese, pois os resultados experimentais obtidos com essa estratégia são similares aos da estratégia Combinação (ver Rossi et al. (2014)), que não é limitada ao uso de apenas dois regressores e não possui parâmetros que precisam ser ajustados.

Rotulação Combinação

A última estratégia de rotulação, denominada *Combinação*, considera a média das predições dos regressores sob análise como uma nova classe, além de considerar cada regressor separadamente. Assim, um meta-exemplo é rotulado como *combinação* se e somente se a média das predições dos regressores resultar em desempenho preditivo superior ao de todas as predições consideradas separadamente. Caso contrário, o rótulo do meta-exemplo indicará o regressor com o menor erro. Portanto, quando um meta-exemplo é predito como sendo da classe "combinação", a média das predições dos modelos será usada como predição do atributo alvo no nível base.

Na estratégia Com-empate, quando um meta-exemplo é predito como sendo da classe *empate*, a média das predições dos regressores também é utilizada na predição do atributo alvo no nível base. Porém, um meta-exemplo é rotulado como *empate* independentemente

de a média das predições ser melhor do que qualquer regressor separadamente, pois a classe "empate" indica apenas que eles possuem desempenho semelhantes. Isso não acontece com Combinação, o que pode ser uma vantagem dessa estratégia.

5.3.3 Meta-aprendizes

O método MetaStream foi desenvolvido para selecionar automaticamente o algoritmo mais promissor a ser empregado no nível base. Visto que a tarefa de seleção de algoritmos é tratada como um problema de classificação nesta tese, é necessário também selecionar o algoritmo de aprendizado (meta-aprendiz) que induzirá o meta-modelo. Esta escolha também pode ser vista como um problema de meta-aprendizado. No entanto, como em outros trabalhos nessa área, aqui a escolha é feita manualmente. Assim como no nível base, algoritmos incrementais ou em lote podem ser usados no nível meta. Devido ao pequeno número de meta-exemplos criados, optou-se por utilizar algoritmos de aprendizado em lote, pois os algoritmos incrementais precisam de um número razoável de exemplos para obter desempenho preditivo similar ao dos algoritmos em lote Domingos e Hulten (2000); Gama (2010). Para os problemas estudados neste trabalho, os algoritmos em lote são capazes de processar cada meta-exemplo antes que o próximo seja gerado. os incrementais são adequados quando o fluxo de dados é abundante. Os seguintes algoritmos foram investigados como meta-aprendizes:

- *Random Forest* (RF): esse algoritmo foi escolhido devido ao seu alto desempenho preditivo em muitos estudos (Musliu e Schwengerer, 2013; Biau, 2012; Nascimento et al., 2009; Caruana e Niculescu-Mizil, 2006) e por ser robusto a ruídos, *overfitting* e a dados com grande dimensionalidade (Breiman, 2001);

- Máquinas de Vetores de Suporte (SVMs): as SVMs foram escolhidas por também apresentarem alto desempenho preditivo em muitos problemas (Steinwart e Christmann, 2008; Cristianini e Shawe-Taylor, 2000; Bennett e Campbell, 2000; Burges, 1998) e já terem sido usadas com sucesso como meta-aprendizes em outros trabalhos (De Souto et al., 2008; Nascimento et al., 2009; Souza, 2010).

Assim como no nível base, os valores padrão dos parâmetros desses algoritmos foram mantidos conforme suas respectivas implementações, que são as mesmas dos pacotes referenciados anteriormente. Esses algoritmos puderam ser usados nos dois níveis de aprendizado, pois são capazes de lidar com problemas de regressão (nível base) e de classificação (nível meta).

5.3.4 Abordagens de seleção de algoritmos SelLote e SelUnit

A fim de comparar o desempenho do método MetaStream para as abordagens de seleção de algoritmos SelLote e SelUnit, os mesmos meta-dados de treinamento e teste

para cada instante de tempo devem ser usados na indução e avaliação do meta-modelo. Porém, a caracterização dos dados é distinta para cada abordagem, resultando em meta-dados diferentes, que variam em relação ao número de meta-exemplos e meta-atributos que descrevem os dados (Seção 5.3.1).

Em SelLote, embora fosse possível utilizar qualquer valor para γ, tal que $\gamma > 1$ ao longo do fluxo de dados, optou-se por fixá-lo a priori para cada conjunto de dados, pois o valor ideal não é conhecido. Assim, a quantidade de exemplos do conjunto de seleção foi definido de acordo com os experimentos para ajuste de parâmetros, apresentado na Seção 5.4. Uma outra possibilidade seria utilizar técnicas de detecção de mudanças para informar quando o algoritmo atual deve ser substituído, mas essas técnicas poderiam influenciar na avaliação dos métodos de seleção, dificultando a análise dos resultados, . como discutido nos capítulos anteriores.

Como não é possível utilizar os mesmos meta-exemplos para treinamento e teste dos meta-modelos, pois as diferenças na caracterização dos dados é algo intrínseco de cada abordagem, é importante garantir que a indução e o teste dos meta-modelos para cada instante de tempo seja realizado para ambas as abordagens usando meta-exemplos que foram gerados a partir dos mesmos dados do nível base. Por exemplo, na Figura 4.8, observa-se que para os mesmos dados do nível base um único meta-exemplo é gerado para a abordagem SelLote, enquanto oito meta-exemplos são gerados para a abordagem SelUnit. Generalizando, para cada γ exemplos do nível base, um meta-exemplo é gerado para SelLote, enquanto γ meta-exemplos são gerados para SelUnit.

Devido ao maior número de meta-exemplos gerados para a abordagem SelUnit, o custo computacional é evidentemente maior do que o de SelLote. Para tentar reduzir esse custo, uma amostragem aleatória é realizada para cada conjunto de treinamento, respeitando a distribuição das classes. A taxa de amostragem foi definida arbitrariamente em 20%. Portanto, se $\omega_m = 100$ para SelLote, para SelUnit, $\omega_m = 0.2 \times 100 \times \gamma$ meta-exemplos são utilizados em cada janela de treinamento e $\lambda_m = \gamma$ meta-exemplos são testados. Outras técnicas de amostragem que não a aleatória poderiam adicionar algum viés ao processo, o que não é desejado para a comparação das abordagens SelLote e SelUnit que é realizada neste trabalho. Por exemplo, eliminar os meta-exemplos que foram rotulados com uma pequena confiança de que o desempenho dos regressores era consistentemente diferente, ou seja, quando não há um ganhador claro, poderia beneficiar a abordagem SelUnit. O desenvolvimento e avaliação de técnicas para esse fim fica para trabalho futuro.

A comparação entre as duas abordagens é trivial no nível base, pois o método de avaliação empregado garante que os meta-exemplos usados no processo de treinamento e teste dos meta-modelos são gerados a partir dos mesmos exemplos do nível base. Porém, no nível meta, o número de meta-exemplos avaliados são diferentes. Na abordagem SelUnit, um meta-modelo prediz a classe para cada γ meta-exemplos, que correspondem a um único meta-exemplo na abordagem SelLote. Para que a comparação entre as duas

abordagens seja mais próxima possível do objetivo final, que é a redução do erro no nível base, o processo de avaliação dos resultados com a abordagem SelLote é adaptado. A predição de cada meta-exemplo na abordagem SelLote é replicada γ vezes, de modo que essas predições possam ser confrontadas com as classes reais dos meta-exemplos da abordagem SelUnit. O método de avaliação para comparação entre SelLote e SelUnit é detalhada na Seção 6.2.

5.3.5 Avaliação

Para a avaliação do método MetaStream para a tarefa de seleção de algoritmos nas Seções 6.1 e 6.3, considerou-se que o custo de uma classificação errônea é idêntico para todas as classes (algoritmos) sob análise. A taxa de erro é uma medida adequada nesse cenário (Hernández-Orallo et al., 2012; Japkowicz e Shah, 2011), além de ser comumente usada na área de FCD com o método *prequential* ou *holdout* (Gama et al., 2013), conforme discutido no Capítulo 2. Assim, o desempenho preditivo dos métodos MetaStream e Default no nível meta é dado pela taxa de erro para todos os m meta-exemplos, conforme a Equação 5.1, em que L é a função de perda 0-1. Com o objetivo de verificar se as diferenças entre os métodos MetaStream e Default para cada par de algoritmos são significativas com 95% de confiança, o teste de McNemar foi aplicado. Para avaliar se um método é significativamente melhor que o outro considerando todos os pares de algoritmos, o teste de Wilcoxon (Demšar, 2006) com 95% de confiança foi utilizado. A hipótese nula h_0 é que os desempenhos preditivos obtidos pelos dois métodos são equivalentes.

$$\text{Taxa de erro} = \frac{1}{m}\sum_{i=1}^{1} L(y_i, \hat{y}_i) \tag{5.1}$$

Para comparar o desempenho do método MetaStream para as abordagens SelLote e SelUnit, a estatística κ, de Cohen (1960), é utilizada neste trabalho. Essa medida é adequada para comparar dados que possuem distribuições de classes distintas, o que pode ocorrer na comparação das abordagens SelLote e SelUnit, pois são gerados diferentes conjuntos de meta-dados. Essa estatística é definida segundo a Equação 2.8, em que p_0 é a acurácia do classificador e p_c é a probabilidade devida à chance, que pode ser obtida a partir da matriz de confusão, exibida na Tabela 2.2 para um problema de duas classes: positiva e negativa. Essa estatística κ foi preferida em relação a outras, como a estatística π de Scott (1955), pois, segundo (Japkowicz e Shah, 2011), é mais adequada quando o objetivo é comparar as predições de um classificador com os valores reais do atributo alvo. Além disso, essa estatística é utilizada no Weka (Witten e Frank, 2005) e, recentemente, serviu de base para a proposta de uma nova medida em Bifet et al. (2013), denominada Kappa Plus (ou κ^+), que subtrai a acurácia do classificador proposto p_0 da acurácia de um classificador simples p_c, que utiliza a classe do último exemplo rotulado para predizer

a classe do próximo exemplo não rotulado (teste). Nesta tese, o κ^+ não é utilizado, mas a comparação com o método Default permite uma avaliação similar à realizada em Bifet et al. (2013).

Para avaliar os rankings de algoritmos gerados com as predições realizadas pelos métodos MetaStream e Default (Seção 6.4), a similaridade entre os rankings gerados (preditos) e os rankings verdadeiros é medida pelo coeficiente de correlação de Spearman ρ (Neave e Worthington, 1992), que também foi utilizado em diversos trabalhos com o mesmo propósito (Brazdil et al., 2003; Soares, 2004; De Souto et al., 2008; Souza, 2010; Kanda, 2012). Seja $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ um vetor de n valores, um ranking desses valores é definido como um vetor $R = (R_1, R_2, \ldots, R_n)$, em que R_i é o ranking do item X_i, $i = \{1, \ldots, n\}$. A correlação de Spearman entre um ranking predito $\hat{R}$ e um ranking verdadeiro R é definido pela Equação 5.2. A correlação de Spearman está relacionada com a MSE, pois basicamente calcula a média normalizada dos erros quadráticos do ranking recomendado em relação ao ranking verdadeiro. Quanto maior é a similaridade entre os dois rankings, maior o coeficiente de Spearman, que pode assumir valores no intervalo $[-1, 1]$. Se os dois rankings possuem uma tendência de crescimento ou declínio, a correlação é positiva, com valor máximo de $\rho = 1$. Se os rankings possuem tendências opostas, a correlação é negativa, com valor mínimo de $\rho = -1$ (completamente invertidos). Quando não há nenhuma correlação entre eles, o coeficiente de Spearman é $\rho = 0$.

$$r_s = 1 - \frac{6 \sum_{i=1}^{n} (\hat{R}_i - R_i)^2}{n^3 - n} \tag{5.2}$$

Além de comparar os métodos MetaStream e Default no nível meta, é importante avaliar se a seleção de algoritmos realizada por esses métodos proporcionou melhorias de desempenho do sistema de aprendizado no nível base. Para isso, o algoritmo selecionado para cada instante de tempo é utilizado para predizer a saída desejada dos respectivos exemplos no nível base. A medida de erro NMSE é calculada para todos os exemplos do nível base para os quais um algoritmo foi selecionado no nível meta. O desempenho dos métodos de seleção de algoritmos no nível base é comparado a uma outra abordagem que consiste em sempre utilizar a média das predições de todos os modelos sob análise para predizer o atributo alvo, ao invés de utilizar apenas a predição de um único modelo. Essa abordagem pode ser entendida como um *ensemble* com uma estratégia de combinação simples, em que todos os modelos têm o mesmo peso. Por isso, essa abordagem é denominada de **Ensemble** no restante desta tese. Para possíveis estratégias de composição de *ensembles* para problemas de regressão vide Mendes-Moreira et al. (2012) e as referências ali contidas. O teste estatístico de Friedman (Demšar, 2006) é aplicado para verificar se há, com 95% de confiança, evidências suficientes para rejeitar a hipótese nula h_0 de que os valores de NMSE obtidos com as previsões a nível base por meio dos métodos MetaStream, Default e Ensemble são equivalentes. Caso a hipótese nula seja rejeitada,

o pós-teste de Holm (Demšar, 2006) é aplicado para encontrar quais são as diferenças significativas.

5.4 Ajuste de parâmetros

Alguns parâmetros mencionados nas seções anteriores podem influenciar no desempenho dos métodos de seleção de algoritmos. Para ajustar os valores desses parâmetros, são realizados experimentos usando o método testar-então-treinar com uma janela deslizante, conforme o esboço apresentado na Figura 5.1. Os primeiros meta-exemplos do fluxo de dados são usados para o ajuste de parâmetros (treinamento e validação), enquanto que os meta-exemplos seguintes são usados para treinamento e teste. Portanto, os dados usados para validação nunca são usados para teste e nunca há sobreposição entre os dados de treinamento e validação/teste para um instante de tempo. Com isso, garante-se que a afinação dos parâmetros não provoca *overfitting*. Para reduzir o custo computacional, os experimentos são realizados apenas com os meta-dados gerados com a estratégia de rotulação Combinação, pois conjectura-se que essa estratégia obtenha melhores resultados no nível base, visto que é possível selecionar uma combinação de algoritmos além de cada algoritmo separadamente.

Os primeiros parâmetros investigados são o tamanho da janela de treinamento (ω_m) e do conjunto de seleção de algoritmos (γ), que podem influenciar diretamente no desempenho dos métodos MetaStream e Default. Se os dados são razoavelmente estáveis, ou seja, não mudam constantemente, uma janela maior é mais adequada, pois haverá uma quantidade maior de exemplos para treinamento dos modelos. Por outro lado, se esses dados mudam rapidamente, uma janela menor é mais apropriada, pois evita exemplos do antigo conceito no conjunto de treinamento do novo conceito. Evidentemente, a taxa dessas mudanças pode variar ao longo do tempo, variando também o tamanho ideal da janela e do conjunto de seleção. Porém, isso não invalida o estudo realizado neste trabalho, pois o objetivo é melhorar o desempenho relativo do sistema de aprendizado no nível base, e não o absoluto. Para que a comparação dos métodos MetaStream e Default seja a mais justa possível, deve-se garantir que os mesmos meta-exemplos sejam usados por ambos os métodos, evitando que a diferença de desempenho entre eles possa ser atribuída ao uso de diferentes dados. Portanto, é necessário que seja definido um único tamanho para os parâmetros ω_m e γ_m.

O método MetaStream também é sensível ao algoritmo de classificação utilizado no nível meta. Assim, o desempenho dos algoritmos RF e SVM como meta-aprendizes também são analisados durante os experimentos de ajuste de parâmetros. As diferenças observadas entre os dois algoritmos na Seção 5.4.1 levou à realização de experimentos para seleção de meta-atributos, como será apresentado na Seção 5.4.2.

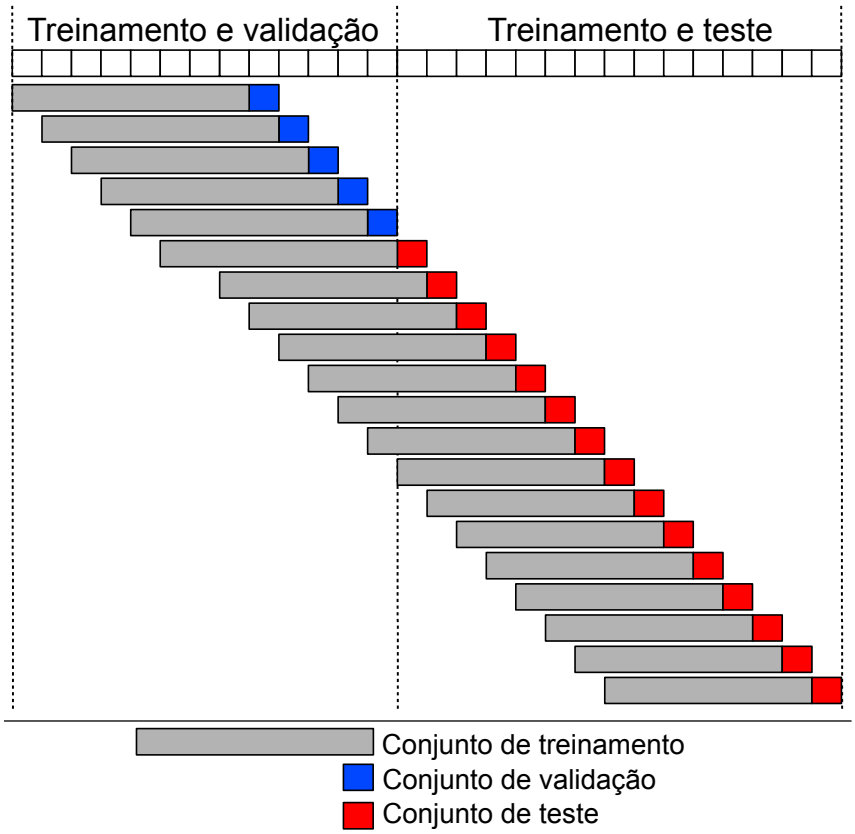


Figura 5.1: Método testar-então-treinar para a realização de experimentos de ajuste de parâmetros (validação) e avaliação dos métodos de seleção de algoritmos (teste).

5.4.1 Tamanho dos conjuntos de treinamento e de seleção de algoritmos

O número de meta-exemplos disponíveis para validação e teste varia de acordo com o tamanho dos conjuntos de treinamento e de seleção de algoritmos. Porém, independentemente desse tamanho, os métodos são sempre avaliados para o mesmo conjunto de validação e teste sob a perspectiva do nível base, garantindo uma comparação justa entre os diferentes tamanhos dos conjuntos de dados. Portanto, para o ajuste dos parâmetros, os métodos são avaliados somente no nível base.

Os diferentes tamanhos investigados para a janela de treinamento foram baseados na quantidade mínima de meta-exemplos necessários para induzir adequadamente o meta-modelo e na quantidade máxima que permitisse que um número razoável de meta-exemplos pudessem ser utilizados para teste, ou seja, quanto maior for o número de meta-exemplos, maior é a confiança nos resultados empíricos. Janelas muito maiores não foram testadas, pois são mais propícias a incluírem dados desatualizados, isto é, possivelmente gerados com conceitos antigos. O tamanho do conjunto de seleção de algoritmos foi definido de modo a balancear a qualidade das estimativas dos valores dos meta-atributos, ou seja, quanto maior, melhor, e a variância do desempenho dos regressores, isto é, quanto menor

a janela, normalmente, maior é a diferença entre os regressores. Outro aspecto considerado para a escolha dos valores investigados foi a periodicidade dos dados. Para os conjuntos de dados TTP e Airline, o número de exemplos gerados para cada dia é variável, mas para o EDP, um exemplo é coletado a cada 30 minutos, e, portanto, são produzidos 24 exemplos por dia.

Nos experimentos de ajuste de parâmetros, a janela de treinamento no nível meta foi testada com três tamanhos $\omega_m = \{100, 200, 300\}$ e o conjunto de seleção com $\gamma = \{10, 20\}$ para os conjuntos de dados TTP e Airline e $\gamma = \{12, 24\}$ para o EDP. Como a maior janela de treinamento investigada possui 300 meta-exemplos, nos experimentos com $w_m = 100$ e $w_m = 200$, os primeiros 200 e 100 meta-exemplos, respectivamente, são descartados. Assim, os mesmos dados de validação são usados para os três tamanhos de janelas investigados. Nos experimentos utilizando o menor conjunto de seleção ($\gamma = 10$ ou $\gamma = 12$), os primeiros 1000 meta-exemplos (10 000 ou 12 000 exemplos no nível base, respectivamente) foram usados para validação, enquanto que, quando o maior tamanho é avaliado ($\gamma = 20$ ou $\gamma = 24$), os primeiros 500 meta-exemplos (10 000 ou 12 000 exemplos no nível base, respectivamente) foram usados para validação. Por uma questão de simplicidade, os tamanhos testados para o conjunto de seleção são múltiplos do tamanho do conjunto de teste (η_b).

Para cada combinação dos valores dos parâmetros ω_m e γ, calculou-se o desempenho preditivo médio de cada método sobre todos os pares de regressores. Os valores de parâmetro que maximizam o desempenho médio de todos os métodos sob análise serão escolhidos para serem utilizados nos próximos experimentos. Evidentemente, os melhores valores de ω_m e γ podem ser diferentes para cada método testado. Nesse caso, serão escolhidos os valores que resultam em perdas de desempenho semelhantes em relação à melhor combinação de parâmetros. O desempenho médio de cada método para cada combinação de parâmetros k é dado pela média dos valores de NMSE para todos os J pares de regressores: $N\bar{M}SE = \frac{1}{J}\sum_{j=1}^{J} NMSE(j)$. Os resultados obtidos para cada combinação de parâmetros ω_m/γ são mostrados nas Tabelas 5.4, 5.5 e 5.6 para os conjuntos de dados TTP, EDP e Airline, respectivamente. Os métodos MetaStream-RF e MetaStream-SVM representam o método MetaStream com o meta-aprendiz RF e SVM, respectivamente. Os menores erros para cada método são destacados em negrito.

Analisando a Tabela 5.4, é possível observar que as combinações de parâmetros que resultaram nas menores taxas de erro para o conjunto de dados TTP são diferentes para cada método. Porém, em geral, os métodos não foram muito afetados por esses valores, sendo que o Default foi mais robusto (isto é, com menor variância do erro) do que o MetaStream, embora tenha tido piores resultados. Outra observação importante é que a taxa de erro dos métodos Default e MetaStream-SVM são muito similares. Na verdade, o segundo método, na maioria das vezes, apenas predisse a classe majoritária, como o método Default faz. Portanto, os valores dos parâmetros serão escolhidos considerando apenas

os métodos Default e MetaStream-RF. O primeiro obteve o melhor desempenho para as combinações 100/20 e 300/20, enquanto que o segundo obteve o melhor desempenho para a combinação 300/10. Como a diferença de desempenho do método MetaStream-RF para as combinações 300/10 e 300/20 é muito pequena e a segunda coincide com a melhor combinação para o método Default, ela foi escolhida para os próximos experimentos com o conjunto de dados TTP.

Tabela 5.4: NMSE dos métodos Default e MetaStream (usando RF e SVM como meta-aprendizes) para cada combinação de ω_m/γ para o conjunto de dados TTP.

Método \ ω_b/γ	100/10	100/20	200/10	200/20	300/10	300/20
Default	0.616	**0.612**	0.614	0.613	0.614	**0.612**
MetaStream-RF	0.605	0.601	0.597	0.597	**0.595**	0.596
MetaStream-SVM	0.615	**0.611**	0.614	0.612	0.614	0.612

A escolha da melhor combinação de valores dos parâmetros ω_m/γ para o conjunto de dados EDP (Tabela 5.5) é trivial, pois todos os métodos conseguiram os melhores resultados com os tamanhos 300/24. O método MetaStream-SVM obteve um desempenho médio similar ao Default para todas as combinações de parâmetros, pois, assim como aconteceu para o conjunto de dados TTP, o método MetaStream-SVM, na maioria das vezes, predisse a classe majoritária. Em geral, os métodos foram mais sensíveis à variação do tamanho do conjunto de seleção do que ao tamanho da janela de treinamento.

Tabela 5.5: Média dos valores de NMSE dos métodos Default e MetaStream (usando RF e SVM como meta-aprendizes) para cada combinação de ω_m/γ para o conjunto de dados EDP.

Método \ ω_b/γ	100/12	100/24	200/12	200/24	300/12	300/24
Default	0.273	0.260	0.272	0.263	0.269	**0.254**
MetaStream-RF	0.266	0.251	0.255	0.251	0.260	**0.249**
MetaStream-SVM	0.274	0.258	0.272	0.263	0.268	**0.254**

A partir dos resultados para o conjunto de dados Airline, apresentados na Tabela 5.6, pode-se verificar que, assim como no conjunto de dados EDP, os métodos são mais sensíveis ao tamanho do conjunto de seleção do que ao de treinamento. Novamente, o método MetaStream-SVM obteve desempenho similar ao do Default e ambos os métodos conseguiram o melhor desempenho com a combinação de parâmetros 300/20. Contudo, essa configuração de parâmetros não foi a mais adequada para o método MetaStrem-RF. Esse método conseguiu o menor NMSE médio para a combinação 200/10, que corresponde ao maior valor para o Default. Devido a essa discordância, a combinação de valores 100/10 foi escolhida, pois resultou na menor perda de desempenho para os dois métodos (0.06 para

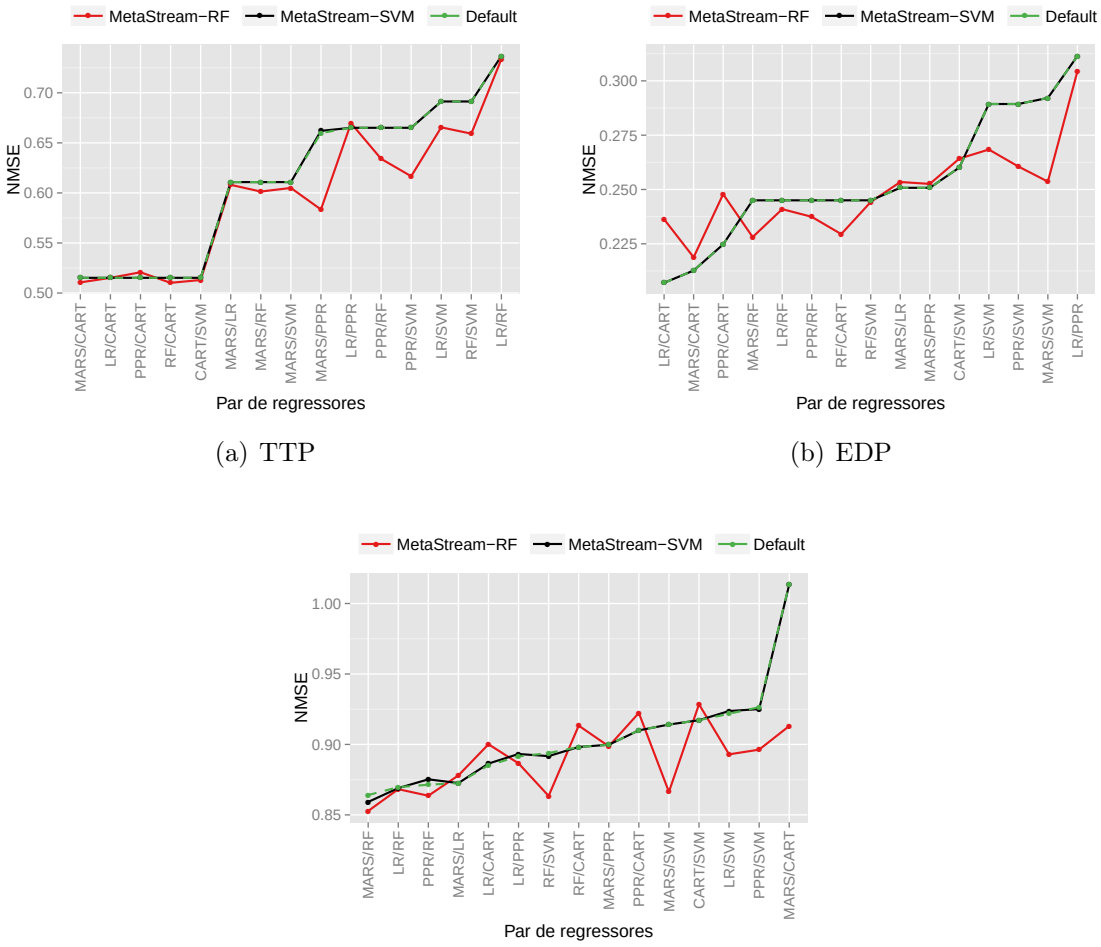
o método Default e 0.05 para o método MetaStream-RF). Qualquer outra combinação resulta em maiores perdas para um dos métodos.

Tabela 5.6: Ranking médio dos métodos Default e MetaStream (usando RF e SVM como meta-aprendizes) para cada combinação de ω_m/γ para o conjunto de dados Airline.

Método \ ω_b/γ	100/10	100/20	200/10	200/20	300/10	300/20
Default	0.903	0.901	0.906	0.902	0.904	**0.897**
MetaStream-RF	0.890	0.898	**0.885**	0.897	0.891	0.895
MetaStream-SVM	0.903	0.901	0.906	0.901	0.903	**0.897**

Os resultados obtidos no nível base com os algoritmos selecionados pelos métodos MetaStream e Default para os dados de validação usando os valores ω_m e γ ajustados anteriormente podem ser vistos nos gráficos da Figura 5.2, para cada conjunto de dados e par de regressores. Os pares de regressores estão em ordem crescente dos valores de NMSE obtidos pelo método Default. Um aspecto importante que pode ser observado nos gráficos dos três conjuntos de dados é a sobreposição dos resultados dos métodos Default e MetaStream-SVM. Isso significa que o meta-aprendiz SVM não foi capaz de mapear as características que descrevem os dados com o desempenho dos regressores. Assim, para a maioria dos pares de regressores, esse método predisse a classe majoritária, como o Default faz. Ao contrário do meta-aprendiz SVM, quando o meta-aprendiz RF é usado, pode-se notar uma vantagem desse em relação ao método Default e, consequentemente, ao MetaStream-SVM para a maior parte dos pares de regressores para os três conjuntos de dados.

O método MetaStream-SVM pode ter apresentado comportamento semelhante ao Default por diversos motivos. A primeira hipótese é de que as características que descrevem os dados não são suficientes para predizer o melhor regressor. Apesar de plausível, os resultados preliminares indicam que o método MetaStream-RF foi capaz de fazer esse mapeamento, contradizendo essa hipótese. Outra possibilidade para explicar a diferença de comportamento entre os dois meta-aprendizes é a capacidade do RF em lidar com atributos irrelevantes e redundantes, pois possui um mecanismo de seleção de atributos embarcado, e é bastante robusto a *overfitting* (Breiman, 2001). As SVMs possuem mecanismos para evitar o *overfitting* quando lidam com dados de grande dimensionalidade (Statnikov et al., 2008; Chapelle et al., 2002; Burges, 1998; Joachims, 1998), mas os resultados experimentais apresentados em alguns estudos, como em Vanschoren (2010) e Ben-Hur et al. (2008), e os resultados obtidos nesta tese, apontam que esses mecanismos não foram suficientes nesses casos. Essa hipótese é avaliada na Seção 5.4.2 com o uso de um método de filtro para seleção de meta-atributos.



(a) TTP (b) EDP

(c) Airline

Figura 5.2: Valores de NMSE de validação obtidos pelos métodos de seleção de algoritmos para os valores de ω_m e γ selecionados para cada conjunto de dados.

5.4.2 Seleção de meta-atributos

Idealmente, o ajuste do tamanho dos conjuntos de treinamento e de seleção deveria ser feito concomitantemente com a seleção de meta-atributos, pois não há garantias de que os melhores valores escolhidos para os parâmetros para o conjunto completo de meta-atributos sejam também os melhores para um subconjunto de meta-atributos (Daelemans et al., 2003). Porém, dado o custo computacional de executar tais experimentos, decidiu-se realizar a seleção de meta-atributos apenas para os melhores valores escolhidos na seção anterior. Além disso, mesmo que os experimentos fossem feitos para todas as combinações de parâmetros, não haveria garantia de que a melhor escolha estaria sendo feita, pois o tamanho ideal pode variar ao longo do tempo.

Os mesmos dados de validação da seção anterior foram utilizados para avaliar se a seleção de meta-atributos pode melhorar o desempenho dos meta-aprendizes, conforme relatam alguns trabalhos da área de meta-aprendizado, como Todorovski et al. (2000),

Kalousis e Hilario (2001) e Kanda (2012). Neste trabalho, o método de seleção de atributos ReliefF (Kononenko, 1994) foi utilizado para esse fim, devido ao seu sucesso em muitas áreas e por ser capaz de detectar a dependência condicional entre atributos (Robnik-Šikonja e Kononenko, 2003; Kalousis et al., 2007). Uma breve introdução sobre seleção de atributos e o método ReliefF pode ser encontrada no Apêndice B.

Uma questão que surge com o uso do ReliefF e de outros métodos que medem a importância dos atributos é como selecionar um subconjunto desses atributos com base nos pesos calculados. Como destaca Tuv et al. (2009), os resultados da seleção de um subconjunto de atributos nesses casos é fortemente dependente do limiar que é usado para descartar os atributos menos relevantes, isto é, com menor peso. Como o algoritmo ReliefF é eficaz em identificar atributos irrelevantes mas não é capaz de identificar atributos redundantes, uma quantidade razoável de atributos pode ser mantida, dependendo do limiar utilizado. Para ajustar o valor desse parâmetro, foram realizados experimentos com diferentes limiares: 0.01, 0.03, 0.05, 0.1 e 0.15. Hall (2000) testou vários desses valores e sugeriu o limiar 0.01, usado também por (Moreira, 2008). O cálculo da importância e a seleção dos meta-atributos é realizada continuamente para cada janela de treinamento do nível meta, pois o subconjunto ótimo de meta-atributos pode variar ao longo do tempo.

Na Tabela 5.7 são mostradas as reduções (valores negativos) ou aumentos (valores positivos) do NMSE médio para o subconjunto de meta-atributos selecionado em comparação com o conjunto completo de meta-atributos para cada limiar considerado. Como é possível observar por essa tabela, o meta-aprendiz SVM foi consideravelmente beneficiado com a seleção de meta-atributos para os conjuntos de dados TTP e Airline, principalmente quando os menores limiares foram usados. Por outro lado, o desempenho do meta-aprendiz RF normalmente piorou para todos os conjuntos de dados, principalmente quando um grande número de meta-atributos eram descartados (grandes valores do limiar). Esse comportamento não é surpreendente, pois o algoritmo RF constrói centenas ou milhares de árvores usando diferentes subconjuntos de meta-atributos selecionados aleatoriamente. Reduzindo a quantidade de meta-atributos, reduz-se também a diversidade das árvores geradas, que é um fator crucial para o sucesso desse algoritmo.

Como o meta-aprendiz RF geralmente piorou com a redução do número de atributos, enquanto o meta-aprendiz SVM geralmente melhorou, comparou-se o desempenho do primeiro usando o conjunto completo de meta-atributos com o segundo após a seleção de meta-atributos. Para o meta-aprendiz SVM, foram escolhidos os limiares que resultaram nas maiores reduções do NMSE para cada conjunto de dados: 0.01, 0.03 e 0.03, para os conjuntos de dados TTP, EDP e Airline, respectivamente. Para esse último conjunto, a redução do NMSE usando os limiares 0.01 e 0.03 foi a mesma, sendo que o limiar de 0.03 foi escolhido por descartar um número maior de meta-atributos. Os gráficos apresentados na Figura 5.3 mostram o NMSE para cada par de regressores usando essas configurações. Comparando os resultados dos métodos MetaStream-SVM e Default para

Tabela 5.7: Diferença do NMSE médio, calculado sobre todos os pares de regressores, para cada limiar quando o conjunto completo de meta-atributos e o subconjunto selecionado são usados.

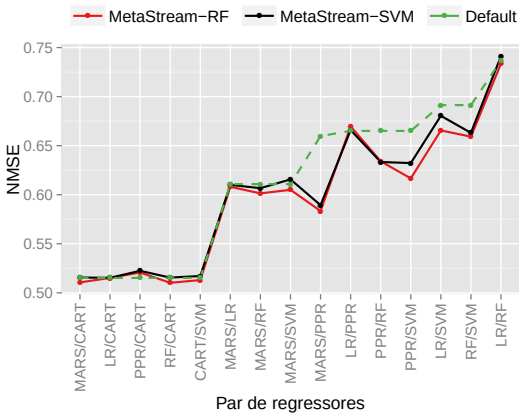
Conjunto de dados	**Meta-aprendiz**	**Limiares**				
		0.01	**0.03**	**0.05**	**0.1**	**0.15**
TTP	RF	**0.015**	0.069	0.161	0.203	0.189
	SVM	**-0.161**	-0.134	-0.114	-0.128	-0.123
EDP	RF	**0.025**	0.123	0.158	0.061	0.044
	SVM	0.041	**0.026**	0.055	0.082	0.082
Airline	RF	-0.005	**-0.013**	0.047	0.145	0.109
	SVM	**-0.199**	**-0.199**	-0.090	-0.056	-0.055

os três conjuntos de dados, nota-se que a seleção de atributos evitou o *overfitting* do meta-aprendiz SVM. Apenas para o conjunto de dados EDP, em que a seleção de meta-atributos produziu pouco efeito, é que o SVM ainda está similar ao Default. Nota-se também que, apesar da melhora consistente desse meta-aprendiz, o RF ainda apresentou, em geral, os melhores resultados. Isso não ocorreu apenas para o conjunto de dados Airline, em que ambos tiveram desempenhos semelhantes. Porém, como a seleção de atributos impõe uma etapa a mais no processo de seleção de algoritmos, decidiu-se utilizar sempre o método MetaStream com o meta-aprendiz RF nos próximos experimentos com os dados de teste.

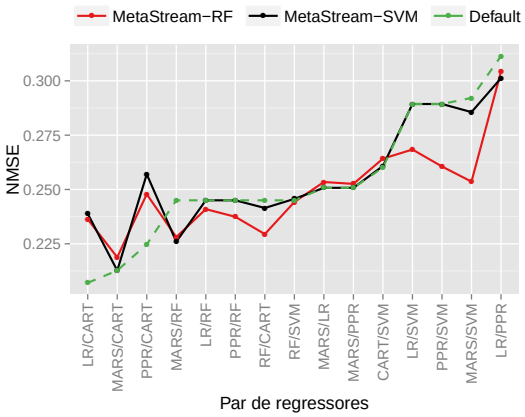
O melhor tamanho da janela de treinamento (ω_m) e do conjunto de seleção de algoritmos (γ) para os experimentos com os dados de validação são apresentados na Tabela 5.8 para cada conjunto de dados. Os valores de ω_m para SelUnit foram ajustados conforme a taxa de amostragem de 20%, discutida na Seção 5.3.4). Nesta tabela também pode-se visualizar o melhor meta-aprendiz e se a seleção de meta-atributos é ou não realizada. Como mencionado anteriormente, não é garantido que os valores selecionados com esses experimentos sejam os melhores possíveis, pois, idealmente, todos os parâmetros deveriam ser ajustados concomitantemente e dinamicamente ao longo do tempo. Porém, como o objetivo deste trabalho é realizar um estudo comparativo do desempenho relativo dos métodos e não do desempenho absoluto, o processo de ajuste utilizado avalia todos os métodos em condições tão semelhantes quanto possível, para além das diferenças intrínsecas entre eles, e propicia um custo computacional muito menor.

Tabela 5.8: Resumo dos parâmetros selecionados para a avaliação experimental.

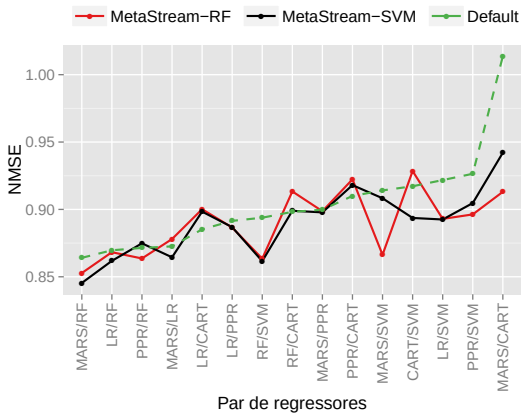
Conjunto de dados	Meta-aprendiz	Seleção de meta-atributos	SelLote		SelUnit	
			ω_m	γ	ω_m	γ
TTP	RF	Não	300	20	1200	1
EDP	RF	Não	300	24	1440	1
Airline	RF	Não	100	10	200	1



(a) TTP



(b) EDP



(c) Airline

Figura 5.3: Valores de NMSE de validação obtidos pelos métodos de seleção de algoritmos com a seleção de meta-atributos para o meta-aprendiz SVM.

5.5 Considerações finais

Neste capítulo foram apresentados os materiais e método envolvidos na avaliação experimental das hipóteses estabelecidas nesta tese. Inicialmente, os três conjuntos de dados reais usados nos experimentos foram descritos. Em seguida, foram apresentados os algoritmos de regressão e os parâmetros empregados no processo de aprendizado e teste desses algoritmos no nível base. O processo de caracterização e rotulação dos dados para as hipóteses investigadas foram abordados na seção seguinte. Adicionalmente, os meta-aprendizes e o processo de seleção de algoritmos para lotes de exemplos ou unitária e o método de avaliação desses experimentos foram descritos na mesma seção. Por último, os parâmetros envolvidos no método MetaStream e no processo de seleção de algoritmos foram ajustados de acordo com experimentos preliminares, para os quais foram utilizados o período inicial dos conjuntos de dados investigados. No próximo capítulo são apresentados e analisados os resultados experimentais dessa avaliação.

Capítulo 6

Resultados Experimentais

Neste capítulo, são apresentados e analisados os resultados da avaliação experimental do método MetaStream a fim de obter evidências sobre as hipóteses estabelecidas neste trabalho. As análises são realizadas para os dois níveis de aprendizado. No nível meta, o objetivo é avaliar o desempenho preditivo dos métodos de seleção de algoritmos, enquanto no nível base o objetivo é avaliar o impacto da seleção de algoritmos no desempenho preditivo do sistema de aprendizado. Os resultados experimentais são apresentados apenas para uma estratégia de rotulação dos meta-exemplos, a fim de facilitar a exposição e evitar a repetição de algumas discussões. Dependendo do desempenho dos métodos, alguns resultados para outras estratégias também serão apresentados.

A relação entre os níveis meta e base dependem, dentre outros fatores, das medidas de avaliação e das estratégias de rotulação dos meta-exemplos. Nesta tese, as estratégias de rotulação dos meta-exemplos e, consequentemente, o aprendizado e a avaliação empregados no nível meta, consideram a comensurabilidade das diferenças do nível base, mas apenas qualitativamente, pois a magnitude absoluta é ignorada. Por isso, os resultados observados nos níveis meta e base podem ser discordantes. Por exemplo, considere a abordagem de seleção SelUnit com três meta-exemplos $\{e_1, e_2, e_3\}$, que caracterizam os exemplos do nível base $\{z_1, z_2, z_3\}$, respectivamente, e dois regressores: A e B. O meta-exemplo e_1 é rotulado como "A", uma vez que o desempenho do regressor A é superior ao do regressor B para o exemplo z_1, enquanto que os outros dois meta-exemplos, $\{e_2, e_3\}$, são rotulados como "B" , de acordo com o mesmo critério. Suponha que o erro médio quando o regressor "A" é usado para predizer $\{z_2, z_3\}$ seja menor do que o erro obtido quando o regressor "B" 'é usado para predizer z_1. Suponha ainda que o método MetaStream prediz corretamente os dois meta-exemplos rotulados como "B", mas falha para o meta-exemplo rotulado como "A", enquanto que o Default prediz corretamente apenas o meta-exemplo rotulado como "A". Nesse caso, a taxa de erro do MetaStream (1/3) será menor que a taxa de erro do Default (2/3), mas resultará em um erro maior no nível base, ou seja, o método MetaStream será melhor no nível meta, mas pior no nível base.

Os resultados e as discussões estão organizadas neste capítulo em seções de acordo

com as hipóteses estabelecidas neste trabalho. Na Seção 6.1, investiga-se a hipótese de que o desempenho preditivo do MetaStream é superior ao do método Default. Em seguida, na Seção 6.2, é analisada a hipótese de que a abordagem de seleção de algoritmos SelUnit é capaz de reduzir o erro preditivo médio no nível base em relação a SelLote. Na Seção 6.3, avalia-se a hipótese de que as características dependentes da morfologia dos dados acrescentam informações relevantes para a seleção do melhor algoritmo. Na Seção 6.4, é investigada a hipótese de que a seleção de algoritmos aos pares pode ser usada com sucesso na recomendação de rankings de algoritmos. Por último, na Seção 6.5, são apresentadas as considerações finais sobre este capítulo.

6.1 Comparação entre MetaStream e Default

Nesta seção, o método MetaStream é avaliado empiricamente para a tarefa de seleção de algoritmos para lotes de exemplos e comparado ao método Default nos níveis meta e base. O objetivo desses experimentos é verificar a veracidade da primeira hipótese específica estabelecida nesta tese:

> *O método MetaStream pode ser aplicado para a seleção de algoritmos de aprendizado para problemas de fluxo de dados com desempenho preditivo superior ao do método de referência Default.*

No nível base, os métodos de seleção de algoritmos MetaStream e Default são também comparados com a abordagem Ensemble. Inicialmente, é apresentada uma visão geral do desempenho obtido pelos métodos de seleção de algoritmos no nível meta e, em seguida, os resultados experimentais nos dois níveis de aprendizado são analisados detalhadamente.

6.1.1 Nível meta

O desempenho dos métodos de seleção de algoritmos no nível meta foi avaliado para cada par de algoritmos por meio do teste estatístico de McNemar com 95% de confiança (Gama et al., 2013). Na Tabela 6.1, cada linha representa um par de algoritmos e cada coluna um conjunto de dados, que é dividida em estratégia de rotulação Sem-empate (SE) e de Combinação (CO). Os símbolos utilizados nessa tabela têm os seguintes significados:

- ▲ MetaStream é significativamente melhor que o Default
- △ MetaStream é melhor que o Default, mas não significativamente
- ▼ MetaStream é significativamente pior que o Default
- ▽ MetaStream é pior que o Default, mas não significativamente
- ⊗ MetaStream e Default possuem o mesmo desempenho

Tabela 6.1: Resultado a nível meta do teste estatístico de McNemar com 95% de confiança na avaliação das diferenças dos desempenhos preditivos dos métodos MetaStream e Default para para cada par de regressores considerado as estratégias de rotulação Sem-Empate (SE) e Combinação (CO) e os conjuntos de dados TTP, EDP e Airline.

Par de algoritmos	**TTP**		**EDP**		**Airline**	
	SE	**CO**	**SE**	**CO**	**SE**	**CO**
MARS/LR	▽	▼	▽	▼	▽	▽
MARS/PPR	△	▽	▽	▽	▽	▽
MARS/RF	▼	▼	▲	▲	▲	△
MARS/CART	△	△	▽	▲	▲	△
MARS/SVM	▼	▼	△	△	▽	▼
LR/PPR	△	⊗	▲	△	▲	△
LR/RF	△	△	▽	▽	△	△
LR/CART	▼	▼	△	▽	△	▲
LR/SVM	▲	▲	▼	▼	▽	▼
PPR/RF	△	△	▽	△	△	▲
PPR/CART	▼	▼	▲	△	△	▽
PPR/SVM	▲	▲	▼	▽	▽	▽
RF/CART	▽	▼	▽	△	▲	△
RF/SVM	▲	▲	▲	▲	▼	▼
CART/SVM	▽	▼	▼	▲	▼	▼
▲ / ▼	**3 / 4**	**3 / 7**	**4 / 3**	**4 / 2**	**4 / 2**	**2 / 4**
△ / ▽	5 / 3	3 / 1	2 / 6	5 / 4	4 / 5	5 / 4
⊗	0	1	0	0	0	0

Como é possível observar na Tabela 6.1, o teste estatístico de McNemar rejeitou, com 95% de confiança, a hipótese nula de que os métodos MetaStream e Default possuem desempenhos preditivos semelhantes para aproximadamente metade das comparações realizadas. O método MetaStream foi significativamente melhor que o Default para a maioria dos pares de algoritmos dos conjuntos de dados EDP e Airline com a estratégia de rotulação Sem-empate: 4/3 e 4/2, respectivamente. Por outro lado, com a estratégia Combinação, o método MetaStream foi significativamente pior que o Default para a maioria dos pares de algoritmos dos conjuntos de dados TTP e Airline: 3/7 e 2/4, respectivamente, mas ainda foi significativamente melhor que o Default para o conjunto de dados EDP.

Embora o número de diferenças significativas varie entre as diferentes estratégias de rotulação (SE e CO) para o mesmo conjunto de dados, pode-se verificar que há uma concordância no sentido das setas (para cima ou para baixo). Isso indica que houve aumento ou redução das diferenças entre os métodos MetaStream e Default, mas a relação de desempenho (pior ou melhor) entre eles foi mantida, principalmente para os pares de algoritmos em que a diferença é significativa para uma das estratégias. As únicas exceções ocorreram para os pares de algoritmos CART/SVM e MARS/CART para o conjunto de dados EDP. Devido ao maior número de classes dos conjuntos de meta-dados gerados com

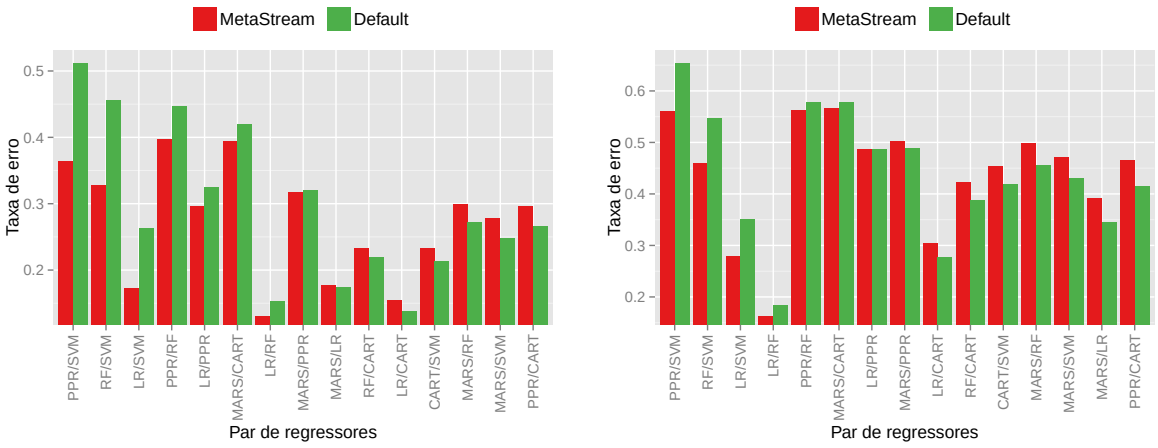
a estratégia Combinação (três classes), supunha-se que a classificação desses dados seria mais complexa do que daqueles gerados com a estratégia Sem-empate (duas classes). Os resultados apresentados evidenciam que tal suposição seja verdadeira para os conjuntos de dados TTP e Airline.

Para avaliar se houve diferença significativa entre os desempenhos dos métodos MetaStream e Default para cada estratégia de rotulação e conjunto de dados considerando todos os pares de algoritmos, o teste estatístico de Wilcoxon foi aplicado com 95% de confiança. Apesar das diferenças observadas com o teste de McNemar para alguns pares de algoritmos, o teste de Wilcoxon não encontrou evidências suficientes para rejeitar a hipótese nula de que os métodos possuem desempenhos comparáveis para nenhuma das comparações realizadas.

Apesar do método MetaStream ter conseguido, em geral, melhor desempenho com a estratégia Sem-empate, os gráficos das taxas de erro (nível meta) e dos valores de NMSE (nível) para os conjuntos de dados EDP e Airline são apresentados a seguir apenas para a estratégia Combinação. Essa decisão deve-se ao fato de que ambos os métodos obtiveram melhores resultados no nível base, que é o objetivo final da seleção de algoritmos, utilizando essa estratégia. Para o conjunto de dados TTP, os gráficos são exibidos para as duas estratégias, para que se possa visualizar e discutir as diferenças entre os resultados obtidos com cada uma delas.

As taxas de erro de classificação dos métodos MetaStream e Default para o conjunto de dados TTP são apresentadas na Figura 6.1 para as estratégias Sem-empate e Combinação. Para os demais conjuntos de dados, as taxas de erro são exibidas apenas para a estratégia Combinação na Figura 6.2. Com o propósito de facilitar a visualização e a interpretação dos gráficos, os pares de algoritmos foram ordenados de acordo com a diferença das taxas de erro de classificação dos métodos MetaStream e Default.

Comparando os métodos MetaStream e Default para o conjunto de dados TTP com os rótulos dos meta-dados definidos pela estratégia Sem-empate (Figura 6.1(a)), é possível observar que as taxas de erro de classificação do método MetaStream são, para a maioria dos pares de regressores, inferiores às do método Default. Ainda assim, esse último foi significativamente melhor que o MetaStream para quatro pares, enquanto o contrário ocorreu para apenas três comparações (Tabela 6.1). Para a estratégia de rotulação Combinação (Figura 6.1(b)), pode-se notar que o Default obteve taxas de erro menores que o MetaStream para uma quantidade ligeiramente maior de pares de algoritmos. Para todos esses pares, a diferença entre os métodos foi significativa, como apontado pelo teste de McNemar. Para os casos em que os erros do MetaStream são menores que o Default, a diferença normalmente é maior do que quando o contrário ocorre. Essa diferença pode ser verificada principalmente para os pares LR/SVM, RF/SVM e PPR/SVM, para os quais o teste estatístico apontou diferença significativa. Observa-se também que o comportamento de ambos os métodos são similares para a maioria dos pares de algoritmos. Como o



(a) TTP - Sem-empate

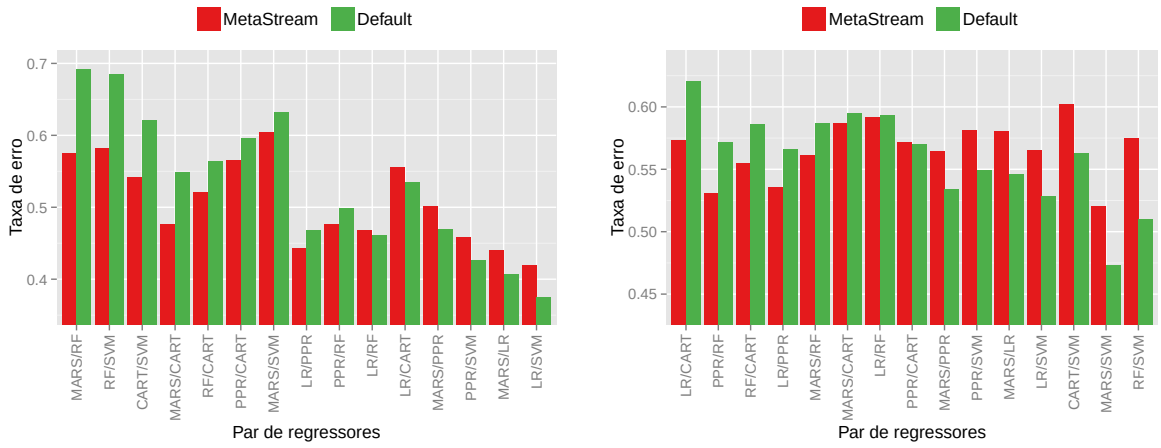
(b) TTP - Combinação

Figura 6.1: Taxas de erro dos métodos de seleção de algoritmos no nível meta para o conjunto de dados TTP usando as estratégias de rotulação Sem-empate e Combinação.

único fator de influência sobre o comportamento do método Default é a distribuição de classes, conjectura-se que esse fator também tenha influenciado o desempenho do método MetaStream. Portanto, a distribuição de classes é analisada detalhadamente mais adiante nesta seção.

Para o conjunto de dados EDP, o desempenho preditivo do método MetaStream é consideravelmente melhor que o do Default para a maioria dos pares de algoritmos, como pode ser visto na Figura 6.2(a). Essa superioridade é visível principalmente para os pares MARS/RF e RF/SVM, em que as diferenças entre os métodos são maiores que 0.1. Em geral, quanto maior é a taxa de erro do método Default, maior também é a vantagem do MetaStream. Apesar disso, é possível notar uma tendência de comportamentos semelhantes para os dois métodos, assim como foi observado para o conjunto de dados TTP. A indução de um meta-modelo para cada par de algoritmos permite analisar se o desempenho de um determinado algoritmo pode ser predito com maior acurácia do que outros. Uma evidência de que isso esteja ocorrendo é quando o método MetaStream é superior ao Default para todos os pares de algoritmos constituídos por um determinado algoritmo. Por exemplo, na Figura 6.2(a), pode-se verificar que o MetaStream é melhor que o Default para todos os pares que são formados com o algoritmo RF, com exceção de LR/RF. O mesmo acontece para o algoritmo CART, com a exceção do par LR/CART. Esses resultados fornecem evidências de que os meta-modelos são capazes de predizer o desempenho desses algoritmos com maior acurácia do que o Default.

Diferentemente do que observou-se para os conjuntos de dados TTP e EDP, os métodos MetaStream e Default possuem comportamentos distintos para o conjunto de dados Airline, como pode ser visto na Figura 6.2(b). Entretanto, deve-se notar que o intervalo no eixo das ordenadas é muito menor nesse gráfico do que os intervalos dos gráficos



(a) EDP (b) Airline

Figura 6.2: Taxas de erro dos métodos de seleção de algoritmos no nível meta para os conjuntos de dados EDP e Airline usando a estratégia de rotulação Combinação.

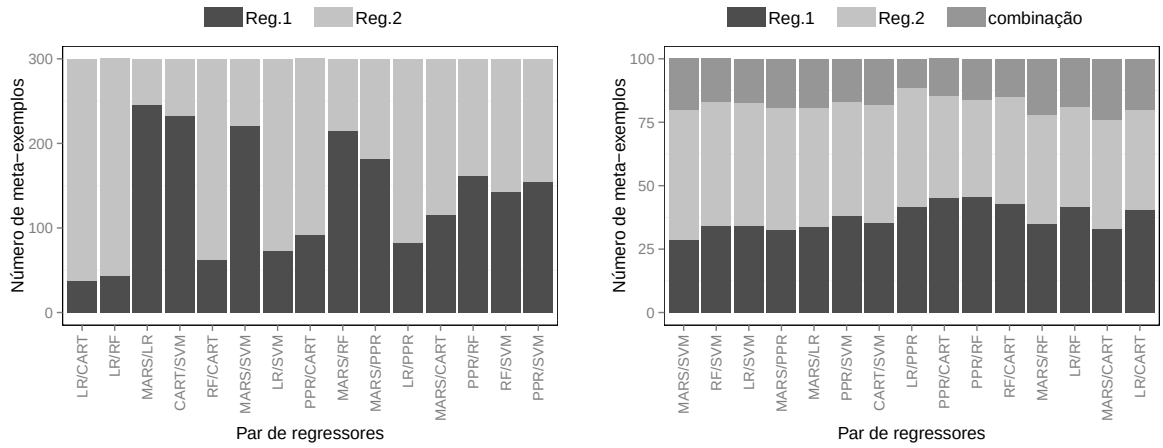
anteriores. O método Default possui menores taxas de erro do que o MetaStream para os primeiros sete pares de algoritmos, sendo que para quatro deles essa diferença é significativa. É interessante notar também que os cinco pares formados com o algoritmo SVM estão entre esses pares. Dos oito restantes, o MetaStream é melhor que o Default para sete deles, mas as diferenças foram significativas apenas para dois casos (PPR/RF e LR/CART).

Como mencionado anteriormente, o comportamento do método Default é guiado pela distribuição de classes do conjunto de treinamento, pois utiliza a classe majoritária para predizer o rótulo de um novo meta-exemplo. As taxas de erro obtidas por esse método nos gráficos anteriores apontam que em alguns pares de algoritmos há um desbalanceamento de classes enquanto em outros as classes parecem estar quase que perfeitamente balanceadas. O método MetaStream, embora em menor grau, também parece ter sido influenciado por esse fator, pois apresentou comportamento similar ao do Default para os conjuntos de dados TTP e EDP. Esses gráficos, no entanto, não são suficientes para analisar a relação entre a distribuição de classes e o desempenho dos métodos de seleção de algoritmos. Portanto, essa questão é investigada a seguir com mais detalhes.

Análise da influência da distribuição de classes no desempenho dos métodos MetaStream e Default

Para auxiliar na análise do comportamento dos métodos de seleção de algoritmos, os gráficos da distribuição média de classes dos dados de treinamento para os conjuntos de dados TTP (rotulação Sem-empate) e Airline (rotulação Combinação) são exibidos na Figura 6.3. O eixo das ordenadas indica o número médio de meta-exemplos de treinamento de cada classe e o eixo das abscissas os pares de algoritmos, que estão em ordem crescente

das taxas de erro do método Default. Nas estratégias Sem-empate e Combinação, um meta-exemplo é rotulado como Reg.1 ou Reg.2 quando o regressor 1 ou o regressor 2, respectivamente, apresentar o melhor desempenho preditivo para os respectivos exemplos base, enquanto que na estratégia Combinação, um meta-exemplo é rotulado como *combinação* quando a média das predições dos regressores é melhor do que qualquer um deles separadamente.



(a) TTP - Sem-empate

(b) Airline - Combinação

Figura 6.3: Média da distribuição de classes dos dados de treinamento para os conjuntos de dados TTP e Airline usando as estratégias Sem-empate e Combinação, respectivamente.

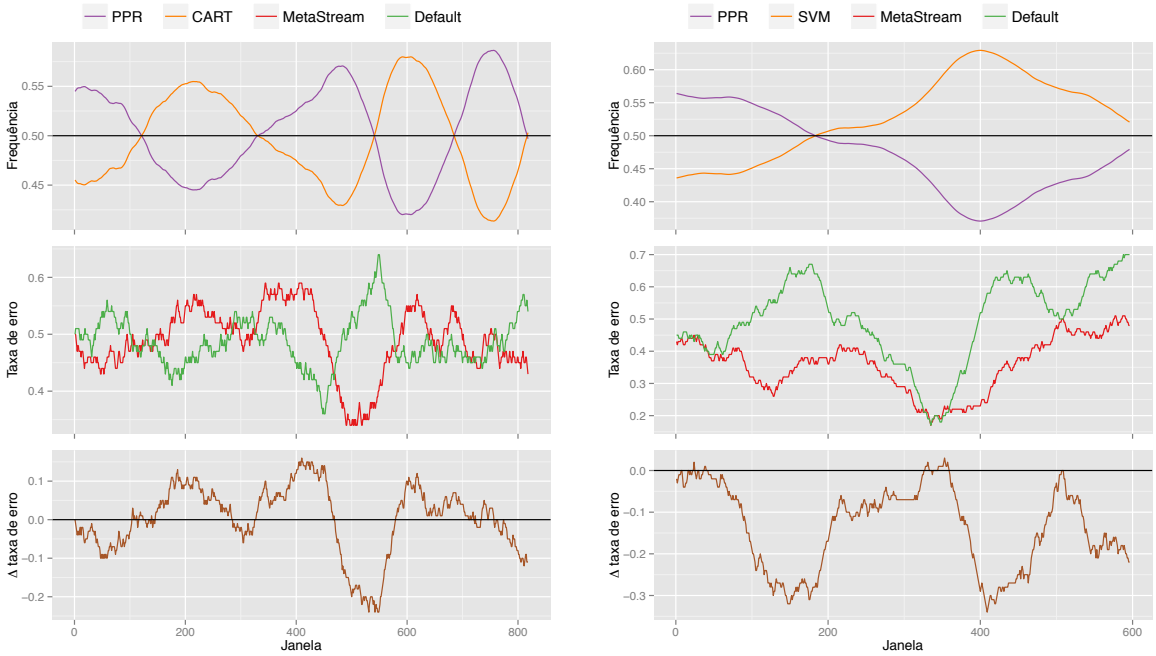
Analisando as figuras 6.1(a) e 6.3(a) conjuntamente, é possível observar que o método Default obteve menores taxas de erro para os pares de algoritmos em que a distribuição de classes é desbalanceada e maiores taxas de erro quando as classes são balanceadas, como seria de esperar. Dessa maneira, os ganhos do método MetaStream em relação ao Default foram maiores, geralmente, quando as classes estão balanceados, que é o caso, por exemplo, dos pares RF/SVM e PPR/SVM. Entretanto, a distribuição de classes não é o único fator que interfere no desempenho do meta-classificador. A sobreposição de classes e o nível de ruído no processo de rotulação dos meta-exemplos, por exemplo, são outros aspectos importantes que influenciam na acurácia de sistemas de aprendizado (Prati et al., 2004; Quinlan, 1986). Portanto, mesmo que haja um perfeito balanceamento entre as classes, o MetaStream pode apresentar um desempenho semelhante ou inferior ao Default para alguns dos problemas de seleção de algoritmos. O contrário também pode ocorrer, caso o problema de classificação não seja complexo e o meta-modelo tenha sido capaz de aprender os padrões das classes minoritárias. Esses dois casos podem ser vistos para os pares de algoritmos MARS/PPR e LR/SVM, respectivamente.

Para o conjunto de dados Airline, a distribuição de classes dos meta-dados rotulados pela estratégia Combinação são mostrados na Figura 6.3(b). Assim, observa-se a distri-

buição de três classes: Reg.1, Reg.2 e *combinação*. A relação entre a distribuição dessas classes e o desempenho dos métodos MetaStream e Default (Fig. 6.2(b)) não é tão clara quanto para os dados TTP. Dado que os pares de algoritmos estão ordenados pelo desempenho preditivo do método Default, esperava-se que as distribuições de classes entre o primeiro e o último par de algoritmos, MARS/SVM e LR/CART, respectivamente, fossem visivelmente distintas, pois a diferença da taxa de erro do método Default para esses pares é maior que 0.2. Porém, não é isso que se observa na Figura 6.3(b). Algo similar aconteceu com o método MetaStream para os pares PPR/CART e PPR/RF. Embora esses pares tenham, praticamente, a mesma distribuição de classes, as taxas de erro obtidas pelo MetaStream são razoavelmente diferentes.

Todavia, antes de afirmar a existência ou inexistência da relação entre taxa de erro e distribuição de classes para os conjuntos de dados TTP e Airline, deve-se lembrar que a distribuição média das classes apresentadas na Figura 6.3 para esses conjuntos de dados é calculada sobre todas as janelas de treinamento do fluxo de dados. Portanto, eventuais variações na distribuição de classes ao longo do tempo não puderam ser visualizadas utilizando apenas a média. Para obter maiores evidências, essa relação é investigada a seguir ao longo do tempo. Essa análise é realizada com os meta-dados rotulados pela estratégia Sem-empate, pois, por se tratarem de problemas de classificação binários, facilitam a visualização e a interpretação dos gráficos, mas o mesmo poderia ter sido feito com os meta-dados rotulados pela estratégia Combinação. Devido ao grande número de pares de algoritmos investigados nesta tese, serão exibidos e discutidos apenas os gráficos necessários para a compreensão das diferentes situações observadas. Cada figura a ser apresentada contém três gráficos, como na Figura 6.4(a), que representam, de cima para baixo: a frequência das classes (PPR e CART no caso dessa figura), as taxas de erro dos métodos MetaStream e Default, e a diferença entre as taxas de erro (valores negativos indicam melhor desempenho preditivo do MetaStream, enquanto valores positivos indicam melhor desempenho do Default). Como há somente duas classes, a curva das frequências dessas classes são complementares, como pode ser observado no gráfico superior da Figura 6.4(a). As curvas foram suavizadas com o uso de uma janela deslizante de tamanho 100 e passo 1 para facilitar a visualização da sua tendência.

O primeiro caso a ser analisado é quando a distribuição de classes não é representada adequadamente pela média. Isso ocorreu, por exemplo, para o par de algoritmos PPR/CART do conjunto de dados Airline, conforme mostrado na Figura 6.4(a). Como há muitas variações das frequências das classes ao longo do tempo, a distribuição média calculada sobre todo o fluxo indica simplesmente que as classes estão perfeitamente balanceadas. Ter conhecimento de que essas variações ocorrem ao longo do tempo é importante para a compreensão dos resultados obtidos, pois elas afetam o desempenho dos métodos de seleção de algoritmos, como pode ser observado pelas taxas de erro dos métodos e da diferença entre elas. Outro exemplo de que a média não representa corretamente a



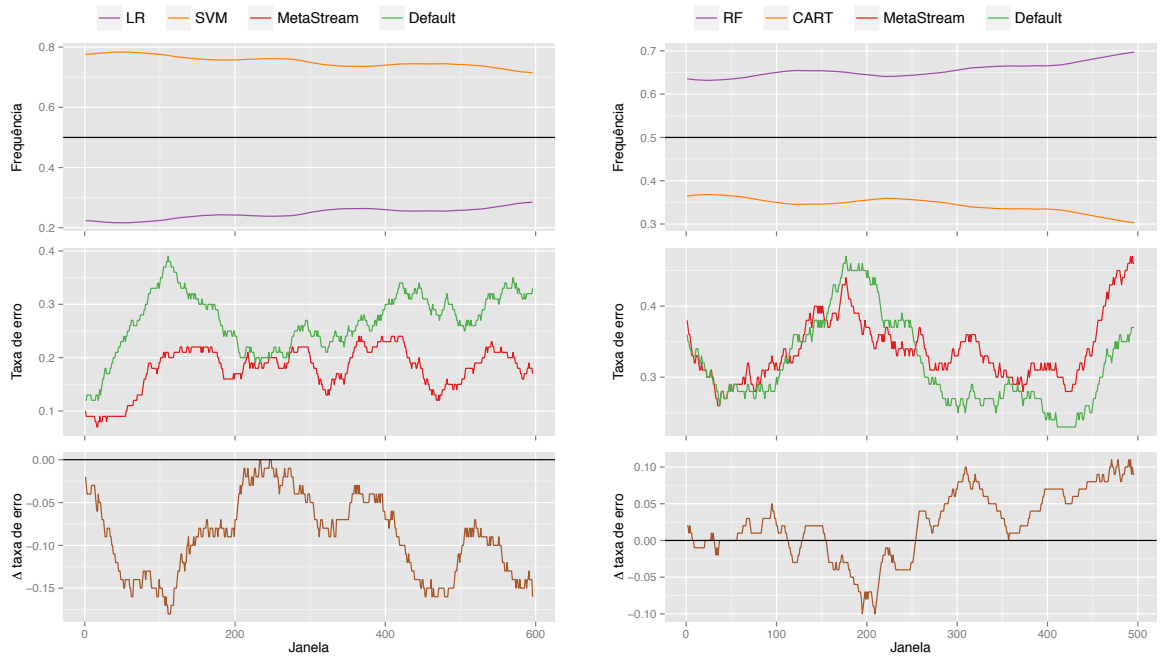
(a) Airline - PPR/CART. (b) TTP - PPR/SVM.

Figura 6.4: Frequências das classes e taxas de erro dos métodos de seleção de algoritmos ao longo do tempo para os pares de regressores PPR/CART, do conjunto de dados Airline, e PPR/SVM, do conjunto de dados TTP.

frequência das classes ao longo do fluxo de dados ocorreu para o par PPR/SVM, do conjunto de dados TTP, exibido na Figura 6.4(b). Nesse caso, há uma única inversão entre as classes minoritária e majoritária próximo da janela 200. No início, a classe minoritária representava 44% dos exemplos, mas chegou a 63% próximo da janela 400, sendo que a distribuição média das classes é de 48% e 52% para os algoritmos PPR e SVM, respectivamente. Nessa figura, verifica-se que o método MetaStream conseguiu menores taxas de erro do que o Default durante todo o fluxo, mas sobretudo quando a diferença das frequências das classes começam a diminuir até chegarem ao balanceamento perfeito de 50%. Essa vantagem do MetaStream é devido, principalmente, ao aumento das taxas de erro do método Default para esses períodos, pois os rótulos da maioria dos exemplos de teste são diferentes da classe majoritária do conjunto de treinamento. Quando isso acontece, a taxa de erro do Default pode ser maior do que 0.5, quando as classes estão perfeitamente balanceadas.

A superioridade do método MetaStream na Figura 6.4(b) pode ser atribuída, em parte, à pequena diferença entre as frequências das duas classes, embora o mesmo não tenha sido observado na Figura 6.4(a), em que essa diferença também é pequena. Em alguns casos, o método MetaStream obteve menores taxas de erro do que o Default, mesmo com o acentuado desbalanceamento de classes no conjunto de treinamento, como pode ser visto na Figura 6.5(a) para o par de algoritmos LR/SVM do conjunto de dados TTP.

Isso mostra que, apesar do MetaStream ser influenciado pela distribuição de classes, esse não é o único fator que guia o comportamento deste método, que conseguiu melhores resultados que o Default mesmo em alguns problemas com desbalanceamento de classes, como ilustrado nessa figura. No entanto, o mais comum em cenários como esse, é observar uma similaridade entre os métodos ou uma vantagem do método Default, como para o par de algoritmos RF/CART do conjunto de dados EDP, mostrado na Figura 6.5(b).



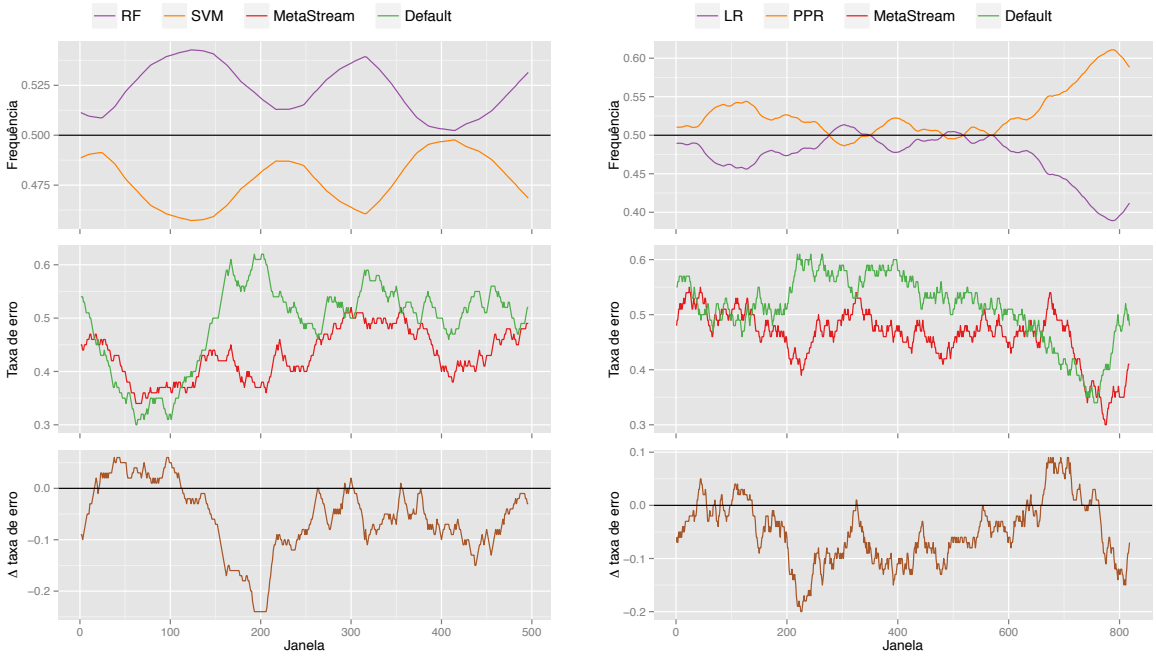
(a) TTP - LR/SVM.

(b) EDP - MARS/CART.

Figura 6.5: Frequências das classes e taxas de erro dos métodos de seleção de algoritmos ao longo do tempo para os pares de regressores LR/SVM, do conjunto de dados TTP, e MARS/CART, do conjunto de dados EDP.

O uso de um método sofisticado para a seleção de algoritmos quando as classes estão desbalanceadas geralmente não é melhor do que sempre predizer a classe majoritária, como o Default faz, pois é trivial escolher o melhor algoritmo com base nos dados passados. Por outro lado, o balanceamento entre as classes pode indicar que há um equilíbrio do desempenho preditivo dos algoritmos entre diferentes subconjuntos de dados, o que torna a distinção das classes, e, consequentemente o problema de classificação, mais complexo. Dessa maneira, o método MetaStream pode encontrar mais dificuldades em identificar o melhor algoritmo. Contudo, como mencionado anteriormente, o MetaStream consegue, em geral, menores taxas de erro do que o Default quando as classes estão balanceadas, como para o par de algoritmos RF/SVM do conjunto de dados EDP e o par LR/PPR do conjunto de dados Airline, mostrados nas Figuras 6.6(a) e 6.6(b), respectivamente. Nesses gráficos, é possível observar que o comportamento dos métodos é afetado mesmo quando há uma pequena variação na frequência das classes. Apesar disso, as taxas de

erro obtidas pelo MetaStream são quase sempre menores do que as obtidas pelo Default, como pode ser visto nos gráficos das diferenças.



(a) EDP - RF/SVM. (b) Airline - LR/PPR.

Figura 6.6: Frequências das classes e taxas de erro dos métodos de seleção de algoritmos ao longo do tempo para os pares de regressores RF/SVM, do conjunto de dados EDP, e LR/PPR, do conjunto de dados Airline.

6.1.2 Nível base

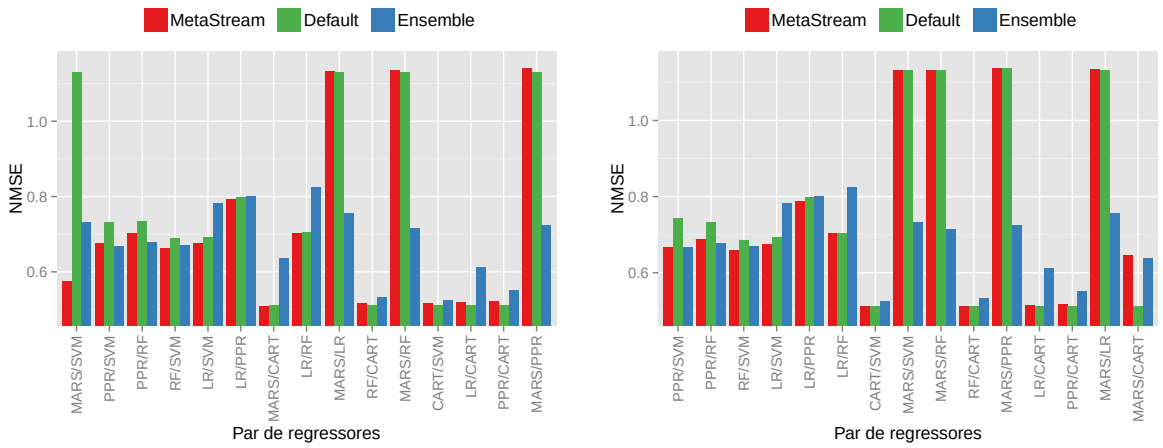
Para avaliar se a seleção de algoritmos realizada pelo método MetaStream é capaz de melhorar o sistema de aprendizado no nível base, o seu desempenho foi comparado ao do método Default e ao da abordagem Ensemble, que prediz a saída desejada como a média das predições de ambos os regressores sob análise, como mencionado na Seção 5.3.5. O teste estatístico de Friedman (Demšar, 2006) foi aplicado para cada conjunto de dados, a fim de verificar se as diferenças de desempenho entre os métodos considerando todos os pares de algoritmos são significativamente diferentes. Com 95% de confiança, o teste rejeitou a hipótese nula de que os métodos possuem desempenhos comparáveis apenas para o conjunto de dados Airline. Portanto, o pós-teste de Holm (Demšar, 2006) foi aplicado em seguida para verificar para quais métodos as diferenças foram significativas, considerando o MetaStream como o método de controle. Esse teste apontou que o Ensemble é significativamente melhor que o MetaStream, que, por sua vez, é significativamente melhor que o Default. Na Tabela 6.2 são apresentados os rankings utilizados no teste de Friedman. A abordagem Ensemble obteve o menor ranking (quanto menor, melhor) para os conjuntos de dados EDP e Airline, enquanto que o método Default obteve o menor

Tabela 6.2: Ranking médio dos métodos MetaStream e Default e da abordagem Ensemble no nível base para todos os pares de regressores.

	MetaStream	Default	Ensemble
TTP	2.00	1.87	2.13
EDP	2.00	2.33	1.67
Airline	2.13	2.87	1.00

ranking para o conjunto TTP. Apesar de não ter sido o melhor para nenhum conjunto, o MetaStream também nunca foi o pior método, o que aconteceu com Default e com a abordagem Ensemble, o que indica que é o método mais robusto.

Para analisar esses resultados para cada par de algoritmos, gráficos similares aos analisados para o nível meta são exibidos a seguir para o nível base. Os gráficos dos valores de NMSE para o conjunto de dados TTP são apresentados na Figura 6.7 para as estratégias de rotulação Sem-empate e Combinação, enquanto que gráficos similares para os conjuntos de dados EDP e Airline são apresentados na Figura 6.8 somente para a estratégia Combinação. Assim como no nível meta, os pares de regressores estão ordenados pela diferença de desempenho entre os métodos MetaStream e Default. Espera-se que as diferenças observadas no nível meta sejam verificadas também no nível base, embora, como discutido anteriormente, isso não possa ser garantido. Ou seja, ainda que a medida de avaliação usada no nível meta seja adequada para estimar o impacto que os métodos de seleção de algoritmos terão no nível base, essa relação entre os dois níveis de aprendizado pode não ser trivial, pois a rotulação dos meta-exemplos considera a comensurabilidade das diferenças, mas apenas qualitativamente, visto que a magnitude absoluta é ignorada.



(a) TTP - Sem-empate

(b) TTP - Combinação

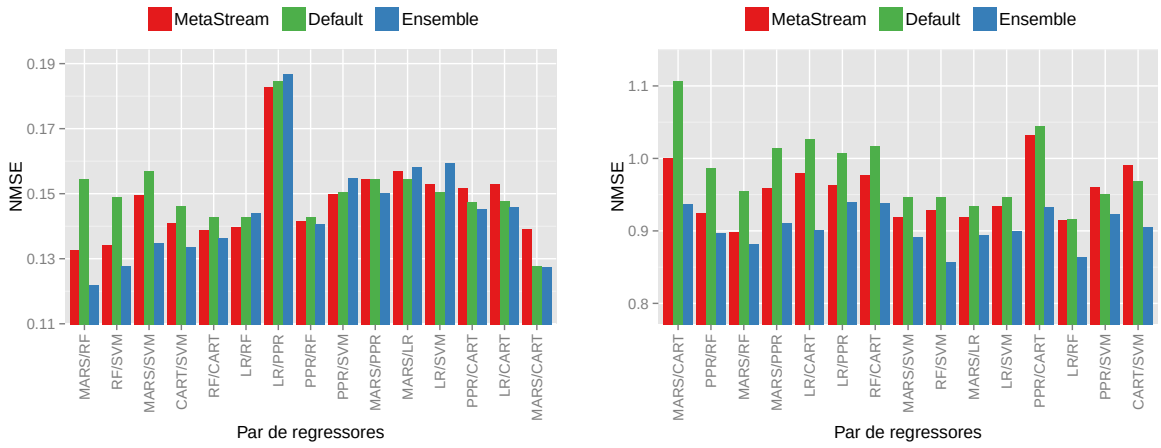
Figura 6.7: NMSE dos métodos de seleção de algoritmos no nível base para o conjunto de dados TTP usando as estratégias de rotulação Sem-empate e Combinação.

Analisando o gráfico da Figura 6.7(a), nota-se que os valores de NMSE dos métodos

MetaStream e Default são, em geral, similares. Deve-se levar em consideração que o intervalo das ordenadas prejudica a visualização das diferenças. Para os pares de algoritmos começando de RF/SVM até PPR/RF, é possível verificar que o MetaStream conseguiu pequenos ganhos de desempenho, sendo o maior deles para o par PPR/SVM. A grande diferença para o par MARS/SVM poderia ter ocorrido também com outros pares em que o MARS é um dos regressores, pois o modelo induzido por esse algoritmo fez algumas predições discrepantes. Consequentemente, quando esse regressor é selecionado erroneamente por um dos métodos, o valor de NMSE resultante será muito maior do que se o melhor regressor fosse selecionado. Portanto, os resultados para os pares formados pelo algoritmo MARS devem ser interpretados considerando esse fato. Em comparação com a abordagem Ensemble, pode-se observar que a seleção de algoritmos resultou em ganhos expressivos para alguns casos, como LR/RF e LR/SVM. Esses ganhos ocorreram, principalmente, para os pares formados pelo regressor LR, que possui um erro muito maior do que os outros algoritmos, prejudicando o desempenho da abordagem Ensemble. Por outro lado, as maiores perdas foram obtidas para a maioria dos pares em que o MARS é um dos algoritmos. Nesses casos, os erros das predições discrepantes feitas por esse regressor são amenizados pelo Ensemble, pois esse utiliza a média das predições de ambos os regressores para predizer a saída desejada. Outro aspecto importante desse gráfico é que os menores valores de NMSE foram obtidos quando o CART é um dos regressores (cinco primeiros pares). Para esses pares, o MetaStream e o Default conseguiram desempenhos similares, pois é trivial escolher o melhor regressor. Os experimentos com os meta-dados rotulados pela estratégia Combinação (Fig. 6.7(b)) resultaram em valores de NMSE muito semelhantes aos da estratégia Sem-empate, embora tenham sido constatadas algumas diferenças entre essas estratégias no nível meta. No nível base, as duas estratégias apenas diferem consideravelmente em relação a alguns pares de algoritmos formados pelo regressor MARS, pelos mesmos motivos já mencionados.

Para o conjunto de dados EDP (Figura 6.8(a)), observa-se que o método MetaStream conseguiu menores valores de NMSE para a maioria dos pares de regressores em comparação com o Default. Essa vantagem é mais evidente para os pares em que o MetaStream também foi melhor que o Default no nível meta, como para RF/SVM e MARS/RF. Porém, essa coerência entre os dois níveis não é verificada para alguns casos, como para MARS/CART e MARS/PPR. Para o primeiro, o MetaStream obteve menor taxa de erro, mas maior NMSE do que o Default, enquanto o oposto ocorreu para o segundo. Apesar da superioridade do método MetaStream em relação ao Default, o Ensemble conseguiu, em geral, menores valores de NMSE do que ambos os métodos de seleção de algoritmos. As exceções vistas no gráfico ocorreram para o par PPR/SVM e para quatro dos cinco pares formados pelo regressor LR, pois esse possui desempenho preditivo inferior aos demais regressores, o que influenciou negativamente a predição do Ensemble. Pelos resultados apresentados, nota-se que a abordagem Ensemble é mais adequada para esse conjunto

de dados, pois a média das predições dos dois regressores evitou que grandes erros fossem cometidos. Por outro lado, quando um regressor é sistematicamente pior, o método Ensemble é mais afetado do que os métodos de seleção de algoritmos. Obviamente, uma maior acurácia dos métodos de seleção de algoritmos implicaria em menores taxas de erros no nível base do que as obtidas pela abordagem Ensemble.



(a) EDP (b) Airline

Figura 6.8: NMSE dos métodos de seleção de algoritmos e da abordagem Ensemble no nível base para os conjuntos de dados EDP e Airline usando a estratégia de rotulação Combinação.

Os resultados similares entre a abordagem Ensemble e o método Default para alguns pares de regressores, assim como as divergências de desempenho entre os níveis meta e base para os métodos MetaStream e Default no conjunto de dados EDP, podem ser explicados pela distribuição de classes. Por exemplo, pelo gráfico de frequência de classes para o par de algoritmos MARS/CART, mostrado na Figura 6.9, é possível verificar que *combinação* é a classe majoritária durante todo o fluxo de meta-dados. Assim, o método Default sempre prediz essa classe, que no nível base equivale à abordagem Ensemble, pois a média das predições dos regressores será utilizada para predizer o atributo alvo. Consequentemente, os valores de NMSE do Default e do Ensemble são similares para esse par de algoritmos (Fig. 6.8(a)) e são melhores do que o MetaStream, mesmo que o Default tenha obtido pior desempenho no nível meta.

Na Figura 6.8(b), são apresentados os resultados para o conjunto de dados Airline. Por esse gráfico, fica evidente a superioridade da abordagem Ensemble em relação aos métodos MetaStream e Default, visto que a abordagem obteve menores valores de NMSE para todos os pares de algoritmos. Esses resultados são reflexo da baixa acurácia dos métodos de seleção de algoritmos no nível meta, com taxas de erro no nível meta sempre acima de 0.5, exceto para um par. O fato de que uma única predição errônea no nível meta pode resultar em um grande erro no nível base, devido às predições discrepantes, também

contribui para as diferenças observadas. Comparando apenas os métodos de seleção de algoritmos, o melhor desempenho do MetaStream é contundente. Para alguns pares de algoritmos, como MARS/RF e PPR/RF, a diferença de NMSE para o método Default é maior que 0.05. Analisando os gráficos do nível meta e do nível base concomitantemente, constata-se que as menores taxas de erro do método Default em relação ao MetaStream não significaram menores valores de NMSE. Para os pares RF/SVM e MARS/PPR, por exemplo, o método MetaStream obteve taxas de erro maiores do que o Default no nível meta, mas melhores resultados no nível base. Os motivos são os mesmos mencionados anteriormente.

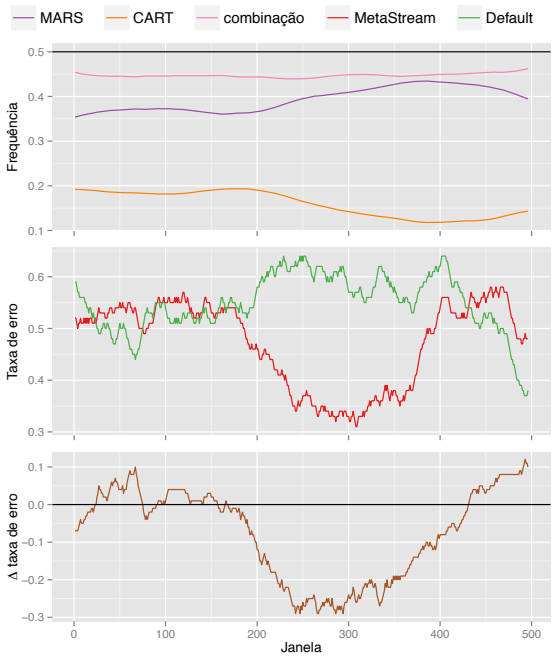


Figura 6.9: Frequências das classes e taxas de erro dos métodos de seleção de algoritmos ao longo do tempo para os pares de regressores MARS/CART, do conjunto de dados EDP.

6.2 Seleção de algoritmos para lotes de exemplos e para cada exemplo

Nesta Seção, são apresentados e discutidos os resultados experimentais do uso do método MetaStream para a tarefa de seleção de algoritmos para lotes de exemplos (SelLote) e para cada exemplo do nível base (SelUnit). O objetivo é avaliar empiricamente a veracidade da segunda hipótese específica estabelecida neste trabalho:

> *O uso do método MetaStream para a seleção de algoritmos para cada exemplo do FCD é viável e resulta em melhor desempenho preditivo do sistema de aprendizado a nível base comparado à seleção para lotes de exemplos.*

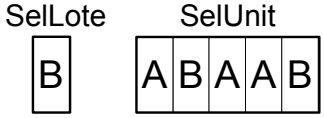


Figura 6.10: Rótulos para o meta-exemplo de SelLote e os meta-exemplos de SelUnit de acordo com os erros dos algoritmos A e B mostrados na Tabela 6.3.

As abordagens de seleção SelLote e SelUnit geram conjuntos de meta-dados distintos, conforme as características intrínsecas de cada uma. Como descrito anteriormente, em SelLote, um meta-exemplo representa um lote de exemplos do nível base, para os quais, um único algoritmo é selecionado, enquanto em SelUnit, um meta-exemplo representa um único exemplo do nível base, ou seja, um algoritmo é selecionado para cada exemplo do nível base.

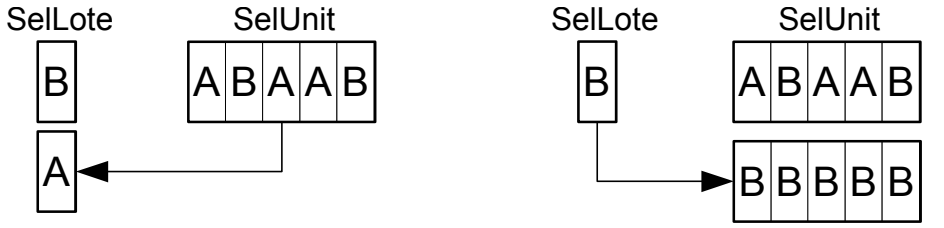
A comparação entre essas abordagens é trivial no nível base, pois, apesar dos conjuntos de meta-dados serem diferentes, os meta-exemplos usados para a indução e teste dos meta-modelos para cada instante de tempo, foram extraídos dos mesmos dados do nível base. No entanto, a comparação entre SelLote e SelUnit no nível meta não é direta, pois a seleção de algoritmos é feita para diferentes granularidades, impossibilitando o pareamento direto das predições realizadas. Dessa maneira, a primeira questão a ser investigada é se existe um método adequado para avaliar os resultados no nível meta sabendo-se que os conjuntos de meta-dados são diferentes.

Para tentar responder a essa questão, é importante retomar como os meta-exemplos são rotulados. Por razões de simplicidade, considere que a tarefa de selecionar um algoritmo, sendo A e B os dois algoritmos sob análise, usando a estratégia de seleção Sem-empate. Em SelLote, um meta-exemplo é rotulado como A se o algoritmo A obteve desempenho preditivo médio melhor ou igual ao algoritmo B para os γ exemplos de teste que são caracterizados por esse meta-exemplo; caso contrário, o meta-exemplo é rotulado como B. Em SelUnit, um meta-exemplo é rotulado como A se o algoritmo A obteve desempenho preditivo melhor ou igual ao algoritmo B para o único exemplo caracterizado pelo respectivo meta-exemplo; caso contrário, o meta-exemplo é rotulado como B. Seja $\gamma = 5$, na Tabela 6.3 são apresentados os erros hipotéticos dos modelos induzidos pelos algoritmos A e B para cada exemplo do nível base (Ex.1, ..., Ex.5) e o erro médio para todos os exemplos (última coluna). De acordo com essa tabela, os meta-exemplos gerados para SelLote e e SelUnit são rotulados como mostrado na Figura 6.10.

Tabela 6.3: Erros preditivos dos algoritmos hipotéticos A e B para cada exemplo do nível base e o erro médio calculado sobre todos os exemplos.

Algoritmo	Ex.1	Ex.2	Ex.3	Ex.4	Ex.5	Erro médio
A	0.25	0.50	0.20	0.20	0.70	0.37
B	0.30	0.20	0.25	0.30	0.15	0.24

O método mais intuitivo para comparar as abordagens SelLote e SelUnit no nível meta é adaptar as predições de uma delas à granularidade da outra, ou seja, agregar as predições de SelUnit ou replicar a predição de SelLote. Na Figura 6.11, são mostradas as duas situações possíveis: agregação das predições de SelUnit (Fig. 6.11(a)) e a replicação da predição de SelLote (Fig. 6.11(b)). No primeiro caso, o rótulo agregado de SelUnit para comparação com SelLote corresponde à classe majoritária das cinco predições realizadas em SelUnit. A divergência entre o rótulo original de SelLote e o rótulo agregado de SelUnit ocorre porque o rótulo de SelLote representa o algoritmo que obteve o menor erro médio para os cinco exemplos correspondentes no nível base (Fig. 6.10), e não a classe majoritária para esses exemplos, como na agregação de SelUnit. No segundo caso, da replicação do rótulo de SelLote, ocorre exatamente o oposto, pois, embora o algoritmo B tenha o menor erro médio, esse algoritmo corresponde a apenas 2/5 das classes de SelUnit.



(a) Agregação das predições de SelUnit para comparação com SelLote.

(b) Replicação da predição de SelLote para comparação com SelUnit.

Figura 6.11: Agregação e replicação das predições para comparação entre SelLote e SelUnit.

Apesar das diferenças entre os dois métodos de avaliação, é necessário usar um deles na comparação entre as abordagens SelLote e SelUnit no nível meta. Para fazer essa escolha, deve-se considerar que o objetivo principal da seleção de algoritmos é melhorar o desempenho preditivo do sistema de aprendizado no nível base. Nesse sentido, a abordagem SelUnit possui a vantagem da seleção de algoritmos ser realizada com a mesma granularidade dos dados no nível base, ou seja, para cada exemplo. Observando novamente a Tabela 6.3 e a Figura 6.10, fica claro que, se os algoritmos forem selecionados com 100% de acurácia nas duas abordagens, a seleção de algorimtos com a abordagem SelUnit produzirá erros menores do que com SelLote (0.20 e 0.24, respectivamente). Com base nesses argumentos e na hipótese investigada nesta seção, decidiu-se utilizar a abordagem da Figura 6.11(b), em que cada predição de SelLote é replicada γ vezes e são confrontadas com as classes reais para os meta-exemplos de SelUnit. Esses resultados são referentes apenas aos meta-dados rotulados com a estratégia Sem-empate, pois, esta gera problemas de classificação binários, que podem ser analisados mais facilmente do que os resultados produzidos com a estratégia Combinação.

Embora o método de avaliação experimental escolhido para o nível meta tenha a mesma granularidade do nível base, ainda podem existir diferenças entre os resultados

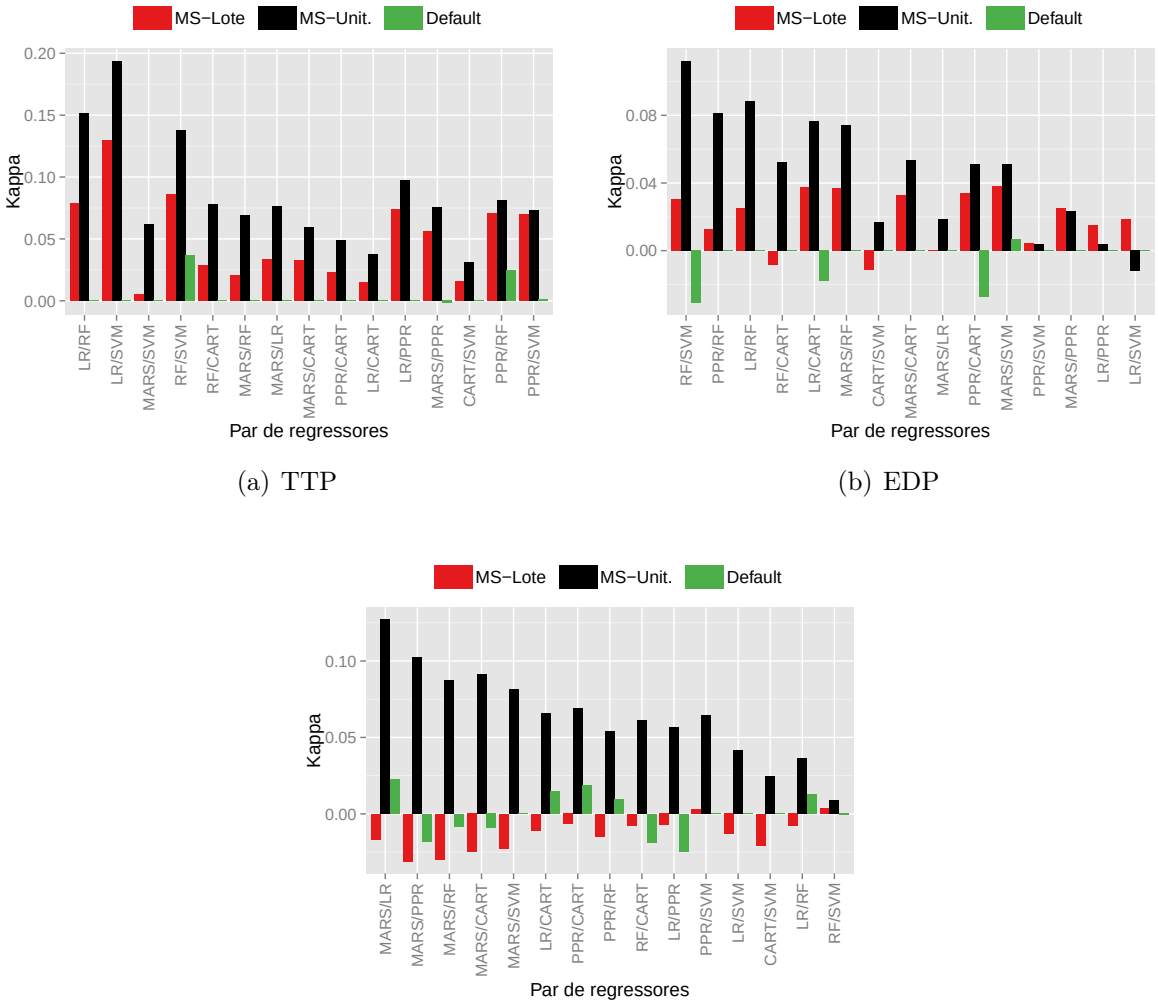
dos dois níveis de aprendizado, pois a avaliação no nível meta não considera a diferença de magnitude dos erros dos regressores. Outro fator que interfere na concordância entre os resultados observados nos dois níveis é a medida de avaliação empregada. Nesta seção, a estatística Kappa é usada no nível meta, pois, como descrito no Capítulo 5, ela é adequada para problemas com diferentes distribuições de classes, como neste caso. Porém, ao contrário da taxa de erro, os resultados obtidos com a estatística Kappa podem ter uma fraca relação com os resultados observados no nível base, principalmente quando a distribuição de classes for muito desbalanceada. Por isso, o melhor método no nível meta pode não ser a melhor opção para melhorar o desempenho do sistema de aprendizado no nível base.

6.2.1 Nível meta

Como as abordagens SelLote e SelUnit geram conjuntos de meta-dados que, provavelmente, possuem distribuições de classes diferentes, a estatística κ é utilizada na avaliação do desempenho dos métodos no nível meta (Sec. 5.3.5). O teste estatístico de Wilcoxon foi aplicado para cada conjunto de dados com 95% de confiança, a fim de obter evidências sobre o desempenho preditivo do método MetaStream com as diferentes abordagens de seleção considerando todos os pares de algoritmos. O teste apontou que o método MetaStream com a abordagem SelUnit é significativamente melhor do que quando o mesmo método é usado com a abordagem SelLote para os três conjuntos de dados investigados. Para avaliar esses resultados mais detalhadamente, na Figura 6.12 são apresentados os gráficos dos valores da estatística κ para cada conjunto de dados e par de regressores. As barras em vermelho representam o desempenho do método MetaStream com a abordagem SelLote (MS-Lote), as barras em preto o método MetaStream com a abordagem SelUnit (MS-Unit) e as barras em verde, o Default. Como supunha-se, os valores de κ para o método Default são sempre próximos de zero, pois esse método faz a seleção de algoritmos com base apenas na distribuição de classes e a estatística κ leva em consideração essa distribuição. Assim, as barras referentes ao Default não aparecem nos gráficos para a maioria dos pares de algoritmos. Esses pares estão ordenados pela diferença dos valores de κ entre MS-Unit e MS-Lote.

Para o conjunto de dados TTP, dois aspectos importantes podem ser visualizados no gráfico da Figura 6.12(a). O primeiro é a diferença entre os métodos MetaStream e Default, independentemente da abordagem de seleção empregada, o que mostra que o meta-aprendiz está sendo capaz de extrair conhecimento a partir dos meta-dados para a maioria dos pares de regressores, ou seja, não está predizendo apenas com base na distribuição de classes, como o método Default faz. A segunda observação é que o método MetaStream conseguiu induzir meta-modelos com melhor desempenho preditivo com a abordagem SelUnit do que com SelLote para todos os pares de regressores, embora as

diferenças não sejam tão grandes. Apesar de serem piores, os resultados para SelLote indicam que o melhor algoritmo para um lote de exemplos do nível base é, geralmente, o melhor algoritmo para cada exemplo desse lote, ou seja, o algoritmo com o menor erro médio é a classe majoritária dos dados de teste. Caso contrário, os valores de Kappa de MS-Lote seriam tão próximos de zero quanto os do Default.



(a) TTP

(b) EDP

(c) Airline

Figura 6.12: Estatística κ do método MetaStream com as abordagens de seleção de algoritmos SelLote (MS-Lote) e SelUnit (MS-Lote) e do método Default para os conjuntos de dados TTP, EDP e Airline usando a estratégia de rotulação Sem-empate.

No gráfico dos valores da estatística κ para o conjunto de dados EDP, mostrado na Figura 6.12(b), é possível notar que o método MS-Unit obteve maiores valores de κ do que MS-Lote para a maioria dos pares de algoritmos. Entretanto, os valores de Kappa desses dois métodos, principalmente de MS-Lote, são apenas ligeiramente melhores do que aqueles obtidos pelo método Default para a maioria dos pares de regressores. Esses resultados apontam que o método MetaStream está cometendo muitos erros com SelLote, ou que essa abordagem não é recomendada para o conjunto de dados EDP, pois o melhor algoritmo

para um lote de exemplos pode não ser o melhor algoritmo para a maioria dos exemplos desse lote, assim como na Figura 6.11(b). As mesmas observações podem ser feitas para os resultados do conjunto de dados Airline, que podem ser vistos na Figura 6.12(c). Para esse conjunto, é ainda mais evidente que os valores de Kappa obtidos por MS-Lote são muito próximos daqueles do método Default, ou seja, as predições de MS-Lote ocorreram ao acaso. Portanto, conclui-se que a abordagem de seleção de um algoritmo para cada exemplo é claramente mais apropriada do que SelLote para o conjunto de dados Airline.

O gráfico exibido na Figura 6.13 pode auxiliar na discussão das diferenças dos resultados apresentados para os conjuntos de dados TTP, EDP e Airline. Nessa figura, são apresentados os valores de Kappa de um método de seleção de algoritmos hipotético que prediz corretamente a classe de todos os meta-exemplos de teste usando a abordagem SelLote e é avaliado com o mesmo método de replicação das predições de SelLote (Figura 6.11(b)). Se o melhor algoritmo para um lote de exemplos de acordo com o erro médio também é o melhor algoritmo para todos os exemplos do lote de acordo com o erro para cada exemplo, então $\kappa = 1$. Quanto mais próximo de zero, maior a discordância entre o rótulo de um lote e os rótulos dos exemplos unitários que pertencem a este lote. Assim, observando a Figura 6.13, é possível afirmar que o melhor algoritmo para um lote de exemplos geralmente não é o melhor algoritmo para todos os exemplos do lote. O caso mais contundente, como conjecturado nos parágrafos anteriores, é para o conjunto de dados Airline, em que os valores de Kappa com SelLote são inferiores a 0.1 para 12 pares de regressores, de um total de 15 investigados. O conjunto de dados TTP é o menos afetado por esse problema, embora ainda apresente uma discordância razoável entre o rótulo dos lotes e dos exemplos unitários. Embora essa questão explique parcialmente os resultados observados no nível meta, outros dois fatores que podem ter influenciado o comportamento dos métodos MS-Lote e MS-Unit são analisados a seguir: a importância dos meta-atributos e o nível de ruído nos meta-dados gerados por cada abordagem.

Importância dos meta-atributos e taxa de ruído de SelLote e SelUnit

Devido às diferenças das abordagens SelLote e SelUnit na caracterização dos dados de teste, como apresentado na Seção 4.4, os meta-atributos gerados por cada abordagem podem ser um dos fatores determinantes do desempenho dos métodos MS-Lote e MS-Unit. Para analisar a influência dos meta-atributos que estão presentes exclusivamente em cada uma das abordagens, a importância que o algoritmo Random Forest (RF) atribui a cada meta-atributo durante o processo de indução de um modelo é analisada. Dentre várias medidas de importância (Sandri e Zuccolotto, 2006), a mais usada na literatura, inclusive para seleção de atributos (Díaz-Uriarte e Alvarez de Andrés, 2006; Strobl et al., 2008; Genuer et al., 2010), é a redução média da acurácia (MDA, do inglês, *mean decrease accuracy*), a qual é investigada neste trabalho. Essa medida está disponível na implementação do algoritmo RF usado como meta-aprendiz (Liaw e Wiener, 2002). Nessa medida,

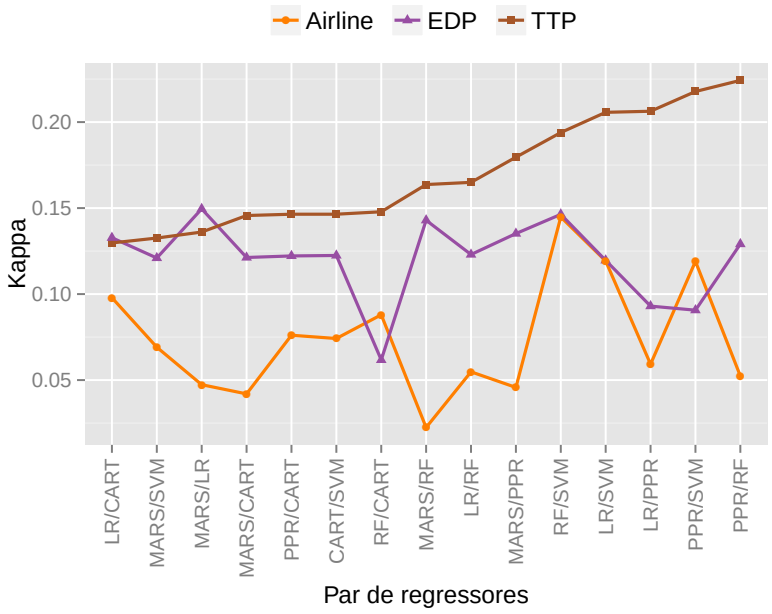


Figura 6.13: Valores da estatística κ de um método de seleção de algoritmos ideal (com 100% de acurácia) para SelLote usando o método de avaliação em que as predições são replicadas para comparação com SelUnit.

a importância de um atributo é dada pela variação do desempenho preditivo quando os valores desse atributo são aleatoriamente permutados (Breiman, 2001; Strobl et al., 2008). Para o propósito deste trabalho, a importância dos meta-atributos exclusivos em cada abordagem é dado pela soma da MDA de todos os meta-atributos exclusivos dividido pela soma da MDA de todos os meta-atributos que são comuns às duas abordagens e possuem peso positivo. Assim, obtêm-se a relevância relativa dos meta-atributos exclusivos de cada abordagem.

Além da importância dos meta-atributos, a taxa de ruído dos conjuntos de meta-dados gerados por cada abordagem também é analisada. Segundo Wu (1995), em problemas de classificação, os ruídos podem estar presentes nos atributos preditivos ou na classe. Para o primeiro caso, os ruídos podem ser consequência de valores errôneos, ausentes ou incompletos. Os ruídos de classe ocorrem quando há exemplos contraditórios, em que o mesmo conjunto de valores para os atributos preditivos aparece mais do que uma vez e é rotulado com classes diferentes, ou quando exemplos são rotulados com a classe incorreta (Zhu e Wu, 2004). Na abordagem SelUnit, se o valor de γ é menor do que λ_b (Figura 4.3), cada subconjunto de γ meta-exemplos possui os mesmos valores para os meta-atributos que foram extraídos dos dados de treinamento. Portanto, esses meta-exemplos diferem apenas nos valores dos meta-atributos obtidos dos exemplos de teste do nível base e cada meta-exemplo é rotulado com base no erro dos regressores para cada exemplo. Por esses motivos, supõe-se que os meta-dados gerados por SelUnit sejam mais ruidosos do que os meta-dados gerados por SelLote. Para avaliar se essa hipótese é verdeira

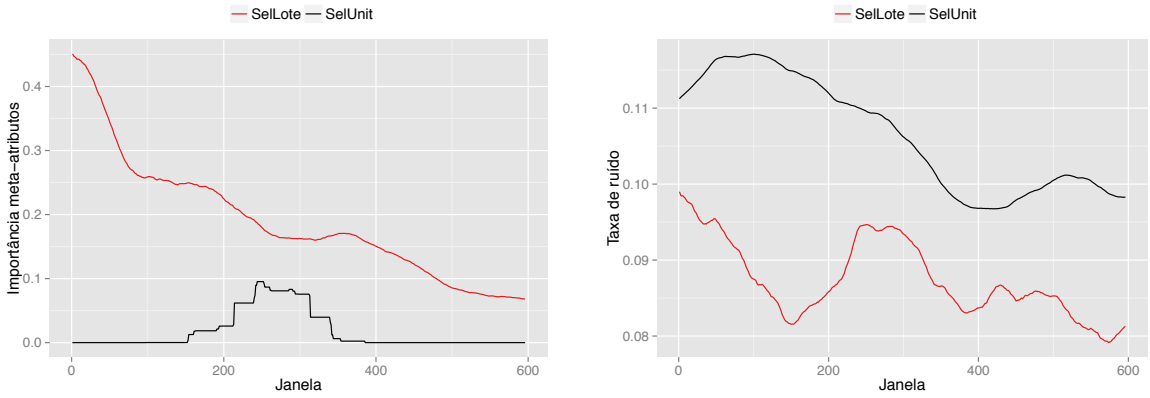
e verificar a influência do ruído nos resultados observados no nível meta, o algoritmo *Repeated Edited Nearest Neighbour* (RENN), proposto por Tomek (1976), é usado com $k = 7$ vizinhos para estimar o número de meta-exemplos ruidosos em cada janela de treinamento. A taxa de ruído é dada pelo número de meta-exemplos potencialmente ruidosos obtido pelo algoritmo sobre o número total de meta-exemplos de treinamento.

Na elaboração dos gráficos que mostram a importância dos meta-atributos e a taxa de ruído ao longo do tempo, é utilizada uma janela deslizante de tamanho 100 e passo 1, assim como foi feito para a análise da distribuição de classes na Seção 6.1. Como é impraticável exibir todos os gráficos, por causa do espaço necessário, apenas alguns casos serão analisados a seguir.

Para o conjunto de dados TTP, a importância dos meta-atributos e a taxa de ruído foram mostradas para os pares de regressores PPR/SVM e MARS/SVM. Para o primeiro par, os método MS-Lote e MS-Unit obtiveram valores de Kappa muito similares, enquanto que para o segundo par, o método MS-Unit obteve desempenho superior. Na Figura 6.14, são apresentados os gráficos de importância dos meta-atributos (esquerda) e da taxa de ruído (direita) para os conjuntos de meta-dados gerados por SelLote e SelUnit para o par de regressores PPR/SVM. Pela Figura 6.14(a), nota-se que os meta-atributos exclusivos de SelLote possuem importância relativa maior do que os meta-atributos exclusivos de SelUnit, o que é um resultado atípico para o conjunto de dados TTP. Por outro lado, as taxas de ruído de SelLote e SelUnit possuem o mesmo aspecto da Figura 6.14(b) para a maioria dos pares de regressores, ou seja, a taxa de ruído em SelUnit é maior do que a de SelLote. Apesar do problema de discordância entre os rótulos para um lote de exemplos e para exemplos unitários (Fig. 6.13), o desempenho similar entre os métodos MS-Lote e MS-Unit pode ser atribuído à maior taxa de ruído em SelUnit e à maior importância dos meta-atributos de SelLote. A análise dos resultados em termos da importância das variáveis e do nível de ruído permite explicar a maior parte dos resultados obtidos.

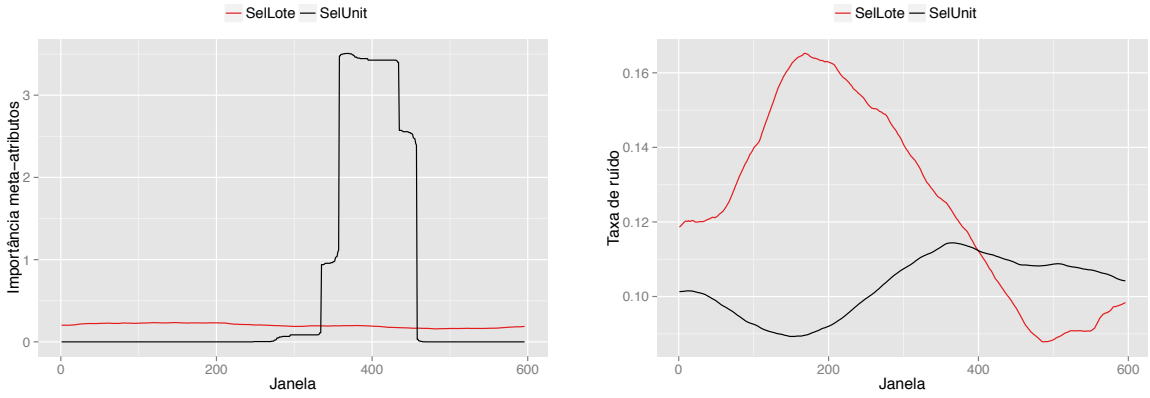
O aspecto dos gráficos para o par de regressores MARS/SVM, exibidos na Figura 6.15, é quase o oposto ao dos gráficos para PPR/SVM. Para o par MARS/SVM, a importância dos meta-atributos exclusivos de SelLote é muito similar à do SelUnit durante a maior parte do tempo. Porém, a importância dos meta-atributos de SelUnit cresceram rapidamente após a janela 300, voltando ao valor anterior próximo à janela 450. Ademais, a taxa de ruído de SelLote é maior do que SelUnit durante a maior parte do tempo, ao contrário do que ocorreu para a maioria dos pares de regressores. A importância dos meta-atributos e a taxa de ruído para MARS/SVM condizem com os valores de Kappa para esse par de algoritmos apresentados na Figura 6.12(a).

Para o conjunto de dados EDP, a importância dos meta-atributos exclusivos de SelLote é maior do que a encontrada em SelUnit para todos os pares de regressores, enquanto a taxa de ruído nos conjuntos de meta-dados de SelLote é, geralmente, similar ou maior do que de SelUnit, contradizendo a hipótese inicial de que os dados de SelUnit seriam



(a) Importância relativa dos meta-atributos. (b) Taxa de ruído.

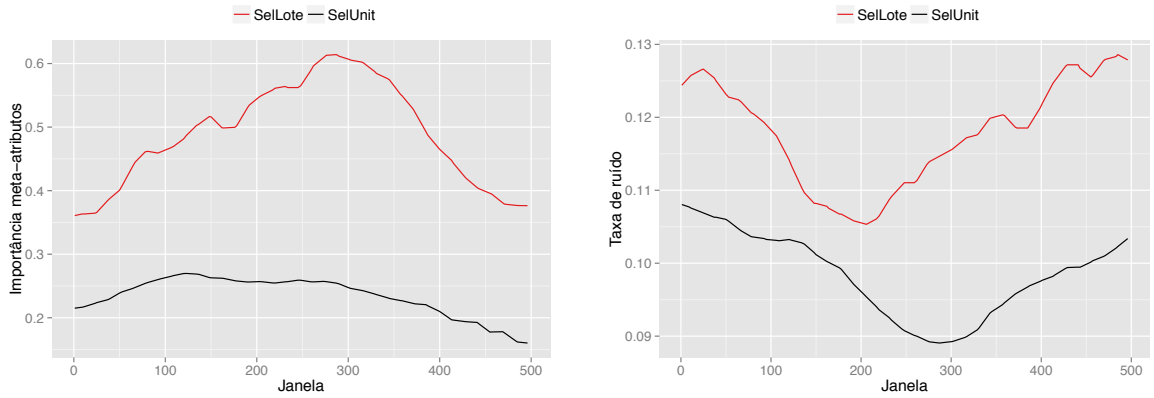
Figura 6.14: Importância relativa dos meta-atributos e taxa de ruído dos conjuntos de meta-dados gerados por SelLote e SelUnit para o par de regressores PPR/SVM do conjunto de dados TTP.



(a) Importância relativa dos meta-atributos. (b) Taxa de ruído.

Figura 6.15: Importância relativa dos meta-atributos e taxa de ruído dos conjuntos de meta-dados gerados por SelLote e SelUnit para o par de regressores MARS/SVM do conjunto de dados TTP.

mais ruidosos. Os gráficos para o par de regressores RF/SVM, mostrados na Figura 6.16, ilustram bem esses casos. A diferença dos valores de Kappa entre MS-Lote e MS-Unit para esse par de regressores é uma das maiores na Figura 6.12(b). Se os meta-atributos exclusivos de SelUnit tivessem maior importância, a vantagem de MS-Unit poderia ser ainda maior, pois a discordância entre os rótulos para um lote de exemplos e para exemplos unitários em SelLote também contribuiu para a deterioração do desempenho de MS-Lote, embora a discordância para RF/SVM seja uma das menores observadas para o conjunto de dados EDP (Fig. 6.13).



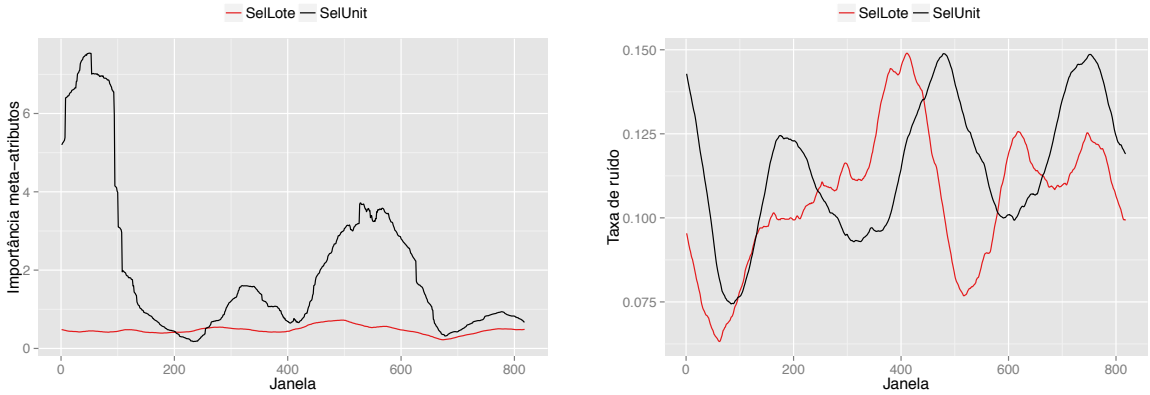
(a) Importância relativa dos meta-atributos.

(b) Taxa de ruído.

Figura 6.16: Importância relativa dos meta-atributos e taxa de ruído dos conjuntos de meta-dados gerados por SelLote e SelUnit para o par de regressores RF/SVM do conjunto de dados EDP.

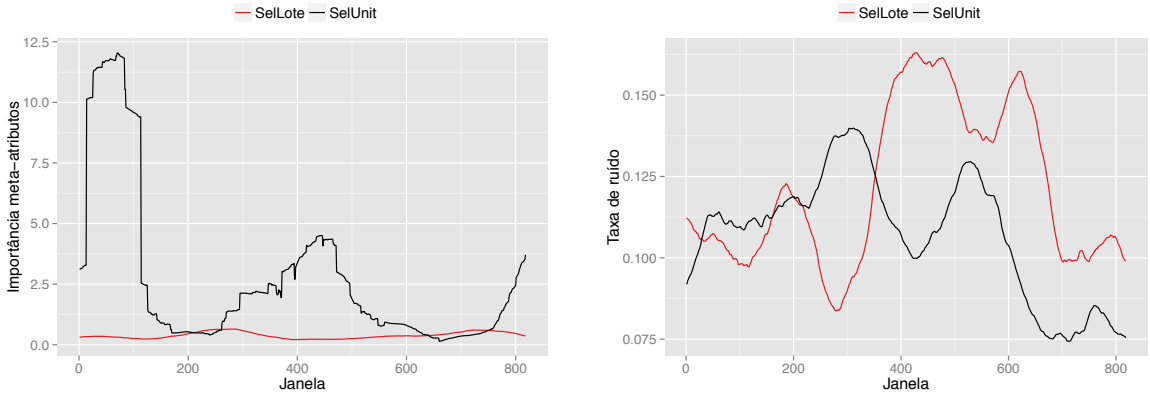
Para o conjunto de dados Airline, a importância dos meta-atributos específicos da abordagem SelUnit e a taxa de ruído de ambas as abordagens são muito instáveis. Devido a essa grande variação, é difícil estabelecer uma relação entre os gráficos mostrados na Figura 6.17 para o par de regressores MARS/LR e o desempenho dos métodos MS-Lote e MS-Unit. Apesar dessa instabilidade, o melhor desempenho do método MS-Unit para todos os pares de algoritmos pode ser parcialmente explicado pela maior importância dos meta-atributos exclusivos de SelUnit, como para o par de algoritmos MARS/LR, para o qual o método MS-Unit obteve o maior valor de Kappa dentre todos os pares. Pelo gráfico apresentado na Figura 6.17(a), nota-se que a importância dos meta-atributos para SelLote é de aproximadamente 0.5, enquanto que para SelUnit é de aproximadamente 1, chegando a 6 em alguns períodos. Comportamentos instáveis similares a esse foram observados para a maioria dos pares de regressores. Entretanto, para alguns deles, a maior importância dos meta-atributos exclusivos de SelUnit não resultou em maiores valores de Kappa em comparação com a abordagem SelLote. Esse é o caso do par de algoritmos LR/RF, cujos gráficos são apresentados na Figura 6.18. Apesar da grande importância dos meta-atributos de SelUnit, o desempenho dos métodos MS-Lote e MS-Unit é comparável (Fig. 6.12). Isso indica que o desempenho desses métodos também foi influenciado por

outros fatores que não foram analisados nesta seção, como, eventualmente, a distribuição de classes dos conjuntos de meta-dados gerados por cada abordagem.



(a) Importância relativa dos meta-atributos. (b) Taxa de ruído.

Figura 6.17: Importância relativa dos meta-atributos e taxa de ruído dos conjuntos de meta-dados gerados por SelLote e SelUnit para o par de regressores MARS/LR do conjunto de dados Airline.



(a) Importância relativa dos meta-atributos. (b) Taxa de ruído.

Figura 6.18: Importância relativa dos meta-atributos e taxa de ruído dos conjuntos de meta-dados gerados por SelLote e SelUnit para o par de regressores LR/RF do conjunto de dados Airline.

6.2.2 Nível base

Como mencionado no início da Seção 6.2, a estatística Kappa é mais adequada do que a taxa de erro para comparar os métodos MS-Lote e MS-Unit no nível meta, pois considera a distribuição de classes dos conjuntos de meta-dados utilizados por cada método. Porém, com essa medida, não é possível estimar o impacto que a seleção de algoritmos terá no nível base a partir do desempenho preditivo dos métodos de seleção no nível meta, principalmente quando há um grande desbalanceamento de classes. Assim, é imprescindível que

Tabela 6.4: Ranking médio dos métodos MS-Lote, MS-Unit e Default e da abordagem Ensemble no nível base para todos os pares de regressores.

	MS-Lote	MS-Unit	Default	Ensemble
TTP	2.40	2.27	2.53	2.80
EDP	2.73	2.53	2.73	2.00
Airline	2.73	2.60	3.67	1.00

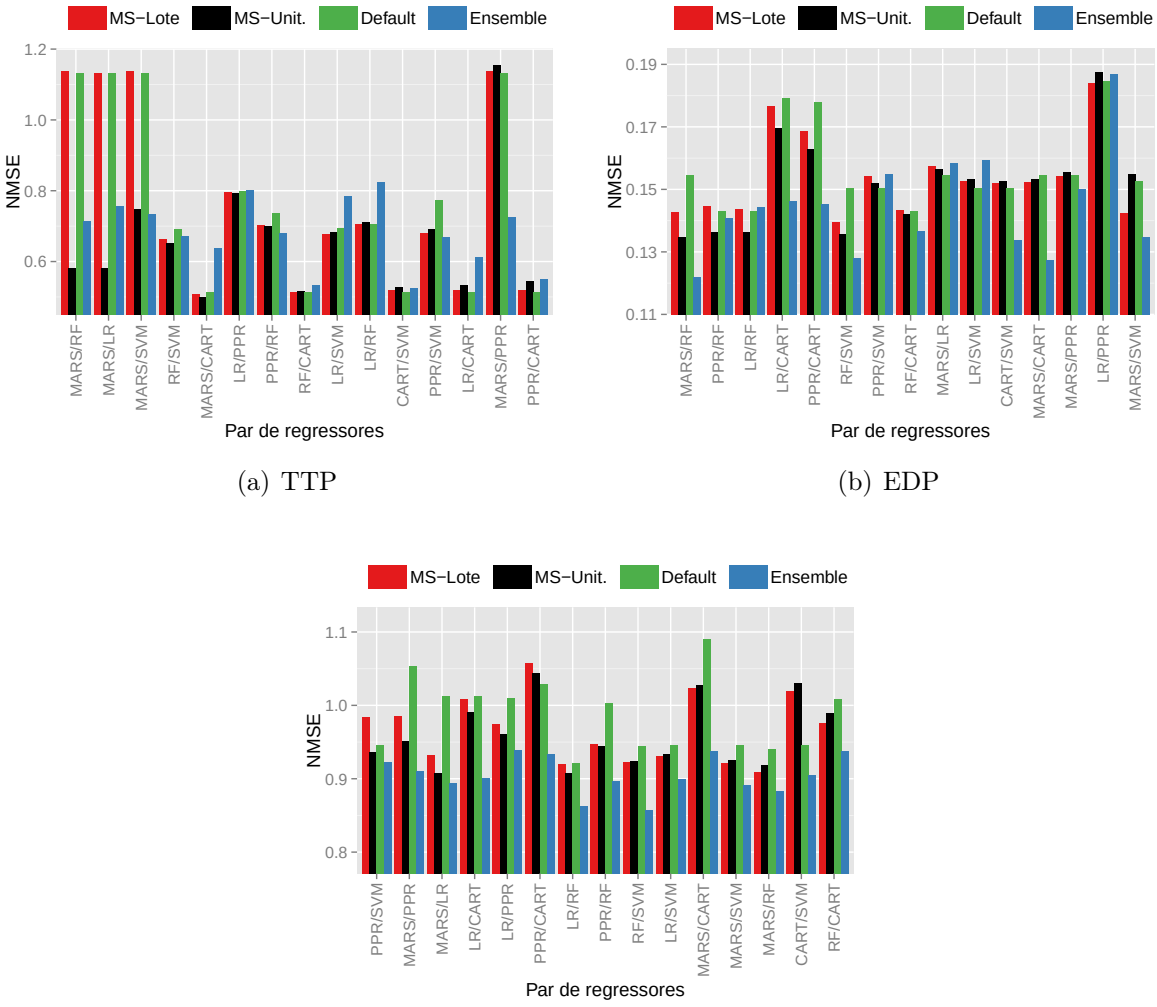
os métodos MS-Lote e MS-Unit sejam avaliados também no nível base. A discordância de resultados entre os dois níveis é maior com a estatística Kappa do que com a taxa de erro, pois a primeira considera a distribuição de classes, o que não ocorre com a taxa de erro e o desempenho preditivo calculado no nível base. Por exemplo, em um problema de classificação binária, com distribuição de classes 75/25, a estatística κ de um método A que sempre prediz a classe majoritária, será $\kappa = 0$, enquanto a de um método B que predisse corretamente 25 exemplos de cada classe, terá valor de $\kappa = 0.2$. Entretanto, a seleção realizada pelo método A, provavelmente resultará em melhor desempenho preditivo no nível base, pois predisse corretamente 75% dos exemplos, enquanto o método B, apenas 25%. Portanto, a medida de taxa de erro usada na avaliação a nível meta dos métodos na Seção 6.1, geralmente, está mais relacionada com o erro do nível base, mas também não considera a magnitude das diferenças dos regressores, o que gerou divergências entre o desempenho dos métodos nos dois níveis de aprendizado.

O teste de Friedman foi aplicado para verificar se as diferenças de desempenho preditivo no nível base resultante da seleção de algoritmos realizada pelos métodos avaliados são significativas. O teste encontrou evidências suficientes para rejeitar a hipótese nula de que essas diferenças são semelhantes com 95% de confiança apenas para o conjunto de dados Airline. Para esse caso, o pós-teste de Holm foi aplicado para identificar quais são os métodos que possuem desempenhos distintos. Considerando o método MS-Unit como o método de controle, o teste apontou, com 95% de confiança, que esse método é comparável com MS-Lote, significativamente melhor que o Default e significativamente pior que o Ensemble. Na Tabela 6.4 são apresentados os rankings utilizados no teste de Friedman. Como pode ser visto, o método MS-Unit teve o melhor ranking para o conjunto de dados TTP e o segundo melhor para os outros dois conjuntos de dados, em que o Ensemble obteve os menores rankings.

Para analisar o desempenho para cada par de regressores, na Figura 6.19 são exibidos os gráficos dos valores de NMSE obtidos pela aplicação dos algoritmos selecionados pelos métodos MS-Lote, MS-Unit e Default, além da abordagem Ensemble para os conjuntos de dados TTP, EDP e Airline. Como o Ensemble sempre prediz o valor de saída como a média das predições dos dois regressores, os valores de NMSE são exatamente iguais para as duas abordagens, lote e unitária. No entanto, o desempenho do método Default para SelUnit pode ser diferente de SelLote, porque a distribuição de classes dos conjuntos de

meta-dados gerados por essas abordagens pode ser diferente.

Para o conjunto de dados TTP, apesar do melhor desempenho de MS-Unit no nível meta, nota-se pequenas diferenças entre os valores de NMSE desses métodos no nível base. O método MS-Lote conseguiu uma pequena vantagem sobre MS-Unit para o par de regressores PPR/CART, enquanto o oposto foi observado para a maioria dos pares em que MARS é um dos algoritmos, principalmente para MARS/LR, MARS/RF e MARS/SVM. Essas diferenças são devidas às predições discrepantes cometidas pelo algoritmo MARS, que faz com que a seleção errada desse algoritmo para alguns exemplos gere um valor de NMSE muito maior do que quando o mesmo é selecionado corretamente. Como não houve grandes variações de NMSE dos métodos investigados, as diferenças para os métodos Default e Ensemble foram similares àquelas analisadas na Seção 6.1.



(a) TTP

(b) EDP

(c) Airline

Figura 6.19: Valores de NMSE obtidos pelos métodos MS-Lote, MS-Unit e Default e pela abordagem Ensemble para os conjuntos de dados TTP, EDP e Airline usando a estratégia de rotulação Sem-empate.

Observando os resultados para o conjunto de dados EDP (Fig. 6.19(b)), nota-se que o

método Ensemble ainda é superior aos demais métodos para a maioria dos pares de regressores. Contudo, é possível observar algumas reduções dos valores de NMSE do método MS-Unit em comparação com MS-Lote. Essas reduções, embora pequenas, ocorreram para pelo menos sete casos, enquanto o MS-Lote obteve uma vantagem clara apenas para dois pares. Esses resultados evidenciam a superioridade de MS-Unit para esse conjunto de dados. Outra observação é que, embora o ranking do MS-Lote e do Default sejam iguais, as diferenças quando o primeiro é melhor que o segundo são maiores do que quando o contrário ocorre.

Para o conjunto de dados Airline (Fig. 6.19(c)), esperava-se uma superioridade considerável do método MS-Unit, dadas as diferenças dos valores de Kappa no nível meta. Porém, assim como para EDP, as vantagens em relação ao método MS-Lote ocorreram para a maior parte dos pares de algoritmos por uma pequena diferença. Por exemplo, a estatística κ do método MS-Unit é muito maior do que a do método MS-Lote para o par MARS/CART (Fig. 6.12), mas os valores de NMSE de ambos são muito similares.

Apesar dos problemas conhecidos da abordagem SelLote, os resultados indicam que ela ainda é similar a SelUnit. Entretanto, devido a essa discordância do melhor algoritmo para lotes de exemplos e do melhor algoritmo para cada exemplo, a margem para a melhoria dessa abordagem é menor do que as possibilidades para o aperfeiçoamento da abordagem SelUnit. Por exemplo, a amostragem dos meta-exemplos de treinamento em SelUnit pode ser feita com base em algum critério, como eliminar os casos prováveis de ruído ou aqueles em que a diferença de desempenho entre os regressores é muito pequena. Nesta tese, a amostragem para reduzir o número de exemplos de treinamento foi aleatória, pois o objetivo era comparar SelLote e SelUnit da maneira mais justa possível.

6.3 Meta-atributos independentes e dependentes

Nesta tese, as características que podem ser extraídas dos dados do nível base são categorizadas em relação à morfologia dos dados, conforme apresentado na Seção 4.3.2. As características que são obtidas de diferentes conjuntos de dados, independentemente da morfologia desses dados, e podem ser usadas como meta-atributos são classificadas como independentes. Essa abordagem é comumente usada em aplicações convencionais de meta-aprendizado. Por exemplo, a assimetria média dos atributos numéricos e a entropia média dos atributos categóricos podem ser obtidas a partir de qualquer conjunto de dados e usadas como meta-atributos. Por outro lado, as características dependentes só podem ser usadas como meta-atributos se os dados tiverem a mesma morfologia. Exemplos dessas características são a assimetria de cada atributo numérico e a entropia de cada atributo nominal. Esse cenário ocorre em problemas de fluxos de dados, pois os dados são descritos pelos mesmos atributos preditivos ao longo do tempo. Assim, a cada instante de tempo, é possível extrair características para cada atributo ou das relações entre esses

atributos, sem a necessidade de agregação, ou seja, podem ser usadas diretamente como meta-atributos.

Nesta seção, são apresentados e analisados os resultados experimentais obtidos pelo método MetaStream com o objetivo de avaliar empiricamente a veracidade da terceira hipótese específica estabelecida nesta tese:

> *Na utilização de meta-aprendizado para dados com a mesma morfologia é possível usar características específicas dos dados, que contêm informação que contribui para melhorar a predição do melhor algoritmo para esses dados.*

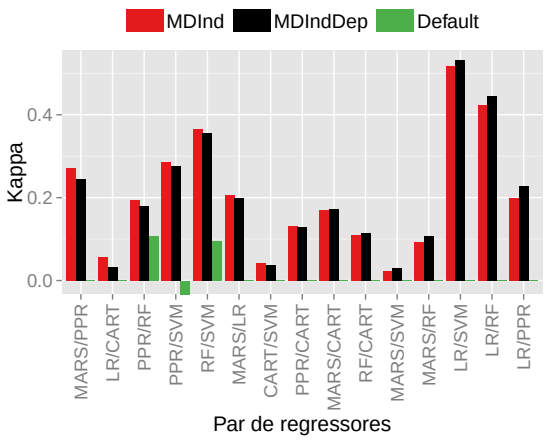
Esses experimentos foram realizados com dois conjuntos de meta-dados, MDInd e MDIndDep, conforme apresentado na Seção 5.3.1. Todos os meta-atributos do conjunto de meta-dados MDInd independem da morfologia dos dados, enquanto que o conjunto MDIndDep é formado pelos mesmos meta-atributos de MDInd mais os meta-atributos dependentes. Como as características independentes e dependentes são extraídas dos mesmos dados, pode haver uma grande correlação entre os meta-atributos. Portanto, apenas os meta-atributos dependentes que não são altamente correlacionados com os meta-atributos independentes são adicionados ao conjunto MDIndDep, conforme descrito na Seção 5.3.1. A seguir, são apresentados e discutidos os resultados para os níveis meta e base usando MDInd e MDIndDep para a estratégia de rotulação Sem-empate.

6.3.1 Nível meta

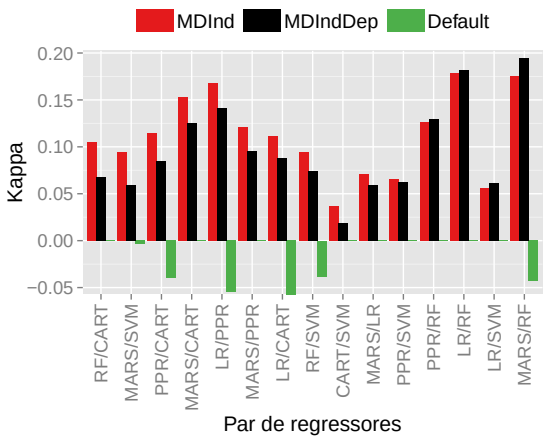
O objetivo principal dos experimentos e das análises realizadas nesta seção é avaliar se as características dependentes da morfologia dos dados possuem informação útil que possa contribuir para a melhoria do aprendizado do meta-modelo. Para isso, o desempenho preditivo dos métodos de seleção de algoritmos para cada conjunto de dados e par de regressores é avaliado no nível meta com a estatística κ. Essa medida permite avaliar se os resultados observados ocorreram ao acaso ou devido aos padrões aprendidos a partir dos meta-dados, pois ela considera a distribuição de classes. Porém, como mencionado na seção anterior, a desvantagem dessa medida em relação à taxa de erro é que as diferenças dos resultados verificados no nível meta não são boas estimativas do efeito da seleção de algoritmos para o desempenho preditivo no nível base. Ainda assim, a estatística κ será empregada, pois está mais alinhada com o objetivo da avaliação experimental realizada nesta seção, que é investigar a importância das características dependentes para o desempenho preditivo do método MetaStream.

Na Figura 6.20 são apresentados os valores da estatística κ do método MetaStream com os conjuntos de meta-dados MDInd e MDIndDep para os conjuntos de dados TTP, EDP e Airline. Os pares de regressores estão ordenados pela diferença dos valores de Kappa do método MetaStream usando os conjuntos de meta-dados MDInd e MDIndDep. Os

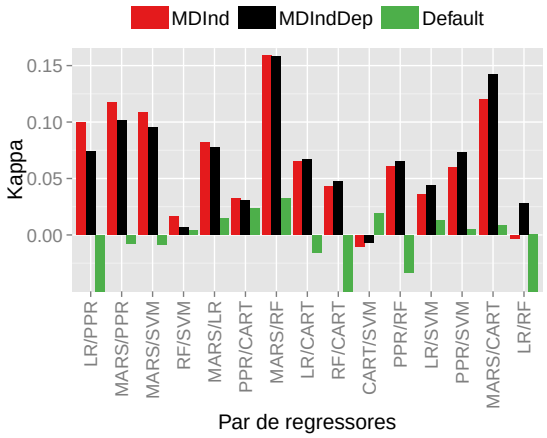
valores obtidos pelo método Default também são apresentados, com o intuito de facilitar a comparação com os resultados obtidos anteriormente, pois as predições feitas por esse método não são influenciadas pelos meta-atributos. É possível observar, nos três gráficos apresentados nessa figura, que os desempenhos preditivos do método MetaStream com MDInd e MDIndDep são muito similares, principalmente para os conjuntos de dados TTP e Airline. O teste estatístico de Wilcoxon apontou, com 95% de confiança, que o método MetaStream com o conjunto de meta-dados MDInd foi significativamente superior ao MetaStream com MDIndDep para o conjunto de dados EDP. Para as demais comparações, o teste não rejeitou a hipótese nula de que os desempenhos dos métodos são semelhantes. Esses resultados não apresentaram evidências suficientes para afirmar que a hipótese de que as características dependentes possuem informação útil que contribui para a melhoria do desempenho preditivo do método MetaStream é verdadeira.



(a) TTP.



(b) EDP.



(c) Airline.

Figura 6.20: Estatística κ dos métodos Default e MetaStream com os conjuntos de meta-dados MDInd e MDIndDep para os conjuntos de dados TTP, EDP e Airline usando o algoritmo RF como meta-aprendiz.

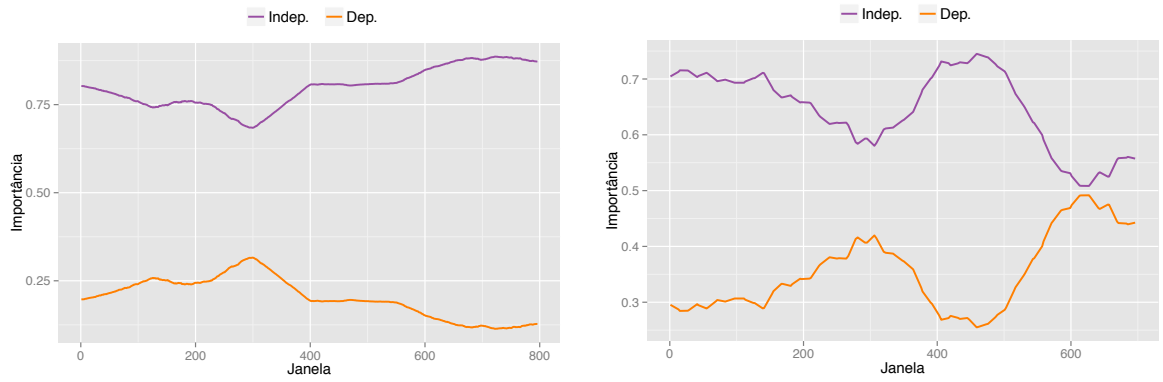
O fato do desempenho preditivo do método MetaStream não ter melhorado com a inclusão de características dependentes é, de certa maneira, surpreendente. Ao contrário do que se conjecturava, a adição dessas características, em alguns casos, influenciou negativamente o desempenho do MetaStream, como para o conjunto de dados EDP. O cálculo da correlação entre os meta-atributos independentes e dependentes mostrou que grande parte desses meta-atributos eram altamente correlacionados. Por esse motivo, um pequeno número de meta-atributos dependentes, considerando a grande quantidade que foi gerada, foi adicionado para formar MDIndDep. Mesmo que os meta-atributos com correlação maior que 0.9 tenham sido eliminados, outros tantos com menor redundância foram incluídos em MDIndDep, o que pode ter influenciado negativamente o desempenho do MetaStream. Esse grande número de meta-atributos correlacionados se deve, em partes, ao pequeno número de atributos dos conjuntos de dados investigados no nível base. Assim, se um conjunto de dados tem, por exemplo, apenas um atributo categórico, a entropia média (característica independente) é idêntica à entropia de cada atributo (característica dependente), ou seja, somente o meta-atributo independente será mantido. Essa questão dificulta a análise da importância das características dependentes.

Para avaliar detalhadamente a influência dos meta-atributos dependentes na indução do meta-modelo, analisou-se o peso que o meta-aprendiz RF atribuiu a cada meta-atributo do conjunto de dados MDIndDep. O peso de cada meta-atributo é calculado pelo algoritmo RF durante o processo de indução do meta-modelo de acordo com a redução média da acurácia (MDA) que cada meta-atributo proporcionou, como mencionado na Seção 6.2. A importância relativa média dos meta-atributos independentes e dependentes é calculada da seguinte maneira. Primeiramente, os meta-atributos são ordenados por ordem decrescente de peso (quanto maior, melhor), e apenas os primeiros 10% são selecionados para formar o conjunto S. Em seguida, a importância média dos meta-atributos independentes (dependentes) é calculada como a soma da importância dos meta-atributos independentes (dependentes) presentes no conjunto S dividido pela soma dos pesos de todos os meta-atributos em S.

Na Figura 6.21 são mostrados os gráficos de importância relativa média dos meta-atributos independentes e dependentes para os pares de regressores MARS/LR e RF/CART dos conjuntos de dados TTP e EDP, respectivamente. A importância é calculada ao longo do tempo usando uma janela deslizante de tamanho 100 e passo 1. Como é possível observar na Figura 6.21(a), referente ao primeiro par, a importância dos meta-atributos dependentes representa, aproximadamente, apenas 25% da importância total dos meta-atributos do conjunto S. Portanto, o meta-aprendiz RF atribuiu pesos maiores para as medidas independentes do que as dependentes durante o processo de indução do meta-modelo.

O cenário para o par RF/CART é similar ao que foi observado para o par MARS/LR, mas, para RF/CART, houve uma variação maior da importância dos meta-atributos.

Próximo à janela 600, a proporção da importância das medidas independentes e dependentes alcançou 50%. Durante esse período, nota-se também uma melhora no desempenho preditivo do método MetaStream usando MDIndDep em relação à MDInd. Esse comportamento também foi observado em alguns outros pares em que os meta-atributos dependentes tiveram grande importância. Entretanto, essa relação não é clara para a maioria dos casos investigados. Uma explicação é a instabilidade da medida MDA (Calle e Urrea, 2010), mas outros fatores que não foram considerados nessa análise, como o processo de aprendizado do meta-aprendiz RF, também podem ter interferido nesses resultados.



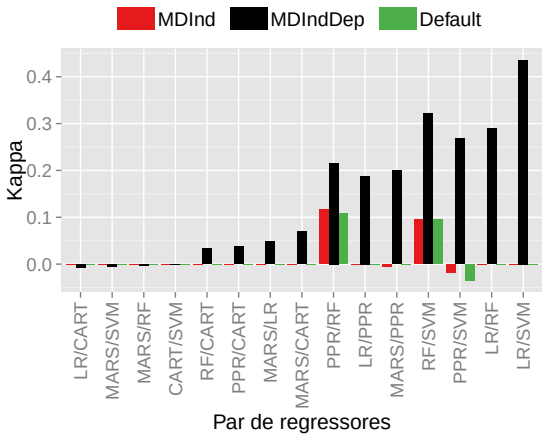
(a) TTP. (b) EDP.

Figura 6.21: Importância relativa dos meta-atributos independentes e dependentes para os pares de regressores MARS/LR e RF/CART dos conjuntos de dados TTP, e EDP, respectivamente. A importância é calculada usando uma janela deslizante de tamanho 100 e passo 1.

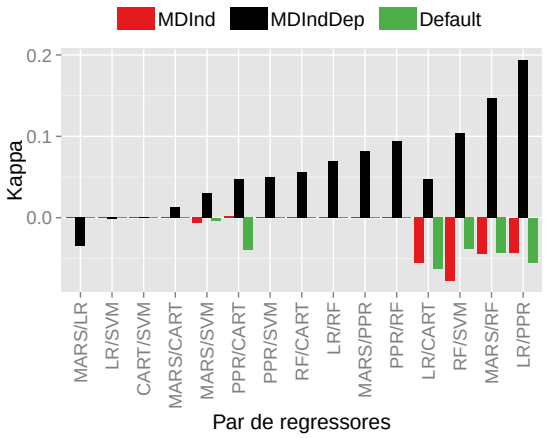
A pequena influência da adição dos meta-atributos dependentes também se deve à robustez do algoritmo RF, que foi usado como meta-aprendiz. Tal robustez já foi notada nos experimentos preliminares para seleção de parâmetros (Sec. 5.4), assim como foi verificado que o meta-aprendiz SVM é sensível ao conjunto de meta-atributos selecionados. Portanto, a seguir, é analisada a influência da inclusão dos meta-atributos dependentes no desempenho do método MetaStream usando o algoritmo SVM como meta-aprendiz. Na Figura 6.22 são apresentados os valores de Kappa do método MetaStream usando os meta-dados MDInd e MDIndDep para os conjuntos de dados TTP, EDP e Airline. Os pares de algoritmos de cada gráfico estão ordenados pela diferença da estatística κ do método MetaStream usando MDInd e MDIndDep.

Os gráficos da estatística κ para os conjuntos de dados TTP (Fig. 6.22(a)) e EDP (Fig. 6.22(b)) possuem características semelhantes no sentido de que o desempenho do método MetaStream com o meta-aprendiz SVM apresentou uma grande melhora com o conjunto de meta-dados MDIndDep em comparação com o conjunto MDInd. Enquanto o comportamento do método MetaStream é muito semelhante ao do método Default para todos os pares de algoritmos usando o conjunto MDInd, o seu desempenho preditivo foi

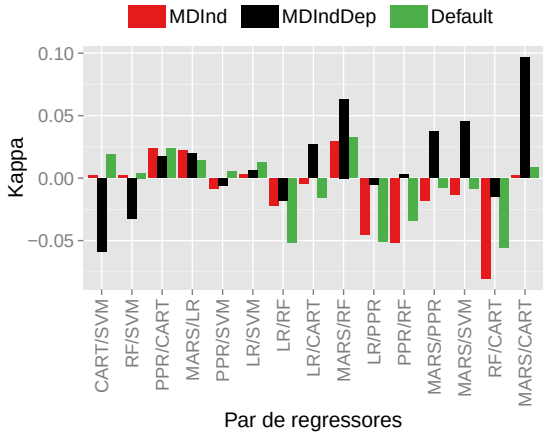
consideravelmente melhor com o conjunto MDIndDep. Essa melhora não ocorreu apenas para alguns pares de algoritmos. Observando os gráficos para os conjuntos de dados TTP e EDP, nota-se também que os valores da estatística Kappa obtidos com o conjunto de meta-dados MDIndDep são, em grande maioria, devidos ao conhecimento extraído desses dados e não ao acaso, como ocorre quando o conjunto MDInd é usado. Esses valores são, inclusive, comparáveis àqueles obtidos quando o algoritmo RF foi usado como meta-aprendiz (Fig. 6.20). Essas observações são reforçadas pelo teste estatístico de Wilcoxon, que apontou que os desempenhos obtidos com MDInd e MDIndDep são significativamente diferentes com 95% de confiança.



(a) TTP.



(b) EDP.



(c) Airline.

Figura 6.22: Estatística κ dos métodos Default e MetaStream com os conjuntos de meta-dados MDInd e MDIndDep para os conjuntos de dados TTP, EDP e Airline usando o algoritmo SVM como meta-aprendiz.

Diferentemente do que foi verificado para os conjuntos de dados TTP e EDP, não é possível notar uma melhora consistente para o conjunto de dados Airline quando as medidas dependentes são usadas (Fig. 6.22(c)). Embora a adição dessas medidas tenham

influenciado o comportamento do MetaStream, os valores de Kappa ainda são, geralmente, próximos de zero. Isso indica que, mesmo com a adição dos meta-atributos dependentes, o meta-aprendiz SVM não foi capaz de induzir um meta-modelo adequado para a tarefa de seleção de algoritmos e, consequentemente, suas predições foram realizadas ao acaso para a maioria dos pares analisados. Essa observação também é confirmada pelo teste de Wilcoxon, que não encontrou evidências suficientes para rejeitar a hipótese nula de que as diferenças de desempenho do método MetaStream com MDInd e MDIndDep são comparáveis. De fato, o conjunto de dados Airline foi o que apresentou maiores dificuldades para o aprendizado no nível meta, inclusive quando o meta-aprendiz RF foi utilizado (Fig. 6.20).

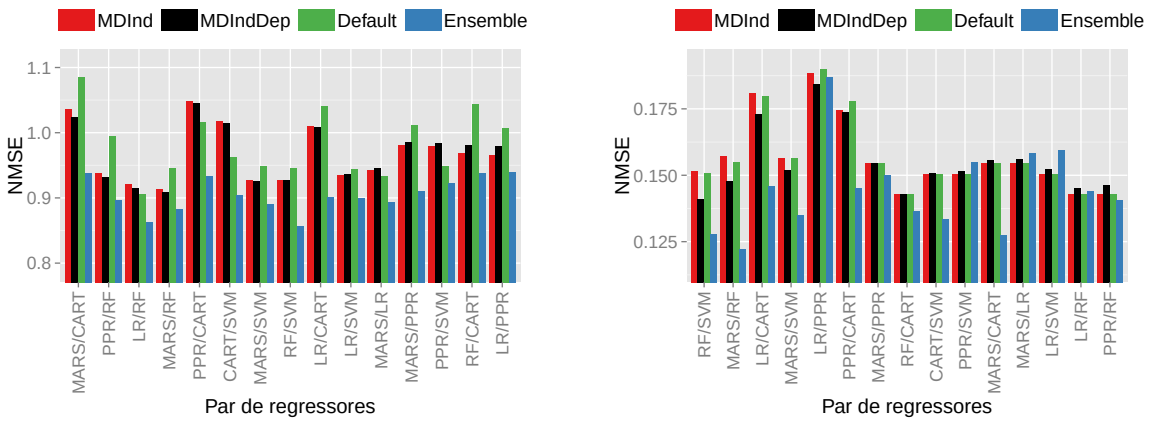
Os resultados obtidos com o meta-aprendiz SVM evidenciam que os meta-atributos dependentes podem fornecer informação útil para a melhoria do desempenho preditivo do MetaStream, ao contrário do que tinha sido verificado para o meta-aprendiz RF. A relevância desses meta-atributos depende, principalmente, das características dos dados do nível base, das medidas usadas para extrair informações desses dados e do algoritmo usado como meta-aprendiz. Uma das implicações da extração de características dependentes é a grande quantidade de meta-atributos que podem ser gerados. Essa quantidade depende, particularmente, do número de atributos que descrevem o problema no nível base e das medidas que serão empregadas para a extração de características. Portanto, a seleção de um subconjunto de meta-atributos é um processo que pode melhorar o desempenho do aprendizado no nível meta, como observou-se nos experimentos de ajuste de parâmetros (Seção 5.4), e facilitar a investigação da importância desses meta-atributos. Essa questão, assim como a análise mais aprofundada da importância dos meta-atributos, serão investigados em trabalhos futuros.

6.3.2 Nível base

O objetivo principal dos experimentos realizados com os conjuntos de meta-dados MDInd e MDIndDep era avaliar a relevância das características dependentes para o desempenho preditivo do método MetaStream no nível meta, o que foi feito na seção anterior. Entretanto, também faz parte do objetivo desta tese avaliar o efeito dos métodos de seleção de algoritmos no nível base. A medida usada na avaliação dos resultados do nível meta não permite estimar a influência da seleção de algoritmos no nível base e, por isso, as diferenças observadas no nível base podem ser totalmente distintas daquelas observadas no nível meta. Portanto, nesta seção, analisa-se o impacto da seleção de algoritmos no nível base quando o método MetaStream é usado com os conjuntos de meta-dados MDInd e MDIndDep. Assim como no nível meta, o desempenho do método MetaStream no nível base é avaliado para os meta-aprendizes RF e SVM.

Quando o algoritmo RF é utilizado como meta-aprendiz, os valores de NMSE para

o método MetaStream são muito semelhantes para os dois conjuntos de meta-dados estudados. Por exemplo, para o conjunto de dados Airline, mostrado na Figura 6.23(a), pode-se verificar que os valores de NMSE são praticamente os mesmos para MDInd e MDIndDep para todos os pares de algoritmos. Como mencionado no nível meta, esses resultados são devidos à robustez do meta-aprendiz RF e da correlação entre a maioria dos meta-atributos dos dois conjuntos de meta-dados investigados. Resultados similares aos apresentados nessa figura também foram obtidos para os outros conjuntos de dados. Embora a medida de avaliação usada no nível meta não permita uma estimativa do efeito que a seleção de algoritmos terá no nível base, a semelhança de comportamento do meta-aprendiz RF com os conjuntos MDInd e MDIndDep no nível meta sugeriam que os resultados no nível base também seriam similares, o que de fato foi verificado.



(a) Conjunto de dados: Airline - Meta-aprendiz: RF.

(b) Conjunto de dados: EDP - Meta-aprendiz: SVM.

Figura 6.23: Valores de NMSE do método MetaStream para os conjuntos de meta-dados MDInd e MDIndDep, do método Default e da abordagem Ensemble para os conjuntos de dados Airline e EDP. Para o primeiro conjunto, o algoritmo RF é usado como meta-aprendiz, enquanto que o segundo, o algoritmo SVM é empregado como meta-aprendiz.

Quando o algoritmo SVM é empregado como meta-aprendiz, os valores de NMSE com MDInd são muito similares aos do Default, assim como ocorreu também no nível meta. Por outro lado, o método MetaStream com o conjunto de meta-dados MDIndDep teve, geralmente, comportamento distinto do Default, como pode ser visto no gráfico dos valores de NMSE para o conjunto de EDP, mostrado na Figura 6.23(b). Apesar desses resultados, as diferenças para os valores de NMSE entre os métodos MetaStream e Default não foram tão expressivas como aquelas obtidas para os valores de Kappa no nível meta. Além disso, essa melhora do método MetaStream quando os meta-dados MDIndDep são usados pelo meta-aprendiz SVM não foi suficiente para superar o desempenho preditivo em relação ao meta-aprendiz RF, independentemente do conjunto de meta-dados. Portanto, não houve uma melhora geral do sistema de aprendizado no nível base, como pode ser notado na

Figura 6.23, tomando como base o desempenho da abordagem Ensemble.

Os resultados apresentados nesta seção mostraram que, em geral, as melhoras obtidas no nível meta, como o uso de medidas dependentes para o meta-aprendiz SVM, tiveram um pequeno impacto no nível base. Essa discordância pode ser em razão das diferenças entre o problema tratado no nível meta e o objetivo a ser alcançado no nível base. Uma dessas diferenças se deve ao fato da magnitude dos erros dos regressores não ser considerada, como já foi mencionado nas seções anteriores. Tratar o problema no nível meta como sendo de regressão ao invés de classificação pode estreitar a relação entre os dois níveis de aprendizado. Porém, os erros discrepantes que os modelos preditivos do nível base podem apresentar em aplicações reais é uma questão que deve ser considerada nesse caso.

6.4 Ranking de algoritmos

Até a seção anterior, os métodos de seleção de algoritmos foram usados para predizer o melhor dentre um par de algoritmos disponíveis, sendo que cada par consistia em um problema de meta-classificação diferente. A opção por utilizar pares de algoritmos foi motivada pela possibilidade de induzir um meta-modelo específico para cada combinação de algoritmos, que constituem problemas de meta-classificação supostamente menos complexos do que se todos os algoritmos fossem considerados, e a oportunidade de analisar uma quantidade maior de problemas a fim de identificar os fatores mais importantes para o sucesso ou fracasso dos métodos de seleção. Em uma aplicação real, a recomendação de algoritmos normalmente ocorre a partir de um conjunto maior de algoritmos. Nesse caso, outras formas de sugestão descritas no Capítulo 3 podem ser empregadas, como rankings (Soares, 2004). A recomendação por rankings é mais flexível do que a seleção de um único algoritmo, pois fornece uma lista ordenada pelo desempenho estimado dos algoritmos ao usuário, que pode empregar um ou uma combinação deles para tratar o problema no nível base.

A recomendação de algoritmos por meio de rankings tem sido estudada em diversos trabalhos recentes de meta-aprendizado e diferentes abordagens para a construção de rankings têm sido propostas e usadas, como por meio de regressão (Köpf et al., 2000; De Souto et al., 2008), árvores de decisão (Todorovski et al., 2002; Souza, 2010) ou aprendizado baseado em instâncias (k-NN) (Sohn, 1999; Soares e Brazdil, 2000; Kanda, 2012). Nesta seção, são apresentados e discutidos os resultados experimentais de uma abordagem para a construção de rankings de algoritmos. O objetivo é obter evidências sobre a veracidade da quarta hipótese estabelecida neste trabalho:

> *As predições realizadas pelo método MetaStream podem ser combinadas para a recomendação de algoritmos em forma de rankings com desempenho superior às predições do método Default.*

A abordagem de construção de rankings avaliada nesta seção é baseada em Kalousis (2002) e consiste em contabilizar o número de vezes que cada algoritmo é selecionado considerando todos os pares de algoritmos. Essa abordagem é doravante denominada **MS-Rank** e as predições dos métodos de seleção de algoritmos para a rotulação Sem-empate são usadas para a criação dos rankings. Quanto mais vezes um algoritmo é selecionado, melhor será sua posição no ranking, ou seja, maior será sua preferência sobre os demais. Seja m o número de algoritmos empregados no nível base, então um total de $\binom{m}{2}$ pares são formados. Cada algoritmo L_i, $i = \{1, \ldots, m\}$, faz parte de $m-1$ pares e, portanto, pode ser selecionado como o melhor algoritmo, no máximo, $m-1$ vezes. O ranking dos algoritmos é definido por um vetor $R = \{R_1, R_2, \ldots, R_m\}$, em que R_i é o ranking do algoritmo L_i. Seja w_i o número de vezes que o algoritmo L_i é selecionado, a sua posição no ranking será melhor do que um outro algoritmo L_j, $j = \{1, \ldots, m\}$, se $w_i > w_j$, para $j \neq i$. Em caso de empate, $w_i = w_j$, o ranking médio é utilizado para os dois algoritmos $w_i = w_j = \frac{w_i + w_j}{2}$. Por exemplo, considere quatro algoritmos, $L = \{L_1, L_2, L_3, L_4\}$, que formam seis pares: L_1/L_2, L_1/L_3, L_1/L_4, L_2/L_3, L_2/L_4 e L_3/L_4. Se o algoritmo L_1 for selecionado pelo método MetaStream em todos os pares dos quais faz parte (L_1/L_2, L_1/L_3, L_1/L_4), o algoritmo L_3 for selecionado em L_2/L_3 e L_3/L_4, e o algoritmo L_4 for selecionado em L_2/L_4, então $w_1 = 3$, $w_2 = 0$, $w_3 = 2$ e $w_4 = 1$. Consequentemente, o ranking dos algoritmos em L será $R = \{1, 4, 2, 3\}$.

Essa abordagem de sugestão de rankings foi preferida em relação a outras mencionadas no Capítulo 3 porque permite utilizar as predições feitas para os pares de algoritmos na construção da recomendação global. Kalousis (2002) também constrói rankings de algoritmos a partir das predições aos pares e avalia se o sistema de meta-aprendizado teve ou não sucesso de duas maneiras. Na primeira, a qual o autor chama de *strict accuracy*, a sugestão fornecida é considerada correta apenas se o primeiro algoritmo do ranking é igual ao algoritmo com o melhor desempenho global. Na segunda, o autor considera que os algoritmos mais bem posicionados no ranking verdadeiro podem ter desempenhos semelhantes, ou seja, não são significativamente diferentes. Nesse caso, se o primeiro algoritmo do ranking construído com as predições do sistema de meta-aprendizado for igual a qualquer um desses algoritmos com desempenhos semelhantes, a recomendação é considerada correta.

Nesta tese, considera-se que o ranking fornecido pelo sistema de meta-aprendizado pode ser utilizado para que um ou mais algoritmos, ou mesmo uma combinação deles, sejam aplicados para a predição no nível base. Entretanto, experimentos com algumas dessas possibilidades são deixadas para trabalhos futuros e a avaliação nesta seção é realizada somente no nível meta. Essa avaliação consiste em comparar o ranking criado a partir das predições dos métodos de seleção de algoritmos MetaStream e Default, denominado de ranking predito, com o ranking verdadeiro (ideal), que é obtido após a observação do desempenho preditivo dos algoritmos no nível base. Neste trabalho, a similaridade en-

Tabela 6.5: Média dos coeficientes de correlação de Spearman entre os rankings criados a partir das predições dos métodos de seleção de algoritmos e o ranking verdadeiro.

Conjunto de dados	MetaStream	Default	Diferença
TTP	0.542	0.494	0.048
EDP	0.325	0.296	0.029
Airline	0.113	0.100	0.013

tre o ranking predito e o ranking verdadeiro é medida pelo coeficiente de correlação de Spearman ρ (Neave e Worthington, 1992), que também foi utilizado em diversos trabalhos com o mesmo propósito (Brazdil et al., 2003; Soares, 2004; De Souto et al., 2008; Souza, 2010; Kanda, 2012). Com o propósito de obter mais evidências sobre as similaridades entre os rankings preditos por ambos os métodos e o ranking real, a medida de correlação *Weighted Goodman-Kruskal* (WGK) (Campello e Hruschka, 2009) também foi utilizada na avaliação. A WGK calcula a correlação entre duas sequências de valores considerando a magnitude dos valores contidos em cada sequência, ou seja, a taxa de variação desses valores também é considerada, e não apenas a tendência (Kanda, 2012). Como as correlações obtidas com a WGK foram muito similares com aquelas calculadas pela correlação de Spearman, apenas os valores dessa última são analisados a seguir, por ser a medida mais comumente usada.

Na Tabela 6.5, são apresentados os coeficientes de correlação de Spearman para os métodos MetaStream e Default e a diferença entre eles para os três conjuntos de dados investigados. Como é possível observar, a correlação entre os rankings preditos pelo método MetaStream são sempre maiores do que aqueles preditos pelo Default, mas as diferenças entre eles são sempre inferiores a 0.05. Esses resultados são coerentes com aqueles apresentados nas seções anteriores, pois, como houve um equilíbrio entre os desempenhos dos métodos MetaStream e Default considerando todos os pares de algoritmos, esperava-se que os desempenhos de ambos os rankings preditos fossem similares. A maior correlação entre os rankings preditos e verdadeiros foram observados para o conjunto de dados TTP, com coeficiente médio de aproximadamente 0.5 e em segundo para o conjunto de dados EDP, cujos valores são de aproximadamente 0.3. A magnitude desses coeficientes confirmam que os rankings não foram preditos ao acaso, mesmo considerando os pares para os quais os métodos de seleção de algoritmos não conseguiram desempenhos satisfatórios. Ao contrário dos resultados para os conjuntos de dados TTP e EDP, a correlação entre os rankings preditos e real foi muito pequena (aproximadamente 0.1) para o conjunto de dados Airline. Apesar de pequena, essa correlação não ocorreu ao acaso, pois ao gerar 1000 rankings aleatórios para cada ranking predito, a maior correlação média observada foi de 0.05.

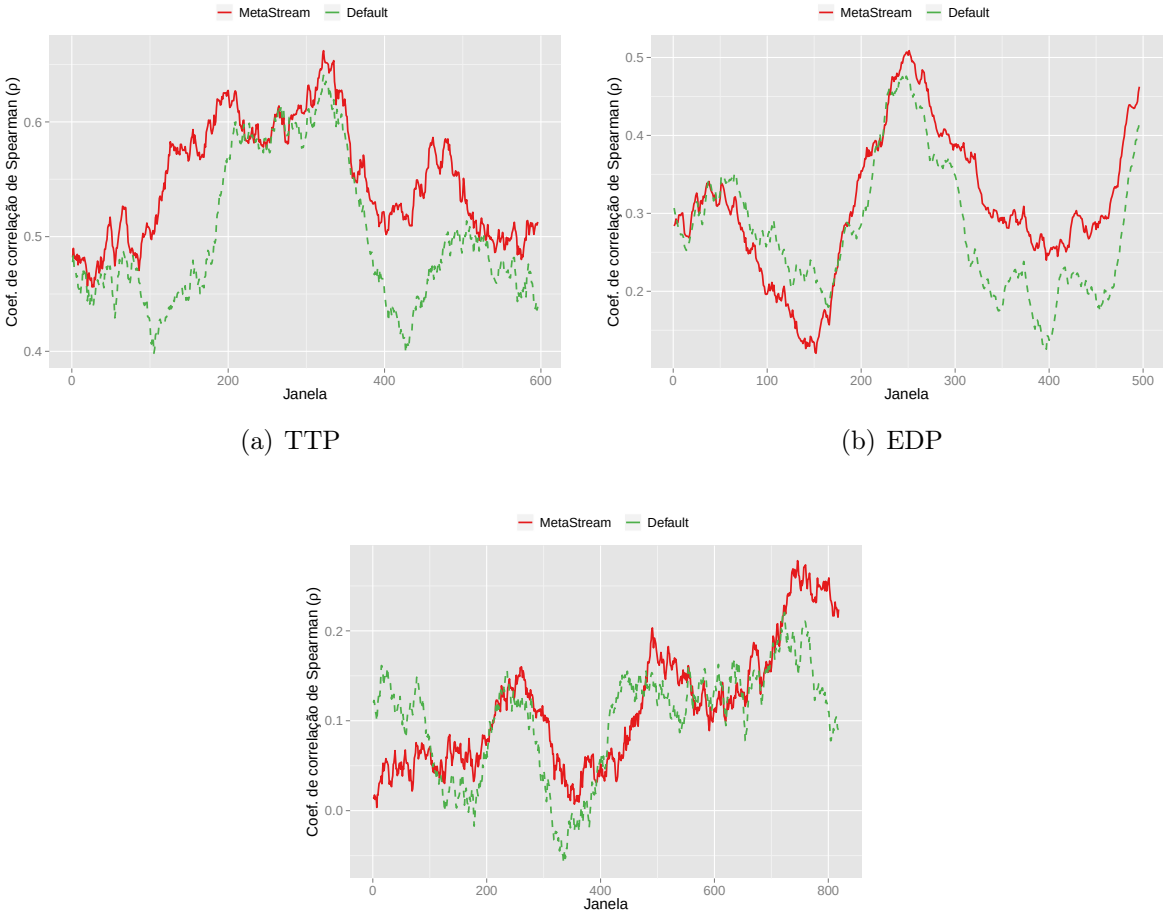
Para analisar a evolução do desempenho dos rankings construídos com as predições dos métodos de seleção de algoritmos ao longo do tempo, os coeficientes de correlação são

apresentados na Figura 6.24 para cada conjunto de dados usando uma janela de tamanho 100 e passo 1, como nas seções anteriores. Para o conjunto de dados TTP (Fig. 6.24(a)), observa-se que a correlação do método MetaStream é maior do que a do Default durante quase todo o fluxo de dados. Embora os comportamentos das curvas sejam similares, o método MetaStream se recuperou mais rapidamente do que o Default após um período de queda, que pode corresponder à mudanças nos dados. O intervalo do coeficiente de correlação para o conjunto de dados EDP (Fig. 6.24(b)) é maior do que o do TTP e os coeficientes de correlação variam mais rapidamente. Durante a primeira metade do gráfico, os desempenhos de ambos os métodos são similares, com pequena vantagem para o Default. Na segunda metade, o cenário se inverte e o método MetaStream passa a ter maior coeficiente correlação, principalmente entre as janelas 350 e 450. Por último, para o conjunto de dados Airline 6.24(c), nota-se um equilíbrio entre os coeficientes de correlação dos dois métodos, sendo que as maiores diferenças ocorreram logo no início e no final do fluxo. As curvas dos coeficientes de correlação para esse conjunto de dados apresentam, em geral, uma tendência de aumento, principalmente a curva do método MetaStream.

Além de mensurar a similaridade entre cada ranking predito e o ranking real, analisou-se também a posição média que cada algoritmo ocupa nos rankings. Nos gráficos da Figura 6.25 é possível visualizar a posição média de cada algoritmo para os rankings dos métodos MetaStream e Default e para o ranking real ao longo do tempo para os conjuntos de dados TTP (superior) e Airline (inferior). Esses gráficos foram novamente gerados usando uma janela deslizante de tamanho 100 e passo 1.

Para o conjunto de dados TTP (Fig. 6.25(a)), observa-se que os algoritmos CART e LR ocupam, na média, a segunda e a penúltima posição do ranking ao longo do tempo, respectivamente. A posição média de alguns algoritmos, como o próprio LR, nos rankings preditos e real é muito estável, ao contrário, por exemplo, do PPR, que ocupava, na média, a segunda posição no início do fluxo e chegou a ocupar a última próximo da janela 400. Outra característica clara nessa figura é a pequena diferença entre as curvas da posição média obtida a partir do ranking real e a posição média do ranking predito pelo método MetaStream para todos os algoritmos. Embora os métodos MetaStream e Default tenham obtido correlação média similares (Fig. 6.24(a)), a média dos rankings gerados pelas predições do MetaStream para cada algoritmo é mais próxima da real. Essa diferença pode ser em razão da grande variação que existe quando a similaridade entre o ranking predito e real é calculada para um pequeno número de algoritmos, como acontece no cálculo da correlação entre o ranking predito e real para cada meta-exemplo. Por exemplo, se em um ranking com 6 algoritmos, apenas as posições 2 e 6 são trocadas, a correlação será de apenas 0.086, enquanto que, em uma janela com 100 meta-exemplos (tamanho da janela usada nos gráficos da Figura 6.25), uma troca de posições, mesmo entre os extremos do ranking, terá uma influência muito menor.

Pelo gráfico das posições médias do ranking de cada algoritmo para o conjunto de



(a) TTP

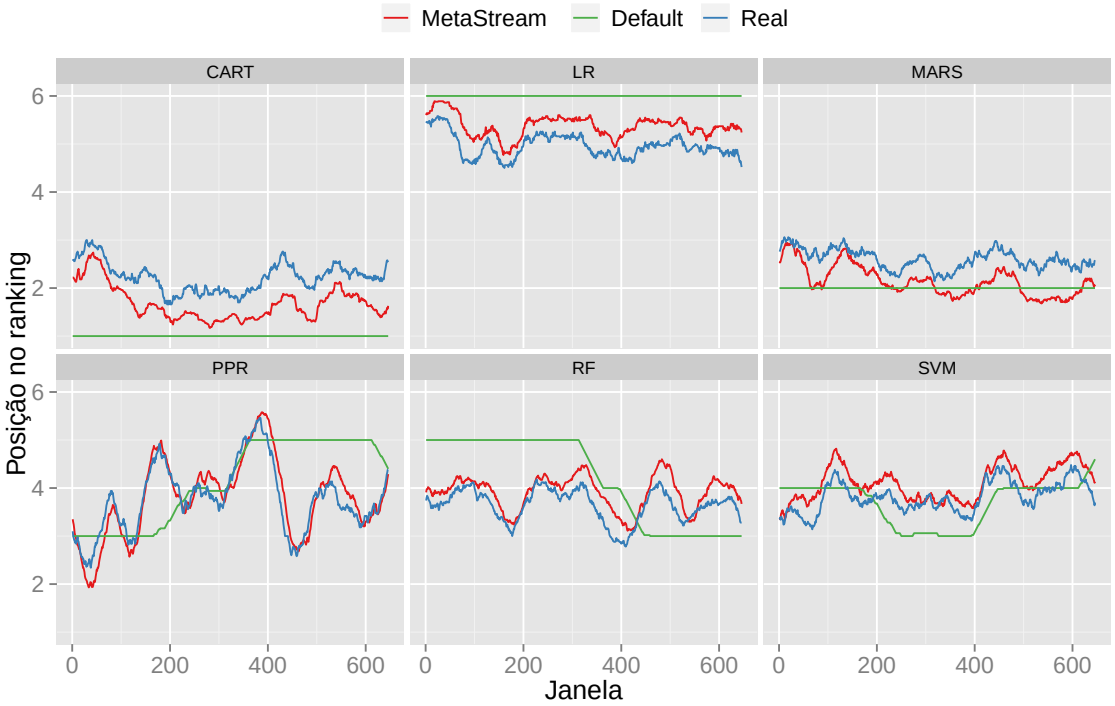
(b) EDP

(c) Airline

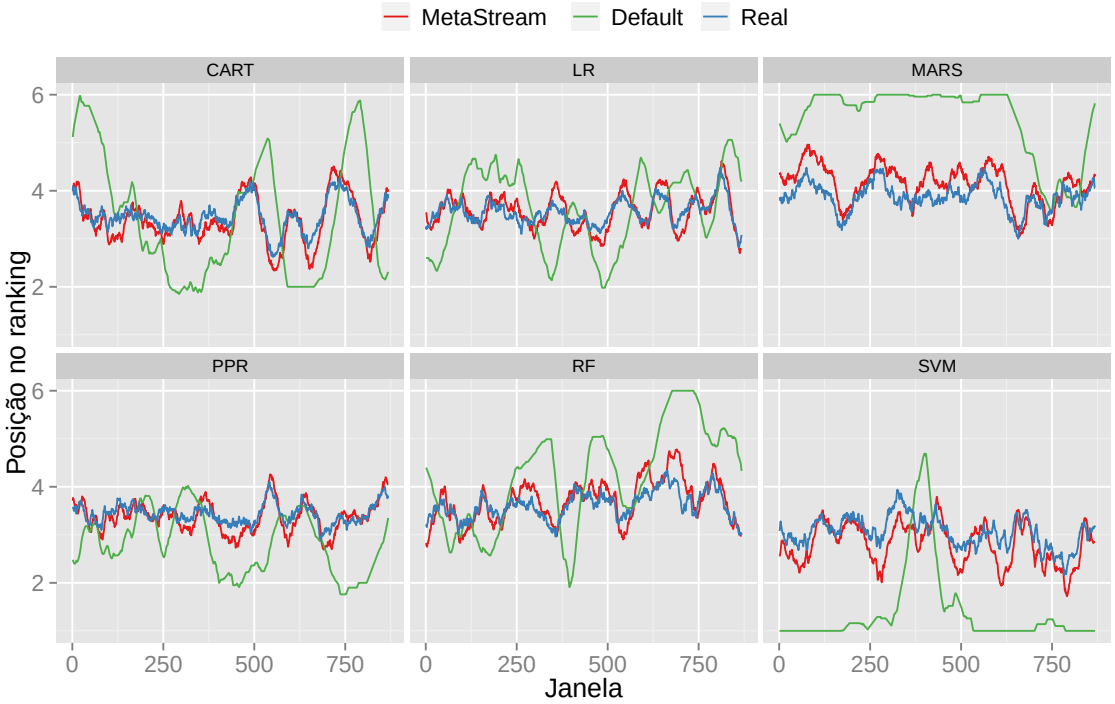
Figura 6.24: Coeficiente de correlação de Spearman entre os rankings preditos pelos métodos de seleção de algoritmos e o ranking verdadeiro ao longo do tempo para os conjuntos de dados TTP, EDP e Airline usando uma janela deslizante de tamanho 100 e passo 1.

dados Airline 6.25(b), é possível observar que a curva do ranking médio predito pelo método MetaStream está muito mais próxima da curva do ranking real do que a curva do Default. A diferença entre os dois métodos de seleção de algoritmos é clara para todos os algoritmos, mas principalmente para MARS e SVM. Essa superioridade do método MetaStream em comparação com o Default diverge do que foi observado no gráfico dos coeficientes de correlação (Fig. 6.25). Os motivos para essa divergência podem ser os mesmos apontados para o caso do conjunto de dados TTP. Uma outra característica que pode ser notada na Figura 6.25(b) é que as posições médias reais dos algoritmos são muito similares. A posição de todos os algoritmos varia sempre entre 3 e 4 e as maiores diferenças ocorreram para os algoritmos MARS e SVM. Esses resultados se devem à grande variação da posição de cada algoritmo nos rankings.

Apesar dos resultados promissores conseguidos com a construção de rankings usando as predições do método MetaStream para pares de algoritmos, a recomendação sugerida



(a) TTP



(b) Airline

Figura 6.25: Posição média de cada algoritmo nos rankings preditos e real para os conjuntos de dados TTP e Airline usando uma janela deslizante de tamanho 100 e passo 1.

por meio desses rankings não são aplicadas no nível base. Como foi discutido no início desta seção, a recomendação de algoritmos em forma de rankings abre inúmeras possibilidades para que a sugestão fornecida possa ser empregada no nível base. Uma delas é o método *Top-N*, que simula um usuário que utiliza os primeiros algoritmos do ranking sucessivamente, até que todos os N algoritmos sejam testados. Esse método também considera o custo computacional de testar os N algoritmos. Uma outra possibilidade é a combinação de algoritmos seguindo a recomendação apresentada por meio do ranking. Essa combinação pode ser realizada, por exemplo, atribuindo peso a cada algoritmo de acordo com seu desempenho passado (Brazdil et al., 2009; Lemke e Gabrys, 2010). A aplicação dos algoritmos recomendados por rankings no problema do nível base será investigada em trabalhos futuros.

6.5 Considerações finais

Neste capítulo foram apresentados e analisados os resultados da avaliação experimental do método proposto, com o propósito de obter evidências sobre a veracidade das hipóteses estabelecidas no início desta tese. Em cada seção do capítulo foi tratada uma das quatro hipóteses específicas estabelecidas na introdução desta tese.

Inicialmente, na Seção 6.1, o método MetaStream foi avaliado para o problema de seleção de algoritmos para lotes de exemplos e foi comparado ao método de referência Default, que consiste em predizer o melhor algoritmo com base na classe majoritária dos meta-dados. Em geral, as taxas de erro de classificação obtidas no nível meta considerando todos os pares de regressores indicaram um equilíbrio entre os dois métodos. Entretanto, o desempenho preditivo do MetaStream geralmente é maior que o do Default para pares em que a seleção do melhor algoritmo não é um problema trivial, ou seja, quando não há um regressor que é claramente superior ao outro ao longo de todo o fluxo. No nível base, os erros produzidos pelo método proposto foram menores do que o Default. No entanto, o desempenho do sistema de meta-aprendizado ainda está aquém do desempenho obtido quando, ao invés de escolher um único algoritmo, a média das predições dos dois regressores sob análise é usada para predizer a saída desejada. Essa abordagem se mostrou muito promissora para dois, dos três conjuntos de dados estudados. No entanto, e embora ainda haja muitos aspectos que podem ser investigados para melhorar o desempenho do sistema, os resultados demonstram claramente que é possível utilizar uma abordagem de meta-aprendizado para selecionar algoritmos em FCD. Isto significa também que, como em outros contextos, em FCD também é possível construir modelos que relacionam as características dos dados com o desempenho dos algoritmos de aprendizado.

Em seguida, a abordagem de seleção de algoritmos para cada exemplo do nível base (SelUnit) foi investigada na Seção 6.2. Como a caracterização dos dados para a seleção de algoritmos para lotes de exemplos (SelLote) e SelUnit são diferentes, as predições realiza-

das com SelLote precisaram ser adaptadas para a comparação com SelUnit no nível meta. No nível base, as duas abordagens podem ser comparadas diretamente, pois os meta-exemplos usados para treinamento e teste em cada instante de tempo foram extraídos a partir dos mesmos dados do nível base. Da comparação no nível meta, observou-se que o método MetaStream obteve melhor desempenho preditivo com a abordagem SelUnit do que com SelLote de acordo com a estatística Kappa. Embora o MetaStream também tenha conseguido desempenho superior no nível base com SelUnit, essa vantagem não é tão clara como no nível meta. As diferenças entre os dois níveis de aprendizado se devem, em partes, às adaptações necessárias para comparação das abordagens no nível meta e à fraca relação entre os problemas e as medidas de avaliação de cada nível. Isso mostra a importância de fazer a avaliação de sistemas de meta-aprendizado ao nível base e não só no nível meta. Os resultados obtidos mostram que é possível modelar a relação entre as características dos dados e o desempenho dos algoritmos em exemplos individuais, e não apenas em lotes de exemplos, como foi investigado na primeira hipótese. Este resultado abre perspectivas de melhorias significativas em problemas de previsão em FCD pela seleção do melhor algoritmo para cada exemplo a processar.

A influência dos meta-atributos dependentes da morfologia dos dados no comportamento do método MetaStream foi investigada na Seção 6.3. Os primeiros experimentos foram realizados apenas com o meta-aprendiz RF, mas nenhuma alteração foi notada quando os meta-atributos dependentes foram adicionados ao conjunto de meta-dados formado, inicialmente, apenas por meta-atributos independentes. Porém, quando o algoritmo SVM foi empregado como meta-aprendiz, os meta-atributos dependentes possibilitaram a indução de meta-modelos que alcançaram desempenho preditivo similar ao do meta-aprendiz RF. Sem a adição de tais meta-atributos dependentes, o meta-aprendiz SVM comportava-se como o Default, predizendo sempre a classe majoritária. Devido ao pequeno número de atributos preditivos dos conjuntos de dados usados, uma grande quantidade de meta-atributos independentes e dependentes eram idênticos ou altamente correlacionados. Portanto, apesar dos resultados obtidos com o meta-aprendizado usando SVM indicarem que é possível obter melhorias de desempenho combinando meta-atributos dependentes da morfologia dos dados com os tradicionais (ou seja, independentes da morfologia), será necessário um estudo mais detalhado e com conjuntos de dados com um maior número de atributos preditivos para confirmar a utilidade dos meta-atributos dependentes.

Por último, as predições realizadas pelos métodos MetaStream e Default para cada par de algoritmos foram usadas para construir rankings de algoritmos, semelhante ao que foi proposto por Kalousis (2002). A similaridade entre os rankings preditos e o ranking verdadeiro foi mensurada pelo coeficiente de correlação de Spearman. Para os três conjuntos de dados, a correlação média dos rankings construídos com as predições do método MetaStream são razoavelmente maiores do que aquelas do Default. É importante

lembrar que esses resultados são, na verdade, um reflexo do desempenho desses métodos na seleção do melhor algoritmo considerando pares de algoritmos. Esses resultados mostram que é possível construir rankings de algoritmos com base em modelos do desempenho entre pares desses algoritmos para FCD.

Capítulo 7

Conclusões

O crescente interesse por técnicas de AM em diferentes áreas do conhecimento acelerou o desenvolvimento de novos algoritmos de aprendizado. A necessidade de novos algoritmos se deve ao fato de que cada um possui um conjunto de suposições embutidas, as quais podem ser as mais adequadas para uma grande quantidade de problemas, mas nunca para todos (Wolpert, 1996). Com o aumento do número de algoritmos disponíveis, há uma grande probabilidade de que um deles seja adequado para as características dos dados sob análise (Murphy, 2012). Porém, para descobrir qual é esse algoritmo, o usuário final precisa do auxílio de um especialista da área ou pode fazer a seleção por tentativa e erro. Nenhuma dessas alternativas é satisfatória para o usuário, que espera ter acesso às técnicas de AM com baixo custo e de maneira efetiva.

Meta-aprendizado tem sido empregado com sucesso para seleção automática de algoritmos de AM para diferentes conjuntos de dados. Métodos baseados em meta-aprendizado para essa tarefa consistem, basicamente, em induzir um meta-modelo que mapeia as características extraídas dos dados para o desempenho preditivo dos modelos induzidos. A maioria dos algoritmos de AM supõe que os dados são gerados por uma função de distribuição desconhecida, mas estacionária. Portanto, um algoritmo que é adequado para uma amostra suficientemente grande dos dados sob análise, supostamente também o será para novos casos. Entretanto, a prática de selecionar um único algoritmo para todos os exemplos de um determinado problema não é propícia para fluxos de dados gerados em ambientes dinâmicos, pois é provável que esses dados mudem ao longo do tempo e, consequentemente, o melhor algoritmo para esses dados também pode mudar.

O objetivo geral desta tese foi o desenvolvimento de um método baseado em meta-aprendizado para a seleção automática de algoritmos de aprendizado em fluxos de dados que mudam ao longo do tempo. Essa e outras contribuições resultantes das investigações realizadas neste trabalho são apresentadas na Seção 7.1. Em seguida, na Seção 7.2, são descrito os principais resultados obtidos durante a avaliação experimental do método proposto. Na Seção 7.3, é apresentada uma lista dos artigos publicados durante o desenvolvimento desta tese. Por último, na Seção 7.4, são apresentadas as limitações do

método proposto e do planejamento experimental para avaliar as hipóteses estabelecidas nesta tese, assim como possíveis direcionamentos para trabalhos futuros.

7.1 Contribuições

A principal contribuição desta tese foi a proposta do método MetaStream para a seleção automática de algoritmos em fluxos de dados que mudam ao longo do tempo. Para o desenvolvimento desse método, as abordagens convencionais de meta-aprendizado para conjuntos de dados estacionários foram adaptadas e estendidas às especificidades de dados que são gerados continuamente e são potencialmente não estacionários. Ao invés de selecionar algoritmos para diferentes conjuntos de dados, o MetaStream seleciona o melhor algoritmo para os dados mais recentes a cada instante de tempo, considerando as características desses dados e experiências passadas. Com o desenvolvimento desse método, atingiu-se o objetivo geral desta tese de doutorado, que foi determinado na introdução desta tese:

> *Desenvolvimento de um método baseado em meta-aprendizado para gerenciar automaticamente, ao longo do tempo, o processo de sistemas de aprendizado em fluxos de dados com mudanças de conceito.*

O MetaStream foi proposto inicialmente para a seleção de algoritmos para lotes de exemplos (SelLote). Assim, o algoritmo selecionado a cada instante de tempo é utilizado no nível base para predizer o atributo alvo para vários exemplos. Entretanto, observou-se que o melhor algoritmo variava mesmo para pequenos lotes. Com base nessa observação, uma extensão do método MetaStream foi proposta, o que possibilitou sua aplicabilidade para a seleção de um algoritmo diferente para cada exemplo do nível base (SelUnit), o que abre inúmeras possibilidades para melhorias do método proposto.

Outras oportunidades existentes no meta-aprendizado para seleção de algoritmos em FCD que foram exploradas no MetaStream estão relacionadas à geração de meta-dados em problemas de fluxos de dados. Em meta-aprendizado convencional, um meta-exemplo é comumente obtido a partir das características extraídas de um conjunto de dados e do desempenho preditivo dos modelos para esse conjunto. Neste trabalho, os meta-dados também são gerados a partir da extração de características dos dados e do desempenho dos modelos, mas cada meta-exemplo é obtido dos exemplos em cada instante de tempo, e não para um conjunto de dados. Essa estratégia não tinha sido investigada em outros estudos que usam meta-aprendizado em problemas de fluxos de dados (Widmer, 1997; Harries et al., 1998; Klinkenberg, 2005; Gama e Kosina, 2011). Para facilitar a discussão sobre a caracterização dos dados e a proposta de novas medidas para fluxos de dados, realizou-se uma organização sistemática dos meta-atributos em termos dos exemplos que caracterizam (conjuntos de treinamento, horizonte de predição e seleção) e das variáveis

que caracterizam (atributos preditivos, atributo alvo e predições dos modelos). A organização obtida (Seção 4.3) não só foi útil nesta tese como poderá ser usada em outros trabalhos de meta-aprendizado para FCD e outras aplicações.

Na abordagem de seleção de algoritmos SelLote, os conjuntos de treinamento e seleção são sempre formados por mais do que um exemplo. Portanto, medidas que extraem características de conjuntos de dados podem ser aplicadas nesses casos. Porém, para a abordagem de seleção SelUnit, essas medidas podem ser aplicadas para o conjunto de treinamento, mas não de seleção, pois este é composto por um único exemplo. Isso cria a oportunidade de usar os valores dos atributos do exemplo de seleção diretamente como meta-atributos. Essa proposta é similar à do trabalho de Gama e Kosina (2011), que usaram os atributos do nível base para caracterizar o problema no nível meta, alterando apenas os valores do atributo alvo, que passam a indicar se o modelo tinha predito o exemplo corretamente ou não. No entanto, nesta tese, além dos valores dos atributos do nível base, um meta-exemplo na abordagem SelUnit é formado também pelos meta-atributos que são obtidos a partir dos dados de treinamento, idêntica à caracterização dos dados de treinamento em SelLote. Como esses dados são usados na indução do modelo, eles também são importantes para predizer o comportamento desse modelo.

A fim de manter os meta-dados sempre atualizados em relação aos dados mais recentes do nível base, os meta-exemplos também são gerados continuamente. Com isso, cria-se um fluxo de meta-dados, dos quais é possível obter informações que podem ser relevantes para a seleção de algoritmos. Essa proposta está relacionada com o trabalho de Klinkenberg (2005), que usa informações históricas do processo de aprendizado no nível meta, como o número de lotes sem mudanças de conceito e o algoritmo com maior sucesso para o último lote, para guiar a busca do meta-aprendiz. A caracterização dos meta-dados históricos é suportada pela hipótese de que os exemplos (ou meta-exemplos, nesse caso) de um fluxo de dados apresentam algum grau de dependência entre eles. Essa hipótese foi confirmada para um conjunto de dados investigado em Bifet et al. (2013), que, portanto, sugeriram que as últimas classes preditas sejam utilizadas como atributos preditivos para melhorar a eficácia do algoritmo de aprendizado para os próximos exemplos. Assim, além das características comumente usadas em meta-aprendizado convencional e daquelas obtidas a partir dos meta-dados históricos, investigou-se também características que mensuram o grau de dependência entre os exemplos, como a auto-correlação, e outras medidas empregadas em problemas de seleção de modelos para séries temporais (Prudêncio e Ludermir, 2004; Wang et al., 2009; Lemke e Gabrys, 2010).

Algumas medidas para a extração de características dos dados geram um valor de saída para cada atributo ou para a relação entre cada dois ou mais atributos. Em uma abordagem baseada em meta-aprendizado para a seleção de algoritmos para diferentes conjuntos de dados, esses valores precisam ser agregados, pois os conjuntos de dados possuem diferentes morfologias e é preciso descrevê-los no nível meta usando os mesmos meta-atributos.

Como os atributos que descrevem um problema de fluxo de dados são os mesmos ao longo do tempo ou raramente mudam, as característica obtidas de cada atributo ou da relação entre dois ou mais atributos podem ser usadas diretamente como meta-atributos, sem necessidade de agregação dessas características. Com base nisto, nesta tese foi proposta uma categorização das características em relação à morfologia dos dados. Aquelas que podem ser extraídas de conjuntos de dados com diferentes morfologias são categorizadas como independentes, enquanto que as características que só podem ser usadas como meta-atributos se os dados tiverem a mesma morfologia são categorizadas como dependentes. Para avaliar se há perda de informação relevante quando as características são agregadas, realizou-se um estudo empírico usando um conjunto de meta-dados apenas com características independentes e outro com características de ambas as categorias na indução do meta-modelo.

7.2 Principais resultados

Os resultados da avaliação empírica do método MetaStream mostraram que ele geralmente é capaz de melhorar o desempenho preditivo geral do sistema de aprendizado em relação ao método Default para problemas em que selecionar o melhor algoritmo de aprendizado não é uma tarefa simples, ou seja, quando um algoritmo não é claramente superior ao outro. Para os casos em que um algoritmo é notadamente o melhor ao longo de todo o fluxo, o método Default, geralmente, obteve resultados similares ou até superiores ao do MetaStream. Como esses resultados são provenientes de experimentos realizados em um conjunto limitado de problemas, não é possível generalizar para todos os fluxos de dados, principalmente de outros domínios. Entretanto, há fortes evidências de que o MetaStream é mais competente que o Default em selecionar o melhor algoritmo quando há diferenças de desempenho preditivo entre os modelos induzidos por esses algoritmos ao longo do tempo.

Apesar de simples, o Default é um método eficiente para problemas de classificação em FCD. Bifet et al. (2013) mostraram que a acurácia de muitos algoritmos de classificação adaptativos propostos na literatura são piores do que um classificador muito simples, denominado *No-Change*, que consiste em predizer a classe do exemplo de teste como sendo a mesma do último exemplo processado. Essa análise foi realizada apenas para um problema comumente estudado na área, mas evidencia a dificuldade de superar a acurácia de estratégias simples como o Default e o No-Change. Embora os resultados obtidos pelos autores mostrem que a acurácia do classificador *No-Change* foi superior ao Default (chamado de *Majority Class* naquele trabalho), nos estudos realizados nesta tese verificou-se o contrário. Uma explicação para essa divergência são as diferenças entre os problemas investigados e também entre o aprendizado no nível base, realizado naquele artigo, e no nível meta, realizado nesta tese.

Os resultados experimentais empregando a abordagem de seleção SelUnit mostraram que o método MetaStream foi, na maioria dos casos, capaz de relacionar as características dos dados ao desempenho dos modelos para cada exemplo do nível base. Com isso, foi possível melhorar o desempenho preditivo no nível base em comparação à abordagem de seleção de algoritmos para lotes de exemplos (SelLote). A abordagem SelUnit conseguiu uma pequena vantagem em relação à SelLote no nível base e uma ampla vantagem no nível meta, de acordo com a estatística Kappa. Essa superioridade de SelUnit no nível meta se deve principalmente ao fato de que o melhor algoritmo para um lote de exemplos geralmente não é o melhor algoritmo para todos os exemplos desse lote.

Os resultados da avaliação empírica do método MetaStream usando os conjuntos de meta-atributos MDInd e MDIndDep mostraram a importância da escolha das características que são extraídas dos dados do nível base. Com o conjunto de meta-atributos MDInd, que possui apenas características independentes da morfologia dos dados, o método MetaStream comportava-se como o Default, ou seja, sempre predizendo a classe majoritária, quando o algoritmo SVM era usado como meta-aprendiz. Esse comportamento se diferenciou do Default quando o conjunto de meta-atributos MDIndDep, que inclui características que são dependentes da morfologia dos dados, foi utilizado na indução dos meta-modelos. Ao contrário do que observou-se com o algoritmo SVM, quando o algoritmo RF foi empregado como meta-aprendiz, este mostrou-se robusto aos diferentes conjuntos de meta-atributos avaliados, principalmente porque o RF possui um mecanismo de seleção de atributos embutido em seu processo de aprendizado.

A proposta de utilizar as predições da seleção de algoritmos a partir de cada par de algoritmos para compor um ranking de algoritmos apresentou resultados promissores. Os rankings construídos a partir das predições do método MetaStream foram razoavelmente melhores, segundo o coeficiente de Spearman, do que aqueles construídos a partir das predições do Default.

Ainda que o MetaStream tenha sido capaz de melhorar o desempenho preditivo do sistema de aprendizado no nível base em relação ao Default, esse método foi superior à abordagem Ensemble apenas em alguns casos. Esses resultados, porém, não inviabilizam a utilidade do método proposto, que obteve melhores resultados do que o Ensemble quando um dos algoritmos do nível base possui desempenho preditivo aquém dos demais. Identificar sob quais circunstâncias cada método é mais promissor ou apresenta maiores dificuldades auxilia na decisão de qual método utilizar para novos problemas. Porém, como os dados e, consequentemente, o desempenho preditivo dos algoritmos mudam ao longo do tempo, mesmo que o Ensemble seja mais adequado para um determinado período de tempo analisado, não há garantias de que ele também será a melhor alternativa para dados futuros. Por outro lado, o MetaStream é constantemente atualizado e é capaz de reagir às mudanças de desempenho dos algoritmos. Ademais, se a interpretabilidade do modelo no nível base é um fator relevante, o MetaStream é mais adequado que o En-

semble, pois seleciona um único algoritmo. Ideias para a melhoria do MetaStream são discutidas na seção 7.4.

7.3 Publicações

Durante o período de doutorado, algumas das contribuições e resultados citados nas Seções 7.1 e 7.2 foram publicados em periódicos e conferências nacionais e internacionais. Os dois primeiros artigos da lista apresentada a seguir são resultados diretos desta tese e focam na seleção de algoritmos para lotes de exemplos. O primeiro recebeu o prêmio de melhor artigo da conferência e os autores foram convidados a enviar uma extensão do trabalho para um período internacional, que foi aceito e aparece como o segundo item dessa lista. Adicionalmente, um artigo que aborda a caracterização e a seleção de um algoritmo para cada exemplo já está em fase final de escrita para ser enviado para uma conferência internacional. As demais publicações dessa lista resultaram de colaborações com outros pesquisadores e grupos de pesquisa e estão relacionadas a pelo menos um dos temas de pesquisa investigados nesta tese.

1. ROSSI, A. L. D.; CARVALHO, A. C. P. L. F.; SOARES, C. Metastream: a meta-learning based method for periodic algorithm selection in time-changing data. *Neurocomputing*, v. 127, p. 52-64, 2014.

2. ROSSI, A. L. D.; CARVALHO, A. C. P. L. F.; e SOARES, C. Meta-learning for periodic algorithms selection in time-changing data. In: *Proceedings of the Brazilian Symposium on Neural Networks*, IEEE Computer Society, p. 7-12, 2012

3. PRIYA, R.; SOUZA, B. F. de; ROSSI, A. L. D.; CARVALHO, A. C. P. L. F. Predicting Execution Time of Machine Learning Tasks for Scheduling. *International Journal of Hybrid Intelligent Systems*, v. 10, p. 23-32, 2013.

4. GOMES, T. A. F.; PRUDÊNCIO, R. B. C.; SOARES, C.; ROSSI, A. L. D.; CARVALHO, A. C. P. L. F. Combining meta-learning and search techniques to select parameters for support vector machines. *Neurocomputing* (Amsterdam), v. 75, p. 3-13, 2012.

5. PRIYA, R.; de SOUZA, B. F.; ROSSI, A. L. D.; CARVALHO, A. C. P. L. F. Predicting execution time of machine learning tasks using metalearning. In: *Proceedings of the World Congress on Information and Communication Technologies*, 2012, p. 1193-1198.

6. PRIYA, R.; de SOUZA, B. F.; ROSSI, A. L. D.; CARVALHO, A. C. P. L. F. . Using Genetic Algorithms to Improve Prediction of Execution Times of ML Tasks. In: *Proceedings of the International Conference on Hybrid Artificial Intelligent Systems*, 2012, p. 196-207.

7. GOMES, T. A. F.; PRUDÊNCIO, R.B.C.; SOARES, C.; ROSSI, A.L.D.; CARVALHO, A. C. P. L. F. Combining Meta-learning and Search Techniques to SVM Parameter Selection. In: *Proceedings of the Brazilian Symposium on Neural Networks*, 2010, p. 79-84.

7.4 Limitações e trabalhos futuros

Embora os resultados empíricos reportados nos capítulos anteriores confirmem que os principais objetivos desta tese foram alcançados, o método MetaStream e o planejamento experimental obviamente possuem limitações. A seguir são descritas as principais limitações e possíveis direcionamentos para trabalhos futuros.

As estratégias de rotulação dos meta-exemplos investigadas neste trabalho consideram a comensurabilidade das diferenças dos desempenhos dos modelos apenas qualitativamente, sendo que a magnitude absoluta é ignorada. Por esse motivo, as diferenças entre os desempenhos preditivos dos métodos de seleção de algoritmos observadas no nível meta nem sempre condizem com as diferenças observadas no nível base, ou seja, o meta-aprendiz é guiado para um espaço de busca em que o erro a nível meta é minimizado, mas não há garantias de que o mesmo ocorrerá para o erro do nível base. Essa limitação das estratégias de rotulação pode prejudicar também a indução dos meta-modelos, pois meta-exemplos similares podem receber rótulos diferentes mesmo que a diferença de desempenho preditivo entre os modelos seja pequena. Uma possibilidade para tentar reduzir a divergência entre os dois níveis é empregar métodos de meta-regressão (Köpf et al., 2000), que consideram a magnitude absoluta das diferenças de desempenho entre os modelos. Assim, ao invés de predizer um rótulo que indica o melhor algoritmo, o meta-modelo prediz os desempenhos preditivos dos modelos. Entretanto, essa forma de recomendação pode sofrer interferência de possíveis predições discrepantes realizadas pelos modelos no nível base. Adicionalmente, os resultados da meta-regressão facilitariam a recomendação de algoritmos em forma de rankings (De Souto et al., 2008; Souza, 2010).

Como não existem muitos algoritmos de regressão incrementais disponíveis (Žliobaite et al., 2012), algoritmos em lote, que requerem que todos os exemplos de treinamento estejam disponíveis, foram empregados para avaliar as hipóteses estabelecidas nesta tese. Embora eles também tenham sido usados em outros estudos para alguns dos problemas investigados neste trabalho (Moreira, 2008; Harries, 1999), o seu uso está restrito a fluxos de dados com baixa taxa de geração de exemplos (Han e Kamber, 2006). Algoritmos

incrementais também podem ser empregados nos dois níveis de aprendizado no método MetaStream. Nesse caso, a seleção não seria de algoritmos e sim de modelos, pois estes sempre são atualizados quando novos dados estão disponíveis, ao contrário dos algoritmos em lote, que sempre induzem novos modelos. Entretanto, seriam necessários novos experimentos para provar a eficácia do MetaStream nesse novo cenário. Isso seria viável apenas para problemas de classificação, para os quais há um número razoável de algoritmos incrementais disponíveis.

Em todos os experimentos realizados nesta tese, os algoritmos de aprendizado foram usados com os valores padrão para os seus parâmetros, os quais podem não ser adequados para alguns casos. Entretanto, o objetivo principal consistiu em avaliar o desempenho preditivo relativo dos modelos induzidos por esses algoritmos para a posterior seleção de algoritmo, ou seja, o objetivo não foi realizar um estudo para maximizar o desempenho absoluto dos algoritmos usados no nível base, mas entender as diferenças entre eles, para que fosse possível selecionar o mais adequado em diferentes instantes de tempo para um problema sob análise. Por outro lado, o ajuste dos parâmetros dos meta-aprendizes, que também foram usados com seus valores padrão, poderia resultar em melhor desempenho preditivo do método MetaStream. Para isso, um novo conjunto de experimentos seria necessário, o que planeja-se realizar como trabalhos futuros. Além dos valores dos parâmetros, o conjunto de meta-atributos utilizados também interfere nos resultados do MetaStream, principalmente quando o meta-aprendiz SVM é empregado, como mostram os resultados apresentados na Seção 6.3. Estudar quais características foram relevantes para a seleção de algoritmos a fim de reduzir o número de meta-atributos pode melhorar a eficácia de meta-aprendizes que são sensíveis a dados de alta dimensionalidade e a meta-atributos irrelevantes. O ajuste de parâmetros e a seleção de um subconjunto de meta-atributos, porém, adicionam mais uma etapa na seleção de modelos a nível meta, o que pode tornar o processo computacionalmente custoso.

Nos experimentos de seleção de algoritmos para SelUnit, outras alternativas para a melhoria do método MetaStream puderam ser identificadas. Uma delas diz respeito à amostragem dos meta-exemplos de treinamento, que foi realizada com o objetivo de reduzir o custo computacional da indução dos meta-modelos. Para que a comparação entre as abordagens SelLote e SelUnit fosse a mais justa possível, o método de amostragem aleatória foi utilizado. Porém, outros métodos poderiam ser usados no intuito de melhorar o desempenho do MetaStream. Por exemplo, selecionar apenas os meta-exemplos que acrescentam alguma informação relevante ao meta-modelo reduziria o número de meta-exemplos redundantes (Prudêncio e Ludermir, 2007). Outro possibilidade é selecionar apenas os meta-exemplos que tenham sido rotulados com grande confiança de que a diferença de desempenho preditivo entre os modelos é consistente, potencialmente reduzindo a contradição de meta-exemplos que possuem valores de meta-atributos preditivos muito similares mas rótulos diferentes.

Nos experimentos de seleção de algoritmos para SelLote, o tamanho de cada conjunto de seleção, que contém os exemplos para os quais um algoritmo é escolhido, foi fixado a priori. Porém, para aplicações que geram dados irregularmente espaçados, pode ser interessante que esse tamanho seja dinâmico ao longo do tempo. Para os conjuntos de dados TTP e Airline, por exemplo, o número de viagens para cada dia pode variar dependendo do dia da semana e de outros fatores. Nesse caso, se o objetivo é selecionar um algoritmo para todas as viagens realizadas para cada dia, o número de exemplos no conjunto de seleção seria variável. Essa questão pode ser tratada trivialmente pelo método MetaStream, mas foi fixada a priori nos experimentos apenas por questão de simplicidade.

Os resultados apresentados no Capítulo 6 mostraram que a seleção de um único algoritmo realizada pelo método MetaStream pode ser vantajosa em relação ao uso do Ensemble para o conjunto de dados TTP, mas, geralmente, foi pior para os conjuntos de dados EDP e Airline. Tendo como base os bons resultados obtidos pelo Ensemble, uma alternativa para tentar melhorar os resultados do método MetaStream no nível base é utilizá-lo para combinar as predições dos modelos (Džeroski e Ženko, 2004; Menahem et al., 2013). Experimentos preliminares já foram realizados e os resultados foram satisfatórios, superando a abordagem Ensemble. Entretanto, é necessário comparar esses resultados com abordagens mais sofisticadas de *ensembles* para problemas de regressão (Mendes-Moreira et al., 2012).

Referências Bibliográficas

Aamodt, A.; Plaza, E. Case-based reasoning: foundational issues, methodological variations, and system approaches. *AI Communications*, v. 7, n. 1, p. 39–59, 1994. (Citado na página 67.)

Adya, M.; Collopy, F.; Armstrong, J. S.; Kennedy, M. Automatic identification of time series features for rule-based forecasting. *International Journal of Forecasting*, v. 17, n. 2, p. 143–157, 2001. (Citado nas páginas 44, 59, 61 e 62.)

Aggarwal, C. C. A framework for diagnosing changes in evolving data streams. In: *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, New York, NY, USA: ACM, 2003, p. 575–586. (Citado nas páginas 3, 25 e 27.)

Aha, D. W.; Kibler, D.; Albert, M. K. Instance-based learning algorithms. *Machine Learning*, v. 6, n. 1, p. 37–66, 1991. (Citado nas páginas 16 e 47.)

Ali, S.; Smith-Miles, K. A. A meta-learning approach to automatic kernel selection for support vector machines. *Neurocomputing*, v. 70, n. 1–3, p. 173 – 186, 2006. (Citado nas páginas 178 e 179.)

Alpaydin, E. *Introduction to machine learning.* 2nd ed. The MIT Press, 2010. (Citado na página 2.)

Amasyali, M.; Ersoy, O. *A study of meta learning for regression.* Relatório Técnico, Purdue University, `http://docs.lib.purdue.edu/ecetr/386/`, 2009. (Citado na página 59.)

Applegate, D. L.; Bixby, R. E.; Chvatal, V.; Cook, W. J. *The traveling salesman problem: A computational study (princeton series in applied mathematics).* Princeton, NJ, USA: Princeton University Press, 2007. (Citado na página 43.)

Armstrong, J.; Collopy, F. Error measures for generalizing about forecasting methods: Empirical comparisons. *International Journal of Forecasting*, v. 8, n. 1, p. 69–80, 1992. (Citado nas páginas 15 e 87.)

ASA, A. S. A. *Data expo 2009 - sections on statistical computing and statistical graphics.* `http://stat-computing.org/dataexpo/2009/`, 2009. (Citado na página 79.)

Babcock, B.; Babu, S.; Datar, M.; Motwani, R.; Widom, J. Models and issues in data stream systems. In: *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, New York, NY, USA: ACM, 2002, p. 1–16. (Citado na página 20.)

Bajwa, S.; Chung, E.; Kuwahara, M. Performance evaluation of an adaptive travel time prediction model. In: *Proceedings of 8th International IEEE Conference on Intelligent Transportation Systems*, IEEE Computer Society, 2005, p. 1000–1005. (Citado na página 77.)

Bartolini, I.; Ciaccia, P.; Ntoutsi, I.; Patella, M.; Theodoridis, Y. The PANDA framework for comparing patterns. *Data and Knowledge Engineering*, v. 68, n. 2, p. 244–260, 2009. (Citado na página 27.)

Ben-Hur, A.; Ong, C. S.; Sonnenburg, S.; Schölkopf, B.; Rätsch, G. Support vector machines and kernels for computational biology. *PLoS Computational Biology*, v. 4, n. 10, 2008. (Citado na página 97.)

Bennett, K. P.; Campbell, C. Support vector machines: hype or hallelujah? *SIGKDD Explorations Newsletter*, v. 2, n. 2, p. 1–13, 2000. (Citado na página 89.)

Bensusan, H. God doesn't always shave with occam's razor - learning when and how to prune. In: *Proceedigs of the 10th European Conference on Machine Learning*, Springer, 1998, p. 119–124. (Citado na página 44.)

Bensusan, H.; Giraud-Carrier, C. Casa batlo is in passeig de gracia or landmarking the expertise space. In: *Proceedings of the ECML'2000 workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination*, ECML'2000, 2000a, p. 29–47. (Citado na página 44.)

Bensusan, H.; Giraud-Carrier, C.; Kennedy, C. A higher-order approach to meta-learning. In: *Proceedings of the ECML - Workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination*, 2000, p. 109–118. (Citado na página 44.)

Bensusan, H.; Giraud-Carrier, C. G. Discovering task neighbourhoods through landmark learning performances. In: *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, London, UK: Springer-Verlag, 2000b, p. 325–330. (Citado nas páginas 82 e 87.)

Beringer, J.; Hüllermeier, E. Efficient instance-based learning on data streams. *Intelligent Data Analysis*, v. 11, n. 6, p. 627–650, 2007. (Citado na página 2.)

Biau, G. Analysis of a random forests model. *Journal of Machine Learning Research*, v. 13, n. 1, p. 1063–1095, 2012. (Citado na página 89.)

Bifet, A. *Adaptive learning and mining for data streams and frequent patterns*. Tese de Doutoramento, Universitat Politècnica de Catalunya, Barcelona, Espanha, 2009. (Citado nas páginas 19, 35 e 37.)

Bifet, A.; Gavaldà, R. Learning from time-changing data with adaptive windowing. In: *Proceedings of the Seventh SIAM International Conference on Data Mining*, Minneapolis, Minnesota, USA: SIAM, 2007. (Citado nas páginas 3 e 81.)

Bifet, A.; Holmes, G.; Pfahringer, B.; Kirkby, R.; Gavaldà, R. New ensemble methods for evolving data streams. In: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA: ACM, 2009, p. 139–148. (Citado nas páginas 3, 72 e 76.)

Bifet, A.; Read, J.; Žliobaite, I.; Pfahringer, B.; Holmes, G. Pitfalls in benchmarking data stream classification and how to avoid them. In: Blockeel, H.; Kersting, K.; Nijssen, S.; Železný, F., eds. *Machine Learning and Knowledge Discovery in Databases*, Springer Berlin Heidelberg, 2013, p. 465–479. (Citado nas páginas 34, 35, 59, 83, 91, 92, 149 e 150.)

Bin, Y.; Zhongzhen, Y.; Baozhen, Y. Bus arrival time prediction using support vector machines. *Journal of Intelligent Transportation Systems*, v. 10, n. 4, p. 151–158, 2006. (Citado na página 77.)

Böttcher, M.; Höppner, F.; Spiliopoulou, M. On exploiting the power of time in data mining. *ACM SIGKDD Explorations Newsletter*, v. 10, n. 2, p. 3–11, 2008. (Citado nas páginas 27 e 28.)

Brazdil, P.; Gama, J.; Henery, B. Characterizing the applicability of classification algorithms using meta-level learning. In: *Proceedings of the European conference on machine learning on Machine Learning*, Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1994, p. 83–102. (Citado na página 82.)

Brazdil, P.; Giraud-Carrier, C.; Soares, C.; Vilalta, R. *Metalearning: Applications to data mining*. Springer Verlag, 2009. (Citado nas páginas xv, 1, 2, 5, 15, 39, 40, 41, 42, 43, 45, 46, 47, 71, 87 e 144.)

Brazdil, P.; Soares, C.; Costa, J. Ranking learning algorithms: Using ibl and meta-learning on accuracy and time results. *Machine Learning*, v. 50, n. 3, p. 251–277, 2003. (Citado nas páginas 92 e 140.)

Breiman, L. Bagging predictors. *Machine Learning*, v. 24, n. 2, p. 123–140, 1996. (Citado na página 31.)

Breiman, L. Random forests. *Machine Learning*, v. 45, n. 1, p. 5–32, 2001. (Citado nas páginas 80, 89, 97 e 123.)

Breiman, L.; Friedman, J.; Olshen, R.; Stone, C. *Classification and regression trees*. Chapman & Hall (Wadsworth, Inc.), 1984. (Citado nas páginas 13, 17, 30 e 80.)

Brown, G.; Wyatt, J. L.; Tiňo, P. Managing diversity in regression ensembles. *Journal of Machine Learning Research*, v. 6, p. 1621–1650, 2005. (Citado nas páginas 182 e 183.)

Burges, C. J. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, v. 2, n. 2, p. 121–167, 1998. (Citado nas páginas 89 e 97.)

Calle, M. L.; Urrea, V. Letter to the editor: Stability of random forest importance measures. *Briefings in Bioinformatics*, v. 12, n. 1, p. 86–89, 2010. (Citado na página 134.)

Campello, R.; Hruschka, E. On comparing two sequences of numbers and its applications to clustering analysis. *Information Sciences*, v. 179, n. 8, p. 1025–1039, 2009. (Citado na página 140.)

Caruana, R.; Niculescu-Mizil, A. An empirical comparison of supervised learning algorithms. In: *Proceedings of the 23rd international conference on Machine learning*, New York, NY, USA: ACM, 2006, p. 161–168. (Citado nas páginas 1, 45 e 89.)

Castiello, C.; Castellano, G.; Fanelli, A. Meta-data: Characterization of input features for meta-learning. In: Torra, V.; Narukawa, Y.; Miyamoto, S., eds. *Modeling Decisions for Artificial Intelligence*, Springer Berlin / Heidelberg, 2005, p. 295–304. (Citado na página 178.)

Catlett, J. *Megainduction: Machine learning on very large databases*. Tese de Doutoramento, Department of Computer Science, University of Sydney, Sydney, Australia, 1991. (Citado na página 28.)

Chapelle, O.; Vapnik, V.; Bousquet, O.; Mukherjee, S. Choosing multiple parameters for support vector machines. *Machine Learning*, v. 46, n. 1-3, p. 131–159, 2002. (Citado na página 97.)

Chen, L.; Zhang, S.; Tu, L. An algorithm for mining frequent items on data stream using fading factor. In: *Annual International Computer Software and Applications*

Conference, Los Alamitos, CA, USA: IEEE Computer Society, 2009, p. 172–177. (Citado nas páginas 22 e 25.)

Cherkassky, V. S.; Mulier, F. *Learning from data: Concepts, theory, and methods.* 1st ed. New York, NY, USA: John Wiley & Sons, Inc., 1998. (Citado na página 14.)

Cohen, J. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, v. 20, n. 1, p. 37–46, 1960. (Citado na página 91.)

Cohen, W. W. Fast effective rule induction. In: *Proceedings of the 12th International Conference on Machine Learning*, 1995, p. 115–123. (Citado na página 32.)

Corder, G. W.; Foreman, D. I. *Nonparametric statistics for non-statisticians: A step-by-step approach.* New Jersey: Wiley, 2009. (Citado na página 27.)

Cristianini, N.; Shawe-Taylor, J. *An introduction to support vector machines: and other kernel-based learning methods.* New York, NY, USA: Cambridge University Press, 2000. (Citado nas páginas 25, 80 e 89.)

Daelemans, W.; Hoste, V.; Meulder, F.; Naudts, B. Combined optimization of feature selection and algorithm parameters in machine learning of language. In: Lavrač, N.; Gamberger, D.; Blockeel, H.; Todorovski, L., eds. *Proceedings of the 14th European Conference on Machine Learning*, Springer Berlin Heidelberg, 2003, p. 84–95. (Citado na página 98.)

Dasu, T.; Krishnan, S.; Lin, D.; Venkatasubramanian, S.; Yi, K. Change (detection) you can believe in: Finding distributional shifts in data streams. In: *Proceedings of the 8th International Symposium on Intelligent Data Analysis: Advances in Intelligent Data Analysis VIII*, Berlin, Heidelberg: Springer-Verlag, 2009, p. 21–34. (Citado nas páginas 63 e 181.)

Dawid, A. P.; Vovk, V. G. Prequential probability: principles and properties. *Bernoulli*, v. 5, n. 1, p. 125–162, 1999. (Citado na página 34.)

De Souto, M. C. P.; Prudencio, R.; Soares, R.; Araujo, D.; Costa, I.; Ludermir, T.; Schliep, A. Ranking and selecting clustering algorithms using a meta-learning approach. In: *IEEE International Joint Conference on Neural Networks*, 2008, p. 3729–3735. (Citado nas páginas 89, 92, 138, 140 e 153.)

Demšar, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, v. 7, p. 1–30, 2006. (Citado nas páginas 91, 92, 93 e 113.)

Díaz-Uriarte, R.; Andrés, S. Gene selection and classification of microarray data using random forest. *BMC Bioinformatics*, v. 7, n. 1, p. 1–13, 2006. Disponível em: `http://dx.doi.org/10.1186/1471-2105-7-3` (Citado na página 122.)

DIETTERICH, T. G. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Compututing*, v. 10, n. 7, p. 1895–1923, 1998. (Citado nas páginas 34 e 36.)

DIETTERICH, T. G. Ensemble methods in machine learning. In: *Proceedings of the First International Workshop on Multiple Classifier Systems*, London, UK: Springer-Verlag, 2000, p. 1–15. (Citado nas páginas 3 e 31.)

DOMINGOS, P.; HULTEN, G. Mining high-speed data streams. In: *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA: ACM, 2000, p. 71–80. (Citado nas páginas 25, 28, 29, 72 e 89.)

DUCH, W.; MASZCZYK, T.; GROCHOWSKI, M. Optimal support features for meta-learning. In: JANKOWSKI, N.; DUCH, W.; GRČABCZEWSKI, K., eds. *Meta-Learning in Computational Intelligence*, v. 358 de *Studies in Computational Intelligence*, Springer Berlin / Heidelberg, p. 317–358, 2011. (Citado na página 178.)

DŽEROSKI, S.; ŽENKO, B. Is combining classifiers with stacking better than selecting the best one? *Machine Learning*, v. 54, p. 255–273, 2004. (Citado na página 155.)

FAN, W. Systematic data selection to mine concept-drifting data streams. In: *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA: ACM, 2004, p. 128–137. (Citado na página 33.)

FAYYAD, U. M.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From data mining to knowledge discovery in databases. *AI Magazine*, v. 17, n. 3, p. 37–54, 1996. (Citado na página 12.)

FERRI, C.; HERNÁNDEZ-ORALLO, J.; MODROIU, R. An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, v. 30, n. 1, p. 27–38, 2009. (Citado na página 17.)

FIDALGO-MERINO, R.; NUNEZ, M. Self-adaptive induction of regression trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 33, n. 8, p. 1659–1672, 2011. (Citado nas páginas 76, 78, 79, 80 e 81.)

FLACH, P. *Machine learning: The art and science of algorithms that make sense of data*. New York, NY, USA: Cambridge University Press, 2012. (Citado nas páginas 1 e 12.)

FREUND, Y.; SCHAPIRE, R. E. Experiments with a new boosting algorithm. In: *Proceedings of the Thirteenth International Conference on Machine Learning*, 1996, p. 148–156. (Citado na página 31.)

FRIEDMAN, J. H. Multivariate adaptive regression splines. *The annals of statistics*, v. 19, n. 1, p. 1–67, 1991. (Citado nas páginas 76 e 80.)

FRIEDMAN, J. H.; STUETZLE, W. Projection pursuit regression. *Journal of the American Statistical Association*, v. 76, n. 376, p. 817–823, 1981. (Citado na página 80.)

GABER, M. M.; ZASLAVSKY, A.; KRISHNASWAMY, S. Mining data streams: a review. *SIGMOD Record*, v. 34, n. 2, p. 18–26, 2005. (Citado na página 19.)

GAMA, J. *Knowledge discovery from data streams.* CRC Press, 2010. (Citado nas páginas 2, 6, 11, 19, 23, 25, 59 e 89.)

GAMA, J.; KOSINA, P. Tracking recurring concepts with meta-learners. In: LOPES, L. S.; LAU, N.; MARIANO, P.; ROCHA, L. M., eds. *EPIA*, Springer, 2009, p. 423–434 (*Lecture Notes in Computer Science*, v.5816). (Citado nas páginas 6 e 49.)

GAMA, J.; KOSINA, P. Learning about the learning process. In: *Proceedings of the 10th International Conference on Advances in Intelligent Data Analysis*, Berlin, Heidelberg: Springer-Verlag, 2011, p. 162–172. (Citado nas páginas 3, 6, 49, 70, 148, 149, 178 e 183.)

GAMA, J.; MEDAS, P.; CASTILLO, G.; RODRIGUES, P. P. Learning with drift detection. In: BAZZAN, A. L. C.; LABIDI, S., eds. *Proceedings of the 17th Brazilian Symposium on Artificial Intelligence*, São Luis, Maranhão: Springer, 2004, p. 286–295. (Citado nas páginas 5, 24, 25, 26, 78 e 81.)

GAMA, J.; RODRIGUES, P. P. Data stream processing. In: GAMA, J.; GABER, M. M., eds. *Learning from Data Streams: Processing Techniques in Sensor Networks*, cap. 3, Springer, p. 25–38, 2007. (Citado na página 25.)

GAMA, J.; RODRIGUES, P. P. An overview on mining data streams. In: ABRAHAM, A.; HASSANIEN, A. E.; CARVALHO, A. C. P.; SNÁSEL, V., eds. *Foundations of Computational (6)*, v. 206 de *Studies in Computational Intelligence*, Springer, p. 29–45, 2009. (Citado nas páginas 11, 19 e 36.)

GAMA, J.; SEBASTIÃO, R.; RODRIGUES, P. P. Issues in evaluation of stream learning algorithms. In: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA: ACM, 2009, p. 329–338. (Citado nas páginas 26, 34, 35 e 36.)

GAMA, J.; SEBASTIÃO, R.; RODRIGUES, P. P. On evaluating stream learning algorithms. *Machine Learning*, v. 90, n. 3, p. 317–346, 2013. (Citado nas páginas 22, 34, 35, 91 e 104.)

Gama, J.; Žliobaite, I.; Bifet, A.; Pechenizkiy, M.; Bouchachia, A. A survey on concept drift adaptation. *ACM Computing Surveys*, v. 46, n. 4, in press, 2014. (Citado nas páginas 2, 19, 23, 25 e 28.)

Ganti, V.; Gehrke, J.; Ramakrishnan, R. A framework for measuring changes in data characteristics. In: *PODS '99: Proceedings of the eighteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, New York, NY, USA: ACM, 1999, p. 126–137. (Citado na página 27.)

Genuer, R.; Poggi, J.-M.; Tuleau-Malot, C. Variable selection using random forests. *Pattern Recognition Letters*, v. 31, n. 14, p. 2225–2236, 2010. (Citado na página 122.)

Ghosh, B.; Sen, P., eds. *Handbook of sequential analysis.* Marcel Dekker, 1991. (Citado na página 34.)

Giraud-Carrier, C.; Vilalta, R.; Brazdil, P. Introduction to the special issue on meta-learning. *Machine Learning*, v. 54, n. 3, p. 187–193, 2004. (Citado nas páginas 2, 39 e 40.)

Gomes, J. B.; Menasalvas, E.; Sousa, P. A. C. Learning recurring concepts from data streams with a context-aware ensemble. In: *Proceedings of the ACM Symposium on Applied Computing*, New York, NY, USA: ACM, 2011, p. 994–999. (Citado nas páginas 26, 72 e 76.)

Gomes, T. A.; Prudêncio, R. B.; Soares, C.; Rossi, A. L.; Carvalho, A. Combining meta-learning and search techniques to select parameters for support vector machines. *Neurocomputing*, v. 75, n. 1, p. 3–13, 2012. (Citado na página 66.)

Gordon, D. F.; Desjardins, M. Evaluation and selection of biases in machine learning. *Machine Learning*, v. 20, n. 1-2, p. 5–22, 1995. (Citado na página 1.)

Guyon, I.; Elisseeff, A. An introduction to variable and feature selection. *Journal of Machine Learning Research*, v. 3, p. 1157–1182, 2003. (Citado nas páginas 187 e 188.)

Hagerup, T.; Rüb, C. A guided tour of chernoff bounds. *Information Processing Letters*, v. 33, n. 6, p. 305–308, 1990. (Citado na página 30.)

Hall, M. A. Correlation-based feature selection for discrete and numeric class machine learning. In: *Proceedings of the International Conference on Machine Learning*, Morgan Kaufmann, 2000, p. 359–366. (Citado nas páginas 99 e 188.)

Han, J.; Kamber, M. *Data mining: Concepts and techniques.* Second ed. San Francisco, CA: Morgan Kaufmann Publishers, 2006. (Citado nas páginas 11, 19, 20, 21, 39 e 153.)

Harries, M. *Splice-2 comparative evaluation: Electricity pricing.* Relatório Técnico 9905, School of Computer Science and Engineering, University of New South Wales, 1999. (Citado nas páginas 78 e 153.)

Harries, M.; Sammut, C.; Horn, K. Extracting hidden context. *Machine Learning*, v. 32, n. 2, p. 101–126, 1998. (Citado nas páginas 6, 23, 69 e 148.)

Hastie, T.; Tibshirani, R.; Friedman, J. *The elements of statistical learning: data mining, inference and prediction.* 2 ed. Springer, 2009. (Citado na página 13.)

Haykin, S. *Neural networks: A comprehensive foundation.* Prentice Hall, 1999. (Citado na página 16.)

Helmbold, D. P.; Long, P. M. Tracking drifting concepts by minimizing disagreements. *Machine Learning*, v. 14, n. 1, p. 27–45, 1994. (Citado na página 24.)

Hernández-Orallo, J.; Flach, P.; Ferri, C. A unified view of performance metrics: translating threshold choice into expected classification loss. *Journal of Machine Learning Research*, v. 13, n. 1, p. 2813–2869, 2012. (Citado na página 91.)

Hoeffding, W. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, v. 58, n. 301, p. 13–30, 1963. (Citado na página 29.)

Hulten, G.; Spencer, L.; Domingos, P. Mining time-changing data streams. In: *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA: ACM, 2001, p. 97–106. (Citado nas páginas 29, 30 e 76.)

Ikonomovska, E.; Gama, J. Learning model trees from data streams. In: *Proceedings of the International Conference on Discovery Science*, 2008, p. 52–63. (Citado na página 30.)

Ikonomovska, E.; Gama, J.; Džeroski, S. Learning model trees from evolving data streams. *Data Mining and Knowledge Discovery*, v. 23, p. 128–168, 2011. (Citado nas páginas 30, 76, 79 e 80.)

Ikonomovska, E.; Gama, J.; Sebastião, R.; Gjorgjevik, D. Regression trees from data streams with drift detection. In: Gama, J.; Costa, V. S.; Jorge, A. M.; Brazdil, P., eds. *Discovery Science*, Springer, 2009, p. 121–135 (*Lecture Notes in Computer Science*, v.5808). (Citado na página 30.)

JAPKOWICZ, N.; SHAH, M. *Evaluating learning algorithms: A classification perspective*. New York, NY, USA: Cambridge University Press, 2011. (Citado nas páginas 17 e 91.)

JOACHIMS, T. Text categorization with suport vector machines: Learning with many relevant features. In: *Proceedings of the 10th European Conference on Machine Learning*, London, UK, UK: Springer-Verlag, 1998, p. 137–142. (Citado na página 97.)

KALOUSIS, A. *Algorithm selection via meta-learning*. Tese de Doutoramento, University of Geneva, Faculty of Sciences, Geneva, Switzerland, 2002. (Citado nas páginas 4, 7, 40, 43, 45, 46, 58, 59, 61, 64, 66, 71, 82, 139, 145, 181 e 183.)

KALOUSIS, A.; GAMA, J.; HILARIO, M. On data and algorithms: Understanding inductive performance. *Machine Learning*, v. 54, n. 3, p. 275–312, 2004. (Citado na página 5.)

KALOUSIS, A.; HILARIO, M. Feature selection for meta-learning. In: CHEUNG, D.; WILLIAMS, G.; LI, Q., eds. *Advances in Knowledge Discovery and Data Mining*, Springer Berlin Heidelberg, 2001, p. 222–233 (*Lecture Notes in Computer Science*, v.2035). (Citado nas páginas 99, 187 e 188.)

KALOUSIS, A.; HILARIO, M. Representational issues in meta-learning. In: FAWCETT, T.; MISHRA, N., eds. *Proceedings of the Twentieth International Conference on Machine Learning*, AAAI Press, 2003, p. 313–320. (Citado nas páginas 67 e 83.)

KALOUSIS, A.; PRADOS, J.; HILARIO, M. Stability of feature selection algorithms: a study on high-dimensional spaces. *Knowledge and Information Systems*, v. 12, n. 1, p. 95–116, 2007. (Citado na página 99.)

KALOUSIS, A.; THEOHARIS, T. Noemon: Design, implementation and performance results for an intelligent assistant for classifier selection. *Intelligent Data Analysis*, v. 3, n. 5, p. 319–337, 1999. (Citado nas páginas 48 e 66.)

KANDA, J.; CARVALHO, A.; HRUSCHKA, E.; SOARES, C. Using meta-learning to classify traveling salesman problems. In: *Proceedings of the 2010 Eleventh Brazilian Symposium on Neural Networks*, Washington, DC, USA: IEEE Computer Society, 2010, p. 73–78. (Citado na página 43.)

KANDA, J. Y. *Uso de meta-aprendizado na recomendação de meta-heurísticas para o problema do caixeiro viajante*. Tese de Doutoramento, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, São Paulo, Brasil, 2012. Disponível em: `http://www.teses.usp.br` (Citado nas páginas 92, 99, 138, 140 e 187.)

KIFER, D.; BEN-DAVID, S.; GEHRKE, J. Detecting change in data streams. In: *Proceedings of the Thirtieth International Conference on Very Large Data Bases*, VLDB Endowment, 2004, p. 180–191. (Citado nas páginas xxiii, 3, 26, 27, 63 e 181.)

KISGYORGY, L.; RILETT, L. Travel time prediction by advanced neural network. *Periodica Polytechnica Civil Engineering*, v. 46, n. 1, p. 15–32, 2002. (Citado na página 77.)

KLINKENBERG, R. Learning drifting concepts: Example selection vs. example weighting. *Intellingent Data Analysis*, v. 8, n. 3, p. 281–300, 2004. (Citado nas páginas 3, 5 e 22.)

KLINKENBERG, R. Meta-learning, model selection, and example selection in machine learning domains with concept drift. In: BAUER, M.; BRANDHERM, B.; FÜRNKRANZ, J.; GRIESER, G.; HOTHO, A.; JEDLITSCHKA, A.; KRÖNER, A., eds. *LWA*, DFKI, 2005, p. 164–171. (Citado nas páginas 3, 5, 6, 22, 23, 24, 48, 49, 55, 68, 69, 72, 148 e 149.)

KLINKENBERG, R.; JOACHIMS, T. Detecting concept drift with support vector machines. In: *Proceedings of the Seventeenth International Conference on Machine Learning*, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, p. 487–494. (Citado nas páginas 25 e 49.)

KLINKENBERG, R.; RENZ, I. Adaptive information filtering: Learning in the presence of concept drifts. In: *Workshop Notes of the ICML/AAAI-98 Workshop on Learning for Text Categorization*, AAAI Press, 1998, p. 33–40. (Citado nas páginas 22, 24 e 25.)

KOHAVI, R.; JOHN, G. H. Wrappers for feature subset selection. *Artificial Intelligence*, v. 97, n. 1-2, p. 273–324, 1997. (Citado na página 187.)

KOLTER, J. Z.; MALOOF, M. A. Dynamic weighted majority: An ensemble method for drifting concepts. *Journal of Machine Learning Research*, v. 8, p. 2755–2790, 2007. (Citado nas páginas 3, 33 e 78.)

KONONENKO, I. Estimating attributes: analysis and extensions of relief. In: *Proceedings of the European conference on machine learning*, ECML-94, Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1994, p. 171–182 (*ECML-94*,). (Citado nas páginas 99 e 188.)

KÖPF, C.; TAYLOR, C.; KELLER, J. Meta-analysis: From data characterisation for meta-learning to meta-regression. In: BRAZDIL, P.; JORGE, A., eds. *Proceedings of the PKDD Workshop on Data Mining, Decision Support, Meta-Learning and ILP: Forum for Practical Problem Presentation and Prospective Solutions Time*, 2000. (Citado nas páginas 46, 82, 138 e 153.)

KOYCHEV, I. Gradual forgetting for adaptation to concept drift. In: *Proceedings of ECAI 2000 workshop on current issues in spatio-temporal reasoning*, ECAI Press, 2000, p. 101–106. (Citado nas páginas 3 e 22.)

KUBA, P.; BRAZDIL, P.; SOARES, C.; WOZNICA, A. Exploiting sampling and meta-learning for parameter setting support vector machines. In: F. GARIJO, J. RIQUELME, M. T., ed. *Proceedings of the Workshop de Minería de Datos Y Aprendizaje of IBERAMIA*, Universidad de Sevilla, 2002, p. 217–225. (Citado na página 59.)

KUNCHEVA, L. I. *Combining pattern classifiers: Methods and algorithms*. Wiley-Interscience, 2004. (Citado na página 31.)

LASKOV, P.; GEHL, C.; KRÜGER, S.; MÜLLER, K.-R. Incremental support vector learning: Analysis, implementation and applications. *Journal of Machine Learning Research*, v. 7, p. 1909–1936, 2006. (Citado na página 28.)

LAW, Y.-N.; ZANIOLO, C. An adaptive nearest neighbor classification algorithm for data streams. In: JORGE, A.; TORGO, L.; BRAZDIL, P.; CAMACHO, R.; GAMA, J., eds. *PKDD*, Springer, 2005, p. 108–120 (*Lecture Notes in Computer Science*, v.3721). (Citado na página 30.)

LEMKE, C.; GABRYS, B. Meta-learning for time series forecasting and forecast combination. *Neurocomputing*, v. 73, n. 10–12, p. 2006–2016, 2010. (Citado nas páginas 4, 44, 48, 59, 61, 62, 66, 87, 144, 149, 180, 182 e 183.)

LIAW, A.; WIENER, M. Classification and regression by randomforest. *R News*, v. 2, n. 3, p. 18–22, 2002.
Disponível em: `http://CRAN.R-project.org/doc/Rnews/` (Citado nas páginas 81 e 122.)

LINDNER, G.; STUDER, R. Ast: Support for algorithm selection with a cbr approach. In: *PKDD '99: Proceedings of the Third European Conference on Principles of Data Mining and Knowledge Discovery*, London, UK: Springer-Verlag, 1999, p. 418–423. (Citado na página 43.)

LITTLESTONE, N. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, v. 2, n. 4, p. 285–318, 1988. (Citado na página 25.)

LIU, H.; MOTODA, H.; SETIONO, R.; ZHAO, Z. Feature selection: An ever evolving frontier in data mining. In: *JMLR Workshop and Conference Proceedings, The Fourth Workshop on Feature Selection in Data Mining*, JMLR.org, 2010, p. 4–13. (Citado nas páginas 67, 187 e 188.)

Liu, J.; Li, X.; Zhong, W. Ambiguous decision trees for mining concept-drifting data streams. *Pattern Recognition Letters*, v. 30, n. 15, p. 1347–1355, 2009. (Citado na página 30.)

Madjarov, G.; Kocev, D.; Gjorgjevikj, D.; Džeroski, S. An extensive experimental comparison of methods for multi-label learning. *Pattern Recognition*, v. 45, n. 9, p. 3084 – 3104, 2012. (Citado na página 88.)

Makridakis, S.; Hibon, M. The m3-competition: results, conclusions and implications. *International Journal of Forecasting*, v. 16, n. 4, p. 451–476, 2000. (Citado na página 48.)

Maloof, M. A.; Michalski, R. S. Selecting examples for partial memory learning. *Machine Learning*, v. 41, n. 1, p. 27–52, 2000. (Citado na página 22.)

Maloof, M. A.; Michalski, R. S. Incremental learning with partial instance memory. *Artificial Intelligence*, v. 154, n. 1-2, p. 95–126, 2004. (Citado na página 28.)

Menahem, E.; Rokach, L.; Elovici, Y. Combining one-class classifiers via meta learning. In: *Proceedings of the 22nd ACM international conference on Conference on information knowledge management*, New York, NY, USA: ACM, 2013, p. 2435–2440. (Citado na página 155.)

Mendes-Moreira, J.; Soares, C.; Jorge, A. M.; Sousa, J. F. D. Ensemble approaches for regression: A survey. *ACM Computing Surveys*, v. 45, n. 1, p. 10–40, 2012. (Citado nas páginas 14, 31, 92 e 155.)

Meyer, D.; Dimitriadou, E.; Hornik, K.; Weingessel, A.; Leisch, F. *e1071: Misc functions of the department of statistics (e1071), tu wien.* R package version 1.6-1, 2012.
Disponível em: `http://CRAN.R-project.org/package=e1071` (Citado na página 81.)

Michie, D.; Spiegelhalter, D.; Taylor, C. Introduction. In: Michie, D.; Spiegelhalter, D.; Taylor, C., eds. *Machine Learning, Neural and Statistical Classification*, Ellis Horwood, 1994. (Citado nas páginas 42, 61, 179, 181 e 182.)

Milborrow, S. *earth: Multivariate adaptive regression spline models.* Derived from mda:mars by Trevor Hastie and Rob Tibshirani. R package version 3.2-3, 2012.
Disponível em: `http://CRAN.R-project.org/package=earth` (Citado na página 81.)

Mitchell, T. M. *Machine learning.* New York: McGraw Hill, 1997. (Citado nas páginas 1, 12, 39 e 40.)

Monard, M. C.; Baranauskas, J. A. Conceitos sobre aprendizado de máquina. In: Rezende, S. O., ed. *Sistemas Inteligentes: Fundamentos e Aplicações*, cap. 4, Editora Manole Ltda, p. 89–114, 2003. (Citado na página 17.)

Moreira, J. P. C. L. M. *Travel time prediction for the planning of mass transit companies: a machine learning approach.* Tese de Doutoramento, Faculty of Engineering of University of Porto, 2008.
Disponível em: `http://www.liaad.up.pt/pub/2008/Mor08` (Citado nas páginas 3, 6, 14, 72, 76, 77, 80, 81, 99 e 153.)

Murphy, K. P. *Machine learning: A probabilistic perspective.* Adaptive Computation and Machine Learning series. The MIT Press, 2012. (Citado nas páginas 32 e 147.)

Musliu, N.; Schwengerer, M. Algorithm selection for the graph coloring problem. In: Nicosia, G.; Pardalos, P., eds. *Proceedings of the Learning and Intelligent Optimization Conference*, Springer Berlin Heidelberg, 2013, p. 389–403. (Citado na página 89.)

Nascimento, A. C. A.; Prudêncio, R. B.; Souto, M. C. P.; Costa, I. G. Mining rules for the automatic selection process of clustering methods applied to cancer gene expression data. In: Alippi, C.; Polycarpou, M.; Panayiotou, C.; Ellinas, G., eds. *Artificial Neural Networks - ICANN 2009*, Springer Berlin Heidelberg, 2009, p. 20–29 (*Lecture Notes in Computer Science*, v.5769). (Citado nas páginas 87 e 89.)

Neave, H.; Worthington, P. *Distribution-free tests.* Londres: Routledge, 448 p., 1992. (Citado nas páginas 92 e 140.)

Nisbet, R.; Elder, J.; Miner, G. *Handbook of statistical analysis & data mining applications.* Academic Press, 2009. (Citado na página 13.)

Pacific Marine Environmental Laboratory *Tropical atmosphere ocean project (TAO).* `http://www.pmel.noaa.gov/tao/`, 2010. (Citado na página 27.)

Page, E. S. Continuous inspection schemes. *Biometrika*, v. 41, n. 1/2, p. 100–115, 1954. (Citado na página 25.)

Pearson, R. K. *Exploring data in engineering, the sciences, and medicine.* Oxford University Press, 2011. (Citado na página 180.)

Peng, Y.; Flach, P. A.; Soares, C.; Brazdil, P. Improved dataset characterisation for meta-learning. In: *DS '02: Proceedings of the 5th International Conference on Discovery Science*, London, UK: Springer-Verlag, 2002, p. 141–152. (Citado na página 44.)

Pfahringer, B.; Bensusan, H.; Giraud-Carrier, C. G. Meta-learning by landmarking various learning algorithms. In: *Proceedings of the Seventeenth International Conference on Machine Learning*, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, p. 743–750. (Citado nas páginas 1, 44 e 71.)

Potts, D.; Sammut, C. Incremental learning of linear model trees. *Machine Learning*, v. 61, p. 5–48, 2005. (Citado na página 80.)

Prati, R. C.; Batista, G. E.; Monard, M. C. Class imbalances versus class overlapping: An analysis of a learning system behavior. In: Monroy, R.; Arroyo-Figueroa, G.; Sucar, L.; Sossa, H., eds. *MICAI 2004: Advances in Artificial Intelligence*, Springer Berlin Heidelberg, 2004, p. 312–321. (Citado na página 109.)

Pratt, L.; Thrun, S. Second special issue on inductive transfer. *Machine Learning*, v. 28, n. 1, 1997. (Citado na página 39.)

Prodromidis, A. L.; Chan, P.; Stolfo, S. J. Meta-learning in distributed data mining systems: Issues and approaches. In: Kargupta, H.; Chan, P., eds. *Advances in Distributed and Parallel Knowledge Discovery*, cap. 3, AAAI Press, p. 81–114, 2000. (Citado na página 48.)

Prudêncio, R.; Guerra, S.; Ludermir, T. Using support vector machines to predict the performance of mlp neural networks. In: *Proceedings of the 10th Brazilian Symposium on Neural Networks*, 2008, p. 201–206. (Citado na página 46.)

Prudêncio, R. B. C.; Ludermir, T. B. Meta-learning approaches to selecting time series models. *Neurocomputing*, v. 61, p. 121–137, 2004. (Citado nas páginas 5, 44, 48, 59, 61, 62, 87, 149 e 180.)

Prudêncio, R. B. C.; Ludermir, T. B. Active selection of training examples for meta-learning. In: *Proceedings of the 7th International Conference on Hybrid Intelligent Systems*, Los Alamitos, CA, USA: IEEE Computer Society, 2007, p. 126–131. (Citado na página 154.)

Quinlan, J. R. The effect of noise on concept learning. In: Michalski, R. S. I.; Carboneel, J. G.; Mitchell, eds. *Machine Learning: An Artificial Intelligence Approach*, Morgan Kaufmann Publishers, p. 149–166, 1986. (Citado na página 109.)

Quinlan, J. R. *C4.5: Programs for machine learning.* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993. (Citado nas páginas 29 e 188.)

R Core Team *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria, ISBN 3-900051-07-0, 2012. Disponível em: `http://www.R-project.org/` (Citado na página 81.)

RAEDT, L. D. *Logical and relational learning.* Cognitive Technologies. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2008. (Citado nas páginas 7, 58, 66 e 67.)

RAJARAMAN, A.; ULLMAN, J. D. *Mining of massive datasets.* New York, NY, USA: Cambridge University Press, 2011. (Citado na página 72.)

RENDELL, L. A.; SHESHU, R.; TCHENG, D. K. Layered concept-learning and dynamically variable bias management. In: *Proceedings of the International Joint Conference on Artificial Intelligence*, Morgan Kaufmann, 1987, p. 308–314. (Citado nas páginas 2 e 40.)

RICE, J. R. The algorithm selection problem. *Advances in Computers*, v. 15, p. 65–118, 1976. (Citado na página 40.)

ROBNIK-ŠIKONJA, M.; KONONENKO, I. Theoretical and empirical analysis of ReliefF and RReliefF. *Machine Learning*, v. 53, n. 1-2, p. 23–69, 2003. (Citado nas páginas 99, 187, 188 e 189.)

ROSSI, A. L. D.; CARVALHO, A. C. P. L. F.; SOARES, C. Metastream: a meta-learning based method for periodic algorithm selection in time-changing data. 2014. (Citado na página 88.)

SAEYS, Y.; INZA, I.; LARRAÑAGA, P. A review of feature selection techniques in bioinformatics. *Bioinformatics*, v. 23, n. 19, p. 2507–2517, 2007. (Citado na página 188.)

SANDRI, M.; ZUCCOLOTTO, P. Variable selection using random forests. In: ZANI, S.; CERIOLI, A.; RIANI, M.; VICHI, M., eds. *Data Analysis, Classification and the Forward Search*, Studies in Classification, Data Analysis, and Knowledge Organization, Springer Berlin Heidelberg, p. 263–270, 2006. (Citado na página 122.)

SCHLIMMER, J. C.; GRANGER, R. H. Beyond incremental processing: Tracking concept drift. In: *Proceedings of the AAAI National Conf. on Artifical Intelligence*, 1986, p. 502–507. (Citado na página 76.)

SCOTT, W. Reliability of content analysis: The case of nominal scale coding. *Public Opinion Q*, v. 19, p. 321–325, 1955. (Citado na página 91.)

SEBASTIÃO, R.; RODRIGUES, P.; GAMA, J. Change detection in climate data over the iberian peninsula. In: *IEEE International Conference on Data Mining Workshops*, 2009, p. 248–253. (Citado nas páginas 63 e 181.)

SHAKER, A.; HÜLLERMEIER, E. Iblstreams: a system for instance-based classification and regression on data streams. *Evolving Systems*, v. 3, n. 4, p. 235–249, 2012. (Citado na página 80.)

SIEGEL, S.; JR., N. J. C. *Nonparametric statistics for the behavioral sciences*. Second ed. McGraw-Hill Book Company, 1988. (Citado na página 36.)

SMITH-MILES, K. A. Cross-disciplinary perspectives on meta-learning for algorithm selection. *ACM Computing Surveys*, v. 41, n. 1, p. 1–25, 2008. (Citado nas páginas 40 e 41.)

SOARES, C. *Learning rankings of learning algorithms: Recomendation of algorithms with meta-learning*. Tese de Doutoramento, Faculdade de Ciências da Universidade do Porto, Porto, Portugal, 2004. (Citado nas páginas 43, 59, 61, 64, 92, 138, 140, 178, 179, 181 e 184.)

SOARES, C.; BRAZDIL, P. Zoomed ranking: Selection of classification algorithms based on relevant performance information. In: *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, London, UK, UK: Springer-Verlag, 2000, p. 126–135. (Citado nas páginas 45, 47, 82 e 138.)

SOARES, C.; BRAZDIL, P. B.; KUBA, P. A meta-learning method to select the kernel width in support vector regression. *Machine Learning*, v. 54, p. 195–209, 10.1023/B:MACH.0000015879.28004.9b, 2004. (Citado nas páginas 66 e 83.)

SOHN, S. Y. Meta analysis of classification algorithms for pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 21, n. 11, p. 1137–1144, 1999. (Citado nas páginas 43 e 138.)

SOUZA, B.; CARVALHO, A.; SOARES, C. Metalearning for gene expression data classification. In: *Proceedings of the 8th International Conference on Hybrid Intelligent Systems*, 2008, p. 441–446. (Citado na página 66.)

SOUZA, B. F. D. *Meta-aprendizagem aplicada à classificação de dados de expressão gênica*. Tese de Doutoramento, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, São Paulo, Brasil, disponível em: `http://www.teses.usp.br/teses/disponiveis/55/55134/tde-04012011-142551`, 2010. (Citado nas páginas 89, 92, 138, 140 e 153.)

STATNIKOV, A.; WANG, L.; ALIFERIS, C. A comprehensive comparison of random forests and support vector machines for microarray-based cancer classification. *BMC Bioinformatics*, v. 9, n. 1, p. 1–10, 2008. (Citado nas páginas 1 e 97.)

STEINWART, I.; CHRISTMANN, A. *Support vector machines*. 1st ed. Springer Publishing Company, Incorporated, 2008. (Citado na página 89.)

STREET, W. N.; KIM, Y. A streaming ensemble algorithm (sea) for large-scale classification. In: *Proceedings of the seventh ACM SIGKDD international conference on*

Knowledge discovery and data mining, New York, NY, USA: ACM, 2001, p. 377–382. (Citado na página 76.)

Strobl, C.; Boulesteix, A.-L.; Kneib, T.; Augustin, T.; Zeileis, A. Conditional variable importance for random forests. *BMC Bioinformatics*, v. 9, n. 1, p. 307, 2008. (Citado nas páginas 122 e 123.)

Tan, P.-N.; Steinbach, M.; Kumar, V. *Introduction to data mining.* Boston, MA, USA: Addison-Wesley, 2005. (Citado nas páginas 11 e 16.)

Tao, Y.; Ozsu, M. T. Mining data streams with periodically changing distributions. In: *Proceeding of the 18th ACM conference on Information and knowledge management*, New York, NY, USA: ACM, 2009, p. 887–896. (Citado nas páginas 27, 63 e 181.)

Therneau, T.; Atkinson, B.; Ripley, B. *rpart: Recursive partitioning.* R package version 3.1-52, 2012.
Disponível em: `http://CRAN.R-project.org/package=rpart` (Citado na página 81.)

Thrun, S.; Mitchell, T. M. Learning one more thing. In: *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 1995, p. 1217–1223. (Citado na página 48.)

Thrun, S.; Pratt, L., eds. *Learning to learn.* Norwell, MA, USA: Kluwer Academic Publishers, 1998. (Citado nas páginas 39, 40 e 41.)

Todorovski, L.; Blockeel, H.; Džeroski, S. Ranking with predictive clustering trees. In: *Proceedings of the 13th European Conference on Machine Learning*, London, UK, UK: Springer-Verlag, 2002, p. 444–455. (Citado na página 138.)

Todorovski, L.; Brazdil, P.; Soares, C. Report on the experiments with feature selection in meta-level learning. In: *Proceedings of the 4th European Conference on Principles on Data Mining and Knowledge Discovery, Workshop on Data Mining, Decision Support, Meta-learning and ILP*, 2000, p. 27–39. (Citado nas páginas 67, 86, 98 e 187.)

Todorovski, L.; Džeroski, S. Experiments in meta-level learning with ilp. In: Żytkow, J.; Rauch, J., eds. *Proceedings of the 3rd European conference on Principles and Practice of Knowledge Discovery in Databases*, Springer Berlin Heidelberg, 1999, p. 98–106. (Citado na página 67.)

Tomek, I. An experiment with the edited nearest-neighbor rule. *Systems, Man and Cybernetics, IEEE Transactions on*, v. SMC-6, n. 6, p. 448–452, 1976. (Citado na página 124.)

TSYMBAL, A. *The problem of concept drift: definitions and related work.* Relatório Técnico TCD-CS-2004-15, School of Computer Science and Statistics - Trinity College Dublin, 2004. (Citado nas páginas 2, 23 e 47.)

TUV, E.; BORISOV, A.; RUNGER, G.; TORKKOLA, K. Feature selection with ensembles, artificial variables, and redundancy elimination. *Journal of Machine Learning Research*, v. 10, p. 1341–1366, 2009. (Citado nas páginas 99 e 188.)

UTGOFF, P. E.; BERKMAN, N. C.; CLOUSE, J. A. Decision tree induction based on efficient tree restructuring. *Machine Learning*, v. 29, n. 1, p. 5–44, 1997. (Citado na página 33.)

UYSAL, I.; GÜVENIR, H. A. An overview of regression techniques for knowledge discovery. *The Knowledge Engineering Review*, v. 14, p. 319–340, 1999. (Citado na página 14.)

VALLIM, R. M.; FILHO, J. A. A.; MELLO, R. F.; CARVALHO, A. C. Online behavior change detection in computer games. *Expert Systems with Applications*, v. 40, n. 16, p. 6258–6265, 2013. (Citado na página 3.)

VANSCHOREN, J. *Understanding Machine Learning Performance with Experiment Databases.* Tese de Doutoramento, Informatics Section, Department of Computer Science, Faculty of Engineering Science, 2010. (Citado nas páginas 66, 97, 179 e 181.)

VILALTA, R.; DRISSI, Y. A perspective view and survey of meta-learning. *Artificial Intelligent Review*, v. 18, n. 2, p. 77–95, 2002. (Citado nas páginas 2 e 40.)

VILALTA, R.; GIRAUD-CARRIER, C.; ; BRAZDIL, P. *Data mining and knowledge. discovery handbook: A complete guide for practitioners and researchers*, cap. Meta-Learning: Concepts and Techniques Kluwer Academic Publishers, p. 1–17, 2005. (Citado na página 44.)

VILALTA, R.; GIRAUD-CARRIER, C. G.; BRAZDIL, P.; SOARES, C. Using meta-learning to support data mining. *International Journal of Computer Science & Applications*, v. 1, n. 1, p. 31–45, 2004. (Citado nas páginas 4 e 42.)

ŽLIOBAITE, I. Controlled permutations for testing adaptive classifiers. In: *Proceedings of the 14th International Conference on Discovery Science*, Berlin, Heidelberg: Springer-Verlag, 2011, p. 365–379. (Citado na página 78.)

ŽLIOBAITE, I.; BIFET, A.; GABER, M.; GABRYS, B.; GAMA, J.; MINKU, L.; MUSIAL, K. Next challenges for adaptive learning systems. *SIGKDD Explorations*, v. 14, n. 1, p. 48–55, 2012. (Citado nas páginas 19, 37, 80 e 153.)

WANG, H.; FAN, W.; YU, P. S.; HAN, J. Mining concept-drifting data streams using ensemble classifiers. In: *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA: ACM, 2003, p. 226–235. (Citado nas páginas xv, 31 e 32.)

WANG, X.; SMITH-MILES, K.; HYNDMAN, R. Rule induction for forecasting method selection: Meta-learning the characteristics of univariate time series. *Neurocomputing*, v. 72, n. 10-12, p. 2581–2594, 2009. (Citado nas páginas 44, 48, 59, 61, 62, 149 e 180.)

WEISS, S.; INDURKHYA, N.; ZHANG, T.; DAMERAU, F. *Text mining: Predictive methods for analyzing unstructured information*. Springer, 2004. (Citado na página 47.)

WIDMER, G. Tracking context changes through meta-learning. *Machine Learning*, v. 27, n. 3, p. 259–286, 1997. (Citado nas páginas 47, 48, 49, 76 e 148.)

WIDMER, G.; KUBAT, M. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, v. 23, n. 1, p. 69–101, 1996. (Citado nas páginas 5, 22, 24, 25 e 81.)

WITTEN, I. H.; FRANK, E. *Data mining: Practical machine learning tools and techniques*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2005. (Citado nas páginas 12, 14, 33, 61, 80 e 91.)

WOLPERT, D. H. Stacked generalization. *Neural Networks*, v. 5, n. 2, p. 241–259, 1992. (Citado na página 31.)

WOLPERT, D. H. The lack of a priori distinctions between learning algorithms. *Neural Computation*, v. 8, p. 1341–1390, 1996. (Citado nas páginas 1, 16, 39 e 147.)

WU, X. *Knowledge acquisition from databases*. Tutorial Monographs in Artificial Intelligence. Ablex Publishing Corp., 1995. (Citado na página 123.)

ZHU, J.; ROSSET, S.; HASTIE, T.; TIBSHIRANI, R. 1-norm support vector machines. In: THRUN, S.; SAUL, L.; SCHÖLKOPF, B., eds. *Advances in Neural Information Processing Systems 16*, Cambridge, MA: MIT Press, 2003.
Disponível em: `http://books.nips.cc/papers/files/nips16/NIPS2003_AA07.pdf`
(Citado na página 188.)

ZHU, X.; WU, X. Class noise vs. attribute noise: a quantitative study of their impacts. *Artificial Intelligence Review*, v. 22, n. 3, p. 177–210, 2004. (Citado na página 123.)

Apêndice A

Medidas de Caracterização

Neste apêndice, é descrito um conjunto de medidas que podem ser utilizadas para a extração de características de problemas de fluxos contínuos de dados. Cada característica é calculada para um conjunto de exemplos do nível base seguindo o mesmo esquema apresentado no Capítulo 4, ou seja, para exemplos dos conjuntos de treinamento, horizonte e seleção. Além dos valores dos atributos preditivos e do atributo alvo, algumas medidas podem ser calculadas para as predições dos modelos. Assim, o termo variáveis é usado no lugar do termo atributos. O tipo da variável também deve ser considerado na geração dos meta-dados. Para facilitar a extração das características, as variáveis são separadas em numéricas e nominais, pois algumas medidas são apropriadas apenas um dos tipos ou para a interação entre variáveis do mesmo tipo ou de tipos diferentes. Se o problema tratado no nível base for de regressão, medidas para variáveis numéricas também podem ser aplicadas para os valores do atributo alvo e para as predições dos modelos. Caso o problema seja de classificação, o mesmo se aplica com medidas para variáveis nominais.

Sejam Z e V duas variáveis numéricas, o valor dessas variáveis para o k-ésimo exemplo do conjunto de dados a ser caracterizado é denotado por z_k e v_k, respectivamente. A média, desvio padrão e variância da variável numérica V são denotados por μ_Z, σ_Z e σ_Z^2, respectivamente, e semelhantemente para V, μ_V, σ_V e σ_V^2.

Sejam A e B duas variáveis nominais, a distribuição conjunta dessas variáveis é representada em uma matriz de contingência com I linhas, que representa o número de valores distintos do atributo A, e J colunas, que representa o número de valores distintos do atributo B. Cada valor é denotado como π_{ij}, que é a distribuição de probabilidade. As distribuições marginais de A e B são dadas pelo total das linhas e colunas e são denotadas como π_{i+} e π_{+j}, respectivamente, em que $\pi_{i+} = p(A = a_i) = \sum_j \pi_{ij}$ e $\pi_{+j} = p(B = b_j) = \sum_i \pi_{ij}$. A distribuição condicional de B dada por A é denotada por $\pi_{j|i}$, em que $\pi_{j|i} = P(B = b_j|A = a_i)$.

Para as medidas que caracterizam o comportamento de um modelo, há a necessidade de distinção entre o atributo alvo e as predições dos modelos. Nesse caso, Y representa o atributo alvo e y_k denota o valor do atributo alvo para o k-ésimo exemplo do conjunto

de dados. Da mesma maneira, $\hat{Y}$ representa a predição para o atributo Y e $\hat{y}_k^u$ denota a predição do u-ésimo modelo, de um total de U modelos, para o k-ésimo exemplo.

A seguir são apresentadas possíveis medidas que podem ser aplicadas para caracterização de fluxos de dados. Essas medidas estão dividas em: 1. descrição de variáveis; 2. relação entre variáveis; 3. comportamento de um modelo e relações entre modelos; 4. descrição dos meta-dados. Por último, são apresentadas as medidas e os dados para os quais elas podem ser aplicadas considerando um problema de regressão no nível base e de classificação no nível meta.

A.1 Descrição de variáveis

- **Valores brutos dos atributos -** $Raw(\mathbf{x_k})$ - usa os valores brutos dos atributos preditivos de um exemplo k, $\mathbf{x_k}$, para caracterizá-lo no nível meta (Gama e Kosina, 2011; Duch et al., 2011).

- **Média aritmética -** μ_Z - mede a tendência central ou o valor típico de uma variável numérica Z.

- **α-média aparada -** μ_{Z^α} - calcula a média aritmética de uma variável numérica Z excluindo uma fração dos maiores e dos menores valores, ou seja, calcula a média sem a interferência dos valores extremos. (Soares, 2004; Ali e Smith-Miles, 2006).

$$\mu_{Z^\alpha} = \frac{1}{\omega_b - 2\alpha} \sum_{q=\lfloor \omega_b \alpha \rfloor + 1}^{\omega_b - \lfloor \omega_b \alpha \rfloor} o(Z)_q \tag{A.1}$$

  em que $o(Z)_q$ é o q-ésimo valor ordenado ascendente da variável Z e α é a fração dos maiores e menores valores que serão removidos, tipicamente $0.025 \leq \alpha \leq 0.1$.

- **Desvio padrão -** σ_Z - mede a dispersão de uma variável numérica Z em relação à sua média μ_Z (Castiello et al., 2005).

- **Mediana-** $M(Z)$ - mede o valor de separação entre a metade superior e a metade inferior de uma variável numérica Z (Ali e Smith-Miles, 2006). Também é uma medida de tendência central.

- **Intervalo interquartil -** $IQR(Z)$ - similarmente ao desvio padrão, o intervalo interquartil mede a dispersão de uma variável numérica Z. Essa medida é computada como a diferença entre o 3^o e o 1^o quartil $(Q_3 - Q_1)$. Para uma distribuição normal, $\frac{IQR}{\sigma} = 1.3$ (Ali e Smith-Miles, 2006).

- **Intervalo -** $Inter(Z)$ - mede a diferença entre os valores máximo e mínimo de uma variável numérica Z (Ali e Smith-Miles, 2006).

- **Máximo -** $max(Z)$ - obtêm o maior valor de uma variável numérica Z.

- **Mínimo -** $min(Z)$ - obtêm o menor valor de uma variável numérica Z.

- ***Outliers* -** $Outl(Z)$ - mede a possibilidade de existência de *outliers* em uma variável numérica Z (Soares, 2004).

$$Outl(Z^{\alpha}) = \frac{\sigma_{Z^{\alpha}}}{\sigma_Z} \tag{A.2}$$

- **Assimetria -** $\gamma(Z)$ - caracteriza o grau de assimetria dos valores de uma variável numérica Z em torno do valor médio (Michie et al., 1994). Também conhecida como o terceiro momento da distribuição de probabilidade. Valores negativos dessa medida indicam que a cauda esquerda é mais alongada do que a cauda direita, enquanto valores positivos significam que a cauda direita é mais alongada que a esquerda. A assimetria de uma distribuição normal é zero, porque ela é simétrica em torno da média.

$$\gamma(Z) = \frac{(Z - \mu_Z)^3}{(\omega_b - 1)\sigma_Z^3} \tag{A.3}$$

- **Curtose -** $\beta(Z)$ - é uma medida que caracteriza se a distribuição de uma variável numérica Z possui muitos picos ou é plana em relação a uma distribuição normal que tem o mesmo desvio padrão (Michie et al., 1994). Também é conhecida como quarto momento da distribuição de probabilidade. Valores positivos de curtose tendem a ter um pico mais agudo próximo à média e caudas mais pesadas. Semelhantemente, valores negativos tendem a ter um topo mais plano próximo da média e caudas mais finas. A distribuição normal tem curtose igual a 3.

$$\beta(Z) = \frac{(Z - \mu_Z)^4}{(\omega_b - 1)\sigma_Z^4} \tag{A.4}$$

- **Coeficiente de variação** - $VarCoef(Z)$ - mede o coeficiente de variação de uma variável numérica Z (Soares, 2004; Vanschoren, 2010).

$$VarCoef(Z) = \frac{\sigma_Z}{\mu_Z} \tag{A.5}$$

- **Entropia -** $H(A)$ - é a medida de incerteza ou aleatoriedade de uma variável nominal A (Michie et al., 1994). O valor de entropia é grande quando todos os l valores

distintos de A possuem igual probabilidade. Ao contrário, quando todos os valores são iguais, a entropia é zero e, portanto, a variável não contém nenhuma informação.

$$H(A) = -\sum_{i=1}^{I} \pi_{i+} log_2(\pi_{i+}) \tag{A.6}$$

- **Frequência de valores -** $Freq(A_i)$ - calcula a frequência de cada valor distinto de uma variável nominal A com I valores distintos.

$$Freq(A_i) = \pi_{i+} \tag{A.7}$$

- **Heterogeneidade -** $Het(Z)$ - mede a heterogeneidade de uma variável numérica Z, o que (Pearson, 2011) chama de extensão da entropia normalizada de Shannon para variáveis numéricas. Essa estratégia divide o intervalo total de uma variável numérica Z em n intervalos de tamanhos iguais. Em seguida, o número de valores que estão em cada intervalo é dividido pelo número total de valores (Pearson, 2011).

- **Teste de correlação serial -** $Q_h(Z)$ - mede o grau de similaridade entre os valores de uma variável numérica Z para vários intervalos. A correlação serial ocorre quando valores antigos de uma variável interferem nos seus valores futuros. Neste estudo, a estatística de Pox-Pierce foi usada para estimar a correlação serial para um intervalo máximo h, $h < n$, em que n é o número de valores da variável Z. (Wang et al., 2009).

$$Q_h(Z) = n \sum_{q=1}^{h} r_q^2 \tag{A.8}$$

$$r_q = \frac{\sum_{k=1}^{n-k}(Z_k - \mu_Z)(Z_{k+q} - \mu_Z)}{\sum_{k=1}^{n}(Z_k - \mu_Z)^2} \tag{A.9}$$

em que r_q é a autocorrelação para o deslocamento (lag) q.

- **Taxa de pontos de inversão -** $rTP(Z)$ - captura o comportamento oscilatório em uma variável numérica Z. Um valor é um ponto de inversão se $z_{k-1} < z_k > z_{k+1}$ ou $z_{k-1} > z_k < z_{k+1}$. Portanto, z_k um valor é um ponto de inversão se ele é um máximo ou mínimo local de acordo com os seus vizinhos mais próximos (Prudêncio e Ludermir, 2004; Lemke e Gabrys, 2010).

$$rTP(Z) = \frac{|TP|}{n} \tag{A.10}$$

em que $|TP|$ é o número de pontos de inversão da variável Z e n é o número de valores da variável Z.

A.2 Relação entre variáveis

- **Coeficiente de correlação -** ρ_{ZV} - mede a força da relação linear entre duas variáveis numéricas Z e V (Michie et al., 1994).

$$\rho_{ZV} = \frac{\sigma_{ZV}}{\sqrt{\sigma_Z^2 \sigma_V^2}} \tag{A.11}$$

- **Coeficiente de concentração -** τ_{AB} - mede a redução proporcional da probabilidade de uma predição incorreta para uma variável nominal B, com J valores distintos, usando uma variável nominal A, com I valores distintos (Kalousis, 2002). O coeficiente de concentração não é simétrico, ou seja, $\tau_{AB} \neq \tau_{BA}$. Essa medida é conhecida também como τ de Goodman e Kruskal.

$$\tau_{AB} = \frac{\sum_{i=1}^{I} \sum_{j=1}^{J} \frac{\pi_{ij}^2}{\pi_{i+}} - \sum_{j=1}^{J} \pi_{+j}^2}{1 - \sum_{j=1}^{J} \pi_{+j}^2} \tag{A.12}$$

- **p-valor da distribuição F -** $p_{val_{AZ}}$ - mede a relação entre uma variável nominal A e uma variável numérica Z usando a distribuição F da mesma maneira como na análise de variância (ANOVA). Ao invés de realizar um teste de significância, apenas o p-valor é reportado como uma medida de relação. A ANOVA examina se uma variável nominal independente afeta uma variável numérica dependente, mas não o oposto. Essa análise consiste em verificar se há diferenças entre as médias de cada grupo I definido em Z por A. A taxa das variâncias entre grupos e a variância dentro dos grupos segue uma distribuição F e o p-valor dessa distribuição fornece a probabilidade de observar um valor específico, sob a suposição de que as médias dos grupos são iguais (Vanschoren, 2010; Kalousis, 2002). Embora o p-valor não seja uma medida de relação, ele fornece uma indicação se a variável A afeta Z e também o nível dessa relação.

- **Distância entre conjuntos de dados -** $Dist(S_a, S_b)$ - para um fluxo de dados, essa medida é aplicada para mensurar a distância entre as distribuições de dados pertencentes a duas janelas distintas: uma janela que contém dados mais antigos, denominada referência (S_a), e uma janela janela que contém os exemplos mais recentes, denominada corrente (S_b). Diferentes medidas já foram usadas na literatura para esse propósito (Tao e Ozsu, 2009), como a entropia relativa ou Kullback-Leibler (Dasu et al., 2009; Sebastião et al., 2009) e a discrepância relativa (Kifer et al., 2004).

- **Ganho de dispersão -** $DispGain(E)$ - mede a melhora que cada atributo preditivo proporciona em uma árvore induzida pelo algoritmo CART (Soares, 2004).

- **Coeficiente de determinação -** R^2 - mede o desempenho de um modelo de regressão linear para os dados (Michie et al., 1994). Essa medida fornece um indicativo da linearidade do problema.

A.3 Comportamento de um modelo e relações entre modelos

- **Taxa do desempenho preditivo -** rD^u - mede a taxa do desempenho preditivo obtido pelo modelo u como a fração entre o seu desempenho e o desempenho preditivo médio de todos os U modelos.

$$rD^u = \frac{D(\hat{Y}^u, Y)}{\sum_u^U D(\hat{Y}^u, Y)} \tag{A.13}$$

  em que D representa uma medida de desempenho de um modelo, como o MSE.

- **Ranking -** $rank^u$ - calcula o ranking de cada modelo de acordo com o seu desempenho em comparação com os demais modelos.

- **Taxa do desvio padrão do erro quadrático -** $r\sigma_{SE^u}$ - mede a taxa de variação dos erros quadráticos de cada modelo.

$$r\sigma_{SE^u} = \frac{\sigma_{SE^u}}{\sum_u \sigma_{SE^u}} \tag{A.14}$$

  em que SE^u representa o erro quadrático do modelo u.

- **Diversidade I** $DivI$ - mede a diversidade dos modelos com base nas medidas propostas por Brown et al. (2005) e Lemke e Gabrys (2010). A medida $DivI$ é dada pela média dos coeficientes de variação dos coeficientes de correlação dos erros quadráticos dos modelos.

$$DivI = \frac{2}{U(U-1)} \sum_{i=1}^{U-1} \sum_{j=i+1}^{U} \frac{\sigma_{\rho_{SE^i SE^j}}}{\mu_{\rho_{SE^i SE^j}}} \tag{A.15}$$

  em que $\rho_{SE^i SE^j}$ é a correlação entre o erro quadrático do modelo i (SE^i) e o erro quadrático do modelo j (SE^j) e U é o número de modelos.

- **Diversidade II -** $DivII$ - mede a diversidade entre os modelos por meio da diferença entre os termos viés/variância e covariância entre os modelos. Essa medida

foi proposta por Brown et al. (2005) e usada em Lemke e Gabrys (2010).

$$DivII = \frac{1}{U}\sum_{u}(\hat{Y}^u - Y)^2 - \kappa\frac{1}{U}\sum_{u}(\hat{Y}^u - \mu_{\hat{Y}})^2 \tag{A.16}$$

em que $\mu_{\hat{Y}}$ é a média das predições de todos os U modelos e κ é um coeficiente escalar entre $[0, 1]$, que controla a extensão do impacto da covariância (segundo termo da equação) no erro.

A.4 Descrição dos meta-dados

Para a extração de características dos meta-dados, considere que os valores reais do meta-atributo alvo são denotados por C e as predições do meta-modelo são denotadas por $\hat{C}$. Para cada meta-classe l, que representa um algoritmo ou modelo, $l = \{1, \ldots, U\}$, C^l denota todos os valores reais do meta-atributo alvo da classe l e $\hat{C}^l$ denota as predições realizadas pelo meta-modelo para C^l. As medidas são sempre calculadas para os meta-exemplos da janela deslizante, conforme descrito na Seção 4.3.3.

- **Taxa de erro para cada meta-classe -** *taxa de erro*(l) - mede o desempenho preditivo do meta-modelo para cada classe do conjunto de meta-dados, que representa um modelo ou algoritmo.

$$\text{taxa de erro}(l) = L(\hat{C}^l, C^l) \tag{A.17}$$

em que L é a função de perda 0-1.

- **Indicativo do resultado da predição meta -** $IndPred(\hat{c}_i, c_i)$ - essa medida retorna 0 se o meta-modelo classificou o último meta-exemplo i corretamente, ou 1 caso contrário (Gama e Kosina, 2011).

- **Frequência das meta-classes -** $Freq(C^l)$ - calcula a frequência de cada meta-classe.

A.5 Medidas e dados

Nesta seção são apresentadas as possibilidades de aplicação das medidas descritas anteriormente para o caso em que o problema do nível base é de regressão e do nível meta é de classificação, como neste trabalho. No entanto, as medidas que não utilizam os valores do atributo alvo e das predições podem também ser usadas para problemas de classificação. Outras medidas para classificação podem ser encontradas em Kalousis

(2002) e Soares (2004). Essa descrição das medidas de caracterização e os dados para os quais elas se aplicam não têm a pretensão de ser completa, mas fornece possibilidades para a geração dos meta-dados com base no método proposto neste trabalho e em outros trabalhos relacionados.

Na Tabela A.1 são mostradas, da coluna da esquerda para a direita, as medidas, o que elas caracterizam (variáveis, relações entre variáveis, modelos, etc.), os tipos de variáveis e os dados para os quais elas se aplicam. A mesma notação usada nas Figuras 4.5 e 4.7 e nas suas respectivas seções será usada nesta seção. As medidas apresentadas na Tabela A.1 se aplicam à caracterização de lotes de exemplos. Para o caso da seleção unitária, as mesmas medidas podem ser usadas, mas não podem ser aplicadas para $\mathbf{X}_{\mathbf{sel.}}$ e $\hat{\mathbf{y}}_{\mathbf{sel.}}$ Por outro lado, os valores dos atributos de $\mathbf{X}_{\mathbf{sel.}}$ podem ser usados como meta-atributos no caso da seleção unitária, mas não se aplica para lotes de exemplos.

Tabela A.1: Medidas para caracterização dos dados e os dados para os quais elas podem ser aplicadas.

Medidas	**Caracteriza o quê?**	**Tipo da variável**	**Dados**
Média aritmética	Variáveis	Numérica	$\mathbf{X}_{\odot}$, $\hat{\mathbf{y}}_{\odot}$, $\mathbf{y}_{\mathbf{trein.}}$
α-média aparada	Variáveis	Numérica	$\mathbf{X}_{\odot}$, $\hat{\mathbf{y}}_{\odot}$, $\mathbf{y}_{\mathbf{trein.}}$
Desvio padrão	Variáveis	Numérica	$\mathbf{X}_{\odot}$, $\hat{\mathbf{y}}_{\odot}$, $\mathbf{y}_{\mathbf{trein.}}$
Mediana	Variáveis	Numérica	$\mathbf{X}_{\odot}$, $\hat{\mathbf{y}}_{\odot}$, $\mathbf{y}_{\mathbf{trein.}}$
Intervalo interquartil	Variáveis	Numérica	$\mathbf{X}_{\odot}$, $\hat{\mathbf{y}}_{\odot}$, $\mathbf{y}_{\mathbf{trein.}}$
Intervalo	Variáveis	Numérica	$\mathbf{X}_{\odot}$, $\hat{\mathbf{y}}_{\odot}$, $\mathbf{y}_{\mathbf{trein.}}$
Máximo	Variáveis	Numérica	$\mathbf{X}_{\odot}$, $\hat{\mathbf{y}}_{\odot}$, $\mathbf{y}_{\mathbf{trein.}}$
Mínimo	Variáveis	Numérica	$\mathbf{X}_{\odot}$, $\hat{\mathbf{y}}_{\odot}$, $\mathbf{y}_{\mathbf{trein.}}$
Outliers	Variáveis	Numérica	$\mathbf{X}_{\odot}$, $\hat{\mathbf{y}}_{\odot}$, $\mathbf{y}_{\mathbf{trein.}}$
Assimietria	Variáveis	Numérica	$\mathbf{X}_{\odot}$, $\hat{\mathbf{y}}_{\odot}$, $\mathbf{y}_{\mathbf{trein.}}$
Curtose	Variáveis	Numérica	$\mathbf{X}_{\odot}$, $\hat{\mathbf{y}}_{\odot}$, $\mathbf{y}_{\mathbf{trein.}}$
Coeficiente de variação	Variáveis	Numérica	$\mathbf{X}_{\odot}$, $\hat{\mathbf{y}}_{\odot}$, $\mathbf{y}_{\mathbf{trein.}}$
Entropia	Variáveis	Nominal	$\mathbf{X}_{\odot}$
Frequência de valores[1]	Variáveis	Nominal	$\mathbf{X}_{\odot}$
Heterogeneidade[2]	Variáveis	Numérica	$\mathbf{X}_{\odot}$, $\hat{\mathbf{y}}_{\odot}$, $\mathbf{y}_{\mathbf{trein.}}$
Teste de correlação serial	Variáveis	Numérica	$\mathbf{X}_{\odot}$, $\hat{\mathbf{y}}_{\odot}$, $\mathbf{y}_{\mathbf{trein.}}$
Taxa de pontos de inversão	Variáveis	Numérica	$\mathbf{X}_{\odot}$, $\hat{\mathbf{y}}_{\odot}$, $\mathbf{y}_{\mathbf{trein.}}$

[1]Se houver muitos valores simbólicos diferentes, essa medida gera muitas saídas. Ao invés disso, outras informações poderiam ser usadas, tais como o valor nominal mais frequente e sua frequência.

[2] Essa medida gera muitos valores de saída: para cada atributo contínuo são gerados q meta-atributos, em que q é um parâmetro da medida.

Coeficiente de correlação	Relação entre variáveis	Numérica	$\mathbf{y_{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{trein.}}$, $\mathbf{X_{trein.}} \leftrightarrow \mathbf{y_{trein.}}$, $\mathbf{X_{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{trein.}}$, $\mathbf{X_{sel.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{sel.}}$, $\mathbf{X_{horiz.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{horiz.}}$, $\mathbf{X}_{\odot} \leftrightarrow \mathbf{X}_{\odot}$
Coeficiente de concentração	Relação entre variáveis	Nominal	$\mathbf{X}_{\odot} \leftrightarrow \mathbf{X}_{\odot}$
p-valor da distribuição F	Relação entre variáveis	Nominal/ Numérica	$\mathbf{X_{trein.}} \leftrightarrow \mathbf{y_{trein.}}$, $\mathbf{X_{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{trein.}}$, $\mathbf{X_{sel.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{sel.}}$, $\mathbf{X_{horiz.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{horiz.}}$, $\mathbf{X}_{\odot} \leftrightarrow \mathbf{X}_{\odot}$
Distância entre conjuntos de dados	Conj. dados/ Relação entre variáveis	Numérica/ Nominal	$\mathbf{X}_{\bowtie} \leftrightarrow \mathbf{X}_{\bowtie}$
Ganho de dispersão	Modelos	Numérica/ Nominal	$\mathbf{X_{trein.}} \leftrightarrow \mathbf{y_{trein.}}$
Coeficiente de determinação	Modelos	Numérica/ Nominal	$\mathbf{y_{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{trein.}}$
Taxa do desempenho preditivo	Modelos	Numérica	$\mathbf{y_{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{trein.}}$
Ranking	Modelos	-	$\mathbf{y_{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{trein.}}$
Taxa do desvio padrão do erro quadrático	Modelos	-	$\mathbf{y_{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{trein.}}$
Diversidade I[3]	Relação modelos	-	$\mathbf{y_{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{trein.}}$
Diversidade II	Relação modelos	-	$\mathbf{y_{trein.}} \leftrightarrow \hat{\mathbf{y}}_{\mathbf{trein.}}$
Taxa de erro para cada meta-classe	Relação meta-variáveis	Nominal	$\mathbf{c_{trein.}} \leftrightarrow \hat{\mathbf{c}}_{\mathbf{trein.}}$
Indicativo do resultado da predição meta	Relação meta-variáveis	Nominal	$\mathbf{c_{trein.}} \leftrightarrow \hat{\mathbf{c}}_{\mathbf{trein.}}$
Frequência das meta-classes	Meta-variáveis	Nominal	$\mathbf{c_{trein.}}$

[3]Aplicável apenas quando mais do que dois modelos forem usados.

Apêndice B

Seleção de Atributos

Seleção de atributos é a tarefa de escolher, de acordo com algum critério, um subconjunto de atributos preditivos que idealmente é necessário e suficiente para descrever o valor do atributo alvo. A seleção de atributos é uma importante técnica em mineração de dados para redução de dimensionalidade, remoção de atributos irrelevantes, redundantes ou com ruído (Robnik-Šikonja e Kononenko, 2003; Liu et al., 2010). Esse procedimento, geralmente acelera o processo de aprendizado de um algoritmo, facilita a visualização dos dados e o entendimento do modelo e aumenta sua acurácia, pois a maioria dos algoritmos de aprendizado não são desenvolvidos para funcionar adequadamente com centenas ou milhares de atributos (Guyon e Elisseeff, 2003; Robnik-Šikonja e Kononenko, 2003).

Na área de meta-aprendizado, alguns trabalhos têm realizado a seleção de meta-atributos com os objetivos de entender os fatores que afetam o desempenho dos algoritmos e para melhorar o desempenho preditivo dos meta-aprendizes (Todorovski et al., 2000; Kalousis e Hilario, 2001; Kanda, 2012). Para isso, os mesmos métodos de seleção de atributos para problemas de aprendizado no nível base têm sido empregadas também no nível meta. Essencialmente, esses métodos podem ser organizados em três categorias, dependendo de como eles combinam a seleção de atributos com a construção do modelo (Guyon e Elisseeff, 2003): wrappers, filtros ou embarcados.

Os wrappers utilizam o algoritmo de aprendizado investigado para avaliar um subconjunto de atributos (Kohavi e John, 1997). Métodos de amostragem, como a validação cruzada, são usados para estimar o desempenho do algoritmo de aprendizado no conjunto reduzido de atributos. Wrappers são frequentemente criticados porque normalmente possuem alto custo computacional. Estratégias de buscas de dois tipos podem acelerar esse processo: seleção para a frente (do inglês, *forward selection*) e eliminação para trás (do inglês, *backward elimination*). Na seleção para a frente, os atributos são progressivamente adicionados a um subconjunto, que inicialmente está vazio, enquanto que na eliminação para trás inicia-se a busca com o conjunto completo de atributos e progressivamente elimina-se aqueles que são menos promissores (Guyon e Elisseeff, 2003).

Os filtros avaliam a relevância dos atributos usando as propriedades intrínsecas dos

dados, como aquelas baseadas na teoria da informação, distância, dependência e correlação (Kalousis e Hilario, 2001; Saeys et al., 2007). A maioria dos métodos de filtro calculam a relevância de cada atributo baseado no peso relativo de cada um. Depois disso, um subconjunto de atributos com maior peso é apresentado a um algoritmo de aprendizado. As vantagens dos métodos de filtro são que eles são facilmente escaláveis para problemas de grande dimensionalidade, são simples, possuem baixo custo computacional e são independentes do algoritmo de aprendizado. Uma crítica comum às técnicas de filtro é que elas podem selecionar um subconjunto redundante de atributos (Hall, 2000). Porém, a redução de ruído dos dados e, consequentemente, uma melhor separação entre as classes, pode ser obtida adicionando atributos que são presumidamente redundantes, como é mostrado em (Guyon e Elisseeff, 2003).

Os métodos embarcados realizam a seleção de atributos como parte do processo de indução dos modelos e, portanto, têm uma grande dependência do algoritmo de aprendizado. Algoritmos como o C4.5 (Quinlan, 1993) e o LARS (Zhu et al., 2003) possuem esse mecanismo interno de seleção de atributos e a importância de cada atributo é obtido com base na sua utilidade para otimizar a função objetivo no processo de treinamento (Liu et al., 2010). Na construção de uma árvore de decisão, por exemplo, a redução da impureza devido à divisão em um atributo indica a importância relativa desse atributo para o modelo. Em ensembles, como RF, calcula-se a média dessa métrica sobre um conjunto de modelos. Nesse caso, a importância relativa dos atributos automaticamente incorpora a interação entre eles, o que não acontece com métodos que assumem a condição de independência dos atributos (Tuv et al., 2009).

Os métodos wrapper e embarcado são conhecidos por selecionarem atributos que resultam em melhor desempenho preditivo. Por outro lado, métodos de filtro são computacionalmente mais rápidos do que wrappers e são independentes do algoritmo de aprendizado, ao contrário dos wrappers e embarcados. O método de filtro ReliefF, por exemplo, é eficiente e não assume a independência condicional dos atributos e, portanto, é mais apropriado para problemas em que há uma forte dependência entre os atributos (Robnik-Šikonja e Kononenko, 2003).

O algoritmo Relief (Kononenko, 1994) (Algoritmo 1), assim como a maioria dos métodos de filtro, atribui pesos aos atributos. Esse peso mede a relevância de cada atributo de acordo com a distância de um exemplo para seus vizinhos mais próximos. O algoritmo ReliefF seleciona aleatoriamente um exemplo R_i (linha 3) e busca pelos k vizinhos mais próximos da mesma classe desse exemplo, chamados de *nearest hits* H_j (linha 4), e também os k vizinhos mais próximos de classes diferentes, chamados de *nearest misses* $M_j(C)$ (linhas 5, 6 e 7). Com esses exemplos, a estimativa $W[A]$ é atualizada para todos os atributos A, dependendo dos seus valores para R_i, H_j e $M_j(C)$ (linhas 8, 9 e 10). A contribuição de cada classe dos *nearest misses* é dividida proporcionalmente, de acordo com a probabilidade de cada classe $P(C)$, estimada a partir do conjunto de treinamento.

O processo é repetido m vezes. O termo $d(i,j)$ no Algoritmo 1 (linha 9) leva em consideração a distância entre os exemplos R_i e H_j. A razão é que os vizinhos mais próximos devem ter uma maior influência. Por isso a influência de um exemplo H_j diminui com a distância para o exemplo R_i:

$$d(i,j) = \frac{d_1(i,j)}{\sum_{l=1}^{k} d_1(i,l)} \tag{B.1}$$

e

$$d_1(i,j) = e^{-\left(\frac{rank(R_i,H_j)}{\sigma}\right)^2} \tag{B.2}$$

em que $rank(R_i, H_j)$ é o ranking do exemplo H_j em uma sequência de exemplos ordenados pela distância de R_i e σ é um parâmetro que controla a influência da distância. A influência dos vizinhos mais próximos deve ser normalizada para cada exemplo R_i selecionado aleatoriamente, a fim de se obter uma interpretação probabilística dos resultados. A implementação fornecida por Robnik-Šikonja e Kononenko (2003) para a linguagem de programação R (pacote CORElearn) foi utilizada neste trabalho com os valores padrão para os parâmetros ($k = 70$, $\sigma = 20$ e m é igual ao tamanho do conjunto de dados), considerando que a influência dos vizinhos diminui exponencialmente com o aumento da distância.

Entrada: para cada exemplo de treinamento um vetor de valores de atributos e o valor da classe
Saída: o vetor W de estimativas da qualidade dos atributos

1 Atribuir todos os pesos $W[A] = 0.0$;
2 **para** $i = 1$ **até** m **faça**
3 | seleciona aleatoriamente um exemplo R_i;
4 | seleciona os k *nearest hits* H_j;
5 | **para cada** *classe* $C \neq classe(R_i)$ **faça**
6 | | seleciona os k *nearest misses* $M_j(C)$;
7 | **fim**
8 **fim**
9 **para** $A = 1$ **até** a **faça**
10 | $W[A] = W[A] - \frac{1}{m}\sum_{j=1}^{k} dif(A, R_i, H_j) \cdot d(R_i, H_j) +$
| $\frac{1}{m}\sum_{C \neq classe(R_i)} \frac{P(C)}{1-P(classe(R_i))} \sum_{j=1}^{k} dif(A, R_i, M_j(C)) \cdot d(R_i, M_j(C))$;
11 **fim**

Algoritmo 1: Algoritmo ReliefF segundo (Robnik-Šikonja e Kononenko, 2003) levando em consideração a distância dos vizinhos mais próximos na atualização dos pesos.